**Diabetes Project-Articles**

**Some information about Diabeties:-**

Diabetes mellitus (DM), commonly known as diabetes, is a group of metabolic disorders characterized by a high blood sugar level over a prolonged period of time.

Symptoms often include:

1. Frequent urination
2. Increased thirst
3. Increased appetite

If left untreated, diabetes can cause many complications

Acute complication can include **diabetic ketoacidosis** (excess Blood acids), **hyperosmolar** **hyperglycemic** **state** , or **death**.

Serious long-term complications include cardiovascular disease, stroke, chronic, kidney disease, foot ulcers, damage to the nerves, damage to the eyes and cognitive impairment

**1.Problem Defination :**

Gestational diabetes is the third main form , and occurs when pregnant women without a previous history of diabetes develop high blood suger level

We have a dataset which contains the number of pregnancy of that particular patient, and will try to identify based on others information about the patient whether the person or patient is diabetic or not diabetic.

**2.Data Analysis:**

Data Analysis is the technique to collect, transform and organize data to make future predictions, and make informed data – driven decisions, it also helps to find possible solutions for a business.

Firstly

* Importing the necessary modules – pandas and numpy

Pandas – it is use for data manipulation

Nupmy – it is use for data calculation

* Collect the datasets
* Check the type of dataset in which form of the data sets is available
* How my datasets look like
* Find the shape of the dataset like how many rows and columns is available in the dataset

* What are columns available in our dataset.
* head – it is use to check first 5 lines dataset
* tail – it is use to check last 5 lines of dataset
* sample – it is use to check random number of sample
* Find the target variable
* Checking the null values in the complete dataset – Since, there are no null values in the data set
* Imputing the datasets –

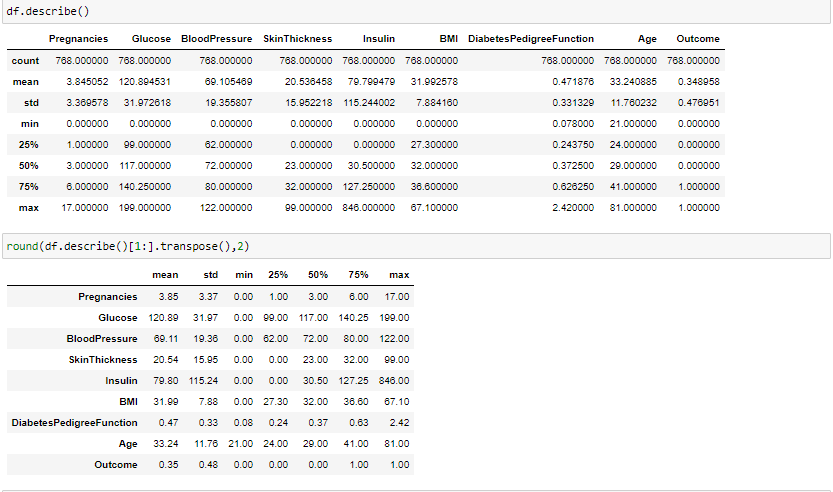
Categorical columns – mode/frequent

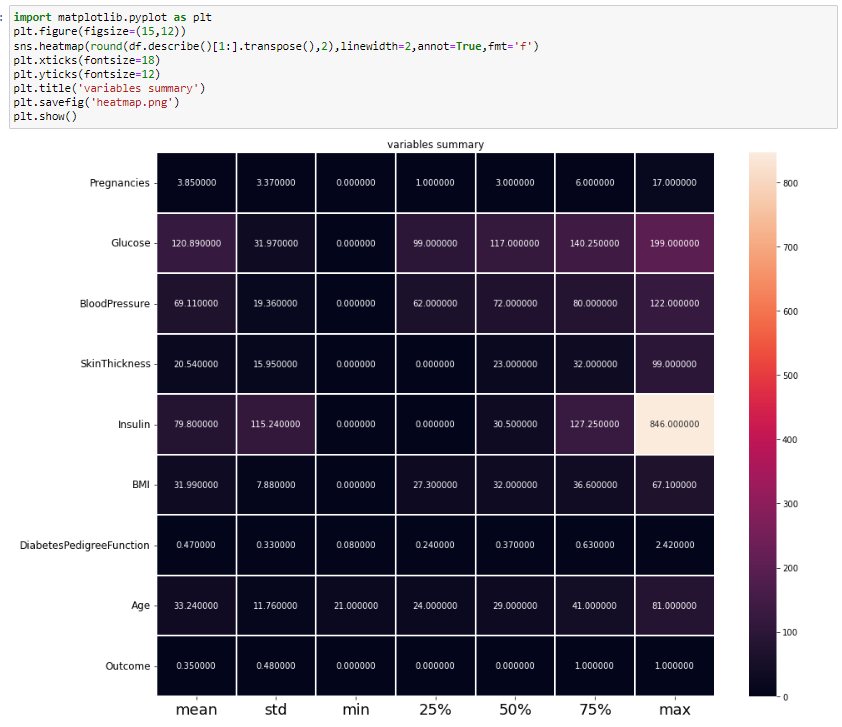
Continuous columns - mean / medium

* Check duplicates in the dataset – According to my ovservation, there are no duplicates values in the dataset
* Information about the dataset.

**3.EDA Concluding Remarks:**

* Descriptive Analytics – Describe the dataset to know more about the dataset





Remarks = **Outcome of the describe the dataset:**

from the above plotting we are determining mean,standard deviation,minimum and maximum value of each column, it helps us further in data cleaning.

Total Number of rows in the datasets.

1. 768 rows
2. 9 columns

Pregnancies:

1. Mean = 3.845052
2. Std = 3.369578
3. Max\_value = 17.000000
4. Min\_value = 0

Glucose:

1. Mean = 120.894531
2. Std = 31.972618
3. Max\_value = 199.000000
4. Min\_value = 0

BloodPressure:

1. Mean = 69.105469

2. Std = 19.355807

3. max\_value = 122.000000

4. min\_value = 0.000000

SkinThickness:

1.Mean = 79.799479

2. Std = 115.244002

3. max\_value = 846.000000

4. min\_value = 0.000000

BMI:

1.mean = 31.992578

2. std = 7.884160

3. max\_value = 67.100000

4. min\_value = 0

DiabetesPedigreefunction:

1. Mean = 0.471876
2. Std = 0.331329
3. Max\_value = 2.420000
4. Min\_value = 0.078000

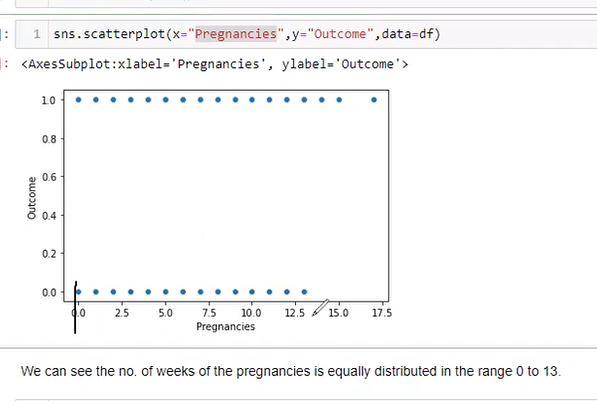
Age : This comes under categorical data

1. Max\_values= 81
2. Min\_values = 21
3. 25% data = 24
4. 50% data = 29
5. 75% data = 41

* Null values Analytics - here from the code we first checked that is there any null values from the isnull() function then we are going to take the sum of all those missing values from the sum() function and the inference we now get is that there are no missing values .
* Visualization of data - We carry out by bivariate analysis, using scatterplot and the columns is which we are using , outcome is our target column

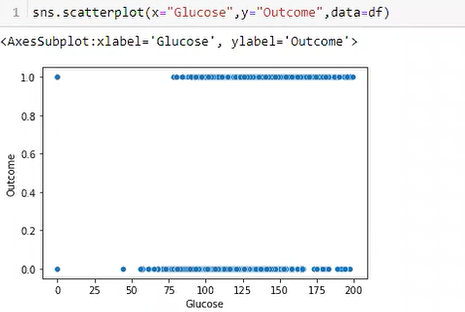
**Scatterplot:**

1. sns.scatterplot(x=‘Pregnancies’ ,y=‘outcome’,data=df)



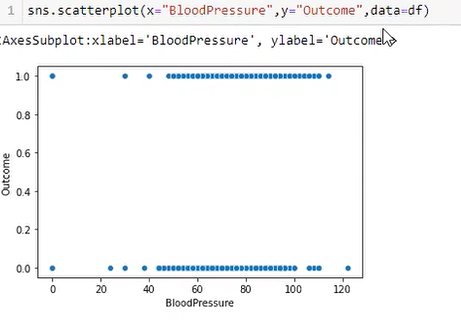
Remarks = we can see the no. of weeks of the pregnancies is equally distributed in the range 0 to 13.

1. sns.scatterplot(x=‘Glucose ,y=‘outcome’,data=df)



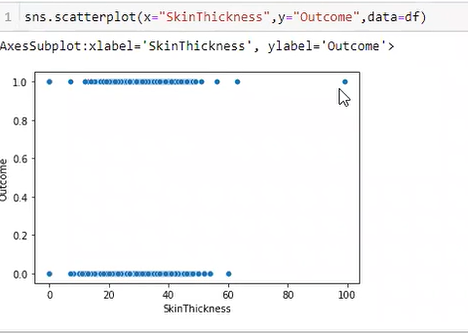
Remarks = we can see the Glucose is equally distributed in the range 75 to 200.

1. sns.scatterplot(x=‘BloodPressure ,y=‘outcome’,data=df)



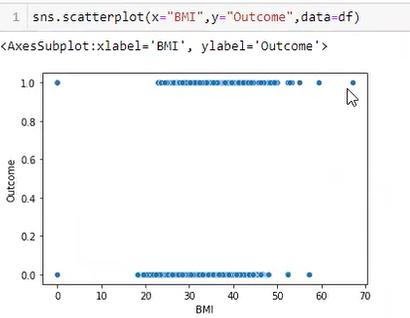
Remarks = we can see the BloodPressure is equally distributed in the range 45 to 100.

1. sns.scatterplot(x=‘SkinThickness ,y=‘outcome’,data=df)



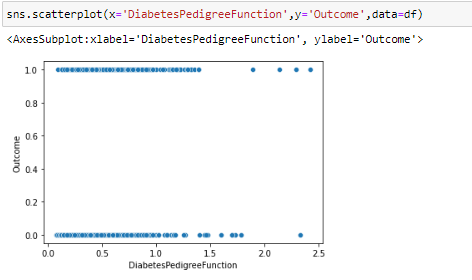
Remarks = we can see the SkinThickness is equally distributed in the range 5 to 55.

1. sns.scatterplot(x=‘BMI ,y=‘outcome’,data=df)



Remarks = we can see the BMI is equally distributed in the range 25 to 50.

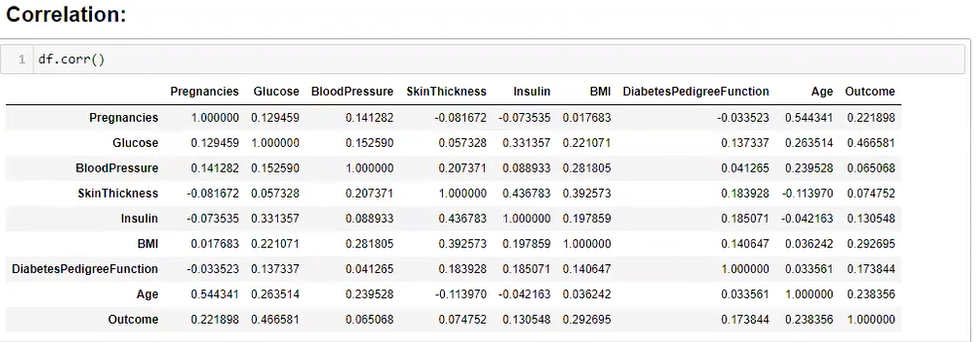
6. sns.scatterplot(x=‘DiabetesPedigreeFunction’ ,y=‘outcome’,data=df)

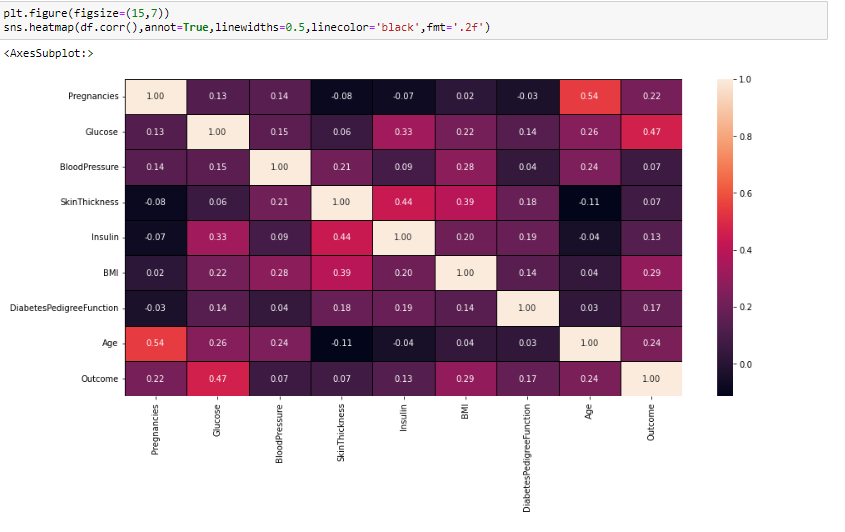


Remarks = we can see the DiabetesPedigreeFunction is equally distributed in the range 0 to 1.25.

* **Correlations** :

Find the relationship between columns





Remarks = **Outcome of Correlation :**

All the columns of the datasets is positively correlated with the target column.

1. Pregnancies has 22 percent correlation with target column which can be considered as a good bond
2. Glucose has 47 percent correlation with the target column which can be considered as a strong bond
3. Blood Pressure has 7 percentage correlation with the target column which can be considered as a weak bond
4. Skin thickness has 7 percent correlation with the target column which can be considered as a week bond.
5. Insulin has 13 percent correlation with the target column which can be considered as a good bond.
6. BMI has 29 percent correlation with the target column which can be considered as a good bond
7. DiabeticPedgreeFunction has 17 percent correlation with the target column which can be considered as a good bond
8. Age has 24 percent percent correlation with the target column which can be considered as a good bond

* df.info() – this code basically gives you information about class,data types, Data columns, memory usage, index range
* **Skewness:**

Check the skewness df.skew()

**Outcome of the skewness:** skewness threshold is taken is +/-0.65. columns which are having skewness:

Pregnancies

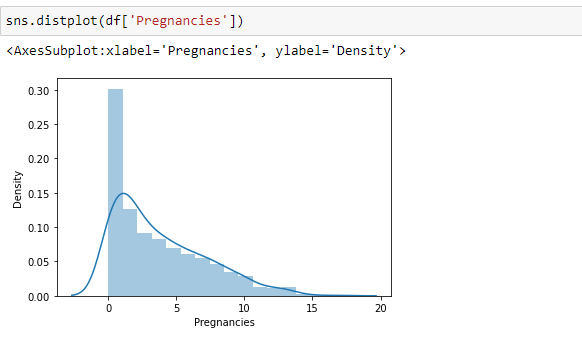
BloodPressure

Insulin

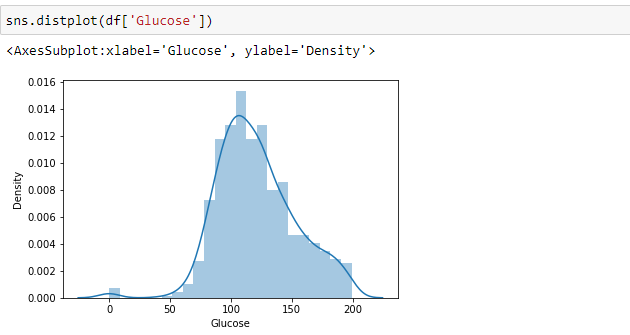
DiabetesPedigreeFunction

Age

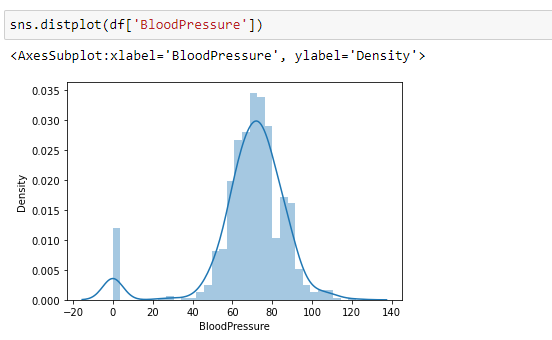
Normal Distribution Curve:



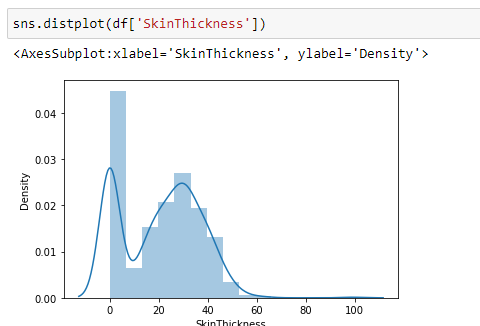
Remarks = the data of the columns is not normalised. The building blocks is out of the normalised curve



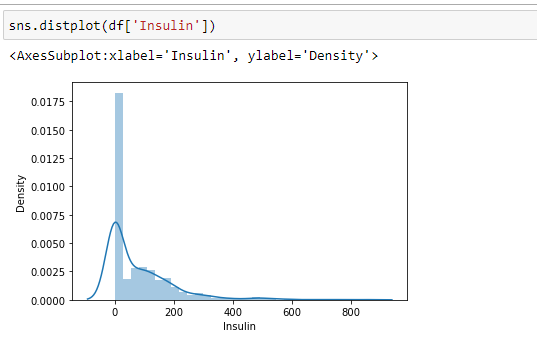
Remarks = the data of the columns is not normalised. The building blocks is out of the normalised curve



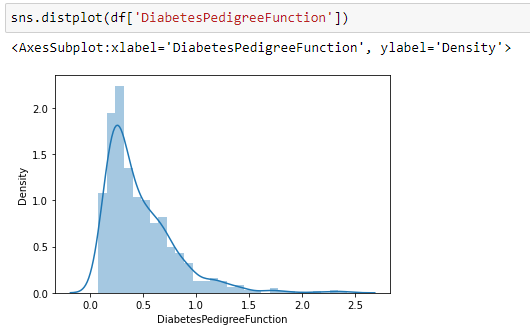
Remarks = the data of the columns is not normalised. The building blocks is out of the normalised curve

****

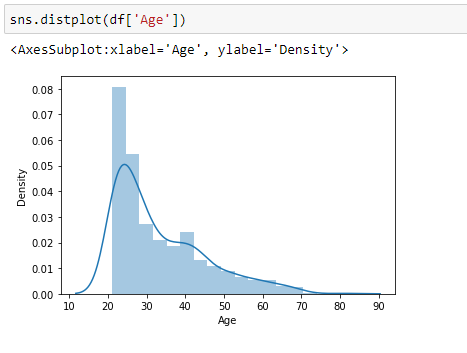
Remarks = the data of the columns is not normalised. The building blocks is out of the normalised curve, the plot is almost bimodal.



Remarks = the data of the columns is not normalised. The building blocks is out of the normalised curve.

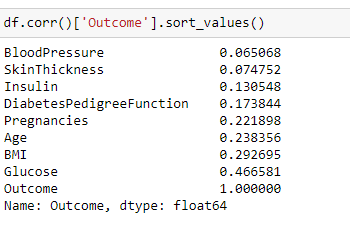


Remarks = the data of the columns is not normalised. The building blocks is out of the normalised curve



Remarks = the data of the columns is not normalised. The building blocks is out of the normalised curve

**Data Cleaning : -**

****

Remarks –as we can see that very low correlations columns Blood Pressure and Skin Thickness with outcome target variable, so we will drop these columns.

**Removing outliers by z-score technique**

**4.Pre-processing Pipeline :**

Transforming data to remove skewness:

From sklearn.preprocessing import power\_transform

Whenever you will apply preprocessing technique then your data will be getting output in numpy array

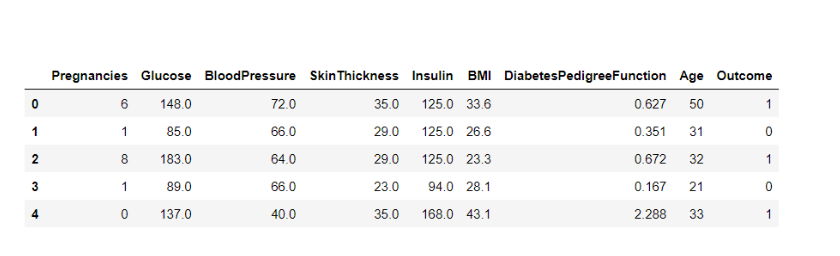
**Balancing the outcome values by smote modules**

From imblearn.over\_sampling import SMOTE

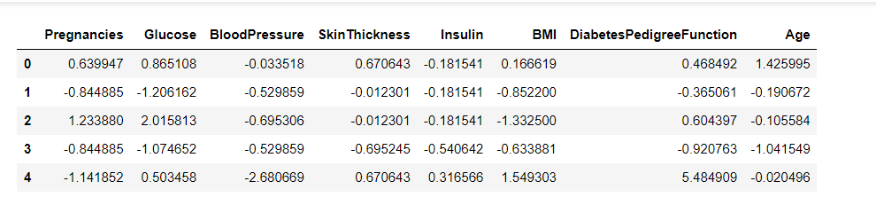
**Scaling the dataset:**

Using Standard scaling:

Lets see before using scaling technique:



After scaling the dataset looks like:



That’s how our dataset will be looking like when it is scaled down or we can see every value now is on the same scale which will help our ML model to give a better result.

**5.Building Machine Learning Models:**

**Spliting the data into testing and training dataset by using train\_test\_split function**

X = df.drop(‘outcome’, axis =1)

Y = df[‘Outcome’]

Importing necessary libraries:

From sklearn.model\_selection import train\_test\_split

x\_train,y\_train,x\_test,y\_test=train\_test\_split(x,y,test\_size=0.33,random\_state=7)

**Now we will using deferent types of models:**

From sklearn.linear\_model import LogisticRegression

Lr = LogisticRegression()

from sklearn.metrics import accuracy\_score,classification report,confusion\_matrix

from sklearn.ensemble import RandomForestClassifier

from sklearn.tree import decisionTreeClassifier

from xgboost import XGBClassifier

from sklearn.svm import SVC

**LogisticRegression:**

After using LogistocRegression – accuracy\_score = 75.10

**RandomForestClassifier:**

Using RandomForestClassifier – accuracy score = 0.76

**DecisionTreeClassifier:**

Using decisionTreeClassifier – accuracy\_score = 0.73

**XGBClassifier**:

Using XGBClassifier - accuracy\_score0.74

**Support Victor Classifier (SVC):**

Using svc = accuracy\_score = 0.74

Therefore Randome forest is the best model for this prediction since it has an accuracy\_socre of 0.76

Saving model: - Random Forest

Import pickle

Firstly we will be using the dump()function to save the model using pickle

Then we will be loading that saved model

Lastly after loading that model we will use this to make predictions

**6.Concluding Remarks:-**

**Conclusion:**

After using all these patient records, we are able to build a machine learning model (random forest - best one) to accurately predict whether or not the patients in the dataset have diabetes or not along with that we were able to draw some insights from the data via data analysis and visualization.