BUSINESS REPORT FOR DATA MINING PROJECT

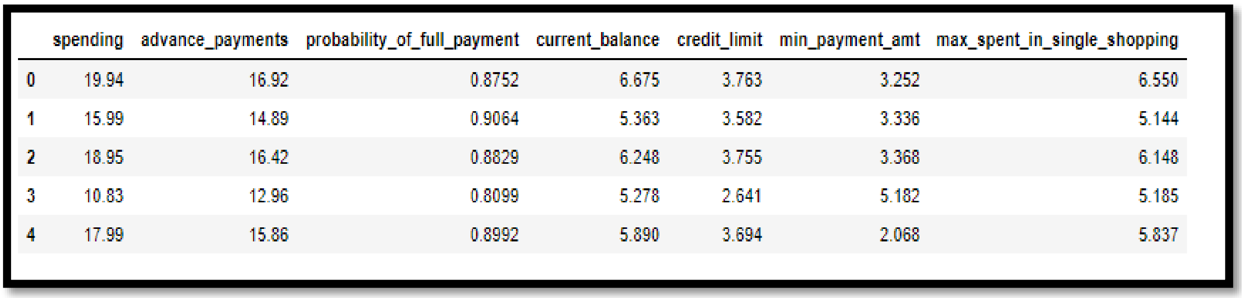
Table of Contents:

Problem 1: Clustering

A leading bank wants to develop a customer segmentation to give promotional offers to its customers. They collected a sample that summarizes the activities of users during the past few months. You are given the task to identify the segments based on credit card usage.

Question 1.1 : Read the data and do exploratory data analysis. Describe the data briefly.

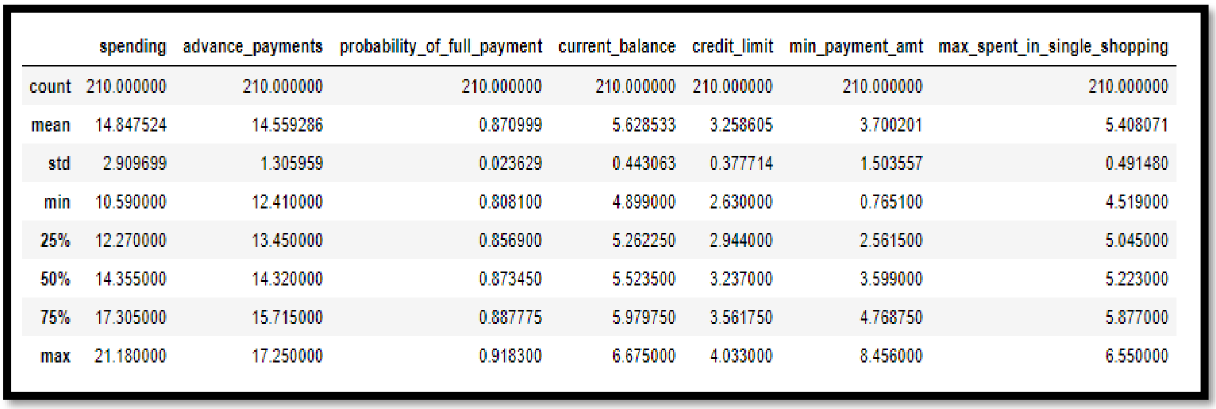
Exploratory Data Analysis:



	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
0	19.94	16.92	0.8752	6.675	3.763	3.252	6.550
1	15.99	14.89	0.9064	5.363	3.582	3.336	5.144
2	18.95	16.42	0.8829	6.248	3.755	3.368	6.148
3	10.83	12.96	0.8099	5.278	2.641	5.182	5.185
4	17.99	15.86	0.8992	5.890	3.694	2.068	5.837

The dataset contains 7 Variables. All the Variables are of float64 datatype.

Descriptive Statistics of the Dataset:

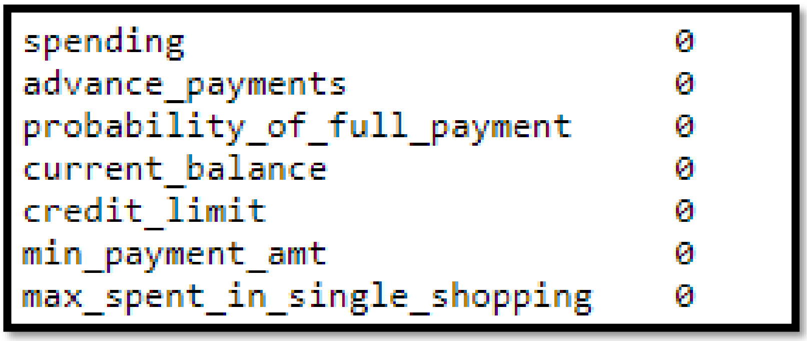


	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
count	210.000000	210.000000	210.000000	210.000000	210.000000	210.000000	210.000000
mean	14.847524	14.559286	0.870999	5.628533	3.258605	3.700201	5.408071
std	2.909699	1.305959	0.023629	0.443063	0.377714	1.503557	0.491480
min	10.590000	12.410000	0.808100	4.899000	2.630000	0.765100	4.519000
25%	12.270000	13.450000	0.856900	5.262250	2.944000	2.561500	5.045000
50%	14.355000	14.320000	0.873450	5.523500	3.237000	3.599000	5.223000
75%	17.305000	15.715000	0.887775	5.979750	3.561750	4.768750	5.877000
max	21.180000	17.250000	0.918300	6.675000	4.033000	8.456000	6.550000

The describe function(describe()) provides us with the information about how much the data is spread across along with the information on the mean, standard deviation, count etc.

Since all the variables is of numerical category, there is no unique values or frequently used categorical variable here. Based on the distribution of values from the above table, we can conclude that these variables may be normally distributed.

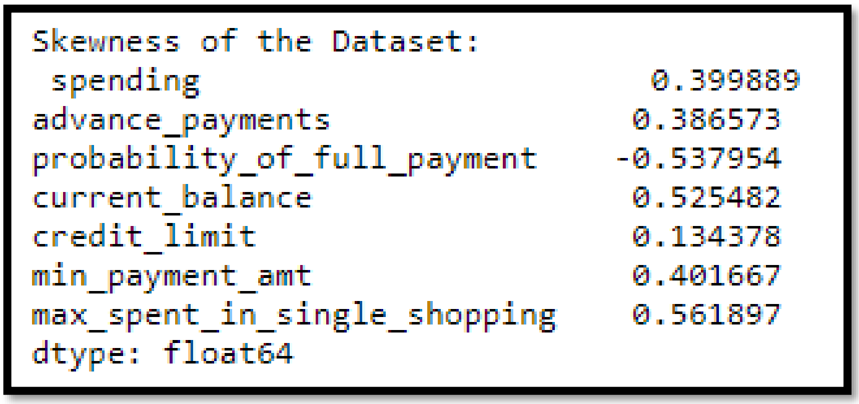
Check for NULL Values:



```
spending                        0
advance_payments                0
probability_of_full_payment     0
current_balance                 0
credit_limit                    0
min_payment_amt                 0
max_spent_in_single_shopping    0
```

The isnull() and sum() function combined can clearly figure out on whether given dataset has any NULL(N/A) values or not. From the above result it is evident that there are no NULL values and therefore we can proceed further in the Exploratory Data Analysis.

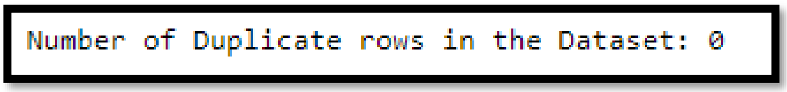
Skewness Check:



```
Skewness of the Dataset:
 spending                        0.399889
advance_payments                0.386573
probability_of_full_payment    -0.537954
current_balance                 0.525482
credit_limit                    0.134378
min_payment_amt                 0.401667
max_spent_in_single_shopping    0.561897
dtype: float64
```

Skewness is a measure of the asymmetry of the probability distribution of a real-valued random variable about its mean. From the above result we can see that only the variable "probability_of_full_payment" is negatively skewed and everything else in right skewed.

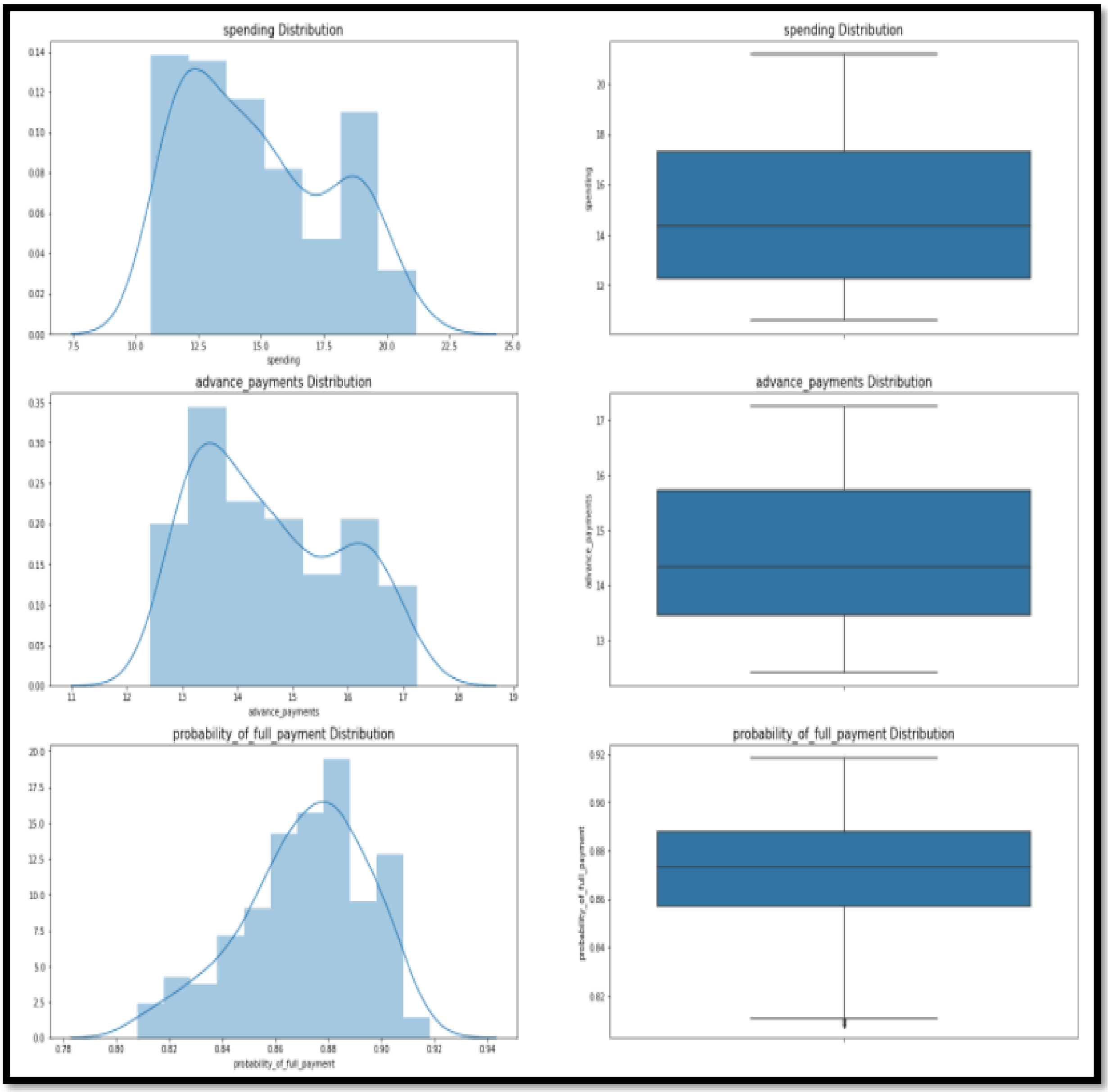
Duplicates Check:



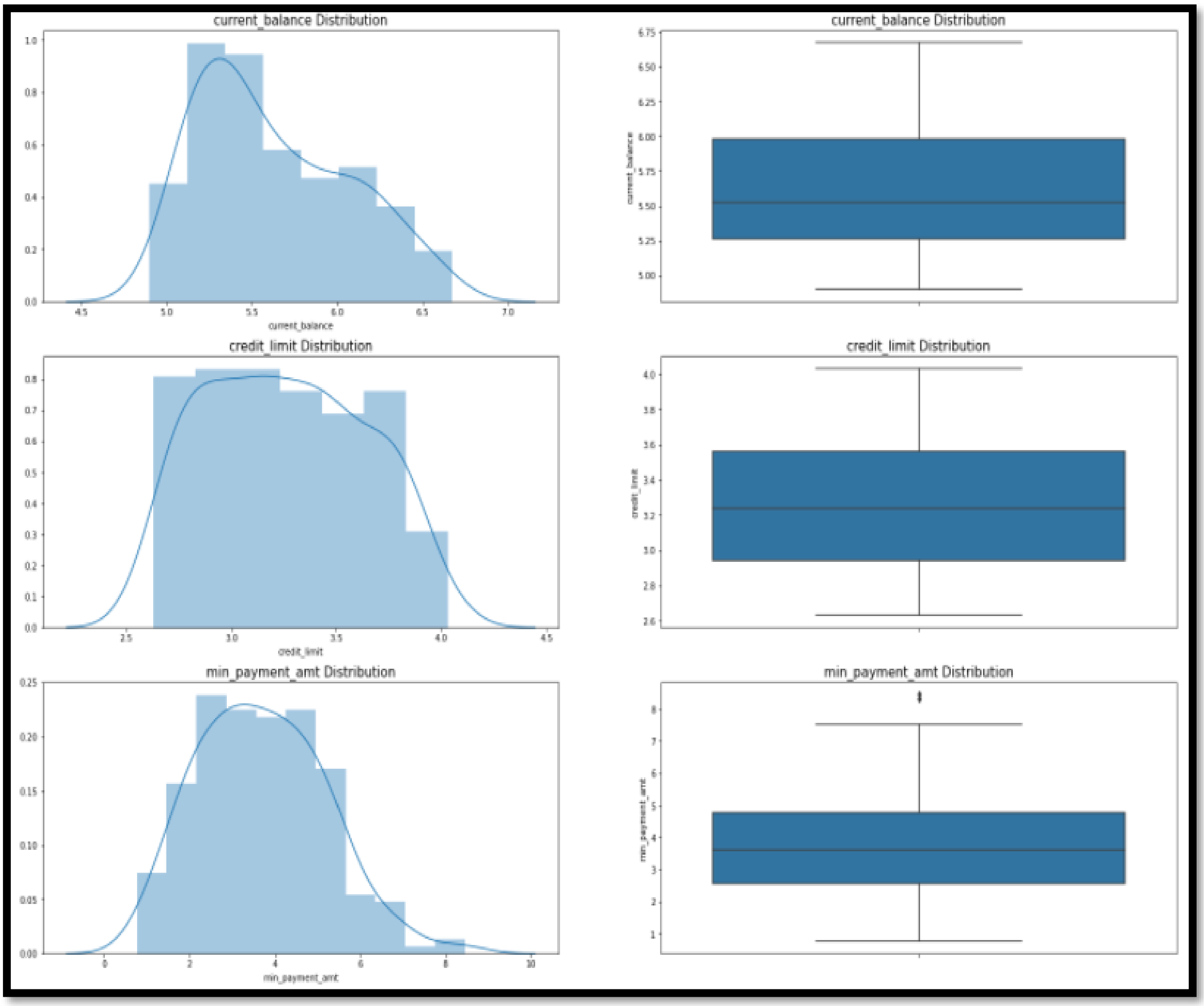
```
Number of Duplicate rows in the Dataset: 0
```

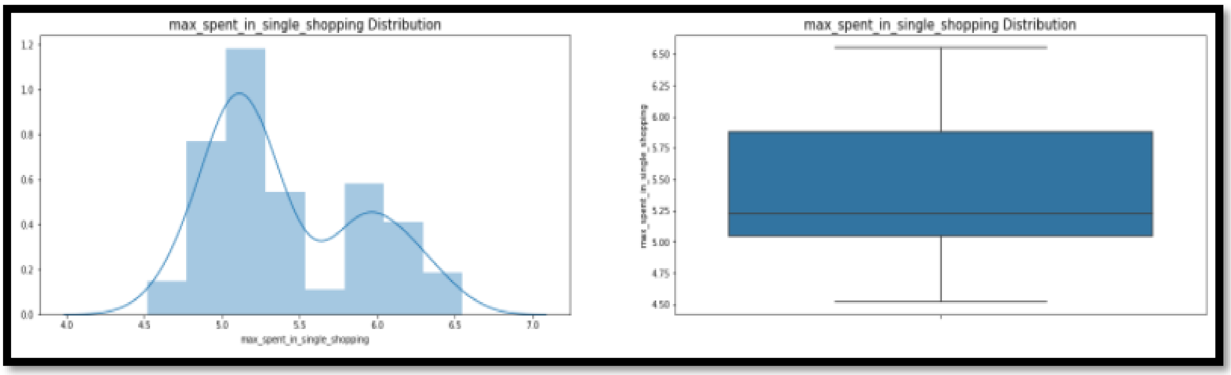
The calculation part is done in the notebook attached and we can see that there is no Duplicates present in the dataset. Therefore, no action is required to correct duplicates so that we can move further in Exploratory Data Analysis.

Univariate Data Visualization (Dist-plot and Boxplot analysis):



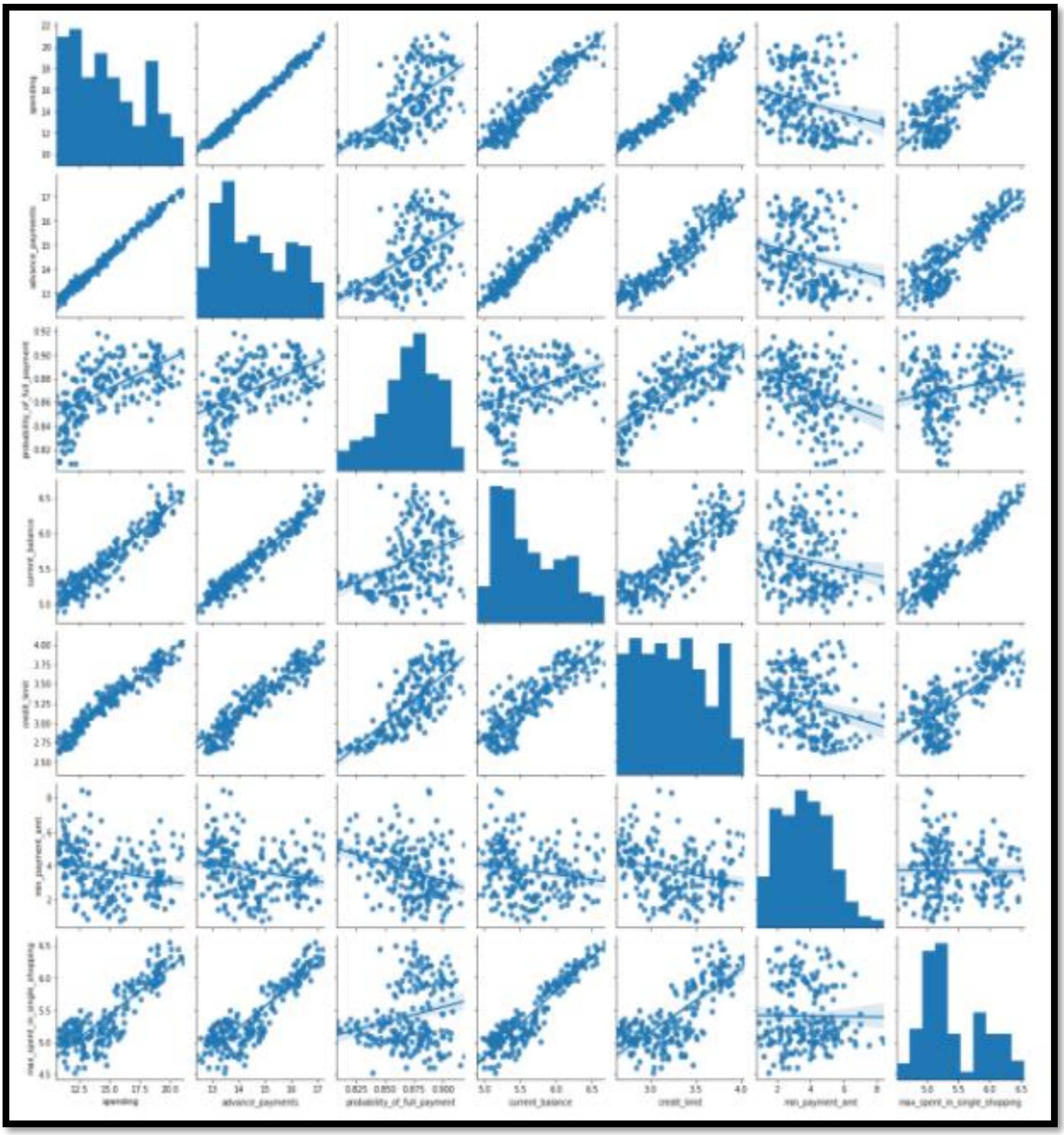
For first three columns “Distplot” and “Boxplot” are done to check the distribution and we can clearly see that all the columns “spending” , “advance_payments” and “probability_of_full_payment” are normally distributed with a little skewness to the right for the first two and left skewness for the last one. There are also outliers in the “probability_of_full_payment” column which can be seen below the Q1, but since they are relatively near, we can give the benefit of the doubt that they are valid outliers and proceed(But Business side confirmation is recommended).





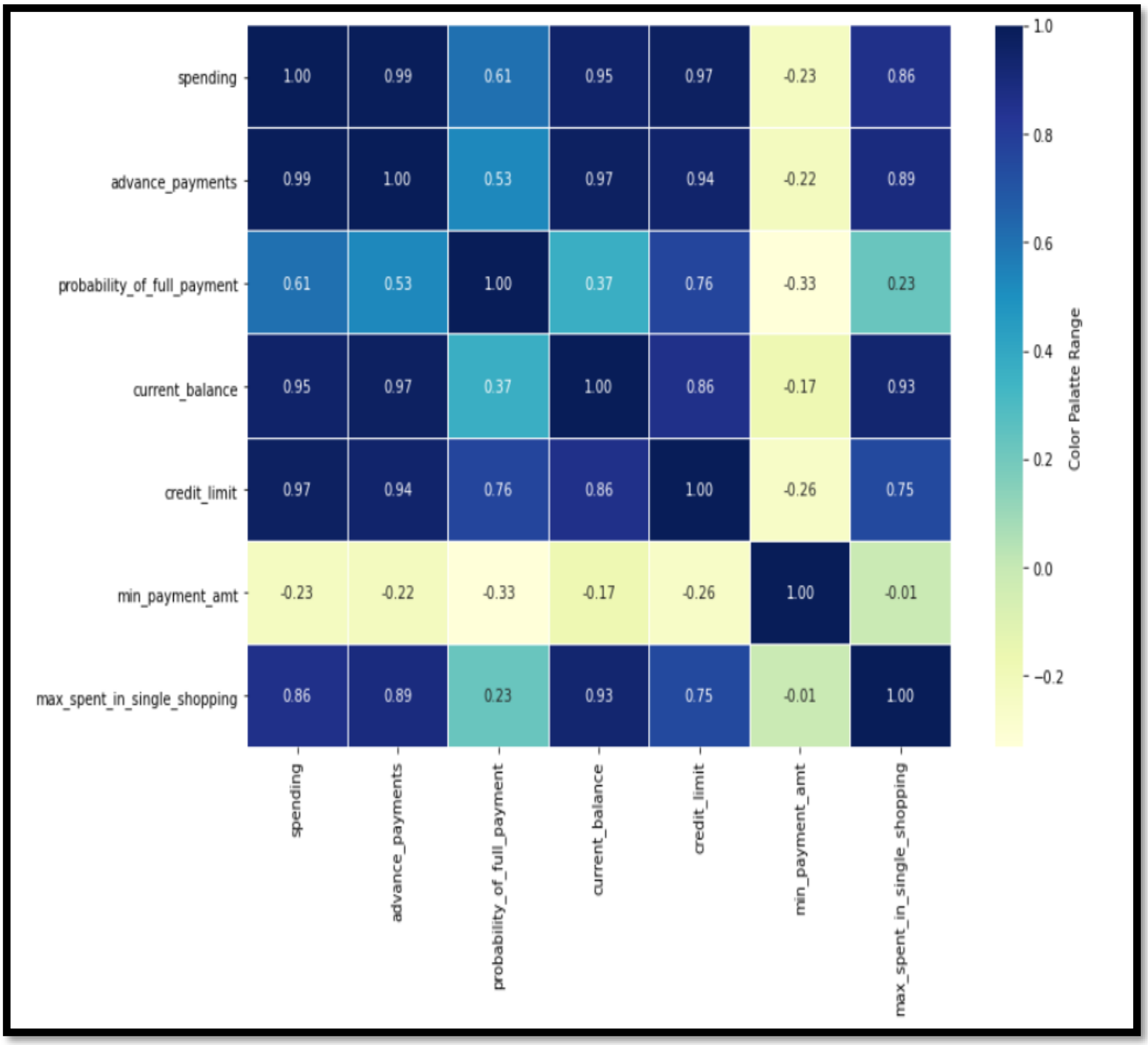
We are also performing the same visualization for the remainder variables to see their distribution. From the above results we can see that all the rest of the variables are normally distributed indeed(relatively). Also, notably they are all right skewed(positive skewness). Similar to "probability_of_full_payment" we also have a variable with outliers called "min_payment_amt", but this time the outliers are present above Q3 and since they are relatively near, we can give the benefit of the doubt that they are valid outliers and proceed(But Business side confirmation is recommended).

Bivariate Data Visualization (Pair-plot analysis):



A regular Pair-Plot analysis is first performed on the dataset to see their interaction with each of the other variables. From the above result we can say that there are some strong relationships with the variables like spending , advance_payments , current_balance , current_balance and credit_limit. The pair-plot analysis can give us only a rough estimate on the linear relationship between the variables and its evident here that we do have strong relationships that we can exploit in our analysis.

Bivariate Data Visualization (Heat Map analysis):



Another most important plot that we must look in the case of analysis is the Correlation plot. The Heat-Map analysis is an visualized part of the Correlation matrix that is used to find the relationship between the variables with numerical values to standpoint on how much they are related as compared to other variables.

The following conclusions can be sought out of the above results:

- Except “min_payment_amt” everything else almost has high positive correlation relationship with each other.
- The highlight being the relationship between “spending” and “current_balance and credit_limit”(99% , 95% and 97% respectively).
- The only negative correlation comes from the “min_payment_amt” with their relationship with every other variable(highest being -33% from “min_payment_amt” and “probability_of_full_payment”).

Question 1.2: Do you think scaling is necessary for clustering in this case? Justify

To answer this question specifically let us look at the Data Dictionary that was provided to us to give us further information on the data.

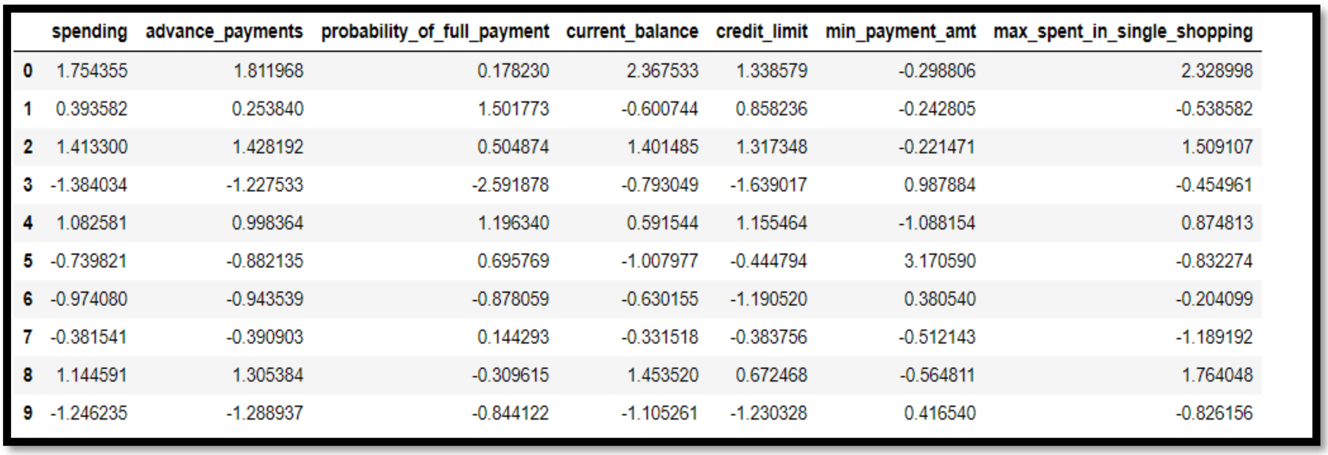
Data Dictionary for Market Segmentation:

1. spending: Amount spent by the customer per month (in 1000s)
2. advance_payments: Amount paid by the customer in advance by cash (in 100s)
3. probability_of_full_payment: Probability of payment done in full by the customer to the bank
4. current_balance: Balance amount left in the account to make purchases (in 1000s)
5. credit_limit: Limit of the amount in credit card (10000s)
6. min_payment_amt : minimum paid by the customer while making payments for purchases made monthly (in 100s)
7. max_spent_in_single_shopping: Maximum amount spent in one purchase (in 1000s)

From the above Data Dictionary, we can see that the data are in various metrics(like 100s,1000s,10000s).

Clustering techniques use Euclidean Distance(or other distance methods too) to form the cohorts and therefore it will be wise to scale the variables having different metrics before calculating the distance for clustering methods. Therefore, by performing a scaling method say Z-Scaling method we can shift these huge differences in metrics to variables of equal weights which can hugely help is clustering methods as many(or most) of the clustering methods are HIGHLY SENSITIVE to variables of different metrics.

After Scaling the variables using Z-score(mean : 0 and STD :1):



	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
0	1.754355	1.811968	0.178230	2.367533	1.338579	-0.298806	2.328998
1	0.393582	0.253840	1.501773	-0.600744	0.858236	-0.242805	-0.538582
2	1.413300	1.428192	0.504874	1.401485	1.317348	-0.221471	1.509107
3	-1.384034	-1.227533	-2.591878	-0.793049	-1.639017	0.987884	-0.454961
4	1.082581	0.998364	1.196340	0.591544	1.155464	-1.088154	0.874813
5	-0.739821	-0.882135	0.695769	-1.007977	-0.444794	3.170590	-0.832274
6	-0.974080	-0.943539	-0.878059	-0.630155	-1.190520	0.380540	-0.204099
7	-0.381541	-0.390903	0.144293	-0.331518	-0.383756	-0.512143	-1.189192
8	1.144591	1.305384	-0.309615	1.453520	0.672468	-0.564811	1.764048
9	-1.246235	-1.288937	-0.844122	-1.105261	-1.230328	0.416540	-0.826156

The above result is a sample 10 records for which we have performed scaling and we can see that the once variables with huge difference in metrics has changed to values ranging from -3 to +3 (mostly fitted data).

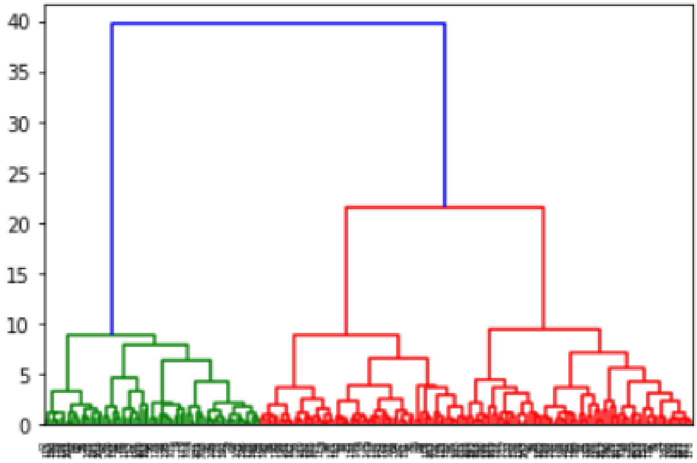
Question 1.3: Apply hierarchical clustering to scaled data. Identify the number of optimum clusters using Dendrogram and briefly describe them:

(Note : Refer the notebook attached for calculations)

For Applying Hierarchical clustering, we use the scaled data that we got from previous results (Question 1.2). Important to note some of the parameters that is involved with the method of clustering.First is the wardlink(linkage method) that is passed on the scaled data(method = ward).

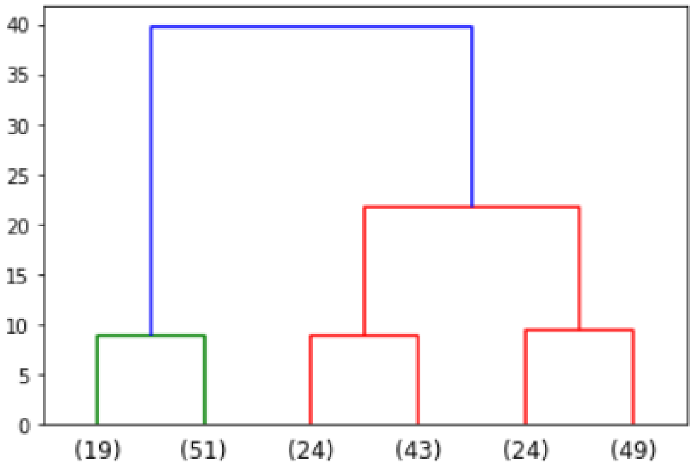
Dendogram:

Next step would be to pass the wardlink that we got previously to the dendogram function.



From the above result we can see that there are too many clusters that are very hard to distinguish and identify as they are too clubbed to one another and this way itll be hard to identify the optimum clusters for the dataset.

Optimized Dendogram:



From the above result which is obtained by optimization(truncate_mode='lastp',p = 6) we can see a clean looking Dendogram which we can use to identify the number of clusters required and continue with the analysis of clusters.

There are usually two major criterions used in identifying the number of clusters for a given problem on Dendogram.

- Maxclust
- Distance

Maxclust uses the number of clusters directly by passing them through the code on fclusters method based on your assumptions from the dendogram that is produced.

Whereas the Distance method uses the distance part of the dendogram that we can manipulate and cut the dendogram so that we can see the optimum number of clusters fit to our assumptions.

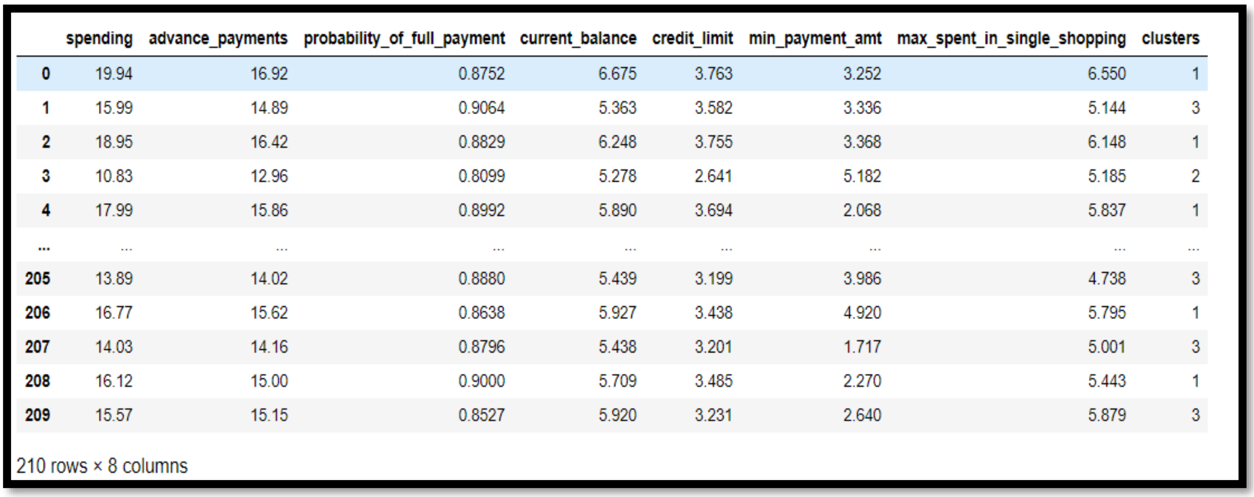
For the sake of making sure that we do correctly, both criterions are used in the fcluster method and we were able to create the clusters.

The optimum number of clusters identified from the Dendogram : 3

Now we can easily add these predicted clusters to the original dataset to see firsthand on the clusters that each row belong to.

The results are added in a file called 'hc_cluster_result.csv' which is attached to the submission.

Result:



	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	clusters
0	19.94	16.92	0.8752	6.675	3.763	3.252	6.550	1
1	15.99	14.89	0.9064	5.363	3.582	3.336	5.144	3
2	18.95	16.42	0.8829	6.248	3.755	3.368	6.148	1
3	10.83	12.96	0.8099	5.278	2.641	5.182	5.185	2
4	17.99	15.86	0.8992	5.890	3.694	2.068	5.837	1
...	...	...	...	...	...	...	...	...
205	13.89	14.02	0.8880	5.439	3.199	3.986	4.738	3
206	16.77	15.62	0.8638	5.927	3.438	4.920	5.795	1
207	14.03	14.16	0.8796	5.438	3.201	1.717	5.001	3
208	16.12	15.00	0.9000	5.709	3.485	2.270	5.443	1
209	15.57	15.15	0.8527	5.920	3.231	2.640	5.879	3

210 rows × 8 columns

The above attached results shows the addition of last column called “Clusters” where the result on which cluster each row belongs to is shown.

Question 1.4: Apply K-Means clustering on scaled data and determine optimum clusters. Apply elbow curve and silhouette score.

Like the Hierarchical clustering method, here we use the scaled data to make sure that the calculations performed are right as K-Means is highly sensitive to data with different metrics.

First step is to identify the number of optimum clusters that we can use to differentiate each row so that we can form an analysis on their profiles to give recommendations.

For that purpose, we use two methods,

1. The Elbow Method
2. The Silhouette Method

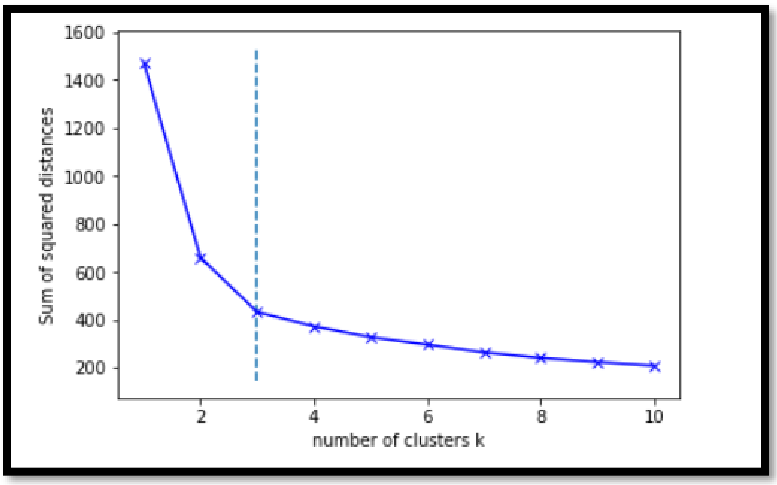
The Elbow Method:

For this method we calculate something called WSS score to determine the optimum number of clusters.

Within-Cluster-Sum of Squared Errors(WSS) score is the sum of the Squared Errors for all the points. Any distance metric like the Euclidean Distance or the Manhattan Distance can be used with the most common being the "Euclidean" distance.

We create a loop passing all values of the number of clusters for "K" there could be and calculate the WSS score for each of them along with inertia (inertia tells how far away the points within a cluster are).

The next step would be to plot the elbow plot which can point to us on the number of clusters that is needed for a dataset.For this problem the make sure that we make no mistakes a library called "KneeLocator" is used to pinpoint on the number of clusters that we can segregate.

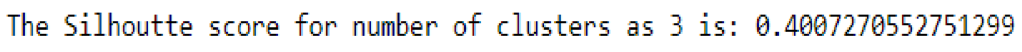


From the above result we can clearly see that the number of clusters is 3.

The Silhoutte Score Method:

The silhouette value measures how similar a point is to its own cluster (cohesion) compared to other clusters (separation)

The values range from -1 to +1. A high value is indicating that we have created an optimum number of clusters. If many points have a negative Silhouette value, it may indicate that we have created too many or too few clusters.Based on the value on the number of optimum clusters from Elbow curve we can pass the 3 clusters for the silhouette score function.



```
The Silhoutte score for number of clusters as 3 is: 0.4007270552751299
```

NOTE: Please note that the same two algorithms were passed on with treating the outliers found in the “min_payment_amt” and “probability of full payment” which came out to be : 0.4008059221522216 which is just a 0.0001 improvement over the dataset without treating the dataset and therefore we can ignore treating the outliers for this dataset.

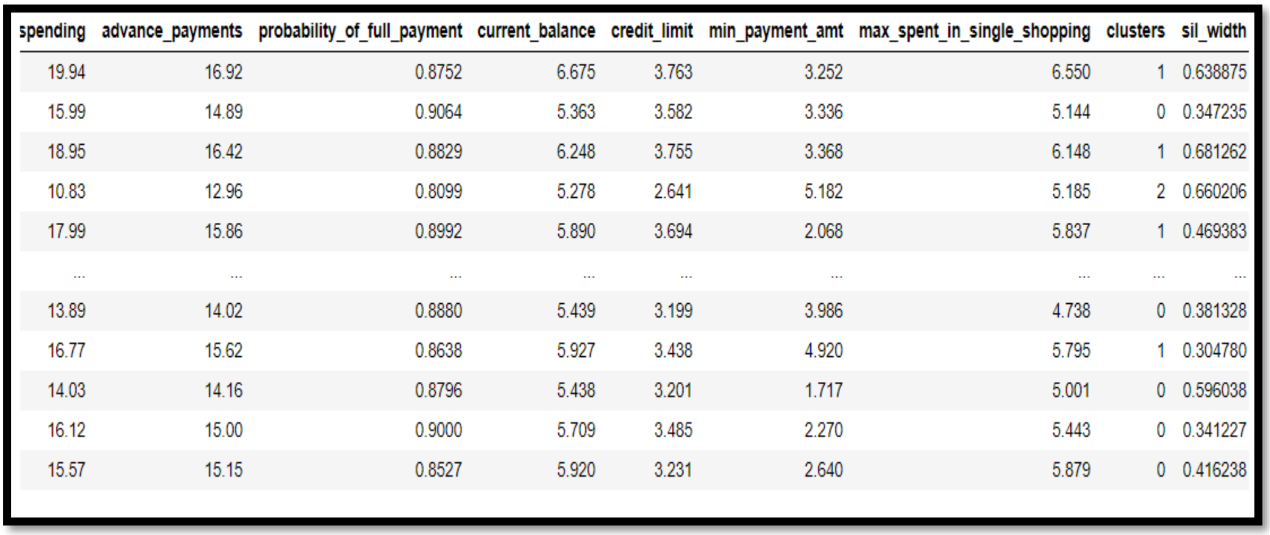
The value indicates that we have relatively made optimum clusters that shows distinctive identification.

We can also add the silhouette width for each row based on clusters to further improve analysis from our side.

Now that we have defined the number of clusters that is optimum, we can go ahead and pass the number of clusters in the KMEANS algorithm to obtain the clusters results.

Like that of Hierarchical clustering, **the results are added in a file called 'kmeans_cluster_result.csv' which is attached to the submission.**

Result:



spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	clusters	sil_width
19.94	16.92	0.8752	6.675	3.763	3.252	6.550	1	0.638875
15.99	14.89	0.9064	5.363	3.582	3.336	5.144	0	0.347235
18.95	16.42	0.8829	6.248	3.755	3.368	6.148	1	0.681262
10.83	12.96	0.8099	5.278	2.641	5.182	5.185	2	0.660206
17.99	15.86	0.8992	5.890	3.694	2.068	5.837	1	0.469383
...	...	...	...	...	...	...	...	...
13.89	14.02	0.8880	5.439	3.199	3.986	4.738	0	0.381328
16.77	15.62	0.8638	5.927	3.438	4.920	5.795	1	0.304780
14.03	14.16	0.8796	5.438	3.201	1.717	5.001	0	0.596038
16.12	15.00	0.9000	5.709	3.485	2.270	5.443	0	0.341227
15.57	15.15	0.8527	5.920	3.231	2.640	5.879	0	0.416238

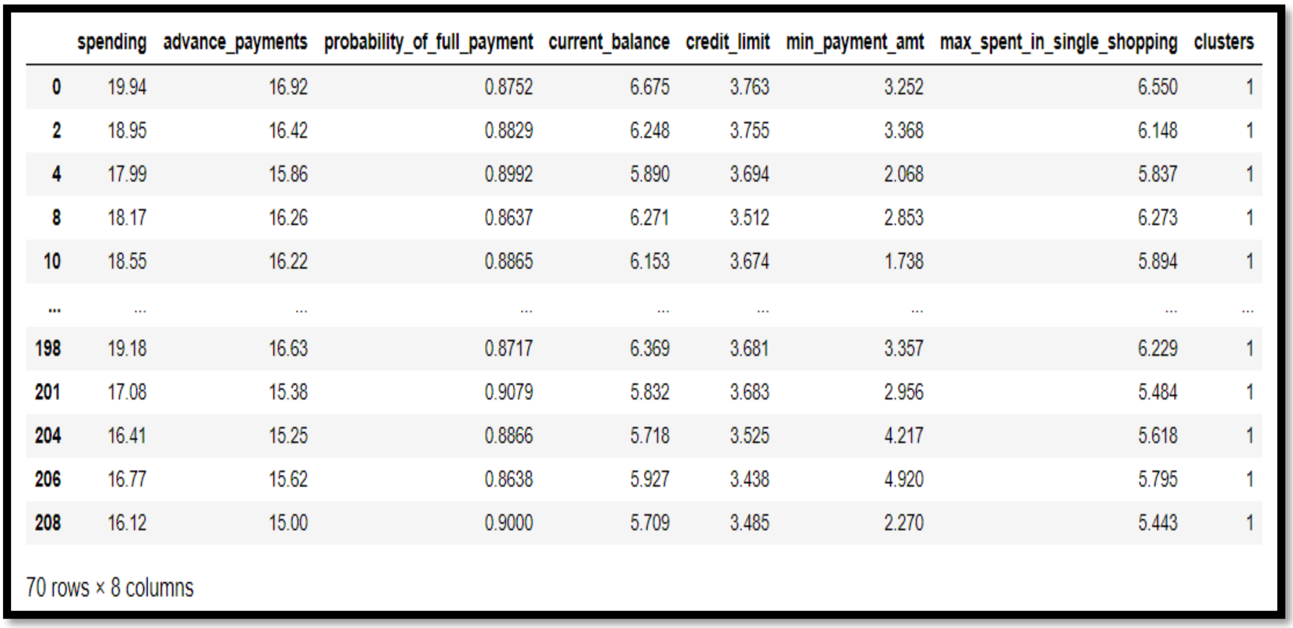
From the above results we can see that each row has been identified as either cluster ‘0’ , ‘1’ or ‘2’ , that we can use to create profiles on them.

Question 1.5 : Describe cluster profiles for the clusters defined. Recommend different promotional strategies for different clusters.

For Hierarchical Clustering:

Firstly, we group the data based on their cluster number to determine their qualities for each variable.

Cluster 1:

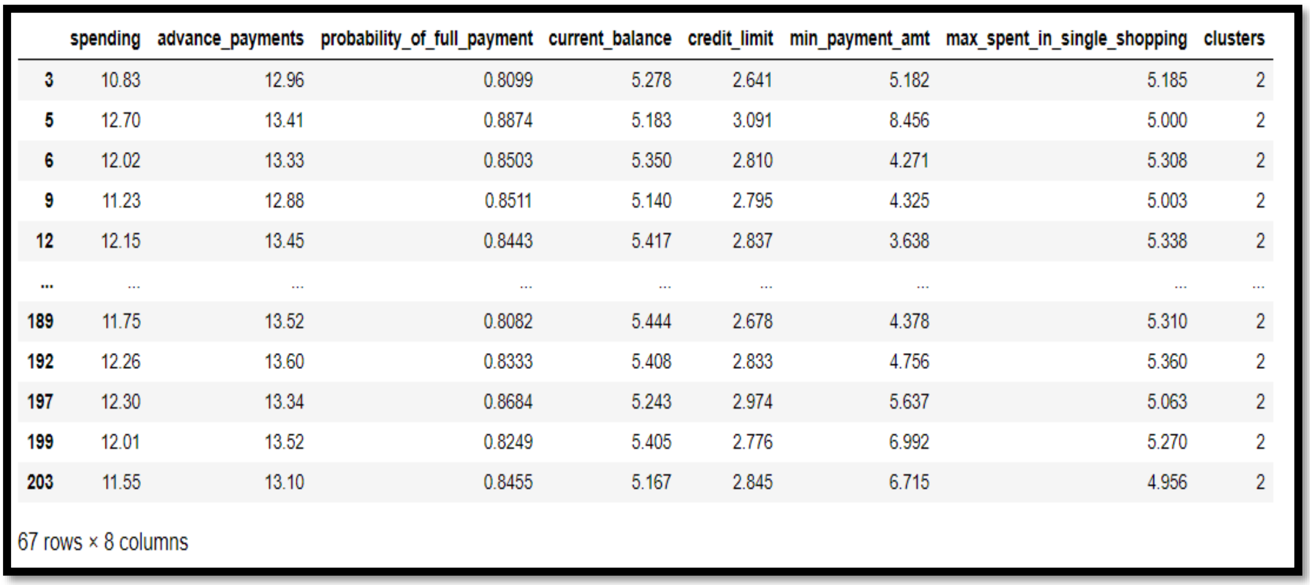


	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	clusters
0	19.94	16.92	0.8752	6.675	3.763	3.252	6.550	1
2	18.95	16.42	0.8829	6.248	3.755	3.368	6.148	1
4	17.99	15.86	0.8992	5.890	3.694	2.068	5.837	1
8	18.17	16.26	0.8637	6.271	3.512	2.853	6.273	1
10	18.55	16.22	0.8865	6.153	3.674	1.738	5.894	1
...	...	...	...	...	...	...	...	...
198	19.18	16.63	0.8717	6.369	3.681	3.357	6.229	1
201	17.08	15.38	0.9079	5.832	3.683	2.956	5.484	1
204	16.41	15.25	0.8866	5.718	3.525	4.217	5.618	1
206	16.77	15.62	0.8638	5.927	3.438	4.920	5.795	1
208	16.12	15.00	0.9000	5.709	3.485	2.270	5.443	1

70 rows × 8 columns

From the above result we can see that the cluster 1 is made of 70 rows (out of 210).

Cluster 2:

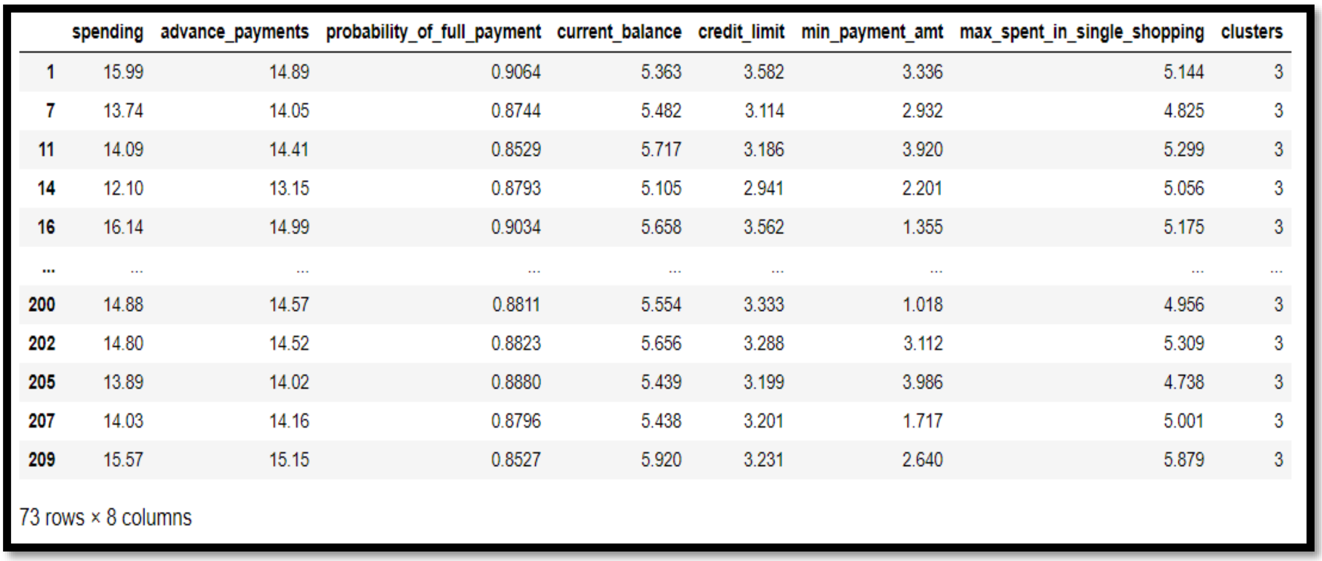


	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	clusters
3	10.83	12.96	0.8099	5.278	2.641	5.182	5.185	2
5	12.70	13.41	0.8874	5.183	3.091	8.456	5.000	2
6	12.02	13.33	0.8503	5.350	2.810	4.271	5.308	2
9	11.23	12.88	0.8511	5.140	2.795	4.325	5.003	2
12	12.15	13.45	0.8443	5.417	2.837	3.638	5.338	2
...	...	...	...	...	...	...	...	...
189	11.75	13.52	0.8082	5.444	2.678	4.378	5.310	2
192	12.26	13.60	0.8333	5.408	2.833	4.756	5.360	2
197	12.30	13.34	0.8684	5.243	2.974	5.637	5.063	2
199	12.01	13.52	0.8249	5.405	2.776	6.992	5.270	2
203	11.55	13.10	0.8455	5.167	2.845	6.715	4.956	2

67 rows × 8 columns

From the above result we can see that the cluster 2 is made of 67 rows(out of 210).

Cluster 3:



	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	clusters
1	15.99	14.89	0.9064	5.363	3.582	3.336	5.144	3
7	13.74	14.05	0.8744	5.482	3.114	2.932	4.825	3
11	14.09	14.41	0.8529	5.717	3.186	3.920	5.299	3
14	12.10	13.15	0.8793	5.105	2.941	2.201	5.056	3
16	16.14	14.99	0.9034	5.658	3.562	1.355	5.175	3
...	...	...	...	...	...	...	...	...
200	14.88	14.57	0.8811	5.554	3.333	1.018	4.956	3
202	14.80	14.52	0.8823	5.656	3.288	3.112	5.309	3
205	13.89	14.02	0.8880	5.439	3.199	3.986	4.738	3
207	14.03	14.16	0.8796	5.438	3.201	1.717	5.001	3
209	15.57	15.15	0.8527	5.920	3.231	2.640	5.879	3

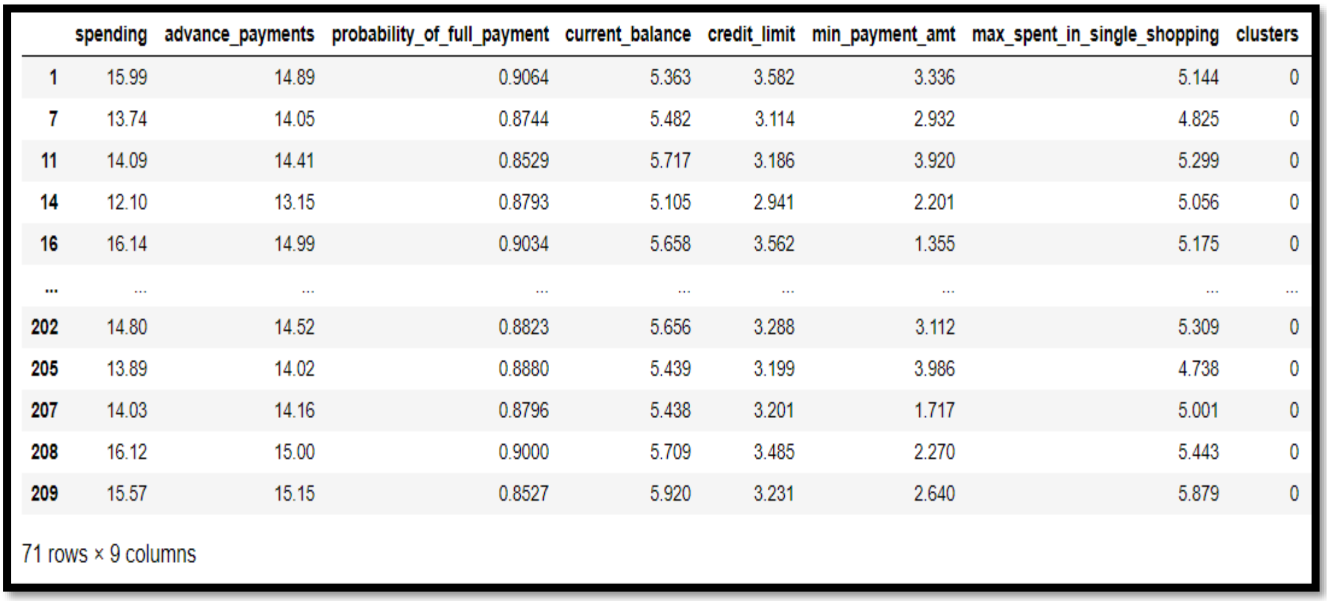
73 rows × 8 columns

From the above result we can see that the cluster 3 is made of 73 rows(out of 210).

For KMeans Clustering:

Firstly, we group the data based on their cluster number to determine their qualities for each variable.

Cluster 0:

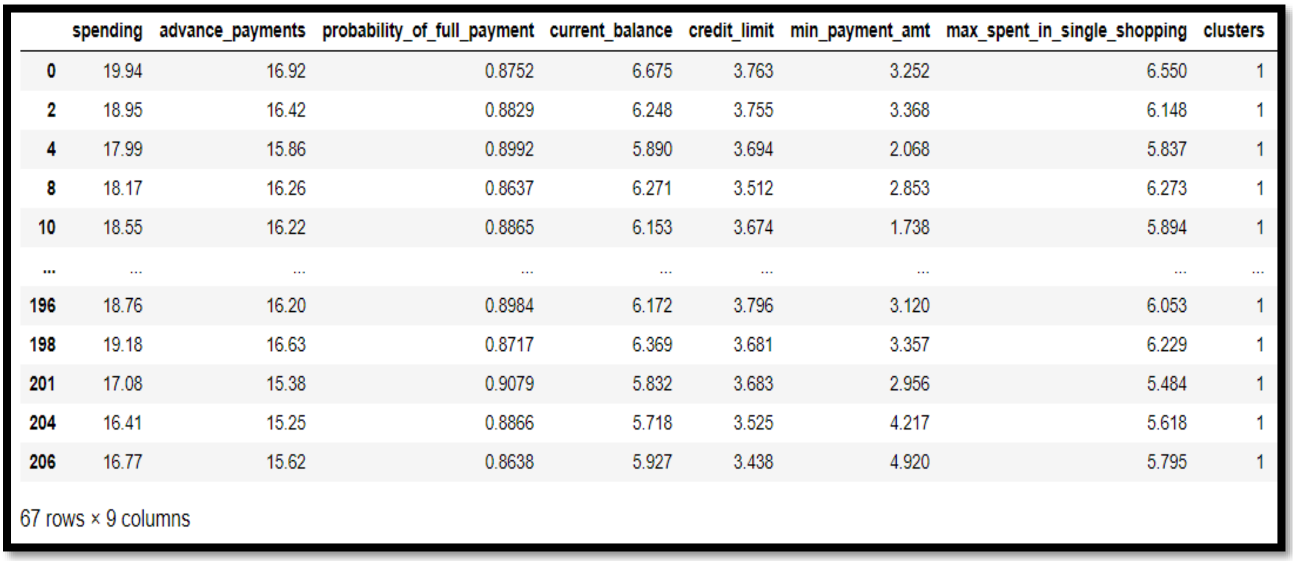


	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	clusters
1	15.99	14.89	0.9064	5.363	3.582	3.336	5.144	0
7	13.74	14.05	0.8744	5.482	3.114	2.932	4.825	0
11	14.09	14.41	0.8529	5.717	3.186	3.920	5.299	0
14	12.10	13.15	0.8793	5.105	2.941	2.201	5.056	0
16	16.14	14.99	0.9034	5.658	3.562	1.355	5.175	0
...	...	...	...	...	...	...	...	...
202	14.80	14.52	0.8823	5.656	3.288	3.112	5.309	0
205	13.89	14.02	0.8880	5.439	3.199	3.986	4.738	0
207	14.03	14.16	0.8796	5.438	3.201	1.717	5.001	0
208	16.12	15.00	0.9000	5.709	3.485	2.270	5.443	0
209	15.57	15.15	0.8527	5.920	3.231	2.640	5.879	0

71 rows × 9 columns

From the above result we can see that the cluster 0 is made of 71 rows(out of 210).

Cluster 1:

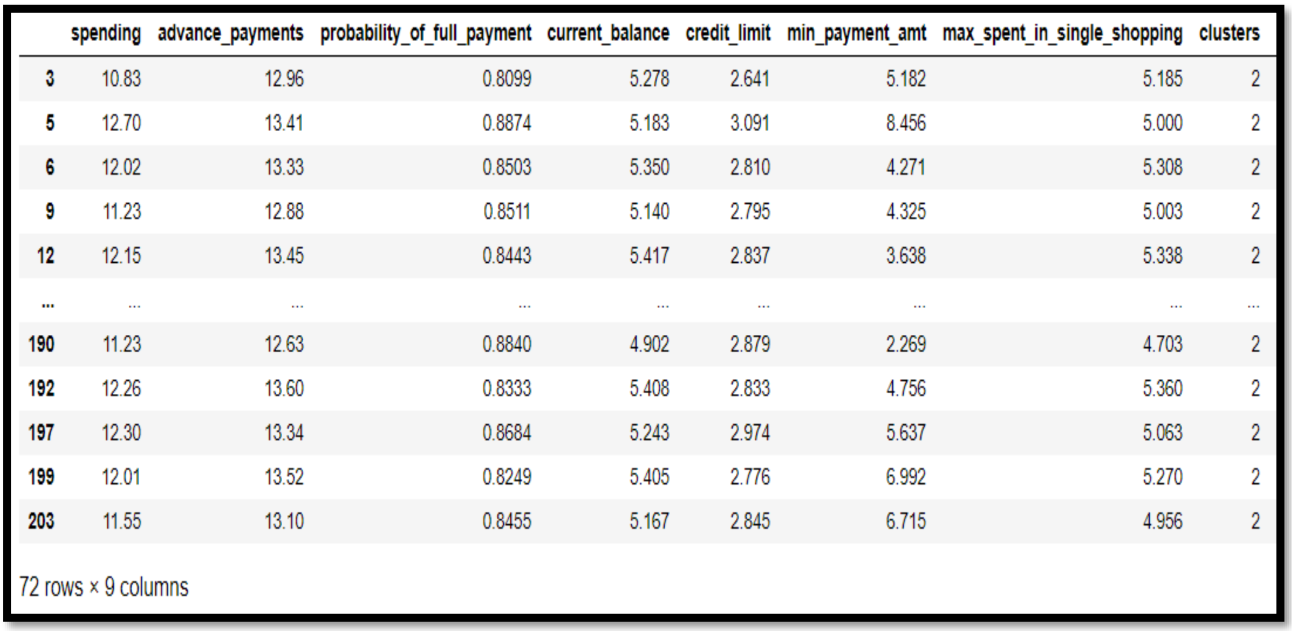


	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	clusters
0	19.94	16.92	0.8752	6.675	3.763	3.252	6.550	1
2	18.95	16.42	0.8829	6.248	3.755	3.368	6.148	1
4	17.99	15.86	0.8992	5.890	3.694	2.068	5.837	1
8	18.17	16.26	0.8637	6.271	3.512	2.853	6.273	1
10	18.55	16.22	0.8865	6.153	3.674	1.738	5.894	1
...	...	...	...	...	...	...	...	...
196	18.76	16.20	0.8984	6.172	3.796	3.120	6.053	1
198	19.18	16.63	0.8717	6.369	3.681	3.357	6.229	1
201	17.08	15.38	0.9079	5.832	3.683	2.956	5.484	1
204	16.41	15.25	0.8866	5.718	3.525	4.217	5.618	1
206	16.77	15.62	0.8638	5.927	3.438	4.920	5.795	1

67 rows × 9 columns

From the above result we can see that the cluster 1 is made of 67 rows(out of 210).

Cluster 2:



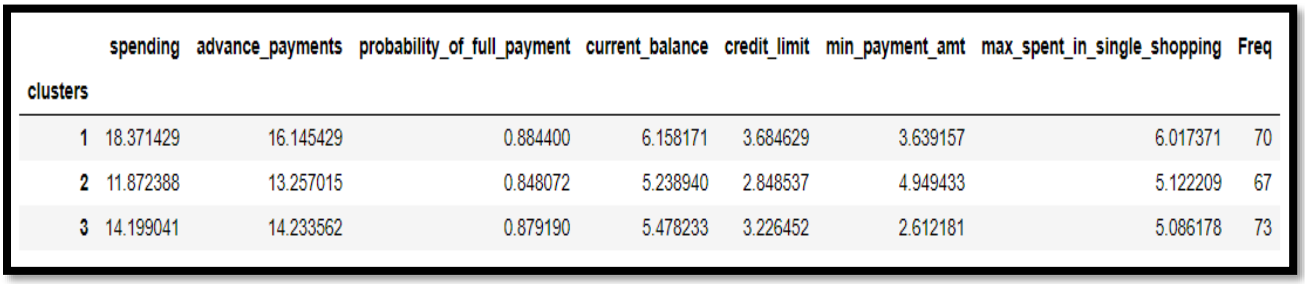
	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	clusters
3	10.83	12.96	0.8099	5.278	2.641	5.182	5.185	2
5	12.70	13.41	0.8874	5.183	3.091	8.456	5.000	2
6	12.02	13.33	0.8503	5.350	2.810	4.271	5.308	2
9	11.23	12.88	0.8511	5.140	2.795	4.325	5.003	2
12	12.15	13.45	0.8443	5.417	2.837	3.638	5.338	2
...	...	...	...	...	...	...	...	...
190	11.23	12.63	0.8840	4.902	2.879	2.269	4.703	2
192	12.26	13.60	0.8333	5.408	2.833	4.756	5.360	2
197	12.30	13.34	0.8684	5.243	2.974	5.637	5.063	2
199	12.01	13.52	0.8249	5.405	2.776	6.992	5.270	2
203	11.55	13.10	0.8455	5.167	2.845	6.715	4.956	2

72 rows × 9 columns

From the above result we can see that the cluster 2 is made of 72 rows(out of 210).

Now that we have the clusters separated , we can go ahead and put a profile on all the three clusters and figure out a recommendation based on analysing them.

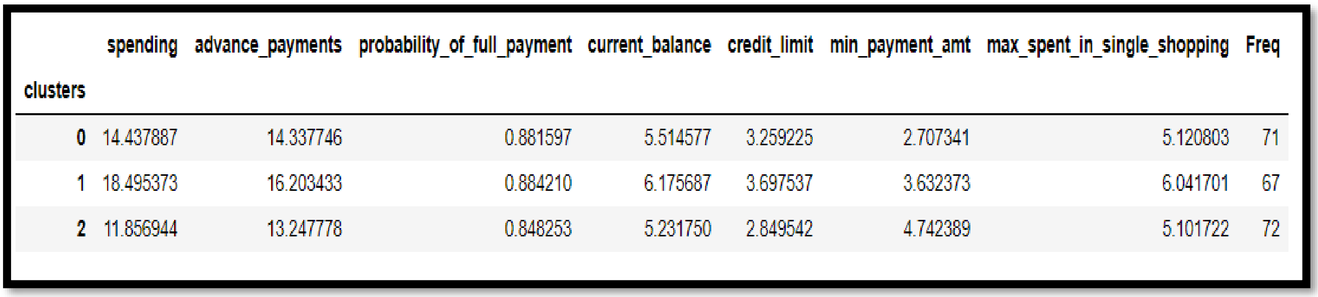
For Hierarchical Clustering:



clusters	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	Freq
1	18.371429	16.145429	0.884400	6.158171	3.684629	3.639157	6.017371	70
2	11.872388	13.257015	0.848072	5.238940	2.848537	4.949433	5.122209	67
3	14.199041	14.233562	0.879190	5.478233	3.226452	2.612181	5.086178	73

The above result is based on the mean calculations for all the variables based on their cluster numbers.

For KMeans Clustering:



clusters	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	Freq
0	14.437887	14.337746	0.881597	5.514577	3.259225	2.707341	5.120803	71
1	18.495373	16.203433	0.884210	6.175687	3.697537	3.632373	6.041701	67
2	11.856944	13.247778	0.848253	5.231750	2.849542	4.742389	5.101722	72

The above result is based on the mean calculations for all the variables based on their cluster numbers.

The above two results on the aggregation of means based on their cluster numbers for both methods show similarities between them.

Like,

- **Cluster 1 of Hierarchical is like Cluster 1 of KMeans clustering algorithm**
- **Cluster 2 of Hierarchical is like Cluster 2 of KMeans clustering algorithm**
- **Cluster 3 of Hierarchical is like Cluster 0 of KMeans clustering algorithm**

Since the two methods have produced similar outputs(with slight change in cluster size and mean score, they are not significant enough to check which method works better), we can go ahead with the results and provide recommendations based on the values obtained.

Recommendations on promotional strategies:

Cluster 2 of Hierarchical is like Cluster 2 of KMeans clustering algorithm:

- These belong to the group who spends the least with their credit cards, as they are clearly seen in their average part of "spending" , "advance_payments" , "probability_of_full_payment" , "current_balance" and "cred_limit".
- These maybe because of the low income of the customers in the group(comparing the rest of the clusters).
- The "min_payment_amt" shows variation to the rest of the which is normal as low income groups can pay more in min_payment_amount since their purchase amount would be less compared to the rest and the important fact is that they are monthly(not years) and therefore they are looking it in short term and try to complete it soon.
- The recommendation would be to:
    1. Reduce the minimum balance limit to the group(near null) to make sure that they are loyal to the bank
    2. Start with the reward points method to make sure that there is no blocking purpose for the usage of credit cards.
    3. Adding extra EMI options to further facilitate trust on further continuing with the bank.

Cluster 3 of Hierarchical is like Cluster 0 of KMeans clustering algorithm:

- These belong to the group who spends the averagely with their credit cards, as they are clearly seen in their average part of "spending" , "advance_payments" , "probability_of_full_payment" , "current_balance" and "cred_limit".
- These maybe because of the average income of the customers in the group(comparing the rest of the clusters).
- The "min_payment_amt" shows variation to the rest of the variables, which is normal as average income groups will pay less compared to low income groups in this criteria(in min_payment_amount) since their purchase amount would be a little more than compared to the low income group and the important fact is that they are monthy(not years) and therefore they are looking a long term payment plan.
- The recommendation would be to:
    1. The minimum balance limit can be slightly increased as these group.
    2. Reward points are to be focused here more than any other groups as these are the main attraction for this groups.
    3. Make attractive offers for these groups to make sure that they are loyal to the bank

Cluster 1 of Hierarchical is like Cluster 1 of KMeans clustering algorithm:

- These belong to the group who spends the more with their credit cards, as they are clearly seen in their average part of "spending" , "advance_payments" , "probability_of_full_payment" , "current_balance" and "cred_limit".
- These maybe because of the high income of the customers in the group(comparing the rest of the clusters).
- The "min_payment_amt" shows variation to the rest of the variables, which is normal as high income groups will pay averagely(not giving in too much) compared to average income groups in this criteria (in min payment amount) since their purchase amount would be a little more than compared to the average income group and the important fact is that they are monthy(not years) and therefore they are looking a long term payment plan..
- The recommendation would be to:
    1. Add offers like Gift Vouchers along with the reward points for these customers as they are already providing a better fit the credit card schemes.
    2. Can give a no limit credit card(not exactly but increase the credit-card usage limit) so that they are comfortable in using them frequently when required.

Problem 1 Summary:

- **1.1)** All the basic EDA along with the univariate and the Bivariate analysis were performed and analysed.
- **1.2)** Standard Scaler is used here and the reason for that have been mentioned.
- **1.3)** Hierarchical clustering is performed on scaled data. The number of optimum clusters are found using Dendrogram and the results are added to a CSV file for analysis
- **1.4)** K-Means clustering is performed on scaled data and the number of optimum clusters is determined. Elbow curve and silhouette score is applied on the data to further add value to determination of the optimum clusters.
- **1.5)** Cluster profiles for the clusters are defined. Recommendations and different promotional strategies are provided for these different clusters.

Conclusion:

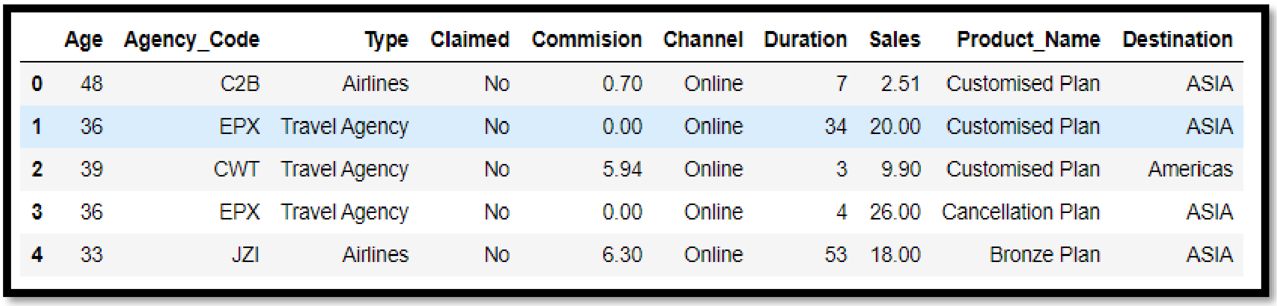
Looking into the data of “bank_marketing_part1_Data.csv”, we saw some interesting insights from EDA methods. We were able to perform two clustering methods on the scaled form of these data to give optimum clusters that the business side can easily utilize to understand the mindset and frequent activities of the customers to provide them better services . Also, various promotional offers could be provided that customers from these three segments which can benefit both customers and the Bank.

Problem 2: CART-RF-ANN

An Insurance firm providing tour insurance is facing higher claim frequency. The management decides to collect data from the past few years. You are assigned the task to make a model which predicts the claim status and provide recommendations to management. Use CART, RF & ANN and compare the models' performances in train and test sets.

Question 2.1 : Data Ingestion: Read the dataset. Do the descriptive statistics and do null value condition check, write an inference on it.

Exploratory Data Analysis:



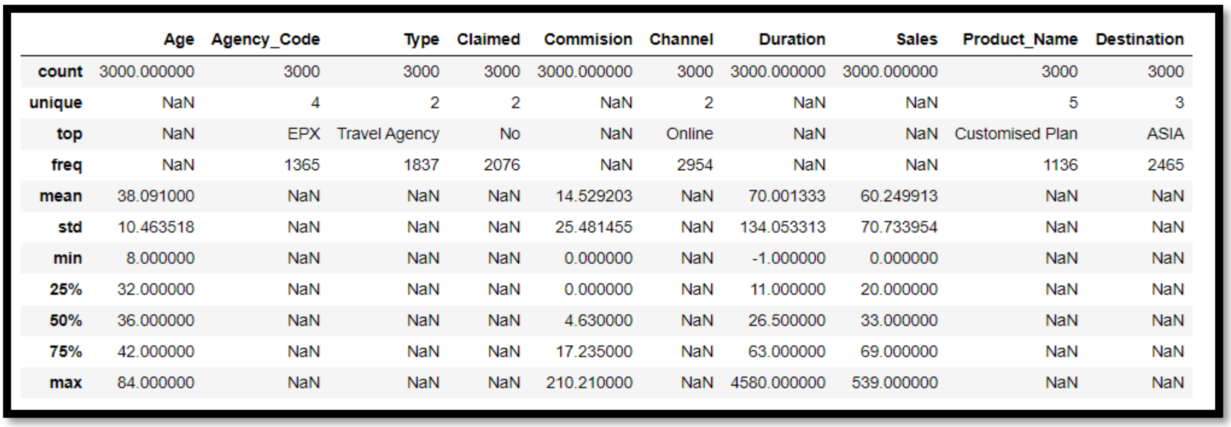
	Age	Agency_Code	Type	Claimed	Commision	Channel	Duration	Sales	Product_Name	Destination
0	48	C2B	Airlines	No	0.70	Online	7	2.51	Customised Plan	ASIA
1	36	EPX	Travel Agency	No	0.00	Online	34	20.00	Customised Plan	ASIA
2	39	CWT	Travel Agency	No	5.94	Online	3	9.90	Customised Plan	Americas
3	36	EPX	Travel Agency	No	0.00	Online	4	26.00	Cancellation Plan	ASIA
4	33	JZI	Airlines	No	6.30	Online	53	18.00	Bronze Plan	ASIA

We have 10 variables in the dataset.

Numerical Variables : Age, Commision, Duration and Sales.

Categorical Variables: Agency_code, Type, Claimed, Channel, Product_Name and Destination.

Descriptive Statistics of the Dataset:



	Age	Agency_Code	Type	Claimed	Commision	Channel	Duration	Sales	Product_Name	Destination
count	3000.000000	3000	3000	3000	3000.000000	3000	3000.000000	3000.000000	3000	3000
unique	NaN	4	2	2	NaN	2	NaN	NaN	5	3
top	NaN	EPX	Travel Agency	No	NaN	Online	NaN	NaN	Customised Plan	ASIA
freq	NaN	1365	1837	2076	NaN	2954	NaN	NaN	1136	2465
mean	38.091000	NaN	NaN	NaN	14.529203	NaN	70.001333	60.249913	NaN	NaN
std	10.463518	NaN	NaN	NaN	25.481455	NaN	134.053313	70.733954	NaN	NaN
min	8.000000	NaN	NaN	NaN	0.000000	NaN	-1.000000	0.000000	NaN	NaN
25%	32.000000	NaN	NaN	NaN	0.000000	NaN	11.000000	20.000000	NaN	NaN
50%	36.000000	NaN	NaN	NaN	4.630000	NaN	26.500000	33.000000	NaN	NaN
75%	42.000000	NaN	NaN	NaN	17.235000	NaN	63.000000	69.000000	NaN	NaN
max	84.000000	NaN	NaN	NaN	210.210000	NaN	4580.000000	539.000000	NaN	NaN

The describe function(describe(include= ‘all’)) provides us with the information about how much the data is spread across along with the information on the mean, standard deviation, count etc.

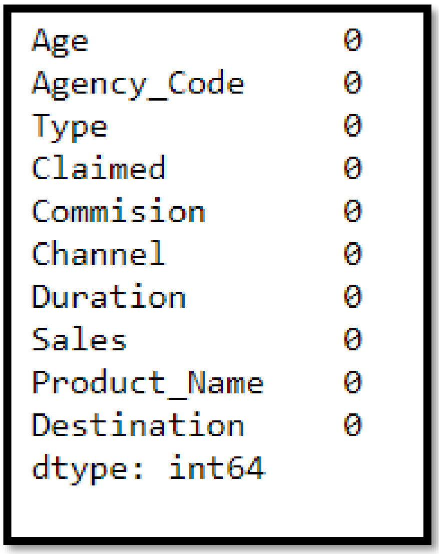
We have many categorical variables with most frequency seen in :

Agency_code as ‘**EPX**’, Type as **‘TravelAgency’** , Claimed as **‘No’**, Channel as **‘Online’**, Product_Name as **‘Customised Plan’** and Destination as **‘ASIA’**.

Also, we can see that for some variables we have standard deviation more the mean which suggests that the data might not be as normally distributed as we want.

Note : There is an invalid data in the column called “Duration”(-1,0) which will be treated later.

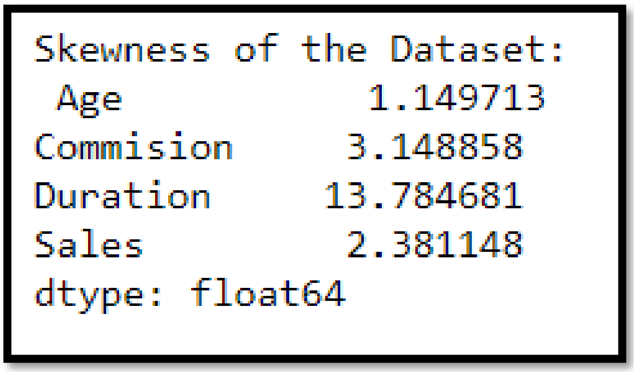
Check for NULL Values:



```
Age             0
Agency_Code     0
Type            0
Claimed         0
Commision       0
Channel         0
Duration        0
Sales           0
Product_Name    0
Destination     0
dtype: int64
```

The isnull() and sum() function combined can clearly figure out on whether given dataset has any NULL(N/A) values or not. From the above result it is evident that there are no NULL values and therefore we can proceed further in the Exploratory Data Analysis.

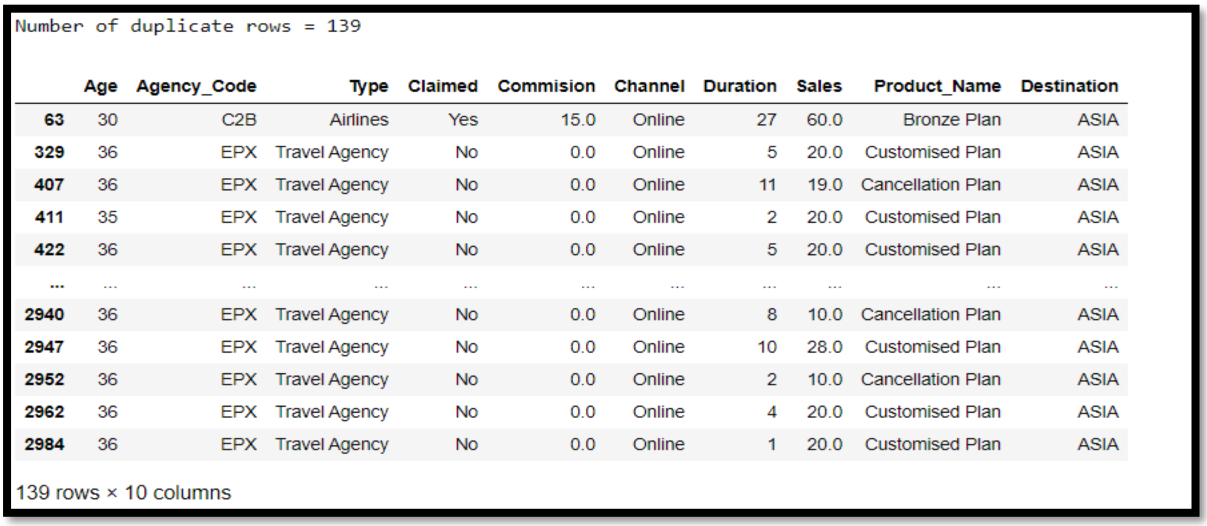
Skewness Check:



```
Skewness of the Dataset:
 Age            1.149713
Commision      3.148858
Duration      13.784681
Sales          2.381148
dtype: float64
```

Skewness is a measure of the asymmetry of the probability distribution of a real-valued random variable about its mean. From the above result we can see that only the variable “Duration” is highly positively skewed compared everything else(which are right skewed as well).

Duplicates Check:



```
Number of duplicate rows = 139
```

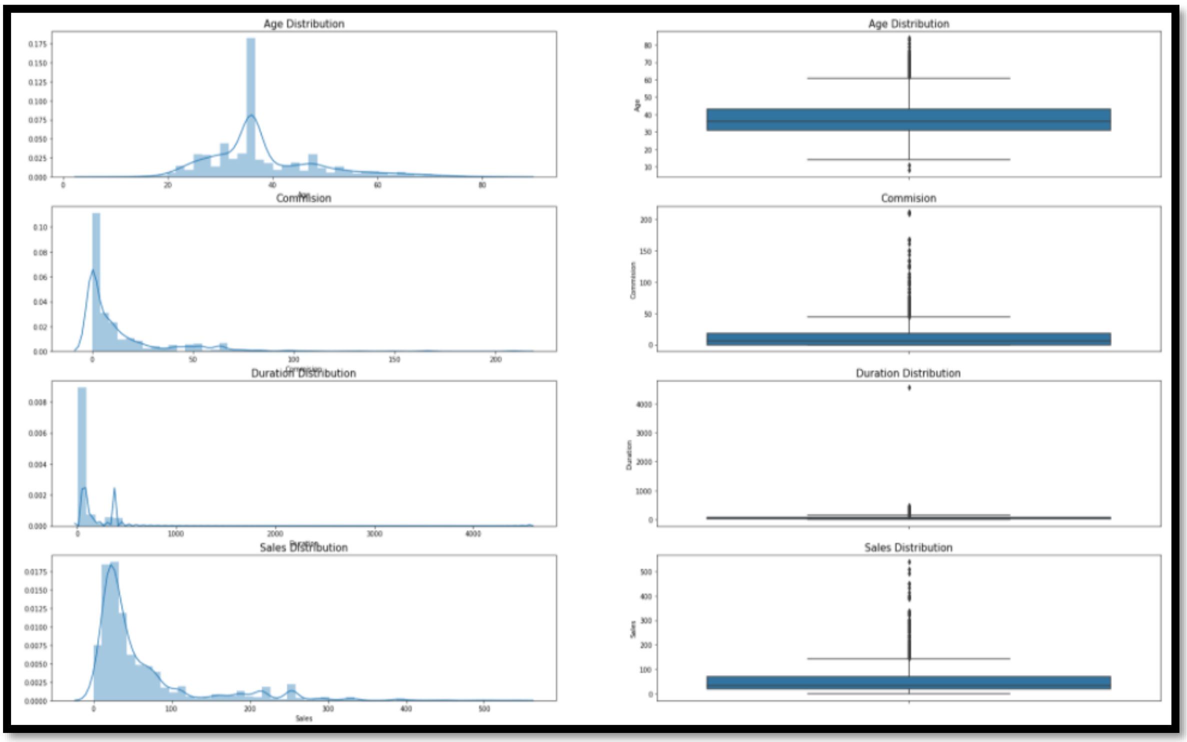
	Age	Agency_Code	Type	Claimed	Commision	Channel	Duration	Sales	Product_Name	Destination
63	30	C2B	Airlines	Yes	15.0	Online	27	60.0	Bronze Plan	ASIA
329	36	EPX	Travel Agency	No	0.0	Online	5	20.0	Customised Plan	ASIA
407	36	EPX	Travel Agency	No	0.0	Online	11	19.0	Cancellation Plan	ASIA
411	35	EPX	Travel Agency	No	0.0	Online	2	20.0	Customised Plan	ASIA
422	36	EPX	Travel Agency	No	0.0	Online	5	20.0	Customised Plan	ASIA
...	...	...	...	...	...	...	...	...	...	...
2940	36	EPX	Travel Agency	No	0.0	Online	8	10.0	Cancellation Plan	ASIA
2947	36	EPX	Travel Agency	No	0.0	Online	10	28.0	Customised Plan	ASIA
2952	36	EPX	Travel Agency	No	0.0	Online	2	10.0	Cancellation Plan	ASIA
2962	36	EPX	Travel Agency	No	0.0	Online	4	20.0	Customised Plan	ASIA
2984	36	EPX	Travel Agency	No	0.0	Online	1	20.0	Customised Plan	ASIA

139 rows × 10 columns

The calculation part is done in the notebook attached and we can see that there are 139 Duplicates present in the dataset. Therefore, we need to remove these duplicates so that we can move further in Exploratory Data Analysis.

Note: These 139 variables are not exactly pure duplicates(two to three columns are always having different values), but for the purpose of this project, we are going to remove these records and continue further. But this decision should be taken after consulting with the Business side before concluding.

Univariate Data Visualization (Dist-plot and Boxplot analysis):

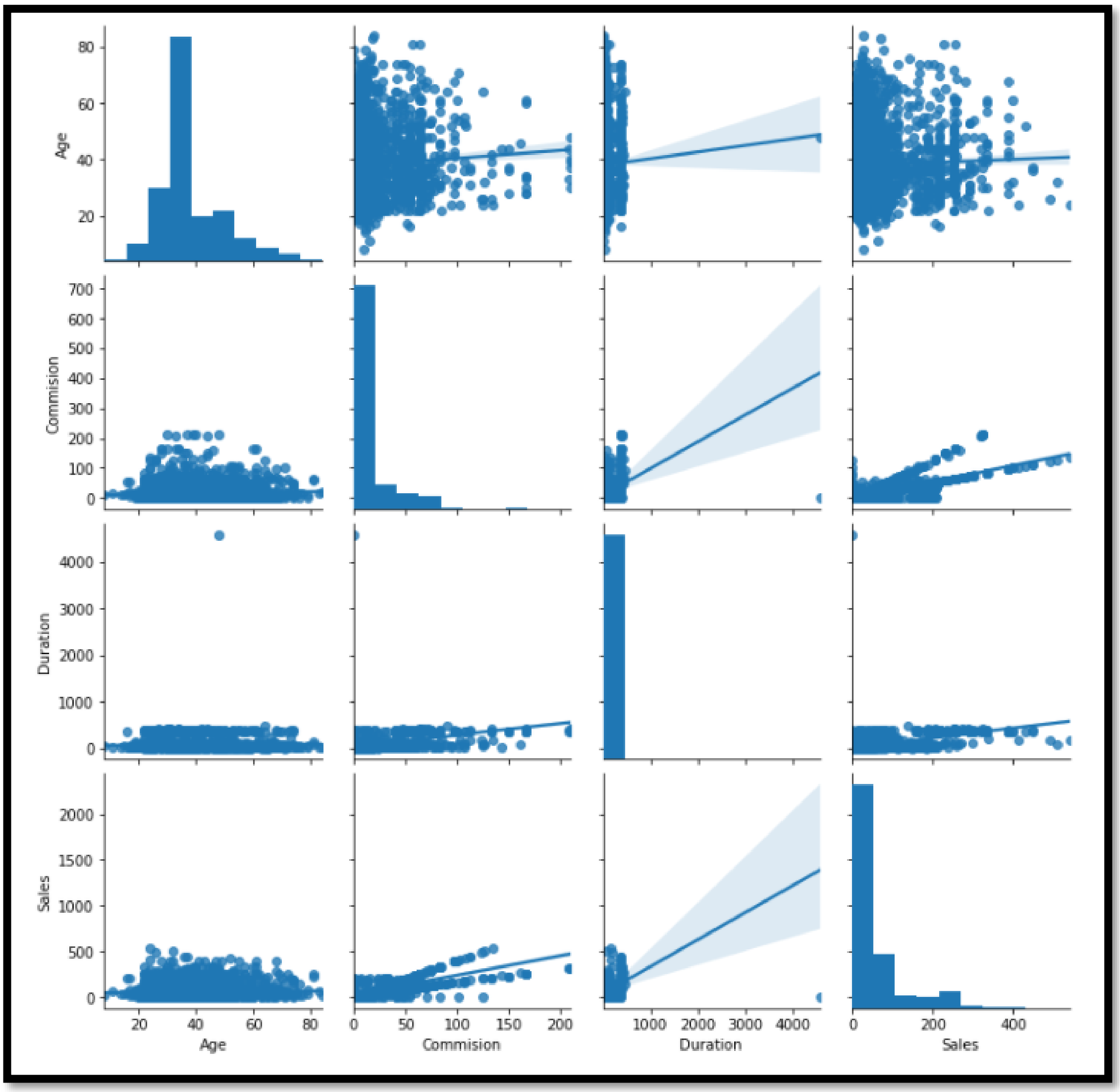


We can see that all the four numerical Variables are normally distributed(not perfectly normal though, but still). There are a lot of outliers that can be seen from the boxplots on the right too.

For the purposes of this project we are going to move forward with dataset which was not treated with outliers as well as with the dataset that was treated with the outlier to better understand how the algorithms work as well to see if the accuracy increases without treatment.

But the one common treatment was performed on both the datasets was on the column called “Duration” . This column contains an invalid data called -1 and 0 as there can be no 0 and -1 in the dataset. These values were replaced by the ‘Median” values and not the “Mean” values as there are many outliers in the dataset(Mean is sensitive to outliers).

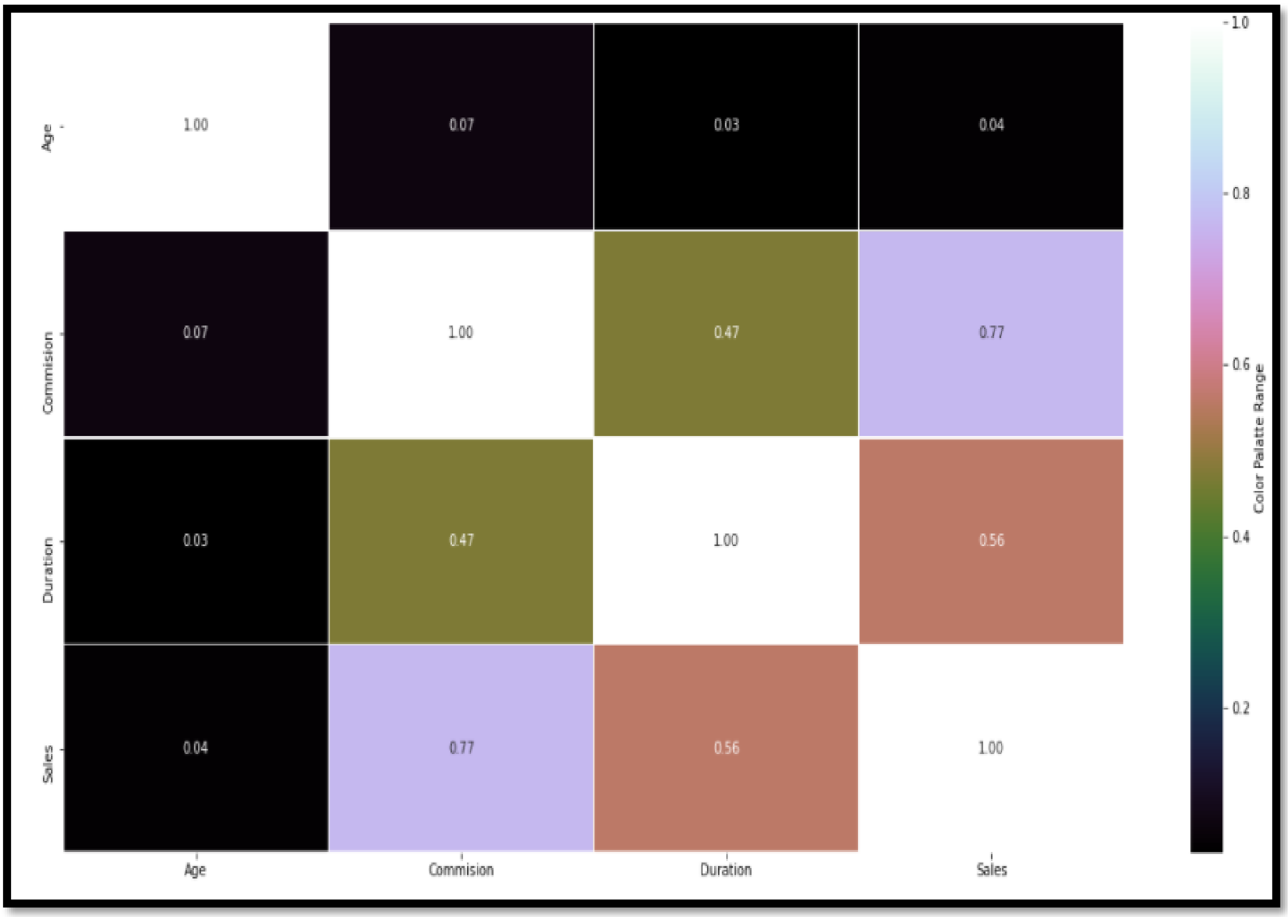
Bivariate Data Visualization (Pair-plot analysis):



A regular Pair-Plot analysis is first performed on the dataset to see their interaction with each of the other variables. From the above result we can say that there are not many strong relationships with the variables. The only notable relationship can be seen between the variables “Commision” and “Sales”. Aslo there are no negative relationship seen here.

The pair-plot analysis can give us only a rough estimate that there is a linear relationship between the variables and its evident here that we do not many strong relationships.

Bivariate Data Visualization (Heat Map analysis):



Another most important plot that we must look in the case of analysis is the Correlation plot. The Heat-Map analysis is an visualized part of the Correlation matrix that is used to find the relationship between the variables with numerical values to standpoint on how much they are related as compared to other variables.

The following conclusions can be sought out of the above results:

- The highest correlation is seen between “Commision” and “sales” (77%)
- Notable positive relationships are between “Duration” and “Sales”(56%) as well as “Duration” and “Commision”(47%)
- Also there are no negative correlation between any of the four numerical values of the dataset.

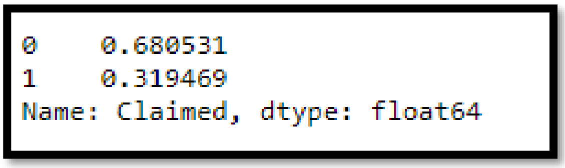
Question 2.2 : Data Split: Split the data into test and train, build classification model CART, Random Forest, Artificial Neural Network

Note : Before the Data splitting and further moving into the Model Building parts, we must remember that we have many categorical variables in the dataset, and they must be converted to numerical type before proceeding further. The reason being all the models mentioned in the question namely “CART”, “Random Forest” and “Artificial Neural Network” only deals with numerical fields.

We also must be clear on the column that we use as a target variable to perform classification. The column that is focused on is “Claimed”, i.e, whether a person has claimed the insurance or not.

Also its important to know about the percentage of distribution of the Target(Claimed) variable as to identify whether there are any class imbalance there or not. Class Imbalance can hugely affect the performance of a model as any huge shortage of a particular value can become hard in training and identifying the variable in real case scenario.

Based on the above statement,



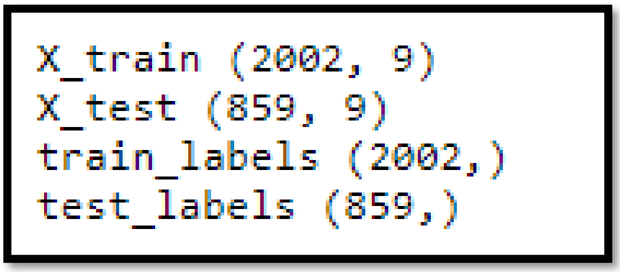
```
0    0.680531
1    0.319469
Name: Claimed, dtype: float64
```

We can see that the values are of 68-32 ratio which is of class imbalance(but not affecting too much as its accepted if the ration is anywhere above 70-30 ratio).

Data Split: Splitting the data into test and train:

We are going to split the data with Train : 70% and Test : 30% with random_state = 1 which is going to be common for both the datasets(treated with outliers and without treating outliers).

After the Train-Test split the shape of the dataset is going to be:



```
X_train (2002, 9)
X_test (859, 9)
train_labels (2002,)
test_labels (859,)
```

Here,

X_train denotes the 70% training dataset that was picked up in random(especially random state = 1) with 9 columns (except the target column called "Claimed").

X_test denotes 30% test dataset that was picked up in random(especially random state = 1) with 9 columns (except the target column called "Claimed").

train_labels denotes the 70% training dataset that was picked up in random(especially random state = 1) with only the target column called "Claimed".

test_labels denotes 30% test dataset that was picked up in random(especially random state = 1) with only the target column called "Claimed".

Model Building:

CART from dataset that has not been treated with outliers:

First step in the model building is to define the classifier. The criterion for creating the Decision Tree Classifier is "gini".

The next logical step would be fitting the model that was built to the training datasets.

After fitting the model, the next step is to create a dot file that contains all the details on the classification tree built from fitting the training datasets.

Upon viewing the criteria via graphical method using "webgraphviz" we can see that the tree is fully grown and too complex to come to a conclusion that it is overfitting(provides significant results in the training, but fail to replicate nearly similar in real life testing scenario) and needs pruning for better understanding the criteria for the split and easy explanation.

For performing pruning(basically trimming the fully outgrown tree to a level that avoids both overfitting and underfitting of data), we need to take care of few parameters that make up the CART model.

Parameters:

- max_depth
- min_samples_leaf
- min_samples_split

These are the parameters that are usually tweaked to bring the best possible model for any dataset(there are more parameters to consider, but these usually form most of the criterion for better built model).

For tweaking these parameters, a method called "GridSearchCV" is performed that can-do multiple combinations of these parameters simultaneously and can provide us with the best optimum results.

Based on the tweaking of parameters the best possible combination came out to be

from

- 'max_depth': [4,5,6,7,8],
- 'min_samples_leaf': [30,35, 40,45,50],
- 'min_samples_split': [60,75, 90,100,120,150]

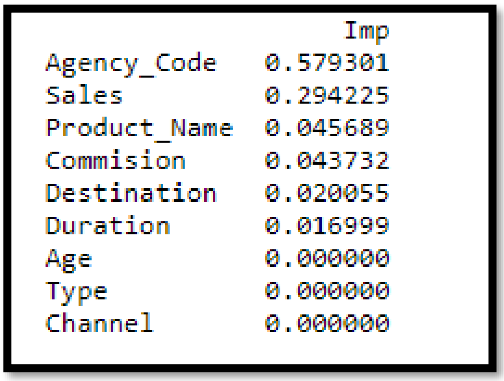
To

- max_depth = 4,
- min_samples_leaf = 30,
- min_samples_split = 120

There can be many such addition to the GridSearchCV to provide the optimum result, but the system cannot handle such aa huge combination especially when mentioned CV = 3(CV is nothing but Cross Validation the randomly shifts the training and test location in the dataset). The reason being the above example handles 5*5*6*3 = 450 combinations of CART models to decide on the optimum one.

After figuring out the parameters that we are going to use to the model, we rebuild the model with the aforementioned parameters and refit the model to check the changes that we have brought.

Also, we can look out for various features like "Feature Importance" that can Help us identify the important variables in the dataset. After rebuilding the model, the Feature Importance came out to be:

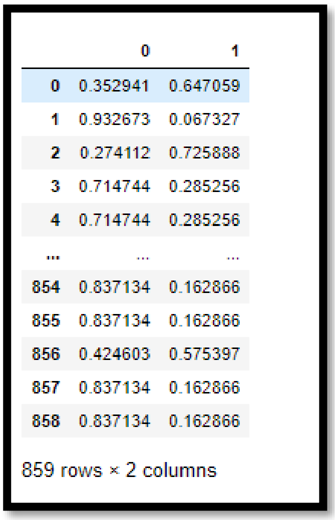


```
                    Imp
Agency_Code    0.579301
Sales          0.294225
Product_Name   0.045689
Commision      0.043732
Destination    0.020055
Duration       0.016999
Age            0.000000
Type           0.000000
Channel        0.000000
```

From the above result we can see that the variable called "Agency_Code" holds the maximum importance(57.9%), followed by "sales"(29.4%) and distant followers "Product_Name"(4.5%) , "Commision"(4.3%),"Destination"(2%) and "Duration"(1%).

Features like "Age", "Type" and "Channel" are showing 0 which says that these variables can be ignored.

For the next step we can create a DataFrame that can clearly define for each record on how much likelihood that they belong to category "0" or "1"(0 being not claimed and 1 being claimed).



	0	1
0	0.352941	0.647059
1	0.932673	0.067327
2	0.274112	0.725888
3	0.714744	0.285256
4	0.714744	0.285256
...	...	...
854	0.837134	0.162866
855	0.837134	0.162866
856	0.424603	0.575397
857	0.837134	0.162866
858	0.837134	0.162866

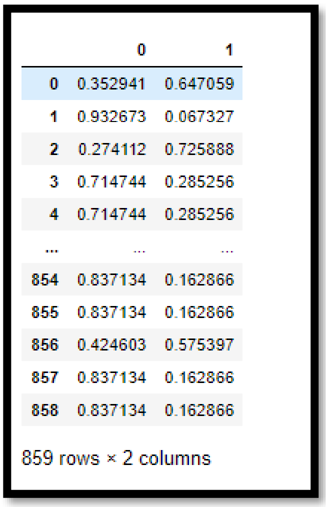
859 rows × 2 columns

We can clearly see from the above result that the newly built model is tested with the “Test” dataset(30% - 859 records) and the results on probability on which side each test record belongs to is shown.

CART from dataset that has not been treated with outliers:

Like the model building that happened without treating the outliers, here we are going to follow the same steps after treating the dataset with the outliers. The treatment of outliers is nothing but changing the values that are more than that upper whisker and less than lower whisker is shifted to the whisker values near to them.

The parameters and the rebuilding of model after pruning(this time with the dataset that was treated with outliers) have shown the same results in prediction to that of the model that was built without treating outliers which can tell us clearly that outlier treatment has no significant effect to the result.



	0	1
0	0.352941	0.647059
1	0.932673	0.067327
2	0.274112	0.725888
3	0.714744	0.285256
4	0.714744	0.285256
...	...	...
854	0.837134	0.162866
855	0.837134	0.162866
856	0.424603	0.575397
857	0.837134	0.162866
858	0.837134	0.162866

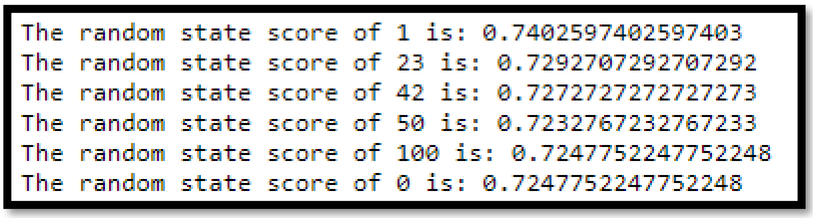
859 rows × 2 columns

We can see further understand from the results predicted with the 30% of test data that the probabilities look exactly the same as last time.

RANDOM FOREST CLASSIFIER without treating outliers:

Random forest is the next classification model that we are going to train with the dataset that didn't treat any outliers.

Firstly, for Random Forest we can even control the random states that we fix in building and predicting the model.



```
The random state score of 1 is: 0.7402597402597403
The random state score of 23 is: 0.7292707292707292
The random state score of 42 is: 0.7272727272727273
The random state score of 50 is: 0.7232767232767233
The random state score of 100 is: 0.7247752247752248
The random state score of 0 is: 0.7247752247752248
```

We can see that for randomly chosen values for Random State, the Random State 1 found to be effective than the rest and therefore we can continue with Random State as 1.

Similar to the CART model we are bound to pruning for the Random Forest model as well and therefore we need to optimize certain parameters.

Parameters:

- max_depth
- min_samples_leaf
- min_samples_split
- max_features
- n_estimators

These are the parameters that are usually tweaked to bring the best possible model for any dataset(there are more parameters to consider, but these usually form most of the criterion for better built model).

For tweaking these parameters, a method called "GridSearchCV" is performed that can-do multiple combinations of these parameters simultaneously and can provide us with the best optimum results.

Based on the tweaking of parameters the best possible combination came out to be

from

- 'max_depth': [5,6,7],
- 'max_features': [7,8,9],
- 'min_samples_leaf': [20, 25,30, 50],
- 'min_samples_split': [45, 60,75,80],
- 'n_estimators': [101,201,301]

To

- max_depth = 7
- max_features = 7
- min_samples_leaf = 20
- min_samples_split = 45
- n_estimators = 201

There can be many such addition to the GridSearchCV to provide the optimum result, but the system cannot handle such aa huge combination especially when mentioned CV = 3(CV is nothing but Cross Validation the randomly shifts the training and test location in the dataset). The reason being the above example handles 3*3*4*4*3*3 = 1296 combinations of Random Forest models to decide on the optimum one.

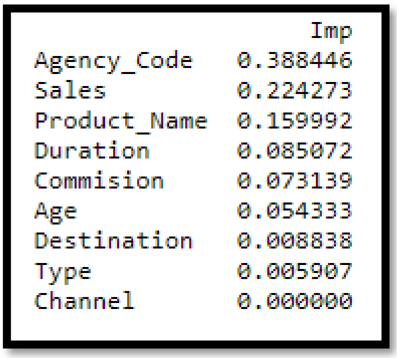
After figuring out the parameters that we are going to use to the model, we build the model with the aforementioned parameters and refit the model to check the changes that we have brought.

The major importance is also given to a parameter called the OOB score (Out Of Bag Score). Out-Of-Bag (OOB) error, also called out-of-bag estimate, is a method of measuring the prediction error of random forests. It is also one of the accuracy measures for Random Forest.

OOB score for the model   : 0.7677322677322678

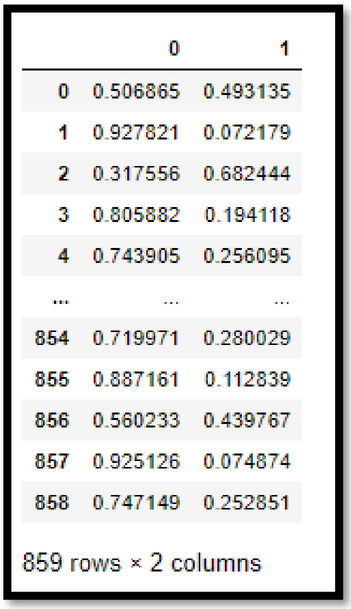
OOB error : 1 – OOB score : 0.2322677322677322 (which is actually a good score)

Also, we can look out for various features like "Feature Importance" that can Help us identify the important variables in the dataset. After rebuilding the model, the Feature Importance came out to be:



```
                   Imp
Agency_Code   0.388446
Sales         0.224273
Product_Name  0.159992
Duration      0.085072
Commision     0.073139
Age           0.054333
Destination   0.008838
Type          0.005907
Channel       0.000000
```

For the next step we can create a DataFrame that can clearly define for each record on how much likelihood that they belong to category "0" or "1"(0 being not claimed and 1 being claimed).



	0	1
0	0.506865	0.493135
1	0.927821	0.072179
2	0.317556	0.682444
3	0.805882	0.194118
4	0.743905	0.256095
...	...	...
854	0.719971	0.280029
855	0.887161	0.112839
856	0.560233	0.439767
857	0.925126	0.074874
858	0.747149	0.252851

859 rows × 2 columns

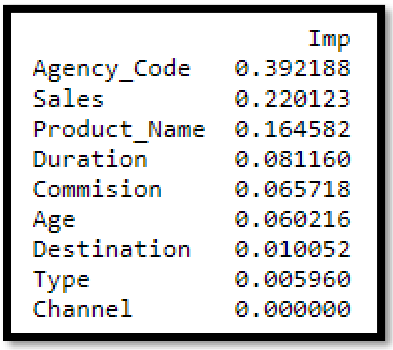
We can clearly see from the above result that the newly built model is tested with the "Test" dataset(30% - 859 records) and the results on probability on which side each test record belongs to is shown.

Random Forest from dataset that has not been treated with outliers:

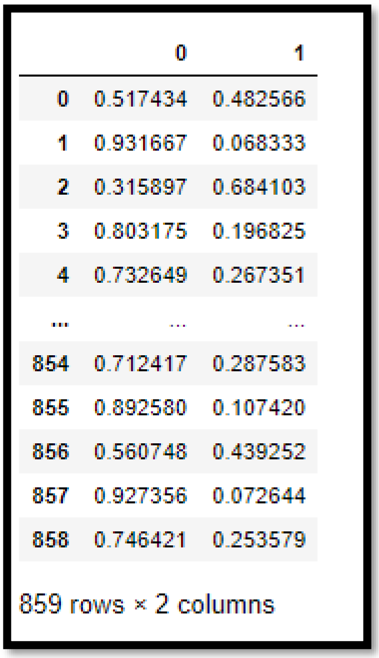
Like the model building that happened without treating the outliers, here we are going to follow the same steps after treating the dataset with the outliers. The treatment of outliers is nothing but changing the values that are more than that upper whisker and less than lower whisker is shifted to the whisker values near to them.

The parameters and the rebuilding of model after pruning(this time with the dataset that was treated with outliers) have shown the varied results in prediction to that of the model that was built without treating outliers which can tell us clearly that outlier treatment has been significant effect to the result.

Also, we can look out for various features like "Feature Importance" that can Help us identify the important variables in the dataset. After rebuilding the model, the Feature Importance came out to be:



```
                   Imp
Agency_Code   0.392188
Sales         0.220123
Product_Name  0.164582
Duration      0.081160
Commision     0.065718
Age           0.060216
Destination   0.010052
Type          0.005960
Channel       0.000000
```



	0	1
0	0.517434	0.482566
1	0.931667	0.068333
2	0.315897	0.684103
3	0.803175	0.196825
4	0.732649	0.267351
...	...	...
854	0.712417	0.287583
855	0.892580	0.107420
856	0.560748	0.439252
857	0.927356	0.072644
858	0.746421	0.253579

859 rows × 2 columns

We can see further understand from the results predicted with the 30% of test data that the probabilities look different from as last time. The effect on how much the result has changed can be viewed from the Accuracy score and the ROC-AUC score which we'll see later in the document.

OOB score for the model   : 0.7667332667332667

OOB error : 1 – OOB score : 0.2332667332667333

MLP CLASSIFIER (ARTIFICIAL NEURAL NETWORK) after treating outliers:

ARTIFICIAL NEURAL NETWORK is the next classification model that we are going to train with the dataset that didn't treat any outliers.

For this model the outlier treatment is a definite necessity as ANN model is usually preferred without any outliers.

Like the past models we can start with the parameters optimization as the model will go for overfitting the training dataset which can hugely affect the final accuracy on the test dataset(and invariably affecting real time data accuracy).

Parameters:

- hidden_layer_sizes
- activation
- solver
- tol
- max_iter

These are the parameters that are usually tweaked to bring the best possible model for any dataset(there are more parameters to consider, but these usually form most of the criterion for better built model).

For tweaking these parameters, a method called "GridSearchCV" is performed that can-do multiple combinations of these parameters simultaneously and can provide us with the best optimum results.

Based on the tweaking of parameters the best possible combination came out to be

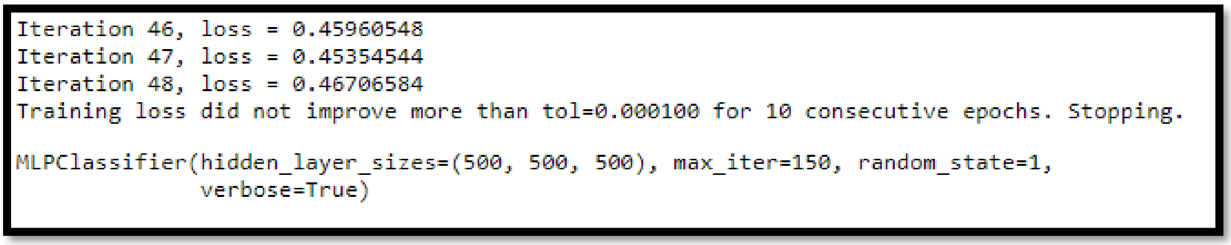
from

- 'hidden_layer_sizes': [(500),(200,200)],
- 'activation': ['logistic', 'relu'],
- 'solver': ['sgd', 'adam'],
- 'tol': [0.1,0.01,0.001],
- 'max_iter' : [150, 300, 500]

To

- hidden_layer_sizes=(500,500,500),
- activation ='relu',
- max_iter = 150,
- solver = 'adam',
- tol = 0.0001

We have also introduced Verbose = True here which can print us the result on the loss function and the number of iterations the model takes so that we can clearly see visually on some of the stuff that happens when the model is building.



```
Iteration 46, loss = 0.45960548
Iteration 47, loss = 0.45354544
Iteration 48, loss = 0.46706584
Training loss did not improve more than tol=0.000100 for 10 consecutive epochs. Stopping.

MLPClassifier(hidden_layer_sizes=(500, 500, 500), max_iter=150, random_state=1,
              verbose=True)
```

The final iterations are seen above, and we can see that the verbose = True has printed the results and the training loss didn't improve after 48th Iteration on building the model.

Further steps on fitting the prediction will be performed after this step.

Question 2.3 : Performance Metrics: Check the performance of Predictions on Train and Test sets using Accuracy, Confusion Matrix, Plot ROC curve and get ROC_AUC score for each model

Performance Metrics:

Usually there are many performance metrics that are generally used in assessing the strength of the model to understand how the model has performed as well as to take an informed decision on whether to go forward with the model in the real time scenario or not.

The industrial standards are generally the following methods:

- Classification Accuracy.
- Confusion Matrix.
- Classification Report.
- Area Under ROC Curve(visualization) and AUC Score

1. Classification Accuracy:

Classification accuracy is the number of correct predictions made as a ratio of all predictions made.

2. Confusion Matrix:

The Confusion Matrix is a table that presents predictions on the x-axis and accuracy outcomes on the y-axis. The cells of the table are the number of predictions made by a machine learning algorithm.

3. Classification Report:

Its a convenience report that is created when working on classification problems to give us a quick idea of the accuracy of a model using a number of measures.

4. Area Under ROC Curve(visualization) and AUC Score:

The ROC Curve measures how accurately the model can distinguish between two things(generally classification information). Larger the curve, the better the model. AUC measures the entire two-dimensional area underneath the ROC curve. This score gives us a good idea of how well the classifier will perform.

CART from dataset that has not been treated with outliers:

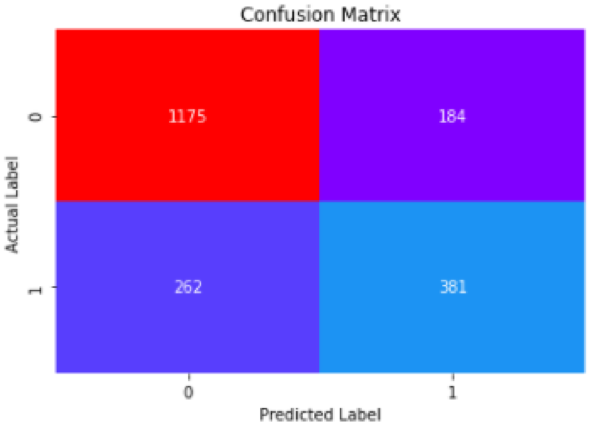
Accuracy:

Training Accuracy : 0.7772227772227772

Testing Accuracy : 0.7788125727590222

Confusion Matrix:

For Training :



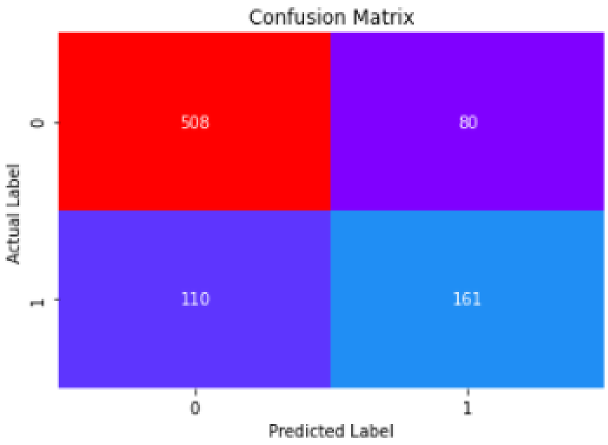
True Negative : 1175

False Positive : 184

False Negative : 262

True Positive : 381

For Testing:



True Negative : 508

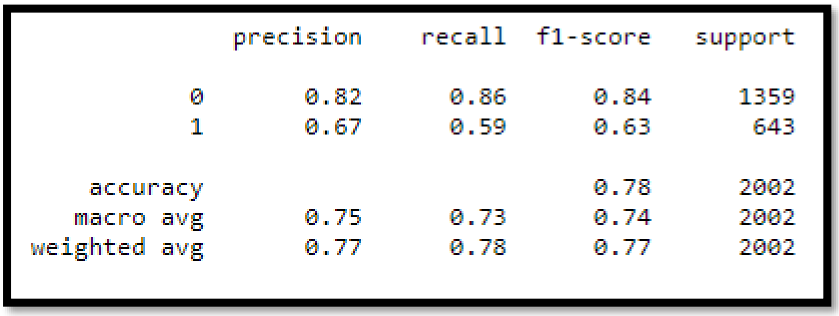
False Positive : 80

False Negative : 110

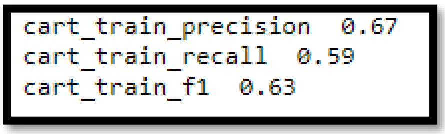
True Positive : 161

Classification Report:

For Training :

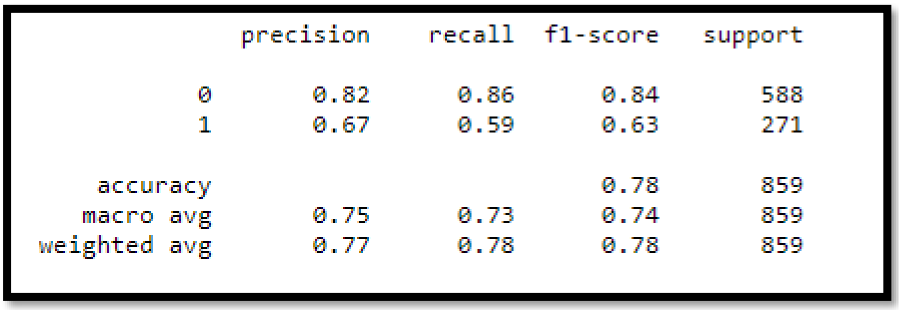


```
              precision    recall  f1-score   support

           0       0.82      0.86      0.84      1359
           1       0.67      0.59      0.63       643

    accuracy                           0.78      2002
   macro avg       0.75      0.73      0.74      2002
weighted avg       0.77      0.78      0.77      2002
```

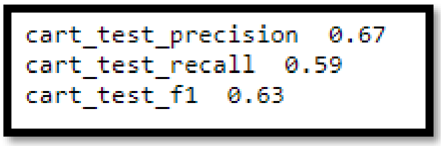


```
cart_train_precision  0.67
cart_train_recall  0.59
cart_train_f1  0.63
```

For Testing :



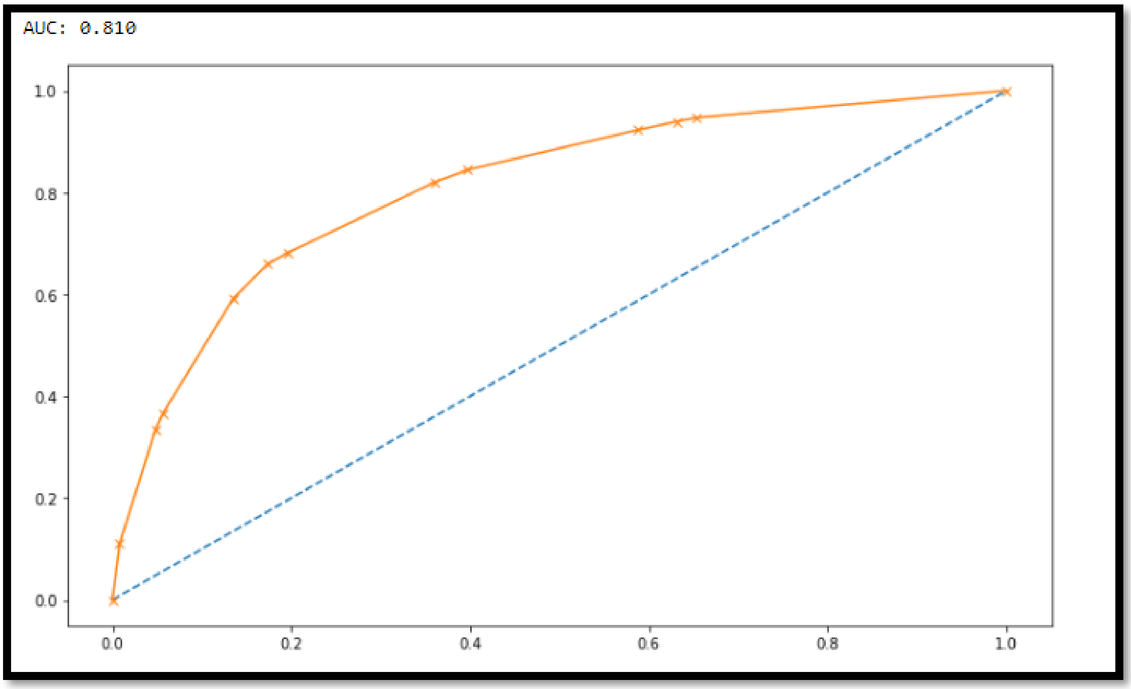
```
              precision    recall  f1-score   support

           0       0.82      0.86      0.84       588
           1       0.67      0.59      0.63       271

    accuracy                           0.78       859
   macro avg       0.75      0.73      0.74       859
weighted avg       0.77      0.78      0.78       859
```



```
cart_test_precision  0.67
cart_test_recall  0.59
cart_test_f1  0.63
```

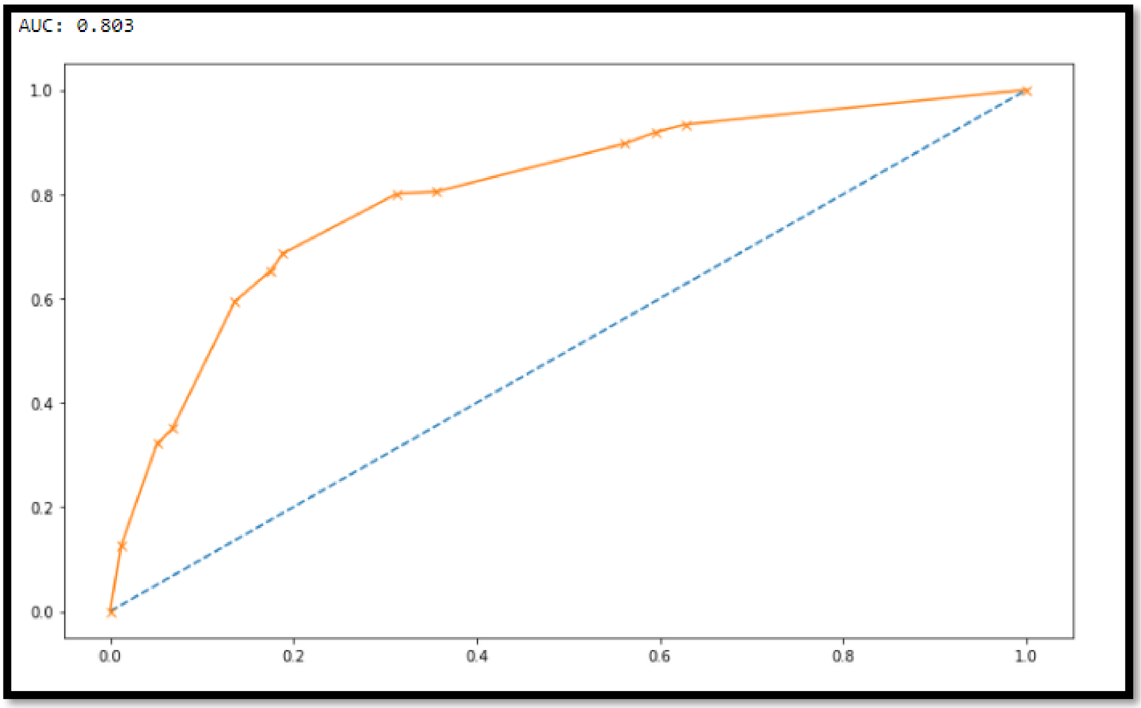
Area Under ROC Curve(visualization) and AUC Score:

For Training :



```
AUC: 0.810
```

For Testing :



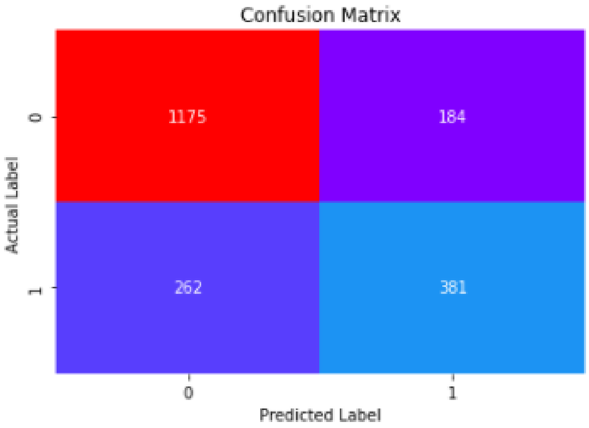
CART from dataset that has been treated with outliers:

Accuracy:

Training Accuracy : 0.7772227772227772

Testing Accuracy : 0.7788125727590222**Confusion Matrix:**

For Training :



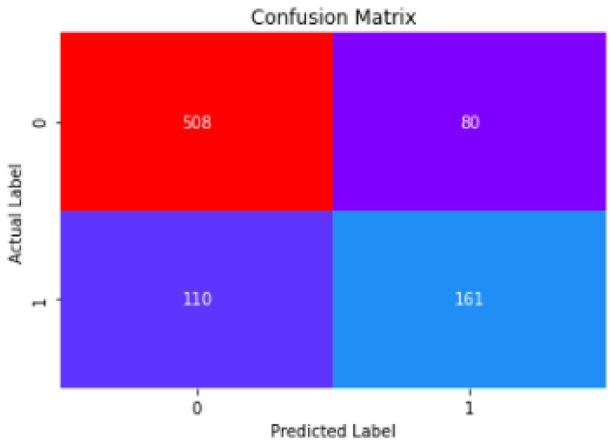
True Negative : 1175

False Positive : 184

False Negative : 262

True Positive : 381

For Testing:



True Negative : 508

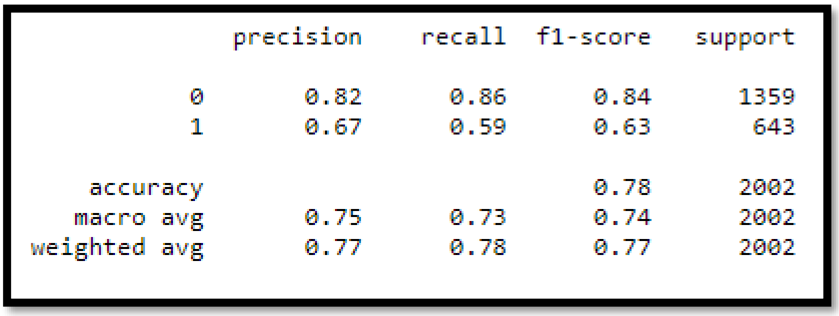
False Positive : 80

False Negative : 110

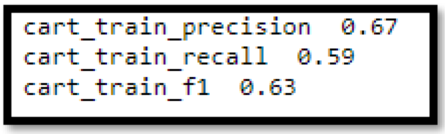
True Positive : 161

Classification Report:

For Training :

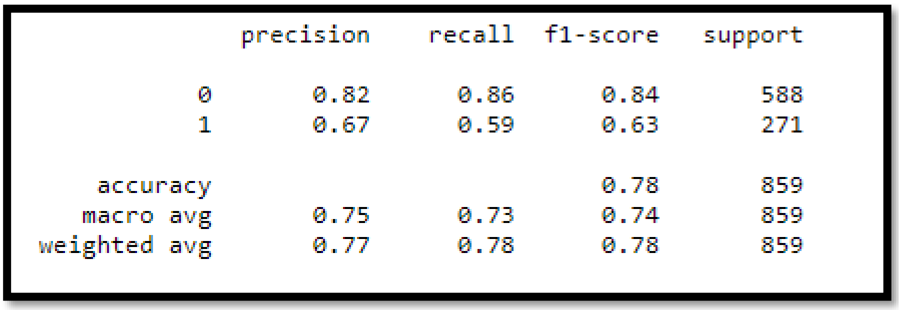


```
              precision    recall  f1-score   support

           0       0.82      0.86      0.84      1359
           1       0.67      0.59      0.63       643

    accuracy                           0.78      2002
   macro avg       0.75      0.73      0.74      2002
weighted avg       0.77      0.78      0.77      2002
```

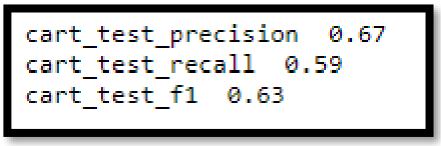


```
cart_train_precision  0.67
cart_train_recall  0.59
cart_train_f1  0.63
```

For Testing :



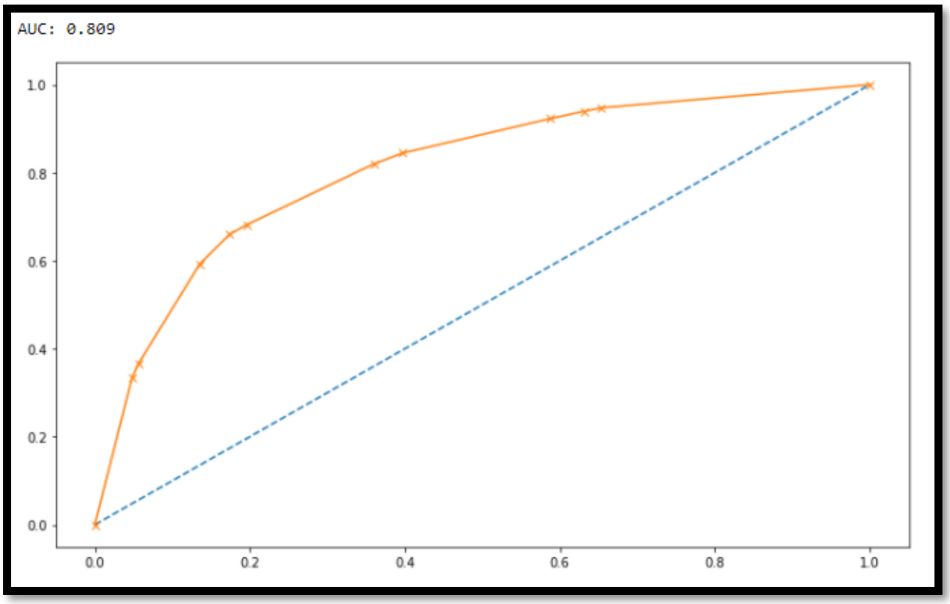
```
              precision    recall  f1-score   support

           0       0.82      0.86      0.84       588
           1       0.67      0.59      0.63       271

    accuracy                           0.78       859
   macro avg       0.75      0.73      0.74       859
weighted avg       0.77      0.78      0.78       859
```



```
cart_test_precision  0.67
cart_test_recall  0.59
cart_test_f1  0.63
```

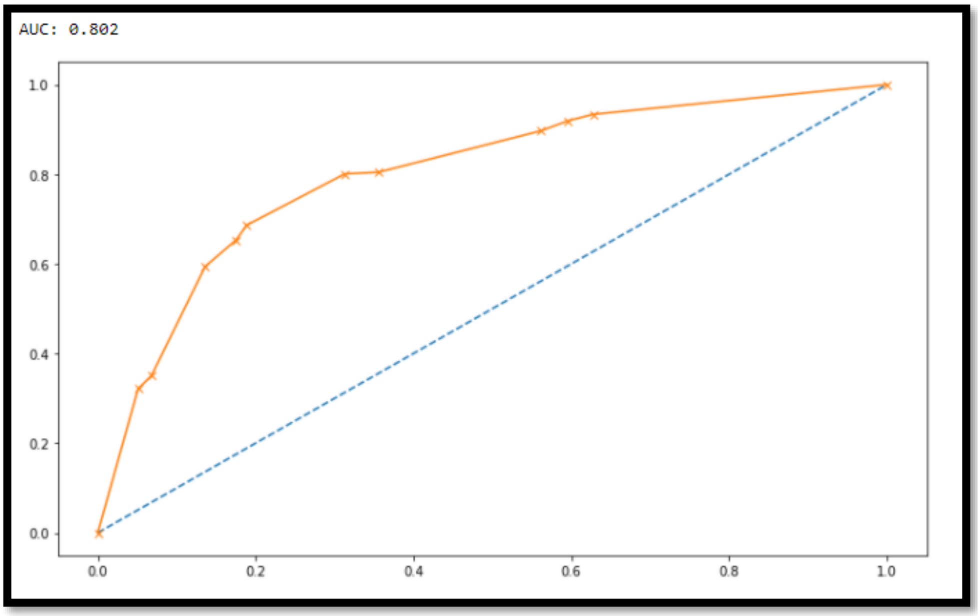
Area Under ROC Curve(visualization) and AUC Score:

For Training :



```
AUC: 0.809
```

For Testing :



RANDOM FOREST CLASSIFIER without treating outliers:

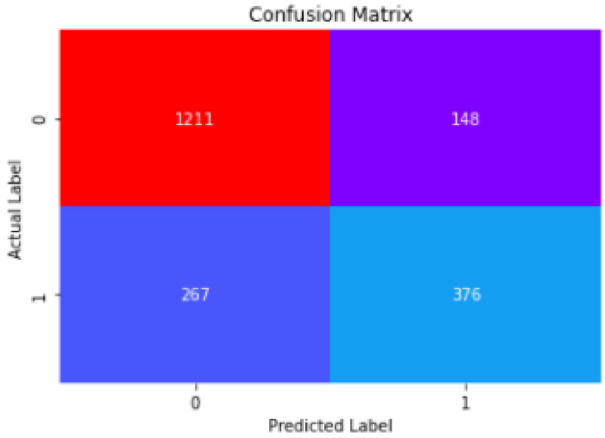
Accuracy:

Training Accuracy : 0.7927072927072927

Testing Accuracy : 0.7823050058207218

Confusion Matrix:

For Training :



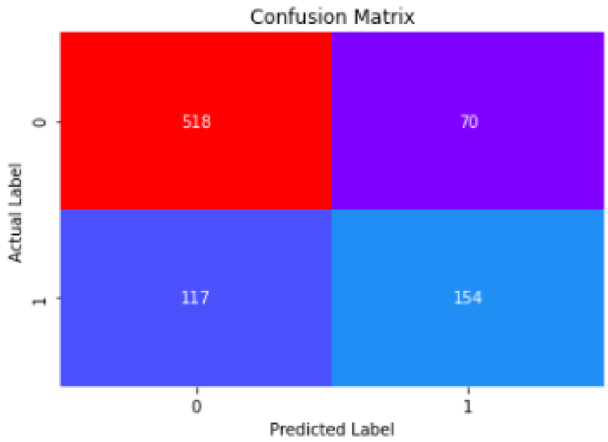
True Negative : 1211

False Positive : 148

False Negative : 267

True Positive : 376

For Testing:



True Negative : 518

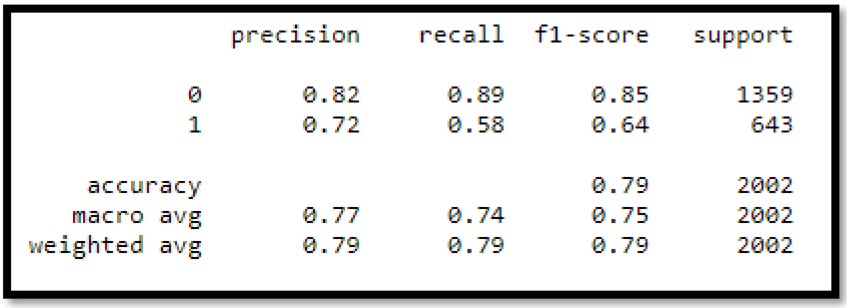
False Positive : 70

False Negative : 117

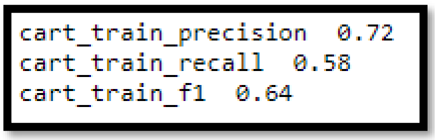
True Positive : 154

Classification Report:

For Training :

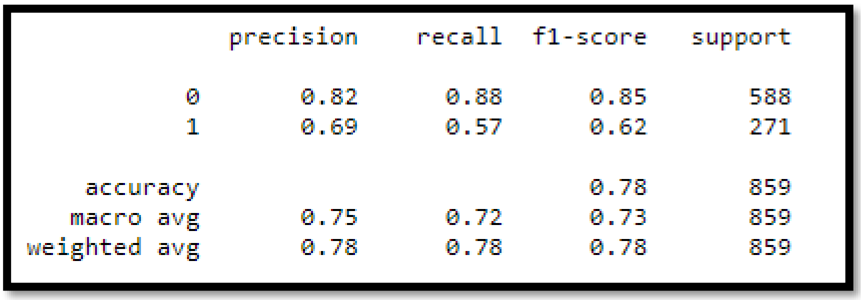


```
              precision    recall  f1-score   support

           0       0.82      0.89      0.85      1359
           1       0.72      0.58      0.64       643

    accuracy                           0.79      2002
   macro avg       0.77      0.74      0.75      2002
weighted avg       0.79      0.79      0.79      2002
```

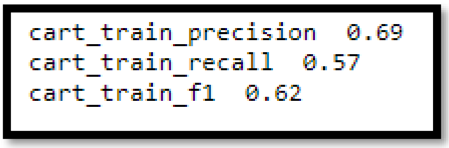


```
cart_train_precision  0.72
cart_train_recall  0.58
cart_train_f1  0.64
```

For Testing :



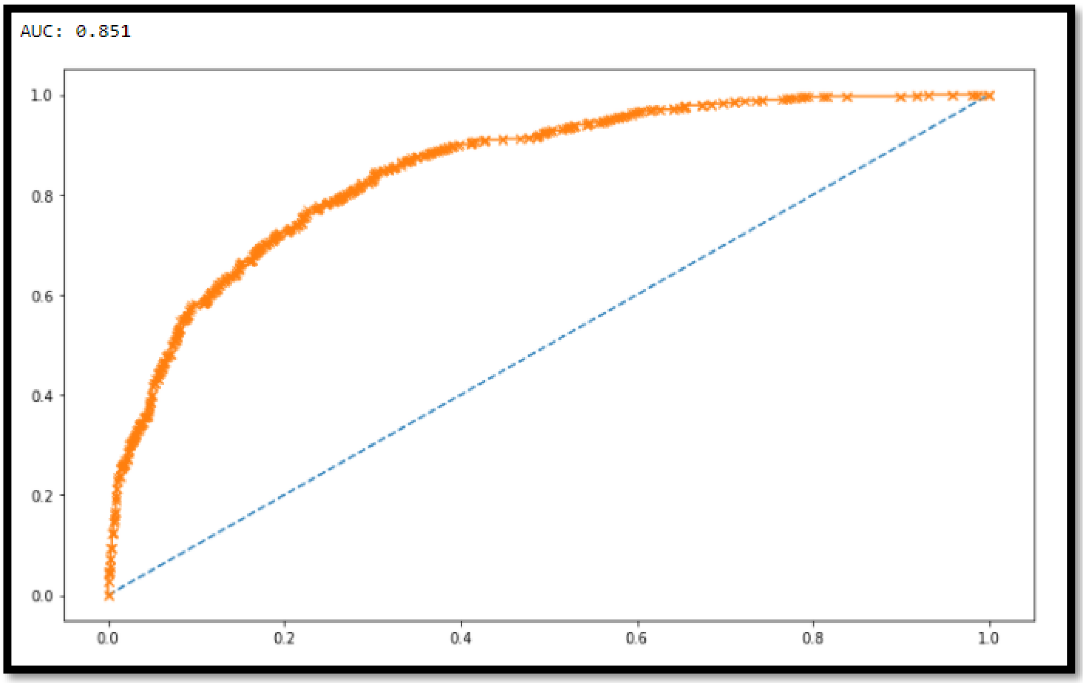
```
              precision    recall  f1-score   support

           0       0.82      0.88      0.85       588
           1       0.69      0.57      0.62       271

    accuracy                           0.78       859
   macro avg       0.75      0.72      0.73       859
weighted avg       0.78      0.78      0.78       859
```



```
cart_train_precision  0.69
cart_train_recall  0.57
cart_train_f1  0.62
```

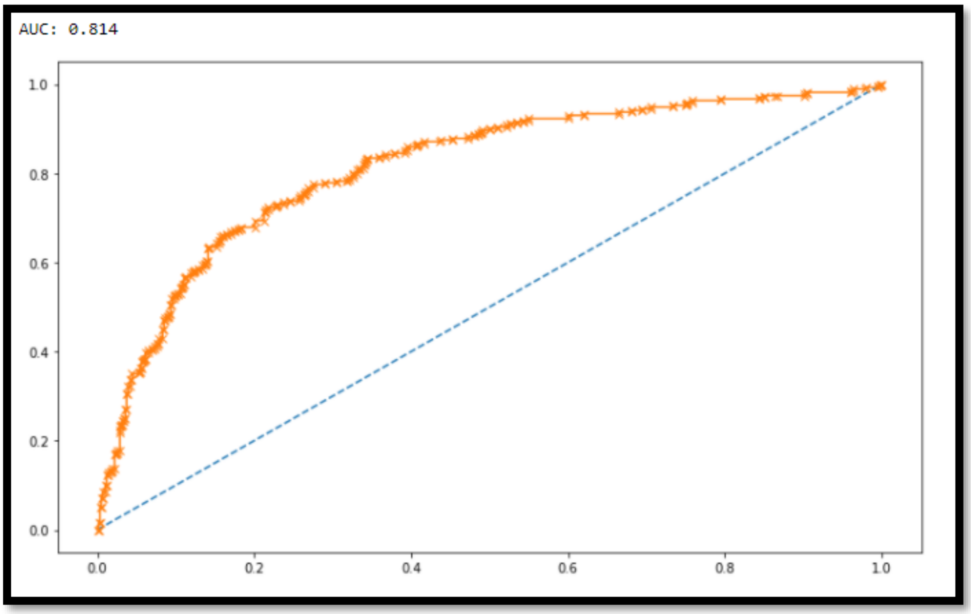
Area Under ROC Curve(visualization) and AUC Score:

For Training :



AUC: 0.851

For Testing :



RANDOM FOREST CLASSIFIER after treating outliers:

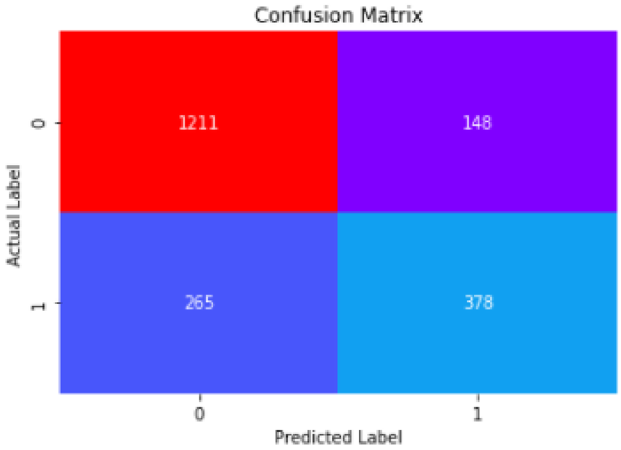
Accuracy:

Training Accuracy : 0.7937062937062938

Testing Accuracy : 0.7846332945285215

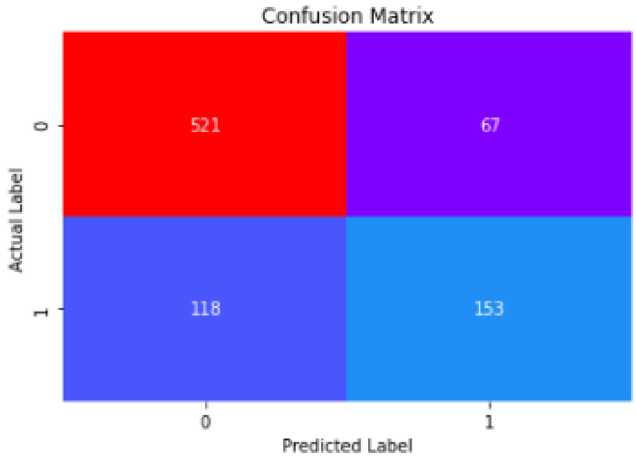
Confusion Matrix:

For Training :



True Negative	: 1211	False Positive	: 148
False Negative	: 265	True Positive	: 378

For Testing :



True Negative : 521

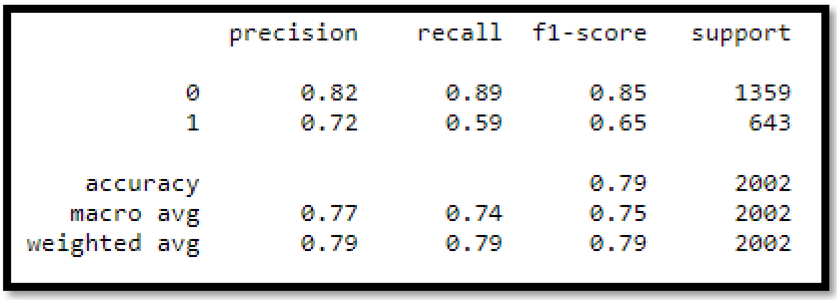
False Positive : 67

False Negative : 118

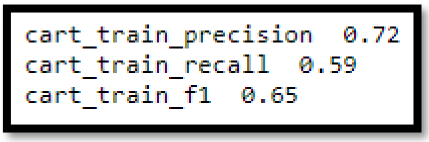
True Positive : 153

Classification Report:

For Training :

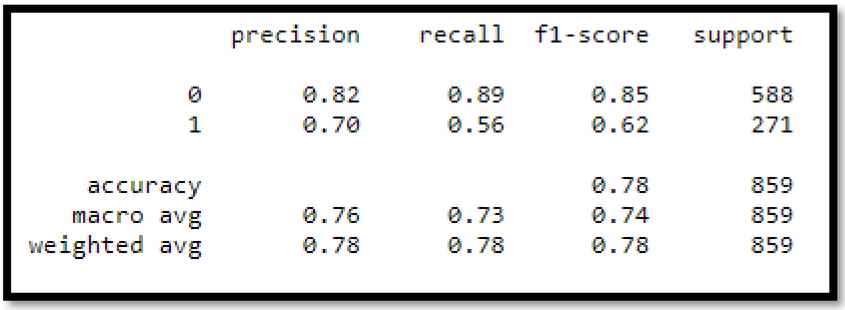


```
              precision    recall  f1-score   support

           0       0.82      0.89      0.85      1359
           1       0.72      0.59      0.65       643

    accuracy                           0.79      2002
   macro avg       0.77      0.74      0.75      2002
weighted avg       0.79      0.79      0.79      2002
```

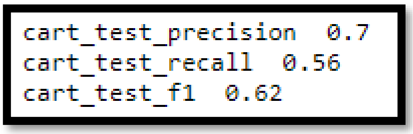


```
cart_train_precision  0.72
cart_train_recall  0.59
cart_train_f1  0.65
```

For Testing :



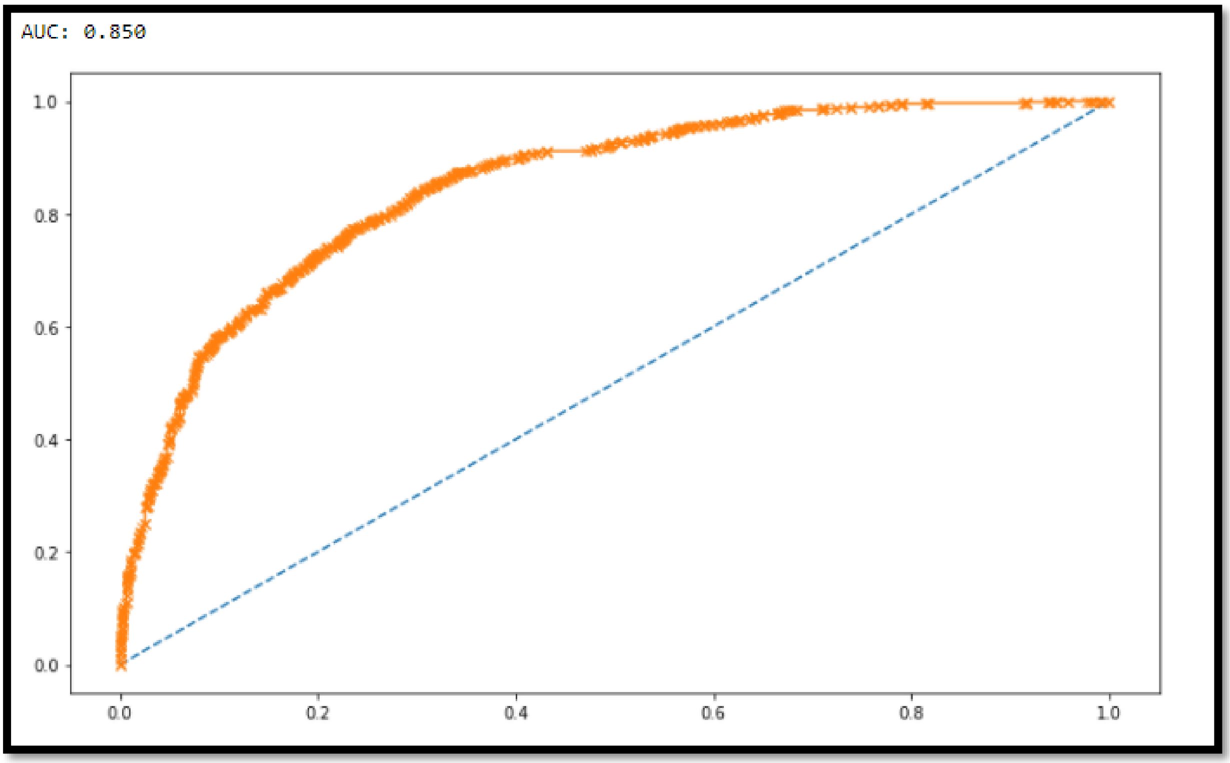
```
              precision    recall  f1-score   support

           0       0.82      0.89      0.85       588
           1       0.70      0.56      0.62       271

    accuracy                           0.78       859
   macro avg       0.76      0.73      0.74       859
weighted avg       0.78      0.78      0.78       859
```



```
cart_test_precision  0.7
cart_test_recall  0.56
cart_test_f1  0.62
```

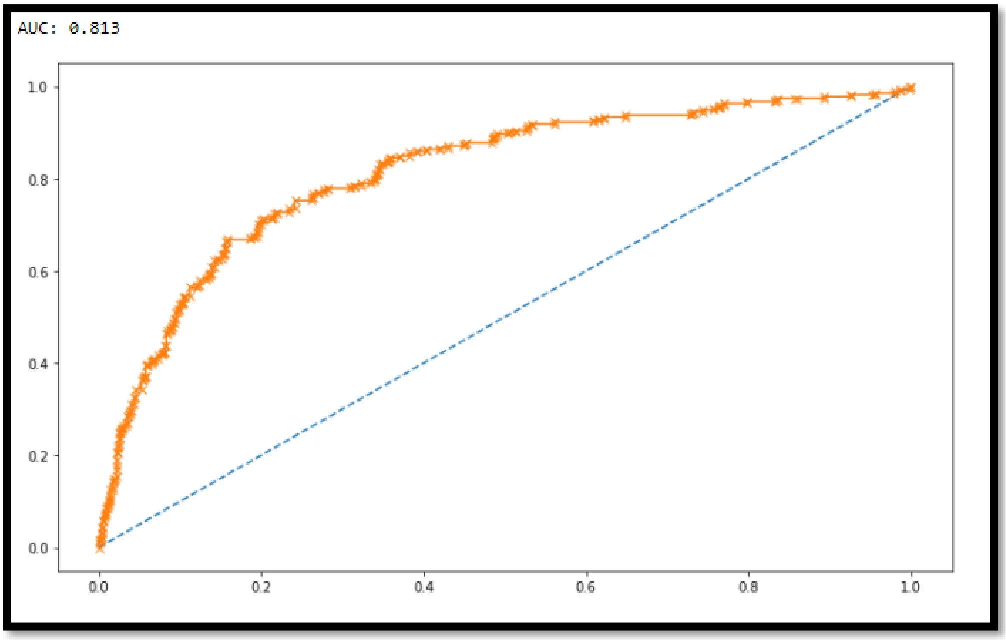
Area Under ROC Curve(visualization) and AUC Score:

For Training :



```
AUC: 0.850
```

For Testing :



MLP CLASSIFIER (ARTIFICIAL NEURAL NETWORK) after treating outliers:

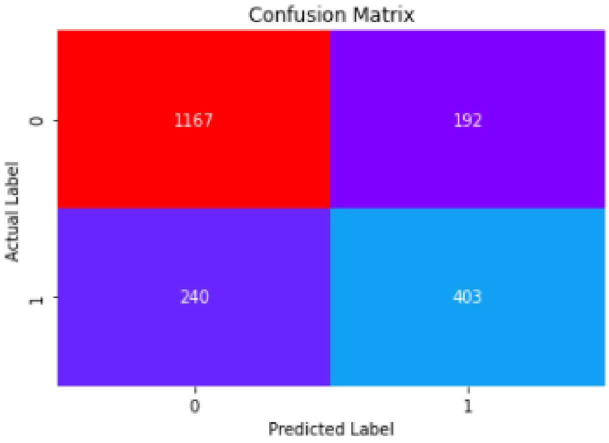
Accuracy:

Training Accuracy : 0.7842157842157842

Testing Accuracy : 0.7811408614668219

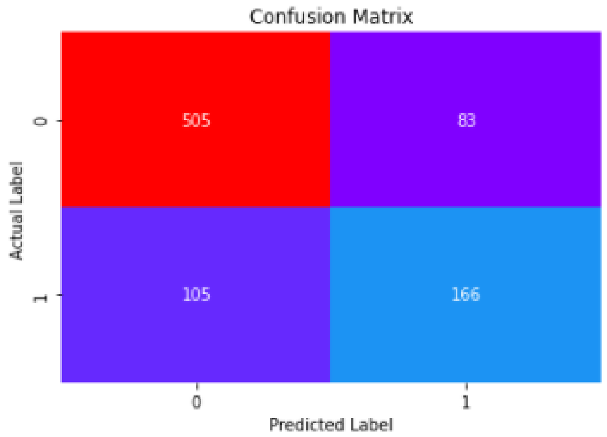
Confusion Matrix:

For Training :



True Negative : 1167	False Positive : 192
False Negative : 240	True Positive : 403

For Testing:



True Negative : 505

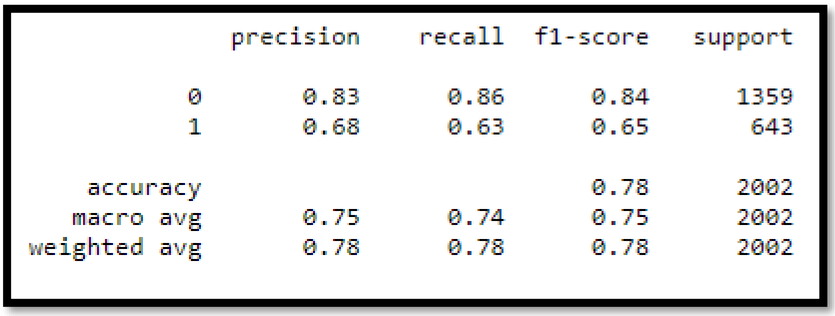
False Positive : 83

False Negative : 105

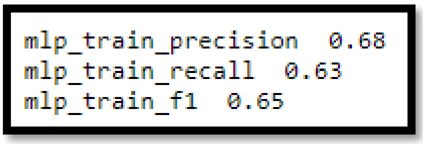
True Positive : 166

Classification Report:

For Training :

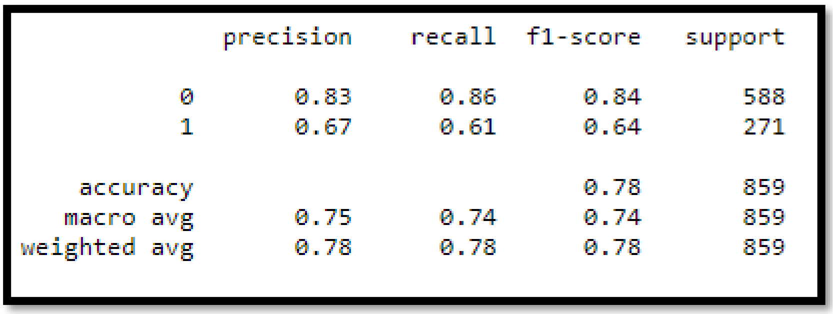


```
              precision    recall  f1-score   support

           0       0.83      0.86      0.84      1359
           1       0.68      0.63      0.65       643

    accuracy                           0.78      2002
   macro avg       0.75      0.74      0.75      2002
weighted avg       0.78      0.78      0.78      2002
```

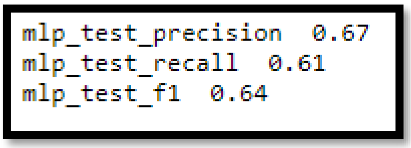


```
mlp_train_precision  0.68
mlp_train_recall  0.63
mlp_train_f1  0.65
```

For Testing :



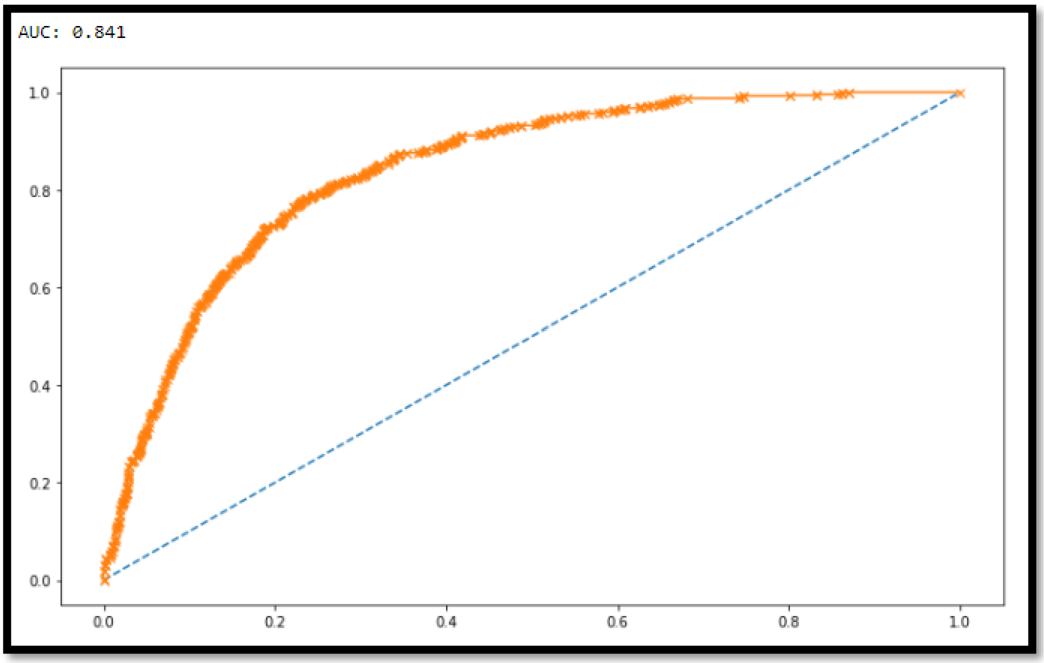
```
              precision    recall  f1-score   support

           0       0.83      0.86      0.84       588
           1       0.67      0.61      0.64       271

    accuracy                           0.78       859
   macro avg       0.75      0.74      0.74       859
weighted avg       0.78      0.78      0.78       859
```



```
mlp_test_precision  0.67
mlp_test_recall  0.61
mlp_test_f1  0.64
```

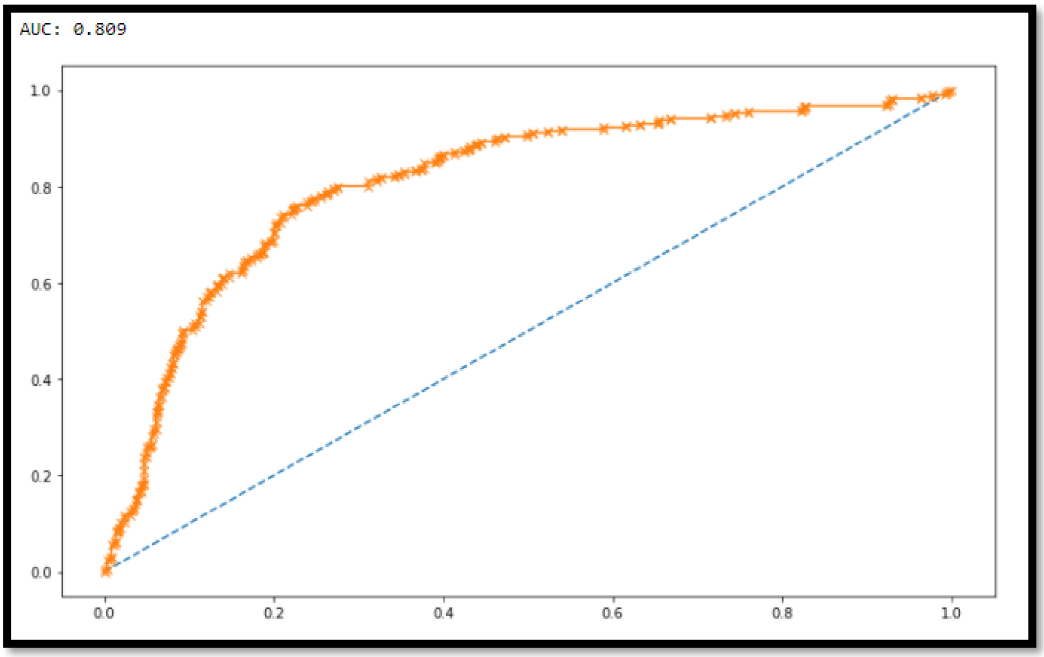
Area Under ROC Curve(visualization) and AUC Score:

For Training :



```
AUC: 0.841
```

For Testing :



```
AUC: 0.809
```

Question 2.4 : Final Model: Compare all the model and write an inference which model is best/optimized.

Now that the three models have been built and their performance metrics have been figured out, the final step is to compare these well-built models to figure out which one to use in the production scenario. That's where model evaluation takes place.

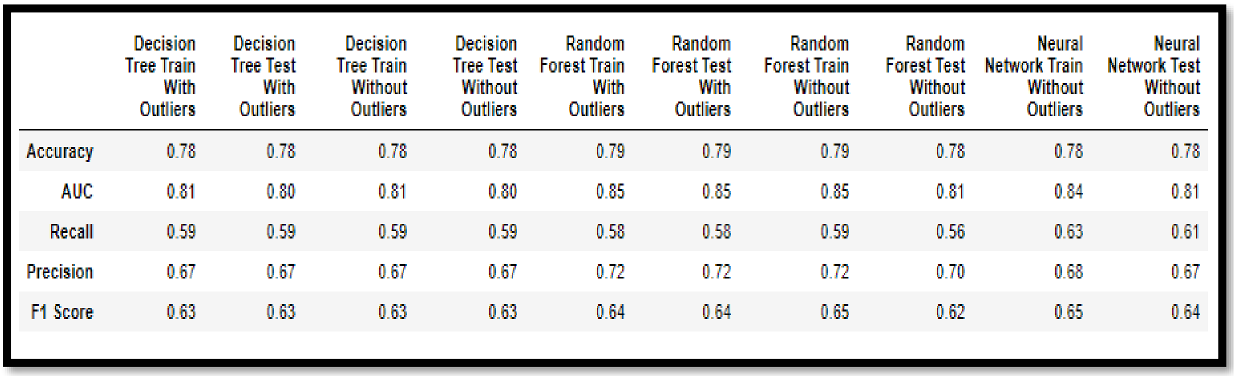
Before looking at the model's performance side by side its first very important to see if you have done everything possible to improve the model's accuracy. Therefore, it's wise to have a look into it before proceeding further.

There are two ways by which we can make the model evaluation easier to make an informed decision.

- **Numeric foundations :** Making a Pivot Table Data Frame with all the models that were built with the performance metrics values to see by first-hand how each model has performed so that we can conclude on which of the models to use in production.

- **Visualization** : Making a clear graph on the performance metrics which contains plots(both with models treated with outliers and without treating outliers so that we can understand the effect of outliers visually).

Numeric foundations :

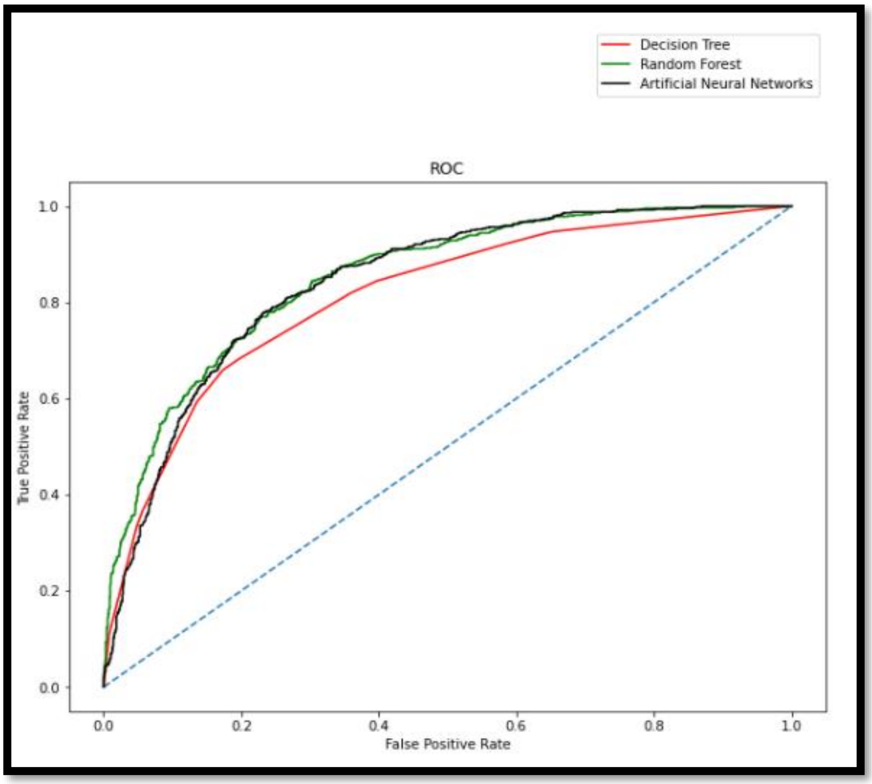


	Decision Tree Train With Outliers	Decision Tree Test With Outliers	Decision Tree Train Without Outliers	Decision Tree Test Without Outliers	Random Forest Train With Outliers	Random Forest Test With Outliers	Random Forest Train Without Outliers	Random Forest Test Without Outliers	Neural Network Train Without Outliers	Neural Network Test Without Outliers
Accuracy	0.78	0.78	0.78	0.78	0.79	0.79	0.79	0.78	0.78	0.78
AUC	0.81	0.80	0.81	0.80	0.85	0.85	0.85	0.81	0.84	0.81
Recall	0.59	0.59	0.59	0.59	0.58	0.58	0.59	0.56	0.63	0.61
Precision	0.67	0.67	0.67	0.67	0.72	0.72	0.72	0.70	0.68	0.67
F1 Score	0.63	0.63	0.63	0.63	0.64	0.64	0.65	0.62	0.65	0.64

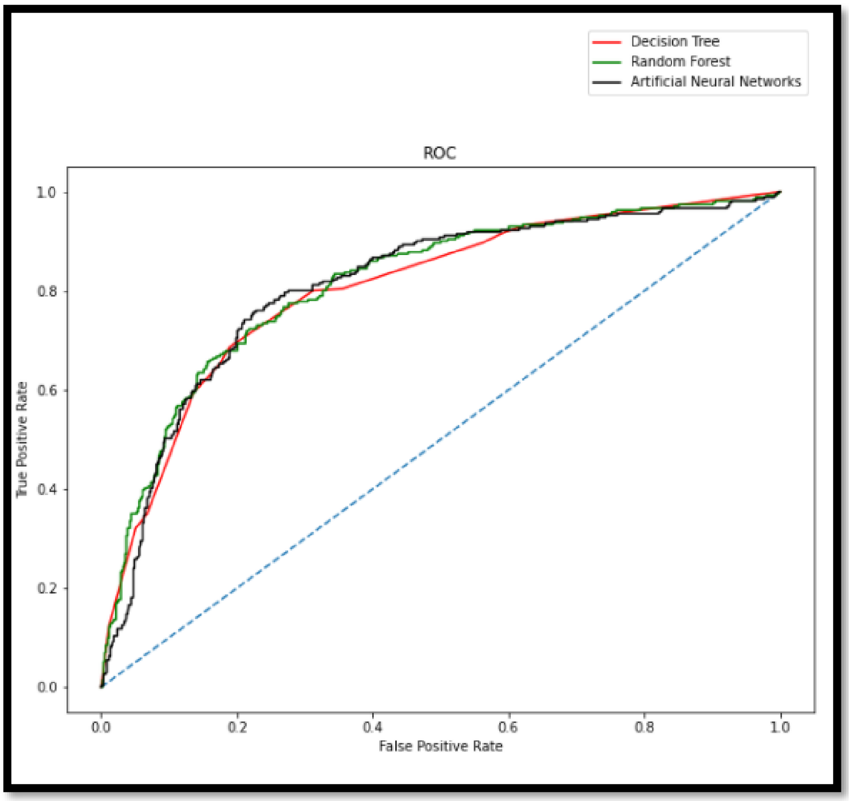
From the above Data Frame we can see the performance metrics like "Accuracy" , "AUC" , "Recall" , "Precision" and "F1 Score" values for all the models(CART , Random Forest and Artificial Neural Networks) with and without treating outliers (Except for ANN where treating outliers and scaling was a necessity).

Visualization :

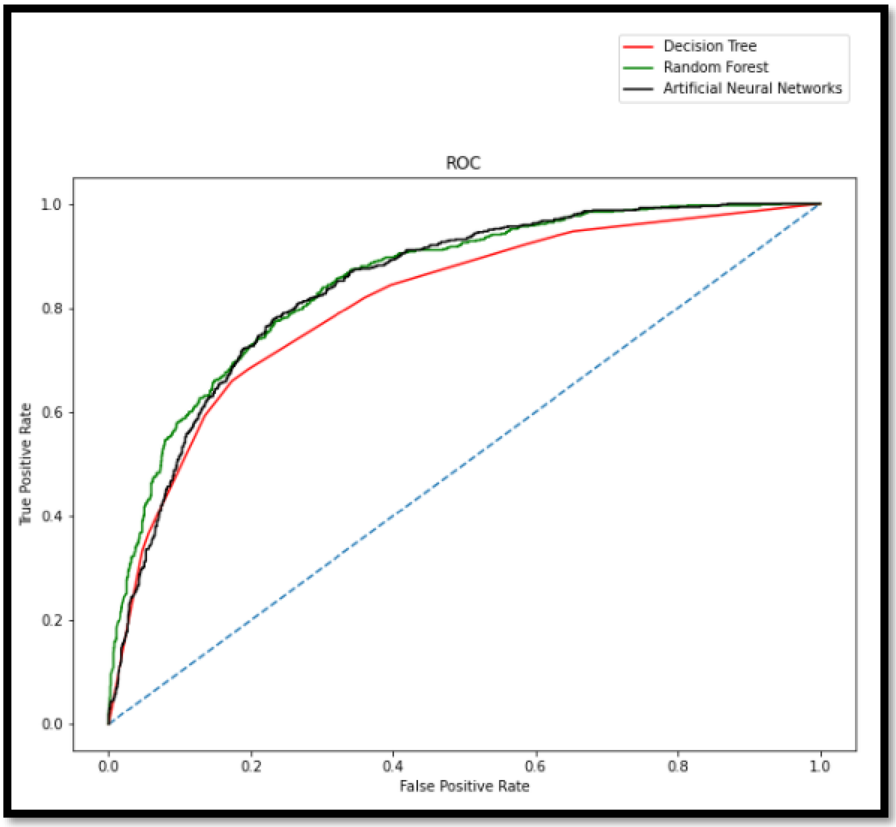
Training – With outliers :



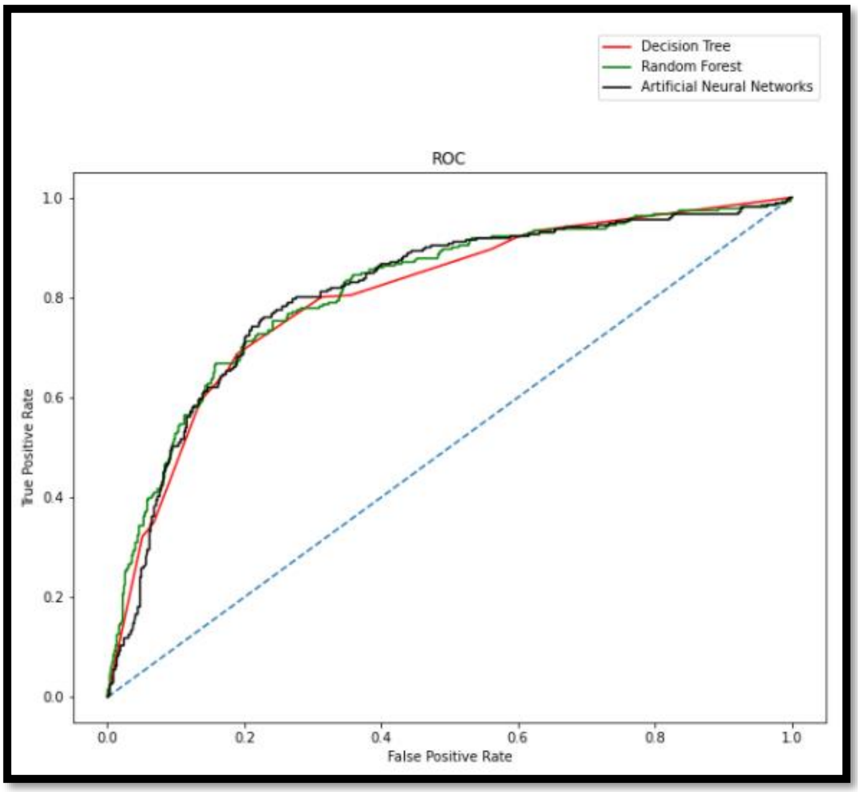
Testing – With outliers :



Training without Outliers :



Testing without outliers :



From the above plots we can see the performance metrics of “ROC-curve” marked for all the models(CART , Random Forest and Artificial Neural Networks) with and without treating outliers (Except for ANN where treating outliers and scaling was a necessity).

Final Verdict:

Final Model selection is a relative concept as Model is not just accuracy score and metrics. The final model selection process is a combined decision of machine learning developers and Business Analysts and Project stakeholders(We need to meet the requirements and constraints of project stakeholders). Therefore, a recommendation is best suited for this case.

Based on the above two methods to showcase their strengths and weaknesses, the following recommendations can be made :

- If **Accuracy** is the best fit for all the stakeholders that are involved, we can go ahead with the **RANDOM FOREST classification method that hasn’t treated any outliers** for this use case. The reason being that it holds around 79% accuracy in both the train and test model’s, and it has even outperformed on a small case with the same model that has been treated with the outliers.

- If **Sensitivity/Recall** is the best fit for all the stakeholders that are involved, we can go ahead with the **Artificial Neural Network Classification method**. The reason being that it holds around 61% test performance which is the highest score considering the other two models. Other than Sensitivity/recall score the Neural Network methods was behind from Random Forest Method in every metrics.

- If **Understandability/Openness**(with clear clarity on the criteria of classification) is the requirement from the project stakeholders, then the high recommendation would be the **CART classification method with/without treating outliers** as it holds 78% accuracy and the other metrics is no short of other models as well (If you take a closer look, all three models they have performed nearly the same in all the metrics with max difference of 5%).

Question 2.5 Inference: Basis on these predictions, what are the business insights and recommendations :

After taking the look on all the model's performance, my recommendation would be to first go with **RANDOM FOREST classification method that hasn't treated any outliers** as it has outclassed all other models when it comes to performance(except Sensitivity/Recall with 5% difference to ANN(test)). It may be a Blackbox technique, but it has shown advantage when it comes to performance and knowing the field that the data has been provided, a maximum importance will be given to the ACCURACY of the model. The Ultimate aim and the purpose of creating these models in to predict the proper high claim frequency and to provide insights and recommendations, and therefore I can go ahead and pitch in ANN model here as well(as a very strong second option, just down by miniscule margins) as looking at the metrics other than just accuracy, the ANN model has beautifully handled a lot of test data(better than CART and Random Forest methods).

Considering these in mind, for further exploring the corners of the model's performance, two things are to be done:

- Class imbalance correction or Resampling the data.
- Increasing the sample by a significant margin.

Class imbalance correction or Resampling the data can be a major factor in influencing the accuracy and performance of the model as less count of values on either side(0 or 1) can affect the prediction of it. In real time we can't expect to have perfect 50-50 class balance, but still we should try to have a 60-40 balance at the max.

Increasing the sample by a significant amount can also be a contributing factor to a model's performance in production scenario. A lot of data can make for a lot of real case scenario's and subsequently increase the model performance by a huge margin.

After performing these two things(if at all possible), to further improve the performance we can still investigate on tuning the parameters that is used in building all the three model. This is specific to the case of artificial neural networks, as there can be a lot of tuning combinations that is possible to improve the models(not a lot of them improve, but still worth experimenting) performance.

Problem 2 Summary:

- **2.1)** Data ingestion was performed. All the basic EDA along with the univariate and the Bivariate analysis were performed and analysed including the descriptive statistics and null value condition check.
- **2.2)** Splitting the data into test and train was performed. Classification models like CART, Random Forest and Artificial Neural Network were built.
- **2.3)** Performance of Predictions on Train and Test sets using Accuracy, Confusion Matrix, Plot ROC curve and get ROC_AUC score for each model was analysed.
- **2.4)** All the models were compared and inference which model is best/optimized is written.
- **2.5)** On the basis on these predictions, the business insights and recommendations were provided appropriately.

Conclusion:

Looking into the data of “insurance_part2_data-1.csv”, we saw some interesting insights from EDA methods. We were able to split the data properly and build three classification models like the CART model, Random Forest model and the Artificial Neural Network model. There were a lot of thoughts poured into the optimization and the analysis of the results via various performance metrics, but we were able to properly utilize the given data to provide three well built model on insurance data that provides proper predictions on the insurance claims. Also, as an added advantage we were also able to provide some insights on improving this data to further improve the performance as well.