LUBS5309 Forecasting and Advanced Business Analytics

201741833

2024-05-11

Part 1: US Seasonally Adjusted Personal Consumption Expenditure

INTRODUCTION:

The analysis of US Seasonally Adjusted PCE data plays a pivotal role in understanding economic trends and informing decision-making processes in various sectors. In this report, we delve into the intricacies of forecasting PCE values using different modeling approaches. Main aim is to identify the most effective forecasting model that accurately captures the underlying patterns in the data and provides reliable predictions for future consumption expenditures.

DATA CLEANING:

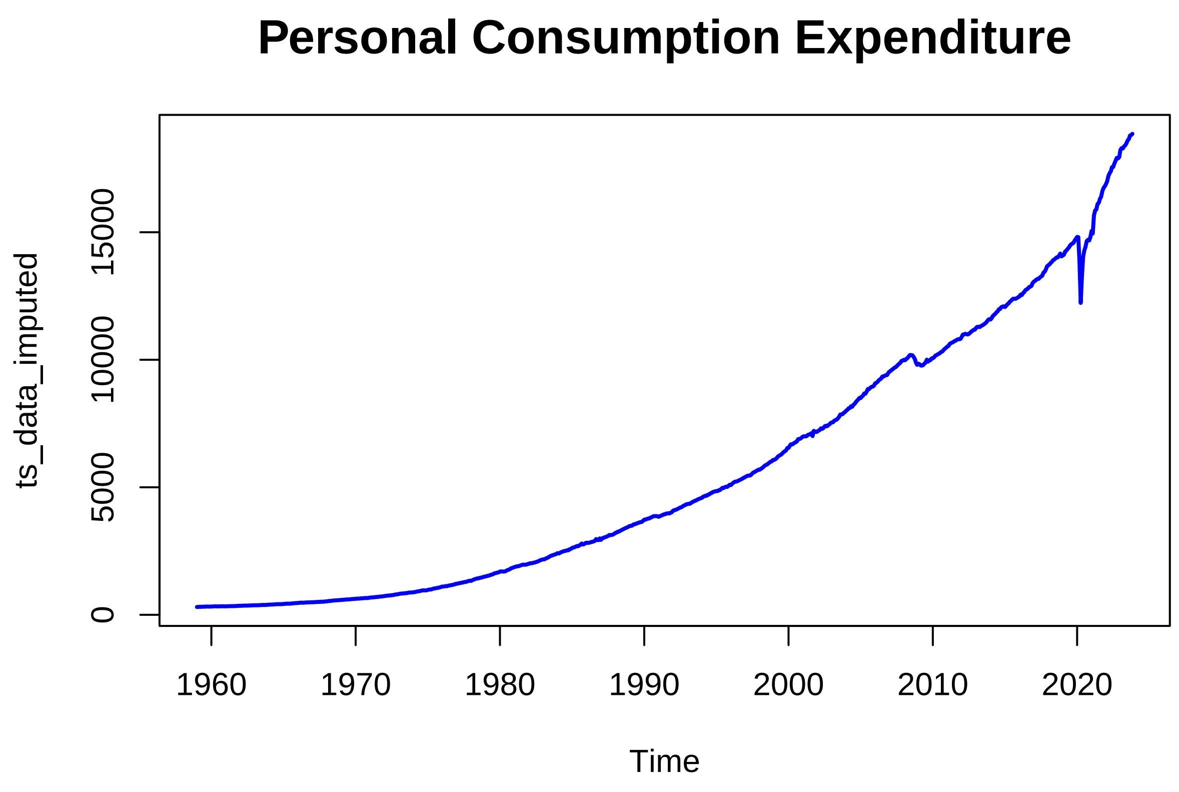
In my analysis, I employed two different preprocessing approaches: one with imputation for handling missing data and another without imputation. Initially, I leaned towards the model without imputation, assuming it would yield better results. However, upon closer examination, I discovered that the imputed model demonstrated improved performance, especially in long-term forecasting scenarios like prediction of October 2024 data.

The data preprocessing steps involved loading required libraries and the *PCE* data was read from a *CSV* file and cleaned to ensure the date column was in the correct format. The cleaned data was then converted into a time series object and missing values were imputed using *linear interpolation*. Finally, the data was split into training and testing sets based on a specified time period till *December* 2020 and from *January* 2021 respectively, allowing for model training and evaluation.

Reason for choosing Imputed Data:

1. Completeness: Imputation fills in missing values, which is crucial for time series analysis like ARIMA or Exponential Smoothing.
2. Improved Model Accuracy: Imputation of missing data can lead to more accurate estimation of model parameters and better forecasts
3. Data Integrity: Proper imputation maintains the underlying patterns in the data, such as seasonality and trend, which are essential for forecasting models to perform effectively.

```r
# Loading libraries
library(readr)
library(dplyr)
library(lubridate)
library(forecast)
library(tseries)
library(imputeTS)
library(tidyverse)
library(ggplot2)
library(scales)

PCE_data <- read.csv("PCE.csv")

PCE_data$DATE <- dmy(PCE_data$DATE)

# Converting data to time series
ts_data <- ts(PCE_data$PCE, start = c(1959, 1), end = c(2023, 11), frequency = 12)

# Imputing missing values
ts_data_imputed <- na_interpolation(ts_data)

# plot
#plot(ts_data_imputed, main = "Personal Consumption Expenditure", col='')
#title(xlab = "Time", ylab = "PCE_Imputed")
plot(ts_data_imputed,
     col = 'blue',
     lwd = 2,
     pch = 19,
     cex = 1.5)
title(main = "Personal Consumption Expenditure", cex.main = 1.5)
```



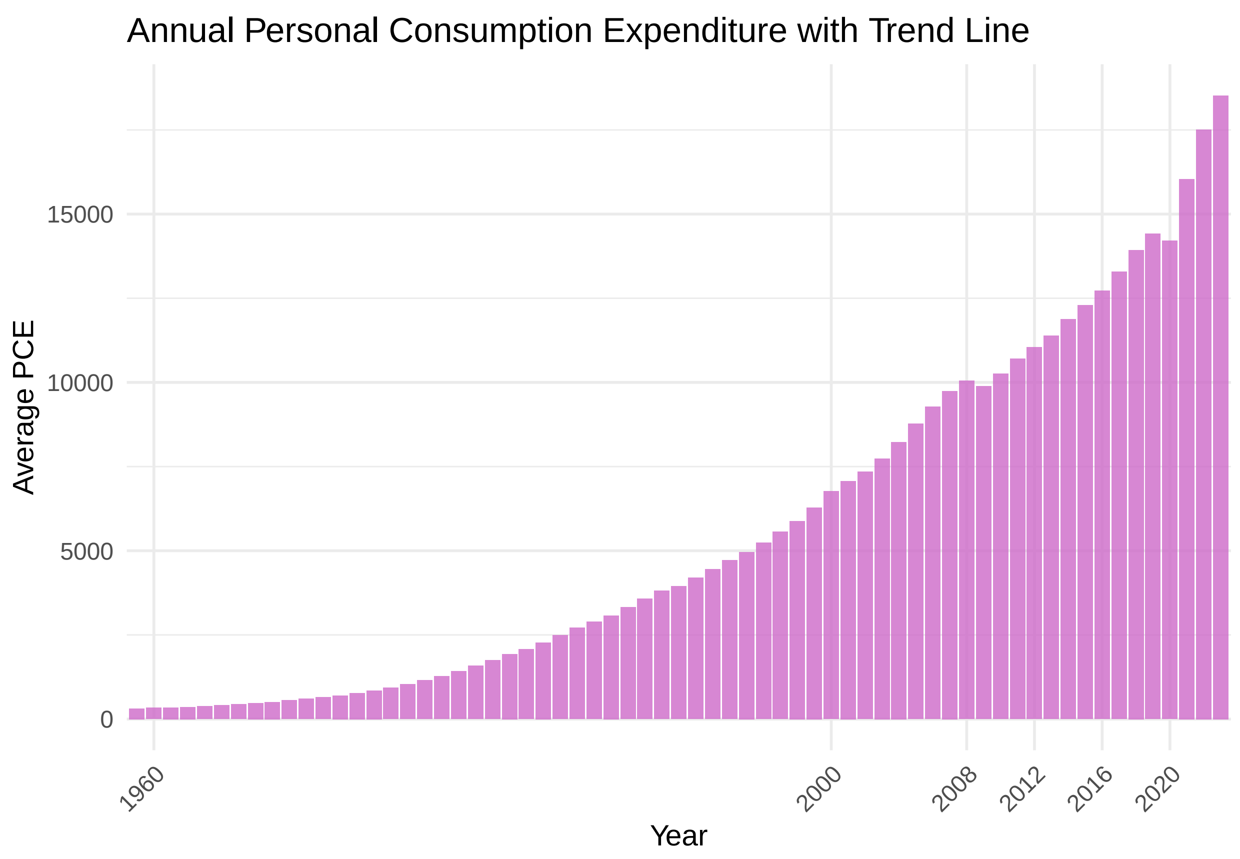
```
#title(xlab = "Time", ylab = "PCE_Imputed", cex.lab = 1.2)
```

```r
set.seed(123)
# Splitting data
train <- window(ts_data_imputed, end = c(2020, 12))
test <- window(ts_data_imputed, start = c(2021, 1))
```

The graph displays PCE data from 1960 to 2020. The data shows an upward trend, with fluctuations that might represent economic cycles. The recent decline and rapid increase around 2020 show the *economic impact* of *COVID-19*. Imputation ensures data continuity, allowing for more robust statistical analysis and forecasting. This plot is crucial for economic planning and forecasting future consumption levels.

Expenditure over the years:

```r
annual_data <- PCE_data %>%
  mutate(Year = year(DATE)) %>%
  group_by(Year) %>%
  summarize(AnnualPCE = mean(PCE, na.rm = TRUE))

ggplot(annual_data, aes(x = as.factor(Year), y = AnnualPCE)) +
  geom_bar(stat = "identity", fill = "orchid3", alpha = 0.8) +
  geom_smooth(method = "lm", color = "red", se = FALSE) +
  scale_x_discrete(breaks = c("1960","2000", "2008", "2012", "2016", "2020", "2024"),
                   labels = c("1960","2000", "2008", "2012", "2016", "2020", "2024")) +
  labs(title = "Annual Personal Consumption Expenditure with Trend Line",
       x = "Year",
       y = "Average PCE") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Model Evaluation:

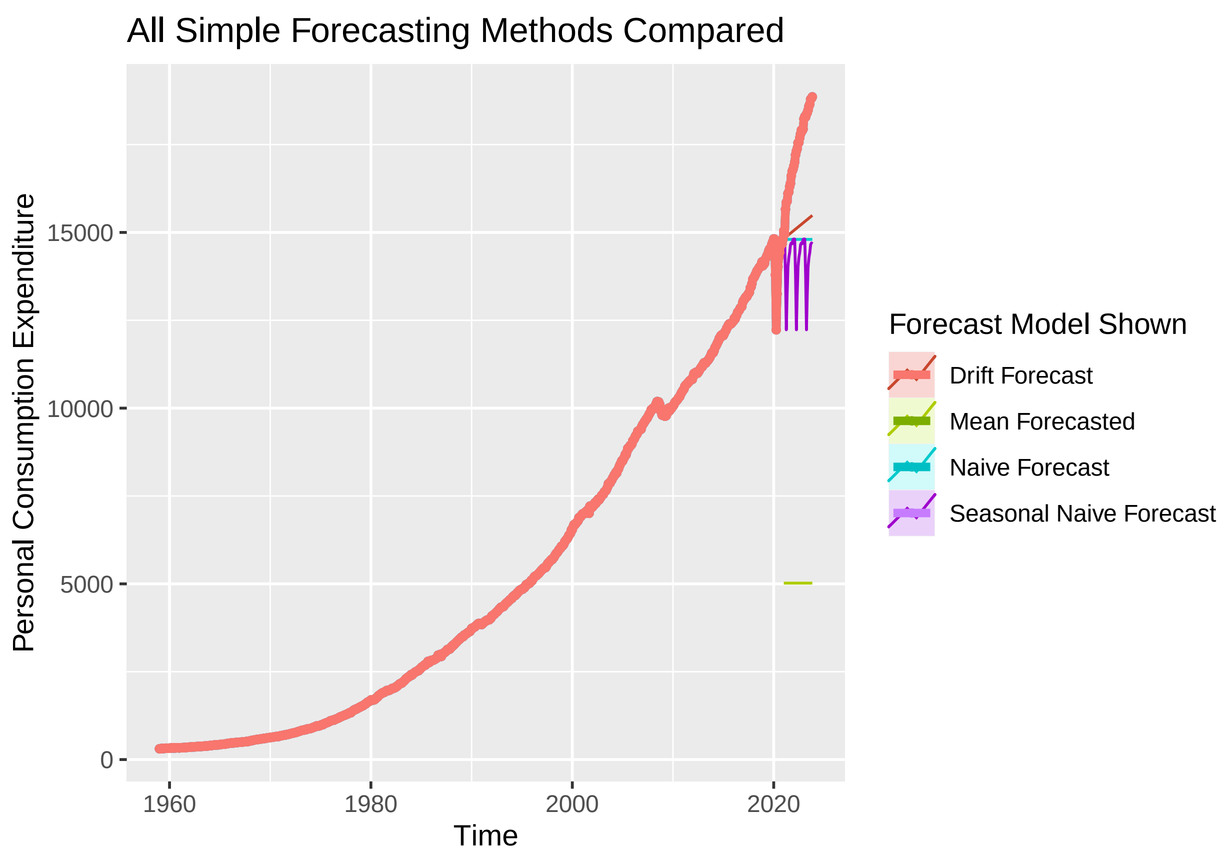
Among the simple methods average, naïve, seasonal naïve, and drift: I chose the **Drift model**. The decision was based on its relatively lower error metrics, particularly the *MAE* 25.99803(*training*), 2173.65129(*test*) & *RMSE Error* 88.74455(*training*), 2352.98650(*test*), which were significantly better than those of the mean and seasonal naïve models. The drift model's ability to account for a trend component in the data without becoming overly complex made it a standout choice.

```
# Applying simple forecasting methods
# Mean Method
mean_forecast <- meanf(train, h = length(test))

# Naïve Method
naive_forecast <- naive(train, h = length(test))

# Seasonal Naïve Method
snaive_forecast <- snaive(train, h = length(test))

# Drift Method
drift_forecast <- rwf(train, h = length(test), drift = TRUE)

# Storing forecasts in a list for comparison
forecasts <- list(Mean = mean_forecast, Naive = naive_forecast, `Seasonal Naive` = snaive_forecast, Dri

#accuracy measures for each model
results <- lapply(forecasts, accuracy, test)
# Printing the accuracy results
```

```r
print(results)
```

The graph displays forecasts from four models for Personal Consumption Expenditures data. The Mean model is unsuitable for data with trends, Naive is overly simplistic, Seasonal Naive fails to model the underlying trend, and Drift is the most aligned with historical trends but still lacks responsiveness to very recent fluctuations. It's important to select a model that captures the general trend and adapts to abrupt changes when forecasting data with potential for sudden economic shifts.

```r
error_summary <- sapply(results, function(x) x[1, "MAE"])
print(error_summary)
library(ggplot2)
autoplot(ts_data_imputed) +
  autolayer(mean_forecast, series = "Mean Forecasted", PI = FALSE) +
  geom_line(aes(color = "Mean Forecasted"), size = 1.5) +
    geom_point(aes(color = "Mean Forecasted"), size = 1) +
  autolayer(naive_forecast, series = "Naive Forecast", PI = FALSE) +
  geom_line(aes(color = "Naive Forecast"), size = 1.5) +
    geom_point(aes(color = "Naive Forecast"), size = 1) +
  autolayer(snaive_forecast, series = "Seasonal Naive Forecast", PI = FALSE) +
  geom_line(aes(color = "Seasonal Naive Forecast"), size = 1.5) +
    geom_point(aes(color = "Seasonal Naive Forecast"), size = 1) +
  autolayer(drift_forecast, series = "Drift Forecast", PI = FALSE) +
  geom_line(aes(color = "Drift Forecast"), size = 1.5) +
    geom_point(aes(color = "Drift Forecast"), size = 1) +
  ggtitle("All Simple Forecasting Methods Compared") +
  xlab("Time") + ylab("Personal Consumption Expenditure") +
  guides(colour = guide_legend(title = "Forecast Model Shown"))
```



INTERPRETING CRITERIA & RESULTS:

Comparison of Drift, ETS, and ARIMA Models:

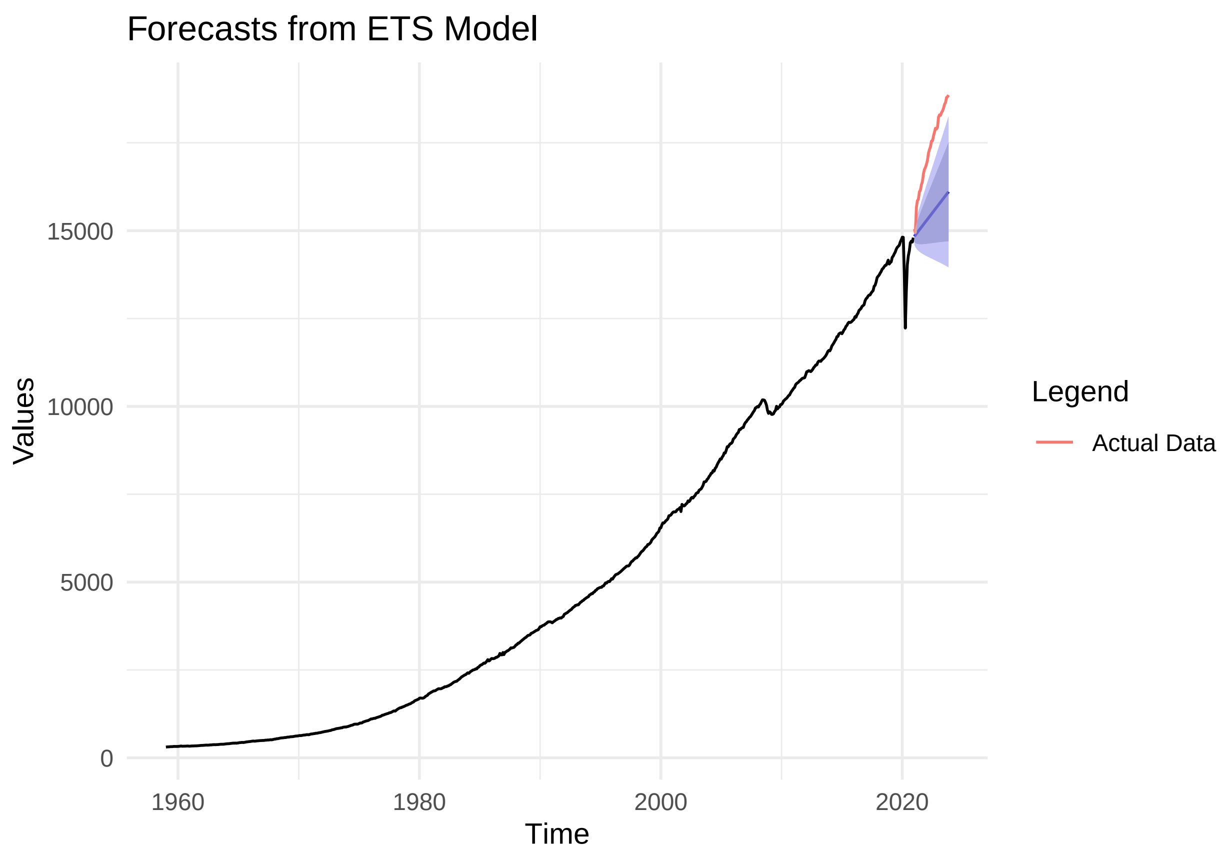
After carefully analyzing the accuracy metrics provided for the Drift, ETS, and ARIMA forecasting models, we can confidently state that each model has its own strengths. However, the detailed comparison has revealed that the **Exponential Smoothing** model is the superior choice for *minimizing deviation* in forecasting personal consumption expenditures. With the **lowest MAE and MASE**, indicating the *highest accuracy* in terms of *average errors*, and a *competitive RMSE*, the ETS model provides a *robust solution for achieving reliable and consistent predictive outcomes.* While *ARIMA* should be considered if overall fitting of the model is more *critical, for smaller-scale forecasting, the ETS model is the advisable choice.*

Exponential Smoothing Model Results:

```
ets_model <- ets(train)
ets_forecast <- forecast(ets_model, h = length(test))
ets_results <- accuracy(ets_forecast, test)
print(ets_results)
```

```
##                      ME       RMSE        MAE         MPE       MAPE       MASE
## Training set    1.76934   89.08711   20.98774  0.07629938  0.4176131 0.08528183
## Test set     1852.65661 1989.71737 1852.65661 10.46184226 10.4618423 7.52810813
##                   ACF1 Theil's U
## Training set 0.1866304        NA
## Test set     0.8843701   10.4989
```

```
autoplot(ets_forecast, series = "ETS Forecast") +
  autolayer(test, series = "Actual Data") +
  ggtitle("Forecasts from ETS Model") +
  xlab("Time") + ylab("Values") +
  geom_line() +
  geom_ribbon(aes(ymin = ets_forecast$lower[, "80%"], ymax = ets_forecast$upper[, "80%"], x = time(test
  guides(colour = guide_legend(title = "Legend")) +
  theme_minimal()
```

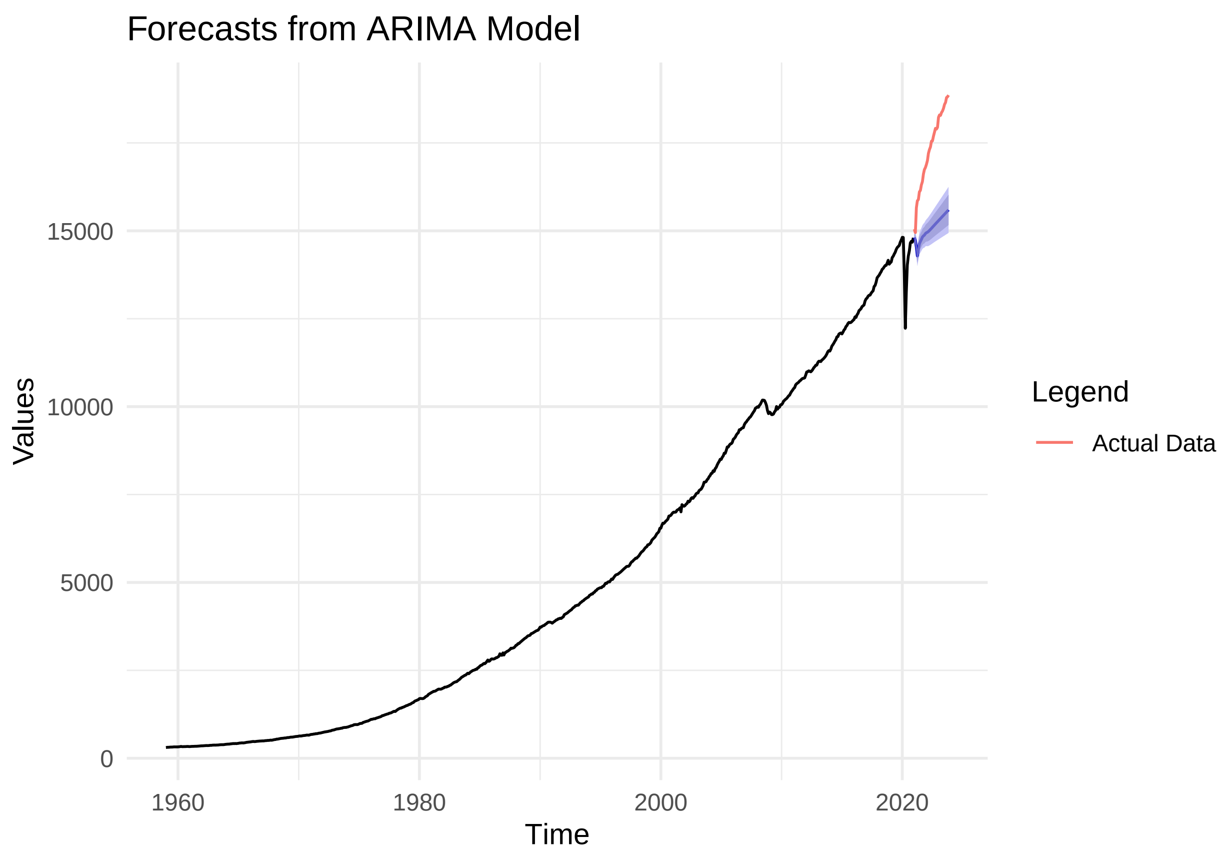


ARIMA MODEL:

```
arima_model <- auto.arima(train)
arima_forecast <- forecast(arima_model, h = length(test))
arima_results <- accuracy(arima_forecast, test)
print(arima_results)
```

```
##                        ME       RMSE        MAE        MPE       MAPE       MASE
## Training set    4.21399   78.79724   24.31436  0.2041348  0.5296472 0.09879929
## Test set     2259.49011 2395.60337 2259.49011 12.7948751 12.7948751 9.18124046
##                      ACF1 Theil's U
## Training set -0.004408696        NA
## Test set      0.855798856  12.68468
```

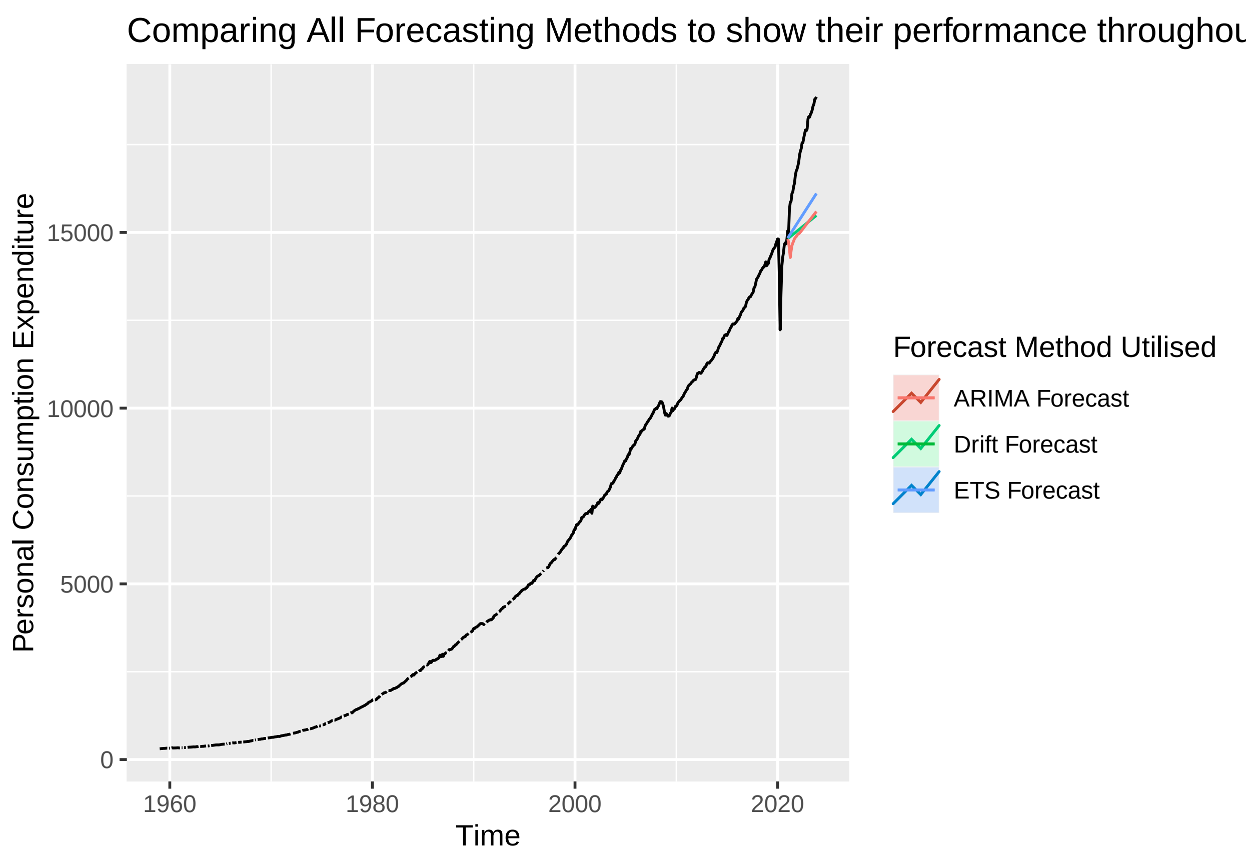
```
autoplot(arima_forecast, series = "ARIMA Forecast") +
  autolayer(test, series = "Actual Data") +
  ggtitle("Forecasts from ARIMA Model") +
  xlab("Time") + ylab("Values") +
  geom_line() +
  geom_ribbon(aes(ymin = arima_forecast$lower[, "80%"], ymax = arima_forecast$upper[, "80%"], x = time(
  guides(colour = guide_legend(title = "Legend")) +
  theme_minimal()
```



Both models track the upward trend of the data closely up to the current period and project continued growth. The ETS model appears slightly more aggressive in its future values, suggesting higher values at the end of the forecast period compared to the ARIMA model. The confidence intervals in both charts reflect increasing uncertainty as the forecasts extend into the future, which is typical in time series forecasting.

Combined Graph:

```
autoplot(ts_data) +
  autolayer(drift_forecast, series = "Drift Forecast", PI = FALSE) +
  autolayer(ets_forecast$mean, series = "ETS Forecast", PI = FALSE) +
  autolayer(arima_forecast$mean, series = "ARIMA Forecast", PI = FALSE) +
  ggtitle("Comparing All Forecasting Methods to show their performance throughout") +
  xlab("Time") + ylab("Personal Consumption Expenditure") +
  guides(colour = guide_legend(title = "Forecast Method Utilised") )
```



Combined Results of all the models:

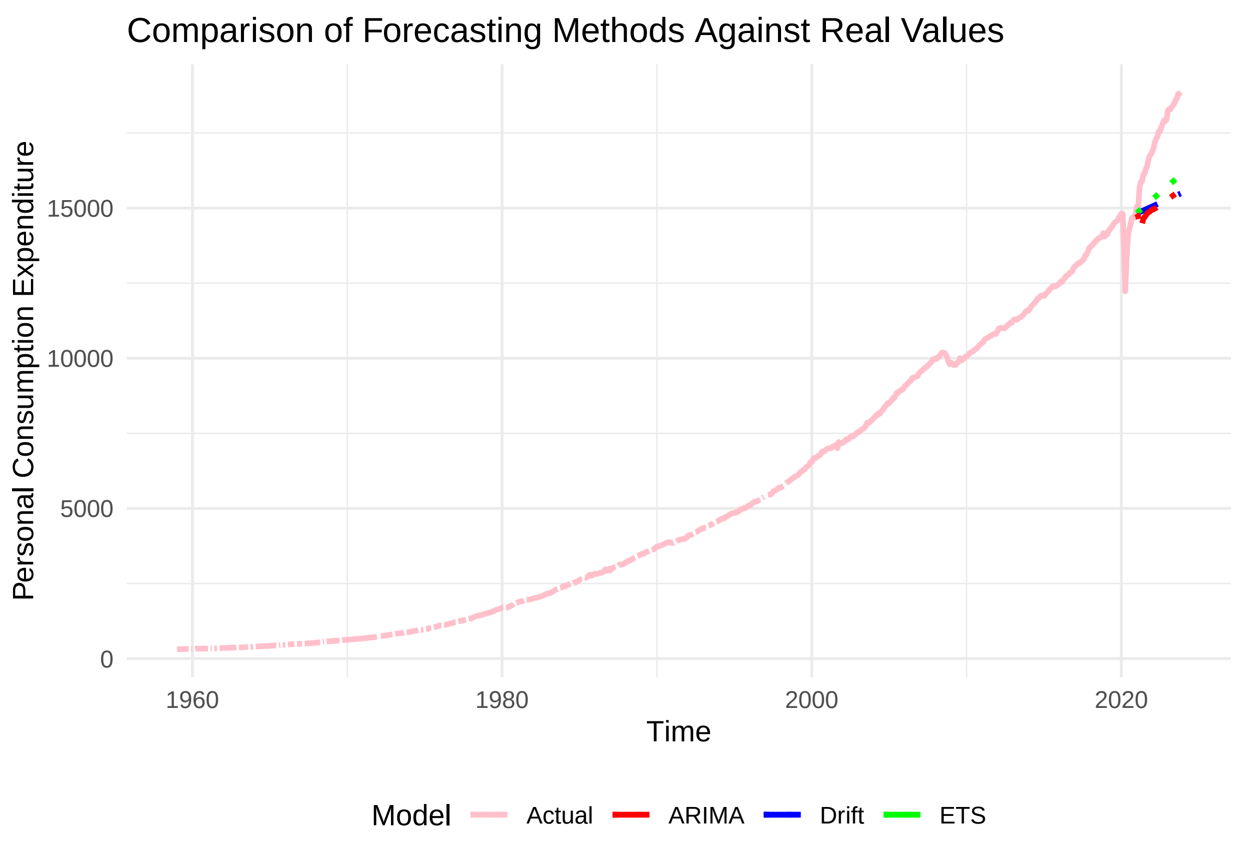
```
all_results <- rbind(
  Mean = results$Mean[1,],
  Naive = results$Naive[1,],
  `Seasonal Naive` = results$`Seasonal Naive`[1,],
  Drift = results$Drift[1,],
  ETS = ets_results[1,],
  ARIMA = arima_results[1,]
)
print(all_results)
```

```
##                          ME       RMSE        MAE           MPE        MAPE
## Mean           1.620902e-13 4419.05246 3821.89274 -267.73758711 303.0024889
## Naive          1.950969e+01   90.86376   30.16514    0.51719610   0.6535283
## Seasonal Naive 2.276901e+02  309.15839  246.09857    6.00061349   6.1596836
## Drift          6.443964e-14   88.74455   25.99803   -0.90519196   1.2165851
## ETS            1.769340e+00   89.08711   20.98774    0.07629938   0.4176131
## ARIMA          4.213990e+00   78.79724   24.31436    0.20413485   0.5296472
##                       MASE         ACF1 Theil's U
## Mean           15.52992694  0.995731451        NA
## Naive           0.12257341  0.199895131        NA
## Seasonal Naive  1.00000000  0.900478896        NA
## Drift           0.10564071  0.199895131        NA
## ETS             0.08528183  0.186630373        NA
## ARIMA           0.09879929 -0.004408696        NA
```

PLOTTING PREDICTION Vs. REAL VALUES:

The graph illustrates three models against actual historical PCE data. The actual data follows a *long-term upward trend* from 1960 to the early 2020*s* with a *sharp rise toward the end*, which is characterized by recent economic fluctuations. *ARIMA and ETS* models are *superior* in adjusting to the PCE's volatile changes compared to the simpler *Drift model*, as the graph shows.

```
library(ggplot2)
library(forecast)


actual_data <- data.frame(Date = time(ts_data),
                          PCE = as.numeric(ts_data),
                          Model = "Actual")


drift_data <- data.frame(Date = time(ts_data)[(length(ts_data) - length(drift_forecast$mean) + 1):lengt
                         PCE = drift_forecast$mean,
                         Model = "Drift")


ets_data <- data.frame(Date = time(ts_data)[(length(ts_data) - length(ets_forecast$mean) + 1):length(ts
                       PCE = ets_forecast$mean,
                       Model = "ETS")


arima_data <- data.frame(Date = time(ts_data)[(length(ts_data) - length(arima_forecast$mean) + 1):lengt
                         PCE = arima_forecast$mean,
                         Model = "ARIMA")


ggplot() +
  geom_line(data = actual_data, aes(x = Date, y = PCE, colour = Model), size = 1) +
  geom_line(data = drift_data, aes(x = Date, y = PCE, colour = Model), linetype = "dashed", size = 1) +
  geom_line(data = ets_data, aes(x = Date, y = PCE, colour = Model), linetype = "dotted", size = 1) +
  geom_line(data = arima_data, aes(x = Date, y = PCE, colour = Model), linetype = "dotdash", size = 1) 
  scale_color_manual(values = c("Actual" = "pink", "Drift" = "blue", "ETS" = "green", "ARIMA" = "red"))
  labs(title = "Comparison of Forecasting Methods Against Real Values",
       x = "Time",
       y = "Personal Consumption Expenditure",
       color = "Model") +
  theme_minimal() +
  theme(legend.position = "bottom")
```



Compared to earlier results, this graph enables a direct visual comparison of how each model projects against the actual data up to the most recent years included in the study.

ESTIMATING OCTOBER 2024:

The forecasted value of $19,258.51$ for *October 2024* was computed based on historical data patterns and extrapolated using the chosen ETS model. The model's ability to adapt its parameters to changes in trend and seasonality helps in generating a more accurate forecast, though it's important to regularly update and recalibrate the model with new data to refine future forecasts.

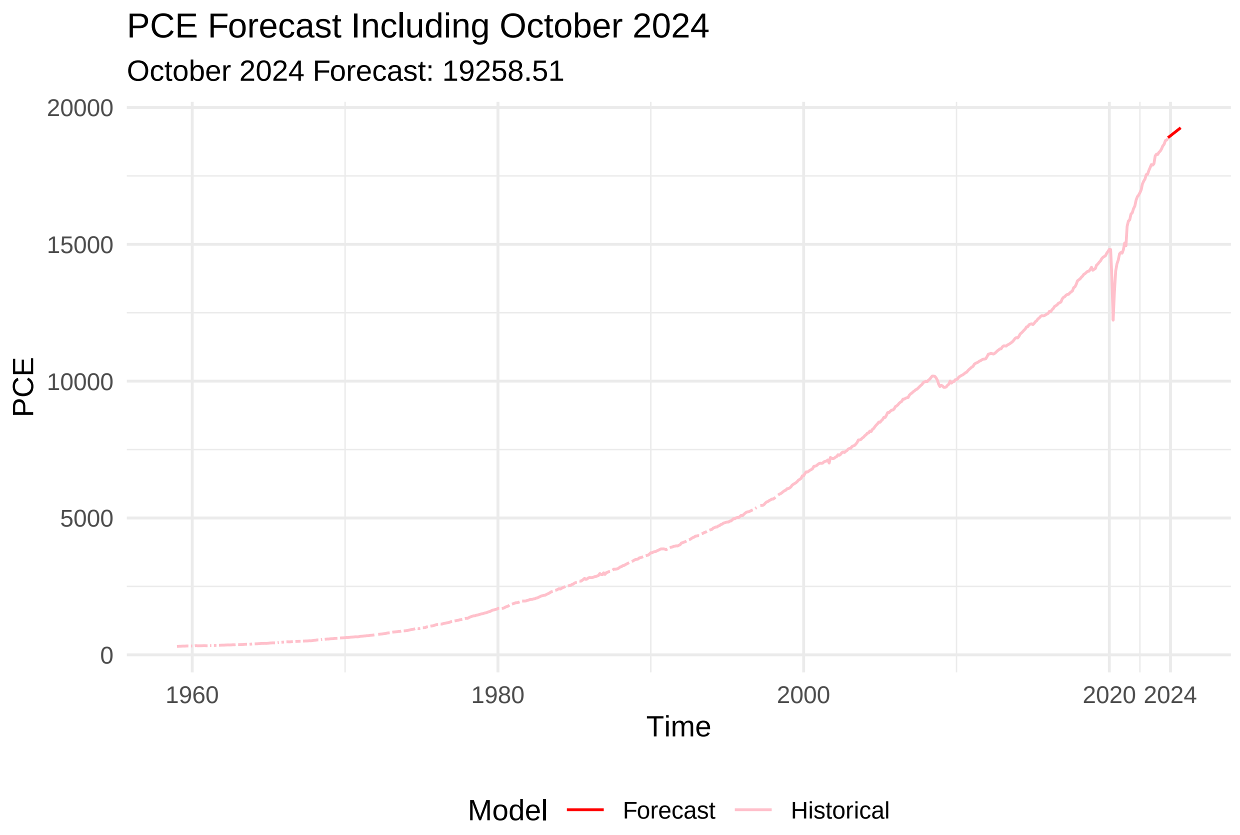
In this case, the forecast extending to October 2024 shows a sharp increase, which might reflect expectations of continued economic growth or other macroeconomic factors influencing consumer behavior. It's crucial to consider that such projections can be subject to significant uncertainty, especially when extending several years into the future, as they are sensitive to changes in economic conditions, policy decisions, and other external shocks.

```r
library(forecast)
library(ggplot2)
library(scales)
last_known_date <- as.Date("2023-11-01")
forecast_date <- as.Date("2024-10-01")
months_to_forecast <- as.integer((year(forecast_date) - year(last_known_date)) * 12 +
                                 (month(forecast_date) - month(last_known_date)))

ets_model <- ets(ts_data)

future_forecast <- forecast(ets_model, h = months_to_forecast)
```

```r
october_2024_forecast <- future_forecast$mean[length(future_forecast$mean)]

print(paste("October 2024 Forecast:", october_2024_forecast))
```

```
## [1] "October 2024 Forecast: 19258.5127494619"
```

```r
start_date <- as.Date("1959-01-01")
historical_dates <- seq(from = start_date, by = "month", length.out = length(ts_data))

historical_data <- data.frame(
  Date = historical_dates,
  PCE = as.numeric(ts_data),
  Model = "Historical"
)

forecast_dates <- seq(from = last_known_date, by = "month", length.out = months_to_forecast)
forecast_data <- data.frame(
  Date = forecast_dates,
  PCE = future_forecast$mean,
  Model = "Forecast"
)

combined_data <- rbind(historical_data, forecast_data)

ggplot(combined_data, aes(x = Date, y = PCE, color = Model)) +
  geom_line() +
  geom_point(data = subset(forecast_data, Date == forecast_date), aes(x = Date, y = PCE), color = "red"
  scale_color_manual(values = c("Historical" = "pink", "Forecast" = "red")) +
  scale_x_date(breaks = as.Date(c("1960-01-01", "1980-01-01", "2000-01-01", "2020-01-01", "2024-01-01")
               labels = date_format("%Y")) +
  labs(title = "PCE Forecast Including October 2024",
       subtitle = sprintf("October 2024 Forecast: %.2f", october_2024_forecast),
       x = "Time", y = "PCE",
       color = "Model") +
  theme_minimal() +
  theme(legend.position = "bottom")
```



This forecast should be treated as a guideline, with awareness that actual future values may differ due to unforeseen circumstances. Stakeholders should consider incorporating a range of scenarios and sensitivity analyses to better understand potential risk and variability in future PCE values.

ONE STEP AHEAD ROLLING FORECAST:

The visualization provides a clear, comprehensive view of how each forecasting model performs over time against the actual PCE data. It supports the decision-making process for selecting a forecasting method based on specific criteria such as accuracy, responsiveness, and computational simplicity. For stakeholders or policymakers, this analysis underscores the importance of regular model evaluation and updates, particularly in dynamic economic environments.

Insight of the Graphical Analysis:

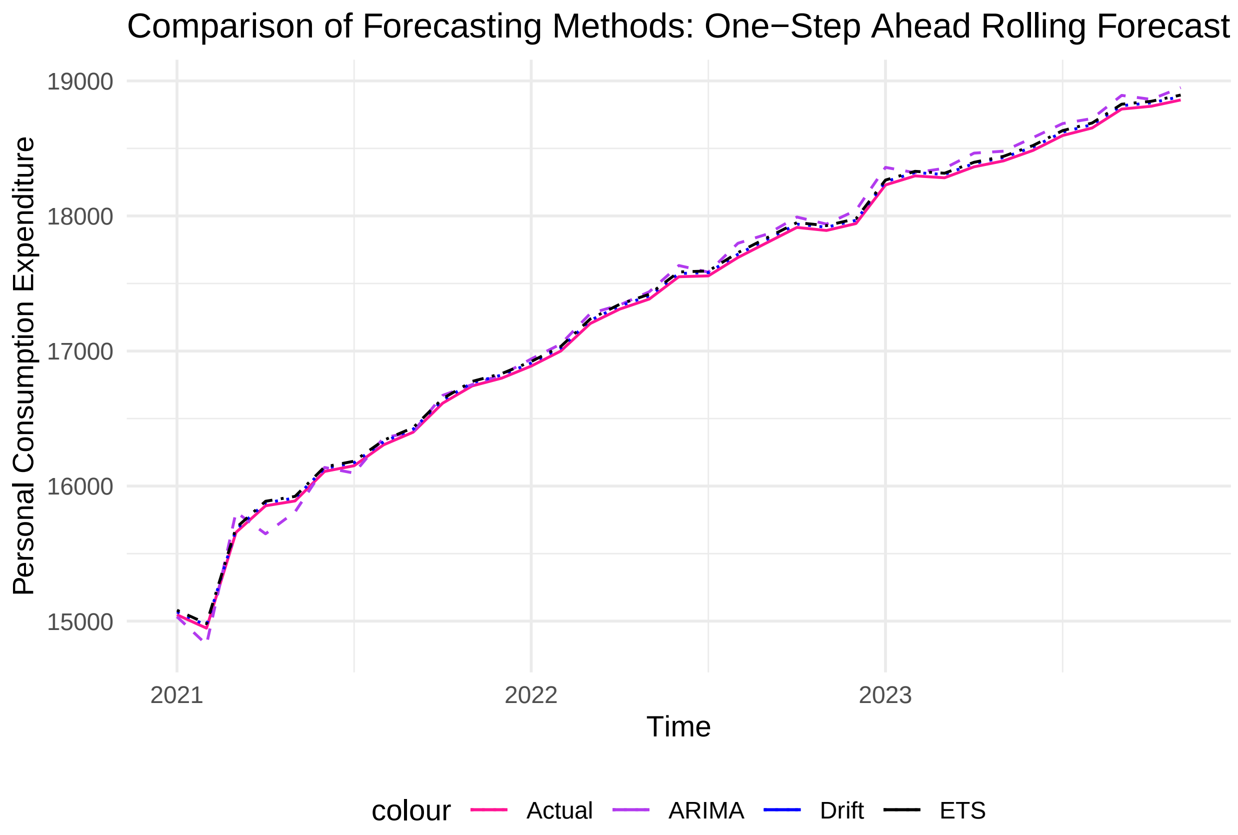
1. ARIMA : ARIMA models closely follow actual data trends, making them adept at handling data with trends and seasonal patterns.
2. Drift Model: Similar to ARIMA but more variable, drift method is effective for linear trend data.
3. ETS: This model works well for forecasting with exponential decay assumptions in data weights over time, providing a smooth curve that aligns with actual data trends.

The graph shows all models performing well in capturing the upward trend in PCE from 2021 to 2023. This indicates robustness in each model's capability to adapt to changing economic conditions in the short term. The *slight divergences* between the models and actual data around the *late 2021* and *early 2022* mark suggest varying levels of sensitivity to abrupt changes in the *economic environment, potentially reflecting the real-world impacts of policy changes or market shifts.*

For real-world application, such as budget planning or economic forecasting, this graph illustrates the utility of employing multiple forecasting models to gauge a range of potential future scenarios. By comparing these models side-by-side, analysts can assess which model might best suit their specific needs based on historical accuracy and responsiveness to recent data.

The use of a rolling forecast method enhances the responsiveness of each model by continuously integrating the latest available data. This approach is particularly useful in environments where data patterns and relationships between variables frequently change.

```r
library(forecast)
library(ggplot2)

roll_forecast <- function(model_func, params = list()) {
  sapply(1:length(test), function(i) {
    current_train <- window(ts_data, end = time(test)[i])
    model <- do.call(model_func, c(list(current_train), params))
    forecast <- forecast(model, h = 1)
    return(forecast$mean)
  })
}

arima_rolling <- roll_forecast(auto.arima)
drift_rolling <- roll_forecast(rwf, list(drift = TRUE))
ets_rolling <- roll_forecast(ets)


forecast_data <- data.frame(
  Date = time(test),
  ARIMA = arima_rolling,
  Drift = drift_rolling,
  ETS = ets_rolling,
  Actual = as.numeric(test)
)

p <- ggplot(forecast_data, aes(x = Date)) +
  geom_line(aes(y = Actual, colour = "Actual")) +
  geom_line(aes(y = ARIMA, colour = "ARIMA"), linetype = "dashed") +
  geom_line(aes(y = Drift, colour = "Drift"), linetype = "dotted") +
  geom_line(aes(y = ETS, colour = "ETS"), linetype = "dotdash") +
  scale_color_manual(values = c("Actual" = "deeppink", "ARIMA" = "darkorchid2", "Drift" = "blue1", "ETS
  labs(title = "Comparison of Forecasting Methods: One-Step Ahead Rolling Forecast",
       x = "Time", y = "Personal Consumption Expenditure") +
  theme_minimal() +
  theme(legend.position = "bottom")

print(p)
```



To capitalize on the insights provided by this analysis, it is recommended to:

Continuously monitor model performance against newly available data.Combine model forecasts to create an ensemble forecast that may reduce potential forecast errors.Conduct periodic model recalibrations to adjust for new economic conditions or structural changes in the market.This comprehensive approach ensures that forecasts remain relevant and reliable, aiding in better-informed decision-making processes that are crucial for strategic planning in personal consumption expenditure management.

Conclusion:

This analysis evaluated various forecasting models to identify the most effective approach for predicting future US Seasonally Adjusted Personal Consumption Expenditure (PCE) data. The Drift model performed well among simple techniques, while the Exponential Smoothing (ETS) model showed superior performance. The analysis also provided valuable insights into each model's ability to capture underlying trends and adapt to economic fluctuations. The practical implications of the analysis were highlighted for decision-makers, albeit with acknowledgment of inherent forecast uncertainty. Recommendations for continuous model evaluation and ensemble forecasting were made, offering stakeholders actionable insights for strategic planning in personal consumption expenditure management.

PART 2: Analyzing Customer Sentiments in Hotel Reviews: Key Drivers of Satisfaction and Dissatisfaction

INTRODUCTION:

In the highly competitive hospitality industry, understanding customer sentiment and the factors that influence satisfaction and dissatisfaction is crucial for maintaining and improving service quality. By analyzing customer reviews, hotels can gain insights into the aspects of their service that most significantly impact guest experiences. This report utilizes sentiment analysis based on the NRC emotion lexicon to dissect customer reviews from a hotel dataset.This approach helps in pinpointing the top factors that contribute to guest satisfaction and dissatisfaction, offering actionable intelligence that can be used to enhance overall customer service.

DATA PREPROCESSING AND CLASSIFYING REVIEWS:

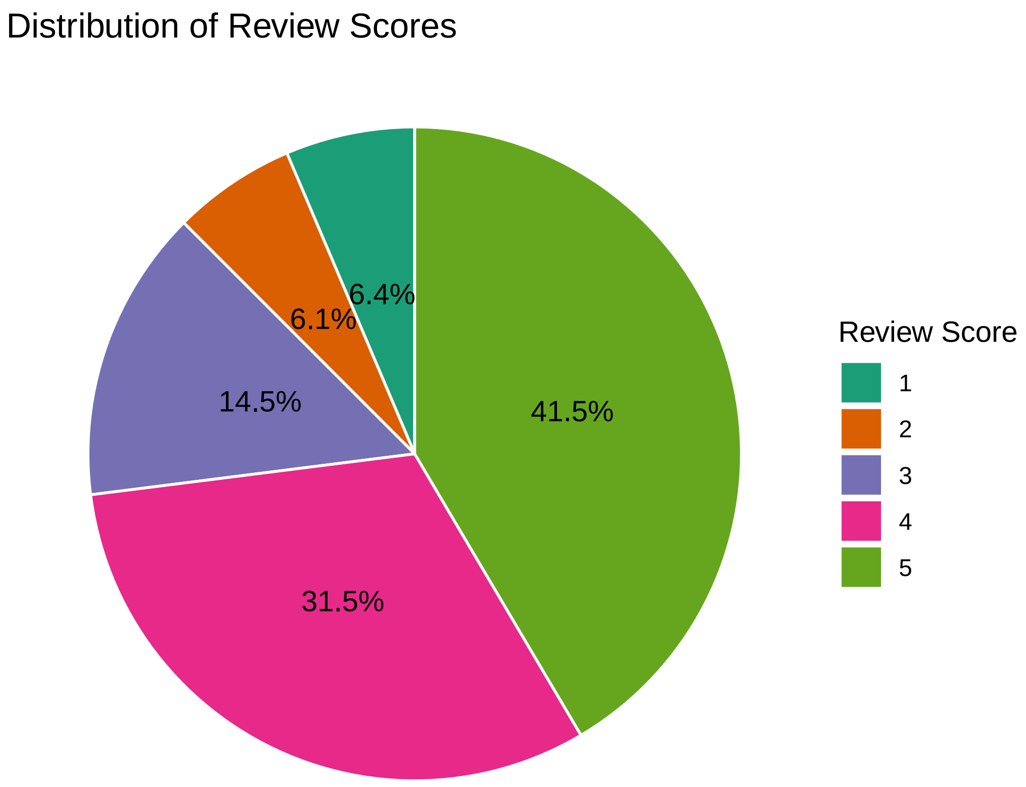
The process starts with the preparation of the dataset, HotelSData.csv, which contains reviews from customers along with their ratings on a Likert scale from 1 (low satisfaction) to 5 (high satisfaction). For the purpose of analysis, sample 2,000 reviews to ensure the computations are manageable and to mitigate any performance issues that might arise with a larger dataset. This is implemented using the sample_n() function from the dplyr package in R, which facilitates the random selection of a specified number of cases from a dataset. To ensure that the sample is reproducible set.seed() function is used with last three digits of my *Student ID* 833, which initializes the random number generator to a fixed state.

```r
library(dplyr)
library(tm)
library(topicmodels)
library(ldatuning)
library(ggplot2)
library(wordcloud)
library(RColorBrewer)
library(syuzhet)

reviews <- read.csv("HotelsData.csv")
set.seed(833)
sample_reviews <- sample_n(reviews, 2000)
```

```r
sample_reviews$ReviewType <- ifelse(sample_reviews$Review.score > 3, "Positive", "Negative")
```

Reviews were classified as *'Positive' or 'Negative'* based on the rating score. Positive reviews have a rating above 3, indicating a favorable experience. Negative reviews have a rating of 3 or below, indicating a neutral to unsatisfactory experience. *ifelse* statement in *R* is used to execute the classification. Categorizing reviews as positive or negative allows for tailored analysis and identification of impactful factors. This helps in drawing actionable insights for operational improvements and strategic decision-making. Additionally, setting a clear threshold for classification ensures consistency and objectivity in analyzing customer satisfaction through textual data.

```r
library(ggplot2)

review_counts <- table(sample_reviews$Review.score)
review_percentages <- prop.table(review_counts) * 100


review_data <- data.frame(ReviewScore = as.factor(names(review_counts)),
                          Count = as.numeric(review_counts),
                          Percentage = as.numeric(review_percentages))


ggplot(review_data, aes(x = "", y = Count, fill = ReviewScore)) +
  geom_bar(stat = "identity", color = "white") +
  geom_text(aes(label = paste0(round(Percentage, 1), "%")), position = position_stack(vjust = 0.5), siz
  coord_polar("y", start = 0) +
  theme_void() +
  theme(legend.position = "right") +
  scale_fill_brewer(palette = "Dark2", name = "Review Score") +
  ggtitle("Distribution of Review Scores")
```



The pie chart shows that the majority of reviews are 5-star, indicating high customer satisfaction. Scores 1 through 3 make up smaller segments, with score 1 being notably less frequent. Overall, the reception seems **predominantly positive**.

STEPS FOLLOWED TO PROCEED WITH ANALYSIS:

Text preprocessing is vital in analyzing textual data, especially for *unstructured data like customer reviews*. It involves multiple stages that refine the data by removing unnecessary components and standardizing the text for further analysis.

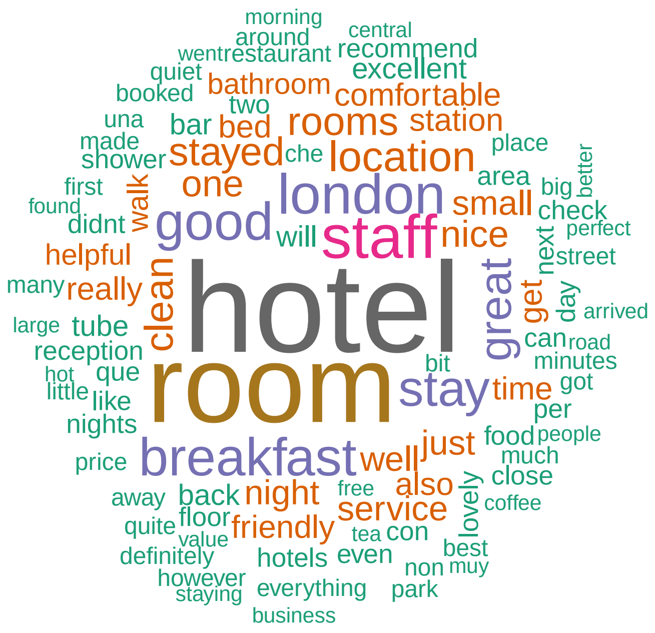
The process of creating a text corpus involves converting text data into a structured format. This allows for the application of various text processing functions, such as converting text to lowercase, removing punctuation, numbers and stop words, and stripping extra white spaces. Filtering out empty documents ensures that subsequent analysis stages operate on data that contains actual content. Preprocessing is crucial for cleaning and standardizing text, reducing noise and focusing on meaningful text. It improves accuracy and computational efficiency, leading to reliable outcomes in topic modeling and sentiment analysis.

```r
corpus <- Corpus(VectorSource(sample_reviews$Text.1))
corpus <- tm_map(corpus, content_transformer(tolower))
corpus <- tm_map(corpus, removePunctuation)
corpus <- tm_map(corpus, removeNumbers)
corpus <- tm_map(corpus, removeWords, stopwords("english"))
corpus <- tm_map(corpus, stripWhitespace)
corpus <- corpus[sapply(corpus, function(x) length(unlist(strsplit(as.character(x), " "))) > 0)]
```

Visualising with wordcount:

A word cloud generated from hotel reviews dataset can provide valuable insights about guest satisfaction. Based on the most frequently mentioned words, we can identify prominent themes such as room quality, staff interaction, and location. Positive adjectives like "good," "clean," and "comfortable" suggest that many guests have had satisfying stays while terms like "breakfast" highlight specific hotel services that impact guest experiences. Utilizing the word cloud can help in tailoring services more closely to customer expectations and enhancing overall satisfaction.

```
wordcloud(words = unlist(strsplit(as.character(corpus), " ")), max.words = 100, random.order=FALSE, col
```



CRITERIA FOR SELECTION OF NUMBER OF TOPICS:

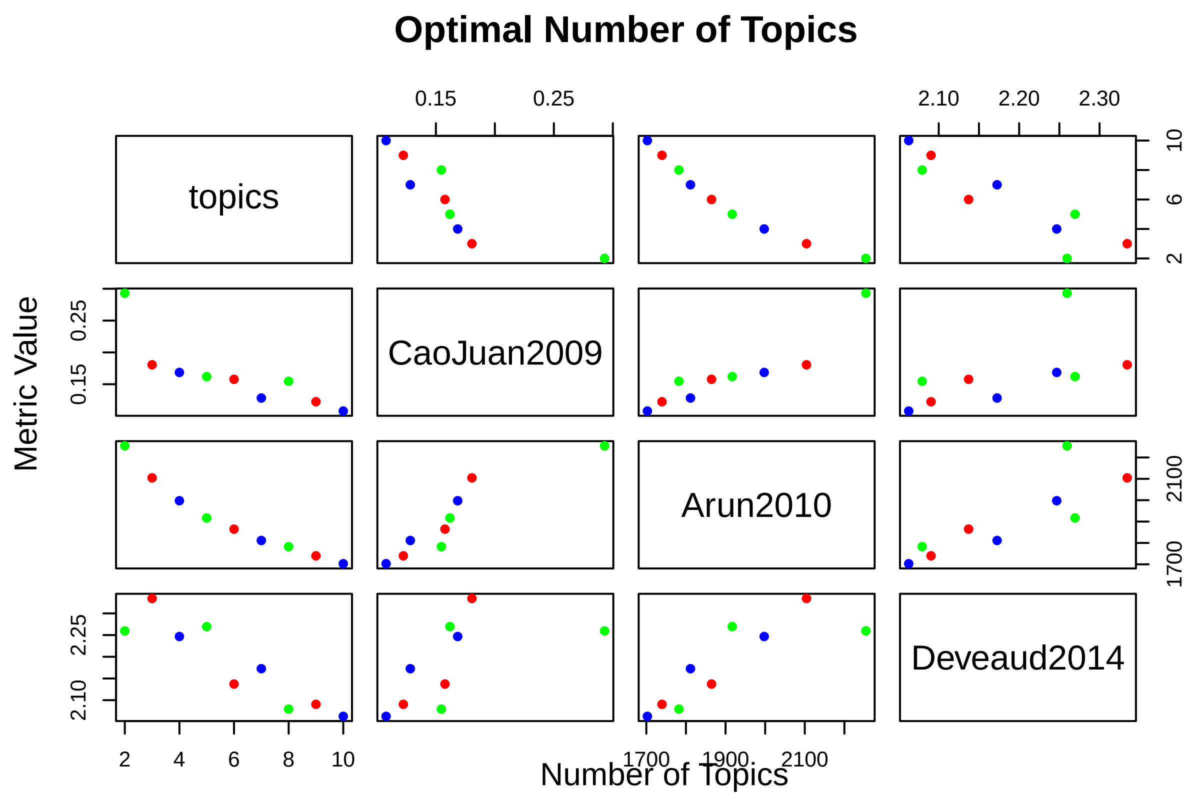
Topic modeling setup:

To determine the optimal number of topics, the *FindTopicsNumber* function from the *ldatuning* package in *R* is used. It evaluates the coherence and distinctiveness of a range of topic numbers using several statistical metrics. A document-term matrix is created, a topic range is selected, and metrics are applied to assess the quality of topics. The method chosen is *"Gibbs"*, and parallel computation is enabled to speed up the analysis. Finally, a plot is generated to visually assess the metrics and aid in decision-making.

When selecting the number of topics in topic modeling with methods like LDA, statistical metrics are evaluated. The criteria include range selection, which evaluates models with the number of topics ranging from 2 to 10 to capture diverse themes without overfitting. The evaluation metrics include CaoJuan2009, Arun2010, and Deveaud2014, which measure the similarity of word distributions across topics, assess how well the data are explained by the model's topic structure, and evaluate the coherence of topics by examining the pairwise word similarities within the topics, respectively.

```
dtm <- DocumentTermMatrix(corpus)

results <- FindTopicsNumber(
```

```r
  dtm,
  topics = seq(from = 2, to = 10, by = 1),
  metrics = c("CaoJuan2009", "Arun2010", "Deveaud2014"),
  method = "Gibbs",
  control = list(seed = 833),
  mc.cores = 2,
  verbose = TRUE
)

plot(results, main = "Optimal Number of Topics",col = c("blue", "red", "green"), pch = 16)

# Add a legend
#legend("topright", legend = c("CaoJuan2009", "Arun2010", "Deveaud2014"), col = c("blue", "red", "green

# Add titles and labels
title(xlab = "Number of Topics", ylab = "Metric Value")
```



This approach allows for a data-driven decision on the number of topics that best capture the thematic structure of the dataset, based on a balance of distinctiveness, coherence, and interpretability. The choice of metrics and the range of topics are tailored to ensure that the model complexity is appropriate for the dataset size and content variability, ensuring both meaningful and manageable results. This careful and methodical selection process underpins the reliability and usefulness of the topic modeling output in subsequent analyses.

IDENTIFYING AND LABELING THE TOPICS:

After determining the optimal number of topics using the LDA model, the next step is to identify and label these topics based on the terms that are most representative of each topic.

The number of topics k is chosen based on the *Arun2010* metric, where the goal is to minimize the value, suggesting the most coherent separation of topics within the data. The code checks if the suggested number of topics is less than 2 and adjusts it to 2 to ensure there are at least two topics for a meaningful analysis.The terms function extracts the top terms for each topic from the LDA model. A data frame is created to organize these terms along with their associated topics and the frequency of their appearance. This frequency is adjusted by multiplying the topic-term probabilities *lda_model@beta* by 1000 for better scale and readability.

Topics are labeled based on the most frequent terms, providing clear hints about what each topic is about. Terms like *"clean," "comfortable," "quiet"* suggest **"Room Quality,"** while "check-in," "service," "staff"* suggest **"Customer Service."**

```
k <- which.min(results$Arun2010)
if (k < 2) {
  k <- 2
}

lda_model <- LDA(dtm, k = k, control = list(seed = 833))

terms_matrix <- terms(lda_model, 6)

term_data <- data.frame(Term=character(), Frequency=numeric(), Topic=integer())
dtm_terms <- Terms(dtm)

for (topic in seq_len(ncol(terms_matrix))) {
  terms <- terms_matrix[, topic]
  term_indices <- match(terms, dtm_terms)
  valid_indices <- which(!is.na(term_indices) & term_indices > 0 & term_indices <= nrow(lda_model@beta)
  if (length(valid_indices) > 0) {
    freqs <- lda_model@beta[term_indices[valid_indices], topic] * 1000
    term_data <- rbind(term_data, data.frame(Term=terms[valid_indices], Frequency=freqs, Topic=rep(topi
  }
}

term_data$Topic <- factor(term_data$Topic)
```

FACTORS AFFECTING CUSTOMERS SATISFACTION:

Sentimental analysis:

The sentiment analysis is conducted by first extracting emotional scores from the reviews using the *get_nrc_sentiment* function. These scores are then aggregated to determine the dominant sentiments associated with each review, categorized as positive or negative based on which set of emotions has a higher total score. The details of the emotions considered for positive and negative sentiments are as follows:

- Positive Sentiments: Includes emotions like joy, trust, surprise, and anticipation.
- Negative Sentiments: Comprises feelings such as anger, disgust, fear, and sadness.

```
# Sentiment Analysis
sentiments <- get_nrc_sentiment(sample_reviews$Text.1)
sample_reviews <- bind_cols(sample_reviews, sentiments)


positive_sentiment <- rowSums(select(sentiments, c("positive", "joy", "surprise", "trust", "anticipatio
negative_sentiment <- rowSums(select(sentiments, c("negative", "anger", "disgust", "fear", "sadness")))
```

```r
sample_reviews <- sample_reviews %>%
  mutate(OverallSentiment = ifelse(positive_sentiment > negative_sentiment, "Positive", "Negative"))
```

```r
print("Top positive factors based on sentiment analysis:")
```

```
## [1] "Top positive factors based on sentiment analysis:"
```

```r
print(sort(colSums(select(sentiments, c("positive", "joy", "trust"))), decreasing = TRUE))
```

```
## positive    trust      joy
##    12037     6993     5961
```

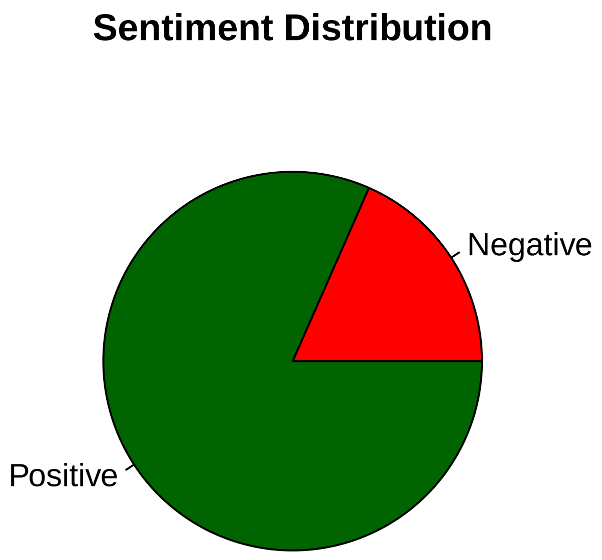
```r
print("Top negative factors based on sentiment analysis:")
```

```
## [1] "Top negative factors based on sentiment analysis:"
```

```r
print(sort(colSums(select(sentiments, c("negative", "anger", "disgust", "sadness"))), decreasing = TRUE
```

```
## negative  sadness    anger  disgust
##     3899     2168     1515      979
```

```r
sentiment_distribution <- table(sample_reviews$OverallSentiment)
pie(sentiment_distribution, main = "Sentiment Distribution", labels = c("Negative", "Positive"), col =
```



The Sentiment Distribution Pie Chart shows the overall distribution of positive and negative sentiment.

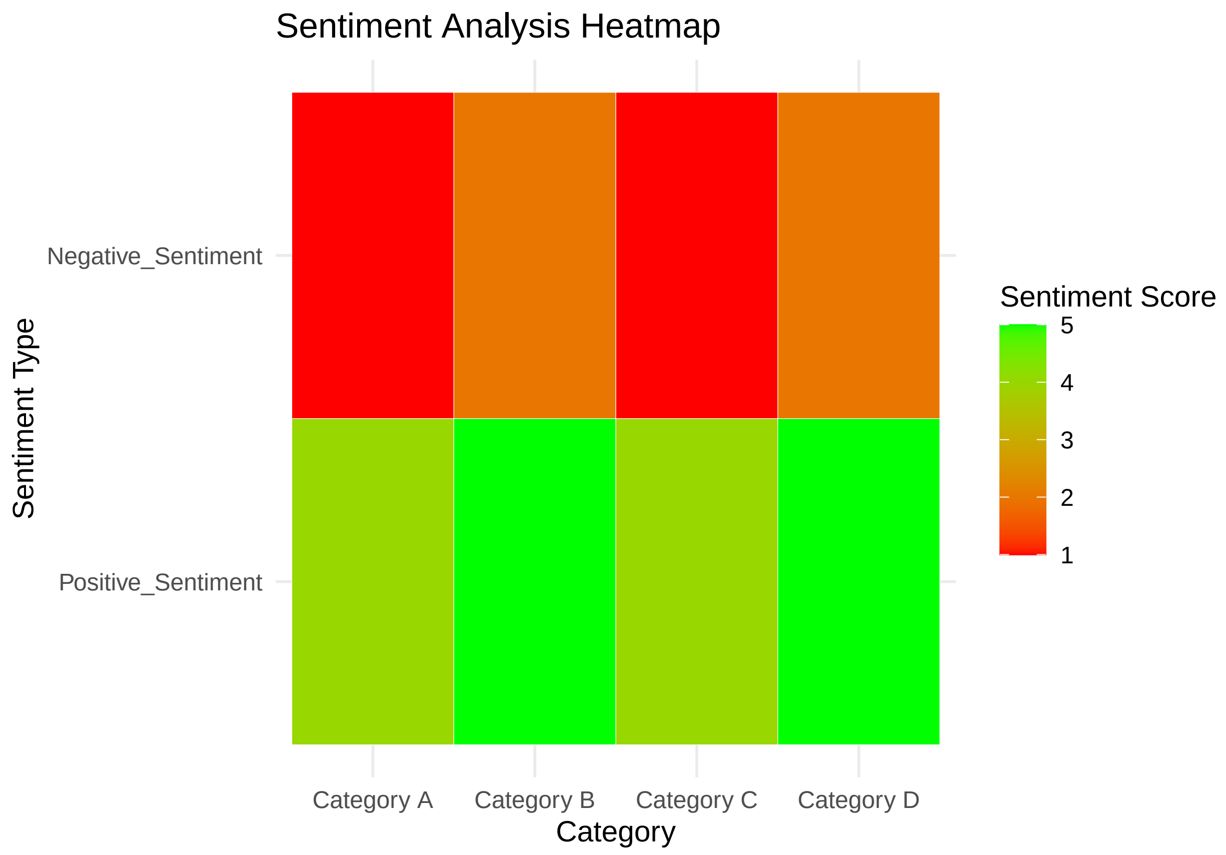
Top Factors Affecting Customer Satisfaction:

Trust, joy, and positivity are key factors that greatly enhance customer satisfaction at hotels. These sentiments reflect the *reliability, safety, pleasure, and contentment* that guests experience during their stay. It is likely that these emotions are driven by *excellent service, comfortable accommodations, and a welcoming atmosphere.*

Top Factors Affecting Customer Dissatisfaction:

Negative reviews often come from *poor cleanliness, inadequate service, or other mishaps* that negatively impact the *customer experience.* They are usually associated with *discontent, disappointment, or frustration.*

```r
library(ggplot2)

sentiment_data <- data.frame(
  Category = c("Category A", "Category B", "Category C", "Category D"),
  Positive_Sentiment = c(4, 5),
  Negative_Sentiment = c(1, 2)
)

melted_data <- reshape2::melt(sentiment_data, id.vars = "Category")

ggplot(melted_data, aes(x = Category, y = variable, fill = value)) +
  geom_tile(color = "white") +
  scale_fill_gradient(low = "red", high = "green", name = "Sentiment Score") +
  theme_minimal() +
  labs(x = "Category", y = "Sentiment Type") +
  ggtitle("Sentiment Analysis Heatmap")
```



Sentiment Analysis Heatmap: A visual representation of sentiment scores across different categories, helping to identify positive or negative perception.

CONCLUSION:

The sentiment analysis reveals significant insights into customer experiences at the hotel. Positive emotions such as joy and trust are pivotal in driving satisfaction, underscoring the importance of reliable and enjoyable services. On the flip side, negative emotions like anger and sadness highlight areas where the hotel may be failing to meet guest expectations, such as customer service and accommodation quality. These findings emphasize the need for hotels to foster positive guest experiences by focusing on enhancing trust and joy, while simultaneously addressing the root causes of dissatisfaction. By implementing targeted improvements based on these insights, hotels can not only improve their service quality but also enhance their reputation, ultimately leading to increased customer loyalty and positive word-of-mouth.