Data Mining PROJECT REPORT

DSBA

By- Aarti Londhe

greatlearning
Learning for Life

Contents

Table of Contents

Problem 1: Clustering

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

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

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

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

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.

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

Dataset for Problem 1: bank_marketing_part1_Data.csv

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.

2.1 Read the data, do the necessary initial steps, and exploratory data analysis (Univariate, Bi-variate, and multivariate analysis).
2.2 Data Split: Split the data into test and train, build classification model CART, Random Forest, Artificial Neural Network
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.
2.4 Final Model: Compare all the models and write an inference which model is best/optimized.
2.5 Inference: Based on the whole Analysis, what are the business insights and recommendations

Dataset for Problem 2: insurance_part2_data-1.csv

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.

1.1 Read the data, do the necessary initial steps, and exploratory data analysis (Univariate, Bi-variate, and multivariate analysis).
1.2 Do you think scaling is necessary for clustering in this case? Justify
1.3 Apply hierarchical clustering to scaled data. Identify the number of optimum clusters using Dendrogram and briefly describe them
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.
1.5 Describe cluster profiles for the clusters defined. Recommend different promotional strategies for different clusters.

Dataset for Problem 1: bank_marketing_part1_Data.csv

Data Dictionary for Market Segmentation:
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)

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

Here, we are given a collected sample of data that summarizes the activities of the credit card users during the past few months.
Dataset file- bank_marketing_part1_Data.csv

Below is the sample of loaded dataset:

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

210 rows × 7 columns

There is total 210 rows and 7 columns in the given dataset.

Let's check the data types of loaded features:

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

- There is total 7 features and all of data type Float.
- Also, from above result we can see that there are no missing values present in the dataset.

Descriptive Statistics:

Descriptive statistics help to describe and understand the features of a specific data set by giving short summaries about the sample and measures of the data. The most recognized types of descriptive statistics are measures of center: the mean, median, and mode, which are used at almost all levels of math and statistics.

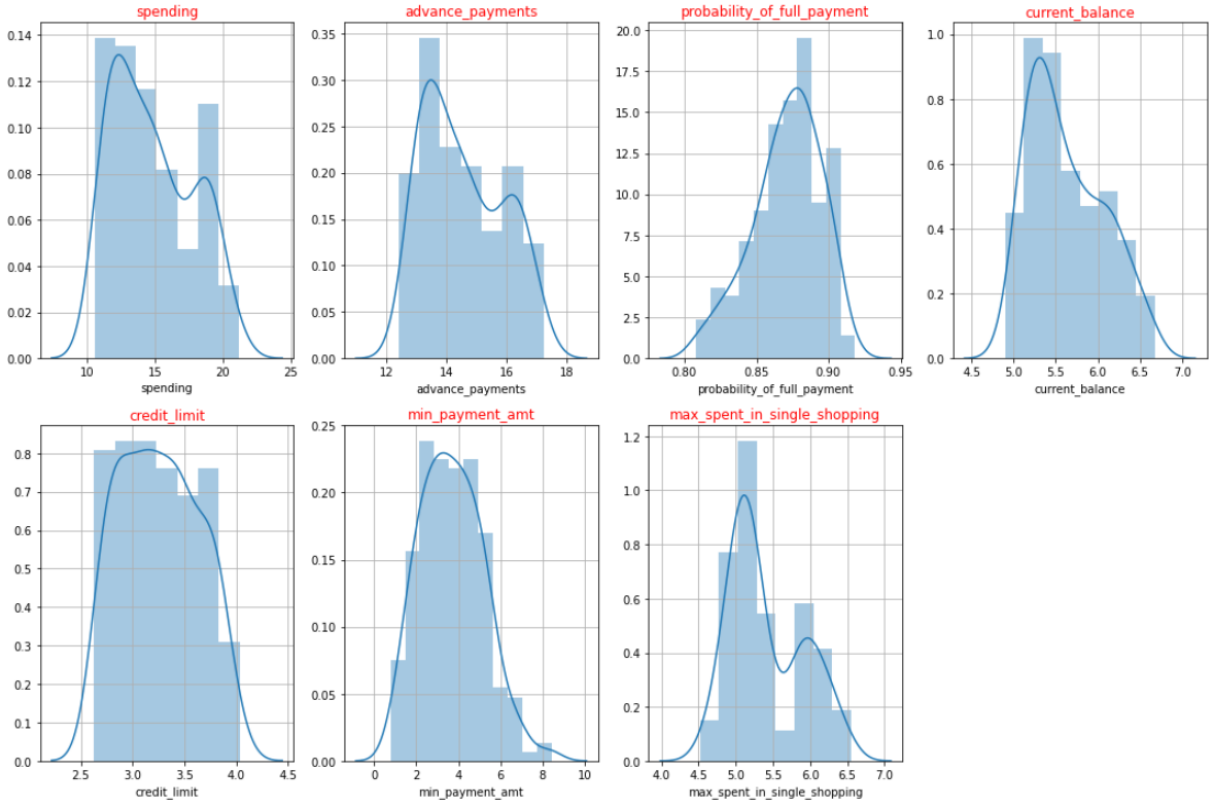
	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
count	210.000000	210.000000	210.000000	210.000000	210.000000	210.000000	210.000000
mean	14.847524	14.559286	0.870999	5.628533	3.258605	3.700201	5.408071
std	2.909699	1.305959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

From the descriptive statistics, we can see that:

- The average spending for customers is around 15000.
- Maximum Amount paid by the customer in advance by cash is around 1700.
- current balance varies from 5000 to 7000.
- 50% of customers are given a credit limit of 32000.
- Maximum amount spent in one purchase is around 7000.
- Minimum amount of 100Rs is paid by the customer while making payments for purchases made monthly.

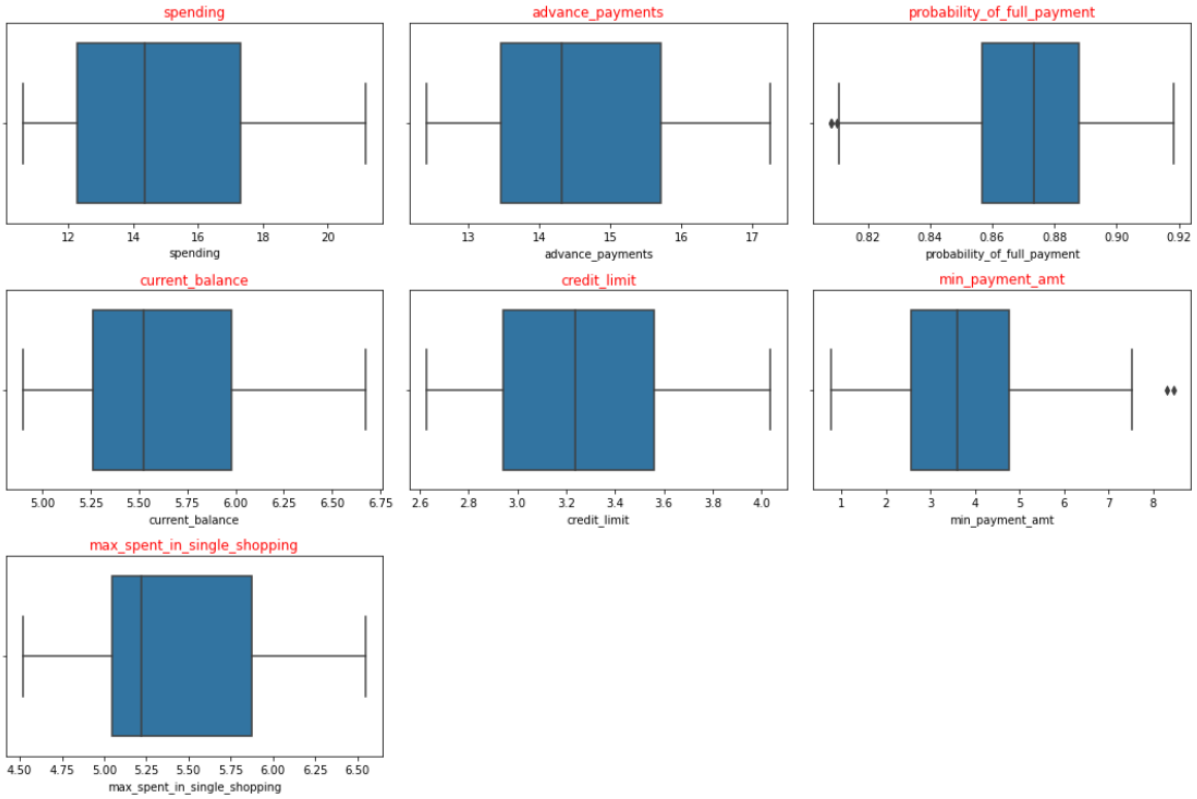
Exploratory Data Analysis:

Distribution of variables- Dist Plot



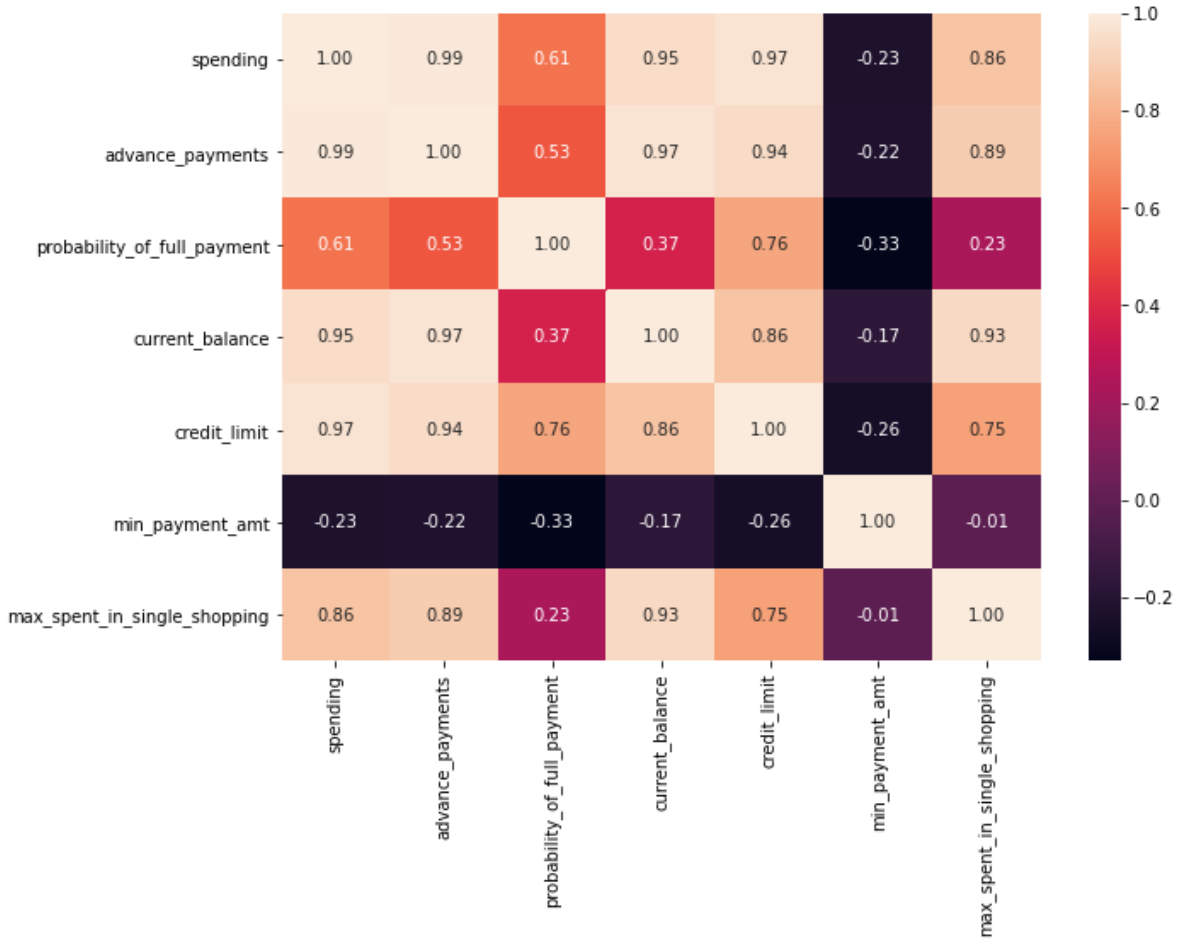
- Credit_limit seems to be normally distributed.
- Max_spent_in_single_shopping, Spending of customer & current balance are right skewed.

Outliers in the dataset: Boxplot



- Outliers are observed for min_payment_amt & probability of full payment.
- Here as we are going to segment the data based on their past usage samples, we won't treat the outliers.

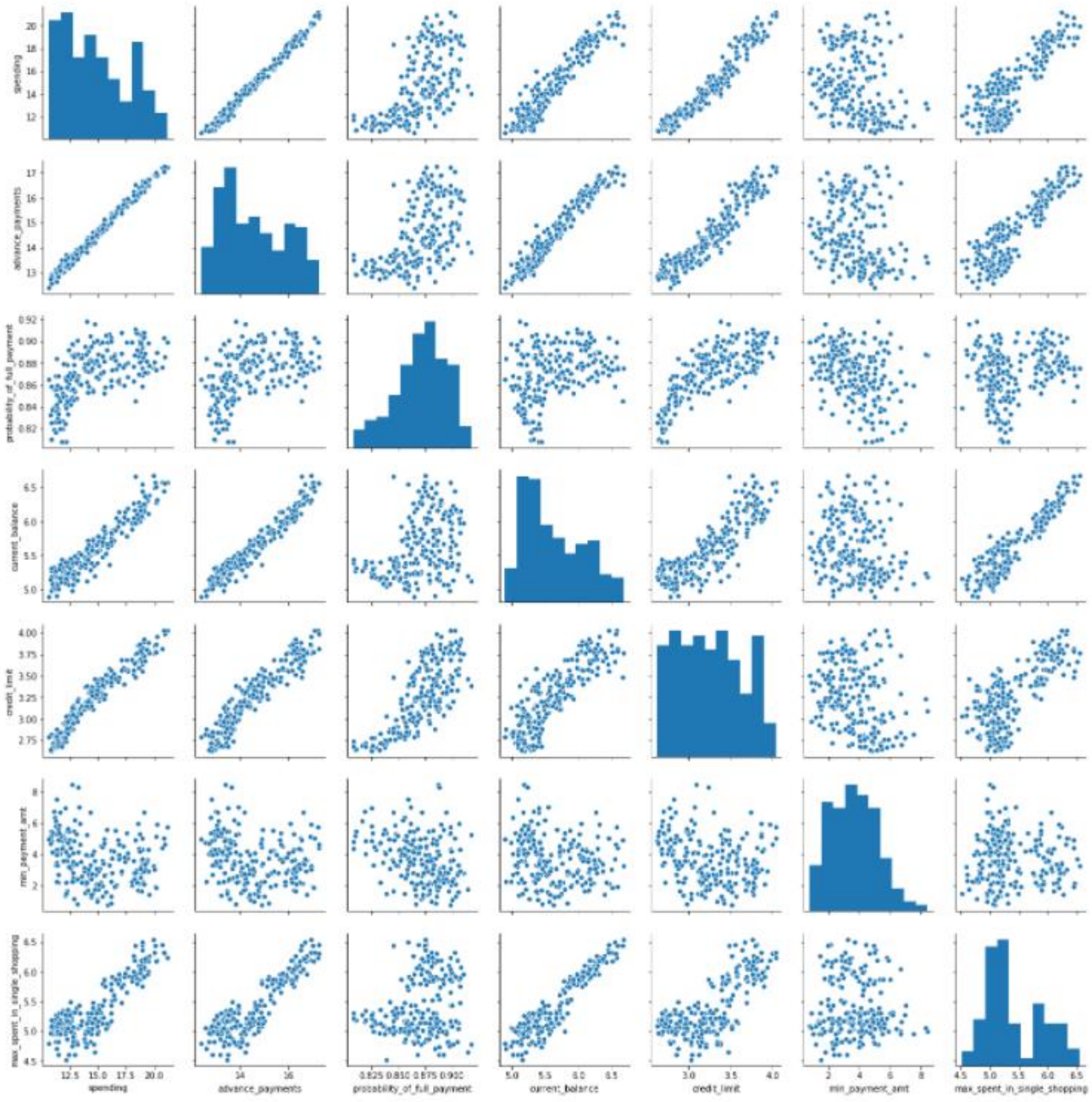
Heatmap:



- From the correlation plot, we can see that various attributes of the dataset are highly correlated to each other except min_payment_amt.
- Correlation values near to 1 or -1 are highly positively correlated and highly negatively correlated respectively.
- Correlation values near to 0 are not correlated to each other.

Pairplot:

Pairplot shows the relationship between the variables in the form of scatterplot and the distribution of the variable in the form of histogram.



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

Below is descriptive statistics summary of the dataset.

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

- From above descriptive statistics, we can see that Mean values are distinctly different from each other.
- So, we need to perform scaling so that data range is same for all and varies within same scale.
- Here we will use standardscaler method which uses zscore technique and coverts data to a mean of 0 and std deviation of 1.
  Basically, it centralizes the data to origin or zero value.

Scaled dataset array:

```
: array([[ 1.75435461,  1.81196782,  0.17822987, ...,  1.33857863,
          -0.29880602,  2.3289982 ],
        [ 0.39358228,  0.25383997,  1.501773  , ...,  0.85823561,
         -0.24280501, -0.53858174],
        [ 1.41330028,  1.42819249,  0.50487353, ...,  1.317348  ,
         -0.22147129,  1.50910692],
        ...,
        [-0.2816364 , -0.30647202,  0.36488339, ..., -0.15287318,
         -1.3221578 , -0.83023461],
        [ 0.43836719,  0.33827054,  1.23027698, ...,  0.60081421,
         -0.95348449,  0.07123789],
        [ 0.24889256,  0.45340314, -0.77624835, ..., -0.07325831,
         -0.70681338,  0.96047321]])
```

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

- In Hierarchical clustering, Records are sequentially grouped to create clusters, based on distances between records and distances between clusters.

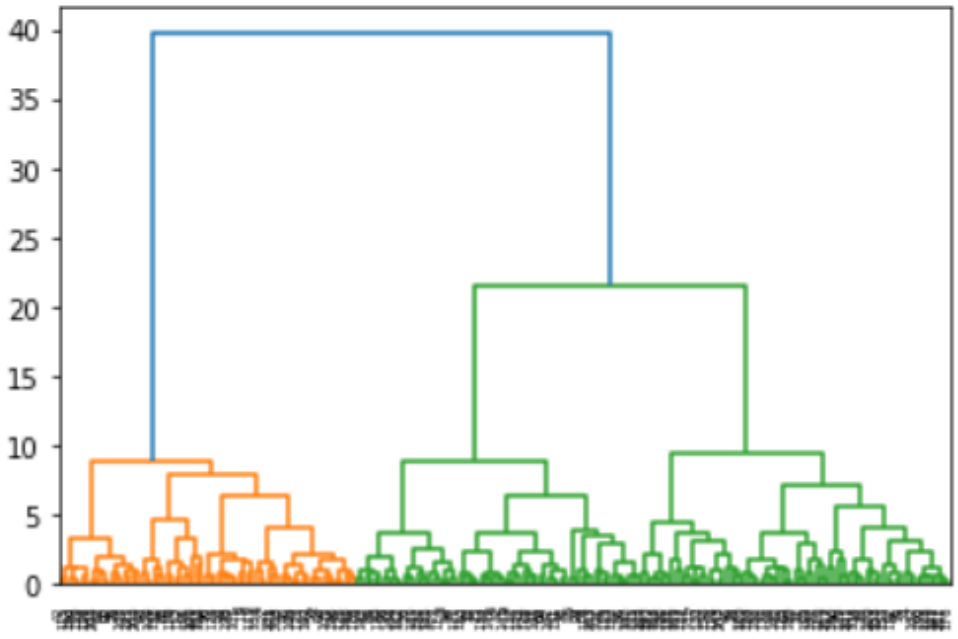
- Distance between records is measured using 'Eucledian distance' method
  And Distance between records is measured using 'Linkage' method

- Linkage method- here, we have used the "ward linkage" method to calculate distance between rows and then merge data based on calculated distances.

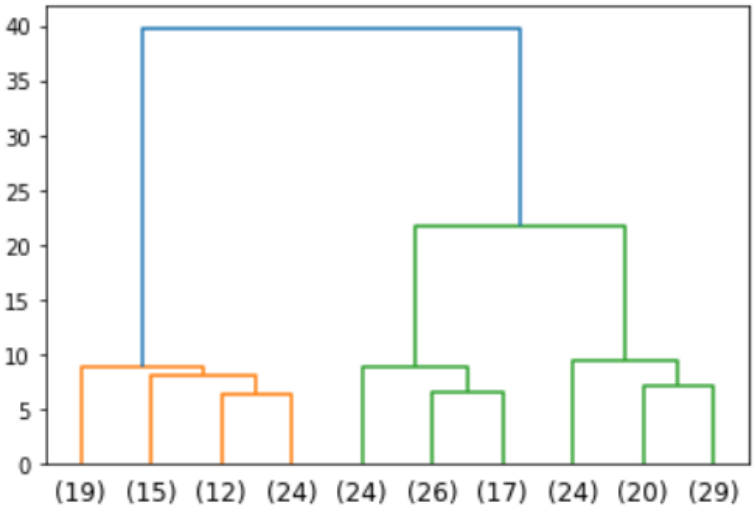
- Hierarchical clustering produces a useful graphical display of the clustering process and results, called a dendrogram.

Dendrogram:

-A dendrogram is a treelike diagram that summarizes the process of clustering.
-On the x-axis are the records. Similar records are joined by lines whose vertical length reflects the distance between the records. The greater the difference in height, the more dissimilarity.



- As there are so many rows, dendrogram looks complicated and is not clear to understand.
- Using truncate method, we can display the dendrogram in less complicated way.
- Below dendrogram shows last 10merge.



Maximum Number of clusters:

- To decide maximum number of clusters, we will use fluster function with criterion maxclust or we can use criterion distance.

- Dendrogram shows 2 main colors Orange and green clusters

- It probably makes more sense to use #3 as that carries more business sense. Generally speaking, 2 clusters really do not make much business impact as it is kind of implicit.
  For example, for the dataset, it is imperative that there will be some high spenders and some low spenders.

- So here we will use 3 in maxclust criterion, which will give us 3 as optimal number of clusters.
- Or we can use distance as 15 which will divide dendrogram as 15 y-level which will again give us 3 clusters.

- And thus, customers can be segmented into High, medium, low spenders i.e. 3 clusters.

Customer segmentation:

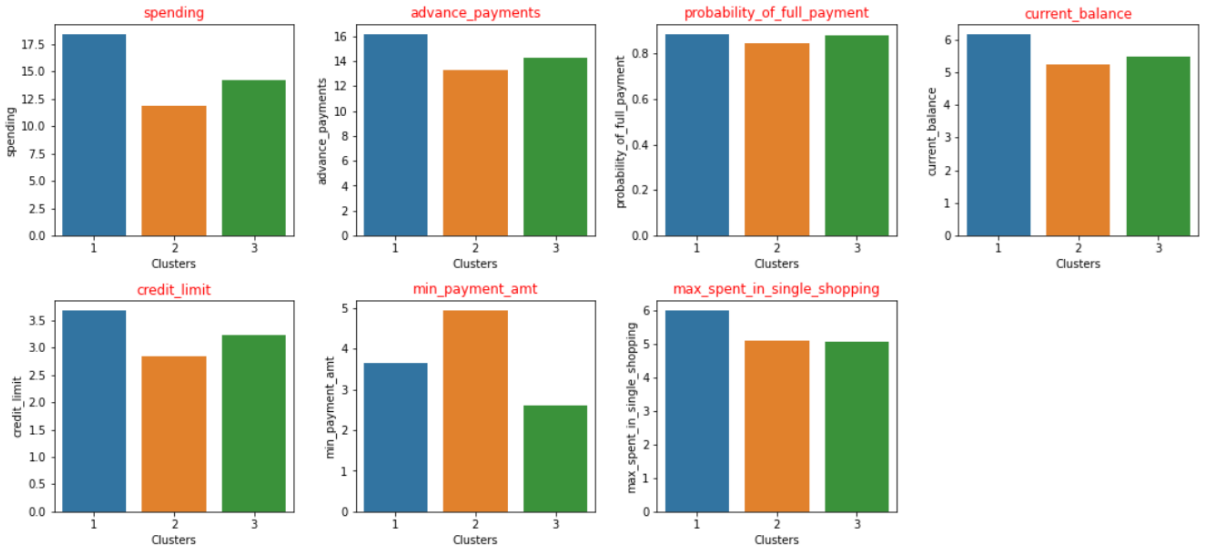
We can add cluster column in dataset to define record belon to which clusters.

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

210 rows × 8 columns

Visualization of customer segments:

Below is the barplot for each variable against their clusters.



Cluster profiles:

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

Observations:

- Number of customers are maximum for Cluster 3 and minimum for cluster 2
- Cluster 1 customers have high spending and high current balance compared to cluster 2 & 3.
- Also, Maximum amount spent in one purchase is observed to be high for cluster 1.
- Minimum payment amount is max for cluster 2 compared to cluster 1 & 3.
- Probability of payment done in full by the customer to the bank is more of less same for cluster 1, 2 & 3.

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.

- KMeans clustering is a type of Non-Hierarchical clustering in which number of clusters are pre decided.

- And then records are partitioned into given number of clusters based on its distance from centriod.

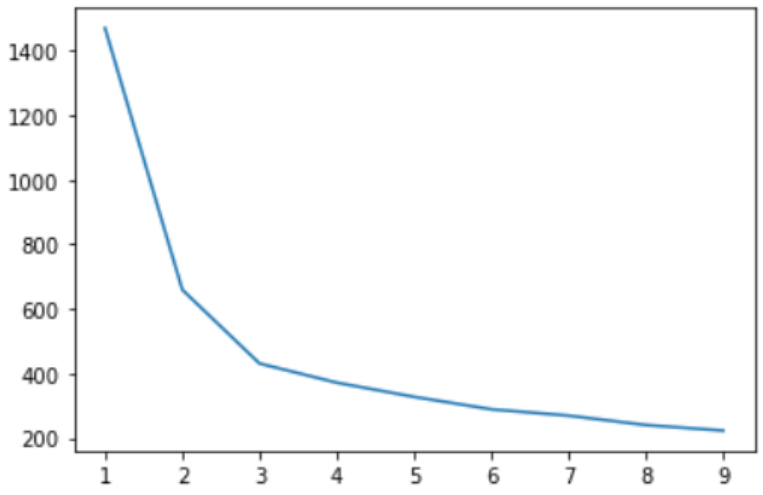
- First, we will apply the KMeans function for different number of clusters 1,2,3,4,5 and will check the respective inertia value i.e., within sum of square for respective number of clusters.

- Same can be achieved using WSS calculation and elbow method. This WSS value can be plotted against each cluster to understand the drop in inertia value and optimum number of clusters.

Below are the WSS values starting from cluster 1 to cluster 9.

```
[1469.9999999999998,
 659.171754487041,
 430.6589731513006,
 371.2834476674333,
 327.3281094192775,
 288.7694577022641,
 269.162141178897,
 240.0076721107359,
 222.95377418405351]
```

Elbow Plot:



- As shown above in WSS plot, there is considerable amount of inertia drop till number of cluster equal to 3.

- So, we can say 3 as optimal number of clusters. After that there is no much difference in inertia value and hence additional clustering is of no use.

- Cluster validation : Rows assigned to respective Clusters can be validated using “Silhouette Sco re” , which calculates average of silhoutte width for each value.

- For number of clusters = 3 , Silhouette Score is 0.40 which is a positive value.

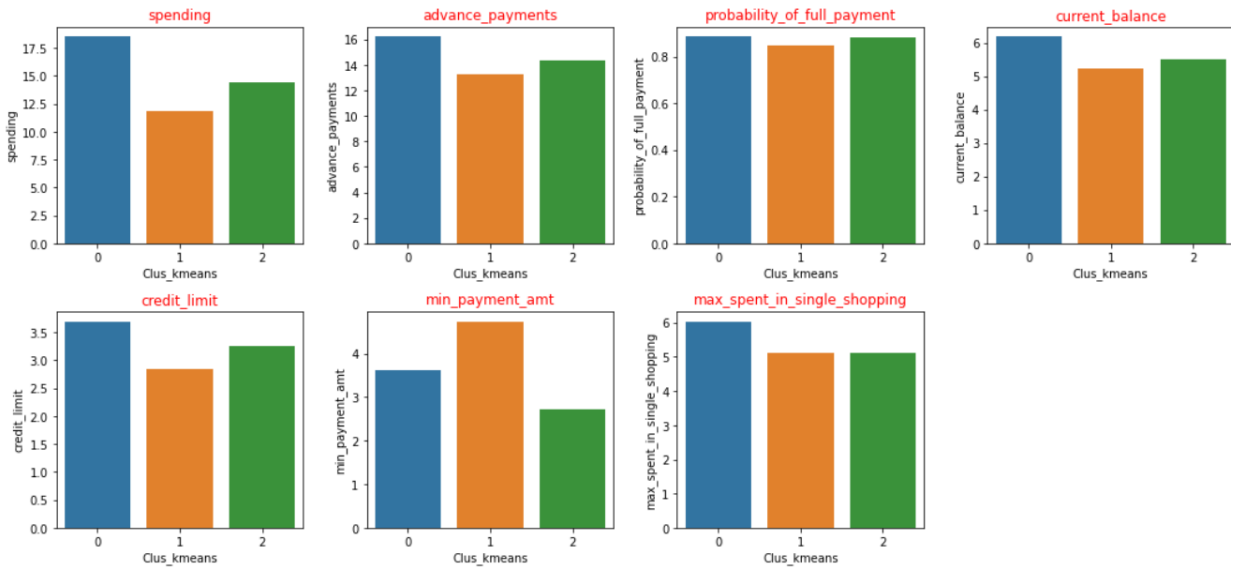
- And thus, customers can be segmented into High, medium, low spenders i.e. 3 clusters.

Customer Segments:

spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	Clusters	Clus_kmeans
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	0
18.95	16.42	0.8829	6.248	3.755	3.368	6.148	1	1
10.83	12.96	0.8099	5.278	2.641	5.182	5.185	2	2
17.99	15.86	0.8992	5.890	3.694	2.068	5.837	1	1
...	...	...	...	...	...	...	...	...
13.89	14.02	0.8880	5.439	3.199	3.986	4.738	3	0
16.77	15.62	0.8638	5.927	3.438	4.920	5.795	1	1
14.03	14.16	0.8796	5.438	3.201	1.717	5.001	3	0
16.12	15.00	0.9000	5.709	3.485	2.270	5.443	1	0
15.57	15.15	0.8527	5.920	3.231	2.640	5.879	3	0

Visualization:

Below is the barplot for each variable against their clusters.



Cluster Profiles:

	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	Clusters	freq
neans									
0	18.495373	16.203433	0.884210	6.175687	3.697537	3.632373	6.041701	1.029851	67
1	11.856944	13.247778	0.848253	5.231750	2.849542	4.742389	5.101722	2.083333	72
2	14.437887	14.337746	0.881597	5.514577	3.259225	2.707341	5.120803	2.873239	71

Observations:

- Number of customers are maximum for Cluster 3 and minimum for cluster 1
- Cluster 0 customers have high spending and high current balance compared to cluster 1 & 2.
- Also, Maximum amount spent in one purchase is observed to be high for cluster 0.
- Minimum payment amount is max for cluster 1 compared to cluster 0 & 2.
- Probability of payment done in full by the customer to the bank is more of less same for cluster 1, 2 & 3.

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

Based on KMeans clustering, below are the recommendations.

Based on the collected samples that summarizes the activities of users during the past few months, customers can be segmented into 3 profiles.

- Cluster 0: high spending,high advance payments,high credit limit,high current balance and medium min_payment_amt
- Cluster 1: low spending, low advance payments, low credit limit,low current balance and high min_payment_amt
- Cluster 2: medium spending, medium advance payments, medium credit limit,medium current balance and low min_payment_amt
- Cluster 0, people are economically stable people can afford to spend and pay the credit card due amount. these are less in number compared to cluster 1 & 2. Also, they have good credit history.
- cluster 1,people are high in number and it seems to they are using basic card which is given at first to customer.
  Based on their usage area we can give them promotional offers to increase their transactions from card.
- cluster 2,people are average salaried or fixed income ones. We can encourage them to upgrade to credit card with more credit limit so that they can pay for big budget items at a time. But it will have high annual fee.

Problem 2:

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.

2.1 Read the data, do the necessary initial steps, and exploratory data analysis (Univariate, Bi-variate, and multivariate analysis).
2.2 Data Split: Split the data into test and train, build classification model CART, Random Forest, Artificial Neural Network
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.
2.4 Final Model: Compare all the models and write an inference which model is best/optimized.
2.5 Inference: Based on the whole Analysis, what are the business insights and recommendations

Dataset for Problem 2: insurance_part2_data-1.csv

Attribute Information:

1. Target: Claim Status (Claimed)
2. Code of tour firm (Agency_Code)
3. Type of tour insurance firms (Type)
4. Distribution channel of tour insurance agencies (Channel)
5. Name of the tour insurance products (Product)
6. Duration of the tour (Duration)
7. Destination of the tour (Destination)
8. Amount of sales of tour insurance policies (Sales)
9. The commission received for tour insurance firm (Commission)
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).

Here, we are given with dataset of an Insurance firm providing tour insurance.

Below is sample 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

Let's Check the data types of dataset:

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

From above details,

- we can see that there is total 3000 rows and 10 columns.
- There are no null values in the dataset.
- Out of 10 columns, 2 are of type float, 2 are of type int & 6 are of type Object.
- There is total 9 independent variables and 1 dependent variable 'Claimed'.

Descriptive Statistics:

Descriptive statistics help to describe and understand the features of a specific data set by giving short summaries about the sample and measures of the data. The most recognized types of descriptive statistics are measures of center: the mean, median, and mode, which are used at almost all levels of math and statistics.

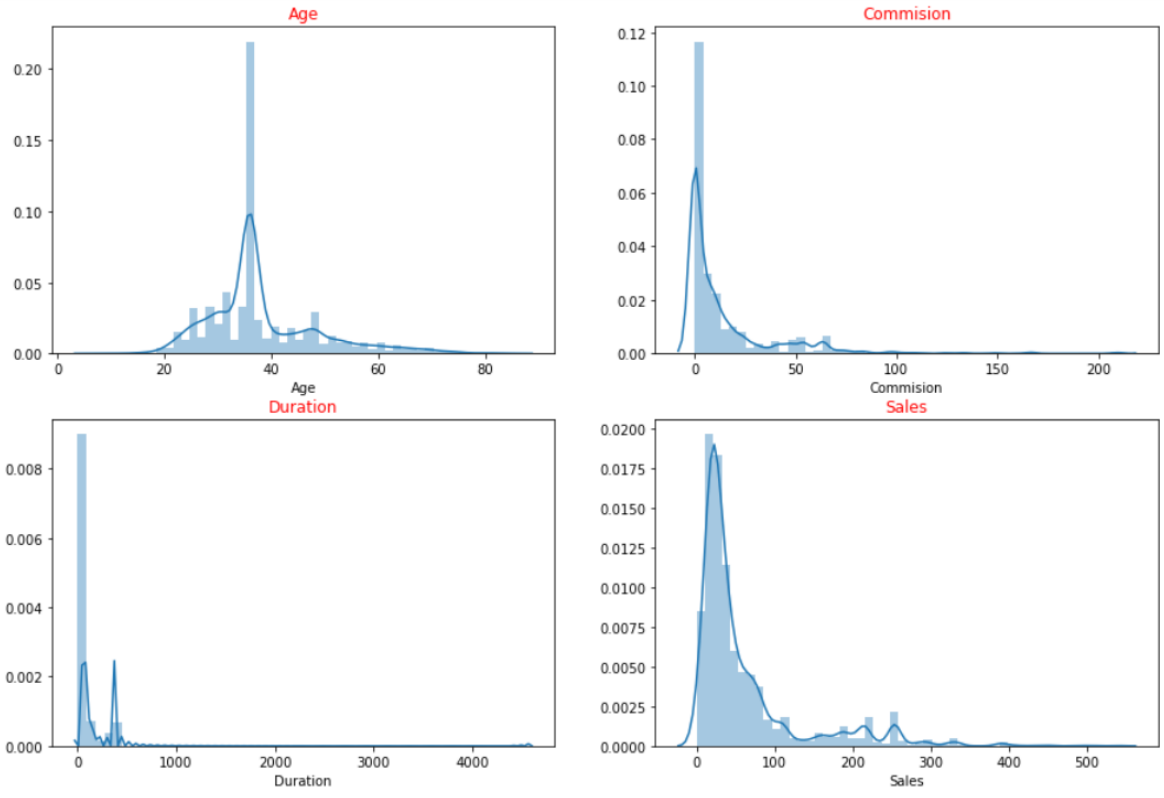
	count	unique	top	freq	mean	std	min	25%	50%	75%	max
Age	3000	NaN	NaN	NaN	38.091	10.4635	8	32	36	42	84
Agency_Code	3000	4	EPX	1365	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Type	3000	2	Travel Agency	1837	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Claimed	3000	2	No	2076	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Commision	3000	NaN	NaN	NaN	14.5292	25.4815	0	0	4.63	17.235	210.21
Channel	3000	2	Online	2954	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Duration	3000	NaN	NaN	NaN	70.0013	134.053	-1	11	26.5	63	4580
Sales	3000	NaN	NaN	NaN	60.2499	70.734	0	20	33	69	539
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

From above descriptive statistics, we can say that-

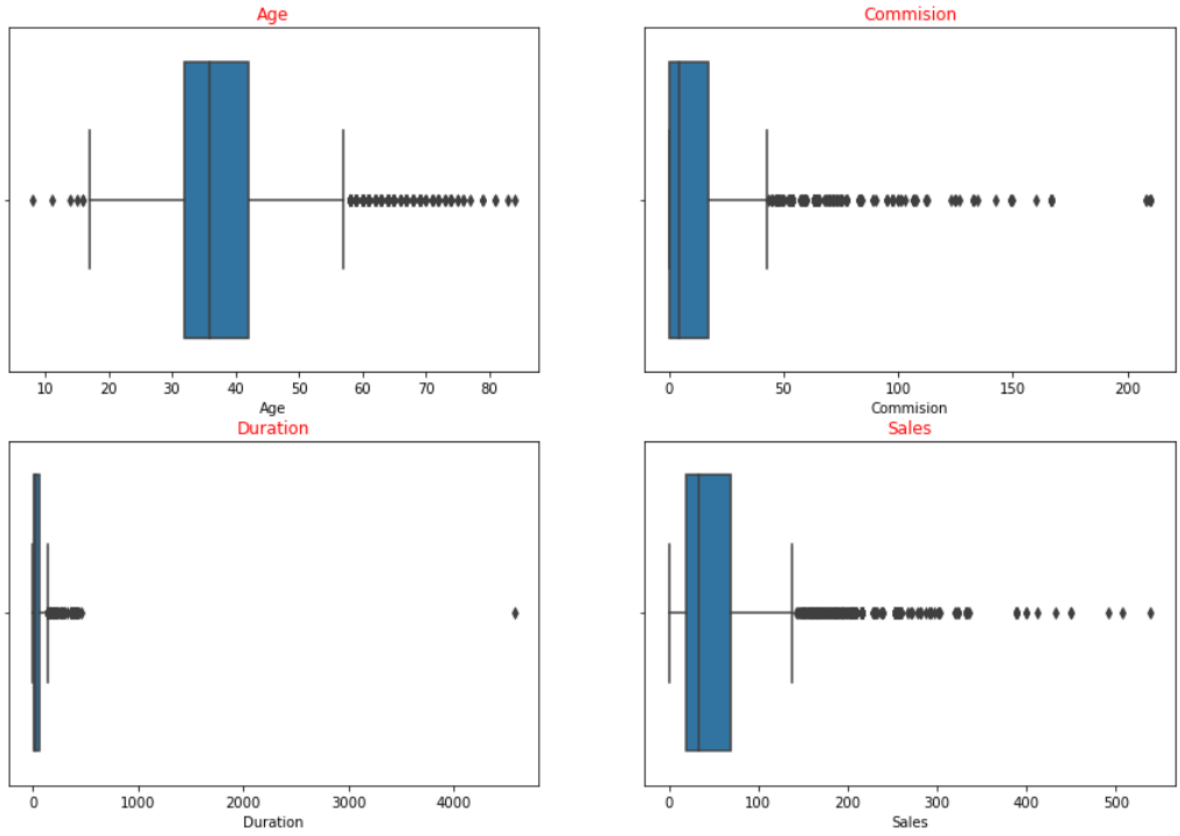
- age of customer varies from 8 to 84 years.
- Average commission received for tour insurance firm 14.53
- 50% of data has duration of the tour as 27
- Average Amount of sales of tour insurance policies is 60.25
- There is a negative value -1 as duration.
- Commission and Sales mean values differ significantly from each other.

Exploratory Data Analysis:

Distribution of variables- Dist plot

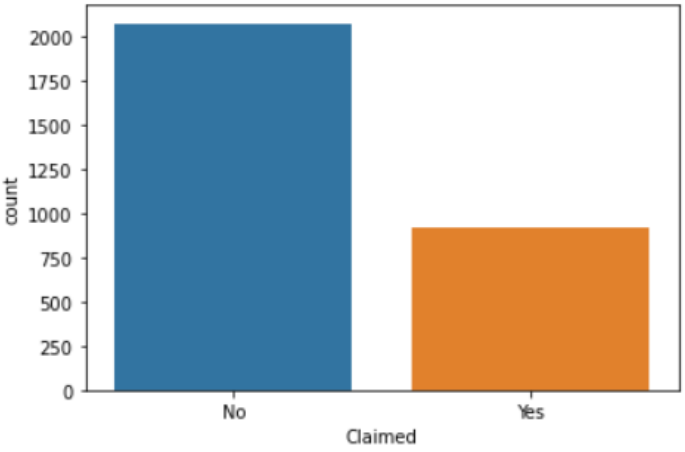


Boxplot:



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

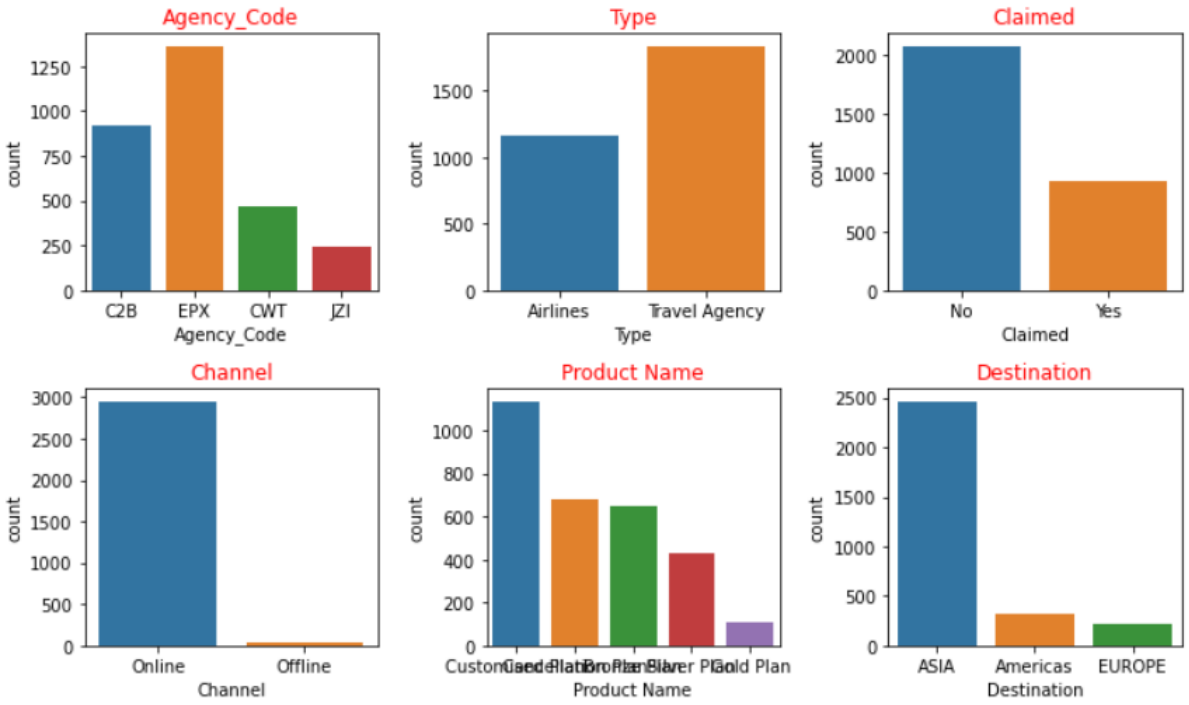
Here Target variable is Claimed.
Let's check the data distribution with respect to target variable Claimed.




```
No     2076
Yes     924
Name: Claimed, dtype: int64
```

Countplot:

Below is the countplot for variables:



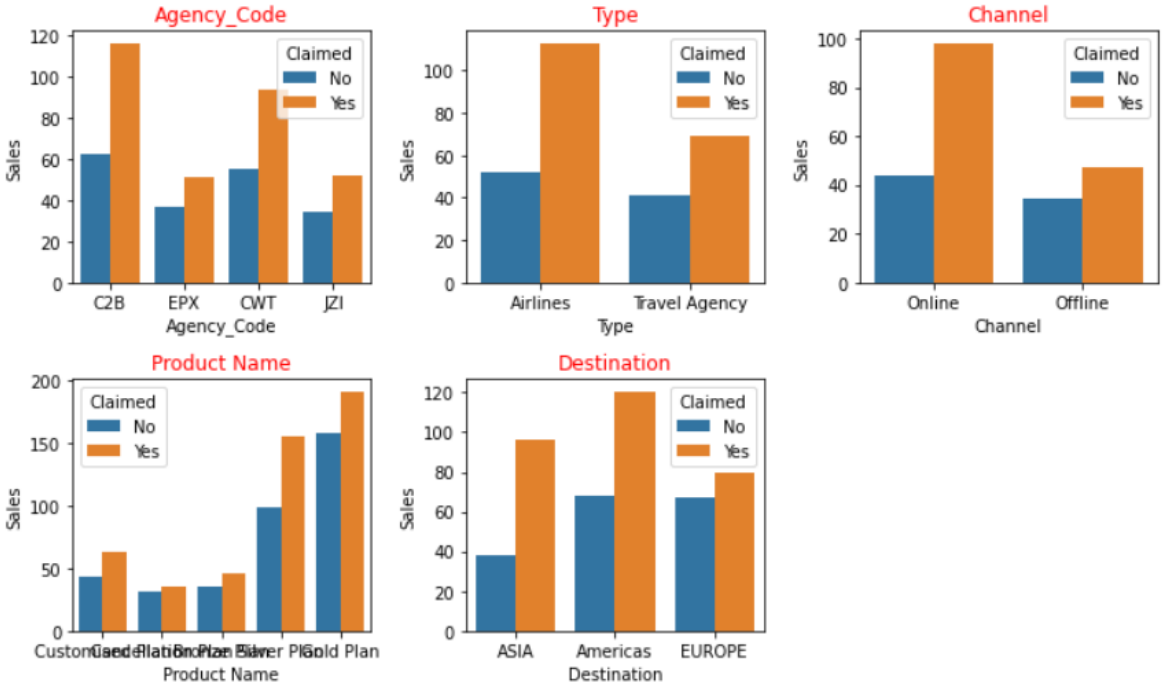
-Number of claims are higher for agency code EPX and of type Travel AGency.

-For Asia Destination company has received large number of insurance data.

-Maximum number of claims are through channel 'Online' and for product 'Customized plan'

Barplot:

Below is barplot of each variable against their Sales values.

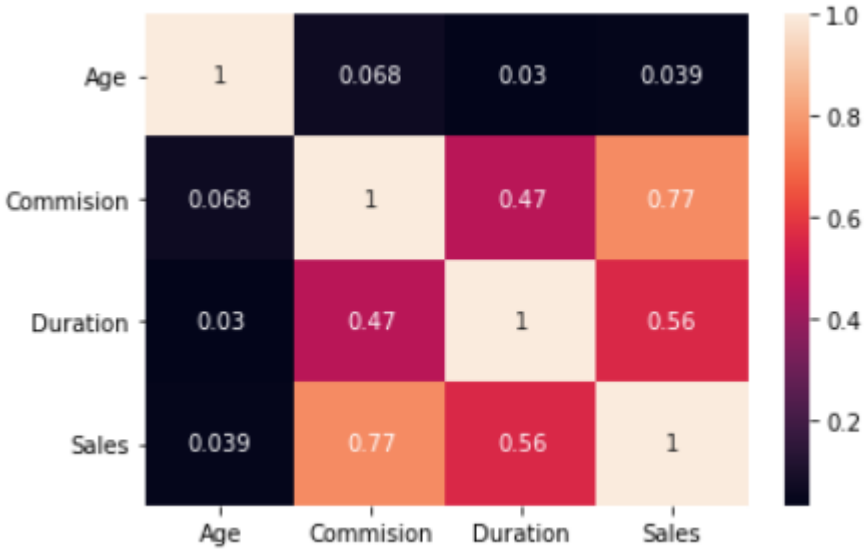


-For Airlines Sales value is greater for claimed status.

-Gold plan has higher sales compared to other plans.

-Americas Destinations have higher sales than Asia and Europe.

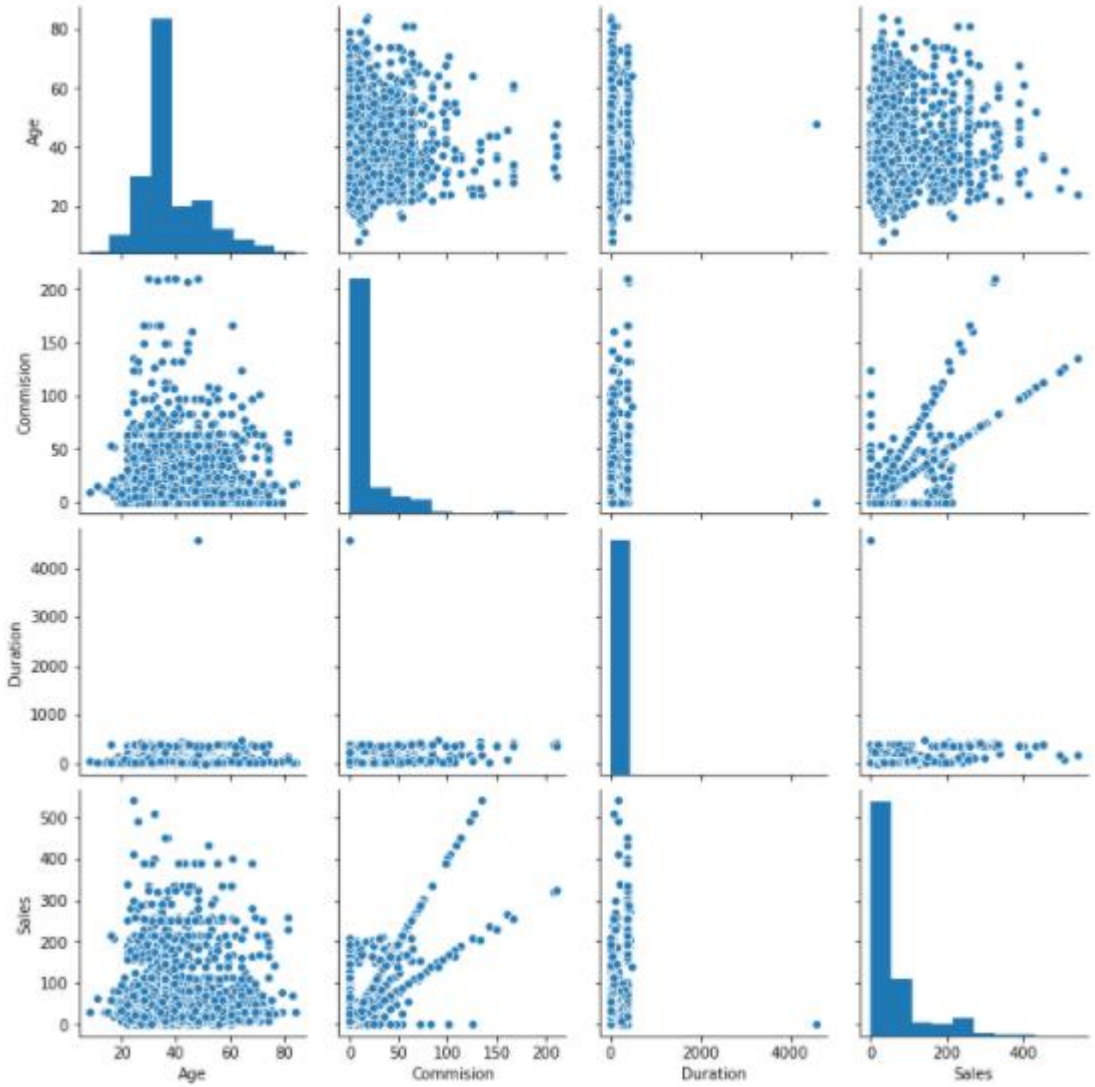
Heatmap:



- Commission and sales are highly correlated to each other.

Pairplot:

Pairplot shows the relationship between the variables in the form of scatterplot and the distribution of the variable in the form of histogram.



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

As Classifier model can be built on numeric data only, we need to first convert data objects to type numeric.

```
RangeIndex: 3000 entries, 0 to 2999
Data columns (total 10 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Age            3000 non-null   int64
 1   Agency_Code    3000 non-null   object
 2   Type           3000 non-null   object
 3   Claimed        3000 non-null   object
 4   Commision      3000 non-null   float64
 5   Channel        3000 non-null   object
 6   Duration       3000 non-null   int64
 7   Sales          3000 non-null   float64
 8   Product Name   3000 non-null   object
 9   Destination    3000 non-null   object
dtypes: float64(2), int64(2), object(6)
```

```
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]
```

```
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
```

-We will now split the data into independent variables (x) and dependent variables (y).

```
x = df.drop('Claimed',axis=1)
y = df.pop('Claimed')
```

```
x.head()
```

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

```
y.head()
```

```
0    0
1    0
2    0
3    0
4    0
Name: Claimed, dtype: int8
```

And using these independent and dependent variables, we will now create 70% train data and 30% Test Data with random state 1.

Train- Test Data split

```
x_train, x_test, train_labels, test_labels = train_test_split(x, y, test_size=.30, random_state=1)
```

```
x_train.shape
```

```
(2100, 9)
```

```
x_test.shape
```

```
(900, 9)
```

We will now build CART model, Random forests model and ANN model.

Below is the features significance—

- Max depth- The maximum depth of the tree.
- Min Sample leaf- The minimum number of samples required to be at a leaf node.
- Min Sample split - The minimum number of samples required to split an internal node
- Max Features- The number of features to consider when looking for the best split
- N_estimators- The number of trees in the forest.
- hidden_layer_sizes : The ith element represents the number of neurons in the ith hidden layer.
- activation: {'identity', 'logistic', 'tanh', 'relu'}, default='relu'
  Activation function for the hidden layer.
- solver : {'lbfgs', 'sgd', 'adam'}, default='adam'
  The solver for weight optimization.
- max_iter : Maximum number of iterations. The solver iterates until convergence.
- tol : Tolerance for the optimization.

CART Model

-Classification and Regression Tree model is build using criterion-'gini' i.e. splitting criterion.

```
param_grid = {
    'criterion': ['gini'],
    'max_depth': [10,20,30],
    'min_samples_leaf': [20,60,100],
    'min_samples_split': [60,180,300,400],
}

dtcl = DecisionTreeClassifier(random_state=1)

grid_search_dt = GridSearchCV(estimator = dtcl, param_grid = param_grid, cv = 10)
```

```
grid_search_dt.fit(x_train,train_labels)
```

```
GridSearchCV(cv=10, estimator=DecisionTreeClassifier(random_state=1),
             param_grid={'criterion': ['gini'], 'max_depth': [10, 20, 30],
                         'min_samples_leaf': [20, 60, 100],
                         'min_samples_split': [60, 180, 300, 400]})
```

```
grid_search_dt.best_params_
```

```
{'criterion': 'gini',
 'max_depth': 10,
 'min_samples_leaf': 60,
 'min_samples_split': 180}
```

```
grid_search_dt.best_estimator_
```

```
DecisionTreeClassifier(max_depth=10, min_samples_leaf=60, min_samples_split=180,
                       random_state=1)
```

```
dt = grid_search_dt.best_estimator_
```

```
ytrain_predict_dt = dt.predict(x_train)
ytest_predict_dt = dt.predict(x_test)
```

- After running GridSearchCV we get best parameters as-

```
{'criterion': 'gini',
        'max_depth': 10,
        'min_samples_leaf': 60,
        'min_samples_split': 180}
```

- **Feature Importance**:

	Imp
Agency_Code	0.612617
Sales	0.261442
Product Name	0.057805
Duration	0.040785
Age	0.019771
Type	0.007580
Commision	0.000000
Channel	0.000000
Destination	0.000000

-Agency Code is the most import feature used for classification of data

Random Forest Model:

- Group of Trees are formed to decide the best performing model

```
param_grid_rfcl = {
    'max_depth': [6],
    'max_features': [3],
    'min_samples_leaf': [8],
    'min_samples_split': [46],
    'n_estimators': [350]
}

rfcl = RandomForestClassifier(random_state=1)

grid_search_rfcl = GridSearchCV(estimator = rfcl, param_grid = param_grid_rfcl, cv = 10)
```

```
grid_search_rfcl.fit(x_train, train_labels)
```

```
GridSearchCV(cv=10, estimator=RandomForestClassifier(random_state=1),
             param_grid={'max_depth': [6], 'max_features': [3],
                         'min_samples_leaf': [8], 'min_samples_split': [46],
                         'n_estimators': [350]})
```

```
grid_search_rfcl.best_params_
```

```
{'max_depth': 6,
 'max_features': 3,
 'min_samples_leaf': 8,
 'min_samples_split': 46,
 'n_estimators': 350}
```

```
rfcl = grid_search_rfcl.best_estimator_
```

```
ytrain_predict_rfcl = rfcl.predict(x_train)
ytest_predict_rfcl = rfcl.predict(x_test)
```

- After running GridSearchCV we get best parameters as-

```
{'max_depth': 6,
 'max_features': 3,
 'min_samples_leaf': 8,
 'min_samples_split': 46,
 'n_estimators': 350}
```

- Feature Importance:

	Imp
Agency_Code	0.263989
Product Name	0.219047
Sales	0.161785
Commision	0.143733
Type	0.076743
Duration	0.075790
Age	0.048086
Destination	0.009889
Channel	0.000937

- Agency code the important feature.

ANN Model:

```
param_grid_nncl = {
    'hidden_layer_sizes': [50,100,200], # 50, 200
    'max_iter': [2500,3000,4000], #5000,2500
    'solver': ['adam'], #sgd
    'tol': [0.01],
}

nncl = MLPClassifier(random_state=1)

grid_search_nncl = GridSearchCV(estimator = nncl, param_grid = param_grid_nncl, cv = 10)
```

```
grid_search_nncl.fit(x_trains,train_labels)
```

```
GridSearchCV(cv=10, estimator=MLPClassifier(random_state=1),
             param_grid={'hidden_layer_sizes': [50, 100, 200],
                         'max_iter': [2500, 3000, 4000], 'solver': ['adam'],
                         'tol': [0.01]})
```

```
grid_search_nncl.best_params_
```

```
{'hidden_layer_sizes': 200, 'max_iter': 2500, 'solver': 'adam', 'tol': 0.01}
```

```
nncl = grid_search_nncl.best_estimator_
```

```
ytrain_predict_nncl = nncl.predict(x_trains)
ytest_predict_nncl = nncl.predict(x_tests)
```

After running GridSearchCV we get best parameters as-

```
{'hidden_layer_sizes': 200, 'max_iter': 2500, 'solver': 'adam', 'tol': 0.
01}
```

Feature Importance:

	Imp
Agency_Code	0.263989
Product Name	0.219047
Sales	0.161785
Commision	0.143733
Type	0.076743
Duration	0.075790
Age	0.048086
Destination	0.009889
Channel	0.000937

- Agency Code is the most important feature used for classification of 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.

CART:

Classification report

```
print(classification_report(train_labels,ytrain_predict_dt))
```

```
              precision    recall  f1-score   support

           0       0.83      0.88      0.85      1471
           1       0.67      0.57      0.62       629

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

```
print(classification_report(test_labels,ytest_predict_dt))
```

```
              precision    recall  f1-score   support

           0       0.78      0.91      0.84       605
           1       0.73      0.48      0.58       295

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

Confusion Matrix

```
confusion_matrix(train_labels, ytrain_predict_dt)
```

```
array([[1293,  178],
       [ 269,  360]], dtype=int64)
```

```
confusion_matrix(test_labels, ytest_predict_dt)
```

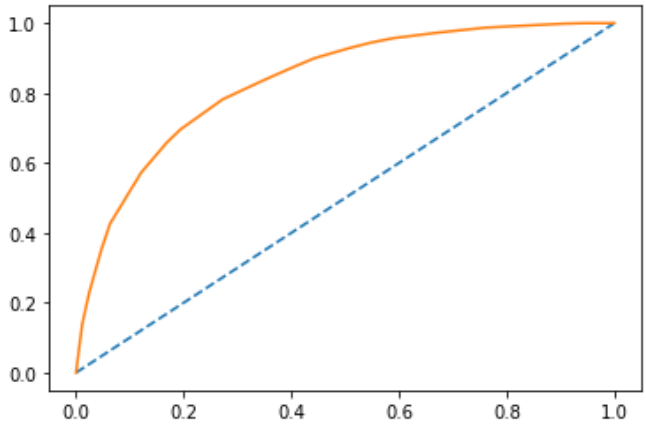
```
array([[551,  54],
       [152, 143]], dtype=int64)
```

- **Accuracy, AUC, Precision and Recall for test data is almost inline with training data. This proves no overfitting or underfitting has happened, and overall the model is a good model for classification**

- **For Train data AUC is 0.83 and for test it is 0.80.**

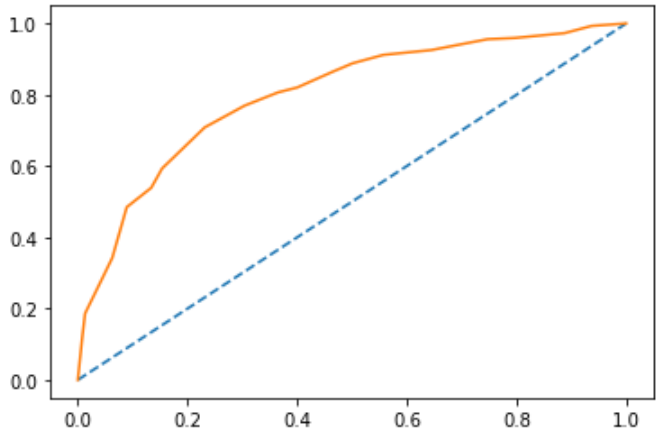
Train Data-

```
AUC: 0.834

[<matplotlib.lines.Line2D at 0x2c6d01003a0>]
```



Test Data:

```
AUC: 0.800

[<matplotlib.lines.Line2D at 0x2c6cff5da60>]
```



Accurancy score

```
dt_train_acc=dt.score(x_train,train_labels)
dt_train_acc
```

```
0.7871428571428571
```

```
dt_test_acc=dt.score(x_test,test_labels)
dt_test_acc
```

```
0.7711111111111111
```

Random Forests:

Classification report

```
print(classification_report(train_labels,ytrain_predict_rfcl))
```

```
              precision    recall  f1-score   support

           0       0.84      0.90      0.87      1471
           1       0.73      0.59      0.65       629

    accuracy                           0.81      2100
   macro avg       0.78      0.75      0.76      2100
weighted avg       0.80      0.81      0.80      2100
```

```
print(classification_report(test_labels,ytest_predict_rfcl))
```

```
              precision    recall  f1-score   support

           0       0.78      0.92      0.84       605
           1       0.74      0.48      0.58       295

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

Confusion matriix

```
print(confusion_matrix(train_labels,ytrain_predict_rfcl))
```

```
[[1331  140]
 [ 258  371]]
```

```
print(confusion_matrix(test_labels,ytest_predict_rfcl))
```

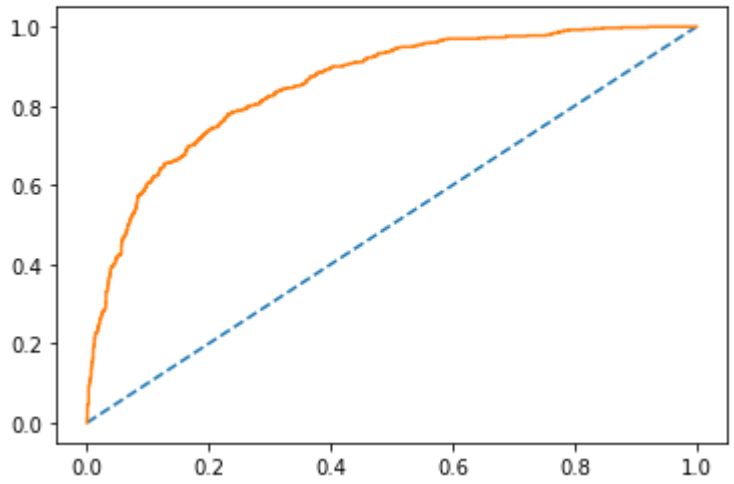
```
[[555  50]
 [154 141]]
```

- **Accuracy, AUC, Precision and Recall for test data is almost inline with training data. This proves no overfitting or underfitting has happened, and overall the model is a good model for classification**

Train Data:

```
AUC: 0.853
```

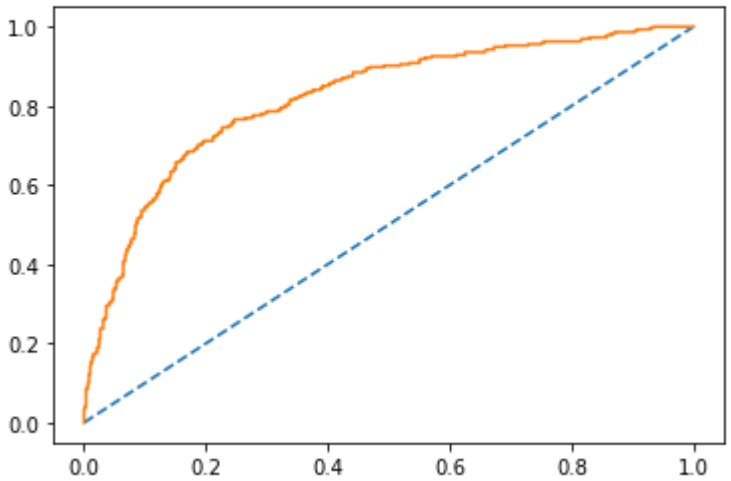
```
[<matplotlib.lines.Line2D at 0x2c6cffb8490>]
```



Test Data:

```
AUC: 0.820
```

```
[<matplotlib.lines.Line2D at 0x2c6d052ffd0>]
```



Accuracy calculation

```
rfcl_train_acc=rfcl.score(x_train,train_labels)
rfcl_train_acc
```

```
0.8104761904761905
```

```
rfcl_test_acc=rfcl.score(x_test,test_labels)
rfcl_test_acc
```

```
0.7733333333333333
```

ANN Model:

Classfication report

```
print(classification_report(train_labels,ytrain_predict_nncl))
```

```
              precision    recall  f1-score   support

           0       0.81      0.89      0.85      1471
           1       0.67      0.51      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
```

```
print(classification_report(test_labels,ytest_predict_nncl))
```

```
              precision    recall  f1-score   support

           0       0.77      0.92      0.84       605
           1       0.72      0.43      0.54       295

    accuracy                           0.76       900
   macro avg       0.75      0.68      0.69       900
weighted avg       0.75      0.76      0.74       900
```

Confusion matrix

```
print(confusion_matrix(train_labels,ytrain_predict_nncl))
```

```
[[1311  160]
 [ 311  318]]
```

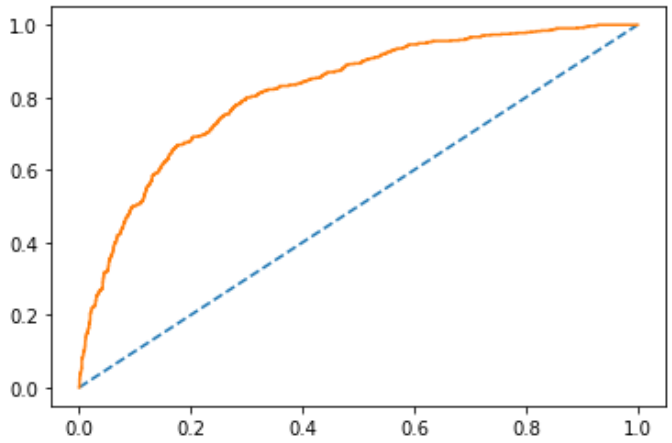
```
print(confusion_matrix(test_labels,ytest_predict_nncl))
```

```
[[556  49]
 [167 128]]
```

- **Accuracy, AUC, Precision and Recall for test data is almost inline with training data. This proves no overfitting or underfitting has happened, and overall the model is a good model for classification**

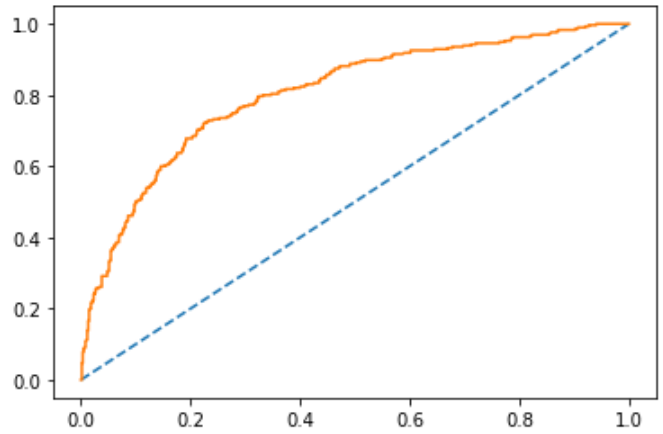
Train Data:

```
AUC: 0.818

[<matplotlib.lines.Line2D at 0x2c6d058c880>]
```



Test Data:

```
AUC: 0.804

[<matplotlib.lines.Line2D at 0x2c6d01132b0>]
```



Accuracy Score:

```
nncl_train_acc=nncl.score(x_trains,train_labels)
nncl_train_acc
```

```
0.7757142857142857
```

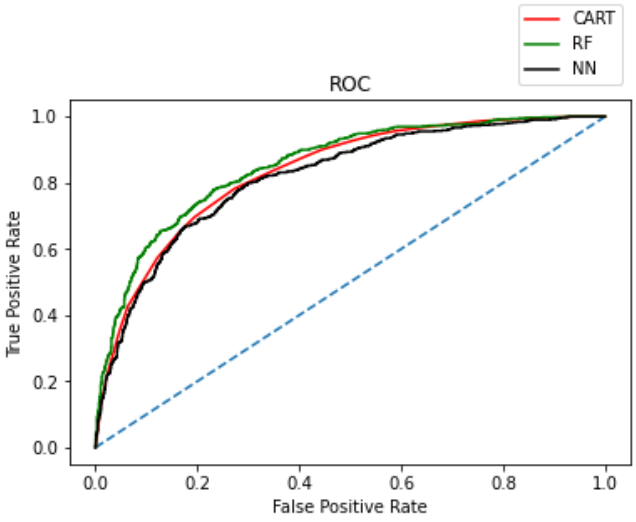
```
nncl_test_acc=nncl.score(x_tests,test_labels)
nncl_test_acc
```

```
0.76
```

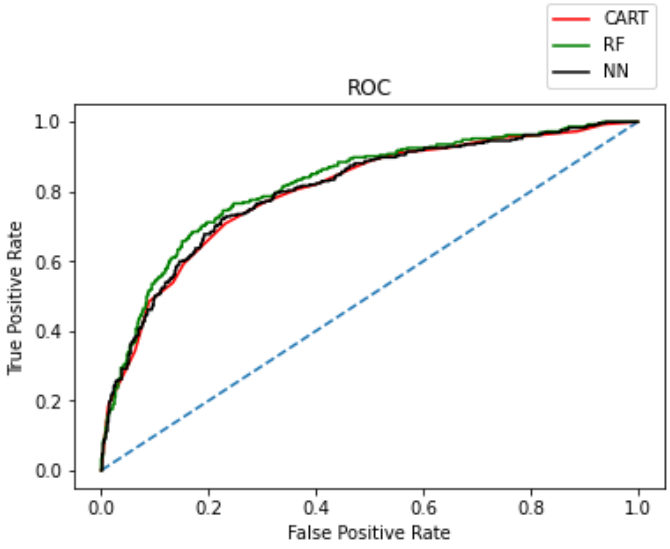
2.4 Final Model: Compare all the models and write an inference which model is best/optimized.

Below is the ROC curve for each model-

Train Data:



Test Data:



-**Performance parameters for CART, Random Forest & Neural Network classifier model**

	CART Train	CART Test	Random Forest Train	Random Forest Test	Neural Network Train	Neural Network Test
Accuracy	0.79	0.77	0.81	0.77	0.78	0.76
AUC	0.83	0.80	0.85	0.82	0.82	0.80
Recall	0.57	0.48	0.59	0.48	0.51	0.43
Precision	0.67	0.73	0.73	0.74	0.67	0.72
F1 Score	0.62	0.58	0.65	0.58	0.57	0.54

- Out of the 3 models, Random Forest has slightly better performance than the Cart and Neural network model.

- Also , AUC for RF model is better compared to CART and ANN model.

- Overall, all the 3 models are reasonably stable enough to be used for making any future predictions.

- From Cart and Random Forest Model, the variable Agency Code is found to be the most useful feature amongst all other features for predicting.

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

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

-Number of claims are High through channel 'Online' as 90% of insurance is done by Online channel.

-Also, its observed that it is almost all the offline business has a claimed associated

-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.

-It's observed that 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?