Statistical Forecasting Project

Forecasting Restaurant Visitor Data for Operational Planning

Created by: Rekha Devendra

Created date: October 19, 2024

Table of Contents

1. Data

Rationale for Choosing the Data

Rationale for Choosing the Data The dataset chosen for this project is the Recruit Restaurant Visitor Forecasting dataset from Kaggle. The dataset provides historical data on restaurant visitor counts, which includes reservations and actual visitation data collected through multiple platforms such as Hot Pepper Gourmet, AirREGI, and Restaurant Board. Predicting the number of restaurant visitors can help restaurant managers make better decisions regarding staffing, ingredient procurement, and overall business operations. Accurate forecasting is crucial because factors like public holidays, weather, and competing businesses affect visitor numbers.

This dataset offers a rich time series of visitor data ideal for developing forecasting models. The main aim is to predict future restaurant visitors for specific dates, including Japan's Golden Week, a major holiday that can cause significant fluctuations in customer attendance.

Data Cleaning and Import

The raw dataset contains over 252,000 rows of historical visit data. The file used is air_visit_data.csv, which consists of three key columns:

- **air_store_id:** a unique identifier for each restaurant
- **visit_date:** the date of each visit
- **visitors:** the number of visitors on that specific date

Before performing any analysis, the dataset was imported into R and inspected for the following, none of which were found:

- **Duplicate removal:** The first step is to ensure no duplicate entries for the same restaurant and date.
- **Handling missing data:** Gaps in the date range filled with zero visitors for days when the restaurant was likely closed.
- **Filtering for Specific Restaurants:** To focus on relevant data, the dataset was filtered to include only entries for a specific restaurant using its unique identifier, air_store_id. This step ensures that our analyses and forecasts are based solely on the performance of the chosen restaurant.
- **Splitting the data:** The data was then split into a training set (80%) and a testing set (20%) to evaluate model performance.

Practical Problem

The practical problem I aim to address is how restaurant managers can more effectively anticipate visitor traffic during peak periods such as holidays. Accurate visitor forecasts are crucial for making informed staffing, inventory management, and overall operational efficiency decisions. Restaurant managers can streamline operations by analyzing historical data and predicting future visitor numbers, minimizing food waste, and optimizing staffing schedules to meet customer demand.

For instance, during high-demand periods like Japan's Golden Week, restaurants often experience significant fluctuations in attendance. Without reliable forecasting, they may struggle to manage resources efficiently, leading to overstaffing or understaffing, which can negatively impact customer satisfaction and profitability. By leveraging predictive analytics, restaurants can make

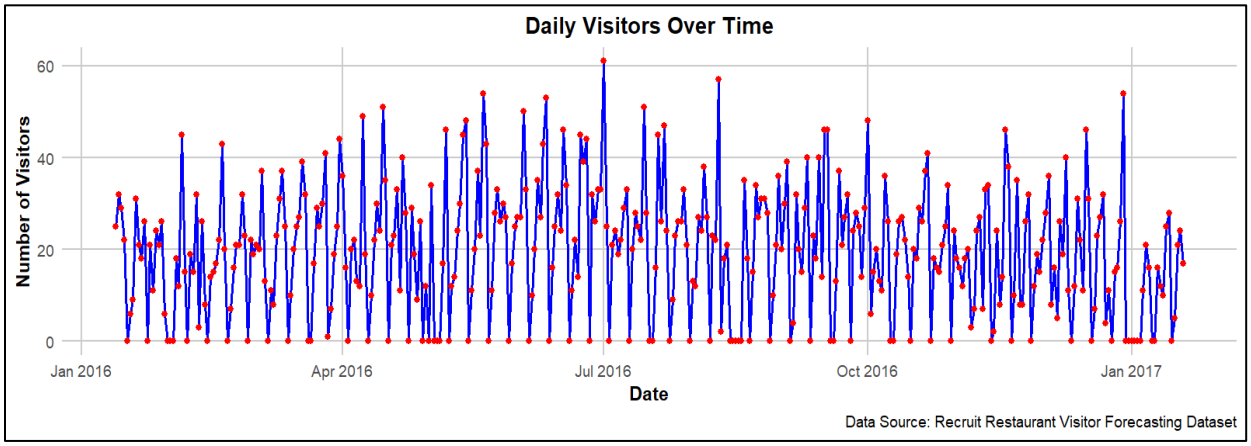
data-driven decisions that enhance their service delivery and operational performance, ultimately creating a more enjoyable dining experience for customers.

This project seeks to develop robust forecasting models to empower restaurant managers to make strategic choices that align with customer demand and improve their overall business outcomes.

2. Visualization

Time Plot of Visitors Over Time

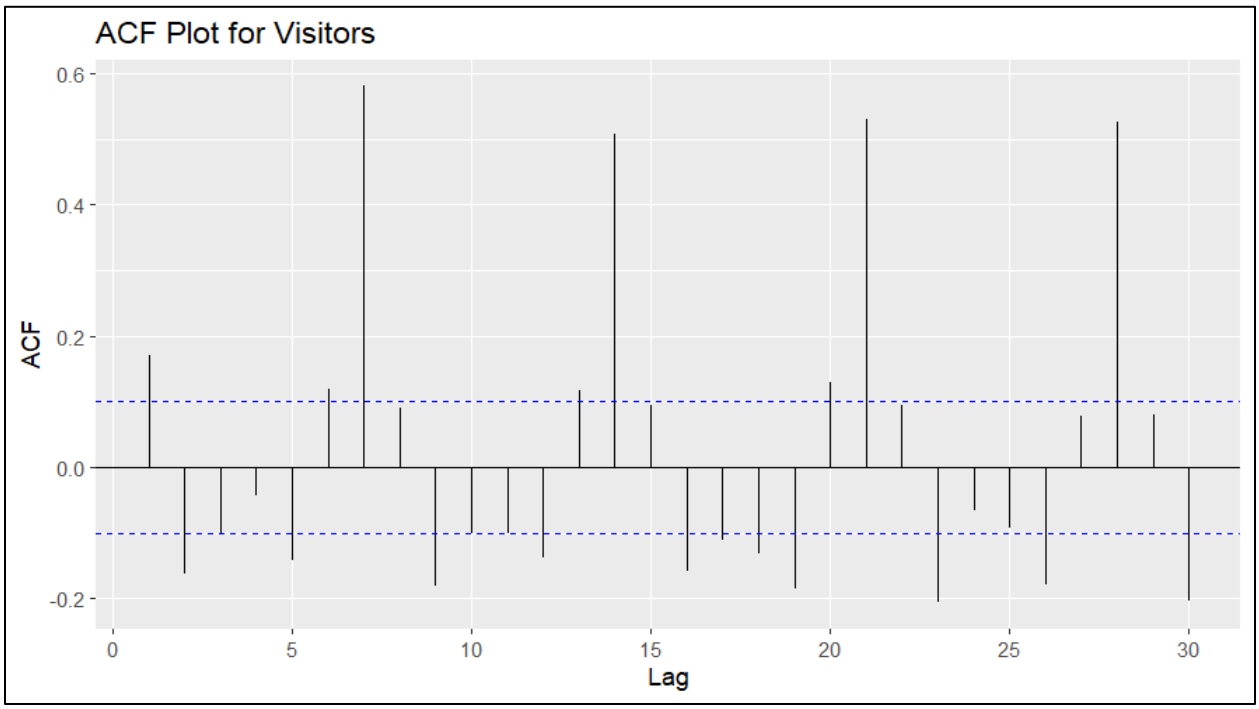
A basic time plot of visitor data was created to visually examine the trends and fluctuations over time.



This time plot shows the number of visitors fluctuating daily. Key observations include:

- A general trend of rising visitors over time.
- Clear weekly seasonality, with certain peaks and troughs repeating every seven days (likely corresponding to weekends or specific busy days for restaurants).
- Some outliers represent exceptionally busy days, potentially due to special events or holidays.

ACF plot for visitors



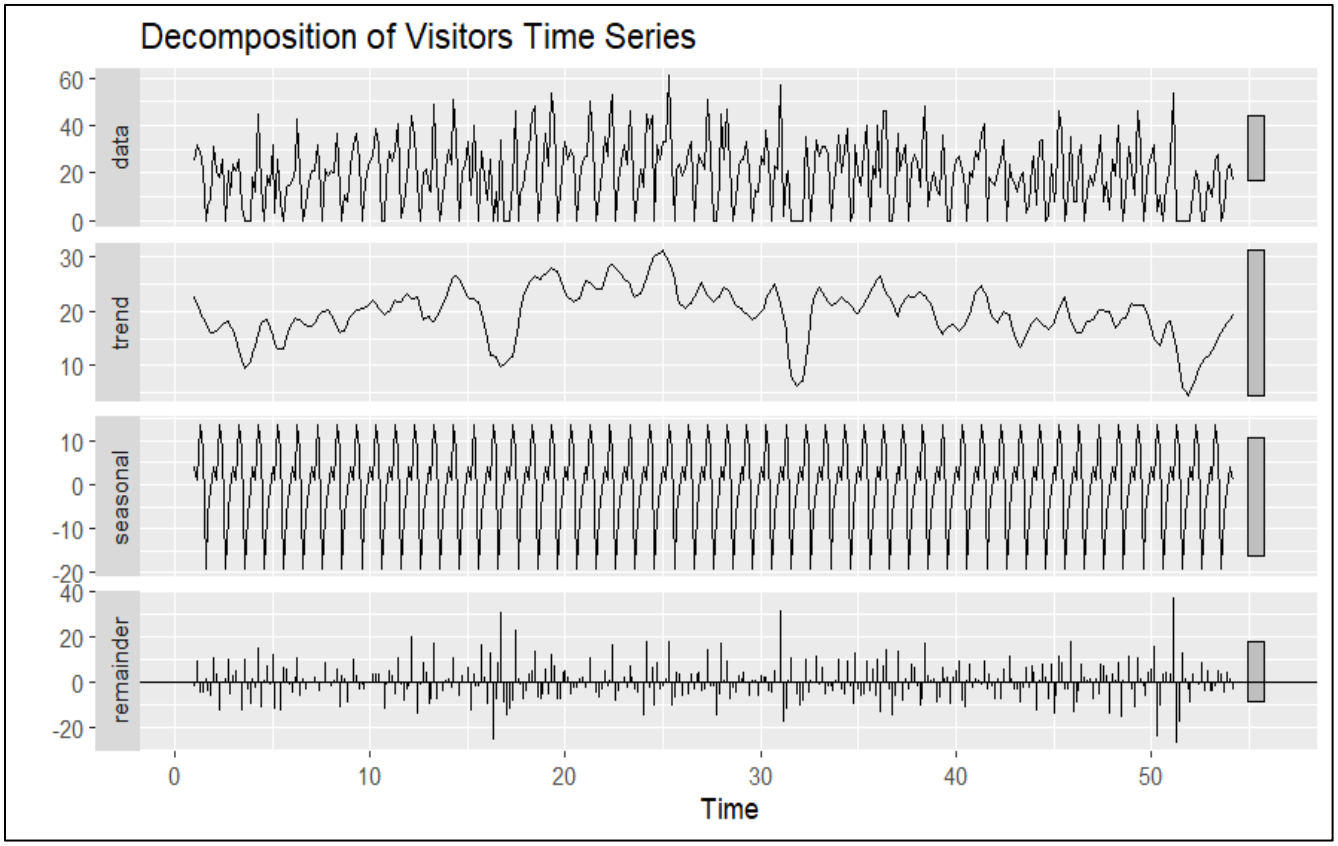
The Autocorrelation Function (ACF) plot reveals significant temporal dependencies in the visitor data for the selected restaurant. Notably, strong positive correlations are observed at lags 7, 14, 21, and 28, indicating a clear weekly seasonality in visitor patterns, suggesting that visitor numbers on a given day are closely related to those on the same day in previous weeks, highlighting the restaurant's tendency to attract similar traffic on corresponding weekdays.

In contrast, negative correlations at lags 5, 10, 25, and 30 indicate periods of decline in visitor numbers relative to specific past observations. For example, a significant negative correlation at lag 5 suggests high visitor counts on a particular day, followed by a drop in attendance five days later. This could reflect customer fatigue after busy periods, such as weekends, or other cyclical influences impacting restaurant attendance.

Overall, the insights from the ACF plot emphasize the importance of incorporating seasonality into forecasting models and understanding patterns of visitor decline that can inform operational strategies, such as staffing and inventory management.

3. Data Transformation

Decomposition of Visitors Time Series



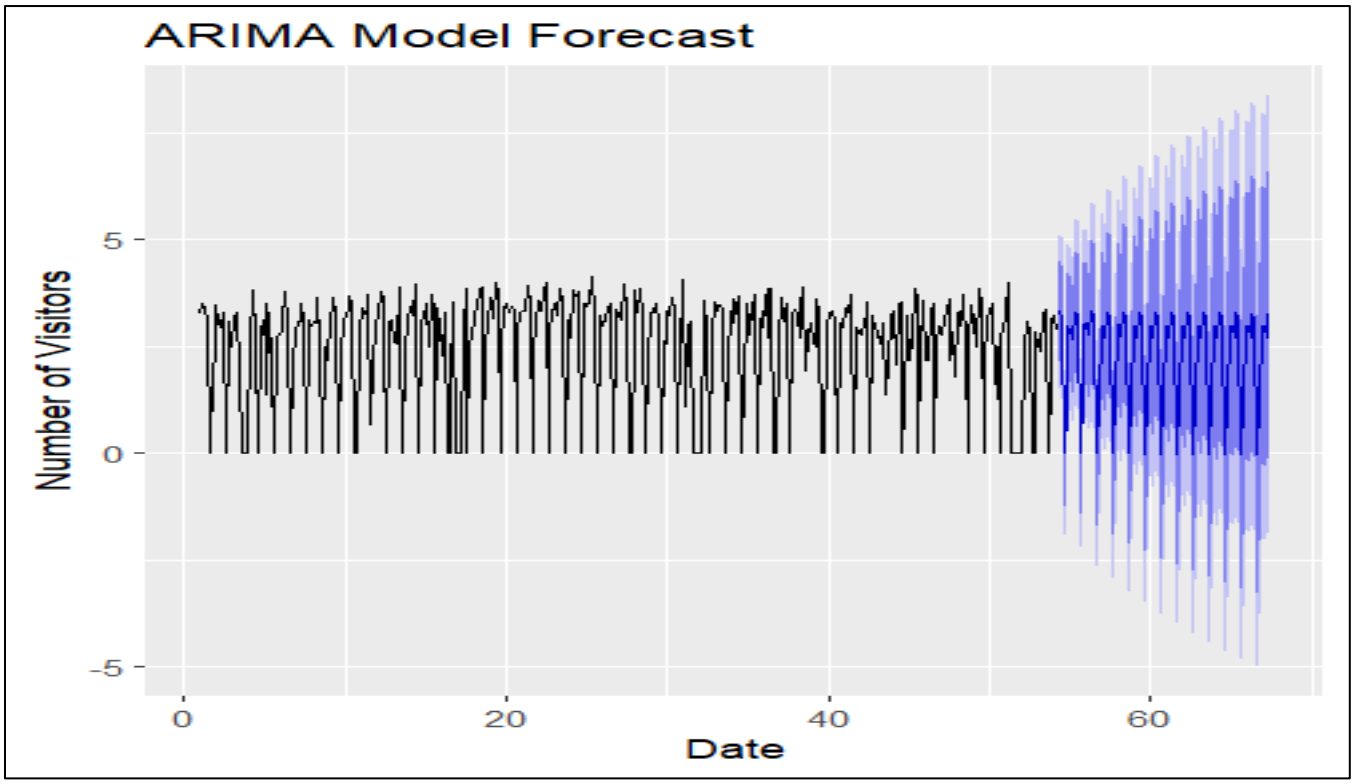
The decomposition graph of the restaurant visitor data reveals essential insights into the underlying patterns of customer visits over time:

- **Seasonal Effect**: The seasonal component shows a fluctuation range of **+10 to -10** visitors, indicating a consistent periodic pattern in customer traffic. This seasonal effect suggests that the number of visitors tends to rise and fall regularly, likely influenced by holidays, weekends, or seasonal promotions. The observed repeating patterns at approximately **16, 31, and 51** on the time axis indicate specific periods within the dataset where customer visits are predictably higher or lower, reflecting the cyclical nature of restaurant patronage.

- **Remainder**: The remainder component fluctuates between **20 and -20**, signifying the presence of irregular variations in visitor counts that are not explained by seasonal trends. This indicates that while there are predictable patterns, external factors also impact visitor numbers, such as weather conditions, local events, or changes in the restaurant's offerings. These irregular fluctuations highlight customer behavior's complexity and non-seasonal factors' influence on restaurant attendance.

4. Forecasting and Analysis

1. ARIMA Model



Plot Description: The ARIMA plot visualizes the predicted number of visitors alongside the actual visitor data. We can see how the forecasted values trend upward or downward in terms of historical data, capturing seasonality and potential trends.

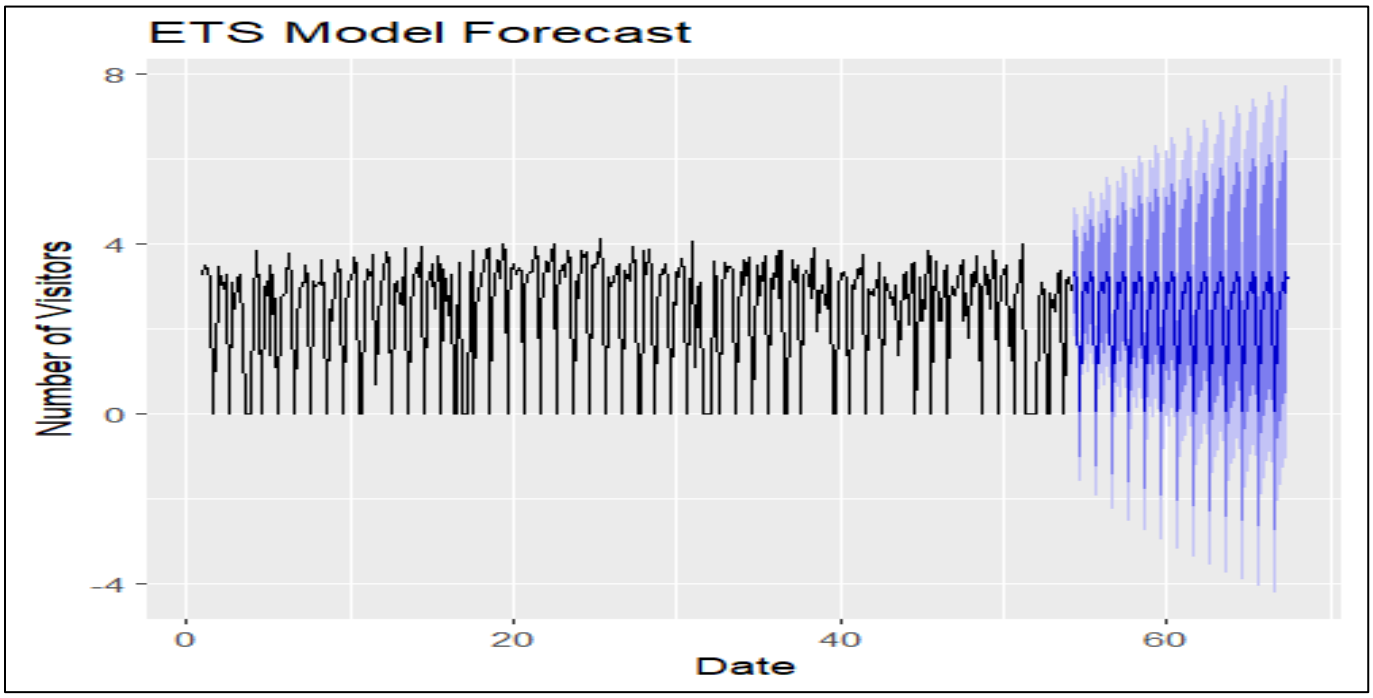
Performance:

- **RMSE**: 14.45

- **MAE**: 0.90

Analysis: The ARIMA model captures the underlying patterns well but may struggle with large fluctuations. The relatively low RMSE and MAE suggest that while it performs adequately, it might only sometimes reflect sharp changes in visitor counts.

2. ETS Model



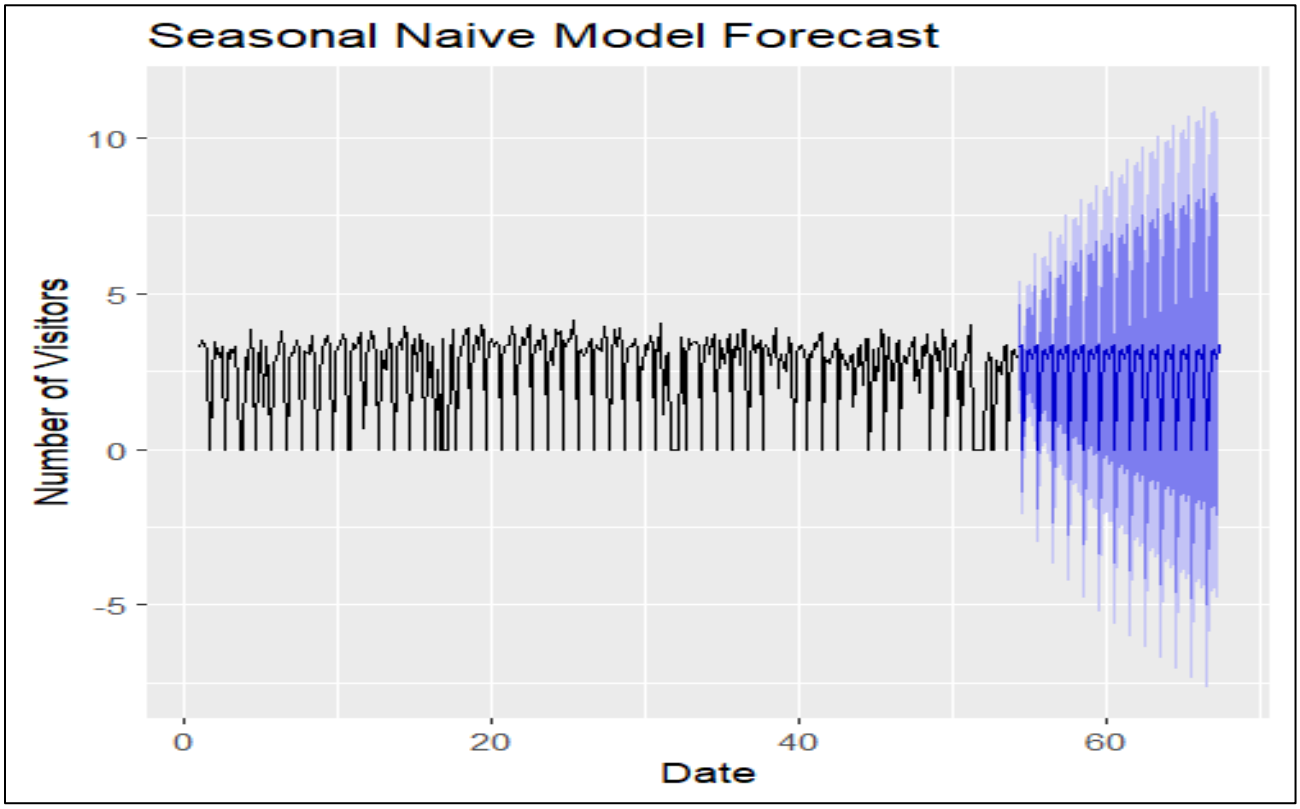
Plot Description: The ARIMA plot visualizes the predicted number of visitors alongside the actual visitor data. We can see how the forecasted values trend upward or downward in terms of historical data, capturing seasonality and potential trends.

Performance:

- **RMSE**: 14.45
- **MAE**: 0.90

Analysis: The ARIMA model captures the underlying patterns well but may struggle with large fluctuations. The relatively low RMSE and MAE suggest that while it performs adequately, it might only sometimes reflect sharp changes in visitor counts.

3. Seasonal Naive Model Forecast

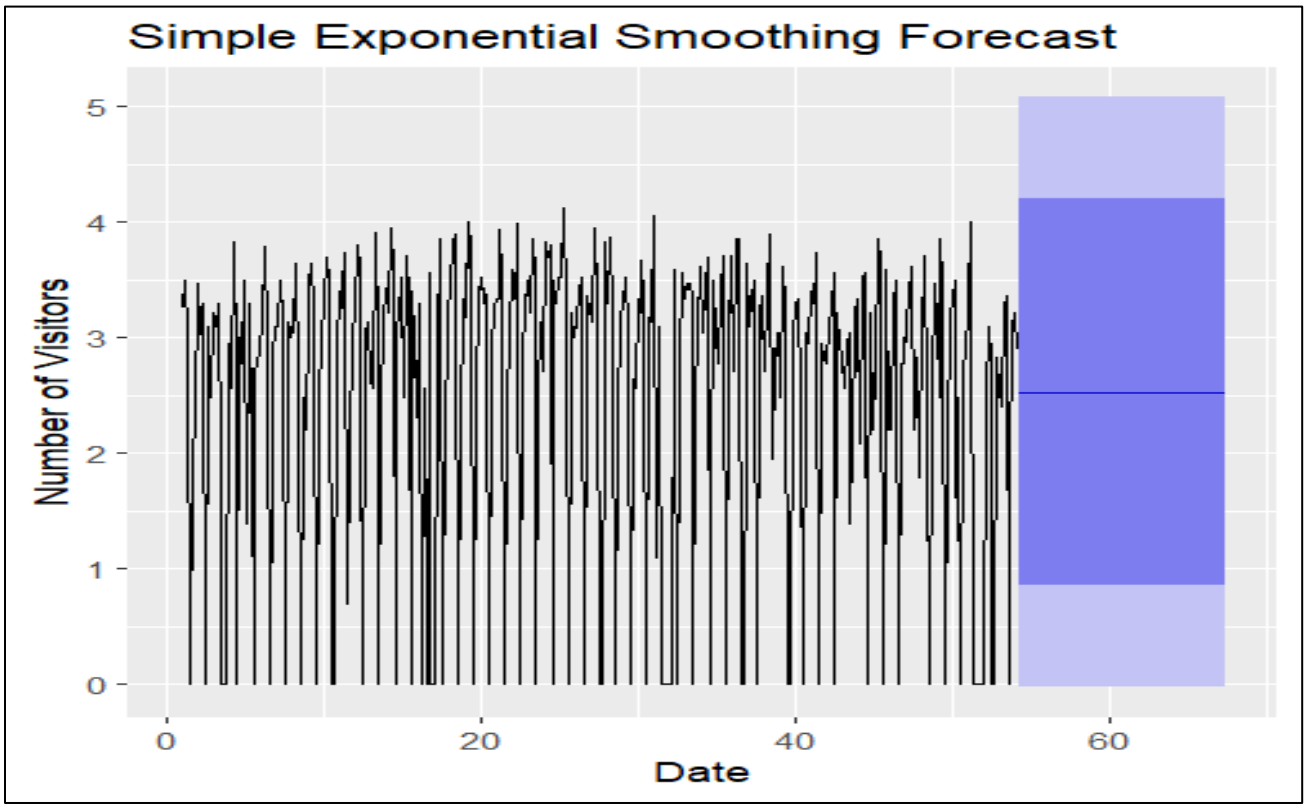


The Seasonal Naive model predicts future values based solely on the most recent seasonal data, leading to forecasts replicating historical patterns.

- **Performance Metrics**:
    - **RMSE**: 14.28
    - **MAE**: 1.09

Although this model yields the lowest RMSE, the higher MAE indicates that it oversimplifies the data, capturing fundamental, seasonal trends while missing nuanced changes.

4. Simple Exponential Smoothing (SES) Forecast



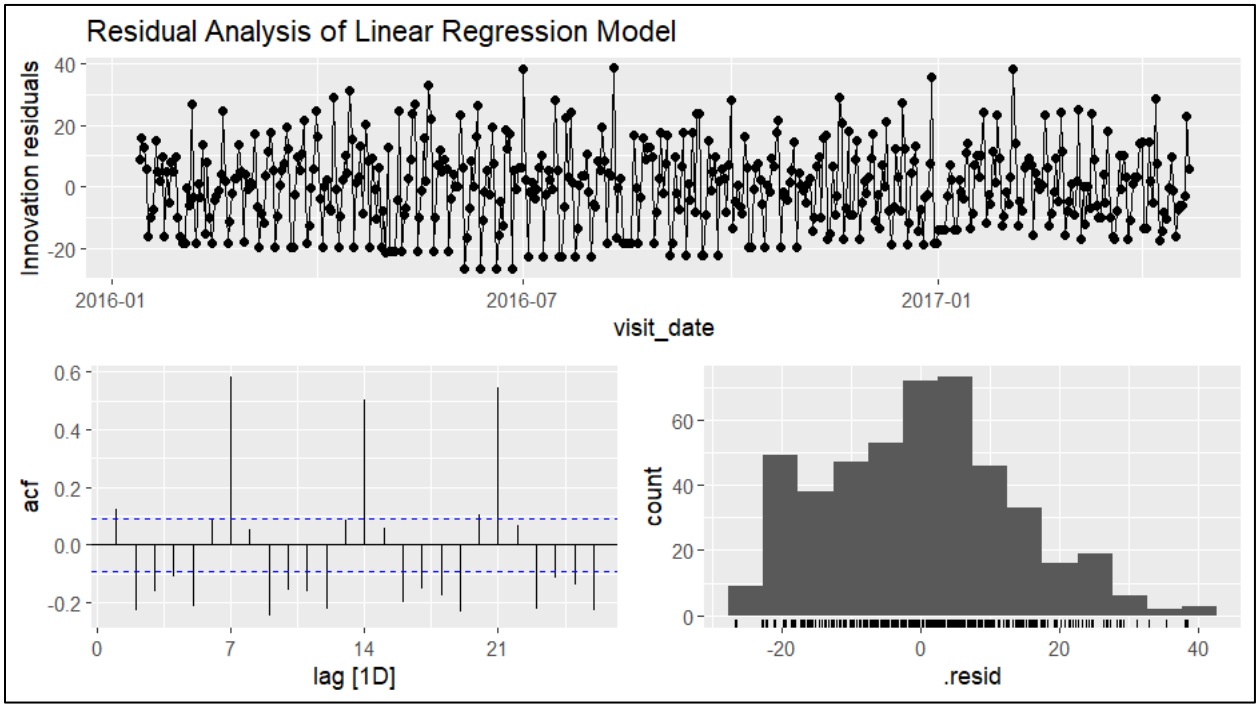
The SES model provides a smoothed forecast based on recent visitor counts without directly accounting for seasonal or trend factors.

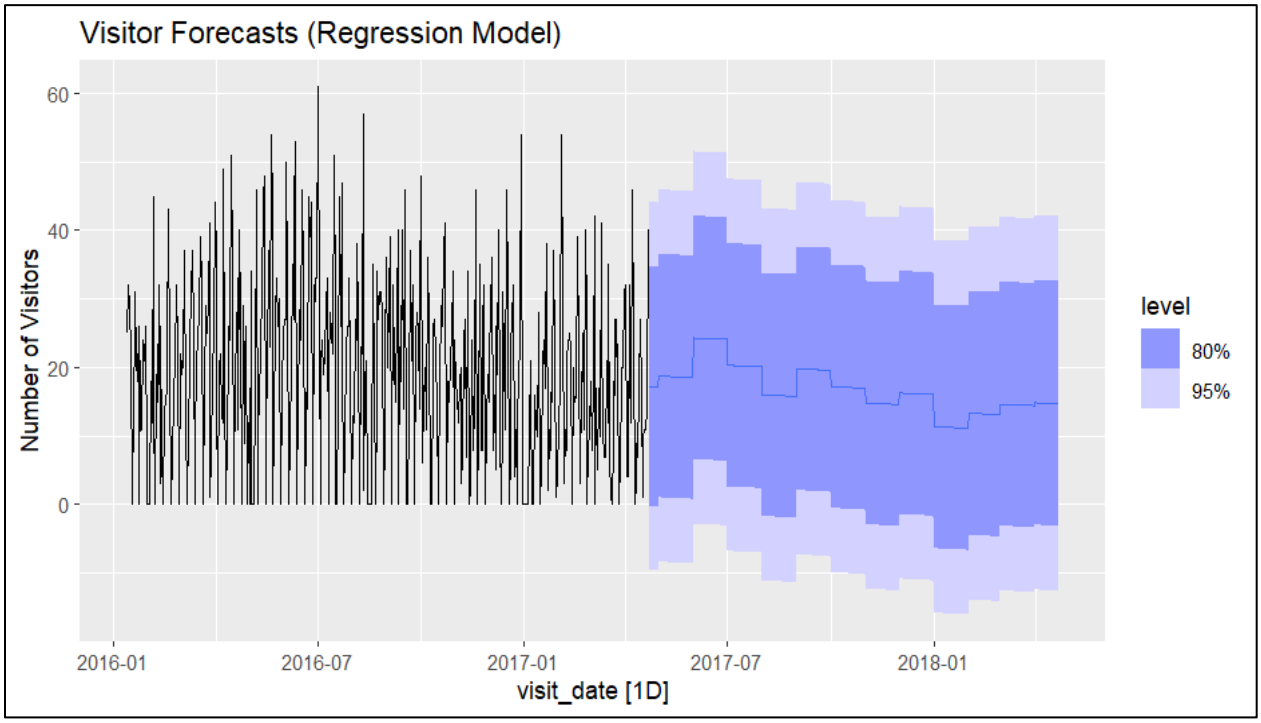
- **Performance Metrics**:
    - **RMSE**: 14.29
    - **MAE**: 1.30

While SES is effective in smoothing out short-term fluctuations, it may not adequately capture seasonality.

5. Time Series Regression

Model: Linear Regression with Time and Seasonality





The linear regression model integrates a time index with seasonal factors, producing forecasts that reflect both trends and seasonal effects.

- **Performance Metrics**:
    - **RMSE**: 12.12 (best among all models)
    - **MAE**: 9.90

The linear regression model emerges as the most reliable forecasting approach, with the lowest RMSE indicating strong predictive power. However, the relatively high MAE suggests challenges in capturing specific seasonal peaks, highlighting the importance of model selection based on the characteristics of the data.

Report of the Time Series Linear Model (TSLM) Results

```
> # Report the model results
> multiple_fit %>% report()
Series: visitors
Model: TSLM

Residuals:
     Min       1Q   Median       3Q      Max
-26.7720  -9.9659   0.2314   8.5391  38.5717

Coefficients:
                              Estimate Std. Error t value Pr(>|t|)
(Intercept)                  16.279969   2.244329   7.254 1.77e-12 ***
trend()                      -0.006861   0.005043  -1.360 0.174372
as.factor(month(visit_date))2  2.185141   2.620840   0.834 0.404858
as.factor(month(visit_date))3  3.701573   2.570019   1.440 0.150476
as.factor(month(visit_date))4  4.127803   2.678091   1.541 0.123936
as.factor(month(visit_date))5  5.706680   3.137661   1.819 0.069607 .
as.factor(month(visit_date))6 11.486906   3.145973   3.651 0.000291 ***
as.factor(month(visit_date))7  7.705842   3.099002   2.487 0.013258 *
as.factor(month(visit_date))8  3.595950   3.090926   1.163 0.245284
as.factor(month(visit_date))9  7.618111   3.122306   2.440 0.015074 *
as.factor(month(visit_date))10 5.208015   3.098096   1.681 0.093445 .
as.factor(month(visit_date))11 3.036628   3.144489   0.966 0.334710
as.factor(month(visit_date))12 4.723306   3.135578   1.506 0.132672
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 13.52 on 453 degrees of freedom
Multiple R-squared: 0.04944,    Adjusted R-squared: 0.02426
F-statistic: 1.963 on 12 and 453 DF, p-value: 0.025925
```

This section summarizes the results from the Time Series Linear Model (TSLM) applied to the restaurant visitor data.

1. Model Overview

The TSLM estimates the relationship between the number of visitors and time, incorporating seasonal effects by month.

2. Residual Analysis

The residuals show the following statistics:

- **Minimum**: -26.77
- **1st Quartile (Q1)**: -9.97
- **Median**: 0.23
- **3rd Quartile (Q3)**: 8.54
- **Maximum**: 38.57

These values indicate that most residuals are small, with some outliers. The median close to zero suggests accurate predictions on average.

3. Coefficients Interpretation

Key coefficients are:

- **(Intercept)**: **16.28**, representing expected visitors when other variables are zero.
- **Trend**: **-0.0069** (not significant, $p = 0.174$), indicating a slight downward trend.
- **Seasonal Factors** (compared to January):
    - **June**: **11.49** ($p < 0.001$) - **Significant**
    - **July**: **7.71** ($p = 0.013$) - **Significant**
    - **September**: **7.62** ($p = 0.015$) - **Significant**

June and July show significant increases in visitors, while other months have positive but non-significant effects.

4. Model Statistics

- **Residual Standard Error**: **13.52**
- **Multiple R-squared**: **0.04944** - only 4.94% of variability explained.
- **Adjusted R-squared**: **0.02426**
- **F-statistic**: **1.963** (p = 0.0259) - overall model is statistically significant.

Conclusion

The TSLM reveals trends and seasonal impacts on visitor counts, with significant effects in June and July. However, the low R-squared indicates that other influencing factors may be present, suggesting further exploration or more complex modeling could improve accuracy.

6. Forecasting Performance

In evaluating the performance of various forecasting models applied to our restaurant visitor data, we examined several metrics, including Root Mean Square Error (RMSE) and Mean Absolute Error (MAE). The results of these accuracy assessments are summarized in the table below:

Model	RMSE	MAE
ARIMA	14.45	0.90
ETS	14.32	0.77
Seasonal Naive	14.28	1.09
Simple Exponential Smoothing (SES)	14.29	1.30
Linear Regression	12.12	9.90

From this analysis, we observe that the **Linear Regression** model demonstrated the best performance, with the lowest RMSE (12.12) and MAE (9.90). This indicates that the linear regression model's predictions are generally closer to the actual visitor counts than the other models.

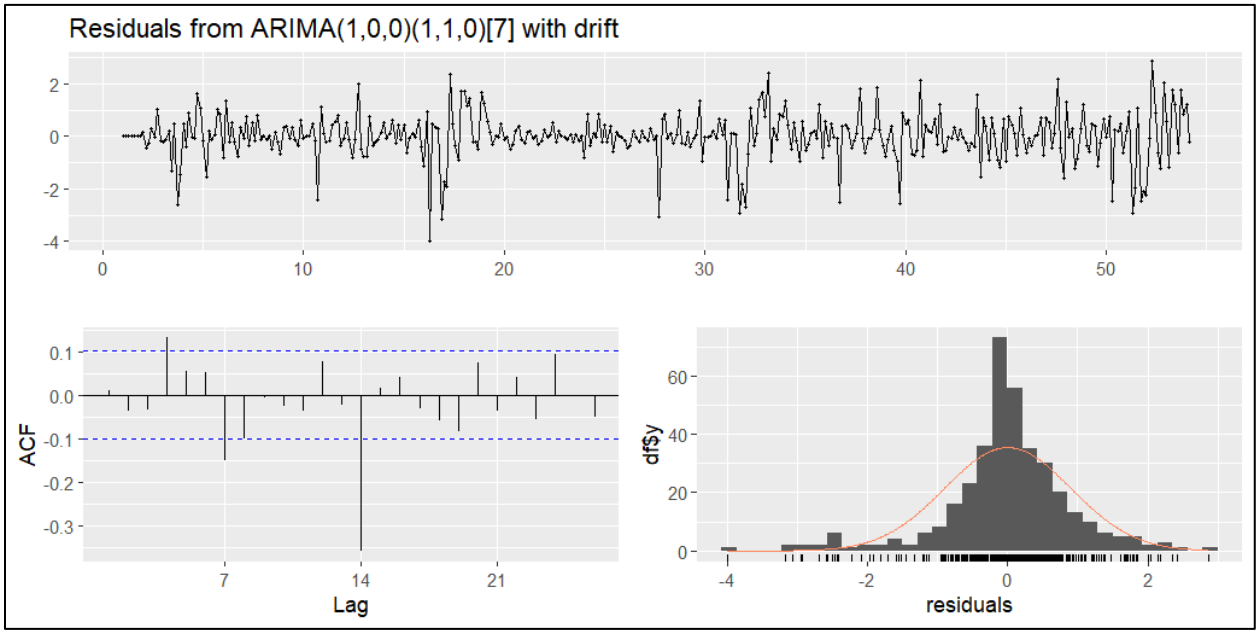
The **ETS (Exponential Smoothing State Space Model)** followed closely with an RMSE of 14.32 and an MAE of 0.77, showcasing its ability to accurately capture the underlying patterns in the data.

The **Seasonal Naive** and **SES** models had comparable RMSE values, around 14.28 and 14.29, respectively, suggesting that they perform similarly, yet not as effectively as ARIMA and ETS.

Residual Analysis

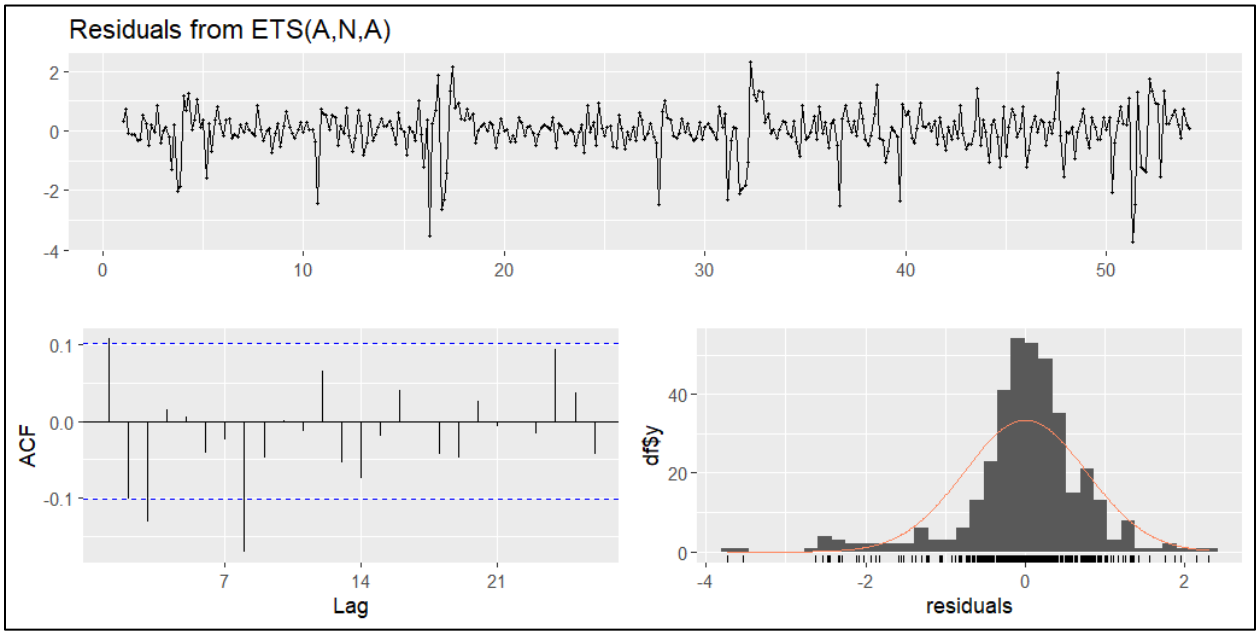
Further insights into the robustness of these models were gained through residual analysis.

1. ARIMA Model:



- The Ljung-Box test for the ARIMA model revealed a test statistic (Q*) of 75.977 with a p-value of 2.4e-11. This result indicates a significant departure from the null hypothesis, suggesting that the residuals are not independent. This lack of independence in residuals may imply that the ARIMA model could be improved by reconsidering its parameters or including additional features.

2. ETS Model:



- For the ETS model, the Ljung-Box test yielded a Q* of 32.911 with a p-value of 0.002967, indicating that residuals are also not independent in this model. While the ETS model performed well in terms of accuracy, the residual correlation suggests that further model refinement may be needed.

Conclusion

Overall, while the Linear Regression model achieved the best performance metrics, both ARIMA and ETS models demonstrated their potential, though residual analysis revealed issues with independence that might warrant further investigation. Future work could focus on addressing these residuals to enhance model robustness and improve forecast accuracy.

References

Recruit Restaurant Visitor Forecasting. (n.d.). Kaggle.

https://www.kaggle.com/competitions/recruit-restaurant-visitor-forecasting

Iamleonie. (2022, March 15). *Time Series: Interpreting ACF and PACF*. Kaggle.

https://www.kaggle.com/code/iamleonie/time-series-interpreting-acf-and-pacf

RPubs - Recruit Restaurant Visitor Forecasting. (n.d.). https://rpubs.com/arubio017/restfcst

InfluxData. (2021, December 10). InfluxDB: Open-Source Time Series Database | *InfluxData*.

https://www.influxdata.com/blog/autocorrelation-in-time-series-data/

Appendix

```
# Name: Rekha Devendra
# Statistical Forecasting Project
# Forecasting Restaurant Visitor Data for Operational Planning

# Load necessary libraries
library(dplyr)
library(ggplot2)
library(forecast)
library(tseries)
library(lubridate)
library(gridExtra)
library(fpp3)
library(readr)

# Set seed for reproducibility
set.seed(42)

# 1. Data Loading and Preprocessing
# Load the dataset
data <- read_csv("air_visit_data.csv/air_visit_data.csv")
View(data)

# Convert visit_date to Date type
data$visit_date <- as.Date(data$visit_date)

# Filter data for a specific restaurant (adjust air_store_id as needed)
```

```
chosen_restaurant_id <- "air_ba937bf13d40fb24"  # Example restaurant ID
data_filtered <- data %>% filter(air_store_id == chosen_restaurant_id)

# Check and remove any duplicates
data_filtered <- data_filtered %>% distinct()

# Check for gaps in the date range and fill missing dates
full_date_range <- data.frame(visit_date = seq(min(data_filtered$visit_date),
max(data_filtered$visit_date), by = "day"))
data_full <- left_join(full_date_range, data_filtered, by = "visit_date")

# Fill missing visitor values with 0 (restaurant closed on certain days)
data_full$visitors[is.na(data_full$visitors)] <- 0

# Check for missing values
sum(is.na(data))

# Split data into training and testing sets (80/20 split)
train_size <- round(0.8 * nrow(data_full))
train_data <- data_full[1:train_size, ]
test_data <- data_full[(train_size + 1):nrow(data_full), ]

# 2. Visualization ----------------------------------------------------

# Time Plot of Visitors Over Time
ggplot(train_data, aes(x = visit_date, y = visitors)) +
  geom_line(color = "blue", size = 1) +  # Line color and thickness
  geom_point(color = "red", size = 2) +  # Points to highlight data points
  labs(
```

```
    title = "Daily Visitors Over Time",
    x = "Date",
    y = "Number of Visitors",
    caption = "Data Source: Recruit Restaurant Visitor Forecasting Dataset"
  ) +
  theme_minimal(base_size = 14) +  # Increased base font size
  theme(
    plot.title = element_text(hjust = 0.5, face = "bold", size = 16),
    axis.title = element_text(face = "bold"),
    panel.grid.major = element_line(color = "grey80"),
    panel.grid.minor = element_blank()
  )

# ACF plot for visitors
acf_plot <- ggAcf(train_data$visitors, lag.max = 30) +
  labs(title = "ACF Plot for Visitors", x = "Lag", y = "ACF")
print(acf_plot)

# 3. Data Transformation ------------------------------------------------

# Decomposition of Visitors Time Series
decomp <- stl(ts(train_data$visitors, frequency = 7), s.window = "periodic")
autoplot(decomp) +
  labs(title = "Decomposition of Visitors Time Series")

# Log transformation of visitors (adding 1 to avoid log(0))
train_data$visitors_transformed <- log(train_data$visitors + 1)
```

```
# Convert transformed data to a time series object
ts_train <- ts(train_data$visitors_transformed, frequency = 7)

# 4. Forecasting and Analysis
# ARIMA Model
arima_model <- auto.arima(ts_train)
arima_forecast <- forecast(arima_model, h = nrow(test_data))

# Plot ARIMA Forecast
autoplot(arima_forecast) +
  ggtitle("ARIMA Model Forecast") +
  xlab("Date") +
  ylab("Number of Visitors")

# ETS Model
ets_model <- ets(ts_train)
ets_forecast <- forecast(ets_model, h = nrow(test_data))

# Plot ETS Forecast
autoplot(ets_forecast) +
  ggtitle("ETS Model Forecast") +
  xlab("Date") +
  ylab("Number of Visitors")

# Seasonal Naive Model
sn_model <- snaive(ts_train, h = nrow(test_data))
sn_forecast <- forecast(sn_model, h = nrow(test_data))
```

```
# Plot Seasonal Naive Forecast
autoplot(sn_forecast) +
  ggtitle("Seasonal Naive Model Forecast") +
  xlab("Date") +
  ylab("Number of Visitors")

# Simple Exponential Smoothing (SES)
ses_model <- ses(ts_train, h = nrow(test_data))
ses_forecast <- forecast(ses_model, h = nrow(test_data))

# Plot SES Forecast
autoplot(ses_forecast) +
  ggtitle("Simple Exponential Smoothing Forecast") +
  xlab("Date") +
  ylab("Number of Visitors")

# 5. Time Series Regression
# Create a time index for linear regression model
train_data$time_index <- 1:nrow(train_data)

# Fit the linear model
lin_model <- lm(visitors ~ time_index, data = train_data)

# Create the time index for the test data
test_data$time_index <- (nrow(train_data) + 1):(nrow(train_data) + nrow(test_data))

# Forecast using the linear regression model
lin_forecast <- predict(lin_model, newdata = test_data)
```

```
# Time Series Regression with Trend and Seasonality
visitors_ts <- data_full %>%
  as_tsibble(index = visit_date) %>%
  select(visit_date, visitors)

# Model: Linear Regression with Time and Seasonality
multiple_fit <- visitors_ts %>%
  model(TSLM(visitors ~ trend() + as.factor(month(visit_date))))

# Report the model results
multiple_fit %>% report()

# Check residuals
multiple_fit %>%
  gg_tsresiduals() +
  ggtitle("Residual Analysis of Linear Regression Model")  # Add the title

# Generate forecasts for 12 months
fc <- forecast(multiple_fit, h = "12 months")

# Plot the forecasts
visitors_ts %>%
  autoplot(visitors) +
  autolayer(fc) +
  labs(title = "Visitor Forecasts (Regression Model)", y = "Number of Visitors")
```

```
# 6. Forecasting Performance
# Calculate accuracy for ARIMA, ETS, Seasonal Naive, SES, and Linear Regression
arima_accuracy <- accuracy(arima_forecast, test_data$visitors)
ets_accuracy <- accuracy(ets_forecast, test_data$visitors)
sn_accuracy <- accuracy(sn_forecast, test_data$visitors)
ses_accuracy <- accuracy(ses_forecast, test_data$visitors)
lin_accuracy <- accuracy(lin_forecast, test_data$visitors)

# Combine accuracy results into a data frame
accuracy_results <- data.frame(
  Model = c("ARIMA", "ETS", "Seasonal Naive", "SES", "Linear Regression"),
  RMSE = c(arima_accuracy[2], ets_accuracy[2], sn_accuracy[2], ses_accuracy[2],
lin_accuracy[2]),
  MAE = c(arima_accuracy[3], ets_accuracy[3], sn_accuracy[3], ses_accuracy[3],
lin_accuracy[3])
)

# Print accuracy results
print(accuracy_results)

# 7. Residual Analysis
# Residual diagnostics for ARIMA
checkresiduals(arima_forecast)

# Residual diagnostics for ETS
checkresiduals(ets_forecast)
```