# Internship Project : Loan status Prediction

**SUBMITTED BY:**

***NAME: PARTHIV SARMA***

***EMAIL ID: parthivk2@gmail.com***

***BATCH: ML WITH PYTHON***

***CERTIFICATION CODE: TCRIG02R68***

***GROUP: OWN***

**QUESTION:**

Please perform the below steps in a Google Colab or Jupyter Notebook as per your convenience:

**In this assignment, we need to predict Loan status detection.**

**Features:**

Company wants to automate the loan eligibility process (real time) based on customer detail provided while filling online application form. These details are Gender, Marital Status, Education, Number of Dependents, Income, Loan Amount, Credit History and others. To automate this process, they have given a problem to identify the customers segments, those are eligible for loan amount so that they can specifically target these customers. Here they have provided a partial data set.

We will create a model with the following steps:

● Import the relevant packages

● Download and explore the dataset

● Prepare the dataset for training

● Use any prediction algorithm based upon the EDA

● Train the model to fit the data

● Make predictions using the trained model

● Create a test case and generate a predicted result from the system

**SOURCE CODE AND OUTPUTS:**

**#Importing the required Libraries and Dataset**

**import numpy as np**

**import pandas as pd**

**import matplotlib.pyplot as plt**

**import seaborn as sns**

**%matplotlib inline**

**import warnings**

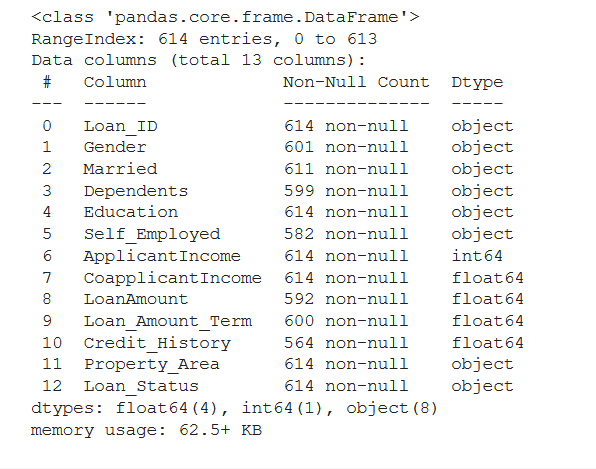
**warnings.filterwarnings('ignore')**

**from google.colab import drive**

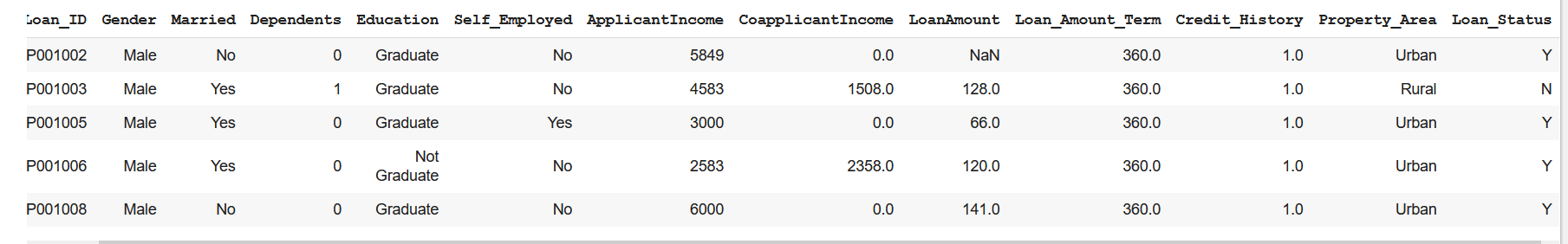
**drive.mount('/content/drive')**

**dataset=pd.read\_csv('/content/drive/MyDrive/ML with PY/final project/Loan\_status.zip')**

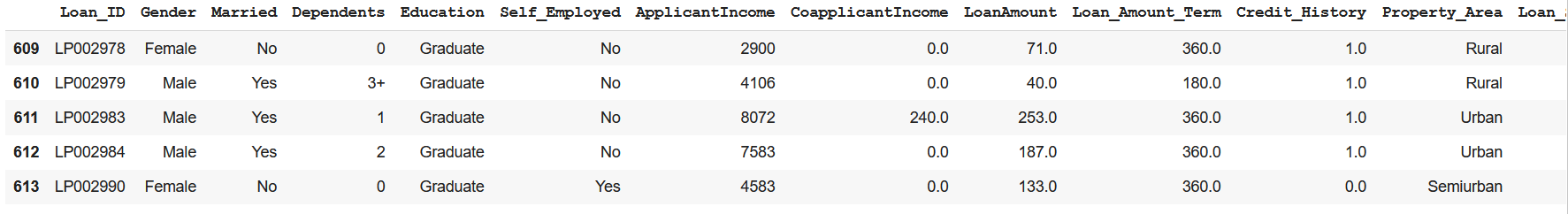
**dataset.info()**

****

**dataset.head()**

****

**dataset.tail()**

****

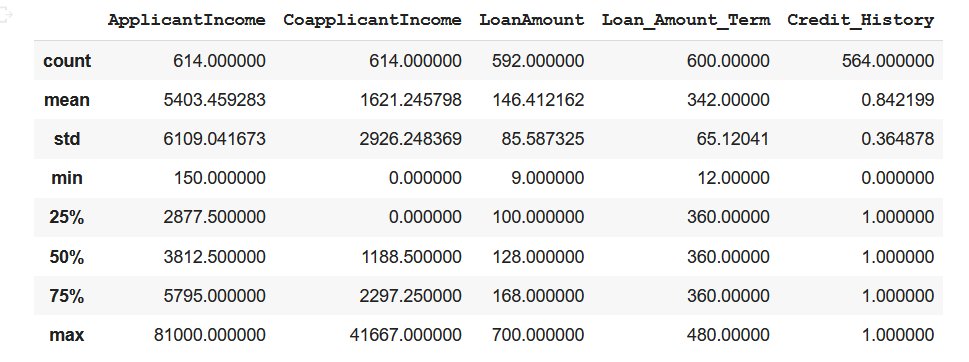
**dataset.\_\_len\_\_()**

****

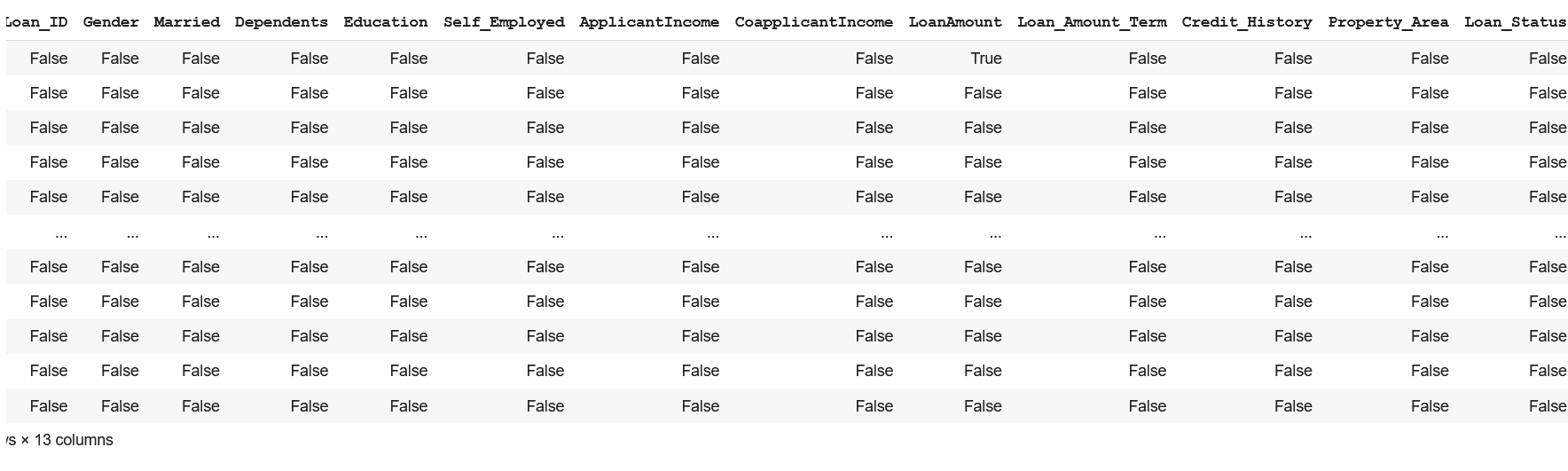
**dataset.shape**

****

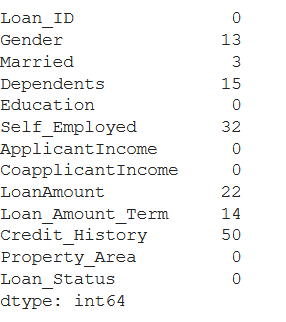
**dataset.describe()**

****

**dataset.isnull()**

****

**dataset.isnull().sum()**

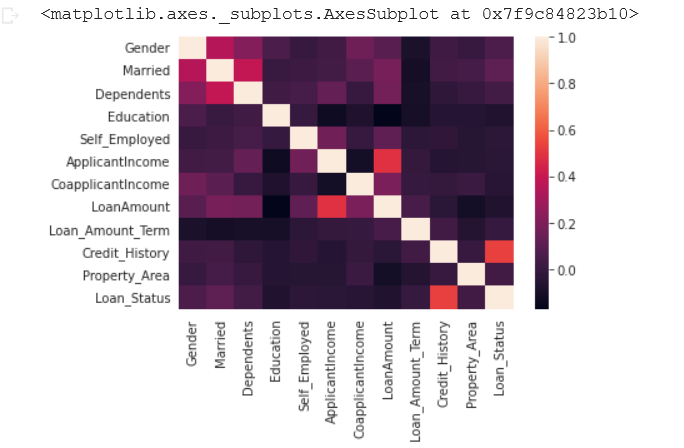
****

**#Checking for null values**

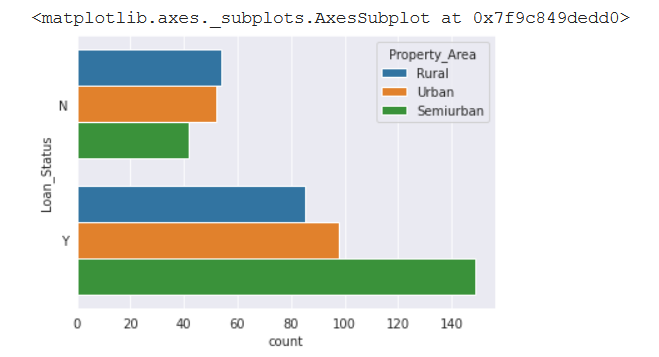
**dataset=dataset.dropna()**

**sns.set\_style("darkgrid")**

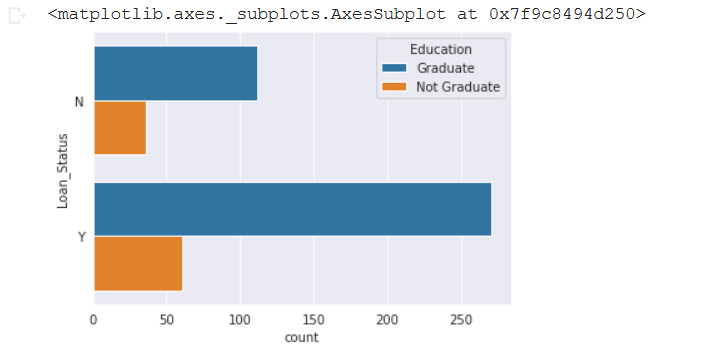
**sns.heatmap(dataset.corr())**

****

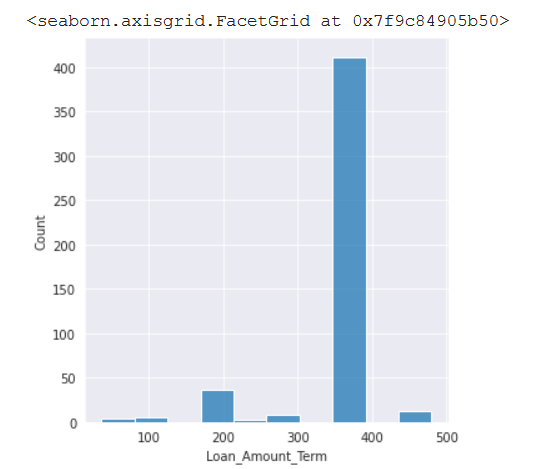
**sns.countplot(y="Loan\_Status", hue="Property\_Area",data=dataset)**

****

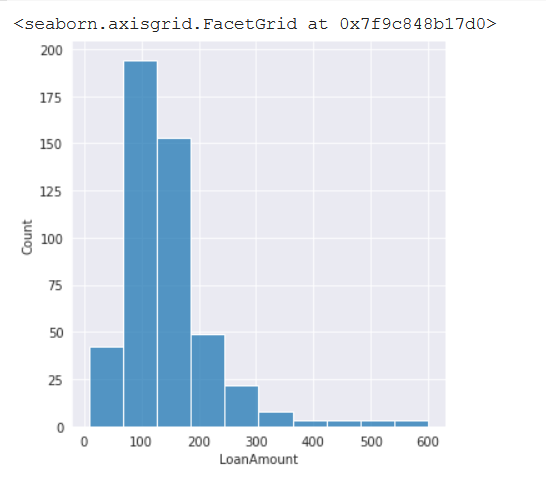
**sns.countplot(y="Loan\_Status",hue="Education",data=dataset)**

****

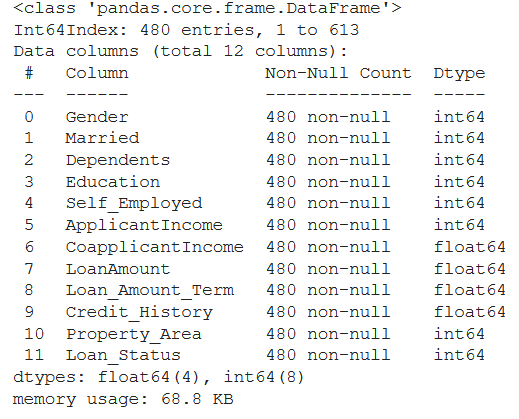
**sns.displot(dataset['Loan\_Amount\_Term'].dropna(),kde=False,bins=10)**

****

**sns.displot(dataset['LoanAmount'].dropna(),kde=False,bins=10)**

****

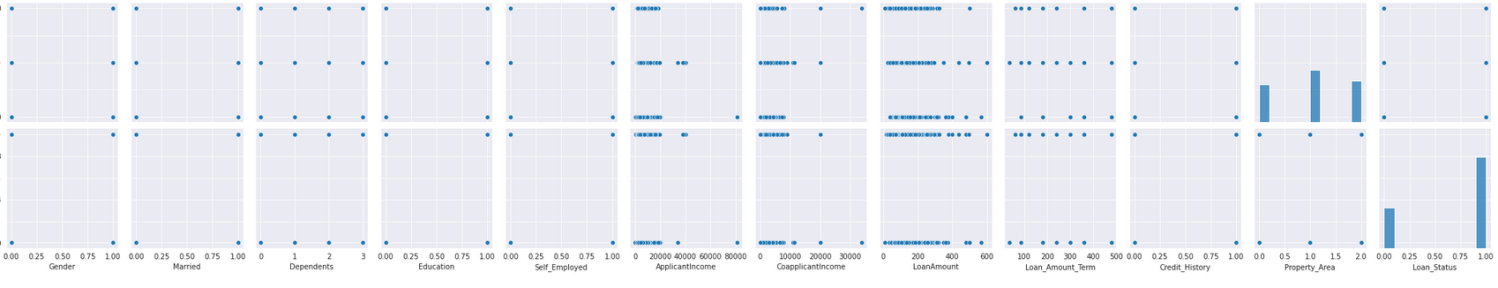
**dataset.info()**

****

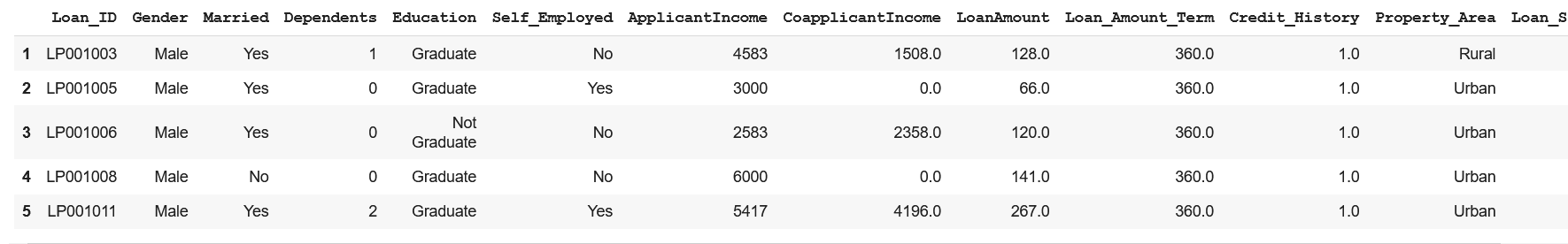
**sns.pairplot(dataset)**

****

****

****

**dataset.head()**

****

**from sklearn.preprocessing import LabelEncoder**

**le = LabelEncoder()**

**dataset['Married']=le.fit\_transform(dataset['Married'])**

**dataset['Self\_Employed']=le.fit\_transform(dataset['Self\_Employed'])**

**dataset['Loan\_Status']=le.fit\_transform(dataset['Loan\_Status'])**

**dataset['Education']=le.fit\_transform(dataset['Education'])**

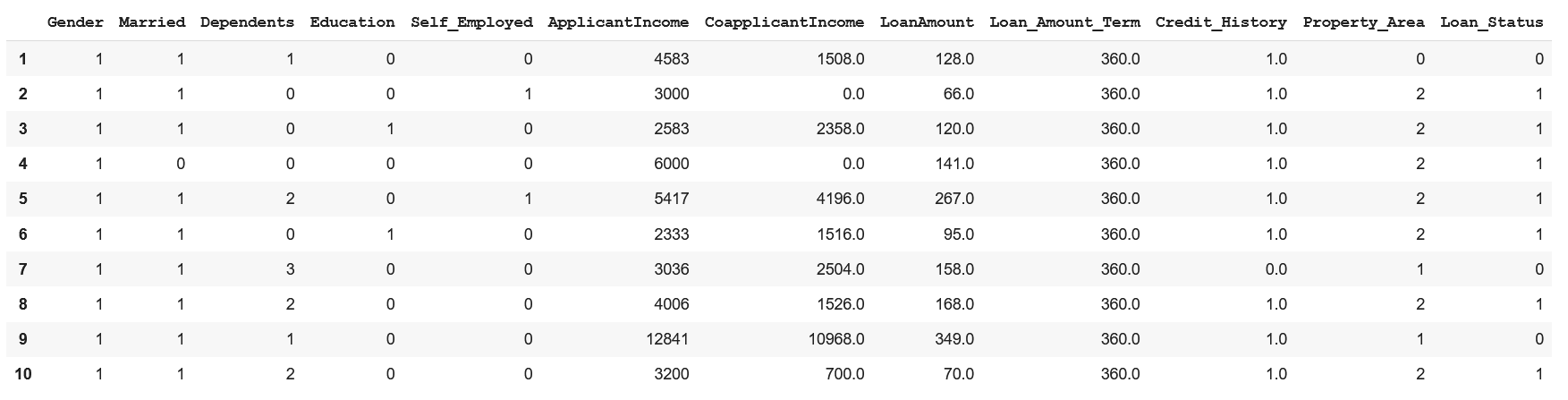
**dataset['Gender']=le.fit\_transform(dataset['Gender'])**

**dataset['Property\_Area']=le.fit\_transform(dataset['Property\_Area'])**

**dataset['Dependents']=le.fit\_transform(dataset['Dependents'])**

**dataset.drop('Loan\_ID', inplace=True, axis=1)**

**dataset.head(10)**



**df=pd.DataFrame(dataset, columns=dataset.columns)**

**X=df.drop('Loan\_Status', axis=1)**

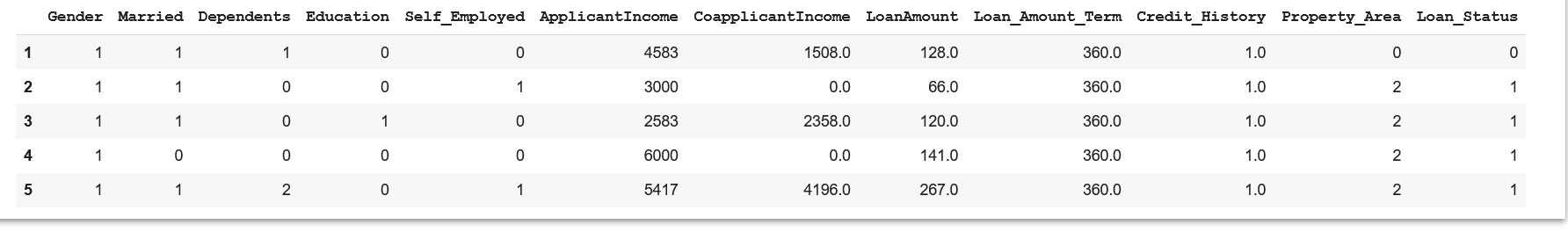
**y=df['Loan\_Status']**

**#Scaling the dataset**

**from sklearn.preprocessing import StandardScaler**

**scaler=StandardScaler()**

**df.head()**

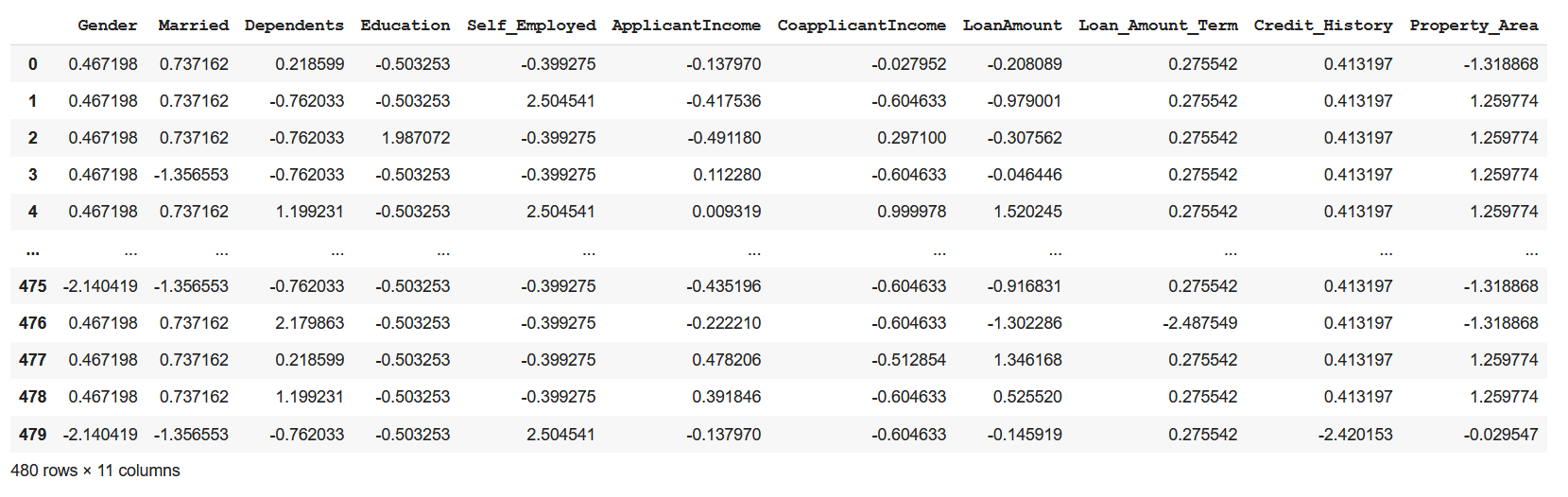
****

**scaler.fit(X)**

****

**X = pd.DataFrame(scaler.transform(X), columns=df.drop('Loan\_Status', axis=1).columns)**

**X**

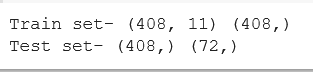
****

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

**X\_train,X\_test,Y\_train,Y\_test=train\_test\_split(X,y,test\_size=0.15,random\_state=42)**

**print('Train set-',X\_train.shape,Y\_train.shape)**

**print('Test set-',Y\_train.shape,Y\_test.shape)**

****

**from sklearn.linear\_model import LogisticRegression**

**loan\_model=LogisticRegression()**

**loan\_model.fit(X\_train,Y\_train)**

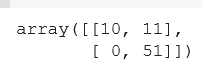
****

**predictions=loan\_model.predict(X\_test)**

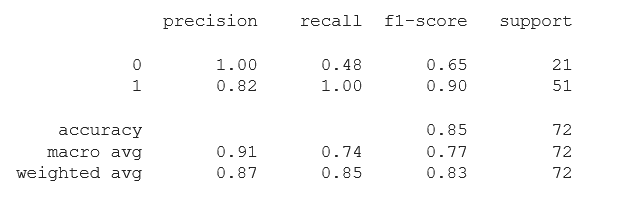
**#Confusion Matrix and Classification Report**

**from sklearn.metrics import confusion\_matrix**

**confusion\_matrix(Y\_test,predictions)**

****

**from sklearn.metrics import classification\_report**

****

**#Measuring Accuracy**

**from sklearn import metrics**

**print('The accuracy achieved using Logistic Regression is: ', metrics.accuracy\_score(predictions, Y\_test))**

****

**# Conclusion:**

**# Thus, the accuracy achieved by using Logistic Regression is around 84.72 %.**

**------------------------------------------------------------------------------------------------------------------**