DIABETES PREDICTION REPORT

EXPOSYS DATA LABS INTERNSHIP

20– 05 – 2021 TO 20– 06 – 2021

Submitted By:

1. Viswanath Allam
(allam_ug@ee.nits.ac.in)
2. Pushpa Latha Devi Yelusuri.
(yelusuri_ug@ece.nits.ac.in)
NIT Silchar.

Under mentorship of

Vishnuvardhan Y
(Founder at Exposys Data Labs)

CONTENTS:

1. INTRODUCTION TO DIABETES:

Diabetes is a disease that occurs when your blood glucose, also called blood sugar, is too high. Blood glucose is your main source of energy and comes from the food you eat. Insulin, a hormone made by the pancreas, helps glucose from food get into your cells to be used for energy. Sometimes your body doesn't make enough—or any—insulin or doesn't use insulin well. Glucose then stays in your blood and doesn't reach your cells.

Over time, having too much glucose in your blood can cause health problems. Although diabetes has no cure, you can take steps to manage your diabetes and stay healthy.

Sometimes people call diabetes "a touch of sugar" or "borderline diabetes." These terms suggest that someone doesn't really have diabetes or has a less serious case, but every case of diabetes is serious.

• Types of diabetes

The most common types of diabetes are type 1, type 2, and gestational diabetes.

Type 1 diabetes

If you have type 1 diabetes, your body does not make insulin. Your immune system attacks and destroys the cells in your pancreas that make insulin. Type 1 diabetes is usually diagnosed in children and young adults, although it can appear at any age. People with type 1 diabetes need to take insulin every day to stay alive. Type 1 diabetes (T1D), previously known as juvenile diabetes, is an autoimmune disease that is a form of diabetes in which very little or no insulin is produced by the islets of Langerhans (containing beta cells) in the pancreas. Insulin is a hormone required for the cells to use blood sugar for energy and it helps regulate normal glucose levels in the bloodstream. Before treatment this results in high blood sugar levels in the body. The common symptoms are frequent urination, increased thirst, increased hunger, and weight loss. Additional symptoms may include blurry vision, tiredness, and slow wound healing. Symptoms typically develop over a short period of time, often a matter of weeks.

The cause of type 1 diabetes is unknown, but it is believed to involve a combination of genetic and environmental factors. Risk factors include having a family member with the condition. The underlying mechanism involves an autoimmune destruction of the insulin-producing beta cells in the pancreas. Diabetes is diagnosed by testing the level of sugar or glycated haemoglobin (HbA1C) in the blood. Type 1 diabetes can be distinguished from type 2 by testing for the presence of autoantibodies.

There is no known way to prevent type 1 diabetes. Treatment with insulin is required for survival. Insulin therapy is usually given by injection just under the skin but can also be delivered by an insulin pump. A diabetic diet and exercise are important parts of management. If left untreated, diabetes can cause many complications. Complications of relatively rapid onset include diabetic ketoacidosis and nonketotic hyperosmolar coma. Long-term complications include heart

disease, stroke, kidney failure, foot ulcers and damage to the eyes. Furthermore, since insulin lowers blood sugar levels, complications may arise from low blood sugar if excessive amount of insulin is taken than necessary.

Type 2 diabetes

If you have type 2 diabetes, your body does not make or use insulin well. You can develop type 2 diabetes at any age, even during childhood. However, this type of diabetes occurs most often in middle-aged and older people. It is the most common type of diabetes. It is characterized by insulin resistance, which may be combined with relatively reduced insulin secretion. The defective responsiveness of body tissues to insulin is believed to involve the insulin receptor. However, the specific defects are not known. Diabetes mellitus cases due to a known defect are classified separately. Type 2 diabetes is the most common type of diabetes mellitus. Many people with type 2 diabetes have evidence of prediabetes (impaired fasting glucose and/or impaired glucose tolerance) before meeting the criteria for type 2 diabetes. The progression of prediabetes to overt type 2 diabetes can be slowed or reversed by lifestyle changes or medications that improve insulin sensitivity or reduce the liver's glucose production.

Type 2 diabetes is primarily due to lifestyle factors and genetics. A number of lifestyle factors are known to be important to the development of type 2 diabetes, including obesity (defined by a body mass index of greater than 30), lack of physical activity, poor diet, stress, and urbanization. Excess body fat is associated with 30% of cases in people of Chinese and Japanese descent, 60–80% of cases in those of European and African descent, and 100% of Pima Indians and Pacific Islanders. Even those who are not obese may have a high waist–hip ratio.

Dietary factors such as sugar-sweetened drinks are associated with an increased risk. The type of fats in the diet is also important, with saturated fat and trans fats increasing the risk and polyunsaturated and monounsaturated fat decreasing the risk. Eating white rice excessively may increase the risk of diabetes, especially in Chinese and Japanese people. Lack of physical activity may increase the risk of diabetes in some people.

Adverse childhood experiences (ACEs), including abuse, neglect, and household difficulties, increase the likelihood of type 2 diabetes later in life by 32%, with neglect having the strongest effect

Gestational diabetes

Gestational diabetes develops in some women when they are pregnant. Most of the time, this type of diabetes goes away after the baby is born. However, if you've had gestational diabetes, you have a greater chance of developing type 2 diabetes later in life. Sometimes diabetes diagnosed during pregnancy is actually type 2 diabetes. Gestational diabetes resembles type 2 diabetes in several respects, involving a combination of relatively inadequate insulin secretion and responsiveness. It occurs in about 2–10% of all pregnancies and may improve or disappear after delivery. It is recommended that all pregnant women get tested starting around 24–28 weeks gestation. It is most often diagnosed in the second or third trimester because of the increase in insulin-antagonist

hormone levels that occurs at this time. However, after pregnancy approximately 5–10% of women with gestational diabetes are found to have another form of diabetes, most commonly type 2. Gestational diabetes is fully treatable, but requires careful medical supervision throughout the pregnancy. Management may include dietary changes, blood glucose monitoring, and in some cases, insulin may be required.

Though it may be transient, untreated gestational diabetes can damage the health of the foetus or mother. Risks to the baby include macrosomia (high birth weight), congenital heart and central nervous system abnormalities, and skeletal muscle malformations. Increased levels of insulin in a foetus's blood may inhibit foetal surfactant production and cause infant respiratory distress syndrome. A high blood bilirubin level may result from red blood cell destruction. In severe cases, perinatal death may occur, most commonly as a result of poor placental perfusion due to vascular impairment. Labour induction may be indicated with decreased placental function. A caesarean section may be performed if there is marked foetal distress or an increased risk of injury associated with macrosomia, such as shoulder dystocia.

- ## Other types of diabetes

Less common types include monogenic diabetes, which is an inherited form of diabetes, and cystic fibrosis-related diabetes. Maturity onset diabetes of the young (MODY) is a rare autosomal dominant inherited form of diabetes, due to one of several single-gene mutations causing defects in insulin production. It is significantly less common than the three main types, constituting 1–2% of all cases. The name of this disease refers to early hypotheses as to its nature. Being due to a defective gene, this disease varies in age at presentation and in severity according to the specific gene defect; thus there are at least 13 subtypes of MODY. People with MODY often can control it without using insulin.

Some cases of diabetes are caused by the body's tissue receptors not responding to insulin (even when insulin levels are normal, which is what separates it from type 2 diabetes); this form is very uncommon. Genetic mutations (autosomal or mitochondrial) can lead to defects in beta cell function. Abnormal insulin action may also have been genetically determined in some cases. Any disease that causes extensive damage to the pancreas may lead to diabetes (for example, chronic pancreatitis and cystic fibrosis). Diseases associated with excessive secretion of insulin-antagonistic hormones can cause diabetes (which is typically resolved once the hormone excess is removed). Many drugs impair insulin secretion and some toxins damage pancreatic beta cells, whereas others increase insulin resistance (especially glucocorticoids which can provoke "steroid diabetes"). The ICD-10 (1992) diagnostic entity, *malnutrition-related diabetes mellitus* (MRDM or MMDM, ICD-10 code E12), was deprecated by the World Health Organization (WHO) when the current taxonomy was introduced in 1999. Yet another form of diabetes that people may develop is double diabetes. This is when a type 1 diabetic becomes insulin resistant, the hallmark for type 2 diabetes or has a family history for type 2 diabetes.

- ## How common is diabetes?

As of 2015, 30.3 million people in the United States, or 9.4 percent of the population, had diabetes. More than 1 in 4 of them didn't know they had the disease. Diabetes affects 1 in 4 people over the age of 65. About 90-95 percent of cases in adults are type 2 diabetes.

You are more likely to develop type 2 diabetes if you are age 45 or older, have a family history of diabetes, or are overweight. Physical inactivity, race, and certain health problems such as high blood pressure also affect your chance of developing type 2 diabetes. You are also more likely to develop type 2 diabetes if you have prediabetes or had gestational diabetes when you were pregnant. Learn more about risk factors for type 2 diabetes.

- ## What health problems can people with diabetes develop?

Over time, high blood glucose leads to problems such as

- Heart Disease
- Stroke
- Kidney Disease
- Eye Problems
- Dental Disease
- Nerve Damage
- Foot Problems

2. OVERVIEW OF DATA SET:

This dataset was taken from Kaggle. This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective of the dataset is to diagnostically predict whether or not a patient has diabetes, based on certain diagnostic measurements included in the dataset. Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage. The datasets consists of several medical predictor variables and one target variable, Outcome. Predictor variables includes the number of pregnancies the patient has had, their BMI, insulin level, age, and so on.

Data Set Link: https://www.kaggle.com/uciml/pima-indians-diabetes-database

- Pregnancies: Number of times pregnant
- Glucose: Plasma glucose concentration a 2 hours in an oral glucose tolerance test
- Blood Pressure: Diastolic blood pressure (mm Hg)
- Skin Thickness: Triceps skin fold thickness (mm)
- Insulin: 2-Hour serum insulin (mu U/ml)
- BMI: Body mass index (weight in kg / (height in m)^2)
- Diabetes Pedigree Function: Diabetes pedigree function
- Age: Age (years)
- Outcome: Class variable (0 or 1) 268 of 768 are 1, the others are 0

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
5	5	116	74	0	0	25.6	0.201	30	0
6	3	78	50	32	88	31.0	0.248	26	1
7	10	115	0	0	0	35.3	0.134	29	0
8	2	197	70	45	543	30.5	0.158	53	1
9	8	125	96	0	0	0.0	0.232	54	1

3. ANALYSIS OF DATA SET:

With the shape attribute applied on the data set, we come to know that the dataset has 768 rows and 9 columns out of which the ‘Outcome’ column is the categorical variable which we need to predict. With the info method applied on the data frame we observe that the pregnancies column has 768 values out of which all the values are non null values and the data type is int64. Coming to the next column that is the glucose column it has 768 values, all of them being non null values with data type int64. The next column is blood pressure with a length of 768, all non null values and the data type is int64. Skin thickness is the next column with same length of 768 non null values of data type int64. Insulin is the next column which indicates the insulin levels of a person with the length of 768 non null values of data type int64. The Body Mass Index (BMI) is the next column with 768 non null float64 values. Next comes the diabetes Pedigree function with 768 non null float 64 values. The last feature column is the age which indicate the age of a person in years

obviously of the data type int64 and 768 non null values. Finally the last column of the data set is the categorical column which is the label to be predicted by the implementation of models that is the outcome column with 768 non null in 64 values which predicts whether the person has diabetes or not. To summarize, we have 7 int64 values along with 2 float64 values which have an overall memory usage of 54.1 KB. Firstly we import the necessary libraries such as the Pandas for the manipulation of data frame, followed by the numpy which stands for numerical Python used to perform several numerical calculations on the data set, consequently we import the matplotlib library which is used to visualise the data with the help of several plots such as scatter plot, pie plot, bar graph etc. Lastly we import seaborn for advanced data visualisation. We import all of the libraries with short names like PD NP PLT SNS respectively for convenience. After importing all the necessary libraries we load the data set which is the diabetes.csv from kaggle as a comma separated file using the read_csv() function of the Pandas. With the help of head() and tail() methods we can have a glimpse of the first 10 as well as last 10 values of the data set. With the isnull().sum() method applied on the data we come to know the number of null values in the data set. After applying the method we find that all the columns in the data set have not even a single null value. With the value_counts() function applied on the data set which is imported as df we come to know the number of values existing in each column and their counts respectively. We observe from the analysis done till now that the columns of glucose blood pressure skin thickness insulin and BMI have some of their values equal to zero which is not at all practically possible, so in order to become convenient with the calculation we convert these values into mean values. After replacing all the 0 values with mean of the numbers. So we treat these zero values as outliers and since the data set is not of large size the best method to treat this outliers is to impute them. In this method of imputation, we replace the zero values either with the mean of the column or the mode or median or any respective statistical parameter of that respective column. Here we implement the practice of replacing the zero values with the mean of the respective column.

```
print("total number of rows : {0}".format(len(df)))
for col in df.columns:
    print("number of rows missing {}: {}".format(col,len(df.loc[df[col] == 0])))
```

```
total number of rows : 768
number of rows missing Pregnancies: 111
number of rows missing Glucose: 5
number of rows missing BloodPressure: 35
number of rows missing SkinThickness: 227
number of rows missing Insulin: 374
number of rows missing BMI: 11
number of rows missing DiabetesPedigreeFunction: 0
number of rows missing Age: 0
number of rows missing Outcome: 500
```

By giving the code of 4 lines we impute the column zero values with mean of respective columns:

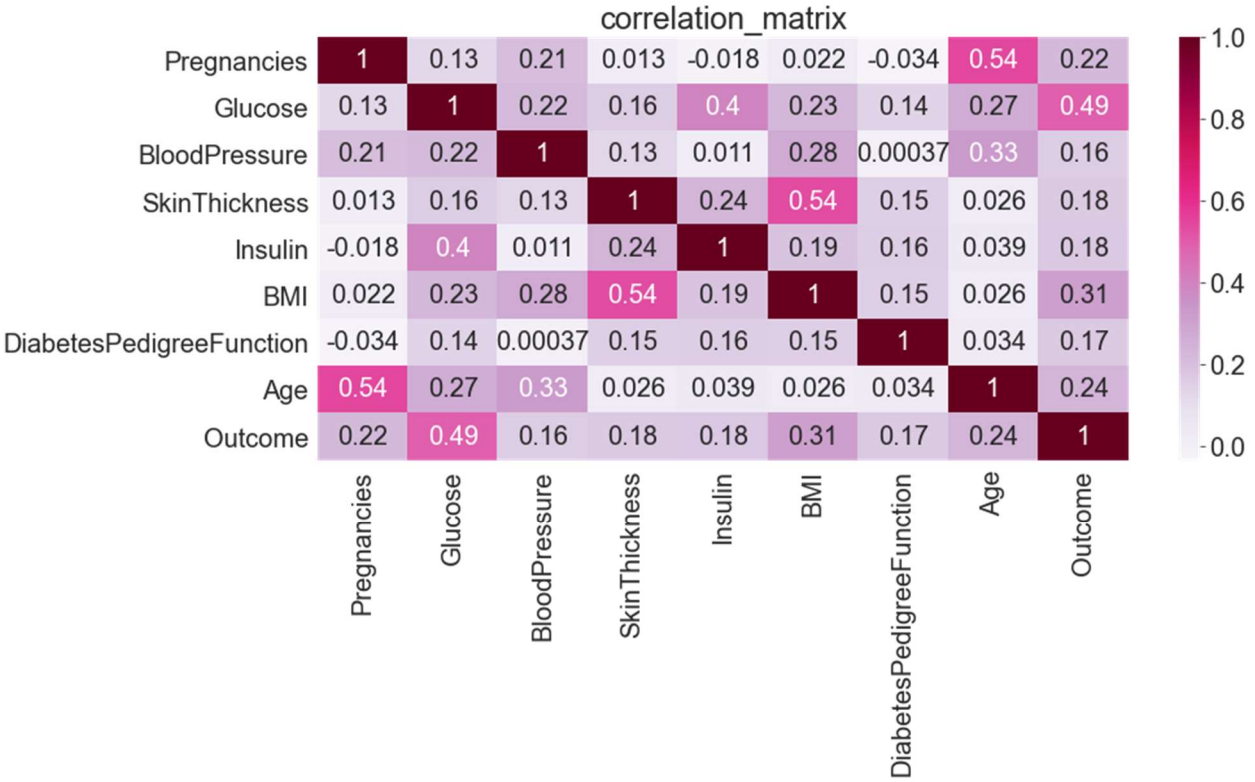
```
df.Glucose[df['Glucose']==0]=df.Glucose.mean()
df.BloodPressure[df['BloodPressure']==0]=df.BloodPressure.mean()
df.SkinThickness[df['SkinThickness']==0]=df.SkinThickness.mean()
df.Insulin[df['Insulin']==0]=df.Insulin.mean()
df.BMI[df['BMI']==0]=df.BMI.mean()
```

With the describe() method applied on the data set we can get the whole statistical summary of the data set columns. We get the gist of statistical summary like the count which represents the length of each column the mean standard deviation minimum value 25th percentile 50th percentile 75th percentile and the maximum values of each column in the data set respectively.

```
df.describe()
```

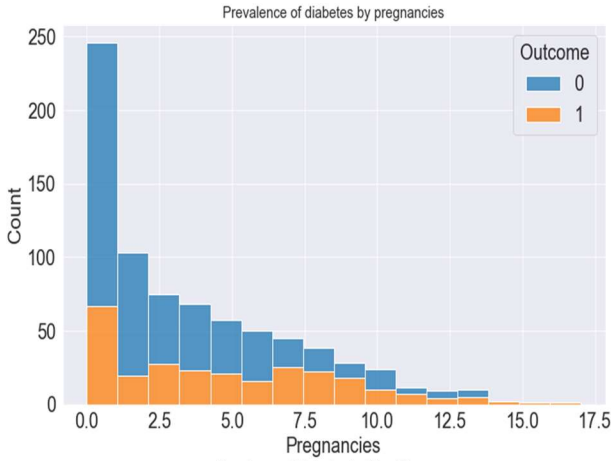
	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	121.681605	72.254807	26.606479	118.660163	32.450805	0.471876	33.240885	0.348958
std	3.369578	30.436016	12.115932	9.631241	93.080358	6.875374	0.331329	11.760232	0.476951
min	0.000000	44.000000	24.000000	7.000000	14.000000	18.200000	0.078000	21.000000	0.000000
25%	1.000000	99.750000	64.000000	20.536458	79.799479	27.500000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	79.799479	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

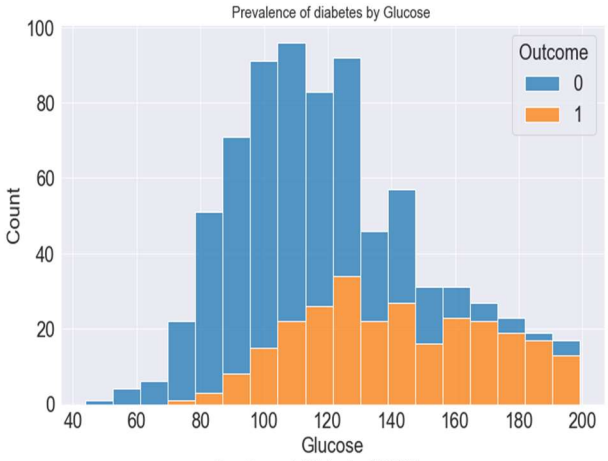
After successful observation on the data we now come to the trend analysing part of the data. We find the respective correlations of each and every column existing in the data. We plot the correlation using the heatmap function from the seaborn library with annotations and the respective formatting of colour rounding off values etc.

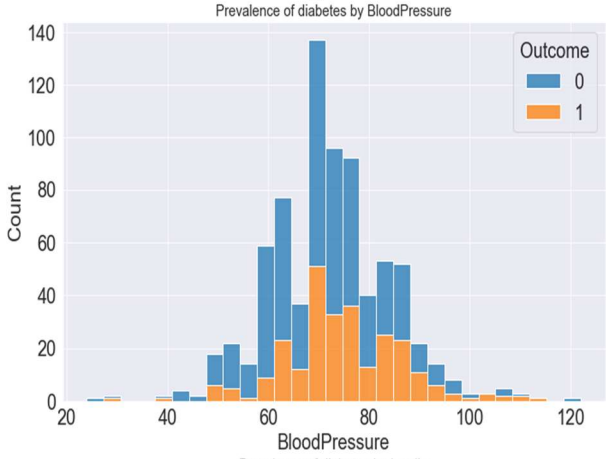


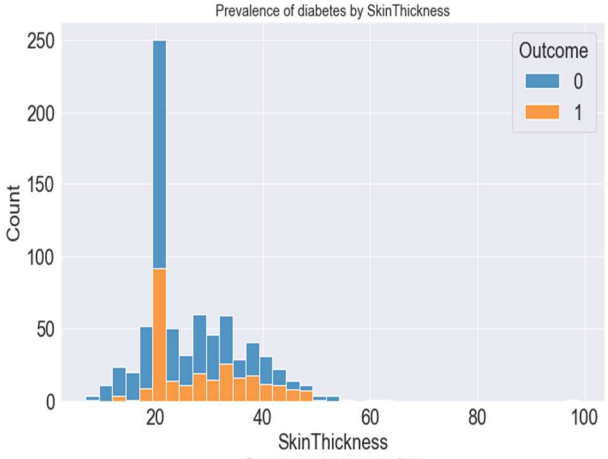
Correlation of two columns indicates the relationship between the two columns in other words it indicates how strongly two columns are related to each other and are affected by each other. From the heat map we can observe that there are no strong relations between any of the columns (>0.5) The greatest correlation is between the pregnancy and the age column followed by the correlation between glucose and output columns respectively followed by glucose and insulin columns. Positive correlation indicates that two columns are proportional to each other directly where as a negative correlation indicates that the two columns are inversely proportional to each other. In order to better visualise the correlation between the existing columns in the data set we use the pair plot function in the seaborn library from this we obtain a scatter plot of the correlations between each and every column in the data set.

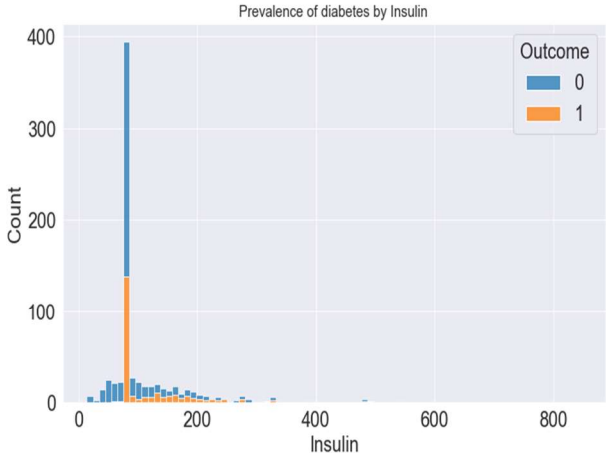
We now visualise the individual relation of each column with the output column. With the histplot() function of the matplotlib library we plot a bar plot of the number of people suffering from diabetes and not suffering from diabetes with respective medical characteristics. We also observe the characteristics of the people by plotting individual histograms of the medical characteristics of the people having diabetes (or output as 1).

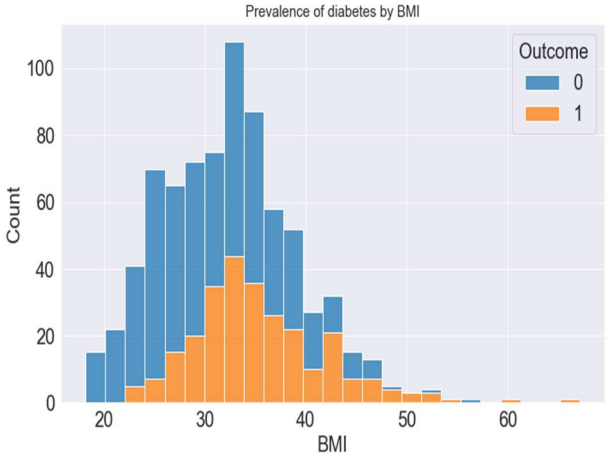

Prevalence of diabetes by pregnancies
Outcome
0
1
Count
Pregnancies

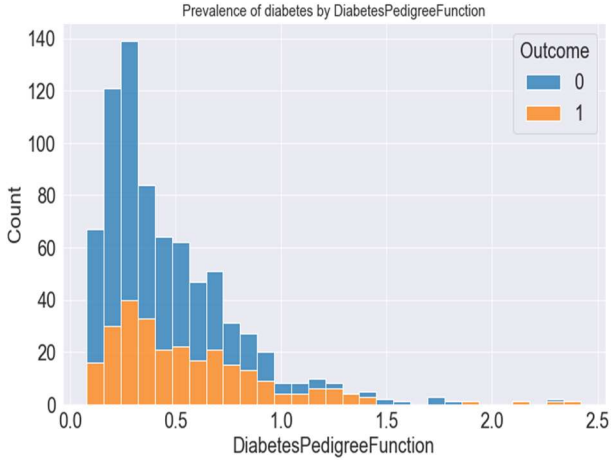

Prevalence of diabetes by Glucose
Outcome
0
1
Count
Glucose

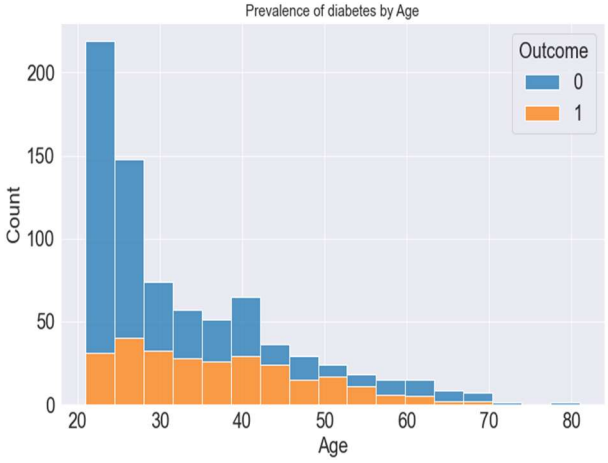

Prevalence of diabetes by BloodPressure
Outcome
0
1
Count
BloodPressure


Prevalence of diabetes by SkinThickness
Outcome
0
1
Count
SkinThickness


Prevalence of diabetes by Insulin
Outcome
0
1
Count
Insulin


Prevalence of diabetes by BMI
Outcome
0
1
Count
BMI


Prevalence of diabetes by DiabetesPedigreeFunction
Outcome
0
1
Count
DiabetesPedigreeFunction


Prevalence of diabetes by Age
Outcome
0
1
Count
Age

After the analysing part we now come to the implementation. We divide the data set into x and y. x stands for the features column where as y stands for the output column we now split the data into training data set and testing data set with the test data set of size 20%. For this we import the train_test_split module from the sklearn.model_selection and observe the lengths of each X_train, X_test, y_train, y_test respectively. Another important feature to be noted while implementing any model is that the difference in the values of the columns can make the model to be biased that implies if a model has relatively higher values than others, then the model might get confused that the particular column has higher significance than others. In order to avoid this chaos, we use the StandardScaler() module of sklearn.preprocessing using which we can scale all the values into a particular range of equal importance so that the model does not get confused. We fit the scaler to the training data set and test data set and once again visualise the statistical trend of each column.

After scaling data will look like:

```
array([[ 0.90832902,  0.93644016,  0.45816047, ...,  0.36864973,
         0.67740401,  1.69955804],
       [ 0.03644676, -0.81628595, -1.03864035, ..., -0.63292879,
        -0.07049698, -0.96569189],
       [-1.12606292,  1.43249471,  1.45602768, ...,  2.81536295,
        -0.11855487, -0.88240283],
       ...,
       [ 0.03644676, -0.91549686, -0.62286235, ..., -1.13371805,
        -0.95656442, -1.04898095],
       [ 2.0708387 , -1.2131296 ,  0.12553806, ..., -0.36107176,
        -0.50001442,  0.11706589],
       [ 0.32707418,  0.4734559 ,  0.79078287, ..., -0.09027668,
         0.52121586,  2.94889395]])
```

4. BULDING THE MODELS:

We start the implementation of each model.

a. LOGISTIC REGRESSION
b. K NEAREST NEIGHBORS
c. RANDOM FOREST CLASSIFIER
d. DECISION TREE CLASSIFIER
e. SUPPORT VECTOR CLASSIFIER

f. LinearDiscrimaninantAnalysis.
g. XG BOOST.

Building the models consumes a higher time for better time usage we are going to import the models from sklearn library by using the import commands as follows:

```
### Importing Models
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
import xgboost
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
```

After importing the modules we are going to create a dictionary of keys and values such that keys contain the name of the model which we desire to name so. Value contain the models that are to be fitted with the data.

For creating the dictionary of models we have used the zip function.

```
models = dict(zip(key,value))
print(models)
```

Models are as follows:

```
{'LogisticRegression': LogisticRegression(), 'LinearDiscriminantAnalysis':
LinearDiscriminantAnalysis(), 'KNeighborsClassifier': KNeighborsClassifier
(n_neighbors=3), 'SVC': SVC(), 'DecisionTreeClassifier': DecisionTreeClass
ifier(), 'RandomForestClassifier': RandomForestClassifier(), 'XGBClassifie
r': XGBClassifier(base_score=None, booster=None, colsample_bylevel=None,
              colsample_bynode=None, colsample_bytree=None, gamma=None,
              gpu_id=None, importance_type='gain', interaction_constraints
=None,
              learning_rate=None, max_delta_step=None, max_depth=None,
              min_child_weight=None, missing=nan, monotone_constraints=Non
e,
              n_estimators=100, n_jobs=None, num_parallel_tree=None,
              random_state=None, reg_alpha=None, reg_lambda=None,
              scale_pos_weight=None, subsample=None, tree_method=None,
              validate_parameters=None, verbosity=None)}
```

Thus using this models dictionary using the accuracy score calculation we have created a for loop such that it loops the models in the models dictionary and append the accuracy score of the each classification model to predicted list. After calculating the accuracy scores the data Is loaded to data frame such that the accuracies using the 7 models are as follows:

```
df_accuracy=pd.DataFrame({'algo_name':key,
                          'accuracy':predicted})
df_accuracy
```

	algo_name	accuracy
0	LogisticRegression	0.831169
1	LinearDiscriminantAnalysis	0.811688
2	KNeighborsClassifier	0.740260
3	SVC	0.792208
4	DecisionTreeClassifier	0.759740
5	RandomForestClassifier	0.818182
6	XGBClassifier	0.824675

From the data frame of accuracies we can observe that the Logistic Regression model is showing the high accuracy of 83.116% thus we can say it is the best fit model to prediction of the diabetes using the PIMA Data Set.

We have performed the hyperparameter tuning the for the best accuracy score model that we got.

5. HYPER PARAMETER TUNING:

A Machine Learning model is defined as a mathematical model with a number of parameters that need to be learned from the data. By training a model with existing data, we are able to fit the model parameters.

However, there is another kind of parameters, known as *Hyperparameters*, that cannot be directly learned from the regular training process. They are usually fixed before the actual training process begins. These parameters express important properties of the model such as its complexity or how fast it should learn.

Some examples of model hyperparameters include:

1. The penalty in Logistic Regression Classifier i.e. L1 or L2 regularization
2. The learning rate for training a neural network.
3. The C and sigma hyperparameters for support vector machines.

4. The k in k-nearest neighbors.

   The aim of this article is to explore various strategies to tune hyperparameter for Machine learning model.

   Models can have many hyperparameters and finding the best combination of parameters can be treated as a search problem. Two best strategies for Hyperparameter tuning are:

- GridSearchCV:

The traditional way of performing hyperparameter optimization has been *grid search*, or a *parameter sweep*, which is simply an exhaustive searching through a manually specified subset of the hyperparameter space of a learning algorithm. A grid search algorithm must be guided by some performance metric, typically measured by cross-validation on the training set or evaluation on a held-out validation set.

Since the parameter space of a machine learner may include real-valued or unbounded value spaces for certain parameters, manually set bounds and discretization may be necessary before applying grid search.

For example, a typical soft-margin SVM classifier equipped with an RBF kernel has at least two hyperparameters that need to be tuned for good performance on unseen data: a regularization constant *C* and a kernel hyperparameter γ. Both parameters are continuous, so to perform grid search, one selects a finite set of "reasonable" values for each.

Grid search then trains an SVM with each pair (*C*, γ) in the Cartesian product of these two sets and evaluates their performance on a held-out validation set (or by internal cross-validation on the training set, in which case multiple SVMs are trained per pair). Finally, the grid search algorithm outputs the settings that achieved the highest score in the validation procedure.

Grid search suffers from the curse of dimensionality, but is often embarrassingly parallel because the hyperparameter settings it evaluates are typically independent of each other

- RandomizedSearchCV:

Random Search replaces the exhaustive enumeration of all combinations by selecting them randomly. This can be simply applied to the discrete setting described above, but also generalizes to continuous and mixed spaces. It can outperform Grid search, especially when only a small number of hyperparameters affects the final performance of the machine learning algorithm. In this case, the optimization problem is said to have a low intrinsic dimensionality. Random Search is also embarrassingly parallel, and additionally allows the inclusion of prior knowledge by specifying the distribution from which to sample. RandomizedSearchCV implements a "fit" and a "score" method. It also implements "score_samples", "predict", "predict_proba", "decision_function", "transform" and "inverse_transform" if they are implemented in the estimator used.

The parameters of the estimator used to apply these methods are optimized by cross-validated search over parameter settings.

In contrast to GridSearchCV, not all parameter values are tried out, but rather a fixed number of parameter settings is sampled from the specified distributions. The number of parameter settings that are tried is given by n_iter.

If all parameters are presented as a list, sampling without replacement is performed. If at least one parameter is given as a distribution, sampling with replacement is used.

In our prediction we have performed the GridSerachCV such that we used the parameters after decomposition of the data using pipeline technique.

Parameters we considered are as follows:

```
n_components = list(range(1,x.shape[1]+1,1))
penalty = ['l1', 'l2']
```

```
params = dict(pca__n_components=n_components,
              logistic_Reg__penalty=penalty)
```

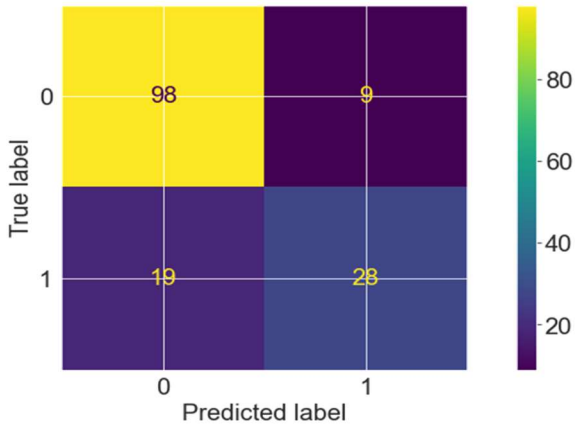
PCA involves the transformation of variables in the dataset into a new set of variables which are called PCs (Principal Components). The principal components would be equal to the number of original variables in the given dataset.

Penalized logistic regression imposes a penalty to the logistic model for having too many variables. This results in shrinking the coefficients of the less contributive variables toward zero. This is also known as **regularization**.

Accuracy Score of Grid Search Cross Validation of the Logistic Regression is less compared to the model while considered the default parameters. It might be a reason that the hyper parameters we considered might not be as same of the default in grid.

Insights: The probability that a person suffers from diabetes or not is highly determined by the level of glucose. Glucose level > 110, Number of pregnancies above 5, Blood Pressure > 70, Skin Thickness > 30, BMI > 31, Diabetes Pedigree Function > 0.33, also tend to indicate the risk of being affected with diabetes. Whereas from the plots, it is evident that there is no particular age limit for diabetes patients. People of any age span might be affected with diabetes if they lack proper nutrition. Since insulin and glucose have strong correlation, it becomes evident that insulin might have an indirect impact on the disease. Since age has a correlation with pregnancies and Blood Pressure, and skin thickness with BMI, so they too have an indirect effect.

Confusion Matrix is as follows :



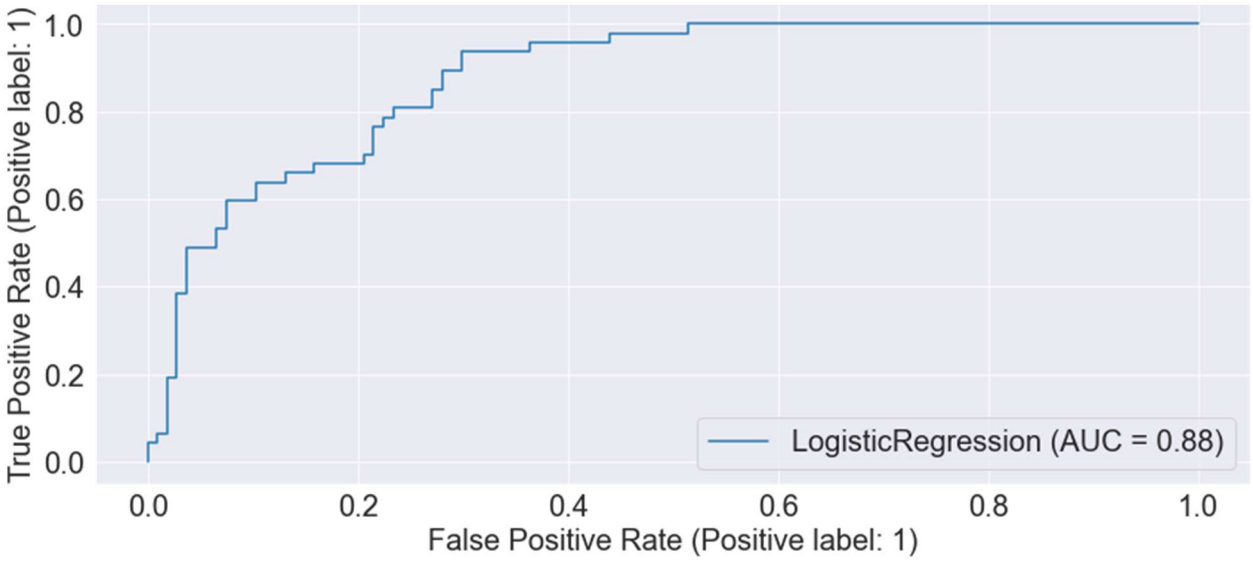
98 -True negative values.

28-True positive values after the prediction is done.

ROC CURVE :

The ROC curve does this by plotting *sensitivity*, the probability of predicting a real positive will be a positive, against *1-specificity*, the probability of predicting a real negative will be a positive. we have plotted the ROC curve using estimating model we considered to be high accurate.

The more the bend is towards the top left corner we can the model is more the accurate.



6. CONCLUSION: After performing the hyper parameter tuning on several models, and testing for the accuracy of each model, we conclude that the LOGISTIC REGRESSION algorithm showed the best performance with an accuracy of 83.11%.

True and Predicted values are as follows.

```
# prediction of the Outcome by using the best fit model.
Outcome_prediction=lr_clf.predict(x_test_sc)
Outcome_prediction
```

```
array([1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1,
       1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1,
       0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0],
      dtype=int64)
```

```
[2]: y_test.values

[2]: array([1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1,
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1,
            1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1,
            1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
            1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1,
            0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
            0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0],
           dtype=int64)
```

At final we are concluding the prediction with plotting of the predicted and true values after the model's prediction.