IntelliPaat

Capstone Project

Walmart

Kavitha Palraj
kavipalraj.9@gmail.com

Table of Contents

Introduction

The retail giant Walmart are renowned for being the go-to-destination for household shopping. In 2020 Walmart generated a revenue of 524 billion dollars (Cascade, 2022) and they operate in more than 10,500 stores globally (Walmart, 2022). This large scale success results in the generation and storage of a colossal influx of data. Understanding how to handle and manipulate this data is key in ensuring Walmart are able to identify successes and analyse failures and pivot into solutions to address any issues.

a. Walmart Data in this report

   Walmart stores across the world collect data regarding various aspects of the store operations. The use of this data varies extensively across the outlets contributing to strategic decision-making. One way this is achieved is through future prediction and forecasting. Performing these tasks guides the company in adapting to market dynamics, improving operational efficiency, and ensuring that it continues to meet the evolving needs of its diverse customer base.

1. 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. Use predictive modelling techniques to forecast the sales for each store for the next 12 weeks.

2. Project Objective

The objective of this project is to develop and implement advanced predictive forecasting models to predict the sales for various Walmart stores for the next 12 weeks. Enabling stores to view this information will enable them to adapt to scenarios, address problems and optimise revenue generation. Alongside this main objective, secondary questions will also be answered such as does temperature affect the weekly sales in any manner, what are the top performing stores according to historical data and do weekly stores show a seasonal trend.

3. Data Description

a. Overview

The Walmart dataset that is available contains weekly sales information from 45 different stores. The data was collected with the aim of recording and storing various characteristics about these stores. There are 6,435 rows.

b. Continuous attributes

Variable	Description	Data Values
Store	Store number	1-45
Date	Week the sales were made	5-2-2010 to 26-10-2012
Weekly_Sales	Sales for the given store in that week	209986.5 - 3818686.45
Holiday_Flag	If it is a holiday week	0, 1
Temperature	Temperature on the day sale	7.46^{o}C – 100.14^{o}C
Fuel_price	Cost of the fuel in the region	$2.472 - $4.468
CPI	Consumer of Price Index	126.064 - 227.232
Unemployment	Unemployment Rate	4.31 - 10.926

c. Data overview

The 'Weekly_Sales' column in the dataset serves as a crucial indicator, capturing the financial performance of various stores. This metric reflects the amount of money generated by each store during a specific period. Complementing this key data point are several other columns that provide insights into the contextual factors influencing sales.

These factors include the 'Holiday_Flag' which denotes the presence of holidays during the sales period, 'Temperature ' which indicates weather conditions, 'Fuel Price' which reveals the cost of fuel at the time, 'CPI' (Consumer Price Index) reflecting inflationary pressures, and 'Unemployment' depicting the employment landscape.

Analysing the correlation between these variables and weekly sales can unveil patterns and trends, offering valuable information for strategic decision-making in the retail domain.

d. Insights from the data

A few insights can be made from the data before the following pre-processing, EDA and model building steps. As mentioned above, the data contains information about 45 different stores.

4. Inspiration

- Explore the potential for dynamic pricing strategies based on real-time influence, workforce demand ,aiding in optimized employees scheduling to meet varying sales needs.
- To find the irregularities in sales patterns ,assisting in the early identification of issues that may impact revenue.
- Understand the correlation between sales and economic indicators like CPI and unemployment, offerings insights into broader economic context affecting consumer behaviour
- Finding out machine learning algorithms to predict future weekly sales based on historical data and the given factors, enabling better inventory management and resource allocation

Data Pre-processing Steps

Data pre-processing is an important step in improving the reliability and interpretability of the dataset (IBM, 2022). **Exploratory Data Analysis (EDA)** helps in showing the initial structure of the data. **Data Cleaning** is included within EDA which ensures the data is well structured and free from issues such as outliers and missing values. This enables us to analyse the data to a higher accuracy and create machine learning models which perform optimally. EDA is also used to create visualizations are such as boxplots and correlation heatmaps which help in the subsequent model building steps.

The individual steps that were carried out and their results are elaborated upon below:

a. Importing the libraries

The first step in EDA is importing all the libraries that will be necessary for the different process carried out.

1. `numpy` is used mainly for scientific computation and mathematical operations. It provides features such as a numpy array which is a powerful N-dimensional array object and functions for linear algebra.

2. `pandas` is used for data cleaning, pre-processing and manipulation. This is achieved using functions such as `.isnull()` and `groupby()` which will be seen later on. Series and DataFrames are used which structure data for manipulation and analysis.

3. `matplotlib` and `seaborn` are both used for data visualization.

b. Loading the dataset

Pandas is used to load and read the dataset from the provided .csv file. This is done using the `pd.read_csv()` function. This .csv file is then converted to a DataFrame which is used for the further pre-processing steps. The DataFrame is stored in the variable `df`.

c. Information summary of the dataset

The `df.info()` function is applied to the DataFrame which provides a breakdown of each column.

There are a total of `6435` entries. The dataset has 8 different columns which were shown in the Data Description. Each column has `6435 non-null` values. There are 2 columns with data type `int64` and 6 columns with data type `float`.

`df.dtypes` shows us the data type of each column that was seen with `df.info()`.

d. Handling null values and missing values

Leaving null and missing values within a dataset can compromise the integrity of the dataset which leads to bias in the analysis and a model which makes inaccurate predictions and conclusions. `df.isnull.sum()` and `df.isna.sum()` show the dataset has no null or missing values that need to be removed.

e. Handling duplicate values

Duplicate values can lead to a biased analysis, inaccurate results and a machine learning model which overfits due to learning patterns from duplicated data. Using `df.duplicated.sum()`, the dataset contains 0 duplicated values.

f. Outlier detection

The detection of outliers is a very important step in data pre-processing. It involves spotting values that deviate significantly from the overall pattern and removing them. Outliers can cause distortions in a dataset which lead to a machine learning model making inaccurate predictions.

Box plots are plotted for numerical columns using `sns.boxplot()` within a for loop which iterates over the dataset columns.

Following this visualization, outliers are detected. Upper and lower quantiles are used to calculate an IQR and values which fall outside this IQR are labelled as outliers. These outliers are removed from the DataFrame using `df=df[(df[i]>=LW) & (df[i]<=UW)].`

g. Feature engineering

Columns are added to the dataset for 'Year', 'Month, 'Week', 'WeekOfYear' and 'Day'. These columns enable us to examine further aspects of the dataset and receive more detailed insights.

h. Label encoding

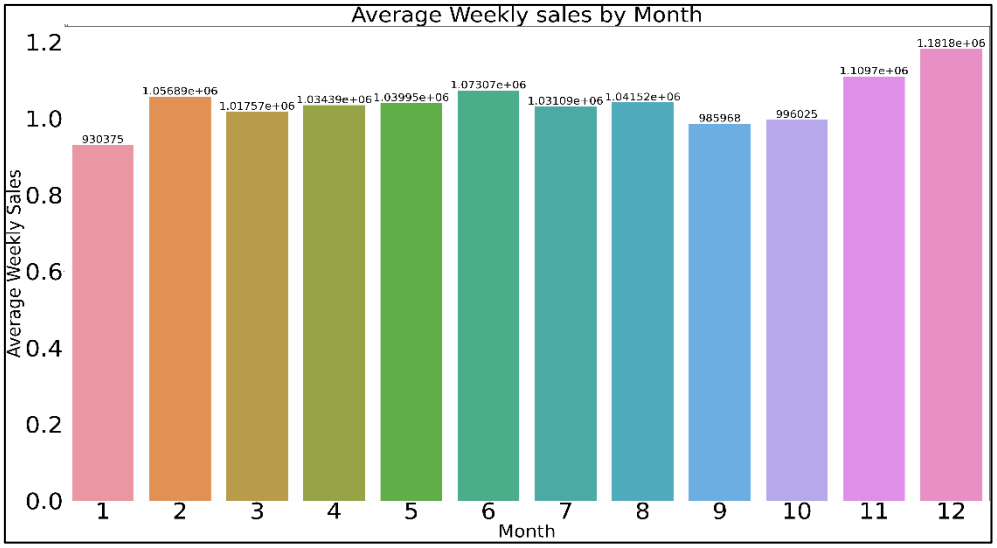
Label encoding is used to transform categorical variables to numerical variables. This process is needed because machine learning models require `int` and `float` as input data types. This is done using the `LabelEncoder()` function. A for loop is used to iterate over all columns with `object` data type. `le.fit_transform()` is used to convert the data in these columns to `int`.

The numerical columns are then visualised in correlation plots as done previously. These plots show very weak correlations between all columns.

Data Visualization

Data visualization is a very important step. It transforms information within the dataset into a visual format where it is easier to spot trends, patterns and draw insights from the data. Alongside the given problem statement, these visualizations provide further data exploration to understand different relationships present (TechTarget, 2022). These relationships can be explained and any outliers or anomalies can also be identified.

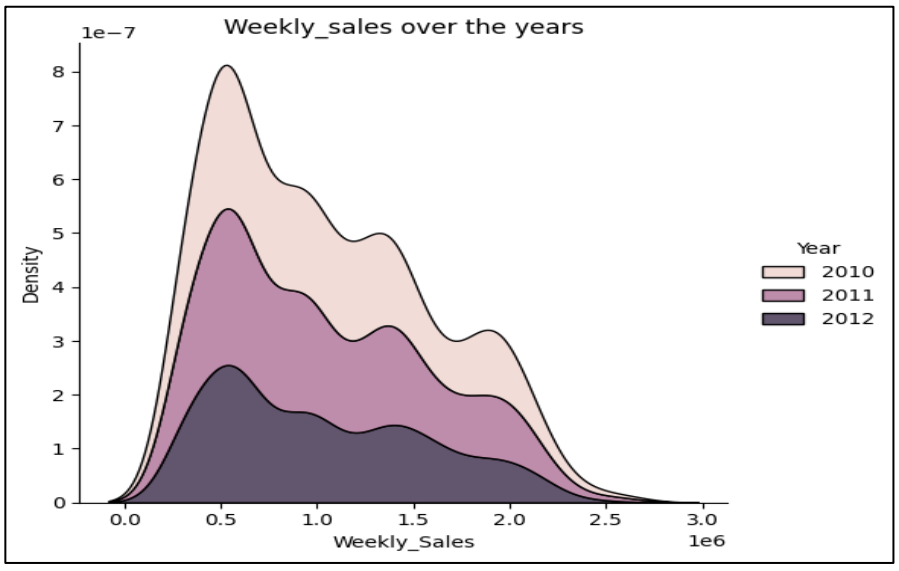
a. Average sales by month



This graph shows the average weekly sales by month remains relatively consistent throughout the year. The largest number of sales were recorded in December at 1,181,800. This may be due to the Christmas and New Year holidays. The lowest number of sales were recorded in January at 930,375.

Data is grouped by 'Month' and 'Weekly_Sales' is aggregated with mean to find out mean sales in each month for every year. Then `sns.barplot()` was used to plot this graph.

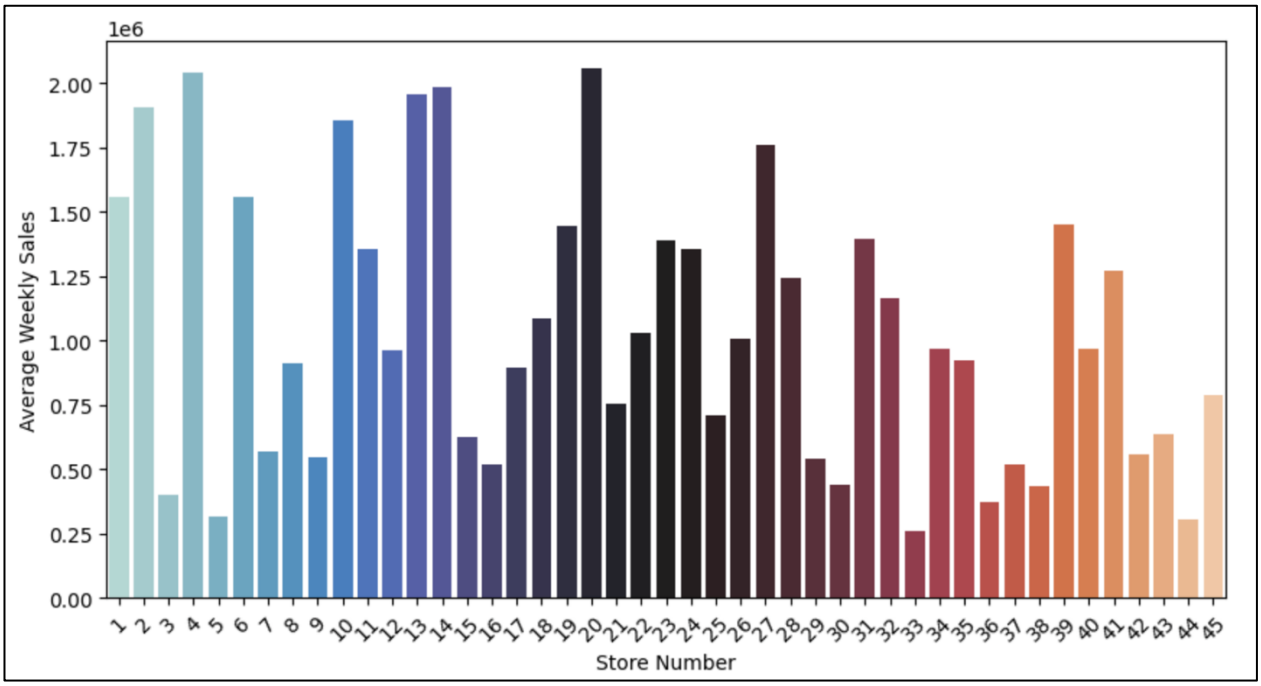
b. Distribution of weekly sales every year



This graph shows the distribution of the number of weekly sales. The highest concentration of sales for all 3 years can be seen around 600,000 a week. For 2012, the KDE plot shows the lowest density, suggesting a more dispersed distribution with a wider range of weekly sales values.

Weekly sales over the years is obtained by plotting a kernel density plot. This is achieved by using `sns.distplot()` with `kind="kde"`.

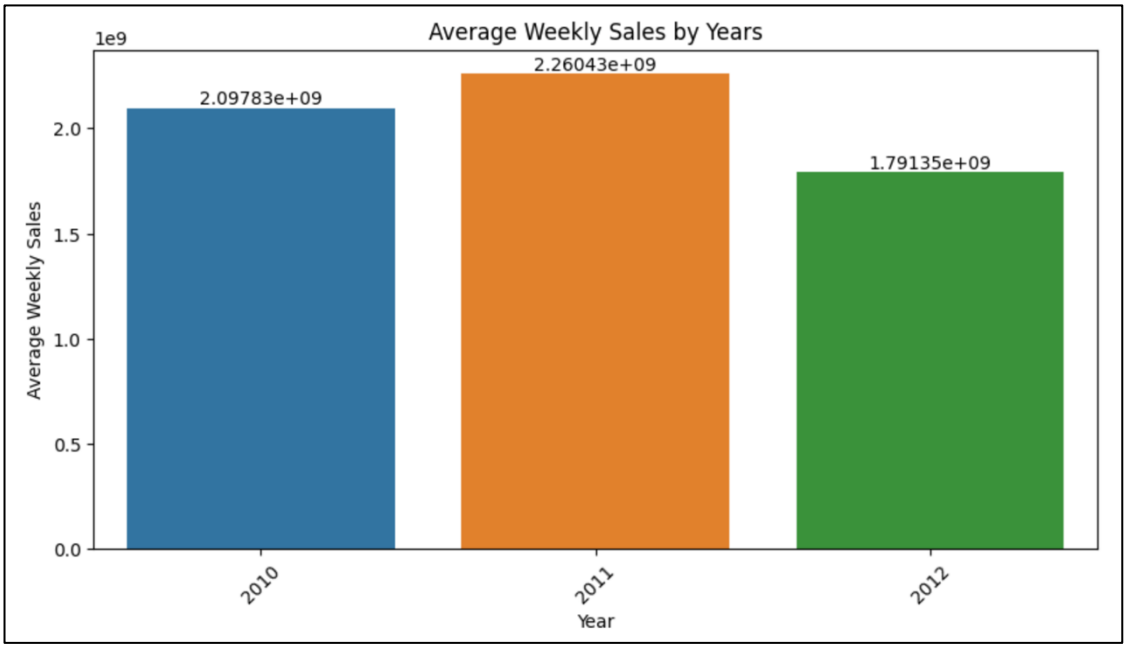
c. Average weekly sales of stores



This graph shows the number of average weekly sales vary greatly between each store. The stores with the highest number of sales are sotres 20, 4 and 14 with 2,058,998, 2,038,739 and 1,986,529 sales respectively. These top-performing stores demonstrate notable success in generating weekly sales revenue.

'Store' was grouped by 'Weekly_Sales' with an aggregation of mean. This new DataFrame was plotted using `sns.barplot()`.

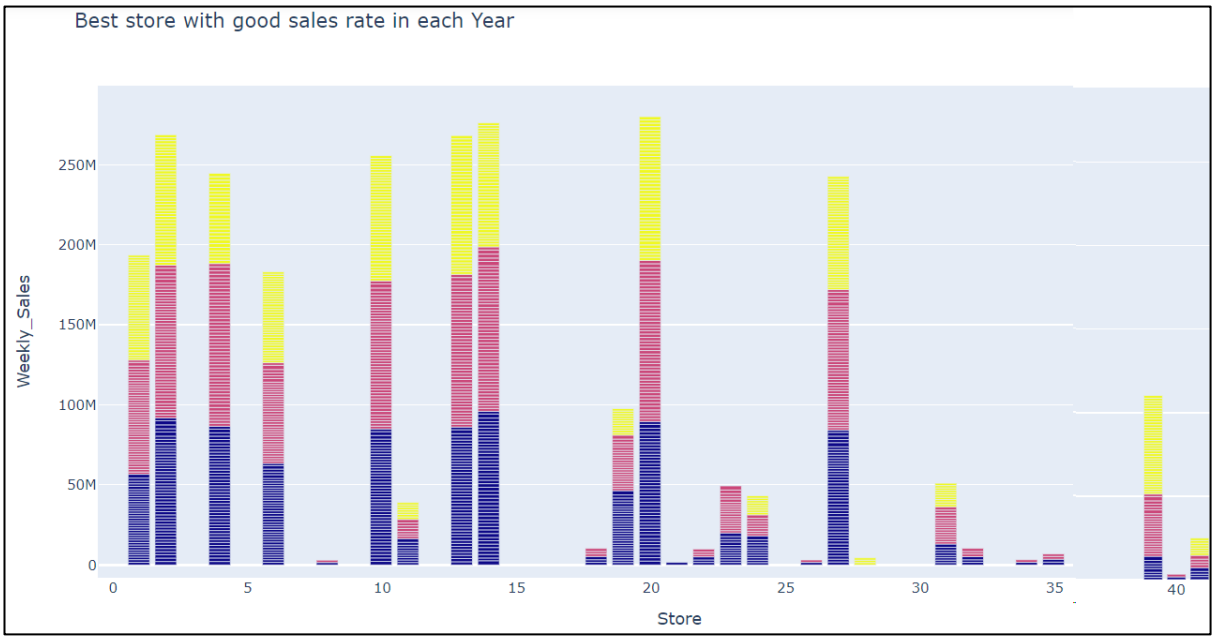
d. Total number of sales each year



The average weekly sales experienced a growth from 2,097,830,000 in 2010 to 2,260,430,000 in 2011. This represents a positive trend in sales performance, indicating an increase in revenue over two years. The revenue then decreases between 2011 and 2012 to 1,791,350,000.

This graph was plotted by creating a new DataFrame where 'Year' was grouped by 'Weekly_Sales' with a sum aggregation. This data was then plotted using `sns.barplot()`.

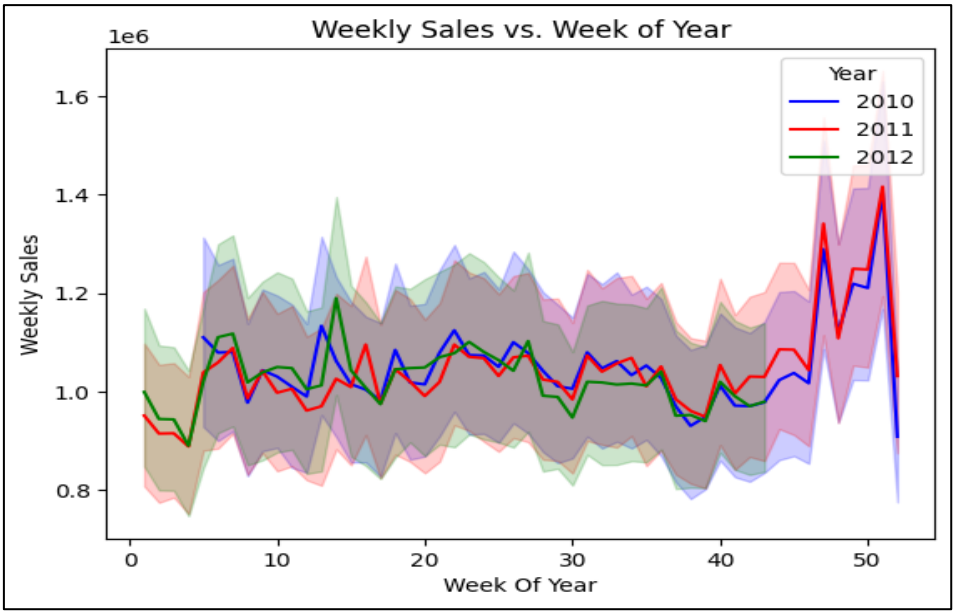
e. Best performing stores each year



Store 20 has the highest number of sales over 1,427,624 at around 2.7 million. From the graph, stores 3, 5, 7, 9, 11, 12, 15, 16, 17, 25, 29, 30, 31, 33, 36, 37, 38, 42, 43 and 44 do not produce Sales above 1,427,624. Store 2 produces high weekly sales in 1.75 million in year 2010.

`df.describe()` was used to identify how many weekly sales were made at 75%. This was found to be 1,427,624. A new DataFrame was created including stores where their 'Weekly_Sales' are above this figure. This is to ensure only stores that are high performing are included. This DataFrame is plotted in a graph using `px.bar()`.

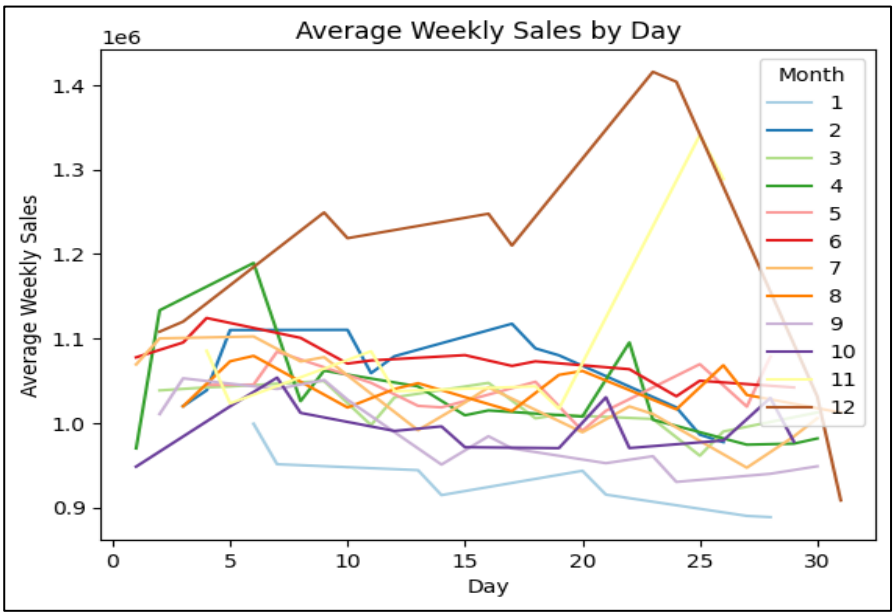
f. Weekly sales over each year



There is a notable increase in weekly sales between weeks 45 and 52 for all 3 years. This is due to the Christmas holiday period. This heightened activity suggests increased consumer spending, likely driven by holiday shopping and festive promotions.

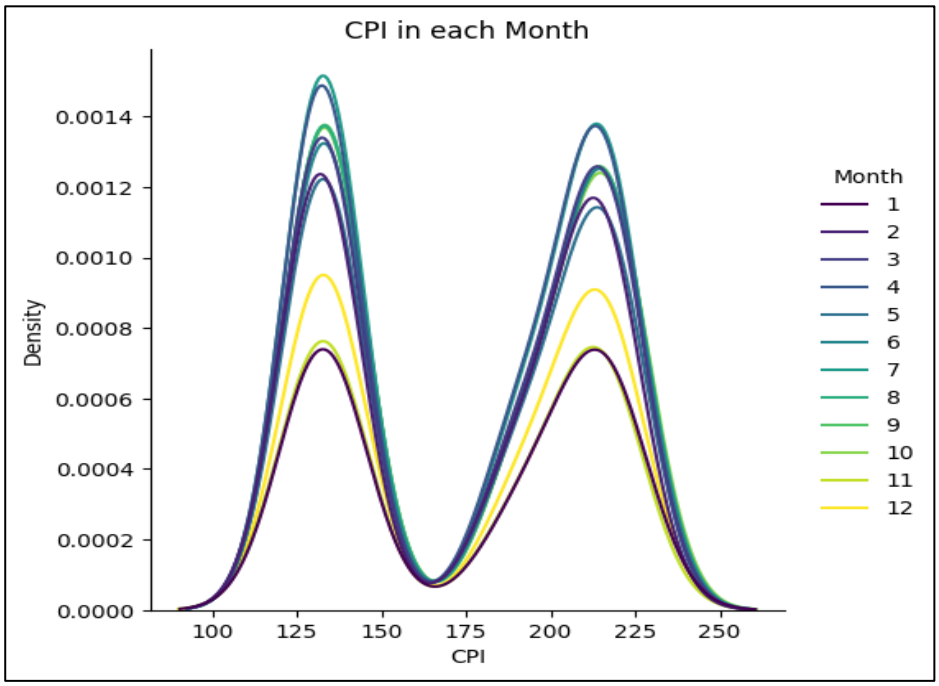
The graph was plotted using `sns.lineplot()` with Weekly_Sales on the y axis.

g. Average weekly sales per day



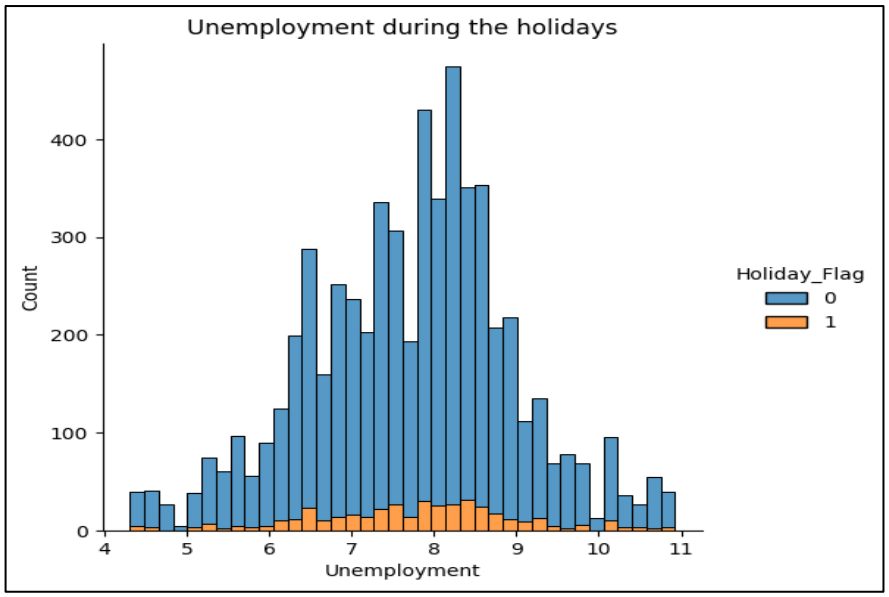
Initially, monthly sales showed a moderate trend, with a decline observed between the 10th and 20th day of each month. However, December stood out with exceptionally high sales, likely attributed to the Christmas holiday period. Store 14 recorded its peak sales of 2,685,351 million on 25th December. Store 33 has produced the lowest sales of 209,986.

h. CPI in each month



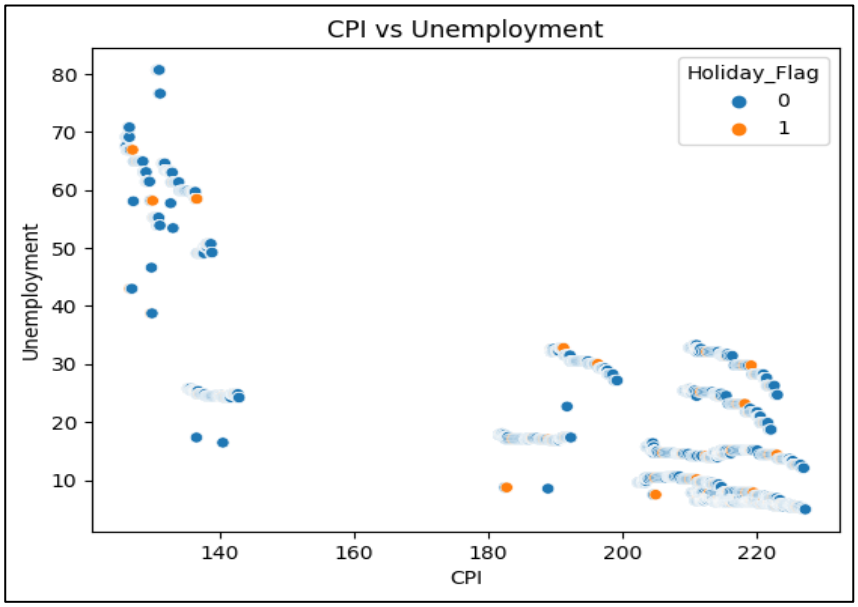
Over the course of last 12 months, the Consumer Price Index has low values in initial month, however a notable increase was observed in 7th and 8th months ,indicating a average price during that period. Subsequently ,in the 12th month ,the CPI experienced a decline has generally maintained a moderate level during the year end.

i. Unemployment during the holidays



Unemployment rate exhibit a noteworthy tendency to be lower during holidays. While the correlation between holidays and lower unemployment is evident , it's essential to recognize the multifaceted nature of the job market dynamics during these festive periods.

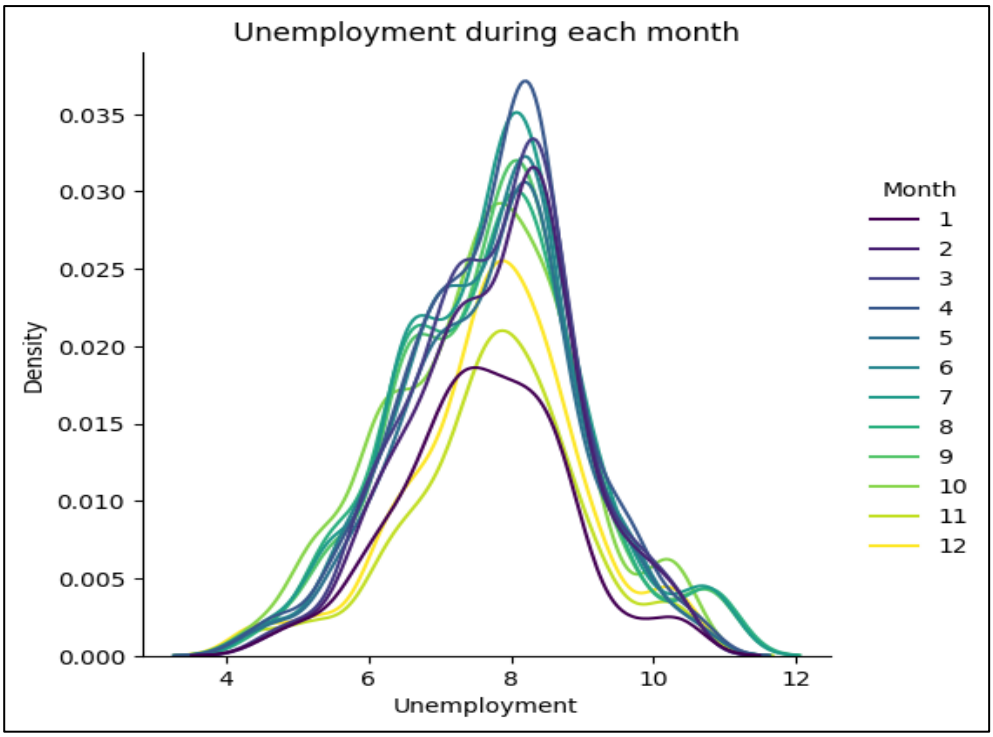
j. CPI vs unemployment



Low CPI may indicate weak consumer demand, leading to lower production and reduced hiring by businesses. This results in higher unemployment rates. Conversely, high inflation might be leading to increase hiring and lower unemployment.

This graph was plotted by grouping ‘CPI’ and ‘Holiday Flag’ with an aggregation of sum in the ‘Unemployment’ column.

k. Unemployment during each month



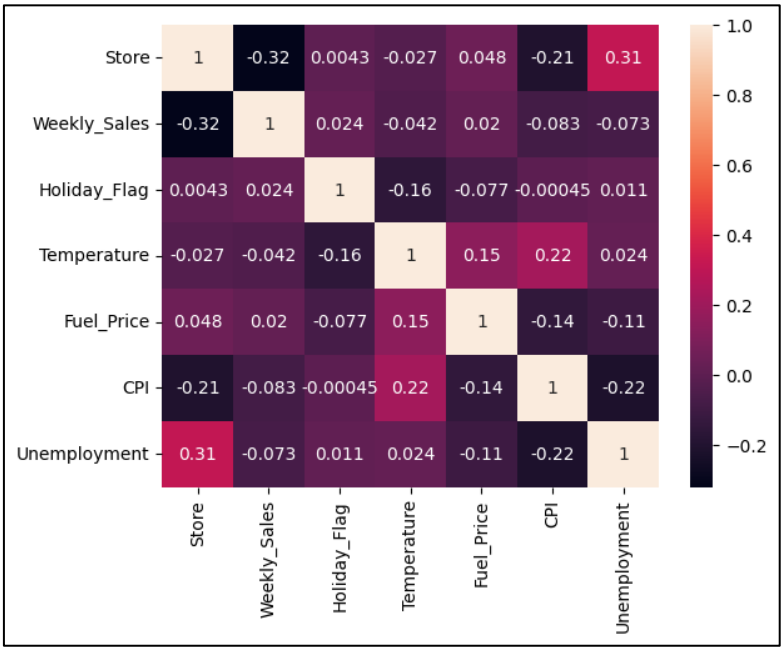
Unemployment trends show a similar pattern with all months indicating there is a weak correlation between unemployment and months. January has the lowest rate of unemployment while April has the highest.

Project Insights

a. If the weekly sales are affected by the unemployment rate, which stores are suffering the most?

i. Heatmap

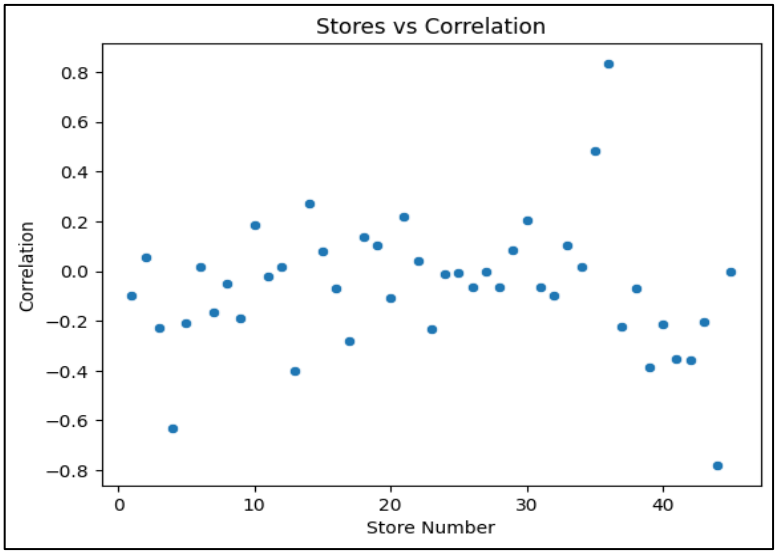
The first step taken was to create a heat map using `sns.heatmap()` to show the correlation between the different columns.



The correlation between Weekly Sales and Unemployment Rate is -0.073. Since this is a low negative number, there is an overall very weak negative correlation between Weekly Sales and Unemployment Rate.

ii. Scatterplot of correlations

However, we can investigate this further by looking at the individual correlations between Weekly Sales and Unemployment for each of the 45 stores. This has been done using a for loop which stores the correlation between these 2 variables in the list `overallCorr`. This is converted to a DataFrame and plotted using `sns.scatterplot()`.



For the majority of stores there is a weak correlation between Weekly Sales and Unemployment but we can see a few stores have strong positive and strong negative correlations.

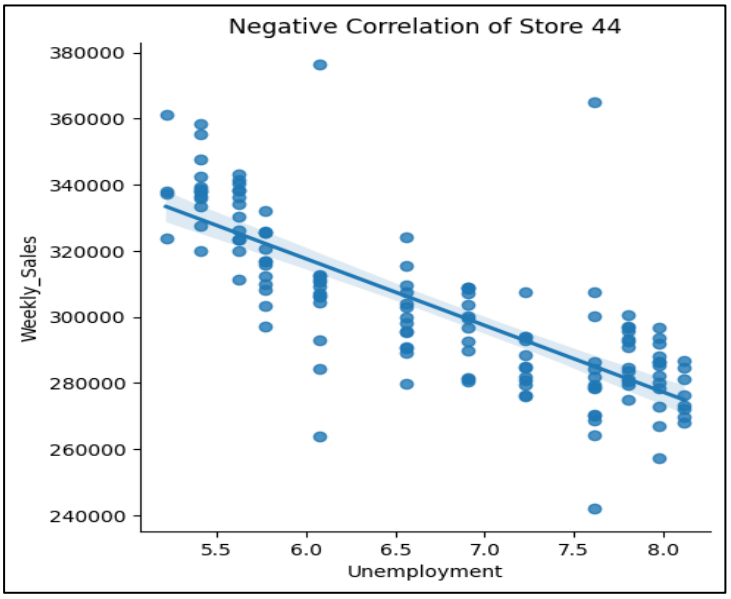
As the unemployment rate increase we would expect to see the number of Weekly Sales decrease. We will therefore first visualise the Weekly Sales and Unemployment variables for the store with the strongest negative correlation.

iii. Strongest negative correlation

Store Number	Correlation
44	-0.780076
4	-0.633422
13	-0.400254
39	-0.384681
42	-0.356355

By sorting the previous DataFrame created, we receive the output above. From this analysis we can see that store number 44 has a -0.78 correlation between Weekly Sales and Unemployment. From this we can infer that as Unemployment increases, Weekly Sales tends to decrease. There is a strong negative correlation between the two variables.

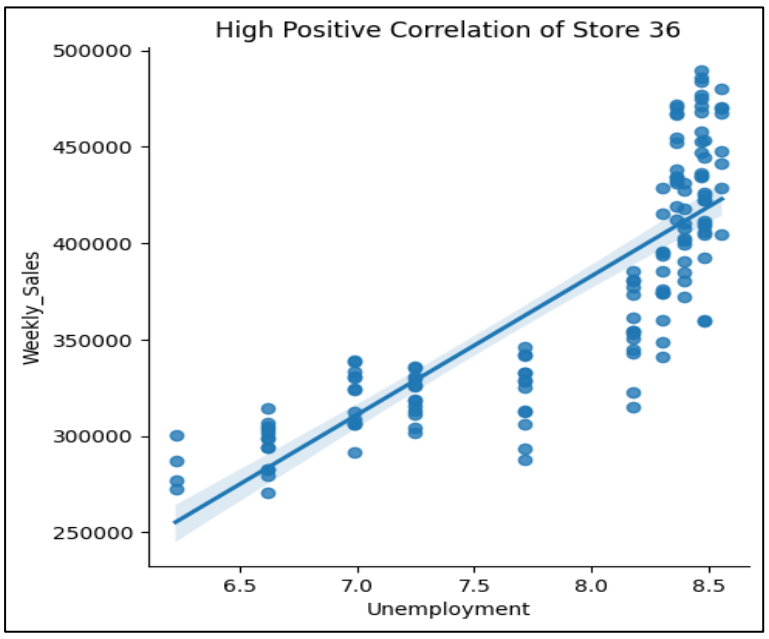
To further analyse this relationship a graph has been plotted below:



iv. Strongest positive correlation

The top 5 stores with the strongest positive correlation are and visualised by sns.lmplot

Store Number	Correlation
30	0.201862
21	0.218367
14	0.269510
35	0.483865
36	0.833734

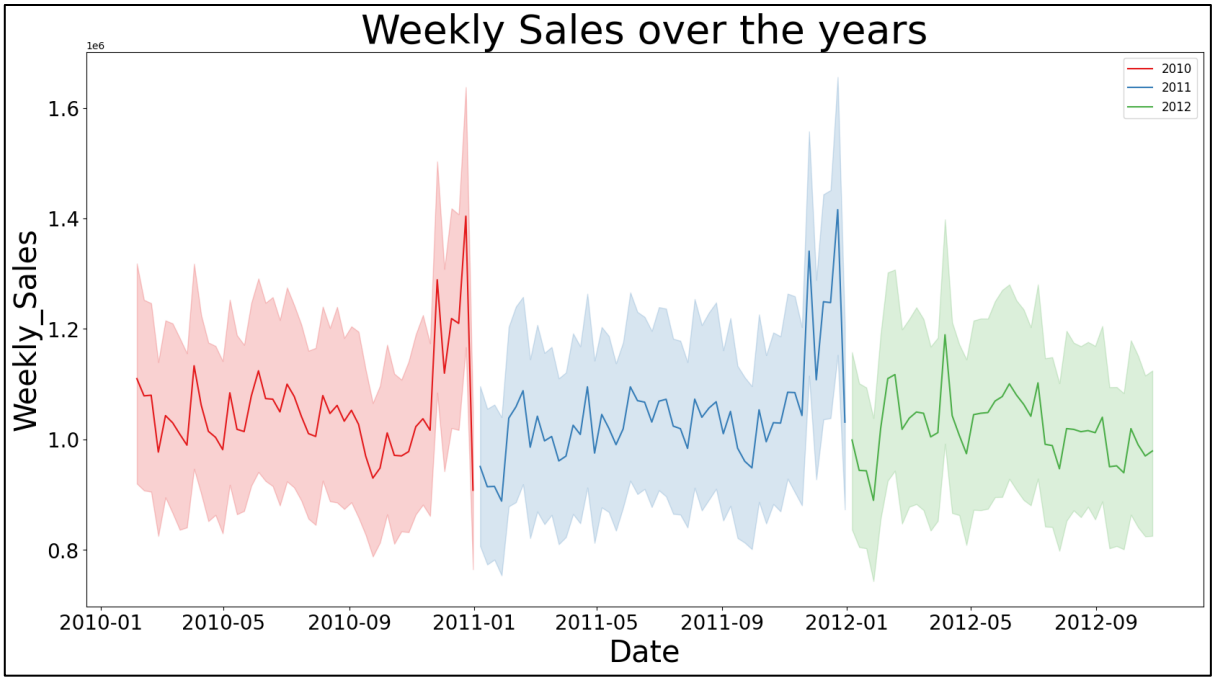


v. Conclusion

From the analysis carried out we can conclude that the relationship between Weekly Sales and Unemployment varies on a store to store basis. The majority of stores have no correlation between Weekly Sales and Unemployment as seen from the -0.073 value in the correlation heatmap.

Store 44 has the strongest negative correlation (-0.78) between Weekly Sales and Unemployment while Store 35 has the strongest positive correlation (0.83).

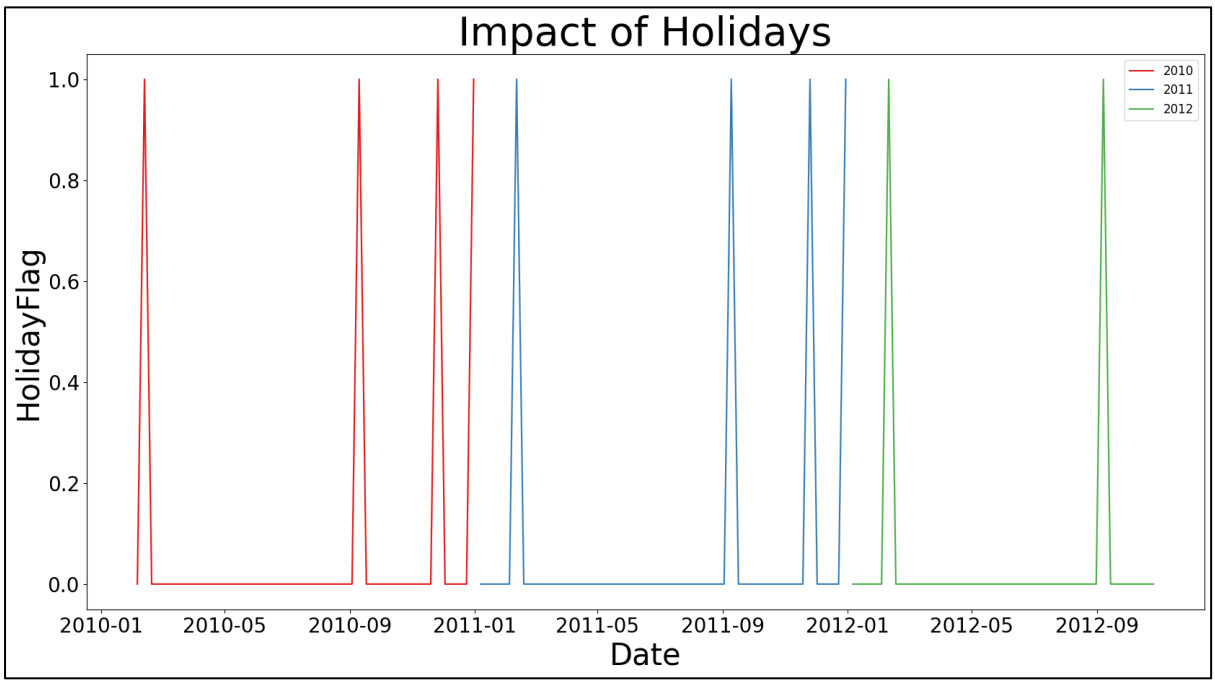
b. If the weekly sales show a seasonal trend, when and what could be the reason?



From the above lineplot we can see in years 2010 and 2011 there are increases in Weekly Sales from November to December. This suggests there is a seasonal trend where Weekly Sales increase during the winter season.

This peak is not present in 2012 because no data is provided after October 2012.

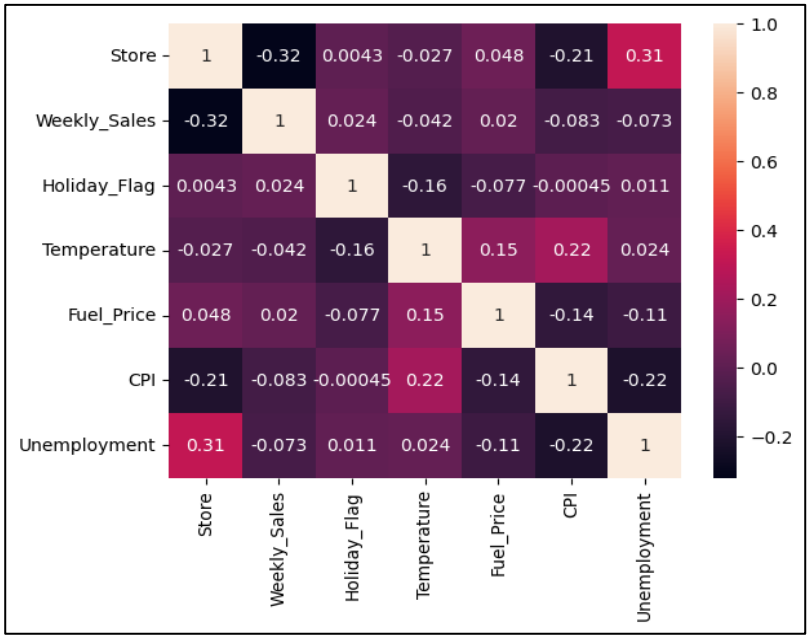
The reason for this happening is investigated below:



From the above analysis we can see the peak in Weekly Sales coincide with holiday dates throughout the year. We can therefore conclude the seasonal trend shown by Weekly Sales happens due to the holiday periods throughout the year.

c. Does temperature affect the weekly sales in any manner?

i. Heatmap



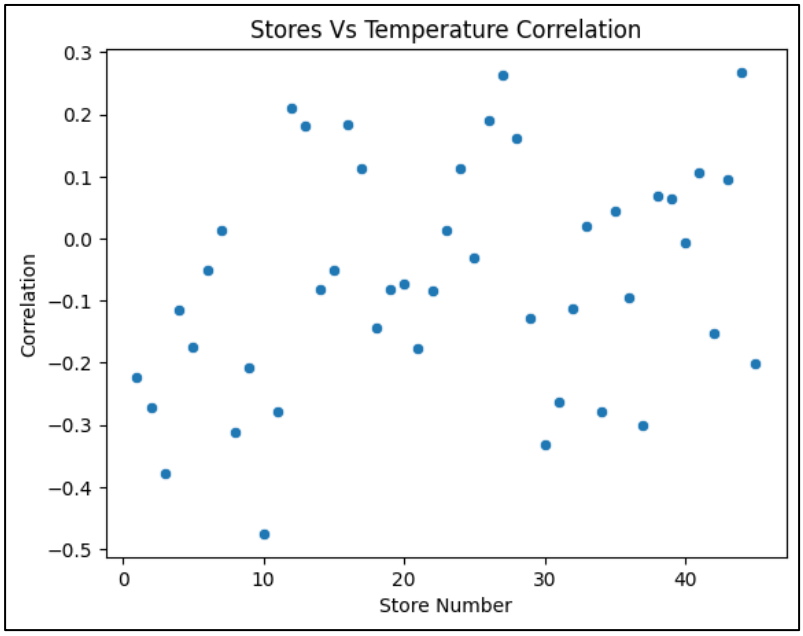
The heatmap above shows a correlation of -0.042 between Temperature and Weekly Sales. This suggests a weak negative correlation: a change in Temperature does not tend to impact the change in Weekly Sales.

We can investigate this further by analysing the correlation for each individual store using the same method as previously for the correlation between Weekly Sales and Unemployment.

ii. Correlation

By using for loop we can store overall correlation with StoreNumber and Temperature. These values are stored in overallCorr2 and converted into DataFrame.

Store Number	Correlation
1	-0.222701
2	-0.272249
3	-0.377524
4	-0.114243
5	-0.175517

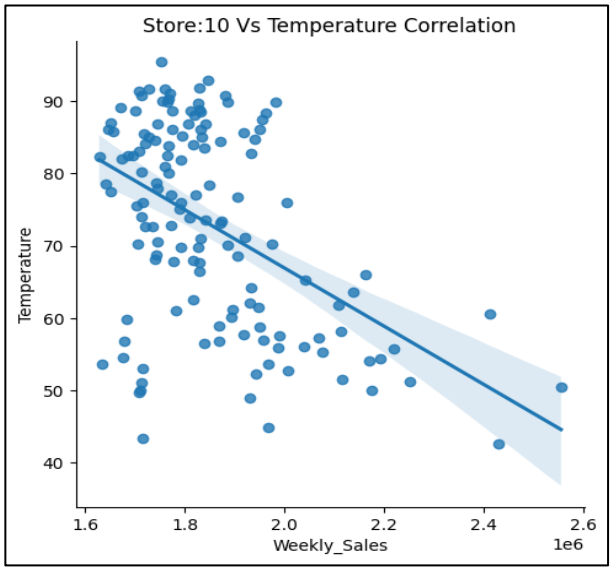


The above scatterplot shows all stores have weak correlations between Temperature and Weekly Sales.

iii. <u>Strongest negative correlation</u>

Store Number	Correlation
10	-0.475243
3	-0.377524
30	-0.330816
8	-0.312324
37	-0.300493

The strongest negative correlation is found using the same method as previously. Here store 10 has the strongest negative correlation and the scatter plot for it can be seen below which plots 'Temperature' against 'Weekly_Sales'.

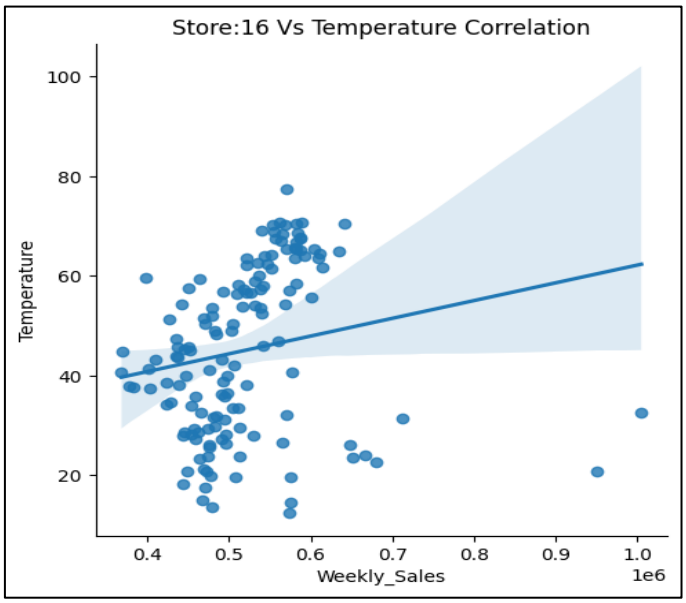


iv. Strongest positive correlation

The strongest positive correlation is found from the tail of the previous DataFrame which showed the correlation values for all stores. The DataFrame below shows store number 16 has the strongest positive correlation.

Store Number	Correlation
16	0.182948
26	0.190021
12	0.210494
27	0.262541
44	0.267822

'Temperature' is plotted against 'Weekly_Sales' for store 16 using `sns.lmplot()`.



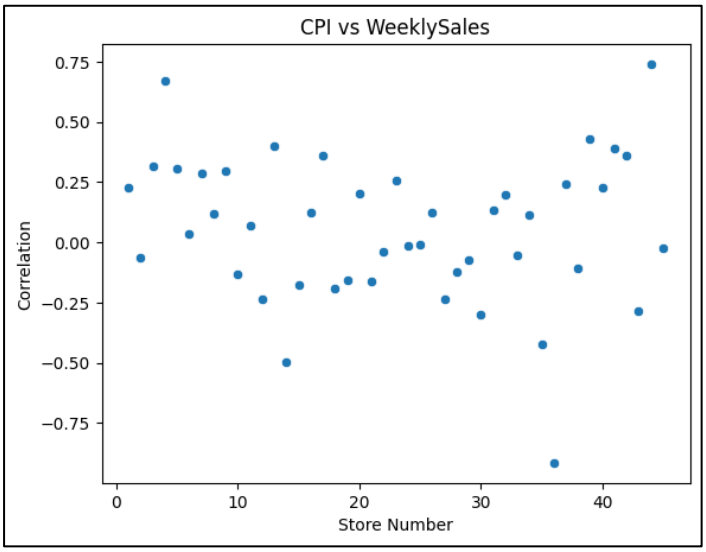
v. Conclusion

The strongest negative correlation of -0.475 implies there is a weak negative correlation between Weekly Sales and Temperature for Store 44.

However, since the majority of the stores have very weak correlations, we can therefore conclude that Temperature does not tend to affect the Weekly Sales.

d. How is the CPI affecting the weekly sales of various stores?

By using the same method, the correlation values related with 'CPI' and 'Weekly_Sales' is stored in a list called 'Corr3'. Then it is converted into DataFrame . These correlation values are then visualised in a scatterplot using `sns.scatterplot()`.



By sorting the correlated values, the most positive and weakly strongly correlated stores can be found.

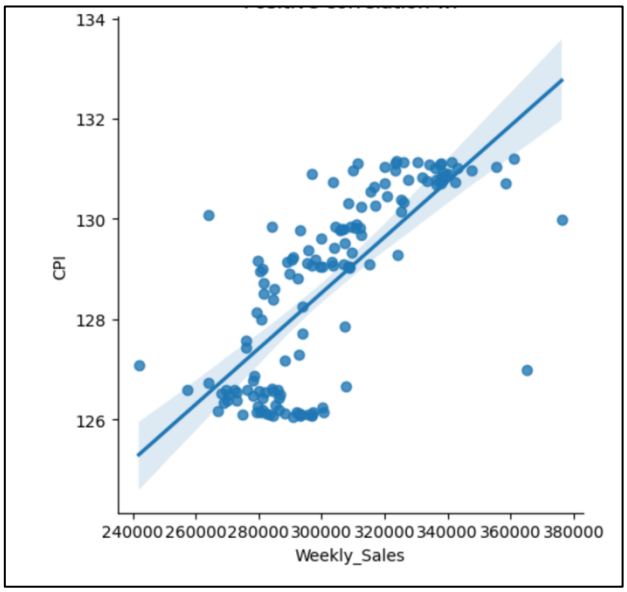
Store Number	Correlation
36	-0.915095
14	-0.498495
35	-0.424107
30	-0.298188
43	-0.285686

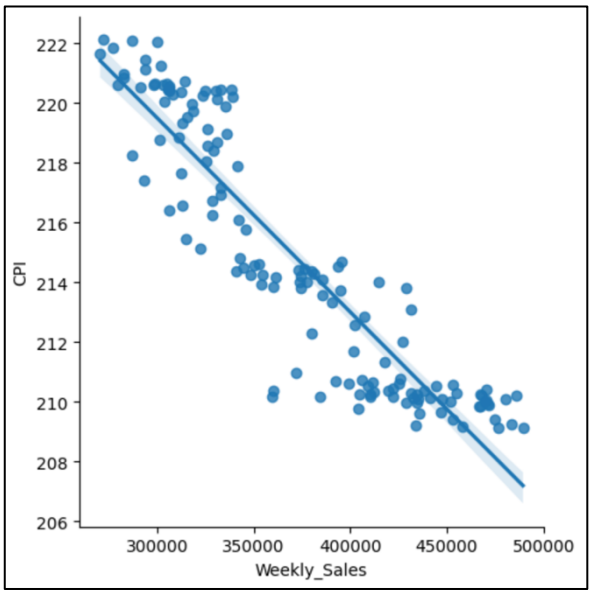
Store Number	Correlation
41	0.392293
13	0.401445
39	0.428043
4	0.669028
44	0.740150

The above analysis shows that CPI impacts Weekly Sales in different stores to different degrees.

Some stores have a strong positive correlation which implies as CPI increases, Weekly Sales tend to increase. Other stores have a strong negative correlation which implies as CPI increases, Weekly Sales tend to decrease.

We can therefore conclude that CPI impacts Weekly Sales differently for different stores. Below are the visualisations for the Store 44 with the strongest positive correlation and Store 36 with the strongest negative correlations between CPI and Weekly Sales





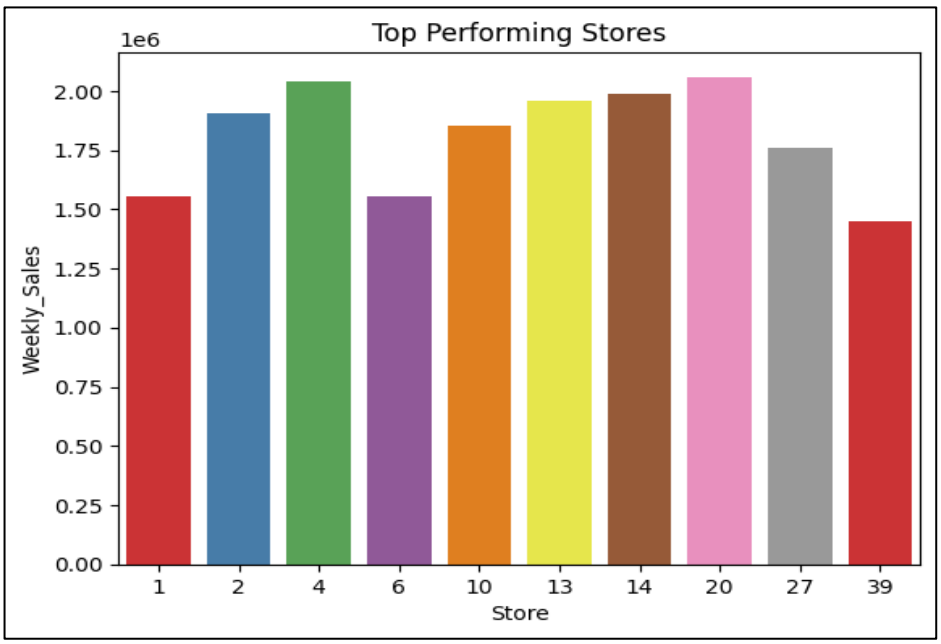
The two visualisations above show the two stores with the strongest positive and negative correlations between CPI and Weekly Sales.

From this analysis we can therefore conclude that CPI tends to impact the Weekly Sales differently on a store to store basis. Store 44 has the strongest positive correlation (0.74) and store 36 has the strongest negative correlation (-0.91).

e. <u>Top performing stores according to the historical data</u>

Based on the data provided, the most optimal way to judge performance of a store is to compare the total Weekly Sales figures for each store. The store with the highest number of Weekly Sales is the top performing store.

Grouping the store with mean of weekly sales is stored in Data Frame 'totalweeklysales'. This DataFrame is then sorted by 'Weekly_Sales' where condition ascending is given as false. This moves the highest sales production stores to the top of the DataFrame. This is visualised in a bar plot using `sns.barplot()`.

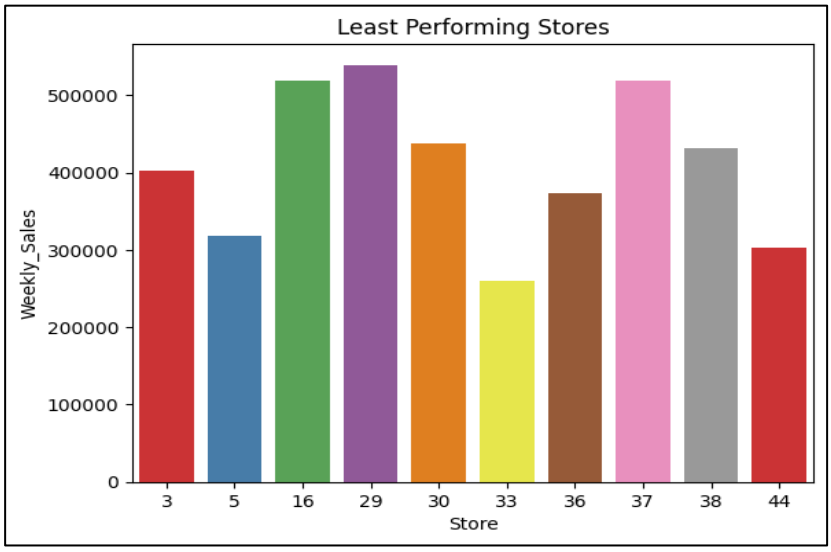


From this analysis, we have found the highest 10 performing stores and visualised this through a bar plot.

The highest performing store overall is store 20 with a total of around 280,023,735 Weekly Sales.

f. Worst performing stores according to the historical data

To answer this question we can use the dataset total WeeklySales we previously created. Since it is sorted in descending order, we can find the worst 10 performing stores by accessing the tail of the DataFrame. The tail of the data frame gives the worst performing stores. By using `sns.barplot()`, the worst performing stored are visualised.



From this analysis, we have found the lowest 10 performing stores and visualised this through a bar plot. We did this using the same methods to find the highest 10 performing stores.

The lowest performing store overall is store 38 with a total of around 7,347,379 Weekly Sales.

g. Difference between highest and lowest performing stores

The difference in the number of Weekly Sales between the highest performing Store 20 and the lowest performing Store 38 is 1,799,136.360 Weekly Sales

Use predictive modelling techniques to forecast the sales for each store for the next 12 weeks

There are a few models we can use to perform sales forecast analysis. As we have been provided with Time Series data for Weekly Sales of each store, we can perform Time Series Analysis to derive insights from the given data and create future predictions.

The two forecasting models we can use here are ARIMA and Exponential Smoothing. Exponential Smoothing is used when the given data doesn't have any clear trend or seasonality.

We will first perform linear regression to see if this model is suitable in answering the problem statement.

1. Feature Scaling

Before data can be used to train the machine learning model, a final step of pre-processing is required. Feature scaling is an important step of the model building process. It involves the normalisation and standardisation of the data which makes sure all the data is on the same scale and improves the performance of the model (Analytics Vidhya, 2024). Since the features are all in the same scale, they will now contribute equally in the machine learning model.

The 2 steps in the feature scaling process are explained below:

a. Splitting the dataset

The dataset is first divided into dependent and independent variables. The dependent variable in this case is 'annual_income' as we are trying to predict whether an individual makes more than or less than $50,000. This is stored in `x` while the independent variables are stored within `y`.

This data is then split into the test and train datasets using the `train_test_split()` function with a test size of 20%.

b. Feature scaling

Feature scaling is the next step to perform. To carry out this step, the `StandardScalar()` object is imported. `sc.fit_transform()` is applied to the x_train dataset and `sc.transform()` is applied to the x_test dataset. These datasets are now ready to be used to train the machine learning models.

2. Machine Learning Models

The main step is the building of the machine learning models, fitting and training these models and then using these models to predict the desired outputs.

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. Use predictive modelling techniques to forecast the sales for each store for the next 12 weeks.

3. Choosing the Algorithm for the Project

Since this is a prediction task, different prediction models will be used and their accuracies compared to determined which model produces the most accurate output to address the problem statement.

The 3 chosen models are explained below and their outputs analysed:

a. Linear Regression
   Training and testing the dataset using linear regression produces an R2 score of 15%. This implies that Linear Regression is not an effective method in answering the problem statement. It can therefore be disregarded.

b. ARIMA
   Autoregressive Integrated Moving Average, or ARIMA, is a model that is used to understand how a dataset behaves and create future predictions based on current behaviours (Investopedia, 2024).

   i. What are the 3 aspects of ARIMA models?

   - **Autoregression (AR)**: This component assumes the linearly dependant relationship between a given point in a time series with its past values.

   - **Integrated (I)**: This stage involves differencing to make the timeseries stationary. This means ensuring the mean and variance of the series do not change over time.

   - **Moving Average (MA)**: This is the relationship between an observation and the residual error. The error is calculated from the moving average model.

ii. What are the 6 aspects in training ARIMA models?

1. Visualize the Time Series Data

2. Ensure the Time Series Data is stationary. If it is not:
    i. Use differencing.
    ii. If the data follows seasonality, then use 12 points differencing. This means scaling down the data by taking differences of not immediate period, rather the next seasonal period (January to January/Q1 to Q1).

3. Plot the autocorrelation (ACF) and Partial autocorrelation (PACF) charts.

4. Select the lags (p, d, q) for the AR, differencing and MA model as per the ACF and PCF charts.

5. Construct the ARIMA models or seasonal AIRMA models based on the stationary data.

6. Use the model to make future predictions.

4. Motivation and Reasons For Choosing the Algorithm

From the analysis carried out above, **ARIMA** is the chosen algorithm because it provides the most accurate and robust prediction.

5. Assumptions

Defining assumptions is an important step before building machine learning models. It enables you to understand the performance and interpretation of the model and define parameters to ensure the predictions are accurate and robust (TowardsDataScience, 2018).

1. The data should be stationary. This means the variance and mean of the data does not change over time. If the data is not stationary, differencing should be used to achieve standardisation.
2. There is a linear relationship between the lagged values and the observations
3. The residual errors produced by the model should be uncorrelated with each other and not produce any discernable patterns.
4. The data is free from outliers which would skew results.

6. Model Evaluation and Techniques

There are 45 stores provided in the dataset. Producing forecasting models for all these stores is unfeasible. Therefore, preliminary analysis is carried out to determine which stores should be analysed and have their weekly sales forecasted.

a. How are the stores chosen?

Three methods are used to choose the stores for the ARIMA model building.

i. 1st method

The first store is using a random selection method. This produced store 1.

ii. 2nd method

For the second method, a list called 'salesCorr' is initialised. To this list, the store number along with the correlation of the weekly sales of each store to the date is appended. This is then converted into a DataFrame.

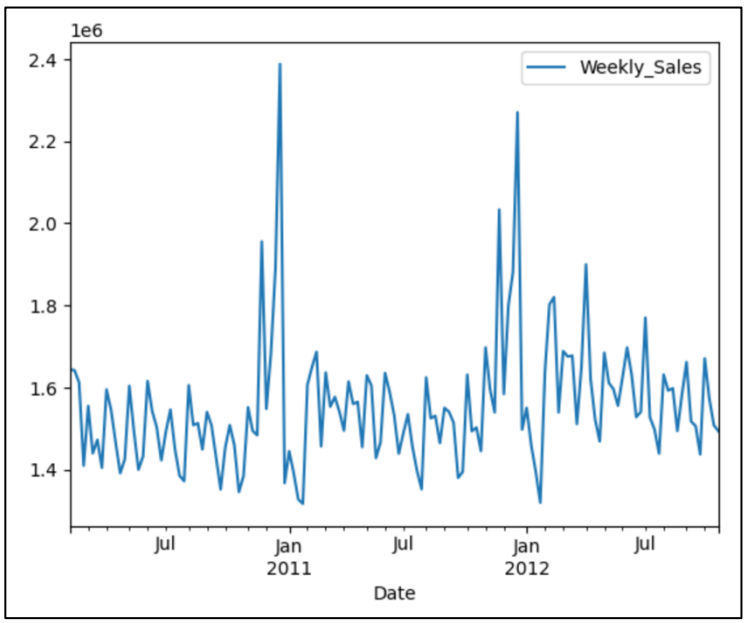
The DataFrame created shows the extent to which the weekly sales are influenced by time. A strong positive correlation implies there is a noticeable increase in sales across the 3 years while a strong negative correlation implies there is a noticeable decrease. The store with the strongest correlation is **store 36** and **store 6** has the weakest correlation.

iii. 3nd method

Along with these 3 stores, stores 20 and 33 were chosen as they were the strongest and weakest performing stores from the project insights previously carried out.

b. Store 1 example

A DataFrame called 'Store' is created which contains the weekly sales data for store 1. This DataFrame is visualised using `store.plot()` which can be seen below. The forecasting model will produce predictions following the end of the graph.



STEP 1: Check if the data is stationary

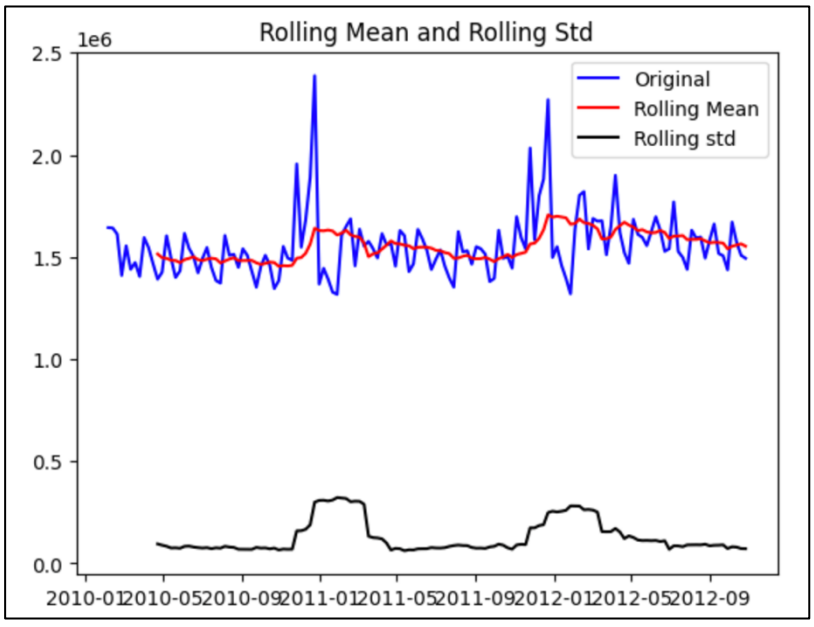
This step is carried out by using the `adfuller` class from the `statsmodels.tsa.stattools` library. The results of this can be seen below:

```
ADF Statistic: -5.294329877929837
p-value: 5.625696763153646e-06
n_lags: 4
Store 6: The series is stationary
--------------
ADF Statistic: 0.21363188452557658
p-value: 0.9730158839006191
n_lags: 8
Store 36: The series is not stationary
--------------
ADF Statistic: -1.8334316831502826
p-value: 0.36399460154866586
n_lags: 12
Store 20: The series is not stationary
--------------
ADF Statistic: -4.523395189776184
p-value: 0.00017836040761491051
n_lags: 5
Store 38: The series is stationary
--------------
```

The data for stores 6 and 38 are stationary while stores 1, 20 and 36 are not stationary. As mentioned previously, when the Time Series data is not stationary, differencing steps should be carried out which can be seen below.

STEP 2: Use differencing

Initially, the mean and standard deviation are calculated for store 1 and plotted. The results of this can be seen below.

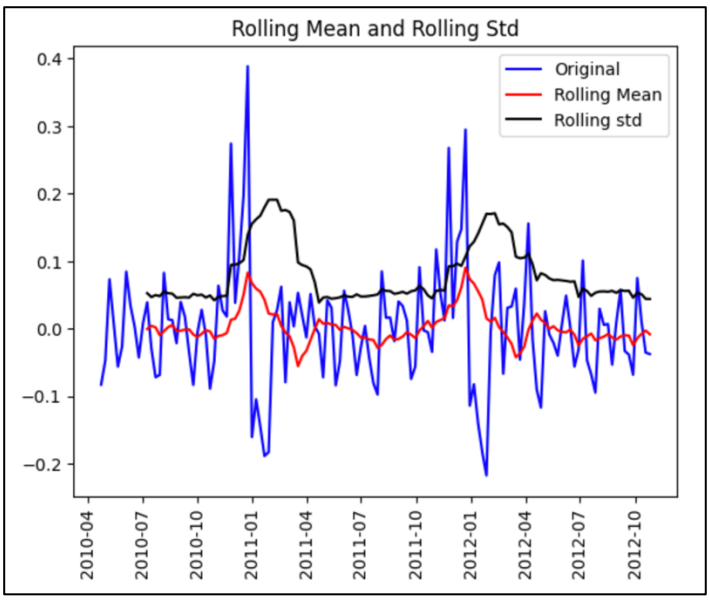


This graph shows the mean and standard deviation are not constant. Therefore a log transformation step is required.

STEP 3: Log transformation

`np.log()` is used to take the natural logarithm of the observed values in the time series data. This process is used to stabilize variance, linearize relationships, normalize distributions, and interpret percentage changes.

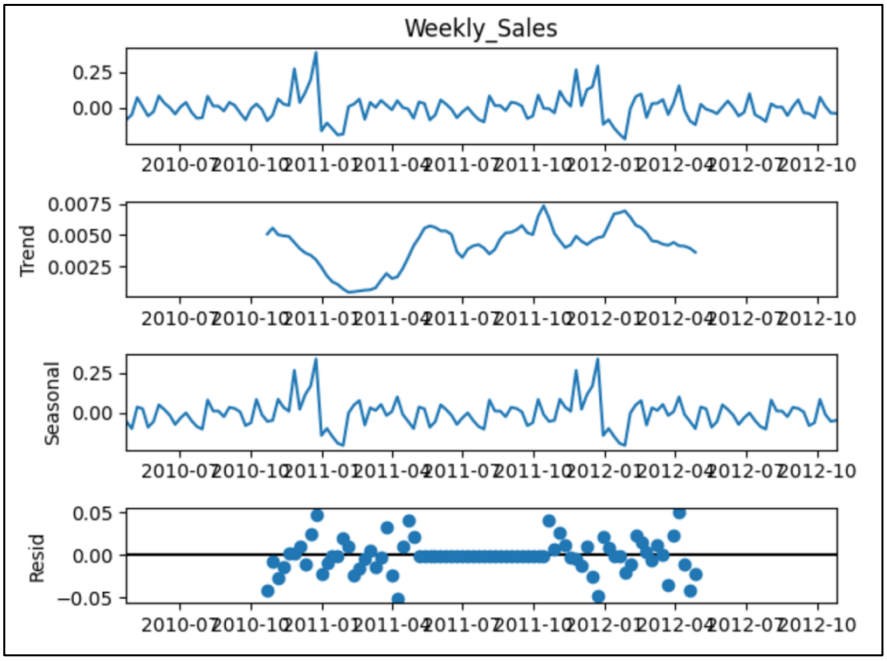
This step is repeated twice which produces the output that can be seen below.



STEP 4: Seasonal decomposition

Seasonal decomposition is performed next to identify any seasonality in the data. If you have Time Series data you can break it down into several components. Seasonal decomposition will break down the Seasonal, Trend and Residual parts of the data separately.

The Seasonal decompose removes the noise in the data and plots the noise in the residuals graph. We can see the Seasonal graph has a similar pattern to our original data.



STEP 5: ADF Test

Visually it looks like our data has now become stationary. Visual confirmation is ambiguous so we need to perform a test to confirm the data is stationary. This is done through the ADF or KPSS tests.

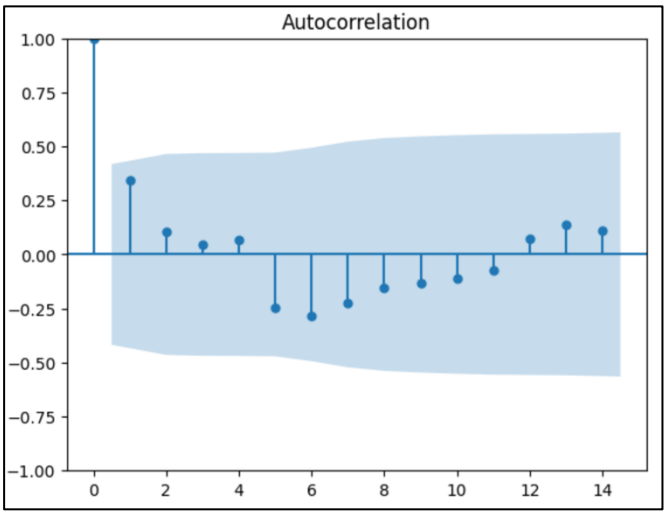
This test confirms the data is now stationary. Therefore, the timeseries step can be carried out.

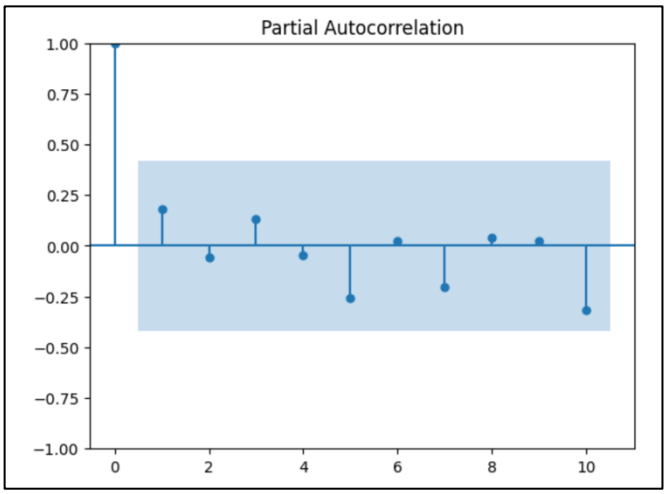
STEP 6: Model transformation

PD and q values are required for ARIMA. We get these values from acf (q) and pacf (p) plots. The d value stands for differencing. If the data is not stationary we perform log transformations. The number of log or any other transformations we use is the d value.

acf (q): We find the q value based on where the first shut off happens in the acf plot. We select the value before the shut off.

pacf (p): We use the same method to find the p value.



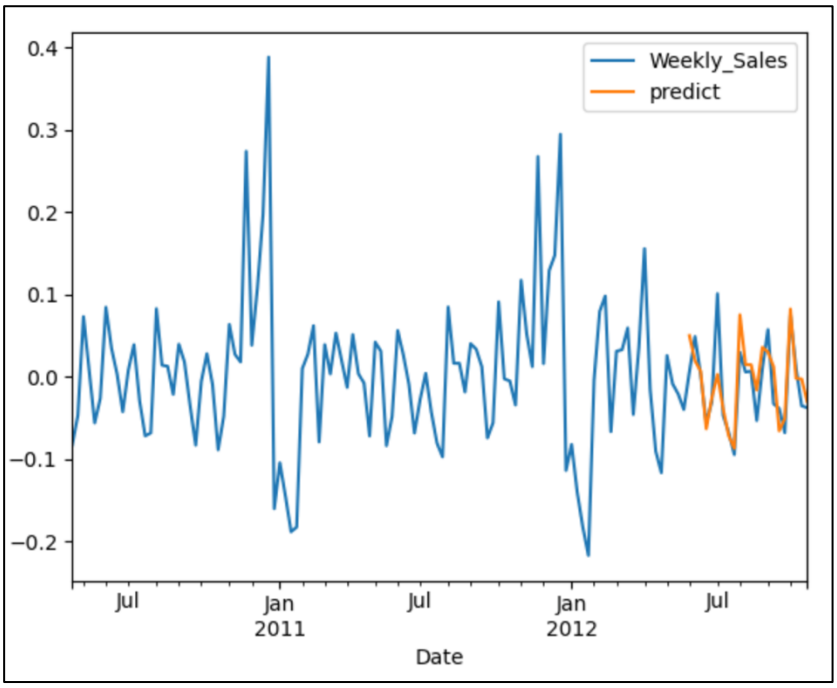


Based on the acf and pacf plots, our q value is 4 and our p value is 1. We can now build the ARIMA model.

STEP 6: Creating the ARIMA model

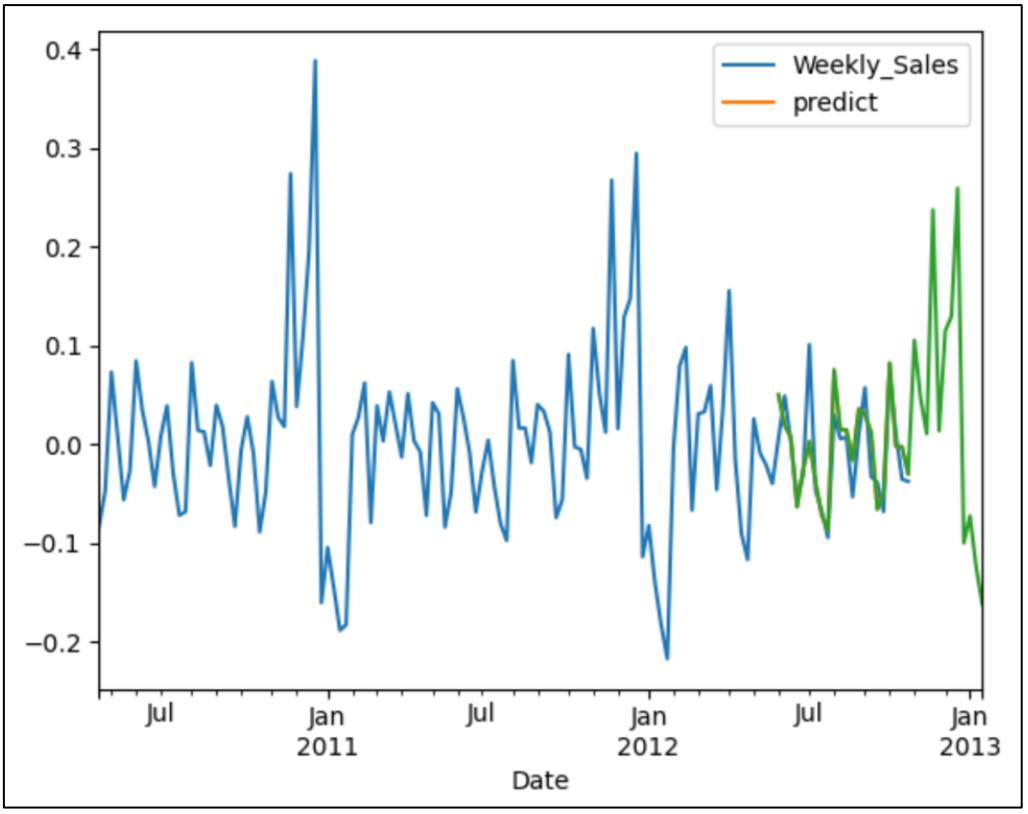
Using `ARIMA()` the ARIMA model is initialised. `order(1,0,4)` is inputted with the p, d and q values. Because the data provided is seasonal, an ARIMA model will not work well. We are now saying although there may be a seasonal trend, there may be other external factors that cause the data to change predictions. Therefore, we will use a SARIMAX model instead. `SARIMAX()` is used with the same p, d and q values, along with 'seasonal_order' where 52 denotes the number of weeks in a year.

Training and plotting this model produces the lineplot that can be seen below.



The blue line represents the current weekly sales data. The orange line shows the values predicted by the model. The next step is to extrapolate this prediction so data for the next 12 weeks can be forecasted.

STEP 7: Forecasting sales for the next 12 weeks



7. Inferences

- The ARIMA model performs well and accurately forecasts the weekly sales for the next 12 weeks.
- Linear regression is an ineffective model to use in addressing the problem statement.

Conclusion

a. Summary

The aim of this project was to address the problem statement and forecast weekly sales predictions for 45 different Walmart stores. Data about the stores was provided with various features such as CPI, unemployment and temperature.

Linear regression was initially used but the R2 score of 15% suggested this model is unsuitable for use in addressing the problem statement. Following this ARIMA and SARIMAX models were created and the data was trained and tested. The weekly sales predictions were accurately forecasted and visualised in a graph.

b. Future possibilities of the project

If this project were to be carried out again, it would be beneficial to use data captured for the Walmart stores over a longer period of time. This would lead to more accurate forecast predictions. Furthermore, an overall better set of results could be achieved by using a combination of machine learning and deep learning models.