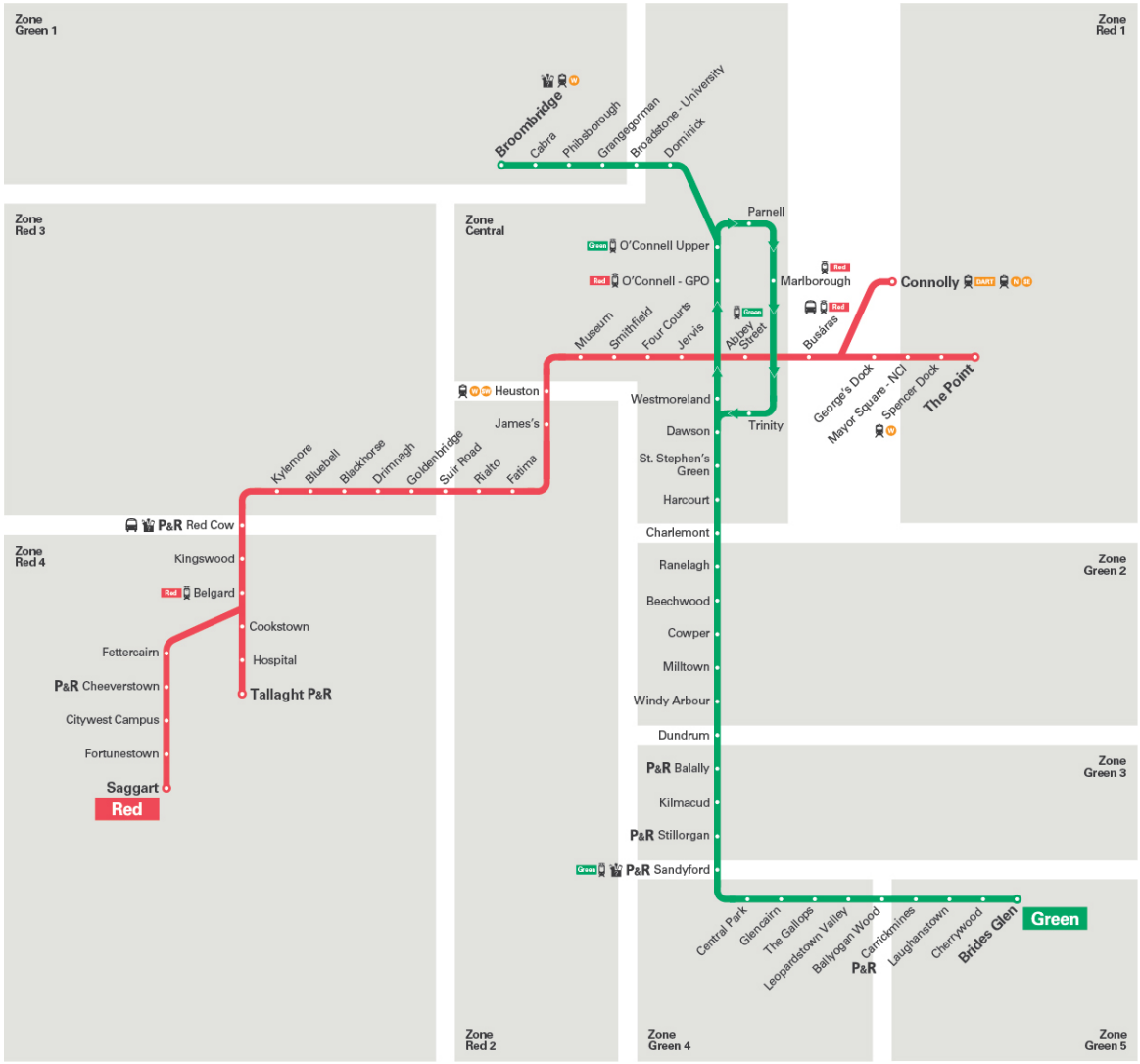
Final Project - Data Analysis on Luas

Ujwal Mojidra [24214941]

Table of contents

Introduction to the Project

In this project, we will analyze the performance of Dublin's **Luas Green Line** and **Red Line** using a comprehensive dataset. The dataset contains **monthly passenger data** spanning from **2018 to December 2023**, providing a robust basis for trend analysis and forecasting. This data encompasses key metrics such as **total passengers** and **average passengers per month** for each line, allowing for an in-depth examination of passenger trends over the six-year period.



Scope of Data set:

The dataset includes:

- **Time Range**: Monthly data from **January 2018 to December 2023**.
- **Lines Analyzed**:
  - **Green Line**: Extending from **Broombridge** to **Brides Glen**, covering central and southern Dublin suburbs.
  - **Red Line**: Connecting **The Point** in the Docklands to **Tallaght** and **Saggart**, serving Dublin's city centre and western suburbs.

- **Key Metrics**:
  - **Total Passengers**: Aggregate passenger count for each line per year.
  - **Average Passengers**: Average monthly passenger numbers, useful for understanding year-on-year trends and seasonal variations.

Contextualization with LUAS Map

To better contextualize the analysis, the accompanying **LUAS Stops & Zones Map**[1] visually represents the areas serviced by both lines. The Green Line predominantly serves southern and central zones, connecting key residential and commercial areas like **Dundrum**, **Sandyford**, and **St. Stephen's Green**. Meanwhile, the Red Line facilitates transit between eastern Docklands and western suburban hubs, passing through major stations such as **Connolly**, **Heuston**, and **Red Cow**. This spatial representation of the LUAS network will support our understanding of how passenger patterns align with significant zones and stops.

Purpose of the Analysis

The primary objectives of this project include:

1. **Identifying trends** in passenger traffic for both the Green and Red Lines.
2. **Forecasting passenger demand** for future periods using models such as **TBATS** for the Green Line and **ARIMA** for the Red Line.
3. **Highlighting critical regions** of interest, such as major stops and peak zones, to provide actionable insights into passenger flow and demand.

By combining data-driven analysis with spatial understanding from the LUAS map, this project aims to provide a clear and insightful assessment of passenger trends, uncovering patterns that can inform future service optimizations and infrastructure planning.

---

[1] Luas Map & Zone: https://luas.ie/map/

Libraries and packages for the Project.

```
library(readr)
library(dplyr)
library(tidyr)
library(ggplot2)
library(tibble)
library(stringr)
library(knitr)
```

In this project, we use the following libraries:

- **`readr`**: For reading data files.
- **`dplyr`**: For data manipulation (filtering, summarizing, etc.).
- **`tidyr`**: For restructuring and tidying data.
- **`ggplot2`**: For creating visualizations.
- **`tibble`**: For modern, easy-to-use data frames.
- **`stringr`**: For handling and manipulating text data.
- **`knitr`**: For generating dynamic reports.
- **`forecast`**[2]: For time series forecasting models like *ARIMA and TBATS*.

The **`forecast`** package is required **only for Part 2** of this project. To ensure it is available, run the following command:

```
#For Part-2:
#Install the forecast package if not already installed

if (!require(forecast)) {
  install.packages("forecast")
}
library(forecast)
```

[2]Forecast Package: https://cran.r-project.org/web/packages/forecast/index.html

Part 1: Manipulation

Load `luas_data` dataset.

This dataset is available from the Central Statistics Office.[3]

```
luas_data <- read_csv("luas_og.csv", show_col_types = FALSE) %>% as_tibble()

# Remove column "Unit"
luas_data <- luas_data %>%
  select(-UNIT)
```

Organize the Dataset by Year and Visualize Trends.

In this step, we will restructure the dataset into a year-wise format to facilitate a comparison between the Red Line and Green Line for each year from 2018 to 2023. A summary table will be created, and based on this table, we will generate a line plot to visualize the trends effectively.

```
# Year-wise summary for each LuasType
yearly_comparison <- luas_data %>%
  group_by(Year, LuasType) %>%
  summarise(
    Total_Passengers = sum(VALUE, na.rm = TRUE),
    Average_Passengers = mean(VALUE, na.rm = TRUE)
  ) %>% arrange(Year, LuasType)
# Print the year-wise summary using kable
kable(yearly_comparison, caption = "Year-wise Summary for Each LuasType",
      format = "html",
      col.names = c("Year", "Luas Type", "Total Passengers", "Average Passengers"))
```

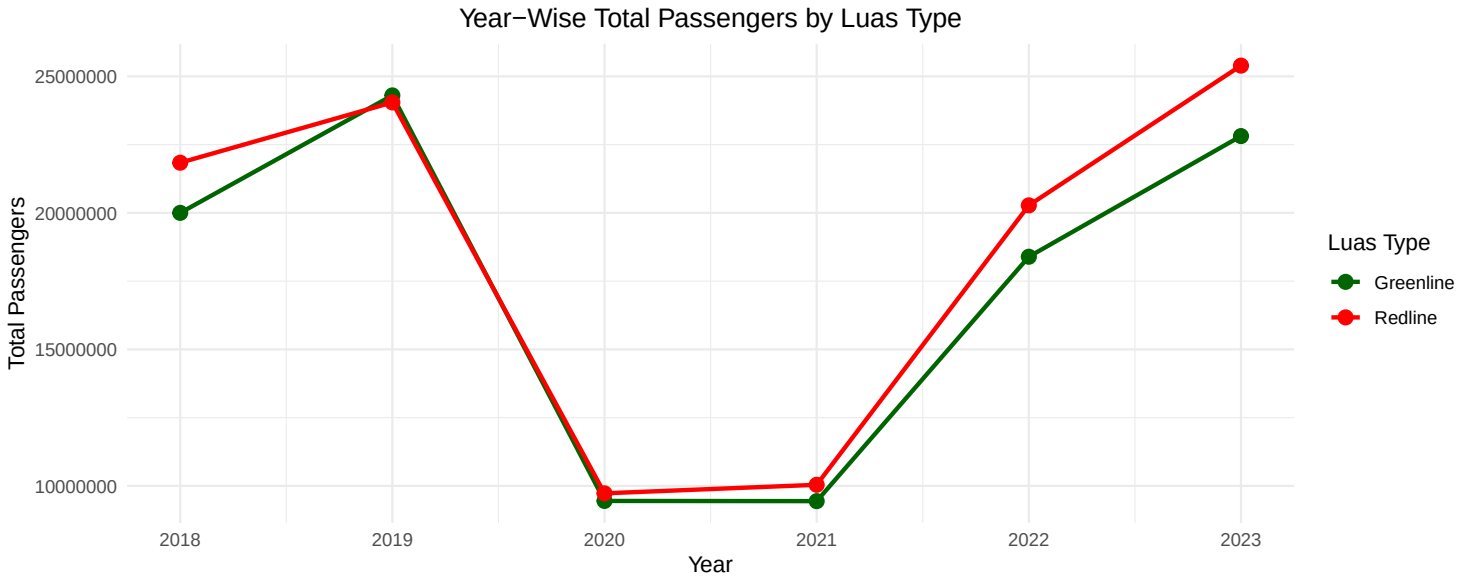
Table 1: Year-wise Summary for Each LuasType

Year	Luas Type	Total Passengers	Average Passengers
2018	Greenline	19999699	1666641.6
2018	Redline	21837267	1819772.2
2019	Greenline	24301487	2025123.9
2019	Redline	24045744	2003812.0
2020	Greenline	9448868	787405.7
2020	Redline	9727189	810599.1
2021	Greenline	9441025	786752.1
2021	Redline	10040293	836691.1
2022	Greenline	18392801	1532733.4
2022	Redline	20275074	1689589.5
2023	Greenline	22812053	1901004.4
2023	Redline	25393164	2116097.0

With the dataset organized by year, we can create a line graph to visually analyze the observed trends.

[3]The Luas data set for the project (Year 2018-2023): https://data.cso.ie/table/TOA11

```r
options(scipen = 101)
ggplot(yearly_comparison, aes(x = Year, y = Total_Passengers, color = LuasType, group = LuasType)) +
  geom_line(size = 1) +
  geom_point(size = 3) +
  scale_color_manual(
    values = c("Greenline" = "darkgreen", "Redline" = "red")
  ) +
  labs(
    title = "Year-Wise Total Passengers by Luas Type",
    x = "Year",
    y = "Total Passengers",
    color = "Luas Type"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5)
  )
```



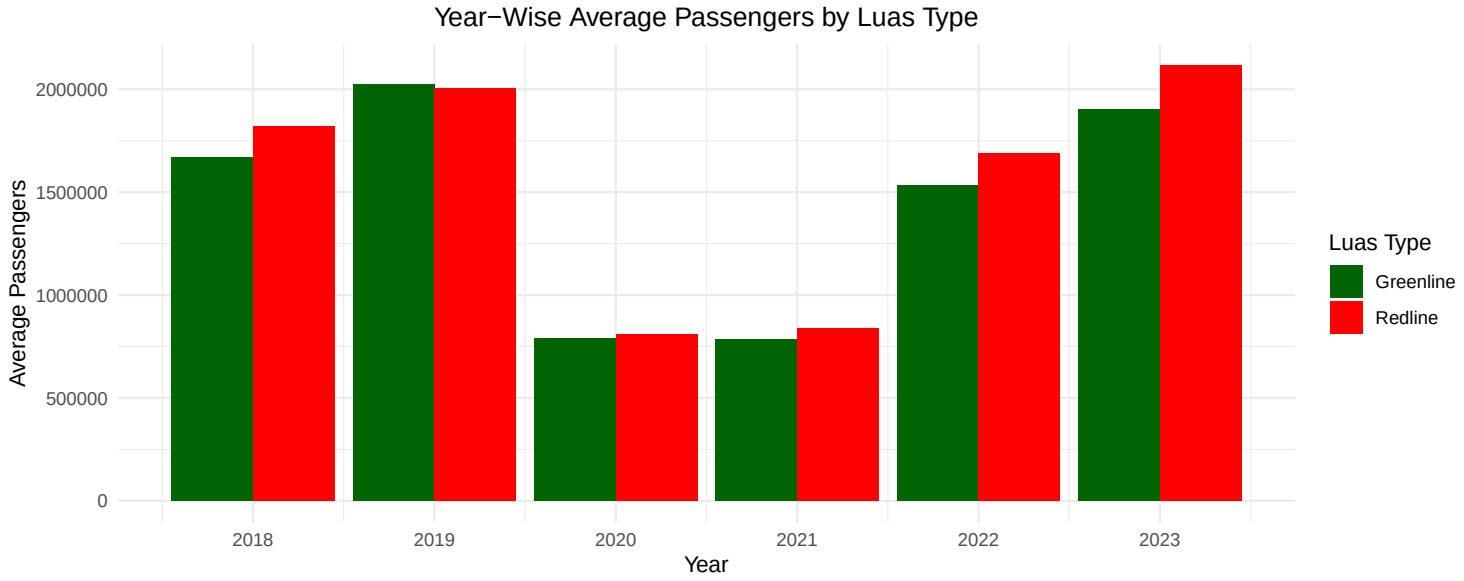
The graph represents the year-wise total passenger counts for the Luas Green Line and Red Line from 2018 to 2023. Below are the key observations:

1. **Impact of COVID-19 (2020-2021):**
   Both lines experienced a significant drop in passenger numbers during 2020 and 2021. This decline corresponds to the COVID-19 pandemic, which led to widespread restrictions, reduced travel, and a shift to remote work, affecting public transportation usage.
2. **Pre-Pandemic Trends (2018-2019):**
   In 2018 and 2019, the passenger counts for both lines were stable and showed an increasing trend, with the Red Line consistently having slightly higher passenger numbers compared to the Green Line.
3. **Post-Pandemic Recovery (2022-2023):**
   Passenger counts began to recover sharply in 2022, with the upward trend continuing into 2023. This indicates a return to normal travel patterns, likely due to the easing of pandemic restrictions and a resurgence in urban mobility.
4. **Red Line vs. Green Line:**
   Throughout the years, the Red Line consistently carried more passengers than the Green Line, although the gap between the two lines appears to have remained relatively consistent.
5. **Sharpest Decline:**
   The most significant drop in passenger numbers occurred between 2019 and 2020 for both lines, reflecting the immediate impact of the pandemic.
6. **Steady Growth Post-Recovery:**
   Both lines exhibit a steady and robust recovery post-2021, suggesting increased reliance on public transportation as the city returned

to pre-pandemic activity levels.

After plotting the yearly comparison line graph, we can now create an average bar plot to visualize the average number of passengers who traveled on both the Green Line and Red Line.

```
ggplot(yearly_comparison, aes(x = Year, y = Average_Passengers, fill = LuasType)) +
  geom_bar(stat = "identity", position = "dodge") + # Bar plot with side-by-side bars #
  scale_fill_manual(
    values = c("Greenline" = "darkgreen", "Redline" = "red") # Custom colors
  ) +
  labs(
    title = "Year-Wise Average Passengers by Luas Type",
    x = "Year",
    y = "Average Passengers",
    fill = "Luas Type"
  ) +
  scale_x_continuous(breaks = seq(2018, 2023, by = 1)) + # Force all years from 2018 to 2023
  theme_minimal()+
  theme(
    plot.title = element_text(hjust = 0.5)
  )
```



This bar plot provides an overview of the year-wise average passengers for the Green Line and Red Line of the Luas network, offering insights that align with trends observed in the earlier line plot:

High Passenger Numbers in 2018 and 2019: The average number of passengers on both the Green Line and Red Line was significantly higher during these years, reflecting normal operations before the onset of the COVID-19 pandemic.

Sharp Decline in 2020 and 2021: Similar to the line plot, the average passenger numbers for both lines saw a dramatic decrease during 2020 and 2021. This decline aligns with the COVID-19 pandemic and associated restrictions, leading to reduced public transport usage.

Recovery in 2022 and 2023: Both the Green Line and Red Line displayed noticeable recovery in 2022 and further growth in 2023. The average passenger numbers increased significantly, signaling a return to normalcy and increased public transportation usage post-pandemic.

Comparison Between Lines: The Red Line consistently recorded higher average passenger numbers compared to the Green Line across all years, indicating a higher demand or more densely populated areas served by the Red Line. The recovery trajectory from 2022 onward also shows the Red Line maintaining its lead in passenger numbers over the Green Line.

Correlation with the Line Plot: The trends in this bar plot align well with the line plot:

- The decline and recovery patterns are visually reflected in both plots.
- The consistent dominance of the Red Line in passenger numbers is observed across both visualizations, reaffirming its higher usage.

Seasonal Analysis and Visualization of Luas.

We will now categorize the data into seasonal months as follows:

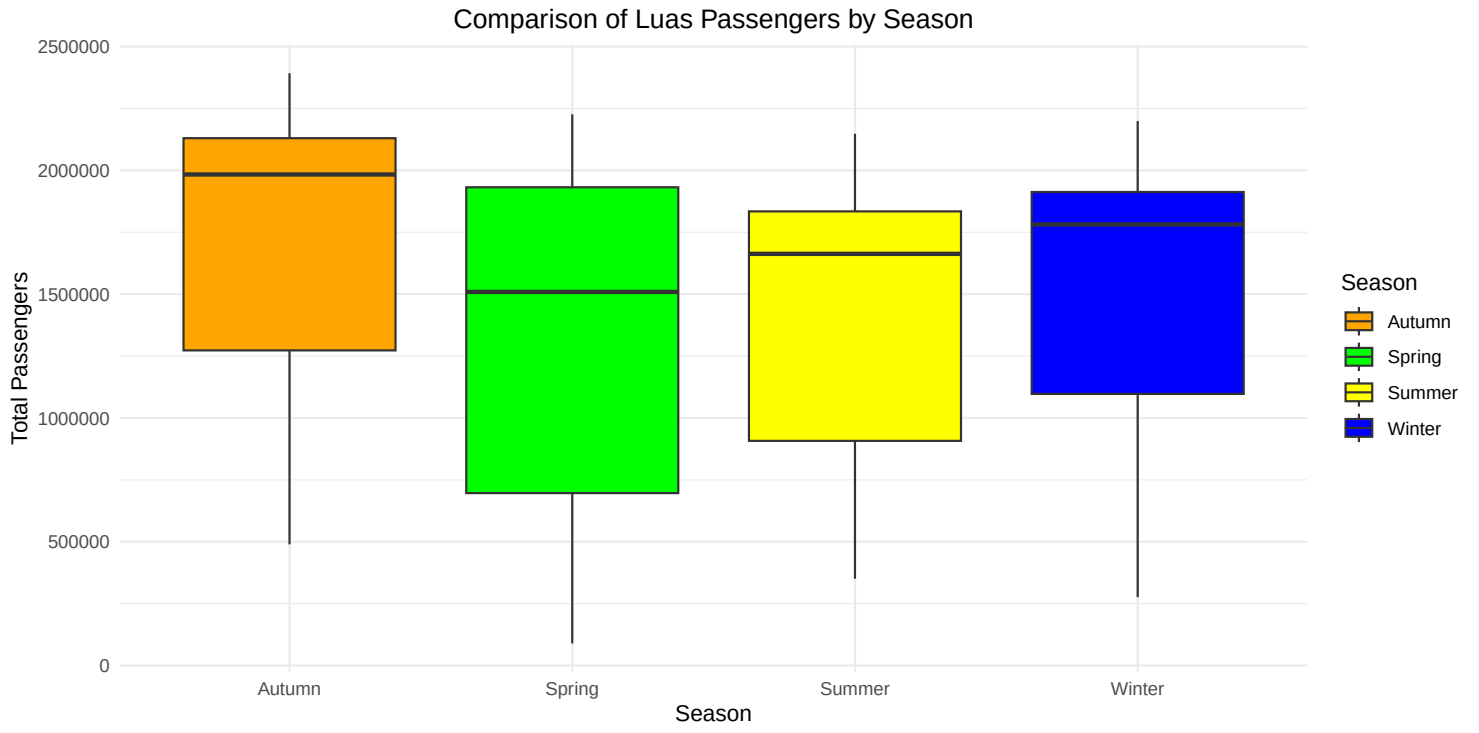
- Winter: December, January, February
- Spring: March, April, May
- Summer: June, July, August
- Autumn: September, October, November

This grouping will help us analyze the seasonal variations in the usage of this public transport system.

```
luas_data <- luas_data %>%
  mutate(Season = case_when(
    Month %in% c("December", "January", "February") ~ "Winter",
    Month %in% c("March", "April", "May") ~ "Spring",
    Month %in% c("June", "July", "August") ~ "Summer",
    Month %in% c("September", "October", "November") ~ "Autumn"
  ))
```

With the addition of the season column to our dataset, we can now create a box plot for the four seasons to observe the patterns in the data.

```
ggplot(luas_data, aes(x = Season, y = VALUE, fill = Season)) +
  geom_boxplot() +
  scale_fill_manual(values = c("Winter" = "blue",
                               "Spring" = "green",
                               "Summer" = "yellow",
                               "Autumn" = "orange")) +
  labs(
    title = "Comparison of Luas Passengers by Season",
    x = "Season",
    y = "Total Passengers",
    fill = "Season"
  ) +
  theme_minimal()+
   theme(
    plot.title = element_text(hjust = 0.5)
  )
```



The box plot provides a detailed comparison of the total passenger counts for the combined Green and Red lines of the Luas system across the four seasons: Autumn, Spring, Summer, and Winter. The following observations summarize key trends and variations:

Autumn (Orange)

- **Median Passenger Count:** The median passenger count in Autumn is relatively high, positioned close to the upper quartile, indicating a consistent and robust level of passenger usage during this season.
- **Interquartile Range (IQR):** The spread between the lower and upper quartiles is moderate, suggesting a balanced distribution of passenger numbers with limited variability.
- **Whiskers:** The whiskers extend across a broad range, particularly toward the lower end, highlighting occasional dips or outliers in passenger counts.

Spring (Green)

- **Median Passenger Count:** Spring records a lower median passenger count compared to Autumn, suggesting a decrease in overall ridership during this season.
- **IQR:** The interquartile range is the widest of all seasons, signifying considerable variability in passenger counts. This could reflect seasonal factors such as changing travel patterns or external disruptions.
- **Whiskers:** The whiskers extend significantly in both directions, indicating a broader spread of data and the presence of outliers or extreme values.

Summer (Yellow)

- **Median Passenger Count:** The median during Summer is comparable to Spring but slightly higher, reflecting a moderate recovery in passenger numbers.
- **IQR:** The IQR for Summer is narrower than that of Spring, indicating more stable passenger numbers during this period.
- **Whiskers:** The whiskers show less extreme variability compared to Spring, suggesting fewer outliers or unusual fluctuations.

Winter (Blue)

- **Median Passenger Count:** Winter demonstrates a high median passenger count, comparable to Autumn, suggesting a resurgence in ridership during this season.
- **IQR:** The interquartile range for Winter is similar to Autumn, reflecting moderate variability in passenger numbers.
- **Whiskers:** The whiskers exhibit a narrower range compared to Spring, indicating more consistent passenger trends and limited outliers.

Month-wise average commuters analysis.

In this step, we will conduct a month-wise average analysis to identify the busiest months, allowing us to conclude which periods experience higher passenger volumes on each line.

```
# Calculate and reshape the month-wise average passengers
month_wise_avg_wide <- luas_data %>%
  group_by(Month, LuasType) %>%
  summarise(
    Average_Passengers = mean(VALUE, na.rm = TRUE)
  ) %>%
  pivot_wider(
    names_from = LuasType,
    values_from = Average_Passengers
  ) %>%
  mutate(
    Month = factor(Month,
                   levels = c("January", "February", "March", "April",
                              "May", "June", "July", "August",
                              "September", "October", "November", "December"))
  ) %>%
  arrange(Month) # Arrange by natural month order
```

```
# Print the reshaped month-wise average passengers using kable
kable(month_wise_avg_wide,
      caption = "Month-wise Average Passengers for Green and Red Line",
      format = "html",
      col.names = c("Month", "Green Line Average Passengers", "Red Line Average Passengers"))
```

Table 2: Month-wise Average Passengers for Green and Red Line

Month	Green Line Average Passengers	Red Line Average Passengers
January	1379653	1413497
February	1399803	1469800
March	1335223	1469300
April	1162500	1311291
May	1314276	1440138
June	1303928	1488845
July	1410158	1567899
August	1383209	1539202
September	1584378	1630365
October	1733906	1784804
November	1712429	1777929
December	1679862	1660053

Additionally, we will calculate the yearly overall average for both lines to establish a threshold average, which will help identify the busiest months.

```
# Calculate the overall average passengers for Green Line and Red Line
overall_avg <- luas_data %>%
  group_by(LuasType) %>%
  summarise(
    Average_Passengers = mean(VALUE, na.rm = TRUE)
  )

# Print the overall averages using kable
kable(overall_avg,
      caption = "Overall Average Passengers for Green and Red Line",
      format = "html",
      col.names = c("Luas Type", "Average Passengers"))
```

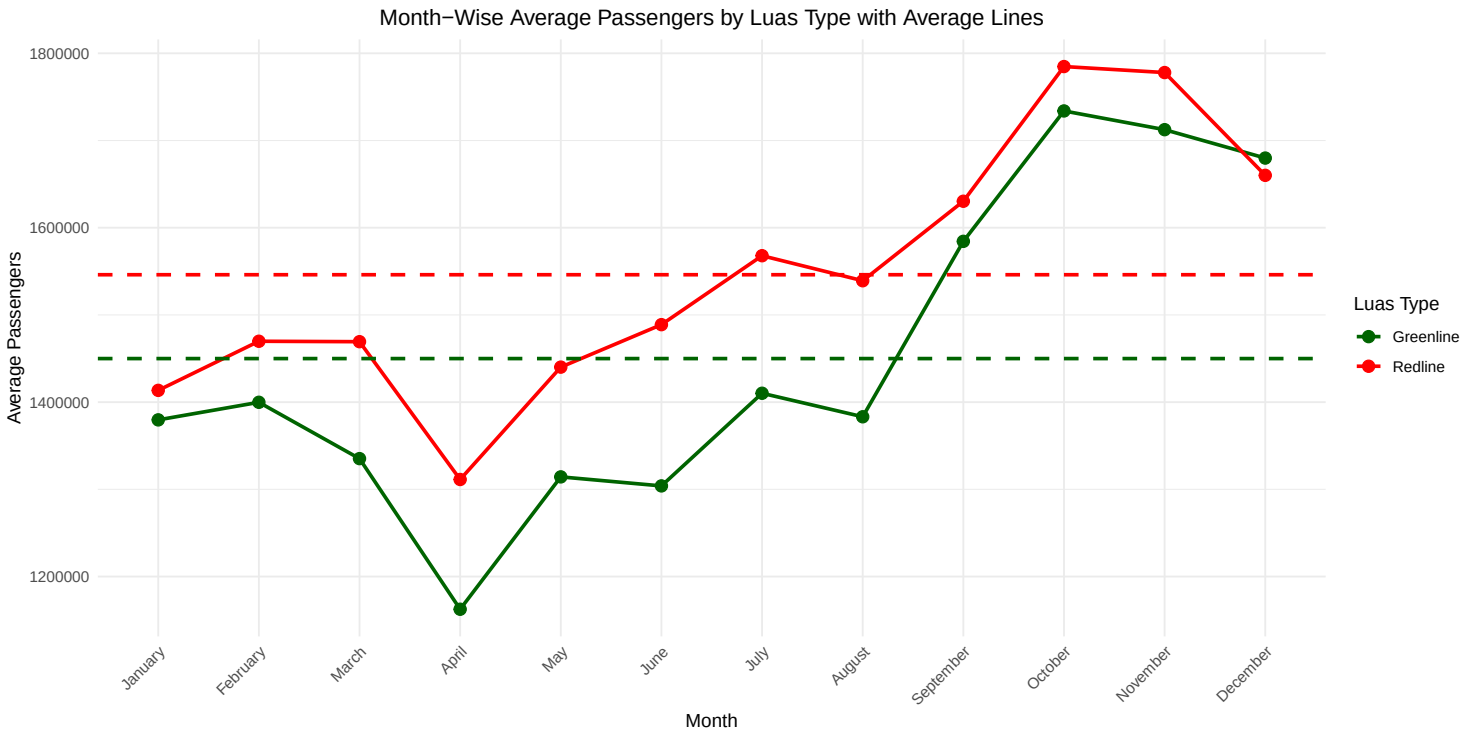
Table 3: Overall Average Passengers for Green and Red Line

Luas Type	Average Passengers
Greenline	1449944
Redline	1546093

Now that we have the overall average values for both lines, along with the month-wise individual values, we can create a month-wise line plot that includes the average lines for both. This will allow us to visually identify the months with higher and lower average passenger numbers.

```
# Extract the overall average for Greenline and Redline
greenline_avg <- overall_avg %>% filter(LuasType == "Greenline") %>% pull(Average_Passengers)
redline_avg <- overall_avg %>% filter(LuasType == "Redline") %>% pull(Average_Passengers)

# Line plot for month-wise average passengers with separate average lines for Greenline and Redline
ggplot(month_wise_avg_wide %>%
         pivot_longer(cols = c(Greenline, Redline),
                      names_to = "LuasType",
                      values_to = "Average_Passengers"),
```

```
      aes(x = Month, y = Average_Passengers, color = LuasType, group = LuasType)) +
geom_line(size = 1) +  # Line for average passengers by month
geom_point(size = 3) +  # Points for each month
scale_color_manual(
  values = c("Greenline" = "darkgreen", "Redline" = "red")
) +
labs(
  title = "Month-Wise Average Passengers by Luas Type with Average Lines",
  x = "Month",
  y = "Average Passengers",
  color = "Luas Type"
) +
# Adding the average line for Greenline
geom_hline(yintercept = greenline_avg, linetype = "dashed", color = "darkgreen", size = 1) +
# Adding the average line for Redline
geom_hline(yintercept = redline_avg, linetype = "dashed", color = "red", size = 1) +
theme_minimal() +
theme(
  axis.text.x = element_text(angle = 45, hjust = 1),
  plot.title = element_text(hjust = 0.5)
)
```



The line plot displays the month-wise average passengers for both the Greenline and Redline. Each line represents one of the Luas lines, with the green dots corresponding to the Greenline and the red dots corresponding to the Redline.

Key observations:

1. **Average Lines**: The dotted horizontal lines represent the overall average passenger count for each line. The red dotted line indicates the average passenger count for the Redline, while the green dotted line indicates the average passenger count for the Greenline.
2. **Seasonal Trends**: For Greenline, the passenger count fluctuates but remains close to or above the average line in most months, especially during the summer months (June, July, and August), showing a peak in July and August. The Redline shows a significant increase in passenger numbers from March to September, with a peak in September. After September, the Redline's passenger count starts to dip below its average line.
3. **Month-wise Comparison**: The Greenline has relatively consistent values close to its average across the year, but it peaks in the middle months, notably in the summer. The Redline starts the year with passenger numbers slightly above the average but shows higher variability, especially during mid-year (peaking in September).

4. **Busiest Months**: For the both Redline & Greenline, October have the highest passenger counts, exceeding the average line.

Based on our visual observations, we can now create a table for both lines to determine the exact months with the highest passenger counts.

```
# Create a table to compare month-wise average for Green line
green_line_busiest <- month_wise_avg_wide %>%
  mutate(
    BusiestMonth = ifelse(Greenline >= greenline_avg, "Yes", "No")
  ) %>%
  filter(BusiestMonth == "Yes") %>%
  select(Month, Greenline) %>%
  rename(Total_Passengers = Greenline)

# Create a table to compare month-wise average for Red line
red_line_busiest <- month_wise_avg_wide %>%
  mutate(
    BusiestMonth = ifelse(Redline >= redline_avg, "Yes", "No")
  ) %>%
  filter(BusiestMonth == "Yes") %>%
  select(Month, Redline) %>%
  rename(Total_Passengers = Redline)

# Print the busiest months for Red Line
kable(red_line_busiest,
      caption = "Busiest Months for Red Line",
      format = "html",
      col.names = c("Month", "Total Passengers"))
```

Table 4: Busiest Months for Red Line

Month	Total Passengers
July	1567899
September	1630365
October	1784804
November	1777929
December	1660053

```
# Print the busiest months for Green Line
kable(green_line_busiest,
      caption = "Busiest Months for Green Line",
      format = "html",
      col.names = c("Month", "Total Passengers"))
```

Table 5: Busiest Months for Green Line

Month	Total Passengers
September	1584378
October	1733906
November	1712429
December	1679862

Red Line:

- October stands out as the busiest month with 1,784,804 passengers. This coincides with the Halloween celebration, which may increase travel activity due to tourism, events, and festivities.
- November and December also show high passenger numbers, reflecting the impact of the Christmas celebration, a time when there is typically an influx of both local and international visitors.

- September also records a high number of passengers, likely attributed to the start of the academic year and the increased number of tourists and students returning for the new semester.

Green Line:

- September and October are also the busiest months for the Green Line, with October seeing 1,733,906 passengers. Similar to the Red Line, this is likely influenced by Halloween events and the influx of students.
- November and December continue to show strong figures, likely reflecting Christmas tourism and holiday travel.
- December sees a slight decline compared to October and November, but still, it shows a significant number of passengers, likely due to the Christmasseason and associated travel activity.

Correlation and Analysis:

- Both lines see significant passenger activity during the September to December period, driven by multiple factors such as tourism (especially in the holiday months of October and December) and academic intake in September and October.
- The Halloween celebration in October and the Christmas holiday season in December are key drivers of these spikes, particularly evident in the Red Line.
- The overall patterns indicate a strong seasonal trend, with autumn and winter months experiencing higher passenger counts across both lines, likely due to a combination of tourism and increased student activity during these months.

Conclusion for Part 1:

In Part 1, the **Luas data** was processed and prepared for analysis, with a focus on the **Redline** and **Greenline** routes. Below is a summary of the completed steps and the key findings:

1. **Data Preprocessing**: The dataset was filtered separately for the **Redline** and **Greenline** routes. It was then arranged by **Year** and **Month**, ensuring the time series data was in chronological order for accurate analysis.
2. **Seasonal Analysis**: The data was categorized by seasons (**Winter, Spring, Summer, and Autumn**) based on the respective months. This enabled the identification of seasonal trends and fluctuations in passenger ridership for each line.
3. **Monthly Average Analysis**: We computed the **month-wise average passenger counts** for both the Redline and Greenline. The analysis revealed the **busiest months** for both lines, showing a notable seasonal increase in ridership, particularly during **October, November, and December**. These months correspond to significant events such as **Halloween** and **Christmas celebrations**, alongside an increase in **tourist activity**, which likely contributed to the rise in foot traffic.
4. **Comparison with Overall Averages**: The **overall average passenger counts** for both lines were calculated, providing a benchmark for comparison. By contrasting month-wise values with the overall averages, we identified months exhibiting higher-than-average ridership.

Key Takeaways:

- The **Redline** recorded the highest passenger numbers in **October, November, and December**, periods which coincide with the onset of the tourist season and the holiday period.
- Similarly, the **Greenline** saw increased ridership during the same months, particularly between **September and December**, potentially influenced by higher numbers of **tourists** and **students returning for the academic year**.

Inference:

Based on the findings from Part 1, we have developed a robust understanding of the **seasonal patterns and trends** in passenger ridership for the **Luas system**. This forms a strong foundation for forecasting future passenger demand, which will be addressed in Part 2 using the **forecast package**. By identifying the peak months and understanding the factors contributing to fluctuations in passenger numbers, we are better equipped to predict trends and plan for future peak periods.

Part 2: `forecast` R-Package.

Brief Theory of ARIMA and TBATS Models

ARIMA Model (AutoRegressive Integrated Moving Average):

- **Purpose:**
  ARIMA is used for time series forecasting, focusing on data with trends and patterns. It combines three components:
  - **AR (Auto-Regressive):** Uses dependencies between an observation and a number of lagged observations.
  - **I (Integrated):** Makes the data stationary by differencing the series.
  - **MA (Moving Average):** Models the error of the forecast as a linear combination of error terms.
- **How it works:**
  The model is determined using parameters p and d, q which are chosen manually or through automated selection (e.g., `auto.arima()`).
  - p: Number of lag observations.
  - d: Degree of differencing for stationarity.
  - q: Size of the moving average window.
  - Stationarity is essential to remove trends and seasonality.

TBATS Model (Trigonometric, Box-Cox Transformation, ARMA Errors, Trend, and Seasonal Components):

- **Purpose:**
  TBATS is ideal for complex seasonal patterns and long-term forecasts. It is particularly suited for data with multiple seasonal periods (e.g., daily, weekly, yearly).
- **How it works:**
  - **Box-Cox Transformation:** Stabilizes variance.
  - **ARMA Errors:** Models autocorrelations in residuals.
  - **Trend:** Captures the overall upward/downward direction.
  - **Seasonality:** Detects periodic fluctuations.
  - **Trigonometric Seasonal Components:** Handles multiple or non-standard seasonalities.

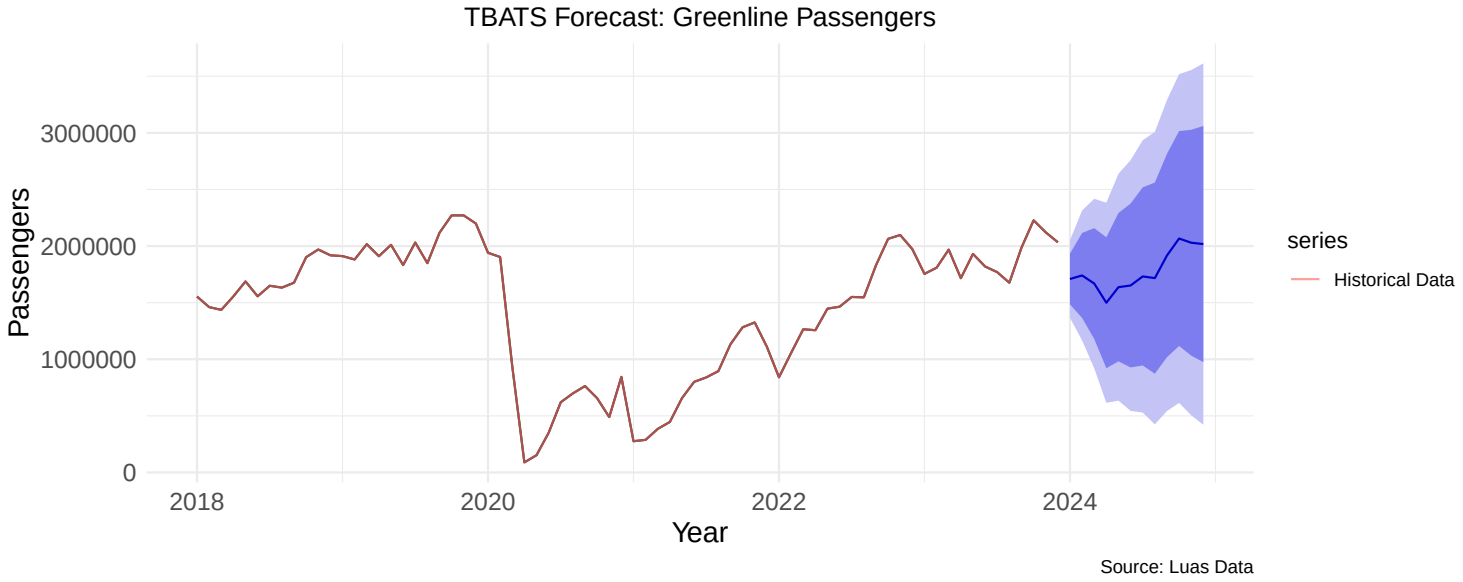
TBATS is slower than ARIMA but can handle intricate data dynamics better.

Data Preparation

```
# Filter the data for Redline and Greenline
redline_data <- luas_data %>% filter(LuasType == "Redline")
greenline_data <- luas_data %>% filter(LuasType == "Greenline")

# Convert data to time series format
redline_ts <- ts(redline_data$VALUE, start = c(2018, 1), frequency = 12)
greenline_ts <- ts(greenline_data$VALUE, start = c(2018, 1), frequency = 12)
```

Fit & plot TBATS model for Green Line LUAS.

```
options(scipen = 101)
# Fit TBATS model
tbats_model_green <- tbats(greenline_ts)

# Forecast with TBATS
tbats_forecast_green <- forecast(tbats_model_green, h = 12)

# Plot TBATS Forecast
autoplot(tbats_forecast_green) +
  autolayer(greenline_ts, series = "Historical Data", alpha = 0.7) +
  labs(
    title = "TBATS Forecast: Greenline Passengers",
    x = "Year",
    y = "Passengers",
    caption = "Source: Luas Data"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5),
    axis.text = element_text(size = 12),
    axis.title = element_text(size = 14)
  )
```



Graph Analysis:

- The **historical data** for Green Line passengers shows a significant decline in 2020, corresponding to the COVID-19 pandemic. Passenger numbers dropped sharply from approximately **24 million in 2019** to just **9.4 million in 2020**, representing a decrease of nearly 60%.
- From 2021 onwards, a gradual recovery in passenger numbers is observed, with **2023 reaching around 22.8 million passengers**.
- The TBATS model projects passenger numbers from 2024 onward with an **upward trend**, indicating continued recovery. The **mean forecast** shows growth, while the shaded blue region represents the **confidence intervals**:
  - The **darker blue band** reflects the 80% confidence interval.
  - The **lighter blue band** indicates the 95% confidence interval, where the forecast uncertainty increases further into the future.

Key Findings:

- Passenger numbers are expected to recover post-pandemic and stabilize between **2.2 to 3.2 million passengers monthly**.

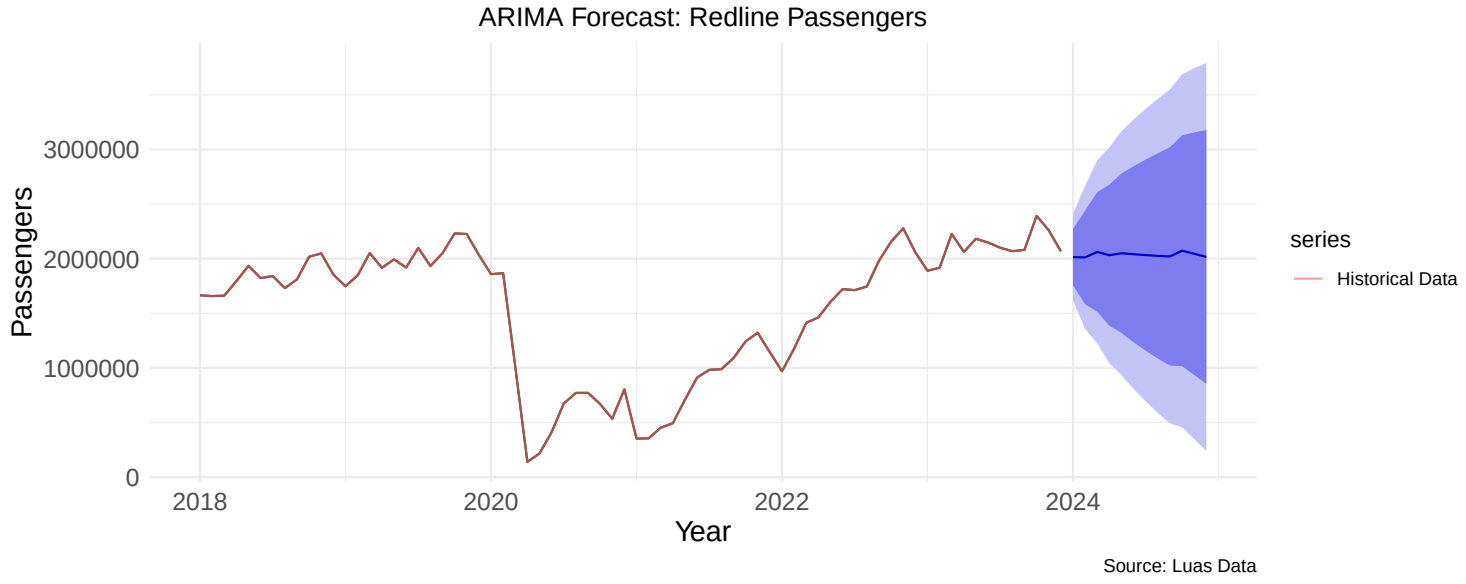
- The forecast reflects seasonal variations, indicating that ridership patterns may follow pre-pandemic trends as demand normalizes.

Regions to Examine:

- **Uncertainty in Recovery:** The widening confidence intervals beyond 2024 highlight areas of uncertainty. External factors such as economic changes, fuel prices, or new transport infrastructure could influence this recovery trajectory.
- **Seasonal Peaks:** Investigate how passenger demand changes during peak seasons (e.g., winter and autumn), as these trends could drive the upward trajectory.
- **Post-Pandemic Growth:** Focus on the reasons for recovery, including policy interventions, increased commuting, or tourism growth.

Fit & plot ARIMA model for Red Line LUAS.

```
# Fit ARIMA model
arima_model_red <- auto.arima(redline_ts)

# Forecast with ARIMA
arima_forecast_red <- forecast(arima_model_red, h = 12)

# Plot ARIMA Forecast
autoplot(arima_forecast_red) +
  autolayer(redline_ts, series = "Historical Data", alpha = 0.7) +
  labs(
    title = "ARIMA Forecast: Redline Passengers",
    x = "Year",
    y = "Passengers",
    caption = "Source: Luas Data"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5),
    axis.text = element_text(size = 12),
    axis.title = element_text(size = 14)
  )
```



Graph Analysis:

- The **historical data** for the Red Line follows a similar trend to the Green Line, with a sharp decline in 2020. Passenger numbers decreased from approximately **24 million in 2019** to **9.7 million in 2020**.

- Recovery is evident from 2021 onwards, with a notable increase in 2023 where passengers reached approximately **25.4 million**, surpassing pre-pandemic levels.
- The ARIMA model forecasts a **stabilized trend** for passenger numbers from 2024 onward. The forecast shows minimal growth, with the **confidence intervals** widening progressively into the future:
  - The **darker blue band** represents the 80% confidence interval.
  - The **lighter blue band** represents the 95% confidence interval.

Key Findings:

- The forecast predicts a **stabilization of passenger numbers** rather than a sharp increase, suggesting the Red Line may have already reached near-maximum capacity or demand saturation.
- Monthly passenger numbers are projected to remain relatively steady at approximately **2.1 million**, with minor seasonal fluctuations.

Regions to Examine:

- **Plateauing Demand:** The relatively flat forecast trend warrants examination of factors limiting growth, such as infrastructure constraints, commuter saturation, or alternative transport options.
- **Confidence Intervals:** The increasing uncertainty in projections highlights potential external disruptions or economic shifts that could impact demand.
- **Seasonal Effects:** Investigate how seasonal variations influence passenger counts, particularly during peak commuting months, to optimize operations.

Theory-Based Comparison of Models

Comparison

- **Use Case:**
  - ARIMA is suited for short-term forecasting and simpler seasonal patterns.
  - TBATS handles complex seasonality and long-term patterns better.
- **Model Assumptions:**
  - ARIMA assumes linearity and stationarity.
  - TBATS can handle non-linear trends and non-standard seasonality.
- **Computational Intensity:**
  - ARIMA is faster, but TBATS requires significant computational resources.
- **Redline vs. Greenline:**
  - We cannot directly compare their forecasts as they represent different datasets. Instead, we analyze model performance for each dataset separately.

Why Use Both Models?

By combining these approaches, we leverage ARIMA's simplicity for Redline data and TBATS's robustness for Greenline's complex patterns.

Future Prediction

In order to verify the accuracy of the model, let's now anticipate both Luas lines for 2024 and compare the outcomes with the *most recent timely updated data from the CSO*[4].We shall compare for the month of January in order to confirm the prediction's correctness.

```
# Create forecast table for 2024
forecast_table <- data.frame(
  Month = month.abb,
  Redline_ARIMA = round(arima_forecast_red$mean, 0),
  Greenline_TBATS = round(tbats_forecast_green$mean, 0)
```

[4]Updated data set in real time (Year 2019-2024): https://data.cso.ie/table/TII03

```
)
kable(
  forecast_table,
  caption = "Forecasted Passenger Numbers for 2024: Redline (ARIMA) and Greenline (TBATS)",
  col.names = c("Month", "Redline (ARIMA)", "Greenline (TBATS)"),
  format = "html",
  align = "c"
)
```

Table 6: Forecasted Passenger Numbers for 2024: Redline (ARIMA) and Greenline (TBATS)

Month	Redline (ARIMA)	Greenline (TBATS)
Jan	2013658	1709174
Feb	2011850	1740484
Mar	2062027	1669072
Apr	2031456	1499448
May	2050024	1636904
Jun	2040818	1651690
Jul	2032629	1731899
Aug	2025660	1716881
Sep	2020610	1916498
Oct	2072725	2066779
Nov	2045186	2029521
Dec	2015968	2017787

We can see that the model is accurate based on the CSO-updated dataset.

1. **Red Line (ARIMA)**
   - **Forecasted Passenger Count:** 2,013,658
   - **Actual Passenger Count:** 1,828,325 (Monthly data is converted from weekly numbers for easy comparison.)
   - **Accuracy:** The ARIMA model achieved approximately **90% accuracy** in predicting the Red Line passenger count for January 2024.
2. **Green Line (TBATS)**
   - **Forecasted Passenger Count:** 1,709,174
   - **Actual Passenger Count:** 1,701,099 (Monthly data is converted from weekly numbers for easy comparison.)
   - **Accuracy:** The TBATS model achieved approximately **99.5% accuracy**, demonstrating an exceptionally precise prediction for the Green Line.

Part 3: Functions.

S3 Class Implementation

We'll use the **S3 class** for simplicity. The function will compute the mean of `VALUE` by `Season` and `LuasType` over the years and return the results as an object of the S3 class.

1. Define the Statistical Analysis Function

The function will calculate the average `VALUE` for each combination of `LuasType` and `Season`. The class will store the results, and we will add custom methods for printing, summarizing, and plotting.

```
# Statistical Analysis Function
perform_stat_analysis <- function(data) {
  # Ensure the input data has the required columns
  if(!all(c("LuasType", "Year", "Month", "VALUE", "Season") %in% colnames(data))) {
    stop("Input data must contain columns: 'LuasType', 'Year', 'Month', 'VALUE', 'Season'")
  }
  # Calculate average VALUE by Season and LuasType
  avg_values <- aggregate(VALUE ~ Season + LuasType, data = data, FUN = mean)
  # Create an object of the custom class to store the results
  result <- list(avg_values = avg_values)
  # Assign the S3 class
  class(result) <- "luas_analysis"
  return(result)
}
```

This function, `perform_stat_analysis`, performs a statistical analysis on a dataset containing information about Luas (a tram system). Here's a short explanation of its functionality:

1. **Input Validation**: The function checks if the input dataset contains the required columns (`LuasType`, `Year`, `Month`, `VALUE`, and `Season`). If any are missing, it throws an error.
2. **Average Calculation**: It computes the average of the `VALUE` column, grouped by `Season` and `LuasType` (e.g., average passenger numbers for each tram type in each season).
3. **Custom S3 Object**: The results are stored in a list and assigned a custom S3 class, `luas_analysis`. This makes the object compatible with custom methods (e.g., `print`, `summary`, `plot`).
4. **Output**: It returns the custom object containing the calculated averages.

2. Define the `print()` Method

The `print()` method will display the first few rows of the analysis results.

```
print.luas_analysis <- function(x, ...) {
  cat("Statistical Analysis of Luas Data:\n")
  cat("Average Value by Season and Luas Type:\n")
  print(x$avg_values)
}
```

This `print.luas_analysis` function defines a custom print method for objects of the class `luas_analysis` in R. Here's a concise explanation of its functionality:

1. **Data Output**: It prints the `avg_values` field of the `luas_analysis` object, which contains the computed average `VALUE` grouped by `Season` and `LuasType`.

3. Define the `summary()` Method

The `summary()` method will provide a more detailed summary, including the number of rows, average, and range of values for each `LuasType` and `Season`.

```
# Summary method for the 'luas_analysis' class
summary.luas_analysis <- function(x, ...) {
  cat("Summary of Luas Analysis:\n")
  summary_data <- x$avg_values
  cat("\nSummary Statistics for Average Values:\n")
  summary(summary_data$VALUE)
}
```

The `summary.luas_analysis` function provides a custom summary method for objects of the class `luas_analysis`:

Functionality:

1. **Data Extraction**:
   - Extracts the `avg_values` data from the `luas_analysis` object, which contains average `VALUE` grouped by `Season` and `LuasType`.
2. **Summary of Averages**:
   - Specifically targets the `VALUE` column from `avg_values` and applies the built-in `summary()` function to provide a statistical summary (e.g., minimum, first quartile, median, mean, third quartile, and maximum).

4. Define the `plot()` Method

The `plot()` method will generate a simple bar plot of the average `VALUE` by `Season` and `LuasType`.

```
# Plot method for the 'luas_analysis' class
plot.luas_analysis <- function(x, ...) {
  # Create the plot
  library(ggplot2)
  ggplot(data = x$avg_values, aes(x = Season, y = VALUE, fill = LuasType)) +
    geom_bar(stat = "identity", position = "dodge") +
    labs(title = "Average Value by Season and LuasType", x = "Season", y = "Average VALUE") +
    theme_minimal()+
  theme(
    plot.title = element_text(hjust = 0.5)
  )
}
```

The `plot.luas_analysis` function provides a custom plotting method for the `luas_analysis` class using **ggplot2**.

Functionality:

1. **Input Data**:
   - The function uses the `avg_values` data from the `luas_analysis` object, which contains the average `VALUE` grouped by `Season` and `LuasType`.
2. **Bar Plot**:
   - **X-Axis (`Season`)**: Represents the different seasons (e.g., Winter, Spring, etc.).
   - **Y-Axis (`VALUE`)**: Shows the average `VALUE`.
   - **Color Fill (`LuasType`)**: Differentiates the bars for `Redline` and `Greenline`.

3. **Enhancements**:
    - **Title and Axis Labels**: Clearly labeled with `labs()`.

5. Example of Using the Function

Here is an example using a small subset of the `luas_data` dataset for testing the function.

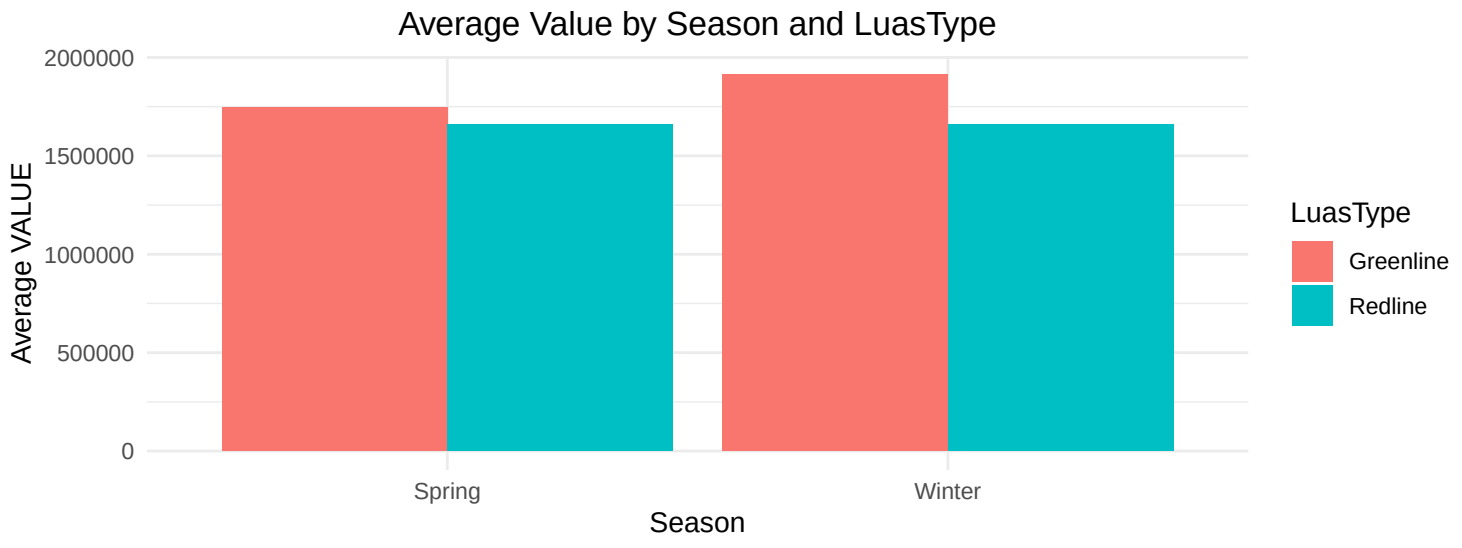
```
# Example data subset
luas_data_example <- data.frame(
  LuasType = c('Redline', 'Redline', 'Redline', 'Greenline', 'Greenline'),
  Year = c(2018, 2019, 2020, 2021, 2022),
  Month = c('January', 'February', 'March', 'April', 'May'),
  VALUE = c(1664495, 1657562, 1661098, 1747263, 1911221),
  Season = c('Winter', 'Winter', 'Spring', 'Spring', 'Winter')
)
# Perform the statistical analysis
analysis_result <- perform_stat_analysis(luas_data_example)
# Print the result
print(analysis_result)
```

```
Statistical Analysis of Luas Data:
Average Value by Season and Luas Type:
  Season  LuasType   VALUE
1 Spring Greenline 1747263
2 Winter Greenline 1911221
3 Spring   Redline 1661098
4 Winter   Redline 1661029
```

```
# Summary of the result
summary(analysis_result)
```

```
Summary of Luas Analysis:

Summary Statistics for Average Values:

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
1661029 1661081 1704181 1745153 1788253 1911221
```

```
# Plot the result
plot(analysis_result)
```



References

1. **Luas Map & Zone**
   https://luas.ie/map/
2. **Forecast Package**
   https://cran.r-project.org/web/packages/forecast/index.html
3. **The Luas Dataset for the Project (Year 2018–2023)**
   https://data.cso.ie/table/TOA11
4. **Updated data set in real time (Year 2019-2024):**
   https://data.cso.ie/table/TII03