Group 32

Project Report

Project – Loan Default Prediction

Introduction to Data Analytics

MS4610 – 2020

Team Members:

CHANDRAMUNI ME18B043

CHHOTU KUMAR ME18B045

1. Introduction

In this project we were given a dataset consisting following details on loans taken by customers:

- ID: A Unique identifier for every financial loan that is being considered.
- Loan type: Types of loan taken by Customer (Two types, ‘A’ or ‘B’)
- Occupation type: Occupation of Customer (Three Occupation types, ‘X’ or ‘Y’, or ‘Z’
- Income: A continuous variable indicative of annual income of Customer
- Expense: A continuous variable indicative of annual expense of Customer
- Age: Age of Customer – Value of ‘0’ is considered as below 50, and value of ‘1’ is considered as above 50
- Score1, Score2, Score3, Score4, Score5: Represents five different metrics calculated by the organization, about the customer and the loan that is being considered
- Label: ‘0’ means loan taken by customer is “non-default”, and ‘1’ means loan taken by customer is “default”

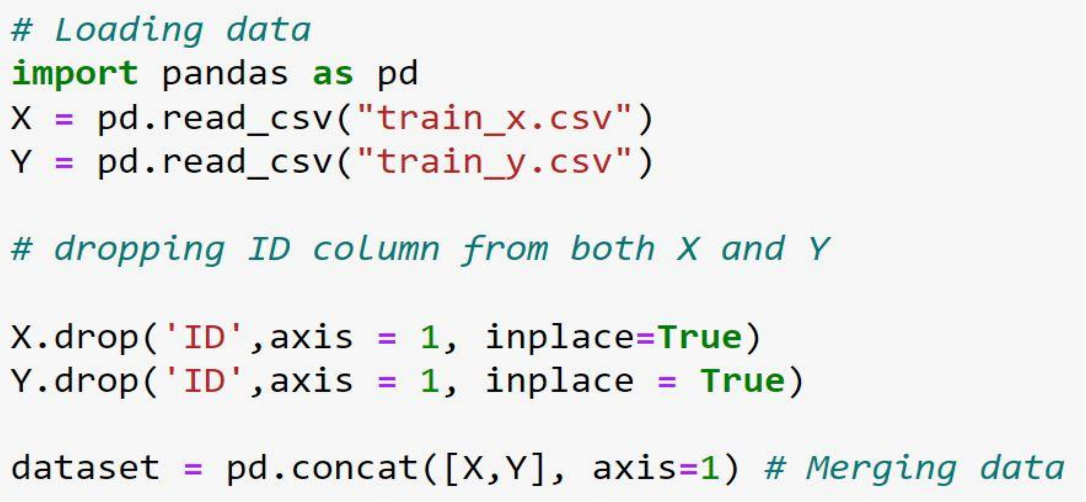
Using these information and data about Customers we are required to build a machine learning model to predict whether a loan will go default or not, and to understand which of the features are important and helpful in the prediction.

2. Exploratory Data Analysis and Visualization:

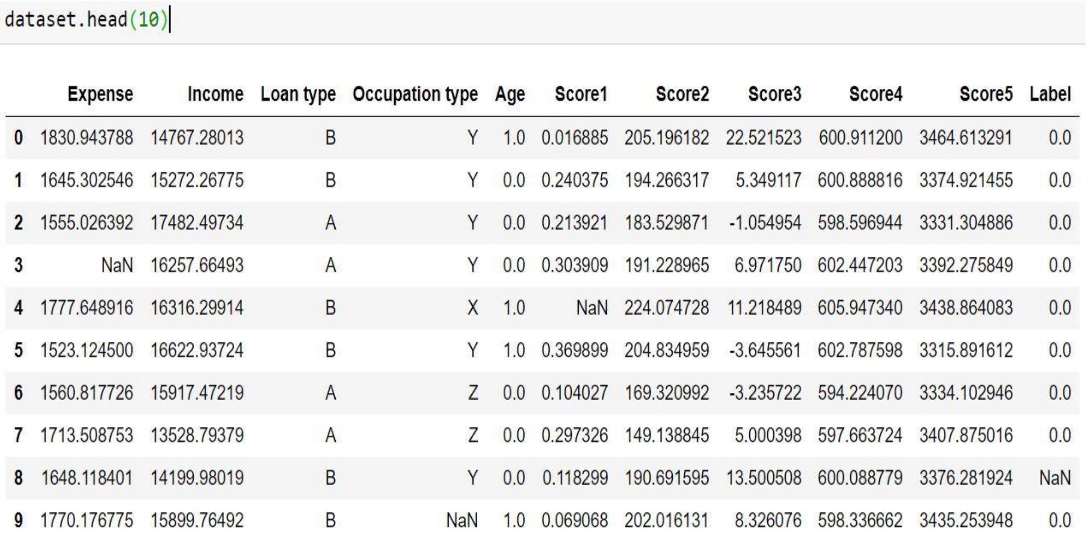
Exploratory Data Analysis, or EDA, is essentially a type of storytelling for statisticians. It allows us to uncover patterns and insights, often with visual methods, within data. It is often the first step of the data processing.

For this, we are using histogram, scatter plots etc. for visualization.

- Loading data: for this we are using pandas read_csv () method.



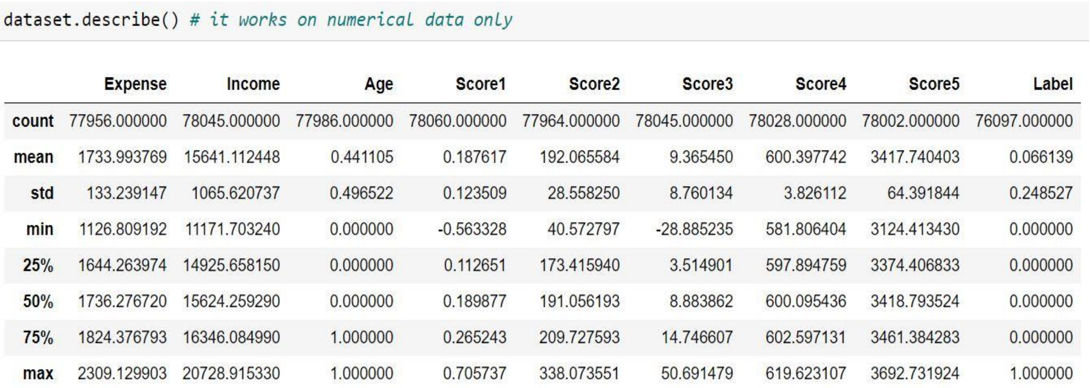
```python
# Loading data
import pandas as pd
X = pd.read_csv("train_x.csv")
Y = pd.read_csv("train_y.csv")

# dropping ID column from both X and Y

X.drop('ID',axis = 1, inplace=True)
Y.drop('ID',axis = 1, inplace = True)

dataset = pd.concat([X,Y], axis=1) # Merging data
```



```python
dataset.head(10)
```

	Expense	Income	Loan type	Occupation type	Age	Score1	Score2	Score3	Score4	Score5	Label
0	1830.943788	14767.28013	B	Y	1.0	0.016885	205.196182	22.521523	600.911200	3464.613291	0.0
1	1645.302546	15272.26775	B	Y	0.0	0.240375	194.266317	5.349117	600.888816	3374.921455	0.0
2	1555.026392	17482.49734	A	Y	0.0	0.213921	183.529871	-1.054954	598.596944	3331.304886	0.0
3	NaN	16257.66493	A	Y	0.0	0.303909	191.228965	6.971750	602.447203	3392.275849	0.0
4	1777.648916	16316.29914	B	X	1.0	NaN	224.074728	11.218489	605.947340	3438.864083	0.0
5	1523.124500	16622.93724	B	Y	1.0	0.369899	204.834959	-3.645561	602.787598	3315.891612	0.0
6	1560.817726	15917.47219	A	Z	0.0	0.104027	169.320992	-3.235722	594.224070	3334.102946	0.0
7	1713.508753	13528.79379	A	Z	0.0	0.297326	149.138845	5.000398	597.663724	3407.875016	0.0
8	1648.118401	14199.98019	B	Y	0.0	0.118299	190.691595	13.500508	600.088779	3376.281924	NaN
9	1770.176775	15899.76492	B	NaN	1.0	0.069068	202.016131	8.326076	598.336662	3435.253948	0.0

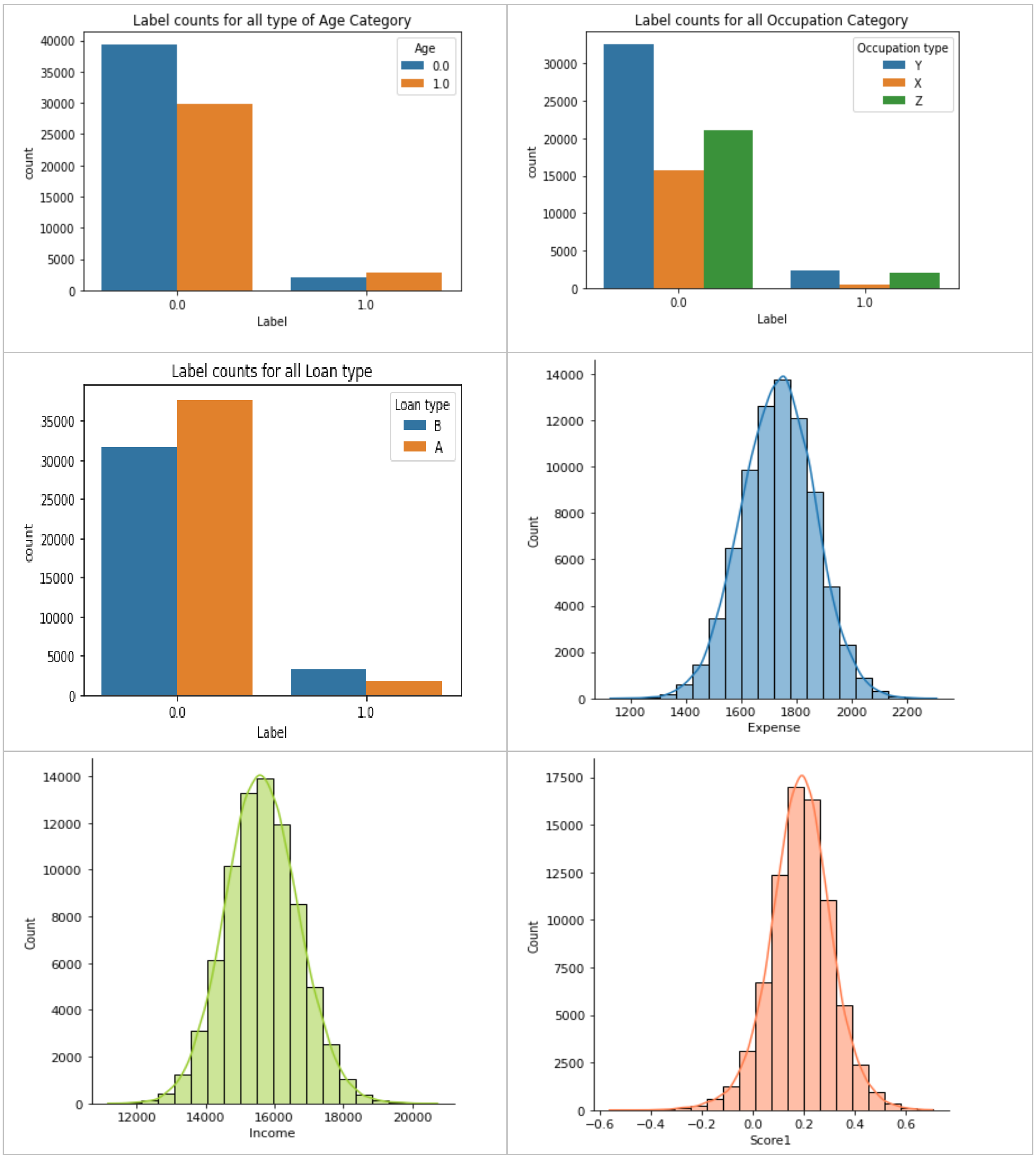
- Statistical Overview of data: for this we are using pandas describe() method

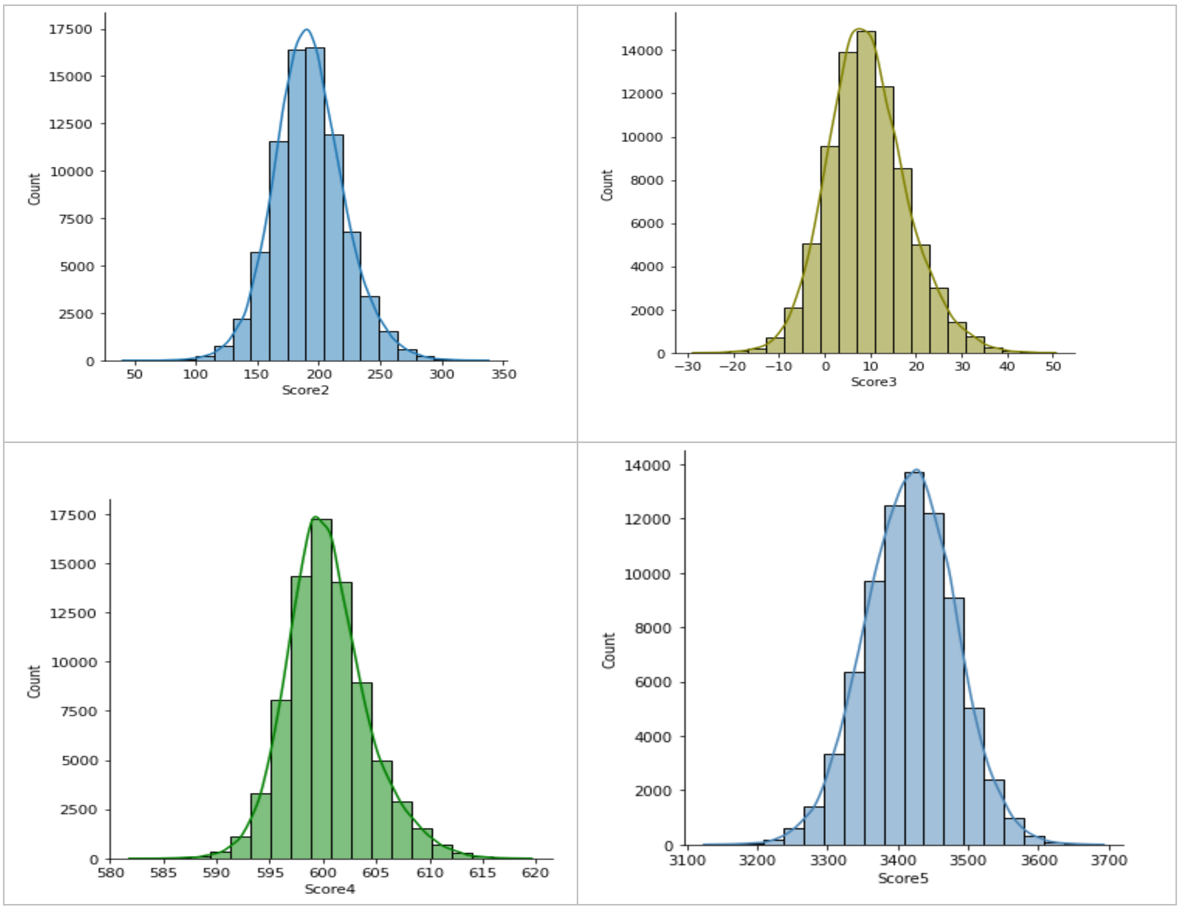


```python
dataset.describe() # it works on numerical data only
```

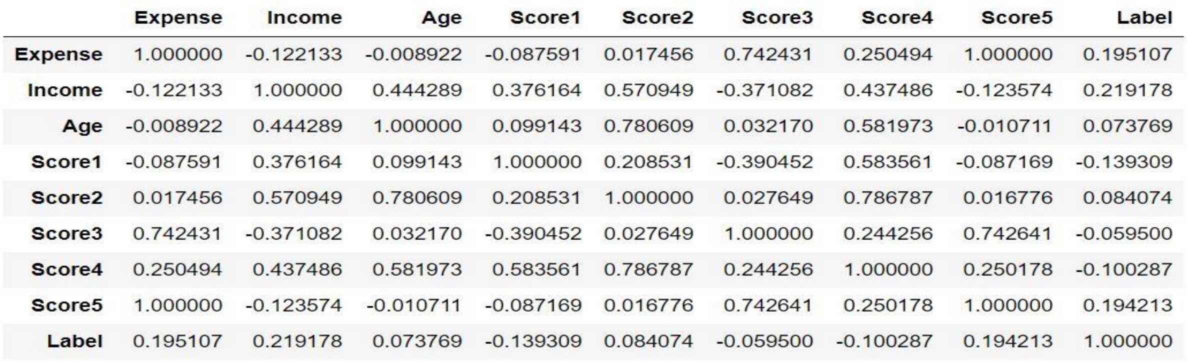
	Expense	Income	Age	Score1	Score2	Score3	Score4	Score5	Label
count	77956.000000	78045.000000	77986.000000	78060.000000	77964.000000	78045.000000	78028.000000	78002.000000	76097.000000
mean	1733.993769	15641.112448	0.441105	0.187617	192.065584	9.365450	600.397742	3417.740403	0.066139
std	133.239147	1065.620737	0.496522	0.123509	28.558250	8.760134	3.826112	64.391844	0.248527
min	1126.809192	11171.703240	0.000000	-0.563328	40.572797	-28.885235	581.806404	3124.413430	0.000000
25%	1644.263974	14925.658150	0.000000	0.112651	173.415940	3.514901	597.894759	3374.406833	0.000000
50%	1736.276720	15624.259290	0.000000	0.189877	191.056193	8.883862	600.095436	3418.793524	0.000000
75%	1824.376793	16346.084990	1.000000	0.265243	209.727593	14.746607	602.597131	3461.384283	0.000000
max	2309.129903	20728.915330	1.000000	0.705737	338.073551	50.691479	619.623107	3692.731924	1.000000

- Distribution of data:

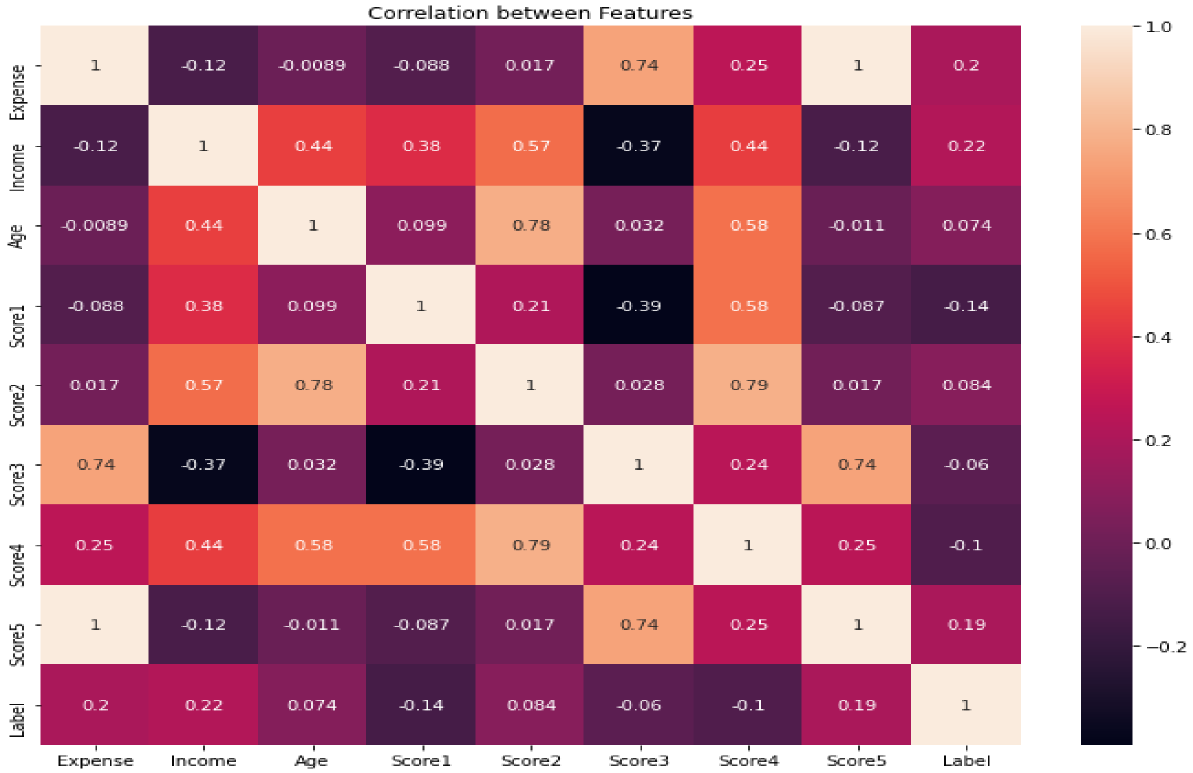


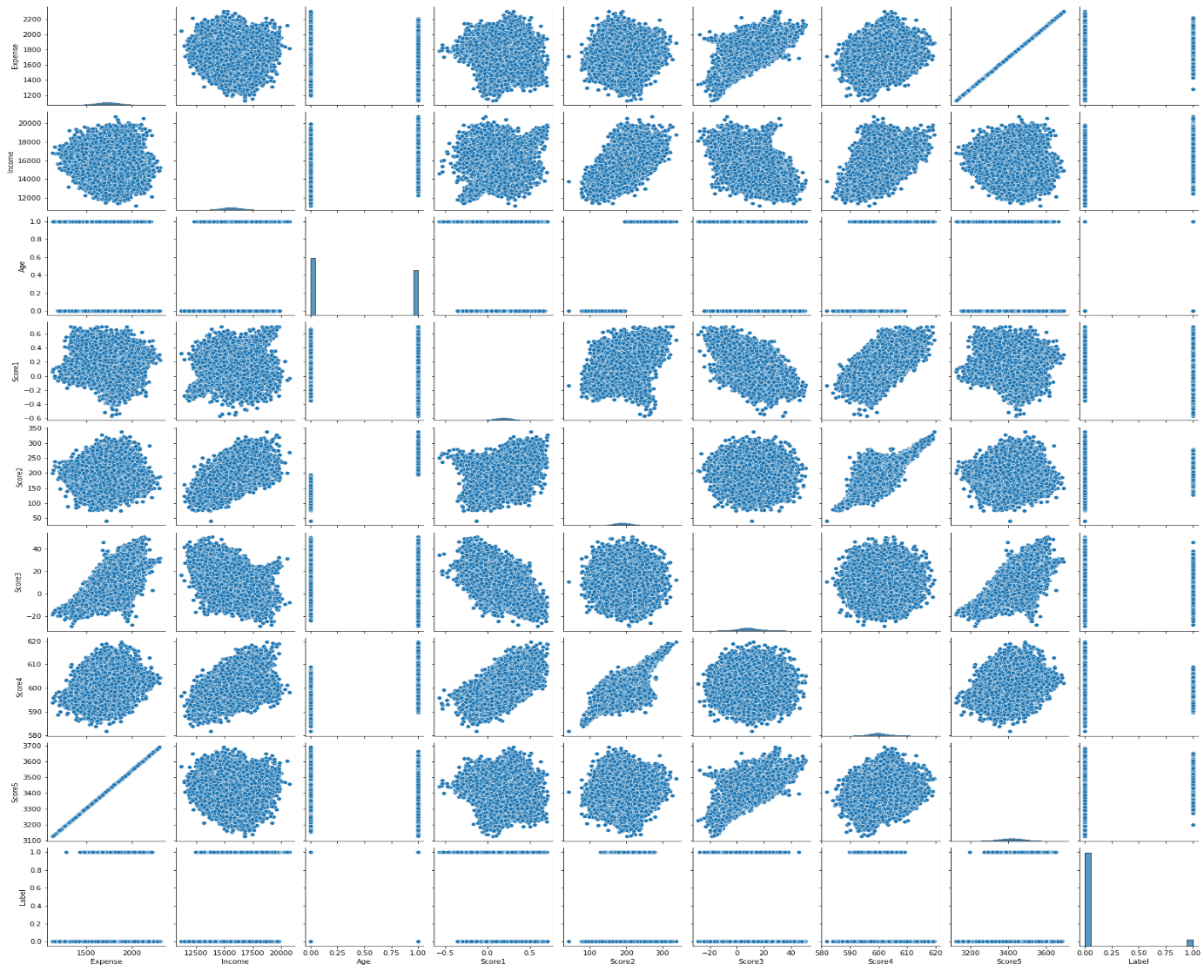


- Correlations: It is used to find how much one feature is related to other features. For this we are using panadas corr() method it gives Pearson Correlation Coefficient. Its value lies between -1 to 1. Negative shows opposite direction and positive shows same direction and magnitude shows strength of relation. Higher value indicates that they are highly correlated and lower value indicates they are less correlated.
  Correlation matrix is shown below.



	Expense	Income	Age	Score1	Score2	Score3	Score4	Score5	Label
Expense	1.000000	-0.122133	-0.008922	-0.087591	0.017456	0.742431	0.250494	1.000000	0.195107
Income	-0.122133	1.000000	0.444289	0.376164	0.570949	-0.371082	0.437486	-0.123574	0.219178
Age	-0.008922	0.444289	1.000000	0.099143	0.780609	0.032170	0.581973	-0.010711	0.073769
Score1	-0.087591	0.376164	0.099143	1.000000	0.208531	-0.390452	0.583561	-0.087169	-0.139309
Score2	0.017456	0.570949	0.780609	0.208531	1.000000	0.027649	0.786787	0.016776	0.084074
Score3	0.742431	-0.371082	0.032170	-0.390452	0.027649	1.000000	0.244256	0.742641	-0.059500
Score4	0.250494	0.437486	0.581973	0.583561	0.786787	0.244256	1.000000	0.250178	-0.100287
Score5	1.000000	-0.123574	-0.010711	-0.087169	0.016776	0.742641	0.250178	1.000000	0.194213
Label	0.195107	0.219178	0.073769	-0.139309	0.084074	-0.059500	-0.100287	0.194213	1.000000


Correlation between Features
Expense
Income
Age
Score1
Score2
Score3
Score4
Score5
Label
1 -0.12 -0.0089 -0.088 0.017 0.74 0.25 1 0.2
-0.12 1 0.44 0.38 0.57 -0.37 0.44 -0.12 0.22
-0.0089 0.44 1 0.099 0.78 0.032 0.58 -0.011 0.074
-0.088 0.38 0.099 1 0.21 -0.39 0.58 -0.087 -0.14
0.017 0.57 0.78 0.21 1 0.028 0.79 0.017 0.084
0.74 -0.37 0.032 -0.39 0.028 1 0.24 0.74 -0.06
0.25 0.44 0.58 0.58 0.79 0.24 1 0.25 -0.1
1 -0.12 -0.011 -0.087 0.017 0.74 0.25 1 0.19
0.2 0.22 0.074 -0.14 0.084 -0.06 -0.1 0.19 1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2


Expense
Income
Age
Score1
Score2
Score3
Score4
Score5
Label
1500 2000
12500 15000 17500 20000
0.00 0.25 0.50 0.75 1.00
−0.5 0.0 0.5
100 200 300
−20 0 20 40
580 590 600 610 620
3200 3400 3600
0.00 0.25 0.50 0.75 1.00

- From Correlation matrix we found that Feature ‘Score5’ and ‘Expense’ are highly correlated, so we need to remove any one feature ‘Score5’ or ‘Expense’ from dataset to make model stable because they bring the same information.

3. Handling missing values in dataset:

There are lots of method to handle missing data.

- Deleting row:
  It is generally used when we have very large dataset and only few rows have missing values. But using it when having small dataset is not a good idea because It leads to loss of information which will not give the expected result while predicting output.
- Replacing with Mean/Median/Mode:
  This strategy is only applicable for feature having numerical data like age of person, income etc. It adds variance to the data set but the loss of the data can be negated by this method which yields better result compared to removal of rows.
- Assigning a unique Category:
  This can be used for handling missing categorical feature but it adds extra information to data set which result change in variance.
- Predicting the missing values:
  Using the features which do not have the missing value we can predict for missing value. This method may result in better accuracy, unless a missing value is expected to have a very high variance.

For our project we have used Replacing with ‘mode’ and applied one-hot-encoding for handling ‘categorical features’ and used Replace with ‘mean’ for ‘numerical features’.

4. Feature Scaling:

Feature scaling is essential for machine learning algorithms that calculate distances between data. If not scale, the feature with a higher numeric value range starts dominating when calculating distances. Because algorithms work on numbers it does not know what the number represent. As human we know how age of customer is different from income of customer but computer see those numbers and give high priority to income i.e. is wrong.

- Machine Learning algorithms which require feature Scaling: K-nearest neighbors (KNN), K-means, Principle Component Analysis (PCA), we can speed up Gradient descent by scaling.
- Machine Learning algorithms which do not require scaling: All Tree based algorithms- CART, Random Forests, Gradient Boosted Decision Trees etc.

In our dataset all features have roughly Gaussian distribution, so we have used python sklearn StandardScaler function to turn them into normal distribution with mean zero and variance 1.

5. Cross Validation:

Cross-validation is a technique to evaluate predictive models by partitioning the original sample into a training set to train the model, and a test set to evaluate it. In k-fold cross-validation, the original sample is randomly partitioned into k equal size subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k-1 subsamples are used as training data. The cross-validation process is then repeated k times (the folds), with each of the k subsamples used exactly once as the validation data. The k results from the folds can then be averaged (or otherwise combined) to produce a single estimation. The advantage of this method is that all observations are used for both training and validation, and each observation is used for validation exactly once.

In our project we have used 10-fold cross-validation on different-different classification model and repeated 5 times and calculating average accuracy for each type of classifier.

Here is the result for all classifier we have tested in cross validation.

i) Logistic Regression:

Accuracy:	95.71%
Precision:	84.01%
Recall:	43.55%
F1 Score:	57.33%

ii) Decision Tree:

Accuracy:	97.05%
Precision:	78.18%
Recall:	76.80%
F1 Score:	77.47%

iii) Random Forest:

Accuracy:	97.67%
Precision:	95.13%
Recall:	68.28%
F1 Score:	79.49%

iv) K-Nearest Neighbors(K=3):

Accuracy:	98.11%
Precision:	92.89%
Recall:	77.36%
F1 Score:	84.41%

v) XGBoost:

Accuracy:	96.45%
Precision:	77.08%
Recall:	66.12%
F1 Score:	71.03%

6. Final Model Selection and Prediction of new Test data:

After comparing the results of all above classifier, we find K-Nearest Neighbors (K =3) has the highest accuracy of 98.11% with Precision of 92.89%, Recall of 77.36% and has F1 score of 84.41%, so we choose K-Nearest Neighbors(k=3) as our final prediction model and using this model we have predicted output for given test data.