Manchester Metropolitan University

# A Comparative Analysis of Machine Learning Algorithms for Diabetic Data Classification

**BSc (Hons)Computer Science 2020**

By Jessica Singer
Supervised by Giovanni Masala

## Declaration

No part of this project has been submitted in support of an application for any other degree or qualification at this or any other institute of learning. Apart from those parts of the project containing citations to the work of others, this project is my own unaided work. This work has been carried out in accordance with the Manchester Metropolitan University research ethics procedures, and has received ethical approval number 13303.

Signed

$j.singer$

**Acknowledgements**
Firstly, id like to thank my supervisor Giovanni Masala for his help and guidance throughout the duration of this project, as well as my family for continuous support.

## Abstract

**The prevalence of diabetes is globally problematic with the number of cases rising, costing the health care industry substantial amounts of money. To reduce costs the health care industry is constantly developing new convenient ways to diagnose a patient with this disease.**

**This paper aims to identify the best suitable classification model for the prediction of diabetes, with the focus of the definite number of non-diabetic cases and the definite number of diabetic cases.**

**Principle Component Analysis (PCA) was used as a feature selection technique identifying seven features as important for the diagnoses of diabetes.**

**The classification models used within this research were K Nearest Neighbors (KNN), Naive Bayes and Multilayer Perceptron (MLP). The measures of accuracy, sensitivity, specificity and AUC score were used to evaluate and compare the performances of classifiers.**

**The evaluation of these measures concluded MLP to be the best suitable classifier obtaining the results, 81% accuracy, 61% sensitivity, 90% specificity and 86% AUC score. K-NN results were comparable with 79% accuracy, 53% sensitivity, 91% specificity and 83% AUC score. Lastly, Naive Bayes obtained the results, 78% accuracy, 55% sensitivity, 88% specificity and 82% AUC score.**

**A graphical user interface (GUI) was developed in the process to allow a user to engage in the findings of this paper with added comparative features of PCA and Linear discriminate Analysis (LDA).**

# Contents

## List of Figures

## List of Tables

# 1 Introduction

## 1.1 Background

Machine Learning (ML) is the study of a range of analytical models in the aim to learn and improve from data to make a decision. Classification is the task of categorizing data into classes using many ML algorithms to train data and identify attributes to predict an outcome [27], verifying what category they classify as. Classification models fall into two different learning types, called supervised and unsupervised. Supervised classification trains data already knowing the desired outcome to predict with labeled attributes. Whereas Unsupervised learning predicts by training data without knowing the desired outcome and no labeled attributes [23]. Various Industry's practice classification tasks to analyze large amounts of data frequently to improve efficiency, reduce costs and increase profit. Specifically, classification is used to improve health care through the recognition of patterns and the development of diagnostic tools in real-time.

## 1.2 The Problem

Due to the rise of diabetes constant scientific research is being applied to the diagnoses, in the aim to improve efficiency and cut costs within the health care industry. ML models are frequently used to predict the diagnoses of diabetes for greater awareness of why a diagnosis result is generated. Furthermore, the identification of valid attributes that contribute towards the majority of a prediction. Much previous research has studied the classification of diabetes in different ways using a range of classifiers [2]. Alghamdi et al used a combination of decision tree models including, Random Forest, Naïve Bayes Tree, and Logistic Model Tree. These models were trained on a cardio-respiratory fitness data set to predict the risk of a non-diabetic patient getting the disease in the future, using the AUC score to evaluate the performance of the classifiers. Alghamdi et al concluded Random Forest and Naive Bayes Tree performed greater than Logistic Model Tree with similar results achieving an AUC score of 0.916 and 0.917.

Many studies have predicted diabetes using a different type of data [22]. Pratt et al developed a Convolutional Neural Network (CNN) to verify features within an eye that classify a patient having diabetic retinopathy, caused by diabetes. Digital fundus images were used as input data to train the CNN. The study achieved successful results producing a sensitivity score of 95% and an accuracy score of 75% derived from 5,000 validation images.

In this paper the Pima Indian diabetes data set was used, taken from Kaggle. The data set was originally sourced from the National Institute of Diabetes and Digestive Kidney Diseases. The features within this data set include nine variables, in the form of Pregnancies, Glucose, Blood Pressure, Skin Thickness, BMI, Diabetes Pedigree Function, Age, and Outcome as the target variable. Extensive amounts of input variables compromise classification results as some features are redundant to the contribution of prediction. The target variable outcome is used to determine the prediction of diagnosis for diabetic, non-diabetic represented in binary data.

This paper studied three classification algorithms for the prediction of diabetes consisting of, K Nearest Neighbor (K-NN), Naïve Bayes and Multilayer Perceptron (MLP) with Principle Component Analysis (PCA) and Linear Discriminant Analysis (LDA) used as exploratory steps to try and improve classification performance. The three chosen classifiers were selected to compare and evaluate their performance, to conclude the best suitable classifier for the diabetes data set.

During the process of this study a Graphical User Interface (GUI) was developed to be used as a diagnostic prediction tool for diabetes, for better visual comparison and the evaluation of classifiers performance. The Pima Indians data set is commonly used for comparative studies using many classifications models making it challenging to produce unique results. This study researched a substantial amount of similar work to differentiate and overcome this problem to provide adequate results.

## 2 Literature Review

Much research has studied the use of classification algorithms to predict a disease. However much preparation is needed before the performance of a classifier can be evaluated. The following headings define the gathering of research taken in a sequence of steps. These steps were essential, to identify a distinctive approach for the comparative classification problem of diabetes prediction. Data preparation identifies several techniques used to format high-quality data, to be used as input for a classifier. Similar Work addresses all studies with a similar problem using different approaches and data sets, underlying the key concepts that should be addressed. Finally Feature Analysis presents possible new findings that could be explored further, in the aim to create an improved classifier.

### 2.1 Data Preparation

Data pre-processing is a preparation stage that is applied to data before training any type of Machine Learning model (ML). The main techniques used for data pre-processing include cleaning, normalization, and data scaling. The product of data pre-processing is the final training set by improving the original set of data [19].

Kotsiantis et al thoroughly investigated these data pre-processing techniques and many more including, discretization and replacing missing values for the reduction of continuous amounts of unnecessary data. This study presented all well know pre-processing algorithms and how they are capable of achieving better results for ML algorithms.

Other studies have specifically researched into two similar data pre-processing techniques to conclude the best suitable one for their data, using classification for comparison [4,31]. Anderson studied the difference between data scaling and normalization to identify their impacts on heart disease data using K-NN for estimation.

Zehra et al research used the Pima Indians data set to compare pre-processed to non-pre-processed data using a range of classification algorithms. The data set where pre-processing was applied used discretization and the replacement of null values. The five following classifiers, Naive Bayes, MLP, Decision Tree, J.48, and Simple Cart were used to compare their results with the measure of accuracy used for evaluation. Their results for each type of input data used for classification can be seen in Table 1, along with a comparison of classifiers and their measure of accuracy. It can be seen that pre-processed data predicted higher accuracy results in contrast to non-pre-processed data.

Table 1: The study of Zehra et al, comparing the effects of pre-processed data to non-pre-processed for classification, applied on the Pima Indians data set.

| Classifier | Pre-processed | non-pre-processed |
|---|---:|---:|
| Naive Bayes | 80.3% | 76.3% |
| MLP | 81% | 75.39% |
| Simple Cart | 79.6% | 75.13% |
| J.48 | 80% | 73.82% |
| Decision Tree | 85.2% | 71.22% |

## 2.2 Similar Work

Diabetes diagnosis is frequently used for classification comparing numerous models to identify a patient, being diabetic or non-diabetic. Much work has used the common Pima Indians data set to identify an outcome comparing several classifiers. Table 2 shows the results of all authors who based their evaluation just on the estimation of accuracy, presenting a list of classifiers they used to compare their results [12,27,24].

### 2.2.1 Diabetes Classification

Table 2: Comparison of all previous work for classification evaluation just based on accuracy, using the Pima Indians data set.

| Author | Classifier | Accuracy |
|---|---|---|
| Hina et al | MLP | 81.8% |
| | Random Forest | 79.2% |
| | Logistic Regression | 79.2% |
| | Regression | 76.8% |
| | Naive Bayes | 76.3% |
| | J.48 | 75.3% |
| | ZeroR | 67.5% |
| Sisodia et al | Naive Bayes | 76.3% |
| | Decision Tree | 73% |
| | Support Vector Machine | 42% |
| Sa'di et al | Naive Bayes | 76.9% |
| | J.48 | 76.52% |
| | Radial Basis | 74.38% |

Hina et al applied seven different classifiers discovering MLP to be the best suitable classifier, obtaining the highest accuracy. However MLP proved to have longer training time compared to the other classifiers, which they believed to be dissatisfactory. As a result, they reviewed their decision to conclude K-NN as the best suitable classifier for the prediction of diabetes. Sisodia et al compared three classification models discovering Naive Bayes to predict the highest accuracy obtaining the same result as Hina et al. Their other accuracy results from other classifiers were significantly low in comparison to other models. Finally, Sa'di et al also studied the comparison of three classifiers concluding the same decision as Sisodia et al that Naive Bayes proved to be the best suitable classifier, by producing the highest accuracy of 76.9% out of all three of the studies using this classifier.

Separate studies using the accuracy measure have also found two classifiers to be just as suitable for this type of prediction [21]. Nurhayati et al compared only two classifiers including, Naive Bayes and K-NN. Their results produced roughly the same obtaining a rounded number of 91% accuracy, predicting a low number of incorrect cases.

Previous work has used more of an extensive amount of measures for better evaluation of classifiers [16]. Table 3 shows the results of Kaur et al, comparing five different classifiers, two of which were Support Vector Machines (SVM).

Table 3: Research of Kaur et al, comparing five different classifiers based on five measures.

| Classifier | Accuracy | Recall | Precision | F1 | AUC |
|---|---|---|---|---|---|
| Linear-Kernel-SVM | 89% | 87% | 88% | 87% | 90% |
| Radial-Basis-SVM | 84% | 83% | 85% | 83% | 85% |
| K-NN | 88% | 90% | 87% | 88% | 92% |
| Artificial Neural Network | 86% | 88% | 85% | 86% | 88% |
| Multifactor Dimensionality Reduction | 83% | 89% | 82% | 84% | 89% |

Kaur et al discovered four attributes were important within the Pima Indians data set, though they were unidentified within the study. The results of this study proved SVM-linear-Kernel and K-NN were the two best suitable classifiers for the Pima Indians data set, evaluated on all performance metrics obtaining the highest predictions, with the use of cross-validation.

All the prior studies mentioned compare completely different classifiers, although some research into diabetes classification focuses on a certain classifier but compare different variances of this one classification algorithm [14]. Jeatrakul & Wong compared the performance of different Neural Networks (NN) for the binary classification problem. This study didn't just compare one classification problem but three one of which was diabetes. They used five neural networks consisting of Back Propagation (BPNN), Radial Basis Function (RBFNN), General Regression (GRNN), Probabilistic (PNN), and lastly Complementary (CMTNN). Cross-validation was used to produce an average of the classification accuracy using this to compare the overall results. The results showed how accurate the predictions were for each Neural Network, deriving similar results. RBFNN obtained the highest accuracy of 76.76% with CMTNN 76.49%, BPNN 76.16%, followed by the lowest GRNN and PNN with accuracy results of 75.26%

### 2.2.2 Classification of Other Diseases

Other approaches, researched classification models on a range of diseases not just diabetes [3,7]. Thalassemia is also commonly researched using many classification models to predict this disease.

Amendolia et al studied the three classifiers of, K-NN, SVM and MLP proposing a two-layer-classifier system based on SVM. Their approach trained the first layer to distinguish between pathological and non-pathological cases, then the second to discriminate between two different Thalassemia carriers. They compared the classifiers based on the results sensitivity and specificity. Their findings showed MLP performed better to K-NN and SVM with MLP obtaining results of 96% specificity and 92% sensitivity. Besides Thalassemia, heart disease is another common practice for classification.

Chaitrali & Sulabha developed on a previous study by discovering the addition of two more input attributes, to the current data of 13 attributes improved the accuracy of predicting heart disease classification. The three following classifiers were used for this study, Decision Tree, Naïve Bayes and a Neural Network. Their results were compared based on accuracy, finding the Neural Network to obtain better results producing an accuracy of 100%, followed by Decision Tree with 99.62% and Naive Bayes with 90.74%.

All similar studies compared different amounts of classifiers evaluating their performance using a range of result metrics. Although much work fixated on one metric accuracy to evaluate the performance of classifiers, whereas limited resources used a comparison of a classification report to conclude their results. A consideration of many performance metrics should be explored more to generate reliable results. Two previous studies proved feature selection to be beneficial for the improvement of a classifier performance [16,7]. Experimenting with feature selection as well as reduction can improve a classification model, hence further research into feature analysis is considered.

## 2.3 Feature Analysis Studies

Many ML tasks consider dimension reduction algorithms to improve the performance of their model. Feature reduction and feature extraction is commonly used as a preliminary phase before training a classifier, as large amounts of non-beneficial data can decrease classification results. The two types differentiate from one another by the type of selective data used to train a modal. Feature extraction uses feature reduction to evaluate variables that contribute the most and are more valuable to the data, whereas feature reduction uses a smaller number of variables by reducing the number of input variables within each feature [25]. The use of feature analysis for an ML model can produce distinctive results compared to an incompetent one, for that reason many studies explore the use of this analysis.

Previous studies have considered feature reduction for visualization, to present data with a cleaner perspective [29]. Vanderplas mentions specifically Principle Component Analysis (PCA) in detail, summarising the benefits of this analysis especially for visualization to identify key features. Vanderplas presented how to estimate the number of components that are needed to describe the data used, which could then be used for classification.

Other approaches examined different feature reduction techniques to be used for classification purposes [18]. Kim et al compared two different analyses Linear Discriminant Analysis, Quadratic Discriminant Analysis, as well as KNN for wrist motion detection. They concluded that KNN was the proposed method producing the highest recognition rates.

PCA is also commonly studied for feature selection purposes to improve a classifier [32]. Zhang et al used feature selection for multi-labeled Naïve Bayes classification. They applied PCA to remove irrelevant and redundant features on several synthetic multi-labeled data sets, in the aim to improve classification performance.

The previous studies involving feature analysis exhibits many ways feature reduction can be applied either purely for visualization, feature selection or for improving a classifiers performance, with dimensionality reduced data. However experimenting with different feature reduction algorithms with the comparison of a range of classifiers is hardly investigated.

# 3 Algorithms

## 3.1 k Nearest Neighbor

K-NN is one of the most common machine learning algorithms used for supervised classification problems [11]. It is a non-parametric lazy learning algorithm with a minimal and fast training phase. Despite this, nearly almost all training data is needed to compare against every training example, making the classifiers testing run time slower, depending on how large the training set is.

The basis of this algorithm is to generate a decision for a data point to determine its class, controlled by its closest associated label within the feature space. This is referred to as majority voting, where a majority vote will be taken on the data points nearest neighbors which are decided by the hyper-parameter of K. The hyper-parameter is an integer number that is used to select the number of nearest neighbors. In order to determine the similarity of the closet associated neighbors the mathematical measure of Euclidean distance is used to find the distance of two points in a plane [13]. The Euclidean is calculated as:

$$d = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + .. + (p_n + q_n)^2} \tag{1}$$

### 3.1.1 Methodology

The value of K is the only input parameter needed for the K-NN algorithm. The amount depends on the size of the data to be trained on the classifier. To identify the suitable value of K for the Pima Indians data, two preliminary steps were taken. First, the square root of testing data items were taken defined as: $k = \sqrt{testdata}$
Secondly a somewhat middle value of K was chosen based on the max value from are testing items derived from the equation above. This middle value was chosen to avoid under-fitting the data. Concluding that K=11 was the suitable value for our data set.

## 3.2 Naïve Bayes

Naive Bayes classifier is a probabilistic algorithm by estimating the likelihood of an outcome, given their corresponding features. It assigns a new data point to an outcome based on its previous attributes by then selecting the highest calculated probability for the outcome belonging to that data point [8]. The classifier is formed using an assortment of probabilistic algorithms, to form this one model. The key algorithm which this classifier is based on is Bayes Theorem.

This algorithm makes a "Naive" assumption using the following two factors. The contribution of independent features, and the consideration of all attributes being equally as valuable as one another. Bayes Theorem is a mathematical calculation that depends on the evidence already provided given in the formula:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{2}$$

Each component for Formula 2 can be interpreted as:

$$P(B|A) = Likelihood \tag{3}$$

$$P(A) = PriorProbability \tag{4}$$

$$P(B) = NormalizingConstant \tag{5}$$

$$P(A|B) = PosteriorProbability \tag{6}$$

Bayes Theorem equation can be expressed as the following. Firstly, likelihood (formula 3), which is the prior probability of hypothesis B given data, is multiplied by the prior probability. Secondly, prior probability which is the hypothesis of A (formula 4) being true given no other data, is divided by the normalizing constant (formula 5). Finally, the normalizing constant described as the probability of the data given with no hypothesis forms the output of posterior probability (formula 6). The prior probability is the assigned outcome given the prior information within the data [5].

There are many variances of Naive Bayes classifiers depending on the assumptions they make referring to the type of distribution. Gaussian Naive Bayes (GNB) was decided as one of the chosen classifiers for this study. GNB is best used for continuous amounts of data as it is minimal compared to other distributions used on training data. For the reason that, only the mean and standard deviation is needed to result in the normal distribution of variables for each class.

## 3.3 Multilayer Perceptron

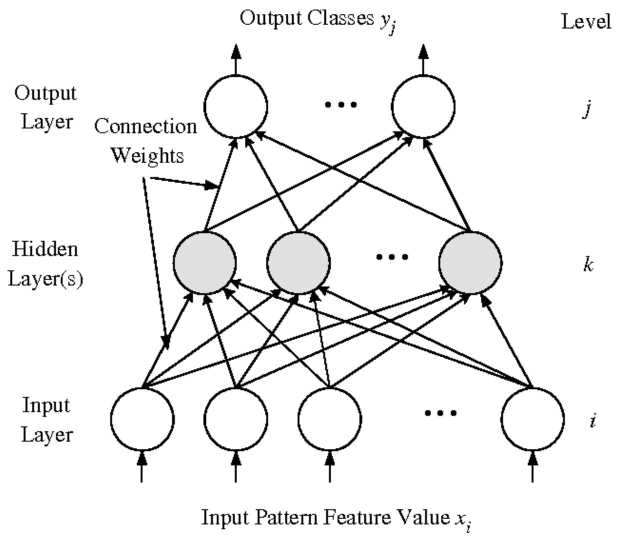


Figure 1: Yan et al 2006, Architecture of a multilayer perceptron network.

A Multilayer Perceptron (MLP) is an Artificial Neural Network (ANN) used for classification tasks by mapping input data to output a suitable class. MLP is a feed-forward ANN that transfers from the input layer to the output layer using connected multiple layers and nodes. The MLP has three components that are responsible for the operation of this ANN. These components consist of an input layer, hidden layers and an output layer demonstrated in Figure 1. The mechanism of an MLP is to start from the input layer where the input is the feature value shown as $(x1, x2, x3..)$. This feature is then multiplied by a weight in the first layer in the form of $W^1 : w^1, x1, w^1, x2, w^1, x3....$ As a result a weighted product is generated, which is added to a bias calculated as [17] :$z = \sum_i^m = 1 w_i x_i + bias$

The sum of the weighted input and bias (Z) is then passed to a non-linear activation function taken place within the perceptrons. The activation function is responsible for associating input Z with a feature. Subsequently the output of the function is passed as the input into the next layer and nodes and repeating the process through each hidden layer, until the output node is reached. This process is known as forward propagation.

Backpropagation is used within an MLP to estimate how well the model performed using the predicted output against the expected output to calculate a loss. This process is sent backward through the hidden layers in order to obtain the contribution towards the loss for each node. Furthermore, the weights are updated using gradient so that it will have a minimal impact on the loss, for the next classification task [14].

#### 3.3.1 Methodology

The choice of the two hyper-parameters, hidden layers and nodes were chosen based on tuning parameters. It was found that three hidden layers with the stack of ten nodes built the optimal MLP for the Pima Indians data. The default activation function Rectified Linear Unit (ReLU) is used for the decision of the output in each node after the calculation of the weighted input and bias, providing a non-linear approach.

### 3.4 Dimensionality Reduction Algorithms

#### 3.4.1 Principle Component Analysis

Principle Component Analysis (PCA) is commonly used within Machine Learning (ML), due to large data can increase the risk of over-fitting during the training of a model. PCA is a multi-functional feature reduction technique as it can be used for several purposes, including visualization, improve ML algorithm performance and feature selection.

#### 3.4.2 Principle Component Analysis Explained

PCA extracts data from a high dimensional space by projecting onto a lower-dimensional subspace [26]. The reduction of feature space is caused by rearranging the original data points using few orthogonal variables [9] so they sit along an axis representing the maximum variance.

The primary mathematical measures to calculate PCA are:

- Covariance Matrix, estimates the relationship between variables within the data. To determine how each variable corresponds to another variable.
- Eigenvector, determines the direction of the data points along the new coordinate axis.
- Eigenvalue, is essentially the value of importance of each Eigenvector plotting the highest eigenvalue first along the axis.

Covariance Matrix is the first step to be calculated for PCA. The Covariance matrix is calculated following the two-core measures:

- Covariance is the measure of the area between two elements to another moving in the same direction.
- Variance is the measure of the variability of the distribution of data [20].

Covariance can be calculated using the equation:

$$Covariance = \frac{\sum(x_i - \vec{x})(y_i - \vec{y})}{n-1} \tag{7}$$

Variance can be calculated using the equation:

$$Variance = \frac{\sum(x_i - \vec{x})^2}{n} \tag{8}$$

Equations 7-8 form the Covariance Matrix(CM) calculated as:

$$CM(x) = \frac{1}{n-1}(x - \vec{x})(x - \vec{x})^T \tag{9}$$

Equation 9 seen above multiples the matrix of data points by its transpose, in order to rearrange the initial data points so their Covariance is presented in a diagonal matrix, including their variances.

The eigenvector and eigenvalue essentially re-plot the newly formatted data from the covariance matrix onto a new axis by multiplying the eigenvectors by the original data. Eigenvectors know as the principal components, specify the directions of the new axis depending on the eigenvalue of each eigenvector. The eigenvectors with the highest eigenvalue is plotted first then descending until all eigenvectors have been plotted.

### 3.4.3 Principle Component Analysis In Practice

PCA was firstly carried out on the diabetes data-set during the pre-processing stage for visualization to explore the features in-depth. Formerly, the explained variance ratio (EVR) was calculated on the data-set to estimate the contribution of each component within the data presented in Table 4. The table shows the first two components constitute approximately 48%, almost half of the overall contribution. As a result, visualization was performed using these two components shown in Figure 2. The graph shows there is only slight linear separability between the two classes of diabetic (1) and non-diabetic (0). Each class differs from how their points are represented on the graph. The points of Class 1 are spread over a greater range, whereas Class 0 is somewhat clustered together.

Table 4: Explained Variance Ratio(EVR)

| Component | EVR |
|---|---|
| Component 1 | 0.26 |
| Component 2 | 0.22 |
| Component 3 | 0.13 |
| Component 4 | 0.11 |
| Component 5 | 0.09 |
| Component 6 | 0.08 |
| Component 7 | 0.05 |
| Component 8 | 0.05 |

PCA was also used as a feature reduction technique on the diabetes data-set to try to improve a classifier's performance, however proved to be unsuccessful. K-NN was used to

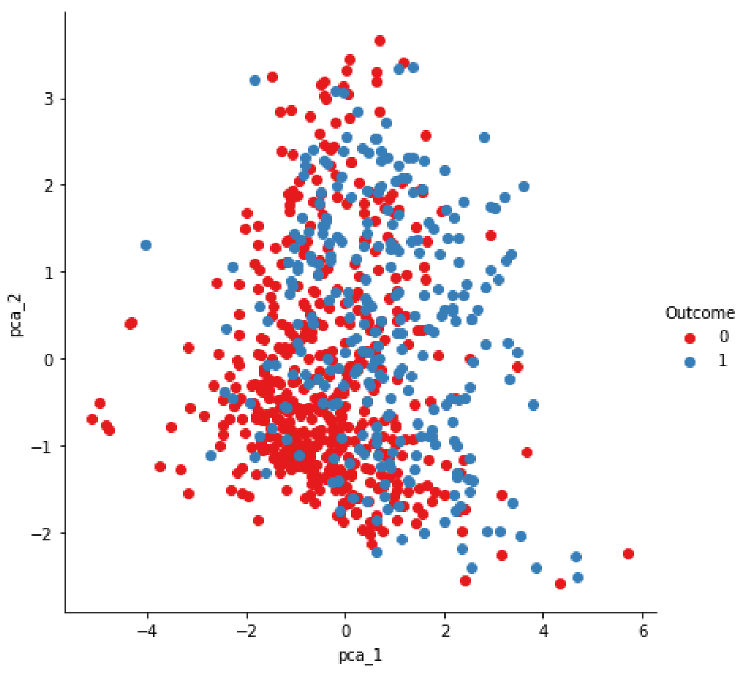


Figure 2: Scatter Graph of Components 1 and 2

estimate these findings using accuracy to evaluate the performance. Table 5 shows the comparison between a K-NN with PCA applied and a K-NN without. PCA fails to improve classification only obtaining 76% accuracy, whereas K-NN alone obtained 77% accuracy.

Table 5: Performance of K-NN

| Classifier | Accuracy |
|---|---|
| K-NN with PCA | 76% |
| K-NN | 77% |

PCA was also explored for feature selection by testing the number of components shown in Table 4 and how they affected the performance of a classifier. The original data set containing all features and all data entries and the classified value they represent, can be described in Table 6 [16].

Once again K-NN was used for estimation and accuracy to evaluate these results. Table 7 shows the results of three different quantities of components tested for classification. The original data with all components tested predicted 76% accuracy while seven components predicted 77% accuracy leaving 6 components predicting only 74% accuracy, thus no further testing for the remaining number of components was needed.

Table 6: Original Pima Indians Data set showing all features and described values

| Feature | Classified Value | Range |
|---|---|---|
| Pregnancies | Integer | [0,17] |
| Glucose | Real | [0,199] |
| Blood Pressure | Real | [0,122] |
| Skin Thickness | Real | [0,99] |
| Insulin | Real | [0,846] |
| BMI | Real | [0,67.2] |
| Diabetes Pedigree Function | Real | [0.0780,2.42] |
| Age | Integer | [21,81] |
| Outcome | Binary | 0 = non-diabetic/ 1 = diabetic |

Table 7: Performance of K-NN

| Number of Components | Accuracy |
|---|---|
| Original data | 76% |
| 7 Components | 77% |
| 6 Components | 74% |

### 3.4.4 Conclusion

PCA proved to be ineffective as a feature reduction technique to improve classification. Although feature selection proved to be effective, identifying the most vital features. The findings from feature extraction improved classification performance using seven features rather than eight. The seven features consist of Pregnancies, Glucose, Blood Pressure, Skin Thickness, Insulin, BMI and Diabetes Pedigree Function. The eighth feature Age was dropped from the data-set as it proved to be inadequate to the diabetes data. Other studies have used different feature selection techniques to identify the most vital features within this data set. Their findings were different with only four features being used for classification, though they were unidentified in the study [16].

Linear Discriminant Analysis (LDA) was another feature reduction technique used to try and improve a classifier performance. However, the findings of PCA produced a different value of input for the diabetes data set, due to improving the performance of the KNN algorithm. As a result of the new input data LDA had no beneficial effect on the performance of a classifier, making this analysis exempt from further evaluation with the chosen classifiers.

Both analyses are feature reduction techniques that reduce the features within the data using a linear approach, but in two different ways. PCA projects a new arrangement of data points to sit along the axis of maximum variance, and is done so linearly. LDA finds the axes that projects the new reduced data so there is maximum linear separability between classes. As both analyses failed to enhance the performance of the classifiers and from the observation of the two-components in Figure 2 we can see the classification problem we are addressing is non-linear.

# 4 Data Preparation

Before the chosen classifiers are trained on the diabetes data set preparation is applied, to format good quality of data to improve the ability of classification. A smaller format of data is used as input for the classifiers due to principal component analysis, for the reason specified in section 3.4.4. The Pima Indians data set now consists of seven features and one target outcome compared to the original data set consisting of eight features and one target outcome. These features in detail represent [31]:

1. Number of times pregnant.
2. 2-hour OGTT plasma glucose
3. Diastolic blood pressure
4. Triceps skin fold thickness
5. 2-hour serum insulin
6. Diabetes pedigree function (DPF)

The first five patient records of the Pima Indians data set used for this study can be shown in the table below. The data contains 768 patient records, categorizing into two classes of diabetic and non-diabetic. The diagnosis is presented as the target outcome in column eight of the table below.

Each class is indicated as a binary number showing all non-diabetic patients records with an outcome of zero, hence all diabetic patients records with an outcome of one. The generality of patients diagnosed diabetic totaled to 268, therefore the remaining 500 were diagnosed as non-diabetic. Before training the chosen classifiers for this study the data set was split into 75% training set and 25% testing set, forming the values of 576 patient records for training and 192 patient records for testing.

| Pregnancies | Glucose | Blood Pressure | Skin Thickness | Insulin | BMI | DPF | Outcome |
|---|---|---|---|---|---|---|---|
| 6 | 148 | 72 | 35 | 0 | 33.6 | 0.627 | 1 |
| 1 | 85 | 66 | 29 | 0 | 26.6 | 0.351 | 0 |
| 8 | 183 | 64 | 0 | 0 | 23.3 | 0.672 | 1 |
| 1 | 89 | 66 | 23 | 94 | 28.1 | 0.167 | 0 |
| 0 | 137 | 40 | 35 | 168 | 43.1 | 2.288 | 1 |

## 4.1 Data Distribution

Before any classification or feature analysis task can be done standardization and normalization must be applied. The table above shows all data entries and how they present a wide scale of values. In preparation for good quality data all data values must be within a similar range to avoid complications.

Sklearn is a machine learning library in Python programming language. The library offers various properties for classification including Min-Max Scaler and Standard Scaler. Both properties are used for data pre-processing to achieve evenly distributed data but

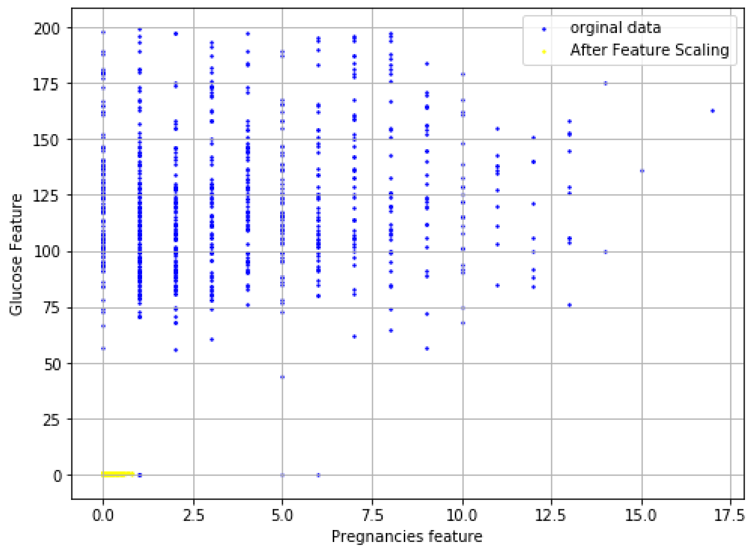


Figure 3: The distribution of Feature values, Pregnancies and Glucose

differ in the way they achieve these results. Min-Max Scaler transforms all values within the training set so they scale between 0-1 using the minimum and maximum values to determine a normalized number. Standard Scaler is more complex as the range from 0-1 is scaled to a unit variance, using the measure of Gaussian distribution [6].

Both properties were applied to the diabetes data set as provision for well-scaled data and to identify the correct method for this study. Min-Max scaler was discovered to achieve better results for scaling the diabetes data set and is used before feature analysis and classification. Figure 3 shows the effect of Min-Max scaler on the feature values of pregnancies and glucose, as well as the variety of distributions of non-normalized data and normalized data.

## 5 Comparative Results and Evaluation

The performance of the chosen classifiers for this study was assessed with the four metrics, accuracy, sensitivity, specificity and AUC score. All the metric results are determined by how many true or false predictions they estimate for diabetic or non-diabetic specified as:

- True Positive (TP)
- False Positive (FP)
- True Negative (TN)
- False Negative (FN)

For this study the above predictions signify the diagnoses of diabetic or non-diabetic, being either true or false tested by the actual outcome. Accuracy is the result of all correct predictions calculated with the addition of true positive cases and true negative cases. Sensitivity is the definite number of positive cases decided by the those predicted true positive, as well as those predicted false negative. Specificity is the inverse in which the definite number of negative cases are decided by those predicted true negative, as well as those predicted false positive. These metrics are calculated with the equations:

$$Sensitivity = (TP)/(TP + FN) \tag{10}$$

$$Specificity = (TN)/(TN + FP) \tag{11}$$

The AUC score is the measure of the area under the receiver operating characteristic (ROC) curve, signifying how well a binary classifier distinguishes between positive or negative, based on different thresholds. The ROC curve is a plot of true positive rate (TRP) on the x-axis, also recognized as sensitivity, along with a false positive rate (FPR) on the y-axis [1]. FPR is a variance of specificity in the form of: $FPR = 1 - Specificity$

Table 8: Comparative Performance of Classifiers

| Algorithm | Accuracy | Sensitivity | Specificity | AUC score |
|---|---|---|---|---|
| K-NN | 79% | 53% | 91% | 83% |
| Naive Bayes | 78% | 55% | 88% | 82% |
| MLP | 81% | 61% | 90% | 86% |

Table 8 which can be seen above presents the performance metrics of the classifiers, K-NN, Naive Bayes and MLP, trained on the Pima Indians data set. In terms of comparability all classifier models produce a similar pattern of results. For all classifiers sensitivity obtained the lowest results, predicting a low definite percentage of diabetic cases, while specificity, conversely obtained the highest results by predicting a high definite percentage of non-diabetic cases. The remaining metrics being accuracy and AUC score fall between the results of sensitivity and specificity.

The accuracy metric shown in column one tells us how accurate these classifiers were at making predictions on the diabetes data set, concluding MLP to produce the most accurate results with an estimation of 81%, leaving K-NN and Naive Bayes to shortfall with the accuracy results of 79% and 78%, respectively. Besides accuracy it is crucial to evaluate the other performance metrics of these classifiers as they determine their overall capability to predict diabetes.

|  |  | Predicted |  |  |
|---|---|---|---|---|
|  |  | Negative(0) | Positive(1) | Total |
| Actual | Negative(0) | 118 | 12 | $118 + 12$ |
|  | Positive(1) | 29 | 33 | $29 + 33$ |
|  | Total | $118 + 29$ | $12 + 33$ |  |

Table 9: K-NN Confusion Matrix

The classifiers capability to predict each outcome of a patient can be determined with their corresponding confusion matrix. Table 9 presents a confusion matrix for K-NN, where the values of individual predictions are shown within each box. The addition of the top boxes are the sum of all predictions made for non-diabetic cases with the contrasting boxes below accounting for the predictions of diabetic cases. The K-NN capability to correctly identify non-diabetic cases is greater than any other prediction with a value of 118. As a result, a minimal number of incorrect predictions were made for this type of diagnosis. Besides the prediction of true non-diabetic the number of incorrect predictions for diabetic cases was relatively high, considering there were a low number of cases for this diagnosis. Overall it can be said that K-NN capability to predict the definite number of non-diabetic cases is better than any other prediction. The specificity for K-NN can validate these findings with a result of 91%.

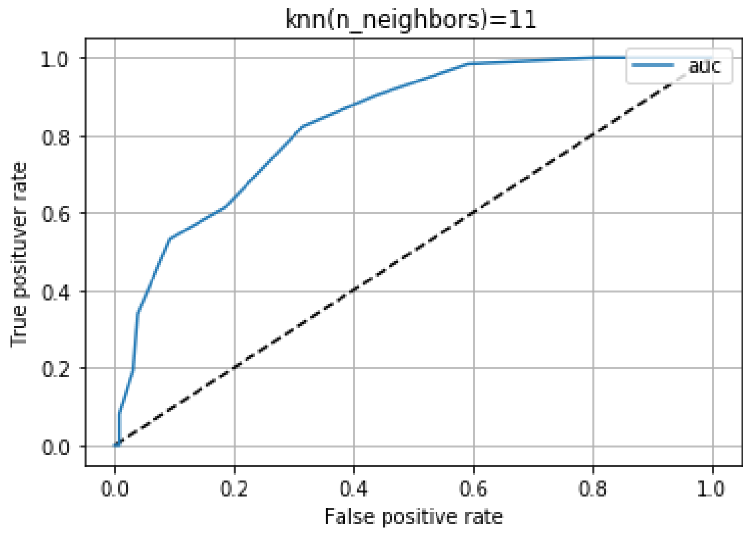


Figure 4: K-NN: AUC-ROC curve

A classifiers overall performance can be visually represented within an AUC-ROC graph shown in Figure 4. AUC-ROC curve correlates with the AUC score, both demonstrated on the previous page. As previously stated the AUC score is the computation of the area under the ROC curve which can be determined in the classifiers AUC-ROC graphs above. K-NN produced a fairly high AUC-score of 83%, as a result a tall ROC curve was generated verifying how many predictions were correctly identified.

| | | Predicted | | |
|---|---|---|---|---|
| | | Negative(0) | Positive(1) | Total |
| Actual | Negative(0) | 115 | 15 | $115 + 15$ |
| | Positive(1) | 28 | 34 | $28 + 34$ |
| | Total | $115 + 28$ | $15 + 34$ | |

Table 10: Naive Bayes Confusion Matrix

Table 10 shows the confusion matrix for Naive Bayes and the values for each type of prediction made by this classifier. Naive Bayes has a lesser capability to correctly predict a true case in comparison to K-NN. However this classifier produced a slightly higher value for true diabetic case with a value of 34, while K-NN obtained the value of 33 for this prediction, seen in Table 9. Despite Naive Bayes generating a lower specificity of 88% this classifiers capability to predict the definite number of diabetic cases is slightly greater then K-NN with 55% sensitivity, while K-NN only obtained 54% sensitivity.

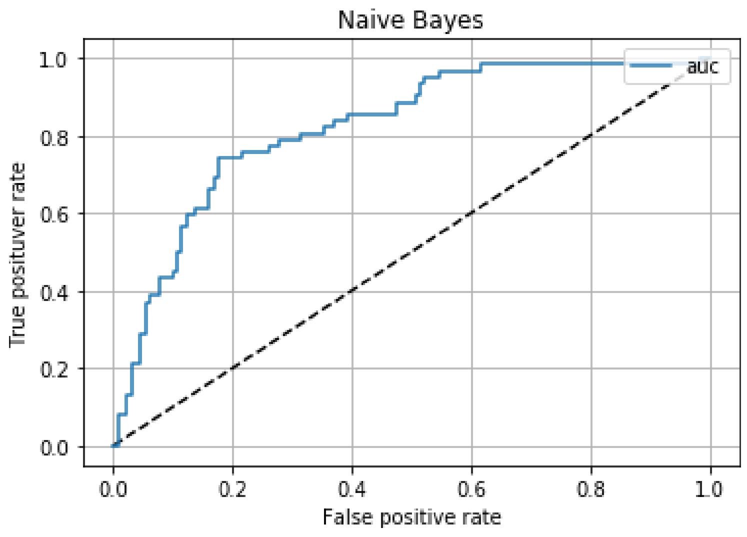


Figure 5: Naive Bayes AUC-ROC Curve

The AUC-ROC curve produced for Naive Bayes shows a slightly dipped curve, because of a higher number of incorrect predictions made by this classifier. Though the curve is still towards the top left meaning overall fairly accurate predictions were made. Naive Bayes managed to achieve an AUC score of 82%.

| | | Predicted | | |
|---|---|---|---|---|
| | | Negative(0) | Positive(1) | Total |
| Actual | Negative(0) | 117 | 13 | $117 + 13$ |
| | Positive(1) | 24 | 38 | $24 + 38$ |
| | Total | $117 + 24$ | $13 + 38$ | |

Table 11: MLP Confusion Matrix

Table 11 presents the confusion matrix for MLP. In comparison to the other classifiers MLP capability to predict the definite number of non-diabetic cases is lower then K-NN, but still maintaining a higher value then Naive Bayes. However this classifier obtained the highest value for predicting the definite number of diabetic cases, also shown as the sensitivity measure, with a result of 61%.

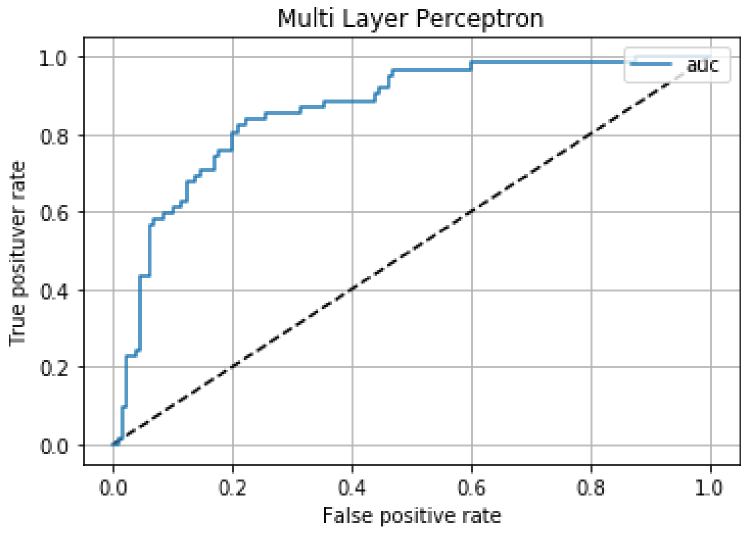


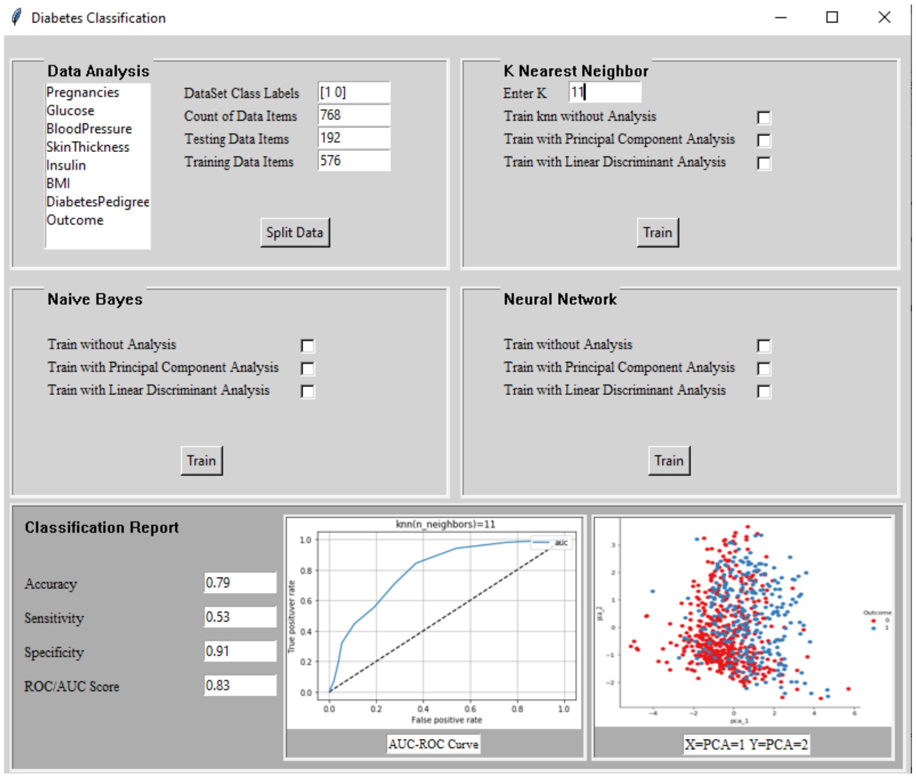
Figure 6: MLP AUC-ROC curve

The AUC-ROC curve for MLP outputted the highest curve compared to the other classifiers, for the reason that MLP obtained the most accurate predictions overall and less incorrect predictions. The AUC score for this classifier is 86%, the highest compared to all classifiers.

# 6 Implementation

## 6.1 Introduction

The key concept of the Graphical User Interface (GUI) is to train the chosen classifier models K-NN, Naive Bayes and MLP for a visual representation of a classification report, developed for comparison and evaluation for the diagnosis of diabetes. The interactive system also includes the choice of two feature reduction techniques Linear Discriminant Analysis (LDA) and Principle Component Analysis (PCA), allowing the user to test the analysis effect on the classifiers performance and compare results methodically. Figure 7 represents the GUI that was implemented to apply the research of this paper.


Diabetes Classification

**Data Analysis**

Pregnancies
Glucose
BloodPressure
SkinThickness
Insulin
BMI
DiabetesPedigree
Outcome

| | |
|---|---|
| DataSet Class Labels | [1 0] |
| Count of Data Items | 768 |
| Testing Data Items | 192 |
| Training Data Items | 576 |

Split Data

**K Nearest Neighbor**

Enter K: 11

Train knn without Analysis ☐
Train with Principal Component Analysis ☐
Train with Linear Discriminant Analysis ☐

Train

**Naive Bayes**

Train without Analysis ☐
Train with Principal Component Analysis ☐
Train with Linear Discriminant Analysis ☐

Train

**Neural Network**

Train without Analysis ☐
Train with Principal Component Analysis ☐
Train with Linear Discriminant Analysis ☐

Train

**Classification Report**

| | |
|---|---|
| Accuracy | 0.79 |
| Sensitivity | 0.53 |
| Specificity | 0.91 |
| ROC/AUC Score | 0.83 |

Figure 7: Fully engaged diabetes interactive system split into five sections consisting of Data Analysis, K-NN, Naive Bayes, MLP and classification report. The results shown are from the selection of K-NN trained without analysis and K=11.

## 6.2 Materials

### 6.2.1 Spyder Integrated Development Environment

Spyder Integrated Development Environment (IDE) was the chosen platform for all programs developed in this study. This IDE was preferred over others, as it is specifically used for scientific programming, especially Machine Learning (ML). This is due to the development platform combining the necessary libraries for data science in order to structure data. Two of the essential libraries to create such programs include NumPy and Matplotlib.

#### 6.2.2 Scikit-Learn

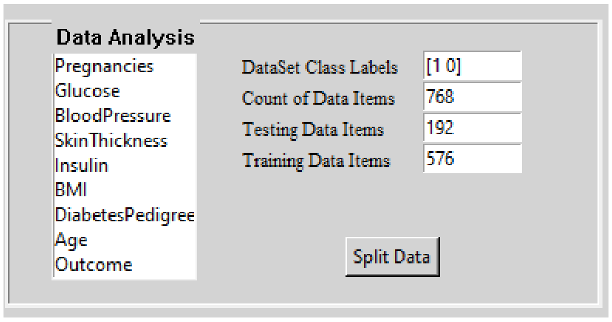
Scikit-Learn supply's many ML algorithms including different classifiers. Scikit-Learn was chosen for the development of the classifiers as they collaborate with the previous libraries mentioned, to structure the results of the classifiers for great documentation.

#### 6.2.3 Tkinter

Tkinter libraries were used for implementing the GUI as Tkinter provides many options for creating a framework for customization, as well as producing a cross-platform GUI supporting the following distributions Windows, Mac-OS , and Linux.

### 6.3 Product Design

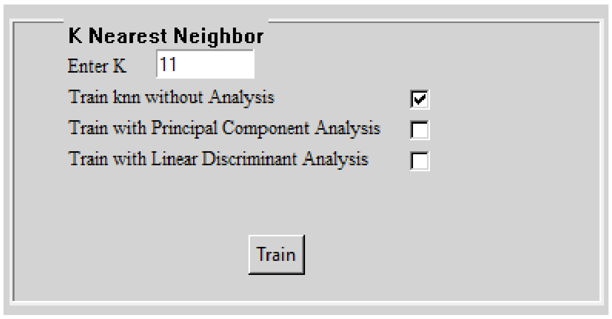
Figure 8 presents the diabetes classification GUI with all outputs engaged. The GUI is split into the four sections of, Data Analysis, K-NN, Naive Bayes, Neural Network and Classification Report.



| | |
|---|---|
| DataSet Class Labels | [1 0] |
| Count of Data Items | 768 |
| Testing Data Items | 192 |
| Training Data Items | 576 |

Figure 8: Data Analysis Section

Data Analysis loads the diabetes data set into the system presenting the list view of features, target variables and the number of data items within the data set. Clicking "Split Data" presents the split of the testing data items and training data items within the two text boxes. In addition, a scatter graph of the first two components is generated in the right-hand corner, intended for visualization to show how two features distinguish from one another.



**K Nearest Neighbor**

Enter K: 11

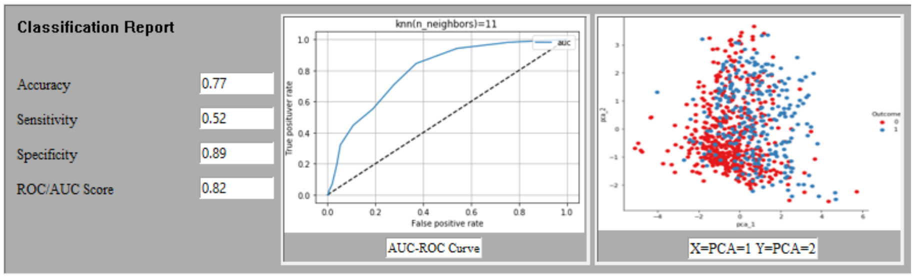
Train knn without Analysis [✓]

Train with Principal Component Analysis [ ]

Train with Linear Discriminant Analysis [ ]

Figure 9: Classifier Example

K-NN (Figure 9) contains a user entry field to test different values of K following three checkboxes, which enables the user to choose the way they want to train their modal, with or without analysis. Clicking "Train" once the user has completed the selection process outputs the modal performance within the Classification Report section.



**Classification Report**

|  |  |
|---|---|
| Accuracy | 0.77 |
| Sensitivity | 0.52 |
| Specificity | 0.89 |
| ROC/AUC Score | 0.82 |

Figure 10: Section of Classification Report

The Classification Report (Figure 10) is generated outputting results such as: Accuracy, Sensitivity, Specificity and ROC/AUC score. Furthermore, an AUC-ROC curve displays alongside the report representing the performance of the selected classifier decided by the user.

## 6.4 Code Demonstration

The following demonstrations shown below present how each section within the GUI is engaged with the user, and the sequential code behind this process. The code demonstrations that are to be mentioned are the key three processes that are linked, in order to output the chosen classifiers results. The first one to be demonstrated is the data analysis section and the main method that is responsible for this formation.

```
def preprocessing(self):
        #clear all boxes within data analysis
        self.clear_results()
        #outcome column with all rows
        x = self.df.drop('Outcome', 1)
        #just outcome column
        y = self.df['Outcome']

        #split training set 75% test set 25%
        self.x_train, self.x_test, self.y_train, self.y_test =
        train_test_split(x, y, test_size=0.25, random_state=0)
        train_set = len(self.x_train + self.y_train)
        test_set = len(self.x_test + self.y_test)

        #add testing and traing data into text boxs
        self.test_datat.insert("end", test_set)
        self.train_datat.insert("end", train_set)

        #scale data
        scaler = MinMaxScaler()
        self.x_train = scaler.fit_transform(self.x_train)
        self.x_test = scaler.transform(self.x_test)

        #add two components scatter graph
        self.img2=img2 =ImageTk.PhotoImage
        (Image.open("pca_scatter.png"))
        self.canvas_im2.create_image(0,0,anchor = "nw",
        image=img2)
```

The code snippet presented above is the method of data pre-processing which outputs its results into the data analysis section of the GUI, beginning with clearing all boxes so an input can be inserted. Then, proceeding to drop the target variable so training and testing can begin on the data. The split of these results are inserted into the text boxes to represent the value of records for each given set. Additionally, data scaling is performed on the data set which can be seen with the call to MinMax Scaler. Finally an image of 2 components in the form of a scatter graph using PCA is formed in the right-hand side. This method is called with the action of "Split data".

**Algorithm 1:** Pseudo code for control flow of user interaction for the choice of classification.

```
Result: Classification report
get values of checkboxes;
clear checkboxes;
Delete the prior AUC-ROC graph;
if K-NN without analysis is checked then
    perform K-NN and output results;
    generate the K-NN AUC-ROC graph;
else
    if K-NN with PCA is checked then
        perform K-NN with PCA and output results;
        generate the k-NN AUC-ROC graph for PCA;
    else
        K-NN with LDA is checked run K-NN with LDA and output results;
        generate the K-NN AUC-ROC graph for LDA;
    end
end
```

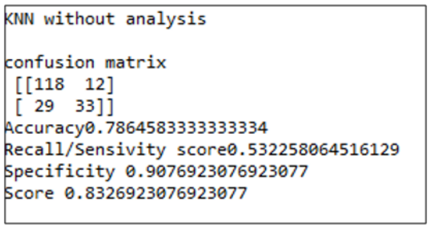
Each interaction with the selected classifier within the GUI can be demonstrated in Algorithm 1 seen above, with the use of Pseudo code. The result of this method is to output the corresponding classification report for the approach chosen by the user. The output of this code is done with the action of the value of checkbox and clicking "Train", which then calls the selected methods shown below.

```
def knn_own(self):
        self.clear_results()
        self.preprocessing()
        self.k_n_n(self.x_train, self.x_test)
        self.get_results(self.y_pred, self.y_score)
        self.get_results(self.y_pred, self.y_score)
        self.insert_results()

  def insert_results(self):
        #insert all results into text boxes in class report
        #round results to two decimal places
        self.class_acc.insert("end",self.acc.round(2))
        self.class_sense.insert("end", self.sens.round(2))
        self.class_spece.insert("end", self.spec.round(2))
        self.rocscore.insert("end", self.roc_score.round(2))
```

The two methods shown above are the final steps into providing the results for classification to be evaluated by the user, making a call to the relevant methods. Once a user has clicked "Train" the pre-processing and training of data will begin, then the classifier will run retrieving the relevant results, and inserting them into the correct report segment.

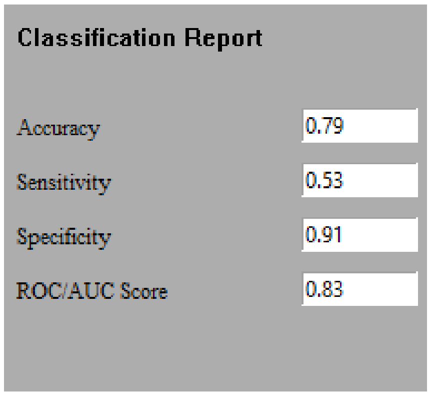
## 6.5 Testing



```
KNN without analysis

confusion matrix
 [[118  12]
 [ 29  33]]
Accuracy0.7864583333333334
Recall/Sensivity score0.532258064516129
Specificity 0.9076923076923077
Score 0.8326923076923077
```

Figure 11: Console output: K-NN without Analysis.



Classification Report

| | |
|---|---|
| Accuracy | 0.79 |
| Sensitivity | 0.53 |
| Specificity | 0.91 |
| ROC/AUC Score | 0.83 |

Figure 12: GUI Output: K-NN without Analysis rounded to 2 decimal places.

Prior to the implementation of the GUI a separate object-oriented program was developed to test the three chosen classifiers as different classes all inheriting from the parent class, which contains all data pre-processing and data analysis methods. The purpose of the program was to ensure all relevant aspects was included within the GUI to produce an informative, precise design.

To select the relevant aspects many data exploration techniques were carried out on the Pima Indians data set, identifying the most important statistics to include in the data analysis section of the GUI. Also, a range of graphs were developed to represent different visualization techniques, selecting the most accurate for interpreting results.

The object-orientated program was an efficient approach to exploring different features as few lines of code were to be refactored. Secondary, the initial program meant continuous checks could be carried out through the development of the GUI to reassure the same results were being outputted, represented by Figures 11-12.

# 7 Discussion

## 7.1 Conclusion

The prevalence of diabetes disease continues to increase within the global population, thus the need for extensive research to manage this disease is pivotal. Machine Learning models are constantly being compared for the prediction of diabetes for better awareness, to enhance medical treatment. Furthermore the comparison of classification models is essential in order to achieve sufficient results to present the best suitable classifier for a diabetes data set.

In this study, three classification models were trained on the Pima Indians data set. The chosen classifiers consisted of K-NN, Naive Bayes and Multi-layer perceptron (MLP). Each classifier was trained using 576 patient records for training and 192 for testing, where feature selection was applied, summarising the importance of seven features to be used for classification.

The overall results obtained concluded MLP to be the best suitable classifier for the Pima Indians data. MLP performance was evaluated with the following results, 81% accuracy, 90% sensitivity, 61% specificity, and 86% AUC-score. Secondly, K-NN performance based on all metrics obtained close results shown as, 79% accuracy, 91% sensitivity, 53% specificity and 83% AUC-score. Lastly, Naive Bayes performance proved to be substandard producing the results of, 78% accuracy, 88% sensitivity, 55% specificity and 82% AUC-Score. These results proved MLP proficiency to predict accurate predictions is substantial in comparison to K-NN and Naive Bayes. Even though MLP performance proved to the best suitable classifier for the diabetes data based on all performance metrics, it can be argued that K-NN outperformed the other classifiers, due to the reason of MLP training time is really slow, also discussed in previous studies [16]. For this study it was found that the training time for an MLP classifier took 6.3 seconds while Naive Bayes and K-NN training times were significantly faster. Taking only 0.4 seconds for K-NN and 0.08 seconds for Naive Bayes.

The chosen classifiers for this study were specifically selected based on their workings behind the training of data, in terms of the measures they use to make a prediction. To summarize these workings K-NN uses statistical measures to predict an outcome, by calculating the distance of the nearest neighbors to a data point and taking a majority vote to conclude the class. Naive Bayes makes a prediction depending on the probabilistic outcome of a class, given the varieties of attributes. MLP isn't based on either statistical or probabilistic measures, instead, it learns from the errors of processed data by comparing the perceptrons outcome to the expected outcome, updating the weights to ensure minimal error loss for the next process. By choosing three diverse classification algorithms, enhances the chance of unique individual results, as well as the conclusion of what training type is best for that set of data.

In regard to, the development of the GUI the decision to implement the classifiers through scikit-learn libraries was for the benefit of producing a fully functional comparative design. Scikit-learn provides many ML algorithms with few lines of code needed to complete a task, while still maintaining high performance. As a consequence, a more complex GUI was developed, due to fewer lines needed for classification. This meant more could be used on the ability of the interactive system, such as giving the user a wider range of approaches to address the problem. As a result a faster and more effective prediction tool was generated.

## 7.2 Reflection

This study intended to predict the outcome of a patient classifying as diabetic or non-diabetic and identifying the best suitable classifier to do so. In terms of how well this study addressed this problem, the performance metrics sensitivity and specificity can be used to determine their capability. These two result metrics measure the exact number of diabetic and non-diabetic cases within the diabetes data. The comprehensive results prove the chosen classifiers capability to predict all non-diabetic cases is greater as to the prediction of diabetes. The result obtained for sensitivity were as follows, MLP 61%, Naive Bayes 55% and K-NN 53%. Specificity obtained the results of, K-NN 91%, MLP 90% and Naive Bayes 88%, respectively. Predominantly, these classifiers proved to produce fairly good results with a high prediction rate for non-diabetic cases, however, their prediction rate for diabetic cases was considerably lower, hence more research is needed to improve this prediction rate. The findings of this study and the approach used was found to be beneficial, especially for Naive Bayes as the accuracy measure was higher compared to most previous studies [12,27,31].

The GUI developed in the process of this study was designed to present a comparative evaluation of classifiers that enables a user to explore the effects of feature analysis. All performance metrics evaluated in this study were presented within a classification report allowing a user to compare the performance of each classifier, based on numerous evidence to conclude the suitable proposal. Furthermore, visualization was also presented within the GUI, engaging a user to see the visual effects of their performance for a better understanding of the chosen classifiers. This allowed more investigation into different ML algorithms not just the study of classifiers, gaining more knowledge on principal component analysis and linear discriminant analysis, that could be applied further to future studies.

## 7.3 Future Exploits

Much work has compared numerous classifiers to predict diabetes using a wide range of techniques. Although, there is limited research using a multi-combination of classifiers for this data set. The combination of different classifiers builds upon a prediction using the class prediction outputted from the previous individual classifiers, using it as input for the final classification task. Further research could investigate this method to try to improve prediction rates.

Feature selection proved to be beneficial within this study, hence potential studies could further examine the selection of features used in this paper to improve the classification of diabetes even more. The consideration to integrate other attributes from different data sets that contribute to the outcome of diabetes could have an improvement in the performance of the classifiers. Additionally, the exploration of different data pre-processing techniques could improve classification performance further, with the use of discretization to produce strong quality input data.

Finally a different selection of classifiers could produce improved results, as well as better identification for the suitable classifier for the diabetes data set. Future research into the comparison of many Artificial Neural Networks could significantly enhance the capability to predict both diagnoses, as the ”learning” approach of a classifier trained on the diabetes data set proved to be successful in this paper, shown in MLP results.

## References

[1] Alford. A,2017"Sensitivity, Specificity, and Predictive Values: Foundations, Pliabilities, and Pitfalls in Research and Practice".Frontiers in Public Health. Vo:5. P:307. Accessed on 28 March 2020.https://www.frontiersin.org/article/10.3389/fpubh.2017.00307

[2] Alghamdi M, Al-Mallah M, Keteyian S, Brawner C, Ehrman J & Sakr S 2017. "Predicting diabetes mellitus using SMOTE and ensemble machine learning approach: the Henry Ford Testing (FIT) project" *PLoS ONE.* 12:e0179805. doi: 10.1371/journal.pone.0179805. Assessed on 4 March 2020, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5524285/

[3] Amendolia S.R, Cossu G, Ganadu M.L, Golosio B, Masala G.L & Mura G.M 2003. "A comparative study of K-Nearest Neighbour, Support Vector Machine and Multi-Layer Perceptron for Thalassemia screening". *Chemometrics and Intelligent Laboratory Systems.* No 69 13-20. Accessed on 26 Sep 2019, http://www.sciencedirect.com

[4] Anderson H.B, 2016. "Preprocessing in Data Science (Part 1): Centering, Scaling, and KNN." DataCamp. Accessed 4 Feb 2020, https://www.datacamp.com/community/tutorials /preprocessing-in-data-science-part-1-centering-scaling-and-knn

[5] Brownlee J, 2016. "Naïve Bayes for Machine Learning". Machine Learning Algorithms. Accessed on 22 Jan 2020, https://machinelearningmastery.com/naive-bayes-for-machine-learning/

[6] Brownlee J, 2019. "How to use Data Scaling Improve Deep Learning Model Stability and performance" Machine Learning Mastery. Accessed on 25 March 2020. https://machinelearn ingmastery.com/how-to-improve-neural-network-stability-and-modeling-performance-with -data-scaling/

[7] Chaitrali S.D & Sulabha S.A, 2012. "Improved Study of Heart Disease Prediction System using Data Mining Classification Techniques" *International Journal of Computer Systems* Vol: 47, No:10, pp:44-48. Accessed on 20 Nov 2020.

[8] Dong L, Li X & Xie G, 2014. "Nonlinear Methodologies for Identifying Seismic Event and Nuclear Explosion Using Random Forest, Support Vector Machine, and Naïve Bayes Classification". *Scaling, Self-Similarity and Systems of Fractional Order, 1085-3375.* Accessed on 22 Jan 2020, https://doi.org/10.1155/2014/45913

[9] Esbensen K.H & Geladi P, 2009. "Principle Component Analysis: Concept, Geometrical Interpretation, Mathematical Background, Algorithms, History, practice". *Comprehensive Chemometrics.*v.2 211-226. Accessed on 2 Feb 2020, https://www.sciencedirect.com/science /article/pii/B9780444527011000430a0005

[10] Forouhi N.G & Warehame N.J, 2014 "Epidemiology of diabetes" Medicines (Abingdon) 42,698 -702. Accessed on 3 March 2020, https://www.ncbi.nlm.nih.gov/pmc/articles /PMC4282306/

[11] Harrison O, 2018. "Machine Learning Basics with the K-Nearest Neighbors Algorithm".

Towards Data Science. Accessed on 14 Jan 2020, https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a71d01761

[12] Hina S, Shaikh A & Sattar SA 2017. “Analyzing Diabetes Datasets using Data Mining”. *Journal of Basic Applied Sciences.* No. 13,,pp:466-371. Accessed 26 Sep 2019, http://www.semanticsscholar.org

[13] Hu LY, Huang M.H, Ke, SW & Tsai C.H, 2016. “The distance function effect on k-nearest neighbour classification on medical datasets” SpringerPlus. No.1304. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4978658/

[14] Jeatrakul P &Wong K, 2009. “Comparing the Performance of Different Neural Networks for Binary Classification Problems” *International Symposium on Natural Language Processing.* PP:111-115. Accessed on 23 Jan 2020, http://researchrepository.murdoch.edu.au/view/author/Wong, Kevin (Kok Wai).html

[15] Kain N.K, 2018. “Understanding of Multilayer Perceptron (MLP)”. Medium. Accessed on 10 Feb 2020, https://medium.com/@AI_with_Kain/understanding-of-multilayer-perceptron-mlp-8f179c4a135f

[16] Kaur Ha & Kumari V 2018. “Predictive modelling and analytics for diabetes”. *Applied Computing and Informatics.* pp:2210-8327. Accessed on 26 Sep 2019, http://www.sciencedirect.com

[17] Kang N, 2017. “Multi-Layer Neural Networks with Sigmoid Function- Deep Learning for Rookies (2)”. Towards Science. Accessed on 6 Feb 2020,https://towardsdatascience.com/multi-layer-neural-networks-with-sigmoid-function-deep-learning-for-rookies-2-bf464f09eb7f

[18] Kim K.S, Choi H.H, Moon C.S & Mun C.H 2011. “Comparison of k-nearest neighbour, quadratic discriminant and linear discriminant analysis in classification of electromyogram signals based on the wrist -motion direction”. *Current Applied Physics.* No. 3, pp 740-745. Accessed on 25 Jan 2020, sciencedirect.com

[19] Kotsiantis S.B, Kanellopoulos D.K & Pintelas P.E 2007. “Data Preprocessing for Supervised Learning”. *International Journal of Computer, Electrical, Automation, Control and Information Engineering.* Vol:1, No:12. Accessed on 3 Feb 2020.https://www.researchgate.net /publication/228084519_Data_Preprocessing_for_Supervised_Learning

[20] Kumar R, 2018. “Understanding Principle Component Analysis”. Medium. Accessed on 4 Feb 2020, https://medium.com/@aptrishu/understanding-principle-component-analysis-e32be0253ef0

[21] Nurhayati A.N.H, 2014. “Implementation of Naïve Bayes and K-Nearest Neighbor Algorithm for the Diagnosis of Diabetes Mellitus”. Applied Computational Science. Accessed on 23 March 2020. http://www.wseas.us/e-library/conferences/2014/Malaysia/ACACOS/ACA COS-15.pdf

[22] Pratt H, Coenen F, Broadbent D.M, Harding P.S & Zheng Y. “Convolutional Neu-

ral Networks for Diabetic Retinopathy" *Procedia Computer Science.* 90. 200-205. Accessed on 4 March, http://www.sciencedirect.com/science/article/pii/S1877050916311929

[23] Rouse M, 2016. "Unsupervised Learning". What Is Tech Target. Accessed on 1 March 2020, https://whatis.techtarget.com/definition/unsupervised-learning

[24] Sa'di S, A Maleki, Hashemi R, Panbechi Z & Chalabi K, 2015. "International Journal on Computational Science & Applications". Vol:5. No:5. Accessed on 10 March 2020, https://s3.amazonaws.com/

[25] Sharma P 2018. "The Ultimate Guide to 12 Dimensionality Reduction Techniques (with Python codes)" Analytics Vidhya. Accessed 21 Jan 2020, http://www.analyticsvidhya.com/blog/2018/08/dimensionality-reduction-techniques-python/

[26] Sharma S, 2019 "Principal Component Analysis Explained". Medium. Accessed on 4 Feb 2020, https://medium.com/@the_change/principal-component-analysis-explained-560df8832b93

[27] Sisodia D & Sisodia S 2018. "Prediction of Diabetes using Classification Algorithms" *Procedia Computer Science.* Vo 132, pp: 1578-1585. Accessed on 22 Jan 2020, https://doi.org/10.1016/j.procs.2018.05.122.

[28] Waseem M, 2019. "How to Implement Classification in Machine Learning". Data Science with Python. Accessed 29 March 2020. https://www.edureka.co/blog/classification-in-machine-learning/

[29] Vanderplas J 2016. "In Depth: Principal Component Analysis". *Python Data Science Handbook.* No 5. Accessed on 21 Jan 2020, http://www.jakevdp.github.io

[30] Yan H, Jiang Y, Zheng J, Peng c  LI Q, 2006. "A multilayer perceptron-based medical decision support system for heart disease diagnosis". *Expert Syst. Appl.*272-281. Accessed on 02 March 2020 , https://www.semanticscholar.org/paper/A-multilayer-perceptron-based-medical-decision-for-Yan-Jiang/2b4924408cc74b407afc8f5fa97e82

[31] Zehra A, Asmawaty T & Aznan M.A.M, 2014. "A Comparative study on the pre-processing and mining of pima Indian Diabetes Dataset". Accessed on 01 April 2020. http://umpir.ump.edu.my/id/eprint/5035/1/31-UMP.pdf

[32] Zhang M.L, Pena J.M & Robles V, 2009. "Feature selection for multi-label naïve Bayes classification" *Information Sciences.* Vol:179, No:19, PP:3218-3229. Accessed on 23 Nov 2020, https://doi.org/10.1016/j.ins.2009.06.010.

# A Comparative Analysis of Machine Learning Algorithms for Diabetic Data Classification

## Project Overview

- The comparison of three classifiers being, K Nearest Neighbors (K-NN), Naïve Bayes and Multilayer Perceptron (MLP) to conclude the best suitable classifier for the prediction of diabetes using the Pima Indians data set. Along with a developed Graphical User Interface (GUI) as a diagnostic prediction tool, to present these findings.

## Problem?

- Diabetes is one of the most common diseases, affecting more then 4 million people [1] and is constantly on the rise. As a result, more time and money is being spent on this disease by the health care industry.
- Faster diagnoses is needed and more convenient ways of treatment, as well as, constant research into the cause of the disease as there is no cure.

## Challenges?

- Insufficient results. Much previous work has studied the Pima Indians data set, using a similar range of  classification models, which is why it is important to take a unique approach to provide beneficial results for future research to build upon.
- A slow and ineffective GUI. There are many processes within this design meaning longer execution time which is why it is important to choose high performance  libraries and refactored code.

## The aim?

- The main aim is to identify the best suitable classifier for the prediction of this disease. Secondly to explore and identify other techniques that could be used to enhance a classifiers performance for the diabetes data set. Lastly, to develop a fully functional GUI for a user to interact with the comparison of classifiers for better evaluation.

## How?

- By substantial amounts of research.
- Testing different techniques, measures and parameters for example: data evaluation, tuning parameters, no of features, range of performance metrics and visualization.
- The selection of vital methods and visuals to be implemented in the GUI.
- Selection of high performance libraries used within the GUI.

## The Proposed Method?

It was found that MLP proved to be the best suitable classifier for the diabetes data set obtaining the highest accuracy score of 81% followed by K-NN 79% and lastly Naïve Bayes estimating 78%.

In addition a reduction of features was applied using principle component analysis, concluding seven instead of eight were classed as the ones with highest value of importance to the data.

## Implementation

- The GUI presents the following features:

- List view of attributes
- The Target variable classes
- The total of patient records
- Value of testing set
- Value of training set
- Scatter graph of two features with PCA applied
- Classification report
- AUC-Roc curve

## Implementation

- The GUI allows the interaction of:

- Testing different values of K
- Choice of Training KNN with or without analysis
- Choice of Training Naïve Bayes with or without analysis
- Choice of Training  MLP with or without analysis

The choice of two analysis to be compared on classifiers include Principle component analysis and Linear Discriminate analysis.

## Future work?

Future studies could build upon this research to enhance classifiers performance by:

- Integration of features from other data sets to add to the contribution to diabetes.
- Comparison of different Artificial Neural Networks
- Apply non-linear dimension reduction techniques
- Multi-Combination of classifiers

[1]Anon 2019.”Diabetes Prevalence”. Diabetes. Accessed on 20 April 2020.https://www.diabetes.co.uk/diabetes-prevalence.html