CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

Testosterone is the most important sex hormone among males and significantly impacts men's physical and psychological well-being. Patients with Testosterone Deficiency Syndrome (TDS) may experience hypogonadism, a condition defined by low serum testosterone levels combined with clinical symptoms. This condition is associated with various comorbidities, such as metabolic syndrome, cardiovascular diseases, erectile dysfunction, atherosclerosis, respiratory problems, depression, and other complications that reduce overall health indicators. Secondary testosterone deficiency is a medical condition where the body fails to produce enough testosterone due to a problem in the pituitary gland or hypothalamus. The project involved studying multiple research papers and implementing a proposed machine learning model. The dataset was imbalanced, so three types of sampling techniques were implemented - under-sampling, over-sampling, and hybrid-sampling. In over-sampling, random over-sampling and renn techniques were used, while in under-sampling, random under-sampling and smote were implemented. Four different combinations of sampling were used in hybrid sampling. A total of eight datasets were created, each with an equal count of affected and unaffected rows.

Multiple classifiers were implemented using the eight datasets, along with stratified k-fold cross-validation. Each model was trained on the respective dataset, and tested on the testing fold to predict the accuracy. Overall, the project involved implementing various sampling techniques and classifiers to handle imbalanced data, along with cross-validation and hyperparameter tuning to improve model performance. The results showed that different datasets produced different accuracy results, and the best accuracy was achieved using the XGBoost classifier with hyperparameter tuning.

1.2 OBJECTIVE

The objective of this project could be to develop a machine learning model that can accurately classify data in an imbalanced dataset, where the number of samples in one class is significantly higher than the other into three levels of abnormality of testosterone hormone (low, medium, high).

1.3 MOTIVATION

The motivation behind this project is to develop a machine learning model that can predict the level of testosterone in a person based on certain inputs, such as age, diabetes, TG (Triglycerides in mg/dl), HT (Hypertension), HDL (High-density lipoprotein in mg/dl), and AC (Abdominal Circumference in cm). Testosterone is an important hormone that affects many aspects of a person's health, including muscle mass, bone density, and sexual function. Predicting the level of testosterone can help doctors and healthcare professionals diagnose and treat conditions related to testosterone deficiency or excess.

By developing an accurate and reliable machine learning model, this project aims to provide a valuable tool for healthcare professionals to assess a person's testosterone level and make informed decisions about their treatment. Accurately predicting secondary testosterone deficiency is important for medical analysis and treatment. However, the existing system for predicting this deficiency is not accurate enough for medical purposes. Accurately predicting secondary testosterone deficiency can lead to better medical outcomes for patients, allowing for earlier detection and treatment of the condition.

1.4 ORGANIZATION OF CHAPTERS

Chapter I: Gives a general introduction of the project, the existing systems, objective and motivation of the study.

Chapter II: Literature survey that is done to support the project and various machine learning algorithms has been discussed.

Chapter III: Explains the existing system architecture, modules and description for the individual modules involved in the work.

Chapter IV: Explains the proposed system, it consists of the framework design, modules and its descriptions.

Chapter V: The implementations and output of the existing system and proposed system screenshots has been displayed.

Chapter VI: Describes the brief summary of the work done and presents some future enhancement work that can be carried over.

CHAPTER 2

LITERATURE REVIEW

2.1 TECHNIQUES

2.1.1 SAMPLING

Sampling is a method that allows us to get information about the population based on the statistics from a subset of the population (sample), without having to investigate every individual"

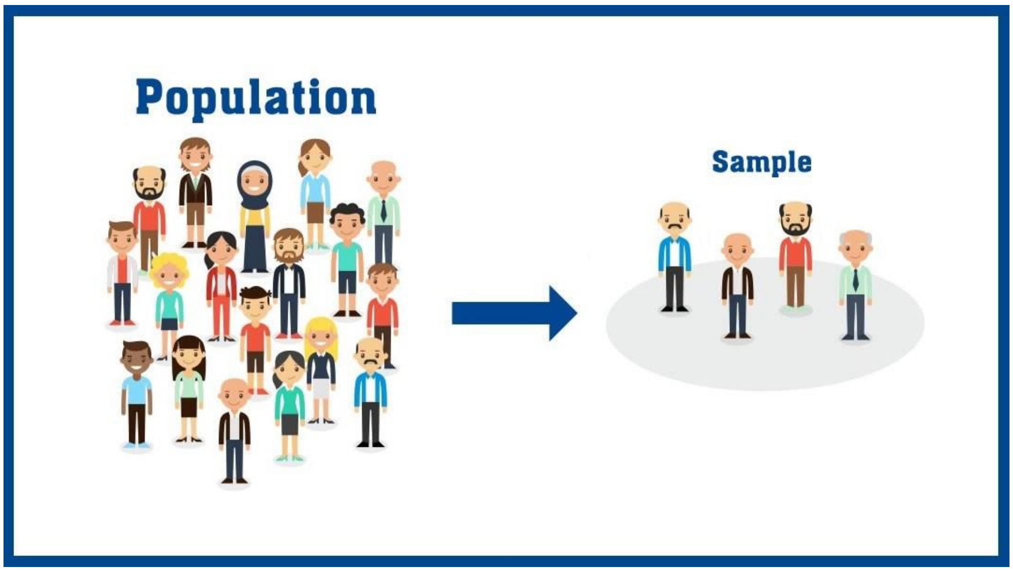


Fig. 2.1: Sampling

Image Source: https://shorturl.at/ghsM5

In the above Fig. 2.1: When you conduct research about a group of people, it's rarely possible to collect data from every person in that group. Instead, you select a sample. The sample is the group of individuals who will actually participate in the research.

2.1.2 CLASSIFICATION ALGORITHM

The Classification algorithm is a Supervised Learning technique that is used to identify the category of new observations on the basis of training data. In Classification, a program learns from the given dataset or observations and then classifies new observations into a number of classes or groups. Such as, **Yes or No, 0 or 1, Spam or Not Spam, cat or dog,** etc.

Classes can be called as targets/labels or categories. Unlike regression, the output variable of Classification is a category, not a value, such as "Green or Blue", "fruit or animal", etc. Since the Classification algorithm is a Supervised learning technique, hence it takes labelled input data, which means it contains input with the corresponding output. In the classification algorithm, a discrete output function(y) is mapped to input variable(x). The best example of an ML classification algorithm is Email Spam Detector. The main goal of the Classification algorithm is to identify the category of a given dataset, and these algorithms are mainly used to predict the output for the categorical data. Classification algorithms can be better understood using the below diagram. In the below diagram, there are two classes, class A and Class B. These classes have features that are similar to each other and dissimilar to other classes. The above example Fig. 2.2 classifies whether it is CLASS A or CLASS B.

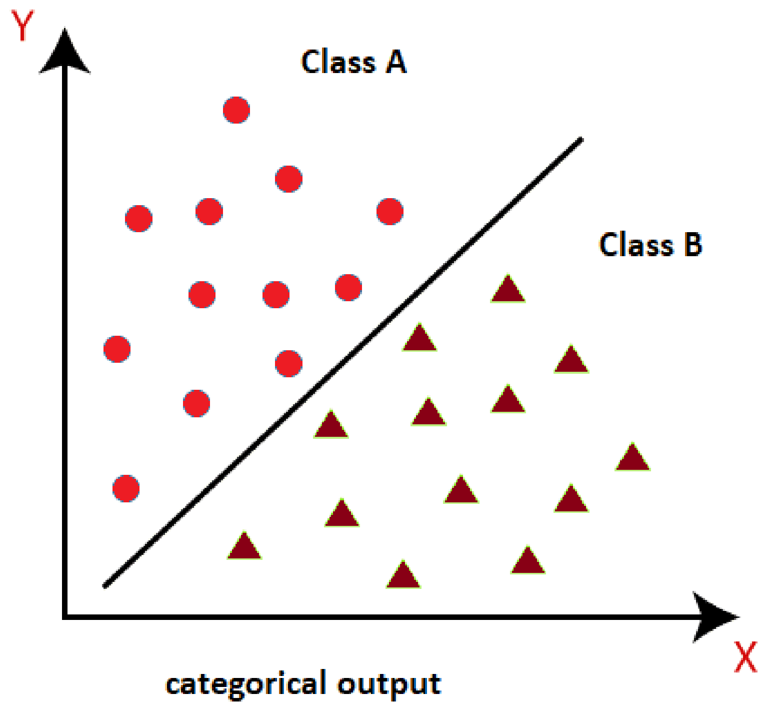


Fig. 2.2: Classifier

Image Source: https://rb.gy/ia8p7

2.2 SURVEY OF THE RELATED WORK

S.No:	Title	Journal Name	Description	Limitations
1	Prediction of secondary testosterone deficiency using machine learning: A comparative analysis of ensemble and base classifiers, probability calibration, and sampling strategies in a slightly imbalanced database. [1]	Informatics in Medicine Unlocked/ 2021	This paper aims to offer a coherent comparative analysis of machine learning methods that can predict testosterone deficiency without having patients undergo costly medical tests	Threshold levels are very high. Performance levels are lower when compared to the normal clinical analysis.
2	Predicting Diabetes Mellitus With Machine Learning Techniques. [2]	Frontiers in Genetics/ 2018	The major objective of this article is to predict the occurrence of diabetes mellitus in men using the testosterone data. Machine learning algorithms decision tree, random forest and neural networks are used.	High Computational Time. Possibility of over thinking.
3	Applying machine learning techniques to the identification of late-onset hypogonadism in elderly men. [3]	Springerplus/ 2016	This Paper uses two well-known classification techniques, the decision tree (DT) and logistic regression to construct LOH prediction models on the basis of the aforementioned features.	High Dimensionality. Low accuracy.
4	Heart Disease Identification Method Using Machine Learning Classification in E-Healthcare [4]	IEEE/ 2020	The Author proposes diagnosis system (FCMIM-SVM) to identify Heart Disease.	The proposed feature selection algorithm (FCMIM) achieves 92.37% accuracy and still need for improvement.

5	Identification of Cardiovascular Diseases Using Machine Learning [5]	IEEE/ 2019	The Author proposes MachineLearning algorithm in identifying a heart disease of a patient and guide a doctor to better diagnose whether a person has cardiovascular disease or not.	Support Vector Machine algorithm achieves 86.8% accuracy and still need for improvement.
6	Identifying The Predictive Capability of Machine Learning Classifiers For Designing Heart Disease Detection System [6]	IEEE/ 2019	The Author proposes various Machine Learning classifiers and deep neural network classifiers for designing a medical expert system.	The Ensemble learning techniques have a great effect on SVM classification accuracy and it improved the classification accuracy of 92.30%.

2.3 SURVEY SUMMARY

In recent years there are much research go for to predict the accuracy level of imbalanced data. In this literature survey, the techniques used and implemented in topic Sampling, classification, and Vector support Machines are discussed. And they discussed many things in their research paper with their own technique that they implement their paper to over the previous paper's existing system.In the majority of classes in the lecturer survey which is which they conclude about 70 to 80 percent. Many research papers are recognized by the limitation of clinical analysis.

CHAPTER 3

EXISTING SYSTEM

3.1 OVERVIEW

In this research, they aim to offer a coherent comparative analysis of machine learning methods that can predict testosterone deficiency without having patients undergo costly medical tests. In this analysis, we used ten base classifiers (optimized with grid search stratified K-fold cross-validation); three ensemble methods; and eight sampling strategies to analyze a total of 3397 patients. The analysis was based on six features (age; abdominal circumference; triglycerides; high-density lipoprotein; diabetes; and hypertension), all of which were obtained by low-cost exams.

3.2 ARCHITECTURE

In the architecture, The methodology is divided into the following steps (Fig. 3.1): base classifiers; sampling strategies; stratified K-fold cross-validation; accuracy analysis. The new dataset is generated using 8 different sampling strategies and generated dataset are used trained using different classifiers and tested using stratified k fold cross validation. Then it predicts average accuracy [1].

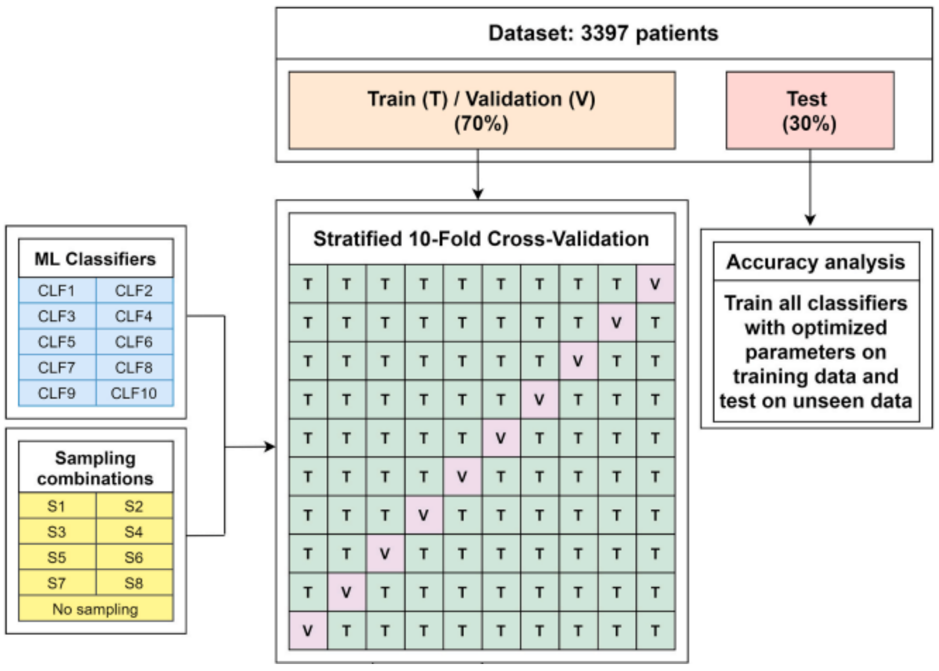

Dataset: 3397 patients
Train (T) / Validation (V) (70%)
Test (30%)
ML Classifiers
CLF1 CLF2 CLF3 CLF4 CLF5 CLF6 CLF7 CLF8 CLF9 CLF10
Sampling combinations
S1 S2 S3 S4 S5 S6 S7 S8
No sampling
Stratified 10-Fold Cross-Validation
T V
Accuracy analysis
Train all classifiers with optimized parameters on training data and test on unseen data

Fig. 3.1: Existing Architecture

3.3 MODULE DESCRIPTION

3.3.1 MODULE 1 - SAMPLING STRATEGIES

In machine learning, sampling techniques are used to balance the distribution of data in the training and testing datasets and to ensure that the model is not biased toward any particular class or group [1].

Here the used sampling techniques are:

- Random under-sampling (RUS),
- Repeated Edited Nearest Neighbours (RENN),
- Random Over-Sampling (ROS),
- Synthetic Minority Over-sampling Technique (SMOTE)

RANDOM UNDER SAMPLING (RUS)

Random under sampling is a simple under sampling technique in which instances are randomly selected and removed from the majority class until the number of instances in both classes is roughly the same. This technique can be effective when the majority class has a significantly larger number of instances than the minority class.

Random under sampling can be a fast and effective way to balance an imbalanced dataset. However, it can also result in the loss of important information and reduce the diversity of the data. As a result, it is important to carefully consider the implications of using random under sampling and evaluate its effectiveness in the context of the specific problem and dataset being analyzed [1].

REPEATED EDITED NEAREST NEIGHBOUR (RENN):

RENN (Repeated Edited Nearest Neighbor) sampling is a data preprocessing technique used in machine learning to balance imbalanced datasets. RENN is an extension of the Edited Nearest Neighbor (ENN) algorithm that iteratively removes noisy instances from the majority class until a balanced dataset is achieved. The RENN algorithm works in two steps. In the first step, the ENN algorithm is applied to remove noisy instances from the majority class. ENN removes an instance from the majority class if its class label differs from the class label of its k-nearest neighbors. The k parameter is a user-defined parameter that controls the level of sensitivity to noise. After applying the ENN algorithm, the majority class is reduced to a smaller subset of instances that are closest to the minority class instances. In the second step, the RENN algorithm repeatedly applies the ENN algorithm to the reduced majority class instances until no more instances can be removed. This process helps to further refine the majority class by removing any remaining noisy instances that were not detected in the first step. The result is a balanced dataset with an equal number of instances for each class.

RENN sampling can be useful when the imbalanced dataset has a high degree of overlap between the minority and majority classes. However, like all sampling techniques, it has its limitations and should be evaluated carefully in the context of the specific problem and dataset being analyzed [1].

RANDOM OVER SAMPLING

Random oversampling is a simple technique for balancing the class distribution in an imbalanced dataset by increasing the number of examples in the minority class. This technique involves randomly duplicating examples from the minority class to increase their representation in the dataset.

Random oversampling can be an effective technique for improving the performance of machine learning models on imbalanced datasets, especially for small datasets. However, this technique can also lead to overfitting and reduced generalization performance if not used appropriately. To avoid overfitting, it is important to use a validation set to evaluate the performance of the machine learning model during training and to select the best hyperparameters for the model [1].

SYNTHETIC MINORITY OVERSAMPLING TECHNIQUE

SMOTE (Synthetic Minority Over-sampling Technique) is a popular data augmentation technique used to address class imbalance in machine learning. The SMOTE algorithm works by randomly selecting a minority class sample and finding its k nearest neighbors in the feature space. The synthetic sample is then created by randomly selecting one of these k neighbors and generating a new sample that is a linear combination of the selected neighbor and the original minority sample. By repeating this process for multiple minority class samples, SMOTE generates a set of synthetic samples that can be added to the original dataset to create a new, balanced dataset. The size of the synthetic dataset can be controlled by adjusting the ratio of minority to majority class instances. SMOTE can be a useful technique for addressing class imbalance in a wide range of machine learning applications. However, it is important to note that SMOTE, like other oversampling techniques, can also introduce some noise or duplication into the data and potentially lead to overfitting. Therefore, it is important to carefully evaluate the effectiveness of SMOTE in the context of the specific problem and dataset being analyzed [1].

3.3.2 MODULE 2 – MODEL WITH CROSS VALIDATION

CLASSIFIER ALGORITHMS WITH CROSS VALIDATION

Classifier algorithms are a type of machine learning algorithm used to predict or classify a new data point into one of several categories based on the characteristics or features of the data. Some common classifier algorithms include: Stratified 10-fold cross-validation is a technique that is used to evaluate the performance of a classifier by dividing the data into 10 equally sized subsets, where each subset is used for testing the model once while the remaining 9 subsets are used for training the model [1].

- Multilayer perceptron
- Artificial neural network
- Random forest
- Extremely randomized trees
- Naïve bayes
- K Nearest Neighbours
- Logistic regression
- AdaBoost
- XgBoost

MULTILAYER PERCEPTRON

The multilayer perceptron (MLP) is a type of artificial neural network that is commonly used for classification and regression tasks in machine learning. It is composed of multiple layers of interconnected nodes (neurons) that process input data and produce an output. The MLP consists of an input layer, one or more hidden layers, and an output layer. Each layer contains one or more neurons, which are responsible for processing and transforming the input data. Each neuron in the hidden layers and output layer applies a weighted sum of the inputs followed by an activation function to produce an output value.During the training process, the MLP learns to adjust the weights of its connections to minimize the error between the predicted output and the actual output. This is typically achieved using an optimization algorithm such as gradient descent or its variants [1].

ARTIFICIAL NEURAL NETWORK

An artificial neural network (ANN) is a computational model that is inspired by the structure and function of biological neural networks, such as the human brain. It is a type of machine learning algorithm that is commonly used for pattern recognition, classification, and regression tasks. The basic building block of an ANN is a neuron, which receives inputs, processes them, and produces an output. A neuron takes inputs from other neurons or from the external environment, applies a mathematical function to the inputs, and produces an output that is transmitted to other neurons. An ANN consists of multiple layers of interconnected neurons, with each layer performing a specific function. The input layer receives the input data and passes it to the hidden layers, which perform computations and transform the input data into a form that is more suitable for the output layer. The output layer produces the final output of the network. During the training process, the ANN learns to adjust the weights of its connections to minimize the difference between the predicted output and the actual output. This is typically achieved using an optimization algorithm such as gradient descent or its variants [1].

RANDOM FOREST CLASSIFIER

Random Forest is a machine learning algorithm that is commonly used for classification and regression tasks. It is an ensemble learning method that combines multiple decision trees to produce a more robust and accurate model. The Random Forest algorithm works by creating a large number of decision trees and aggregating their outputs to make a final prediction. Each decision tree is trained on a random subset of the training data and a random subset of the input features. This randomness helps to reduce overfitting and increase the diversity of the individual trees. During the training process, each decision tree is constructed by recursively partitioning the input space into smaller regions, based on the values of the input features. At each node of the tree, the algorithm selects the best feature to split the data and the best split point based on a measure of the split's effectiveness, such as the Gini impurity or information gain. To make a prediction using the Random Forest algorithm, the input data is passed through each of the individual trees in the forest, and the outputs of the trees are combined using a majority voting scheme for classification or an averaging scheme for regression [1] [2].

EXTREMELY RANDOMIZED TREES

Extremely Randomized Trees (ExtraTrees) is an ensemble learning method that is based on the random forest algorithm. It was introduced to address some of the limitations of the standard random forest algorithm, such as its computational complexity and variance. Like the random forest algorithm, ExtraTrees builds an ensemble of decision trees based on random subsets of the input features and the training data. However, there are a few key differences between the two algorithms. First, instead of selecting the best-split point for each feature at each node of the tree, ExtraTrees randomly selects a split point for each feature and chooses the best among them. This makes the algorithm faster and less sensitive to noise in the data. Second, in ExtraTrees, each decision tree is built using a random subset of the training data without replacement. This increases the diversity of the trees and reduces the variance of the predictions. Finally, the number of features and the number of trees used in the ExtraTrees algorithm can be set to much higher values than in the random forest algorithm, which can lead to better performance in certain situations [1].

NAIVE BAYES CLASSIFIER

Naive Bayes Classifier is a probabilistic algorithm that is commonly used in machine learning for classification tasks. It is based on Bayes' theorem and makes some simplifying assumptions about the data to make computation easier and more efficient. The algorithm assumes that all features are independent of each other, which is where the "naive" part of the name comes from. This means that the probability of a certain feature occurring given the class is not dependent on any of the other features. This assumption is often not true in real-world data, but the algorithm can still be effective in practice. To use the Naive Bayes Classifier, the algorithm must first be trained on a dataset with known labels. It then uses this training data to calculate the probabilities of each feature given each class. When making a prediction on new, unseen data, the algorithm calculates the probability of each class given the observed features and chooses the class with the highest probability. The Naive Bayes Classifier is particularly useful for text classification tasks, such as spam filtering and sentiment analysis, where the independence assumption is often more accurate [1].

K NEAREST NEIGHBOUR

k-Nearest Neighbors (k-NN) is a popular supervised machine learning algorithm used for classification and regression tasks. It is a non-parametric method, meaning that it does not make any assumptions about the underlying distribution of the data. The k-NN algorithm works by comparing a new observation to the k-nearest observations in the training set based on some distance metric, typically Euclidean distance. The algorithm then assigns the class or predicts the value of the new observation based on the majority class or the average value of the k-nearest neighbors. The value of k is a hyperparameter that needs to be tuned to obtain the best performance on the validation set. A smaller value of k leads to more complex decision boundaries and can result in overfitting, while a larger value of k results in smoother decision boundaries and can lead to underfitting. The k-NN algorithm is easy to understand and implement, and it works well on small datasets with simple structures. However, it can be computationally expensive for large datasets, as the algorithm requires storing and searching the entire training set for each prediction. Additionally, the algorithm can be sensitive to irrelevant features or noisy data points [1].

LOGISTIC REGRESSION

Logistic regression is a popular machine learning algorithm used for binary classification tasks, such as predicting whether a customer will buy a product or not, or whether an email is spam or not. It is a linear model that uses a logistic function to model the probability of a binary outcome based on one or more input variables (also called features). The logistic regression algorithm works by minimizing a cost function, such as the binary cross-entropy loss, to find the optimal weights for the input variables. During training, the algorithm iteratively adjusts the weights using gradient descent, a first-order optimization algorithm that minimizes the cost function by moving in the direction of steepest descent. Logistic regression is a powerful and widely used algorithm that can handle linear and non-linear decision boundaries. It is also computationally efficient and easy to interpret, as the coefficients of the input variables can be directly interpreted as the impact of each variable on the predicted outcome [1] [3].

ADABOOST:

AdaBoost, short for Adaptive Boosting, is a popular ensemble learning algorithm used for classification tasks. It combines multiple weak classifiers into a strong classifier, which achieves higher accuracy than any individual classifier. The AdaBoost algorithm works by iteratively training a sequence of weak classifiers on the training data, and assigning weights to each training example based on whether it was correctly or incorrectly classified by the previous classifiers. The weights are then used to re-weight the training examples, so that the next classifier focuses more on the misclassified examples. During prediction, the final classifier is a weighted sum of the weak classifiers, where the weights are determined by their performance on the training data. The weights of the classifiers are adjusted so that the classifiers that perform better on the training data have more weight in the final ensemble. AdaBoost is a flexible and powerful algorithm that can be used with any base classifier, such as decision trees, logistic regression, and support vector machines. It is also robust to overfitting, as the re-weighting of the training examples focuses the algorithm on the most difficult examples [1].

XGBOOST

XGBoost, short for eXtreme Gradient Boosting, is a popular and powerful ensemble learning algorithm used for classification and regression tasks. It is an extension of the Gradient Boosting algorithm, which iteratively trains weak learners to create a strong learners. The XGBoost algorithm works by training decision trees on the training data and combining the results of these trees to make a final prediction. Unlike traditional decision trees, which partition the data into a set of rectangles in feature space, the decision trees used by XGBoost use gradients to determine the optimal split points. During training, XGBoost adds new decision trees to the ensemble and adjusts the weights of the training examples to focus on the misclassified examples. The algorithm also includes regularization to prevent overfitting, by penalizing complex trees and reducing their contribution to the final prediction. During prediction, XGBoost applies the ensemble of decision trees to the test data, and calculates a weighted sum of their predictions to make the final prediction [1].

3.3.3 LIMITATIONS IN EXISTING SYSTEM

- It produce less accuracy ranges from 70% - 80%. For medical analysis accuracy should be above 95%.
- Abnormality level cannot be identified in existing system.
- More time consuming for results.

CHAPTER 4

PROPOSED SYSTEM

4.1 OVERVIEW

We propose a classification model for detecting testosterone deficiency using XGBoost, a popular gradient boosting framework. The proposed system addresses the issue of class imbalance in the dataset using SMOTE, a data resampling technique. The system then performs hyperparameter tuning using randomized search with stratified 10-fold cross-validation to find the best set of hyperparameters for the XGBoost model. The best model is then used to make predictions on the original dataset, and the accuracy scores are added as a new column to the original dataset. The system also creates a new column 'abnormality_level' based on the accuracy score, categorizing the predicted results into low, medium, and high levels of abnormality provides a more nuanced and clinically relevant interpretation of the results, allowing healthcare providers to tailor treatment to individual patients' needs. Overall, the proposed system achieves accuracy of 96%.

4.2 ARCHITECTURE

In the proposed architecture, the methodologies are divided into following steps (Fig. 4.1): SMOTE sampling is done for balancing the dataset; XGB classifier is improved using hyperparameter tuning with random search to train the dataset; stratified k fold cross validation;

In the target column the data is either 0(No Testosterone Deficiency) or 1(Testosterone Deficiency). In the proposed model the testosterone deficiency is identified only for the rows with testosterone deficiency. So its filtered the dataset with only the rows has testosterone deficiency and identified the deficiency accuracy with abnormality levels like Low, Medium and High.

This helps to patient to know about the deficiency level and generates into new dataset with new columns deficiency level and abnormality level.

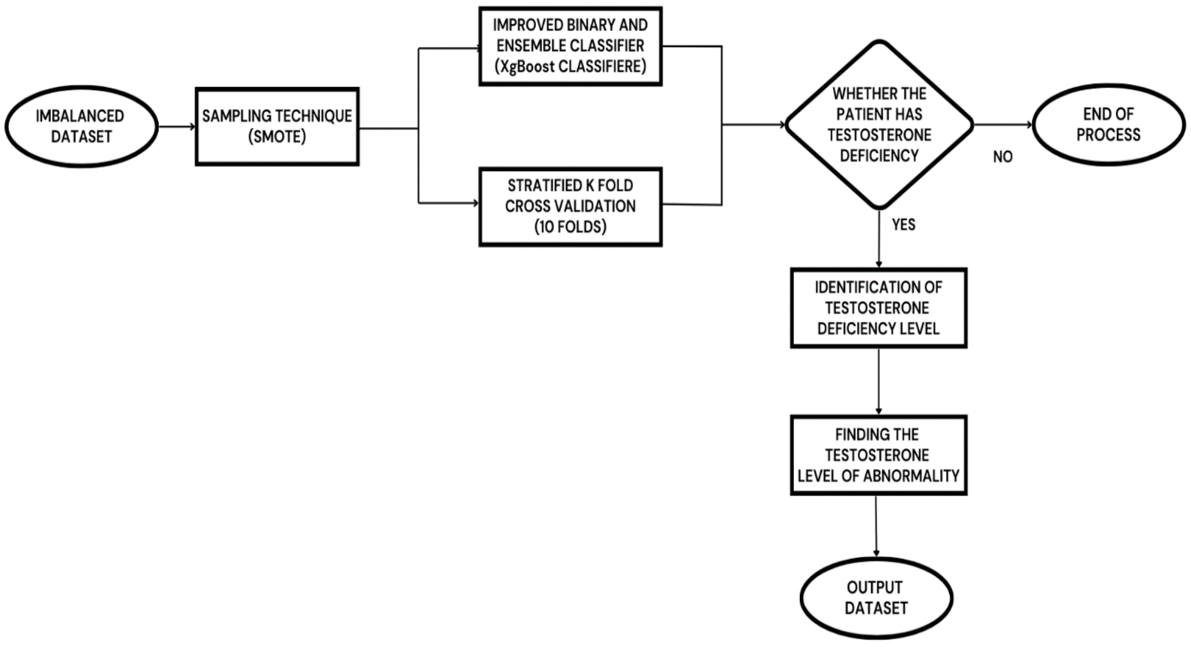

IMBALANCED DATASET
SAMPLING TECHNIQUE (SMOTE)
IMPROVED BINARY AND ENSEMBLE CLASSIFIER (XgBoost CLASSIFIERE)
STRATIFIED K FOLD CROSS VALIDATION (10 FOLDS)
WHETHER THE PATIENT HAS TESTOSTERONE DEFICIENCY
NO
END OF PROCESS
YES
IDENTIFICATION OF TESTOSTERONE DEFICIENCY LEVEL
FINDING THE TESTOSTERONE LEVEL OF ABNORMALITY
OUTPUT DATASET

Fig. 4.1: Architecture of Proposed system

4.3 MODULE DESCRIPTION

4.3.1 MODULE 1 – SAMPLING STRATEGIES

SYNTHETIC MINORITY OVER SAMPLING

Synthetic Minority Over-sampling Technique (SMOTE) is a popular oversampling technique for balancing the class distribution in an imbalanced dataset by generating synthetic examples for the minority class.

The process of SMOTE involves selecting an example from the minority class, finding its k nearest neighbors in the feature space, and generating synthetic examples by interpolating between the feature vectors of the example and its neighbors. Specifically, SMOTE generates new examples by randomly selecting one of the k nearest neighbors of the selected example, computing the difference between the selected example and the neighbor, and adding a fraction of this difference to the selected example. This process is repeated until the desired level of balance between the minority and majority classes is achieved [1].

4.3.2 MODULE 2 – MODEL WITH CROSS VALIDATION

XGBOOST (EXTREME GRADIENT BOOSTING)

XGBoost is an implementation of Gradient Boosted decision trees. This library was written in C++. It is a type of Software library that was designed basically to improve speed and model performance. It has recently been dominating in applied machine learning. XGBoost models majorly dominate in many Kaggle Competitions. In this algorithm, decision trees are created in sequential form. Weights play an important role in XGBoost. Weights are assigned to all the independent variables which are then fed into the decision tree which predicts results. The weight of variables predicted wrong by the tree is increased and the variables are then fed to the second decision tree. These individual classifiers/predictors then ensemble to give a strong and more precise model. It can work on regression, classification, ranking, and user-defined prediction problems [1].

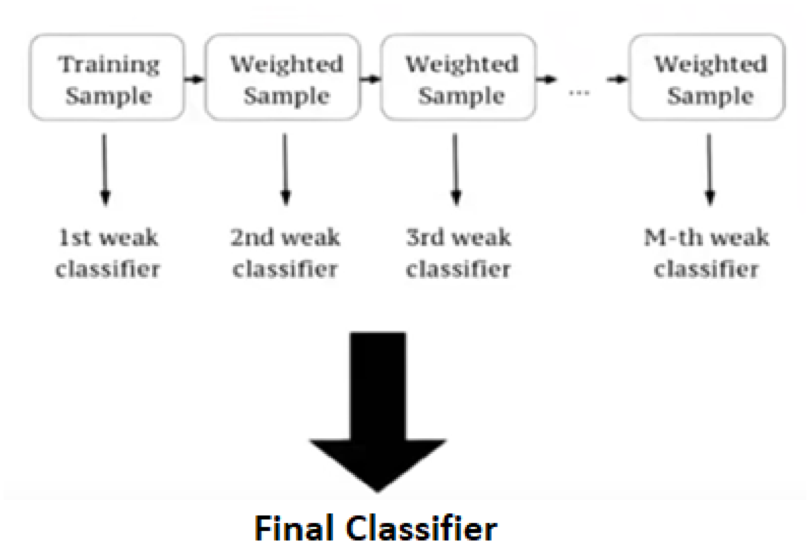


Fig. 4.2: XGB Classifier

Image source: https://rb.gy/n0x94

HYPERPARAMETER TUNING USING RANDOM SEARCH

Hyperparameter tuning is an essential step in machine learning model development. It involves selecting the optimal combination of hyperparameters for a given model to achieve the best possible performance on a specific dataset. However, manually tuning hyperparameters can be time-consuming and challenging, especially for complex models.

Random search is a hyperparameter tuning technique that addresses this issue by searching over a range of hyperparameters randomly. This technique selects random combinations of hyperparameters and evaluates their performance on a validation set to identify the best set of hyperparameters for the model. Random search is an effective approach because it can efficiently explore a wide range of hyperparameters and identify a good set of hyperparameters quickly.

In the proposed system, we use random search to tune the hyperparameters of the XGBoost classifier. The hyperparameters we tune include the maximum depth of each tree (max_depth), the learning rate (learning_rate), the number of estimators (n_estimators), the minimum sum of weights of all observations required in a child (min_child_weight), and the minimum loss reduction required to make a further partition on a leaf node (gamma). We define a search space for each hyperparameter with a range of possible values and use randomized search to sample from each search space.

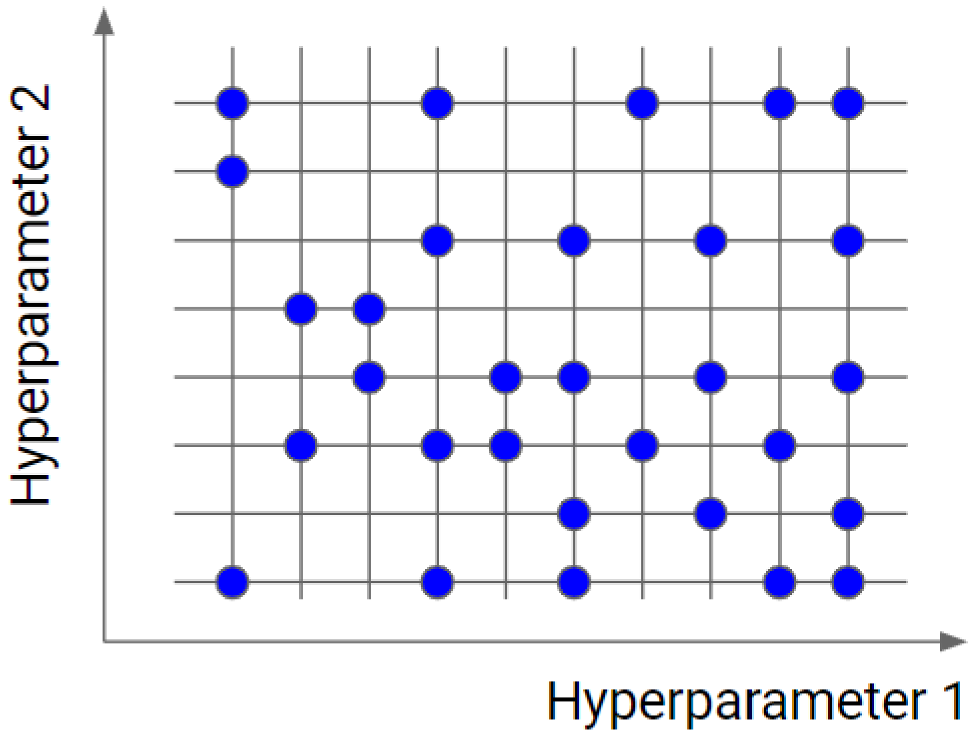


Fig. 4.3: Hyperparameter tuning

Image Source: https://rb.gy/oezj8

STRATIFIED K FOLD CROSS VALIDATION

Cross-validation is a technique used to evaluate the performance of machine learning models on a limited dataset. Stratified k-fold cross-validation is a type of cross-validation that is particularly useful for classification problems when the target variable is imbalanced. In stratified k-fold cross-validation, the dataset is divided into k folds, with each fold containing roughly the same proportion of positive and negative examples as the original dataset. During training, k-1 folds are used for training the model, and the remaining fold is used for validation. This process is repeated k times, with each fold used for validation once. The final performance of the model is then calculated by averaging the performance across all k folds. In the proposed system, we use stratified 10-fold cross-validation to evaluate the performance of different hyperparameter combinations for the XGBoost classifier. We ensure that each fold contains the same proportion of positive and negative examples, preserving the class distribution of the data. This technique allows us to identify the optimal set of hyperparameters for the classifier that achieves the highest accuracy score on the resampled dataset while maintaining good generalization performance [1].

4.4 IMPLEMENTATION

SMOTE SAMPLING

```
import pandas as pd

from sklearn.model_selection import train_test_split

from imblearn.over_sampling import SMOTE

df = pd.read_excel(r'C:\Users\awzma\Testosterone Deficiency\dataset.xlsx')

X_train, X_test, y_train, y_test = train_test_split(df.iloc[:,:-1], df.iloc[:,-1], test_size=0.2, random_state=42)

sm = SMOTE(random_state=42)

X_train_res, y_train_res = sm.fit_resample(X_train, y_train)

train_df = pd.concat([pd.DataFrame(X_train_res), pd.DataFrame(y_train_res)], axis=1)
```

XGB CLASSIFIER

```
import xgboost as xgb

X = df.iloc[:, :-1].values

y = df.iloc[:, -1].values

xgb_classifier = xgb.XGBClassifier(objective='binary:logistic', random_state=42)

xgb_classifier.fit(X_resampled, y_resampled)
```

STRATIFIED 10 FOLD CROSS VALIDATION

```
from sklearn.model_selection import StratifiedKFold, RandomizedSearchCV

from sklearn.metrics import accuracy_score, make_scorer

skf = StratifiedKFold(n_splits=10, shuffle=True, random_state=42)
```

HYPERPARAMETER TUNING WITH RANDOM SEARCH

```
import numpy as np

import pandas as pd

from sklearn.model_selection import StratifiedKFold, RandomizedSearchCV
from sklearn.metrics import accuracy_score, make_scorer

param_grid = {'max_depth': [3, 4, 5],

            'learning_rate': [0.05, 0.1, 0.15],

            'n_estimators': [100, 200, 300],

            'min_child_weight': [1, 3, 5],

            'gamma': [0, 0.1, 0.2]}

scoring_metric = make_scorer(accuracy_score)

random_search = RandomizedSearchCV(xgb_classifier, param_grid, n_iter=50, cv=skf,
scoring=scoring_metric, random_state=42, verbose=1, n_jobs=-1)

random_search.fit(X_resampled, y_resampled)

print(f"Best hyperparameters: {random_search.best_params_}")
```

ABNORMALITY AND ITS LEVEL

```
best_model = random_search.best_estimator_

accuracy_scores = best_model.predict_proba(X)[:,1]

accuracy_percents = np.round(accuracy_scores * 100, 2)

df['Level'] = accuracy_percents

df['Abnormality_Level'] = pd.cut(df['Level'], bins=[0, 40, 70, 100], labels=['Low', 'Medium',
'High'])

df_filtered = df.loc[df['T'] == 1, df.columns != 'T']
```

GENERATING NEW DATASET

```
df_filtered.to_excel(r'C:/Users/awzma/Testosterone
Deficiency/Proposed/FINAL_OUTPUT_DATASET4.xlsx', index=False)
```

CHAPTER 5

SIMULATION RESULTS/EXPERIMENTAL RESULTS

5.1 TOOLS/PLATFORM:

5.1.1 HARDWARE REQUIREMENTS:

- **RAM:** Minimum 4GB
- **ROM:** Minimum 10GB

5.1.2 SOFTWARE REQUIREMENTS:

- **OPERATING SYSTEM:** Windows 7/8/9/10/11
- **SOFTWARE TOOL:** Jupyter Notebook
- **VIRTUAL ENVIRONMENT:** Virtualenvwrapper
- **CLOUD:** Google Cloud Platfrom (GCP)

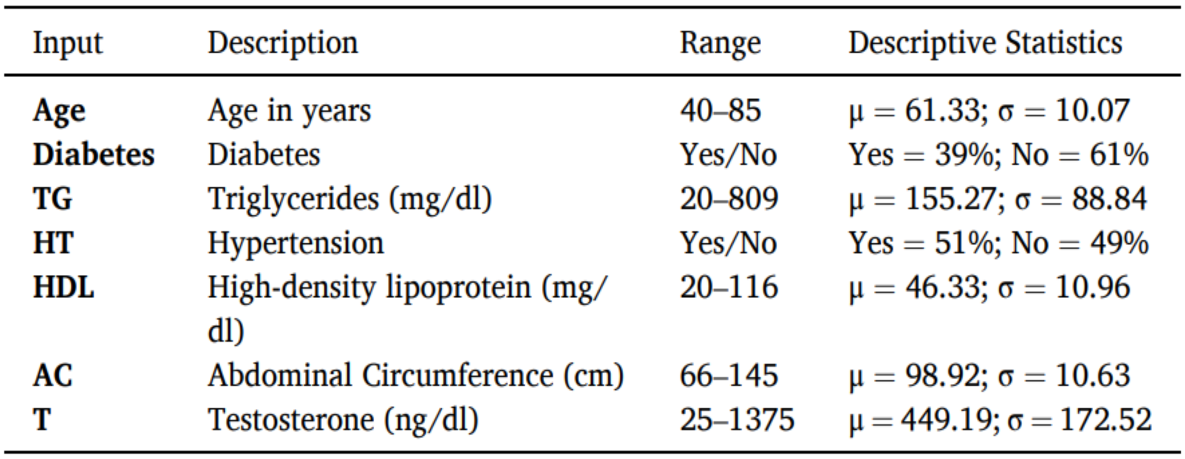
5.2 DATASET:

The dataset contains of a sample of 3397 patients between the ages of 40 and 85 drawn from a urology clinic in Feira de Santana, Brazil.

Name: Testosterone-Deficiency-Dataset .

Dataset Link: https://github.com/osmarluiz/Testosterone-Deficiency-Dataset

DATASET ATTRIBUTES:



Input	Description	Range	Descriptive Statistics
Age	Age in years	40–85	$\mu = 61.33$; $\sigma = 10.07$
Diabetes	Diabetes	Yes/No	Yes = 39%; No = 61%
TG	Triglycerides (mg/dl)	20–809	$\mu = 155.27$; $\sigma = 88.84$
HT	Hypertension	Yes/No	Yes = 51%; No = 49%
HDL	High-density lipoprotein (mg/dl)	20–116	$\mu = 46.33$; $\sigma = 10.96$
AC	Abdominal Circumference (cm)	66–145	$\mu = 98.92$; $\sigma = 10.63$
T	Testosterone (ng/dl)	25–1375	$\mu = 449.19$; $\sigma = 172.52$

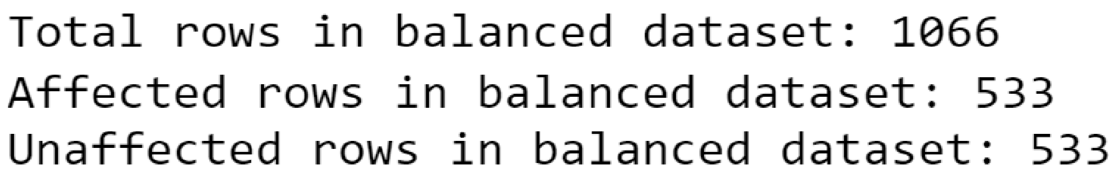
Fig. 5.2: Dataset Attributes

5.3 EXISTING SYSTEM OUTPUT:

5.3.1 MODULE 1 - SAMPLING TECHNIQUES:

UNDER SAMPLING:

- Output for Random Under Sampling is shown below:

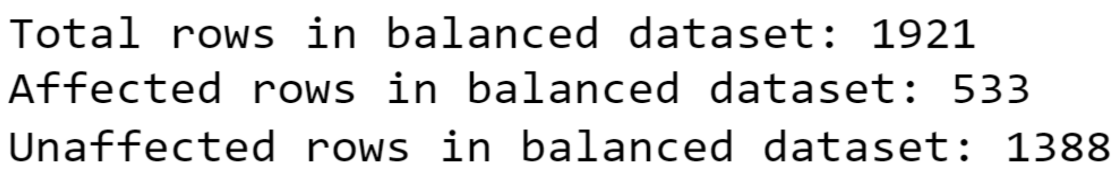


```
Total rows in balanced dataset: 1066
Affected rows in balanced dataset: 533
Unaffected rows in balanced dataset: 533
```

Fig. 5.3: RUS

The above Fig. 5.2 displays the total rows, total affected and unaffected rows produced using RUS.

- Output for Repeated Edited Nearest Neighbours Sampling is shown below:



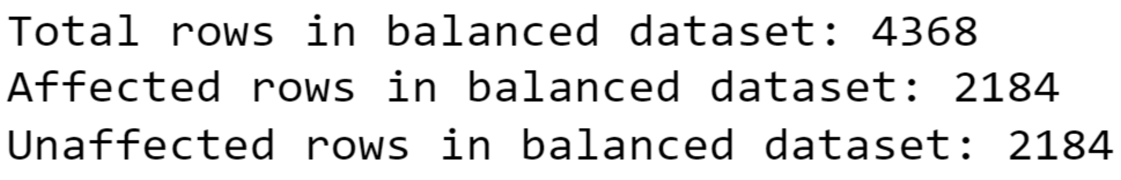
```
Total rows in balanced dataset: 1921
Affected rows in balanced dataset: 533
Unaffected rows in balanced dataset: 1388
```

Fig. 5.4 RENN

The above Fig. 5.3 displays the total rows, total affected and unaffected rows produced using RENN.

OVER SAMPLING:

- Output for Random Over Sampling is shown below:

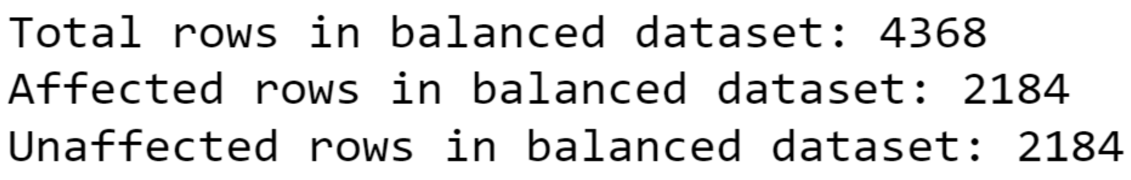


```
Total rows in balanced dataset: 4368
Affected rows in balanced dataset: 2184
Unaffected rows in balanced dataset: 2184
```

Fig. 5.5 ROS

The above Fig. 5.4 displays the total rows, total affected and unaffected rows produced using ROS.

- Output for Synthetic Minority Over Sampling is shown below:



```
Total rows in balanced dataset: 4368
Affected rows in balanced dataset: 2184
Unaffected rows in balanced dataset: 2184
```

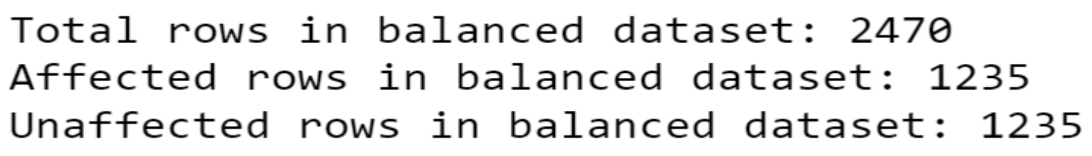
Fig. 5.6 SMOTE

The above Fig. 5.4 displays the total rows, total affected and unaffected rows produced using SMOTE.

HYBRID SAMPLING:

In hybrid sampling, it was implemented in 4 combinations as RENN + SMOTE, RUS + ROS, RENN + RUS, and RUS + SMOTE.

- Output for Repeated Edited Nearest Neighbors (RENN) + Synthetic Minority Over-sampling Technique (SMOTE) is shown below:

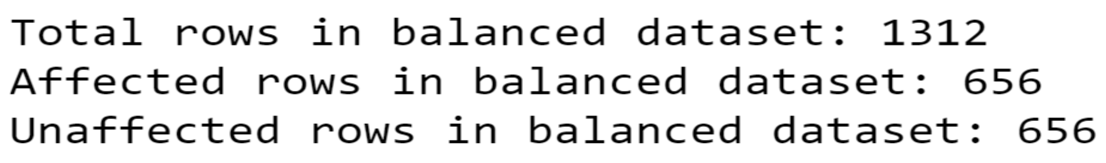


```
Total rows in balanced dataset: 2470
Affected rows in balanced dataset: 1235
Unaffected rows in balanced dataset: 1235
```

Fig. 5.7 RENN + SMOTE

The above Fig. 5.6 displays the total rows, total affected and unaffected rows produced using RENN+SMOTE.

- Output for Random Under Sampling (RUS) + Random Over Sampling (ROS) is shown below:

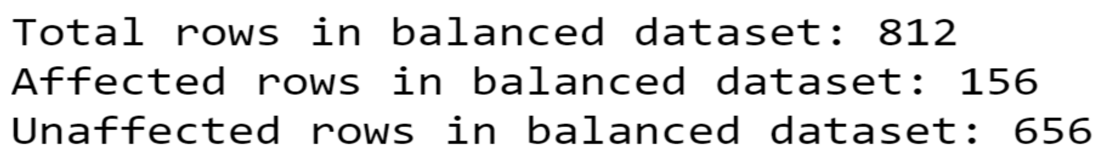


```
Total rows in balanced dataset: 1312
Affected rows in balanced dataset: 656
Unaffected rows in balanced dataset: 656
```

Fig. 5.8 RUS + ROS

The above Fig. 5.7 displays the total rows, total affected and unaffected rows produced using RUS+ROS.

- Output for Repeated Edited Nearest Neighbors (RENN) + Random Under Sampling (RUS) is shown below:

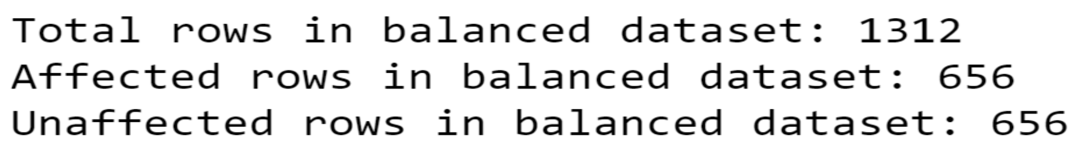


```
Total rows in balanced dataset: 812
Affected rows in balanced dataset: 156
Unaffected rows in balanced dataset: 656
```

Fig. 5.9 RENN + RUS

The above Fig. 5.8 displays the total rows, total affected and unaffected rows produced using RENN+RUS.

- Output for Random Under Sampling (RUS) + Synthetic Minority Over-sampling Technique (SMOTE) is shown below:

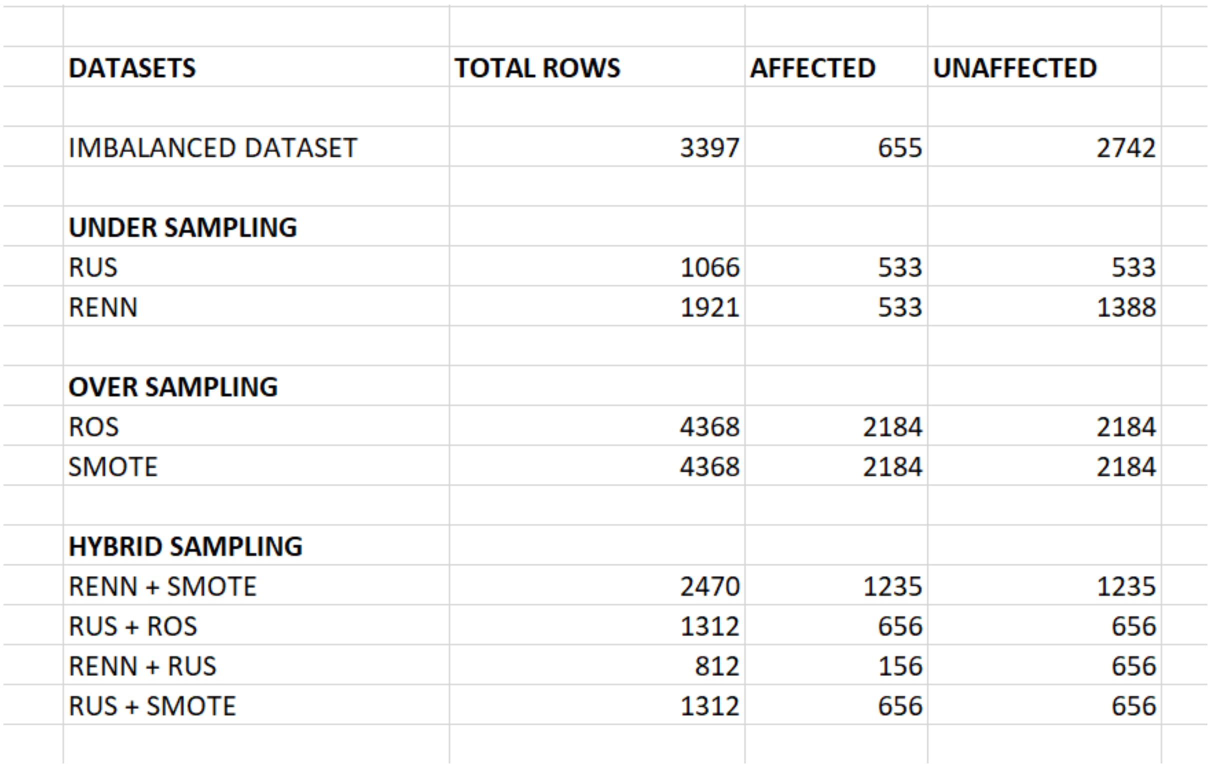


```
Total rows in balanced dataset: 1312
Affected rows in balanced dataset: 656
Unaffected rows in balanced dataset: 656
```

Fig. 5.10 RUS + SMOTE

The above Fig. 5.9 displays the total rows, total affected and unaffected rows produced using RUS+SMOTE.

COMPARISON TABLE OF ALL SAMPLING TECHNIQUES:



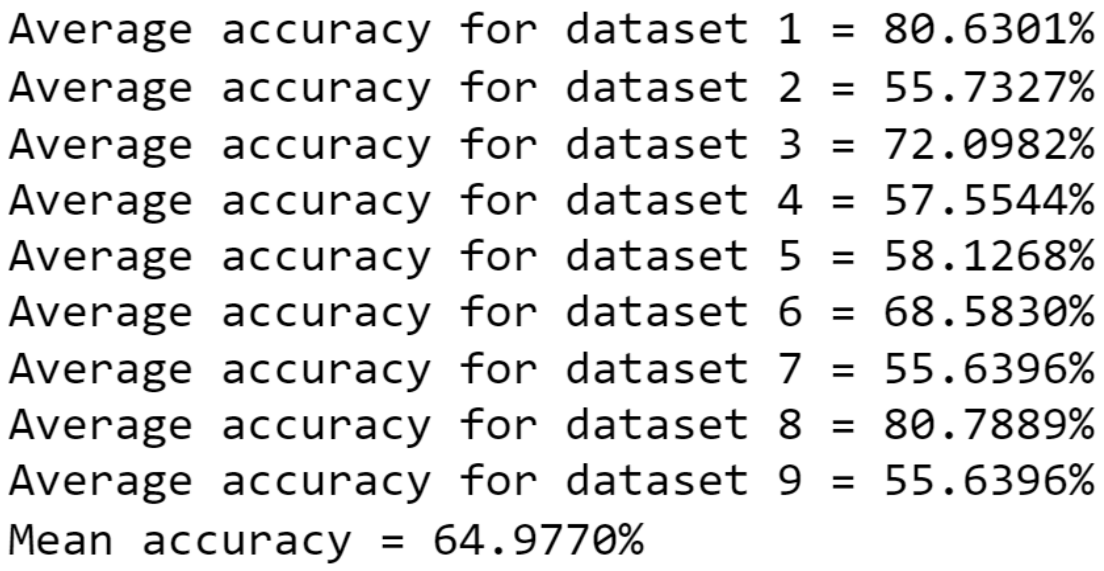
DATASETS	TOTAL ROWS	AFFECTED	UNAFFECTED
IMBALANCED DATASET	3397	655	2742
UNDER SAMPLING			
RUS	1066	533	533
RENN	1921	533	1388
OVER SAMPLING			
ROS	4368	2184	2184
SMOTE	4368	2184	2184
HYBRID SAMPLING			
RENN + SMOTE	2470	1235	1235
RUS + ROS	1312	656	656
RENN + RUS	812	156	656
RUS + SMOTE	1312	656	656

Fig. 5.11: Comparison of all Samplings

5.3.2 MODULE 2 – MODELS WITH CROSS VALIDATION:

MULTILAYER PERCEPTRON MODEL:

- The following Fig. 5.12 shows the output of accuracy and average accuracy of different sampling dataset using MLP Classifier along with cross validation:

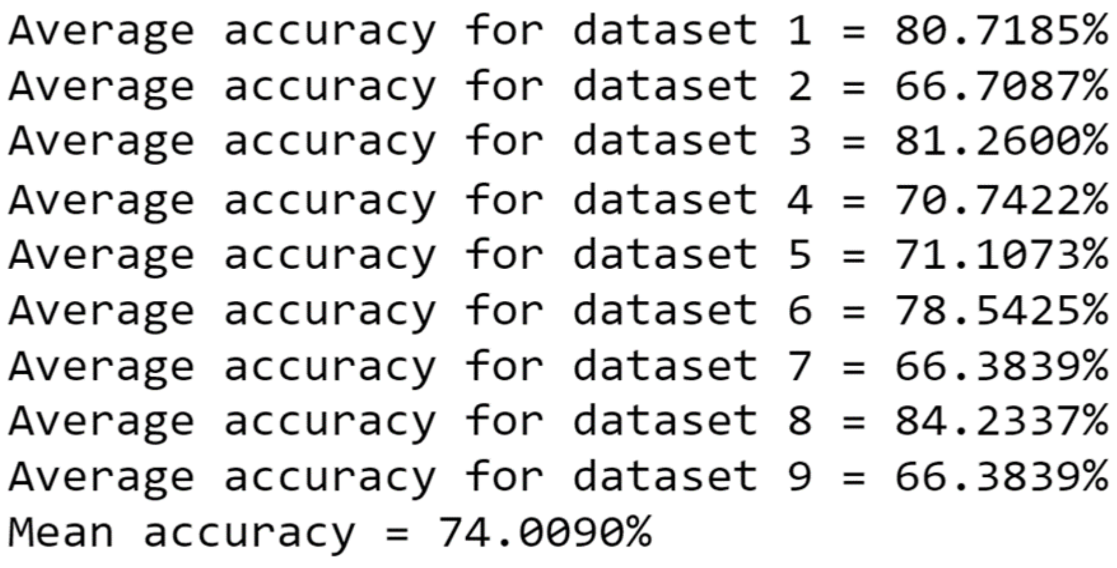


```
Average accuracy for dataset 1 = 80.6301%
Average accuracy for dataset 2 = 55.7327%
Average accuracy for dataset 3 = 72.0982%
Average accuracy for dataset 4 = 57.5544%
Average accuracy for dataset 5 = 58.1268%
Average accuracy for dataset 6 = 68.5830%
Average accuracy for dataset 7 = 55.6396%
Average accuracy for dataset 8 = 80.7889%
Average accuracy for dataset 9 = 55.6396%
Mean accuracy = 64.9770%
```

Fig. 5.12: MLP Output

ARTIFICIAL NEURAL NETWORK:

- The following Fig. 5.13 shows the output of accuracy and average accuracy of different sampling dataset using ANN Classifier along with cross validation:

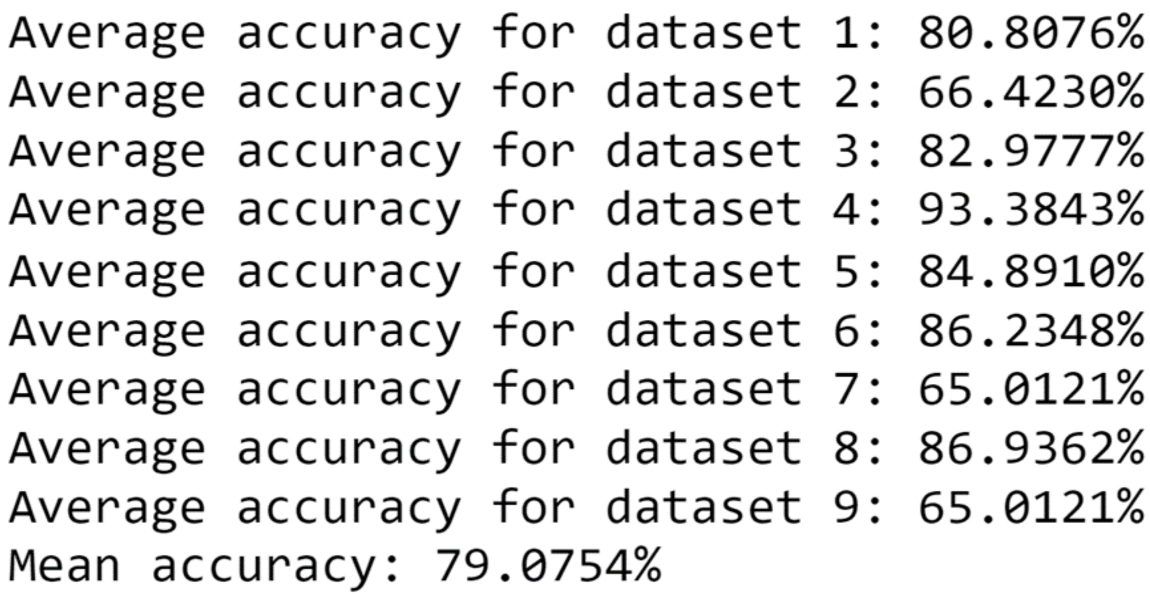


```
Average accuracy for dataset 1 = 80.7185%
Average accuracy for dataset 2 = 66.7087%
Average accuracy for dataset 3 = 81.2600%
Average accuracy for dataset 4 = 70.7422%
Average accuracy for dataset 5 = 71.1073%
Average accuracy for dataset 6 = 78.5425%
Average accuracy for dataset 7 = 66.3839%
Average accuracy for dataset 8 = 84.2337%
Average accuracy for dataset 9 = 66.3839%
Mean accuracy = 74.0090%
```

Fig. 5.13: ANN Output

RANDOM FOREST CLASSIFIER MODEL:

- The following Fig. 5.14 shows the output of accuracy and average accuracy of different sampling dataset using RF Classifier along with cross validation:

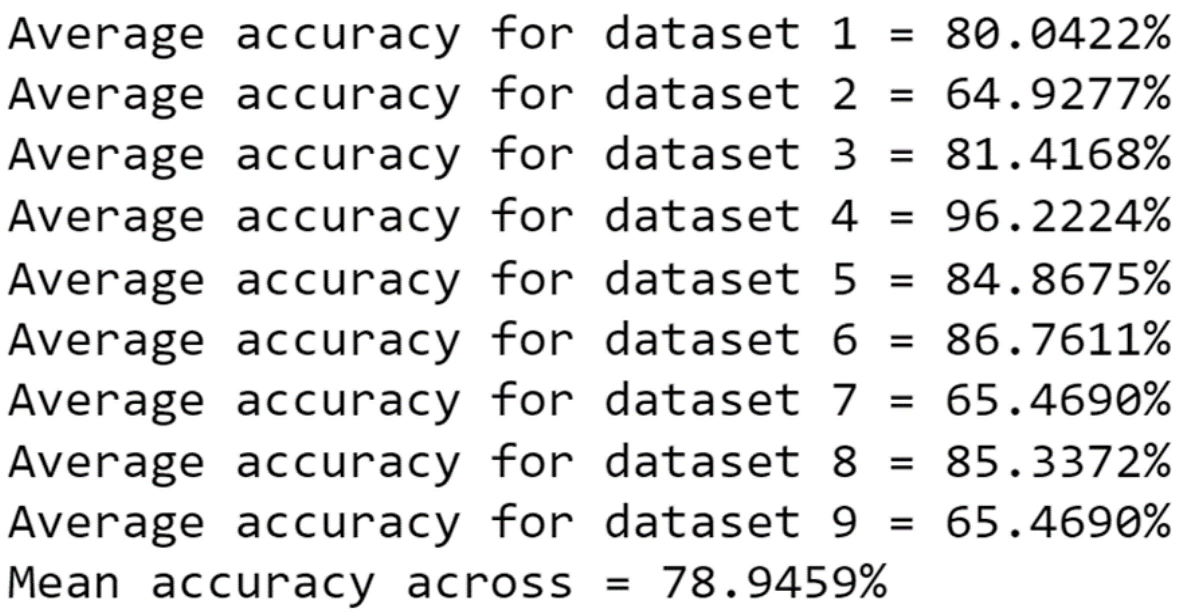


```
Average accuracy for dataset 1: 80.8076%
Average accuracy for dataset 2: 66.4230%
Average accuracy for dataset 3: 82.9777%
Average accuracy for dataset 4: 93.3843%
Average accuracy for dataset 5: 84.8910%
Average accuracy for dataset 6: 86.2348%
Average accuracy for dataset 7: 65.0121%
Average accuracy for dataset 8: 86.9362%
Average accuracy for dataset 9: 65.0121%
Mean accuracy: 79.0754%
```

Fig. 5.14: RF Output

EXTEMELY RANDOMIZED TREES:

- The following Fig. 5.15 shows the output of accuracy and average accuracy of different sampling dataset using ERT Classifier along with cross validation:

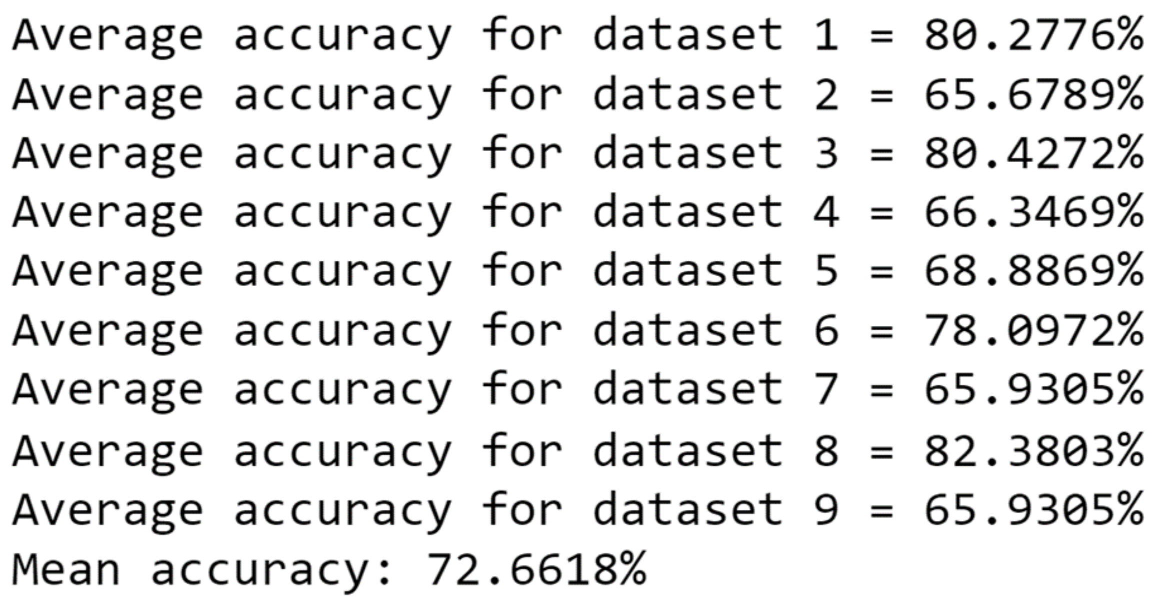


```
Average accuracy for dataset 1 = 80.0422%
Average accuracy for dataset 2 = 64.9277%
Average accuracy for dataset 3 = 81.4168%
Average accuracy for dataset 4 = 96.2224%
Average accuracy for dataset 5 = 84.8675%
Average accuracy for dataset 6 = 86.7611%
Average accuracy for dataset 7 = 65.4690%
Average accuracy for dataset 8 = 85.3372%
Average accuracy for dataset 9 = 65.4690%
Mean accuracy across = 78.9459%
```

Fig. 5.15: ERT Output

NAÏVE BAYES MODEL:

- The following Fig. 5.16 shows the output of accuracy and average accuracy of different sampling dataset using NB Classifier along with cross validation:

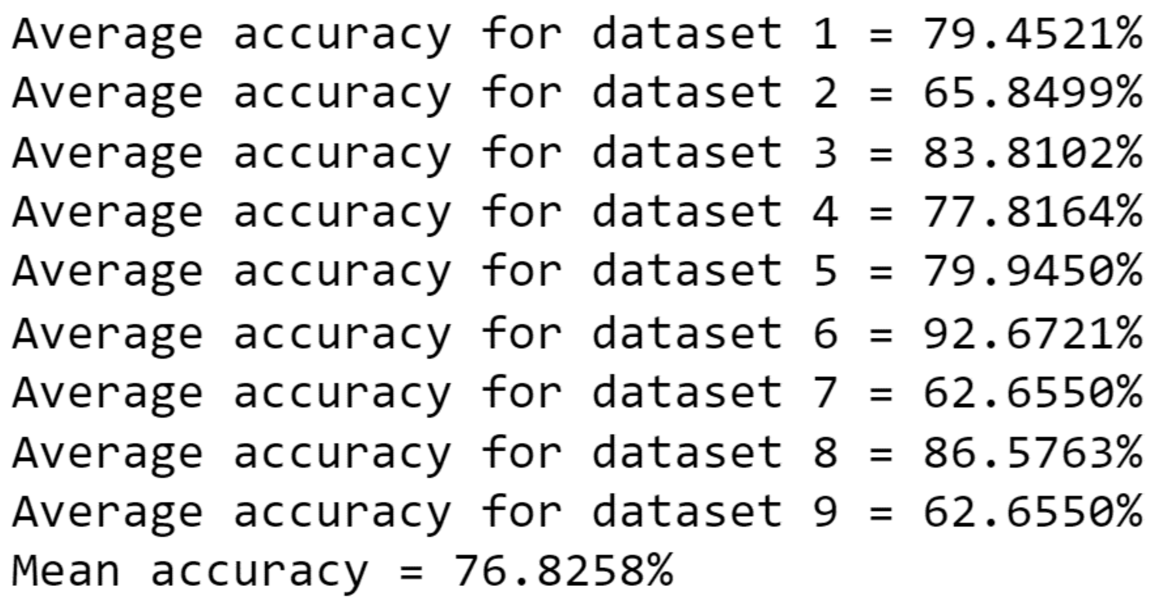


```
Average accuracy for dataset 1 = 80.2776%
Average accuracy for dataset 2 = 65.6789%
Average accuracy for dataset 3 = 80.4272%
Average accuracy for dataset 4 = 66.3469%
Average accuracy for dataset 5 = 68.8869%
Average accuracy for dataset 6 = 78.0972%
Average accuracy for dataset 7 = 65.9305%
Average accuracy for dataset 8 = 82.3803%
Average accuracy for dataset 9 = 65.9305%
Mean accuracy: 72.6618%
```

Fig. 5.16: NB Output

K NEAREST NEIGHBOUR MODEL:

- The following Fig. 5.17 shows the output of accuracy and average accuracy of different sampling dataset using KNN Classifier along with cross validation:

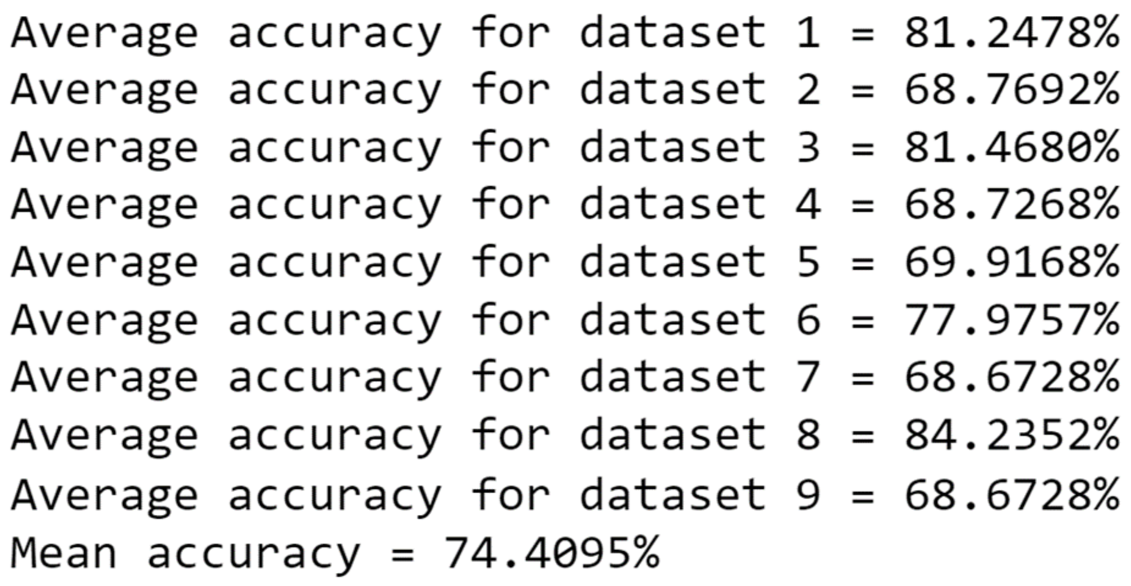


```
Average accuracy for dataset 1 = 79.4521%
Average accuracy for dataset 2 = 65.8499%
Average accuracy for dataset 3 = 83.8102%
Average accuracy for dataset 4 = 77.8164%
Average accuracy for dataset 5 = 79.9450%
Average accuracy for dataset 6 = 92.6721%
Average accuracy for dataset 7 = 62.6550%
Average accuracy for dataset 8 = 86.5763%
Average accuracy for dataset 9 = 62.6550%
Mean accuracy = 76.8258%
```

Fig. 5.17: KNN Output

LOGISTIC REGRESSION MODEL:

- The following Fig. 5.18 shows the output of accuracy and average accuracy of different sampling dataset using LR Classifier along with cross validation:

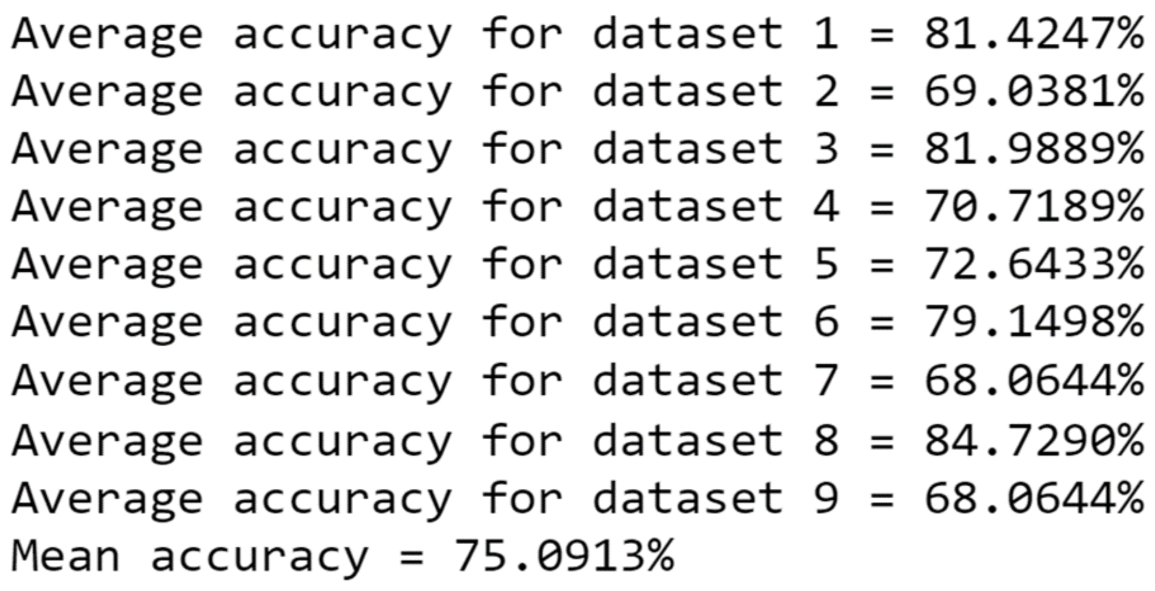


```
Average accuracy for dataset 1 = 81.2478%
Average accuracy for dataset 2 = 68.7692%
Average accuracy for dataset 3 = 81.4680%
Average accuracy for dataset 4 = 68.7268%
Average accuracy for dataset 5 = 69.9168%
Average accuracy for dataset 6 = 77.9757%
Average accuracy for dataset 7 = 68.6728%
Average accuracy for dataset 8 = 84.2352%
Average accuracy for dataset 9 = 68.6728%
Mean accuracy = 74.4095%
```

Fig. 5.18: LR Output

ADA BOOST MODEL:

- The following Fig. 5.19 shows the output of accuracy and average accuracy of different sampling dataset using ADABOOST Classifier along with cross validation:

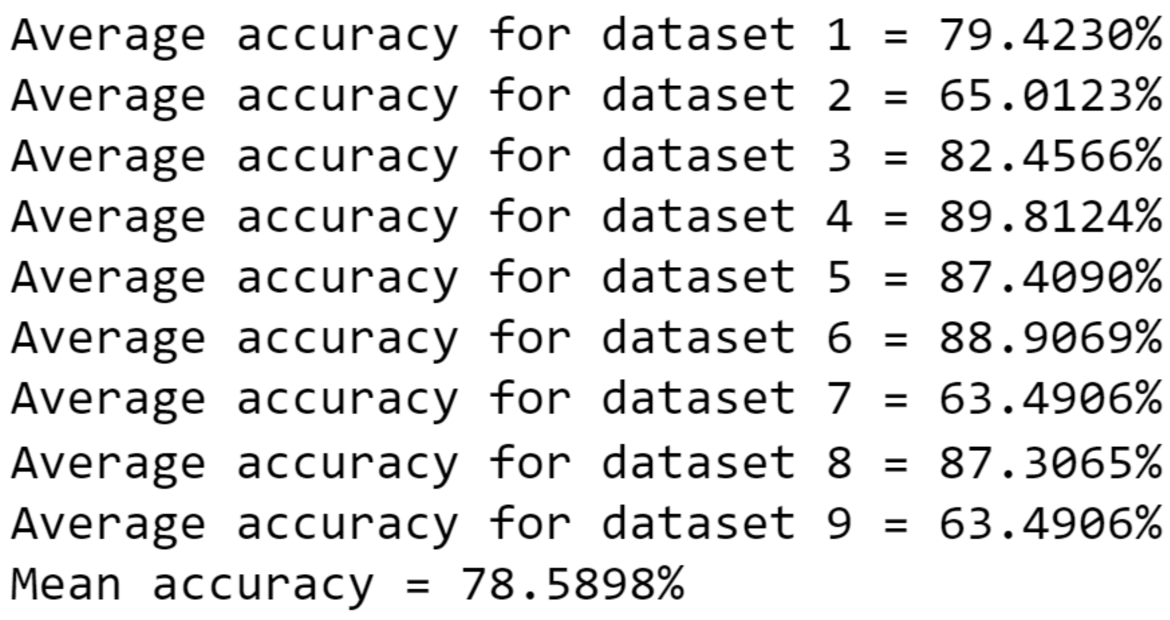


```
Average accuracy for dataset 1 = 81.4247%
Average accuracy for dataset 2 = 69.0381%
Average accuracy for dataset 3 = 81.9889%
Average accuracy for dataset 4 = 70.7189%
Average accuracy for dataset 5 = 72.6433%
Average accuracy for dataset 6 = 79.1498%
Average accuracy for dataset 7 = 68.0644%
Average accuracy for dataset 8 = 84.7290%
Average accuracy for dataset 9 = 68.0644%
Mean accuracy = 75.0913%
```

Fig. 5.19: AdaBoost Output

XGBOOST MODEL:

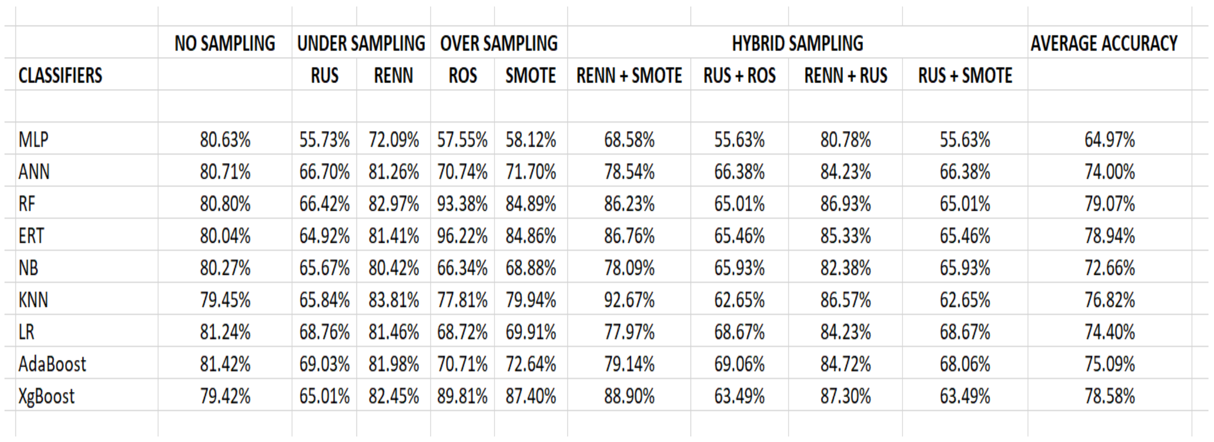
- The following Fig. 5.20 shows the output of accuracy and average accuracy of different sampling dataset using XGB Classifier along with cross validation:



```
Average accuracy for dataset 1 = 79.4230%
Average accuracy for dataset 2 = 65.0123%
Average accuracy for dataset 3 = 82.4566%
Average accuracy for dataset 4 = 89.8124%
Average accuracy for dataset 5 = 87.4090%
Average accuracy for dataset 6 = 88.9069%
Average accuracy for dataset 7 = 63.4906%
Average accuracy for dataset 8 = 87.3065%
Average accuracy for dataset 9 = 63.4906%
Mean accuracy = 78.5898%
```

Fig. 5.20: XgBoost Output

COMPARISON OF ALL CLASSIFIERS WITH CROSS VALIDATION FOR DIFFERENT SAMPLING DATASETS:



	NO SAMPLING	UNDER SAMPLING		OVER SAMPLING		HYBRID SAMPLING				AVERAGE ACCURACY
CLASSIFIERS		RUS	RENN	ROS	SMOTE	RENN + SMOTE	RUS + ROS	RENN + RUS	RUS + SMOTE	
MLP	80.63%	55.73%	72.09%	57.55%	58.12%	68.58%	55.63%	80.78%	55.63%	64.97%
ANN	80.71%	66.70%	81.26%	70.74%	71.70%	78.54%	66.38%	84.23%	66.38%	74.00%
RF	80.80%	66.42%	82.97%	93.38%	84.89%	86.23%	65.01%	86.93%	65.01%	79.07%
ERT	80.04%	64.92%	81.41%	96.22%	84.86%	86.76%	65.46%	85.33%	65.46%	78.94%
NB	80.27%	65.67%	80.42%	66.34%	68.88%	78.09%	65.93%	82.38%	65.93%	72.66%
KNN	79.45%	65.84%	83.81%	77.81%	79.94%	92.67%	62.65%	86.57%	62.65%	76.82%
LR	81.24%	68.76%	81.46%	68.72%	69.91%	77.97%	68.67%	84.23%	68.67%	74.40%
AdaBoost	81.42%	69.03%	81.98%	70.71%	72.64%	79.14%	69.06%	84.72%	68.06%	75.09%
XgBoost	79.42%	65.01%	82.45%	89.81%	87.40%	88.90%	63.49%	87.30%	63.49%	78.58%

Fig. 5.21: Comparison of all Classifiers

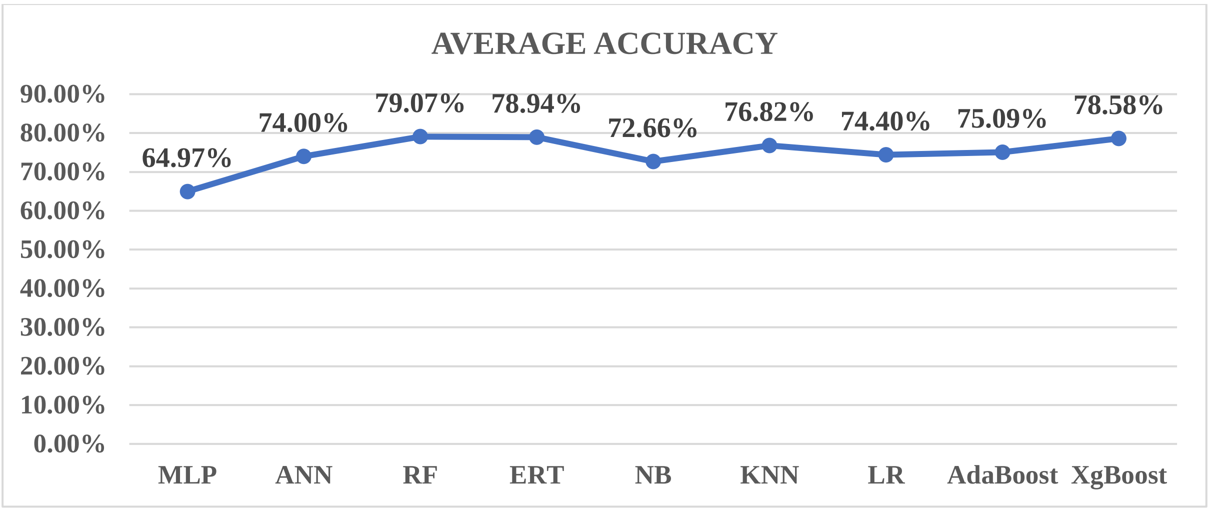


Fig. 5.22 Comparison Graph

The above Fig. 5.22 represents among all the Classifiers with Over Sampling the RF(Random Forest) and ERF(Extremely Random Forest) performed the best with the accuracy of 93 and 96 percent.

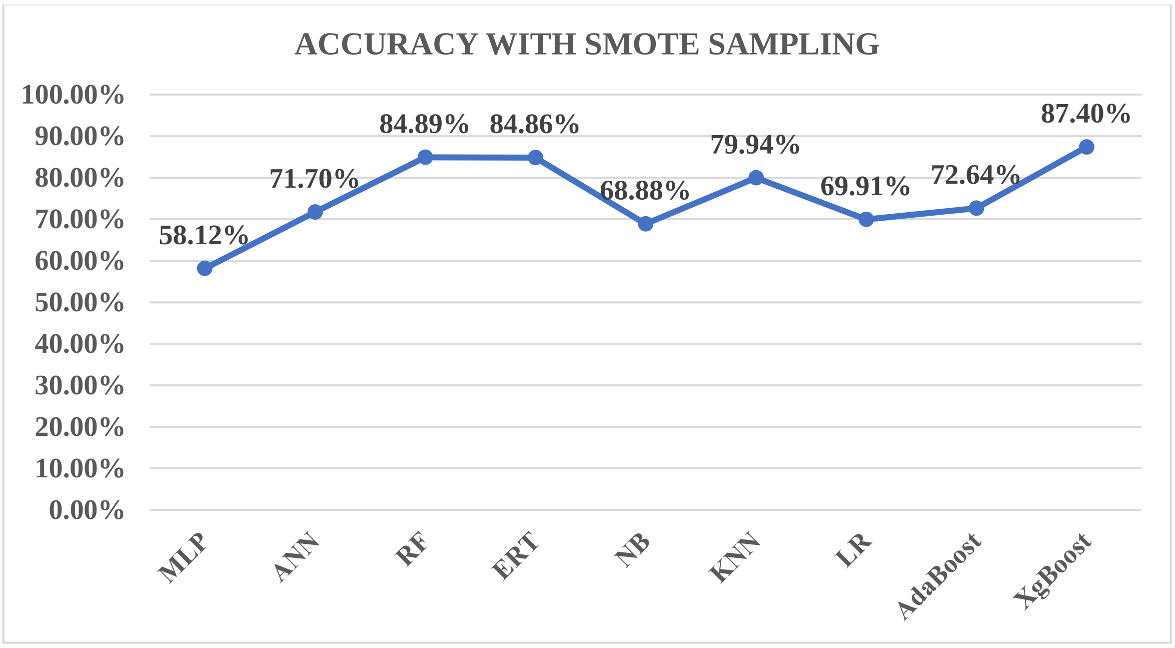


Fig. 5.23 Comparison Graph for SMOTE

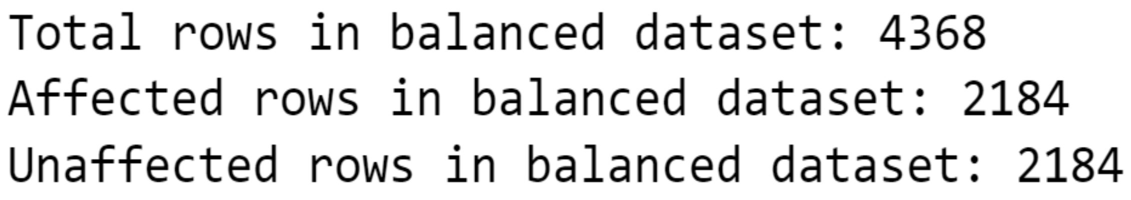
The above Fig. 5.23 represents using Cross Validation with all datasets RF(79.07%) , ERT(78.94%) and XgBoost(78.58%) produced superior results among all the classifiers used.

5.4 PROPOSED SYSTEM OUTPUT:

5.4.1 MODULE 1 - SAMPLING TECHNIQUE:

SMOTE SAMPLING:

- Output for Synthetic Minority Over Sampling is shown below:



```
Total rows in balanced dataset: 4368
Affected rows in balanced dataset: 2184
Unaffected rows in balanced dataset: 2184
```

Fig. 5.24: SMOTE Output

The above Fig. 5.24 displays the total rows, total affected and unaffected rows produced using SMOTE.

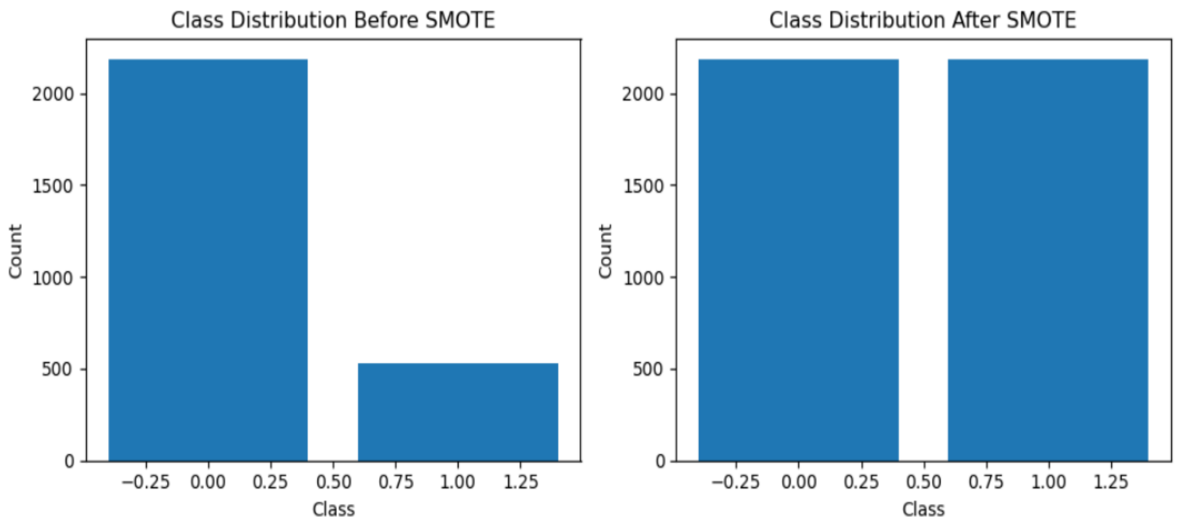


Fig. 5.25: Before & After of SMOTE

The above Fig. 5.25 represents the total rows of both classes (0 and 1) in dataset of before and after SMOTE sampling.

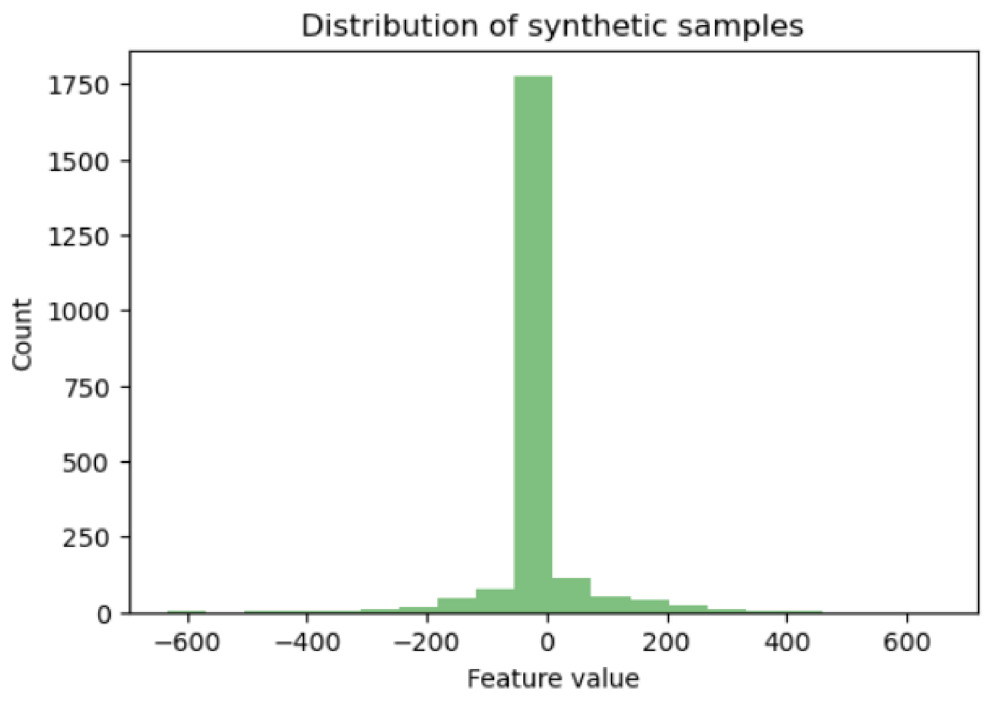
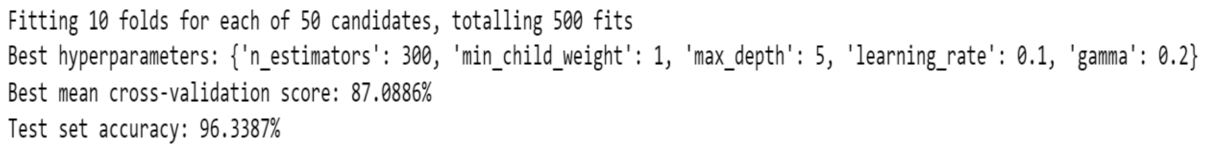


Fig. 5.26: Synthetic samples in dataset

The above Fig. 5.26 represents the total synthetic samples generated in dataset using SMOTE sampling.

5.4.2 MODULE 2 – MODEL WITH CROSS VALIDATION:

- Using XgBoost with stratified k fold cross validation and hyperparameter tuning using random search helps to predicts the accuracy of 96.33%.

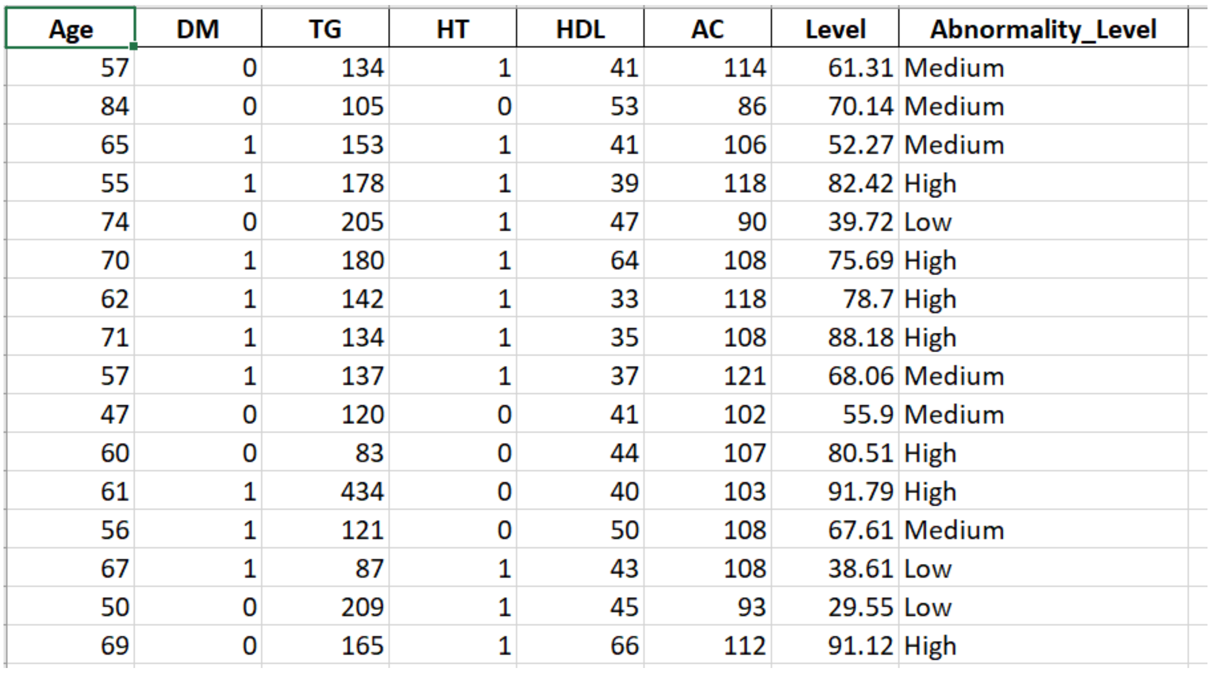


```
Fitting 10 folds for each of 50 candidates, totalling 500 fits
Best hyperparameters: {'n_estimators': 300, 'min_child_weight': 1, 'max_depth': 5, 'learning_rate': 0.1, 'gamma': 0.2}
Best mean cross-validation score: 87.0886%
Test set accuracy: 96.3387%
```

Fig. 5.27: Model Accuracy

The above Fig. 5.27 shows the final accuracy of 96.33% and best combination of hyperparameter using XGB classifier along with cross validation and hyperparameter tuning using random search.

- Here, final output dataset with deficiency level for each rows and abnormality level.



Age	DM	TG	HT	HDL	AC	Level	Abnormality_Level
57	0	134	1	41	114	61.31	Medium
84	0	105	0	53	86	70.14	Medium
65	1	153	1	41	106	52.27	Medium
55	1	178	1	39	118	82.42	High
74	0	205	1	47	90	39.72	Low
70	1	180	1	64	108	75.69	High
62	1	142	1	33	118	78.7	High
71	1	134	1	35	108	88.18	High
57	1	137	1	37	121	68.06	Medium
47	0	120	0	41	102	55.9	Medium
60	0	83	0	44	107	80.51	High
61	1	434	0	40	103	91.79	High
56	1	121	0	50	108	67.61	Medium
67	1	87	1	43	108	38.61	Low
50	0	209	1	45	93	29.55	Low
69	0	165	1	66	112	91.12	High

Fig. 5.28: Final Output DS

CONCLUSION

Testosterone Deficit Syndrome (TDS) significantly impairs men's quality of life, but its timely and accurate diagnosis poses a challenge in areas with poor access to public health services. TT and FT levels are not measured during routine inspections and their tests are often expensive. This leads to an effort to obtain cheaper predictive methods to filter and guide patients in undergoing further, more specific, and high-priced exams. Secondary hypogonadism occurs in conjunction with other disorders, which facilitates its detection. This paper presented a broad comparison of ten well-known ML classifiers, eight sampling methods for imbalanced datasets, and the calibration of the predicted probabilities to identify a proper method to improve accuracy in TDS diagnosis using only features obtained from low-cost routine exams.

In the existing system, they try to improve the accurate level, which they got around 70 to 80% by using of the Classification algorithm, Sampling and Probability calibration. The use of different classifiers bearing low correlations between one another presents better results for the ensemble classifiers. In Sampling strategies, ML algorithms often present bad classification results in imbalanced data. This research compared eight sampling strategy combinations using the open-source python toolbox Imbalanced-Learn Random under sampling (RUS), Repeated Edited Nearest Neighbours (RENN), Random Over-Sampling (ROS), Synthetic Minority Over-sampling Technique (SMOTE), RENN + SMOTE, RUS + ROS, RENN + RUS, and RUS + SMOTE.

In our proposed system, In our dataset, most of the data is unaffected which is mean the data set is imbalanced To overcome this problem we using sampling technique which keep balance of the data set by using different sampling techniques. By using sampling techniques (SMOTE) we create a new dataset that is trained under XgBoost classifiers and stratified K fold cross-validation. Finally, we conclude whether the patient has testosterone or not. When the result shows a positive it further moves into Next which they find the level of abnormality. When the testosterone is negative they move to end the process. The level of abnormality is presented by the age of the people lies between 0 to 40 is low, 41 to 70 is medium, 71 to 100 is High.

Overall, we found the accuracy level up to 96% by using various algorithms mentioned above.

FUTURE ENHANCEMENT

Expanding the dataset used for training the machine learning model to improve its accuracy and effectiveness. Additionally, the current model could be further optimized by incorporating deep learning techniques such as neural networks and add more input attributes, such as lifestyle factors or medication usage, to further improve the accuracy of the model.

REFERENCES

[1] Monique Tonani Novaes, Osmar Luiz Ferreira de Carvalho, Pedro Henrique Guimar Ferreira, Taciana Leonel Nunes Tiraboschi, Caroline Santos Silva, Jean Carlos Zambrano, Cristiano Mendes Gomes, Eduardo de Paula Miranda, Osmar Abílio de Carvalho Junior, and Jos´e de Bessa Junior,"Prediction of secondary testosterone deficiency using machine learning: A comparative analysis of ensemble and base classifiers, probability calibration, and sampling strategies in a slightly imbalanced dataset" , Science Direct Journal Informatics in Medicine Unlocked 23 (2021) 100538

[2] Quan Zou1, Kaiyang Qu, Yamei Luo, Dehui Yin, Ying Ju, and Hua Tang," Predicting Diabetes Mellitus With Machine Learning Techniques", Article of Frontiers in Genetics November 2018 | Volume 9 | Article 515

[3] Ti Lu, Ya-Han Hu, Chih-Fong Tsai, Shih-Ping Liu, and Pei-Ling Chen," Applying machine learning techniques to the identification of late-onset hypogonadism in elderly men", SpringerPlus Article June 2016

[4] Shaker El-Sappagh, Mohammed Elmogy , Farman Ali, Tamer ABUHMED, S. M. Riazul Islam, and Kyung-Sup Kwak," A Comprehensive Medical Decision–Support Framework Based on a Heterogeneous Ensemble Classifier for Diabetes Prediction", MDPI Article May 2019.

[5] Rahul C, Merin Meleet, G.N. Srinivasan, and Nagaraj G Cholli," Clinical Data Analysis For Recognizing Named Entities", 2021 IEEE International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS) November 2021

[6] Amin Ul Haq1, Jianping Li1*, Jalaluddin Khan1, Muhammad Hammad Memon1, Shadma Parveen3, Mordecai Folarin Raji1, Wasif Akbar1, Tanvir Ahmad2, Sana Ullah4, Latipova Shoista1, Happy N. Monday1," Identifying The Predictive Capability of Machine Learning Classifiers for Designing Heart Disease Detection System", 16th International Computer Conference on Wavelet Active Media Technology and Information Processing 2019

[7] Nabaouia Louridi, Meryem Amar, Bouabid and El Ouahidi, "Identification of Cardiovascular Diseases Using Machine Learning", 7th Mediterranean Congress of Telecommunications (CMT) 2019

[8] Jian Ping Li ;Amin Ul Haq;Salah Ud Din, Jalaluddin Khan, Asif Khan, and Abdus Saboor,"Heart Disease Identification Method Using Machine Learning Classification in E-Healthcare", IEEE Journal Vol: 8, 2020

[9] Carrageta DF, Oliveira PF, Alves MG, Monteiro MP. Obesity and male hypogonadism: tales of a vicious cycle. Obes Rev 2019

[10] Pizzol D, Smith L, Fontana L, Caruso MG, Bertoldo A, Demurtas J, et al. Associations between body mass index, waist circumference and erectile dysfunction: a systematic review and META-analysis. Rev Endocr Metab Disord 2020