Capstone Project- Walmart Sales Data Project

Submitted By- Diwakar Acharya

Table of Contents

Problem Statement

A retail store that has multiple outlets across the country are facing issues in managing the inventory- to match the demand with respect to supply.

Project Objective

With the project we have to come up with the useful insights using the data and make prediction models to forecast the sales for X number of months/years.

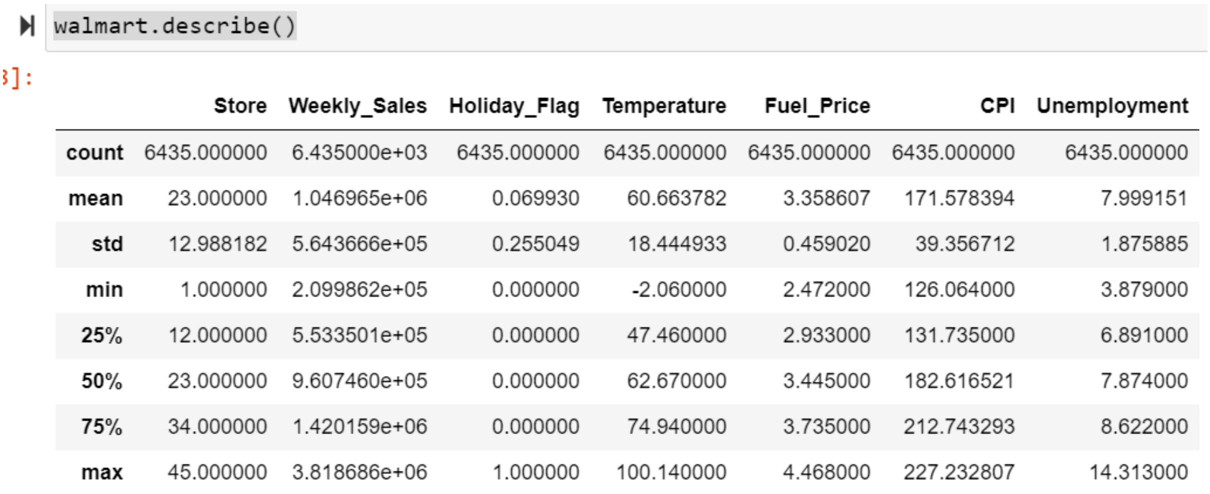
Data Description

Given data is Walmart sales data in CSV format. It is having 6435 rows and 8 columns. These columns represents features name.

Feature Name	Description
Store	Store number
Date	Week of Sales
Weekly_Sales	Sales for the given store in that week
Holiday_Flag	If it is a holiday week
Temperature	Temperature on the day of the sale
Fuel_Price	Cost of the fuel in the region
CPI	Consumer Price Index
Unemployment	Unemployment Rate

Insights from the data:

1) All statistical figure

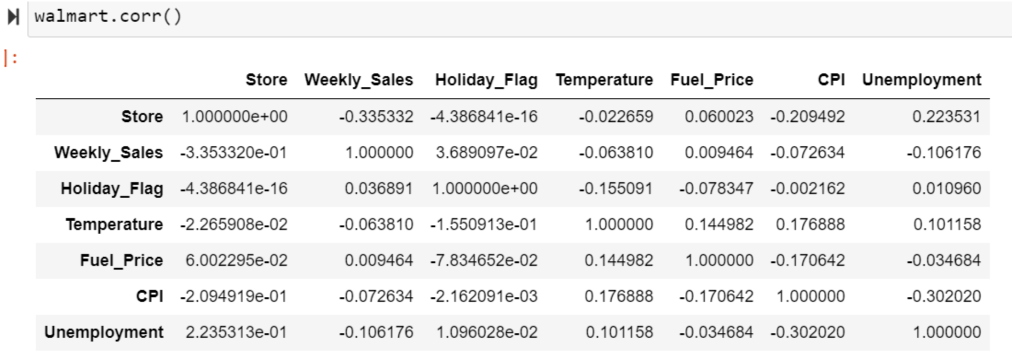


```
walmart.describe()
```

]:

	Store	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment
count	6435.000000	6.435000e+03	6435.000000	6435.000000	6435.000000	6435.000000	6435.000000
mean	23.000000	1.046965e+06	0.069930	60.663782	3.358607	171.578394	7.999151
std	12.988182	5.643666e+05	0.255049	18.444933	0.459020	39.356712	1.875885
min	1.000000	2.099862e+05	0.000000	-2.060000	2.472000	126.064000	3.879000
25%	12.000000	5.533501e+05	0.000000	47.460000	2.933000	131.735000	6.891000
50%	23.000000	9.607460e+05	0.000000	62.670000	3.445000	182.616521	7.874000
75%	34.000000	1.420159e+06	0.000000	74.940000	3.735000	212.743293	8.622000
max	45.000000	3.818686e+06	1.000000	100.140000	4.468000	227.232807	14.313000

2) Correlation between features:

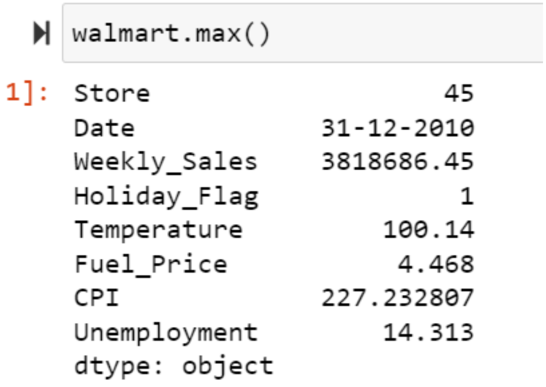


```
walmart.corr()
```

]:

	Store	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment
Store	1.000000e+00	-0.335332	-4.386841e-16	-0.022659	0.060023	-0.209492	0.223531
Weekly_Sales	-3.353320e-01	1.000000	3.689097e-02	-0.063810	0.009464	-0.072634	-0.106176
Holiday_Flag	-4.386841e-16	0.036891	1.000000e+00	-0.155091	-0.078347	-0.002162	0.010960
Temperature	-2.265908e-02	-0.063810	-1.550913e-01	1.000000	0.144982	0.176888	0.101158
Fuel_Price	6.002295e-02	0.009464	-7.834652e-02	0.144982	1.000000	-0.170642	-0.034684
CPI	-2.094919e-01	-0.072634	-2.162091e-03	0.176888	-0.170642	1.000000	-0.302020
Unemployment	2.235313e-01	-0.106176	1.096028e-02	0.101158	-0.034684	-0.302020	1.000000

3) Maximum



```
walmart.max()
```

```
1]: Store                   45
    Date            31-12-2010
    Weekly_Sales    3818686.45
    Holiday_Flag             1
    Temperature         100.14
    Fuel_Price           4.468
    CPI             227.232807
    Unemployment        14.313
    dtype: object
```

1) **Statistics insights that can be used by each of the stores to improve in various areas:**

I have analysed below task and basis insights is provided.

1. **Which store has maximum sales?**

   Sales_groupby = walmart.groupby('Store')['Weekly_Sales'].sum()

   print("Store Number {} has maximum Sales. Sum of Total Sales {}".

   format(Sales_groupby.idxmax(),Sales_groupby.max()))

Result- Store Number 20 has maximum Sales. Sum of Total Sales 301397792.46

2. **Which store has maximum standard deviation i.e., the sales vary a lot. Also, will find out the coefficient of mean to standard deviation.**

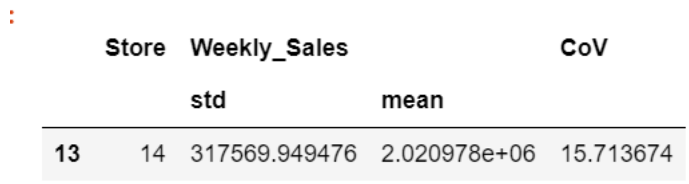
maxstd=pd.DataFrame(walmart.groupby('Store').agg({'Weekly_Sales':['std','mean']}))

maxstd = maxstd.reset_index()

maxstd['CoV'] =(maxstd[('Weekly_Sales','std')]/maxstd[('Weekly_Sales','mean')])*100

maxstd.loc[maxstd[('Weekly_Sales','std')]==maxstd[('Weekly_Sales','std')].max()]

Result- Store 14 has maximum deviation in sales.



	Store	Weekly_Sales		CoV
		std	mean	
13	14	317569.949476	2.020978e+06	15.713674

3. **Which store/s has good quarterly growth rate in Q3’2012?**

Qrt_growth = walmart.groupby('Store').agg({'Weekly_Sales':['mean','std']})

```
Qrt_growth.head()

data_Q32012 = walmart[(pd.to_datetime(walmart['Date']) >= pd.to_datetime('07-01-2012')) & (pd.to_datetime(walmart['Date']) <= pd.to_datetime('09-30-2012'))]

data_growth = data_Q32012.groupby(['Store'])['Weekly_Sales'].sum()

print("Store Number {} has Good Quartely Growth in Q3'2012 {}".format(data_growth.idxmax(),data_growth.max()))
```

Result- Store Number 4 has good Quarterly growth in Q3'2012- 25652119.35

4. **Some holidays have a negative impact on sales. Find out holidays which have higher sales than the mean sales in non-holiday season for all stores together**

Result- The sales increased during thanksgiving and the sales decreased during Christmas.

5. **Provide a monthly and semester view of sales in units and give insights.**

Result- Monthly Sales Graph indicates that highest sum of sales is recorded in between jan-2011 to march-2011.

Quarterly Sales Graph indicates that highest sum of sales is recorded in Q1 of 2011 and 2012.

Semester Sales graph indicates that at beginning of 1st sem of 2010 and 1st sem of 2013 sales are lowest.

6. **Analysing data wrt. weekly sales**

```
def scatter(walmart, column):

    plt.figure()

    plt.scatter(walmart[column] , walmart['Weekly_Sales'])

    plt.ylabel('Weekly_Sales')

    plt.xlabel(column)

scatter(walmart, 'Fuel_Price')  # with respect to Fuel_Price

scatter(walmart, 'Date')  # with respect to date

scatter(walmart, 'CPI')  # with respect to CPI

scatter(walmart, 'Holiday_Flag') # with respect to Holiday

scatter(walmart, 'Unemployment')  # with respect to Unemployment

scatter(walmart, 'Temperature') # with respect to Temperature

scatter(walmart, 'Store') # with respect to Store
```

2. Various Insights by data visualisation

1) When the temp is between 20 to 60, weekly sales have outliers and are on higher side.

2) Weekly sales are highest when fuel_d price is between 2.75 to 3.75.

3) When there is a holiday, weekly sales shoot up.

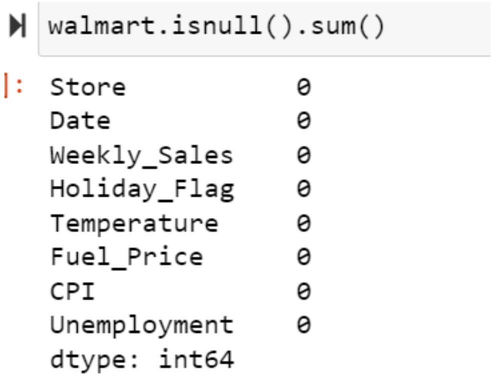
4) When CPI is between 140 to 180 weekly sales is very less.

5) When unemployment is going beyond 9 %, weekly sales is drastically falling.

Data pre-processing Steps and Inspiration

1) Data cleaning- The data can have many irrelevant and missing parts. To handle this part, data cleaning is done. It involves handling of missing data, noisy data etc

   Here, we don’t have any missing values.

   ```
   walmart.isnull().sum()
   ```

   ```
   Store           0
   Date            0
   Weekly_Sales    0
   Holiday_Flag    0
   Temperature     0
   Fuel_Price      0
   CPI             0
   Unemployment    0
   dtype: int64
   ```

2) Outliers- Identifying and removing outliers

   ```
   import seaborn as sns
   # find outliers
   fig, axs = plt.subplots(4,figsize=(6,18))
   X = walmart[['Temperature','Fuel_Price','CPI','Unemployment']]
   for i,column in enumerate(X):
       sns.boxplot(walmart[column], ax=axs[i])
   # drop the outliers
   without_outlier = walmart[(walmart['Unemployment']<10) & (walmart['Unemployment']>4.5) & (walmart['Temperature']>10)]
   without_outlier
   ```

3) Data integration: Combining multiple data sources- Here, there is only one source i.e. Walmart.csv
4) Data transformation: Normalizing or aggregating data- We will do this during model building.
5) Data reduction: Reducing the amount of data through sampling or feature selection- We have train and test set.
6) Data encoding: Encoding categorical data as numerical data to be used in machine learning models.

Choosing the Algorithm for the project

I have chosen 2 algorithm 1) Linear Regression 2) Random Forest classifier

1) **Linear Regression:**

```
# Import sklearn

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

# Select features and target

X = without_outlier[['Store','Fuel_Price','CPI','Unemployment']]

y = without_outlier['Weekly_Sales']

# Split data to train and test (0.80:0.20)

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

# Linear Regression model

print('Linear Regression:')

print()

reg = LinearRegression()

reg.fit(X_train, y_train)

y_pred = reg.predict(X_test)

print('Accuracy:',reg.score(X_train, y_train)*100)

print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))

print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))

print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test,
y_pred)))
```

output- Linear Regression:

```
Accuracy: 12.71529491777188
Mean Absolute Error: 456168.6161742247
Mean Squared Error: 295612821026.9555
Root Mean Squared Error: 543702.8793623918
```

2) Random Forest Regressor:

```
# Random Forest Regressor
```

```
print('Random Forest Regressor:')
print()
rfr = RandomForestRegressor(n_estimators = 400,max_depth=15,n_jobs=5)
rfr.fit(X_train,y_train)
y_pred=rfr.predict(X_test)
print('Accuracy:',rfr.score(X_test, y_test)*100)
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
sns.scatterplot(y_pred, y_test);
```

Output:

```
Random Forest Regressor:

Accuracy: 92.91994873531797
Mean Absolute Error: 78415.75517606804
Mean Squared Error: 23439158248.54872
Root Mean Squared Error: 153098.52464523856
```

Sales forecasting for all store for next 12 weeks:

```
#Forecasting Sales Using ARIMA model
from statsmodels.tsa.arima.model import ARIMA
# convert the date column to a datetime type
walmart['Date'] = pd.to_datetime(walmart['Date'])
# group the data by store
store_groups = walmart.groupby('Store')
# create a dictionary to hold the forecasts
forecasts = {}
# loop through each store group and make a forecast
for name, group in store_groups:
  # set the date column as the index
    group = group.set_index('Date')
    # resample data to monthly frequency
    group = group.resample('M').sum()
    # Fit the ARIMA model
    model = ARIMA(group['Weekly_Sales'], order=(1,1,0))
    model_fit = model.fit()
    # Forecast for next 12 months
    forecast = model_fit.forecast(steps=12)[0]
    # add the forecast to the dictionary
    forecasts[name] = forecast
# View the forecasts for each store
for store, forecast in forecasts.items():
    print(f'Forecast for Store {store}:')
    print(forecast)
```

Output:-

```
Forecast for Store 1:
1590529.1618978747
Forecast for Store 2:
1908339.8789546136
Forecast for Store 3:
```

```
420698.3514374672
Forecast for Store 4:
2130494.288875375
Forecast for Store 5:
329254.8204031246
Forecast for Store 6:
1484297.5645178498
Forecast for Store 7:
485753.6759919293
Forecast for Store 8:
924066.563073879
Forecast for Store 9:
574062.9958583638
Forecast for Store 10:
1749809.2607451344
Forecast for Store 11:
1306488.464688095
Forecast for Store 12:
984654.5889333592
Forecast for Store 13:
2036122.811781726
Forecast for Store 14:
1798054.9947419153
Forecast for Store 15:
564546.505319537
Forecast for Store 16:
486481.66381001717
Forecast for Store 17:
931736.7941198689
Forecast for Store 18:
1068157.9968441057
Forecast for Store 19:
1395674.5037825727
Forecast for Store 20:
2165407.4587068902
Forecast for Store 21:
651665.7723245686
Forecast for Store 22:
1001263.6064295232
Forecast for Store 23:
1370514.5901378184
Forecast for Store 24:
1388917.8604630858
Forecast for Store 25:
716101.4206918592
Forecast for Store 26:
1052870.6190285995
Forecast for Store 27:
1666160.1154564698
Forecast for Store 28:
```

```
1233071.6231484716
Forecast for Store 29:
520783.8498678634
Forecast for Store 30:
435242.884959159
Forecast for Store 31:
1397432.9362338963
Forecast for Store 32:
1181378.053561292
Forecast for Store 33:
293626.8026182313
Forecast for Store 34:
949081.3388914403
Forecast for Store 35:
844917.008707711
Forecast for Store 36:
313290.3015886876
Forecast for Store 37:
524672.3712902293
Forecast for Store 38:
433844.60288276966
Forecast for Store 39:
1483884.9267091278
Forecast for Store 40:
975516.2879850105
Forecast for Store 41:
1384567.1787942827
Forecast for Store 42:
627195.2994189997
Forecast for Store 43:
628736.6518115152
Forecast for Store 44:
339383.9543482377
Forecast for Store 45:
750637.0784765178
```

Motivation and Reasons for choosing the Algorithm

The motivation for choosing a regression algorithm in this case could be because:

1) Linear regression is a simple and widely used method for predicting a continuous target variable based on one or more predictor variables.

2) Linear regression can be applied to a wide range of problems, such as forecasting sales, predicting stock prices, and estimating the relationship between different variables.

3) Linear regression is easy to interpret, as the coefficients of the independent variables can be used to estimate the effect of each variable on the target variable.

4) Linear regression is efficient and can be applied to large datasets.

5) Linear Regression assumes that the relationship between the independent variables and the dependent variable is linear, which is a reasonable assumption in this case as the sales are generally assumed to be affected by other variables in a linear fashion.

6) Given the dataset, the sales are continuous variable, this makes linear regression a good fit.

7) In this specific case, linear regression is a good choice because the goal is to predict a continuous target variable (weekly sales) based on several predictor variables such as store and item number, temperature, fuel price. Linear regression is well suited for this type of problem, and it's a good starting point for modelling the data.

Assumptions

1) Linear Relationship- The relationship between the independent and dependent variables is linear.

2) Independence: The observations in the dataset are independent of each other.

3) Homoscedasticity: The variance of the errors is constant across all levels of the independent variables.

4) Normality: The errors are normally distributed.

5) No multicollinearity: The independent variables are not highly correlated with each other.

6) No autocorrelation: The errors are not correlated with each other (i.e. no temporal correlation).

7) The data is resampled by weekly frequency and it's the assumption that the data is uniform across all weeks.

8) The dataset is large enough to make meaningful predictions.

9) The features are independent of each other and they are not correlated with the target variable.

10) The Linear Regression model will be able to capture the relationship between the independent variables and the dependent variable.

Model Evaluation and techniques

Model evaluation and techniques are important steps in the process of building and using a machine learning model. They help to ensure that the model is accurate, reliable, and generalizable to new data. Some common techniques and metrics used to evaluate linear regression models include:

1) Mean Squared Error (MSE): This metric measures the average difference between the predicted values and the actual values. A lower MSE indicates a better fit.

2) Root Mean Squared Error (RMSE): This metric is the square root of the MSE and is more interpretable in terms of the original units of the target variable.

3) R-squared (R2): This metric measures the proportion of variance in the target variable that is explained by the predictor variables. A higher R2 indicates a better fit.
4) Mean Absolute Error (MAE): This metric measures the average absolute difference between the predicted values and the actual values.

5) Residual plots: This technique plots the residuals (prediction errors) against the predicted values. This can reveal patterns in the errors that may indicate problems with the model such as non-linearity or heteroscedasticity.

6) Cross-validation: This technique involves splitting the data into multiple subsets, training the model on different subsets, and evaluating it on the remaining subset. This can give a better estimate of the model's performance on new data.

7) Regularization: Regularization is a technique used to prevent overfitting by adding a penalty term to the cost function. Two common types of regularization used in linear regression are L1 and L2 regularization.

8) Hyperparameter tuning: This technique involves adjusting the parameters of the model, such as the regularization strength, to optimize the model's performance.
9) Ensemble methods: This technique involves combining multiple models to improve the performance.

Inferences from the same

Inferences from the evaluation of a linear regression model can include:

1) The model's overall performance: The MSE, RMSE, and R2 scores can provide a sense of how well the model is fitting the data. A lower MSE, RMSE and a higher R2 indicate a better fit.

2) The importance of individual predictor variables: The coefficients of the independent variables can be used to estimate the effect of each variable on the target variable. This can provide insight into which variables are most important for predicting the target variable.

3) The presence of bias or variance issues: Residual plots can reveal patterns in the errors that may indicate problems with the model such as non-linearity or heteroscedasticity. This can suggest that the model is underfitting or overfitting the data.

4) The model's ability to generalize: Cross-validation can give a better estimate of the model's performance on new data, providing a sense of how well the model will perform when applied to unseen data.

5) The effect of regularization: Regularization can help prevent overfitting by adding a penalty term to the cost function. By comparing the performance of models with different regularization strengths, one can understand how regularization affects the model's performance.

6) The effect of hyperparameter tuning: By adjusting the parameters of the model and comparing the performance, one can understand how different hyperparameters affect the model's performance.

7) The benefit of ensemble methods: Ensemble methods can improve the performance by combining multiple models. The performance of the ensemble model can be compared with the performance of individual models to understand the benefit of ensemble methods.

Overall, inferences from the evaluation of a linear regression model can provide valuable insights into the model's performance, the importance of predictor variables, and the model's ability to generalize to new data. These insights can be used to improve the model and make more accurate predictions.

Future possibilities of the project

There are several future possibilities for a sales forecasting project using a Walmart dataset:

1) Incorporating more data: The model could be improved by incorporating additional data, such as economic indicators, demographic data, or data on specific product categories.

2) Improving feature engineering: The model could be improved by creating new features or transforming existing features to better capture the relationship between the predictor variables and the target variable.

3) Incorporating other models: The model could be improved by incorporating other types of models, such as Random Forest, XGBoost, LSTM or other time series models.

4) Incorporating external data: The model could be improved by incorporating external data such as weather data, population data, or other relevant datawhich could help the model make more accurate predictions.

5) Improving the model evaluation: The model could be improved by using more advanced evaluation techniques such as AUC-ROC, precision-recall, F1-score, or other evaluation metrics.

6) Automating the model: The model could be improved by automating the process of data cleaning, feature engineering, model training, and prediction.

7) Incorporating online learning: The model could be improved by incorporating online learning techniques, which allow the model to continuously update its predictions as new data becomes available.

8) Incorporating causal inference: The model could be improved by incorporating causal inference techniques, which can help identify the underlying causal relationships between the predictor variables and the target variable.

9) Incorporating unsupervised learning: The model could be improved by incorporating unsupervised learning techniques such as clustering, which can help identify patterns in the data that are not easily visible using supervised learning methods.

Overall, there are many possibilities for future work on this project, and by continuing to explore different techniques and incorporating new data, the model's performance can be improved and made more useful for forecasting sales at Walmart.

Conclusion:

In this project, one possible conclusion would be that the model was able to accurately predict weekly sales for the stores. Factors that may have contributed to the model's success could include the use of relevant features such as holidays, as well as the use of appropriate modelling techniques. Other factors such as the quality of the data and the size of the dataset may also have played a role in the model's performance.

There is still a room for improvement in the model's performance, and recommendations for future work such as trying different modelling techniques or incorporating additional data sources could be provided.

References

- Intellipaat live session recording
- Intro to Linear Regression, Doughlas Montgomery, 1982
- Wikipedia
- Google.com
- Geeksforgeeks.com