**Phase-2**

**Data Preprocessing**

|  |  |
| --- | --- |
| Date | 10-10-2023 |
| Team ID |  |
| Project Name | AI Based Diabetes Prediction. |
| Maximum Mark |  |

Data preprocessing is an important step in the data mining process. It refers to the cleaning, transforming, and integrating of data in order to make it ready for analysis. The goal of data preprocessing is to improve the quality of the data and to make it more suitable for the specific data mining tasks.

**Program:**

**#Import package:**

Numpy , pandas , sklearn , matplotlib.pyplot , Seaborn.

**Explanation:**

* Numpy :(import numpy as np) a library for mathematical operations and handling arrays.
* pandas :(import pandas as pd) a library for data manipulation and analysis.
* Matplotlib.pyplot: (import as plt) a library for creating visualization.
* Seaborn : as a library for creating additional data visualization.
* mlxtend.frquent\_paterns: a module for performing frequent itemset
* mining and association rule learning.
* Sklearn:( preproccesing and evaluate model )

**Code:**

import numpy as np

import pandas as pd

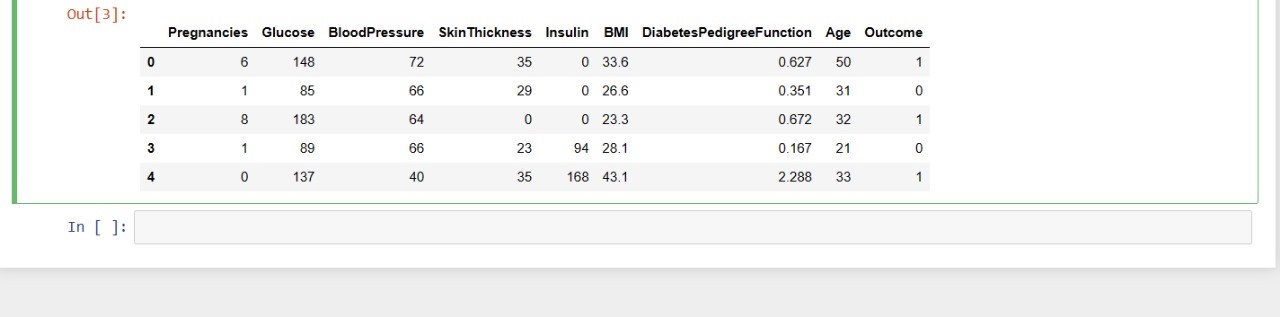
from sklearn.preprocessing import StandardScaler , Normalizer

from sklearn.compose import make\_column\_transformer, make\_column\_selector

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

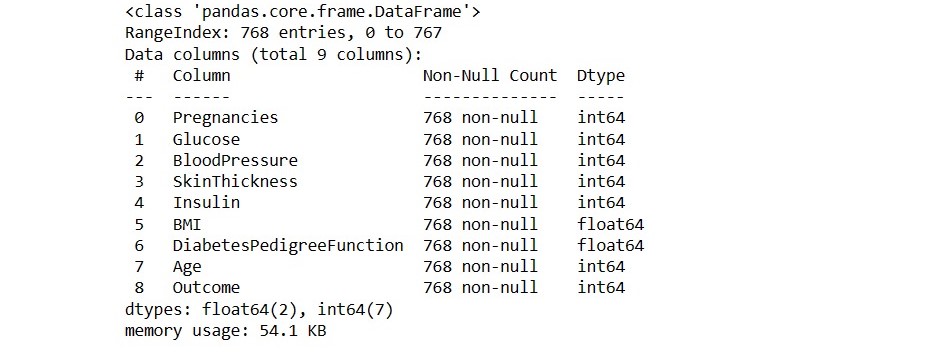
dataset = pd.read\_csv('C:/Users/91638/Documents/diabetes.csv')

dataset.head()



The sklearn.preprocessing package provides several common utility functions and transformer classes to change raw feature vectors into a representation that is more suitable for the downstream estimators. In general, many learning algorithms such as linear models benefit from standardization of the data set. Standardization of datasets is a common requirement for many machine learning estimators implemented in scikit-learn; they might behave badly if the individual features do not more or less look like standard normally distributed data: Gaussian with zero mean and unit variance . The StandardScaler utility class is a quick and easy way to perform standardization on an array-like dataset .

dataset.info()

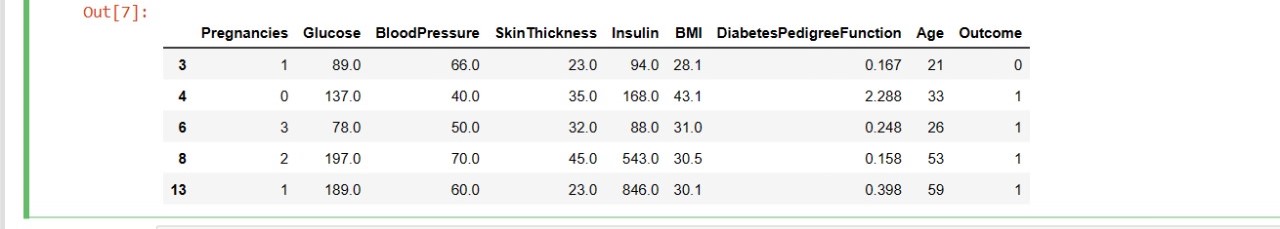


The code dataset.info() is a method call in python to display the information about dataset . The .info() method provides such as number of columns and rows datatypes of columns and memory usage of the dataset.

X = dataset.drop('Outcome', axis=1)

y = dataset['Outcome']

dataset.head()



The code dataset.head() is calling th head() function on the dataset onjecr.the head() function is used display first few rows of a data set.

Preprocessor=make\_column\_transformer(

(StandardScaler(),

Make\_column\_selector(dtype\_include=np.number));

)

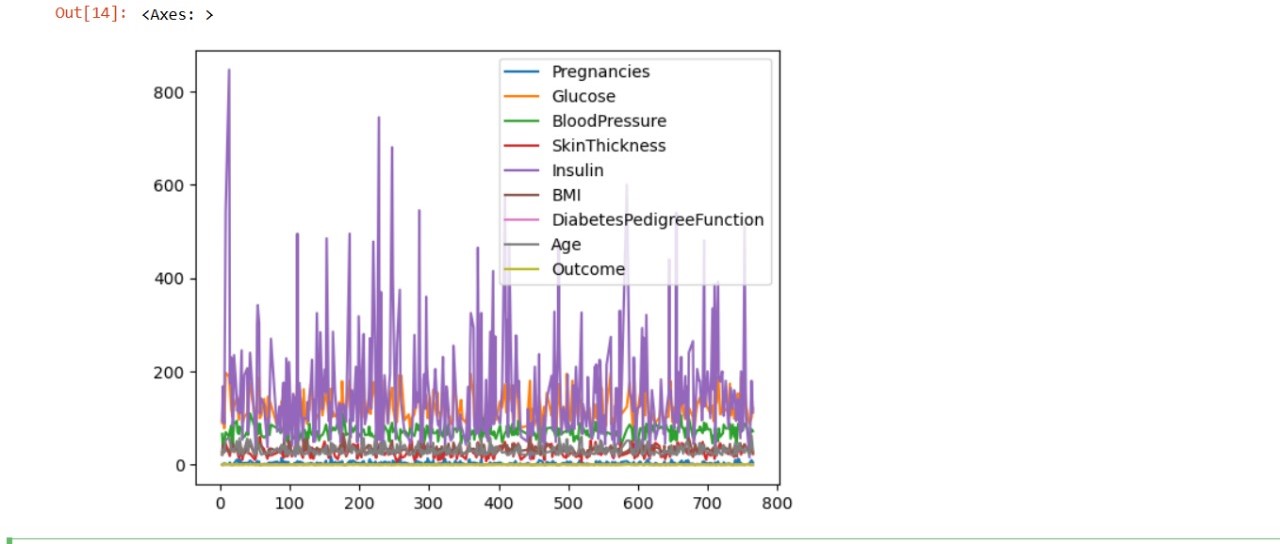
Preprocessor.fit(X)

X=preprocessor.transform(X)

dataset.head()



dataset.plot()

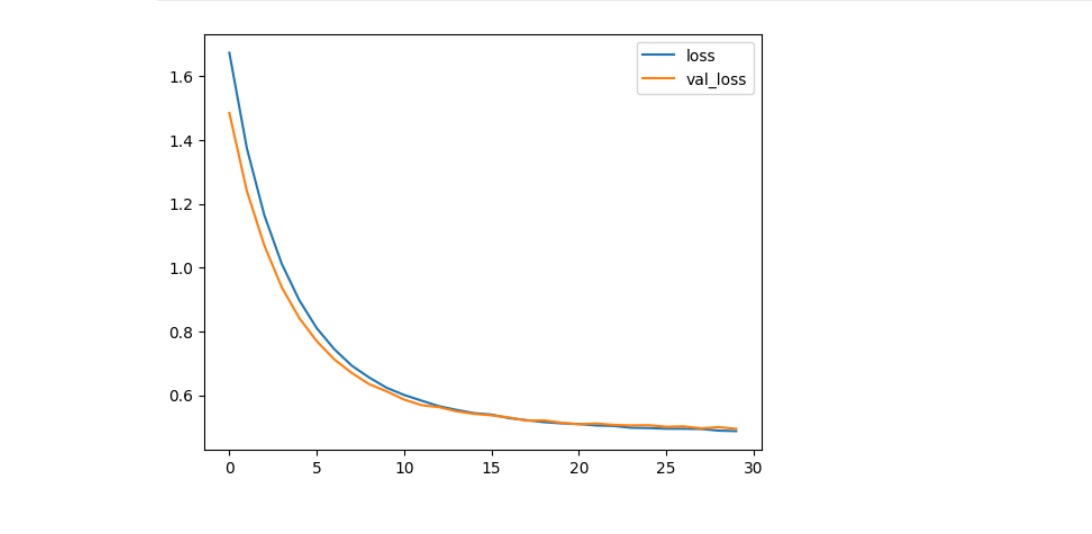


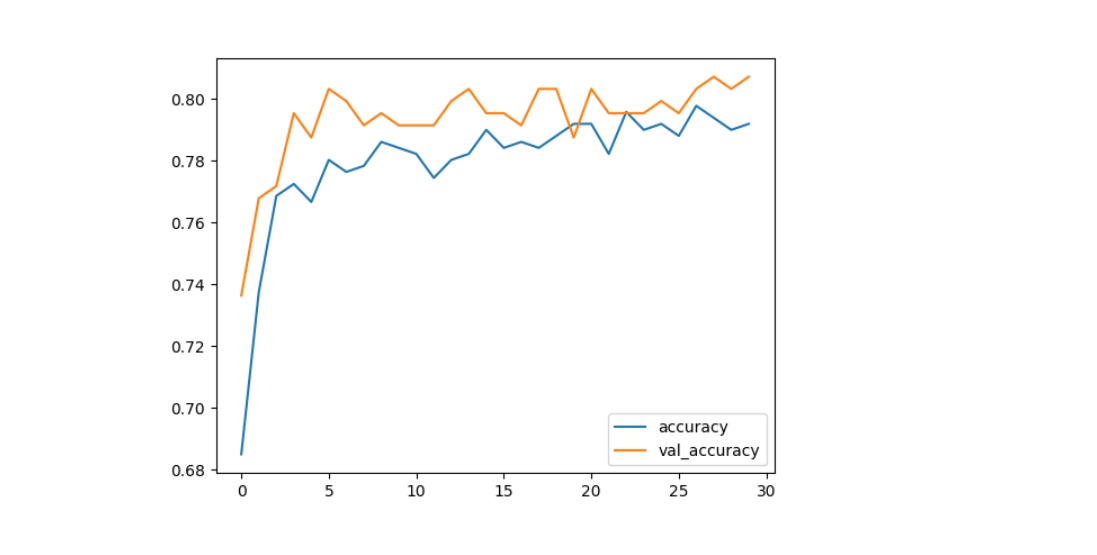
This code generates two arrays of random numbers with a normal distribution, assigns them to the variables x and y, plots them as a scatterplot using the scatter() function from the pyplot module of the matplotlib library, adds labels to the x-axis and y-axis, sets a title the plot, and displays the plot.

history\_df = pd.DataFrame(history.history)

history\_df.loc[:, ['loss','val\_loss']].plot();

history\_df.loc[:, ['accuracy','val\_accuracy']].plot();





from sklearn.metrics import confusion\_matrix

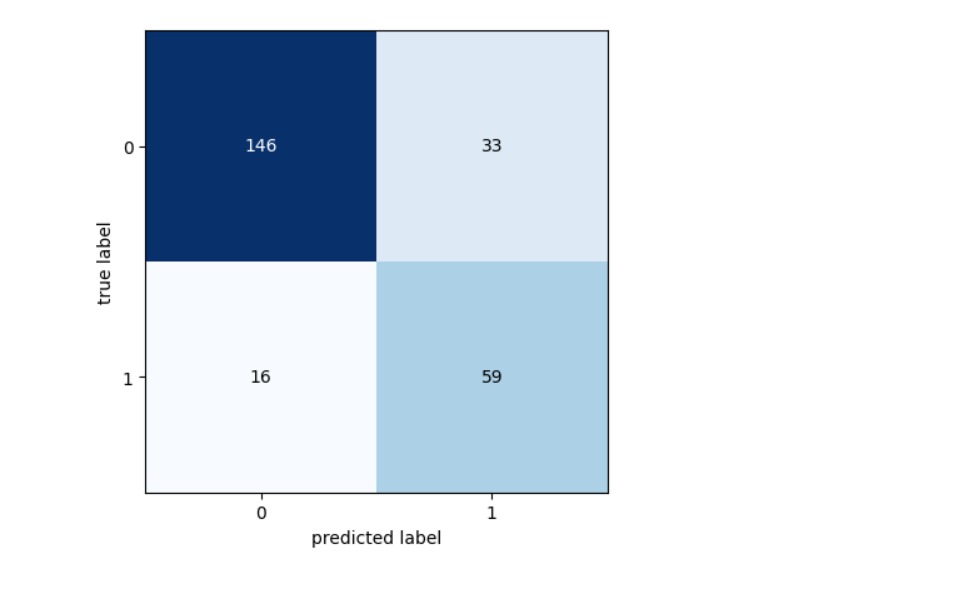
import matplotlib.pyplot as plt

cm = confusion\_matrix(y\_\_predict, y\_\_real)

from mlxtend.plotting import plot\_confusion\_matrix

fig, ax = plot\_confusion\_matrix(conf\_mat=cm)

plt.show()



dataset.rename(columns={'DiabetesPedigreeFunction': 'DPF'}, inplace= True)

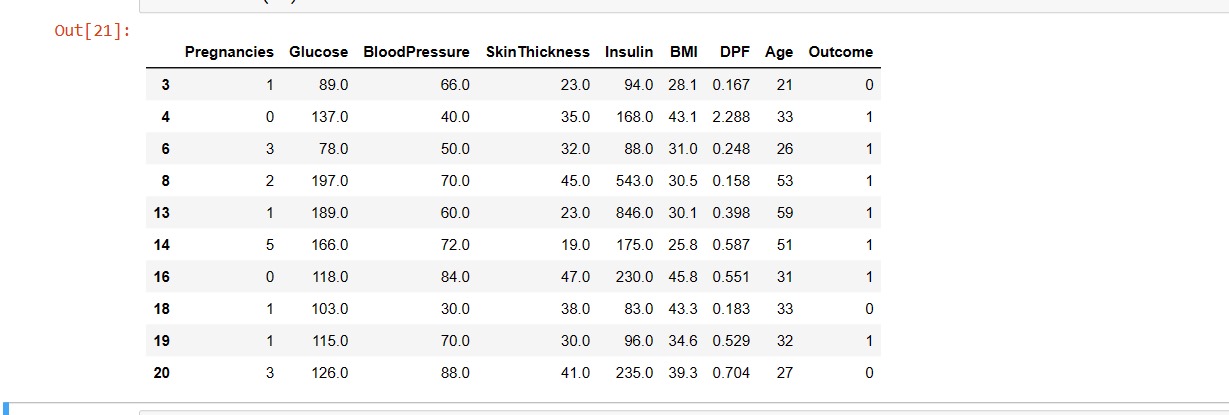
to\_nan = ['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin']

to\_nan.append(['BMI', 'DPF', 'Age'])

for i in range(len(to\_nan)):

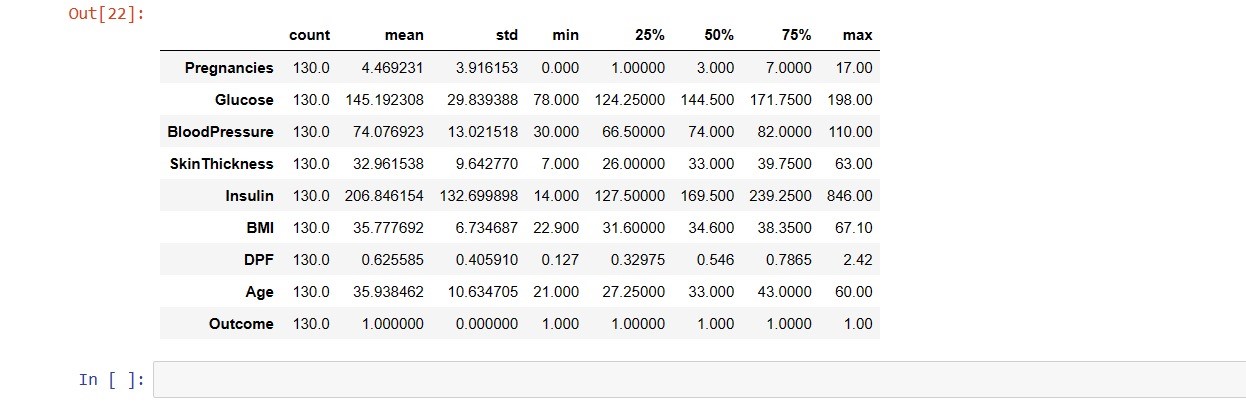
dataset[to\_nan[i]] = dataset[to\_nan[i]].replace(0, np.nan)

dataset.head(10)



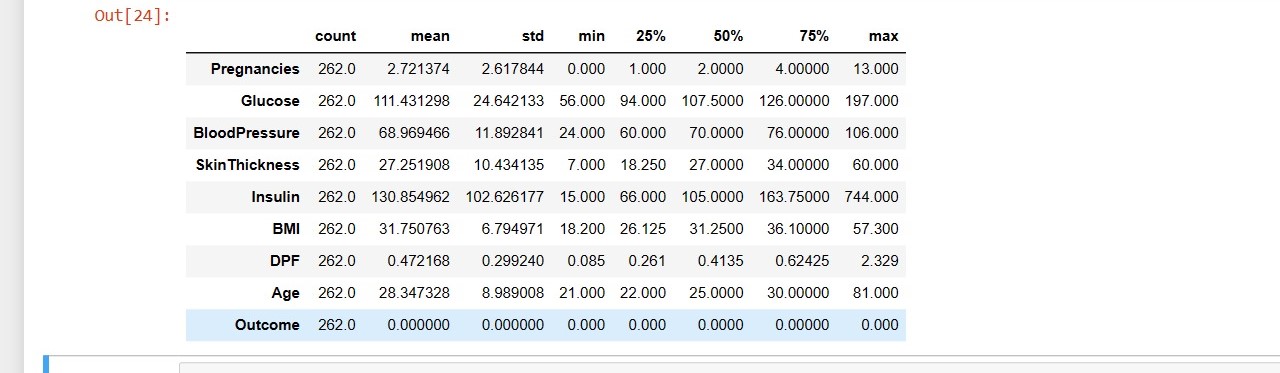
dataset\_true=dataset[(dataset.Outcome>0)]

dataset\_true.describe().T



dataset\_false = dataset[(dataset.Outcome<1)]

dataset\_false.describe().T



dataset.describe().T

