SMDM PROJECT SAMPLE REPORT

DSBA

Name:
Aishwarya G

PGP-DSDA Online

JUNE-22

DATE: 03-06-2022

Table of the content

List of the tables+

List of the figure

Clustering

Problem 1

Introduction

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

Data Dictionary for Market Segmentation

- 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)

Sample of the data

	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

Table of the sample of the data

Exploratory Data Analysis

Checking Missing values

```
<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
```

Table Checking Missing values

Pairplot

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

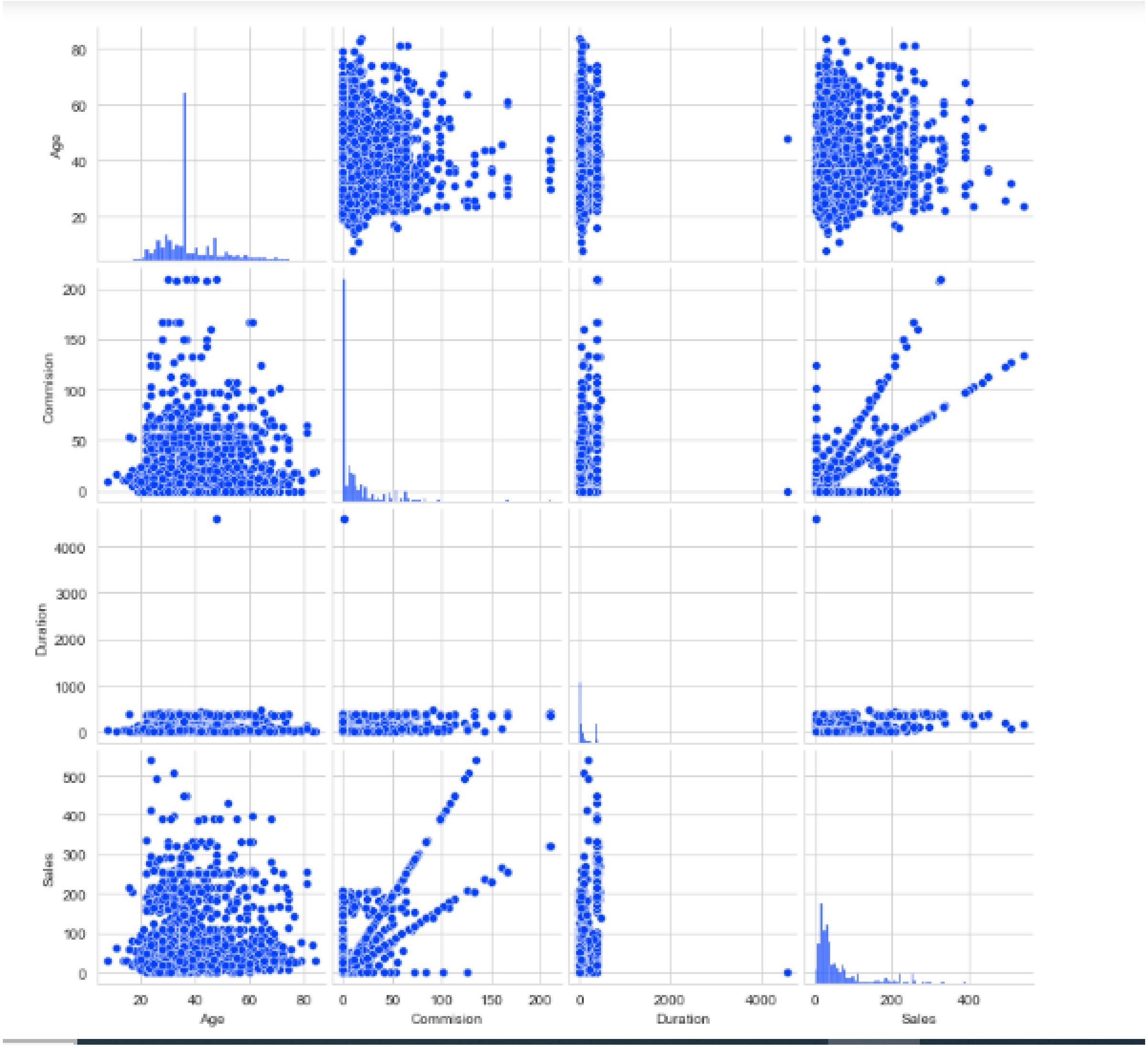


Fig pair plot

Heat map

: <AxesSubplot:>

Fig heat map

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

Loading the dataset

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

Missing values

	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

Table 1: Sample space

Checking the information

```
<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
```

Figure 1. Info of data

The given data set has 7 columns, the data type is float, the range index is 210 from 0 to 209 and 11.6 KB memory space is used

Checking the shape of the data set

The given data set has 210 rows and 7 columns

Checking the summary of 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

Table2. Summary of the data

- The average probability_of_full_payment is 87.10%
- the average of the spending and advance_payments is 14.48 and 14.55 they are almost similar to each other
- The maximum of the probability_of_full_payment is 0.9183
- the mean of credit_limit and min_payment_amt is 3.25 and 3.70
- cuurent balance mean and max_spent_in_single_shopping both arecorelated to each with mean is 5.6 and 5.4
- The average of max_spent_in_single_shopping is 5.408. The maximum of max_spent_in_single_shopping is 6.550

Checking the null values ormissing values

```
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
```

Fig 2. Null values

There is no null values in the given dataset

Checking the duplicate data

There is no duplicating data

Checking the data type

```
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
```

Fig 3. Data type

Count plot

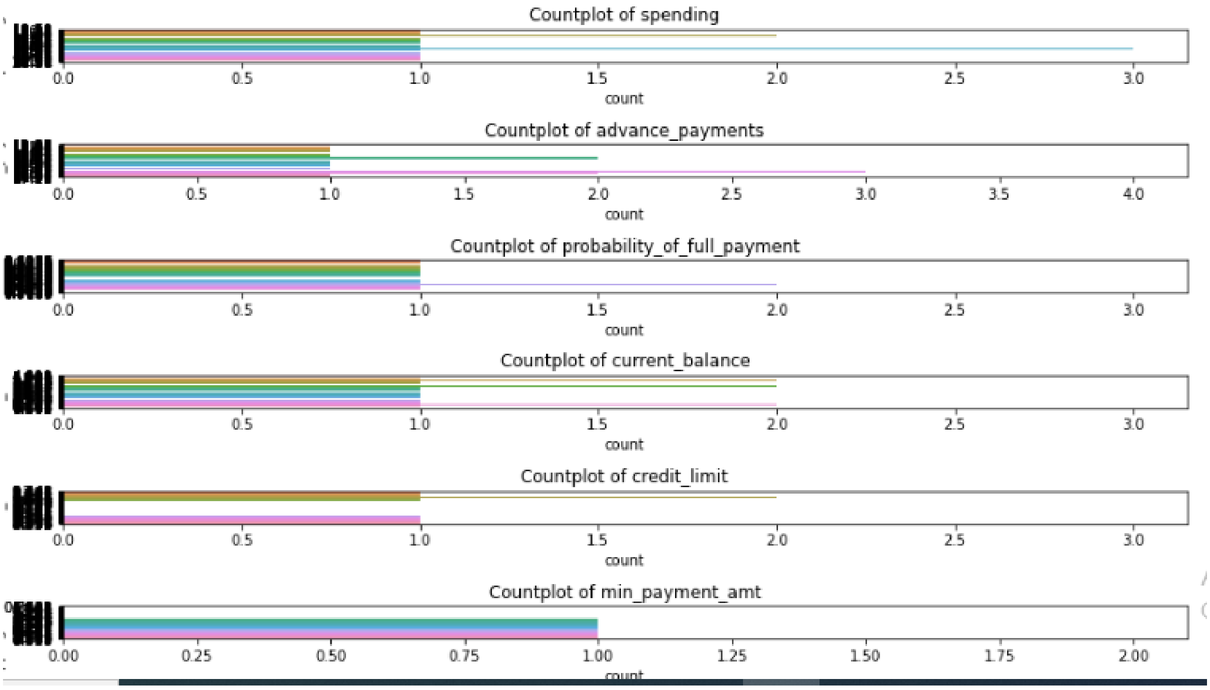


Fig 4. Countplot

Distribution plot

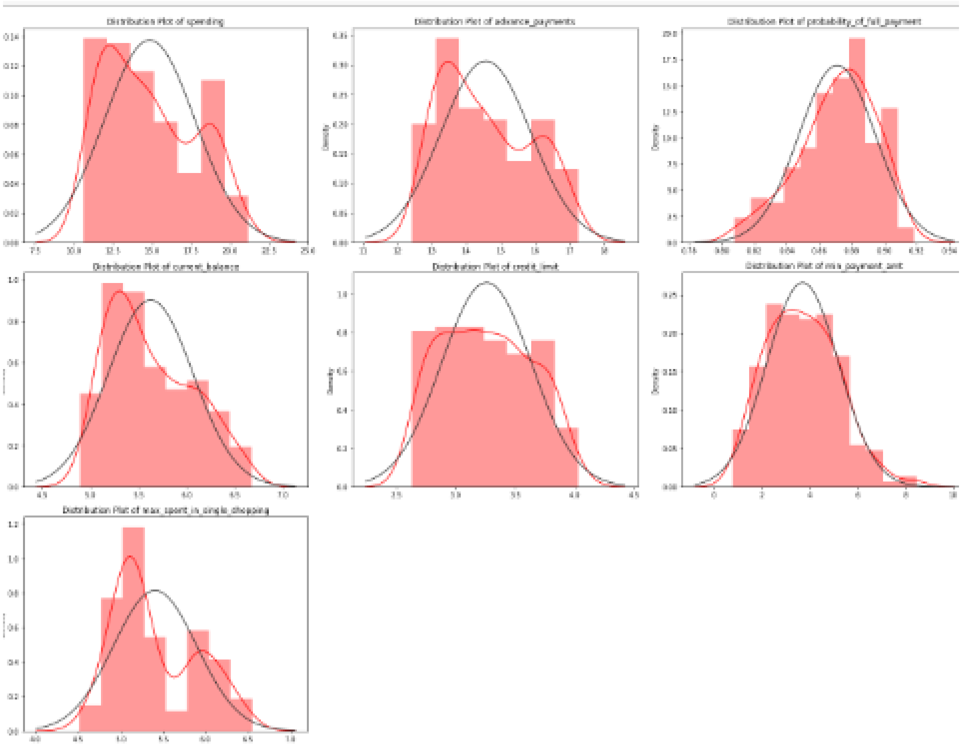


Fig 5 dist plot

Histplot

<AxesSubplot:ylabel='Frequency'>

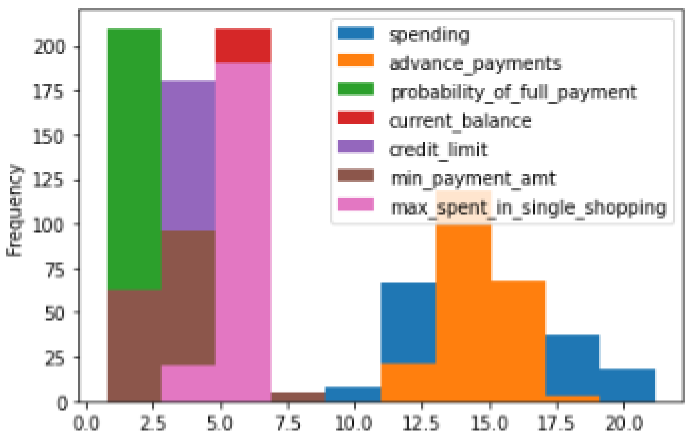
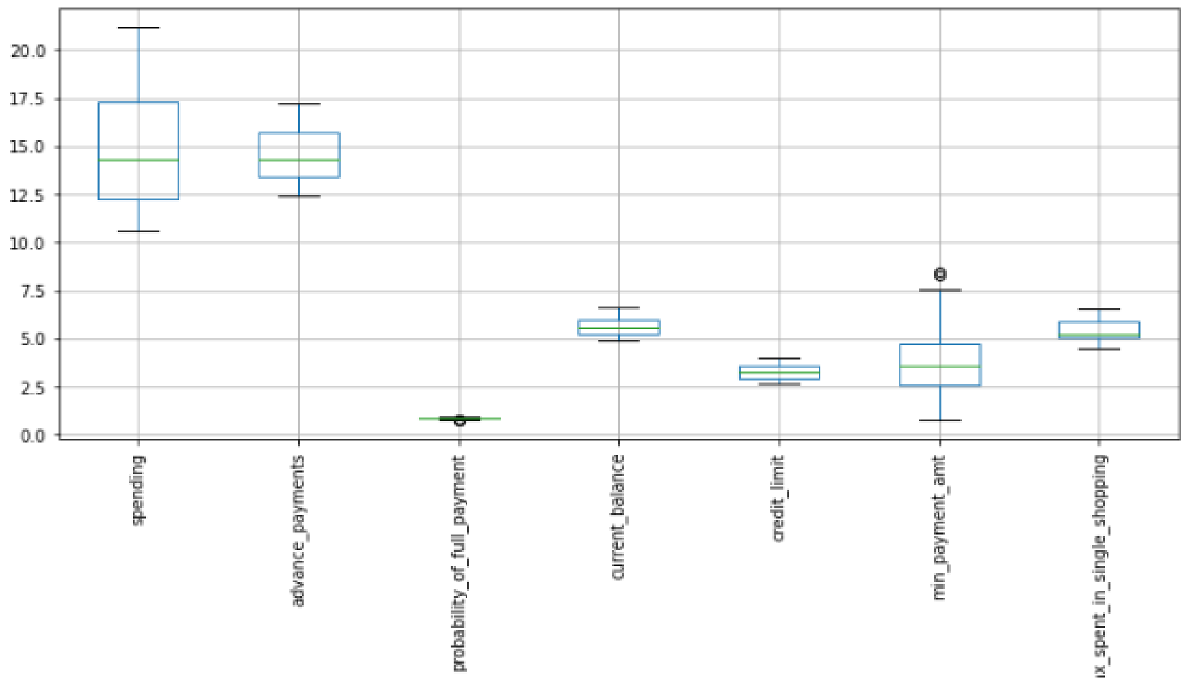


Fig 6. Hist plot

- maximum frequence is max_spent_in_single_shopping
- current payment has low frequence

BOX plot



- the given dataset has only two outliers
- min_payment_amt has outliers
- product_payments has outliers

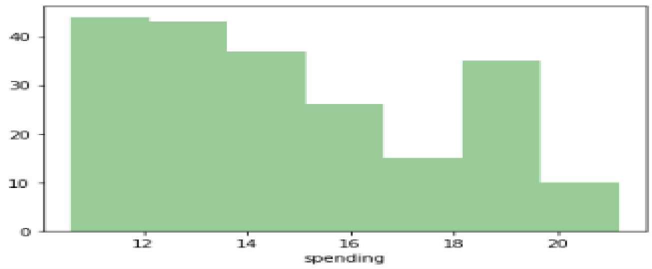
Skewnes and Kurtosis

- 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

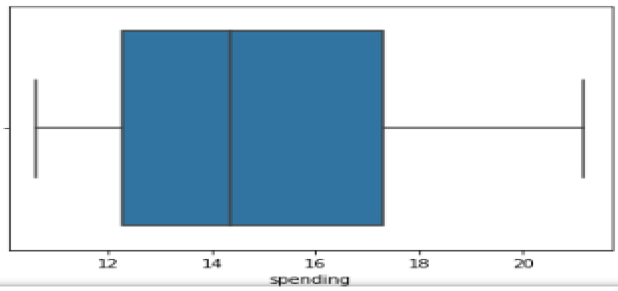
Univeriant

```
Description of spending
----------------------------------------------------------------------------
count     210.000000
mean       14.847524
std         2.909699
min        10.590000
25%        12.270000
50%        14.355000
75%        17.305000
max        21.180000
Name: spending, dtype: float64 Distribution of spending
----------------------------------------------------------------------------


----------------------------------------------------------------------------
```




```
BoxPlot of spending
----------------------------------------------------------------------------
```



Bivariant analysis

Pairplot

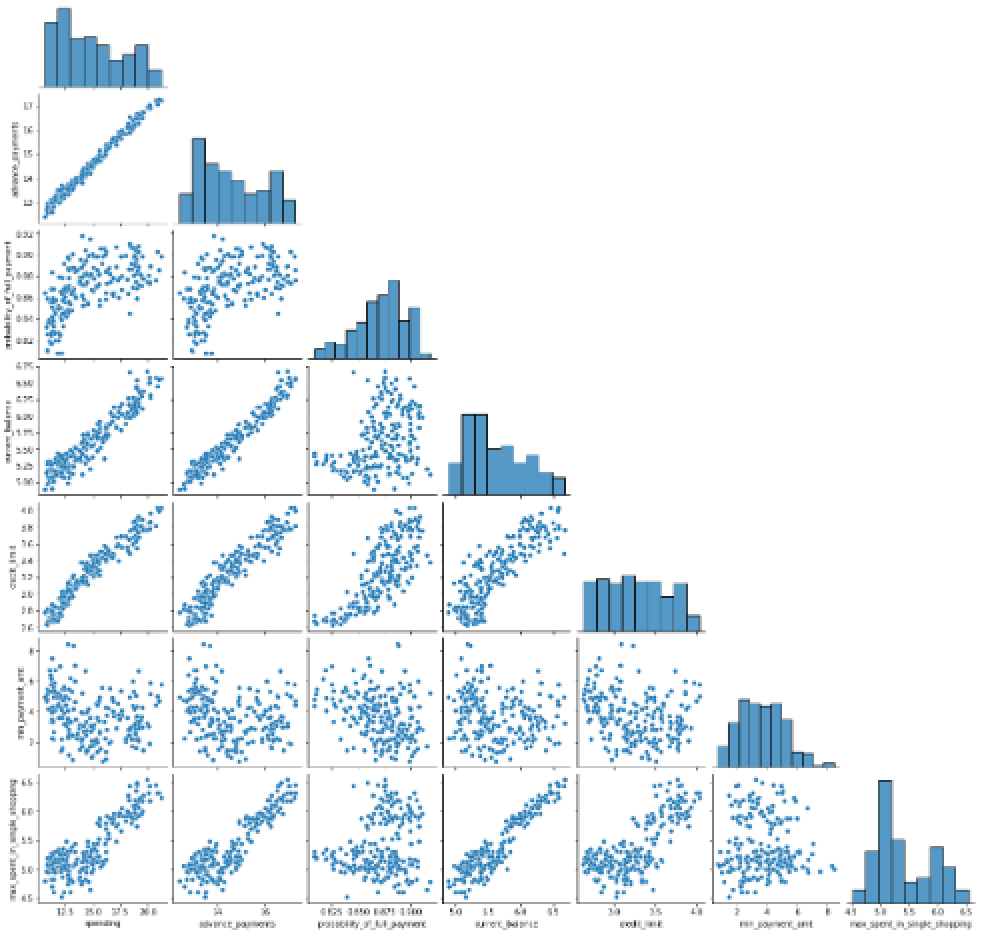


Fig 8. Pairplot

Heatmap

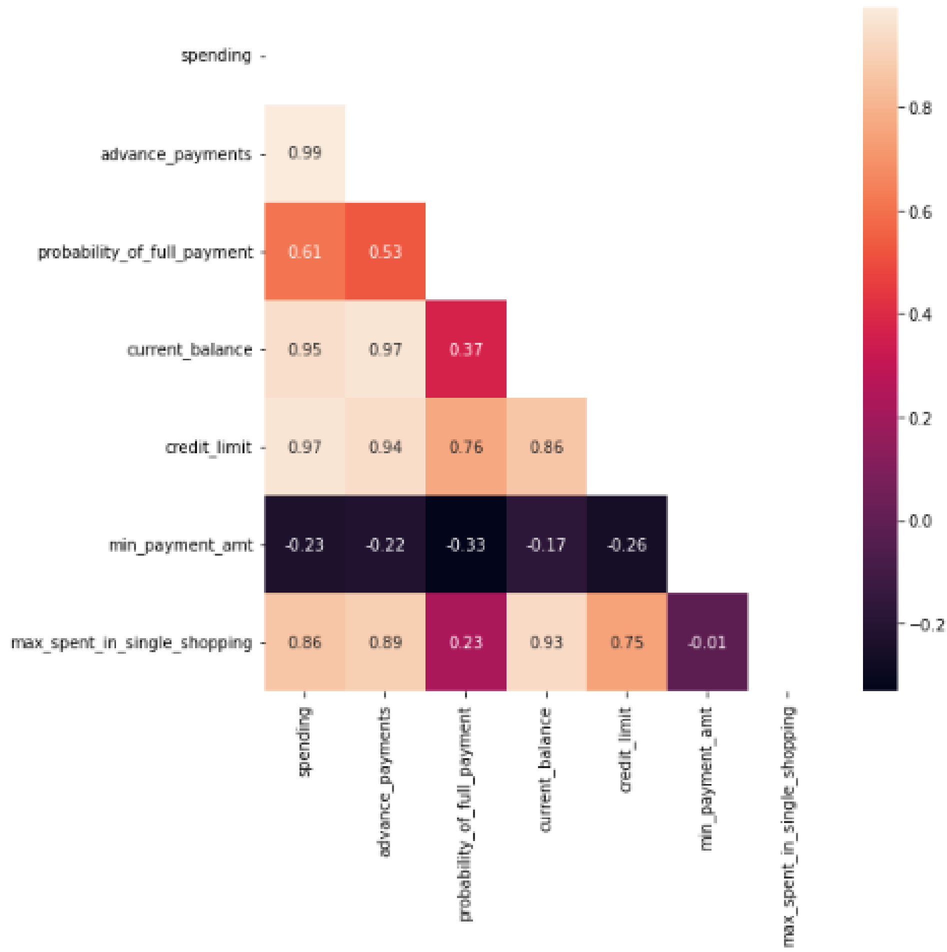


Fig 9 heatmap

- spending has low corelation
- credit_limt and current_balance has the similar corelation
- credit limit has the high corelation

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

Yes. Clustering algorithms such as K-means do need feature scaling before they are fed to the algo

- When we standardize the data prior to performing cluster analysis, the clusters change. We find that with more equal scales, the Percent Native American variable more significantly contributes to defining the clusters.
- Standardizing data is recommended because otherwise the range of values in each feature will act as a weight when determining how to cluster data, which is typically undesired.

	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	clusters
0	1.754355	1.811968	0.178230	2.367533	1.338579	-0.298806	2.328998	-1.414214
1	0.393582	0.253840	1.501773	-0.600744	0.858236	-0.242805	-0.538582	0.707107
2	1.413300	1.428192	0.504874	1.401485	1.317348	-0.221471	1.509107	-1.414214
3	-1.384034	-1.227533	-2.591878	-0.793049	-1.639017	0.987884	-0.454961	0.707107
4	1.082581	0.998364	1.196340	0.591544	1.155464	-1.088154	0.874813	-1.414214

Table 3 scaled data

- zscore method was using to scale the data
- The main advantage of scaling is to avoid attributes in greater numeric ranges dominating those in smaller numeric range
- The performing scaling to bring the measurements so the ranges are close, then I can see that there is an increase in computation performance. Distances can be computed quickly since all columns are scaled in the same manner
- not having to calculate very large distances due to differences in ranges
- Min-Max Normalization: This technique re-scales a feature or observation value with distribution value between 0 and 1

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

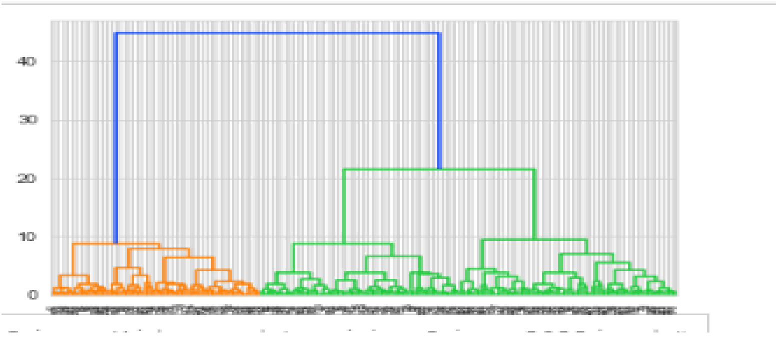


Fig 10 dendrogram

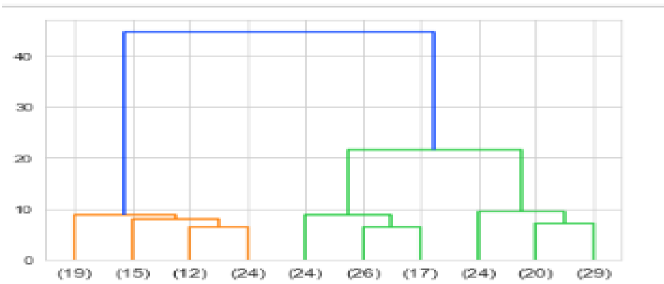


Fig 11 dendogram

```
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)
```

	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

Fig 12 cluster

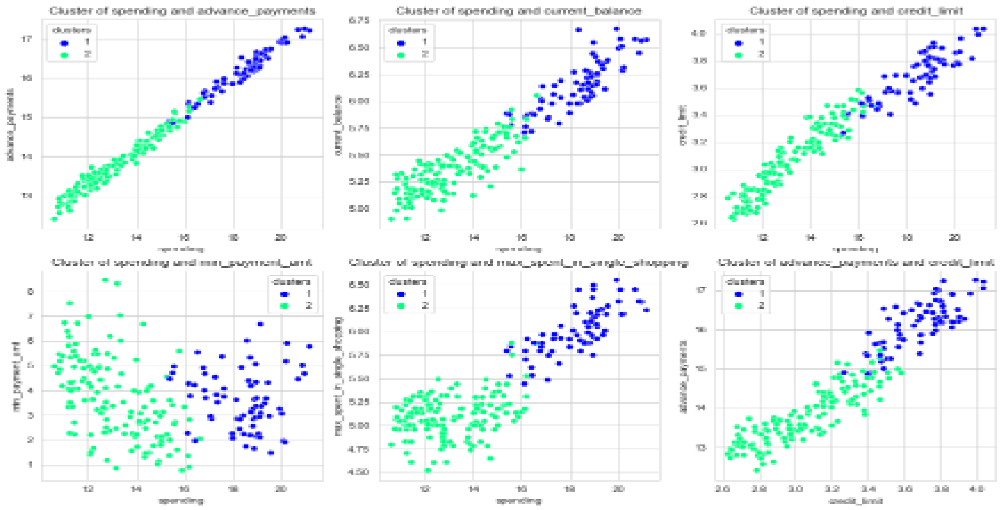


Fig 13 scatter plot

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.**

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

Fig 14 K mean labels

Wss

```
[1680.0000000000002,
 674.3326593640829,
 438.6638700226693,
 388.4350896236265 4,
 339.1866500960827,
 304.3765542616248 5,
 272.1689081624432 4,
 246.0043537917782 1,
 227.2700055707915 7,
 213.7915744318934 2]
```

```
[<matplotlib.lines.Line2D at 0x26e2db19d60>]
```

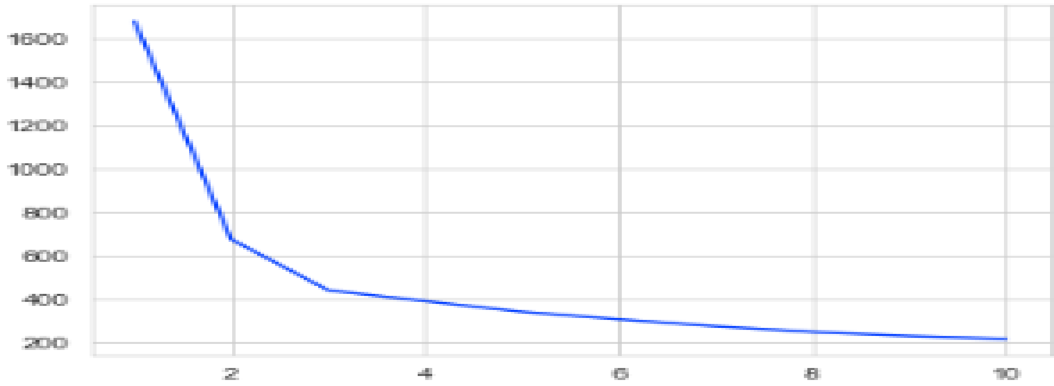


Fig 15 range of wss

	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	clusters	Clus_kme
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	

Table 4. sil_width

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

Cluster 1: high-level customers Cluster 3: low-level customers Cluster 2: middle-level customers

Customers under cluster 1 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.

- Tie up with luxary brands, which will drive more one_time_maximun spending
- maximum max_spent_in_single_shopping is high for this group, so can be offered discount/offer on next transactions upon full payment

Customers under cluster 3have 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 should be given remainders for payments.
- Offers can be provided on early payments to improve their payment rate.
- Increase there spending habits by tieing up with grocery stores, utlities (electircity, phone, gas, others)

Customers under cluster 2 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.
- Promote premium cards/loyality cars to increase transcations.
- Increase spending habits by trying with premium ecommerce sites, travel portal, travel airlines/hotel, as this will encourge them to spend more

CART-RF-ANN

Problem 2

Introduction

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.

Sample of the data

	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

Table sample of the data

Exploratory Data Analysis

Checking Missing values

```
<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
```

checking missing values

Pairplot

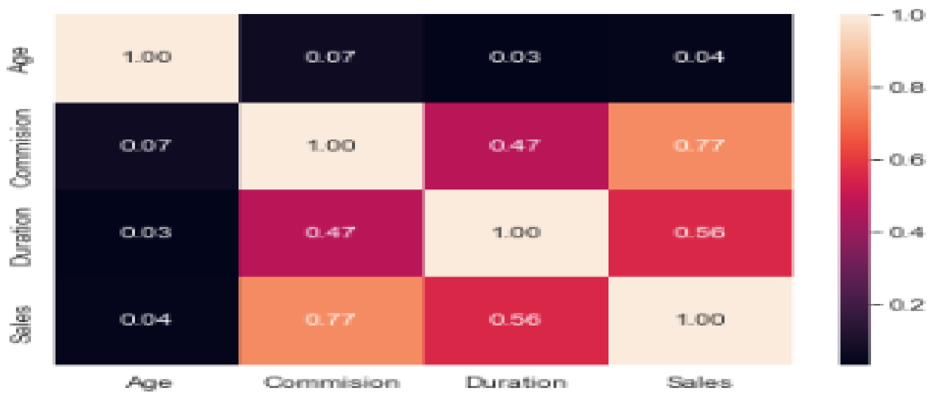
Pairplot shows the relationship between the variables in the form of scatterplot and the distribution of the variable in the form of histogram.

```
95]: sns.pairplot(df)
95]: <seaborn.axisgrid.PairGrid at 0x26e2db316d0>
```

pair plot

Heat map

```
<AxesSubplot:>
```



Heat map

QUESTIONS

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

View the top 5 rows:

	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

Table 4. Top 5 rows

Check the describe the data

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

Table 5 summary of the data

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
Age	3000.0	NaN	NaN	NaN	38.091	10.463518	8.0	32.0	36.0	42.0	84.0
Agency_Code	3000	4	EPX	1365	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Type	3000	2	Travel Agency	1837	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Claimed	3000	2	No	2076	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Commision	3000.0	NaN	NaN	NaN	14.529203	25.481455	0.0	0.0	4.63	17.235	210.21
Channel	3000	2	Online	2954	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Duration	3000.0	NaN	NaN	NaN	70.001333	134.053313	-1.0	11.0	26.5	63.0	4580.0
Sales	3000.0	NaN	NaN	NaN	60.249913	70.733954	0.0	20.0	33.0	69.0	539.0
Product Name	3000	5	Customised Plan	1136	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Destination	3000	3	ASIA	2465	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Table 6: summary of the data including all

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

Fig null values

- There is no null values in the gievn data set

Information about the 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
```

Fig16. Information of the data

- 10 variables
- Age, Commision, Duration, Sales are numeric variable
- rest are categorial variables
- 3000 entries, from 0 to 2999
- 7 columns
- 9 independant variable that are age,agency_code,type,commision,channel,duration,sales,product name,destination
- Clamied is only one target variable

Checking the duplicated values to given the dataset

There are 139 duplicating values. We are not removing duplicating values because there is no unique identifier and Id is not species in the given data dataset

Shape function is used to identified the number of rows and columns

There are 3000 rows and 10 columns in this dataset

Checking the type of the data set

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

Fig 17 data types

- 3 types of data tpye
- age,duration both are the int data type
- Agency_Code,Type,Claimed,channel,Product name and destination these are the object data type
- commisiion and sales are the floate data type

Checking the unique values

```
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
```

Checking histplot

- destination has high frequency
- sales and duration has equal frequence
- age has the meadian frequency
- product name, clamed, type, comisiion has the minimum frequence

<AxesSubplot:ylabel='Frequency'>

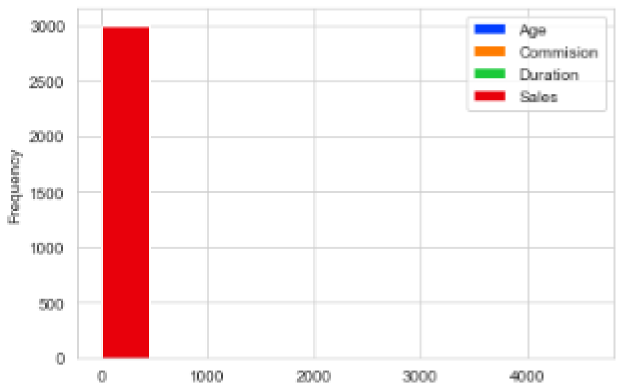


Fig 18 histplot

Checking the categorical values to using the count plot

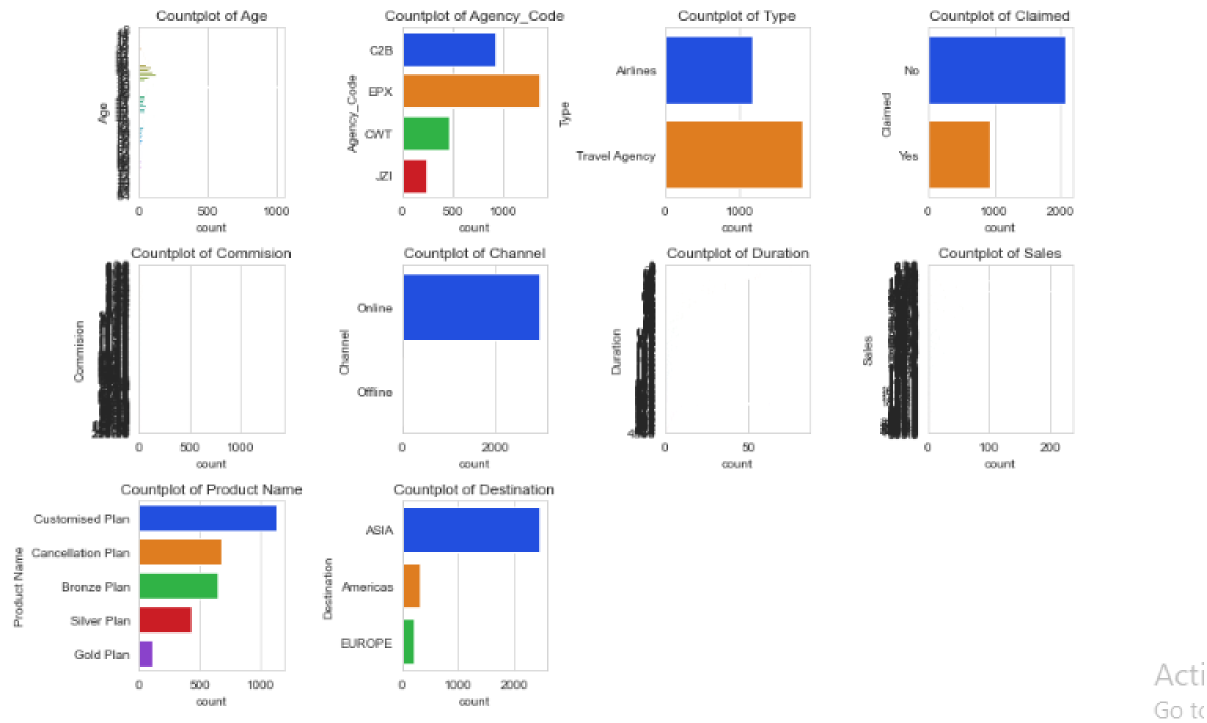


Fig 19. countplot

Converting object data type into numeric data type

	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

Table 8. Numeric type data

Checking the Skewness and kurtosis

Skewness of Age is 1.15
Kurtosis of Age is 1.65
Skewness of Agency_Code is -0.16
Kurtosis of Agency_Code is -1.3
Skewness of Type is -0.46
Kurtosis of Type is -1.79
Skewness of Claimed is 0.83
Kurtosis of Claimed is -1.31

Skewness of Commision is 3.15
Kurtosis of Commision is 13.98
Skewness of Channel is -7.89
Kurtosis of Channel is 60.34
Skewness of Duration is 13.78
Kurtosis of Duration is 427.59
Skewness of Sales is 2.38
Kurtosis of Sales is 6.16
Skewness of Product Name is 0.43
Kurtosis of Product Name is -0.58
Skewness of Destination is 2.19
Kurtosis of Destination is 3.49

Checking the outliers to using the boxplot

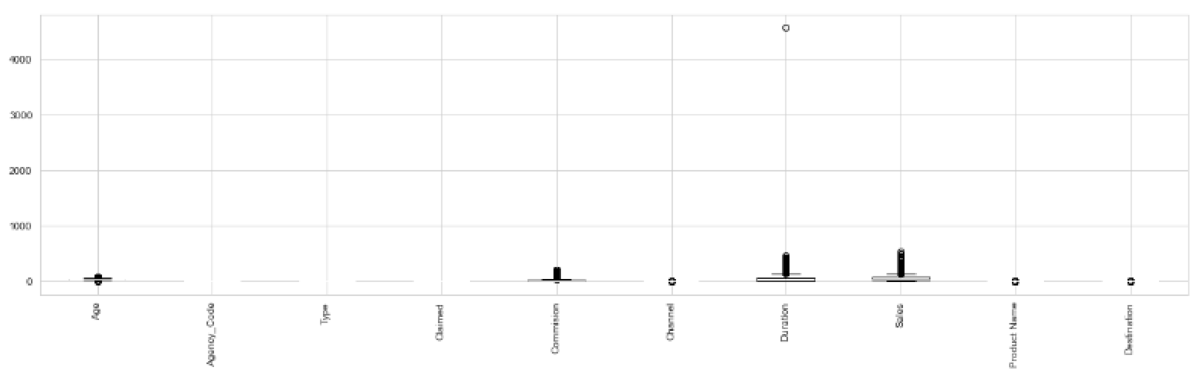


Fig 20. Box plot

Treating the outliers

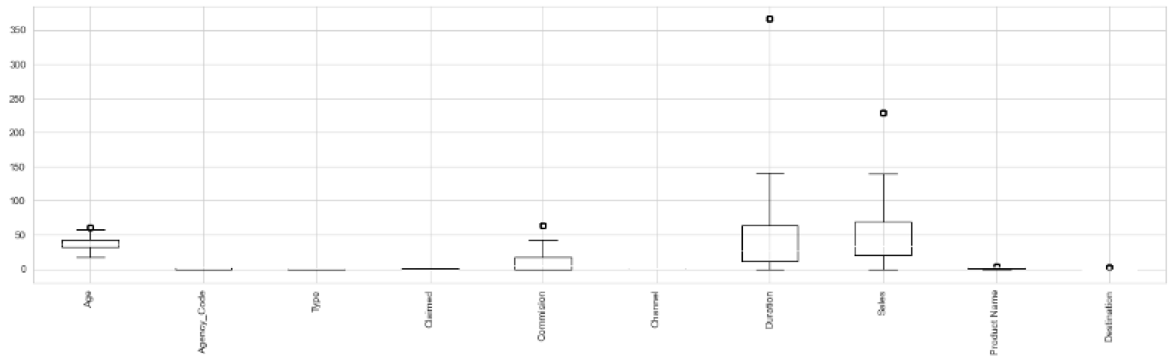


Fig 21. After treating the outlier boxplot

Univariate Analysis

```
Description of Age
--------------------------------------------------------------------------------
count    3000.000000
mean       37.785333
std         9.513889
min        17.000000
25%        32.000000
50%        36.000000
75%        42.000000
max        60.000000
Name: Age, dtype: float64 Distribution of Age
--------------------------------------------------------------------------------
```

Table 9. description of the Age

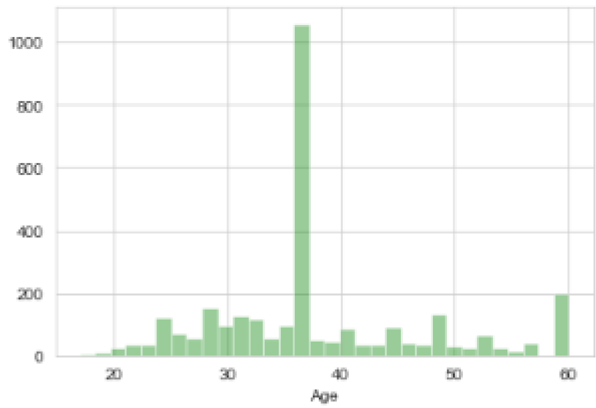


Fig 22. Distplot

```
BoxPlot of Age
--------------------------------------------------------------------------------
```

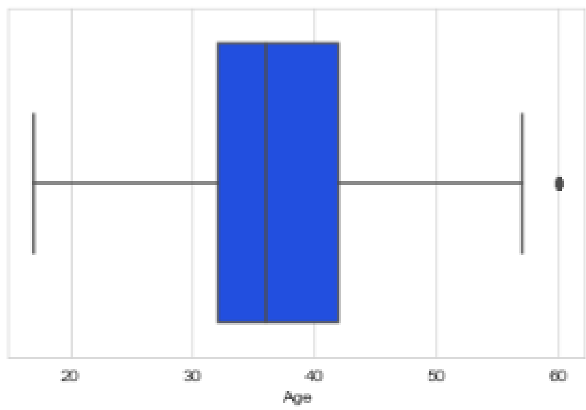


Fig 23. Box plot of the age

bivariant analysis

Pairplot

GREATE LEARNING

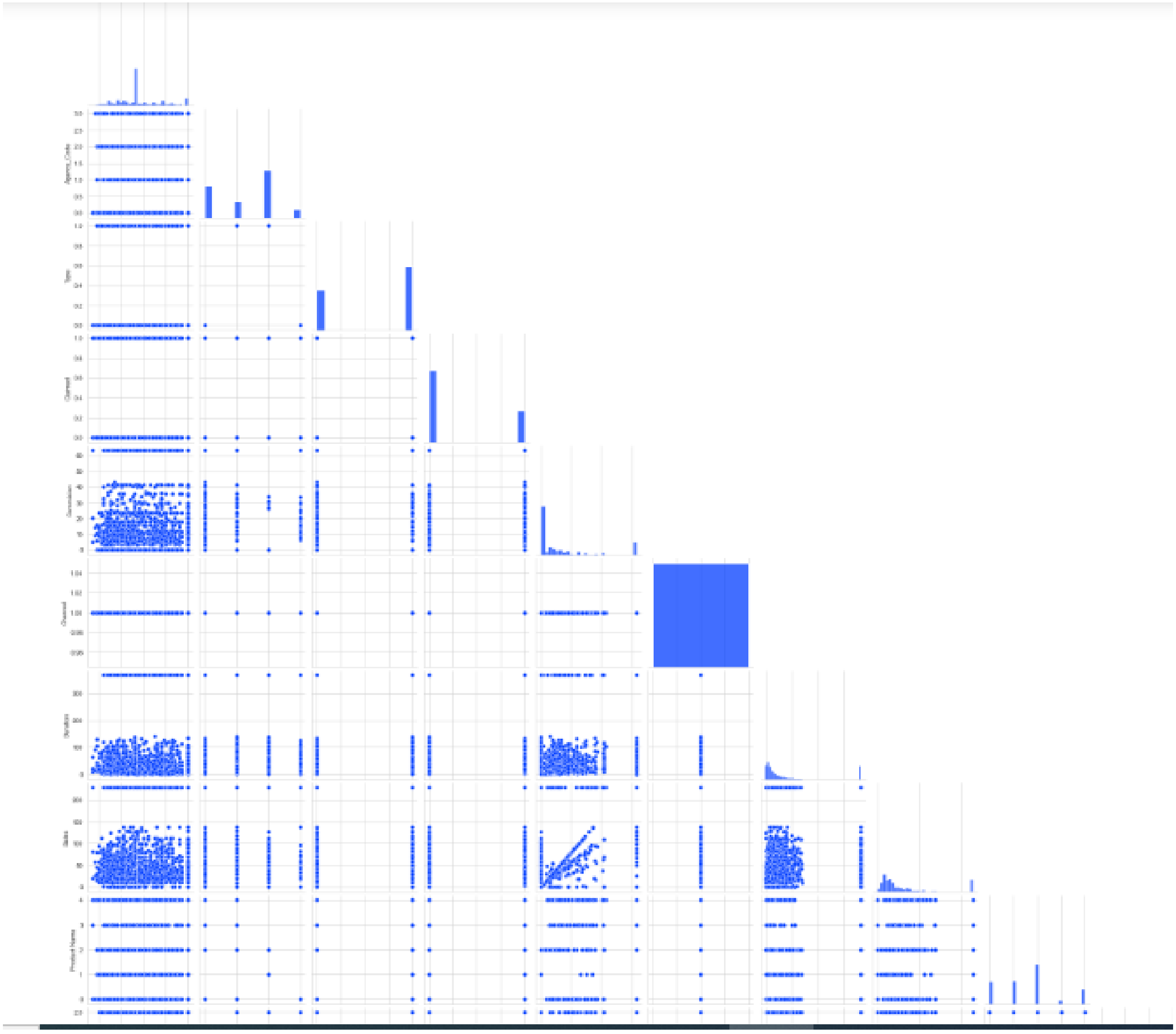


Fig 24 pair plot

HeatMap

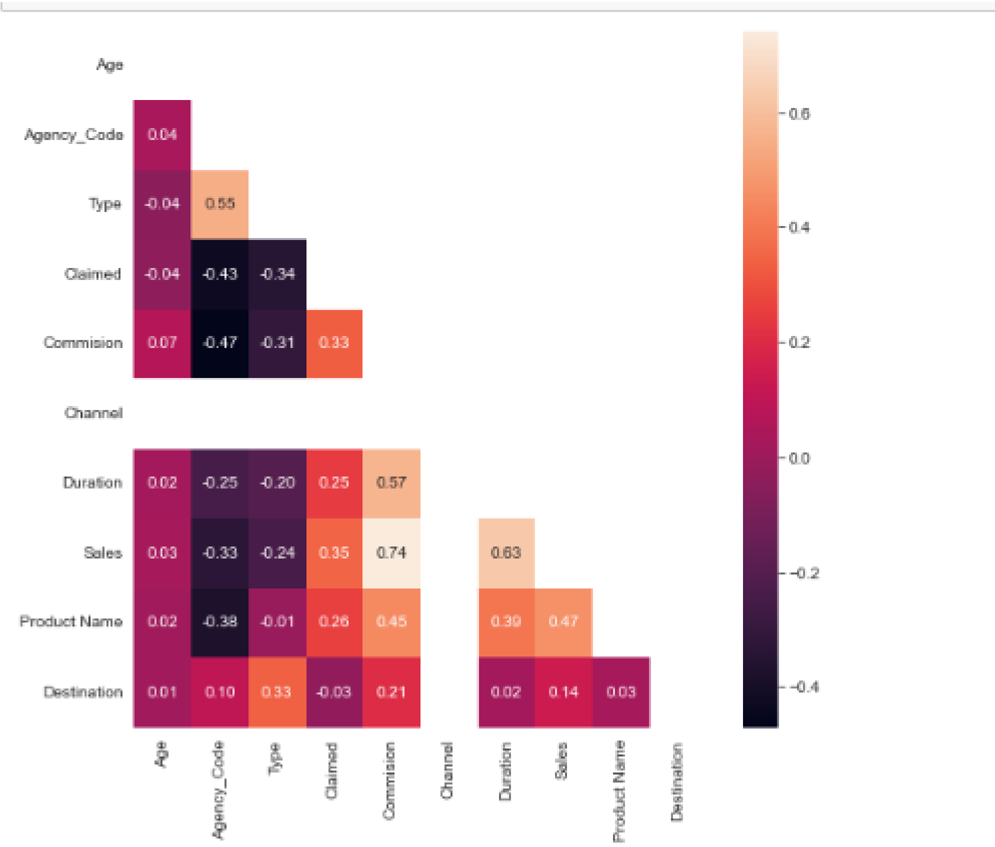


Fig 25 heat map

- Product name and duration are the corelation
- channel variable has low correlated
- channel and typ correlated each other
- sales has the high corelated
- 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.

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

```
dataframe.head()
```

	Age	Agency_Code	Type	Claimed	Commision	Channel	Duration	Sales	Product Name	Destination
0	48.0	0	0	0	0.70	1.0	7.0	2.51	2.0	0.0
1	36.0	2	1	0	0.00	1.0	34.0	20.00	2.0	0.0
2	39.0	1	1	0	5.94	1.0	3.0	9.90	2.0	2.0
3	36.0	2	1	0	0.00	1.0	4.0	26.00	1.0	0.0
4	33.0	3	0	0	6.30	1.0	53.0	18.00	0.0	0.0

Table 10. Numeric data type table

Capture the target column ("Claimed") 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	0.70	1.0	7.0	2.51	2.0	0.0
1	36.0	2	1	0.00	1.0	34.0	20.00	2.0	0.0
2	39.0	1	1	5.94	1.0	3.0	9.90	2.0	2.0
3	36.0	2	1	0.00	1.0	4.0	26.00	1.0	0.0
4	33.0	3	0	6.30	1.0	53.0	18.00	0.0	0.0

Table 11. drop and pop the claimed variable

Splitting data into training and test set for independent attributes

Checking the dimensions of the training and test data

X_train (2100, 9)

X_test (900, 9)

train_labels (2100,)

test_labels (900,)

Decision Tree Classifier

Importing the DecisionTreeClassifier library before classifying the decision tree

Using the fit function

Importing the tree

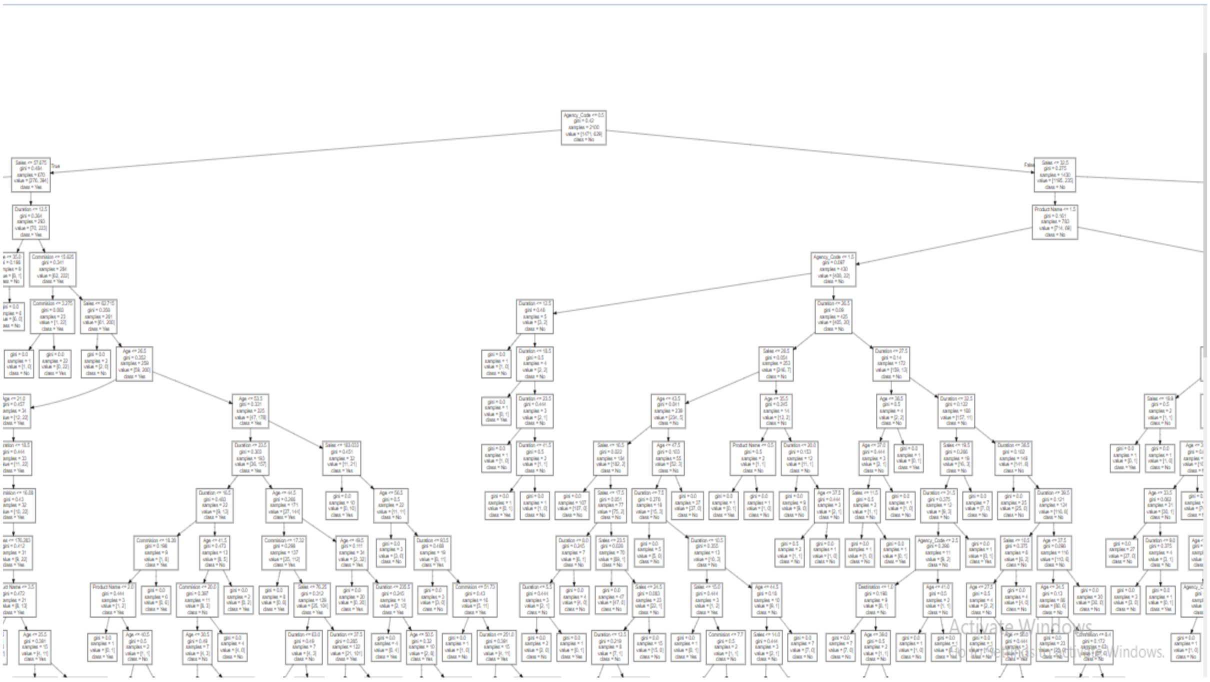


Fig 26. Tree

Variable Importance – DTCL

	Imp
Age	0.188118
Agency_Code	0.200248
Type	0.002422
Commision	0.079819
Channel	0.000000
Duration	0.257644
Sales	0.211639
Product Name	0.046894
Destination	0.013215

Regularising the Decision Tree

Table 12 Regularising the Decision Tree

Descision tree classifiers

DecisionTreeClassifier(max_depth=7, min_samples_leaf=10, min_samples_split=30)

Predicting on Training and Test dataset

Getting the Predicted Classes and Probs

	0	1
0	0.900000	0.100000
1	0.539474	0.460526
2	0.539474	0.460526
3	0.157895	0.842105
4	0.909722	0.090278

Table 13. probs of predicting values

Building a Random Forest Classifier

- Importing thr RandomForestClassifier library
- Ensemble RandomForest Classifier using fit function
- Gridesearch

Importing the GridSearchCV from skilearin

- Using grid search fit function

- GridSearchCV(cv=3, estimator=RandomForestClassifier(),

  param_grid = {'max_depth': [7, 10], 'max_features': [4, 6],

  'min_samples_leaf': [50, 100],

  'min_samples_split': [150, 300],

  'n_estimators': [301, 501]})

- Best parameter

{'max_depth': 7, 'max_features': 4, 'min_samples_leaf': 100, 'min_samples_split': 150, 'n_estimators': 301}

RandomForestClassifier(max_depth=7, max_features=4, min_samples_leaf=100, min_samples_split=150, n_estimators=301)

- predicting the train and test set

  Traing Data Accuracy=0.7888

  Test Data Accuracy =0.7655

- Getting the Predicted Classes and Probs

	0	1
0	0.716335	0.283665
1	0.553505	0.446495
2	0.569011	0.430989
3	0.339873	0.660127
4	0.913227	0.086773

Building Neural Network Classifier

- Splitting the data but already we are splitting data into training and test set for independent attributes
- Importing the StandardScaler library
- Fit transform to the train data

```
array([[-0.1807363 ,  0.72815922,  0.80520286, ..., -0.60667267,
         0.24642411, -0.47078709],
       [-0.1807363 ,  0.72815922,  0.80520286, ..., -0.28128834,
         0.24642411,  2.1241024 ],
       [-1.03725814, -1.28518425, -1.24192306, ...,  2.47804469,
         1.83381865, -0.47078709],
       ...,
       [-0.1807363 ,  0.72815922,  0.80520286, ...,  0.02930579,
         0.24642411, -0.47078709],
       [ 0.67578553,  1.73483096, -1.24192306, ..., -0.63625306,
        -1.34097044, -0.47078709],
       [-0.1807363 , -1.28518425, -1.24192306, ..., -0.56969718,
         1.83381865, -0.47078709]])
```

- Importing the MLPClassifier from skilearn
- Predict from the ML classifier

Iteration 1, loss = 3.91635353
Iteration 2, loss = 1.14759977
Iteration 3, loss = 0.87399930
Iteration 4, loss = 0.85114685
Iteration 5, loss = 0.82770191
Iteration 6, loss = 0.65782017
Iteration 7, loss = 0.70015815
Iteration 8, loss = 0.70203240
Iteration 9, loss = 0.60669455
Iteration 10, loss = 0.63072823
Iteration 11, loss = 0.53976011
Iteration 12, loss = 0.58051712
Iteration 13, loss = 0.55163058
Iteration 14, loss = 0.62117620
Iteration 15, loss = 0.56366316

Iteration 16, loss = 0.55016429
Iteration 17, loss = 0.53132821
Iteration 18, loss = 0.58627305
Iteration 19, loss = 0.52923445
Iteration 20, loss = 0.65944381
Iteration 21, loss = 0.53323320
Iteration 22, loss = 0.51428589
Iteration 23, loss = 0.51379583
Iteration 24, loss = 0.55026791
Iteration 25, loss = 0.53932687
Iteration 26, loss = 0.53330740
Iteration 27, loss = 0.50616965
Iteration 28, loss = 0.52665876
Iteration 29, loss = 0.50453283
Iteration 30, loss = 0.50528523
Iteration 31, loss = 0.51388876
Iteration 32, loss = 0.50566301
Iteration 33, loss = 0.53102493
Training loss did not improve more than tol=0.010000 for 10 consecutive epochs. St opping.

- Predicting the Training and Testing data
- Getting the Predicted Classes and Probs

	0	1
0	0.585071	0.414929
1	0.847220	0.152780
2	0.773200	0.226800
3	0.423054	0.576946
4	0.866583	0.133417

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 Model

- confusion matrix for training: where True positive (TP); False negative: True Negative; False Negative

```
array([[1317,  154],
       [ 255,  374]], dtype=int64)
```

- classification report for training

	precision	recall	f1-score	support
0	0.84	0.90	0.87	1471
1	0.71	0.59	0.65	629
accuracy			0.81	2100
macro avg	0.77	0.74	0.76	2100
weighted avg	0.80	0.81	0.80	2100

- matrices for training

DC_train_precision 0.71 # TP/TP+FP

DC_train_recall 0.59 #TP/TP+FN

DC_train_f1 0.65 #2TP/2TP+EP+FN

- Train Data Accuracy: 0.805238

TP+TN/TP+TN+FP+FN

- AUC and ROC for the training data

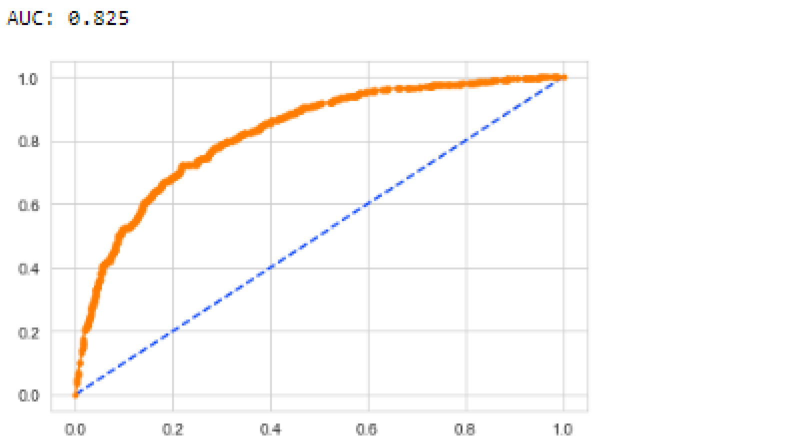


Fig 27. CART AUC and ROC curve for training data

CART testing data

- confusion matrix for testing

array([[539, 66],

[154, 141]], dtype=int64)

- classification report for testing

	precision	recall	f1-score	support
0	0.78	0.89	0.83	605
1	0.68	0.48	0.56	295
accuracy			0.76	900
macro avg	0.73	0.68	0.70	900
weighted avg	0.75	0.76	0.74	900

- metrics for testing

  DC_test_precision 0.68

  DC_test_recall 0.48

  DC_test_f1 0.56

- Test Data Accuracy: 0.7555555555
- AUC and ROC for the test data

AUC: 0.864

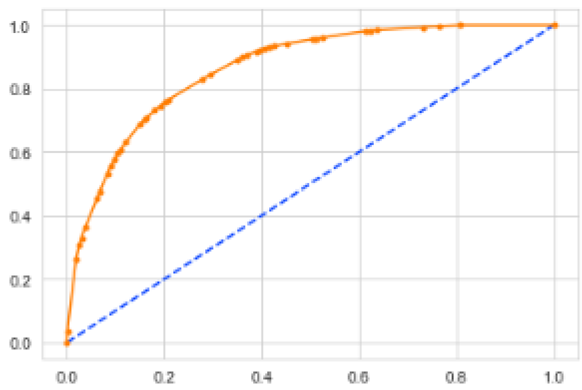


Fig 28. CART AUC and ROC curve for testing data

Random Forest Model

Random Forest Model Performance Evaluation on Training data

- Confusion matrix for training data

  array([[1342, 129],

  [315, 314]], dtype=int64)

- Classification report for training data

	precision	recall	f1-score	support
0	0.81	0.91	0.86	1471
1	0.71	0.50	0.59	629
accuracy			0.79	2100
macro avg	0.76	0.71	0.72	2100
weighted avg	0.78	0.79	0.78	2100

- Random forest metric for training data

  RF_train_precision 0.71

RF_train_recall 0.5

RF_train_f1 0.59

- Training Data Accuracy:0.7885
- AUC and ROC curve for training data

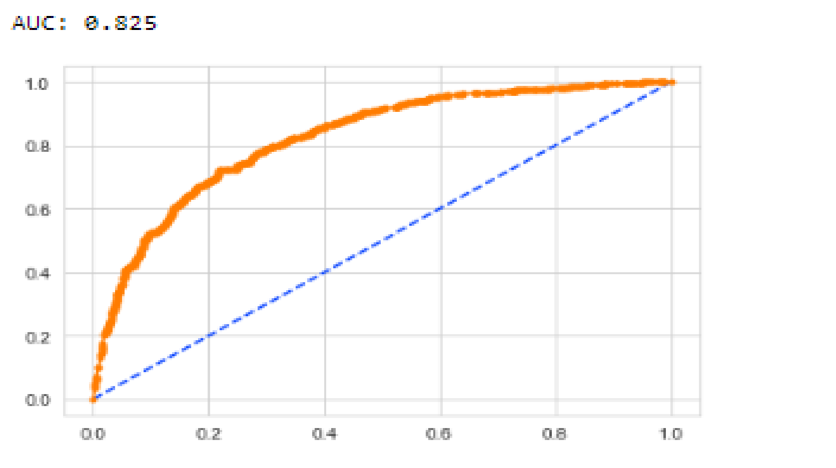


Fig 29. RF AUC and ROC curve for training data

Random Forest Model Performance Evaluation on testing data

- Confusion matrix for testing data

array([[563, 42],
       [177, 118]], dtype=int64)

- Classification report for testing data

	precision	recall	f1-score	support
0	0.76	0.93	0.84	605
1	0.74	0.40	0.52	295
accuracy			0.76	900
macro avg	0.75	0.67	0.68	900
weighted avg	0.75	0.76	0.73	900

- Random forest metric for testing data

RF_test_precision 0.74

RF_test_recall 0.4

RF_test_f1 0.52

- Testing Data Accuracy: 0.566666666
- AUC and ROC curve for testing data

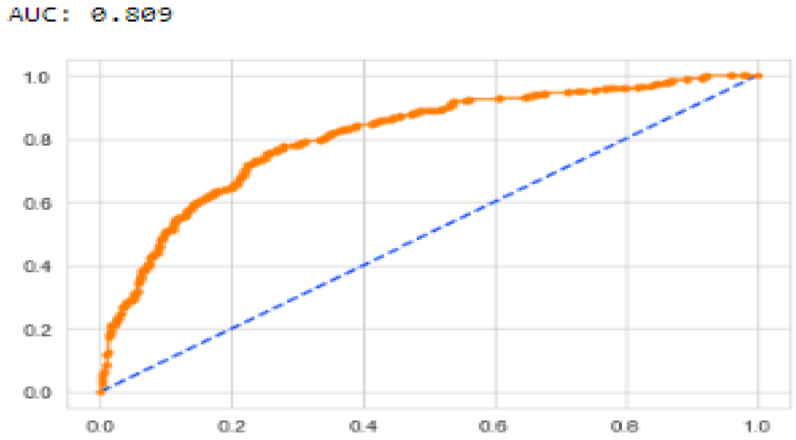


Fig 30. RF AUC and ROC curve for testin data

NN Model Performance Evaluation on Training data

- Confusion matrix for training data

  array([[1471,   0],
  [629,   0]], dtype=int64)

- Classification report for training data

	precision	recall	f1-score	support
0	0.70	1.00	0.82	1471
1	0.00	0.00	0.00	629
accuracy			0.70	2100
macro avg	0.35	0.50	0.41	2100
weighted avg	0.49	0.70	0.58	2100

- Random forest metric for training data

  NN_train_precision 0.0

  NN_train_recall 0.0

  NN_train_f1 0.0

- Training Data Accuracy:0.75285
- AUC and ROC curve for training data

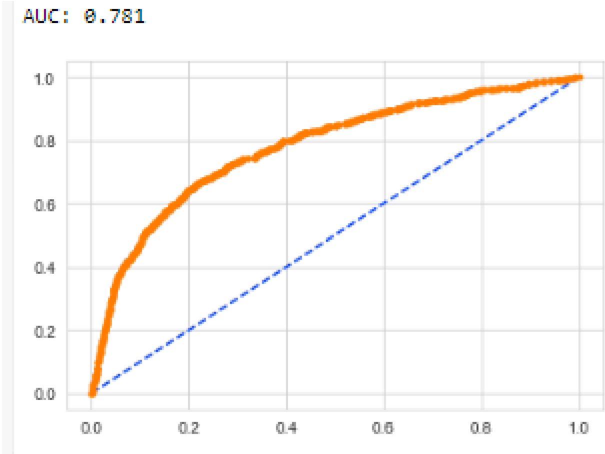


Fig 31. NN AUC and ROC curve for training data

NN Model Performance Evaluation on testing data

- Confusion matrix for testing data

  array([[585, 20],

  [234, 61]], dtype=int64)

- Classification report for testing data

	precision	recall	f1-score	support
0	0.71	0.97	0.82	605
1	0.75	0.21	0.32	295
accuracy			0.72	900
macro avg	0.73	0.59	0.57	900
weighted avg	0.73	0.72	0.66	900

- Random forest metric for testing data

  NN_test_precision 0.75

  NN_test_recall 0.21

  NN_test_f1 0.32

- Testing Data Accuracy:0.7177777
- AUC and ROC curve for testing data

AUC: 0.732

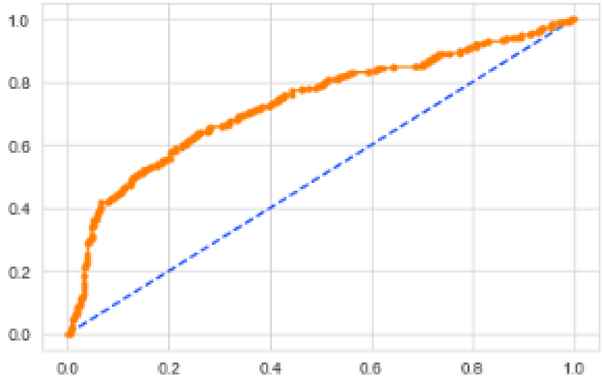


Fig 32. NN AUC and ROC curve for testing data

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	0.81	0.76	0.79	0.76	0.75	0.72
AUC	0.86	0.79	0.82	0.81	0.78	0.73
Recall	0.59	0.48	0.50	0.40	0.00	0.21
Precision	0.71	0.68	0.71	0.74	0.00	0.75
F1 Score	0.65	0.56	0.59	0.52	0.00	0.32

Table 14. Compare all model

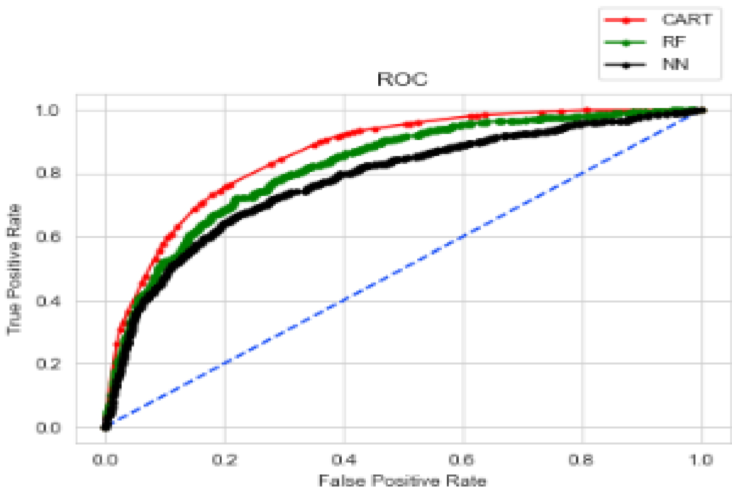


Fig 33.Train ROC curve for all model

: <matplotlib.legend.Legend at 0x26e3459a7f0>

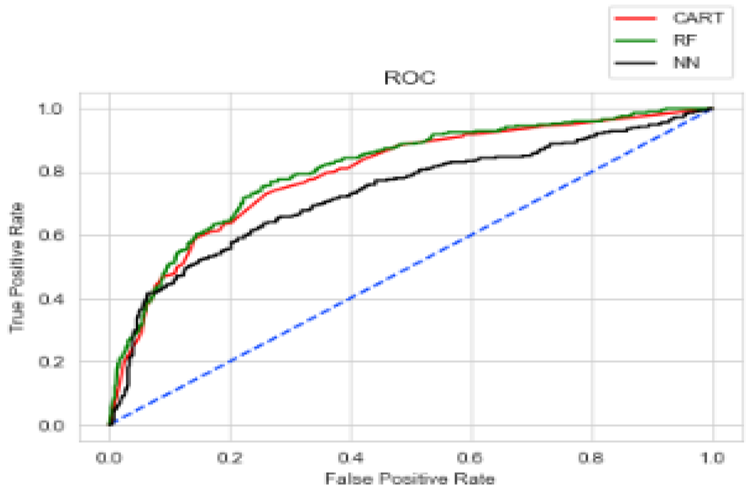


Fig 34. Test ROC curve for all model

I am selecting the classification and regration model, as it has better accuracy, precsion, recall, f1 score better than other two RF and NN.

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

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 locate. These data set has 10 columns and 210 rows Sales has high mean, commission has the low mean, and duration has negative values. Product name and duration are the correlation, channel variable has low correlated, channel and type correlated each other, sales has the high correlated, positive linear relationship between advance payments and spending, current balance and spending,, redit_limit and spending, current balance and advance payments, credit limit and advance payments, max_spent_in_single_shopping and current balance

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

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.

**********************************end***********************************