Walmart Weekly Sales Prediction & Formula 1 World Championship Prediction

Contents

1. Introduction

This project focuses on building predictive models for two distinct datasets: one with **seasonal data - Walmart Weekly Sales Prediction**, and the other with **non-seasonal data - Formula 1 World Champion Prediction**. Each dataset presents unique challenges and opportunities for analysis, enabling the application of diverse forecasting and performance evaluation techniques.

Walmart Weekly Sales Prediction (Seasonal Data):
This part of the project focuses on forecasting Walmart's sales by analyzing seasonal trends, which are heavily influenced by holidays, promotions, and annual events. Accurate predictions are essential to avoid inventory mismanagement, revenue losses, and investor concerns. The objectives include developing a robust sales forecasting model to optimize inventory, estimate revenue, and support investment planning. Through time series analysis, the project uncovers seasonal patterns, enabling Walmart to make data-driven decisions, streamline operations, and execute targeted campaigns for improved efficiency and financial stability.

Formula 1 Drivers and Teams Performance Prediction (Non-Seasonal Data):
The Formula 1 analysis explores historical data from 1950 to 2024 to evaluate and predict the performance of drivers and teams. Key goals include analyzing trends, identifying factors influencing race outcomes, and understanding the role of strategies, lap times, and team decisions in achieving success. Given Formula 1's data-driven nature, this project provides insights into performance metrics and strategic factors, emphasizing the role of data analysis in car design, race preparation, and competitive success, both on and off the racetrack.

Project Overview

This project bridges the gap between **time-series forecasting** and **performance analysis** by utilizing two diverse datasets: **Walmart Sales Data** for seasonal sales forecasting and **Formula 1 Data** for non-seasonal performance prediction and analysis. The methodology includes comprehensive **data preprocessing** to clean and organize the data, followed by **exploratory data analysis (EDA)** to uncover patterns, trends, and anomalies within each dataset. Subsequently, suitable predictive models are developed and implemented, leveraging statistical methods and time series techniques. The insights derived from these models aim to enhance Walmart's sales management strategies and provide an in-depth evaluation of Formula 1 drivers and team performance.

2. Walmart Weekly Sales Prediction (Seasonal Analysis)

2.1 Problem Statement

Walmart, as a global retail leader, faces significant fluctuations in consumer demand due to seasonal variations, holidays, weather patterns, sales campaigns, and economic factors like inflation and fuel prices. Periods such as Thanksgiving, Christmas, and Black Friday see sharp sales surges, while off-season months witness reduced activity. Misjudging seasonal demand can lead to excess inventory, stockouts, revenue loss, and poor investor confidence. Accurate sales forecasting is crucial for managing inventory, optimizing revenue projections, and making data-driven strategic decisions. This project aims to develop a predictive model using historical data and advanced time-series techniques to identify trends, provide actionable insights, and enable Walmart to optimize performance and drive sustained growth.

2.2 Aim

The aim of this project is to develop a robust predictive model for store sales that will enable Walmart to make data-driven and informed decisions. By leveraging historical sales data and advanced analytical techniques such as time-series analysis and machine learning, the model will provide accurate sales forecasts. This will help Walmart optimize stock arrangement to efficiently meet demand, avoiding overstocking or stockouts. Additionally, the model will support strategic planning for resource allocation, improve revenue projections for precise financial planning, and guide investment strategies to enhance operational efficiency. Accurate forecasts will also aid in workforce planning, ensuring optimal staffing during peak seasons, and inform the timing of promotional campaigns to maximize sales. Ultimately, this predictive model will empower Walmart to enhance customer satisfaction, streamline operations, and drive sustained business growth in a competitive retail environment.

2.3 Dataset Overview

The dataset consists of four files containing Walmart store sales data spanning from **2010 to 2012**, providing a comprehensive view of sales performance across multiple stores, departments, and external factors. The **Stores.csv** file includes data for 45 stores, classified into three types (**A, B, and C**) based on size, along with their respective square footage. The **Features.csv** file provides additional information that influences sales, such as weekly temperature, fuel prices, Consumer Price Index (CPI), unemployment rates, and indicators for holiday weeks, which highlight major events like **Super Bowl**, **Labor Day**, **Thanksgiving**, and **Christmas**. The **Train.csv** file contains weekly sales data for each department within every store, including identifiers for the store, department, date, weekly sales, and a flag for holidays. Similarly, the **Test.csv** file mirrors the structure of the train dataset but excludes the weekly sales column, making it ideal for model validation. The data spans approximately three years with weekly observations, covering up to **99 departments** across the stores, though not all departments appear in every store. External factors such as holidays, fuel prices, CPI, and unemployment play a crucial role in influencing sales trends, while seasonal variations cause significant spikes or dips in demand. This dataset provides a solid foundation for analyzing sales patterns, identifying key drivers, and building predictive models to forecast future sales, enabling Walmart to optimize inventory management, revenue planning, and operational efficiency.

2.4 Methodology

The methodology involves a series of systematic steps to ensure the dataset is clean, well-structured, and ready for building accurate predictive models.

Data Preprocessing:
The raw data underwent cleaning to address missing and null values, which were identified across various parts of the dataset. Missing data was either imputed or removed, ensuring the integrity of the dataset for analysis. Duplicate entries, inconsistencies, and outliers were also handled to maintain data quality.

Exploratory Data Analysis (EDA): Extensive EDA was performed to uncover insights and patterns within the data. Key areas of focus included analyzing the impact of holidays on sales, identifying which holidays had the most significant effect, and examining sales trends over the years to determine periods of peak performance. Insights were derived to identify which stores and departments achieved the highest sales and during which times of the year these peaks occurred. Additionally, seasonal trends and variations were analyzed to understand fluctuations in sales behavior across different store types (A, B, and C) and departments.

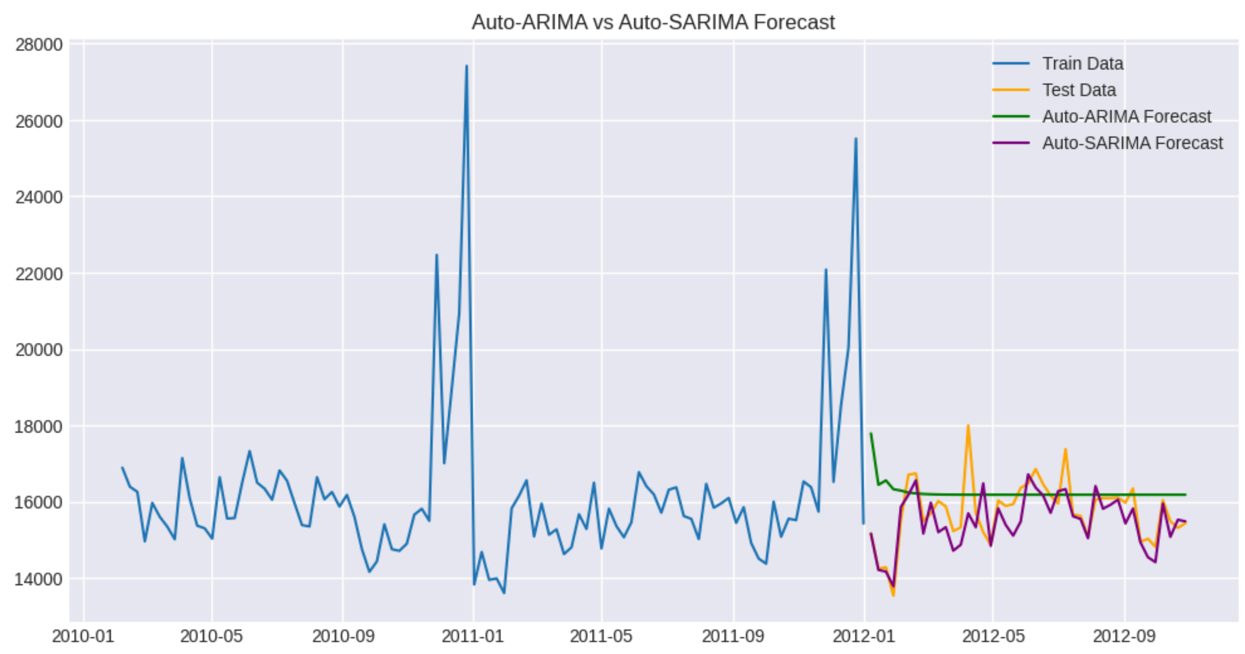
Feature Engineering:
Insights from the exploratory analysis were used to engineer new features that could enhance the predictive power of the model. For instance, holiday indicators, year-on-year sales trends, seasonal lags, and moving averages were added to capture temporal and event-based variations in sales.

Model Selection:
Based on the time-series nature of the data, methods such as ARIMA, SARIMA, and Auto-ARIMA were considered. These models were chosen to capture trends, seasonality, and external influences like holidays or economic indicators (e.g., CPI, fuel prices, and unemployment rates).

Model Evaluation:
To evaluate model performance, metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Weighted Mean Absolute Error (WMAE) were employed. Comparisons between models were conducted to select the most accurate and reliable forecasting approach. The evaluation focused on how well each model captured sales trends and seasonal variations while minimizing prediction errors.

This methodology ensures a comprehensive approach to data preparation, feature extraction, model building, and performance evaluation, leading to a robust predictive model that addresses the project's objectives.

2.5 Results



Final Results and Model Comparison

The evaluation of the Auto-ARIMA and Auto-SARIMA models highlights the superior performance of the **Auto-SARIMA model**. The performance metrics for both models are summarized below:

- **Auto-ARIMA Results**:
    - Mean Squared Error (MSE): **1,049,606.66**
    - Root Mean Squared Error (RMSE): **1,024.50**
    - Weighted Mean Absolute Error (WMAE): **594.47**
- **Auto-SARIMA Results**:
    - Mean Squared Error (MSE): **329,967.81**
    - Root Mean Squared Error (RMSE): **574.43**
    - Weighted Mean Absolute Error (WMAE): **374.39**

The results clearly demonstrate that **Auto-SARIMA outperforms Auto-ARIMA** across all evaluation metrics, particularly in **RMSE** and **WMAE**. The significantly lower RMSE of **574.43** for Auto-SARIMA, compared to **1,024.50** for Auto-ARIMA, indicates more accurate predictions. This improvement can be attributed to Auto-SARIMA's ability to effectively capture **seasonal patterns** in the data, which were evident in the sales trends.

The forecast visualization further reinforces this conclusion, as Auto-SARIMA predictions closely follow the actual test data, while Auto-ARIMA exhibits larger deviations. In conclusion, **Auto-SARIMA** is the superior model for this dataset, delivering more precise and reliable forecasts for future sales.

2.6 Conclusion

The analysis highlights that holidays, particularly **Thanksgiving** and **Christmas**, are critical drivers of Walmart's sales peaks, with sharp increases observed toward the end of the year. **Type A stores** consistently achieve the highest average weekly sales during these periods, followed by Type B and C stores. Clear seasonal trends in sales patterns emphasize the need for accurate demand forecasting to optimize inventory and operations. To address this, Walmart should increase stock levels during peak holidays to prevent stockouts, focus promotional campaigns on high-impact holidays targeting Type A and B stores, and enhance staffing and operational capacity to ensure smooth customer experiences. Implementing robust forecasting models like SARIMA can further improve demand anticipation, enabling Walmart to make data-driven decisions that optimize resources, enhance customer satisfaction, and drive sustained growth.

3. Formula 1 World Championship Prediction (Non-Seasonal Analysis)

3.1 Problem Statement

The objective of this project is to analyze the performance of **Formula 1 teams and drivers** to predict the future **World Champion** with the highest points for upcoming seasons. This task involves addressing key challenges, such as the variability in team performance, driver consistency, and dynamic race conditions over time. To achieve accurate predictions, extensive **data preprocessing** and **exploratory data analysis (EDA)** were conducted to clean, transform, and extract meaningful insights from historical race data.

The project aims to answer critical questions, including:

- Which drivers and teams demonstrate the highest probability of winning the championship?
- What key factors influence the overall championship points?
- How can historical trends be used to forecast future performance in such a highly competitive sport?

By addressing these challenges, the project strives to provide reliable predictions and actionable insights into **Formula 1 team and driver performance**, contributing to a better understanding of the factors that drive success in this data-driven sport.

3.2 Aim

The objective of this Formula 1 prediction analysis is to evaluate team-wise and driver-wise performance to predict the World Champion with the highest points for upcoming seasons. To achieve this, we conducted an extensive analysis of historical Formula 1 data, including key performance indicators across drivers, teams (constructors), and circuits. A variety of metrics and patterns were studied to gain actionable insights, as depicted in the detailed sections. This involved analyzing race-specific data such as **lap times**, **race results**, and **altitude impact on engine performance**, along with **driver performance trajectory** and **constructor reliability improvements** over time. Additionally, we examined **qualifying vs. race performance**, the **impact of regulation changes**, and how **age and experience factors** influence driver success.

The study also explored broader trends, such as **nationality patterns**, to identify competitive advantages across regions and historical trends in **circuits and race analysis**. By incorporating these detailed analyses, the project aims to develop a predictive framework capable of identifying the key drivers and teams poised to succeed in the upcoming championships. This holistic approach combines data-driven insights with advanced statistical techniques, enabling precise performance predictions and offering valuable contributions to strategic decisions in the world of Formula 1 racing.

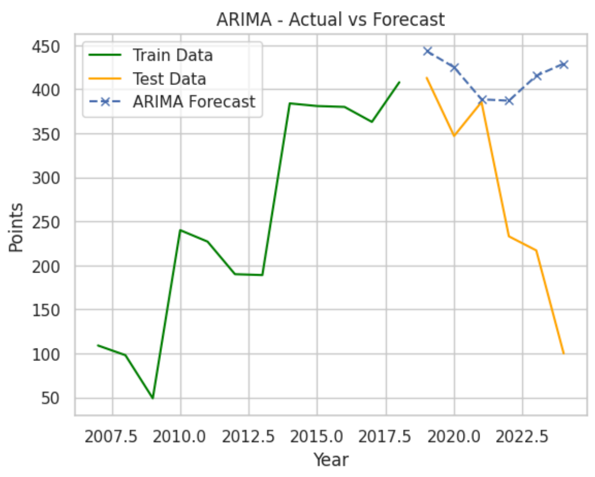
3.3 Dataset Overview

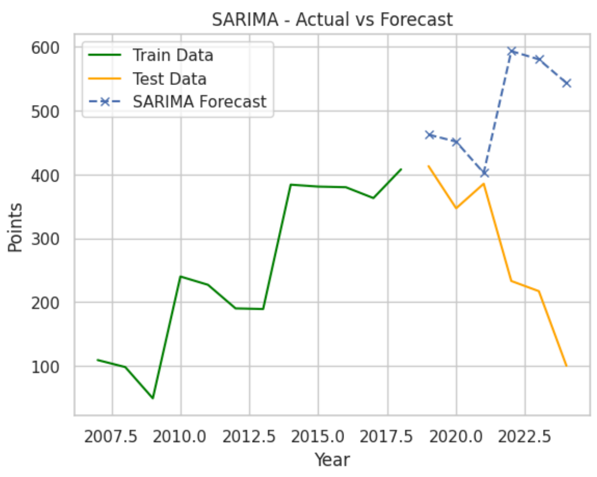
The Formula 1 dataset spans 75 years (1950–2024), covering 1,144 races, 859 drivers, and 212 teams, making it a rich resource for performance analysis. It includes 14 files capturing key aspects such as race details, driver and team standings, qualifying results, pit stops, lap times, and championship outcomes. This comprehensive data provides insights into driver and team trajectories, race strategies, and performance trends. By analyzing historical metrics, the dataset supports the development of predictive models to forecast championship results and evaluate critical performance indicators.

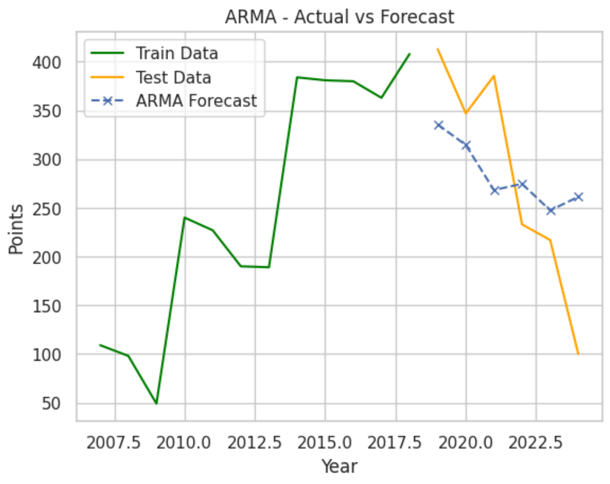
3.4 Methodology

To predict the points scored by a driver in upcoming seasons based on historical performance, we adopted a systematic time series forecasting approach. The data was preprocessed by cleaning missing values and formatting it to ensure consistency, with a focus on driver points and season-wise performance. To assess stationarity, a critical requirement for time series models, we applied the **Augmented Dickey-Fuller (ADF) test** and implemented necessary transformations such as differencing, logging, and shifting where required. Using the **Box-Jenkins methodology**, we developed and implemented the **ARMA** (Auto-Regressive Moving Average) and **Auto-ARIMA** models. The Auto-ARIMA model was particularly advantageous as it automated the selection of optimal values for p (AR term), d (differencing), and q (MA term), ensuring accuracy and reducing manual effort. Both models were evaluated using performance metrics such as Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) to validate their accuracy. Through rigorous model comparison and validation, we identified trends and patterns in driver performance, enabling reliable forecasts for upcoming seasons based on historical data.

3.5 Results







The Formula 1 World Championship prediction analysis demonstrates the effectiveness of time series models in forecasting driver points based on historical performance. Three models were implemented: ARIMA, SARIMA, and ARMA, with a focus on comparing their predictive accuracy using RMSE as the evaluation metric. The ARMA model emerged as the best-performing model, achieving the lowest **RMSE of 90.78**, compared to ARIMA (**RMSE = 172.47**) and SARIMA (**RMSE = 280.61**). This highlights ARMA's capability to capture the underlying patterns in the driver points data efficiently.

From the plots, it is evident that ARMA forecasts align more closely with the observed test data, while ARIMA and SARIMA showed relatively higher deviations. ARMA's superior performance can be attributed to the dataset's non-seasonal nature, where simpler models like ARMA are more effective than seasonal alternatives.

Overall, this analysis confirms that historical performance is a reliable predictor for future outcomes, enabling teams and stakeholders to anticipate championship results accurately. By leveraging the ARMA model, Formula 1 teams can make data-driven decisions, optimize strategies, and enhance their competitiveness for upcoming seasons.

3.6 Conclusion

The Formula 1 performance analysis highlights key insights into driver and team performance, as well as external factors like altitude affecting race outcomes. Using ARMA and Auto-ARIMA models, we successfully forecasted individual driver points up to 2030, demonstrating that historical performance is a strong predictor of future results.

A notable finding was the impact of altitude: Mercedes excelled at high altitudes, Ferrari performed best at sea level, Red Bull showed consistent adaptability, while McLaren and Williams struggled at higher elevations. These insights emphasize the importance of altitude-specific strategies, such as fine-tuning car setups, optimizing aerodynamics, and enhancing driver adaptability through training and preparation.

For teams and stakeholders, leveraging predictive models and data-driven insights can optimize resource allocation, improve pre-race planning, and enhance overall performance. This analysis underscores the value of integrating historical data and environmental considerations to gain a competitive edge in Formula 1 racing.

4. Overall Conclusion

This project successfully tackled two distinct prediction tasks: **Walmart Weekly Sales Prediction** and the **Formula 1 World Championship Prediction**, showcasing the effectiveness of data-driven forecasting techniques. For Walmart, time series analysis highlighted strong seasonal trends, particularly during holidays like Thanksgiving and Christmas, which significantly drive sales spikes. The **Auto-SARIMA model** outperformed Auto-ARIMA, providing more accurate forecasts and equipping Walmart with actionable insights for optimizing inventory management, enhancing revenue projections, and improving strategic planning.

In the **Formula 1 analysis**, driver points were forecasted for upcoming seasons using historical performance data, with **ARMA** emerging as the best-performing model, achieving the lowest RMSE compared to ARIMA and SARIMA. The analysis also revealed the **impact of altitude** on team and driver performance: Mercedes excelled at high altitudes, Ferrari performed best at sea level, and Red Bull showcased adaptability across all conditions. These insights offer valuable recommendations for race strategy optimization, car tuning, and driver preparation.

Overall, this project highlights the power of predictive modeling in addressing real-world challenges. By uncovering seasonal sales trends for Walmart and forecasting championship performance for Formula 1, the project demonstrates how data-driven techniques enable organizations and teams to make informed decisions, optimize performance, and achieve sustained success.

5. References

1. Walmart Sales Forecast Dataset Source - Walmart Sales Forecast

Description: This dataset contains historical sales data for 45 Walmart stores located in different regions, including weekly sales, store information, and additional features like temperature, fuel prices, and markdown events.

2. Formula 1 World Championship Dataset (1950-2024) Source - Formula 1 World Championship 1950-2024

Description: This dataset includes Formula 1 race results, driver standings, constructor standings, and other related data from 1950 to 2024.

Appendix

A. Data Dictionary

A.1 Walmart Dataset

Variable	**Description**	**Data Type**
Store	Store number	Integer
Date	Week of sales	Date
Weekly_Sales	Sales for the given store	Float
IsHoliday	Whether the week is a special holiday week	Boolean
Temperature	Average temperature in the region	Float
Fuel_Price	Cost of fuel in the region	Float
MarkDown1-5	Promotional markdowns	Float

CPI	Consumer Price Index	Float
Unemployment	Unemployment rate	Float
Type	Type of store	String
Size	Size of store	Integer

A.2 Formula 1 Dataset

Variable	**Description**	**Data Type**
driverId	Unique identifier for each driver	Integer
constructorId	Unique identifier for each constructor	Integer
circuitId	Unique identifier for each circuit	Integer
position	Finishing position of the driver	Integer

points	Championship points awarded	Integer
lap	Lap number	Integer
time	Lap time	Time
milliseconds	Lap time in milliseconds	Integer

B. Data Sources

1. Walmart Sales Forecast Dataset

Source: Kaggle - Walmart Sales Forecast

Description: This dataset contains historical sales data for 45 Walmart stores located in different regions, including weekly sales, store information, and additional features like temperature, fuel prices, and markdown events.

2. Formula 1 World Championship Dataset (1950-2024)

Source: Kaggle - Formula 1 World Championship 1950-2024

Description: This dataset includes Formula 1 race results, driver standings, constructor standings, and other related data from 1950 to 2024.

C. Data Preprocessing

C.1 Walmart Sales Forecast Dataset

```
# Merging datasets
df = df_train.merge(df_features, on=['Store', 'Date'],
how='inner').merge(df_store, on=['Store'], how='inner')
# Handling missing values (example for MarkDown columns)
```

```
df['MarkDown1'].fillna(df['MarkDown1'].mean(), inplace=True)
df['MarkDown2'].fillna(df['MarkDown2'].mean(), inplace=True)
df['MarkDown3'].fillna(df['MarkDown3'].mean(), inplace=True)
df['MarkDown4'].fillna(df['MarkDown4'].mean(), inplace=True)
df['MarkDown5'].fillna(df['MarkDown5'].mean(), inplace=True)

# Converting Date column to datetime
df['Date'] = pd.to_datetime(df['Date'])

# Setting Date as index
df.set_index('Date', inplace=True)
```

C.2 Formula 1 World Championship Dataset

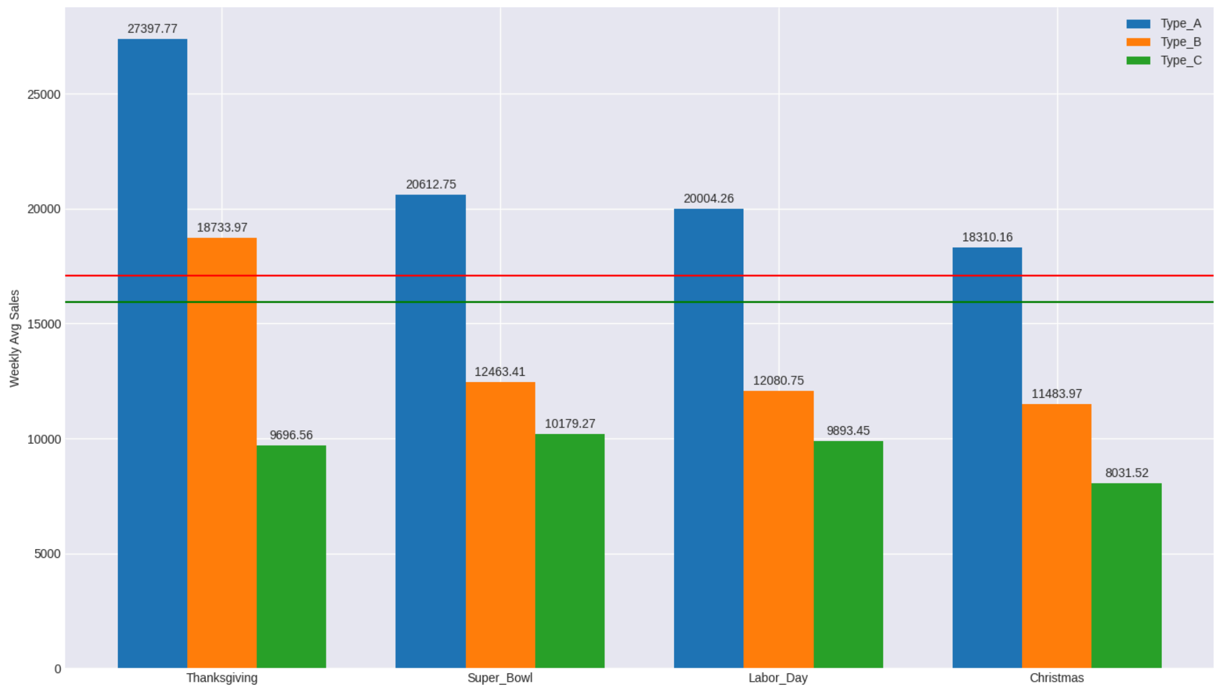
```
# Merging datasets
df = df_train.merge(df_features, on=['Store', 'Date'],
how='inner').merge(df_store, on=['Store'], how='inner')
# Handling missing values (example for MarkDown columns)
df['MarkDown1'].fillna(df['MarkDown1'].mean(), inplace=True)
df['MarkDown2'].fillna(df['MarkDown2'].mean(), inplace=True)
df['MarkDown3'].fillna(df['MarkDown3'].mean(), inplace=True)
df['MarkDown4'].fillna(df['MarkDown4'].mean(), inplace=True)
df['MarkDown5'].fillna(df['MarkDown5'].mean(), inplace=True)

# Converting Date column to datetime
df['Date'] = pd.to_datetime(df['Date'])

# Setting Date as index
df.set_index('Date', inplace=True)
# Loading CSV files
circuits = pd.read_csv('circuits.csv')
constructor_results = pd.read_csv('constructor_results.csv')
constructor_standings = pd.read_csv('constructor_standings.csv')
constructors = pd.read_csv('constructors.csv')
driver_standings = pd.read_csv('driver_standings.csv')
drivers = pd.read_csv('drivers.csv')
lap_times = pd.read_csv('lap_times.csv')
pit_stops = pd.read_csv('pit_stops.csv')
qualifying = pd.read_csv('qualifying.csv')
races = pd.read_csv('races.csv')
results = pd.read_csv('results.csv')
seasons = pd.read_csv('seasons.csv')
```

```
sprint_results = pd.read_csv('sprint_results.csv')
status = pd.read_csv('status.csv')

# Converting date columns to datetime (example for races dataset)
races['date'] = pd.to_datetime(races['date'])
```

D. Exploratory Data Analysis

D.1 Walmart Sales Forecast Dataset (Non-seasonal)

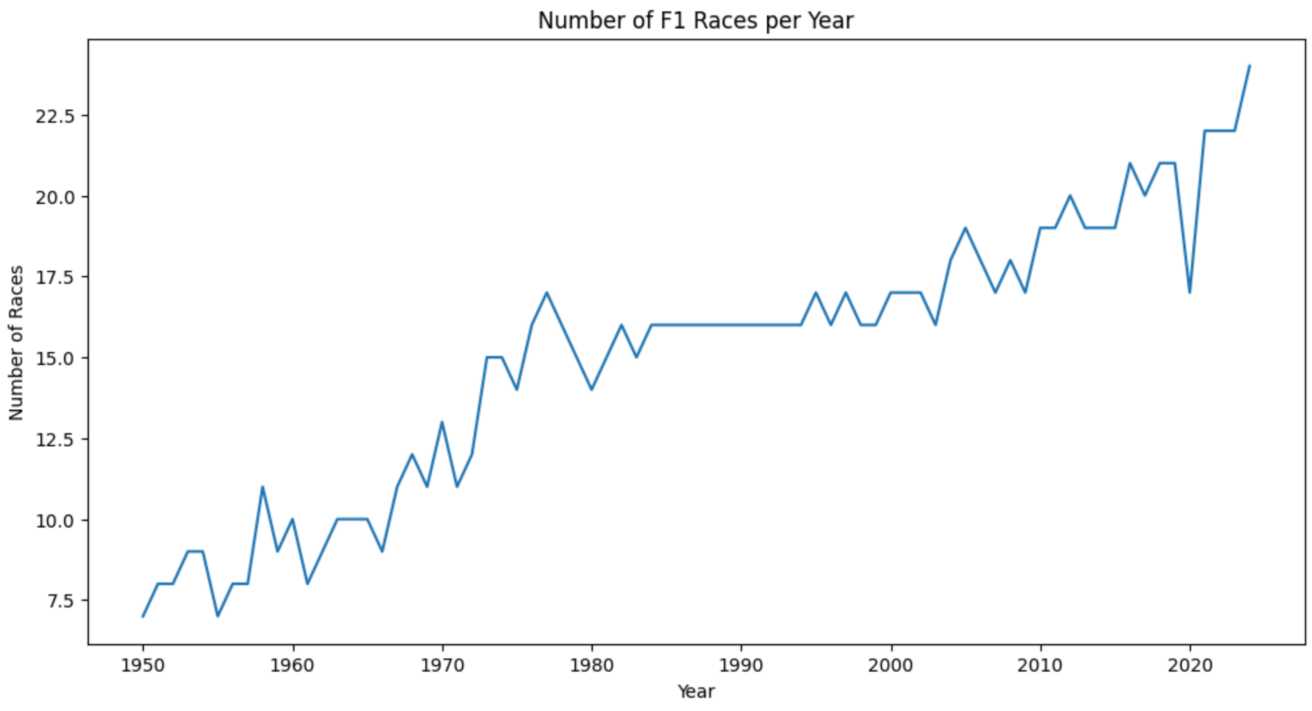
```
# Plotting avg weekly sales according to holidays by types
plt.style.use('seaborn-v0_8-darkgrid')
labels = ['Thanksgiving', 'Super_Bowl', 'Labor_Day', 'Christmas']
A_means = [27397.77, 20612.75, 20004.26, 18310.16]
B_means = [18733.97, 12463.41, 12080.75, 11483.97]
C_means = [9696.56, 10179.27, 9893.45, 8031.52]
x = np.arange(len(labels)) # the label locations
width = 0.25 # the width of the bars
fig, ax = plt.subplots(figsize=(14, 8))
rects1 = ax.bar(x - width, A_means, width, label='Type_A')
rects2 = ax.bar(x, B_means, width, label='Type_B')
rects3 = ax.bar(x + width, C_means, width, label='Type_C')
# Add some text for labels, title and custom x-axis tick labels, etc.
ax.set_ylabel('Weekly Avg Sales')
ax.set_xticks(x)
ax.set_xticklabels(labels)
ax.legend()
def autolabel(rects):

for rect in rects:
height = rect.get_height()
ax.annotate('{}'.format(height),
xy=(rect.get_x() + rect.get_width() / 2, height),
xytext=(0, 3), # 3 points vertical offset
textcoords="offset points",
ha='center', va='bottom')
autolabel(rects1)
autolabel(rects2)
autolabel(rects3)
plt.axhline(y=17094.30, color='r') # holidays avg
plt.axhline(y=15952.82, color='green') # not-holiday avg
fig.tight_layout()
plt.show()
```

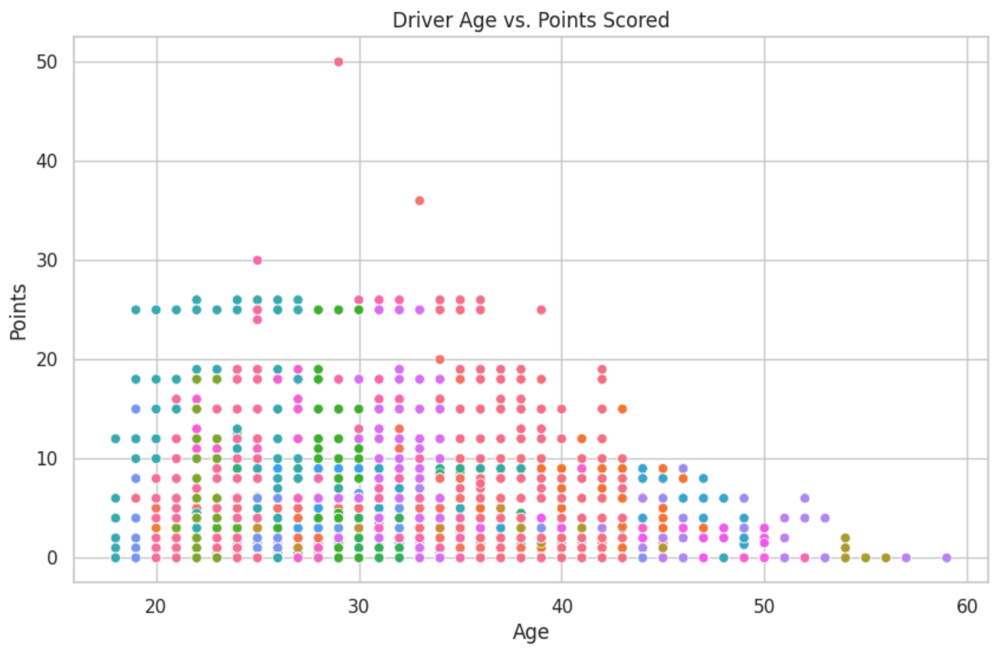


D.2 Formula 1 World Championship Dataset (Seasonal)

```
# Number of races per year
races_per_year = races.groupby('year').size()
plt.figure(figsize=(15, 6))
races_per_year.plot(kind='bar')
plt.title('Number of Races per Year')
plt.xlabel('Year')
plt.ylabel('Number of Races')
plt.show()

# Top 10 drivers by wins
top_drivers=results[results['position']=='1'].groupby('driverId').size().sort_val
ues(ascending=False).head(10)
plt.figure(figsize=(12, 6))
top_drivers.plot(kind='bar')
```

```
plt.title('Top 10 Drivers by Number of Wins')
plt.xlabel('Driver ID')
plt.ylabel('Number of Wins')
plt.show()
```



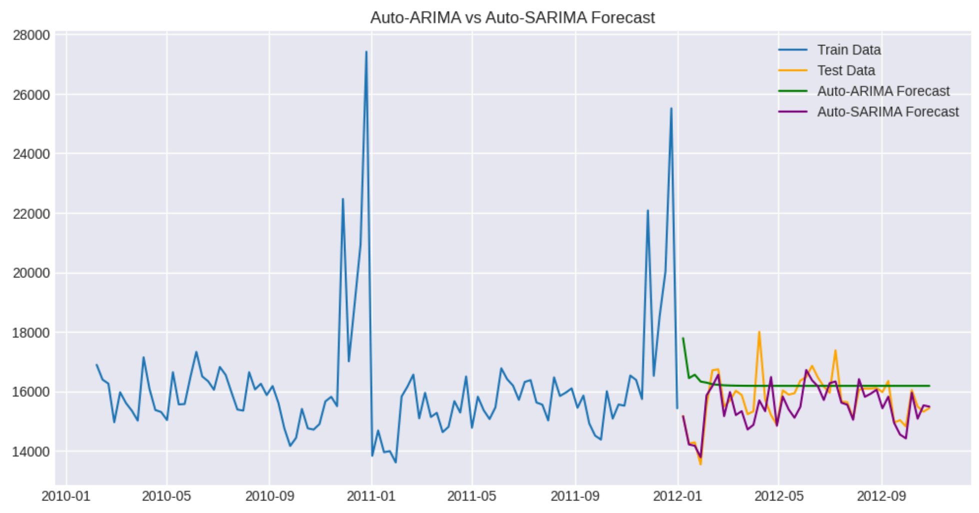


E. Model Implementation

E.1 Walmart

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from pmdarima import auto_arima
from sklearn.metrics import mean_squared_error, mean_absolute_error

df = pd.read_csv("clean_data.csv", parse_dates=['Date'], index_col='Date')
df_week = df['Weekly_Sales'].resample('W').mean().dropna()

train_size = int(len(df_week) * 0.7)
train, test = df_week[:train_size], df_week[train_size:]

def calculate_wmae(actual, predicted, weights):
    return np.sum(weights * np.abs(actual - predicted)) / np.sum(weights)

#Auto-ARIMA
print("Fitting Auto-ARIMA Model...")
auto_arima_model = auto_arima(train,
                              seasonal=False, stepwise=True, trace=True,
                              error_action="ignore", suppress_warnings=True,
                              information_criterion='aic')

auto_arima_forecast = auto_arima_model.predict(n_periods=len(test))
mse_auto_arima = mean_squared_error(test, auto_arima_forecast)
rmse_auto_arima = np.sqrt(mse_auto_arima)
wmae_auto_arima  =  calculate_wmae(test,  auto_arima_forecast,  weights=np.arange(1,
len(test)+1))

#SARIMA
print("\nFitting Auto-SARIMA Model...")
auto_sarima_model = auto_arima(train,
                               seasonal=True, m=52, stepwise=True, trace=True,
                               error_action="ignore", suppress_warnings=True,
                               information_criterion='aic')

#Auto-SARIMA
auto_sarima_forecast = auto_sarima_model.predict(n_periods=len(test))
```

```
mse_auto_sarima = mean_squared_error(test, auto_sarima_forecast)
rmse_auto_sarima = np.sqrt(mse_auto_sarima)
wmae_auto_sarima  =  calculate_wmae(test,  auto_sarima_forecast,  weights=np.arange(1,
len(test)+1))

plt.figure(figsize=(12, 6))
plt.plot(train, label='Train Data')
plt.plot(test, label='Test Data', color='orange')
plt.plot(test.index,     auto_arima_forecast,     label='Auto-ARIMA     Forecast',
color='green')
plt.plot(test.index,     auto_sarima_forecast,     label='Auto-SARIMA     Forecast',
color='purple')
plt.title("Auto-ARIMA vs Auto-SARIMA Forecast")
plt.legend()
plt.show()

print("Evaluation Metrics Comparison:")
print("\nAuto-ARIMA:")
print(f"MSE: {mse_auto_arima:.2f}")
print(f"RMSE: {rmse_auto_arima:.2f}")
print(f"WMAE: {wmae_auto_arima:.2f}")

print("\nAuto-SARIMA:")
print(f"MSE: {mse_auto_sarima:.2f}")
print(f"RMSE: {rmse_auto_sarima:.2f}")
print(f"WMAE: {wmae_auto_sarima:.2f}")

if rmse_auto_arima < rmse_auto_sarima:
    print("\nAuto-ARIMA performs better based on RMSE.")
else:
    print("\nAuto-SARIMA performs better based on RMSE.")
```

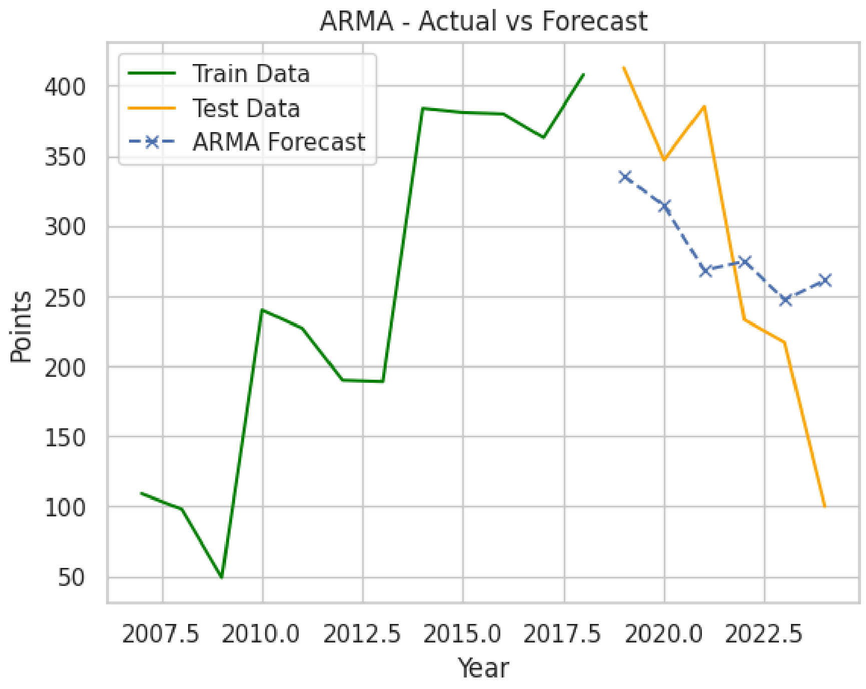


E.2 Formula 1

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from statsmodels.tsa.stattools import adfuller
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.tsa.statespace.sarimax import SARIMAX
from statsmodels.tsa.arima.model import ARIMA
from pmdarima import auto_arima
from sklearn.metrics import mean_squared_error

results_file_path = '/content/results.csv'
races_file_path = '/content/races.csv'
results_df = pd.read_csv(results_file_path)
races_df = pd.read_csv(races_file_path)

if 'date' in races_df.columns:
    races_df['year'] = pd.to_datetime(races_df['date']).dt.year

results_with_year  =  results_df.merge(races_df[['raceId',  'year']],
on='raceId', how='left')
```

```
driver_id = 1
driver_points          =          results_with_year.groupby(['driverId',
'year'])['points'].sum().reset_index()
driver_data = driver_points[driver_points['driverId'] == driver_id]
driver_data = driver_data.set_index('year')['points'].sort_index()

train_size = int(len(driver_data) * 0.7)
train_data = driver_data.iloc[:train_size]
test_data = driver_data.iloc[train_size:]

def evaluate_model(model_name, model, train, test, **kwargs):
    print(f"\nTraining {model_name}...")
    if model_name == 'SARIMA':
        model_fit = model(train, **kwargs).fit(disp=False)
    else:
        model_fit = model(train, **kwargs).fit()

    forecast = model_fit.forecast(steps=len(test))
    rmse = np.sqrt(mean_squared_error(test, forecast))
    print(f"{model_name} RMSE: {rmse:.4f}")

    plt.plot(train, label='Train Data', color='green')
    plt.plot(test.index, test, label='Test Data', color='orange')
      plt.plot(test.index,  forecast,  label=f'{model_name}  Forecast',
linestyle='--', marker='x')
    plt.title(f"{model_name} - Actual vs Forecast")
    plt.xlabel("Year")
    plt.ylabel("Points")
    plt.legend()
    plt.show()

    return rmse

#Comparing

arima_rmse = evaluate_model(
    "ARIMA", ARIMA, train_data, test_data, order=(2, 1, 2)
)
```

```
sarima_rmse = evaluate_model(
       "SARIMA",  SARIMAX,  train_data,  test_data,  order=(2,  1,  2),
seasonal_order=(1, 1, 1, 12)
)

arma_rmse = evaluate_model(
    "ARMA", ARIMA, train_data, test_data, order=(2, 0, 2)
)

model_rmse = {
    'ARIMA': arima_rmse,
    'SARIMA': sarima_rmse,
    'ARMA': arma_rmse
}

best_model = min(model_rmse, key=model_rmse.get)
print("\nModel Performance Comparison:")
for model, rmse in model_rmse.items():
    print(f"{model}: RMSE = {rmse:.4f}")

print(f"\nBest   Performing   Model:   {best_model}   with   RMSE   =
{model_rmse[best_model]:.4f}")
```


ARMA - Actual vs Forecast
Train Data
Test Data
ARMA Forecast
Points
Year