Discrimination of Signal-Background Events with Supervised and Weakly Supervised Learning in the Search for New Bosons Decaying to $Z + \gamma$ Final State

---

Nkateko Baloyi

Supervisor(s):
Bruce Mellado
Xifeng Ruan

A research report submitted in accordance with the requirements for the degree of Master of Science in the field of e-Science

in the

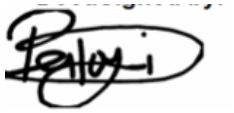
School of Computer Science and Applied Mathematics
University of the Witwatersrand, Johannesburg

14 July 2020

Declaration

I, Nkateko Baloyi, declare that the content in this research report is my own original work and It has not been submitted to any other university for a degree.

Part of this research have contributed and submitted to the High Energy Particle Physics (HEPP) 2019 and conference proceedings and the South African Institute of High Energy Physics (SAIP) 2019 conference proceedings to be published under the topics, **Application of Boosted Decision trees in the Search for a New Boson decaying to Z_γ Final State** and **Discrimination of Signal-Background Events with Supervised and Semi-supervised Learning in the Search for New Bosons Decaying to $Z + \gamma$ Final State**, respectively.



Nkateko Baloyi

14 July 2020

Abstract

The theory of Standard Model (SM) has successfully driven experiments and predictions since it was developed in 1975 and it has never been contradicted by experimental results. In 2012, the SM led to the discovery of Higgs boson (h) at the Large Hadron Collider (LHC) which completed the particle spectrum of the SM. The discovery of h inspired experimental studies to further understand the h scalar properties, opening the search for Beyond the Standard Model physics (BSM). BSM physics searches for new particles that can help understand and answer phenomena that cannot be explained by the SM. The LHC collides protons at high luminosity and high energy trying to recreate particles that occurred moments after the big bang and the BSM particles. The data produced during the collision requires advanced techniques that can search for relevant information in the data for signal (S) events to be identified. The production of the S events comes with a huge amount of background (B) production to which the S events cannot be easily identified. advanced machine learning (ML) and statistical techniques can be used to isolate the S events from the B events. ML is a subset of artificial intelligence that provides systems the ability to automatically learn and improve from experience. This research focuses on the application of boosted decision trees (BDT) and deep neural networks (DNN) on the Monte Carlo simulated data. Supervised learning and weakly-supervised learning (WSL) approaches are implemented to discriminate the S from the B events. The supervised learning is used as a benchmark to measure the performance of the WSL approach. Pre-selection cuts are applied on the data and four different models are applied to classify the S and B events for both BDT and DNN using the supervised learning and WSL approach. The WSL approach use two samples, one sample with only B events and the other sample mixed with S and B events to train the model. The DNN model is trained on the samples and applied to classify the S and B events on the same test data used in the supervised learning approach. The performance of the WSL models are compared to the supervised learning models performance. The results show a strong bias in the WSL approach.

Acknowledgements

I sincerely and gratefully acknowledge my supervisor Prof. Bruce Mellado and my co-supervisor Dr. Xifeng Ruan for their guidance, patience, constant support throughout this research. I am tremendously fortunate to have the opportunity to be mentored and learn from them with the depth of knowledge they have in the field. I am also truly grateful for the opportunity to travel to CERN to be associated and learn from experts in the field of Machine Learning from different parts of the world, which was made possible by my supervisor.
In addition, I gratefully acknowledge the e-Science programme for this amazing opportunity to enroll in the programme. I am truly grateful for the funding I received from DST-CSIR through the programme, and I thank SA-CERN for making it possible for me to travel to CERN.
Lastly, I would like to express my heartiest gratitude to my parents, Ntombhi Mazini and David Baloyi, for their support and prayers throughout my academic journey. Special thanks to my friends Risuna Nkolele, Zwidofhelangani Ndamulelo, Orapeleng Mogawana for their support, and my amazing team, Joshua Choma, Theodore Gaelejwe, Ntsoko Phuti and Kehinde Tomiwa for the role they played in me executing this research.

Contents

Declaration i

Abstract ii

Acknowledgements iii

List of Figures vi

List of Tables vii

1 Introduction 1
- 1.1 Background . . . 2
- 1.2 Problem Statement . . . 3
- 1.3 Research Question . . . 4
- 1.4 Research Aims . . . 4
- 1.5 Objectives . . . 4
- 1.6 Overview . . . 4
- 1.7 Outline . . . 5

2 Literature Review 6
- 2.1 Introduction . . . 6
  - 2.1.1 Supervised Learning . . . 6
  - 2.1.2 Unsupervised Learning . . . 7
  - 2.1.3 Semi-supervised Learning . . . 7
  - 2.1.4 Weakly Supervised Learning . . . 8
  - 2.1.5 Boosted Decision Trees (BDT) . . . 9
    - The Mathematics of Decision Trees and Gradient Boosting . . . 10
  - 2.1.6 Deep Neural Networks . . . 13
    - The Mathematics of Neural Networks . . . 13

2.1.7 Activation Functions . . . 16
2.1.8 Classifier Performance . . . 17
2.2 Related Studies . . . 19

3 Research Methodology **22**
3.1 Data . . . 22
3.2 Methods . . . 24
3.2.1 Supervised Learning . . . 24
3.2.2 Weakly Supervised Learning . . . 24
3.2.3 Application . . . 25
Application of Boosted Decision Trees . . . 25
Application of Deep Neural Network . . . 26
3.2.4 Performance Metrics . . . 27
3.3 Summary . . . 27

4 Results and Discussion **28**
4.1 Boosted Decision Trees Results . . . 28
4.1.1 Supervised Learning . . . 28
4.1.2 Weakly Supervised Learning . . . 31
4.2 Deep Neural Networks Results . . . 34
4.2.1 Supervised Learning . . . 34
4.2.2 Weakly Supervised Learning . . . 36
4.3 Discussion . . . 39

5 Conclusions and Future Work **41**
5.1 Conclusions . . . 41
5.2 Future Work . . . 41

A **42**
A.1 1D Distributions . . . 42

Bibliography **44**

List of Figures

2.1 Illustration of a Decision Tree [34] 11
2.2 Illustration of a shallow Neural Network (Left) [41] and Deep Neural Network (Right) [42] 14
2.3 Plot of the ReLu Activation function (Left) and the Leaky ReLu activation function (Right) 17
2.4 Plot of the Sigmoid function 17

4.1 ROC curves illustrating the performance of the the BDT supervised learning model. 29
4.2 *S* and *B* Output distributions 30
4.3 ROC curves illustrating the performance of the the BDT WSL model. 32
4.4 *S* and *B* Output distributions 33
4.5 ROC curves illustrating the performance of the DNN supervised learning model. 35
4.6 *S* and *B* Output distributions 36
4.7 ROC curves illustrating the performance of the DNN WSL model. . . 37
4.8 *S* and *B* Output distributions 38

A.1 1 Dimensional distributions of the input variables 43

List of Tables

2.1 Confusion Matrix 18

3.1 Input variables description 23
3.2 BDT Hyper-parameters 26
3.3 DNN Hyper-parameters 26

4.1 BDT supervised learning Table of Results 29
4.2 BDT WSL Table of Results 31
4.3 DNN supervised learning Table of Results 34
4.4 DNN WSL Table of Results 37

Chapter 1

Introduction

The Large Hadron Collider (LHC) is the largest and most powerful particle accelerator in the world. It was built between 1998 and 2008 by the European Organization for Nuclear Research for the purpose of testing predictions of different theories in high energy physics [1, 2]. The LHC runs under the Swiss-French border near Geneva in a circular tunnel of 27 km circumference. The accelerator consists of two proton beams that travel in opposite directions at near the speed of light with very high energy before colliding [2, 3]. In 2010, the LHC was successfully commisioned for proton-proton (pp) collision with 7 tera electron Volts (TeV) centre of mass energy [4].

The phenomena that physicists are looking for in the collisions have a very low probability of occurrence. High luminosity is required to produce more data events in order to increase the occurrence of the phenomena. Luminosity gives the measure of the number of potential collisions that occur in a given period of time, an important indicator of the performance of the accelerator [5]. An integral of the luminosity over time known as the integrated luminosity gives a measure of the collected data size measured in inverse femtobarn (fb^{-1}). The value of $1\ fb^{-1}$ equates to 100 million collisions [4, 5].

During the pp collision in the LHC, the protons can either collide violently and produce any new particles or they can glance off each other. The probability of production of any of the particles is different, each particle have a different cross-section. The rare particles such as the Higgs bosons have a very small cross-section [6]. The large luminosity accumulated by the LHC opens the possibility of a new range of Beyond the Standard Model (BSM) physics discoveries. The data collected

in the LHC during the collisions is complex, high dimensional, highly imbalanced with a strong overlap of the classes, which requires more advanced techniques with the potential to extract information produced in the pp collision. The information that can be obtained from the collisions includes, but not limited to the half-lives of particles, the velocity, decay rate of the particles, transverse momentum. Statistical tools and advanced techniques, such as machine learning (ML) can be used to improve data processing, classi cation and regression tasks in particle physics data and directly improves the probability of discovering new particles. ML is developed with the concept of allowing computers to learn by themselves and improve ef ciency with experience and is applied in various professional elds and industries to improve the processing and the quality of results by nding patterns in data that lead to accurate decision making [7, 8, 9].

Various ML algorithms have been applied in binary classi cation tasks and have achieved accuracy above the random guess threshold. Boosting techniques and deep learning techniques have gained popularity in classi cation and regression tasks due to their ability to improve accuracy when applied on weak learning algorithms and trained on huge amount of data, respectively. Boosted decision trees and deep neural networks have been applied widely in different professions, including high energy physics data, and they have proven to be superior in terms of accuracy [10, 11].
This study aims to highlights machine learning approaches that can be used to discriminate signal and background events on simulated data and help develop a model for real data. We develop two machine learning models following the supervised learning and the weakly supervised learning approach. The background, research aims and objectives of this study are described in this chapter.

1.1 Background

The standard model (SM) of physics is a theory that describes three of the four fundamental forces, the electromagnetic, weak, and strong interactions, and how these forces and elementary particles that makes up matter as we know it, relate to each other. The SM has successfully driven experiments and predictions since it was introduced in 1975 [12, 13]. This theory is a well-tested method and it has never

been contradicted by experimental results. In 2012, the theory led to the discovery of the Higgs boson (h) [14], which completed the particle spectrum of the SM. The particle h is a very important particle in the SM as it signals the existence of the Higgs eld, an invisible energy eld that is accompanied by the h, and generates mass through its interaction with other particles. The discovery of h inspired many theoretical and experimental studies conducted to understand the scalar properties of this particle and how new physics can be identi ed.

Although the SM has successfully explained experiments and was validated by the discovery of h, it still fails to explain some phenomena [15]. The discovery of h ended the era of theories driving the experimental results. This means that experiments must be conducted in search of new physics Beyond the SM (BSM), to help understand and explain some phenomena that cannot be explained by the SM. The LHC data has been made public to learn more about the data and analysis studies have been done on this data [16]. The studies conducted show that there are excesses observed in the LHC data release, which suggests the possibility of new physics [17, 18]. These excesses have been observed in both the ATLAS and CMS experiments and a number of nal states resulting in these excesses has been identi- ed. The Wits Institute for collider Particle Physics and their collaborators proposed a hypothesis called the Madala hypothesis, an extension of the SM, which searches for BSM physics.

1.2 Problem Statement

BSM physics requires particles to be accelerated at extremely high energy and high luminosity to increase the probability of the BSM events to be created. During the collision, a large amount of background is created with a small amount of signal. For new particles to be discovered, the signal and background events must be isolated. This requires statistical tests and advanced techniques that can extract the small signal within the large background. Machine learning techniques can be developed to discriminate the signal and background events, and improve the purity of signal in the search for new physics.

1.3 Research Question

From highly imbalanced dataset, can we identify and discriminate signal from the background events with supervised learning, to help improve the purity of the signal in order to perform the search for BSM physics?

When working with mixed unlabeled dataset, can we develop a model from data using the weakly supervised learning approach that can isolate signal from background?

1.4 Research Aims

This study aims to develop machine learning models that can discriminate signal and background events in the search for new bosons decaying to Z + g nal state.

1.5 Objectives

The primary goal is to develop a supervised learning model trained on labeled data to distinguish signal and background events. The secondary goal is to develop a weakly supervised learning model that can be employed on unlabeled data by assigning weak-labels on the unlabeled data. The weakly supervised approach implemented on weak-labeled data is used to distinguish the same test data used in the supervised learning approach to compare the output in order to help develop a model that can be implemented on real data.

1.6 Overview

Studies show that excesses have been observed in the LHC Run1 and Run2 data release, which suggest that BSM physics is possible. This inspired the search for new BSM physics that can explain and help understand life moments after the big bang. The search for BSM physics requires particles to be accelerated at extreme high energy and luminosity to increase the BSM cross-section. This ends up with large background and small signal events produced. In order to search for new

physics, the signal must be isolated from the background to check for unknown signal. With the vast amount of data generated by the LHC during collisions, advanced machine learning and statistical techniques are the key to isolating signal from background to improve the quality of signal used in detecting possible BSM physics. In this study, we aim to develop supervised learning and weakly supervised learning models using simulated Monte Carlo data, that can be implemented on real data.

1.7 Outline

The remaining chapters of this research report are designed as follows. Chapter 2 describes the concepts of machine learning and the performance metrics. Boosted decision trees and deep neural networks are described in detail. The related study is presented in the form of literature review. Chapter 3 describes the data and outlines the research methodology. The data preparation, application of boosted decision trees and deep neural networks are described. The results and discussion of the boosted decision trees and deep neural networks ndings are presented in Chapter 4. The conclusions and possible future research are presented in Chapter 5.

Chapter 2

Literature Review

2.1 Introduction

Machine Learning (ML) is the umbrella term for a set of statistical and computational techniques for interpreting data which evolves from computational learning and the study of pattern recognition. ML interprets data and make predictions by exploring algorithms that learn and extract patterns in data such as various regression methods, classi cation methods, etc., and improve automatically through experience [19]. This chapter focus on the binary classi cation tasks using two ML approaches, supervised and weakly supervised learning. The literature of these two ML approaches, the ML algorithms implemented and the performance metrics that are used in this study are de ned in this chapter.

2.1.1 Supervised Learning

Supervised learning is a type of ML that learns from labeled data in order to make predictions on new unseen data. The supervised learning algorithm learns a function based on the input-output pairs to map the input vector to the output. The training sample consist of labeled data with instance x and a corresponding label y, that is, $f(x_i, y_i)g_{i=1}^{n}$. For a given set of training data, $f(x_i, y_i)g$ with x_i representing a feature vector and y_i representing the label of training example i, the algorithm learns a function $f : c \; ! \; Y$, where the goal of the function $f(x)$ is to predict the correct label y when given test data x such that c is the domain of instances and Y is the domain of labels [20, 21].

There are two types of Supervised learning, classi cation and regression. Classi- cation is a technique of identifying to which category does a set of observations belong to with discrete y labels. Regression techniques are used to predict a target continuous value Y based on the relationship of variables in the data. This study focus on the classi cation techniques since the goal of the study is to discriminate instances from two classes.

2.1.2 Unsupervised Learning

Unsupervised learning is a type of ML that trains a machine to estimate a model that can nd unknown patterns in data that is not labeled, classi ed or categorized. Unsupervised learning is divided into the parametric unsupervised learning and the non-parametric supervised learning. The parametric unsupervised learning assumes that the data points are independently and identically drawn from a common distribution, based on a xed set of parameters. For a given set of input examples $X = x_1, x_2, ..., x_n$, the machine estimate a model representing the probability distributions for a new input instance x_i, such that $P(x_i | x_1, ..., x_{i-1})$. Non-parametric unsupervised learning groups the data into clusters based on shared attributes. In clustering, the common attributes in the data are identi ed and the data is then grouped based on the similarities and dissimilarities of the instances [22, 23].

2.1.3 Semi-supervised Learning

Semi-supervised learning is a class of ML techniques that make use of unlabeled data and labeled data for training. Semi-supervised learning use unlabeled data with a small proportion of labeled instances to model a classi er that takes an advantage of using labeled and unlabeled instances. Labeled data is dif cult to obtain since it requires experts to draw conclusion on the labels of the instances. Most often data collected is unlabeled and advanced techniques such as unsupervised learning techniques are used to label the data. Semi-supervised learning used to label unlabeled instances use self training which employs a large amount of unlabeled data with a proportion of labeled data to build a classi cation model that can label unlabeled instances. The model is built on the labeled instances, and then

tries to add labels to the unlabeled instances. For a given set of n examples containing independently identically distributions, $(x_1...x_n)ec$ with the corresponding labels $(y_1...y_n)eY$ and another given set of unlabeled examples $x_{n+1}...x_{n+u}$, the semi-supervised learning make use of the combined given data to build a classi er that could be obtained by following the supervised learning approach where the unlabeled data is discarded or make use of the unsupervised learning by discarding the labeled data. Assuming that the data comes from a Gaussian mixture model in Equation2.1

$$f(xjq) = \sum_{k=1}^{n} a_k f(xjq_k), \tag{2.1}$$

given that a_k is de ned as the mixture coef cient, $\sum_{k=1}^{n} a_i = 1$ and $q = q_i$ are the parameters. Let the target label y_i be a random variable with distribution $P(y_i jx_i, z_i)$, where z_i is a mixture variable that determines the random label distribution with the feature vector x_i. The maximum posterior gives:

$$g(x) = \underset{Ee(Y,N)}{\operatorname{argmax}} \sum_{k=1}^{n} P(y_i = E)jz_i = k, x_i)P(z_i = k)jx_i, \tag{2.2}$$

and

$$P(z_i kjx_i) = \frac{a_i f(x_i jq_k)}{\sum_{j=1}^{n} a_j f(x_i jq_j)}. \tag{2.3}$$

The goal is to estimate $P(y_i = Ejz_i = k, x_i)$ and $P(z_i = kjx_i)$ from the training data [24, 25, 26].

2.1.4 Weakly Supervised Learning

Weakly supervised learning is a term that covers the processes applied to build a model by learning with weak supervision. Weak supervision uses labeled data, but the labels are not quite correct. There might be noise in the labeling process of the data, or it might be that the labels represents which of the collection of classes a given observation belongs to. The goal of weak supervision is the same as the supervised learning approach, to develop a model that accurately outputs the labels, however it is not trained on correctly labeled training data. For a given unlabeled

data $X = (x_1, ..., x_n)$ there are weak labeled training instances from the weak supervision $P_i(y|x), i = 1$, such that each instance has a coverage set C_i which represents the set points x. The accuracy of the model is de ned as the probability of the correct target label y over C_i, assumed to be < 1. Weak supervision is divided into three classes, the incomplete supervision, inexact supervision and the inaccurate supervision. The incomplete supervision is derived from the semi-supervised or active learning where only a proportion of instances in the training data are labeled. This method uses the active learning to label the unlabeled instances assuming that the cost of the labeling of these instances depends only by the number of queries. In the inaccurate supervision, the training data consists of instances with correct and incorrect labels. This method employs several learners to identify the unlabeled instances and examine with the training data in order to make corrections. In the inexact supervision, the training data is assigned labels but the labels are not as exact as desired. In this method, the training data consists of labeled bags which contains unlabeled instances. Each bag consists of the collection of unlabeled instances where a bag is negatively labeled if all the instances in the bag are negative. The task is to learn $F : c \; ! \; Y$ from a given training data $f(x_1, y_1), ..., (x_n, y_n)g$. The bag is de ned as $X_i = f x_{i1}, ..., x_{i,n_i} \; cg$ where n_i is the number of instances in the bag X_i and $y_i eY = f Y, Ng$, and each instance is de ned as $x_{ik} \; c(kef 1, ..., n_i g)$. A bag X_i is positively labeled if there exist x_{iP} which is positive and $Pef 1, ..., n_i g$ is unknown. The goal of this method is to correctly predict the labels of the new unseen bags [27, 28, 26].

2.1.5 Boosted Decision Trees (BDT)

Decision tree is data mining tree like structure model that performs a top-down split greedy approach [29]. The model structure consists of a root node, branch and leaf node where each internal node represents the input attribute, the branch represents the outcome and the leaf node represents the target output. The root node is at the top and instances are sorted down from the root node to the leaf node, in which splitting is done on the entire training set. Decision trees are very popular and they are commonly used in data mining to derive strategy for decision making making, and they have gained popularity and have been widely used in ML to predict target output. Although decision trees are very powerful, they are prone to bias

and variance. Studies have been done to simplify decision trees and the techniques developed by J.Quinlan (1999) [30] show that decision trees can be simpli ed while still retaining their accuracy. Ensemble techniques have been implemented on decision trees to improve the prediction of decision trees by constructing more than one decision tree to build a better model. The principle behind ensemble methods is to combine several weak learners in order to form one strong optimal model with improved prediction performance. Bagging and boosting are some of the popular ensemble techniques implemented to improve the performance of weak models by lowering error rates and prediction errors in classi cation and regression tasks, respectively. Bagging and boosting in decision trees are used to reduce variance of the model and reduce error from the prior tree, respectively. In bagging, each model is trained on samples independently with replacement from the original training dataset and the target output is the weighted sum of the predictions made by all the models. Boosting trains the initial model on the training set with replacement from the training dataset while passing through all the training patterns in order to note patterns that gives the least errors [31, 32].

The Mathematics of Decision Trees and Gradient Boosting

Figure 2.1 is an illustration of a decision tree. Each node represents the attributes (features), each branch and leaf represents the decision and the outcome, respectively. To build a tree, the attribute with the highest information gain (IG) is chosen. The entropy (H), known as the information theory, is calculated rst according to Equation 2.4

$$H(s) = \sum_{c2C} p(c)\log_2 p(c), \tag{2.4}$$

where $H(s)$ is the entropy (H) in a set, measuring the amount of uncertainty in the set, s is the set for which H is calculated, C is the set of classes in s, and $p(c)$ is the proportion of the number of entries in class C to the entries in set s. The IG is then calculated according to Equation 2.5

$$IG(A,s) = H(s) \sum_{t2T} p(t)H(t), \tag{2.5}$$

where A is an attribute, $IG(A)$ is the measure of a decrease in entropy H after splitting s into A, T is a subset created when s was split to A, $p(t)$ is the proportion of

the total entries in subset t to the entries in s and $H(t)$ is the entropy H of subset t [33, 34].

FIGURE 2.1: Illustration of a Decision Tree [34]

In gradient boosting, weak learning models are combined together to build one complex model [35, 36]. Consider the input training data $(x_i, y_i)_{i=1}^{n}$, loss function $L(y, F(x))$ and iterations m.
An initial model consist of a function $F_0(x)$ de ned with a constant value where the function is de ned according to Equation 2.6:

$$F_0(x) = \operatorname{argmin}_g \sum_{i=1}^{n} L(y_i, g) \tag{2.6}$$

$$F_m(x) = F_{m-1}(x) + \operatorname{argmin}_{h_m \in H} \left[\sum_{i=1}^{n} L(y_i, F_{m-1}(x_i + h_m(x_i)) \right]. \tag{2.7}$$

The model predicts y by minimizing the mean squared error.

(2.8)

The differential of Equation 2.6 with respect to g, minimizes to Equation 2.9

$$F_0(x) = \frac{\sum_{i=1}^{n} y_i}{n}, \tag{2.9}$$

where i is the indexes of the training set. The pseudo-residual of each error is calculated according to Equation 4

$$r_{im} = \left[\frac{dL(y_i, F(x_i))}{dF(x_i)}\right]_{F(x)=F_{m-1}(x)}. \tag{2.10}$$

From the given dataset, new data is derived and modi ed as de ned in Equation 2.11

$$D = (x_i, r_{im}). \tag{2.11}$$

A base learner $h(x)$ which represent the direction in which the loss function is minimum with respect to F_{m-1} is trained and tted on the modi ed dataset. $F_m(x)$ is then calculated according to Equation 2.12:

$$F_m(x) = F_{m-1}(x) + \gamma h_m(x), \tag{2.12}$$

where $h_(x)$ is calculated using Equation 2.13

$$h_x = F_{m+1} \quad F_{m(x)}, \tag{2.13}$$

where F_{m+1} attempts to correct the errors made by the previous model. The base learner $h_m(x)$ is then tted on the pseudo-residuals r_m. Weights W are moved in the direction of decreasing loss function. The learning rate hyper-parameter controls how much weights moves. The weight W are updated according to Equation 2.14

$$W \quad W \quad \alpha\frac{dL}{dW}, \tag{2.14}$$

optimum γ found by solving Equation 2.5

$$\gamma_m = \arg\min_\gamma \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i)). \tag{2.15}$$

When the optimum γ_m is solved, Equation 2.12 is updated.

2.1.6 Deep Neural Networks

Neural networks are a set of algorithms inspired by the biological neural networks that recognise patterns in data and improves automatically through experience. Neural networks consists of the input layer, hidden layer with nodes and an output layer. The goal of the neural network is to take an input and learn a function that can map the input to the output [37]. The nodes are linked to each other by a connection represented by weights, where each node has an activation function. In a feed-forward network, the patterns of the input information are fed into the input layer which the neural network learns, triggering the hidden layers which apply different transformations to the information received from the previous layer, allowing the neural network to learn more complicated patterns. The hidden layer send the information extracted from the patterns to the output layer [38].

Neural network learns by a process of back propagation, where the goal of the back propagation is to optimize the weights by computing the gradient descent with regard to the weights and biases. A loss function is used to measure the difference between the neural network output to the desired output and to estimate the loss. Once the loss is estimated, the neural network propagates the loss information back-wards, from the output layer, through the nodes in the hidden layers, to the input layer, causing the neural network to adjust weights. The loss function is minimized by using the gradient descent technique which allows the weights to be updated by calculating the gradient of the loss function in order to determine the direction of the global minimum [39, 40].

The Mathematics of Neural Networks

Figure 2.2 on the Left illustrate the structure of a neural network with dimension of 4 input, a hidden layer and 2 output layers. The input x_i is connected to the output node by weights w_i. Each node receives a vector (x) which contains the feature values and compute a corresponding output target (y). Each node consists of a set of parameters, weights (w) and a bias (b). The node calculates the weighted average of the vector x and add b such that:

$$z = \sum_{i=1}^{m} w_i x_i + b, \tag{2.16}$$

where m is the total number of inputs. The input x and w represented as vectors gives the dot product of the two product shown in Equation 2.17 below:

$$z = w^T.x + b. \tag{2.17}$$

The output y is passed through an activation function g such that:

$$y = g(z). \tag{2.18}$$

FIGURE 2.2: Illustration of a shallow Neural Network (Left) [41] and Deep Neural Network (Right) [42]

The advancement of computers and technology improved the computation, the amount of data generated and data collection processes and advanced neural networks algorithms. These advancements allowed the improvement of neural networks to run high dimensional parameter and deep neural, known as deep neural network (DNN). Figure 2.2 on the Right shows an architecture of DNN. DNNs are different from shallow neural networks since they have more than one hidden layer, whereas shallow neural networks have only one. When writing the notation of a selected layer in DNN, vector a, which is the activation function corresponding to the selected layer, is used. The layers in the DNN performs calculations following Equation 2.19:

$$z_i^l = w_i^T.a^{l-1} + b_i, \tag{2.19}$$

and the activation is calculated according to Equation 2.20:

$$a_i^l = g^l(z_i^l), \tag{2.20}$$

where i represent the index of the node in a layer and l represent the layer. The b of each node in a layer and vectors of w are independently stacked together to form a vertical vector b and matrix W, respectively. The matrix W^l is represented as:

$$\begin{bmatrix} w_1^{l\,T} \\ w_2^{l\,T} \\ w_3^{l\,T} \\ \vdots \\ w_{n\ 1}^{l\,T} \end{bmatrix}$$

and the vertical vector b^l is represented as follows:

$$\begin{bmatrix} b_1^l \\ b_2^l \\ b_3^l \\ \vdots \\ b_{n\ 1}^l \end{bmatrix}$$

When considering multiple entries instead of working with one layer as we have from the equations derived above, vectorization is applied across the multiple entries. Given m entries and j features, the vertical vectors x,a and z create X,A and Z, respectively. Equation 2.19 and Equation 2.20 becomes:

$$z = W^l.A^{l\ 1} + b^l, \tag{2.21}$$

and

$$A^l = g^l(z^l). \tag{2.22}$$

Different activation functions can be used for each layer, which introduce non-linearity in the output of the node. The W and b parameters are adjusted using the Equation 2.23 and Equation 2.24 in the back propagation:

$$W^l = W^l \quad a\frac{dL}{dW^l}, \tag{2.23}$$

and

$$b^l = b^l \quad a\frac{dL}{db^l}, \tag{2.24}$$

where dW and db are the partial derivatives of W and b, and a is the learning rate. The learning rate a is a hyper-parameter which allows control of the adjustments. Setting a is important since it controls how your neural network model learns. If the a is set too low, the model learns too slow and if set too high, the model might miss global minimum [41, 43].

2.1.7 Activation Functions

Activation functions determines whether a node should be activated or not activated given an input or a set of inputs. They introduce non-linear properties in neural network, allowing the network to learn and "represents non-linear complex arbitrary functional mappings between inputs and outputs". There are several activation functions used in neural networks, recti ed linear units (ReLu) and sigmoid are some of the popular and commonly used activation functions. ReLu is the most commonly used activation function in neural networks that gained its popularity in the past few years, almost all deep learning applications use ReLu. ReLu is linear for all positive values and zero for all the negative values.

$$\text{ReLu} : f(x) = \begin{cases} 0, & \text{for } x < 0 \\ 1, & \text{for } x \quad 0 \end{cases}$$

This means that ReLu does not saturate when x gets larger [44], it allows for quick training of neural networks, and it avoids and recti es the vanishing gradient problem. X.Gloret al [45], demonstrated that ReLu allows for better training of deeper neural networks when compared with the other activation functions. Although ReLu is commonly used in all deep learning applications and has improved the training of deep learning, it has some drawbacks. One of its major drawbacks is a dying ReLu problem which causes several nodes to die and remain inactive [45, 46]. This problem can be solved with the use of Leaky ReLu which introduces a small slope to the gradient of ReLu. Figure 2.3 is an illustration of the ReLu and Leaky ReLu activation functions .

FIGURE 2.3: Plot of the ReLu Activation function (Left) and the Leaky ReLu activation function (Right)

Another popular and commonly used activation function is the sigmoid function, de ned in Equation 2.25.

$$f(x) = \frac{1}{1 + e^{x}}. \tag{2.25}$$

Sigmoid is characterised by a 'S' shaped curve, ranging between 0 and 1. Sigmoid functions have a problem of vanishing gradient which prevents the weights from changing its value, results in back-propagation errors [47].

FIGURE 2.4: Plot of the Sigmoid function

2.1.8 Classi er Performance

In ML, the performance measurement of a model is an essential task. In classi cation tasks, the classi er produce a discrete class label indicating the predicted class of the instance. To measure the performance of the classi er, classi cation metrics are used. The area under the curve (AUC) and receiver operating characteristics

(ROC) curves analysis are one of the most important evaluation metrics to measure the performance of the classi er. The AUC and ROC analysis have been adopted in different elds of study as a standard evaluation metrics to measure how well a model is capable of distinguishing classes. To distinguish between the actual class label and predicted class label, the labels f Y, N g are used for the classi er's predicted class label [48]. In a binary classi cation task, for each instance in a given classi er, there are four possible outcomes:

True Positive (TP) = Number of positive instances correctly classi ed as positive.
False Positive (FP) = Number of negative instances incorrectly classi ed as positive.
True Negatives (TN) = Number of negative instances correctly classi ed as negative.
False Negative (FN) = Number of positive instances incorrectly classi ed as positive.

Table 2.1 represents the confusion matrix. The major diagonal (TP and TN) represents the number of correctly classi ed instances and the second diagonal (FP and FN) represent the number of instances that are incorrectly classi ed.

TABLE 2.1: Confusion Matrix

	Positive (1)	Negative (0)
Positive (1)	TP	FP
Negative (0)	FN	TN

The true positive rate (TPR) and the false positive rate (FPR) are de ned in Equation 2.26 and Equation 2.27, respectively, where tp represents the number of correctly classi ed positives, P represents the total number of positive instances, fn represents the number of incorrectly classi ed negatives and N represents the total number of negative instances.

$$TPR \quad \frac{tp}{P}, \tag{2.26}$$

$$FPR \quad \frac{fn}{N}. \tag{2.27}$$

The ROC curve is a plot of the TPR against the FPR at different classification threshold. The AUC measure the area under the ROC curve to provide the overall measure of performance ranging from 0 to 1. An excellent performing classifier has AUC closer to 1 indicating that the classifier has a good measure of separability. A poor performing classifier has AUC closer to 0, indicating that the classifier has the worst measure of separability. The higher the AUC, the better the model is at predicting instances of different classes.

Other common evaluation metrics are defined in Equation 2.28 - Equation 2.30. Precision and recall are also important evaluation metrics in machine learning. Precision indicate the ability of the classifier to predict relevant instances, that is, how well the classifier can label negative instances as negatives and not as positives. Recall indicates how well the classifier can correctly classify all the positive instances by providing a percentage of total positive results correctly classified. Accuracy provides the percentage of the correctly classified instances [49]. Accuracy is not a reliable metric for the real performance of a classifier, however, it is commonly used with other evaluation metrics.

$$precision = \frac{TP}{TP + FP}, \tag{2.28}$$

$$accuracy = \frac{TP + TN}{P + N}, \tag{2.29}$$

$$recall = \frac{TP}{P}. \tag{2.30}$$

2.2 Related Studies

Discrimination of signal and background instances produced during the PP collision in the LHC is a very crucial step in searching for new signal instances produced during the PP collision without loosing or missing the signal instances [10, 50]. Studies have been conducted using modern techniques that can discriminate these two types of instances with maximum accuracy and also out perform the traditional physics methods.

High energy physics data is often imbalanced with the class of interest being the minority class. H.Han et al. [51] compares different techniques that can be used to handle highly imbalanced dataset and highlights metrics that can be a measure of a classi er performance when working with imbalanced dataset. There are different techniques used to correct imbalance dataset, synthetic minority Over-sampling technique (SMOTE) is one of the methods used to over-sample the minority class. In this study, SMOTE is compared with other over-sampling techniques. A confusion matrix is used to measure the performance of the classi er, and the results obtained show that the SMOTE approach achieves better results.

BDT and ANN have been widely used in high-energy physics. A comparison study of this two techniques applied on particle identi cation using Monte Carlo samples is discussed in this paper [11]. The results obtained show that both BDT and ANN techniques in this task are very competitive. The BDT performance improved as number of variables (features) are increased, and proved to be more ef cient and superior than the ANN. The results also show that BDT also improves performance when the number of trees are increased.

L.Dery et al. [52] introduces a new machine paradigm for classi cation task called weakly supervised classi cation that can be trained on incomplete and/or unlabeled data and still match the performance of a fully supervised learning technique. The classi er is trained on unlabeled data using an approach called learning with label proportions (LLP). Data is split into bags containing unlabeled instances. "For a binary classi cation task with classes, A and B, the training set contains at least one instance of A classi er". The goal is to optimize a model that can at least identify one instance of A on unseen bag, given only a fraction of class A in a given bag.

E.Metodiev et al. [53] introduces another type of weak learning paradigm that can also be used in binary classi cation, called classi cation without labels (CWoLa) approach. This approach allows a classi er to be trained on real data set, which implies that this can be a better approach when dealing with the LHC data since the data is not labeled and mixed. The classi er is trained on two mixed samples, $M1$ and $M2$, consisting of both signal and background instances in each sample with

different proportions. Using full supervision, the classi er is trained to discriminate $M1$ from $M2$ without providing the classi er with the proportions of each sample. The same classi er used to discriminate $M1$ from $M2$, is then applied to discriminate signal instances from background instances without labels. This approach is found to match the performance of full supervision approach.

Chapter 3

Research Methodology

3.1 Data

This study employs the Monte Carlo dataset implementing the LHC data recorded from 2015-2017 in the pp collision at $\sqrt{s}$ = 13 TeV generated using PYTHIA8 [54, 55, 56] . One sample follows the 2015 and 2016 LHC data normalized to luminosity of 36.1 fb ^{1}, and the other sample follows the 2017 LHC data normalized to luminosity of 43.8 fb ^{1}. The signal data consist of Z boson produced from gluon-gluon fusion sample, real missing transverse energy (E_T^{miss}) from neutrinos produced by Z decay, and the heavy pseudo-scalar particle decaying to Z boson and heavy scalar sample. Missing energy is the energy that is expected to be detected during particle collision but is not detected on the particle detector since it is carried by particles that do not interact with strong forces. The background data is a mixture of the tt g and g Jet background samples and is produced . The background contains fake E_T^{miss}. The dataset for the supervised approach consist of labeled signal (S) and background (B)events with 11730 S events and 151534 B events decaying to the Z + g nal state. The second dataset for the weakly supervised learning approach used in this study consists of two different samples of unlabeled data. The rst sample is the Monte Carlo data collected from 2015 to 2017 in the pp collision that consists of the Z + g events with the invariant mass outside 240-280 GeV. The second sample consist of different statistics of the Z + g events of the rst sample and the BSM monte carlo data. The rst sample is made up of only B events, while the second sample consist of mixed S and B events. Pre-selection cuts are applied on the data for both the supervised and weakly supervised approach into three categories with respect to the missing transverse energy signi cance ($S_{E_T^{miss}}$). The cuts are applied based on

the following:

low (Low) category : 2.5 $S_{E_T^{miss}}$ 3.5 $\sqrt{GeV}$

The intermediate (Int) category : 3.5 $S_{E_T^{miss}}$ 5.5 $\sqrt{GeV}$

The high (High) category : $S_{E_T^{miss}}$ 5.5 $\sqrt{GeV}$.

The pre-selection cuts are applied to maximize S and minimize B, where the cut points are scanned regions of S to B efficiency. The training data consists of 114284 events with only 8211 S events and 106073 B, and the test data contains 48980 events with 3515 S events and 45461 B, before applying the pre-selection cuts.The Low category consists of 1336 S and 7875 B training events, and 529 S and 3369 B test events. The Int category consists of 3486 S and 3996 B events in the training data and only 1744 B and 1524 S events in the test data. The High category consist of 2017 S and 1393 B events in the training data and 874 S and 588 B in the test data. Table 3.1 below is the description of the features used in this study. The features used include two leptons, a photon and multiple jets.

TABLE 3.1: Input variables description

Variable	Description
mu	Average bunch crossing
dphifjmet	The angular distance D(f) between forward jets and MET
dphisjmet	The angular distance D(f) between soft jets and MET
ssumpt2	Subleading vertex sum P_t squared
djpt	Scalar difference between the vectorial sum P_t of all the jets and leading vectorial sum P_t
dsumpt2	Difference between leading and subleading vertex sum P_t squared
dphirefjetmet	The angular distance D(f) between vectorial sum P_t of all jets and MET

The 1-dimensional distributions of these input variables are provided in Appendix A. The distributions for the S and B have been normalised in order for the distributions to be comparable. All the variables except for the metsig variable , E_{miss}^T, strongly overlap. This demonstrate that the B and S events share similar characteristics that cannot easily be distinguished .

3.2 Methods

This section describes the methods applied in this study. The method is divided into two sections, the supervised learning and the weakly supervised learning approach.

3.2.1 Supervised Learning

The ROOT data format is converted into a python readable format using uproot I/O library, primarily intended to stream data into ML libraries in python. The data is split into 70:30 train and test for each category before pre-processing. The MinMax scaler is applied to transform the feature events by scaling to a range of 0-1. Min-max scaler is used instead of standard scaler since we are not concerned with the standardisation along the variance axes. The data described in Section 3.1 shows that the data highly imbalanced, with 1:13 B to S ratio. The training data for each category is split into 80 % training and 20 % validation before oversampling the minority class. Mixmax scaler is then used to scale the data points between 0 and 1 on the training data, and is then tted on the validation and test data independently. Random oversampling technique is applied to improve the S to B ratio by supplementing multiple copies of some of the S events. Oversampling is only applied on the training data of each category after splitting the train, validation and test data to avoid having multiple copies of events instances of the training data also in the validation and test data.

3.2.2 Weakly Supervised Learning

The weakly supervised learning method employed in this study is a new method derived from the inaccurate weak supervision. This approach uses the unlabeled data samples of only background and the mixed sample. The two samples are assigned weak labels, where the background only events in sample 1 (M 1) are labeled 0 and the mixed S and B events in sample 2 (M 2) are labeled 1 . The samples are split into 70:30 training and testing data and a model is trained to distinguish events from M 1 and M 2. The same model trained to distinguish M 1 and M 2 events, is used to discriminate the S and B events in the test data used in the supervised learning approach. E.Metodiev et al. [53] shows that a weakly learning model trained

to distinguish events from different samples can be employed to discriminate the S and B and still performs just as well as the supervised learning model. This study takes a different approach by using the same model trained to distinguish events from M1 and M2, to discriminate S and B from the supervised learning approach in order to measure the performance of the weakly supervised learning with the supervised learning results to help develop a model for real data. The performance of the model is measured by comparing the ROC curves of the supervised learning categories and the output distributions.

3.2.3 Application

Application of Boosted Decision Trees

The K-fold cross validation is used to first find the optimal maximum depth. The optimal maximum depth is determined by evaluating the boosted decision tree on a held-out data set using the 10-fold cross validation for the maximum depth range of 1 to 25. The data is re-sampled, randomly split into 10 folds and fit the trees using K-1 folds and validate on the remaining folds. The scores are noted and the process is repeated until all the K-folds serves as the validation set. The classification accuracy is measured on the validation folds. The optimal maximum depth is chosen based on the best bias-variance trade-off. The optimal maximum depth is then fixed, and a GridSearchCV hyper-parameter tuning is used to apply all the possible combinations of parameters provided in the grid through a list of dictionaries to find the optimal parameters to build a BDT model. The BDT is trained for each value in the grid and the performance score of each iteration is provided. The optimal parameter selected is the one that gave the best score. The validation data is used to evaluate the performance of the model while fine tuning the parameters to ensure there is little overtraining observed from the training and validation accuracy before applying it to the test data. The parameters used to build a BDT model for each category are defined in Table 3.2.

TABLE 3.2: BDT Hyper-parameters

Hyper-parameters	Low Category	Int category	High Category	No-cut
NTrees	700	800	100	1500
Max-Depth	3	3	1	3
Learning rate	0.01	0.01	0.01	0.1
BoostType	Gradient	Gradient	Gradient	Gradient

Application of Deep Neural Network

The k-fold cross validation is used to first determine the activation functions. The held-out approach 5-fold cross validation is applied for ReLu, Sigmoid and Leaky ReLu activation function. The activation functions for the layers were then chosen based on the accuracy and the highest AUC score. The activation functions are fixed during the optimal hyper-parameter search. GridSearchCV is used to find the optimal batch size, number of hidden layers, drop out rates and the epochs. A dictionary of hyper-parameters with the parameter name and an array of values are provided to evaluate a model in the parameter grid. The gridsearch construct and evaluate the model for each combination of parameters. The combination of parameters which gives the best score are used. The optimal hyper-parameters are added to the sequential model. Adam Optimizer is used to find the individual learning rate of each parameter. Table 3.3 show the parameters used to build the DNN model. ReLu activation function is used to activate the hidden layers, and sigmoid function is used to activate the output layer.

TABLE 3.3: DNN Hyper-parameters

Parameters	Low Category	Int category	High Category	No-cut
No. Hidden Layers	3	3	3	4
Batch Size	100	100	60	20
Epochs	100	100	100	10
Dropout	0.1	0.1	0.1	0.2

3.2.4 Performance Metrics

To measure the performance of the models, ROC curves, AUC score, confusion matrix and output distributions are used. The ROC curve is a summary of the confusion matrix, however, we use both the ROC AUC score and the confusion matrix for analysis. The output distributions are used for easy read and interpretation of the results. For a model that can distinguish S and B events with no misclassi cations, we expect a ROC curve with an AUC score of 1, and an output distribution that shows an ideal separatibity with no overlapping of the S and B classes.

3.3 Summary

Simulated Monte Carlo dataset is used in this study. Optimal BDT and DNN models used in this study are developed following the supervised and weakly supervised learning approaches. The weakly supervised learning approach implemented is called weakly supervised learning where two samples of unlabeled data are used, with one sample containing mixed S and B events. A model is trained to rst distinguish events from each sample, and then used to discriminate S and B events from new dataset.

Chapter 4

Results and Discussion

In this Chapter the supervised learning and the weakly supervised learning (WSL) results achieved with BDT and DNN are discussed. The output distributions show how well the models can distinguish S and B events. The ROC curve is the plot of signal ef ciency against background rejection, a summary of the confusion matrix. Using the output distributions, table of results and the ROC curves, the supervised learning results the WSL and both the BDT and DNN overall performance are presented. The instances will be referred to as events in this chapter.

4.1 Boosted Decision Trees Results

The results obtained using BDT are presented in this section. The BDT supervised learning approach results are presented rst in Subsection 4.1.1. The WSL approach results are presented in Subsection 4.1.2.

4.1.1 Supervised Learning

Table 4.1 shows the performance of the supervised learning BDT. The Low category correctly classi ed 75% of S and 71% B, misclassifying 25% of S and 29% of B. The Int category model correctly classi ed 68% S and 74% B. The High category model correctly classi ed 68% of B and 52% of S events. The No-cut category model correctly classi ed 85% of S and 89% of B.

TABLE 4.1: BDT supervised learning Table of Results

	Correctly Classi ed		Misclassi ed	
	S	B	S	B
Low Category	0.75	0.71	0.25	0.29
Int Category	0.68	0.74	0.32	0.26
High Category	0.52	0.68	0.48	0.32
No-cut Category	0.85	0.89	0.15	0.11

Figure 4.1 represent the ROC curves of the BDT supervised learning. The Low category model achieved 78% AUC represented by the blue curve showing that the is 22% likelihood of misclassifying events. The Int category model achieved 79% AUC as shown by the green curve and the High category represented by the maroon curve achieved 69% AUC. The is 21% and 31% likelihood of misclassifying events with the Int and Hihg categories, respectively. The red curve represents the No-cut category with 94% AUC. The No-cut illustrates a low likelihood of misclassifying events with the model.

FIGURE 4.1: ROC curves illustrating the performance of the the BDT supervised learning model.

The BDT output in Figure 4.2 shows how well the BDT can distinguish between the S and B events. The red step lled distribution represent the training B events and the the blue step lled distribution represent the S train events. The purple and green step distribution represent the B and S test. The Low BDT output shows that the model distinguished the S and B with less uncertainty. A separation of S and B is observed with a small amount of S and B overlapping between -2 and 2 regions. The output distribution shows that the is a close t between the train and test output, which demonstrate that the model is slightly overtraining. The S and B separation is also observed on the Int category distribution with a slight moderate amount of S and B overlapping. The distribution demonstrate a close t of the test to the train events, with more overtraining observed in the S region. The High category model performed poorly. The is a huge overlap of S and B with little separation observed. The No-cut category BDT output demonstrate a clear separation of S and B with little overlapping observed. The is little overtraining observed on the S region and an almost perfect t on the B region.

FIGURE 4.2: S and B Output distributions

4.1.2 Weakly Supervised Learning

Table 4.2 shows the performance of the models when predicting S and B. The Low category model is bias towards the B. The model correctly classi ed 97% of B and misclassi ed 84% of S events. The Int category correctly classi ed 69% of S and 74% of B events, misclassifying 31% and 26% of S and B, respectively. The High category model is bias towards the S. The model correctly classi ed 99% of S and misclassi ed 94% of B. The No-cut category is also bias towards the B with 97% of the B correctly classi ed and only 61% of S events classi ed correctly.

TABLE 4.2: BDT WSL Table of Results

	Correctly Classi ed		Misclassi ed	
	S	B	S	B
Low Category	0.16	0.97	0.84	0.03
Int Category	0.69	0.74	0.31	0.26
High Category	0.99	0.06	0.01	0.94
No-cut Category	0.61	0.97	0.39	0.3

Figure 4.3 shows the WSL ROC curves, a summary of Table 4.2. The Low category represented by the blue curve shows that the model achieved 77% AUC, which illustrates that the model have 23% likelihood of misclassifying events. The Int and High category curves in green and maroon, respectively, achieved 79% and 64% AUC. The No-cut category model achieved 93% AUC. The likelihood of the No-cut model misclassifying events is low. The high AUC scores achieved by the models are due to the high ratio of B over S.

FIGURE 4.3: ROC curves illustrating the performance of the the BDT WSL model.

The output distributions in Figure 4.4 represent the BDT training output of M1 and M2 (Left) and the performance of the BDT model when predicting the test S and B (Right). The Low category train output shows that the is less certainty in the results. The is more M1 and M2 events overlapping and a slight separation achieved. The Low category S and B test output distribution shows an improvement when the model predicts S and B. The is less overlapping of S and B observed as compared to the M1 and M2 distribution. The Int category distributions shows that the performance of the model on the M1 and M2 train is close to the performance of the S and B test. The is S and B separation observed on the distribution with a moderate amount of events overlapping. The High category BDT output distributions shows that the model performed poorly on classi cation of both M1 and M2 events, and also the S and B events. The is a strong overlap of events on both the train and test distributions. The No-cut category BDT output distribution of the M1 and M2 shows that the model failed to distinguish M2 events from M1 events. The distributions shows that the M2 events are misclassi ed as M1 events. The S and B output distribution shows a separation of the S and B events, with the B events correctly classi ed and fewer S events correctly classi ed. The No-cut model is bias towards the B events.

FIGURE 4.4: S and B Output distributions

4.2 Deep Neural Networks Results

The DNN results presented in this Section illustrate the performance of the DNN models implemented following the supervised approach and WSL approach. Subsection 4.2.1 presents the results of the supervised DNN models and the WSL results are presented in Subsection 4.2.2.

4.2.1 Supervised Learning

Table 4.3 shows a summarised classi cation results of the DNN supervised learning approach. The is a slight bias towards S observed in the Low category where 80% of S are correctly classi ed and 37% of B are misclassi ed as S. 63% of B are classi ed correctly and only 20% of S events are misclassi ed as B. The Int category achieved 77% and 62% of correctly classi ed S and B events, respectively. The High category shows bias towards S, where 82% of S are correctly classi ed and 66% of B events are misclassi ed as S. The No-cut category achieved 84% and 90%S and B events classi ed correctly, respectively, with only 16 % of S misclassi ed as B and 10% of B misclassi ed as S.

TABLE 4.3: DNN supervised learning Table of Results

	Correctly Classi ed		Misclassi ed	
	S	B	S	B
Low Category	0.80	0.63	0.20	0.37
Int Category	0.77	0.62	0.23	0.38
High Category	0.82	0.34	0.18	0.66
No-cut Category	0.84	0.90	0.16	0.1

The ROC curves in Figure 4.5 summarises Table 4.3. The Low category represented by the blue curve shows that the model achieved 78% AUC. This means the is a 22% likelihood of misclassifying events with the Low category model. The Int category represented by the green curve shows that the model achieved 75% AUC. The High category model achieved 61% AUC which illustrate that the is 39% likelihood of misclassifying events and only 61% likelihood of correctly classifying the events.

The No-cut category represented by the red curve shows a high likelihood of correctly classifying the events. The model achieved 94% AUC.

FIGURE 4.5: ROC curves illustrating the performance of the DNN supervised learning model.

The DNN output distribution in Figure 4.6 shows the S and B separation achieved by the DNN models. The Low and Int category distributions shows that the models could separate the S and B events uncertainty, demonstrated by the overlapping S and B events. The is close t of the test distribution to the train, with overtraining observed on both the Low and Int Categories. The High category model performed poorly, with the S and B strongly overlapping. The No-cut category distribution shows a clear separation of the S and B and an almost perfect t on the B region with slightly-to-no overtraining observed on the S region.

FIGURE 4.6: S and B Output distributions

4.2.2 Weakly Supervised Learning

The WSL tabulated results in Table 4.4 shows the performance of the models on the Low, Int, High and No-cut category. The Low category model demonstrate a strong bias towards the B where 97% of B is classified correctly as B and only 3% is misclassified as S. The model misclassified 82% of S as B and only 18% of S is correctly classified. The Int category shows that the model achieved 67% and 71% of S and B events correctly classified. The High category correctly classified 99% of S with 96% of misclassified B, demonstrating a strong bias towards S. The No-cut model is also bias towards the B. The is 97% of B events classified correctly and only 58% of S is correctly classified. The is a strong bias demonstrated by the WSL models.

TABLE 4.4: DNN WSL Table of Results

	Correctly Classi ed		Misclassi ed	
	S	B	S	B
Low Category	0.18	0.97	0.82	0.03
Int Category	0.67	0.71	0.33	0.29
High Category	0.99	0.04	0.01	0.96
No-cut Category	0.58	0.97	0.42	0.03

Figure 4.7 shows that ROC curves and AUC of the WSL models. The Low and Int categories achieved 77% and 72% AUC represented by the blue and green curves, respectively. The Low and Int models have 23% and 28% likelihood of misclassifying events. The High category achieved 60% AUC and the No-cut category achieved 93% AUC. The No-cut category represented by the red curve shows that the is low likelihood of misclassifying events. The high AUC score achieved by the models shows that the is a high ratio of B over S as Table 4.4 demonstrated a strong bias in the Low, Int and No-cut categories.

FIGURE 4.7: ROC curves illustrating the performance of the DNN WSL model.