UCONN

SCHOOL OF BUSINESS

OPIM -5512
Data Science using Python

Sales Forecasting for Better Business Planning

Group 8

Rushikethan Dudipala

Manisha Kharche

Dhatri Datta Kiran Nutakki

Santosh Kumar Viranchi Gandham

Venkata Vinayaka Durga Prashant Lakamsani

Duggempudi Lokesh Reddy

Table Of Contents:

Section 1: Executive Summary

The aim of this project is to effectively forecast the sales of wholesale retailers to help address business goals and answer common questions that people may have before making any investment or shopping decisions. Also, It helps in better planning of business and can improve their strategy accordingly. Sales forecasting can be used as both a long-term component of a diversified portfolio and a short-term day trading asset.

To conduct the analysis we used Python programming language and obtained a data set from Kaggle. We cleaned the dataset and made it free of inconsistencies and missing values. We then performed Decision Tree, Random Forest ,Exponential Smoothing Model,linear Regression and time series exploration in the modeling part and came to the conclusion that among all models, Random Forest model is the best and most effective model for our analysis.

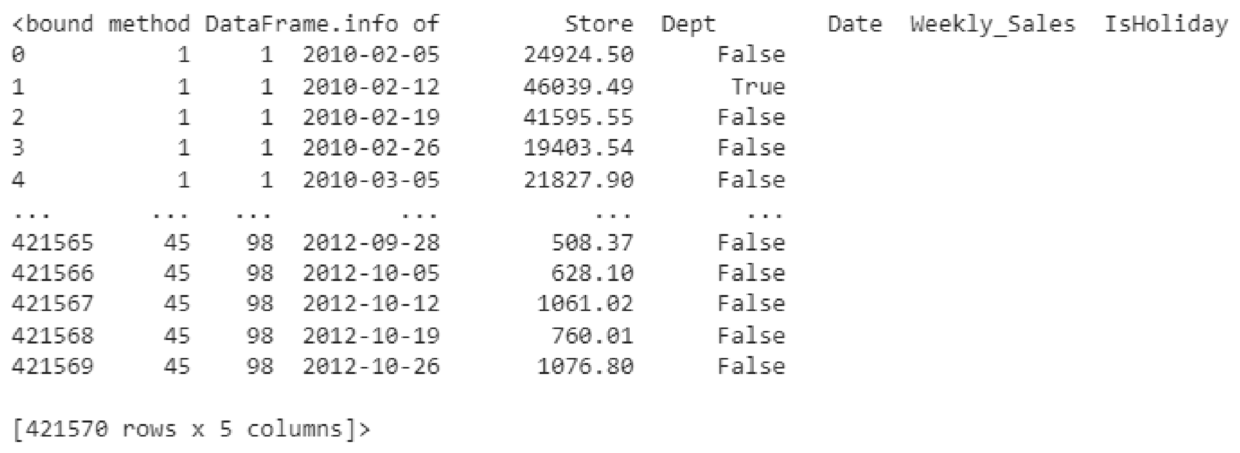
Section 2: Project Introduction.

In today's world, sales forecasting plays a crucial role in making important business decisions. Accurate forecasting allows businesses to plan better, achieve their sales targets, and make meaningful insights from data. The objective of this project is to implement an effective sales forecasting system that provides accurate predictions and enables businesses to identify trends and patterns in sales data to improve decision-making.

Problem statement and research questions: The lack of accurate sales forecasting can result in lost revenue for companies, particularly during peak seasons. Failure to achieve projected sales targets can have a negative impact on stock prices, investor confidence, and the overall reputation of the company. Inaccurate sales forecasting can lead to overproduction, overstocking, and increased waste, which can result in financial losses for the company. The research questions are: Can we create an accurate sales forecasting model? Can we identify different sales patterns in the market? Can we provide meaningful insights to help businesses plan better?

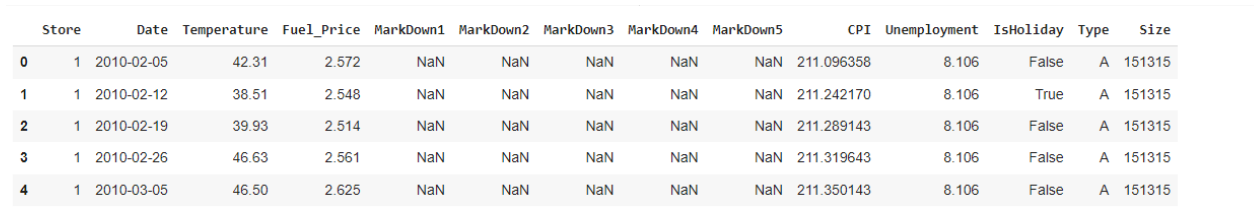
Section 3: Analysis.

Data collection and preparation: The data sets used in this project are stores.csv, train.csv, test.csv, and features.csv. The stores.csv file contains anonymized information about the 45 stores, indicating the type and size of the store. The train.csv file is the historical training data, covering 2010-02-05 to 2012-11-01, containing the store number, department number, date, weekly sales for the given department in the given store, and whether the week is a special holiday week. The test.csv file is identical to train.csv, except the weekly sales are withheld, and we must predict the sales for each triplet of store, department, and date in this file. The features.csv file contains additional data related to the store, department, and regional activity for the given dates, such as the average temperature in the region, the cost of fuel in the region, promotional markdowns, consumer price index, unemployment rate, and whether the week is a special holiday week. Predicting the department-wide weekly sales for each of these stores will be the main objective of this study. The dataset has already been split into separate training and testing data; other than the weekly sales data, the testing data and training dataset are identical. Weekly sales data for the stores and departments from 2010-02-05 to 2012-11-01 are included in the training dataset. Additionally, a column in it indicates whether a given date falls on a holiday or not. The training dataset is summarized in the table below-



```
<bound method DataFrame.info of         Store  Dept        Date  Weekly_Sales  IsHoliday
0           1     1  2010-02-05      24924.50      False
1           1     1  2010-02-12      46039.49       True
2           1     1  2010-02-19      41595.55      False
3           1     1  2010-02-26      19403.54      False
4           1     1  2010-03-05      21827.90      False
...       ...   ...         ...           ...        ...
421565     45    98  2012-09-28        508.37      False
421566     45    98  2012-10-05        628.10      False
421567     45    98  2012-10-12       1061.02      False
421568     45    98  2012-10-19        760.01      False
421569     45    98  2012-10-26       1076.80      False

[421570 rows x 5 columns]>
```

Below is the summary of the feature dataset-



	Store	Date	Temperature	Fuel_Price	MarkDown1	MarkDown2	MarkDown3	MarkDown4	MarkDown5	CPI	Unemployment	IsHoliday	Type	Size
0	1	2010-02-05	42.31	2.572	NaN	NaN	NaN	NaN	NaN	211.096358	8.106	False	A	151315
1	1	2010-02-12	38.51	2.548	NaN	NaN	NaN	NaN	NaN	211.242170	8.106	True	A	151315
2	1	2010-02-19	39.93	2.514	NaN	NaN	NaN	NaN	NaN	211.289143	8.106	False	A	151315
3	1	2010-02-26	46.63	2.561	NaN	NaN	NaN	NaN	NaN	211.319643	8.106	False	A	151315
4	1	2010-03-05	46.50	2.625	NaN	NaN	NaN	NaN	NaN	211.350143	8.106	False	A	151315

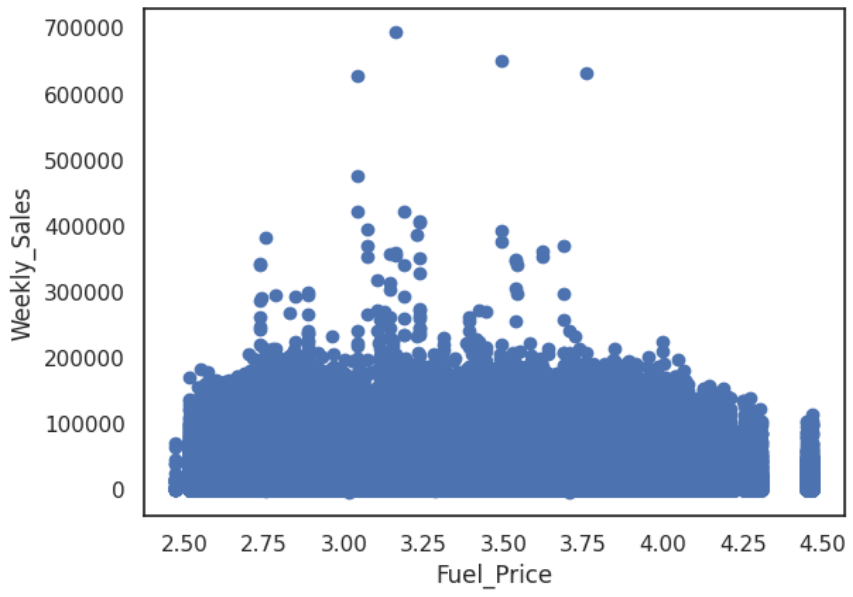
Section 3A: Data Exploration and Data Visualization

The data was explored to find hidden patterns and insights using visualization techniques. The correlations between different features were also explored. As expected, holiday average sales were higher than normal dates. Sales seemed to be highest during the week of Thanksgiving and 3 weeks after Thanksgiving and Christmas time. The data was also split into train and test sets for model selection and evaluation.It is crucial to have an in-depth understanding of the dataset that is used in this analysis to understand the models that would give the most accurate prediction. Several times there are underlying patterns or trends in the data that would not be identified as easily, hence the need for an extensive exploratory data analysis. This thorough examination is necessary to understand the underlying structure of the dataset and to draw conclusions or insight about the validity of our analysis.

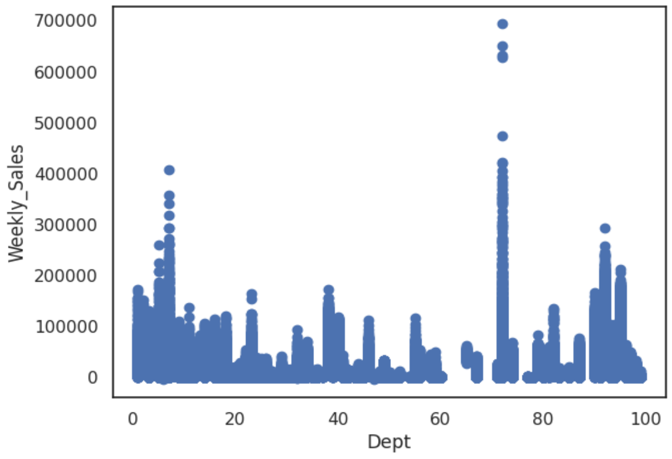
In this study, several other packages, including 'ggplot2','matplotlib', and'seaborn', were also used to produce visualizations that show data on weekly sales by store and department, weekly sales on holidays compared to on regular days, weekly sales based on region, store type, and store size, average sales per year, change in sales as a result of factors like CPI, fuel price, temperature, and unemployment, etc.The findings and prospective modeling work that will be done in the project's later stages are discussed in brief explanations that go along with these visualizations.

Impact of various attributes on the weekly Sales are shown below:

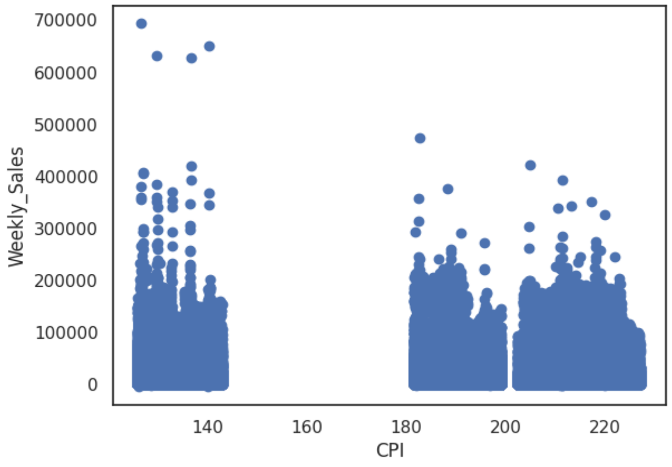
When the correlation between the size of stores and sales is examined more closely, a linear pattern may be seen, which suggests that sales generally increase as shop size increases. There are a few exceptions, nevertheless, when smaller Type "B" stores have outperformed their larger Type "A" in terms of overall sales. This shift from the normal pattern raises the possibility that other elements, such as geographic location and marketing tactics, may be having an impact on sales success. In conclusion, the evidence supports the general trend even though there may be a few small outliers.



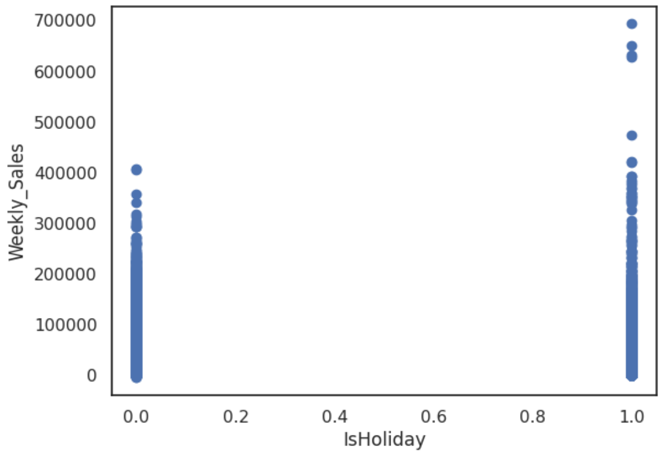
A study of the data was done to learn more about the sales outcomes of various departments in various shop types. Based on the findings, it is clear that Department 72 has the most weekly sales among all store types. This discovery raises the possibility that this division may play a significant role in how well retail outlets function overall. It is crucial to remember that this is only one element of the bigger picture and that other divisions could also have a big impact on sales.



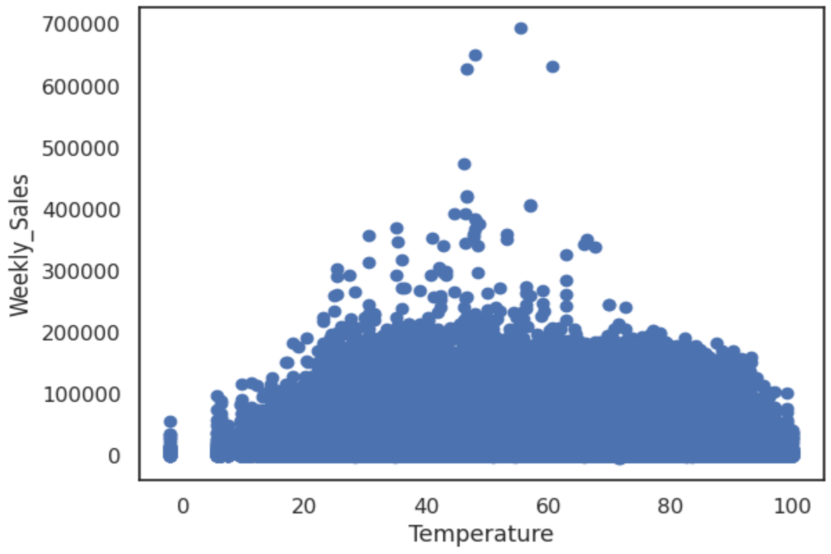
A popular metric for determining how price changes affect people's cost of living is the Consumer Price Index (CPI). Three unique clusters were discovered when the data was examined based on various CPI values. Surprisingly, there doesn't seem to be a connection between weekly sales changes at Walmart shops and CPI increases, suggesting that sales go on even when CPI rates are high. It is important to point out that there had been a small but intriguing finding: Type B retailers sold more goods than other store types when the CPI was low (more precisely, when it was 140). Although this finding might seem insignificant, it might indicate that Type B stores are more sensitive to changes in CPI than other store types.



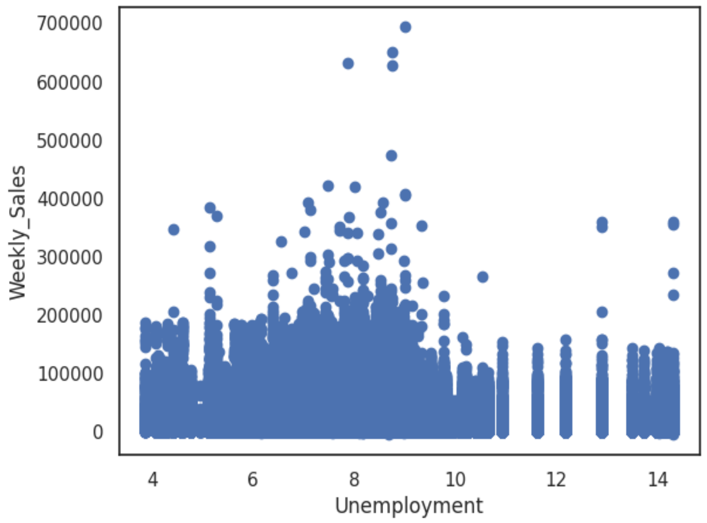
A comparison of sales during festive days and those at regular days was done to get insight into whether holiday times contribute to increased sales statistics. The data indicated that, while accounting for just around 7% of the overall days in a year, festive dates provided much higher weekly sales results than all other days of the calendar year . This research highlights the enormous influence of holiday periods on retail sales performance, as well as the need of capitalizing on such opportunities to optimize revenue and profitability.



Weather has a tremendous influence on sales, as the retail sector is well aware. Warmer weather frequently boosts sales, but chilly or extremely hot weather discourages people from going out to make purchases. The scatter plot demonstrates that a majority of store types have their highest sales when temperatures are between 40F and 80F. Regardless of the fact that sales are predicted to be reduced in freezing or hot conditions, they are still sufficiently significant when the climate is pleasant.

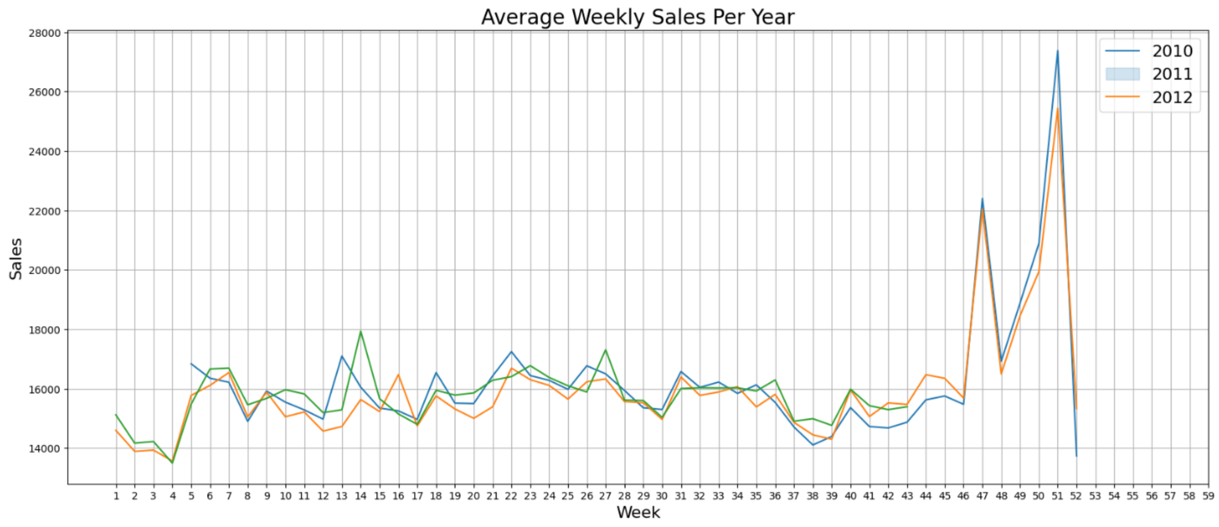


We may get a number of significant conclusions thanks to our scatter plot representation. First, once the unemployment rate exceeds 11, we see a definite fall in revenues for the designated store categories. Surprisingly, despite the general reduction in sales,some stores do not see a major decline in average sales. Additionally, we observe a substantial drop in revenue when the unemployment index rises. It's interesting to note that sales for a few stores peak between 8 and 10 percent .



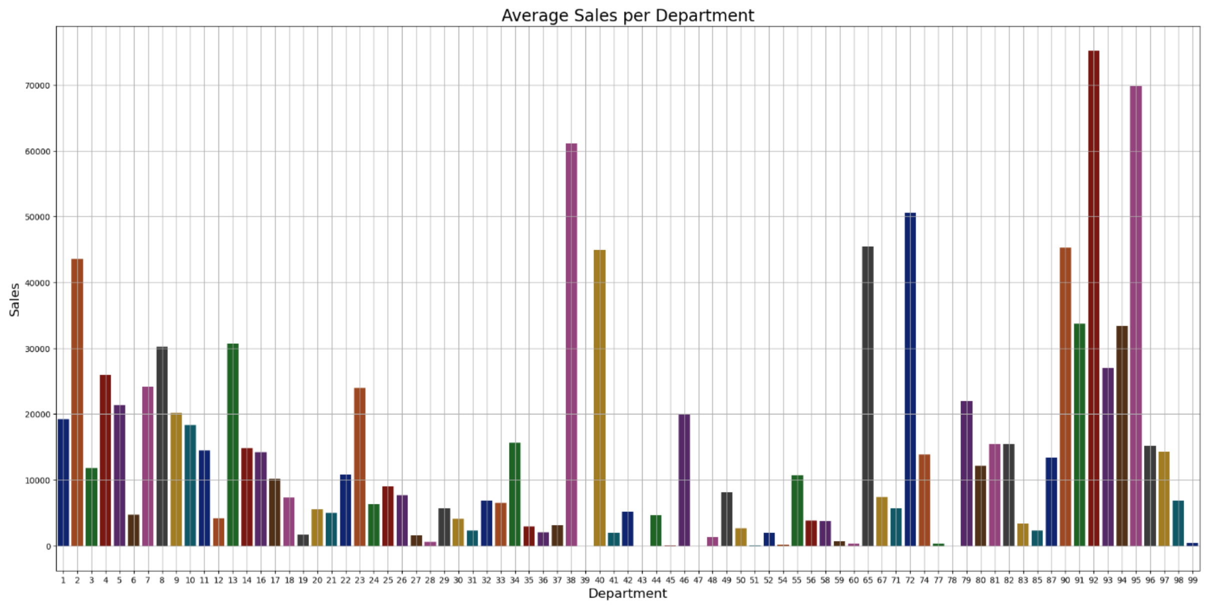
Section 3B: Data Modeling and Forecasting Section

Average weekly sales per Year -

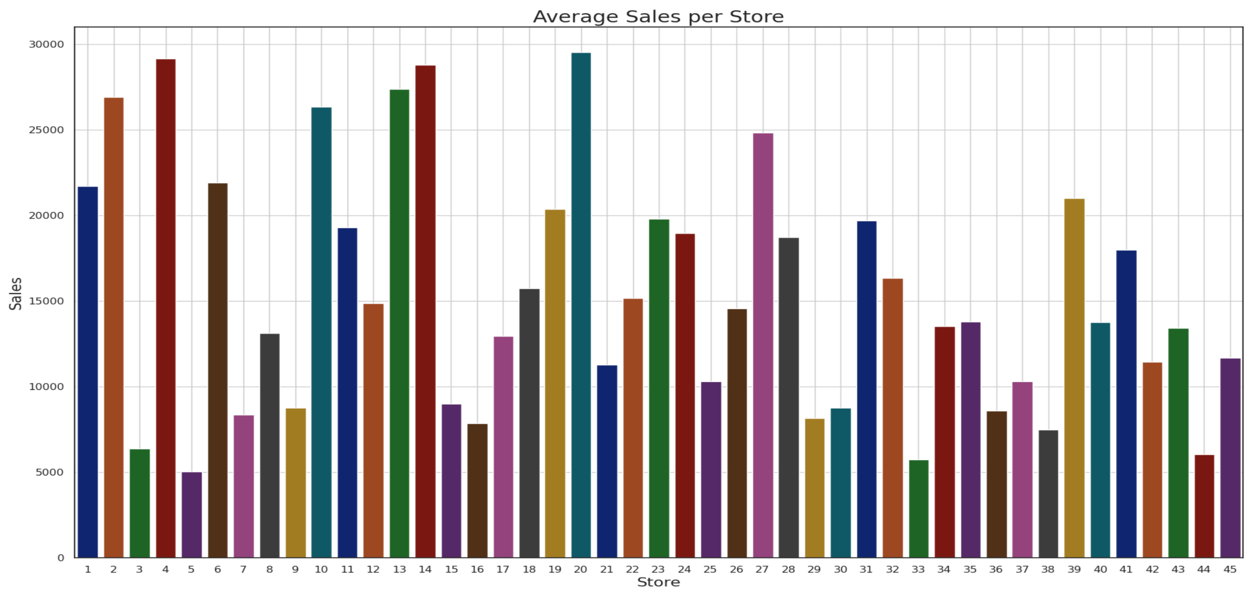


The week-over-week comparison once more enables us to determine whether sales grow during annual holiday weeks, such as the weeks of Thanksgiving, Christmas, Labor Day, etc. Sales clearly increase in the weeks 47 and 51 that correspond to Thanksgiving and Christmas, respectively, demonstrating once more that the holiday season is a time when sales increase. These findings have only been drawn using the data available from 2010 and 2011 due to the dearth of data for the year 2012. This graph also shows that there is a clear trend of drop right after Christmas and New Year's.

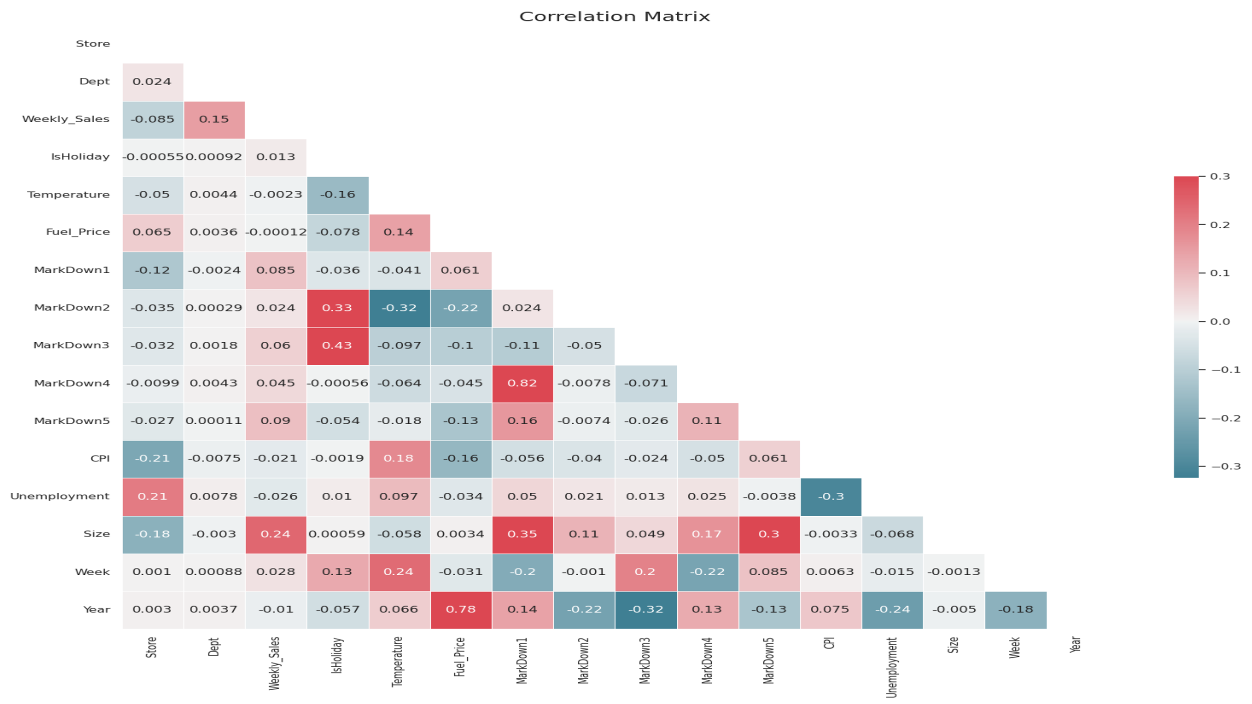
Average Sales per Department-



Average Sales per Stores-



A correlation matrix describes the correlation between the various variables of a dataset. Each variable in the table is correlated to each of the other variables in the table and helps in understanding which variables are more closely related to each other.



- There is a slight correlation between weekly sales and store size, type, and department
- There seems to be a negative correlation between weekly sales and temperature, unemployment, CPI, and fuel price. This could suggest that sales are not impacted by changes in these factors
- Markdowns 1-5 also seem to have no distinct correlation with weekly sales, thus they are not as important a factor in the study.

Data preprocessing and feature engineering: The data was cleaned, processed, and prepared for analysis by handling missing values, scaling features, and encoding categorical variables. Feature engineering was performed to create new features that capture important trends and patterns in the data.

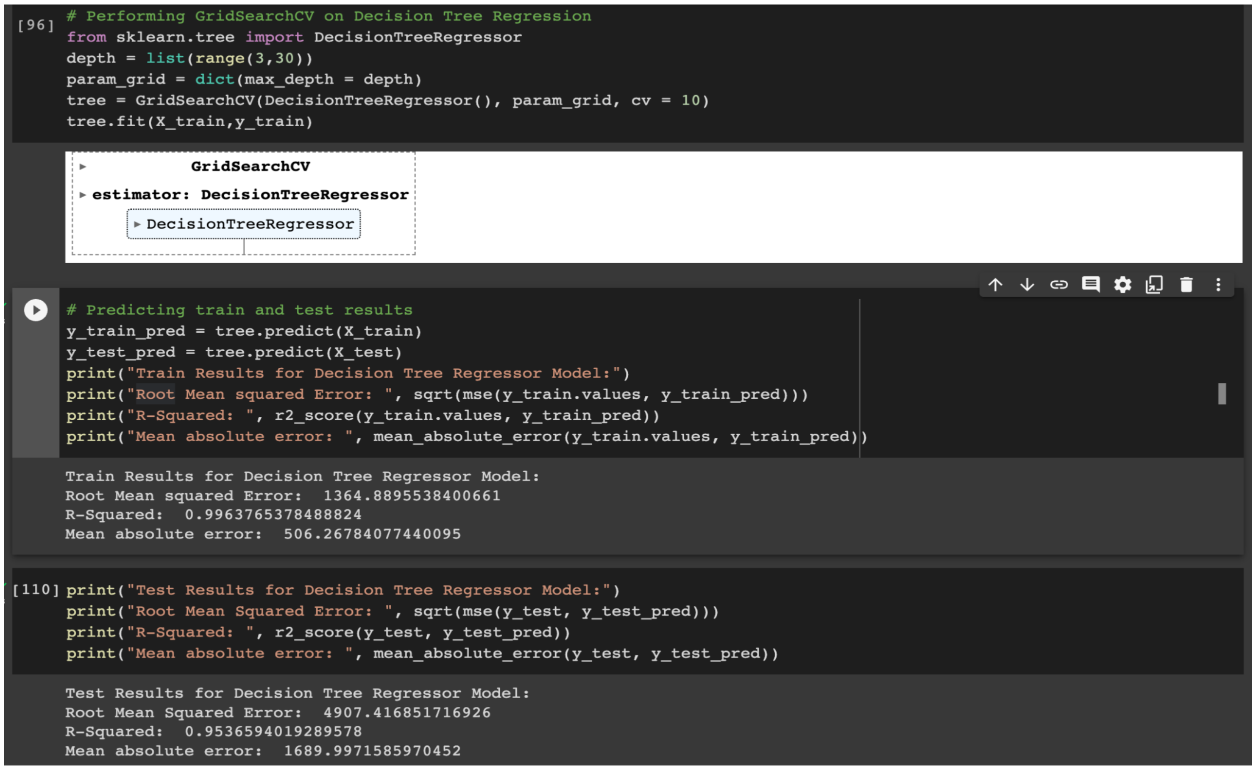
Splitted with Train set of 70% and Test set of 30%

Section 4: Modeling

Machine learning model selection and evaluation: The following machine learning models were selected for forecasting the sales:

1. Decision Tree Regression
2. Random Forest Regression
3. Time Series Model (Prophet Model)
4. Exponential Smoothing Model
5. Linear Regression

1. Decision Tree Regression



```
[96] # Performing GridSearchCV on Decision Tree Regression
     from sklearn.tree import DecisionTreeRegressor
     depth = list(range(3,30))
     param_grid = dict(max_depth = depth)
     tree = GridSearchCV(DecisionTreeRegressor(), param_grid, cv = 10)
     tree.fit(X_train,y_train)
```

```
GridSearchCV
estimator: DecisionTreeRegressor
DecisionTreeRegressor
```

```
# Predicting train and test results
y_train_pred = tree.predict(X_train)
y_test_pred = tree.predict(X_test)
print("Train Results for Decision Tree Regressor Model:")
print("Root Mean squared Error: ", sqrt(mse(y_train.values, y_train_pred)))
print("R-Squared: ", r2_score(y_train.values, y_train_pred))
print("Mean absolute error: ", mean_absolute_error(y_train.values, y_train_pred))
```

```
Train Results for Decision Tree Regressor Model:
Root Mean squared Error:  1364.8895538400661
R-Squared:  0.9963765378488824
Mean absolute error:  506.26784077440095
```

```
[110] print("Test Results for Decision Tree Regressor Model:")
      print("Root Mean Squared Error: ", sqrt(mse(y_test, y_test_pred)))
      print("R-Squared: ", r2_score(y_test, y_test_pred))
      print("Mean absolute error: ", mean_absolute_error(y_test, y_test_pred))
```

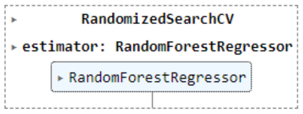
```
Test Results for Decision Tree Regressor Model:
Root Mean Squared Error:  4907.416851716926
R-Squared:  0.9536594019289578
Mean absolute error:  1689.9971585970452
```

We perform a grid search to find the optimal value of the hyperparameter 'max_depth' for the decision tree regression model. The range of values for 'max_depth' is defined as a list ranging from 3 to 29, and then passed to the 'param_grid' dictionary. The GridSearchCV function from the Scikit-learn library is used to perform a 10-fold cross-validation on the training data to identify the best value of 'max_depth' that minimizes the mean squared error (MSE) of the model. The optimal value of 'max_depth' is obtained using the 'best_params_' attribute of the GridSearchCV object.

After obtaining the optimal hyperparameter value, the decision tree regression model is fit on the training data using this value. The 'predict' method is used to obtain the predicted values of the target variable for both the training and testing data. The root mean squared error (RMSE), R-squared (R2), and mean absolute error (MAE) are calculated as evaluation metrics for the model's performance on the training and test data.

2.Random Forest

```
# Performing RandomsearchCV on Random Forest Regression
from sklearn.ensemble import RandomForestRegressor
tuned_params = {'n_estimators': [100, 200], 'min_samples_split': [2, 5, 10], 'min_samples_leaf': [1, 2, 4]}
random_regressor = RandomizedSearchCV(RandomForestRegressor(), tuned_params, n_iter = 3, scoring = 'neg_mean_absolute_error', cv = 3, n_jobs = -1)
random_regressor.fit(X_train, y_train)
```



```
# Predicting train and test results
y_train_pred = random_regressor.predict(X_train)
y_test_pred = random_regressor.predict(X_test)
```

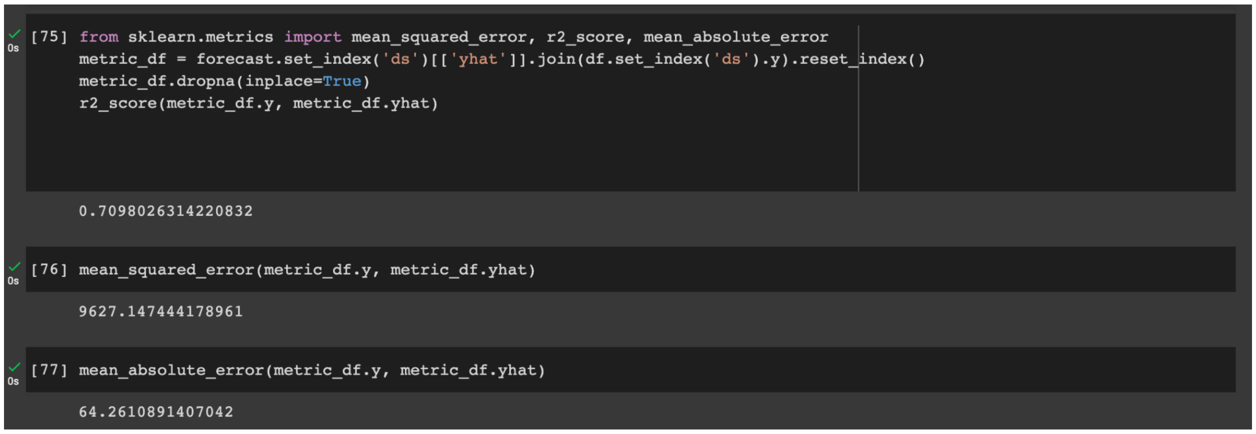
```
## R square and mean sqaure values for Forest Regression model
from sklearn.metrics import r2_score, mean_squared_error
# Predict using the trained model
y_pred = random_regressor.predict(X_test)
# Calculate R-squared
r2 = r2_score(y_test, y_pred)

# Calculate RMSE
rmse = mean_squared_error(y_test, y_pred, squared=False)
print("R-squared: {:.2f}".format(r2))
print("RMSE: {:.2f}".format(rmse))
print("Mean absolute error: ", mean_absolute_error(y_test, y_pred))
```

```
R-squared: 0.97
RMSE: 3720.11
Mean absolute error:  1391.6370590439703
```

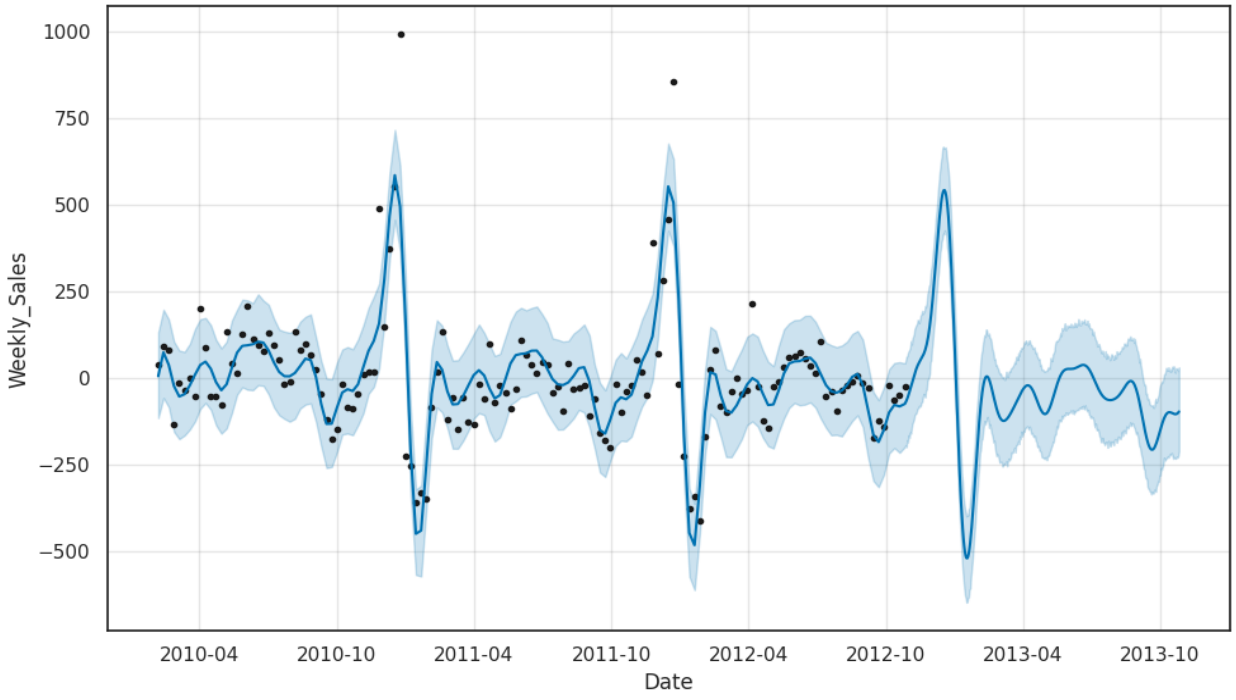
Next, we performed a random search to find the optimal combination of hyperparameters for the random forest regression model. A dictionary named 'tuned_params' is defined, which contains the values of hyperparameters such as 'n_estimators', 'min_samples_split', and 'min_samples_leaf'. The RandomizedSearchCV function from the Scikit-learn library is used to randomly select a combination of hyperparameters from the 'tuned_params' dictionary. The number of iterations of the random search is set to 3. The mean absolute error (MAE) is chosen as the evaluation metric, which is computed using a 3-fold cross-validation on the training data. The trained model is then used to predict the target variable for both the training and testing data.

Next, the R-squared and root mean squared error (RMSE) is calculated as evaluation metrics for the performance of the random forest regression model on the testing and training data sets. This gives an estimate of the average difference between the actual and predicted values of the target variable.

3. Time Series Model (Prophet Model)



```
[75] from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error
     metric_df = forecast.set_index('ds')[['yhat']].join(df.set_index('ds').y).reset_index()
     metric_df.dropna(inplace=True)
     r2_score(metric_df.y, metric_df.yhat)

     0.7098026314220832

[76] mean_squared_error(metric_df.y, metric_df.yhat)

     9627.147444178961

[77] mean_absolute_error(metric_df.y, metric_df.yhat)

     64.2610891407042
```



The above code performs customer segmentation using the K-Means clustering algorithm. Firstly, the 'df_features' DataFrame is backed up using the 'copy' method to create a deep copy of the original DataFrame. Next, the 'LabelEncoder' and 'StandardScaler' classes from the Scikit-learn library are imported. The 'LabelEncoder' class is used to convert categorical variables such as 'Store', 'Weekly_Sales', 'Dept', and 'IsHoliday' into numerical values, while the 'StandardScaler' class is used to standardize the values of the 'Store', 'Weekly_Sales', and 'Dept' variables.

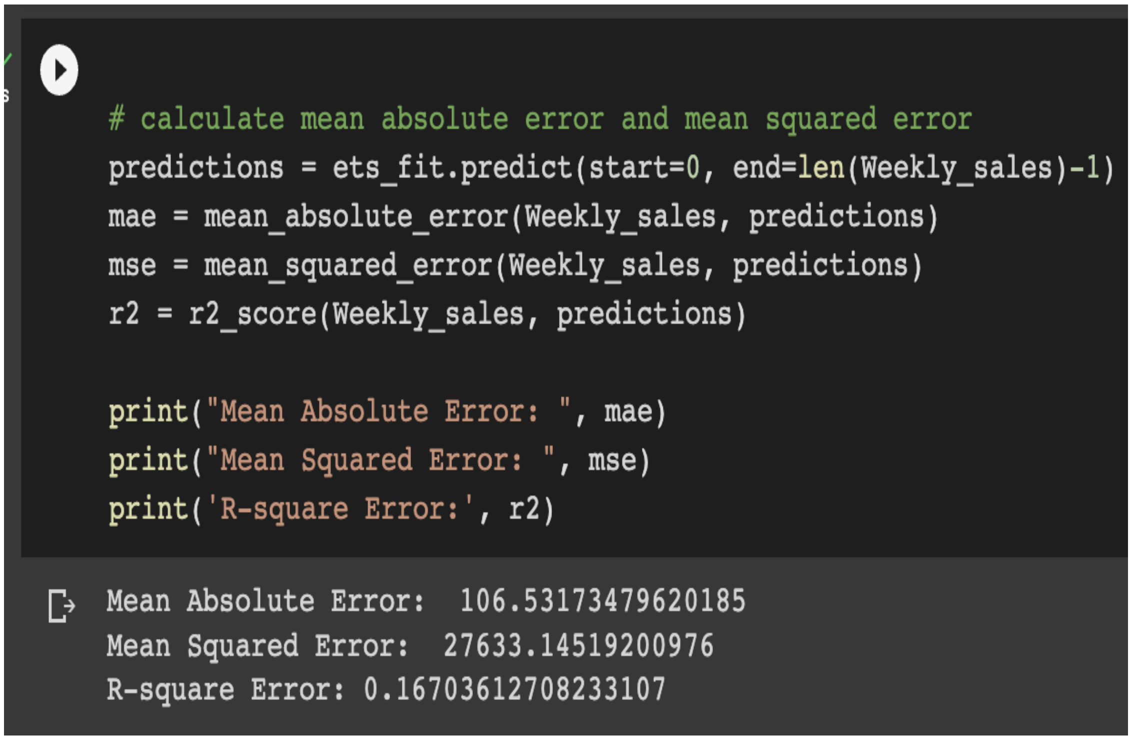
Next, the 'KMeans' class is imported from Scikit-learn and instantiated with the number of clusters set to 4 and a random state of 42. The 'fit_predict' method of the 'KMeans' class is used to fit the model to the standardized variables of 'Store', 'Weekly_Sales', and 'Dept' and predict the cluster labels for each observation. Finally, a scatterplot is created using the 'sns.scatterplot' function from the Seaborn library, with the 'Weekly_Sales' and 'Store' variables plotted against each other and colored according to the predicted cluster labels. This allows for the visualization of the customer segmentation based on their purchasing behavior, which can be used to identify groups of customers with similar behavior and create targeted marketing strategies to improve sales.

4. Exponential Smoothing Model

The code is performing single exponential smoothing or ETS (Error, Trend, Seasonality) on a time series data called "Weekly_sales". The ETS model is used to forecast the future values of the time series by modeling its trend and seasonality.

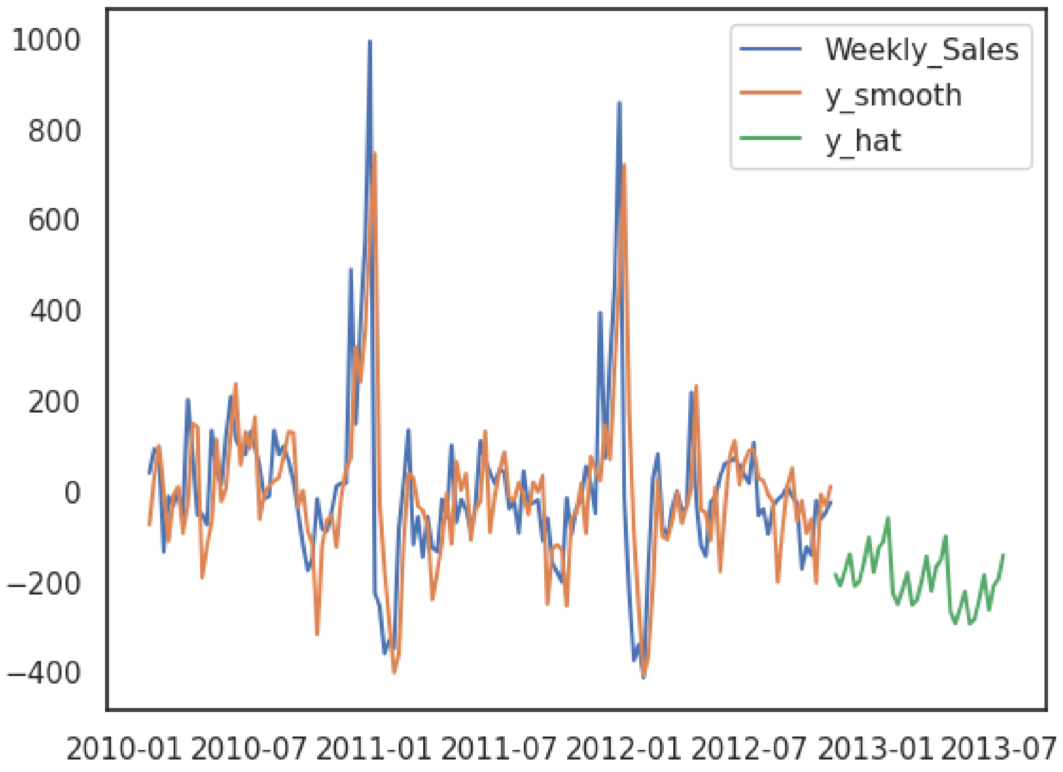
The parameters alpha, beta, and gamma are the smoothing parameters used to estimate the level, trend, and seasonality components of the time series, respectively. The model is initialized with these parameters and is fitted using the "fit" method of the ExponentialSmoothing class from statsmodels.

Once the model is fitted, then we used to forecast the future values of the time series using the "forecast" method, which takes the number of future periods to be forecasted as an argument. Finally, the original time series, the smoothed values, and the forecasted values are plotted using Matplotlib.



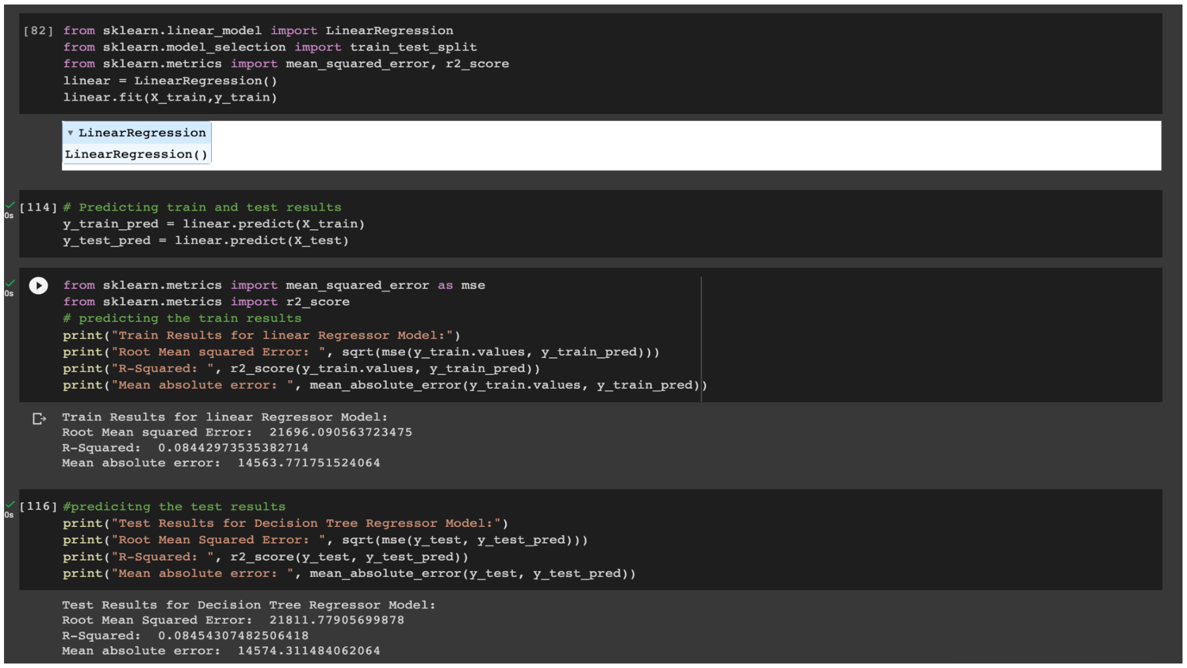
```
# calculate mean absolute error and mean squared error
predictions = ets_fit.predict(start=0, end=len(Weekly_sales)-1)
mae = mean_absolute_error(Weekly_sales, predictions)
mse = mean_squared_error(Weekly_sales, predictions)
r2 = r2_score(Weekly_sales, predictions)

print("Mean Absolute Error: ", mae)
print("Mean Squared Error: ", mse)
print('R-square Error:', r2)
```

```
Mean Absolute Error:  106.53173479620185
Mean Squared Error:  27633.14519200976
R-square Error: 0.16703612708233107
```



5. Linear Regression-

The ‘scikit-learn’ library along with the ‘Linear Regression’ function in Python has been used to create the linear regression model.We got MAE -14574.3114



```
[82] from sklearn.linear_model import LinearRegression
     from sklearn.model_selection import train_test_split
     from sklearn.metrics import mean_squared_error, r2_score
     linear = LinearRegression()
     linear.fit(X_train,y_train)
```

```
▾ LinearRegression
LinearRegression()
```

```
[114] # Predicting train and test results
      y_train_pred = linear.predict(X_train)
      y_test_pred = linear.predict(X_test)
```

```
from sklearn.metrics import mean_squared_error as mse
from sklearn.metrics import r2_score
# predicting the train results
print("Train Results for linear Regressor Model:")
print("Root Mean squared Error: ", sqrt(mse(y_train.values, y_train_pred)))
print("R-Squared: ", r2_score(y_train.values, y_train_pred))
print("Mean absolute error: ", mean_absolute_error(y_train.values, y_train_pred))
```

```
Train Results for linear Regressor Model:
Root Mean squared Error:  21696.090563723475
R-Squared:  0.08442973535382714
Mean absolute error:  14563.771751524064
```

```
[116] #predicitng the test results
      print("Test Results for Decision Tree Regressor Model:")
      print("Root Mean Squared Error: ", sqrt(mse(y_test, y_test_pred)))
      print("R-Squared: ", r2_score(y_test, y_test_pred))
      print("Mean absolute error: ", mean_absolute_error(y_test, y_test_pred))
```

```
Test Results for Decision Tree Regressor Model:
Root Mean Squared Error:  21811.77905699878
R-Squared:  0.08454307482506418
Mean absolute error:  14574.311484062064
```

Section 5: Model Comparison

Model	R-Square(R2)	RMSE	MAE
Decision Tree Regression	0.953	4907.4	1689.9
Random Forest Regression	0.97	3912.1	1463.4
Time Series Prophet Model	0.71	98.2	64.26
Exponential Smoothing Model	0.167	27633.2	106.5
Linear Regression	0.08	21811.8	14574.3

The models were trained and evaluated using different performance metrics such as R-Square, Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE).

Results and performance metrics: The Random Forest Regression model provided the best results in terms of R-Square (0.97), RMSE (3912.1), and MAE (1463.4). The Time Series Prophet Model provided the lowest R-Square (0.71), RMSE (98.2), and MAE (64.26). The Exponential Smoothing Model and Linear Regression provided the worst results in terms of R-Square, RMSE, and MAE.

Section 7: Conclusions/ Business Values

From the results obtained, it can be concluded that Random Forest Regression is the most accurate model for sales forecasting.

Store with high sales :Type A stores has big store size ,large number of departments and small markdown values.

- Stores with high sales : Type A stores has big store size,large number of departments and small markdown values.
- Stores with low sales: Type C stores has small store size,small number of departments and low markdown values.
- Promotional Markdown events before holidays seem to increase sales.
- Sales seems to be highest during the week of Thanksgiving and 3 weeks after Thanksgiving and Christmas time.
- An important aspect of this is to try and understand buying behavior based on regional and departmental sales.
- Walmart can grow its online retail business massively and gather huge profits with already established stores and warehouses.
- It is easier for the organization to create a nationwide reach,limiting the presence of physical stores and helping their customers save on fuel costs by delivering goods at their doorstep.

Section 8: References

1. Walmart sales data: KaggleSource: "https://www.kaggle.com/datasets/aslanahmedov/walmart-sales-forecast?datasetId=2107830&sortBy=voteCount&select=features.csv"
2. Walmart Sales Forecast Source : Kaggel "TP2 - Walmart Sales Forecast | Kaggle".
3. Codes from OPIM-5512-Data Science using Python-SECB13-1233
4. Predictive Modeling
5. Chat GPT