Project Report

A report submitted in partial fulfillment of the requirements for the

Diabetes Prediction

By

Vedant Sandip Salunke

Under Supervision of

Sushank Dahiwadkar

Wisdom Sprouts, Pune

Table of Contents

SR. No	Content	Page NO.
1	Abstract	2
2	Technology Used	3
3	Algorithms Used	3-4
4	Components	5-6
5	Component Diagram	6
6	Project Algorithm Steps	7
7	Comparison of all the Algorithms Tested	8
8	Output Screenshot	8-23
9	Github Link of the project	23
10	Future Scope	23

ABSTRACT

- Diabetes is one of the chronic diseases that causes blood sugar levels to rise. If diabetes is left untreated and undiagnosed, it can lead to complications. The time-consuming identification process leads to a patient's referral to a diagnostic Centre and consultation with a doctor.
- The aim of this project is to create a model that can reliably predict the accuracy of diabetes in patients. Dataset splits into three then classification techniques are implemented. Training Dataset, Dataset sample that is used to fit the model. Using machine learning methods, this project aims to assist doctors and physicians in the early detection of diabetes.

- Machine learning has the great ability to revolutionize the diabetes risk prediction with the help of advanced computational methods and availability of a large amount of epidemiological and genetic diabetes risk dataset. Detection of diabetes in its early stages is the key for treatment. This work has described a machine learning approach to predicting diabetes or not. The technique may also help researchers to develop an accurate and effective tool that will reach at the table of clinicians to help them make better decisions about disease status.

- To detect diabetes at an early stage, this project employs machine learning classification algorithms: Logistic Regression, Gaussian Naive Bayes, K Neighbours, SVM, Decision tree, Random Forest, Bagging Classifier, Ada Boost Classifier and Gradient Boosting Classifier are implemented. The Pima Indians Diabetes Database (PIDD) is used in the experiments. The National Institute of Diabetes and Digestive and Kidney Diseases provided the results. The dataset's purpose is to diagnose whether a patient has diabetes using diagnostic measures included in the dataset. Various measures like Precision, Accuracy, Specificity, and Recall are measured over classified instances using Confusion Matrix. The accuracy of the algorithms used are compared and discussed. The study's comparison of the various machine learning techniques shows which algorithm is better suited for diabetes prediction. Using machine learning methods, this project aims to assist doctors and physicians in the early detection of diabetes.

Technologies Used for Churn Analysis:

1. **Programming Language:**
   - Employed programming languages such as Python for their extensive libraries and tools tailored for data analysis, machine learning, and statistical modelling.
2. **Data Analysis Libraries**
   - Utilized Pandas and NumPy to manipulate and analyze the dataset efficiently, facilitating tasks such as data cleaning, preprocessing, and exploratory data analysis (EDA).
3. **Data Visualization:**
   - Leveraged Matplotlib and Seaborn to create informative visualizations, aiding in the interpretation of trends, patterns, and correlations within the dataset.
4. **Machine Learning Libraries:**
   - Incorporated Scikit-Learn for the implementation of machine learning models, enabling tasks such as model selection, hyper-parameter tuning, and performance evaluation.
5. **Jupyter Notebooks:**
   - Utilized Jupyter Notebooks for an interactive and iterative approach to data analysis, allowing for real-time visualization and documentation of the analysis process.
6. **Collaboration Platforms:**
   - Utilized collaborative platforms like GitHub for version control, enabling seamless collaboration, code sharing, and tracking changes throughout the churn analysis project.
7. **Front-end development:**
   - For the user interface html page have been created.
8. **Flask:**
   - Considered Flask, a lightweight web framework, for potential integration to create a web-based quiz application and expand accessibility.

Algorithms Used:

1. **Linear Regression (LR) :-**

This method predicts the class of numerical variable. To predict binary results ($y = 0$ or 1) LR use statistical techniques. The possibilities of an event occurring are used to make LR predictions. The sigmoid function in the LR algorithm maps each data point. The standard logistic function results in an S-shaped curve. The sigmoid function is displayed in the following equation:

2. **Decision Tree Classifier:**
    - Decision Tree Classifier is utilized for its ability to handle non-linear relationships and hierarchical decision-making. It excels in segmenting the dataset based on features, aiding in the identification of influential factors contributing to customer churn.
    - Decision trees are constructed by recursively splitting the dataset based on features that maximize information gain or Gini impurity. The resulting tree structure provides insights into the decision logic.
3. **Random Forest Classifier:**
    - Random Forest Classifier extends the capabilities of decision trees by employing an ensemble learning approach. It aggregates predictions from multiple decision trees, reducing overfitting and enhancing predictive accuracy.
    - The Random Forest algorithm builds multiple decision trees on different subsets of the dataset. It then combines their predictions through voting or averaging, resulting in a more robust and generalizable model.
    - Random Forests excel in handling complex datasets, providing improved accuracy and reducing the risk of overfitting associated with individual decision trees. They offer robustness and scalability for churn prediction tasks.

4. **Support Vector Machine:-**
    - SVM the hyperplane is constructed among various classes or objects. Calculating the size of the problem space produces the hyperplane. SVM also allows for data reduction to balance data dimensions. Using the support vectors and class corner points, the marginal distance between the classes is calculated from the centre of the hyperplane. Some of the variables used in SVM are kernels, C coefficients, and intercepts.

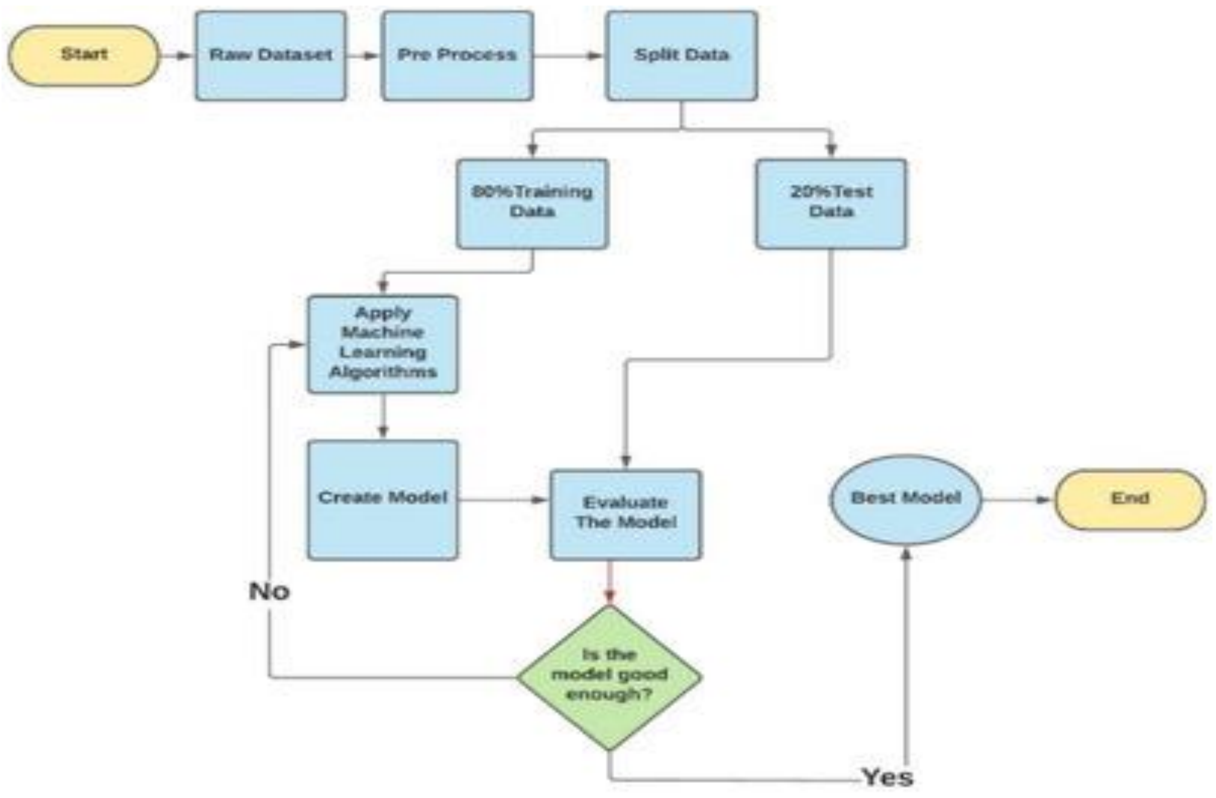
Components/Modules of the Churn Analysis Project:

- **Data Collection***:* We collected two alternative datasets, each with a different number of factors or features, to ensure the model's robustness. The datasets were compiled from a wide variety of sources, including diabetes statistics and health characteristics obtained from people around the world and from various health institutes.
- **Data Analysis and Data Preprocessing***:* Several pre-processing techniques are applied on the datasets before feeding these datasets into the machine learning model so that the performance of the model is improved. The pre-processing tasks include removing outliers and dealing with missing values, data standardization, encoding, and so on.
- **Outliers Removal** - Attributes' values that are beyond acceptable boundaries and have high variation from the rest of the respective attribute's value might be present in the dataset. Such attributes' value might degrade the machine learning algorithm's performance. To eliminate such outliers, we applied the IQR (Inter-quartile Range) approach.
- **Missing value Handling** - To improve model performance, the mean value of each attribute was employed for handling the missing values.
- **Label Encoding** - Label encoding is the process of converting the labels of text/categorical values into a numerical format that ML algorithms can interpret.

For example, the categorical values of *Junkfood* consumption status yes to '1' and No to '0' have been converted.

- **Model Construction and Prediction***:* To construct the predictive model, 80% of the pre-processed data has been used for training while the remaining 20% data is used for the testing purpose.
- **Performance Analysis***:* We have analyzed the results of the proposed model in terms of several performance metrics. The algorithm that provides highest prediction accuracy is selected as the best algorithm for the web application development.
- **Performance Comparison***:* In this step, the accuracy of the proposal has been compared with some recent works related to diabetes prediction. The performance results indicate that the proposal can improve the performance compared to the recent related research.
- **Web Application development***:* To develop a smart web application, we have used the Flask micro-framework and integrated the best model. To predict diabetes, a user is required to submit a form with necessary numbers of diabetes related parameters.

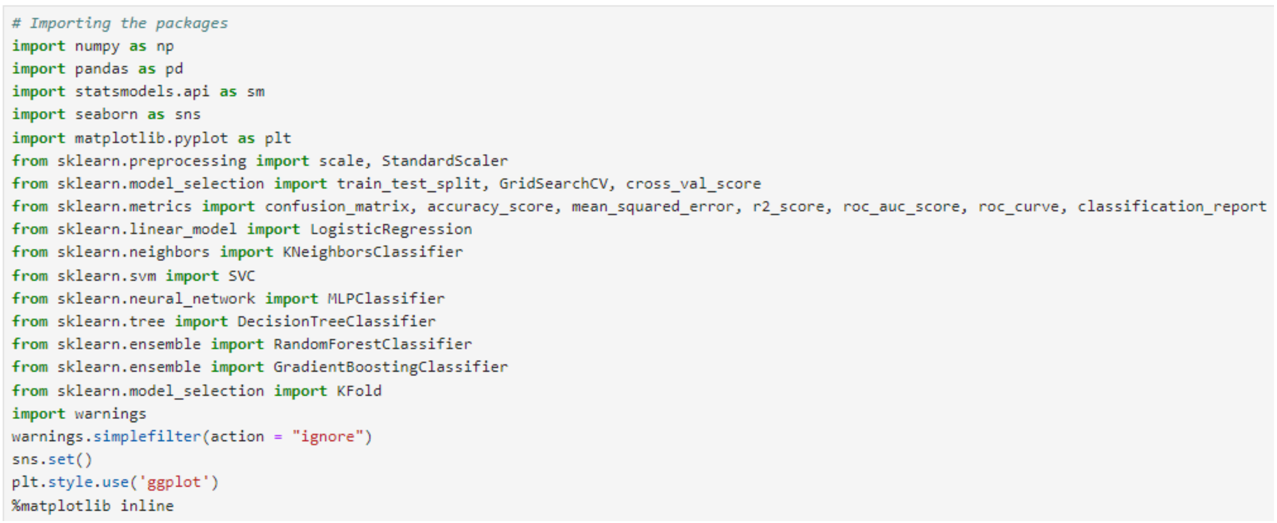
Component Diagram:



Project Algorithm Steps:

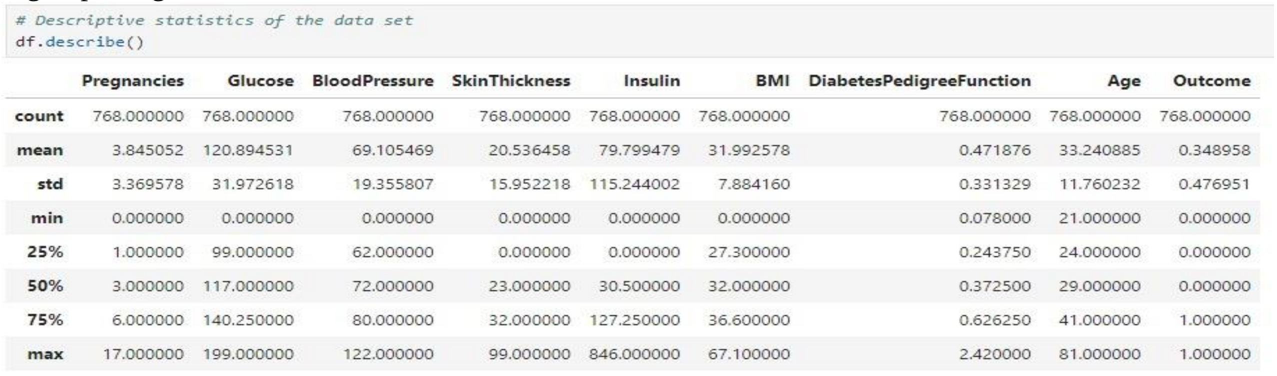
1. STRUCTURE DATA ANALYSIS:-

The datasets consist of several medical predictor variables and one target variable, Outcome. Predictor variables include the number of pregnancies the patient has had, their BMI, insulin level, age etc.



```
# Importing the packages
import numpy as np
import pandas as pd
import statsmodels.api as sm
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import scale, StandardScaler
from sklearn.model_selection import train_test_split, GridSearchCV, cross_val_score
from sklearn.metrics import confusion_matrix, accuracy_score, mean_squared_error, r2_score, roc_auc_score, roc_curve, classification_report
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.neural_network import MLPClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.model_selection import KFold
import warnings
warnings.simplefilter(action = "ignore")
sns.set()
plt.style.use('ggplot')
%matplotlib inline
```

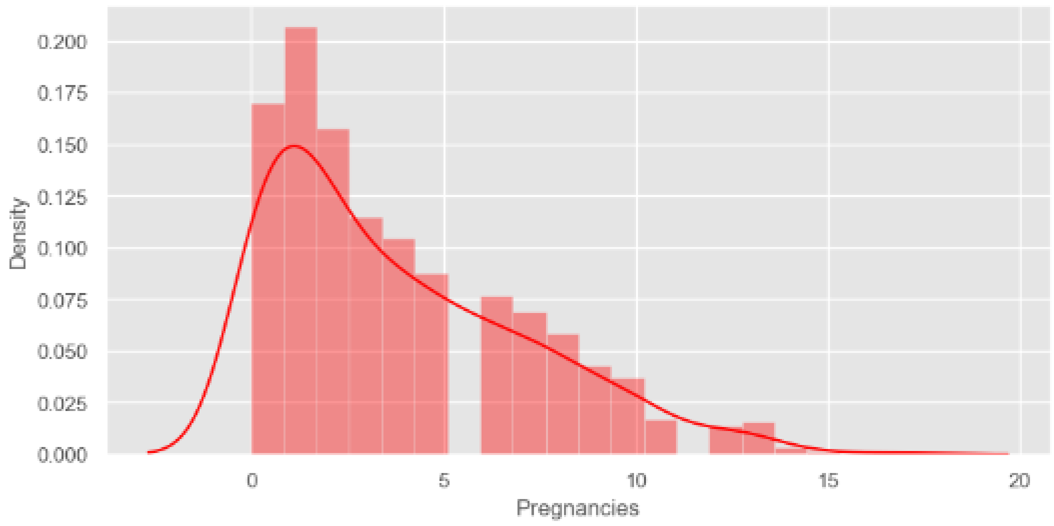
Fig Importing Libraries



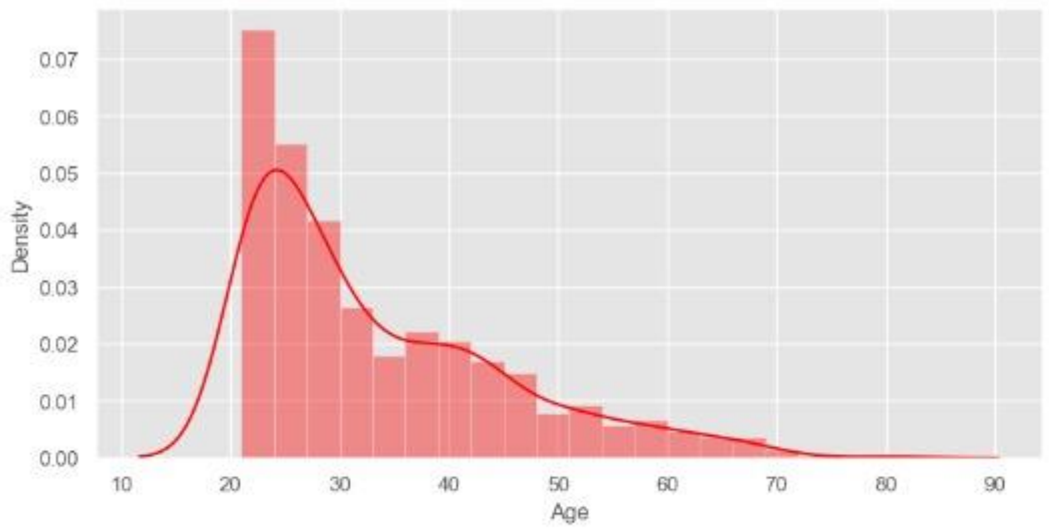
```
# Descriptive statistics of the data set
df.describe()
```

	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	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

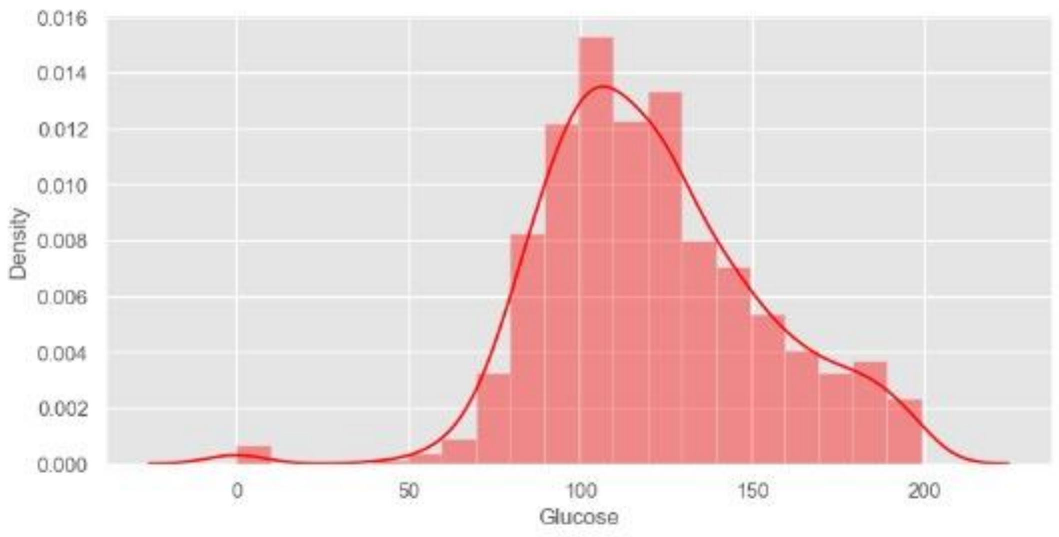
2. HISTOGRAM PLOT OF DATA



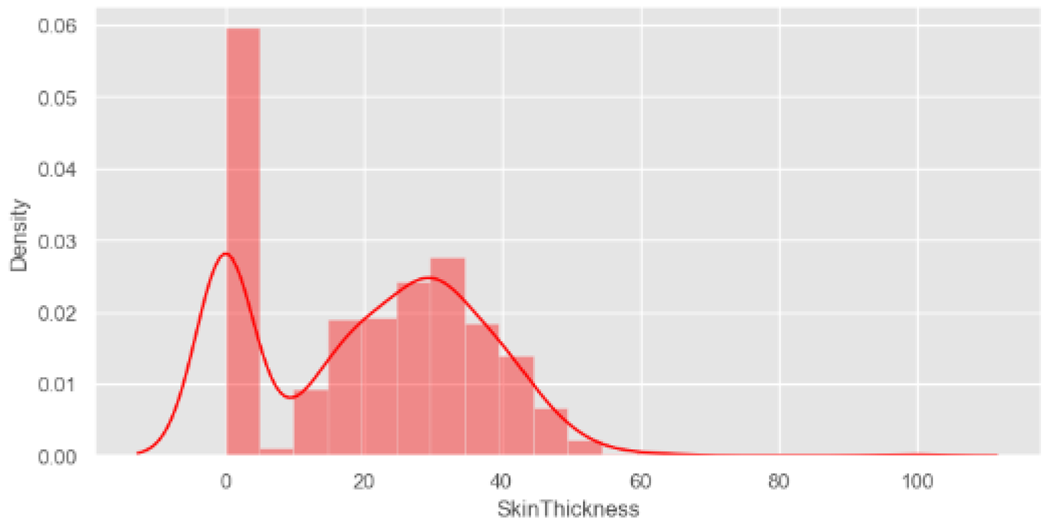
Wisdom Sprouts, Pune



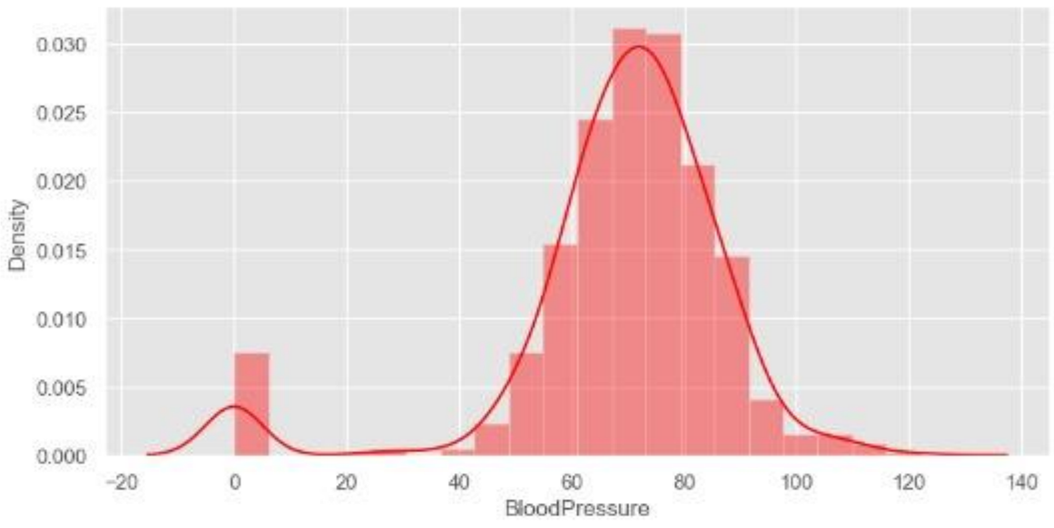
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
Density
10
20
30
40
50
60
70
80
90
Age

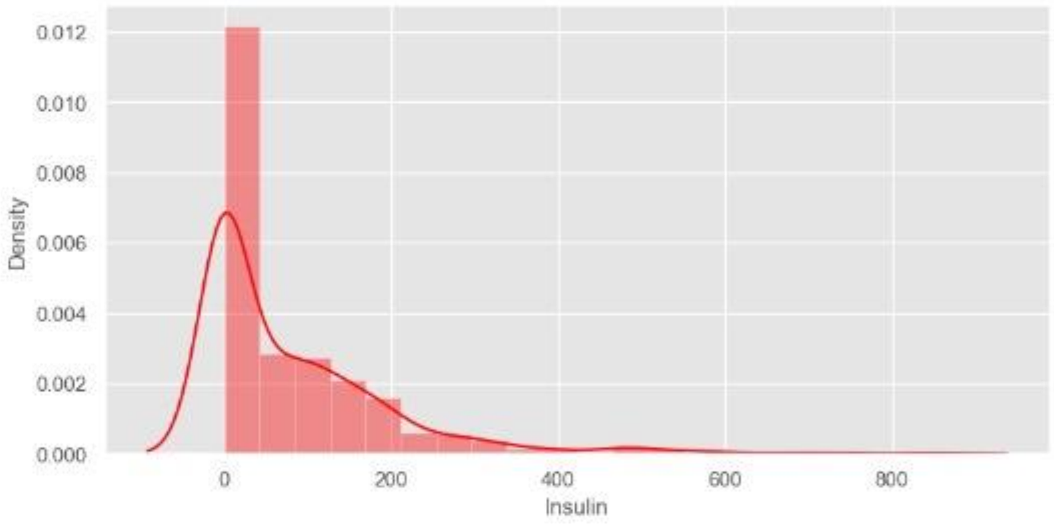


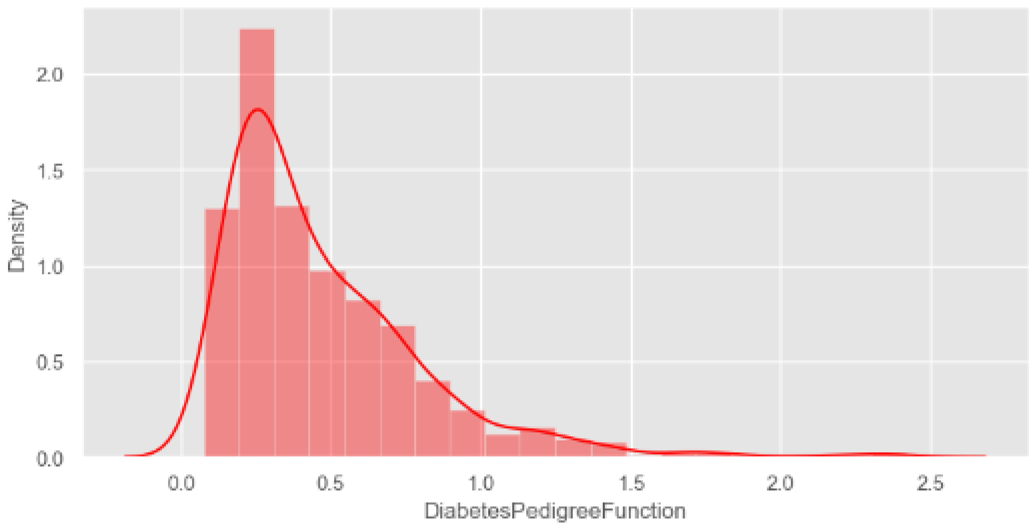
0.016
0.014
0.012
0.010
0.008
0.006
0.004
0.002
0.000
Density
0
50
100
150
200
Glucose



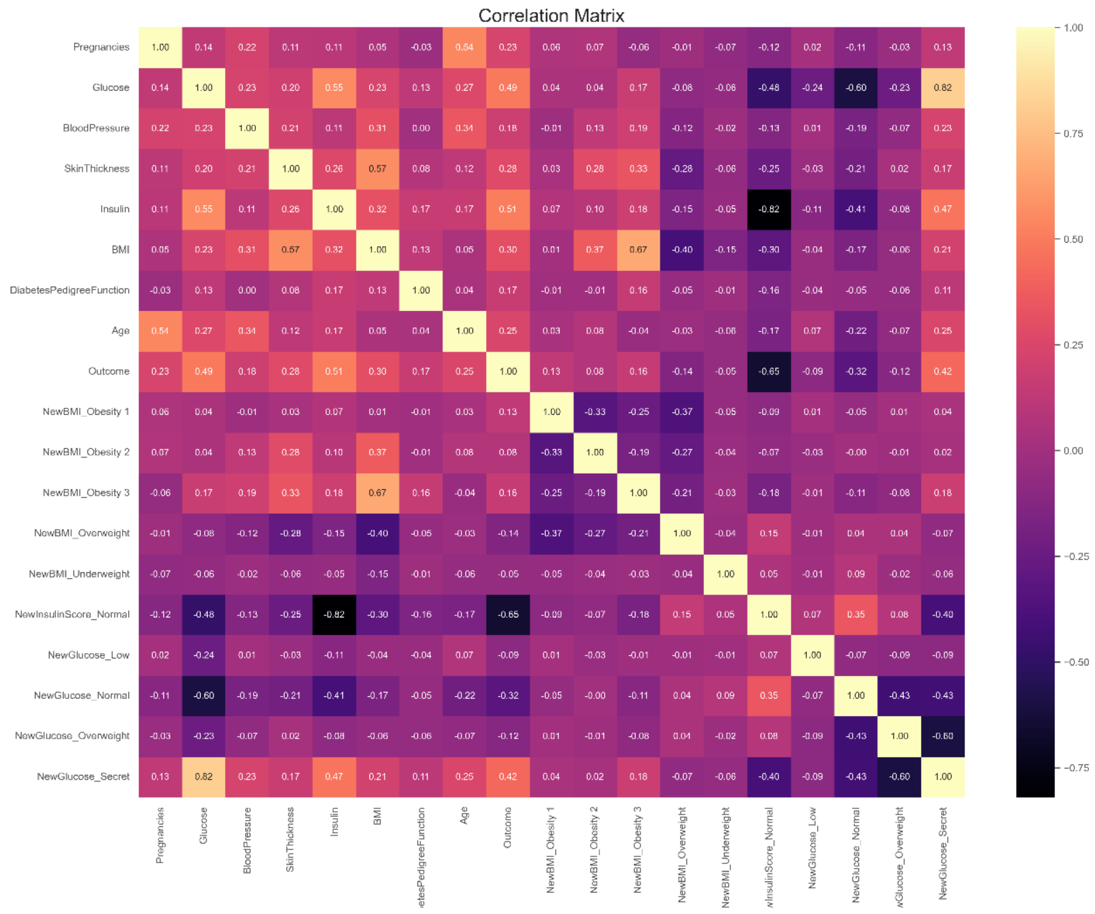
0.06
0.05
0.04
0.03
0.02
0.01
0.00
Density
0
20
40
60
80
100
SkinThickness


Density
0.030
0.025
0.020
0.015
0.010
0.005
0.000
−20
0
20
40
60
80
100
120
140
BloodPressure


Density
0.012
0.010
0.008
0.006
0.004
0.002
0.000
0
200
400
600
800
Insulin

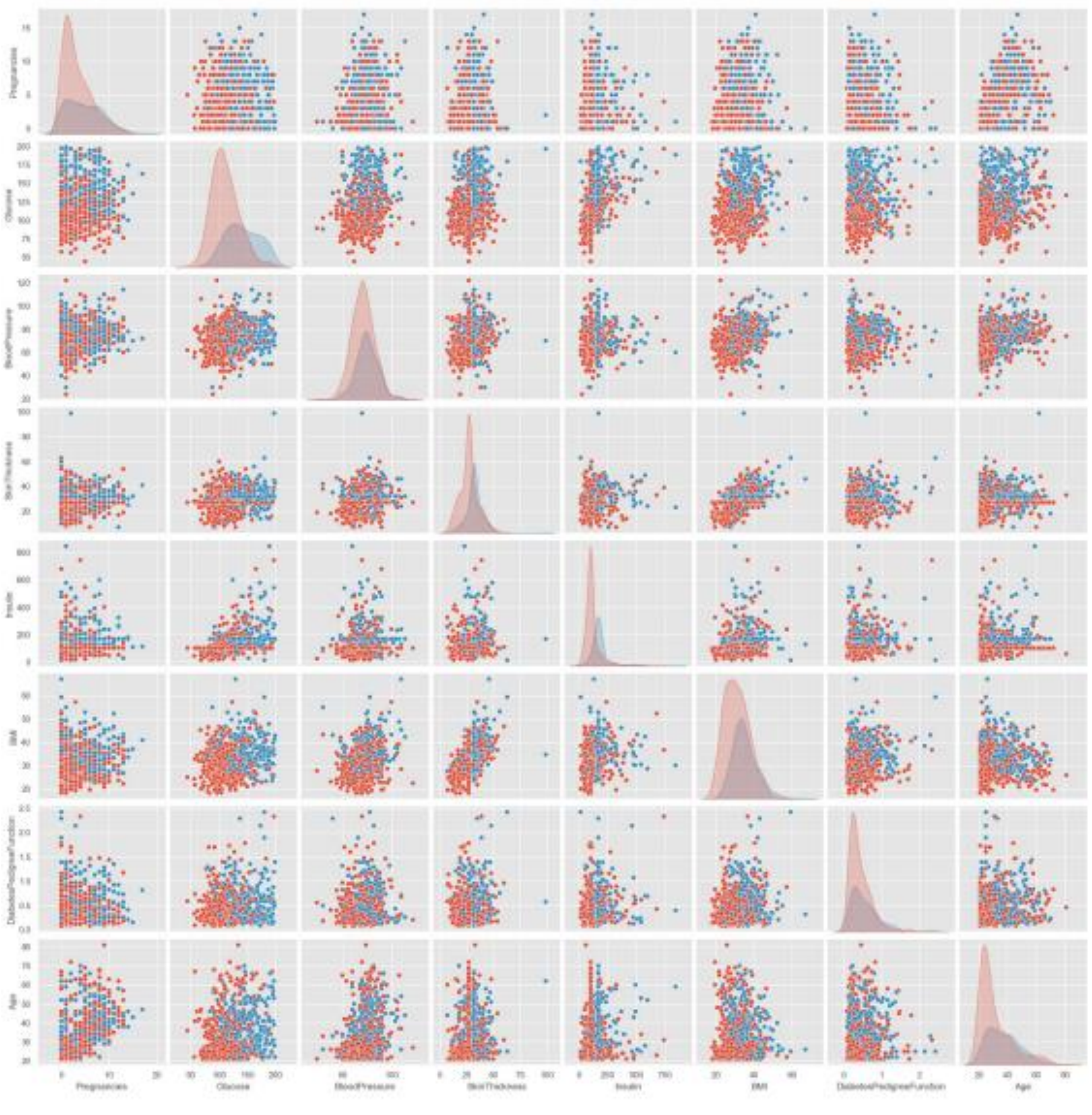

Density
2.0
1.5
1.0
0.5
0.0
0.0
0.5
1.0
1.5
2.0
2.5
DiabetesPedigreeFunction

3. CORRELATION MATRIX

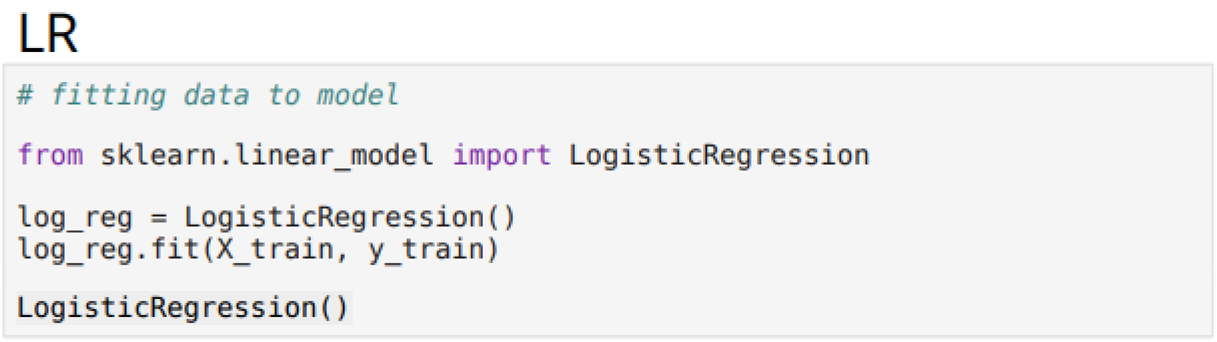


	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	etesPedigreeFunction	Age	Outcome	NewBMI_Obesity 1	NewBMI_Obesity 2	NewBMI_Obesity 3	NewBMI_Overweight	NewBMI_Underweight	wInsulinScore_Normal	NewGlucose_Low	NewGlucose_Normal	wGlucose_Overweight	NewGlucose_Secret
Pregnancies	1.00	0.14	0.22	0.11	0.11	0.05	-0.03	0.54	0.23	0.06	0.07	-0.06	-0.01	-0.07	-0.12	0.02	-0.11	-0.03	0.13
Glucose	0.14	1.00	0.23	0.20	0.55	0.23	0.13	0.27	0.49	0.04	0.04	0.17	-0.08	-0.06	-0.48	-0.24	-0.60	-0.23	0.82
BloodPressure	0.22	0.23	1.00	0.21	0.11	0.31	0.00	0.34	0.18	-0.01	0.13	0.19	-0.12	-0.02	-0.13	0.01	-0.19	-0.07	0.23
SkinThickness	0.11	0.20	0.21	1.00	0.26	0.57	0.08	0.12	0.28	0.03	0.28	0.33	-0.28	-0.06	-0.25	-0.03	-0.21	0.02	0.17
Insulin	0.11	0.55	0.11	0.26	1.00	0.32	0.17	0.17	0.51	0.07	0.10	0.18	-0.15	-0.05	-0.82	-0.11	-0.41	-0.08	0.47
BMI	0.05	0.23	0.31	0.57	0.32	1.00	0.13	0.05	0.30	0.01	0.37	0.67	-0.40	-0.15	-0.30	-0.04	-0.17	-0.06	0.21
DiabetesPedigreeFunction	-0.03	0.13	0.00	0.08	0.17	0.13	1.00	0.04	0.17	-0.01	-0.01	0.16	-0.05	-0.01	-0.16	-0.04	-0.05	-0.06	0.11
Age	0.54	0.27	0.34	0.12	0.17	0.05	0.04	1.00	0.25	0.03	0.08	-0.04	-0.03	-0.06	-0.17	0.07	-0.22	-0.07	0.25
Outcome	0.23	0.49	0.18	0.28	0.51	0.30	0.17	0.25	1.00	0.13	0.08	0.16	-0.14	-0.05	-0.65	-0.09	-0.32	-0.12	0.42
NewBMI_Obesity 1	0.06	0.04	-0.01	0.03	0.07	0.01	-0.01	0.03	0.13	1.00	-0.33	-0.25	-0.37	-0.05	-0.09	0.01	-0.05	0.01	0.04
NewBMI_Obesity 2	0.07	0.04	0.13	0.28	0.10	0.37	-0.01	0.08	0.08	-0.33	1.00	-0.19	-0.27	-0.04	-0.07	-0.03	-0.00	-0.01	0.02
NewBMI_Obesity 3	-0.06	0.17	0.19	0.33	0.18	0.67	0.16	-0.04	0.16	-0.25	-0.19	1.00	-0.21	-0.03	-0.18	-0.01	-0.11	-0.08	0.18
NewBMI_Overweight	-0.01	-0.08	-0.12	-0.28	-0.15	-0.40	-0.05	-0.03	-0.14	-0.37	-0.27	-0.21	1.00	-0.04	0.15	-0.01	0.04	0.04	-0.07
NewBMI_Underweight	-0.07	-0.06	-0.02	-0.06	-0.05	-0.15	-0.01	-0.06	-0.05	-0.05	-0.04	-0.03	-0.04	1.00	0.05	-0.01	0.09	-0.02	-0.06
NewInsulinScore_Normal	-0.12	-0.48	-0.13	-0.25	-0.82	-0.30	-0.16	-0.17	-0.65	-0.09	-0.07	-0.18	0.15	0.05	1.00	0.07	0.35	0.08	-0.40
NewGlucose_Low	0.02	-0.24	0.01	-0.03	-0.11	-0.04	-0.04	0.07	-0.09	0.01	-0.03	-0.01	-0.01	-0.01	0.07	1.00	-0.07	-0.09	-0.09
NewGlucose_Normal	-0.11	-0.60	-0.19	-0.21	-0.41	-0.17	-0.05	-0.22	-0.32	-0.05	-0.00	-0.11	0.04	0.09	0.35	-0.07	1.00	-0.43	-0.43
NewGlucose_Overweight	-0.03	-0.23	-0.07	0.02	-0.08	-0.06	-0.06	-0.07	-0.12	0.01	-0.01	-0.08	0.04	-0.02	0.08	-0.09	-0.43	1.00	-0.60
NewGlucose_Secret	0.13	0.82	0.23	0.17	0.47	0.21	0.11	0.25	0.42	0.04	0.02	0.18	-0.07	-0.06	-0.40	-0.09	-0.43	-0.60	1.00

4. SNS PAIR PLOT

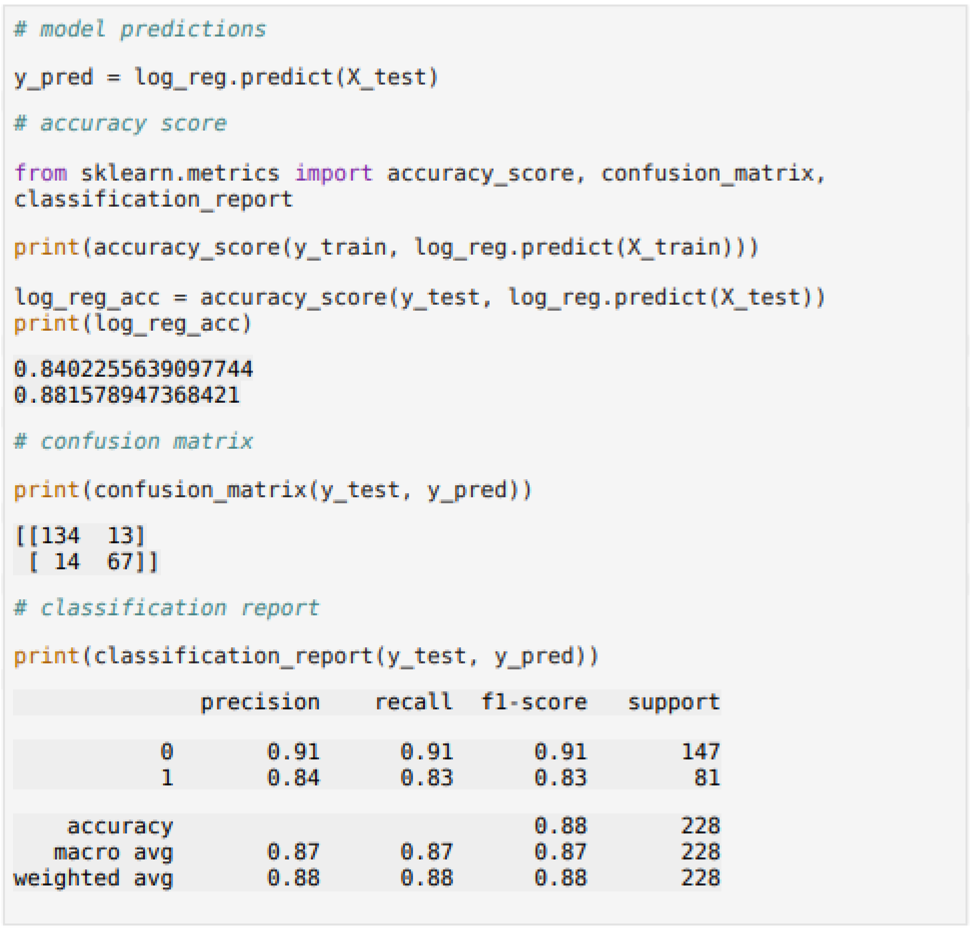


5. LINEAR REGRSSION



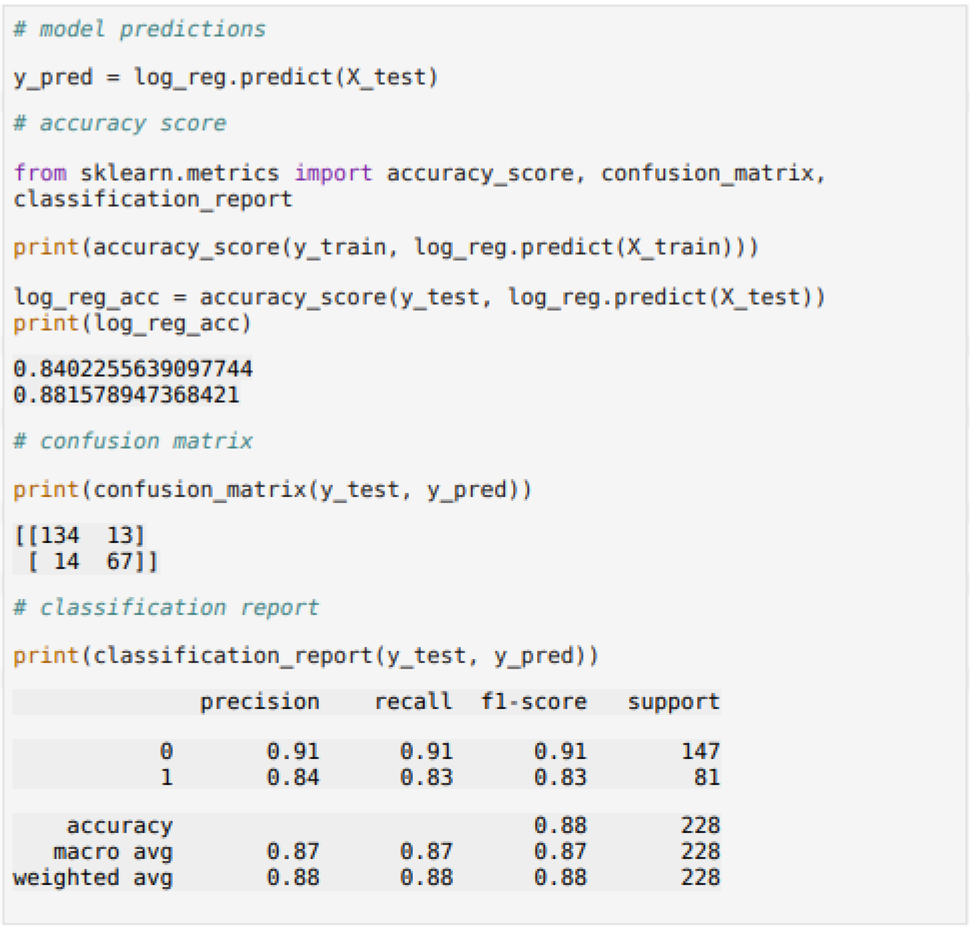
LR

```
# fitting data to model

from sklearn.linear_model import LogisticRegression

log_reg = LogisticRegression()
log_reg.fit(X_train, y_train)

LogisticRegression()
```



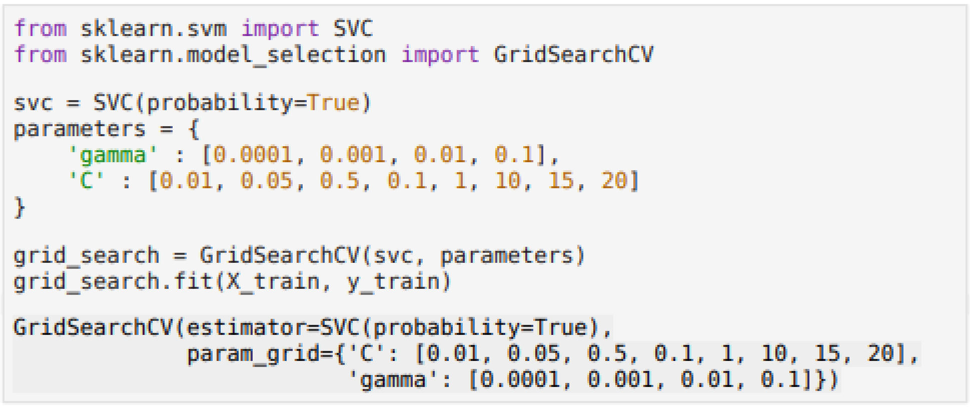
```
# model predictions

y_pred = log_reg.predict(X_test)

# accuracy score

from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report

print(accuracy_score(y_train, log_reg.predict(X_train)))

log_reg_acc = accuracy_score(y_test, log_reg.predict(X_test))
print(log_reg_acc)

0.8402255639097744
0.881578947368421

# confusion matrix

print(confusion_matrix(y_test, y_pred))

[[134  13]
 [ 14  67]]

# classification report

print(classification_report(y_test, y_pred))

              precision    recall  f1-score   support

           0       0.91      0.91      0.91       147
           1       0.84      0.83      0.83        81

    accuracy                           0.88       228
   macro avg       0.87      0.87      0.87       228
weighted avg       0.88      0.88      0.88       228
```

6. KNN



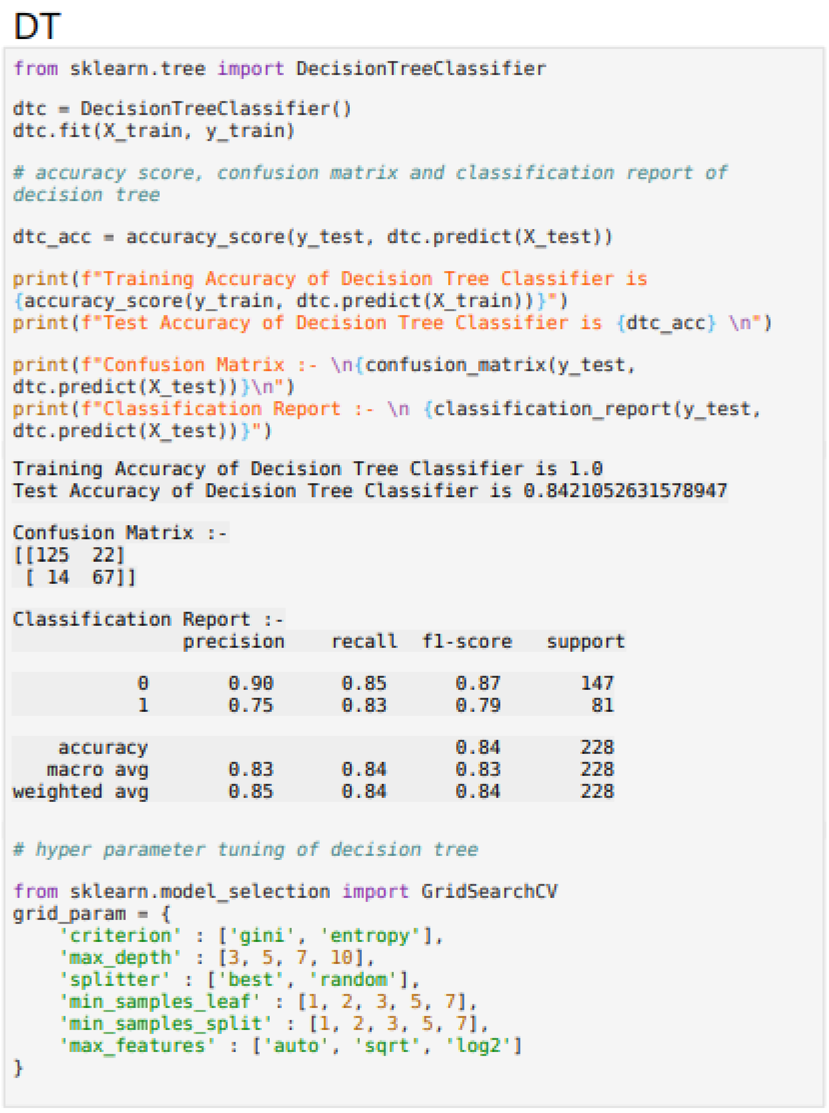
```
# model predictions

y_pred = log_reg.predict(X_test)

# accuracy score

from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report

print(accuracy_score(y_train, log_reg.predict(X_train)))

log_reg_acc = accuracy_score(y_test, log_reg.predict(X_test))
print(log_reg_acc)

0.8402255639097744
0.881578947368421

# confusion matrix

print(confusion_matrix(y_test, y_pred))

[[134  13]
 [ 14  67]]

# classification report

print(classification_report(y_test, y_pred))

              precision    recall  f1-score   support

           0       0.91      0.91      0.91       147
           1       0.84      0.83      0.83        81

    accuracy                           0.88       228
   macro avg       0.87      0.87      0.87       228
weighted avg       0.88      0.88      0.88       228
```

7. SVM



```
from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV

svc = SVC(probability=True)
parameters = {
    'gamma' : [0.0001, 0.001, 0.01, 0.1],
    'C' : [0.01, 0.05, 0.5, 0.1, 1, 10, 15, 20]
}

grid_search = GridSearchCV(svc, parameters)
grid_search.fit(X_train, y_train)

GridSearchCV(estimator=SVC(probability=True),
             param_grid={'C': [0.01, 0.05, 0.5, 0.1, 1, 10, 15, 20],
                         'gamma': [0.0001, 0.001, 0.01, 0.1]})
```

8. DT



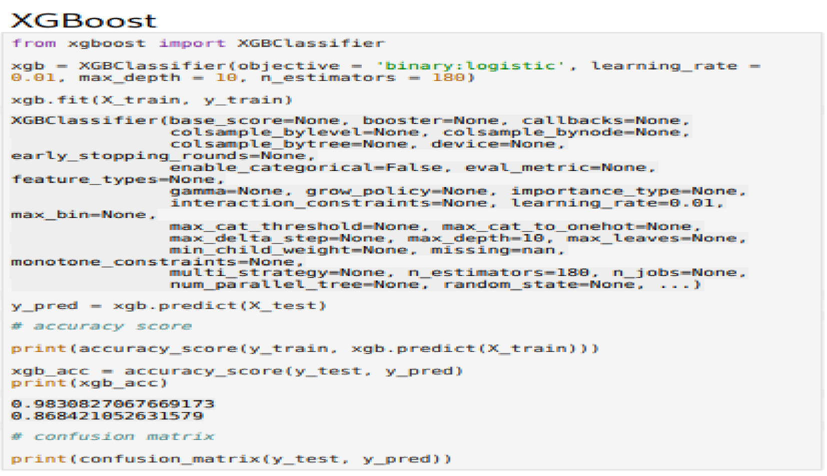
DT

```
from sklearn.tree import DecisionTreeClassifier

dtc = DecisionTreeClassifier()
dtc.fit(X_train, y_train)

# accuracy score, confusion matrix and classification report of
decision tree

dtc_acc = accuracy_score(y_test, dtc.predict(X_test))

print(f"Training Accuracy of Decision Tree Classifier is
{accuracy_score(y_train, dtc.predict(X_train))}")
print(f"Test Accuracy of Decision Tree Classifier is {dtc_acc} \n")

print(f"Confusion Matrix :- \n{confusion_matrix(y_test,
dtc.predict(X_test))}\n")
print(f"Classification Report :- \n {classification_report(y_test,
dtc.predict(X_test))}")
```

```
Training Accuracy of Decision Tree Classifier is 1.0
Test Accuracy of Decision Tree Classifier is 0.8421052631578947

Confusion Matrix :-
[[125  22]
 [ 14  67]]

Classification Report :-
               precision    recall  f1-score   support

           0       0.90      0.85      0.87       147
           1       0.75      0.83      0.79        81

    accuracy                           0.84       228
   macro avg       0.83      0.84      0.83       228
weighted avg       0.85      0.84      0.84       228
```

```
# hyper parameter tuning of decision tree

from sklearn.model_selection import GridSearchCV
grid_param = {
    'criterion' : ['gini', 'entropy'],
    'max_depth' : [3, 5, 7, 10],
    'splitter' : ['best', 'random'],
    'min_samples_leaf' : [1, 2, 3, 5, 7],
    'min_samples_split' : [1, 2, 3, 5, 7],
    'max_features' : ['auto', 'sqrt', 'log2']
}
```

9. XGBOOST



XGBoost

```
from xgboost import XGBClassifier

xgb = XGBClassifier(objective = 'binary:logistic', learning_rate =
0.01, max_depth = 10, n_estimators = 180)

xgb.fit(X_train, y_train)
```

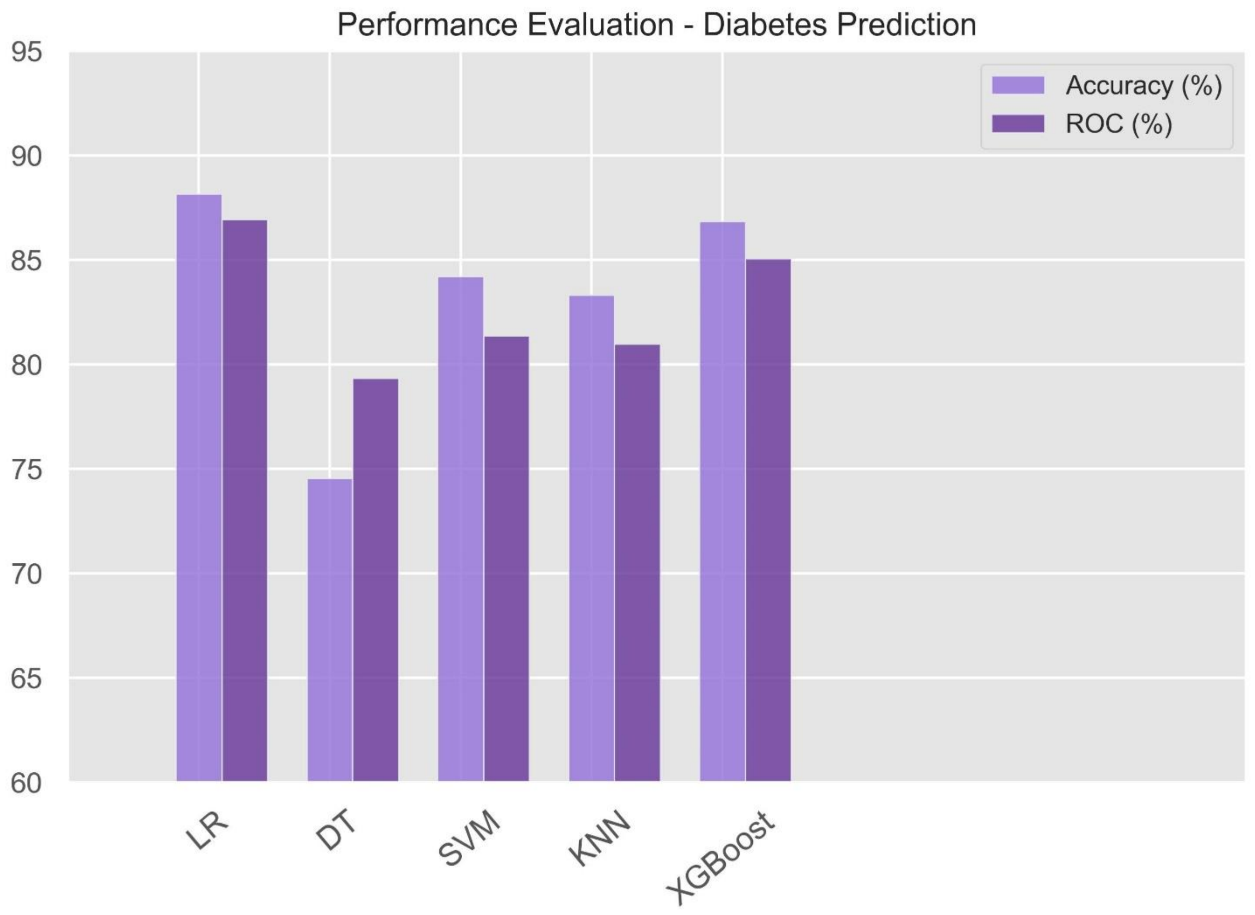
```
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device=None,
early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None,
feature_types=None,
              gamma=None, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=0.01,
max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=10, max_leaves=None,
              min_child_weight=None, missing=nan,
monotone_constraints=None,
              multi_strategy=None, n_estimators=180, n_jobs=None,
              num_parallel_tree=None, random_state=None, ...)
```

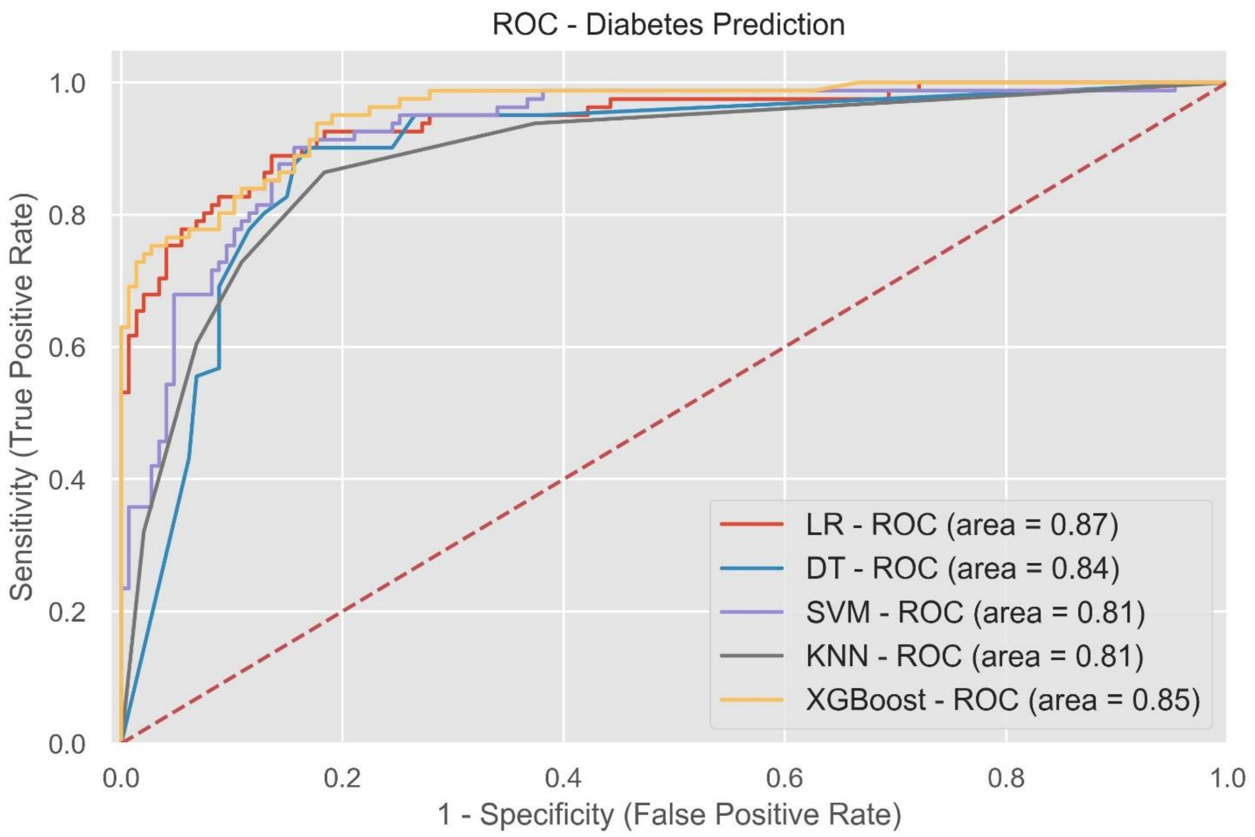
```
y_pred = xgb.predict(X_test)

# accuracy score

print(accuracy_score(y_train, xgb.predict(X_train)))

xgb_acc = accuracy_score(y_test, y_pred)
print(xgb_acc)
```

```
0.9830827067669173
0.868421052631579
```

```
# confusion matrix

print(confusion_matrix(y_test, y_pred))
```

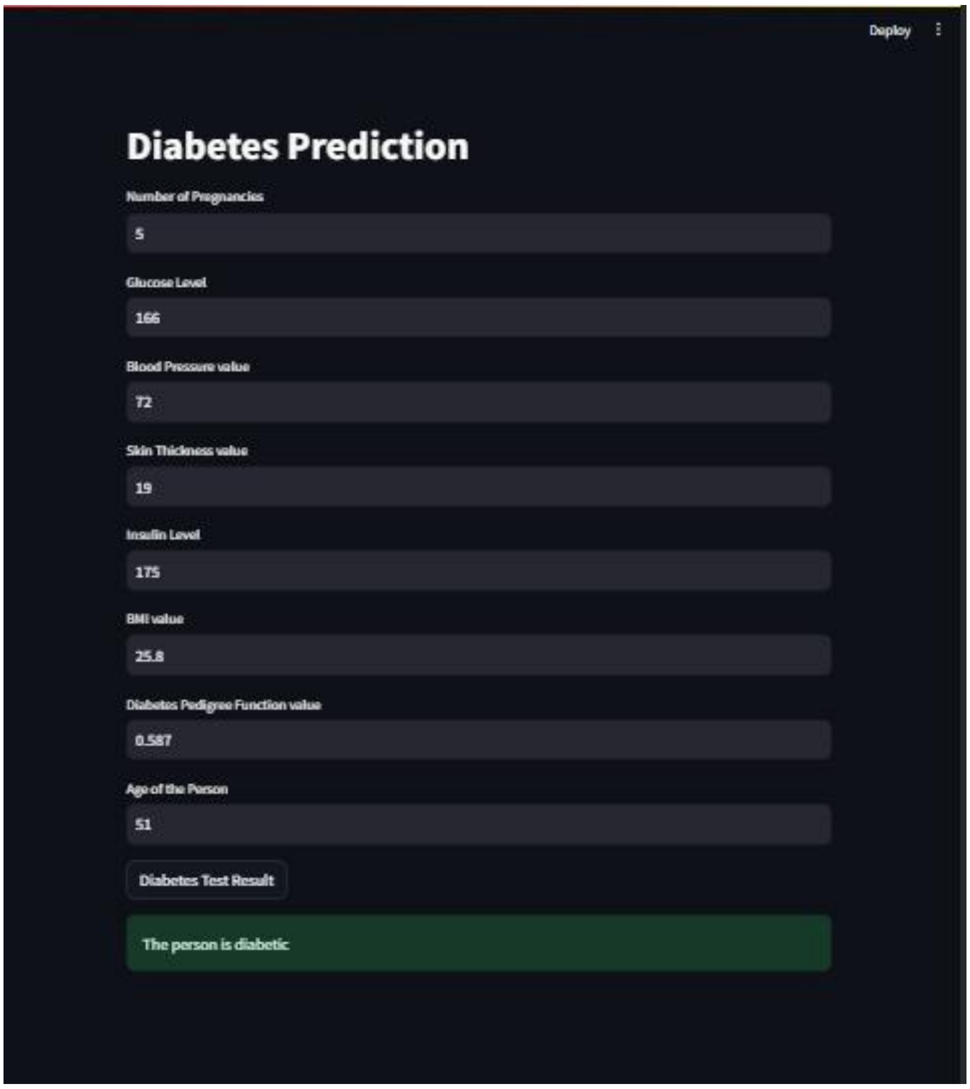
Model Comparison





OUTPUT SCREENSHOT

POSITIVE SCREENSHOT:-



Diabetes Prediction

Number of Pregnancies

5

Glucose Level

166

Blood Pressure value

72

Skin Thickness value

19

Insulin Level

175

BMI value

25.8

Diabetes Pedigree Function value

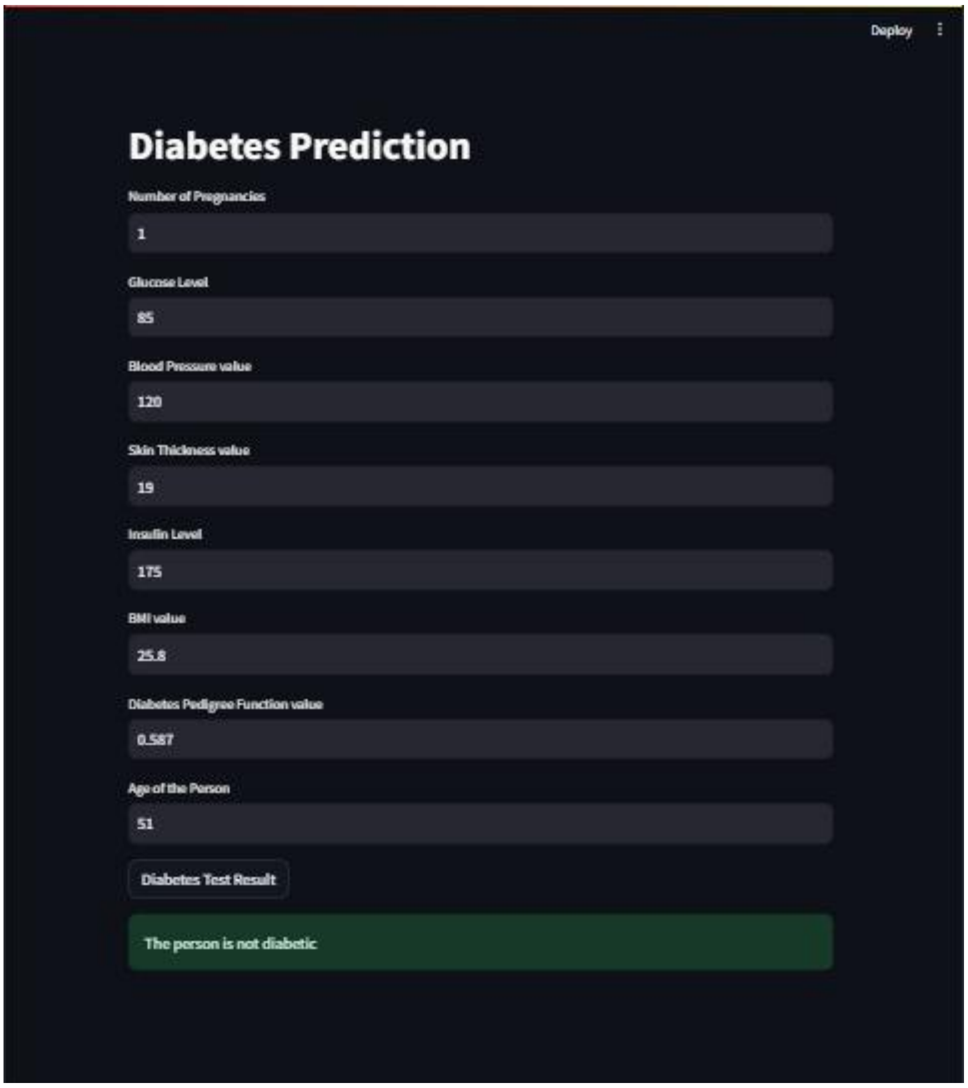
0.587

Age of the Person

51

The person is diabetic

NEGATIVE SCREENSHOT:-



Diabetes Prediction

Number of Pregnancies

1

Glucose Level

85

Blood Pressure value

120

Skin Thickness value

19

Insulin Level

175

BMI value

25.8

Diabetes Pedigree Function value

0.587

Age of the Person

51

Diabetes Test Result

The person is not diabetic

Github Link of the project :-

Future scope:-

Healthcare professions found it hard to find healthcare data and perform analysis on them due to lack of tools, resources. But using ML, we can overcome this and can perform analysis on real-time data leading to better modeling, predictions. This enhances and improves overall healthcare services. Now, IoTs being integrated with ML in order to make smart healthcare devices that sense if there is any change in the person's body, health data when he uses the device (Pacemaker, Stethoscope, etc.) and this will notify the person regarding this through an app. This helps in easy monitoring, advanced prediction and analysis thereby reducing errors, saving time and life of people.

Machine learning has the great ability to revolutionize the diabetes prediction with the help of advanced computational methods.

- Detection of Diabetes in its early stage is the key for treatment.
- The technique may also help researchers to develop an accurate and efficient tool that will reach at the table of clinicians to help them make better decisions about the disease.
- More parameters and factors would be involved in the future scope of this project.
- The accuracy will increase even more when the parameters increase.
- Using traditional techniques and algorithms, we can enhance the accuracy by improving the data.