**INDUSTRIAL TRAINING REPORT**

**ON PROJECT**

**“DIABETES DETECTION USING MACHINE LEARNING ALGORITHMS”**

**AT EXPOSYS DATA LABS**

**Submitted By:**

**Pathipati Venkatesh**

**11189A182- CSE**



###### Department OF COMPUTER SCIENCE & ENGineering

SRI CHANDRASEKHARENDRA SARASWATHI VISWA MAHAVIDYALAYA

**Abstract**

Diabetes is an illness caused because of high glucose levels in the human body. Diabetes should not be ignored if it is untreated then Diabetes may cause some major issues in a person like heart-related problems, kidney problems, blood pressure, eye damage and it can also affect other organs of the human body. Diabetes can be controlled if it is predicted earlier.  To achieve this goal this project work will do early prediction of Diabetes in a human body or a patient for higher accuracy through applying, Various Machine Learning Techniques. Machine learning techniques Provide better results for prediction by constructing models from datasets collected from patients. In this work, Machine Learning Classification and ensemble techniques are used on a dataset to predict diabetes. Which are K-Nearest Neighbor (KNN), Logistic Regression (LR), Decision Tree (DT), Extreme Gradient Boosting, Support Vector Classifier (SVC), Gaussian Naïve Bayes, and Random Forest (RF). The accuracy is different for every model when compared to other models. The Project work gives the accurate or higher accuracy model shows that the model is capable of predicting diabetes effectively. The result shows that Logistic Regression achieved higher accuracy compared to other machine learning techniques.

**Contents**

1. **Introduction 4**
2. **Existing Method 5**
3. **Proposed method with Architecture 7**
4. **Methodology 10**
5. **Implementation 18**
6. **Conclusion 24**

**Introduction**

Diabetes is a noxious disease in the world. Diabetes is caused because of obesity or high blood glucose level, and so forth. It affects the hormone insulin, resulting in the abnormal metabolism of crabs and improves the level of sugar in the blood. Diabetes occurs when the body does not make enough insulin. According to the World Health Organization (WHO), about 422 million people are suffering from diabetes particularly from low or idle-income countries. And this could be increased to 490 billion up to the year 2030. However, the prevalence of diabetes is found among various Countries like Canada, China, and India, etc. The population of India is now more than 100 million so the actual number of diabetics in India is 40 million. Diabetes is the major cause of death in the world. Early prediction of diabetes can be controlled and save human life. To accomplish this, this work explores the prediction of diabetes by taking various attributes related to diabetes disease. For this purpose, the Pima Indian Diabetes Dataset is used and various Machine Learning classification and ensemble Techniques are applied to predict diabetes. Machine Learning is a method that is used to train computers or machines explicitly. Various Machine Learning Techniques provide efficient results to collect Knowledge by building various classification and ensemble models from the collected dataset. Such collected data can be useful to predict diabetes. Various techniques of Machine Learning are capable of prediction, however, it is tough to choose the best technique. Thus for this purpose popular classification and ensemble methods are applied to the dataset for prediction.

**Existing Method**

K. VijayaKumar et al. proposed a Random Forest algorithm for the Prediction of diabetes develops a system that can perform early prediction of diabetes for a patient with higher accuracy by using the Random Forest algorithm in machine learning technique. The proposed model gives the best results for diabetic prediction and the result showed that the prediction system is capable of predicting the diabetes disease effectively, efficiently, and most importantly, instantly.

Nonso Nnamoko et al. presented predicting diabetes onset: an ensemble supervised learning approach they used five widely used classifiers are employed for the ensembles and a meta-classifier is used to aggregate their outputs. The results are presented and compared with similar studies that used the same dataset within the literature. It is shown that by using the proposed method, diabetes onset prediction can be done with higher accuracy.

Tejas N. Joshi et al. presented Diabetes Prediction Using Machine Learning Techniques aimed to predict diabetes via three different supervised machine learning methods including SVM, Logistic regression, ANN. This project proposes an effective technique for the earlier detection of diabetes disease.

Dheeraj Shetty et al. proposed diabetes disease prediction using data mining assemble Intelligent Diabetes Disease Prediction System that gives an analysis of diabetes malady utilizing diabetes patient’s database. In this system, they propose the use of algorithms like Bayesian and KNN (K-Nearest Neighbor) to apply to diabetes patient’s databases and analyze them by taking various attributes of diabetes for the prediction of diabetes disease.

Muhammad Azeem Sarwar et al. proposed a study on the prediction of diabetes using machine learning algorithms in healthcare they applied six different machine learning algorithms Performance and accuracy of the applied algorithms are discussed and compared. A comparison of the different machine learning techniques used in this study reveals which algorithm is best suited for the prediction of diabetes. Diabetes Prediction is becoming the area of interest for researchers to train the program to identify if the patient is diabetic or not by applying a proper classifier on the dataset. Based on previous research work, it has been observed that the classification process is not much improved. Hence a system is required as Diabetes Prediction is an important area in computers, to handle the issues identified based on previous research.

**The proposed method with Architecture**

The proposed method uses various Machine Learning Algorithms for predicting diabetes. The following phases are present in the architecture of the proposed method.

1. Dataset Collection
2. Data Pre-processing
3. Clustering
4. Model Building
5. Evaluation

Pi

**Pre-processing Data**

**Clustering**

**Build Model**

**Dataset**

Training Data



Testing Data



**Evaluation**

**Diabetes Prediction Model**

**Dataset Collection**

This phase includes data collection and understanding the data to study the patterns and trends which helps in predicting and evaluating the results. Dataset description is given below

The dataset used in this project is Pima Indians Diabetes Dataset and it contains 769 records and 9 attributes.

**Dataset Information**

|  |  |
| --- | --- |
| **Attribute** | **Type** |
| Pregnancies | N |
| Glucose | N |
| blood pressure | N |
| skin thickness | N |
| Insulin | N |
| BMI | N |
| Diabetes Pedigree Function | N |
| Age | N |
| Outcome | C |

**Data Pre-processing**

This phase of the model handles inconsistent data to get more accurate and precise results. This dataset contains missing values. The missing values are imputed for a few selected attributes like Glucose level, Blood Pressure, Skin Thickness, BMI, and Age because these attributes cannot have values zero. Then the dataset is scaled to normalize all values.

**Clustering**

In this phase, K-means clustering is implemented on the dataset to classify each patient into either a diabetic or non-diabetic class. Before performing K-means clustering, highly correlated attributes were found which were, Glucose and Age. K-means clustering was performed on these two attributes. After implementation of this clustering class labels (0 or 1) for each record was obtained.

**Model Building**

This is the most important phase which includes model building for the prediction of diabetes. For this purpose various machine learning algorithms for diabetes prediction. The algorithms include Logistic Regression, K-Nearest Neighbor, Gaussian Naïve Bayes, Support Vector Classifier, Decision Tree, XGBoost, and Random Forest.

**Evaluation**

This is the final step of the prediction model. Here, the prediction results are evaluated using classification accuracy.

**Methodology**

For classifying each patient into either a diabetic or non-diabetic class K-means clustering is used.

**K-means Clustering**

K-Means Clustering is an unsupervised learning algorithm that is used to solve clustering problems in machine learning or data science. It groups the unlabeled dataset into different clusters. Here K defines the number of predefined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on.

Algorithm-

* Select the number K to decide the number of clusters.
* Select random K points or centroids. (It can be different from the input dataset).
* Assign each data point to their closest centroid, which will form the predefined K clusters.
* Calculate the variance and place a new centroid of each cluster.
* Repeat the third step, which means reassign each data point to the new closest centroid of each cluster.
* If any reassignment occurs, then go to step 4 else go to FINISH.
* The model is ready.

For predicting diabetes the following machine learning algorithms were implemented.

**Support Vector Classification**

SVC is a nonparametric clustering algorithm that does not make any assumption on the number or shape of the clusters in the data. In our experience, it works best for low-dimensional data, so if your data is high-dimensional, a preprocessing step, e.g. using principal component analysis, is usually required.

Algorithm-

* Select the hyperplane which divides the class better.
* To find a better hyperplane you have to calculate the distance between the planes and the data which is called Margin.
* If the distance between the classes is low then the chance of miss conception is high and vice versa. So we need to
* Select the class which has the highest margin. Margin = distance to positive point + Distance to a negative point.

**K-Nearest Neighbor**

KNN is also a supervised machine learning algorithm. KNN helps to solve both the classification and regression problems. KNN is a lazy prediction technique. KNN assumes that similar things are near to each other. Many times data points that are similar are very near to each other. KNN helps to group new work based on similarity measures. KNN algorithm records all the records and classifies them according to their similarity measure. Finding the distance between the points uses a tree-like structure. To predict a new data point, the algorithm finds the closest data points in the training data set its nearest neighbors. Here K= Number of nearby neighbors, it’s always a positive integer. Neighbor’s value is chosen from the set of classes. Closeness is mainly defined in terms of Euclidean distance. The Euclidean distance between two points P and Q i.e. P (p1,p2,…,pn) and Q (q1, q2,...,qn) is defined by the following equation:-

…….+

Algorithm-

* + Take a sample dataset of columns and rows named Pima Indian Diabetes data set.
  + Take a test dataset of attributes and rows.
  + Find the Euclidean distance with the help of formula
  + Then, Decide a random value of K. is the no. of nearest neighbors
  + Then with the help of these minimum distance and Euclidean distance find out the nth column of each.
  + Find out the same output values.
  + If the values are the same, then the patient is diabetic, otherwise not.

**Decision Tree**

A decision tree is a basic classification method. It is a supervised learning method. A decision tree is used when the response variable is categorical. The decision tree has a tree-like structure-based model which describes the classification process based on input features. Input variables are any types like graph, text, discrete, continuous, etc. Steps for Decision Tree

Algorithm-

* + Construct a tree with nodes as input features.
  + Select feature to predict the output from input feature whose information gain is highest.
  + The highest information gain is calculated for each attribute in each node of the tree.
  + Repeat step 2 to form a subtree using the feature which is not used in the above node.

**Logistic Regression**

Logistic regression is also a supervised learning classification algorithm. It is used to estimate the probability of a binary response based on one or more predictors. They can be continuous or discrete. Logistic regression is used when we want to classify or distinguish some data items into categories.

It classifies the data in binary form only in 0 and 1 which refer to a case to classify a patient that is positive or negative for diabetes.

The main aim of logistic regression is to best fit which is responsible for describing the relationship between target and predictor variables. Logistic regression is based on a Linear regression model. The logistic regression model uses a sigmoid function to predict the probability of positive and negative classes.

Sigmoid function P = 1/1+ Here P is probability and a, b are parameters of the Model.

**Random Forest**

It is a type of ensemble learning method and is also used for classification and regression tasks. The accuracy it gives is greater than compared to other models. This method can easily handle large datasets. Random Forest was developed by Leo Bremen. It is a popular ensemble Learning Method. Random Forest Improves Performance of Decision Tree by reducing variance. It operates by constructing a multitude of decision trees at training time and outputs the class that is the mode of the classes or classification or mean prediction (regression) of the individual trees.

Algorithm-

* + The first step is to select the R features from the total features m where R<<M.
  + Among the R features, the node uses the best split point.
  + Split the node into sub-nodes using the best split.
  + Repeat 1 to 3 steps until the number of nodes has been reached.
  + Build a forest by repeating steps 1 to 4 several times to create several trees.

The random forest finds the best split using the Gin-Index Cost Function which is given by:

The first step is to take a glance at choices and use the foundations of each indiscriminately created decision tree to predict the result and store the anticipated outcome at intervals at the target place. Secondly, calculate the votes for each predicted target and ultimately, admit the high voted predicted target as a result of the ultimate prediction from the random forest formula. Some of the options of Random Forest do correct predictions results for a spread of applications are offered.

**Gaussian Naïve Bayes**

Naive Bayes classifiers are a collection of classification algorithms based on *Bayes’ Theorem****.*** It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other.

When working with continuous data, an assumption often taken is that the continuous values associated with each class are distributed according to a normal (or Gaussian) distribution. The likelihood of the features is assumed to be-

Sometimes assume variance

* is independent of Y (i.e., ),
* or independent of Xi (i.e., ),
* or both (i.e., σ)

**Gradient Boosting**

Gradient Boosting is the most powerful ensemble technique used for prediction and it is a classification technique. It combines weak learners to make strong learner models for prediction. It uses the Decision Tree model. It classifies complex data sets and it is a very effective and popular method. In the gradient boosting model, performance improves over iterations.

Algorithm-

* + Consider a sample of target values as P
  + Estimate the error in target values.
  + Update and adjust the weights to reduce error M.
  + P[x] =P[x] +alpha M[x]
  + Model Learners are analyzed and calculated by loss function F
  + Repeat steps till desired & target result P.

For evaluating the prediction results classification accuracy evaluation metric is used.

**Classification Accuracy**

It is the ratio of the number of correct predictions to the total number of input samples. It is given as-

**Confusion Matrix**

It gives a matrix as output and describes the complete performance of the model.

Where, TP- True Positive

FP- False Positive

FN- False Negative

FN- false Negative

TN- True Negative

|  |  |
| --- | --- |
|  | Actual Values |
| Predicted Values |  |

Accuracy for the matrix can be calculated by taking the average of the values lying across the main diagonal. It is given as-

Where N is the total number of samples

N = TP + FP + FN + TN

**Implementation**

The proposed method is implemented using PyCharm and Python 3.6 interpreters.

* Importing required libraries and diabetes dataset.

The dataset used is stored in a comma-separated values (.csv) file.

**import** matplotlib.pyplot **as** plt  
**import** pandas **as** pd  
**import** seaborn **as** sns  
  
**from** warnings **import** filterwarnings  
filterwarnings(action=**'ignore'**)  
data = pd.read\_csv(**"diabetes.csv"**)  
print(**"Successfully Imported Data!"**)  
print(data.head())  
print(data.shape)

* Pre-process data to remove missing data.

print(data.isna().sum())  
  
print(data.corr())  
  
print(data.groupby(**'Age'**).mean())  
  
print(data[**'Outcome'**].value\_counts())

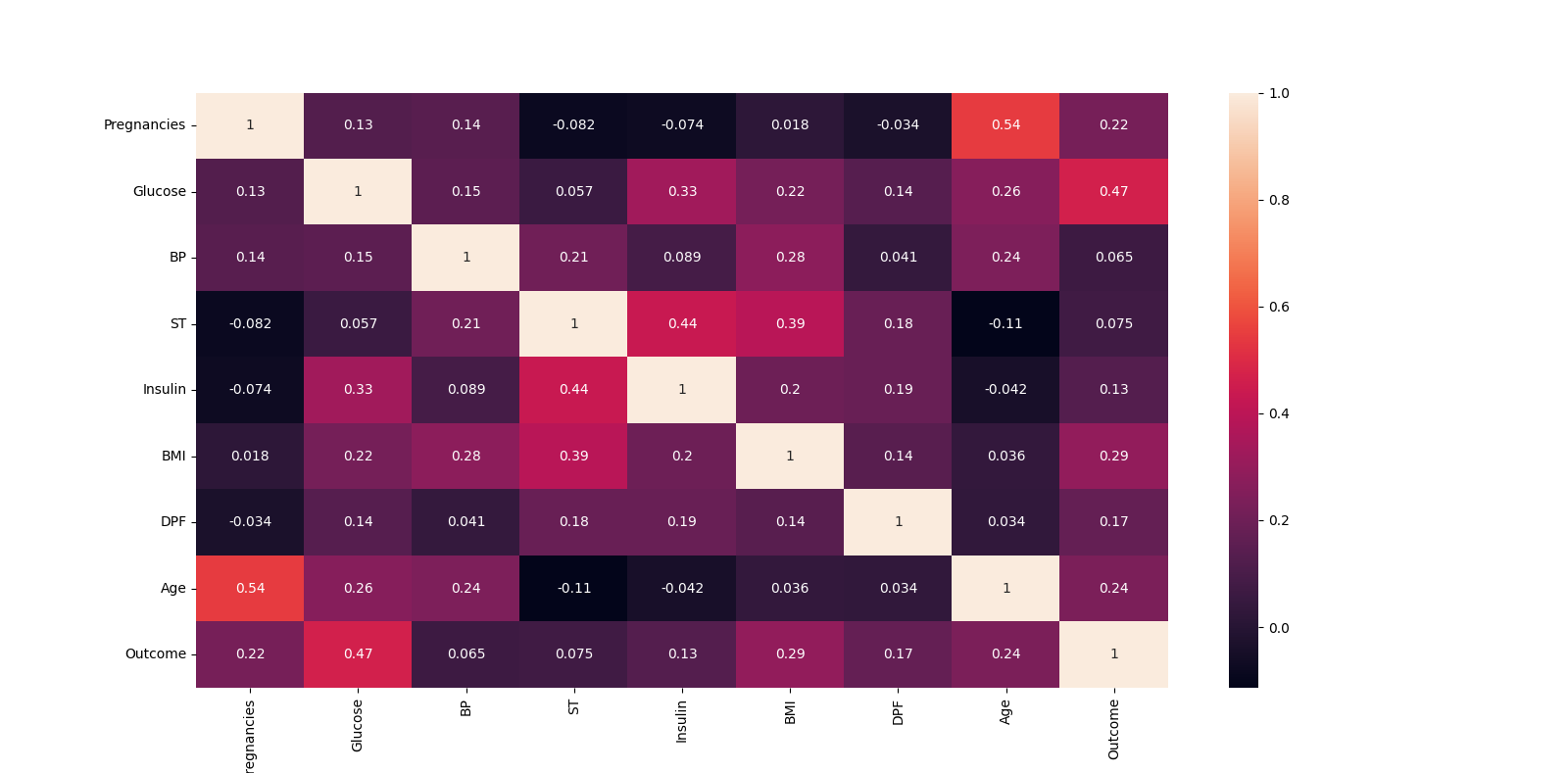
* Analysis of data using various plots

*# Distplot:*

sns.distplot(data[**'Outcome'**])  
plt.show()  
sns.distplot(data[**'BMI'**])  
plt.show()  
sns.distplot(data[**'Pregnancies'**])  
plt.show()  
sns.distplot(data[**'Age'**])  
plt.show()  
sns.distplot(data[**'BP'**])  
plt.show()  
sns.distplot(data[**'ST'**])  
plt.show()  
data.plot(kind =**'box'**,subplots = **True**, layout =(4,4),sharex = **False**)  
  
data.plot(kind =**'density'**,subplots = **True**, layout =(4,4),sharex = **False**)  
  
*# Histogram*data.hist(figsize=(10,10),bins=50)  
plt.show()

*# Heatmap for expressing the correlation*corr = data.corr()  
sns.heatmap(corr,annot=**True**)  
plt.show()

**Heatmap for Expressing Correlation**



*# Box plot for outlier visualization*sns.set(style=**"whitegrid"**)  
data.boxplot(figsize=(15,6))  
plt.show()

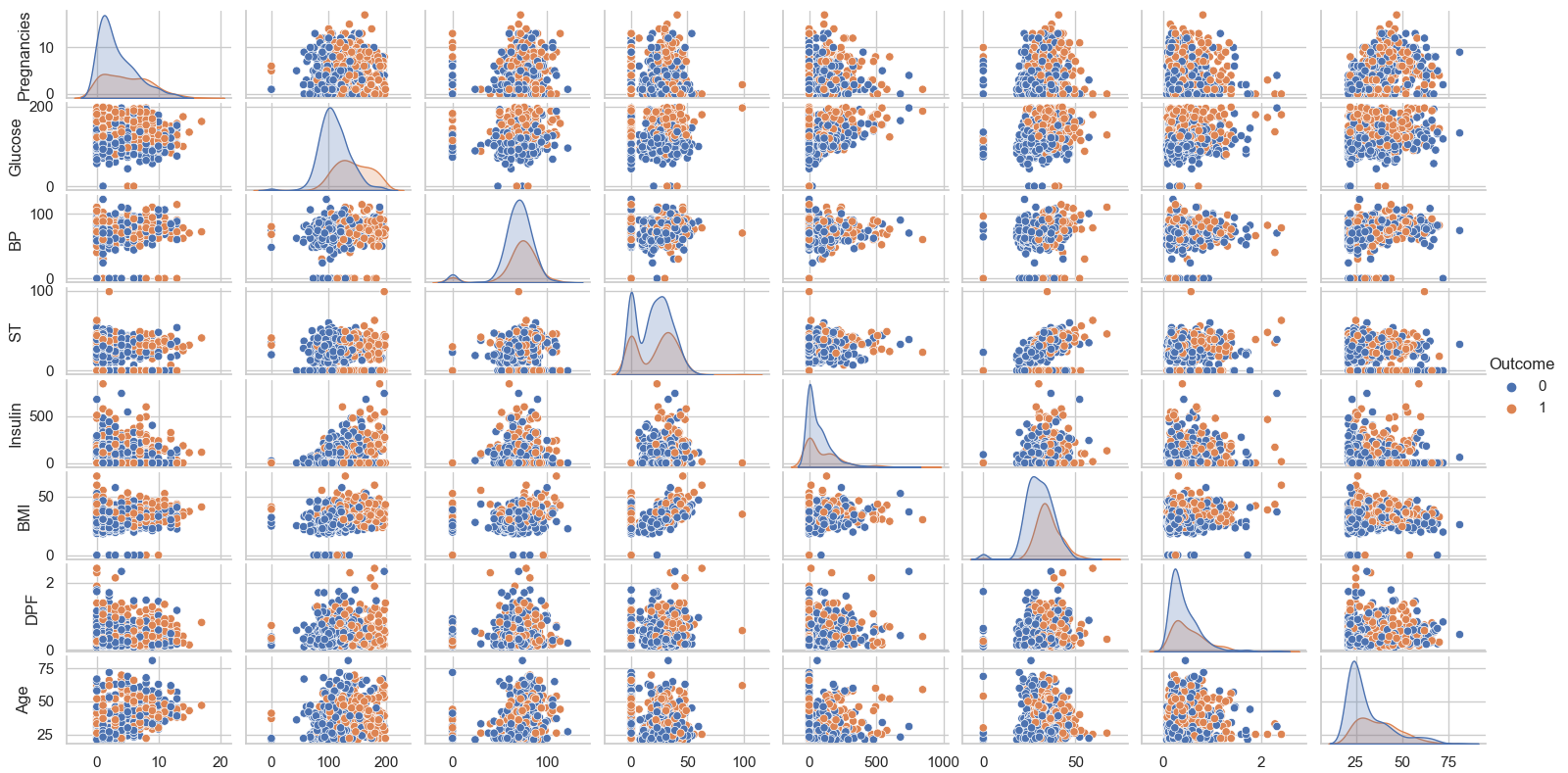
*# Pairplot:*sns.pairplot(data)  
plt.show()

*# Violinplot:*sns.violinplot(x=**'Outcome'**, y=**'Age'**, data=data)  
plt.show()

*# sns.violinplot(x='Outcome', y='BloodPressure', data=data)*sns.violinplot(x=**'Outcome'**, y=**'ST'**, data=data)  
plt.show()

*# Pairplot:*sns.pairplot(data,hue=**'Outcome'**);  
plt.show()

**Pair plot of the data and Outcome**



* Select the machine learning algorithm i.e. K- Nearest Neighbor, Support Vector Classification, Decision Tree, Logistic regression, Random Forest and Gradient boosting algorithm, Gaussian NB.
* Build the classifier model for the mentioned machine learning algorithm based on the training set.

*# Using Logistic Regression***from** sklearn.linear\_model **import** LogisticRegression  
model = LogisticRegression()  
model.fit(X\_train,Y\_train)  
Y\_pred = model.predict(X\_test)  
  
**from** sklearn.metrics **import** accuracy\_score,confusion\_matrix  
print(**"--Logistic Regression--"**)  
print(**"Accuracy Score:"**,accuracy\_score(Y\_test,Y\_pred))  
  
confusion\_mat = confusion\_matrix(Y\_test,Y\_pred)  
print(confusion\_mat)

*# Using KNN***from** sklearn.neighbors **import** KNeighborsClassifier  
model = KNeighborsClassifier(n\_neighbors=3)  
model.fit(X\_train,Y\_train)  
y\_pred = model.predict(X\_test)  
  
**from** sklearn.metrics **import** accuracy\_score  
print(**"--KNN--"**)  
print(**"Accuracy Score:"**,accuracy\_score(Y\_test,y\_pred))

*# Using SVC***from** sklearn.svm **import** SVC  
model = SVC()  
model.fit(X\_train,Y\_train)  
pred\_y = model.predict(X\_test)  
  
**from** sklearn.metrics **import** accuracy\_score  
print(**"--SVC--"**)  
print(**"Accuracy Score:"**,accuracy\_score(Y\_test,pred\_y))

*# Using Decision Tree***from** sklearn.tree **import** DecisionTreeClassifier  
model = DecisionTreeClassifier(criterion=**'entropy'**,random\_state=7)  
model.fit(X\_train,Y\_train)  
y\_pred1 = model.predict(X\_test)  
  
**from** sklearn.metrics **import** accuracy\_score  
print(**"--Decision Tree--"**)  
print(**"Accuracy Score:"**,accuracy\_score(Y\_test,y\_pred1))

*# Using GaussianNB***from** sklearn.naive\_bayes **import** GaussianNB  
model3 = GaussianNB()  
model 3.fit(X\_train,Y\_train)  
y\_pred3 = model 3.predict(X\_test)  
  
**from** sklearn.metrics **import** accuracy\_score  
print(**"--GaussianNB--"**)  
print(**"Accuracy Score:"**,accuracy\_score(Y\_test,y\_pred3))

*# Random Forest***from** sklearn.ensemble **import** RandomForestClassifier  
model2 = RandomForestClassifier(random\_state=1)  
model2.fit(X\_train, Y\_train)  
y\_pred2 = model2.predict(X\_test)  
  
**from** sklearn.metrics **import** accuracy\_score  
print(**"--Random Forest--"**)  
print(**"Accuracy Score:"**,accuracy\_score(Y\_test,y\_pred2))

*# Using Xgboost***import** xgboost **as** xgb  
model5 = xgb.XGBClassifier(random\_state=1,eval\_metric=**'mlogloss'**)  
model5.fit(X\_train, Y\_train)  
y\_pred5 = model 5.predict(X\_test)  
  
**from** sklearn.metrics **import** accuracy\_score  
print(**"--Xgboost--"**)  
print(**"Accuracy Score:"**,accuracy\_score(Y\_test,y\_pred5))

* Test the Classifier model for the mentioned machine learning algorithm based on the test set.
* Perform Comparison Evaluation of the experimental performance results obtained for each classifier.

*# Results*results = pd.DataFrame({  
 **'Model'**: [**'Logistic Regression'**,**'KNN'**, **'SVC'**,**'Decision Tree'** ,**'GaussianNB'**,**'Random Forest'**,**'Xgboost'**],  
 **'Score'**: [accuracy\_score(Y\_test,Y\_pred),accuracy\_score(Y\_test,y\_pred),accuracy\_score(Y\_test,pred\_y),accuracy\_score(Y\_test,y\_pred1),accuracy\_score(Y\_test,y\_pred3),accuracy\_score(Y\_test,y\_pred2),accuracy\_score(Y\_test,y\_pred5)]})  
  
result\_df = results.sort\_values(by=**'Score'**, ascending=**False**)  
result\_df = result\_df.set\_index(**'Score'**)  
print(result\_df)

Confusion Matrix for Logistic Regression

TP=97, FP=11, FN=20 and TN=26.

Accuracy =

The accuracy score is calculated using the formula by substituting the values obtained from a confusion matrix

|  |  |
| --- | --- |
| **Score** | **Model** |
| 0.779 | Logistic Regression |
| 0.753 | Random Forest |
| 0.740 | Xgboost |
| 0.733 | SVC |
| 0.733 | GaussianNB |
| 0.720 | KNN |
| 0.681 | Decision Tree |

**Accuracy Scores**

* After analyzing based on various measures conclude the best performing algorithm.

Here we can observe that the Logistic Regression algorithm has the highest accuracy when compared to other machine learning algorithms. So the Logistic Regression algorithm is used for training the model.

**Conclusion**

The main aim of this project was to design and implement Diabetes Prediction Using Machine Learning Methods and Performance Analysis of those methods and it has been achieved successfully. The proposed approach uses various classification and ensemble learning methods in which SVC, KNN, Random Forest, Decision Tree, Gaussian NB, Logistic Regression, and Gradient Boosting classifiers are used. And 77% classification accuracy has been achieved. The Experimental results will assist health care to take early predictions and make early decisions to cure diabetes and save humans life.