

https://dspace.org/

Rationalization of Deep Neural Networks in Credit Scoring

Dastile, Xolani Collen

2023-07
University of the Witwatersrand, Johannesburg
https://hdl.handle.net/10539/42298
Celik, Turgay. (2023). Rationalization of Deep Neural Networks in Credit Scoring. [PhD thesis, University of the Witwatersrand, Johannesburg]. https://hdl.handle.net/10539/42298

Downloaded from a DSpace repository. https://dspace.org/

UNIVERSITY OF THE WITWATERSRAND



Rationalization of Deep Neural Networks in Credit Scoring

by

Xolani Dastile

Supervised by

Professor Turgay Celik

A thesis submitted in fulfillment for the
degree of Doctor of Philosophy

in the
Science Faculty
School of Computer Science and Applied Mathematics

July 2023

Declaration of Authorship

I, XOLANI DASTILE, declare that this thesis titled, 'RATIONALIZATION OF DEEP NEURAL NETWORKS IN CREDIT SCORING' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: 

Date: 08/07/2023

"The aim of science is to seek the simplest explanations of complex facts."

Alfred North Whitehead

UNIVERSITY OF THE WITWATERSRAND

Abstract

Science Faculty
School of Computer Science and Applied Mathematics

Doctor of Philosophy

by Xolani Dastile

Machine learning and deep learning, which are subfields of artificial intelligence, are undoubtedly pervasive and ubiquitous technologies of the 21st century. This is attributed to the enhanced processing power of computers, the exponential growth of datasets, and the ability to store the increasing datasets. Many companies are now starting to view their data as an asset, whereas previously, they viewed it as a by-product of business processes. In particular, banks have started to harness the power of deep learning techniques in their day-to-day operations; for example, chatbots that handle questions and answers about different products can be found on banks' websites. One area that is key in the banking sector is the credit risk department. Credit risk is the risk of lending money to applicants and is measured using credit scoring techniques that profile applicants according to their risk. Deep learning techniques have the potential to identify and separate applicants based on their lending risk profiles. Nevertheless, a limitation arises when employing deep learning techniques in credit risk, stemming from the fact that these techniques lack the ability to provide explanations for their decisions or predictions. Hence, deep learning techniques are coined as non-transparent models. This thesis focuses on tackling the lack of transparency inherent in deep learning and machine learning techniques to render them suitable for adoption within the banking sector. Different statistical, classic machine learning, and deep learning models' performances were compared qualitatively and quantitatively. The results showed that deep learning techniques outperform traditional machine learning models and statistical models. The predictions from deep learning techniques were explained using state-of-the-art explanation techniques. A novel model-agnostic explanation technique was also devised, and credit-scoring experts assessed its validity. This thesis has shown that different explanation techniques can be relied upon to explain predictions from deep learning and machine learning techniques.

Acknowledgements

To God be the glory, I thank God for strength and hope. I am deeply indebted to my supervisor, Professor Turgay Celik, for his invaluable guidance and patience. Words cannot express my gratitude for how Prof has ensured that I complete this work. I also could not have undertaken this journey without funding from Banking Sector Education and Training Authority (BANK SETA). I am also grateful to Doctor Tjedza Ndlovu for taking the time to proofread my thesis and providing valuable feedback.

I would also like to thank my mother, Monica Dastile, for constantly checking on me and motivating me to complete this work. I would like to extend my thanks to my mother-in-law, Mapule Machaua, and father-in-law, Ben Machaua, especially in the early stages of this work, they were looking after Oyama while I was busy with my studies.

Finally and most importantly, I would be remiss in not mentioning my wife, Princess Dastile, and kids, Oyama Dastile and Amyoli Dastile. Their belief in me has kept my spirits and motivation high during this process.

Contents

Declaration of Authorship i

Abstract iii

Acknowledgements iv

List of Figures vii

1 Introduction 1
- 1.1 Literature review . . . 2
- 1.2 Application Scorecard . . . 5
  - 1.2.1 Default Definition . . . 5
  - 1.2.2 Scoring . . . 6
  - 1.2.3 Research Questions . . . 8
  - 1.2.4 Hypotheses . . . 8
  - 1.2.5 Aims and Research Objectives . . . 8
  - 1.2.6 Contributions . . . 9
  - 1.2.7 Thesis by publication layout . . . 9

2 Thesis Publications 10
- 2.1 Journal Article I . . . 12
- 2.2 Journal Article II . . . 13
- 2.3 Journal Article III . . . 14
- 2.4 Journal Article IV . . . 15

3 Discussion and Conclusion 16
- 3.1 Discussion . . . 16
  - 3.1.1 Article I . . . 16
  - 3.1.2 Article II . . . 17
  - 3.1.3 Article III . . . 17
  - 3.1.4 Article IV . . . 18
- 3.2 Conclusion . . . 18

Bibliography **21**

List of Figures

1.1 Default definition [1]. 6
1.2 Stabilization of bad rates [1]. 6

List of Research Papers

Authorś publications as submitted for PhD by publication:

1. X. Dastile, T. Celik and M. Potsane, Statistical and machine learning models in credit scoring: A systematic literature survey, *Applied Soft Computing*, vol. 91, pp. 106263, 2020.

2. X. Dastile and T. Celik, Making Deep Learning-Based Predictions for Credit Scoring Explainable, *IEEE Access*, vol. 9, pp. 50426-50440, 2021.

3. X. Dastile, T. Celik and H. Vandierendonck, Model-agnostic Counterfactual Explanations in Credit Scoring, *IEEE Access*, vol. 10, pp. 69543-69554, 2022.

4. X. Dastile and T. Celik, Counterfactual Explanations with Multiple Properties in Credit Scoring, *Engineering Applications of Artificial Intelligence (In review)*

Dedicated to my wife

Princess Dastile

and kids
Oyama and Amyoli.

Chapter 1

Introduction

Artificial Intelligence (AI) is a pervasive and ubiquitous technology of the 21st century. The technological revolution that we are witnessing today has been spurred on by previous revolutions. The revolutions were named the First, Second and Third revolutions, and each has significantly impacted societies and economies at large [2]. The First Industrial revolution started around 1765 and was mainly driven by mechanization powered by steam engines [2]. The Second Industrial revolution started in 1870, and the driving force was electricity, and as a result, mass production was realized [2]. The Third Industrial Revolution started in 1969 and focused on information technology powered by the advent of computers [2]. Today, we live in the Fourth Industrial Revolution, the digitalization era, mainly driven by AI. The AI revolution has impacted our day-to-day lives, for example, AI makes recommendations for us regarding which movies to watch on Netflix, friend suggestions on Facebook, and products to buy from Google advertisements. Many companies have grown an interest in adopting AI in their day-to-day operations.

AI is not a new phenomenon; it was first coined in 1956 by John McCarthy during a summer conference at Dartmouth University [3]. The advent of computers in the 1950s played a pivotal role in AI, and surprisingly this is when credit scoring was invented [4]. The early days of credit scoring relied more on credit managers, where credit managers used qualitative methods to either grant or deny loan applications [5]. The reliance on credit managers was not a viable option due to poor judgments (which may have been potentially biased). The credit managers were also not able to process many applications over a few days. This gave rise to the advent of a credit-scoring technique known today as a scorecard.

Credit scoring involves the allocation of scores to applicants. The allocated scores are used to manage the risk of defaulting on a loan [1]. Machine learning and deep learning

techniques have the potential to identify and separate applicants based on their lending risk profiles.

Recent years have witnessed a proliferation of machine learning models in credit scoring [6–10]. Machine learning and deep learning models found in credit scoring literature give better predictive accuracy rates than conventional models (i.e. logistic regression and decision trees) [11–14]. Hence, the improved accuracy rates can help banks curb losses by identifying applicants that are more likely to default on their loans. Although machine learning and deep learning models are showing promising results, they are in the *trough of disillusionment* according to Gartner's hype cycle [15]. This is due to the inability of deep learning and some machine learning algorithms to explain their predictions. In literature, this issue is referred to as the black-box nature of machine learning and deep learning techniques. Hence, tasks that involve critical decision-making, such as detecting the absence or presence of a disease; or rejection or acceptance of a loan application, require machine learning algorithms to provide reasons behind their decisions. The hype around the use of machine learning and deep learning techniques in credit scoring has been slowed down by the black-box nature of these techniques.

However, recent years have shown an emerging field known as *eXplainable Artificial Intelligence (XAI)* [16], which focuses on explaining the predictions from black-box models. The XAI is the window that brings hope to the use and acceptance of machine learning and deep learning models in credit scoring. According to Gartner's hype cycle, this window is known as the *slope of enlightenment* [15].

The XAI field is mainly fuelled by a community that consists mainly of researchers from two main streams that are in pursuit of XAI, i.e., the first stream is FAT (Fairness, Accountability, and Transparency) and the second stream is DARPA (Defence Advanced Research Projects Agency) [17]. The need for XAI arises from the ethical considerations or regulatory requirements [17]. Hence, this thesis aims to make black-box algorithms explainable so that banks meet ethical and regulatory considerations, and be able to leverage the high-performance nature of these algorithms. Please note that without creating confusion regarding the thesis title, deep neural networks are synonymous with deep learning.

1.1 Literature review

In literature, different words are used for the explanation/rationalization of black-box models, e.g., *interpretability, comprehensibility*, and *explainability*. In this study, the words mentioned above will be used interchangeably. An explanation is required when

one makes crucial decisions, e.g. when a medical doctor prescribes medication or treatment to a patient.

A significant drawback of black-box algorithms, e.g., Deep Neural Networks and Random Forests, is that the underlying representations they learn are impossible to understand. This could result in unintentional discrimination of customers or wrong prescriptions to patients. The transparency of black-box models has been studied extensively in the literature [18–25]. Shen et al. [26], presented an interpretable Deep Hierarchical Semantic Convolutional Neural Network (HSCNN) to classify a provided pulmonary nodule observed on a Computed Tomography (CT) scan as either malignant or benign. The results showed that the proposed method produces interpretable lung cancer predictions and achieves significantly better results than 3D Convolutional Neural Network (CNN) alone. Liu et al. [27], proposed a technique called Convolutional Neural Network Interpretation (CNN-INTE) to interpret deep Convolutional Neural Networks via meta-learning. The method achieves global explanation for test records on the hidden layers without compromising the accuracy achieved by the original deep CNN model. Shirakawa and Fukumura [28] proposed a five-layer auto-encoder based on a sensory integration model. Their objective was to show that extracting appropriate features and obtaining the desired expression in the middle layer of an autoencoder is possible. Ueno and Zhao [29] suggested using Decision Trees (DTs) to interpret each layer of Deep Neural Networks. The experimental results showed that accurate DTs could be extracted from the hidden layers, and the size of the DTs can be used to determine the needed number of layers. Kim and Canny [30] used a two-stage approach. In the first stage, they used a visual attention model to train a convolutional neural network from images to the steering angle. The attention model highlights regions on the images that possibly affect the network's output. In the second stage, they applied a causal filtering step to determine which input regions affect the output. Their approach produced succinct visual explanations and more accurately exposed the network's behavior.

Yeganejou and Dick [31] proposed to create more explainable deep networks by hybridizing convolutional neural networks with fuzzy logic. The fuzzy logic increases interpretability because it handles complexity and uncertainty in a human-like manner. The experimental results showed that the proposed classifier performs similarly to other deep learning algorithms, and significantly performs better than the same fuzzy classifier applied to the original image.

Baesens et al. [32] evaluated and contrasted three neural network rule extraction techniques for credit-risk evaluation, namely, Neurorule, Trepan, and Nefclass. The conclusion was that the neural network rule extraction is an effective tool that allows the development of efficient decision-support systems for credit risk determination. Setiono

and Liu [33] developed a NeuroLinear model for obtaining decision rules from neural networks. The experimental results showed that NeuroLinear effectively extracts concise and understandable rules with high accuracy from neural networks. Bologna and Hayashi [34] trained Discretized Multi-Layer Perceptron (DIMLP) by deep learning on the MNIST dataset, and then symbolic rules were extracted more efficiently concerning standard MLPs.

Most of the methods found in the literature explain why a default decision was taken. However, these methods do not suggest what actions should be taken to result in the desired outcome such as loan application approval. A counterfactual explanation is suitable for explaining what steps must be taken by an applicant whose loan application has been denied. Recently, there has been a growing interest in counterfactual explanations. In a literature survey by Verma et al. [35], a rubric was designed to evaluate the desired properties of counterfactuals, and 39 articles were compared and contrasted. The aim of Verma et al. [35] is to help organizations better understand the advantages and disadvantages of different methods used to generate counterfactuals. In another literature survey on counterfactuals, Guidotti [36] proposed categorizing methods that generate counterfactuals, and the properties of the generated counterfactuals. The findings revealed that there had been an increase in the development of counterfactual explanation methods in the last two years. The empirical results showed that the counterfactual explainers that generate a single counterfactual have more properties and require the highest computational time.

1.2 Application Scorecard

When an applicant applies for a loan, a bank collects applicant's information. The collected information consists of *demographic information* (e.g. employment, education, income, marital status). The *bureau information*, also known as bureau data, is also obtained from local bureaus (e.g. number of credits, name of the employer, and payment history). For existing bank customers, internal behavioral data is collected (e.g. payment history). After that, an applicant gets scored using application data, bureau data, and internal behavioral data (internal behavioral data is only applicable to existing customers of the bank). Each feature is allocated points, and the final score is an average of the points allocated to the features. If an applicant's score is below/above a chosen threshold (i.e. a chosen score), then the loan application of an applicant gets rejected/accepted. Once accepts (i.e. applicants who have been given loans) are identified, their loan repayments are tracked over an identified period (e.g. 12 months). A *target flag* (i.e. Good/Bad Flag) gets created based on the loan repayments history of the accepts.

An applicant is *good* if they are not in arrears for three consecutive months or more, otherwise, an applicant is *bad*. For rejects, good and bad applicants are inferred using *reject inference* [1]. The reject inference refers to the possibility to infer whether a rejected applicant would have been good or bad if the loan had been offered.

Note that, within the accepts, there are applicants who do not take the loan offer (referred to as non-take-ups), and there are applicants who do take the loan offer (referred to as take-ups). Hence, only take-ups get used for scorecard development. Essential variables are selected by using *Information Value* (IV), *correlation*, and a *business sense* [1].

The application scorecard is finally developed on accepts and rejects (i.e. Through-The-Door (TTD) population).

1.2.1 Default Definition

Before we define default, we first discuss the development and performance windows. The development window is defined as a period from which the observed good and bad cases are chosen for developing application scorecards as outlined by Siddiqi [1].

The performance window is defined as a time where the performance of accounts opened at a specific time is tracked over time (generally 12 months) [1]. Thus, at the end of 12 months, a good/bad flag is assigned to accounts opened at a specific time. See Figure 1.1 for a schematic view. Figure 1.2 shows a general view of how accounts perform over time.

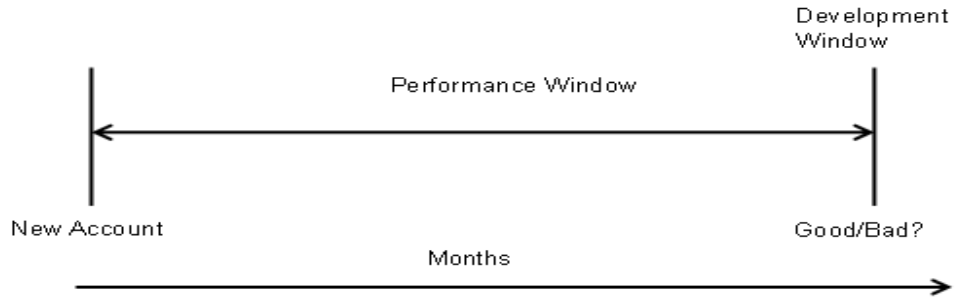

FIGURE 1.1: Default definition [1].

Low bad rates are observed in the first few months, and as time progresses, these bad rates gradually increase because approved applicants are likely to default on their loans. The development period is then chosen at a time when bad rates have stabilized. This minimizes the chances of misclassifying performances [1], giving all accounts enough time to go bad. The bad definition is derived from *roll rate analysis*[1]. The roll rate analysis determines a bucket where most delinquent accounts roll forward to the worst bucket. The bucket refers to the number of days in delinquency/arrears. The bad definition differs from product to product. Generally, an account is bad if the number of days in arrears is 90 or more Siddiqi [1].

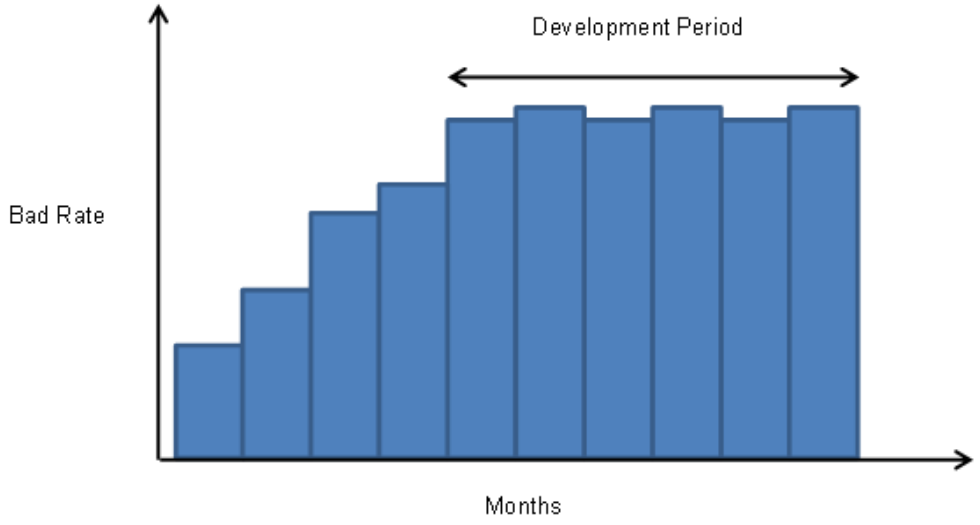

FIGURE 1.2: Stabilization of bad rates [1].

1.2.2 Scoring

An applicant's creditworthiness is assessed using a score generated by a scorecard. This score is an average of all the points allocated to the features of a loan applicant. The key to the allocation of points to features is the *Weight of Evidence* (WOE). The WOE

is calculated as follows:

$$WOE = \ln\left(\frac{Distr.Goods}{Distr.Bads}\right),$$

where *Distr.Goods* is the distribution of goods and *Distr.Bads* is the distribution of bads.

The weights of evidence act as independent inputs in a logistic regression function given below:

$$\ln(odds) = \beta_0 + \sum_{i=1}^{d} \beta_i \sum_{j=1}^{ki} WOE_{ij},$$

where odds=probability(of an event happening)/probability(of an event not happening), d is the number of features and ki is the number of groups (i.e., attributes) in the ith feature. The β_0 and β_i are the intercept term and the coefficients, respectively. Finally, a score is given as follows:

$$Score = \beta_0 + \sum_{i=1}^{d} \beta_i \sum_{j=1}^{ki} WOE_{ij}. \tag{1.1}$$

However, in practice, scaling is applied which refers to the scale and format of scores in a scorecard and the change rate in odds for increases in scores [1]. When scaling is applied, equation 1.1 becomes:

$$Score = Offset + Factor * \ln(odds). \tag{1.2}$$

Introducing a term called *points double the odds (pdo)*, equation 1.2 becomes:

$$Score + pdo = Offset + Factor * \ln(2 * odds). \tag{1.3}$$

Solving equation 1.2 and equation 1.3 simultaneously, results in the following:

$$pdo = Factor * \ln(2).$$

Thus $Factor$ is given as follows:

$$Factor = \frac{pdo}{\ln(2)}.$$

Typically, a default value for *pdo* is 20 [1], and the scorecard developer can give the offset upfront (e.g., 600). A cut-off score is determined upfront for a loan application to be accepted/rejected. Several methods are used to determine the cut-off score in practice. In [1], Siddiqi mentions *adverse codes* where WOE is set to zero. The other method uses

a Kolmogorov-Smirnov (K-S) statistic score, i.e., a score where the distance between the cumulative distribution of goods and the cumulative distribution of bads is maximal and is chosen as a cut-off score. A score that is less than a cut-off score is attributable to features with fewer points. Hence, the features with fewer points are used to explain a loan rejection.

1.2.3 Research Questions

1. What is the comparative performance of deep learning models, traditional machine learning models, and statistical models in credit scoring?
2. How effectively can explanation techniques be enhanced to better explain predictions from deep learning models and other machine learning models?

1.2.4 Hypotheses

This research will focus on the following hypotheses:

1. Deep learning models outperform traditional machine learning models and statistical models in credit scoring.
2. The rationalization of deep learning and other machine learning models will uncover meaningful and informative features that are valuable to both credit applicants and decision-makers in credit scoring.

1.2.5 Aims and Research Objectives

The specific *aims* of the thesis would involve examining the performances of different models, exploring the capabilities of explaining model predictions; and determining the value and utility of the features (identified through the rationalization process) for various stakeholders involved in credit scoring.

The specific *objectives* of the thesis are set as follows:

1. Evaluate and compare (both quantitatively and qualitatively) the performance of deep learning models, traditional machine learning models, and statistical models in credit scoring.
2. Investigate the extent to which rationalization of deep learning models and other machine learning models can explain credit decision outcomes.

3. Assess the effectiveness of the rationalization process in uncovering meaningful and informative features for credit applicants and decision-makers.

1.2.6 Contributions

The contributions of this research are:

1. A guiding machine learning framework for credit risk scoring practitioners.
2. Discovering meaningful and informative features for various stakeholders in credit scoring through model rationalization.
3. To make 2D Convolutional Neural Networks applicable to tabular datasets and explain their predictions.
4. Empirically showing that the explanations can be used as good features for classification.
5. Assessing the validity of deep learning and machine learning prediction explanations using credit scoring experts.
6. A novel formulation of the optimization problem that encompasses multiple counterfactual properties simultaneously without compromising any of the proposed counterfactual properties.

1.2.7 Thesis by publication layout

- Chapter 1 discusses a brief history of artificial intelligence and credit scoring. A brief literature review on the rationalization of black-box models is presented. An overview of how credit scoring works is discussed in detail.
- In Chapter 2, four journal publications are arranged logically, where the succeeding journals extend from the preceding journals.
- Chapter 3 discusses all four articles and summarizes critical findings in the conclusion.

Chapter 2

Thesis Publications

In this thesis, applications of machine learning in credit scoring are investigated. Further, the explanations of machine learning predictions are studied. **Article I** investigates and compares different statistical and machine learning techniques in credit scoring that are found in the literature. Limitations in credit scoring literature are identified and discussed. A guiding machine learning framework for credit scoring practitioners is proposed. The results show that ensemble techniques perform better than statistical and traditional machine learning techniques. Even though deep learning techniques are not studied extensively in credit scoring literature, deep learning techniques show better performances compared to statistical and classic machine learning models. This propelled us to investigate the performances of deep learning techniques in publicly available credit scoring datasets, and after that, explain their predictions.

Article II focuses on applying deep learning techniques in credit scoring, specifically convolutional neural networks. Firstly, tabular data is converted into images using a novel transformation technique. Secondly, a 2D convolutional neural network is trained on the converted image data. Thirdly, convolutional neural network predictions are explained using state-of-the-art explanation techniques. Lastly, explanations are shown to be good features for classification. According to our knowledge, this is the first study that converts tabular data into images and then explains the predictions made on the images. Despite this achievement, the explanations cannot tell the applicant whose loan has been denied how to make changes to get the loan approved in the future. The following article addresses this issue.

Article III proposes a novel explanation technique using counterfactual explanations derived from a custom genetic algorithm. The proposed method is model agnostic and non-intrusive (i.e. the proposed technique does not tamper with the inner workings of the actual black-box model). The counterfactual explanations are sparse, focusing

on changing a few features to affect the desired outcome. The explanations from the proposed method are validated by using credit scoring experts. The proposed method does not only explain rejected loans; it can also be used to explain accepted loans to help applicants with their future financial decisions. This article did not consider the performances of the generated counterfactuals. Hence, the following article looks into the performances of the generated counterfactual explanations.

Article IV proposes a novel optimization formulation to generate counterfactuals that possess multiple properties simultaneously. The counterfactual properties considered are validity, similarity, sparsity, actionability, and plausibility. The proposed approach is assessed using three optimization techniques: genetic algorithm, particle swarm optimization, and Bayesian optimization. In addition, the properties of the cost function, such as fairness and robustness, were also assessed. The results show that it is possible to generate counterfactuals that possess multiple properties simultaneously with a cost function that is fair and robust. Further, we performed statistical significance testing for performance comparison between our approach and state-of-the-art approaches. The empirical results show that there is a trade-off between validity and sparsity. Further, the empirical results show that our proposed approach does not compromise the proposed counterfactual properties.

In the sequel, we show original copies of the articles which discuss the above points in detail.

2.1 Journal Article I

X. Dastile, T. Celik and M. Potsane, Statistical and machine learning models in credit scoring: A systematic literature survey, *Applied Soft Computing*, vol. 91, pp. 106263, 2020.

Number of citations to date: 141
Publisher: Elsevier
Cite Score: 11.2
Impact Factor: 6.725
Abstracting and Indexing: SCOPUS
Science Citation Index Expanded
CompuMath Citation Index
Current Contents-Engineering, Computing & Technology
INSPEC SNIP: 2.472

Applied Soft Computing Journal 91 (2020) 106263

Contents lists available at ScienceDirect

Applied Soft Computing Journal

journal homepage: www.elsevier.com/locate/asoc

Statistical and machine learning models in credit scoring: A systematic literature survey

Xolani Dastile [a,*], Turgay Celik [a,b], Moshe Potsane [a]

[a] School of Computer Science and Applied Mathematics, University of the Witwatersrand, Johannesburg, South Africa
[b] Wits Institute of Data Science, University of the Witwatersrand, Johannesburg, South Africa

ARTICLE INFO

Article history:
Received 3 June 2019
Received in revised form 23 February 2020
Accepted 19 March 2020
Available online 25 March 2020

Keywords:
Credit scoring
Statistical learning
Machine learning
Deep learning
Systematic literature survey

ABSTRACT

In practice, as a well-known statistical method, the logistic regression model is used to evaluate the credit-worthiness of borrowers due to its simplicity and transparency in predictions. However, in literature, sophisticated machine learning models can be found that can replace the logistic regression model. Despite the advances and applications of machine learning models in credit scoring, there are still two major issues: the incapability of some of the machine learning models to explain predictions; and the issue of imbalanced datasets. As such, there is a need for a thorough survey of recent literature in credit scoring. This article employs a systematic literature survey approach to systematically review statistical and machine learning models in credit scoring, to identify limitations in literature, to propose a guiding machine learning framework, and to point to emerging directions. This literature survey is based on 74 primary studies, such as journal and conference articles, that were published between 2010 and 2018. According to the meta-analysis of this literature survey, we found that in general, an ensemble of classifiers performs better than single classifiers. Although deep learning models have not been applied extensively in credit scoring literature, they show promising results.

© 2020 Elsevier B.V. All rights reserved.

1. Introduction

The history of credit scoring dates back to the 1950s [1]. During those early years, credit decisions were made using a judgmental approach commonly known as 5C's approach [2]:

> Character: do you know the person or their family?
> Capital: how much is being asked for?
> Collateral: what is the borrower willing to put up from their resources?
> Capacity: what is their repayment ability?
> Condition: what are the conditions in the market?

The drawback of using the 5C's approach was the incapability of processing a huge number of applications daily. This resulted in the advent of scorecards, which make consistent nonjudgmental decisions that treat all borrowers fairly. The scorecards generate a score which quantifies the risk of lending money to borrowers.

When a borrower applies for a loan, the financial institution, without losing generality hereafter referred to as the bank, collects information from the borrower. This information is called application data and it consists of *demographic information*, e.g., the number of dependents, time at current address, time at current employment, etc. The *bureau information* is also collected from the local bureaus, and it includes the number of inquiries, judgments, number of delinquencies, etc. Once the accepted population, i.e. borrowers who have been granted loans, are identified, their loan repayment history is tracked for a period of time, e.g. 24-months. A *target flag* (i.e. good/bad flag) gets created based on loan repayment history of the accepted population. If the number of days past due (or missed payments) is less than a certain number of days, e.g. 90 days, then the borrower is flagged as a *good* borrower, otherwise a *bad* borrower. The known goods and bads are then used to develop a scorecard. The cut-off score is determined by the *Kolmogorov–Smirnov statistic*, and it measures the distance between the cumulative distribution of goods and bads. The score which gives the maximum distance between the distribution of goods and bads is regarded as the cut-off score and is used to predict the goods and the bads. If a score of a borrower is larger than or equal to the cut-off score, then the borrower is predicted as a good borrower otherwise a bad borrower. The scorecard is then applied to the rejected population to predict goods and bads and this is referred to as *reject inference* [3]. The final application scorecard is built on the accepted and rejected populations, i.e. Through-The-Door (TTD) population. Please see Fig. 1 for a schematic view of data flowchart for application scorecards. Traditionally, financial institutions use a logistic regression to score borrowers. The choice of using the logistic regression

* Corresponding author.
E-mail addresses: xdastile12@gmail.com (X. Dastile), celikturgay@gmail.com (T. Celik), Moshe.Potsane@wesbank.co.za (M. Potsane).

https://doi.org/10.1016/j.asoc.2020.106263
1568-4946/© 2020 Elsevier B.V. All rights reserved.

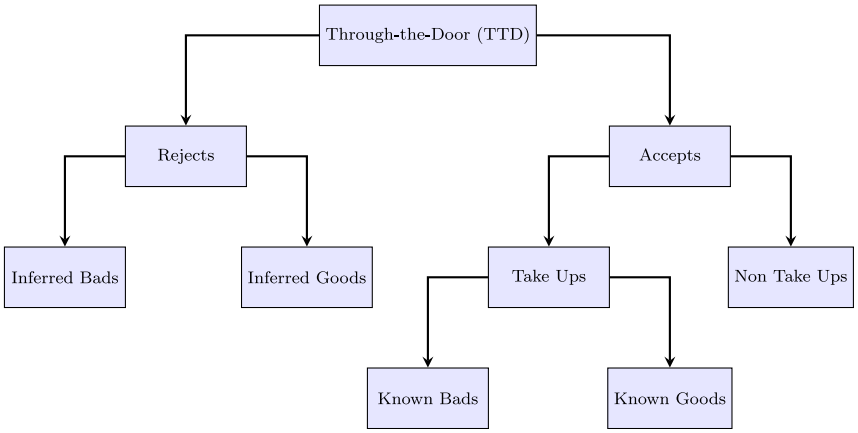

Fig. 1. Schematic view of data flowchart for application scorecards.

model is due to its simplicity and transparency. The scores from logistic regression are calculated using the following formula [3]

$$Score = \frac{pdo}{\ln(2)} \times \left(\beta_0 + \sum_{i=1}^{n} \beta_i \sum_{j=1}^{k_i} WOE_{i,j}\right), \tag{1}$$

where

$$WOE_{i,j} = \ln\left(\frac{F_{G_j}}{F_{B_j}}\right) \tag{2}$$

and n is the number of features, k_i is the number of groups or attributes in the ith feature, pdo is points double the odds and is used as a scaling factor and F_{G_j} and F_{B_j} are distribution of good borrowers and bad borrowers in the jth attribute in feature i, respectively. The β_i coefficients are estimated using *Maximum Likelihood Estimation* [4].

In literature, sophisticated machine learning (ML) models can be found that can replace the logistic regression model. In spite of high accuracies from ML models; ML models are generally unable to explain their predictions. Financial institutions are regulated entities and are required to be transparent in their decisions when using application scorecards. However, techniques that can deduce rules to mitigate the lack of transparency without even compromising accuracy are suggested in the literature, e.g. [5]. In this paper, we aim to present a systematic literature survey of statistical and machine learning models which are employed in credit scoring between 2010 and 2018 and propose a guiding ML framework for credit scoring. The remainder of this paper is organized as follows. In Section 2 we cover the methodology followed for conducting this systematic literature survey. In Section 3 we highlight feature selection/engineering methods. The learning models are covered in Section 4. In Section 5 we present evaluation metrics. The data imbalance is covered in Section 6. In Section 7 we cover model transparency. In Section 8 we discuss limitations and assumptions of different models. In Section 9 we discuss emerging trends. Section 10 discusses limitations in literature. In Section 11 we discuss results. In Section 12 we propose a guiding framework for machine learning in credit scoring and finally Section 13 provides conclusion and future work.

2. Survey methodology

This study employs a systematic literature survey approach to

(1) systematically review the most commonly used statistical and machine learning techniques in credit scoring;
(2) identify limitations in literature;
(3) propose a guiding machine learning framework to perform credit scoring;
(4) point to emerging directions.

In this survey, the statistical techniques considered include Linear Discriminant Analysis (LDA), Logistic Regression (LR) and Naïve Bayes (NB), and the machine learning include k-Nearest Neighbor (k-NN), Decision Trees (DTs), Support Vector Machines (SVMs), Artificial Neural Networks (ANNs), Random Forests (RFs), Boosting, Extreme Gradient Boost (XGBoost), Bagging, Restricted Boltzmann Machines (RBMs), Deep Multi-Layer Perceptron (DMLP), Convolutional Neural Networks (CNNs) and Deep Belief Neural Networks (DBNs). Note that the list of these techniques for both statistical and machine learning is not exhaustive, the literature provides a myriad of techniques that are used to model credit scoring. However, this study only reviews the most commonly used techniques as it would be almost impossible to look at all techniques applied in credit scoring.

A process flowchart of the systematic literature survey methodology is presented in Fig. 2. Note that this study follows the methodology of the recently published systematic literature survey on bankruptcy prediction models [6]. The search words of the current study are "Statistical Learning in Credit Scoring", "Machine Learning in Credit Scoring" and "Deep Learning in Credit Scoring". The inclusion criterion for articles is based on the recently published work in credit scoring and the period considered is the year 2010 to the year 2018. The study selects peer-reviewed journal and conference articles as these are considered to be of high quality [7]. The articles are selected by reading the abstract and the conclusion, in some instances the entire article is read. This review is based on articles that are written in English only. This overlooks one of the requirements in systematic literature review where language constraints are discouraged. The following databases are used in searching papers: Google Scholar, Science Direct, IEEEXplore, ACM and SpringerLink. These databases include studies which are undertaken all over the world, hence geographical bias is removed.

All unpublished work and dissertations are not included in this current study. In the end, 74 primary studies were selected for this systematic literature survey. The primary studies include models in their hybrid form (i.e. where feature selection/feature engineering is combined with a classifier or ensemble classifiers) and in their standalone form. A meta-analysis of the results from the selected articles is done by producing summary tables, pie-charts and histograms. The German and Australian credit datasets are the most frequently used datasets in credit scoring, hence we used these two datasets for model performance comparisons.

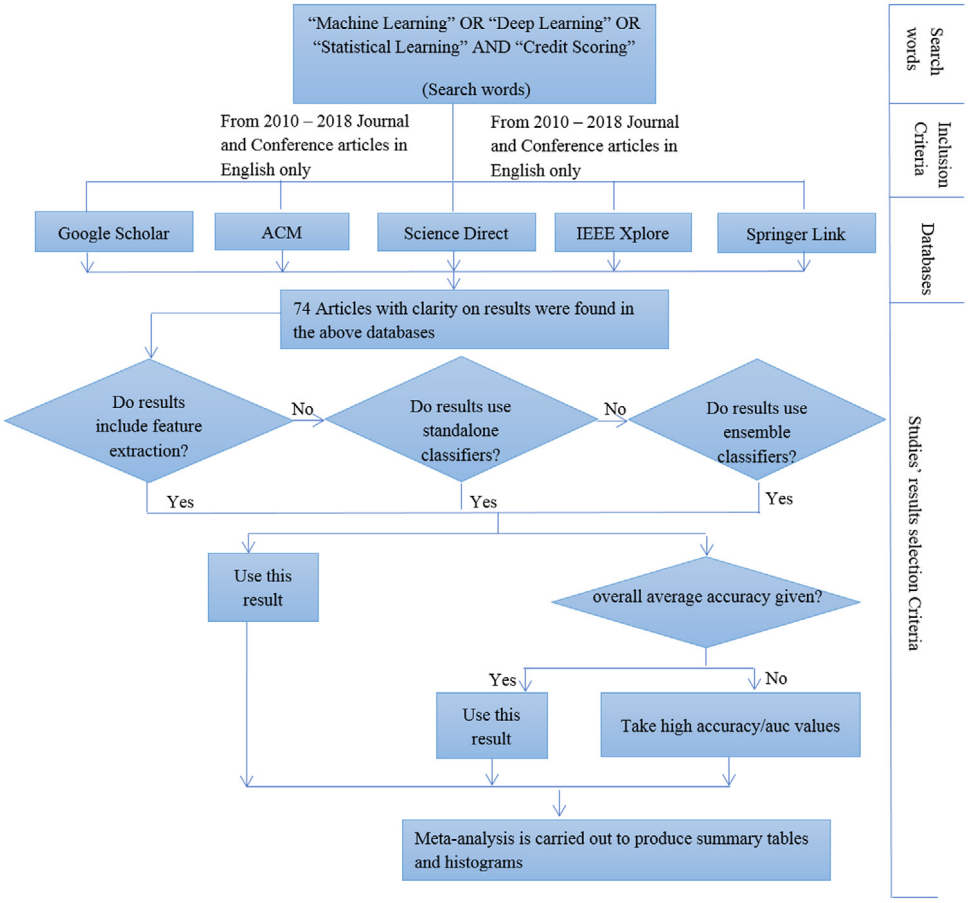

Fig. 2. Process flowchart of the systematic literature survey methodology.

2.1. Existing literature surveys on bankruptcy prediction or credit scoring

There are several literature surveys on bankruptcy prediction or credit scoring [6,8–13]. We briefly highlight the objectives of each literature survey and show where they differ with our study in Table 1. The literature surveys in Table 1 focus mostly on comparing models based on their performances. However, the listed literature surveys ignore key factors in credit scoring, such as model transparency and the nature of datasets. Furthermore, deep learning models are not considered in the listed literature surveys in Table 1.

Among the existing literature surveys shown in Table 1, [12] provide the most recent and the most comprehensive systematic literature survey in credit scoring. The main objective of [12] is "proposing a new method for rating, comparing traditional techniques, conceptual discussions, feature selection, literature review, performance measures studies and, at last, other issues". Their literature survey covers a period from January 1992 to December 2015 and focuses on 12 questions on the conceptual scenario over the techniques. Comparing [12] with our literature survey, there is an overlap from 2010 to 2015. However, our literature survey includes the most recent years. According to the best of our knowledge, there is no systematic literature survey for credit scoring that has been published which encompasses, statistical learning, traditional machine learning and deep learning models.

3. Feature selection and feature engineering

This section provides a review of most commonly used *feature selection* (FS) and *feature engineering* (FE) techniques used in credit scoring. The distinction between feature selection and feature engineering is that the feature selection selects a subset of features from the entire feature set whereas feature engineering creates a new set of features from the existing features. Albeit, Liang et al. [14] investigated the effect of feature selection and concluded that performing feature selection does not always improve the prediction performance, the studies in [15–19] showed that removing redundant features can improve model performance in credit scoring. There are also studies [20–22] which employed meta-heuristic approach such as *Genetic Algorithm* for feature selection. We divided the feature selection into *filter*, *wrapper* and *embedded* methods.

3.1. Filter methods

The filter methods perform feature selection based on the univariate analysis of the features without using a predictor. They compute a score for each feature and select a subset of features based on their scores. In the following, we present most commonly used filter methods in credit scoring.

3.1.1. F-score

F-score measures the discrimination of two sets of real numbers [17]. Given m training vectors $\mathbf{x}_k$, where $k = 1, 2, \ldots, m$, and if the number of positive and negative instances are $m^{(+1)}$ and $m^{(-1)}$, respectively, then the F-score of the ith feature $F(i)$ is defined as follows

$$F(i) = \frac{(\bar{x}_i^{(+1)} - \bar{x}_i)^2 + (\bar{x}_i^{(-1)} - \bar{x}_i)^2}{\frac{1}{m^{(+1)}-1}\sum_{k=1}^{m^{(+1)}}(x_{k,i}^{(+1)} - \bar{x}_i^{(+1)})^2 + \frac{1}{m^{(-1)}-1}\sum_{k=1}^{m^{(-1)}}(x_{k,i}^{(-1)} - \bar{x}_i^{(-1)})^2} \quad (3)$$

Table 1
Existing literature surveys on bankruptcy prediction or credit scoring and their differences from this survey paper.

Survey paper	Articles searched	Years	Objective	Difference from this survey
[8]	165	1965–2010	✓ Investigation of model type by decade ✓ Analysis of Number of features used by decade ✓ Compare model performance	✓ Transparency is not reported ✓ Deep learning models are not covered ✓ Nature of datasets (balanced vs imbalanced) is not reported
[9]	214	Not specified	✓ Comparing models based on performance	✓ Transparency is not reported ✓ Deep learning models are not covered ✓ Nature of datasets (balanced vs imbalanced) is not reported
[10]	130	1995–2010	✓ Models are compared based on design, datasets and baselines	✓ Transparency is not covered ✓ Deep learning models are not covered ✓ Nature of datasets (balanced vs imbalanced) is not reported
[11]	Not specified	Not specified	✓ Comparing models based on performance	✓ Transparency is not reported ✓ Deep learning models are not covered ✓ Nature of datasets (balanced vs imbalanced) is not reported
[12]	187	1992–2015	✓ Proposing a new method for scoring ✓ Compare traditional techniques ✓ Conceptual discussion	✓ Transparency is not reported ✓ Deep learning models are not covered ✓ Nature of datasets (balanced vs imbalanced) is not reported
[13]	6	Not specified	✓ Compare model performance	✓ Transparency is not reported ✓ Deep learning models are not covered ✓ Nature of datasets (balanced vs imbalanced) is not reported
[6]	49	2010–2015	✓ Compare models based on accuracy, transparency, data size capability, data dispersion and etc.	✓ Deep learning models are not covered ✓ Nature of datasets (balanced vs imbalanced) is not reported

where $x_{k,i}$ is the ith feature of the kth sample, $\bar{x}_i$, $\bar{x}_i^{(+1)}$ and $\bar{x}_i^{(-1)}$ are the averages of the ith feature of the whole, positive, and negative datasets, respectively [17]. The numerator indicates the discrimination between the positive and negative sets, and the denominator indicates the one within each of the two sets. The larger the F-score value is, the more discriminative power the feature has [17,23].

3.1.2. Rough set theory

In rough set theory, the sample dataset is called information system, denoted by $I = (U, A)$, where U is a non-empty finite set of observations called the universe and A is a non-empty finite set of features. Firstly, a notion of *indiscernibility relation* on a universe set is defined. With every subset $B \subseteq A$, an indiscernibility relation on U is defined as follows

$$I(B) = \{(\mathbf{x}, \mathbf{y}) \in U \times U : f_i(\mathbf{x}) = f_i(\mathbf{y}), \forall i \in B\}, \tag{4}$$

where $f_i : U \rightarrow V_i$ is the ith feature function and V_i is a set of values associated with feature i. Feature i is redundant in B if $I(B) = I(B - \{i\})$, otherwise feature i is important in B. If all of the features in B are important, then B is said to be a reduced set of features [24–26].

3.2. Wrapper methods

The wrapper methods select a subset of features based on an evaluation metric, such as accuracy, of a pre-determined predictor. Each subset of the features are scored according to their predictive power, thus, the wrapper methods are computationally expensive compared to the filter methods. However, since they select features based on the performance of the pre-determined predictor, the wrapper methods usually outperform filter methods in credit scoring. In the following, we present the most commonly used wrapper methods in credit scoring.

3.2.1. Stepwise selection

The stepwise feature selection has three components [3], namely, *forward feature selection*, *backward feature elimination* and a *stepwise feature selection* which is a combination of both forward feature selection and backward feature elimination. All three components use linear regression and a *p-value* for feature selection. The *forward feature selection* starts by regressing one feature and if the feature is significant according to the *p-value* then that feature is retained. This process is repeated by adding one feature at a time until the list of features is exhausted, and the significant features are retained and insignificant features are discarded. The *backward feature elimination* works the opposite way, you start with the entire set of features and you keep discarding features with *p-values* less than the chosen level of significance [27]. The *stepwise feature selection* adds and removes features.

3.2.2. Genetic algorithm

A genetic algorithm [28] uses genetic inspired operators to evolve an initial population into a new population. These operators are the *selection*, *crossover* and *mutation*. Each population comprises of chromosomes (a string of bits (0/1)) that represent genetically encoded individual solutions to a specific problem. Selection operator selects chromosomes in the population (a set of solutions) for reproduction. Crossover operator randomly chooses a position and exchanges the subsequent bits before and after that position between two chromosomes to create two offsprings. Mutation operator randomly flips some of the bits in a chromosome. Each individual has a fitness score assigned to them, which represents its ability of be selected. For feature selection, the individuals are subsets of features that are encoded as bits where the ith feature is selected if the corresponding bit is set to 0. The fitness value is some measure of model performance, such as classification accuracy. A new population is evolved by using operators of crossover, where selection is based on individual's fitness function and its ability to reproduce the next

generation [29]. Hence, genetic algorithm is an evolutionary algorithm problem that is solved by searching a space (in our case a space of features). However, the key elements of problem solving by search are *exploitation* and *exploration*. The exploitation uses the information from previously visited points to determine the prospect of finding profitable regions to be visited next and the exploration is the process of visiting new regions of a search space to uncover promising offsprings [30]. "How and when to control and balance exploration and exploitation in the search process to obtain even better fitness results and/or convergence faster are still on-going research" [31]. It is key that a diversified population is maintained during the whole search process. The measure of diversity is *entropy* and it represents the amount of population disorder, where an increase in entropy results in an increase in diversity [30].

3.3. Embedded methods

The embedded methods optimize an objective function or learning classifier with a goodness-of-fit term and a penalty for a large number of features [32]. The most commonly used embedded method in credit scoring is the least absolute shrinkage and selection operator (LASSO) [33]. Given a linear regression model to predict dependent variable y_i using independent variables $x_{i,j}$s for ith sample in the training dataset, i.e.

$$y_i = \sum_{i=1}^{n} x_{i,j}\beta_j + \beta_0, \tag{5}$$

where $j = 1, 2, \ldots, n$, the LASSO [33] solves the $L1$-penalized regression problem of finding the set of parameters $\beta = \{\beta_j\}_{j=1}^{n}$ to minimize

$$\sum_{i=1}^{m}\Big(y_i - \sum_{j} x_{i,j}\beta_j\Big)^2 + \lambda \sum_{j=1}^{n} |\beta_j|, \tag{6}$$

where m is the number of instances in the training dataset and $\lambda \sum_{j=1}^{n} |\beta_j|$ represents the penalty term. The penalty term reduces coefficients β_j and simplifies the linear regression model. Thus, the LASSO performs variable selection and model shrinkage based on the magnitude of β_js.

3.4. Feature engineering

The original features may be dependent on each other which may reduce the performance of a predictor. To tackle this problem, feature engineering is used to create a new set of engineered features from the original ones. The feature engineering methods can either decrease or increase the dimensionality of feature vectors [34]. In the following, we provide a brief review of most commonly used feature engineering methods in credit scoring.

3.4.1. Principal Component Analysis

Principal Component Analysis (PCA) aims to transform/compress the data from a higher dimensional space $\mathbb{R}^n$ to a lower dimensional space $\mathbb{R}^k$ [35], where $k \ll n$. Note that relevant feature information (variance) is not lost during this transformation. The reduced feature space consists of *principal components* which are orthogonal to each other. Each principal component represents the direction of maximum variance. The principal components are computed via *eigenvalue* decomposition of the *covariance matrix* of features. The resulting *eigenvectors* from the decomposition are used as principal components. The following steps are used when constructing principal components:

1. Standardize the n dimensional dataset. Each feature $x_{i,j}$ of the feature vector $\mathbf{x}_i$ is standardized according to
$$x_{i,j} \leftarrow \frac{x_{i,j} - \mu_j}{\sigma_j}, \tag{7}$$
where μ_j and σ_j are the mean and standard deviation of jth feature.
2. Construct the *covariance matrix* $\boldsymbol{\Sigma}$. The covariance matrix is computed as
$$\boldsymbol{\Sigma} = \frac{1}{m-1}\sum_{i=1}^{m} \mathbf{x}_i \mathbf{x}_i^T, \tag{8}$$
where $\mathbf{x}_i$ is assumed a column vector.
3. Apply eigenvalue decomposition on the covariance matrix to compute eigenvalues (λ_js) and eigenvectors ($\mathbf{e}_j$s).
4. Select k eigenvectors that correspond to the k largest eigenvalues, where k is the dimensionality of the new feature subspace. Generally, the *variance explained ratio* is used to select the number of principal components (eigenvectors). For a set of all sorted eigenvalues
$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n, \tag{9}$$
the variance explained ratio is defined as follows
$$\frac{\sum_{j=1}^{k} \lambda_j}{\sum_{j=1}^{n} \lambda_j}. \tag{10}$$
The chosen k eigenvectors will have a high total explained variance.
5. Finally, use k eigenvectors and project each feature vector $\mathbf{x}$ to k-subspace, i.e.,
$$\hat{\mathbf{x}} = (\mathbf{x} - \boldsymbol{\mu})^T[\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_k], \tag{11}$$
where $\boldsymbol{\mu} = [\mu_1, \mu_2, \ldots, \mu_n]^T$ is the mean vector and $\hat{\mathbf{x}} \in \mathbb{R}^k$ is the projection of $\mathbf{x}$ onto k-subspace.

3.4.2. Autoencoders

Fig. 3 shows an autoencoder which is a type of neural network with three layers, namely; input, hidden and output layers. For a typical autoencoder, the input variables are the same as the target variables. The autoencoder learns representations of its inputs at hidden layer and tries to reconstruct its inputs from the learned representations by optimizing a cost function on a training dataset, i.e.

$$E = \frac{1}{2}\sum_{i=1}^{m} \|a_2(a_1(\boldsymbol{\alpha}^{(1)}\mathbf{x}_i)\boldsymbol{\alpha}^{(2)}) - \mathbf{x}_i\|_2^2, \tag{12}$$

where $\boldsymbol{\alpha}^{(1)}$, $\boldsymbol{\alpha}^{(2)}$ are weights and a_1, a_2 are activation functions between input and hidden layer, hidden layer and output layer, respectively. Once $\boldsymbol{\alpha}^{(1)}$ and $\boldsymbol{\alpha}^{(2)}$ are learned, a feature vector $\mathbf{x}$ is mapped to $\hat{\mathbf{x}}$ as follows $\hat{\mathbf{x}} = \sigma_1(\boldsymbol{\alpha}^{(1)}\mathbf{x})$ which is further used as engineered feature vector. One can choose different number of hidden layers for the autoencoder to learn complex representations in the data. Furthermore, the autoencoder can both decrease or increase the dimensionality of the projected vectors $\hat{\mathbf{x}}$ with respect to the dimensionality of the input feature vector $\mathbf{x}$.

3.4.3. Linear discriminant analysis

Linear discriminant analysis (LDA) was first introduced by Fisher [36]. The following outlines the steps taken when using LDA for feature selection [37].

1. Calculate n-dimensional mean vectors $\boldsymbol{\mu}^{(+1)}$ and $\boldsymbol{\mu}^{(-1)}$ for "good" (or positive (+1)) and "bad" (or negative (−1))

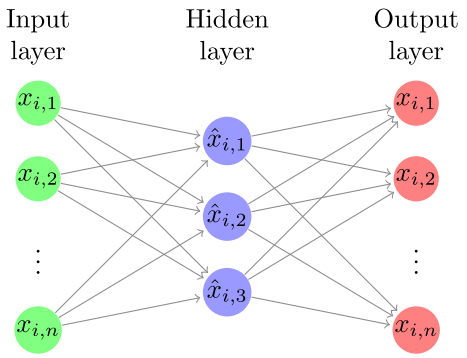

Fig. 3. An example of an autoencoder with one hidden layer. The green circles represent inputs, the blue circles represent engineered features, the red circles are reconstructed inputs.

classes, respectively:

$$\boldsymbol{\mu}^{(+1)} = \frac{1}{m^{(+1)}} \sum_{k=1}^{m^{(+1)}} \mathbf{x}_k^{(+1)}, \boldsymbol{\mu}^{(-1)} = \frac{1}{m^{(-1)}} \sum_{k=1}^{m^{(-1)}} \mathbf{x}_k^{(-1)}. \tag{13}$$

2. Construct the within-class scatter matrix $\mathbf{S}_w$ and the between-class scatter matrix $\mathbf{S}_b$:

$$\mathbf{S}_w = \sum_{\forall c \in \{+1,-1\}} \frac{m^{(c)}}{m^{(+1)} + m^{(-1)}} \sum_{k=1}^{m^{(c)}} \left(\mathbf{x}_k^{(c)} - \boldsymbol{\mu}^{(c)}\right) \left(\mathbf{x}_k^{(c)} - \boldsymbol{\mu}^{(c)}\right)^T \tag{14}$$

$$\mathbf{S}_b = \sum_{\forall c \in \{+1,-1\}} \frac{m^{(c)}}{m^{(+1)} + m^{(-1)}} \left(\boldsymbol{\mu}^{(c)} - \boldsymbol{\mu}\right) \left(\boldsymbol{\mu}^{(c)} - \boldsymbol{\mu}\right)^T \tag{15}$$

where $\boldsymbol{\mu} = \frac{m^{(+1)}}{m^{(+1)}+m^{(-1)}} \boldsymbol{\mu}^{(+1)} + \frac{m^{(-1)}}{m^{(+1)}+m^{(-1)}} \boldsymbol{\mu}^{(-1)}$ is the overall mean of the dataset.

3. Apply eigenvalue decomposition on the matrix $\mathbf{inv}(\mathbf{S}_w)\mathbf{S}_b$ to compute eigenvalues (λ_js) and eigenvectors ($\mathbf{e}_j$s), where $\mathbf{inv}(\cdot)$ is matrix inverse.
4. Similar to PCA, select k eigenvectors that correspond to the k largest eigenvalues.
5. Finally, use k eigenvectors and project each feature vector $\mathbf{x}$ to k-subspace, i.e.,

$$\hat{\mathbf{x}} = \mathbf{x}^T[\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_k], \tag{16}$$

where $\hat{\mathbf{x}} \in \mathbb{R}^k$ is the projection of $\mathbf{x}$ onto k-subspace.

4. Supervised learning

Credit scoring is a *supervised learning* problem, specifically it is a binary classification problem, where the aim is to classify good borrowers and bad borrowers. In this section we first define supervised learning problem and then present most commonly used statistical and machine learning methods in credit scoring.

4.1. Supervised learning problem

In the context of credit scoring, supervised learning problem can be defined as searching for a model $\phi : \mathbb{R}^n \mapsto \{+1, -1\}$ which maps a feature vector $\mathbf{x} \in \mathbb{R}^n$ to a predicted class label $\hat{y} \in \{+1, -1\}$, i.e

$$\phi_{\boldsymbol{\alpha}} : \mathbf{x} \rightarrow \hat{y}, \tag{17}$$

$\boldsymbol{\alpha}$ is the set of the model parameters. For the ease of reading $\phi_{\boldsymbol{\alpha}}$ and ϕ are used interchangeably.

In order to learn the model parameters, information from the previously processed applicants are collected to form a labeled dataset

$$\mathcal{D} = \{(\mathbf{x}_k, y_k)\}_{k=1}^m,$$

where $\mathbf{x}_k \in \mathbb{R}^n$ denotes kth applicant's feature vector and $y_k \in \{+1, -1\}$ is the corresponding label from the set of classes (good borrower $= +1$) and (bad borrower $= -1$). The dataset $\mathcal{D}$ is split into a training set $\mathcal{D}_{train} \subset \mathcal{D}$ and a test dataset $\mathcal{D}_{test} \subset \mathcal{D}$ where $\mathcal{D}_{train} \cap \mathcal{D}_{test} = \emptyset$ and $\mathcal{D}_{train} \cup \mathcal{D}_{test} = \mathcal{D}$.

Once the training and testing datasets are formed, the parameters $\boldsymbol{\alpha}$ of the model ϕ are learned on the training dataset $\mathcal{D}_{train}$ by minimizing a cost function C

$$C(\mathcal{D}_{train}, \phi) = \sum_{\forall \mathbf{x}_k \in \mathcal{D}_{train}} d(y_k, \phi(\mathbf{x}_k)), \tag{18}$$

$d(y_k, \phi(\mathbf{x}_k))$ is a distance measure, such as the Mean Squared Error (MSE) or Cross-Entropy, between y_k and its prediction $\hat{y}_k = \phi(\mathbf{x}_k)$ from the model. The optimal parameters $\boldsymbol{\alpha}^*$ of the model ϕ are obtained according to

$$\boldsymbol{\alpha}^* = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \; C(\mathcal{D}_{train}, \phi_{\boldsymbol{\alpha}}). \tag{19}$$

The performance of the learned model $\phi_{\boldsymbol{\alpha}^*}$ is assessed on the test dataset $\mathcal{D}_{test}$.

4.2. Statistical learning models

This section discusses popular statistical learning techniques in credit scoring.

4.2.1. Linear discriminant analysis

The *Linear Discriminant Analysis (LDA)* develops a linear combination of independent features which yield the largest mean difference between classes. Under the assumption of equal class covariances [38], we have the following solution in Eq. (20) for a binary classification problem, such as credit scoring. For a given feature vector $\mathbf{x}$ we decide for class $\hat{y} = +1$ if the expression on the right hand side is greater than the expression on the left hand side, otherwise we decide for class $\hat{y} = -1$

$$(\boldsymbol{\mu}^{(+)} - \boldsymbol{\mu}^{(-1)})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \gtrless \log(P(y = -1)) - \log(P(y = +1)), \tag{20}$$

where $P(\cdot)$ denotes probability.

4.2.2. Logistic regression

The Logistic Regression (LR) is the most commonly used statistical model in credit scoring due to interpretability of its decisions. The LR model is defined as

$$P(y = +1|\mathbf{x}) = \frac{1}{1 + \exp(\alpha_0 + \boldsymbol{\alpha}^T \mathbf{x})} \tag{21}$$

and

$$P(y = -1|\mathbf{x}) = 1 - P(y = +1|\mathbf{x}) = \frac{\exp(\alpha_0 + \boldsymbol{\alpha}^T \mathbf{x})}{1 + \exp(\alpha_0 + \boldsymbol{\alpha}^T \mathbf{x})}, \tag{22}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the feature vector, $P(y = +1|\mathbf{x})$ is the probability of classifying $\mathbf{x}$ as a good borrower, $P(y = -1|\mathbf{x})$ is the probability of classifying $\mathbf{x}$ as a bad borrower and $\{\alpha_0, \boldsymbol{\alpha}\}$ are the model parameters estimated by using, e.g., maximum likelihood estimation on the training dataset [4]. Once the model parameters are estimated, the decision on an input feature vector $\mathbf{x}$ is made in favor of $\hat{y} = +1$ if

$$P(y = +1|\mathbf{x}) \geq P(y = -1|\mathbf{x}), \tag{23}$$

which is equivalent to the following decision rule

$$\hat{y} = \begin{cases} +1 & \text{for } 1 \geq \exp(\alpha_0 + \boldsymbol{\alpha}^T \mathbf{x}); \\ -1 & \text{otherwise.} \end{cases} \tag{24}$$

4.2.3. Naïve Bayes

The Naïve Bayes (NB) classifier is based on Bayes decision rule [39]. The adjective "naive" comes from the assumption that the features in a dataset are mutually independent. The Naïve Bayes classifier decides for class $y = +1$ over $y = -1$ if

$$P(y = +1|\mathbf{x}) \geq P(y = -1|\mathbf{x}), \tag{25}$$

where

$$P(y = +1|\mathbf{x}) = \frac{p(\mathbf{x}|y = +1)P(y = +1)}{p(\mathbf{x})} \tag{26}$$

and

$$P(y = -1|\mathbf{x}) = \frac{p(\mathbf{x}|y = -1)P(y = -1)}{p(\mathbf{x})}. \tag{27}$$

The probability density $p(\mathbf{x})$ of observing $\mathbf{x}$ is defined according to

$$p(\mathbf{x}) = p(\mathbf{x}|y = +1)P(y = +1) + p(\mathbf{x}|y = -1)P(y = -1) \tag{28}$$

and the conditional density functions $p(\mathbf{x}|y = -1)$ and $p(\mathbf{x}|y = +1)$ are modeled according to

$$p(\mathbf{x}|y = -1) = \frac{1}{(2\pi)^{\frac{m}{2}} |\boldsymbol{\Sigma}^{(-)}|^{\frac{1}{2}}} \exp\{-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}^{(-)})^T \mathbf{inv}\left(\boldsymbol{\Sigma}^{(-)}\right)(\mathbf{x} - \boldsymbol{\mu}^{(-)})\} \tag{29}$$

and

$$p(\mathbf{x}|y = +1) = \frac{1}{(2\pi)^{\frac{m}{2}} |\boldsymbol{\Sigma}^{(+)}|^{\frac{1}{2}}} \exp\{-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}^{(+)})^T \mathbf{inv}\left(\boldsymbol{\Sigma}^{(+)}\right)(\mathbf{x} - \boldsymbol{\mu}^{(+)})\}, \tag{30}$$

where $\boldsymbol{\Sigma}^{(-)}$ and $\boldsymbol{\Sigma}^{(+)}$ are covariance matrixes computed on negative and positive instances from the training dataset. For the Naïve Bayes classifier, $\boldsymbol{\Sigma}^{(-)}$ and $\boldsymbol{\Sigma}^{(+)}$ are diagonal matrices.

Note that $p(\mathbf{x}|y = -1)$ and $p(\mathbf{x}|y = +1)$ are uni-modal multivariate Gaussian distribution density functions which may not adequately model multi-modal data. In this case, one can employ multi-variate *Gaussian Mixture Model* [40] and Expectation Maximization (EM) [41] to learn the model parameters on the training dataset.

4.3. Machine learning models

This section discusses popular machine learning techniques in credit scoring.

4.3.1. k-Nearest neighbor

A k-Nearest Neighbor (k-NN) [42] assigns to an input feature vector $\mathbf{x}$ the class of the majority of its k nearest neighbors in the training dataset. The nearest neighbors are determined by calculating the *Euclidean distance* or *Mahalanobis distance* between the input feature vector $\mathbf{x}$ and the training dataset $\{\mathbf{x}_k\}_{k=1}^m$. Thus the class for the new data point is

$$y = \underset{y \in \{+1,-1\}}{majority}\left[\underset{k}{argmin} \left\|\{\mathbf{x}_k\}_{k=1}^m - \mathbf{x}\right\|\right]. \tag{31}$$

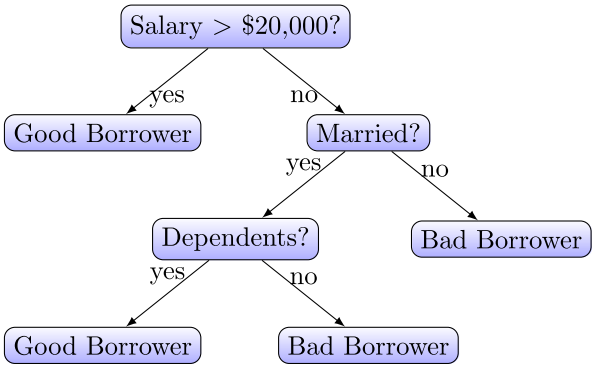

Fig. 4. An example of a decision tree in credit scoring.

4.3.2. Decision trees

As shown in Fig. 4 a Decision Tree (DT) [39] asks a series of questions in order to arrive to an answer (which is a class label). The root of the Decision Tree is called the *root node* and it is the most discriminating feature. The *leaf nodes* denote the classes.

4.3.3. Support vector machines

A Support Vector Machine (SVM) [43] uses an idea of a hyperplane (which is a decision boundary) that separates classes in a high dimensional feature space. The linear SVM focuses on maximizing the margin $\|\boldsymbol{\alpha}\| = \sqrt{\sum_{i=1}^m \alpha_i^2}$ between the negative and positive hyperplanes. The correct class is assigned by using the following equation:

$$y = \begin{cases} +1, \ if \ b + \boldsymbol{\alpha}^T \mathbf{x} \geq +1 \\ -1, \ if \ b + \boldsymbol{\alpha}^T \mathbf{x} \leq -1. \end{cases} \tag{32}$$

where b is the bias. For non-linear cases, a kernel trick [44] is used to project features into a high dimensional space.

4.3.4. Artificial neural networks

An Artificial Neural Network (ANN) [45] is a system which is motivated by a biological neural network system. An ANN emulates the way in which a biological neural network of the brain processes information by means of interconnected neurons [2,46,47]. Typically, a neural network consists of three layers, namely an input, hidden and output layers [48]. Essentially, training a neural network involves the process of finding optimal weights that map the input and output layers by means of *back-propagation*.

For a given input feature vector $\mathbf{x}$, a three-layer ANN computes the output $\hat{\mathbf{y}}$ according to

$$\hat{\mathbf{y}} = a_2(a_1(\boldsymbol{\alpha}^{(1)}\mathbf{x} + \alpha_0^{(1)})\boldsymbol{\alpha}^{(2)} + \alpha_0^{(2)}), \tag{33}$$

where $(\alpha_0^{(1)}, \boldsymbol{\alpha}^{(1)})$, $(\alpha_0^{(2)}, \boldsymbol{\alpha}^{(2)})$ are weights and a_1, a_2 are activation functions between input and hidden layer, hidden layer and output layer, respectively. The parameters are learned on a training set. The ANN performs final decision by applying a decision function, such as soft-max, on $\hat{\mathbf{y}}$. The ANN was first applied in credit scoring by Odom and Sharda [49].

4.3.5. Random forests

A Random Forest (RF) is an ensemble of decision trees [50], i.e. K decision trees are built on bootstrapped samples with m observations. Each decision tree is developed using a subset of randomly chosen k features. Each decision tree will give a class of a new feature vector. Thereafter, for overall classification, a RF assigns the class of the new feature vector by using majority vote based on the outputs from the decision trees.

4.3.6. Boosting

The Boosting works by estimating multiple models iteratively and assigning weights to data instances [51]. Boosting starts by developing a weak model (i.e. shallow decision tree). Thereafter a better model that will address errors of the previous model is developed. The instances which were incorrectly classified by the previous model will be assigned higher weights. The popular boosting technique is *Adaptive Boosting* (Ada-boost). The Ada-boost assigns a class to an input feature vector $\mathbf{x}$ in the following way

$$\hat{y} = sign\left(\sum_{t=1}^{T} \alpha_t \phi_t(\mathbf{x})\right), \tag{34}$$

where α_t is the weight of classifier $\phi_t(\mathbf{x})$ and T is the total number of classifiers. The parameters α_ts are learned on the training dataset.

4.3.7. Extreme gradient boost

The XGBoost [52] is short for extreme gradient boosting. The XGBoost is famously known for its processing speed and performance. It works similar to gradient boosting but builds the decision trees in parallel instead of building the decision trees in a series [53]. The optimization function to minimize is

$$\mathcal{L}^{(t)} = \sum_{k=1}^{n} l(y_k, \hat{y}_k^{(t-1)} + \phi_t(\mathbf{x}_k)) + \Omega(\phi_t) \tag{35}$$

where $l(\cdot)$ is a loss function and $\Omega(\phi_t)$ is a regularization term that penalizes complexity of the model. The goal of XGBoost is to find the ϕ_t which minimizes the objective function $\mathcal{L}^{(t)}$ [54].

4.3.8. Bagging

The Bagging classifier is also known as the *bootstrap aggregation* [55]. The Bagging technique takes K bootstraps from the underlying datasets and builds a classifier for every bootstrap. Then a class label is assigned by using a majority vote from votes of the K classifiers

$$y = \underset{y\in\{+1,-1\}}{argmax} \sum_{i=1}^{K} 1\left(y = \phi_i(\mathbf{x})\right), \tag{36}$$

where

$$1\left(y = \phi_i(\mathbf{x})\right) = \begin{cases} 1, & if\ y = \phi_i(\mathbf{x}); \\ 0, & if\ y \neq \phi_i(\mathbf{x}). \end{cases} \tag{37}$$

4.4. Deep learning models

Deep learning algorithms have been successfully applied in literature since the 1980s in an attempt to improve classification accuracy. Moreover, deep learning models with optimal hidden layers have been developed to reveal information not easily detectable with traditional statistical and machine learning models. The following subsections highlight the deep learning models that are used in credit risk datasets.

4.4.1. Restricted Boltzmann machines

As shown in Fig. 5, a Restricted Boltzmann Machine (RBM) can be perceived as an undirected neural network with two layers. The two layers can be called the hidden and the visible layers. The hidden layer is used as a feature detector while the visible layer is used to train the input data [56]. Given n visible layers $\mathbf{v}$, and m hidden layers $\mathbf{h}$, the energy function is

$$E(v, h) = -\sum_{i=1}^{n} a_i v_i - \sum_{j=1}^{m} b_j h_j - \sum_{i=1}^{n}\sum_{j=1}^{m} \alpha_{ij} v_i h_j, \tag{38}$$

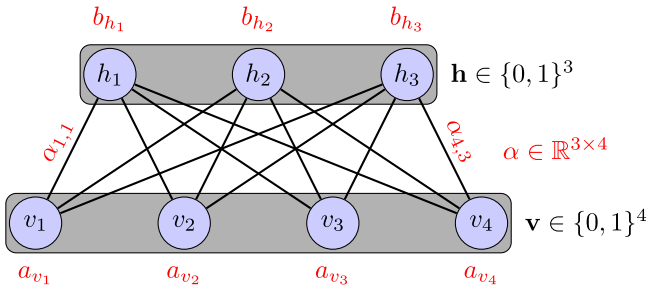

Fig. 5. A restricted Boltzmann machine architecture with a visible layer $\mathbf{v}$ and a hidden layer $\mathbf{h}$.

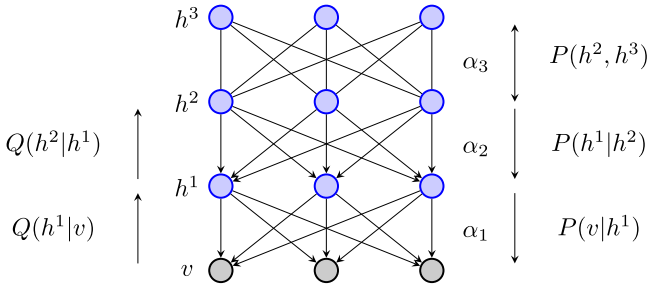

Fig. 6. A deep belief neural network architecture which is composed of several hidden layers of restricted Boltzmann machines.

where a_i and b_j are biases for binary variables v_i and h_j respectively, while α_{ij} are the weights between unit i and j.

4.4.2. Deep belief neural networks

Deep Belief Networks (DBNs), introduced by Hinton et al. [57], is a class of deep neural networks. As shown in Fig. 6 a typical DBN is composed of several hidden layers of RBM. Essentially, an output of a lower level RBM can be perceived as input of the higher level RBM. Fig. 6 shows a graphical view of the DBN layers.

4.4.3. Convolutional neural networks

Convolutional Neural Networks (CNNs) were first introduced by LeCun et al. [58] and have been mainly used in image processing [59–61], and on time series data [60,62,63]. The convolutional neural network consists of an *input layer*, *convolutional layers*, *pooling layers* and *fully connected layers* (see Fig. 7). The convolutional and pooling layers are responsible for extracting data representations [64].

Input. A convolutional neural network takes inputs as *tensors* of shape (*height*, *width*, *channel*). In image classification, the height and width values represent the height and width of an image. The channel represents the depth/color of an image (e.g. 1 represents a gray-scale image and 3 represents an image with color). A tensor is a multidimensional array that contains numerical values or in some instances non-numeric values. The input shape is normally changed into a shape that the convolutional neural network anticipates and the input is scaled so that all values are in the [0, 1] interval [64]. Since credit scoring data is not an image data, a 1*D* convolutional neural network is normally used for non-image data with the exception of speech/voice data. Hence, for credit scoring, the CNN architecture consists of an input layer which is a tensor of shape $(m, 1, n)$, where m is the number of instances and n is the number of features. Each instance is a feature vector with three channels that has a shape $(1, 1, n)$. All inputs are scaled into [0, 1] interval.

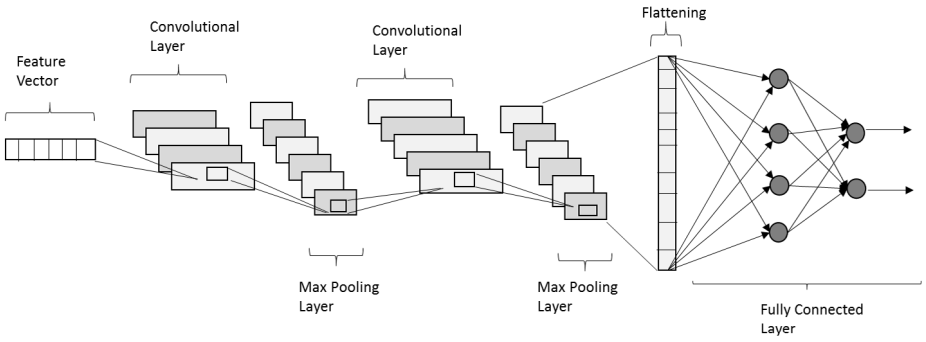

Fig. 7. A 1*D* convolutional neural network architecture which consists of two convolutional layers, two pooling layers and a fully connected layer.

Convolutional layer. The word "convolution" refers to a mathematical operation which is a specialized kind of linear operation [65]. Below is a convolution function

$$(f * g)(t) = \int_{-\infty}^{\infty} f(x)g(t-x)dx. \tag{39}$$

A convolutional layer learns local patterns [64], for example, in image classification, a convolutional layer breaks down an image into edges and textures. These local patterns are *feature maps* and are also known as *activation maps*. A convolution operation is applied on the input tensor and the *feature detector*. The feature detector also known as the *filter* or *kernel* is a tensor of shape (*height*, *width*). The feature detector slides along different locations of an input tensor to form a feature map. The feature map is a reduced and transformed input tensor. A non-linear activation function such as *ReLU* (Rectified Linear Unit) or a *Sigmoid* function is applied on activation maps to introduce non-linearity.

Pooling layer. A pooling layer is responsible for *downsampling* feature maps. At pooling layer, either a *max pooling* or an *average pooling* is used. For max pooling, each local input location is transformed by taking the maximum value of each channel over the location, whereas for average pooling, an average value of each channel is used. A convolutional neural network has a property called *spatial invariance*, which occurs at a pooling layer. The spatial invariance assures that the network is not influenced by input distortions or variations. Thus, the pooling layer is capable of preserving important features.

Flattening. The pooled feature maps are then converted into a single vector by a process called *flattening*. This single vector becomes an input to a fully connected artificial neural network.

Fully connected layer. A Fully Connected Layer is a feed forward artificial neural network. It consists of an input layer, hidden layer/s and an output layer. At output layer, a *loss function* or *cost function* is defined as follows

$$C(y, \hat{y}) = \sum_{\forall i} d(y_i, \hat{y}_i), \tag{40}$$

where $y_i, \hat{y}_i \in \{+1, -1\}$ and $d(y_i, \hat{y}_i)$ is a measure between y_i and $\hat{y}^{(i)}$ such as the Mean Squared Error(MSE) or Cross-Entropy. The mean squared error is

$$\text{MSE} = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2, \tag{41}$$

and the cross-entropy is

$$H(y, \hat{y}) = -\sum_{i=0}^{1} y_i \log \hat{y}_i. \tag{42}$$

Table 2
A confusion matrix with correct predictions, True Positives (TP) and True Negatives (TN), and incorrect predictions, False Positives (FP) and False Negatives (FN).

		Predicted	
		Positives	Negatives
Actual	Positives	TP	FN
	Negatives	FP	TN

A neural network tries to minimize the cost function by learning and updating the connection weights using *back propagation* [66].

4.4.4. Deep Multi-Layer Perceptron

A Multi-Layer Perceptron with multiple hidden layers is called Deep Multi-Layer Perceptron. Deep Multi-Layer Perceptron is a directed neural network and training is performed by back-propagation. The cost or loss function for Deep Multi-Layer Perceptron uses *Softmax* and Cross-Entropy to update the weights. The Softmax function is

$$f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}} \tag{43}$$

and $f_j(z) \in [0, 1]$ and z_j is a probability output from the network.

5. Evaluation metrics

There are many evaluation metrics which are used in literature and the following metrics are the most popular metrics for assessing the performance of the models in credit scoring. The *Percentage Correctly Classified* (PCC), *Type I Error*, *Type II Error*, *Kolmogorov–Smirnov Statistic* (K-S), *Sensitivity/Recall*, *Specificity*, *Geometric-Mean*(G-mean), *F-measure* and *Area Under Receiver Operating Characteristics Curve* (AUC). Each of these metrics is shown in the sequel.

A *confusion matrix* (see Table 2) consists of *True Positives (TP)*, *True Negatives (TN)*, *False Positives (FP)* and *False Negatives (FN)* and is used for calculating the metrics which are discussed in this section. Based on credit scoring classification, TN is the number of borrowers who are correctly classified as non-defaults, FP is the number of non-defaulted borrowers who are incorrectly classified as defaults, FN is the number of defaulted borrowers who are incorrectly classified as non-defaults and TP is the number of borrowers who are correctly classified as defaults.

From the confusion matrix the performance metrics can be derived, such as

$$\text{PCC} = \frac{(TP + TN)}{(TP + FP + FN + TN)}, \tag{44}$$

$$\text{Type I} = \frac{(FP)}{(FP + TN)}, \tag{45}$$

$$\text{Type II} = \frac{(FN)}{(TP + FN)}, \tag{46}$$

$$\text{Sensitivity/Recall} = \frac{TP}{(TP + FN)}, \tag{47}$$

$$\text{Specificity} = \frac{TN}{(FP + TN)}, \tag{48}$$

$$\text{G–mean} = \sqrt{\frac{TN}{(FP + TN)} \times \frac{TP}{(TP + FN)}}, \tag{49}$$

$$\text{F–measure} = \frac{2 \times Recall \times Precision}{Precision + Recall}. \tag{50}$$

The Kolmogorov–Smirnov Statistic

$$\text{K–S} = max|P(s|G) - P(s|B)| \tag{51}$$

calculates the maximum distance between the cumulative score distribution of bads $P(s|B)$ and goods $P(s|G)$

$$P(s|B) = \sum_{t \leq s} p(t|B) \tag{52}$$

and

$$P(s|G) = \sum_{t \leq s} p(t|G) \tag{53}$$

respectively and where $s \in \mathbb{Z}^+$ denotes a score in $500 \leq s \leq 1000$.

The AUC [67] is

$$\text{AUC} = \frac{1}{2}\left(1 + \frac{TP}{TP + FN} - \frac{FP}{FP + TN}\right) \tag{54}$$

and measures the discriminative power of a model between classes.

6. Data imbalance

Imbalanced datasets occur as the number of observations in one class (referred to as a minority class) in a dataset is usually much lower than the number of observations in the other class (referred to as a majority class). There are studies that have dealt with imbalanced datasets. For example, Brown and Mues [15] showed that the random forest and gradient boosting classifiers perform very well in a credit scoring context and are able to cope comparatively well with pronounced class imbalances in the datasets. On the other hand, when faced with a large class imbalance, the C4.5 decision tree algorithm, quadratic discriminant analysis and k-nearest neighbors perform significantly worse than the best performing classifiers. Douzas and Bacao [68] tackle the problem of imbalanced datasets by using a novel oversampling method, Self-Organizing Map-based Oversampling (SOMO).

There are a number of over-sampling (applied on minority class) or under-sampling techniques (applied on majority class) that can be found in literature. For example, Chawla et al. [69] proposed Synthetic Minority Over-sampling Technique (SMOTE). The SMOTE over-samples the minority class by taking each minority class sample and creating synthetic examples (along the line segments joining any/all of the k minority class nearest neighbors). Thereafter, neighbors from the k nearest neighbors are randomly chosen, depending on the amount of over-sampling required. For instance, if the amount of over-sampling needed is 300%, only three neighbors are chosen and one sample is generated in the direction of each. Synthetic samples are generated by taking the difference between the feature vector (sample) under consideration and its nearest neighbor. Thereafter this difference is multiplied by a random number between 0 and 1, and is added to the feature vector under consideration to form a synthetic feature vector.

There are other techniques that neither do over-sampling nor under-sampling to deal with class imbalance, such as *wavelet data transformation* and *linear dependence approach*. Saia et al. [70] proposed a discrete wavelet transformation to deal with imbalanced data in credit scoring. Wavelets are small waves and wavelet transform captures both the time and frequency domains [71]. The discrete wavelet transform disintegrates a signal wave into a set of wavelets that is mutually orthogonal [71]. Saia et al. [70] approach outperformed the random forest model regardless of data distributions. Saia and Carta [72] proposed a linear dependence approach for imbalanced data. The idea was to exploit one class (the majority class) to overcome imbalanced class distribution issue. The linear dependence is determined by calculating the determinant of a square and non-square matrices. The determinant is a real number that is calculated from a matrix. When vectors are dependent, their determinant is zero. Saia and Carta [72] used an average of sub-matrix determinants and reliability band of the majority class to classify new instances. The proposed approach performed very similarly to the random forest model.

7. Model transparency

This section discusses different model transparency techniques. The aim of model transparency is to make non-transparent models explainable. In a case of credit scoring, this will help in explaining why a borrower was not granted a loan. In the following, we present the most commonly used model transparency techniques in credit scoring.

7.1. NeuroRule

The NeuroRule was first introduced by Setiono and Liu [73]. The NeuroRule uses three-layer feed-forward neural network. The NeuroRule consists of six steps:

(Step-1) Build and train a neural network;
(Step-2) Prune the neural network to remove irrelevant connections;
(Step-3) Discretize the hidden unit activation values of the pruned neural network by clustering;
(Step-4) Extract rules that describe the network outputs in terms of the discretized hidden activation values;
(Step-5) Extract rules that describe the discretized hidden unit activation values in terms of the network inputs;
(Step-6) Combine the two sets of rules extracted in steps 4 and 5 to obtain a final set of rules that relate the inputs and outputs of the network.

7.2. Trepan

Trepan was first introduced by Craven and Shavlik [74]. Trepan is an hybrid intelligent system. It [Trepan] induces a tree to approximate any classifier's predictions. Hence, Trepan is not restricted to neural networks. Trepan learns queries as opposed to normal decision trees which learn from data. At each node, Trepan stores (i) a subset of training observations, (ii) a set of *query instances* and (iii) a set of constraints (the conditions that observations must satisfy in order to reach the node).

7.3. LIME

LIME stands for Local Interpretable Model-Agnostic Explanations. LIME is capable of explaining a prediction of any classifier

by learning an interpretable model (e.g. a Decision Tree or a linear model) around a prediction [75]. The logic behind explaining predictions is for human subjects to have trust in the predictions if actions need to be taken. For example, if a doctor depends on a model to predict presence/absence of any disease, he/she will need to trust the model predictions in order for him/her to prescribe medication for patients. The doctor has prior knowledge in the field of medicine, he/she will either accept or reject the explanation based on his/her expertise. Hence, human prior knowledge plays an important role for accepting or rejecting explanations for predictions. LIME is designed to provide *trust* to human subjects.

7.3.1. Interpretable data representation

Before acquiring explanations, a data set needs to be in a format that is understandable to humans. This is applicable, e.g., in image classification or text classification, where features in the input space need to be transformed to a vector of ones and zeros to indicate the presence or absence of feature components. An observation $\mathbf{x} \in \mathbb{R}^n$ is transformed into $\hat{\mathbf{x}} \in \{0,1\}^{n'}$ for interpretability.

7.3.2. Local fidelity

The behavior of a classifier in the vicinity of an individual prediction determines the local faithfulness of an explanation. Let an explanation be denoted by $e \in E$ where E is a space of interpretable models, e.g., decision trees or linear models. The complexity of an explanation is given by $\Omega(e)$, e.g., for decision trees the complexity is the depth of a tree. Let $\phi : \mathbb{R}^n \rightarrow \mathbb{R}$ be a classifier which needs to be explained and let $\pi_{\mathbf{x}}(\mathbf{y})$ be a proximity measure between observation $\mathbf{y}$ to observation $\mathbf{x}$ which defines the locality around $\mathbf{x}$. The measure of how unfaithful is e in explaining $\phi(\mathbf{x})$ in the locality given by $\pi_{\mathbf{x}}(\mathbf{y})$ is

$$\psi\{\phi, e, \pi_{\mathbf{x}}\}, \tag{55}$$

where

$$\pi_{\mathbf{x}}(\mathbf{y}) = exp\left(-D(\mathbf{x},\mathbf{y})^2/\sigma^2\right) \tag{56}$$

is an exponential kernel defined on some distance D (e.g. Mahalanobis distance).

To obtain interpretability and local fidelity, an optimization is performed to minimize equation Eq. (55) and $\Omega(e)$ is kept low. Hence, the prediction explanation for observation $\mathbf{x}$ is

$$\zeta(x) = \underset{e \in E}{\operatorname{argmin}} \psi\{\phi, e, \pi_{\mathbf{x}}\} + \Omega(e). \tag{57}$$

7.3.3. Model agnostic

For explanations to be model-agnostic, we do not need to make assumptions about ϕ. This will ensure that any prediction of *any* black-box model can be explained.

7.3.4. Sampling

The Eq. (55) is approximated by drawing samples which are weighted by $\pi_{\mathbf{x}}$ around an observation of interest $\mathbf{x}$. The sampling is done by drawing non-zero elements of $\mathbf{x}$ uniformly at random [75]. Then $\hat{\mathbf{x}} \in \{0,1\}^{n'}$ is a perturbed sample which contains non-zero elements of $\mathbf{x}$. The perturbed data set is denoted by $\Re$. The outcome/label prediction for $\hat{\mathbf{x}}$ is given by $\phi(\mathbf{x})$.

7.3.5. Sparse linear explanations

This is a part where predictions are explained. In this stage an *explainer* is a linear model. Hence, $e(\hat{\mathbf{x}})$ is a linear model

$$e(\hat{\mathbf{x}}) = \boldsymbol{\alpha}^T \hat{\mathbf{x}}. \tag{58}$$

Let $\mathbf{x}$ be the original representation of $\hat{\mathbf{x}}$. Then the Eq. (55) becomes a locally weighted square loss

$$\psi\{\phi, e, \pi_{\mathbf{x}}\} = \sum_{\mathbf{x}, \hat{\mathbf{x}} \in \Re} \pi_{\mathbf{x}}(\Phi(\mathbf{x}) - e(\hat{\mathbf{x}}))^2. \tag{59}$$

8. Model limitations and imposed assumptions

The Linear Discriminant Analysis (LDA) technique is one of the two popular statistical techniques applied in credit scoring, and is subject to a parametric assumption which is defined as the underlying multivariate normal distribution of the features. Eisenbeis [76] argues that categorical features violate this parametric assumption. The other model which uses two strong statistical assumptions (i.e. i. the assumption that the features are conditionally independent ii. the values of numeric features are normally distributed) is the Naïve Bayes Classifier. The normally distributed assumption in Naïve Bayes classifier does not always hold in other domains, hence a need to estimate other continuous distributions is required [77].

The most popular statistical technique in credit scoring is Logistic Regression (LR). The Logistic Regression assumes a linear relationship between the inputs and the log odds. However, the linearity assumption does not always hold, there are other cases where the relationship between the independent variables and the log odds is non-linear. In such cases kernel Support Vector Machines (SVM) serves as one of the non-linear classification techniques to be used. However, Yu et al. [78] argues that the performance of SVM model is sensitive not only to the algorithm for solving the Quadratic Programming problem but also to the parameters setting in learning (i.e. regularized parameter balancing the classification margin and tolerable misclassification errors, and the kernel parameter). Yu et al. [78] propose Least Squares SVM (LSSVM) to solve the quadratic programming problem and the design of the experiment for parameter selection in SVM modeling.

Henley and Hand [42] propose the use of a non-parametric technique, k-nearest neighbor method in credit scoring. However, the k-nearest neighbor method is costly because it requires more computations (i.e. it calculates a metric distance for each data record stored during classification). Jiang [79] proposes the use of a decision tree (C4.5) in conjunction with an approach called Simulating Annealing Algorithm (SAA) which performs global optimization. In this study, Jiang highlights the shortcoming of the Decision Tree approach which is the inability to perform global optimization because of its local search strategy, hence the need to use SAA. Albeit the ANNs better performance when compared to other techniques in terms of accuracy, it lacks interpretability [5]. Baesens et al. [5] propose Decompositional and Pedagogical approaches to extract rules (without compromising the accuracy) from the ANN for interpretability.

Deep Learning models have not been applied extensively in credit scoring. However, the problem with deep learning models is interpretability. Since banks are governed by regulators, banks are required to be transparent in their credit scoring process. A bank needs to tell a borrower why his/her loan application has been rejected.

9. Emerging trends

Most banks have developed an interest in applying machine learning models (including deep learning modes) in their credit scoring systems. However, the regulators still maintain that the models which are used for credit scoring should be transparent. Despite their non-transparency, machine learning models are gaining traction in credit scoring. One way of mitigating the non-transparency issue of machine learning models is to rationalize

Table 3
Datasets that are used in literature for credit scoring. Each dataset has a number of good borrowers and a number of bad borrowers.

Dataset	Sample size	#Goods	#Bads	# of features
European Credit Bureau	186,574	179,544	7,030	324
UK	30,000	28,800	1,200	14
Barbados	21,117	20,614	503	20
Indonesia	14,700	14,290	4,410	31
Benelux 2 (Belgium, Netherlands, and Luxembourg)	7,190	5,033	2,157	28
Brazilian	4,504	4,144	360	5
Benelux 1 (Belgium, Netherlands, and Luxembourg)	3,123	1,040	2,083	27
University of California, San Diego	2,435	1,836	599	38
China	1,057	552	505	10
German	1,000	700	300	20
Iranian	1,000	950	50	27
Australian	690	307	383	14
Japanese	653	296	357	15
Polish	240	128	112	30
Texas Banks	162	81	81	19

(i.e. give justification as to why a certain decision has been made) the predictions.

9.1. Transparency

Methods that involve *rule extractions* for opening-up non-transparent models have been suggested in the literature. Baesens et al. [5] evaluated and contrasted three neural network rule extraction techniques, namely, Neurorule, Trepan, and Nefclass for credit-risk evaluation. They concluded that neural network rule extraction is an effective and powerful management tool which allows the development of advanced and user-friendly decision-support systems for credit-risk evaluation. Setiono and Liu [80] proposed a NeuroLinear method for extracting oblique decision rules from neural networks. The experimental results showed that NeuroLinear is effective in extracting compact and comprehensible rules that have high predictive accuracy from neural networks.

The techniques highlighted above are based on rules, and recently techniques such as *SHAP values*, *Partial Dependence* and *Explainable Boosting Machines* that do not require rules have been applied in domains (other than credit scoring). In their empirical study [81] proposed intelligible model that can easily explain its predictions. The study applied generalized additive models with feature interactions (also referred to as explainable boosting machines) on real health care problems. Their results showed that the proposed generalized additive model can perform comparably with best performing machine learning models such as random forest and logistic regression. Lundberg and Lee [82] used SHAP (SHapley Additive exPlanations) as a unified framework to interpret predictions. The SHAP method assesses contribution of each feature towards a prediction. On the other hand, the partial dependence checks how the prediction changes when different feature values are used. The partial dependence plots show the marginal impact one or two features have on the prediction of a machine learning model [83].

9.2. Deep learning in credit scoring

This section covers deep learning techniques in credit scoring. There is an emerging trend to replace statistical and classical machine learning techniques with deep learning techniques in credit scoring. For instance, Luo et al. [84] used Corporate Default Swaps (CDS) data to compare performances of deep belief networks with well-known credit scoring models such as logistic regression, multi-layer perceptron and support vector machine. Deep belief networks showed better performance. In their study, Ramasamy and Rajaraman [85] compared meta-cognitive restricted Boltzmann machine with extreme learning machines, support vector machines and multi-layer perceptron using Australian, German and Kaggle credit datasets. The meta-cognitive restricted Boltzmann machine showed superior performance. Tomczak and Zieba [67] assessed and compared performances of classification restricted Boltzmann machine with several traditional statistical and machine learning models, such as logistic regression, decision trees, adaboost, random forest etc., using German, Australian, Kaggle and Short-Term Loans data. The classification restricted Boltzmann machine showed comparable results on German, Kaggle and Short-Term Loan datasets, and better results on Australian dataset. Tran et al. [86] proposed a hybrid genetic programming and stacked auto-encoder network model. The proposed hybrid model was compared to logistic regression, $k-$NN, support vector machines, artificial neural networks and decision trees using German and Australian credit datasets. The hybrid model showed a better accuracy rate. Yeh et al. [87] used daily stock returns to predict defaults and non-defaults using deep belief network and support vector machines. The proposed deep belief network outperformed support vector machines. In their empirical study, Neagoe et al. [88] compared deep convolutional neural networks with multi-layer perceptron using German and Australian datasets. The results showed a superior overall accuracy rate for deep convolutional neural networks. These studies have shown the superiority of deep learning models in credit scoring. However, Hamori et al. [89] argue that the performance of deep learning models is dependent on the choice of activation function, the number of hidden layers and the dropout rate. The results in [89] showed a better performance for ensemble methods, such as boosting and bagging, when compared with deep neural networks using Taiwan credit dataset. These studies highlight and reiterate the applicability of deep learning algorithms to credit scoring data.

9.2.1. Data augmentation

Deep learning models require more data for training to avoid overfitting [90]. Data augmentation is normally performed to increase the number of training data points. This is done by applying several distortions to the original training images, such as changing the brightness and rotation of images and the distortions should not change the spatial pattern of target classes [91]. In [91], several data augmentations such as *random shift*, *random zoom*, *random horizontal flip* and *random rotation* were applied on image data. The random shift randomly shifts the images by a factor, the random zoom randomly zooms the images by a certain range, the random rotation randomly rotates the images by a certain angle and the random horizontal flip randomly flips the images horizontally to produce additional images. Their results [91] showed that data augmentation did not significantly improve the accuracy (i.e. accuracy increased by 1%). Kvamme

et al. [92] argue that data augmentation has been mostly done in image processing, and does not necessarily generalize well to other data sources.

On the other hand, Krizhevsky et al. [93] achieved significant improvements on error rates when data augmentation was applied on *ImageNet* dataset. A huge deep neural network (60 million parameters and 650,000 neurons) was used for this task, and for this they needed to increase the dataset size. Perez and Wang [94] used data augmentation on *gold fish vs. dogs* and *cats vs. dogs* datasets. The results showed better improvements in terms of accuracy on both datasets. Salamon and Bello [95] combined deep learning neural network with data augmentation to classify *audio* data. They used four different data augmentations and the proposed combination significantly outperformed deep neural network without augmentation. Frid-Adar et al. [96] proposed a combination of classic data augmentation with *Generative Adversarial Network (GAN)* data augmentation. The GAN was used to synthesize new *images of liver lesion*. This combination resulted in significant accuracy improvement. All of these studies used image datasets and credit scoring data is not in the form of images. However, the credit scoring data can be converted into images [97].

The above authors have contrasting views on the effectiveness of data augmentation on model accuracy. The conclusion from Krizhevsky et al. [93] is that when data augmentation is used in conjunction with large deep neural networks, the accuracy increases significantly compared to using data augmentations with "shallow" neural networks. Also Salamon and Bello [95] found that class accuracy is influenced differently by each augmentation.

10. Limitations in credit scoring literature

Below is the list of limitations which have been partially or completely ignored in credit scoring literature:

- ✓ No inclusion of macro-economic variables;
- ✓ The time it takes for borrowers to default is not determined;
- ✓ Exploratory Data Analysis: detection of outliers and distribution of variables (checking zero-variance for variables) is not performed;
- ✓ Time/Model Complexity is not factored;
- ✓ Creation of new features instead of performing feature space reduction techniques/feature selection is not taken into account;
- ✓ Correlation between dependent variable (or target variable) and independent variables is not assessed.
- ✓ Most studies in literature focus on homogeneous base classifiers and ignore heterogeneous base classifiers for ensemble methods;
- ✓ Using different cut-offs (instead of 0.5) to classify borrowers as either non-defaults or defaults is not covered in literature;
- ✓ Few studies incorporate Type II error. The type II error is more costly in credit scoring, since Type II (False Negative ratio) predicts a borrower as a good borrower but in actual fact he or she is a bad borrower.

The increases in macro-economic variables such as interest rate, inflation rate and unemployment rate may increase the risk of a borrower defaulting. Hence, it is key to incorporate macro-economic variables in credit scoring. Since macro-economic variables are time-varying, it is key to develop forward-looking credit scoring techniques. The survival analysis can cater for forward-looking credit scoring techniques. This can help not only to classify a borrower as default but to know when will a borrower default. The univariate analysis can help in identifying non-predictive variables and also detect outliers. Model/time complexity stems from models with too many parameters. Hence, it is key to keep a few parameters during model development. Although feature reduction/selection improves model accuracy, the literature needs to look at other feature engineering techniques such as taking the sum/product of two features to create a new feature. In cases where there are few independent features, a correlation between the target variable and the independent features can be performed to determine the high predictive features. The literature needs to focus on other ensemble techniques where base classifiers/learners are heterogeneous (i.e. models coming from different model classes). The default cut-off for classifying borrowers is 0.5, e.g. in logistic regression, the literature needs to assess the impact of using different cut-offs. The error which is of interest in credit scoring community is Type II error and the aim is to minimize this error. The literature needs to report more on this error when developing credit scoring models.

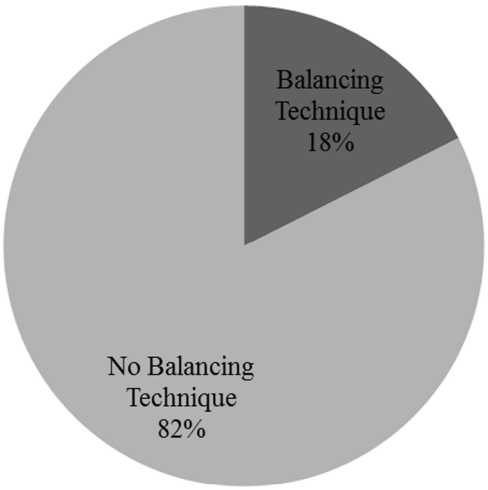

Fig. 8. A pie-chart that shows a proportion of literature papers that used data balancing techniques and a proportion of papers that did not balance their datasets.

11. Results

This section focuses on the meta-analysis of the results from primary studies. The frequency distributions of feature extraction techniques, the models and evaluation metrics from all primary studies are analyzed. The pie-charts for distribution of data balancing techniques and transparency techniques are also shown. The German and Australian credit datasets are selected for model performance comparison, as they are the most frequently used datasets in credit scoring.

11.1. Data balancing

Credit risk data is generally imbalanced (see Table 3). The non-default borrowers are usually more than the defaulted borrowers. In cases where there is a high imbalance in data, the models turn to be biased towards the majority class. To mitigate this bias, techniques such as over-sampling or under-sampling are usually performed. However, this study shows that only 18% of the primary studies in this literature survey have balanced their datasets (see Fig. 8). The most used technique is under-sampling the majority class (see Fig. 9).

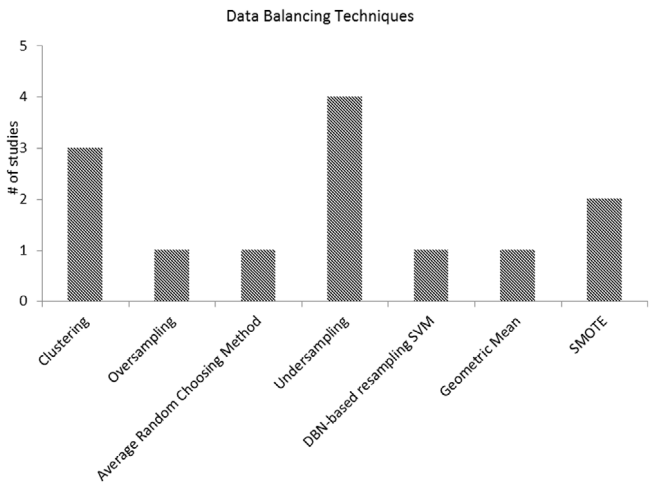

Fig. 9. A frequency distribution of data balancing techniques that are used in primary studies of this literature survey.

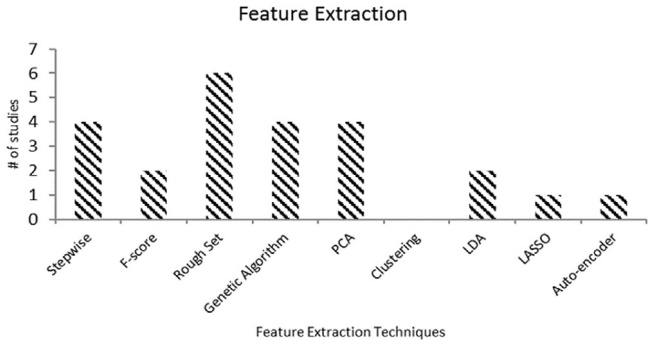

Fig. 10. A frequency distribution of feature extraction and engineering techniques that are used in primary studies of this literature survey.

11.2. Feature extraction

The distribution of feature extraction techniques, such as feature selection and feature engineering for this literature survey is shown in Fig. 10. This shows that out of 17 studies (see Table 4 for details) which used feature extraction, Rough Set technique was the most frequently used feature extraction technique, followed by Stepwise, Genetic Algorithm and PCA.

11.3. Evaluation metrics

Based on this current literature survey, Fig. 11 shows commonly used evaluation metrics in credit scoring. The most frequently used evaluation metrics are PCC and AUC (see Table 5 for details).

11.4. Models

This literature survey focuses on statistical, traditional and state-of-the-art machine learning models. Based on the results of this literature survey, Fig. 12 shows that LR, SVM and ANN were the most frequently used single classifiers (see Table 6 for details). For ensemble of classifiers, Boosting was the most frequently used ensemble classifier. The deep learning classifiers are among the least frequently used classifiers and this is attributable to the fact that deep learning classifiers are not applied extensively in credit scoring.

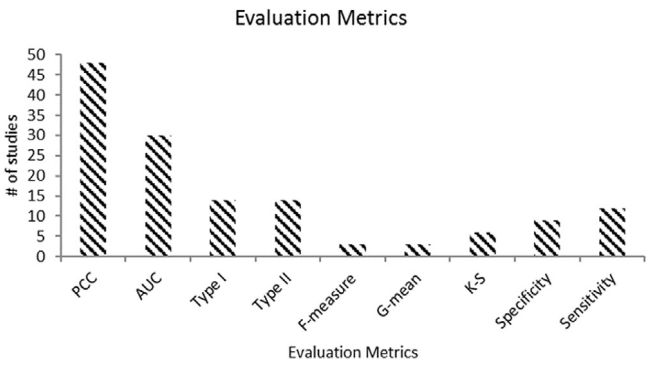

Fig. 11. A frequency distribution of evaluation metrics that are used in primary studies of this literature survey.

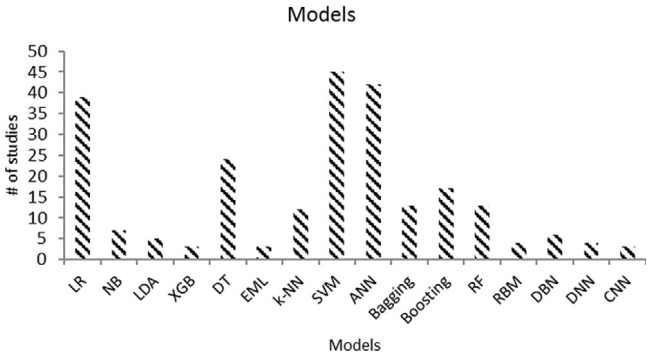

Fig. 12. A frequency distribution plot of most frequently used statistical and machine learning models in credit scoring.

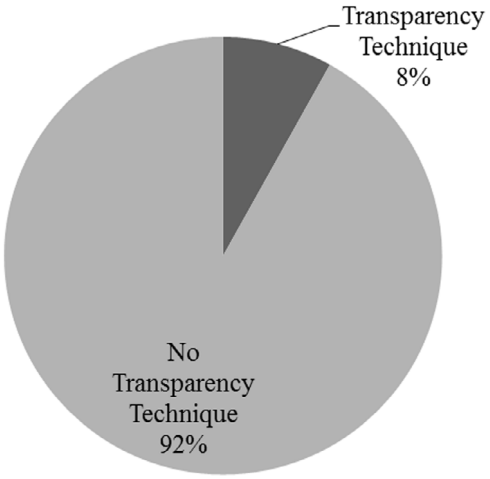

Fig. 13. A pie-chart that shows a proportion of studies that used transparency techniques and a proportion of studies that did not use transparency techniques.

11.5. Transparency

Under Basel II Accord [149], credit scoring models are required to be transparent (i.e. to explain their predictions). The majority of machine learning models are not transparent in nature. Hence, a need to make the models transparent in credit scoring is paramount. However, as can be seen in Fig. 13, this survey shows that only 8% of the primary studies have looked into techniques which make models transparent.

Table 4
Feature extraction methods used in primary studies. Legend: ST (Stepwise), FS (F-score), RS (Rough Set), GA (Genetic Algorithm), PCA (Principal Component Analysis), AE (AutoEncoder), LDA (Linear Discriminant Analysis), and LASSO (Least Absolute Shrinkage and Selection Operator).

Source	ST	FS	RS	GA	PCA	AE	LDA	LASSO
[17]		✓	✓					
[98]	✓						✓	
[25]			✓					
[99]			✓					
[100]					✓			
[101]	✓		✓					
[20]				✓				
[15]	✓							
[102]			✓					
[16]	✓							
[103]					✓			
[104]			✓					
[18]		✓						
[105]					✓			
[22]				✓				
[86]				✓				
[106]								✓
[107]								
[108]					✓	✓		
[109]							✓	
[110]				✓				

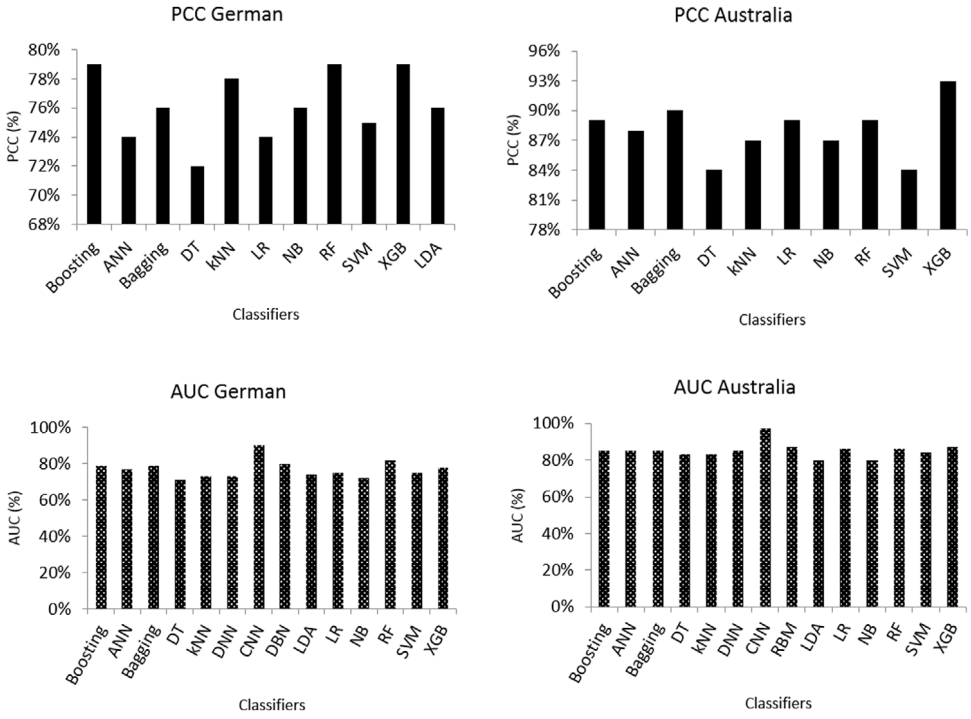

Fig. 14. Averages of pcc and auc that are calculated from the results reported on the papers considered for this literature survey that used German and Australian credit datasets.

11.6. Model performance on German and Australian datasets

Out of 74 primary studies, 39 primary studies either used German credit dataset, Australian credit dataset or both. The results for both PCC and AUC on both datasets were collected from the primary studies. The results (see Fig. 14) show that on average on both datasets, the Convolutional Neural Networks performed better in terms of PCC, followed by ensemble classifiers such as Random Forests, Bagging and Boosting. The high performance of CNN can be attributed to the fact that they can detect features that are more discriminative between borrowers. In terms of AUC, the ensemble of classifiers performed better than all single classifiers. We noted that different studies reported differently on PCC and AUC for each of the classifiers. This could be attributed to data pre-processing of each study.

12. Guiding machine learning framework for credit scoring

Based on the results of this literature survey, we propose a framework which is shown in Fig. 15, which serves as a guideline for credit scoring analysts. The nature of the data in credit scoring is either balanced or imbalanced. For both types of datasets, this framework suggests exploratory data analysis, data pre-processing, feature extraction techniques, the sampling methodology, the models to use as benchmarks, the best performing

Table 5
Evaluation metrics from primary studies. Legend: PCC (Percentage Correctly Classified), AUC (Area Under ROC Curve), T1 (Type I Error), T2 (Type II Error), F-M (F-measure), G-M (G-measure), K-S (Kolmogorov–Smirnov Statistic), SP (Specificity), SE (Sensitivity)

Source	PCC	AUC	T1	T2	F-M	G-M	K-S	SP	SE
[111]	✓								
[25]		✓			✓			✓	✓
[112]	✓								
[113]	✓								
[98]	✓								
[17]		✓							
[25]		✓			✓			✓	✓
[99]	✓								
[114]	✓		✓	✓					
[100]	✓								
[115]	✓		✓	✓					
[116]	✓							✓	✓
[117]	✓		✓	✓					
[118]			✓	✓				✓	✓
[114]		✓							✓
[101]	✓								
[119]		✓							
[20]		✓					✓		
[120]	✓		✓	✓					
[121]		✓	✓	✓					
[15]		✓							
[122]	✓								
[102]					✓			✓	✓
[16]		✓					✓		
[123]	✓							✓	✓
[104]	✓								
[18]	✓								
[103]	✓							✓	✓
[47]									
[124]		✓							
[22]	✓								
[105]	✓								
[125]		✓							
[126]	✓								
[127]		✓							
[67]		✓				✓		✓	✓
[128]	✓								
[87]	✓								
[129]									
[130]	✓	✓	✓	✓					
[131]		✓							
[132]	✓		✓	✓					
[133]		✓							
[86]	✓		✓	✓					
[134]	✓		✓	✓					
[135]	✓								
[46]	✓	✓	✓	✓					
[136]	✓	✓							
[137]	✓								
[107]		✓	✓	✓			✓		
[84]	✓	✓							
[138]	✓					✓			
[136]	✓		✓	✓					
[19]	✓								✓
[139]	✓							✓	✓
[85]						✓			✓
[140]	✓								
[141]	✓		✓	✓					
[106]		✓							
[142]	✓	✓							
[110]	✓	✓							
[143]							✓		
[92]	✓	✓						✓	✓
[108]		✓					✓		
[97]	✓	✓					✓		
[144]		✓							
[88]	✓								
[89]	✓	✓			✓				
[109]	✓								
[145]	✓	✓							
[56]	✓								
[146]	✓	✓							
[147]		✓							

Table 6
Models used in credit scoring. Legend: LR (Logistic Regression), NB (Naïve Bayes), LDA (Linear Discriminant Analysis), XGB (XGBoost), EML (Extreme Learning Machines), *k*-NN (*k*-Nearest Neighbor), SVM (Support Vector Machine), ANN (Artificial Neural Network), BA (Bagging), BO (Boosting), RF (Random Forest), RBM (Restricted Boltzmann Machine), DBN (Deep Belief Network), DMLP (Deep Multi-Layer Perceptron), and CNN (Convolutional Neural Network).

Source	LR	NB	LDA	XGB	DT	EML	*k*-NN	SVM	ANN	BA	BO	RF	RBM	DBN	DMLP	CNN
[111]	✓															
[98]	✓								✓							
[99]					✓					✓						
[112]	✓	✓			✓		✓		✓	✓	✓					
[113]								✓	✓	✓						
[25]	✓							✓	✓							
[17]								✓								
[114]								✓								
[100]								✓								
[115]	✓				✓			✓	✓	✓	✓					
[116]								✓								
[117]								✓					✓	✓		
[119]	✓				✓											
[114]	✓	✓	✓		✓		✓	✓								
[101]								✓								
[118]	✓				✓											
[20]	✓															
[120]	✓	✓			✓		✓	✓	✓	✓	✓					
[15]	✓		✓		✓		✓	✓	✓		✓	✓				
[121]	✓				✓		✓	✓	✓	✓	✓	✓				
[16]	✓				✓											
[123]	✓															
[122]								✓	✓							
[102]	✓				✓				✓							
[103]	✓							✓								
[18]								✓								
[104]								✓								
[18]								✓								
[47]					✓			✓	✓	✓	✓					
[124]					✓											
[105]								✓					✓			
[22]									✓							
[125]	✓							✓								
[126]								✓								
[127]	✓		✓					✓	✓		✓	✓				
[129]									✓							
[131]	✓		✓		✓		✓	✓	✓		✓	✓				
[128]								✓								
[87]	✓							✓						✓		
[67]	✓				✓			✓		✓	✓	✓	✓			
[132]	✓							✓	✓							
[130]	✓		✓		✓		✓	✓	✓		✓	✓				
[135]										✓						
[86]	✓				✓		✓	✓	✓		✓				✓	
[134]	✓				✓			✓	✓					✓		
[133]		✓			✓			✓	✓			✓				
[142]	✓				✓			✓	✓	✓	✓					
[141]	✓								✓							
[46]	✓							✓	✓	✓	✓	✓				
[136]	✓				✓	✓	✓	✓	✓	✓	✓					
[137]	✓										✓					
[140]		✓														
[106]								✓								
[107]	✓															
[84]	✓							✓	✓					✓		
[138]	✓							✓	✓						✓	
[136]	✓				✓	✓	✓	✓	✓							
[19]	✓															
[139]								✓						✓		
[85]						✓		✓	✓				✓			
[97]	✓											✓				✓
[144]	✓								✓		✓	✓			✓	
[88]									✓							✓
[109]		✓			✓		✓	✓	✓			✓				
[145]	✓			✓	✓			✓				✓				
[143]	✓															
[92]																✓
[108]																
[146]				✓												
[147]				✓												
[148]	✓		✓	✓	✓			✓	✓			✓				
[110]		✓					✓	✓								
[56]								✓						✓		
[89]									✓	✓	✓	✓			✓	

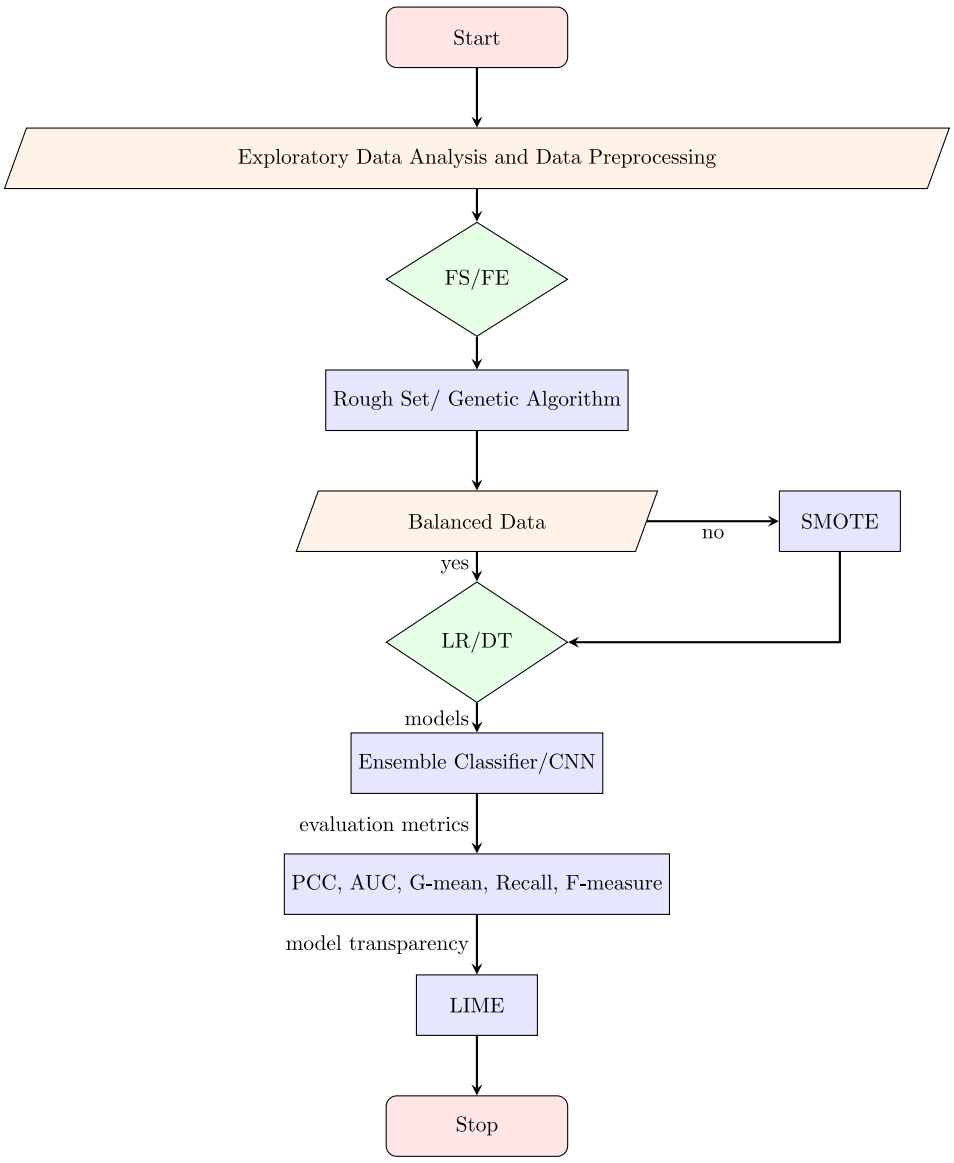

Fig. 15. The guiding machine learning credit scoring framework that is proposed in this literature survey.

models in credit scoring, the evaluation metrics and the technique which explains model predictions. This literature survey showed that the most frequently used techniques for feature extraction are Rough Set and Genetic Algorithm. The sampling methodology that we are suggesting for imbalanced datasets is SMOTE. The sampling is only applied on the training set. The models which can be used as benchmarks is Logistic Regression (LR) and Decision Trees (DT). The LR performs similarly to most traditional Machine learning models and the choice of selecting DT is its capability of explaining predictions. This literature survey shows that ensemble classifiers and CNN performed better when compared to other models, hence we suggest these two models for use in credit scoring. For imbalanced datasets, evaluation metrics which can be applied are G-mean, Recall and F-measure. Since predictions from ensemble classifiers and CNN cannot be explained, we suggest LIME for predictions' explanations.

13. Conclusion and future work

Many studies over the years have evaluated and contrasted the performances of different statistical and classical machine learning models in credit scoring. However, no consensus has been reached in identifying the best performing model. This literature survey systematically reviewed the most commonly used statistical, classical machine learning and deep learning models in credit scoring. The performances (on German and Australian credit datasets) of statistical, classical machine learning and deep learning models that were reported in literature were compared in this literature survey. The literature results showed that an ensemble of classifiers generally outperform single classifiers. Despite the minimal application of deep learning models in credit scoring literature, deep learning models such as convolutional neural networks showed better results compared to statistical and classical machine learning models. This literature survey also highlighted limitations in credit scoring literature. The credit scoring literature often ignores exploratory data analysis, omits inclusion of macro-economic variables and does not determine correlation between the target variable and the independent features, to mention a few. Furthermore, in this survey we proposed a guiding machine learning framework for analysts to perform credit scoring. This framework includes feature selection methods, data balancing technique, models to

consider for bench-marking, best performing models in credit scoring, relevant evaluation metrics and a method for making models transparent. This survey pointed to emerging directions in credit scoring such as the use of techniques that make model predictions explainable and the applications of deep learning models in credit scoring.

Future research should focus more on balancing classes of datasets in credit scoring. Balancing techniques that neither do over-sampling nor under-sampling such as linear dependence approach and wavelet data transformation should be explored in credit scoring. Future research should incorporate macro-economic variables such as interest rates, unemployment rates and inflation rates. An increase in any of the mentioned macro-economic variables may increase the risk of a borrower defaulting. Future studies should consider the time it takes for borrowers to default and this will allow commercial banks to be forward-looking and proactive. The literature does not take into consideration the exploratory data analysis, hence future research should focus on this aspect to better understand for example the distributions of features. Future studies should focus more on time/model complexity to allow efficiency in model development in credit scoring. Future research should determine the correlation between the target variable and the independent variables/features in-order to identify predictive variables/features. There are few studies that focused on using heterogeneous base classifiers for ensemble methods [113,145,147]. However, future studies should focus more on using heterogeneous instead of homogeneous base classifiers in ensemble methods to allow diversity of base classifiers.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors would like to thank the Bankseta, South Africa (The Sector Education and Training Authority (SETA) for the banking industry) for making funds available to Xolani Dastile's PhD study. The authors also would like to thank the anonymous reviewers for providing valuable feedback on initial versions of this paper.

References

[1] L.C. Thomas, J. Crook, D. Edelman, Credit Scoring and Its Applications, Society for Industrial and Applied Mathematics, 2002.
[2] L.C. Thomas, A survey of credit and behavioural scoring: forecasting financial risk of lending to consumers, Int. J. Forecast. 16 (2) (2000) 149–172.
[3] N. Siddiqi, Credit Risk Scorecards: Developing and Implementing Intelligent Credit Scoring, SAS Publishing, 2005.
[4] I.J. Myung, Tutorial on maximum likelihood estimation, J. Math. Psych. 47 (1) (2003) 90–100.
[5] B. Baesens, R. Setiono, C. Mues, J. Vanthienen, Using neural network rule extraction and decision tables for credit-risk evaluation, Manage. Sci. 49 (3) (2003) 312–329.
[6] H.A. Alaka, L.O. Oyedele, H.A. Owolabi, V. Kumar, S.O. Ajayi, O.O. Akinade, M. Bilal, Systematic review of bankruptcy prediction models: Towards a framework for tool selection, Expert Syst. Appl. 94 (2018) 164–184.
[7] R. Schlosser, Appraising the Quality of Systematic Reviews, Focus: Technical Briefs 17, 2007, pp. 1–8.
[8] J.L. Bellovary, D.E. Giacomino, M.D. Akers, A review of bankruptcy prediction studies: 1930 to present, J. Financial Educ. 33 (2007) 1–42.
[9] H.A. Abdou, J. Pointon, Credit scoring, statistical techniques and evaluation criteria: A review of the literature, Int. J. Intell. Syst. Account. Financ. Manage. 18 (2–3) (2011) 59–88.
[10] W. Lin, Y. Hu, C. Tsai, Machine learning in financial crisis prediction: A survey, IEEE Trans. Syst. Man Cybern. C 42 (4) (2012) 421–436.
[11] X. Wang, M. Xu, Ö.T. Pusatli, A survey of applying machine learning techniques for credit rating: existing models and open issues, in: S. Arik, T. Huang, W.K. Lai, Q. Liu (Eds.), Neural Information Processing, Springer International Publishing, 2015, pp. 122–132.
[12] F. Louzada, A. Ara, G.B. Fernandes, Classification methods applied to credit scoring: Systematic review and overall comparison, Surv. Oper. Res. Manag. Sci. 21 (2) (2016) 117–134.
[13] S.S. Devi, Y. Radhika, A Survey on Machine Learning and Statistical Techniques in Bankruptcy Prediction, 2018.
[14] D. Liang, C.-F. Tsai, H.-T. Wu, The effect of feature selection on financial distress prediction, Knowl.-Based Syst. 73 (2015) 289–297.
[15] I. Brown, C. Mues, An experimental comparison of classification algorithms for imbalanced credit scoring data sets, Expert Syst. Appl. 39 (3) (2012) 3446–3453.
[16] K. Bijak, L.C. Thomas, Does segmentation always improve model performance in credit scoring? Expert Syst. Appl. 39 (3) (2012) 2433–2442.
[17] F.-L. Chen, F.-C. Li, Combination of feature selection approaches with SVM in credit scoring, Expert Syst. Appl. 37 (7) (2010) 4902–4909.
[18] W. Chen, L. Shi, Credit scoring with F-score based on support vector machine, in: Proceedings 2013 International Conference on Mechatronic Sciences, Electric Engineering and Computer, MEC, 2013, pp. 1512–1516.
[19] H. Chen, Y. Xiang, The study of credit scoring model based on group lasso, Procedia Comput. Sci. 122 (2017) 677–684, 5th International Conference on Information Technology and Quantitative Management, ITQM 2017.
[20] B.-W. Chi, C.-C. Hsu, A hybrid approach to integrate genetic algorithm into dual scoring model in enhancing the performance of credit scoring model, Expert Syst. Appl. 39 (3) (2012) 2650–2661.
[21] B. Back, T. Laitinen, K. Sere, Neural networks and genetic algorithms for bankruptcy predictions, Expert Syst. Appl. 11 (4) (1996) 407–413.
[22] S. Oreski, G. Oreski, Genetic algorithm-based heuristic for feature selection in credit risk assessment, Expert Syst. Appl. 41 (4, Part 2) (2014) 2052–2064.
[23] Q. Song, H. Jiang, J. Liu, Feature selection based on FDA and F-score for multi-class classification, Expert Syst. Appl. 81 (2017) 22–27.
[24] Z. Pawlak, Rough set approach to knowledge-based decision support, European J. Oper. Res. 99 (1) (1997) 48–57.
[25] J. Wang, K. Guo, S. Wang, Rough set and tabu search based feature selection for credit scoring, Procedia Comput. Sci. 1 (1) (2010) 2425–2432, ICCS 2010.
[26] Q. Zhang, Q. Xie, G. Wang, A survey on rough set theory and its applications, CAAI Trans. Intell. Technol. 1 (4) (2016) 323–333.
[27] C.-F. Tsai, Feature selection in bankruptcy prediction, Knowl.-Based Syst. 22 (2) (2009) 120–127.
[28] M. Mitchell, An Introduction to Genetic Algorithms, MIT Press, 1996.
[29] V. Kozeny, Genetic algorithms for credit scoring: Alternative fitness function performance comparison, Expert Syst. Appl. 42 (6) (2015) 2998–3004.
[30] M. Crepinsek, S.-H. Liu, M. Mernik, Exploration and exploitation in evolutionary algorithms: A survey, ACM Comput. Surv. 45 (2013) 35:1–35:33.
[31] S.-h. Liu, M. Mernik, B. Bryant, To explore or to exploit: An entropy-driven approach for evolutionary algorithms, KES J. 13 (2009) 185–206.
[32] J.M. Cadenas, M.C. Garrido, R. Martínez, Feature subset selection filter–wrapper based on low quality data, Expert Syst. Appl. 40 (16) (2013) 6241–6252.
[33] R. Tibshirani, Regression shrinkage and selection via the lasso: a retrospective, J. R. Stat. Soc. Ser. B Stat. Methodol. 73 (3) (2011) 273–282.
[34] A. Zheng, A. Casari, Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists, first ed., O'Reilly Media, Inc., 2018.
[35] S. Sehgal, H. Singh, M. Agarwal, V. Bhasker, Shantanu, Data analysis using principal component analysis, in: 2014 International Conference on Medical Imaging, M-Health and Emerging Communication Systems, MedCom, 2014, pp. 45–48.
[36] R.A. Fisher, The use of multiple measurements in taxonomic problems, Ann. Eugen. 7 (2) (1936) 179–188.
[37] A.M. Martinez, A.C. Kak, PCA versus LDA, IEEE Trans. Pattern Anal. Mach. Intell. 23 (2) (2001) 228–233.
[38] C. Rao, The utilization of multiple measurements in problems of biological classification, J. R. Stat. Soc. (1948) 159–203.
[39] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, second ed., Wiley, 2001.
[40] D. Reynolds, Gaussian mixture models, in: Encyclopedia of Biometrics, Springer US, Boston, MA, 2015, pp. 827–832.
[41] A.P. Dempster, N.M. Laird, D.B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, J. R. Stat. Soc. Ser. B Stat. Methodol. 39 (1) (1977) 1–38.
[42] W.E. Henley, D.J. Hand, A *k*-nearest-neighbour classifier for assessing consumer credit risk, J. R. Stat. Soc. 45 (1) (1996) 77–95.

[43] C. Cortes, V. Vapnik, Support-vector networks, Mach. Learn. 20 (3) (1995) 273–297.
[44] B. Schölkopf, The kernel trick for distances, in: Proceedings of the 13th International Conference on Neural Information Processing Systems, MIT Press, 2000, pp. 283–289.
[45] T.M. Mitchell, Machine Learning, first ed., McGraw-Hill, Inc., New York, NY, USA, 1997.
[46] F. Barboza, H. Kimura, E. Altman, Machine learning models and bankruptcy prediction, Expert Syst. Appl. 83 (2017) 405–417.
[47] C.-F. Tsai, Y.-F. Hsu, D.C. Yen, A comparative study of classifier ensembles for bankruptcy prediction, Appl. Soft Comput. 24 (2014) 977–984.
[48] C.-F. Tsai, J.-W. Wu, Using neural network ensembles for bankruptcy prediction and credit scoring, Expert Syst. Appl. 34 (4) (2008) 2639–2649.
[49] M.D. Odom, R. Sharda, A neural network model for bankruptcy prediction, in: 1990 IJCNN International Joint Conference on Neural Networks, vol. 2, 1990, pp. 163–168.
[50] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.
[51] Y. Freund, R.E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, J. Comput. System Sci. 55 (1) (1997) 119–139.
[52] T. Chen, C. Guestrin, XGBoost: A scalable tree boosting system, in: Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016, pp. 785–794.
[53] J. Nobre, R. Neves, Combining principal component analysis, discrete wavelet transform and xgboost to trade in the financial markets, Expert Syst. Appl. 125 (2019).
[54] Y. Xia, C. Liu, Y. Li, N. Liu, A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring, Expert Syst. Appl. 78 (2017).
[55] L. Breiman, Bagging predictors, Mach. Learn. 24 (2) (1996) 123–140.
[56] L. Yu, R. Zhou, L. Tang, R. Chen, A DBN-based resampling SVM ensemble learning paradigm for credit classification with imbalanced data, Appl. Soft Comput. 69 (2018) 192–202.
[57] G.E. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, Neural Comput. 18 (7) (2006) 1527–1554.
[58] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel, Backpropagation applied to handwritten zip code recognition, Neural Comput. 1 (4) (1989) 541–551.
[59] Y. Seo, K. shik Shin, Hierarchical convolutional neural networks for fashion image classification, Expert Syst. Appl. 116 (2019) 328–339.
[60] Y. Lecun, Y. Bengio, The Handbook of Brain Theory and Neural Networks, MIT Press, 1995, chapter Convolutional networks for images, speech, and time-series.
[61] F.F. Ting, Y.J. Tan, K.S. Sim, Convolutional neural network improvement for breast cancer classification, Expert Syst. Appl. 120 (2019) 103–115.
[62] O.B. Sezer, A.M. Ozbayoglu, Algorithmic financial trading with deep convolutional neural networks: time series to image conversion approach, Appl. Soft Comput. 70 (2018) 525–538.
[63] B. Zhao, H. Lu, S. Chen, J. Liu, D. Wu, Convolutional neural networks for time series classification, J. Syst. Eng. Electron. 28 (2017) 162–169.
[64] F. Chollet, Deep Learning with Python, first ed., Manning Publications Co., Greenwich, CT, USA, 2017.
[65] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016.
[66] C.M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, Inc., New York, NY, USA, 1995.
[67] J.M. Tomczak, M. Zieba, Classification restricted Boltzmann machine for comprehensible credit scoring model, Expert Syst. Appl. 42 (4) (2015) 1789–1796.
[68] G. Douzas, F. Bacao, Self-organizing map oversampling (SOMO) for imbalanced data set learning, Expert Syst. Appl. 82 (2017) 40–52.
[69] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: Synthetic minority over-sampling technique, J. Artificial Intelligence Res. 16 (2002) 321–357.
[70] R. Saia, S. Carta, G. Fenu, A Wavelet-Based Data Analysis to Credit Scoring, 2018.
[71] J. Nobre, R. Neves, Combining principal component analysis, discrete wavelet transform and xgboost to trade in the financial markets, Expert Syst. Appl. 125 (2019).
[72] R. Saia, S. Carta, A Linear-Dependence-Based Approach to Design Proactive Credit Scoring Models, 2016.
[73] R. Setiono, H. Liu, Symbolic representation of neural networks, Computer 29 (3) (1996) 71–77.
[74] M.W. Craven, J.W. Shavlik, Extracting tree-structured representations of trained networks, in: Proceedings of the 8th International Conference on Neural Information Processing Systems, MIT Press, 1995, pp. 24–30.
[75] M.T. Ribeiro, S. Singh, C. Guestrin, "Why should I trust you?": Explaining the predictions of any classifier, CoRR abs/1602.04938 (2016).
[76] R.A. Eisenbeis, Problems in applying discriminant analysis in credit scoring models, J. Bank. Financ. 2 (3) (1978) 205–219.
[77] G.H. John, P. Langley, Estimating continuous distributions in Bayesian classifiers, in: Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann Publishers Inc., 1995, pp. 338–345.
[78] L. Yu, X. Yao, S. Wang, K. Lai, Credit risk evaluation using a weighted least squares SVM classifier with design of experiment for parameter selection, Expert Syst. Appl. 38 (12) (2011) 15392–15399.
[79] Y. Jiang, Credit scoring model based on the decision tree and the simulated annealing algorithm, in: 2009 WRI World Congress on Computer Science and Information Engineering, Vol. 4, 2009, pp. 18–22.
[80] R. Setiono, H. Liu, Neurolinear: From neural networks to oblique decision rules, Neurocomputing 17 (1) (1997) 1–24.
[81] R. Caruana, Y. Lou, J. Gehrke, P. Koch, M. Sturm, N. Elhadad, Intelligible models for healthCare: Predicting pneumonia risk and hospital 30-day readmission, in: KDD '15, 2015.
[82] S. Lundberg, S. Lee, A unified approach to interpreting model predictions, CoRR abs/1705.07874 (2017).
[83] J.H. Friedman, Greedy function approximation: A gradient boosting machine, Ann. Statist. 29 (2000) 1189–1232.
[84] C. Luo, D. Wu, D. Wu, A deep learning approach for credit scoring using credit default swaps, Eng. Appl. Artif. Intell. 65 (2017) 465–470.
[85] S. Ramasamy, K. Rajaraman, A hybrid meta-cognitive restricted Boltzmann machine classifier for credit scoring, in: TENCON 2017 - 2017 IEEE Region 10 Conference, 2017, pp. 2313–2318.
[86] K. Tran, T. Duong, Q. Ho, Credit scoring model: A combination of genetic programming and deep learning, in: 2016 Future Technologies Conference, FTC, 2016, pp. 145–149.
[87] S.H. Yeh, C.J. Wang, M.F. Tsai, Deep belief networks for predicting corporate defaults, in: 2015 24th Wireless and Optical Communication Conference, WOCC, 2015, pp. 159–163.
[88] V. Neagoe, A. Ciotec, G. Cucu, Deep convolutional neural networks versus multilayer perceptron for financial prediction, in: 2018 International Conference on Communications, COMM, 2018, pp. 201–206.
[89] S. Hamori, M. Kawai, T. Kume, Y. Murakami, C. Watanabe, Ensemble learning or deep learning? Application to default risk analysis, J. Risk Financial Manag. 11 (1) (2018).
[90] C. Shorten, T.M. Khoshgoftaar, A survey on image data augmentation for deep learning, J. Big Data 6 (1) (2019) 60.
[91] A. Gómez-Ríos, S. Tabik, J. Luengo, A.S.M. Shihavuddin, B. Krawczyk, F. Herrera, Towards highly accurate coral texture images classification using deep convolutional neural networks and data augmentation, CoRR abs/1804.00516 (2018).
[92] H. Kvamme, N. Sellereite, K. Aas, S. Sjursen, Predicting mortgage default using convolutional neural networks, Expert Syst. Appl. 102 (2018).
[93] A. Krizhevsky, I. Sutskever, G. Hinton, Imagenet classification with deep convolutional neural networks, Neural Inf. Process. Syst. 25 (2012).
[94] L. Perez, J. Wang, The effectiveness of data augmentation in image classification using deep learning, CoRR (2017).
[95] J. Salamon, J.P. Bello, Deep convolutional neural networks and data augmentation for environmental sound classification, CoRR (2016) http://arxiv.org/abs/1608.04363.
[96] M. Frid-Adar, E. Klang, M. Amitai, J. Goldberger, H. Greenspan, Synthetic data augmentation using GAN for improved liver lesion classification, CoRR (2018).
[97] B. Zhu, W. Yang, H. Wang, Y. Yuan, A hybrid deep learning model for consumer credit scoring, in: 2018 International Conference on Artificial Intelligence and Big Data, ICAIBD, 2018, pp. 205–208.
[98] M.F. Kiani, F. Mahmoudi, A new hybrid method for credit scoring based on clustering and support vector machine (ClsSVM), in: 2010 2nd IEEE International Conference on Information and Financial Engineering, 2010, pp. 585–589.
[99] D. Zhang, X. Zhou, S.C. Leung, J. Zheng, Vertical bagging decision trees model for credit scoring, Expert Syst. Appl. 37 (12) (2010) 7838–7843.
[100] M.A.H. Farquad, V. Ravi, Sriramjee, G. Praveen, Credit scoring using PCA-SVM hybrid model, in: V.V. Das, J. Stephen, Y. Chaba (Eds.), Computer Networks and Information Technologies, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 249–253.
[101] Y. Ping, L. Yongheng, Neighborhood rough set and SVM based hybrid credit scoring classifier, Expert Syst. Appl. 38 (9) (2011) 11300–11304.
[102] J. Wang, A.-R. Hedar, S. Wang, J. Ma, Rough set and scatter search metaheuristic based feature selection for credit scoring, Expert Syst. Appl. 39 (6) (2012) 6123–6128.
[103] L. Han, L. Han, H. Zhao, Orthogonal support vector machine for credit scoring, Eng. Appl. Artif. Intell. 26 (2) (2013) 848–862.
[104] J. Shi, S.-y. Zhang, L.-m. Qiu, Credit scoring by feature-weighted support vector machines, J. Zhejiang Univ. Sci. C 14 (3) (2013) 197–204.
[105] Q. Li, J. Zhang, Y. Wang, K. Kang, Credit risk classification using discriminative restricted boltzmann machines, in: 2014 IEEE 17th International Conference on Computational Science and Engineering, 2014, pp. 1697–1700.

[106] S. Maldonado, J. Pérez, C. Bravo, Cost-based feature selection for support vector machines: An application in credit scoring, European J. Oper. Res. 261 (2) (2017) 656–665.
[107] H. Sutrisno, S. Halim, Credit scoring refinement using optimized logistic regression, in: 2017 International Conference on Soft Computing, Intelligent System and Information Technology, ICSIIT, 2017, pp. 26–31.
[108] R.A. Mancisidor, M. Kampffmeyer, K. Aas, R. Jenssen, Segment-based credit scoring using latent clusters in the variational autoencoder, 2018, arXiv:1806.02538.
[109] X. Zhang, Y. Yang, Z. Zhou, A novel credit scoring model based on optimized random forest, in: 2018 IEEE 8th Annual Computing and Communication Workshop and Conference, CCWC, 2018, pp. 60–65.
[110] S. Jadhav, H. He, K. Jenkins, Information gain directed genetic algorithm wrapper feature selection for credit rating, Appl. Soft Comput. 69 (2018) 541–553.
[111] G. Dong, K.K. Lai, J. Yen, Credit scorecard based on logistic regression with random coefficients, Procedia Comput. Sci. 1 (1) (2010) 2463–2468, ICCS 2010.
[112] B. Twala, Multiple classifier application to credit risk assessment, Expert Syst. Appl. 37 (4) (2010) 3326–3336.
[113] N.-C. Hsieh, L.-P. Hung, A data driven ensemble classifier for credit scoring analysis, Expert Syst. Appl. 37 (1) (2010) 534–545.
[114] L. Yu, X. Yao, S. Wang, K. Lai, Credit risk evaluation using a weighted least squares SVM classifier with design of experiment for parameter selection, Expert Syst. Appl. 38 (12) (2011) 15392–15399.
[115] G. Wang, J. Hao, J. Ma, H. Jiang, A comparative assessment of ensemble learning for credit scoring, Expert Syst. Appl. 38 (1) (2011) 223–230.
[116] Q. Wang, K.K. Lai, D. Niu, Green credit scoring system and its risk assessemt model with support vector machine, in: 2011 Fourth International Joint Conference on Computational Sciences and Optimization, 2011, pp. 284–287.
[117] B. Ribeiro, N. Lopes, Deep belief networks for financial prediction, in: B.-L. Lu, L. Zhang, J. Kwok (Eds.), Neural Information Processing, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 766–773.
[118] B.W. Yap, S.H. Ong, N.H.M. Husain, Using data mining to improve assessment of credit worthiness via credit scoring models, Expert Syst. Appl. 38 (10) (2011) 13274–13283.
[119] F. Louzada, O. Anacleto-Junior, C. Candolo, J. Mazucheli, Poly-bagging predictors for classification modelling for credit scoring, Expert Syst. Appl. 38 (10) (2011) 12717–12720.
[120] A. Marqués, V. García, J. Sánchez, Exploring the behaviour of base classifiers in credit scoring ensembles, Expert Syst. Appl. 39 (11) (2012) 10244–10250.
[121] A. Marqués, V. García, J. Sánchez, Two-level classifier ensembles for credit risk assessment, Expert Syst. Appl. 39 (12) (2012) 10916–10922.
[122] B. Tang, S. Qiu, A new credit scoring method based on improved fuzzy support vector machine, in: 2012 IEEE International Conference on Computer Science and Automation Engineering, CSAE, Vol. 3, 2012, pp. 73–75.
[123] F. Louzada, P.H. Ferreira-Silva, C.A. Diniz, On the impact of disproportional samples in credit scoring models: An application to a Brazilian bank data, Expert Syst. Appl. 39 (9) (2012) 8071–8078.
[124] J. Abellán, C.J. Mantas, Improving experimental studies about ensembles of classifiers for bankruptcy prediction and credit scoring, Expert Syst. Appl. 41 (8) (2014) 3825–3830.
[125] T. Harris, Credit scoring using the clustered support vector machine, Expert Syst. Appl. 42 (2) (2015) 741–750.
[126] B. Yi, J. Zhu, Credit scoring with an improved fuzzy support vector machine based on grey incidence analysis, in: 2015 IEEE International Conference on Grey Systems and Intelligent Services, GSIS, 2015, pp. 173–178.
[127] S. Jones, D. Johnstone, R. Wilson, An empirical evaluation of the performance of binary classifiers in the prediction of credit ratings changes, J. Bank. I Finance 56 (2015) 72–85.
[128] J. Chen, L. Xu, A method of improving credit evaluation with support vector machines, in: 2015 11th International Conference on Natural Computation, ICNC, 2015, pp. 615–619.
[129] Z. Zhao, S. Xu, B.H. Kang, M.M.J. Kabir, Y. Liu, R. Wasinger, Investigation and improvement of multi-layer perceptron neural networks for credit scoring, Expert Syst. Appl. 42 (7) (2015) 3508–3516.
[130] R. Florez-Lopez, J.M. Ramon-Jeronimo, Enhancing accuracy and interpretability of ensemble strategies in credit risk assessment. A correlated-adjusted decision forest proposal, Expert Syst. Appl. 42 (13) (2015) 5737–5753.
[131] R. Florez-Lopez, J.M. Ramon-Jeronimo, Enhancing accuracy and interpretability of ensemble strategies in credit risk assessment. A correlated-adjusted decision forest proposal, Expert Syst. Appl. 42 (13) (2015) 5737–5753.
[132] M. Aláraj, M. Abbod, A systematic credit scoring model based on heterogeneous classifier ensembles, in: 2015 International Symposium on Innovations in Intelligent SysTems and Applications, INISTA, 2015, pp. 1–7.
[133] M. Aláraj, M.F. Abbod, Classifiers consensus system approach for credit scoring, Knowl.-Based Syst. 104 (2016) 89–105.
[134] L. Yu, Z. Yang, L. Tang, A novel multistage deep belief network based extreme learning machine ensemble learning paradigm for credit risk assessment, Flex. Serv. Manuf. J. 28 (4) (2016) 576–592.
[135] H. Xiao, Z. Xiao, Y. Wang, Ensemble classification based on supervised clustering for credit scoring, Appl. Soft Comput. 43 (2016) 73–86.
[136] A. Bequé, S. Lessmann, Extreme learning machines for credit scoring: An empirical evaluation, Expert Syst. Appl. 86 (2017) 42–53.
[137] A. Lawi, F. Aziz, S. Syarif, Ensemble gradientboost for increasing classification accuracy of credit scoring, in: 2017 4th International Conference on Computer Applications and Information Processing Technology, CAIPT, 2017, pp. 1–4.
[138] Y. Li, X. Lin, X. Wang, F. Shen, Z. Gong, Credit risk assessment algorithm using deep neural networks with clustering and merging, in: 2017 13th International Conference on Computational Intelligence and Security, CIS, 2017, pp. 173–176.
[139] Z. Li, Y. Tian, K. Li, F. Zhou, W. Yang, Reject inference in credit scoring using semi-supervised support vector machines, Expert Syst. Appl. 74 (2017) 105–114.
[140] O.J. Okesola, K.O. Okokpujie, A.A. Adewale, S.N. John, O. Omoruyi, An improved bank credit scoring model: A Naïve Bayesian approach, in: 2017 International Conference on Computational Science and Computational Intelligence, CSCI, 2017, pp. 228–233.
[141] H. Chen, M. Jiang, X. Wang, Bayesian ensemble assessment for credit scoring, in: 2017 4th International Conference on Industrial Economics System and Industrial Security Engineering, IEIS, 2017, pp. 1–5.
[142] J. Abellán, J.G. Castellano, A comparative study on base classifiers in ensemble methods for credit scoring, Expert Syst. Appl. 73 (2017) 1–10.
[143] B. Vanderheyden, J. Priestley, Logistic ensemble models, 2018.
[144] P. Martey Addo, D. Guegan, B. Hassani, Credit risk analysis using machine and deep learning models, Risks 6 (2018) 38.
[145] Y. Xia, C. Liu, B. Da, F. Xie, A novel heterogeneous ensemble credit scoring model based on bstacking approach, Expert Syst. Appl. 93 (C) (2018) 182–199.
[146] Y.-C. Chang, K.-H. Chang, G.-J. Wu, Application of extreme gradient boosting trees in the construction of credit risk assessment models for financial institutions, Appl. Soft Comput. 73 (2018).
[147] W. Li, S. Ding, Y. Chen, S. Yang, Heterogeneous ensemble for default prediction of peer-to-peer lending in China, IEEE Access 6 (2018) 54396–54406.
[148] A. Cao, H. He, Z. Chen, W. Zhang, Performance evaluation of machine learning approaches for credit scoring, Int. J. Econ. Finance Manag. Sci. 6 (2018) 255–260.
[149] Basel Committee on Banking Supervision, Basel II: International convergence of capital measurement and capital standards: A revised framework - comprehensive version, bank for international settlements, BIS (2006).

2.2 Journal Article II

X. Dastile and T. Celik, Making Deep Learning-Based Predictions for Credit Scoring Explainable, *IEEE Access*, vol. 9, pp. 50426-50440, 2021.

Number of citations to date: 11
Publisher: Institute of Electrical and Electronics Engineers
Cite Score: 3.367
Impact Factor: 4.8
Abstracting and Indexing: Scopus (Elsevier)
Web of Science (Clarivate Analytics)
ProQuest
IET (The Institution of Engineering and Technology)
NLM (US National Library of Medicine)
CrossRef
SNIP:

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 11, 2021, accepted March 21, 2021, date of publication March 24, 2021, date of current version April 7, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3068854

Making Deep Learning-Based Predictions for Credit Scoring Explainable

XOLANI DASTILE[1] AND TURGAY CELIK[2,3,4], (Member, IEEE)
[1]School of Computer Science and Applied Mathematics, University of the Witwatersrand, Johannesburg 2000, South Africa
[2]School of Electrical and Information Engineering, University of the Witwatersrand, Johannesburg 2000, South Africa
[3]Wits Institute of Data Science, University of the Witwatersrand, Johannesburg 2000, South Africa
[4]School of Information Science and Technology, Southwest Jiaotong University, Chengdu 610031, China

Corresponding author: Turgay Celik (celikturgay@gmail.com)

ABSTRACT Credit scoring has become an important risk management tool for money lending institutions. Over the years, statistical and classical machine learning models have been the most researched risk management tools in credit scoring literature, and recently the focus has turned to deep learning models. This transition is due to better performances that are shown by deep learning models in different domains. Despite deep learning models' superior performances, there is still a need for explaining how these models make their predictions. The non-transparency nature of deep learning models has created a bottleneck for their use in credit scoring. Explanations of decisions are important for lending institutions since it is a requirement for automated decisions that are generated by non-transparent models to be explained. The other issue in using deep learning models, specifically 2D Convolutional Neural Networks (CNNs), in credit scoring is the need to have the data in image format. We propose an explainable deep learning model for credit scoring which can harness the performance benefits offered by deep learning and yet comply with the legislation requirements for the automated decision-making processes. The proposed method converts tabular datasets into images and thus allowing the application of 2D CNNs in credit scoring. Each pixel of the image corresponds to a feature bin of the tabular dataset. The predictions from the 2D CNNs were explained using state-of-the-art explanation methods. Furthermore, explanations were evaluated using a sanity check methodology and also performances of the explanation methods were compared quantitatively. The proposed explainable deep learning model outperforms the other credit scoring methods on publicly available credit scoring datasets.

INDEX TERMS Credit scoring, convolutional neural network, deep learning, explainable artificial intelligence.

I. INTRODUCTION

Credit scoring, proposed by Durand [1], has become an important risk management tool for money lending institutions. Over the years, statistical and classical machine learning models have been the most researched risk management tools in credit scoring literature, and recently the focus has turned to deep learning models [2]. This transition is due to better performances that are shown by deep learning models in different domains. Despite better performances, there is still a need for explaining how deep learning models make their predictions. According to Liu *et al.* [3], an explanation is defined as ''the ability to provide a visual or textual presentation of connections between input features and output predictions''. Similarly, Doshi-Velez and Kim [4] defined an explanation as ''the ability to explain or to provide the meaning in understandable terms to a human''. Note that in this study we treat explanation and interpretability as interchangeable terms although in literature some argue that these terms differ [5], Rothman [6] annotates that an explanation creates comprehensibility and clarity, on the other hand interpretability tells the meaning. This study intends not to pursue the philosophical debate on the differences between explanation and interpretability. The aim of this current study is to explain predictions made by a deep learning model in a credit scoring setting.

Explaining a prediction is particularly important in credit scoring since it is now a requirement for lending institutions under the Basel Accord [7] to explain to an applicant why her/his loan application was denied. Regulations such

The associate editor coordinating the review of this manuscript and approving it for publication was Wei Xiang.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

as the European Union General Data Protection Regulation (GDPR) [8] have made it compulsory for machine learning models to explain their predictions under the notion ''right to explanation''. Explanations foster trust in model predictions and assure that no discriminations are incurred during the application process when assessing credit worthiness of loan applicants. Explanations are playing an important role in areas where decisions are critical, for example in healthcare and in banking, to mention a few. This requirement of the ''right to explanation'' is advancing the research field that is coined as Explainable Artificial Intelligence (XAI) [6]. XAI is not a new concept and it has a rich history of more than 50 years [9]. Breiman was amongst the early adopters of interpretable systems and Breiman is known for developing Classification and Regression Trees and is quoted in one of the papers stating that ''On interpretability, trees are an A+'' [10].

Samek and Müller [11] highlighted at high level, the different techniques (old and emerging) that are used for XAI. The need, the purpose of different users, the evaluation methods and the future directions of XAI were also discussed in [11]. However, Rudin [12] provides a contrasting view on XAI and highlights the reasons why interpretable models should be used over explainable black box models for high-stakes decisions such as criminal justice, healthcare and credit scoring. Their argument was based on the premise that if a second model is used to explain a black box model (i.e. post-hoc explanation), there is a high chance that the explanation is not ''faithful to what the original model computes'' [12]. The recommendation was that instead of using post-hoc explanations, inherently interpretable models (i.e. ante-hoc explanations) that perform similarly to some of the black box models should be used for high-stake decisions. It was argued further that there is no truth in the statement ''there is a trade-off between accuracy and interpretability'', in other words it is a myth that non-interpretable models are more accurate than inherently interpretable models. To decide whether to use inherently interpretable models or not should be based on the circumstantial needs. In their review article Chari *et al.* [13] synthesised a taxonomy of explanations from the literature. This taxonomy consists of nine different types of explanations such as case-based, contextual, contrastive, scientific, statistical, trace-based, everyday and simulation-based explanations. The aim was to help produce explanations that are aligned to the circumstantial needs. Despite these different views from the literature, our study was motivated by [14]. However, our study differs from [14] in such a way that we have systematically discretized data into optimal categories using weights of evidence and we used both categorical and continuous features as opposed to [14].

Our contributions in this paper are as follows. 1) This is the first study to convert tabular data into images using a novel method and also it is the first method after such a conversion takes place that employs different explanation methods to explain the decision on each predicted credit sample. 2) We empirically showed that the explanations from different explanation methods can be used as optimal features and we observed that there is a huge boost in the classifiers' performance. In this work, we showed for the first time that the explanations are good features that can be directly applied on the classifiers and perform better when compared to the same classifiers operating on the raw data.

The remainder of this paper is organised as follows, Section II reviews and examines previous research of deep learning models in credit scoring, explanations of convolutional neural networks in other domains as well as the conversion of tabular data into images. Section III discusses the proposed framework of this study. Section IV discusses conversion of tabular datasets into images, training of 2D convolutional neural networks and the techniques used for explaining individual predictions. The experiments are described in Section V. The results are presented in Section VI and the conclusion is in Section VII.

II. RELATED WORK

In recent years, researchers have shown an interest in the application of deep learning models in credit scoring. For instance, Zhu *et al.* [14] used a hybrid method by combining a relief algorithm (which performs feature selection) with Convolutional Neural Network (CNN) to perform credit scoring and the hybrid relief-CNN model showed better performances when compared to Logistic Regression and Random Forest (RF) models. Tomczak and Zieba [15] used Restricted Boltzmann Machine (RBM) in credit scoring. A comprehensible scoring table from RBM showed better performances compared to classical machine learning models. Li *et al.* [16] conducted a study of deep learning with clustering and merging and the results showed high prediction accuracies. Deep Multi-Layer Perceptron (DMLP) and Deep CNN (DCNN) were used by Neagoe *et al.* [17] to assess the credit worthiness of applicants. The DCNN significantly outperformed DMLP. Hamori *et al.* [18] compared Deep Neural Network (DNN) with RF, bagging and boosting methods on credit datasets and the results showed that the ensemble methods performed better than DNN.

Deep learning with RF feature importance approach was proposed by Ha and Nguyen [19] for credit scoring. The empirical results showed that the proposed approach performed better than baseline methods such as the Decision Trees, k-Nearest Neighbour, Naíve-Bayes, Multi-Layer Perceptron and Random Forest with the exception of Support Vector Machine on German credit dataset. The proposed approach outperformed all baseline methods on Australian credit dataset. In their empirical study Sirignano *et al.* [20] developed a cohort of neural network ensemble models to assist in predicting the probability of default in credit risk using German and Australian credit datasets. The results showed that the proposed neural network ensemble models provide similar or better performance results compared to the literature results as well as to baseline single classifiers. Tripathi *et al.* [21] proposed a novel algebraic activation function to improve the performance of Extreme Learning

Machine (ELM) model in credit scoring. The ELM consists of an input layer, a single hidden layer and an output layer. The study also proposed Bat algorithm which is an evolutionary approach to initialise the weights and biases of the ELM. The results showed a significant improvement on German and Australian credit datasets. Edla *et al.* [22] combined a binary particle swarm optimization and gravitational search algorithm (BPSOGSA) with a multi-layer ensemble classifier using five heterogeneous classifiers. The purpose of BPSOGSA was to select predictive features. The results showed that the proposed hybrid model outperforms the Random Forest model and other majority voting ensemble techniques on German, Australian and Japanese credit datasets. Similarly, Tripathi *et al.* [23] proposed a hybrid credit scoring model using a dimension reduction approach (i.e. neighbourhood rough set) and a multi-layer ensemble classifier. The experimental results showed that the proposed model can achieve satisfactory performance in credit scoring.

Regulated institutions are not willing to adapt models that cannot explain predictions and this is hampering the use of machine learning, specifically deep learning models. However, Ariza-Garzón *et al.* [24] conducted a study in credit scoring where the focus was to make non-transparent machine learning models explainable. In the study, classical machine learning models were compared to a logistic regression model for Peer-to-Peer (P2P) lending and predictions were explained using SHAP values. The results showed that the classical machine learning models not only perform better in terms of classification but also perform better in terms of explanations. Moreover, classical machine learning models showed that they can reflect dispersion, non-linearity and structural breaks in the relationships between each independent variable and the response variable [24]. The credit scoring literature has not researched extensively in explanations of deep learning predictions.

There has been interests in credit scoring and other fields for converting tabular datasets into images in order to apply 2D CNN models. For example, Zhu *et al.* [14] combined relief algorithm with CNN model. The relief algorithm was used to select predictive features and the CNN was used for classification. The study converted credit scoring tabular data into images by bucketing features and mapping them into image pixels. However, the study considered only numeric features. The results showed that the relief-CNN hybrid model performed better than the benchmark models in credit scoring such as the random forest and the logistic regression. Sharma *et al.* [25] proposed a novel approach called DeepInsight to convert non-image data into images and apply CNN models. The types of data that were used in the study were Ribonucleic Acid (RNA) sequence data, vowels data, text data and artificial data. The results showed a better performance from the DeepInsight approach compared to the state-of-the-art machine learning models such as the Ada-boost and the Random Forest models. Yang *et al.* [26] converted a multivariate time-series data firstly into two-dimensional colored images and secondly the two-dimensional colored images were combined to form a single image and lastly a CNN was applied on a single image for classification. The study compared three transformation methods for converting time series data into images. The transformation methods were Gramian Angular Summation Field (GASF), Gramian Angular Difference Field (GADF), and Markov Transition Field (MTF). The study was focusing on assessing the effect of using different transformation methods, the sequences of combining images, and the complexity of CNN architectures on classification accuracy. The results showed that the selection of transformation methods and the sequence of combination do not significantly impact the prediction outcome. Further, the results also showed that the proposed framework performed better than the other classification models in terms of the accuracy metric. Buturovi'c and Miljkovi'c [27] proposed a novel approach called TAbular Convolution (TAC) for converting tabular dataset into images for the application of 2D CNN models. The study used gene expression data obtained from blood samples of patients with bacterial or viral infections. The results showed a similar performance between the TAC approach and the state-of-the-art non-CNN machine learning classifiers in terms of accuracy when classifying gene expression data. Singh *et al.* [28] converted sensor dataset obtained from the floor surface pressure mapping into image data. Thereafter, a pre-trained CNN was applied on the image data and the results showed that the proposed method performed significantly better (by 10%) than the traditional machine learning methods. Note that none of these methods used the proposed conversion method that was undertaken in this current study. Furthermore, to the best of our knowledge, there is no other method in the literature which transforms tabular data into images in the way we proposed and applies explanation methods on predictions from 2D CNN.

Several studies have also focused on making CNNs interpretable in other domains. For instance, Tamajka *et al.* [29] used activations of the CNN as discreptors. Firstly the data was split into training and test sets, then the CNN was trained using a shallow network, the activations from the first pooling layer were used as a database of activations from known observations. Secondly, the test dataset was fed through a similar CNN and activations were extracted. Thirdly, to identify the class of each test set observation, its activation was compared to the activations of the training set samples and the class was assigned using a majority vote. This is similar to k-Nearest Neighbour approach and the authors used both cosine and euclidean as distance measures. The study used MNIST dataset and the results showed that the proposed interpretable model achieved almost similar results to the trained original CNN. Simonyan *et al.* [30] trained a DCNN on ILSVRC-2013 dataset which consists of 1.2 million images and two visualisation techniques were considered based on computing the gradient of the class score with respect to the input image. The finding was an establishment of the connection between the gradient-based CNN visualisation methods and deconvoltional networks. Liu *et al.* [31] applied CNN on MNIST dataset with the aim of interpreting

the inner workings of the Fully-Connected (FC) layer. Firstly, the activations of the hidden layer of the FC layer were clustered to form factors. A factor is simply a label/class of a cluster. Secondly, identifications (IDs) were assigned to each cluster. Thirdly, the IDs were combined with the original data to form a meta-level data. Lastly, a decision tree was trained on meta-level data. For interpreting the hidden layer of the FC layer, the activations of the hypothesis class vs. true class were plotted on the x-y plane using all rows/depths of the decision tree. If there was an overlap between the activations, then this implied that there was no separation between the hypothesis class and the true class, this process was repeated until there was only one class or there was less overlap between the activations. The finding was that the proposed process was faithful to the original CNN model, i.e. accuracy was not compromised by the interpretable CNN.

III. PROPOSED FRAMEWORK

The proposed framework serves as a guideline for using 2D CNNs on tabular datasets. Firstly, this framework proposes that the feature values from tabular datasets should be mapped into different bins that are used to calculate weights of evidence and thereafter images to be created from the bins. After creating the images, the image dataset is split into training and test sets. Secondly, a 2D CNN needs to be trained on the training set and its performance to be assessed on the test set. Thirdly, the predictions from the 2D CNN need to be explained using state-of-the-art methods such as the Grad-CAM, SHAP values, Saliency Map and LIME. Fourthly, explanations must be validated using sanity check methodology [32]. Lastly, all explanation techniques must be compared quantitatively for the purpose of identifying the best performing explanation technique.

IV. METHODOLOGY

This section discusses the methodology undertaken in this study and its components in detail.

A. WEIGHT OF EVIDENCE

The *Weight of Evidence* (WOE) is used as a transformation technique for features in credit scoring [33]. The first step in calculating WOE is to create bins for each feature. Thereafter, the WOE is calculated for each bin. We denote the weight of evidence for bin b in feature f as $WOE_b^{(f)}$ which is calculated as follows

$$WOE_b^{(f)} = \ln\left(\frac{\%Dst.Gds_b^{(f)}}{\%Dst.Bds_b^{(f)}}\right), \tag{1}$$

where $Dst.Gds_b^{(f)}$ and $Dst.Bds_b^{(f)}$ are the distribution of good and bads for bin b in feature f computed according to Eq. (2) and Eq. (3), respectively.

$$Dst.Gds_b^{(f)} = \frac{\#ofgoods_b^{(f)}}{(\#ofgoods_b^{(f)} + \#ofbads_b^{(f)})} \tag{2}$$

$$Dst.Bds_b^{(f)} = \frac{\#ofbads_b^{(f)}}{(\#ofgoods_b^{(f)} + \#ofbads_b^{(f)})} \tag{3}$$

WOE computations are performed as follows [33]. Each bin is created in such a way that the bin contains at least 5% of data. This is to ensure that no bins are empty. There are no bins with 0 counts for goods and bads. This means that even if the data is not balanced, each bin will have both goods and bads and no bin will contain one type of class. The bins should have a monotonically decreasing or increasing relationship with the response/target variable. This is to assure that there are no reversals in the relationship between the features and the response variable. To crystallise this, suppose a feature age has a relationship with a default risk target variable. In general, younger people tend to be at a higher risk of defaulting on their loans compared to older people, and this implies that risk decreases monotonically with increasing age. Note that in this study we are not directly using WOE for converting tabular datasets into images but bins that are used to calculate WOE. The WOE are only used as inputs to information value calculation which is discussed in the following section.

Table 1 shows an example calculation of WOE. The `Age` feature is bucketed into different bins. For each bin, the distribution of goods and bads are calculated. Each bin has at least 5% of data and no bin contains 0 counts of either bads or goods. The goods are non-defaults and the bads are defaults. The WOEs are showing a decreasing trend. Later, we will show how we use bins for calculating WOE in order to convert our dataset into images. The bins for WOE calculations are optimal bins in terms of class distribution and also no bins contain single class information.

TABLE 1. Calculating Weights Of Evidence (WOE) for `Age` feature. The `Age` feature is bucketed into different bins. For each bin, the distribution of goods (Dst.Gds) and bads (Dst.Bds) are calculated. Each bin has at least 5% of data and no bin contains 0 counts of either bads or goods. The goods are non-defaults and the bads are defaults. The WOEs are showing a decreasing trend.

Age	Count	Dst.count	#of Goods	Dst.Gds	# of Bads	Dst.Bds	WOE
Missing	35	18.13%	30	18.75%	5	15.15%	21.31%
18 - 22	50	25.91%	42	26.25%	8	24.24%	7.96%
23 - 29	30	15.54%	25	15.63%	5	15.15%	3.08%
30 - 34	47	24.35%	38	23.75%	9	27.27%	-13.83%
34+	31	16.06%	25	15.63%	6	18.18%	-15.15%
Total	193		160		33		

B. INFORMATION VALUE

The Information Value (IV) is used to select predictive features and is calculated as follows for each feature f,

$$IV^{(f)} = \sum_{b=1}^{B^{(f)}} (\%Dst.Gds_b^{(f)} - \%Dst.Bds_b^{(f)}) \times WOE_b^{(f)}, \tag{4}$$

where $B^{(f)}$ represents the number of bins for feature f. The following thresholds apply as a general rule of thumb when using IV [33]:

TABLE 2. Bins for WOE calculations in (a) Australian dataset, (b) German dataset, and (c) HMEQ dataset. Legend for German dataset: `DM` (Deutsche Mark), `Acc Bal` (Account Balance), `Age` (years), `DCredit` (Duration of credit), `LEmploy` (Length of current employment), `Asset` (Most valuable available asset), `VStocks` (Value of savings/stocks) and `PStatus` (Payment status of previous credit). Legend for HMEQ dataset: `Loan` (Requested loan amount), `Mortdue` (Amount due on existing mortgage), `Value` (Value of current property), `Job` (Occupancy), `Derog` (Number of major derogatory reports), `Delinq` (Number of delinquent credit lines), `Clage` (Age of oldest credit line in months), `Ninq` (Number of recent credit enquiries) and `Debtinc` (Debt-to-income ratio).

A2	A3	A4	A5	A6	A7	A8	A9	A10	A12	A13	A14
[13.75,24.08]	[0,1.5]	g	aa	bb	[0,0.17]	f	f	[0,1]	g	[0,160]	[1,6]
[24.17,34.08]	[1.54,5.04]	gg	c	dd	[0.21,1]	t	t	[2,67]	p	[163,2000]	[7,100001]
[34.17,80.25]	[5.09,28.00]	p	cc	ff	[1.54,2.63]				s		
			d	h	[2.71,28.5]						
			e	j							
			ff	o							
			i	v							
			j	z							
			k								
			m								
			q								
			r								
			w								
			x								

(a)

Acc Bal	Age	DCredit	LEmploy	Asset	VStocks	PStatus
$\geq$ 200 DM	[19,33]	[13,24]	$\geq$ 7 year	Life insurance	$100 \leq \cdots < 500$ DM	All credits duly paid
$<$ 200 DM	[34,75]	[26,72]	$1 \leq \cdots \leq 3$ years	No assets	$500 \leq \cdots < 1000$ DM	Account in arrears
Negative DM		[4,12]	$4 \leq \cdots \leq 6$ years	Owns a car	No savings	Delay in payment
No account			Unemployed	Real estate	$<$ 100 DM	Existing credits duly paid
			$<$ 1 year		$\geq$ 1000 DM	No credit lines

(b)

Loan	Mortdue	Value	Job	Derog	Delinq	Clage	Ninq	Debtinc
[1100,12900]	[2063,42000]	[106450,855909]	Migrant worker	[0,1]	[0,1]	[0,90]	[0,1]	[0.52,31.92]
[13000,20800]	[42003,61712]	[72045,106431]	Office worker	[2,10]	[2,15]	[108,129]	[2,17]	[31.93,203.31]
[20900,89900]	[61767,88197]	[8000,72000]	Other			[130,166]		
	[88210,399550]		Professional Executive			[167,194]		
			Sales person			[195,226]		
			Self employed			[227,278]		
						[279,1168]		
						[91,107]		

(c)

R1) $<$ 0.02: unpredictive;
R2) 0.02 to 0.1: weak predictor;
R3) 0.1 to 0.3: medium predictor;
R4) 0.3 to 0.5: strong predictor; and
R5) $\geq$ 0.5: suspicious or too good to be true.

Before converting our datasets into images, we first use IV to select predictive features, i.e., a feature f is selected if $IV^{(f)} \geq 0.1$.

C. CONVERSION OF TABULAR DATA INTO IMAGES

Once the data is transformed using bins for WOE calculation, we then create a *one-hot-encoding* transformation for each binned feature. Below we show a matrix example for each applicant, i.e. $\forall i \in \{1, 2, \cdots, N\}$ where N is the number of applicants:

$$\mathbf{x}^{(i)} = \begin{bmatrix} 0 & 0 & 1 & \cdots & 0 \\ 1 & 0 & 0 & \cdots & 1 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix}_{B \times D}, \tag{5}$$

where $B = \max\{B^{(f)}\}$ which is a maximum number of bins for feature f.

Let $\mathbf{x}^{(i)}$ be a feature image representation of record $\mathbf{f}^{(i)} = [f_1^{(i)}, f_2^{(i)}, \ldots, f_D^{(i)}]$ and $y^{(i)}$ be a corresponding label for a dataset $\{\mathbf{f}^{(i)}, y^{(i)}\}_{i=1}^{N}$ of N records. The feature image is nothing but a sparse binary matrix (see Eq. (5)) of size $B \times D$ representing one-hot-encoding of each feature according to

$$\mathbf{x}^{(i)} = \left[E(f_1^{(i)}), E(f_2^{(i)}), \ldots, E(f_D^{(i)})\right], \tag{6}$$

where $E : f_j \mapsto \{0, 1\}^B$ is a function to perform one-hot-encoding for a feature f_j and B is defined as a maximum number of distinct bins of features, i.e. $B = \max\{B^{(f_j)}\}_{j=1}^{D}$. It is worth to note that for some features $B^{(f_j)} < B$ and in such cases, function $E(f_j)$ appends $(B - B^{(f_j)})$ zeros to one-hot-encoding representation of f_j. It is also worth to note that features become nominal with one-hot-encoding, hence some features in Table 2 do not follow a logical order. For example, the feature `Duration of Credit` in Table 2(b) has bin [13, 24] as the first bin and [4, 12] as the last bin. For this feature, one would expect the bins to be arranged in an increasing order, however it is not always the case with one-hot-encoding.

Each entry of $\mathbf{x}^{(i)}$ is then mapped into a pixel, where ones correspond to white pixels (and white pixels symbolise the presence of a bin) and zeros to black pixels (and black pixels

symbolise the absence of a bin). Thus, each white pixel of an image represents a defined bin for WOEs calculations.

D. CONVOLUTIONAL NEURAL NETWORKS

Convolutional Neural Networks (CNNs) [34] are widely known for solving tasks in image recognition, speech recognition and time series problems. The CNN consists of *convolutional layers*, *pooling layers* and *dense layers* (please see Figure 1). In what follows we briefly discuss layers of a CNN. For more details on CNNs please refer to our previous paper [2].

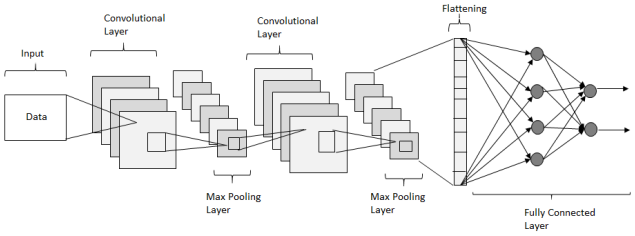

FIGURE 1. Schematic view of a Convolutional Neural Network [2].

1) INPUT

In image classification tasks, an input is a *tensor* of shape (*height*, *width*, *channel*). The channel represents a color of an image where 1 denotes a gray-scale image and 3 denotes an image with color/s.

2) CONVOLUTIONAL LAYER

Local patterns in image classification such as edges and textures are learned at convolutional layers [35]. Each convolutional layer consists of feature/activation maps that are responsible for learning different local patterns in an image. The feature maps are formed by sliding a learned *feature detector* also known as a learned filter on different parts of an image. Thereafter, a non-linear function such as the Rectified Linear Unit (ReLU) function is applied on feature maps.

3) POOLING LAYER

A pooling layer is responsible for downsizing feature maps by performing either max pooling or average pooling. The *spatial invariance* occurs at a pooling layer to assure that features are preserved irrespective of where they are located in an image.

4) FLATTENING

Flattening transforms all pooled feature maps into a single vector which acts as an input in a dense layer/fully connected layer.

5) CALCULATION IN EACH LAYER OF CNN

Normally, a deep learning neural network learns a set of parameters such as the weights during training. The parameters are learned in the convolutional layers and dense layers. To calculate the number of learnable parameters in the convolutional layers, we multiply the shapes of the filters such as the width w, height h, previous layer's filters p and filters of the current layer c resulting in

$$\#\text{ of parameters} = ((w \times h \times p) + 1) \times c), \tag{7}$$

where 1 is for the bias term, w is the shape of width of the filter, h is the shape of height of the filter, p is the number of filters in the previous layer and c is the number of filters in the current layer. In the convolutional layers, a feature map (i.e. a matrix of numbers) is produced by striding a filter along an input image (in the input layer) or along another feature map (in the convolutional layer). Each entry of the feature map is calculated as follows

$$K[r, c] = (f * t)[r, c] = \sum_{i=1}^{r} \sum_{j=1}^{c} t[i, j] f[r - i, c - j], \tag{8}$$

where r and c represent rows and columns, respectively, of a resulting feature map, f denotes an input image, t represents a kernel or a filter, $*$ denotes a convolutional operator. The feature map $\mathbf{A}^{(l-1)}$ is then multiplied by a weight matrix $\mathbf{W}^{(l)}$ and added with a bias term $\mathbf{b}^{(l)}$ to form

$$\mathbf{Z}^{(l)} = \mathbf{A}^{(l-1)} \cdot \mathbf{W}^{(l)} + \mathbf{b}^{(l)}, \tag{9}$$

and a ReLU function σ is applied on $\mathbf{Z}^{(l)}$ to propagate the feature maps into subsequent convolutional layers

$$\mathbf{A}^{(l-1)} = \sigma^{(l)}(\mathbf{Z}^{(l)}), \tag{10}$$

where l denotes the l-th convolutional layer. In the pooling layer, either a max pooling or mean pooling is applied to produce down-sized feature maps. Each entry of the pooled feature map is

$$P[i, j] = max[\mathbf{A}^{*}_{i,j} \subset \mathbf{A}^{(l)}]. \tag{11}$$

In the dense layer, the number of learn-able parameters is

$$\#\text{ of parameters} = ((c \times p) + 1 \times c), \tag{12}$$

where c denotes current layer neurons, p denotes previous layer neurons and 1 is for the bias term. For dense layers or fully connected layers, each neuron in the l-th layer receives signals from neurons of the previous layers $l - 1$. The signals are multiplied by the corresponding weights and are summed together with a bias term and thereafter a transfer function is applied on the summed product to form an activation. Suppose $w_{ji}^{(l)}$ is a weight connection from neuron i to neuron j in the l-th layer. The activation $a_j^{(1)}$ of each neuron in the first hidden layer of the fully connected layers is

$$a_j^{(1)} = \sigma(\sum_{i=1}^{d} w_{ji}^{(1)} x_i + b_j^{(1)}) \tag{13}$$

and the activation in the subsequent hidden layers is

$$a_k^{(l)} = \sigma(\sum_{j=1}^{n} w_{kj}^{(l)} a_j^{(l-1)} + b_k^{(l)}) \tag{14}$$

where d denotes the number of input features, n denotes the number of hidden neurons in hidden layer $l-1$ and σ is a transfer function (e.g sigmoid or ReLU function). In the output layer of the fully connected layers, a binary cross-entropy function is

$$H(p,q) = -\frac{1}{2}\sum_{i=1}^{2} y_i log(p(y_i)) + (1-y_i)log(q)), \quad (15)$$

where $p(y_i)$ is the predicted output, y_i is the actual target value and $q = 1 - p(y_i)$. The binary cross-entropy function is used to propagate errors backwards to update the weights of the CNN during training.

E. EXPLANATIONS

1) SALIENCY MAP

A Saliency Map [9] provides a visual explanation by highlighting important regions of an image that result in a prediction of a specific class. Specifically in this study, the image pixels correspond to bins that were created for WOE calculations. There are several variants of Saliency Map explanations, e.g., gradient explanation, gradient class activation map (Grad-CAM) explanation and layer-wise relevance propagation (LRP) explanation to mention a few. However, in this study we focused on gradient explanation and Grad-CAM. For an image $\mathbf{x}$ belonging to a class c, the Saliency Map $\mathbf{M}$ is computed as a derivative

$$\omega = \frac{\partial \hat{g}(\mathbf{x})}{\partial \mathbf{x}} \quad (16)$$

of a prediction $\hat{g}(\mathbf{x})$ with respect to an input image $\mathbf{x}$ where $\hat{g}$ is a cost function. Each pixel of $\mathbf{M}$ corresponds to the importance of the relative pixel of an input image [32], which is normally a class-specific score. The gradient measures how much a change in each input dimension would change the prediction $\hat{g}(\mathbf{x})$ in a small neighborhood around the input [32].

2) GRAD-CAM

Gradient weighted Class Activation Map (Grad-CAM) [36] is determined using a last convolutional layer of a CNN model. Firstly, a gradient of the score with respect to an activation map of the last convolutional layer is determined via a back-propagation approach

$$\frac{\partial y^c}{A^k},$$

where y^c is a score for class c and A^k is a feature map activation. Secondly, the neuron importance weights are calculated as follows

$$\alpha_k^c = \frac{1}{Z}\sum_i \sum_j \frac{\partial y^c}{A_{ij}^k},$$

where $\frac{1}{Z}\sum_i \sum_j$ is a global average pooling and $Z = i \times j$. Lastly, each importance weight is multiplied by the corresponding feature map activation and the products are summed up to form a saliency map given as follows

$$\mathcal{L}_{Grad-CAM}^c = \sum_k \alpha_k^c A^k,$$

where $\mathcal{L}_{Grad-CAM}^c \in \mathbb{R}^{u \times v}$ and $u \times v$ represent the size of the feature map. The saliency map that is produced by the Grad-CAM highlights the important regions in an image that are corresponding to a predicted class.

3) LIME

Local Interpretable Model-Agnostic Explanations (LIME) [37] is used to explain predictions of black-box models. LIME uses for example a decision tree or a linear model around an instance of interest to explain the instance's class prediction [37]. The idea behind explaining predictions is to foster trust in predictions if any actions need to be taken. For example, if a credit analyst depends on a model for granting/rejecting a loan application, he/she will need to have a model's prediction explanation in order to trust the prediction and subsequently make a decision. We use LIME to create explanations as follows.

a: CREATE PERTURBATIONS OF AN IMAGE OF INTEREST:

Let $\mathbf{x}$ be an *original representation* of an image. Image $\mathbf{x}$ is then segmented into M super-pixels. A super-pixel is a group of pixels in an image. Let K be a number of perturbed images. A perturbation of image $\mathbf{x}$ is created by randomly turning on and off some of the super-pixels. This results into $\hat{\mathbf{X}} = \{0,1\}^{K \times M}$ where $M = B \times D$ (i.e. B is the maximum number of bins and D is the number of features) and each row of $\hat{\mathbf{X}}$ is an *interpretable representation* of each perturbed image. Here, a 1 represents that a super-pixel is on and a 0 represents that a super-pixel is off.

b: PREDICT CLASSES OF THE PERTURBED IMAGES:

A non-linear machine learning model g (which is in our case the model we have trained on the images) is used to predict a class of each perturbed image $\mathbf{x}^{(i)\prime}$. The prediction of $\mathbf{x}^{(i)\prime}$ is $\hat{g}(\mathbf{x}^{(i)\prime})$.

c: COMPUTE DISTANCES BETWEEN THE ORIGINAL IMAGE AND EACH OF THE PERTURBED IMAGES:

The distance between each perturbed image and the image being explained is computed using the cosine distance. The cosine distance is given as

$$D_i(\mathbf{x}, \mathbf{x}^{(i)\prime}) = \frac{\mathbf{x} \cdot \mathbf{x}^{(i)\prime}}{\|\mathbf{x}\| \|\mathbf{x}^{(i)\prime}\|}, \quad (17)$$

$\forall i = \{1, 2, \cdots, K\}$ where $\mathbf{x} = \{1\}_{i=1}^{M}$ is the original image with all super-pixels enabled and $\mathbf{x}^{(i)\prime} = \{0,1\}_{i=1}^{M}$ are perturbed images.

d: COMPUTE THE WEIGHT FOR EACH PERTURBED IMAGE:

This is achieved by mapping each of the distances calculated above into a kernel function that has values between

zero and one. This is shown by the formula below

$$\pi_{\mathbf{x}^{(i)\prime}} = \exp\big(-D_i(\mathbf{x}, \mathbf{x}^{(i)\prime})^2/\sigma^2\big), \tag{18}$$

where σ is the width of the kernel.

e: FIT A SPARSE LINEAR MODEL:

The interpretable representations $\hat{\mathbf{X}}$ of the perturbed images, the predictions $\hat{g}(\mathbf{x}^{(i)\prime})$ and the weights $\pi_{\mathbf{x}^{(i)\prime}}$ are then used to fit a sparse linear model. The sparse linear model is given as

$$\psi(\mathbf{x}^{(i)\prime}, \hat{g}(\mathbf{x}^{(i)\prime}), \pi_{\mathbf{x}^{(i)\prime}}) = \beta_0^{(i)} + \sum_{j=1}^{M} \beta_j^{(i)}(\pi_{\mathbf{x}^{(i)\prime}} x_j^{(i)\prime}), \tag{19}$$

$\forall i = \{1, 2, \cdots, K\}$ where M represents the number of super-pixels. Each coefficient $\beta_j^{(i)}$ in the sparse linear model corresponds to one super-pixel in the segmented image. The coefficients represent how important each super-pixel is for the prediction of the class of interest. Thus, the top super-pixels (in terms of the coefficient magnitudes) are used to explain the prediction made by the model g.

4) SHAP VALUES

The SHAP values [38] use an idea of coalition game theory, where there are players, a game and a payout. The aim of game theory is to distribute the payout between the players based on their contributions. In machine learning, the players are the feature values of a tabular dataset, the game is the prediction task, the payout is the prediction minus the average prediction for all dataset records. Hence, the SHAP value for any record is the average marginal contribution of a feature across all possible feature coalitions. Suppose we have a linear model and we want to explain a prediction for a record $\mathbf{f}^{(i)}$. The prediction for $\mathbf{f}^{(i)}$ is

$$\hat{g}(\mathbf{f}^{(i)}) = \beta_0 + \beta_1 f_1^{(i)} + \cdots + \beta_p f_d^{(i)}, \tag{20}$$

where $f_1^{(i)}, f_2^{(i)}, \cdots, f_d^{(i)}$ are feature values. Let ϕ_j be a contribution of feature j on prediction $\hat{g}(\mathbf{f}^{(i)})$. The contribution ϕ_j is given as

$$\begin{aligned}\phi_j(\hat{g}) &= \beta_j f_j^{(i)} - E(\beta_j \mathbf{F}_j)\\ &= \beta_j f_j^{(i)} - \beta_j E(\mathbf{F}_j),\end{aligned}$$

where $E(\mathbf{F}_j)$ is the expected value for feature j. If all feature contributions are summed up for one record, the result is

$$\begin{aligned}\sum_{j=1}^{d} \phi_j(\hat{g}) &= \sum_{j=1}^{d}(\beta_j f_j^{(i)} - E(\beta_j \mathbf{F}_j))\\ &= (\beta_0 + \sum_{j=1}^{d} \beta_j f_j^{(i)}) - (\beta_0 + \sum_{j=1}^{d} E(\beta_j \mathbf{F}_j))\\ &= \hat{g}(\mathbf{f}^{(i)}) - E(\hat{g}(\mathbf{F})),\end{aligned}$$

which is the difference between the predicted value for record $\mathbf{f}^{(i)}$ and the average predicted value for dataset $\mathbf{F} = \{\mathbf{F}_j\}_{j=1}^{d}$ where $\mathbf{F}_j = \{f_{ij}\}_{i=1}^{N}$.

For machine learning models, we determine M coalitions/combinations of feature values for a record $\mathbf{f}^{(i)}$. There are 2^d possible coalitions for feature values and this makes computations to be expensive and as a result we choose M such that $M \ll 2^d$. In order to assess contributions of each feature when explaining a prediction of a record, we make predictions for all M coalitions. Since we are using a subset of coalitions, the contribution of each feature is an estimate given as

$$\hat{\phi}_j = \frac{1}{M}\sum_{m=1}^{M}(\hat{g}(\mathbf{f}_{+_j}^m) - \hat{g}(\mathbf{f}_{-_j}^m)), \tag{21}$$

where $\hat{g}(\mathbf{f}_{+_j}^m)$ is the prediction for record $\mathbf{f}$, but with a random number of feature values replaced by some feature values from a random data point $\mathbf{z}$ in dataset $\mathbf{F}$ with the exception of the respective value for feature j. The prediction $\hat{g}(\mathbf{f}_{-_j}^m)$ is identical to $\hat{g}(\mathbf{f}_{+_j}^m)$ but the respective value of feature j is taken from a feature value of $\mathbf{z}$. Let

$$\mathbf{f} = (f_1, f_2, \cdots, f_d) \tag{22}$$

and

$$\mathbf{z} = (z_1, z_2, \cdots, z_d) \tag{23}$$

then

$$\mathbf{f}_{+_j}^m = (f_1, \cdots, f_{j-1}, f_j, z_{j+1}, \cdots, z_d) \tag{24}$$

and

$$\mathbf{f}_{-_j}^m = (f_1, \cdots, f_{j-1}, z_j, z_{j+1}, \cdots, z_d). \tag{25}$$

In general, Eq. (21) is

$$\phi_j(g) = \sum_{S \subseteq \{f_1,\ldots,f_d\}\setminus\{f_j\}} \frac{|S|!(d-|S|-1)!}{d!}(\hat{g}(S \cup \{f_j\}) - \hat{g}(S)), \tag{26}$$

where $d! = (d) \times (d-1) \times (d-2) \times \cdots \times (3) \times (2) \times (1)$ and S is a subset of feature values for record $\mathbf{f}$ that needs to be explained.

For images, a player is a group of features, for example, pixels can be grouped together to form super-pixels. Hence, each super-pixel represents a player.

V. EXPERIMENTS

A. DATA

Three real world credit datasets were used in this study, i.e. German, Australian [39] and Home Loan Equity (HMEQ) [40]. The number of samples, the number of features and the types of features for each dataset are shown in Table 3. All three datasets are publicly available credit scoring datasets on *UCI* repository and *Kaggle*. The first two datasets are the most used datasets in credit scoring literature [2]. The German credit dataset has 20 features of which 7 are numerical and 13 are categorical. These features include `status of existing checking account`, `duration in month`, `credit history` and `purpose`, to mention

TABLE 3. Credit datasets with sample sizes, number of defaults, number of non-defaults, number of categorical features and number of continuous features.

	Dataset		
	German	Australian	HMEQ
sample size	1,000	690	5,960
# of non-defaults	700	307	4,771
# of defaults	300	383	1,189
# of features	20	14	13
categorical	13	8	2
continuous	7	6	11

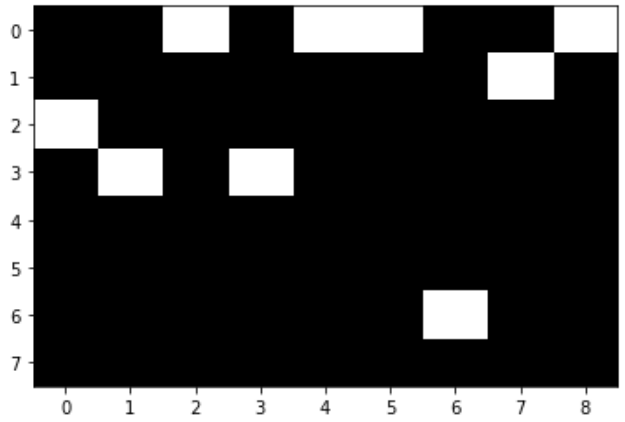

FIGURE 2. An example of applicant information is converted into a binary image for HMEQ tabular dataset. The columns represent features and the rows represent bins for WOE calculations.

a few. The Australian credit dataset has in total 14 features and 6 of the features are numerical and 8 are categorical. The original feature names have been changed for confidential purposes. The HMEQ dataset has 13 features of which 11 are numerical and 2 are categorical. The features include `amount of loan request`, `amount due on existing mortgage`, `value of the current property`, `years at present job`, `number of credit lines` and etc. The `response variable` for all three datasets is binary, i.e. applicants are classified either as "defaults" or "non-defaults". Table 2 shows WOE bins for each feature in German, Australian and HMEQ datasets. Note that unpredictive and weak features were removed according to the information value. Figure 2 shows an example where an applicant information is converted into an image, and in Figure 2 the columns represent features and the rows represent bins for WOE calculations.

B. MODEL

Each of the datasets was split into 70% training set and 30% test set following the common practise in the literature. The samples in each split were shuffled and thereafter stratified random sampling was performed for each split. The training set was used to determine the optimal parameters of the CNN model. The test set was used to assess the performance of the CNN model. Table 4, Table 5 and Table 6 show different

TABLE 4. CNN Architecture #1: Parameters and architectures of the CNN model in (a) Australian dataset, (b) German dataset, and (c) HMEQ dataset. Legend: Conv (Convolutional Layer), PL (Pooling Layer), FC (Fully Connected Layer).

Layer	Parameters and Architecture
Input	input shape: $k = 14$, $d = 12$, c=1
Conv1	filters:128, kernel size: 3×3, , dropout: 0.25
Conv2	filters: 256, kernel size: 3×3, dropout: 0.25
FC1	neurons: 64, dropout: 0.25
FC2	neurons: 2, activation: softmax
Activation functions	ReLU

(a)

Layer	Parameters and Architecture
Input	input shape:$k = 5$, $d = 7$, c=1
Conv1	filters:64, kernel size: 3×3, dropout: 0.25
Conv2	filters: 128, kernel size: 3×3, dropout: 0.25
Conv3	filters: 256, kernel size: 3×3, dropout: 0.25
PL	type: max, size: 2×2, dropout: 0.25
FC1	neurons: 128, dropout: 0.5
FC2	neurons: 2, activation: softmax
Activation functions	ReLU

(b)

Layer	Parameters and Architecture
Input	input shape:$k = 8$, $d = 9$, c=1
Conv1	filters:128, kernel size: 3×3, dropout: 0.25
Conv2	filters: 256, kernel size: 3×3, dropout: 0.25
FC1	neurons: 64, dropout: 0.25
FC2	neurons: 2, activation: softmax
Activation functions	ReLU

(c)

TABLE 5. CNN Architecture #2: Parameters and architectures of the second CNN model in (a) Australian dataset, (b) German dataset, and (c) HMEQ dataset. Legend: Conv (Convolutional Layer), PL (Pooling Layer), FC (Fully Connected Layer).

Layer	Parameters and Architecture
Input	input shape: $k = 14$, $d = 12$, c=1
Conv1	filters:16, kernel size: 3×3, dropout: 0.25
Conv2	filters: 32, kernel size: 3×3, dropout: 0.25
Conv3	filters: 64, kernel size: 3×3, dropout: 0.25
PL	type: max, size: 2×2
FC1	neurons: 32
FC2	neurons: 64, dropout: 0.5
FC3	neurons: 2, activation: softmax
Activation functions	Tanh

(a)

Layer	Parameters and Architecture
Input	input shape: $k = 5$, $d = 7$, c=1
Conv1	filters:16, kernel size: 3×3, dropout: 0.25
Conv2	filters: 32, kernel size: 3×3, dropout: 0.25
Conv3	filters: 64, kernel size: 3×3, dropout: 0.25
PL	type: max, size: 2×2
FC1	neurons: 64, dropout: 0.5
FC2	neurons: 2, activation: softmax
Activation functions	Tanh

(b)

Layer	Parameters and Architecture
Input	input shape: $k = 8$, $d = 9$, c=1
Conv1	filters:16, kernel size: 3×3, dropout: 0.25
Conv2	filters: 32, kernel size: 3×3, dropout: 0.25
Conv3 #2	filters: 64, kernel size: 3×3, dropout: 0.25
PL	type: max, size: 2×2, dropou=0.25
FC1	neurons: 32
FC2	neurons: 64, dropout: 0.5
FC3	neurons: 2, activation: softmax
Activation functions	Tanh

(c)

architectures of the CNN models for Australian, German and HMEQ datasets, respectively. For Australian dataset, the CNN model was trained using 100 *epochs* and a *batch*

TABLE 6. CNN Architecture #3: Parameters and architectures of the third CNN model in (a) Australian dataset, (b) German dataset, and (c) HMEQ dataset. Legend: Conv (Convolutional Layer), PL (Pooling Layer), FC (Fully Connected Layer).

Layer	Parameters and Architecture
Input	input shape: $k = 14$, $d = 12$, c=1
Conv1	filters:8, kernel size: 3×3
Conv2	filters: 16, kernel size: 3×3
Conv3	filters: 32, kernel size: 3×3
FC1	neurons: 16
FC2	neurons: 32
FC3	neurons: 2, activation: softmax
Activation functions	Selu

(a)

Layer	Parameters and Architecture
Input	input shape: $k = 5$, $d = 7$, c=1
Conv1	filters:8, kernel size: 3×3,
Conv2	filters: 16, kernel size: 3×3
Conv3	filters: 32, kernel size: 3×3
FC1	neurons: 16
FC2	neurons: 32
FC3	neurons: 2, activation: softmax
Activation functions	Selu

(b)

Layer	Parameters and Architecture
Input	input shape: $k = 8$, $d = 9$, c=1
Conv1	filters:8, kernel size: 3×3
Conv2	filters: 16, kernel size: 3×3
Conv3	filters: 32, kernel size: 3×3
FC1	neurons: 16
FC2	neurons: 32
FC3	neurons: 2, activation: softmax
Activation functions	Selu

(c)

TABLE 7. CNN Architectures: Parameters and architectures for 1D CNN model: (i) 1D CNN Architecture #1, (ii) 1D CNN Architecture #2, Legend: Conv (Convolutional Layer), PL (Pooling Layer), FC (Fully Connected Layer).

Layer	Parameters for 1D CNN Architecture #1
Conv1	filters:128, kernel size: 1×3, dropout: 0.25
PL1	type: max, size: 1×2
Conv2	filters: 256, kernel size: 1×3, dropout: 0.25
PL2	type: max, size: 1×2
FC1	neurons: 256, dropout: 0.25
FC2	neurons: 2, activation: softmax
Activation functions	ReLU

(i)

Layer	Parameters for 1D CNN Architecture #2
Conv1	filters:16, kernel size: 1×3
Conv2	filters: 32, kernel size: 1×3
Conv3	filters: 64, kernel size: 1×3
FC1	neurons: 64
FC2	neurons: 32
FC3	neurons: 2, activation: softmax
Activation functions	Tanh

(ii)

size of 54. For both German and HMEQ datasets, the CNN model was trained with 100 epochs and a batch size of 64. The 2D CNN model was then compared to a 1D CNN model. The 1D CNN model was trained using 1000 epochs and a batch size of 32 on each dataset. The purpose of using a 1D CNN model is to see the effect of the model when the data is not converted into images. For 1D CNN model, we used the raw tabular dataset. Hence, Table 7 shows two different architectures for a 1D CNN model. Please note that 2D CNNs are not applicable to raw tabular datasets, hence we opted for a 1D CNN model.

C. PREDICTION PERFORMANCE MEASURES

For each of the datasets, we calculated "Accuracy" which measures the proportion of the correctly classified examples and is defined as

$$\text{Accuracy} = \frac{(TP + TN)}{(TP + FP + FN + TN)}. \tag{27}$$

We also calculated other performance metrics which are defined next. The "Area Under the Curev" (AUC) [15] which is a metric that measures the predictive power of a model is calculated as follows

$$\text{AUC} = \frac{1}{2}\left(1 + \frac{TP}{TP + FN} - \frac{FP}{FP + TN}\right). \tag{28}$$

The higher the AUC, the better the predictive power of the model. The "Brier Score", named after Glenn Brier [41], calculates the mean squared error between the predicted probabilities and their respective true class values and is calculated as follows

$$\text{Brier Score} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2, \tag{29}$$

where $\hat{y}_i$ is the predicted probability of the true class, y_i is the true class and N is the number of records. The Brier Score takes values between 0 and 1. The lower the Brier Score, the better the predictions are calibrated. On the other hand the "H-measure" proposed by Hand [42] assesses the cost of misclassification and it uses a severity ratio (SR) given as

$$\text{SR} = \frac{c_0}{c_1}, \tag{30}$$

where $c_0 > 0$ is the cost of misclassifying a class 0 record as class 1. Hence, the higher the H-measure the lower the misclassification cost.

D. EXPLANATIONS

After training the CNN models and assessing their performances, explanation methods such as LIME, SHAP values, Grad-CAM and Saliency Map were used to explain individual predictions. For LIME, we set the kernel width $\sigma^2 = 0.25$ and the number of perturbed images $K = 150$. This study focused on *local explanations* (i.e. individual explanation) as opposed to *global explanations* (i.e. explaining holistically how the model makes predictions).

VI. RESULTS

A. MODEL PERFORMANCE

We created three CNN architectures (please refer to Table 4, Table 5 and Table 6) to perform contrast experiments on the design of the architectures. For each architecture we used different activation functions, different number of layers and different number of filters in each layer. Also, Table 8, Table 9 and Table 10 show confusion matrices, AUC, Brier Score and H-measure for each of the CNN architectures on all three

TABLE 8. Performance metrics resulted from the CNN Architecture #1 model for all three datasets for both training and test sets.

Dataset	Test (30%)		Train (70%)	
German	TP = 72	FN = 18	TP = 173	FN = 37
	FP = 19	TN = 191	FP = 36	TN = 454
	Accuracy = 0.88		Accuracy = 0.90	
	AUC = 0.80		AUC =0.79	
	Brier Score = 0.21		Brier Score = 0.22	
	H-measure = 0.42		H-measure = 0.42	
Australian	TP = 111	FN = 4	TP = 260	FN = 8
	FP = 6	TN = 86	FP = 8	TN = 207
	Accuracy = 0.95		Accuracy = 0.97	
	AUC = 0.96		AUC = 0.96	
	Brier Score = 0.54		Brier Score = 0.54	
	H-measure = 0.89		H-measure = 0.88	
HMEQ	TP = 1181	FN = 250	TP =2742	FN = 598
	FP = 69	TN = 288	FP = 145	TN = 687
	Accuracy = 0.82		Accuracy = 0.82	
	AUC = 0.83		AUC = 0.82	
	Brier Score = 0.16		Brier Score = 0.18	
	H-measure = 0.44		H-measure = 0.44	

TABLE 9. Performance metrics resulted from the CNN Architecture #2 model for all three datasets for both training and test sets.

Dataset	Test (30%)		Train (70%)	
German	TP = 41	FN = 49	TP =147	FN =63
	FP = 34	TN =176	FP = 18	TN = 472
	Accuracy = 0.72		Accuracy = 0.88	
	AUC = 0.65		AUC = 0.83	
	Brier Score = 0.21		Brier Score = 0.20	
	H-measure = 0.11		H-measure =0.54	
Australian	TP = 98	FN = 17	TP = 250	FN = 18
	FP = 14	TN = 78	FP = 6	TN = 209
	Accuracy = 0.85		Accuracy = 0.95	
	AUC = 0.85		AUC = 0.95	
	Brier Score = 0.50		Brier Score = 0.50	
	H-measure = 0.54		H-measure = 0.85	
HMEQ	TP =1398	FN =33	TP =	FN = 42
	FP = 208	TN = 149	FP = 411	TN = 421
	Accuracy = 0.87		Accuracy = 0.89	
	AUC = 0.70		AUC = 0.75	
	Brier Score = 0.13		Brier Score = 0.11	
	H-measure = 0.41		H-measure = 0.41	

datasets. Amongst the three tables, Table 8 which corresponds to CNN Architecture #1 shows better performances in almost all three datasets. Hence, our proposed method is based on CNN Architecture #1. The results show that the proposed method achieves similar performances on both training and test set, hence it does achieve a good generalisation performance. We compared the 2D CNN model with 1D CNN model to see the impact of converting data into images. Table 11 and Table 12 show the results of the 1D CNN model. It can be seen that when comparing the results in Table 8 with the results in Table 11 and Table 12, the 2D CNN model performs better than the 1D CNN model on German and Australian datasets when looking at all the performance metrics. However, on the HMEQ datasets, both 2D CNN and 1D CNN perform similarly regarding accuracies and brier scores, but 2D CNN show better results when looking at the AUC and the H-measure. These results support the conversion of tabular datasets into images in order to leverage the high performance nature of 2D CNNs.

TABLE 10. Performance metrics resulted from the CNN Architecture #3 model for all three datasets for both training and test sets.

Dataset	Test (30%)		Train (70%)	
German	TP = 49	FN = 41	TP =181	FN =29
	FP = 40	TN =170	FP = 15	TN = 475
	Accuracy = 0.73		Accuracy = 0.94	
	AUC = 0.68		AUC = 0.92	
	Brier Score = 0.26		Brier Score = 0.25	
	H-measure = 0.15		H-measure = 0.75	
Australian	TP =103	FN =12	TP =260	FN =8
	FP =13	TN =79	FP =3	TN =212
	Accuracy = 0.88		Accuracy = 0.98	
	AUC =0.88		AUC =0.98	
	Brier Score =0.52		Brier Score = 0.53	
	H-measure = 0.62		H-measure = 0.93	
HMEQ	TP =1370	FN = 61	TP =3296	FN = 44
	FP = 184	TN =173	FP = 284	TN = 548
	Accuracy = 0.86		Accuracy = 0.92	
	AUC = 0.72		AUC = 0.82	
	Brier Score = 0.13		Brier Score = 0.08	
	H-measure = 0.57		H-measure = 0.57	

TABLE 11. Performance metrics resulted from the 1D CNN Architecture #1 model for all three datasets for both training and test sets.

Dataset	Test (30%)		Train (70%)	
German	TP = 48	FN = 42	TP = 116	FN = 94
	FP = 41	TN = 169	FP = 64	TN = 426
	Accuracy = 0.72		Accuracy = 0.77	
	AUC = 0.67		AUC = 0.71	
	Brier Score = 0.28		Brier Score = 0.25	
	H-measure = 0.10		H-measure = 0.27	
Australian	TP = 96	FN = 19	TP = 250	FN = 18
	FP = 23	TN =69	FP = 14	TN = 201
	Accuracy = 0.79		Accuracy = 0.93	
	AUC = 0.79		AUC = 0.93	
	Brier Score = 0.20		Brier Score = 0.06	
	H-measure = 0.41		H-measure = 0.74	
HMEQ	TP = 1415	FN = 16	TP = 3272	FN = 68
	FP = 310	TN = 47	FP = 684	TN = 148
	Accuracy = 0.82		Accuracy = 0.82	
	AUC = 0.56		AUC = 0.56	
	Brier Score = 0.18		Brier Score = 0.18	
	H-measure = 0.07		H-measure = 0.08	

The results that were found in literature which are shown in Table 13 were compared with the proposed method results. Hence, the proposed method shows in most cases better performances compared to the methods from the literature considered in this paper. The results in Table 13 and Table 8 support the efficacy of the proposed 2D CNN model in accurately assessing the credit applicants. However, we also want to make sure that we have meaningful and good explanations for the proposed model's decision. In the following, we qualitatively and quantitatively evaluate the explanations of the proposed method generated by four different explanation methods (i.e LIME, Saliency Map, Grad-CAM and SHAP values).

B. EXPLANATIONS

For each of the datasets, we randomly selected an image representing the corresponding record in the tabular dataset. Thereafter, we predicted each class for each of the randomly

TABLE 12. Performance metrics resulted from the 1D CNN Architecture #2 model for all three datasets for both training and test sets.

Dataset	Test (30%)		Train (70%)	
German	TP = 32	FN = 58	TP = 75	FN =135
	FP = 22	TN = 188	FP = 22	TN = 468
	Accuracy = 0.73		Accuracy = 0.78	
	AUC = 0.63		AUC = 0.66	
	Brier Score = 0.27		Brier Score = 0.22	
	H-measure = 0.13		H-measure = 0.19	
Australian	TP = 92	FN = 23	TP = 226	FN = 42
	FP = 28	TN = 64	FP = 39	TN = 176
	Accuracy = 0.75		Accuracy = 0.83	
	AUC = 0.75		AUC = 0.83	
	Brier Score = 0.25		Brier Score = 0.17	
	H-measure = 0.34		H-measure = 0.66	
HMEQ	TP =1422	FN = 9	TP = 3294	FN =46
	FP =323	TN = 34	FP = 739	TN = 93
	Accuracy = 0.81		Accuracy = 0.81	
	AUC = 0.54		AUC = 0.55	
	Brier Score = 0.19		Brier Score = 0.19	
	H-measure = 0.06		H-measure = 0.05	

TABLE 13. Performances of related models in credit scoring literature.

Source	Year	German	Astralian	HMEQ
		Accuracy	Accuracy	Accuracy
[43]	2000	0.76	0.87	-
[19]	2016	0.75	0.86	-
[44]	2016	0.78	0.88	-
[45]	2017	0.77	0.88	-
[46]	2018	0.77	0.87	-
[22]	2018	0.86	0.93	
[23]	2018	0.87	**0.95**	
[21]	2020	0.81	0.90	
Our Study	2021	**0.88**	**0.95**	0.82

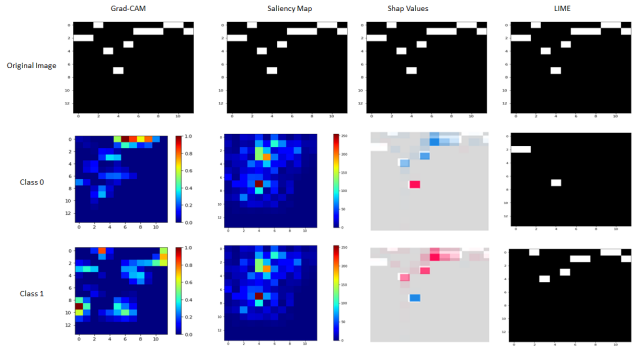

FIGURE 3. The rows in this figure refer to the original image, class 0 and class 1 and the columns refer to the Grad-CAM, Saliency Map, Shap values and LIME. Prediction explanation for a random image belonging to class 0 on Australian dataset. Legend: class 0 (non-defaults), class 1 (defaults), column 0 (A2), column 1 (A3), column 2 (A4), column 3 (A5), column 4 (A6), column 5 (A7), column 6 (A8), column 7 (A9), column 8 (A10), column 9 (A12), column 10 (A13) and column 11 (A14). All the (A's) are feature names. All columns are features from tabular dataset and the rows are bins for WOE calculations. See Table 2 for further details on features.

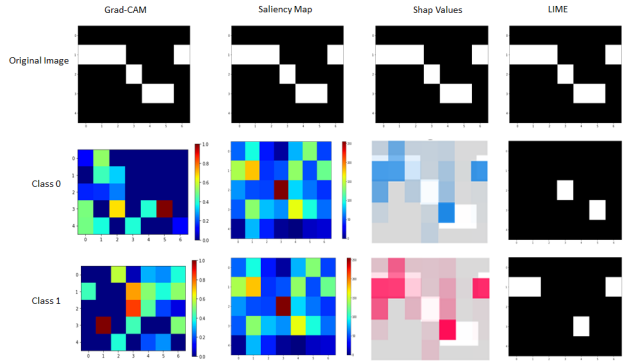

FIGURE 4. The rows in this figure refer to the original image, class 0 and class 1 and the columns refer to the Grad-CAM, Saliency Map, Shap values and LIME. Prediction explanation for a random image belonging to class 1 on German dataset. Legend: class 0 (non-defaults), class 1 (defaults), column 0 (Acc Bal), column 1 (Age), column 2 (DCredit), column 3 (LEmploy), column 4 (Asset), column 5 (VStocks) and column 6 (PStatus). All columns are features from tabular dataset and the rows are bins for WOE calculations. See Table 2 for further details on features.

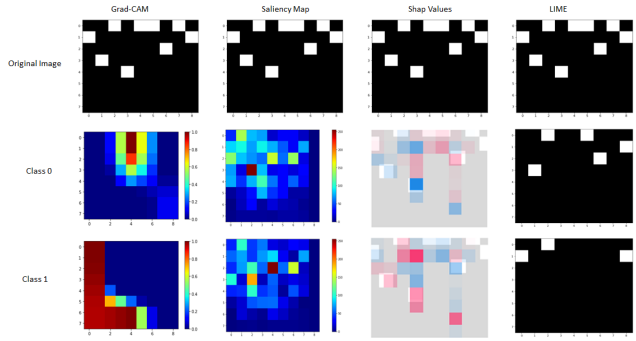

FIGURE 5. The rows in this figure refer to the original image, class 0 and class 1 and the columns refer to the Grad-CAM, Saliency Map, Shap values and LIME. Prediction explanation for a random image belonging to class 1 on HMEQ dataset. Legend: class 0 (non-defaults), class 1 (defaults), column 0 (Loan), column 1 (Mortdue), column 2 (Value), column 3 (Job), column 4 (Derog), column 5 (Delinq), column 6 (Clage), column 7 (Ninq) and column 8 (Debtinc). All columns are features from tabular dataset and the rows are bins for WOE calculations. See Table 2 for further details on features.

selected images. We then used Grad-CAM, LIME, SHAP values and Saliency Map to explain each predicted class (i.e. defaults = 1 and non-defaults = 0). Figure 3, Figure 4 and Figure 5 show explanations for each of the predictions. The white blocks/pixels in LIME are super-pixels that are important in making a prediction of a class. For Saliency Map, the color-bar shows which pixels are important in making a prediction of a class. For SHAP values, the red pixels contribute more in making a prediction of a class and the blue pixels contribute less. To explain in an understandable form to a human, we use Table 2 where each bin corresponds to a pixel.

For example, LIME in Figure 5 explains the reason why a customer is flagged as a defaulted customer. The reason is that the amount due on existing mortgage is at least $88,210, the value of the current property is at least $106,450, the number of credit default lines is at most 1, the age of the oldest credit line is between 130 and 166 months and the debt to income ratio is at least 31%. We can interpret this explanation as follows: the customer has defaulted once, he/she still owes a huge amount on his/her mortgage, almost a third of his/her salary/income pays debt and he/she has one credit line

that is older than 11 years. Saliency Map in Figure 5 explains that the oldest credit line for the same customer is between 130 and 166 months, the amount due on existing mortgage is less than $\$42,000$ and the value of the current property is less than $\$72,000$. SHAP values in Figure 5 explain the following: the amount due on existing mortgage is at least $\$88,210$ for the customer and the customer is doing an office work, the value of the existing property is at least $\$106,450$ and the oldest credit line is between 279 and 1168 months. Grad-CAM in Figure 5 explains that the customer requested a loan of atleast $\$13,000$ and the amount due on existing property is atleast $\$88,210$.

It is worth to mention that for Australian credit dataset all feature names and values were anonymized to protect confidentiality of the data. Hence, we can't really explain Figure 3. Next, we explain Figure 4 where the correct class is a defaulted customer. Starting with LIME, the defaulted customer is 34 years or older, owns a real estate, his/her account is in arrears and his/her account balance is less than 200 Deutsche Marks. The same defaulted customer is explained by the Saliency Map as 34 years or older and unemployed. For SHAP values, the customer is 34 years and older, account in arrears, owns a real estate and the account balance is less than 200 Deutsche Marks. For Grad-CAM in Figure 4, the customer has a balance that is less than 200 Deutsche Marks and she/he has been working for atleast 4 years but less than 7 years.

This is an illustration of how one would explain the predictions using the corresponding bins in Table 2. Please note that we did padding on the images to compensate for empty bins, if an explanation highlights a pixel that falls on a padding, we ignore the importance of that pixel because it carries no meaning.

C. SANITY CHECK FOR EXPLANATIONS

Once an explanation is determined for a prediction, the next thing to do is to assess its validity. To do this, we followed a process suggested in [32]. Firstly, a model is trained on the original labels and an explanation is provided to explain a prediction of image $\mathbf{x}$. Secondly, another model having the exact architecture as the previous model is trained on a copy of the original data but the labels are randomly permuted. The reason for randomizing the labels is to break the relationship that exists between the input features and the labels [32]. An explanation is then provided to explain a prediction of a randomly permuted label for image $\mathbf{x}$. Thirdly, the explanations on both scenarios for image $\mathbf{x}$ are compared. If the explanation for image $\mathbf{x}$ did not change after randomly permuting the labels, then the explanation on the original labels did not explain anything about the relationship between the inputs and the prediction. The visual comparison of the explanations could be misleading, to ascertain that the explanations are the same/differ, a quantitative measure such as the *Spearman rank correlation* is used. The Spearman rank correlation is calculated as

$$\rho = 1 - \frac{6 \times \sum_i^l m_i^2}{l(l^2 - 1)} \tag{31}$$

where l is a number of categories that need to be ranked and $m = Rank(\mathbf{x})_o - Rank(\mathbf{x})_p$ is the difference between the rankings of the explanations on the original and the permuted labels, respectively.

TABLE 14. Spearman rank correlations to quantitatively perform sanity check for Figure 3, Figure 4 and Figure 5.

Explanation	Dataset		
	Australian	German	HMEQ
Saliency Map	$\rho = 0.85$	$\rho = 0.21$	$\rho = 0.90$
LIME	$\rho = -0.04$	$\rho = 0.69$	$\rho = 0.06$
SHAP values	$\rho = -0.25$	$\rho = 0.11$	$\rho = -0.60$
Grad-CAM	$\rho = -0.35$	$\rho = 0.13$	$\rho = -0.50$

Table 14 shows correlations for each explanation technique for Figure 3, Figure 4 and Figure 5 on each dataset. For Australian data, LIME shows a weaker correlation. For German data, SHAP values and Grad-CAM show a weaker correlation. For HMEQ, LIME shows a weaker correlation. According to [32], weaker correlations mean that the explanations do pass the sanity check.

D. QUANTITATIVE COMPARISON OF EXPLANATION METHODS

This section looks at comparing all four explanation methods quantitatively. Suppose $\mathcal{L}_R$ is an xgboost model trained on raw dataset and $\mathcal{L}_E$ is another xgboost model trained on explanations of predictions from the model. The performance of $\mathcal{L}_R$ on raw test set is denoted as P_R and the performance on explanations test set is denoted as P_E. We hypothesise that for good explanations, we expect $P_E > P_R$. Note that the performance metric we are using is the accuracy which is a proportion of the correctly classified records. We tested our hypothesis in all three datasets and we found that for all four explanations $P_{E_i} > P_R, \forall i \in \{\text{Grad-CAM, Saliency Map, LIME, SHAP values}\}$. Thereafter, we compared performances of the explanations methods used in this study. The best performing explanation method will have $P_{E_i} > P_{E_j}, \forall j$ where $j \neq i$. To conduct the comparisons, firstly we split each raw dataset into 70% training and 30% test sets. To avoid confusion, for each dataset we only performed a single data split, where we randomly shuffled the records and performed stratified sampling. Secondly, we predict the class of each record for each of the datasets (both training and test sets) using our CNN model. Thirdly, we explain each of the predictions using Grad-CAM, Saliency Map, LIME and SHAP values. Fourthly, we train $\mathcal{L}_R$ on 70% raw dataset and assess its performance P_R on 30% raw test set. Thereafter, we train $\mathcal{L}_E$ on explanations of the predictions from the 70% raw dataset and assess the performance P_E on explanations of the predictions from the 30% raw test set. Table 15 shows performances of the explanation methods together with performances on the raw datasets

TABLE 15. Performances on the 30% test sets based on raw dataset and explanations of predictions from the CNN model. Accuracy is denoted as Acc.

	Performance				
Dataset	Raw	Saliency Map	LIME	SHAP values	Grad-CAM
German	Acc = 0.70	Acc = 0.76	Acc = 0.74	Acc = 0.99	Acc = 0.91
	AUC = 0.63	AUC= 0.65	AUC = 0.64	AUC = 0.98	AUC = 0.87
	Brier Score = 0.31	Brier Score = 0.25	Brier Score = 0.26	Brier Score= 0.02	Brier Score = 0.09
	H-measure = 0.08	H-measure= 0.13	H-measure = 0.12	H-measure = 0.93	H-measure = 0.63
Australian	Acc=0.83	Acc = 0.90	Acc = 0.91	Acc = 1.00	Acc = 0.93
	AUC = 0.82	AUC = 0.85	AUC = 0.87	AUC =1.00	AUC = 0.93
	Brier Score = 0.17	Brier Score = 0.15	Brier Score =0.13	Brier Score =0.00	Brier Score = 0.07
	H-measure = 0.48	H-measure = 0.54	H-measure =0.61	H-measure = 1.00	H-measure = 0.78
HMEQ	Acc = 0.81	Acc = 0.84	Acc = 0.84	Acc = 1.00	Acc = 0.94
	AUC = 0.76	AUC = 0.63	AUC = 0.64	AUC = 1.00	AUC = 0.93
	Brier Score = 0.13	Brier Score = 0.15	Brier Score = 0.16	Brier Score = 0.00	Brier Score = 0.05
	H-measure = 0.40	H-measure= 0.20	H-measure = 0.21	H-measure = 1.00	H-measure=0.80

when using xgboost model. Note that the performances were based on 30% test set. In all three datasets, the SHAP values improve accuracy to 99% and 100% and these are state-of-the-art results for all three datasets. This shows that the SHAP values may be used for explanations as well as for feature selection for improving model performances.

VII. CONCLUSION AND FUTURE WORK

In this paper, tabular datasets were converted into images using bins that were used to calculate weights of evidence. Each pixel of a feature image corresponds to a feature bin. The purpose of using weights of evidence was to create meaningful bins that are monotonic to the response variable. For instance, in credit scoring, a feature such as `age` should decrease monotonically with a default risk response variable and this means that the younger applicants are more riskier than the older applicants in terms of defaulting on their loans. Hence, the risk of defaulting decreases monotonically with increasing age. The other purpose for using bins that were used to calculate weights of evidence in this current study is that both continuous and categorical features were accommodated. The images were then used to train 2D convolutional neural networks. The performances of the trained convolutional neural networks were compared with literature results. We found that the trained convolutional neural networks performed better when compared to the literature results. However, the aim of our study was not only on model performance but rather on explaining predictions. The predictions of the models were then explained using Grad-CAM, LIME, SHAP values and Saliency Map. Each of these explanation methods highlight important regions/pixels in an image that correspond to the output/prediction class. To explain in an understandable form, all important pixels were linked to their respective bins that were used to calculate weights of evidence. The explanation techniques were validated using sanity check methodology. Furthermore, performances of explanation methods were compared quantitatively by using an xgboost model. The explanation method which performed better compared to other methods in all three datasets was SHAP values. The purpose of the xgboost model was to evaluate quantitatively the performances of the explanation methods. However, the future research should focus on validating and evaluating performances of the explanations by using domain experts in the field of credit scoring such as the credit risk analysts or managers. Also, future work should focus on comparing the performances of the methods that perform tabular data conversion into images. Further, for future work the proposed framework can be extended to other deep learning architectures such as ResNet, Deep Graph Neural Networks and others.

REFERENCES

[1] D. C. Sowers and D. Durand, "Risk elements in consumer instalment financing," *J. Marketing*, vol. 6, no. 4, p. 407, Apr. 1942.

[2] X. Dastile, T. Celik, and M. Potsane, "Statistical and machine learning models in credit scoring: A systematic literature survey," *Appl. Soft Comput.*, vol. 91, Jun. 2020, Art. no. 106263.

[3] X. Liu, X. Wang, and S. Matwin, "Interpretable deep convolutional neural networks via meta-learning," *CoRR*, vol. abs/1802.00560, pp. 1–5, Jul. 2018.

[4] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," 2017, *arXiv:1702.08608*. [Online]. Available: http://arxiv.org/abs/1702.08608

[5] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, "Explaining explanations: An overview of interpretability of machine learning," in *Proc. IEEE 5th Int. Conf. Data Sci. Adv. Analytics (DSAA)*, Oct. 2018, pp. 80–89.

[6] D. Rothman, *Hand-on Explainable AI (XAI) with Python*. Birmingham, U.K.: Packt, 2020.

[7] *Basel II: International Convergence of Capital Measurement and Capital Standards: A Revised Framework—Comprehensive Version, Bank for International Settlements*, Basel Committee on Banking Supervision, Basel, Switzerland, 2006.

[8] GDPR. (2018). *Reform of Data Protection Rules*. Accessed: Sep. 17, 2020. [Online]. Available: https://www.legislation.gov.uk/ukpga/2018/12/part/2/chapter/2/enacted

[9] W. Samek, G. Montavon, A. Vedaldi, L. K. Hansen, and K.-R. Mülle, Eds., *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Springer, 2019, doi: 10.1007/978-3-030-28954-6.

[10] L. Breiman, "Statistical modeling: The two cultures," *Stat. Sci.*, vol. 16, no. 3, pp. 199–215, 2001.

[11] W. Samek and K.-R. Müller, "Towards explainable artificial intelligence," in *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*. Springer, 2019, pp. 5–22, doi: 10.1007/978-3-030-28954-6_1.

[12] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Mach. Intell.*, vol. 1, no. 5, pp. 206–215, May 2019.

[13] S. Chari, D. M. Gruen, O. Seneviratne, and D. L. McGuinness, "Directions for explainable knowledge-enabled systems," 2020, *arXiv:2003.07523*. [Online]. Available: https://arxiv.org/abs/2003.07523

[14] B. Zhu, W. Yang, H. Wang, and Y. Yuan, "A hybrid deep learning model for consumer credit scoring," in *Proc. Int. Conf. Artif. Intell. Big Data (ICAIBD)*, May 2018, pp. 205–208.
[15] J. M. Tomczak, "Classification restricted Boltzmann machine for comprehensible credit scoring model," *Expert Syst. Appl.*, vol. 42, no. 4, pp. 1789–1796, Mar. 2015.
[16] Y. Li, X. Lin, X. Wang, F. Shen, and Z. Gong, "Credit risk assessment algorithm using deep neural networks with clustering and merging," in *Proc. 13th Int. Conf. Comput. Intell. Secur. (CIS)*, Dec. 2017, pp. 173–176.
[17] V.-E. Neagoe, A.-D. Ciotec, and G.-S. Cucu, "Deep convolutional neural networks versus multilayer perceptron for financial prediction," in *Proc. Int. Conf. Commun. (COMM)*, Jun. 2018, pp. 201–206.
[18] S. Hamori, M. Kawai, T. Kume, Y. Murakami, and C. Watanabe, "Ensemble learning or deep learning? Application to default risk analysis," *J. Risk Financial Manage.*, vol. 11, no. 1, p. 12, Mar. 2018.
[19] S. Ha and H.-N. Nguyen, "Credit scoring with a feature selection approach based deep learning," *MATEC Web Conf.*, vol. 54, p. 05004, Dec. 2016.
[20] J. Sirignano, A. Sadhwani, and K. Giesecke, "Deep learning for mortgage risk," *SSRN Electron. J.*, 2018, doi: 10.2139/ssrn.2799443.
[21] D. Tripathi, D. R. Edla, V. Kuppili, and A. Bablani, "Evolutionary extreme learning machine with novel activation function for credit scoring," *Eng. Appl. Artif. Intell.*, vol. 96, Nov. 2020, Art. no. 103980.
[22] D. R. Edla, D. Tripathi, R. Cheruku, and V. Kuppili, "An efficient multilayer ensemble framework with BPSOGSA-based feature selection for credit scoring data analysis," *Arabian J. Sci. Eng.*, vol. 43, no. 12, pp. 6909–6928, Dec. 2018.
[23] D. Tripathi, D. R. Edla, and R. Cheruku, "Hybrid credit scoring model using neighborhood rough set and multi-layer ensemble classification," *J. Intell. Fuzzy Syst.*, vol. 34, no. 3, pp. 1543–1549, Mar. 2018.
[24] M. J. Ariza-Garzon, J. Arroyo, A. Caparrini, and M. Segovia-Vargas, "Explainability of a machine learning granting scoring model in peer-to-peer lending," *IEEE Access*, vol. 8, pp. 64873–64890, 2020.
[25] A. Sharma, E. Vans, D. Shigemizu, K. A. Boroevich, and T. Tsunoda, "DeepInsight: A methodology to transform a non-image data to an image for convolution neural network architecture," *Sci. Rep.*, vol. 9, no. 1, Dec. 2019, Art. no. 11399.
[26] C.-L. Yang, Z.-X. Chen, and C.-Y. Yang, "Sensor classification using convolutional neural network by encoding multivariate time series as two-dimensional colored images," *Sensors*, vol. 20, no. 1, p. 168, Dec. 2019.
[27] L. Buturović and D. Miljković, "A novel method for classification of tabular data using convolutional neural networks," Cold Spring Harbor Lab., Cold Spring Harbor, NY, USA, May 2020, doi: 10.1101/2020.05.02.074203.
[28] M. S. Singh, V. Pondenkandath, B. Zhou, P. Lukowicz, and M. Liwicki, "Transforming sensor data to the image domain for deep learning—An application to footstep detection," in *Proc. Int. Joint Conf. Neural Netw.*, 2017, pp. 2665–2672.
[29] M. Tamajka, W. Benesova, and M. Kompanek, "Transforming convolutional neural network to an interpretable classifier," in *Proc. Int. Conf. Syst., Signals Image Process. (IWSSIP)*, Jun. 2019, pp. 255–259.
[30] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," in *Proc. 2nd Int. Conf. Learn. Represent.*, 2014, pp. 1–8.
[31] X. Liu, X. Wang, and S. Matwin, "Interpretable deep convolutional neural networks via meta-learning," in *Proc. Int. Joint Conf. Neural Netw.*, Jul. 2018, pp. 1–9.
[32] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim, "Sanity checks for saliency maps," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.* Red Hook, NY, USA: Curran Associates, 2018, pp. 9525–9536.
[33] N. Siddiqi, *Credit Risk Scorecards: Developing And Implementing Intelligent Credit Scoring*. Chennai, India: SAS, 2005.
[34] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, Dec. 1989.
[35] F. Chollet, *Deep Learning With Python*, 1st ed. Greenwich, CT, USA: Manning, 2017.
[36] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 618–626.
[37] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should i trust you?' Explaining the predictions of any classifier," *CoRR*, vol. abs/1602.04938, pp. 1135–1144, Aug. 2016.
[38] S. Lundberg and S. Lee, "A unified approach to interpreting model predictions," *CoRR*, vol. abs/1705.07874, pp. 1–10, May 2017.
[39] D. Dua and C. Graff. (2017). *UCI Machine Learning Repository*. [Online]. Available: http://archive.ics.uci.edu/ml
[40] Kaggle. (2017). *Home Loan Equity Data*. Accessed: Jun. 3, 2020. [Online]. Available: https://www.kaggle.com/ajay1735/hmeq-data
[41] G. W. Brier, "Verification of forecasts expressed in terms of probability," *Monthly Weather Rev.*, vol. 78, no. 1, pp. 1–3, Jan. 1950.
[42] D. J. Hand, "Measuring classifier performance: A coherent alternative to the area under the ROC curve," *Mach. Learn.*, vol. 77, no. 1, pp. 103–123, Oct. 2009.
[43] D. West, "Neural network credit scoring models," *Comput. Oper. Res.*, vol. 27, nos. 11–12, pp. 1131–1152, Sep. 2000.
[44] M. Ala'raj and M. F. Abbod, "Classifiers consensus system approach for credit scoring," *Knowl.-Based Syst.*, vol. 104, pp. 89–105, Jul. 2016.
[45] Y. Xia, C. Liu, Y. Li, and N. Liu, "A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring," *Expert Syst. Appl.*, vol. 78, pp. 225–241, Jul. 2017.
[46] L. Munkhdalai, O.-E. Namsrai, and K. Ryu, "Credit scoring with deep learning," in *Proc. 4th Int. Conf. Inf. Syst. Appl.*, 2018, pp. 1–6.



XOLANI DASTILE received the bachelor's degree (Hons.) in mathematics and the bachelor's (Hons.) and M.Sc. degrees in mathematical statistics from Rhodes University, South Africa. He is currently pursuing the Ph.D. degree in computer science in the field of deep learning with the University of the Witwatersrand, Johannesburg, South Africa. He is also a Senior Data Scientist with Fuel Management Company in South Africa. He previously worked in major banks in South Africa under credit risk departments as a Quantitative Analyst as well as a Data Scientist.



TURGAY CELIK (Member, IEEE) received the Ph.D. degree from The University of Warwick, Coventry, U.K., in 2011. He is currently a Professor of Digital Transformation and the Director of the Wits Institute of Data Science, University of the Witwatersrand, Johannesburg, South Africa. His research interests include signal and image processing, machine intelligence, robotics, data science, and remote sensing. He is also an Associate Editor of the *IET Electronics Letters* (ELL), the IEEE Geoscience and Remote Sensing Letters (GRSL), and *Signal, Image and Video Processing* (SIVP, Springer).

2.3 Journal Article III

X. Dastile, T. Celik and H. Vandierendonck, Model-agnostic Counterfactual Explanations in Credit Scoring, *IEEE Access*, vol. 10, pp. 69543-69554, 2022.

Number of citations to date: 1
Publisher: Institute of Electrical and Electronics Engineers
Cite Score: 3.367
Impact Factor: 4.8
Abstracting and Indexing: Scopus (Elsevier)
Web of Science (Clarivate Analytics)
ProQuest
IET (The Institution of Engineering and Technology)
NLM (US National Library of Medicine)
CrossRef

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 7 February 2022, accepted 26 April 2022, date of publication 25 May 2022, date of current version 6 July 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3177783

Model-Agnostic Counterfactual Explanations in Credit Scoring

XOLANI DASTILE[1], TURGAY CELIK[2,3,4], AND HANS VANDIERENDONCK[5], (Senior Member, IEEE)

[1]School of Computer Science and Applied Mathematics, University of the Witwatersrand, Johannesburg 2000, South Africa
[2]School of Electrical and Information Engineering, University of the Witwatersrand, Johannesburg 2000, South Africa
[3]Wits Institute of Data Science, University of the Witwatersrand, Johannesburg 2000, South Africa
[4]Faculty of Engineering and Science, University of Agder, 4630 Kristiansand, Norway
[5]School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, Belfast BT9 5BN, U.K.

Corresponding author: Turgay Celik (celikturgay@gmail.com)

ABSTRACT The past decade has shown a surge in the use and application of machine learning and deep learning models across various domains. One such domain is credit scoring, where applicants are scored to assess their creditworthiness for loan applications. It is essential to ensure that no biases or discriminations are incurred during the scoring process. Most machine learning and deep learning models are prone to unintended bias and discrimination in the datasets. Therefore, it is imperative to explain each prediction from the models during the scoring process to avoid the element of model bias and discrimination. Our study proposes a novel optimization formulation that generates sparse counterfactual explanations via a custom genetic algorithm to explain the black-box model's predictions. We evaluated the efficacy of the proposed method on publicly available credit scoring datasets by comparing the counterfactual explanations generated by the proposed method with explanations from credit scoring experts. The proposed counterfactual explanation method does not only explain rejected loan applications but also can be used to explain approved loan applications.

INDEX TERMS Credit scoring, machine learning, counterfactual explanation, explainable AI, genetic algorithm.

I. INTRODUCTION

The pervasiveness and ubiquitous nature of machine learning and deep learning models brings about positive change in society with the risk of unintended consequences such as algorithmic bias. The algorithms are not intrinsically biased but inherit the bias from human activities captured in the data during the training process. Thus, the models need to be transparent at all costs. The Basel Accord [1] requires explanations for denied loan applications to ensure that transparency is maintained in automated decisions in the financial sector. Emerging regulations such as the European Union General Data Protection Regulation (GDPR) [2] stipulate that, for automated decisions, a ''right to explanation'' needs to be maintained. Explaining predictions is crucial for high-stake decisions, such as the presence or absence of a disease in the healthcare sector, the rejection or acceptance of a loan application in the finance sector, and the denial or approval of parole in the criminal justice sector. Hence, our study focuses on using a counterfactual explanation technique, which is an instance-based explanation, for explaining predictions from black-box models. Wachter et al. [3] highlighted three key benefits of using the counterfactual explanation, (1) to inform and assist applicants in understanding why a certain decision was made, (2) to give grounds to contest unfair decisions, and (3) to understand what could be changed to attain a desired outcome in the future. A typical example of a counterfactual explanation is a loan application [4], [5]: *"Imagine you filed a credit application at a bank. Unfortunately, the bank rejects your application. Now, you would like to know why. In particular, you would like to know what would have to be different so that your application would have been accepted. A possible explanation might be that you would have been accepted if you would earn 500$ more per month and if you would not have a second credit card."*

Therefore, it is imperative to explain each prediction from the models during the scoring process to avoid the element of model bias and discrimination. We propose a novel optimization formulation that generates sparse counterfactual explanations via a custom genetic algorithm to explain the

The associate editor coordinating the review of this manuscript and approving it for publication was Long Wang.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

black-box model's predictions. Our contributions are as follows: 1) a novel formulation of the optimization problem that leads to a single sparse counterfactual explanation using a custom genetic algorithm; 2) a computationally efficient generation of counterfactuals achieved by the normalization of continuous features and selection of predictive features; and 3) validation of automatically generated counterfactuals with the credit scoring experts. Without loss of generality, we used the terms ''counterfactual'' and ''counterfactual explanation'' interchangeably. In addition, the words ''sample'', ''instance'' and ''record'' are used interchangeably.

The remainder of this paper is organized as follows. Section II discusses background and related work on counterfactual explanations. Section III introduces our proposed method for generating counterfactual explanations. The experiment setup is described in Section IV and the results are given in Section V. Finally in Section VI, we summarize the paper and discuss our future work.

II. BACKGROUND AND RELATED WORK

The literature has done a great amount of work on eXplainable Artificial Intelligence (XAI), an emerging artificial intelligence branch. The main purpose of XAI is to make deep learning and machine learning models interpretable. A black-box model can be explained by either using a *post-hoc*, an *ante-hoc* or an *instance-based* explanation. A post-hoc explanation uses another model such as a linear regression or a decision tree to explain the behaviour of a black-box model (e.g. Local Interpretable Model-Agnostic Explanation (LIME) [6]). On the other hand, an ante-hoc explanation is an inherently interpretable model (e.g. Bayesian Rule List [7]), and an instance-based explanation uses an instance to explain the behaviour of a black-box model (e.g. Visual Counterfactual Explanations (ViCE) [8]). Rudin [9] posited that inherently interpretable models (i.e. ante-hoc explanations) should be used for high-stake decisions in lieu of explainable machine learning (referring to post-hoc explanations). The study argued that explainable machine learning generates explanations that are not faithful to what the black-box model computes. Hence, to avoid this shortcoming, this study proposes the use of counterfactuals to explain black-model predictions.

Although counterfactuals seem to produce intuitive explanation systems, some problems remain. Most counterfactual explanation methods generate more than one counterfactual explanation, and this problem is referred to as *Rashomon effect* [5]. The generated explanations might have contradicting ''paths'' on how a certain output was reached. It becomes more difficult and unclear for the user or applicant if there is more than one option of explanations to select from. The other issue with counterfactual explanations is the time it takes to generate the counterfactuals [10]. The time metric can be measured as an average time taken over the generation of a counterfactual for a group of records or the generation of multiple counterfactuals for a single input record [10].

Despite these problems, the research in recent years has shown an increase in the number of studies that focused on explaining machine learning predictions using counterfactual explanations. Grath et al. [11] proposed two weighted approaches to produce counterfactuals for credit scoring data, where one approach derives weights from feature importance, and the other depends on nearest neighbours. Empirical results showed that the weights that are produced from feature importance result in more compact counterfactuals. Furthermore, the study produced positive counterfactuals for accepted loan applications to assist individuals when they make future financial decisions.

In most cases, counterfactuals are generated by using a metaheuristic approach (i.e. an optimization approach that is not problem/task dependent). Guidotti et al. [12] proposed a method that learns a local and interpretable classifier on a synthetic neighbourhood of a record of interest (i.e. an instance that its prediction needs to be explained). The synthetic neighbourhood is generated by a genetic algorithm. The proposed method produces a local explanation that consists of logical rules explaining a decision of the instance of interest and a set of counterfactuals that suggest changes in the instance of interest to produce a desirable outcome. The results showed that the proposed method outperforms previous methods with regards to the quality of the produced explanations and the faithfulness to the black-box model. Sharma et al. [13] proposed a unified and model agnostic approach to address non-transparency (among other issues) of black-box models by using counterfactuals that are generated via a genetic algorithm. The proposed approach achieved robustness, transparency, interpretability, and fairness of black-box models. Further, the study intends to improve the speed of genetic algorithms in their future work. Sharma et al. [13] is the closest study to our approach.

Once the counterfactuals are generated, the next thing to look at is the feasibility of the generated counterfactuals. A change in a few features makes counterfactuals to be feasible. Van Looveren et al. [14] proposed a framework for generating sparse and in-distribution counterfactuals. A sparse counterfactual refers to a counterfactual that requires minimum feature changes to belong to the desired class.

Poyiadzi et al. [15] argued that the current methods that generate counterfactuals for explanation are not considering the feasibility of the generated counterfactuals in the real world. This is attributable to counterfactuals that do not represent the underlying data distribution. The study proposed a method that generates feasible and actionable counterfactual explanations based on the shortest path distance determined by density-weighted metrics. The proposed approach generates counterfactuals that are logical and consistent with the underlying data distribution, making counterfactuals feasible and actionable. Mothilal et al. [16] posited that counterfactuals should be feasible and diversified. The study proposed a framework for generating and assessing the diversity of counterfactuals based on a determinant of a kernel matrix.

The proposed framework generates diverse counterfactuals as opposed to previous methods.

Efficiency and speed are key factors when generating counterfactuals. It is better to use fewer computer resources to speed up the computation time. Resource-intensive methods tend to result in high computation time. Artelt and Hammer [17] investigated how to efficiently compute counterfactuals for prototype-based classifiers. The study discovered that in most cases, either a set of linear or convex quadratic programs that generate counterfactuals can be solved efficiently. Van Looveren and Klaise [18] proposed the use of class prototypes to speed up the generation of counterfactuals. The class prototypes are either attained by employing an encoder or class-specific *k-d* trees. Efficiency is synonymous with the generation time of counterfactuals.

Not only speed and efficiency are essential when generating counterfactuals, but the safety of the black-box models. Counterfactuals guarantee the safety of the black-box models. Sokol and Flach [19] showed that when making AI systems explainable, there is a chance of compromising the safety and security of the system and the possibility of data leakage. This poses a challenge to Explainable AI systems, and the study suggests that the security of AI systems will not be compromised when counterfactuals are used in lieu of other explainable AI techniques.

In general, the credit scoring literature is not extensive when it comes to counterfactual explanations. Hence, this study aims to expand the use of counterfactual explanations in credit scoring to address the issue of multiple counterfactuals that are generated to explain a single instance. Further, this study aims to suggest alternative ways to deal with the efficiency and sparseness of counterfactuals.

III. METHODOLOGY

This section outlines the methodology that is undertaken in this study. The key design decisions were to collect the datasets, select predictive features, calculate correlations between the target variable and the predictors, normalize each continuous feature of the datasets, formulate an optimization problem that will help to generate the counterfactuals, measure the time it takes to generate the counterfactuals, and to compare the generated counterfactuals with experts' opinions. Each of the above steps helped us to obtain counterfactuals that are sparse, generated fast, and robustly tested.

Figure 1 shows the flowchart of our proposed methodology. Firstly, we focus on feature selection, where we select predictive features using a random forest model. Secondly, we calculate Spearman's correlation coefficient between the target variable and the features. The aim of calculating the correlations is to create sparse counterfactuals. This means that we will only focus on features that are better correlated with the target variable when generating the counterfactuals. The feature selection together with sparsity ties back to the feasibility of the counterfactuals, which requires a minimal number of features to be changed. Thereafter, the dataset is split into categorical and continuous features, and the continuous features are normalized. The purpose of normalizing continuous features is to ensure that the features have the same scale. This allows counterfactuals to be generated much quicker because there is no huge varying degree of values within each feature. We proceed by merging the categorical and continuous features. The dataset is then split using the K-fold cross validation and a classifier is trained and its performance is assessed. The purpose of the K-fold cross validation is to ensure that model robustness is maintained during training. The final step is to generate a counterfactual that will explain the predicted outcome for a data point that we are interested in (i.e. a defaulted loan).

A. DATA PREPROCESSING

We selected predictive features via a random forest. The random forest is an ensemble of decision trees. At each node of the decision tree, a split is determined by the measure of impurity of each feature, either by using the *entropy* or the *gini index*. The importance or predictive power of each feature is derived from the impurity calculation, which is shown by either the entropy

$$I_E = -\sum_{j=0}^{1} p_j \log_2 p_j \tag{1}$$

or the gini index

$$I_G = 1 - \sum_{j=0}^{1} p_j \tag{2}$$

where p_j represents proportion of samples in each class. The more "pure" a feature is, the more important it is. The importance of each feature is obtained from how "pure" a feature is. The more the feature reduces impurity, the more important the feature is.The final importance of the feature is the average impurity decrease across all decision trees. Thus, all selected features will have a maximum average impurity decrease across all decision trees.

B. COUNTERFACTUAL SPARSITY

To ensure that the counterfactuals that are generated are sparse, we used Spearman's rank correlation coefficient. Spearman's rank correlation coefficient is based on the rankings of a feature as opposed to using raw feature values. The benefit of using rankings is that both continuous and categorical features can be used to calculate the correlation. Hence, sparsity is determined by the correlation between the target variable (which is categorical in nature) and the predictors (which are either categorical or continuous). Spearman's rank correlation coefficient is given as

$$\rho = 1 - \frac{6 \times \sum z_i^2}{n(n^2 - 1)} \tag{3}$$

where z_i is the difference between two ranks of each record and n is the number of records. The aim of calculating the correlation is to focus only on features that are correlated with the target variable when the counterfactuals are generated.

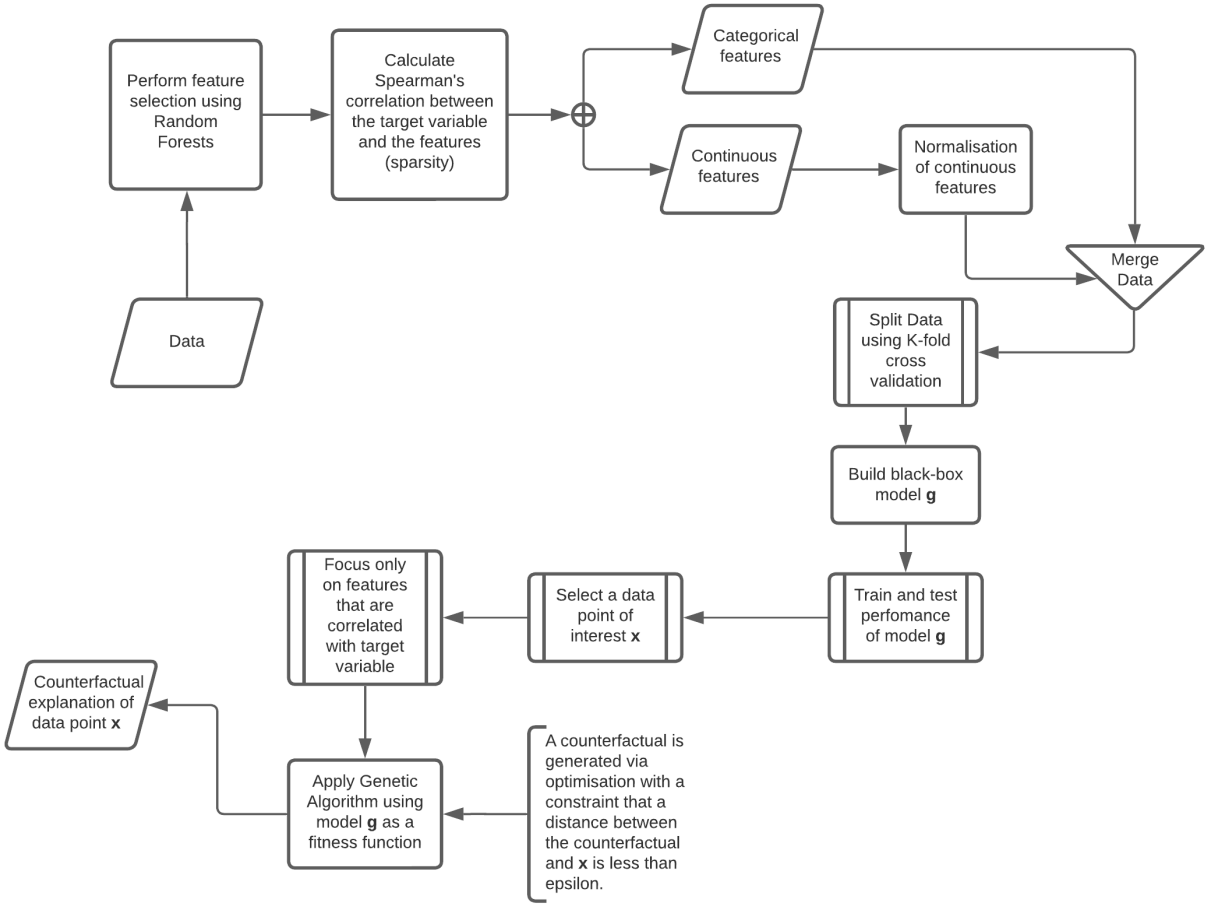

FIGURE 1. The flowchart of the proposed methodology.

The statistical significance of a Spearman's rank correlation is tested by using a hypothesis testing. The test will measure the association between the target variable and the predictors. The null hypothesis and the alternative hypothesis are stated as

- H_0 = There is no monotonic association between variables.
- H_a = There is a monotonic association between variables.

The level of significance for the hypothesis test is $\alpha = 5\%$. The null hypothesis H_0 is rejected when the p-value $< \alpha$. If the null hypothesis is rejected, then the conclusion will be that there is statistical evidence that there is association between variables.

C. NORMALIZATION OF CONTINUOUS FEATURES

Let $\mathbf{x}_k \in \mathbb{R}^d$ denote a feature vector for record k, where d represents the number of features in a dataset. The labeled dataset is represented by $\mathbf{X} = \{(\mathbf{x}_k, y_k)\}_{k=1}^{n}$, where $y_k \in \{0, 1\}$ represents the target/response flag and $\mathbf{x}_k = \{f^i\}_{i=1}^{d}$ is the feature vector, and n denotes the number of records in the dataset. Let $\mathbf{f}^i$ represent each feature in $\mathbf{X}$, $\forall i \in \{1, 2, \cdots, d\}$. Continuous feature values were converted by using a normalization technique

$$f^i_{norm} = \frac{f^i - \min(\mathbf{f}^i)}{\max(\mathbf{f}^i) - \min(\mathbf{f}^i)}, \tag{4}$$

such that

$$f^i_{norm} \in [0, 1], \tag{5}$$

where $\min(\mathbf{f}^i)$ and $\max(\mathbf{f}^i)$ are the minimum and the maximum values for feature i, respectively.

D. GENETIC ALGORITHM

The genetic algorithm is a heuristic search approach that is motivated by natural evolution [20]. The genetic algorithm searches for optimal values that can either minimize or maximize a certain function. Hence, genetic algorithms are used for optimization problems. The genetic algorithm involves a selection of individuals (i.e. parents) (based on some defined fitness function) from an initial population for reproduction. Individuals that are unfit for reproduction are omitted. The parents produce an offspring that inherits their (i.e. parents) characteristics. The offspring then gets included in the next generation of the population. The process repeats itself until

it converges into a solution. The genetic algorithm process involves five phases i.e. the *initial population*, the *fitness function*, the *selection*, the *cross-over*, and the *mutation*. In the context of credit scoring, a population individual is a feature vector and this feature vector is regarded as a counterfactual. To select the best counterfactuals, a fitness function is used. The fitness function in this study is the black-box model that is required to be explained by the counterfactual. The fitness function output is probabilistic and has values in the range [0, 1]. The quality of each counterfactual is determined by the output of the fitness function and the distance between the counterfactual and the feature vector of interest (i.e. a feature vector with a default response flag) should be less than some *epsilon* value. Selected counterfactuals go to a mating pool and these counterfactuals are known as parents. Every two parents will produce two offspring (i.e. two counterfactuals). The mating process is a cross-over phase. The cross-over is based on a cross-over rate which is defined as the probability of two parents crossing over at a single point [20]. Mating high-quality parents will generate better-quality offspring with similar traits as the parents. This removes bad population individuals from generating more bad individuals. Note that, the offspring will have similar drawbacks as their parents. To overcome the drawbacks, some changes will need to be made to the offspring to generate new offspring. The changes are known as the mutation phase. The mutation is responsible for randomly changing values in the counterfactual based on a mutation rate which is defined as the probability of determining the number of counterfactuals that should be mutated in a single population [21]. The main purpose of mutation is to preserve diversity in a population, and this prevents early convergence to a solution.

E. FORMULATION OF THE OPTIMIZATION PROBLEM

This section defines our formulation of the optimization problem which is solved via a custom genetic algorithm. Let $\mathbf{c}^* \in \mathbb{R}^d$ denote a counterfactual for record $\mathbf{x} \in \mathbb{R}^d$ and g denote a black-box model. Let g have a probabilistic cartesian product output. Then g is given as

$$g : \mathbb{R}^d \rightarrow A \times B, \tag{6}$$

where

$$A \times B = \{(a, b) | a \in [0, 1], b \in [0, 1], a + b = 1\} \tag{7}$$

and a represents a probability of belonging to class 0 and b is a probability of belonging to class 1. The aim is to find $\mathbf{c}^* \in \mathbb{R}^d$ such that

$$\mathbf{c}^* = \underset{\mathbf{c} \in \mathbb{R}^d}{\arg\min}\ g_{b \in [0,1]}(\mathbf{c}) \tag{8}$$

subject to

$$\text{dist}(\mathbf{c}^*, \mathbf{x}) < \epsilon \tag{9}$$

and

$$c^{*,i} \in [\min(\mathbf{f}^i), \max(\mathbf{f}^i)], \quad \forall i \in \{1, 2, \cdots, d\}. \tag{10}$$

The main idea behind the above optimization problem is to find optimal $\mathbf{c}^* \in \mathbf{C}$ (i.e. $\mathbf{C}$ is the set of counterfactuals) that result in a non-default class, ensuring that the generated counterfactual $\mathbf{c}^*$ is as close as possible to the instance of interest $\mathbf{x}$ and that $\mathbf{c}^*$ comes from the same data distribution as $\mathbf{x}$. The output of each class (i.e. default or non-default) is

$$\text{class output} = \begin{cases} 0, & \text{if } (a, b) \in [0.5, 1] \times [0, 0.5) \\ 1, & \text{if } (a, b) \in [0, 0.5) \times [0.5, 1]. \end{cases} \tag{11}$$

In Eq. (8), we ensure that $b \in [0, 1]$ is minimized so that $a \in [0.5, 1]$ makes the counterfactual to belong to a non-default class. Please note that class output $= 0$ and class output $= 1$, means that $\mathbf{x}$ belongs to a non-default class and default class, respectively. According to Wachter et al. [3], closeness of $\mathbf{c}^*$ to $\mathbf{x}$ should be measured by the L_1 norm distance divided by the mean absolute deviation (MAD). Wachter et al. [3] showed that the L_1 norm distance divided by MAD is better than the L_1 or L_2 norm distances. The first part of Equation (12) is for continuous features and the latter part is for categorical features. Since we are treating categorical values as discrete, hence we used the mean deviation which is suitable for discrete values. The choice of distance in our study is

$$\text{dist}(\mathbf{c}^*, \mathbf{x}) = \sum_{i=1}^{d^*} \frac{|c^{*,i} - f^i|}{MAD^i} + \sum_{i=d^*+1}^{d} \frac{|c^{*,i} - f^i|}{MD^i}, \tag{12}$$

where MAD^i and MD^i denote a mean absolute deviation and a mean deviation for feature i, respectively, and d^* denotes the number of continuous features. The mean absolute deviation for feature i is

$$MAD^i = \frac{1}{n} \sum_{j=1}^{n} |f_j^i - \bar{\mathbf{f}}^{(i)}|, \tag{13}$$

where n denotes the number of records and $\bar{\mathbf{f}}^{(i)}$ denotes the mean or average of feature i. The mean deviation for feature i is

$$MD^i = \frac{1}{m} \sum_{l=1}^{m} o_l |f_l^i - \tilde{\mathbf{f}}^{(i)}|, \tag{14}$$

where m is the number of categories, o_l is the frequency of each category and $\tilde{\mathbf{f}}^{(i)}$ is the median of feature i.

The threshold for the chosen distance is defined by ϵ which is provided by the user. The fitness function in our optimization problem is $g(\mathbf{c}^*)$. To explain predictions of applicants that are approved for loans (i.e. class output $= 0$), the aim is to find optimal values of $\mathbf{c}^*$ such that

$$\mathbf{c}^* = \underset{\mathbf{c} \in \mathbb{R}^d}{\arg\min}\ g_{a \in [0,1]}(\mathbf{c}) \tag{15}$$

subject to the same constraints of Equation (8). In Eq. (15), we ensure that $a \in [0, 1]$ is minimized so that $b \in [0.5, 1]$ makes the counterfactual to belong to a default class.

The major differences between our study and that of Sharma et al. [13] are 1) the formulation of the optimization

problem, 2) the distance calculation for categorical features and 3) the generation of sparse counterfactuals. Sharma et al. [13] used the following fitness function,

$$\text{fitness function} = \frac{1}{\text{dist}(\mathbf{c}^*, \mathbf{x})} \tag{16}$$

where

$$\text{dist}(\mathbf{c}^*, \mathbf{x}) = \frac{n_{con}}{n} \sum_{i=1}^{n_{con}} \frac{|c^{*,i} - f^i|}{MAD^i} + \frac{n_{cat}}{n} \text{SimpleMat}(\mathbf{c}^*{}_{cat}, \mathbf{x}_{cat}) \tag{17}$$

and n_{con} and n_{cat} denote the number of continuous and categorical features, respectively, and $\mathbf{c}^*{}_{cat}$ and $\mathbf{x}_{cat}$ are categorical attributes for the counterfactual $\mathbf{c}^*$ and record $\mathbf{x}$, respectively. The simple matching coefficient is used in Sharma et al. [13] to deal with categorical features and is given as

$$\text{SimpleMat} = \frac{\text{number of matching attributes}}{\text{total number of attributes}}, \tag{18}$$

and has values between 0 and 1.

IV. EXPERIMENTS

A. DATA

Publicly available credit scoring datasets were used in this study, i.e. German [22] and Home Loan Equity (HMEQ) [23]. The German dataset can be accessed on the *UCI* repository, and the HMEQ dataset can be accessed on *Kaggle*. The German credit dataset has 20 features, where 7 of the features are numerical and the other 13 are categorical. The German credit dataset has features such as `status of existing checking account`, `duration in month`, `credit history` and `purpose` to mention a few. The HMEQ dataset has 13 features, where 11 of the features are numerical and the other 2 are categorical. The features are the `amount of loan request`, `amount due on existing mortgage`, `value of the current property`, `years at present job`, `number of credit lines` to mention a few. The `target variable` for each of the above credit scoring datasets is binary, i.e. applicants are classified either as "default" (i.e. bad applicants denoted by a 1) or "non-default" (i.e. good applicants denoted by a 0).

B. DATA SPLIT, MODEL TRAINING AND MODEL PERFORMANCE

For robustness of the classifier g, a K-fold cross validation [24] was used. The K-fold cross validation splits the data into K equal folds $\mathcal{D}_1, \mathcal{D}_2, \cdots, \mathcal{D}_K$, where

$$\bigcup_{k=1}^{K} \mathcal{D}_k = \mathbf{X}. \tag{19}$$

Each fold $\mathcal{D}_k$ is used as a test set and the remaining $\mathcal{D} = \{\mathcal{D}_j\}_{j=1, j\neq k}^{(K-1)}$ folds are used for model training. The overall performance metric for the model is

$$g_p = \frac{1}{K} \sum_{k=1}^{K} g_p^k \tag{20}$$

and g_p represents the average model performance metric, e.g. accuracy or Area Under the Curve (AUC), and g_p^k is a model performance metric in each of the test folds $\mathcal{D}_k$. Note that the remaining folds $\mathcal{D} = \{\mathcal{D}_j\}_{j=1, j\neq k}^{(K-1)}$ were used in turn for training, we did not build K distinct models.

C. EXPERIMENT SETUP

For the genetic algorithm, we ran four experiments on each dataset to select optimal parameters based on the execution times of the optimization problem, the resulting class output and the distance between the record that needs to be explained and the generated counterfactual. Table 1 shows different parameter values that were used for the genetic algorithm. We then measured the times in seconds for the execution of the optimization problem, the resulting class outputs, and the distances were also assessed and are shown in Table 1. For German dataset, we chose $\epsilon = 4$, mutation rate $= 0.01$, cross-over rate $= 0.40$ and population size $= 30$, since they are giving a least amount of execution time, the desired class output (i.e. the non-default output), and the minimal distance. For HMEQ dataset, we chose $\epsilon = 7$, mutation rate $= 0.04$, cross-over rate $= 0.70$ and population size $= 100$, since they are giving the desired class output (i.e. the non-default output) and the least distance. The models that were used in our study were random forest and artificial neural networks (this was to ensure that our approach is model-agnostic). The choice of these models was based on our previous literature review study [25]. The parameters for the random forest model were set as follows, the maximum depth for each decision tree was set to 5, and the number of decision trees was set to 100. For the artificial neural networks, we used 3 hidden layers, the first hidden layer had 50 neurons, and the second hidden layer had 30 neurons and the third hidden layer had 5 neurons, with the output layer having 2 neurons. The parameters for both neural networks and random forest were obtained by using a grid-search approach. To train the models, we used K-fold cross validation, where we set K = 5.

D. EXPERIMENTS FOR CREDIT SCORING EXPERTS

Initially, we contacted about six credit scoring experts and there were only three experts who showed interest in taking part in the experiments of this study. It is challenging to find credit scoring experts since they have busy schedules and are mostly not available. The use of several experts in this study was to ensure that we get different views from industry experts. This will then ensure that the counterfactuals are robustly assessed. All the experts in this study have more than 7 years of working experience in the financial sector, working as quantitative analysts in credit scoring departments. They all have a background in developing credit scorecards from scratch. The experts were given a data dictionary with a

TABLE 1. Selection of parameters for genetic algorithm based on execution times (measured in seconds), the resulting objective function and the distance between the original feature vector and the counterfactual vector on German and HMEQ credit datasets. Legend: time (execution time for optimization problem), func (predicted class output), dist (distance between record that needs to be explained and the counterfactual). For predicted class output, 0 denotes non-default class and 1 denotes default class.

Experiments	German	HMEQ
Experiment 1	$\epsilon = 4$	$\epsilon = 4$
	mutation rate = 0.01	mutation rate = 0.01
	cross-over = 0.40	cross-over = 0.40
	population = 30	population = 30
	time = 386, func = 0, dist=3.75	**time = 60, func = 0, dist=24**
Experiment 2	$\epsilon = 5$	$\epsilon = 5$
	mutation rate = 0.02	mutation rate = 0.02
	cross-over = 0.50	cross-over = 0.50
	population = 40	population = 40
	time = 531, func = 0, dist=3.75	**time = 87, func = 0, dist=15**
Experiment 3	$\epsilon = 6$	$\epsilon = 6$
	mutation rate = 0.03	mutation rate = 0.03
	cross-over = 0.60	cross-over = 0.60
	population = 60	population = 60
	time = 811, func = 0, dist=5	**time = 123, func = 0, dist=195**
Experiment 4	$\epsilon = 7$	$\epsilon = 7$
	mutation rate = 0.04	mutation rate = 0.04
	cross-over = 0.70	cross-over = 0.70
	population = 100	population = 100
	time = 1378, func = 0, dist=3.75	**time = 192, func = 0, dist=6**

detailed description of each feature. The experts were asked to identify features that they deem fit in influencing the prediction of the record of interest (i.e. the record that belongs to a default class) based on their domain expertise. This was to allow credit experts to have a subjective opinion on the features that they thought were key contributors in predicting the class of the record of interest. The credit risk experts then created their counterfactuals and they assessed the prediction of their counterfactuals using an online user interface, to see if the counterfactuals will belong to the desired class (i.e. a non-default class). The experts were also asked to normalize all continuous features before they use the online platform. The online user interface can be found on this link (https://xolani-explanation-research.herokuapp.com/). We created this user interface from scratch and the scoring models that were used in the user interface for predictions are the ones that we are explaining using the counterfactuals in this study.

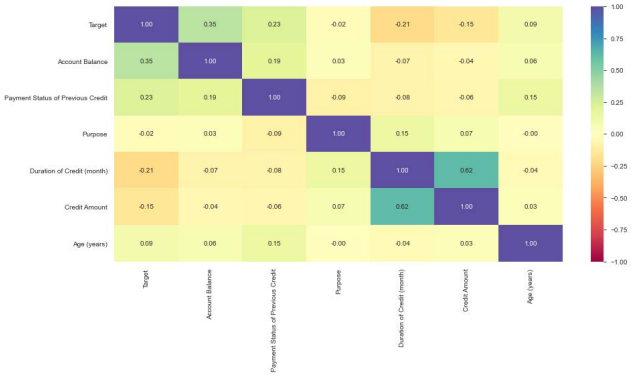

FIGURE 2. Spearman's rank correlation coefficient for the German credit dataset. The selected features are `Account Balance`, `Payment Status of Previous Credit` and `Duration of Credit (month)` based on the correlation coefficients with the `Target` variable. The p-values of the correlation coefficients for all selected features are ≈ 0.

V. RESULTS

This section starts by looking at the features that are correlated with the target variable and thereafter the model performances of the random forest classifier on German credit dataset and the artificial neural networks classifier on HMEQ credit dataset. Lastly, explanations from our approach, Sharma et al. [13] approach and from the experts are examined.

A. COUNTERFACTUAL SPARSITY

We assessed the correlations between the target variable and the predictors. The aim is to focus only on predictors that are better correlated with the target variable when we are generating our counterfactuals. Figure 2 and Figure 3 show correlation matrices for the German and HMEQ credit datasets, respectively. In Figure 2, the predictors that are better correlated with the target variable are `Account Balance` and `Payment Status of Previous Credit`. Please note that by *"better correlation"* we mean that compared to the rest of the predictors, the selected predictors have a *"higher"* correlation with the target variable. In Figure 3, the predictors that are better correlated with the target variable are `DEBTINC`, `DEROG` and `DELINQ`. Thereafter, we tested the statistical significance of the correlations. All the selected features in both datasets that are correlated with the target variable show that the associations are statistically significant since their p-values $\ll 5\%$.

B. MODEL PERFORMANCE

Table 2 shows the classifier performance results that were reported in literature, and we compared those results to the performances of the models that were used in our study.

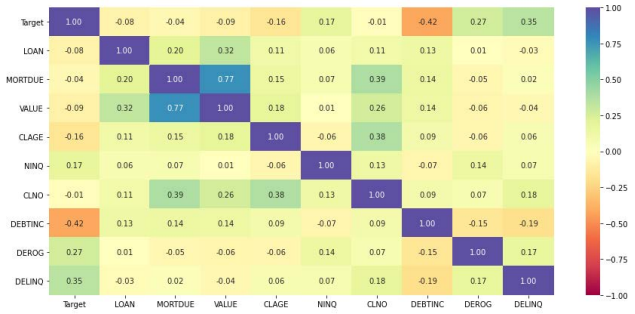

FIGURE 3. Spearman's rank correlation coefficient for the HMEQ credit dataset. The selected features are `DEBTINC`, `DEROG` and `DELINQ` based on the correlation coefficients with the `Target` variable. The p-values of the correlation coefficients for all selected features are ≈ 0.

TABLE 2. Credit scoring model performances that are reported in the literature.

Source	Year	Model	German		HMEQ	
			Accuracy	AUC	Accuracy	AUC
[26]	2000	Logistic Regression	0.76	-	-	-
[27]	2016	Neural Network	0.75	-	-	-
[28]	2017	XGBoost	0.77	-	-	-
[29]	2018	Neural Network	0.78	0.80	-	-
Our Study	2022	Random Forest	0.74	0.76	-	-
Our Study	2022	Neural Network	-	-	0.78	0.79

A detailed comparison of model performances in credit scoring can be found in our previous study [25]. In Table 2, the results do not significantly differ from each other, and this proves the efficacy of our model choice for our study. The main purpose of our study is not to compare model performances but to explain how a model makes its prediction and what actions are required to effect a desired outcome for the loan applicant.

C. EXPLANATIONS

1) PREDICTION EXPLANATIONS FROM OUR APPROACH AND FROM SHARMA et al. [13] APPROACH

Please note that in this study, we also implemented from scratch, the approach that was used in [13]. The explanations are given in Figure 4 and Figure 5, for German and HMEQ credit datasets, respectively. The feature names are on the y-axis, the x-axis represent the change between the original feature value and the counterfactual feature value. Please note that all continuous values in the figures were normalized, however when providing explanations using text, the normalized values were denormalized to make sense out of the explanations. Using our approach on German credit data, the applicant would qualify for the loan if the `Payment Status of Previous Credit` *decreases* by 3 and `Account Balance` *decreases* by 1. Using the approach by Sharma et al. [13] on German credit data, the applicant would qualify for the loan if the `Account Balance` *decreases* by 1, `Credit Amount` *increases* by 15630, `Duration of Credit (month)` *increases* by 43, `Purpose` *increases* by 5 and `Age (years)` *decreases* by 6. Using our approach on HMEQ credit data, the applicant would qualify for the loan if the `DEBTINC` *decreases* by 32, `DELINQ` *increases* by 1 and `DEROG` *increases* by 3. Using the approach by Sharma et al. [13] on HMEQ credit data, the applicant would qualify for the loan if the `LOAN` *increases* by 39072, `MORTDUE` *increases* by 325939, `VALUE` *increases* by 84791, `CLAGE` *increases* by 1051, `NINQ` *increases* by 4, `CLNO` *increases* by 9, `DEBTINC` *decreases* by 24 and `DEROG` *increases* by 10.

TABLE 3. Comparison of our approach with the approach from Sharma et al. [13] using different metrics. Legend: time (execution time for optimization problem measured in seconds), func (predicted class output), dist (distance between the record that needs to be explained and the counterfactual). For predicted class output, 0 denotes non-default class and 1 denotes default class.

	Performance metrics	
	German	HMEQ
Sharma et al. [13]	time=879, func=0, dist= 5	time=722, func=0, dist=16
Our approach	time=386, func=0, dist=3.75	time=192, func=0, dist=6

The difference between our approach and that of Sharma et al. [13] is around the time it takes to generate a counterfactual and the distance between the record of interest and the generated counterfactual. The generated counterfactual must not be far off from the data point of interest, this ensures that the counterfactual is feasible. Our approach uses sparse counterfactuals that result in small distances between the counterfactual and the record of interest. This is illustrated in Table 3.

2) EXPLANATIONS FROM CREDIT SCORING EXPERTS

Credit scoring experts generated their own counterfactual explanations based on their domain knowledge. Please refer to Figure 4 and Figure 5 to visually see the explanations that are given in the text below. Expert number 1 on German credit dataset, stated that the applicant would qualify for the loan if the `Payment Status of Previous Credit` *decreases* by 4, `Duration of Credit (month)` *increases* by 10, `Credit Amount` *increases* by 5702, `Account Balance` *increases* by 1 and `Age (years)` *increases* by 7. Expert number 2 on German credit dataset, suggested that the applicant would qualify for the loan if the `Duration of Credit (month)` *increases* by 57, `Credit Amount` *increases* by 10359 and `Purpose` *decreases* by 1. Expert number 3 on German credit dataset, stated that the applicant would qualify for the loan if the `Duration of Credit (month)` *increases* by 38, `Age (years)` *decreases* by 10 and `Payment Status of Previous Credit` *decreases* by 3. Expert number 1 on HMEQ credit dataset, stated that the applicant would qualify for the loan if the `LOAN` *increases* by 11544,`DEROG` *increases* by 2, `VALUE` *increases* by 373079, `CLAGE` *increases* by 456, `NINQ` *increases* by 8, `CLNO` *increases* by 42 and `DEBTINC` *decreases* by 14. Expert number 2 on HMEQ credit dataset, stated that the applicant would qualify for the loan if the the `CLAGE` *increases* by 362. Expert

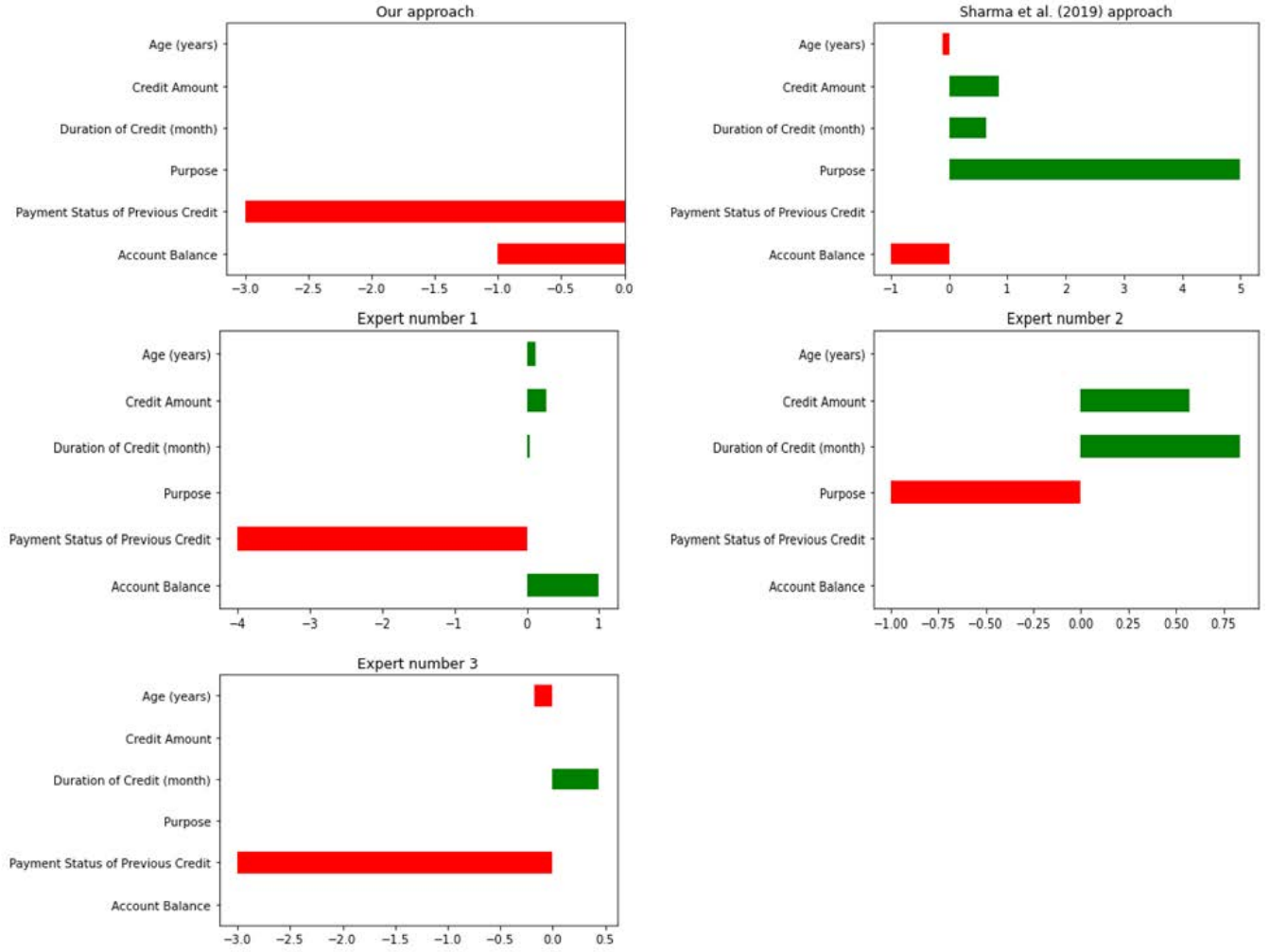

FIGURE 4. Explanation on German credit dataset record using our approach, Sharma et al. [13] and experts. The red bars represent a decrease in a feature value and the green bars represent an increase in a feature value. The x-axis on each figure represent the change between the original feature value and the counterfactual feature value.

number 3 on HMEQ credit dataset, stated that the applicant would qualify for the loan if the the CLAGE *increases* by 304.

For counterfactuals that were generated by the experts, in some cases the features that needed to be changed overlapped with the features that were changed when using our approach to generate counterfactuals. The experts select features that are more influential in determining the output of the model. On the other hand, our approach selects features that are more likely to change the outcome of a prediction. Our approach can play a key role in credit scoring for black-box model explanations and the credit scoring experts can leverage off the explanations from our approach, and this can result in a human-machine relationship.

D. MEAN OPINION SCORE (MOS)

The set of features that were used in the counterfactual explanations which were generated when using our approach, were compared to the set of features that were chosen by the credit scoring experts. Let L denote the number of credit scoring experts and I denote the number of features used to generate the counterfactuals when using our approach. We define below a_l^i which determines whether there is an overlap between a feature from our approach and the ith-feature from the lth-expert opinion,

$$a_l^i = \begin{cases} 1, & \text{if } e_l^i = m^i \\ 0, & \text{otherwise,} \end{cases} \tag{21}$$

where e_l^i is the lth-expert feature opinion and m^i is the feature which is selected by our approach. Hence, the mean opinion score is given as follows,

$$MOS = \frac{1}{I}\sum_{i=1}^{I}\frac{1}{L}\sum_{l=1}^{L}a_l^i. \tag{22}$$

The *MOS* ranges between 0 and 1, values that are close to 1 would mean that the credit experts agree more with the features that need to be changed for the generation of a counterfactual using our approach. The values that are close

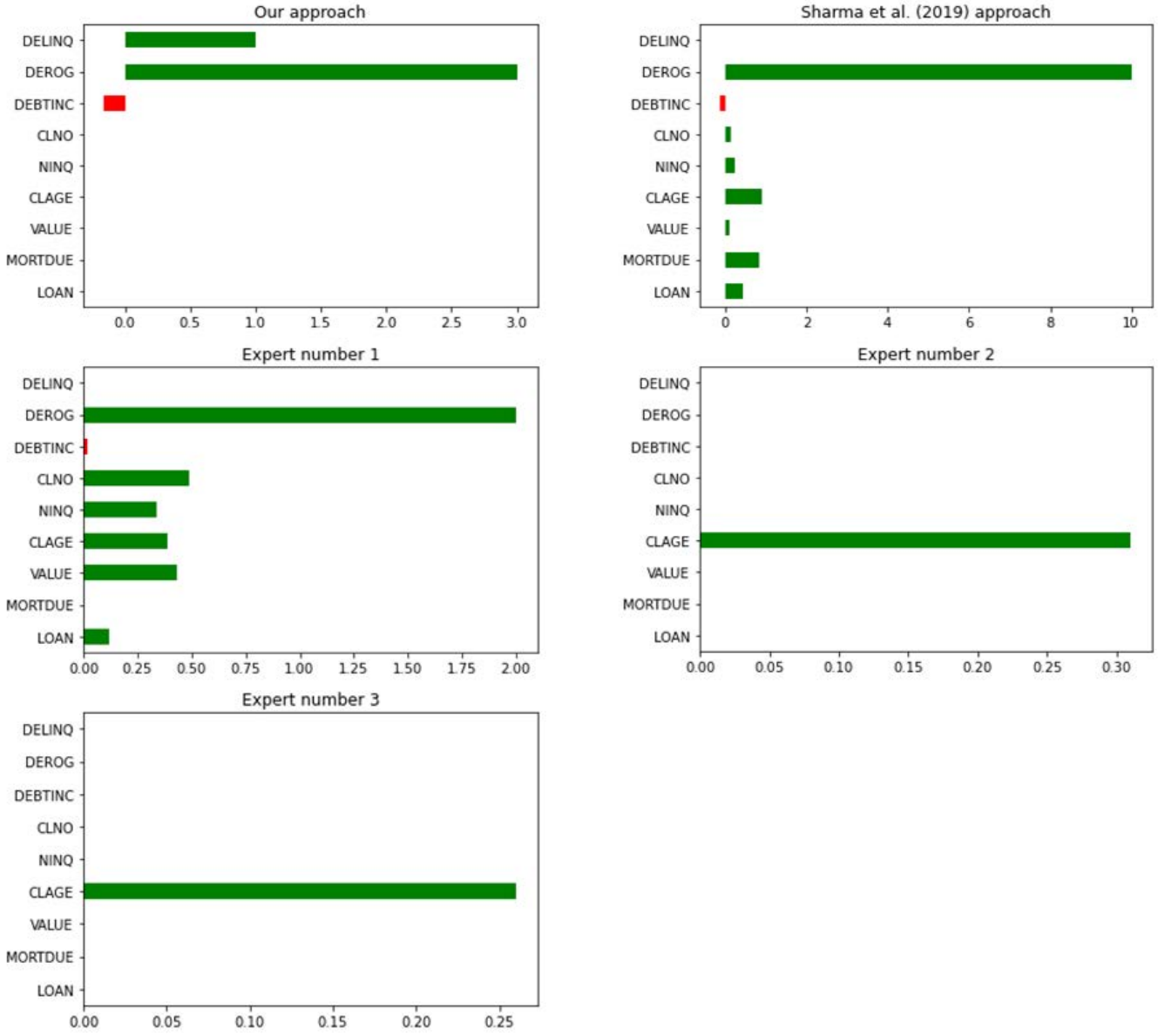

FIGURE 5. Explanation of on HMEQ credit dataset record using our approach, Sharma et al. [13] and experts. The red bars represent a decrease in a feature value and the green bars represent an increase in a feature value. The x-axis on each figure represent the change between the original feature value and the counterfactual feature value.

to 0, it would mean that the credit experts agree less with the values that are suggested for creating a counterfactual using our approach. For German credit scoring dataset, the $MOS = 0.17$, indicating that the credit experts agree 17% with the features that needed to be changed to generate a counterfactual. For HMEQ credit dataset, the $MOS = 0.083$, this indicates that the credit experts agree 8.30% with the features that needed to be changed to generate a counterfactual. The low values of the MOS are due to the fact that our approach looks at sparse counterfactuals, this results in few features that need to be changed, whereas for credit scoring experts there is no restriction in the number of required features.

E. COMPARISON WITH STATE-OF-THE-ART COUNTERFACTUAL EXPLANATION METHODS

Table 4 shows comparisons with state-of-the-art counterfactual explanation methods. Factors such as the black-box nature of the models, the applicability of the explanation approach using different classes of models (i.e. model-agnostic behaviour), involvement of domain experts, the sparseness of the counterfactuals, and quantitative assessments of the resulting counterfactuals, were used for comparison purposes. We observed that most methods are focusing on black-box models and model-agnostic behaviour of the explanations, except [3], [16], [30]. Our approach looks at all the suggested factors. The factors such as expert domain

TABLE 4. Comparison with state-of-the-art counterfactual methods with our approach. Legend: Q-Measured is Quantitatively Measured.

Source	Year	Black-box	Model-Agnostic	Experts	Sparse	Q-Measured
[11]	2018	✓	✓			
[3]	2018	✓	✓		✓	
[13]	2019	✓	✓			
[16]	2019	✓	✓		✓	✓
[31]	2019	✓	✓			
[32]	2019	✓	✓			✓
[12]	2019	✓	✓			
[33]	2020	✓	✓			
[15]	2020	✓	✓			
[30]	2020	✓	✓	✓	✓	
Our approach	2022	✓	✓	✓	✓	✓

knowledge, the sparseness of counterfactual explanations, and also the ability to quantitatively assess the generated counterfactuals, ensure the robustness of the generated counterfactual explanations. Please note that the list of sources in Table 4 is not exhaustive.

VI. CONCLUSION AND FUTURE WORK

The non-transparent nature of machine learning and deep learning models hampers the application of these models in credit scoring. We address this challenge of non-transparency by generating counterfactuals via a custom genetic algorithm to explain model predictions. We select predictive features and determine Spearman's rank correlations between the target flag and the predictors to enforce sparseness. We normalize all continuous features to expedite the generation of counterfactuals. We show the efficacy of the proposed approach on German and HMEQ credit scoring datasets. The experimental results indicate that the proposed approach efficiently generates sparse counterfactuals compared to similar methods. We also test the accuracy of our approach by using counterfactuals from credit scoring experts. Our results show an overlap between some features selected by the credit scoring experts and our approach in creating the counterfactuals.

Although the proposed method produces satisfactory results with the default parameter settings of the genetic algorithm, optimal parameter settings may improve the performance of the counterfactual generation. Furthermore, an optimal fitness function capturing different properties of counterfactuals using a genetic programming approach can enhance the performance of the proposed method. In addition, explaining the overall working mechanism of the black-box models instead of explaining individual instances can further improve the transparency and explainability of the credit scoring models.

ACKNOWLEDGMENT

The authors would like to thank the following credit risk experts for providing their subject matter expertise, Dr. Luyanda Ndlovu (Senior Credit Risk Specialist, South African Reserve Bank), Esinako Makaluza (Senior Credit Risk Specialist, South African Reserve Bank) and Lunga Dludla (Senior Manager, Credit Risk, African Bank), and also would like to thank the anonymous reviewers for providing valuable feedback on initial versions of this paper.

REFERENCES

[1] *Basel II: International Convergence of Capital Measurement and Capital Standards: A Revised Framework—Comprehensive Version, Bank for International Settlements*, BIS, Basel Committee Banking Supervision, Basel, Switzerland, 2006.

[2] P. Voigt and A. V. D. Bussche, *The EU General Data Protection Regulation (GDPR): A Practical Guide*, 1st ed. Cham, Switzerland: Springer, 2017.

[3] S. Wachter, B. Mittelstadt, and C. Russell, "Counterfactual explanations without opening the black box: Automated decisions and the GDPR," *Harvard J. Law Technol.*, vol. 31, no. 2, pp. 87–841, 2018.

[4] A. E. Khandani, A. J. Kim, and A. W. Lo, "Consumer credit-risk models via machine-learning algorithms," *J. Bank Financ.*, vol. 34, no. 11, pp. 2767–2787, Nov. 2010.

[5] C. Molnar, *Interpretable Machine Learning*. 2019. [Online]. Available: https://christophm.github.io/interpretable-ml-book/

[6] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should I trust you?: Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, pp. 1135–1144, 2016.

[7] H. Yang, C. Rudin, and M. Seltzer, "Scalable Bayesian rule lists," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 3921–3930.

[8] O. Gomez, S. Holter, J. Yuan, and E. Bertini, "Vice: Visual counterfactual explanations for machine learning models," in *Proc. 25th Int. Conf. Intell. User Interfaces*, 2020, pp. 531–535.

[9] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nature Mach. Intell.*, vol. 1, pp. 206–215, May 2019.

[10] S. Verma, J. P. Dickerson, and K. Hines, "Counterfactual explanations for machine learning: A review," 2020, *arXiv:2010.10596*.

[11] R. Mc Grath, L. Costabello, C. Le Van, P. Sweeney, F. Kamiab, Z. Shen, and F. Lecue, "Interpretable credit application predictions with counterfactual explanations," 2018, *arXiv:1811.05245*.

[12] R. Guidotti, A. Monreale, F. Giannotti, D. Pedreschi, S. Ruggieri, and F. Turini, "Factual and counterfactual explanations for black box decision making," *IEEE Intell. Syst.*, vol. 34, no. 6, pp. 14–23, Nov. 2019.

[13] S. Sharma, J. Henderson, and J. Ghosh, "CERTIFAI: Counterfactual explanations for robustness, transparency, interpretability, and fairness of artificial intelligence models," 2019, *arXiv:1905.07857*.

[14] A. Van Looveren, J. Klaise, G. Vacanti, and O. Cobb, "Conditional generative models for counterfactual explanations," 2021, *arXiv:2101.10123*.

[15] R. Poyiadzi, K. Sokol, R. Santos-Rodriguez, T. De Bie, and P. Flach, "FACE: Feasible and actionable counterfactual explanations," in *Proc. AAAI/ACM Conf. AI, Ethics, Soc.*, Feb. 2020, pp. 344–350.

[16] R. K. Mothilal, A. Sharma, and C. Tan, "Explaining machine learning classifiers through diverse counterfactual explanations," in *Proc. Conf. Fairness, Accountability, Transparency*, Jan. 2020, pp. 607–617.

[17] A. Artelt and B. Hammer, "Efficient computation of counterfactual explanations of LVQ models," 2019, *arXiv:1908.00735*.

[18] A. Van Looveren and J. Klaise, "Interpretable counterfactual explanations guided by prototypes," 2019, *arXiv:1907.02584*.

[19] K. Sokol and P. A. Flach, "Counterfactual explanations of machine learning predictions: Opportunities and challenges for ai safety," in *Proc. SafeAI@ AAAI*, 2019, pp. 1–4.

[20] M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, MA, USA: MIT Press, 1996.

[21] A. Hassanat, K. Almohammadi, E. Alkafaween, E. Abunawas, A. Hammouri, and V. B. S. Prasath, "Choosing mutation and crossover ratios for genetic algorithms—A review with a new dynamic approach," *Information*, vol. 10, no. 12, p. 390, Dec. 2019.

[22] D. Dua and C. Graff. (2017). *UCI Machine Learning Repository*. [Online]. Available: http://archive.ics.uci.edu/ml

[23] (2017). Kaggle. *Home Loan Equity Data*. Accessed: Jun. 17, 2021. [Online]. Available: https://www.kaggle.com/ajay1735/hmeq-data

[24] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proc. 14th Int. Joint Conf. Artif. Intell.*, vol. 2, Aug. 1995, pp. 1137–1143.

[25] X. Dastile, T. Celik, and M. Potsane, "Statistical and machine learning models in credit scoring: A systematic literature survey," *Appl. Soft Comput.*, vol. 91, Jun. 2020, Art. no. 106263.

[26] D. West, "Neural network credit scoring models," *Comput. Oper. Res.*, vol. 27, no. 11, pp. 1131–1152, Sep. 2000.

[27] S. Ha and H.-N. Nguyen, "Credit scoring with a feature selection approach based deep learning," in *Proc. MATEC Web Conf.*, vol. 54, Dec. 2016, p. 05004.

[28] Y. Xia, C. Liu, Y. Li, and N. Liu, "A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring," *Expert Syst. Appl.*, vol. 78, pp. 225–241, Jul. 2017.
[29] L. Munkhdalai, O.-E. Namsrai, and K. Ryu, "Credit scoring with deep learning," in *Proc. 4th Int. Conf. Inf., Syst. Converg. Appl.*, 2018, pp. 1–6.
[30] F. Cheng, Y. Ming, and H. Qu, "DECE: Decision explorer with counterfactual explanations for machine learning models," *IEEE Trans. Vis. Comput. Graphics*, vol. 27, no. 2, pp. 1438–1447, Feb. 2020.
[31] D. Mahajan, C. Tan, and A. Sharma, "Preserving causal constraints in counterfactual explanations for machine learning classifiers," 2019, *arXiv:1912.03277*.
[32] Y. Goyal, Z. Wu, J. Ernst, D. Batra, D. Parikh, and S. Lee, "Counterfactual visual explanations," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 2376–2384.
[33] P. Wang and N. Vasconcelos, "SCOUT: Self-aware discriminant counterfactual explanations," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 8981–8990.



XOLANI DASTILE received the bachelor's degree majored in mathematics and mathematical statistics and the M.Sc. degree in mathematical statistics from Rhodes University, South Africa. He is currently pursuing the Ph.D. degree in computer science in the field of deep learning with the University of the Witwatersrand, Johannesburg, South Africa. He is also a Senior Data Scientist in a fuel management company, South Africa. Previously, he worked in major banks in South Africa under credit risk departments as a Quantitative Analyst and a Data Scientist.



TURGAY CELIK received the Ph.D. degree from the University of Warwick, Coventry, U.K., in 2011. He is currently a Professor in digital transformation and the Director of the Wits Institute of Data Science, University of Witwatersrand, Johannesburg, South Africa. His research interests include the areas of computer vision, (explainable) artificial intelligence, (health) data science, data-driven optimal control, and remote sensing. He is an Associate Editor of *BMC Medical Informatics and Decision Making*, *IET ELL*, IEEE Access, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, and *SIVP* (Springer).



HANS VANDIERENDONCK (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees from Ghent University, Belgium, in 1999 and 2004, respectively. He is currently a Professor in high-performance and data-intensive computing with the School of Electronics, Electrical Engineering and Computer Science, and the Director of the Centre for Data Science and Scalable Computing, Institute on Electronics, Communications and Information Technology, Queen's University Belfast, Belfast, U.K. His research aims to build efficient and scalable computing systems for data-intensive applications.

2.4 Journal Article IV

X. Dastile and T. Celik, Multi-property Counterfactual Explanations with Multiple Properties in Credit Scoring, *Engineering Applications of Artificial Intelligence*, (In review)

Number of citations to date: 0
Publisher: Elsevier
Cite Score: 11
Impact Factor: 7.802
Abstracting and Indexing: Scopus (Elsevier)
Engineering Index Monthly
Web of Science

Engineering Applications of Artificial Intelligence

Counterfactual Explanations with Multiple Properties in Credit Scoring

--Manuscript Draft--

Manuscript Number:	EAAI-23-1780
Article Type:	Research paper
Keywords:	Counterfactual Explanations; Credit Scoring; Optimization; Deep Learning; eXplainable Artificial Intelligence
Corresponding Author:	Xolani Dastile, MSc University of the Witwatersrand Johannesburg Johannesburg, SOUTH AFRICA
First Author:	Xolani Dastile, MSc
Order of Authors:	Xolani Dastile, MSc
	Turgay Celik, PhD
Abstract:	Explainable artificial intelligence aims at revealing the reasons behind predictions from non-transparent classifiers. Researchers and practitioners in the machine learning community are contributing to developing techniques that explain predictions from non-transparent classifiers. For instance, an explanation technique called counterfactual explanation has recently been gaining traction across various domains. The interest in counterfactual explanations stems from the ability of the explanations to reveal what could have been different to achieve a desired outcome, as opposed to only highlighting features that contribute to a prediction. For example, for a customer that has been denied a loan by a bank, a counterfactual will highlight what needs to be different for the customer to qualify for the loan, and this will help the applicant with future loan applications. For a counterfactual to be considered effective, several counterfactual properties are assessed. This paper proposes a novel optimization formulation that generates counterfactual explanations that possess several properties simultaneously. The efficacy of the proposed method is assessed on a publicly available credit dataset. The results showed that there is a trade-off between validity and sparsity, which are both parts of a suite of counterfactual properties. Further, the results showed that our proposed approach does not compromise any of the properties as opposed to some of the counterfactual generating approaches.
Suggested Reviewers:	Dongshu Wang, PhD Zhengzhou University School of Electrical Engineering wangdongshu@zzu.edu.cn Dr Wang has done a research in particle swarm optimization, he has a good understand on the subject.
	Arnaud Van Looveren avl@seldon.io Arnaud has done some work on counterfactual explanations.

Powered by Editorial Manager® and ProduXion Manager® from Aries Systems Corporation

Manuscript Click here to view linked References

Counterfactual Explanations with Multiple Properties in Credit Scoring

Xolani Dastile[a,*], Turgay Celik[b,c,d]

[a] *School of Computer Science and Applied Mathematics, University of the Witwatersrand, Johannesburg, South Africa.*
[b] *School of Electrical and Information Engineering, University of the Witwatersrand, Johannesburg 2000, South Africa*
[c] *Wits Institute of Data Science, University of the Witwatersrand, Johannesburg 2000, South Africa*
[d] *Faculty of Engineering and Science, University of Agder, 4630 Kristiansand, Norway*

Abstract

Explainable artificial intelligence aims at revealing the reasons behind predictions from non-transparent classifiers. Researchers and practitioners in the machine learning community are contributing to developing techniques that explain predictions from non-transparent classifiers. For instance, an explanation technique called counterfactual explanation has recently been gaining traction across various domains. The interest in counterfactual explanations stems from the ability of the explanations to reveal what could have been different to achieve a desired outcome, as opposed to only highlighting features that contribute to a prediction. For example, for a customer that has been denied a loan by a bank, a counterfactual will highlight what needs to be different for the customer to qualify for the loan, and this will help the applicant with future loan applications. For a counterfactual to be considered effective, several counterfactual properties are assessed. This paper proposes a novel optimization formulation that generates counterfactual explanations that possess several properties simultaneously. The efficacy of the proposed method is assessed on a publicly available credit dataset. The results showed that there is a trade-off between validity and sparsity, which are both parts of a suite of counterfactual properties. Further, the results showed that our proposed approach does not compromise any of the properties as opposed to some of the counterfactual generating approaches.

Keywords: Counterfactual Explanations, Credit Scoring, Optimization, Deep Learning, eXplainable Artificial Intelligence

*Corresponding author
Email addresses: xdastile12@gmail.com (Xolani Dastile), celikturgay@gmail.com (Turgay Celik)

1. Introduction

Automated decisions that are generated by some of the machine learning techniques may have unfavorable consequences. For example, in a banking setting, if a loan application decision relies on a machine learning technique, the loan application may be denied due to unintended bias that may reside in data. As a result, the bank may be subjected to a regulatory fine. The consequence for the applicant may be that he/she is unable to send his/her child to university if the primary purpose of the loan was to pay university intuition.

An emerging field known as eXplainable Artificial Intelligence (XAI) seeks to remediate machine learning techniques' lack of explanation. There are several explanation techniques that can be found in the literature and these include global and local explanation techniques Guidotti (2022). A global explanation refers to the ability of the explanation technique to explain how an entire model makes its predictions. A local explanation refers to an explanation technique that only explains a single prediction at a time. Over and above that, the explanations are either referred to as ante-hoc or post-hoc Guidotti (2022). An ante-hoc explanation is intrinsically explainable. A post-hoc explanation seeks to explain the behavior of the already trained model. An example of an ante-hoc explanation is a decision tree and that of the post-hoc is a counterfactual explanation.

The focus of this paper is on counterfactual explanations. Counterfactual explanations are not new and have a long history in fields such as psychology, philosophy, and social sciences Verma et al. (2020), and now are being used in machine learning. Despite the prevalence of counterfactual explanations, there still remain limitations. The first limitation is that a counterfactual explanation assumes that a non-transparent classifier will not change over time, whereas in reality, it is possible for a classifier to change due to data drift Verma et al. (2020). The second limitation is that the counterfactual explanations are unable to handle missing data Verma et al. (2020). Despite these limitations, there is an appetite in using counterfactual explanations within the machine learning community. However, there is no consensus within the machine learning community on how to assess the performance of counterfactual explanations Guidotti (2022). The simplest way to measure the performance of counterfactual explanations is to look at several properties Verma et al. (2020). These properties include, *validity*, *sparsity*, *similarity*, *actionability*, *plausibilty* to mention a few. Please note that this list of properties is not exhaustive. It is rare to find counterfactual explanations that encompass all of the properties simultaneously Guidotti (2022). This paper investigates the use of counterfactual explanations in credit scoring, which possess several properties. Our contribution is the novel formulation of the optimization problem that generates counterfactuals with multiple properties simultaneously without compromising any of the counterfactual properties. In addition, according to the best of our knowledge, this is the first study to statistically test and contrast the performances of counterfactual explanations generated by different approaches.

The organization of this paper is as follows. In Section 2, related work is

presented. In Section 3 the methodology is discussed. The experiments are covered in Section 4. In Section 5 the results are presented. The conclusion is in Section 6.

2. Related Work

Machine learning classifiers that are unable to explain their predictions are not suitable for use in highly regulated entities such as the financial and healthcare sectors. There are several explanation techniques that can be found in the literature that mitigate the lack of explainability of the machine learning classifiers, and one of those techniques is the counterfactual explanation. For instance, Förster et al. (2021) proposed a novel approach to generate coherent counterfactual explanations. A counterfactual is coherent if the counterfactual scenario is realistic and typical as well as suitable to explain the factual situation Förster et al. (2021). The results showed that the proposed approach produces explanations that are significant and realistic and typical as well as suitable to explain the factual situation when compared to state-of-the-art counterfactual explanations. However, the actionability of the generated counterfactuals was not considered in Förster et al. (2021). Downs et al. (2020) proposed a Counterfactual Recourse Using Disentangled Subspaces (CRUDS) approach that generates algorithmic recourse to achieve more favorable outcomes. The CRUDS approach generated recourse that is more realistic and actionable than those of baseline counterfactual explanations. The generated counterfactuals obey causal relations between features. However, the approach uses only the training data and it is not based on the trained opaque model to generate counterfactuals. Laugel et al. (2018) proposed a method that identifies a close neighbor classified differently to the datapoint that needs to be explained, where the closeness definition includes a sparsity constraint. The empirical results showed that the proposed approach can be used to attain knowledge about the opaque classifier. The study did not consider the actionability and plausibility of the generated counterfactuals. Rathi (2019) proposed a model agnostic approach and a systemic implementation to generate counterfactual explanations using shapely additive explanations (SHAP). The approach proved to be model agnostic and provided a more consistent way of generating explanations. However, the approach did not consider sparsity, plausibility, and actionability.

It is important to have counterfactuals that are actionable and sparse. The sparseness property ensures simpler explanations that are easy to help an applicant to focus only on making a few changes. Also, the counterfactuals have to be realistic and actionable. For example, it would not benefit an applicant if the recommendation from a counterfactual suggests a decrease in the applicant's age.

Looveren and Klaise (2019) adopted the use of class prototypes to generate counterfactual explanations. The class prototypes are generated by an encoder. The use of class prototypes sped up the search process for the counterfactuals. The study did not consider the actionability of the generated counterfactuals. Samoilescu et al. (2021) proposed the use of a deep reinforcement

learning approach that converts the optimization process into an end-to-end learnable process, allowing the generation of counterfactual explanations in a single forward pass. The empirical results showed that the proposed approach can produce sparse, in-distribution counterfactual explanations across different datasets. The study by Samoilescu et al. (2021) looked at all of the counterfactual properties that are considered in our paper. However, the study only used a single optimization technique to generate counterfactuals as opposed to our approach. Fernández et al. (2020) proposed the use of a partial fusion of tree predictors from a random forest into a single decision tree using a modification of the classification and regression tree algorithm. The proposed approach obtained a counterfactual set that guaranteed an optimal counterfactual. However, the proposed method did not consider the actionability and plausibility of the generated counterfactuals.

Most of the above methods did not consider some of the key counterfactual properties such as sparsity and actionability when generating counterfactuals. We propose a model-agnostic approach that considers multiple counterfactual properties simultaneously.

3. Methodology

This section focuses on the methods that were used to generate counterfactuals. Before formulating the new optimization problem, definitions are provided. Suppose $\mathbf{x} \in \mathbb{R}^d$ is a feature vector, where d denotes the number of features. In the context of credit scoring, which is a binary classification, the black-box model g maps the feature vector $\mathbf{x}$ to a predicted class label $\hat{y} \in \{0, 1\}$, i.e.,

$$g : \mathbf{x} \rightarrow \hat{y}, \tag{1}$$

where 0 represents a non-default class and 1 represents a default class. The probability of $\mathbf{x}$ being mapped to the non-default class and default class is $P(g(\mathbf{x}) = 0)$ and $P(g(\mathbf{x}) = 1)$, respectively. Next, a counterfactual explanation and a counterfactual explainer are defined.

- **Counterfactual explanation**: A counterfactual explanation for a feature vector $\mathbf{x}$, is a feature vector $\mathbf{c}$ such that the output $\hat{y}' = g(\mathbf{c})$ differs from that of $\hat{y} = g(\mathbf{x})$, and that the distance between $\mathbf{x}$ and $\mathbf{c}$ is minimal Guidotti (2022).

- **Counterfactual explainer**: Is a function h that takes as input, the feature vector $\mathbf{x}$ of the dataset $\mathbf{X}$, the black-box model g, and it returns a set of counterfactuals $\mathbf{C} = \{\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_k\}$ Guidotti (2022).

To find a counterfactual $\mathbf{c}$, the following cost function h must be minimized:

$$h(\mathbf{x}, \mathbf{c}, g) = \gamma \times (1 - P(g(\mathbf{c}) = 0)) + \text{dist}(\mathbf{x}, \mathbf{c}) \tag{2}$$

subject to:

$$P(g(\mathbf{c}) = 0) \geq 0.5 \tag{3}$$

and

$$\frac{1}{1 + \exp^{-\zeta^i}} < \delta, \quad \forall i \in \{1, 2, \cdots, d\}, \tag{4}$$

and

$$c^i = \begin{cases} c^i, \text{ if } f^i \text{ is mutable} \\ f^i, \text{ otherwise,} \end{cases} \tag{5}$$

and

$$c^i \in [\min(\mathbf{f}^i), \max(\mathbf{f}^i)], \quad \forall i \in \{1, 2, \cdots, d\}, \tag{6}$$

where γ controls the contribution of the first term against the second term in Eqn. 2, and $\gamma > 1$, f^i and c^i represent the ith feature value for a given $\mathbf{x}$ and $\mathbf{c}$, respectively, $P(g(\mathbf{c}) = 0)$ denotes the probability of a counterfactual belonging to a non-default class, and $\mathbf{f}^i$ represents feature values of the ith feature in dataset $\mathbf{X}$ (i.e., $\mathbf{f}^i$ is column i in $\mathbf{X}$). The choice of distance is

$$\text{dist}(\mathbf{x}, \mathbf{c}) = \sum_{i=1}^{d^*} \frac{|c^i - f^i|}{MAD^i} + \sum_{i=d^*+1}^{d} \frac{|c^i - f^i|}{MD^i}, \tag{7}$$

where MAD^i and MD^i denote a mean absolute deviation and a mean deviation for the ith feature, respectively, and d^* denotes the number of continuous features Dastile et al. (2022). Please note that the first term of Eqn. 7 is the calculation of distance for continuous features and the second term is the distance calculation for categorical features. The mean absolute deviation for the ith feature in $\mathbf{X}$ is

$$MAD^i = \frac{1}{n} \sum_{j=1}^{n} |f_j^i - \bar{\mathbf{f}}^i|, \tag{8}$$

where n denotes the number of records and $\bar{\mathbf{f}}^i$ denotes the mean or average of the ith feature Dastile et al. (2022). The mean deviation for the ith feature is

$$MD^i = \frac{1}{m} \sum_{l=1}^{m} o_l |f_l^i - \tilde{\mathbf{f}}^i|, \tag{9}$$

where m is the number of categories, o_l is the frequency of each category and $\tilde{\mathbf{f}}^i$ is the median of the ith feature Dastile et al. (2022).

The five counterfactual properties that are included in the optimization problem are validity, sparsity, similarity, actionability, and plausibility. Validity measures the ability of a counterfactual to change the decision of an outcome Guidotti (2022). Sparsity measures the minimal changes in data features/variables that are required to effect the desired outcome Guidotti (2022).

Similarity measures how far a counterfactual is from the original instance Guidotti (2022). Actionability measures the ability of a counterfactual to change mutable features only Guidotti (2022). Plausibility ensures that the counterfactual and the original instance come from the same distribution and that the counterfactual is not an outlier Guidotti (2022).

The validity property is captured in Eqn. 3. The sparsity property is represented in Eqn. 4, the similarity property is captured in the actual cost function in Eqn. 2 by the distance calculation term, the actionability property is captured in Eqn. 5, and the plausibility is represented in Eqn. 6. The term ζ^i in Eqn. 4 measures the absolute percentage change between the ith feature value in the vector of interest $\mathbf{x}$ and the generated counterfactual $\mathbf{c}$ and is given as

$$\zeta^i = \frac{|c^i - f^i|}{f^i + \epsilon}, \quad \forall i \in \{1, 2, \cdots, d\}, \tag{10}$$

where ϵ is a small positive arbitrary number that guarantees a non-zero division. The absolute percentage change values ζ^i can be greater than 1. For ease of use, the logistic function (i.e., the left-hand side of Eqn. 4) is used to ensure the values are not greater than 1, where the range of the logistic function is $[0.5, 1]$. Since the absolute change values ζ^i are inputs to the logistic function and are non-negative, the lower value of the logistic function is 0.5.

To guarantee sparsity, a threshold value $0 < \delta < 1$ must be chosen so that there are only a few feature values that are changed in $\mathbf{x}$ when generating a counterfactual $\mathbf{c}$. Any feature values in $\mathbf{x}$ that result in the left-hand side of Eqn. 4 to be greater than δ, those feature values will not be changed when generating the counterfactual $\mathbf{c}$.

In the sequel, details of the optimization techniques that this study focuses on are provided and these include Genetic Algorithm (GA), Particle Swarm Optimization (PSO), and Bayesian Optimization (BO).

3.1. Genetic Algorithm (GA)

The genetic algorithm is an evolutionary search approach that is motivated by natural evolution Mitchell (1996). The genetic algorithm process involves five phases i.e. the *initial population*, the *fitness function*, the *selection*, the *cross-over*, and the *mutation*. The genetic algorithm involves a selection of individuals (based on some defined fitness function) from an initial population for reproduction/mating (i.e. cross-over). The selected individuals produce a diverse offspring (i.e. mutation) that inherits the initial individual's characteristics. The diverse offspring gets included in the next generation of the population. The process repeats itself until it converges to a solution. For more details on genetic algorithms please refer to our previous article Dastile et al. (2022).

3.2. Particle Swarm Optimization (PSO)

The PSO mimics the navigation of a flock of birds or a school of fish Dongshu et al. (2018). The original inventors of the PSO are Eberhart and Kennedy

(1995). Since its inception in 1995, PSO has gained a huge amount of improvements over time Dongshu et al. (2018).

The PSO is a stochastic search strategy and uses position vectors and velocities of the particles in a swarm Dongshu et al. (2018). The particles are assigned initial random position and velocity values. Each particle in the swarm has its own objective function which is governed by the position of the particle and the velocity at which the particle moves Dongshu et al. (2018). To find the optimal minimum for the objective function, the particle will have to consider its personal best-optimized value and the global best-optimized value in the entire swarm. The final global best value is found when all the particles have converged to the global best. The current position of particle j is represented by Z_j^t and the position of particle j in the next iteration/generation is

$$Z_j^{t+1} = Z_j^t + V_j^{t+1}, \tag{11}$$

where V_j^{t+1} represents the velocity of moving from position Z_j^t to position Z_j^{t+1} and t is the indexing/iteration value. The velocity at the next iteration is represented by the following formula

$$V_j^{t+1} = \omega V_j^t + a_1 r_1 (P_j^t - Z_j^t) + a_2 r_2 (G_j^t - Z_j^t). \tag{12}$$

The first part of Eqn. 12 is the *inertia* where ω represents the weight parameter Dongshu et al. (2018). The second part of Eqn. 12 is called *cognitive component* and P_j^t denotes the personal best-optimized value Dongshu et al. (2018). The third part of Eqn. 12 represents the *social component* and G_j^t is the global best-optimized value Dongshu et al. (2018). The particle j is a feature vector and the aim is to find the feature values that result in the global best-optimized value (i.e. the minimal value of the objective function) in a search space. Eqn. 12 has several parameters that may influence the performance of the PSO. The initial researchers who studied the impact of the weight parameter on the performance of the PSO were Shi and Eberhart (1998). The weight parameter ω balances exploration and exploitation Shi and Eberhart (1998). The exploitation refers to the particles converging to a known solution in the search space Shi and Eberhart (1998). The exploration refers to how well a particle explores other regions in a search space that may have possible optimized objective function values Shi and Eberhart (1998). The value of ω should gradually decrease with time/iterations Dongshu et al. (2018). Shi and Eberhart (1998) proposed that the weight parameter must change during the search from 0.9 to around 0.2.

The impact of the parameters a_1 and a_2 (which denote the acceleration of the particles towards the personal best solution and the global best solution, respectively), is assessed by setting one parameter to 0 and setting the other parameter to 2. If $a_1 = 0$, it means that there is no personal best in the search space and when $a_2 = 0$ it means that there is no information sharing between the particles, each particle depends on its personal best. It is noted that when $a_1 = 0$, the particles do converge to a minimum, however, when $a_2 = 0$, the

Algorithm 1 Particle Swarm Optimization (PSO) Van den Bergh and Engelbrecht (2004)

Require: The controlling parameters. The population of N particles.
  while The end condition is not satisfied **do**
    for Each particle **do**
      Calculate the objective function
      Update personal best (PBEST) if required
      Update global best (GBEST) if required
    end for
    Update the inertia weight
    for Each particle **do**
      Update the velocity (V)
      Update the position (Z)
    end for
  end while
  return GBEST

particles do not converge to a minimum. Thus, when $a_1 = 0$, this means that the exploration is at the lowest level but the exploitation is at the highest level. When $a_2 = 0$, this means that the exploration is at the highest level but the exploitation is at the lowest level. It is therefore key to balancing exploration and exploitation. In most cases $a_1 = a_2 = 2$ to ensure that there is a balance between the exploration and the exploitation Dongshu et al. (2018). Carlisle and Dozier (2001) conducted experiments and found that the optimal values for a_1 and a_2 are 2.8 and 1.3, respectively, and this was further confirmed by Schutte and Groenwold (2005).

The parameters r_1 and r_2 are random numbers ranging between 0 and 1, and their purpose is to create randomness in the search space.

3.3. Bayesian Optimization (BO)

Bayesian Optimization is another heuristic search approach. Bayesian optimization uses a notion of a surrogate function Shahriari et al. (2015). The purpose of the surrogate function is to approximate the original cost function during optimization Shahriari et al. (2015). Figure 1 illustrates the phenomenon of Bayesian Optimization. The first frame (1) in Figure 1 shows the original cost function $g(\mathbf{x})$ that needs to be optimized based on input $\mathbf{x}$. The surrogate function is represented by the dashed line in frame (2) of Figure 1 and it is formed based on sampled data points. In each iteration, new points are sampled and the surrogate function is updated based on the new sampled data points. When the iterations are exhausted, the process reaches a global minimum. The good thing about Bayesian Optimization is that there are no assumptions about the original cost function such as the need for the function to be differentiable as opposed to other techniques such as gradient descent. The original cost function may be complex (in terms of evaluation), however, the complexity is taken away

by the surrogate functions which are normally cheap (in terms of evaluation) Bergstra et al. (2011). In the following section, the Bayes Theorem is discussed, which serves as the basis for Bayesian Optimization.

3.3.1. Bayes Theorem

Before the discussion of the Bayes decision rule, the following notation is introduced:

- $P(g(\mathbf{x})|\mathbf{x})$ denotes the *a posterior probability* Duda et al. (2001) (i.e. the probability of the black-box model given the feature vector $\mathbf{x}$);
- $P(g(\mathbf{x}))$ denotes the *a priori probability* Duda et al. (2001) (i.e. the probability distribution of the black-box model);
- $p(\mathbf{x}|g(\mathbf{x}))$ represents the *conditional density* Duda et al. (2001) (i.e. the distribution of the feature vector $\mathbf{x}$ which results in $g(\mathbf{x})$);
- $p(\mathbf{x})$ represents the *evidence* Duda et al. (2001) (it can be viewed as the scaling factor that guarantees that the *posterior probabilities* sum to one).

The Bayes decision rule Duda et al. (2001) is defined as follows:

$$P(g(\mathbf{x})|\mathbf{x}) = \frac{p(\mathbf{x}|g(\mathbf{x}))P(g(\mathbf{x}))}{p(\mathbf{x})}. \tag{13}$$

Note that $p(\mathbf{x}|g(\mathbf{x}))$ is a multivariate Gaussian distribution density function. The Gaussian distribution density function is given as:

$$p(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{\frac{n}{2}}|\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\}$$

where the mean, $\boldsymbol{\mu}$, and covariance-matrix, $\boldsymbol{\Sigma}$, are estimated by using Expectation Maximization (EM) Reynolds (2015).

3.3.2. Surrogate and acquisition functions

The Gaussian distribution density function $p(\mathbf{x}|g(\mathbf{x}))$ also known as the *Tree Parzen Estimator* (TPE) Bergstra et al. (2011) is actually a surrogate function. The TPE is basically a distribution of the feature vector $\mathbf{x}$ given the cost function $g(\mathbf{x})$. The acquisition function is applied to the surrogate function to find better-sampled data points. The commonly used acquisition function is the *Expected Improvement* (EI) Bergstra et al. (2011). The formula for EI is:

$$\mathbf{EI}_{g(\mathbf{x})}(\mathbf{x}) := \int_{-\infty}^{\infty} \max(g(\mathbf{x}) - g(\mathbf{c}), 0) P(g(\mathbf{c})|\mathbf{x})\, dg(\mathbf{c}). \tag{14}$$

The term $P(g(\mathbf{c})|\mathbf{x})$ in Eqn. 14 is known as the Gaussian Process which represents a surrogate model Bergstra et al. (2011). The $g(\mathbf{c})$ is the predicted value for counterfactual $\mathbf{c}$ and $g(\mathbf{x})$ is the predicted value for the feature vector of interest $\mathbf{x}$. The aim is to maximize the acquisition function Shahriari et al. (2015) to find better data points (i.e. feature vectors) when sampling, which

will potentially result in a minimum for the cost function. Thereafter, the surrogate function is updated using the sampled data points. This is repeated until a global minimum/number of iterations is reached. The acquisition function is responsible for newly sampled data points and the notion of exploration and exploitation is applicable when looking for new data points Shahriari et al. (2015). The exploitation allows the acquisition function to sample where the surrogate function results in an optimized cost function. The exploration allows the acquisition function to sample in regions with lots of uncertainty in hopes that better data points will be found that will result in an optimized cost function. Thus, the acquisition function must balance the exploration and the exploitation when sampling for new data points.

3.4. Statistical significance testing

Statistical significance testing of the counterfactual generating approaches is assessed. The aim is to see if there are performance differences in the way counterfactual explanations are generated. The counterfactual properties that were used for significance testing are validity, similarity, and sparsity. All statistical significance tests in this study were based on *Friedman test* Friedman (1937) which is a non-parametric test. The Friedman test consists of four steps, i.e., a hypothesis, a test statistic, a rejection region, and a conclusion. Below are the steps involved in the Friedman test,

Step-1) State a null hypothesis H_0 and an alternative hypothesis H_1;

- H_0: The counterfactual generating approaches rank counterfactual performances for validity/similarity/sparsity the same.
- H_1: Atleast one counterfactual generating approach ranks counterfactual performances for validity/similarity/sparsity differently.

Step-2) Calculate the test statistic;

- In the following equation, q denotes the number of counterfactuals, o denotes the number of counterfactual generating approaches and R denotes the sum of ranks in each counterfactual generating approach,

$$\text{F-score} = \frac{12}{oq(q+1)} \sum_{j=1}^{q} R_j^2 - 3o(q+1).$$

Step-3) Define a rejection region;

- Rejection region is defined by the level of significance $\alpha = 0.05$, using the upper critical values from the chi-square table.
- The critical value is defined as a chi-squared value χ^2_{b-1}, where $b-1$ is the degrees of freedom.

Step-4) Draw a conclusion;

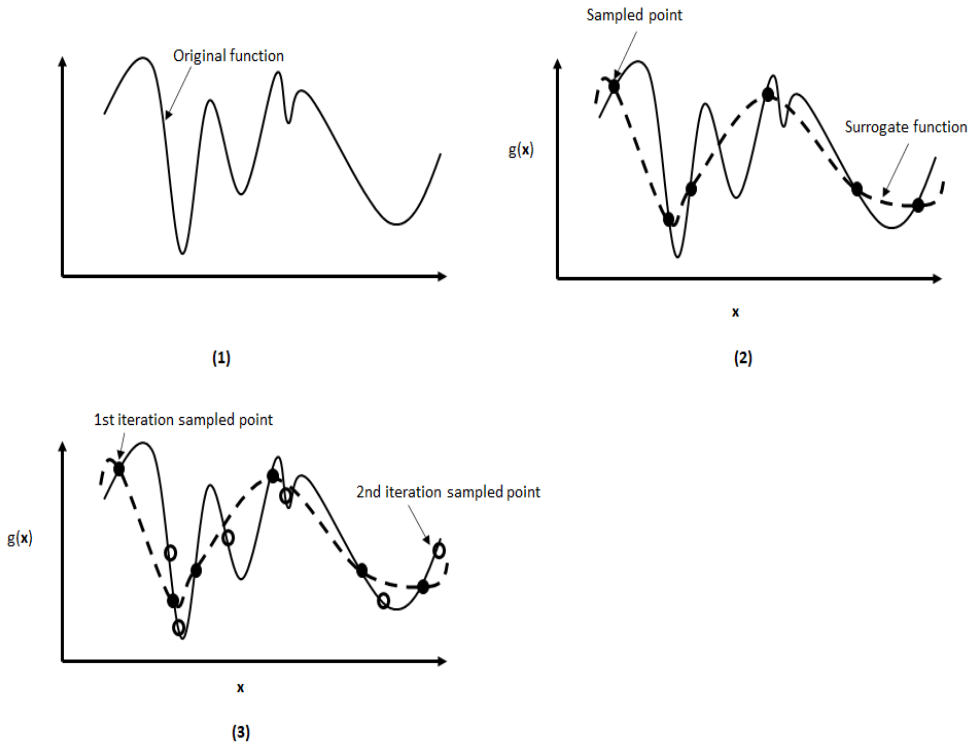

Figure 1: Bayesian Optimization Process Ye (2020 accessed 11 November 2022)

- The null hypothesis is rejected if F-score $> \chi^2_{b-1}$ or p-value < 0.05 and a conclusion is that there is no significant evidence that the counterfactual generating approaches rank counterfactual performances equally/perform similarly.

In essence, the Friedman test assesses whether there exist significant differences in how counterfactuals perform when generated by more than two approaches. To find exactly which counterfactual generating approaches differ in terms of counterfactual performances, a post-hoc test known as Nemenyi test Nemenyi (1963) is used.

4. Experiments

4.1. Black-box model

The black-box model that was used is a deep neural network consisting of three hidden layers. The first hidden layer had 20 neurons, the second hidden layer had 10 neurons and the third hidden layer had 6 neurons. The input

Algorithm 2 Bayesian Optimization (PSO) Bergstra et al. (2011)

Require: Number of iterations T, cost function g, surrogate function M and acquisition function S
$\mathcal{H} = (\mathbf{x}, g(\mathbf{x}))$
for Each iteration t **do**
$\mathbf{c} \leftarrow \underset{\mathbf{x}}{\operatorname{argmin}}\, S(\mathbf{x}, M_{t-1})$
$\mathcal{H} \leftarrow \mathcal{H} \cup (\mathbf{c}, g(\mathbf{c}))$
Fit new model to M_t
end for
Sort $\mathcal{H}$ using $g(\mathbf{c})$ in ascending order
return First pair of $\mathcal{H}$

layer and the output layer had 12 and 2 neurons, respectively. The activation functions were *Rectified Linear Unit* (ReLu).

4.2. Data

The dataset that was used in this study is the German credit dataset Kaggle (2017 accessed 17 June 2022). The credit dataset is publicly available on *Kaggle* website. The dataset has 20 features, where 7 of the features are continuous and the other 13 are categorical. The German credit dataset has features such as `status of existing checking account`, `duration in month`, `credit history`, `purpose`, `Age (years)`, `Sex & Marital Status` and `Length of current employment` to mention the least. Features such as `Age (years)` and `Sex & Marital Status` are treated as immutable. All other features are mutable. A pre-processing to select predictive features was performed, and the features were reduced to 12. The response variable is binary where one class refers to default and the other to non-default.

4.3. Optimization algorithms' parameters

The genetic algorithm had a population size of 100, the mutation probability was set to 0.01, the crossover probability was 0.2 and the number of iterations was 500. The particle swarm optimization had a population size (i.e. the number of particles) of 40, the weight parameter was decreasing from 0.9 to 0.2, the acceleration parameters were set to 2 and the number of iterations was 50. The maximum iterations parameter for the Bayesian optimization was set to 1000. For the optimization formulation, the threshold value δ was set to 0.9, and there was no scientific approach that was used to come up with the threshold value. The threshold value is provided by a user and is user dependent. If the user wants to make counterfactuals sparser, the threshold value will be small.

5. Results

This section shows the results that were obtained for counterfactuals using three optimization techniques, i.e., Genetic Algorithm, Particle Swarm Opti-

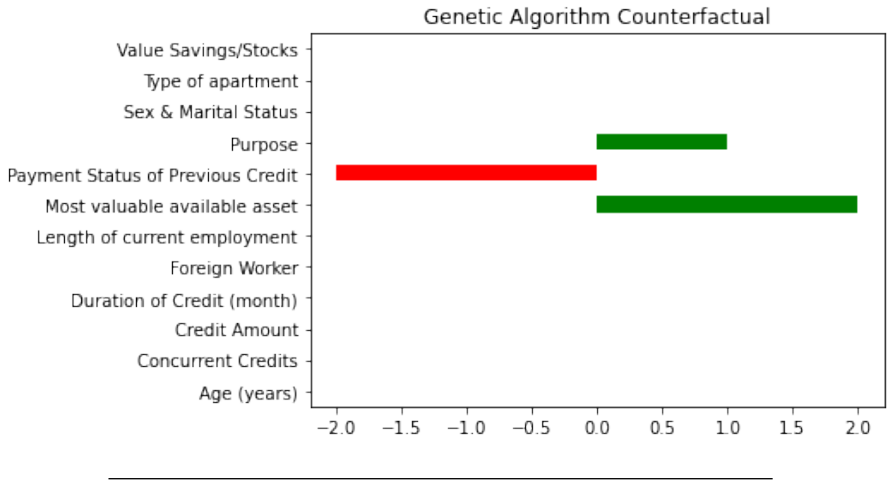

Figure 2: A visual explanation of the counterfactual **c** generated from Genetic Algorithm process. The red bars denote a decrease in a feature and the green bars denote an increase in a feature.

mization, and Bayesian Optimization. In addition, qualitative and quantitative countarfactual comparisons were performed.

5.1. Genetic Algorithm

Figure 2 shows the visual explanation of the counterfactual from GA. The counterfactual explanation suggests that for the feature vector of interest **x** to qualify for a loan application, the `Purpose` must increase by 1, the `Payment Status of Previous Credit` must decrease by 2 and the `Length of current employment` must increase by 2.

5.2. Particle Swarm Optimization

Figure 3 shows the visual explanation of the counterfactual from PSO. The counterfactual explanation suggests that for the feature vector of interest **x** to qualify for a loan application, the `Purpose` and `Concurrent Credits` must both decrease by 2, the `Type of apartment` must increase by 1 and the `Duration of Credit (month)` must increase by 38.

5.3. Bayesian Optimization

Figure 4 shows the visual explanation of the counterfactual from BO. The counterfactual explanation suggests that for the feature vector of interest **x** to qualify for a loan application, the `Purpose` must increase by 3, the `Payment Status of Previous Credit` must decrease by 4, the `Most valuable available asset` must increase by 1 and the `Concurrent Credits` must decrease by 2.

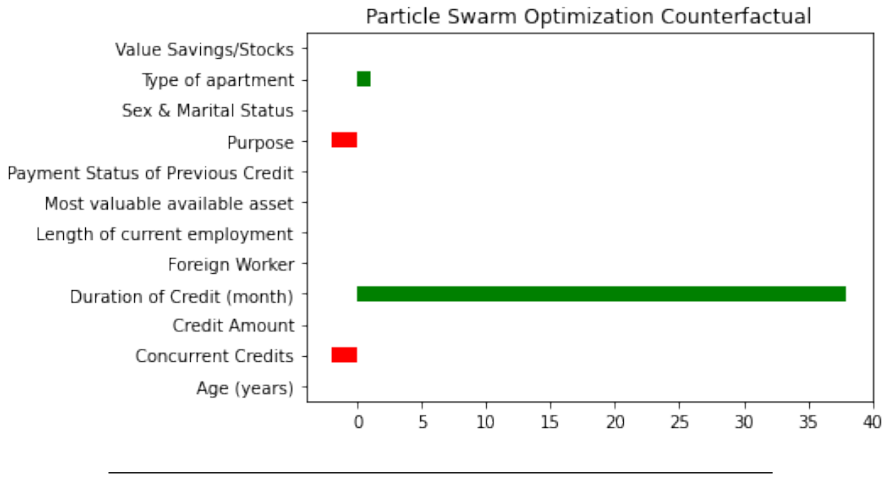

Figure 3: A visual explanation of the counterfactual $\mathbf{c}$ generated from Particle Swarm Optimization process. The red bars denote a decrease in a feature and the green bars denote an increase in a feature.

5.4. Explainer Comparison

Three optimization techniques that generate counterfactuals were proposed. The aim is to see which optimization technique results in a better explainer function h. According to Guidotti (2022), explainers can be compared using their properties such as *efficiency*, *fairness*, and *stability*. The efficiency property will not be considered here for comparisons purpose, since the optimization techniques are using different hyper-parameters and the hyper-parameters may influence the processing time of each optimization technique. The fairness property is defined in Guidotti (2022) as the ability of the explainer h to give the same decision based on the same feature changes in $\mathbf{x}$ irrespective of the demographic group. Suppose a feature vector $\mathbf{x}$ is rejected for a loan application (i.e. $g(\mathbf{x}) = 1$), and that $\mathbf{c}_s$ and $\mathbf{c}_t$ are counterfactuals for feature vector $\mathbf{x}$ where the *gender* feature value for $\mathbf{c}_s$ is male and for $\mathbf{c}_t$ is female. The explainer h is fair if and only if the features that are changed in $\mathbf{x}$ are the same for $\mathbf{c}_s$ and $\mathbf{c}_t$ and that

$$g(\mathbf{c}_s) = g(\mathbf{c}_t) = 0. \tag{15}$$

The stability (also known as *robustness*) property is defined in Guidotti (2022) as the ability of the explainer to generate similar counterfactuals (i.e. $\mathbf{c}_1$ and $\mathbf{c}_2$) when the feature vectors (i.e. $\mathbf{x}_1$ and $\mathbf{x}_2$) of interest are similar and their predictions are the same (i.e. $g(\mathbf{x}_1) = g(\mathbf{x}_2)$). The similarity will be measured by the cosine similarity given as

$$cosine(\mathbf{c}_1, \mathbf{c}_2) = \frac{\mathbf{c}_1 \cdot \mathbf{c}_2}{\| \mathbf{c}_1 \| \| \mathbf{c}_2 \|}. \tag{16}$$

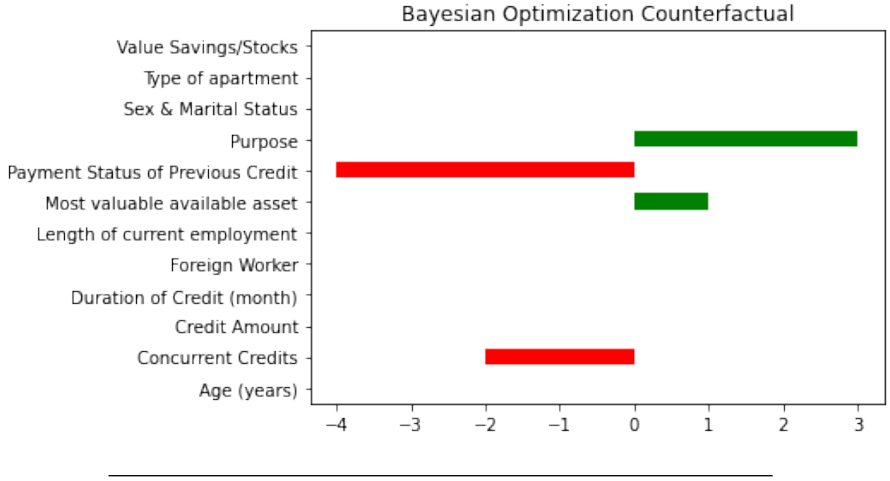

Figure 4: A visual explanation of the counterfactual **c** generated from Bayesian Optimization process. The red bars denote a decrease in a feature and the green bars denote an increase in a feature.

	Optimization Techniques		
Properties	GA	PSO	BO
fairness	yes	yes	yes
robustness	0.99	0.99	0.99

Table 1: Comparison of the explainer function h across all three optimization techniques. The legend: GA (Genetic Algorithm), PSO (Particle Swarm Optimization), BO (Bayesian Optimization)

The cosine similarity has values between 0 and 1 (inclusive). The closer the cosine similarity value is to 1, the more similar the feature vectors. The closer the cosine similarity is to 0, the more dissimilar are the feature vectors. The results of fairness and robustness are shown in Table 1. The results show that the explainer h performs similarly across all three optimization techniques. Thus, the explainer h is fair and robust across all three optimization techniques.

5.5. Qualitative comparison of counterfactuals

One way to assess the performances of counterfactuals is to use a group of experts. The experts have first-hand experience and knowledge domain. It is challenging to get hold of experts since they always have busy schedules. However, in our previous article Dastile et al. (2022), a group of experts was able to qualitatively assess counterfactuals. Their main focus was on actionability, plausibility, and sparsity of the counterfactuals. The first author of this article has domain knowledge as he has worked as a credit risk model developer. The first author agrees that to have effective and plausible counterfactuals, only a few features need to be changed and the changes must be within the variable/feature ranges to avoid having outliers. Thus, counterfactuals that do not

Source	Year	Validity	Similarity	Sparsity	Actionability	Plausibility
Sharma et al. (2019)	2019	✓	✓		✓	
Lucic et al. (2019)	2019	✓	✓	✓		
Poyiadzi et al. (2020)	2020	✓	✓			✓
Yang et al. (2021)	2021	✓	✓			✓
Dastile et al. (2022)	2022	✓	✓	✓		✓
Our Approach	2022	✓	✓	✓	✓	✓

Table 2: Comparing state-of-the-art counterfactual methods with our approach.

possess properties such as sparsity, actionability, and plausibility are deemed as not useful in the real world of credit risk scoring.

Based on what the experts noted in our previous article Dastile et al. (2022), the counterfactuals are qualitatively assessed in Table 2. Counterfactual properties such as validity, similarity, sparsity, actionability, and plausibility were used for comparison purposes. Table 2 shows that most methods are focusing on validity and similarity. Our approach looks at all the suggested counterfactual properties. There is only one method Sharma et al. (2019) that focuses on the actionability of the generated counterfactuals. Please note that the list of sources in Table 2 is not exhaustive.

5.6. Quantitative comparison of counterfactuals

Statistical significance tests of the counterfactual generating approaches in terms of counterfactual performances using validity, similarity, and sparsity were performed. The validity property is assessed using the probability of counterfactual $\mathbf{c}$ belonging to the non-default class, i.e., $P(g(\mathbf{c}) = 0)$. The higher the probability, the more valid the counterfactual is. The similarity property is assessed using the distance between the record of interest $\mathbf{x}$ and the generated counterfactual $\mathbf{c}$, i.e., $\text{dist}(\mathbf{x}, \mathbf{c})$. The lower the distance, the more similar $\mathbf{x}$ and $\mathbf{c}$ are. The sparsity property is measured by using the following formula

$$\text{sparsity} = \frac{\sum_{(c^i = f^i)} 1}{d}, \quad \forall i = \{1, 2, \cdot, d\}. \tag{17}$$

Eqn. 17 counts the number of features that have the same values between $\mathbf{x}$ and $\mathbf{c}$. The higher the sparsity value, the more sparse the generated counterfactual is. This study is compared quantitatively to Dastile et al. (2022), Sharma et al. (2019), and Wachter et al. (2017) using counterfactual properties (i.e. similarity, validity, and sparsity). There were 30 defaulted records that were randomly selected in the dataset, and counterfactuals were generated for the randomly selected records using approaches from Dastile et al. (2022), Sharma et al. (2019), and Wachter et al. (2017). Figure 5 shows the results for each of the counterfactual properties.

By glancing at Figure 5, it is not clear on how to tell which method performs better when it comes to counterfactual properties. Hence, a statistical significance test is required to assess if the methods perform differently from each other. This brings us to the Friedman test. There are three hypotheses:



Figure 5: Performance analysis of our approach and that of Wachter et al. (2017), Sharma et al. (2019), and Dastile et al. (2022). Three counterfactual properties (i.e. sparsity, validity, and similarity) were compared using 30 samples that were selected randomly in the dataset.

Hypothesis-1) For validity;

- H_0: The counterfactual generating approaches rank counterfactual performances the same for validity.
- H_1: Atleast one counterfactual generating approach ranks counterfactual performances differently for validity.

Hypothesis-2) For similarity;

- H_0: The counterfactual generating approaches rank counterfactual performances the same for similarity.
- H_1: Atleast one counterfactual generating approach ranks counterfactual performances differently for similarity.

Hypothesis-3) For sparsity;

- H_0: The counterfactual generating approaches rank counterfactual performances the same for sparsity.

	Sharma et al. (2019)	Dastile et al. (2022)	Wachter et al. (2017)	**Our Approach**
Sharma et al. (2019)	1.00	0.01	0.00	0.00
Dastile et al. (2022)	0.01	1.00	0.00	0.90
Wachter et al. (2017)	0.00	0.00	1.00	0.00
Our Approach	0.04	0.90	0.00	1.00

Table 3: The p-values for the Nemenyi test on validity property. The p-values, at the intersections, that are less than the level of significance, i.e, 0.05, suggest that there is a significant difference in how the counterfactuals perform from different counterfactual generating approaches.

- H_1: Atleast one counterfactual generating approach ranks counterfactual performances differently for sparsity.

The p-values for the hypotheses testing are 5.09e-11, 3.19e-12, and 1.91e-08, for validity, similarity, and sparsity, respectively. Since all the p-values are less than the level of significance, i.e., 0.05, we reject the null hypotheses and conclude that atleast one counterfactual generating approach ranks counterfactual performances differently for each of the counterfactual properties. A post-hoc test is required to identify which of the counterfactual generating approaches differs significantly in terms of counterfactual performances for each of the counterfactual properties. Hence, a Nemenyi test was used for that. The Nemenyi test generates a cross-tabular table consisting of p-values, for any two counterfactual generating approaches where they intersect, if the p-value is less than the level of significance, the conclusion is that those two counterfactual generating approaches are significantly different in terms of the counterfactual performances. Looking at Table 3, where Wachter et al. (2017) intersects with all other approaches, the p-values are less or equal to 0.01, and testing at 0.05 level of significance, the conclusion is that for Wachter et al. (2017) the counterfactual performances for validity differ significantly with other approaches. This is confirmed in Figure 5 under the validity graph, Wachter et al. (2017) has high validity scores. For similarity property, Table 4 shows that Sharma et al. (2019) significantly differs compared to other approaches since the p-values where they intersect are 0.00. This is also shown in Figure 5 under the similarity graph, Sharma et al. (2019) has the highest similarity scores compared to other approaches. Please note that the higher the similarity score, the more dissimilar the counterfactual to the data point of interest is. This means that Sharma et al. (2019) performs poorly compared to other approaches when it comes to similarity property. For sparsity property, Table 5 shows that at 0.05 level of significance, Sharma et al. (2019), Dastile et al. (2022) and our proposed approach differ significantly compared to that of Wachter et al. (2017). Overall, the results show that Wachter et al. (2017) is more valid but less sparse Figure 5. In addition, Dastile et al. (2022) is more sparse but less valid Figure 5. This suggests that there is a trade-off between validity and sparsity. Looking at Figure 5, our proposed approach does not compromise any of the counterfactual properties.

	Sharma et al. (2019)	Dastile et al. (2022)	Wachter et al. (2017)	**Our Approach**
Sharma et al. (2019)	1.00	0.00	0.00	0.00
Dastile et al. (2022)	0.00	1.00	0.84	0.38
Wachter et al. (2017)	0.00	0.84	1.00	0.84
Our Approach	0.00	0.38	0.84	1.00

Table 4: The p-values for the Nemenyi test on similarity property. The p-values, at the intersections, that are less than the level of significance, i.e, 0.05, suggest that there is a significant difference in how the counterfactuals perform from different counterfactual generating approaches.

	Sharma et al. (2019)	Dastile et al. (2022)	Wachter et al. (2017)	**Our Approach**
Sharma et al. (2019)	1.00	0.90	0.00	0.90
Dastile et al. (2022)	0.90	1.00	0.00	0.90
Wachter et al. (2017)	0.00	0.00	1.00	0.00
Our Approach	0.00	0.90	0.00	1.00

Table 5: The p-values for the Nemenyi test on sparsity property. The p-values, at the intersections, that are less than the level of significance, i.e, 0.05, suggest that there is a significant difference in how the counterfactuals perform from different counterfactual generating approaches.

6. Conclusion

In this article, the challenge of the non-transparent nature of machine learning models in credit scoring is addressed using counterfactual explanations. Three optimization techniques such as genetic algorithm, particle swarm optimization, and Bayesian optimization are used to generate the counterfactuals. The efficacy of the proposed approach is demonstrated using the German credit dataset which is widely used in credit scoring literature. The results indicate that the proposed approach generates actionable counterfactuals that encompass several properties simultaneously. In addition, the results indicate that the cost function of the proposed approach is robust and fair across all three optimization techniques. Further, the performances of the counterfactuals which are generated using different approaches are quantitatively compared using validity, similarity, and sparsity. The empirical results show that there is a trade-off between validity and sparsity. Furthermore, the empirical results show that our proposed approach does not compromise any of the three counterfactual properties (i.e. validity, similarity, and sparsity). In addition, when looking at qualitative comparisons, our approach encompasses more counterfactual properties compared to some of the state-of-the-art methods. A missing counterfactual property that needs to be assessed in the future study is causal relations between features.

Acknowledgments

The first author would like to thank Professor Turgay Celik for making funds available for his (first author) PhD study.

References

Van den Bergh, F., Engelbrecht, A.P., 2004. A cooperative approach to particle swarm optimization. IEEE transactions on evolutionary computation 8, 225–239.

Bergstra, J., Bardenet, R., Bengio, Y., Kégl, B., 2011. Algorithms for hyper-parameter optimization, in: Shawe-Taylor, J., Zemel, R., Bartlett, P., Pereira, F., Weinberger, K. (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc.

Carlisle, A., Dozier, G., 2001. An off-the-shelf pso, in: in: Proceeding of Workshop on Particle Swarm Optimization.

Dastile, X., Celik, T., Vandierendonck, H., 2022. Model-agnostic counterfactual explanations in credit scoring. IEEE Access 10, 69543–69554.

Dongshu, W., Dapei, T., Lei, L., 2018. Particle swarm optimization algorithm: an overview. Soft Computing 22, 387 – 408.

Downs, M., Chu, J., Yacoby, Y., Doshi-Velez, F., WeiWei, P., 2020. Cruds: Counterfactual recourse using disentangled subspaces. ICML Workshop on Human Interpretability in Machine Learning , 1–23.

Duda, R.O., Hart, P.E., Stork, D.G., 2001. Pattern Classification (2nd Ed). Wiley.

Eberhart, R., Kennedy, J., 1995. A new optimizer using particle swarm theory, in: MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, pp. 39–43.

Fernández, R.R., Martín de Diego, I., Aceña, V., Fernández-Isabel, A., Moguerza, J.M., 2020. Random forest explainability using counterfactual sets. Information Fusion 63, 196–207.

Friedman, M., 1937. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. Journal of the American Statistical Association 32, 675–701.

Förster, M., Hühn, P., Klier, M., Kluge, K., 2021. Capturing users' reality: A novel approach to generate coherent counterfactual explanations.

Guidotti, R., 2022. Counterfactual explanations and how to find them: literature review and benchmarking. Data Mining and Knowledge Discovery , 1–55.

Kaggle, 2017 accessed 17 June 2022. Home Loan Equity Data. URL: `https://www.kaggle.com/ajay1735/hmeq-data`.

Laugel, T., Lesot, M.J., Marsala, C., Renard, X., Detyniecki, M., 2018. Comparison-based inverse classification for interpretability in machine learning, in: Medina, J., Ojeda-Aciego, M., Verdegay, J.L., Pelta, D.A., Cabrera, I.P., Bouchon-Meunier, B., Yager, R.R. (Eds.), Information Processing and Management of Uncertainty in Knowledge-Based Systems. Theory and Foundations, Springer International Publishing, Cham. pp. 100–111.

Looveren, A.V., Klaise, J., 2019. Interpretable counterfactual explanations guided by prototypes. CoRR abs/1907.02584.

Lucic, A., Oosterhuis, H., Haned, H., de Rijke, M., 2019. Actionable interpretability through optimizable counterfactual explanations for tree ensembles. CoRR .

Mitchell, M., 1996. An Introduction to Genetic Algorithms. MIT Press.

Nemenyi, P., 1963. Distribution-Free Multiple Comparisons. PhD dissertation. Princeton University.

Poyiadzi, R., Sokol, K., Santos-Rodriguez, R., De Bie, T., Flach, P., 2020. Face: feasible and actionable counterfactual explanations, in: Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society, pp. 344–350.

Rathi, S., 2019. Generating counterfactual and contrastive explanations using SHAP. CoRR abs/1906.09293.

Reynolds, D., 2015. Gaussian Mixture Models. Springer US, Boston, MA. pp. 827–832.

Samoilescu, R., Looveren, A.V., Klaise, J., 2021. Model-agnostic and scalable counterfactual explanations via reinforcement learning. CoRR abs/2106.02597.

Schutte, J., Groenwold, A., 2005. A study of global optimization using particle swarms. Journal of Global Optimization 31, 93 – 108.

Shahriari, B., Swersky, K., Wang, Z., Adams, R.P., De Freitas, N., 2015. Taking the human out of the loop: A review of bayesian optimization. Proceedings of the IEEE 104, 148–175.

Sharma, S., Henderson, J., Ghosh, J., 2019. Certifai: Counterfactual explanations for robustness, transparency, interpretability, and fairness of artificial intelligence models. arXiv preprint arXiv:1905.07857 .

Shi, Y., Eberhart, R., 1998. A modified particle swarm optimizer, in: 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360), pp. 69–73.

Verma, S., Dickerson, J.P., Hines, K., 2020. Counterfactual explanations for machine learning: A review. CoRR abs/2010.10596.

Wachter, S., Mittelstadt, B., Russell, C., 2017. Counterfactual explanations without opening the black box: Automated decisions and the gdpr .

Yang, F., Alva, S.S., Chen, J., Hu, X., 2021. Model-based counterfactual synthesizer for interpretation. CoRR .

Ye, A., 2020 accessed 11 November 2022. The Beauty of Bayesian Optimization, Explained in Simple Terms. URL: https://medium.com/towards-data-science/the-beauty-of-bayesian-optimization-explained-in-simple-terms-81f3ee13b10f.

Declaration of interests

☒The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Chapter 3

Discussion and Conclusion

3.1 Discussion

This chapter discusses all four articles in this thesis and summarizes all the findings in all four articles.

3.1.1 Article I

This literature survey has shown that ensemble classifiers perform better than individual classifiers. The ensemble classifiers remove the idea of relying on one classifier to make a decision. In an ensemble, base classifiers make their own decisions, and the final decision is based on a voting mechanism. This reduces potential bias/variance, and as a result, this ensures that the model does not underfit/overfit. However, it is difficult to interpret and explain decisions from ensemble methods. Thus, financial institutions are less interested in models that cannot explain their (i.e. models') predictions. Similarly, deep learning models show better performances and suffer from the lack of explaining their predictions. This limitation is discussed in the literature survey, and the methods that deal with this limitation are also discussed. The literature survey focused on the performances of different classifiers and tried to identify a consensus amongst the researchers regarding a method that outperforms other methods. The consensus of the best performing method is hard to reach because researchers have done different data pre-processing techniques. There is no certainty in knowing how these pre-processing techniques have improved the models' performances. This literature survey could have also focused on finding the impact of pre-processing techniques on the models' performances. Further, this literature survey could have done a quantitative analysis by repeating the experiments done on the research articles. This could have ascertained the

results that were obtained in the research articles. To improve the literature survey, the authors recommend a taxonomy approach. The models have to be grouped according to their types, how their data was prepared, and according to the performance metrics. This will then allow a better comparison between the methods. Further, the literature survey should have a section on the taxonomy of explanation techniques in credit scoring. The authors did not consider the explanation techniques' taxonomy when writing the literature survey. This was due to the hypothesis that was tested, mainly around the fact that deep learning techniques outperform classic statistical and machine learning methods.

3.1.2 Article II

The conversion of tabular data into images has helped apply 2D-convolutional neural networks in credit scoring. The conversion of the tabular data was novel and state-of-the-art prediction performances were obtained. This work showed that it is possible to convert tabular data into images and apply image processing techniques. The work further showed that it is possible to extract essential features from the explanations when explaining predictions made on the images. Although the work showed state-of-the-art results, this work can only be applied by those scorecard developers who have used or understand 2D-convolutional neural networks. The other issue with this work is the lack of validation of the explanations by the domain experts. The article did not measure the accuracy of the explanations from the domain experts' point of view. However, the article tried to mitigate this flaw by quantitatively assessing the explanations' accuracy. This is not sufficient, as we want this work to be adopted by banks. Future work should include domain experts in the loop to assess the validity of the explanations.

3.1.3 Article III

This article has included domain experts in the loop to measure the soundness of the generated counterfactual explanations. The mean opinion score was the metric of choice to measure the soundness of the counterfactual explanations. The article focused on the "opinions" of the experts and compared those "opinions" to the "opinions" of the counterfactuals. The "opinions" in this context refer to features that resulted in desired outcomes. The experts did not directly assess the generated counterfactuals simply because the authors did not want the experts to blindly agree/disagree with what the counterfactuals were suggesting. The authors were mainly interested in the opinions

of the domain experts. However, a follow-up study is required where the domain experts will be granted an opportunity to directly assess the generated counterfactuals to agree/disagree with the generated counterfactuals.

The article also used a broad assumption that the target flag is linearly related to the features. The sparsity will not hold if all the features are not linearly related to the target flag. Despite this flaw, the study tried to use monotonicity which measures the degree of association between the target flag and the features. Irrespective of the sign and magnitude of the correlation, monotonicity measures the association between features. Future study needs to look at a better way of measuring the association between the target flag and the features. The other possible way to cater to sparsity, other than looking at the correlation between the target flag and predictors, is to directly include sparsity when formulating the optimization problem.

3.1.4 Article IV

The article looked at the novel formulation of the optimization problem that generates counterfactuals with multiple properties simultaneously. The performance of counterfactual explanations is generally measured by using several properties. These properties include, *validity*, *sparsity*, *similarity*, *actionability*, and *plausibility*, to mention a few. Guidotti [36] posited that it is hard to find counterfactuals that encompass many properties simultaneously.

Further, the article looked at the properties (i.e., robustness and fairness) of the actual explainer to see if these properties are violated. This was tested using three optimization techniques: the genetic algorithm, particle swarm optimization, and Bayesian optimization. The results showed that the explainer h (i.e. the cost function) is robust and fair across all three optimization techniques. Further, the study statistically tested the performance differences between our proposed approach and some of the state-of-the-art counterfactual generating methods. The empirical results showed that there is a trade-off between validity and sparsity. The more valid a counterfactual is, the less sparse it is and vice versa. Our proposed approach seeks not to compromise any of the proposed counterfactual properties.

3.2 Conclusion

This thesis investigated the performances of statistical and classical machine learning models in credit scoring. A literature survey was conducted based on 74 journal and

conference articles published between 2010 and 2018. The literature survey identified some limitations, and these limitations include the omission of 1) macroeconomic factors, 2) different threshold values for classification, 3) the relationship between the response variable and the independent features, and 4) exploratory data analysis, mentioning a few. Further, a guiding machine learning framework was proposed for credit scoring practitioners. This framework consists of steps identified in credit scoring as critical requirements from our analysis. The results showed that the ensemble methods perform better than single classifiers. However, in the literature, the ensemble methods only used homogeneous base classifiers, and ensemble methods that consisted of heterogeneous base classifiers were not extensively explored.

Convolutional neural networks, a deep learning model, showed promising results in credit scoring. These promising results spurred us to investigate the performances of convolutional neural networks. Publicly available credit-scoring datasets were used to assess the performances of convolutional neural networks. The available tabular datasets were converted into images using a novel transformation method. The reason for converting the tabular datasets into images was the use of 2D convolutional neural networks. The other benefit of converting tabular datasets into images is that each column is considered a set of pixels; even if there are many independent features, those can be reduced into image pixels. After dataset conversions, the 2D convolutional neural networks were applied, and their predictions were explained using state-of-the-art explanation techniques. The explanations from the predictions showed that they possess good features for classification, which was supported by the state-of-the-art results obtained when explanations were used as training data for 2D convolutional neural networks. The only problem with the explanations was that they were not suggesting which actions should be taken to get the desired outcome (i.e., approval of a loan application).

The next step was to understand the reasons for declining loan applications and provide actions that would result in the approval of loan applications. This led us to the proposal of counterfactual explanations. A few features/variables must be changed to achieve the desired outcome for a counterfactual explanation to be effective and applicable. In other words, a counterfactual explanation needs to be sparse. We looked at independent features correlated with the response flag and only focused on those features to generate counterfactuals using a custom genetic algorithm. An assumption that was made is that the independent features are somehow correlated with the response flag. This assumption doesn't always hold. In cases where the assumption doesn't hold, monotonicity can be used to measure the association between independent features and the response flag/variable. Despite this limitation, the counterfactuals were then compared with those obtained from the experts using a mean opinion score. The results showed some overlaps between the independent features that were changed when using

our approach and that of the experts. Hence, the proposed method for explaining predictions can be used by credit scoring experts as a tool to assist in explaining automated decisions.

Further, we looked at the properties that make counterfactuals considered good. These properties include but are not limited to validity, similarity, plausibility, sparsity, and actionability. We developed a novel optimization formulation that captures these counterfactual properties mentioned in the preceding sentence. Further, the properties of the explainers/cost functions, such as fairness and stability/robustness, were assessed. The proposed explainer captured counterfactual properties and was fair and robust across three different optimization techniques.

Further, a statistical significance test was performed, and the proposed approach was contrasted with state-of-the-art approaches using counterfactual properties such as validity, similarity, and sparsity. The empirical results showed that there is a trade-off between validity and sparsity. Further, the empirical results showed that our proposed approach does not compromise counterfactual properties. By and large, the work in this thesis has shown that deep learning models (non-transparent and referred to as black-box models) can be applied in credit scoring without mistrusting their predictions.

The future works of this thesis should look at the inclusion of macroeconomic factors since any change in those factors would affect a change in loan application decisions. The impact of different threshold values when classifying an applicant must be investigated in credit scoring. The literature must explore the usage of 2D convolutional neural networks in credit scoring since these models perform better in other domains, such as computer vision [37–40]. A method that explains the entire system's behavior (i.e., a global explainer) needs to be developed for counterfactual explanations. The extent of trade-offs of counterfactual properties on prediction explanations needs to be addressed in future studies.

Bibliography

[1] N. Siddiqi. *Credit Risk Scorecards: Developing And Implementing Intelligent Credit Scoring*. SAS Publishing, 2005.

[2] Sentryo. The 4 industrial revolutions, 2017 accessed 5 May 2022. URL https://www.sentryo.net/the-4-industrial-revolutions/.

[3] M. Wooldridge. *Artificial intelligence*. United Kingdom: Penguin Random House UK, 2018.

[4] L. C. Thomas, J. Crook, and D. Edelman. *Credit Scoring and Its Applications*. Society for Industrial and Applied Mathematics, 2002.

[5] L. C. Thomas. A survey of credit and behavioural scoring: forecasting financial risk of lending to consumers. *International Journal of Forecasting*, 16(2):149 – 172, 2000.

[6] D. lter, E. Deniz, and O. Kocadagli. Hybridized artificial neural network classifiers with a novel feature selection procedure based genetic algorithms and information complexity in credit scoring. *Applied Stochastic Models in Business and Industry*, 37(2):203–228, 2021.

[7] H. Li, A. Feng, B. Lin, H. Su, Z. Liu, X. Duan, H. Pu, and Y. Wang. A novel method for credit scoring based on feature transformation and ensemble model. *PeerJ Computer Science*, 7:e579, 2021.

[8] C. Wu, S. Huang, C. Chiou, and Y. Wang. A predictive intelligence system of credit scoring based on deep multiple kernel learning. *Applied Soft Computing*, 111: 107668, 2021.

[9] E. Dumitrescu, S. Hue, C. Hurlin, and S. Tokpavi. Machine learning for credit scoring: Improving logistic regression with non-linear decision-tree effects. *European Journal of Operational Research*, 297(3):1178–1192, 2022.

[10] W. Yotsawat, P. Wattuya, and A. Srivihok. A novel method for credit scoring based on cost-sensitive neural network ensemble. *IEEE Access*, 9:78521–78537, 2021.

[11] S. Bequé, A.and Lessmann. Extreme learning machines for credit scoring: An empirical evaluation. *Expert Systems with Applications*, 86:42 – 53, 2017.

[12] M. Aláraj and M.F. Abbod. Classifiers consensus system approach for credit scoring. *Knowledge-Based Systems*, 104:89 – 105, 2016.

[13] L. Yu, X. Yao, S. Wang, and K.K. Lai. Credit risk evaluation using a weighted least squares svm classifier with design of experiment for parameter selection. *Expert Systems with Applications*, 38(12):15392 – 15399, 2011.

[14] Z. Zhao, S. Xu, B. H. Kang, M. M. J. Kabir, Y. Liu, and R. Wasinger. Investigation and improvement of multi-layer perceptron neural networks for credit scoring. *Expert Systems with Applications*, 42(7):3508 – 3516, 2015.

[15] Gartner. Gartner hype cycle, 2019 accessed 6 June 2022. URL https://www.gartner.com/en/research/methodologies/gartner-hype-cycle.

[16] W. Samek and R. K. Müller. Towards Explainable Artificial Intelligence. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11700:5–22, 2019.

[17] A. Adadi and M. Berrada. Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6:52138–52160, 2018.

[18] R. Setiono, B. Baesens, and C. Mues. Recursive neural network rule extraction for data with mixed attributes. *IEEE Transactions on Neural Networks*, 19(2): 299–307, 2008.

[19] M. H. Khalid, P. K. Tuszynski, J. Szlek, R. Jachowicz, and A. Mendyk. From black-box to transparent computational intelligence models: A pharmaceutical case study. In *2015 13th International Conference on Frontiers of Information Technology (FIT)*, pages 114–118, 2015.

[20] A. Gupta, S. Park, and S. M. Lam. Generalized analytic rule extraction for feedforward neural networks. *IEEE Transactions on Knowledge and Data Engineering*, 11(6):985–991, 1999.

[21] R. Setiono and H. Liu. A connectionist approach to generating oblique decision trees. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 29(3):440–444, 1999.

[22] A. Hara and Y. Hayashi. Ensemble neural network rule extraction using re-rx algorithm. In *The 2012 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6, 2012.

[23] R. Setiono and H. Liu. Symbolic representation of neural networks. *Computer*, 29 (3):71–77, 1996.

[24] M. W. Craven and J. W. Shavlik. Extracting tree-structured representations of trained networks. pages 24–30. MIT Press, 1995.

[25] Y. Hayashi. Application of a rule extraction algorithm family based on the re-rx algorithm to financial credit risk assessment from a pareto optimal perspective. *Operations Research Perspectives*, 3:32 – 42, 2016.

[26] S. Shen, S. X. Han, D. R. Aberle, A.A. Bui, and W. Hsu. An interpretable deep hierarchical semantic convolutional neural network for lung nodule malignancy classification. *Expert Systems with Applications*, 2019.

[27] X. Liu, X. Wang, and S. Matwin. Interpretable deep convolutional neural networks via meta-learning. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–9, 2018.

[28] S. Shirakawa and N. Fukumura. Extraction of easily interpretable representation using five-layered autoencoder. In *2017 International Conference on Advanced Informatics, Concepts, Theory, and Applications (ICAICTA)*, pages 1–4, 2017.

[29] T. Ueno and Q. Zhao. Interpretation of deep neural networks based on decision trees. In *2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*, pages 256–261, 2018.

[30] J. Kim and J. Canny. Interpretable learning for self-driving cars by visualizing causal attention. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2961–2969, 2017.

[31] M. Yeganejou and S. Dick. Classification via deep fuzzy c-means clustering. In *2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–6, 2018.

[32] B. Baesens, R. Setiono, C. Mues, and J. Vanthienen. Using neural network rule extraction and decision tables for credit-risk evaluation. *Management Science*, 49 (3):312–329, 2003.

[33] R. Setiono and H. Liu. Neurolinear: From neural networks to oblique decision rules. *Neurocomputing*, 17(1):1 – 24, 1997.

[34] G. Bologna and Y. Hayashi. A rule extraction study on a neural network trained by deep learning. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 668–675, 2016.

[35] S. Verma, J.P. Dickerson, and K. Hines. Counterfactual explanations for machine learning: A review. *CoRR*, 2020.

[36] R. Guidotti. Counterfactual explanations and how to find them: literature review and benchmarking. *Data Mining and Knowledge Discovery*, pages 1–55, 2022.

[37] Zhiwei Xiong, Zhan Shi, Huiqun Li, Lizhi Wang, Dong Liu, and Feng Wu. Hscnn: Cnn-based hyperspectral image recovery from spectrally undersampled projections. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 518–525, 2017.

[38] Jiqing Wu, Jonas Aeschbacher, and Radu Timofte. In defense of shallow learned spectral reconstruction from rgb images. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 471–479, 2017.

[39] Aitor Alvarez-Gila, Joost Weijer, and Estibaliz Garrote. Adversarial networks for spatial context-aware spectral image reconstruction from rgb. 09 2017.

[40] S. Koundinya, H. Sharma, M. Sharma, A. Upadhyay, R. Manekar, R. Mukhopadhyay, A. Karmakar, and S. Chaudhury. 2d-3d cnn based architectures for spectral reconstruction from rgb images. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 957–9577. IEEE Computer Society, 2018.