MACY'S INC. SUMMARY

- Macy's inc. is a trusted source for quality brands at great values from off-price to luxury. Macy's, Inc. operates department stores under the nameplates Macy's and Bloomingdale's, and specialty stores that include Bloomingdale's The Outlet, Bluemercury and Macy's Backstage. Macy's inc. currently have 783 boxes(stores) in 722 locations. The boxes include Macy's furniture stores, Department Stores, Bloomingdales's etc. Macy's inc. net worth is estimated $6.14B. Macy's hub location is New York Herald Square. Macy's conduct the annual Macy's Thanksgiving Day Parade in New York City since 1924 and has sponsored the city's annual Fourth of July fireworks display since 1976. Macy's Herald Square is one of the largest department stores in the world.

DATA DESCRIPTION 1

- The data we will be using is titled 'Macys_Store_sales'
- This is historical data that report sales from 2010-02-05 to 2012-11-01 over 45 stores.
- The areas covered in the dataset are:

- Store - the store number
- Date - the week of sales
- Weekly_Sales - sales for the given store
- Holiday_Flag - whether the week is a special holiday week 1 – Holiday week 0 – Non-holiday week
- Temperature - Temperature on the day of sale
- Fuel_Price - Cost of fuel in the region
- CPI – Prevailing consumer price index
- Unemployment - Prevailing unemployment rate

DATA DESCRIPTION 2

- The second dataset that we will create includes the four main holidays in America with different dates over four years.
- The below data set includes:
- Super Bowl: 12-Feb-10, 11-Feb-11, 10-Feb-12, 8-Feb-13
- Labour Day: 10-Sep-10, 9-Sep-11, 7-Sep-12, 6-Sep-13
- Thanksgiving: 26-Nov-10, 25-Nov-11, 23-Nov-12, 29-Nov-13
- Christmas: 31-Dec-10, 30-Dec-11, 28-Dec-12, 27-Dec-13

OBJECTIVE

- The aim of the analysis is to understand the sales trends within Macy's stores over the span of 4 years. This includes holiday sales periods alongside non-holiday sale periods. I will analyse the current data and look at trends that have been produced over the years and where showcase what period displays the highest sales.
- The current model is inadequate at predicting the sale patterns which is resulting in lack of stock on the days and return on sales. Due to this I will use machine learning and create an ideal algorithm which will predict demand accurately and ingest factors like economic conditions including CPI, Unemployment Index, etc. I will than build a model to display this.

IMPORTING THE DATA

I first start by telling python to 'import' the appropriate library's and view the data using '.head' view the data

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import sklearn
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline
```

```
# Import Macys retail dataset
data = pd.read_csv('Macys_Store_sales.csv')
# View a snapshot of the data
data.head()
```

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment
0	1	05-02-2010	1643690.90	0	42.31	2.572	211.096358	8.106
1	1	12-02-2010	1641957.44	1	38.51	2.548	211.242170	8.106
2	1	19-02-2010	1611968.17	0	39.93	2.514	211.289143	8.106
3	1	26-02-2010	1409727.59	0	46.63	2.561	211.319643	8.106
4	1	05-03-2010	1554806.68	0	46.50	2.625	211.350143	8.106

DATA PREPERATION

I will use '.dtypes' to see what type of data we are using

```
data.dtypes
```

```
Store             int64
Date             object
Weekly_Sales    float64
Holiday_Flag      int64
Temperature     float64
Fuel_Price      float64
CPI             float64
Unemployment    float64
dtype: object
```

'Date' is an object type data that needs to be converted to Datetime

```
from datetime import datetime
data['Date'] = pd.to_datetime(data['Date'])
data.dtypes
```

```
Store                    int64
Date            datetime64[ns]
Weekly_Sales           float64
Holiday_Flag             int64
Temperature            float64
Fuel_Price             float64
CPI                    float64
Unemployment           float64
dtype: object
```

DATA INFORMATION

Using '.shape' explains how many rows and row columns are in the table

```
data.shape
```

```
(6435, 12)
```

There are 6435 rows and 12 columns.

The 'isnull' and 'sum' function can be used together to show any missing data.

```
data.isnull().sum()
```

```
Store           0
Date            0
Weekly_Sales    0
Holiday_Flag    0
Temperature     0
Fuel_Price      0
CPI             0
Unemployment    0
month           0
Year            0
Month           0
Day             0
dtype: int64
```

There is no missing data in the table.

WHICH STORE HAS MAXIMUM SALES?

I created a new variable, then grouped by 'Store' and used function 'sum()' to acquire total sales.

```
pd.DataFrame(max_sales).head(5)
```

Store	Weekly_Sales
20	301397792.0
4	299543953.0
14	288999911.0
13	286517704.0
2	275382441.0

Store '20' has the maximum sales of $301,397,792.

WHICH STORE HAS MAXIMUM STANDARD DEVIATION? ALSO, THE COEFFICIENT OF MEAN TO STANDARD DEVIATION?

The new variable is grouped by 'Store and the function '.std()' will be used to find standard deviation.

```
data_std = data.groupby('Store')['Weekly_Sales'].std().round(2).sort_values(ascending=False)
pd.DataFrame(data_std).head()
```

Store	Weekly_Sales
14	317569.95
10	302262.06
20	275900.56
4	266201.44
13	265507.00

'Store 14' has the maximum standard deviation with $317569.95

I then found the mean to standard deviation of store 14 by using '.mean()*100'

```
macys14 = data[data.Store == 14].Weekly_Sales
m_s_d = macys14.std()/macys14.mean()*100
m_s_d
```

```
15.713673600948338
```

The mean to standard deviation is 15.71%

WHICH STORE/S HAS A GOOD QUARTERLY GROWTH RATE IN Q3 2012?

I created a new variable and use pandas 'datetime' to isolate dates between '07-01-2012' through to '09-30-2012. I then apply these dates to another variable and use '.sum()' with 'Store' and 'Weekly_Sales' to get growth rate in Q3 2012. I use print to showcase the results.

```
macys_data_Q32012 = data[(pd.to_datetime(data['Date']) >= pd.to_datetime('07-01-2012')) & (pd.to_datetime(data['Date'])
<= pd.to_datetime('09-30-2012'))]
macys_data_growth = macys_data_Q32012.groupby(['Store'])['Weekly_Sales'].sum()
print("Store Number {} has Good Quartely Growth in Q3'2012 {}".format(macys_data_growth.idxmax(),macys_data_growth.max()))
```

```
Store Number 4 has Good Quartely Growth in Q3'2012 25652119.35
```

Store 4 has good quarterly growth in Q3 2012 with $25652119.35

MACY'S HAS FOUR HOLIDAYS IN THE YEAR. SOME HOLIDAYS HAVE A NEGATIVE IMPACT ON SALES. WHAT HOLIDAYS HAVE HIGHER SALES THAN THE MEAN SALES IN THE NON-HOLIDAY SEASON FOR ALL THE STORES TOGETHER?

I first input the holiday dates given as a list. Allowing for Python to read and interpret the data.

```
super_bowl = ['12-2-2010', '11-2-2011', '10-2-2012', '8-2-2013']
labour_day = ['10-3-2010', '9-9-2011', '7-9-2012', '6-9-2013']
thanksgiving = ['26-11-2010', '25-11-2011', '23-11-2012' ,'29-11-2013']
christmas = ['31-12-2010', '30-12-2011', '28-12-2012', '27-12-2013']
```

Using '.loc' I specify the specific dates in each holiday, find the mean of each 'Weekly_Sales' and round the number.

```
super_bowl_s = data.loc[data.Date.isin(super_bowl)]['Weekly_Sales'].mean()
labour_day_s = data.loc[data.Date.isin(labour_day)]['Weekly_Sales'].mean()
thanksgiving_s = data.loc[data.Date.isin(thanksgiving)]['Weekly_Sales'].mean()
christmas_s = data.loc[data.Date.isin(christmas)]['Weekly_Sales'].mean()
```

The same is done with non-holiday sales.

```
non_holiday_s = data[(data['Holiday_Flag'] == 0)]['Weekly_Sales'].mean()
non_holiday_s = round(non_holiday_s,2)
non_holiday_s
```

The results are displayed in a data frame.

```
Highest_Selling_Period = pd.DataFrame([{'Super Bowl Sales':super_bowl_s,
                                        'Labour Day Sales':labour_day_s,
                                        'Thanksgiving Sales':thanksgiving_s,
                                        'Christmas Sales':christmas_s,
                                        'Non-Holiday Sales':non_holiday_s}])
```

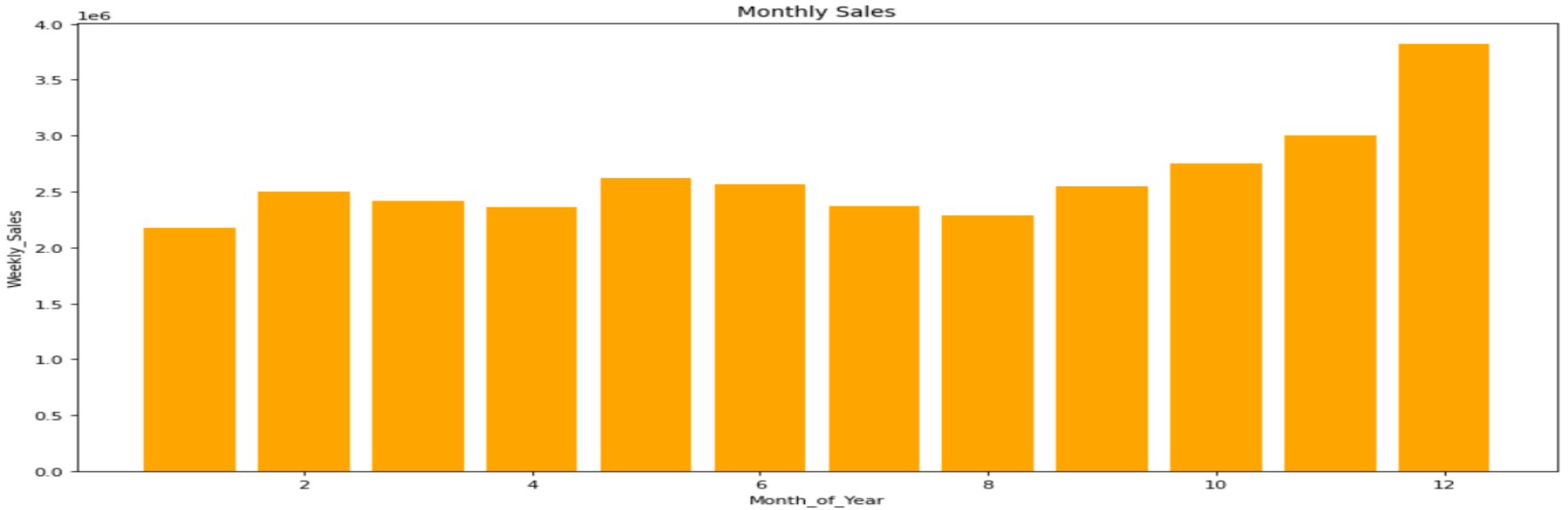
	Super Bowl Sales	Labour Day Sales	Thanksgiving Sales	Christmas Sales	Non-Holiday Sales
0	1079127.99	1056592.08	1471273.43	960833.11	1041256.38

Thanksgiving has the highest sales when compared to Non-Holiday Sales by amount $430017.05
Furthermore, Superbowl and Labour Day Sales exceed Non-Holiday Sales.

PROVIDE A MONTHLY AND SEMESTER VIEW OF SALES IN UNITS AND GIVE INSIGHTS.

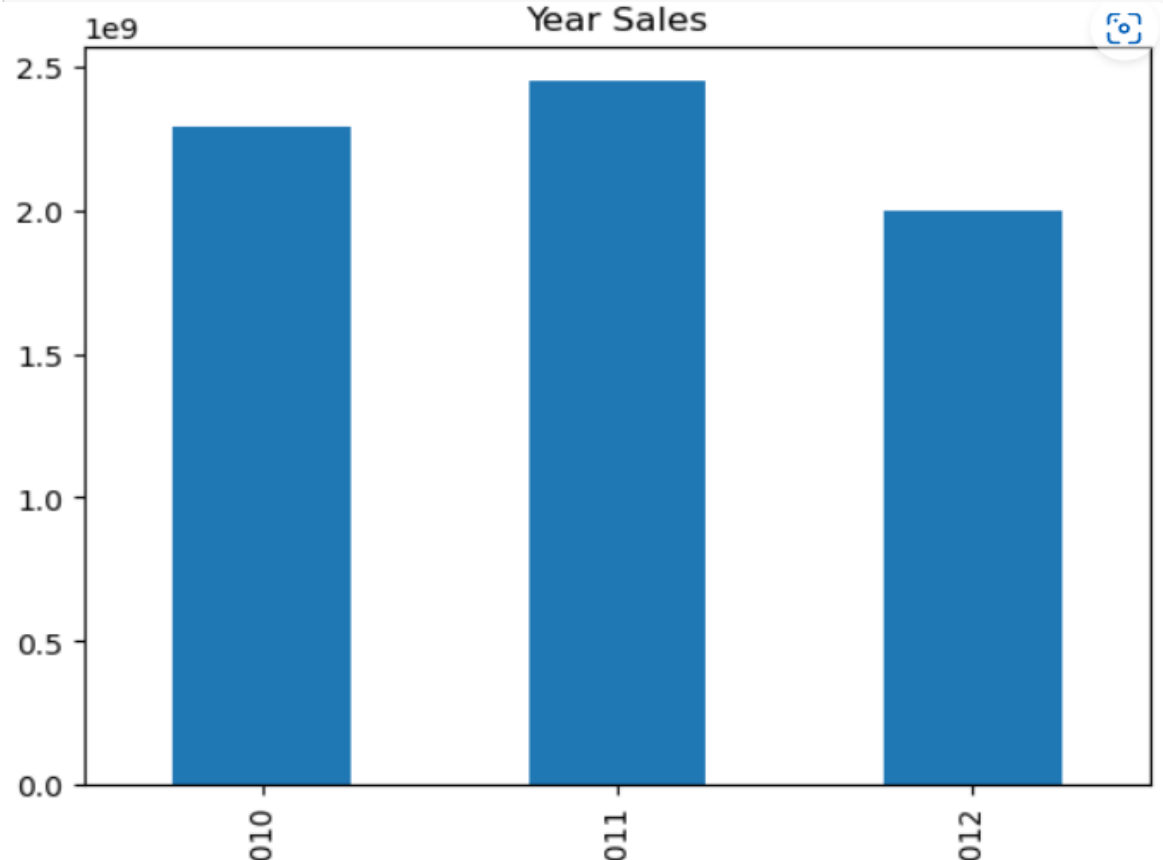
I will create 2 graphs to display the sales to draw insights. One will be a month based bar graph showing max sales. The second will be a yearly sales bar graph. They will be displayed using 'plt.show()'

```
data['month'] = data['Date'].dt.month
plt.figure(figsize=(14,7), dpi=80)
plt.bar(data['month'], data['Weekly_Sales'], color = 'orange')
plt.xlabel('Month_of_Year')
plt.ylabel('Weekly_Sales')
plt.title('Monthly Sales')
plt.show()
```



CONT.

```
data['Year']= data['Date'].dt.year
plt.figure(figsize=(11,8), dpi=85)
data.groupby('Year')[['Weekly_Sales']].sum().plot(kind='bar', legend=False)
plt.title('Year Sales')
plt.show()
```



Insights

The two graphs give the incite that December has the highest monthly sales. This supports my previous analysis research showing Thanksgiving sales being the highest, alongside having another holiday in Christmas to support sales. Both holidays provide higher sale days than non-holidays sales supporting higher earnings in those months.

The highest selling year occurred in 2011. This trend is supported by 'Macrotrends.net' which shows a large spike in quarterly growth in December 2011 at around 9%. This also could be attributed to higher inflation rate in 2011 at around 1.52% higher than 2010.

BUILD PREDICTION MODEL TO FORECAST DEMAND

I imported the appropriate library's.

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.linear_model import LinearRegression
```

I defined which variables are independent or dependent. The 'Weekly_Sales' is the dependent variable stored under variable y. Store, fuel price, CPI, unemployment, day, month, and year are the independent variables stored under x.

```
data['Year']= data['Date'].dt.year
data['Month']= data['Date'].dt.month
data['Day']= data['Date'].dt.day

X = data[['Store','Fuel_Price','CPI','Unemployment','Day','Month','Year']]
y = data['Weekly_Sales']
```

I train the model using 'Import train_test_spit from sklearn.model_selection' and use 80% of the data to train and leave 20% to test.

```
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2)
```

The Standard Scaler function is used to standardise the data and bring all values to a common scale.

```
from sklearn.preprocessing import StandardScaler
stc = StandardScaler()
X_train = stc.fit_transform(X_train)
X_test = stc.fit_transform(X_test)
```

LINEAR REGRESSION

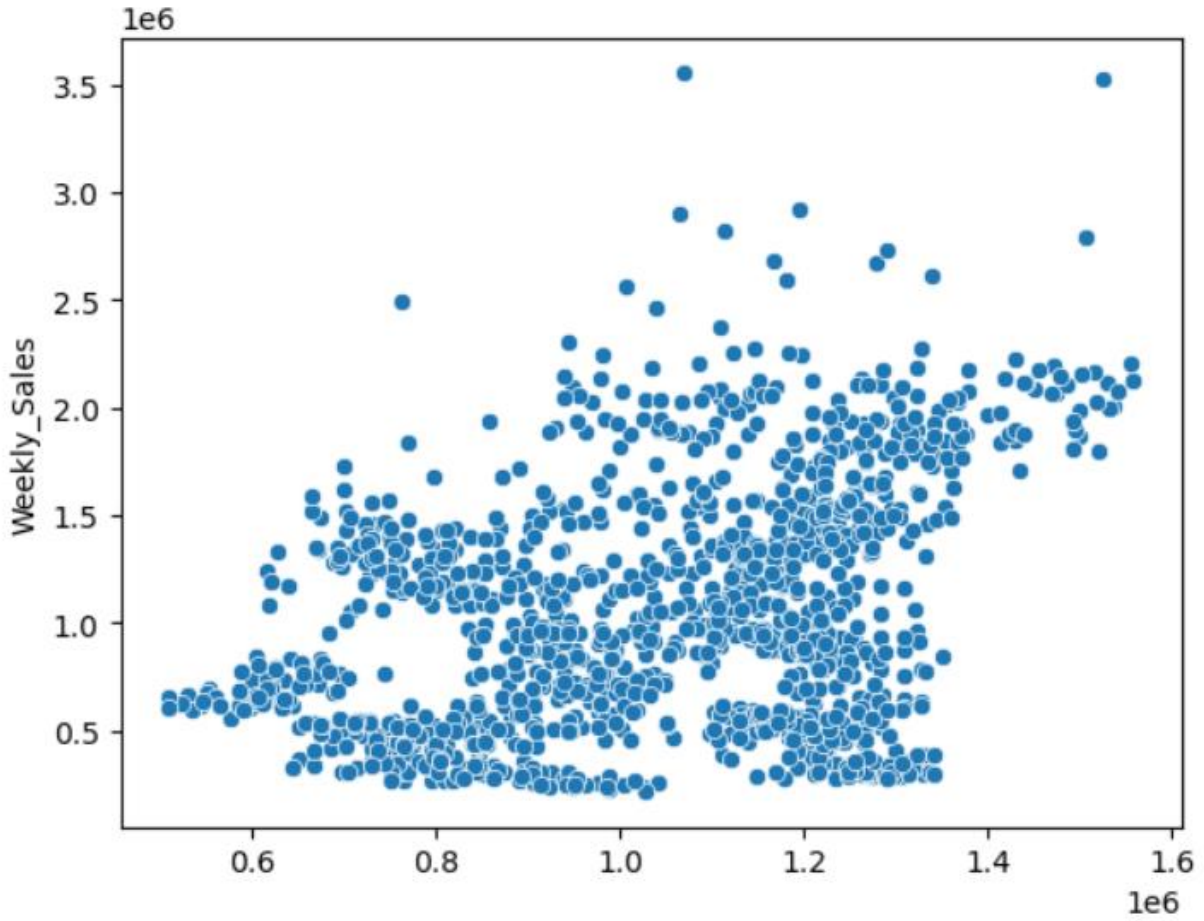
The first model is a Linear Regression model.

```
print('Linear Regression:')
print()
lin_reg = LinearRegression()
lin_reg.fit(X_train, y_train)
y_pred = lin_reg.predict(X_test)
print('Accuracy:',lin_reg.score(X_train, y_train)*100)
print('MAE:', metrics.mean_absolute_error(y_test, y_pred))
print('MSE:', metrics.mean_squared_error(y_test, y_pred))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))

sns.scatterplot(y_pred, y_test)
```

```
Linear Regression:

Accuracy: 14.67056782075895
MAE: 426167.3210650867
MSE: 271541991743.21048
RMSE: 521096.9120453608
```



RANDOM FOREST REGRESSOR

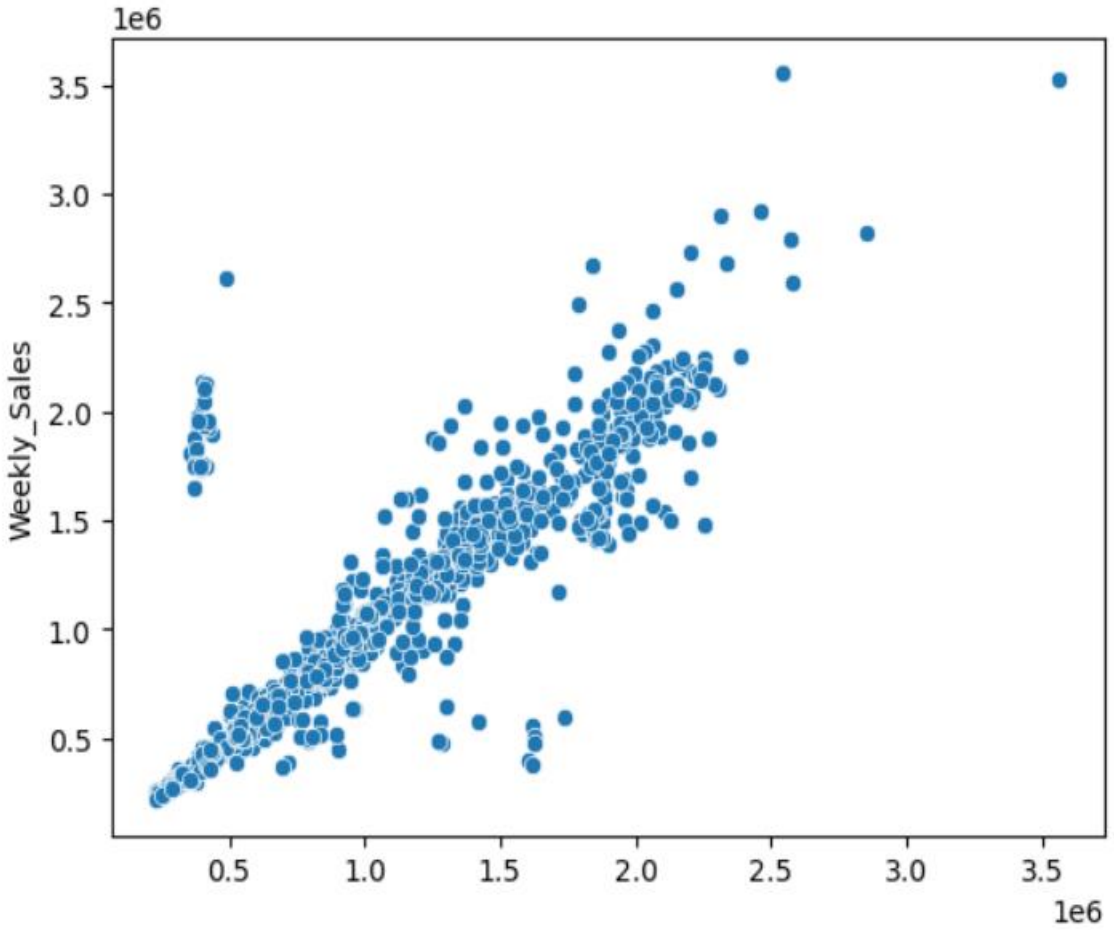
The second model is Random Forest Regressor model

```
print('Random Forest Regressor:')
print()
forest = RandomForestRegressor(n_estimators = 400,max_depth=15,n_jobs=5)
forest.fit(X_train,y_train)
y_pred=forest.predict(X_test)
print('Accuracy:',forest.score(X_test, y_test)*100)
print('MAE:', metrics.mean_absolute_error(y_test, y_pred))
print('MSE:', metrics.mean_squared_error(y_test, y_pred))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))


sns.scatterplot(y_pred, y_test)
```

```
Random Forest Regressor:

Accuracy: 75.06199142252306
MAE: 116590.67203873517
MSE: 78432937146.24594
RMSE: 280058.810156449
```



75.06199142252306

INSIGHTS

After creating two algorithms I have concluded that the Random Forest Regressor (RFR) model is the ideal fit. The accuracy score for RFR was 75.06.%. The score for Linear Regression (LR) model was 14.67%. The line of best fit is a positive indicator showing as X variable increases so does weekly sales. This in turn allows you to forecast weekly sales accurately.

RFR MODEL