Problem 1: Clustering

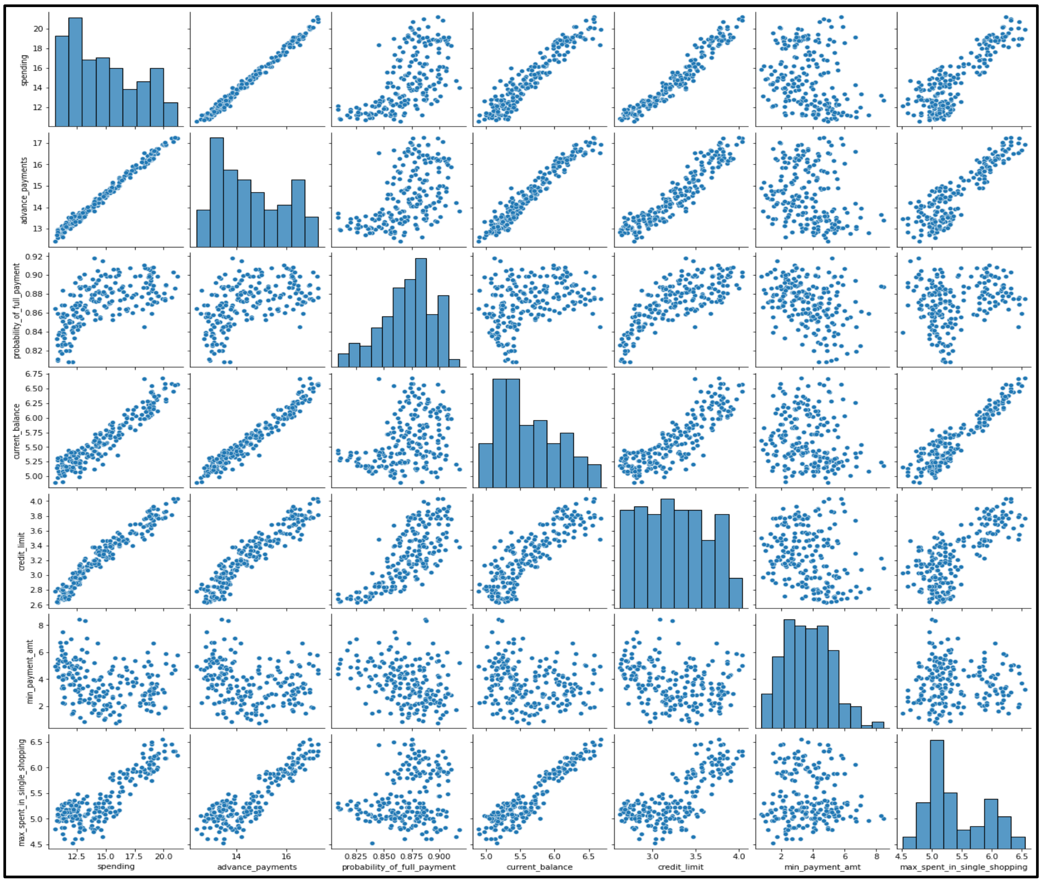
A leading bank wants to develop a customer segmentation to give promotional offers to its customers. They collected a sample that summarizes the activities of users during the past few months. You are given the task to identify the segments based on credit card usage.

There are 210 records in the given data set. Following are the variables in the given dataset-

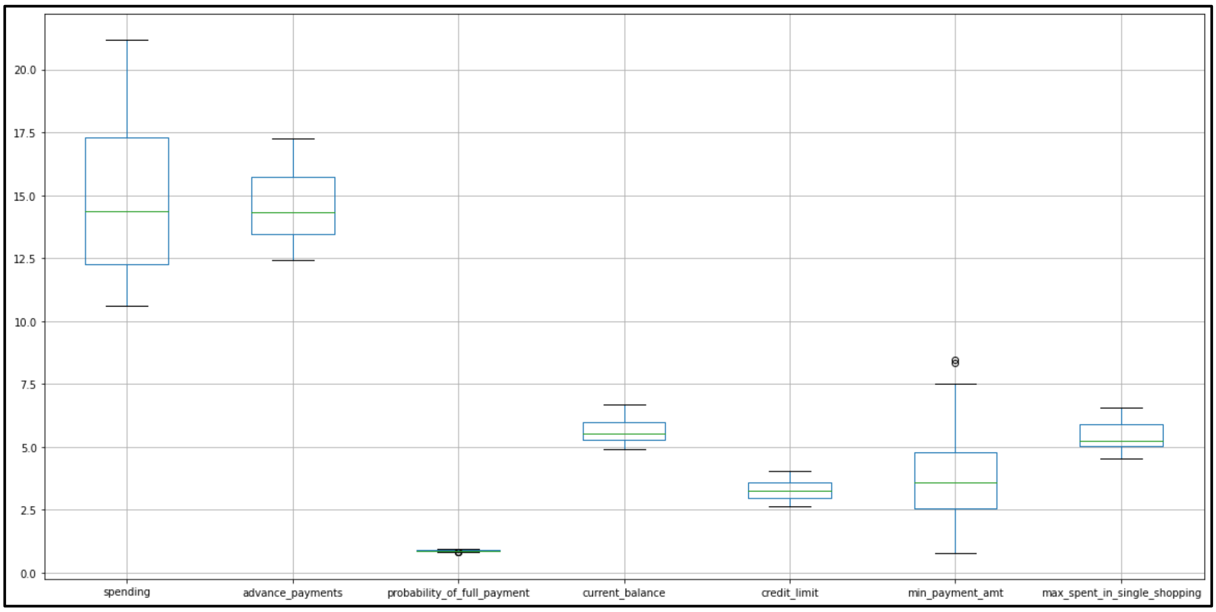
spending: Amount spent by the customer per month (in 1000s)
advance_payments: Amount paid by the customer in advance by cash (in 100s)
probability_of_full_payment: Probability of payment done in full by the customer to the bank
current_balance: Balance amount left in the account to make purchases (in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

1.1 Read the data and do exploratory data analysis. Describe the data briefly.

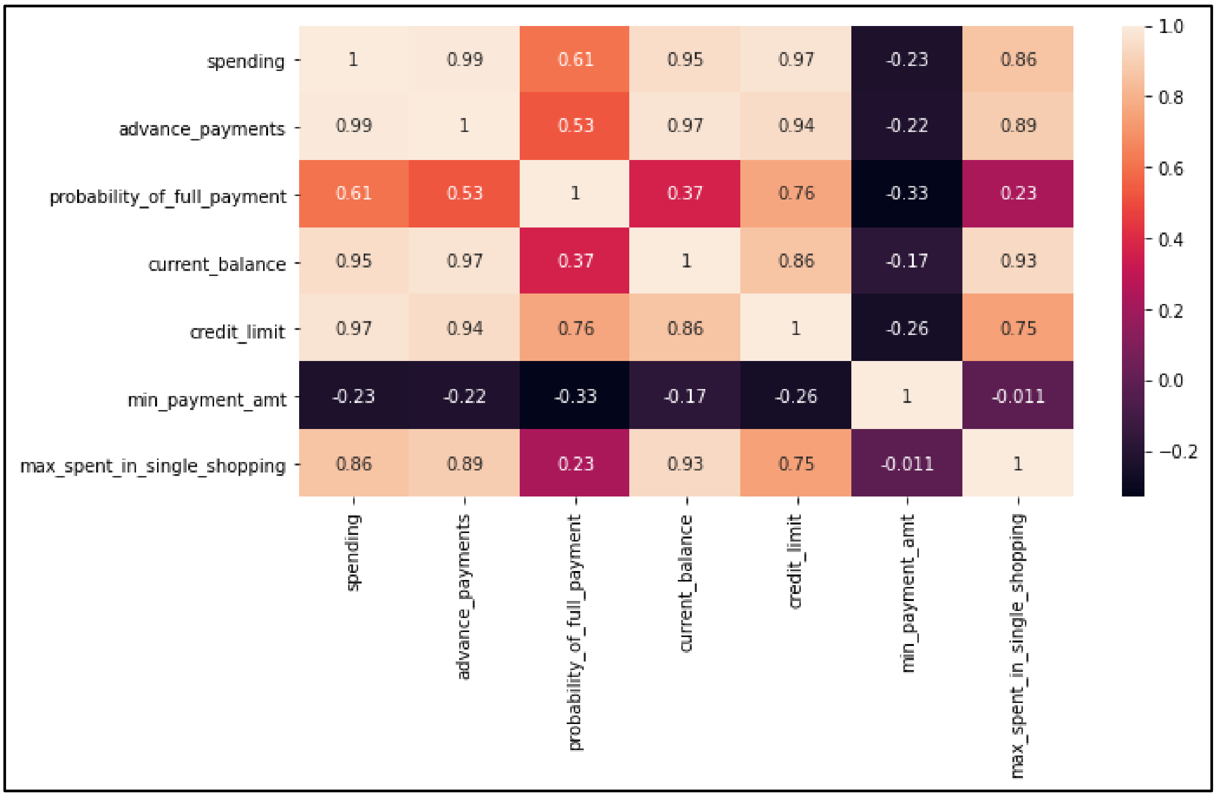
We read the data (csv file) using the **pd.read_csv()**, check the dimension size using **shape,** check the number of rows, datatypes and if it has any null values using info(). The describe() helps us with the 5-number analysis. And the duplicated() is used to find number of duplicate values in dataset which is 0. Following are some of the inferences after doing the EDA : -



This is the **pairplot** which shows us the distribution of a variable and its relationship with other variables in the dataframe.



The above **boxplot** shows the **distribution of data for a particular variable** and also the number of variables that have **outliers**. In the given dataset, **Probability_of_full_payment** and **Min_payment_amt** have outliers.



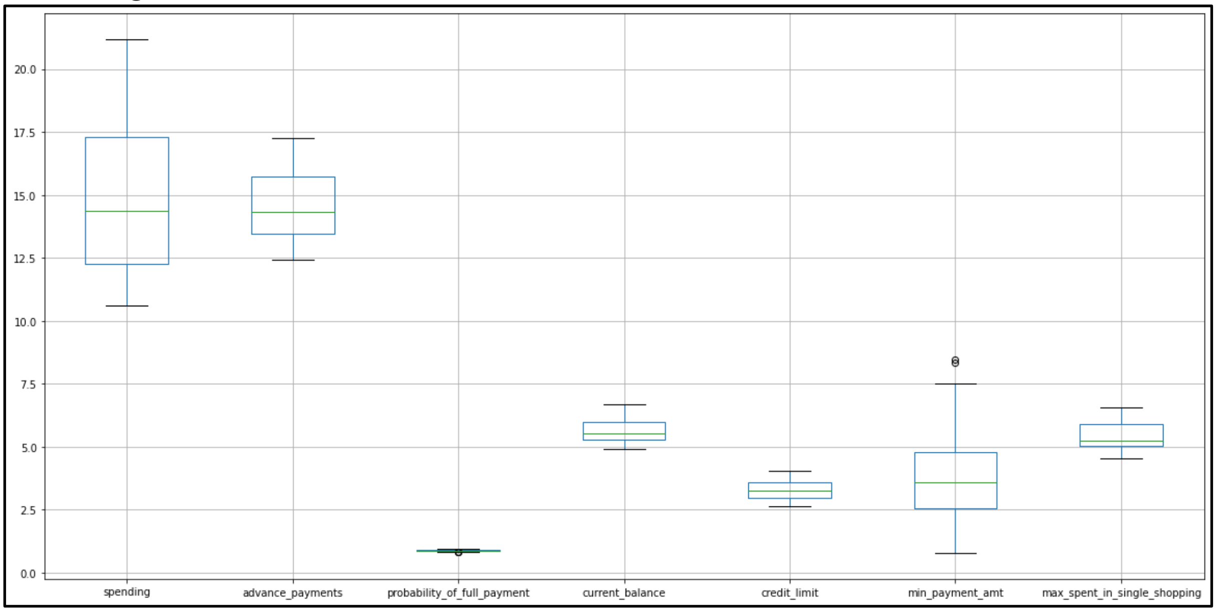
The above figure shows the **heatmap of correlation among different variables**. Here we can see that all the variables except min_payment_amt are highly correlated with each other.

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

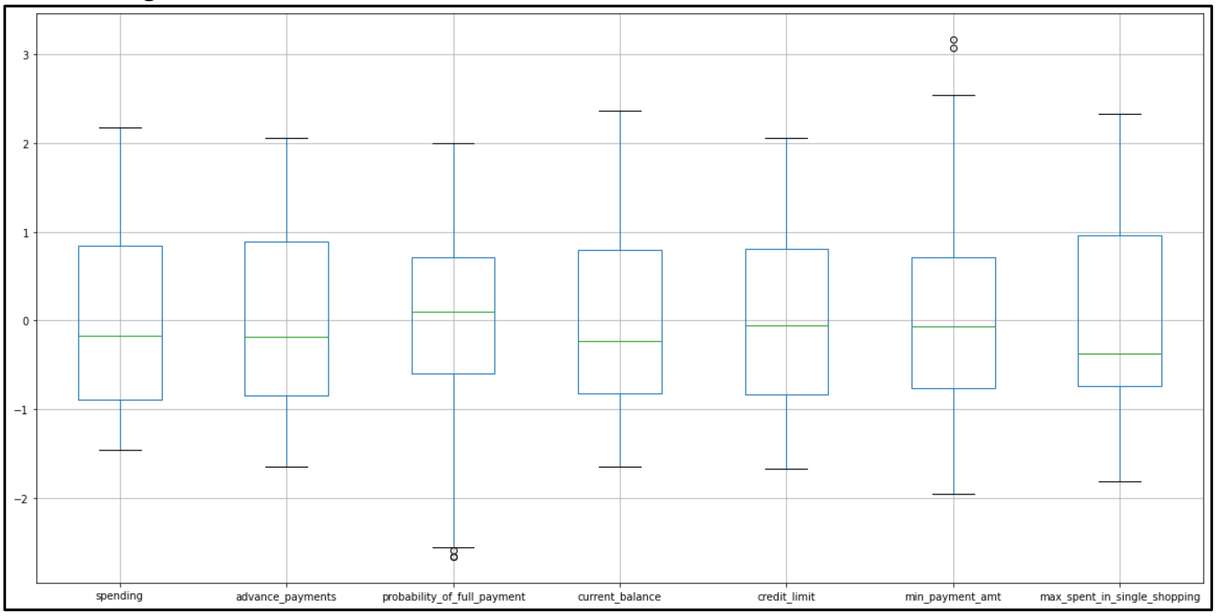
Yes, scaling becomes an important step while performing clustering. This is because the **clustering is done on distance-based computations** and since all the variables are not on a single scale, it may lead to incorrect results. For example, if there are variables of different natures like age and salary, there might be extreme differences since the groupings are distance based. **Standardization prevents variables with larger scales from dominating how clusters are defined. It allows all variables to be considered by the algorithm with equal importance. Bringing them on a single scale will make the computations and cluster**

formations more accurate. Hence, we use the Z-scaling using the StandardScaler(). The difference can be clearly seen using a boxplot.

Pre-Scaling



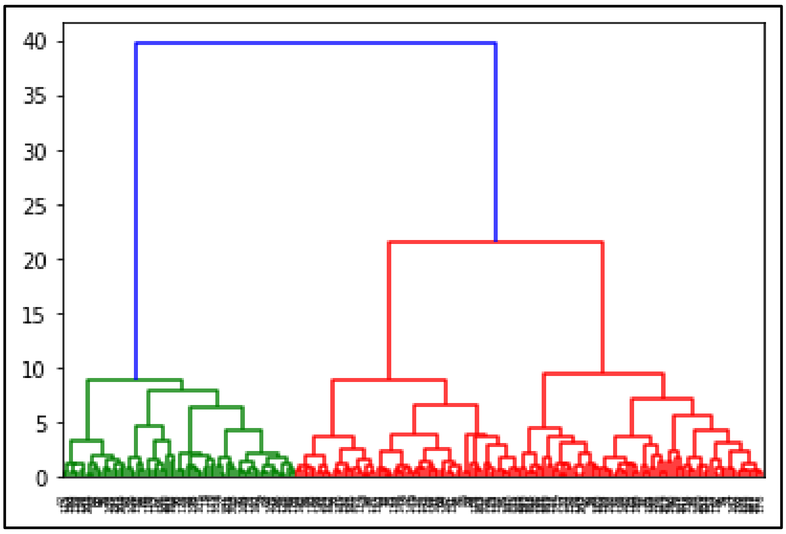
Post-Scaling



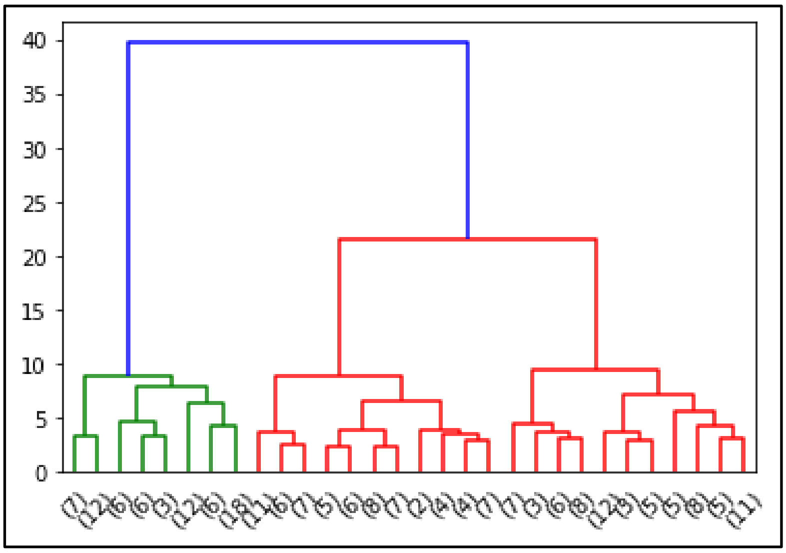
1.3 Apply hierarchical clustering to scaled data. Identify the number of optimum clusters using Dendrogram and briefly describe them

Hierarchical Clustering is a form of clustering where the groupings are done on similarities in patterns and distances. The agglomerative hierarchical clustering follows a bottom-up approach, here there are no assumptions made in selecting the number of clusters. The distance is calculated between points and clusters are formed. The distance between clusters are called as Linkages and we represent the Cluster chart using Dendrograms.

In our dataset, we have used Ward as the Linkage method.



Since the dendrogram looks very messy because of all the cluster combinations, we can prune and look at the last n clusters using the truncate_mode='lastp' and specifying a value for p which in our case is 30. Following is a visual post truncation : -

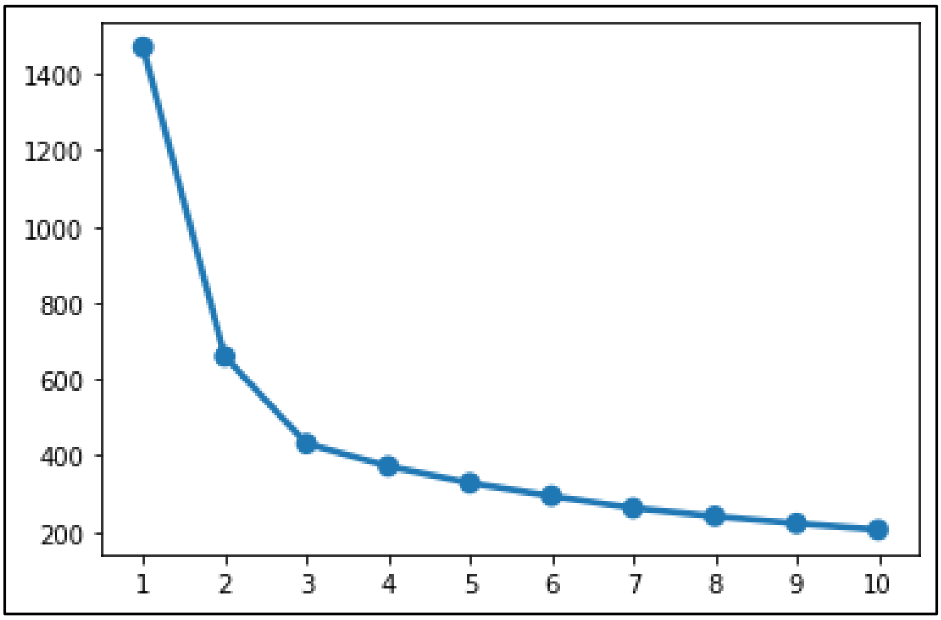


The optimum number of clusters is 2. When we see the cluster formations based on distance and putting 25 as threshold, we get 2 clusters which seems optimal in this case. For the first group of clusters, the average spending is 18,000 and for the second group, average spending is 13,000. Out of the 210 records, 70 falls in first group while 140 falls in the other.

1.4 Apply K-Means clustering on scaled data and determine optimum clusters. Apply elbow curve and silhouette score. Interpret the inferences from the model.

K-Means Clustering follows a partition-based approach where the number of clusters are specified and based on the distance, it starts clustering. Initially, we mention the number of clusters that we want and then calculate the inertia. Looking at the drop in inertia values, we can select the optimal number of clusters. We can visualize the same using the Elbow curve. The pointplot() from seaborn library is used to plot the Elbow Curve. We use KMeans from sklearn.cluster library.

Silhouette_score for the data set is used for measuring the mean of the Silhouette Coefficient for each sample belonging to different clusters. Silhouette_samples provides the Silhouette scores for each sample of different clusters. When **we consider 3 clusters as the optimum number**, the Silhouette score is 0.4. The silhouette score of 1 means that the clusters are very dense and nicely separated. The score of 0 means that clusters are overlapping. The score of less than 0 means that data belonging to clusters may be wrong/incorrect.



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

Cluster_Hierarchy	1	2
spending	18.371429	13.085571
advance_payments	16.145429	13.766214
probability_of_full_payment	0.884400	0.864298
current_balance	6.158171	5.363714
credit_limit	3.684629	3.045593
min_payment_amt	3.639157	3.730723
max_spent_in_single_shopping	6.017371	5.103421
count	70.000000	140.000000

Observations : - After profiling based on the Hierarchical Clustering, we find that customers in Cluster 1 tend to spend higher and also they make higher Advance_Payments. They have higher Current_balance in their accounts and tend to have higher Credit_limits. These customers tend to spend higher than the customers in cluster 2 in Single Shopping. These customers maybe the customers that maybe earning more and hence all these data reflect the same observations. Their count is half as compared to number of customers in cluster 2.

Recommendations :- More clients that belong to higher income group be should be focused, this can be done by offering higher rates of interests and giving extra bonus points for paying pending amount on time. Advance payments to be rewarded with cashbacks and bonuses. Newer categories like silver, gold and platinum cards and higher credit limits would result in more customers going for the upper categories.

Cluster_Kmeans	0	1	2
spending	11.856944	18.495373	14.437887
advance_payments	13.247778	16.203433	14.337746
probability_of_full_payment	0.848253	0.884210	0.881597
current_balance	5.231750	6.175687	5.514577
credit_limit	2.849542	3.697537	3.259225
min_payment_amt	4.742389	3.632373	2.707341
max_spent_in_single_shopping	5.101722	6.041701	5.120803
count	72.000000	67.000000	71.000000

Similarly like the Hierarchical clustering, the K-means profiling shows 3 different categories-customers spending around 11,000 , 14,000 and 18,000. The ones in the category of 18,000 make more advance payments than customers in other 2 clusters. Since all the parameters except Min_payment_amt are correlated , it shows a similar trend. Here also we should focus more on customers in cluster 1 and cluster 2 and offer some additional rewards and cashbacks to customers who pay in advance and offer higher credit limits so as to generate higher returns.

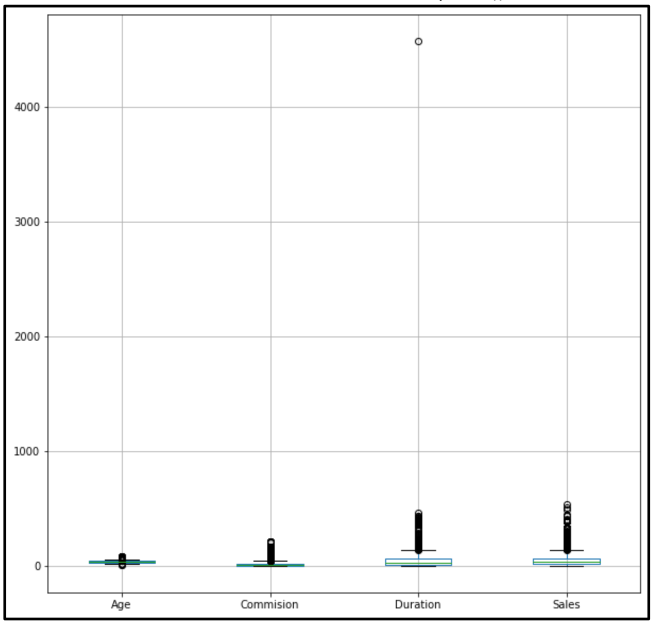
Problem 2: CART-RF-ANN

An Insurance firm providing tour insurance is facing higher claim frequency. The management decides to collect data from the past few years. You are assigned the task to make a model which predicts the claim status and provide recommendations to management. Use CART, RF & ANN and compare the models' performances in train and test sets.

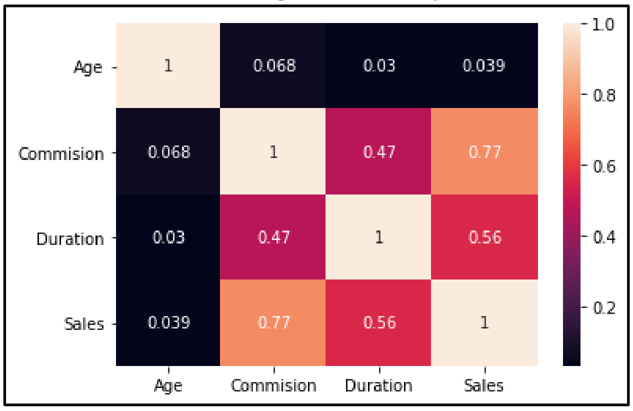
In the given scenario and dataset to support it, there are 10 variables including the claim status (dependant) variable which depends on the other 9 variables (independent). There are 3000 records and we have to design a model to predict the status.

2.1 Data Ingestion: Read the dataset. Do the descriptive statistics and do null value condition check, write an inference on it.

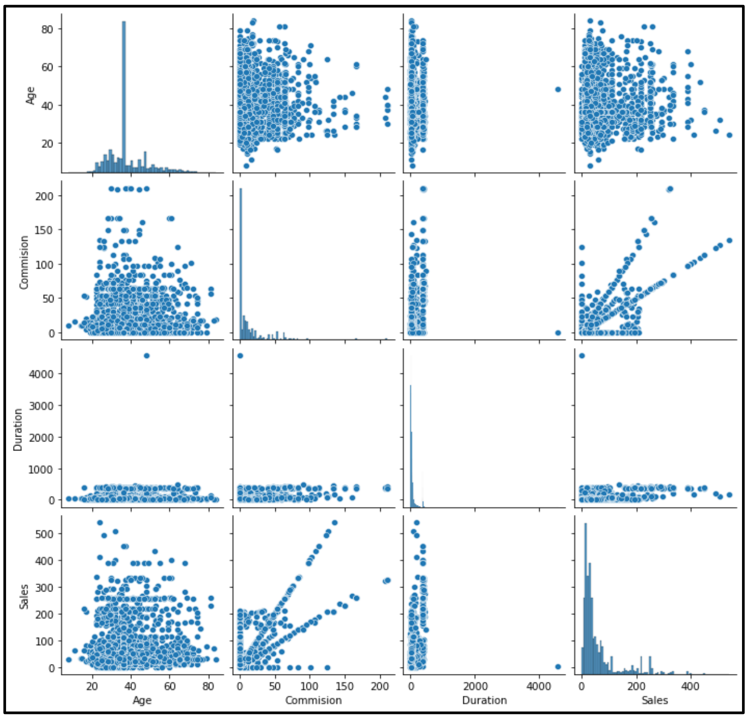
We read the data using read_csv() and then did the basic analysis using shape [dimensions- columns and rows] ,info() to read number of entries, datatypes, and no. of non-null values. Describe() for the 5-point summary and duplicated().sum() to find number of duplicate values which is 139 in our cases. The boxplot() is used to find the number of outliers.



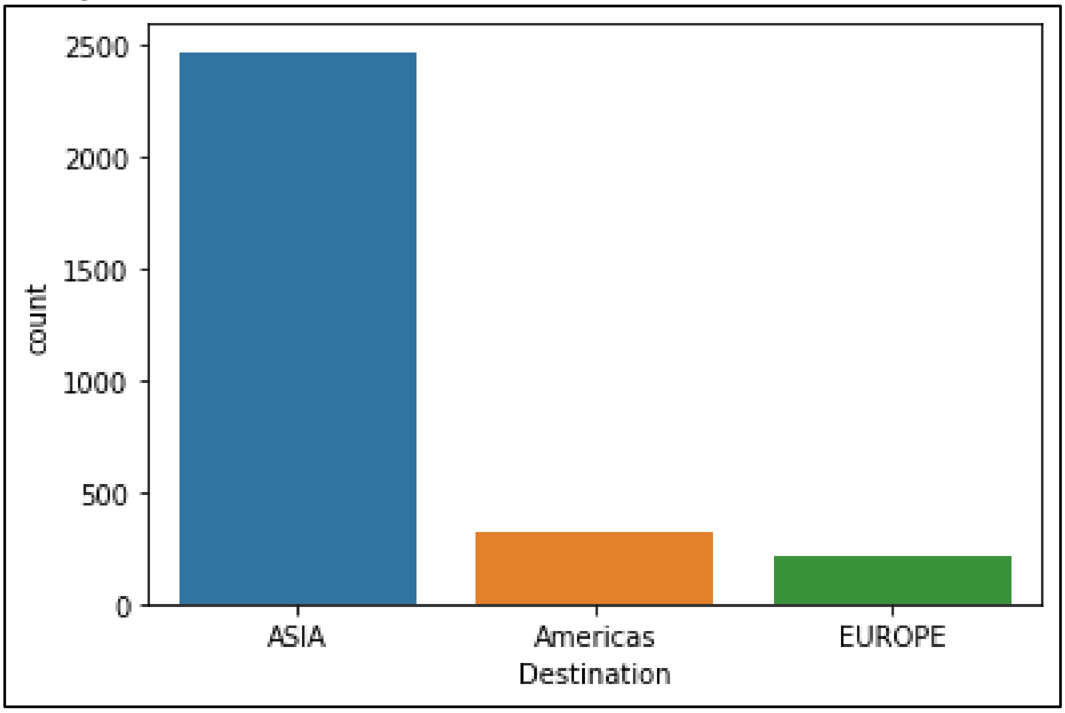
The heatmap here shows the correlation between different variables. The heatmap shows that there is no strong relationships between variables.



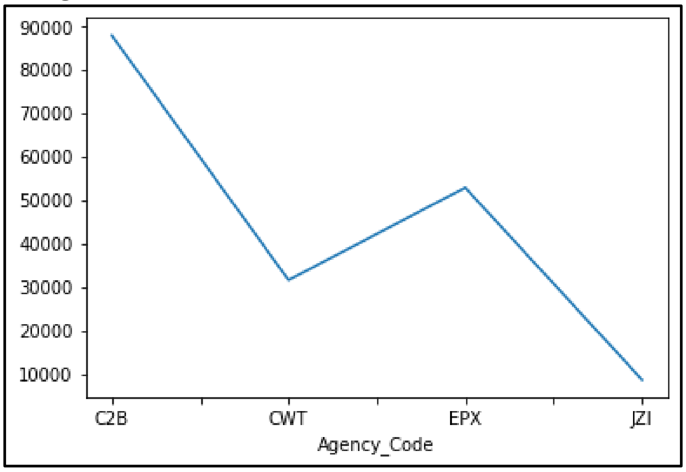
The pairplot from seaborn is used to see the relationships of different variables and the distribution of data of a particular variable.



The following graphs shows the number of people choosing destinations where Asia being the highest.



The following data shows the sales done per agency which shows C2B being highest and JZL being lowest.



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

Before splitting the data, we convert the variables of type object to categorical variables and used codes to represent them as numerical values. This is done using pd.Categorical().codes. After this step, we split the dependant and independent variables. This is done by pop() and drop() functions.
Finally, we can split the test and train data using train_test_split from sklearn.model_selection.
Train set is used to build a model that can be used to predict the data and test data is used to validate the model against the test set. The test data is generally smaller in size(20-30% data). We have taken 20% as test data and 80% as train data in our model.
Supervised learning is a learning mechanism in which the input and output is defined and different approaches can be used for the same, like Decision Tress (CART), Random Forest(large number of decision trees to overcome the greedy approach used by CART) and Neural Networks.

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neural_network import MLPClassifier
```

These are the different libraries used to build the models.
In case of CART, we use gini index as a criterion to build the model and depending on the gini index of the variable, the variable with maximum value is selected to build the model. In our dataset, the variable or feature with highest importance was Agency_Code after pruning the decision tree. We use Tree from sklearn to build a tree.

	Feature_Importance
Age	0.000000
Agency_Code	0.894961
Type	0.000000
Commision	0.000000
Channel	0.000000
Duration	0.000000
Sales	0.000000
Product Name	0.105039
Destination	0.000000

Different functions like predict and proba are used to predict the variables and find the probability in train and test sets.

While building a Random Forest model (overcoming the greedy approach used in CART), we do Bootstraping and use random subsets of variables and data to fit a forest. We then combine the classifications/predictions from different individual tress to get optimized predictions and then use voting for categorical variables and averaging for continuous variables. We can use different combinations using the GridSearchCV() and find the optimized values to build a model.
And finally, neural networks are based on the concept of neurons which function similar to human brain. Input layers give inputs and feed weights to hidden layer, the summation is used to compute values and similarly passed on to further layers and then a result is generated on output layer. If the value doesn't match to the desired values, re-computation is done by adjusting the weights.

Following are the values that are models considered for generating outputs: -

CART

```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                       max_depth=30, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=100, min_samples_split=1000,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=1, splitter='best')
```

Random Forest

```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=12, max_features=7,
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=10, min_samples_split=80,
                       min_weight_fraction_leaf=0.0, n_estimators=101,
                       n_jobs=None, oob_score=False, random_state=1, verbose=0,
                       warm_start=False)
```

ANN

```
MLPClassifier(activation='relu', alpha=0.0001, batch_size='auto', beta_1=0.9,
              beta_2=0.999, early_stopping=False, epsilon=1e-08,
              hidden_layer_sizes=(100, 100, 100), learning_rate='constant',
              learning_rate_init=0.001, max_fun=15000, max_iter=50000,
              momentum=0.9, n_iter_no_change=10, nesterovs_momentum=True,
              power_t=0.5, random_state=None, shuffle=True, solver='adam',
              tol=0.01, validation_fraction=0.1, verbose=False,
              warm_start=False)
```

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

Accuracy is the correctness of the model prediction. **Confusion Matrix** gives the number of True Positives, False Positives, False Negatives and True Negatives. **Classification Report** gives the precision, recall, accuracy,f1-score and support for the model.
AUC is Area Under the Curve (higher the area, better the model) and ROC is Receiver Operating Characteristic curve which is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters: True Positive Rate. False Positive Rate.

CART

Accuracy: -

```
Accuracy for Training data : 0.755
Accuracy for Testing data : 0.7666666666666667
```

Confusion Matrix: -

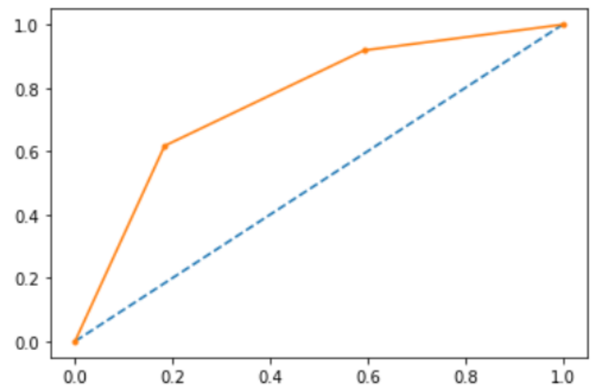
```
Confusion Matrix for Training Data
[[1356  305]
 [ 283  456]]


Confusion Matrix for Testing Data
[[356  59]
 [ 81 104]]
```

Classification Report: -

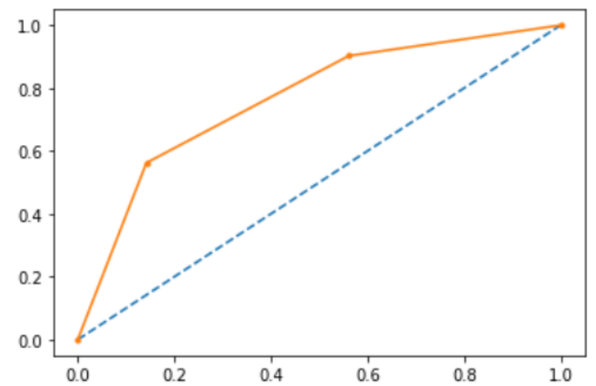
```
Classification Report for Training Data
              precision    recall  f1-score   support

           0       0.83      0.82      0.82      1661
           1       0.60      0.62      0.61       739

    accuracy                           0.76      2400
   macro avg       0.71      0.72      0.71      2400
weighted avg       0.76      0.76      0.76      2400


Classification Report for Testing Data
              precision    recall  f1-score   support

           0       0.81      0.86      0.84       415
           1       0.64      0.56      0.60       185

    accuracy                           0.77       600
   macro avg       0.73      0.71      0.72       600
weighted avg       0.76      0.77      0.76       600
```

ROC and AUC: -

AUC for Train set: 0.761



AUC for Test set: 0.764



Random Forest

Confusion Matrix: -

```
Confusion Matrix for Training Data
[[1487  174]
 [ 286  453]]


Confusion Matrix for Testing Data
[[377  38]
 [ 91  94]]
```

Classification Report: -

```
Classification Report for Training Data
              precision    recall  f1-score   support

           0       0.84      0.90      0.87      1661
           1       0.72      0.61      0.66       739

    accuracy                           0.81      2400
   macro avg       0.78      0.75      0.76      2400
weighted avg       0.80      0.81      0.80      2400


Classification Report for Testing Data
              precision    recall  f1-score   support

           0       0.81      0.91      0.85       415
           1       0.71      0.51      0.59       185

    accuracy                           0.79       600
   macro avg       0.76      0.71      0.72       600
weighted avg       0.78      0.79      0.77       600
```

Accuracy: -

Train Set-

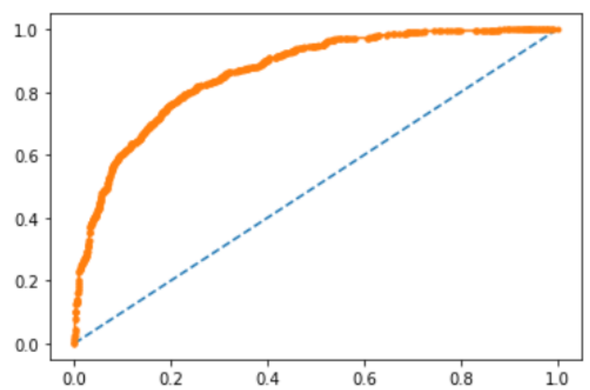
```
accuracy                                      0.81
```

Test Set-

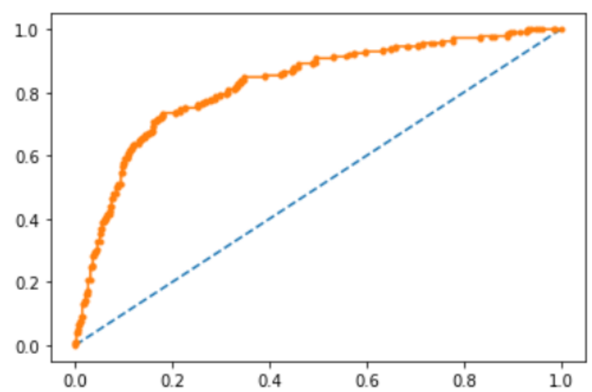
```
accuracy                                      0.79
```

ROC and AUC: -

AUC for Train set: 0.864



AUC for Test set: 0.825



ANN

Accuracy: -

Train Set-

```
accuracy                              0.78
```

Test Set-

```
accuracy                              0.79
```

Confusion Matrix: -

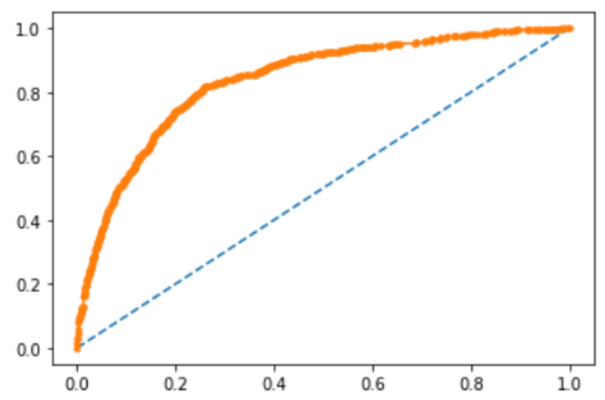
```
Confusion Matrix for Training Data
[[1392  250]
 [ 268  490]]


Confusion Matrix for Testing Data
[[361  73]
 [ 56 110]]
```

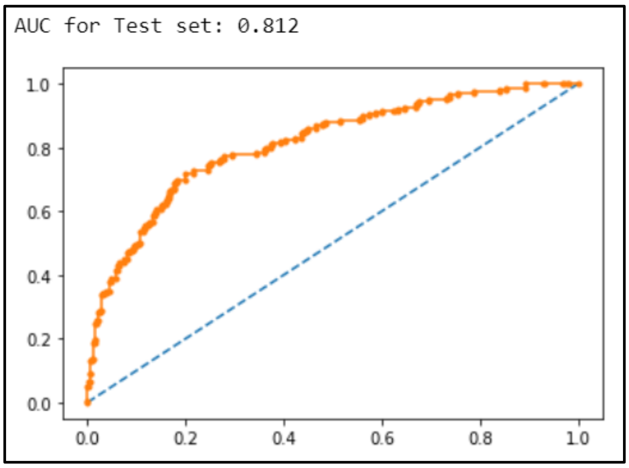
Classification Report: -

```
Classification Report for Training Data
              precision    recall  f1-score   support

           0       0.84      0.85      0.84      1642
           1       0.66      0.65      0.65       758

    accuracy                           0.78      2400
   macro avg       0.75      0.75      0.75      2400
weighted avg       0.78      0.78      0.78      2400



Classification Report for Testing Data
              precision    recall  f1-score   support

           0       0.87      0.83      0.85       434
           1       0.60      0.66      0.63       166

    accuracy                           0.79       600
   macro avg       0.73      0.75      0.74       600
weighted avg       0.79      0.79      0.79       600
```

ROC and AUC: -

AUC for Train set: 0.838





Based on the Model Performance measures, we can clearly see that CART doesn't perform as good as ANN and Random Forest because of the greedy approach. Random Forest performs really well in terms of accuracy and similarly, ANN can be trained better with more iterations. All the models are valid models i.e neither underfit, nor overfit.

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

Based on the model performance measure, we got the following output: -
Accuracy: TP+TN/TP+FP+TN+FN
Precision: TP/TP+TN
Sensitivity: TP/TP+FN
Specificity: TN/TN+FP

Parameter	Type	CART	Random Forest	ANN
Accuracy	Train	75.50%	81.00%	78.00%
	Test	76.67%	79.00%	79.00%
Precision for 0	Train	83.00%	84.00%	84.00%
	Test	81.00%	81.00%	87.00%
Precision for 1	Train	60.00%	72.00%	66.00%
	Test	64.00%	71.00%	60.00%
Recall for 0/Sensitivity	Train	82.00%	90.00%	75.00%
	Test	86.00%	91.00%	83.00%
Recall for 1/Specificity	Train	62.00%	61.00%	65.00%
	Test	56.00%	51.00%	66.00%
AUC	Train	76.10%	86.40%	83.80%
	Test	76.40%	82.50%	81.20%

In terms of Accuracy, Random Forest is the best followed by ANN and CART. Similarly, for Precision, Random Forest performs better than ANN and CART. Sensitivity is high in Random Forest and Specificity is high in ANN. Both, Random Forest and ANN have a significant area under the curve.
Looking at the numbers, we can say that our Random Forest model is performing well than the two models, ANN can be trained and with some iterations it can also perform similarly well. The CART model, because of greedy approach doesn't perform rationally which impacts its performance.

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

Based on the analysis, we found that the variables Commission and Sales were highly correlated, C2B outperformed all the other agencies and most of the customers preferred Asia as a destination, so we can assume that C2B provided better packages to customer travelling to Asia and might be a bit reasonable. So considering the data of past few years (3000 records), we had split it in train and test data and built 3 models on it- CART, Random Forest and ANN.

The CART model had an accuracy of predicting Claim Status on Training set is 75.5% and predicting on Test set is 76.7%. This model gives more weightage to the Agency_code as a feature considering the criteria as Gini Index. This model has not performed as good as the other two models as it follows a greedy approach.

The Random Forest model has performed the best since its accuracy was 81% and 79% on Train and Test data respectively. This model uses voting and averaging mechanisms and initially does bootstrapping to consider the input data. Then GridSearchCV is used to take multiple conditions and select the best combination.

The ANN model which functions as a neuron based on the concept of human brain does better than CART model, but it requires some iterations and inputs to get the best/optimal results. It has an accuracy of 78% and 79% on train and test data sets respectively.

So, we can implement the Random Forest approach for better predictions and get the best results which would help the insurance company save money.