Project – Data Mining

Contents:

Problem 1: Clustering

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

Data Dictionary for Market Segmentation:

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

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

Head of the data set:

	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

Data types:

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

Information about data set:

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

Observation:

- 210 entries and 7 variables are found.
- No missing value found.
- All variable is float type.

Checking For Missing value:

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

Observation: No missing values.

Summary of the Data

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

Observation:

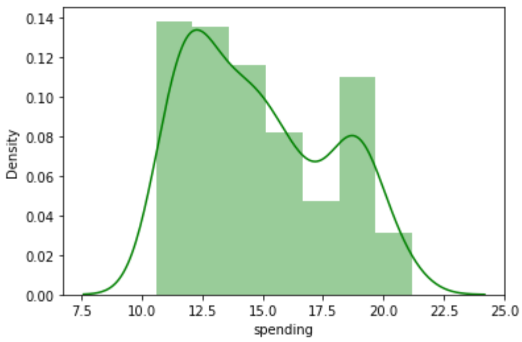
-Standard deviation is high for spending variable

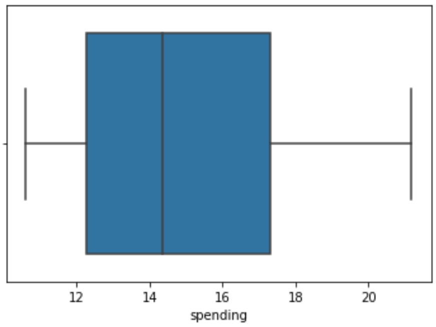
-Most of the variable distributed evenly

Univariate Analysis

Summary of spending
count   210.000000
mean   14.847524
std   2.909699
min   10.590000
25%   12.270000
50%   14.355000
75%   17.305000
max   21.180000
Name: spending, dtype: float64
median ------- 14.355
Null value ---- False
Skew ---------- 0.3998891917177586

Distribution Plot of spending   Box Plot of spending





No outlier is found in this variable
The Spending variable is positively skewed

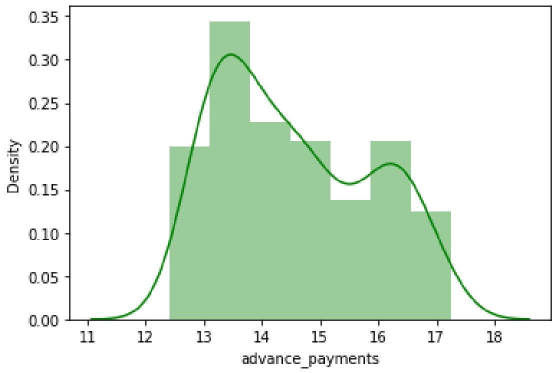
Summary of advance_payments

count 210.000000
mean 14.559286
std 1.305959
min 12.410000
max 17.250000
Name: advance_payments, dtype: float64
median ------- 14.32
Nullvalue ---- False
Skew ---------- 0.3865727731912213

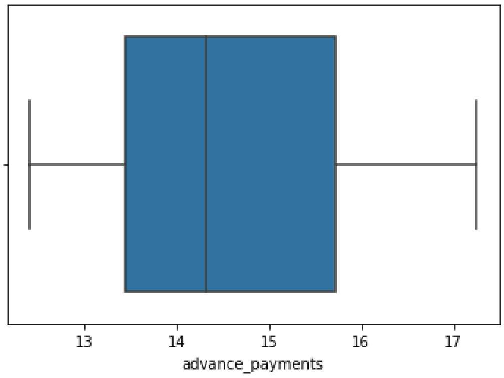
-No outlier is found in this variable

-The **advance_payments** variable is positively skewed

Distribution Plot of advance_payments



Box Plot of advance_payments



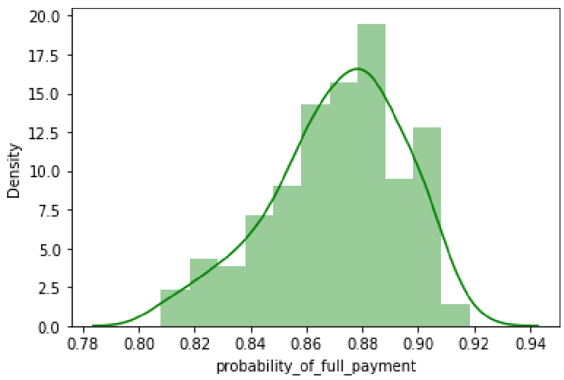
Summary of probability_of_full_payment

count 210.000000
mean 0.870999
std 0.023629
min 0.808100
max 0.918300
Name: probability_of_full_payment, dtype: float64
median ------- 0.8734500000000001
Nullvalue ---- False
Skew ---------- -0.5379537283982823

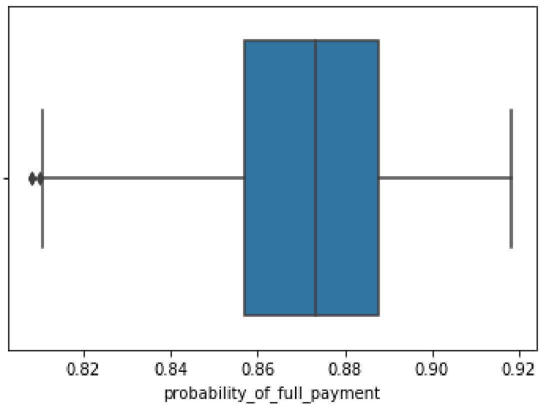
-Outlier is found in this variable.

-probability_of_full_payment variable is negatively skewed

Distribution Plot of probability_of_full_payment



Box Plot of probability_of_full_payment



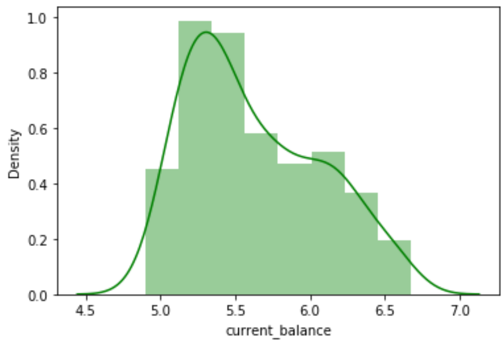
Summary of current_balance

count   210.000000
mean   5.628533
std   0.443063
min   4.899000
max   6.675000
Name: current_balance, dtype: float64
median ------- 5.5235
Nullvalue ---- False
Skew ---------- 0.5254815601318906

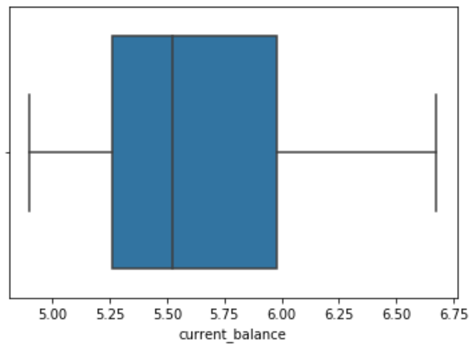
-No outlier is found in this variable

-The **current_balance** variable is positively ske wed

Distribution Plot of current_balance



Box Plot of current_balance



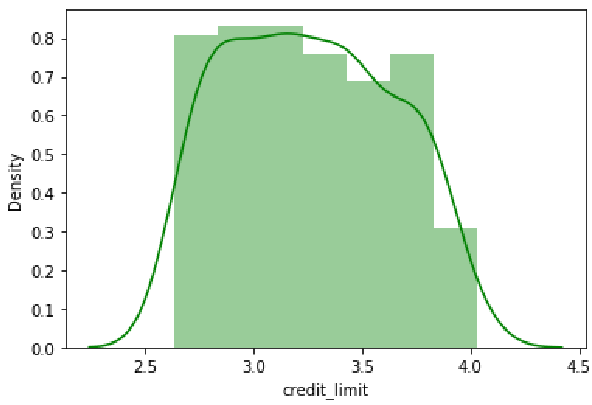
Summary of credit_limit

count   210.000000
mean   3.258605
std   0.377714
min   2.630000
max   4.033000
Name: credit_limit, dtype: float64
median ------- 3.237
Nullvalue ---- False
Skew ---------- 0.1343782451316215

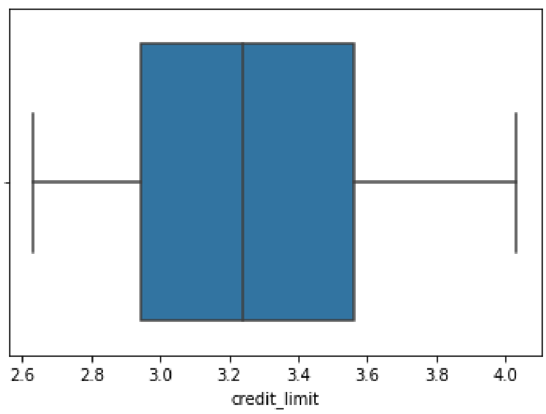
-No outlier is found in this variable

-The **credit_limit** variable is positively skewe

Distribution Plot of credit_limit



Box Plot of credit_limit



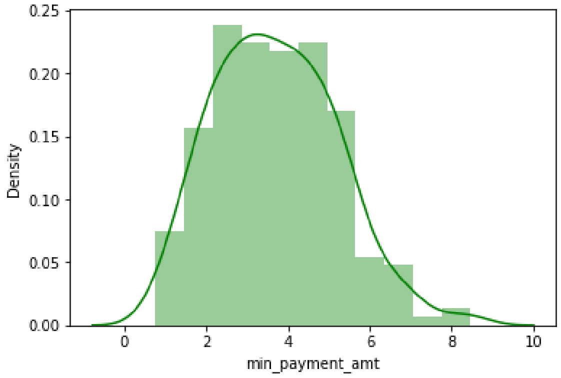
Summary of min_payment_amt

count   210.000000
mean   3.700201
std   1.503557
min   0.765100
max   8.456000
Name: min_payment_amt, dtype: float64
median ------- 3.599
Nullvalue ---- False
Skew ---------- 0.40166734329025183

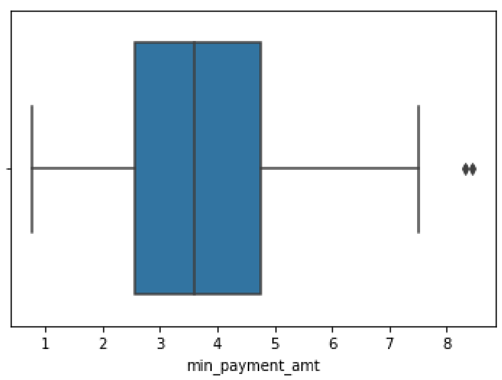
-Outlier is found in this variable

-The **min_payment_amt** variable is positively s kewed

Distribution Plot of min_payment_amt



Box Plot of min_payment_amt



Summary of max_spent_in_single_shopping

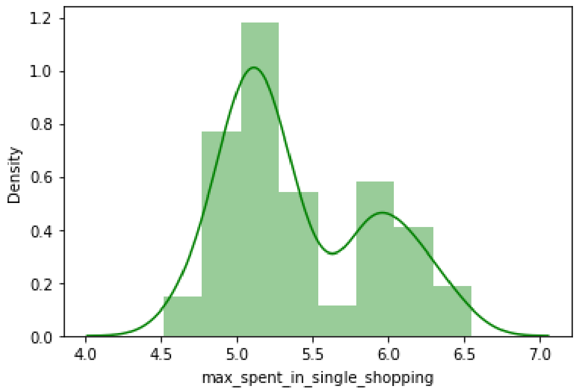
count   210.000000
mean   5.408071
std   0.491480
min   4.519000
max   6.550000
Name: max_spent_in_single_shopping, dtype: float64
median ------- 5.223000000000001
Nullvalue ---- False
Skew ---------- 0.561897374954866

-No outlier is found in this variable

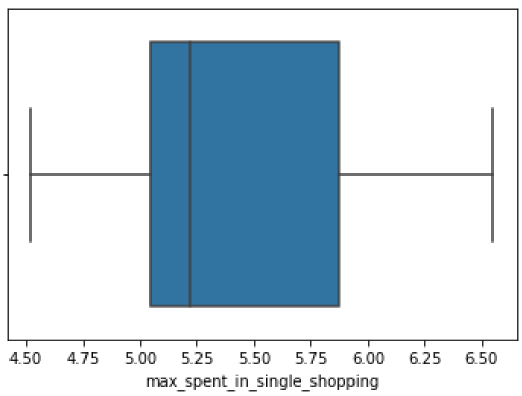
-The **max_spent_in_single_shopping** variable is positively skewed

---

Distribution Plot of max_spent_in_single_shopping



Box Plot of max_spent_in_single_shopping



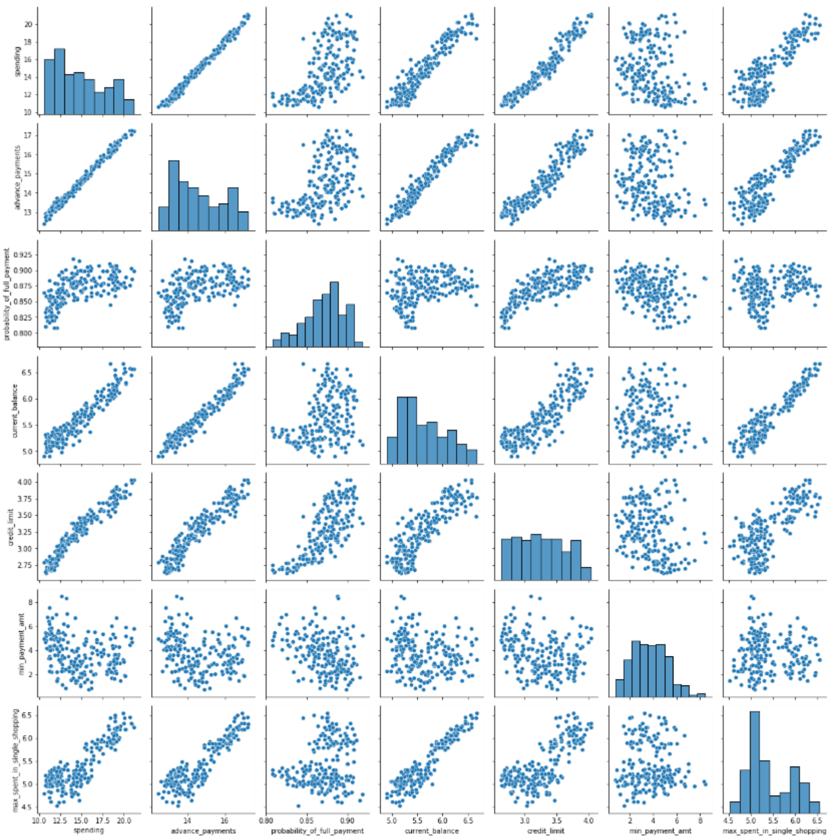
Multivariate Analysis

Heat Map:

Positive correlation established between

1) spending & advance_payments
2) spending & advance_payments
3) credit_limit & spending
4) spending & current_balance
5) max_spent_in_single_shopping &current_balance

Pair plot:

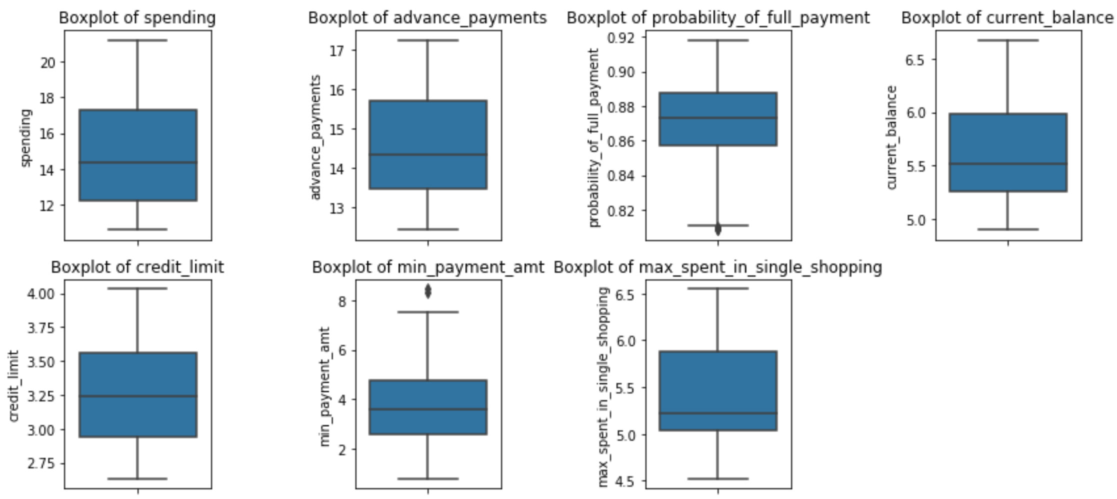


Skewness:

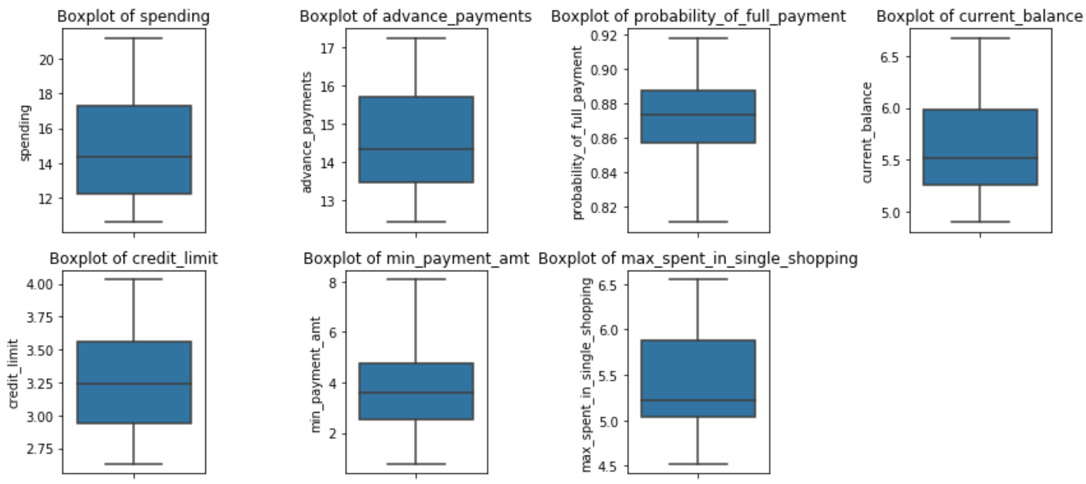
```
<bound method Series.sort_values of
spending                       0.399889
advance_payments               0.386573
probability_of_full_payment   -0.537954
current_balance                0.525482
credit_limit                   0.134378
min_payment_amt                0.401667
max_spent_in_single_shopping   0.561897
dtype: float64>
```

Probability_of_full_payment variable is negatively skewed and remaining all are posi tively skewed.

Outlier:



After treating outlier:



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

Solution:

The model such as works under the principle on the distance-based computations needs necessary scaling.

It is done for values of the variables are in different scales. In the particular dataset spending, advance_payments are in different values and this may get more weightage.

So scaling is necessary for clustering in this particular case.

I have used z-score to standardised the data to relative same scale -3 to +3.

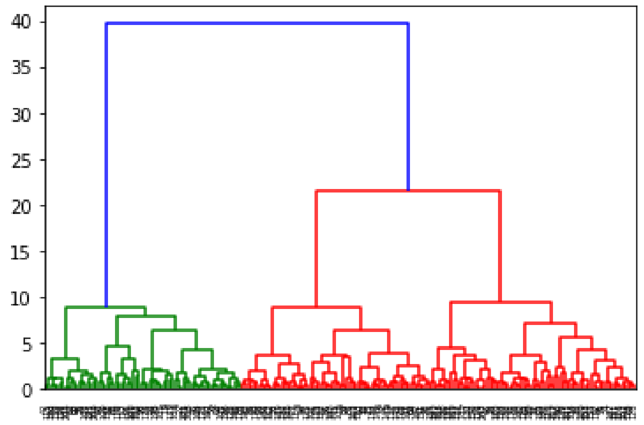
Data after Scaling:

	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
0	1.754355	1.811968	0.177628	2.367533	1.338579	-0.298625	2.328998
1	0.393582	0.253840	1.505071	-0.600744	0.858236	-0.242292	-0.538582
2	1.413300	1.428192	0.505234	1.401485	1.317348	-0.220832	1.509107
3	-1.384034	-1.227533	-2.571391	-0.793049	-1.639017	0.995699	-0.454961
4	1.082581	0.998364	1.198738	0.591544	1.155464	-1.092656	0.874813

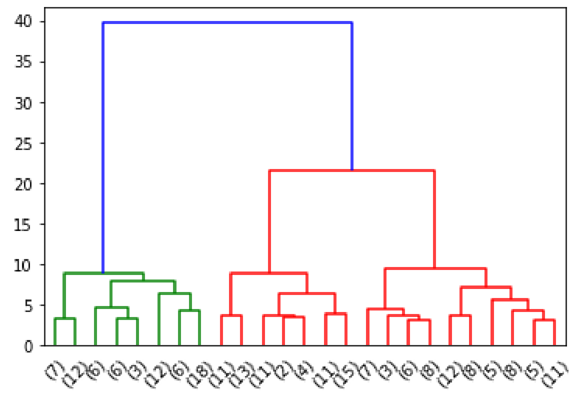
1.2 Apply hierarchical clustering to scaled data. Identify the number of optimum clusters using Dendrogram and briefly describe them

Hierarchical Clustering

Performing Hierarchical Clustering with the **Ward's linkage** method and plot the dendrogram:



Ploting the truncated dendrogram with the last 25 clusters:



Identify the number of clusters based on the dendrogram and add the cluster numbers to the original dataframe.

Importing fcluster module to create clusters

Setting criterion as maxclust, then create 3 clusters, and store the result in another object:

```
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)
```

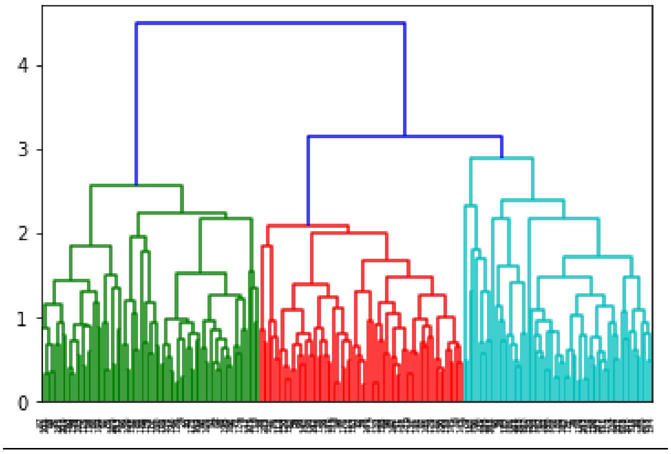
Adding the clusters to original dataframe:

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

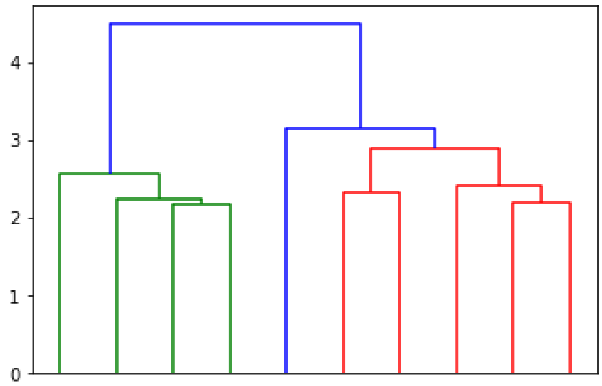
Cluster Frequency

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

<u>Performing Hierarchical Clustering with the average linkage method and plot the dendrogram:</u>



Ploting the truncated dendrogram with the last 25 clusters.



Identify the number of clusters based on the dendrogram and add the cluster numbers to the original dataframe.

Importing fcluster module to create clusters

<u>Set criterion as maxclust, then create 3 clusters, and store the result in another object:</u>

```
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)
```

Adding the clusters to original dataframe:

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

Cluster Frequency:

1   75
2   70
3   65
Name: Clusters, dtype: int64

Creating 3 cluster group table of Hierarchical Clustering:

Clusters	1	2	3
spending	18.1	11.9	14.2
advance_payments	16.1	13.3	14.2
probability_of_full_payment	0.9	0.8	0.9
current_balance	6.1	5.3	5.4
credit_limit	3.6	2.8	3.3
min_payment_amt	3.7	4.6	2.8
max_spent_in_single_shopping	6.0	5.1	5.1

Almost same mean and minor variation seen in both Ward linkage method and Average linkage method of hierarchical clustering

Agglomerative Hierarchical Clustering

Setting number of clusters is 3 , affinity is euclidean, linkage as average and store the result in another object

[1 0 1 2 1 0 2 2 1 2 1 1 2 1 0 0 0 2 2 2 2 2 1 2 0 1 0 2 2 2 2 2 2 0 2 2 2
 2 2 1 1 0 1 1 2 2 0 1 1 1 2 1 1 1 1 1 2 2 2 1 0 2 2 1 0 1 1 0 1 2 0 2 1 1
 2 1 0 2 1 0 0 0 0 1 2 1 1 1 1 0 0 1 0 2 2 1 1 1 2 1 0 1 0 1 0 1 1 2 0 1 1
 0 1 2 2 1 0 0 2 1 0 2 2 2 0 0 1 2 0 0 2 0 0 1 2 1 1 2 1 0 0 0 2 2 2 2 1 2
 0 2 0 2 0 1 0 0 2 2 0 1 1 2 1 1 1 2 1 0 0 2 0 2 0 1 1 1 0 2 0 2 0 2 0 0 1
 1 0 1 0 2 0 0 2 1 0 1 1 2 1 2 0 0 0 2 1 0 1 0 0 1]

Adding the Agglomerative clusters to original dataframe:

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

Agglomerative Cluster Frequency

0   65
1   75
2   70
Name: Agglo_CLusters, dtype: int64

Creating 3 cluster group table of Agglomerative **Clustering** :

Agglo_CLusters	0	1	2
spending	14.2	18.1	11.9
advance_payments	14.2	16.1	13.3
probability_of_full_payment	0.9	0.9	0.8
current_balance	5.4	6.1	5.3
credit_limit	3.3	3.6	2.8
min_payment_amt	2.8	3.7	4.6
max_spent_in_single_shopping	5.1	6.0	5.1

Hierarchical clustering through Ward linkage and Average linkage has almost equal values compare to Agglomerative Hierarchical Cluster

Observation from Hierarchical clustering and Agglomerative Clustering:

Both Hierarchical clustering and Agglomerative Cluster giving almost same value

Almost same mean and minor variation seen in both methods of hierarchical clustering

On the dendrogram 3 group clustering looks good. So further analysis did based on 3 group cluster solution based on the hierarchical clustering

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.

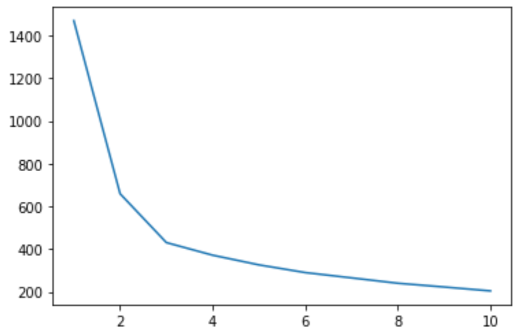
Fitting the scaled data to KMeans and Calculate the Inertia value for few random cluster value:

(n_clusters = 2) 659.1474009548498
(n_clusters = 3) 430.29848175122294
(n_clusters = 4) 370.86859623942064
(n_clusters = 5) 327.5507168609346

Creating loop for Within Sum of Squares (WSS) from 1 to 11 which help to find optimum number and draw Elbow curve:

```
[1470.0,
 659.1474009548498,
 430.29848175122294,
 371.2217639268479,
 325.85307021907124,
 289.7773106221679,
 264.89633867403785,
 239.75942734996403,
 222.1535317260895,
 203.86939776234152]
```

Plotting Elbow Curve:



Elbow curve showing that after 3 cluster there is no drop in the values

Cheking silhouette score for 3 clustering:

Labels:

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

Silhouette score: 0.40080592215222155

Silhouette score for 3 cluster is good and Elbow curve showing that after 3 cluster there is no drop in the values, so we select 3 cluster groups.

Adding the KMeans clusters to original dataframe: **(KMeans clusters=Clus_kmeans)**

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

Note: Hierarchical and Agglomerative Cluster retained in this table to compare with with KMeans Clusters

Adding the silhouette samples width clusters to dataframe:

	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	Clusters	Clus_kmeans	sil_width
0	19.94	16.92	0.875200	6.675	3.763	3.252	6.550	1	2	0.573278
1	15.99	14.89	0.906400	5.363	3.582	3.336	5.144	3	1	0.365564
2	18.95	16.42	0.882900	6.248	3.755	3.368	6.148	1	2	0.637092
3	10.83	12.96	0.810588	5.278	2.641	5.182	5.185	2	0	0.515595
4	17.99	15.86	0.899200	5.890	3.694	2.068	5.837	1	2	0.360972

Frequency of KMean Clusters:

0 72
1 71
2 67
dtype: int64

Creating 3 cluster group table of KMeans Clustering:

Clus_kmeans	0	1	2
spending	14.4	11.9	18.5
advance_payments	14.3	13.2	16.2
probability_of_full_payment	0.9	0.8	0.9
current_balance	5.5	5.2	6.2
credit_limit	3.3	2.8	3.7
min_payment_amt	2.7	4.7	3.6
max_spent_in_single_shopping	5.1	5.1	6.0

Observation in K Mean Clustering:

-The silhouette score seems to very less indicates all the data points are properly clustered to the clusters.

-We consider the optimal number as 3 after there is no huge drop in inertia value of 3 clustering. It also graphically shown in Elbow Curve

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

3 group cluster via hierarchical clustering

Clusters	1	2	3
spending	18.1	11.9	14.2
advance_payments	16.1	13.3	14.2
probability_of_full_payment	0.9	0.8	0.9
current_balance	6.1	5.3	5.4
credit_limit	3.6	2.8	3.3
min_payment_amt	3.7	4.6	2.8
max_spent_in_single_shopping	6.0	5.1	5.1

3 group cluster via Kmeans

Clus_kmeans	0	1	2
spending	14.4	11.9	18.5
advance_payments	14.3	13.2	16.2
probability_of_full_payment	0.9	0.8	0.9
current_balance	5.5	5.2	6.2
credit_limit	3.3	2.8	3.7
min_payment_amt	2.7	4.7	3.6
max_spent_in_single_shopping	5.1	5.1	6.0

There are 3 types of groups in hierarchical clustering

-Cluster 1 = Group 1: High Spending Group
-Cluster 3 = Group 2: Medium Spending Group
-Cluster 2 = Group 3: Low Spending Group

There are 3 types of groups in KMeans Clustering

-Cluster 0 = Group 1: high spending Group
-Cluster 2 = Group 2: medium spending Group
-Cluster 1 = Group 3: low spending Group

Cluster Group Profiles:

Group 1: High Spending
Group 2: Medium Spending
Group 3: Low Spending

Group 1: High Spending Group

- Providing loan for customers with good repayment record.

- Increase their credit limit

- Giving any reward points might increase their purchases.

- Offering special and discounts for certain limit to increase their purchases.

- Sending Emails about new arrivals and outgoing discount.

Group 2: Medium Spending Group

- Bundle the complimentary goods and make it available at discount price

- More discount provided during month end (salary day)

- Products with less price and good quality should be made available in the selection.

-Providing reward points and loyalty points for purchases

Group 3: Low Spending Group

-Increase their spending habits by tying up with electricity, grocery stores, utilities etc

- Offers can be provided on early payments to improve their payment rate.

-Customers should be given remainders for payments.

Problem 2: CART-RF-ANN

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

Attribute Information:

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

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

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

Checking for missing value

Age	0
Agency_Code	0
Type	0
Claimed	0
Commision	0
Channel	0
Duration	0
Sales	0
Product Name	0
Destination	0

No Missig values.

Information of the data:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3000 entries, 0 to 2999
Data columns (total 9 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   Commision     3000 non-null   float64
 4   Channel       3000 non-null   int8
 5   Duration      3000 non-null   int64
 6   Sales         3000 non-null   float64
 7   Product Name  3000 non-null   int8
 8   Destination   3000 non-null   int8
dtypes: float64(2), int64(2), int8(5)
memory usage: 108.5 KB
```

- One target variable - Clamied and total 9 independant variable
- Age, Commision, Duration, Sales are numeric variable
- Agency_Code, Type, Claimed, Channel, Product Name, Destination are categorial variables

Checking for duplicates

Total duplicates = 139

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

139 rows × 10 columns

Proportion of 1s and 0s in target variable:

```
No     0.692
Yes    0.308
Name: Claimed, dtype: float64
```

Univariate Analysis

Summary of Data

	count	mean	std	min	25%	50%	75%	max
Age	3000.0	38.091000	10.463518	8.0	32.0	36.00	42.000	84.00
Commision	3000.0	14.529203	25.481455	0.0	0.0	4.63	17.235	210.21
Duration	3000.0	70.001333	134.053313	-1.0	11.0	26.50	63.000	4580.00
Sales	3000.0	60.249913	70.733954	0.0	20.0	33.00	69.000	539.00

Observation:

- Commission variable more than 50% only.

- Minimum Duration is -1, and which is not possible.

For Numerical Value

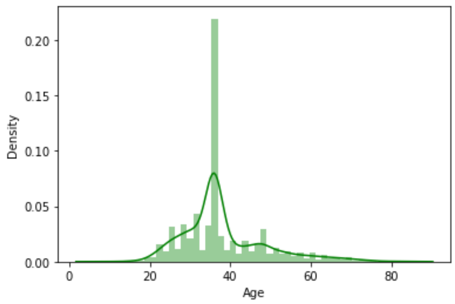
Summary of Age

count 3000.000000
mean 38.091000
std 10.463518
min 8.000000
max 84.000000
Name: Age, dtype: float64
median ------- 36.0
Nullvalue ---- False
Skew ---------- 1.149712770495169

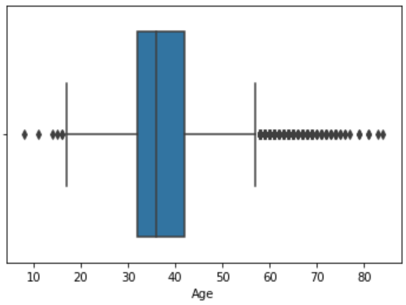
-Outliers found in **Age** variable

- The **Age** variable positively skewed.

Distribution Plot of Age



Box Plot of Age



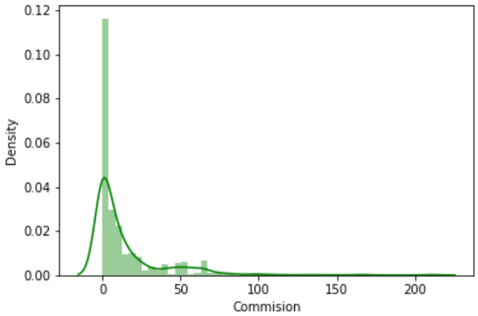
Summary of Commission

count   3000.000000
mean   14.529203
std   25.481455
min   0.000000
max   210.210000
Name: Commission, dtype: float64
median ------- 4.63
Nullvalue ---- False
Skew ---------- 3.148857772356885

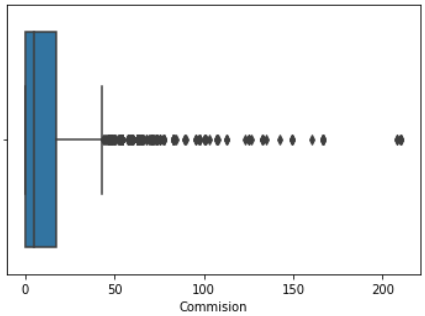
-Outliers found in **Commission** variable

- The **Commission** variable positively skewed.

Distribution Plot of Commission



Box Plot of Commission



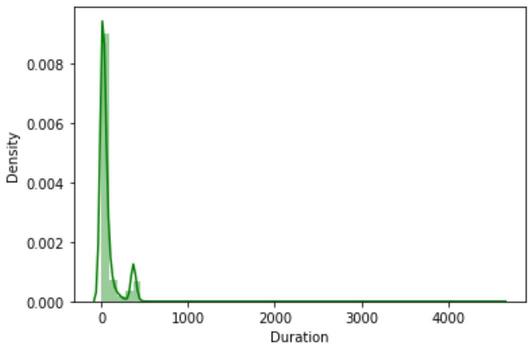
Summary of Duration

count   3000.000000
mean   70.001333
std   134.053313
min   -1.000000
max   4580.000000
Name: Duration, dtype: float64
median ------- 26.5
Nullvalue ---- False
Skew ---------- 13.784681027519602

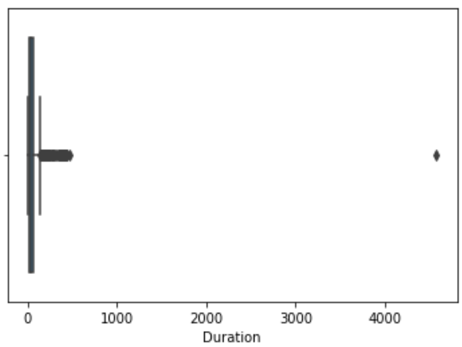
-Outliers found in **Duration** variable

- The **Duration** variable positively skewed.

Distribution Plot of Duration



Box Plot of Duration

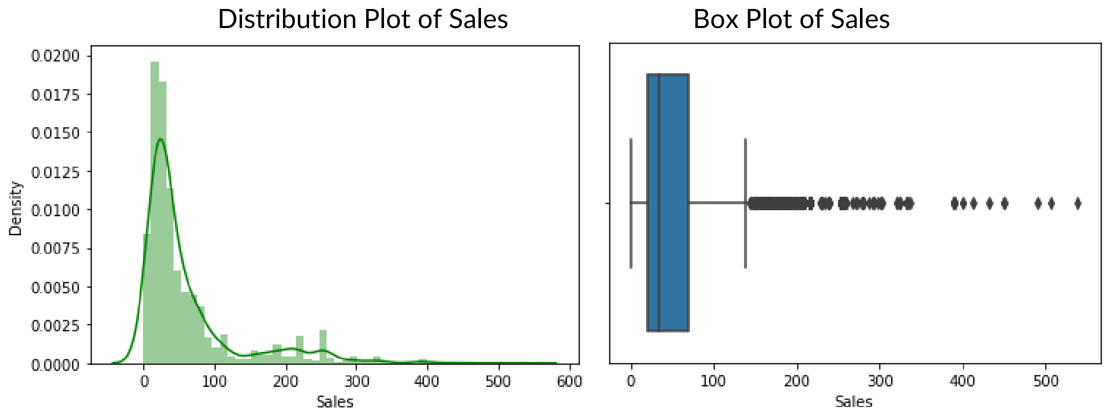


Summary of Sales

count 3000.000000
mean 60.249913
std 70.733954
min 0.000000
max 539.000000
Name: Sales, dtype: float64
median ------- 33.0
Nullvalue ---- False
Skew ---------- 2.381148461687274

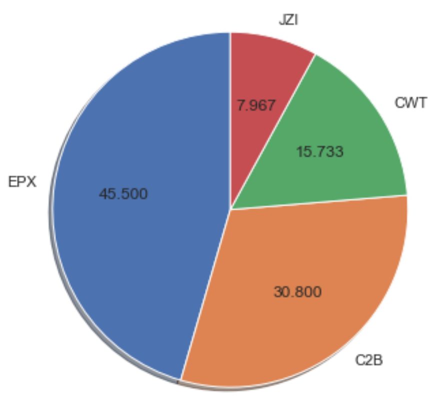
-Outliers found in **Sales** variable

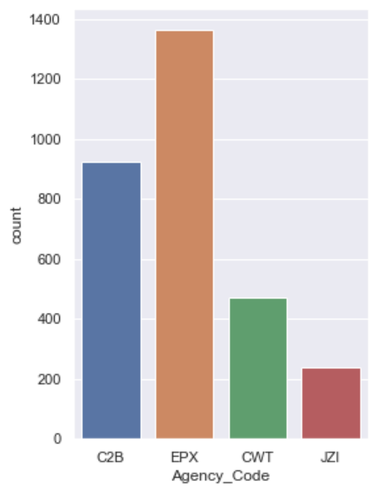
- The **Sales** variable positively skewed.



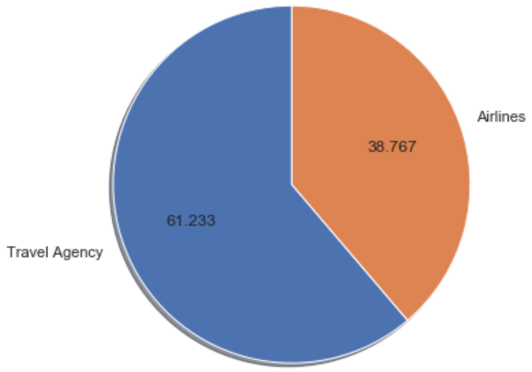
For Categorical Value

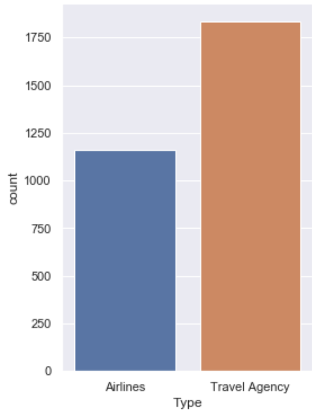
Agency_Code



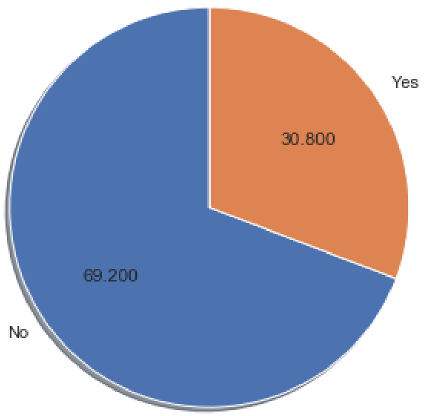


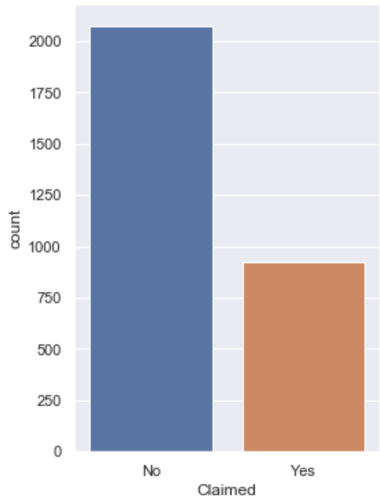
Type



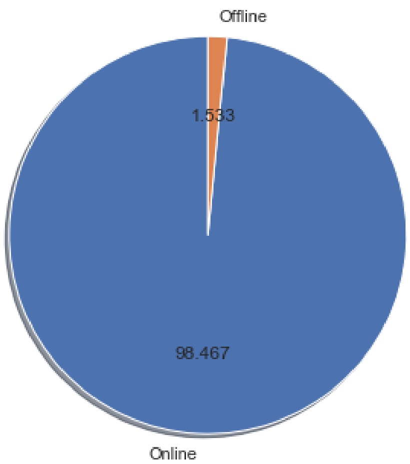


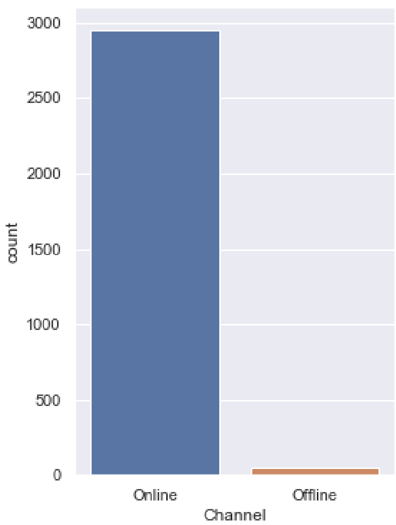
Claimed



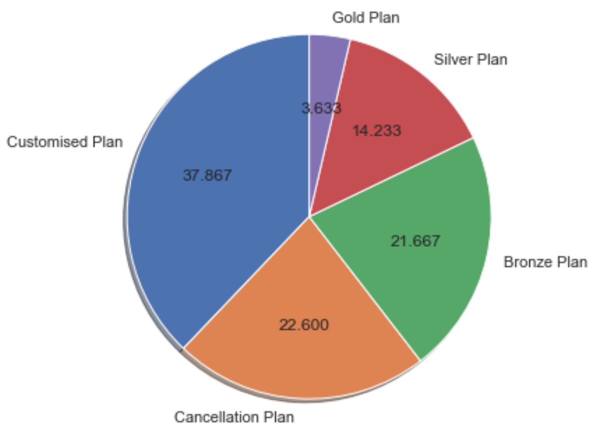


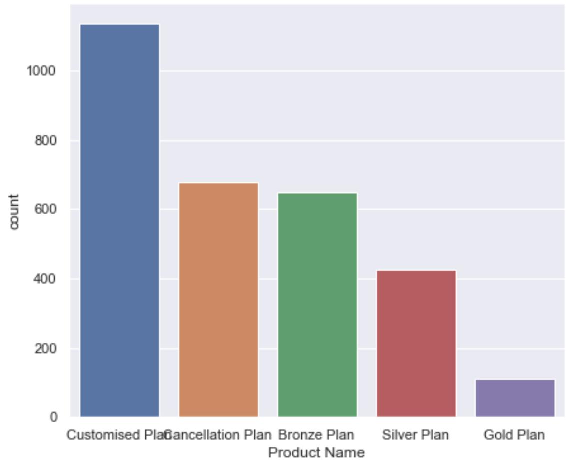
Channel





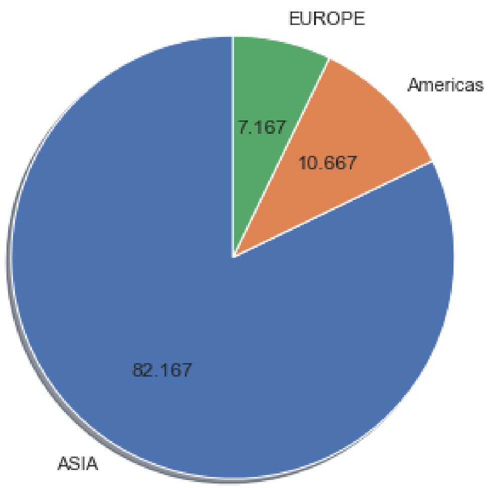
Product Name

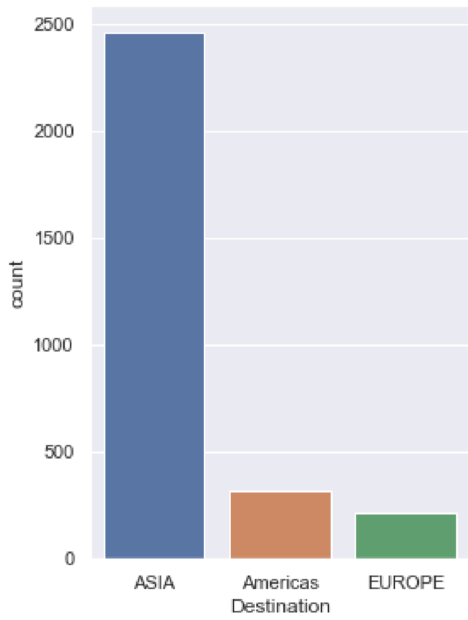




- Customized plan is high demand and gold plan very less in demand

Destination

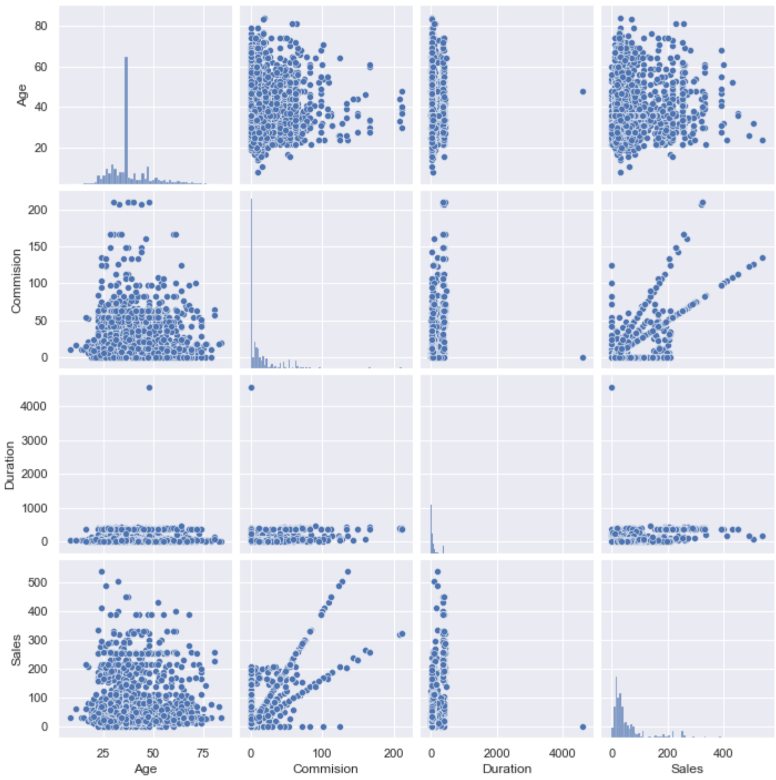




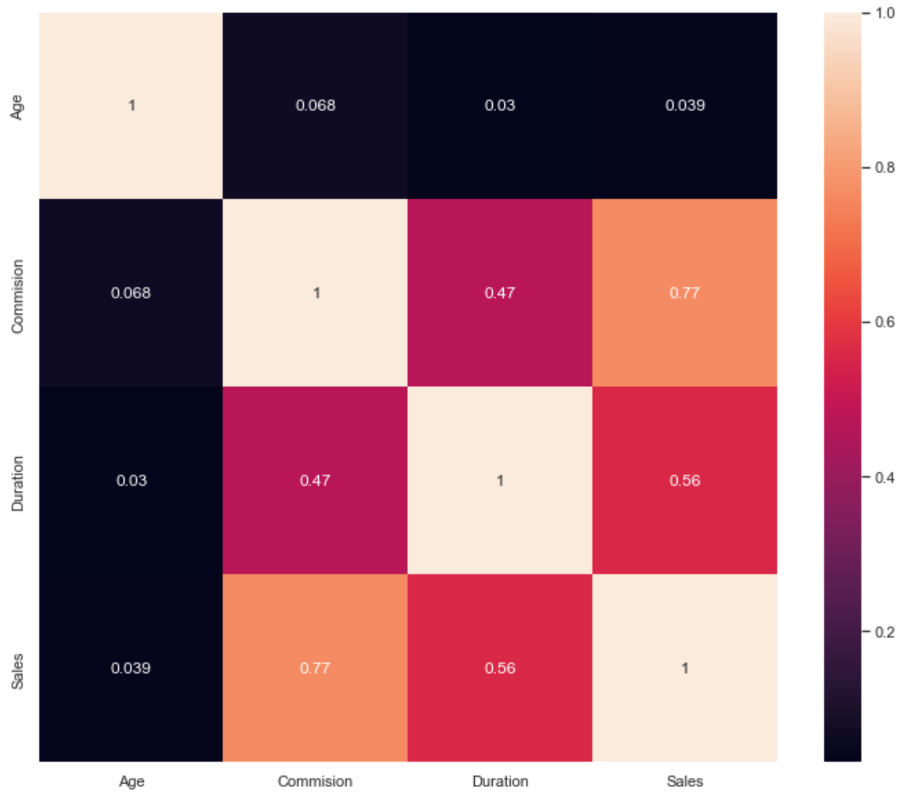
- Most favourite is Asia.
- America's place and Europe comes as second priority

Multivariate analysis:

Pair wise distribution of the continuous variables



Checking for Correlations using heatmap



Observations:

No Negative correlations found

There are no such correlations seen in this data

Converting all objects to categorical codes

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

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

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

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

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

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

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

Building Decision Tree Classifier

Step 1: Extract the target column into separate vectors for training set and test set.

Step 2: Splitting data into training and test set for independent attributes.

Step3: Search for optimal grid.

```
GridSearchCV(cv=3, error_score=nan,
        estimator=DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None,
                                         criterion='gini', max_depth=None,
                                         max_features=None,
                                         max_leaf_nodes=None,
                                         min_impurity_decrease=0.0,
                                         min_impurity_split=None,
                                         min_samples_leaf=1,
                                         min_samples_split=2,
                                         min_weight_fraction_leaf=0.0,
                                         presort='deprecated',
                                         random_state=None,
                                         splitter='best'),
        iid='deprecated', n_jobs=None,
        param_grid={'max_depth': [4, 5, 6],
                    'min_samples_leaf': [10, 20, 40, 60],
                    'min_samples_split': [100, 150, 200, 250]},
        pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
        scoring=None, verbose=0)
```

Step4: fitting grid search for train data and train labels for optimal grid

Step5: best_params

{'max_depth': 4, 'min_samples_leaf': 10, 'min_samples_split': 100}

Step6: Generating new tree using graphviz (http://webgraphviz.com/)

Feature Importances:

	Imp
Agency_Code	0.608425
Sales	0.249026
Product Name	0.076765
Duration	0.035874
Commision	0.029910
Age	0.000000
Type	0.000000
Channel	0.000000
Destination	0.000000

Step7: Predicting on Training and Test dataset

Random Forest Classifier

Step 1: Extracting the target column into separate vectors for training set and test set

Step 2: Splitting data into training and test set for independent attributes

Step3: Search for optimal grid.

```
GridSearchCV(cv=3, error_score=nan,
        estimator=RandomForestClassifier(bootstrap=True, ccp_alpha=0.0,
                                    class_weight=None,
                                    criterion='gini', max_depth=None,
                                    max_features='auto',
                                    max_leaf_nodes=None,
                                    max_samples=None,
                                    min_impurity_decrease=0.0,
                                    min_impurity_split=None,
                                    min_samples_leaf=1,
                                    min_samples_split=2,
                                    min_weight_fraction_leaf=0.0,
                                    n_estimators=100, n_jobs=None,
                                    oob_score=False,
                                    random_state=None, verbose=0,
                                    warm_start=False),
        iid='deprecated', n_jobs=None,
        param_grid={'max_depth': [4, 5, 6], 'max_features': [2, 3, 4, 5],
                'min_samples_leaf': [8, 9, 10, 11, 12],
                'min_samples_split': [45, 50, 55],
                'n_estimators': [300, 350, 400]},
        pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
        scoring=None, verbose=0)
```

Step4: fitting grid search for train data and train labels for optimal grid

Step5: best_params

```
{'max_depth': 6,
 'max_features': 5,
 'min_samples_leaf': 10,
 'min_samples_split': 50,
 'n_estimators': 300}
```

Variable Importance via RF:

	Imp
Agency_Code	0.350674
Product Name	0.204103
Sales	0.175205
Commision	0.115021
Duration	0.068407
Age	0.049928
Type	0.028776
Destination	0.007463
Channel	0.000423

Step6: Predicting the Training and Testing data

```
ytrain_predict_rf = best_grid_rf.predict(X_train)

ytest_predict_rf = best_grid_rf.predict(X_test)
```

Building a Neural Network Classifier

Step1: Extracting the target column into separate vectors for training set and test set:

Step2: Splitting data into training and test set for independent attributes

Step3: Using **Standard scaler** method fit and transform the train and only transform for test data

Step4: Search for optimal grid

```
GridSearchCV(cv=3, error_score=nan,
        estimator=MLPClassifier(activation='relu', alpha=0.0001,
                                batch_size='auto', beta_1=0.9,
                                beta_2=0.999, early_stopping=False,
                                epsilon=1e-08, hidden_layer_sizes=(100,),
                                learning_rate='constant',
                                learning_rate_init=0.001, max_fun=15000,
                                max_iter=200, momentum=0.9,
                                n_iter_no_change=10,
                                nesterovs_momentum=True, power_t=0.5,
                                random_state=None, shuffle=True,
                                solver='adam', tol=0.0001,
                                validation_fraction=0.1, verbose=False,
                                warm_start=False),
        iid='deprecated', n_jobs=None,
        param_grid={'activation': ['logistic', 'relu'],
                    'hidden_layer_sizes': [50, 100, 150],
                    'max_iter': [10000], 'solver': ['sgd', 'adam'],
                    'tol': [0.1, 0.01]},
        pre_dispatch='2*n_jobs', refit=True, return_train_score=False,
        scoring=None, verbose=0)
```

Step5: fitting grid search for train data and train labels for optimal grid

Step6: Getting best params

{'activation': 'relu',
'hidden_layer_sizes': 150,
'max_iter': 10000,
'solver': 'adam',
'tol': 0.01}

Step7: Predicting the Training and Testing data

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.

Check the performance of Predictions on Train and Test sets using Accuracy, Confusion Matrix, Plot ROC curve and get ROC_AUC score for,

Decision Tree Model

Accuracy for train data = 0.7928571428571428
Accuracy for test data = 0.7811111111111111

Classification Report for train data

	precision	recall	f1-score	support
0	0.85	0.86	0.85	1471
1	0.66	0.64	0.65	629
accuracy			0.79	2100
macro avg	0.75	0.75	0.75	2100
weighted avg	0.79	0.79	0.79	2100

train data f1-score = 0.65
train data recall = 0.64
train data precision = 0.66

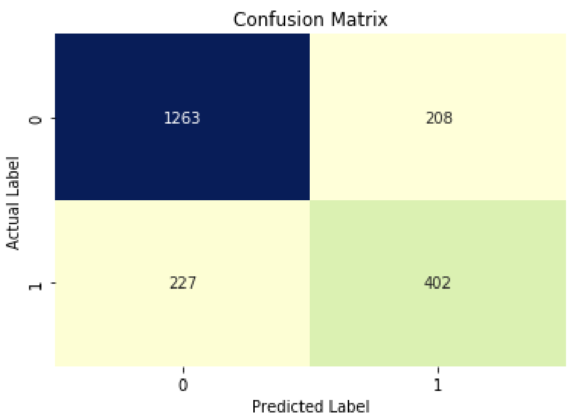
Classification Report for test data

	precision	recall	f1-score	support
0	0.81	0.89	0.84	605
1	0.71	0.57	0.63	295
accuracy			0.78	900

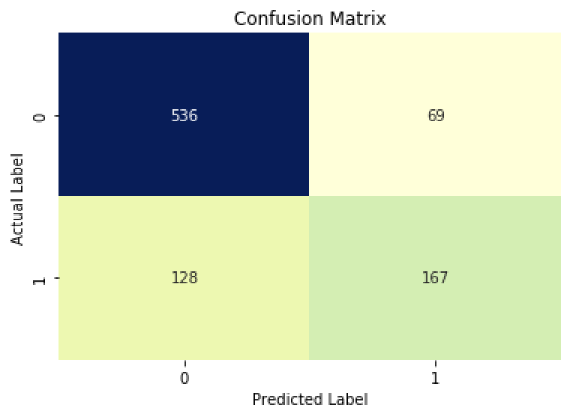
macro avg	0.76	0.73	0.74	900
weighted avg	0.77	0.78	0.77	900

test data f1-score = 0.63
test data recall = 0.57
test data precision = 0.71

Confusion Matrix for train data

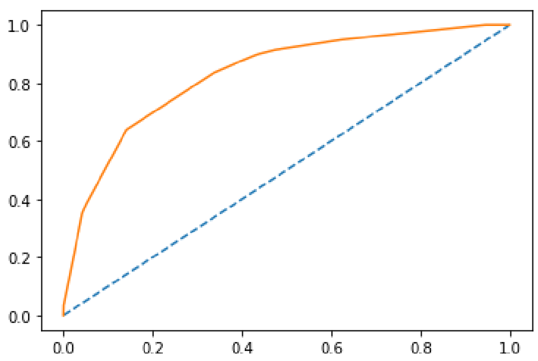


Confusion Matrix for test data



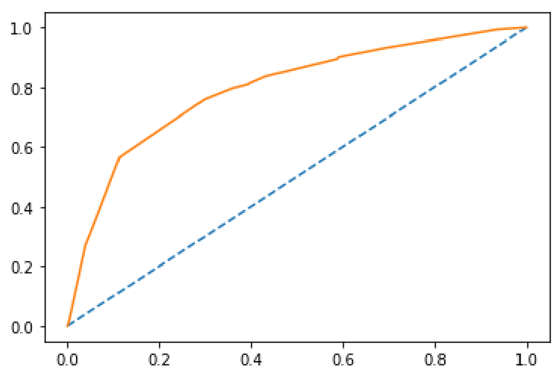
AUC and ROC for train data

AUC: 0.830



AUC and ROC for test data

AUC: 0.794



Check the performance of Predictions on Train and Test sets using Accuracy, Confusion Matrix, Plot ROC curve and get ROC_AUC score for

Random forest Model

Accuracy for train data = 0.81
Accuracy for test data = 0.77

Classification Report for train data

	precision	recall	f1-score	support
0	0.84	0.90	0.87	1471
1	0.72	0.60	0.65	629
accuracy			0.81	2100
macro avg	0.78	0.75	0.76	2100
weighted avg	0.80	0.81	0.81	2100

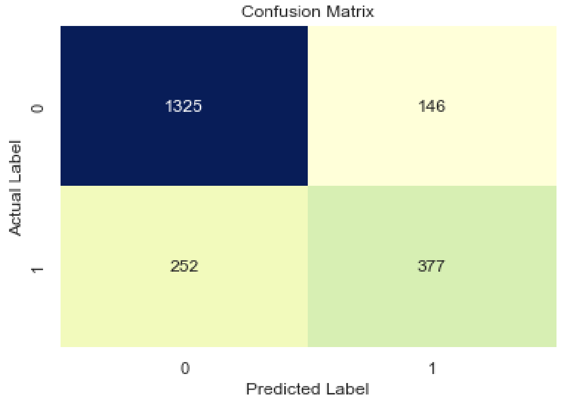
train data f1-score = 0.65
train data recall = 0.6
train data precision = 0.72

Classification Report for test data

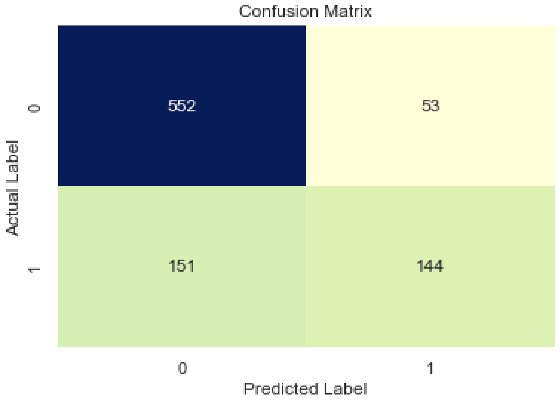
	precision	recall	f1-score	support
0	0.79	0.91	0.84	605
1	0.73	0.49	0.59	295
accuracy			0.77	900
macro avg	0.76	0.70	0.71	900
weighted avg	0.77	0.77	0.76	900

test data f1-score = 0.59
test data recall = 0.49
test data precision = 0.73

Confusion Matrix for train data

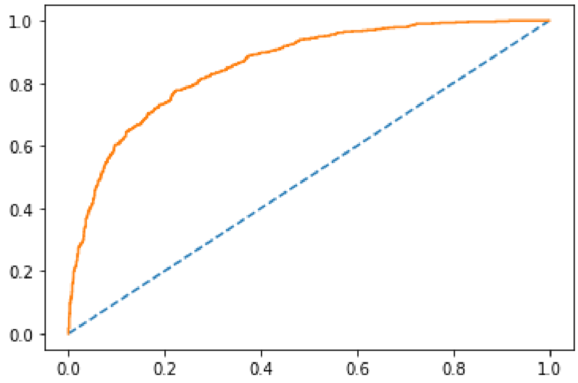


Confusion Matrix for test data



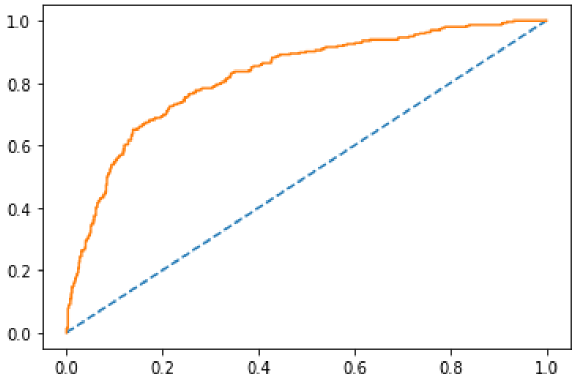
AUC and ROC for train data

AUC: 0.858



AUC and ROC for test data

AUC: 0.822



Check the performance of Predictions on Train and Test sets using Accuracy, Confusion Matrix, Plot ROC curve and get ROC_AUC score for,

Artificial Neural Network

Accuracy for train data = 0.79
Accuracy for test data = 0.60

Classification Report for train data

	precision	recall	f1-score	support
0	0.81	0.90	0.85	1471
1	0.67	0.49	0.57	629
accuracy			0.78	2100
macro avg	0.74	0.70	0.71	2100
weighted avg	0.77	0.78	0.77	2100

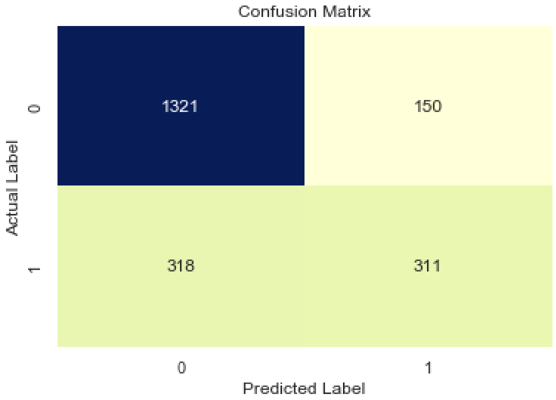
train data f1-score = 0.57
train data recall = 0.49
train data precision = 0.67

Classification Report for test data

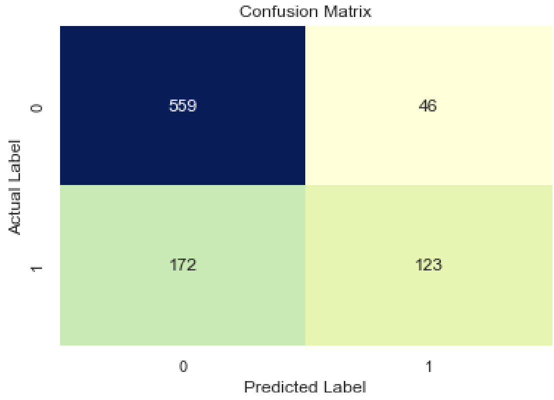
	precision	recall	f1-score	support
0	0.76	0.92	0.84	605
1	0.73	0.42	0.53	295
accuracy			0.76	900
macro avg	0.75	0.67	0.68	900
weighted avg	0.75	0.76	0.74	900

test data f1-score = 0.53
test data recall = 0.42
test data precision = 0.73

Confusion Matrix for train data

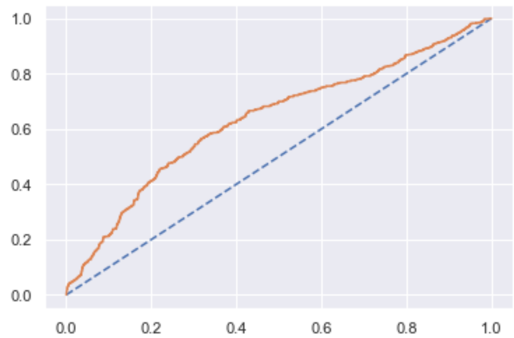


Confusion Matrix for test data



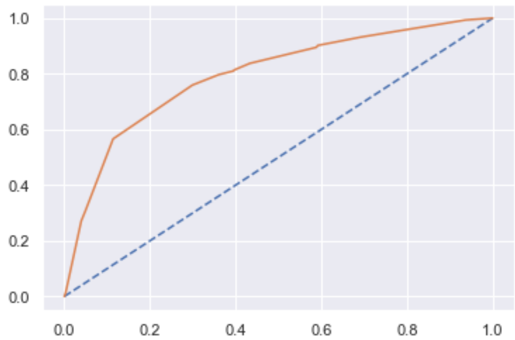
AUC and ROC for train data

AUC: 0.635



AUC and ROC for test data

AUC: 0.794

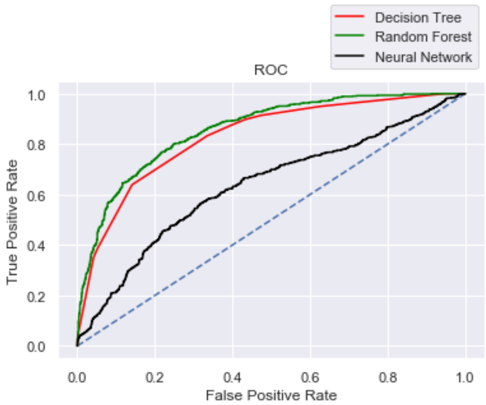


2.4 Final Model: Compare all the models and write an inference which model is best/optimized.

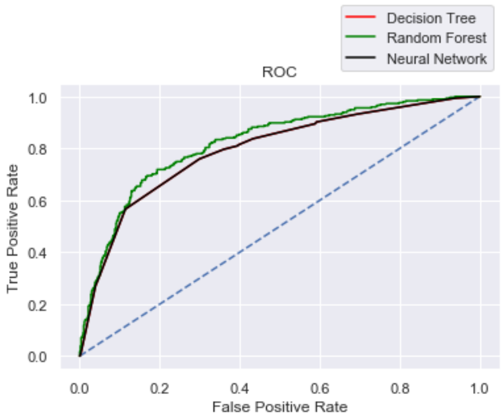
<u>Compare all the models</u>

	Decision Tree Train	Decision Tree Test	Random Forest Train	Random Forest Test	Neural Network Train	Neural Network Test
Accuracy	0.79	0.78	0.81	0.77	0.79	0.61
AUC	0.83	0.79	0.86	0.82	0.63	0.79
Recall	0.64	0.57	0.60	0.49	0.49	0.42
Precision	0.66	0.71	0.72	0.73	0.67	0.73
F1 Score	0.65	0.63	0.65	0.59	0.57	0.53

ROC Curve for the 3 models on the Training data



ROC Curve for the 3 models on the Test data



Selecting the model:
Random Forerst model is the most suitable model for this data, because it has better accuracy, precsion, f1 score better than other two decision tree & Neural Network.

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

It needs more data to build a good model. Data deficiency seen in this model

1) Its better to approach to book airline tickets or plans, cross sell the insurance based on the claim data pattern due to the model we are getting 80%accuracy
2) 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
3) Other interesting fact, is almost all the offline business has a claimed associated, need to find why?
4) As per the data 90% of insurance is done by online channel.
5) Streamlining online experiences benefitted customers, leading to an increase in conversions, which subsequently raised profits.

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

Increase the claims recovery.
Fight against fraud
Balance distribution of insurance business data
Reduce claims cycle time.
Increase customer satisfaction.
Combat fraud.