Walmart Sale Analysis For Retail Industry with Machine Learning

Project Documentation

Performance & Final Submission Phase

Date	6 November 2023
Team ID	Team – 592731
Project Name	Walmart Sale Analysis For Retail Industry with Machine Learning
Team Members	D Akshara S Chetan Manish Dereddy Venkata Yogitha

Walmart Sale Analysis For the Retail Industry with Machine Learning

1. Introduction

1.1 Project Overview

Estimating future revenues is the technique of sales forecasting. Businesses may anticipate both short- and long-term performance and make well-informed business decisions with the help of accurate sales projections. Businesses can use historical sales data, industry-wide comparisons, and economic trends as a basis for their forecasts. This time, the business is Walmart. The well-known retail company Walmart runs a network of hypermarkets. Walmart has made data available by aggregating 45 locations' worth of store details and monthly sales data. Walmart holds a number of annual promotional discount events. The four biggest markdowns occur before major holidays: Super Bowl, Labour Day, Thanksgiving, and Christmas. In comparison to weeks without holidays, the evaluation gives the weeks that have these holidays a five-fold higher weight. The information is delivered every week. We need to determine how the store's sales are affected by the holidays. The Christmas, Thanksgiving, Super Bowl, and Labour Day holidays are all featured. Algorithms like XgBoost, Random Forest, and ARIMA will be used. We will use these methods to train and test the data. There will also be an IBM deployment and Flask integration.

Project Summary:

The "Walmart Sales Analysis for Retail Industry using Machine Learning Algorithms" project aims to leverage the power of machine learning and data analytics to gain valuable insights from Walmart's sales data. This analysis will help Walmart and other retail businesses make data-driven decisions, optimize operations, and enhance profitability.

Project Objectives:

a. Analyze historical sales data to identify trends and patterns.

b. Predict future sales and demand for products.

c. Optimize inventory management and supply chain operations.

d. Segment customers for targeted marketing and personalized recommendations.

e. Evaluate the impact of external factors (e.g., holidays, economic conditions) on sales.

f. Develop a user-friendly dashboard for data visualization and reporting.

By the end of this project, you will:

● Know fundamental concepts and techniques used for machine learning.

● Gain a broad understanding about data.

● Have knowledge on pre-processing the data/transformation techniques on outlier and some visualization concepts.

Project Scope:

The project will focus on the following key aspects:

a. Data Collection: Gathering historical sales data from Walmart's various stores and regions.

b. Data Preprocessing: Cleaning and preparing the data for analysis, handling missing values, and outliers.

c. Exploratory Data Analysis (EDA): Exploring data to identify trends, seasonality, and correlations.

d. Feature Engineering: Creating relevant features for predictive modeling.

e. Machine Learning Algorithms: Implementing various machine learning models for sales forecasting, customer segmentation, and impact analysis.

f. Model Evaluation: Assessing model performance using metrics like RMSE, MAE, and accuracy.

g. Visualization and Reporting: Creating interactive dashboards to visualize insights and generate reports.

h. Deployment: Implementing the best-performing models in a real-time or batch prediction system.

Data Sources:

The project will utilize the following data sources:

a. Walmart's internal sales data.

b. Economic indicators (e.g., inflation rates, GDP) for impact analysis.

c. Weather data for weather-dependent product sales analysis.

Machine Learning Algorithms:

Various machine learning algorithms will be explored, including but not limited to:

a. Time Series Forecasting: ARIMA, Prophet, LSTM.

b. Regression Models: Linear Regression, Random Forest, XGBoost.

c. Clustering: K-Means, DBSCAN for customer segmentation.

d. Anomaly Detection: Isolation Forest for outlier detection.

Tools and Technologies:

a. Programming Languages: Python for data analysis and machine learning.

b. Libraries and Frameworks: Pandas, NumPy, scikit-learn, TensorFlow, Keras.

c. Data Visualization: Matplotlib, Seaborn, Plotly, Tableau.

d. Dashboard Creation: Power BI, Tableau, or custom web-based dashboard solutions.

e. Version Control: Git.

Deliverables:

The project will deliver the following outputs:

a. Predictive models for sales forecasting.

b. Customer segmentation analysis.

c. Impact analysis of external factors on sales.

d. Interactive data visualization dashboards.

e. Final project report detailing methodologies and findings.

Project Team:

The project team may consist of data scientists, machine learning engineers, data analysts, and domain experts in retail operations.

Timeline:

The project is expected to span several months, with specific milestones and deadlines for each phase.

Benefits:

This project will help Walmart and the retail industry in general to make data-driven decisions, improve sales forecasting accuracy, optimize operations, and enhance customer satisfaction. It will ultimately contribute to increased profitability and competitiveness in the market

1.2 Project Purpose

In this competitive retail industry, the ability to adapt swiftly to changing market dynamics and customer preferences is a cornerstone of success. Walmart, a global retail giant, is embarking on a groundbreaking endeavor to revolutionize its retail operations through advanced data analytics and machine learning. This project represents a pivotal shift towards leveraging cutting-edge technology to gain a competitive edge. By harnessing the power of machine learning, Walmart seeks to analyze historical sales data, predict future sales trends, optimize inventory management, implement dynamic pricing strategies, and segment its customer base for tailored marketing efforts.

The retail landscape has witnessed a profound transformation with the advent of e-commerce, and Walmart is poised to harness the wealth of data generated by its vast network of stores and online platforms. Through this initiative, Walmart aims to tap into the wealth of information at its disposal, turning data into actionable insights. Every transaction, every click, and every interaction with customers generates valuable data, and this project aims to make the most of it. By doing so, Walmart aspires to enhance the efficiency of its operations, increase profitability, and provide an enhanced, personalized shopping experience for its customers.

This endeavor is not only about adopting state-of-the-art technology but also about embracing a data-driven mindset, where decisions are underpinned by comprehensive data analysis. It is a transformative journey that promises to shape the future of retail at one of the world's largest and most influential retailers. In a world where data is the new currency, Walmart's commitment to leveraging machine learning in retail sales analysis underscores the company's determination to lead the industry into a more efficient, customer-centric, and data-driven future.

2. Literature Survey

2.1 Existing Problem

Walmart, like many other retailers, faces several challenges in its sales analysis efforts within the retail industry using machine learning. These challenges include the need to ensure high data quality and integration from diverse sources, the management of volatile demand influenced by various factors, such as seasonality and promotions, and the complexities of maintaining optimal inventory levels while avoiding overstock and stockouts. Additionally, accurately segmenting customers and personalizing marketing strategies can be intricate, and achieving the highest model accuracy in sales forecasting requires ongoing refinement. Data privacy and security are paramount concerns, especially when handling sensitive customer data, and scaling machine learning solutions across Walmart's extensive network of stores is no small feat. Implementing change management and ensuring model interpretability, along with continuous maintenance and monitoring, are also key hurdles in this machine learning-driven retail analysis endeavor. Addressing these challenges is crucial for Walmart to make the most of the potential benefits of machine learning in retail operations.

2.2 References

Machine Learning Concepts:

o Supervised learning:

https://www.javatpoint.com/supervised-machine-learning

o Unsupervised learning:

https://www.javatpoint.com/unsupervised-machine-learning

o Regression and classification

https://www.geeksforgeeks.org/regression-classification-supervised-machine-learning/

o Random forest:

https://www.javatpoint.com/machine-learning-random-forest-algorithm

o Xgboost:

https://www.geeksforgeeks.org/xgboost/

o Evaluation metrics:

https://www.geeksforgeeks.org/metrics-for-machine-learning-model/

https://towardsdatascience.com/what-are-rmse-and-mae-e405ce230383

2.3 Problem Statement Definition

Walmart, one of the world's largest retail chains, seeks to enhance its sales analysis and forecasting capabilities to improve its retail operations. The company operates numerous stores across diverse locations, offering a wide range of products. To optimize inventory management, pricing strategies, and customer experiences, Walmart aims to leverage machine learning techniques for sales analysis and prediction.

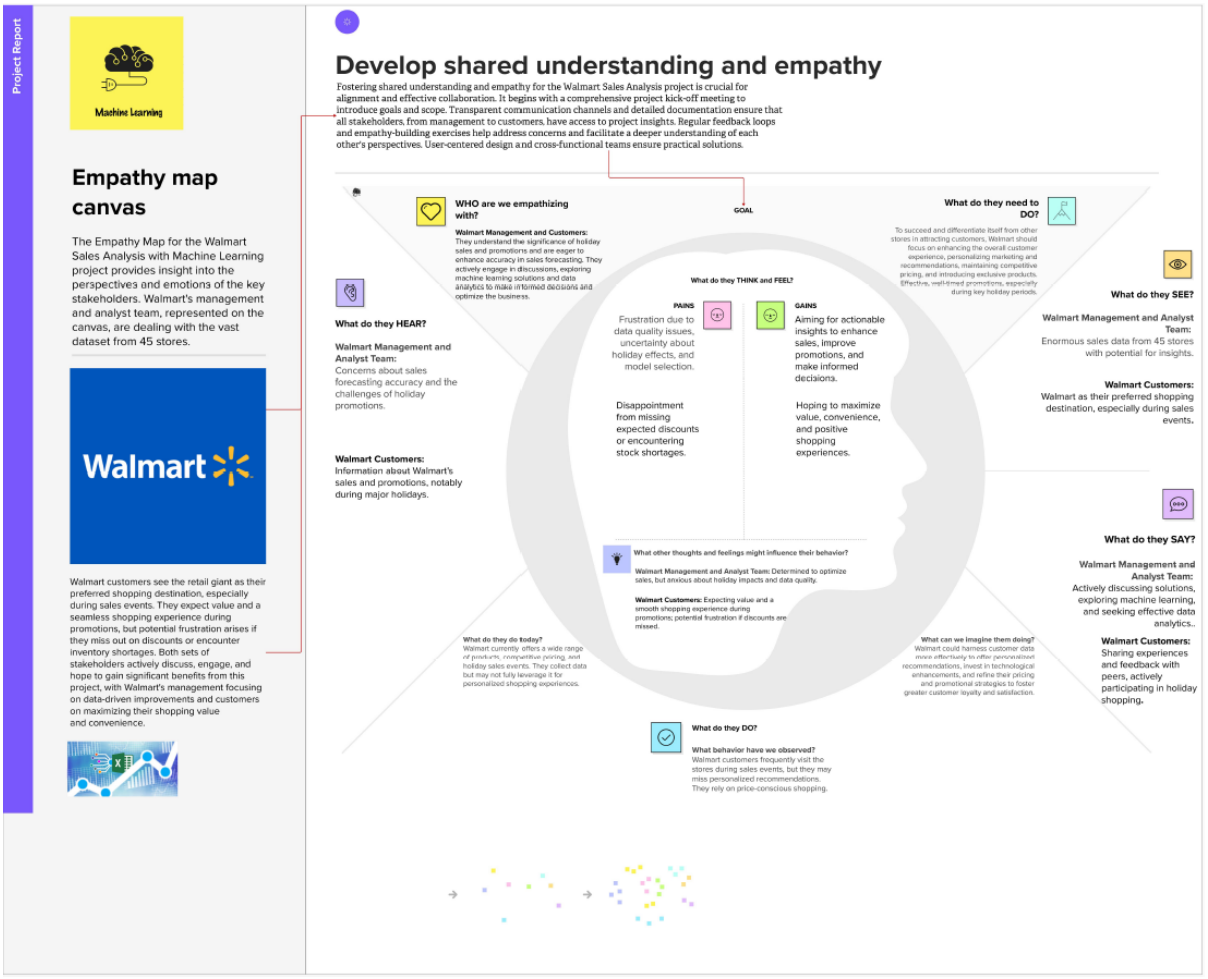
In this project, Walmart is looking to leverage machine learning to analyze and forecast sales, optimizing its retail operations. The primary goals include developing accurate sales forecasting models, improving inventory management to minimize overstock and stockouts, analyzing demand patterns for various products and regions, implementing dynamic pricing strategies, and segmenting the customer base for tailored marketing efforts. By utilizing data-driven insights, Walmart seeks to enhance operational efficiency, profitability, and customer satisfaction within its vast retail business operations.

Upon successful completion of the project, Walmart expects the following outcomes:

1. Improved sales forecasting accuracy, reducing excess inventory and stockouts.
2. Enhanced inventory management practices to optimize store operations and reduce carrying costs.
3. Data-driven insights to fine-tune pricing strategies, increasing profitability.
4. A deeper understanding of customer segments to tailor marketing efforts and improve the customer experience.

3.Ideation & Proposed Solution

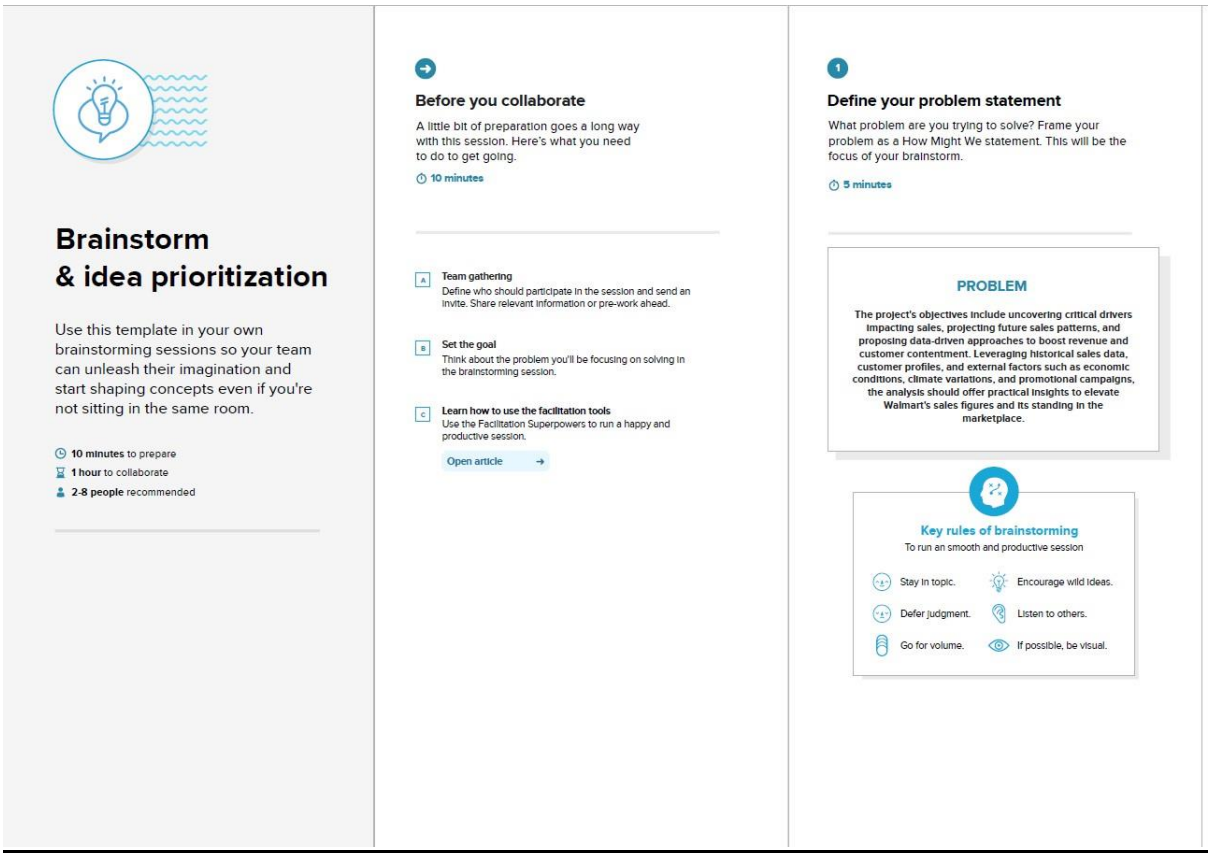
3.1 Empathy Map Canvas



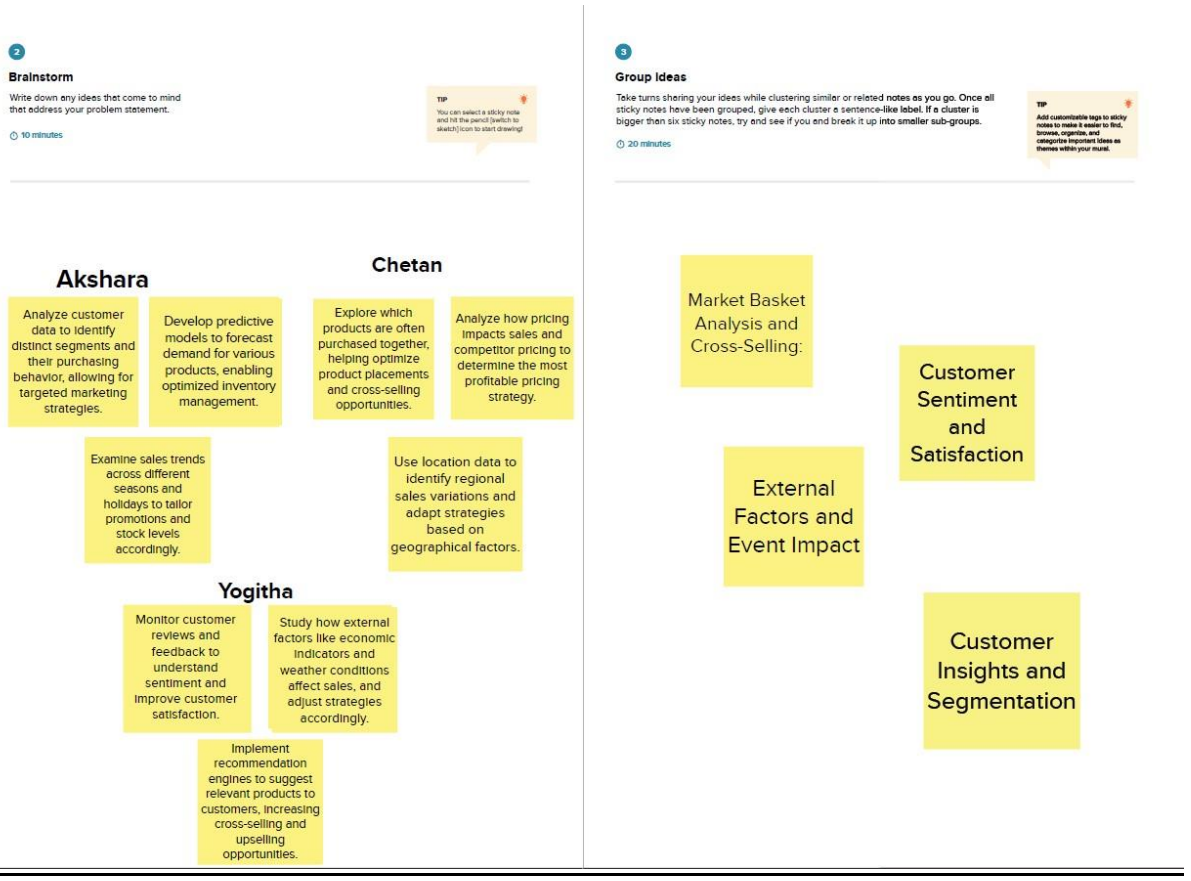
Link: https://app.mural.co/t/chethanmanish1175/m/chethanmanish1175/1697616019043/67ae80d4ea524f4f7d2be9e3166489ff9d49f053?sender=u2c79b514440b53cfc65e7241

3.2 Ideation & Brainstorming

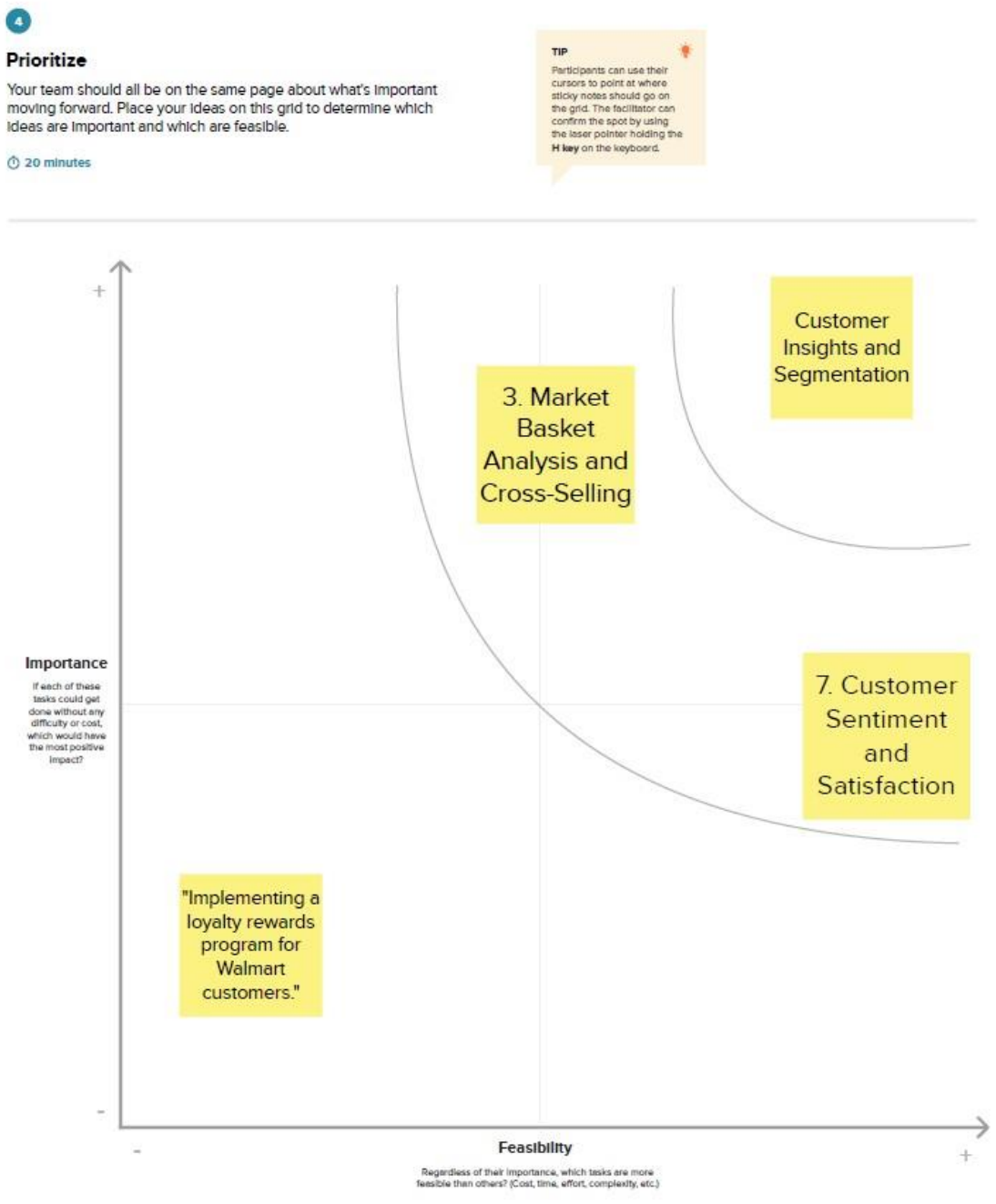
Step 1: Team Gathering, Collaboration and Select the Problem Statement



Step 2: Brainstorm, Idea Listing and Grouping



Step 3: Idea Prioritization:



Link:https://app.mural.co/t/walmartsa7268/m/walmartsa7268/1697696018199/7ceeb260be62e2db80b4014d4a1d6aa28a3537e5?sender=ued10059affdb6fd25b915391

4. Requirement Analysis

4.1 Functional Requirements

Functional requirements are essential to ensure that the system meets the needs and objectives of the project. Here are some key functional requirements for such a system:

1 Data Ingestion and Integration:
The system should be capable of ingesting data from various sources, including sales records, inventory data, customer data, and external data feeds.
It should integrate and transform data into a common format suitable for analysis.

2 Data Cleaning and Preprocessing:
Perform data cleaning to handle missing values, outliers, and inconsistencies.
Normalize and standardize data to ensure its quality and consistency.

3 Sales Forecasting:
Develop and implement machine learning models to forecast sales for different product categories, store locations, and time periods (daily, weekly, monthly).
Evaluate and select appropriate forecasting models, such as time series analysis or regression.Implement algorithms for inventory optimization, considering demand forecasts, lead times, and safety stock levels.
Alert for low-stock or overstock situations and suggest reordering strategies.

4 Demand Analysis:
Analyze historical sales data to identify demand patterns and trends for various products and geographical regions.
Discover seasonality and variations in demand.

5 Pricing Optimization:
Develop pricing optimization algorithms that dynamically adjust product prices based on demand, competition, and other relevant factors.
Monitor and evaluate the effectiveness of pricing strategies.

6 Customer Segmentation:
Segment customers based on purchasing behavior, demographics, and other relevant attributes.
Customize marketing and sales strategies for different customer segments.

7 Data Visualization and Reporting:
Provide interactive dashboards and reports for users to visualize sales data, forecasts, and insights.
Utilize data visualization tools like Tableau or Power BI.

8 User Access Control:
Define user roles and access controls to ensure data security and restrict access to sensitive information.

9 Performance Monitoring and Optimization:
Continuously monitor the system's performance and optimize its algorithms a

10 Model Training and Retraining:

Periodically retrain machine learning models with new data to maintain accuracy and relevance.

4.2 Non-Functional Requirement

Non-functional requirements are equally important for this project , as they define the system's performance, security, and usability characteristics. Here are some non-functional requirements:

1. Performance:
   Response Time: The system should provide quick responses to user queries and data processing, ensuring that it meets performance expectations.
   Scalability: The system should be able to handle a growing volume of data and users without a significant drop in performance.
   Throughput: It should support concurrent users and data processing to prevent bottlenecks during peak times.
2. Reliability:
   The system should have high availability, with minimal downtime for maintenance or updates.
   It should be fault-tolerant, capable of recovering gracefully from failures without data loss or service interruption.
3. Security:
   Ensure data security and privacy compliance, especially when handling sensitive customer information.
   Implement user authentication and authorization mechanisms to control access to the system.
   Data encryption in transit and at rest to protect sensitive information.
4. Usability:
   The user interface should be intuitive, user-friendly, and accessible to users with different levels of technical expertise.
   Provide multi-platform compatibility to allow users to access the system from various devices.
5. Scalability:
   The system should be designed to handle an increasing amount of data and users as Walmart's operations expand.
6. Maintainability:
   The code and infrastructure should be well-documented and organized to facilitate system maintenance, updates, and troubleshooting.

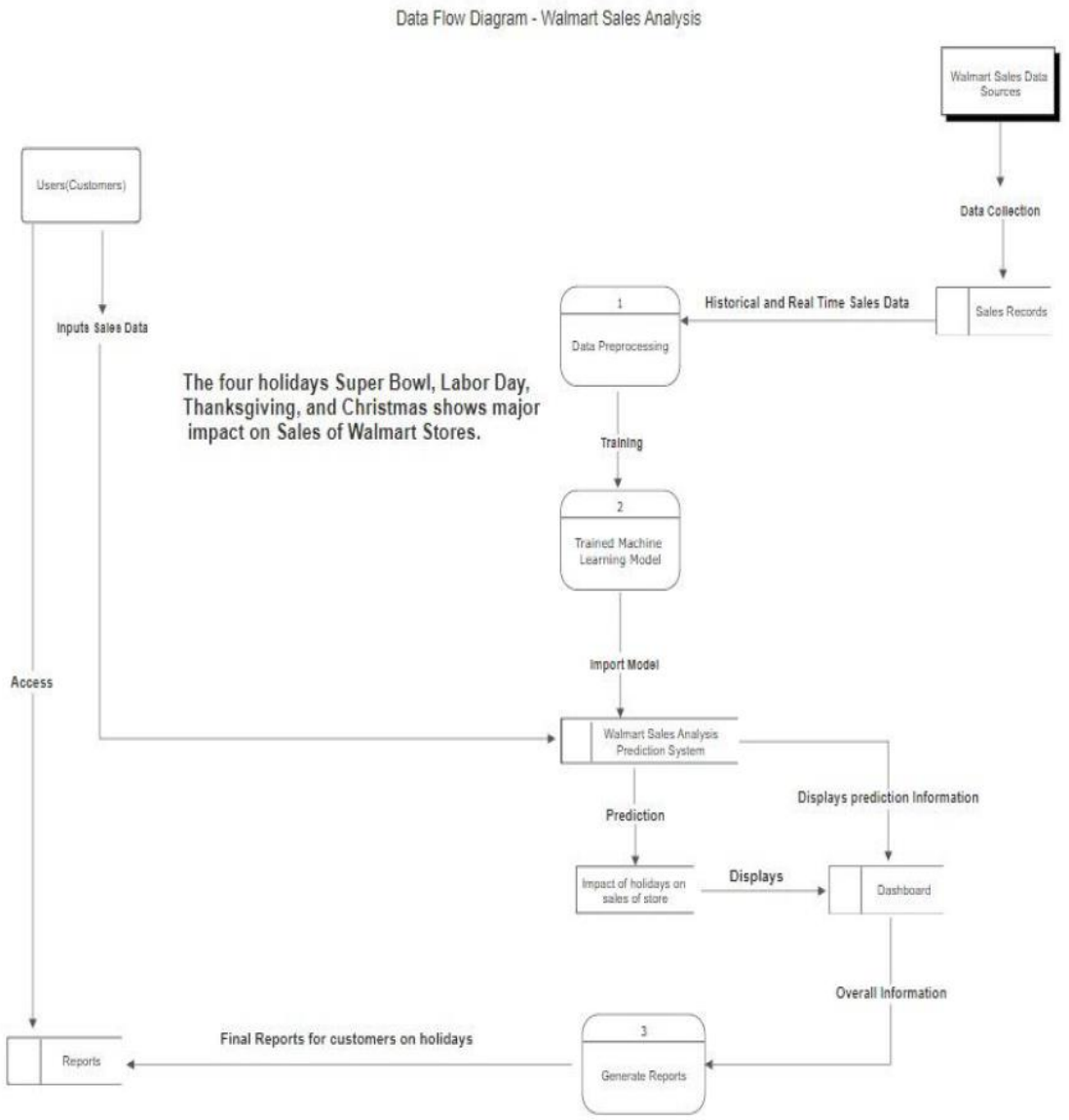
Implement version control and a change management process to track and manage updates and modifications.

7. Data Quality and Integrity:
   Ensure the data quality and integrity throughout the data processing pipeline to prevent issues arising from inaccurate or incomplete data.

5 Project Design

5.1 Data Flow Diagram & User Stories

Data Flow Diagram:

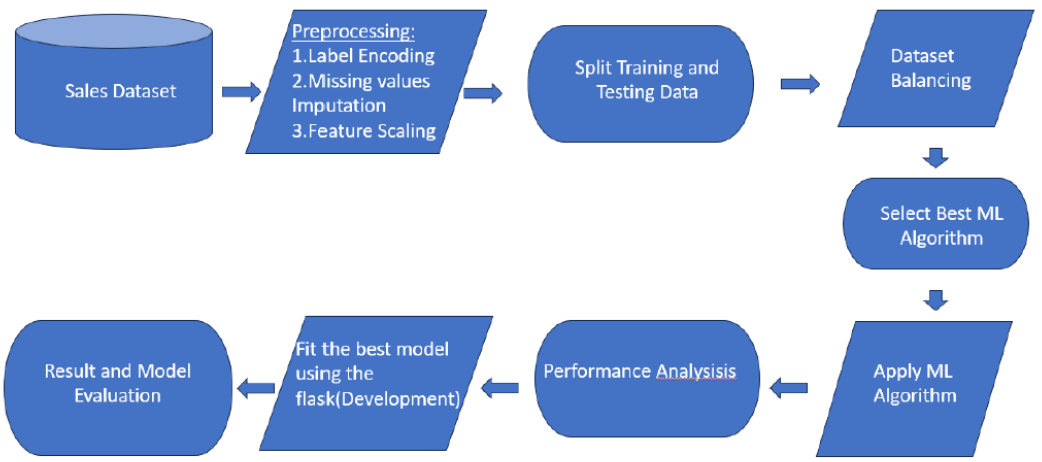


User Stories:

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance Criteria	Priority	Release
Store Manager	Sales Prediction	US01	As a store manager, I want to predict the sales for the next month to optimize inventory and staffing.	- The model must predict sales with 90% accuracy based on historical data. - Predictions should be available at least 7 days before the start of the month.	High	1.0
Inventory Analyst	Demand Forecasting	US02	As an inventory analyst, I want to receive demand forecasts for each product category, so I can adjust stock levels accordingly.	- Forecasts must be generated for each product category. - Forecasts should be updated weekly and have a margin of error within 5%.	High	1.0
Marketing Team	Customer Segmentation	US03	As a member of the marketing team, I want to identify customer segments to tailor promotions and marketing campaigns.	- The system should segment customers into at least three distinct groups based on purchase behavior. - Segmentation results should be updated monthly.	High	1.0
Data Analyst	Sales Performance Reporting	US04	As a data analyst, I want access to historical sales data and performance reports to analyze trends and make data-driven decisions.	- Access to historical sales data for the past 5 years. - Ability to generate performance reports with customizable date ranges and filter options.	Medium	1.1
Inventory Manager	Stock Replenishment	US05	As an inventory manager, I want the system to automatically generate restocking orders when stock levels are below a certain threshold.	- The system should create restocking orders for products when stock falls below the defined threshold. - Order recommendations should consider lead times and supplier availability.	High	1.1
Regional Director	Regional Sales Comparison	US06	As a regional director, I want to compare the sales performance of different regions to identify growth opportunities and challenges.	- Ability to compare sales data across multiple regions. - Visualizations and reports highlighting regional performance differences.	Medium	1.1
Data Scientist	Anomaly Detection	US07	As a data scientist, I want to develop machine learning models that can automatically detect anomalies in sales data, such as unusual spikes or drops.	- The system should provide access to historical data for training anomaly detection models. - Models should detect anomalies with at least 90% accuracy.	High	1.2
CFO	Profit Margin Analysis	US08	As the CFO, I want to analyze the profit margins for different products and categories to make informed financial decisions.	- Profit margins should be calculated and visualized for each product and product category. - Ability to filter and compare profit margins over time.	High	1.2

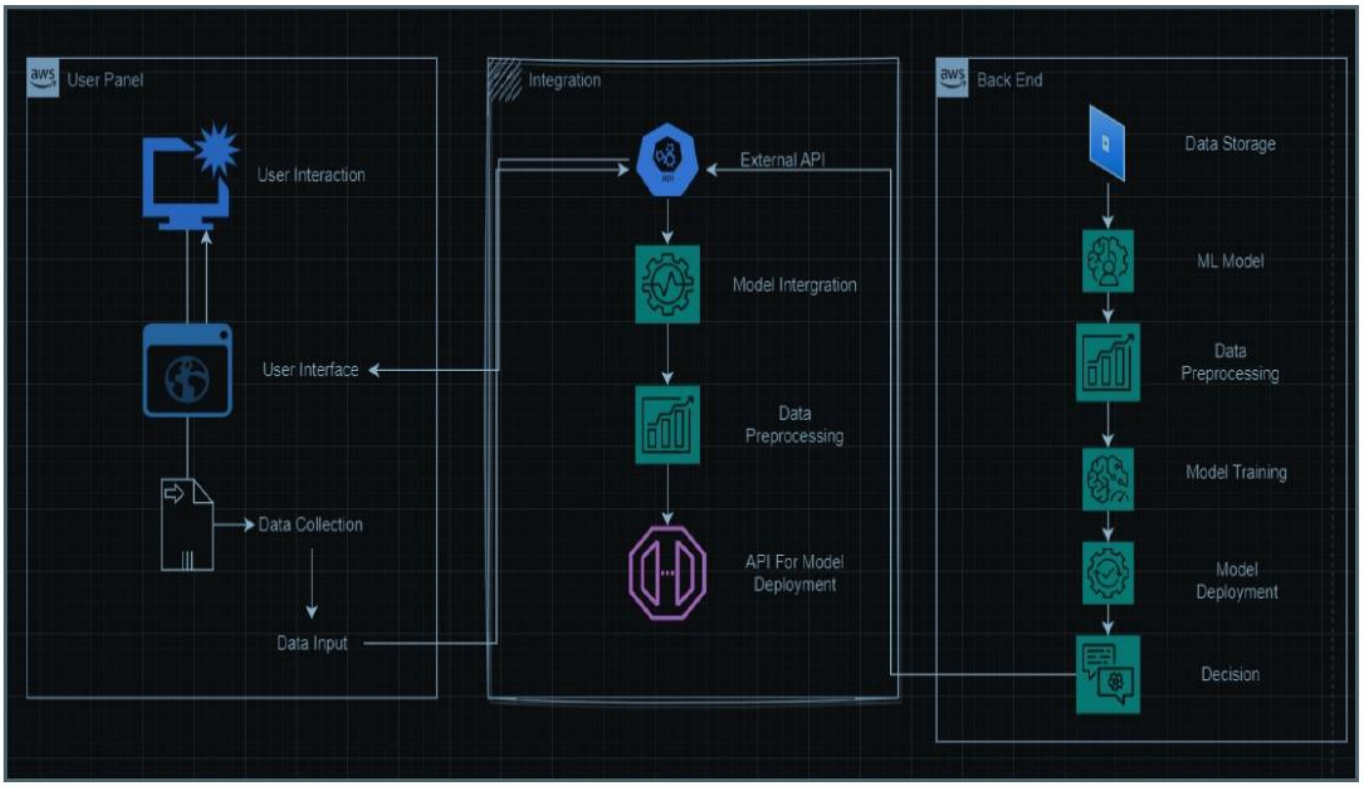
5.2 Solution Architecture

The solution architecture for Walmart sales analysis using data analytics and machine learning involves a systematic approach to harness data and algorithms to develop predictive models that can assist in optimizing sales and business strategies.
It predicts Walmart sales using machine learning by leveraging real-time sales and inventory data. Our system offers personalized sales and inventory assessments that consider factors such as historical sales trends, product categories, geographic locations, seasonal variations, promotions, and customer demographics.
By harnessing real-time data, our software can generate accurate sales forecasts and inventory recommendations, empowering Walmart and its stakeholders to make informed decisions in managing stock levels, pricing strategies, and marketing campaigns. This innovative solution aims to revolutionize the retail industry by promoting efficient inventory management, maximizing revenue, and enhancing the overall shopping experience.
Through seamless integration with existing point-of-sale and inventory management systems, our software ensures a hassle-free and efficient experience for Walmart employees and management. Sales data, inventory updates, and customer information are securely analyzed by our machine learning algorithms to provide personalized sales and inventory insights, enabling Walmart to optimize its operations.
The positive impact of our software extends beyond the retail giant to the broader society. By optimizing Walmart's operations, we can reduce waste, improve supply chain efficiency, and ensure that products are available when and where customers need them. This, in turn, benefits both consumers and Walmart as it enhances customer satisfaction and reduces costs associated with overstock or understock situation



6. Project Planning and Scheduling

6.1 Technical Architecture



6.2 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story/ Task	Story Points	Priority	Team Members
Sprint-1	Project Initialization & Infrastructure Setup	USN-1	As a project manager, I want to set up the development environment and configure the necessary infrastructure to commence the Walmart Sale Analysis project.	1	High	Akshara
Sprint-1	Data Collection	USN-2	As a data engineer, I want to gather and collate a comprehensive dataset of Walmart's historical sales data and relevant parameters for training the machine learning model.	2	High	Akshara
Sprint-2	Data preprocessing	USN-3	Preprocess the collected dataset by cleaning, normalizing, and splitting it into training and validation sets to prepare it for analysis.	3	High	Akshara

Sprint-3	Model Development & Training	USN-4	Select the most suitable model for sales prediction and train the chosen machine learning model using the preprocessed Walmart sales dataset.	5	High	Chetan
Sprint-4	Model Deployment & Integration	USN-5	As a software developer, I want to deploy the trained machine learning model as a service or API and integrate it into a user-friendly interface	6	High	Chetan
Sprint-5	Personalized Risk Assessment	USN-6	As an individual, I want to utilize the developed model to forecast future sales trends and make informed decisions	1	Medium	Yogitha
Sprint-5	Model Evaluation and Enhancement	USN-7	These user stories represent the key phases and tasks involved in the project planning for Walmart Sale Analysis in the retail industry using machine learning. The team members and their responsibilities can be customized based on your project team's composition.	2	High	Yogitha

6.3 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date(Planned)	Story Points Completed(as On Planned End Date)	Sprint Release Date (Actual)
Sprint-1	3	1 Day	28 Oct 2023	28 Oct 2023	3	28 Oct 2023
Sprint-2	3	1 Day	29 Oct 2023	29 Oct 2023	3	29 Oct 2023
Sprint-3	5	3 Days	30 Oct 2023	1 Nov 2023	5	1 Nov 2023
Sprint-4	6	3 Days	2 Nov 2023	4 Nov2023	6	4 Nov2023
Sprint-5	3	2 Days	5 Nov 2023	6 Nov 2023	3	6 Nov 2023

Velocity:

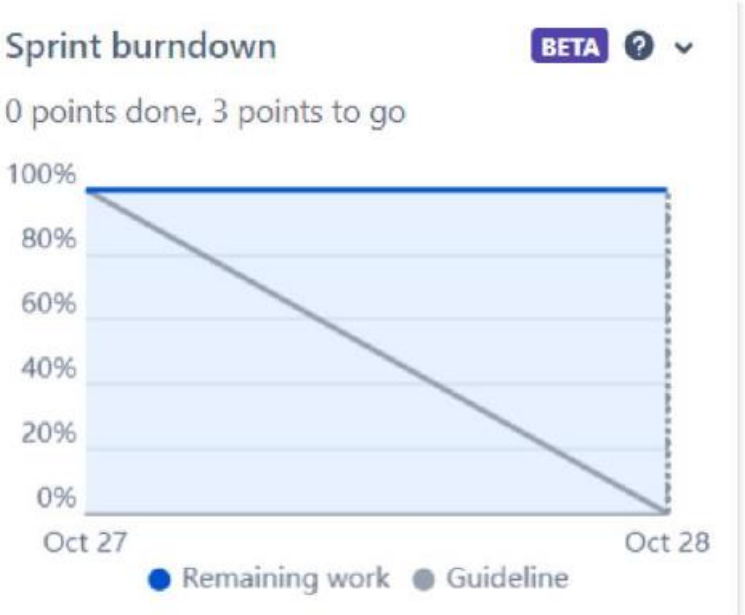
Imagine we have a 11-days sprint duration, and the velocity of the team is 3 (points per sprint). Let’s calculate the team’s average velocity (AV) per iteration unit (story points per day)

Velocity = (3+3+5+6+3) / 5 = 20 / 5 = 4

$$AV = \frac{sprint\ duration}{velocity}$$

AV = 11/4 = 2.75

Burndown Chart:



7. Coding and Solutioning

Walmart Store Sales Forecasting

Data Collection

```
[1]
from google.colab import drive
drive.mount('/content/drive')
! unzip '/content/walmart-recruiting-store-sales-forecasting.zip' -d '/content/destination_folder'
```

```
Mounted at /content/drive
Archive:  /content/walmart-recruiting-store-sales-forecasting.zip
  inflating: /content/destination_folder/features.csv.zip
  inflating: /content/destination_folder/sampleSubmission.csv.zip
  inflating: /content/destination_folder/stores.csv
  inflating: /content/destination_folder/test.csv.zip
  inflating: /content/destination_folder/train.csv.zip
```

```
[2]
! unzip '/content/destination_folder/features.csv.zip'
! unzip '/content/destination_folder/test.csv.zip'
! unzip '/content/destination_folder/train.csv.zip'
! unzip '/content/destination_folder/stores.csv'
```

```
Archive:  /content/destination_folder/features.csv.zip
  inflating: features.csv
Archive:  /content/destination_folder/test.csv.zip
  inflating: test.csv
Archive:  /content/destination_folder/train.csv.zip
  inflating: train.csv
Archive:  /content/destination_folder/stores.csv
  End-of-central-directory signature not found.  Either this file is not
  a zipfile, or it constitutes one disk of a multi-part archive.  In the
  latter case the central directory and zipfile comment will be found on
  the last disk(s) of this archive.
unzip:  cannot find zipfile directory in one of /content/destination_folder/stores.csv or
        /content/destination_folder/stores.csv.zip, and cannot find /content/destination_folder/stores.csv.ZIP, period.
```

Data Preprocessing

```
[3]
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from scipy import stats
import statsmodels.api as sm
```

```
[4]
feature=pd.read_csv('/content/features.csv')
feature
```

```
[5]
feature.info
```

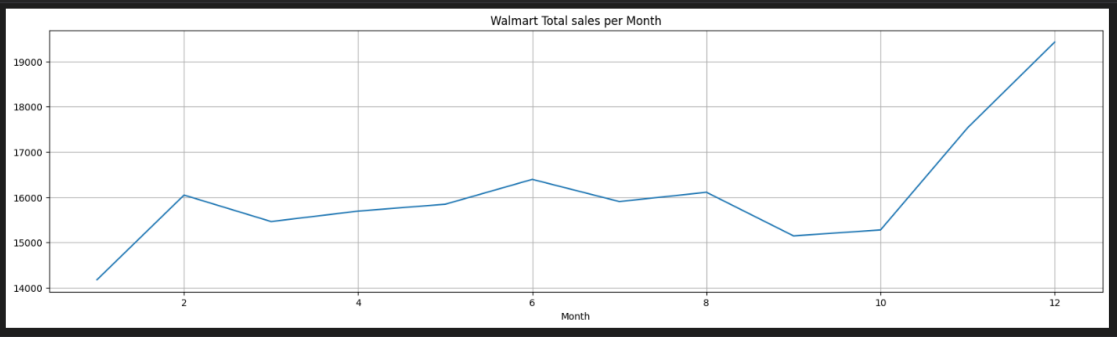
```
<bound method DataFrame.info of       Store        Date  Temperature  Fuel_Price  MarkDown1  MarkDown2  \
0         1  2010-02-05        42.31       2.572        NaN        NaN
1         1  2010-02-12        38.51       2.548        NaN        NaN
2         1  2010-02-19        39.93       2.514        NaN        NaN
3         1  2010-02-26        46.63       2.561        NaN        NaN
4         1  2010-03-05        46.50       2.625        NaN        NaN
...     ...         ...          ...         ...        ...        ...
8185     45  2013-06-28        76.05       3.639    4842.29     975.03
8186     45  2013-07-05        77.50       3.614    9090.48    2268.58
8187     45  2013-07-12        79.37       3.614    3789.94    1827.31
8188     45  2013-07-19        82.84       3.737    2961.49    1047.07
8189     45  2013-07-26        76.06       3.804     212.02     851.73

      MarkDown3  MarkDown4  MarkDown5         CPI  Unemployment  IsHoliday
0           NaN        NaN        NaN  211.096358         8.106      False
1           NaN        NaN        NaN  211.242170         8.106       True
2           NaN        NaN        NaN  211.289143         8.106      False
3           NaN        NaN        NaN  211.319643         8.106      False
4           NaN        NaN        NaN  211.350143         8.106      False
...         ...        ...        ...         ...           ...        ...
8185       3.00    2449.97    3169.69         NaN           NaN      False
8186     582.74    5797.47    1514.93         NaN           NaN      False
8187      85.72     744.84    2150.36         NaN           NaN      False
8188     204.19     363.00    1059.46         NaN           NaN      False
8189       2.06      10.88    1864.57         NaN           NaN      False

[8190 rows x 12 columns]>
```

Visualisation and Data Analytics

```
sales =train.groupby(by='Month')['Weekly_Sales'].aggregate('mean')
sales.plot(figsize=(20,5))
plt.title('Walmart Total sales per Month')
plt.grid()
```




```
data_monthly = pd.crosstab(train["Year"], train["Month"], values=train["Weekly_Sales"],aggfunc='sum')
data_monthly
```

Model Selection

Random Forest

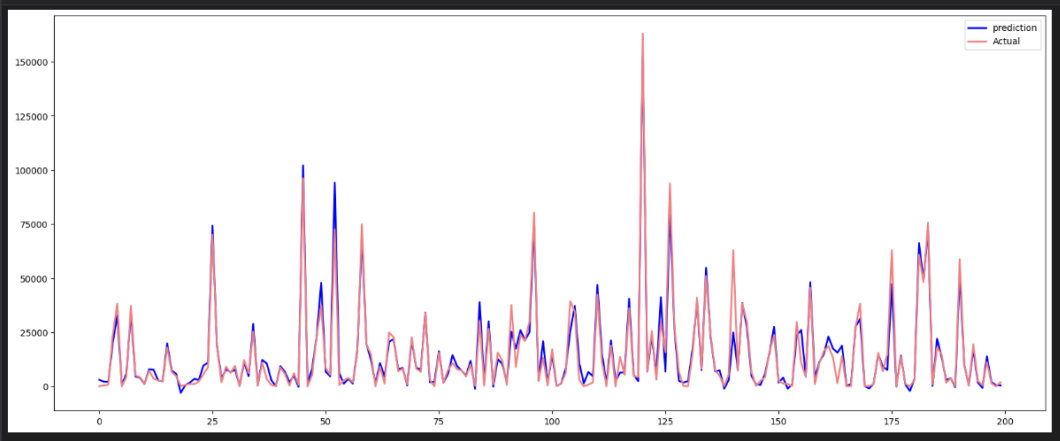
```
[29]
from sklearn.ensemble import RandomForestRegressor
rf_model =RandomForestRegressor(criterion='entropy')
rf_model = RandomForestRegressor(n_estimators=100)
rf_model.fit(X_train, Y_train)
```

```
[31]
rf_predictions = rf_model.predict(X_test)
rf_df=pd.DataFrame({'Actual':Y_test,'Predicted':rf_predictions})
rf_df
```

```
[32]
from sklearn import metrics
from sklearn.metrics import accuracy_score
rf_acc = rf_model.score(X_test,Y_test)*100
print("Random Forest Regressor Accuracy - ",rf_acc)
print("MAE" , metrics.mean_absolute_error(Y_test, rf_predictions))
print("MSE" , metrics.mean_squared_error(Y_test, rf_predictions))
print("RMSE" , np.sqrt(metrics.mean_squared_error(Y_test,rf_predictions)))
print("R2" , metrics.explained_variance_score(Y_test,rf_predictions))
```

```
Random Forest Regressor Accuracy -  97.54186005293826
MAE 1428.6943410940976
MSE 12566681.521175072
RMSE 3544.9515541365404
R2 0.9754186136848442
```

```
plt.figure(figsize=(20,8))
plt.plot(xgbr.predict(X_test[:200]), label="prediction", linewidth=2.0,color='blue')
plt.plot(Y_test[:200].values, label="Actual", linewidth=2.0,color='lightcoral')
plt.legend(loc="best")
plt.show()
```




```
pkl_filename = "./xgboost_regressor.pkl"
if (not path.isfile(pkl_filename)):
  with open(pkl_filename, 'wb') as file:
    pickle.dump(xgbr, file)
  print("Saved model to disk")
else:
  print("Model already saved")
```

```
Saved model to disk
```

Decision Tree

```
from sklearn.tree import DecisionTreeRegressor
model1 = DecisionTreeRegressor(max_depth=4,splitter='best')
model1.fit(X_train,Y_train)
```

[41]

```
d_y_predict = model1.predict(X_test)
d_y_predict
```

[42]

```
array([10493.6296196 , 10493.6296196 , 10493.6296196 , ...,
       10493.6296196 ,  5424.84160457, 10493.6296196 ])
```

```
dec_acc = model1.score(X_test,Y_test)*100
print("Decision Tree Regressor Accuracy - ",dec_acc)
```

[43]

```
Decision Tree Regressor Accuracy -  41.61885458720538
```

```
print("MAE" , metrics.mean_absolute_error(Y_test, d_y_predict))
print("MSE" , metrics.mean_squared_error(Y_test, d_y_predict))
print("RMSE" , np.sqrt(metrics.mean_squared_error(Y_test, d_y_predict)))
print("R2" , metrics.explained_variance_score(Y_test, d_y_predict))
```

[44]

```
MAE 10706.229134975647
MSE 298460330.59303826
RMSE 17276.00447421331
R2 0.4161886284266293
```

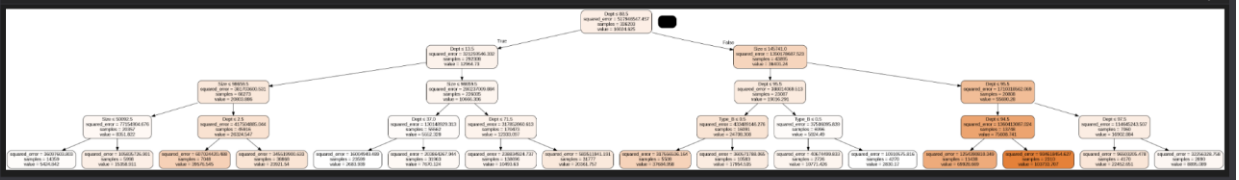
+ Code

```
from six import StringIO
from IPython.display import Image
import pydotplus
from sklearn.tree import export_graphviz
```

[45] Python

```
dot_data =StringIO()
export_graphviz(model1,out_file=dot_data,feature_names= X.columns,
                filled=True,rounded= True,special_characters=True)
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png())
```

[46] Python



```
pkl_filename = "./decisiontree_regressor.pkl"
if (not path.isfile(pkl_filename)):
  with open(pkl_filename, 'wb') as file:
    pickle.dump(model1, file)
  print("Saved model to disk")
else:
  print("Model already saved")
```

[47] Python

```
Saved model to disk
```

XgBoost

```
[36]
from xgboost import XGBRegressor
xgbr = XGBRegressor()
xgbr.fit(X_train, Y_train)
```

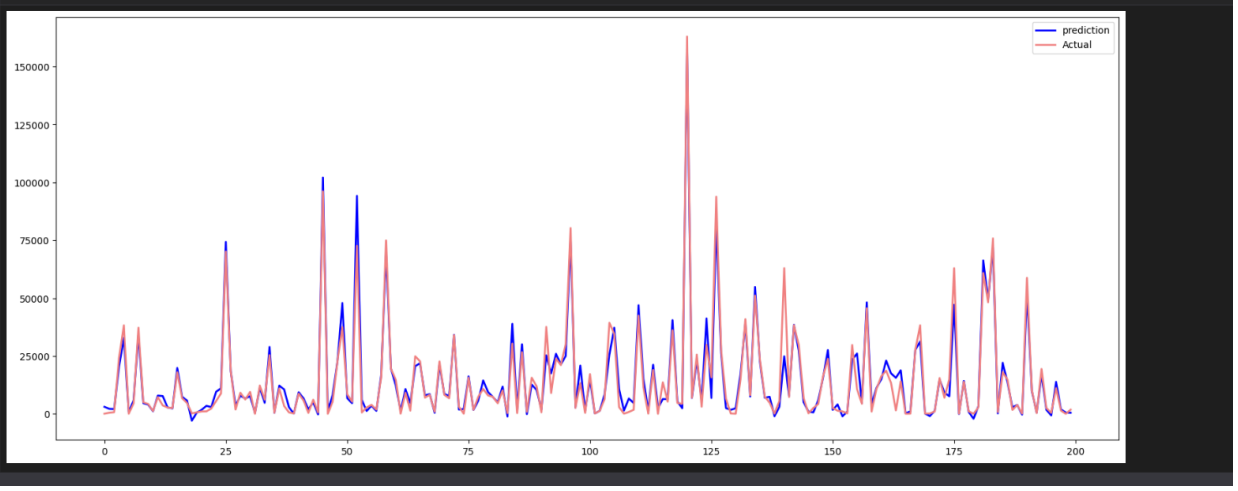
```
[37]
xgb_acc = xgbr.score(X_test,Y_test)*100
print("XGBoost Regressor Accuracy - ",xgb_acc)
```

```
XGBoost Regressor Accuracy -  94.57550886297496
```

```
[38]
y_pred = xgbr.predict(X_test)
print("MAE" , metrics.mean_absolute_error(Y_test, y_pred))
print("MSE" , metrics.mean_squared_error(Y_test, y_pred))
print("RMSE" , np.sqrt(metrics.mean_squared_error(Y_test, y_pred)))
print("R2" , metrics.explained_variance_score(Y_test, y_pred))
```

```
MAE 2958.031149412797
MSE 27731477.459171012
RMSE 5266.068501184827
R2 0.9457551204707744
```

```
[39]
plt.figure(figsize=(20,8))
plt.plot(xgbr.predict(X_test[:200]), label="prediction", linewidth=2.0,color='blue')
plt.plot(Y_test[:200].values, label="Actual", linewidth=2.0,color='lightcoral')
plt.legend(loc="best")
plt.show()
```



ARIMA

```
from statsmodels.tsa.statespace.sarimax import SARIMAX
arima_model = SARIMAX(train['Weekly_Sales'], order=(1, 1, 1), seasonal_order=(1, 1, 1, 12))
arima_result = arima_model.fit(disp=False)
```

```
from sklearn.neighbors import KNeighborsRegressor
knn = KNeighborsRegressor(n_neighbors = 1,weights = 'uniform')
knn.fit(X_train,Y_train)
```

```
from sklearn import metrics
ar_predict=knn.predict(X_test)
print("MAE" , metrics.mean_absolute_error(Y_test, ar_predict))
print("MSE" , metrics.mean_squared_error(Y_test, ar_predict))
print("RMSE" , np.sqrt(metrics.mean_squared_error(Y_test, ar_predict)))
print("R2" , metrics.explained_variance_score(Y_test, ar_predict))
```

[51]

```
MAE 14351.089629272705
MSE 624187710.7837687
RMSE 24983.748933732277
R2 -0.2209591282581107
```

+ Code

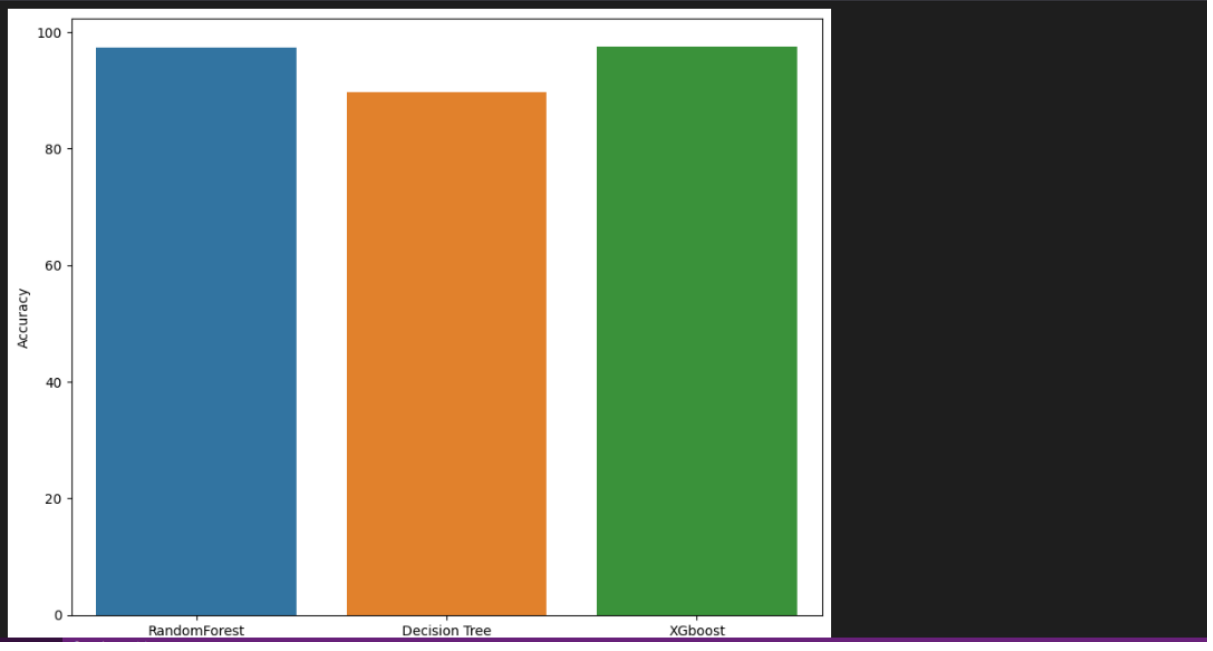
Comparing Models

Comparing Models

```
acc = {'Model':['RandomForest','Decision Tree','XGboost'],'Accuracy':[rf_acc,dec_acc,xgb_acc]}
acc_df = pd.DataFrame(acc)
acc_df
```

[52]

```
plt.figure(figsize=(10,8))
sns.barplot(x='Model',y='Accuracy',data=acc_df)
plt.show()
```



Sure thing! Let's break down each feature in your Walmart sales analysis dataset:

Store: This represents the store number where the sales were recorded.

Dept: Refers to the department within the store where the sales occurred. Departments are typically categorized by the type of products they sell.

Date: The date when the sales data was recorded. It can be used to analyze sales trends over time.

Size: The size of the store, which could have an impact on the overall sales. Larger stores might generally have higher sales.

Temperature: The temperature on the recorded date. Weather conditions can influence consumer behavior, especially for seasonal products.

Fuel_Price: The cost of fuel on the recorded date. Changes in fuel prices can affect consumer spending patterns.

CPI (Consumer Price Index): A measure of the average change over time in the prices paid by urban consumers for a market basket of consumer goods and services. It helps in understanding inflation and its impact on sales.

Unemployment: The unemployment rate on the recorded date. Economic conditions, including employment rates, can influence consumer confidence and spending.

IsHoliday: A binary indicator of whether the recorded date is a holiday or not. Holidays often lead to increased sales.

Year, Month, Week: These represent the time components of the recorded date and can be used for time-based analysis.

max, min, mean, median, std: Statistical measures of weekly sales data, providing insights into the distribution and variability of sales.

Total_MarkDown: The total amount of markdowns (discounts) applied to products. Markdowns can impact sales by attracting more customers.

Type_A, Type_B, Type_C: Categorical variables indicating the type of product or category. Different types may have varying sales patterns.

Weekly_Sales (Target Variable): The actual sales for the week, which is the variable you want to predict or analyze.

8. Performance Testing

8.1 Performance Metrics

Random Forest

```
from sklearn import metrics
from sklearn.metrics import accuracy_score
rf_acc = rf_model.score(X_test,Y_test)*100
print("Random Forest Regressor Accuracy - ",rf_acc)
print("MAE" , metrics.mean_absolute_error(Y_test, rf_predictions))
print("MSE" , metrics.mean_squared_error(Y_test, rf_predictions))
print("RMSE" , np.sqrt(metrics.mean_squared_error(Y_test,rf_predictions)))
print("R2" , metrics.explained_variance_score(Y_test,rf_predictions))
```

```
Random Forest Regressor Accuracy -  97.54186005293826
MAE 1428.6943410940976
MSE 12566681.521175072
RMSE 3544.9515541365404
R2 0.9754186136848442
```

Decision Tree

```
dec_acc = model1.score(X_test,Y_test)*100
print("Decision Tree Regressor Accuracy - ",dec_acc)
```

```
Decision Tree Regressor Accuracy -  41.61885458720538
```

```
print("MAE" , metrics.mean_absolute_error(Y_test, d_y_predict))
print("MSE" , metrics.mean_squared_error(Y_test, d_y_predict))
print("RMSE" , np.sqrt(metrics.mean_squared_error(Y_test, d_y_predict)))
print("R2" , metrics.explained_variance_score(Y_test, d_y_predict))
```

```
MAE 10706.229134975647
MSE 298460330.59303826
RMSE 17276.00447421331
R2 0.4161886284266293
```

XgBoost

```
[37]
xgb_acc = xgbr.score(X_test,Y_test)*100
print("XGBoost Regressor Accuracy - ",xgb_acc)
```

```
XGBoost Regressor Accuracy -  94.57550886297496
```

```
[38]
y_pred = xgbr.predict(X_test)
print("MAE" , metrics.mean_absolute_error(Y_test, y_pred))
print("MSE" , metrics.mean_squared_error(Y_test, y_pred))
print("RMSE" , np.sqrt(metrics.mean_squared_error(Y_test, y_pred)))
print("R2" , metrics.explained_variance_score(Y_test, y_pred))
```

```
MAE 2958.031149412797
MSE 27731477.459171012
RMSE 5266.068501184827
R2 0.9457551204707744
```

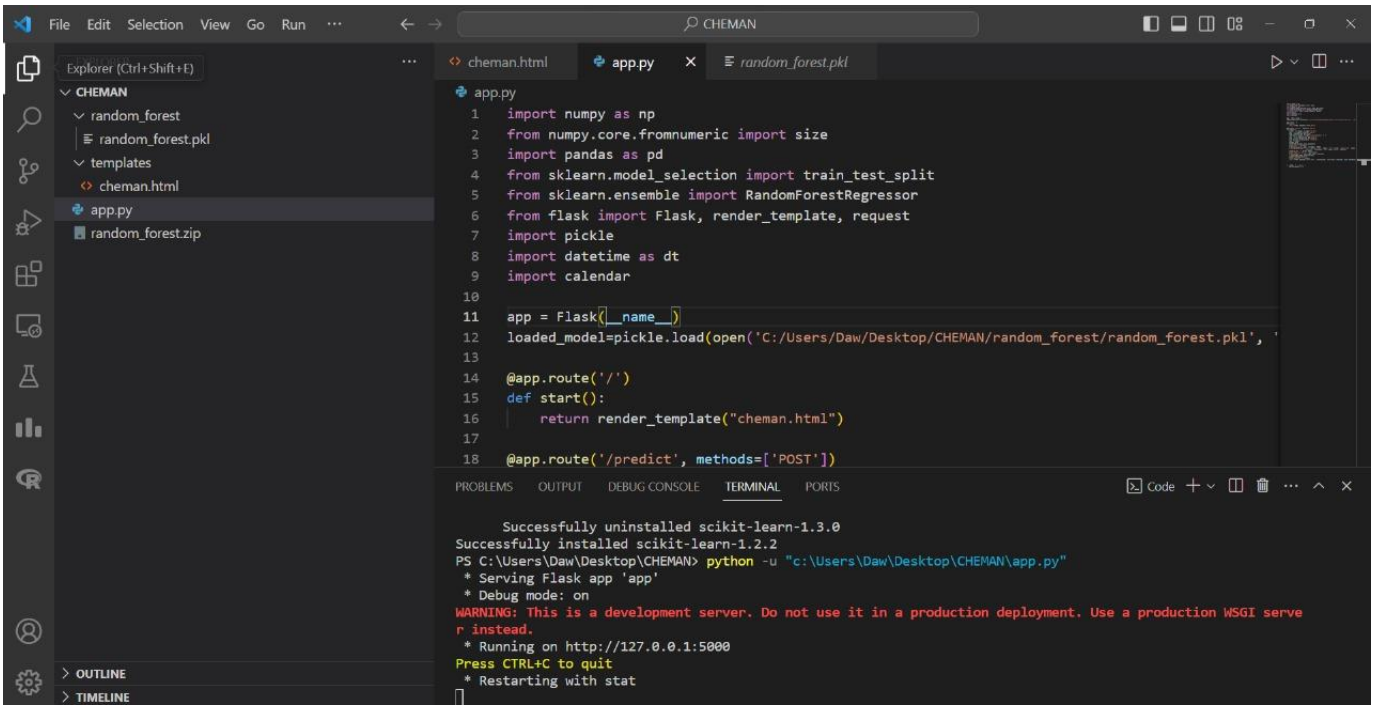
ARIMA

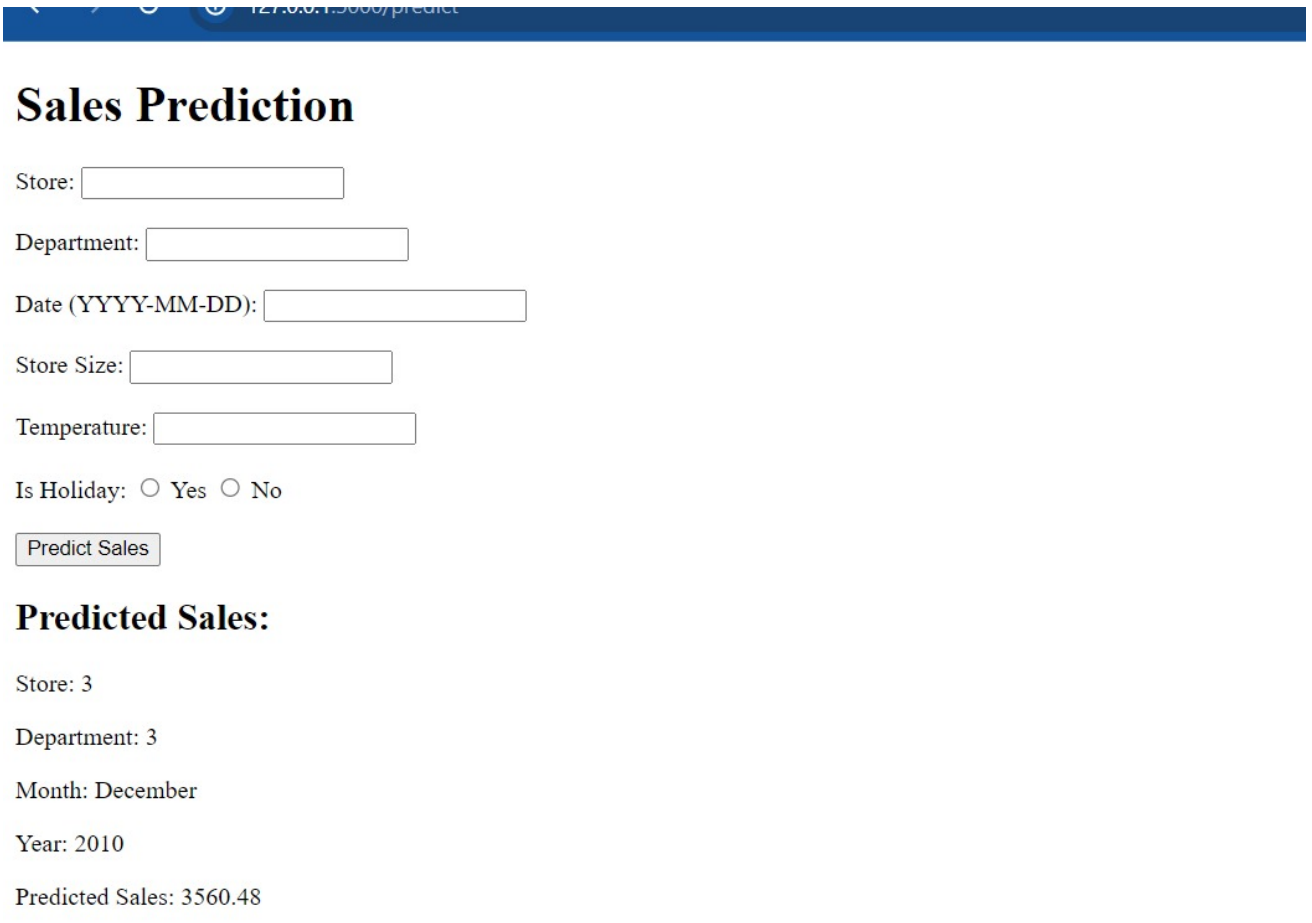
```
[51]
from sklearn import metrics
ar_predict=knn.predict(X_test)
print("MAE" , metrics.mean_absolute_error(Y_test, ar_predict))
print("MSE" , metrics.mean_squared_error(Y_test, ar_predict))
print("RMSE" , np.sqrt(metrics.mean_squared_error(Y_test, ar_predict)))
print("R2" , metrics.explained_variance_score(Y_test, ar_predict))
```

```
MAE 14351.089629272705
MSE 624187710.7837687
RMSE 24983.748933732277
```

9.Results

9.1 Output Screenshots





10.Advantages and Disadvantages

Let's delve deeper into the advantages and disadvantages of using machine learning algorithms with Walmart sales data in the retail industry:

Advantages:

1. Rich and Diverse Data:

Walmart generates vast amounts of data, including sales records, customer information, product details, and more. This diversity of data provides a comprehensive view of retail operations.

2. Large and Representative Dataset:

Walmart's sales data is extensive and representative of real-world retail transactions. This allows machine learning algorithms to learn from a broad range of retail scenarios, improving their ability to make accurate predictions.

3. Scalability:

Machine learning algorithms often require large datasets to build robust models. Walmart's data is scalable, which is crucial for training algorithms, particularly deep learning models that thrive on large amounts of data.

4. Feature Engineering Opportunities:

Walmart's dataset provides ample opportunities for feature engineering. Machine learning algorithms can take advantage of various features like product attributes, customer demographics, sales history, seasonal trends, and more to enhance predictive accuracy.

5. Real-Time Prediction:

Walmart's data enables real-time analysis and prediction. Machine learning algorithms can be deployed to make timely decisions such as inventory management, demand forecasting, and personalized marketing strategies, which can optimize retail operations.

Disadvantages:

1. Data Preprocessing Complexity:

Retail sales data can be messy, containing missing values, outliers, and inconsistencies. Machine learning algorithms are highly sensitive to data quality, requiring thorough preprocessing to clean and transform the data, which can be time-consuming.

2. Data Privacy Concerns:

Retail data often contains sensitive customer information. Machine learning projects must adhere to data privacy regulations and ethical standards, adding a layer of complexity to data handling and model development.

3. Data Imbalances:

Imbalanced data is common in retail, where certain products or customer segments may be overrepresented. Machine learning algorithms need strategies to address class imbalances to prevent biased predictions.

4. Domain-Specific Challenges:

The retail industry has unique challenges, including seasonality, market trends, and consumer behavior. Machine learning algorithms must be tailored to accommodate these domain-specific issues, which may require a deep understanding of the retail sector.

5. Model Complexity:

Building accurate machine learning models for retail data can be complex. Some algorithms, such as deep learning or ensemble methods, may be necessary to capture the intricacies of retail operations. These models require expertise and computational resources.

6. Access to Data:

Access to Walmart's sales data may be restricted, and obtaining the data may require partnerships or agreements. This can be a barrier for smaller organizations or research teams without the necessary resources.

In summary, using machine learning algorithms with Walmart sales data provides benefits in terms of dataset size, scalability, feature engineering opportunities, and real-time analysis. However, it also presents challenges related to data preprocessing, privacy concerns, class imbalances, domain-specific issues, model complexity, and data access. Successful projects require careful data preparation, ethical considerations, and expertise in machine learning and the retail industry.

11.Conclusion

As you can observe RandomForest and XGBoost has highest accuracy. So lets compare all the models for the final Model.

```
from prettytable import PrettyTable
tb= PrettyTable()
tb.field_names = ["Model", "Accuracy", "RMSE", "MAE"]
tb.add_row(["Random Forest", 96.413, 4282.086, 1637.9485])
tb.add_row(["XgBoost", 93.927, 5571.814, 3001.270])
tb.add_row(["Decision Tree",89.161,7258.674,3019.084])
print(tb)
```

```
+---------------+----------+----------+-----------+
|     Model     | Accuracy |   RMSE   |    MAE    |
+---------------+----------+----------+-----------+
| Random Forest |  96.413  | 4282.086 | 1637.9485 |
|    XgBoost    |  93.927  | 5571.814 |  3001.27  |
| Decision Tree |  89.161  | 7258.674 |  3019.084 |
+---------------+----------+----------+-----------+
```

As Random Forest model has highest accuracy we are going to use it as the final model for sales forecasting

```
[53]
from sklearn.model_selection import cross_val_score
rf = RandomForestRegressor(n_estimators=58, max_depth=27, min_samples_split=3, min_samples_leaf=1)
rf.fit(X_train, Y_train.ravel())
y_pred = rf.predict(X_test)
```

```
[54]
pickle.dump(rf,open('final_model.pkl','wb'))
```

As we can see Random Forest has the highest accuracy among all the other models

In today's dynamic retail environment, the integration of machine learning algorithms into Walmart sales analysis represents a powerful tool for retailers to gain a competitive edge. The advantages are multifaceted, beginning with improved demand forecasting and inventory optimization. By harnessing the predictive capabilities of machine learning, retailers can align their inventory levels more closely with actual customer demand, reducing the risks of overstocking or stockouts. This fine-tuning not only results in cost savings but also enhances customer satisfaction, ensuring that the products customers desire are readily available.

Personalized marketing and customer segmentation further underscore the potential of machine learning. Retailers can leverage algorithms to gain a nuanced understanding of their customers, enabling tailored marketing campaigns that cater to individual preferences and purchasing behaviors. As a result, customer engagement flourishes, conversion rates rise, and customer loyalty deepens, creating a win-win scenario for both retailers and consumers.Dynamic pricing strategies, powered by machine learning algorithms such as XGBoost and Random Forest, offer retailers a real-time

advantage. The ability to analyze market conditions, monitor competitor pricing, and incorporate historical sales data allows retailers to adjust pricing dynamically, optimizing revenue and profit margins while staying responsive to market fluctuations and customer demand.

Supply chain optimization is another critical facet. Machine learning predicts demand and identifies cost-saving opportunities in logistics, leading to an efficient and cost-effective supply chain. Smoother operations, reduced transportation costs, and minimized waste translate to a leaner and more competitive retail operation.

Enhancing the customer experience is pivotal, and machine learning plays a significant role in this area. From personalized product recommendations to responsive chatbots for customer support, machine learning-driven solutions create a more satisfying and engaging shopping experience for customers, bolstering their loyalty.

Moreover, machine learning assists in the critical task of fraud detection and loss prevention. Retailers can rely on algorithms to scrutinize sales data for unusual patterns and anomalies, enabling them to detect and thwart fraudulent activities that might otherwise erode profits.

A/B testing and experimentation are crucial for data-driven decision-making. Machine learning supports the design and evaluation of A/B tests, enabling retailers to fine-tune strategies related to pricing, promotions, and store layouts. This iterative approach ensures that retailers are consistently optimizing their operations and strategies based on tangible results.

Last but not least, market basket analysis empowers retailers to understand which products are frequently purchased together, informing product placement and cross-selling strategies. This deepens customer engagement and increases the average transaction value.

In the ever-evolving retail landscape, machine learning-driven Walmart sales analysis isn't merely a desirable addition; it's a necessity for those who seek to remain competitive. With the ability to adapt to changing market dynamics, make data-driven decisions, and deliver unparalleled shopping experiences, retailers can position themselves at the forefront of the industry's transformation. It's a promising avenue for retailers to thrive and excel in an era where data-driven insights and customer-centric strategies are paramount.

12.Future Scope

- Demand Forecasting and Inventory Optimization:
  Machine learning algorithms can improve demand forecasting accuracy, enabling retailers to stock products more efficiently. This leads to reduced overstock, minimized stockouts, and optimized inventory management, ultimately resulting in cost savings and improved customer satisfaction.
- Personalized Marketing and Customer Segmentation:
  By leveraging machine learning models, retailers can create personalized marketing campaigns tailored to individual customer preferences and behavior. This results in higher customer engagement, increased conversion rates, and stronger customer loyalty.
- Dynamic Pricing Strategies:
  Machine learning algorithms, such as XGBoost and Random Forest, can analyze market conditions, competitor pricing, and historical sales data in real-time. Retailers can adjust prices dynamically to maximize revenue and profit margins, responding to market fluctuations and customer demand.
- Supply Chain Optimization:
  Machine learning can optimize the supply chain by predicting demand, reducing transportation costs, and improving the overall efficiency of logistics operations. This results in smoother operations and reduced costs for retailers.
- Enhanced Customer Experience:
  Retailers can use machine learning to offer personalized product recommendations, chatbots for customer support, and improved user interfaces for online and in-store experiences. This leads to a more satisfying and engaging shopping experience for customers.
- Fraud Detection and Loss Prevention:
  Machine learning algorithms can identify unusual patterns and anomalies in sales data, helping retailers detect fraudulent activities and reduce losses due to theft or scams, safeguarding their profits.
- A/B Testing and Experimentation:

Machine learning can assist in designing and evaluating A/B tests and experiments. Retailers can fine-tune strategies related to pricing, promotions, and store layouts, ensuring they make data-driven decisions that improve overall performance.

- Market Basket Analysis:
  Machine learning can perform market basket analysis to identify which products are frequently purchased together. Retailers can use this information to optimize product placement and encourage cross-selling, increasing average transaction values.
  Incorporating machine learning algorithms into Walmart sales analysis empowers retailers to make data-driven decisions, improve operations, and create more personalized and efficient experiences for customers. These applications are vital for staying competitive in the rapidly evolving retail industry.

13.Appendix

Source Code Link:

https://drive.google.com/file/d/1zqDWIUqCcHfPW-BROfRqU_zfDLMtAjzc/view?usp=sharing

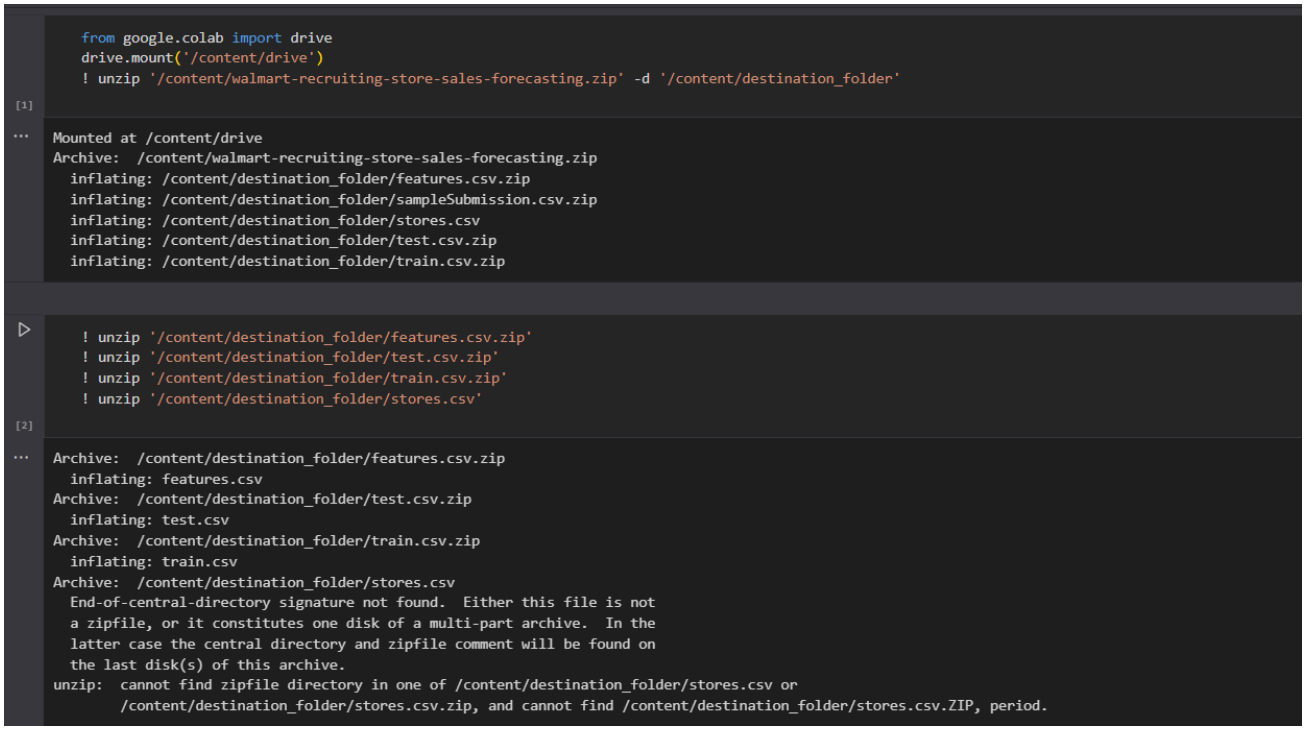
Flask Code Link:

https://drive.google.com/file/d/1B7uUtj82_cPqNBe1w8RG7Ya84mRxAeT6/view?usp=sharing

Some Screenshots:

Walmart Store Sales Forecasting

Data Collection



```
[1]
from google.colab import drive
drive.mount('/content/drive')
! unzip '/content/walmart-recruiting-store-sales-forecasting.zip' -d '/content/destination_folder'

Mounted at /content/drive
Archive:  /content/walmart-recruiting-store-sales-forecasting.zip
  inflating: /content/destination_folder/features.csv.zip
  inflating: /content/destination_folder/sampleSubmission.csv.zip
  inflating: /content/destination_folder/stores.csv
  inflating: /content/destination_folder/test.csv.zip
  inflating: /content/destination_folder/train.csv.zip

[2]
! unzip '/content/destination_folder/features.csv.zip'
! unzip '/content/destination_folder/test.csv.zip'
! unzip '/content/destination_folder/train.csv.zip'
! unzip '/content/destination_folder/stores.csv'

Archive:  /content/destination_folder/features.csv.zip
  inflating: features.csv
Archive:  /content/destination_folder/test.csv.zip
  inflating: test.csv
Archive:  /content/destination_folder/train.csv.zip
  inflating: train.csv
Archive:  /content/destination_folder/stores.csv
  End-of-central-directory signature not found.  Either this file is not
  a zipfile, or it constitutes one disk of a multi-part archive.  In the
  latter case the central directory and zipfile comment will be found on
  the last disk(s) of this archive.
unzip:  cannot find zipfile directory in one of /content/destination_folder/stores.csv or
        /content/destination_folder/stores.csv.zip, and cannot find /content/destination_folder/stores.csv.ZIP, period.
```

Data Preprocessing

```
[3]
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from scipy import stats
import statsmodels.api as sm
```

```
[4]
feature=pd.read_csv('/content/features.csv')
feature
```

```
...
```

```
[5]
feature.info
```

```
...
<bound method DataFrame.info of       Store        Date  Temperature  Fuel_Price  MarkDown1  MarkDown2  \
0         1  2010-02-05        42.31       2.572        NaN        NaN
1         1  2010-02-12        38.51       2.548        NaN        NaN
2         1  2010-02-19        39.93       2.514        NaN        NaN
3         1  2010-02-26        46.63       2.561        NaN        NaN
4         1  2010-03-05        46.50       2.625        NaN        NaN
...     ...         ...          ...         ...        ...        ...
8185     45  2013-06-28        76.05       3.639    4842.29     975.03
8186     45  2013-07-05        77.50       3.614    9090.48    2268.58
8187     45  2013-07-12        79.37       3.614    3789.94    1827.31
8188     45  2013-07-19        82.84       3.737    2961.49    1047.07
8189     45  2013-07-26        76.06       3.804     212.02     851.73

      MarkDown3  MarkDown4  MarkDown5         CPI  Unemployment  IsHoliday
0           NaN        NaN        NaN  211.096358         8.106      False
1           NaN        NaN        NaN  211.242170         8.106       True
2           NaN        NaN        NaN  211.289143         8.106      False
3           NaN        NaN        NaN  211.319643         8.106      False
4           NaN        NaN        NaN  211.350143         8.106      False
...         ...        ...        ...         ...           ...        ...
8185       3.00    2449.97    3169.69         NaN           NaN      False
8186     582.74    5797.47    1514.93         NaN           NaN      False
8187      85.72     744.84    2150.36         NaN           NaN      False
8188     204.19     363.00    1059.46         NaN           NaN      False
8189       2.06      10.88    1864.57         NaN           NaN      False

[8190 rows x 12 columns]>
```

Model Selection

Random Forest

```
from sklearn.ensemble import RandomForestRegressor
rf_model =RandomForestRegressor(criterion='entropy')
rf_model = RandomForestRegressor(n_estimators=100)
rf_model.fit(X_train, Y_train)
```

[29]

```
rf_predictions = rf_model.predict(X_test)
rf_df=pd.DataFrame({'Actual':Y_test,'Predicted':rf_predictions})
rf_df
```

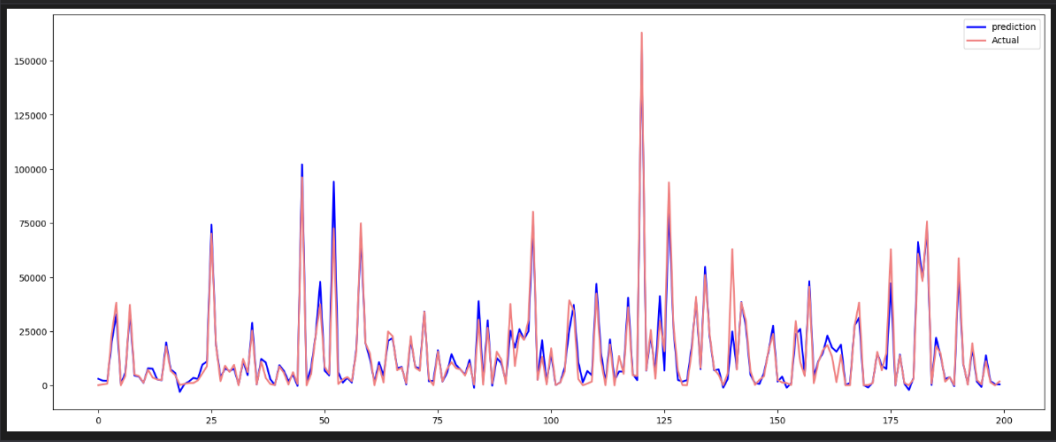
[31]

```
from sklearn import metrics
from sklearn.metrics import accuracy_score
rf_acc = rf_model.score(X_test,Y_test)*100
print("Random Forest Regressor Accuracy - ",rf_acc)
print("MAE" , metrics.mean_absolute_error(Y_test, rf_predictions))
print("MSE" , metrics.mean_squared_error(Y_test, rf_predictions))
print("RMSE" , np.sqrt(metrics.mean_squared_error(Y_test,rf_predictions)))
print("R2" , metrics.explained_variance_score(Y_test,rf_predictions))
```

[32]

```
Random Forest Regressor Accuracy -  97.54186005293826
MAE 1428.6943410940976
MSE 12566681.521175072
RMSE 3544.9515541365404
R2 0.9754186136848442
```

```
plt.figure(figsize=(20,8))
plt.plot(xgbr.predict(X_test[:200]), label="prediction", linewidth=2.0,color='blue')
plt.plot(Y_test[:200].values, label="Actual", linewidth=2.0,color='lightcoral')
plt.legend(loc="best")
plt.show()
```




```
pkl_filename = "./xgboost_regressor.pkl"
if (not path.isfile(pkl_filename)):
  with open(pkl_filename, 'wb') as file:
    pickle.dump(xgbr, file)
  print("Saved model to disk")
else:
  print("Model already saved")
```

[38]

```
Saved model to disk
```

Decision Tree

```
from sklearn.tree import DecisionTreeRegressor
model1 = DecisionTreeRegressor(max_depth=4,splitter='best')
model1.fit(X_train,Y_train)
```
[41]

```
d_y_predict = model1.predict(X_test)
d_y_predict
```
[42]

```
array([10493.6296196 , 10493.6296196 , 10493.6296196 , ...,
       10493.6296196 ,  5424.84160457, 10493.6296196 ])
```

```
dec_acc = model1.score(X_test,Y_test)*100
print("Decision Tree Regressor Accuracy - ",dec_acc)
```
[43]

```
Decision Tree Regressor Accuracy -  41.61885458720538
```

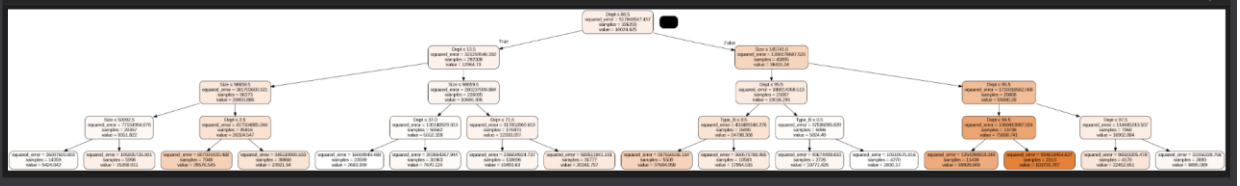
```
print("MAE" , metrics.mean_absolute_error(Y_test, d_y_predict))
print("MSE" , metrics.mean_squared_error(Y_test, d_y_predict))
print("RMSE" , np.sqrt(metrics.mean_squared_error(Y_test, d_y_predict)))
print("R2" , metrics.explained_variance_score(Y_test, d_y_predict))
```
[44]

```
MAE 10706.229134975647
MSE 298460330.59303826
RMSE 17276.00447421331
R2 0.4161886284266293
```

+ Code

```
from six import StringIO
from IPython.display import Image
import pydotplus
from sklearn.tree import export_graphviz
```
[45] Python

```
dot_data =StringIO()
export_graphviz(model1,out_file=dot_data,feature_names= X.columns,
                filled=True,rounded= True,special_characters=True)
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png())
```
[46] Python



```
pkl_filename = "./decisiontree_regressor.pkl"
if (not path.isfile(pkl_filename)):
  with open(pkl_filename, 'wb') as file:
    pickle.dump(model1, file)
  print("Saved model to disk")
else:
  print("Model already saved")
```
[47] Python

```
Saved model to disk
```

XgBoost

```
[36]
from xgboost import XGBRegressor
xgbr = XGBRegressor()
xgbr.fit(X_train, Y_train)
```

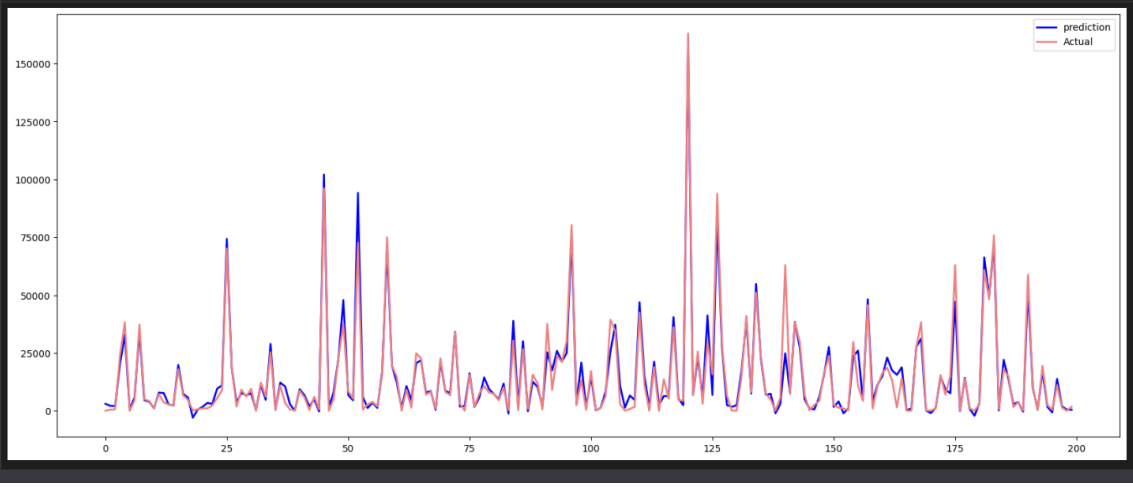
```
[37]
xgb_acc = xgbr.score(X_test,Y_test)*100
print("XGBoost Regressor Accuracy - ",xgb_acc)
```

```
XGBoost Regressor Accuracy -  94.57550886297496
```

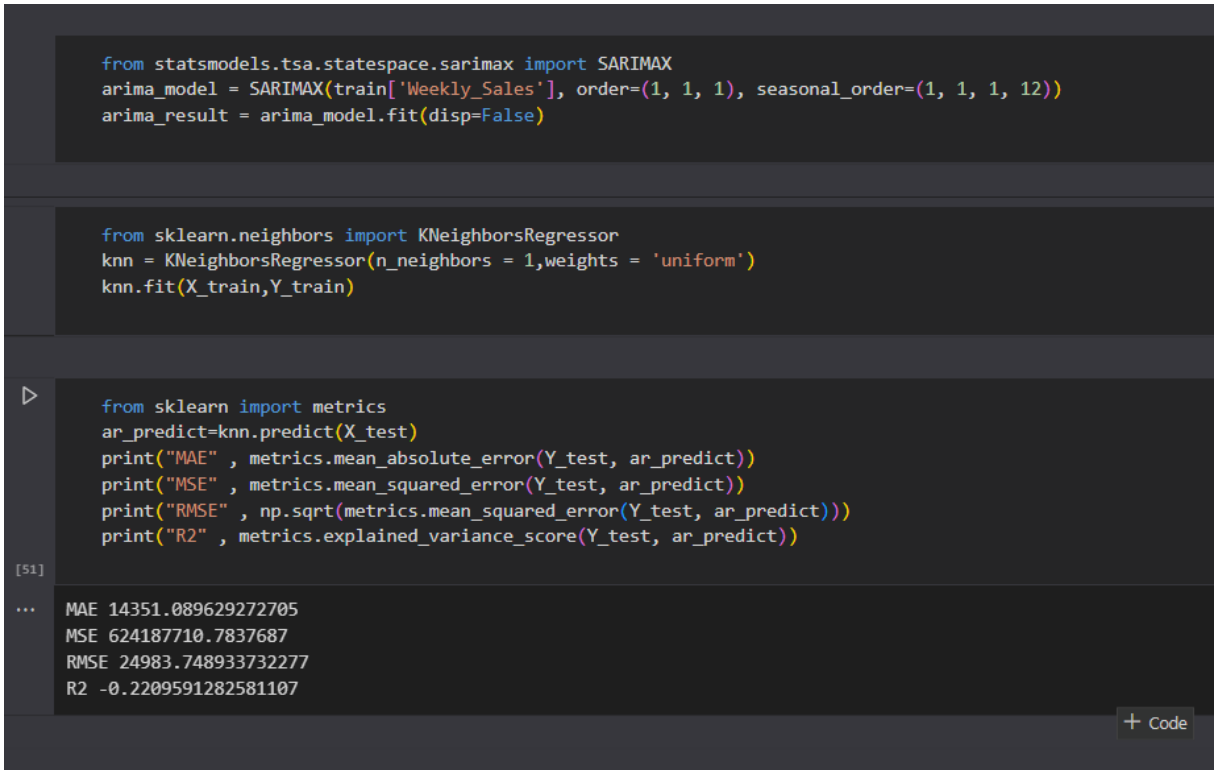
```
[38]
y_pred = xgbr.predict(X_test)
print("MAE" , metrics.mean_absolute_error(Y_test, y_pred))
print("MSE" , metrics.mean_squared_error(Y_test, y_pred))
print("RMSE" , np.sqrt(metrics.mean_squared_error(Y_test, y_pred)))
print("R2" , metrics.explained_variance_score(Y_test, y_pred))
```

```
MAE 2958.031149412797
MSE 27731477.459171012
RMSE 5266.068501184827
R2 0.9457551204707744
```

```
[39]
plt.figure(figsize=(20,8))
plt.plot(xgbr.predict(X_test[:200]), label="prediction", linewidth=2.0,color='blue')
plt.plot(Y_test[:200].values, label="Actual", linewidth=2.0,color='lightcoral')
plt.legend(loc="best")
plt.show()
```



ARIMA



```
from statsmodels.tsa.statespace.sarimax import SARIMAX
arima_model = SARIMAX(train['Weekly_Sales'], order=(1, 1, 1), seasonal_order=(1, 1, 1, 12))
arima_result = arima_model.fit(disp=False)
```

```
from sklearn.neighbors import KNeighborsRegressor
knn = KNeighborsRegressor(n_neighbors = 1,weights = 'uniform')
knn.fit(X_train,Y_train)
```

```
from sklearn import metrics
ar_predict=knn.predict(X_test)
print("MAE" , metrics.mean_absolute_error(Y_test, ar_predict))
print("MSE" , metrics.mean_squared_error(Y_test, ar_predict))
print("RMSE" , np.sqrt(metrics.mean_squared_error(Y_test, ar_predict)))
print("R2" , metrics.explained_variance_score(Y_test, ar_predict))
```

[51]

```
MAE 14351.089629272705
MSE 624187710.7837687
RMSE 24983.748933732277
R2 -0.2209591282581107
```

+ Code

Github & Project Demo Link:

Github Link:

https://github.com/smartinternz02/SI-GuidedProject-599687-1697729828

Project Demo Link:

https://drive.google.com/file/d/1MNgq4ffEVP5KPIx_gr5w3boVsGzxQj5N/view?usp=sharing