Unilag Journal of Mathematics and Applications,
Volume 3, (2023), Pages 53--70.
ISSN: 2805 3966. URL: http://lagjma.edu.ng

A COMPARATIVE ANALYSIS OF SOME SELECTED MACHINE LEARNING ALGORITHMS FOR CLASSIFICATION AND PREDICTION OF DIABETES TYPES

ROTIMI OGUNDEJI* AND HOPE ADEGOKE

ABSTRACT. Diabetes mellitus is one of the most common human diseases worldwide and may cause several health-related complications. It is responsible for considerable morbidity, mortality and economic loss. A timely diagnosis and prediction of this disease could provide patients with an opportunity to take the appropriate preventive and treatment strategy. To gain an understanding of risk factors, the study explores the comparison of six classification algorithms along with the artificial neural network algorithm in the prediction of diabetes types. The analysis of data from patient records shows four main predictors of type 1 & type 2 diabetes: Age, Ulcer Duration, White Blood Cell (WBC) and Erythrocyte Sedimentation Rate (ESR). All the classification algorithms based on accuracy, precision, recall, and F-measure show good results for the parameters with accuracy greater than 95%. However, random forest and k-nearest neighbor algorithms provided 99% accuracy for the train/test split. Thus, these models can be applied to make a reasonable classification of type 1 and type 2 diabetes. Hence, the study ensures a timely diagnosis and prediction of this disease for appropriate preventive and treatment strategy.

1. INTRODUCTION

Diabetes is a common condition that affects millions of individuals throughout the world. Diabetes type categorization is critical for disease therapy and management and for the treatment and control of the condition. However, contemporary diabetes categorization methods are frequently reliant on clinical observation and laboratory testing, which can be subjective and error-prone [1]. Machine learning techniques have been presented as a method of automating and enhancing diabetes categorization accuracy. However, there is no agreement on which machine learning method is best suited for this task.

Diabetes, a metabolic illness defined by high blood sugar levels, is divided into Type 1 and Type 2 diabetes. Type 1 diabetes, also referred to as insulin-dependent, juvenile, or childhood-onset diabetes, is caused by insufficient insulin production and necessitates daily insulin injections. As of 2017, around 9 million individuals worldwide had type 1 diabetes, with most residing in high-income nations. Despite extensive research, the root cause of the disease remains unknown, and no prevention methods have been identified [2]. Type 2 diabetes, formerly known as non-insulin-dependent or adult-onset diabetes, occurs due to the body's inefficient utilization of insulin. Over

2010 *Mathematics Subject Classification*. Primary: 22E30. Secondary: 58J05.
Keywords and phrases. Algorithm Accuracy, Classification Models, Diabetes Types, Neutral Network,

Submitted: December 29, 2023. Revised: May 06, 2024. Accepted: May 14, 2024. *Correspondence

95% of people with diabetes have type 2 diabetes and it is mostly linked to excessive body weight and a lack of physical activity [2], [3].

The symptoms of type 2 diabetes, while resembling those of type 1 diabetes, are usually milder, which can delay the diagnosis of the disease and lead to complications. Although it used to affect only adults, it is becoming more prevalent in children. Symptoms of type 2 diabetes include polyuria (excessive urination), polydipsia (excessive thirst), constant hunger, weight loss, changes in vision, and fatigue. These symptoms may manifest abruptly [2], [3].

In the early 2000s, [4] applied logistic regression and decision trees for diabetes categorization. Small sample numbers and a lack of advanced feature selection methods hindered these early investigations. However, as data availability and computer resources have risen, more complex techniques such as neural networks and support vector machines have been used to classify diabetes. Recent research employing multiple datasets examined the effectiveness of several machine learning algorithms on diabetes categorization tasks. According to this research, ensemble approaches, which aggregate the predictions of numerous models, perform well on diabetes classification tasks [5].

Several research has demonstrated that random forest and gradient boosting algorithms yield good accuracy rates [6]. Deep learning models have shown potential in diabetes categorization tasks, with some research claiming accuracy rates equivalent to or greater than typical machine learning methods [7]. The interpretability of the model is a factor that is frequently examined when comparing classification methods. Some algorithms, like decision trees and logistic regression, are easier to understand than others, unlike neural networks. This can be useful in a medical setting since it helps doctors to identify the characteristics that are most essential for diabetes type categorization. Another factor to consider when comparing classification algorithms is their ability to manage unbalanced datasets. Diabetes classification datasets frequently feature unequal class distributions, with a greater proportion of samples belonging to one class (i.e. Type 2 diabetes) than the other (i.e. Type 1 diabetes) [3]. This can result in skewed model performance, with the dominant class predicted more frequently than the minority class. To solve this issue, numerous oversampling and undersampling strategies, such as SMOTE and Tomek links, have been developed, which may be used to balance the class distribution before training a model [8]. To summarize, the use of classification algorithms for diabetes prediction is an active field of study that has seen significant development in recent years. Deep learning models and ensemble approaches, notably random forest and gradient boosting, have shown promise in reaching high accuracy rates [9]. However, interpretability and dealing with skewed datasets are also key considerations to consider when choosing a diabetes classification method.

[10] developed two approaches for predicting type 2 diabetes using neural networks classifiers. [11] examined the performance of various machine learning algorithms for diabetes detection using the MATLAB classification learner tool. The study included decision tree, discriminant analysis, SVM (Support Vector Machine), k-NN (k-Nearest Neighbor), Logistic regression, and ensemble learners, with a total of 26 classifiers being considered. The results are evaluated using a 10-fold cross-validation method and the average classification accuracy was used as the performance measure. [12] used SVM with different kernel functions to classify diabetes. Different kernel functions such as linear, polynomial, and radial kernel were implemented, and the prediction accuracy was calculated using the confusion matrix. [1] applied various machine-learning techniques to three different disease datasets for disease prediction. Five algorithms were compared: decision tree, logistic regression, random forest, adaptive boosting, and support vector machine. [13] developed a web application using Tensorflow which required patient data for

diagnosis and techniques such as SVM, ANN (Artificial Neural Network), KNN, and Naive Bayes for the successful prediction of diabetes. [14] proposed and developed a model for diagnosing type 2 diabetes using the ELM (Extreme Learning Machine) technique. This model would assist medical experts in forecasting type 2 diabetes. More recent literatures on machine learning algorithms for classification and prediction can be seen in [15], [16], [17], [18] and [19].
Based on their performances in previous research, this study selected six classification algorithms for diabetes categorization. The study aims to provide insights into the factors that contribute to the prediction of diabetes. With a comparative analysis, this study identifies the most accurate classification algorithm for predicting the types of diabetes and understand each algorithm's strengths and limitations to inform on effective strategies for early diagnosis and management of the disease.

2. MATERIALS AND METHODS

2.1 Data Source and Description

Dataset of patient records that included information about their medical history, lifestyle, and laboratory test results were collected from Lagos State University Teaching Hospital (LASUTH), Nigeria. The dataset comprises both males and females who are at least 19 years old. The dataset contains information about 336 patients and their corresponding five unique attributes such as Age, Ulcer duration, White blood cell, Erythrocyte Sedimentation Rate, and Diabetes mellitus type. The "Diabetes mellitus type" attribute is taken as a dependent or study variable, and the remaining four attributes are taken as independent/feature variables. The diabetes attribute "Diabetes mellitus type" consists of a binary value where 0 means "type 1", and 1 implies "type 2". This research use data mining and machine learning algorithms to predict whether a patient has type 1 or type 2 diabetes with enhanced accuracy.

2.2 Method of Analysis

The method of analysis explores the comparison of six classification algorithms in the prediction of diabetes types. Figure 1 below summarises the analysis, the first step starts with the data preprocessing through applied algorithm, performance evaluation on various measures, comparative analysis based on accuracy to give the final result.

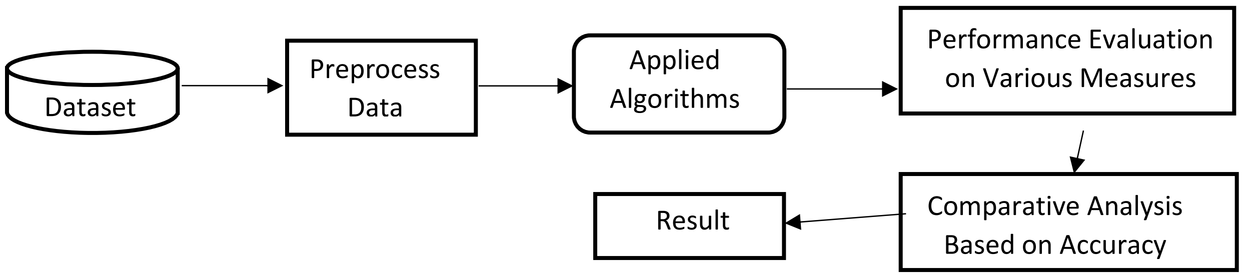

Dataset
Preprocess Data
Applied Algorithms
Performance Evaluation on Various Measures
Comparative Analysis Based on Accuracy
Result

Figure 1: Proposed model diagram for analysis

2.2.1 Data Cleaning

Data cleaning involves removing or correcting any errors or inconsistencies in the data. This can include dealing with missing values, outliers, or duplicate data. Using excel and SPSS, we got the missing values in the datasets, we replaced the missing value with the corresponding mean value.

2.2.2 Feature selection

Feature selection is the process of selecting the most relevant features for the classification task. This can be done by using techniques such as correlation analysis or mutual information.
Feature selection is the process of selecting a subset of features (i.e., variables, predictors, inputs) from a larger set of features to improve the performance of a machine learning model. The goal of feature selection is to identify the most relevant features that can be used to predict the target variable. There are several ways to perform feature selection, including Filter methods, Wrapper methods, Embedded methods, Dimensionality reduction methods, Domain knowledge-based methods, and Genetic algorithms [20].

2.2.3 Data Augmentation

Data augmentation is the process of creating new data points from existing ones by applying various transformations such as rotation, flipping, and zooming. This is useful when the dataset is small and to prevent overfitting. Since our dataset is imbalanced the data will have to be augmented. An imbalanced dataset is one in which the classes (or categories) are not evenly represented. In our research where the goal is to predict whether a patient has type 1 or type 2 diabetes, only 4% of the patients in the dataset have type 1 diabetes, therefore the dataset is imbalanced because the majority class is far more prevalent than the minority class. Imbalanced datasets might be challenging since machine learning algorithms are frequently intended to perform effectively when the classes are balanced. When the classes are imbalanced, the method will often have a greater accuracy rate just by predicting the majority class all of the time, although not useful for making predictions. One effective method of dealing with imbalanced data is by using SMOTE NC technique. This may be expressed mathematically as:

$$Xs = Xm + (rand * (Xn - Xm)) \tag{1}$$

where Xs is the synthetic sample, Xm is the original minority class sample, Xn is the selected nearest minority class neighbor, and rand is a random number between 0 and 1.
After SMOTE, NC is used to eliminate synthetic samples that are too similar to existing minority samples. The approach computes the Euclidean distance between each synthetic sample and all existing minority samples. The synthetic sample is eliminated if the distance between it and an existing minority sample is smaller than a certain threshold. The Euclidean distance formula is:

$$d(x,y) = \sqrt{\sum_{i=1}^{k}(x_i - y_i)^2} \tag{2}$$

The final dataset will have a balanced class distribution, and the synthetic examples will be less likely to be misclassified as actual samples, enhancing the model's overall performance [21].

2.2.4 Dataset splitting

Dataset splitting is the process of dividing a dataset into multiple subsets to test the performance of a machine learning model. The most common way of splitting a dataset is into a training set and a test set. We split the dataset into training and test sets using stratified sampling. The training set will be used to train the classification algorithms, and the test set will be used to evaluate their performance. The training set is used to train the model, and it usually represents most of the data (e.g., 80%). The model uses the training set to learn the patterns and relationships in the data. The test set is used to evaluate the performance of the trained model, and it usually represents a smaller portion of the data (e.g., 20%). The test set is used to assess how well the model will perform on unseen data. Another way of splitting a dataset is to use cross-validation. In cross-validation, the data is split into multiple subsets, typically referred to as folds. Each fold is used as the test set in turn, while the remaining folds are used as the training set. This allows for a more robust evaluation of the model, as the model is trained and tested on different subsets of the data.

2.3 Algorithm selection

Some set of classification algorithms were chosen to compare, including logistic regression, support vector machines, K-mean nearest neighbor, decision trees, random forest, and naïve Bayes.

2.3.1 Logistics Regression

Logistic regression is a type of generalized linear model (GLM) that predicts the likelihood of a binary outcome using one or more independent variables, which originated in the 1940s by Joseph Berkson [22]. It is a method of modeling the connection between a dependent variable and one or more independent variables by fitting the data to a logistic function. The logistic function is a sigmoid function that converts every real-valued integer to a probability value between 0 and 1. In logistic regression, the model is trained on a dataset with a binary outcome variable and continuous or categorical independent variables. Given the values of the independent variables, the model predicts the likelihood of the result being one (e.g., success, positive, etc.). Once trained, the model may be used to estimate the likelihood of an outcome for additional data points.

$$p(y=1|x) = \frac{1}{(1+e^{-(\beta_0+\beta_1 x_1+\beta_2 x_2+\cdots+\beta_n x_n)})} \qquad (3)$$

where p(y=1|x) is the probability of the outcome being one given the values of the independent variables x, β_0, β_1, β_2, ..., β_n are the parameters of the model that are learned from the data, and x_1, x_2, ..., x_n are the independent variables. The coefficients (β_0, β_1, β_2, ..., β_n) are estimated using maximum likelihood estimation (MLE) or maximum a posteriori estimation (MAP) during the training process. These coefficients represent the relationship between the independent variables and the outcome variable.

2.3.2 Support Vector Machine (SVM)

SVMs are based on the idea of locating a hyperplane that best separates the different classes in the data. The hyperplane is chosen such that it has the greatest margin, which is the distance between the hyperplane and the nearest data points from either class. The support vectors are the nearest data points. SVMs may also be used for non-linear classification by translating the input data into a higher-dimensional space that contains a linear boundary. This is accomplished using a technique known as the kernel trick, which transfers the input data into a higher-dimensional feature space without actually computing the data's coordinates in that space. The polynomial kernel and the radial basis function (RBF) kernel are two popular kernels used in SVMs [23].

2.3.3 K-Nearest Neighbors Algorithm

The k-Nearest Neighbors (KNN) algorithm is a non-parametric approach to classification that is often straightforward yet effective. To classify a given data record t, the algorithm retrieves its k-nearest neighbors, which form a neighborhood around t. Typically, the classification of t is determined through majority voting among the data records in this neighborhood, with or without considering distance-based weighting. However, the success of the KNN method depends heavily on the appropriate selection of k. Essentially, the choice of k introduces a bias to the algorithm. There exist various methods to choose an appropriate k value, with one simple approach being to run the algorithm multiple times using different k values and selecting the k value that yields the best performance [24].

2.3.4 Decision Tree

A decision tree is a tree-based model that predicts or makes choices depending on some criteria. It is a graphical depiction of all feasible decisions depending on specified criteria. The tree structure is made up of core nodes that represent data aspects or qualities, branches that reflect decision rules, and leaf nodes that provide the result or prediction. The tree is constructed by recursively separating the data into subgroups depending on the features or attributes' values. This method aims to generate subsets that are as homogenous to the target variable as feasible. The subsets are then separated again, and so on until a stopping requirement is fulfilled. Once generated, the tree may be used to forecast the result of additional data points by traversing it from root to leaf node. The decision tree is a popular algorithm for both classification and regression problems. In classification, the target variable is categorical, and the tree is used to predict the class of a new data point. In regression, the target variable is continuous, and the tree is used to predict the value of the target variable.

The impurity measure is used to assess the quality of a split, or how successfully data is segregated into groups based on feature or attribute values [25]. The following are the most widely used impurity measurements for decision trees:

Gini impurity: It is used for classification problems and is defined as the probability of incorrectly classifying a randomly chosen element from the set, it is calculated as

$$Gini\ Impurity = 1 - \sum_i (p_i)^2 \tag{4}$$

Entropy: It is also used for classification problems and is defined as the impurity of the set, it is calculated as:

$$Entropy(S) = -\sum_{c \epsilon C} p(c) log_2 p(c) \tag{5}$$

S represents the data set that entropy is calculated;
c represents the classes in set, S;
p(c) represents the proportion of data points that belong to class c to the number of total data points in set, S.

2.3.5 Random Forest

A random forest is an ensemble of decision trees. A decision tree is a flowchart-like tree structure, where each internal node represents a feature (attribute), each branch represents a decision rule, and each leaf node represents the outcome. The goal of a decision tree is to learn a model that can accurately predict the target variable based on the input features.

To construct a random forest, multiple decision trees are trained on different subsets of the data, also known as bootstrap aggregating or bagging. The subsets are created by randomly selecting data samples from the original dataset with replacements. This means that some samples may be repeated in a subset while others may be omitted. Each decision tree is trained on a different subset, so the trees are different from one another. Once all the decision trees are trained, the random forest makes a prediction by averaging the predictions of all the trees for regression problems or by voting for classification problems. Because each tree was trained on a different subset of the data, the trees are likely to make different predictions and capture different patterns in the data. The final prediction of the random forest is an ensemble of the predictions of all the trees, which is generally more robust and accurate than a single decision tree. Another feature of the random forest is the random feature selection, which means that when splitting a node during the training of each decision tree, the algorithm is not considering all the features but a random subset of them. This has the effect of decorrelating the trees and making the ensemble stronger [26].

2.3.6 Naïve Bayes

Naïve Bayes is a classification method that predicts using Bayes' theorem. The "naive" component of the term refers to the assumption that the data characteristics are independent of one another, which is not necessarily true in real-world data. Despite this assumption, the technique frequently performs well in practice, and it is especially beneficial for huge datasets. The method comes in numerous types, including Gaussian Naive Bayes, Multinomial Naive Bayes, and Bernoulli Naive Bayes. They're utilized in things like text categorization, spam filtering and sentiment analysis [27]. The main idea behind Naive Bayes is to utilize Bayes' theorem to compute the likelihood of a specific class given some input data.
Bayes' theorem:

$$P(C|X) = P(X|C) * P(C) / P(X) \tag{6}$$

where $P(C|X)$ is the posterior probability of class C given instance X, $P(X|C)$ is the conditional probability of instance X given class C, P(C) is the prior probability of class C, and P(X) is the marginal probability of instance X [28].
Naive assumption:
The algorithm assumes that the features of an instance are conditionally independent given the class label. This means that:

$$P(X|C) = P(x_1|C) * P(x_2|C) * \ldots * P(x_r|C) \tag{7}$$

where $x_1, x_2, \ldots, x_r$ are the features of the instance.
(i) Maximum a posteriori (MAP) estimation:
The algorithm estimates the conditional probabilities $P(x_i|C)$ and the prior probabilities P(C) from the training data using the maximum a posteriori (MAP) estimation. This involves estimating the parameters of a probability distribution function that best fits the data [29].
(ii) Class prediction:
To predict the class of a new instance X, the algorithm computes the posterior probability of each class using Bayes' theorem, and selects the class with the highest probability:

$$P(C_i|X) = P(X|C_i) * P(C_i) / P(X) \tag{8}$$

The algorithm computes the product of the conditional probabilities $P(x_i|C)$ for each feature of the instance and multiplies it by the prior probability $P(C)$ for each class. The denominator $P(X)$ is a normalization constant that ensures that the posterior probabilities sum to 1 [29].

2.4 Algorithm Comparison

Algorithm comparison is the process of evaluating and comparing the performance of different algorithms on a specific task. This is done by training each algorithm on the same dataset and then measuring their performance using a set of metrics. There are several metrics commonly used for comparing algorithms, including:

2.4.1 Accuracy: Accuracy is the most intuitive metric for classification tasks. It is defined as the proportion of correct predictions made by the model out of all the predictions made [30]. It is calculated as:

$$Accuracy = (Number\ of\ Correct\ Predictions) / (Total\ Number\ of\ Predictions) \tag{9}$$

2.4.2 Confusion Matrix: A confusion matrix is a table that describes the performance of a classification algorithm (see table 1). It gives the number of correct and incorrect predictions made by the model on a test set. The confusion matrix can be used to compute various evaluation metrics such as precision, recall, and F1 score [31].

Table 1: Confusion Matrix

	Predicted type 1 DM	Predicted Type 2 DM
Actual Type 1 DM	TN	FP
Actual Type 2 DM	FN	TP

2.4.3 Precision: Precision is a measure of the accuracy of the classifier when it predicts the positive class. It is defined as the proportion of correct positive predictions made by the model out of all the positive predictions made [30] It is calculated as:

$$Precision = (True\ Positives) / (True\ Positives + False\ Positives) \tag{10}$$

2.4.4 Recall: Recall is a measure of the ability of the classifier to find all the positive instances in the test set. It is defined as the proportion of correct positive predictions made by the model out of all the actual positive instances in the test set [30]. It is calculated as:

$$Recall = (True\ Positives) / (True\ Positives + False\ Negatives) \tag{11}$$

2.4.5 F1 Score: The F1 score is a measure of the balance between precision and recall. It is calculated as the harmonic mean of precision and recall. It is a good metric to use when you want to balance precision and recall [30]. The F1 score is calculated as:

$$F1\ Score = 2 * (Precision * Recall) / (Precision + Recall) \tag{12}$$

AUC-ROC Curve: AUC-ROC (Area Under the Receiver Operating Characteristic (ROC) Curve) is a popular metric for evaluating the performance of binary classifiers. It is a curve that plots the true positive rate against the false positive rate at various classification thresholds. The AUC-ROC curve is useful because it is insensitive to class imbalance and is a good metric to use when you have imbalanced classes in your dataset [30].

2.5 Artificial Neural Network

An artificial intelligence (AI) system known as a neural network is based on how the human brain works and is structured. It is created with the ability to spot patterns and base judgments on such patterns. Predictive modeling, image identification, and natural language processing are just a few of the activities that neural networks are utilized for. The artificial neurons that make up a neural network are several linked nodes that collaborate to process information. Each neuron processes information from other neurons and outputs the results of straightforward mathematical computations [32]. One neuron's output is frequently utilized as the input for other neurons in the network, creating a complex web of relationships. The strength of the connections between neurons, known as weights, determines the importance of each input in the final output of the neuron. The weights are adjustable, allowing the neural network to learn from experience and improve its accuracy over time. There are several types of neural networks, including feedforward networks, recurrent networks, and convolutional networks.

In this research, three different neural network models were constructed with varying numbers of hidden layers. The neural network with hidden layers 1, 2, and 3 was implemented with different epoch values (200, 400, 800), and the outcomes were compared. The activation function in the hidden layer of an Artificial Neural Network (ANN) processes the weighted sum of input. Two activation functions, namely sigmoid and RELU, were employed in the study. The neural network models were created using Keras and Tensor Flow libraries, and the Sequential class from the Keras library was utilized. The 'DMtype' attribute served as the target variable. An optimizer is essential in ANN to minimize the output error during the backpropagation method. In this work, Adam optimization which is a stochastic gradient descent (SGD) method that is based on adaptive estimation of first-order and second-order moments was used as the optimizer. Learning rate is a parameter that regulates the weight adjustment with respect to the loss gradient in an optimization algorithm. To find an effective learning rate, different values were experimented with. The train-test-split function from the sci-kit-learn library was employed to carry out the train/test splitting process.

3. RESULTS AND DISCUSSION

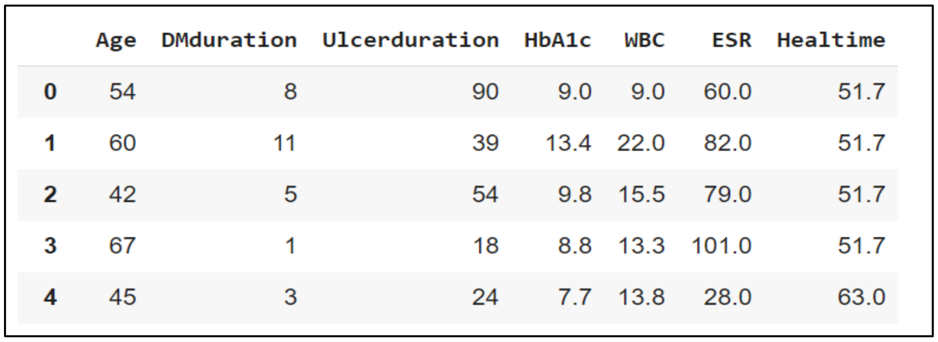
3.1 Data Preprocessing

Using Excel and SPSS for data preprocessing, missing values were observed in the datasets as shown in Table 2 below. White Blood Cell (WBC), Erythrocyte Sedimentation Rate (ESR), Diabetes Mellitus duration (DMduration), Hemoglobin (HbA1c), and Heal time had missing values which were replaced with the corresponding mean value. A preview of the features in the dataset can be seen in table 3 below.

Table 2: Number of Missing Values in the Features

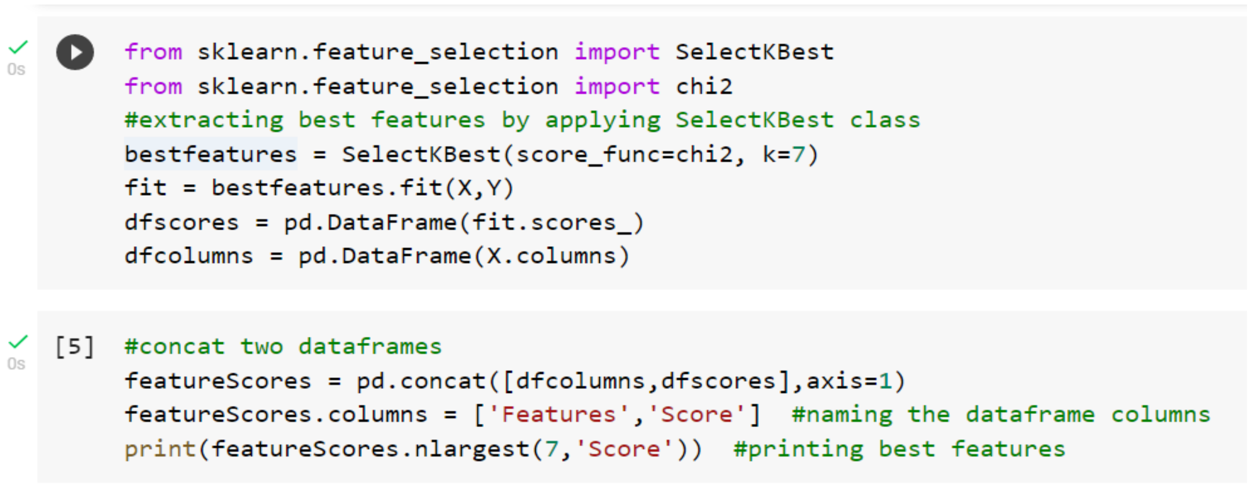
Features	No of Missing Values
Age	0
Ulcerduration	0
WBC	29
ESR	59
Dmtype	0
DMduration	36
HbA1c	40
Healtime	241

Table 3: Preview of Features in the Dataset



	Age	DMduration	Ulcerduration	HbA1c	WBC	ESR	Healtime
0	54	8	90	9.0	9.0	60.0	51.7
1	60	11	39	13.4	22.0	82.0	51.7
2	42	5	54	9.8	15.5	79.0	51.7
3	67	1	18	8.8	13.3	101.0	51.7
4	45	3	24	7.7	13.8	28.0	63.0

Filter method was used for the features selection in this analysis, statistical tests were conducted to determine which attributes have the strongest association with the output variable. The SelectKBest class in the scikit-learn library was utilized along with the chi-squared (χ^2) statistical test for non-negative features, to select a specific number of features.



```
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
#extracting best features by applying SelectKBest class
bestfeatures = SelectKBest(score_func=chi2, k=7)
fit = bestfeatures.fit(X,Y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(X.columns)
```

```
[5] #concat two dataframes
featureScores = pd.concat([dfcolumns,dfscores],axis=1)
featureScores.columns = ['Features','Score']  #naming the dataframe columns
print(featureScores.nlargest(7,'Score'))  #printing best features
```

Figure 2: Feature selection analysis script

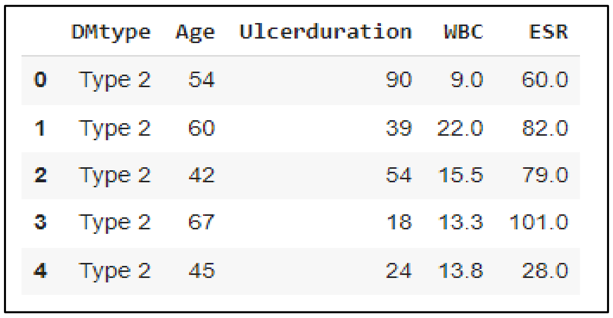
From Table 4 below, the top four (4) features: Age, Ulcer duration, ESR and WBC were selected since they have the highest scores after running the features selection script. Therefore, the analysis was done on the diabetes dataset with 4 features each. “Diabetes mellitus types” is the feature to be predicted and Age, Ulcer duration, Erythrocyte sedimentation rate (ESR), and white blood cell count (WBC) are the independent variables.

Table 4: Scores from Features Selection Analysis

Features	Score
Age	172.90
Ulcerduration	76.41
ESR	26.27
WBC	8.25
Healtime	3.28
DMduration	1.99
HbA1c	0.01

Table 5 below displays a preview of the dataset after it has been thoroughly cleaned and the best features have been chosen for the purpose of this analysis. Table 6 offers a comprehensive description of all features that were used for the prediction of diabetes types.

Table 5: Preview of the Dataset After Features Selection



	DMtype	Age	Ulcerduration	WBC	ESR
0	Type 2	54	90	9.0	60.0
1	Type 2	60	39	22.0	82.0
2	Type 2	42	54	15.5	79.0
3	Type 2	67	18	13.3	101.0
4	Type 2	45	24	13.8	28.0

Table 6: Description of Dataset Used for Prediction of Diabetes Types

Features	Description	Type	Average
Age	Age of the patient (Years)	Numeric	56
Ulcerduration	Duration of ulcer (Days)	Numeric	45
WBC	White blood cell count	Numeric	12.17
ESR	Erythrocyte sedimentation rate	Numeric	63.31
Dmtype	Diabetes mellitus type (Type 1, Type 2)	Nominal	-

3.2 Data Augmentation

As seen in Figure 4 below, out of 336 patients, only 13 have type 1 diabetes and 323 have type 2 diabetes, meaning it is an imbalanced dataset that can lead to overfitting of the models.

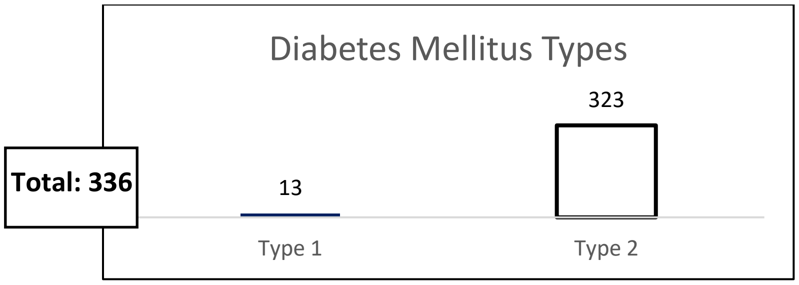
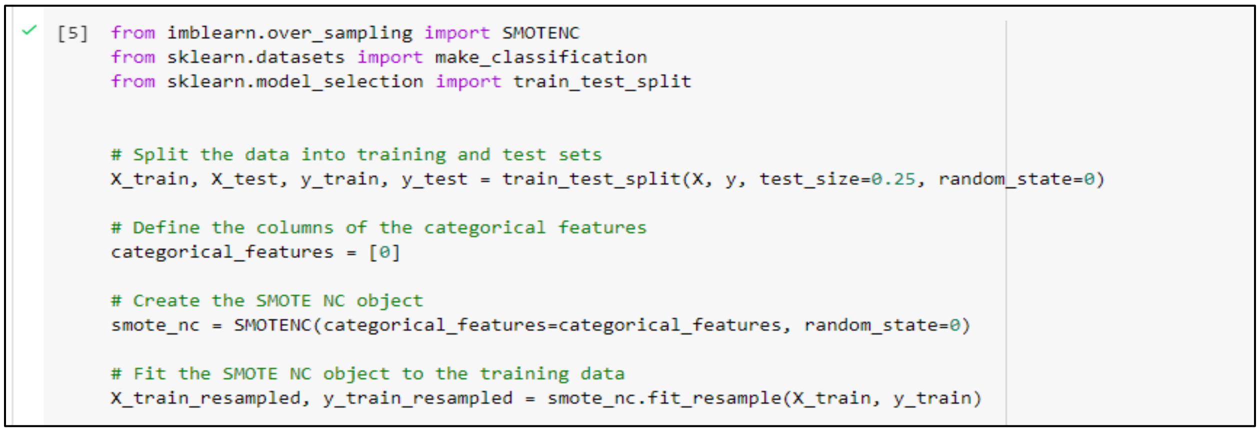


Figure 3: Distribution of Classes in the Response Variable

To address this issue, Synthetic Minority Over-sampling Technique Neighborhood Cleaning (SMOTE NC) was applied which equalize the classes thereby increasing the size of the dataset. This technique is implemented using the script below.



```
[5] from imblearn.over_sampling import SMOTENC
    from sklearn.datasets import make_classification
    from sklearn.model_selection import train_test_split


    # Split the data into training and test sets
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)

    # Define the columns of the categorical features
    categorical_features = [0]

    # Create the SMOTE NC object
    smote_nc = SMOTENC(categorical_features=categorical_features, random_state=0)

    # Fit the SMOTE NC object to the training data
    X_train_resampled, y_train_resampled = smote_nc.fit_resample(X_train, y_train)
```

Figure 4: Implementation of SMOTE NC Technique

The synthetic minority oversampling technique has oversampled the type 1 diabetes class by increasing the sample size and making the class of type 1 and type 2 diabetes equal. As shown in Figure 5 below, this resulted in 486 samples, with an equal number of patients, 243 each, diagnosed with type 1 and type 2 diabetes.

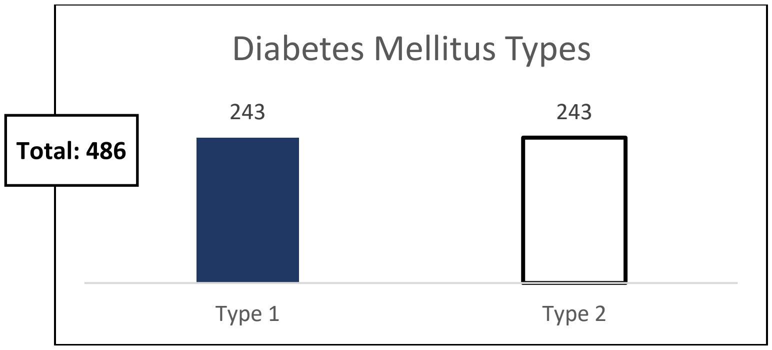


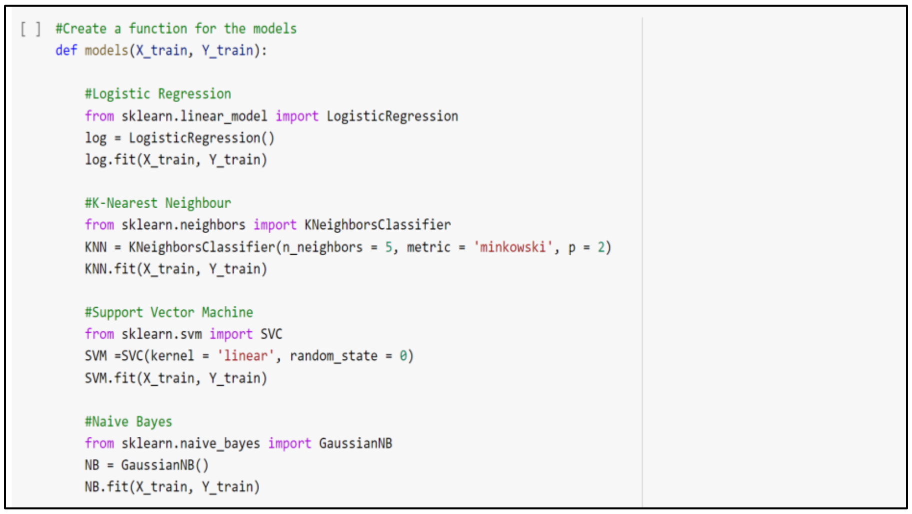
Figure 5: Distribution of Classes in the Response Variable after Data Augmentation

3.3 Dataset Train and Test

After data preprocessing, the dataset becomes ready to train (75%) and (25%) test using our proposed classification algorithms. This means that 75% of the dataset will be used in training the classification algorithm and the remaining 25% will be used in testing the classification algorithms. In the train/split method, the dataset was randomly split into the training and testing set and the different classification algorithms fitted to the training set. See figure 4 above for the train/test split script.

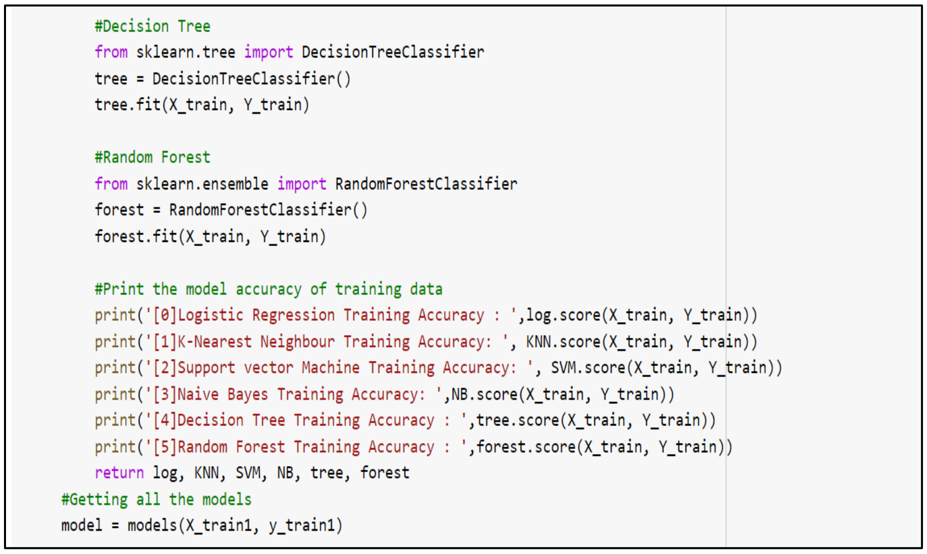
3.4 Applying the Algorithms

After splitting the data set, the training set were used to train the classification algorithms. Figure 6 and 7 below shows the script that was used in training the classification algorithms.



```
[ ] #Create a function for the models
    def models(X_train, Y_train):

        #Logistic Regression
        from sklearn.linear_model import LogisticRegression
        log = LogisticRegression()
        log.fit(X_train, Y_train)

        #K-Nearest Neighbour
        from sklearn.neighbors import KNeighborsClassifier
        KNN = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p = 2)
        KNN.fit(X_train, Y_train)

        #Support Vector Machine
        from sklearn.svm import SVC
        SVM =SVC(kernel = 'linear', random_state = 0)
        SVM.fit(X_train, Y_train)

        #Naive Bayes
        from sklearn.naive_bayes import GaussianNB
        NB = GaussianNB()
        NB.fit(X_train, Y_train)
```

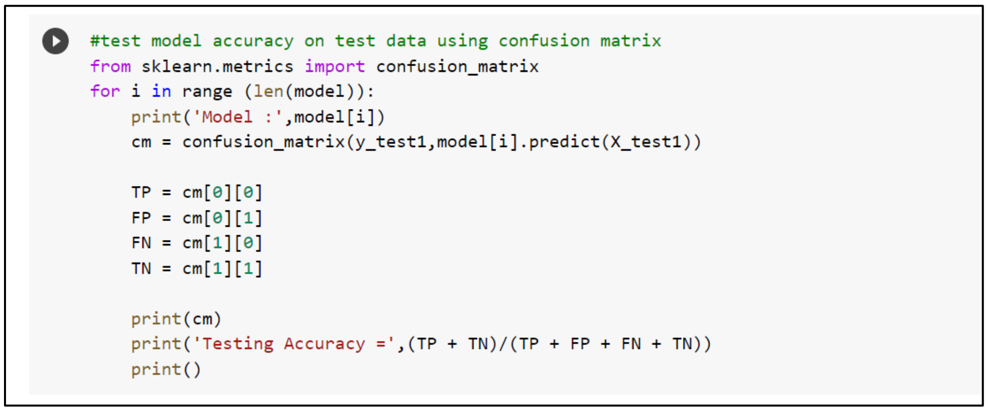
Figure 6: Training of Data on the Algorithms (a)



```
        #Decision Tree
        from sklearn.tree import DecisionTreeClassifier
        tree = DecisionTreeClassifier()
        tree.fit(X_train, Y_train)

        #Random Forest
        from sklearn.ensemble import RandomForestClassifier
        forest = RandomForestClassifier()
        forest.fit(X_train, Y_train)

        #Print the model accuracy of training data
        print('[0]Logistic Regression Training Accuracy : ',log.score(X_train, Y_train))
        print('[1]K-Nearest Neighbour Training Accuracy: ', KNN.score(X_train, Y_train))
        print('[2]Support vector Machine Training Accuracy: ', SVM.score(X_train, Y_train))
        print('[3]Naive Bayes Training Accuracy: ',NB.score(X_train, Y_train))
        print('[4]Decision Tree Training Accuracy : ',tree.score(X_train, Y_train))
        print('[5]Random Forest Training Accuracy : ',forest.score(X_train, Y_train))
        return log, KNN, SVM, NB, tree, forest
    #Getting all the models
    model = models(X_train1, y_train1)
```

Figure 7: Training of Data on the Algorithms (b)

Once the data has been trained using all six classification algorithms, the accuracy of the classification algorithms were then evaluated on the test data.



```
#test model accuracy on test data using confusion matrix
from sklearn.metrics import confusion_matrix
for i in range (len(model)):
    print('Model :',model[i])
    cm = confusion_matrix(y_test1,model[i].predict(X_test1))

    TP = cm[0][0]
    FP = cm[0][1]
    FN = cm[1][0]
    TN = cm[1][1]

    print(cm)
    print('Testing Accuracy =',(TP + TN)/(TP + FP + FN + TN))
    print()
```

Figure 8: Computation of Confusion Matrix

Here, FP= False Positive, FN= False Negative, TN=True Negative, TP=True Positive
The confusion matrices of logistics regression (LR), k-nearest neighbor (KNN), support vector machine (SVM), naïve Bayes (NB), decision tree (DT), and random forest (RF) classifier for Train/Test splitting is shown in Table 7. The performance measure value of all the classification algorithms used on the dataset is shown in Table 8.

Table 7: Confusion Matrix for the Classification Algorithms

	LR		KNN		SVM		NB		DT		RF	
	Type 1	Type 2	Type 1	Type 2	Type 1	Type 2	Type 1	Type 2	Type 1	Type 2	Type 1	Type 2
Type 1	60	3	62	0	61	1	62	0	61	1	62	0
Type 2	3	57	1	59	3	57	2	58	1	59	1	59

From the confusion matrices above, our test set has 122 observations with 62 patients that has type 1 diabetes and 60 patients with type 2 diabetes. The confusion matrices show how well each model can predict the diabetes type that the patient has.
In Table 8 below, the performance of all classification algorithms are indicated showing the strengths of each algorithm. The precision, recall, f-score, and accuracy of all classification algorithms exceeds 95%. Among these algorithms, the k-nearest neighbor (KNN) and random forest (RF) algorithms exhibit the highest accuracy at 99%.

Table 8: Performance of the Classification Algorithms

Models	Precision	Recall	F-score	Accuracy
LR	0.95	0.97	0.96	0.96
KNN	0.98	1	0.99	0.99
SVM	0.95	0.98	0.97	0.97
NB	0.97	1	0.98	0.98
DT	0.98	0.98	0.98	0.98
RF	0.98	1	0.99	0.99

3.5 Artificial Neural Network (ANN) Results

In Table 9 below, the effect of changing the learning rate to 0.1, 0.01, and 0.005 in a neural network with a single hidden layer and 200 epochs is shown. The results indicate that a learning rate of 0.01 leads to improved accuracy. Therefore, this value was utilized in each case.

Table 9: Impact of Learning Rate on Accuracy Measurement

Learning Rate	Accuracy
0.1	0.9672
0.01	0.9754
0.005	0.9672

Table 10 below displays the effect of changing the number of epochs in neural networks with one, two, and three hidden layers with a fixed learning rate of 0.01 since it was identified to lead to improved accuracy (see table 9). 200, 400, and 800 epochs were used for each of the hidden layers.

Table 10: Impact on the Accuracy at Learning Rate of 0.01 with Hidden Layer Changes

Hidden Layer	Epochs	Training Accuracy	Testing Accuracy
1	200	97.53%	96.72%
	400	96.43%	97.54%
	800	98.35%	97.54%
2	200	98.35%	95.9%
	400	95.88%	95.9%
	800	98.63%	98.36%
3	200	97.53%	95.08%
	400	99.18%	95.08%
	800	98.08%	96.72%

The best accuracy of 98.36% was achieved using a network with two hidden layers and 800 epochs. The corresponding ROC curve, displayed in Figure 9, was also generated using this configuration.

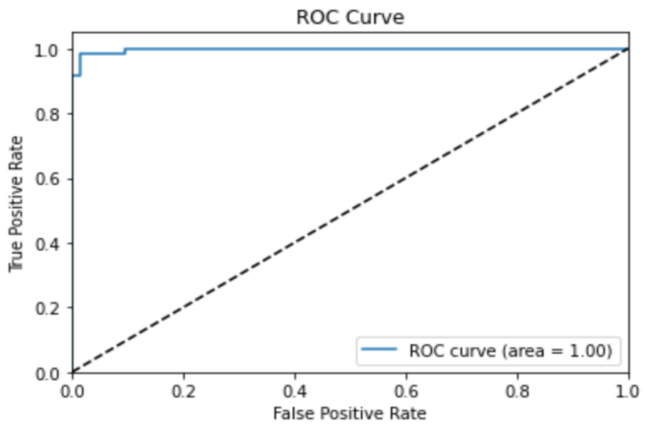


Figure 9: ROC Curve for 2 Hidden Layers NN with 800 Epochs

As shown in Figure 9, the area under the curve (AUC) is close to 1, indicating that the neural network with two hidden layers and 800 epochs provides an almost accurate prediction of diabetes type.

4. CONCLUSION

After data preprocessing was done and feature reduction method dropped three features, four input features (Age, Ulcer duration, ESR, and WBC) and one output feature (outcome) were remaining in the dataset. Six different classification algorithms were applied on the dataset to predict diabetes types and evaluated the performance on various measures. All classification algorithms show good results based on some parameters like accuracy, precision, recall, and F-measure. All classification algorithms provided an accuracy greater than 95%. However, Random Forest and K-Nearest Neighbor provided 99% accuracy for the train/test split. Also, using the 1, 2, and 3 hidden layers in the neural network model varying the epochs 200, 400, and 800. Hidden layer 2 with 800 epochs provided 98.36% accuracy. Among all the proposed algorithms, the Random Forest and K-Nearest Neighbor is considered the most efficient and promising for analyzing diabetes types with an accuracy rate of approximately 99%.
In conclusion, this research highlights the importance of using machine learning algorithms in diabetes classification and prediction. The findings suggest that the Random Forest and the k-nearest neighbor algorithms are more accurate in predicting diabetes types than the other algorithms. However, further research can identify the optimal algorithm and method for diabetes classification and prediction.

REFERENCES

[1] Y. Heianza, N. Kato, M. E. Cooper, L. Groop, E. Ferrannini, R. A. DeFronzo, K. Yano and T. Kadowaki (2020). Challenges and Opportunities in the Classification and Treatment of Diabetes. *The Lancet Diabetes & Endocrinology*, **8**(6): 474-486.

[2] WHO (2022), Newsroom, 16 September, 2022. Retrieved from: https://www.who.int/news-room/fact-sheets/detail/diabetes
[3] N. Singh, R. Kesharwani, A. Tiwari & D. Patel (2016). A review on diabetes mellitus. *The Pharma Innovation*. **5**. 36-40.
[4] J. Lindström & J. Tuomilehto (2003). The diabetes risk score: A practical tool to predict type 2 diabetes risk. *Diabetes Care*, **26**(3): 725-731.
[5] J. K. Jobeda, Y. F. Simon (2021) A Comparison of Machine Learning Algorithms for Diabetes Prediction, *ICT Express*, **7**(4): 432-439.
[6] K. VijiyaKumar, B. Lavanya, I. Nirmala & S. S. Caroline (2019). Random Forest Algorithm for the Prediction of Diabetes. *In 2019 IEEE International Conference on System, Computation, Automation and Networking* (ICSCAN) (pp.1-5). IEEE.
[7] S. Kumar, B. Bhushan, D. Singh & D. Choubey (2020). Classification of Diabetes using Deep Learning. *In 2020 6th International Conference on Computing and Sustainable Societies (ICCSS) (pp. 651-655). IEEE.* https://doi.org/10.1109/ICCSP48568.2020.9182293
[8] N. V. Chawla, K. W. Bowyer, L. O. Hall & W. P. Kegelmeyer (2002). Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning. *In Advances in neural information processing systems* (pp. 307-314).
[9] P. S. Kohli & S. Arora (2018). Application of Machine Learning in Disease Prediction. *In 2018 4th International Conference on Computing Communication and Automation (ICCCA) (pp. 1-4). Greater Noida, India: IEEE.* https://doi.org/10.1109/CCAA.2018.8777449.
[10] Y. K. Sajratul, R. Monibor & H. Kamrul (2018). Important Feature Selection & Accuracy Comparisons of Different Machine Learning Models for Early Diabetes Detection, In *2018 International Conference on Innovation in Engineering and Technology (ICIET).*
[11] A. Adel & S. Abdulkadir (2019). Performance Comparison of Machine Learning Techniques on Diabetes Disease Detection, *1st International Informatics and Software Engineering Conference* (UBMYK).
[12] G. A. Pethunachiyar (2020). Classification of Diabetes Patients Using Kernel Based Support Vector Machines, *In 2020 International Conference on Computer Communication and Informatics* (ICCCI -2020), Coimbatore, INDIA (pp. 168-171).
[13] K. D. Samrat, H. Ashraf & R. Mahbubur (2018). Implementation of a Web Application to Predict Diabetes Disease: An Approach Using Machine Learning Algorithm, *In 2018 21st International Conference of Computer and Information Technology (ICCIT)*, pp. 21-23.
[14] M. Shanthi, R. Marimuthu, S. N. Shivapriya & R. Navaneethakrishnan (2019). Diagnosis of Diabetes using an Extreme Learning Machine Algorithm based Model. *In 2019 IEEE 10th International Conference on Awareness Science and Technology* (iCAST) (pp. 1-5). IEEE.
[15] M. J. Uddin, M. M. Ahamad, M. N. Hoque, M. A. A. Walid, S. Aktar, N. Alotaibi, S. A. Alyami, M. A. Kabir, M. A. Moni. (2023). Comparison of Machine Learning Techniques for the Detection of Type-2 Diabetes Mellitus: Experiences from Bangladesh. *Information*. 2023; 14(7):376. https://doi.org/10.3390/info14070376
[16] R. Patil & S. Tamane (2018). A Comparative Analysis on the Evaluation of Classification Algorithms in the Prediction of Diabetes. *International Journal of Electrical and Computer Engineering*. 8. 3966-3975. 10.11591/ijece.v8i5.pp3966-3975.

[17] S. L. Cichosz, , C. Bender, & O. Hejlesen (2024). A Comparative Analysis of Machine Learning Models for the Detection of Undiagnosed Diabetes Patients. *Diabetology,* 5(1), 1–11. https://doi.org/10.3390/diabetology5010001

[18] U. M. Butt, S. Letchmunan, M. Ali, F. H. Hassan, A. Baqir & H. H. R. Sherazi (2021). Machine Learning Based Diabetes Classification and Prediction for Healthcare Applications. *Journal of Healthcare Engineering*, *2021*, 9930985. https://doi.org/10.1155/2021/9930985

[19] V. Rawat, S. Joshi, S. Gupta, D. P. Singh, N. Singh (2022). Machine learning algorithms for early diagnosis of diabetes mellitus: A comparative study, Materials Today: Proceedings, Volume 56, Part 1,2022, Pages 502-506, ISSN 2214-7853, https://www.sciencedirect.com/science/article/pii/S2214785322007507

[20] Y. Yang, J. O. Pedersen & Q. Diao (2016). A comparative study of feature selection methods for text classification. *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, (pp.153–162).

[21] M. Mukherjee & M. Khushi (2021). SMOTE-ENC: A Novel SMOTE-Based Method to Generate Synthetic Data for Nominal and Continuous Features. *Applied System Innovation*, **4**(1): 18 – 27

[22] J. S. Cramer (2002). The Origins of Logistic Regression. Tinbergen Institute, *Tinbergen Institute Discussion Papers*. 10.2139/ssrn.360300.

[23] C. Cortes & V. Vapnik (1995). Support-Vector Networks. *Machine Learning*, **20**(3): 273-297.

[24] G. Guo, H. Wang, D. Bell & Y. Bi (2004). KNN Model-Based Approach in Classification. *Proceedings of the International Conference on Natural Computation,* 2004, (pp. 163-167).

[25] J. R. Quinlan (1986). Induction of decision trees. *Machine Learning*, **1**(1), 81-106.

[26] L. Breiman (2001). Random forests. *Machine Learning*, **45**(1), 5-32.

[27] A. M. Kibriya, E. Frank, B. Pfahringer & G. Holmes (2004). Multinomial Naive Bayes for Text categorization Revisited. *In Proceedings of the 22nd International Conference on Machine Learning* (pp. 633-640).

[28] R. Ogundeji, J. Akinyemi and M. Tijani (2022). Na¨Ive Bayes Algorithm for Document Classification. *Annals of Mathematics and Computer Science* 7(2022): 54 – 65.

[29] N. Djuric, J. Zhou & J. R. Finkelstein (2010). Bayesian Network Classifiers for Categorical Data. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, **40**(3): 881-890.

[30] N. Japkowicz & M. Shah (2011). *Evaluating Learning Algorithms: A Classification Perspective.*Cambridge: Cambridge University Press, England.

[31] G. V. Klepac & T. Šmuc, (2019). Evaluation of Classifier Performance with Confusion Matrix and Related Measures. *Biochemia Medica*, **29**(2): 1-15.

[32] E. Çoban (2016). Neural Networks and Their Applications. Retrieved from: https://www.researchgate.net/publication/294085530.

ROTIMI OGUNDEJI*

DEPARTMENT OF STATISTICS, FACULTY OF SCIENCE, UNIVERSITY OF LAGOS, AKOKA, NIGERIA.
E-mail address: rogundeji@unilag.edu.ng

HOPE ADEGOKE
DEPARTMENT OF STATISTICS, FACULTY OF SCIENCE, UNIVERSITY OF LAGOS, AKOKA, NIGERIA.
E-mail address: hopeadegoke13@gmail.com