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 After importing the file to the jupyter notebook. We need to do the exploratory analysis to understand the basic information from the dataset. We have selected first five rows to know about the variables and datatypes.

	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

Then we have checked the shape of the dataset and understood that the dataset has 210 rows and 7 columns. Using the info function we have understood that there is no null values and all datatypes are in floats which is helpful for our prediction.

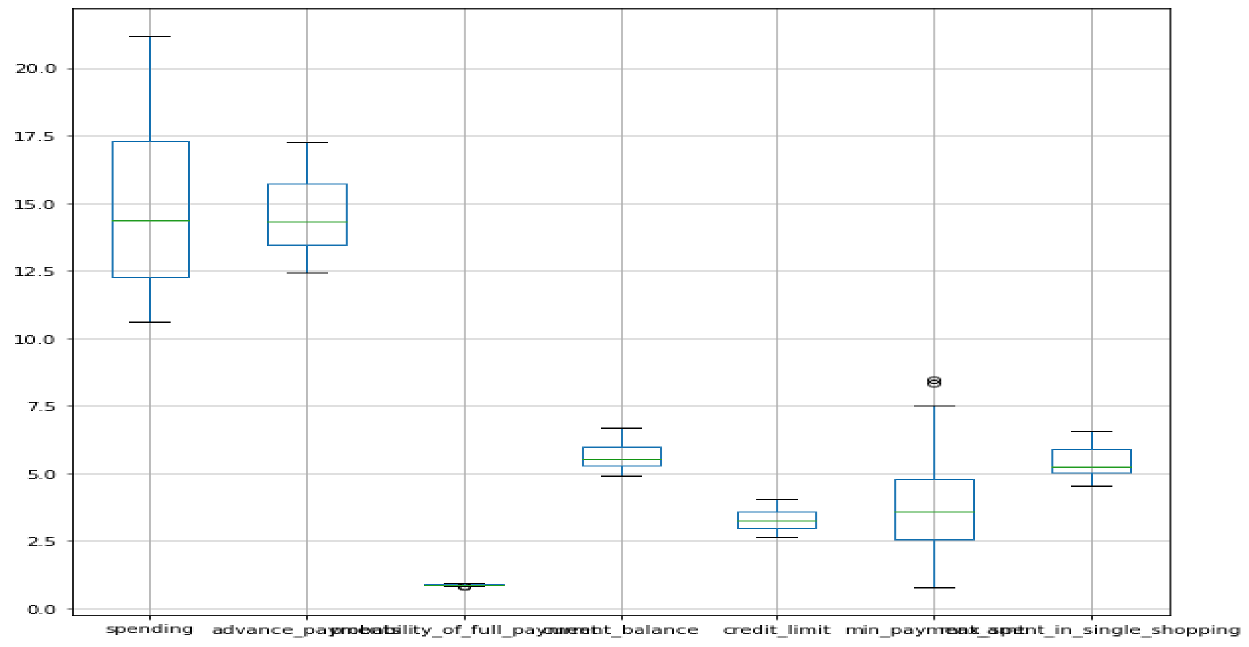
```
Data columns (total 7 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   spending                      210 non-null    float64
 1   advance_payments              210 non-null    float64
 2   probability_of_full_payment   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
```

Now we have to apply some descriptive analysis to understand the statistical summary of the dataset. We have used the describe function to the dataset, since all the variables are numericals , Lets evaluate their mean, mode ,median, max and min values .

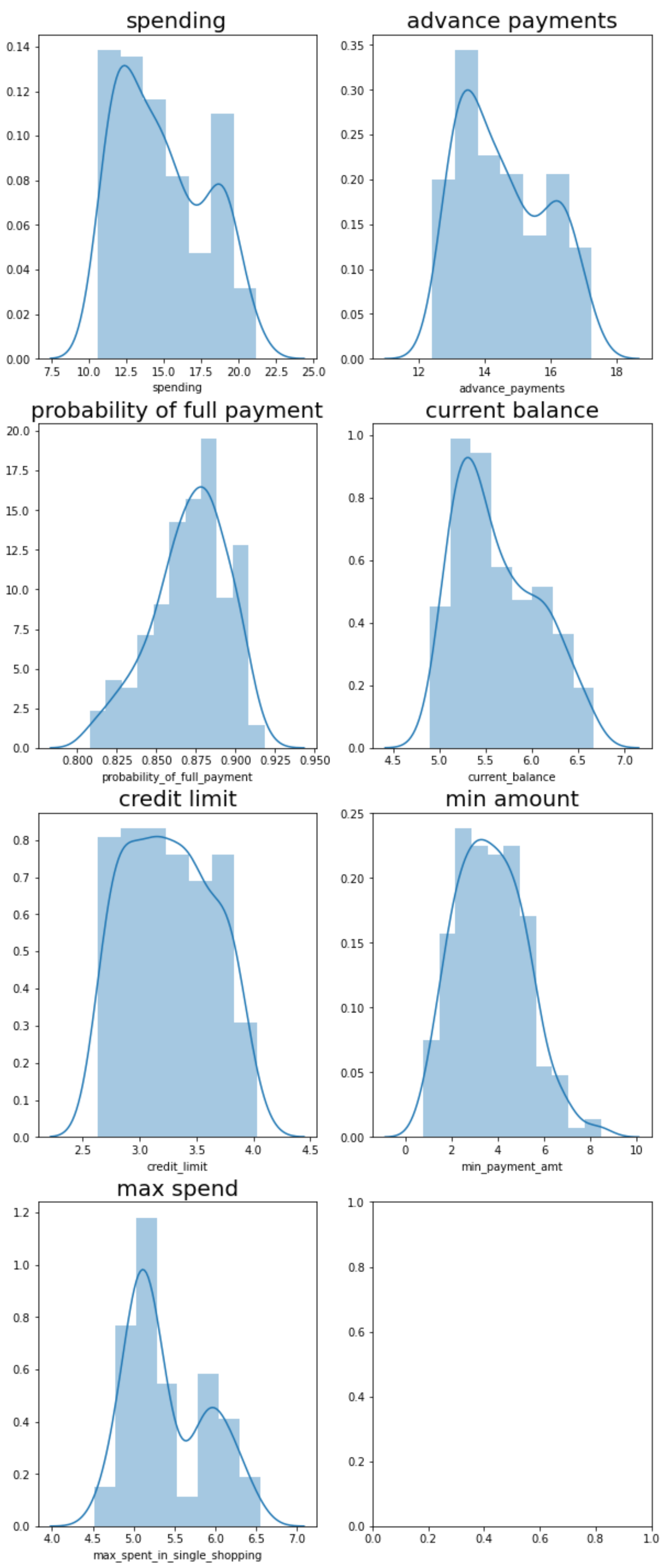
	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
count	210.000000	210.000000	210.000000	210.000000	210.000000	210.000000	210.000000

	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
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

Most of the variables mean and medians are almost equal interms . Except for probability of full payment variables every other variables are +ve skewed. Let us use the boxplot to see the outliers and understand the visual statistical summary.

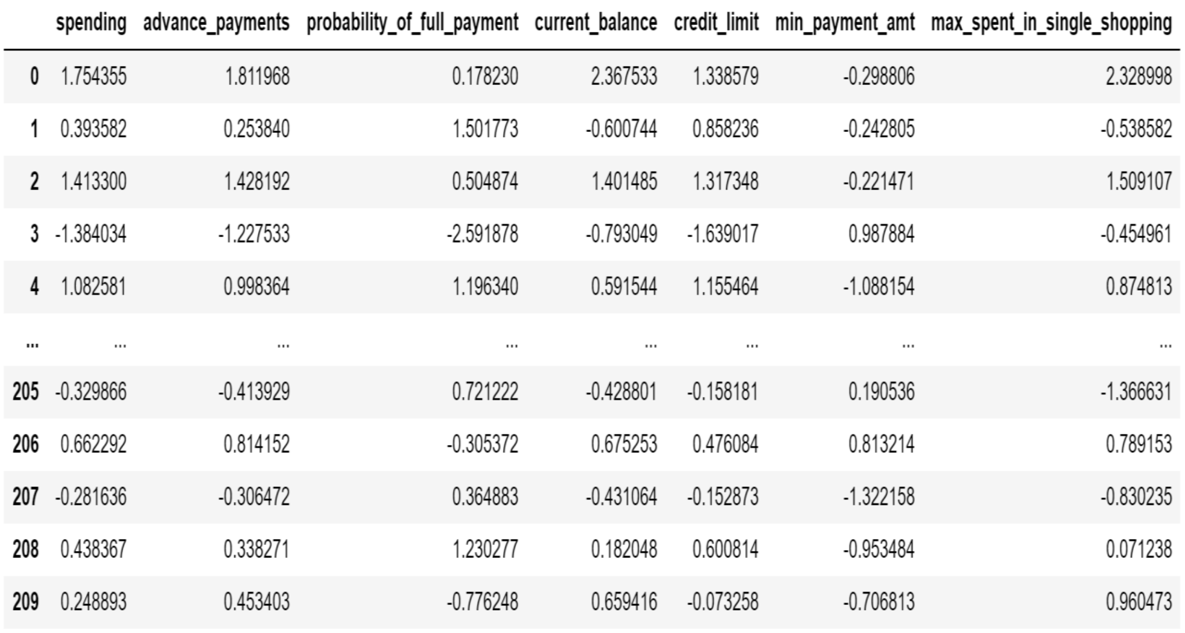


Most of the variables doesn't have any outliers and except min payment and prob of full payment variables. Most of the variables mean and medians are almost equal to get the uniformity in the distribution of dataset.



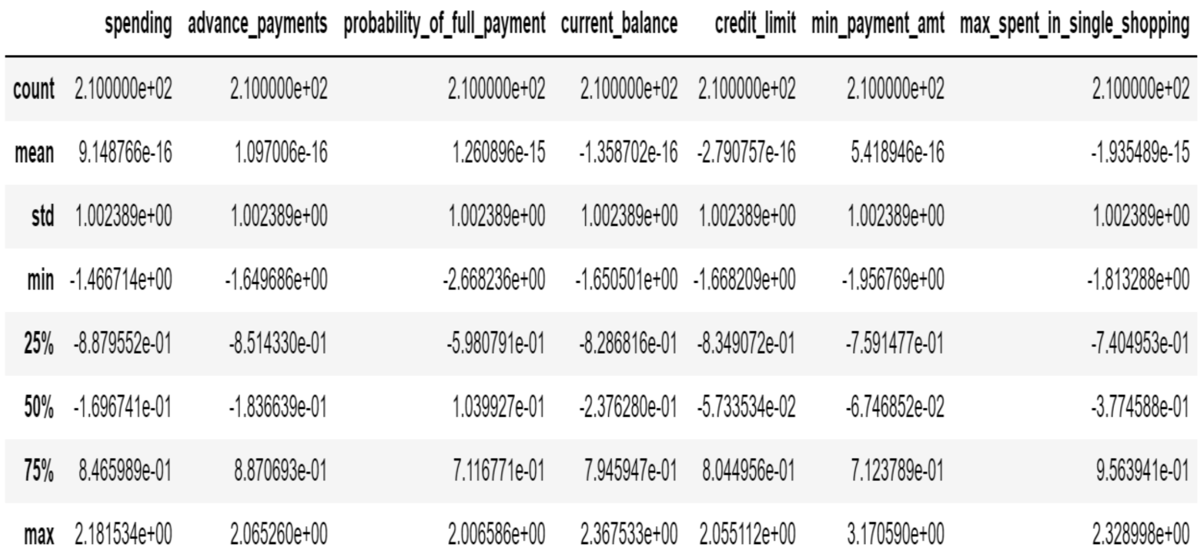
1.2 Scaling is necessary for this dataset for a better prediction and well to reduce the outlier from the dataset. If we look into the variables behavior such as df[probability of full payment] is from scale 0.1 to 0.9 ,where as spending ranges from 10.59 to 21.In the original dataset the values are mentioned in the 100s ,1000s, and 10000s. To make a uniformity in the distribution and to increase the chance of prediction accuracy , it is better to scale the given data.

Let us save the new scaled dataset into a variable named scaled_df



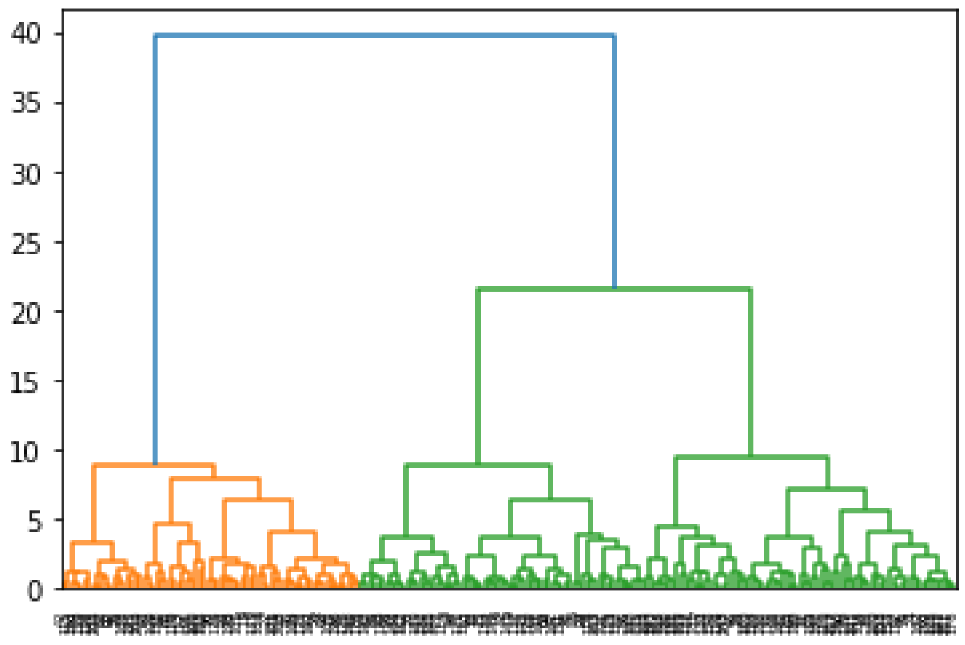
	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
0	1.754355	1.811968	0.178230	2.367533	1.338579	-0.298806	2.328998
1	0.393582	0.253840	1.501773	-0.600744	0.858236	-0.242805	-0.538582
2	1.413300	1.428192	0.504874	1.401485	1.317348	-0.221471	1.509107
3	-1.384034	-1.227533	-2.591878	-0.793049	-1.639017	0.987884	-0.454961
4	1.082581	0.998364	1.196340	0.591544	1.155464	-1.088154	0.874813
...	...	...	...	...	...	...	...
205	-0.329866	-0.413929	0.721222	-0.428801	-0.158181	0.190536	-1.366631
206	0.662292	0.814152	-0.305372	0.675253	0.476084	0.813214	0.789153
207	-0.281636	-0.306472	0.364883	-0.431064	-0.152873	-1.322158	-0.830235
208	0.438367	0.338271	1.230277	0.182048	0.600814	-0.953484	0.071238
209	0.248893	0.453403	-0.776248	0.659416	-0.073258	-0.706813	0.960473

After the scaling the mean of all variables becomes o and standard deviation equal to 1.Now all the variables are uniformity standard distribution.

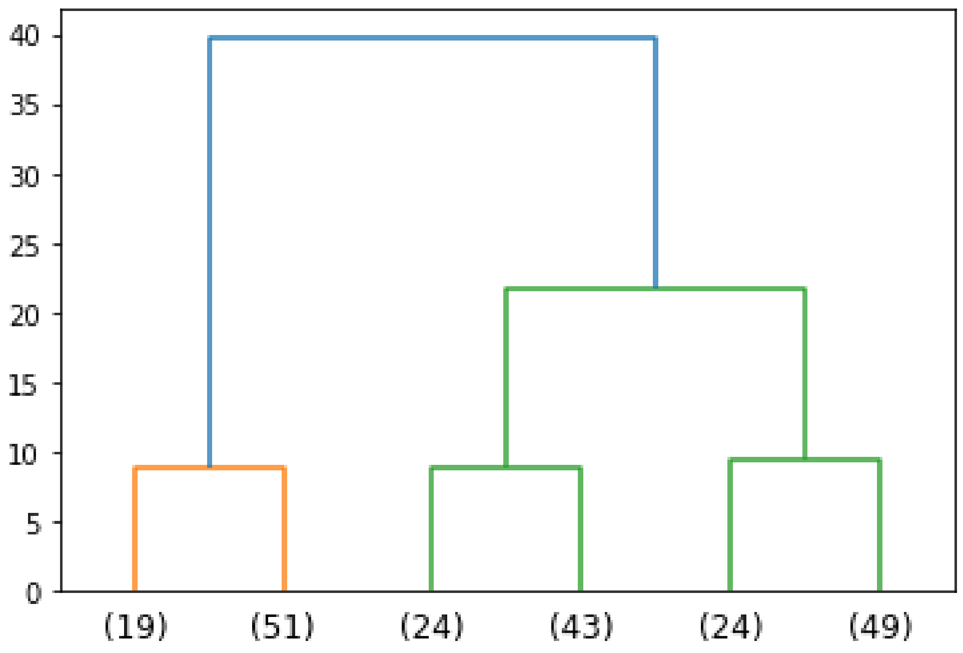


	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
count	2.100000e+02	2.100000e+02	2.100000e+02	2.100000e+02	2.100000e+02	2.100000e+02	2.100000e+02
mean	9.148766e-16	1.097006e-16	1.260896e-15	-1.358702e-16	-2.790757e-16	5.418946e-16	-1.935489e-15
std	1.002389e+00	1.002389e+00	1.002389e+00	1.002389e+00	1.002389e+00	1.002389e+00	1.002389e+00
min	-1.466714e+00	-1.649686e+00	-2.668236e+00	-1.650501e+00	-1.668209e+00	-1.956769e+00	-1.813288e+00
25%	-8.879552e-01	-8.514330e-01	-5.980791e-01	-8.286816e-01	-8.349072e-01	-7.591477e-01	-7.404953e-01
50%	-1.696741e-01	-1.836639e-01	1.039927e-01	-2.376280e-01	-5.733534e-02	-6.746852e-02	-3.774588e-01
75%	8.465989e-01	8.870693e-01	7.116771e-01	7.945947e-01	8.044956e-01	7.123789e-01	9.563941e-01
max	2.181534e+00	2.065260e+00	2.006586e+00	2.367533e+00	2.055112e+00	3.170590e+00	2.328998e+00

1.3



After importing the dendrogram and linkage function from sklearn libaray.From dendrogram function applying the wardlink method to calculate the distance between the observation. We have performed the dendrogram to find the optimum number of clusters required for the prediction. From the above figure , we can see that the dendrogram has suggested **two clusters i.e orange and green**. But from the given data we can truncate the dendrogram for a better visualization. Let us show the last 6 points where clusters are getting merged by giving an additional parameters into the dendrogram function.



From the above figure we can see that orange clusters have 70 observation and green cluster have 140 observation. Let us use fcluster function to for cluster labelling.

```
1      70
2     140
Name: clusters, dtype: int64
```

	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	2
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

After that we have added one more columns named cluster and merged all the observation to their corresponding clusters in the dataset. Using pivot_table function we can segregate the clusters with their mean values in the columns for recommendation.

	advance_payments	credit_limit	current_balance	max_spent_in_single_shopping	min_payment_amt	probability_of_full_payment	spending
clusters							
1	16.235077	3.705569	6.183723	6.047231	3.636262	0.884386	18.569231
2	13.808069	3.058241	5.379655	5.121552	3.728863	0.864997	13.179172

From the above figure we can see that cluster 1 have more spending nature than cluster 2 . People from cluster 1 have more credit limits comparing to cluster two. The maximum spending in a single shopping for cluster two has to be increased .

If we try out with agglomerative clustering , we will get the same result as above with fcluster techniques.

```
0    140
1     70
Name: Agglo_CLusters, dtype: int64
```

From the above pivot table we can conclude that cluster 1 has more spending nature comparing to cluster 2. Since their credit card limit average is also high compares to cluster 2 . One thing to mention that from the inference we got is that the probability of paying full amount to the bank for cluster 1 is low comparing to cluster 2 .

1.4 Let us apply kmean clustering technique to the scaled dataset. From the dendrogram we understood that two cluster is the ideal optimum cluster required for this dataset. Let us prove it with kmean clustering by checking with their respective Silhouette score.

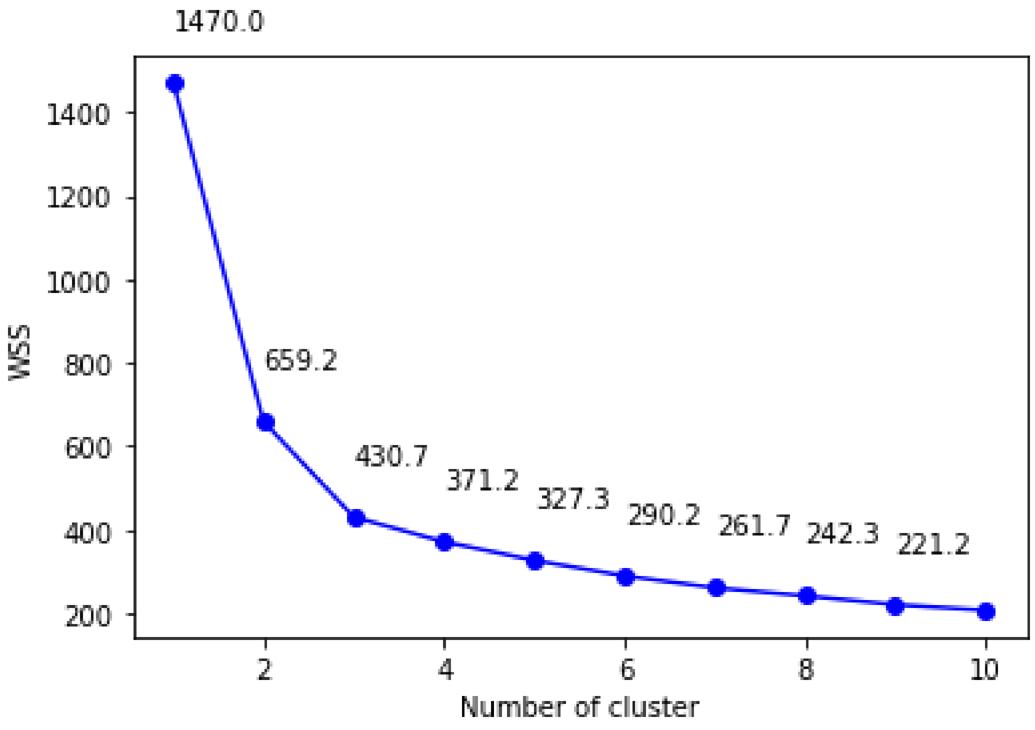
First let us apply kmean function with cluster value equal 2 and random state value equal to 20.We got the result for inertia and its respective silhouette score.

```
K mean inertia score for two clusters is  659.171754487041

labels= [0 1 0 1 0 1 1 1 0 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1 1 0 1 1 1 1 1
1 1 0 1 1 1
 1 1 0 0 1 0 0 1 1 1 0 0 0 1 0 0 0 0 0 1 1 1 0 1 1 1 1 1 0 0 1 0 1 1 1
0 0
 1 0 1 1 0 1 1 1 1 0 1 1 0 0 0 1 1 0 1 1 1 0 0 0 1 0 1 0 1 0 1 0 0 1 1
0 0
 1 0 1 1 0 1 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1 1 0 1 0 0 1 0 1 1 1 1 1 1 1
0 1
 1 1 1 1 1 0 1 1 1 1 1 0 0 1 0 0 0 1 0 1 1 1 0 1 1 0 0 0 1 1 0 1 1 1 1
1 0
 0 1 1 1 1 1 1 1 0 1 0 0 1 0 1 1 0 1 1 0 1 0 1 0 0]

Silhouette score for two clusters 0.46577247686580914
```

Above is the result with kmean function for two clusters. Now lets see whether three clusters makes any impact with k=3. From the jupyter notebook , we got the result as follows :-

```
K mean inertia score for three clusters is 430.6589731513006

Silhouette score for three clusters is 0.4007270552751299
```

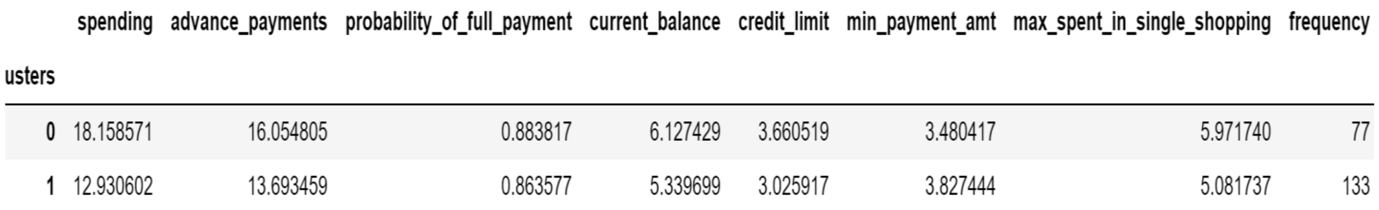
From the above result, we can see our silhouette score got dropped down for k=3. So let us stick to k=2 which has better silhouette score. Alternatively we can prove this with ELBOW METHOD and represent using a elbow curve. After applying the algorithm we get the result as follows:-



```
[1469.9999999999998,
 659.171754487041,
 430.6589731513006,
 371.1846125351018,
 327.3281094192775,
 290.1747600952111,
 261.700182170189,
 242.34148143430295,
 221.1964380482988,
 207.7267784204649]
```

From the above indicate that there is good gap between cluster 1 and cluster 2.We can see that there is a huge step down of distance from cluster 1 to cluster 2 than cluster 2 to cluster 3. From the silhouette score it is again clear that the all observation can be separated into two cluster which is good for our model.

1.5 Now let us add one more column to label the dataset with their respective kmean clusters. Lets us do the cluster profiling to give better insights.



	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	frequency
usters								
0	18.158571	16.054805	0.883817	6.127429	3.660519	3.480417	5.971740	77
1	12.930602	13.693459	0.863577	5.339699	3.025917	3.827444	5.081737	133

Above figure shows the cluster profiling output. After sorting the number of observation for cluster 0 and 1 and grouping them to their respective columns means values gives us some better insights. K means clustering and hierarchical clustering techniques are saying the same information. We can recommend the following points to improvise the marketing campaigns for the respective clusters

1.) Cluster 0 has good spending nature compares to cluster 1.

2.) Cluster 0 customers use to pay advance amounts to their accounts better than cluster 1 customers.

3.) Cluster 0 customers are good in paying the money in full respective to their credit limits. They have more credit limits and have good chance to pay the amount in one-shot. Whereas the cluster 1 has 2% less chance of paying the full amount .

4.) There is only slight difference between the current balance of both parties.

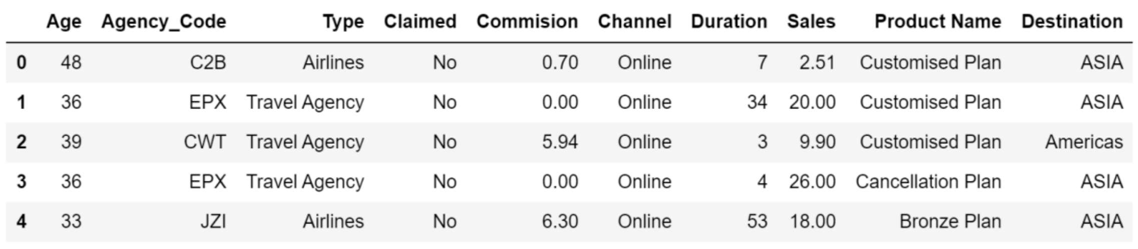
5.) Customers from the cluster 1 has more chance to pay the minimum amount comparing to cluster 1. We can improvise this factor giving more promotional offers to cluster 1 to increase the minum payment amount from cluster 1 .

6.) Giving more promotional offers to cluster 0 can also increase the maximum spend in a single day. Because they use to shop for higher values compares to cluster1 which can increase the revenue.

7.) It is better to give more promotional offers to cluster 0 w.r.t cluster 1 since cluster 1 always have a good nature of spending. By giving customized promotion for both parties can increase the usage of the respective cards.

Problem 2

2.1. After importing the dataset into a variable called df2. Let us explore the dataset using head function and also check for any null values in the dataset.



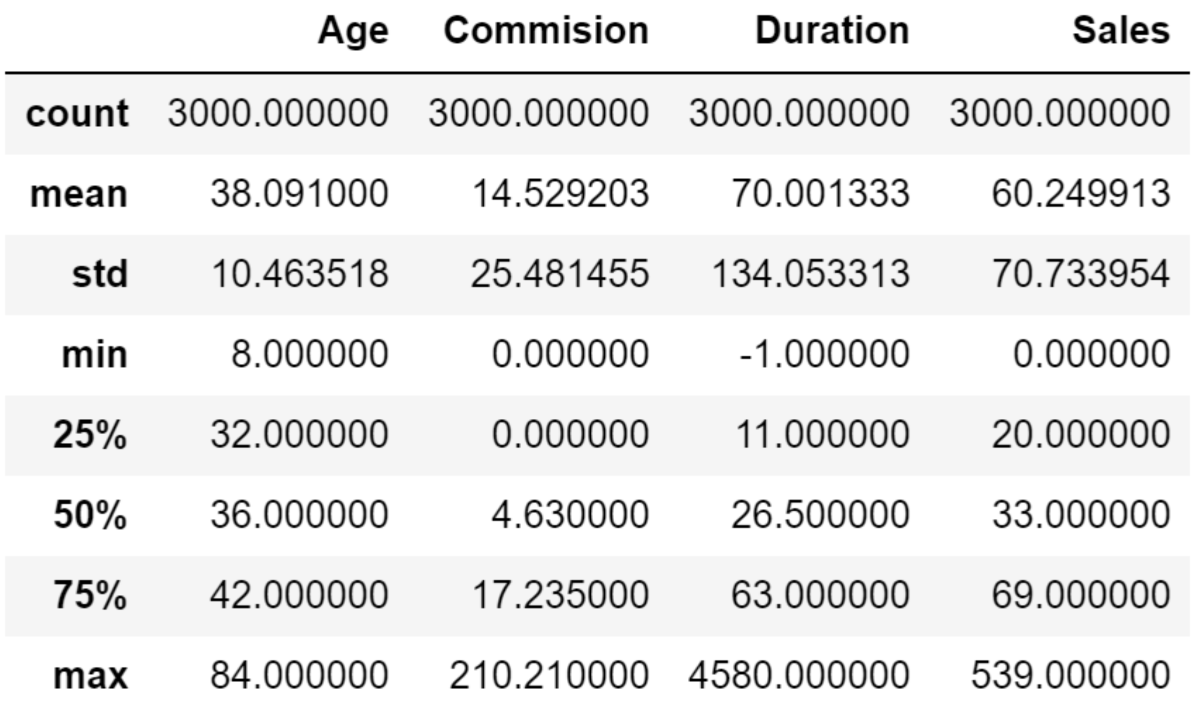
	Age	Agency_Code	Type	Claimed	Commision	Channel	Duration	Sales	Product Name	Destination
0	48	C2B	Airlines	No	0.70	Online	7	2.51	Customised Plan	ASIA
1	36	EPX	Travel Agency	No	0.00	Online	34	20.00	Customised Plan	ASIA
2	39	CWT	Travel Agency	No	5.94	Online	3	9.90	Customised Plan	Americas
3	36	EPX	Travel Agency	No	0.00	Online	4	26.00	Cancellation Plan	ASIA
4	33	JZI	Airlines	No	6.30	Online	53	18.00	Bronze Plan	ASIA

This represent the age of the customer, code of the tour agency, type of insurance claim, total hour of tour duration and respective sales. In this column , claimed variable is the target variable and remaining all are independent variable of the dataset.

From the null function , we got to know that there is no null values in the dataset.

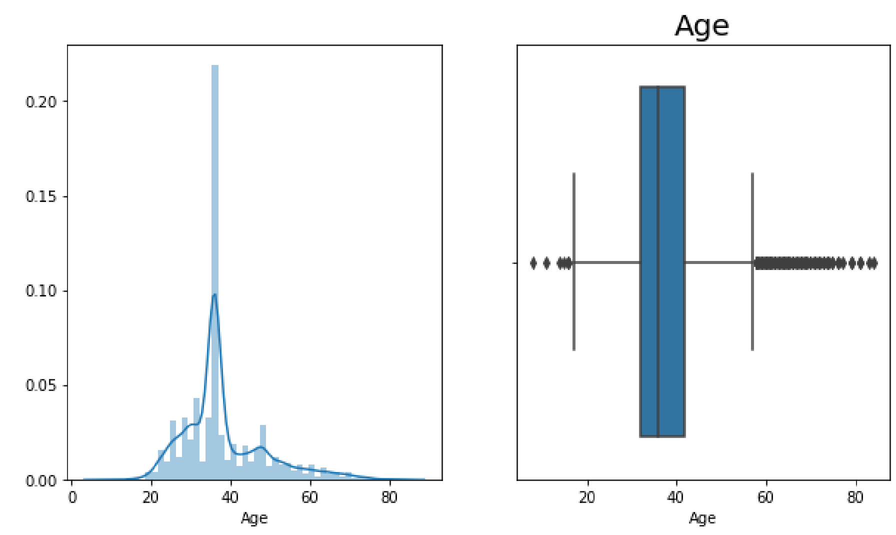
```
Age             0
Agency_Code     0
Type            0
Claimed         0
Commision       0
Channel         0
Duration        0
Sales           0
Product Name    0
Destination     0
```

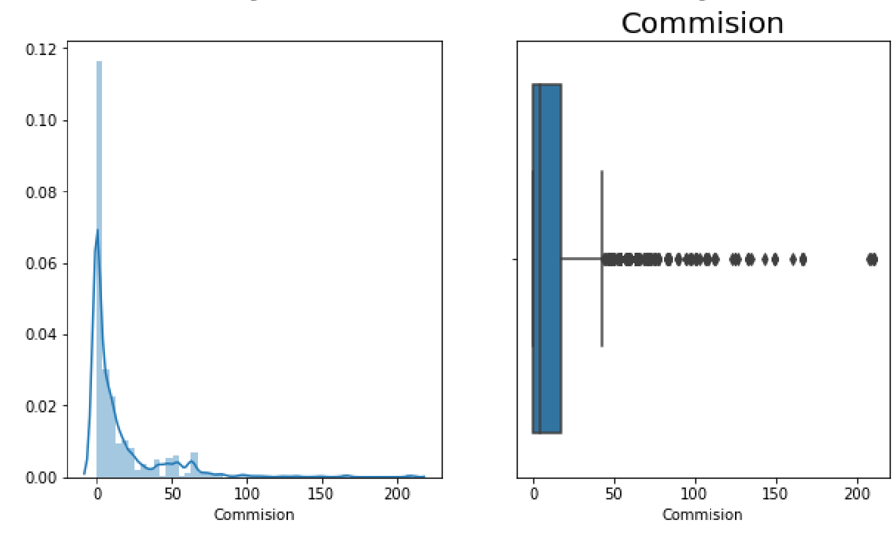
Then lets us do the descriptive summary of the dataset . Since we are evaluating numerical values only. In this output , we can see that most of the variables means are somewhat to equal to their median values. There is no high amount of variance in the variables.

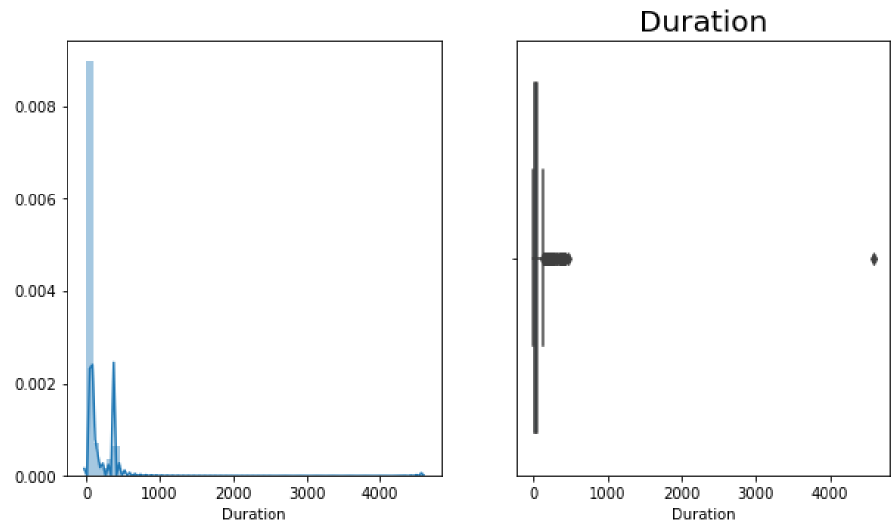


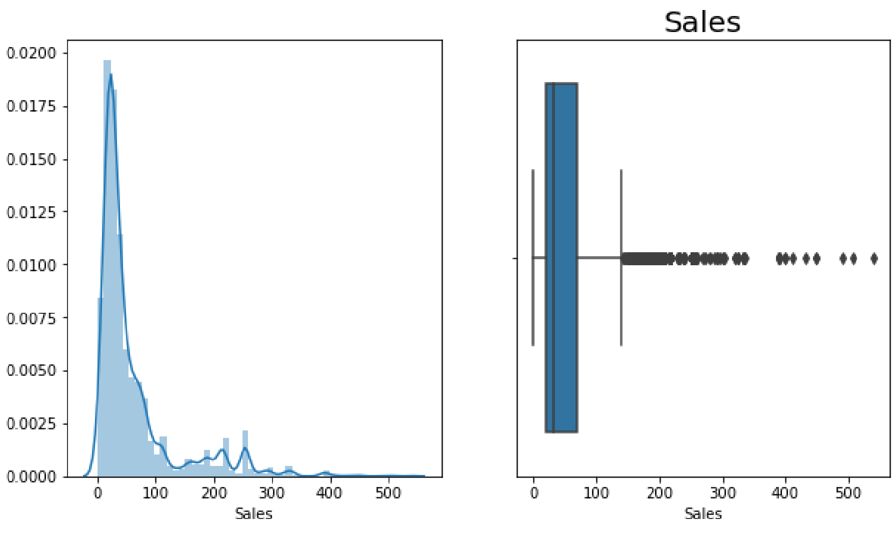
	Age	Commision	Duration	Sales
count	3000.000000	3000.000000	3000.000000	3000.000000
mean	38.091000	14.529203	70.001333	60.249913
std	10.463518	25.481455	134.053313	70.733954
min	8.000000	0.000000	-1.000000	0.000000
25%	32.000000	0.000000	11.000000	20.000000
50%	36.000000	4.630000	26.500000	33.000000
75%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

Let us check the histogram representation to understand the skewness and distribution of the variables.


Age
0.20
0.15
0.10
0.05
0.00
0
20
40
60
80
Age
20
40
60
80
Age

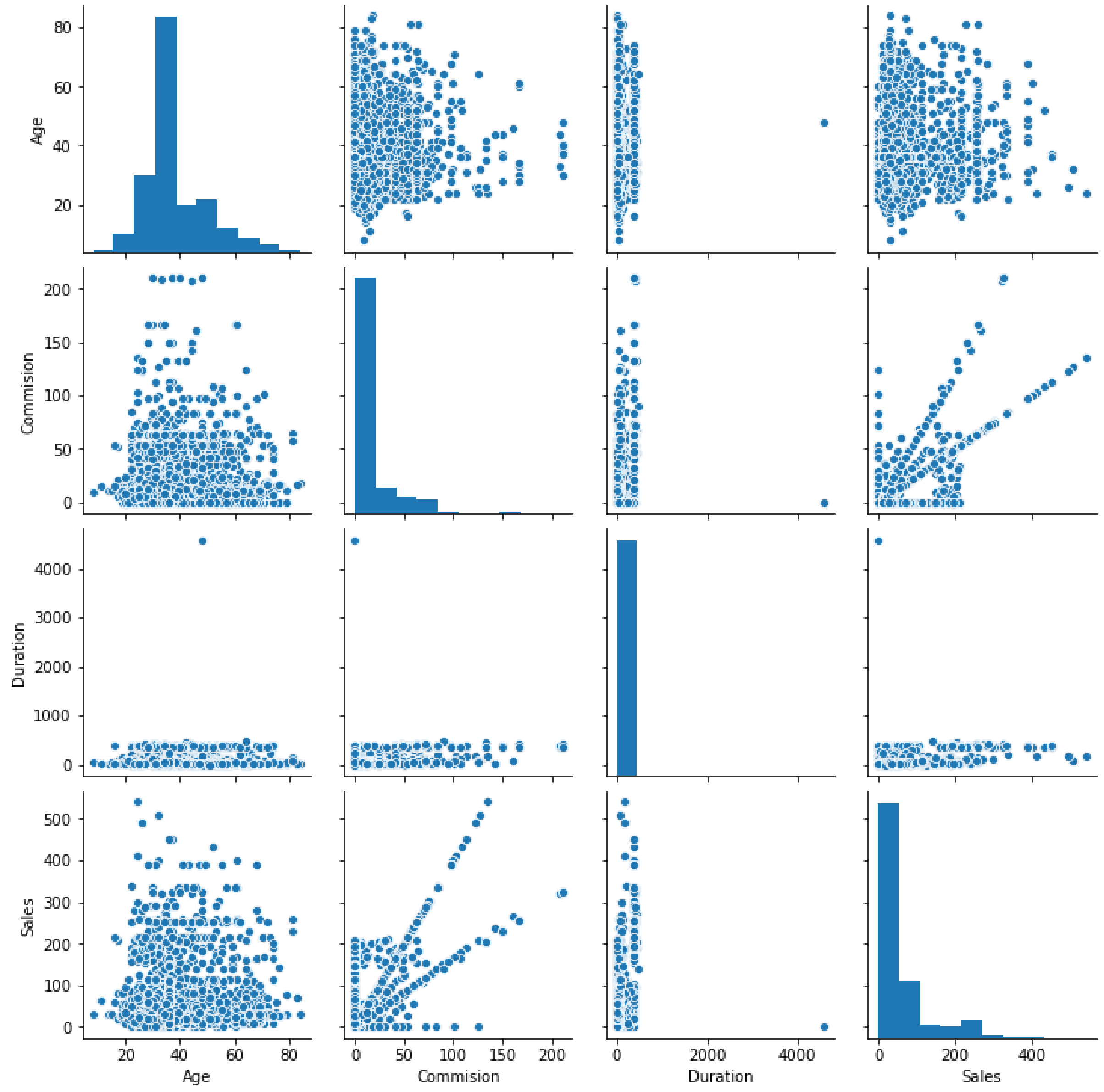

Commision
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0
50
100
150
200
Commision
0
50
100
150
200
Commision


Duration
0.008
0.006
0.004
0.002
0.000
0
1000
2000
3000
4000
Duration
0
1000
2000
3000
4000
Duration


Sales
0.0200
0.0175
0.0150
0.0125
0.0100
0.0075
0.0050
0.0025
0.0000
0
100
200
300
400
500
Sales
0
100
200
300
400
500
Sales

Most of the variables have outliers and that is not affected for our random forest and CART modelling. Let us perform outlier treatment for ANN modelling.After checking the skewness almost all numerical values have positive skewness, only duration have higher skewness on the side

```
Age            1.149713
Commision      3.148858
Duration      13.784681
Sales          2.381148
dtype: float64
```



Let us know check the datatypes of the data set.

```
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)
```

We have to convert all object datatypes into numerical datatypes for modelling.

```
---  ------         --------------  -----
 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)
```

After the conversion we can the see output with zero null values.Now our dataset is good for splitting into training and testing dataset for modelling.

Let us also check the value counts and their respective codes. Also we have converted all categorical variabels into their respectives codes. So that we can easily count the types of observations in each variables.

```
2    1365
0     924
1     472
3     239
Name: Agency_Code, dtype: int64

1    1837
0    1163
Name: Type, dtype: int64

0    2076
1     924
Name: Claimed, dtype: int64

2    1136
1     678
0     650
4     427
3     109
Name: Product Name, dtype: int64

1    2954
0      46
Name: Channel, dtype: int64

0    2465
1     320
2     215
Name: Destination, dtype: int64
```

These are converted categorical variables into their code values. Let us also check the imbalance of the target variable.

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

Our dataset doesn’t hold any imbalance problems. 70% with 0 codes and 30% have claimed for the insurance. This good is for our modelling purposes.

2.2 CART

We have split the dataset into training and testing data. We are taking 30% dataset for testing and 70% data for training .Let us check their respective observations.

```
Age              66
Agency_Code       4
Type              2
Commision       273
Channel           2
Duration        228
Sales           319
Product Name      5
Destination       3
dtype: int64


shape of training data = (2100, 9)


Age              60
Agency_Code       4
Type              2
Commision       165
Channel           2
Duration        180
Sales           217
Product Name      5
Destination       3
dtype: int64

shape of testing data = (900, 9)
```

Let us build the CART model to predict the test data from the training variables. Let us do the CART modelling into two ways, with and without hypertuning. Let us use the Decision trees for the CART modelling and with random state=1 .

After fitting the training variables to the model. Let us now check the variable importance of the decision tree.

	Importance
Duration	0.271340
Sales	0.211506
Age	0.179722
Agency_Code	0.175714
Commision	0.100519
Product Name	0.041354
Destination	0.013214
Channel	0.004510
Type	0.002121

After applying different parameter into gridsearchCV function , we got the best parameters from the gridsearchCV function. We will save this parameter into a new variable and fit into the model to get the accurate prediction.

```
                   Imp
Agency_Code   0.333839
Sales         0.225027
Duration      0.164016
Age           0.113349
Commision     0.088623
Product Name  0.052862
Destination   0.017179
Channel       0.005104
Type          0.000000
```

After applying the new model with hypertuning , we get these variables as most important. We can see that the importance percentage has increased for most of the variables. Agency_code and Sales remains the important variables from the both models.

Random Forest

Since random forest is working with ensembling , let us use the variable without scaling. After importing the variables from sklearn. Perform the gridsearchCV function to tune the parameters to get the best parameters.

By default function using sklearn with 501 n_estimators , we have predicted the test variables. Below o/p shows the probability of customer claiming for the insurance for each observation.

```
array([[0.9760479 , 0.0239521 ],
       [0.24351297, 0.75648703],
       [0.98802395, 0.01197605],
       ...,
       [0.98403194, 0.01596806],
       [0.86826347, 0.13173653],
       [0.55489022, 0.44510978]])
```

Also below o/p shows the variable importance using randomforestclassifiier

```
                   Imp
Duration      0.259742
Sales         0.214311
Age           0.179722
Commision     0.121695
Agency_Code   0.088201
Product Name  0.088129
Destination   0.021718
Type          0.021011
Channel       0.005470
```

We can see the variable we got through decision tree and random forest are having equal importances.

With help of hypertuning , let again do the modelling using randomforest. Below represent the o/p for the best parameter for fitting the model.

```
GridSearchCV(cv=9, estimator=RandomForestClassifier(), n_jobs=-1,
             param_grid={'max_depth': [7], 'max_features': [5],
                         'min_samples_leaf': [5, 6],
                         'min_samples_split': [240, 245],
                         'n_estimators': [101, 501]})
```

From the above parameters, we got the best estimators as below :-

```
{'max_depth': 7,
 'max_features': 5,
 'min_samples_leaf': 5,
 'min_samples_split': 240,
 'n_estimators': 501}
```

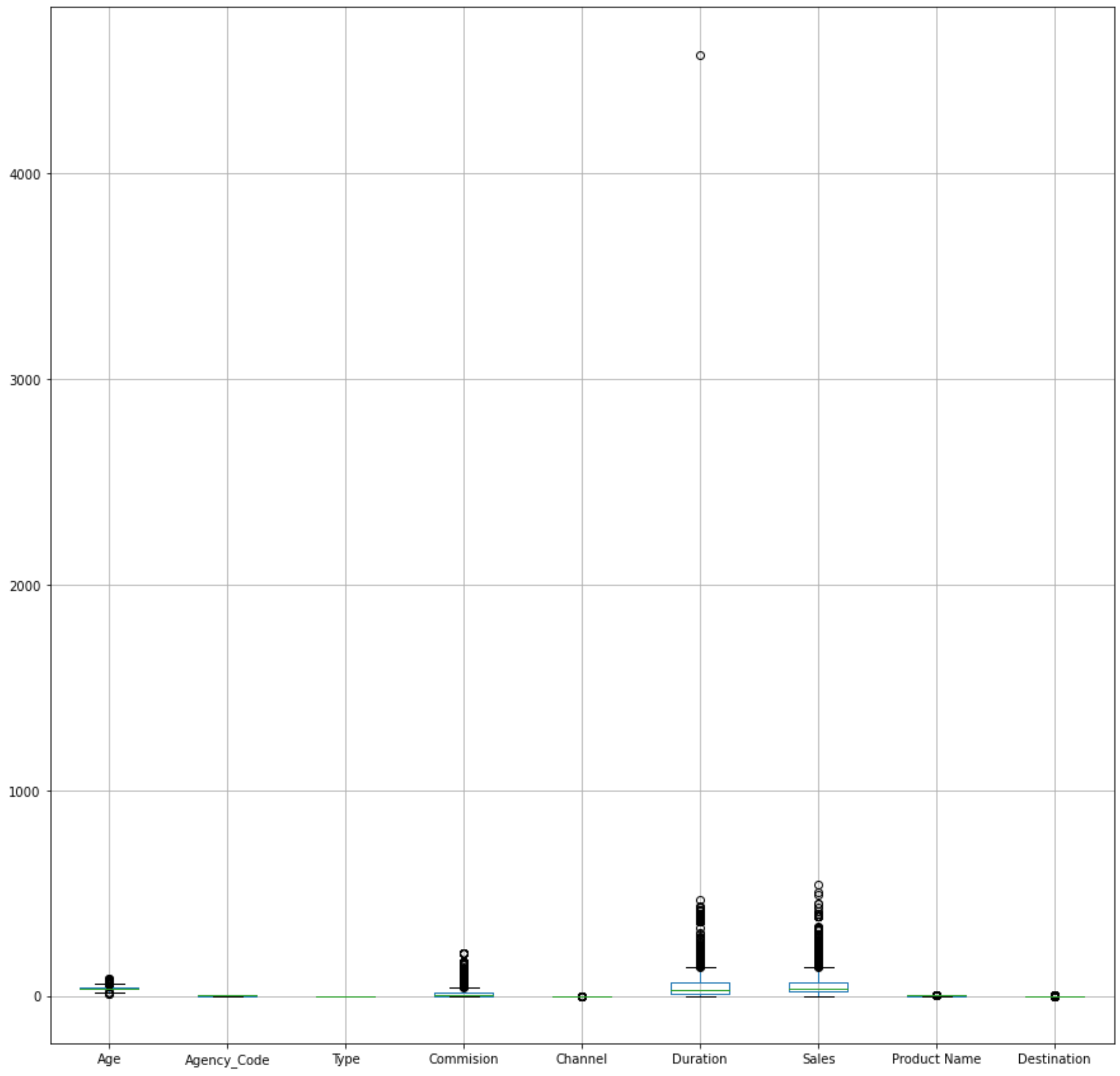
After tuning , lets check for the variable importance .

```
                    Imp
Agency_Code    0.390939
Product Name   0.285229
Sales          0.158513
Commision      0.086614
Duration       0.033348
Type           0.027288
Age            0.010039
Destination    0.005387
Channel        0.002642
```

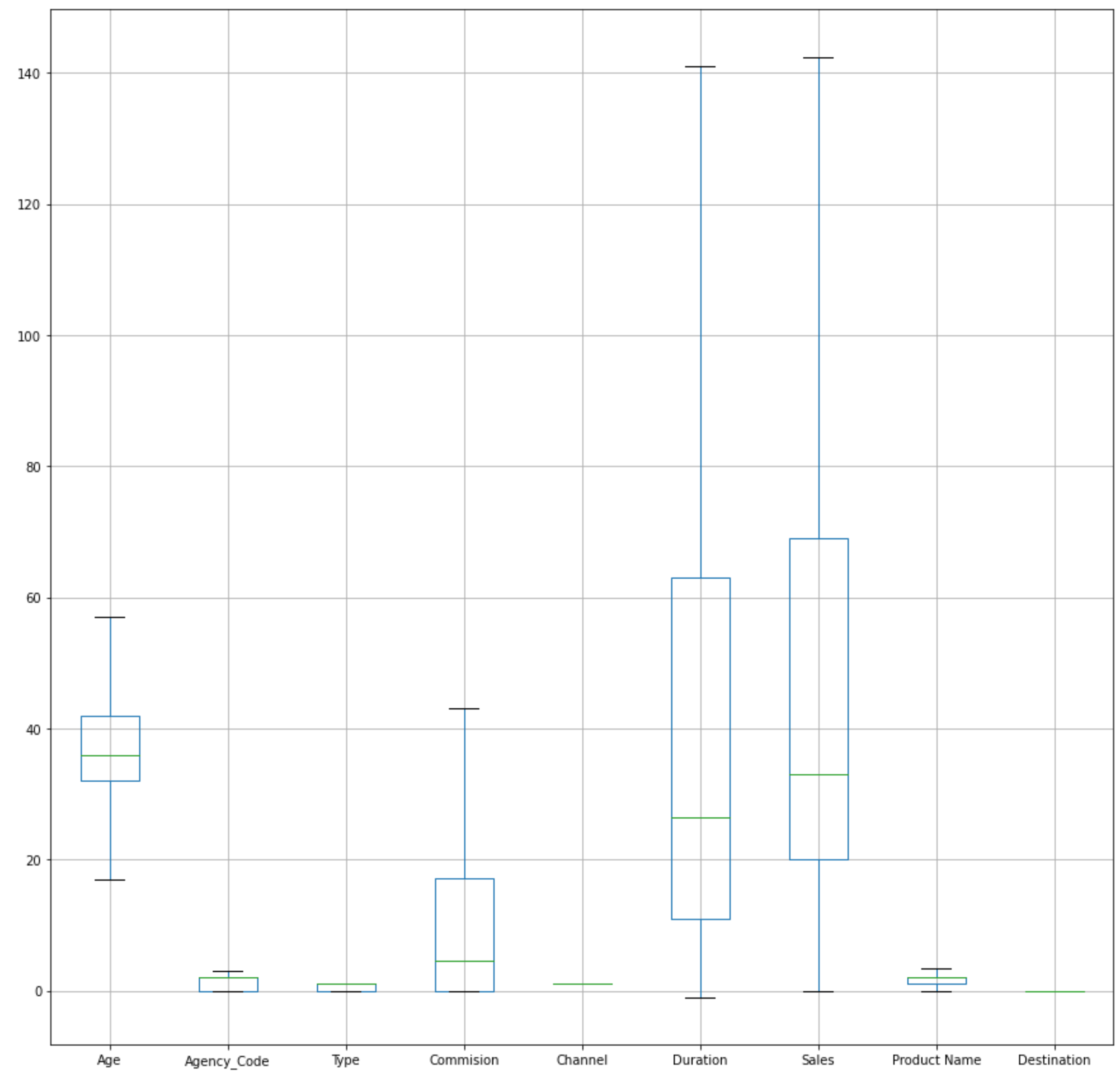
Both the tuned and non-tuned models represent same variables as important.

ANN

For ANN model, its better to scale the model and also let us check for the outliers too.



Most of the variables has outliers, let us remove these outliers using IQR methods.



Now lets import the StandardScaler function and save into a new variable.We have fit and transform the scaled training variables and predict the testing data. After importing the MLPClassifier , lets fit the model using the variables.

We have calculate the hidden layer by calculating

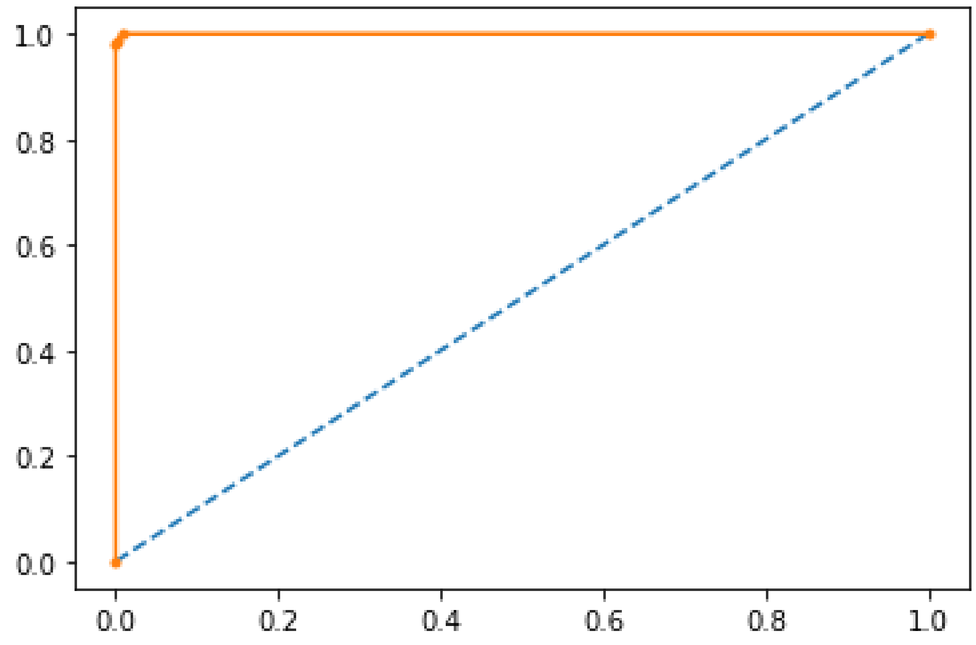
Total observation/ constant*(i/p variables +1) . With this function , we have selected the number of hidden layer as 136 with max iteration of 5000. Random state keeping as 1 and tolerance value as 0.01 using adam solver , lets us find out the loss function for the respective iteration with tolerance value of 0.01

```
Iteration 1, loss = 0.69182255
Iteration 2, loss = 0.59551182
Iteration 3, loss = 0.54260270
Iteration 4, loss = 0.51614776
Iteration 5, loss = 0.50436064
Iteration 6, loss = 0.49802134
Iteration 7, loss = 0.49488356
Iteration 8, loss = 0.49262648
Iteration 9, loss = 0.49123177
Iteration 10, loss = 0.48961417
Iteration 11, loss = 0.48835178
Iteration 12, loss = 0.48756995
Iteration 13, loss = 0.48635169
Iteration 14, loss = 0.48548007
Iteration 15, loss = 0.48452412
Iteration 16, loss = 0.48361920
Training loss did not improve more than tol=0.010000 for 10 consecutive epo
chs. Stopping.
MLPClassifier(hidden_layer_sizes=136, max_iter=5000, random_state=1, tol=0.
01,
              verbose=True)
```

2.3 Performance metrics :

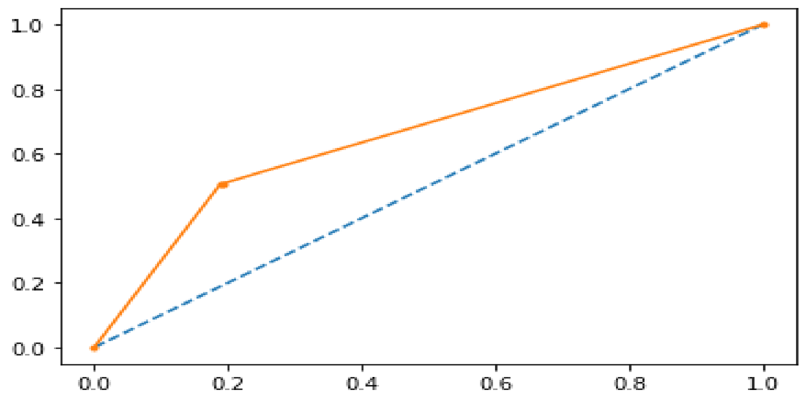
CART

We can see that Duration, sales , Age , Agency_code, Comission these variables are highly important for the insurance claim. Now lets us Predict the test variables from the training set. With help of ROC and AUC curve function let us determine the area under the curve and receiver characteristic operator.



```
AUC: 1.000
```

Above figure shows the ROC curve for training variables with 100% area under curve.Now let us check for the test variables with the same procedures.



```
AUC: 0.658
```

We can see the area under the curve for testing variable is 65.8% only. Let evaluate more in details by applying confusion matrix and classification report for both testing and training variables.

```
array([[1463,    1],
       [  10,  626]], dtype=int64)
```

Above o/p represent the FP ,TP, TN,TP values for training variables with a model accuracy of 99.47%.

Let us now check the classification report to analyse the precision, sensitivity, F1-score of the training model with the accuracy value.

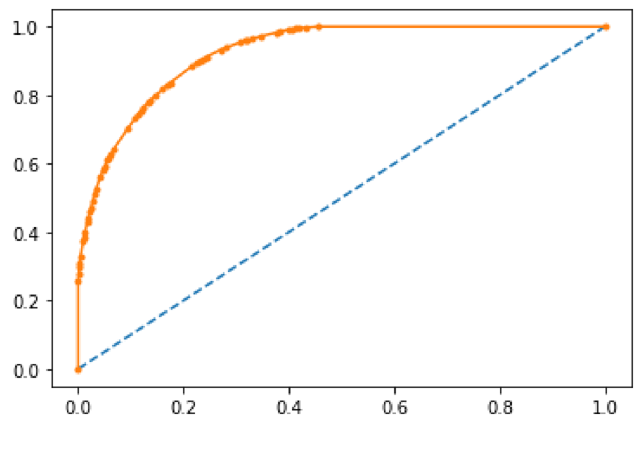
```
              precision    recall  f1-score   support

           0       0.99      1.00      1.00      1464
           1       1.00      0.98      0.99       636

    accuracy                           0.99      2100
   macro avg       1.00      0.99      0.99      2100
weighted avg       0.99      0.99      0.99      2100
```

We can see the training variables have 98% recall and 100% precision and 99% of model accuracy. This looks good !. Now lets evaluate the confusion matrix and classification report for predicted test variables.

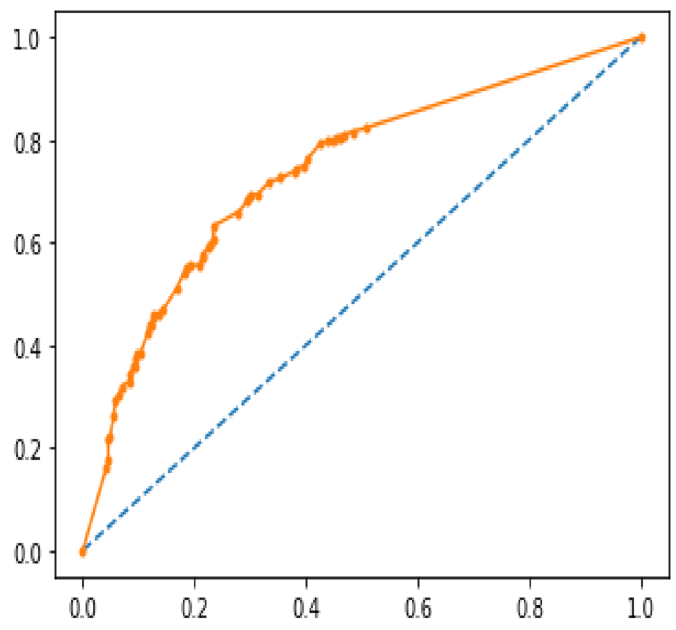
```
             precision    recall  f1-score   support

           0       0.78      0.81      0.79       612
           1       0.56      0.50      0.53       288

    accuracy                           0.71       900
   macro avg       0.67      0.66      0.66       900
weighted avg       0.71      0.71      0.71       900
```

We got precision value of 56% ,recall 50% and model accuracy of 71%. There is a great drop down from 99% to 71% on predicted variables. This shows our model is overfitted. Let us reduce this with help of tuning using GridSearchCV function.

Now lets us check the ROC and AUC curve for both hypertuned testing and training variables .



```
AUC: 0.924
```



```
AUC: 0.739
```

The first figure shows the ROC and AUC for training and second figure represent the testing variables. Let us represent the confusion matrix and classification report for both,

Training

```
array([[1366,   98],
       [ 229,  407]], dtype=int64)

              precision    recall  f1-score   support

           0       0.86      0.93      0.89      1464
           1       0.81      0.64      0.71       636

    accuracy                           0.84      2100
   macro avg       0.83      0.79      0.80      2100
weighted avg       0.84      0.84      0.84      2100
```

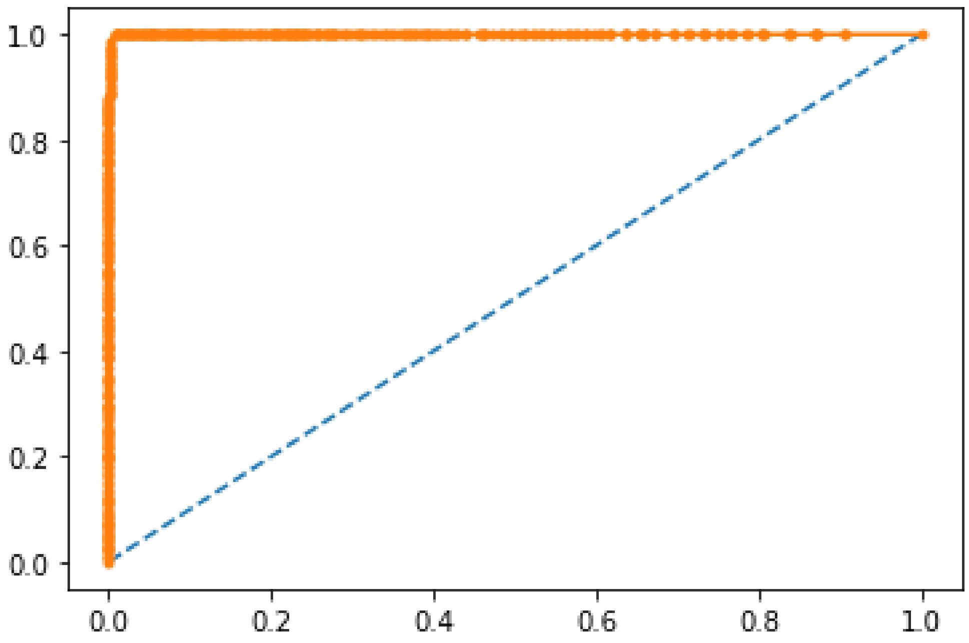
Testing

```
array([[524,  88],
       [153, 135]], dtype=int64)
              precision    recall  f1-score   support

           0       0.77      0.86      0.81       612
           1       0.61      0.47      0.53       288

    accuracy                           0.73       900
   macro avg       0.69      0.66      0.67       900
weighted avg       0.72      0.73      0.72       900
```

Random Forest

Let's us find out the performance of the model with hyper tuned as well without tuned model. First lets see the ROC and AUC for non-tuned model

```
[0.00399202 0.68263473 0.99401198 ... 0.00598802 0.00199601 0.05189621]
AUC: 1.000
```

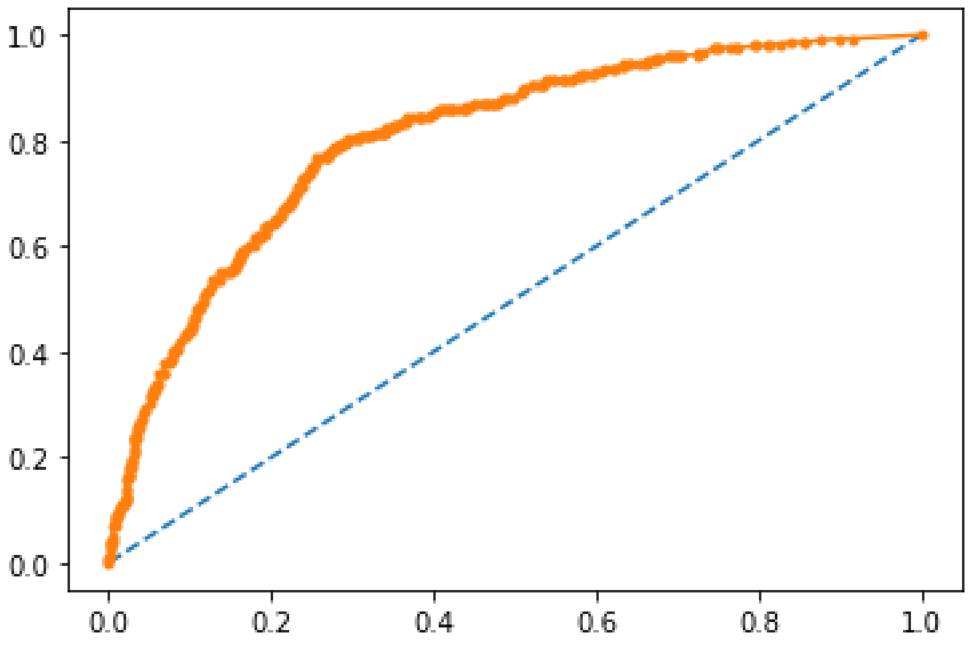


We can see that Area under the curve is 100% and training obererervation as in well manner same as of decision tree.

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      1464
           1       0.99      0.99      0.99       636

    accuracy                           0.99      2100
   macro avg       0.99      0.99      0.99      2100
weighted avg       0.99      0.99      0.99      2100
```

Above shows the classification report for the training dataset with **model accuracy of 99%.** Now lets us calculate the same steps for testing variables

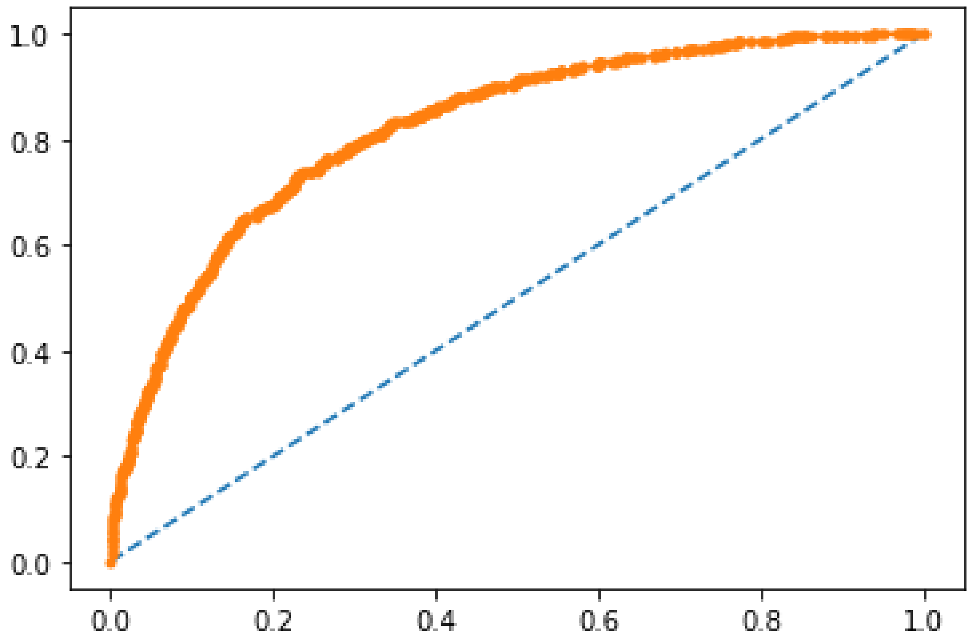
```
AUC: 0.807
```



```
              precision    recall  f1-score   support

           0       0.79      0.88      0.83       612
           1       0.66      0.51      0.58       288

    accuracy                           0.76       900
   macro avg       0.73      0.70      0.71       900
weighted avg       0.75      0.76      0.75       900
```

We can see the area under the curve improved , comparing to the decision tree. The precision and recall have also got improved . This model has an accuracy level of **76%** which can be also overfitted compared to the training model accuracy. Let's us tune the model with some parameters using GRIDSearchCV function. And show the o/p of their AUC and ROC curve.

```
AUC: 0.823
```



Classification report-training

```
              precision    recall  f1-score   support

           0       0.81      0.89      0.85      1464
           1       0.68      0.51      0.58       636

    accuracy                           0.78      2100
   macro avg       0.74      0.70      0.72      2100
weighted avg       0.77      0.78      0.77      2100
```

Confusion matrix-training

```
array([[1316,  148],
       [ 310,  326]], dtype=int64)
```

Let us do the same performance metrics analysis for testing data using randomforest.

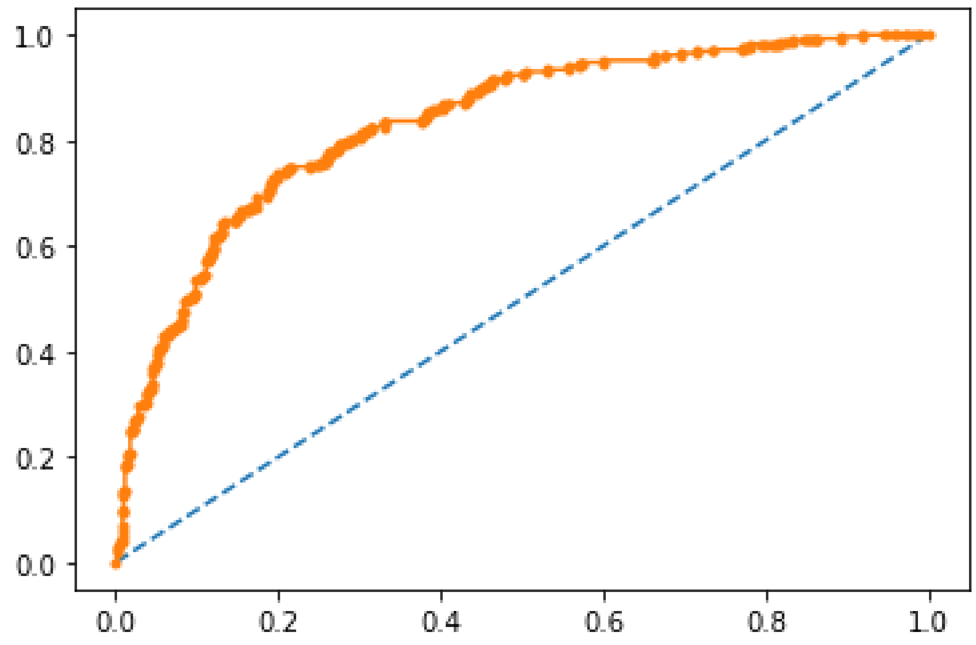
Confusion matrix-testing

```
array([[560,  52],
       [145, 143]], dtype=int64)
```

Classification report-testing

```
                  precision    recall  f1-score   support

           0         0.79      0.92      0.85       612
           1         0.73      0.50      0.59       288

    accuracy                             0.78       900
   macro avg         0.76      0.71      0.72       900
weighted avg         0.77      0.78      0.77       900
```

AUC: 0.835



ANN

For ANN , we have to scale the data and remove the outlier for better treatment. Using MLPclassifier ,with an hiddenlayer of 136,lets us findout the performance of the model.

Confusion matrix -training
```
array([[1317,  337],
       [ 147,  299]], dtype=int64)
```

Model accuracy training : 0.7695238095238095

Classification report-training
```
              precision    recall  f1-score   support

           0       0.90      0.80      0.84      1654
           1       0.47      0.67      0.55       446

    accuracy                           0.77      2100
   macro avg       0.68      0.73      0.70      2100
weighted avg       0.81      0.77      0.78      2100
```

Let us do the same for the testing variables

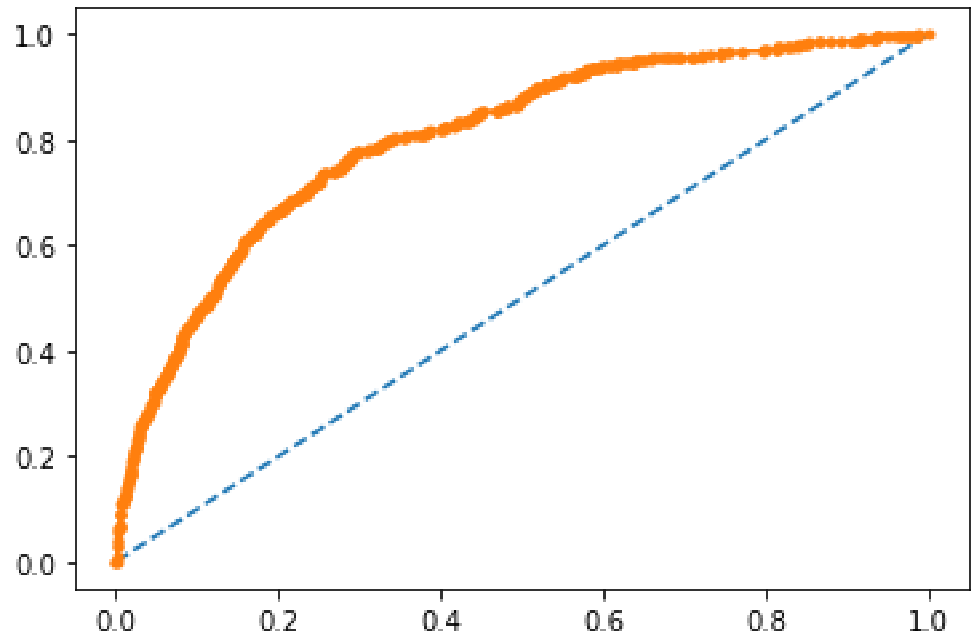
Confusion matrix -testing
```
array([[569, 153],
       [ 43, 135]], dtype=int64)
```

Model accuracy testing :78.22%

Classification report-testing

```
              precision    recall  f1-score   support

           0       0.93      0.79      0.85       722
           1       0.47      0.76      0.58       178

    accuracy                           0.78       900
   macro avg       0.70      0.77      0.72       900
weighted avg       0.84      0.78      0.80       900
```

```
AUC: 0.805
```



The above figure shows the AOC and ROC curve for training dataset using MLP classifier.Lets perform the same procedures for testing variables and represent their ROC, AUC, Confusion matrix , Model accuracy and classification report.

Confusion matrix -testing

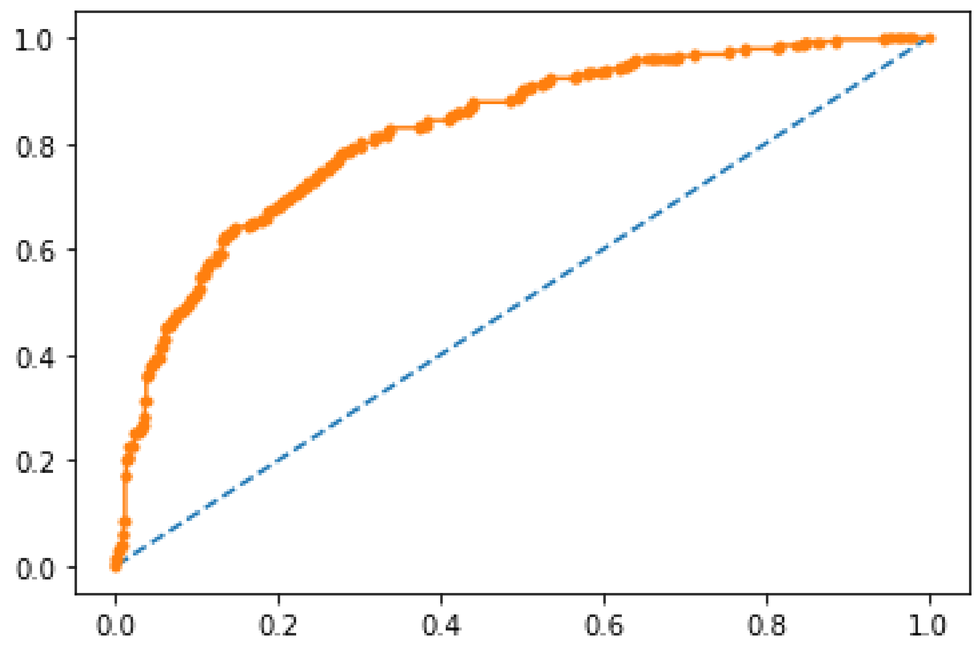
```
array([[569, 153],
       [ 43, 135]], dtype=int64)
```

Model accuracy testing = 0.7822222222222223

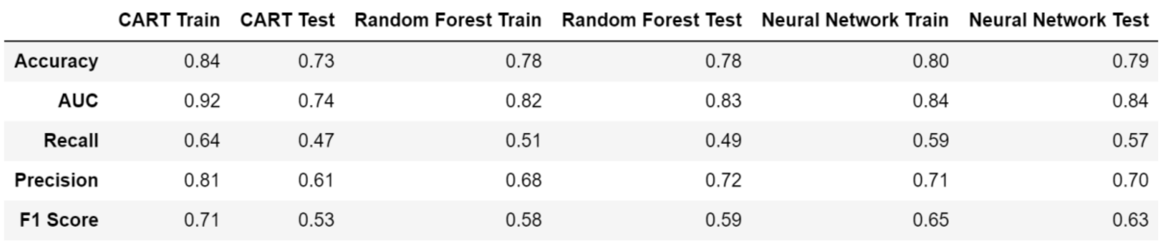
Classification report-testing

```
              precision    recall  f1-score   support

           0       0.93      0.79      0.85       722
           1       0.47      0.76      0.58       178

    accuracy                           0.78       900
   macro avg       0.70      0.77      0.72       900
weighted avg       0.84      0.78      0.80       900
```

```
AUC: 0.825
```



2.4



	CART Train	CART Test	Random Forest Train	Random Forest Test	Neural Network Train	Neural Network Test
Accuracy	0.84	0.73	0.78	0.78	0.80	0.79
AUC	0.92	0.74	0.82	0.83	0.84	0.84
Recall	0.64	0.47	0.51	0.49	0.59	0.57
Precision	0.81	0.61	0.68	0.72	0.71	0.70
F1 Score	0.71	0.53	0.58	0.59	0.65	0.63

From the above table, we can see the respective models accuracy. The CART hold a good training accuracy with 73% training accuracy.But there is overfitting problem for CART model and the precision and recall rate is poor compares to RF and ANN. Whereas the RF model is better than CART model which hold better precision and recall rates.NN model have a lesser accuracy compares to CART model. But the testing precision rate is better comparing to any model which good for our prediction. We can conclude that neural network is the suitable model comparing to any other models

	CART		Random forest		ANN	
	Training	Testing	Training	Testing	Training	Testing
Without tuning	99%	71	99%	76%	77%	78%
With tuning	84%	73%	78%	78%	80%	80%

	CART AUC value		Random forest AUC value		ANN AUC value	
	Training	Testing	Training	Testing	Training	Testing
Without tuning	100%	65.8%	100%	80.7%	80.5%	82.5%
With tuning	92.4%	73.9%	82.3%	83.5%	84%	84%

If we check the AUC percentage , we can see huge difference in training AUC and testing AUC for CART and random forest model.Whereas the NN model's Area under curve is somewhere equal in proportion which doesn't hold any overfitting problem too.

2.5 Comparing to all model, ANN networks hold less overfitting problem compares to CART model. RF and CART models accuracy is to an extend hold good for our prediction. Once we check the variable importance of all model , we can see that, SALES,COMISSION, FIRM , DURATION have an effect for customer claim. Most of the customer claiming because of high duration of flight journey. Since agency firm also have good importance , which we can identify from which agencies the insurance claim is high. So if we are checking their respective sales value for each agency firm can helps to find out at which avg sales the customers are claiming the insurance. For a better recommendation , improving the tour insurance sales performance with respect to agency_code can improves the sales performance of the company.