Business Report

SMDM Project Business Report DSBA

Sanjay Srinivasan

PGP-DSBA Online

JULY' 21 Batch

Date: 21-11-2021

INDEX

List Of Tables

List Of Figures

Problem - 1

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 of 210 customers. From this 210 customers data, we group them based on the spending's , current balance, amount paid in advance

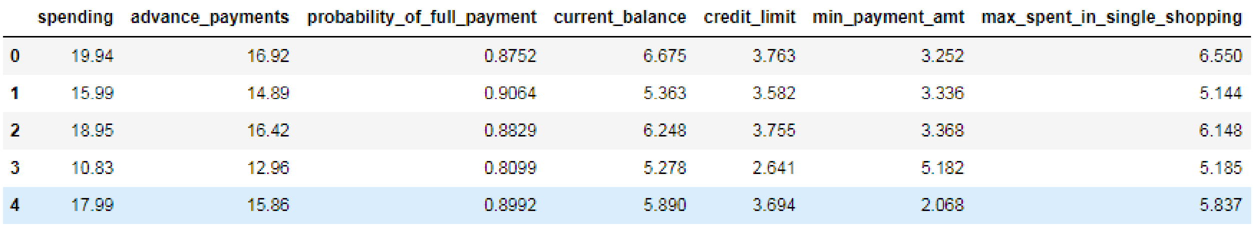
Introduction

The purpose of this exercise is to explore the dataset and make the clustering for these customers, based on the group we can categorize the promotional offer to increase their usage of credit cards.

Data Description

- Spending: Amount spent by the customer per month (in 1000s)
- Advance payments: Amount paid by the customer in advance by cash (in 100s)
- Probability of full payment: Probability of payment done in full by the customer to the bank
- Current balance: Balance amount left in the account to make purchases (in 1000s)
- Credit limit: Limit of the amount in credit card (10000s)
- Min payment amt : minimum paid by the customer while making payments for purchases made monthly (in 100s)
- Max spent in single shopping: Maximum amount spent in one purchase (in 1000s)

Sample of the dataset:

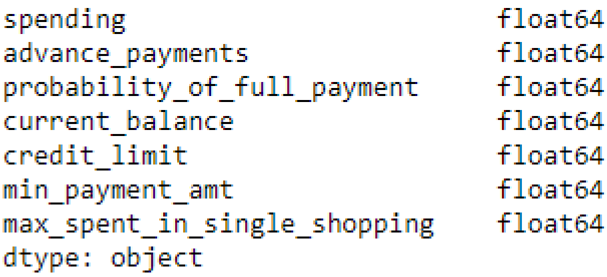


	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
0	19.94	16.92	0.8752	6.675	3.763	3.252	6.550
1	15.99	14.89	0.9064	5.363	3.582	3.336	5.144
2	18.95	16.42	0.8829	6.248	3.755	3.368	6.148
3	10.83	12.96	0.8099	5.278	2.641	5.182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

Table 1.1 Dataset Sample

Exploratory Data Analysis

Let us check the types of variables in the data frame.

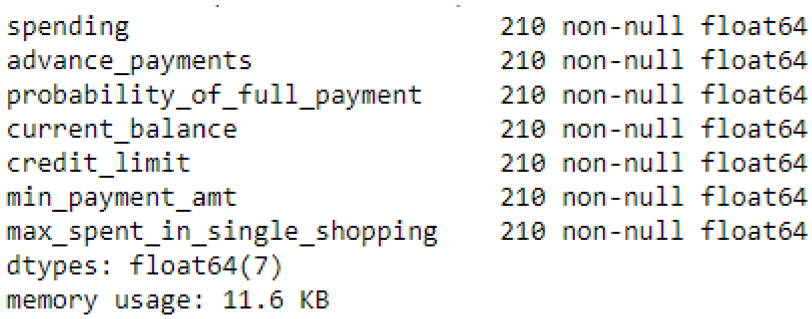


```
spending                        float64
advance_payments                float64
probability_of_full_payment     float64
current_balance                 float64
credit_limit                    float64
min_payment_amt                 float64
max_spent_in_single_shopping    float64
dtype: object
```

Table- 1.2. Datatypes of the variable

There are total 210 rows and 7 columns in the dataset. 7 columns are of float64 type

Check for missing values in the dataset:



```
spending                          210 non-null float64
advance_payments                  210 non-null float64
probability_of_full_payment       210 non-null float64
current_balance                   210 non-null float64
credit_limit                      210 non-null float64
min_payment_amt                   210 non-null float64
max_spent_in_single_shopping      210 non-null float64
dtypes: float64(7)
memory usage: 11.6 KB
```

Table- 1.3. Check null values

Problem 1A:

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

Uni-Variate Analysis:

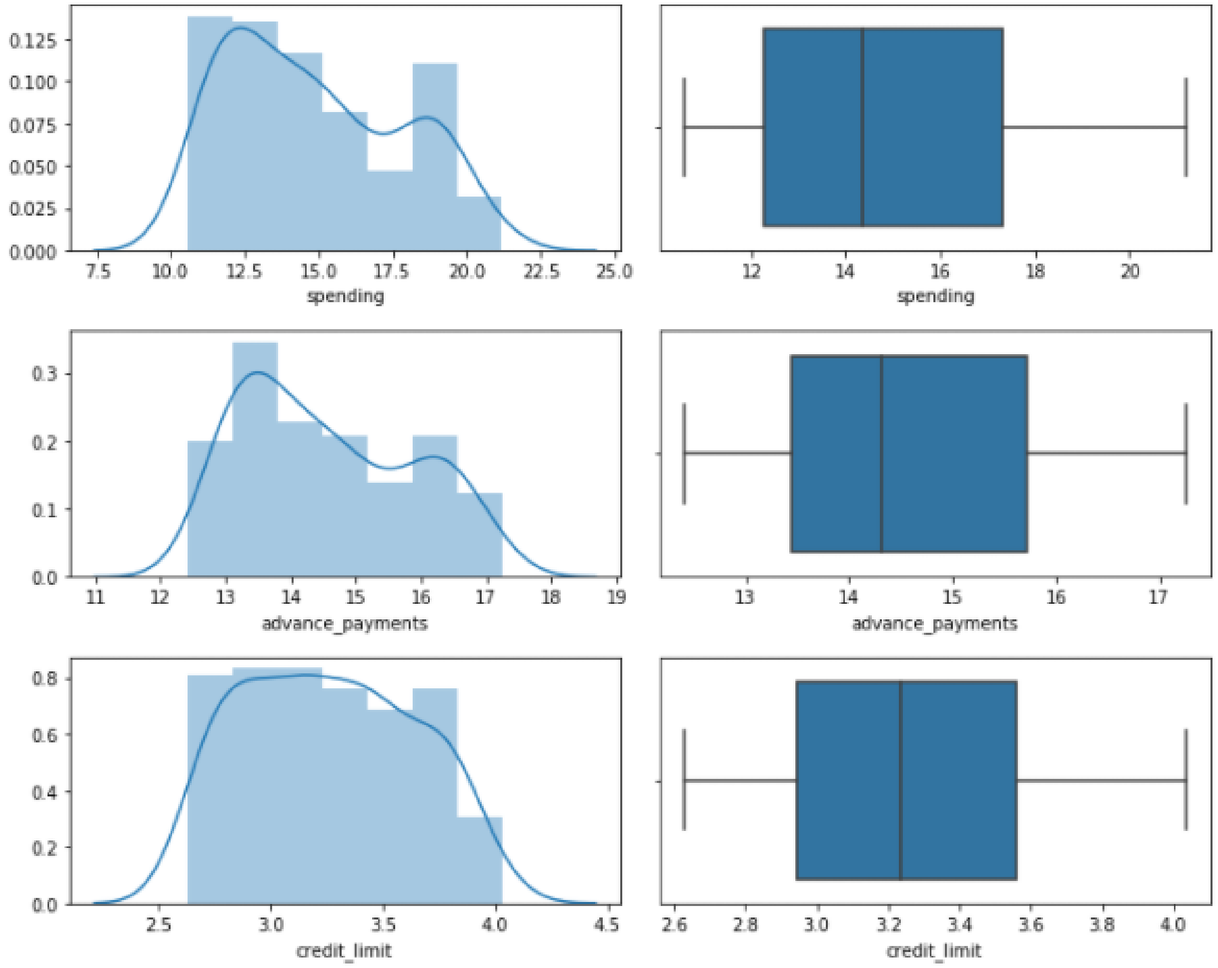


Fig – 1.1 Univariate Analysis

From the above chart (displot and boxplot), there are no outliers.

Bi – variate Analysis:

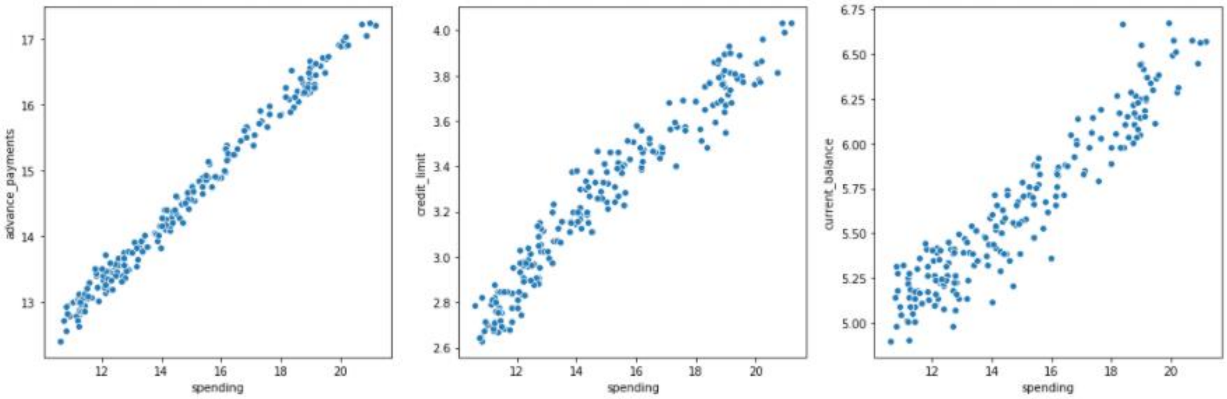


Fig – 1.2 Bivariate Analysis

From the scatterplot, we can infer that as the 'spending' increases, the 'advance_payments', 'credit_limit', 'current_balance' increases. From this, the customer has the high 'current_balance' have the high 'spending'.

Multi – variate Analysis:

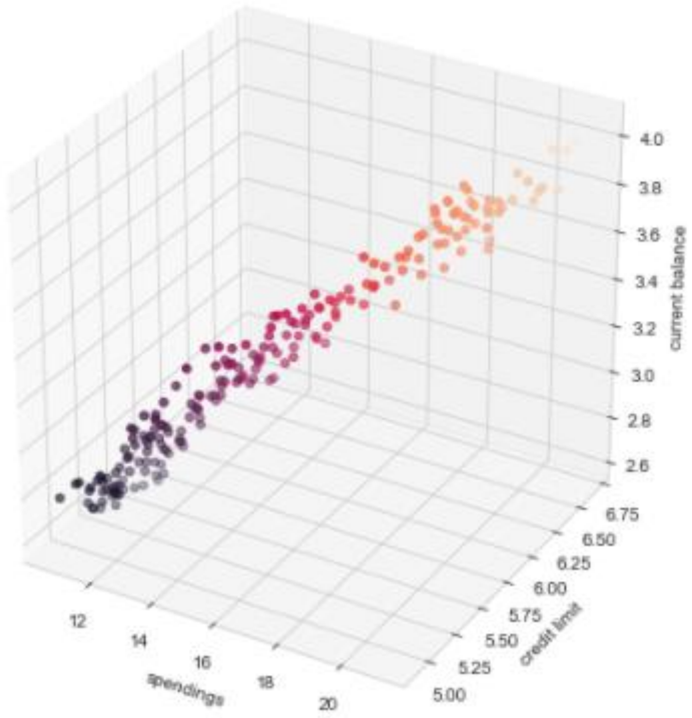


Fig – 1.3 Multivariate Analysis for spending's vs. credit limit vs. current balance

From this 3D Scatter plot, we can infer that, If the spending increases, the credit limit and current balance also increases.

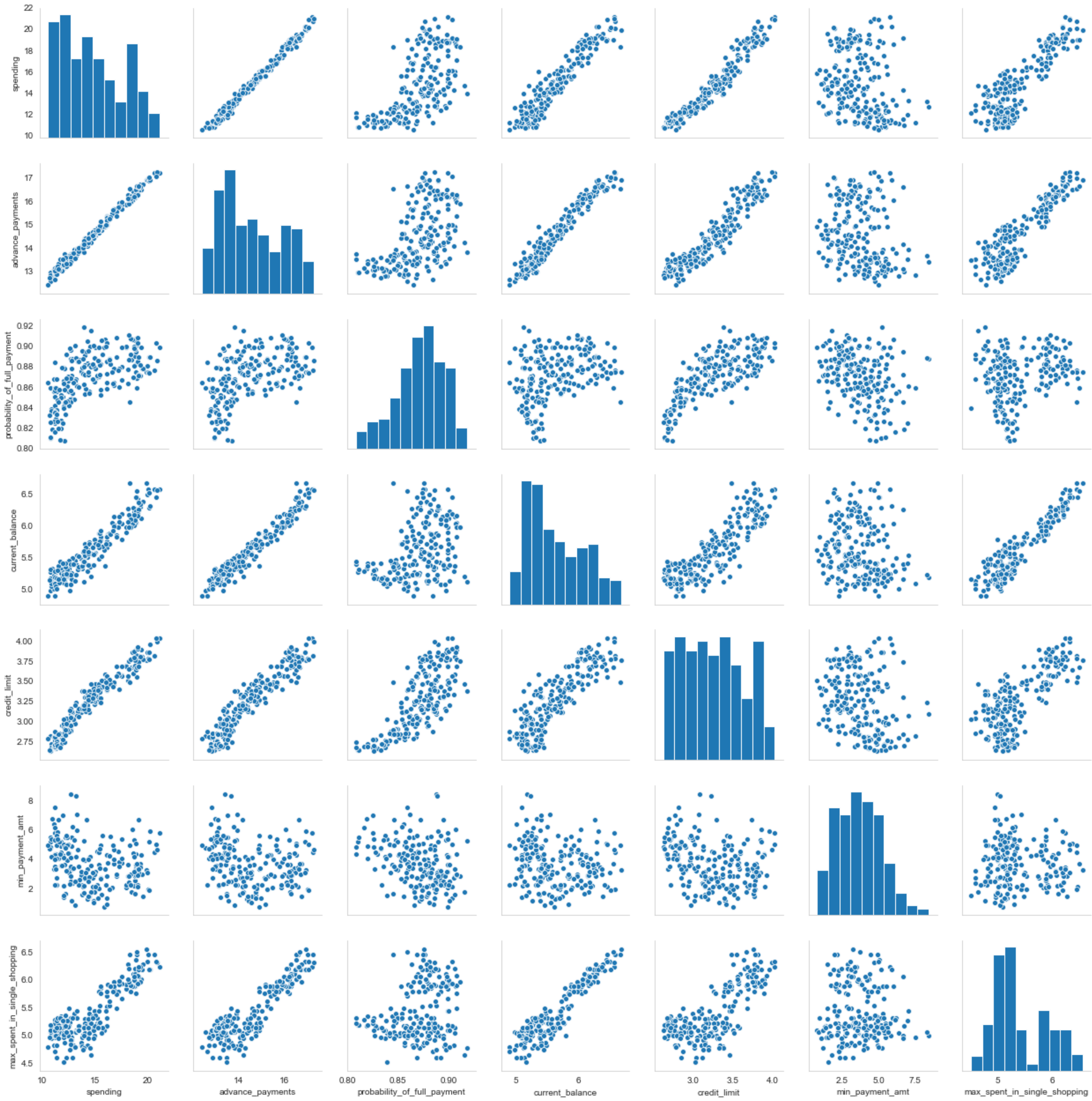


Fig – 1.4 Multivariate analysis of pairplot

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

<u>Before Scaling:</u>

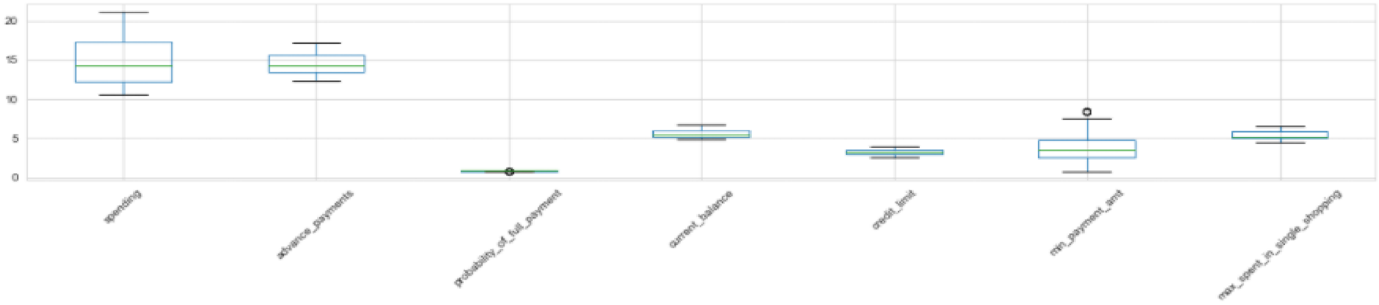


Fig – 1.5 Before Scaling

After Scaling:

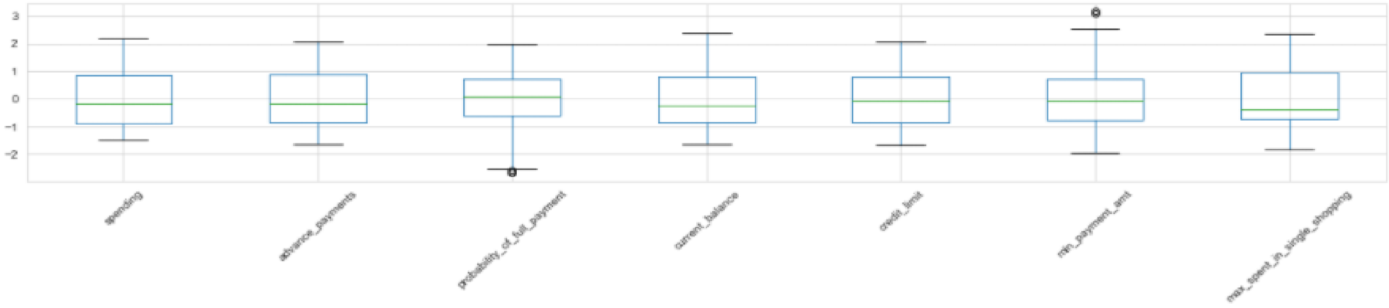


Fig – 1.6 After Scaling

Yes, Scaling needs to be done as the values of the variables are different. spending, advance_payments are in different values and this may get more weightage. The plot of the data prior and after scaling. Scaling will have all the values in the relative same range. I have used z-score to standarised the data to relative same scale -3 to +3

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

Before pruning:

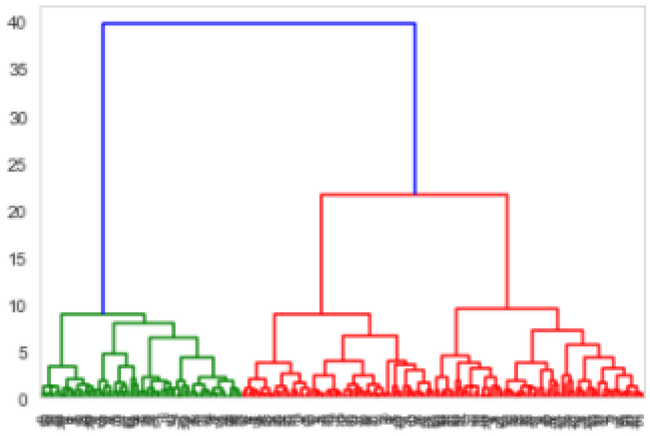


Fig – 1.7 Before pruning

From the above dendogram, we can infer that there are two clusters are present in the data.

After pruning:

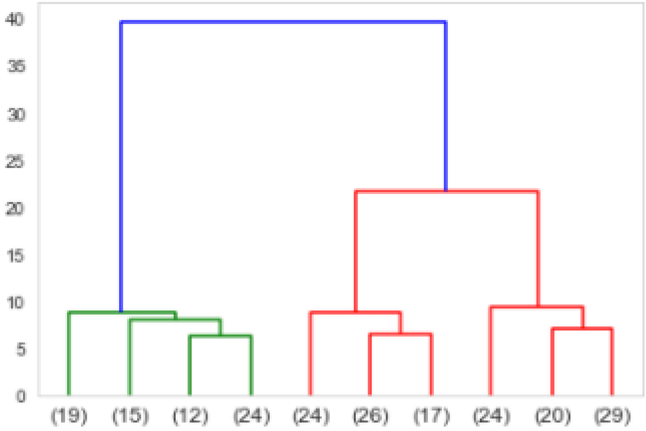
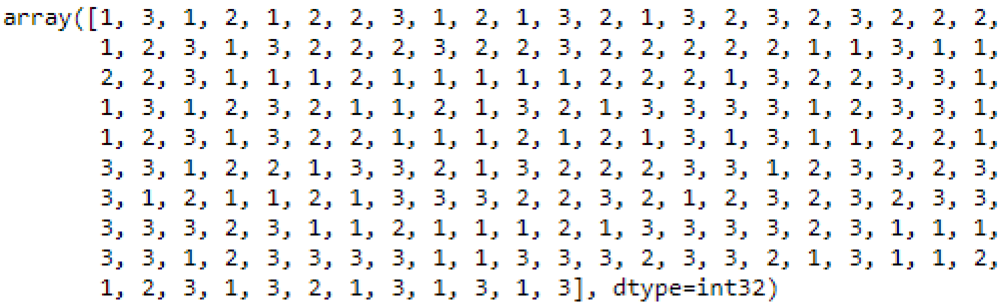


Fig – 1.8 After pruning

We are pruning the last 10 values of the dendogram.

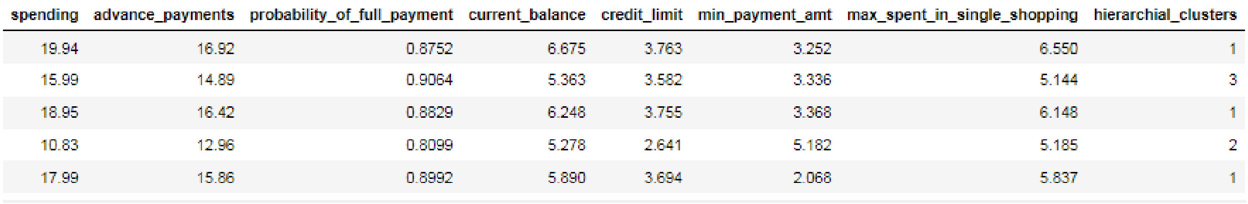


```
array([1, 3, 1, 2, 1, 2, 2, 3, 1, 2, 1, 3, 2, 1, 3, 2, 3, 2, 3, 2, 2, 2,
       1, 2, 3, 1, 3, 2, 2, 2, 3, 2, 2, 3, 2, 2, 2, 2, 2, 1, 1, 3, 1, 1,
       2, 2, 3, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 2, 2, 1, 3, 2, 2, 3, 3, 1,
       1, 3, 1, 2, 3, 2, 1, 1, 2, 1, 3, 2, 1, 3, 3, 3, 3, 1, 2, 3, 3, 1,
       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)
```

Fig – 1.9 Pruning to the last 10 values

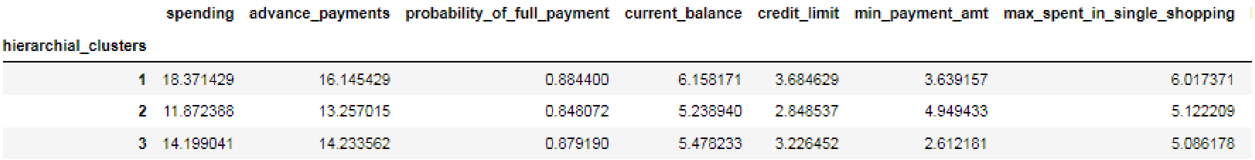
The cluster grouping linkage based on the dendrogram, 3 or 4 looks good. The further analysis, and based on the dataset had gone for 3 linkage solution based on the hierarchical clustering.

Sample data of linkage based on hierarchical clustering.



spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	hierarchial_clusters
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	3
18.95	16.42	0.8829	6.248	3.755	3.368	6.148	1
10.83	12.96	0.8099	5.278	2.641	5.182	5.185	2
17.99	15.86	0.8992	5.890	3.694	2.068	5.837	1

Table – 1.4 Sample dataframe after linkage in hierarchical clustering



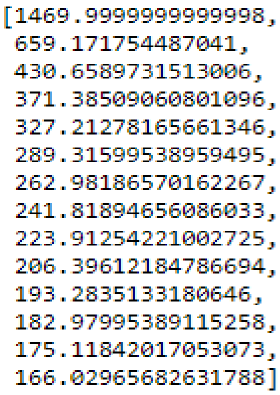
	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
hierarchial_clusters							
1	18.371429	16.145429	0.884400	6.158171	3.684629	3.639157	6.017371
2	11.872388	13.257015	0.848072	5.238940	2.848537	4.949433	5.122209
3	14.199041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

Table – 1.5 Grouping linkage based on hierarchical clustering

From this hierarchical clustering, we can infer that high Spending has been done with medium frequency. Moderate spending of customer are high with frequency when compared with other 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.

In K-means clustering, the inertia is calculated, from that finding plotting the elbow curve and finding the clusters from the elbow curve



```
[1469.9999999999998,
 659.171754487041,
 430.6589731513006,
 371.38509060801096,
 327.21278165661346,
 289.31599538959495,
 262.98186570162267,
 241.81894656086033,
 223.91254221002725,
 206.39612184786694,
 193.2835133180646,
 182.97995389115258,
 175.11842017053073,
 166.0296568263178 8]
```

Fig – 1.10 K – means inertia value

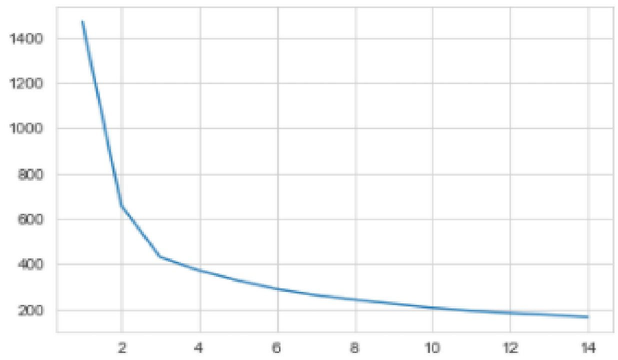
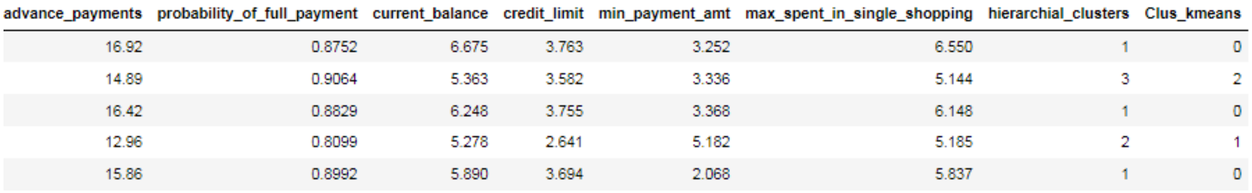


Fig – 1.11 Elbow curve for K - means

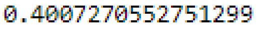
The k-means cluster linkage value is concatenated in dataframe. Sample dataframe is shown below.



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

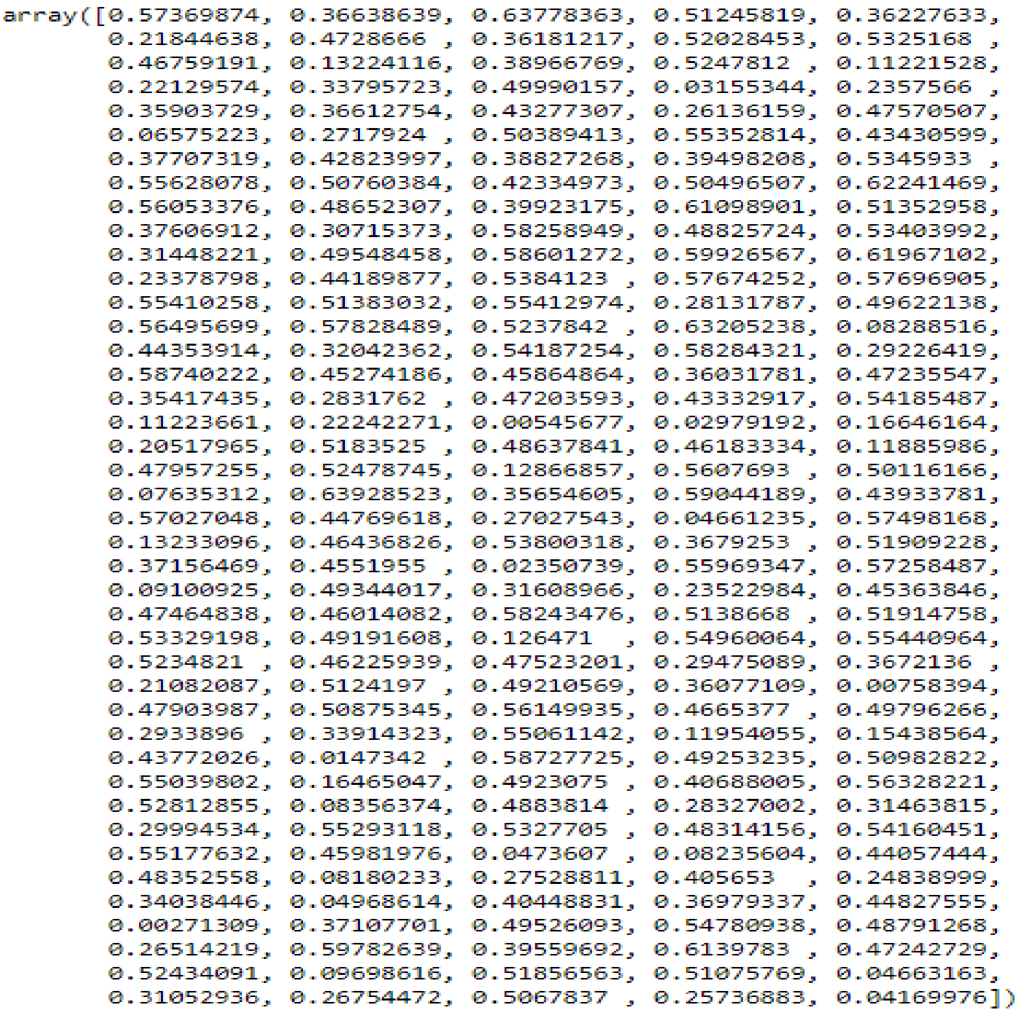
Table – 1.6 linkage based on K - means clustering

Silhouette Score and Silhouette width:



```
0.4007270552751299
```

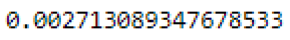
Fig –1.12 Silhouette score



```
array([0.57369874, 0.36638639, 0.63778363, 0.51245819, 0.36227633,
       0.21844638, 0.4728666 , 0.36181217, 0.52028453, 0.5325168 ,
       0.46759191, 0.13224116, 0.38966769, 0.5247812 , 0.11221528,
       0.22129574, 0.33795723, 0.49990157, 0.03155344, 0.2357566 ,
       0.35903729, 0.36612754, 0.43277307, 0.26136159, 0.47570507,
       0.06575223, 0.2717924 , 0.50389413, 0.55352814, 0.43430599,
       0.37707319, 0.42823997, 0.38827268, 0.39498208, 0.5345933 ,
       0.55628078, 0.50760384, 0.42334973, 0.50496507, 0.62241469,
       0.56053376, 0.48652307, 0.39923175, 0.61098901, 0.51352958,
       0.37606912, 0.30715373, 0.58258949, 0.48825724, 0.53403992,
       0.31448221, 0.49548458, 0.58601272, 0.59926567, 0.61967102,
       0.23378798, 0.44189877, 0.5384123 , 0.57674252, 0.57696905,
       0.55410258, 0.51383032, 0.55412974, 0.28131787, 0.49622138,
       0.56495699, 0.57828489, 0.5237842 , 0.63205238, 0.08288516,
       0.44353914, 0.32042362, 0.54187254, 0.58284321, 0.29226419,
       0.58740222, 0.45274186, 0.45864864, 0.36031781, 0.47235547,
       0.35417435, 0.2831762 , 0.47203593, 0.43332917, 0.54185487,
       0.11223661, 0.22242271, 0.00545677, 0.02979192, 0.16646164,
       0.20517965, 0.5183525 , 0.48637841, 0.46183334, 0.11885986,
       0.47957255, 0.52478745, 0.12866857, 0.5607693 , 0.50116166,
       0.07635312, 0.63928523, 0.35654605, 0.59044189, 0.43933781,
       0.57027048, 0.44769618, 0.27027543, 0.04661235, 0.57498168,
       0.13233096, 0.46436826, 0.53800318, 0.3679253 , 0.51909228,
       0.37156469, 0.4551955 , 0.02350739, 0.55969347, 0.57258487,
       0.09100925, 0.49344017, 0.31608966, 0.23522984, 0.45363846,
       0.47464838, 0.46014082, 0.58243476, 0.5138668 , 0.51914758,
       0.53329198, 0.49191608, 0.126471  , 0.54960064, 0.55440964,
       0.5234821 , 0.46225939, 0.47523201, 0.29475089, 0.3672136 ,
       0.21082087, 0.5124197 , 0.49210569, 0.36077109, 0.00758394,
       0.47903987, 0.50875345, 0.56149935, 0.4665377 , 0.49796266,
       0.2933896 , 0.33914323, 0.55061142, 0.11954055, 0.15438564,
       0.43772026, 0.0147342 , 0.58727725, 0.49253235, 0.50982822,
       0.55039802, 0.16465047, 0.4923075 , 0.40688005, 0.56328221,
       0.52812855, 0.08356374, 0.4883814 , 0.28327002, 0.31463815,
       0.29994534, 0.55293118, 0.5327705 , 0.48314156, 0.54160451,
       0.55177632, 0.45981976, 0.0473607 , 0.08235604, 0.44057444,
       0.48352558, 0.08180233, 0.27528811, 0.405653  , 0.24838999,
       0.34038446, 0.04968614, 0.40448831, 0.36979337, 0.44827555,
       0.00271309, 0.37107701, 0.49526093, 0.54780938, 0.48791268,
       0.26514219, 0.59782639, 0.39559692, 0.6139783 , 0.47242729,
       0.52434091, 0.09698616, 0.51856563, 0.51075769, 0.04663163,
       0.31052936, 0.26754472, 0.5067837 , 0.25736883, 0.04169976])
```

Fig – 1.13 Silhouette width

Silhouette sample value:

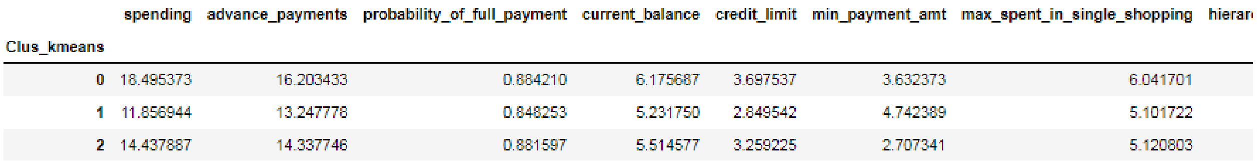


```
0.002713089347678533
```

Fig – 1.14 Silhouette Sample value

From the elbow curve and the silhouette sample output 3 is the ideal clusters for the data

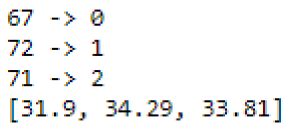
1.5 Describe cluster profiles for the clusters defined. Recommend different promotional strategies for different clusters.



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

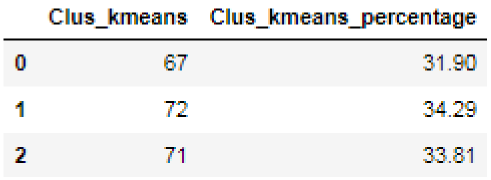
Table – 1.7Grouping linkage based on K - means clustering

Calculating the number of K-means cluster percentage and loading into the dataframe.



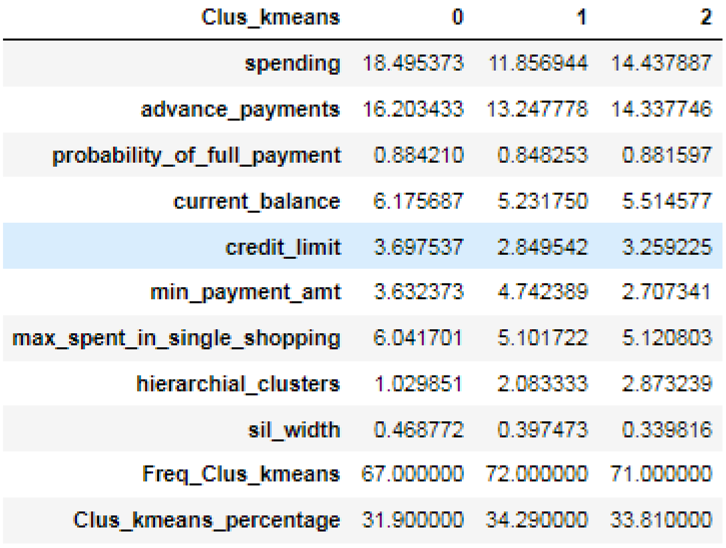
```
67 -> 0
72 -> 1
71 -> 2
[31.9, 34.29, 33.81]
```

Fig – 1.15 Count k – means linkage



	Clus_kmeans	Clus_kmeans_percentage
0	67	31.90
1	72	34.29
2	71	33.81

Table- 1.8. Count k – means linkage into dataframe



Clus_kmeans	0	1	2
spending	18.495373	11.856944	14.437887
advance_payments	16.203433	13.247778	14.337746
probability_of_full_payment	0.884210	0.848253	0.881597
current_balance	6.175687	5.231750	5.514577
credit_limit	3.697537	2.849542	3.259225
min_payment_amt	3.632373	4.742389	2.707341
max_spent_in_single_shopping	6.041701	5.101722	5.120803
hierarchial_clusters	1.029851	2.083333	2.873239
sil_width	0.468772	0.397473	0.339816
Freq_Clus_kmeans	67.000000	72.000000	71.000000
Clus_kmeans_percentage	31.900000	34.290000	33.810000

Table- 1.9. Grouping k – means linkage into dataframe

Inference from the above, Group 2 has the highest spending, whereas Group 1 has the least spending.

Promotional offer 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_maximum**” spending

Promotional offer 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.
- Increase spending habits by trying with premium Travel tickets, luxurious stay as this will encourage them to spend more

Promotional offer Low spending group:

- Customers should be given remainders for payments. Offers can be provided on early payments to improve their payment rate.

Problem – 2

Summary

The data is gathered from an Insurance firm providing tour insurance is facing higher claim frequency. The management decides to collect data from the past few years. The task is to make a model which predicts the claim status and provide recommendations to management. By using CART, RF & ANN model and comparing the models' performances by training and testing the dataset.

Introduction

The purpose of this exercise to create model using CART (Decision Tree), RF (Random Forest), ANN (Artificial Neural Networks) algorithm and predict the claim status and provide recommendations from the dataset. This dataset consist of 3000 rows and 10 columns,

Data Description

1. Claimed: Claim Status (Target)

2. Agency_Code: Code of tour firm

3. Type: Type of tour insurance firms

4. Channel: Distribution channel of tour insurance agencies

5. Product: Name of the tour insurance products

6. Duration: Duration of the tour (in days)

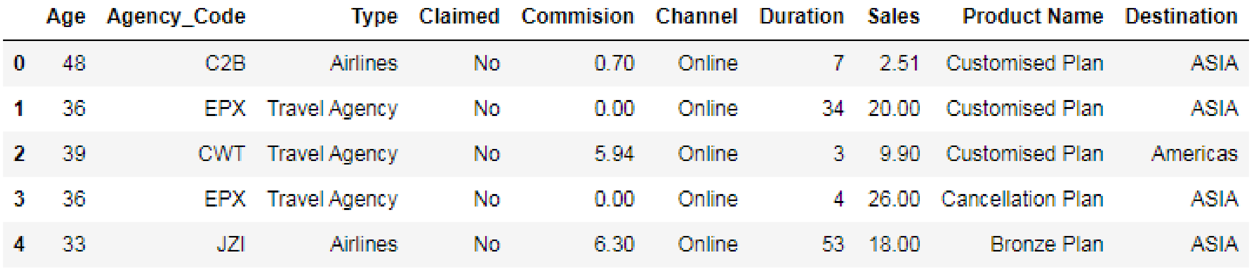
7. Destination: Destination of the tour

8. Sales: Amount worth of sales per customer in procuring tour insurance policies in rupees (in 100's)

9. Commission: The commission received for tour insurance firm (Commission is in percentage of sales)

10. Age: Age of insured

Sample of the dataset:



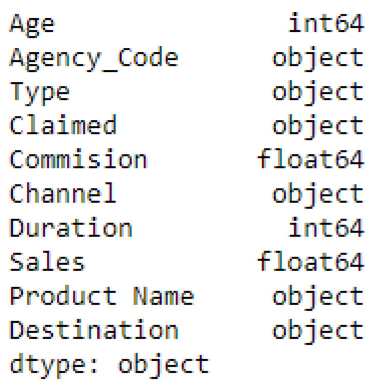
	Age	Agency_Code	Type	Claimed	Commision	Channel	Duration	Sales	Product Name	Destination
0	48	C2B	Airlines	No	0.70	Online	7	2.51	Customised Plan	ASIA
1	36	EPX	Travel Agency	No	0.00	Online	34	20.00	Customised Plan	ASIA
2	39	CWT	Travel Agency	No	5.94	Online	3	9.90	Customised Plan	Americas
3	36	EPX	Travel 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 Dataset Sample

Dataset has 10 variables with insurance details. Based on the insurance details, who have enrolled in different colleges is defined.

Exploratory Data Analysis

Let us check the types of variables in the data frame.

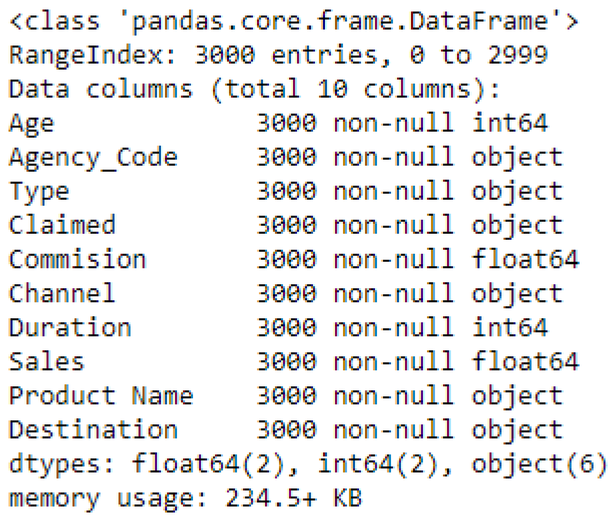


```
Age                int64
Agency_Code       object
Type              object
Claimed           object
Commision        float64
Channel           object
Duration           int64
Sales            float64
Product Name      object
Destination       object
dtype: object
```

Table 2.2 Datatypes of the variable

There are total 3000 rows and 10 columns in the dataset. Out of 10, 2 column is of integer type, 2 column is of float (Decimal value) type and rest 6 are of integer data type.

Check for missing values in the dataset:



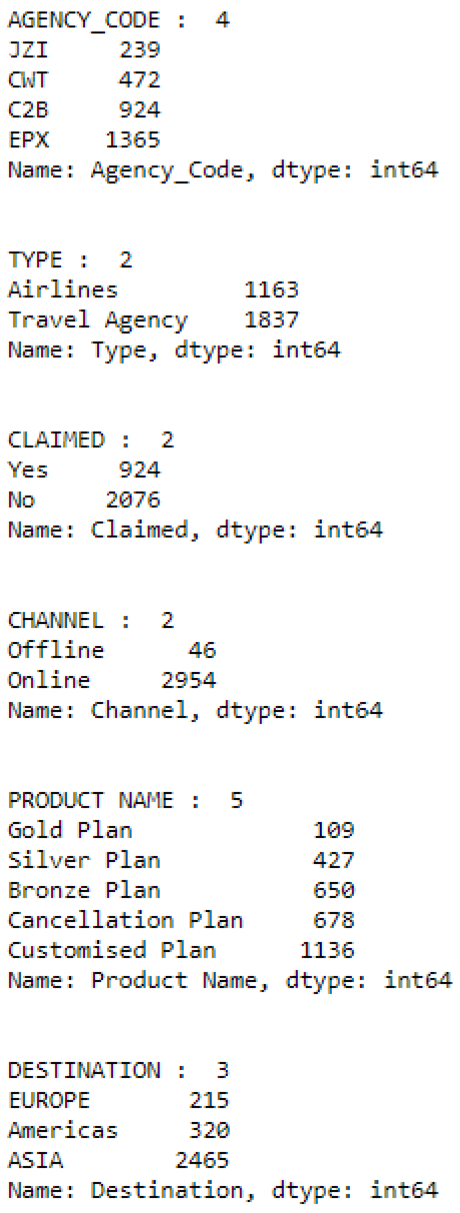
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3000 entries, 0 to 2999
Data columns (total 10 columns):
Age             3000 non-null int64
Agency_Code     3000 non-null object
Type            3000 non-null object
Claimed         3000 non-null object
Commision       3000 non-null float64
Channel         3000 non-null object
Duration        3000 non-null int64
Sales           3000 non-null float64
Product Name    3000 non-null object
Destination     3000 non-null object
dtypes: float64(2), int64(2), object(6)
memory usage: 234.5+ KB
```

Table 2.3 Check null values

From this, it is clear that there are no null values present in the dataset.

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

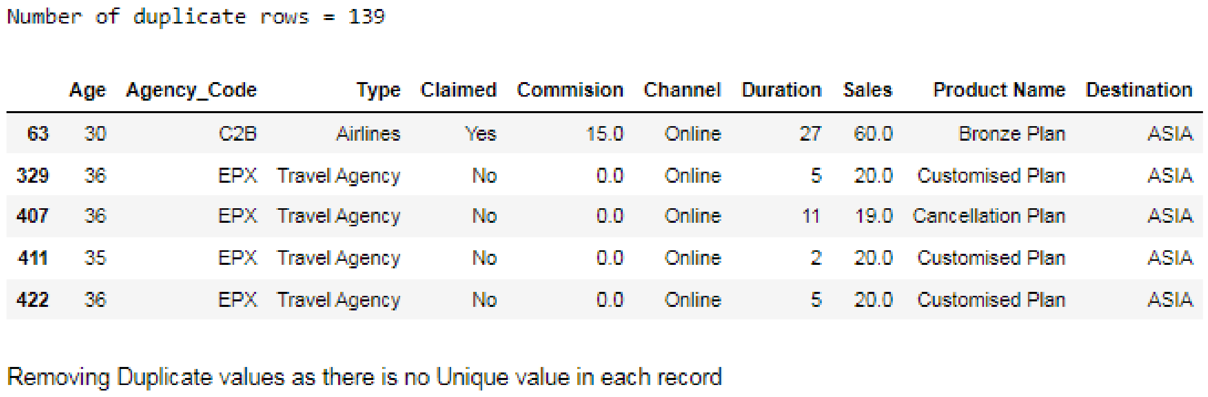
The numbers of unique variables are taken from the categorical column.



```
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
```

Fig – 2.1 Categorical value count

The numbers of duplicate values are taken from the dataset and duplicate records have been dropped from the dataset.



```
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

Removing Duplicate values as there is no Unique value in each record

Fig – 2.2 Number of duplicate rows and sample duplicate rows

Finding the outliers for the Numerical data from the dataset

<u>Before Treating Outliers:</u>

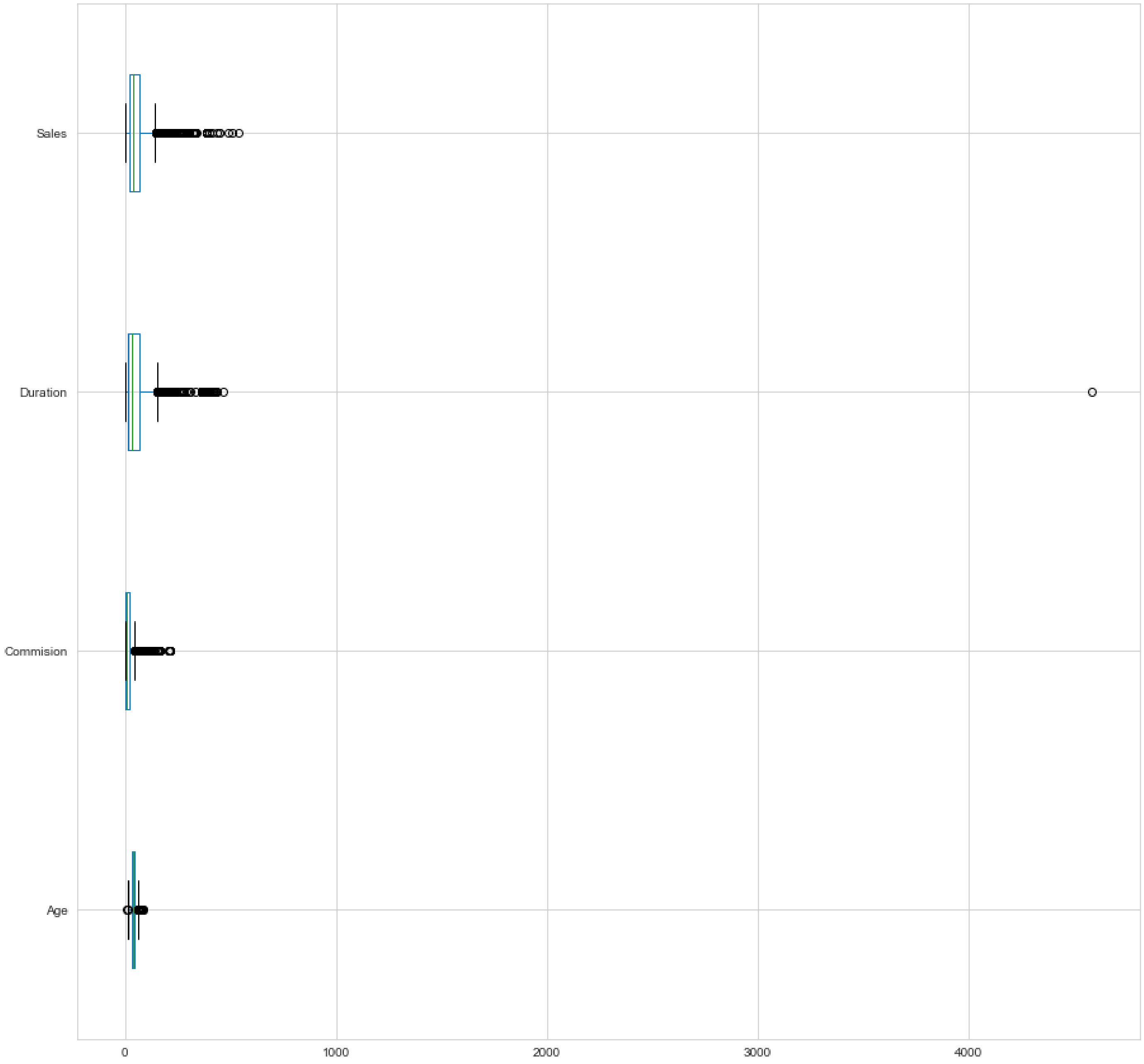


Fig – 2.3 Dataframe before treating outliers

After Treating Outliers:

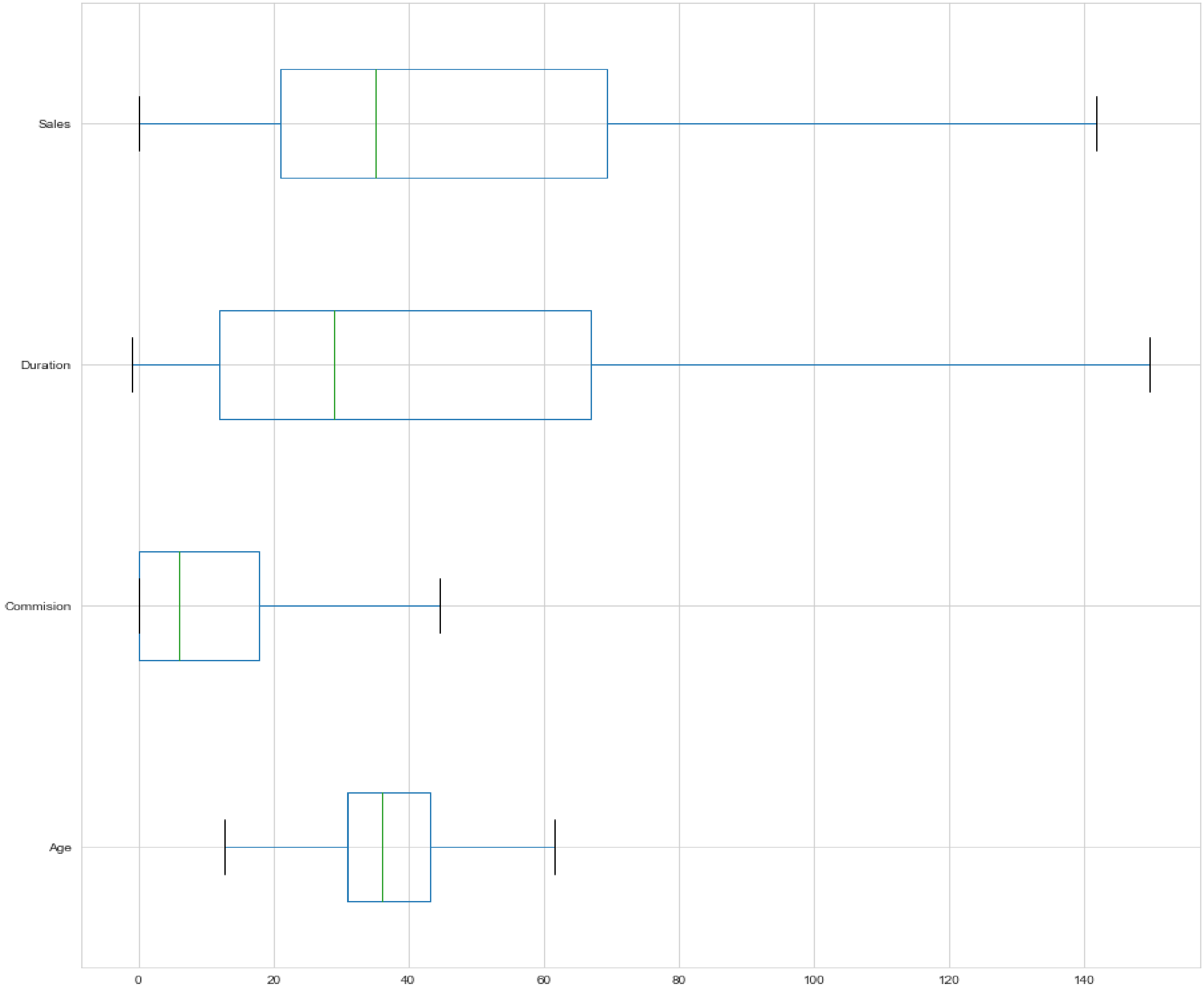


Fig – 2.4 After treating outliers

Univariate Analysis:

Univariate analysis is the simplest form of analysing data. Analysing each variable in detail.

Insights:

1163 customers have opted for Airlines and 1867 customers have opted for Airlines and Travel Agency

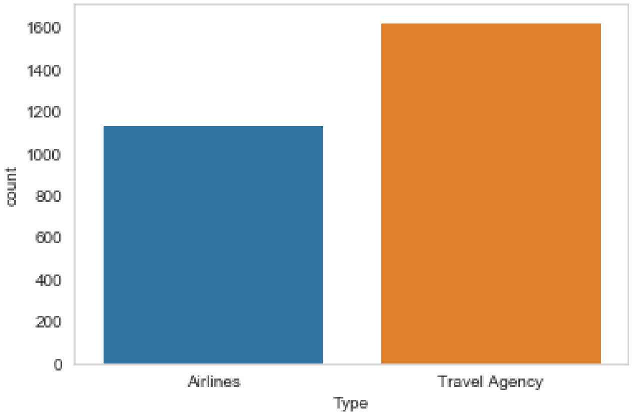


Fig – 2.5 Univariate analysis for Type

The value counts from the claimed variable are, 924 customers have claimed the insurance whereas 2076 customers have not claimed the insurance.

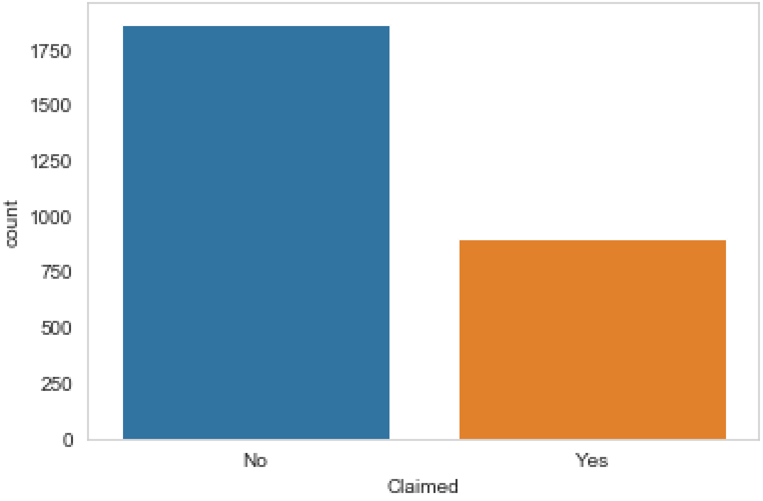


Fig – 2.6 Univariate analysis for Claimed

Bivariate Analysis:

Bivariate analysis is the simplest form of analysing data. Analysing a single variable with another variable in detail.

Insights:

The Amount of sales per customer for procuring travel insurance with the type of insurance that customer has purchased. From the below boxplot, we can infer that customers purchased through Airlines bought more travel insurance than the customers bought the travel insurance from the Travel agency.

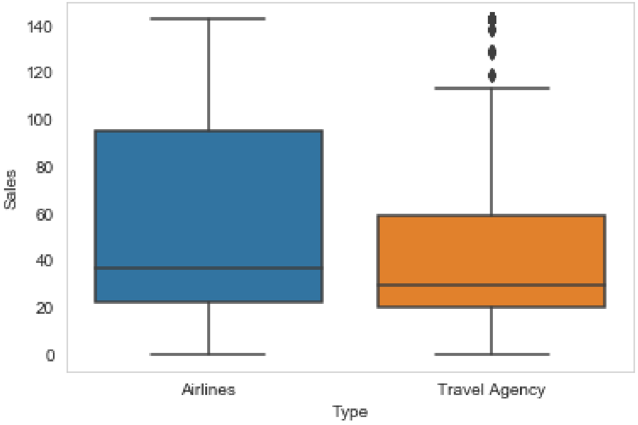


Fig – 2.7 Bivariate analysis Type vs. Sales

The customers traveling to their destination with the type of insurance that customer has purchased. From the below boxplot, we can infer that customers have purchased the travel insurance for reaching their destination. The more travel insurance is bought by the customers travelling to the America as their destination.

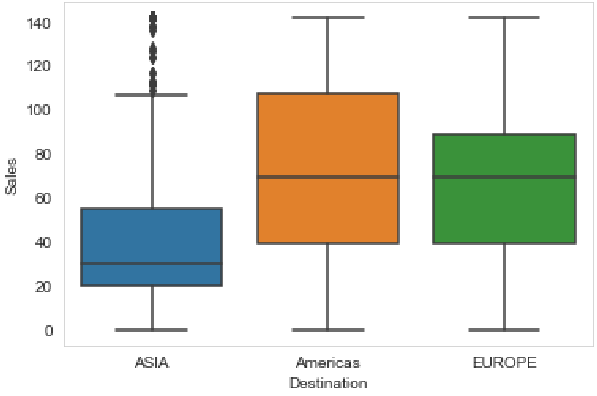


Fig – 2.8 Bivariate analysis Destination vs. Sales

Multivariate Analysis:

Analysing the data with two or more variable.

Insights:

There is a strong correlation observed between few fields.

From the pairplot and correlation, we can infer that, the Strong correlation is between the sales and commission.

And weak correlation is between the Age and Duration

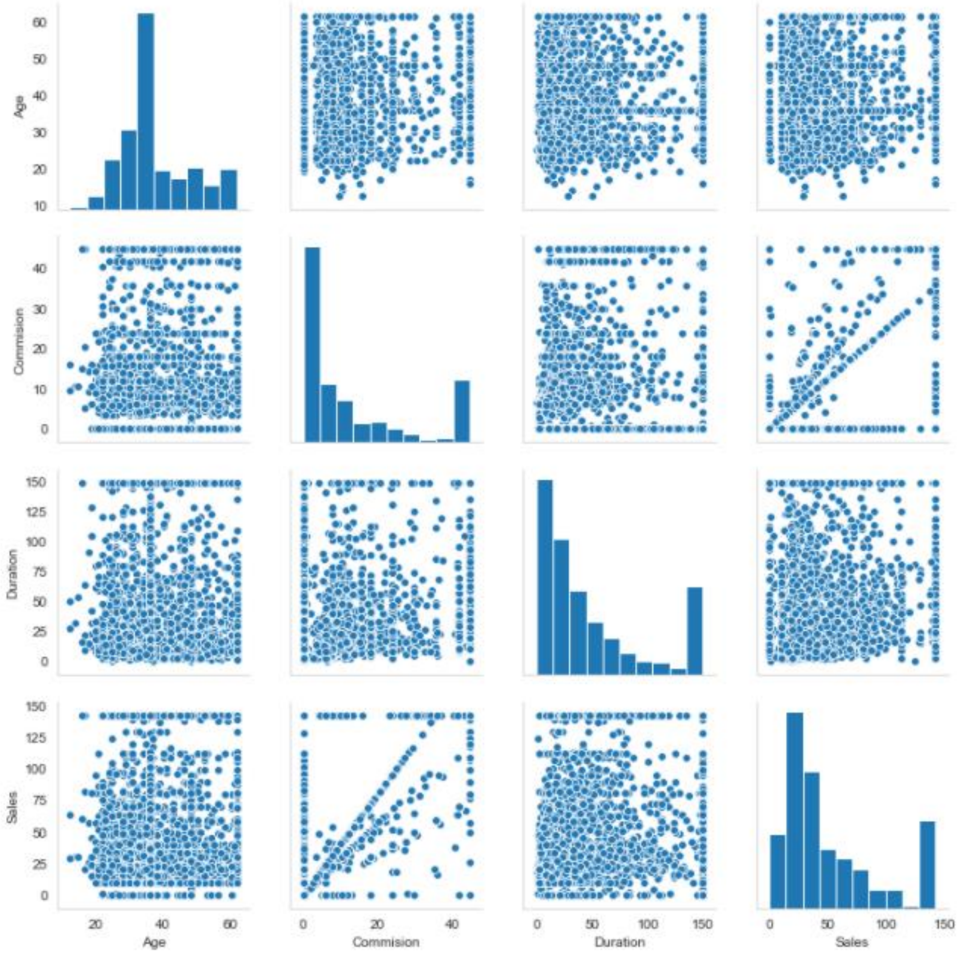


Fig – 2.9 Multivariate analysis of pairplot

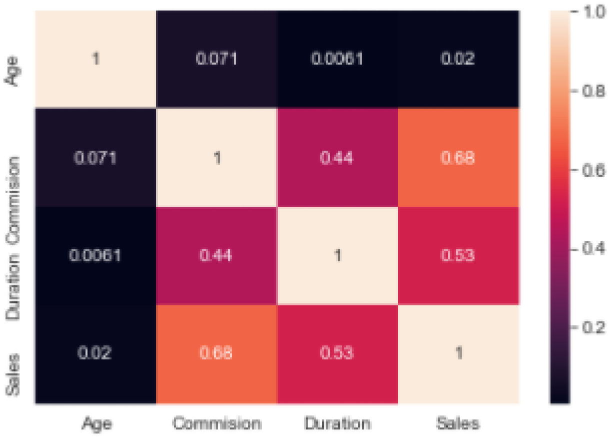
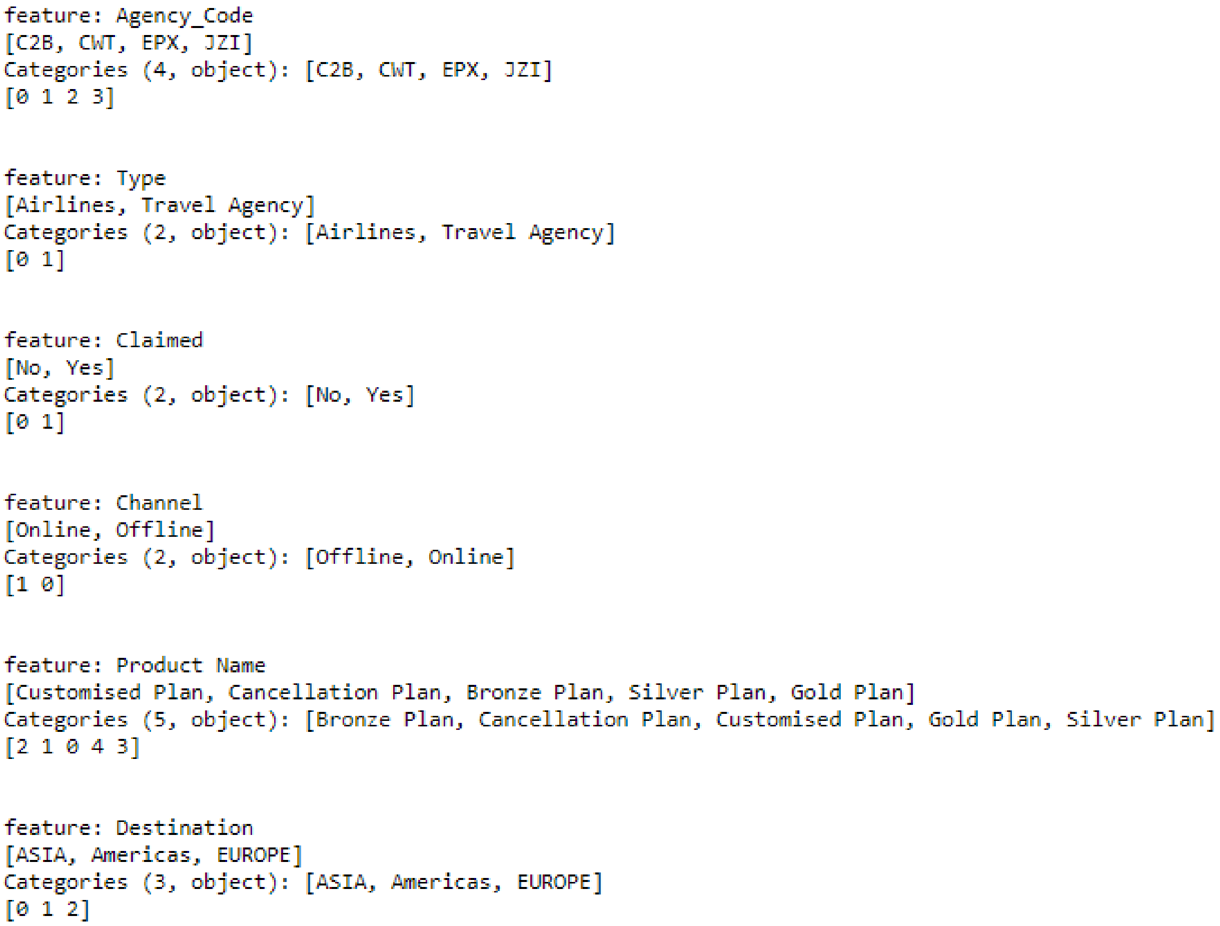


Fig – 2.10 Multivariate analysis heatmap

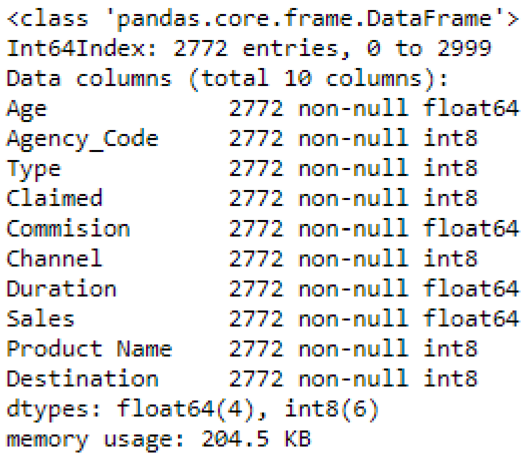
The Categorical variable is converted into the numerical variable (Integer)



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

Fig – 2.11 Categorical values to categorical codes

The dataframe is initialized after dropping the duplicate records, treating the outliers and converting the categorical value into integer value.

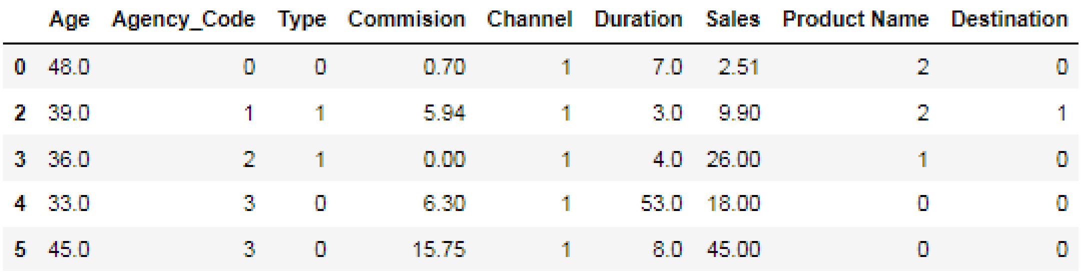


```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2772 entries, 0 to 2999
Data columns (total 10 columns):
Age             2772 non-null float64
Agency_Code     2772 non-null int8
Type            2772 non-null int8
Claimed         2772 non-null int8
Commision       2772 non-null float64
Channel         2772 non-null int8
Duration        2772 non-null float64
Sales           2772 non-null float64
Product Name    2772 non-null int8
Destination     2772 non-null int8
dtypes: float64(4), int8(6)
memory usage: 204.5 KB
```

Table – 2.4 Check data types of the dataframe after converting into integer.

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

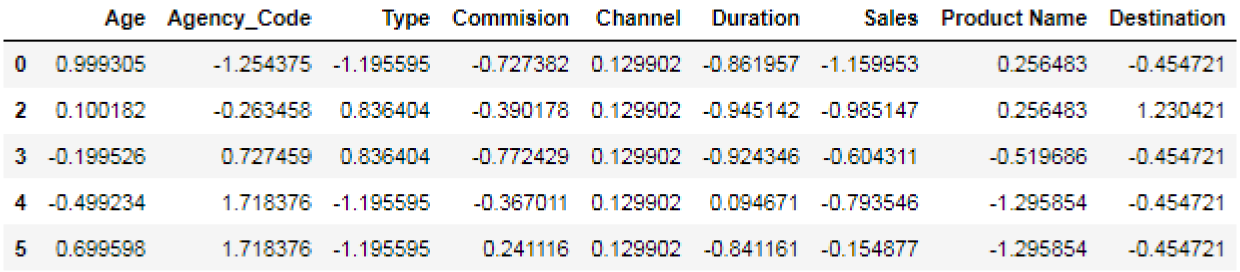
The sample dataframe is shown below, after dropping the target variable from the dataframe and before scaling is performed.



	Age	Agency_Code	Type	Commision	Channel	Duration	Sales	Product Name	Destination
0	48.0	0	0	0.70	1	7.0	2.51	2	0
2	39.0	1	1	5.94	1	3.0	9.90	2	1
3	36.0	2	1	0.00	1	4.0	26.00	1	0
4	33.0	3	0	6.30	1	53.0	18.00	0	0
5	45.0	3	0	15.75	1	8.0	45.00	0	0

Table 2.5 Dataframe for train dataset

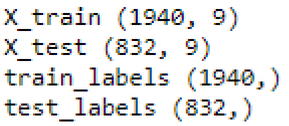
The data in the dataframe has been standardised to the same range using the standardScalar method.



	Age	Agency_Code	Type	Commision	Channel	Duration	Sales	Product Name	Destination
0	0.999305	-1.254375	-1.195595	-0.727382	0.129902	-0.861957	-1.159953	0.256483	-0.454721
2	0.100182	-0.263458	0.836404	-0.390178	0.129902	-0.945142	-0.985147	0.256483	1.230421
3	-0.199526	0.727459	0.836404	-0.772429	0.129902	-0.924346	-0.604311	-0.519686	-0.454721
4	-0.499234	1.718376	-1.195595	-0.367011	0.129902	0.094671	-0.793546	-1.295854	-0.454721
5	0.699598	1.718376	-1.195595	0.241116	0.129902	-0.841161	-0.154877	-1.295854	-0.454721

Table 2.6 Standardising the dataframe

The dataframe are split into train and test data from the dataframe. The train data has 70% of the data and test data has 30% of the data from the dataframe.

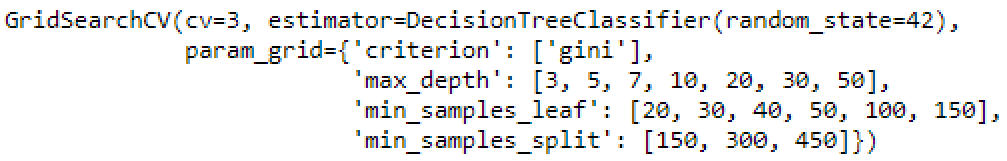


```
X_train (1940, 9)
X_test (832, 9)
train_labels (1940,)
test_labels (832,)
```

Fig – 2.12 Shape of trained and test dataframe

Decision Tree Classifier:

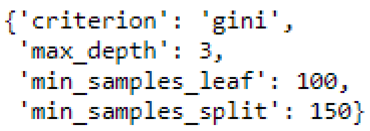
The decision tree classifier with the parameters using the grid search CV. The model is fitted into the decision tree algorithm using the grid search CV.



```
GridSearchCV(cv=3, estimator=DecisionTreeClassifier(random_state=42),
             param_grid={'criterion': ['gini'],
                         'max_depth': [3, 5, 7, 10, 20, 30, 50],
                         'min_samples_leaf': [20, 30, 40, 50, 100, 150],
                         'min_samples_split': [150, 300, 450]})
```

Fig – 2.13 Parameters for GridsearchCV in Decision Tree Classifier

The best parameters are identified from the decision tree algorithm by using the grid search CV.

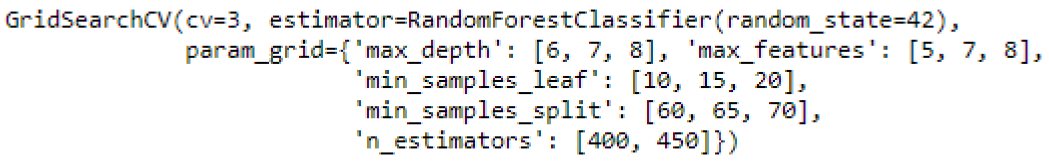


```
{'criterion': 'gini',
 'max_depth': 3,
 'min_samples_leaf': 100,
 'min_samples_split': 150}
```

Fig – 2.14 Best parameter for Decision tree classifier

Random Forest Algorithm:

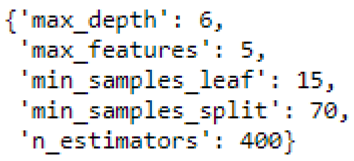
The Random forest algorithm with the parameters using the grid search CV. The model is fitted into the Random forest algorithm using the grid search CV.



```
GridSearchCV(cv=3, estimator=RandomForestClassifier(random_state=42),
             param_grid={'max_depth': [6, 7, 8], 'max_features': [5, 7, 8],
                         'min_samples_leaf': [10, 15, 20],
                         'min_samples_split': [60, 65, 70],
                         'n_estimators': [400, 450]})
```

Fig – 2.15 Parameters for GridsearchCV in Random Forest Algorithm

The best parameters are identified from the Random forest algorithm by using the grid search CV.

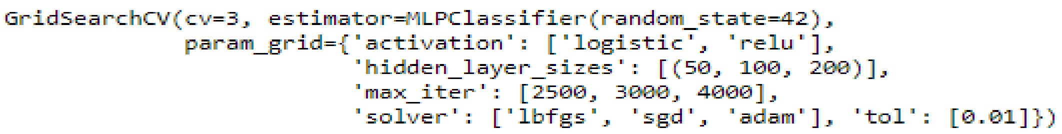


```
{'max_depth': 6,
 'max_features': 5,
 'min_samples_leaf': 15,
 'min_samples_split': 70,
 'n_estimators': 400}
```

Fig – 2.16 Best parameter for Random Forest Algorithm

Artificial Neural Networks Algorithm:

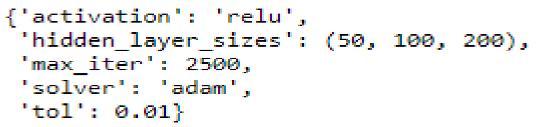
The Artificial Neural Networks algorithm with the parameters using the grid search CV. The model is fitted into the Artificial Neural Networks algorithm using the grid search CV.



```
GridSearchCV(cv=3, estimator=MLPClassifier(random_state=42),
             param_grid={'activation': ['logistic', 'relu'],
                         'hidden_layer_sizes': [(50, 100, 200)],
                         'max_iter': [2500, 3000, 4000],
                         'solver': ['lbfgs', 'sgd', 'adam'], 'tol': [0.01]})
```

Fig – 2.17 Parameters for GridsearchCV in Artificial Neural Networks Algorithm

The best parameters are identified from the Artificial Neural Networks algorithm by using the grid search CV.



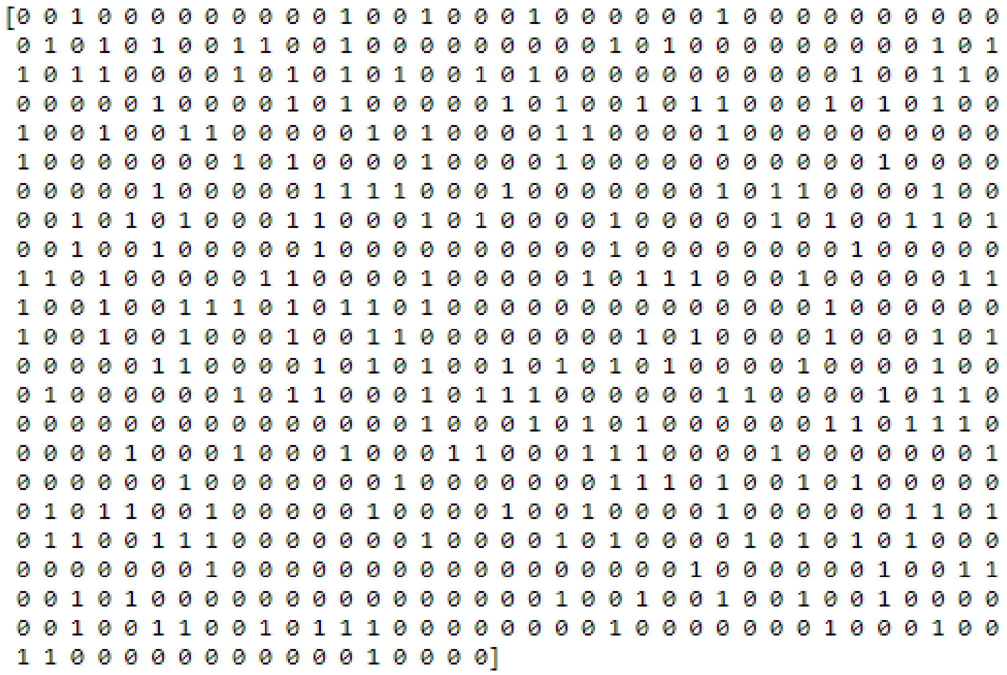
```
{'activation': 'relu',
 'hidden_layer_sizes': (50, 100, 200),
 'max_iter': 2500,
 'solver': 'adam',
 'tol': 0.01}
```

Fig – 2.18 Best parameter for Artificial Neural Networks Algorithm

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.

Decision Tree Classifier:

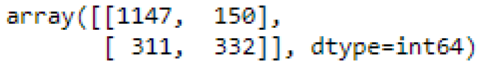
The values are predicted from the train data.



```
[0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
 0 1 0 1 0 1 0 0 1 1 0 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 1 0 1
 1 0 1 1 0 0 0 0 1 0 1 0 1 0 1 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 0
 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 0 0 0 1 0 1 0 0 1 0 1 1 0 0 0 1 0 1 0 1 0 0
 1 0 0 1 0 0 1 1 0 0 0 0 0 1 0 1 0 0 0 0 1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
 1 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0
 0 0 0 0 0 1 0 0 0 0 0 1 1 1 1 0 0 0 1 0 0 0 0 0 0 0 1 0 1 1 0 0 0 0 1 0 0
 0 0 1 0 1 0 1 0 0 0 1 1 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0 1 0 0 1 1 0 1
 0 0 1 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0
 1 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0 0 0 0 1 0 1 1 1 0 0 0 1 0 0 0 0 0 1 1
 1 0 0 1 0 0 1 1 1 0 1 0 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
 1 0 0 1 0 0 1 0 0 0 1 0 0 1 1 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 1 0 1
 0 0 0 0 0 1 1 0 0 0 0 1 0 1 0 1 0 0 1 0 1 0 1 0 1 0 0 0 0 1 0 0 0 0 1 0 0
 0 1 0 0 0 0 0 0 1 0 1 1 0 0 0 1 0 1 1 1 0 0 0 0 0 0 1 1 0 0 0 0 1 0 1 1 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 1 0 1 0 0 0 0 0 0 1 1 0 1 1 1 0
 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 1 0 0 0 1 1 1 0 0 0 0 1 0 0 0 0 0 0 0 1
 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1 1 0 1 0 0 1 0 1 0 0 0 0 0
 0 1 0 1 1 0 0 1 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 0 0 0 0 1 1 0 1
 0 1 1 0 0 1 1 1 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 0 0 1 0 1 0 1 0 1 0 0 0
 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 1 1
 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 1 0 0 1 0 0 1 0 0 0 0
 0 0 1 0 0 1 1 0 0 1 0 1 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0
 1 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]
```

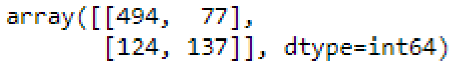
Fig – 2.19 Predicted values from the train dataset of CART model

Confusion Matrix is obtained from the train data and test data using decision tree classifier.



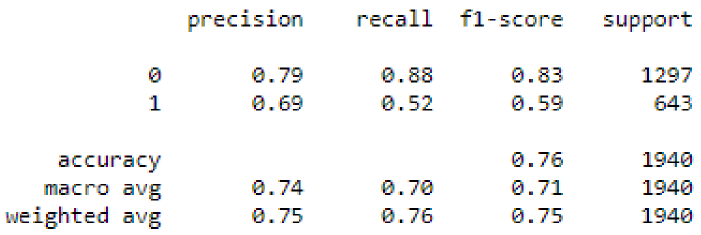
```
array([[1147,  150],
       [ 311,  332]], dtype=int64)
```

Fig 2.20 confusion matrix from Train data of CART model



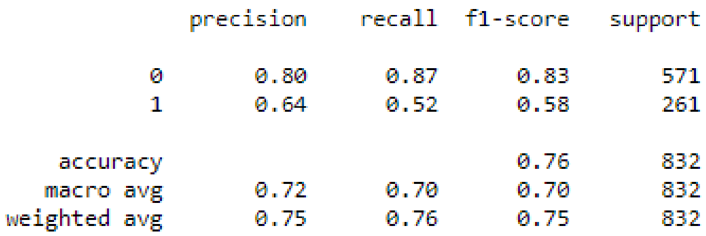
```
array([[494,  77],
       [124, 137]], dtype=int64)
```

Fig 2.21 confusion matrix from test data of CART model



```
              precision    recall  f1-score   support

           0       0.79      0.88      0.83      1297
           1       0.69      0.52      0.59       643

    accuracy                           0.76      1940
   macro avg       0.74      0.70      0.71      1940
weighted avg       0.75      0.76      0.75      1940
```

Fig 2.22 Classification Report from train data of CART model



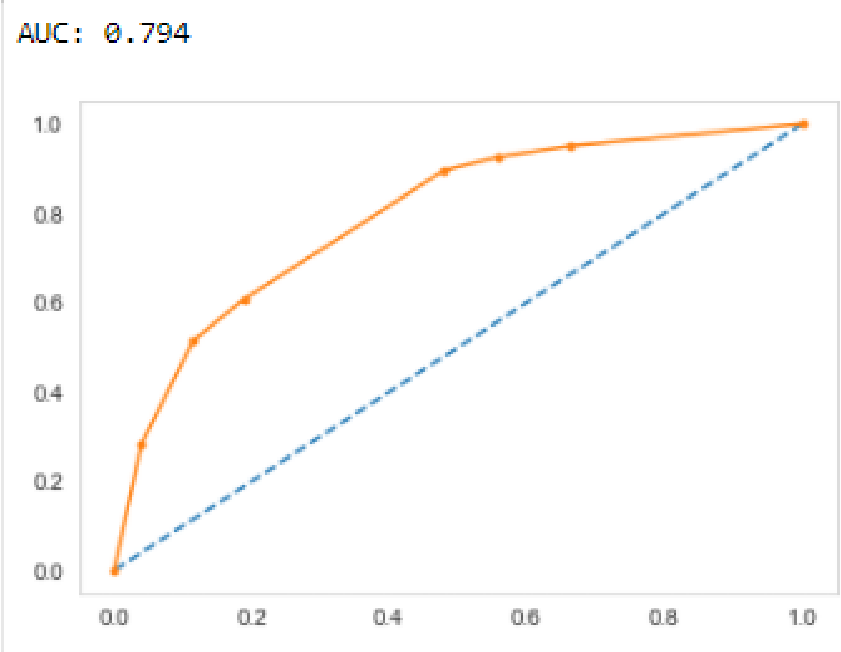
```
              precision    recall  f1-score   support

           0       0.80      0.87      0.83       571
           1       0.64      0.52      0.58       261

    accuracy                           0.76       832
   macro avg       0.72      0.70      0.70       832
weighted avg       0.75      0.76      0.75       832
```

Fig 2.23 Classification Report from test data of CART model

ROC curve (**receiver operating characteristic curve**) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters.

- True Positive Rate
- False Positive Rate

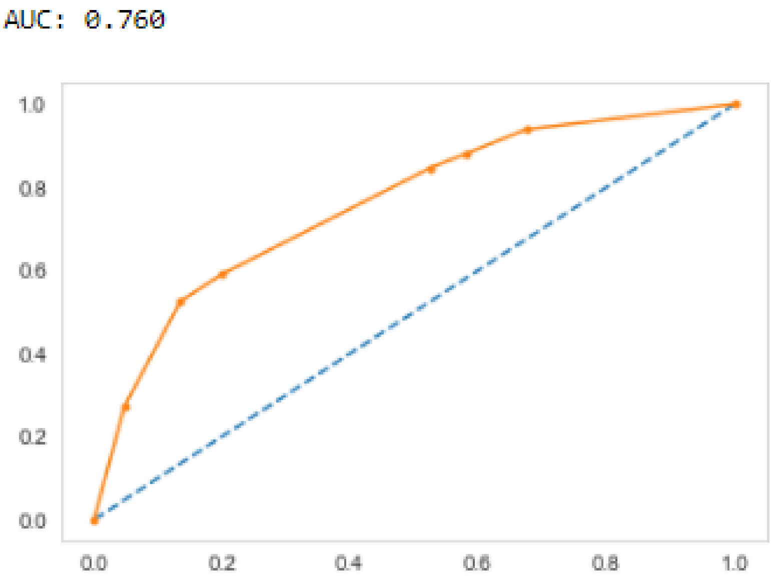
The probability of the Area under the ROC curve for the train data is 79.4%



```
AUC: 0.794
```

Fig 2.24 AUC and ROC curve train data of CART model

The probability of the Area under the ROC curve for the train data is 79.4%

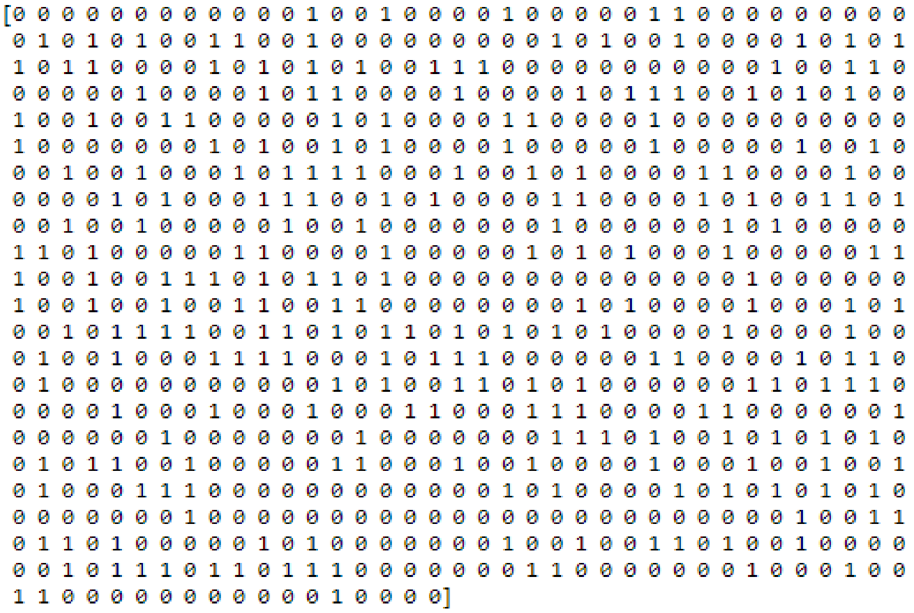


```
AUC: 0.760
```

Fig 2.25 AUC and ROC curve test data of CART model

Random Forest Algorithm:

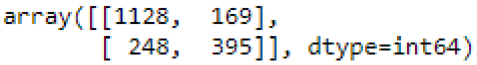
The values are predicted from the train data.



```
[0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 1 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0
 0 1 0 1 0 1 0 0 1 1 0 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 0 1 0 0 0 0 1 0 1 0 1
 1 0 1 1 0 0 0 0 1 0 1 0 1 0 1 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 0
 0 0 0 0 0 1 0 0 0 0 1 0 1 1 0 0 0 0 1 0 0 0 0 1 0 1 1 1 0 0 1 0 1 0 1 0 0
 1 0 0 1 0 0 1 1 0 0 0 0 0 1 0 1 0 0 0 0 1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
 1 0 0 0 0 0 0 0 1 0 1 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 1 0
 0 0 1 0 0 1 0 0 0 1 0 1 1 1 1 0 0 0 1 0 0 1 0 1 0 0 0 0 1 1 0 0 0 0 1 0 0
 0 0 0 0 1 0 1 0 0 0 1 1 1 0 0 1 0 1 0 0 0 0 1 1 0 0 0 0 1 0 1 0 0 1 1 0 1
 0 0 1 0 0 1 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 0 0 0
 1 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0 0 0 0 1 0 1 0 1 0 0 0 1 0 0 0 0 0 1 1
 1 0 0 1 0 0 1 1 1 0 1 0 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
 1 0 0 1 0 0 1 0 0 1 1 0 0 1 1 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 1 0 1
 0 0 1 0 1 1 1 1 0 0 1 1 0 1 0 1 1 0 1 0 1 0 1 0 1 0 0 0 0 1 0 0 0 0 1 0 0
 0 1 0 0 1 0 0 0 1 1 1 1 0 0 0 1 0 1 1 1 0 0 0 0 0 0 1 1 0 0 0 0 1 0 1 1 0
 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 1 1 0 1 0 1 0 0 0 0 0 0 1 1 0 1 1 1 0
 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 1 0 0 0 1 1 1 0 0 0 0 1 1 0 0 0 0 0 0 1
 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1 1 0 1 0 0 1 0 1 0 1 0 1 0
 0 1 0 1 1 0 0 1 0 0 0 0 0 1 1 0 0 0 1 0 0 1 0 0 0 0 1 0 0 0 1 0 0 1 0 0 1
 0 1 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 1 0 1 0 1 0 1 0
 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1
 0 1 1 0 1 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 1 0 0 1 0 0 1 1 0 1 0 0 1 0 0 0 0
 0 0 1 0 1 1 1 0 1 1 0 1 1 1 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0
 1 1 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0]
```

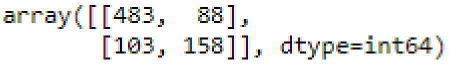
Fig – 2.26 Predicted values from the train dataset of RF model

Confusion Matrix is obtained from the train data and test data using Random Forest Algorithm.



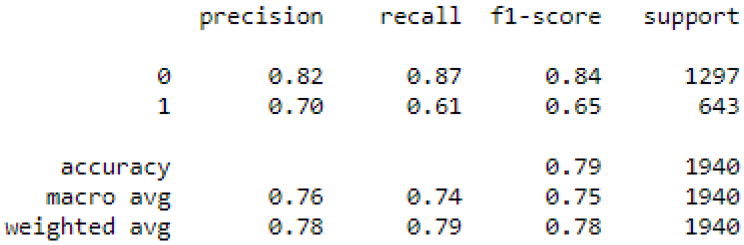
```
array([[1128,  169],
       [ 248,  395]], dtype=int64)
```

Fig 2.27 confusion matrix from Train data of RF Model



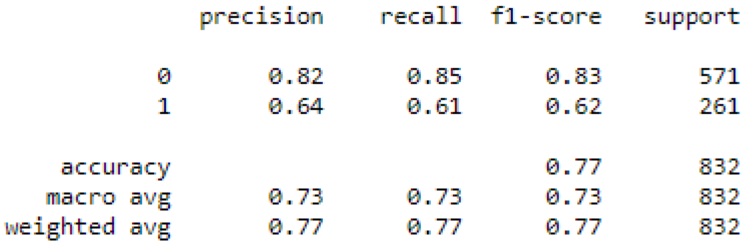
```
array([[483,  88],
       [103, 158]], dtype=int64)
```

Fig 2.28 confusion matrix from test data of RF Model



```
              precision    recall  f1-score   support

           0       0.82      0.87      0.84      1297
           1       0.70      0.61      0.65       643

    accuracy                           0.79      1940
   macro avg       0.76      0.74      0.75      1940
weighted avg       0.78      0.79      0.78      1940
```

Fig 2.29 Classification Report from train data of RF Model



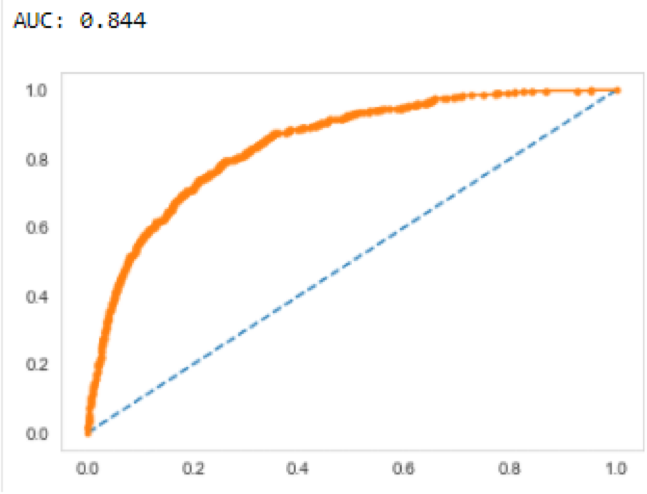
```
              precision    recall  f1-score   support

           0       0.82      0.85      0.83       571
           1       0.64      0.61      0.62       261

    accuracy                           0.77       832
   macro avg       0.73      0.73      0.73       832
weighted avg       0.77      0.77      0.77       832
```

Fig 2.30 Classification Report from test data of RF Model

ROC curve (**receiver operating characteristic curve**) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters.

- True Positive Rate
- False Positive Rate

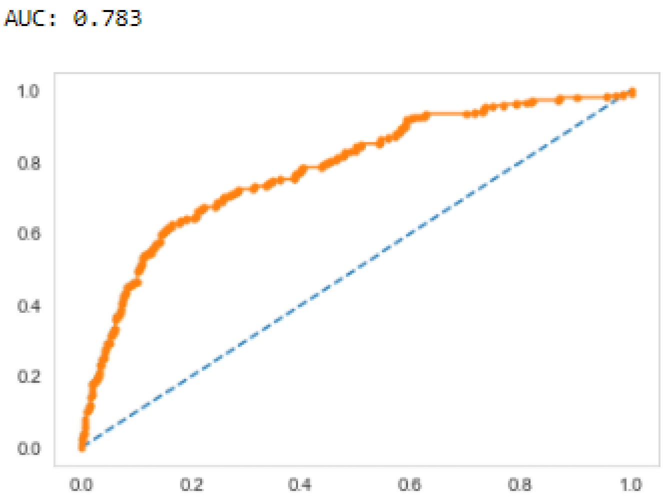
The probability of the Area under the ROC curve for the train data is 84.4%



```
AUC: 0.844
```

Fig 2.31 AUC and ROC curve train data of RF model

The probability of the Area under the ROC curve for the train data is 78.3%

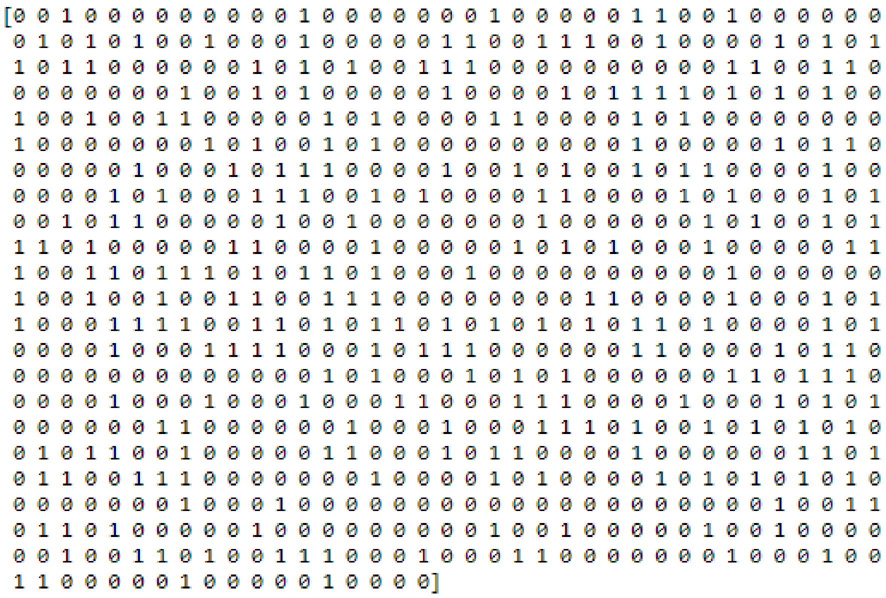


```
AUC: 0.783
```

Fig 2.32 AUC and ROC curve test data of RF model

Artificial Neural Network Algorithm:

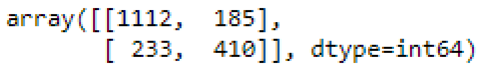
The values are predicted from the train data.



```
[0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 1 1 0 0 1 0 0 0 0 0 0
 0 1 0 1 0 1 0 0 1 0 0 0 1 0 0 0 0 0 1 1 0 0 1 1 1 0 0 1 0 0 0 0 1 0 1 0 1
 1 0 1 1 0 0 0 0 0 0 1 0 1 0 1 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0
 0 0 0 0 0 0 0 1 0 0 1 0 1 0 0 0 0 0 1 0 0 0 0 1 0 1 1 1 1 0 1 0 1 0 1 0 0
 1 0 0 1 0 0 1 1 0 0 0 0 0 1 0 1 0 0 0 0 1 1 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0
 1 0 0 0 0 0 0 0 1 0 1 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 1 1 0
 0 0 0 0 0 1 0 0 0 1 0 1 1 1 0 0 0 0 1 0 0 1 0 1 0 0 1 0 1 1 0 0 0 0 1 0 0
 0 0 0 0 1 0 1 0 0 0 1 1 1 0 0 1 0 1 0 0 0 0 1 1 0 0 0 0 1 0 1 0 0 0 1 0 1
 0 0 1 0 1 1 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 0 1 0 0 1 0 1
 1 1 0 1 0 0 0 0 0 1 1 0 0 0 0 1 0 0 0 0 0 1 0 1 0 1 0 0 0 1 0 0 0 0 0 1 1
 1 0 0 1 1 0 1 1 1 0 1 0 1 1 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
 1 0 0 1 0 0 1 0 0 1 1 0 0 1 1 1 0 0 0 0 0 0 0 0 1 1 0 0 0 0 1 0 0 0 1 0 1
 1 0 0 0 1 1 1 1 0 0 1 1 0 1 0 1 1 0 1 0 1 0 1 0 1 0 1 1 0 1 0 0 0 0 1 0 1
 0 0 0 0 1 0 0 0 1 1 1 1 0 0 0 1 0 1 1 1 0 0 0 0 0 0 1 1 0 0 0 0 1 0 1 1 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 1 0 1 0 1 0 0 0 0 0 0 1 1 0 1 1 1 0
 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 1 0 0 0 1 1 1 0 0 0 0 1 0 0 0 1 0 1 0 1
 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 1 1 0 1 0 0 1 0 1 0 1 0 1 0
 0 1 0 1 1 0 0 1 0 0 0 0 0 1 1 0 0 0 1 0 1 1 0 0 0 0 1 0 0 0 0 0 0 1 1 0 1
 0 1 1 0 0 1 1 1 0 0 0 0 0 0 0 1 0 0 0 0 1 0 1 0 0 0 0 1 0 1 0 1 0 1 0 1 0
 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1
 0 1 1 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 1 0 0 1 0 0 0 0
 0 0 1 0 0 1 1 0 1 0 0 1 1 1 0 0 0 1 0 0 0 1 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0
 1 1 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0]
```

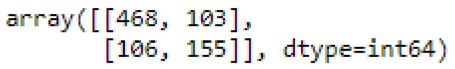
Fig – 2.33 Predicted values from the train dataset of ANN model

Confusion Matrix is obtained from the train data and test data using Artificial Neural Network Algorithm.



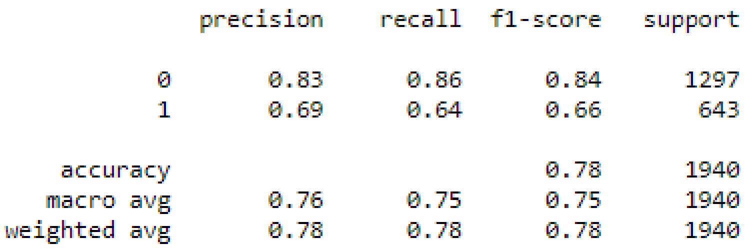
```
array([[1112,  185],
       [ 233,  410]], dtype=int64)
```

Fig 2.34 confusion matrix from Train data of ANN Model



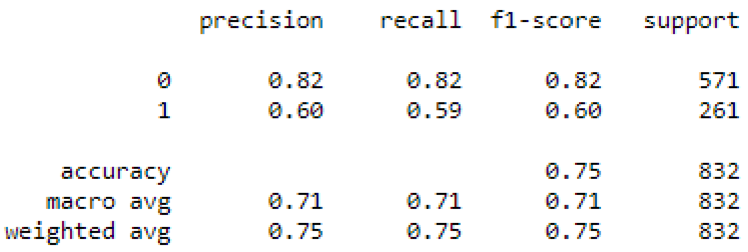
```
array([[468, 103],
       [106, 155]], dtype=int64)
```

Fig 2.35 confusion matrix from Test data of ANN Model



```
              precision    recall  f1-score   support

           0       0.83      0.86      0.84      1297
           1       0.69      0.64      0.66       643

    accuracy                           0.78      1940
   macro avg       0.76      0.75      0.75      1940
weighted avg       0.78      0.78      0.78      1940
```

Fig 2.36 Classification Report from train data of ANN model



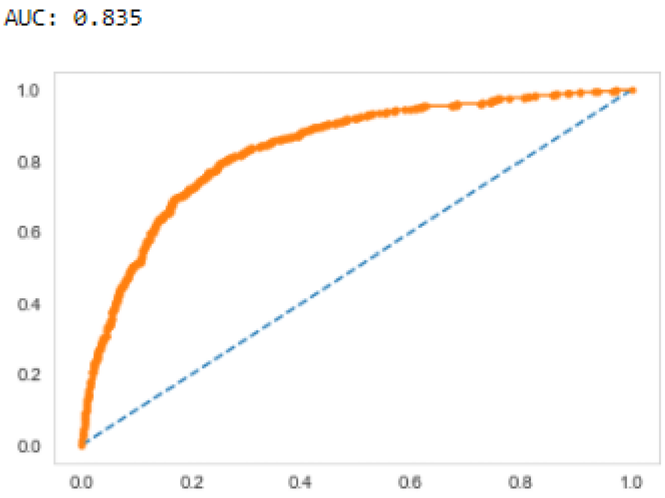
```
              precision    recall  f1-score   support

           0       0.82      0.82      0.82       571
           1       0.60      0.59      0.60       261

    accuracy                           0.75       832
   macro avg       0.71      0.71      0.71       832
weighted avg       0.75      0.75      0.75       832
```

Fig 2.37 Classification Report from test data of ANN model

ROC curve (**receiver operating characteristic curve**) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters.

- True Positive Rate
- False Positive Rate

The probability of the Area under the ROC curve for the train data is 83.5%



```
AUC: 0.835
```

Fig 2.38 AUC and ROC curve train data of ANN model

The probability of the Area under the ROC curve for the train data is 78.3%

```
AUC: 0.783
```

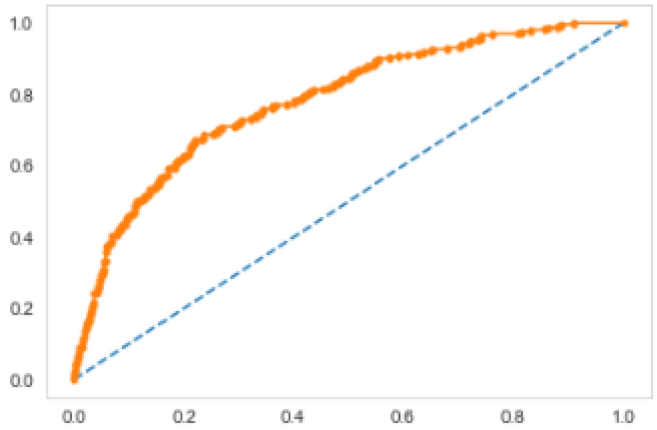


Fig 2.39 AUC and ROC curve train data of ANN model

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

Inference from the above Confusion matrix, classification report, Auc and Roc Curve, we can conclude that, for Random Forest classifier has the Highest accuracy in both train and test data.

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

Business Insights And Recommendations:

I strongly recommended we collect more real time unstructured data and past data if possible.

This is understood by looking at the insurance data by drawing relations between different variables such as day of the incident, time and associating it with other external information such as 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.

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

- Reduce claims cycle time
- Increase customer satisfaction
- Combat fraud