APPLIED MACHINE LEARNING AND DATA MINING (AMLDM) COURSEWORK REPORT

Do not write your name on your work unless your lecturer has explicitly told you to do so.

Student ID number	Title of degree studying	**Level/Year**
2200918	Bachelor of Science (Honours) Data Science and Analytics	1

Short unit name:	M32365 - AMLDM				Due date: 20 February 2023		Deadline: 20 February
Full unit name:	APPLIED MACHINE LEARNING AND DATA MINING (AMLDM)						
Unit lecturer name:	Akshay Sachdeva					Group: *(if applicable)*	
Additional items e.g. CD/disk/USB:	Yes		No	✓	**Details:**		

All additional items should be clearly labelled with ID number and unit name and securely attached to your work.

Candidates are reminded that the following are defined as Assessment Offences and will be dealt with in accordance with the University's Code of Student Discipline:

a) Any attempt to complete an assessment by means considered to be unfair;
b) Plagiarism, which the University defines as the incorporation by a student in work for assessment of material which is not their own, in the sense that all or a substantial part of the work has been copied without any adequate attempt at attribution or has been incorporated as if it were the student's own work when in fact it is wholly or substantially the work of another person or persons.

Please note: Group **coursework** will be filed under the first Student ID number at the top of the list. Ensure you know all **group member's ID numbers.**

NB: **Coursework not collected** will be disposed of six months **after** the hand-in date.

FOR OFFICIAL USE ONLY

Date received/Office stamp	Provisional mark % / Comments
Administration Office	Academic Staff Member

1. Task A – Supervised learning

In supervised learning, machines are trained using labeled data sets. Models are trained over time using labeled data to classify a new data point. Supervised learning can be broadly classified into two types: classification and regression. The two types of regression discussed in the coursework are simple linear regression and multiple linear regression.

Two tasks were given under supervised learning. The two projects I chose for A.1.2 were to identify gender based on data pertaining to measurement of facial features and weather analysis to predict if it is going to rain next day or not. For each project I used four classification models: Logistic Regression, Random Forest, Decision Tree and Support Vector.

Logistic Regression

Logistic regression is a statistical technique for analyzing a dataset in which one or more independent variables influence the outcome. It is used to predict one of two possible outcomes (e.g., yes/no, success/failure) in binary classification problems. The logistic regression model is an equation that calculates the likelihood of the positive class (e.g., yes, success) based on the input features, which are then converted into a value between 0 and 1 using the logistic function. Following that, the predicted probabilities are thresholded to produce the final binary predictions.

Random Forest Classifier

The Random Forest Classifier is an ensemble learning method for classification problems. It works by constructing multiple decision trees from a random subset of the data and features and then aggregating their predictions to make the final prediction for new instances. The aggregation process is usually performed by majority voting for binary classification problems or by averaging for multiclass classification problems.

Decision Tree Classifier

The Random Forest Classifier is a classification problem ensemble learning method. It works by building multiple decision trees from a random subset of the data and features and then combining their predictions to make the final prediction for new instances. Aggregation is typically accomplished through majority voting for binary classification problems or averaging for multiclass classification problems.

SVC (Support Vector Classifier): linear

SVC Linear (SVC Linear) is a type of Support Vector Classifier (SVC) that is used for linear binary classification problems. The algorithm finds the best hyperplane by drawing a straight line between the two classes. The SVC Linear algorithm attempts to maximize the margin between the two classes by locating the line (hyperplane) that is closest to both classes while also ensuring that the line is equidistant from the support vectors, which are the closest data points from each class.

1.2 Building Classification Model

1.2.1 Gender data

```
# First 5 rows
gender.head()
```

	long_hair	forehead_width_cm	forehead_height_cm	nose_wide	nose_long	lips_thin	distance_nose_to_lip_long	gender
0	1	11.8	6.1	1	0	1	1	Male
1	0	14.0	5.4	0	0	1	0	Female
2	0	11.8	6.3	1	1	1	1	Male
3	0	14.4	6.1	0	1	1	1	Male
4	1	13.5	5.9	0	0	0	0	Female

Summary of Gender Data

The aim of gender data is to identify gender based on data pertaining to measurement of facial features. Gender data contain 5001 rows and 8 columns. There are 7 columns (numerical) and 1 column (categorical). Gender data has no missing values across all columns. There are duplicated values.

```
gender.info()
```

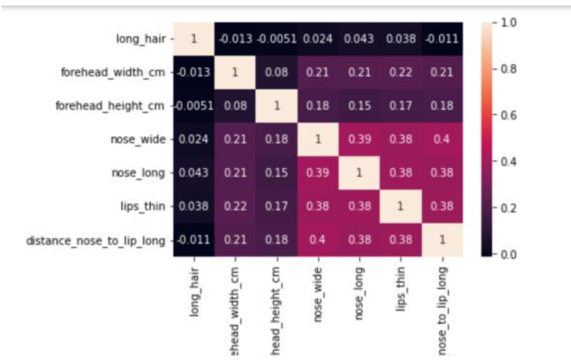
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5001 entries, 0 to 5000
Data columns (total 8 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   long_hair                  5001 non-null   int64
 1   forehead_width_cm          5001 non-null   float64
 2   forehead_height_cm         5001 non-null   float64
 3   nose_wide                  5001 non-null   int64
 4   nose_long                  5001 non-null   int64
 5   lips_thin                  5001 non-null   int64
 6   distance_nose_to_lip_long  5001 non-null   int64
 7   gender                     5001 non-null   object
dtypes: float64(2), int64(5), object(1)
```

```
gender.isna().sum()
```

```
long_hair                    0
forehead_width_cm            0
forehead_height_cm           0
nose_wide                    0
nose_long                    0
lips_thin                    0
distance_nose_to_lip_long    0
gender                       0
dtype: int64
```

Data preparation steps carried out for building the model are:

- Remove 1,768 duplicated rows.
- Label Encoder is used to transform [‘gender’] column values to 0 or 1 values only.
- MinMax Scaler is used to transform predictor attribute X from 0 to 1 range.



All columns except gender have correlation.

```python
X = gender.iloc[:,:-1] # All except last column(Gender)
y = gender.iloc[:,-1] # Gender only
```

Predictor attribute X, all except 'gender' which is the target attribute y to predict whether it will rain on the next day or not.

```python
# trasnform gender columns values to 0 or 1 only
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
y = le.fit_transform(y)
le_g = pd.DataFrame(y)
```

```python
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1, shuffle =True)
print('X_train shape is ' , X_train.shape)
print('X_test shape is ' , X_test.shape)
print('y_train shape is ' , y_train.shape)
print('y_test shape is ' , y_test.shape)
```

```
X_train shape is  (3500, 7)
X_test shape is  (1501, 7)
y_train shape is  (3500,)
y_test shape is  (1501,)
```

```python
# transform all columns values except gender column to 0 to 1 range only
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler(copy=True, feature_range=(0, 1))
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

The predictor attribute X is feature scaled from 0 to 1 and the target attribute y is encoded to 0 or 1. Both predictor attribute X and target attribute y are then split in two by the train test model in a 70:30% ratio.

Results

	Model	Accuracy Score
0	Logistic Regression	95.57%
2	Decision Tree	95.15%
1	Random Forest	94.85%
3	SVC Linear	94.33%

	Model	Run Time
2	Decision Tree	0.053s
0	Logistic Regression	0.106s
3	SVC Linear	0.139s
1	Random Forest	0.316s

Logistic Regression

	precision	recall	f1-score	support
0	0.95	0.94	0.95	425
1	0.96	0.97	0.96	545
accuracy			0.96	970
macro avg	0.96	0.95	0.95	970
weighted avg	0.96	0.96	0.96	970

Decision Tree

	precision	recall	f1-score	support
0	0.94	0.96	0.95	457
1	0.96	0.94	0.95	513
accuracy			0.95	970
macro avg	0.95	0.95	0.95	970
weighted avg	0.95	0.95	0.95	970

Random Forest

	precision	recall	f1-score	support
0	0.95	0.94	0.95	457
1	0.95	0.95	0.95	513
accuracy			0.95	970
macro avg	0.95	0.95	0.95	970
weighted avg	0.95	0.95	0.95	970

SVC

	precision	recall	f1-score	support
Female	0.95	0.93	0.94	457
Male	0.94	0.96	0.95	513
accuracy			0.94	970
macro avg	0.94	0.94	0.94	970
weighted avg	0.94	0.94	0.94	970

The logistic regression model has the greatest accuracy score (95.57%). Then there was a decision tree classifier with an accuracy score of 95.15%. Random Forest comes in third, with a slightly lower accuracy score of 94.85%. SVC linear has the lowest accuracy score of 94.33%.

Although the logistic regression execution time of 0.106s is not the fastest of the three classifications, it is only slightly slower than the decision tree's fastest speed of 0.053s. I will recommend logistic regression as the best model because of its overall high accuracy, precision, and recall scores.

2.2 Weather data

```
# First 5 rows
weather.head()
```

	Date	Location	MinTemp	MaxTemp	Rainfall	Evaporation	Sunshine	WindGustDir	WindGustSpeed	WindDir9am	...	Humidity9am
0	2008-12-01	Albury	13.4	22.9	0.6	NaN	NaN	W	44.0	W	...	71.
1	2008-12-02	Albury	7.4	25.1	0.0	NaN	NaN	WNW	44.0	NNW	...	44.
2	2008-12-03	Albury	12.9	25.7	0.0	NaN	NaN	WSW	46.0	W	...	38.
3	2008-12-04	Albury	9.2	28.0	0.0	NaN	NaN	NE	24.0	SE	...	45.
4	2008-12-05	Albury	17.5	32.3	1.0	NaN	NaN	W	41.0	ENE	...	82.

5 rows × 23 columns

Summary of Weather data

The data's goal is to predict whether or not it will rain the next day. It is made up of 145,460 rows and 23 columns. Except for the Date and Location columns, there were many missing values.

```
weather.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 145460 entries, 0 to 145459
Data columns (total 23 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   Date           145460 non-null  object
 1   Location       145460 non-null  object
 2   MinTemp        143975 non-null  float64
 3   MaxTemp        144199 non-null  float64
 4   Rainfall       142199 non-null  float64
 5   Evaporation    82670 non-null   float64
 6   Sunshine       75625 non-null   float64
 7   WindGustDir    135134 non-null  object
 8   WindGustSpeed  135197 non-null  float64
 9   WindDir9am     134894 non-null  object
 10  WindDir3pm     141232 non-null  object
 11  WindSpeed9am   143693 non-null  float64
 12  WindSpeed3pm   142398 non-null  float64
 13  Humidity9am    142806 non-null  float64
```

```
#Check for any missing value
weather.isnull().sum()
```

```
Date                 0
Location             0
MinTemp           1485
MaxTemp           1261
Rainfall          3261
Evaporation      62790
Sunshine         69835
WindGustDir      10326
WindGustSpeed    10263
WindDir9am       10566
WindDir3pm        4228
WindSpeed9am      1767
WindSpeed3pm      3062
Humidity9am       2654
Humidity3pm       4507
Pressure9am      15065
Pressure3pm      15028
Cloud9am         55888
Cloud3pm         59358
Temp9am           1767
Temp3pm           3609
RainToday         3261
RainTomorrow      3267
```

Data preparation steps carried out for building the model are:

- Separated the columns into two lists: categorical features (Data type object) and numerical features (Data type float).
- Fill in the missing values of categorical features using the column's mode.
- Fill in missing numerical features values with the column's mean.
- The 'Year', 'Month', and 'Day' columns were derived from the Date column, the 'date' column is then removed from the weather data.

- Label Encoder was used to convert categorical features values to numeric values.
- Standard Scaler was used to transform predictor attribute x to 0 to 1 value only.

```
x = weather.drop(['RainTomorrow'],axis=1).values # Predictor value
y = weather['RainTomorrow'].values # Target value
```

Predictor attribute x, all columns except 'RainTomorrow,' which is the target attribute y to predict whether it will rain the next day.

Results

	Model	Accuracy Score
1	Random Forest	85.60%
0	Logistic Regression	84.23%
3	SVC Linear	84.17%
2	Decision Tree	83.78%

	Model	Run Time
0	Logistic Regression	0.408s
2	Decision Tree	1.002s
3	SVC Linear	1468.513s
1	Random Forest	21.787s

Logistic Regression	precision	recall	f1-score	support
0	0.87	0.95	0.90	34057
1	0.71	0.48	0.57	9581
accuracy			0.84	43638
macro avg	0.79	0.71	0.74	43638
weighted avg	0.83	0.84	0.83	43638

Decision Tree	precision	recall	f1-score	support
0	0.86	0.94	0.90	34057
1	0.69	0.47	0.56	9581
accuracy			0.84	43638
macro avg	0.78	0.70	0.73	43638
weighted avg	0.83	0.84	0.83	43638

Random Forest	precision	recall	f1-score	support
0	0.87	0.96	0.91	34057
1	0.76	0.50	0.60	9581
accuracy			0.86	43638
macro avg	0.82	0.73	0.76	43638
weighted avg	0.85	0.86	0.84	43638

SVC Linear	precision	recall	f1-score	support
0	0.86	0.95	0.90	34057
1	0.73	0.44	0.55	9581
accuracy			0.84	43638
macro avg	0.79	0.70	0.73	43638
weighted avg	0.83	0.84	0.83	43638

I will recommend the Random Forest Classifier Model as the best model because it has the highest accuracy score of 85.49%. Then came Logistic Regression, with an accuracy score of 84.23% and a slightly lower score of 84.17% for SVC linear. Decision Tree Classifier had the lowest accuracy score of 83.78%.

Although the Random Forest model had the highest precision and recall score, its recall score for predicting rain days is only 0.50. This means that the model correctly predicted 50% of the rainy days in the dataset.

3. Task A.2 - Building Regression Model

3.1 Regression Technique

- **Simple Linear Regression**
  Simple Linear Regression is a statistical method used to examine the relationship between two continuous variables where one variable (the dependent variable) is predicted from the values of the other variable (the independent variable).

- **Multiple Linear Regression**
  Multiple Linear Regression, on the other hand, involves more than one independent variable to predict the value of a dependent variable. It is used to model the relationship between two or more independent variables and a dependent variable.

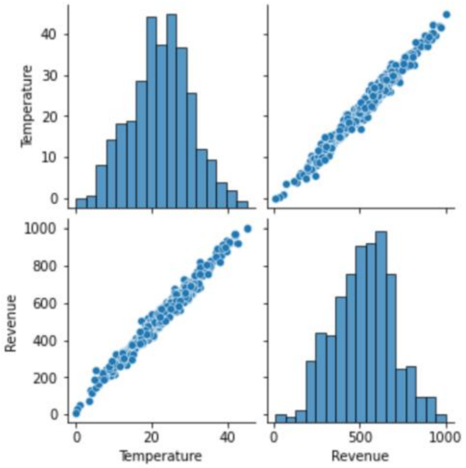
3.2.1 Ice Cream Data

```
# First 5 rows
icecream.head()
```

	Temperature	Revenue
0	24.566884	534.799028
1	26.005191	625.190122
2	27.790554	660.632289
3	20.595335	487.706960
4	11.503498	316.240194

Summary of IceCream data

The goal of IceCream data is to forecast daily revenue (in USD) based on the outside temperature (°C). It has 500 rows and 2 columns. IceCream data contains no missing values.



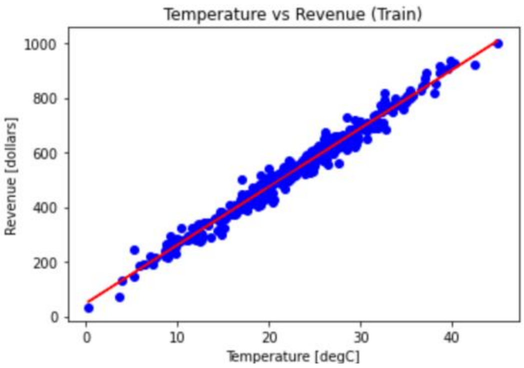
	Temperature	Revenue
Temperature	1.000000	0.989802
Revenue	0.989802	1.000000

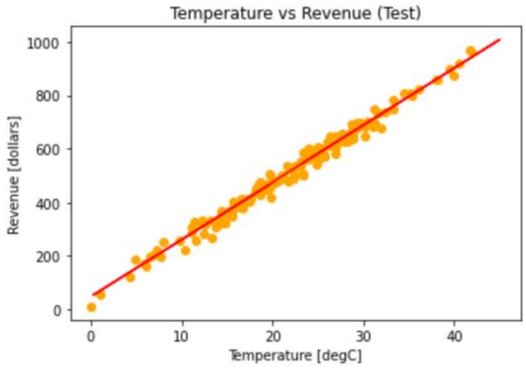
As shown in the graph above, there is a strong positive correlation between temperature and revenue. Furthermore, we can deduce that as the temperature rises, so will revenue.

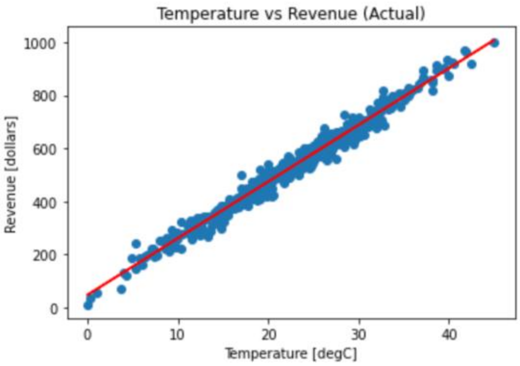
```
X = icecream.iloc[:,0].values.reshape(-1,1)
y = icecream.iloc[:,-1].values
```

Since this IceCream dataset only has two columns, the predictor attribute X, temperature, will be used to predict the daily revenue generated, which is the target attribute y.

Results








```
Mean Absolute Error: 17.976807512195343
Mean Squared Error: 503.3547092334282
Root Mean Squared Error: 22.43556794987433
R2 Score: 0.9854012979456143
```

The results show little difference between the train, test, and actual datasets. Temperature and revenue continue to have a strong positive correlation. The R^2 score, also known as the coefficient of determination, quantifies how well a regression model predicts the dependent variable based on the independent variables. The better the model fits the data, the higher the R^2 score. This dataset has an R^2score of 0.98, which is very close to one, indicating that it is a perfect model.

3.2.2 Car Price Data

```
# First 5 rows
car.head()
```

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	...	enginesize	fu
0	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	front	88.6	...	130	
1	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	front	88.6	...	130	
2	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	front	94.5	...	152	
3	4	2	audi 100 ls	gas	std	four	sedan	fwd	front	99.8	...	109	
4	5	2	audi 100ls	gas	std	four	sedan	4wd	front	99.4	...	136	

5 rows × 26 columns

Summary of CarPrice data

The goal of CarPrice data is to create a prediction model that can explain how car prices vary depending on their specifications. It is made up of 205 rows and 26 columns. There are no duplicated or missing values.

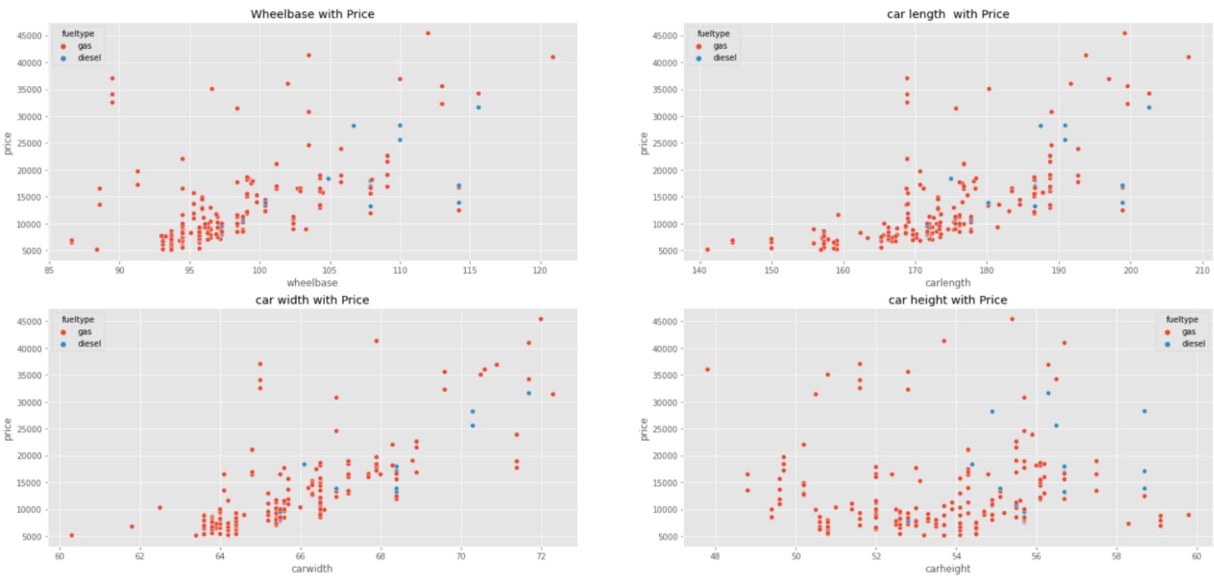
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 26 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   car_ID            205 non-null    int64
 1   symboling         205 non-null    int64
 2   CarName           205 non-null    object
 3   fueltype          205 non-null    object
 4   aspiration        205 non-null    object
 5   doornumber        205 non-null    object
 6   carbody           205 non-null    object
 7   drivewheel        205 non-null    object
 8   enginelocation    205 non-null    object
 9   wheelbase         205 non-null    float64
 10  carlength         205 non-null    float64
 11  carwidth          205 non-null    float64
 12  carheight         205 non-null    float64
 13  curbweight        205 non-null    int64
 14  enginetype        205 non-null    object
```

```
car.isnull().sum()

car_ID              0
symboling           0
CarName             0
fueltype            0
aspiration          0
doornumber          0
carbody             0
drivewheel          0
enginelocation      0
wheelbase           0
carlength           0
carwidth            0
carheight           0
curbweight          0
enginetype          0
cylindernumber      0
enginesize          0
fuelsystem          0
boreratio           0
stroke              0
compressionratio    0
horsepower          0
peakrpm             0
citympg             0
highwaympg          0
price               0
```

Data preparation steps carried out for building the model are:

- The car ID column was removed.
- Label Encoder was used to convert categorical features to a numeric value.
- All columns except price were transformed to 0 to 1 value using the standard scaler.

Exploratory Data Analysis



According to the graphs above, prices rise when the relevant features rise as well.

```
car.corr()['price'].sort_values(ascending=False)>=0.5
```

```
price               True
enginesize          True
curbweight          True
horsepower          True
carwidth            True
carlength           True
drivewheel          True
wheelbase           True
boreratio           True
fuelsystem          True
enginelocation     False
aspiration         False
carheight          False
stroke             False
compressionratio   False
enginetype         False
cylindernumber     False
doornumber         False
symboling          False
```

```
#Defined X value and y value , and split the data train
X = car.drop(columns="price")          #all except price , predictor attribute
y = car["price"]     # y = price , price column only , target attribute
```

Only columns in red box will be selected for Predictor attribute X, which is based on car features to predict price which is target attriubte y.

Results

```
from sklearn import metrics
from sklearn.metrics import r2_score
print("R square (R²) score is :",r2_score(y_test, y_pred)*100,'%')
print("MAE", metrics.mean_absolute_error(y_test,y_pred))
print("RMSE", np.sqrt(metrics.mean_squared_error(y_test,y_pred)))

#To retrieve the intercept:
print('Intercept: \n', reg.intercept_)
#For retrieving the slope:
print('Coefficients: \n', reg.coef_)
```

```
R square (R²) score is : 82.19678883899657 %
MAE 2241.236397093437
RMSE 3278.4003626692274
Intercept: 
 13188.785115274359
Coefficients: 
 [-381.55170111 -272.61931251  -60.00285415  777.87465479 -381.48146321
 1631.42243674 1756.69397444 1264.82876481 3554.0608345 ]
```

As R^2 score evaluation shows how well the data fits the regression model, we have a score of 82.19% for this model, but other metrics such as Mean Absolute Error and Root Mean Squared Error had very high values. This could be due to several factors, including a poorly chosen model, insufficient data, or data outliers. As a result, we can conclude that the model only works well to a certain extent.

4. Task B – Unsupervised Learning

4.1 Cluster Analysis

K-Means clustering

K-means is a centroid-based or distance-based algorithm that uses distances to assign a point to a cluster. Each cluster in K-Means is associated with a centroid. The k-means clustering algorithm relies heavily on optimization.

The optimization process seeks the set of centroids that minimizes the sum of squared distances between each data point and its nearest centroid. This process is repeated until convergence occurs, resulting in the best clustering solution.

Hierarchical clustering

Hierarchical Clustering is a clustering algorithm that uses unsupervised machine learning. It creates a cluster hierarchy, with each cluster subdivided into smaller sub-clusters.

Agglomerative Hierarchical Clustering begins by treating each data point as its own cluster, then merges the closest pairs of clusters successively until all the points are in one large cluster, or a stopping criterion is met. Hierarchical Clustering is useful for visualizing data structure and exploring various levels of abstraction.

DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is the foundation algorithm for density-based clustering. It can find clusters of various shapes and sizes in a large amount of data that contains noise and outliers.

4.2 Applying Cluster Analysis in Coursework

4.2.1 Credit Card data

```
credit
```

	CUST_ID	BALANCE	BALANCE_FREQUENCY	PURCHASES	ONEOFF_PURCHASES	INSTALLMENTS_PURCHASES	CASH_ADVANCE	PURCHA
0	C10001	40.900749	0.818182	95.40	0.00	95.40	0.000000	
1	C10002	3202.467416	0.909091	0.00	0.00	0.00	6442.945483	
2	C10003	2495.148862	1.000000	773.17	773.17	0.00	0.000000	
3	C10004	1666.670542	0.636364	1499.00	1499.00	0.00	205.788017	
4	C10005	817.714335	1.000000	16.00	16.00	0.00	0.000000	
...	...	...	...	...	...	...	...	

Summary of Credit Card data

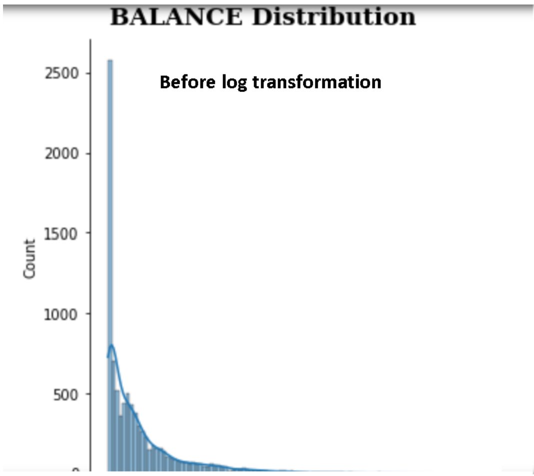
Credit card information consists of 8950 rows and 18 columns. There are no duplicate values. Missing values are found in the Credit limit and Minimum Payment columns.

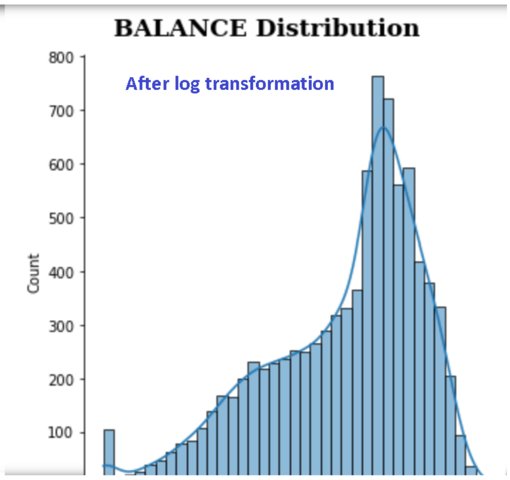
```
credit.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8950 entries, 0 to 8949
Data columns (total 18 columns):
 #   Column                            Non-Null Count  Dtype
---  ------                            --------------  -----
 0   CUST_ID                           8950 non-null   object
 1   BALANCE                           8950 non-null   float64
 2   BALANCE_FREQUENCY                 8950 non-null   float64
 3   PURCHASES                         8950 non-null   float64
 4   ONEOFF_PURCHASES                  8950 non-null   float64
 5   INSTALLMENTS_PURCHASES            8950 non-null   float64
 6   CASH_ADVANCE                      8950 non-null   float64
 7   PURCHASES_FREQUENCY               8950 non-null   float64
 8   ONEOFF_PURCHASES_FREQUENCY        8950 non-null   float64
 9   PURCHASES_INSTALLMENTS_FREQUENCY  8950 non-null   float64
 10  CASH_ADVANCE_FREQUENCY            8950 non-null   float64
 11  CASH_ADVANCE_TRX                  8950 non-null   int64
 12  PURCHASES_TRX                     8950 non-null   int64
 13  CREDIT_LIMIT                      8949 non-null   float64
 14  PAYMENTS                          8950 non-null   float64
 15  MINIMUM_PAYMENTS                  8637 non-null   float64
 16  PRC_FULL_PAYMENT                  8950 non-null   float64
 17  TENURE                            8950 non-null   int64
```

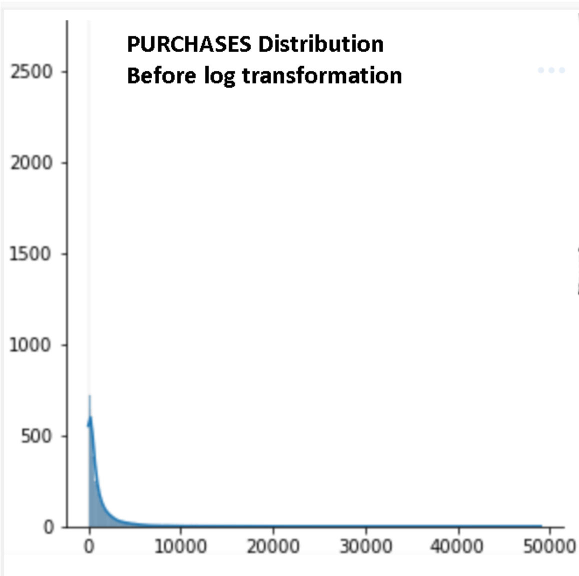
```
credit.isna().sum()

CUST_ID                               0
BALANCE                               0
BALANCE_FREQUENCY                     0
PURCHASES                             0
ONEOFF_PURCHASES                      0
INSTALLMENTS_PURCHASES                0
CASH_ADVANCE                          0
PURCHASES_FREQUENCY                   0
ONEOFF_PURCHASES_FREQUENCY            0
PURCHASES_INSTALLMENTS_FREQUENCY      0
CASH_ADVANCE_FREQUENCY                0
CASH_ADVANCE_TRX                      0
PURCHASES_TRX                         0
CREDIT_LIMIT                          1
PAYMENTS                              0
MINIMUM_PAYMENTS                    313
PRC_FULL_PAYMENT                      0
TENURE                                0
```

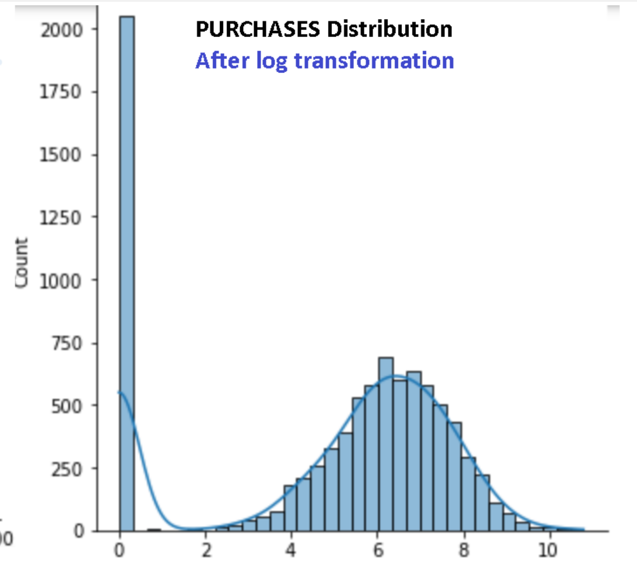
Data preparation steps carried out for building the model are:

- The Customer ID column was removed.
- Numerical features (Dtypes float and int) are grouped together.
- Fill in the missing values in the credit limit column with mean ()
- Fill in the missing values in the minimum payments column with backward fill.
- Log transformation for highly skewed features





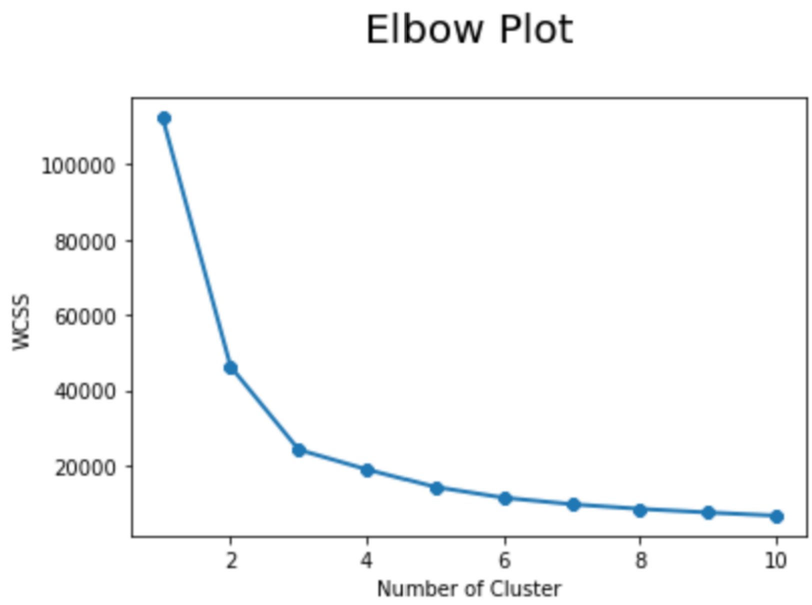




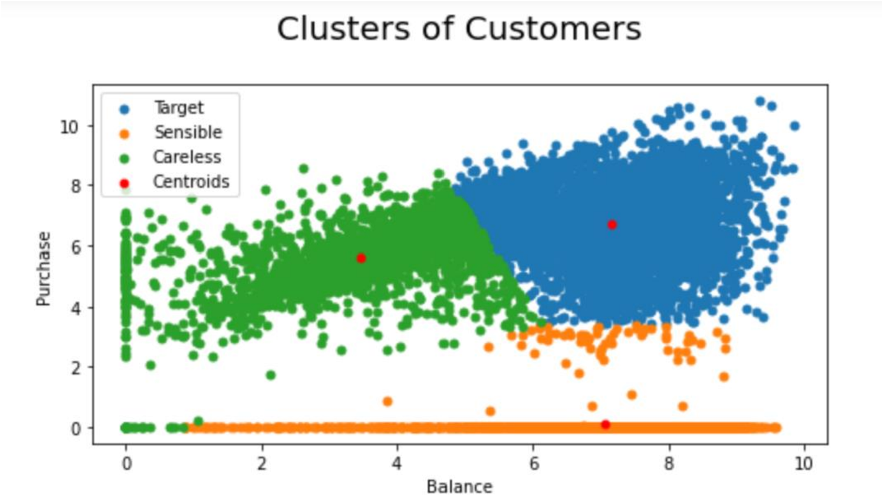
Results

```
# BALANCE and PURCHASES Predictor
x = credit.iloc[:, [0,2]].values
```

- K-Means Clustering

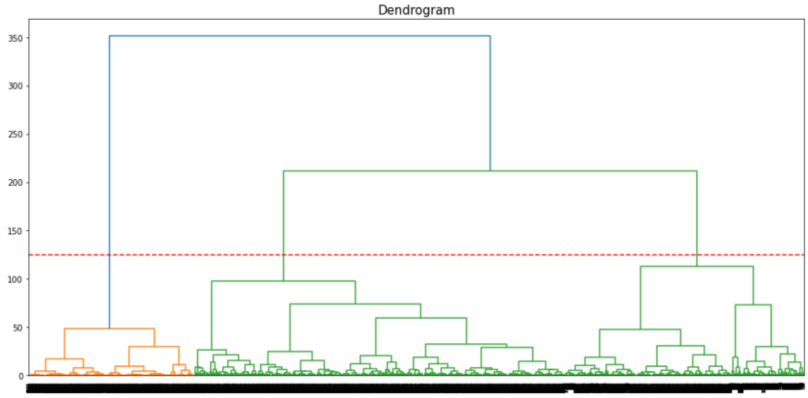


From the elbow plot above, we can see that the best fit for number of clusters is 3.

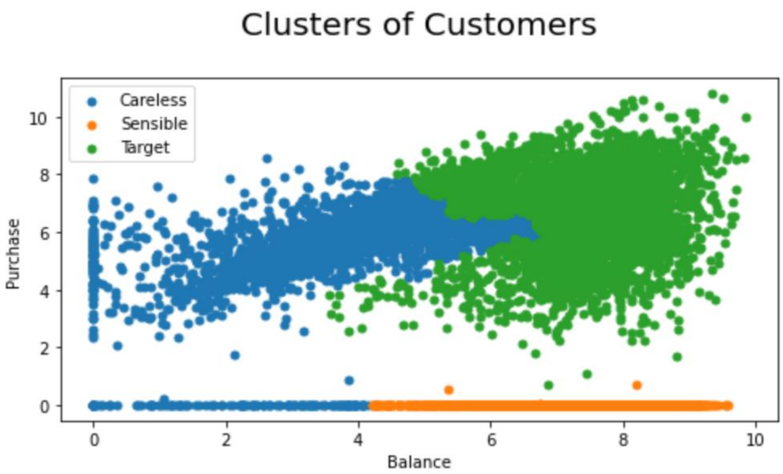


Target (Blue) group have a higher balance and higher purchases.
Sensible (Orange) group have a high balance but low purchases.
Careless (Green) group have a lower balance but high purchases.

- Hierarchical Clustering



From the dendrogram above, a red dotted horizontal line that passes through longest distance, we can see that best fit for number of clusters is 3.



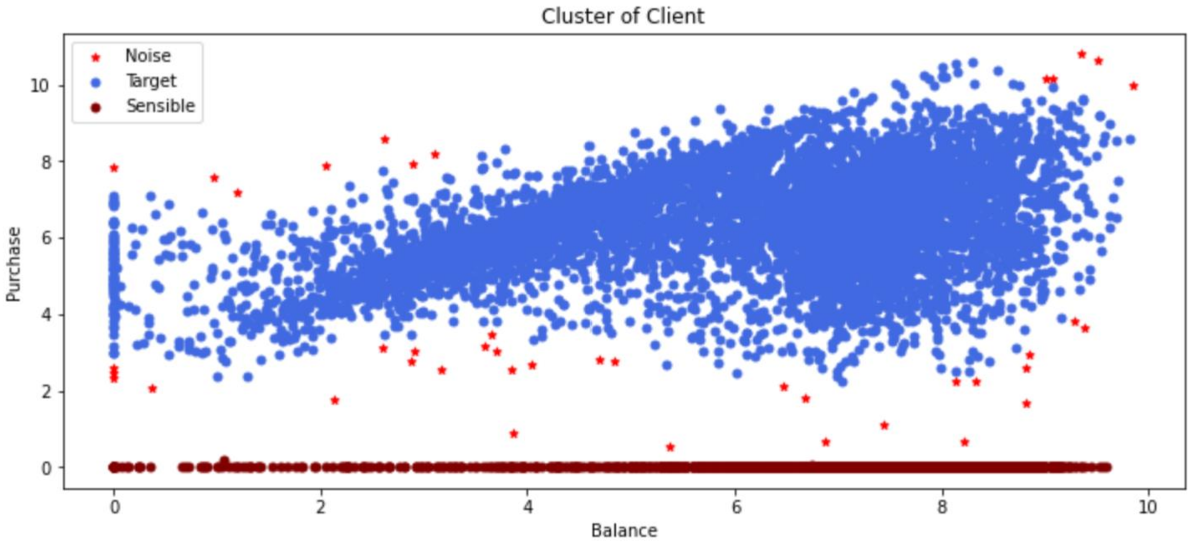
Target (Green) group have a higher balance and higher purchases.
Sensible (Orange) group have a high balance but low purchases.
Careless (Blue) group have a lower balance but high purchases.

- DBSCAN

```
# print number of instances in each cluster group, Note that -1 represents noises/outliers
credit['cluster group'].value_counts()
```

```
 0    6878
 1    2050
-1      22
Name: cluster group, dtype: int64
```

DBSCAN no need to define any cluster, it will automatically group the data into clusters on its own.



According to K-Means and Hierarchical clustering analysis, balance and purchases are directly proportional to the Target, Sensible, and Careless groups. They both performed well in customer segmentation, forming three similar clusters. DBSCAN can be used to identify noise, demonstrating that K-Means and Hierarchical clustering do not work well with noise.

5. Task B.2 Association Rule

5.1 Association Rules – Apriori Algorithm

Apriori is an algorithm used in association rule learning to discover frequent item sets in a transactional database. There is no need to define target variable as it figures out relationship among the data itemsets.

The process is repeated, using the frequent itemsets from the previous iteration as the basis for generating the next set of candidate itemsets. This process continues until no more frequent itemsets can be found. The resulting frequent itemsets can then be used to generate association rules, which describe relationships between items in the database.

5.2 Applying Apriori Algorithm in Coursework

5.2.1 Online Retail Dataset

```
data.head()
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom

Summary of Online Retail data

Online Retail data contain 541,909 rows and 8 columns. There are missing values in ‘Description’ and ‘CustomerID’. There are 5,231 duplicated rows in Online Retail data.

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541909 entries, 0 to 541908
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   InvoiceNo    541909 non-null  object
 1   StockCode    541909 non-null  object
 2   Description  540455 non-null  object
 3   Quantity     541909 non-null  int64
 4   InvoiceDate  541909 non-null  datetime64[ns]
 5   UnitPrice    541909 non-null  float64
 6   CustomerID   406829 non-null  float64
 7   Country      541909 non-null  object
```

```
data.isna().sum()
```

```
InvoiceNo              0
StockCode              0
Description         1454
Quantity               0
InvoiceDate            0
UnitPrice              0
CustomerID        135080
Country                0
TotalAmount            0
InvoiceYear            0
InvoiceMonth           0
InvoiceYearMonth       0
dtype: int64
```

Data preparation steps carried out for building the model are:

- 'TotalAmount' column was added by deriving from 'Quantity' * 'UnitPrice'.
- Remove 10,624 'Quantity' rows with negative values.
- Remove the duplicated rows (5,231 in total).
- Remove 1,174 rows of ’UnitPrice' with 0 values.

Exploratory Data Analysis

```
# Top 20 Most Frequently sold items by quantitiy
gp_stockcode_frq_quantitiy.head(20)
```

```
Description
PAPER CRAFT , LITTLE BIRDIE          80995
MEDIUM CERAMIC TOP STORAGE JAR       77916
WORLD WAR 2 GLIDERS ASSTD DESIGNS    54319
JUMBO BAG RED RETROSPOT              46078
WHITE HANGING HEART T-LIGHT HOLDER   36706
ASSORTED COLOUR BIRD ORNAMENT        35263
PACK OF 72 RETROSPOT CAKE CASES      33670
POPCORN HOLDER                       30919
RABBIT NIGHT LIGHT                   27153
MINI PAINT SET VINTAGE               26076
PACK OF 12 LONDON TISSUES            25329
PACK OF 60 PINK PAISLEY CAKE CASES   24230
BROCADE RING PURSE                   22927
VICTORIAN GLASS HANGING T-LIGHT      22404
ASSORTED COLOURS SILK FAN            21876
RED  HARMONICA IN BOX                20945
JUMBO BAG PINK POLKADOT              20148
SMALL POPCORN HOLDER                 18241
```

```
# Frequently sold items by total amount
gp_stockcode_frq_amount.head(20)
```

```
Description
PAPER CRAFT , LITTLE BIRDIE           168469.60
REGENCY CAKESTAND 3 TIER              142264.75
WHITE HANGING HEART T-LIGHT HOLDER    100392.10
JUMBO BAG RED RETROSPOT                85040.54
MEDIUM CERAMIC TOP STORAGE JAR         81416.73
POSTAGE                                77803.96
PARTY BUNTING                          68785.23
ASSORTED COLOUR BIRD ORNAMENT          56413.03
Manual                                 53419.93
RABBIT NIGHT LIGHT                     51251.24
CHILLI LIGHTS                          46265.11
PAPER CHAIN KIT 50'S CHRISTMAS         42584.13
PICNIC BASKET WICKER 60 PIECES         39619.50
BLACK RECORD COVER FRAME               39045.80
JUMBO BAG PINK POLKADOT                37254.36
DOORMAT KEEP CALM AND COME IN          35880.85
SPOTTY BUNTING                         35509.55
WOOD BLACK BOARD ANT WHITE FINISH      34414.71
```

According to the table above, the most popular item is "paper craft, little birdie," with total sales of $168,459.60 and 80,995 units sold.

```
# Order countries by total amount
gp_country['TotalAmount'].sum().sort_v
```

```
Country
United Kingdom    7285024.644
Netherlands        285446.340
EIRE               265262.460
Germany            228678.400
France             208934.310
Australia          138453.810
Spain               61558.560
Switzerland         56443.950
Belgium             41196.340
Sweden              38367.830
Japan               37416.370
Norway              36165.440
Portugal            33375.840
Finland             22546.080
Singapore           21279.290
Channel Islands     20440.540
Denmark             18955.340
Italy               17483.240
```

```
# Order countries by number of i
gp_country['InvoiceNo'].unique()
```

```
Country
United Kingdom     16646
Germany              457
France               389
EIRE                 260
Belgium               98
Netherlands           94
Spain                 90
Australia             57
Portugal              57
Switzerland           51
Finland               41
Italy                 38
Norway                36
Sweden                36
Channel Islands       26
Poland                19
Japan                 19
Denmark               18
```

The United Kingdom, the Netherlands, EIRE, Germany, and France are the top five revenue-generating countries for the online retail store. The United Kingdom was the largest customer, with 16,646 invoices. Following that, we will apply the association rule to the United Kingdom, France, and EIRE to determine which items are the most popular in each country.

Results

United Kingdom Rules

```python
# The most significant rules
uk_final_results = uk_final_results.sort_values(by='Confidence', ascending=0)
uk_final_results.head(10)
```

	GeneralRules	LeftRules	RightRules	Support	Confidence	Lift
1	PINK REGENCY TEACUP AND SAUCER , ROSES REGENCY...	PINK REGENCY TEACUP AND SAUCER , GREEN REGENCY...	ROSES REGENCY TEACUP AND SAUCER	0.020485	0.844059	20.723028
0	PINK REGENCY TEACUP AND SAUCER , GREEN REGENCY...	PINK REGENCY TEACUP AND SAUCER	GREEN REGENCY TEACUP AND SAUCER	0.024270	0.819473	22.289120

With defined mini support of 0.02, mini confidence of 0.8, and mini lift of 2, there is 84.40% confidence that customer bought **'PINK REGENCY TEACUP AND SAUCER'**, **'GREEN REGENCY TEACUP AND SAUCER'** will also be likely to purchase with **'ROSES REGENCY TEACUP AND SAUCER'** in United Kingdom.

France Rules

```python
# The most significant rules
france_final_results = france_final_results.sort_values(by='Confidence', ascending=0)
france_final_results.head(10)
```

	GeneralRules	LeftRules	RightRules	Support	Confidence	Lift
0	DOLLY GIRL BABY GIFT SET , SPACEBOY BABY GIFT...	DOLLY GIRL BABY GIFT SET	SPACEBOY BABY GIFT SET	0.023136	1.0	24.312500
265	SET/6 RED SPOTTY PAPER CUPS , SET/6 RED SPOTTY...	SET/6 RED SPOTTY PAPER PLATES , PLASTERS IN TI...	SET/6 RED SPOTTY PAPER CUPS	0.023136	1.0	7.203704
207	RABBIT NIGHT LIGHT , MINI LIGHTS WOODLAND MUSH...	RED TOADSTOOL LED NIGHT LIGHT , MINI LIGHTS WO...	RABBIT NIGHT LIGHT	0.020566	1.0	5.328767
208	SET/20 RED RETROSPOT PAPER NAPKINS , MINI PAI...	SET/20 RED RETROSPOT PAPER NAPKINS , MINI PAI...	SET/6 RED SPOTTY PAPER CUPS	0.035990	1.0	7.203704
209	SET/20 RED RETROSPOT PAPER NAPKINS , MINI PAI...	SET/20 RED RETROSPOT PAPER NAPKINS , MINI PAI...	SET/6 RED SPOTTY PAPER PLATES	0.035990	1.0	7.780000
211	SET/6 RED SPOTTY PAPER CUPS , PACK OF 20 NAPKI...	SET/6 RED SPOTTY PAPER CUPS , PACK OF 20 NAPKI...	SET/6 RED SPOTTY PAPER PLATES	0.020566	1.0	7.780000
220	SET OF 9 BLACK SKULL BALLOONS , PACK OF 20 SKU...	SET OF 9 BLACK SKULL BALLOONS , PACK OF 20 SKU...	PACK OF 6 SKULL PAPER PLATES	0.025707	1.0	17.681818

The itemset that is most frequently purchased together in France is **"SET/6 RED SPOTTY PAPER PLATES"** and **"SET/6 RED SPOTTY PAPER CUPS"** with defined mini support of 0.02, mini confidence of 0.8, and mini lift of 2. It has 100% confidence.

EIRE Rules

```
eire_final_results = eire_final_results.sort_values(by='Confidence', ascending=0)
eire_final_results.head(10)
```

	GeneralRules	LeftRules	RightRules	Support	Confidence	Lift
688	ROSES REGENCY TEACUP AND SAUCER , PINK REGENC...	SET OF 3 REGENCY CAKE TINS , REGENCY TEA PLATE...	PINK REGENCY TEACUP AND SAUCER , ROSES REGENCY...	0.023077	1.0	9.629630
448	PINK REGENCY TEACUP AND SAUCER , ROSES REGENCY...	PINK REGENCY TEACUP AND SAUCER , REGENCY TEA P...	ROSES REGENCY TEACUP AND SAUCER	0.034615	1.0	5.909091
467	PINK REGENCY TEACUP AND SAUCER , ROSES REGENCY...	PINK REGENCY TEACUP AND SAUCER , REGENCY SUGAR...	ROSES REGENCY TEACUP AND SAUCER , REGENCY TEA...	0.023077	1.0	16.250000
466	PINK REGENCY TEACUP AND SAUCER , ROSES REGENCY...	PINK REGENCY TEACUP AND SAUCER , REGENCY SUGAR...	ROSES REGENCY TEACUP AND SAUCER	0.026923	1.0	5.909091
464	PINK REGENCY TEACUP AND SAUCER , ROSES REGENCY...	PINK REGENCY TEACUP AND SAUCER , REGENCY SUGAR...	ROSES REGENCY TEACUP AND SAUCER	0.042308	1.0	5.909091
463	PINK REGENCY TEACUP AND SAUCER , REGENCY SUGAR...	PINK REGENCY TEACUP AND SAUCER , REGENCY TEA P...	REGENCY SUGAR BOWL GREEN	0.030769	1.0	10.833333
1162	REGENCY TEA PLATE PINK , ROSES REGENCY TEACUP ...	PINK REGENCY TEACUP AND SAUCER , REGENCY CAKES...	REGENCY TEA PLATE PINK , REGENCY TEA PLATE GRE...	0.023077	1.0	18.571429
1164	ROSES REGENCY TEACUP AND SAUCER , PINK REGENC...	PINK REGENCY TEACUP AND SAUCER , REGENCY CAKES...	REGENCY TEA PLATE ROSES , ROSES REGENCY TEACU...	0.023077	1.0	17.333333
459	PINK REGENCY TEACUP AND SAUCER , ROSES REGENCY...	PINK REGENCY TEACUP AND SAUCER , REGENCY SUGAR...	ROSES REGENCY TEACUP AND SAUCER	0.042308	1.0	5.909091
1166	ROSES REGENCY TEACUP AND SAUCER , REGENCY TEA...	PINK REGENCY TEACUP AND SAUCER , REGENCY CAKES...	REGENCY TEA PLATE ROSES , ROSES REGENCY TEACU...	0.023077	1.0	18.571429

The item set that is most frequently purchased together in EIRE is **"ROSES REGENCY TEACUP AND SAUCER"** which has defined mini support of 0.02, mini confidence of 0.8, and mini lift of 2. It has 100% confidence.

5.2.2 Titanic Dataset

```
titanic.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

Summary of titanic data

The Titanic data set is made up of 891 rows and 12 columns. The 'Age,' 'Cabin,' and 'Embarked' Colmuns have missing values. There are no data duplicates found.

```
titanic.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
```

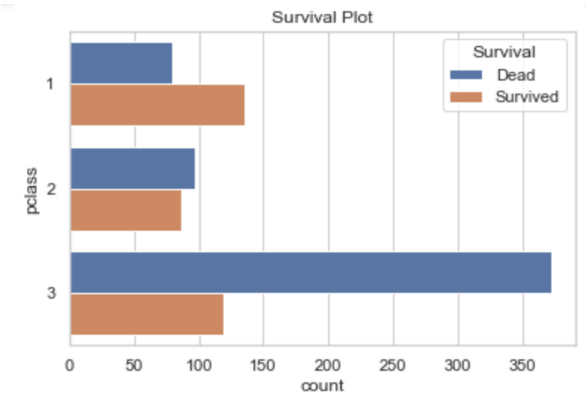
```
#Check any missing values
titanic.isnull().sum()
```

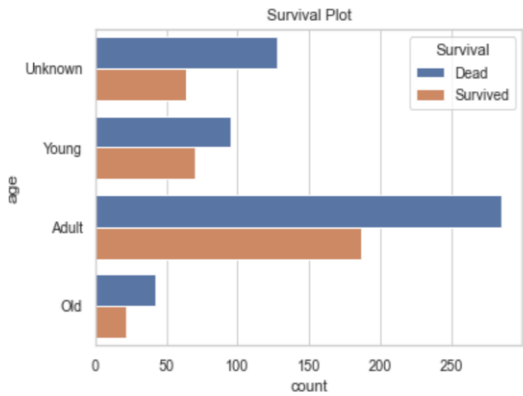
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
Title            0
dtype: int64
```

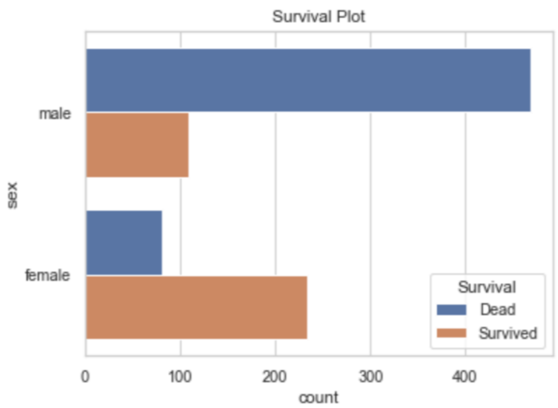
Data preparation steps carried out for building the model are:

- Selected features such as 'age,' 'pclass, 'survived,' and 'sex' are grouped into a list.
- Rename the column 'survived' to 'survival'. Replace binary survival values with 0 for 'Dead' and 1 for 'Survived'.
- Filled up 'age' missing values with 0.
- Using the binning method, 'age' column is divided into 4 categories, unknown, young, adult and old.
- Transaction Encoder is used to convert records to a binary array or matrix with only true or false values.

Exploratory Data Analysis







We can see from the distribution of male passengers in pclass 3 that the majority did not survive.

Results

```
rules = rules[['antecedents','consequents','confidence','support','lift']]
rules = rules.sort_values(by="confidence",ascending=False)
rules.head(20)
```

	antecedents	consequents	confidence	support	lift
12	(Dead, 1, Old)	(male)	1.000000	0.023569	1.544194
8	(Dead, Old)	(male)	0.976190	0.046016	1.507428
2	(female, 1)	(Survived)	0.968085	0.102132	2.522116
11	(Adult, female, 1)	(Survived)	0.965517	0.062851	2.515426
0	(Dead, 1)	(male)	0.962500	0.086420	1.486287
18	(Young, female, Dead)	(3)	0.958333	0.025814	1.739053
10	(Adult, Dead, 1)	(male)	0.948718	0.041526	1.465005
14	(Adult, 2, male)	(Dead)	0.941176	0.071829	1.527483
3	(Dead, 2)	(male)	0.938144	0.102132	1.448677
13	(Adult, 2, Dead)	(male)	0.927536	0.071829	1.432296
16	(Adult, Survived, 2)	(female)	0.925926	0.056117	2.627389
4	(female, 2)	(Survived)	0.921053	0.078563	2.399584
15	(Adult, female, 2)	(Survived)	0.909091	0.056117	2.368421
7	(Dead, female)	(3)	0.888889	0.080808	1.613035
17	(3, male, Unknown)	(Dead)	0.887755	0.097643	1.440783
9	(male, Old)	(Dead)	0.872340	0.046016	1.415766

The minimum support in this model is set at 0.02. We can conclude with 100% confidence based on the results that the individual involved is old, deceased, and a member of PClass 1. Female survivor from PClass 1 who is adult-aged, with 96.55% confidence. We can also draw the conclusion that, regardless of PClass, the deaths for most men were high.

Conclusion

This coursework made it easier for me to understand various machine learning techniques. Classification and regression are both supervised learning techniques that are used to predict or classify new data based on labeled examples. Classification is used when the output is a categorical variable, while regression is used when the output is a continuous variable.

Clustering, on the other hand, is an unsupervised learning technique that groups data points into clusters based on similarities between them. Clustering can be used to identify patterns and relationships in data that may not be immediately apparent and can be applied in a variety of fields, such as market segmentation and social network analysis.

Association rule mining, specifically the Apriori algorithm, is a technique used to uncover hidden relationships between variables in large datasets. It is commonly used in retail and e-commerce for product recommendation and in healthcare for disease diagnosis and treatment. The Apriori algorithm generates a set of rules that can be used to make predictions or guide decision-making.

References

1. https://www.javatpoint.com/supervised-machine-learning
2. https://www.javatpoint.com/unsupervised-machine-learning
3. https://www.kdnuggets.com/2020/04/dbscan-clustering-algorithm-machine-learning.html#:~:text=low%20point%20density.-,Density%2DBased%20Spatial%20Clustering%20of%20Applications%20with%20Noise%20(DBSCAN),is%20containing%20noise%20and%20outliers.
4. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4120293/#:~:text=The%20log%20transformation%20is%2C%20arguably,normal%20or%20near%20normal%20distribution.
5. https://www.analyticsvidhya.com/blog/2020/07/types-of-feature-transformation-and-scaling
6. https://www.jmp.com/en_sg/statistics-knowledge-portal/what-is-regression.html
7. https://www.jmp.com/en_sg/statistics-knowledge-portal/what-is-multiple-regression.html
8. https://www.javatpoint.com/apriori-algorithm-in-machine-learning
9. https://rasbt.github.io/mlxtend/user_guide/frequent_patterns/association_rules/