Data Mining Project

Name: Swetha Kunapuli

Batch & Course: PGP-DSBA

Online June Batch

Date: 24/10/2021

Table of Contents

List of Tables:

List of Figures:

PROBLEM 1

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. We are given the task to identify the segments based on credit card usage.

bank_marketing_part1_Data.csv

Data Introduction:

The purpose of this whole exercise is to explore the dataset and is recommended for learning and practicing our skills using clustering techniques.

The dataset contains 210 rows and 7 columns.

Description:

Description of variables is as follows:

- **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_sin
0	19.94	16.92	0.8752	6.675	3.763	3.252	
1	15.99	14.89	0.9064	5.363	3.582	3.336	
2	18.95	16.42	0.8829	6.248	3.755	3.368	
3	10.83	12.96	0.8099	5.278	2.641	5.182	
4	17.99	15.86	0.8992	5.890	3.694	2.068	

Table 1.1

Exploratory Data Analysis:

Check for types of variables in the data frame:

```
<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
```

Table 1.2

Observation:

7 variables and 210 records. No missing record based on initial analysis. All the variables numeric type.

Check for missing values in the dataset:

```
spending                      0
advance_payments              0
probability_of_full_payment   0
current_balance               0
credit_limit                  0
min_payment_amt               0
max_spent_in_single_shopping  0
dtype: int64
```

Table 1.3

Observation:

No missing values in the dataset.

Check for duplicate values in the dataset:

```
0
```

Observation:

No duplicate values in the dataset

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

Univariate analysis:

Checking the Summary Statistic:

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

Table 1.1.1

Observation:

- Based on summary descriptive, the data looks good.
- We see for most of the variable, mean/medium are nearly equal.
- Include a 90% to see variations and it looks distributed evenly.
- Std Deviation is high for spending variable.

Box Plot and Histogram for 'spending' variable:

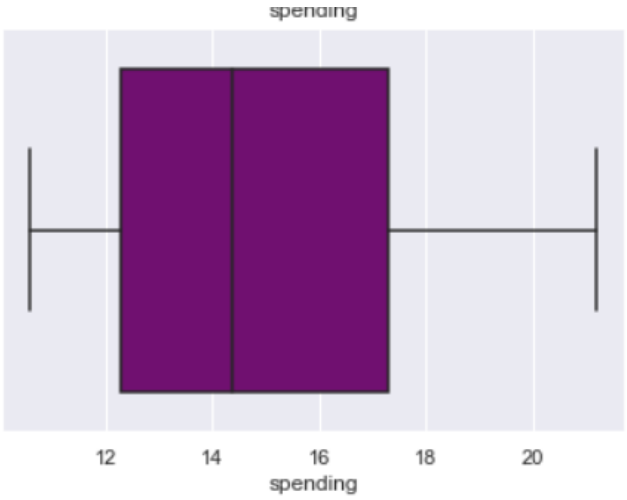

Fig 1.1.1

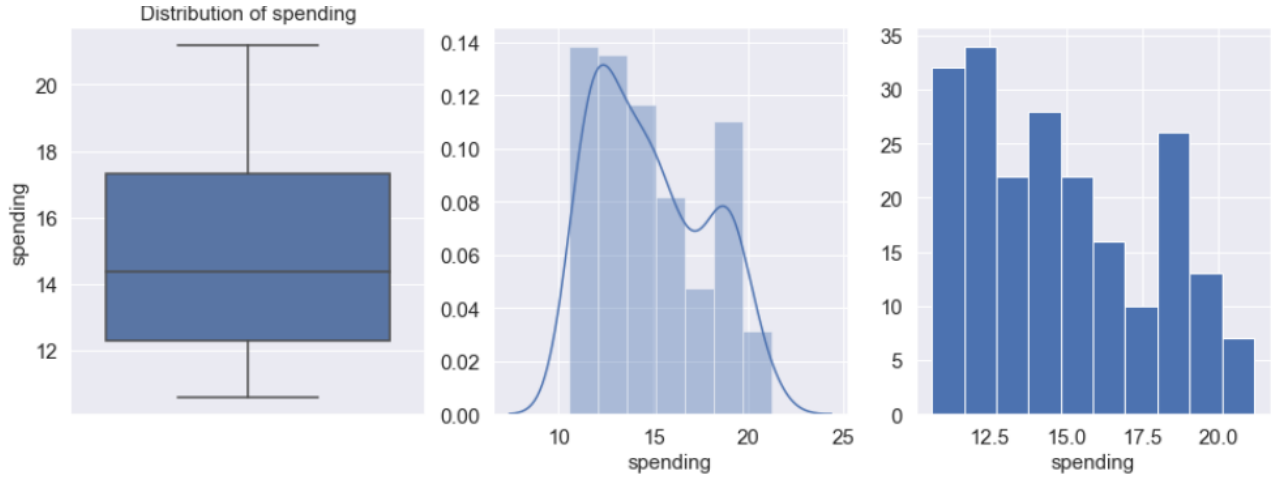

Fig 1.1.2

Box Plot and Histogram for ‘advance_payments’ variable:

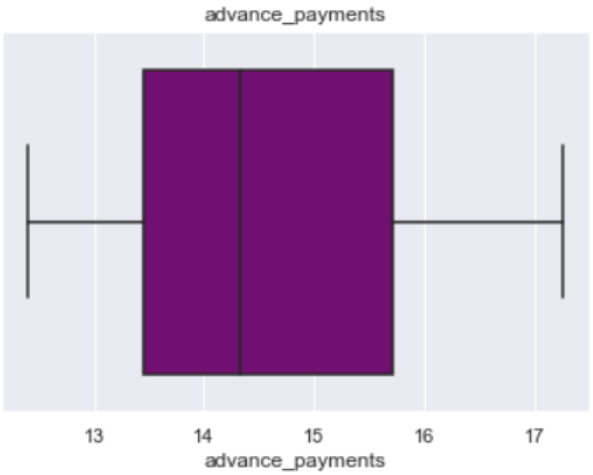

Fig 1.1.3

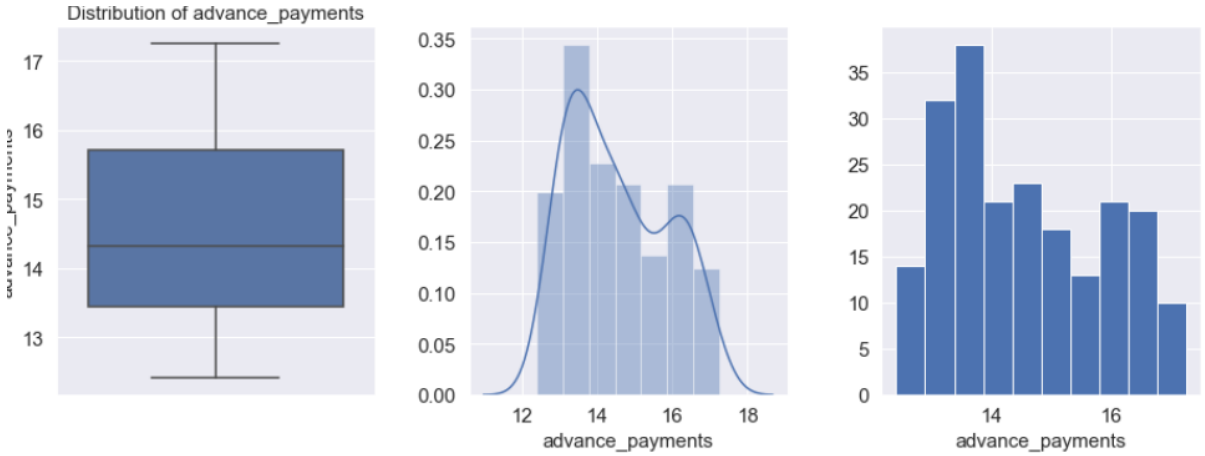

Fig 1.1.4

Box Plot and Histogram for ‘probability_of_full_payment’ variable:

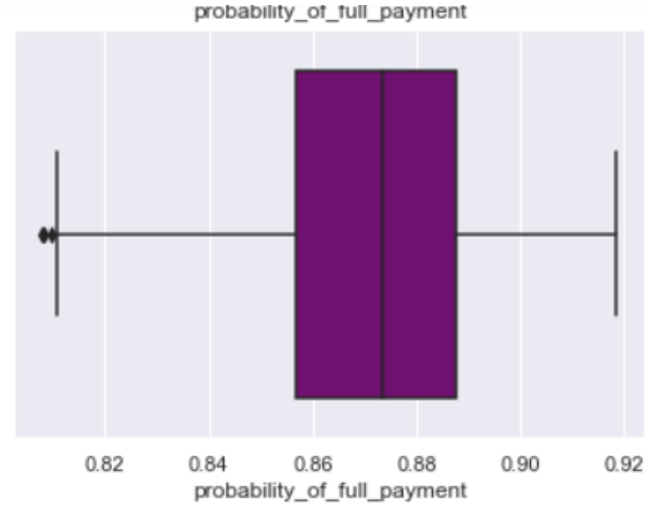

Fig 1.1.5

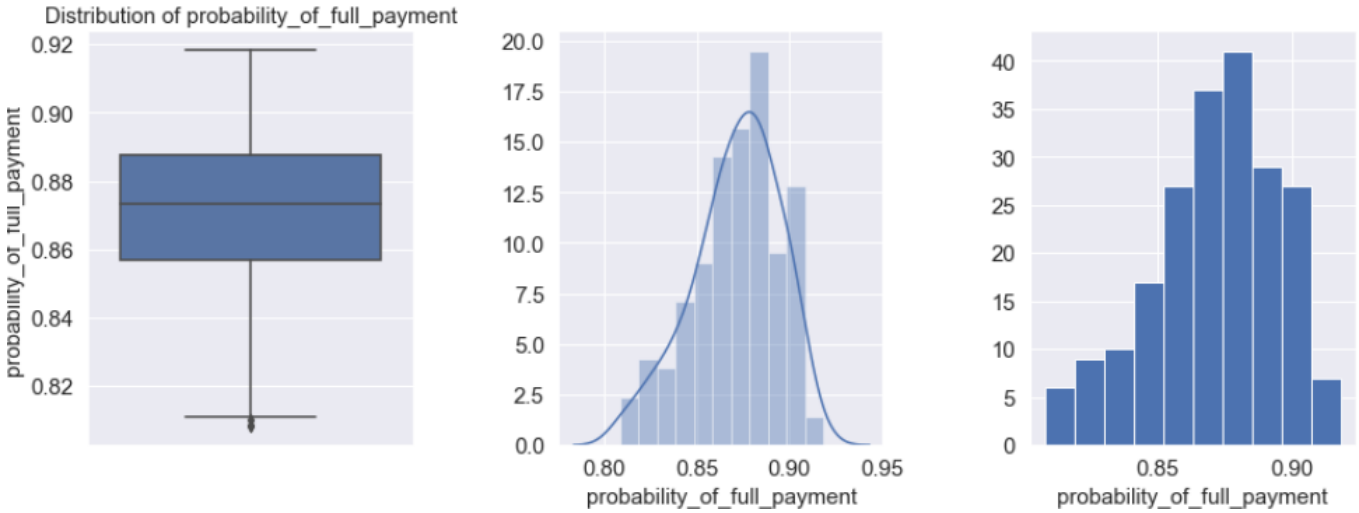

Fig 1.1.6

Box Plot and Histogram for ‘current_balance’ variable:

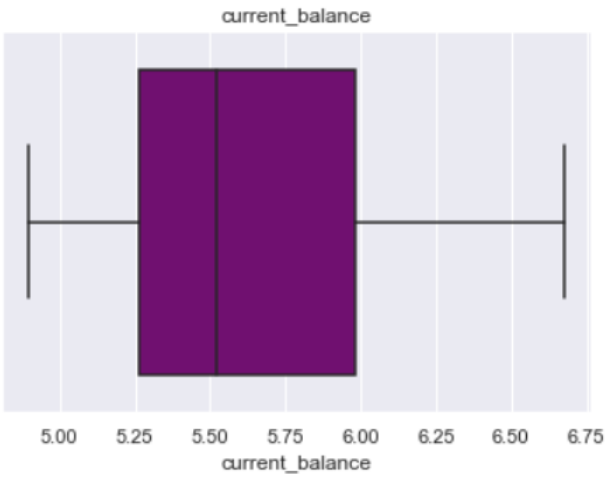

Fig 1.1.7

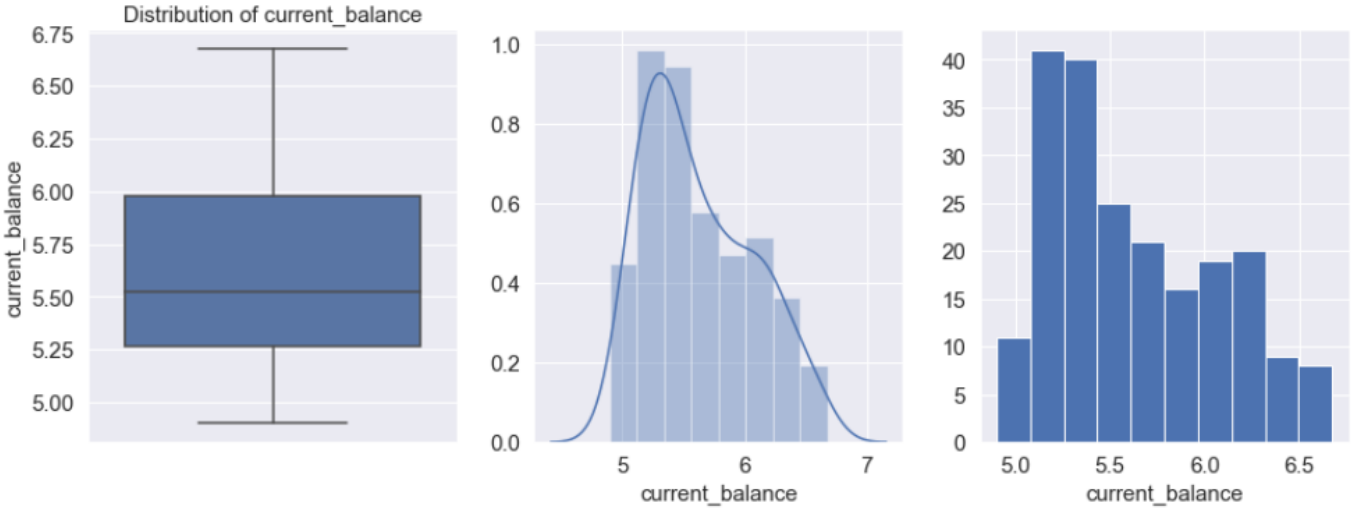

Fig 1.1.8

Box Plot and Histogram for ‘credit_limit’ variable:

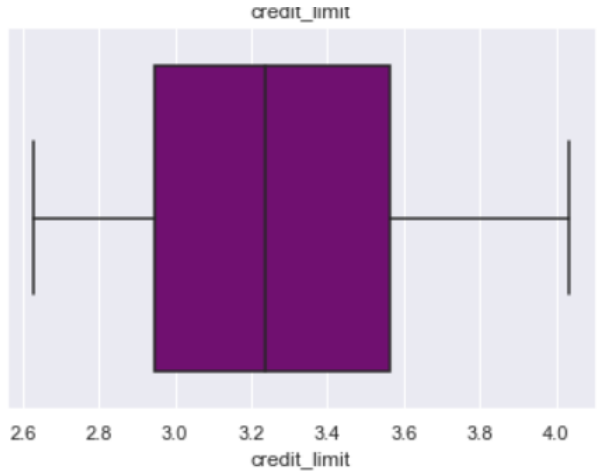

Fig 1.1.9

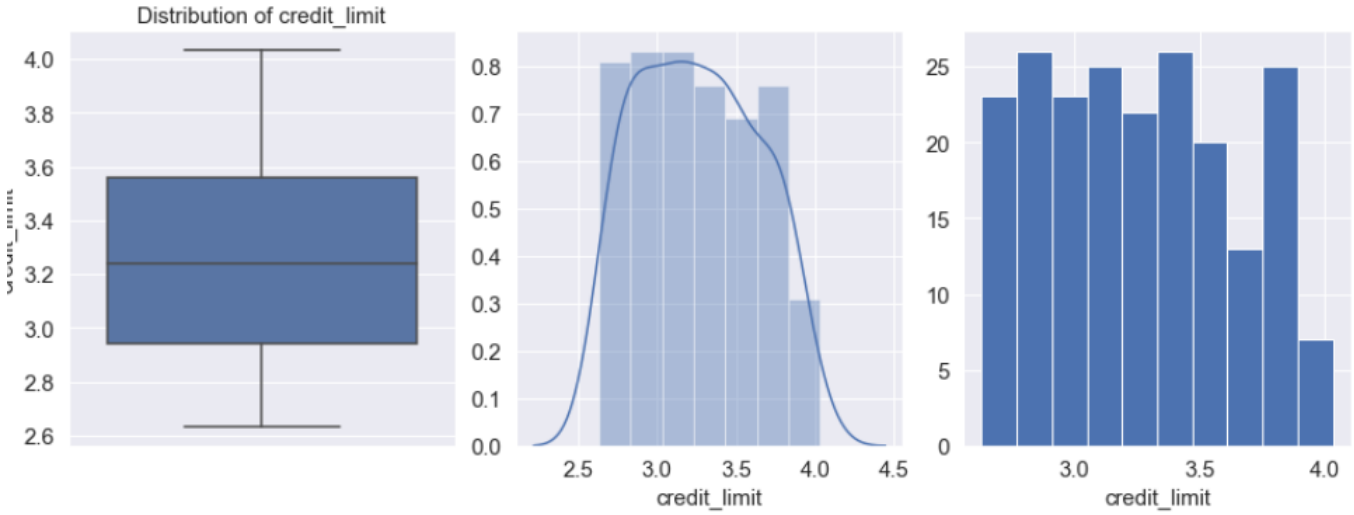

Fig 1.1.10

Box Plot and Histogram for ‘min_payment_amt’ variable:

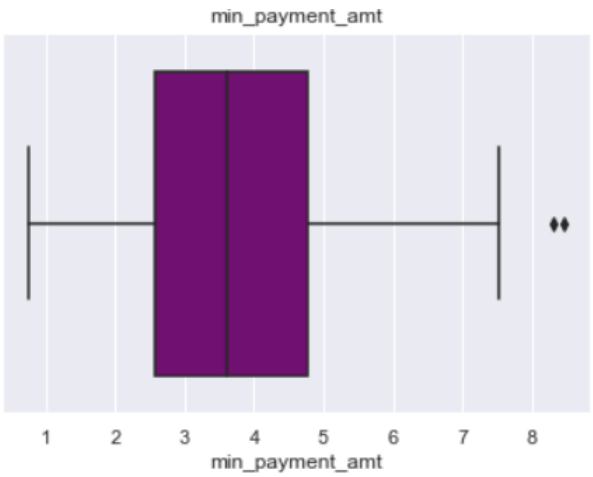

Fig 1.1.11

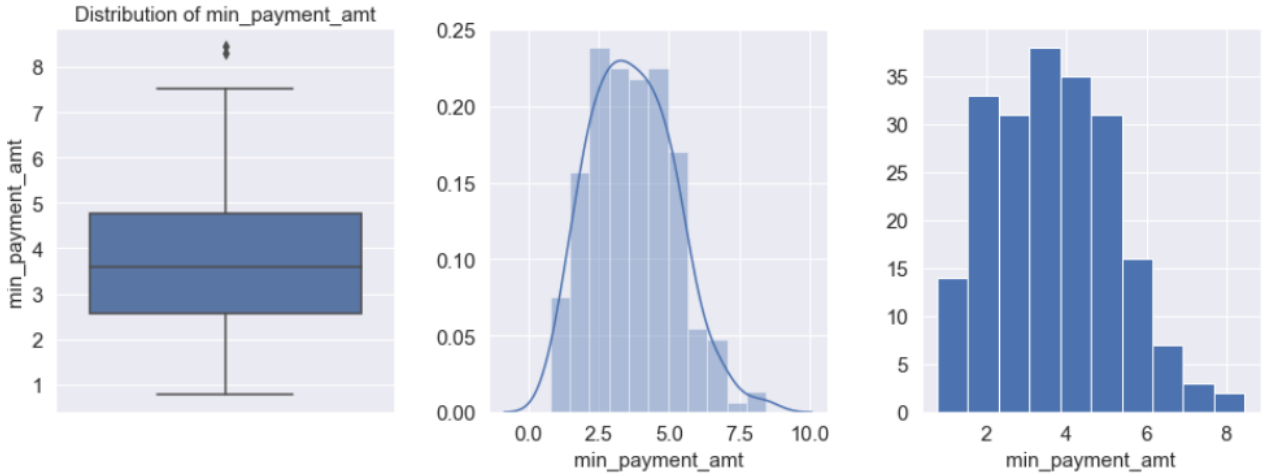

Fig 1.1.12

Box Plot and Histogram for ‘max_spent_in_single_shopping’ variable:

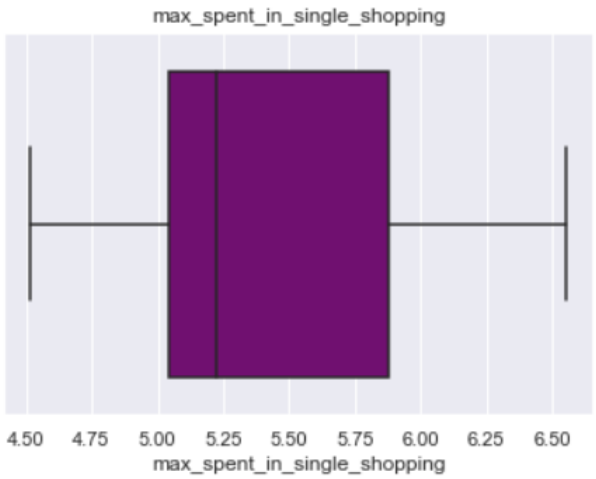

Fig 1.1.13

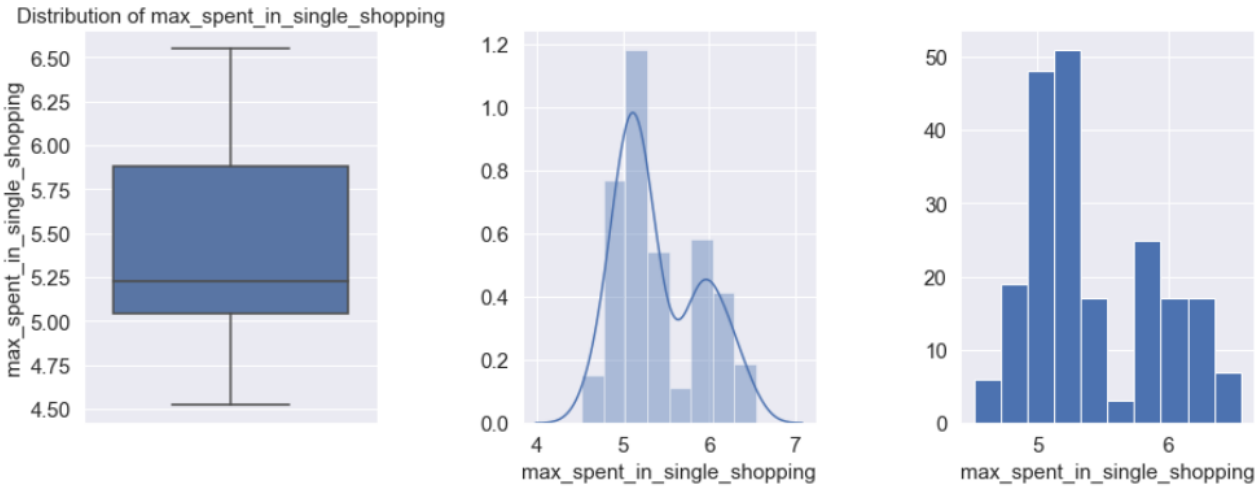

Fig 1.1.14

Histogram for all Variables:

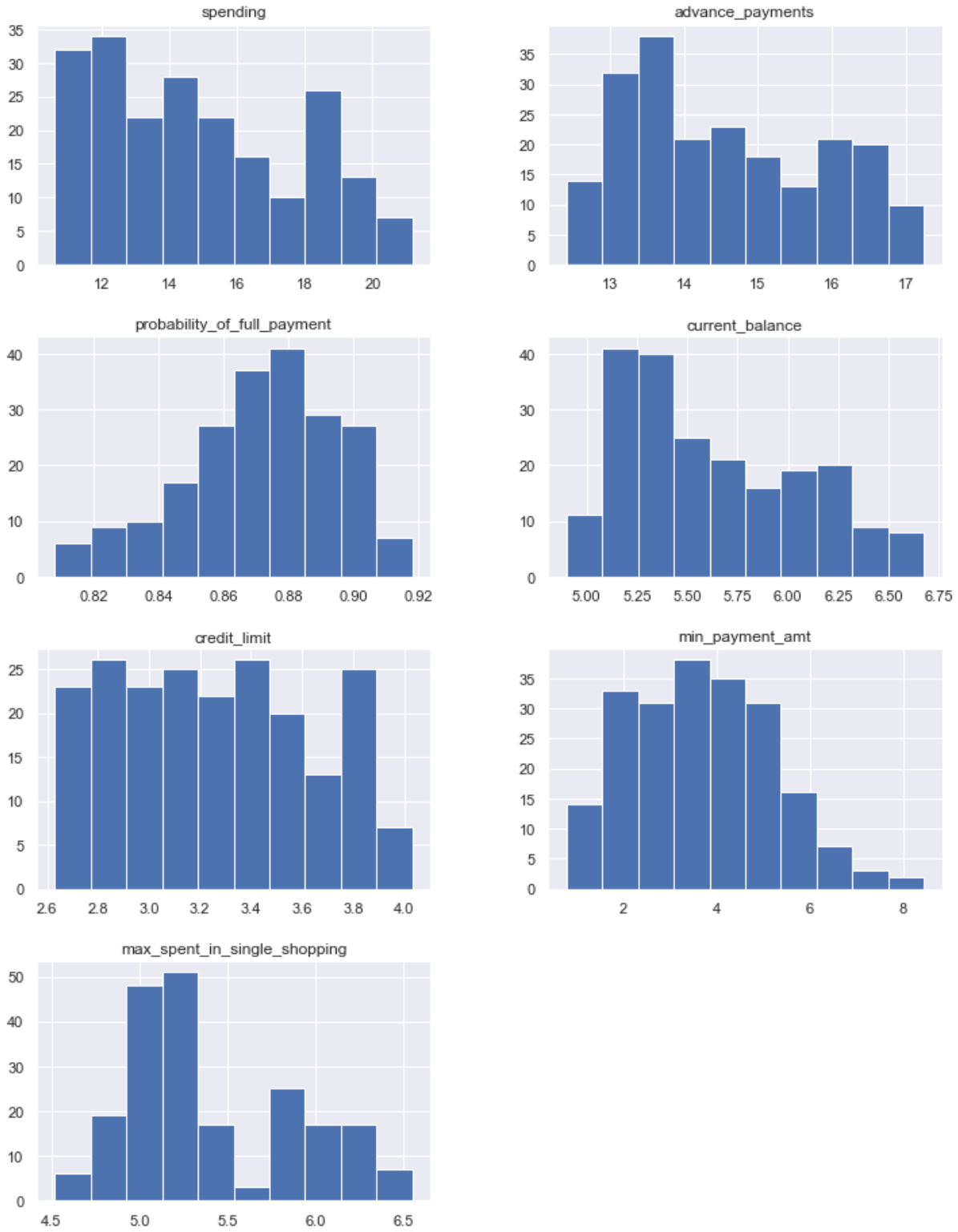

Fig 1.1.15

Multivariate analysis

Check for multicollinearity

Pair Plot:

Fig 1.1.16

Observation:

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	max_spent_in_single_shopping
spending	1.000000	0.994341	0.608288	0.949985	0.970771	0.863693
advance_payments	0.994341	1.000000	0.529244	0.972422	0.944829	0.890784
probability_of_full_payment	0.608288	0.529244	1.000000	0.367915	0.761635	0.226825
current_balance	0.949985	0.972422	0.367915	1.000000	0.860415	0.932806
credit_limit	0.970771	0.944829	0.761635	0.860415	1.000000	0.749131
min_payment_amt	-0.229572	-0.217340	-0.331471	-0.171562	-0.258037	-0.011079
max_spent_in_single_shopping	0.863693	0.890784	0.226825	0.932806	0.749131	1.000000

Table 1.1.2

Heat Map:

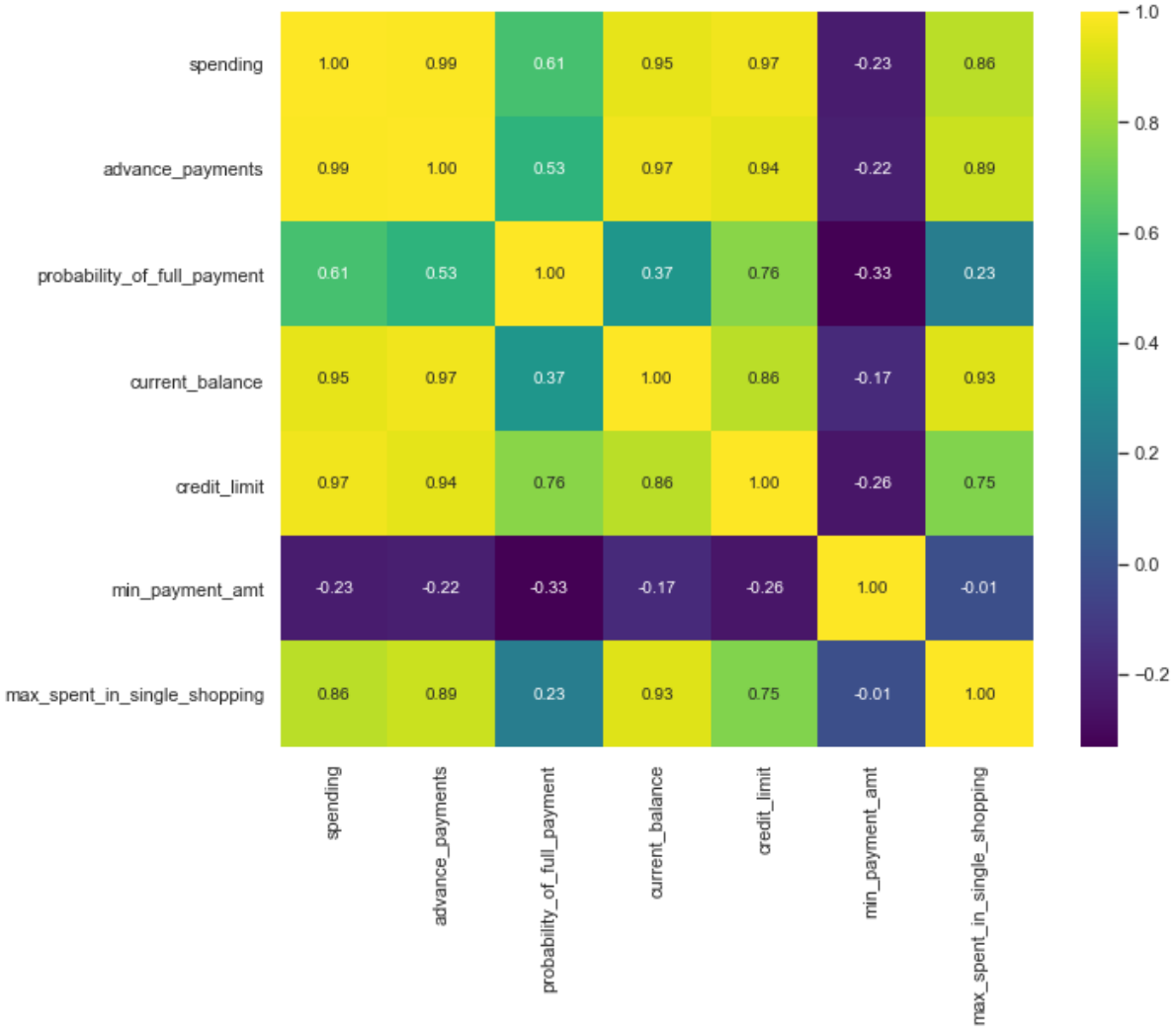

Fig 1.1.17

Box Plot for Outliers:

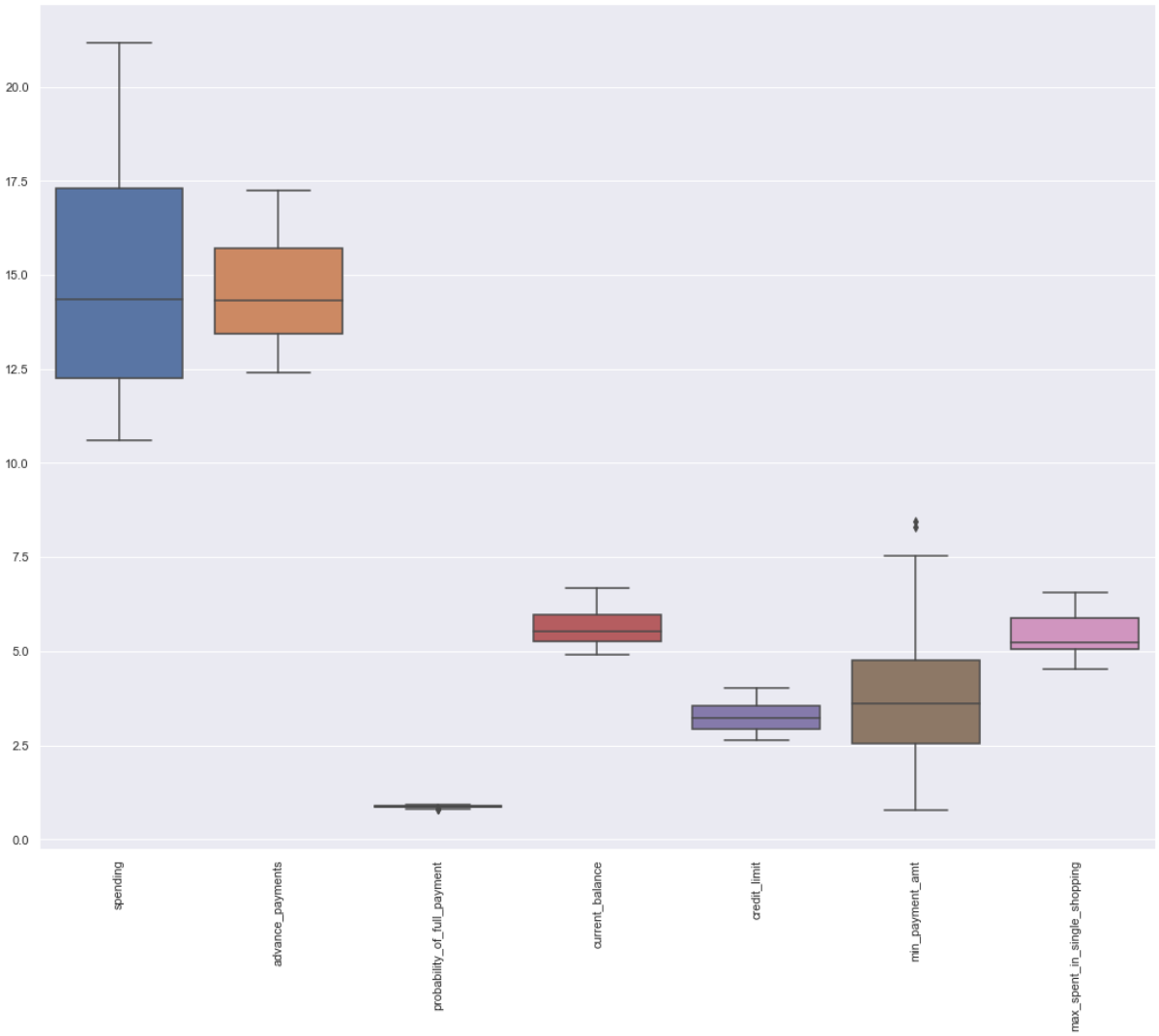

Fig 1.1.18

Observation:

We see one outlier as per the boxplot, it is okay, as it has no extreme and on lower band.

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

Scaling needs to be done as the values of the variables are different.

'spending', 'advance_payments' variables are in different values, and this may get more weightage.

Also have shown below 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 standardize the data to relative same scale -3 to +3.

	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
0	1.754355	1.811968	0.178230	2.367533	1.338579	-0.298806	2.328998
1	0.393582	0.253840	1.501773	-0.600744	0.858236	-0.242805	-0.538582
2	1.413300	1.428192	0.504874	1.401485	1.317348	-0.221471	1.509107
3	-1.384034	-1.227533	-2.591878	-0.793049	-1.639017	0.987884	-0.454961
4	1.082581	0.998364	1.196340	0.591544	1.155464	-1.088154	0.874813

Table 1.2.1

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

Imported dendrogram and linkage module:

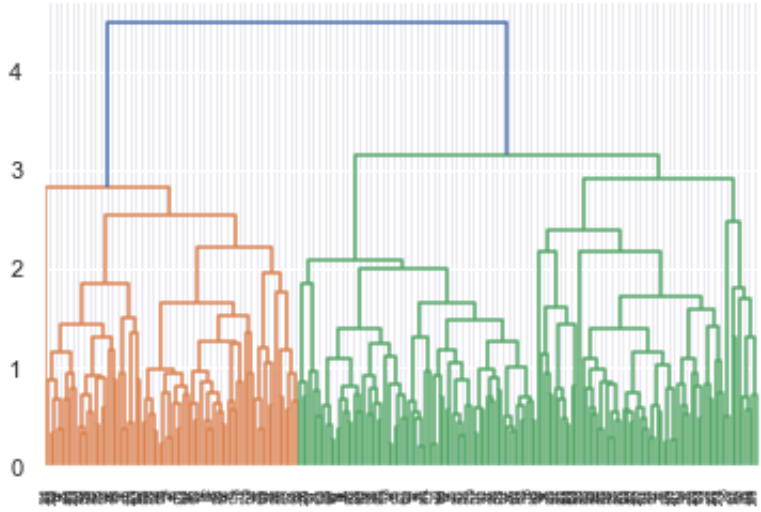

Fig 1.3.1

Dendrogram with truncate method as ‘lastp’ and with p value which is 10 using linkage method:

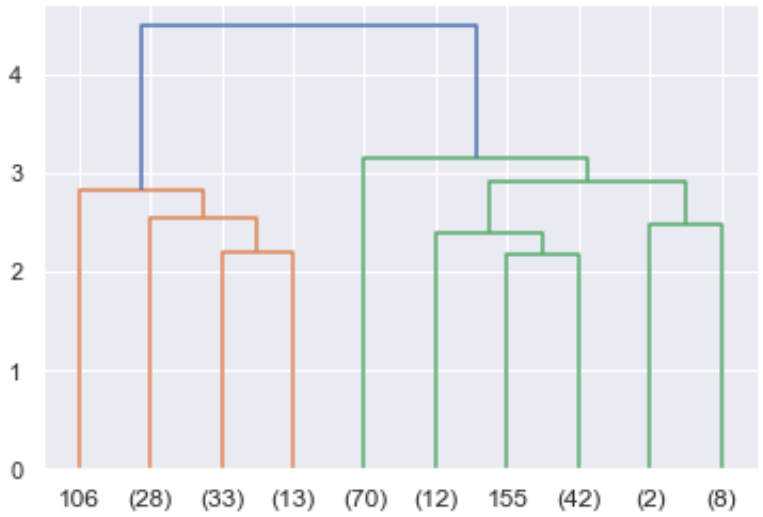

Fig 1.3.2

Dendrogram with truncate mode as ‘lastp’ and with p value which is 25 using linkage method:

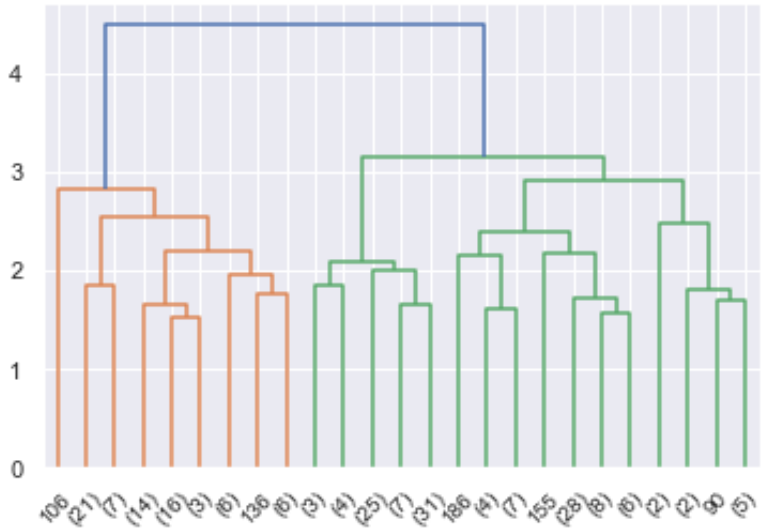

Fig 1.3.3

Next step is to import fcluster module to create clusters.

Below is the matrix of 3 clusters created using linkage method with criterion as ‘maxclust’:

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

Once 3 clusters are created, added clusters-3 as new column to the original dataset for further analysis.

Below is the head of the dataset with 3 cluster groups created using linkage method:

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

Table 1.3.1

Cluster Frequency for linkage method:

```
1    75
2    70
3    65
Name: clusters-3, dtype: int64
```

Cluster Profiles for linkage method:

clusters-3	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	Freq
1	18.129200	16.058000	0.881595	6.135747	3.648120	3.650200	5.987040	75
2	11.916857	13.291000	0.846766	5.258300	2.846000	4.619000	5.115071	70
3	14.217077	14.195846	0.884869	5.442000	3.253508	2.768418	5.055569	65

Table 1.3.2

Another method is using ward-link method

Dendrogram after using ward-link method:

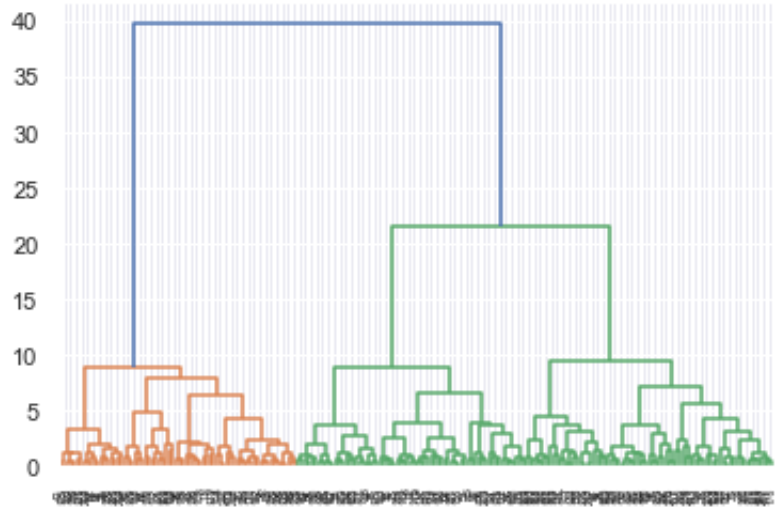

Fig 1.3.4

Dendrogram with truncate mode as ‘lastp’ where p is the value of clusters which is 10.

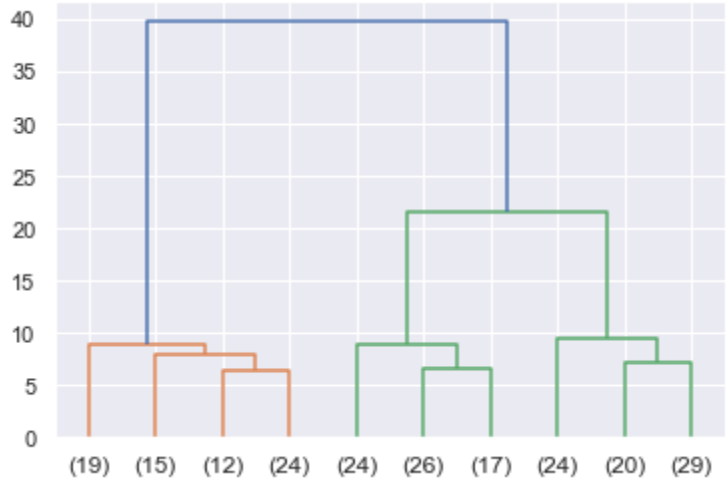

Fig 1.3.5

Below is the matrix of 3 clusters created using ward-link method with criterion as ‘maxclust’:

```
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)
```

Once 3 clusters are created using ward-link method, created ‘clusters-3’ as new column and added to original dataset.

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

Table 1.3.3

Cluster Frequency for ward-link method:

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

Clusters Profiles for ward-link method:

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

Table 1.3.4

Observation:

- Both the methods are almost similar which means, minor variation which we know it occurs.
- For cluster grouping based on the dendrogram, 3 or 4 looks good.
- Did the further analysis and based on the dataset had gone for 3 group cluster solution based on the hierarchical clustering.
- Cluster 2 is the least spending group; Cluster 3 is the medium spending group and Cluster 1 is the highest spending group.
- Also in real time, there could have been more variables value captured - tenure, balance_frequency, balance, purchase, instalment of purchase, others.
- 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).

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 value for scaled data for 1 cluster:

```
1469.9999999999995
```

Inertia value for scaled data for 2 clusters in K-Means:

```
659.1717544870411
```

Inertia value for scaled data for 3 clusters in K-Means:

```
430.65897315130064
```

K-Means value for scaled data for 4 clusters in K-Means:

```
371.28344766743334
```

Inertia values within 1 to 10 range:

```
[1469.9999999999995,
 659.1717544870411,
 430.65897315130064,
 371.6531439995162,
 326.3676022658374,
 289.50200732273396,
 262.7710624151513,
 242.1401080953628,
 224.7210999213561,
 203.88950413782644]
```

Visualising data using Elbow curve:

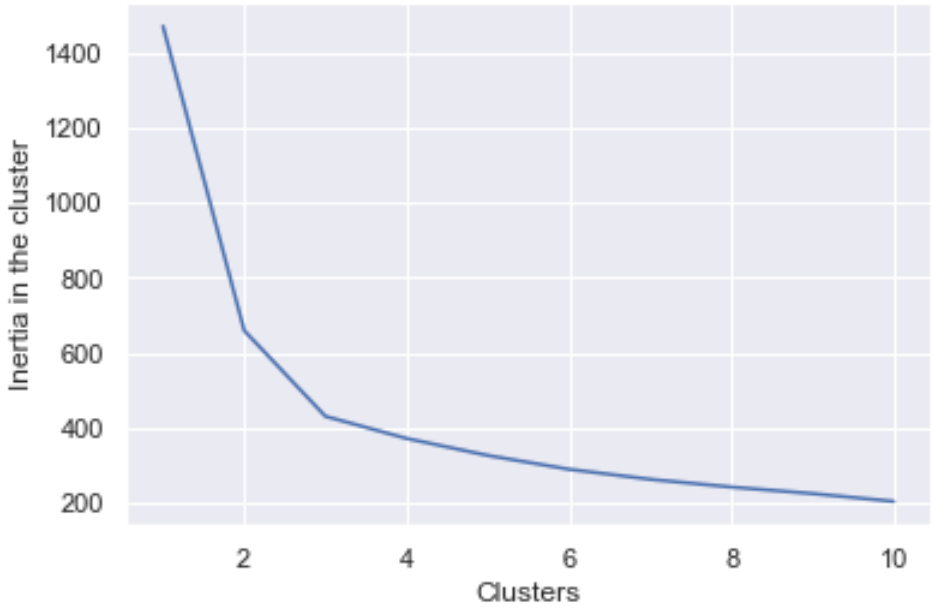

Fig 1.4.1

Silhouette Score:

```
[0.46577247686580914,
 0.40072705527512986,
 0.3291966792017613,
 0.28316654897654814,
 0.2897583830272519,
 0.2694844355168536,
 0.2543731602750563,
 0.2623959398663564,
 0.2673980772529918]
```

Plot for Silhouette Score:

Fig 1.4.2

Insights:

From SC Score, the number of optimal clusters could be 3 or 4.

Head of the dataset after adding Sil-Width:

	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	Clus_kmeans	sil_width
0	19.94	16.92	0.8752	6.675	3.763	3.252	6.550	2	0.445327
1	15.99	14.89	0.9064	5.363	3.582	3.336	5.144	0	0.049939
2	18.95	16.42	0.8829	6.248	3.755	3.368	6.148	2	0.443575
3	10.83	12.96	0.8099	5.278	2.641	5.182	5.185	3	0.532008
4	17.99	15.86	0.8992	5.890	3.694	2.068	5.837	2	0.081568

Table 1.4.1

Minimum value of Silhoutte samples for scaled data.

```
-0.053840826993600814
```

Insights:

I am choosing 3-cluster solution based on the above analysis did for inertia visualization as well as silhouette score visualization.

Cluster Percentage for 3-cluster solution:

	Cluster_Size	Cluster_Percentage
1	67	31.90
2	72	34.29
3	71	33.81

Table 1.4.2

Transpose of 3-cluster solution:

cluster	1	2	3
spending	18.5	11.9	14.4
advance_payments	16.2	13.2	14.3
probability_of_full_payment	0.9	0.8	0.9
current_balance	6.2	5.2	5.5
credit_limit	3.7	2.8	3.3
min_payment_amt	3.6	4.7	2.7
max_spent_in_single_shopping	6.0	5.1	5.1

Table 1.4.3

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

3 group cluster via hierarchical clustering:

clusters-3	1	2	3
spending	18.371429	11.872388	14.199041
advance_payments	16.145429	13.257015	14.233562
probability_of_full_payment	0.884400	0.848072	0.879190
current_balance	6.158171	5.238940	5.478233
credit_limit	3.684629	2.848537	3.226452
min_payment_amt	3.639157	4.949433	2.612181
max_spent_in_single_shopping	6.017371	5.122209	5.086178
Freq	70.000000	67.000000	73.000000

Table 1.5.1

Insights:

Cluster Group Profiles

Group 1 : High Spending

Group 3 : Medium Spending

Group 2 : Low 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_maximum spending

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/loyalty cars to increase transactions.
- Increase spending habits by trying with premium ecommerce sites, travel port al, travel airlines/hotel, as this will encourage them to spend more.

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 tie up with grocery stores, utilities (electricity , phone, gas, others).

PROBLEM 2

Executive Summary:

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

insurance_part2_da ta-2.csv

Data Introduction:

The purpose of this whole exercise is to explore the dataset and is recommended for learning and practicing our skills using CART, RF & ANN models.

The dataset contains 3000 rows and 10 columns.

Description:

Description of variables is as follows:

- Target: Claim Status (Claimed)
- Code of tour firm (Agency_Code)
- Type of tour insurance firms (Type)
- Distribution channel of tour insurance agencies (Channel)
- Name of the tour insurance products (Product)
- Duration of the tour (Duration in days)
- Destination of the tour (Destination)
- Amount worth of sales per customer in procuring tour insurance policies in rupees (in 100's)
- The commission received for tour insurance firm (Commission is in percentage of sales)
- Age of insured (Age)

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

Exploratory Data Analysis:

Check for types of variables in the data frame:

```
<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   float6
 5   Channel       3000 non-null   object
 6   Duration      3000 non-null   int64
 7   Sales         3000 non-null   float6
 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
```

Table 2.2

Observations:

- 10 variables
- Age, Commission, Duration, Sales are numeric variable
- rest are categorial variables
- 3000 records, no missing one
- 9 independent variable and one target variable – Claimed

Check for missing values in the dataset:

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

Table 2.3

Observation:

No missing values

Check for duplicate values in the dataset:

```
Number of duplicated rows = 139
```

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

Table 2.4

Observations:

Removing Duplicates - Not removing them - no unique identifier, can be different customer.

Though it shows there are 139 records, but it can be of different customers, there is no customer ID or any unique identifier, so I am not dropping them off.

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

Univariate analysis

Checking the Summary Statistic:

	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 2.1.1

Observation:

Categorial code variable maximum unique count is 5.

Box Plot and Histogram for ‘Age’ Variable:

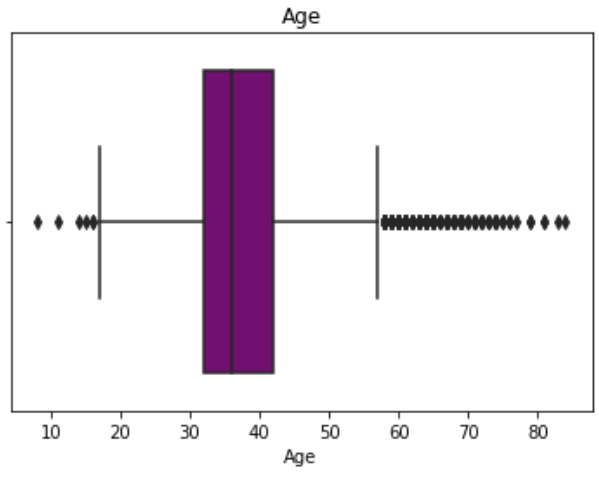

Fig 2.1.1

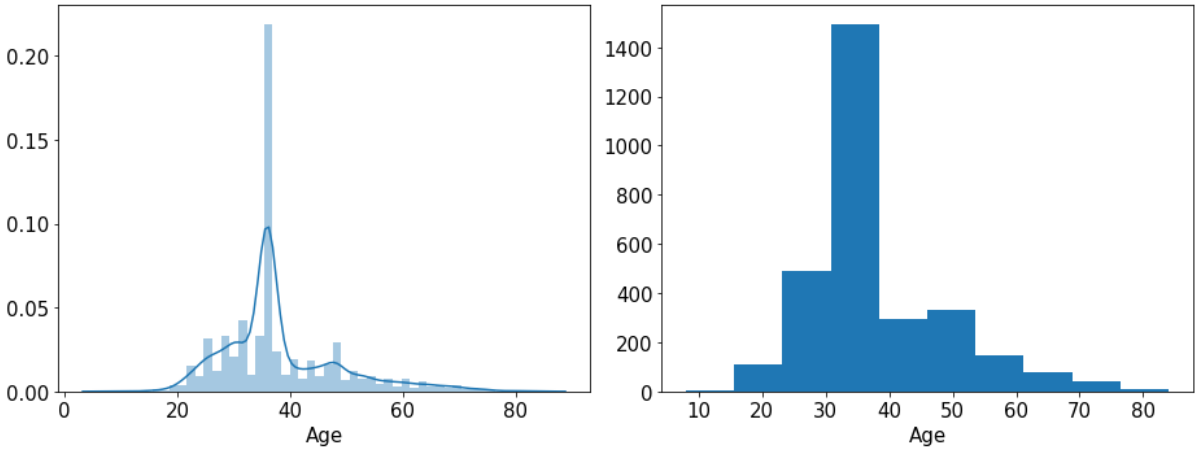

Fig 2.1.2

Box Plot and Histogram for 'Commission' Variable:

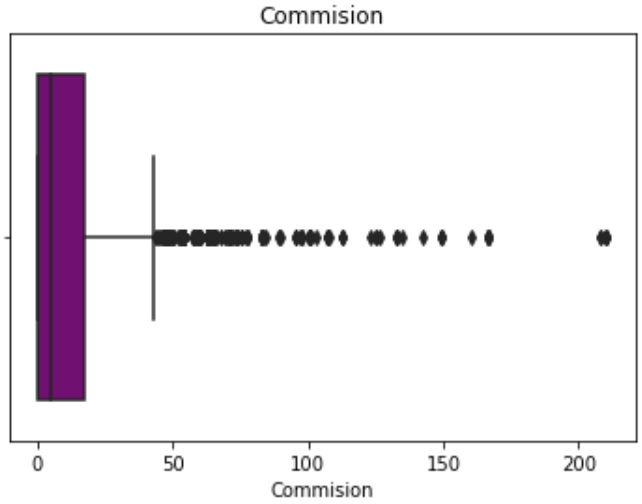

Fig 2.1.3

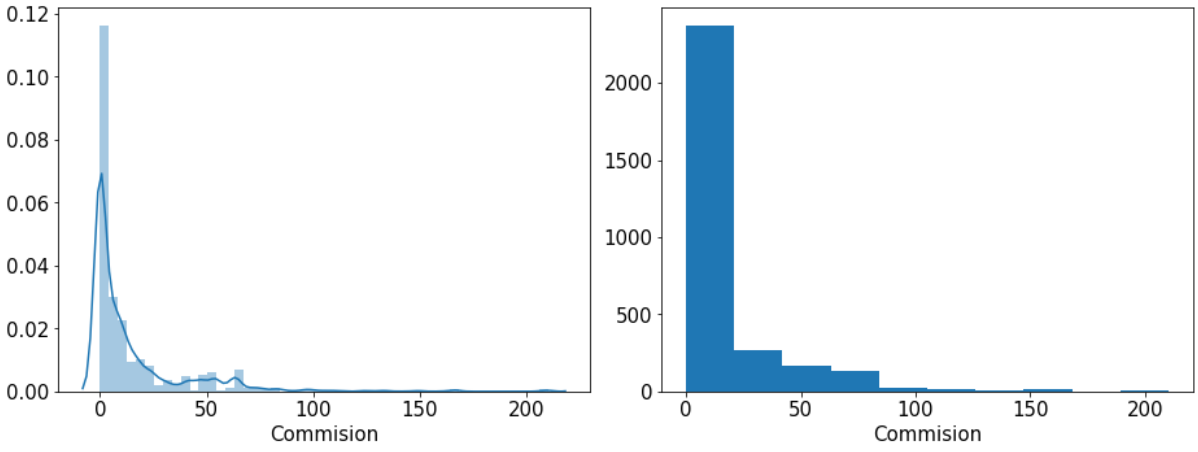

Fig 2.1.4

Box Plot and Histogram for ‘Duration’ Variable:

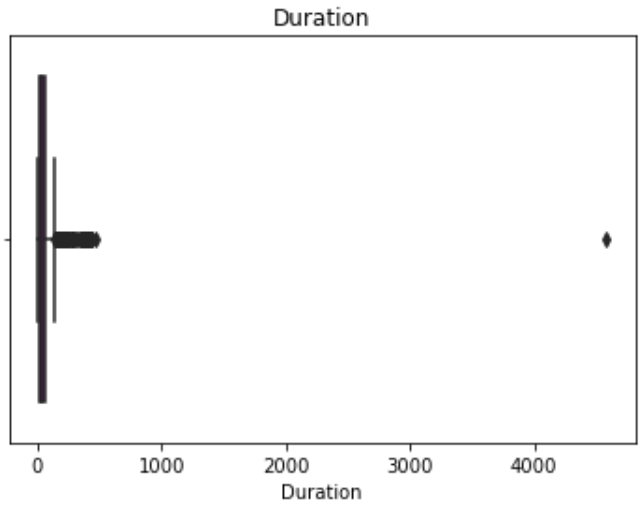

Fig 2.1.5

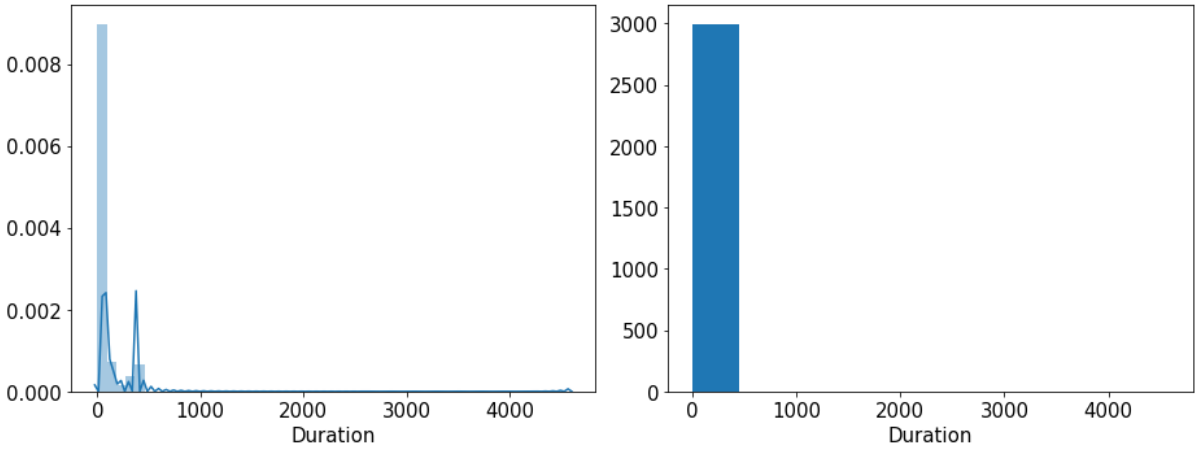

Fig 2.1.6

Box Plot and Histogram for ‘Sales’ Variable:

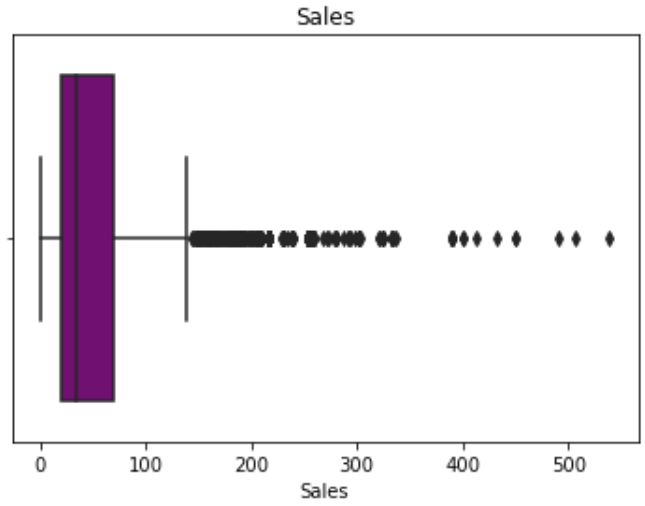

Fig 2.1.7

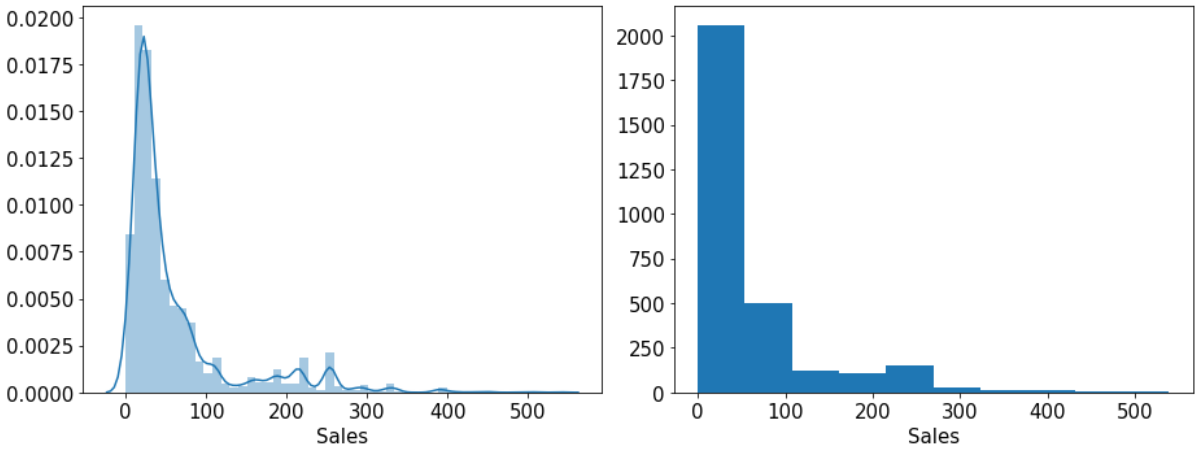

Fig 2.1.8

Observations:

There are outliers in all the variables, but the sales and commission can be a genuine business value. Random Forest and CART can handle the outliers. Hence, Outliers are not treated for now, we will keep the data as it is.

I will treat the outliers for the ANN model to compare the same after the all the steps just for comparison.

Categorical Variables

Count Plot for ‘Agency_Code’ Variable:

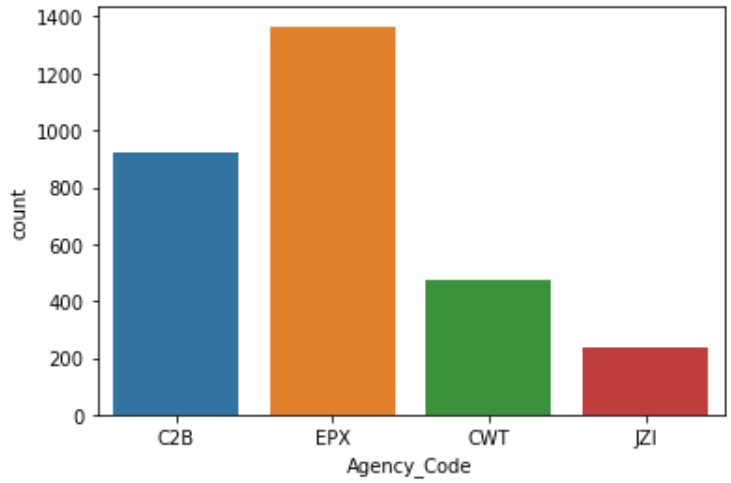

Fig 2.1.9

Box Plot and Swarm Plot comparing ‘Sales’ & ‘Agency_Code’:

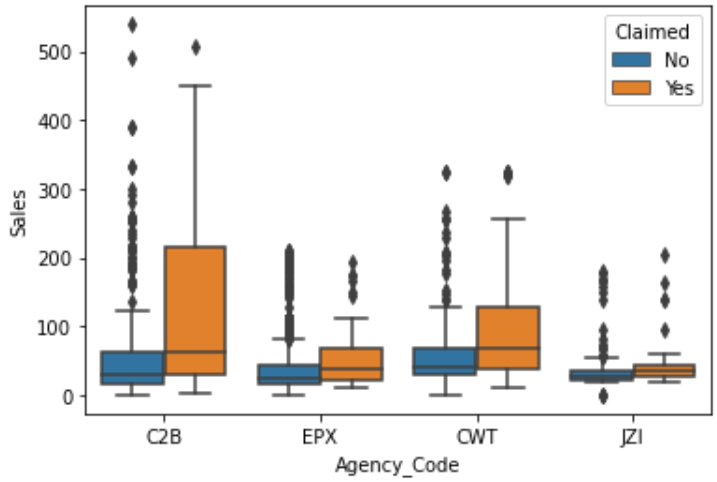

Fig 2.1.10

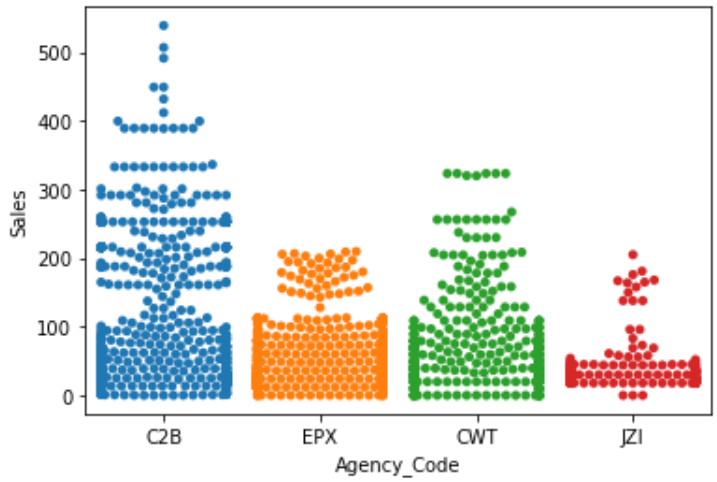

Fig 2.1.11

Count Plot for 'Type' Variable:

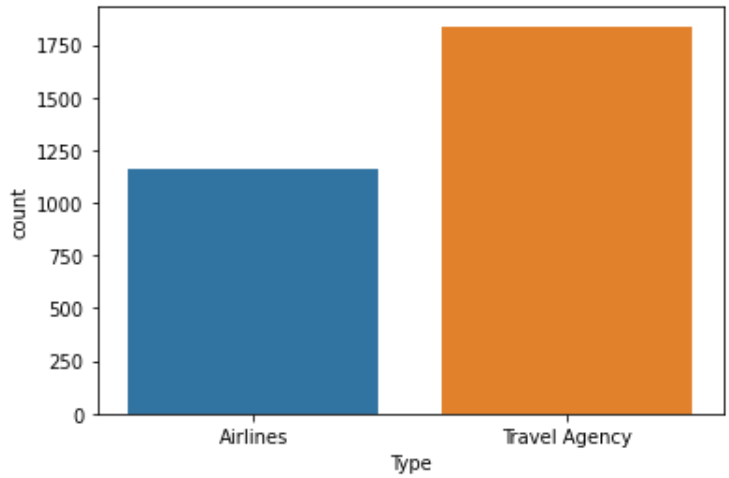

Fig 2.1.12

Box Plot and Swarm Plot comparing ‘Sales’ & ‘Type’:

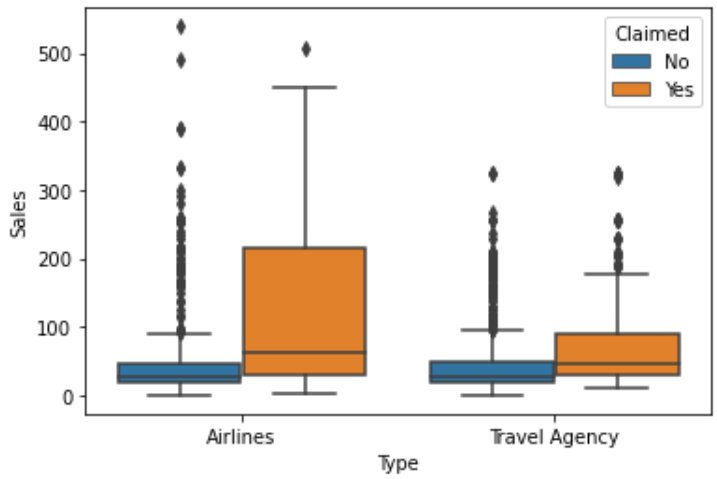

Fig 2.1.13

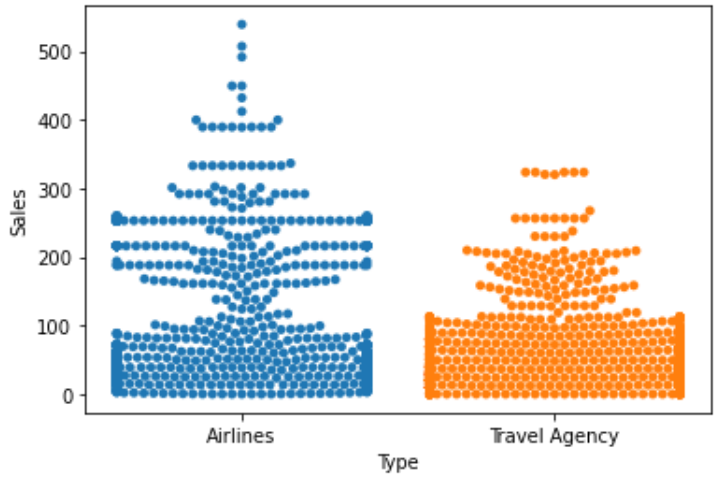

Fig 2.1.14

Count Plot for ‘Channel’ Variable:

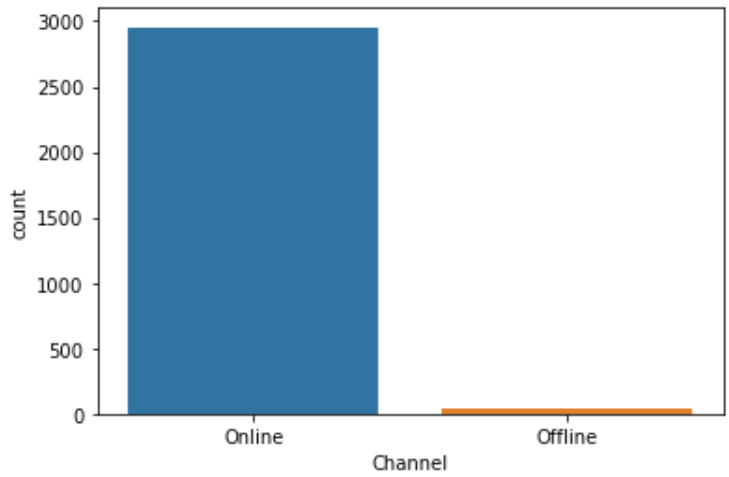

Fig 2.1.15

Box Plot and Swarm Plot comparing ‘Sales’ & ‘Channel’:

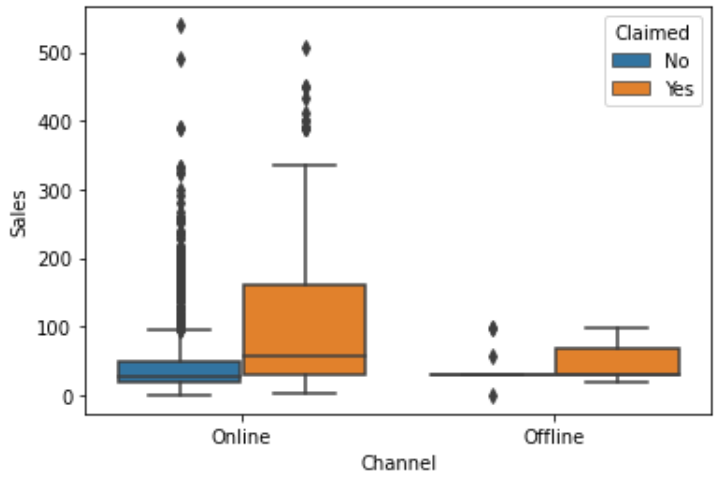

Fig 2.1.16

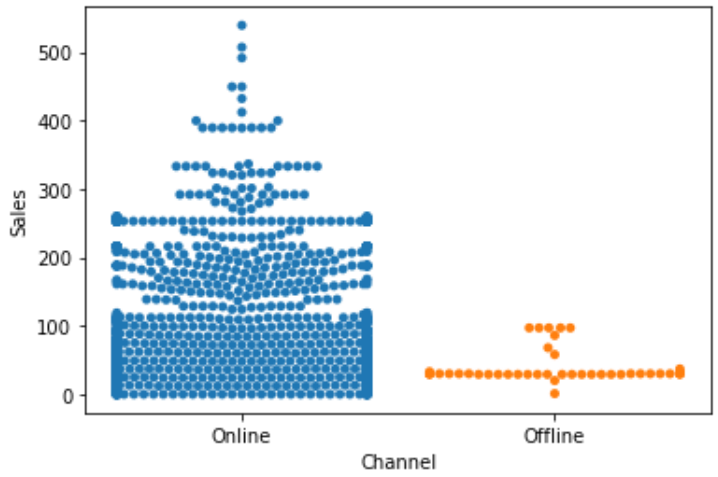

Fig 2.1.17

Count Plot for 'Product Name' Variable:

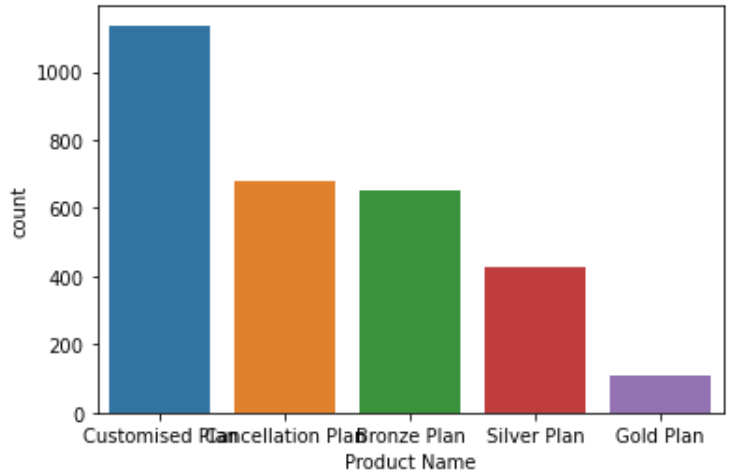

Fig 2.1.18

Box Plot and Swarm Plot comparing ‘Sales’ & ‘Product Name’:

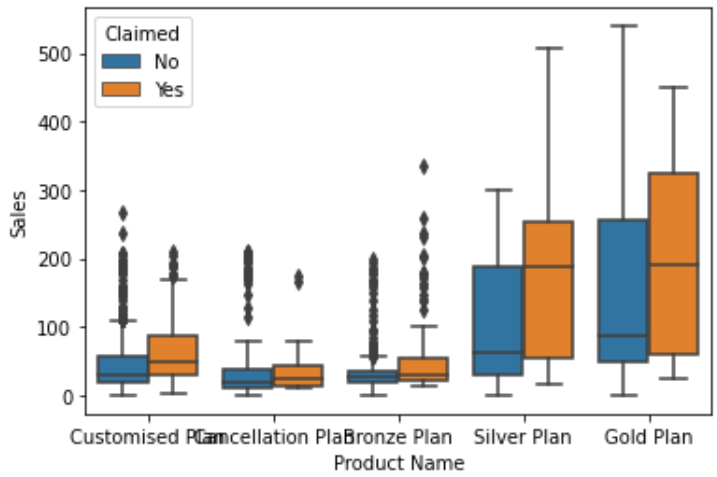

Fig 2.1.19

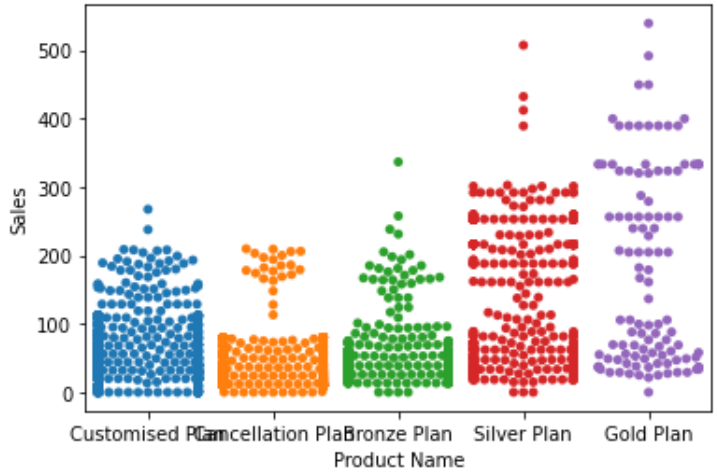

Fig 2.1.20

Count Plot for ‘Destination’ Variable:

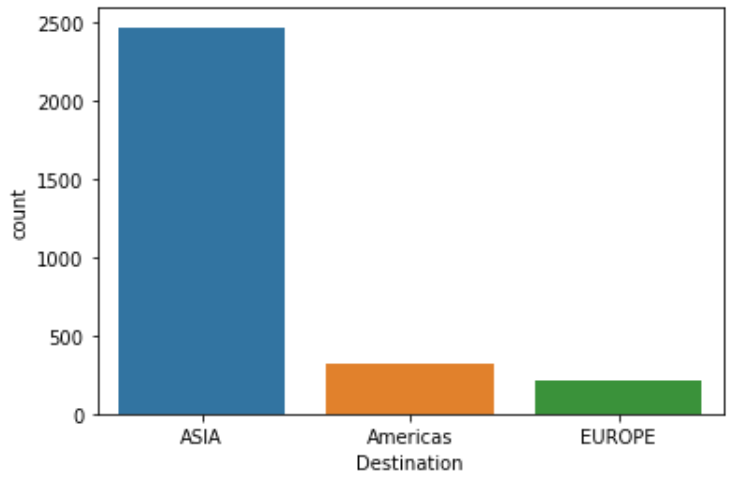

Fig 2.1.21

Box Plot and Swarm Plot comparing ‘Sales’ & ‘Destination’:

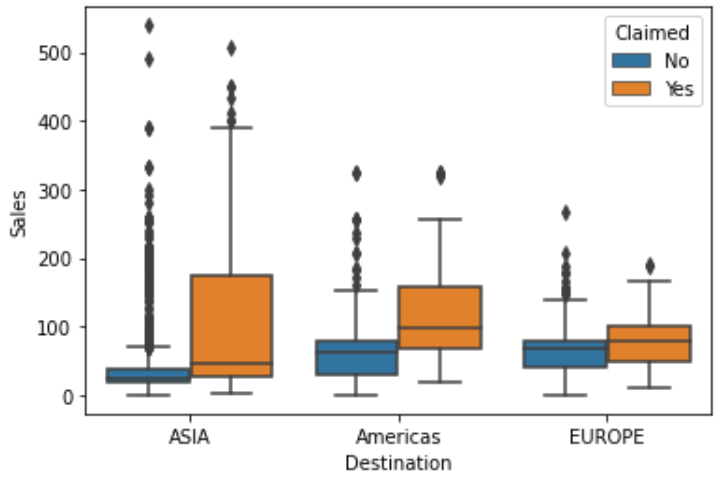

Fig 2.1.22

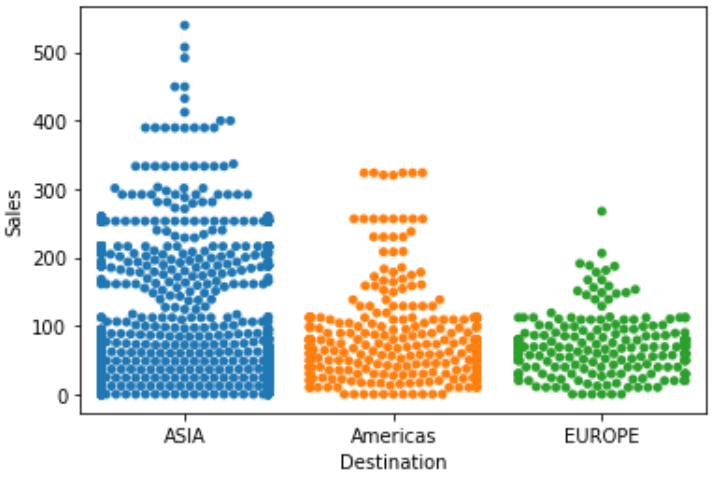
500
400
300
200
100
0
Sales
ASIA
Americas
EUROPE
Destination

Fig 2.1.23

Checking pairwise distribution of the continuous variables:

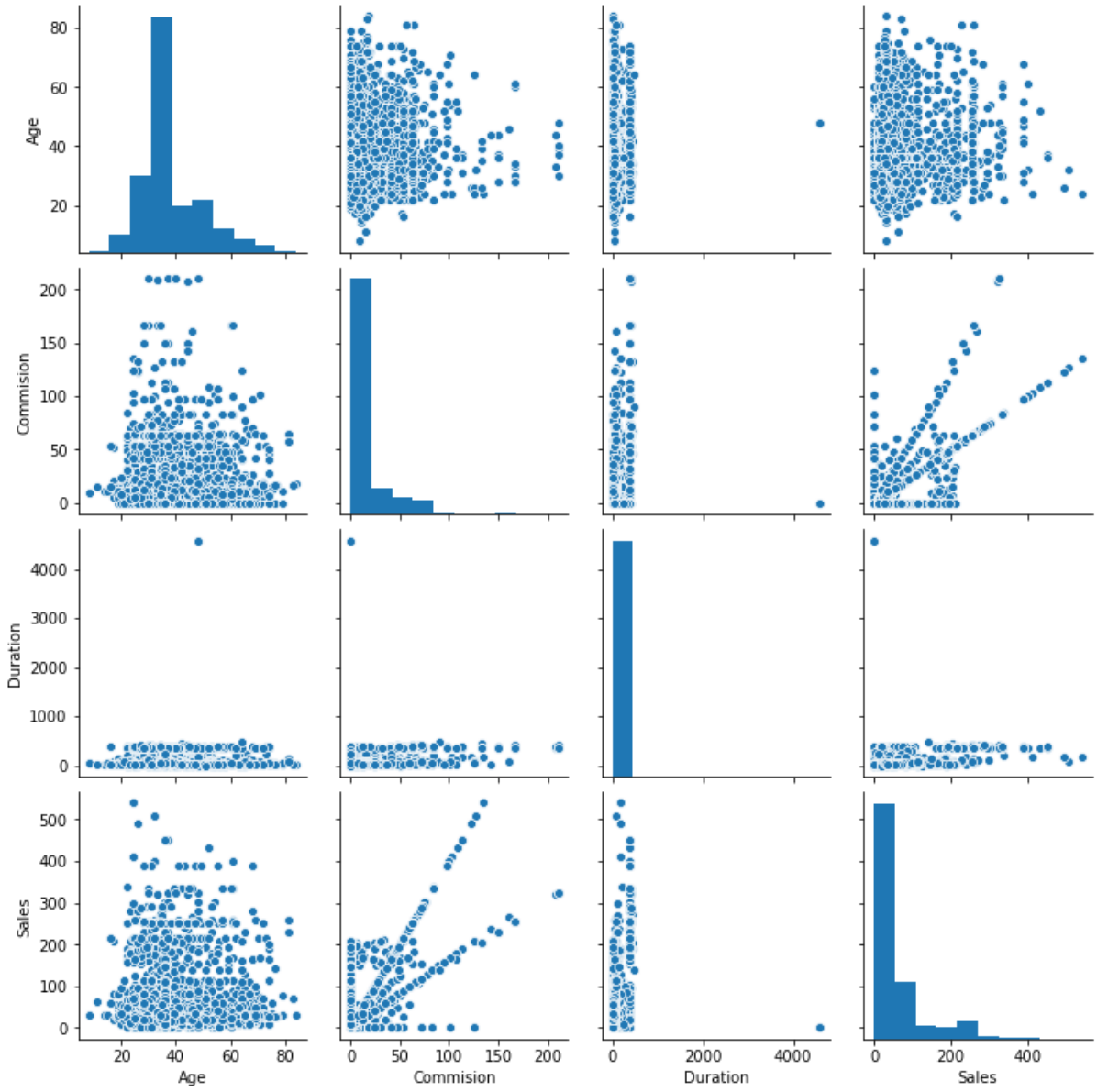

Fig 2.1.24

Checking for Correlations:

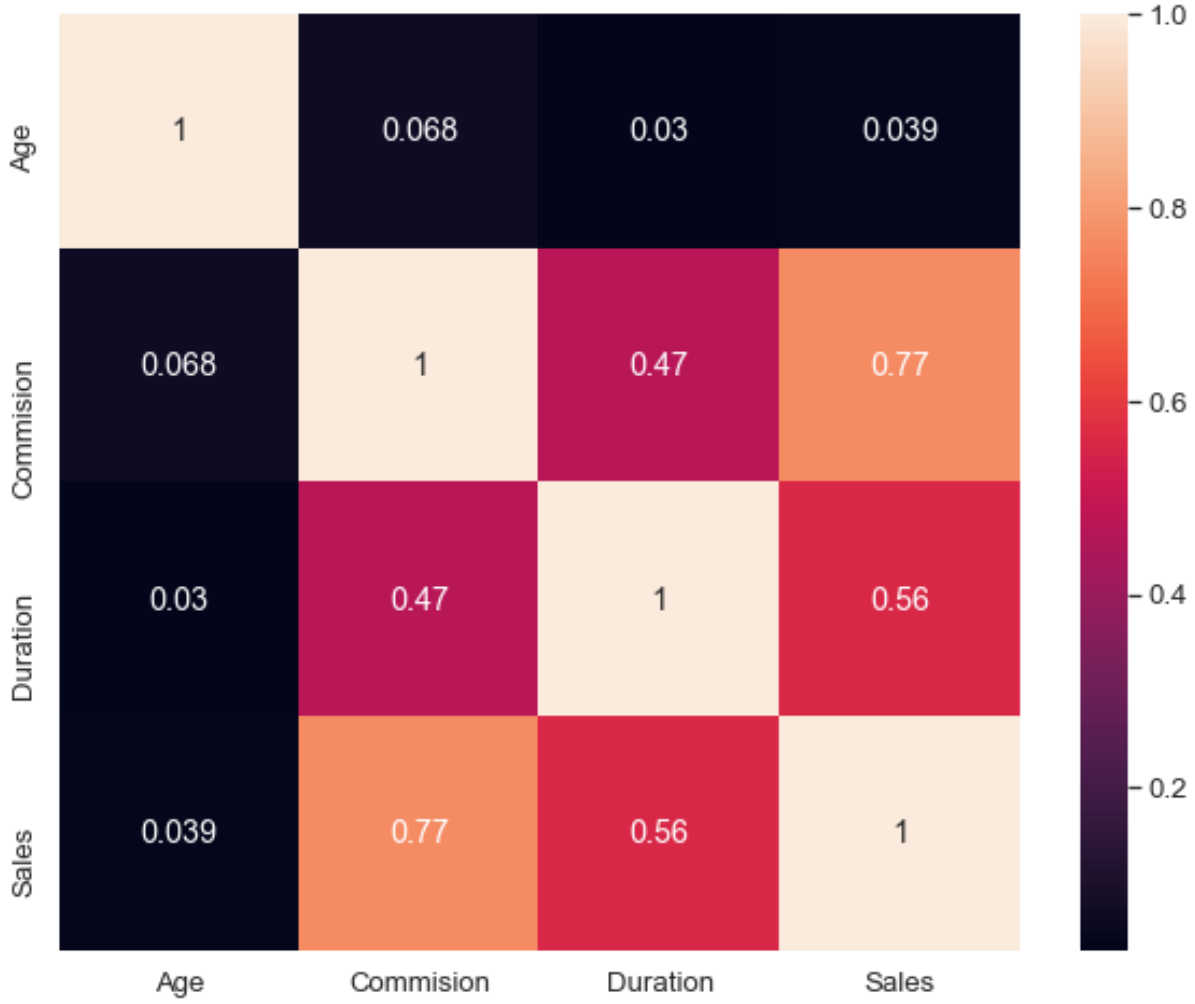

Fig 2.1.25

Converting all objects to categorical codes:

Below is the head of the dataset after all object type variables are converted into categorical codes.

```
<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
```

Table 2.1.2

Proportion of 1s and 0s:

```
0    0.692
1    0.308
Name: Claimed, dtype: float64
```

Table 2.1.3

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

To perform Train and Test, we must bifurcate data into dependent and independent variables.

'Claimed' is the target variable, hence this need to be dropped from original dataset which will be assigned to 'X' variable and dropped column will be assigned to another 'y' variable.

Once dataset is separated with target variable, data is splitted into training and testing for further analysis.

Decision Tree Classifier

Decision Trees (CART) 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.

For Decision Tree Classifier, Gini gain is used as criterion and by using grid search method we can identify best features (where the tree needs to be pruned).

After building Decision Tree model below is the best params displayed by using Grid search method.

```
{'criterion': 'gini', 'max_depth': 5.0, 'min_samples_leaf': 42, 'min_samples_split': 200}

DecisionTreeClassifier(max_depth=5.0, min_samples_leaf=42,
                       min_samples_split=200, random_state=1)
```

Below is the table of Variable Importance:

	Imp
Agency_Code	0.621974
Sales	0.257721
Product Name	0.057386
Commision	0.023406
Duration	0.023111
Age	0.016403
Type	0.000000
Channel	0.000000
Destination	0.000000

Table 2.2.1

Below is the table of Predicted Classes and Probs:

	0	1
0	0.887805	0.112195
1	0.432432	0.567568
2	0.432432	0.567568
3	0.208163	0.791837
4	0.937143	0.062857

Table 2.2.2

Random Forest Classifier

Random forest classifier creates a set of decision trees from randomly selected subset of training set. It then aggregates the votes from different decision trees to decide the final class of the test object.

For Random Forest Classifier, it is the estimator and by using grid search method we can identify best features.

After building Random Forest model below is the best params displayed by using Grid search method.

```
{'max_depth': 30, 'max_features': 7, 'min_samples_leaf': 50, 'min_samples_split': 150, 'n_estimator
s': 301}

RandomForestClassifier(max_depth=30, max_features=7, min_samples_leaf=50,
                       min_samples_split=150, n_estimators=301, random_state=1)
```

Below is the table of Variable Importance:

	Imp
Agency_Code	0.501401
Product Name	0.186148
Sales	0.181234
Commision	0.060033
Duration	0.032852
Age	0.027966
Type	0.008261
Destination	0.002105
Channel	0.000000

Table 2.2.3

Below is the table of Predicted Classes and Probs:

	0	1
0	0.755286	0.244714
1	0.537622	0.462378
2	0.635122	0.364878
3	0.286223	0.713777
4	0.920926	0.079074

Table 2.2.4

Artificial Neural Network

Artificial Neural networks (ANN) or neural networks are computational algorithms. It intended to simulate the behaviour of biological systems composed of “neurons”. ANNs are computational models inspired by an animal's central nervous systems. It is capable of machine learning as well as pattern recognition.

For ANN model, before model is built data need to be scaled and train data need to be fit and transformed whereas test data need to be only transformed.

After building ANN model below is the best params displayed by using Grid search method.

```
MLPClassifier(hidden_layer_sizes=200, max_iter=2500, random_state=1,
              solver='sgd', tol=0.01)
```

Below is the table of Predicted Classes and Probs:

	0	1
0	0.833619	0.166381
1	0.610337	0.389663
2	0.600408	0.399592
3	0.443018	0.556982
4	0.805135	0.194865

Table 2.2.5

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.

CART - AUC and ROC for the training data:

```
AUC: 0.825
[<matplotlib.lines.Line2D at 0x20db631b100>]
```

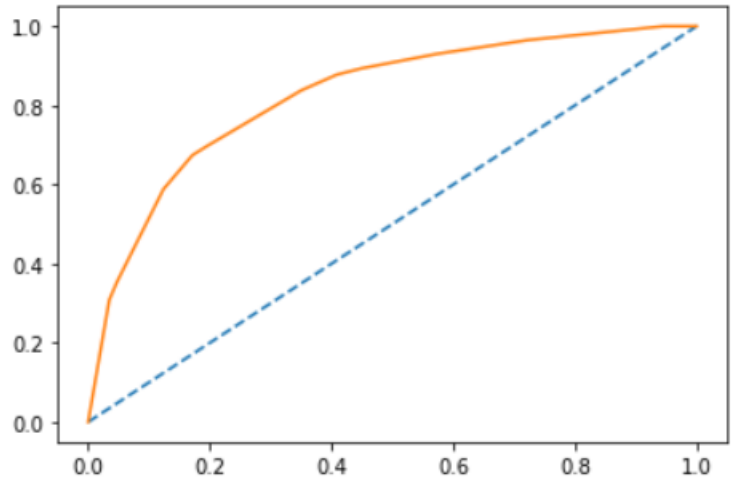

Fig 2.3.1

CART -AUC and ROC for the test data:

```
AUC: 0.792
[<matplotlib.lines.Line2D at 0x20db6402970>]
```

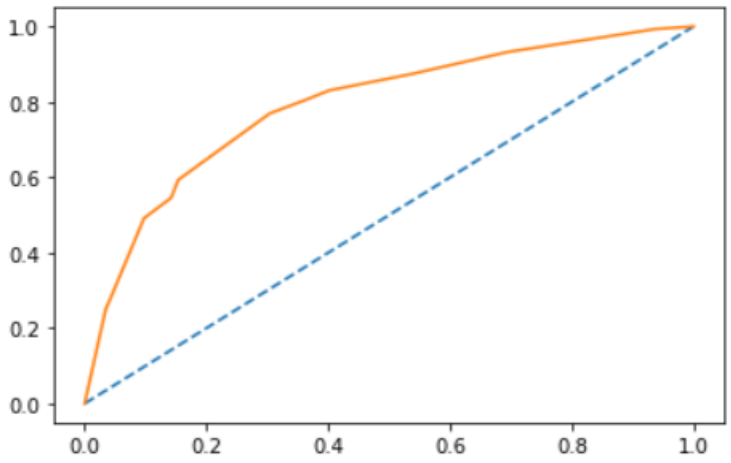

Fig 2.3.2

CART Confusion Matrix for training data:

```
array([[1289,  182],
       [ 259,  370]], dtype=int64)
```

CART Confusion Matrix for test data:

```
array([[546,  59],
       [150, 145]], dtype=int64)
```

CART Classification Report for the training data:

	precision	recall	f1-score	support
0	0.83	0.88	0.85	1471
1	0.67	0.59	0.63	629
accuracy			0.79	2100
macro avg	0.75	0.73	0.74	2100
weighted avg	0.78	0.79	0.79	2100

Table 2.3.1

CART Classification Report for the test data:

	precision	recall	f1-score	support
0	0.78	0.90	0.84	605
1	0.71	0.49	0.58	295
accuracy			0.77	900
macro avg	0.75	0.70	0.71	900
weighted avg	0.76	0.77	0.75	900

Table 2.3.2

CART metrics for training data:

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

CART metrics for test data:

```
cart_test_precision  0.71
cart_test_recall  0.49
cart_test_f1  0.58
```

CART training data accuracy:

```
0.79
```

CART test data accuracy:

```
0.7677777777777778
```

Decision Tree Model Conclusion:

Train Data:

- AUC: 83%
- Accuracy: 79%
- Precision: 67%
- f1-Score: 63%

Test Data:

- AUC: 79%
- Accuracy: 77%
- Precision: 71%
- f1-Score: 58%

Training and Test set results are almost similar, and with the overall measures high, the model is a good model.

RF - AUC and ROC for the training data:

```
Area under Curve is 0.8364490375127397
```

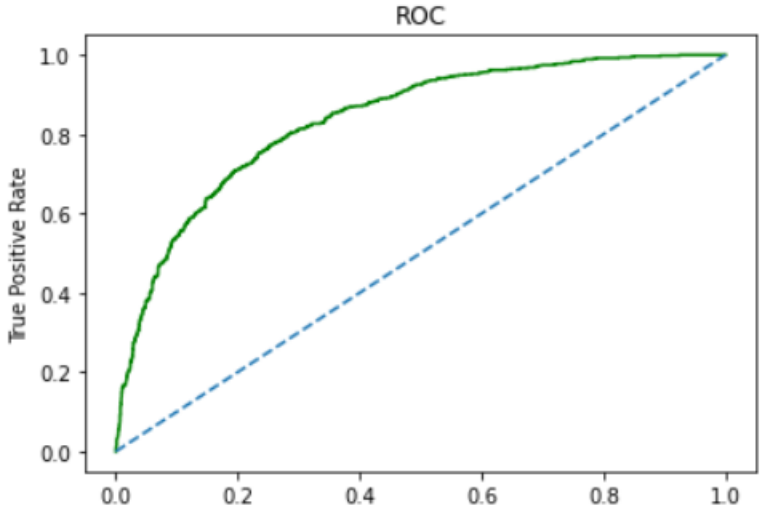

Fig 2.3.3

RF - AUC and ROC for the test data:

```
Area under Curve is 0.8161283092870151
```

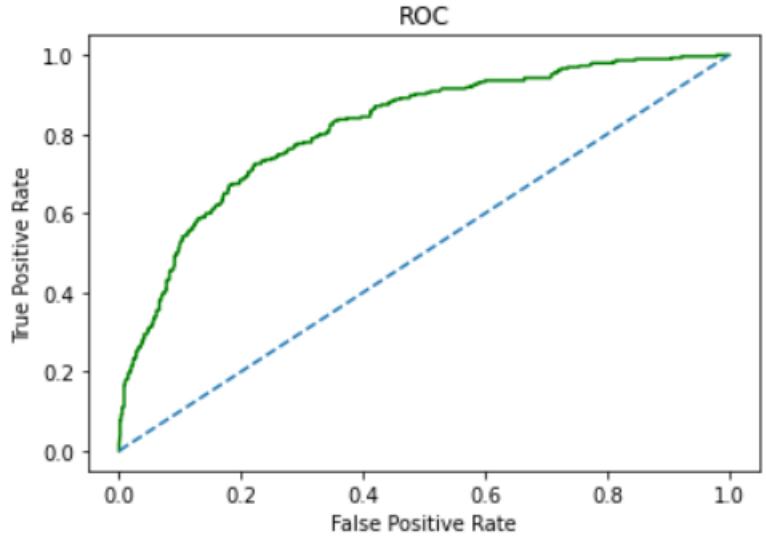

Fig 2.3.4

RF Confusion Matrix for training data:

```
array([[1309,  162],
       [ 278,  351]], dtype=int64)
```

RF Confusion Matrix for test data:

```
array([[553,  52],
       [160, 135]], dtype=int64)
```

RF Classification Report for the training data:

	precision	recall	f1-score	support
0	0.82	0.89	0.86	1471
1	0.68	0.56	0.61	629
accuracy			0.79	2100
macro avg	0.75	0.72	0.74	2100
weighted avg	0.78	0.79	0.78	2100

Table 2.3.3

RF Classification Report for the test data:

	precision	recall	f1-score	support
0	0.78	0.91	0.84	605
1	0.72	0.46	0.56	295
accuracy			0.76	900
macro avg	0.75	0.69	0.70	900
weighted avg	0.76	0.76	0.75	900

Table 2.3.4

RF metrics for training data:

```
rf_train_precision  0.68
rf_train_recall  0.56
rf_train_f1  0.61
```

RF metrics for test data:

```
rf_test_precision  0.72
rf_test_recall  0.46
rf_test_f1  0.56
```

RF training data accuracy:

```
0.7904761904761904
```

RF test data accuracy:

```
0.7644444444444445
```

Random Forest Model Conclusion:

Train Data:

- AUC: 84%
- Accuracy: 79%
- Precision: 68%
- f1-Score: 61%

Test Data:

- AUC: 82%
- Accuracy: 76%
- Precision: 72%
- f1-Score: 56

Training and Test set results are almost similar, and with the overall measures high, the model is a good model.

ANN - AUC and ROC for the training data:

```
Area under Curve is 0.7837702740529948
```

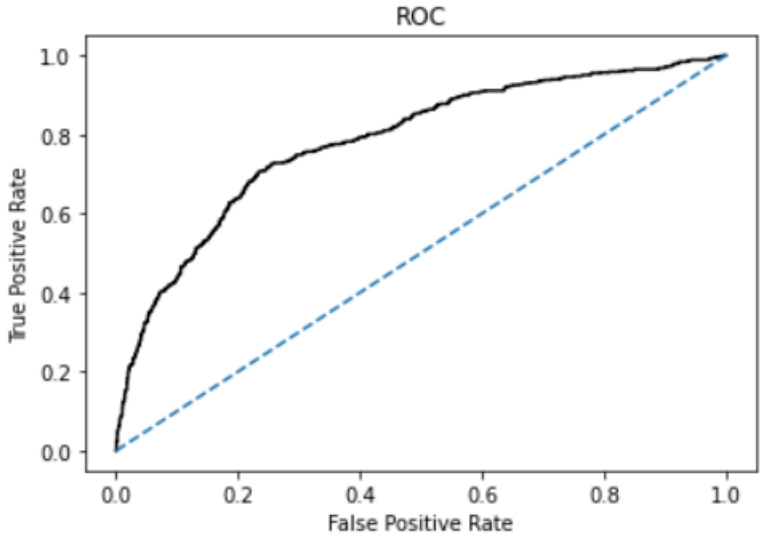

Fig 2.3.5

ANN - AUC and ROC for the test data:

```
Area under Curve is 0.7449642807115843
```

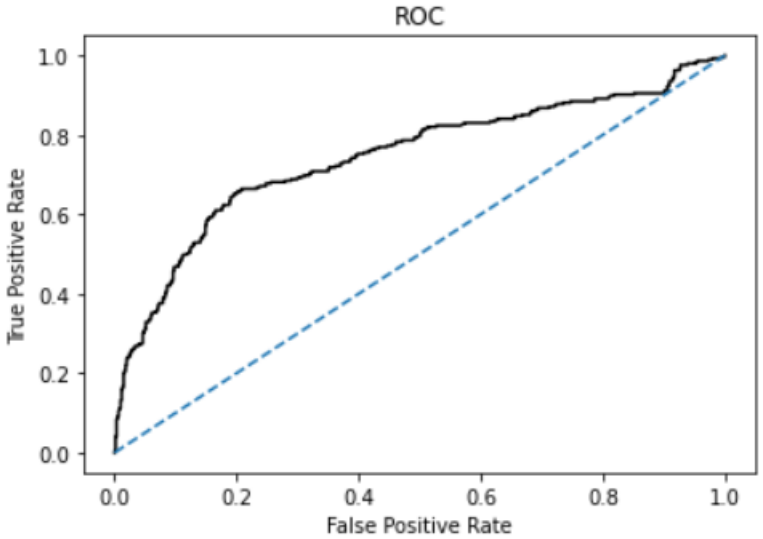

Fig 2.3.6

ANN Confusion Matrix for training data:

```
array([[1378,   93],
       [ 395,  234]], dtype=int64)
```

ANN Confusion Matrix for test data:

```
array([[575,  30],
       [203,  92]], dtype=int64)
```

ANN Classification Report for the training data:

	precision	recall	f1-score	support
0	0.78	0.94	0.85	1471
1	0.72	0.37	0.49	629
accuracy			0.77	2100
macro avg	0.75	0.65	0.67	2100
weighted avg	0.76	0.77	0.74	2100

Table 2.3.5

ANN Classification Report for the test data:

	precision	recall	f1-score	support
0	0.74	0.95	0.83	605
1	0.75	0.31	0.44	295
accuracy			0.74	900
macro avg	0.75	0.63	0.64	900
weighted avg	0.74	0.74	0.70	900

Table 2.3.6

ANN metrics for training data:

```
nn_train_precision  0.72
nn_train_recall  0.37
nn_train_f1  0.49
```

ANN metrics for test data:

```
nn_test_precision  0.75
nn_test_recall  0.31
nn_test_f1  0.44
```

ANN training data accuracy:

```
0.7676190476190476
```

ANN test data accuracy:

```
0.7411111111111112
```

Artificial Neural Network Conclusion:

Train Data:

- AUC: 78%
- Accuracy: 77%
- Precision: 72%
- f1-Score: 49%

Test Data:

- AUC: 74%
- Accuracy: 74%
- Precision: 75%
- f1-Score: 44%

Training and Test set results are almost similar, and with the overall measures high, the model is a good model.

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

Table comparing all models:

	CART Train	CART Test	Random Forest Train	Random Forest Test	Neural Network Train	Neural Network Test
Accuracy	0.79	0.77	0.79	0.76	0.77	0.74
AUC	0.83	0.79	0.84	0.82	0.78	0.74
Recall	0.59	0.49	0.56	0.46	0.37	0.31
Precision	0.67	0.71	0.68	0.72	0.72	0.75
F1 Score	0.63	0.58	0.61	0.56	0.49	0.44

Table 2.4.1

ROC Curve for the 3 models on the Training data:

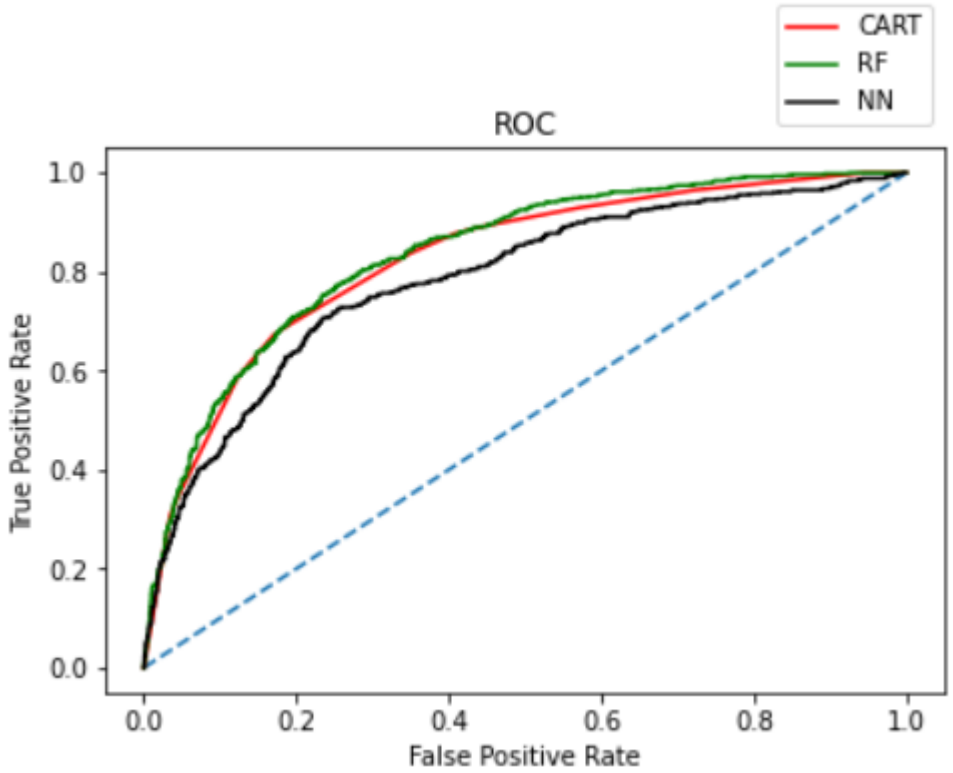

Fig 2.4.1

ROC Curve for the 3 models on the Test data:

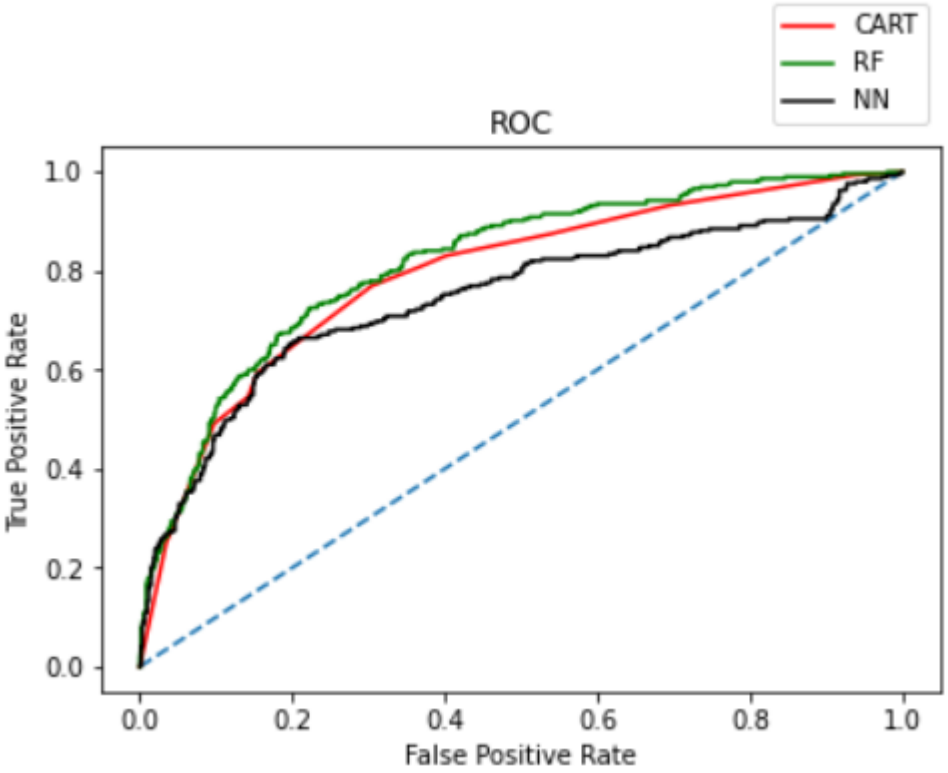

Fig 2.4.2

Conclusion:

I am selecting the RF model, as it has better accuracy, precision, recall, f1 score better than other two models CART & ANN.

2.5 Inference: Based on the whole Analysis, what are the 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, 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.

----------END----------