3/8/2022

DATA MINING PROJECT

Clustering AND CART-RF-ANN

Rekha Wankhede

Contents

Executive Summary:

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

Introduction:

The purpose of this exercise is to identify the segments based on credit card usage. Based on that bank can give promotional offer accordingly to their customer. 210 customers information of their spending, advance payments, probability_of_full_payment, current_balance, credit limit, min_payment_amt, max_spent_in_single_shopping. To analysis customer segmentation, various technic and models are used such as EDA, DATA visualizing, K-Mean, RF-Model, CART model and helpful insights for business purpose.

Data Description:

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

Sample of the dataset:

	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
118	11.48	13.05	0.8473	5.180	2.758	5.876	5.002
138	17.55	15.66	0.8991	5.791	3.690	5.366	5.661
77	12.13	13.73	0.8081	5.394	2.745	4.825	5.220
157	11.26	13.01	0.8355	5.186	2.710	5.335	5.092
105	18.94	16.49	0.8750	6.445	3.639	5.064	6.362

Check the shape of data

```
df.shape
```

```
(210, 7)
```

Dataset has 7 variables and 210 rows.

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

Let's check the information of data set:

```
<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 dataset has total 7 column and 210 rows, all data are in float data type. We can see that no null value present in dataset. Lets check whether any missing value present in dataset or not.

```
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
```

Its look like no miss value present in dataset.

DATA TYPES:

```
spending                        float64
advance_payments                float64
probability_of_full_payment     float64
current_balance                 float64
credit_limit                    float64
min_payment_amt                 float64
max_spent_in_single_shopping    float64
dtype: object
```

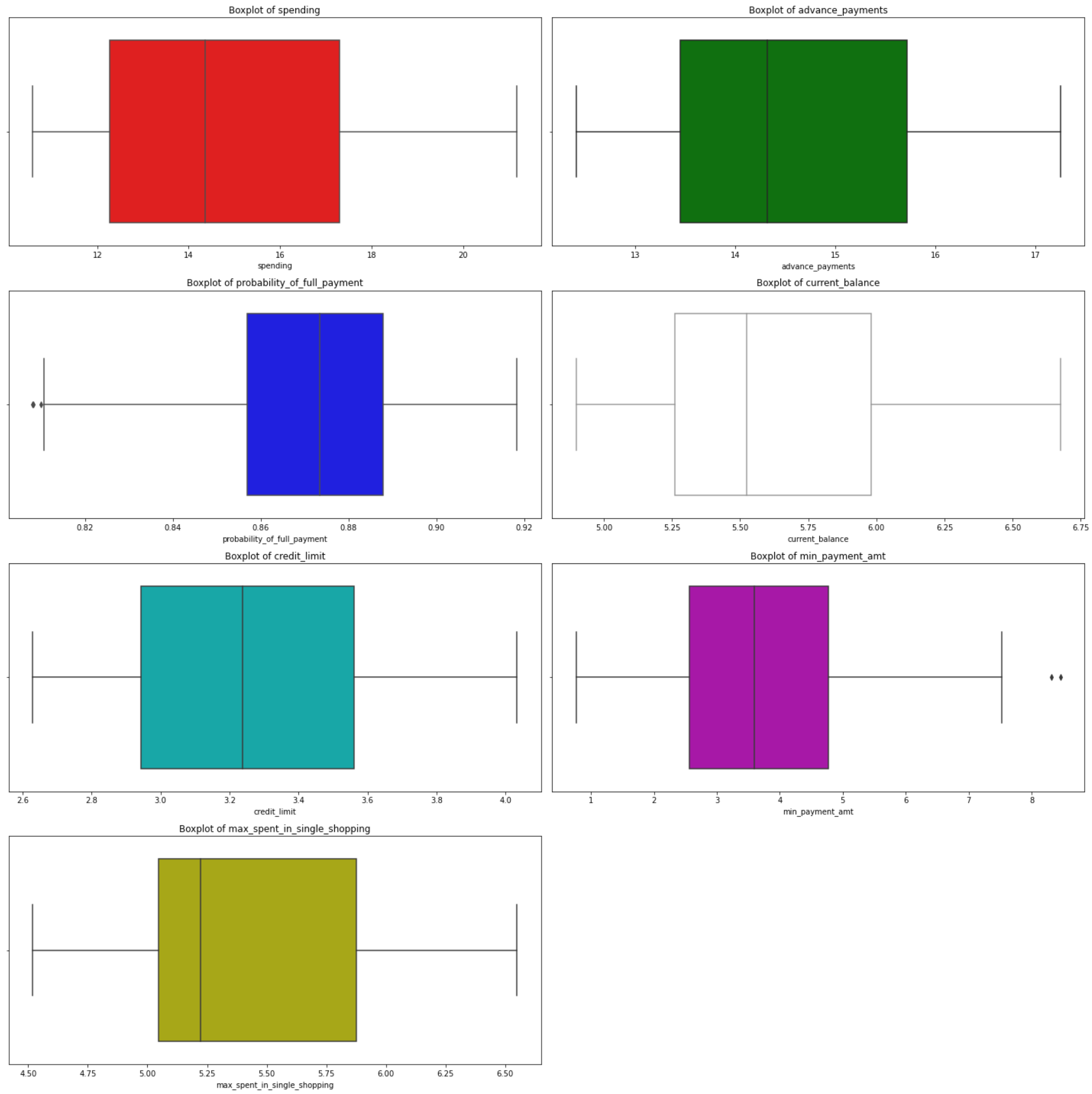
Describing the dataset:

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

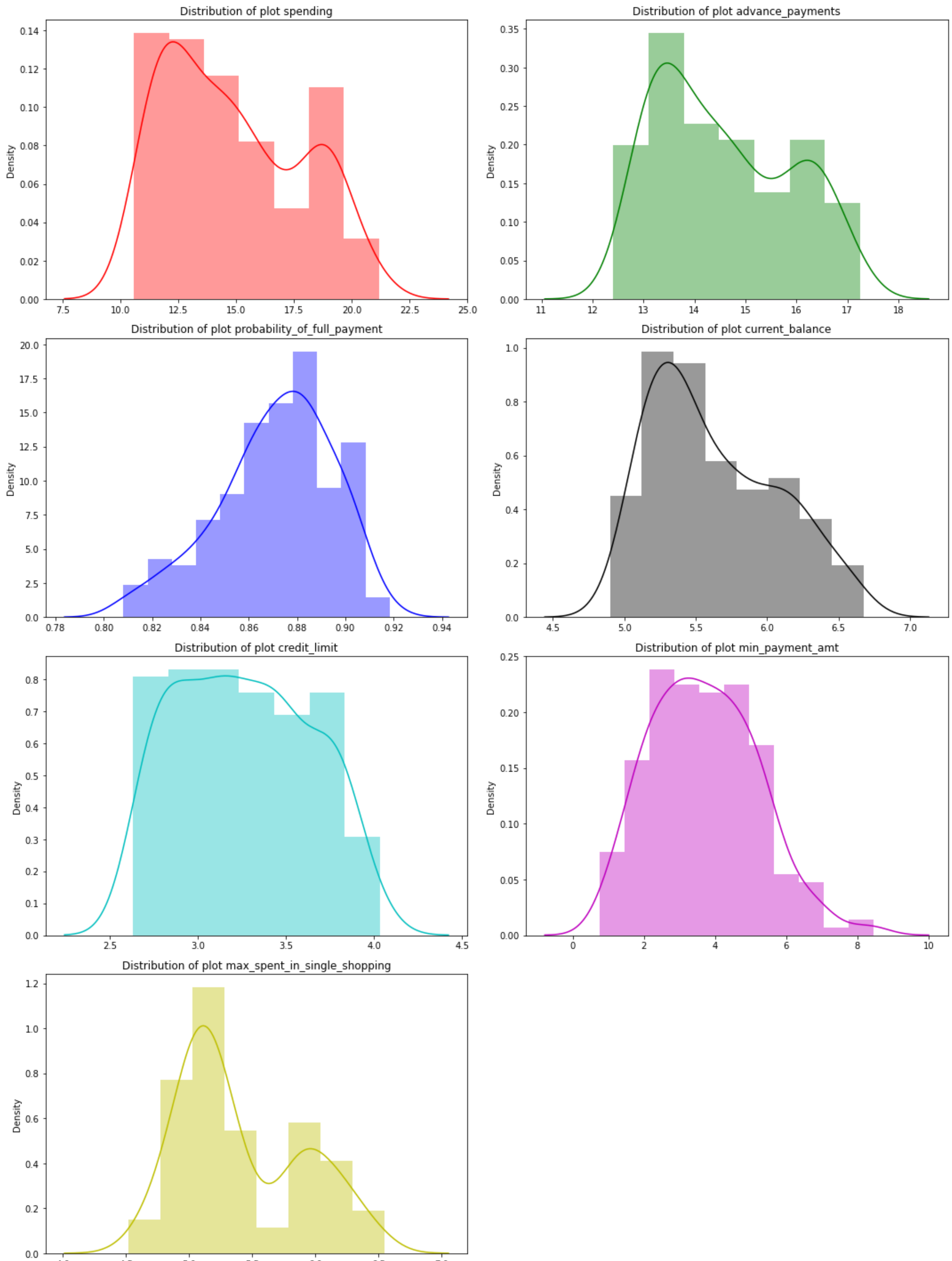
- From the above table below are the observations.
- Spending which target variable looks like its normally distributed as we can see that mean and medians are same
- Advance payment also seems to be normal distributed, this variable might be use as it shows that customers are paying the amount in advance which is timely payment of bank.
- The average of probability_of_full_payment is 87.099% hence we need to analysis the rest of customer who falls in 13% who have not done the payment in full. This variable is normally distributed
- Minimum current balance held by customer is 4899.0 and maximum is 6675.0.
- credit limit range between 2630.0 to 4033.0, The average credit limit of customers is 32586.05.
- minimum min_payment amount is 76.51 and maximum is 845.6. This suggest data is widely spread for this variable and might have outlier.
- The average of max_spent_in_single_shopping is 5408.07. The maximum of max_spent_in_single_shopping is 6550.00.

Exploratory Data Analysis:

Lets check symmetry and skewed with BOX PLOT:



Distribution Plot:

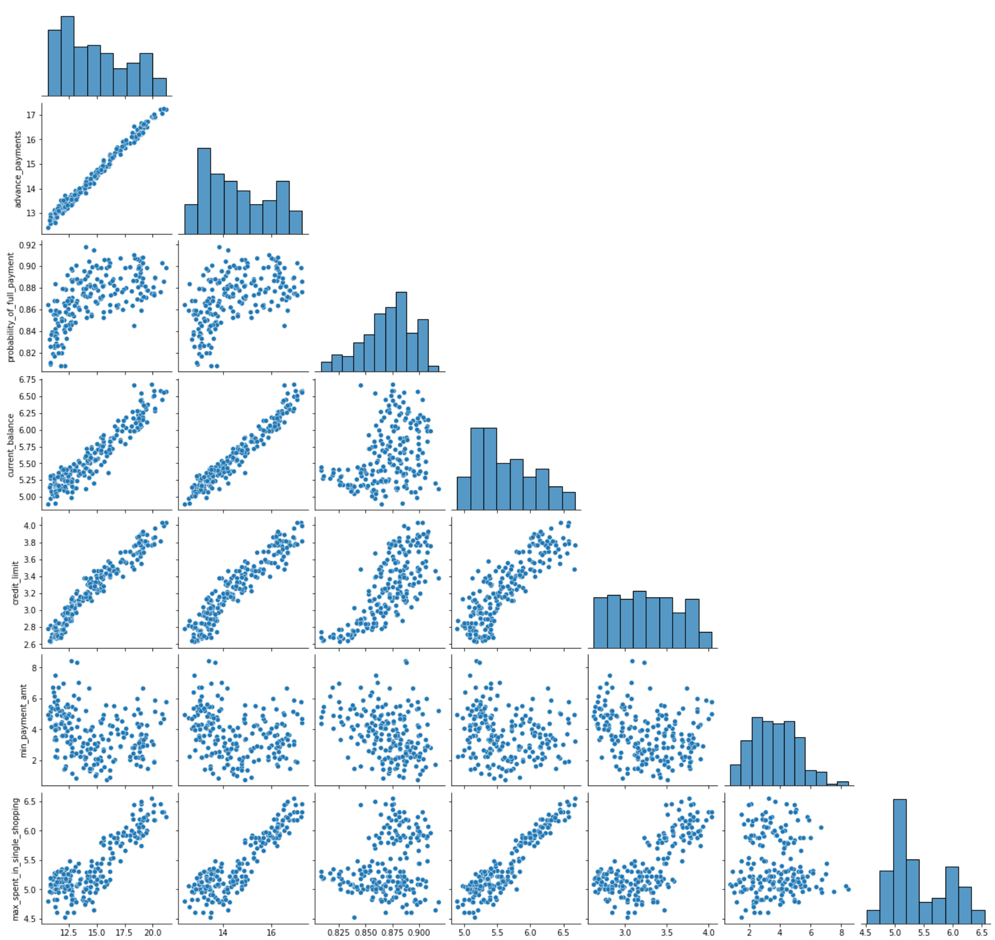

Distribution of plot spending
Density
Distribution of plot advance_payments
Density
Distribution of plot probability_of_full_payment
Density
Distribution of plot current_balance
Density
Distribution of plot credit_limit
Density
Distribution of plot min_payment_amt
Density
Distribution of plot max_spent_in_single_shopping
Density

Skewness and Kurtosis

```
skewness of spending is 0.4
Kurtosis of spending is -1.08
skewness of advance_payments is 0.39
Kurtosis of advance_payments is -1.11
skewness of probability_of_full_payment is -0.54
Kurtosis of probability_of_full_payment is -0.14
skewness of current_balance is 0.53
Kurtosis of current_balance is -0.79
skewness of credit_limit is 0.13
Kurtosis of credit_limit is -1.1
skewness of min_payment_amt is 0.4
Kurtosis of min_payment_amt is -0.07
skewness of max_spent_in_single_shopping is 0.56
Kurtosis of max_spent_in_single_shopping is -0.84
```

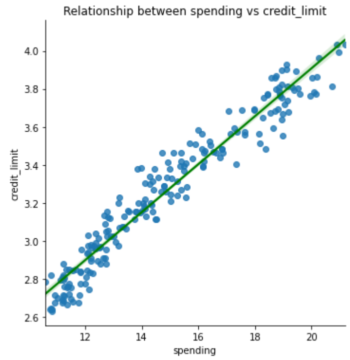
Bivariate Analysis

#Pairplots



#Lmplots


Relationship between spending vs credit_limit
current_balance
spending

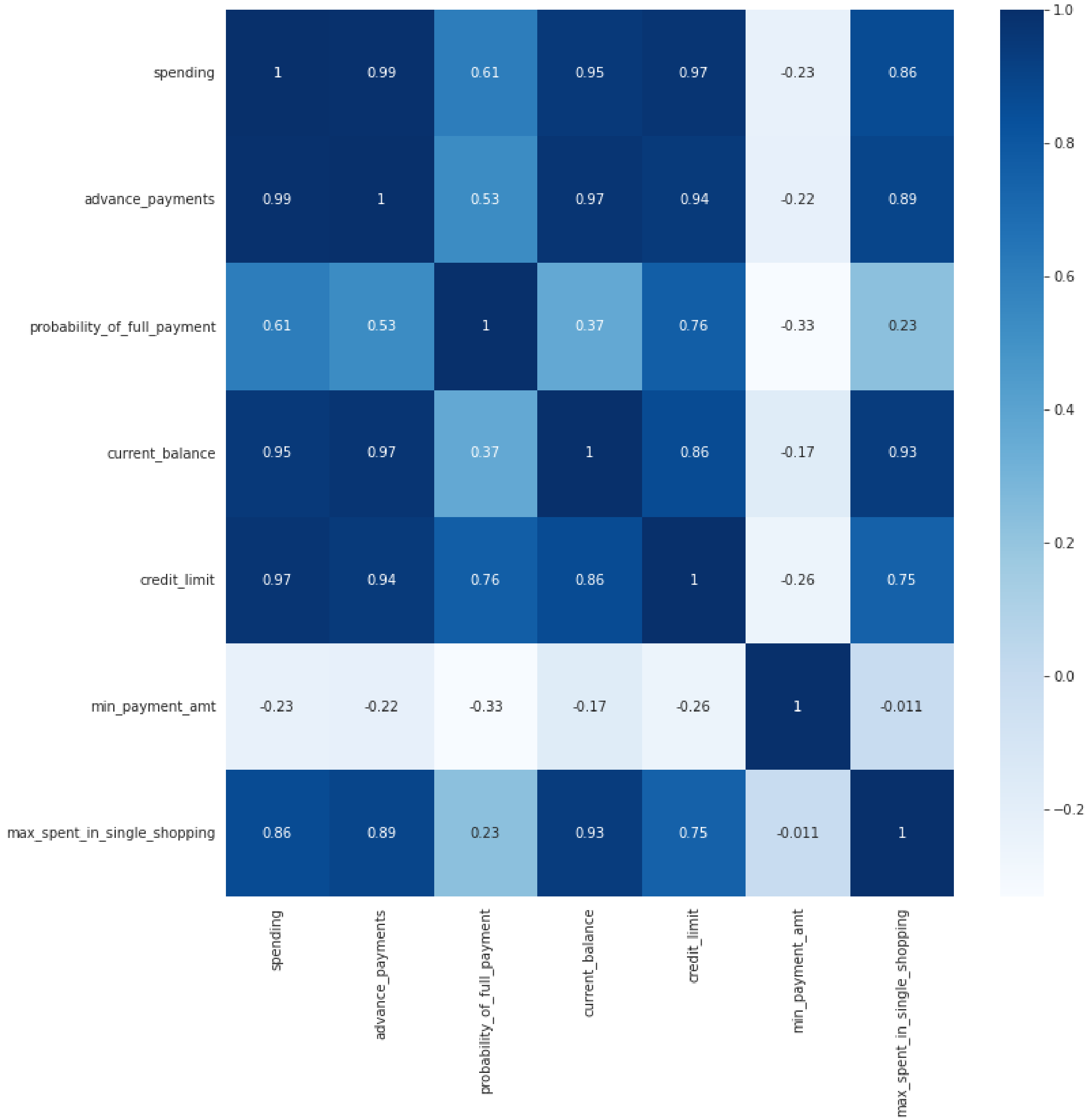

Relationship between spending vs credit_limit
credit_limit
spending


Relationship of spending w.r.t max_spent_in_single_shopping
max_spent_in_single_shopping
current_balance


Relationship between spending vs credit_limit
current_balance
spending


Relationship of spending w.r.t max_spent_in_single_shopping
max_spent_in_single_shopping
current_balance

Correlation Heatmaps



from above pair plot and heatmap we can see that there is positive linear relationship between advance payments and spending, current_balance and spending, credit_limit and spending, current_balance and advance_payments, credit_limit and advance_payments, max_spent_in_single_shopping and current_balance. This suggests that there is Multicollinearity between the variables.

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

Let us see the variance of each variable

```
spending                        8.466351
advance_payments                1.705528
probability_of_full_payment     0.000558
current_balance                 0.196305
credit_limit                    0.142668
min_payment_amt                 2.260684
max_spent_in_single_shopping    0.241553
dtype: float64
```

From the above table though there is not much variance between most of the variables,

our target variable spending has a variance of 8.46 whereas other variables variance lie between 0 and 2.

Hence scaling is necessary.

We will be using the Standard Scaler method for scaling our data. This method will calculate the z-score for each data point and then scale the data such that mean = 0 and variance/standard deviation = 1.

Larger differences between the data points of input variables increase the uncertainty in the results of the model. ... Scaling the target value is a good idea ; scaling of the data makes it easy for a model to learn and understand the problem The most common techniques of feature scaling are Normalization and Standardization. Normalization is used when we want to bound our values between two numbers, typically, between [0,1] or [-1,1]. While Standardization transforms the data to have zero mean and a variance of 1, they make our data unitless.

```
scaled_df
```

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

```
scaled_df.shape
```

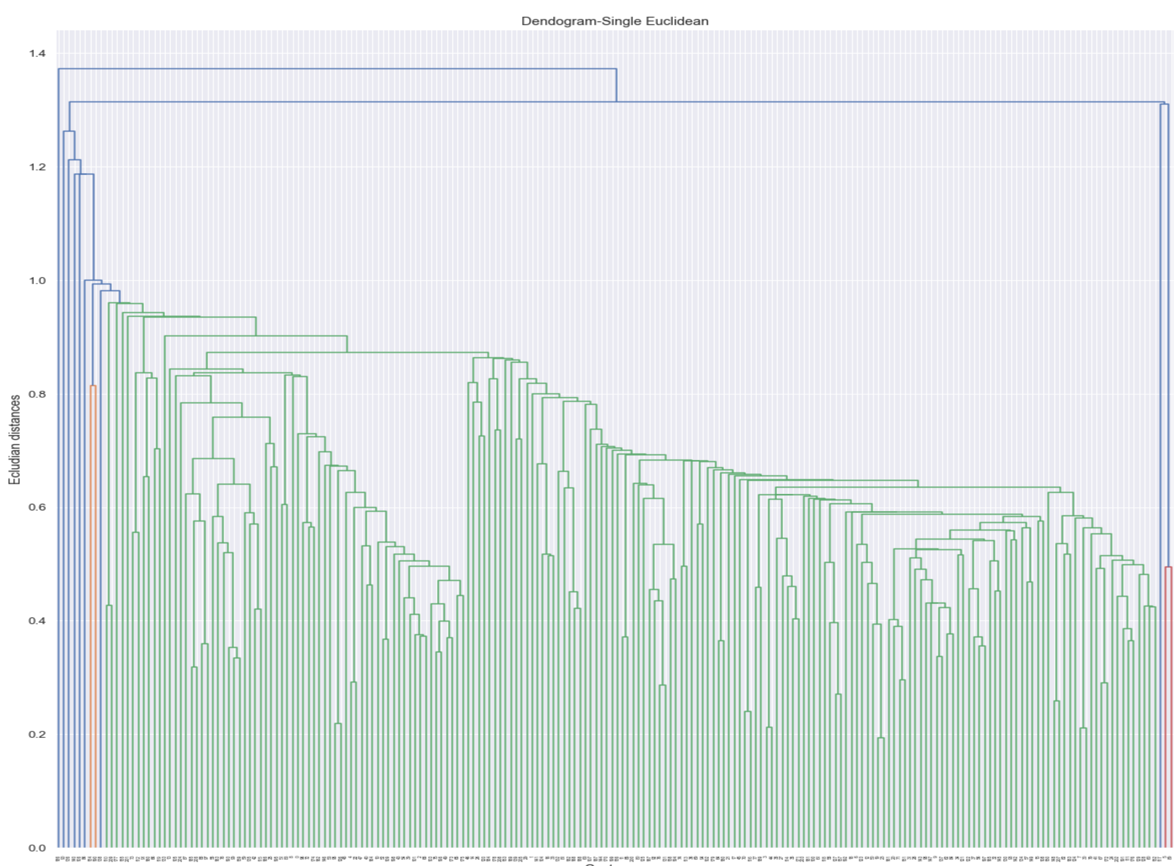
```
(210, 7)
```

Transforming scaled data array back to pandas data frame

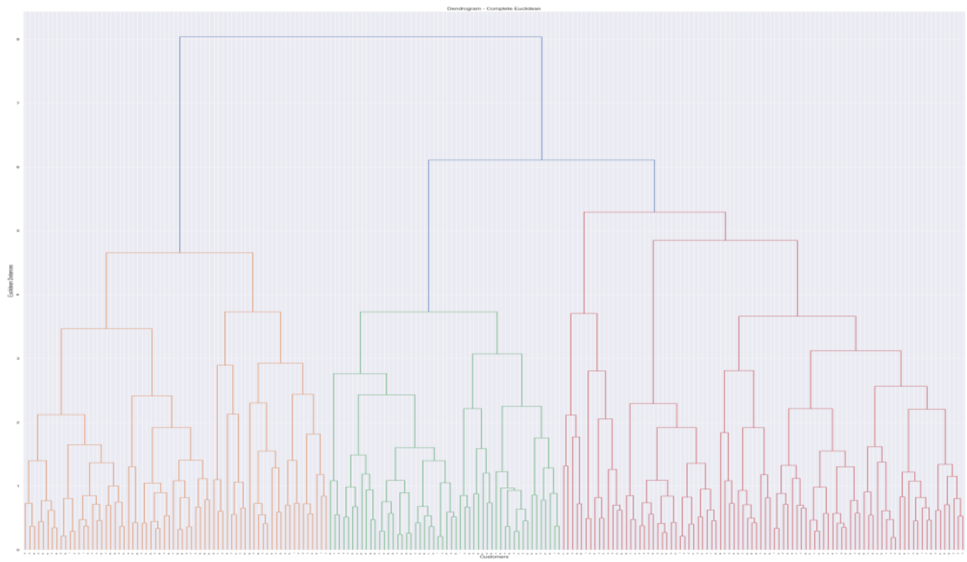
	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
0	1.754355	1.811968	0.178230	2.367533	1.338579	-0.298806	2.328998
1	0.393582	0.253840	1.501773	-0.600744	0.858236	-0.242805	-0.538582
2	1.413300	1.428192	0.504874	1.401485	1.317348	-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

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

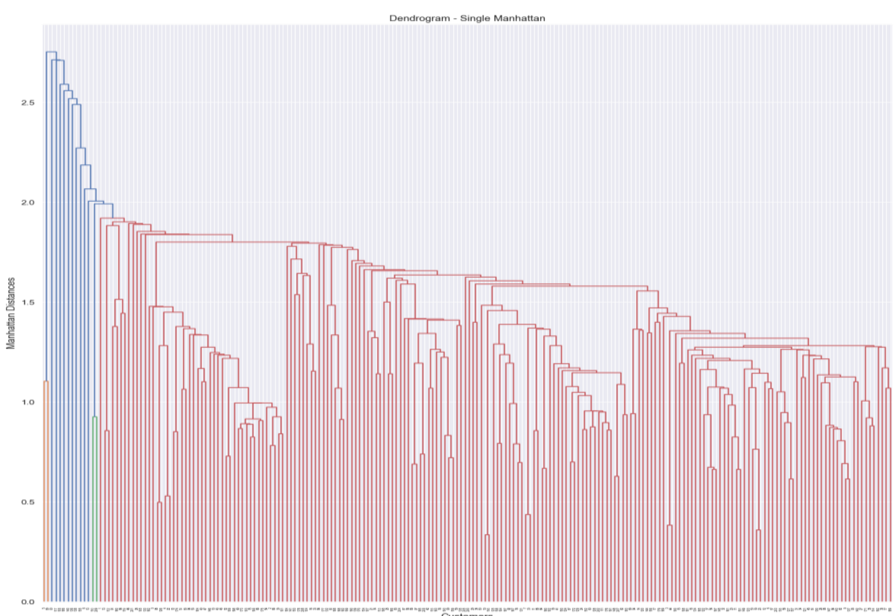
Dendrogram method='single' , "metrics Euclidean", Ecludian distances



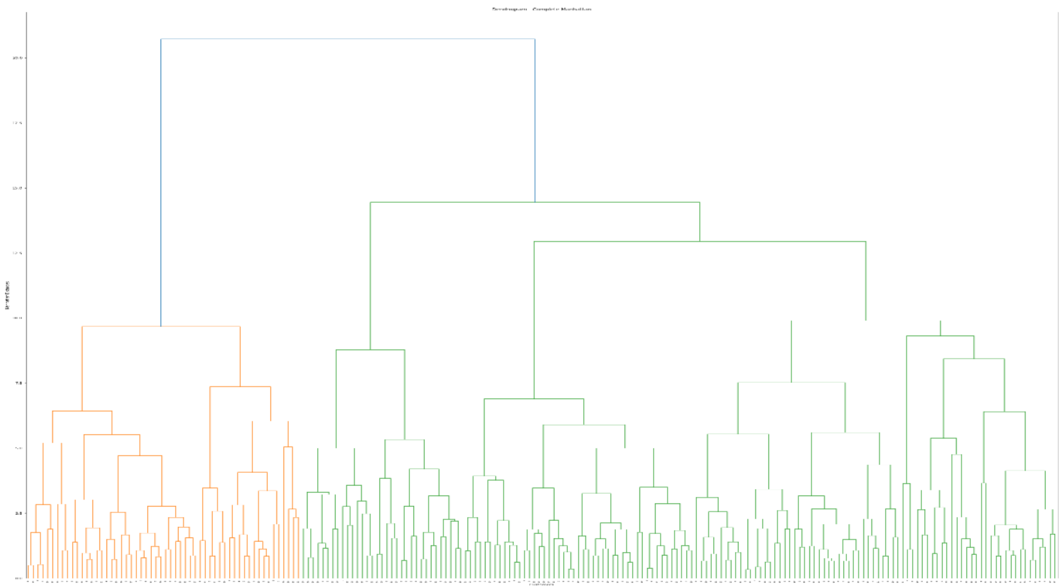
method='complete', metric='euclidean', Euclidean Distances



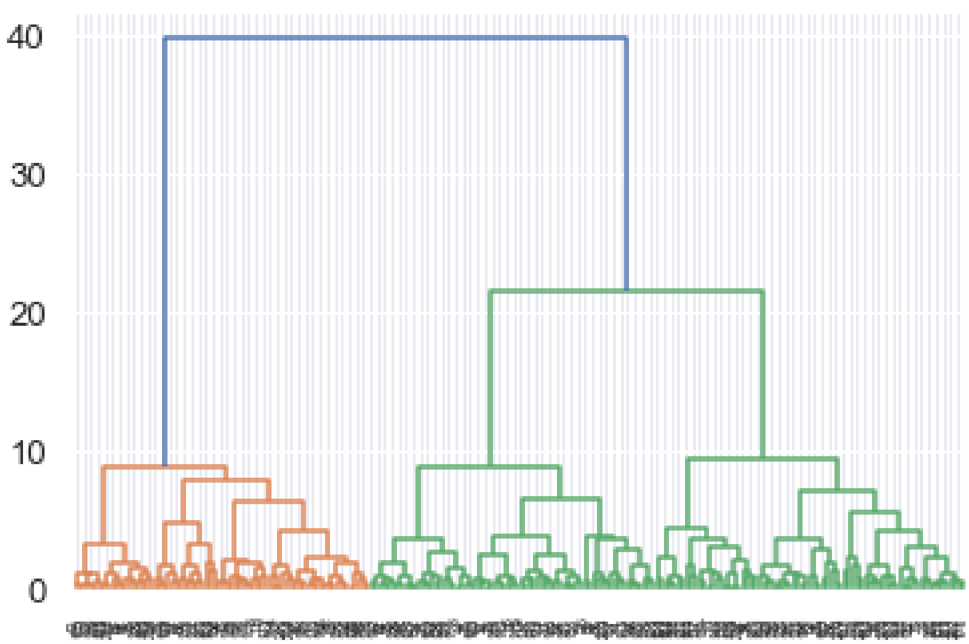
method='single', metric='cityblock', Manhattan Distances



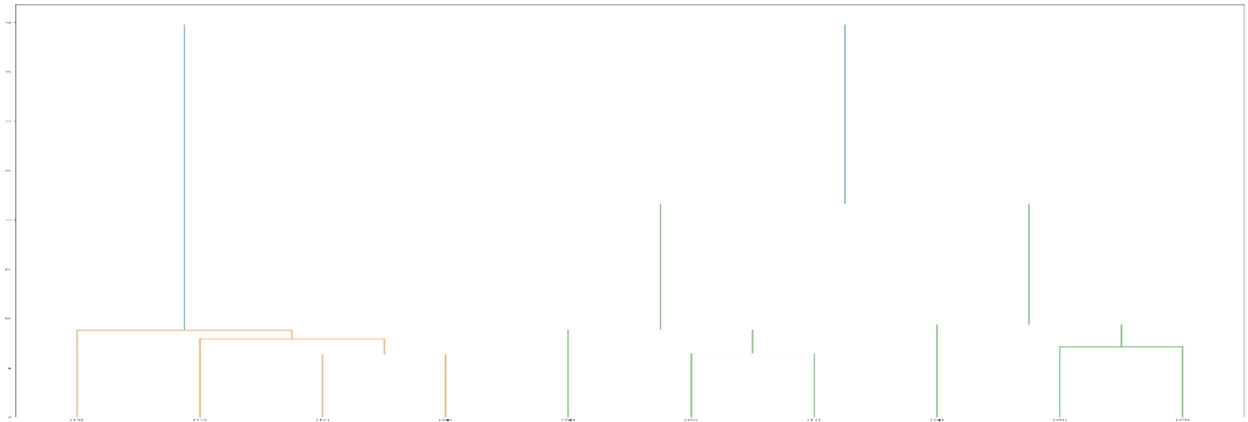
method='complete', metric='cityblock, Manhattan Distances



dendrogram(wardlink), method='ward'



dendrogram(wardlink, truncate_mode='lastp', p=10)



clusters are in array :

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

Adding the cluster profiles to the original dataset

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

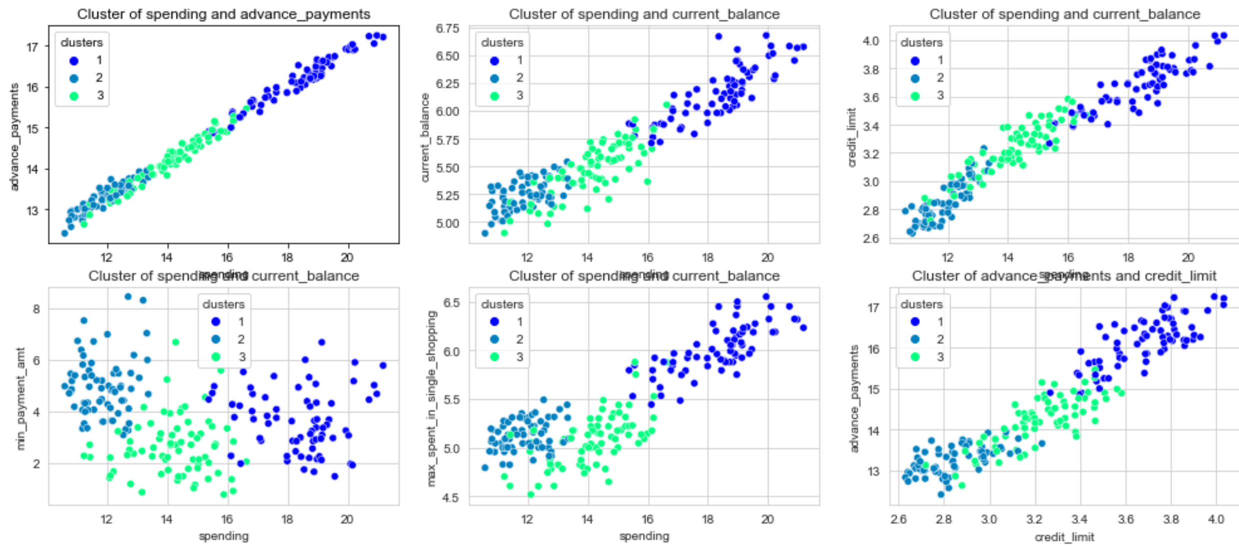
check the Cluster frequency

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

#Cluster profile

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

Hierarchical Clusters Scatterplot



cluster 1: Gold cluster-
SPENDING IS HIGH VALUE , WORTH of advance payment, current balance, min_payament _amount, max_spent in single shopping , advance payment.

cluster 2: silver cluster-

average spending on WORTH of advance payment, current balance, min_payament _amount, max_spent in single shopping, advance payment.

cluster 0: bronze CLuster-

very less value- WORTH of advance payment, current balance, min_payament _amount, max_spent in single shopping , advance payment.

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.¶

forming 3 clusters with K = 3

```
KMeans(n_clusters=3)
```

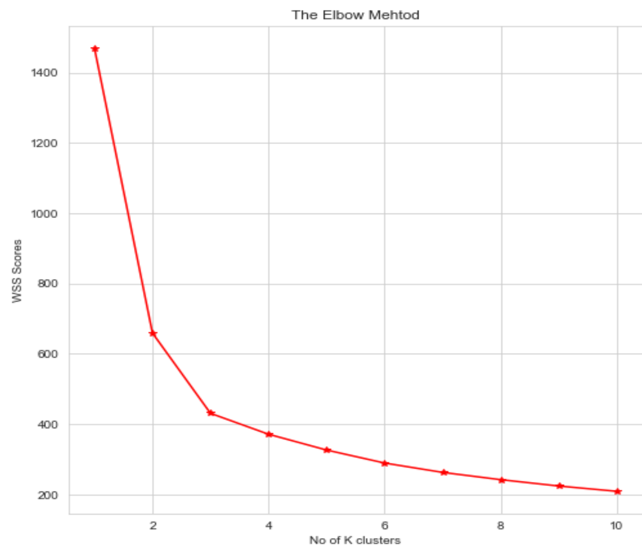
```
array([1, 0, 1, 2, 1, 2, 2, 0, 1, 2, 1, 0, 2, 1, 0, 2, 0, 2, 2, 2, 2, 2,
       1, 2, 0, 1, 0, 2, 2, 2, 0, 2, 2, 0, 2, 2, 2, 2, 2, 1, 1, 0, 1, 1,
       2, 2, 0, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 2, 2, 1, 0, 2, 2, 0, 0, 1,
       1, 0, 1, 2, 0, 2, 1, 1, 2, 1, 0, 2, 1, 0, 0, 0, 0, 1, 2, 0, 1, 0,
    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

WSS scores for K-Mean cluster 1 to 10 is

```
[1469.9999999999995,
 659.1717544870411,
 430.65897315130064,
 371.18461253510196,
 326.228916829726 6,
 289.203968672384,
 262.5643765876243,
 241.88830098980065,
 223.84263561564586,
 208.78789592737888]
```

WSS scores keep reducing as we increase the number of clusters.

Checking with Elbow Method



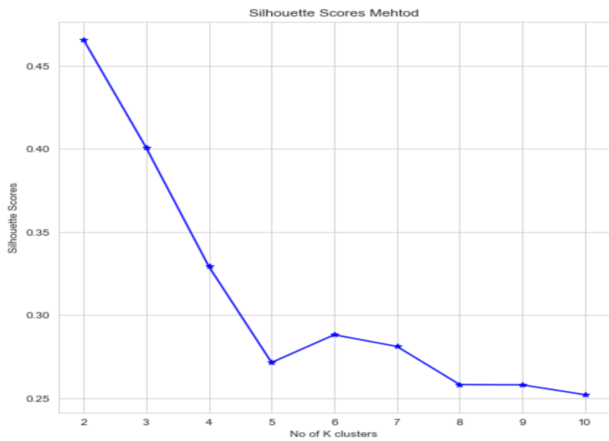
Cluster evaluation for 3 clusters:

The Silhouette score for 3 cluster is 0.40072705527512986

Silhouette score for other clusters are :

```
i 2 0.46577247686580914
i 3 0.40072705527512986
i 4 0.329196679201761З
i 5 0.27140135891439404
i 6 0.288094613574792 8
i 7 0.2810662000732231 4
i 8 0.2581529969450029 3
i 9 0.2579506855914407 4
i 10 0.252012223881712
```

Elbow method with Silhouette score vs No of K-clusters



Silhouette score is the best for 3 clusters hence we will go with 3 cluster profiling for this dataset, which is 0.40072705527512986 score.

Adding the cluster profiles to the original dataset

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

Cluster frequency

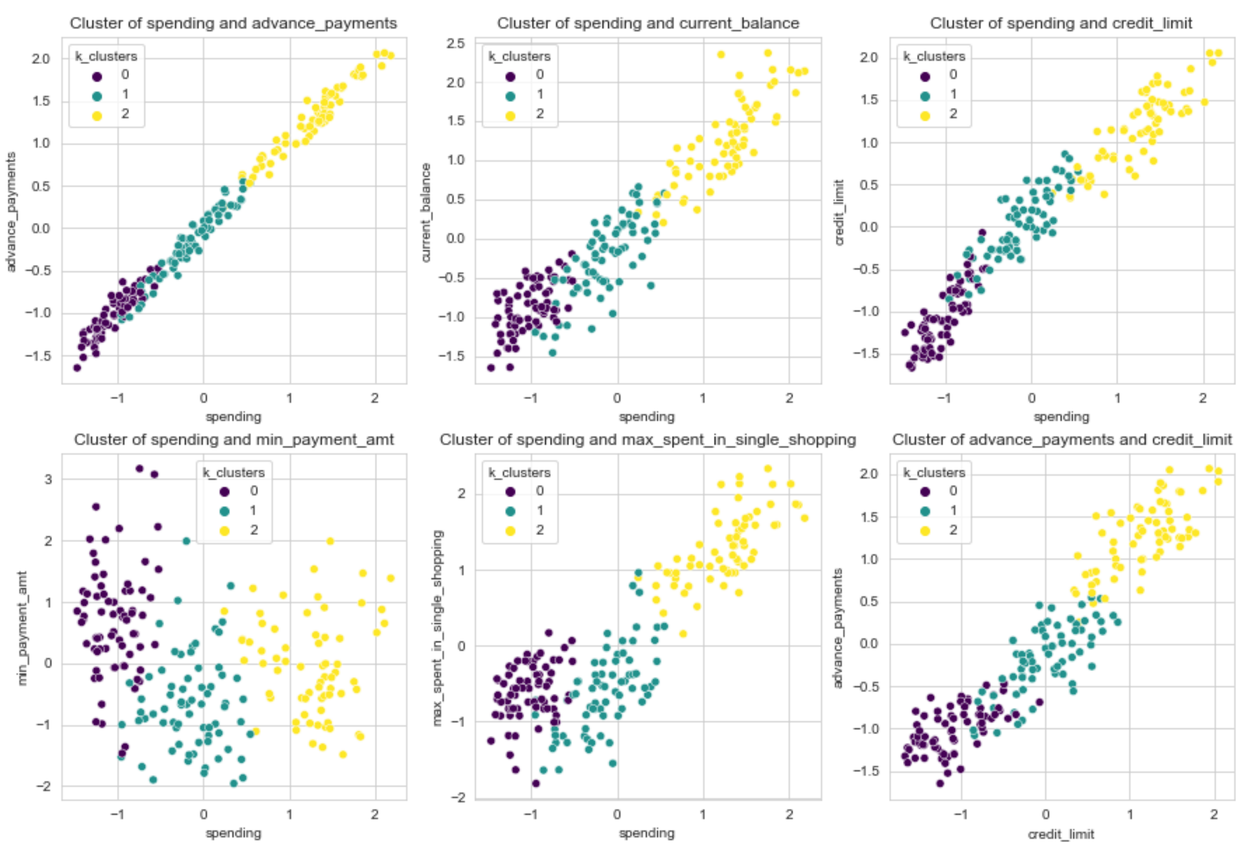
```
0    72
1    71
2    67
Name: k_clusters, dtype: int64
```

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

Cluster profiles¶

k_clusters	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	frequency
0	-1.030253	-1.006649	-0.964905	-0.897685	-1.085583	0.694804	-0.624809	72
1	-0.141119	-0.170043	0.449606	-0.257814	0.001647	-0.661919	-0.585893	71
2	1.256682	1.261966	0.560464	1.237883	1.164852	-0.045219	1.292308	67

K-means Clusters Scatterplot



Business insights based on Cluster profiles:

When we look at final cluster merge at original data set and take average value of the variable . below is the recommendation of each cluster profile.

cluster 2:Platinum cluster

cluster 1: Gold cluster

cluster 0: Silver Cluster

Customers under cluster 2 have a high spending, current balance, credit_limit and max_spent_in_single_shopping which clearly shows that they are premium high-net worth customers who make expensive purchases on their credit cards.

Customers under cluster 1 have a relatively lesser spending, current balance, credit_limit and max_spent_in_single_shopping which indicate that they are upper middle-class customers. The bank can provide promotional offers to this segment such that they increase their spending and are potential customers who can move into premium segments.

Customers under cluster 0(3rd cluster) have the least spending and credit_limits compared to other clusters. This signifies that they are customers who have recently bought credit cards or youths who have started working recently. Bank can provide customized offers to this segment to promote more spending on credit cards.

CART-RF-ANN

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

#Checking the data sample

	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

Shape of dataset:

```
(3000, 10)
```

```
Information of dataset:
<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
```

Obesrvation

-no missing value present, 3000 records present -10 variables -Age, Commision, Duration, Sales are numeric variable

- rest are categorial variables
- 9 independent variable and one target variable – Claimed.

Checking the datatypes:

```
Age              int64
Agency_Code     object
Type            object
Claimed         object
Commision      float64
Channel         object
Duration         int64
Sales          float64
Product Name    object
Destination     object
dtype: object
```

#Null value check

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

#|Observation

No missing value present in dataset

#Descriptive Statistics Summary

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

#Observation

- duration has negative valu, it is not possible. Wrong entry.
- Commision & Sales- mean and median varies signficantly

#Descriptive Statistics Summary including all:

	Age	Agency_Code	Type	Claimed	Commision	Channel	Duration	Sales	Product Name	Destination
count	3000.000000	3000	3000	3000	3000.000000	3000	3000.000000	3000.000000	3000	3000
unique	NaN	4	2	2	NaN	2	NaN	NaN	5	3
top	NaN	EPX	Travel Agency	No	NaN	Online	NaN	NaN	Customised Plan	ASIA
freq	NaN	1365	1837	2076	NaN	2954	NaN	NaN	1136	2465
mean	38.091000	NaN	NaN	NaN	14.529203	NaN	70.001333	60.249913	NaN	NaN
std	10.463518	NaN	NaN	NaN	25.481455	NaN	134.053313	70.733954	NaN	NaN
min	8.000000	NaN	NaN	NaN	0.000000	NaN	-1.000000	0.000000	NaN	NaN
25%	32.000000	NaN	NaN	NaN	0.000000	NaN	11.000000	20.000000	NaN	NaN
50%	36.000000	NaN	NaN	NaN	4.630000	NaN	26.500000	33.000000	NaN	NaN
75%	42.000000	NaN	NaN	NaN	17.235000	NaN	63.000000	69.000000	NaN	NaN
max	84.000000	NaN	NaN	NaN	210.210000	NaN	4580.000000	539.000000	NaN	NaN

#Observation

Categorial code variable maximum unique count is 5

	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
5	45	JZI	Airlines	Yes	15.75	Online	8	45.00	Bronze Plan	ASIA
6	61	CWT	Travel Agency	No	35.64	Online	30	59.40	Customised Plan	Americas
7	36	EPX	Travel Agency	No	0.00	Online	16	80.00	Cancellation Plan	ASIA
8	36	EPX	Travel Agency	No	0.00	Online	19	14.00	Cancellation Plan	ASIA
9	36	EPX	Travel Agency	No	0.00	Online	42	43.00	Cancellation Plan	ASIA

#Observation

- Data looks good at first glance

	Age	Agency_Code	Type	Claimed	Commision	Channel	Duration	Sales	Product Name	Destination
2990	51	EPX	Travel Agency	No	0.00	Online	2	20.00	Customised Plan	ASIA
2991	29	C2B	Airlines	Yes	48.30	Online	381	193.20	Silver Plan	ASIA
2992	28	CWT	Travel Agency	No	11.88	Online	389	19.80	Customised Plan	ASIA
2993	36	EPX	Travel Agency	No	0.00	Online	234	10.00	Cancellation Plan	ASIA
2994	27	C2B	Airlines	Yes	71.85	Online	416	287.40	Gold Plan	ASIA
2995	28	CWT	Travel Agency	Yes	166.53	Online	364	256.20	Gold Plan	Americas
2996	35	C2B	Airlines	No	13.50	Online	5	54.00	Gold Plan	ASIA
2997	36	EPX	Travel Agency	No	0.00	Online	54	28.00	Customised Plan	ASIA
2998	34	C2B	Airlines	Yes	7.64	Online	39	30.55	Bronze Plan	ASIA
2999	47	JZI	Airlines	No	11.55	Online	15	33.00	Bronze Plan	ASIA

#Observation

- Data looks good at first glance

#Geting unique counts of all Nominal Variables

```
AGENCY_CODE : 4
JZI     239
CWT     472
C2B     924
EPX    1365
Name: Agency_Code, dtype: int64


TYPE : 2
Airlines         1163
Travel Agency    1837
Name: Type, dtype: int64


CLAIMED : 2
Yes     924
No     2076
Name: Claimed, dtype: int64


CHANNEL : 2
Offline      46
Online     2954
Name: Channel, dtype: int64


PRODUCT NAME : 5
Gold Plan             109
Silver Plan           427
Bronze Plan           650
Cancellation Plan     678
Customised Plan      1136
Name: Product Name, dtype: int64


DESTINATION : 3
EUROPE      215
Americas    320
ASIA       2465
Name: Destination, dtype: int64
```

#Check for duplicate data

```
Number of duplicate rows = 139
```

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

139 rows × 10 columns

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

Univariate Analysis

Age variable

#central Value

```
Range of values : 76
Minimum Age:  8
Maximum Age:  84
Mean value:  38.091
Median value:  36.0
Standard deviation:  10.463518245377944
Null values:  False
```

#Quartiles

```
spending - 1st Quartile (Q1) is:  32.0
spending - 3st Quartile (Q3) is:  42.0
Interquartile range (IQR) of Age is  10.0
```

#Outlier detection from Interquartile range (IQR) in original data

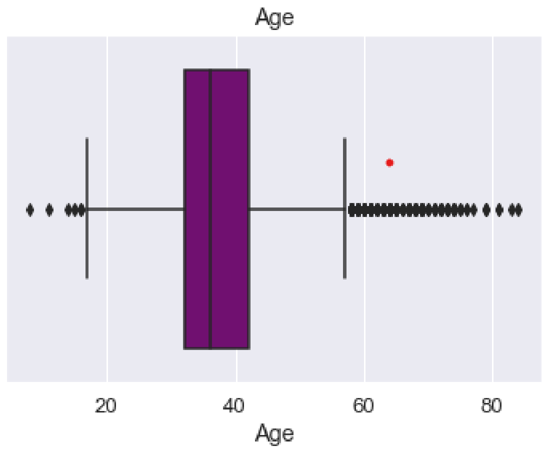
```
Lower outliers in Age:  17.0
Upper outliers in Age:  57.0
```

```
Number of outlier in Age Upper: 198
Number of outlier in Age Lower: 6
```

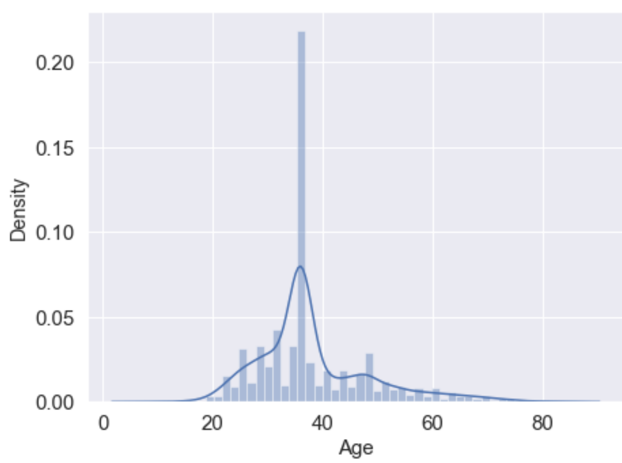

```
Number of outlier in Age Upper: 198
Number of outlier in Age Lower: 6
```

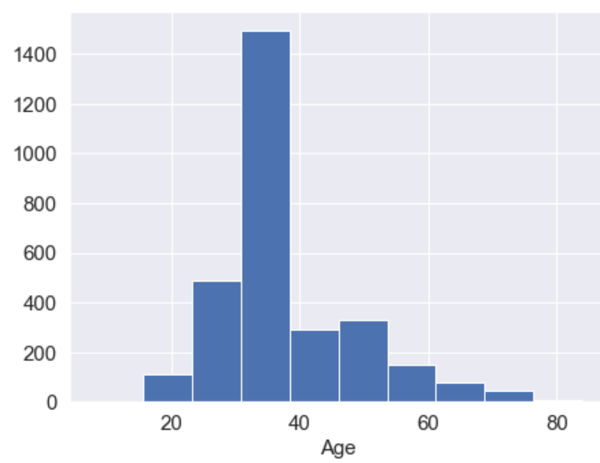
Box Plot for Age :

```
Text(0.5, 1.0, 'Age')
```



DistPlot and histogram





Commission variable

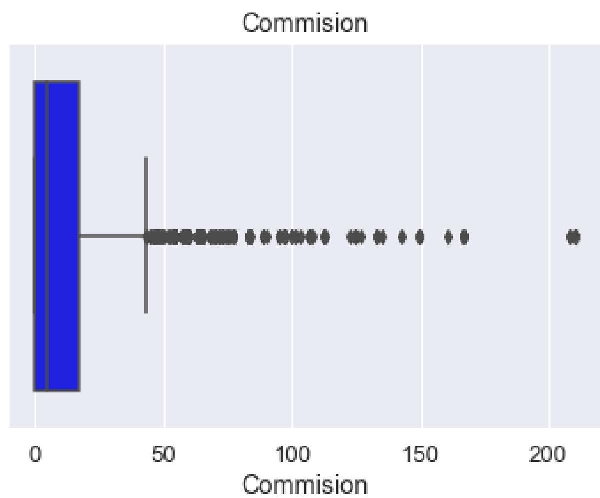
#Central values Checking :

```
Range of value : 210.21

Minimum Commision:  0.0
Maximum Commision:  210.21
Mean value:  14.529203333333266
Median value:  4.63
Standard deviation:  25.48145450662553
Null values:  False

#Quartiles
Commision - 1st Quartile (Q1) is:  0.0
Commision - 3st Quartile (Q3) is:  17.235
Interquartile range (IQR) of Commision is  17.235
```

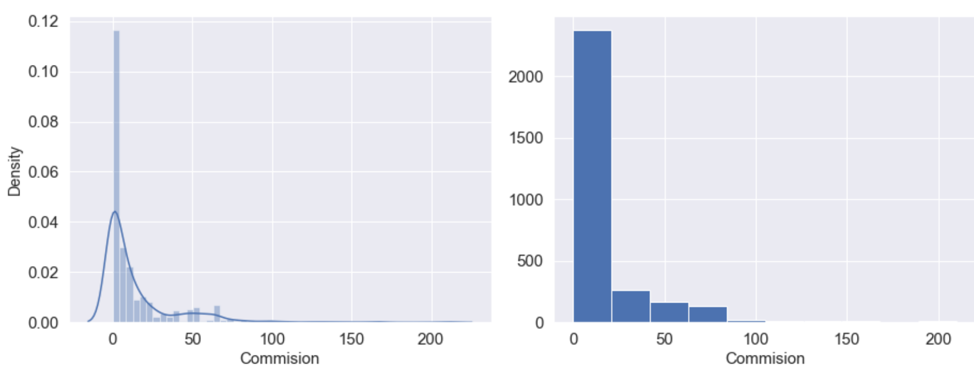
#Outlier detection from Interquartile range (IQR) in original data :

```
lower outlier in commision is -25.8525
upper outlier in commision is 43.0875


Number of outliers in Commision upper :  362
Number of outliers in Commision lower :  0
% of Outlier in Commision upper:  12 %
% of Outlier in Commision lower:  0 %
```

BOX plot to check outlier:



#distplot and Histogram:



#Duration variable

```
Range of values:  4581
Minimum Duration:  -1
Maximum Duration:  4580
Mean value:  70.00133333333333
Median value:  26.5
```

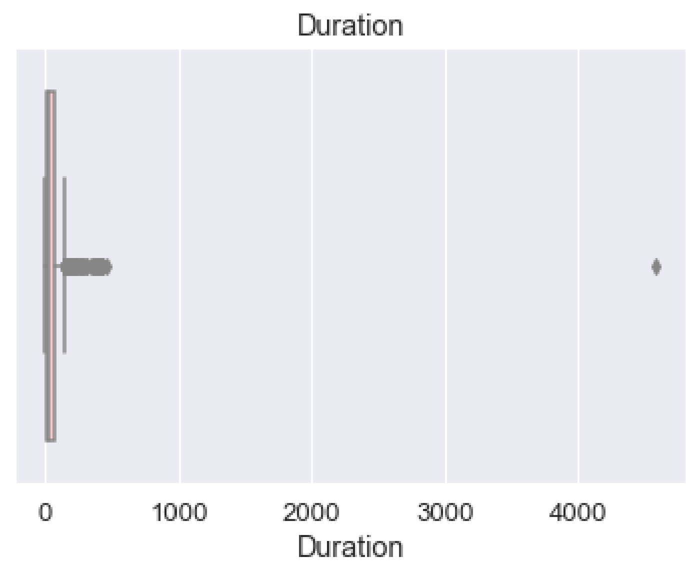
```
Standard deviation:  134.05331313253495
Null values:  False
```

#Quartiles

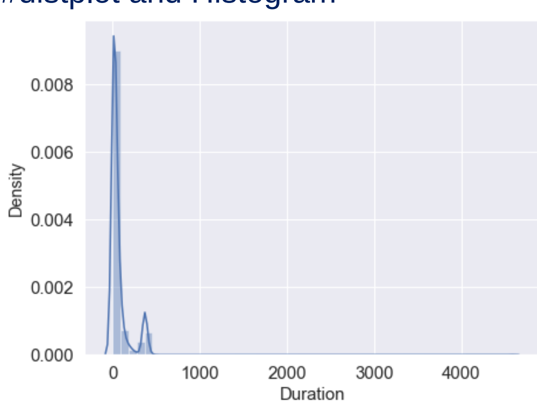
```
Duration - 1st Quartile (Q1) is:  11.0
Duration - 3st Quartile (Q3) is:  63.0
Interquartile range (IQR) of Duration is  52.0
```

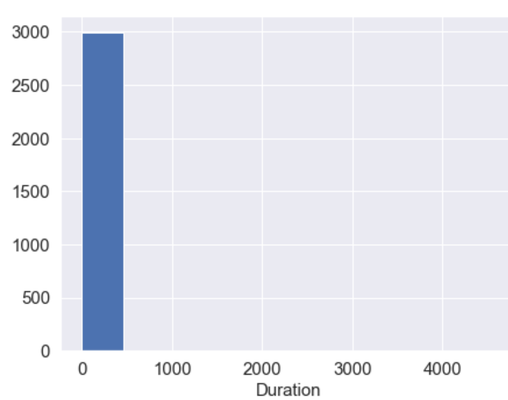
#Outlier detection from Interquartile range (IQR) in original data

```
Lower outliers in Duration:  -67.0
Upper outliers in Duration:  141.0


Number of outliers in Duration upper :  382
Number of outliers in Duration lower :  0
% of Outlier in Duration upper:  13 %
% of Outlier in Duration lower:  0 %
```



#distplot and Histogram





#Sales variable

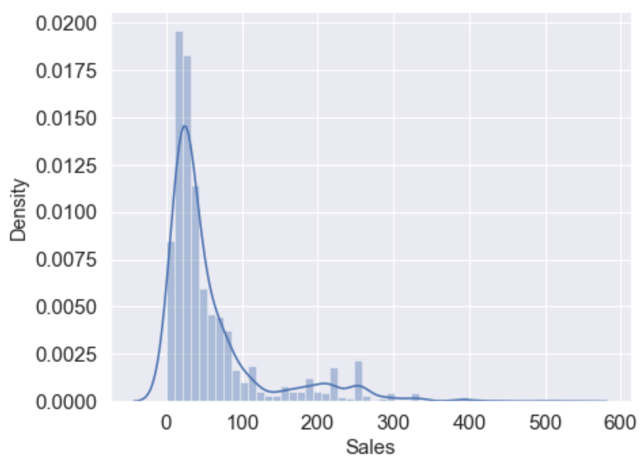
#checking central values:

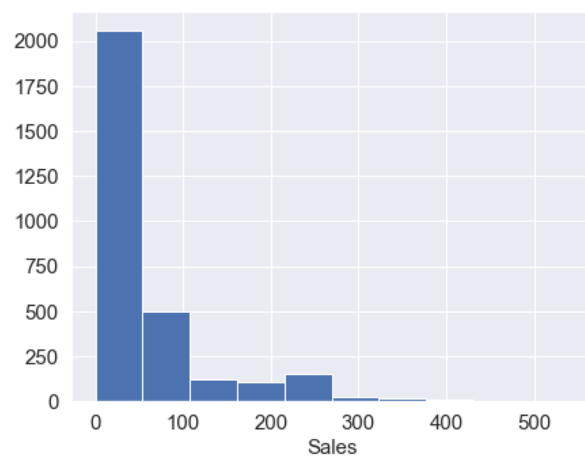
```
Range of values:  539.0
Minimum Sales:  0.0
Maximum Sales:  539.0
Mean value:  60.24991333333344
Median value:  33.0
Standard deviation:  70.73395353143047
Null values:  False
```

#Quartiles

```
Sales - 1st Quartile (Q1) is:  20.0
Sales - 3st Quartile (Q3) is:  69.0
Interquartile range (IQR) of Sales is  49.0

#Outlier detection from Interquartile range (IQR) in original data
Lower outliers in Sales:  -53.5
Upper outliers in Sales:  142.5


Lower outliers in Sales:  -53.5
Upper outliers in Sales:  142.5
```

#distplot and histogram





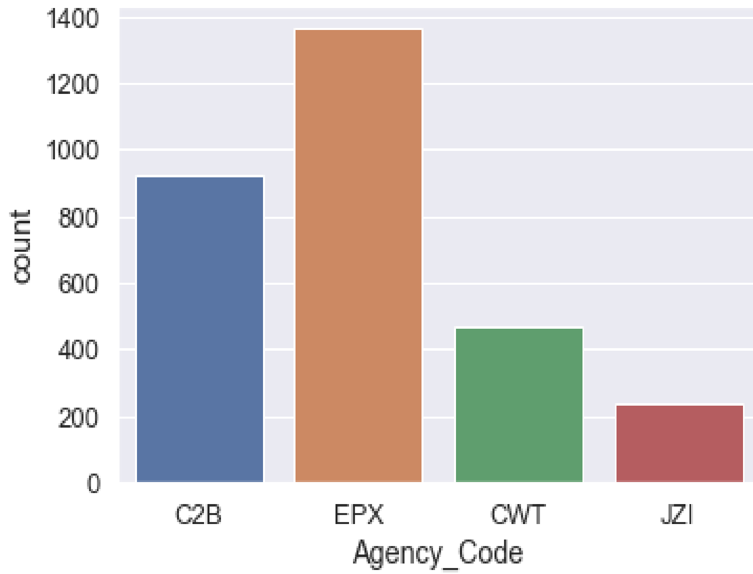
There are outliers in all the variables, but the sales and commission can be a genius business value. Random Forest and CART can handle the outliers. Hence, Outliers are not treated for now, we will keep the data as it is.

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

Categorical Variables¶

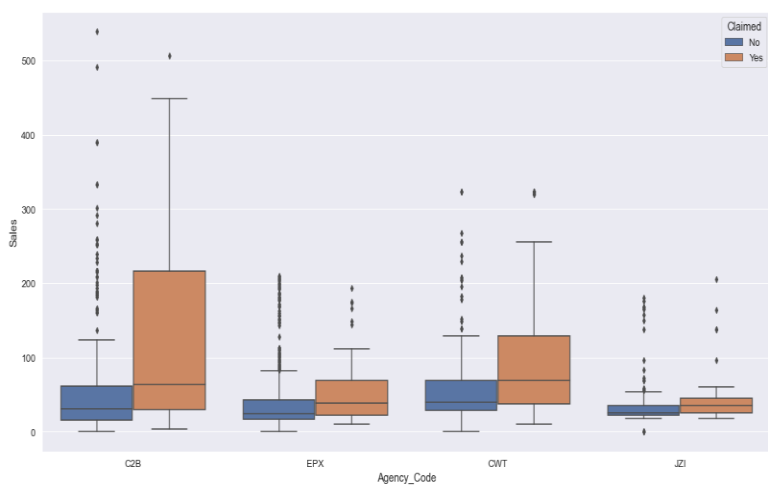
Agency_Code:

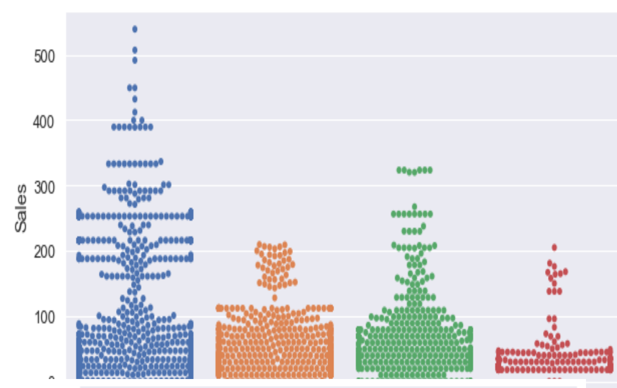
#Count Plot-EPIX code is large and CWT code count is smaller.

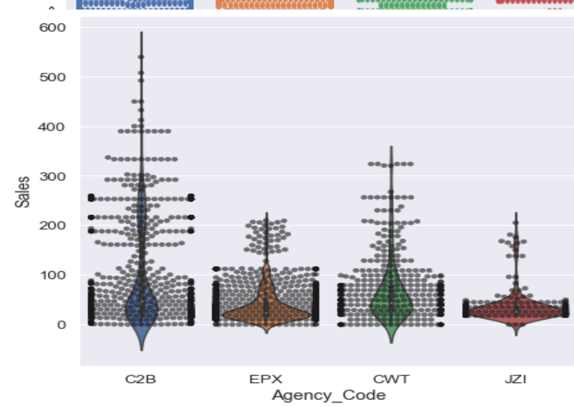


Swarmpot

#Boxplot

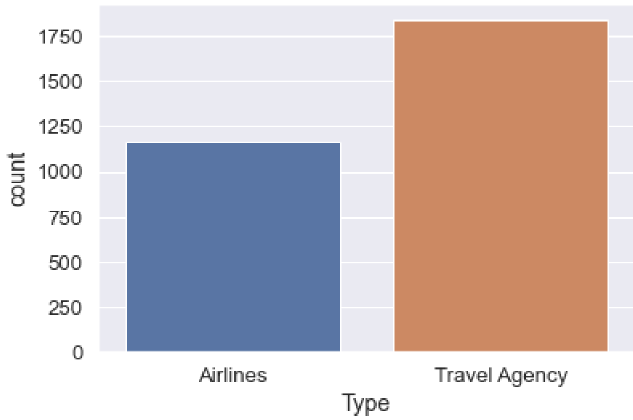




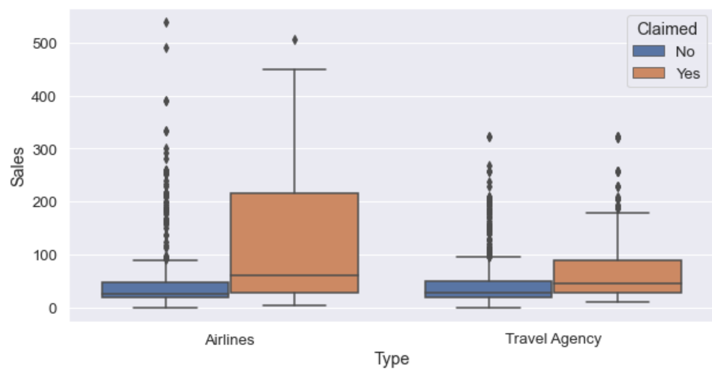


#Combine Violin plot and Swarmp plot

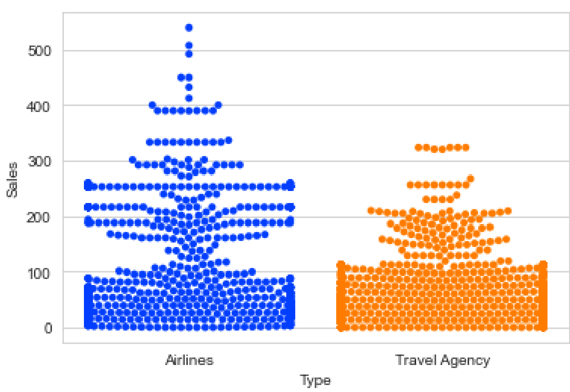
#TYPE: Travel Agency has more count than Airline.



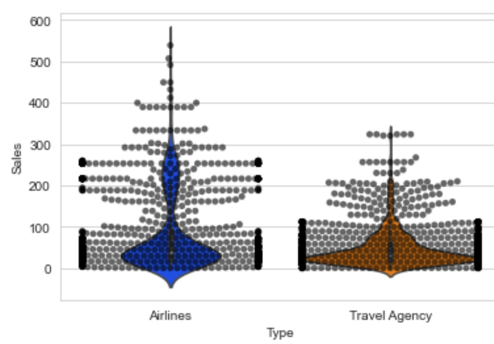
#BOX PLOT



SWARM PLOT

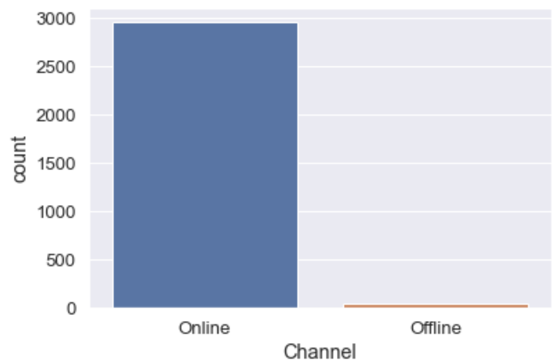


Combine Violin plot and Swarm plot

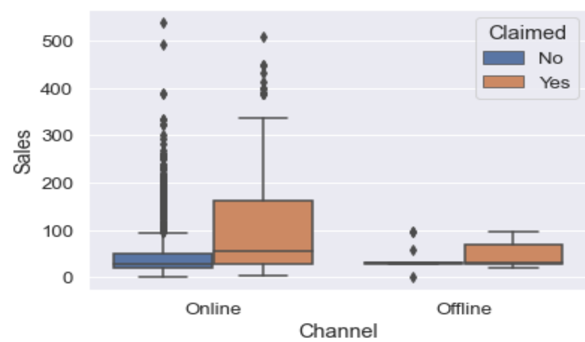


Channel

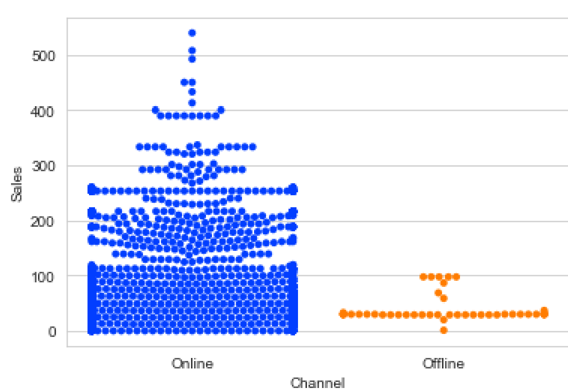
#Count Plot: the offline customers are very less



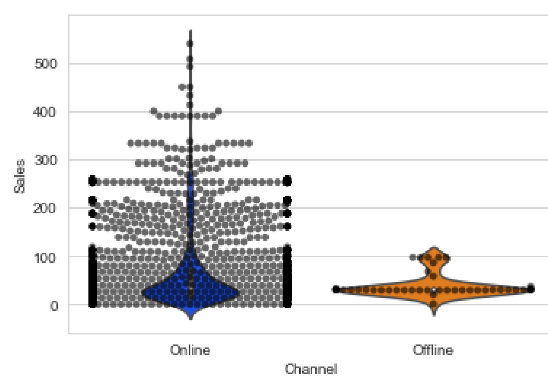
#BOX PLOT



#SWARM PLOT

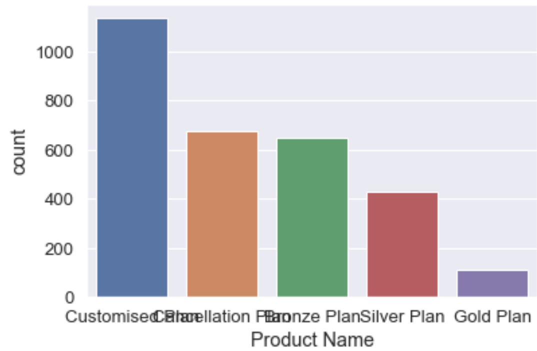


Combine Violin plot and Swarmplot

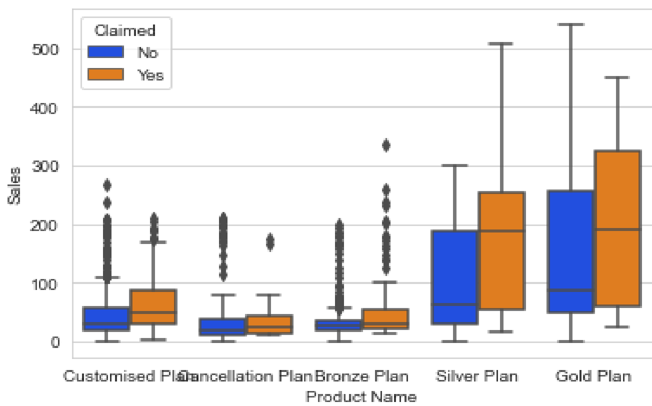


Product Name

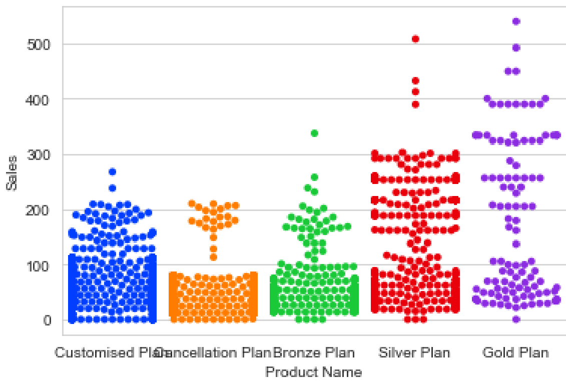
Count Plot:

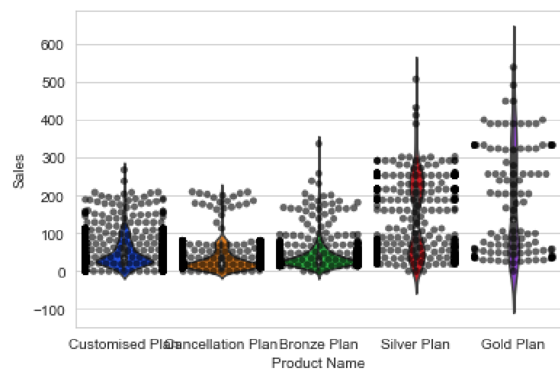


#BOX PLOT



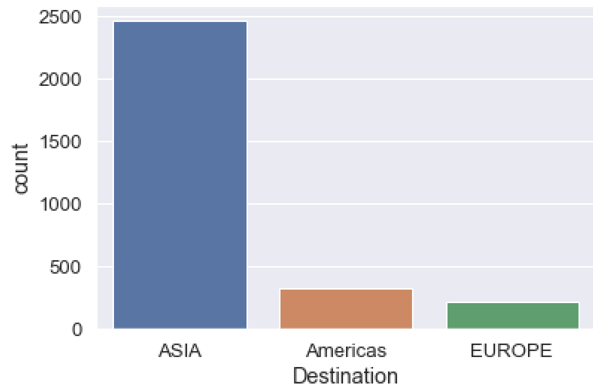
SWARM PLOT



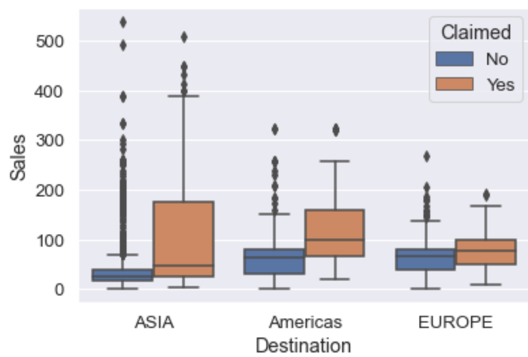


Destination

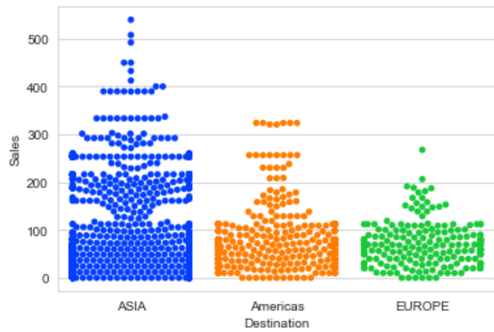
Count Plot

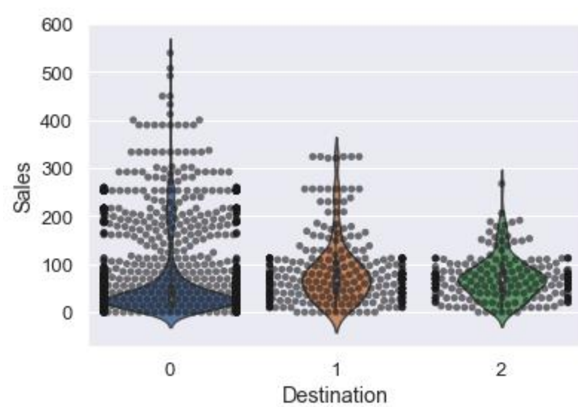


#BOX PLOT

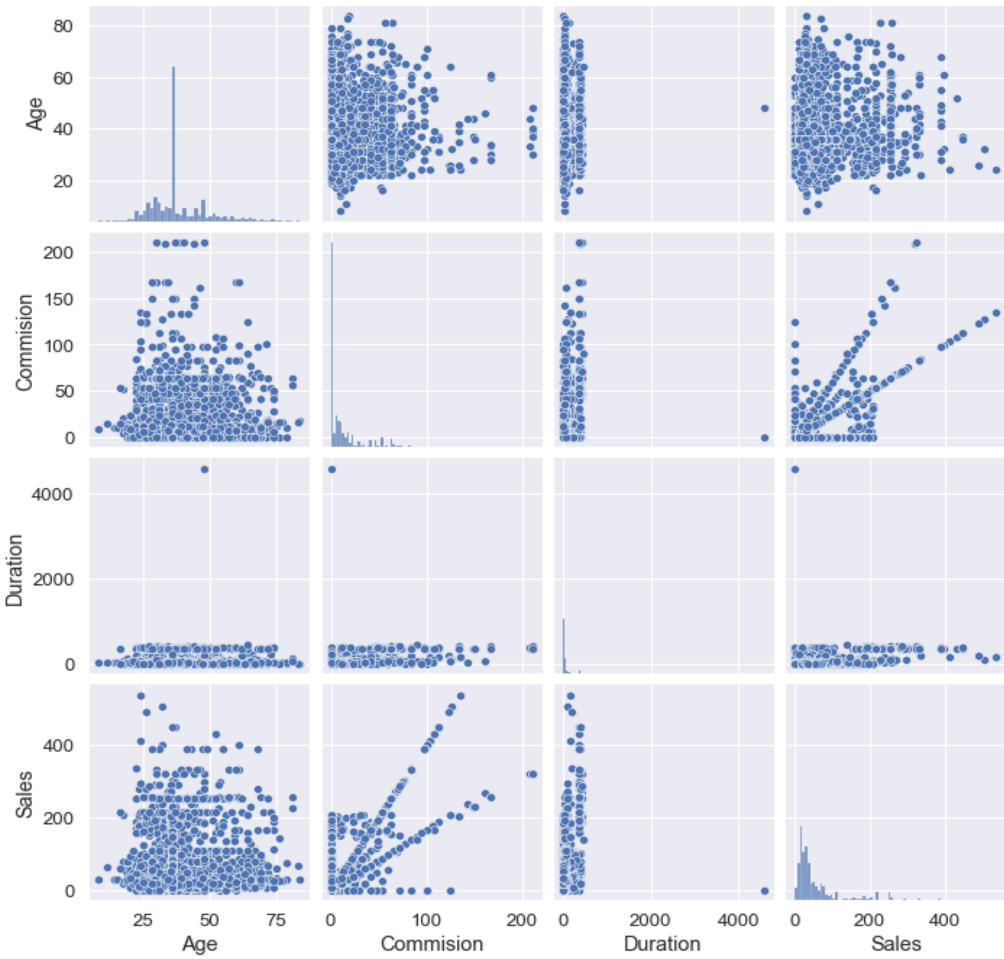


SWARM PLOT

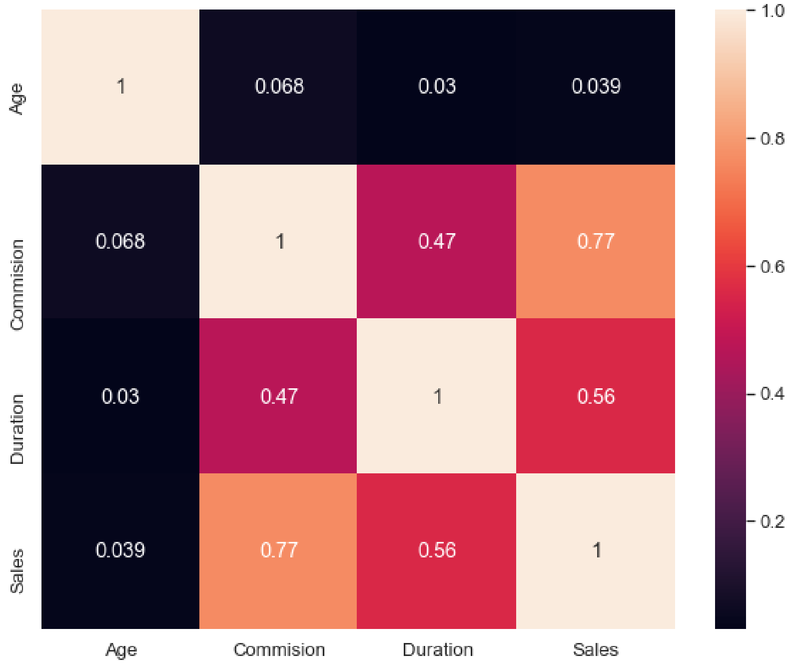




#Checking pairwise distribution of the continuous variables



#Checking for Correlations



#Converting all objects to categorical codes

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


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


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


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


feature: Product Name
['Customised Plan', 'Cancellation Plan', 'Bronze Plan', 'Silver Plan', 'Go
ld Plan']
Categories (5, object): ['Bronze Plan', 'Cancellation Plan', 'Customised P
lan', 'Gold Plan', 'Silver Plan']
```

```
[2 1 0 4 3]


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

#Checking information:

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

check some sample data:

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

#Proportion of 1s and 0s

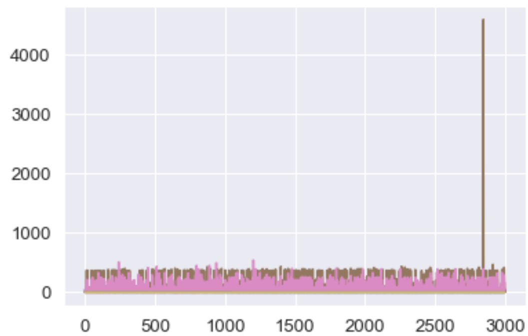
```
0    0.692
1    0.308
Name: Claimed, dtype: float64
```

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

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

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

prior to scaling



Scaling the attributes.

	Age	Agency_Code	Type	Commision	Channel	Duration	Sales	Product Name	Destination
0	0.947162	-1.314358	-1.256796	-0.542807	0.124788	-0.470051	-0.816433	0.268835	-0.434646
1	-0.199870	0.697928	0.795674	-0.570282	0.124788	-0.268605	-0.569127	0.268835	-0.434646
2	0.086888	-0.308215	0.795674	-0.337133	0.124788	-0.499894	-0.711940	0.268835	1.303937
3	-0.199870	0.697928	0.795674	-0.570282	0.124788	-0.492433	-0.484288	-0.525751	-0.434646
4	-0.486629	1.704071	-1.256796	-0.323003	0.124788	-0.126846	-0.597407	-1.320338	-0.434646

#Splitting data into training and test set

#Checking the dimensions of the training and test data

```
X_train (2100, 9)
X_test (900, 9)
train_lables (2100,)
test_labels (900,)
```

#Building a Decision Tree Classifier

grid search

```
{'criterion': 'gini', 'max_depth': 4.85, 'min_samples_leaf': 44, 'min_samples
_split': 260}
```

Out[229]:

```
DecisionTreeClassifier(max_depth=4.85, min_samples_leaf=44,
                       min_samples_split=260, random_state=1)
```

Generating Tree
#Variable Importance – DTCL

	Imp
Agency_Code	0.634112
Sales	0.220899
Product Name	0.086632
Commision	0.021881
Age	0.019940
Duration	0.016536
Type	0.000000
Channel	0.000000
Destination	0.000000

#Predicting on Training and Test dataset¶

#Getting the Predicted Classes and Probs

	0	1
0	0.697947	0.302053
1	0.979452	0.020548
2	0.921171	0.078829
3	0.510417	0.489583
4	0.921171	0.078829

Building a Random Forest Classifier

param_grid_rfcl = { 'max_depth': [5,10,15],#20,30,40 'max_features': [4,5,6,7],## 7,8,9 'min_samples_leaf': [10,50,70],## 50,100 'min_samples_split': [30,50,70], ## 60,70 'estimators': [200, 250,300] ## 100,200 }

rfcl = RandomForestClassifier(random_state=1)

grid_search_rfcl = GridSearchCV(estimator = rfcl, param_grid = param_grid_rfcl, cv = 5)

grid_search_rfcl. Fit(X_train, train labels) print(grid_search_rfcl.bestparams) best_grid_rfcl = grid_search_rfcl.bestestimator best_grid_rfcl

```
Best grid_rfcl:
```

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

RandomForestClassifier(max_depth=6, max_features=3, min_samples_leaf=8,
                       min_samples_split=46, n_estimators=350, random_state=1
)
```

#Predicting the Training and Testing data¶

#Getting the Predicted Classes and Probs

	0	1
0	0.778010	0.221990
1	0.971910	0.028090
2	0.904401	0.095599
3	0.651398	0.348602
4	0.868406	0.131594

Building a Neural Network Classifier¶

```
MLPClassifier(hidden_layer_sizes=200, max_iter=2500, random_state=1, tol=0
.01)
```

#Predicting the Training and Testing data

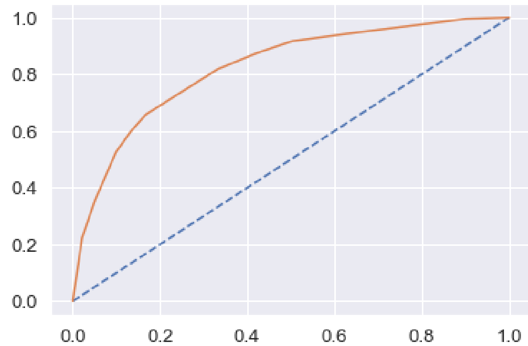
#Getting the Predicted Classes and Probs

	0	1
0	0.822676	0.177324
1	0.933407	0.066593
2	0.918772	0.081228
3	0.688933	0.311067
4	0.913425	0.086575

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

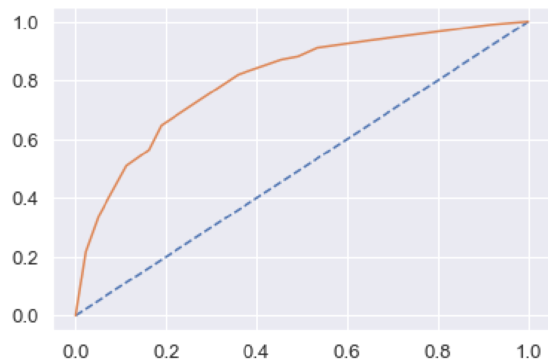
CART - AUC and ROC for the training data

#AUC: AUC: 0.823



#AUC and ROC for the test data:

AUC: 0.801



#CART Confusion Matrix and Classification Report for the training data

```
array([[1309,  144],
       [ 307,  340]], dtype=int64)
```

#Train Data Accuracy : 0.7852380952380953

check the classification report for train data

	precision	recall	f1-score	support
0	0.81	0.90	0.85	1453
1	0.70	0.53	0.60	647
accuracy			0.79	2100
macro avg	0.76	0.71	0.73	2100
weighted avg	0.78	0.79	0.78	2100

```
cart_train_precision  0.7
cart_train_recall  0.53
cart_train_f1  0.6
```

#CART Confusion Matrix and Classification Report for the testing data

```
array([[553,  70],
       [136, 141]], dtype=int64)
```

```
#Test Data Accuracy: 0.7711111111111111
```

	precision	recall	f1-score	support
0	0.80	0.89	0.84	623
1	0.67	0.51	0.58	277
accuracy			0.77	900
macro avg	0.74	0.70	0.71	900
weighted avg	0.76	0.77	0.76	900

```
cart_test_precision  0.67
cart_test_recall  0.51
cart_test_f1  0.58
```

Cart Conclusion

Train Data: - AUC: 82%

- Accuracy: 79%
- Precision: 70%
- f1-Score: 60%

Test Data: - AUC: 80%

- Accuracy: 77%
- Precision: 80%
- f1-Score: 84%

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

Change is the most important variable for predicting diabetes

RF Model Performance Evaluation on Training data

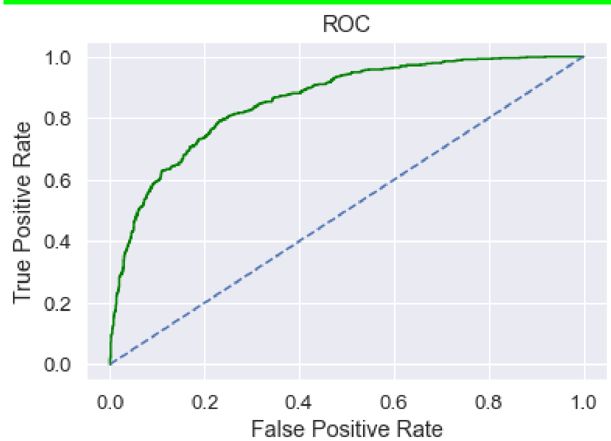
```
array([[1297,  156],
       [ 255,  392]], dtype=int64)
```

```
Accurancy: 0.8042857142857143
```

	precision	recall	f1-score	support
0	0.84	0.89	0.86	1453
1	0.72	0.61	0.66	647
accuracy			0.80	2100
macro avg	0.78	0.75	0.76	2100
weighted avg	0.80	0.80	0.80	2100

```
rf_train_precision  0.72
rf_train_recall  0.61
rf_train_f1  0.66
```

```
Area under Curve is 0.8563713512840778
```



RF Model Performance Evaluation on Test data

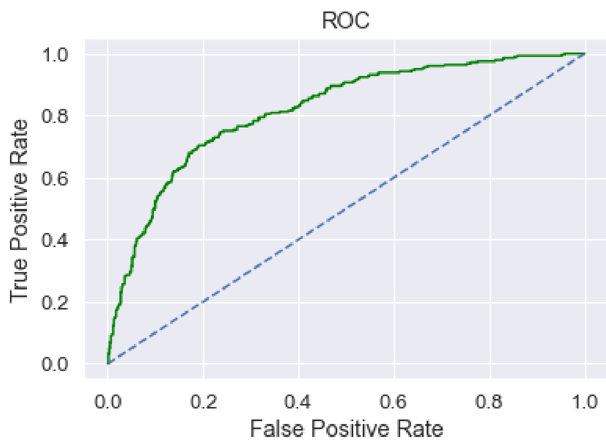
```
array([[550,  73],
       [121, 156]], dtype=int64)
```

```
Accuracy: 0.7844444444444445
```

	precision	recall	f1-score	support
0	0.82	0.88	0.85	623
1	0.68	0.56	0.62	277
accuracy			0.78	900
macro avg	0.75	0.72	0.73	900
weighted avg	0.78	0.78	0.78	900

```
rf_test_precision  0.68
rf_test_recall  0.56
rf_test_f1  0.62
```

```
Area under Curve is 0.8181994657271499
```



Random Forest Conclusion

Train Data:

- AUC: 86%
- Accuracy: 80%
- Precision: 72%
- f1-Score: 66%

`Test Data:`

- AUC: 82%
- Accuracy: 78%
- Precision: 68%
- f1-Score: 62%

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

Change is again the most important variable for predicting diabetes

NN Model Performance Evaluation on Training data

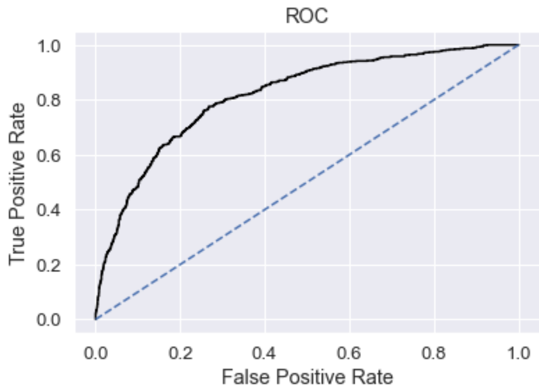
```
array([[1298,  155],
       [ 315,  332]], dtype=int64)
```

```
Accuracy: 0.7761904761904762
```

	precision	recall	f1-score	support
0	0.80	0.89	0.85	1453
1	0.68	0.51	0.59	647
accuracy			0.78	2100
macro avg	0.74	0.70	0.72	2100
weighted avg	0.77	0.78	0.77	2100

```
nn_train_precision  0.68
nn_train_recall  0.51
nn_train_f1  0.59
```

Area under Curve is 0.8166831721609928



NN Model Performance Evaluation on Test data¶¶

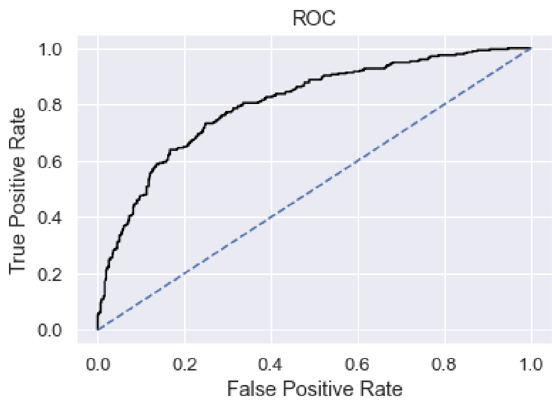
```
array([[553,  70],
       [138, 139]], dtype=int64)
```

Accuracy :0.7688888888888888

	precision	recall	f1-score	support
0	0.80	0.89	0.84	623
1	0.67	0.50	0.57	277
accuracy			0.77	900
macro avg	0.73	0.69	0.71	900
weighted avg	0.76	0.77	0.76	900

```
nn_test_precision  0.67
nn_test_recall  0.5
nn_test_f1  0.57
```

Area under Curve is 0.8044225275393896



Neural Network Conclusion

Train Data:

- AUC: 82%
- Accuracy: 78%
- Precision: 68%
- f1-Score: 59

Test Data:

- AUC: 80%
- Accuracy: 77%
- Precision: 67%
- f1-Score: 57%

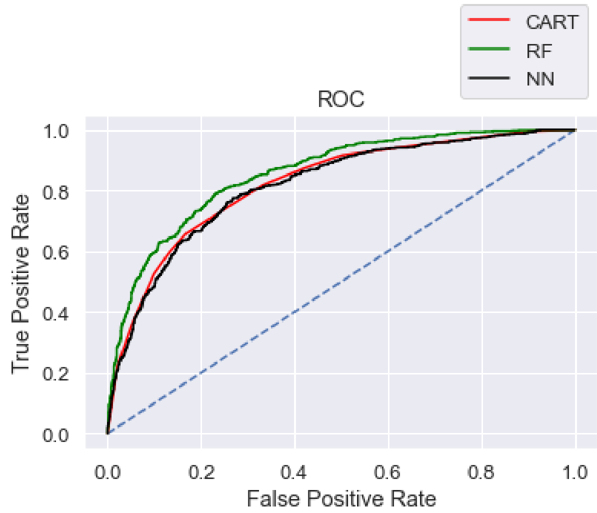
Training and Test set results are almost similar, and with the overall measures high, the model is a good model.

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

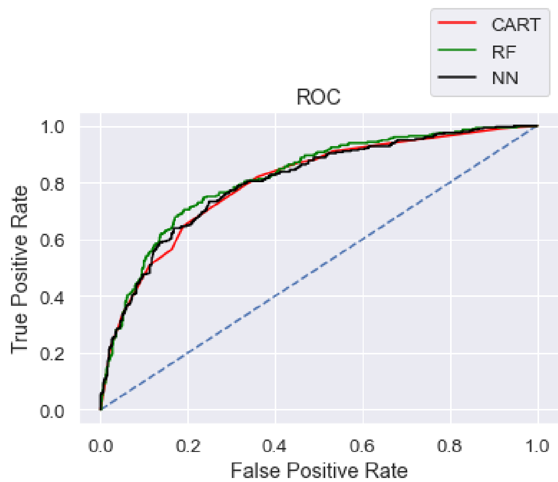
Comparison of the performance metrics from the 3 models

	CART Train	CART Test	Random Forest Train	Random Forest Test	Neural Network Train	Neural Network Test
Accuracy	0.79	0.77	0.80	0.78	0.78	0.77
AUC	0.82	0.80	0.86	0.82	0.82	0.80
Recall	0.53	0.51	0.61	0.56	0.51	0.50
Precision	0.70	0.67	0.72	0.68	0.68	0.67
F1 Score	0.60	0.58	0.66	0.62	0.59	0.57

#ROC Curve for the 3 models on the Training data



#ROC Curve for the 3 models on the Test data



CONCLUSION:

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

Comparing the 3 models, recall of 61% is obtained for the RF model, which is good for this model as it's above 0.5. Recall = TP/TP+FN.

High precision relates to the low false positive rate. RF model has relatively high precision rate of 68% and for train data 72%. This is higher when compared to others. Hence RF seems to be a better model as a conclusion.

2.5 Inference: Basis on these predictions, what are the business insights and recommendations:

I strongly recommended we collect more real time unstructured data and past data if possible.

This is understood by looking at the insurance data by drawing relations between different variables such as day of the incident, time, age group, and associating it with other external information such as location, behavior patterns, weather information, airline/vehicle types, etc.

• Streamlining online experiences benefitted customers, causes to an increase in conversions, which helps raised profits. • As per the data 90% of insurance is done by online channel.

• Other interesting fact, is almost all the offline business has a claimed associated, need to find why?

• Need to train the JZI agency resources to pick up sales as they are in bottom, need to run promotional marketing campaign or evaluate if we need to tie up with alternate agency • Also based on the model we are getting 80%accuracy, so we need customer books airline tickets or plans, cross sell the insurance based on the claim data pattern.

• Other interesting fact is more sales happen via Agency than Airlines and the trend shows the claim are processed more at Airline. So, we may need to deep dive into the process to understand the workflow and why?

Key performance indicators (KPI) The KPI’s of insurance claims are: • Reduce claims cycle time • Increase customer satisfaction • Combat fraud • Optimize claims recovery • Reduce claim handling costs Insights gained from data and AI-powered analytics could expand the boundaries of insurability, extend existing products, and give rise to new risk transfer solutions in areas like a non-damage business interruption and reputational damage.