BUSINESS REPORT

PGP-DSBA online [July, 2021]
Athisya Nadar
athisya@gmail.com

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).

A predictive modelling problem is a **supervised** learning since it contains a designated response variable which depends on the set of predictors. For **unsupervised** learning problems there does not exist any response.

Clustering is an unsupervised learning technique used to form homogeneous partitions out of heterogeneous observations. We can also say, clustering is a technique of grouping objects with heterogeneity between groups and homogeneity within groups. Distance calculation done to find similarity and dissimilarity.

There are two primary approaches to clustering; namely **hierarchical** and **non-hierarchical**. Hierarchical clustering can be further categorised as **agglomerative** and **divisive**. Non-hierarchical clustering involves creating buckets like **k-means** clustering.

In hierarchical clustering, the closest points are combined in a pairwise manner to form the clusters. It is an iterative procedure, where at every step of the iteration, two points or two clusters are combined to form a bigger cluster. At the end of the process all points are combined into a single cluster. The number of clusters is not predetermined.

Once at least one multiple-element cluster is formed, a process needs to be devised how to compute distances between a set of observations and a singleton, or between two sets of observations. This may be defined in various manner:

- Single Linkage: this linkage method calculates shortest distance 2 points
- Complete Linkage: this method calculates farthest distance between 2 points
- Average Linkage: Distance between two clusters is defined as the average distance between each pair of points, one from each cluster.
- Centroid Linkage: Centroid linkage between two clusters is the distance between the respective centroids.
- Ward's Linkage: an iterative method where after every merge, the distances are updated successively

choice of distance and linkage method uniquely define the outcome of a clustering procedure. It is possible that the same set of observations may be clustered in different partitions based on the choice of distance and linkage method.

In the k-means clustering process, k denotes the number of clusters, which has to be predetermined. Once k is fixed, the observations are allocated to one and only one cluster so that the closest points belong to one cluster. The cluster size is not controlled.

Since the clustering process is completely controlled by the distance between two points and distance between two clusters, the following are the Common Measures of Distance:

- Euclidean Distance
- Squared Euclidean Distance
- Manhattan Distance
- Minkowski's Distance
- Chebyshev distance

Now let us use this method for the given problem statement, the dataset consists of 210 records and 7 columns. The head of the dataset looks like this:

	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

The following data dictionary explains the variables we will be working with:

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)

The datatypes of the variables can be summarised as follows:

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

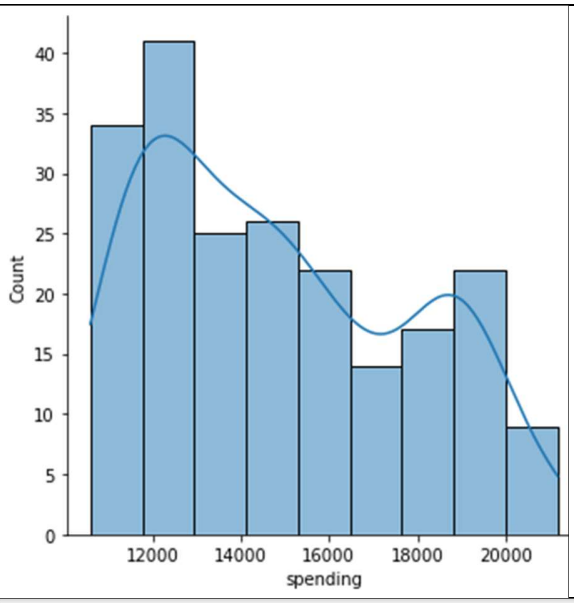
The description of the data can be summarised as follows:

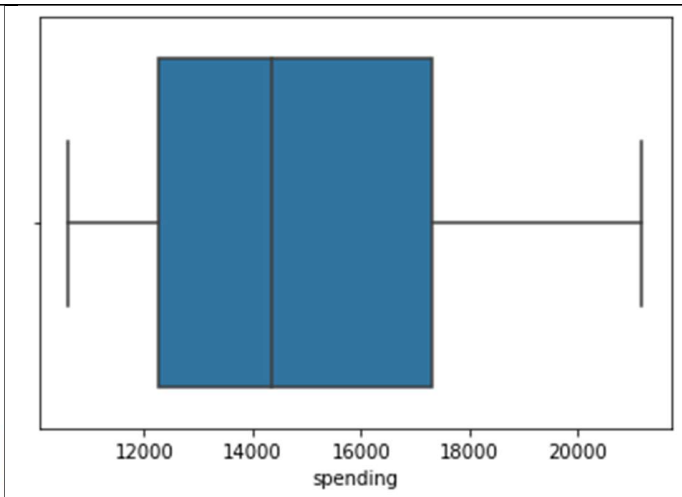
	count	mean	std	min	25%	50%	75%	max
spending	210.0	14847.523810	2909.699431	10590.0000	12270.0000	14355.00000	17305.000000	21180.0000
advance_payments	210.0	1455.928571	130.595873	1241.0000	1345.0000	1432.00000	1571.500000	1725.0000
probability_of_full_payment	210.0	0.870999	0.023629	0.8081	0.8569	0.87345	0.887775	0.9183
current_balance	210.0	5628.533333	443.063478	4899.0000	5262.2500	5523.50000	5979.750000	6675.0000
credit_limit	210.0	32586.047619	3777.144449	26300.0000	29440.0000	32370.00000	35617.500000	40330.0000
min_payment_amt	210.0	370.020095	150.355713	76.5100	256.1500	359.90000	476.875000	845.6000
max_spent_in_single_shopping	210.0	5408.071429	491.480499	4519.0000	5045.0000	5223.00000	5877.000000	6550.0000

After checking for null values, we can see there are no null values in the dataset:

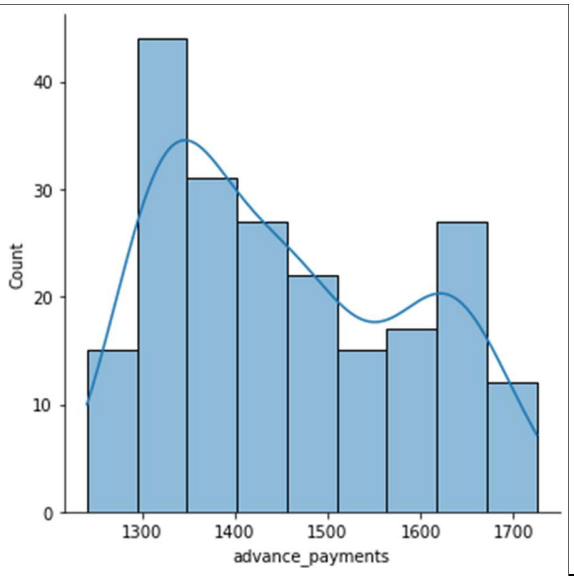
```
spending                      0
advance_payments              0
probability_of_full_payment   0
current_balance               0
credit_limit                  0
min_payment_amt               0
max_spent_in_single_shopping  0
dtype: int64
```

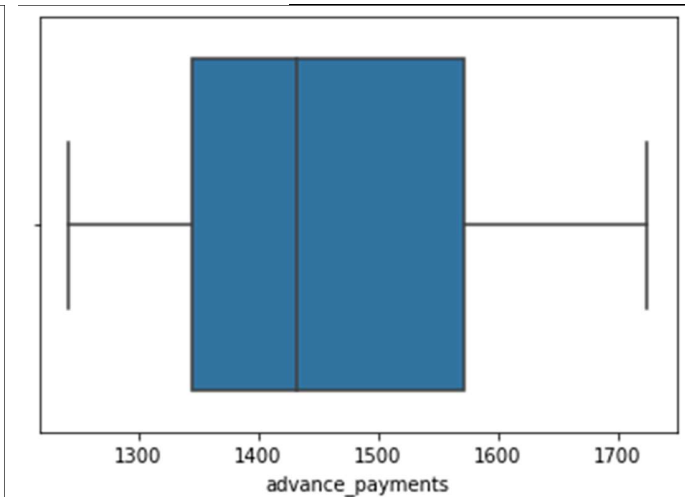
Let us begin exploratory data analysis by performing the univariate analysis to understand each variable better:



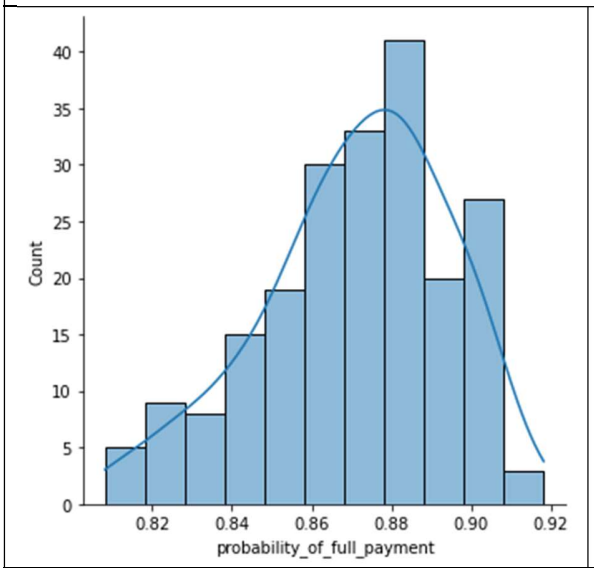


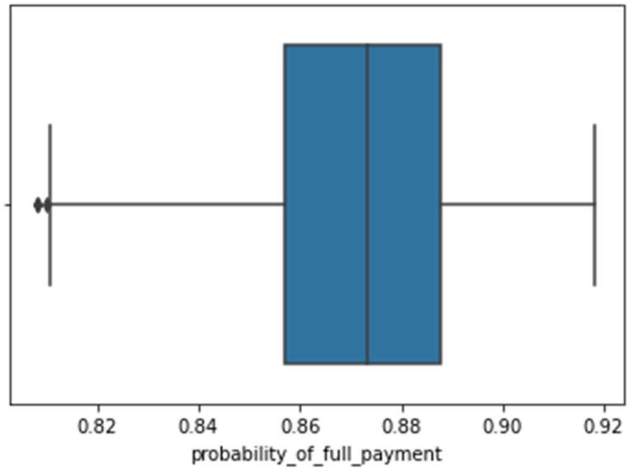

```
Description of spending
-----------------------------------------------------
count      210.000000
mean     14847.523810
std       2909.699431
min      10590.000000
25%      12270.000000
50%      14355.000000
75%      17305.000000
max      21180.000000
Name: spending, dtype: float64 Distribution of spending
-----------------------------------------------------
```



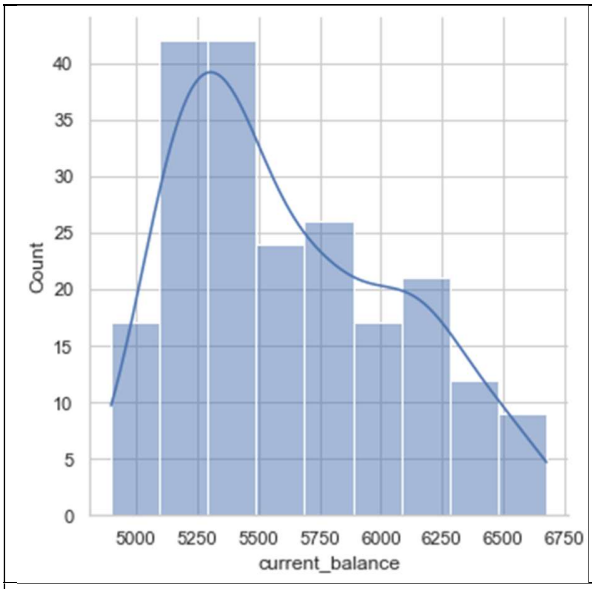


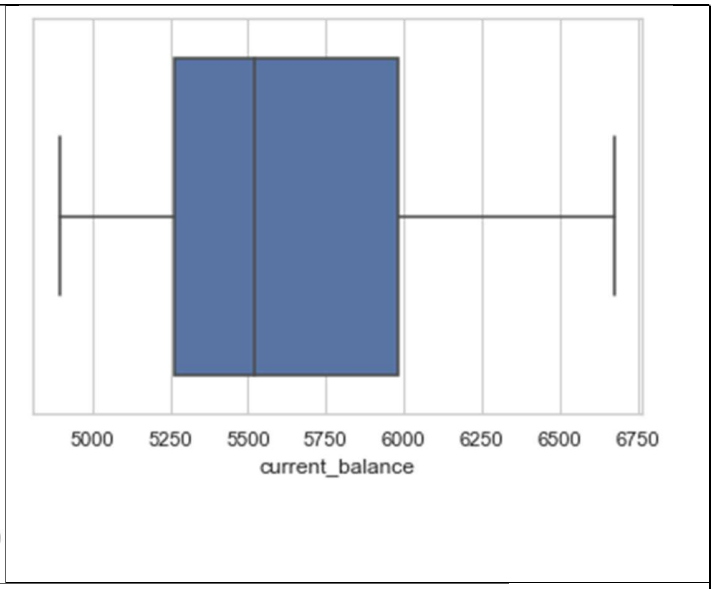
```
Description of advance_payments
-------------------------------------------------------------------------
count     210.000000
mean     1455.928571
std       130.595873
min      1241.000000
25%      1345.000000
50%      1432.000000
75%      1571.500000
max      1725.000000
Name: advance_payments, dtype: float64 Distribution of advance_payments
-------------------------------------------------------------------------
```



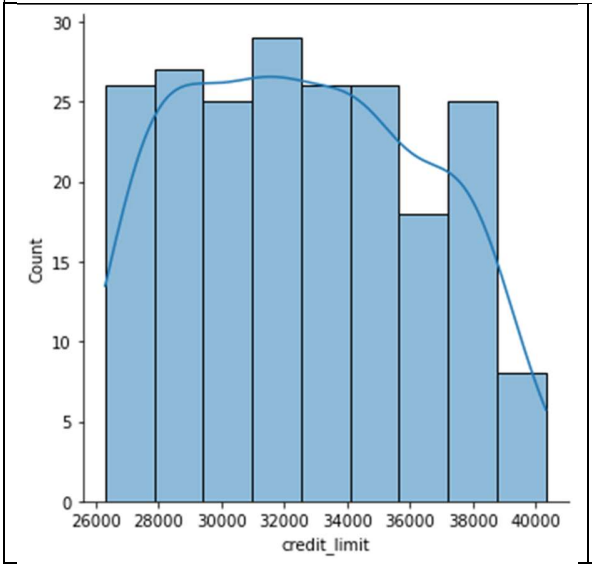


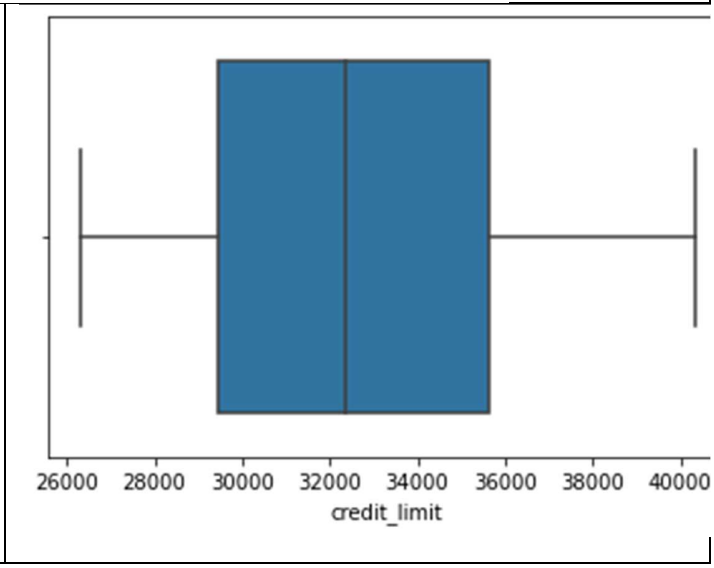

```
Description of probability_of_full_payment
-------------------------------------------------------------------------
count    210.000000
mean       0.870999
std        0.023629
min        0.808100
25%        0.856900
50%        0.873450
75%        0.887775
max        0.918300
Name: probability_of_full_payment, dtype: float64 Distribution of probability_of_full_payment
-------------------------------------------------------------------------
```



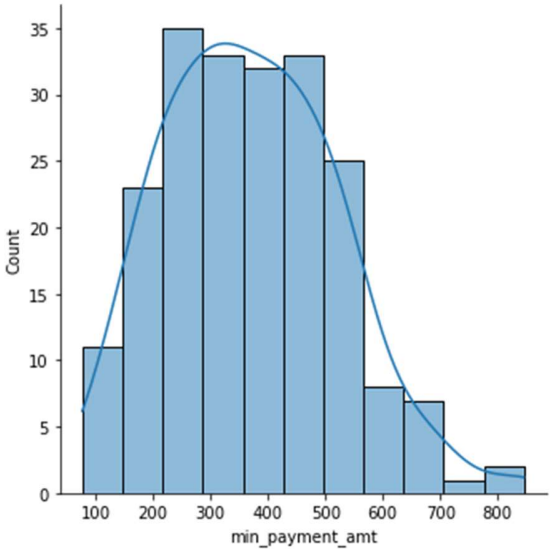


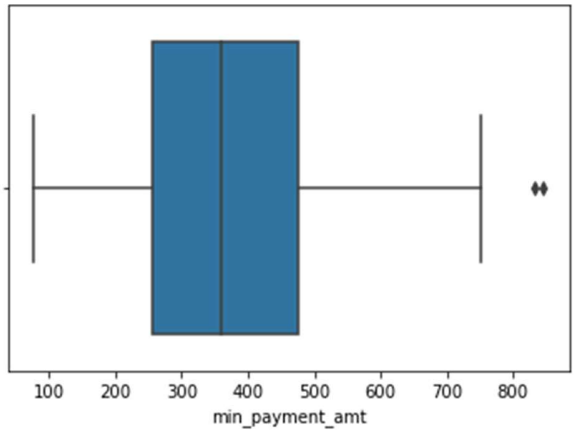

```
Description of current_balance
--------------------------------------------------------------------------------
count     210.000000
mean     5628.533333
std       443.063478
min      4899.000000
25%      5262.250000
50%      5523.500000
75%      5979.750000
max      6675.000000
Name: current_balance, dtype: float64 Distribution of current_balance
--------------------------------------------------------------------------------
```



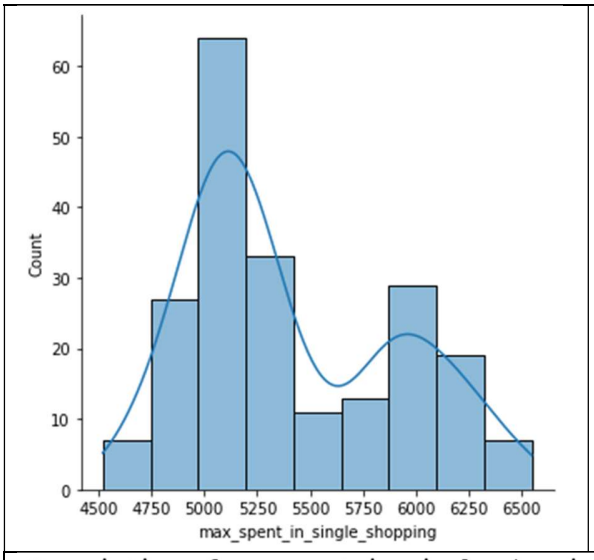


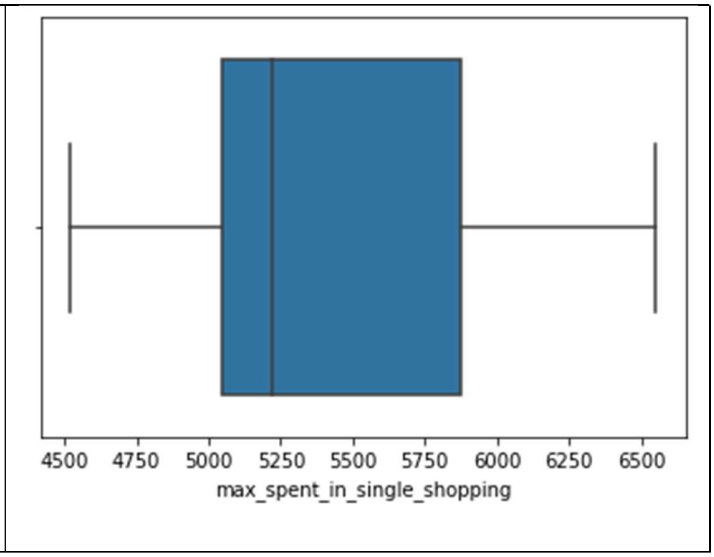
```
Description of credit_limit
-----------------------------------------------------------------
count      210.000000
mean     32586.047619
std       3777.144449
min      26300.000000
25%      29440.000000
50%      32370.000000
75%      35617.500000
max      40330.000000
Name: credit_limit, dtype: float64 Distribution of credit_limit
-----------------------------------------------------------------
```






```
Description of min_payment_amt
-----------------------------------------------------------------
count    210.000000
mean     370.020095
std      150.355713
min       76.510000
25%      256.150000
50%      359.900000
75%      476.875000
max      845.600000
Name: min_payment_amt, dtype: float64 Distribution of min_payment_amt
-----------------------------------------------------------------
```






```
Description of max_spent_in_single_shopping
-------------------------------------------------------------------------------
count     210.000000
mean     5408.071429
std       491.480499
min      4519.000000
25%      5045.000000
50%      5223.000000
75%      5877.000000
max      6550.000000
Name: max_spent_in_single_shopping, dtype: float64 Distribution of max_spent_in_single_shoppin
-------------------------------------------------------------------------------
```

Let us look at the skewness now:

```
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
```

A normal distribution has a skew of zero

As a general rule of thumb:

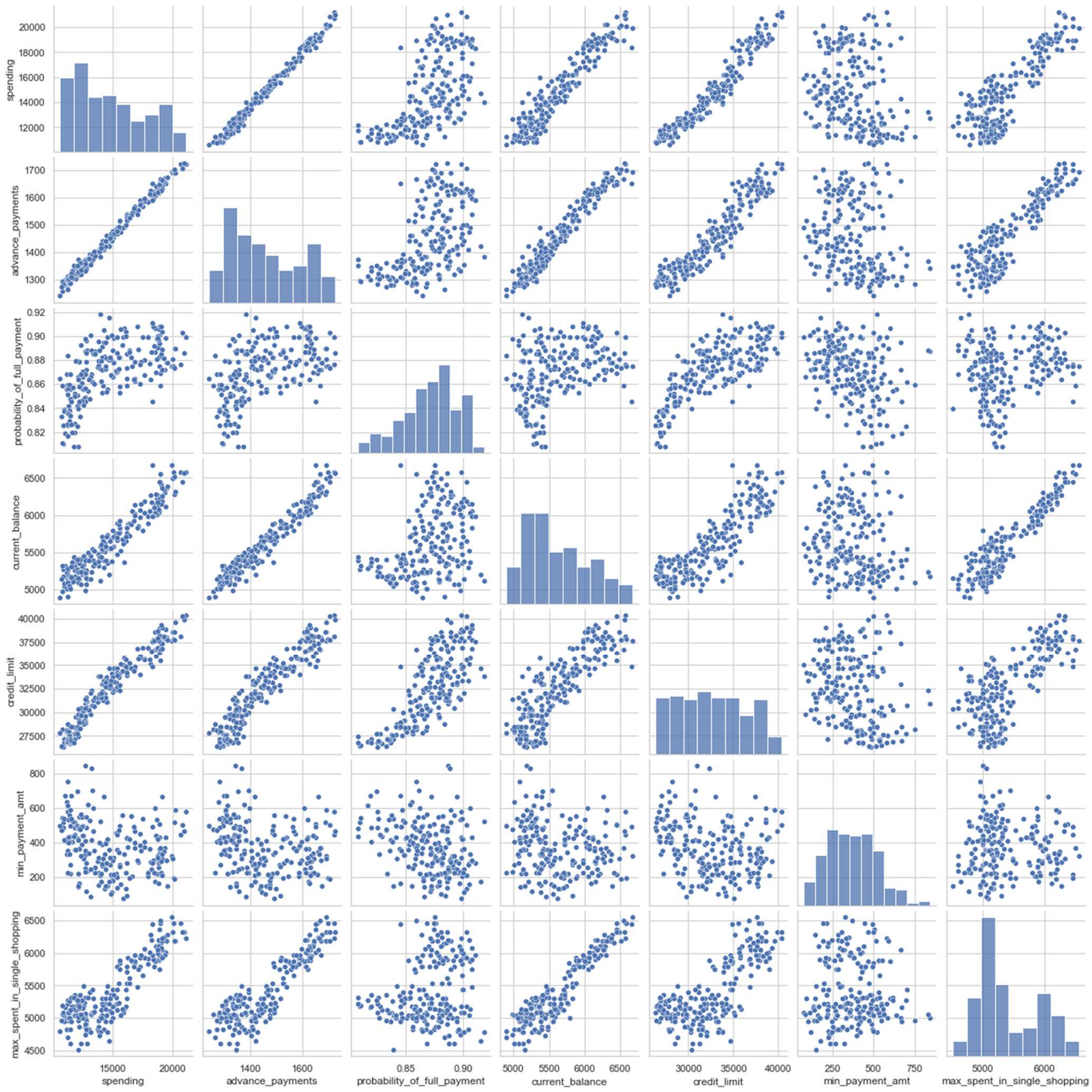
If skewness is less than -1 or greater than 1, the distribution is highly skewed.

If skewness is between -1 and -0.5 or between 0.5 and 1, the distribution is moderately skewed.

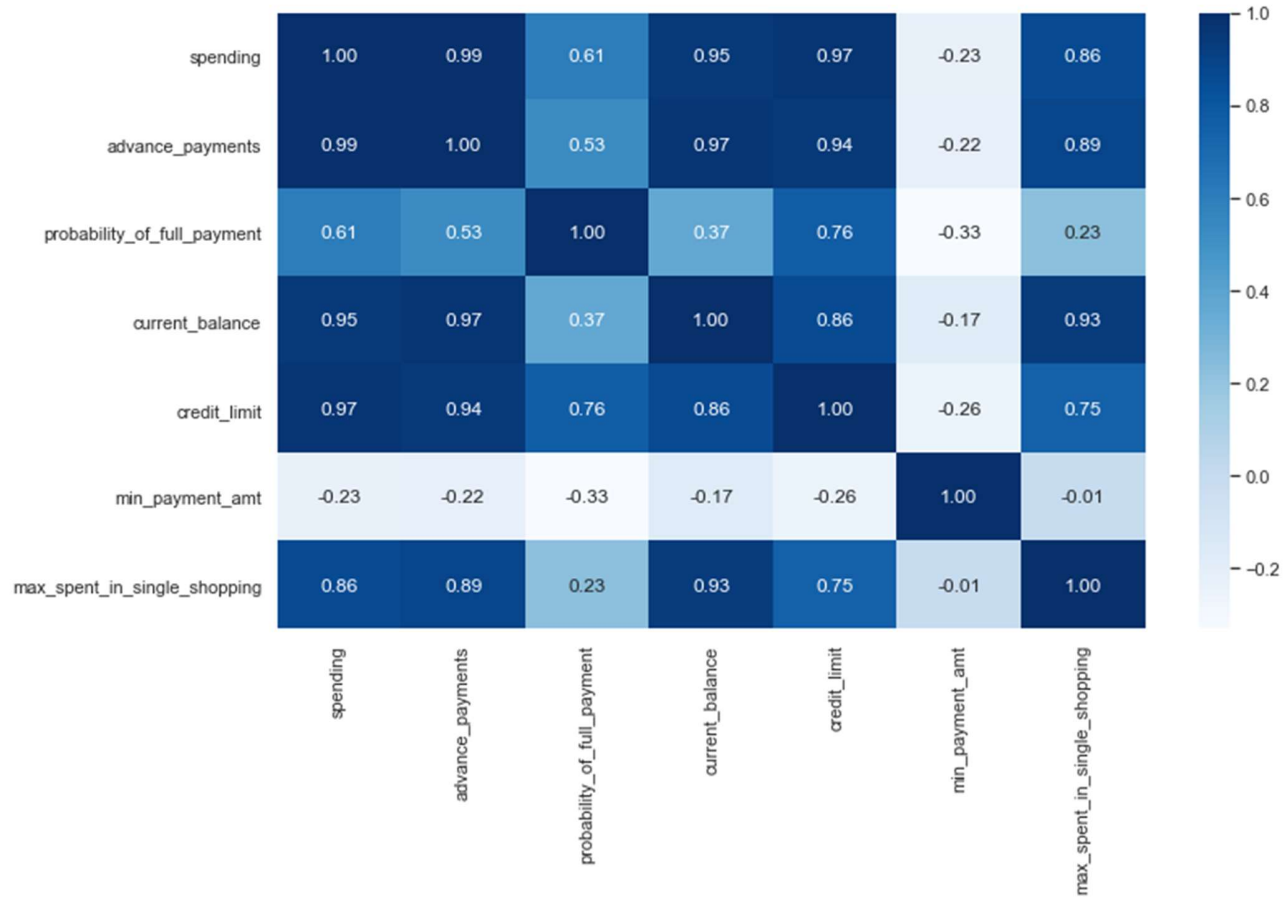
If skewness is between -0.5 and 0.5, the distribution is approximately symmetric.

We can now perform the bivariate analysis:

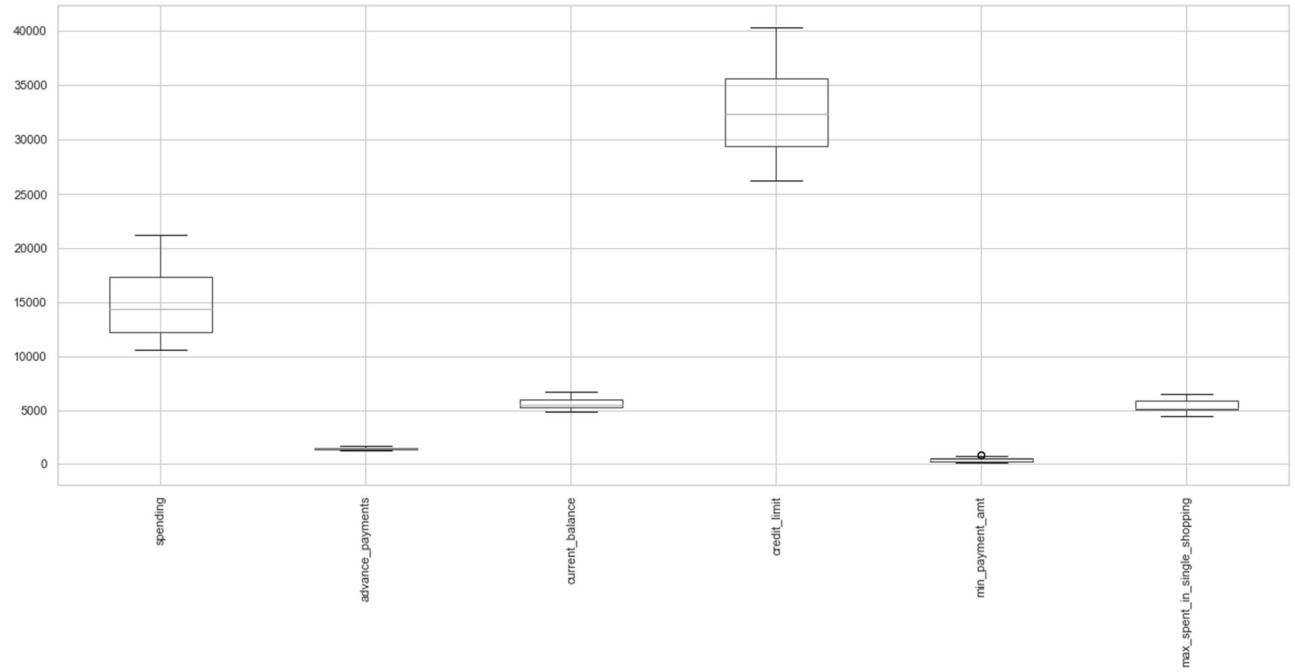
Pair plot is used to understand the best set of features to explain a relationship between two variables or to form the most separated clusters. It also helps to form some simple classification models by drawing some simple lines or make linear separation in our data-set.



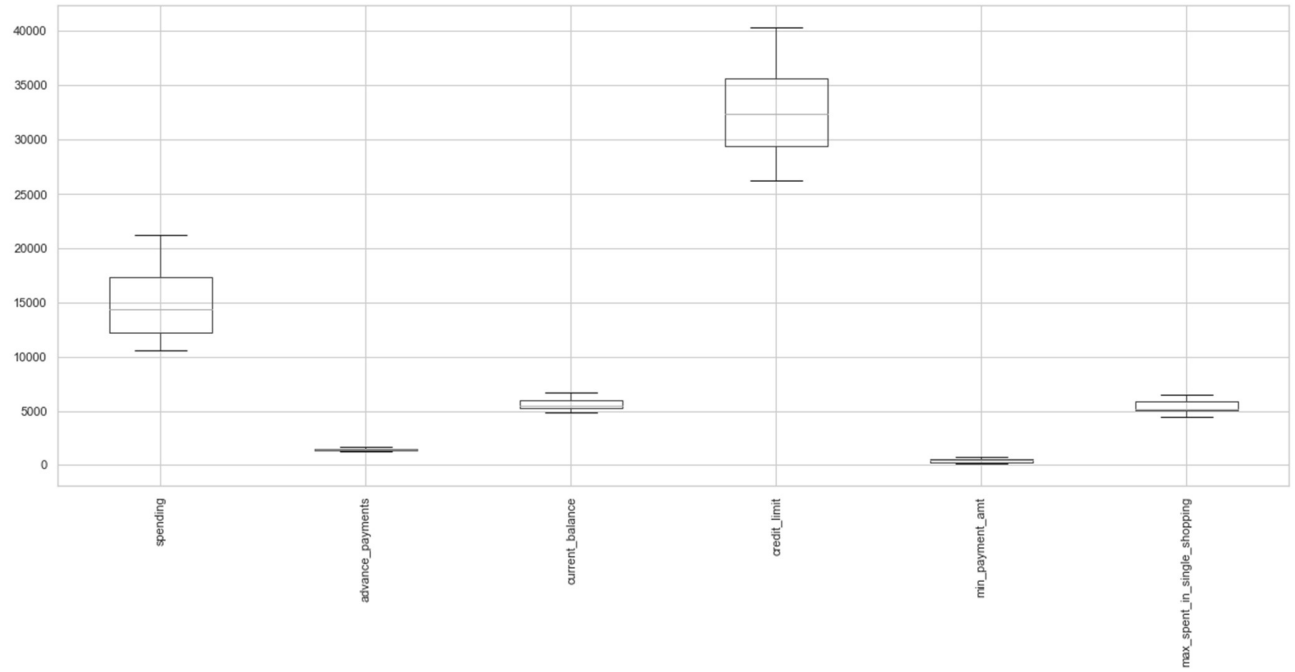
After drawing the correlation heatmap, we can see the variables spending in 1 month, advance payments, current balance, credit limit, max spent in single shopping are highly correlated while minimum payment is least correlated.



Next step is checking for outliers, we can see the minimum payment amount column has outliers, so we treat the outliers first:



After treating the outliers, by capping using the upper bound and lower bound values the dataset can me summarised in the following boxplot:



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

Feature Scaling is a technique often applied as part of data preparation for machine learning. The goal of scaling (sometimes also referred as normalization) is to change the values of numeric columns in the dataset to use a common scale, without distorting differences in the ranges of values or losing information.

In most practical cases, all different variables need to be converted to one scale in order to perform meaningful analysis.

Scaling is advised in the following situations:

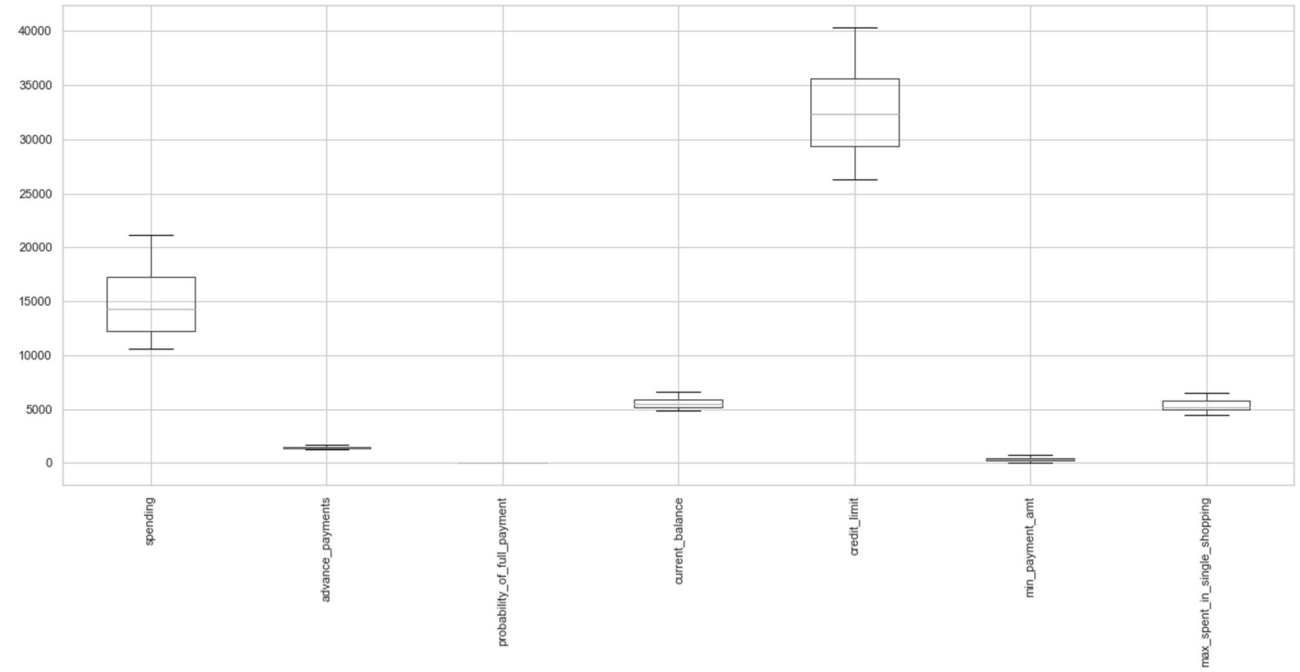
- When using **Distance or Variance** Based Algorithms: Scaling is essential in Distance based (for eg. k-nearest neighbours with a Euclidean distance measure) or variance-based models (eg. linear discriminant analysis, principal component analysis) if we want all the features to contribute equally in the final output.
- **Gradient Descent optimizations**: When the feature data value is large, scaling can be used to obtain better Convergence in lesser number of iterations and speed up the overall iterative process.

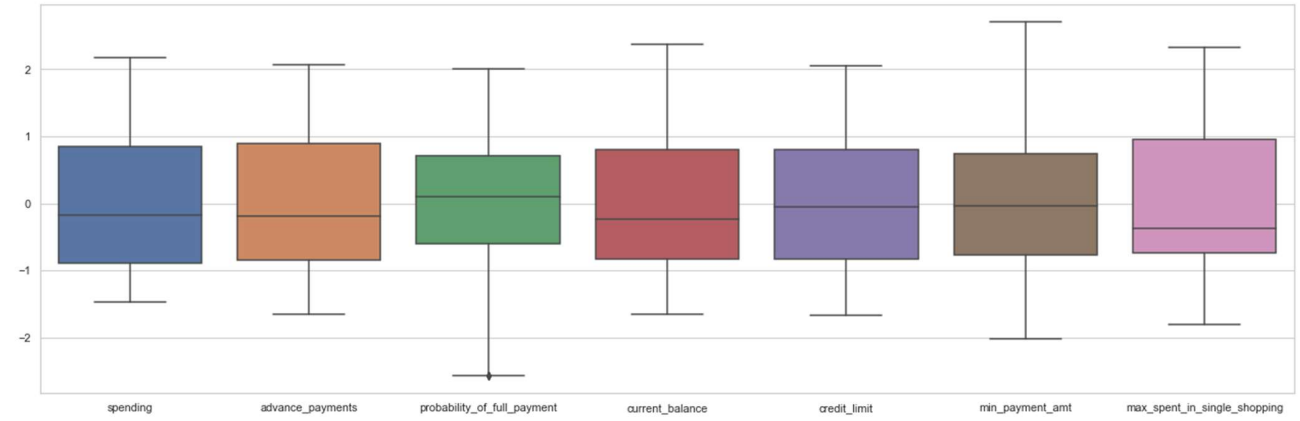
The two most commonly used scaling techniques are **Z-score** and **Min-Max** Scaling.

Z-score removes the mean i.e. brings the mean to zero and scales the data to unit variance. Z-Score, can be helpful in cases where the **data follows a Normal (or near normal) distribution** or the algorithm demands a Normal distribution of the features.

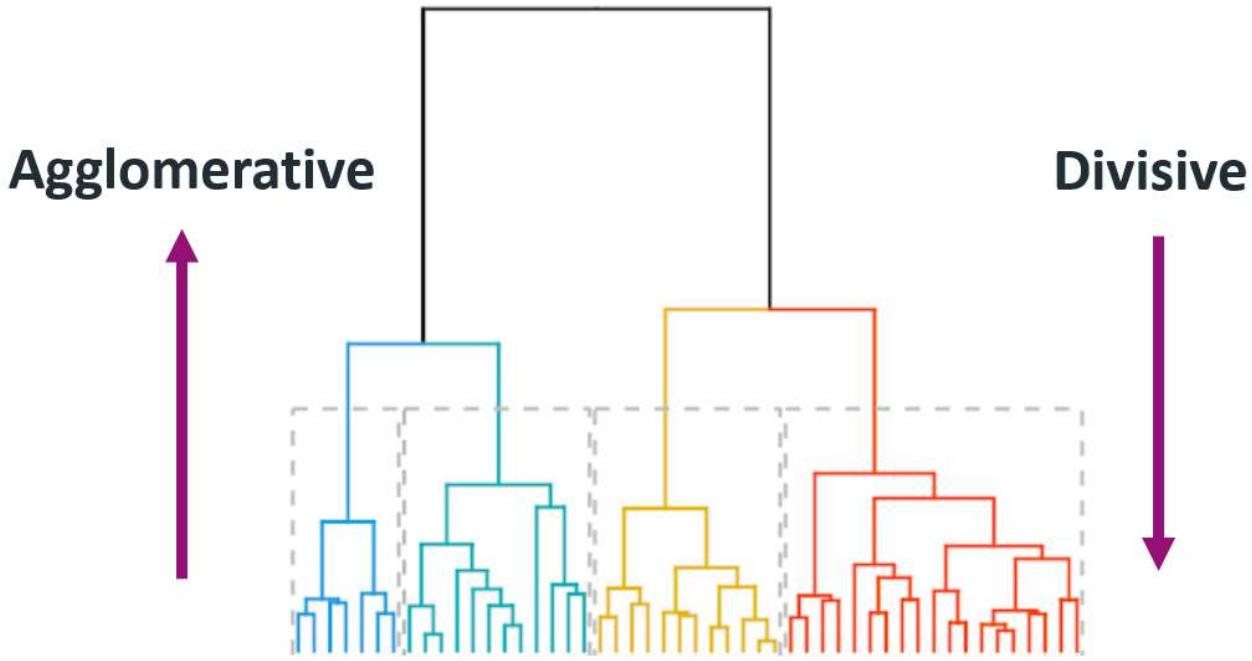
The MinMax method linearly rescales every feature to the [0,1] interval. Min-Max is good to use when you know that the distribution of your **data does not follow a Gaussian distribution**. In addition, this can be useful in algorithms that do not assume any distribution of the data like K-Nearest Neighbours and Neural Networks.

Since clustering uses distance-based algorithm, scaling is necessary for kmeans clustering and hierarchical clustering to yield better results. Kmeans uses Euclidean Distance, Squared Euclidean Distance,Manhattan Distance,Minkowski's Distance,Chebyshev distance while hierarchical clustering uses Single Linkage, Complete Linkage, Average Linkage, Centroid Linkage, Ward's Linkage. Also, scaling helps us to obtain better Convergence in lesser number of iterations and speed up the overall iterative process. The following boxplot shows the data before and after scaling:


40000
35000
30000
25000
20000
15000
10000
5000
0
spending
advance_payments
probability_of_full_payment
current_balance
credit_limit
min_payment_amt
max_spent_in_single_shopping

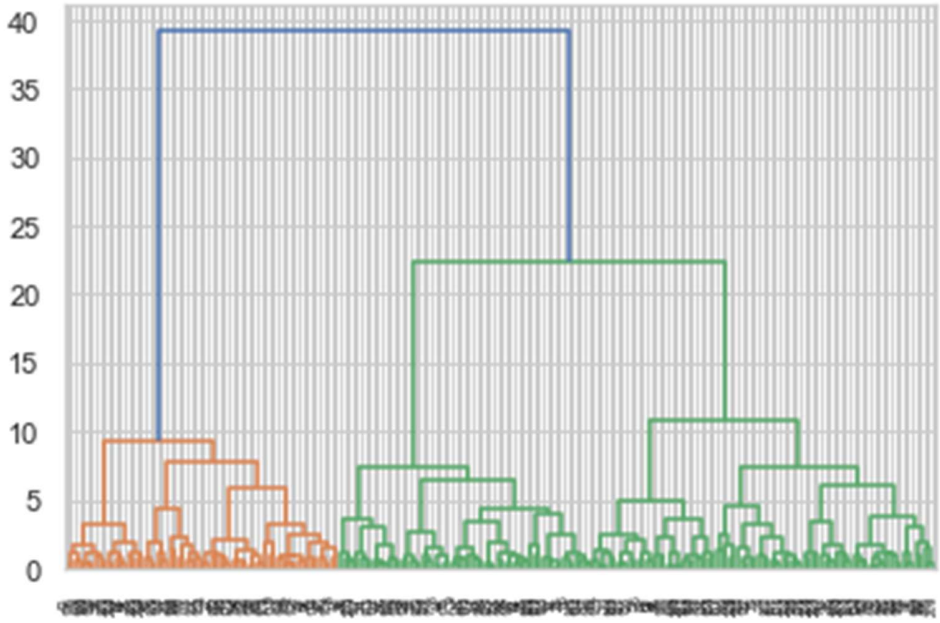

2
1
0
−1
−2
spending
advance_payments
probability_of_full_payment
current_balance
credit_limit
min_payment_amt
max_spent_in_single_shopping

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



After applying hierarchical clustering to the scaled data we obtain the following dendogram: A dendrogram is a pictorial way to visualize hierarchical clustering. It is mainly used to show the outcome of hierarchical clustering a tree like diagram that records the sequences of merges and splits.

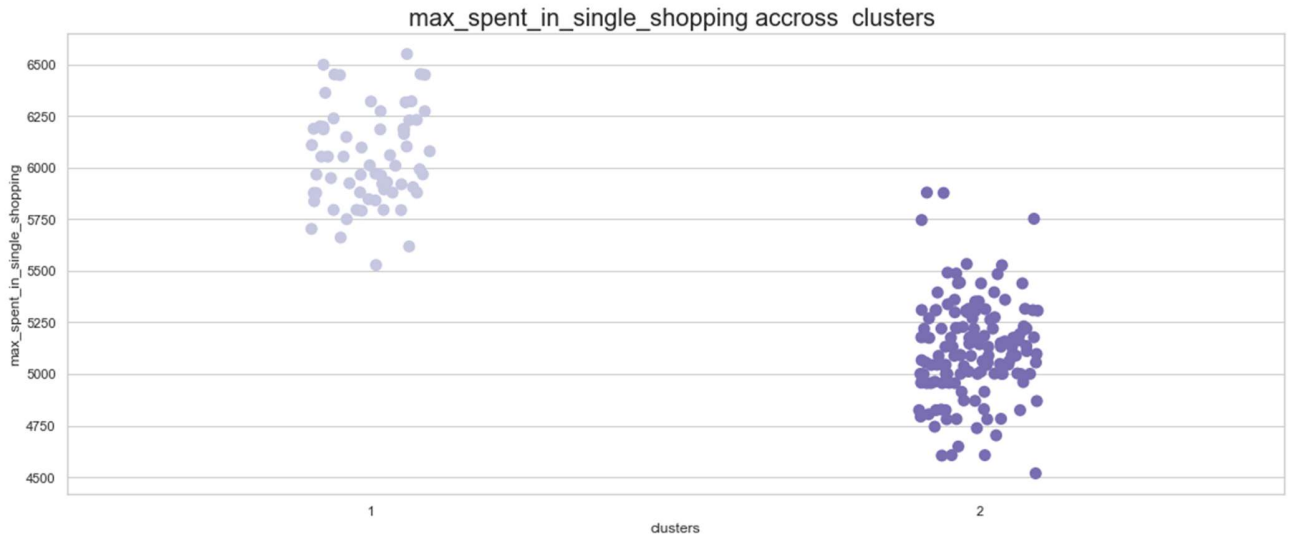
How to choose the optimal number of clusters based on the output of this analysis, the dendrogram? As a rule of thumb, look for the clusters with the longest ‘branches’, the shorter they are, the more similar they are to following ‘twigs’ and ‘leaves’. But keep in mind that, as always, the optimal number will also depend on the context, expert knowledge, application, etc.

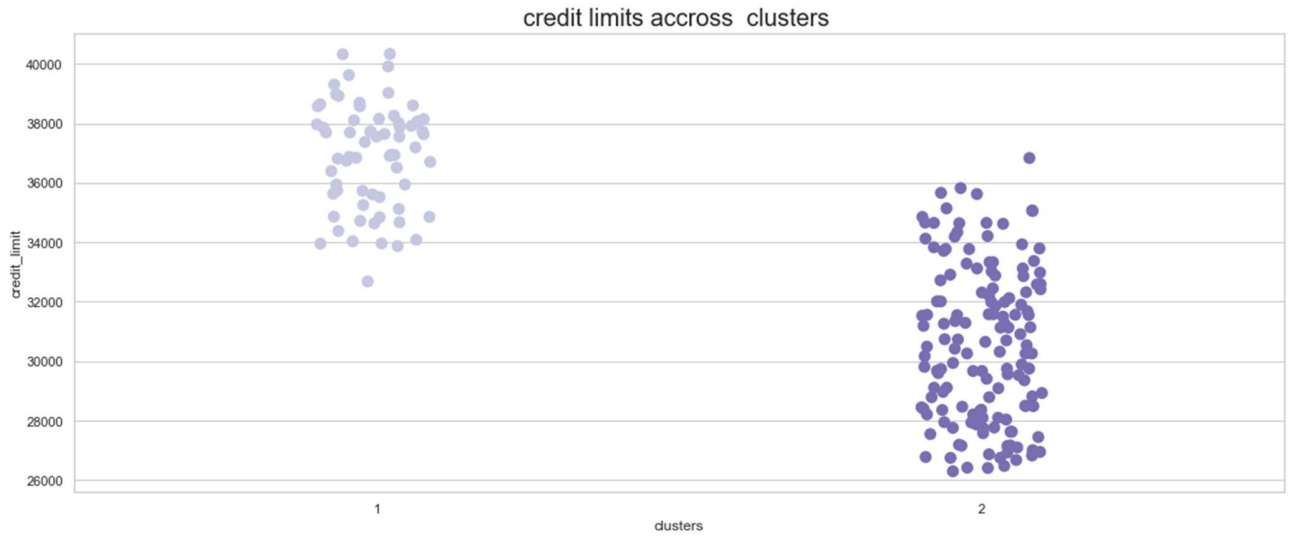


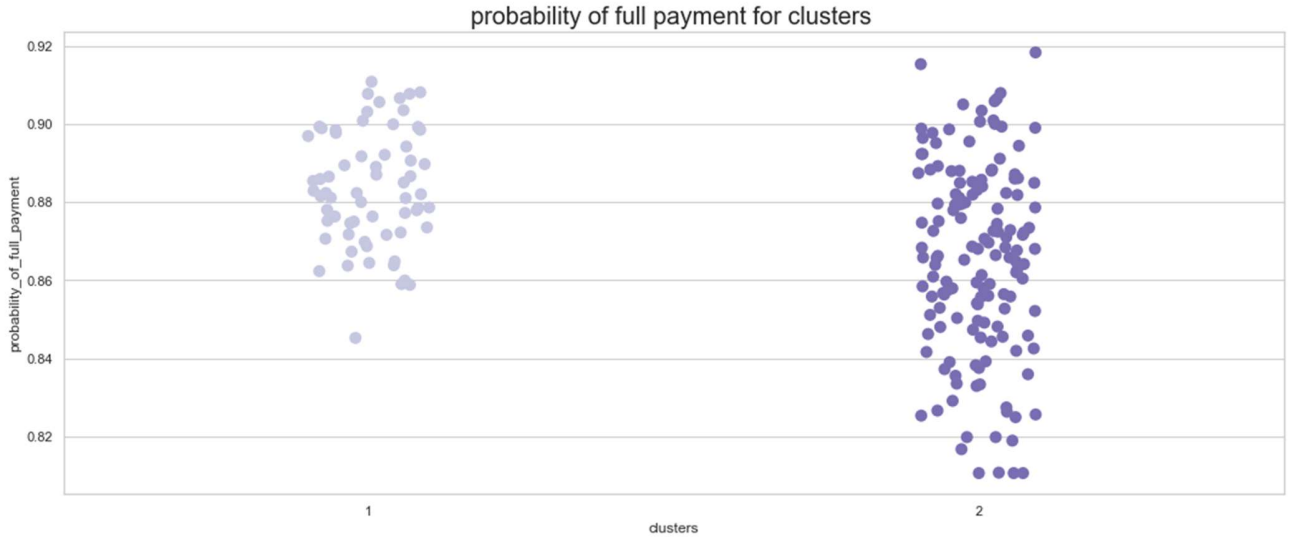
After comparing the vertical distance of k=2 and k=3, we can decide that the optimum number of clusters will be 2.

```
1     66
2    144
Name: clusters, dtype: int64
```

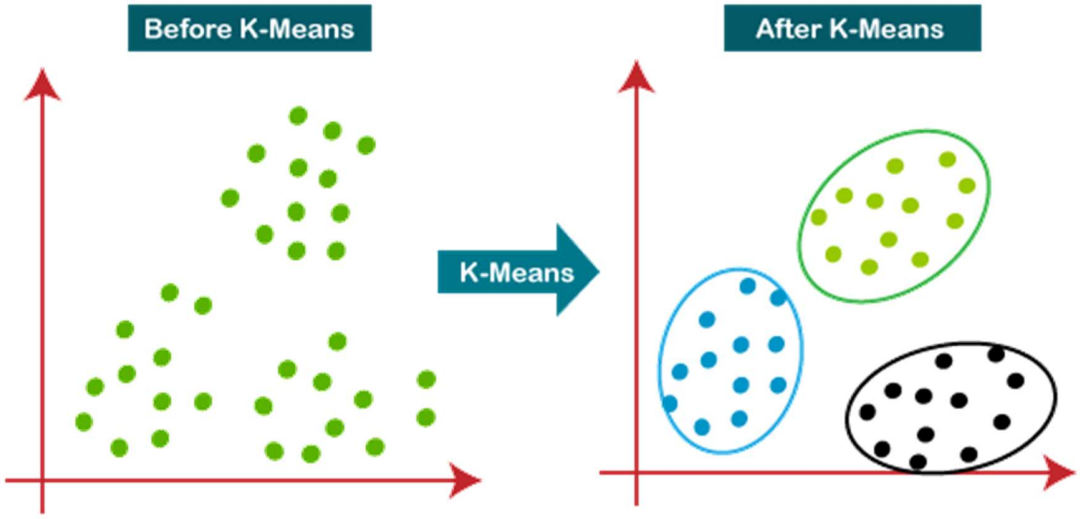
We are going to analyse the clusters patters for different variables using the scatter plots:


max_spent_in_single_shopping accross clusters
max_spent_in_single_shopping
6500
6250
6000
5750
5500
5250
5000
4750
4500
1
2
dusters


credit limits accross clusters
credit_limit
40000
38000
36000
34000
32000
30000
28000
26000
1
2
dusters


probability of full payment for clusters
probability_of_full_payment
0.92
0.90
0.88
0.86
0.84
0.82
1
2
dusters

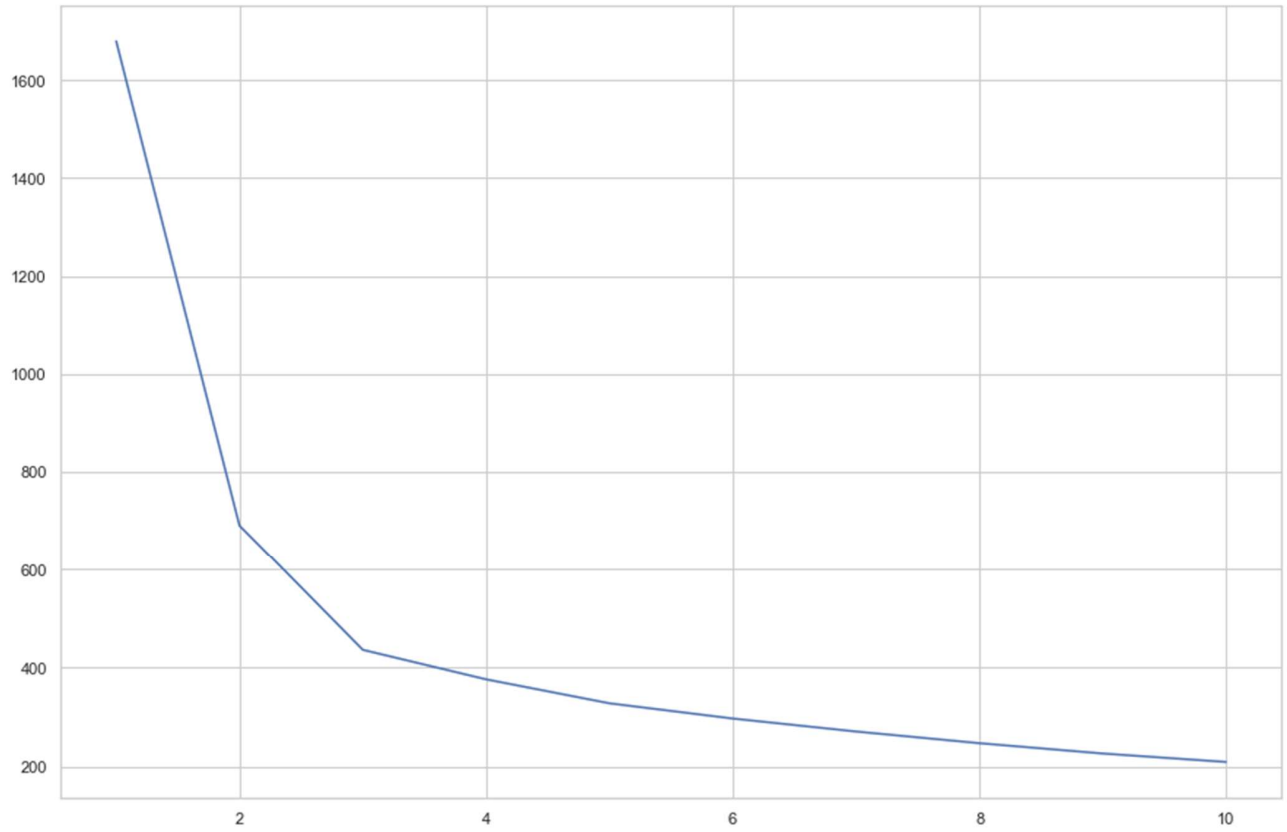
1.4 Apply K-Means clustering on scaled data and determine optimum clusters. Apply elbow curve and silhouette score. Explain the results properly. Interpret and write inferences on the finalized clusters.



After applying k-means clustering on scaled data we get the following wss plot

```
wss
```

```
[1680.0,
 689.9143787997622,
 435.98348962964315,
 376.17561135843664,
 327.59436592973555,
 296.72315794294667,
 270.4151438558264,
 246.7600877711705,
 225.84242190070634,
 208.7332928941947]
```



WSS means the sum of distances between the points and the corresponding centroids for each cluster. From this elbow plot, we check the significant drop numbers and decide 3 as the optimum number of clusters.

Elbow method: It is most popular and well-known method to find the optimal no. of clusters or the value of k in the process of clustering. This method is based of plotting the value of cost function against different values of k. As the number of clusters (k) increase lesser number of points fall within clusters or around the centroids. Hence the average distortion decreases with the increase of number of clusters. The point where the distortion declines most is said to be the elbow point and define the optimal number of clusters for dataset. As it is clear from above figure, the distortion declines most at 3. Hence the optimal value of k will be 3 for performing the clustering. In other words the plot looks as an arm with an elbow at k = 3.

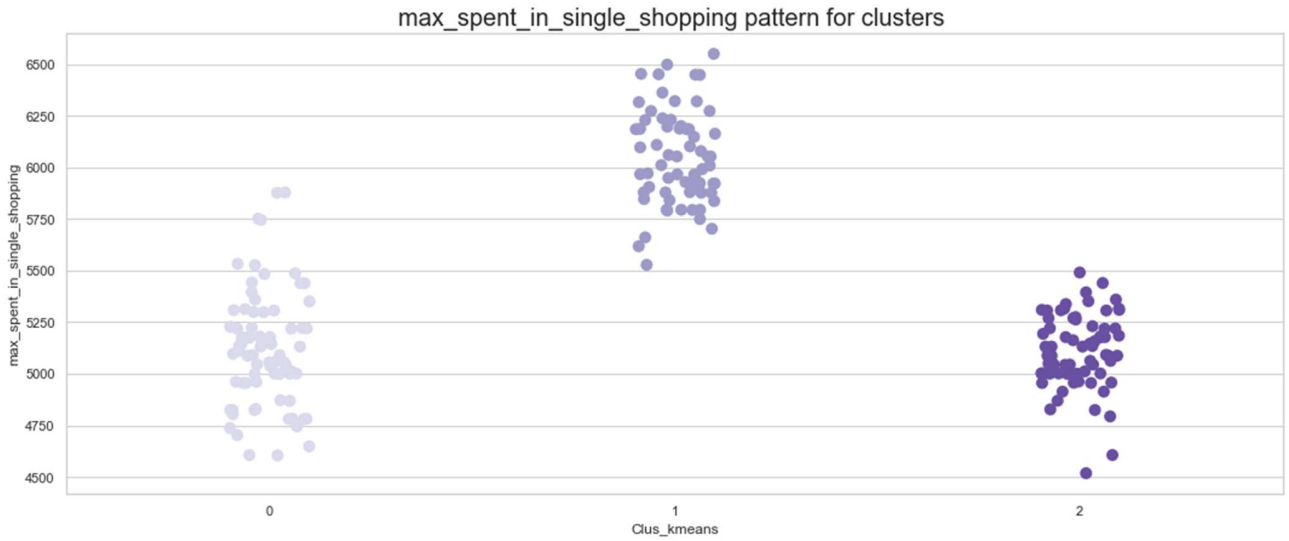
Silhouette is a different method to determine optimal number of clusters for given dataset. It defines as a coefficient of measure of how similar an observation to its own cluster compared to that of other clusters. The range of silhouette coefficient varies between -1 to 1. 1 value indicate that an observation is far from its neighbouring cluster and close to its own whereas -1 denotes that an observation is close to neighbouring cluster than its own cluster. The 0 value indicate the presence of observation on boundary of two clusters. For this dataset the silhouette score calculation is 0.4380280576697396

We obtain the following cluster counts:

```
df_data.Clus_kmeans.value_counts()
```

```
0    75
2    69
1    66
Name: Clus_kmeans, dtype: int64
```

Now we plot the cluster profiles using scatter plot:



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

After performing hierarchical clustering, we have obtained 2 cluster profiles

	Cluster 1(count = **66**) **High credit card usage customers**	**Cluster 2**(count = **144**) **Low credit card usage customers**
Average monthly spending	**18473.34**	**13185.69**

Minimum advance payment	**1486**	**1241**
Maximum advance payment	**1725**	**1555**
Average probability of full payment	**88.37%**	**86.52%**
Average current balance	**6178.74**	**5376.35**
Average credit limit	**36921.97**	**30598.75**
Minimum payment ranges between	**147.20 to 668.20**	**76.51 to 752.40**
Average of Maximum amount spent in single shopping	**6045.61**	**5115.87**

After performing kmeans clustering we have obtained following 3 cluster profiles:

	Cluster 1(count = **75**) **moderate credit card usage customers**	**Cluster 2**(count = **66**) **High credit card usage customers**	**Cluster 1**(count = **69**) **Low credit card usage customers**
Average monthly spending	14428.93	18473.33	11834.34
Minimum advance payment	1263	1486	1241
Maximum advance payment	1555	1725	1395

Average probability of full payment	88.22%	88.37%	84.66%
Average current balance	5504.40	6178	5237.17
Average credit limit	32620.93	36921	28400.72
Minimum payment ranges between	76.51 to 668.50	147.20 to 668.20	150.20 to 752.40
Average of Maximum amount spent in sing shopping	5121.60	6045.60	5109.63

The 2 clusters we identified after applying hierarchical clustering can be categorized as "**High credit card usage customers**" and "**Low credit card using customers**"

- high credit usage customer use up 50% of their credit limit on an average every month, compared to 43% of low credit card usage customers, so the financial institution can offer promotion to increase credit limit.
- customers from both clusters on an average spend 16% apprx of their credit limit on single shopping we can offer customer option to split payment into 3- or 6-months EMI on the mobile app easily
- credit cards build credit history, come with popular Rewards programs like Points systems, Frequent flyer cards, Rewards in common spending categories like restaurant dining, gas, or groceries, track your spending for you, Free borrowing....to change cluster 2 customers into high credit card usage customers we can offer easy balance transfer option so that the can usage up the current balance
- to manage risk we need a pool of customers who spend regularly but do not default on payments…. There's no better incentive then cash, introducing cashback offers on spending will encourage customers to increase spending
- similarly for the 3 clusters identified from kmeans clustering namely moderate, high, low credit card usage customers we can identify high monthly usage customers and reward them with gift vouchers and movie tickets to move the moderate and low spending customers to high spenders

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).

Classification or Decision Tree CART CLASSIFICATION AND REGRESSION TREE is a supervised learning technique which means a response variable is involved.

Classification and Regression Tree (CART): –

- Binary Decision Tree
- Classification (Categorical output variable)
- Regression (Continuous output variable)
- Uses Gini Index

CHAID – CHI-squared Automatic Interaction Detector: –

- Non-Binary Decision Tree –
- Use statistical significance of proportions

Lets now look at the dataset beginning with the head:

	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

Now let us check the data type of each variable:

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

The data consists of 3000 rows and 10 columns, out of which 6 are categorical variables and 4 continuous variables. The description can be summarised as follows:

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

Let us now look at the value counts of each categorical variable:

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

Online     2954
Offline      46
Name: Channel, dtype: int64
```

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


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


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


Travel Agency    1837
Airlines         1163
Name: Type, dtype: int64
```

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)

Let us begin the exploratory data analysis by treating anomaly in data:

	Age	Agency_Code	Type	Claimed	Commision	Channel	Duration	Sales	Product Name	Destination
1508	25	JZI	Airlines	No	6.3	Online	-1	18.0	Bronze Plan	ASIA

Since duration cannot be negative, we have imputed this anomaly with the mean of the column. Now lets check for missing values or null values:

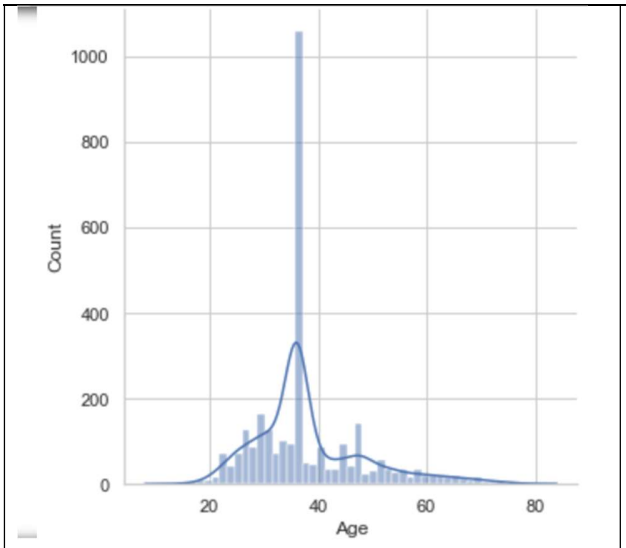
```
df_insurance.isnull().sum()
```

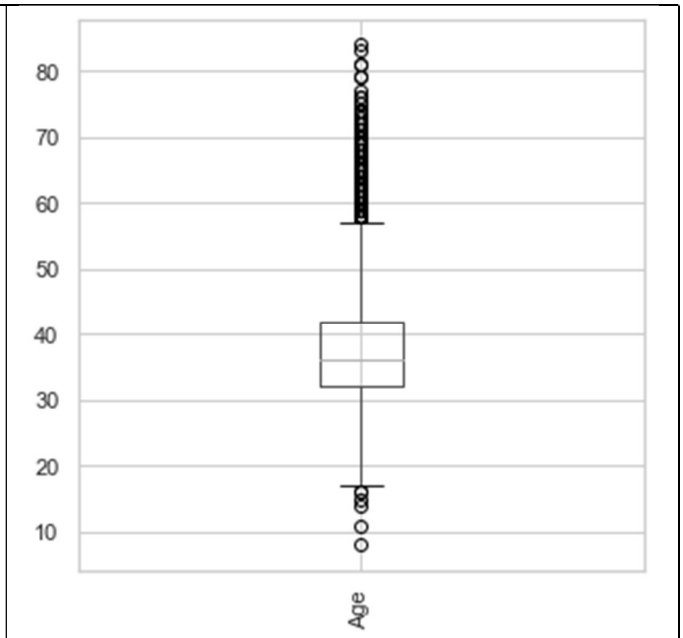
```
Age             0
Agency_Code     0
Type            0
Claimed         0
Commision       0
Channel         0
Duration        0
Sales           0
Product Name    0
Destination     0
dtype: int64
```

```
df_insurance.isna().sum()
```

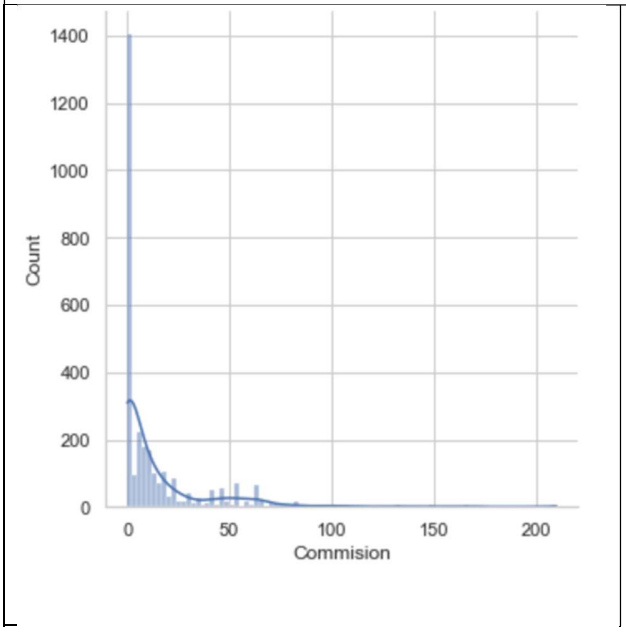
```
Age             0
Agency_Code     0
Type            0
Claimed         0
Commision       0
Channel         0
Duration        0
Sales           0
Product Name    0
Destination     0
dtype: int64
```

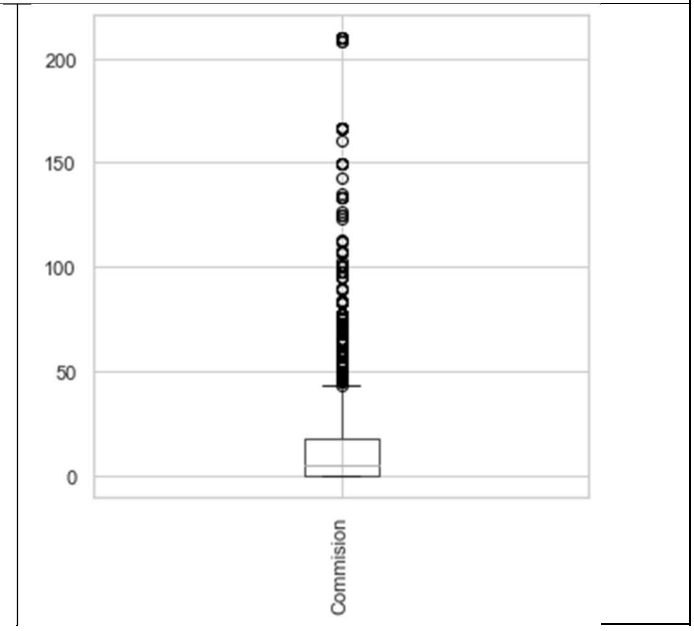
There are no null or missing values in the dataset. Next step is to perform univariate analysis:



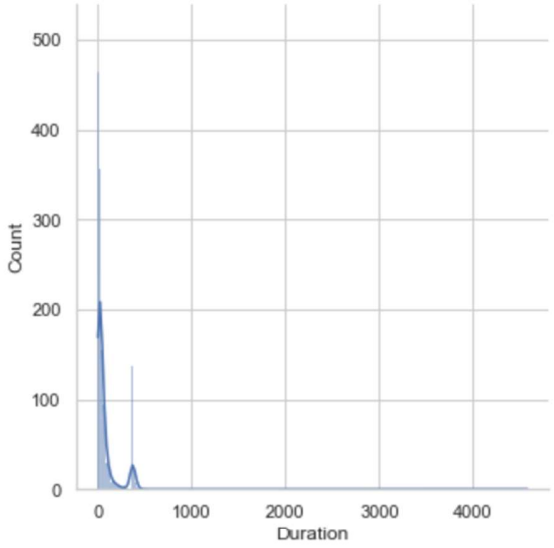


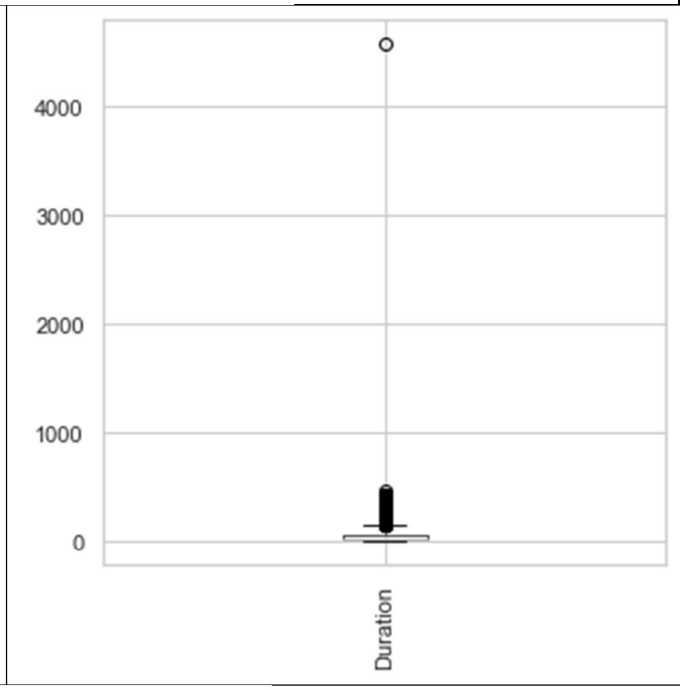

```
Description of Age
-----------------------------------------------------------------------------
count    3000.000000
mean       38.091000
std        10.463518
min         8.000000
25%        32.000000
50%        36.000000
75%        42.000000
max        84.000000
Name: Age, dtype: float64 Distribution of Age
-----------------------------------------------------------------------------
```



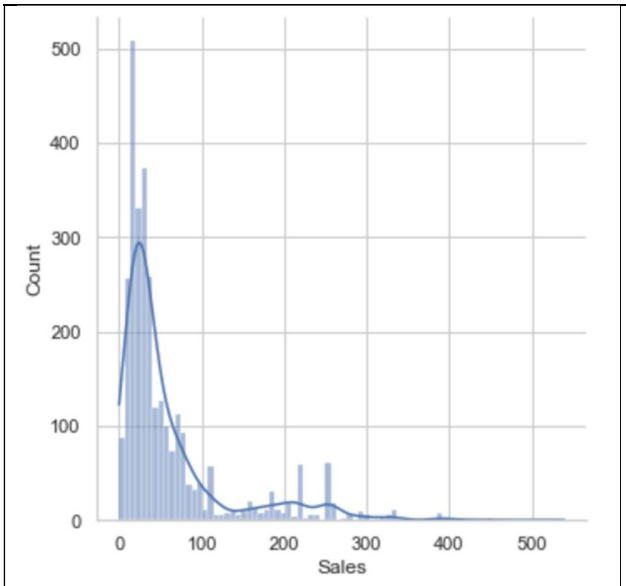


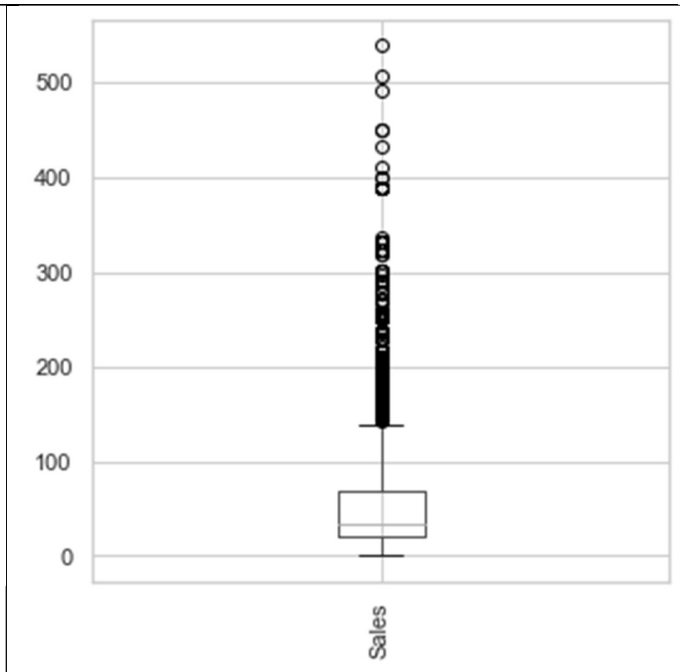
```
Description of Commision
-----------------------------------------------------------
count    3000.000000
mean       14.529203
std        25.481455
min         0.000000
25%         0.000000
50%         4.630000
75%        17.235000
max       210.210000
Name: Commision, dtype: float64 Distribution of Commision
-----------------------------------------------------------
```






```
Description of Duration
-----------------------------------------------------------
count    3000.000000
mean       70.025000
std       134.047041
min         0.000000
25%        11.000000
50%        27.000000
75%        63.000000
max      4580.000000
Name: Duration, dtype: float64 Distribution of Duration
-----------------------------------------------------------
```



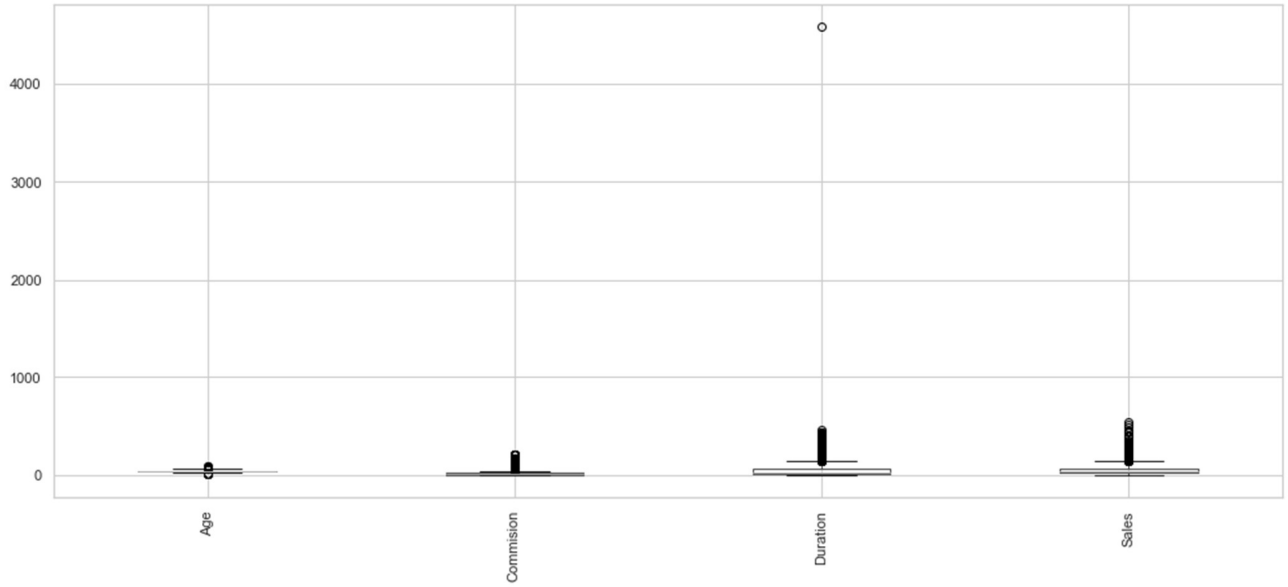



```
Description of Sales
--------------------------------------------------
count    3000.000000
mean       60.249913
std        70.733954
min         0.000000
25%        20.000000
50%        33.000000
75%        69.000000
max       539.000000
Name: Sales, dtype: float64 Distribution of Sales
--------------------------------------------------
```

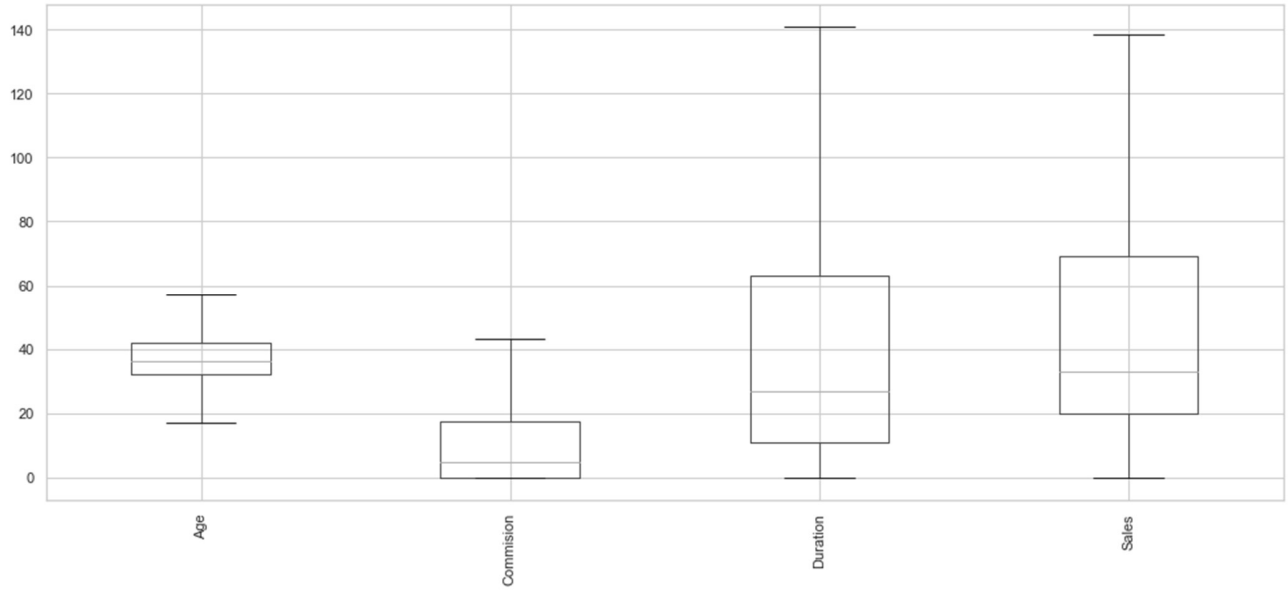
Now we can code the categorical variables to numeric representation:

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

In the next step we treat outliers by capping upper bound and lower bound values,



After treating outliers we get the following boxplot,

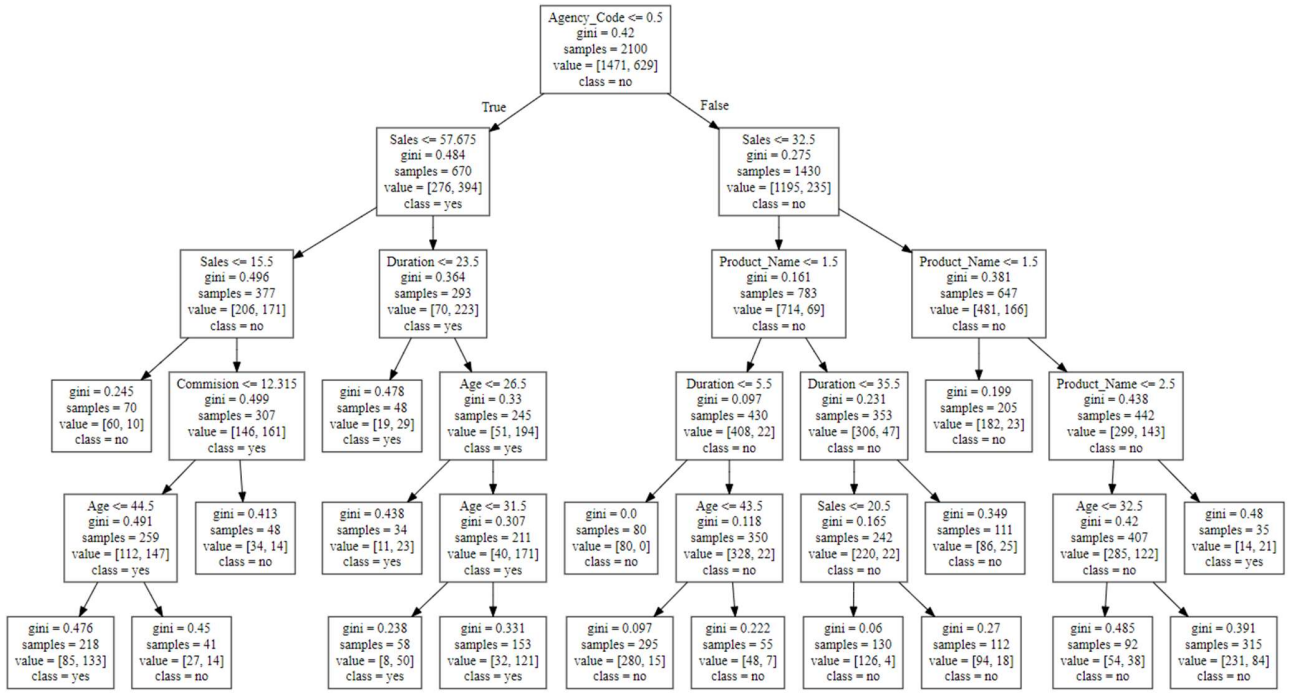


2.2 Data Split: Split the data into test and train, build classification model CART, Random Forest, Artificial Neural Network

Classification And Regression Trees (CART) is a classification technique which is part of supervised learning. In the given problem statement, the response variable is the claim status for travel insurance. We are going to build the Cart model to understand which feature has the most impact on the claim status for insurance. CART works on Gini gain concept; root node has the lowest gini index. It further branches out into parent and child nodes.

$$Gini = 1 - \sum_{j} p_j^2$$

After building the CART model, since it is a white box model, we can summarize the data as follows:



CART CONCLUSION:

Train Data:

AUC: 83.4%

Accuracy: 80% (0.7995238095238095)

Precision: 69%

f1-Score: 64%

Test Data:

AUC: 80.3%

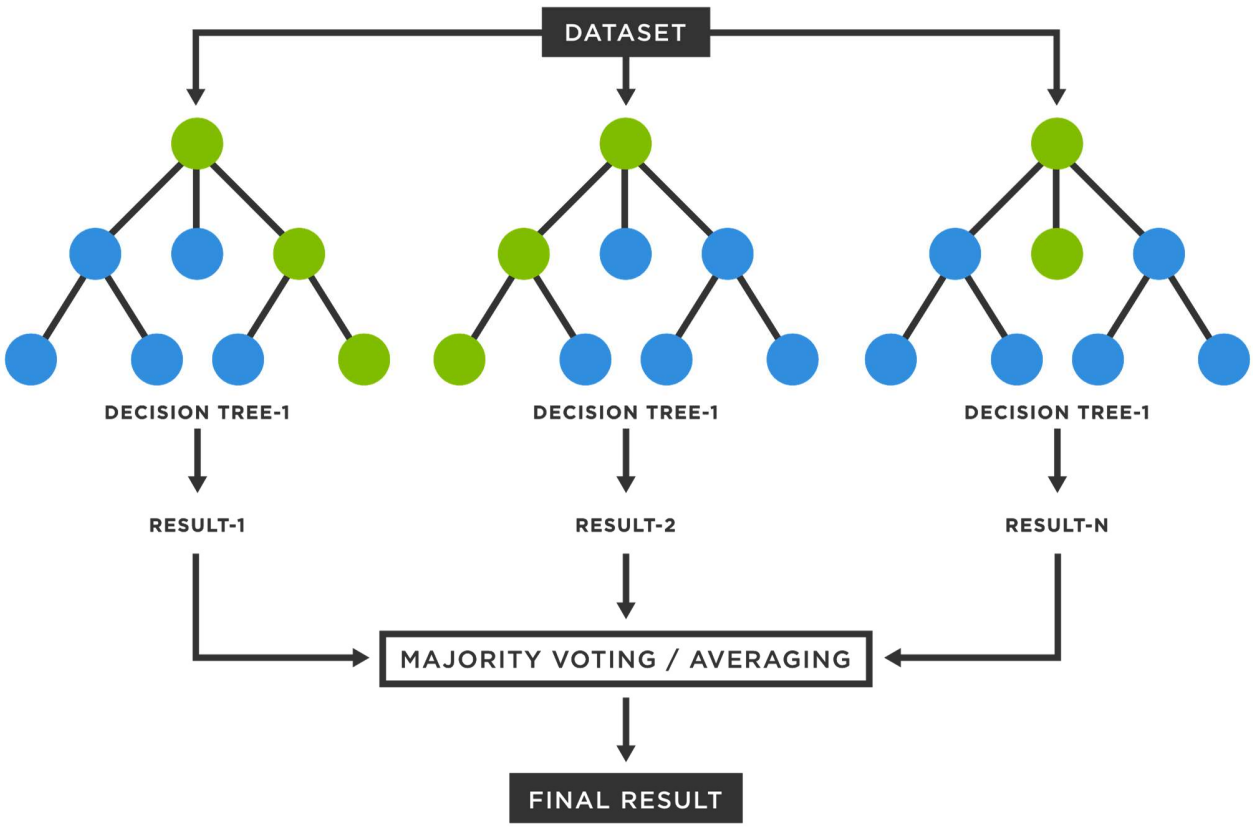
Accuracy: 77% (0.7677777777777778)

Precision: 72%

f1-Score: 58%

- Training and Test set results are almost similar, and with the overall measures high, the model is a good model.
- Agency code is the most important variable for predicting claim status

The random forest is a Machine learning technique that combines several base models in order to produce one optimal predictive model. Consists of a large number of individual decision trees that operate as an ensemble. Each tree in the random forest spits out a class prediction. class with most votes becomes model's prediction. fundamental concept-wisdom of crowds. A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual models.



We perform the following steps to generate the random forest:

- Read csv file
- Feature engineering – convert relevant variables to categorical
- Find Baseline Yclass % to checkclass imbalance
- EDA –Univariate • Boxplot for numvar • Barplot for catvar
- EDA –bivariate • Boxplot – num X vs catY • Stacked bar – cat X vsY
- Split into training and test sets
- Build arandom forest model
- Tune ntree & mtry
- Predict for train& test
- Model performance • Acc, sens, spec • AUC
- Variable importance

Random Forest Conclusion:

Train Data:

AUC: 84.45% (0.8444646309844054)

Accuracy: 80.4% (0.8042857142857143)

Precision: 72%

f1-Score: 63%

Test Data:

AUC: 0.82% (0.8174926460288557)

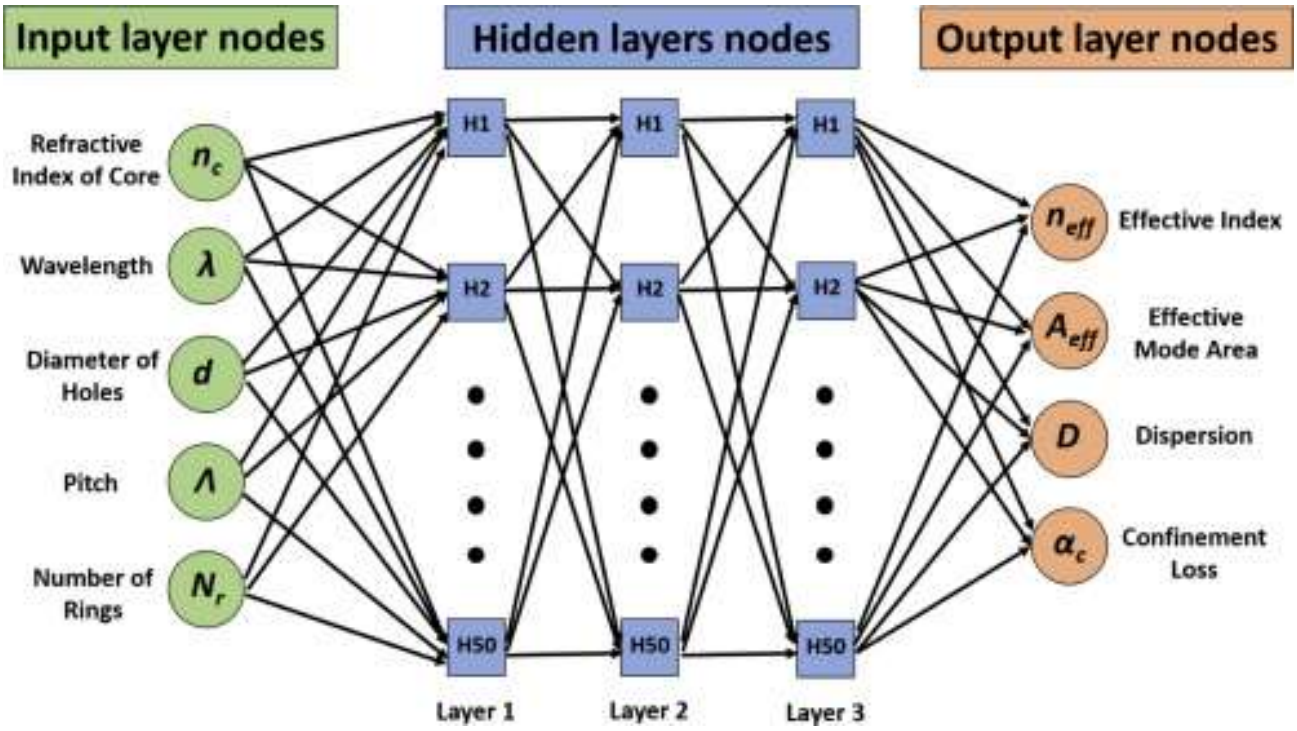
Accuracy: 76.4% (0.7633333333333333)

Precision: 73%

f1-Score: 55%

- Training and Test set results are almost similar, and with the overall measures high, the model is a good model.
- Agency_Code is again the most important variable for claim status

Artificial Neural Network is made of layers with many interconnected nodes (neurons). There are three main layers, specifically – Input Layer – Hidden Layer – Output Layer. Hidden Layer can be one or more. This deep learning technique is a black box model.



Steps involved in artificial neural network:

- Read csv file
- Convert relevant variables to categorical (if required)
- Find Baseline Y class % to check class imbalance
- EDA – Univariate • Boxplot for num var • Barplot for cat var
- EDA – bivariate • Boxplot – num X vs cat Y • Stacked bar – cat X vs Y
- Create dummy Variables (if required)
- Scale X variables
- Combine with Y variable
- Split into training and test sets
- Create formula for Y~X
- Build neural network model
- Predict for train & test – probability
- Set cutoff to get class prediction for train & test
- Model performance • Acc, sens, spec • AUC

Artificial Neural Network Conclusion:

Train Data:

AUC: 81.5% (0.8145708390839754)

Accuracy: 77.6% (0.7761904761904762)

Precision: 72%

f1-Score: 63%

Test Data:

AUC: 78.82% (0.7882252416304805)

Accuracy: 75.78% (0.7577777777777778)

Precision: 73%

f1-Score: 55%

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

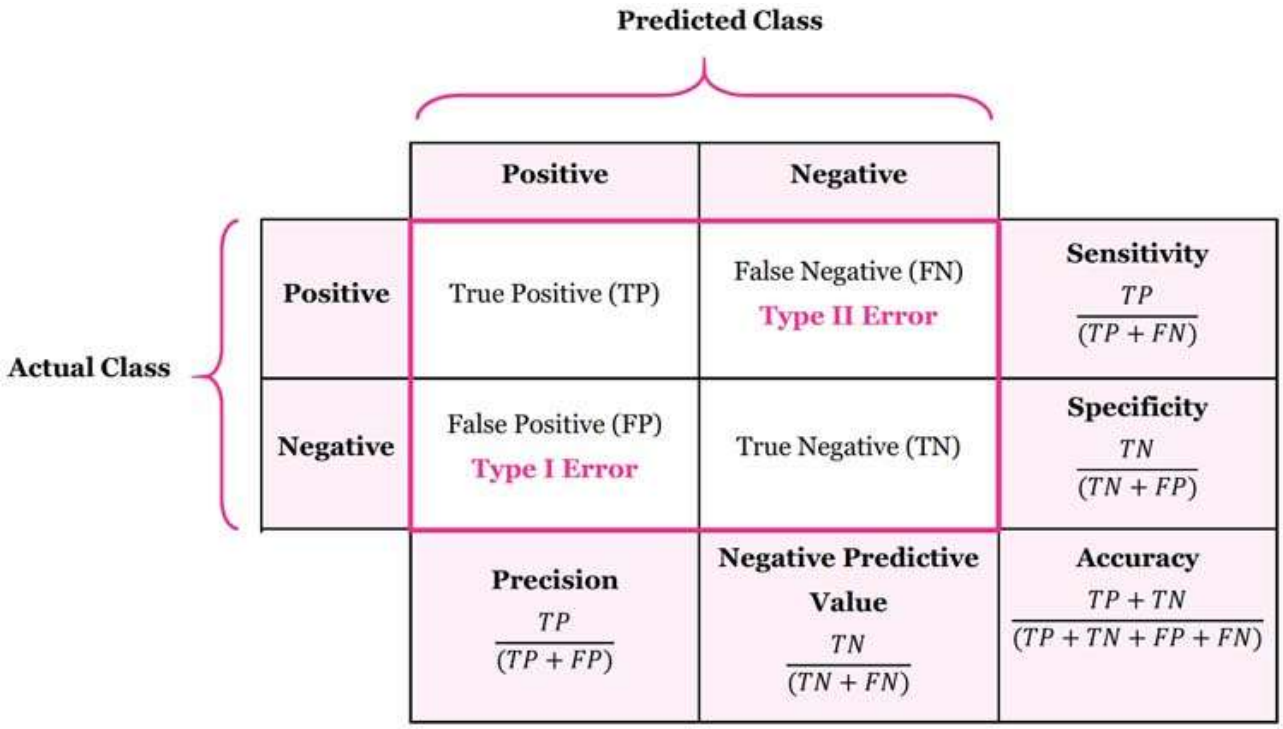
2.3 Performance Metrics: Comment and Check the performance of Predictions on Train and Test sets using Accuracy, Confusion Matrix, Plot ROC curve and get ROC_AUC score, classification reports for each model.

We need to evaluate the model that we have built and validate how good (or bad) it is, so you can then decide on whether to implement it. That's where the AUC-ROC curve comes in.

The Receiver Operator Characteristic (ROC) curve is an evaluation metric for binary classification problems. It is a probability curve that plots the TPR against FPR at various threshold values and essentially separates the 'signal' from the 'noise'. The Area Under the Curve (AUC) is the measure of the ability of a classifier to distinguish between classes and is used as a summary of the ROC curve.

The higher the AUC, the better the performance of the model at distinguishing between the positive and negative classes.

To evaluate our models, we are going to use accuracy, sensitivity, specificity, precision, F1 score.

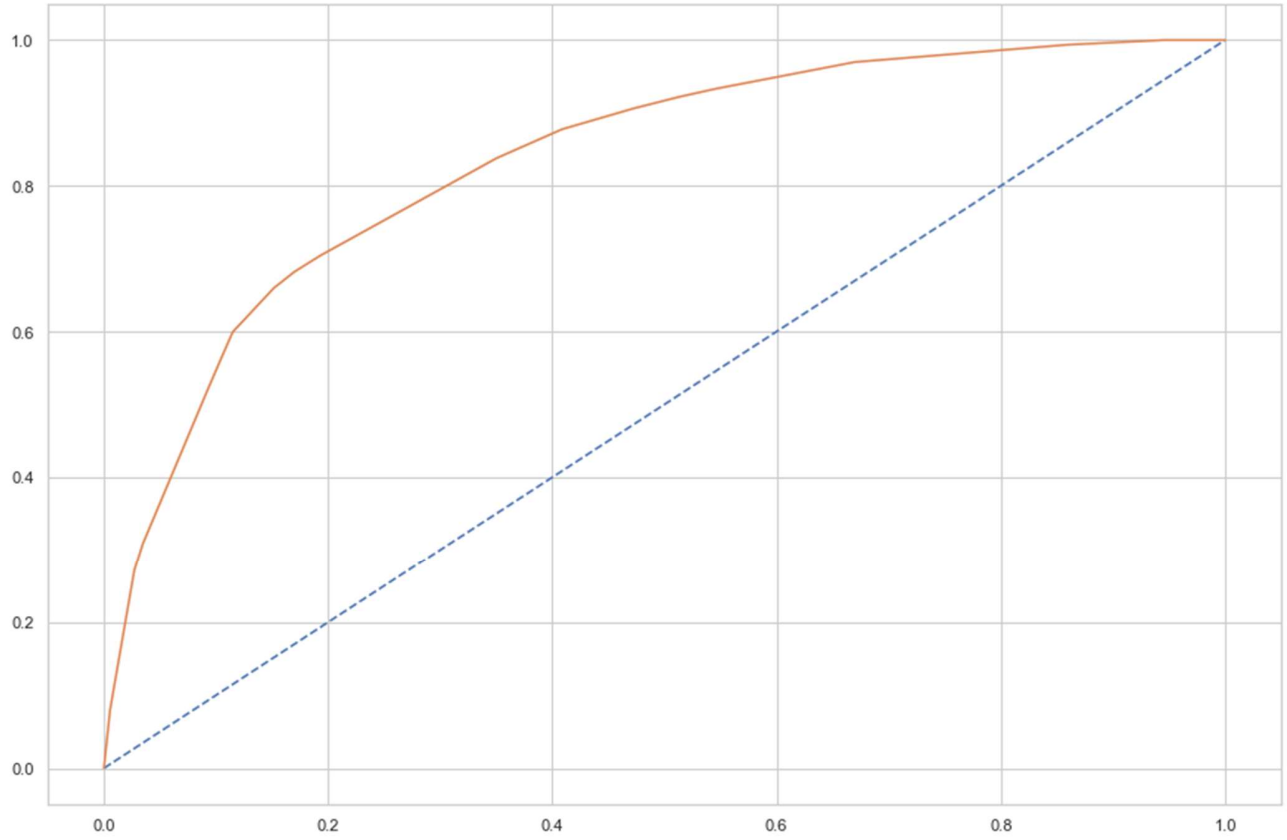


1. **Accuracy**: from the table we can see the formula for accuracy which is crucial because the goal of our models is to build accurate models with minimum errors
2. **Sensitivity/recall/true positive rate:** this calculates out of all true, how many were predicted positively. A higher TPR and a lower FNR is desirable since we want to correctly classify the positive class.
3. **Specificity/true negative rate:** out of all the true, how many predicted negatively.
4. **Precision**: calculates out of all predicted positive, how many are true
5. **F1 score**: The harmonic mean of precision and recall gives a score call f1 score which is a measure of performance of the model's classification ability.

F1 score = 2 * (precision * recall)/ (precision + recall)

F1 score is considered a better indicator of the classifier's performance than the regular accuracy measure.

Let us now look at the train data of the Cart model:

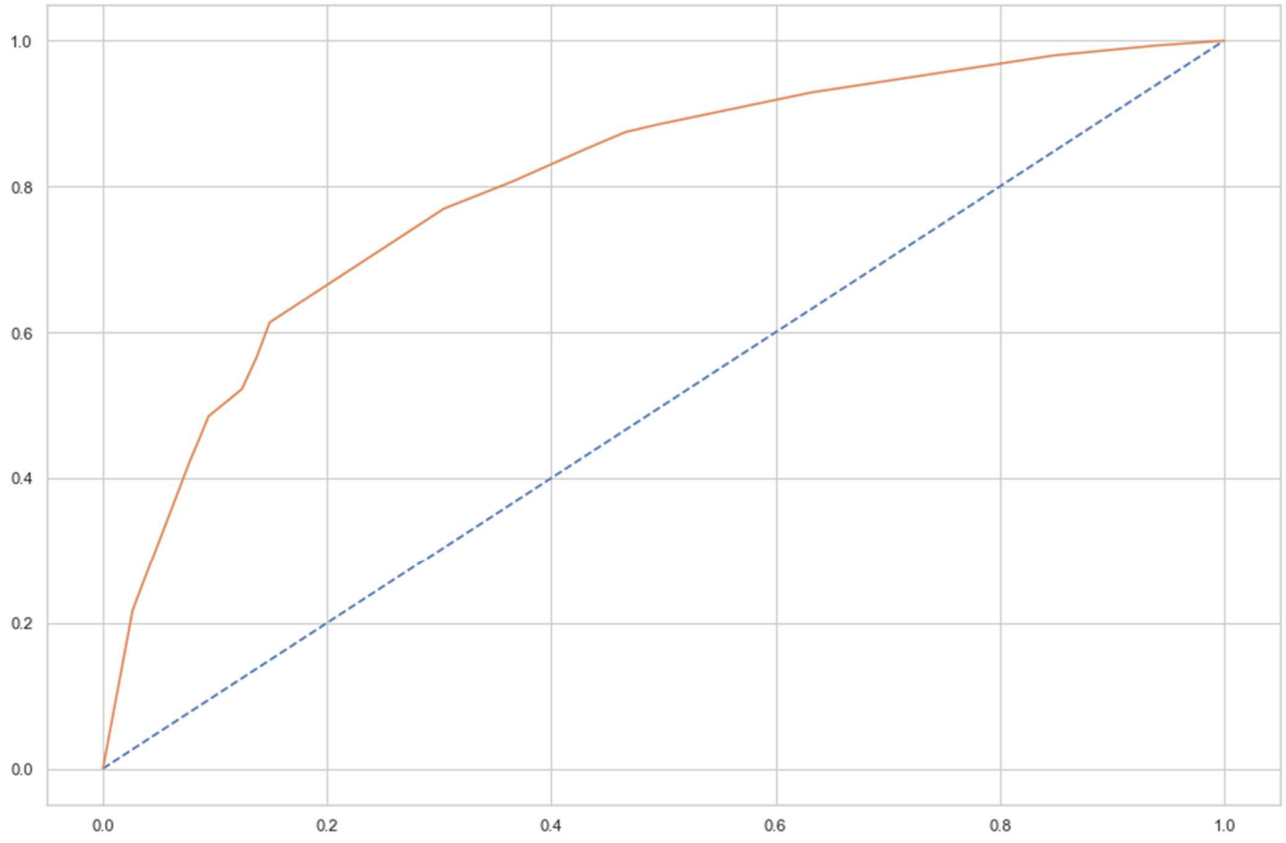



```
confusion_matrix(train_labels, ytrain_predict)
```

```
array([[1302,  169],
       [ 252,  377]], dtype=int64)
```

	precision	recall	f1-score	support
0	0.84	0.89	0.86	1471
1	0.69	0.60	0.64	629
accuracy			0.80	2100
macro avg	0.76	0.74	0.75	2100
weighted avg	0.79	0.80	0.80	2100

Cart test




```
confusion_matrix(test_labels, ytest_predict)
```

```
array([[548,  57],
       [152, 143]], dtype=int64)
```

	precision	recall	f1-score	support
0	0.78	0.91	0.84	605
1	0.71	0.48	0.58	295
accuracy			0.77	900
macro avg	0.75	0.70	0.71	900
weighted avg	0.76	0.77	0.75	900

When the difference between train and test data is within 10%, we call that a robust model.

Train Data:

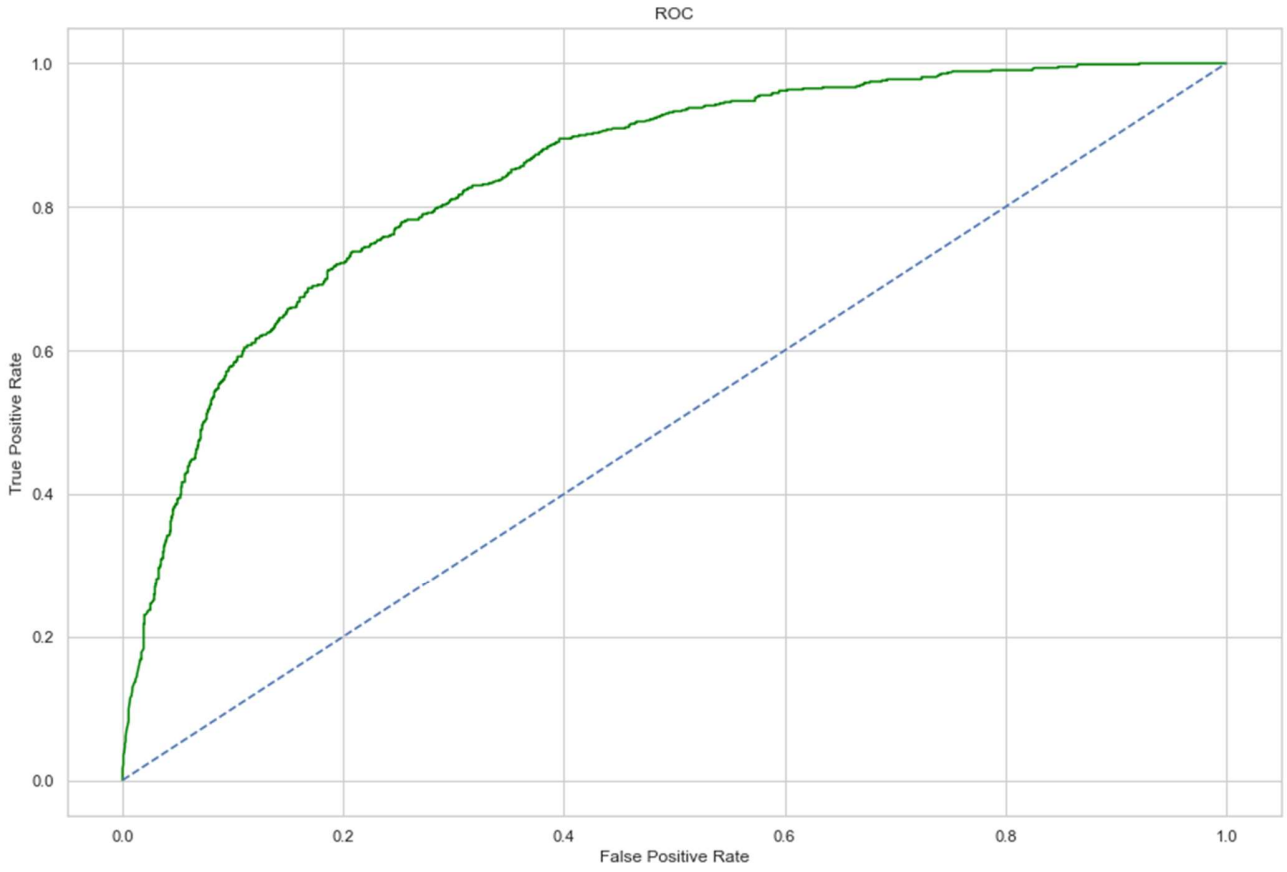
AUC: 83.4% , Accuracy: 80% , Precision: 69% ,f1-Score: 64%

Test Data:

AUC: 80.3% ,Accuracy: 77% ,Precision: 72% , f1-Score: 58%

Training and Test set results are almost similar, and with the overall measures high, the model is a good model. Agency code is the most important variable for predicting claim status. This is a robust model, since train and test accuracy difference is within 10%

Random forest train

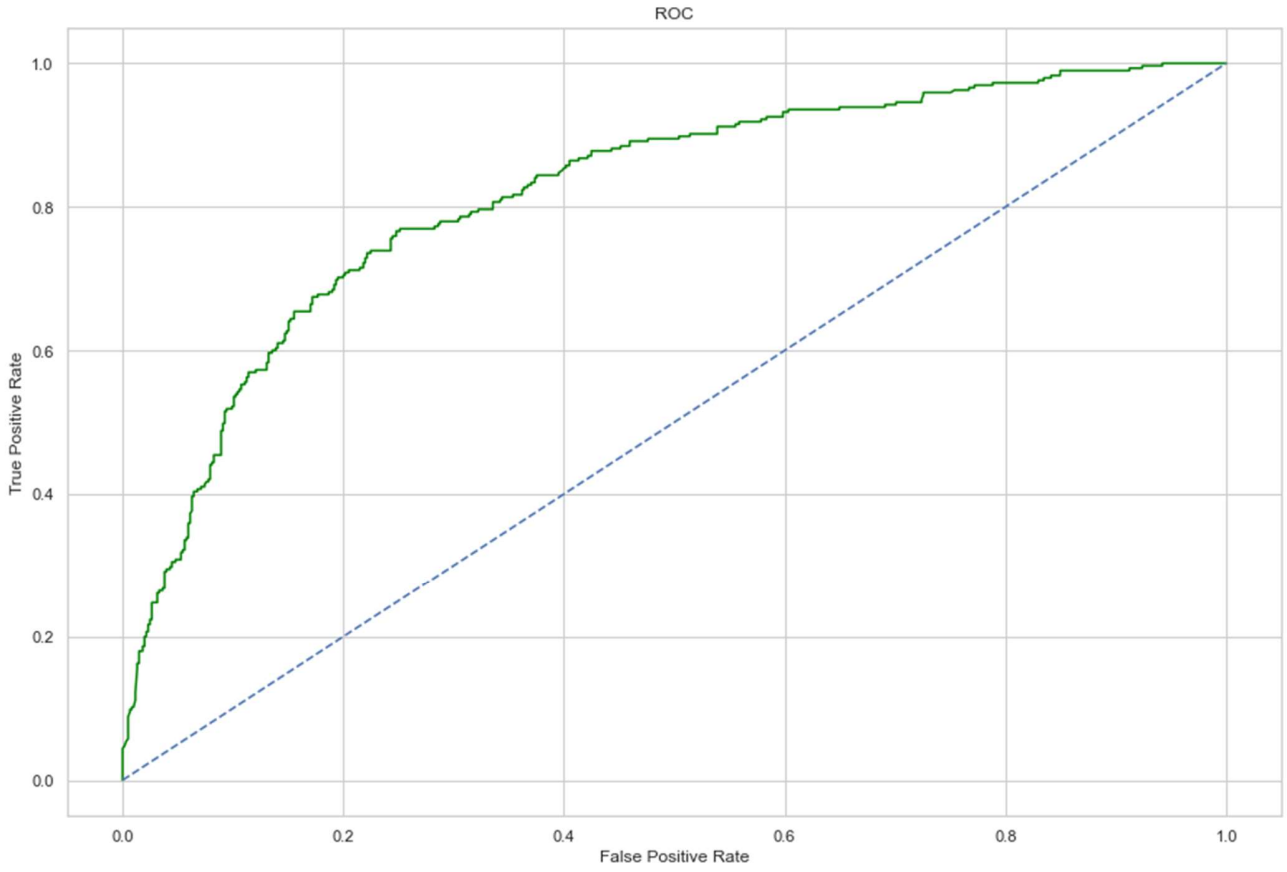



```
confusion_matrix(train_labels,ytrain_predict)
```

```
array([[1334,  137],
       [ 274,  355]], dtype=int64)
```

	precision	recall	f1-score	support
0	0.83	0.91	0.87	1471
1	0.72	0.56	0.63	629
accuracy			0.80	2100
macro avg	0.78	0.74	0.75	2100
weighted avg	0.80	0.80	0.80	2100

Random forest test




```
confusion_matrix(test_labels,ytest_predict)
```

```
array([[555,  50],
       [163, 132]], dtype=int64)
```

	precision	recall	f1-score	support
0	0.77	0.92	0.84	605
1	0.73	0.45	0.55	295
accuracy			0.76	900
macro avg	0.75	0.68	0.70	900
weighted avg	0.76	0.76	0.75	900

Train Data:

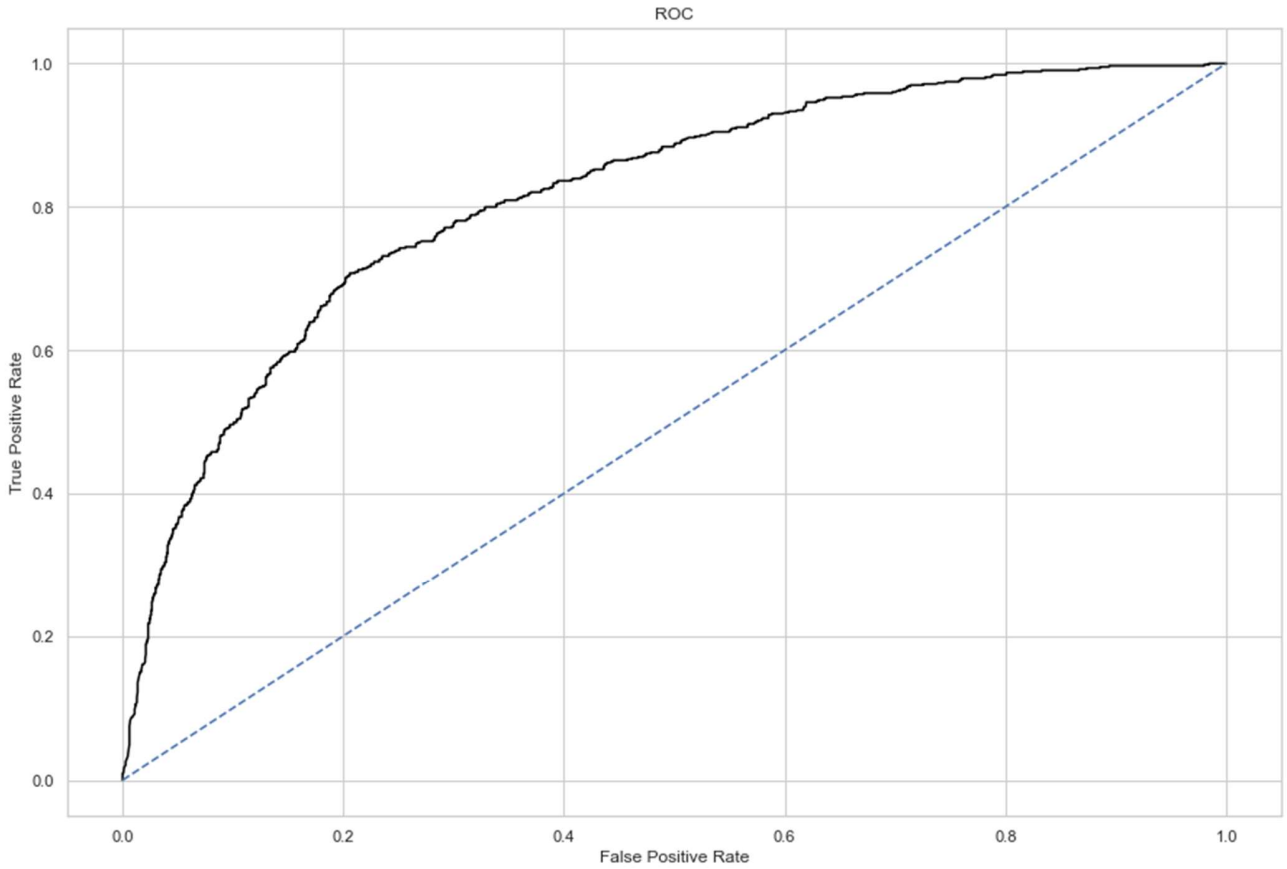
AUC: 84.45% , Accuracy: 80.4% ,Precision: 72% ,f1-Score: 63%

Test Data:

AUC: 0.82% ,Accuracy: 76.4% ,Precision: 73% , f1-Score: 55%

Training and Test set results are almost similar, and with the overall measures high, the model is a good model. Agency Code is again the most important variable for claim status. This is a robust model, since train and test accuracy difference is within 10%

Artificial Neural network train

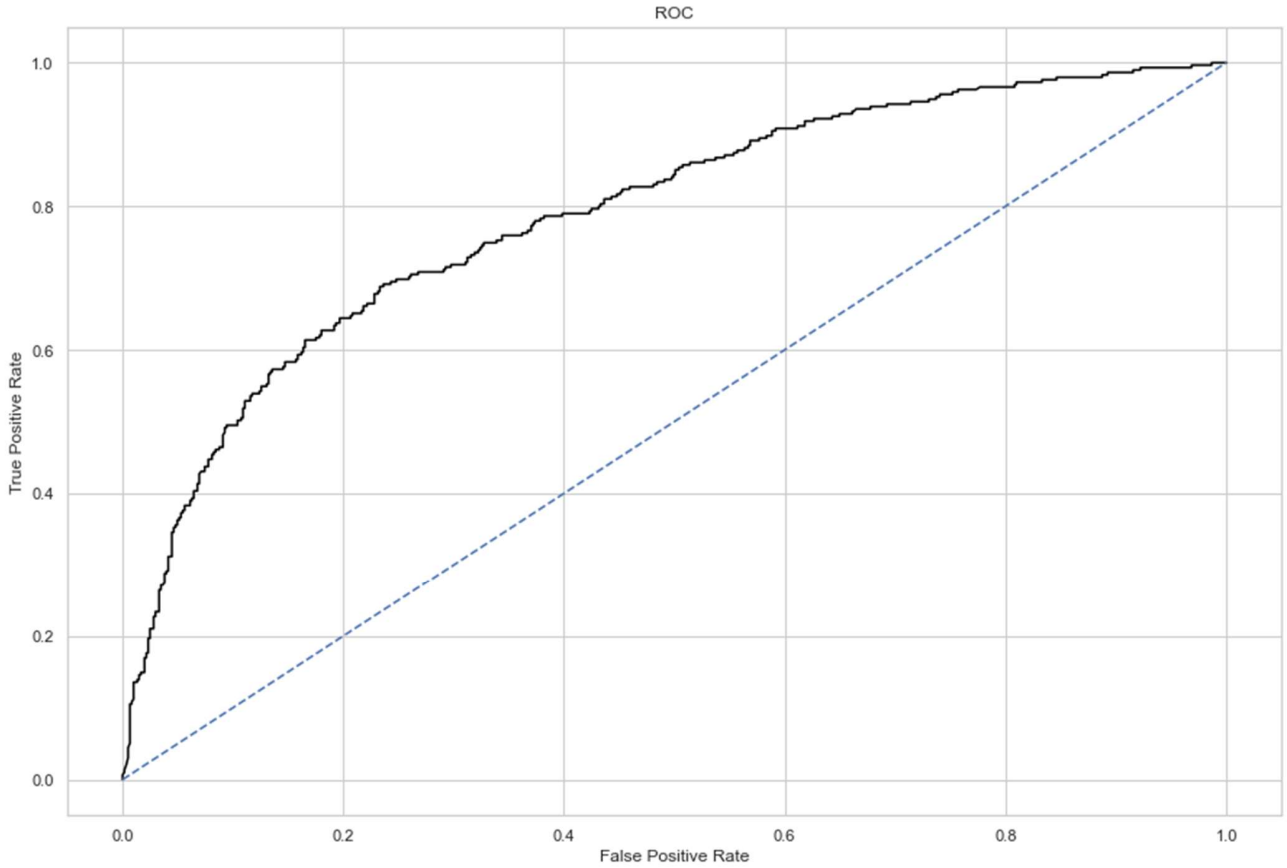



```
confusion_matrix(train_labels,ytrain_predict)
```

```
array([[1363,  108],
       [ 362,  267]], dtype=int64)
```

	precision	recall	f1-score	support
0	0.79	0.93	0.85	1471
1	0.71	0.42	0.53	629
accuracy			0.78	2100
macro avg	0.75	0.68	0.69	2100
weighted avg	0.77	0.78	0.76	2100

Neural network test




```
confusion_matrix(test_labels,ytest_predict)
```

```
array([[572,  33],
       [185, 110]], dtype=int64)
```

	precision	recall	f1-score	support
0	0.76	0.95	0.84	605
1	0.77	0.37	0.50	295
accuracy			0.76	900
macro avg	0.76	0.66	0.67	900
weighted avg	0.76	0.76	0.73	900

Train Data:

AUC: 81.5% , Accuracy: 77.6% ,Precision: 72% ,f1-Score: 63%

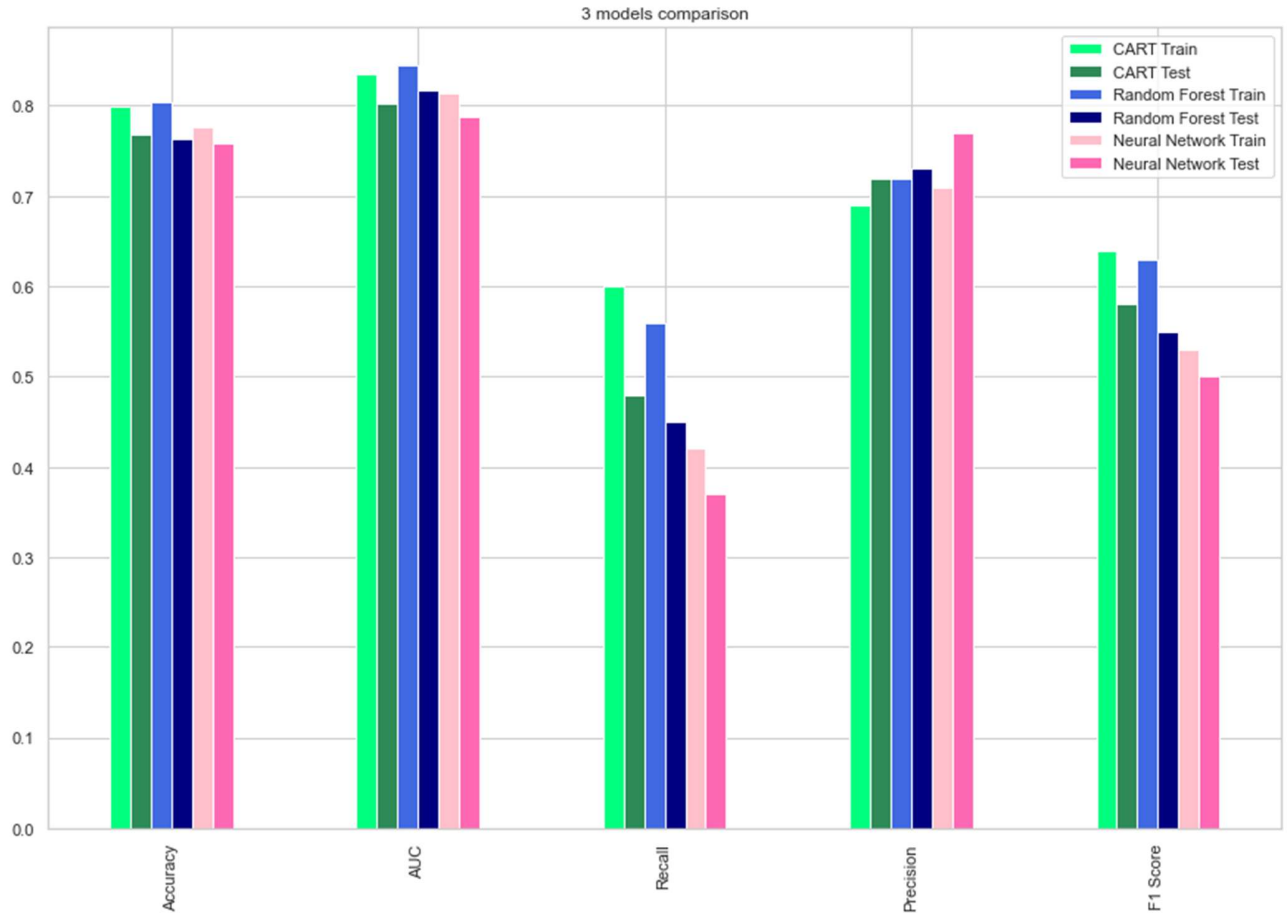
Test Data:

AUC: 78.82% ,Accuracy: 75.78% ,Precision: 73% ,f1-Score: 55%

Training and Test set results are almost similar, and with the overall measures high, the model is a good model. This is a robust model, since train and test accuracy difference is within 10%

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

	CART Train	CART Test	Random Forest Train	Random Forest Test	Neural Network Train	Neural Network Test
Accuracy	0.80	0.77	0.80	0.76	0.78	0.76
AUC	0.83	0.80	0.84	0.82	0.81	0.79
Recall	0.60	0.48	0.56	0.45	0.42	0.37
Precision	0.69	0.72	0.72	0.73	0.71	0.77
F1 Score	0.64	0.58	0.63	0.55	0.53	0.50



Speaking of accuracy according to industry standards,

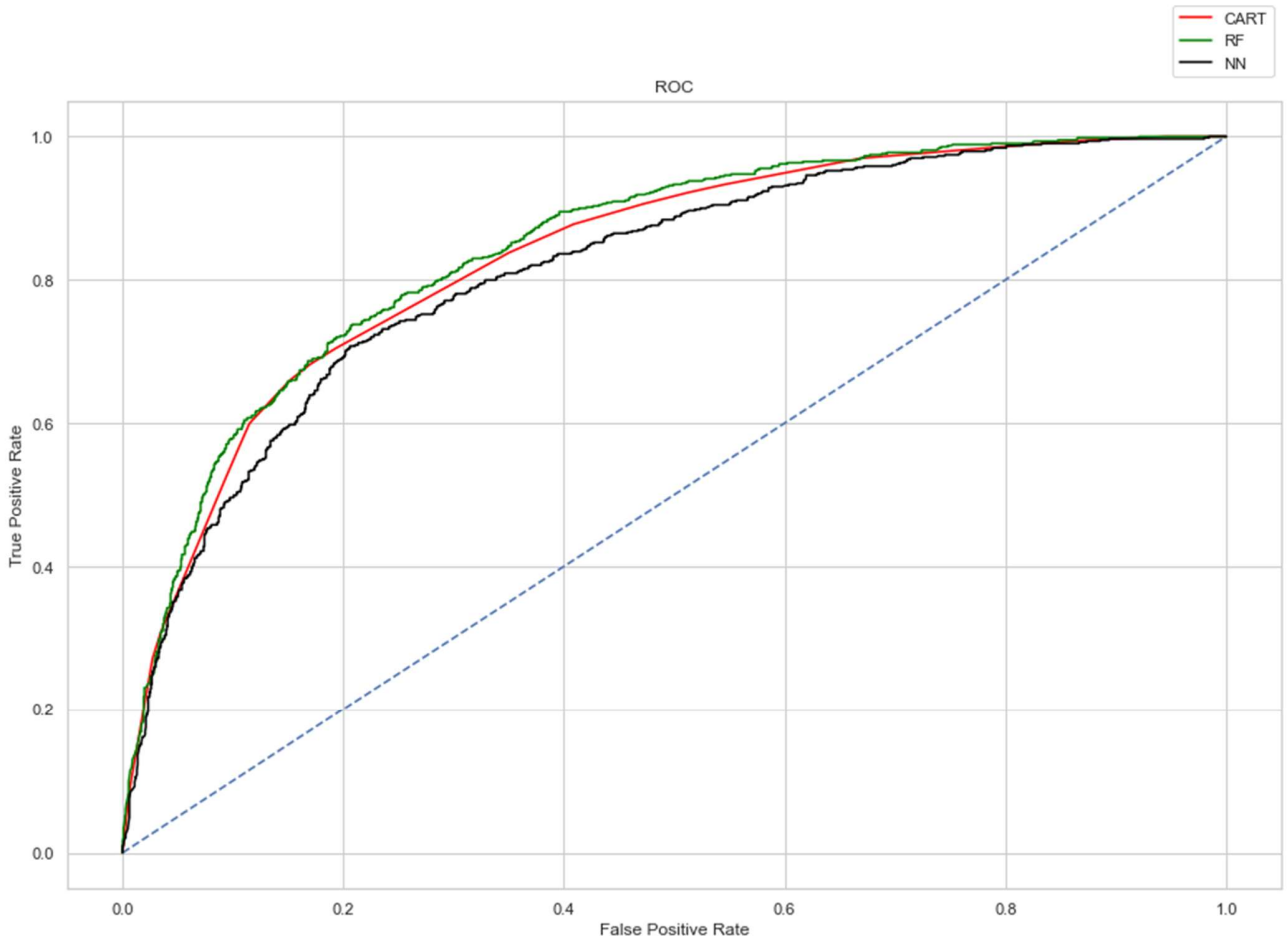
If your 'X' value is between 60% and 70%, it's a poor model.

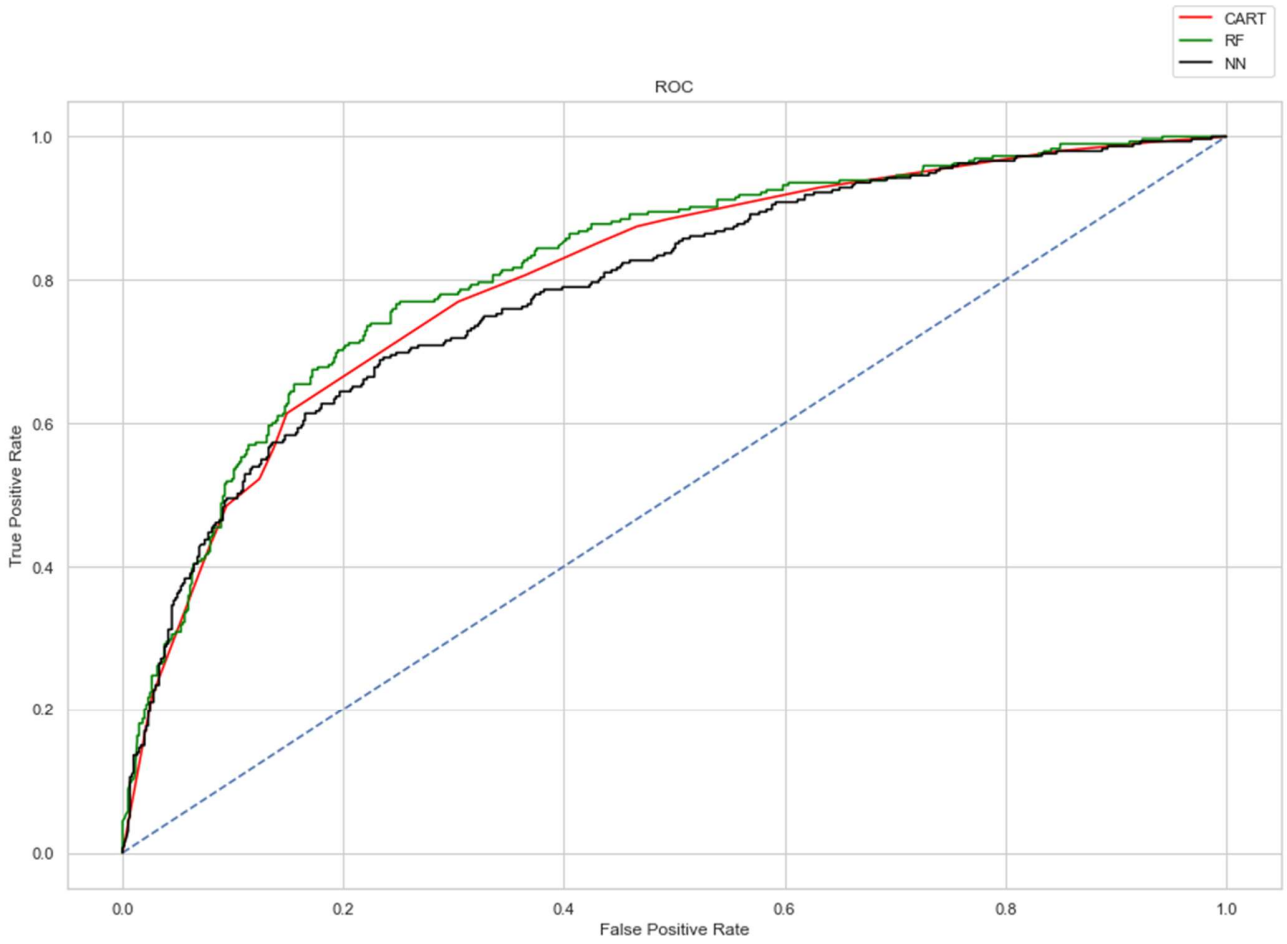
If your 'X' value is between 70% and 80%, you've got a good model.

If your 'X' value is between 80% and 90%, you have an excellent model.

If your 'X' value is between 90% and 100%, it's a probably an overfitting case.

The accuracy for all 3 models for test and train ranges between 70% and 80%, so we can say we have got good model.


CART
RF
NN
ROC
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0



AREA UNDER THE ROC CURVE

AUC is an effective way to summarize the overall accuracy of the test. It takes values from 0 to 1, where a value of 0 indicates a perfectly inaccurate test and a value of 1 reflects a perfectly accurate test. AUC can be computed using the trapezoidal rule.3 In general, an AUC of 0.7 to 0.8 is considered acceptable, 0.8 to 0.9 is considered excellent, and more than 0.9 is considered outstanding.

Except for artificial neural network the models have generated AUC more than 0.8, we can say the models have acceptable to excellent AUC

Recall (Sensitivity) - Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes. The question recall answers is: Of all the travellers that truly claimed, how many did we label correctly? Except for cart train and random forest train We have got recall of less than 0.5 which is bad for this model as it's below 0.5. Recall is a useful metric in cases where False Negative trumps False Positive.

Precision tells us how many of the correctly predicted cases actually turned out to be positive. Recall tells us how many of the actual positive cases we were able to predict correctly with our model. 70 to

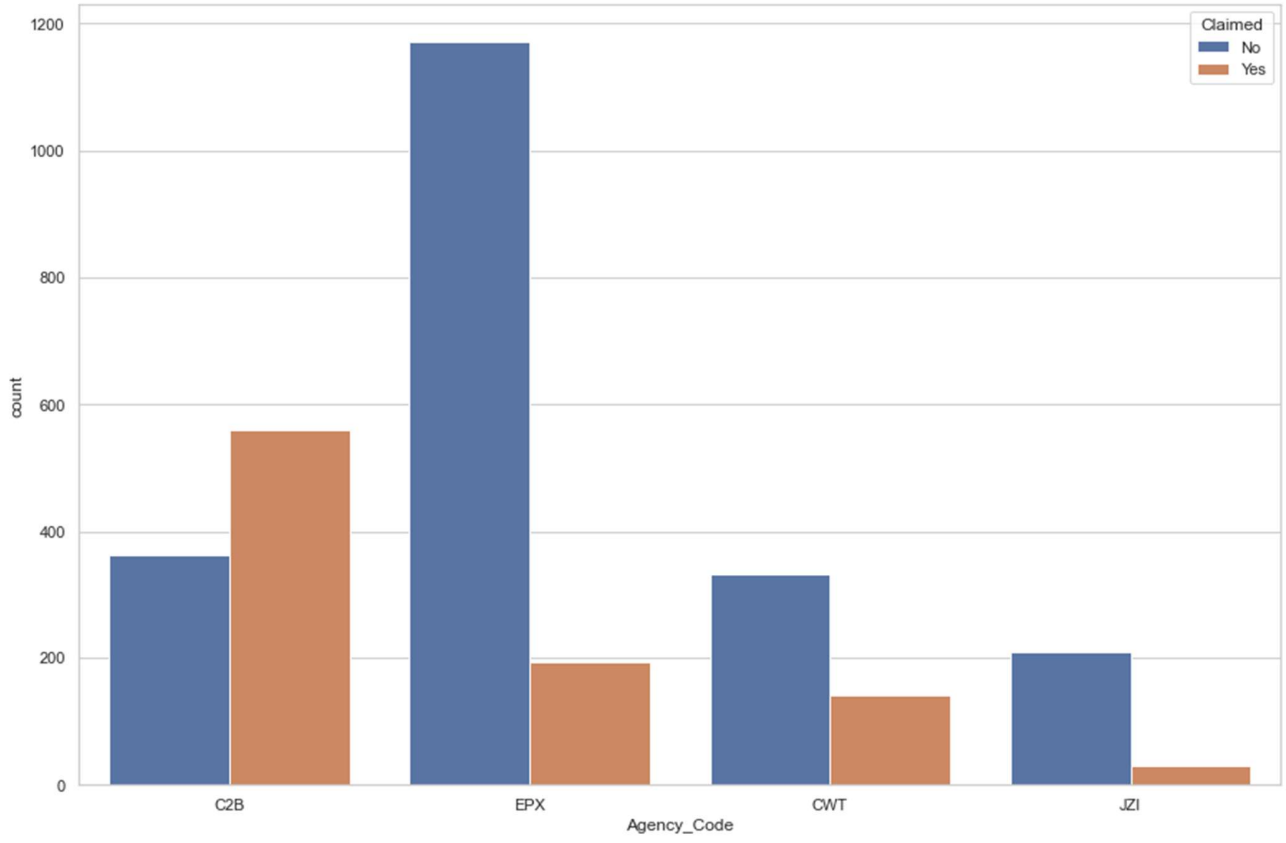
80% percent of the correctly predicted cases turned out to be positive cases. Whereas 40 to 50% of the positives were successfully predicted by our model. Anything below 0.5 is not acceptable. Precision is a useful metric in cases where False Positive is a higher concern than False Negatives.

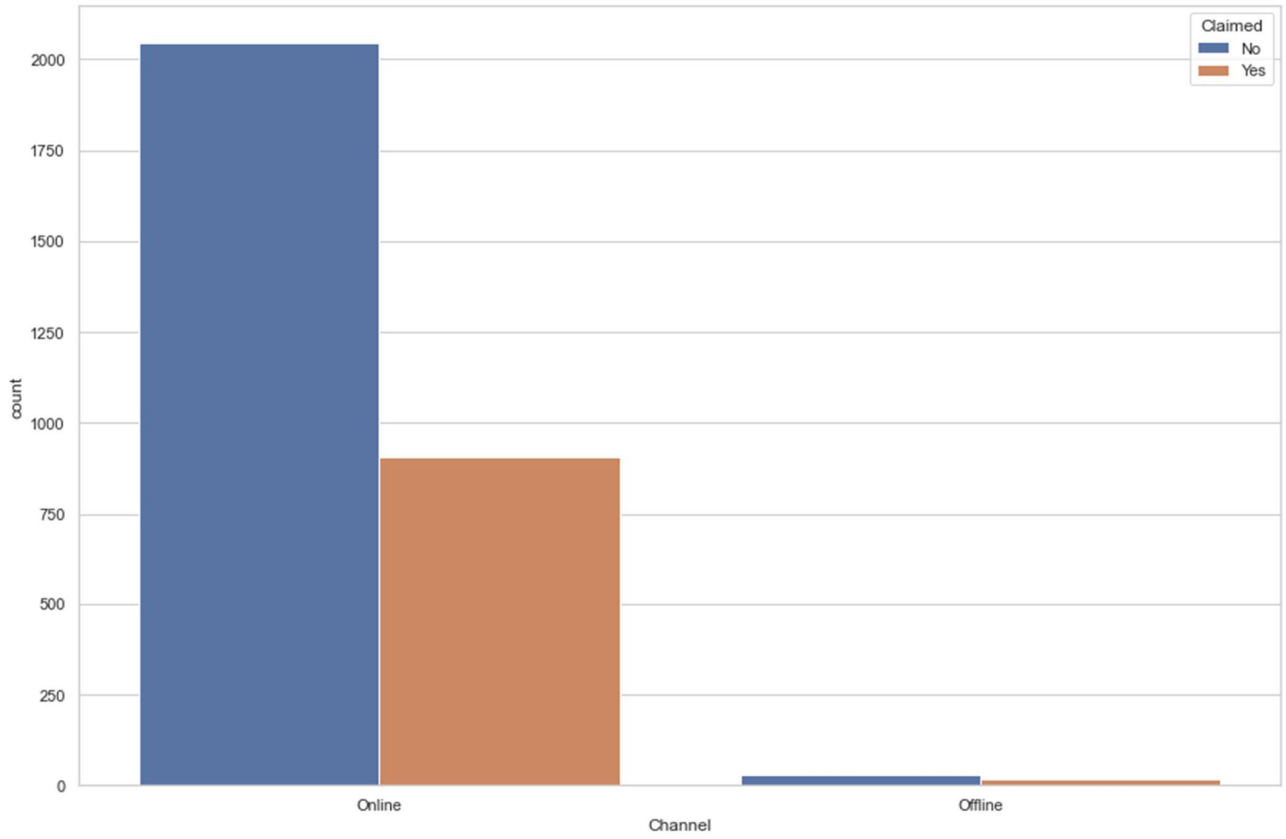
F1 score is an overall measure of a model's accuracy that combines precision and recall. a good F1 score means that you have low false positives and low false negatives, so you're correctly identifying real threats and you are not disturbed by false alarms. An F1 score is considered perfect when it's 1, while the model is a total failure when it's 0. We can say, 0.5 to to 0.6 is acceptable but not that great.

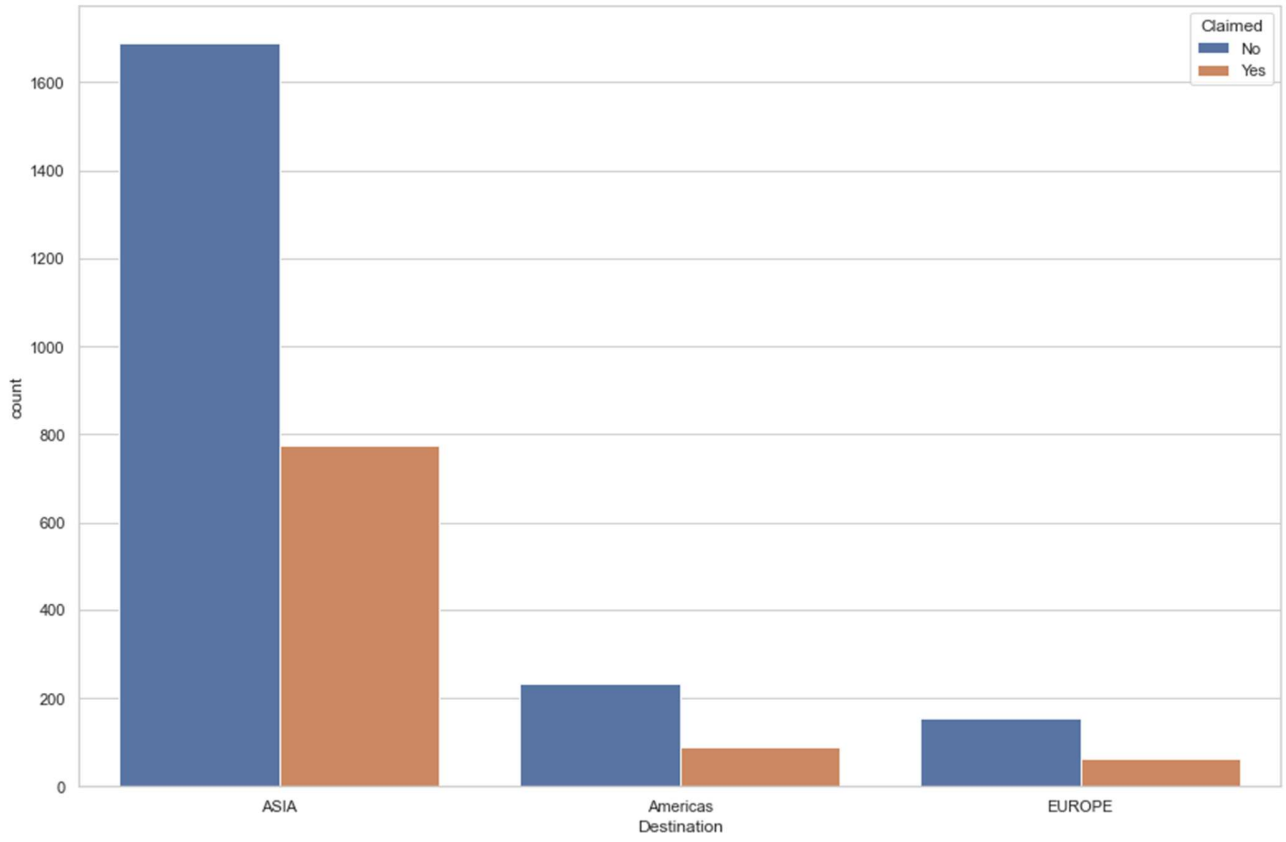
Based on the **recall**, **precision**,**f1 score** and **accuracy** value of **random forest**, in my opinion this model seems to have yielded better results in comparison to CART and Artificial neural network

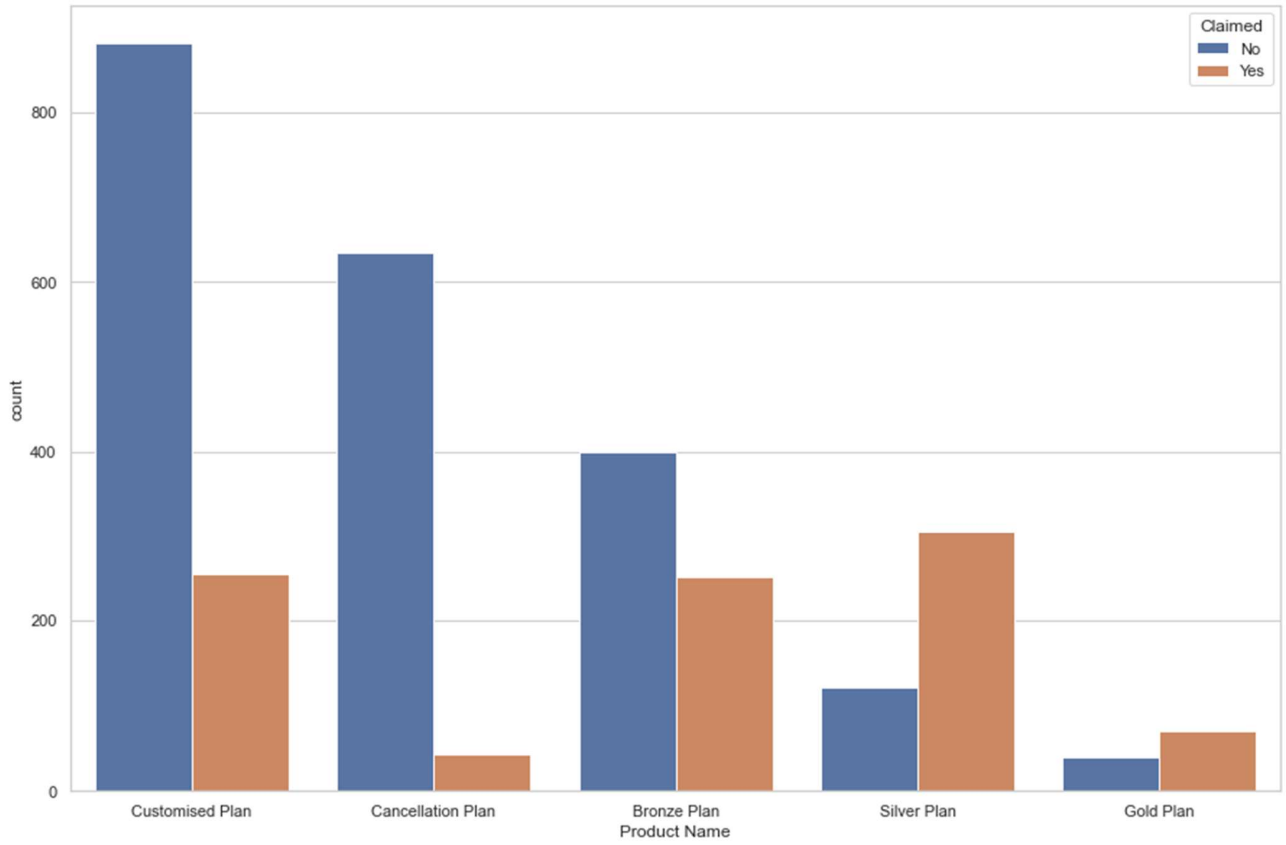
2.5 Inference: Based on the whole Analysis, what are the business insights and recommendations

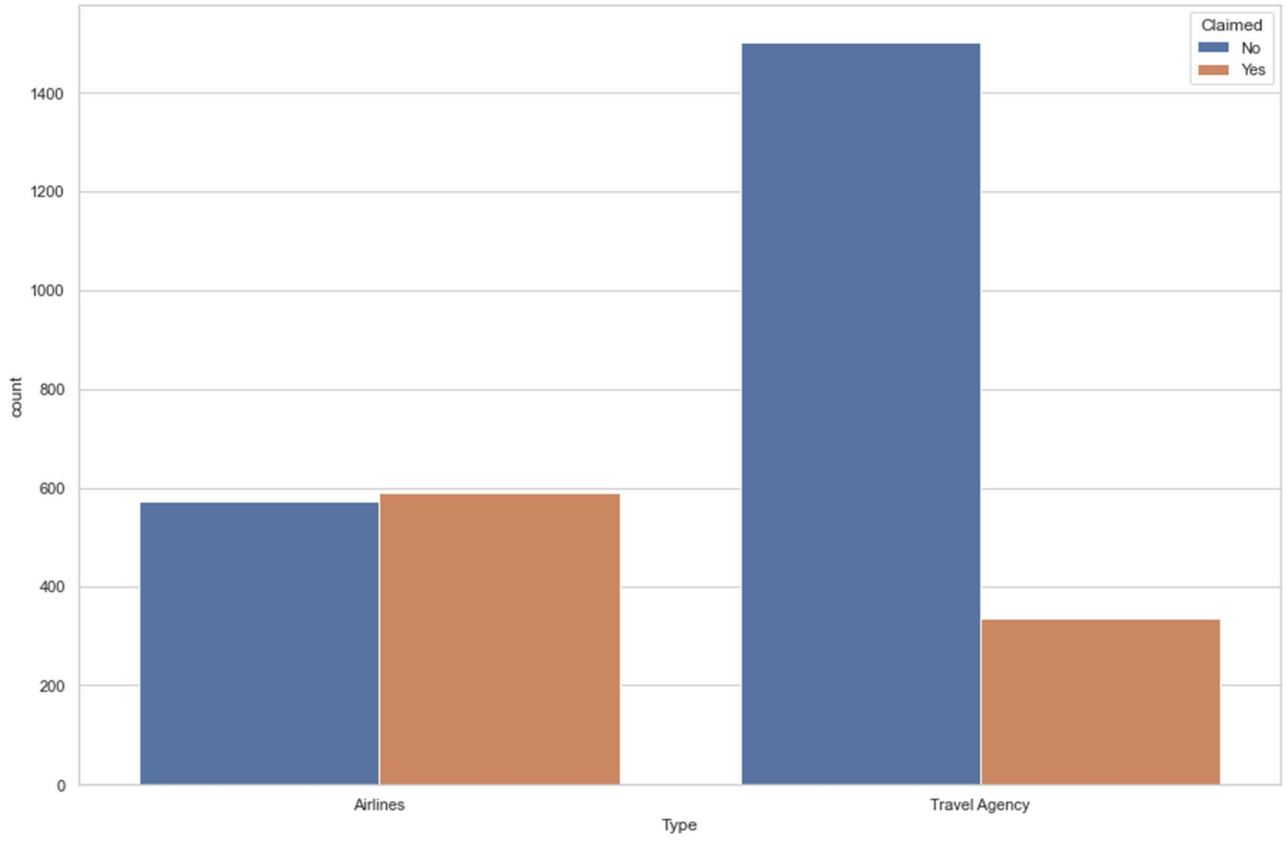
When we summarize the categorical variables and the claimed status we get the below bar plot:




Claimed
No
Yes
2000
1750
1500
1250
1000
750
500
250
0
count
Online
Offline
Channel

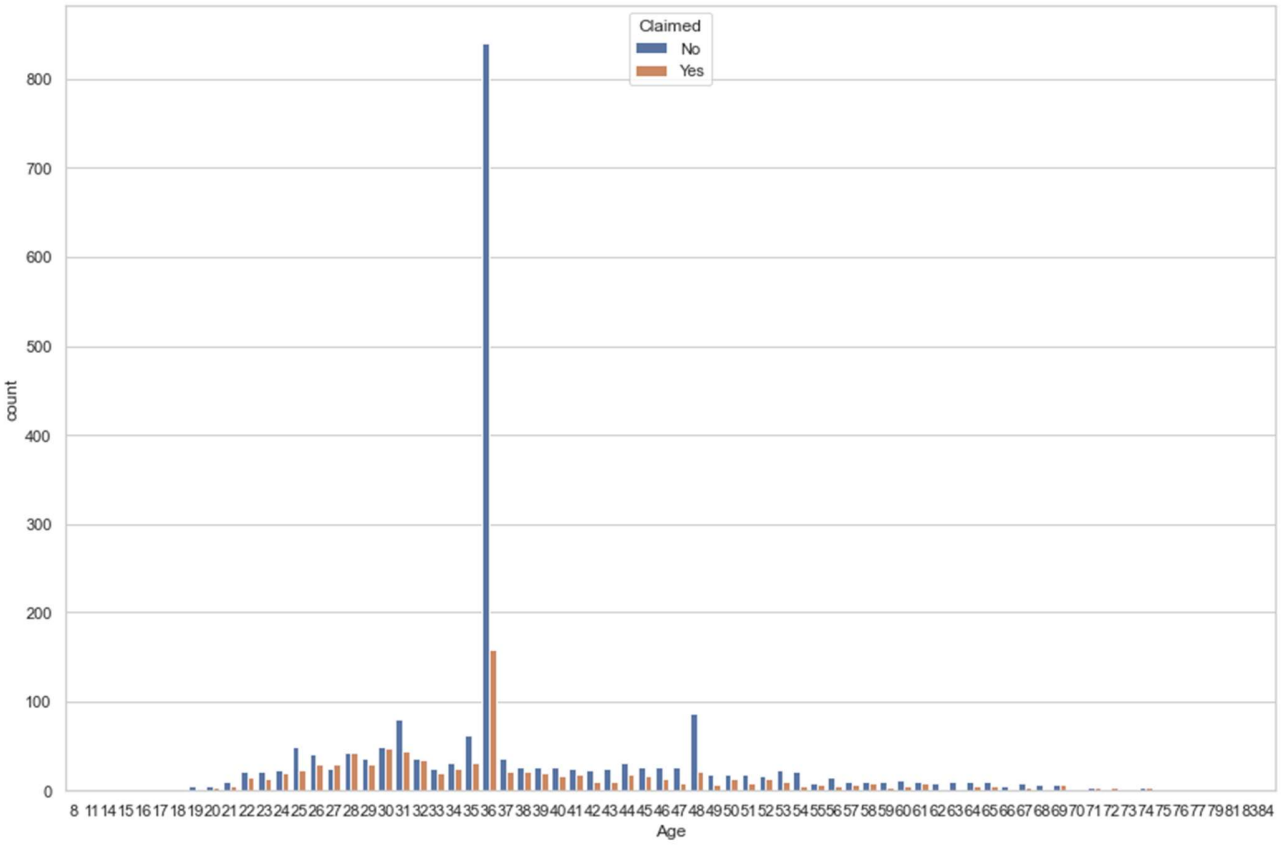

Claimed
No
Yes
1600
1400
1200
1000
800
600
400
200
0
count
ASIA
Americas
EUROPE
Destination


Claimed
No
Yes
800
600
400
200
0
count
Customised Plan
Cancellation Plan
Bronze Plan
Silver Plan
Gold Plan
Product Name


Claimed
No
Yes
1400
1200
1000
800
600
400
200
0
count
Airlines
Travel Agency
Type

- When we deep dive into the data and check claim status across each variable, **Type** of tour insurance firms **Airlines** seems to claim more.
- Among Name of the tour insurance **products**, **gold and silver plan** are claiming more
- Among **Agency Code** of tour firm, **C2B** are claiming more

We need to investigate claims from these categories more, the business may want to tie up with other agencies to avoid fraudulent claims. The products need to reviewed to understand if the claims are genuine. We need more information about airline Type of tour insurance firms to understand how exactly the claims are increasing.

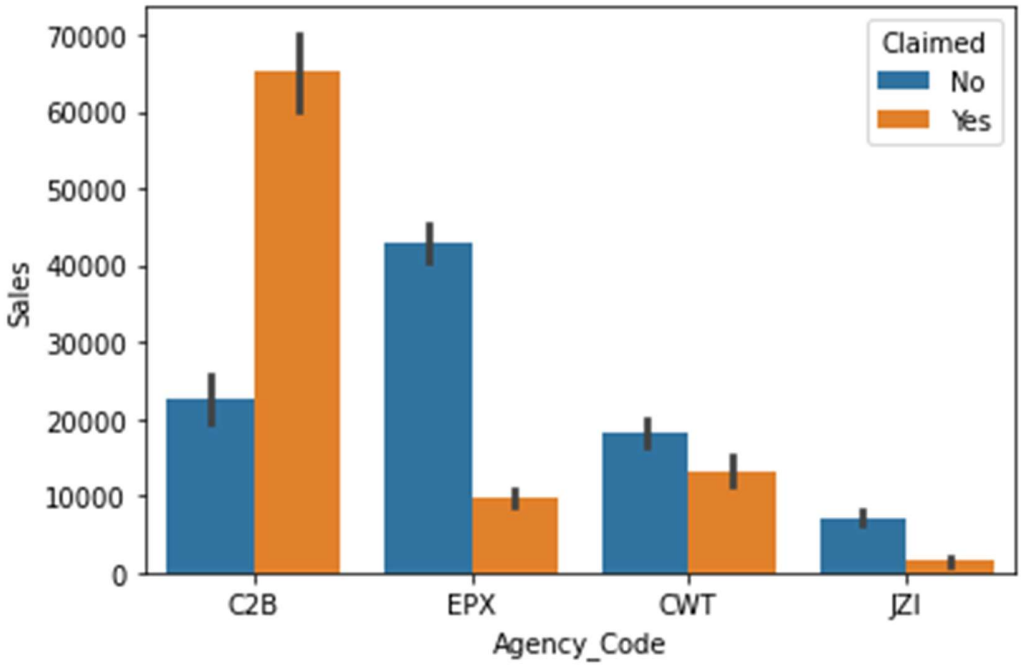


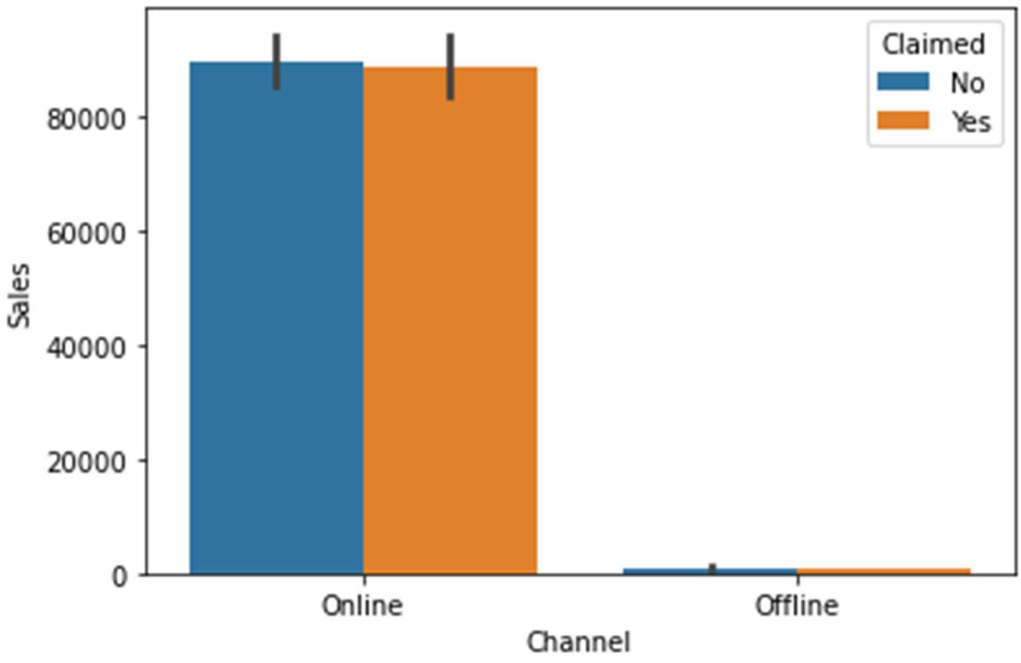
We can also notice that claims from 25-50 age is more in comparison with other age groups, which could be working professionals. Also, 90% claims are processed online and almost all offline customers have claimed, the business needs to check these customer profiles.

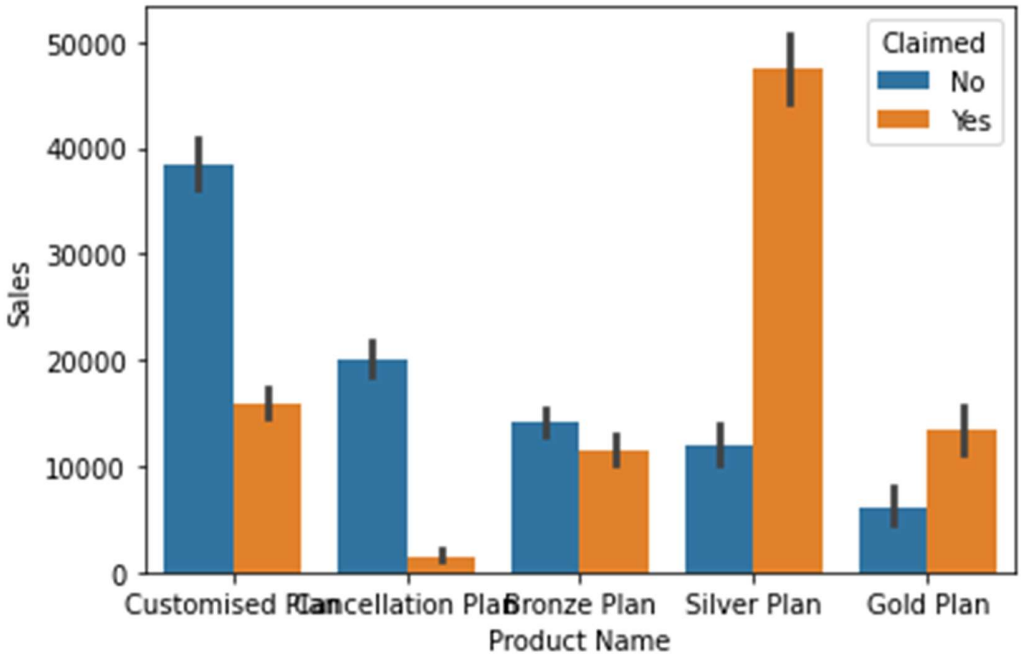
According to CART, top 5 contributing variables are Agency_Code, Sales, Product_Name, Age , Commision

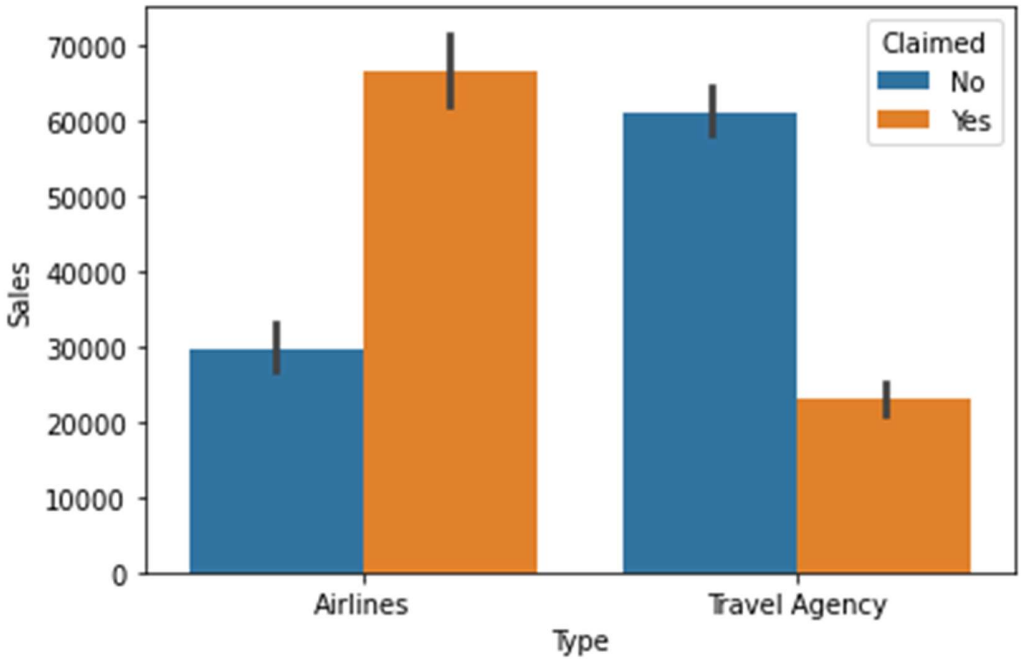
According to Random Forest, major contributors include Agency_Code, Product_Name, Sales, Commision ,Type

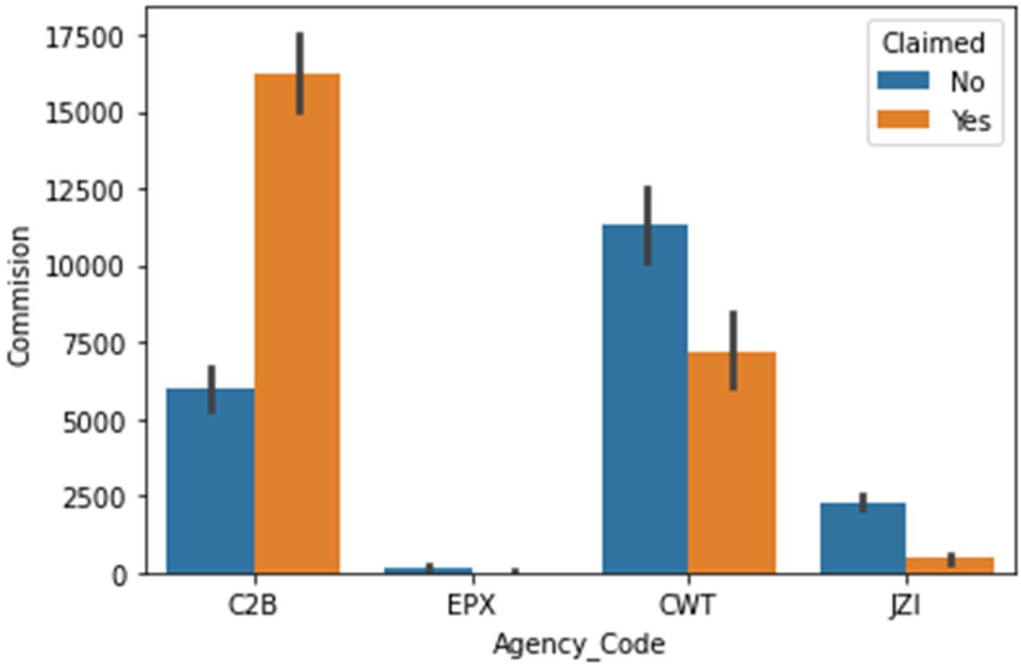
When we try to plot the **commission** and **sales** against **agncy code** ,**product name** with **claim** status we get the below bar plot:

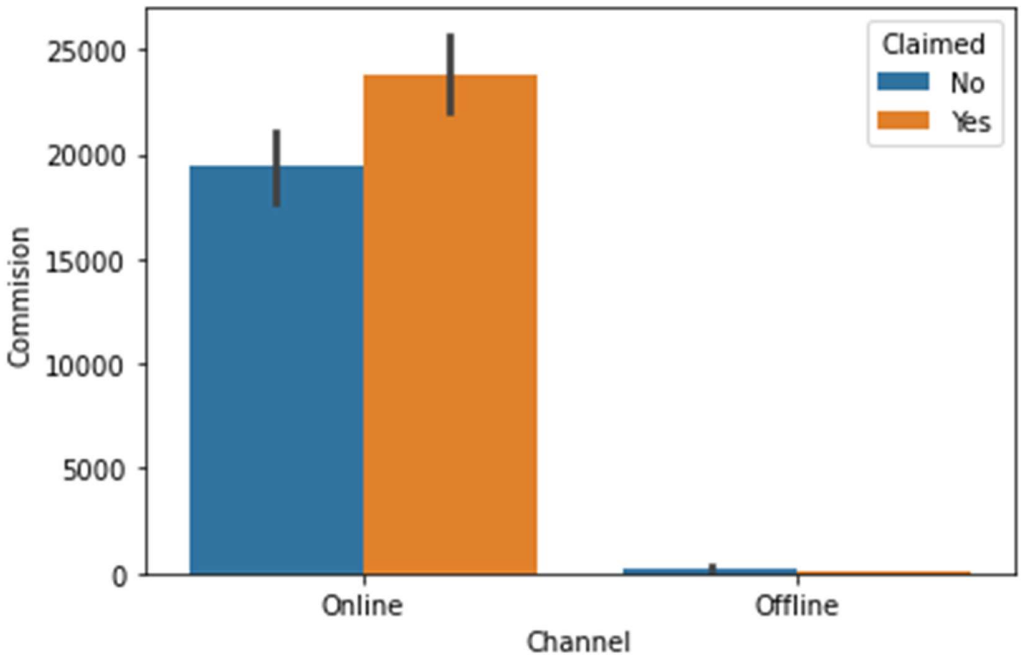

Claimed
No
Yes
70000
60000
50000
40000
30000
20000
10000
0
Sales
C2B
EPX
CWT
JZI
Agency_Code

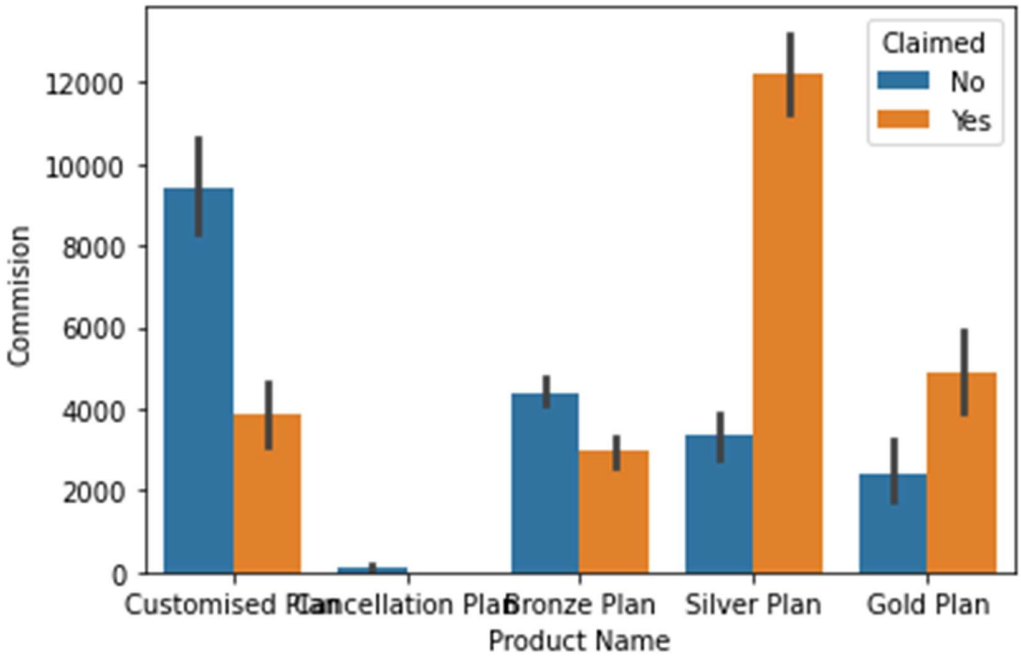

Claimed
No
Yes
80000
60000
40000
20000
0
Sales
Online
Offline
Channel

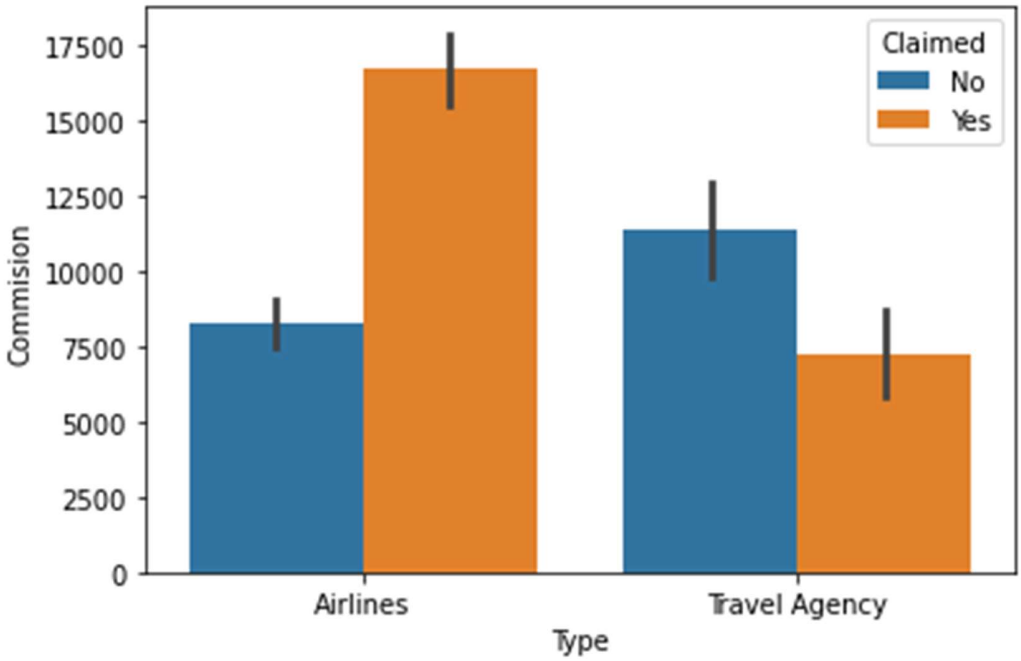

Claimed
No
Yes
Sales
50000
40000
30000
20000
10000
0
Customised Plan
Cancellation Plan
Bronze Plan
Silver Plan
Gold Plan
Product Name


Claimed
No
Yes
Sales
70000
60000
50000
40000
30000
20000
10000
0
Airlines
Travel Agency
Type


Claimed
No
Yes
Commision
17500
15000
12500
10000
7500
5000
2500
0
C2B
EPX
CWT
JZI
Agency_Code


Claimed
No
Yes
Commision
25000
20000
15000
10000
5000
0
Online
Offline
Channel





We can conclude that commission and sales are contributing to product name, agency code, type which is affecting claim status. The business needs to come up with better commission plan and sales structure to reduce the claim status.

Bibliography:

- https://scikit-learn.org/stable/modules/preprocessing.html