Group Exercise 2 (Ungraded) – Group E

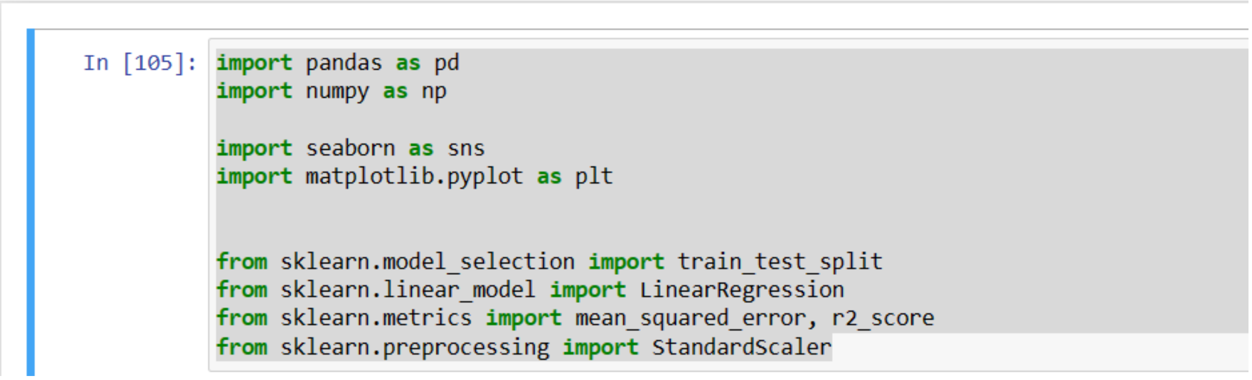
Student Details:

Name	Student ID
Baldeep Arora	500227219
Dharmik Bhatt	500228482
Hiral shah	500226537
Vishal Girase	500227109
Oluwakanyinsola Adebanjo	500228268
Shambhabi Pandit	500226139
Akash Rajput	
Atif Ahmed	

Index:

1) Problem Define

The problem states to accurately predict the orders from the store given different competition situations, market situations, temporal situations etc; while utilizing 3 datasets: Retail_Data_Orders_W23.csv, Retail_Data_W23.csv and Store.csv.



In [105]:

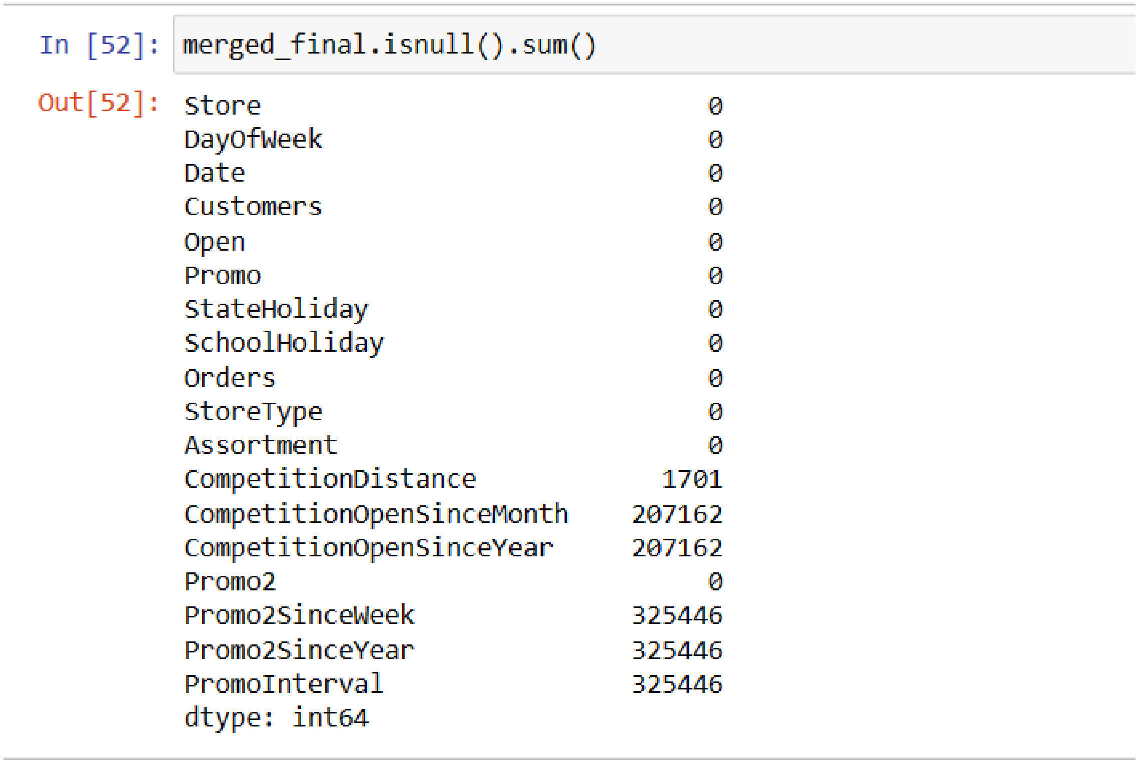
```
import pandas as pd
import numpy as np

import seaborn as sns
import matplotlib.pyplot as plt


from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.preprocessing import StandardScaler
```

We have imported the libraries:

- Pandas
- Numpy
- Seaborn
- Matplotlib.pyplot
- from sklearn.model_selection import train_test_split
- from sklearn.linear_model import LinearRegression
- from sklearn.metrics import mean_squared_error, r2_score
- from sklearn.preprocessing import StandardScaler

2) Data Cleaning and Pre-processing

To provide a solution we have cleaned the dataset using pandas and numpy. We are counting the number of missing values in each column of the DataFrame merged_final. It's part of the data cleaning and helps us handle the missing values.

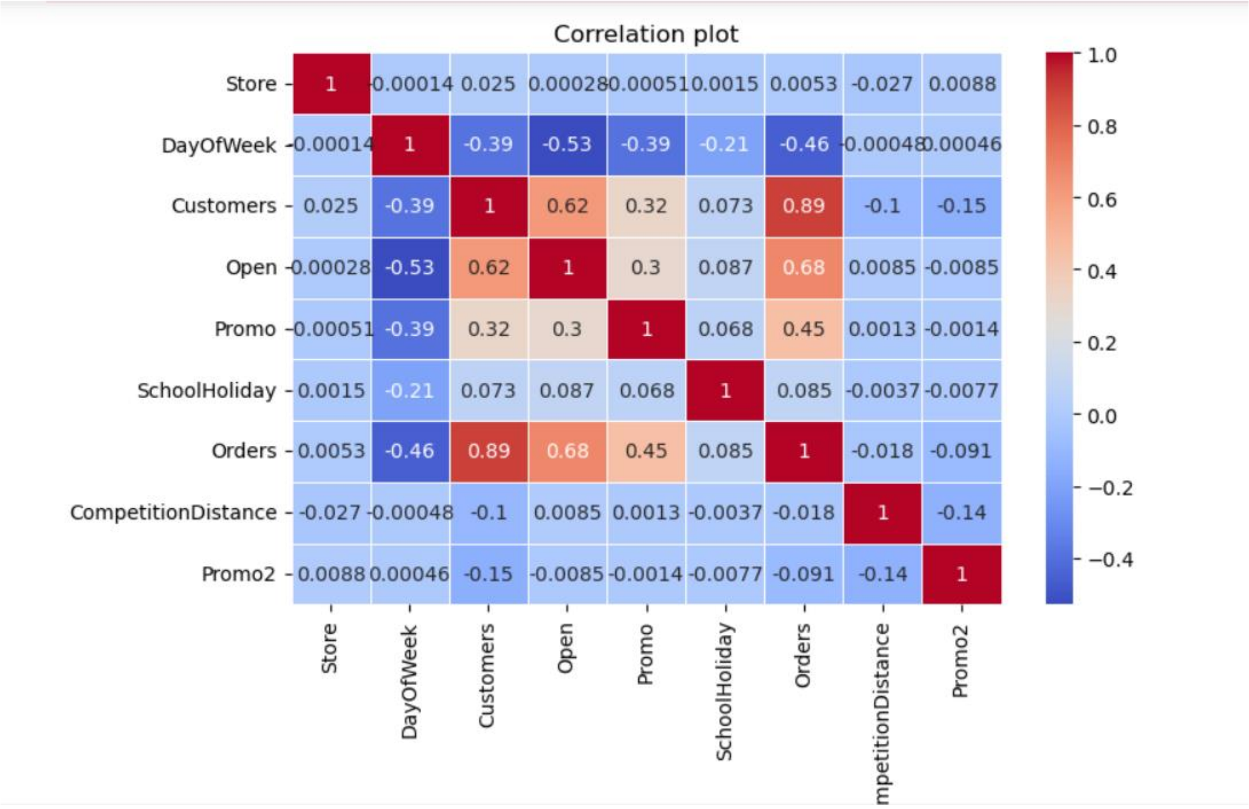


```
In [52]: merged_final.isnull().sum()

Out[52]: Store                            0
         DayOfWeek                        0
         Date                             0
         Customers                        0
         Open                             0
         Promo                            0
         StateHoliday                     0
         SchoolHoliday                    0
         Orders                           0
         StoreType                        0
         Assortment                       0
         CompetitionDistance           1701
         CompetitionOpenSinceMonth   207162
         CompetitionOpenSinceYear    207162
         Promo2                           0
         Promo2SinceWeek             325446
         Promo2SinceYear             325446
         PromoInterval               325446
         dtype: int64
```

We have removed all the null values by considering each column separately and have utilized different techniques to replace the null values with the corresponding values.

3) Exploratory Data Analysis

We have visualized it using seaborn and matplotlib. We have performed Exploratory Data Analysis by comparing the target variable, which is Orders to the dependent variables such as Promo, Customers, Store, etc. We have visualized correlation matrix, frequency distribution of the target variable, as well as a scatter plot of Orders vs Customers, Orders vs Stores, etc.

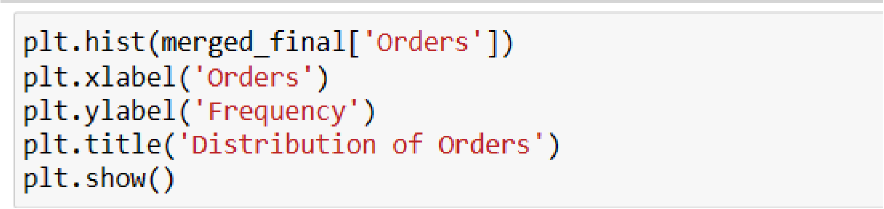


Correlation plot

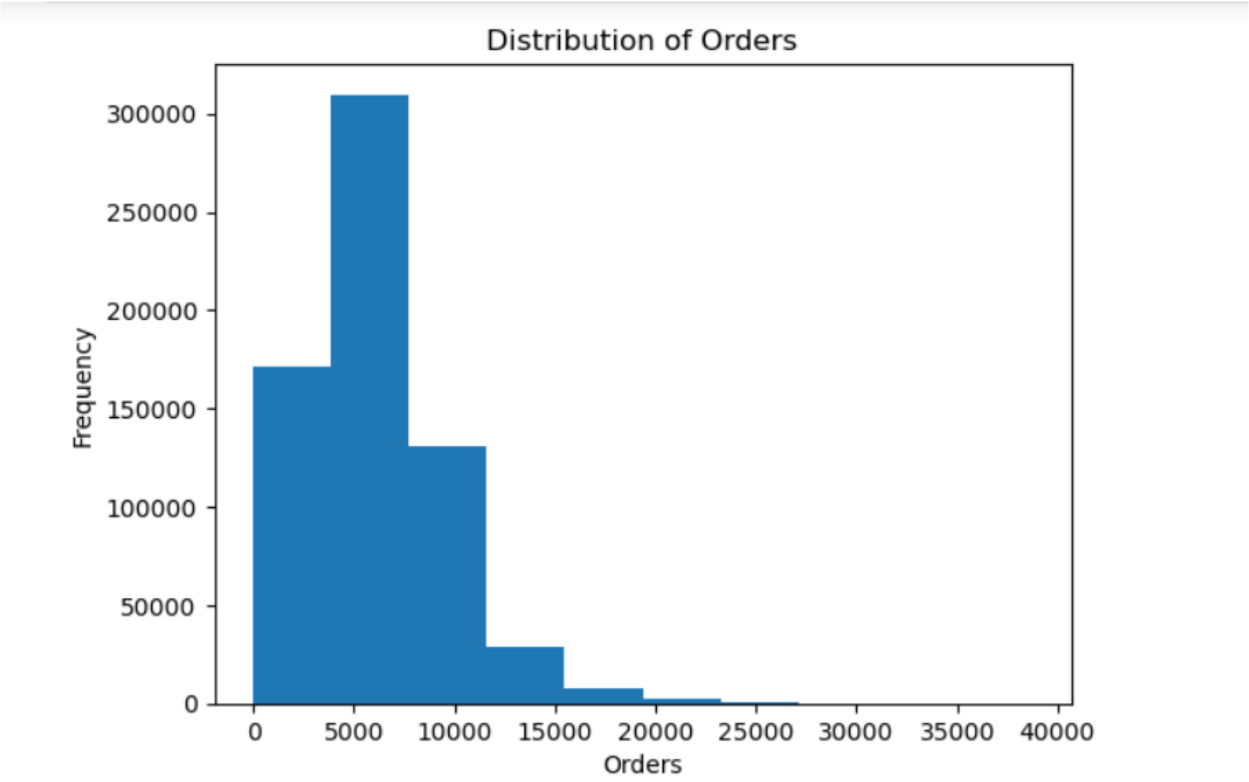
	Store	DayOfWeek	Customers	Open	Promo	SchoolHoliday	Orders	mpetitionDistance	Promo2
Store	1	-0.00014	0.025	0.00028	0.00051	0.0015	0.0053	-0.027	0.0088
DayOfWeek	-0.00014	1	-0.39	-0.53	-0.39	-0.21	-0.46	-0.00048	0.00046
Customers	0.025	-0.39	1	0.62	0.32	0.073	0.89	-0.1	-0.15
Open	-0.00028	-0.53	0.62	1	0.3	0.087	0.68	0.0085	-0.0085
Promo	-0.00051	-0.39	0.32	0.3	1	0.068	0.45	0.0013	-0.0014
SchoolHoliday	0.0015	-0.21	0.073	0.087	0.068	1	0.085	-0.0037	-0.0077
Orders	0.0053	-0.46	0.89	0.68	0.45	0.085	1	-0.018	-0.091
CompetitionDistance	-0.027	-0.00048	-0.1	0.0085	0.0013	-0.0037	-0.018	1	-0.14
Promo2	0.0088	0.00046	-0.15	-0.0085	-0.0014	-0.0077	-0.091	-0.14	1

We can identify that the target variable, Orders has a strong correlation with Customers, Open and Promo. Moreover, we are not considering the columns where the correlation is highly negative such as Date.

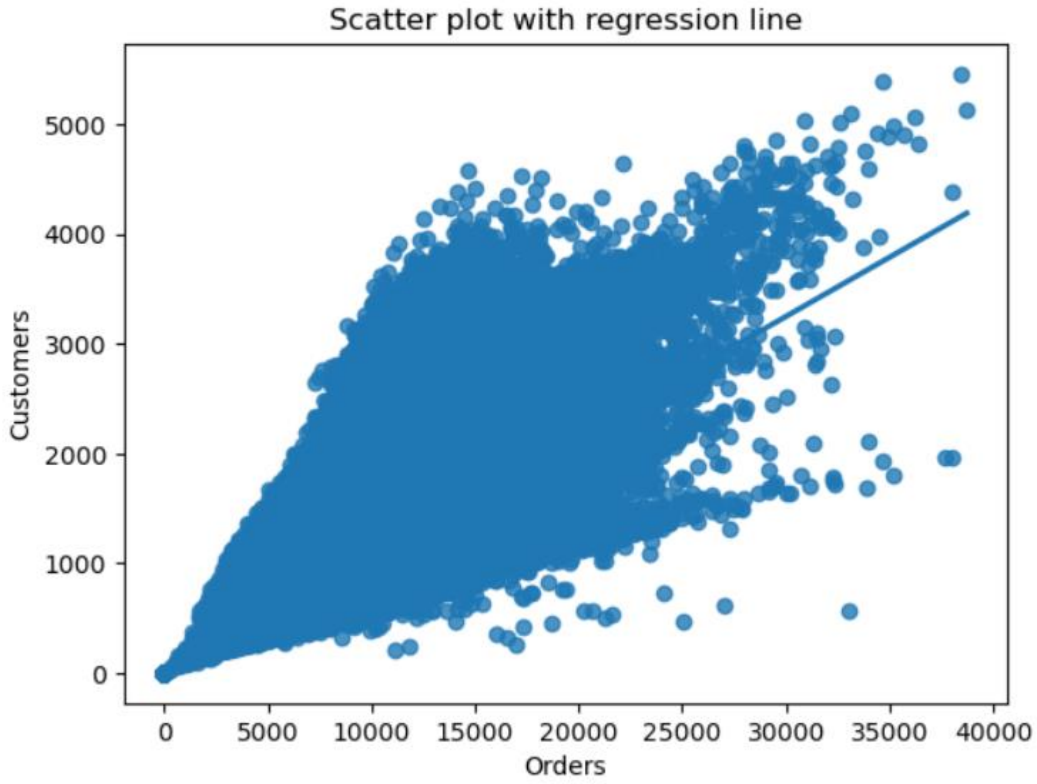
We have visualized the distribution of the 'Orders' column with the help of the datasets with the help of a histogram. Histograms give insights into how values are spread across different ranges and help us identify patterns or trends in the data. The plt.hist() function is used to generate the histogram.



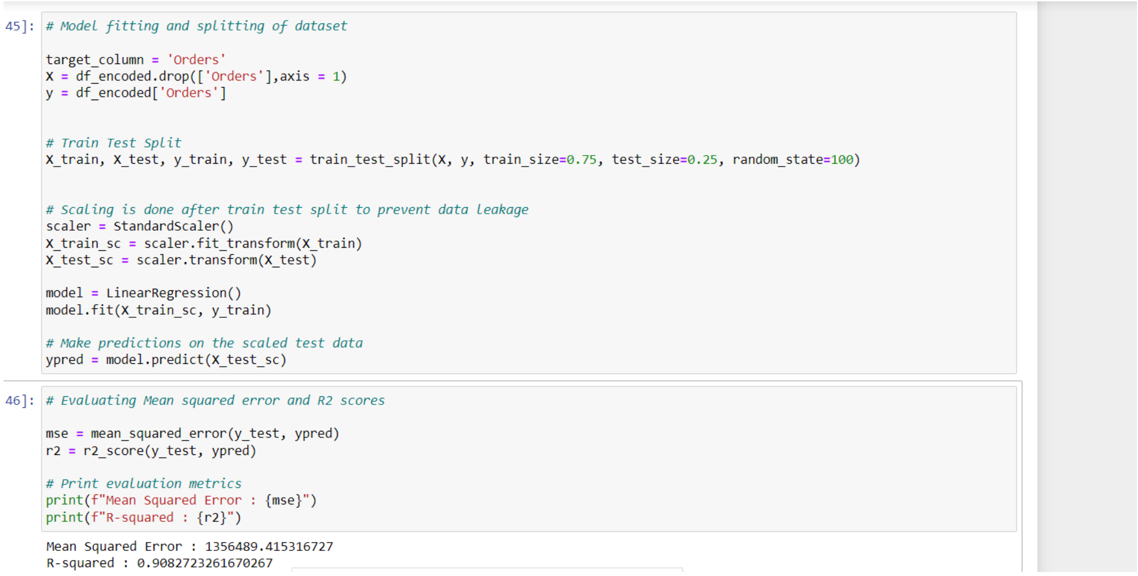
```
plt.hist(merged_final['Orders'])
plt.xlabel('Orders')
plt.ylabel('Frequency')
plt.title('Distribution of Orders')
plt.show()
```



Similarly, we have created a scatter plot for other relationships between the target variable and dependent variables as well.



5) Linear Regression



```
45]: # Model fitting and splitting of dataset

     target_column = 'Orders'
     X = df_encoded.drop(['Orders'],axis = 1)
     y = df_encoded['Orders']


     # Train Test Split
     X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.75, test_size=0.25, random_state=100)


     # Scaling is done after train test split to prevent data leakage
     scaler = StandardScaler()
     X_train_sc = scaler.fit_transform(X_train)
     X_test_sc = scaler.transform(X_test)

     model = LinearRegression()
     model.fit(X_train_sc, y_train)

     # Make predictions on the scaled test data
     ypred = model.predict(X_test_sc)
```

```
46]: # Evaluating Mean squared error and R2 scores

     mse = mean_squared_error(y_test, ypred)
     r2 = r2_score(y_test, ypred)

     # Print evaluation metrics
     print(f"Mean Squared Error : {mse}")
     print(f"R-squared : {r2}")
```

```
Mean Squared Error : 1356489.415316727
R-squared : 0.9082723261670267
```

6) Conclusion

The code is sets up the basic linear regression model which predict the orders datasets. It includes the rain-test split, feature scaling, model training, and making predictions. We can measure performance of the model by Mean Squared Error (MSE) or R-squared on the test set.

The Mean Squared Error (MSE): It is Measures the average squared difference between the predicted values and the actual values.

R-squared (R2): It represents the proportion of the variance in the dependent variable that is predictable from the independent variables.

The Mean Squared Error is: 1356489.41
The R-squared is: 0.90827