Diabetes Prediction Based On Lifestyle

The project is to determine whether the patient has diabetes or not based on their lifestyle. The is the classification problem of supervised machine learning.

Dataset:-
The dataset that we have consist of 9 columns which are-

1. Pregnancies :- tell about the number of pregnancies
2. Glucose:- Amout of Plasma Glucose concentration
3. BloodPressure:- BloodPressure of a patient(mm Hg)
4. SkinThickness:- Triceps skinfold thickness (mm)
5. Insulin:-2-Hour serum insulin (mu U/ml)
6. BMI:- Body Mass Index
7. Diabetes pedigree function`:-` Diabetes pedigree function
8. Age:- age of a patient
9. Outcome:- whether the person has diabetes or not (0 denotes person has not diabetes whereas 1 denotes the person has diabetes)

In the dataset we have a data of 768 person. If we divide the data into the target outcomes then there are 500 data of person not having diabetes and 268 data of person has diabetes.

First import all the necessary libraries in the notebook like pandas, numpy, matplotlib and then import the dataset in the notebook.
Check the necessary detail about the dataset like columns in dataset(**df.columns**) , shape of the dataset(**df.shape**) and take a brief description of data by **describe()** function in pandas, and checking the null values in data. In this dataset there is no null values.

For seeing the more details of the data we visualize the data.we will create graphs to displays different distributions of the features and available relationships that allow us to understand about the data much better.
Firstly I use the pairplot in seaborn library to visualize the relationship between different variables in the dataset, then I use the hist in seaborn to see the distribution of each feature in the dataset. After that we visualize the distribution of all the feature with respect to outcome of whether the patient has diabetes or not.

Then we visualize the distribution of outcome where we see that there are more data that belong to the patient that has not diabetes.

Next, we will proceed in checking the relationships between different features by visualizing correlations for that we use the heatmap of seaborn library

"***fig= plt.figure(figsize=(10,7))***
sns.heatmap(data=df_data_1.corr(),cmap="Greens",annot=True)"

And then visualize how the other feature is related to the target feature "Outcome" and visualize this by barplot

"***c=pd.DataFrame(df_data_1.corr()['Outcome'])***
c=c.sort_values(['Outcome'],ascending=True)
sns.barplot(y=c.index,x=c['Outcome'])"

from the output we get it is clearly be seen that the Outcome is more coorelated with Glucose and BMI and least coorelated with BloodPressure and SkinThickness.

After the visualization done We check the zero value in the dataset from that we conclude:-

number of rows missing Pregnancies : 111
number of rows missing glucose : 5
number of rows missing in BloodPressure: 35
number of rows missing in SkinThickness : 227
number of rows missing in Insulin : 374
number of rows missing in BMI : 11
number of rows missing in DiabetesPedigreeFunction : 0
number of rows missing in Age : 0

This zero value make our model not efficient. We therefore, need to impute the zero values by using the mean of the other values in the same column
so we replace al the zero values in the dataset by the mean of that particular column by the function called SimpleImputer from the sklearn.impute library.

"missing=SimpleImputer(missing_values=0,strategy="mean")"

Next step is to rescale the features so rescale all the feature in the specified range of 0 and 1.

Divide the data into the input and the target variable. so x varible contain the input features and y variable contain the output features.

Then divide our data into train and test data such that 20% of the data is in test data while 80% of the data is in train data. we have to divide the train and test data so that it is uniformly distrbuted on the target value.

"X_train,X_test,y_train,y_test=train_test_split(x,y,test_size=0.20,stratify=df["Outcome"])"

After spliting the dataset we see the distribution of the data in train and test split .

"***Train_dia=len(y_train[y_train[:]==1])***
Train_non_dia =len(y_train[y_train[:]==0])
Test_dia=len(y_test[y_test[:]==1])
Test_non_dia =len(y_test[y_test[:]==0])
print("number of person having Diabeties :",dia)
print("number of person not having Diabeties :",non_dia)
print("number of person having Diabeties in training dataset :",Train_dia)
print("number of person not having Diabeties in training dataset :",Train_non_dia)
print("number of person having Diabeties in testing dataset:",Test_dia)
print("number of person not having Diabeties in testing dataset :",Test_non_dia)"

from above code we see that

number of person having Diabeties : 268
number of person not having Diabeties : 500
number of person having Diabeties in training dataset : 214
number of person not having Diabeties in training dataset : 400

number of person having Diabeties in testing dataset: 54
number of person not having Diabeties in testing dataset : 100

So our model is uniformly distributed.

After all that data processing we train our model using the classification algorithm I choose the Logistic Regression for this problem. Fit our training data in the model.

Then, evaluating the model the accuracy of the model is 79.22%

We can Predict the outcome for the input or testing dataset and see the result.

After that next step is to evaluate our model and visualize the confusion matrix for our model prediction.

From the confusion matrix that is visualize in the notebook we see that:-
90 values are predicted 1 which is actually 1
22 values which is predicted 1 is actually 0
10 vaues which is predicted 0 is actually 1
32 values which is predicted is actually 0

So our model is build which give an accuracy of 79.22%

At last I conclude this was a very interesting project and it was helpful for me in understanding the basic concepts of machine learning in diabetes prediction.