CLASSIFYING CANCEROUS TUMOURS USING MACHINE LEARNING TECHNIQUES

by

Jingyuan Gong

Supervisor

Dr **Rendani Mbuvha**

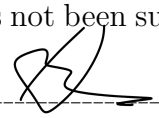
A Dissertation submitted to the Faculty of Science, University of the Witwatersrand, Johannesburg, in fulfilment of the requirements for the degree of Master of Science

March 9, 2022

Contents

Declaration

I declare that this Dissertation is my own, unaided work. It is being submitted for the Master of Science at the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination at any other University.



(Signature of candidate)

9th day of March 2022 in Johannesburg, South Africa.

Abstract

Cancer has become a leading cause of death in the modern world. Literature suggests that in the modern world, a third of the population will develop cancer during their lifetime. The focus of the dissertation was to classify tumours as malignant or benign tumours. The data was obtained via the Surveillance, Epidemiology and End Results (SEER) program, which collected data from 1973 to 2018. The SEER program gives data on cancer incidences obtained from the United States population and represents 28% of the United States population. The data set contained variables such as age, race, sex, year of diagnosis and tumour classification, along with 14 other variables. The methods used for modeling were K-Nearest Neighbours (KNN), Weighted K-Nearest Neighbours, Artificial Neural Networks, Naive Bayes classifier and Bayesian Neural Networks. All models above used Synthetic Minority Oversampling Techniques (SMOTE), as the data set was imbalanced with a ratio of 40 to 1 for the malignant tumours. The best model for the data set was the KNN model with five neighbours and SMOTE application, with an area under the curve (AUC) of 0.781.

List of Figures

List of Tables

1 Introduction

1.1 Background

Cancer has become a leading cause of death in the modern world. Chakraborty and Rahman (2012) mentions that in the modern world, a third of the population will develop cancer during their lifetime. According to the National Cancer Institute (NCI) cancer is "a term for diseases in which abnormal cells divide without control and can invade nearby tissues". The cancer type will be determined based on the area where the tumour first originated, the types of cancer include brain cancer, breast cancer, colorectum (cancer found in the colon or rectum), leukemia, lymphoma and many more. The location of the tumour will cause different symptoms (Nall, 2018).

The World Health Organisation (WHO) found that the most common types of cancer in 2020 were breast cancer (2.26 million cases), lung cancer (2.21 million cases), colon and rectum cancer (1.93 million cases), prostate cancer (1.41 million cases) and skin cancer (1.2 million cases) (WHO, 2021). WHO (2021) further noted that lung cancer was the most lethal type of cancer with 1.8 millions deaths followed by colon (935 000 deaths), liver (830 000 deaths), stomach (769 000 deaths) and breast (685 000 deaths). The total deaths caused by cancer in 2020 is estimated to be nearly 10 million (Ferlay et al., 2020).

The symptoms of cancer vary depending on the type, with brain cancer victims suffering from visual disorders, motor dysfunction, communication deficit, headaches, seizures and bladder control (Taphoorn et al., 2010). Leukemia symptoms include skin bleeding, fever, bone pain, night sweats, recurrent of infections and exhaustion (Bernbeck et al., 2009). The symptoms affect the diagnosis and mortality of the patient, hence there will be different incidence rates and mortality rates depending on the type of cancer.

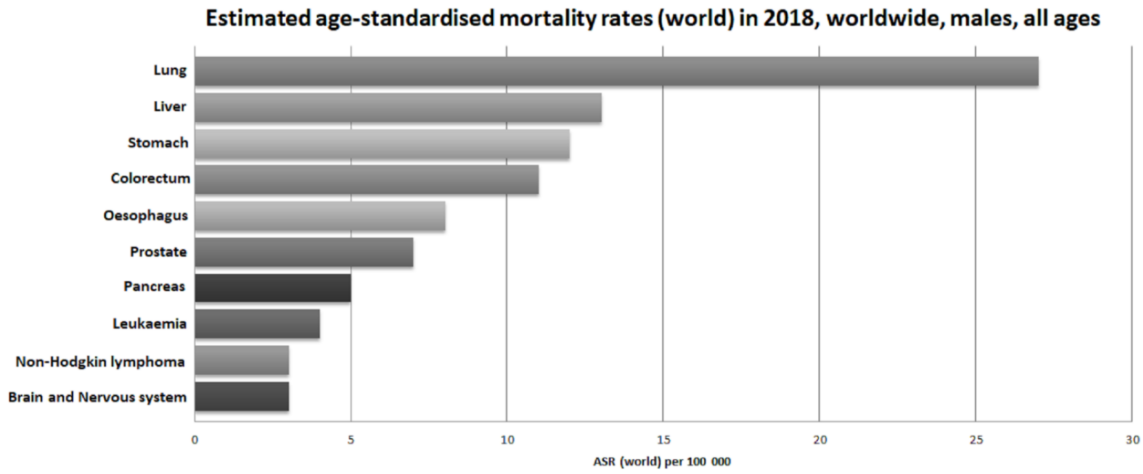


Figure 1.1: Male mortality of different cancers. Re-created from: Rawla and Barsouk (2019)

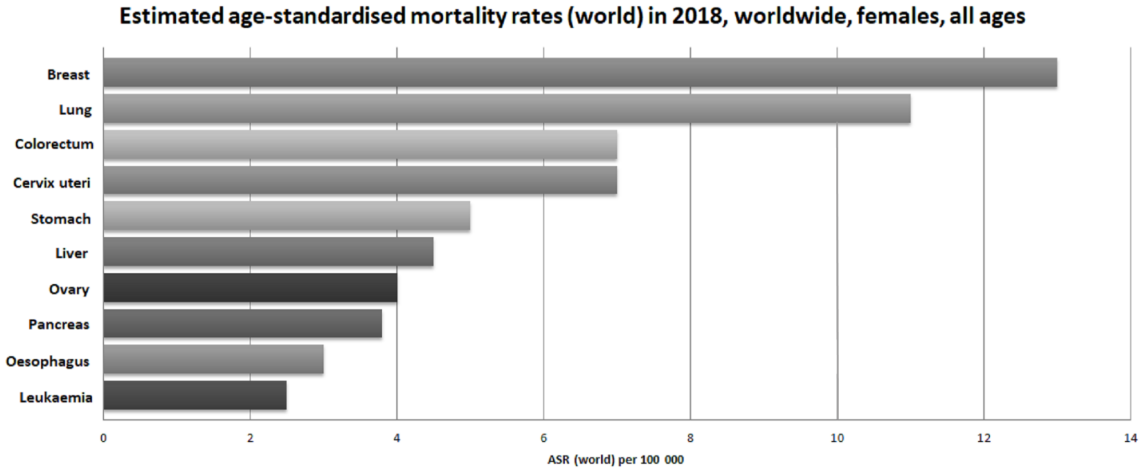


Figure 1.2: Female mortality of different cancers. Re-created from: Rawla and Barsouk (2019)

Figure 1.1 and Figure 1.2 shows the mortality rate for different types of cancer along with different genders. Rawla and Barsouk (2019) showed that lung cancer has the highest mortality rate for males, while breast cancer has the highest mortality for females. The incidence rates are for per 100 000 in the population.

Rawla and Barsouk (2019) lists genetics as a cause of gastric cancer due to the CDH1 gene being an inherited mutation which have been found to increase the risk of stomach cancer. The authors further noted that blood groups also have a relationship with stomach cancer, with blood group A being associated with a higher risk of stomach cancer.

While genetics are important, the lifestyle of an individual may also become important with reducing the risk of cancer, with potential risk factors being alcoholism, diet, lifestyle, smoking and viral infection (Chakraborty and Rahman, 2012). Alcoholism and the use of tobacco have been shown to increase the risk of gastric cancer by 39% and 11% respectively (Rawla and Barsouk, 2019). The consumption of harmful products and behaviour which increase the risk of cancer are due to the understanding of cancer and its symptoms.

Symptom detection is very important to reduce the patient interval time (Emery, 2015). Patient interval is the time between developing a symptom and seeking professional help. The other delay known as primary care interval is the interval from a patient seeking a general practitioner to referral for a specialist (Emery, 2015). Primary care interval arises due to physicians needing to differentiate the patients who need urgent care and the majority of patients who do not need further testing (Al-Azri and Mohammed, 2016). Al-Azri and Mohammed (2016) also note delays may also occur due to the medical equipment available to the practitioner along with the patients financial status.

Delays may also occur when a specialist needs to determine if the tumour is benign or malignant (Omondiagbe et al., 2019). Pietrangelo (2019) defines benign tumours as tumours that do not contain any cancerous cells, thus the benign tumour can not spread to other site. While malignant tumours do contain cancerous cells, therefore the malignant tumour could spread to other sites. The accuracy of identifying a tumour as being benign or not is important, as detecting benign tumour may prevent unnecessary treatments (Sahu et al., 2019). The models built in this thesis was to address these problems using machine learning.

The usage of machine learning on medical databases has increased drastically, as data mining techniques can be used to explain or reveal relationships (Ahmad et al., 2013). Machine learning can explain the attributes which would identify a tumour as benign or malignant.

1.2 Statement of the Problem, Aims and Objectives

1.2.1 Statement of the problem

The main focus of this study is to accurately classify a tumour as benign or malignant in order to reduce the delay and error in patient diagnosis.

1.2.2 Motivation of the study

In order to reduce delay and increase accuracy of cancerous tumour classification, an extensive study of Bayesian and Frequentist machine learning techniques were compared with(out) class balancing.

The comparison of Bayesian and Frequentist machine learning techniques allows for a deeper understanding of which techniques should be applied for classifying cancerous tumours.

1.2.3 Contribution(s) of the Study

Current literature has a lack of comparison between the Bayesian and Frequentist classification models used for classifying cancerous tumours. Furthermore there was a lack of studies done on the SEER data set with Bayesian approaches. The Bayesian approaches on the SEER data set allows for the doctors to model uncertainty in diagnosis.

1.2.4 Scope and Delineation of the study

In this thesis the models was not built using:

- Radiological scans
- Methods considering DNA

1.2.5 Organization of the thesis

In Chapter one the introduction of the thesis was presented along with the problem of the statement, aims and objectives.
In Chapter two the literature about using machine learning techniques for tumour classification was discussed, along with reviews about the lack of Bayesian methods used for tumour classification.

1.2.6 Objectives

The objectives of the study are:

1. Using Naive Bayes (NB) classifier, K-Nearest Neighbours (KNN) and Weighted Nearest Neighbours of Samworth (2012) (WNN) as baseline classification models.

2. Compare the NB models which used SMOTE and threshold optimization with other NB models without SMOTE and threshold optimization.

3. Compare the KNN models which used SMOTE with models without SMOTE.

4. Compare the WKNN models with different kernels and distances.

5. Apply Artificial Neural Networks (ANN) with different activation functions to classify tumours as benign or malignant.

6. Compare the different ANN models which use different activation function, different dropout layers and differences in SMOTE and threshold optimization.

7. Apply Bayesian Neural Network (BNN) with Markov Chain Monte Carlo to classify tumours as benign or malignant.

8. Apply BNN with variational inference to classify tumours as benign or malignant.

9. Compare the different BNN models which use MCMC or Variational inference, different dropout layers and differences in SMOTE and threshold optimization.

10. Compare the models using accuracy, sensitivity, specificity, Area under Curve (AUC) and Cohen's Kappa.

2 Literature Review

The application of machine learning in the medical field has rapidly increased. The increased usage is due to machine learning being able to learn complex patterns or relationships from complex data sets (Cruz and Wishart, 2006).

Ryu et al. (2019) focused on predicting the survival of patients with Spinal Ependymoma. The aim of the authors were to understand the demographic and clinical factors that would influence the survival of the patients, along with trying to predict the overall survival using machine learning. They use random forest and step-wise logistic regression methods to predict the survival rates. The results from the step-wise logistic regression model preformed better in prediction. Ryu et al. (2019) also used the SEER database, however, they did not consider to use Bayesian methods.

Literature that focuses on cancer prediction and prognosis have three main goals in common, which are to predict cancer susceptibility, predict cancer recurrence and predict cancer survivabilty. Cruz and Wishart (2006) use machine learning techniques to accomplish the three goals mentioned above. Unlike Ryu et al. (2019), they considered genomic data (mutations), proteomic data (protein biomarkers) and clinical data (tumour size, age etc.).

The usage of Artificial Neural Network (ANN) and the three types of data mentioned provided an improvement in predictive accuracy of most prognosis, with the authors noting that there is a 15-25% improvement in accuracy when comparing the results of machine learning methods and simple statistical methods.

Kourou et al. (2015) notice that many studies related to early cancer diagnosis and prognosis while using miRNAs, which are good for detection of cancer, have suffered from low sensitivity towards in screening for early stages of cancer. The low sensitivity results in tumours being misdiagnosed as benign or malignant.

While the three previous works all used regression type techniques, Doppalapudi et al. (2021) used both regression and classification techniques to predict the survival period of cancer patients. The main focus was to use both techniques to ensure a better prediction of the survival period to help patients and physicians to make better decisions. The model building phase was split into regression and classification, however, both techniques used ANN, Convolution Neural Networks (CNN) and Recurrent Neural Networks (RNN) as machine learning techniques. Random Forrest

(RF), Support Vector Machines (SVM) and NB were used as statistical classifiers. The most accurate model for classification was the ANN model with a 71.18% accuracy.

Amrane et al. (2018) use two basic classification methods, the NB classifier and KNN, to determine if a tumour found in the breasts were benign or malignant. They considered the mutation of the gene, pain level, change in tumour size, redness and the skin texture around the breast. Amrane et al. (2018) concluded that the KNN method provided a higher accuracy (97.51%) compared to the NB classifier (96.19%).

A finding of KNN being a superior classification measure compared to the NB classifier is further supported by Sharma et al. (2018). Sharma et al. (2018) consider KNN, NB and RF classifiers to determine if the tumour was benign or malignant. The model performance metrics used to evaluate the models were accuracy, precision and recall.

Classification methods were also considered by Ahmad et al. (2013) for predicting the recurrence of breast cancer. Ahmad et al. (2013) focus on three main data mining techniques, being ANN, Decision Trees (DT) and SVM. The justification for using data mining techinques was to reduce the number of false positive and false negative decisions made by the physician. The SVM model was the best classifier followed by ANN and DT respectively. The models could have been improved by using more variables and a larger data set.

Sahu et al. (2019) uses PCA and ANN as a hybrid model that can process data while using the most relevant variables. Other models such as KNN, NB, RF, RF with PCA and ANN without PCA were also considered. The most accurate model was the ANN and PCA hybrid model, with an accuracy of 97%, while the NB model had a accuracy of 81%. The study used classification techniques to identify tumours as being benign or malignant, because classification of benign tumours can avoid patients from undergoing treatments which are deemed unnecessary.

The importance of correctly identifying a tumour as benign or malignant was further highlighted by Omondiagbe et al. (2019). Omondiagbe et al. (2019) used linear discriminant analysis (LDA) and PCA to reduce the variables and computation time of the models. They first used LDA and PCA on the data then applied the ANN, SVM and NB, with NB being a baseline model. The model which preformed the best was NN with the use of LDA, which resulted in an accuracy of 98.82%, thus showing that dimensionality reduction (using LDA and PCA) are significant in the classification models that use machine learning.

The idea of machine learning algorithms and cancer classification is well established in literature, with ANN, KNN and SVM dominating the research in this field. There is, however, a lack of ANN models with a Bayesian component to classify tumours as benign or malignant.

Yu and Chen (2005) used BNN models for ovarian cancer identification. BNN usage allowed the model to reduce the computational power required, as BNN allowed the use of prior probabilities. The BNN models were split between k-fold cross validation with k ranging from 2 to 10. The average accuracy of all BNN models were around 97%, which was higher than the accuracy obtained by the ANN models with one (89.93%) and two hidden layers (84.09%).

Kourou et al. (2020) also used Bayesian methods, however, the data set used was based on timer series microarray data. The model used were Dynamic Bayesian Network, because the model could explain the importance of significant genes and key regulatory molecules. The accuracy and Area Under the curves (AUC) was used as the performance measure, which were 70.8% to 98.5% for accuracy and 0.562 to 0.985 for AUC.

Abdar et al. (2021) provided an insight of using images along with Bayesian deep learning to quantify the uncertainty of skin cancer classification. Abdar et al. (2021) used three uncertainty quantification methods, which were Monte Carlo dropout, Ensemble Monte Carlo dropout and Deep Ensemble. The model performance used were the F1 score, which was 88.95% to 91%.

The Bayesian component with ANN allows for the model to consider uncertainty in its prediction. In this dissertation machine learning techniques will be used to classify tumours as being benign or malignant.

3 Methodology

This section is split between explaining the data set, classification models and model accuracy measures.

3.1 Data set

The data was obtained via the Surveillance, Epidemiology and End Results (SEER) program. The SEER program gives data on cancer incidences obtained from the United States population and represents 28% of the United States population (Ryu et al., 2019). Surveillance Research Program (SRP), part of the National Cancer Institute's Division of Cancer Control and Population Sciences (DCCPS), supports the SEER program (NCI, 2021).

The SEER database contains data about cancer incidences from 1973 to 2018. The program collected data based on the patient's age, race, sex, year of diagnosis, primary tumour site, tumour morphology and tumour behaviour. The data set extracted contains the tumours behaviour re-code which can be explained in Table 3.1.

The data set was only acquired after completing a data-use agreement, which protects the identities of the cancer patients, with every effort of excluding features which can lead to an individual patient. The agreement further states the data will only be used for statistical reporting and analysis for research purposes.

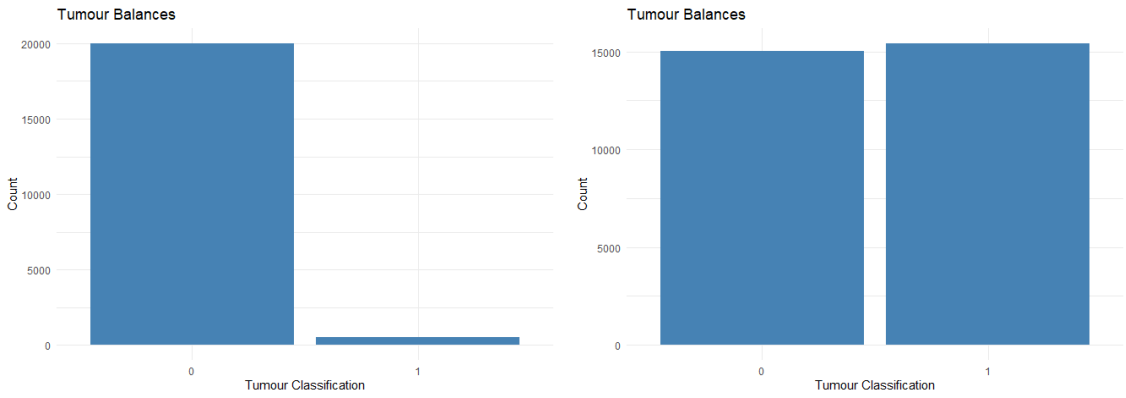
Table 3.1: Table explaining tumour recode, name and explanation

Tumour Classification	**Name**	**Explanation**
0	Benign	Tumour that is not cancerous.
1	Malignant	Tumour that is cancerous.

3.1.1 Class imbalance and SMOTE

To understand the data the benign and malignant tumours were plotted. The graphs were drawn from the SEER data set, with 0 representing malignant tumours and 1 representing benign tumours.

Figure 3.1a showed a bar graph with the class imbalance within the benign and malignant tumours of a ratio 40:1 in favour of malignant tumours. This was due to how the data was collected, with data collection occurring during or after tests. Data augmentation techniques, such as Synthetic Minority Oversampling Technique (SMOTE) and threshold optimization were used to handle the class imbalance. Figure 3.1b displayed the data set imbalance after SMOTE was applied. The number of cases in each class was shown to be at a ratio of 40:1, which has now been converted to a 1:1 ratio.



(a) Bar chart showing class imbalance (b) Bar chart showing data set after SMOTE

Figure 3.1: Bar chart showing class imbalance and after applying SMOTE class balance

3.1.2 Data Exploration

To gain further insight into the data set, Figure 3.2 was analysed. Table 3.2 showed the distribution of ages according to their age groups. The majority of tumour classifications occurred during the ages of 55 to 79, the ages were close to the United States of America's retirement age. Devesa et al. (1995) noted the mortality rates for cancer during 1975-1991 increased for older people, while for younger people the mortality rates decreased. The increase of mortality rate would increase the understanding and awareness of cancer, thus increasing number of tests. Devesa et al. (1995) used the SEER data set, but only used the years 1975 to 1991, and noted that the incident rates increase for all adjusted age groups during the same period.

Figure 3.2 demonstrates that for the benign tumours there was one outlier, at the age of 20 to 24. While for the malignant cases there were 6 outliers all below the age of 25. The outliers was due to the physician allowing for testing and therefore the patient becoming registered into the SEER database. The models built for this thesis was to reduce the number of people being tested while being in benign.

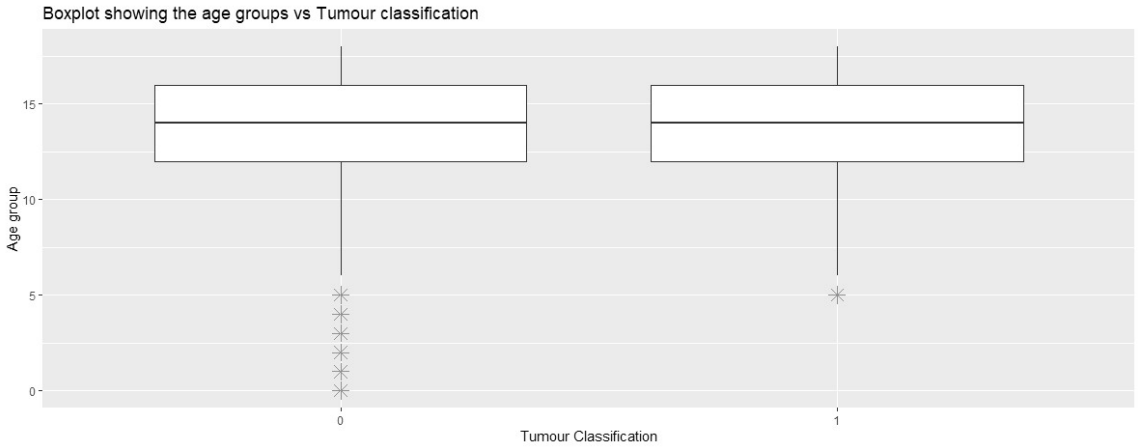


Figure 3.2: A boxplot showing the age group vs tumour classification

Table 3.2: Age group distribution

Age Group	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Age Range	0	1-4	5-9	10-14	15-19	20-24	25-29	30-34	35-39	40-44

Age Group	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**
Age Range	45-49	50-54	55-59	60-64	65-79	70-74	75-79	80-84	85+

3.1.3 Data partition

Fitting and evaluating statistical or machine learning models requires two different sets of data, being the training and test data sets.

Let I represent the original data set. The data set I was split into two mutually exclusive subset being L and T, with L being the training data set and T the test data set. Each method will require the same data sets to be used. The following steps are taken to partition the data sets:

1. **Split**
   Create a split of 75% for the training data set and 25% for the test data set.

2. **Creating partition**
   The portioning of the data set was done by using the *caret* library (Kuhn et al., 2020).

All analysis and coding from this paper was done in R-Studio package version 3.6.2 (Allaire, 2012).

3.2 Classification Models

3.2.1 Naive Bayes Classifier

The NB classifier is a probabilistic classifiers which uses the Bayes theorem. Naive Bayes classifier assumes a strong independence between the features, however, Rish et al. (2001) demonstrated that the NB works well for independent features and functionally dependent features. Zhang (2004) mentioned that one way to overcome the limitation of independence between features is to extend the model to have dependencies among the features. Figure 3.3a shows an example of a basic Naive Bayes classifier's relationship between a class C and the associated features (F_i), and Figure 3.3b shows a Naive Bayes classifier's relationship between a class C and the associated dependent features (F_i).

The Bayes theorem given as:

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}. \tag{1}$$

with X being the predictor, Y being the class, P(Y|X) being the posterior probability of class Y given X, P(X|Y) being the likelihood, P(X) being the prior probability of X and P(Y) being the prior probability of the predictor Y.

The Bayes theorem expresses the posterior probability in terms of the prior probability, the probability of the predictor, and with the conditional probability. The conditional probability is therefore needed in order to obtain the posterior probability. The conditional independence assumption must hold to calculate the conditional probability.

The conditional independence is given as:

$$P(X|Y = y) = \prod_{i=1}^{z} P(X_i|Y = y). \tag{2}$$

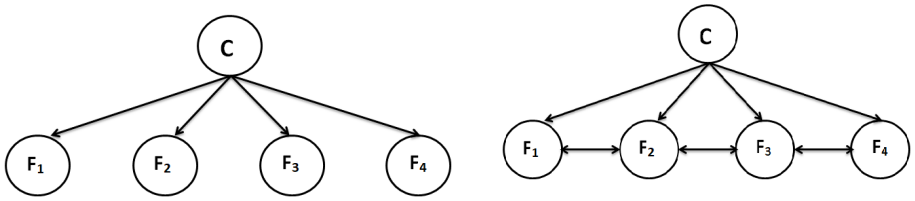
with the set $X : \{X_1, X_2, ..., X_z\}$ and z being the number of features.

The NB classifier combines equation 1 and equation 2 to calculate the posterior probability for all classes of Y. The NB classifier is given as:

$$P(Y|X) = \frac{P(Y)\prod_{i=1}^{z} P(X_i|Y)}{P(X)}. \tag{3}$$

with P(X) being stationary for every Y, thus the classes are determined by maximising equation 2. The probability of every observation from the test data set is calculated for every class. Each test observation will be classified to the class with the highest probability.

We use the *e1071* library of Meyer et al. (2019) to train the naive Bayes classifier.



(a) Naive Bayes (b) Bayes with dependent features

Figure 3.3: Examples of Naive Bayes. Re-created from: Zhang (2004)

3.2.2 K-Nearest Neighbour

KNN was first introduced by Fix (1985), with the KNN being used to preform a non-parametric test for pattern classification. KNN attempts to classify an observation into a particular class if the observation has the highest probability of being in that particular class. KNN has since become one of the most used classification techniques (Peterson, 2009).

Before KNN can be used, the data set needs to be split into a training data set and a test data set. The training data set is used to train a certain model, while the test data set is used to evaluate the performance of that set certain model. KNN looks to find a neighbourhood around a certain observed value in the test data set with the help of observations found in the training data set.

KNN are based on the distance between the test data set observation and specified observations in the training data set. The between sample geometric distances, $d(x, y)$, used by KNN include:

- Minkowski distance $((\sum_{i=1}^{n} \|x_i - y_i\|^c)^{1/c})$,
- Manhattan distance $(\sum_{i=1}^{n} |(x_i - y_i)|)$
- Euclidean distance $(\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2})$.

The Minkowski distance is a generalized metric, in the normed vector space, to calculate distance between two points (Singh et al., 2013). Minkowski distance allows for other measurements when c is set to different values (De Amorim and Mirkin, 2012). For $c = 1$, the Manhattan distance is obtained. For $c = 2$, the Euclidean distance is obtained. Minkowski distance only works for $c \geq 1$, as a metric requires that the

- triangle inequality $(d(x, z) \leq d(x, y) + d(y, z))$
- symmetric $(d(x, y) = d(y, x))$
- positive properties$(d(x, y) = 0$ if $x = y)$.

For $c \geq 1$ the triangle inequality holds for the Minkowski distance. However, when $c < 1$ then $1/c > 1$ occurs, which results in $d(x, z) > d(x, y) + d(y, z)$. The triangle inequality does not hold for when $c < 1$ (Gohrani, 2019).

The Manhattan distance is calculated using the Minkowski distance by setting $c = 1$. The Manhattan metric measures the distance between two points using right angles. The Manhattan distance also known as the taxicab distance, a realistic use of the Manhattan distance is the path a person would drive that covers the streets and not through buildings (Khan et al., 2019). The Manhattan and Euclidean distances are popular metrics used to determine similarities between data points (Vadivel et al., 2003).

The Euclidean distance is calculated using the Minkowski distance by setting $c = 2$, also known as the L2 metric. The Euclidean distance is a metric structure that considers length between two points x_i and y_i for all i being a subset of $1, 2, ..., n$ (Strichartz, 2000). Strichartz (2000) also stated the Euclidean distance was used the most in practice due to three properties: the distance between two points is positive, the distance is symmetric and the distance follows triangle inequality.

For a positive integer k and an observation x_0 in the test data set, KNN obtains the k number of observations in the training set that are the closest to x_0. The conditional probability for a certain class j is then estimated as a fraction of points inside the neighbourhood of N_0.
The probability can be calculated as:

$$P(Y = j|X = x_0) = \frac{1}{k} \sum_{i \in N_0} I_{y_i = j}. \tag{4}$$

Equation 4 calculates a probability of the test observation belonging in every class. The test observation will be classified to the class with the highest probability.

The usage of KNN provides another problem. The problem is that the usage of the geometric distance does not consider the weights of the features, thus assuming that all features have the same effect on the classification. KNN also does not consider how the features are related. In order to correct these mistakes, the WNN will be used.

Use the *knn* function, written by Ripley et al. (2015) from the library *class*, to find the k nearest training set for each observation from the test data set.

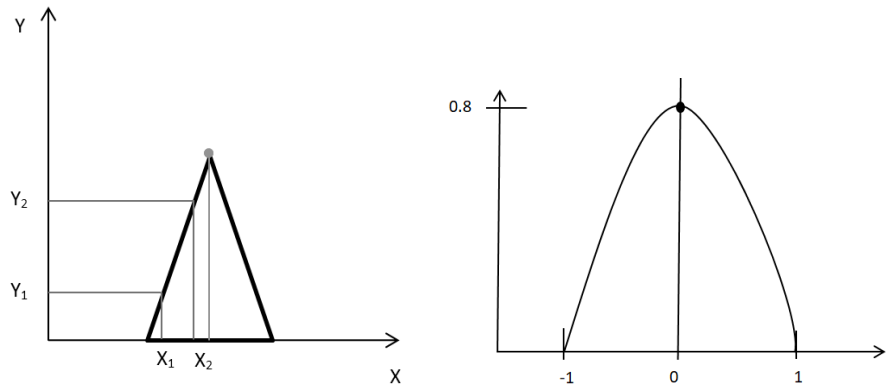
3.2.3 Weighted Nearest Neighbour

Dudani (1976) introduced the idea of weighting close neighbours more than further neighbours. The WNN was created as an expansion of KNN. WNN uses kernel functions to assign weights with accordance to their distances. The assignment of the weights are important and can be assigned using four methods: random weights, instance weighted based on gradient descent, feature weighted based on gradient descent and kernel function to assign weights.

The random weights method is to randomly assigning weights according to the features. The classification error rate will be calculated, then the weights are adjusted based on the classification error rate. The process is repeated until the classification error rate is acceptable.

The instance weighted method would assign weights randomly in the training instances and the weights would be trained by cross validation. The feature weighted method would randomly assign weights to all the features and the weights would be trained by cross validation.

Given a test observation, a kernel function would assign more weight to the observations that is closer to the test observation. Samworth et al. (2012) noted that there are various number of kernels, with kernels being rectangular, Epanechnikov, triangular, bi-weight and tri-weight.



(a) Triangular Kernel (b) Epanechnikov Kernel

Figure 3.4: Triangular and Epanechnikov kernel function

The kernel shape affects how the weights are assigned. Figure 3.4a shows a triangular kernel function, with the point at the peak of the triangle being the test observation, and points from the training data set being points (X_1,Y_1) and (X_2,Y_2). The weights of the point (X_2,Y_2) would be higher than point (X_1,Y_1) as the point (X_2,Y_2) is closer to the test observation. Figure 3.4b follows the same principle, however, any points falling outside the curve will be given no weights and points closer to the test observation will be given the varying weights. The optimal number of neighbourhood being used to calculate the kernel is a function of k and d with k being the number of neighbourhood for an unweighted KNN.

The probabilities would be calculated similarly to the KNN probability (equation 4), and will calculate the probability of every observation from the test data set belonging to each of the classes. The function *kknn* from the library *kknn*, written by Schliep et al. (2016), allows each observation from the test data set to find the k nearest training set vectors.

3.2.4 Artificial Neural Network

The ANN is a machine learning technique which was made to resemble the human brain. Yegnanarayana (2009) considered ANN as a simplified model of the biological neural network. The ANN function is to take in data, train itself to recognize patterns in the data and provide an output. The ANN architect may be split into three sections, the input layer, hidden layers and output layer. The features are fed into the input layer, the results from the input layer will be processed in the hidden layer(s) and will be passed out to the output layer. These steps are known as forward propagation.

The number of nodes in the input layer ($L^{(0)}$) will depend on the number of features presented to the neural network. In Figure 3.5 the are three nodes in the input layer, meaning there are three features presented to the network. The hidden layer(s) ($L^{(r)}$) with r being the r_{th} hidden layer, can have different numbers of neurons (units where calculations are done) in the different hidden layers. Each hidden layer contains one or more neuron(s), with the neurons of one layer being connected to the neurons of the next layer through channels as seen in Figure 3.5.

Each channel contains a weight, the weight will affect the input received by the next layer's neurons. The weights from i_{th} neurons of the layer $(L^{(r-1)})$ to the j_{th} neurons of layer $(L^{(r)})$ is represented as $w_{j,i}^{(r,r-1)}$.

Each of the neurons also contains a bias (b), the bias of each neurons in the same layer may be the same, or each neurons in the layer may have its own bias. The neurons in the hidden layer are affected by the input from the layer before, the weights of each channel and its own neuron's bias. The nodes inside the input layer do not have bias terms. The ANN structure is an isomorphic set of functions to the possible bias and weights of the structure (Jospin et al., 2020).

There further exist an activation function, that will determine if the neuron will be activated into the next layer. The are a numerous number of activation functions, such as the hyperbolic tangent (Tanh) function, binary sigmoid function and Rectified Linear Unit (ReLU). The activation functions can be seen in Figure 3.6 and 3.7.

The Tanh(x) function can be expressed as a ratio of $sinh(x)/cosh(x)$, with $sinh(x) = (e^x - e^{-x})/2$. Cosh(x) can be expressed as $(e^x + e^{-x})$. Therefore, $Tanh(x) = (e^x - e^{-x})/(e^x + e^{-x})$. For a large value of x, the Tanh function gets close to one. For sinh(x) being greater than cos(x), Tanh tends to -1. thus the curve resulted into a S-shaped curve (Mathcentre, 2006). The S-curve seen in Figure 3.6, shows that the function is between 0 and 1.

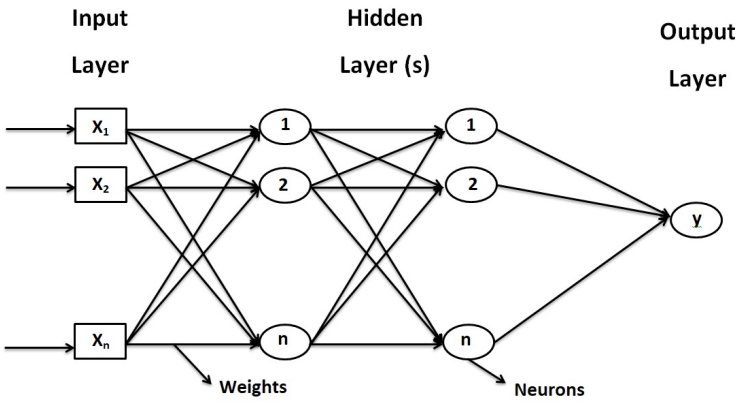


Figure 3.5: An Artificial Neural Network with two hidden layers, n inputs and one output.

The sigmoid function $f(x) = (1)/(1 + e^{-x})$ being a monotonic, bounded and differentiable function makes the function well suited for the activation function, as the function has a non-negative derivative at each point (Han and Moraga, 1995). The differentiable function meant that the slope can be found at any two points (Sharma, 2017). The S-curve seen in Figure 3.7a , shows that the function is between 0 and 1.

The ReLU $f(x) = \max(0, x)$, x $\in \Re$, is a more modern activation function. Brownlee (2019) notes that the ReLU function is a nonlinear function which allows for complex relationships between features. The use of the ReLU activation function has increased due to the ReLU function being able to overcome the vanishing gradient problem, thus increasing the models performance time and accuracy (Brownlee, 2019). Figure 3.7b shows the ReLU activation function.

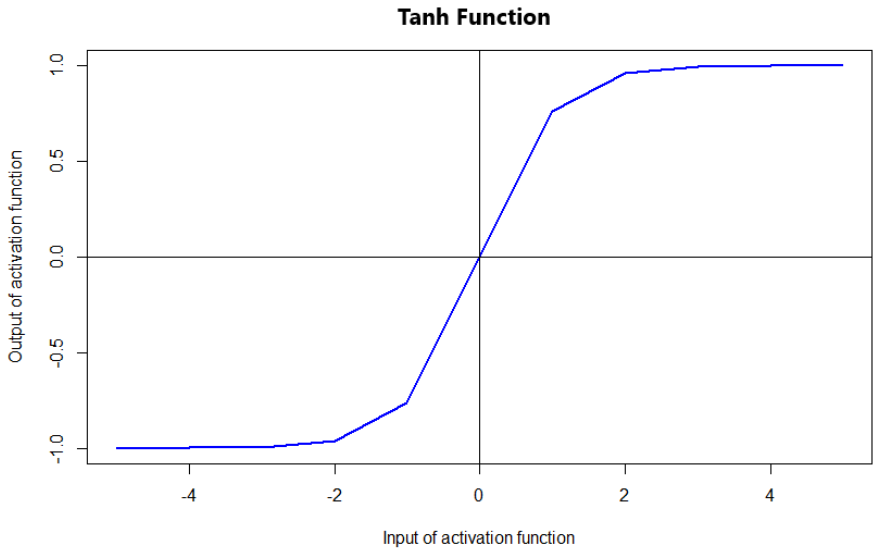
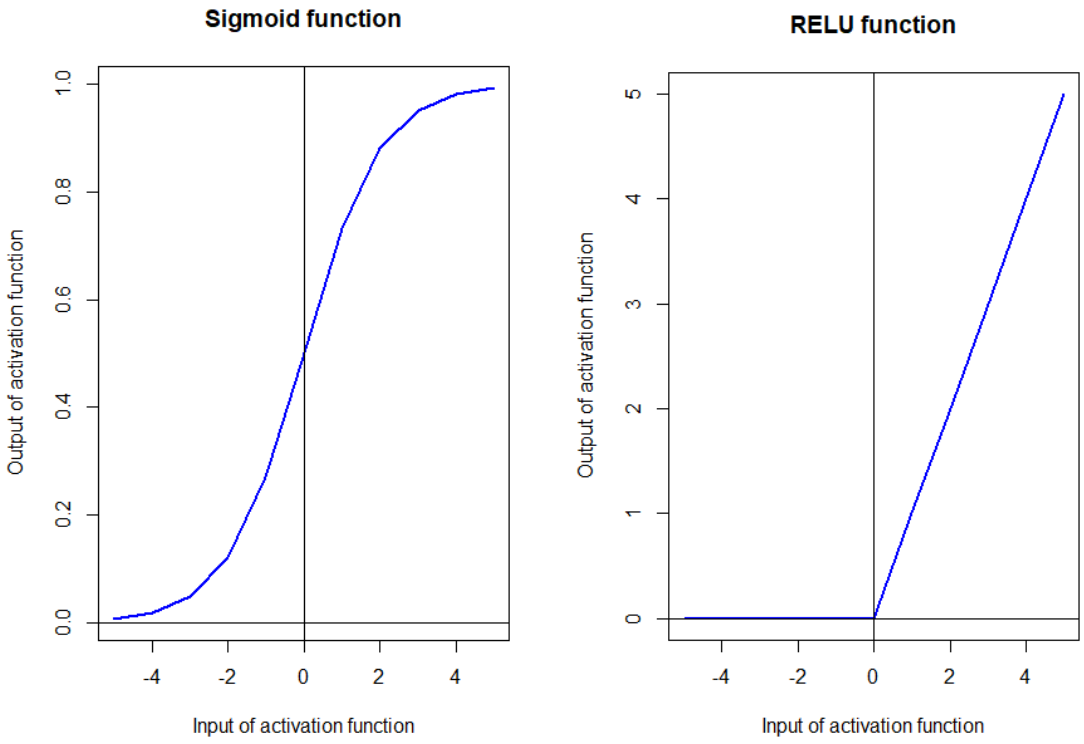


Figure 3.6: Tanh activation function



(a) Sigmoid function (b) ReLU

Figure 3.7: Sigmoid and ReLU activation functions

One major problem about the ReLU activation function is when the inputs are smaller than 0 the gradient becomes 0, this causes the neurons to deactivate if the input is 0 or less. The movement of data from one neuron to the next can be explained by equation 5:

$$y_j = \mathrm{f}_j\left(\sum_{i=1}^{n} w_i x_i + b_i\right). \tag{5}$$

with y_j being input at the next layer's j_{th} neuron, f_j being activation function for the j_{th} neuron in the next layer, w_i being weight of neuron i from the previous layer, x_i being input of neuron i from the previous layer and b_i being bias of neuron i from the previous layer.

In the output layer the neuron with the highest probability will determine the output. Since the true value is known from the training data set, the predicted output will be compared to the true value from the training data. The adjustment of weights will be done until the network can correctly predict the output. This step is known as back-propagation and was introduced by Rumelhart et al. (1986). Back-propagation uses gradient decent of the errors between the predicted value and true value.

The network's loss function (cross entropy loss) is defined as:

$$L = -\sum_{i=1}^{N} t_i log(y_i). \tag{6}$$

with t_i being the true label of the network and i_{th} node, and y_i being the predicted output seen in the network.

Gradient decent used to update the weights can be shown in the equation below:

$$w^{(t+1)} = w^{(t)} - \eta \frac{\partial E(w)}{\partial w}. \tag{7}$$

with η being the learning rate.

In equation 7 the partial derivative for a single input node is separated into two parts, the hidden layers to output layer and the input layer to the hidden layers. The hidden layers to output layer's gradient for biases b_k is described as:

$$\frac{\partial L}{\partial b_k} = -(t_k - y_k) * f'(y_k). \tag{8}$$

with t_{ik} being the true value of the k_{th} class and i_{th} node, y_{ik} being the output value seen in the network and $f()$ being the activation function.

The partial derivative from the hidden layers to output layer is described as:

$$\frac{\partial L}{\partial w_{jk}} = -(t_k - y_k) f'(y_k) y_j. \tag{9}$$

with y_j being the hidden layer's outputs and with y_k being the output layer's output.

Jain et al. (1996) mention that the back-propagation algorithm can be broken into the following steps:

Algorithm 1: Back-propagation algorithm

```
Initialise the weights from a standard N(0, 1);
while Error is high do
    Choose an input pattern for the neural network;
    Feed the network to get an output layer;
    Compute the i_th neuron in the output layer;
    Compute the previous neurons by propagating the errors backwards;
    Update the weights;
end
```

Training the artificial neural network by means of back-propagation with weight backtracking with the *neuralnet()* function. The *neuralnet ()* function taken from the library *neuralnet* by Fritsch et al. (2019).

3.2.5 Bayesian Neural Network

The deep learning models such as ANN with more than two hidden layers, tend to be overconfident in their predictions. The overconfidence is problematic for medical diagnostics, as the failure in classification could lead to dramatic outcomes (Jospin et al., 2020). The overconfidence and lack of insight of how the calculations are happening in the neurons of an ANN model has also been noted by Nixon et al. (2019) and Guo et al. (2017). One way to reduce the risk of overconfidence and provide some insight of what is happening in the model is to introduce a neural network with a stochastic nature.

A Bayesian neural network is defined as a neural network of stochastic nature, that was trained by using Bayesian inference (MacKay, 1992). Bayesian inference is based on probability instead of certainty, and that the prior probability influences the posterior probability. The first use of Bayesian analysis was used by Thomas Bayes and Laplace, they used a constant prior distribution for unknown parameter estimation (Berger, 2000). The Bayes theorem summarises the use of priors and likelihood usage to get the posterior distribution, the theorem can be see in equation 1. One of the main strengths of Bayesian analysis is the usage of the prior knowledge (Berger, 2000).

The structure of a BNN is similar to that of an ANN, but there is either a stochastic activation function or stochastic weights involved. The stochastic activation function or stochastic weights allows for multiple models, therefore there are sets of models trained and their prediction is aggregated to get one outcome. Galton (1907) states that the aggregate predictors from a large set of independent outcomes can lead to a better predictor than a model that is a single well-performing model.

The BNN modeling phase can be split into choosing the neural network structure, and choosing the stochastic model. The choice of neural network has been described in Section 2.2.4, the BNN neural network structure is similar to the one of ANN. The choice of the stochastic model requires probabilistic graphic models.

Probabilistic graphic models are used to represent the inter-dependence between multivariate stochastic variables and their probability distributions using graphs. A specific cyclic directed graph called the Bayesian Belief Network (BNN) shows the inner structure of the joint probability distributions between the features. The joint distributions of the nodes can be calculated as:

$$p(n_1, ..., n_m) = \prod_{i=1}^{m} p(n_i|parents(n_i)). \tag{10}$$

with n_i being the node i.

To ensure independence between features and applying the Bayes theorem from equation 1, the Bayesian posterior may be written as:

$$p(\theta|L) = \frac{p(L_y|L_x, \theta)p(\theta)}{\int_\theta p(L_y|L_x, \theta)p(\theta)d\theta} \; \alpha \; p(L_y|L_x, \theta)p(\theta). \tag{11}$$

with θ being the model parametrisation, L being the training data set, L_x the input feature from the training data and L_y the label/result from the training data.

The computation of equation 11's denominator is difficult using standard methods, since it is an intractable problem. There are two possible methods to compute equation 11, Markov Chain Monte Carlo (MCMC) algorithm and the variational inference method. Berger (2000) states the reason why MCMC is popular is due to MCMC being able to handle complex data sets, complex calculations and is easily understood. MCMC's purpose is to allow sampling from probability distributions using Markov chains. Using Markov chains allows the method to focus on the more important regions of the data.

The Markov chain is a sequence of random variables from a random sample S_i, which depend on the previous sample S_{i-1} and none before. An initial burn in period is required, but the underlying Markov chain then converges to the posterior distribution.

The Monte Carlo simulation is to obtain an independent, identically distributed set of N samples from a density distribution. The number of samples allows for an approximation of the posterior distribution (Andrieu et al., 2003). Berger (2000) justifies the use of Monte Carlo sampling, as the method works with large data sets and produces reliable measures of accuracy.

The most popular MCMC method is the Metropolis-Hasting algorithm, as the algorithm does not require knowledge about the exact distribution (Andrieu et al., 2003). Metropolis-Hasting uses a accept/reject rule to converge to the posterior distribution. Allow $f(x)$ to be a function that is proportional to the exact distribution, then the algorithm is given as:

Algorithm 2: Metropolis-Hasting algorithm

Draw an initial point (x_0);
while $i = 0$ *to* N **do**
  Sample a new candidate point (x) around x_0 from a distribution $D(x, x_i)$;
  Calculate the acceptance probability as $p =$ min $(1, \frac{D(x,x_i)}{D(x_i,x)} \frac{f(x')}{f(x)})$;
  Draw k $\sim$ Bernoulli(p);
  if k **then**
    $x = x_{n+1}$;
    $i = i + 1$;
  end
end

The lack of scalability of the MCMC method makes the variational inference method more attractive. Variational inference makes use of a variational distribution $q_\chi(H)$, this distribution's parameters are sets of χ. The variational distribution infer its parameters, such that $q_\chi(H)$ is as close to the posterior distribution $P(X|D)$. The idea of Kullback-Leibler divergence, which measures how close one probability distribution is to another probability distribution can be used, this closeness is a function of χ.

The closeness, $D_{KL}(q_\chi||P(X|D))$, can be measured as:

$$D_{KL}(q_\chi||P(X|D)) = \int_H q_\chi(H)\ \log(\frac{q_\chi(H)}{P(X|D)})dH. \tag{12}$$

where the $P(X|D)$ is the posterior distribution.

By using the evidence lower bound (ELBO) the posterior distribution would not be needed. The ELBO is given as:

$$ELBO\ = \int_H q_\chi(H)log(\frac{P(H,D)}{q_\chi(H)})dH. \tag{13}$$

$$ELBO\ =\ \log(P(D)) - D_{KL}(q_\chi||P). \tag{14}$$

The ELBO maximisation is equivalent to minimizing the $D_{KL}(q_\chi||P)$.

By using MCMC or variational inference the marginal probability distribution of the output is computable, and represents the model's uncertainty. The marginal probability distribution is given as:

$$p(y|x, L) = \int_\theta p(y|x, \theta)p(\theta)d_\theta. \tag{15}$$

with x being the input feature and y being the result.

The prediction of the probability of each class can be calculated as:

$$\hat{p} = \frac{1}{|\Omega|} \sum_{\theta_i \in \Omega} NN_{\theta_i}(x). \tag{16}$$

where $NN_{\theta_i}(x)$ is the neural network function,
Ω the population and
θ_i the i_{th} sample from Ω.

The prediction of a tumour being in a certain class is calculated as:

$$\hat{y} = \arg\max_i p_i \in \hat{p}. \tag{17}$$

The MCMC will be done with the help of the Metropolis-Hasting method by means of the *MHadaptive* library written by Chivers and Chivers (2011).

3.3 Model Accuracy Measures

The model accuracy measures allows the models' performances to be evaluated and see which model performed the best. In this paper model accuracy measures were done by using the accuracy, sensitivity, specificity and Cohen's Kappa.

Table 3.3 shows a confusion matrix for binary classification, along with the true positive (+), true negative (-), false positives and false negatives.

The true positive is the number of cases which were predicted as positive, and are observed to be positive. The true negative shows the number of cases which were predicted as negative, and are observed to be negative. The false positive is the number of cases which were predicted as positive but are observed to be negative (Type I error). The false negative is the number of cases which were predicted as negative but are observed to be positive (Type II error).

The accuracy is the proportion of correct predictions in the model, or how often is the classifier correct. Sensitivity is the proportion of positive observations that have been correctly identified. The specificity is the proportion of negative observations that have been correctly identified.

Table 3.3 shows a confusion matrix for binary classification. Below are formulae for certain accuracy measures using Table 3.3, accuracy (TP + TN)/($\sum_{all}$), Sensitivity (TP)/(P_{act}) and Specificity (TN)/(N_{act}).

Table 3.3: Confusion matrix for binary classification

	Observed +	**Observed -**	**Total**
Expected +	True + (TP)	False + (FP)	+ Predicted (P_{pred})
Expected -	False - (FN)	True - (TN)	- Predicted (N_{pred})
Total	Actual + (P_{act})	Actual - (N_{act})	Size of data set ($\sum_{all}$)

The Cohen's Kappa allows for unbalanced data, and measures the agreement between an expected classifiers (expected positive and negative) and the observed values (observed positive and negative) (Cohen, 1960).

The relative observed agreement (p_0) is the proportion of correctly classified observations. The relative observed agreement is calculated as:

$$p_0 = \frac{TP + TN}{\sum_{all}}. \tag{18}$$

The probability of agreement (p_e) is the probability of random agreement. The probability of agreement is calculated as:

$$p_e = \frac{P_{pred}}{\sum_{all}} * \frac{P_{act}}{\sum_{all}} + \frac{N_{pred}}{\sum_{all}} * \frac{N_{act}}{\sum_{all}}. \tag{19}$$

From equation 18 and 19 Cohen's Kappa is calculated as:

$$\text{Cohen's Kappa } = \frac{p_0 - p_e}{1 - p_e}. \tag{20}$$

Cohen suggest thresholds that if Cohen's Kappa is ≤ 0 there is no agreement between the values. Between 0.01-0.2 there is no to slight agreement, between 0.21-0.4 there is a fair agreement, between 0.41-0.6 there is moderate agreement, between 0.61-0.8 there is substantial agreement and between 0.81 - 1 there is prefect agreement.

4 Analysis and Results

The type of tumour classification were taken from the SEER data set from the years 1973 to 2018. The data was used for analysis in this section. Data description was given in Section 3.1 of the final proposal.

4.1 Frequentist Results

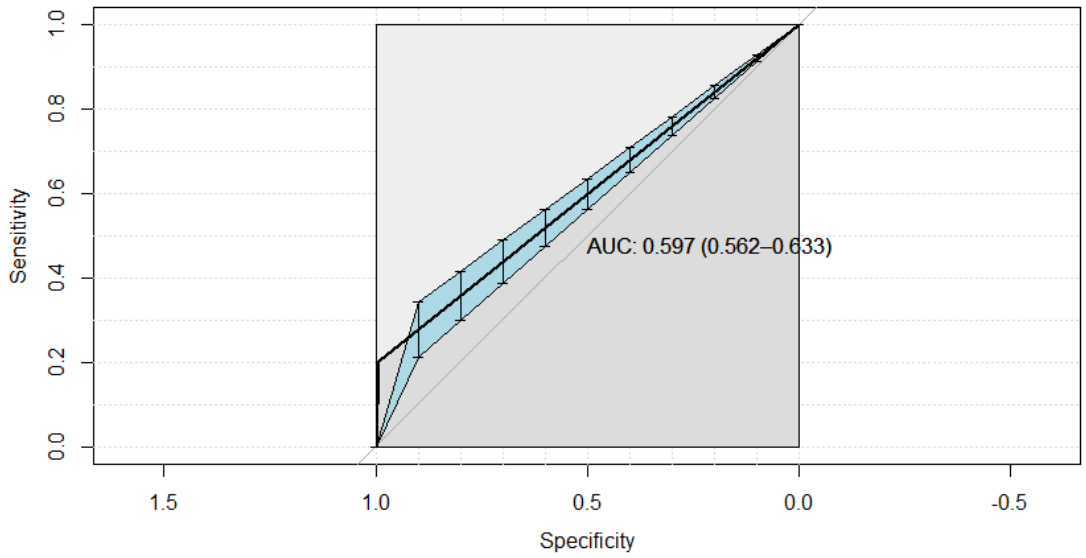
The data partition process was applied to both the Frequentist and Bayesian approaches. See Section 3.2 of the final proposal for the data partition process.

4.1.1 K-Nearest Neighbours

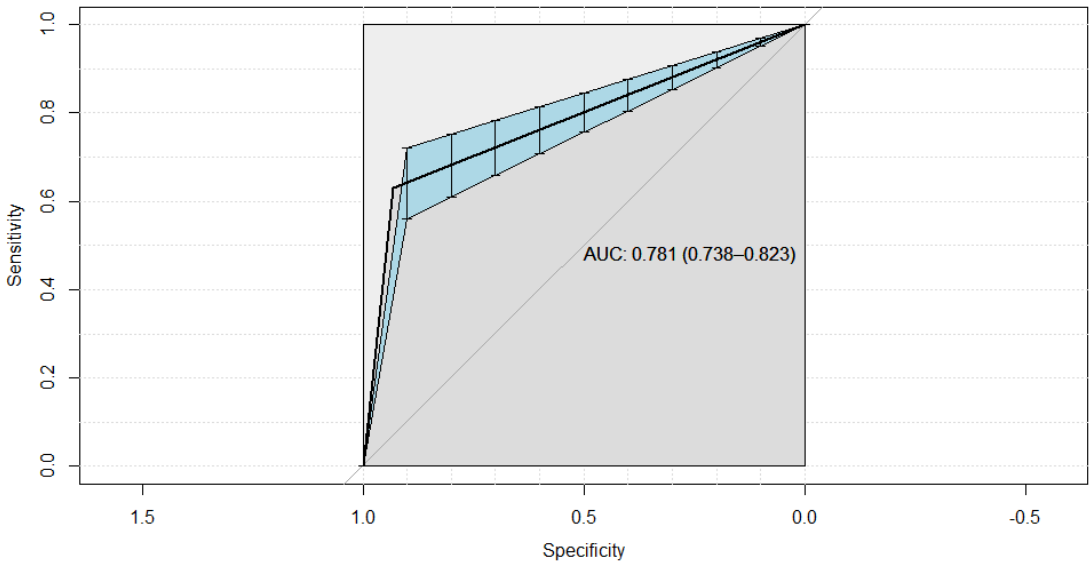
One of the objectives of the thesis was to use K-Nearest Neighbours as one of the baseline classification models. The KNN models were split between the number of neighbours and models that used and did not use SMOTE techniques. The four types of KNN models were, KNN with K=5, KNN with K= 5 and SMOTE techniques, KNN with K=10 and KNN with K=10 and SMOTE techniques. Another aim of the thesis was to compare models who use SMOTE and\or threshold optimization with models that do not use SMOTE and\or threshold optimization.

In Figure 4.1 the Receiver Operating Characteristic (ROC) curves of two models with five neibourhoods were drawn. The ROC curve plots the trade-off between sensitivity and specificity. The performance of each model was decided based on how close the curve was to the top left corner or closer to the 45° line. The closer the curve was to the top left corner of the graph, the better the model performed. The Area under curve (AUC) shows a summarized performance of how well the model can distinguish between two classes. The AUC was a summarised value of the ROC curve, which allows for better comparison between models. The AUC of 0.5 indicates a weak model, while an AUC of 1 shows the model being able to distinguish the two tumour classifications perfectly.

In Figure 4.1a the ROC is quite close to the 45° line, along with an AUC of 0.597 indicating a low performance level. While in Figure 4.1b shows the ROC being quite close to the top left corner, and having an AUC of 0.781. Judging solely on the ROC curves, the application of SMOTE to a KNN model with K=5 improved a models prediction.

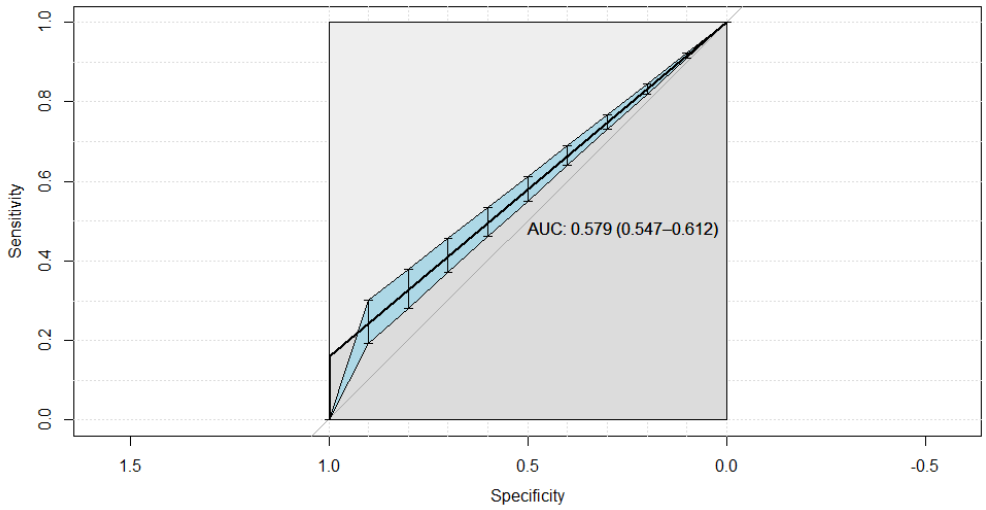


(a) KNN with K=5

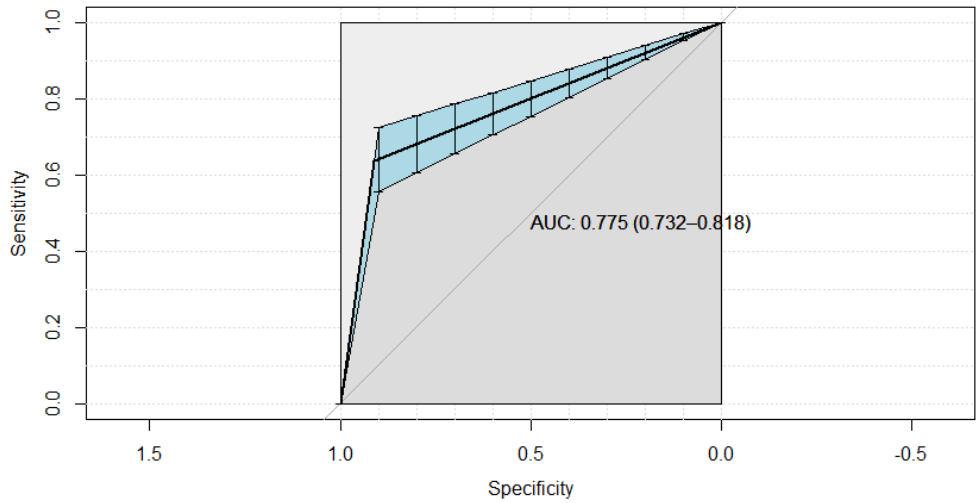


(b) KNN with K=5 and SMOTE

Figure 4.1: Figures showing KNN with K=5's ROC curve



(a) KNN with K=10



(b) KNN with K=10 and SMOTE

Figure 4.2: Figures showing KNN with K=10's ROC curve

In Figure 4.2a the AUC for the model with K=10 was 0.572, which indicated the KNN with 5 neighbourhoods without SMOTE was a better model to classify the tumours in the SEER data set. The ROC curve had a very steep rise in the beginning, however, the curve was quickly drawn towards the 45° line.

In Figure 4.2b the AUC of the KNN model with K=10 and SMOTE techniques was 0.775. The AUC value indicated a model that performed well. The ROC curve was quite close to the top left corner, thus also indicating a good model. The model showed a better performance compared to the KNN with K=10 model, but did not out perform the model with 5 neighbourhoods with SMOTE. Both models with 10 neighbourshood, appeared to perform worse then their counterparts with 5 neighbourshoods.

The confusion matrix showed how each tumour was classified by the model and what the true observed tumour type was. Table 4.1 and Table 4.2 show the confusion matrix for the KNN models with K=5 and with K=10 respectively. Both models had predicted high numbers of malignant tumours, while very low number of benign tumours. The low number of benign predictions indicated that KNN without any manipulation, such as SMOTE, can not accurately model the imbalanced data set.

SMOTE was used to modify the data for KNN models to be able to handle imbalanced data sets. Table 4.3 and Table 4.4 represent the KNN models with SMOTE techniques applied. Table 4.3 represented the K=5 model while Table 4.4 represented the K=10 model.

Table 4.1: Confusion matrix for KNN with $K = 5$

	Observed malignant	**Observed benign**
Predicted malignant	4966	99
Predicted benign	34	25

Table 4.2: Confusion matrix for KNN with $K = 10$

	Observed malignant	**Observed benign**
Predicted malignant	4991	106
Predicted benign	9	18

Table 4.3: Confusion matrix for KNN with $K = 5$ and SMOTE techniques

	Observed malignant	**Observed benign**
Predicted malignant	4660	46
Predicted benign	340	78

Table 4.4: Confusion matrix for KNN with $K = 10$ and SMOTE techniques

	Observed malignant	**Observed benign**
Predicted malignant	4563	45
Predicted benign	437	79

The KNN model with SMOTE and K=5 performed well, as there were 78 correct benign tumours predicted and only 46 incorrect benign predictions. The number of correctly predicted malignant tumours were high, but were lower than the KNN models without SMOTE. The application of SMOTE allowed the model's to gain the ability to predict benign tumours, instead of just assigning most tumours to the malignant class. The model was one of two KNN models which displayed a higher percentage in classifying the benign tumours correctly, due to SMOTE.

Table 4.4 was the confusion matrix for the KNN model with K=10 and SMOTE application. The confusion matrix indicated that 79 benign tumours were correctly classified, which was higher than the KNN model with K=5 and SMOTE application. The K=10 with SMOTE model did misclassify 97 more incorrect malignant cases compared the K=10 and SMOTE model. The 97 more misclassification of malignant tumours was not worth the trade off of getting one more correct benign tumour prediction, thus the KNN model with K=5 and SMOTE techniques was preferred.

Table 4.5: Table showing the accuracy measures for KNN methods

	$K = 5$	$K = 5$ **SMOTE**	$K = 10$	$K = 10$ **SMOTE**
Accuracy	97.4	92.46	**97.75**	90.59
Sensitivity	99.32	93.2	**99.74**	91.26
Specificity	20.16	62.9	16.13	**63.71**
Precision	98.05	**99.02**	97.96	**99.02**
F1 Score	98.68	96.02	**98.84**	94.98
Cohen's Kappa	0.97	0.92	**0.978**	0.9
AUC	0.597	**0.781**	0.572	0.775

Table 4.5 captured the performance measures discussed in Section 3.3. For all KNN models, the accuracy, sensitivity and precision were high, while the specificity was quite low for the KNN models without SMOTE. The sensitivity indicated number of correctly identified malignant cases, while the specificity indicates number of correctly identified benign cases. Hence the models without SMOTE had low accuracy when predicting benign cases, while the models with SMOTE had higher accuracy for predicting benign cases.

The accuracy for the KNN models were all above 90%, the best accuracy belongs to the KNN model with K=10. The models' accuracy decreases as SMOTE was used, however, the number of neighbourhoods did not influence the accuracy.

The sensitivity for all models were very high with the range being from 91.26% to 99.74%. The F1 score is a standardized value for the sensitivity and precision. The values for F1 score ranged from 94.98% to 98.84%. These values indicate a sight case of over-fitting in the model, signs of over-fitting is further supported by the confusion matrices Table 4.1 to 4.4 having low numbers for the predicted benign cases.

The high F1 score is desirable, however, with the specificity being very low for KNN models without SMOTE, indicates an issue of classifying benign cases. The models with SMOTE does have an increased specificity of 62.9% and 63.71% for K=5 and K=10 respectively.

The four models all have high values for the accurate measures, except for the specificity and AUC. The low AUC and specificity indicate that the KNN models do not handle class imbalance well for the SEER data set. The models without SMOTE should not be used to classify tumours as benign or malignant, as the two models will not be able to distinguish between benign and malignant cases.

The Cohen's Kappa of each model indicated values of 0.9 to 0.978 with SMOTE techniques being lower, and K=5 SMOTE model being lower than K=10 SMOTE model. According to Cohen's Kappa the four models all have almost perfect agreement. The best model judging solely by the Cohen's Kappa is the KNN model with K=10 and SMOTE techniques being applied.

The best KNN model shown is the KNN with K=5 and SMOTE techniques being applied, the choice was to ensure a high specificity, while maintaining a high AUC. The KNN model with K=10 and SMOTE had with a lower AUC, along with it not being worth to get one more benign prediction correct, while misclassifying 97 malignant cases. Hence the K=10 with SMOTE KNN model was not chosen as the best KNN model.

4.1.2 Weighted Nearest-Neighbours

The next baseline classification model was the WKNN models. The WKNN models were built by using different distance, kernel shapes, SMOTE techniques and threshold optimization. Only the best WKNN performing models are discussed in this section, for all the other models' results see the appendix. The six best WKNN models all used SMOTE and threshold optimization, however, the distance and kernels were all different. The models were split by the kernels being Epanechinkov or Triangular kernel and the distances differed by being the Minkowski, Euclidean or Manhattan distance.

Figure 4.3, 4.5 and 4.6 shows the six models' ROC curves. The ROC curves for the six models were all similar with the curve increasing rapidly in the beginning then a slow incline towards the top right corner. The AUC ranged from 0.588 to 0.671, with the Minkowski distance with Epanechinkov kernel being the lowest AUC model.

In Figure 4.3 the WKNN with Euclidean distances, but different kernels are compared. Figure 4.3 showed the WKNN model with Epanechinkov kernel, SMOTE and threshold optimization, while Figure 4.4 showed the WKNN model with Triangular kernel, SMOTE and threshold optimization. Both AUC and ROC are similar with a slight difference of 0.07 in the AUC. The ROC curve also indicated no or slight difference in the shape. These two models can not be separated by looking at the ROC curves. The Euclidean distance WKNN models showed that the kernel differences, does not make a huge difference in the model building process.

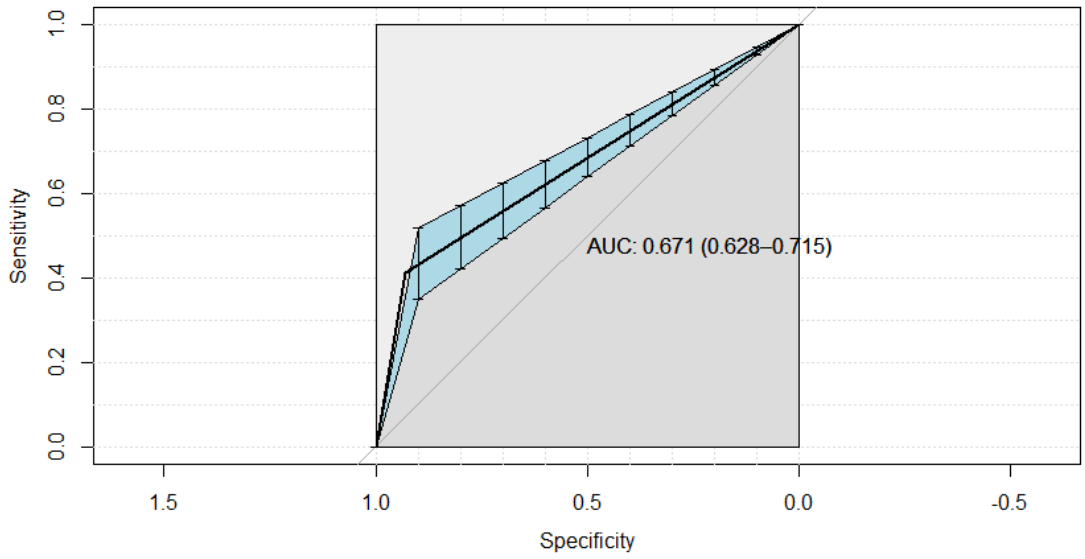


Figure 4.3: WKNN with Euclidean distance and Epanechinkov kernel

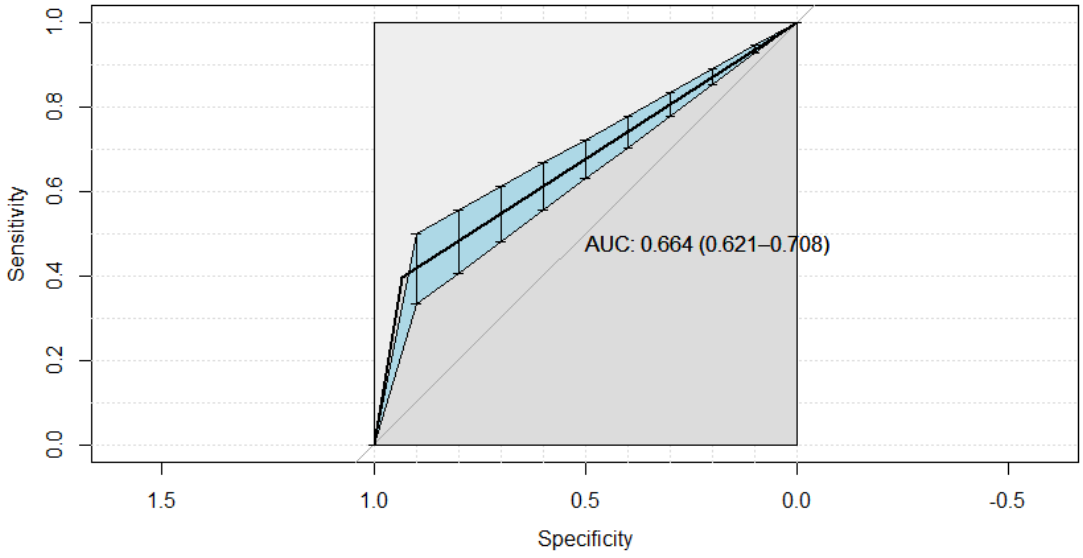
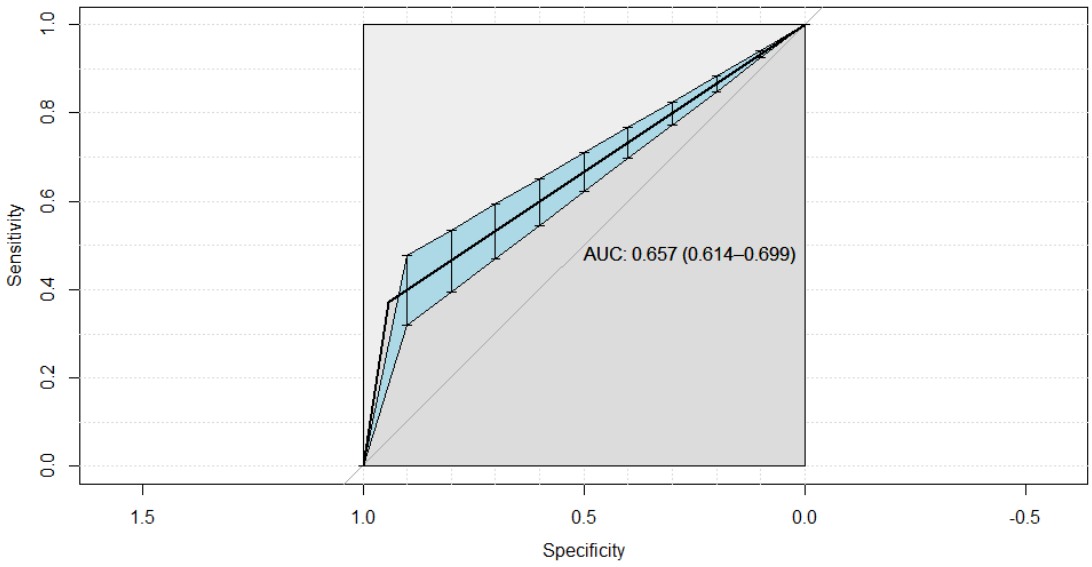
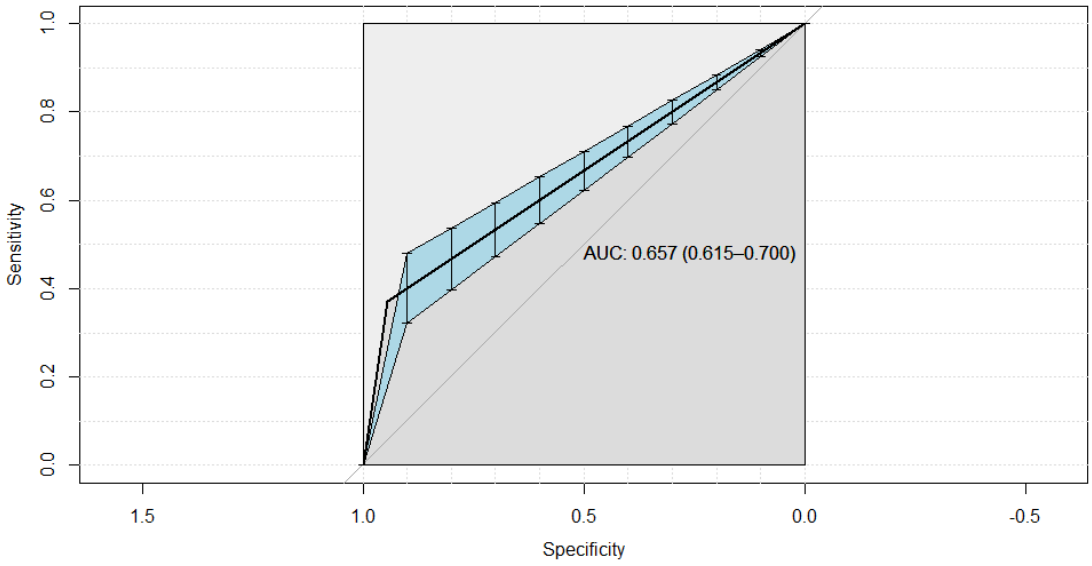


Figure 4.4: WKNN with Euclidean distance and Triangular kernel

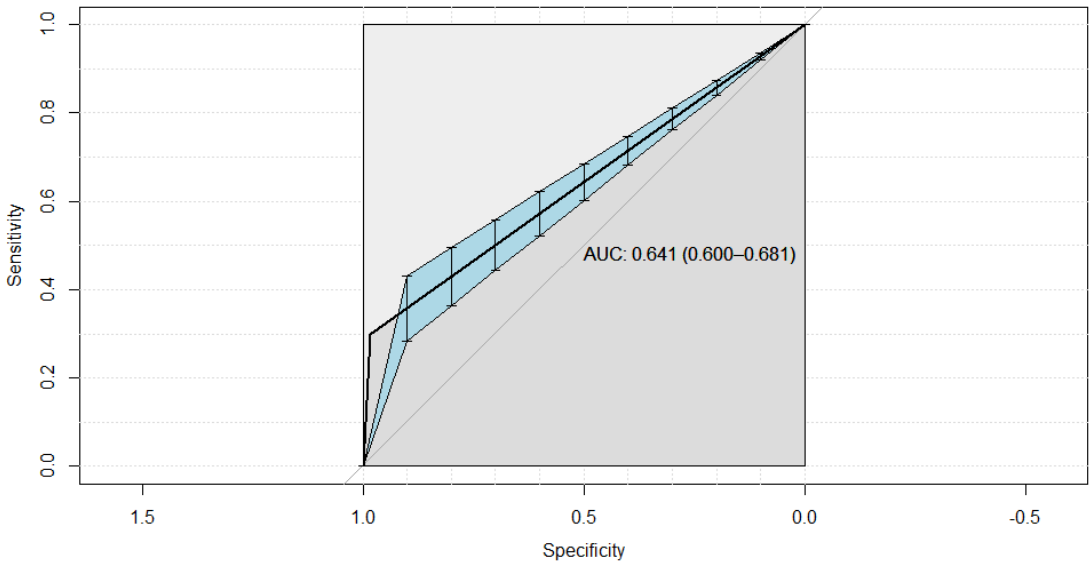


(a) WKNN with Manhattan distance and Epanechinkov kernel

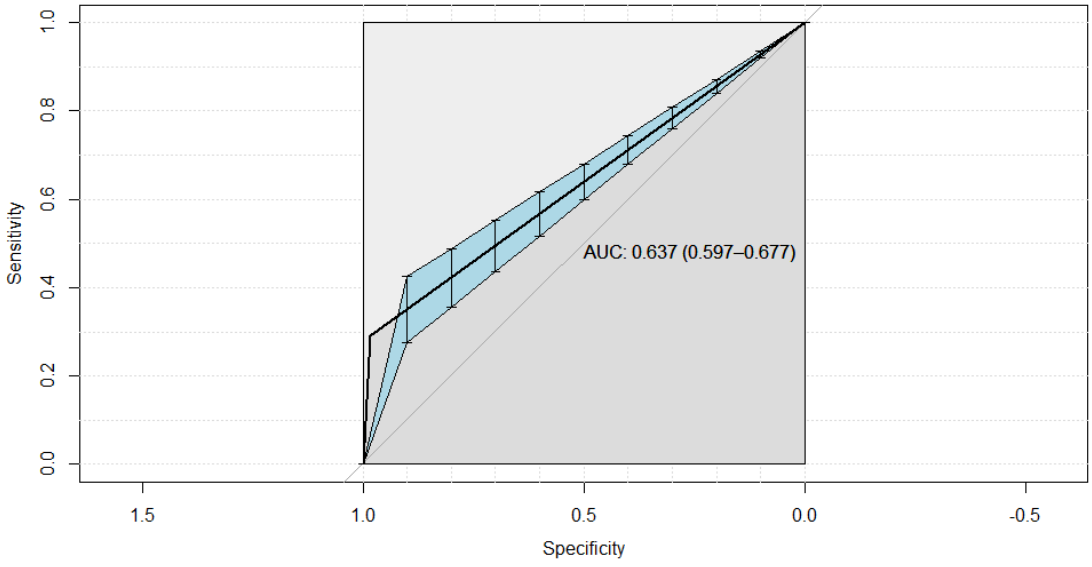


(b) WKNN with Manhattan distance and Triangular kernel

Figure 4.5: Figures showing WKNN with Manhattan distance, SMOTE and threshold optimization's ROC curve



(a) WKNN with Minkowski distance and Epanechinkov kernel



(b) WKNN with Minkowski distance and Triangular kernel

Figure 4.6: Figures showing WKNN with Minkowski distance, SMOTE and threshold optimization's AUC

The WKNN with the same Manhattan distances and Epanechinkov kernel or Triangular kernel can be seen in Figure 4.5a or Figure 4.5b respectively. The ROC curve's shape and growth indicate the two Manhattan distance WKNN models are not different or have a very slight difference when the kernel was changed. The AUC of each model further support that the models do not have a significant difference. Thus for WKNN models with Manhattan distances, the kernel does not influence the performance of the model.

The Minkowski distance WKNN had a similar behaviour as the other four WKNN models. The ROC curve for the Minkowski distances does not differ when changing the kernel type. The AUC of the Minkowski WKNN models differ by 0.004, again indicating no difference. Figure A.1, A.2 and A.3 all indicate that the kernel type does not influence the performance of the model.

Table 4.6: Confusion matrix for WKNN with Minkowski distance and Epanechinkov kernel.

	Observed malignant	**Observed benign**
Predicted malignant	4988	102
Predicted benign	12	22

Table 4.7: Confusion matrix for WKNN with Minkowski distance, Triangular kernel, SMOTE and threshold optimization.

	Observed malignant	**Observed benign**
Predicted malignant	4921	88
Predicted benign	79	36

Table 4.8: Confusion matrix for WKNN with Manhattan distance, Triangular kernel, SMOTE and threshold optimization.

	Observed malignant	**Observed benign**
Predicted malignant	4766	82
Predicted benign	234	42

Table 4.9: Confusion matrix for WKNN with Manhattan distance, Epanechinkov kernel, SMOTE and threshold optimization.

	Observed malignant	**Observed benign**
Predicted malignant	4762	81
Predicted benign	238	43

Table 4.10: Confusion matrix for WKNN with Euclidean distance, Triangular kernel, SMOTE and threshold optimization.

	Observed malignant	**Observed benign**
Predicted malignant	4709	78
Predicted benign	291	46

Table 4.11: Confusion matrix for WKNN with Euclidean distance, Epanechinkov kernel, SMOTE and threshold optimization.

	Observed malignant	**Observed benign**
Predicted malignant	4676	75
Predicted benign	324	49

Table 4.6 and 4.7 showed both models with Minkowski distance, but different kernels tend to over-fit. Both models predicted 4921 or more malignant cases with a total of 5124 cases, while not enough predictions were benign tumours. These two models should not be used for predicting tumour types, as the models could not build accurate models using the SEER data set.

The Manhattan and Euclidean distance WKNN models' confusion matrices were similar. Each model being able to predict few benign cases, however, the maximum number of accurate benign tumours being predicted were 49. These values were considerably lower than 78 and 79, see Table 4.3 and 4.4.

The performance of the WKNN models was summarised in Table 4.12. The accuracy for all six models were 92% or more. The accuracy was the highest when the Minkowski distance was used (96.74% and 97.77%), followed by the Manhattan distance (93.83% and 93.77%) and lastly the Euclidean distance's models had the lowest accuracy (92.8% and 92.21%). Therefore accurarcy is not influenced by the kernel shape. The highest accuracy was the model with Minkowski distance, Epanechinkov kernel, SMOTE and threshold optimization.

Table 4.12: Table showing the accuracy measures for WKNN models

	Min-Tri	**Man-Tri**	**Euc-Tri**	**Min-Epa**	**Man-epa**	**Euc-epa**
Accuracy	96.74	93.83	92.8	**97.77**	93.77	92.21
Sensitivity	98.42	95.32	94.18	**99.76**	95.24	93.52
Specificity	29.03	33.87	37.1	17.74	34.68	**39.52**
Precision	98.24	98.31	98.37	98	98.33	**98.42**
F1 Score	98.33	96.79	96.23	**98.87**	96.76	95.91
Cohen's Kappa	0.967	0.938	0.928	**0.98**	0.94	0.922
AUC	0.637	0.657	0.664	0.588	0.657	**0.671**

The sensitivity measured the accuracy of each model correctly predicting the malignant tumours. High sensitivity indicated that each model was influenced by the class imbalance in the data set. Sensitivity was also affected by the change of distance, the best sensitivity was from the Minkowski distance models, followed by the Manhattan distance models and lastly the Euclidean distance models. The Minkowski distance, Epanechinkov kernel, SMOTE and threshold optimization model has the highest sensitivity.

The order was reversed when the specificity was considered. The best model according to the specificity was WKNN models with Euclidean distance models, followed by the Manhattan distance models and lastly the Minkowski distance models. The specificity of WKNN models were much lower compared to the KNN models with SMOTE techniques, therefore KNN models were better at predicting benign tumours.

The F1 score reflected same the performance rankings as the accuracy. F1 score for all six models were above 95%. The high F1 scores along with low specificity further supports that the WKNN models do not handle class imbalance as well as the KNN models with SMOTE techniques.

Cohen's Kappa of each model was ranged between 0.92 to 0.98, with the Minkowski distance models having the best performance, followed by the Manhattan distance models and finally the Euclidean distance models. According to the Cohen's Kappa all of the six models have almost perfect agreement. The best model judging solely by the Cohen's Kappa is the WKNN model with Minkowski distance, with Epanechinkov kernel.

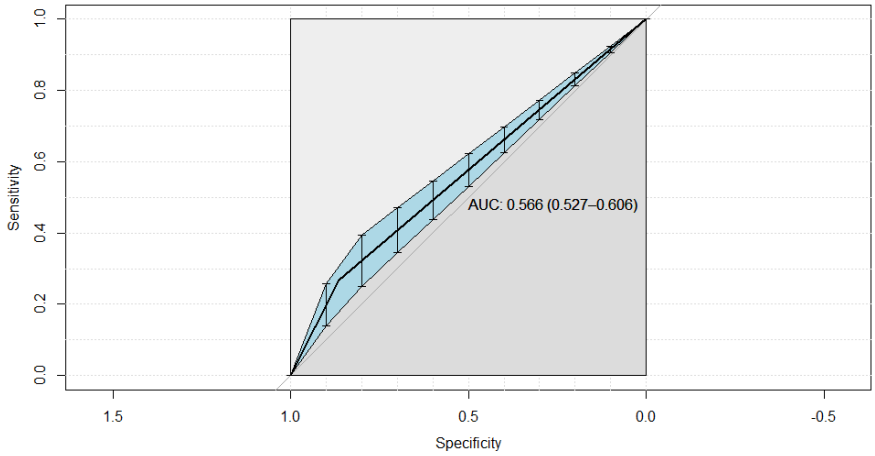
The AUC for all six models were moderately adequate, however, the Minkowski distance, with Epanechinkov kernel WKNN model was low. The AUC being low while accuracy, sensitivity, F1 score and Cohen's Kappa being high might indicate over-fitting. The best model without over-fitting was the WKNN model with Euclidean distance, Epanechinkov kernel, SMOTE and threshold optimization techniques.

4.1.3 Artificial Neural Network

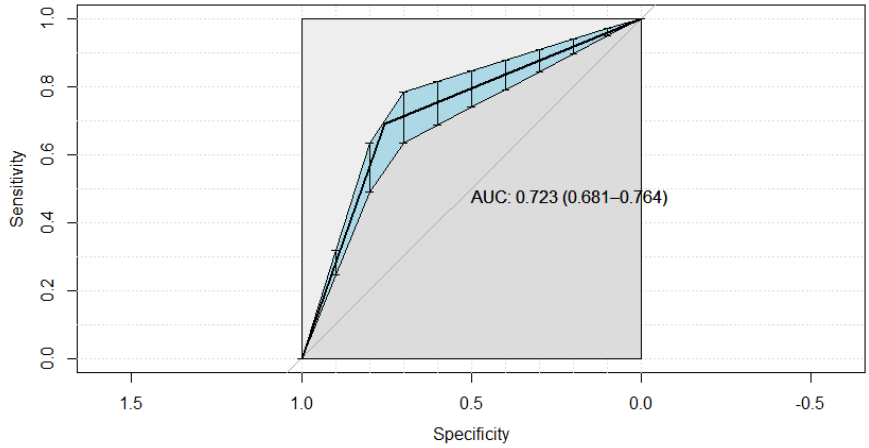
The ANN models built for the dissertation with 3 hidden layers, but differed in activation function used on the first two hidden layers, dropout layers, SMOTE and threshold optimization. In this section only the four best ANN models were discussed, the other ANN models' results can be found in the appendix. The four chosen ANN models were, ANN with ReLU activation function, ANN with Tanh activation function, ANN with Tanh activation function, dropout in the last layer, SMOTE and threshold optimization and the last model was an ANN with ReLU activation function, SMOTE and threshold optimization.

Figure 4.7a indicated an ANN model with Tanh activation function. The model's AUC of 0.566 represented a weak model. The ROC curve's shape was quite linear and close to the 45° line, thus suggesting that the model was just a little better than randomly assigning a tumour as benign or malignant.

Figure 4.7b was an ANN model with ReLU activation function. The model's performance was good, with an AUC of 0.723 and ROC curve was closer to the top left corner. According to the ROC curves the ANN model with ReLU activation function outperformed the Tanh activation function.

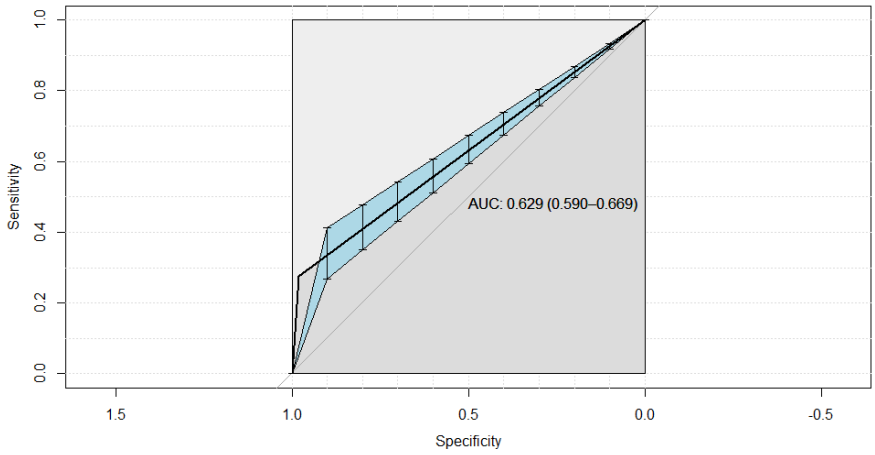


(a) ANN with Tanh activation function

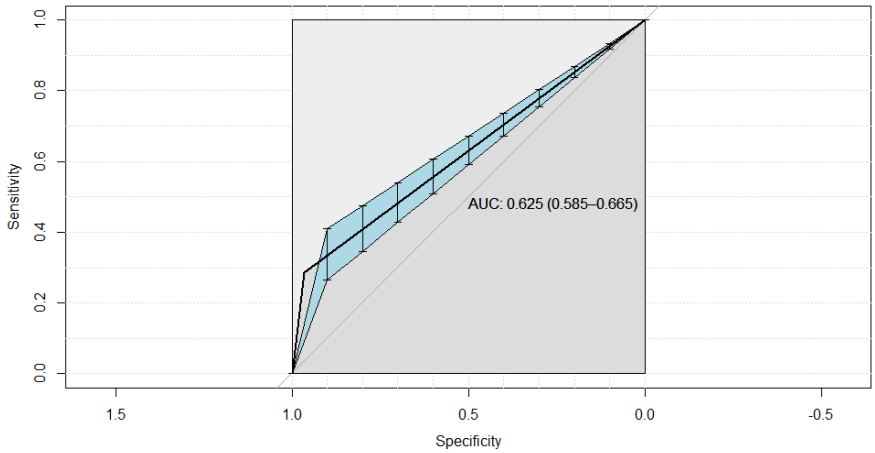


(b) ANN with ReLU activation function

Figure 4.7: Standard ANN's ROC curve



(a) ANN with Tanh activation function, dropout layers, SMOTE and threshold optimization



(b) ANN with ReLU activation function, dropout layers, SMOTE and threshold optimization's AUC

Figure 4.8: ANN with SMOTE and threshold optimization's AUC

Figure 4.8 showed an ANN with last layer dropout, SMOTE and threshold optimizations. The ROC curves for Tanh and ReLU activation functions were close. The AUC of both ANN models were also similar with only a difference of 0.004, indicating no or slight difference between the two models.

According to the ROC curves and AUC using SMOTE, threshold optimization and dropout layers made the Tanh activation function ANN perform better. The addition of SMOTE, threshold optimization and dropout layers in the last layer deteriorated the ANN model with ReLU activation function. The decrease in performance resulted in the AUC dropping from 0.723 to 0.625.

The ROC curves in Figure 4.7b and 4.8b showed the ANN with ReLU activation functions' performance were declining with addition of SMOTE, threshold optimization and last layer dropout. To understand the decline in performance Table 4.13 and 4.14 were compared.

Table 4.13 represented the ANN models with ReLU activation function. Table 4.13 showed a high number of cases being classified as benign cases with 85 correctly classified benign cases, however, there were many misclassified malignant cases. This model does not have a high indication of over-fitting.

Table 4.14 represented the ANN models with ReLU activation function with dropout layers, SMOTE and threshold optimization. The model does not predict many benign cases, with only 35 correctly classified benign cases. This model indicated a slight over-fit, hence does not perform as well as the ANN model with ReLU activation function. The over-fitting of the model was corresponding with the low AUC in Figure 4.8b.

Table 4.13: Confusion matrix for ANN with ReLU activation function.

	Observed malignant	**Observed benign**
Predicted malignant	3800	38
Predicted benign	1238	85

Table 4.14: Confusion matrix for ANN with ReLU activation function, dropout layers, SMOTE and threshold optimization

	Observed malignant	**Observed benign**
Predicted malignant	4867	88
Predicted benign	171	35

Table 4.15: Confusion matrix for ANN with Tanh activation function.

	Observed malignant	**Observed benign**
Predicted malignant	4356	90
Predicted benign	682	33

Table 4.16: Confusion matrix for ANN with Tanh activation function with threshold optimization and dropout.

	Observed malignant	**Observed benign**
Predicted malignant	4948	89
Predicted benign	90	34

The ANN with Tanh activation function showed that SMOTE, threshold optimization and last layer dropout improved the model, as improvements can seen in the AUC and ROC curves. Table 4.15 and 4.16 further supports the AUC and ROC curves' result as there were more correctly classified malignant tumours in the Tanh activation function's ANN with dropout layers, SMOTE and threshold optimization compared to the ANN with only Tanh activation function model.

The SMOTE Tanh activation function ANN still improved on the prediction of benign cases thus outperforming the other Tanh activation function ANN model. Table 4.15 shows 682 misclassified malignant cases, and 33 correctly classified benign cases. This model does not indicate over-fitting, but had many misclassified malignant cases.

Table 4.16 had 34 correctly classified benign cases, which was 1 more correctly classified benign cases compared to the ANN with Tanh activation function. SMOTE, threshold optimization and dropout layer had improve the benign classification by a little, but had drastically decrease the number of misclassified malignant cases.

Table 4.17: Table showing the accuracy measures for ANN models

	ReLU	**ReLU threshold dropout**	**Tanh**	**Tanh threshold dropout**
Accuracy	75.28	94.98	85.04	**96.53**
Sensitivity	75.43	96.61	86.46	**98.21**
Specificity	**69.11**	28.46	26.83	27.64
Precision	**99.01**	98.22	97.98	98.23
F1 Score	85.62	97.41	91.86	**98.22**
Cohen's Kappa	0.75	0.95	0.85	**0.97**
AUC	**0.72**	0.625	0.57	0.63

The accuracy of the ANN models ranged from 75.28% to 96.53%, with the lowest accuracy belong to the ANN with ReLU activation function and the highest accuracy belonging to the ANN model with Tanh activation function, last layer dropout, SMOTE and threshold optimization. The high accuracy of all models, except the ReLU activation function ANN, was due to the high sensitivity, but all three mentioned models had low specificity. The three high accuracy, but low specificity indicate that the model does not handle the majority class well.

The sensitivity of all models were high, indicating many or most malignant tumours were identified by each model, but only the ReLU activation function ANN model had less than 50 misclassified benign tumours. The high sensitivity increased the accuracy rate mentioned above, but high sensitivity along with low specificity does indicate that the model was not able to handle class imbalance data sets.

The specificity were low for all models, except the ReLU activation function ANN model. Hence the only model that was able to predict benign cases with a high accuracy was the ReLU activation function model. The other three models indicated that about 25% of the benign cases were identified, thus showing again that the models did not handle the class imbalance.

The F1 score supported the performance ranking of the accuracy metric, with Tanh with additional techniques being the highest ranking model in performance, followed by the ReLU model with additional techniques, ANN with only Tanh activation function and lastly by the ReLU activation function model. The Cohen's Kappa metric also supported the performance rankings of the accuracy and F1 score, however, these metrics were inflated by over-fitting and the class imbalance problem in the data set.

The AUC of the ANN models have the reversed performance ranking, with the ReLU activation function ANN being the highest ranked with 0.72. The other three models' specificity decreased each models' performance, hence they displayed lower AUC values. The high accuracy, sensitivity, F1 score and Cohen's Kappa for the three models along with low AUC and specificity indicate that there was over-fitting occurring, which was due to the class imbalance data set. The ANN's best model was the ReLU activation function model, as the model had a high rate of specificity as well as a high AUC, while still maintaining a sensitivity of 75.43%.

4.2 Bayesian Models

The Bayesian models were split into Naive Bayes models and BNN models.

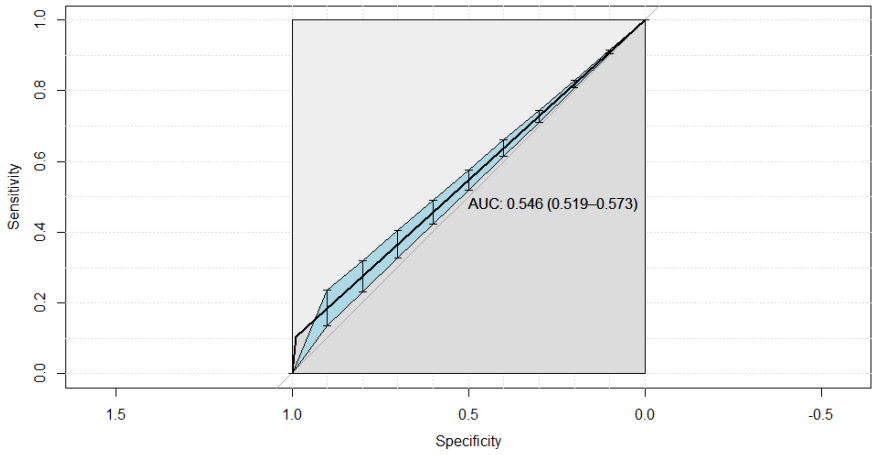
4.2.1 Naive Bayes Classifier

The NB classifier had three models, a NB classifier without SMOTE or threshold optimization, NB with SMOTE and NB with SMOTE and threshold optimization. One of the aims of the thesis was to compare the NB models who used SMOTE and\or threshold optimization with other NB models without SMOTE and\or threshold optimization, and was completed in this section.

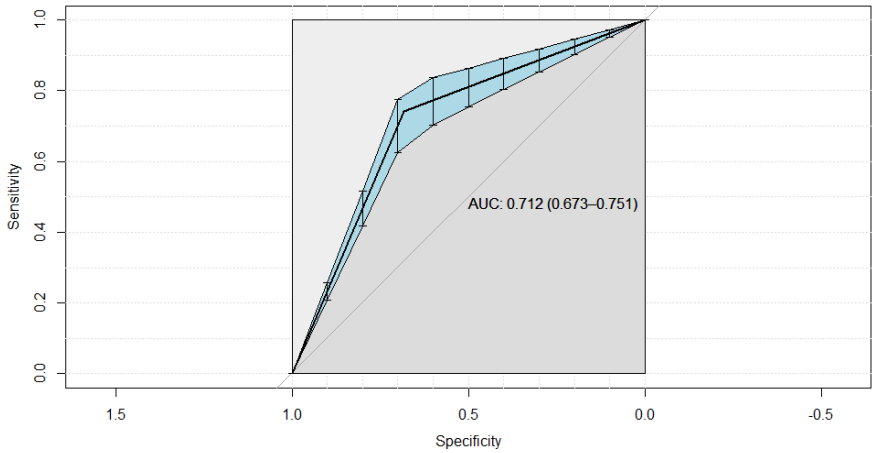
Figure 4.9a displayed a ROC curve for the NB without any SMOTE or threshold optimization. The model's performance was weak, as a result the ROC curve hugging the 45° line. The AUC of 0.546 showed that the model was a little better than randomly assigning the classification of the tumour, but still a weak model. The NB ROC curve might indicate that NB without any manipulation may be too weak in performance and can not distinguish between the two types of tumours.

Figure 4.9b was a NB model with SMOTE technique being applied. With the addition of SMOTE to the NB classifier, the model's performance has drastically outperformed the standard NB model. The ROC curve shows that the model performs well, as the curve leans towards the top left corner. The AUC of 0.712 further supports the idea of an improved model compared to the standard NB model.

Figure 4.10 indicated a NB model with SMOTE and threshold optimization. The third NB model did outperform the the standard NB model, however, it did not outperform the NB model with only SMOTE applied. The ROC curve of the NB model with SMOTE and threshold optimization did rise in the beginning, and later declined towards the 45° line. The AUC of 0.624 indicated an adequate model, but the lower performance after applying threshold optimization might indicate a slight over-fit.

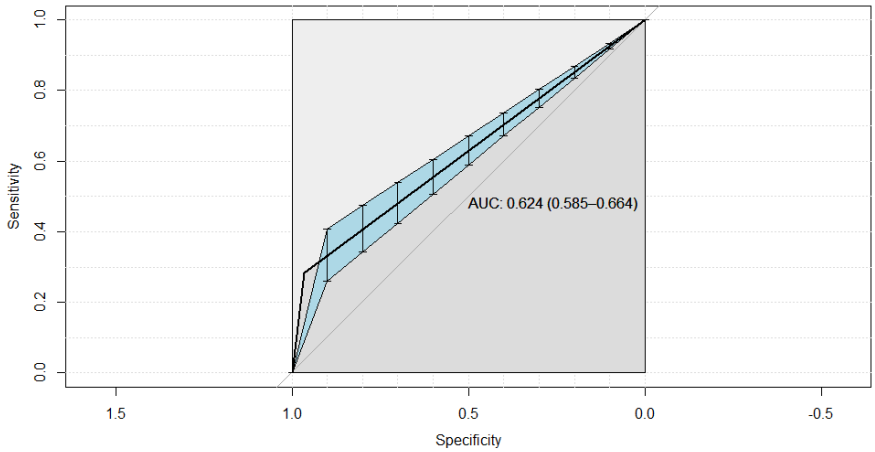


(a) Standard Naive Bayes' ROC



(b) Naive Bayes with SMOTE techniques

Figure 4.9: Naive Bayes' ROC



(a) Naive Bayes with SMOTE and threshold optimization

Figure 4.10: Naive Bayes' ROC

Table 4.18 represented the NB confusion matrix. The confusion matrix indicated 13 benign tumours were correctly identified, while 111 benign tumours were identified incorrectly. The majority of predictions were malignant, thus indicating that the model did not handle the class imbalance well for the SEER data set. The poor performance of the model may be due to the standard NB model not being strong enough.

Table 4.19 showed the confusion matrix for the NB model with SMOTE techniques. The model did correctly identify 92 benign cases while also making a lower number of inaccurate prediction for the benign cases. The addition of SMOTE allowed the NB model to perform

Table 4.18: Confusion matrix for Naive Bayes.

	Observed malignant	**Observed benign**
Predicted malignant	4939	111
Predicted benign	61	13

Table 4.19: Confusion matrix for Naive Bayes with SMOTE manipulation.

	Observed malignant	**Observed benign**
Predicted malignant	3412	32
Predicted benign	1588	92

Table 4.20: Confusion matrix for Naive Bayes with SMOTE and threshold optimization techniques.

	Observed malignant	**Observed benign**
Predicted malignant	4833	89
Predicted benign	167	35

Table 4.21: Table showing the accuracy measures for Naive Bayes models

	Naive Bayes	**Naive Bayes SMOTE**	**Naive Bayes Threshold**
Accuracy	**96.64**	68.38	95
Sensitivity	**98.78**	68.24	96.66
Specificity	10.48	**74.19**	28.23
Precision	98.29	**99.07**	98.19
F1 Score	**98.29**	80.81	97.41
Cohen's Kappa	**0.966**	0.68	0.95
AUC	0.546	**0.712**	0.624

Table 4.20 was the confusion matrix for the NB model with SMOTE and threshold optimization present. The number of correctly predicted benign tumours (35) were lower than the NB with SMOTE, but higher than the standard NB model. The addition of threshold optimization might introduce some over-fitting, as many of the observations fall under the predicted malignant cases.

Table 4.21 summarised the performance metrics of the NB models. The standard NB and the NB with SMOTE and threshold optimization have high accuracy. The accuracy of 95% or more indicated a slight concern, as the high value indicated over-fitting. The over-fitting was supported when looking at both their respective confusion matrices. The accuracy of 68.23% for the NB with SMOTE also indicated a concern, as the low value suggested that the model could be not always predict malignant tumours.

The high sensitivity and low specificity for the NB and NB with SMOTE and threshold optimization both indicate an over-fitted model. These two models do not handle the class imbalanced data set, instead both models simply classify many predictions as malignant. The NB with SMOTE techniques had a lower sensitivity compared to the specificity, indicating that the model is better at predicting benign tumours than malignant tumours.

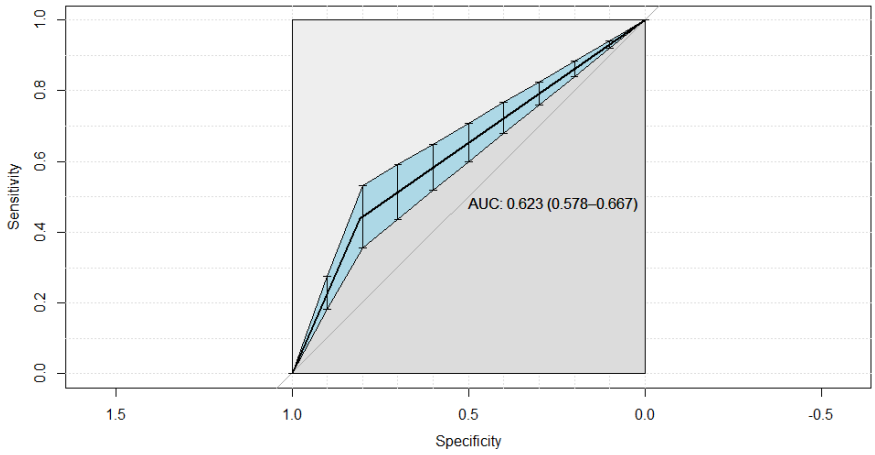
The performance ranking for accuracy, sensitivity, F1 score and Cohen's Kappa was that NB performed the best, followed by NB with SMOTE and threshold optimization and lastly the NB model with SMOTE. The ranking for the specificity and AUC were reversed, with the AUC of 0.712 for NB with SMOTE. The best NB model would be the NB with SMOTE, as the model does not suggest over-fitting, but the accuracy was low for the model.

4.2.2 Bayesian Neural Network

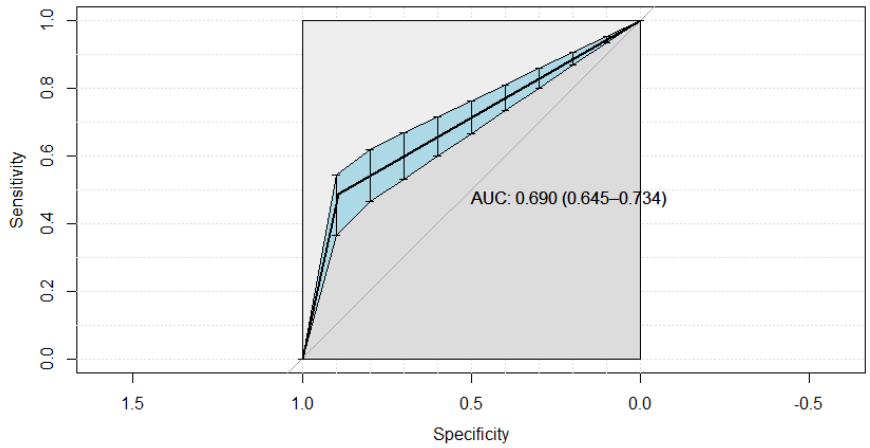
In this section only the best four BNN models were discussed. The best BNN all used SMOTE and threshold optimization in the model building. In this section SMOTE and threshold were not mentioned for convenience, but were used in all models discussed.

Figure 4.11a was the ROC curve for the BNN model with MCMC and Tanh activation function. ROC curve did not deviate too far from the 45° line, but did increase steadily in the beginning. The curve slowly regressed back to the 45° line. The AUC of 0.623 showed that the model performed adequately.

Figure 4.11b was the ROC curve for the BNN model with MCMC and ReLU activation function. The ROC curve had a steeper rise in the beginning compared to the Tanh model. The ROC curve does suggest a better performance compared to the MCMC BNN model with Tanh activation function. The AUC value for the ReLU model is also 0.05 more than the Tanh model. The ROC for the two MCMC curves suggest that the ReLU activation function outperformed the Tanh activation function model.

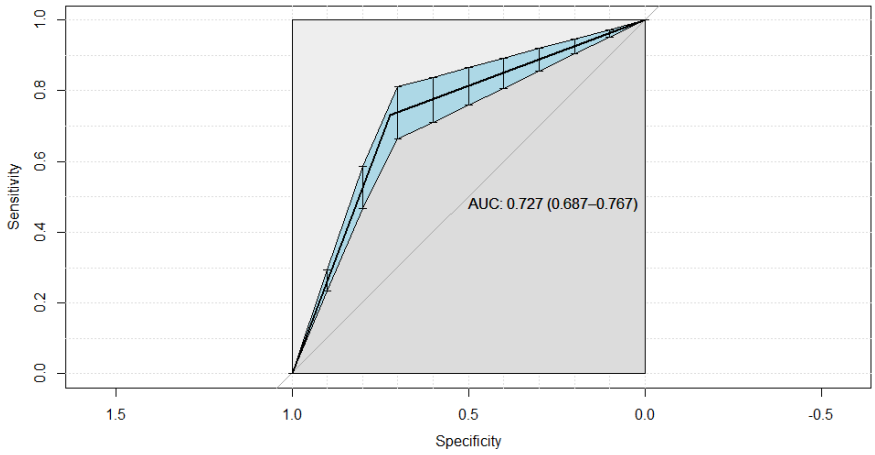


(a) BNN with MCMC, Tanh activation function, SMOTE and threshold

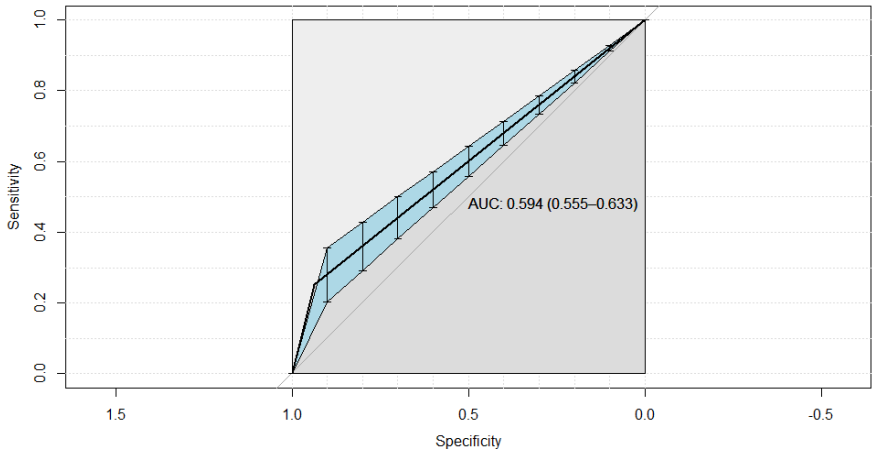


(b) BNN with MCMC, ReLU activation function, SMOTE and threshold

Figure 4.11: BNN with SMOTE and threshold optimization's ROC curve



(a) BNN with Variation inference, Tanh activation function, SMOTE and threshold optimization



(b) BNN with Variation inference, ReLU activation function, SMOTE and threshold optimization

Figure 4.12: BNN with SMOTE and threshold optimization's ROC curve

Figure 4.12a showed a BNN model with Variational inference and Tanh activation function. The ROC curve seems to be the best out of all BNN models, as the curve is strongly attracted to the top left corner. The AUC of 0.727 suggest that the Variational inference and Tanh activation function BNN model can distinguish between malignant and benign tumours quite well.

Figure 4.12b was the ROC curve for a BNN model with Variation inference and ReLU activation function. The model did not outperform any previously mentioned BNN models, with only an AUC of 0.594. The ROC curve for the Variation inference and ReLU activation function BNN model did not move toward the top left hand corner of the graph, instead was more attracted to the 45° line. This model might represent some over-fitting.

The confusion matrix for the BNN model with MCMC and ReLU activation function can be found in Table 4.22. The model predicted 54 correct benign cases, with 1028 number of predicted benign cases. The model does not show over-fitting, hence the model can somewhat handle the class imbalance.

Table 4.22: Confusion matrix for BNN (MCMC Tanh) with threshold and SMOTE manipulation.

	Observed malignant	**Observed benign**
Predicted malignant	4064	69
Predicted benign	974	54

Table 4.23: Confusion matrix for BNN (MCMC ReLU) with threshold and SMOTE manipulation.

	Observed malignant	**Observed benign**
Predicted malignant	4494	63
Predicted benign	544	60

Table 4.24: Confusion matrix for BNN (Var Tanh) with threshold and SMOTE manipulation.

	Observed malignant	**Observed benign**
Predicted malignant	3637	33
Predicted benign	1401	90

Table 4.25: Confusion matrix for BNN (Var ReLU) with threshold and SMOTE manipulation.

	Observed malignant	**Observed benign**
Predicted malignant	4717	92
Predicted benign	321	31

Table 4.23 was the confusion matrix for the BNN model with MCMC and ReLU activation function. The model did predict less benign predictions (604) compared to the MCMC and Tanh activation function model, but was able to predict more correct malignant and benign cases. Table 4.22 and 4.23 suggested that the BNN model with ReLU activation function was able to perform better than the Tanh activation MCMC model. The result was also highlighted in the ROC curves.

Table 4.24 was the confusion matrix for the BNN model with Variation inference and with Tanh activation function. This model had the lowest amount of correctly predicted malignant cases (3637), but did have the highest number of correctly predicted benign cases. The model did not indicate over-fitting as there were 1491 number of benign prediction, but there were many malignant cases that were wrongfully predicted.

Table 4.25 was the confusion matrix for the BNN model with Variation inference and with ReLU activation function. This model had the lowest number of correct prediction for benign cases, and only predicted 352 number of benign cases. The low number of prediction for the benign cases indicate a possibility of over-fitting in the model.

Considering all BNN confusion matrix mentioned, the only model which suggested over-fitting was the BNN model with Variation inference and with ReLU activation function. Hence was determined as the worse model for BNN models using the ROC curves and the confusion matrices. The best BNN model was harder to distinguish using the confusion matrices.

Table 4.26 summarised the performance metric of the BNN models with SMOTE and threshold optimization. The MCMC techniques seem to have little differences between each metric, such as the accuracy difference being 8%. The MCMC models suggested that MCMC BNN models are more consistent, compared to the Variational inference models which had a difference of 19.79%.

The accuracy of each model was shown at a good percentage, with the exception of the Variation inference with ReLU activation function at 92%. The high accuracy might indicate that the model may only be able to predict malignant cases, and can not handle the class imbalance presented in the data set.

The performance ranking by sensitivity, precision, F1 score and Cohen's Kappa metric all suggested that the Variation inference with ReLU model was the best, followed by the MCMC with ReLU activation function, Tanh activiation function MCMC model and lastly by the Variation inference with Tanh activation function. These metrics suggest that the ReLU activation function out performed their counterpart Tanh activation function models, however, the height in the mention performance metrics does not mean the model can predict the benign cases.

The only model with higher than 50% specificity was the Variation interference with Tanh activation function, thus suggesting that this model would be able to classify 72% of the benign cases correctly. This model also had the highest AUC of the BNN models, with the AUC being 0.727. The model further has a good sensivitivity of 72%, which was appromiately the same percentage as the specificity. The Variation inference model with Tanh activation function can, therefore correctly classify 72% or more cases, regardless if the actual class was benign or malignant. The best BNN model was therefore the Variation interference with Tanh activation function.

Table 4.26: Table showing the accuracy measures for BNN models

	MCMC ReLU	**MCMC Tanh**	**Var inf ReLU**	**Var inf Tanh**
Accuracy	88.24	79.79	**92**	72.21
Sensitivity	89.2	80.67	**93.63**	72.19
Specificity	48.78	43.9	25.2	**73.17**
Precision	**99.62**	98.33	98.09	99.1
F1 Score	93.67	88.63	**95.82**	83.53
Cohen's Kappa	0.88	0.8	**0.9581**	0.72
AUC	0.69	0.623	0.594	**0.727**

4.3 Comparison table of best models

Four of the overall best models were discussed in this section. The best four models were the KNN with K=5 and SMOTE techniques, ANN with ReLU activation function, NB with SMOTE and the Variation inference BNN model with Tanh activation function, SMOTE and threshold optimization.

The best model was judged based on if the model's sensitivity and specificity were equal, other considerations was to focus on the AUC and the impact of using the model. All four models were already checked for over-fitting and if the model could handle the class imbalance in the data set. The one notable result showed that the Bayesian models provided a high specificity, while maintain decent metric across the table.

Four of the overall best models were discussed in this section. The best four models were the KNN with K=5 and SMOTE techniques, ANN with ReLU activation function, NB with SMOTE and the Variation inference BNN model with Tanh activation function, SMOTE and threshold optimization.

The best model was judged based on if the model's sensitivity and specificity were equal, other considerations was to focus on the AUC and the impact of using the model. All four models were already checked for over-fitting and if the model could handle the class imbalance in the data set. The one notable result showed that the Bayesian models provided a high specificity, while maintain decent metric across the table.

Table 4.27: Table showing the accuracy measures for best models

	K=5 SMOTE	**ReLU ANN**	**Naive Bayes SMOTE**	**Var inf Tanh**
Accuracy	**92..46**	75.28	68.38	72.21
Sensitivity	**93.2**	75.43	68.24	72.19
Specificity	62.9	69.11	**74.19**	73.17
Precision	99.02	99.01	99.07	**99.1**
F1 Score	**96.02**	85.62	80.81	83.53
Cohen's Kappa	**0.92**	0.75	0.68	0.72
AUC	**0.781**	0.72	0.712	0.727

The process of judging for the best model, was to first eliminate models who had flaws or undesirable characteristics. The first model eliminated was the NB model with SMOTE techniques. The model was eliminated from the process as the sensitivity was 68% suggesting that there would be an error of 32% in classification of malignant cases. The model's sensitivity, AUC and Cohen's Kappa was also lowest compared to the other three models.

The other three models were harder to distinguish. The neural network models presented similar values with the Bayesian model presenting higher specificity, but lower sensitivity. The values of specificity and sensitivity for both models were around 70%. The AUC of both models differed by only 0.007, thus suggesting that both models are similar in performance. The choice of the model would depend on the doctor. The doctor could use the BNN model if the test was concentrated on whether a patient is benign or not, while the use of the ANN model would depend on if the tumour was tested for malignant tumours.

The last comparison is to compare the KNN models with the ANN model. The comparison was focused on the ANN, because if the doctor wanted to test for benign tumours, the doctor would use the BNN model. If the doctor wanted to test for malignant tumours, then either the ANN model or KNN model should be considered. The choice of KNN would be favorable when testing for malignant tumours, as all metrics except for the specificity was lower than the ANN model.

Considering the best models, the KNN model with K=5 and SMOTE should be used for testing malignant tumours. The BNN model with Variational inference, Tanh activation function, SMOTE and threshold optimization should be considered when testing for benign tumours.

4.4 Discussion

From a statistical view point the best model to use was the KNN model with K=5 and SMOTE techniques or the BNN model with Variational inference, Tanh activation function, SMOTE and threshold optimization. The discussion in this section was to highlight the other practical views of using the models.

Table 4.28 was the confusion matrix for the best BNN model, in the matrix 1 401 out of 5 161 people were misclassifed as being benign cases while actually being malignant. Al-Azri and Mohammed (2016) state that if certain types of cancer were to be detected earlier, the chances of the patient being cured is higher. Hence the model being used for cancer detection is every important. The model should therefore try to minimize the number of misclassified malignant cases.

Table 4.29 was the confusion matrix for the KNN model with K=5 and SMOTE techniques applied. The number of misclassified malignant tumours is significantly less compared the Table 4.28. From a statistical and practical viewpoint the best model to use when diagnosing cancer types is the KNN model with K=5 and SMOTE techniques.

Table 4.28: Confusion matrix for BNN (Var Tanh) with threshold and SMOTE manipulation.

	Observed malignant	**Observed benign**
Predicted malignant	3637	33
Predicted benign	1401	90

Table 4.29: Confusion matrix for KNN with $K = 5$ and SMOTE techniques

	Observed malignant	**Observed benign**
Predicted malignant	4660	46
Predicted benign	340	78

5 Conclusion

5.1 Synthesis of analysis and conclusion

Tumours were classified using two types of models, Bayesian and frequentist models. The frequentist models included KNN, WKNN and ANN models, while the Bayesian models included Naive Bayes and BNN models. The best frequentist model was the KNN model with SMOTE application and $K = 5$, which had an AUC of 0.781. The WKNN models did not address the imbalance within the data, as the specificity of all the best WKNN models were lower than 40%, while displaying 90% or more for each models' accuracy. The ANN models, provided varying results as the specificity ranged between 26% to 69%. The stand out result of the ANN models was the ReLU activation model, however, the KNN model with SMOTE application outperformed the ANN model.

The second class of models used for classification of tumours were the Bayesian models. The two Bayesian models were further split into Naive Bayes models and BNN models. Both types of models had improved specificity, however, there was a drop in accuracy for all Bayesian models. The best performing Bayesian model was the Variational inference BNN model with Tanh activation function, with an AUC of 0.727.

The frequentist models were able to classify malignant tumours, however, the models could not predict benign tumours well. The use of SMOTE allowed frequentist models to improve classifying benign tumours, however, the Bayesian models predicted benign tumours better. The prediction of malignant tumours and benign tumours were both of importance, and the best classification model was the KNN model with SMOTE and $K = 5$.

The classification models allows for a faster and more accurate diagnosis. The models built may reduce errors made by the doctors, and reduce the amount of time used between primary care interval. The models should be used to help in decision making and not be used to diagnosis a patient, given the misclassification risk that still remains.

5.2 Limitations of the study

The ANN and BNN model only used up to three hidden layers and contained limited number of activation functions. The BNN models could include more layers that used Bayesian inference methods, instead of only the last layer contain Bayesian methods.

5.3 Recommendations for future work

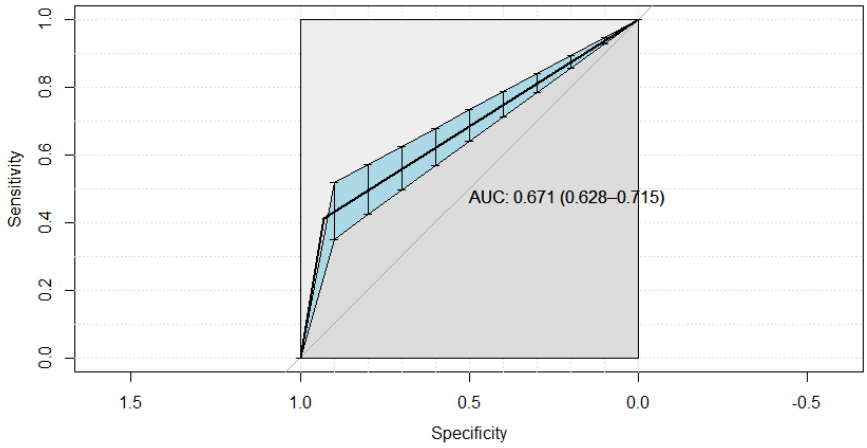
The models built were based on the data sets, which did not include scanned imaging of tumours. As future work for classifying tumours different models which use images may be used. Other models such as Convolutional Neural Networks, Decision Trees or Support vector machines and generative adversarial networks should also be considered.

Appendix

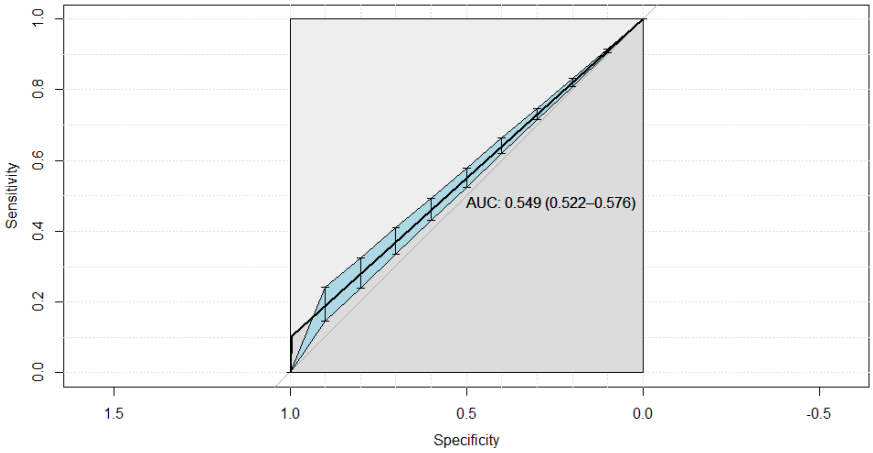
A Some Results

A.1 Weighted Nearest-Neighbours

A.1.1 ROC Curves

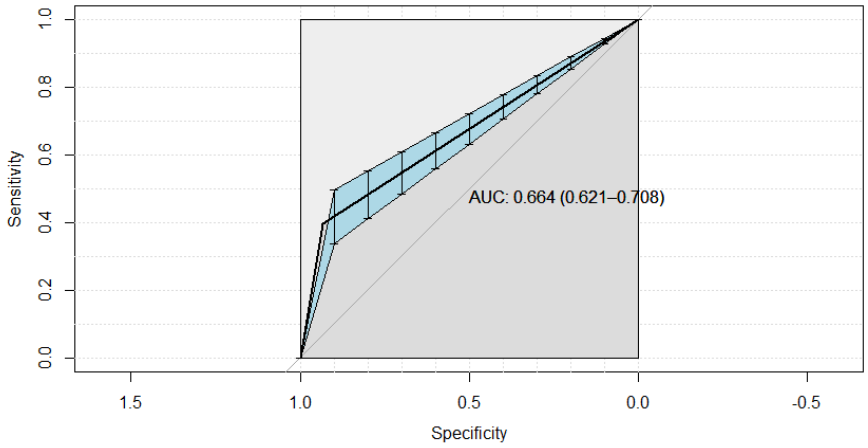


(a) WKNN with Euclidean distance, Epanechinkov kernel and SMOTE' ROC

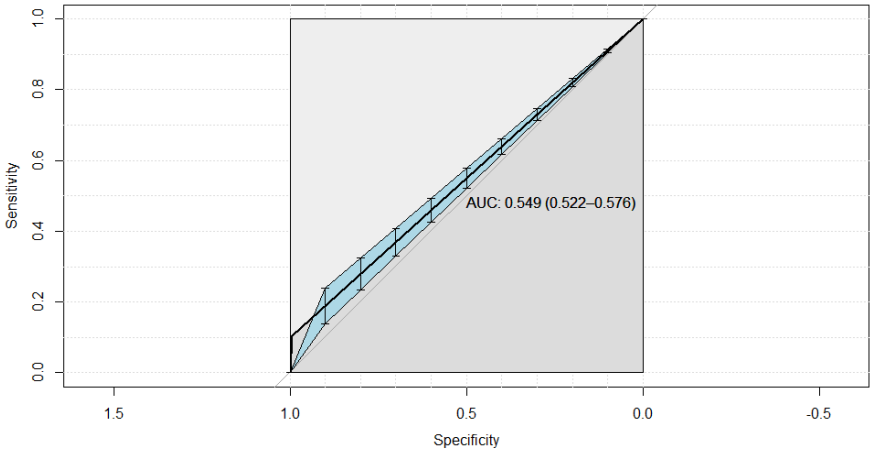


(b) WKNN with Euclidean distance and Epanechinkov kernel's ROC

Figure A.1: WKNN with Euclidean distance and Epanechinkov kernel's ROC curve

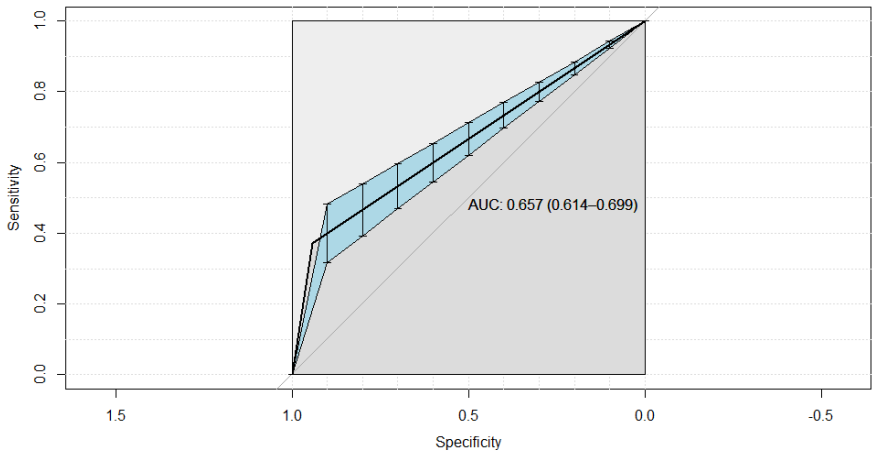


(a) WKNN with Euclidean distance, Triangular kernel and SMOTE' ROC curve

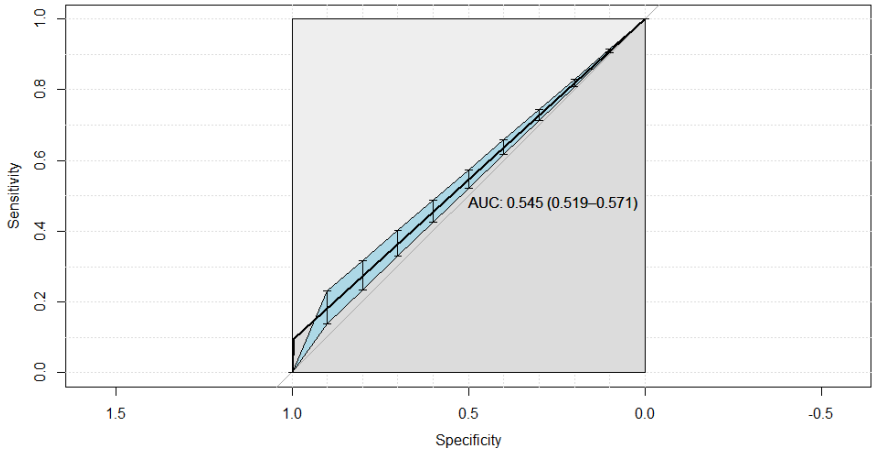


(b) WKNN with Euclidean distance and Triangular kernel's ROC curve

Figure A.2: WKNN with Euclidean distance and Triangular kernel's ROC curve

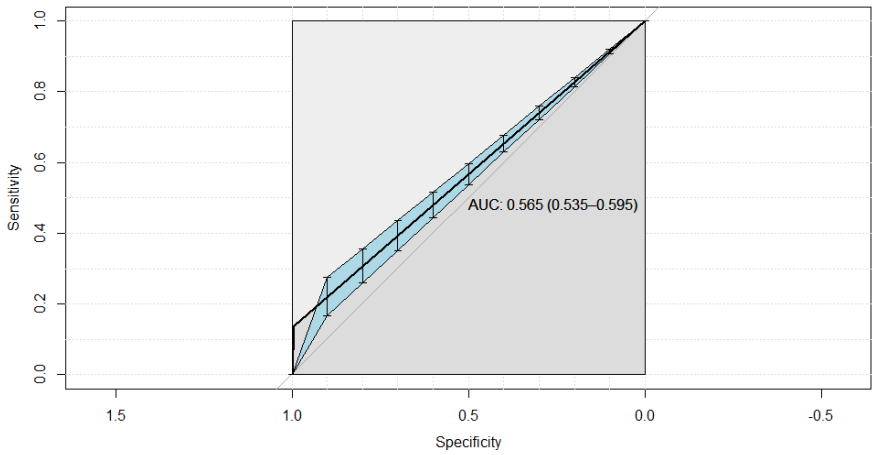


(a) WKNN with Manhattan distance, Epanechinkov kernel and SMOTE' ROC

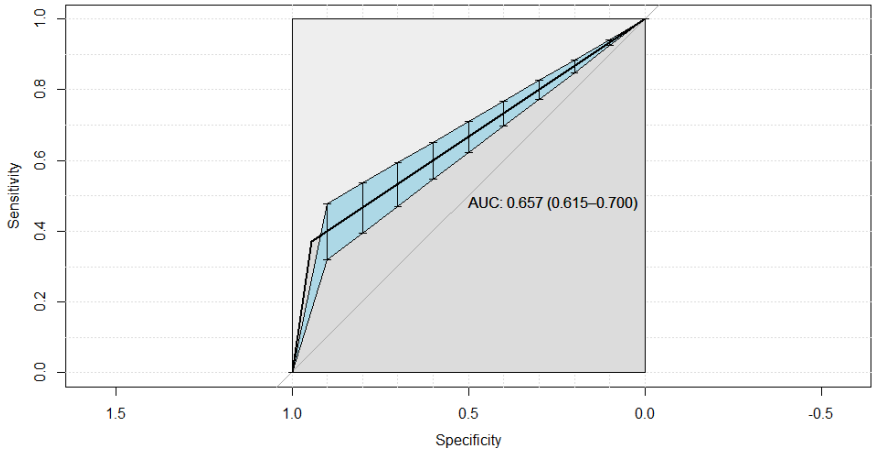


(b) WKNN with Manhattan distance and Epanechinkov kernel 's ROC

Figure A.3: WKNN with Manhattan distance and Epanechinkov kernel 's ROC curve

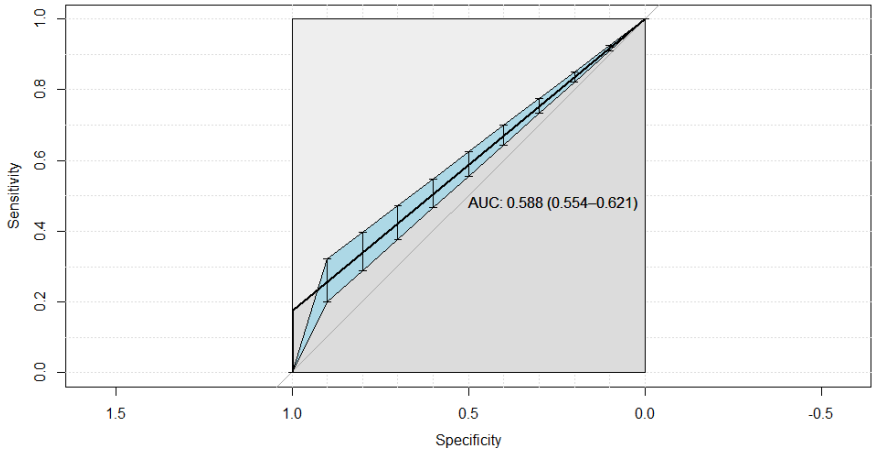


(a) WKNN with Manhattan distance and Triangular kernel ROC curve

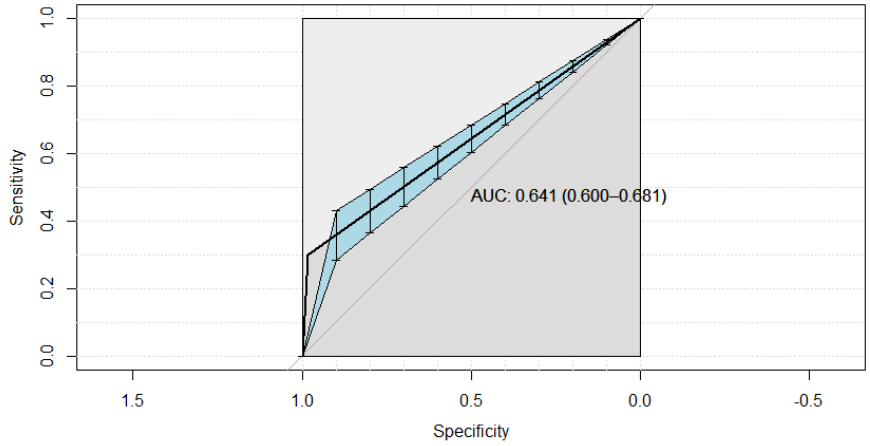


(b) WKNN with Manhattan distance, Triangular kernel and SMOTE's ROC curve

Figure A.4: WKNN with Manhattan distance, Triangular kernel 's ROC curve



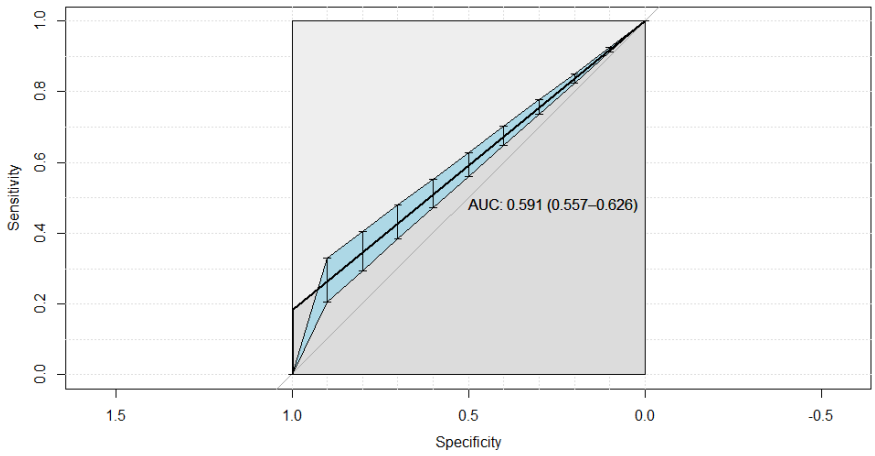
(a) WKNN with Minkowski distance and Epanechinkov kernel ROC



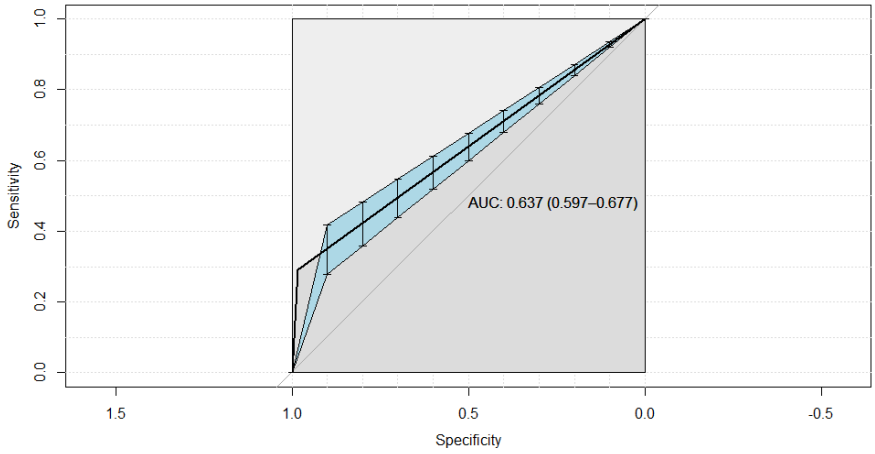
(b) WKNN with Minkowski distance, Epanechinkov kernel and SMOTE's ROC

Figure A.5: WKNN with Minkowski distance, Epanechinkov kernel 's ROC curve

A.1.2 Confusion matrix



(a) WKNN with Minkowski distance and Triangular kernel ROC curve



(b) WKNN with Minkowski distance, Triangular kernel and SMOTE's ROC curve

Figure A.6: WKNN with Minkowski distance, Triangular kernel 's ROC curve

Table A.1: Confusion matrix for WKNN Euclidean distance, Epanechinkov kernel and SMOTE

	Observed malignant	**Observed benign**
Predicted malignant	4656	73
Predicted benign	344	51

Table A.2: Confusion matrix for WKNN Euclidean distance and Epanechinkov kernel

	Observed malignant	**Observed benign**
Predicted malignant	4969	111
Predicted benign	31	13

Table A.3: Confusion matrix for WKNN Euclidean distance, Triangular kernel and SMOTE

	Observed malignant	**Observed benign**
Predicted malignant	4668	75
Predicted benign	332	49

Table A.4: Confusion matrix for WKNN Euclidean distance and Triangular kernel

	Observed malignant	**Observed benign**
Predicted malignant	4964	111
Predicted benign	36	13

Table A.5: Confusion matrix for WKNN Manhattan distance, Epanechinkov kernel and SMOTE

	Observed malignant	**Observed benign**
Predicted malignant	4711	78
Predicted benign	289	46

Table A.6: Confusion matrix for WKNN Manhattan distance and Epanechinkov kernel

	Observed malignant	**Observed benign**
Predicted malignant	4964	112
Predicted benign	36	12

Table A.7: Confusion matrix for WKNN Minkowski distance and Triangular kernel

	Observed malignant	**Observed benign**
Predicted malignant	4986	101
Predicted benign	14	23

Table A.8: Confusion matrix for WKNN Minkowski distance, Triangular kernel and SMOTE

	Observed malignant	**Observed benign**
Predicted malignant	4916	88
Predicted benign	86	36

Table A.9: Confusion matrix for WKNN Minkowski distance, Epanechinkov kernel and SMOTE

	Observed malignant	**Observed benign**
Predicted malignant	4916	88
Predicted benign	86	36

Table A.10: Confusion matrix for WKNN Minkowski distance and Triangular kernel

	Observed malignant	**Observed benign**
Predicted malignant	4986	101
Predicted benign	14	23

Table A.11: Confusion matrix for WKNN Minkowski distance, Triangular kernel and SMOTE

	Observed malignant	**Observed benign**
Predicted malignant	4986	101
Predicted benign	14	23

A.1.3 Accuracy table

Table A.12: Table showing the accuracy measures for WKNN models

	Euc-Epa S	**Euc-Epa**	**Euc-Tri S**	**Euc-Tri**
Accuracy	91.86	97.22	92.05	97.13
Sensitivity	93.12	99.38	93.36	99.28
Specificity	41.13	10.48	39.52	10.48
Precision	98.46	97.81	98.42	97.81
F1 Score	95.71	98.59	95.82	98.54
Cohen's Kappa	0.918	0.972	0.92	0.971
AUC	0.671	0.549	0.664	0.549

Table A.13: Table showing the accuracy measures for WKNN models

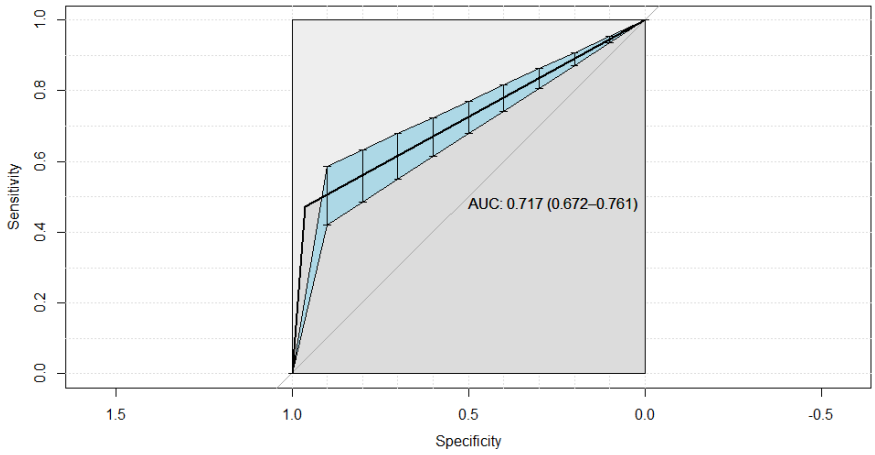
	Man-epa S	**Man-Epa**	**Man-Tri**	**Man-Tri S**
Accuracy	92.83	97.11	97.21	92.99
Sensitivity	94.22	99.28	99.28	94.38
Specificity	37.1	9.68	13.71	37.1
Precision	98.37	97.79	97.89	98.37
F1 Score	96.25	98.53	98.58	96.34
Cohen's Kappa	0.928	0.971	0.972	0.93
AUC	0.657	0.545	0.565	0.657

Table A.14: Table showing the accuracy measures for WKNN models

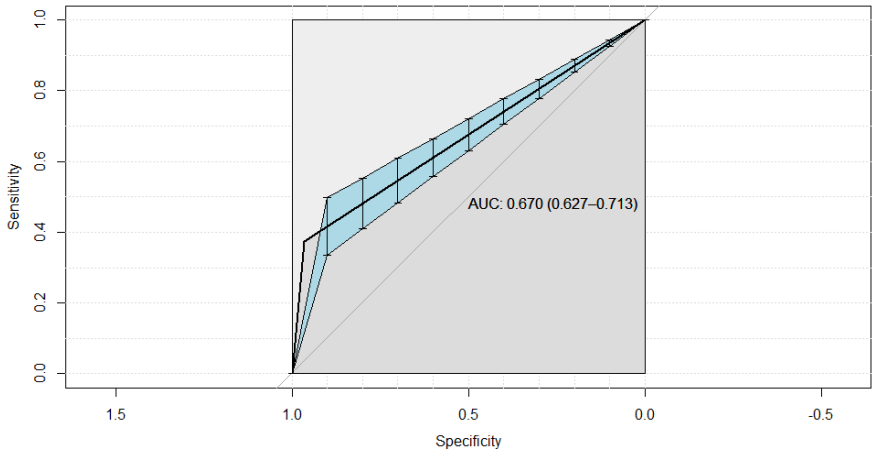
	Min-Epa	**Min-Epa S**	**Min-Tri**	**Min-Tri S**
Accuracy	97.77	96.68	97.75	96.64
Sensitivity	99.76	98.34	99.72	98.32
Specificity	17.74	29.84	18.54	29.03
Precision	98	98.26	98.01	98.24
F1 Score	98.87	98.3	98.86	98.28
Cohen's Kappa	0.98	0.966	0.978	0.966
AUC	0.588	0.641	0.591	0.637

A.2 Artifical Neural Network

A.2.1 ROC Curves

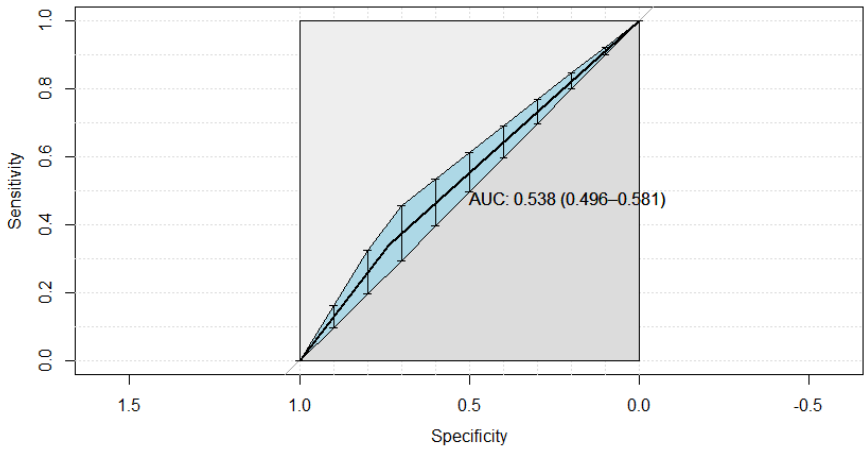


(a) ANN with Tanh activation function, SMOTE and threshold optimization's ROC curve

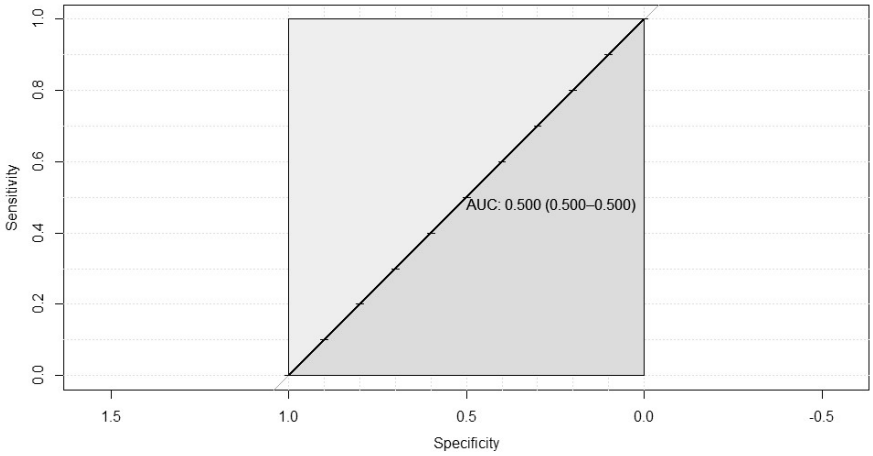


(b) ANN with ReLU activation function, no dropout layers, SMOTE and threshold optimization

Figure A.7: ANN with SMOTE and threshold optimization's ROC curve



(a) ANN with Tanh activation function and dropout layers' ROC curve



(b) ANN with ReLU activation function and dropout layers' ROC curve

Figure A.8: ANN with dropout layers' ROC curve

A.2.2 Confusion matrix

Table A.15: Confusion matrix for ANN with ReLU activation function with threshold. manipulation.

	Observed malignant	**Observed benign**
Predicted malignant	4866	77
Predicted benign	172	46

Table A.16: Confusion matrix for ANN with Tanh activation function, SMOTE and threshold optimization

	Observed malignant	**Observed benign**
Predicted malignant	4845	65
Predicted benign	193	58

Table A.17: Confusion matrix for ANN with Tanh activation function and dropout layers

	Observed malignant	**Observed benign**
Predicted malignant	3705	81
Predicted benign	1333	42

Table A.18: Confusion matrix for ANN with ReLU activation function and dropout layers

	Observed malignant	**Observed benign**
Predicted malignant	5038	123
Predicted benign	0	0

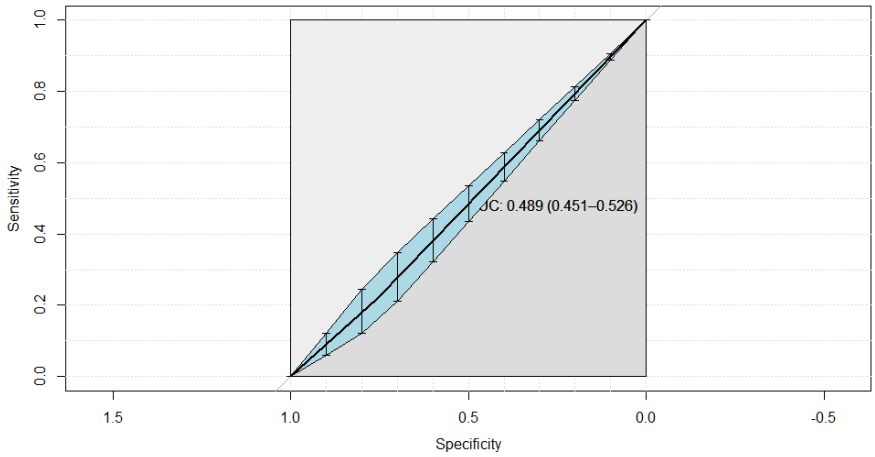
A.2.3 Accuracy table

Table A.19: Table showing the accuracy measures for ANN models

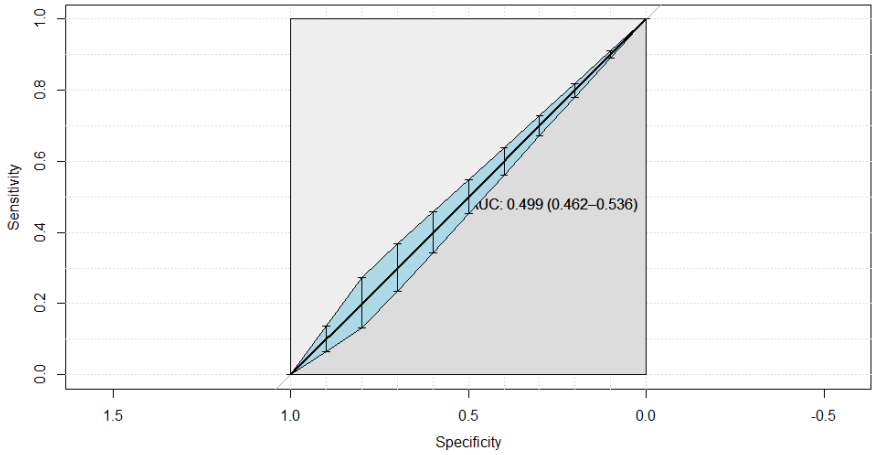
	Tanh SMOTE	**ReLU SMOTE**	**Tanh dropout**	**ReLU dropout**
Accuracy	95	95.17	72.6	97.62
Sensitivity	96.17	96.59	73.54	97.62
Specificity	47.15	37.4	34.14	0
Precision	98.68	98.44	97.86	97.62
F1 Score	97.41	97.51	83.98	97.62
Cohen's Kappa	0.95	0.951	0.724	0.97
AUC	0.717	0.67	0.538	0.5

A.3 Bayesian Neural Network

A.3.1 ROC Curves



(a) BNN with MCMC and ReLU activation function's ROC curve



(b) BNN with MCMC and Tanh activation function's ROC curve

Figure A.9: BNN with MCMC's ROC curve

A.3.2 Confusion matrix

Table A.20: Confusion matrix for BNN with ReLU activation function and MCMC

	Observed malignant	**Observed benign**
Predicted malignant	3817	96
Predicted benign	1221	27

Table A.21: Confusion matrix for BNN with Tanh activation function and MCMC

	Observed malignant	**Observed benign**
Predicted malignant	3963	97
Predicted benign	1075	26

A.3.3 Accuracy table

Table A.22: Table showing the accuracy measures for BNN models

	ReLU MCMC	**Tanh MCMC**
Accuracy	74.48	77.29
Sensitivity	75.76	78.66
Specificity	21.95	21.14
Precision	97.55	97.61
F1 Score	85.29	87.12
Cohen's Kappa	0.74	0.77
AUC	0.489	0.599

B R Code

The code is split into the following:

- Explortory data analysis
- Naive Bayes models
- KNN models
- WKNN models
- ANN models
- BNN models

B.1 Explortory data analysis

```
#JingYuan Gong
#August 2021
#Masters thesis coding

#-----------------------------------------------------------------------------------------------------------
#Work session and data importing
setwd("E:/University/Wits/Masters/Thesis/Coding/v17 Nb, KNN, WKNN + balance + thres/Nb, KNN, WKNN 1")
set.seed("1")
data<-read.csv("E:/University/Wits/Masters/Thesis/Coding/v5/Data 5 Reduced data.csv",sep=";")

#-----------------------------------------------------------------------------------------------------------
#Library used
#library(gmodels)
#library(MASS)
#library(crone)
library(caret)
library(class)
library(e1071)
library(ggplot2)
library(kknn)
library(pROC)
library(tidyverse)
library(unbalanced)

#-----------------------------------------------------------------------------------------------------------
#Basic EDA
names(data)[7] = "Tumour Classification"
names(data)[2] = "Age group"
names(data)[3] = "Ethinicity group"
names(data)[5] = "Year of diagnosis"
names(data)[6] = "Site of cancer"
names(data)[8] = "Type of cancer cell"
names(data)[14]="Number of positive nodes found"
names(data)[16]="Number of months survived"

str(data)
data$`Tumour Classification` <- factor(data$`Tumour Classification`)
str(data$`Tumour Classification`)
table(data$`Tumour Classification`)
data1<-data[,-c(1,10,11,12,13,15,16,17,18)]
str(data1)

#Ratio is 40:1 for M:B -> 0:1
#M -> 40000
#B ->  1000
#Very unbalanced data
```

```
#Graphing unbalanced data
ggplot(data, aes(x=factor(`Tumour Classification`)))+
geom_bar(fill="steelblue")+
theme_minimal()+
ggtitle("Tumour Balances")+
xlab("Tumour Classification")+
ylab("Count")

#checking all the types
str(data1)
data1$Sex<-as.integer(data1$Sex)-1
str(data1)

#-------------------------------------------------------------------------------------------------------
#Data partition
no=c(1:nrow(data1))
dat=cbind(no,data1)
split <- 0.75
Data_index<- createDataPartition(dat$no, p=split, list=FALSE)
data_train <- dat[Data_index,]
data_test <- dat[-Data_index,]
training<-data_train[,-1]
test<-data_test[,-1]

#-------------------------------------------------------------------------------------------------------
#Using Smote

smoted<-ubSMOTE(data_train[,-c(1,7)],factor(data_train$`Tumour Classification`),perc.over = 4000,k=5,perc.under = 100)
smoted_data<-cbind(smoted$X,smoted$Y)

names(smoted_data)[10] = "Tumour Classification"
table(smoted_data$`Tumour Classification`)
str(smoted_data)

ggplot(smoted_data, aes(x=factor(`Tumour Classification`)))+
geom_bar(fill="steelblue")+
theme_minimal()+
ggtitle("Tumour Balances")+
xlab("Tumour Classification")+
ylab("Count")

#-------------------------------------------------------------------------------------------------------
#EDA smote data
smoted_data_1 <- smoted_data
str(smoted_data_1)

#Graphing unbalanced data
ggplot(smoted_data_1, aes(x=factor(`Tumour Classification`)))+
geom_bar(fill="steelblue")+
theme_minimal()+
ggtitle("Tumour Balances")+
xlab("Tumour Classification")+
ylab("Count")
#Confusion matrix of the form of  0 = postive
#    |-----0-----|-----1------|
#  0 |TP (tb[1]) | FP (tb[3]) |
#  1 |FN (tb[2]) | TN (tb[4]) |

#-------------------------------------------------------------------------------------------------------
#Accuracy function
calculate_stats<- function(tb, model_name) {
Accuracy <- (tb[1] + tb[4])/(tb[1] + tb[2] + tb[3] + tb[4])
Sensitivity <- tb[1]/(tb[1] + tb[2])
Specificity <- tb[4]/(tb[4] + tb[3])
Precision <- tb[1]/(tb[1]+tb[3])
f1 <- (2*Precision * Sensitivity) / (Precision + Sensitivity)
NIR<-max(tb[1]+tb[2],tb[3]+tb[4])/(tb[1] + tb[2] + tb[3] + tb[4])
pe<- (tb[1] + tb[2])/(tb[1] + tb[2] + tb[3] + tb[4])*(tb[1] + tb[3])/(tb[1] + tb[2] + tb[3] + tb[4])*
(tb[4] + tb[2])/(tb[1] + tb[2] + tb[3] + tb[4])*(tb[4] + tb[3])/(tb[1] + tb[2] + tb[3] + tb[4])
cohen<-(Accuracy-pe)/(1-pe)


cat(model_name, ": \n")
cat("\tAccuracy = ", Accuracy*100, "%.")
cat("\n\tSensitivity = ", Sensitivity*100, "%.")
cat("\n\tSpecificity = ", Specificity*100, "%.")
cat("\n\tPrecision = ", Precision*100, "%.")
```

```
cat("\n\tF1 Score = ", f1*100, "%.")
cat("\n\tNIR Score = ", NIR*100, "%.")
cat("\n\tCohen's kappa = ", cohen, ".\n\n")
}
```

B.2 Naive Bayes models

```
#Naive Bayes
nb=naiveBayes(smoted_data_1$`Tumour Classification`~., data=smoted_data_1)
nb1=predict(nb,test[,-6],type="raw")

#-----------------------------------------------------------------------------------------------------
#NB threshold + predictions
pred_y<-cbind(nb1,nrow(nb1))
threshold_nb<-0.089
for (i in 1:nrow(nb1)) {
if(pred_y[i,1] > threshold_nb){
pred_y[i,3] = 0
} else if(pred_y[i,1] < threshold_nb){
pred_y[i,3] = 1
}
}
```

B.3 KNN models

```
kn5=class::knn(train=smoted_data_1[,-10],test=test[,-6],cl=smoted_data_1[,10],k=5,prob = TRUE)
pred_knn5<-attributes(kn5)$prob #for winning class
pred_knn5
```

B.4 WKNN models

```
Min_Tri_kknn <- kknn(`Tumour Classification`~., smoted_data_1, test, distance=0.1,kernel = "triangular")
Min_Tri_fit <- fitted(Min_Tri_kknn)

Min_Tri_pred_y<-cbind(Min_Tri_kknn$prob,nrow(Min_Tri_kknn$prob))

threshold_Min_Tri_pred_y<-0.47
for (i in 1:nrow(Min_Tri_kknn$prob)) {
if(Min_Tri_pred_y[i,1] > threshold_Min_Tri_pred_y){
Min_Tri_pred_y[i,3] = 0
} else if(Min_Tri_pred_y[i,1] < threshold_Min_Tri_pred_y){
Min_Tri_pred_y[i,3] = 1
}
}

Min_Tri_tb_nn_weighted <-table(Predicted=Min_Tri_pred_y[,3],Actual=test$`Tumour Classification`)
Min_Tri_tb_nn_weighted

calculate_stats(Min_Tri_tb_nn_weighted,test$`Tumour Classification`)
```

B.5 ANN models

```
#*****************************************************************************************************

#Tanh with dropout ANN

#*****************************************************************************************************
#-----------------------------------------------------------------------------------------------------
#Tanh + Drop
model_dropout_Tanh<-keras_model_sequential()
model_dropout_Tanh %>%
layer_dense(units = 50,activation = "Tanh",input_shape = c(9)) %>%
layer_dropout(0.2) %>%
layer_dense(units=40,activation = 'tanh')  %>%
```

```
layer_dropout(0.2) %>%
layer_dense(units=30,activation = 'tanh')  %>%
layer_dropout(0.1) %>%
layer_dense(units = 2,activation = "sigmoid")
summary(model_dropout_Tanh)

#-------------------------------------------------------------------------------------------------------------
#Tanh + Drop loss and optimizer
model_dropout_Tanh%>%
compile(loss="binary_crossentropy",
optimizer="adam",
metrics="accuracy")

#-------------------------------------------------------------------------------------------------------------
#Tanh + Drop fitting
history_dropout_Tanh<-model_dropout_Tanh %>%
fit(training,
trainlabels,
epochs = 100,
batch_size = 5,
validation_split=0.2,
class_weight=list("0"=1,"1"=40))
plot(history_dropout_Tanh)

#-------------------------------------------------------------------------------------------------------------
#Tanh + Drop accuracy and threshold optimization
model_dropout_Tanh %>%
evaluate(test,testlabels)

pred_dropout_Tanh<-model_dropout_Tanh %>%
predict(test)

pred_y_dropout_Tanh<-cbind(pred_dropout_Tanh,nrow(pred_dropout_Tanh))
threshold_Tanh_dropout<-0.0021
for (i in 1:nrow(pred_dropout_Tanh)) {
if(pred_y_dropout_Tanh[i,1] >threshold_Tanh_dropout){
pred_y_dropout_Tanh[i,3] = 0
} else if(pred_y_dropout_Tanh[i,1] < threshold_Tanh_dropout){
pred_y_dropout_Tanh[i,3] = 1
}
}

tb_nn_dropout_Tanh <-table(Predicted=pred_y_dropout_Tanh[,3],Actual=testtarget)
tb_nn_dropout_Tanh

calculate_stats(tb_nn_dropout_Tanh,testlabels)
```

B.6 BNN models

```
#BNN tanh with Var Inf
use_backend("tensorflow") #might need to run this line again
model_tanh<-keras_model_sequential()
model_tanh %>%
layer_dense(units = 50,activation = "tanh",input_shape = c(9))%>%
layer_dropout(0.2) %>%
layer_dense(units=40,activation = 'tanh') %>%
layer_dropout(0.2) %>%
layer_dense(units=30,activation = 'tanh') %>%
layer_dropout(0.1) %>%
layer_dense_variational(
units = 2,
activation = "sigmoid",
make_posterior_fn = posterior_mean_field,
make_prior_fn =prior_trainable,
kl_use_exact =  FALSE
)
summary(model_tanh)
#-------------------------------------------------------------------------------------------------------------
#BNN tanh with loss and optimizer
model_tanh%>%
compile(loss="binary_crossentropy",
optimizer="adam",
metrics="accuracy")

#-------------------------------------------------------------------------------------------------------------
```

```
#BNN tanh fit
history_tanh<-model_tanh %>%
fit(training,
trainlabels,
epochs = 100,
batch_size = 5,
validation_split=0.25,
class_weight=list("0"=1,"1"=40))
plot(history_tanh)

#-----------------------------------------------------------------------------------------------------------
#BNN tanh accuracy + threshold optimization
model_tanh %>%
evaluate(test,testlabels)

pred_tanh<-model_tanh %>%
predict(test)

summary(pred_tanh[,1])
sd(pred_tanh[,1])
#use this to find the top ten percent etc.

pred_y_tanh<-cbind(pred_tanh,nrow(pred_tanh))
Threshold_tanh<-0.00982
for (i in 1:nrow(pred_tanh)) {
if(pred_y_tanh[i,1] > Threshold_tanh){
pred_y_tanh[i,3] = 0
} else if(pred_y_tanh[i,1] < Threshold_tanh){
pred_y_tanh[i,3] = 1
}
}

tb_nn_tanh <-table(Predicted=pred_y_tanh[,3],Actual=testtarget)
tb_nn_tanh

calculate_stats(tb_nn_tanh,testlabels)
```

References

Abdar, Moloud, Maryam Samami, Sajjad Dehghani Mahmoodabad, Thang Doan, Bogdan Mazoure, Reza Hashemifesharaki, Li Liu, Abbas Khosravi, U Rajendra Acharya, Vladimir Makarenkov, et al. (2021), "Uncertainty quantification in skin cancer classification using three-way decision-based bayesian deep learning." *Computers in biology and medicine*, 135, 104418.

Ahmad, L Gh, AT Eshlaghy, A Poorebrahimi, M Ebrahimi, and AR Razavi (2013), "Using three machine learning techniques for predicting breast cancer recurrence." *J Health Med Inform*, 4, 3.

Al-Azri and H Mohammed (2016), "Delay in cancer diagnosis: causes and possible solutions." *Oman medical journal*, 31, 325.

Allaire, J (2012), "Rstudio: integrated development environment for r." *Boston, MA*, 770, 165–171.

Amrane, Meriem, Saliha Oukid, Ikram Gagaoua, and Tolga Ensari (2018), "Breast cancer classification using machine learning." In *2018 Electric Electronics, Computer Science, Biomedical Engineerings' Meeting (EBBT)*, 1–4, IEEE.

Andrieu, Christophe, Nando De Freitas, Arnaud Doucet, and Michael I Jordan (2003), "An introduction to mcmc for machine learning." *Machine learning*, 50, 5–43.

Berger, James O (2000), "Bayesian analysis: A look at today and thoughts of tomorrow." *Journal of the American Statistical Association*, 95, 1269–1276.

Bernbeck, B, D Wüller, G Janssen, R Wessalowski, U Göbel, and DT Schneider (2009), "Symptoms of childhood acute lymphoblastic leukemia: red flags to recognize leukemia in daily practice." *Klinische Pädiatrie*, 221, 369–373.

Brownlee, Jason (2019), "A gentle introduction to the rectified linear unit (relu)."

Burke, Harry B (1994), "Artificial neural networks for cancer research: outcome prediction." In *Seminars in Surgical Oncology*, volume 10, 73–79, Wiley Online Library.

Chakraborty, Sajib and Taibur Rahman (2012), "The difficulties in cancer treatment." *ecancermedicalscience*, 6.

Chivers, Corey and Maintainer Corey Chivers (2011), "Package 'mhadaptive'."

Cohen, Jacob (1960), "A coefficient of agreement for nominal scales." *Educational and psychological measurement*, 20, 37–46.

Cruz, Joseph A. and David S. Wishart (2006), "Applications of machine learning in cancer prediction and prognosis." *Cancer Informatics*, 2, 117693510600200030, URL https://doi.org/10.1177/117693510600200030.

De Amorim, Renato Cordeiro and Boris Mirkin (2012), "Minkowski metric, feature weighting and anomalous cluster initializing in k-means clustering." *Pattern Recognition*, 45, 1061–1075.

Devesa, Susan S, William J Blot, BJ Stone, Barry A Miller, Robert E Tarone, and Joseph F Fraumeni Jr (1995), "Recent cancer trends in the united states." *JNCI: Journal of the National Cancer Institute*, 87, 175–182.

Doppalapudi, Shreyesh, Robin G. Qiu, and Youakim Badr (2021), "Lung cancer survival period prediction and understanding: Deep learning approaches." *International Journal of Medical Informatics*, 148, 104371, URL https://www.sciencedirect.com/science/article/pii/S1386505620319079.

Dudani, Sahibsingh A (1976), "The distance-weighted k-nearest-neighbor rule." *IEEE Transactions on Systems, Man, and Cybernetics*, 325–327.

Emery, Jon D (2015), "The challenges of early diagnosis of cancer in general practice." *Medical Journal of Australia*, 203, 391–393.

Ferlay, J, M Ervik, F Lam, M Colombet, and L Mery (2020), "Global cancer observatory: Cancer today."

Fix, Evelyn (1985), *Discriminatory analysis: nonparametric discrimination, consistency properties*, volume 1. USAF school of Aviation Medicine.

Fritsch, Stefan, Frauke Guenther, and Maintainer Frauke Guenther (2019), "Package 'neuralnet'." *Training of Neural Networks. Recuperado de https://cran. r-project. org/web/packages/neuralnet/neuralnet. pdf*.

Galton, Francis (1907), "Vox populi."

Gohrani, Kunal (2019), "Different types of distance metrics used in machine learning." https://medium.com/@kunal_gohrani/different-types-of-distance-metrics-used-in-machine-learning-e9928c5e26c7.

Guo, Chuan, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger (2017), "On calibration of modern neural networks." In *International Conference on Machine Learning*, 1321–1330, PMLR.

Han, Jun and Claudio Moraga (1995), "The influence of the sigmoid function parameters on the speed of backpropagation learning." In *International workshop on artificial neural networks*, 195–201, Springer.

Jain, Anil K, Jianchang Mao, and K Moidin Mohiuddin (1996), "Artificial neural networks: A tutorial." *Computer*, 29, 31–44.

Jospin, Laurent Valentin, Wray Buntine, Farid Boussaid, Hamid Laga, and Mohammed Bennamoun (2020), "Hands-on bayesian neural networks–a tutorial for deep learning users." *arXiv preprint arXiv:2007.06823*.

Khan, Sarwar Shah, Qiong Ran, Muzammil Khan, and Mengmeng Zhang (2019), "Hyperspectral image classification using nearest regularized subspace with manhattan distance." *Journal of Applied Remote Sensing*, 14, 032604.

Kourou, Konstantina, Themis P. Exarchos, Konstantinos P. Exarchos, Michalis V. Karamouzis, and Dimitrios I. Fotiadis (2015), "Machine learning applications in cancer prognosis and prediction." *Computational and Structural Biotechnology Journal*, 13, 8–17, URL https://www.sciencedirect.com/science/article/pii/S2001037014000464.

Kourou, Konstantina, George Rigas, Costas Papaloukas, Michalis Mitsis, and Dimitrios I Fotiadis (2020), "Cancer classification from time series microarray data through regulatory dynamic bayesian networks." *Computers in Biology and Medicine*, 116, 103577.

Kuhn, Max, Jed Wing, Steve Weston, Andre Williams, Chris Keefer, Allan Engelhardt, Tony Cooper, Zachary Mayer, Brenton Kenkel, R Core Team, et al. (2020), "Package 'caret'." *The R Journal*, 223.

MacKay, David JC (1992), "A practical bayesian framework for backpropagation networks." *Neural computation*, 4, 448–472.

Mathcentre (2006), "Hyperbolic functions."

Meyer, David, Evgenia Dimitriadou, Kurt Hornik, Andreas Weingessel, Friedrich Leisch, Chih-Chung Chang, Chih-Chen Lin, and Maintainer David Meyer (2019), "Package 'e1071'." *The R Journal*.

Miller, Kimberly D, Miranda Fidler-Benaoudia, Theresa H Keegan, Heather S Hipp, Ahmedin Jemal, and Rebecca L Siegel (2020), "Cancer statistics for adolescents and young adults, 2020." *CA: A Cancer Journal for Clinicians*, 70, 443–459.

Mujtaba, Hussain (2020), "What is rectified linear unit (relu)? — introduction to relu activation function."

Nall, Rachel (2018), "How does a doctor diagnose cancer?"

NCI (2021), "National cancer institute."

Nixon, Jeremy, Michael W Dusenberry, Linchuan Zhang, Ghassen Jerfel, and Dustin Tran (2019), "Measuring calibration in deep learning." In *CVPR Workshops*, volume 2.

Omondiagbe, David A, Shanmugam Veeramani, and Amandeep S Sidhu (2019), "Machine learning classification techniques for breast cancer diagnosis." In *IOP Conference Series: Materials Science and Engineering*, volume 495, 012033, IOP Publishing.

Peterson, Leif E (2009), "K-nearest neighbor." *Scholarpedia*, 4, 1883.

Pietrangelo, Ann (2019), "Benign and malignant tumors: How do they differ?"

R Core Team (2013), *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, URL http://www.R-project.org/. ISBN 3-900051-07-0.

Rawla, Prashanth and Adam Barsouk (2019), "Epidemiology of gastric cancer: global trends, risk factors and prevention." *Przeglad gastroenterologiczny*, 14, 26.

Ripley, Brian, William Venables, and Maintainer Brian Ripley (2015), "Package 'class'." *The Comprehensive R Archive Network*.

Rish, Irina et al. (2001), "An empirical study of the naive bayes classifier." In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, 41–46.

Rumelhart, David E, Geoffrey E Hinton, and Ronald J Williams (1986), "Learning representations by back-propagating errors." *nature*, 323, 533–536.

Ryu, Sung Mo, Sun-Ho Lee, Eun-Sang Kim, and Whan Eoh (2019), "Predicting survival of patients with spinal ependymoma using machine learning algorithms with the seer database." *World neurosurgery*, 124, e331–e339.

Sahu, Bibhuprasad, Sachi Mohanty, and Saroj Rout (2019), "A hybrid approach for breast cancer classification and diagnosis." *EAI Endorsed Transactions on Scalable Information Systems*, 6.

Sairamya, NJ, L Susmitha, S Thomas George, and MSP Subathra (2019), "Hybrid approach for classification of electroencephalographic signals using time–frequency images with wavelets and texture features." In *Intelligent Data Analysis for Biomedical Applications*, 253–273, Elsevier.

Samworth, Richard J et al. (2012), "Optimal weighted nearest neighbour classifiers." *The Annals of Statistics*, 40, 2733–2763.

Schliep, Klaus, Klaus Hechenbichler, and Maintainer Klaus Schliep (2016), "Package 'kknn'."

Sharma, Sagar (2017), "Activation functions in neural networks."

Sharma, Shubham, Archit Aggarwal, and Tanupriya Choudhury (2018), "Breast cancer detection using machine learning algorithms." In *2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS)*, 114–118, IEEE.

Singh, Archana, Avantika Yadav, and Ajay Rana (2013), "K-means with three different distance metrics." *International Journal of Computer Applications*, 67.

Strichartz, Robert S (2000), *The way of analysis*. Jones & Bartlett Learning.

Taphoorn, Martin JB, Lily Claassens, Neil K Aaronson, Corneel Coens, Murielle Mauer, David Osoba, Roger Stupp, René O Mirimanoff, Martin J van den Bent, and Andrew Bottomley (2010), "An international validation study of the eortc brain cancer module (eortc qlq-bn20) for assessing health-related quality of life and symptoms in brain cancer patients." *European Journal of Cancer*, 46, 1033–1040.

Vadivel, AKMSSA, AK Majumdar, and Shamik Sural (2003), "Performance comparison of distance metrics in content-based image retrieval applications." In *International Conference on Information Technology (CIT), Bhubaneswar, India*, 159–164.

WHO (2021), "Detailed fact sheet of cancer statistic for 3 march 2021."

Yegnanarayana, Bayya (2009), *Artificial neural networks*. PHI Learning Pvt. Ltd.

Yu, Jiangsheng and Xue-Wen Chen (2005), "Bayesian neural network approaches to ovarian cancer identification from high-resolution mass spectrometry data." *Bioinformatics*, 21, i487–i494.

Zhang, Harry (2004), "The optimality of naive bayes." *AA*, 1, 3.