The report covers various types of data mining models such as CLUSTERING, CART, RANDOM FOREST CLASSIFICATION & ANN.

BUSSINESS REPORT

DATA MINING

PARAKANDLA. MANISHA (PGP-DSBA ONLINE) GREAT LEARNING

Contents

LIST OF FIGURES:

LIST OF TABLES:

Problem 1: Clustering (Bank market segmentation)

EXECUTIVE SUMMARY:

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.

Dataset for Problem 1: bank_marketing_part1_Data.csv

Data Dictionary for Market Segmentation:

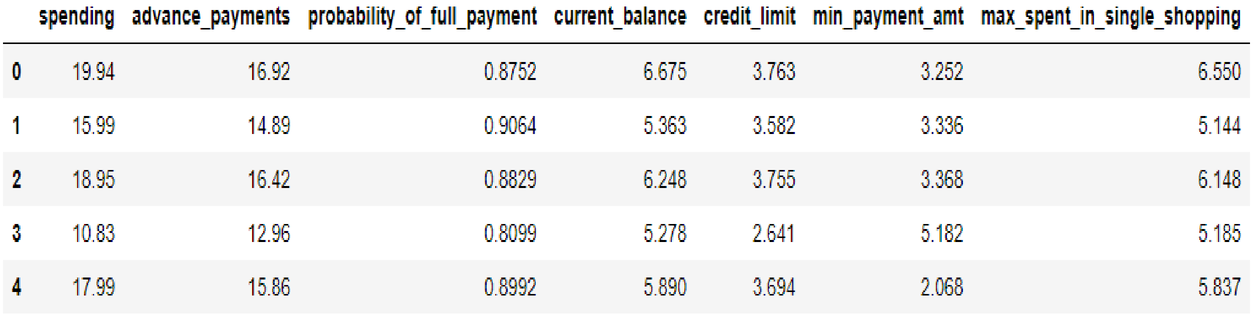
1. spending: Amount spent by the customer per month (in 1000s)
2. advance_payments: Amount paid by the customer in advance by cash (in 100s)
3. probability_of_full_payment: Probability of payment done in full by the customer to the bank
4. current_balance: Balance amount left in the account to make purchases (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)

INTRODUCTION:

In this problem, we perform clustering method for customer segmentation based on the bank marketing dataset. The sample dataset consists of 210 records and 7 Features. For performing clustering technique, we need to import all the required libraries and initial descriptive data analysis to be done.

1.1 Read the data, do the necessary initial steps, and exploratory data analysis (Univariate, Bi-variate, and multivariate analysis).

Summary of the Dataset:



	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

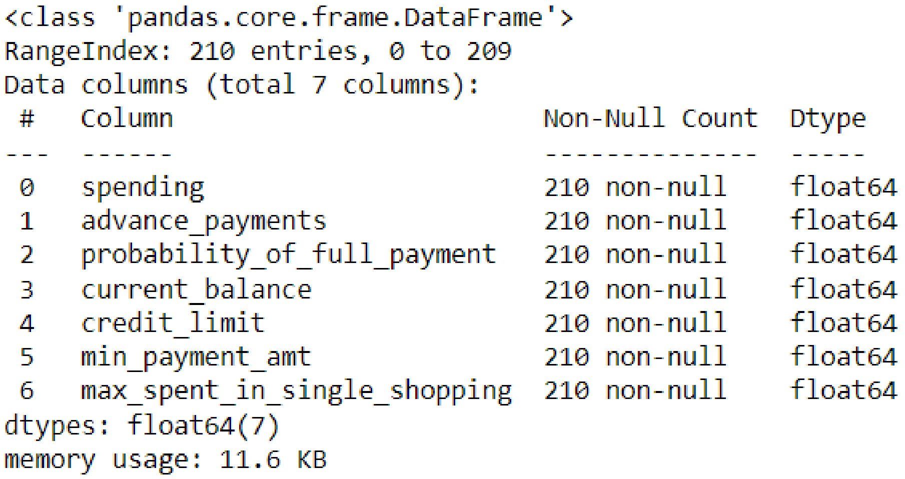
Table 1.1 Summary of bank market segmentation

→The dimensions of the dataset are (210,7).

→the dataset has 210 entries with 7 variables(features).

→The data has no missing values and all the variables are numeric type.

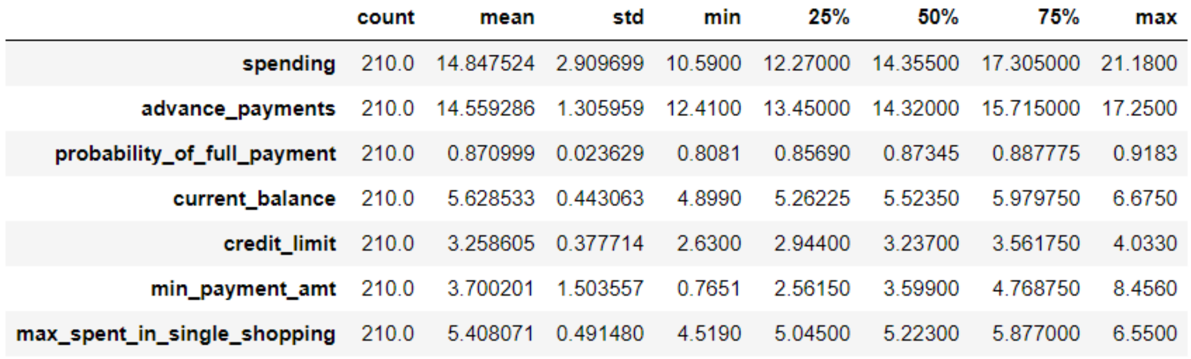
→Info () function gives the brief understanding of the variables in the data.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 210 entries, 0 to 209
Data columns (total 7 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   spending                      210 non-null    float64
 1   advance_payments              210 non-null    float64
 2   probability_of_full_payment   210 non-null    float64
 3   current_balance               210 non-null    float64
 4   credit_limit                  210 non-null    float64
 5   min_payment_amt               210 non-null    float64
 6   max_spent_in_single_shopping  210 non-null    float64
dtypes: float64(7)
memory usage: 11.6 KB
```

→There are no garbage values present in the columns.

→The describe () method is used for **calculating some statistical data** like percentile, mean and std of the numerical values in the dataset. It analyses both numeric and object series and also the column sets of mixed data types.



	count	mean	std	min	25%	50%	75%	max
spending	210.0	14.847524	2.909699	10.5900	12.27000	14.35500	17.305000	21.1800
advance_payments	210.0	14.559286	1.305959	12.4100	13.45000	14.32000	15.715000	17.2500
probability_of_full_payment	210.0	0.870999	0.023629	0.8081	0.85690	0.87345	0.887775	0.9183
current_balance	210.0	5.628533	0.443063	4.8990	5.26225	5.52350	5.979750	6.6750
credit_limit	210.0	3.258605	0.377714	2.6300	2.94400	3.23700	3.561750	4.0330
min_payment_amt	210.0	3.700201	1.503557	0.7651	2.56150	3.59900	4.768750	8.4560
max_spent_in_single_shopping	210.0	5.408071	0.491480	4.5190	5.04500	5.22300	5.877000	6.5500

Table 1.2. Descriptive summary of bank market data

UNIVARIATE ANALYSIS:

Univariate analysis is **the simplest form of analysing data**. It takes data, summarizes that data and finds patterns in the data. Like other forms of statistics, it can be inferential or descriptive. It helps us to understand the distribution of data in the dataset. With univariate analysis we can find patterns and summarize the data.

Spending Variable:

Range of the spending variable is 10.59

Maximum spending: 21.18

Minimum spending: 10.59

Mean: 14.847523809523818

Median: 14.355

Standard deviation: 2.909699430687361

Null values: False

spending - First Quartile(Q1) is: 12.27

spending - Third Quartile(Q3) is: 17.305

Interquartile range (IQR) for spending is 5.035

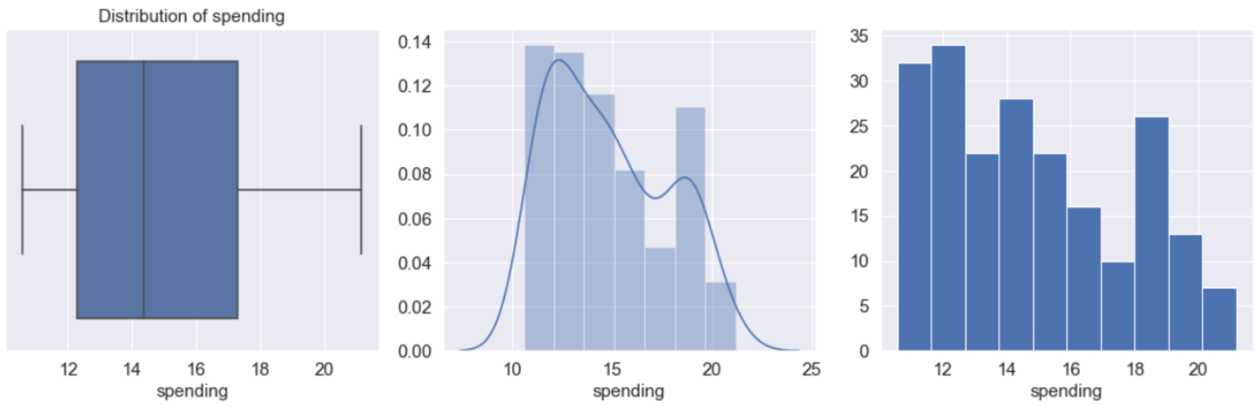


Fig.1 plots for spending variable

→The box plot of the spending variable shows no outliers.

→Spending is positively skewed.

→We could also understand there could be chance of multi modes.

→The dist plot shows the distribution of data from 10 to 22.

advance_payments:

Range: 4.84
Maximum advance_payments: 17.25
Minimum advance_payments: 12.41
Mean: 14.559285714285727
Median: 14.32
Standard deviation: 1.305958726564022

Null values: Falseadvance_payments - First Quartile(Q1) is: 13.45
advance_payments - Third Quartile(Q3) is: 15.715
Interquartile range (IQR) for advance_payments is 2.2650000000000006

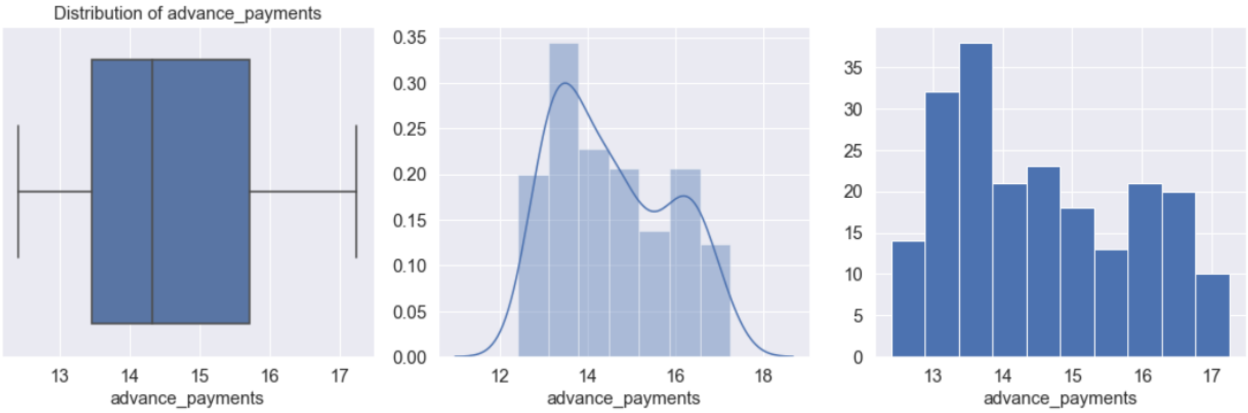


Fig2. Plots for advance_payments variable

→ The boxplot for advance_payments has no outliers.

→advance_payments is positively skewed.

→ There could be chance of multi modes in the dataset.

→The dist plot shows the distribution of data from 12 to 17

Probability_of_full_payment:

Range: 0.11019999999999996
Maximum probability_of_full_payment: 0.9183
Minimum probability_of_full_payment 0.8081
Mean value: 0.8709985714285714
Median value: 0.8734500000000001
Standard deviation: 0.0236294165838465
Null values: False
probability_of_full_payment - First Quartile (Q1) is: 0.8569
probability_of_full_payment - Third Quartile (Q3) is: 0.887775
Interquartile range (IQR) for probability_of_full_payment is 0.030874999999999986

→ The boxplot for probability_of_full_payment has outliers.

→ probability_of_full_payment is positively skewed.

→ The probability values is good above 80%

→The dist plot shows the distribution of data from 0.80 to 0.92

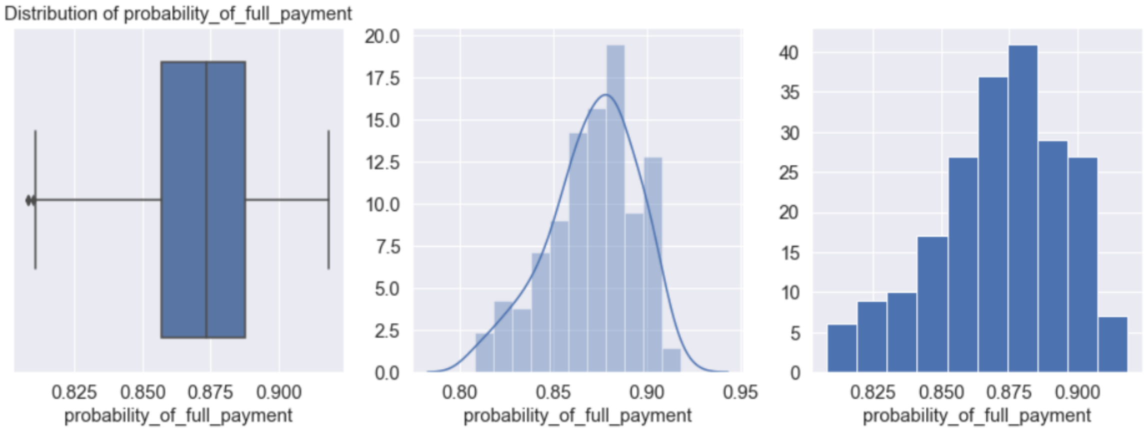


Fig3.Plots for probability_of_full_payment

Current_balance:

Range: 1.7759999999999998
Maximum current_balance: 6.675
Minimum current_balance: 4.899
Mean: 5.628533333333335
Median: 5.5235
Standard deviation: 0.44306347772644944
Null values: False
current_balance - First Quartile (Q1) is: 5.26225
current_balance - Third Quartile (Q3) is: 5.97975
Interquartile range (IQR) for current_balance is 0.7175000000000002

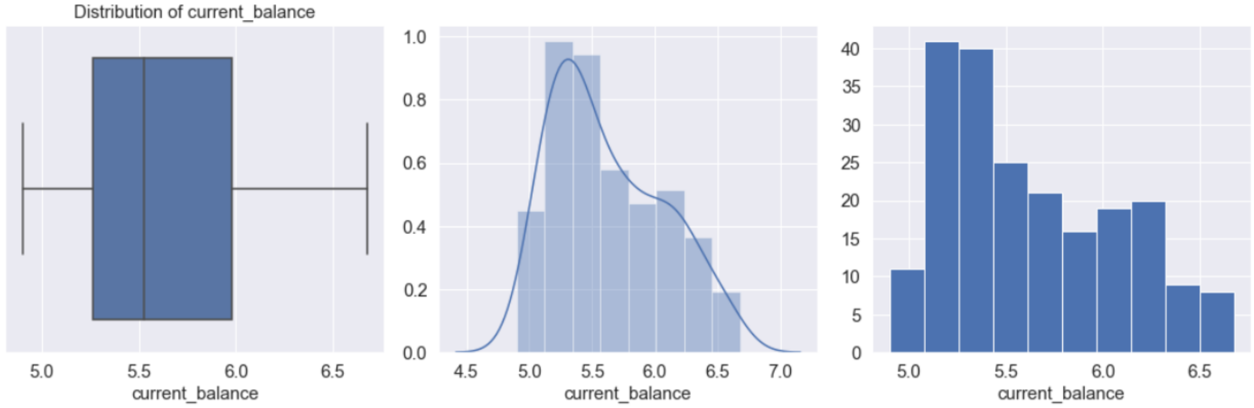


Fig4.Plots for current_balance

→ The boxplot for current_balance has no outliers.

→Current_balance is positively skewed.

→The dist plot shows the distribution of data from 5.0 to 6.5.

credit_limit:

Range: 1.4030000000000005
Maximum credit_limit: 4.033
Minimum credit_limit: 2.63
Mean: 3.258604761904763
Median: 3.237
Standard deviation: 0.37771444490658734
Null values: False
credit_limit - First Quartile (Q1) is: 2.944
credit_limit - Third Quartile (Q3) is: 3.56175
Interquartile range (IQR) of credit_limit is 0.61775

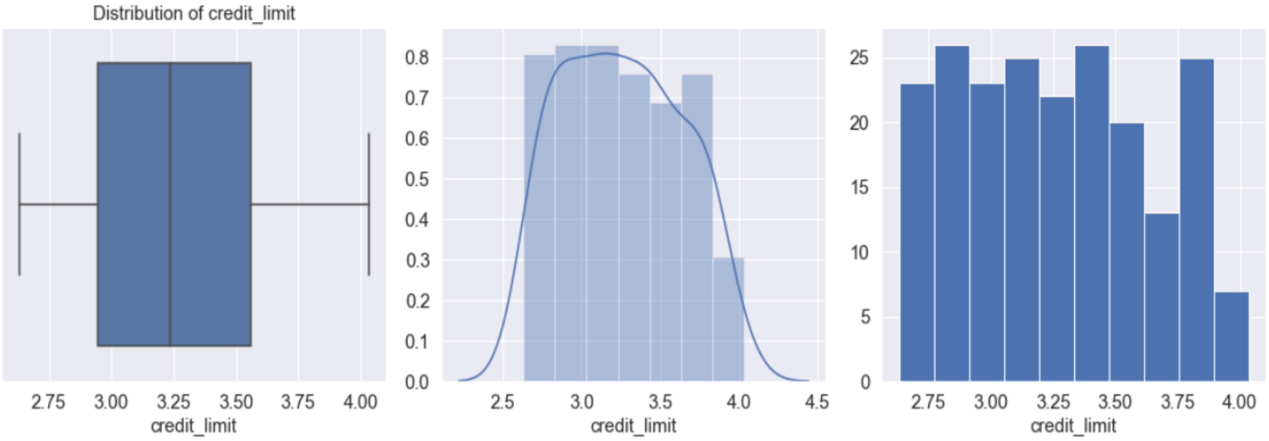


Fig5. Plots for credit_limit

→The box plot of the credit_limit variable has no outliers.

→Credit_limit is positively skewed.

→The dist plot shows the distribution of data from 2.5 to 4.0

min_payment_amt Variable:

Range of values: 7.690899999999999
Maximum min_payment_amt: 8.456
Minimum min_payment_amt: 0.7651
Mean: 3.7002009523809503
Median: 3.599
Standard deviation: 1.5035571308217792
Null values: False

min_payment_amt - First Quartile (Q1) is: 2.5615

min_payment_amt - Third Quartile (Q3) is: 4.76875
Interquartile range (IQR) for min_payment_amt is 2.2072499999999997

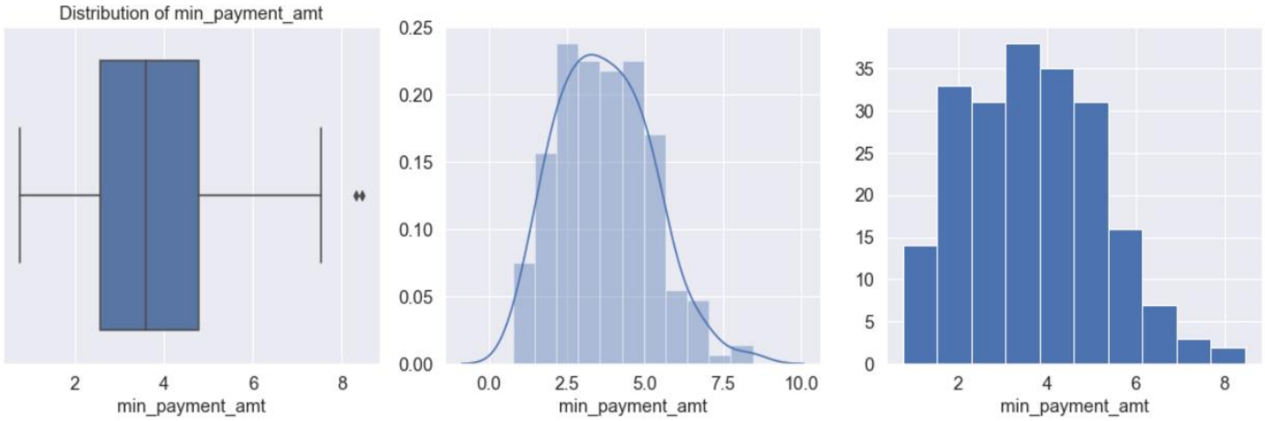


Fig6.Plots for min_payment_amt

→The box plot of the min payment amount variable shows few outliers.

→Min payment amount is positively skewed.

→The dist plot shows the distribution of data from 2 to 8

Max_spent_in_single_shopping:

Range: 2.0309999999999997
Maximum max_spent_in_single_shoppings: 6.55
Minimum max_spent_in_single_shopping: 4.519
Mean: 5.408071428571429
Median: 5.223000000000001
Standard deviation: 0.49148049910240543
Null values: False
max_spent_in_single_shopping - First Quartile (Q1) is: 5.045
max_spent_in_single_shopping - Third Quartile (Q3) is: 5.877
Interquartile range (IQR) of max_spent_in_single_shopping is 0.8319999999999999

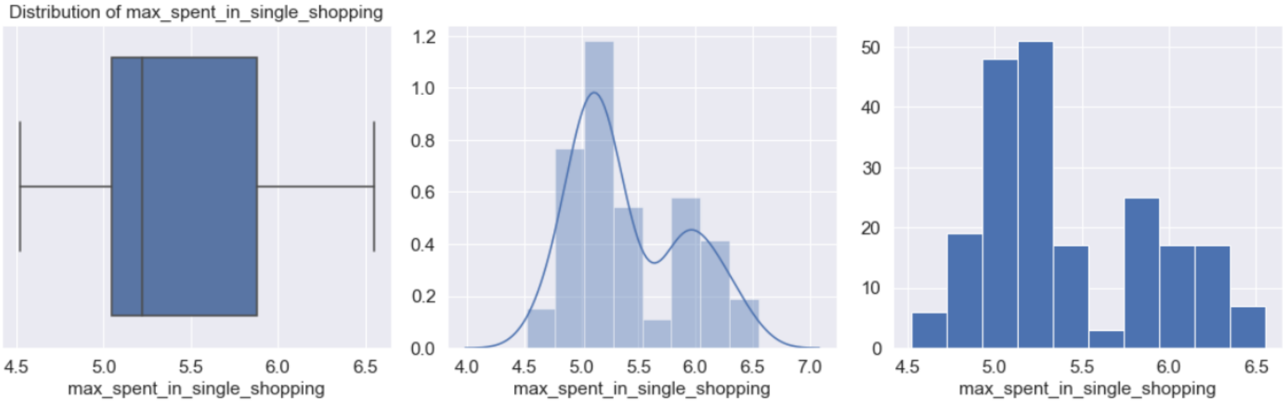


Fig7.Plots for max_spent_in_single_shopping

→The box plot of the max spent in single shopping variable has no outliers.

→Max spent in single shopping is positively skewed

→The dist plot shows the distribution of data from 4.5 to 6.5

Skewness:

max_spent_in_single_shopping : 0.561897
current_balance :0.525482
min_payment_amt:0.401667
spending:0.399889
advance_payments:0.386573
credit_limit: 0.134378
probability_of_full_payment:-0.537954

Outlier Treatment:

Boxplot before treating outliers:

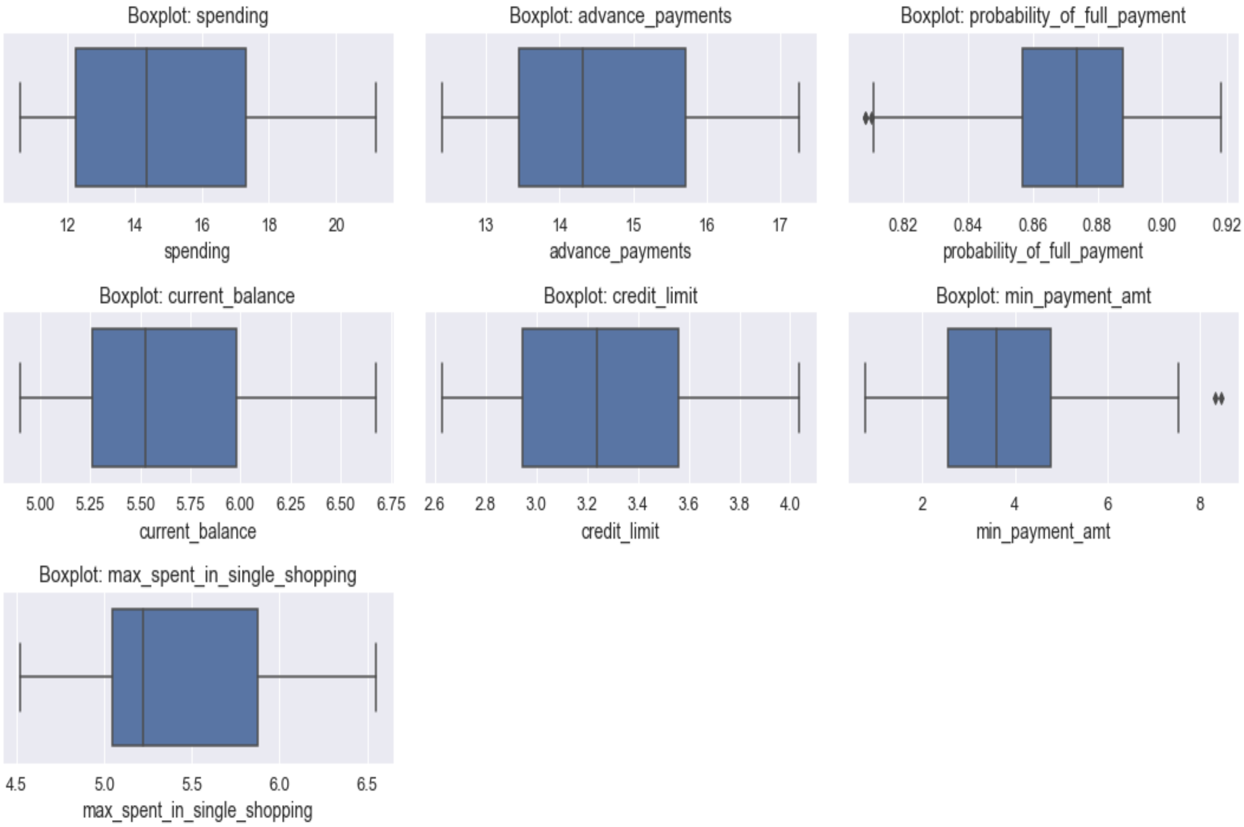


Fig8.Boxplot before treating outliers

There are very less outliers in this dataset. Only probability_of_full_payment & min_payment_amt has outliers.

We choose to drop them for clustering, because clustering is sensitive to outliers.

Steps for outlier treatment:

1. Sort the dataset in ascending order.
2. calculate the 1st and 3rd quartiles (Q1, Q3)
3. compute IQR=Q3-Q1.
4. compute lower bound = (Q1–1.5*IQR), upper bound = (Q3+1.5*IQR)
5. loop through the values of the dataset and check for those who fall below the lower bound and above the upper bound and mark them as outliers.

Boxplot After treating Outliers:

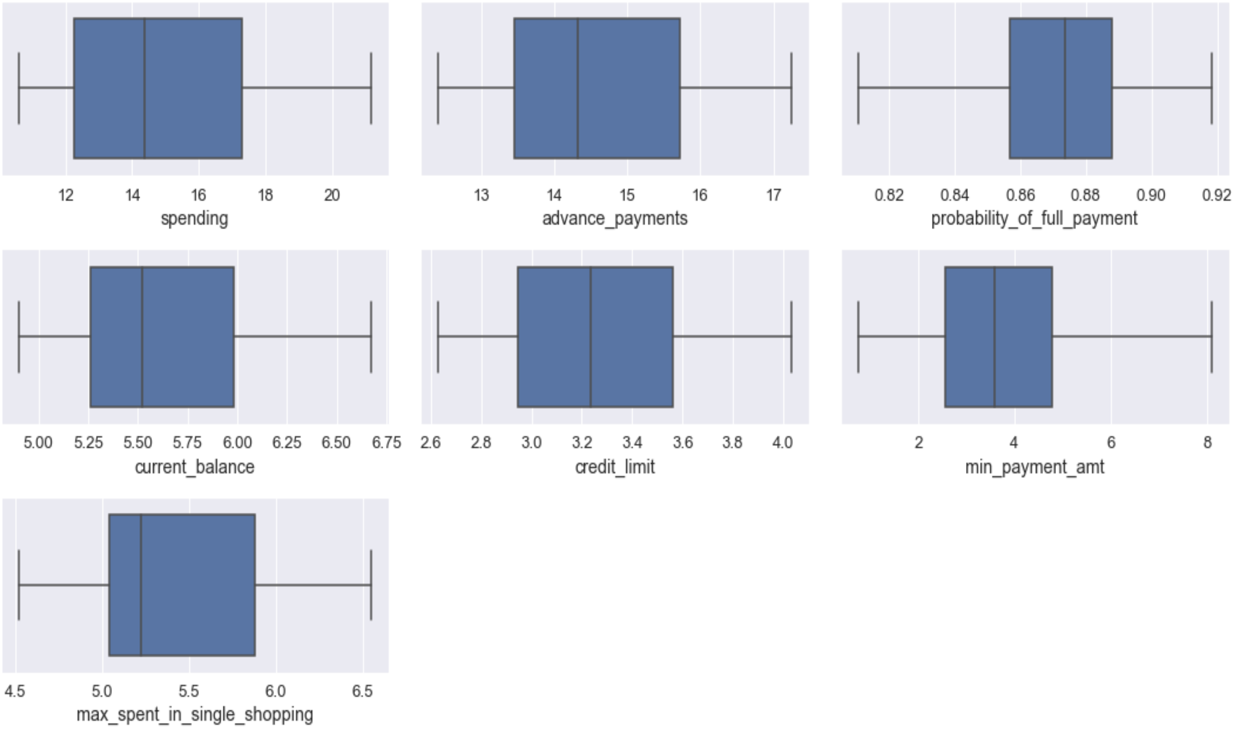


Fig9.Boxplot After treating outliers

Outliers are removed because clustering is sensitive to outliers.

MULTIVARIATE ANALYSIS:

Multivariate analysis is a Statistical procedure for analysis of data involving more than one type of measurement or observation. It may also mean solving problems where more than one dependent variable is analysed simultaneously with other variables.

we can perform multivariate analysis using **Pairplot** & **Correlation plot** as shown below:

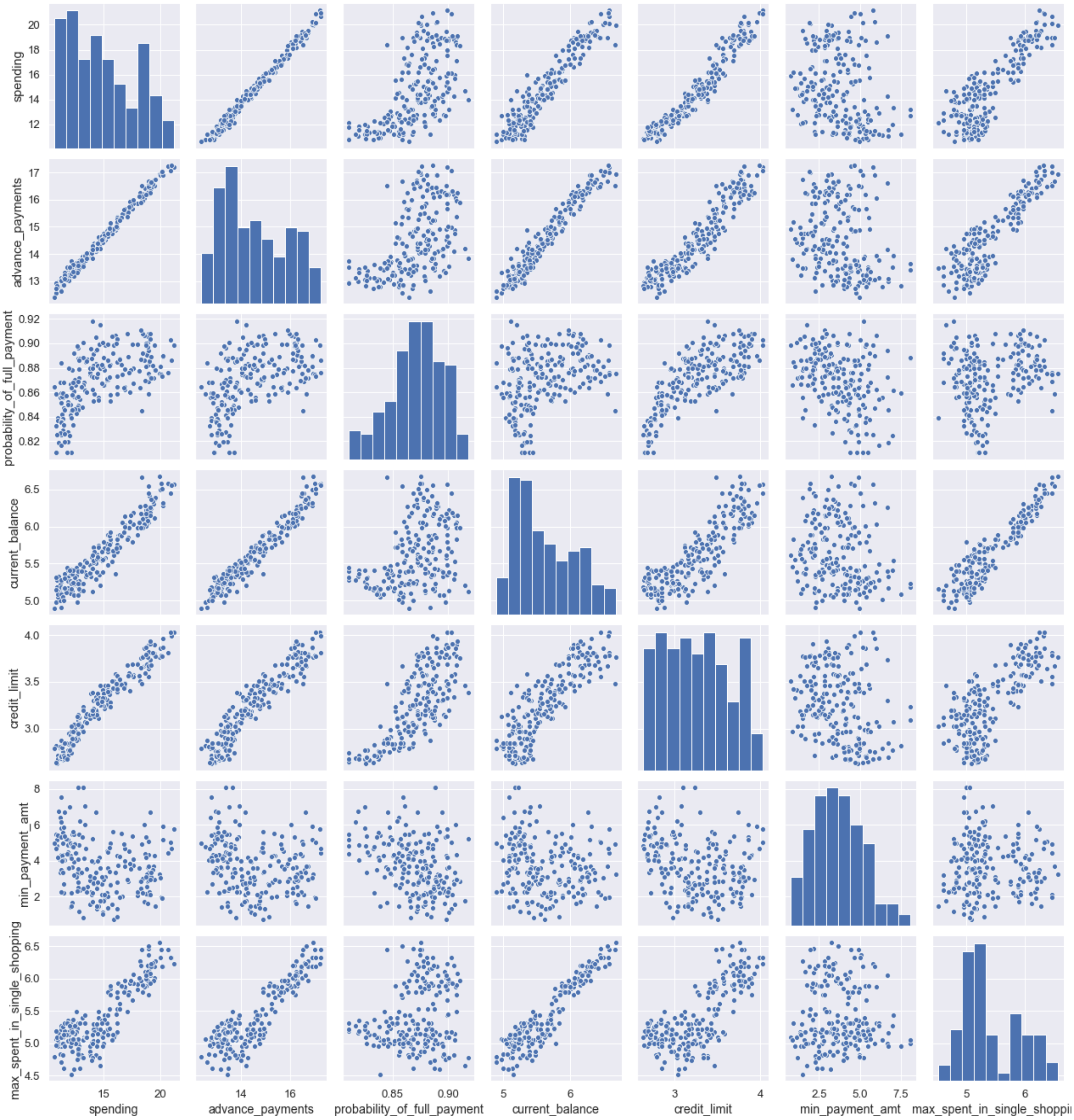


Fig10. Pairplot for bank market dataset

From the plot, we observe that there is strong positive correlation between

→ spending & advance_payments,

→ advance_payments & current_balance,

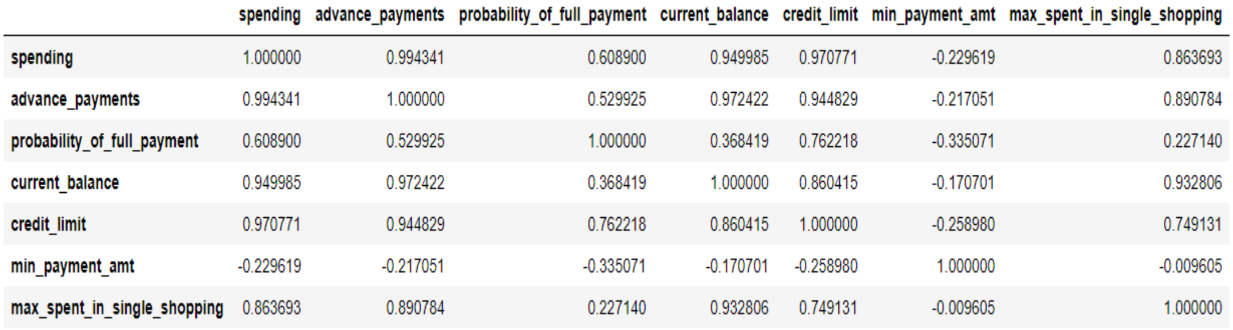
→ credit_limit & spending

→ spending & current_balance

→ credit_limit & advance_payments

→ max_spent_in_single_shopping current_balance

Correlation Matrix:



	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
spending	1.000000	0.994341	0.608900	0.949985	0.970771	-0.229619	0.863693
advance_payments	0.994341	1.000000	0.529925	0.972422	0.944829	-0.217051	0.890784
probability_of_full_payment	0.608900	0.529925	1.000000	0.368419	0.762218	-0.335071	0.227140
current_balance	0.949985	0.972422	0.368419	1.000000	0.860415	-0.170701	0.932806
credit_limit	0.970771	0.944829	0.762218	0.860415	1.000000	-0.258980	0.749131
min_payment_amt	-0.229619	-0.217051	-0.335071	-0.170701	-0.258980	1.000000	-0.009605
max_spent_in_single_shopping	0.863693	0.890784	0.227140	0.932806	0.749131	-0.009605	1.000000

Table 1.3 Correlation matrix

Heatmap:

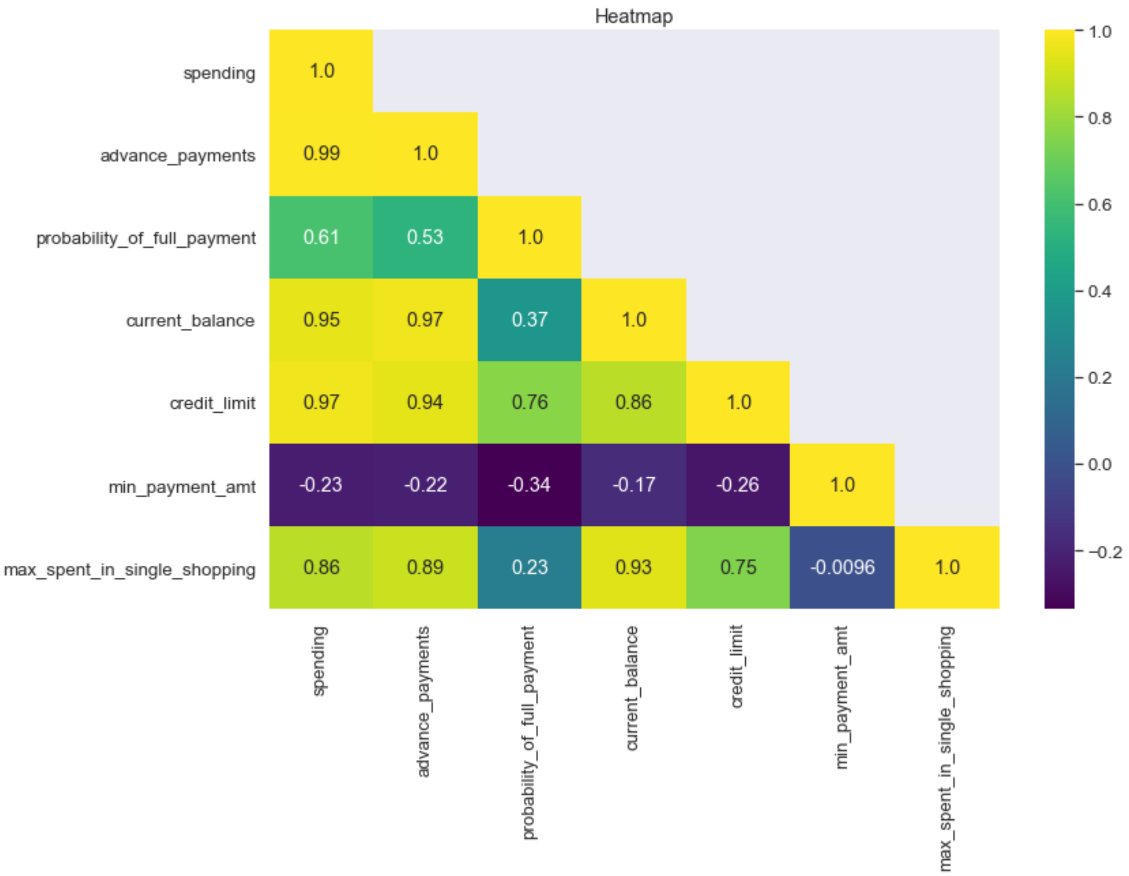
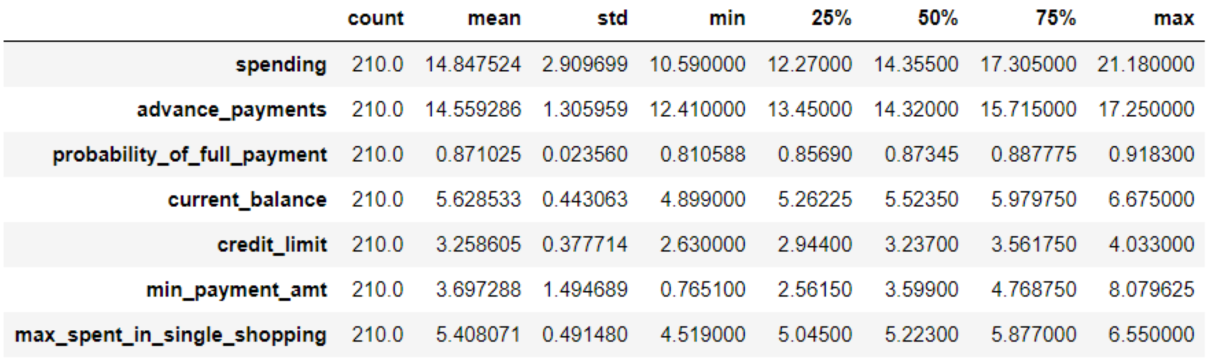


Fig.11 Heatmap

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

Scaling is a process of **transforming the data so that it fits within a specific scale**.

Data scaling is a recommended pre-processing step when working with all the methods in datamining. Data scaling can be achieved by normalizing or standardizing real-valued input and output variables.



	count	mean	std	min	25%	50%	75%	max
spending	210.0	14.847524	2.909699	10.590000	12.27000	14.35500	17.305000	21.180000
advance_payments	210.0	14.559286	1.305959	12.410000	13.45000	14.32000	15.715000	17.250000
probability_of_full_payment	210.0	0.871025	0.023560	0.810588	0.85690	0.87345	0.887775	0.918300
current_balance	210.0	5.628533	0.443063	4.899000	5.26225	5.52350	5.979750	6.675000
credit_limit	210.0	3.258605	0.377714	2.630000	2.94400	3.23700	3.561750	4.033000
min_payment_amt	210.0	3.697288	1.494689	0.765100	2.56150	3.59900	4.768750	8.079625
max_spent_in_single_shopping	210.0	5.408071	0.491480	4.519000	5.04500	5.22300	5.877000	6.550000

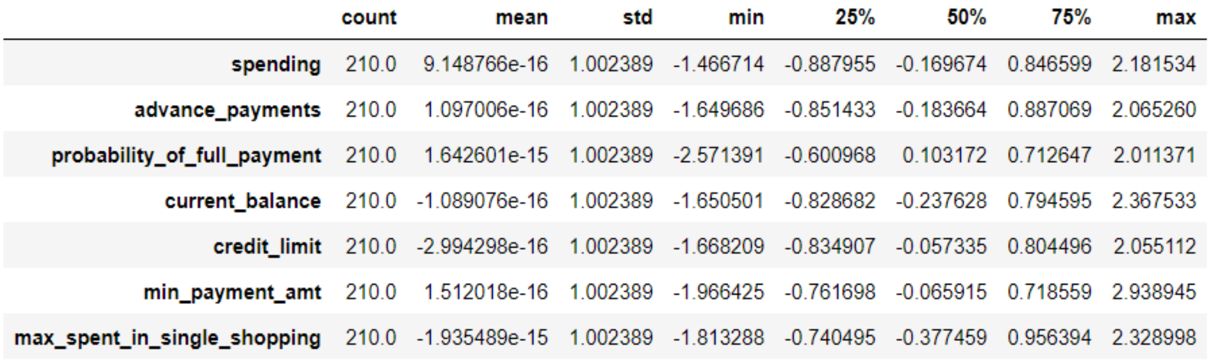
Table 1.4 Data before Scaling

→Scaling is necessary for clustering in this case as the values of variables are in different ranges with difference in variance and std.

→The spending and advance_payments have different scales and this may get more weightage.

→Hence, we need to scale the data to bring all of them to same scale.

→Here, I have used sklearn Standard scaler method to scale the data.



	count	mean	std	min	25%	50%	75%	max
spending	210.0	9.148766e-16	1.002389	-1.466714	-0.887955	-0.169674	0.846599	2.181534
advance_payments	210.0	1.097006e-16	1.002389	-1.649686	-0.851433	-0.183664	0.887069	2.065260
probability_of_full_payment	210.0	1.642601e-15	1.002389	-2.571391	-0.600968	0.103172	0.712647	2.011371
current_balance	210.0	-1.089076e-16	1.002389	-1.650501	-0.828682	-0.237628	0.794595	2.367533
credit_limit	210.0	-2.994298e-16	1.002389	-1.668209	-0.834907	-0.057335	0.804496	2.055112
min_payment_amt	210.0	1.512018e-16	1.002389	-1.966425	-0.761698	-0.065915	0.718559	2.938945
max_spent_in_single_shopping	210.0	-1.935489e-15	1.002389	-1.813288	-0.740495	-0.377459	0.956394	2.328998

Table 1.5 Data after Scaling

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

Hierarchical clustering, also known as hierarchical cluster analysis, is **an algorithm that groups similar objects into groups called clusters**. The endpoint is a set of clusters, where each cluster is distinct from each other cluster, and the objects within each cluster are broadly similar to each other. In order to decide which clusters should be combined (for agglomerative), or where a cluster should be split (for divisive), a measure of dissimilarity between sets of observations is required. In most methods of hierarchical clustering, this is achieved by use of an appropriate metric (a measure of distance between pairs of observations), and a linkage criterion which specifies the dissimilarity of sets as a function of the pairwise distances of observations in the sets.

Strategies for hierarchical clustering generally fall into two types:

- **Agglomerative**: This is a "bottom-up" approach: each observation starts in its own cluster, and pairs of clusters are merged as one moves up the hierarchy.
- **Divisive**: This is a "top-down" approach: all observations start in one cluster, and splits are performed recursively as one moves down the hierarchy.

For **Hierarchical clustering**, I am using Ward's method.

Ward's Method: The approach of calculating the similarity between two clusters is exactly the same as Group Average except that Ward's method calculates the sum of the square of the distances. Ward's method approach also does well in separating clusters if there is noise between clusters. Ward's method approach is also biased towards globular clusters.

For visualization, we use dendrogram. This also helps in finding the optimum number of clusters.

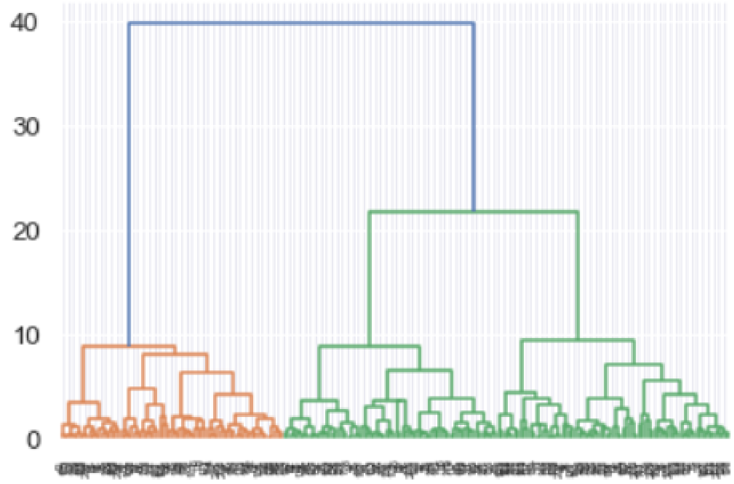


Fig.12 Dendogram for Hierarchial clustering

The above dendrogram indicates all the data points have clustered to different clusters by wards method.

To find the optimal number clusters through which we can solve our business objective we use truncate mode = lastp.

Here, I gave p=10.

Now the dendrogram is represented as below:

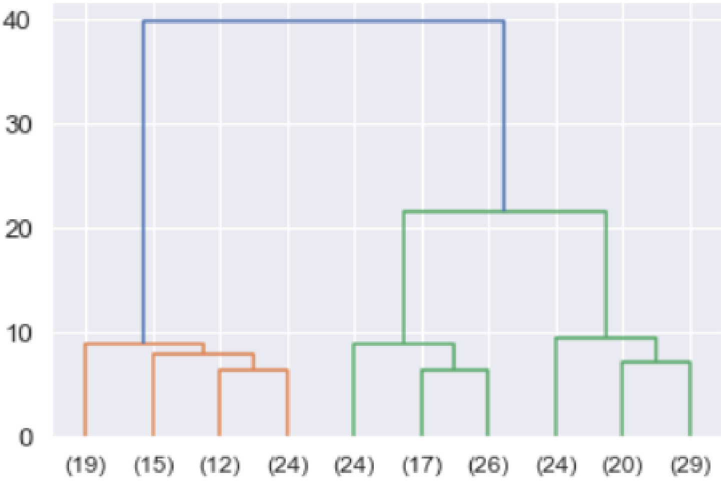
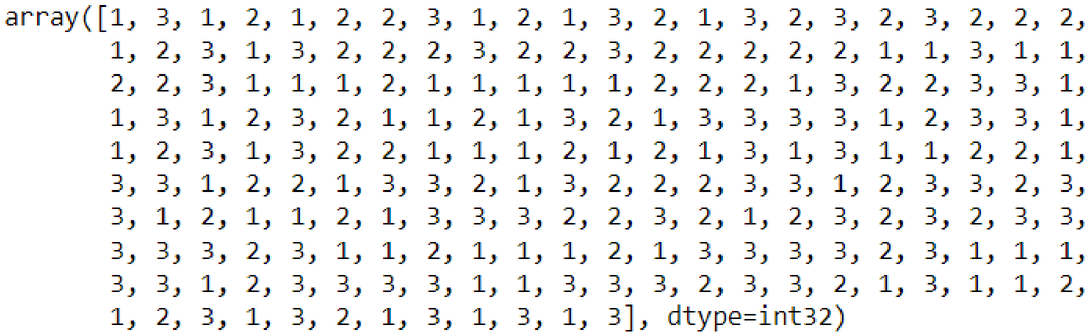


Fig13.Dendogram truncated

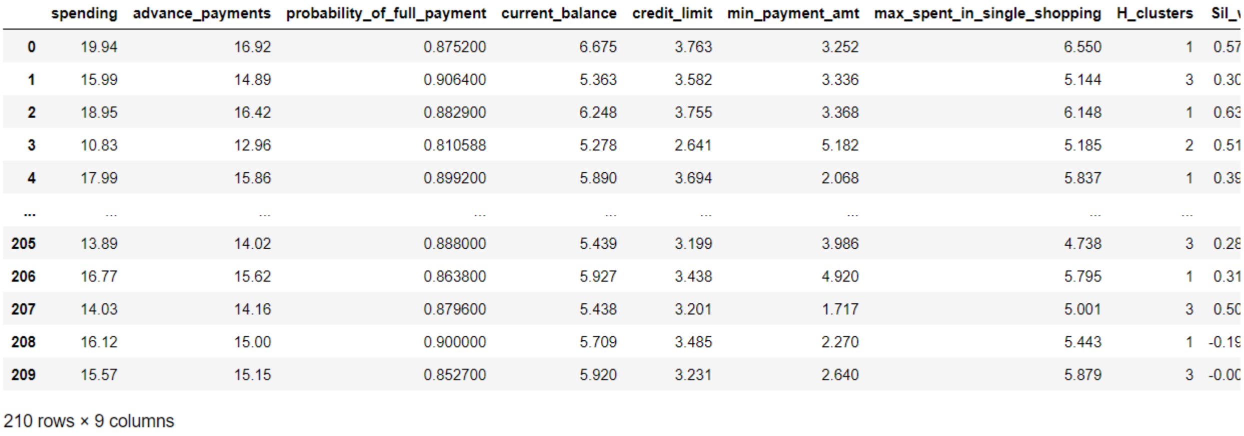
Now, we can understand all the data points have clustered into 3 clusters.

To map these clusters, we are using Fcluster with lnk, max number of clusters requested, and the Criterion 'max_clust'.

maxclust:
Finds a minimum threshold r so that the cophenetic distance between any two original observations in the same flat cluster is no more than r and no more than *t* flat clusters are formed.



```
array([1, 3, 1, 2, 1, 2, 2, 3, 1, 2, 1, 3, 2, 1, 3, 2, 3, 2, 3, 2, 2, 2,
       1, 2, 3, 1, 3, 2, 2, 2, 3, 2, 2, 3, 2, 2, 2, 2, 2, 1, 1, 3, 1, 1,
       2, 2, 3, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 2, 2, 1, 3, 2, 2, 3, 3, 1,
       1, 3, 1, 2, 3, 2, 1, 1, 2, 1, 3, 2, 1, 3, 3, 3, 3, 1, 2, 3, 3, 1,
       1, 2, 3, 1, 3, 2, 2, 1, 1, 1, 2, 1, 2, 1, 3, 1, 3, 1, 1, 2, 2, 1,
       3, 3, 1, 2, 2, 1, 3, 3, 2, 1, 3, 2, 2, 2, 3, 3, 1, 2, 3, 3, 2, 3,
       3, 1, 2, 1, 1, 2, 1, 3, 3, 3, 2, 2, 3, 2, 1, 2, 3, 2, 3, 2, 3, 3,
       3, 3, 3, 2, 3, 1, 1, 2, 1, 1, 1, 2, 1, 3, 3, 3, 3, 2, 3, 1, 1, 1,
       3, 3, 1, 2, 3, 3, 3, 3, 1, 1, 3, 3, 3, 2, 3, 3, 2, 1, 3, 1, 1, 2,
       1, 2, 3, 1, 3, 2, 1, 3, 1, 3, 1, 3], dtype=int32)
```

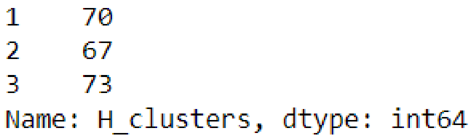


	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	H_clusters	Sil_
0	19.94	16.92	0.875200	6.675	3.763	3.252	6.550	1	0.57
1	15.99	14.89	0.906400	5.363	3.582	3.336	5.144	3	0.30
2	18.95	16.42	0.882900	6.248	3.755	3.368	6.148	1	0.63
3	10.83	12.96	0.810588	5.278	2.641	5.182	5.185	2	0.51
4	17.99	15.86	0.899200	5.890	3.694	2.068	5.837	1	0.39
...	...	...	...	...	...	...	...	...	
205	13.89	14.02	0.888000	5.439	3.199	3.986	4.738	3	0.28
206	16.77	15.62	0.863800	5.927	3.438	4.920	5.795	1	0.31
207	14.03	14.16	0.879600	5.438	3.201	1.717	5.001	3	0.50
208	16.12	15.00	0.900000	5.709	3.485	2.270	5.443	1	-0.19
209	15.57	15.15	0.852700	5.920	3.231	2.640	5.879	3	-0.00

210 rows × 9 columns

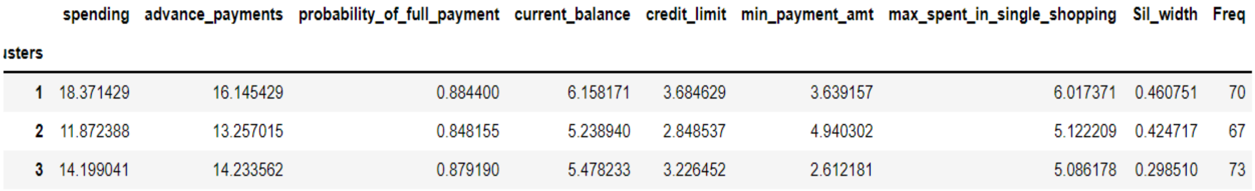
Table1.6 Clustered dataset

Now, we can look at the cluster frequency in our dataset,



```
1    70
2    67
3    73
Name: H_clusters, dtype: int64
```

Cluster profiling to understand the business problem.



	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	Sil_width	Freq
ısters									
1	18.371429	16.145429	0.884400	6.158171	3.684629	3.639157	6.017371	0.460751	70
2	11.872388	13.257015	0.848155	5.238940	2.848537	4.940302	5.122209	0.424717	67
3	14.199041	14.233562	0.879190	5.478233	3.226452	2.612181	5.086178	0.298510	73

Table1.7 Cluster Profiling

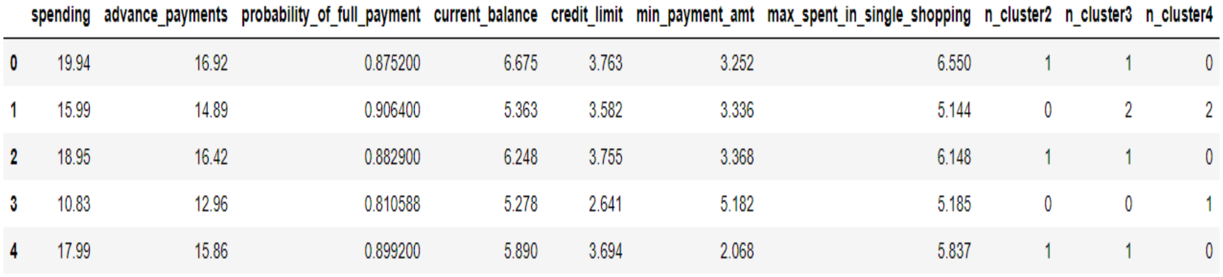
Agglomerative clustering:

1. The process starts by calculating the dissimilarity between the N objects.
2. Then two objects which when clustered together minimize a given agglomeration criterion, are clustered together thus creating a class comprising these two objects.
3. Then the dissimilarity between this class and the N-2 other objects is calculated using the agglomeration criterion. The two objects or classes of objects who's clustering together minimizes the agglomeration criterion are then clustered together.

In Agglomerative clustering, we chose maximum number of clusters as '2', linkage = 'average', affinity = 'cityblock'

Average-linkage is where **the distance between each pair of observations in each cluster are added up and divided by the number of pairs to get an average** inter-cluster distance.

Affinity- metric used to compute the linkage.



	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	n_cluster2	n_cluster3	n_cluster4
0	19.94	16.92	0.875200	6.675	3.763	3.252	6.550	1	1	0
1	15.99	14.89	0.906400	5.363	3.582	3.336	5.144	0	2	2
2	18.95	16.42	0.882900	6.248	3.755	3.368	6.148	1	1	0
3	10.83	12.96	0.810588	5.278	2.641	5.182	5.185	0	0	1
4	17.99	15.86	0.899200	5.890	3.694	2.068	5.837	1	1	0

Table1.8 Agglomerative cluster sample

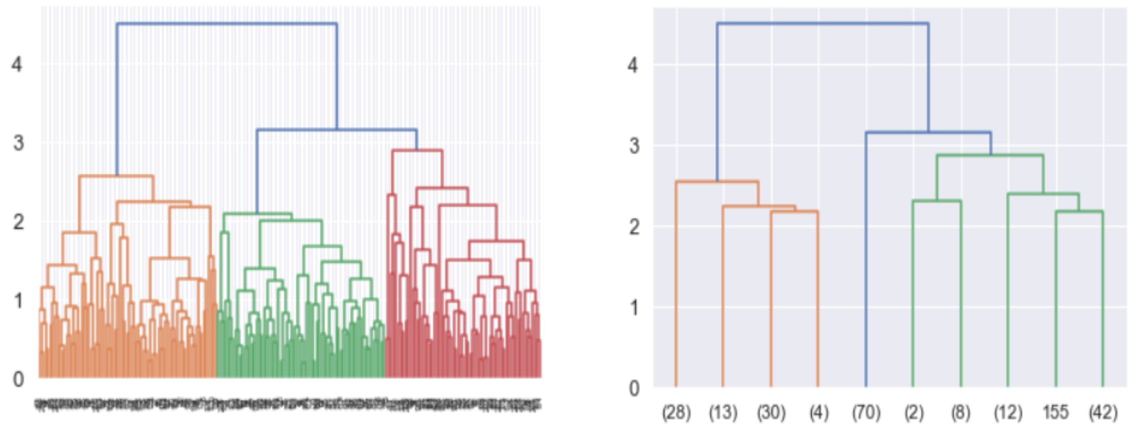


Fig14.Dendogram Unstructured & Truncated

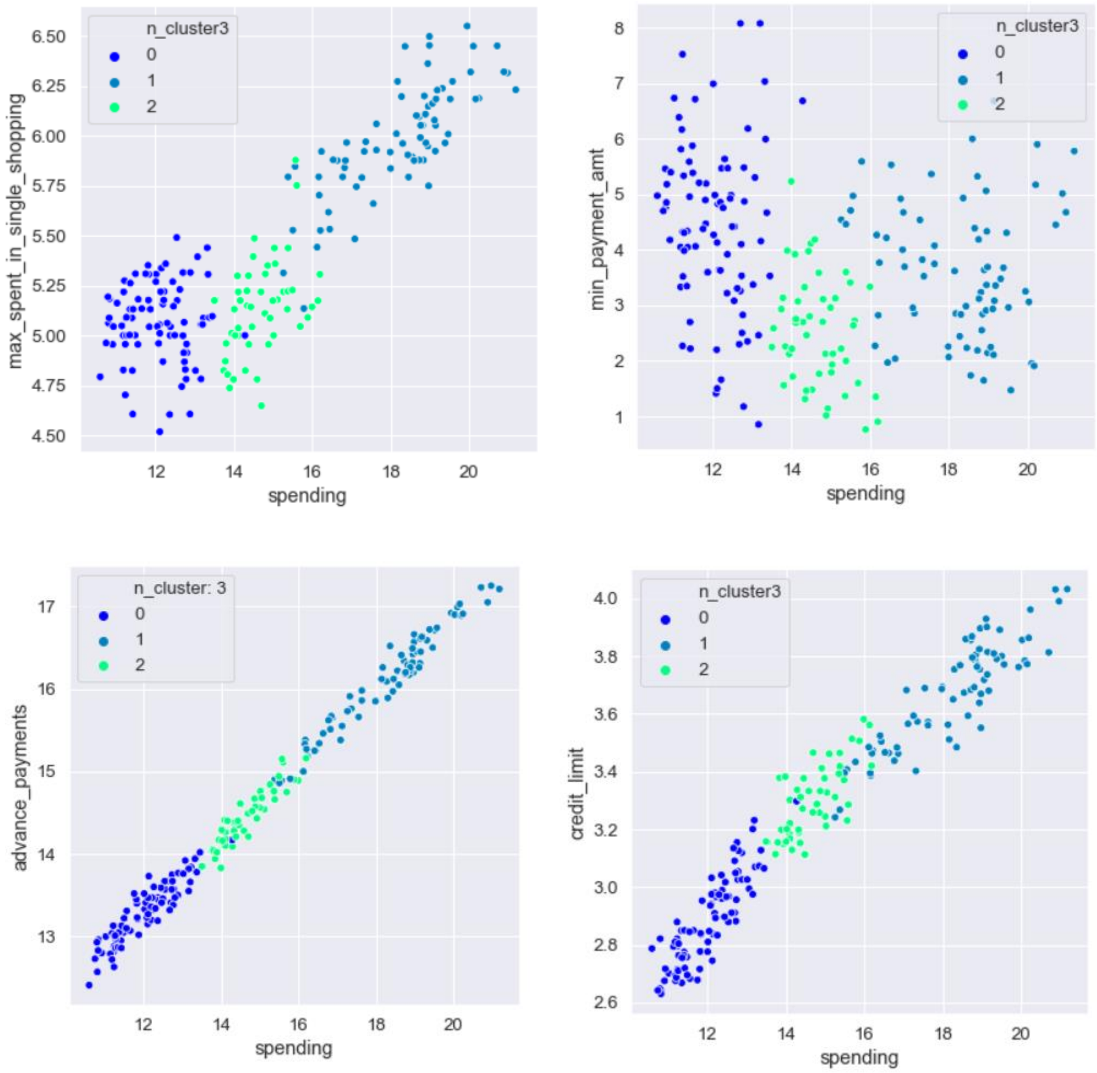


Fig15.Scatterplots for continuous features based on Agglomerative clusters

From the Analysis, we can conclude that

→Both the methods have almost similar means with some minimum variation.
→Cluster grouping based on the dendrogram, 3 or 4 clusters look good.
→After further analysis, based on the dataset we get 3 group cluster solution for hierarchical clustering
→ The 3 group cluster solution gives us the pattern based on high/medium/low spending with max_spent_in_single_shopping (high value item) and probability_of_full_payment(payment made).

→Hence, we can consider 3 group clusters are the number of optimum clusters.

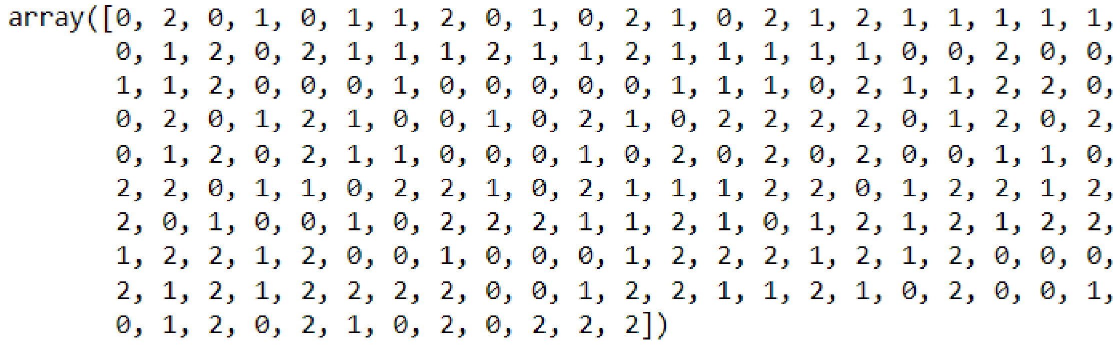
1.4 Apply K-Means clustering on scaled data and determine optimum clusters. Apply elbow curve and silhouette score. Explain the results properly. Interpret and write inferences on the finalized clusters.

K-Means Clustering is an **Unsupervised Learning algorithm**, which groups the unlabelled dataset into different clusters. Here K defines the number of pre-defined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on. It is an iterative algorithm that divides the unlabelled dataset into k different clusters in such a way that each dataset belongs only one group that has similar properties. It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabelled dataset on its own without the need for any training. It's a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

Randomly we decide to give n_clusters = 3 and we look at the distribution of clusters according to the n_clusters.

Now we apply K-means clustering to the sacled data.

Cluster output of all observations in the data:

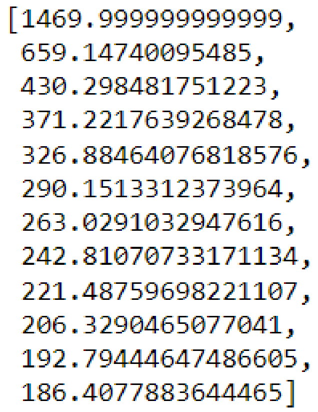


```
array([0, 2, 0, 1, 0, 1, 1, 2, 0, 1, 0, 2, 1, 0, 2, 1, 2, 1, 1, 1, 1, 1,
       0, 1, 2, 0, 2, 1, 1, 1, 2, 1, 1, 2, 1, 1, 1, 1, 1, 0, 0, 2, 0, 0,
       1, 1, 2, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 2, 1, 1, 2, 2, 0,
       0, 2, 0, 1, 2, 1, 0, 0, 1, 0, 2, 1, 0, 2, 2, 2, 2, 0, 1, 2, 0, 2,
       0, 1, 2, 0, 2, 1, 1, 0, 0, 0, 1, 0, 2, 0, 2, 0, 2, 0, 0, 1, 1, 0,
       2, 2, 0, 1, 1, 0, 2, 2, 1, 0, 2, 1, 1, 1, 2, 2, 0, 1, 2, 2, 1, 2,
       2, 0, 1, 0, 0, 1, 0, 2, 2, 2, 1, 1, 2, 1, 0, 1, 2, 1, 2, 1, 2, 2,
       1, 2, 2, 1, 2, 0, 0, 1, 0, 0, 0, 1, 2, 2, 2, 1, 2, 1, 2, 0, 0, 0,
       2, 1, 2, 1, 2, 2, 2, 2, 0, 0, 1, 2, 2, 1, 1, 2, 1, 0, 2, 0, 0, 1,
       0, 1, 2, 0, 2, 1, 0, 2, 0, 2, 2, 2])
```

We have 3 clusters 0,1,2

To find the optimal number of clusters, we can use k-elbow method.

Calculating WSS for other values of K-Elbow Method:



```
[1469.999999999999,
 659.14740095485,
 430.298481751223,
 371.2217639268478,
 326.88464076818576,
 290.1513312373964,
 263.0291032947616,
 242.81070733171134,
 221.48759698221107,
 206.3290465077041,
 192.79444647486605,
 186.4077883644465]
```

To find the inertia value for all the clusters from 1 to 13, used a for loop to find the optimal number of clusters.

The silhouette score for 3 clusters is good.

Sil_score_3: 0.4008059221522216

The elbow curve below also shows, after 3 clusters there is no huge drop in the values, so we select 3 clusters. So, adding the cluster results to our dataset to solve our business objective.

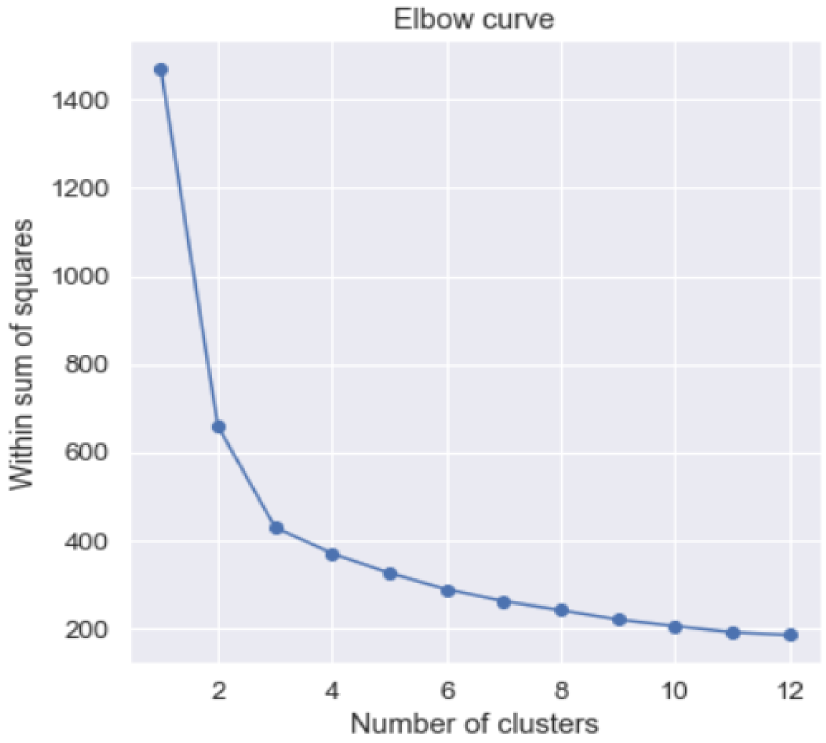
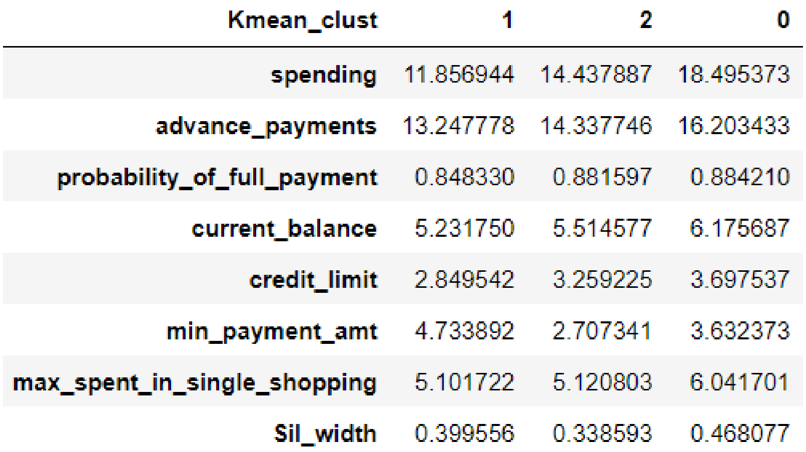


Fig16.Elbow Curve



Kmean_clust	1	2	0
spending	11.856944	14.437887	18.495373
advance_payments	13.247778	14.337746	16.203433
probability_of_full_payment	0.848330	0.881597	0.884210
current_balance	5.231750	5.514577	6.175687
credit_limit	2.849542	3.259225	3.697537
min_payment_amt	4.733892	2.707341	3.632373
max_spent_in_single_shopping	5.101722	5.120803	6.041701
Sil_width	0.399556	0.338593	0.468077

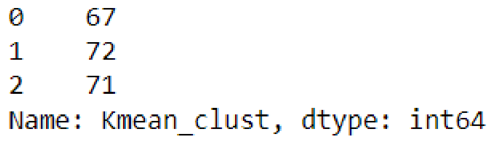
Table 1.9 K-Means clustering sample

This table shows the clusters to the dataset and also individual

sil_width score.

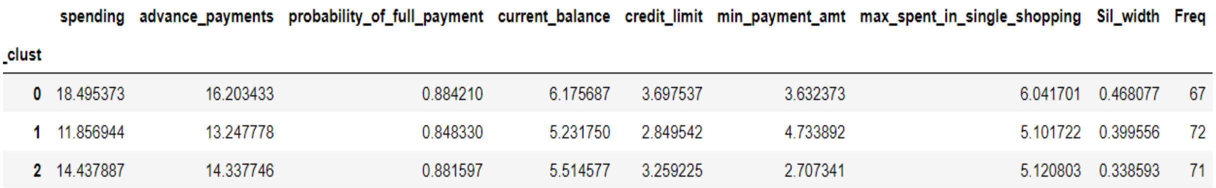
Cluster profiling:

Now, we can look at the cluster frequency in our dataset,



```
0    67
1    72
2    71
Name: Kmean_clust, dtype: int64
```

This frequency shows frequency of clusters to the dataset.



	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	Sil_width	Freq
_clust									
0	18.495373	16.203433	0.884210	6.175687	3.697537	3.632373	6.041701	0.468077	67
1	11.856944	13.247778	0.848330	5.231750	2.849542	4.733892	5.101722	0.399556	72
2	14.437887	14.337746	0.881597	5.514577	3.259225	2.707341	5.120803	0.338593	71

Table 1.10 K-Means cluster profiling

By K- Mean’s method, cluster we find cluster3 is optimal because after 3 there is no huge drop in inertia values. Also, the elbow curve seems to show similar results.

The silhouette width score of the K – means also seems very less value that indicates all the data points are properly clustered to the cluster. There is no mismatch in the data points with regards to clustering

Cluster grouping based on the dendrogram, 3 or 4 looks good. After further analysis, we conclude that group 3 cluster is the optimal cluster.

And three group cluster solution gives a pattern based on high/medium/low spending with max_spent_in_single_shopping (high value item) and probability_of_full_payment (payment made).

Customer segmentation Visualization:

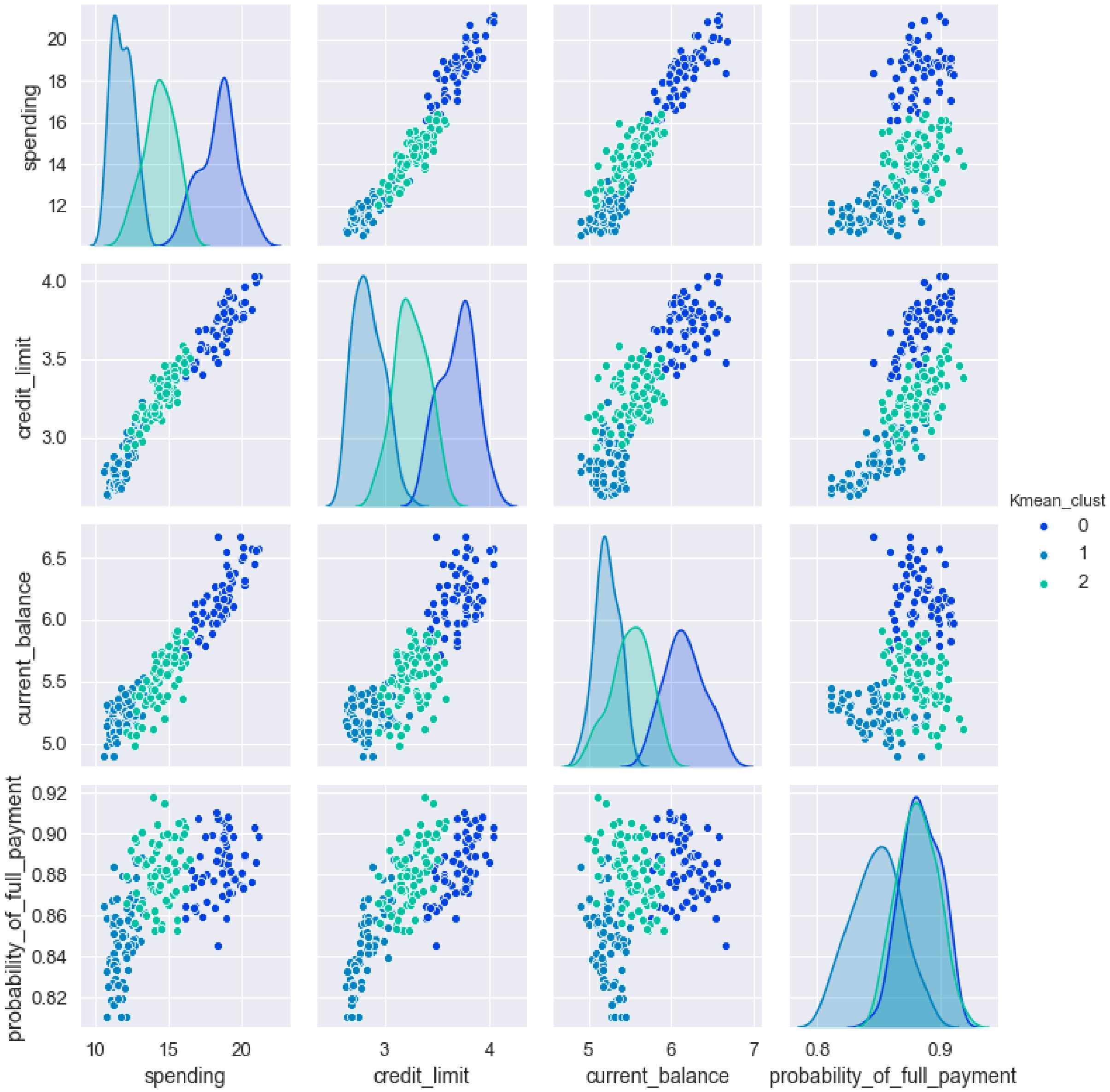


Fig17. Pairplot classified by Hierarchial clusters

→Here we can see that, cluster 1 is the highest spender & cluster_0 is the least spender among them

→In cluster_2, those who have less current_balance and high spending offer them specail loan to fulfil their requirements would increase revenue for the bank (spending vs current_balance plot).

→In cluster_0 those whose spending is equal or close to average spending of cluster_2 increase their credit_limit would lead them to purchase more with their card and ultimately benefit the bank (credit_limit vs spending).

→One thing is abnormal, that is high minimum payment amount for cluster_0 who are the least spender while the same for cluster_1 and cluster_2 are significantly lower.

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

Group 1: High Spending

Group 2: Low Spending

Group 3: Medium Spending

Promotional strategies for each cluster:

Group 1: High Spending Group

- Giving any reward points might increase their purchases.
- maximum max_spent_in_single_shopping is high for this group, so can be offered discount/offer on next transactions upon full payment.
- Increase their credit limit.
- Increase spending habits.
- Give loan against the credit card, as they are customers with good repayment record.
- Tie up with luxury brands, which will drive more one_time_maximun spending.

Group 2: Low Spending Group

- customers should be given remainders for payments. Offers can be provided on early payments to improve their payment rate.
- Increase their spending habits by tying up with grocery stores, utilities (electircity, phone, gas, others)

Group 3: Medium Spending Group

- They are potential target customers who are paying bills and doing purchases and maintaining comparatively good credit score. So we can increase credit limit or can lower down interest rate.
- Promote premium cards/loyality cars to increase transcations.
- Increase spending habits by trying with premium ecommerce sites, travel portal, travel airlines/hotel, as this will encourge them to spend more.

Problem 2: CART-RF-ANN

EXECUTIVE SUMMARY:

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.

Dataset for Problem 2: insurance_part2_data-1.csv

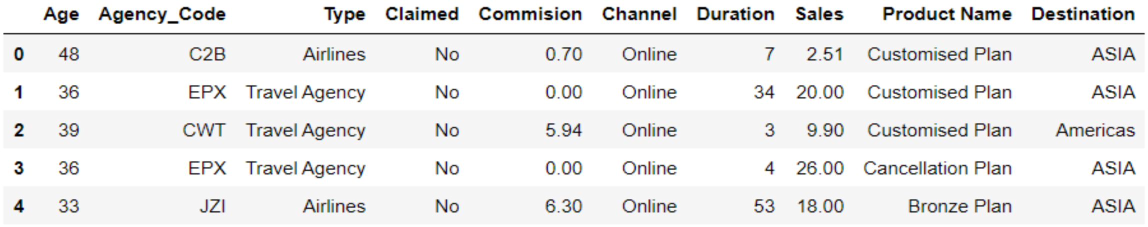
Attribute Information:

1. Target: Claim Status (Claimed)
2. Code of tour firm (Agency_Code)
3. Type of tour insurance firms (Type)
4. Distribution channel of tour insurance agencies (Channel)
5. Name of the tour insurance products (Product)
6. Duration of the tour (Duration in days)
7. Destination of the tour (Destination)
8. Amount worth of sales per customer in procuring tour insurance policies in rupees (in 100's)
9. The commission received for tour insurance firm (Commission is in percentage of sales)
10.Age of insured (Age)

INTRODUCTION:

In this problem, we perform CART-RF-ANN models to predict the claim status and provide recommendations to management. The dataset consists of 3000 records and 1o features. For Performing CART-RF-ANN models, we need to import all the required libraries and initial descriptive analysis to be performed.

SAMPLE OF A DATASET:



	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

Table 2.1 Sample of insurance dataset

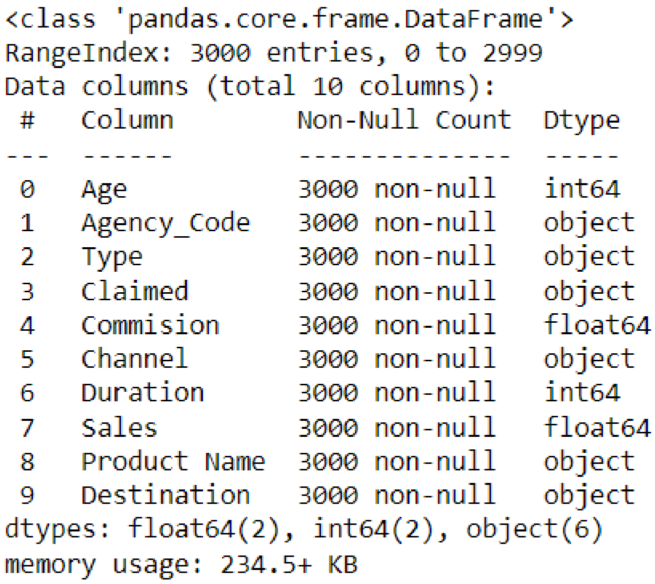
2.1 Read the data, do the necessary initial steps, and exploratory data analysis (Univariate, Bi-variate, and multivariate analysis).

EXPLORATORY DATA ANALYSIS:

→The dataset having 10 features with 3000 records and no missing values.

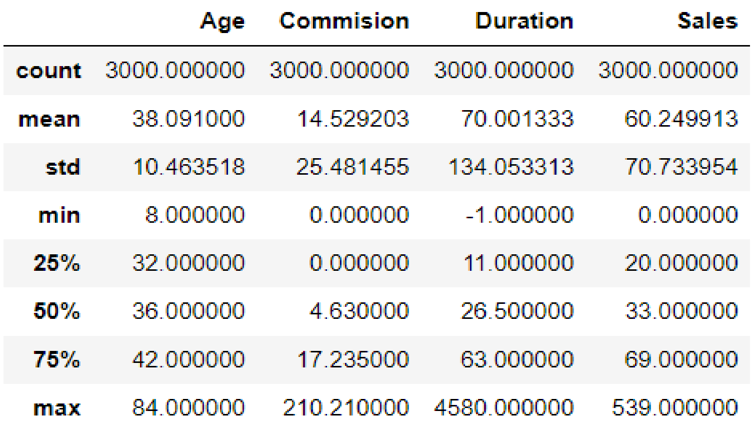
→Age, commission, duration, sales are numeric variables and the other variables are categorical.

→There are 9 independent variables and one target variable -- Claimed.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3000 entries, 0 to 2999
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Age           3000 non-null   int64  
 1   Agency_Code   3000 non-null   object 
 2   Type          3000 non-null   object 
 3   Claimed       3000 non-null   object 
 4   Commision     3000 non-null   float64
 5   Channel       3000 non-null   object 
 6   Duration      3000 non-null   int64  
 7   Sales         3000 non-null   float64
 8   Product Name  3000 non-null   object 
 9   Destination   3000 non-null   object 
dtypes: float64(2), int64(2), object(6)
memory usage: 234.5+ KB
```

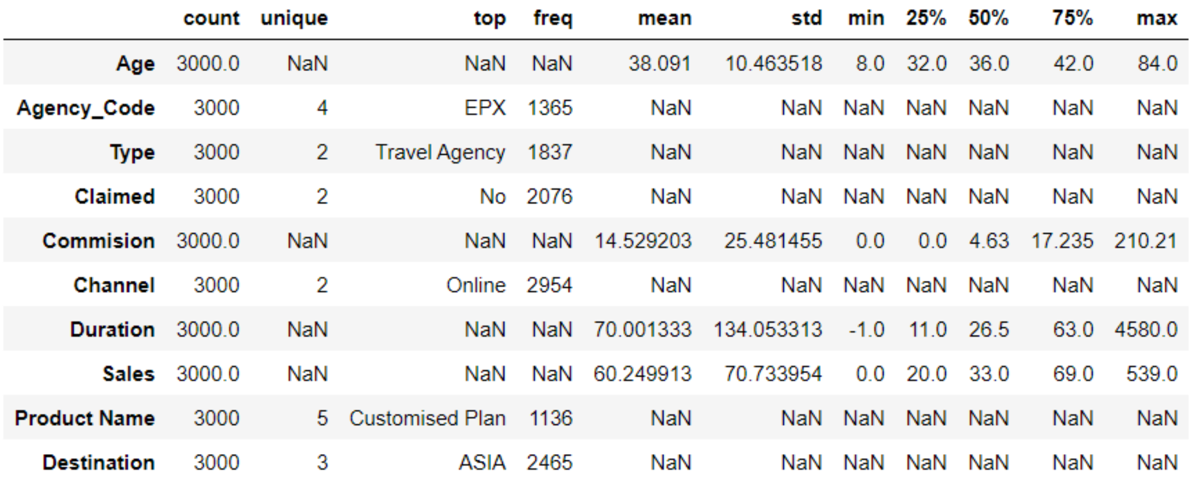
Summary of the data:



	Age	Commision	Duration	Sales
count	3000.000000	3000.000000	3000.000000	3000.000000
mean	38.091000	14.529203	70.001333	60.249913
std	10.463518	25.481455	134.053313	70.733954
min	8.000000	0.000000	-1.000000	0.000000
25%	32.000000	0.000000	11.000000	20.000000
50%	36.000000	4.630000	26.500000	33.000000
75%	42.000000	17.235000	63.000000	69.000000
max	84.000000	210.210000	4580.000000	539.000000

Table2.2 Summary of dataset

→The feature duration has negative values, this might be the wrong entry. For Commission & sales, mean and median significantly vary.



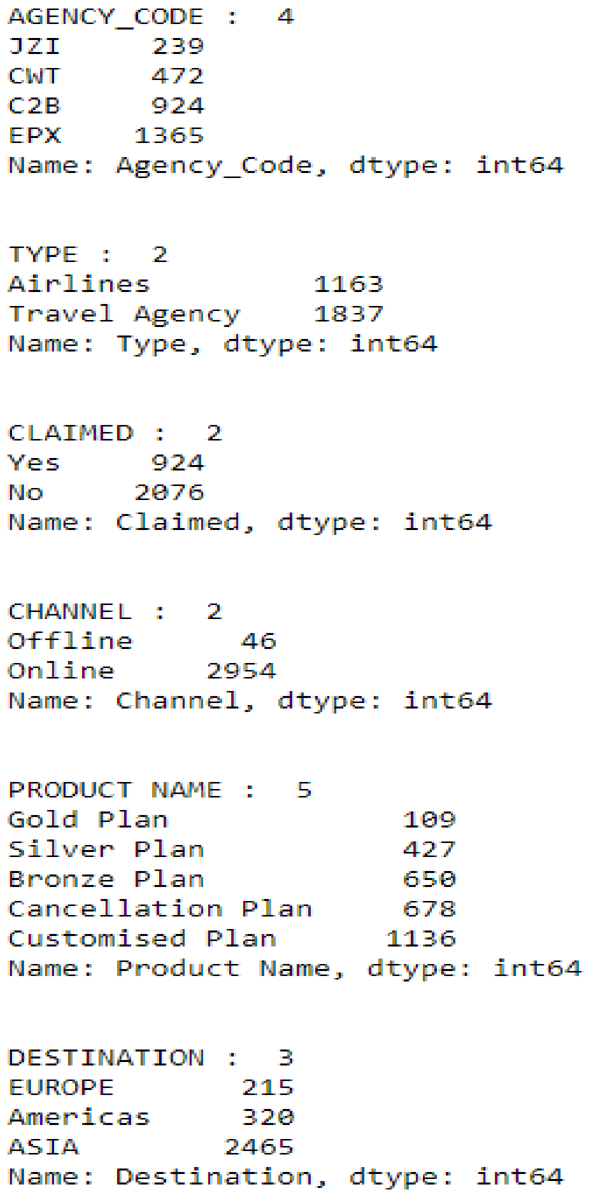
	count	unique	top	freq	mean	std	min	25%	50%	75%	max
Age	3000.0	NaN	NaN	NaN	38.091	10.463518	8.0	32.0	36.0	42.0	84.0
Agency_Code	3000	4	EPX	1365	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Type	3000	2	Travel Agency	1837	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Claimed	3000	2	No	2076	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Commision	3000.0	NaN	NaN	NaN	14.529203	25.481455	0.0	0.0	4.63	17.235	210.21
Channel	3000	2	Online	2954	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Duration	3000.0	NaN	NaN	NaN	70.001333	134.053313	-1.0	11.0	26.5	63.0	4580.0
Sales	3000.0	NaN	NaN	NaN	60.249913	70.733954	0.0	20.0	33.0	69.0	539.0
Product Name	3000	5	Customised Plan	1136	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Destination	3000	3	ASIA	2465	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Table 2.3 Descriprive Summary including NAN values

→categorical variable maximum unique count is 5.

→There are 139 duplicate rows present in the data. We should not drop duplicates from the data. Because we don't know whether these are same passengers or different as id or name is not given

→ Info function clearly indicates the dataset has object, integer and float so we have to change the object data type to numeric value



```
AGENCY_CODE :  4
JZI      239
CWT      472
C2B      924
EPX     1365
Name: Agency_Code, dtype: int64


TYPE :  2
Airlines         1163
Travel Agency    1837
Name: Type, dtype: int64


CLAIMED :  2
Yes      924
No      2076
Name: Claimed, dtype: int64


CHANNEL :  2
Offline      46
Online     2954
Name: Channel, dtype: int64


PRODUCT NAME :  5
Gold Plan             109
Silver Plan           427
Bronze Plan           650
Cancellation Plan     678
Customised Plan      1136
Name: Product Name, dtype: int64


DESTINATION :  3
EUROPE       215
Americas     320
ASIA        2465
Name: Destination, dtype: int64
```

UNIVARIATE ANALYSIS:

Distplot showing distribution of data:

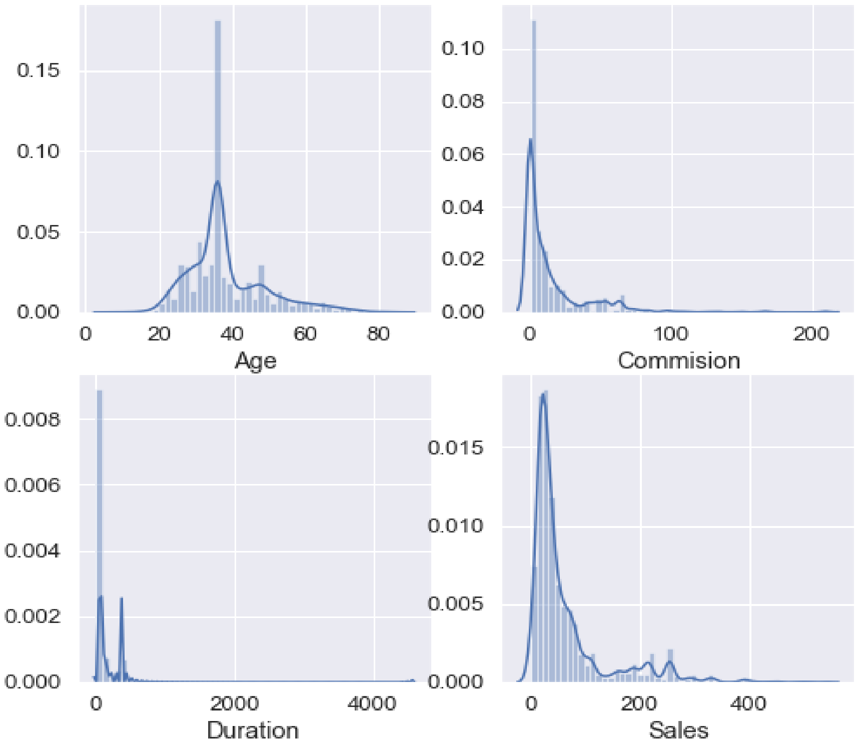


Fig18. Distplot

Boxplot showing outliers in the data:

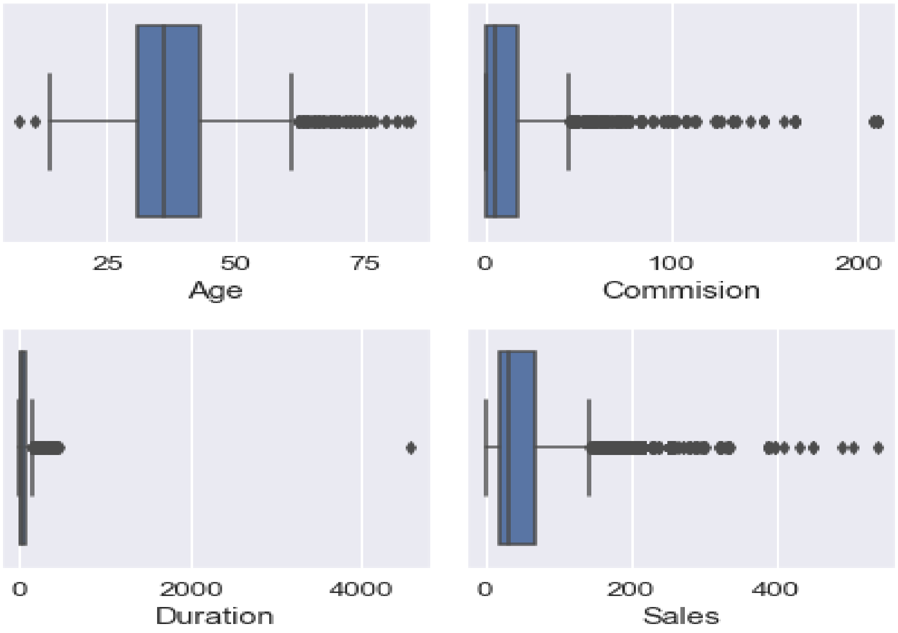


Fig19. Boxplot

Age:

The box plot of the age variable shows outliers.

Data is positively skewed.

The dist plot shows the distribution of data.

In the range of 30 to 40 is where the majority of the distribution lies.

Commission:

The box plot of the commission variable shows outliers.

Data is positively skewed.

The dist plot shows the distribution of data from 0 to 30s.

Duration:

The box plot of the duration variable shows outliers.

Data is positively skewed.

The dist plot shows the distribution of data from 0 to 100

Sales:

The box plot of the sales variable shows outliers.

Data is positively skewed.

The dist plot shows the distribution of data from 0 to 300.

Categotical Variables:

Agency_Code

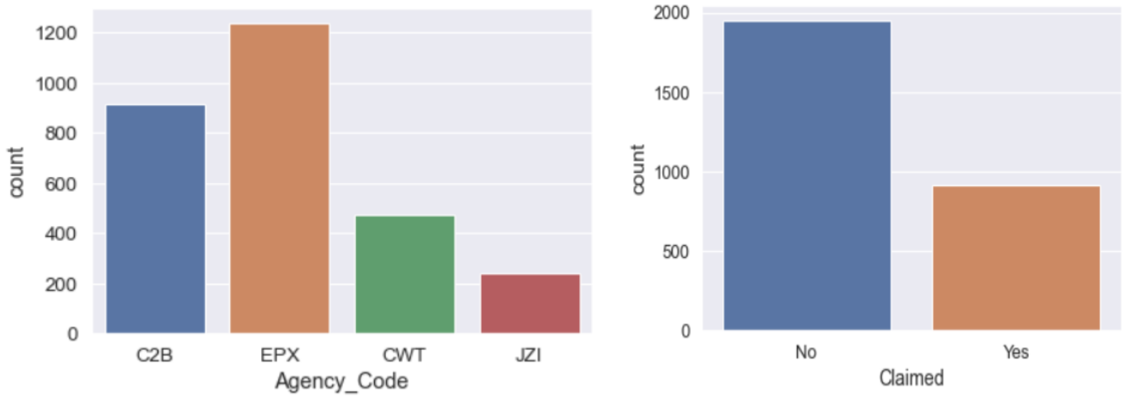


Fig20.Countplot for Agency_Code & Claimed

The distribution of the agency code, shows us EPX with maximum frequency. The distribution of the claimed, having No with maximum frequency.

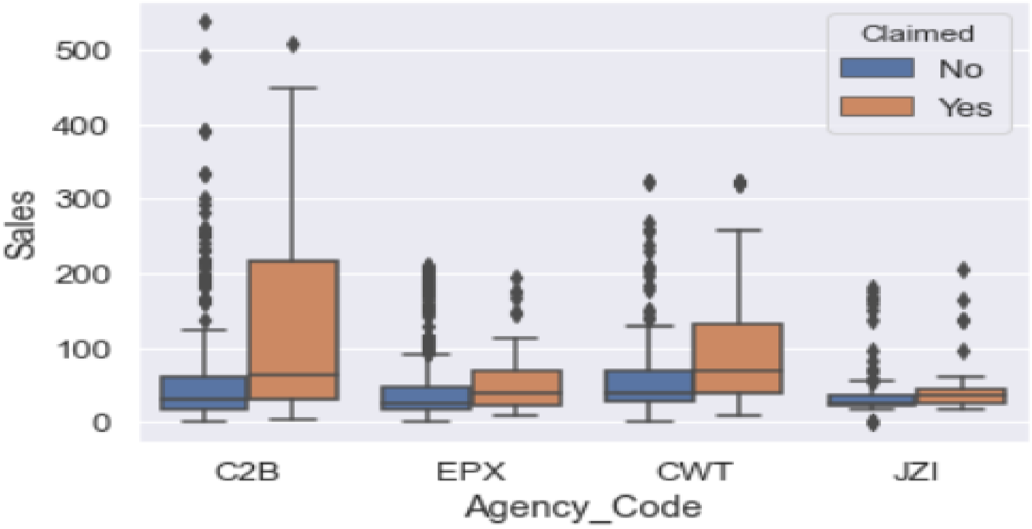


Fig21. Boxplot (Agency_code with hue=claimed)

The box plot shows the split of sales with different agency code and also hue having claimed column. It seems that C2B have claimed more claims than other agencies.

Type:

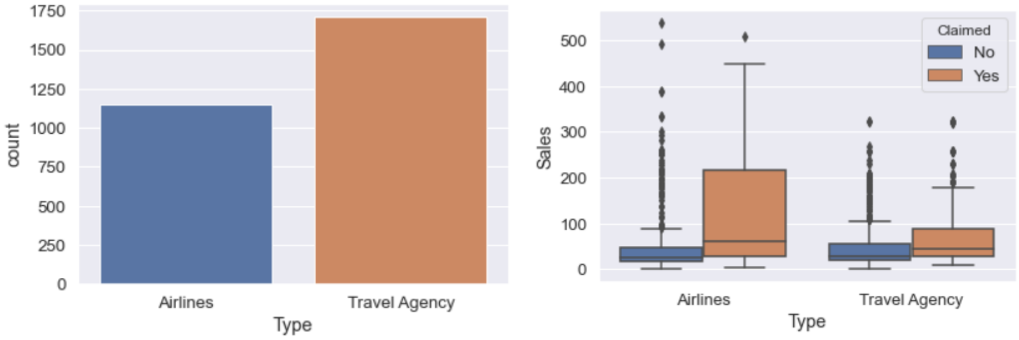


Fig22. Countplot & Boxplot for Type variable

The box plot shows the split of sales with different type and also hue having claimed column.

We could understand airlines type has more claims.

Channel:

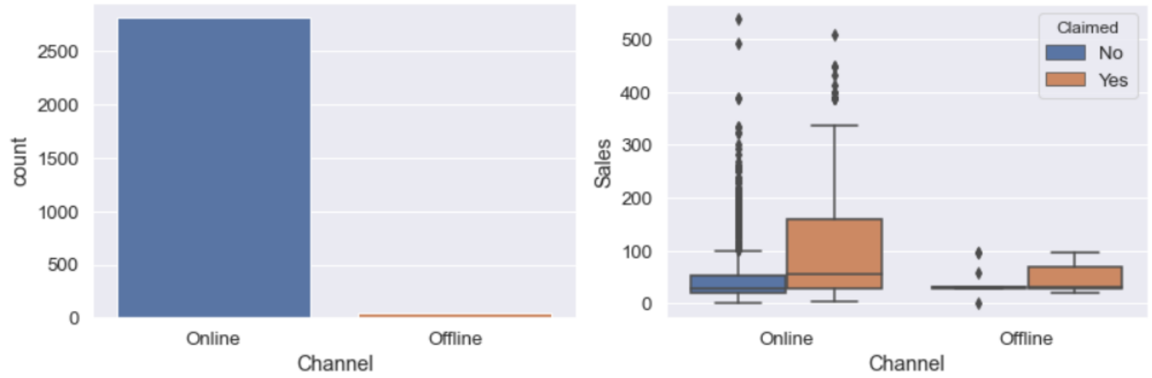


Fig23. Countplot & Boxplot for Channel variable

The majority of customers have used online medium, very less with offline medium. The box plot shows the split of sales with different channel and also hue having claimed column.

Product Name:

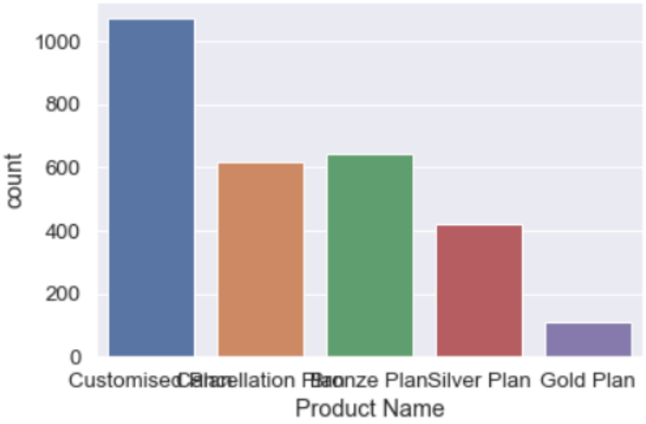


Fig24. Countplot for Product name

Customized plan seems to be most liked plan by customers when compared to all other plans.

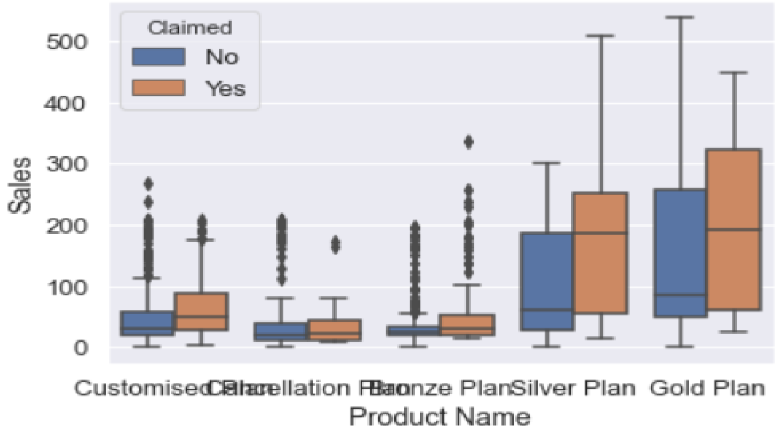


Fig25.Boxplot for Product Name

The box plot shows the split of sales with different product name and also hue having claimed column.

Destination:

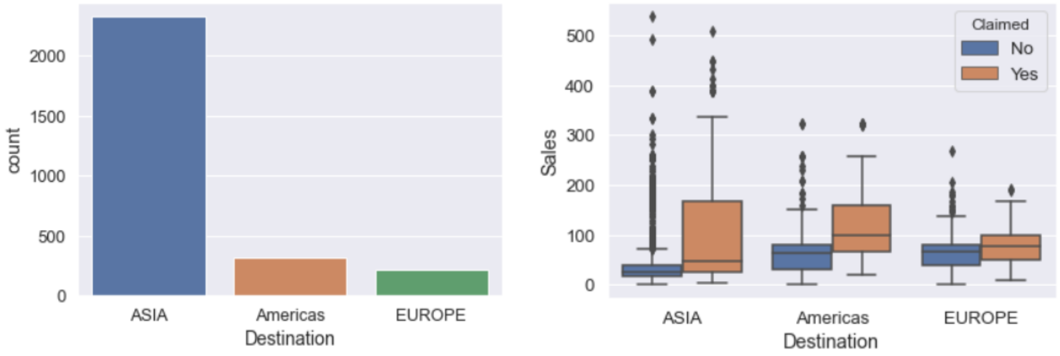


Fig26.Countplot & Boxplot for Destination

Asia is where customers choose when compared with other destination places. The box plot shows the split of sales with different destination and also hue having claimed column.

MULTIVARIATE ANALYSIS:

→Not much of multi collinearity observed.

→No negative correlation.

→Only positive correlation.

Checking Pairwise distribution of continuous variables:

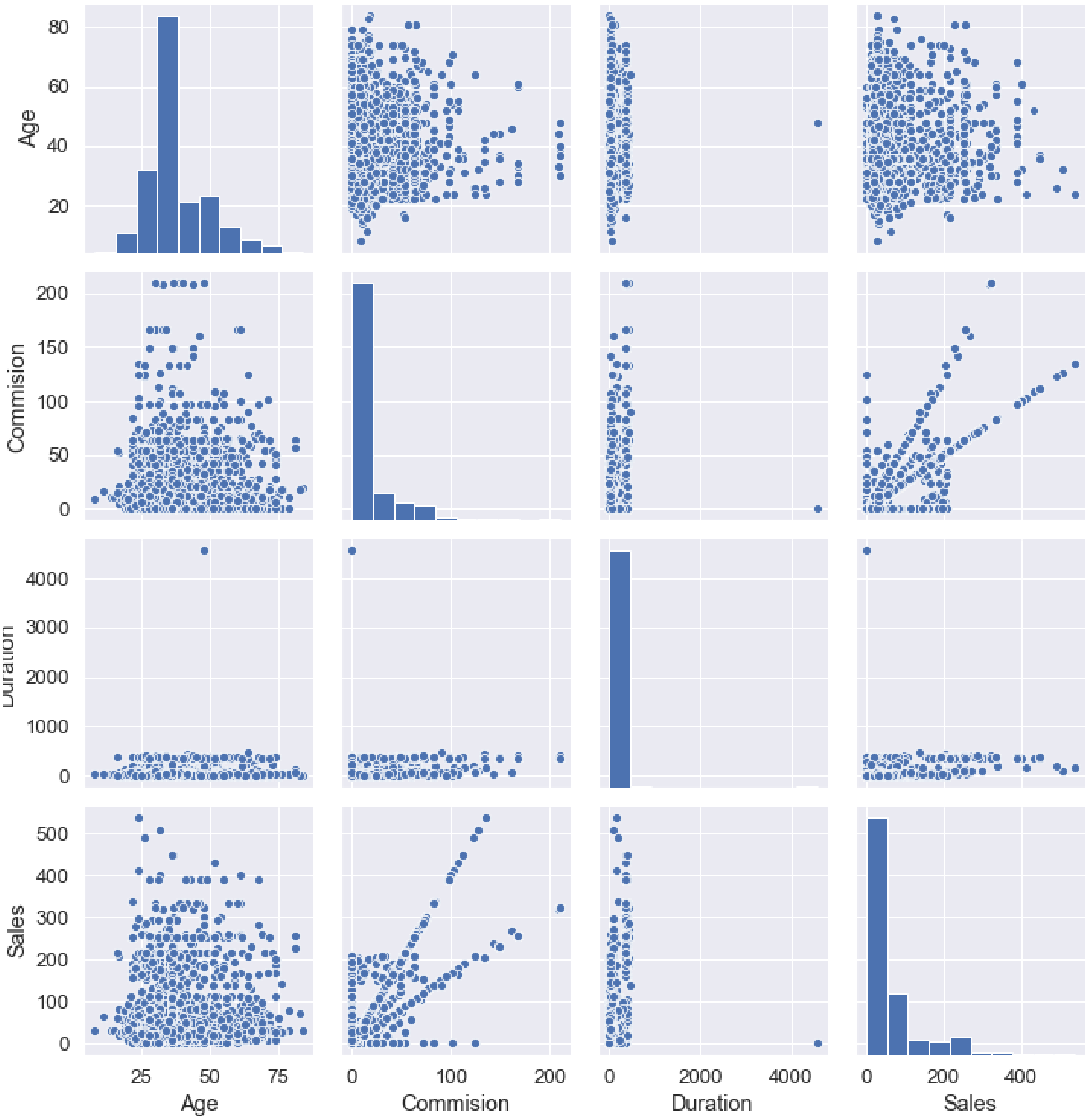


Fig27. Pairplot fot Multivariate analysis

Correlation Heatmap:

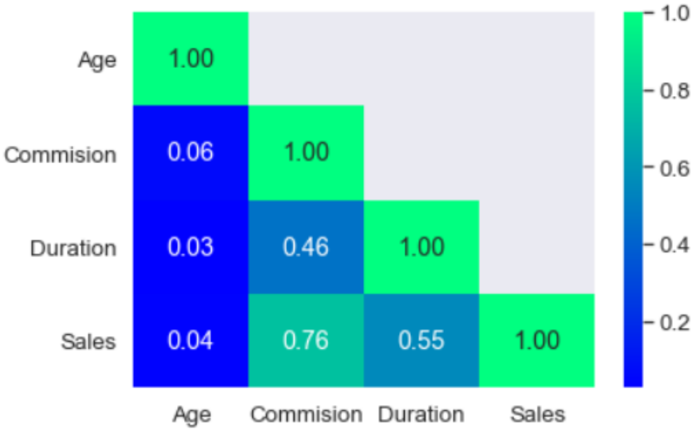
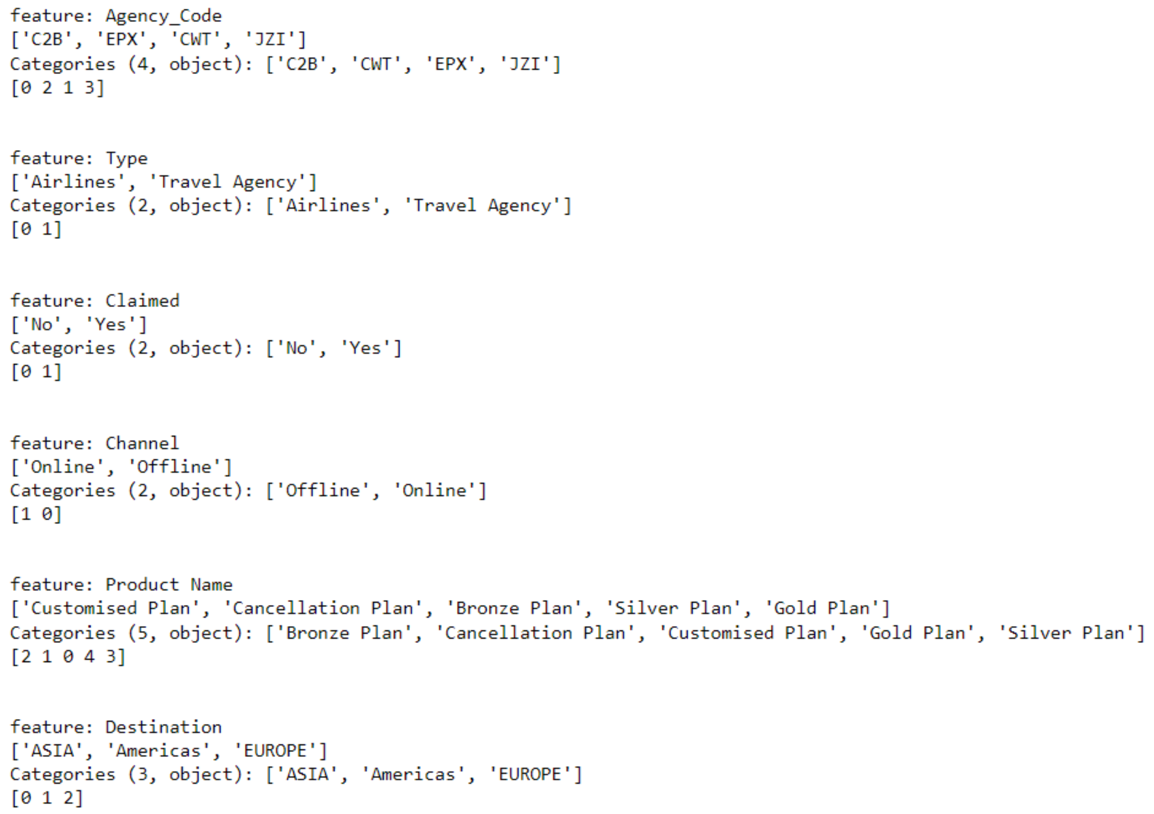


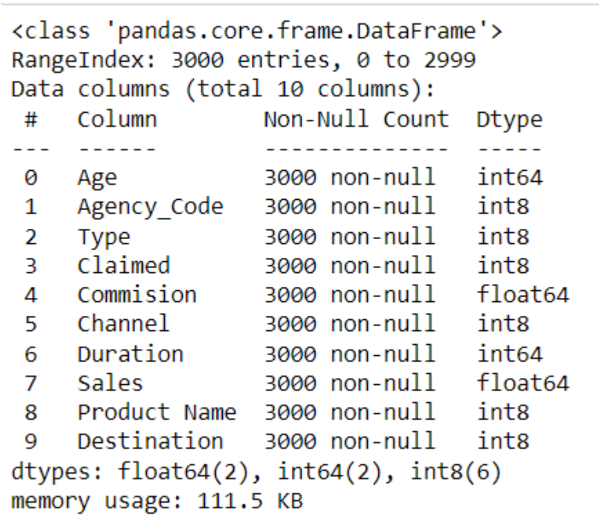
Fig28.Heatmap

Converting all objects to Categorical codes:

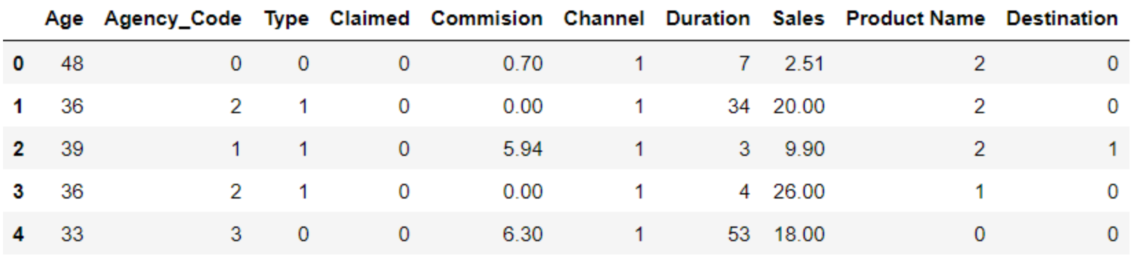


```
feature: Agency_Code
['C2B', 'EPX', 'CWT', 'JZI']
Categories (4, object): ['C2B', 'CWT', 'EPX', 'JZI']
[0 2 1 3]


feature: Type
['Airlines', 'Travel Agency']
Categories (2, object): ['Airlines', 'Travel Agency']
[0 1]


feature: Claimed
['No', 'Yes']
Categories (2, object): ['No', 'Yes']
[0 1]


feature: Channel
['Online', 'Offline']
Categories (2, object): ['Offline', 'Online']
[1 0]


feature: Product Name
['Customised Plan', 'Cancellation Plan', 'Bronze Plan', 'Silver Plan', 'Gold Plan']
Categories (5, object): ['Bronze Plan', 'Cancellation Plan', 'Customised Plan', 'Gold Plan', 'Silver Plan']
[2 1 0 4 3]


feature: Destination
['ASIA', 'Americas', 'EUROPE']
Categories (3, object): ['ASIA', 'Americas', 'EUROPE']
[0 1 2]
```

Checking info & sample of data:



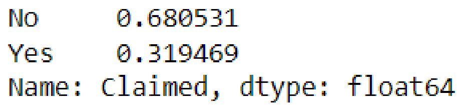
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3000 entries, 0 to 2999
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Age           3000 non-null   int64  
 1   Agency_Code   3000 non-null   int8   
 2   Type          3000 non-null   int8   
 3   Claimed       3000 non-null   int8   
 4   Commision     3000 non-null   float64
 5   Channel       3000 non-null   int8   
 6   Duration      3000 non-null   int64  
 7   Sales         3000 non-null   float64
 8   Product Name  3000 non-null   int8   
 9   Destination   3000 non-null   int8   
dtypes: float64(2), int64(2), int8(6)
memory usage: 111.5 KB
```



	Age	Agency_Code	Type	Claimed	Commision	Channel	Duration	Sales	Product Name	Destination
0	48	0	0	0	0.70	1	7	2.51	2	0
1	36	2	1	0	0.00	1	34	20.00	2	0
2	39	1	1	0	5.94	1	3	9.90	2	1
3	36	2	1	0	0.00	1	4	26.00	1	0
4	33	3	0	0	6.30	1	53	18.00	0	0

Table 2.4 Sample for Numeric data

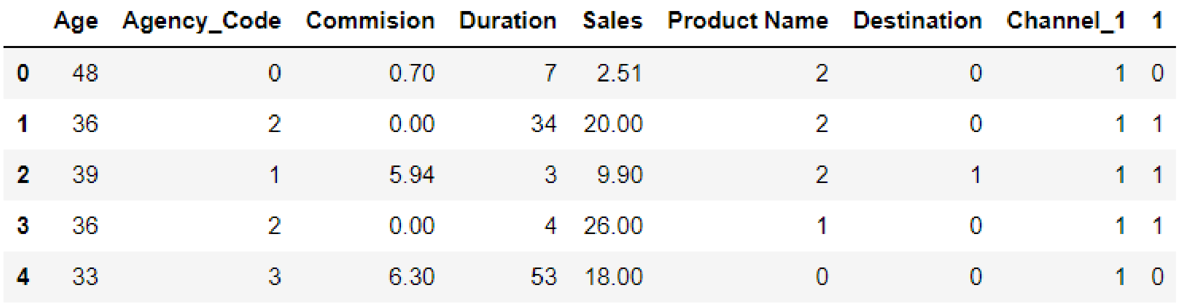
Proportion of 0’s and 1’s:



```
No     0.680531
Yes    0.319469
Name: Claimed, dtype: float64
```

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

→Extracting the target column into separate vectors for training and test set.

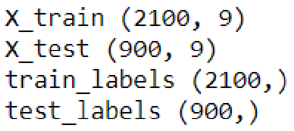


	Age	Agency_Code	Commision	Duration	Sales	Product Name	Destination	Channel_1	1
0	48	0	0.70	7	2.51	2	0	1	0
1	36	2	0.00	34	20.00	2	0	1	1
2	39	1	5.94	3	9.90	2	1	1	1
3	36	2	0.00	4	26.00	1	0	1	1
4	33	3	6.30	53	18.00	0	0	1	0

Table2.5 Sample of training data

→For training and testing purpose we are splitting the dataset into train and test data in the ratio of 70:30.

→Checking the dimensions of train and the test data



```
X_train (2100, 9)
X_test (900, 9)
train_labels (2100,)
test_labels (900,)
```

→Now, we have bifurcated the dataset into train and test.

→We have also taken out the target column out of train and test data into separate vector for evaluation purposes.

CART (Decision Tree Classifier)

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piecewise constant approximation.

- Simple to understand and to interpret. Trees can be visualised.
- Requires little data preparation. Other techniques often require data normalisation, dummy variables need to be created and blank values to be removed. Note however that this module does not support missing values.

DecisionTreeClassifier is a class capable of performing multi-class classification on a dataset.

As with other classifiers, **DecisionTreeClassifier** takes as input two arrays: an array X, sparse or dense, of shape (n_samples, n_features) holding the training samples, and an array Y of integer values, shape (n_samples,), holding the class labels for the training samples:

We can also export the tree in Graphviz format using the **export_graphviz** exporter. If you use the conda package manager, the graphviz binaries and the python package can be installed with conda install python-graphviz.

TRAIN AND TEST SPLIT:

→Before moving to train test split the encoded data set is segregated based on dependent variable (‘claimed’) and independent variables (rest other features).

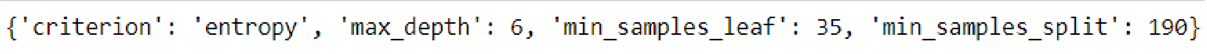
→Train test split is made using train_size=0.70 and random_state=0

→X_train and X_test shape is (2100,9) and (900,9) respectively.

CART:

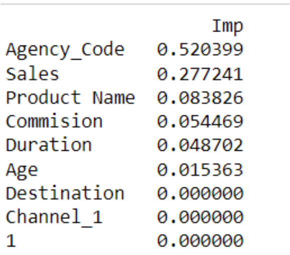
→Grid search method has been used to get the best parameter for the model.
→Hyperparameters used- ‘criterion': ['gini', 'entropy'],
'min_samples_split': [190,200, 205],
'min_samples_leaf': [35,40,45],
'max_depth': [6,7,8]

→Min samples leaf is generally taken as 1% of the total no of rows in the dataset. Min samples split is thrice the min samples leaf and after various iteration the above hyperparameters were selected to choose the best estimator.
→Max depth is pruning parameter, level at which you want to prune the tree.
Best estimators were found to be



```
{'criterion': 'entropy', 'max_depth': 6, 'min_samples_leaf': 35, 'min_samples_split': 190}
```

FEATURE IMPORTANCE:



```
                 Imp
Agency_Code   0.520399
Sales         0.277241
Product Name  0.083826
Commision     0.054469
Duration      0.048702
Age           0.015363
Destination   0.000000
Channel_1     0.000000
1             0.000000
```

Agency_Code, Product Name and Sales are the important and deciding feature for RF classification with 52%, 27% and 8% importance respectively, rest all can be removed for Decision Tree classification testing.

RANDOM FOREST MODEL:

A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

→Grid search method has been used to get the best parameter for the model.
→Hyperparameters used-

```
                  'n_estimators': [300],
                 'criterion': ['gini', 'entropy'],
                 'min_samples_split': [200],
                 'min_samples_leaf': [ 35, 45, 55],
                 'max_depth': [6,7,8],
                'max_features': [4,5,6]
```

→Min samples leaf is generally taken as 1% of the total no of rows in the dataset. Min samples split is thrice the min samples leaf and after various iteration the above hyperparameters were tuned to the above-mentioned values and gridsearchcv best estimator was used to choose the best estimator.

ANN MODEL (MLP CLASSIFIER):

The general form of an ANN is a **"black box" model** of a type that is often used to model high-dimensional, nonlinear data. However, most ANNs are used to solve prediction problems for some system, as opposed to formal model-building or development of underlying knowledge of how the system works.

Class **MLPClassifier** implements a multi-layer perceptron (MLP) algorithm that trains using Backpropagation.

MLP trains on two arrays: array X of size (n_samples, n_features), which holds the training samples represented as floating point feature vectors; and array y of size (n_samples,), which holds the target values (class labels) for the training samples:

Before creating the model, we need to scale the X_train and X_test and we will use StandardScaler technique for scaling. Grid search method has been used to get the best parameter for the model.

Hyperparameters used-

```
               'hidden_layer_sizes': [(900)],
                'activation': ['relu'],
                'tol':[.00001 ],
```

'max_iter':[1000],
'verbose':[True],
'solver':['sgd']

Hidden layer sizes are used to define number of hidden layers and number of nodes used to make the perceptron or the neural network, after many iterations 900 nodes single layered neural network was chosen as it was able to generate the best f1 and precision scores. Rectified Linear unit is chosen as the activation function for this neural network.

Tolerance is set to 0.00001 as lower the tolerance, higher the time to converge and higher the accuracy thus better the model, it helps in minimizing the loss function slowly and accurately. Solver is stochastic gradient descent. Min samples split is thrice the min samples leaf and after various iteration the above hyperparameters were tuned to the above-mentioned values and gridsearchcv best estimator was used to choose the best estimator.

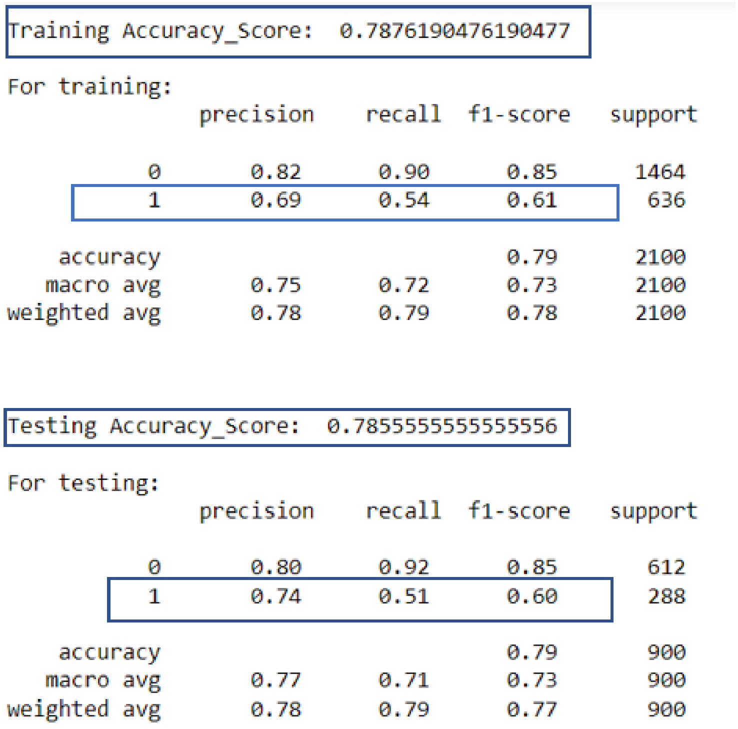
2.3 Performance Metrics: Comment and Check the performance of Predictions on Train and Test sets using Accuracy, Confusion Matrix, Plot ROC curve and get ROC_AUC score, classification reports for each model.

PERFORMANCE METRICS:

CART:

Classification report:

For Decision tree model training accuracy is 78% while for testing it is 79% so there is no drop in accuracy. On a cursory note, almost all the metrics including accuracy, precision, recall, sensitivity is more or less same for training and testing. f1-score for testing is 60% but still can be used if we have no other choice. Recall for training the positive claim is 54% whereas for 51% for testing. Precision for training the positive claim is 69% whereas for 74% for testing.



```
Training Accuracy_Score:  0.7876190476190477

For training:
              precision    recall  f1-score   support

           0       0.82      0.90      0.85      1464
           1       0.69      0.54      0.61       636

    accuracy                           0.79      2100
   macro avg       0.75      0.72      0.73      2100
weighted avg       0.78      0.79      0.78      2100


Testing Accuracy_Score:  0.7855555555555556

For testing:
              precision    recall  f1-score   support

           0       0.80      0.92      0.85       612
           1       0.74      0.51      0.60       288

    accuracy                           0.79       900
   macro avg       0.77      0.71      0.73       900
weighted avg       0.78      0.79      0.77       900
```

Confusion Matrix:

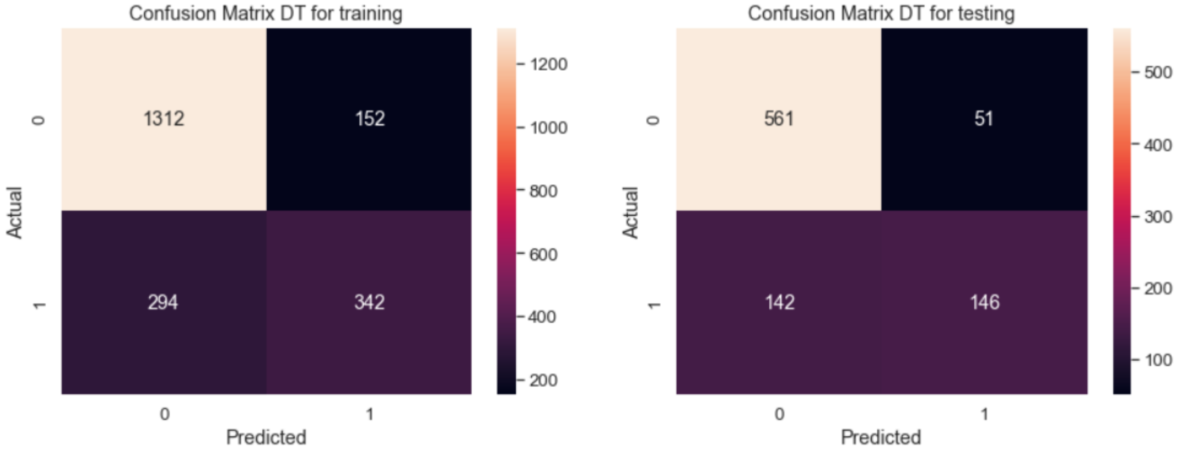


Fig29.Confusion Matrix for CART

146 true positives have been predicted while testing out of 288 actual positives. 142 positives(claims) out of 290 total actual positives this model is unable to predict them as positives, thus considered as False Negative.

ROC CURVES FOR TRAINING AND TESTING:

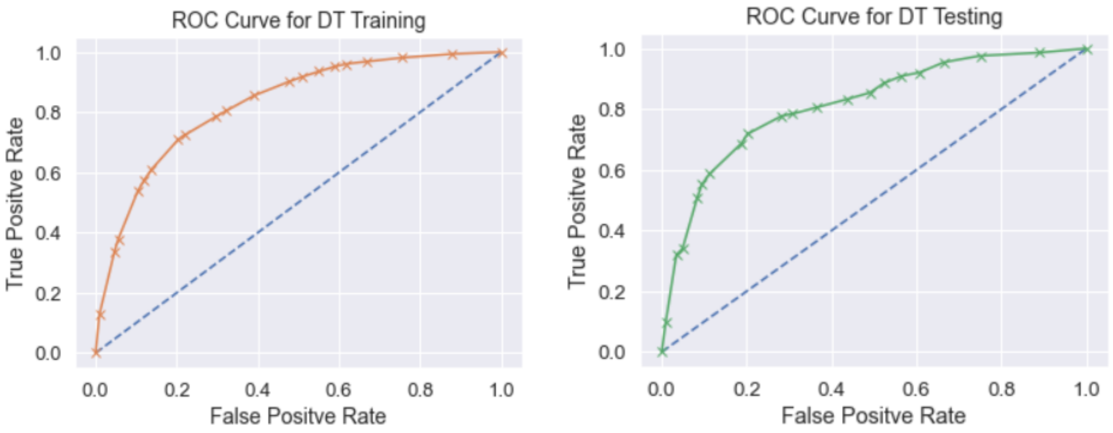


Fig30 ROC Curves for traing vs testing

In the Decision Tree Model, Area Under Curve for training is approx. 83% whereas for testing its approx. 82%, that means the model is performing perfectly as there is no significant loss or drop in AUC for training and testing. The area under curve is almost same for training as well as for testing for the Decision tree model.

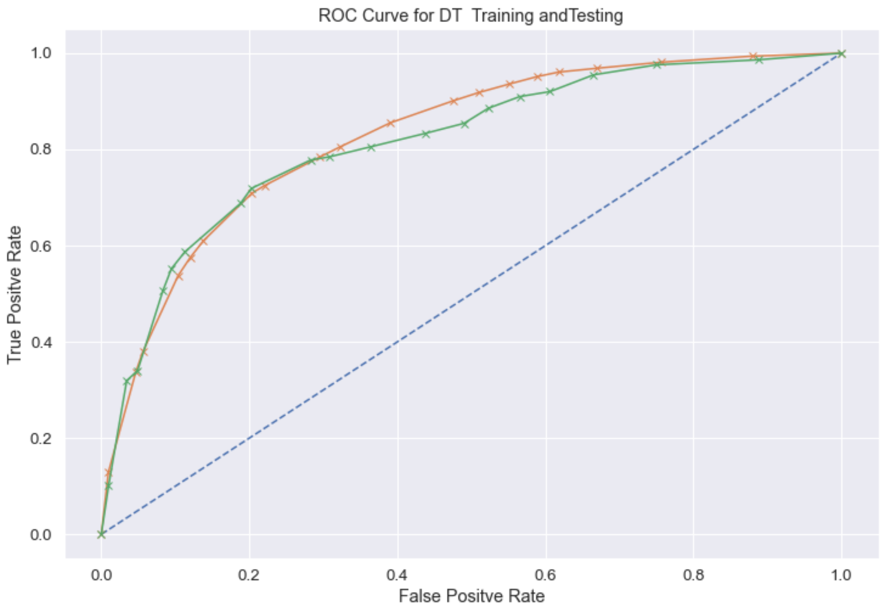
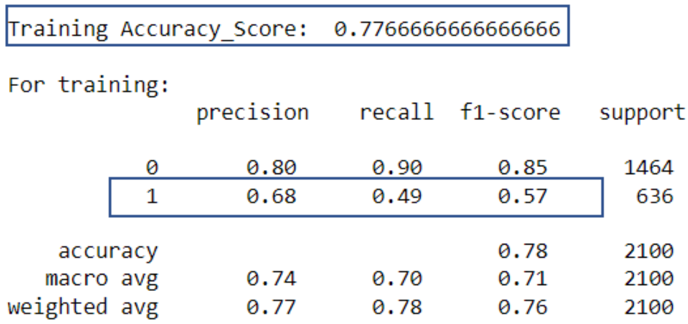


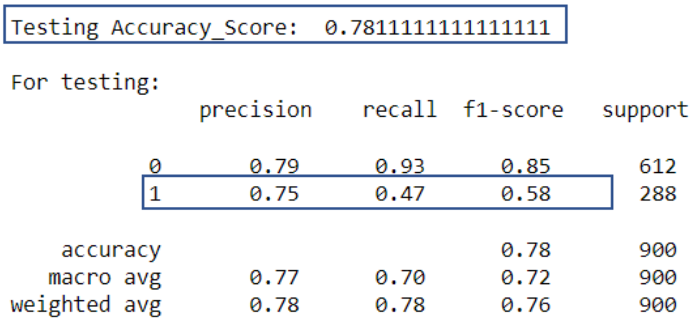
Fig31.ROC Curve for DT training & Testing

RANDOM FOREST MODEL:

Classification Report:



```
Training Accuracy_Score:  0.7766666666666666

For training:
              precision    recall  f1-score   support

           0       0.80      0.90      0.85      1464
           1       0.68      0.49      0.57       636

    accuracy                           0.78      2100
   macro avg       0.74      0.70      0.71      2100
weighted avg       0.77      0.78      0.76      2100
```



```
Testing Accuracy_Score:  0.7811111111111111

For testing:
              precision    recall  f1-score   support

           0       0.79      0.93      0.85       612
           1       0.75      0.47      0.58       288

    accuracy                           0.78       900
   macro avg       0.77      0.70      0.72       900
weighted avg       0.78      0.78      0.76       900
```

The recall for Random Forest model for testing is 53% while for training it is 52% for test. The precision is 67% for training and 73% for testing, this shows that 73% observation this model predicts are true out of all the positives predicted.

Confusion Matrix:

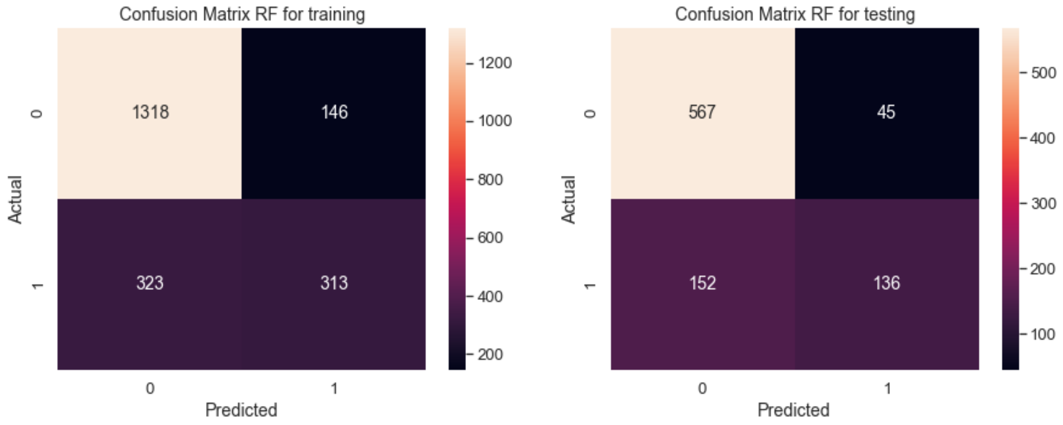
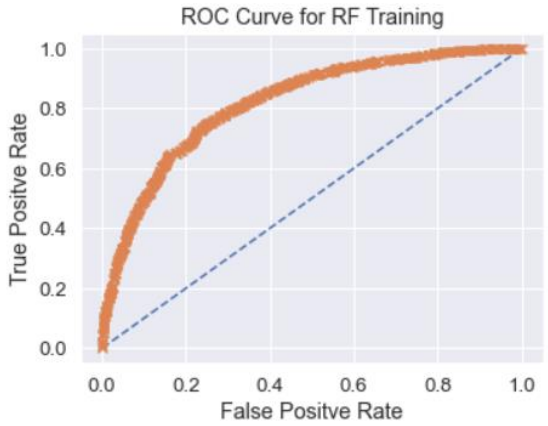


Fig32.Confusion matrix for RF

ROC CURVES FOR TRAINING AND TESTING:



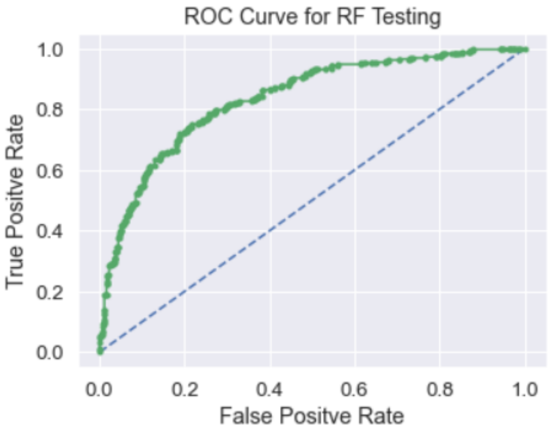


Fig33.ROC curve for RT training Vs testing

From random forest model, the roc curve shows insignificant drop in AUC score i.e., the random forest model is performing same for testing and training The AUC model for training in Random Forest model is 82% while the same for testing is 83%. The Performance of the Random Forest model while testing is even better than training. We can observe this from the plot below.

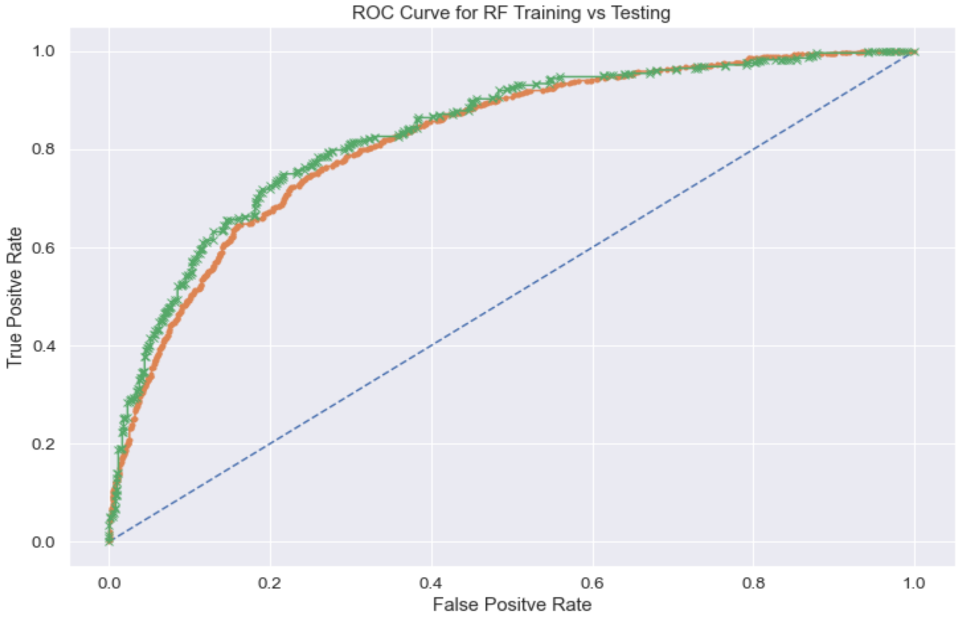
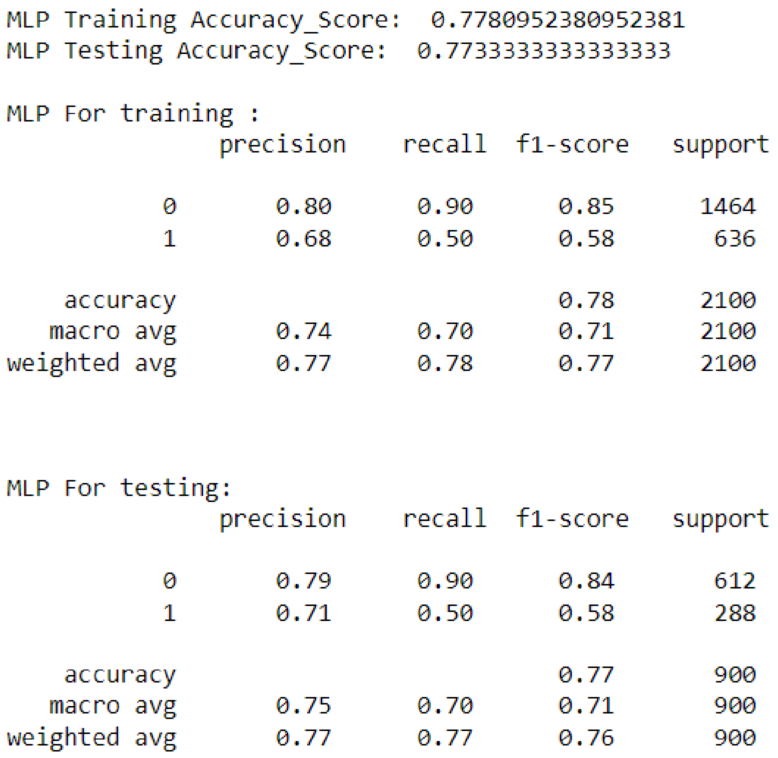


Fig34.ROC curve for RF training & testing

ANN MODEL (MLP CLASSIFIER):

Classification Report:



```
MLP Training Accuracy_Score:  0.7780952380952381
MLP Testing Accuracy_Score:  0.7733333333333333

MLP For training :
              precision    recall  f1-score   support

           0       0.80      0.90      0.85      1464
           1       0.68      0.50      0.58       636

    accuracy                           0.78      2100
   macro avg       0.74      0.70      0.71      2100
weighted avg       0.77      0.78      0.77      2100



MLP For testing:
              precision    recall  f1-score   support

           0       0.79      0.90      0.84       612
           1       0.71      0.50      0.58       288

    accuracy                           0.77       900
   macro avg       0.75      0.70      0.71       900
weighted avg       0.77      0.77      0.76       900
```

Training Precision for ANN model is 68% while the same for testing is 71%
Training recall for ANN model is 50% while the same for testing is 50%.

Confusion Matrix:

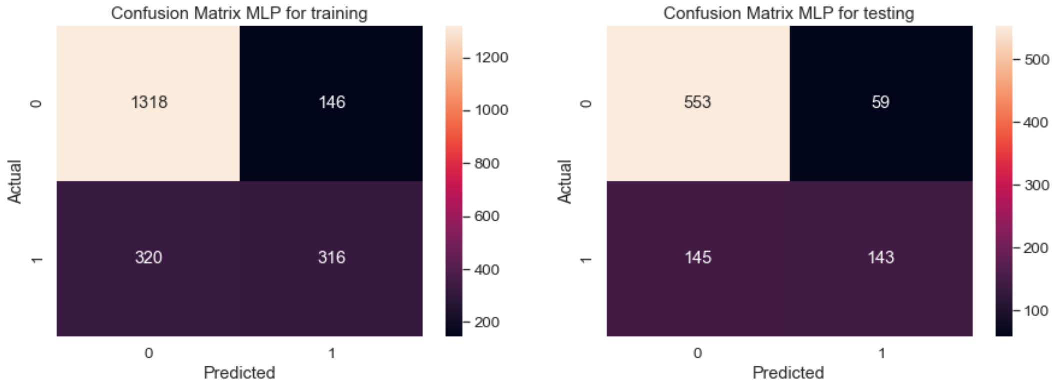


Fig 35. Confusion Matrix for MLP

143 True positives have been predicted out of 288 actual positives. The model is unable to convert 145 false negatives to true positives.

ROC CURVES FOR TRAINING AND TESTING:

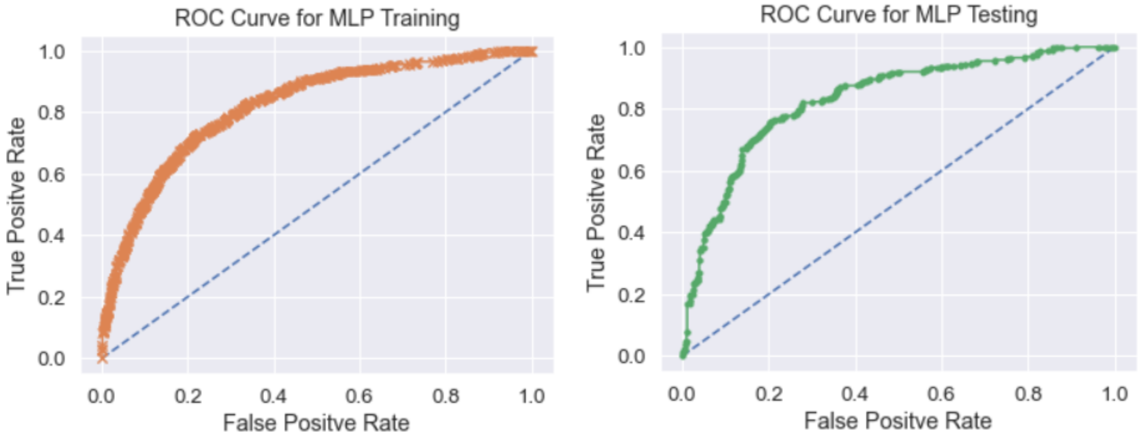


Fig36. ROC Curves for MLP traing vs testing

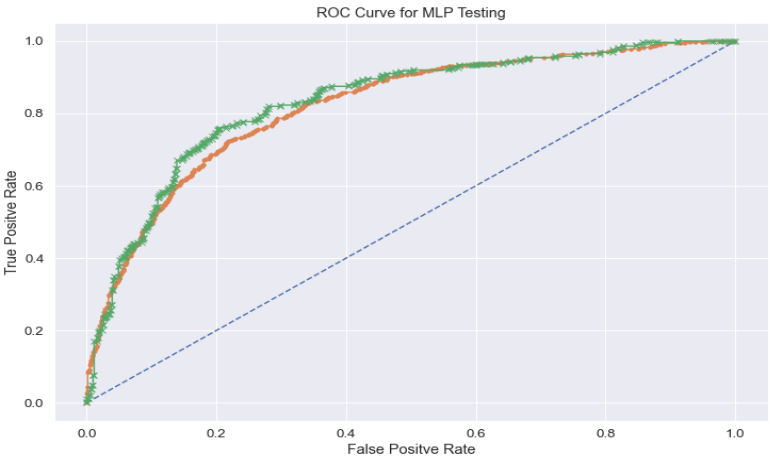


Fig37.ROC curve for MLP training & testing

→AUC score for training :0.82

→AUC score for testing: 0.83

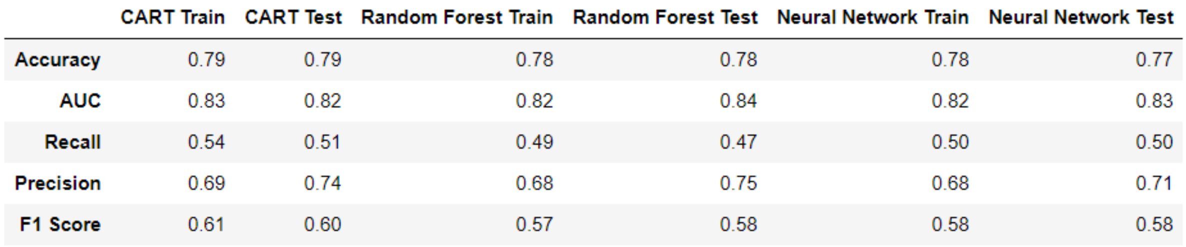
In the ANN model the area under curve for training and testing are 81% and 79%.

There is improvement in area under curve from while testing compared with training

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

The precision, recall, f1-score and AUC score for various models classified on training and testing in the table below:

COMPARISON TABLE:



	CART Train	CART Test	Random Forest Train	Random Forest Test	Neural Network Train	Neural Network Test
Accuracy	0.79	0.79	0.78	0.78	0.78	0.77
AUC	0.83	0.82	0.82	0.84	0.82	0.83
Recall	0.54	0.51	0.49	0.47	0.50	0.50
Precision	0.69	0.74	0.68	0.75	0.68	0.71
F1 Score	0.61	0.60	0.57	0.58	0.58	0.58

Table 2.6 Comparison table for 3 models

ROC CURVE:

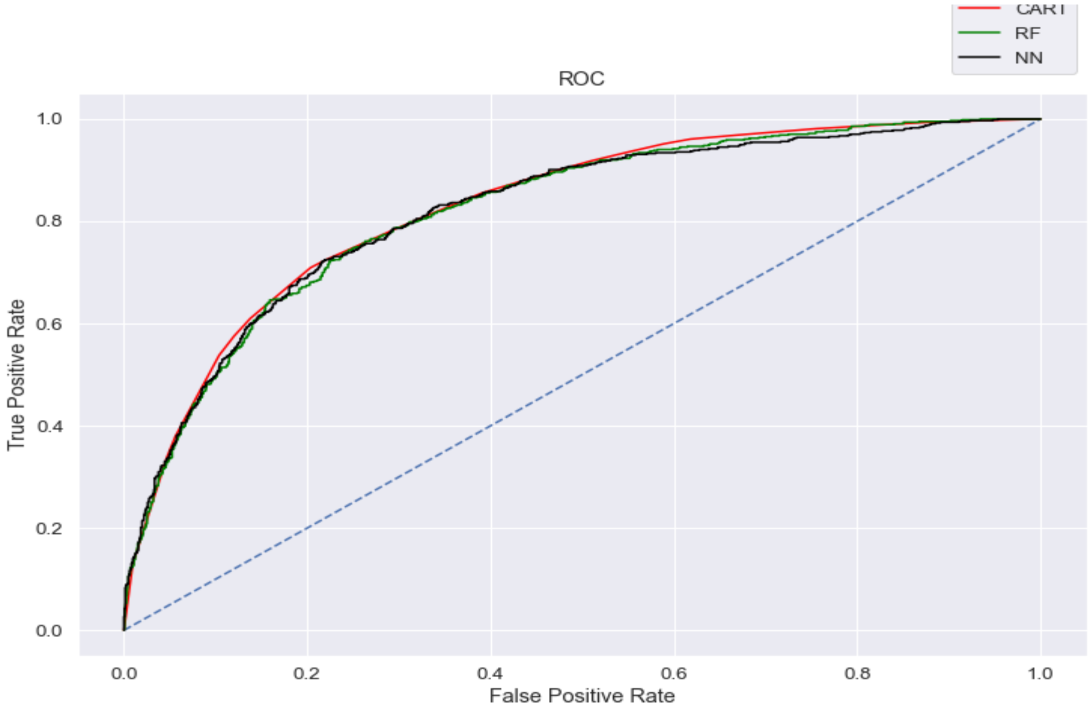


Fig38.ROC Curve for 3 models training data

Since there are very high claim frequencies faced by the company thus it becomes important for it to predict the claims that could occur based on the data we have. Two important things are recall and precision. In addition to it we will consider the AUC score as well.

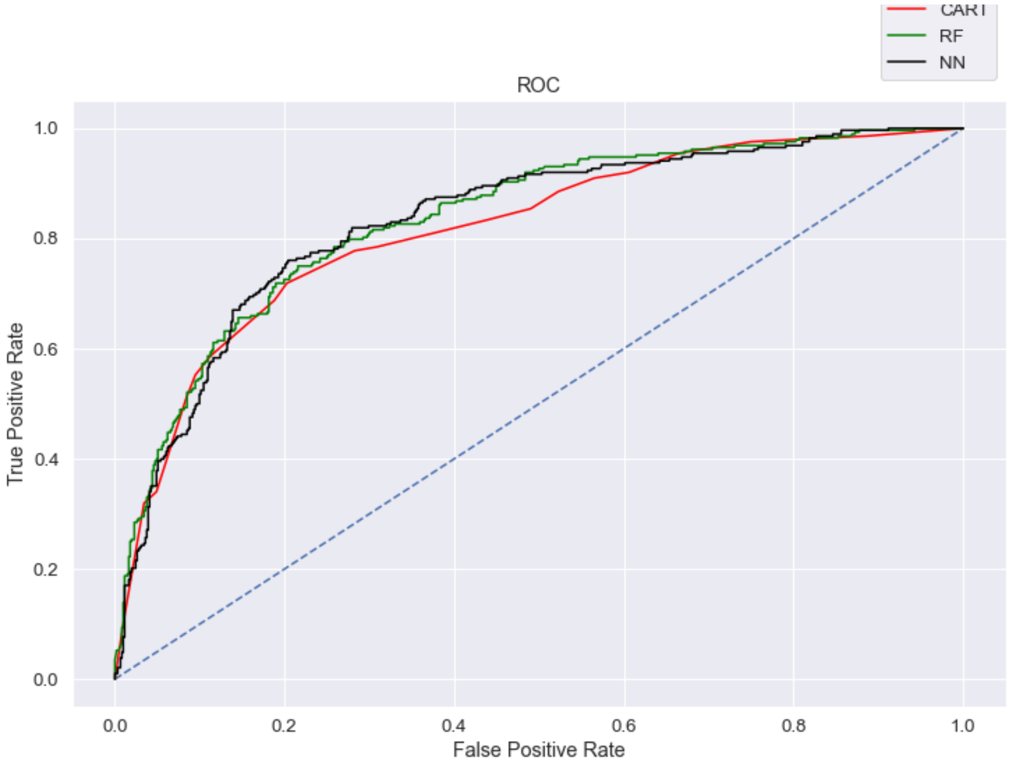


Fig39.ROC Curve for 3 models testing data

Recall- out of all actual positive claims how many positive claims does the model detects.

Precision- Out of all the positives predicted by the model how many are true positives.

f1-score - What percent of positive predictions were correct?

AUC for Random Forest model for testing is highest among all other models i.e.,0.84%.

Random Forest is the best Model as it has highest AUC and precision is also very high, 2nd highest among all other models, and it gives 152 True positives that is also highest among all other models for testing.

2.5 Inference: Based on the whole Analysis, what are the business insights and recommendations.

Insights and Recommendations:

→This is understood by looking at the insurance data by drawing insights between different variables such as day of the incident, time, age group, and associating it with other external information such as location, behaviour patterns, weather information, airline/vehicle types, etc.

• Streamlining online experiences benefitted customers, leading to an increase in conversions, which subsequently raised profits.

• As per the data 90% of insurance is done by online channel.

• Other interesting fact, is almost all the offline business has a claimed associated, need to find why?

• Need to train the JZI agency resources to pick up sales as they are in bottom, need to run promotional marketing campaign or evaluate if we need to tie up with alternate agency

• Also based on the model we are getting 80%accuracy, so we need customer books airline tickets or plans, cross sell the insurance based on the claim data pattern.

• Other interesting fact is more sales happen via Agency than Airlines and the trend shows the claim are processed more at Airline. So, we may need to deep dive into the process to understand the workflow and why?

→Key performance indicators (KPI) The KPI's of insurance claims are:

• Reduce claims cycle time • Increase customer satisfaction

• Combat fraud • Optimize claims recovery

• Reduce claim handling costs Insights gained from data and AI-powered analytics could expand the boundaries of insurability, extend existing products, and give rise to new risk transfer solutions in areas like a non-damage business interruption and reputational damage.