Predicting Diabetes using Machine Learning

1. Introduction

Diabetes is one of deadliest diseases in the world. It is not only a disease but also a creator of different kinds of diseases like heart attack, blindness, kidney diseases, etc. The normal identifying process is that patients need to visit a diagnostic centre, consult their doctor, and sit tight for a day or more to get their reports. Moreover, every time they want to get their diagnosis report, they have to waste their money in vain. Diabetes Mellitus (DM) is defined as a group of metabolic disorders mainly caused by abnormal insulin secretion and/or action. Insulin deficiency results in elevated blood glucose levels (hyperglycemia) and impaired metabolism of carbohydrates, fat and proteins. DM is one of the most common endocrine disorders, affecting more than 200 million people worldwide.

Early prediction of diabetes for a patient with a higher accuracy by combining the results of different machine learning techniques also can draw meaningful insights on most significant parameters causing Diabetes. so that the people can change their lifestyles to avoid diabetes.

2. Problem statement

The main objective of the project is to predict diabetes different supervised machine learning methods including: SVM, Logistic regression, KNN and deep learning. This project also aims to propose an effective technique for earlier detection of the diabetes disease and also draw meaningful insights which can be useful for the people to avoid diabetes by making lifestyle changes.

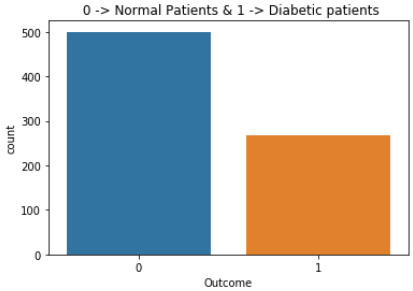
3. Data

To solve the problem, we will need the good number of patient data with all factors(predictor features) causing diabetes. I have received data from kaggle with 8 most probable medical predictor features are:

1. Pregnancies :Number of times pregnant
2. Glucose : Plasma glucose concentration a 2 hours in an oral glucose tolerance test
3. Blood Pressure : Diastolic blood pressure (mm Hg) ·
4. Skin Thickness : Triceps skin fold thickness (mm)
5. Insulin : 2-Hour serum insulin (mu U/ml)
6. BMI : Body Mass Index (weight in kg/(height in m)^2)
7. Diabetes Pedigree Function: Diabetes pedigree function on genetic influence and hereditary risk
8. Age : Age (years)

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 consist of several medical predictor (independent) variables and one target (dependent) variable, Outcome. Independent variables include the number of pregnancies the patient has had, their BMI, insulin level, age, and so on.
https://www.kaggle.com/uciml/pima-indians-diabetes-database/data/diabetes.csv



Data consistency is very important. If any missing values or zero values in the data are to be replaced in a meaningful way. It is done in the next section predictive modaling.

4. Feature selection

After data cleaning, there were 768 samples and 8 features in the data with no null and no redundancy in features that may affect modal accuracy.

:

	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

```
Missing values :   0

Unique values :
 Pregnancies                 17
Glucose                     136
BloodPressure                47
SkinThickness                51
Insulin                     186
BMI                         248
DiabetesPedigreeFunction    517
Age                          52
Outcome                       2
dtype: int64
```

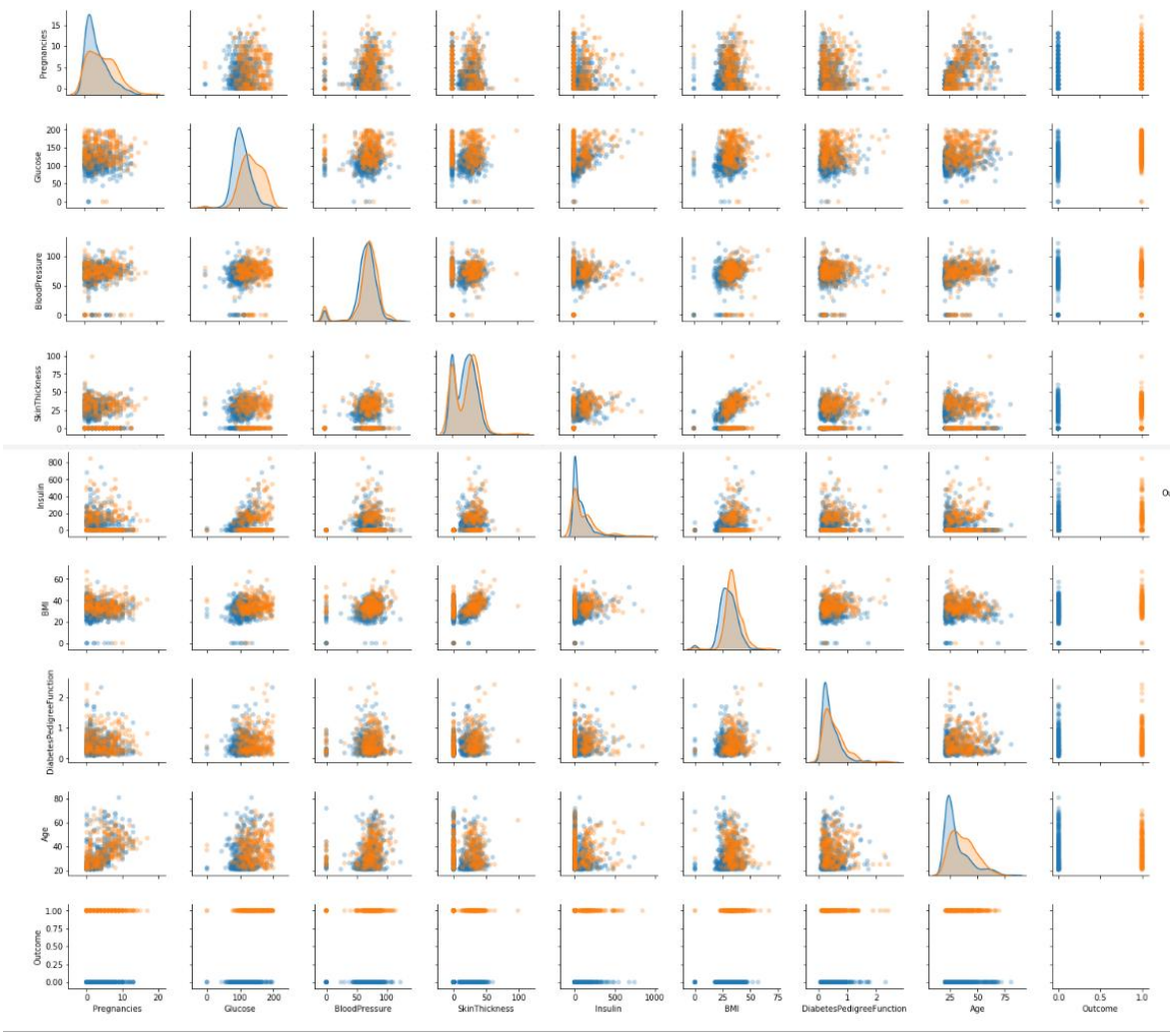
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
Pregnancies                 768 non-null int64
Glucose                     768 non-null int64
BloodPressure               768 non-null int64
SkinThickness               768 non-null int64
Insulin                     768 non-null int64
BMI                         768 non-null float64
DiabetesPedigreeFunction    768 non-null float64
Age                         768 non-null int64
Outcome                     768 non-null int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

5. Predictive Modelling

Predicting the diabetes is a classification problem because we have to classify particular patient is weather diabetic **Yes/No** by generating weights for 8 features.

	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	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	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

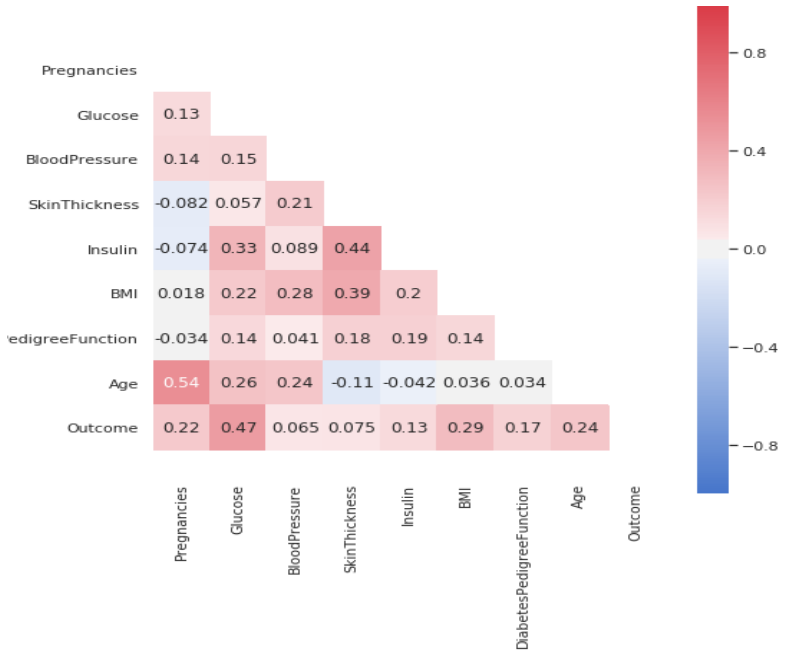
Description of the data Set



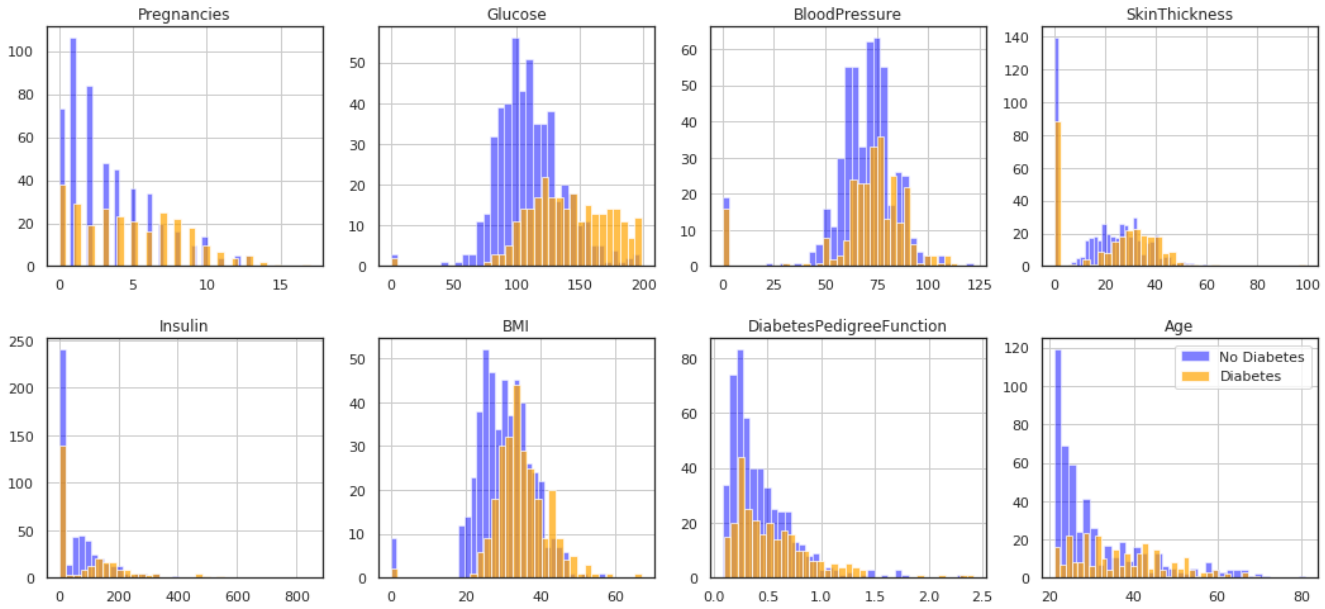
Relation between various features and outcome

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
Pregnancies	1.000000	0.129459	0.141282	-0.081672	-0.073535	0.017683	-0.033523	0.544341	0.221898
Glucose	0.129459	1.000000	0.152590	0.057328	0.331357	0.221071	0.137337	0.263514	0.466581
BloodPressure	0.141282	0.152590	1.000000	0.207371	0.088933	0.281805	0.041265	0.239528	0.065068
SkinThickness	-0.081672	0.057328	0.207371	1.000000	0.436783	0.392573	0.183928	-0.113970	0.074752
Insulin	-0.073535	0.331357	0.088933	0.436783	1.000000	0.197859	0.185071	-0.042163	0.130548
BMI	0.017683	0.221071	0.281805	0.392573	0.197859	1.000000	0.140647	0.036242	0.292695
DiabetesPedigreeFunction	-0.033523	0.137337	0.041265	0.183928	0.185071	0.140647	1.000000	0.033561	0.173844
Age	0.544341	0.263514	0.239528	-0.113970	-0.042163	0.036242	0.033561	1.000000	0.238356
Outcome	0.221898	0.466581	0.065068	0.074752	0.130548	0.292695	0.173844	0.238356	1.000000

Correlation between features and outcome



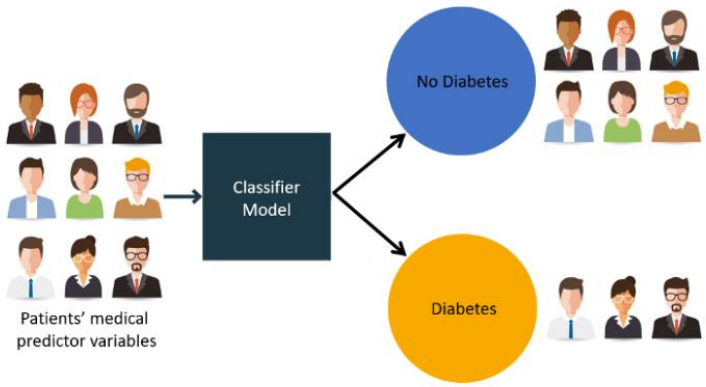
Correlation between various features and out come



Relation between each feature with out come

From the above graphs we can clearly see zero values in Glucose, Blood Pressure, Skin Thickness, Insulin, BMI. They can't be zero because with zero values patient can't be alive, they are outliers. They are replaced with their mean values.

5. Methodology



To solve this classification problem by building a predictive machine learning model based on diagnostic measurements whether a patient has diabetes. This is a binary (2-class) classification project with supervised learning.

We apply different supervised machine learning classification methods like:

1. KNN
2. Logistic Regression
3. Decision Tree
4. Random Forest
5. Gradient Boosting
6. SVM
7. Deep Learning
8. Gaussian Naïve Bayes

We compare the results from these machine learning models using performance metrics. Based on the performance of the model we can select best models and using the data science technics we can fine tune the modal for best prediction modal with best possible accuracy.

I have used the Jupiter Note Book with Python and various standard libraries of Pandas and Numpy are imported, along with visualisation libraries of Matplotlib and Seaborn. There are also a host of models and measurement metrics imported from Scikit-Learn library.

To identify the best performing model:

1. Cross validate all the above models and then find the top performing two or three models.
2. Now optimise parameters for these models to pick the best tuned model.

5.1. Cross Validation

Cross validation of all the above models can be performed using Scikit-learn StratifiedKFold.

We will get summary of their performance in the form of a following table

	model	accuracy	precision	recall	f1score	rocauc	logloss	timetaken
0	GaussianNB	0.757257	0.671205	0.592470	0.624538	0.824943	10.316912	0
1	DeepLearningMLP	0.693762	0.548362	0.373090	0.500218	0.710875	9.195539	0
2	LogisticRegression	0.758856	0.694358	0.536323	0.600393	0.821686	9.868305	0
3	KNN	0.723044	0.607567	0.578627	0.589537	0.758864	11.214008	0
4	DecisionTree	0.703545	0.542561	0.564673	0.579366	0.658880	10.316876	0
5	RandomForest	0.768693	0.704132	0.508638	0.562494	0.778645	10.092593	0
6	SVC	0.651473	0.000000	0.000000	0.000000	0.671100	12.111000	0

From the above table based on the Recall and F1 score best performing models are

1. Gaussian Naïve Bayes
2. KNN
3. Decision Tree

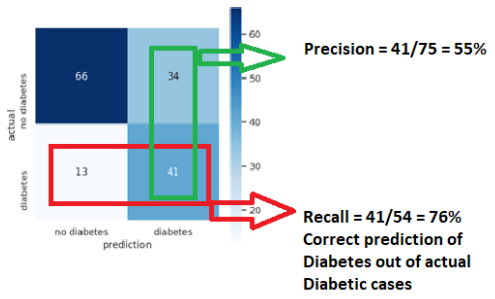
5.2. Optimising /Parameter Tuning

Now we tune the above three models for their optimum parameter values

```
GaussianNB accuracy score is
Training: 76.38%
Test set: 70.13%

Adjust threshold to 0.25:
Precision: 0.5493,   Recall: 0.7222,   F1 Score: 0.6240
GaussianNB confusion matrix:
 [[68 32]
 [15 39]]

Default threshold of 0.50:
Precision: 0.5667,   Recall: 0.6296,   F1 Score: 0.5965
GaussianNB confusion matrix:
 [[74 26]
 [20 34]]

Adjust threshold to 0.75:
Precision: 0.6136,   Recall: 0.5000,   F1 Score: 0.5510
GaussianNB confusion matrix:
 [[83 17]
 [27 27]]

Optimal threshold 0.220
Precision: 0.5467,   Recall: 0.7593,   F1 Score: 0.6357
GaussianNB confusion matrix:
 [[66 34]
 [13 41]]
GaussianNB AUC: 0.7646
GaussianNB Log-loss: 0.9440
```

```
KNearest Neighbour accuracy score is
Training: 80.13%
Test set: 67.53%

Adjust threshold to 0.25:
Precision: 0.4800,   Recall: 0.6667,   F1 Score: 0.5581
KNearest Neighbour confusion matrix:
 [[61 39]
 [18 36]]

Default threshold of 0.50:
Precision: 0.5400,   Recall: 0.5000,   F1 Score: 0.5192
KNearest Neighbour confusion matrix:
 [[77 23]
 [27 27]]

Adjust threshold to 0.75:
Precision: 0.6333,   Recall: 0.3519,   F1 Score: 0.4524
KNearest Neighbour confusion matrix:
 [[89 11]
 [35 19]]

Optimal threshold 0.000
Precision: 0.4352,   Recall: 0.8704,   F1 Score: 0.5802
KNearest Neighbour confusion matrix:
 [[39 61]
 [ 7 47]]
KNearest Neighbour AUC: 0.7005
KNearest Neighbour Log-loss: 2.6937
```

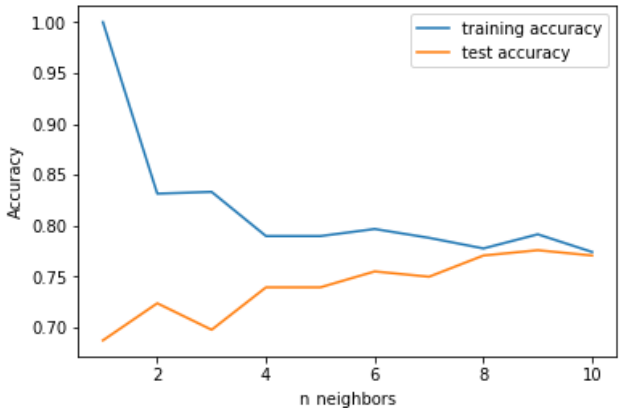
By comparing the optimum parameters of the three models Gaussian Naïve Bayes is the best model with accuracy of 76.46% with low log loss and very good Recall score.



6.1. k-Nearest Neighbours

The k-NN algorithm is arguably the simplest machine learning algorithm. Building the model consists only of storing the training data set. To make a prediction for a new data point, the algorithm finds the closest data points in the training data set—its "**nearest neighbours**."

Let's investigate the connection between model complexity and accuracy:



The above plot shows the training and test set accuracy on the y-axis against the setting of **n_neighbours** on the x-axis. Considering if we choose one single nearest neighbour, the prediction on the training set is perfect. But when more neighbours are considered, the training accuracy drops, indicating that using the single nearest neighbour leads to a model that is too complex.

The best performance is somewhere around 9 neighbours. The plot suggests that we should choose **n_neighbours=9** , which gives the results as follows:

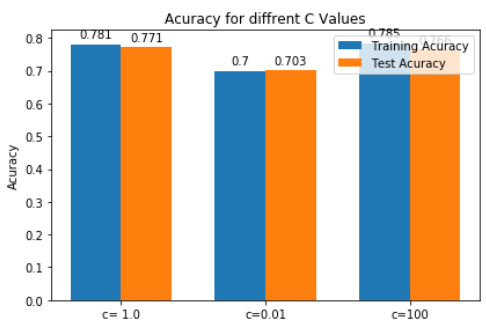
Accuracy of K-NN classifier on training set: 0.79
Accuracy of K-NN classifier on test set: 0.78

6.2. Logistic Regression

Logistic Regression is one of the most frequently used and simple classification algorithms.

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist.
In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model (a form of binary regression).

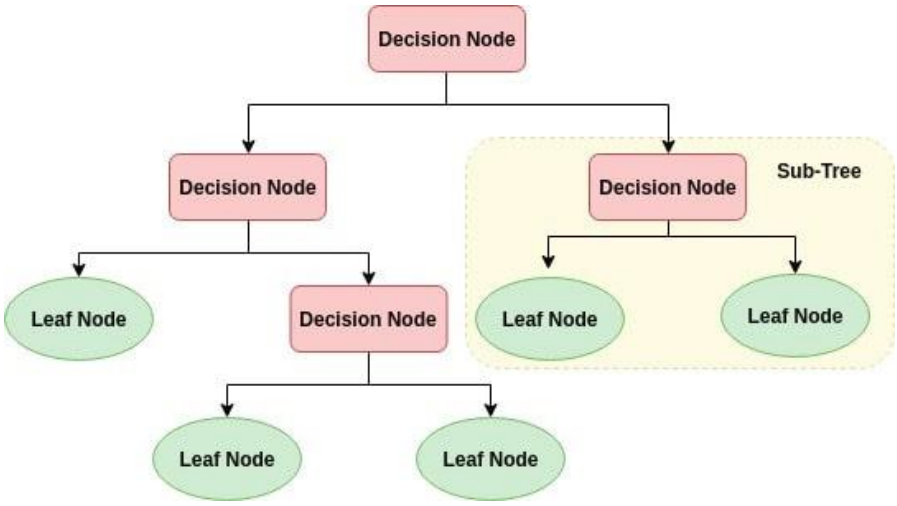
Like all regression analyses, the logistic regression is a predictive analysis. Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.



From the above figure we can easily say Logistic Regression with default setting with c=1 giving the better accuracy with both testing and training data.

6.3. Decision Tree

The decision tree classifier (Pang-Ning, 2006) creates the classification model by building a decision tree. Each node in the tree specifies a test on an attribute, each branch descending from that node corresponds to one of the possible values for that attribute.



6.4. Deep Learning MLP

A multilayer perceptron (MLP) is a class of feed forward artificial neural network. A MLP consists of at least three layers of nodes: an input layer, a hidden layer and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called back propagation for training. Its multiple layers and non-linear activation distinguish MLP from a linear perceptron. It can distinguish data that is not linearly separable**.**

Disadvantage is that the number of total parameters can grow to very high . This is inefficient because there is redundancy in such high dimensions. Another disadvantage is that it disregards spatial information. It takes flattened vectors as inputs. A light weight MLP (2–3 layers) can easily achieve high accuracy with MNIST dataset.

6.5. Gaussian Naive Bayes

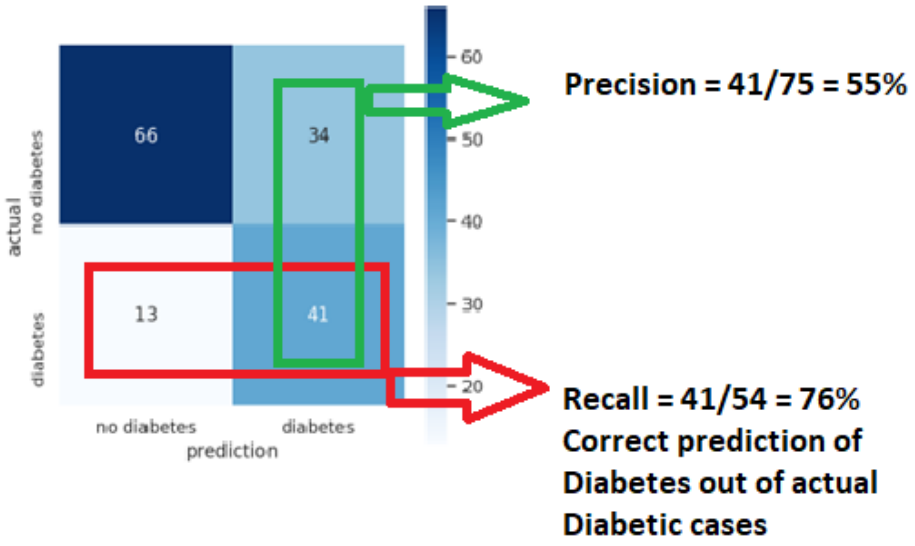
Naive Bayes is a simple but surprisingly powerful algorithm for predictive modelling.

Naive Bayes can be extended to real-valued attributes, most commonly by assuming a Gaussian distribution.

This extension of naive Bayes is called Gaussian Naive Bayes. Other functions can be used to estimate the distribution of the data, but the Gaussian (or Normal distribution) is the easiest to work with because you only need to estimate the mean and the standard deviation from our training data.

7. Final Result

We applied eight different ML classification models on the data, by careful observation of various parameters Importantly Recall metric, F1 score and Log Loss, Gaussian Naïve Bayes is performed well. It can be further visualised using Confusion Matrix shown below



8. Conclusion

In this project, the Gaussian Naive Bayes model has achieved a prediction (Recall) score of 76%, ie, out of all diabetic patients, 76% of them will be correctly classified using medical diagnostic measurements. **'Glucose'** and **'BMI'** are the most important medical predictor features.

For a healthy living, look after your sugar intake and your weight**.**

9. Limitations and future scope

1. Due to time constraint I have used the data that is available. To get correct results we have to get the latest medical records available or to conduct a survey in a scientific way. Soon I will try to get the latest data of Indian patients in my region and I will rerun the project.
2. Results may be limited because Size of the data set is small, very old and limited to Pima Indian women patient only.

3. These days they are using HBAc blood Test to confirm Diabetes; these reading are not considered either.
4. Recall accuracy can be improved

Reference:

https://github.com/topics/data-science

https://www.kaggle.com/

http://www.kmdatascience.com/2017/07/k-folds-cross-validation-in-python.html

https://scikitlearn.org/stable/modules/generated/sklearn.model_selection.StratifiedKFold.html#sklearn.model_selection.StratifiedKFold