CS337 - Project Report

210050108 Preetham
210050161 Mahanth
210050050 Faiz
210050046 Subhash

August 14, 2024

Abstract

In this project, we will be analysing a large number of machine learning models in order to be used in the field of Internet of Medical Things (IoMT), specifically in the case of predicting diabetes mellitus (type 2 diabetes). Using the Pima Indians Diabetes database in order to train and validate the models, we analyze the accuracy, precision, AUC and many other metrics in order to evaluate the best working model for the task in hand. This project shows the significance of machine learning in healthcare and predicive analysis, and can act as a secondary opinion to provide assistance to workers of the medical field.

1 Introduction

Diabetes mellitus is a chronic metabolic disorder characterized by elevated levels of blood glucose, resulting from either insufficient insulin production, ineffective utilization of insulin, or a combination of both. This condition starts from disruptions in the body's ability to regulate blood sugar, leading to long-term complications affecting various organs and systems. It is caused due to multiple factors such as Insulin Resistance, genetics, age, genetics and other lifestyle factors like poor diet, obesity and lack of physical activity. Although there is no cure for the disease, we can however predict early onset of the disease, and take many treatment measures that can reduce the future cost of treatment, as well as prolong life.

2 Literature Review

2.1 Anomaly Detection and Removal

Anomaly detection is the identification of rare datapoints in our dataset that deviates from the majority of the data. These points are called outliers/anomalies, and may cause inconsistent training for our machine learning models. There are a large number of anomaly detection methods, and in this paper, we implement IQR and Z-score methods for outlier detection.

2.1.1 Inter Quartile Range

Q1 (First Quartile): This is the median of the lower half of the dataset, representing the point below which 25% of the data falls. **Q3 (Third Quartile)**: This is the median of the upper half of the dataset, representing the point below which 75% of the data falls. Mathematically, the IQR is expressed as:

$$\text{IQR} = Q3 - Q1$$

Outliers are identified as data points falling below

$$\text{Lower\_Bound} = \text{Q1} - 1.5 \times \text{IQR}$$

$$\text{Upper\_Bound} = \text{Q3} + 1.5 \times \text{IQR}$$

It's particularly useful because it's less influenced by extreme values or outliers in the dataset compared to the range, making it a more robust measure of dispersion the outliers are handled by either **removing or replacing the values** i.e values greather than Upper_Bound by Upper_Bound and lower than Lower_Bound by Lower_Bound .Here we removed the outliers and trained the models on cleaned dataset

2.1.2 Z-score

The Z-score, also known as the standard score or z-value, is a measure of how many standard deviations a data point is from the mean of a dataset.

$$\text{Z-score} = \frac{X - \mu}{\sigma}$$

Data points with Z-scores beyond a certain threshold (commonly 3) for any 1 of the features are considered potential outliers.i.e points which are atleast 3 standard deviations from mean of data . the outliers are handled by either **removing or replacing the values** i.e features which are farther than 3 std are replaced such that they are at 3 std from center

2.2 The Selected Machine Learning Algorithms

Among the machine learning models which we have chosen, here is a review of a few machine learning algorithms that are relatively advanced. The other models which we have chosen are Logistic Regression, Decision Trees, Random Forests, K-Nearest Neighbours, AdaBoost and Support Vector Machines.

2.2.1 SMOTE based LSTM

The challenge is to achieve diabetic classification, with good classifier performance, from a class-imbalance problem. The proposed SMOTE-based deep LSTM architecture overcomes these class imbalance challenges. Initially, the data is preprocessed and then the outlier detection is done through EDA. The normalization of the data set is performed by using the min-max transformation method. The min-max normalization formula is given in below equation :

$$x' = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}} \tag{1}$$

where x_i represents each instance of input, x' represents scaled input data and x_{max},x_{min} denotes maximum and minimum value in the input x. This preserves the relationship between data.

The robustness and generalization performance of classifiers trained on imbalanced data sets should be reduced. This is a key problem of machine learning. The SMOTE method handles imbalanced data and as all the columns in our dataset are numerical, it is acceptable for SMOTE. SMOTE's core idea is to insert new samples at random between minority samples and their neighbors. The K-nearest neighbors is first examined from the samples of minority-class samples. Assuming that the data set's sampling magnification is N, there are N samples from K-NN and K > N. The interpolation formula of SMOTE is given by the below equation:

$$S_i = X + \text{rand}(0,1) \times (y_i - X) \tag{2}$$

where X denotes a data sample in minority-class samples, $rand(0,1)$ denotes a random number from the interval (0,1), y_i represents the ith nearest neighbor, and S_i is the interpolated sample.

For LSTM, the input data needs to be time-series data. Hence, here the balanced data set needs to be reshaped into 3D data. The input data set consists of eight attributes represented as x_i in below equation. Further, the raw balanced samples are reshaped into 3D data as in below equation, which satisfies the input requirements for deep-learning model.

$$\text{X-train} = \text{X-train.reshape}(\text{X-train.shape}[0], \text{X-train.shape}[1], 1)$$

$$\text{X-test} = \text{X-test.reshape}(\text{X-test.shape}[0], \text{X-test.shape}[1], 1)$$

Each row in your reshaped data now represents a sequence of length 8, and at each time step, there is a single feature. This can be useful when you want to model temporal dependencies in your data using neural networks designed for sequence processing.
SMOTE is a common oversampling strategy for dealing with class-imbalanced data sets that produces new minority-class samples. After oversampling, the balanced data is reshaped into 3D data. Then the data is divided into training and testing data. SMOTE is based on selecting the number of nearest neighbors, K = 5. **our model has 1 LSTM layer**. As the number of layers increases, the test accuracy increases upto a level and after that due to overfitting the accuracy decreases. The overfitting problem is solved by using a dropout hyperparameter. Adaptive moment estimation (Adam) is used for optimization.

Why this **SMOTE** mixed with **LSTM** model has **high accuracy** ?

- **Feature Extraction:**
  LSTMs can automatically extract relevant features from sequential data. In diabetes reports, various factors like glucose readings, insulin intake, bloodpressure, age, and other health metrics are often interconnected. LSTMs can learn to extract meaningful representations from these complex interrelationships.

- **Enhanced Learning for Minority Class:**
  Medical datasets, including diabetes reports, often suffer from class imbalance, where the occurrences of certain medical conditions (like a rare diabetic condition) might be significantly lower than others. SMOTE creates synthetic samples for the minority class, providing the model with more instances to learn from. This prevents the model from being biased toward the majority class and helps it better discern patterns in the minority class.

2.2.2 XGBoost Classifier

Extreme Gradient Boosting is a tree-based algorithm,objective functions is that they consist of two parts: training loss(log loss or binary crosss entrophy) and regularization term:

$$\text{obj} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t)}) + \sum_{i=1}^{t} \omega(f_i)$$

there is one important thing, the regularization term! the complexity of the tree $\omega(f)$. first refine the definition of the tree $f(x)$ as

$$f_t(x) = w_{q(x)}, w \in R^T, q : R^d \to \{1, 2, \cdots, T\}.$$

Here w is the vector of scores on leaves, q is a function assigning each data point to the corresponding leaf, and T is the number of leaves. In XGBoost, we define the complexity as

$$\omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

We write the prediction value at step t as $\hat{y}_i^{(t)}$. Then we have

$$\begin{aligned}
\hat{y}_i^{(0)} &= 0 \\
\hat{y}_i^{(1)} &= f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i) \\
\hat{y}_i^{(2)} &= f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_2(x_i) \\
&\dots \\
\hat{y}_i^{(t)} &= \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i)
\end{aligned}$$

$$\begin{aligned}
\text{obj}^{(t)} &= \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t)}) + \sum_{i=1}^{t} \omega(f_i) \\
&= \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \omega(f_t) + \text{constant}
\end{aligned}$$

Here is the magical part of the derivation. After reformulating the tree model, we can write the objective value with the t-th tree as:

$$\begin{aligned}
\text{obj}^{(t)} &\approx \sum_{i=1}^{n} [g_i w_{q(x_i)} + \frac{1}{2} h_i w_{q(x_i)}^2] + \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \\
&= \sum_{j=1}^{T} [(\sum_{i \in I_j} g_i) w_j + \frac{1}{2}(\sum_{i \in I_j} h_i + \lambda) w_j^2] + \gamma T
\end{aligned}$$

where $I_j = \{i | q(x_i) = j\}$ is the set of indices of data points assigned to the j-th leaf. Notice that in the second line we have changed the index of the summation because all the data points on the same leaf get the same score. We could further compress the expression by defining $G_j = \sum_{i \in I_j} g_i$ and $H_j = \sum_{i \in I_j} h_i$:

$$\text{obj}^{(t)} = \sum_{j=1}^{T} [G_j w_j + \frac{1}{2}(H_j + \lambda) w_j^2] + \gamma T$$

In this equation, w_j are independent with respect to each other, the form $G_j w_j + \frac{1}{2}(H_j + \lambda) w_j^2$ is quadratic and the best w_j for a given structure $q(x)$ and the best objective reduction we can get is:

$$\begin{aligned}
w_j^* &= -\frac{G_j}{H_j + \lambda} \\
\text{obj}^* &= -\frac{1}{2} \sum_{j=1}^{T} \frac{G_j^2}{H_j + \lambda} + \gamma T
\end{aligned}$$

2.2.3 Linear Discriminant Analysis

Linear Discriminant Analysis is a method used in statistics in order to perform dimensionality reduction or classification tasks. LDA attempts to find a linear transformation of the original features that maximizes the distance between the means of the different classes, as well as minimize the variance of each class. It is similar to PCA in the fact that they both try to perform dimensionality reduction. The class which the row belongs to is found using Bayes' Theorem. This method is commonly used for machine learning tasks in the medical field.

2.2.4 Quadratic Discriminant Analysis

Quadratic Discriminant Analysis is a method that is similar to LDA, in the fact that it also performs dimensionality reduction. However, while in LDA, every class had the same covariance matrix, and was also assumed to be in a Gaussian distribution, in QDA, each class can have its own separate Covariance matrix, and the condition that the features have to be in a normal distribution is relaxed. Hence, QDA is able to recognize complex relationships between the features, and can form better decision boundaries as compared to LDA.

3 Dataset and Data Preprocessing

Over the years of research conducted, the Pima Indian Diabetes database has remained a benchmark in the training and validation of machine learning algorithms in the field of diabetes prediction. This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases.

The dataset contains 768 entries, and 9 columns, with the last column being the outcome of whether the person got diabetes or not. Due to the presence of the outcome column, we can use supervised learning algorithms. Each record contains 8 features that could be indicative of risk for diabetes - Number of Pregnancies, Plasma Glucose concentration, Diastolic blood pressure, skin thickness, insulin level, Body Mass Index, diabetes pedigree function and Age. Out of the 768 rows, 500 of them are non-diabetics while the other 268 are diabetics. The outcome variable is 0 for non-diabetics and 1 for diabetics.

The dataset has no null values or missing values, but it does contain many inconsistent values. For the features of Insulin, BMI, Glucose, Blood Pressure and Skin Thickness, there are instances which take 0 values, which in reality cannot be possible and can lead to inaccurate training.

Due to the low number of training instances, it would be betteer if rows were not removed and were instead standardized. In order to process our data, we impute the median value on the features that had invalid zero values. as median is less sensitive to outliers than mean

Using boxplots on the data, we can observe that there are many outliers present in the data. In order to remove these outliers, we use the help of two outlier detection methods - IQR and Z-score.

We perform the experiments on three datasets : a dataset that has median values imputed, a dataset that has outlier removal using IQR and finally, a dataset that uses outlier removal using Z-score.

Column	Range	Dtype
Pregnancies	[0, 17]	int64
Glucose	[0, 199]	int64
BloodPressure	[0, 122	int64
SkinThickness	[0, 99]	int64
Insulin	[0, 846]	int64
BMI	[0, 67.1]	float64
DiabetesPedigreeFunction	[0.078, 2.42]	float64
Age	[21, 81]	int64
Outcome	[0,1]	int64

Table 1: Description of Columns

4 Experiments

The dataset was split into two subsets for training and validation in a 80/20 split.The same training and test sets are used for all models in order to keep the environment same for better comparison. Finally, the metrics used to compare between the models are accuracy, precision and F-score.

Accuracy is the percentage of all samples that have been estimated correctly. It refers to the ratio of the sum of true positives and true negatives to the total number of predictions made.

Precision refers to the percentage of all samples that have been correctly predicted as true among all those which were predicted as true, even if they were false.

The standard F-score (F1-score) is an indicator of a binary classification model's accuracy, calculated by the weighted average of the precision and sensitivity. To be specific, it is calculated by dividing the product of the precision and sensitivity by the sum of the precision and sensitivity and multiplying the result by two.

4.0.1 AdaBoost Classifier

AdaBoost is an ensemble method.AdaBoost combines multiple weak learners to create a strong model.AdaBoost gives more weight to misclassified points in each iteration. If there are outliers that are initially misclassified, AdaBoost may allocate a significant amount of attention to them, leading to an overemphasis on the noisy data. AdaBoost being sensitive to outliers Gives better results after cleaning the dataset the accuracies before outlier detection are 84% and 87.6% with IQR and 86.3% with Z-score

	Actual Positive	Actual Negative
Predicted Positive	84	15
Predicted Negative	9	46

Table 2: AdaBoost Confusion Matrix for dataset without outlier removal

	Actual Positive	Actual Negative
Predicted Positive	90	9
Predicted Negative	10	45

Table 3: AdaBoost Confusion Matrix for dataset using IQR

	Actual Positive	Actual Negative
Predicted Positive	88	11
Predicted Negative	10	45

Table 4: AdaBoost Confusion Matrix for dataset using Z-score

4.0.2 XGBoost Classifier

XGboost is less sensitive to noise and outliers.XGBoost incorporates regularization terms in its objective function.These regularization terms penalize overly complex models and help prevent the algorithm from fitting the noise in the data.the ensemble method accuracy decreased after data cleaning might be due to potential loss of information

	Actual Positive	Actual Negative
Predicted Positive	88	11
Predicted Negative	9	46

Table 5: XGBoost Confusion Matrix for dataset without outlier removal

	Actual Positive	Actual Negative
Predicted Positive	85	14
Predicted Negative	10	45

Table 6: XGBoost Confusion Matrix for dataset using IQR

	Actual Positive	Actual Negative
Predicted Positive	88	11
Predicted Negative	13	42

Table 7: XGBoost Confusion Matrix for dataset using Z-score

4.0.3 Linear Discriminant Analysis

In order to find the best parameters for the model, we perform grid search over different solvers such as SVD, eigenvalue decomposition and least squares method. Finally, we obtain the confusion shown in Table 12 for data without anomaly removal, Table 13 for data with IQR method used, and Table 14 for Z-score.

	Actual Positive	Actual Negative
Predicted Positive	85	14
Predicted Negative	19	36

Table 8: LDA Confusion Matrix for dataset without outlier removal

	Actual Positive	Actual Negative
Predicted Positive	79	20
Predicted Negative	8	47

Table 9: LDA Confusion Matrix for dataset using IQR

	Actual Positive	Actual Negative
Predicted Positive	81	18
Predicted Negative	17	38

Table 10: LDA Confusion Matrix for dataset using Z-score

4.0.4 Quadratic Discriminant Analysis

Quadratic Discriminant analysis on training and testing with the three different datasets give us the confusion matrix shown in Table 15. In order to find the best hyperparameters, we also use grid search over different regularization parameters and whether or not we store the covariance. Table 16 gives us the confusion matrix of the model with IQR and Table 17 for Z-score.

	Actual Positive	Actual Negative
Predicted Positive	85	14
Predicted Negative	9	46

Table 11: QDA Confusion Matrix for dataset without outlier removal

	Actual Positive	Actual Negative
Predicted Positive	80	19
Predicted Negative	5	50

Table 12: QDA Confusion Matrix for dataset using IQR

	Actual Positive	Actual Negative
Predicted Positive	83	16
Predicted Negative	7	48

Table 13: QDA Confusion Matrix for dataset using Z-score

4.0.5 Logistic Regression

Logistic regression assumes a linear relationship between the input features and the log-odds of the binary outcome. However, in the context of the Pima Indian Diabetes dataset, the relationships between the health-related features and the likelihood of diabetes onset may not be strictly linear. hence giving limited accuracies 77.27% without outlier detection 81.81% with IQR ,77.92% with Z-score

	Actual Positive	Actual Negative
Predicted Positive	83	16
Predicted Negative	19	36

Table 14: Logistic Confusion Matrix for dataset without outlier removal

	Actual Positive	Actual Negative
Predicted Positive	79	20
Predicted Negative	8	47

Table 15: Logistic Confusion Matrix for dataset using IQR

	Actual Positive	Actual Negative
Predicted Positive	82	17
Predicted Negative	17	38

Table 16: Logistic Confusion Matrix for dataset using Z-score

4.0.6 SVM Classifier

The SVM when implemented with linear Kernel is not able to find the dependencies of features .The model being simple linear model was not able to find a good hyperplane for the diabetes dataset which is not linear. The accuracies **before outlier detection are 81.2%** and after outlier detection with **IQR gave 80.1%** and with **Z-score gave 80.92%**

	Actual Positive	Actual Negative
Predicted Positive	88	11
Predicted Negative	18	37

Table 17: SVM Confusion Matrix for dataset using IQR

	Actual Positive	Actual Negative
Predicted Positive	85	14
Predicted Negative	13	42

Table 18: SVM Confusion Matrix for dataset using Z-score

4.0.7 SMOTE based LSTM

The features are passed as time steps in the SMOTE based LSTM as it tries to find the correlation and temporal dependencies between features. This gives better results after outlier removal as its able to find better dependencies in a cleaned dataset. With accuracies, it gave an accuracy of 83% without outlier detyection and 86.5% with IQR outlier detection and 88% with z-score

	Actual Positive	Actual Negative
Predicted Positive	81	18
Predicted Negative	14	87

Table 19: LSTM Confusion Matrix for dataset without outlier removal

	Actual Positive	Actual Negative
Predicted Positive	86	13
Predicted Negative	11	90

Table 20: LSTM Confusion Matrix for dataset using IQR

	Actual Positive	Actual Negative
Predicted Positive	85	14
Predicted Negative	11	90

Table 21: LSTM Confusion Matrix for dataset using Z-score

4.0.8 Random Forest Classifier

Random Forest builds multiple decision trees independently and averages their predictions Random Forest introduces additional randomness by considering only a subset of features at each split. This feature randomization reduces the chance that a single noisy feature dominates the decision-making process. hence random forest are less sensitive to noise gave approximately same results on oulier detection

	Actual Positive	Actual Negative
Predicted Positive	85	14
Predicted Negative	12	43

Table 22: Random Forest Confusion Matrix for dataset without outlier removal

	Actual Positive	Actual Negative
Predicted Positive	88	11
Predicted Negative	9	46

Table 23: Random Forest Confusion Matrix for dataset using IQR

	Actual Positive	Actual Negative
Predicted Positive	87	12
Predicted Negative	10	45

Table 24: Random Forest Confusion Matrix for dataset using Z-score

4.0.9 Decision Tree Classifier

In order to find the best parametersm we used grid search. Decision tree being sensitive to outliers especially when determining split points gave better results after data cleaning.
The accuracy before outlier detection was 83.7% and after outlier detection are **86.3% with IQR and 85.7% with Z-score respectively**

	Actual Positive	Actual Negative
Predicted Positive	89	10
Predicted Negative	9	46

Table 25: Decision Tree Confusion Matrix for dataset using IQR

	Actual Positive	Actual Negative
Predicted Positive	87	12
Predicted Negative	11	44

Table 26: Decision Tree Confusion Matrix for dataset using Z-score

4.0.10 k-Nearest Neighbours Classifier

K-NN, being data sensitive, gives varying results with different outlier techniques.

	Actual Positive	Actual Negative
Predicted Positive	90	9
Predicted Negative	15	40

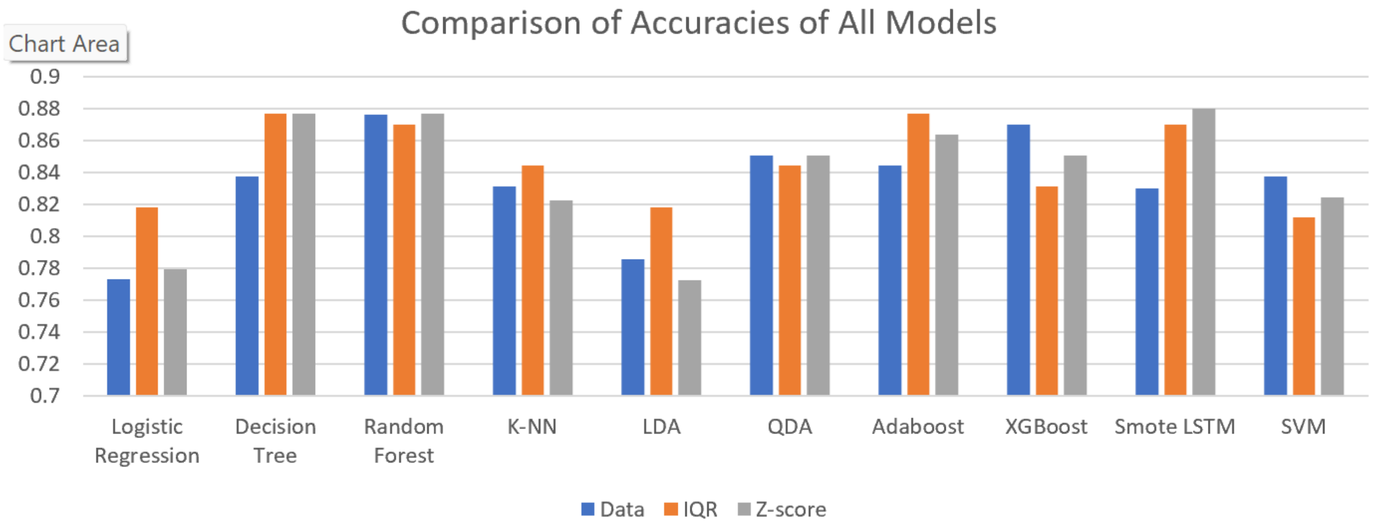
Table 27: k-NN Confusion Matrix for dataset using IQR

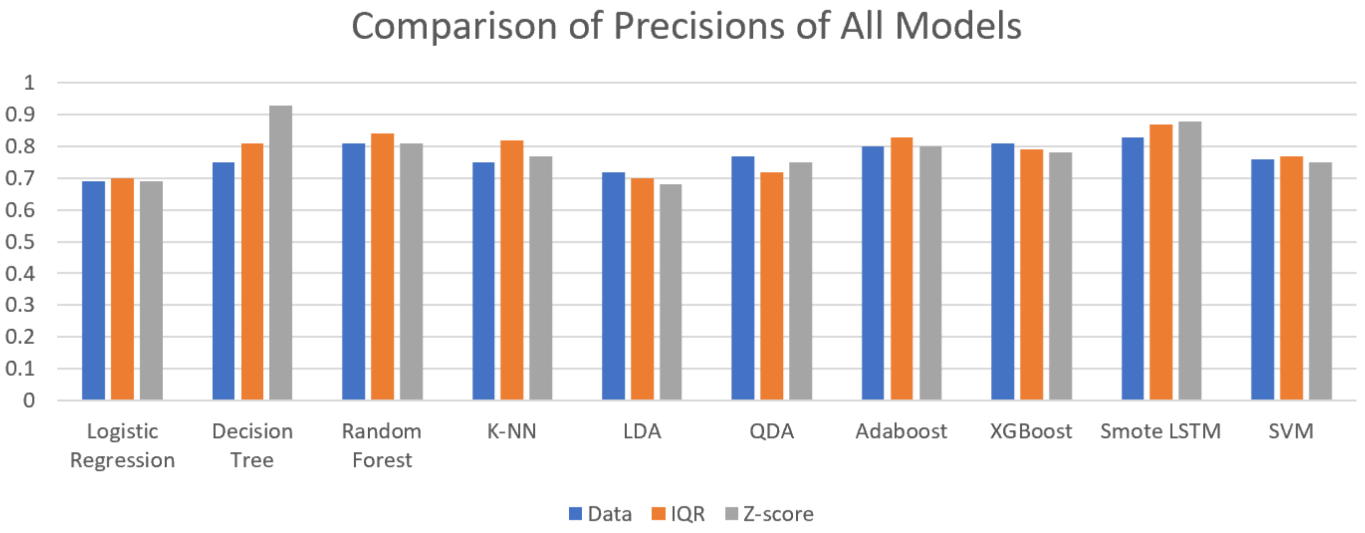
	Actual Positive	Actual Negative
Predicted Positive	87	12
Predicted Negative	15	40

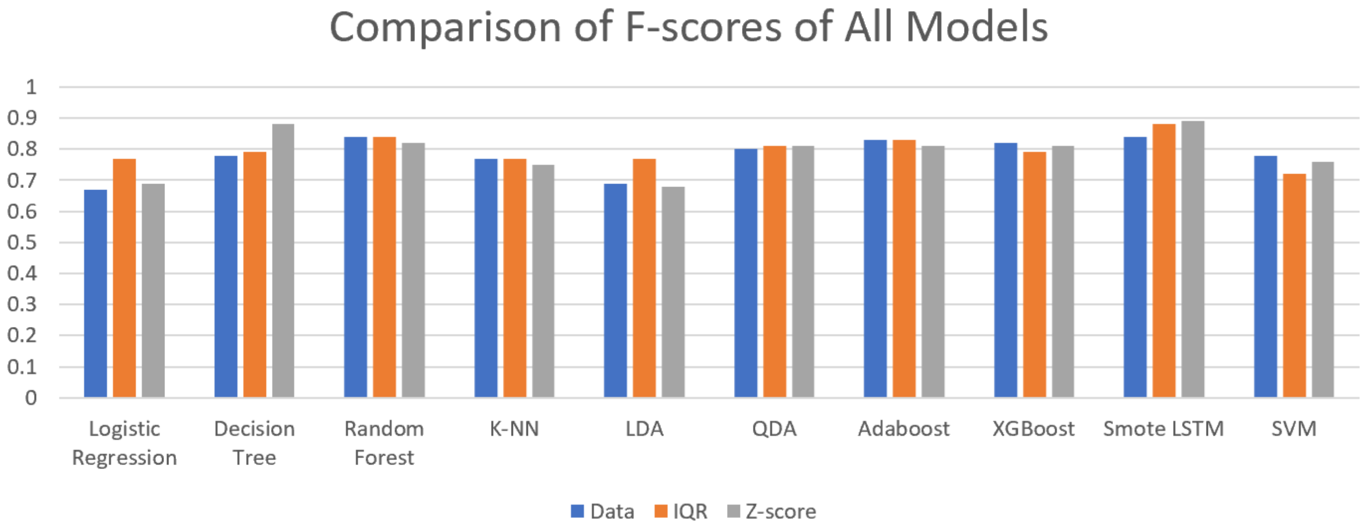
Table 28: k-NN Confusion Matrix for dataset using Z-score

5 Results

The charts represent the accuracy, precision and F-score of the different models. We can see that in accuracy, the SMOTE-based deep LSTM outperforms all the other models. The random forest classifier is not far behind, and is comparable.







6 References

https://machinelearningmastery.com/how-to-use-statistics-to-identify-outliers-in-data/

https://thesai.org/Publications/ViewPaper?Volume=12&Issue=4&Code=IJACSA&SerialNo=66

https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database