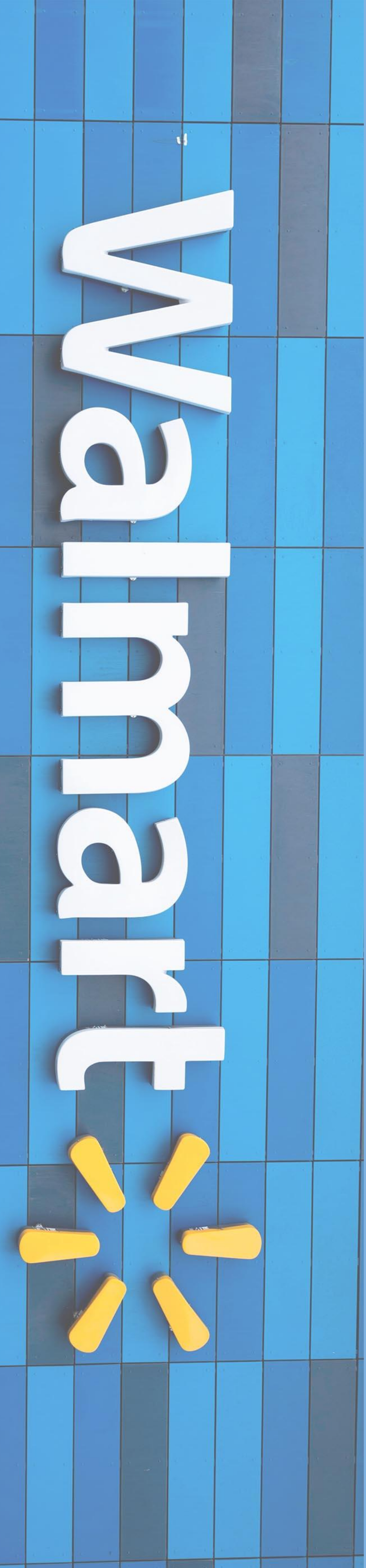


CAPSTONE PROJECT

Topic

Prediction of Weekly Sales for Walmart Stores across the US

Mentored by:

Animesh Tiwari

Submitted by:

Bala Murali Krishna Vaka

Vaishnavi Ravi

Shubhash AV T

Tanmay Agarwal

Saurabh Walde

Table of Contents

Page No.

1.0 Industry Review

1.1 Current practices:

- Walmart with more than 265 million customers per week, 524.14 billion of revenue (by 2020 ESG report) and 11,500 stores across the world, It is the world's largest retailer.
- Walmart has stores under 56 banners in 27 countries and E-Commerce websites employs over 2.2 million associates worldwide.
- EDLC (Every Day Low Cost) is the Walmart commitment to control expenses and savings can be passed along to customers.
- It has Omni-channel presence to provide customers access to broad assortments of goods at any in many locations in US and internationally.
- Alphabot is the automated technology at Walmart that picks and packs online grocery orders at high speed. Alphabot operates in a 20,000 square foot facility that Walmart built onto one of its stores in Salem
- Alphabot helps in making the work 10 times faster than human and takes the head on competition with giant like Amazon in grocery business.
- Walmart + is the new introduction from Walmart that is the membership program that combines instore and online benefits with Unlimited free delivery, Scan and go services (a fast way of sale - quick , easy and touch free service) , Fuel discounts.

Key Enablers for Walmart:

1. Associates
2. Low-cost structure
3. Technology, data and analytics
4. Sustainable sourcing and operations
5. Supply chain design and innovation
6. Ecosystem thinking and partnerships

Every day Low cost structure:

- By procuring material at bulk volumes at low price.
- Though the margins are slimmer are slimmer, volume of sales make up to the profit markup.
- Bargaining power that would enable Walmart to remake the supply sector and the retail landscape, to suit its own schemes.
- Minimization of operating costs.
- Supply chain management – key role.

Analytics used in the Walmart:

- To make Walmart purchases more efficient.
- To improve store checkout.
- To manage supply chain.
- To optimize product assortment.
- To personalize the shopping experience.

1.2 Background Research:

- Retail analytics tools provide the insight you need to understand customer buying behavior, so you can forecast and plan for future demand. This data can then inform how to price and promote products or services to generate more revenue, and how to optimize the supply chain
- The global retail analytics market size to grow from USD 4.3 billion in 2020 to USD 11.1 billion by 2025, at a Compound Annual Growth Rate (CAGR) of 21.2% during the forecast period.
- In order to stay ahead of the game in today's age of e-commerce, retail merchants need to learn handle the incoming data and get it ready for analytics.

Predictive analytics can be called the proactive part of data analytics.

Predictive analytics on main areas in retail:

1. Personalization for customers
2. Inventory and supply chain management
3. Customer segmentation and customer journey
4. Customer behavior analytics
5. Campaign management

Marketing analytics used by some of the companies as below:

- Spotify's brand strategy: behavioral analytics, ads that came from data of customers streaming habits targeting music list for people coming different demographics. That has worked for the company and got millions of loyal customers.
- Easy Jet using marketing analytics for the campaign (personalized emails): with customers data from their first travel with airlines they have created campaign targeting where might the customer's next travel and released offers on flight tickets.
- Under Armour uses marketing analytics to come up with new products:
- Netflix uses marketing analytics to keep the content engaging for the customers with recommendation based on pattern of content that viewed and percentage of likeliness towards the content personalized.
- Sephira used predictive analysis to improve digital experience, product discovery tools for the customer loyalty rewards and suggestions based on recently viewed SKU's and Past purchases

Marketing analytics leverage advanced data analytics and social listening technologies to develop more comprehensive customer profiles. This is done by analyzing transaction records and researching customer's social media behaviors to uncover their shopping preferences and future needs.

Benefits of marketing analytics in Retail:

1. Increase customers shopping frequency and loyalty
2. Identify target customers and increase sales revenue by 15%
3. Improve marketing operation efficiency
4. Optimize marketing resources, increase ROI
5. Obtain customer profile.

1.3 Literature Survey - Publications, Application, past and undergoing research

- Marketing analytics involves collection, management, and analysis—descriptive, diagnostic, predictive, and prescriptive— of data to obtain insights into marketing performance, maximize the effectiveness of instruments of marketing control, and optimize firms' return on investment (ROI).

- Many of the methods developed by marketing academics since the 1960s have now found their way into practice and support decision making in areas such as CRM, marketing mix, and personalization and have increased the financial performance of the firms deploying them.

- The two pillars of the successful development and implementation of marketing analytics in firms:

  (1) The adoption of organizational structures and cultures that foster data-driven decision making.

  (2) The education and training of analytics professionals.

- New forms of marketing have emerged, including recommendations, geo-fencing, search marketing, and retargeting. Marketing analytics has come to play a central role in these developments, and there is urgent demand for new, more powerful metrics and analytical methods.

Past and undergoing research

- Retailers utilize the latest technology for obtaining consumer data and marketing analytics including IP addresses, geo-fencing data, beacons, behavioral data from the Internet of Things, automated facial recognition data, and radio frequency identification (RFID) tags

- CSAR the **C**ross-**S**ectional **A**uto **R**egression model which combines the qualities of cross-sectional forecasting with the adaptability of ARIMA (Auto Regressive Integrated Moving Average). Therefore, CSAR meets the aforementioned requirements and is adaptable to a wide range of application domains

- The application of traditional forecast techniques like ARIMA or Exponential Smoothing, although, they are successfully applied in a wide variety of application scenarios

- Return on Marketing Investment (ROMI) is an OPEX model that compares non-CAPEX spend to yield.

- New approach - *Diffusion-based Iterative Characterizing region Exploration* (*DICE*), that combines store sales and geographic proximity to find a set of coherent regions for a given product.

Solutions to big data analytics in the future will use the following:

- Developments in high-performance computing, including MapReduce frameworks for parallel processing, grid and cloud computing, and computing on graphic cards.

- Simpler descriptive modelling approaches, such as probability models, or computer science and machine learning approaches that facilitate closed-form computations, possibly in combination with model averaging and other divide-and-conquer strategies to reduce bias.

- Speed improvements in algorithms provided by variational inference, scalable rejection sampling, resampling and reweighting, sequential MCMC, and parallelization of likelihood and MCMC algorithms; and Application of aggregation, data fusion, selection, and sampling methods that reduce the dimensionality of data.

2.0 Project Statement

Conventional retail stores still play a prominent role in a world dominated by Ecommerce. As time has passed, retailers have had to evolve in order to keep up with changes in demands and the ever-changing mindset of customers. One such retail industry juggernaut that has kept up with the demands of customers as well changed the face of the retail industry for the better is Walmart Inc. We are trying to build a model to improve the sales of the organization as well as better service for the customers.

Dataset and Domain

This dataset was downloaded from the Kaggle platform which has three years weekly sales data and under the domain of Marketing Analysis.

Data Dictionary and Variable Categorization

The data sources for the project have been downloaded from Kaggle and was used in Competitions hosted on Kaggle.

The datasets consist of historic data of weekly sales for different stores of Walmart. There are 400000+ records classified into different stores, departments, size and has markdowns (promotions).

Below table lists the number of features that is available for this dataset. We can classify the features based on the datatype or their behaviour. We can broadly classify the features as either numeric or categorical.

Quantitative variables are any variables where the data represent amounts (e.g., weekly sales, weight, or age). These are broadly called as numerical variables or features.

Categorical variables are any variables where the data represent groups. This includes rankings (e.g., finishing places in a race), classifications (e.g., brands of cereal), and binary outcomes (e.g., coin flips).

2.1 Data Dictionary and Pre-processing Data Analysis

S.No	Variables Names	Categorization of Variable	Null values Check
1.	Store	Numeric	421570 non_null int 64
2.	Dept	Numeric	421570 non_null int 64
3.	Date	Discrete(Numeric)	421570 non_null datetime64
4.	Weekly_Sales	Numeric	421570 non_null float64
5.	IsHoliday	Binary(Categorical)	421570 non_null bool
6.	Type	Categorical	421570 non_null object
7.	Size	Numeric	421570 non_null int 64
8.	Temperature	Continuous (Numeric)	421570 non_null float
9.	Fuel_Price	Continuous (Numeric)	421570 non_null float

10.	MarkDown1	Numeric	150681 non_null float
11.	MarkDown2	Numeric	111248 non_null float
12.	MarkDown3	Numeric	137091 non_null float
13.	MarkDown4	Numeric	134967 non_null float
14.	MarkDown5	Numeric	151432 non_null float
15.	CPI	Continuous (Numeric)	421570 non_null float
16.	Unemployment	Continuous (Numeric)	421570 non_null float

Attribute information

Feature	Type	Description
Store	Numeric	Store details with Unique identifier -Number was given to each 45 stores.
Dept	Numeric	Department detail with Unique identifier was given to each 81 departments.
Date	Discrete(Numeric)	Given every Friday date of the week .Date is between 05/02/2010 and 22/10/2010. The format of the date is in YYYY-MM-DD. There is a weak difference in each record.
Weekly_Sales	Numeric	Sales per week at a given store.
IsHoliday	Binary(Categorical)	Given every Friday date of the week. Holidays in a week are given in True or False, This is the data where a weekday is a holiday or not in the week. If weekday is a holiday it is True or it is False.
Type	Categorical	A, B, C are the type of Walmart stores given. Considering the description we are taking these stores as Hypermarkets, Discount department Stores, Grocery Stores.
Size	Numeric	Size of the store is given.
Temperature	Continuous (Numeric)	Average temperature in the region (in °F) where the store is located.
Fuel_Price	Continuous (Numeric)	Cost of fuel in the region.
MarkDown1	Numeric	Promotion activities to boost sales on a special Holiday, Markdown are the discounts given in a particular store in a given date
MarkDown2	Numeric	Promotion activities to boost sales on a special Holiday, Markdown are the discounts given in a particular store in a given date.
MarkDown3	Numeric	Promotion activities to boost sales on a special Holiday, Markdown are the discounts given in a particular store in a given date.

MarkDown4	Numeric	Promotion activities to boost sales on a special Holiday, Markdown are the discounts given in a particular store in a given date.
MarkDown5	Numeric	Promotion activities to boost sales on a special Holiday, Markdown are the discounts given in a particular store in a given date.
CPI	Continuous (Numeric)	The Consumer Price Index (CPI) measures the change in prices paid by consumers for goods and services. It indicates the changes in the value at present to the base value.
Unemployment	Continuous (Numeric)	Unemployment rate of the customers that visit the store.

2.2 Project Justification

- This is a sales data set from Walmart for recruitment of Data Scientist for their organization.
- The Data set is about Walmart analytics on Weekly sales.
- This is a regression problem. The dependent variable is **Weekly_sales.**
- We can use Regression model algorithms like Linear Regression, Decision Tree, Random Forest, etc.,
- We can use bagging and boosting techniques for increasing the accuracy and performance of the model.
- Explore important questions such as ‘How the weekly sales is affected by the independent features’.

2.3 Data Understanding

There are 4 data sets namely Features, Stores, Train, Test.

Shape of the data sets (Rows, Columns)

```
Features
(8190, 12)

stores
(45, 3)

train
(421570, 5)

test
(115064, 4)
```

Data Preparation and Feature Engineering

Merging the data sets to further perform Exploratory Data Analysis, Data pre-processing and Modelling:

- Merging features and store datasets

  ```
  df1 = pd.merge(features,stores,on='Store',how='inner')
  ```

- Merging df1 to train and test data sets simultaneously

  ```
  df_train = pd.merge(df1,train,on=['Date','Store','IsHoliday'],how='inner')
  df_test = pd.merge(df1,test,on=['Date','Store','IsHoliday'],how='inner')
  ```

- Labeling train and test data in a new column (train/test)

  Creating both columns to identify the test and train data

  ```
  df_train['train/test'] = 'train'
  df_test['train/test'] = 'test'
  ```

- Adding weekly sales column to test data set

  Since, we do not have weekly_sales data in test data set, so we are imputing the column with null values.

  ```
  df_test['Weekly_Sales'] = np.nan
  ```

- Concatenating train and test data set

  Concatenating the data sets to perform data pre-processing on the whole data.

  ```
  data = pd.concat([df_train,df_test],axis=0,ignore_index=True)
  ```

Understanding the Final Dataset

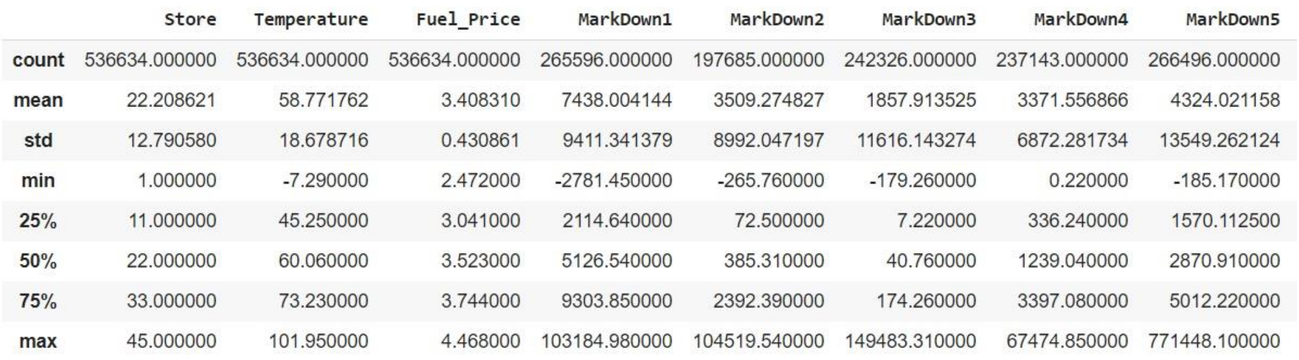
Shape of final dataset: (536634, 17)

Info:`RangeIndex: 536634 entries, 0 to 536633`

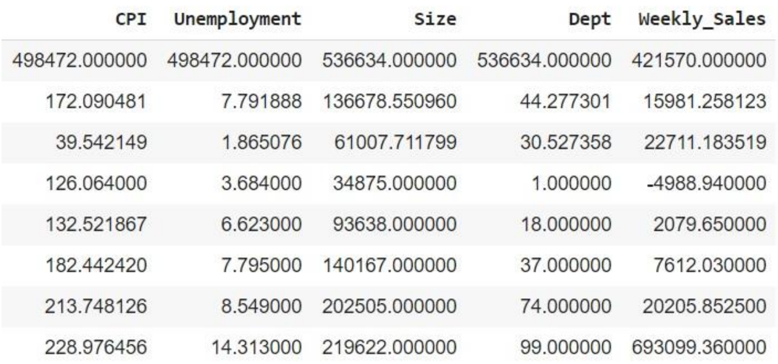
```
Data columns (total 17 columns):
 #   Column        Non-Null Count   Dtype  
---  ------        --------------   -----  
 0   Store         536634 non-null  int64  
 1   Date          536634 non-null  object 
 2   Temperature   536634 non-null  float64
 3   Fuel_Price    536634 non-null  float64
 4   MarkDown1     265596 non-null  float64
 5   MarkDown2     197685 non-null  float64
 6   MarkDown3     242326 non-null  float64
 7   MarkDown4     237143 non-null  float64
 8   MarkDown5     266496 non-null  float64
 9   CPI           498472 non-null  float64
 10  Unemployment  498472 non-null  float64
 11  IsHoliday     536634 non-null  bool   
 12  Type          536634 non-null  object 
 13  Size          536634 non-null  int64  
 14  Dept          536634 non-null  int64  
 15  Weekly_Sales  421570 non-null  float64
 16  train/test    536634 non-null  object 
dtypes: bool(1), float64(10), int64(3), object(3)
```

Five Point Summary

Five-number summary is used to describe the distribution of data without assuming a specific data distribution. For example, the mean and standard deviation are used to summarize a gaussian distribution (normal distribution). The five-number summary can describe a data sample with any distribution.



	Store	Temperature	Fuel_Price	MarkDown1	MarkDown2	MarkDown3	MarkDown4	MarkDown5
count	536634.000000	536634.000000	536634.000000	265596.000000	197685.000000	242326.000000	237143.000000	266496.000000
mean	22.208621	58.771762	3.408310	7438.004144	3509.274827	1857.913525	3371.556866	4324.021158
std	12.790580	18.678716	0.430861	9411.341379	8992.047197	11616.143274	6872.281734	13549.262124
min	1.000000	-7.290000	2.472000	-2781.450000	-265.760000	-179.260000	0.220000	-185.170000
25%	11.000000	45.250000	3.041000	2114.640000	72.500000	7.220000	336.240000	1570.112500
50%	22.000000	60.060000	3.523000	5126.540000	385.310000	40.760000	1239.040000	2870.910000
75%	33.000000	73.230000	3.744000	9303.850000	2392.390000	174.260000	3397.080000	5012.220000
max	45.000000	101.950000	4.468000	103184.980000	104519.540000	149483.310000	67474.850000	771448.100000



CPI	Unemployment	Size	Dept	Weekly_Sales
498472.000000	498472.000000	536634.000000	536634.000000	421570.000000
172.090481	7.791888	136678.550960	44.277301	15981.258123
39.542149	1.865076	61007.711799	30.527358	22711.183519
126.064000	3.684000	34875.000000	1.000000	-4988.940000
132.521867	6.623000	93638.000000	18.000000	2079.650000
182.442420	7.795000	140167.000000	37.000000	7612.030000
213.748126	8.549000	202505.000000	74.000000	20205.852500
228.976456	14.313000	219622.000000	99.000000	693099.360000

The weekly sales column has the min value of -4988.94, we have to treat the values below zero because sales cannot be zero and below, so we removing the rows that containing negative value and zero.

- Checking the number of rows having weekly sales 0 and below
```
data[data['Weekly_Sales']<=0].shape
```

  There are 1358 rows that have weekly sales value below or equal to zero

- Removing the rows having weekly sales 0 and below
```
data = data.drop(data[(data['Weekly_Sales']<=0)&(data['train/test']=='train')].index)
```

We are dropping the 1358 rows from our final dataset as it is not useful for our analysis.

2.4 Null Values

Data contain null values for many reasons such as observing the data is not recorded, data corruption. So when your data containing the null value that means we don't get the right analysis on our data and many of machine learning algorithm doesn't support these missing values. That is the reason behind handling the missing values.

There are two important process to handling this missing value

1. Dropping (or)
2. Imputation of null value

<u>Null values and its percentage that are in our dataset</u>

```
                  Null Values                           % of Null Values

     Store                 0             Store               0.000000
     Date                  0             Date                0.000000
     Temperature           0             Temperature         0.000000
     Fuel_Price            0             Fuel_Price          0.000000
     MarkDown1        270180             MarkDown1          50.474895
     MarkDown2        337935             MarkDown2          63.132851
     MarkDown3        293390             MarkDown3          54.810976
     MarkDown4        298582             MarkDown4          55.780943
     MarkDown5        269283             MarkDown5          50.307318
     CPI               38162             CPI                 7.129406
     Unemployment      38162             Unemployment        7.129406
     IsHoliday             0             IsHoliday           0.000000
     Type                  0             Type                0.000000
     Size                  0             Size                0.000000
     Dept                  0             Dept                0.000000
     Weekly_Sales     115064             Weekly_Sales       21.496200
     train/test            0             train/test          0.000000
```

Treating the null values

1. Since, Markdown columns are explaining the marketing campaign details and the percentage of null values are above 50%, so we will impute the null values with 0.

2. Treating the null values in cpi and unemployment columns by imputing with 'forward fill' method because it is a huge data, and we are having only 7% missing values in these two columns.

3. The missing values in weekly sales column are from test data, so it’s not missing values, so we leave it as it is.

2.5 Feature Engineering

1. Handling ‘Markdown’ columns

- As we have imputed zero for all missing values in all five markdown columns, we have to perform feature engineering for the promotional markdowns to identify the significance.

- In order to identify and analyze whether the promoting markdown campaigns have any significant impact on sales or not, we will create a new feature markdown where 0 signifies promotional campaign not done and 1 if it is done.

- Since, we know that we imputed the markdown1, markdown2, markdown3, markdown4, markdown5 columns with 0's initially. And now we have created a separate markdown column with 0 and 1 to tell if there was any markdown (promotional) activity was done by the particular store or not. We will remove the promotional markdown columns as more than 50% of the values were missing.

2. **‘Dept_Type’ Column:**

   Since, there are 81 departments for each store we will group the departments on the basis of the frequency of purchases from that departments and make it as a new feature called ‘Dept_Type’ then we can drop the column ‘Dept’. The frequency can be obtained by percentiles of the weekly sales for each departments.

   We are grouping the departments into five types namely:
   Rare, Less Frequent, Moderately Frequent, Very Frequently, Most Frequently.

3. **‘Week’ and ‘Year’ Column:**

   Converting Date into Date-time format

   ```
   data_n['Date'] = pd.to_datetime(data_n['Date'],format='%Y-%m-%d',)
   ```

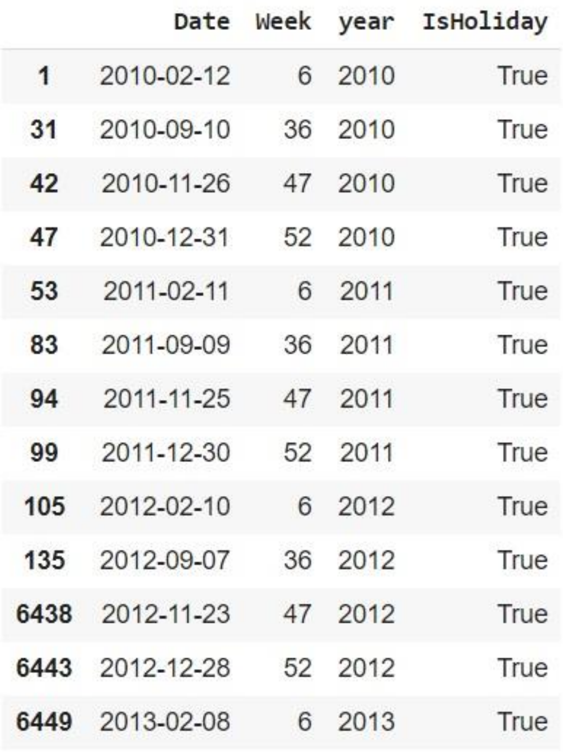
   Extracting week and year from the dates as we will not use dates in the model building

   ```
   from datetime import date as dt
   data_n['Week'] = data_n['Date'].dt.week
   data_n['year'] = data_n['Date'].dt.year
   ```

   We have created a two new column ‘week’ and ‘year’ from ‘Date’.

Checking the Holiday weeks y-o-y

```
data_n[data_n['IsHoliday']==True][['Date','Week','year','IsHoliday']].drop_
duplicates()
```



	Date	Week	year	IsHoliday
1	2010-02-12	6	2010	True
31	2010-09-10	36	2010	True
42	2010-11-26	47	2010	True
47	2010-12-31	52	2010	True
53	2011-02-11	6	2011	True
83	2011-09-09	36	2011	True
94	2011-11-25	47	2011	True
99	2011-12-30	52	2011	True
105	2012-02-10	6	2012	True
135	2012-09-07	36	2012	True
6438	2012-11-23	47	2012	True
6443	2012-12-28	52	2012	True
6449	2013-02-08	6	2013	True

Inference:

1. We can see that Holidays are occurring in the same weeks for all the years.

3.0 EXPLORATORY DATA ANALYSIS (EDA)

Exploratory Data Analysis (EDA) is the process of investigating the datasets in order to understand the important characteristics, underlying patterns, relationship between the features, presence of anomalies and gathering insights from the data.

EDA often combines graphical as well as non-graphical methods. Non graphical methods include summary statistics of the data, whereas various visualization techniques are used in graphical methods. When analysis is performed on a single feature, which helps in understanding the range, distribution, mean, mode, anomalies etc. of that feature alone, it is called as univariate analysis, while multivariate (or bivariate) analysis includes examining and representing the relationships between two or more variables.

In this section, a univariate analysis of all the independent variables (features) will be carried out. Furthermore, the impact of these features on the target variable will also be analysed. Let us visualize and explain the analysis.

Relationship between variables:

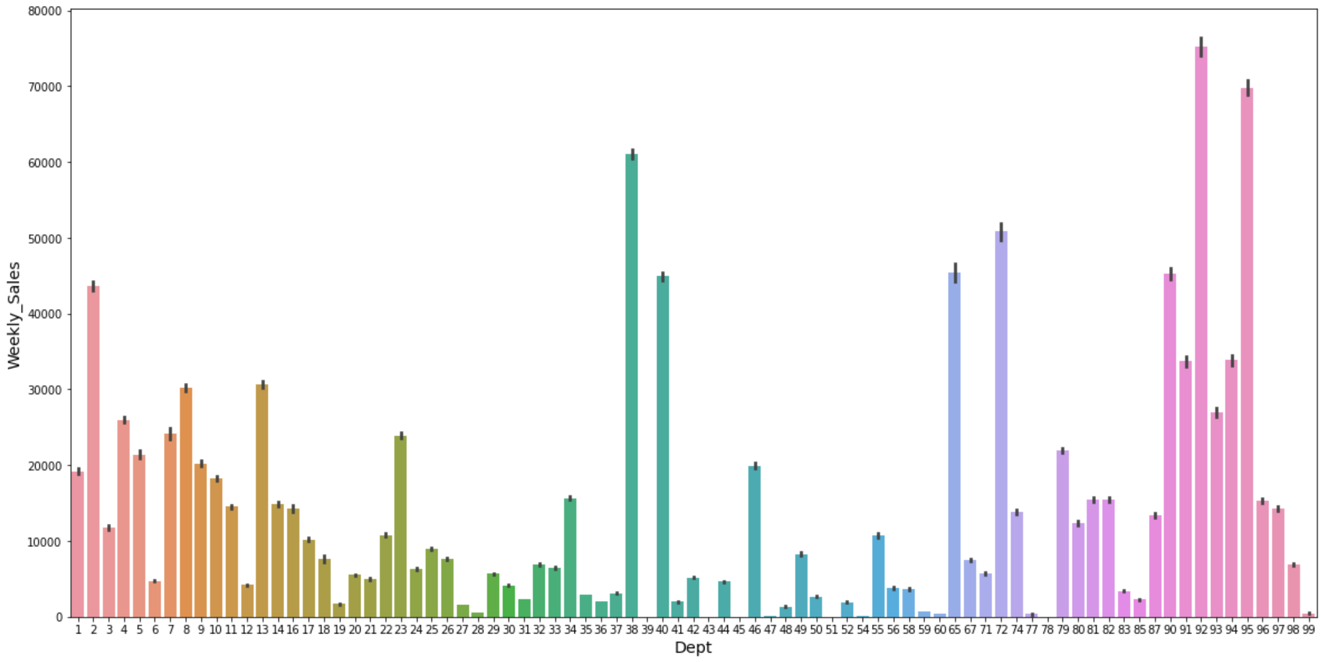
Analysis of Categorical Features in the dataset

- There are a total of 10 features. Out of which Weekly Sales is the target variable. The various visual analytics as follows:
- There are three categorical variables present in the dataset – 'IsHoliday', 'Type', 'Dept_Type'.
- The scope of EDA on categorical variables includes the important features for getting Business Insights and understanding the weekly sales.
- We will visualize some of the important business questions through EDA.

Dept vs Weekly-Sales:

Dept is a categorical feature. There are about 81 departments.

Average Weekly_Sales sales for each Dept:



Inference based on the above graph:

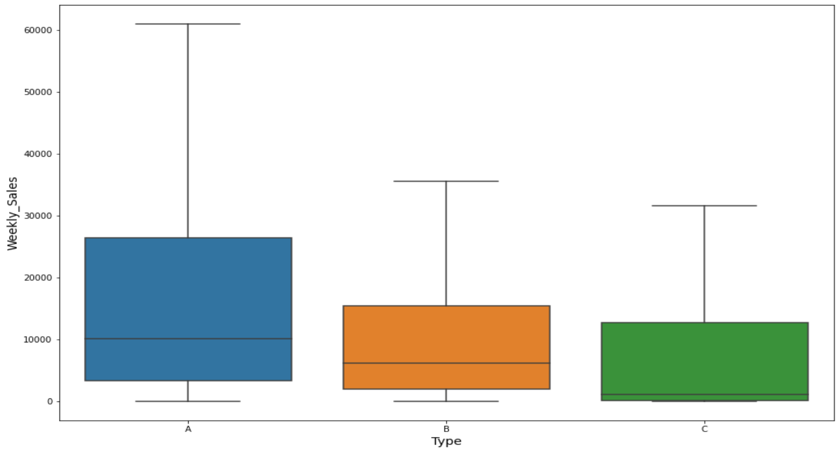
From the above visualizations, we can see that the average weekly_sales of each Dept varies.

- Departments 19,27,28,39,43,45,47,51,54,59,60,77,78,99 have either very low average weekly sales or no Weekly sales at all
- Departments 38,40, from 90 to 95 have high average weekly sales
-

Type vs Weekly-Sales:

Type is a categorical feature. There are about 3 type of stores.

The relation of Type with respect to average Weekly_Sales:



Inference based on the above graph:

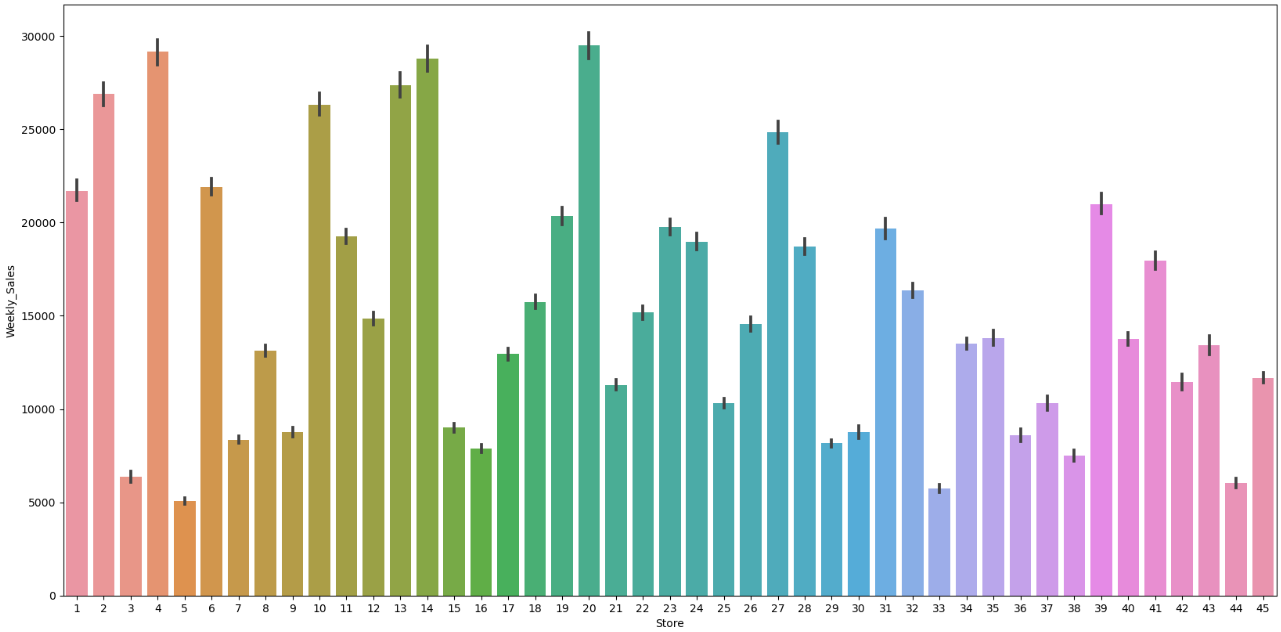
From the above visualizations, we can conclude that the average weekly_sales:

- Type A has the highest average weekly sales.
- Type B has less sales when compared to Type A
- Type C has the least average weekly sales.

Store vs Weekly-Sales:

Store is a categorical feature. There are about 45 stores.

The distribution of Type with respect to Average Weekly Sales:



Inference based on the above graph:

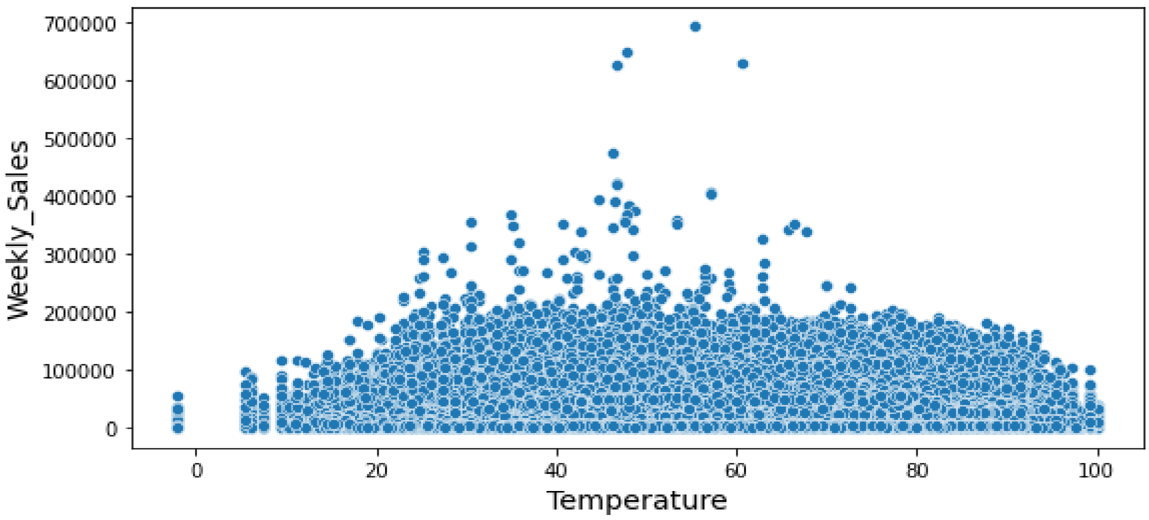
From the above visualizations, we can conclude that the average weekly_sales:

- We can observe significant difference in average weekly sales among various stores of Walmart across the country.

Temperature vs Weekly_Sales:

Temperature is a continuous feature.

The relationship of Temperature with respect to average Weekly_Sales:



Inference based on the above graph:

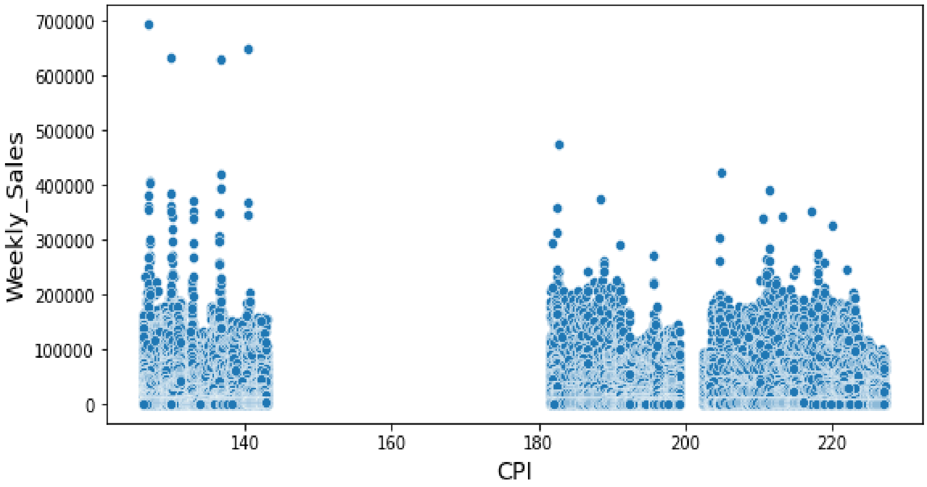
From the above visualizations, we can conclude that the average weekly sales:

- Average remains constant when the temperature between 20 to 100 F
- But we can observe extremely high average weekly sales when the temperature is between 30 to 70 F

CPI vs Weekly_Sales:

CPI is a continuous feature.

The relationship of CPI with respect to Average Weekly_Sales:



Inference based on the above graph:

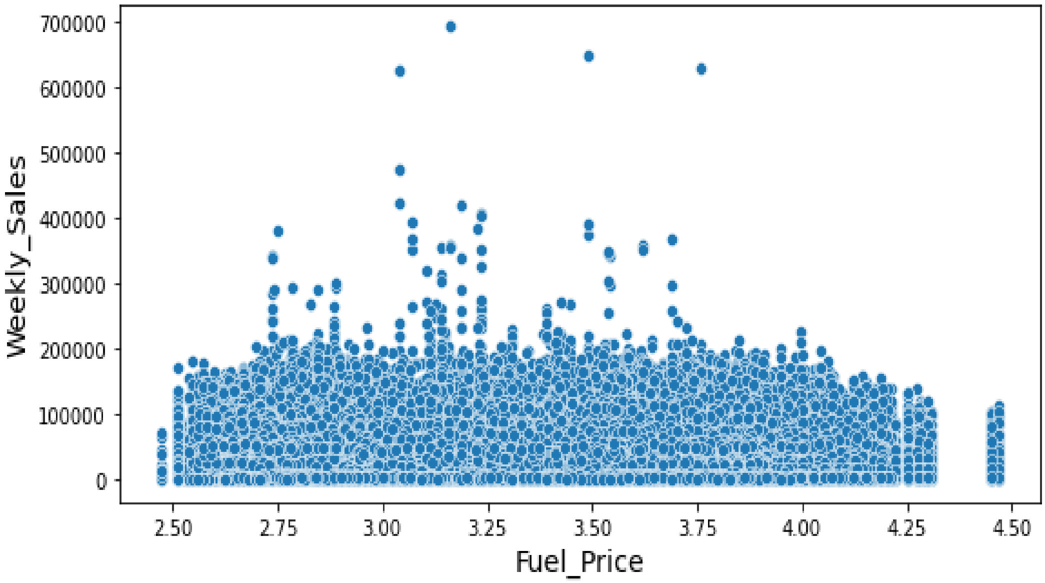
From the above visualizations, we can conclude that the average weekly_sales:

- When the CPI is low (below 140) we can observe some very high weekly sales.
- When the CPI is high (above 180) we can observe no significant difference in weekly sales.

Fuel_Price vs Weekly_Sales:

Fuel Price is a continuous feature.

The relation of Fuel_Price with respect to Average Weekly_Sales:



Inference based on the above graph:

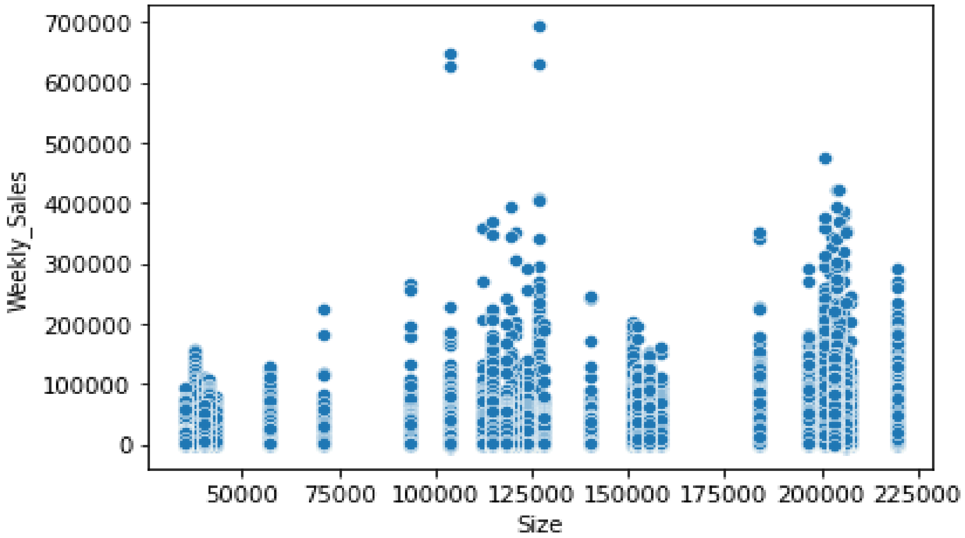
From the above visualizations, we can conclude that the average weekly_sales:

- There is no significant difference in the weekly sales based on fuel price.
- We can observe some extreme weekly sales when fuel price lies between 2.5$ to 3.75$

Size vs Weekly-Sales:

Size is a continuous feature.

The relationship of Type with respect to average Weekly_Sales:



Inference based on the above graph:

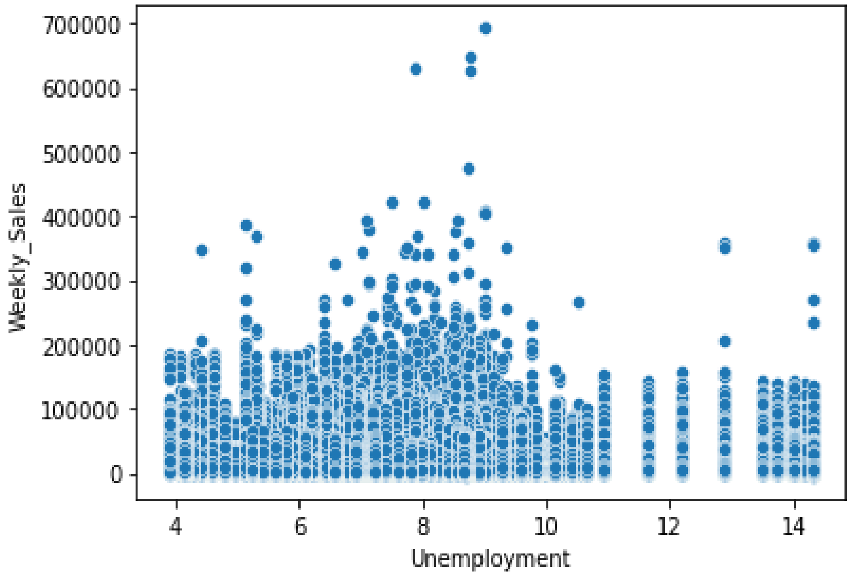
From the above visualizations, we can conclude that the average weekly_sales:

- Varies based on the size of the store.
- The store with size 125000 and 200000 has high weekly sales
- From store size above 100000 we can observe moderate to high weekly sales.

Unemployment vs Weekly-Sales:

Unemployment is a continuous feature.

The realtionship of Unemployment with respect to average Weekly_Sales:



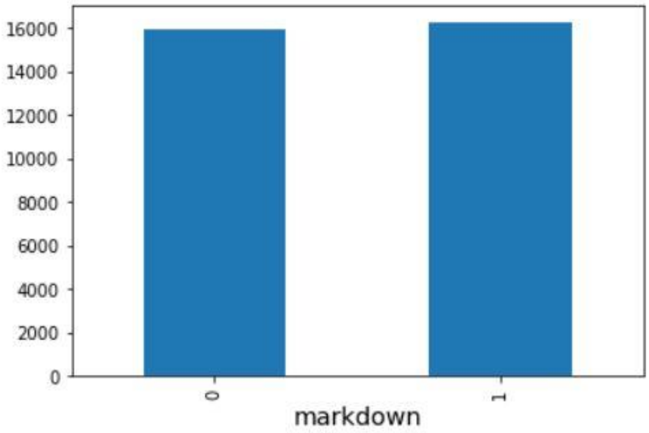
Inference based on the above graph:

From the above visualizations, we can conclude that the average weekly_sales:

- Is high when the unemployment rate is 6% to 8%.
- From unemployment 10 to 14 there is gaps in weekly sales or moderate weekly sales happened.
- There is no significant difference in weekly sales based on the unemployment rate of a particular region.
- We can observe some outliers when the unemployment rate is between 6% to 10%.

Markdown vs Weekly-Sales:

Markdown is a categorical feature.



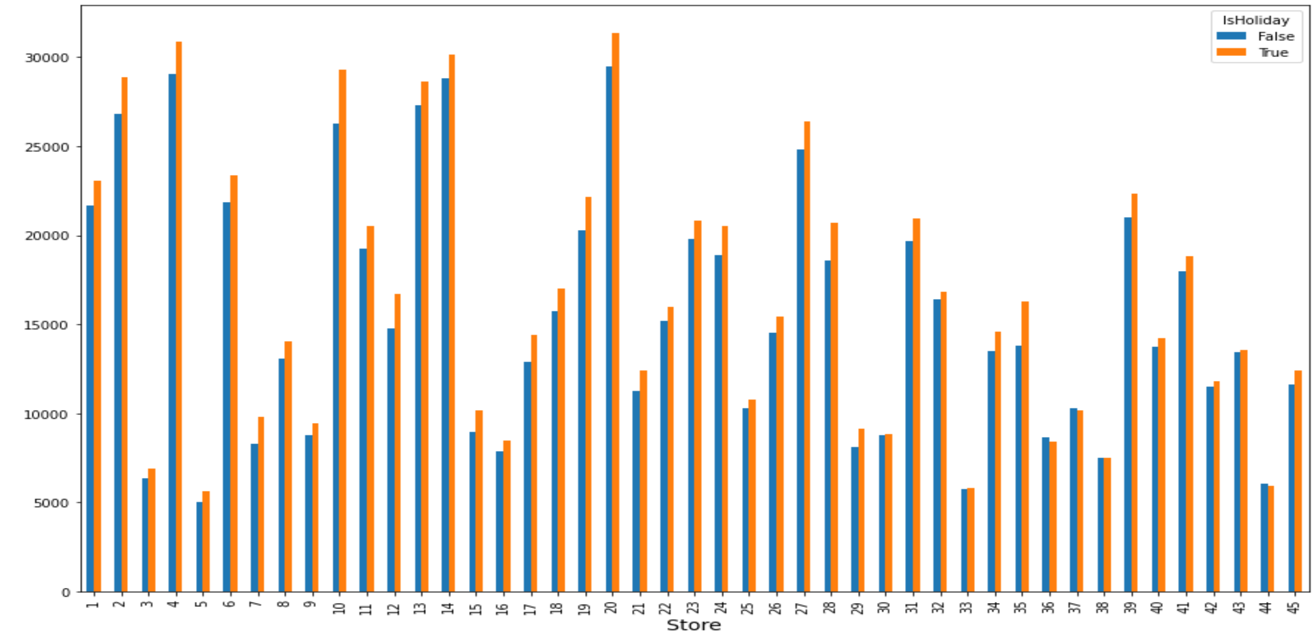
Inference based on the above graph:

From the above visualizations, we can conclude that the average Weekly_Sales:

IsHoliday vs Weekly-Sales:

Is Holiday is a categorical feature.

The relationship of Is holiday and Store with respect to average Weekly_Sales:

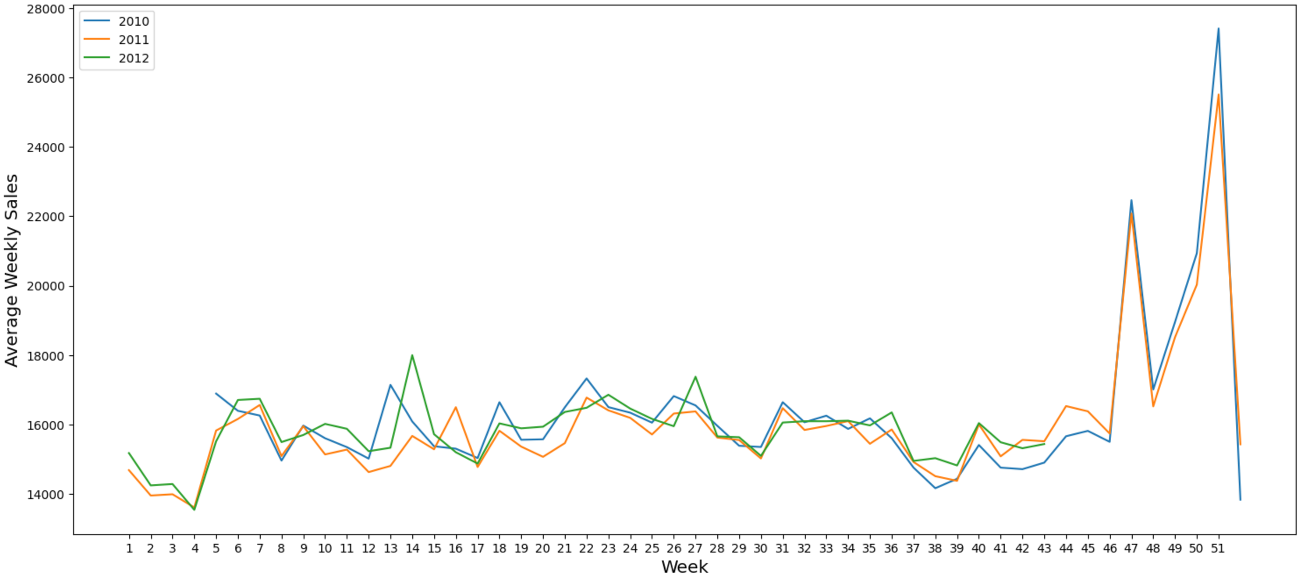


Inference based on the above graph:

From the above visualizations, we can conclude that the average Weekly_Sales:

- Average weekly sales are high during Holiday.

Seasonality plot



Inference based on the above graph:

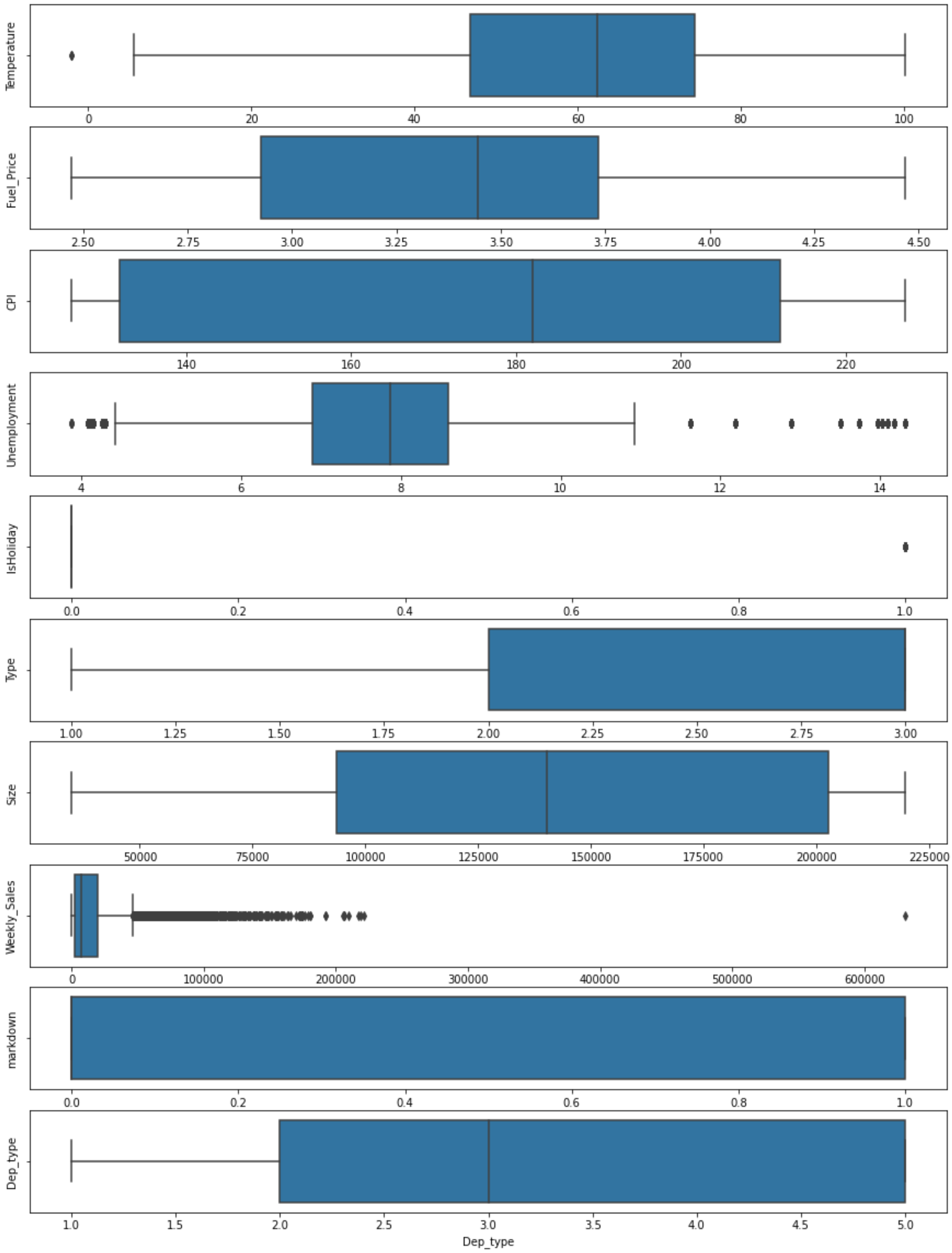
- We can observe a seasonality in the average week sales of different stores of Walmart from the year 2010-2013.
- Average Weekly sales tend to rise during some particular weeks, at the end of each year.

OUTLIERS:

An outlier is an observation that lies an abnormal distance from other values in a random sample from a population. In a sense, this definition leaves it up to the analyst to decide what will be considered abnormal. Before abnormal observations can be singled out, it is necessary to characterize normal observations.

Outliers present is the data and its treatment:

Presence of Outliers



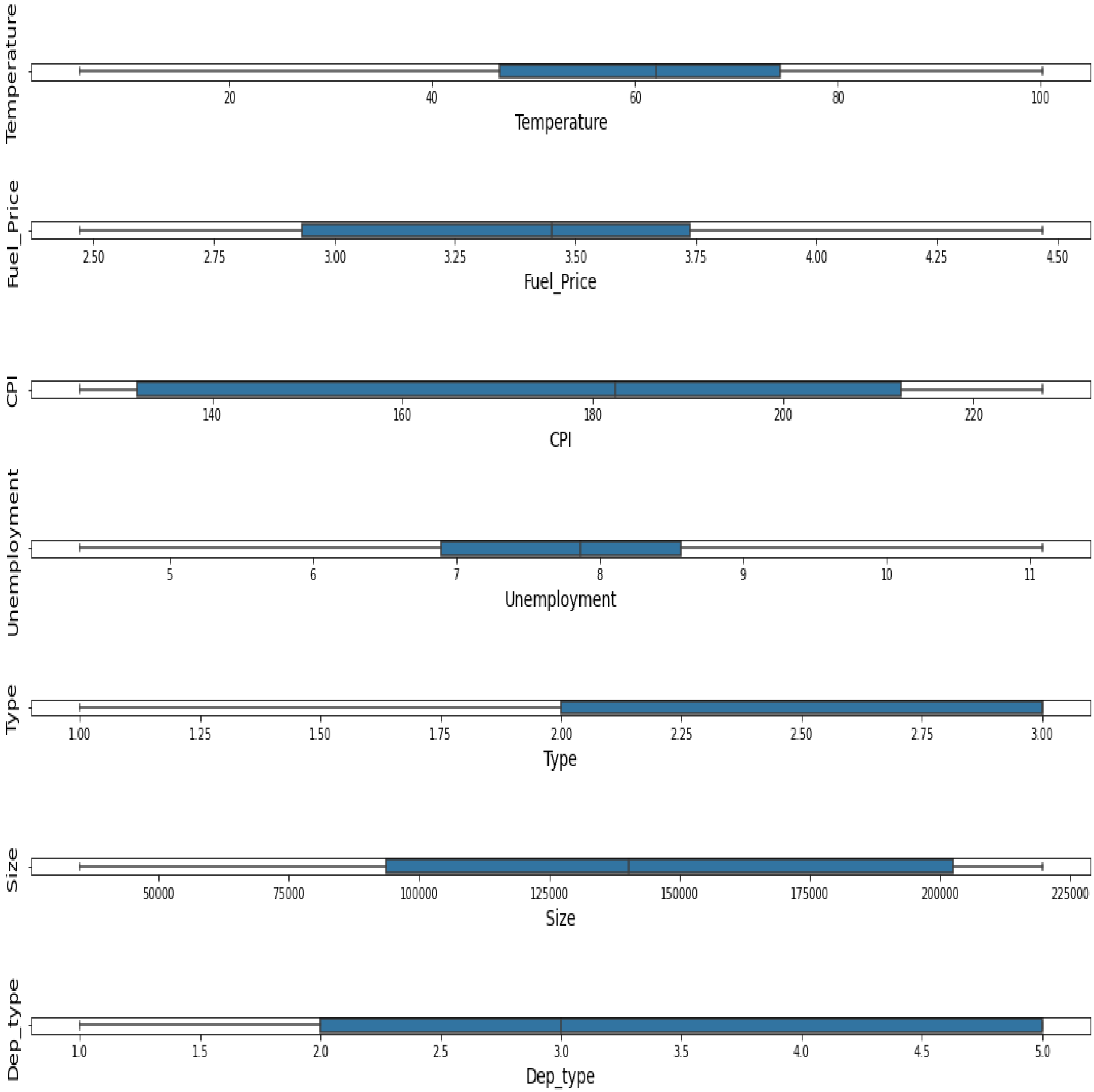
Transformations:

Log Transform has been applied to reduce the outliers in the continuous features. Since our outliers in weekly_sales are important, we are not treating it, but we are creating two models one with outliers and without outliers.

The features that have Log Transform performed on them are:

- Temperature
- Fuel Price
- Unemployment
- Size
- CPI
- Dept_Type

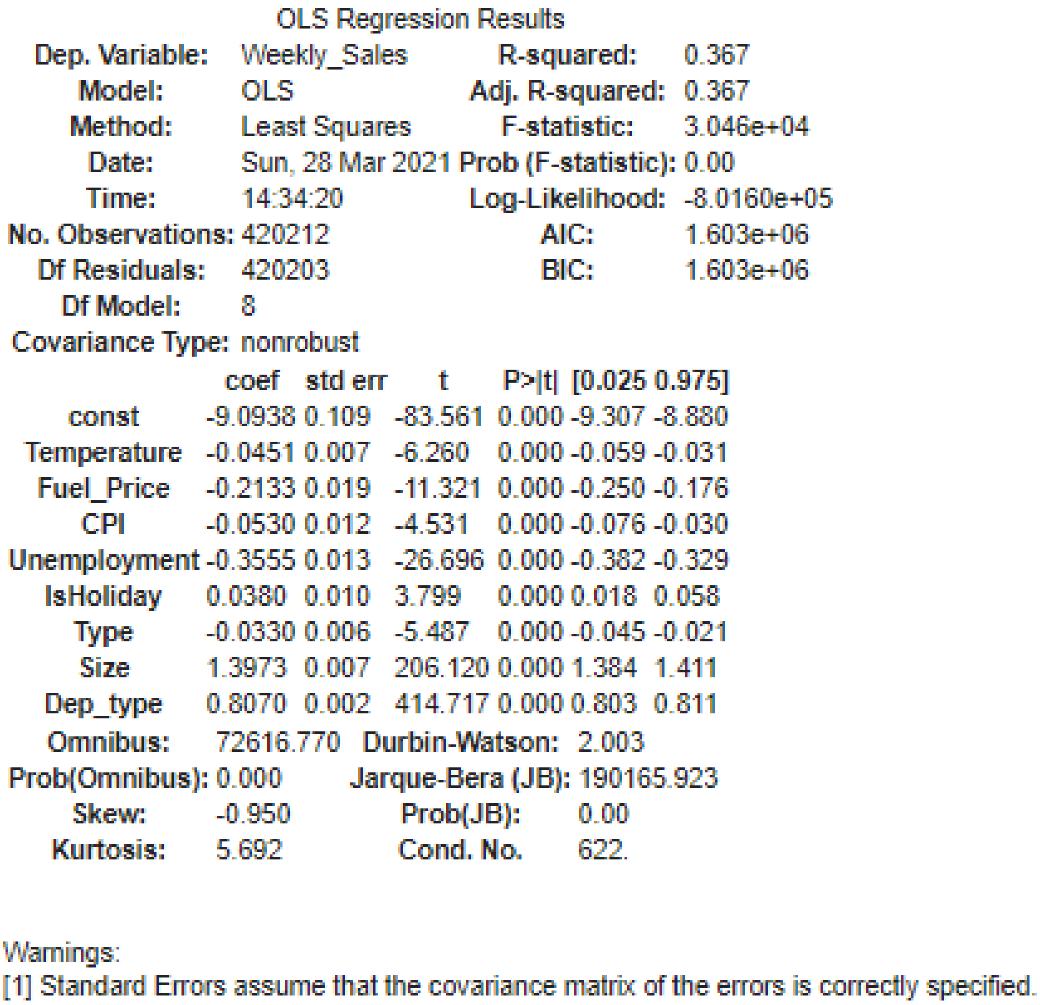
Boxplot after Log Transformation:



4.0 STATISTICAL MODELLING AND ASSUMPTIONS

4.1 Statistical Model (Base Model)

Based on the EDA and business problem we select the variables that are varying with the target variable and are significant in predicting the revenue. There are a few numerical columns that could have high multicollinearity among them, all this can be tested during assumptions testing after fitting a statistical model to the data. Before fitting the model, we have a number of categorical columns for which we need to create dummies. So, we now use an Ordinary Least Squares to create a statistical model using the data.



```
                          OLS Regression Results
     Dep. Variable:   Weekly_Sales         R-squared:        0.367
          Model:      OLS               Adj. R-squared:      0.367
         Method:      Least Squares        F-statistic:      3.046e+04
          Date:       Sun, 28 Mar 2021 Prob (F-statistic): 0.00
          Time:       14:34:20          Log-Likelihood:     -8.0160e+05
 No. Observations: 420212                    AIC:            1.603e+06
    Df Residuals:     420203                 BIC:            1.603e+06
       Df Model:      8
 Covariance Type:  nonrobust
                       coef   std err    t      P>|t|  [0.025 0.975]
      const        -9.0938 0.109   -83.561  0.000 -9.307 -8.880
   Temperature     -0.0451 0.007   -6.260   0.000 -0.059 -0.031
     Fuel_Price    -0.2133 0.019   -11.321  0.000 -0.250 -0.176
        CPI        -0.0530 0.012   -4.531   0.000 -0.076 -0.030
  Unemployment -0.3555 0.013   -26.696  0.000 -0.382 -0.329
     IsHoliday      0.0380  0.010   3.799    0.000 0.018  0.058
        Type        -0.0330 0.006   -5.487   0.000 -0.045 -0.021
        Size         1.3973  0.007   206.120 0.000 1.384  1.411
     Dep_type       0.8070  0.002   414.717 0.000 0.803  0.811
     Omnibus:      72616.770  Durbin-Watson:   2.003
 Prob(Omnibus): 0.000        Jarque-Bera (JB): 190165.923
       Skew:       -0.950         Prob(JB):       0.00
     Kurtosis:     5.692          Cond. No.       622.


Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Given above are the results of OLS regression, as we see the R-squared value is 0.367 which is mediocre so we need to improve the model, but before we try and improve the model we need to test the assumptions for the model. There are mainly 5 assumptions that we need to test:

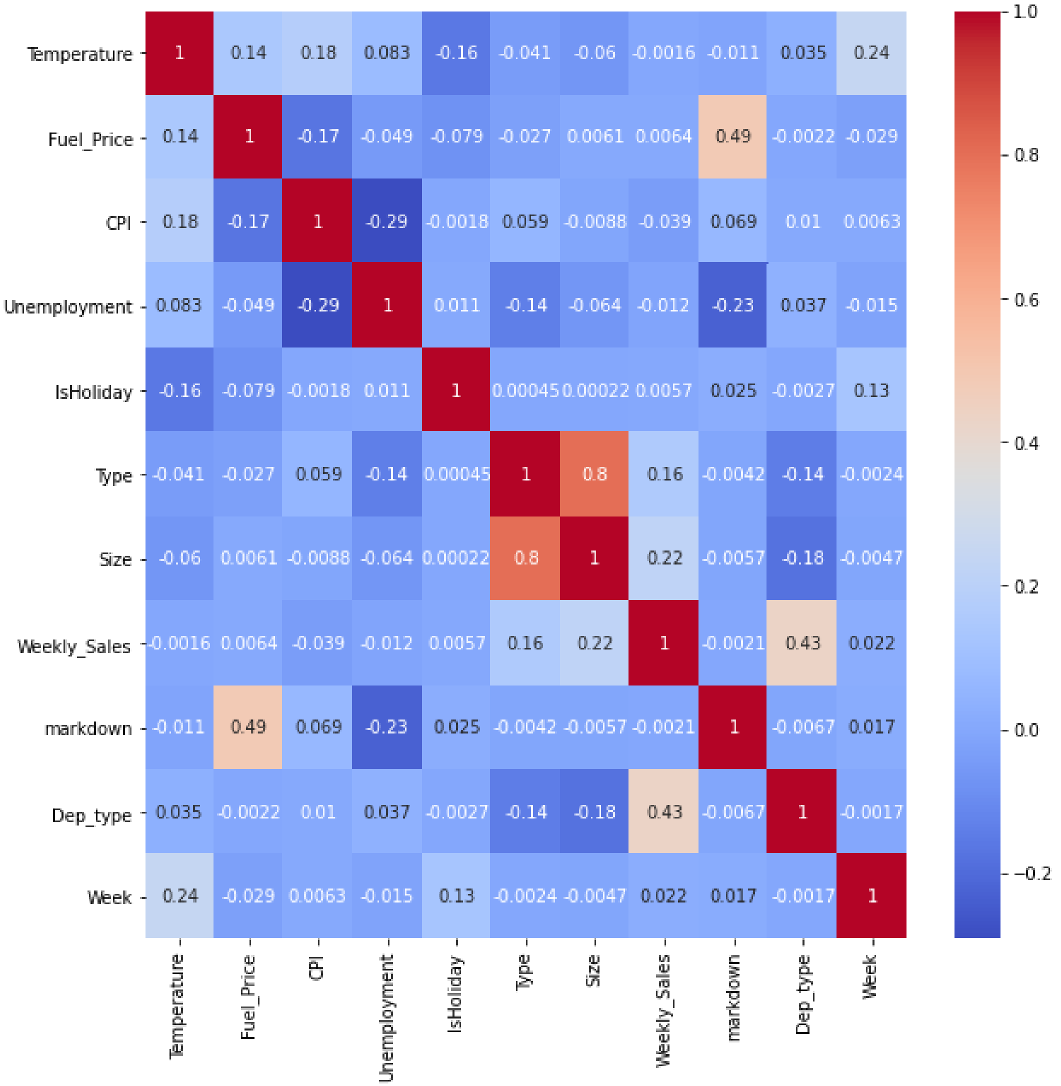
- Test of Multicollinearity
- Normality of Residuals
- Constant Variance of Residuals (Test for Homoscedasticity)
- Test for Autocorrelation
- Linearity of Relationship

We need to check these assumptions and need to be satisfied to go ahead with this model.

4.2 Assumption Testing

Test of Multicollinearity:

When checking for multicollinearity among the independent variables, we can first take a look at the correlation matrix to understand if there is any high correlation between the features. For this we take a look at the numerical features and not the dummy variables. The heatmap for that is given below:



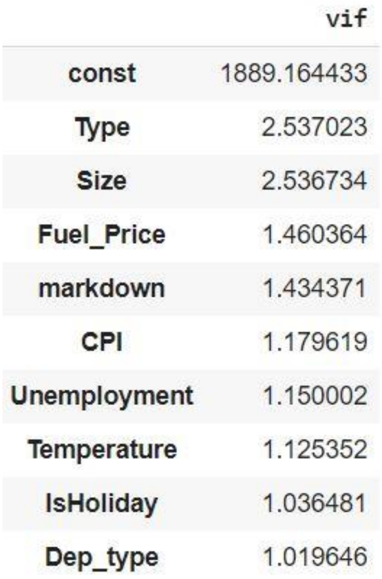
	Temperature	Fuel_Price	CPI	Unemployment	IsHoliday	Type	Size	Weekly_Sales	markdown	Dep_type	Week
Temperature	1	0.14	0.18	0.083	-0.16	-0.041	-0.06	-0.0016	-0.011	0.035	0.24
Fuel_Price	0.14	1	-0.17	-0.049	-0.079	-0.027	0.0061	0.0064	0.49	-0.0022	-0.029
CPI	0.18	-0.17	1	-0.29	-0.0018	0.059	-0.0088	-0.039	0.069	0.01	0.0063
Unemployment	0.083	-0.049	-0.29	1	0.011	-0.14	-0.064	-0.012	-0.23	0.037	-0.015
IsHoliday	-0.16	-0.079	-0.0018	0.011	1	0.00045	0.00022	0.0057	0.025	-0.0027	0.13
Type	-0.041	-0.027	0.059	-0.14	0.00045	1	0.8	0.16	-0.0042	-0.14	-0.0024
Size	-0.06	0.0061	-0.0088	-0.064	0.00022	0.8	1	0.22	-0.0057	-0.18	-0.0047
Weekly_Sales	-0.0016	0.0064	-0.039	-0.012	0.0057	0.16	0.22	1	-0.0021	0.43	0.022
markdown	-0.011	0.49	0.069	-0.23	0.025	-0.0042	-0.0057	-0.0021	1	-0.0067	0.017
Dep_type	0.035	-0.0022	0.01	0.037	-0.0027	-0.14	-0.18	0.43	-0.0067	1	-0.0017
Week	0.24	-0.029	0.0063	-0.015	0.13	-0.0024	-0.0047	0.022	0.017	-0.0017	1

Inference

- There seems to be high co relation between Size and Type of Store with a value of 0.81.
- Markdown and Fuel Price seems to be moderately co related with a value of 0.49.
- Other than this there is not much correlation between variables.
- There is no significant negative correlation found between the variables.
- No independent variable is correlated with the target variable.
- We can remove Type feature because it has collinearity with size feature.

From the heatmap we can understand that there is high correlation between all the numerical variables. So, there is very good possibility for multicollinearity among some of variables namely “Type”, “Size”. So, we go ahead and check the multicollinearity among the independent variables of the data. This is done by calculating the variance inflation factor (VIF) for the independent variables.

VIF – VARIANCE INFLATION FACTOR



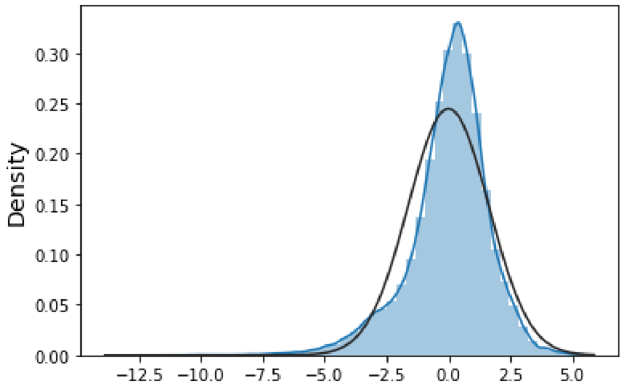
	vif
const	1889.164433
Type	2.537023
Size	2.536734
Fuel_Price	1.460364
markdown	1.434371
CPI	1.179619
Unemployment	1.150002
Temperature	1.125352
IsHoliday	1.036481
Dep_type	1.019646

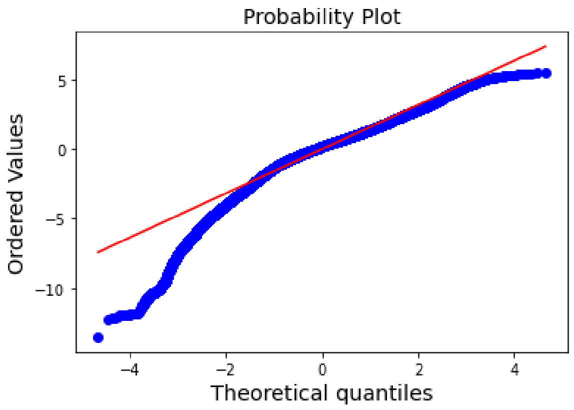
The variance inflation factor (VIF) for the independent variables are not above 3 in any of the features, so there is negligible variance influence.

Normality of Residuals:

Residuals in a machine learning or statistical model are the differences between the and predicted value. The residuals should be normally distributed. While a residual plot, or normal plot of the residuals can identify non-normality, you can formally test the hypothesis using the Jarque-Bera or similar test.

The null hypothesis states that the residuals are normally distributed, against the alternative hypothesis that they are not normally-distributed. If the test p-value is less than the predefined significance level, you can reject the null hypothesis and conclude the residuals are not from a normal distribution. If the p-value is greater than the predefined significance level, you cannot reject the null hypothesis.

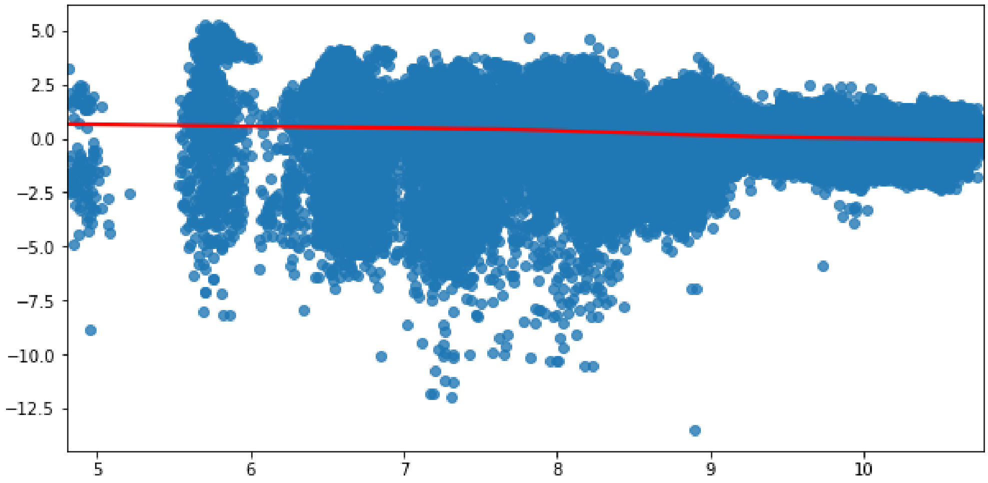




In the above plot we can see that the residuals are very high at 0 and tapers drastically towards either ends. The Jarque-Bera test also gives a p-value of 0.0, from which we understand that they are not normally distributed.

Constant Covariance of Residuals (Test for Homoscedasticity):

If the residuals are centred around 0 and they do not increase or decrease with the increase in the fitted line. To check the constant covariance, we use a residual versus fitted values plot. We can also use the Goldfeldquandt test. The plot is given below.



The points on the plot above appear to be randomly scattered around zero, so assuming that the error terms have a mean of zero is reasonable. The vertical width of the scatter doesn't appear to show a major increase or decrease across the fitted values, so we can assume that the variance in the error terms is constant. To confirm this we can go ahead with the Goldfeldquandt test. The null hypothesis is that, variance of residuals is constant across the range of data and the alternate hypothesis is that, variance of residuals is not constant across the range of data. From the test we got a p-value of 0.24 which is greater than the significance level. Hence, we fail to reject the null hypothesis and conclude that the residuals have constant covariance across the range of the data.

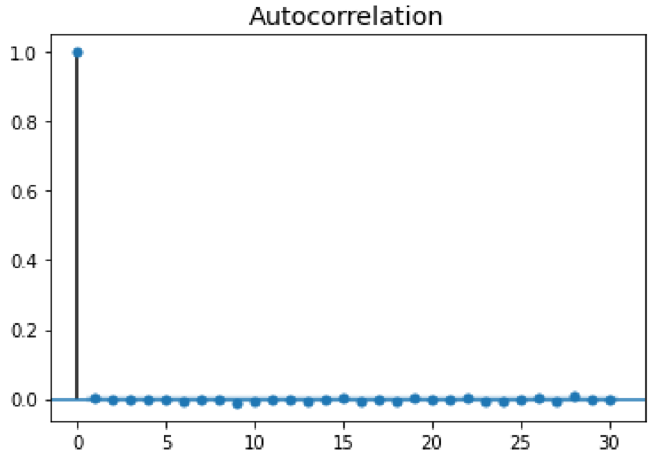
Test for Autocorrelation:

Autocorrelation refers to the degree of correlation between the values of the same variables across different observations in the data. The concept of autocorrelation is most often discussed in the context of time series data in which observations occur at different points in time. However, autocorrelation can also occur in cross-sectional data when the observations are related in some other way. Thus, autocorrelation can occur if observations are dependent in aspects other than time.

Autocorrelation can cause problems in conventional analyses (such as ordinary least squares regression) that assume independence of observations. In a regression analysis, autocorrelation of the regression residuals can also occur if the model is incorrectly specified. For example, if you are attempting to model a simple linear relationship but the observed relationship is non-linear (i.e., it follows a curved or U-shaped function), then the residuals will be auto correlated. fail to reject the null hypothesis and conclude that the residuals have constant covariance across the range of the data.

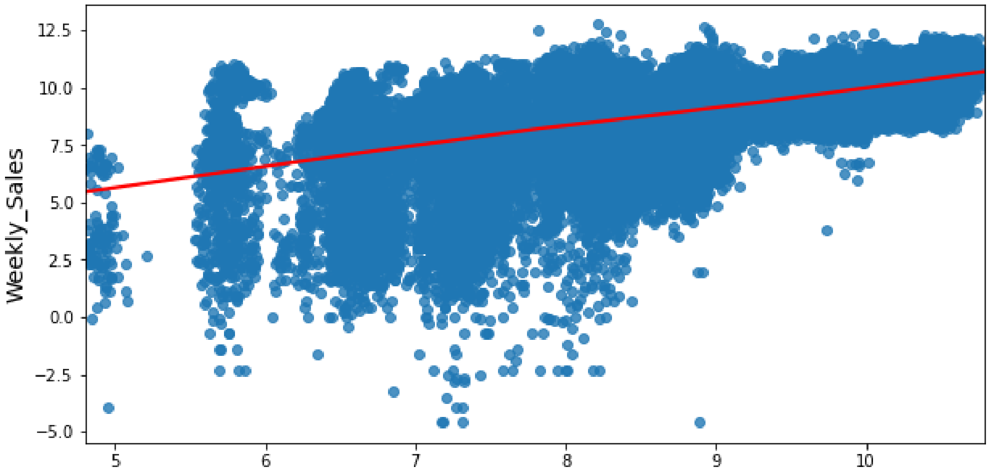
We can for autocorrelation using two methods; an ACF plot and the Durbin Watson Test. From the ACF plot for the data we find no pattern or relationship between the observations in the data. All the values are within the safe range. To back up this inference we do the Durbin-Watson test.

The Durbin Watson test values ranges from 0 to 4; 0 to <2 is positive autocorrelation, >2 to 4 is negative autocorrelation and 2 is no autocorrelation. Here we get the Durbin-Watson test value to be 2.003, which is very close 2, hence we conclude that there is no almost auto correlation in the data. The ACF plot for the data is given below.



Test for Linearity of Relationship:

Linearity of relationship refers to the relationship between the observed values and the predicted values. Looking at this we can understand if the statistical model built is appropriate or not. For this we look at the relationship of observed values with predicted values, and the variation of residuals with the predicted values.



We also conduct linear rainbow test. The hypothesis for the linear rainbow tests is as follows:

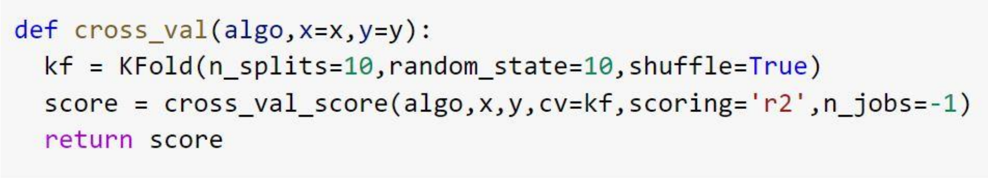
- Ho: fit of model using full sample = fit of model using a central subset (linear relationship)
- Ha: fit of model using full sample is worse compared to fit of model using a central subset

From the test we get a p-value of 0.857 which is greater than 0.05, so accept null.

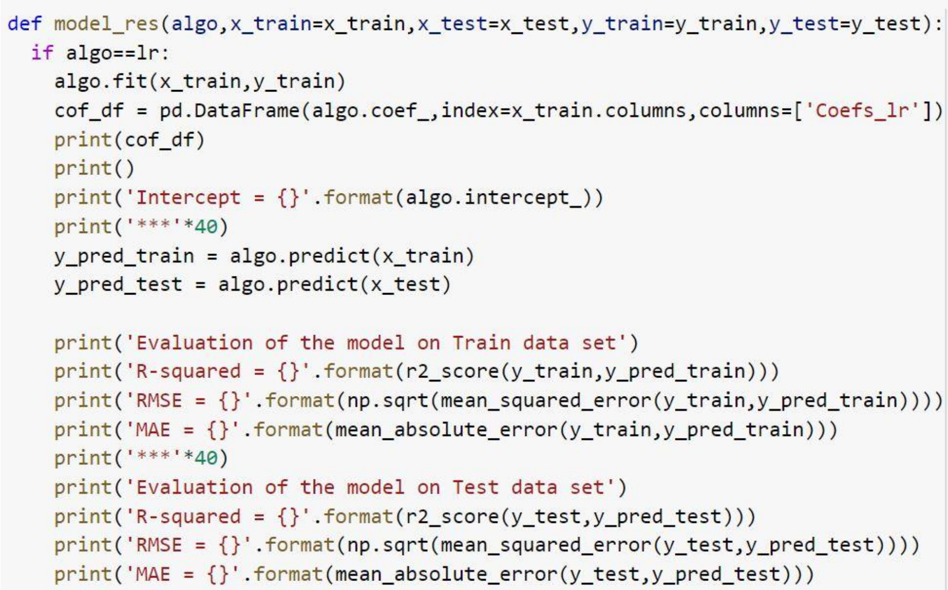
5.0 Regression Modelling.

Before we create a machine learning base model, we need to create two separate Data sets: With outliers in Target Variable, Without Outliers in the Target Variable. We also split the data into train and test using an 80:30 split for train and test respectively.

For ease of calculating we are creating some function for calculating cross-validation, model evaluation are creating a user defined function ‘cross_val’, ‘model_res’.

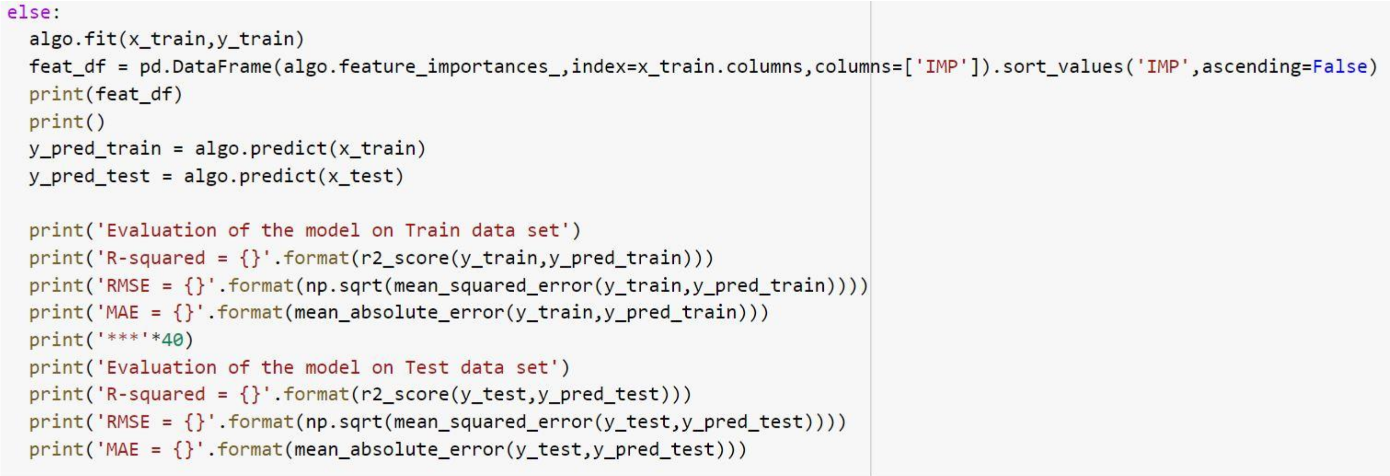


```
def cross_val(algo,x=x,y=y):
  kf = KFold(n_splits=10,random_state=10,shuffle=True)
  score = cross_val_score(algo,x,y,cv=kf,scoring='r2',n_jobs=-1)
  return score
```



```
def model_res(algo,x_train=x_train,x_test=x_test,y_train=y_train,y_test=y_test):
  if algo==lr:
    algo.fit(x_train,y_train)
    cof_df = pd.DataFrame(algo.coef_,index=x_train.columns,columns=['Coefs_lr'])
    print(cof_df)
    print()
    print('Intercept = {}'.format(algo.intercept_))
    print('***'*40)
    y_pred_train = algo.predict(x_train)
    y_pred_test = algo.predict(x_test)

    print('Evaluation of the model on Train data set')
    print('R-squared = {}'.format(r2_score(y_train,y_pred_train)))
    print('RMSE = {}'.format(np.sqrt(mean_squared_error(y_train,y_pred_train))))
    print('MAE = {}'.format(mean_absolute_error(y_train,y_pred_train)))
    print('***'*40)
    print('Evaluation of the model on Test data set')
    print('R-squared = {}'.format(r2_score(y_test,y_pred_test)))
    print('RMSE = {}'.format(np.sqrt(mean_squared_error(y_test,y_pred_test))))
    print('MAE = {}'.format(mean_absolute_error(y_test,y_pred_test)))
```



```
else:
  algo.fit(x_train,y_train)
  feat_df = pd.DataFrame(algo.feature_importances_,index=x_train.columns,columns=['IMP']).sort_values('IMP',ascending=False)
  print(feat_df)
  print()
  y_pred_train = algo.predict(x_train)
  y_pred_test = algo.predict(x_test)

  print('Evaluation of the model on Train data set')
  print('R-squared = {}'.format(r2_score(y_train,y_pred_train)))
  print('RMSE = {}'.format(np.sqrt(mean_squared_error(y_train,y_pred_train))))
  print('MAE = {}'.format(mean_absolute_error(y_train,y_pred_train)))
  print('***'*40)
  print('Evaluation of the model on Test data set')
  print('R-squared = {}'.format(r2_score(y_test,y_pred_test)))
  print('RMSE = {}'.format(np.sqrt(mean_squared_error(y_test,y_pred_test))))
  print('MAE = {}'.format(mean_absolute_error(y_test,y_pred_test)))
```

5.1.1 Linear Regression:

The most important aspect 0f linear regression is the Linear Regression line, which is also known as the best fit line. When dealing with a dataset in 2-dimensions, we come up with a straight line that acts as the prediction. If the data is in 3 dimensions, then Linear Regression fits a plane. However, if we are dealing with more than 3 dimensions, it comes up with a hyper-plane.

To keep things simple, we will discuss the line of best fit. If we were to establish a relationship between one independent and a dependent variable, this relationship could be understood as Y = 35 mx+c. The value of m is the coefficient, while c is the constant. To identify the value of m and c, we can use statistical formulas.

Following is the method for calculating the best value of m and c:
m = correlation between X and Y * (standard deviation of Y / standard deviation of X)
c = Mean of Y – (m *Mean of X)

Apart from this statistical calculation, as mentioned before, the line of best fit can be found by finding that value of m and c where the error is minimum. This is done by using optimization algorithms such as gradient descent, where the objective function is to minimize the sum of squared error (SSE).

Linear Regression also runs multiple statistical tests internally through which we can identify the most important variables. If the data is standardized, i.e., we are using the z scores rather than using the original variables. The value of coefficients becomes “calibrated,” i.e., we can directly look at the beta’s absolute value to understand how important a variable is. However, this is not true if we are using non-metric free variables. If our input variables are on different scales, then the absolute value of beta cannot be considered “weights” as these coefficients are “non-calibrated.”

To solve such a problem, Linear Regression runs multiple one sample t-tests internally where the null hypothesis is considered as 0, i.e., the beta of the X variable is 0. In contrast, the Alternative Hypothesis states that the coefficient of the X variable is not zero. This way, we take a clue from the p-value where if the p-value comes out to be high, we state that the value of the coefficient for that particular X variable is 0. The value we are seeing is statistically insignificant. Similarly, if we find the value of p to be lower than 0.05 or 0.1, then we state that the value of the coefficient is statistically significantly different from 0, and thus, that variable is important.

Once important variables are identified by using the p-value, we can understand their relative importance by referring to their t-value (or z-value), which gives us an in-depth understanding of the role played by each of the X variables in predicting the Y variable. Therefore, running a linear regression algorithm can provide us with dynamic results, and as the level of interpretability is so high, strategic problems are often solved using this algorithm.

5.1.2 Decision Tree

A decision tree is a tree-based supervised learning method used to predict the output of a target variable. Supervised learning uses labelled data (data with known output variables) to make predictions with the help of regression and classification algorithms. Supervised learning algorithms act as a supervisor for training a model with a defined output variable. It learns from simple decision rules using the various data features. Decision trees in Python can be used to solve both classification and regression problems—they are frequently used in determining odds

Important Terms Used in Decision Trees:
Entropy: Entropy is the measure of uncertainty or randomness in a data set. Entropy handles how a decision tree splits the data.
It is calculated using the following formula:

$$\sum_{i=1}^{k} P(value_i).log_2(P(value_i)$$

Information Gain: The information gain measures the decrease in entropy after the data set is split. It is calculated as follows:
IG (Y, X) = Entropy (Y) - Entropy (Y | X)

Gini Index: The Gini Index is used to determine the correct variable for splitting nodes. It measures how often a randomly chosen variable would be incorrectly identified.
Root Node: The root node is always the top node of a decision tree. It represents the entire population or data sample, and it can be further divided into different sets.

Decision Node: Decision nodes are sub nodes that can be split into different sub nodes; they contain at least two branches.

Leaf Node: A leaf node in a decision tree carries the final results. These nodes, which are also known as terminal nodes, cannot be split any further.

In machine learning, we use decision trees also to understand classification, segregation, and arrive at a numerical output or regression. In an automated process, we use a set of algorithms and tools to do the actual process of decision making and branching based on the attributes of the data. The originally unsorted data—at least according to our needs—must be analysed based on a variety of attributes in multiple steps and segregated to reach lower randomness or achieve lower entropy. While completing this segregation (given that the same attribute may appear more than once), the algorithm needs to consider the probability of a repeat occurrence of an attribute. Therefore, we can also refer to the decision tree as a type of probability tree. The data at the root node is quite random, and the degree of randomness or messiness is called entropy. As we break down and sort the data, we arrive at a higher degree of accurately-sorted data and achieve different degrees of information, or '"Information gain."

5.1.3 Random Forest

Random Forest is an ensemble machine learning technique capable of performing both regression and classification tasks using multiple decision trees and a statistical technique called bagging. Bagging along with boosting are two of the most popular ensemble techniques which aim to tackle high variance and high bias. A RF instead of just averaging the prediction of trees it uses two key concepts that give it the name random:

Random sampling of training observations when building trees
Random subsets of features for splitting nodes
In other words, Random forest builds multiple decision trees and merge their predictions together to get a more accurate and stable prediction rather than relying on individual decision trees.

Random sampling of training observations:

Each tree in a random forest learns from a random sample of the training observations. The samples are drawn with replacement, known as bootstrapping, which means that some samples will be used multiple times in a single tree. The idea is that by training each tree on different samples, although each tree might have high variance with respect to a particular set of the training data, overall, the entire forest will have lower variance but not at the cost of increasing the bias. In Sklearn implementation of Random forest the sub-sample size of each tree is always the same as the original input sample size but the samples are drawn with replacement if bootstrap=True. If bootstrap=False each tree will use exactly the same dataset without any randomness.

Random Subsets of features for splitting nodes:

The other main concept in the random forest is that each tree sees only a subset of all the features when deciding to split a node. In Sklearn this can be set by specifying max_features = sqrt(n_features) meaning that if there are 16 features, at each node in each tree, only 4 random features will be considered for splitting the node.

Why a Random Forest is better than a single decision tree?

The fundamental idea behind a random forest is to combine the predictions made by many decision trees into a single model. Individually, predictions made by decision trees may not be accurate but combined together, the predictions will be closer to the true value on average.

The objective of a machine learning model is to generalize well to new data it has never seen before. Overfitting occurs when a very flexible model (high capacity) memorizes the training data by fitting it closely. The problem is that the model learns not only the actual relationships in the training data but also any noise that is present. A flexible model is said to have high variance because the learned parameters (such as the structure of the decision tree) will vary considerably with the training data.

On the other hand, an inflexible model is said to have high bias because it makes assumptions about the training data (it's biased towards pre-conceived ideas of the data). An inflexible model may not have the capacity to fit even the training data and in both cases — high variance and high bias — the model is not able to generalize well to new data.

5.1.4 Light GBM

Light GBM is a framework that provides an implementation of gradient boosted decision trees. It's created by the researchers and developers' team at Microsoft. Light GBM is known for its faster-training speed, good accuracy with default parameters, parallel, and GPU learning, low memory footprint, and capability of handling large dataset.

The simplest way to create an estimator in LightGBM is by using the train() method. It takes as input estimator parameter as dictionary and training dataset. It then trains the estimator and returns an object of type Booster which is a trained estimator that can be used to make future predictions.

LGBMRegressor

LGBMRegressor is another wrapper estimator around the Booster class provided by LightGBM which has the same API as that of sklearn estimators. As its name suggests, it's designed for regression tasks. LGBMRegressor is almost the same as that of LGBMModel with the only difference that it's designed for only regression tasks. LGBMRegressor provides the score () method which evaluates the R2 score for us which we used to evaluate using the sklearn metric.

5.2 MODEL WITHOUT OUTLIERS

5.2.1 Cross validation scores of all Regression Models

Cross-validation is a statistical method used to estimate the skill of machine learning models. It is commonly used in applied machine learning to compare and select a model for a given predictive modelling problem because it is easy to understand, easy to implement, and results in skill estimates that generally have a lower bias than other methods.

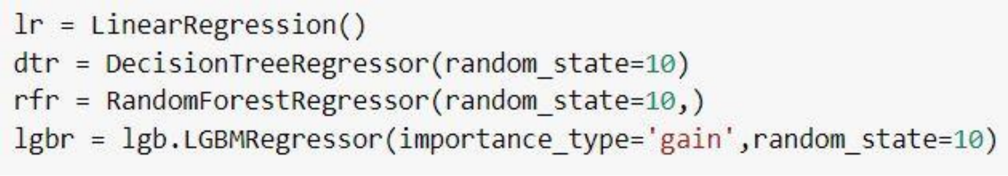
KFold Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample. The procedure has a single parameter called k that refers to the number of groups that a given data sample is to be split into. As such, the procedure is often called k-fold cross-validation. When a specific value for k is chosen, it may be used in place of k in the reference to the model, such as k=10 becoming 10-fold cross-validation.

Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.

It is a popular method because it is simple to understand and because it generally results in a less biased or less optimistic estimate of the model skill than other methods, such as a simple train/test split.

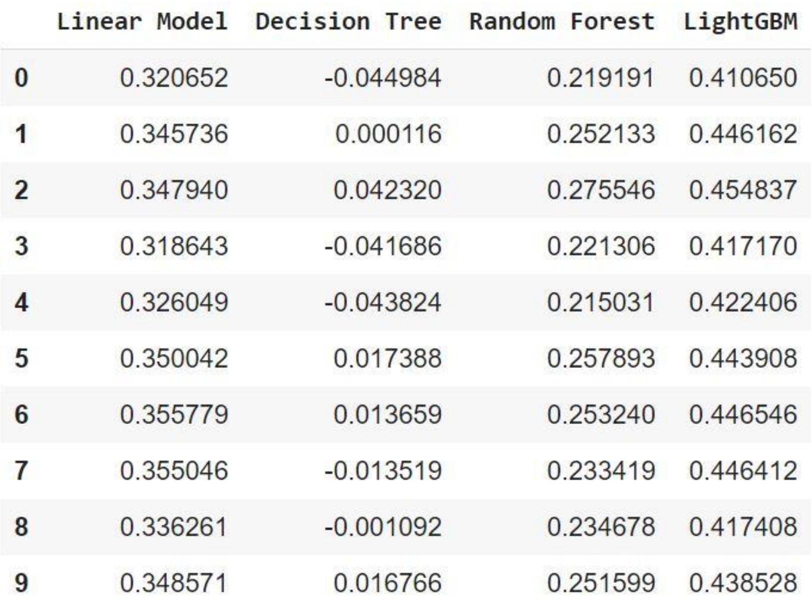
We took 10% sample of the dataset and done significance test to know that the sample taken represents the original dataset, the results from Shapiro and Mannwhitneyu Test results show that there is no significant difference between the means of the columns of the two datasets. Thus, we can say that the two datasets are similar.

Cross validating the below Regressors with kFold = 10



```
lr = LinearRegression()
dtr = DecisionTreeRegressor(random_state=10)
rfr = RandomForestRegressor(random_state=10,)
lgbr = lgb.LGBMRegressor(importance_type='gain',random_state=10)
```

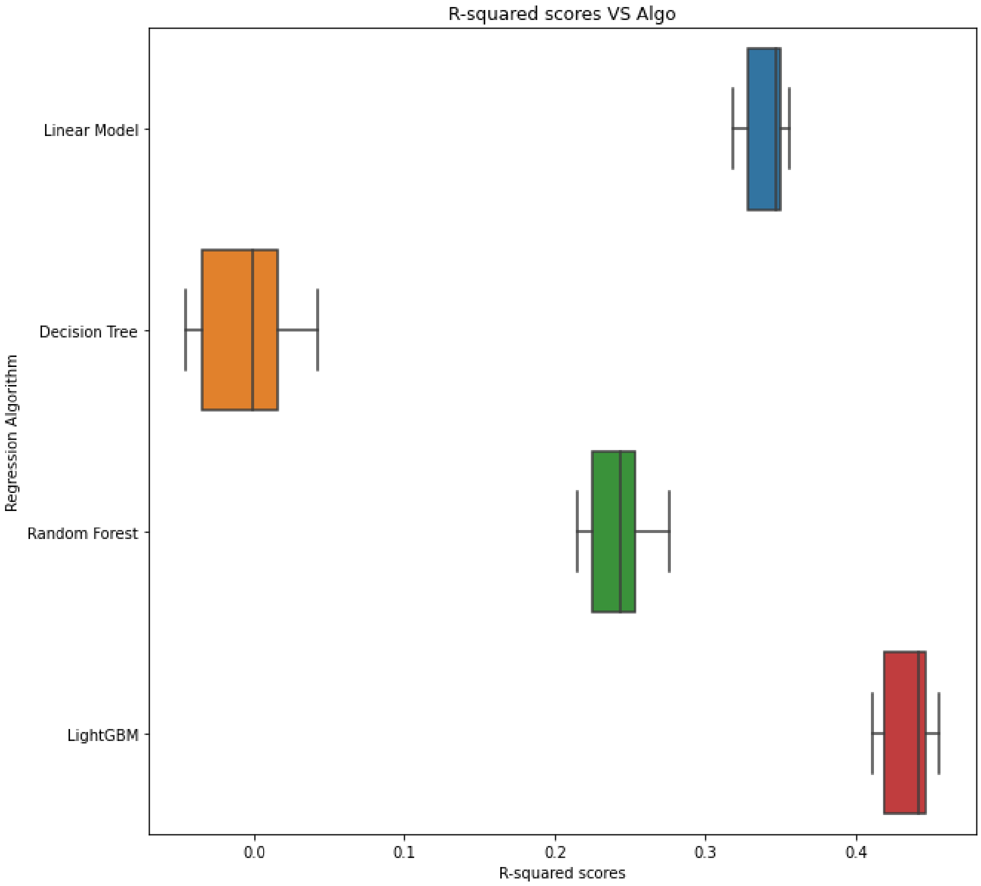
```
cross_score_df = pd.DataFrame({'Linear Model':cross_val(lr),'Decision Tree'
:cross_val(dtr),'Random Forest':cross_val(rfr),'LightGBM':cross_val(lgbr)})
```



	Linear Model	Decision Tree	Random Forest	LightGBM
0	0.320652	-0.044984	0.219191	0.410650
1	0.345736	0.000116	0.252133	0.446162
2	0.347940	0.042320	0.275546	0.454837
3	0.318643	-0.041686	0.221306	0.417170
4	0.326049	-0.043824	0.215031	0.422406
5	0.350042	0.017388	0.257893	0.443908
6	0.355779	0.013659	0.253240	0.446546
7	0.355046	-0.013519	0.233419	0.446412
8	0.336261	-0.001092	0.234678	0.417408
9	0.348571	0.016766	0.251599	0.438528

Box-plot of Regression Models:

From the boxplot for the regression models we can understand, the two models that give the best result are the Linear Regressor and LightGBMRegressor but looking at the boxplots for the two we can see that, even though the LightGBM gives a higher score, we can see a greater variation in the scores. The scores vary more for each cross validation set in LightGBM, but the scores are slightly more consistent for Linear Regression. But we can tune LightGBM to attain good results.



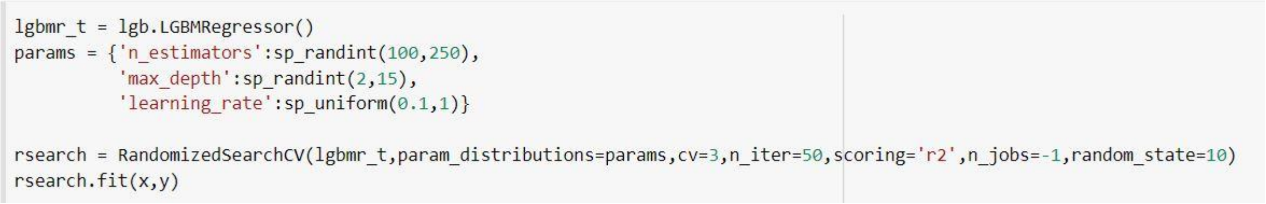
Inference based on the above graph:

- After performing the cross validation for all the Regression models, we found out that the average R-squared value for LightGBM regressor model is the highest.
- We can also see from the boxplot that there is not much variation in the obtained r-squared values for each split of train and test data through cross validation method.
- Thus, we will consider the LightGBM regressor model to predict the best model to fit our training data set and predict the target values for the test data.

5.2.2 Hyper-parameter tuning of Best Models

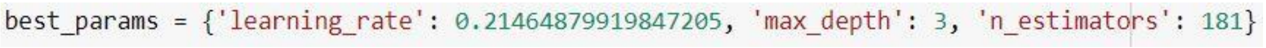
From above Boxplot of Regression, we observe that LightGBM performed the best, so we are to tune LightGBM.

Hyperparameter-tuning of **LightGBM regressor - LightGBM regressor** best hyperparameters.

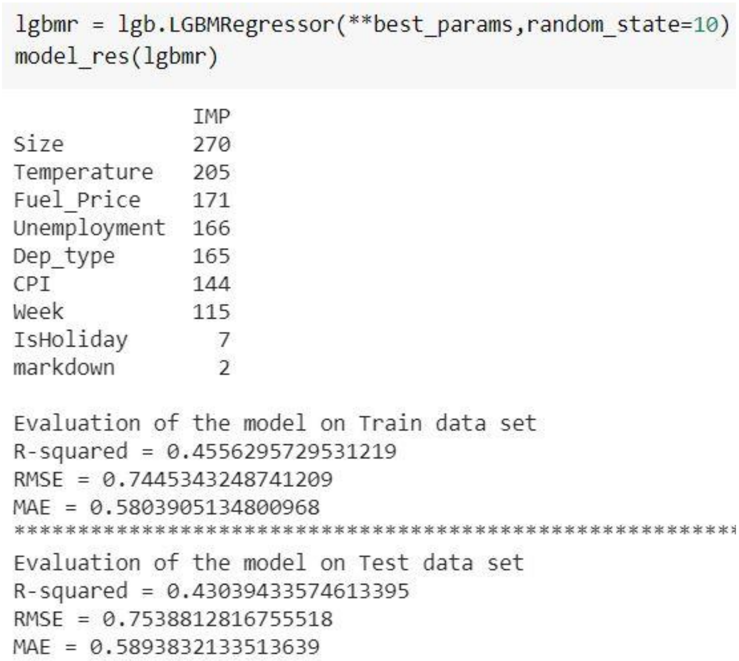


```
lgbmr_t = lgb.LGBMRegressor()
params = {'n_estimators':sp_randint(100,250),
          'max_depth':sp_randint(2,15),
          'learning_rate':sp_uniform(0.1,1)}

rsearch = RandomizedSearchCV(lgbmr_t,param_distributions=params,cv=3,n_iter=50,scoring='r2',n_jobs=-1,random_state=10)
rsearch.fit(x,y)
```



```
best_params = {'learning_rate': 0.21464879919847205, 'max_depth': 3, 'n_estimators': 181}
```



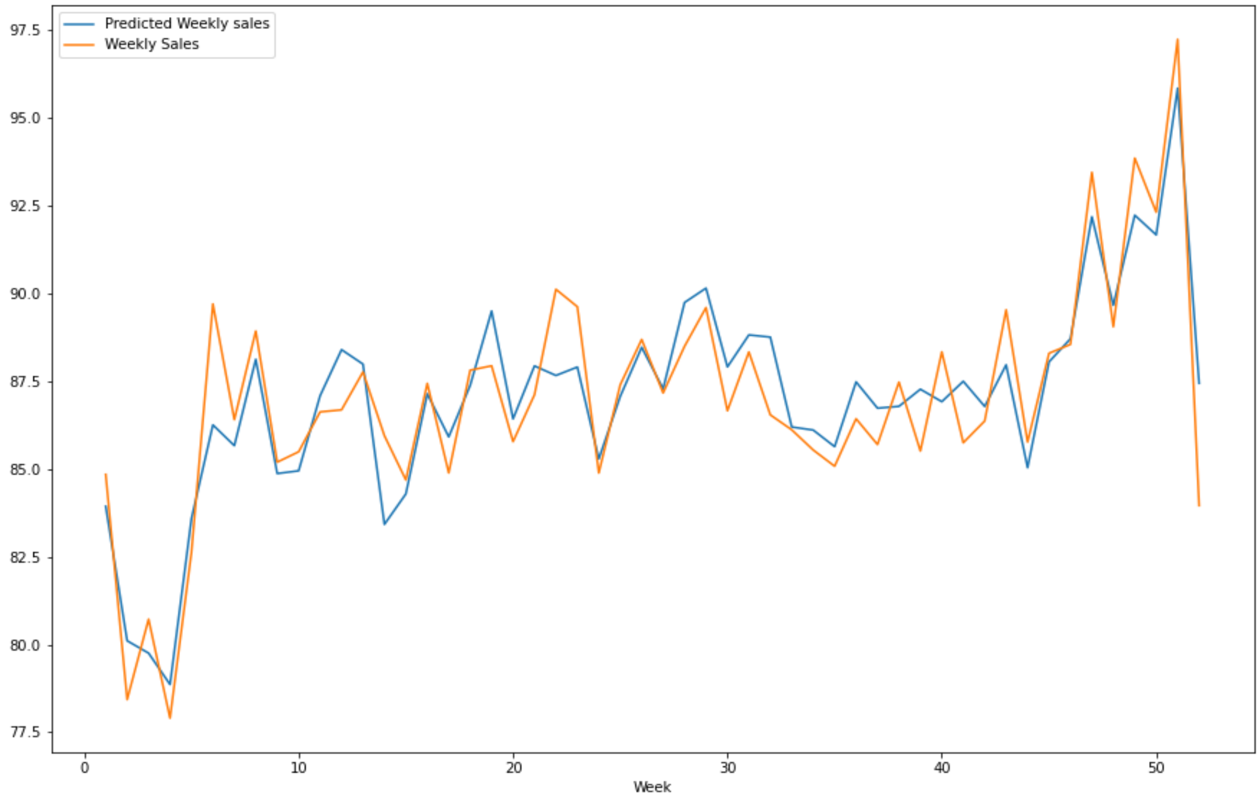
```
lgbmr = lgb.LGBMRegressor(**best_params,random_state=10)
model_res(lgbmr)
```

```
              IMP
Size          270
Temperature   205
Fuel_Price    171
Unemployment  166
Dep_type      165
CPI           144
Week          115
IsHoliday       7
markdown        2

Evaluation of the model on Train data set
R-squared = 0.4556295729531219
RMSE = 0.7445343248741209
MAE = 0.5803905134800968
*********************************************************
Evaluation of the model on Test data set
R-squared = 0.43039433574613395
RMSE = 0.7538812816755518
MAE = 0.5893832133513639
```

After tuning the results are slightly increased, we can plot the predicted and actual weekly sales to know our results better.

5.2.3 Predicted results using LightGBM



Inference based on the above graph:

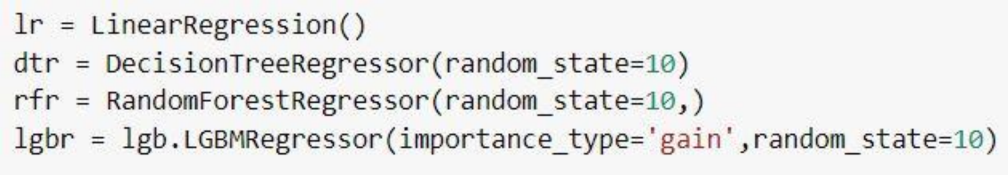
- From the above graph we can see that the predicted sales from the model built, is showing similar trend in the average weekly sales.
- Due to less accuracy of the model we can observe some biased results.

5.3 MODEL WITH ONLY OUTLIERS

The above same procedure is followed for this modelling also, so directly we move to results.

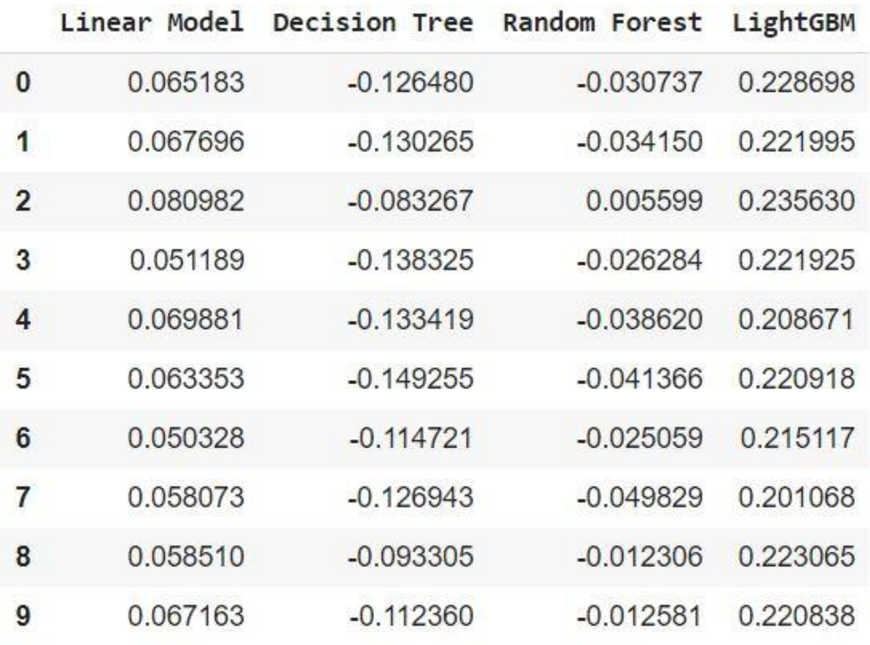
5.3.1 Cross validation scores of all Regression Models

Cross validating the below Regressors with kFold = 10



```
lr = LinearRegression()
dtr = DecisionTreeRegressor(random_state=10)
rfr = RandomForestRegressor(random_state=10,)
lgbr = lgb.LGBMRegressor(importance_type='gain',random_state=10)
```

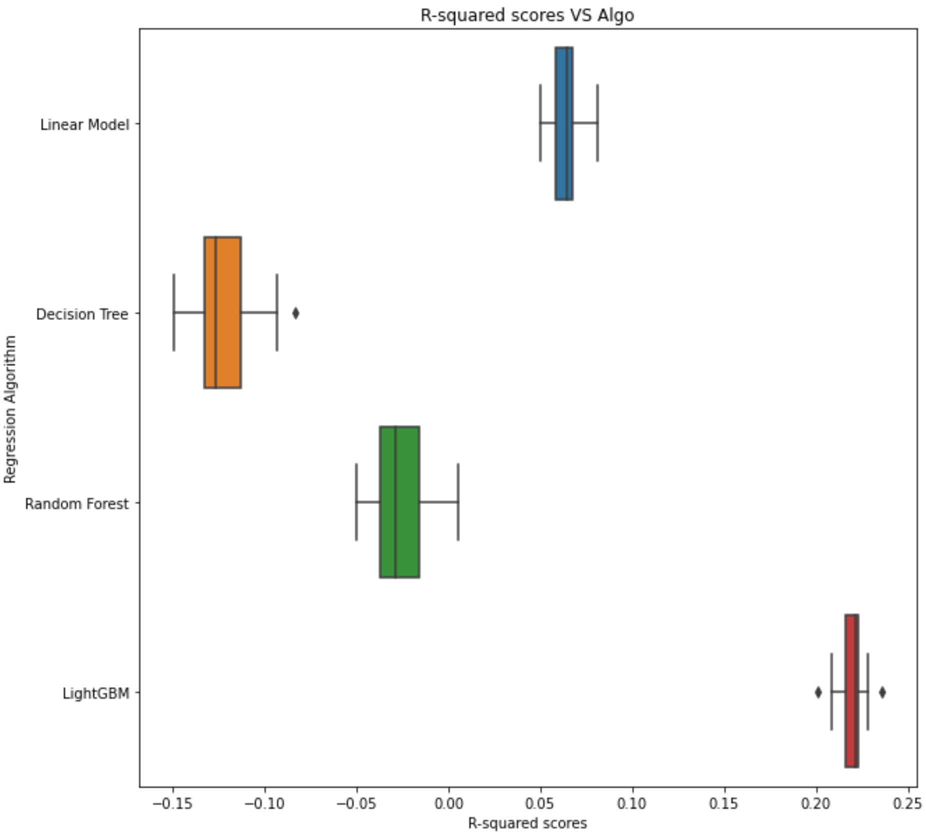
```
cross_score_df = pd.DataFrame({'Linear Model':cross_val(lr),'Decision Tree'
:cross_val(dtr),'Random Forest':cross_val(rfr),'LightGBM':cross_val(lgbr)})
```



	Linear Model	Decision Tree	Random Forest	LightGBM
0	0.065183	-0.126480	-0.030737	0.228698
1	0.067696	-0.130265	-0.034150	0.221995
2	0.080982	-0.083267	0.005599	0.235630
3	0.051189	-0.138325	-0.026284	0.221925
4	0.069881	-0.133419	-0.038620	0.208671
5	0.063353	-0.149255	-0.041366	0.220918
6	0.050328	-0.114721	-0.025059	0.215117
7	0.058073	-0.126943	-0.049829	0.201068
8	0.058510	-0.093305	-0.012306	0.223065
9	0.067163	-0.112360	-0.012581	0.220838

Box-plot of Regression Models:

From the boxplot for the regression models we can understand, the two models that give the best result are the Linear Regressor and LightGBM Regressor but looking at the boxplots for the two we can see that, even though the LightGBM gives a higher score, we can see a greater variation in the scores. The scores vary more for each cross validation set in LightGBM, but the scores are slightly more consistent for Linear Regression. But we can tune LightGBM to attain good results.



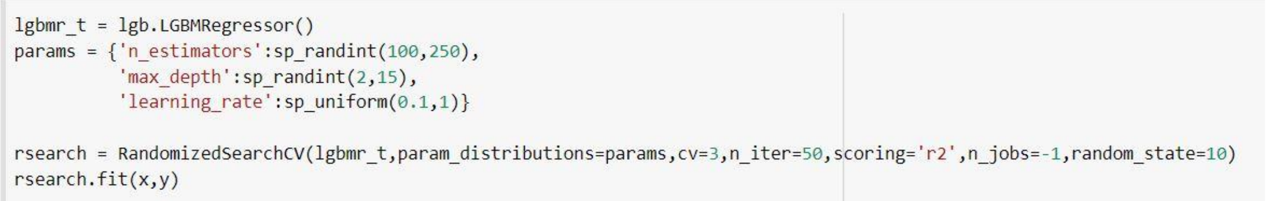
Inference based on the above graph:

- After performing the cross validation for all the Regression models, we found out that the average R-squared value for LightGBM regressor model is the highest.
- We can also see from the boxplot that there is not much variation in the obtained r-squared values for each split of train and test data through cross validation method.
- Thus, we will consider the LightGBM regressor model to predict the best model to fit our training data set and predict the target values for the test data.

5.3.2 Hyper-parameter tuning of Best Models

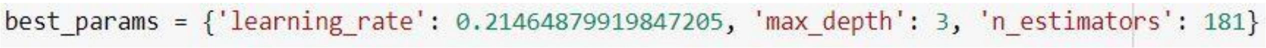
From above Boxplot of Regression we observe that LightGBM performed the best, so we are to tune LightGBM.

Hyperparameter-tuning of **LightGBM regressor - LightGBM regressor** best hyperparameters.

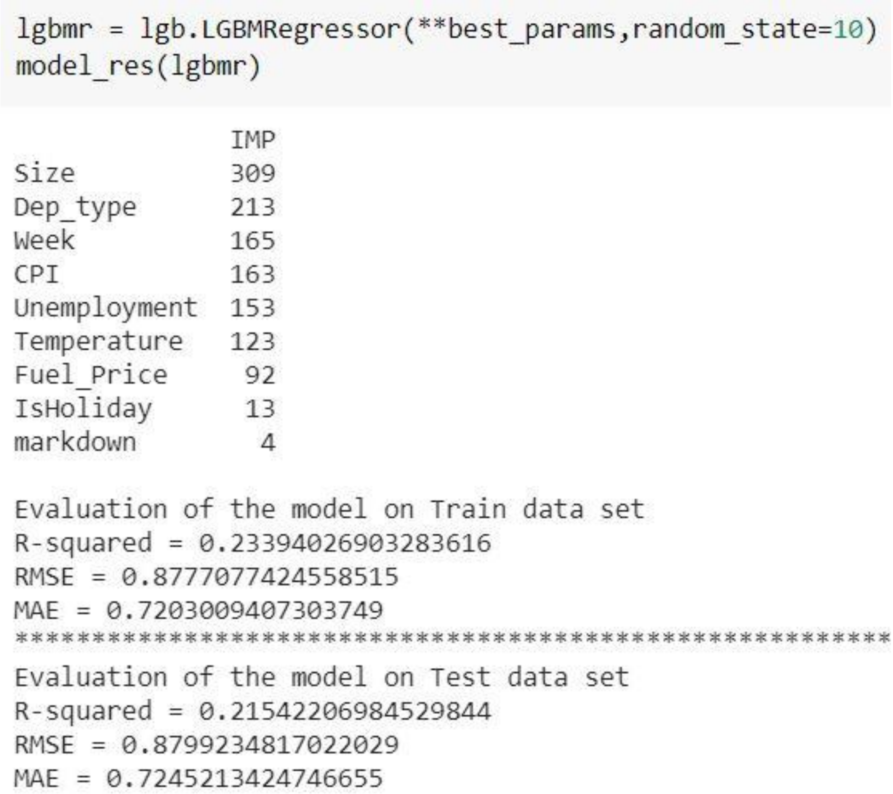


```
lgbmr_t = lgb.LGBMRegressor()
params = {'n_estimators':sp_randint(100,250),
          'max_depth':sp_randint(2,15),
          'learning_rate':sp_uniform(0.1,1)}

rsearch = RandomizedSearchCV(lgbmr_t,param_distributions=params,cv=3,n_iter=50,scoring='r2',n_jobs=-1,random_state=10)
rsearch.fit(x,y)
```



```
best_params = {'learning_rate': 0.21464879919847205, 'max_depth': 3, 'n_estimators': 181}
```

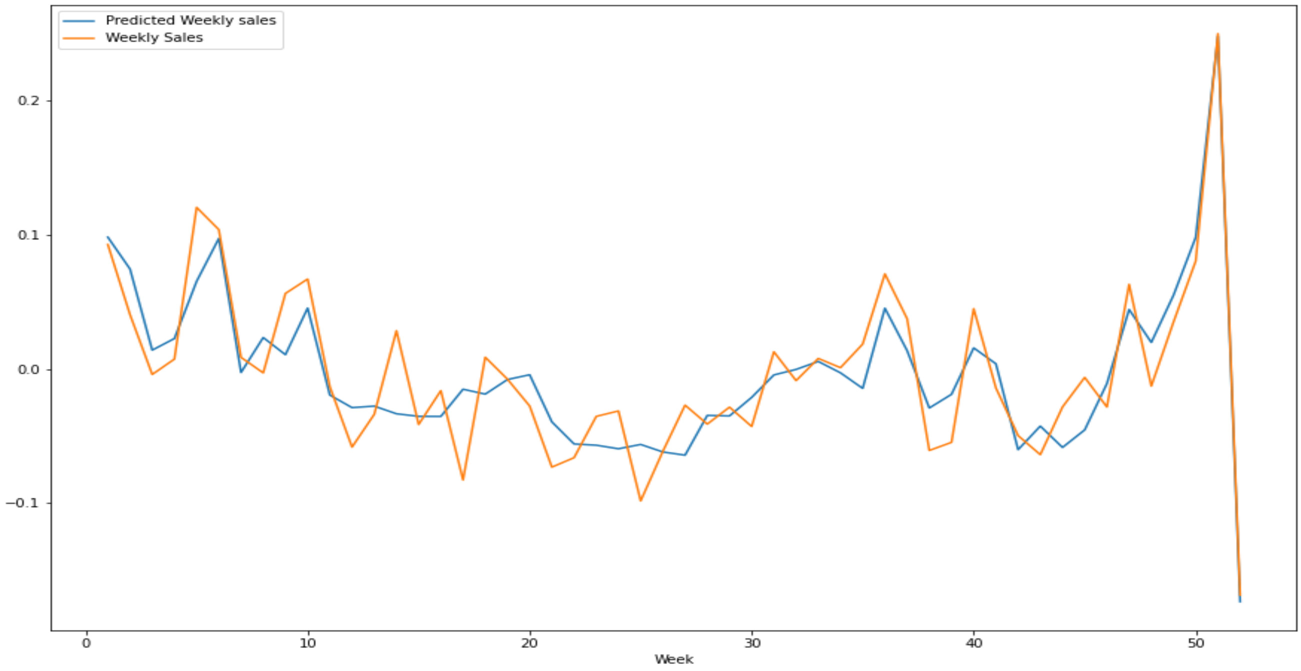


```
lgbmr = lgb.LGBMRegressor(**best_params,random_state=10)
model_res(lgbmr)
```

```
              IMP
Size          309
Dep_type      213
Week          165
CPI           163
Unemployment  153
Temperature   123
Fuel_Price     92
IsHoliday      13
markdown        4

Evaluation of the model on Train data set
R-squared = 0.23394026903283616
RMSE = 0.8777077424558515
MAE = 0.7203009407303749
*********************************************************
Evaluation of the model on Test data set
R-squared = 0.21542206984529844
RMSE = 0.8799234817022029
MAE = 0.7245213424746655
```

5.3.3 Predicted results using LightGBM



Inference based on the above graph:

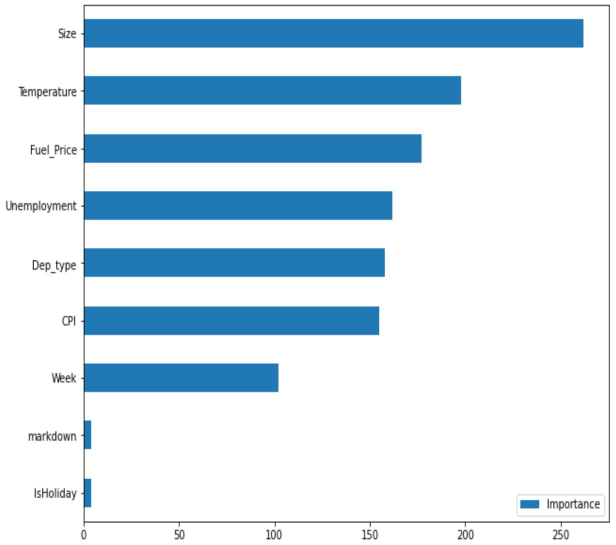
- From the above graph we can see that the predicted sales from the model built, is showing similar trend in the average weekly sales.
- Since, the accuracy of the overall model is very less thus, the predicted model is more biased to predict the outliers.

6.0 RESULT

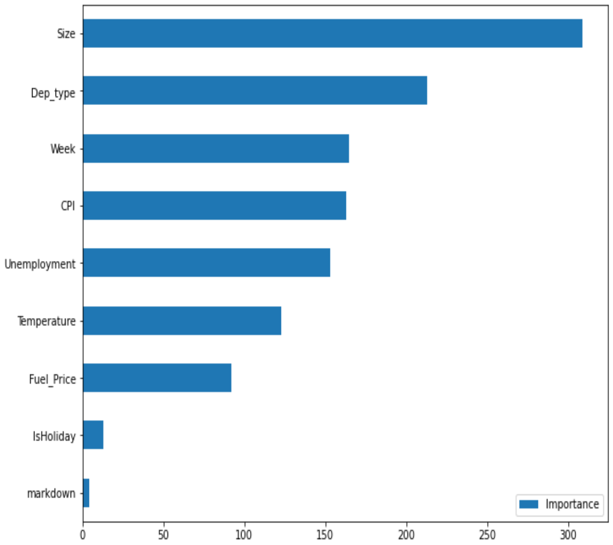
Feature importance of both models.

The feature importance is set to ‘split’ and calculation are carried out in LightGBM.

Model without Outliers



Model with Outliers



Metrics	Train	Test
R2_score	0.455	0.430
RMSE	41.193	41.697
MAE	32.13	32.61

Metrics	Train	Test
R2_score	0.233	0.215
RMSE	0.877	0.879
MAE	0.720	0.724

Inferences:

- We can observe that the model to predict the outliers in the weekly sales is less accurate than the model to predict weekly sales without outliers.
- The RMSE values have negligible difference for train and test datasets for both the models.
- The Mean Absolute Error also shows negligible difference for train and test data for each of the two models.
- Thus, we can conclude that each of the models created are neither overfitting or underfitting the training and test datasets.

7.0 CONCLUSION

1. From the data we were able to design three models including the statistical model as the base model to check the statistical significance of the variables.
2. The other two models were designed for data with and without outliers in the Weekly Sales.
3. The accuracy of the regression model is relatively high in the dataset without the outliers as compared to the dataset having the outliers in the Target variable.
4. From the above two models (both with and without outliers) we can infer that the weekly sales are not dependent on whether any promotion is running in the store or not. It also tells us that there is no significant change in the weekly sales on the holidays.
5. Though, we got decent accuracy for the model without the outliers, the model is still insignificant for the weekly sales prediction as some of the variables which have shown more significance in the final model are not in control of the company e.g. Temperature, fuel price, CPI and Unemployment rate.
6. From the model designed for outliers in the weekly sales we can observe that size, dept_type, week, are the most important features to predict extremely high weekly sales.

Limitations

1. The data set contained more than 50% of null values in some features. Thus, such features were dropped.
2. A significant amount of outliers were present in the target variable, making it difficult to predict a good regression model for the given data.
3. No information given on the different departments; thus we cannot make any business interpretation out of that feature.
4. Many features are not in control of the company to manipulate, but are showing significant effect in predicting the target variable.

References

Novel retail technologies and marketing analytics Maria Petrescu and Anjala S. Krishen (2017)

Literature review - A review of the literature on Marketing Analytics for Data Rich environments by Michel Wedel & P.K. Kannan (Journal of Marketing: AMA/MSI Special Issue, November 2016)

Statistics, 4th ed., by David Freedman, Robert Pisani, and Roger Purves (W. W. Norton, 2007) has an excellent discussion of correlation.

Modern Data Science with R by Benjamin Baumer, Daniel Kaplan, and Nicholas Horton (Chapman & Hall/CRC Press, 2017) has an excellent presentation of "a grammar for graphics"

Any introductory statistics text will have illustrations of the t-statistic and its uses; two good ones are *Statistics*, 4th ed., by David Freedman, Robert Pisani, and Roger Purves (W. W. Norton, 2007), and *The Basic Practice of Statistics*, 8th ed., by David S. Moore, William I. Notz, and Michael A. Fligner (W. H. Freeman, 2017).

An excellent short treatment of multi-arm bandit algorithms is found in *Bandit Algorithms for Website Optimization*, by John Myles White (O'Reilly, 2012).

An Introduction to Statistical Learning by Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani (Springer, 2013).

Allen, D. M. [1971], "Mean square error of prediction as a criterion for selecting variables," *Technometrics*, **13**, 469–475.

Andrews, D. F. [1971], "Significance tests based on residuals," *Biometrika*, **58**, 139–148.

Andrews, D. F., P. J. Bickel, F. R. Hampel, P. J. Huber, W. H. Rogers, and J. W. Tukey [1972], *Robust Estimates of Location*, Princeton University Press, Princeton, N.J.

Prairie, Y. T. 1996. Evaluating the predictive power of regression models. Can. J. Fish. Aquat. Sci **53**: 490-492.

Prairie, Y. T., R. H. Peters, & D. F. Bird 1995. Natural variablilty and the estimation of empirical relationships: a reassesssment of regression methods. Can. J. Fish. Aquat. Sci **52**, 490-492.

Yule, G. Udny (1897). "On the Theory of Correlation". Journal of the Royal Statistical Society. ***60*** *(4): 812–54. doi:10.2307/2979746. JSTOR 2979746.*

Pearson, Karl; Yule, G.U.; Blanchard, Norman; Lee,Alice (1903). "The Law of Ancestral Heredity". Biometrika. ***2*** *(2): 211–236. doi:10.1093/biomet/2.2.211. JSTOR 2331683.*

Fisher, R.A. (1922). "The goodness of fit of regression formulae, and the distribution of regression coefficients". Journal of the Royal Statistical Society. ***85*** *(4): 597 612. doi:10.2307/2341124. JSTOR 2341124. PMC 1084801.*

Ronald A. Fisher (1954). Statistical Methods for Research Workers (Twelfth ed.). Edinburgh: Oliver and Boyd. ISBN 978-0-05-002170-5.

Aldrich, John (2005). "Fisher and Regression". Statistical Science. ***20*** *(4): 401 417. doi:10.1214/088342305000000331. JSTOR 20061201.*