Project 1

S03 Var05, S03 Var07, S05 Var03

Group 3: Shaya Engelman, Julia Ferris, Amanda Fox, Jean Jimenez

2024-06-23

# Introduction

This project is an attempt to forecast various variables in a dataset. The dataset is completely deidentified and contains 6 ‘categories’ with 5 time series for each category. The dataset consists of of 1622 observations and another 140 empty observations to be forecasted.

Here, the focus is on forecasting the variables S03 Var05, S03 Var07, and S05 Var03.

# S03

## Data

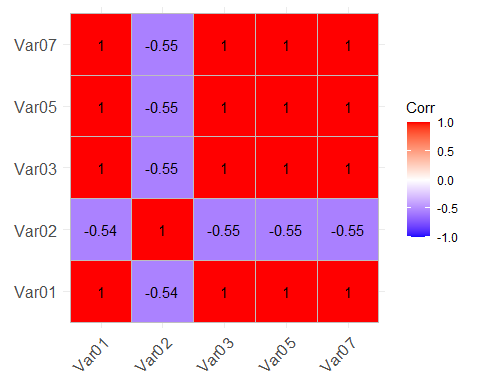
The data is loaded and the category of interest is extracted. The following table is the first 6 rows of the data. The SeriesInd column is the index of the time series, the other columns are the various time series. As mentioned, the data, including the index is completely deidentified, so the actual meaning of both the variables and what the time step being measured is unknown.

First 6 rows of the S03 data

| SeriesInd | Var01 | Var02 | Var03 | Var05 | Var07 |
| --- | --- | --- | --- | --- | --- |
| 40669 | 30.64286 | 123432400 | 30.34000 | 30.49000 | 30.57286 |
| 40670 | 30.79857 | 150476200 | 30.46428 | 30.65714 | 30.62571 |
| 40671 | 30.74714 | 138040000 | 30.10714 | 30.62571 | 30.13857 |
| 40672 | 30.28572 | 119282800 | 29.86429 | 30.25000 | 30.08286 |
| 40673 | 30.28572 | 111902700 | 29.86572 | 30.04286 | 30.28286 |
| 40676 | 30.42857 | 115557400 | 29.77857 | 30.40000 | 30.01571 |

While only two variables, Var05 and Var07, are of interest, is is important to take a glance at the other variables in the dataset. By visualizing the time series and the correlation between the variables, it becomes apparent that many of these variables are not just highly correlated, but also have almost identical patterns. In particular, the two variables of interest, Var05 and Var07, seem to be almost identical. Only Var02 seems to be onan entirely different order of magnitude with much less correlation to the other variables.

A graph of different colored lines

Description automatically generated

The summary statistics reveal some interesting information. The two variables of interest, Var05 and Var07, have almost identical values for the mean, median, and standard deviation. The minimum and maximum values are also very similar.

Since the data is completely deidentified, it is not known what the ‘SeriesInd’ column represents. The actual values appear to be Excel date serials (the number of days since December 30, 1899), but since this is not confirmed, it will be left as a blank index unless there is a reason to change it. An interesting insight from the summary statistics

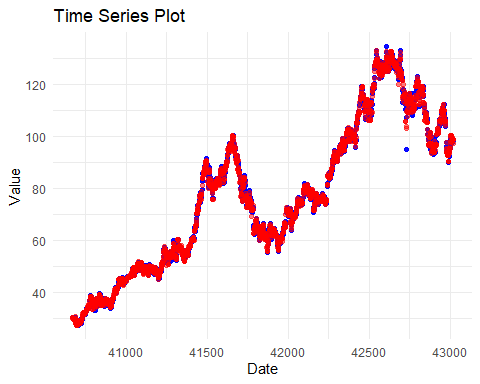
There are 144 missing values for both ‘Var05’ and ‘Var07’. The majority of the missing values are in the last 140 rows of the dataset, which are the rows to be forecasted. There are 4 rows of missing data in the first 1482 rows. By checking those rows, it is observed that both ‘Var05’ and ‘Var07’ are missing for the same rows. This does not allow us to use the other variable to fill in the missing values. Since there are only 4 rows out of 1622 missing, the method of imputation should not have a significant impact on the forecast. The method chosen was to use an interpolation method to fill in the missing values with the average of the two closest non-missing values.

Summary Statistics for Variables ‘Var05’ and ‘Var07’

| Variable | Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. | NA’s |
| --- | --- | --- | --- | --- | --- | --- | --- |
| SeriesInd | 40669.00 | 41304.00 | 41946.00 | 41945.00 | 42586.00 | 43221.00 | NA |
| Var05 | 27.48 | 53.30 | 75.59 | 76.90 | 98.55 | 134.46 | 144 |
| Var07 | 27.44 | 53.46 | 75.71 | 76.87 | 98.61 | 133.00 | 144 |

Additionally, to fill in the implicitly missing rows to make the data continuous, a different imputation method was used. For this, the rows were filled with the most recent non-missing value. This method was chosen because the data appears to have zero change for those observations based on the reasoning why they might be missing, such as only operaing on business days.

The individual time series for ‘Var05’ and ‘Var07’ are plotted below and it is noted that there does not seem to be any major outliers that need to be addressed. The data does display an overall upward trend with significant variability. Initially, the data seems to have very minor fluctuationsm, but later on, the variability increases significantly. Additionally, the data seems to have an either seasonal or cyclical component, with the data appearing to have a pattern of peaks and valleys.

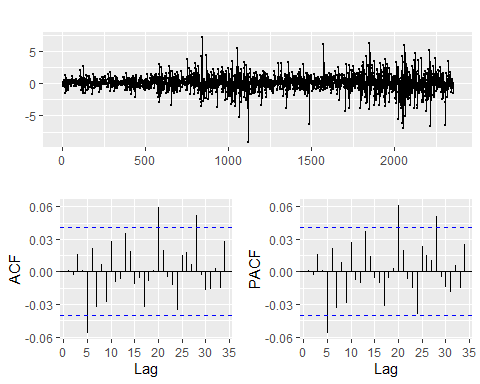
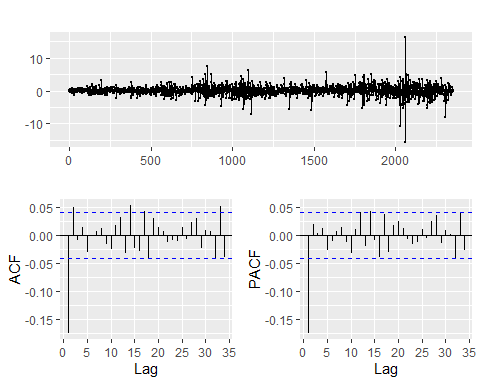


## Data Preparation

Checking for seasonality revealed a frequency of 1, which indicates that there is no seasonality in the data and that the data does not have any cyclical patterns. However, the data clearly was non-stationary, meaning, the value of any given observation is dependent on the value of the previous observation. This was confirmed by the Augmented Dickey-Fuller test, which showed an extremely high p-value, indicating that the data is non-stationary. To make the data stationary, differencing was used. The first difference was taken, meaning that the value of each observation is the difference between the value of that observation and the value of the previous observation. The Augmented Dickey-Fuller test was run again and the p-value was now less than 0.05, indicating that the data is now stationary.

To ensure that the data is stationary, the autocorrelation function (ACF) and partial autocorrelation function (PACF) were plotted. The ACF shows the correlation between the current observation and the previous observation at various lags. The PACF shows the correlation between the current observation and the previous observation at various lags, but it removes the effect of the intermediate observations. The dotted blue lines indicate the 95% confidence interval, and if the bars extend beyond the confidence interval, it indicates that the correlation is statistically significant. The ACF and PACF plots show that the data is stationary, as the bars do not extend beyond the confidence interval. The plot also shows the overall pattern of the data, which is now a flat line with no significant correlation between the observations.

It was observed that although until now there was no significant differences between the two variables, the ACF and PACF plots show that the two variables have different patterns. To account for this difference, the two variables were modeled and forecasted separately.



## Modeling

### S03 Var05

The auto.arima function was used to determine the best ARIMA model for the data. The auto.arima function uses the Akaike Information Criterion (AIC) to determine the best model. The AIC is a measure of the quality of a model, with lower values indicating a better model. The auto.arima function determined that the best model for the data was an ARIMA (1, 0, 0) model. Since the data was differenced once, the model for the original data is an ARIMA (1, 1, 0) model.

An ETS model was also fit to the differenced data. The ETS model is a time series model that accounts for error, trend, and seasonality. The ETS model determined that the best model for the data was an ETS (A, N, N) model. The ETS model is a simple model that does not account for seasonality or trend, which is why the model is ETS (A, N, N). The lack of seasonality is due to the data being differenced, which removed the seasonality.

Finally, another ETS model was fit to the original data. The ETS model determined that the best model for the data was an ETS (M, A, N) model. The ETS model is a simple model that accounts for seasonality but not trend, which is why the model is ETS (M, A, N). The model accounts for seasonality because the original data had a seasonal component, which was removed by differencing.

The coefficients table summarizes the results from fitting different time series models to the data. The ARIMA model, specifically ARIMA (1, 1, 0), reveals a significant negative impact from the first lag (ar1 = -0.1738014), indicating a strong dependence on the previous period’s value. In contrast, the ETS (Diff) model, characterized by parameters like alpha = 0.0001000 and l = 0.0298683, focuses on adjusting for level and trend in the differenced data. Meanwhile, the ETS model applied directly to the original series suggests a higher level dependence (alpha1 = 0.8605962) and incorporates seasonality (l1 = 31.1662894). These coefficients provide insights into how each model type captures temporal dependencies and adjusts for underlying patterns, crucial for accurate forecasting in time series analysis.

Model Coefficients for S03 Var05

|  | Coefficient | Model |
| --- | --- | --- |
| ar1 | -0.1738014 | ARIMA |
| alpha | 0.0001000 | ETS (Diff) |
| l | 0.0298683 | ETS (Diff) |
| alpha1 | 0.8605962 | ETS |
| beta | 0.0001004 | ETS |
| l1 | 31.1662894 | ETS |
| b | 0.0476719 | ETS |

Comparing the models using the AIC, BIC and log-likelihood lead to the decision to use the ETS model for forecasting. While the ARIMA model has a lower AIC and BIC and higher log-likelihood, the difference was close enough to the ETS model that the fact that the ETS model captures some of the trend made it a better choice. The ARIMA model would predict the same value for every forecast, while the ETS model would predict a value that is slightly different for each forecast. If the goal of this would be to predict the singular next value, the ARIMA model would be the better choice. However, since the goal is to predict the next 140 values, the ETS model is the better choice.

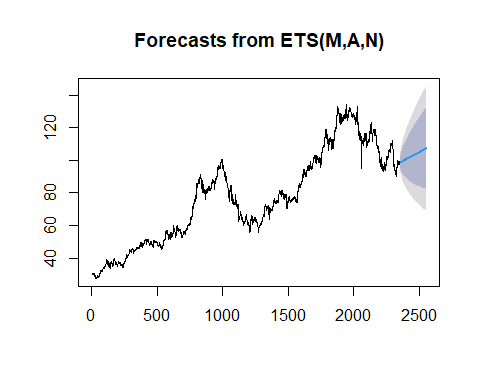
Model Comparison for S03 Var05

| Model | AIC | BIC | Log\_Likelihood |
| --- | --- | --- | --- |
| ARIMA | 7695.798 | 7707.324 | -3845.899 |
| ETS(A,N,N) | 19352.894 | 19370.183 | -9673.447 |
| ETS(M,A,N) | 18684.968 | 18713.785 | -9337.484 |

### Forecasting

After selecting the ETS model, the residuals were checked to ensure that the model was a good fit for the data. The residuals are the difference between the actual value and the predicted value. The residuals should be random and have a mean of zero. The residuals for the ETS model were plotted and the histogram shows that the residuals are normally distributed. The ACF and PACF plots show that the residuals are random and do not have any significant correlation. The Ljung-Box test was run to confirm that the residuals are random. The p-value was greater than 0.05, indicating that the residuals are random.

Finally, the ETS model was used to forecast the next 200 values. Although the task was to forecast the next 140 values, the model was used to forecast 200 values to account for the rows that will need to be removed to return the data to its original state. The forecasted values were then added to the original data and the extra rows removed to return the data to the original state. The new combined dataset was then saved to a new Excel file.



### S03 Var07

A similar process was followed for ‘Var07’. The auto.arima function was used to determine the best ARIMA model for the data. The auto.arima function determined that the best model for the differenced data was an ARIMA (0, 0, 0) model. After accounting for the differencing, the model for the original data is an ARIMA (0, 1, 0) model. This model is a simple model that only accounts for the error term, making it a not very useful model (but perhaps the best for this data).

Another ARIMA model was created using the parameters from Var05 since the data had looked so similar. The model was an ARIMA (1, 1, 0) model.

A third ARIMA model was created using the same order as the original data, an ARIMA (0, 1, 0) model, but with a drift term added. The drift term is a constant that is added to the model to account for a trend in the data. The drift term was added because the data appeared to have a trend.

Finally, an ETS model was fit to the data. The ETS model determined that the best model for the data was an ETS (M, A, N) model. The ETS model is a simple model that accounts for seasonality but not trend, which is why the model is ETS (M, A, N).

The coefficients table summarizes the results from fitting different time series models, specifically ARIMA and ETS, to the data series S03 Var05. The ARIMA (1, 1, 0) model indicates a very strong positive autoregressive effect (ar1 = 0.9993729), suggesting that each period’s value is nearly identical to the previous period’s adjusted value after differencing. The intercept (intercept = 76.7231540) and drift (drift = 0.0283874) components further refine this model by incorporating a constant term and a linear trend, respectively. In contrast, the ETS model, characterized by parameters such as alpha = 0.9998964, beta = 0.0001380, l = 30.8774992, and b = 0.0467874, emphasizes smoothing, trend, and damping effects. These coefficients highlight how each model type captures and adjusts for different temporal patterns, essential for accurate forecasting and understanding the underlying dynamics of S03 Var05.

Model Coefficients for S03 Var05

|  | Coefficient | Model |
| --- | --- | --- |
| ar1 | 0.9993729 | ARIMA (1, 1, 0) |
| intercept | 76.7231540 | ARIMA (1, 1, 0) |
| drift | 0.0283874 | ARIMA (0, 1, 0) with Drift |
| alpha | 0.9998964 | ETS |
| beta | 0.0001380 | ETS |
| l | 30.8774992 | ETS |
| b | 0.0467874 | ETS |

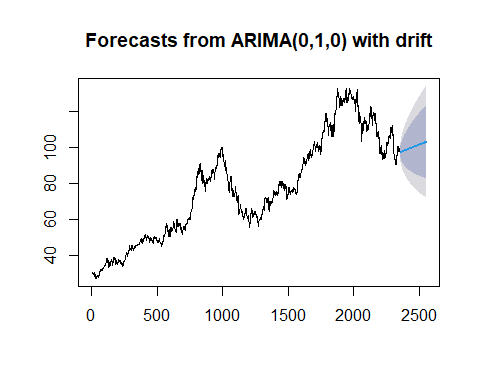
Comparing the models using the AIC, BIC and log-likelihood lead to the decision to use the ARIMA model with a drift term for forecasting. The ARIMA model with a drift term had slightly worse AIC and BIC values than the Auto ARIMA model, but would likely be a better model for forecasting since it accounts for the trend in the data. Interestingly, the manually created ARIMA model was almost as good as the other ARIMA models, reinforcing its similarity to the Var05 data.

## Model AIC BIC Log\_Likelihood  
## 1 AUTO.ARIMA 7189.837 7195.600 -3593.919  
## 2 ARIMA(1,0,0) 7202.973 7220.263 -3598.486  
## 3 ARIMA(0,1,0) with drift 7190.313 7201.839 -3593.156  
## 4 ETS 18265.686 18294.503 -9127.843

### Forecasting

After selecting the ARIMA model with a drift term, the residuals were checked to ensure that the model was a good fit for the data. The residuals are the difference between the actual value and the predicted value. The residuals should be random and have a mean of zero. The residuals for the ARIMA model with a drift term were plotted and the histogram shows that the residuals are normally distributed. The ACF and PACF plots show that the residuals are random and do not have any significant correlation. The Ljung-Box test was run to confirm that the residuals are random. The p-value was greater than 0.05, indicating that the residuals are random.

Same as with Var05, the ARIMA model with a drift term was used to forecast the next 200 values. The forecasted values were then added to the original data and the extra rows removed to return the data to the original state. The new combined dataset was then saved to a new Excel file.



# S05

## Data

A similar process was followed for ‘S05 Var03’. The data was loaded and the category of interest was extracted. The first 6 rows of the data were displayed. The SeriesInd column is the index of the time series, the other columns are the various time series. As mentioned, the data, including the index is completely deidentified, so the actual meaning, both the variables and what the time step being measured is unknown.

First 6 rows of the S03 data

| SeriesInd | Var01 | Var02 | Var03 | Var05 | Var07 |
| --- | --- | --- | --- | --- | --- |
| 40669 | 30.64286 | 123432400 | 30.34000 | 30.49000 | 30.57286 |
| 40670 | 30.79857 | 150476200 | 30.46428 | 30.65714 | 30.62571 |
| 40671 | 30.74714 | 138040000 | 30.10714 | 30.62571 | 30.13857 |
| 40672 | 30.28572 | 119282800 | 29.86429 | 30.25000 | 30.08286 |
| 40673 | 30.28572 | 111902700 | 29.86572 | 30.04286 | 30.28286 |
| 40676 | 30.42857 | 115557400 | 29.77857 | 30.40000 | 30.01571 |

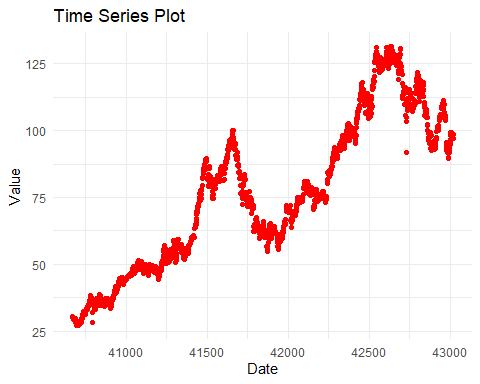
As with the previous data, checking the summary statistics revealed some interesting information. Again it is noted that the SeriesInd column appears to be Excel date serials, but since this is not confirmed, it will be left as a blank index unless there is a reason to change it. The summary statistics show that there are 145 missing values for ‘Var03’. The majority of the missing values are in the last 140 rows of the dataset, which are the rows to be forecasted. There are 5 rows of missing data in the first 1622 rows. As with the previous data, the method of imputation should not have a significant impact on the forecast. The method chosen was to use an interpolation method to fill in the missing values with the average of the two closest non-missing values.

As with the previous data, the gaps were filled in with the most recent non-missing value. This method was chosen because the data appears to have zero change for those observations based on the reasoning why they might be missing, such as only operating on business days.

Summary Statistics for Variable ‘Var03’

| Variable | Min. | 1st Qu. | Median | Mean | 3rd Qu. | Max. | NA’s |
| --- | --- | --- | --- | --- | --- | --- | --- |
| SeriesInd | 40669.00 | 41304.0 | 41946.00 | 41945.00 | 42586.00 | 43221.0 | NA |
| Var03 | 27.18 | 52.7 | 74.85 | 76.11 | 97.65 | 131.4 | 144 |

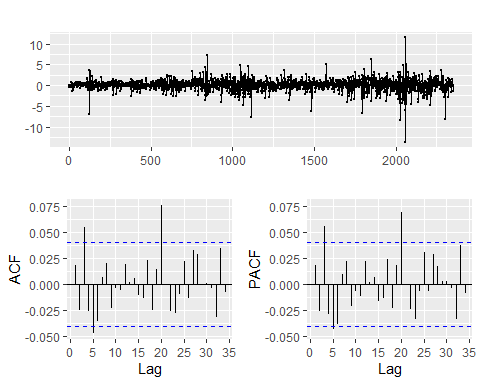
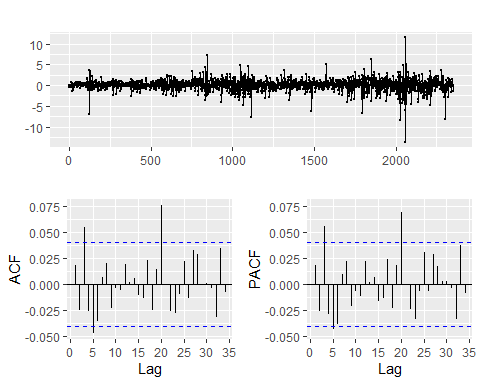
The individual time series for ‘Var03’ is plotted below and there does not appear to be any major outliers, however, there are some minor outliers at just under index 52750. Additionally, the data seems to have an overall upwards trend but has a lot of noise and that might just be an illusion. If there is no overall trend, then those points mentioned earlier are not outliers. They are well within the range of the overall data, merely far from the majority of the data points at that point in time.



## Data Preparation

Checking for seasonality revealed a frequency of 1, which indicates that there is no seasonality in the data and that the data does not have any cyclical patterns. However, the data clearly was non-stationary, meaning, the value of any given observation is dependent on the value of the previous observation. This was confirmed by the Augmented Dickey-Fuller test, which showed an extremely high p-value, indicating that the data is non-stationary. To make the data stationary, differencing was used. The first difference was taken, meaning that the value of each observation is the difference between the value of that observation and the value of the previous observation. The Augmented Dickey-Fuller test was run again and the p-value was now less than 0.05, indicating that the data is now stationary.

To ensure that the data is stationary, the autocorrelation function (ACF) and partial autocorrelation function (PACF) were plotted for this as well. The ACF and PACF plots show that the data is stationary, as the bars do not extend beyond the confidence interval. The plot also shows the overall pattern of the data, which is now a flat line with no significant correlation between the observations.



## Modeling

### S05 Var03

The auto.arima() function was used to determine the best ARIMA model for the data. The auto.arima() function uses the Akaike Information Criterion (AIC) to determine the best model. The AIC is a measure of the quality of a model, with lower values indicating a better model. The auto.arima() function determined that the best model for the data was an ARIMA (0, 0, 1) model. Since the data was differenced once, the model for the original data is an ARIMA (0, 1, 1) model. This model is a simple model that only accounts for the error term and the moving average term, but no auto-regressive term.

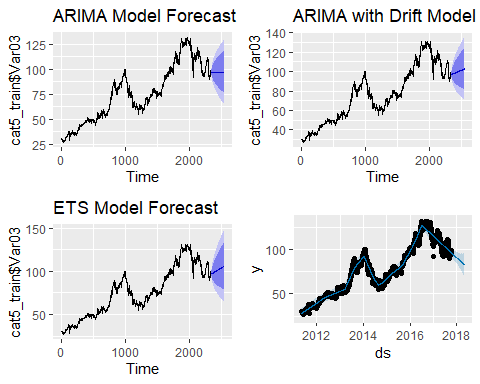
An ARIMA model was also created with the same order as the previous model but with a drift term added. The drift term is a constant that is added to the model to account for a trend in the data. The drift term was added because the data appeared to have a trend.

Additionally, an ETS model was fit to the data. The ETS model determined that the best model for the data was an ETS (A, N, N) model. The ETS model is a simple model that does not account for seasonality or trend, which is why the model is ETS (A, N, N).

The final model applied to the data is the Prophet model, developed by Facebook for handling time series data with inherent seasonality and holiday effects. Unlike previous models, Prophet introduces assumptions about the data’s nature, specifically its daily frequency with missing values on weekends. Up to this point, the data was anonymized to prevent assumptions. However, Prophet requires assuming this daily pattern. To fit the Prophet model, I will utilize the prophet() function from the prophet package, which automatically selects the optimal model configuration based on the Akaike Information Criterion (AIC).

The residuals of all four models were checked to ensure that the models were a good fit for the data. All the models had normally distributed residuals, ACF and PACF plots that showed no autocorrelation and all seemes to fit the well. The Prophet model had normally distributed residuals but the ACF plot did seem to show some autocorrelation.

Predictions were generated for the next 200 periods for all four models and plotted to visualize the forecasted values and to see which seems to be the best fit for the data.



The ARIMA model, despite its wide confidence intervals due to high data variability, provides forecasts that are merely the means of the data, making it less useful especially with the lack of trend in the data. Its variant, the ARIMA model with drift, offers similar results but with slightly varied successive forecasts, potentially marking an improvement. The ETS model, while still having wide confidence intervals, is slightly better than the ARIMA models, but the extreme data variability results in identical successive forecasts. The Prophet model stands out with its significantly different forecasts, smaller confidence intervals, and a clear negative trend, suggesting it could be the best model if the forecasts are accurate. Despite the ARIMA model with drift’s risk of overfitting and wide confidence interval, and the ETS model’s identical forecasted values due to noise and lack of trend or seasonality, the Prophet model appears to be the best fit for this data. However, my hesitation stems from its requirement of data assumptions, unclean residuals, and significantly different forecasts. My lack of experience with the Prophet model also raises concerns about potential implementation errors.

Despite these concerns, the Prophet model was selected for the final forecast due to its meaningful forecasts, with plans to revisit this decision once the actual forecast data is available to determine if the Prophet model is indeed the best or the most inaccurate. The forecasts were generated for the next 200 periods, which were then added to the original data. The extra rows were removed to return the data to its original state. The new combined dataset was then saved to a new Excel file.

# Appendix: Code

library(tidyverse)  
library(readxl)  
library(here)  
library(tsibble)  
library(fable)  
library(ggcorrplot)  
library(forecast)  
library(tseries)  
library(zoo)  
library(writexl)  
library(prophet)  
library(knitr)  
library(kableExtra)  
library(gridExtra)  
library(ggplotify)  
complete\_data <- read\_excel(here("Project\_1", "Prompt", "Data Set for Class.xls"))  
  
# Split the data into 6 different data frames by the column "category"  
data\_list <- split(complete\_data, complete\_data$category)  
  
# Extract category three  
category\_three <- data\_list$`S03` |>  
 select(-category) # drop the category column  
  
summary(category\_three)  
kable(head(category\_three), caption = "First 6 rows of the S03 data")  
# Plot the five variables as faceted plots to allow for different scales on the y-axis  
category\_three |>  
 pivot\_longer(cols = -SeriesInd, names\_to = "variable", values\_to = "value") |>  
 ggplot(aes(x = SeriesInd, y = value, color = variable)) +  
 geom\_line() +  
 labs(title = "Time Series Plot", x = "Date", y = "Value", color = "Variable") +  
 theme\_minimal() +  
 facet\_wrap(~ variable, scales = "free\_y", ncol = 1)  
# Compute correlation matrix  
category\_three |>  
 select(Var01, Var02, Var03, Var05, Var07) |>  
 na.omit() |> # cor() does not handle missing values  
 cor() |>  
 ggcorrplot(lab = TRUE)  
cat3\_var5\_var7 <- category\_three |>  
 select(SeriesInd, Var05, Var07)  
  
# Convert to tsibble  
cat3\_var5\_var7 <- as\_tsibble(cat3\_var5\_var7, index = SeriesInd)   
  
# Check the tsibble object  
#length(cat3\_var5\_var7$SeriesInd)  
  
# Generate summary statistics  
summary\_stats <- summary(cat3\_var5\_var7)  
  
# Convert the summary table to a data frame  
summary\_df <- as.data.frame(as.table(summary\_stats))  
  
# Clean up the data frame  
summary\_df <- summary\_df %>%  
 rename(Statistic = Var1, Variable = Var2, Value = Freq) %>%  
 mutate(Value = trimws(Value))  
  
# Split the Value column into separate statistic and value columns  
summary\_df <- summary\_df %>%  
 separate(Value, into = c("Statistic", "Value"), sep = ":", convert = TRUE) %>%  
 mutate(Statistic = trimws(Statistic), Value = as.numeric(trimws(Value)))  
  
# Remove the extra 'NA' column, if present  
summary\_df <- summary\_df %>%  
 filter(!Statistic %in% NA)  
  
# Pivot the summary data frame wider  
summary\_wide <- summary\_df %>%  
 pivot\_wider(names\_from = Statistic, values\_from = Value)  
  
# Print the summary table  
kable(summary\_wide, caption = "Summary Statistics for Variables 'Var05' and 'Var07'")  
# Remove last 140 rows  
n <- nrow(cat3\_var5\_var7)  
cat3\_train <- cat3\_var5\_var7[-c((n-139):n), ]  
  
# Print the rows of missing values  
cat3\_train |>  
 filter(is.na(Var05) | is.na(Var07))  
  
# Impute missing values  
cat3\_train$Var05 <- na.approx(cat3\_train$Var05)  
cat3\_train$Var07 <- na.approx(cat3\_train$Var07)  
  
# Check for missing values to make sure they were filled in  
cat3\_train |>  
 filter(is.na(Var05) | is.na(Var07))  
# Add the implicitly missing rows  
cat3\_train <- cat3\_train |>   
 fill\_gaps()  
  
# Fill NAs with the last value  
cat3\_train$Var05 <- na.locf(cat3\_train$Var05)  
cat3\_train$Var07 <- na.locf(cat3\_train$Var07)  
  
# Check for missing values  
cat3\_train |>  
 filter(is.na(Var05) | is.na(Var07))  
# Save as separate time series objects  
var5 <- ts(cat3\_train$Var05)  
var7 <- ts(cat3\_train$Var07)  
  
# Plot the two variables  
ggplot() +  
 geom\_point(aes(x = cat3\_train$SeriesInd, y = var5), color = "blue", alpha = 0.95) +  
 geom\_point(aes(x = cat3\_train$SeriesInd, y = var7), color = "red", alpha = 0.5) +  
 labs(title = "Time Series Plot", x = "Date", y = "Value", color = "Variable") +  
 theme\_minimal()  
findfrequency(var5)  
findfrequency(var7)  
  
adf.test(var5)  
adf.test(var7)  
  
ndiffs(var5)  
ndiffs(var7)  
  
# Take the first difference  
diff\_var5 <- diff(var5)  
diff\_var7 <- diff(var7)  
  
# Plot the differenced data  
ggplot() +  
 geom\_line(aes(x = cat3\_train$SeriesInd[-1], y = diff\_var5), color = "blue", alpha = 0.5) +  
 geom\_line(aes(x = cat3\_train$SeriesInd[-1], y = diff\_var7), color = "red", alpha = 0.5) +  
 labs(title = "Time Series Plot", x = "Date", y = "Value", color = "Variable") +  
 theme\_minimal()  
  
adf.test(diff\_var5)  
adf.test(diff\_var7)  
# ACF and PACF plots  
ggtsdisplay(diff\_var5)  
ggtsdisplay(diff\_var7)  
# Fit the ARIMA model  
fit\_var5 <- auto.arima(diff\_var5)  
  
summary(fit\_var5)  
# Fit ARIMA model  
arima\_var5 <- auto.arima(var5)  
summary\_arima <- summary(arima\_var5)  
# Fit ETS model  
ets\_var5\_diff <- ets(diff\_var5)  
  
ets\_diff\_summary <- summary(ets\_var5\_diff)  
ets\_var5 <- ets(var5)  
  
ets\_summary <- summary(ets\_var5)  
# Extract coefficients  
arima\_var5\_coef <- arima\_var5$coef  
ets\_var5\_diff\_params <- ets\_var5\_diff$par  
ets\_var5\_params <- ets\_var5$par  
  
# Convert to data frames  
arima\_var5\_coef\_df <- as.data.frame(arima\_var5\_coef)  
ets\_var5\_diff\_params\_df <- as.data.frame(ets\_var5\_diff\_params)  
ets\_var5\_params\_df <- as.data.frame(ets\_var5\_params)  
  
# Rename columns to "Coefficient" (assuming they are single-column data frames)  
names(arima\_var5\_coef\_df) <- "Coefficient"  
names(ets\_var5\_diff\_params\_df) <- "Coefficient"  
names(ets\_var5\_params\_df) <- "Coefficient"  
  
# Add model names  
arima\_var5\_coef\_df <- arima\_var5\_coef\_df %>%  
 mutate(Model = "ARIMA")  
  
ets\_var5\_diff\_params\_df <- ets\_var5\_diff\_params\_df %>%  
 mutate(Model = "ETS (Diff)")  
  
ets\_var5\_params\_df <- ets\_var5\_params\_df %>%  
 mutate(Model = "ETS")  
  
# Combine data frames  
coef\_df <- rbind(arima\_var5\_coef\_df, ets\_var5\_diff\_params\_df, ets\_var5\_params\_df)  
  
# Print table  
library(knitr)  
kable(coef\_df, caption = "Model Coefficients for S03 Var05")  
# Create a data frame to compare the models  
model\_comparison <- data.frame(  
 Model = c("ARIMA", "ETS(A,N,N)", "ETS(M,A,N)"),  
 AIC = c(arima\_var5$aic, ets\_var5\_diff$aic, ets\_var5$aic),  
 BIC = c(arima\_var5$bic, ets\_var5\_diff$bic, ets\_var5$bic),  
 Log\_Likelihood = c(arima\_var5$loglik, ets\_var5\_diff$loglik, ets\_var5$loglik)  
)  
  
# Print the table  
kable(model\_comparison, caption = "Model Comparison for S03 Var05")  
# Residuals for ETS  
checkresiduals(ets\_var5)  
summary(ets\_var5)  
# Generate predictions for the next 200 periods  
ets\_var5\_forecast <- forecast(ets\_var5, h = 200)  
ets\_var5\_forecast$mean  
plot(ets\_var5\_forecast)  
ets\_var5\_forecast$mean <- ts(ets\_var5\_forecast$mean, start = start(var5), frequency = frequency(var5))  
  
# Now you can add them together  
var5\_complete <- c(var5, ets\_var5\_forecast$mean)  
length(var5)  
length(ets\_var5\_forecast$mean)  
length(var5\_complete)  
# Convert the forecasts to a dataframe to bind with the category\_three dataframe  
var5\_complete\_df <- data.frame(  
 SeriesInd = time(var5\_complete, origin = "1899-12-30"),  
 Var05 = as.numeric(var5\_complete)  
)  
  
var5\_complete\_df$SeriesInd <- as.numeric(var5\_complete\_df$SeriesInd) + 40668  
  
# Replace Var05 in category\_three with Var05 from var5\_complete  
category\_three\_test <- category\_three |>  
 left\_join(var5\_complete\_df, by = "SeriesInd") |>  
 select(-Var05.x) |>  
 rename(Var05 = Var05.y)  
  
# Check the updated category\_three  
tail(category\_three\_test)  
# Fit ARIMA model  
arima\_var7\_diff <- auto.arima(diff\_var7)  
summary\_arima <- summary(arima\_var7\_diff)  
arima\_var7 <- auto.arima(var7)  
summary\_arima <- summary(arima\_var7)  
# Fit ARIMA model  
arima\_var7\_man <- Arima(var7, order = c(1,0,0))  
summary\_arima\_man <- summary(arima\_var7\_man)  
# Fit ARIMA model  
arima\_var7\_drift <- Arima(var7, order = c(0,1,0), include.drift = T)  
summary\_arima\_drift <- summary(arima\_var7\_drift)  
# Fit ETS model  
ets\_var7 <- ets(var7)  
summary\_ets <- summary(ets\_var7)  
# Extract coefficients  
arima\_var7\_coef <- arima\_var7$coef  
arima\_var7\_man\_coef <- arima\_var7\_man$coef  
arima\_var7\_drift\_coef <- arima\_var7\_drift$coef  
ets\_var7\_params <- ets\_var7$par  
  
# Convert to data frames  
arima\_var7\_coef\_df <- as.data.frame(arima\_var7\_coef)  
arima\_var7\_man\_coef\_df <- as.data.frame(arima\_var7\_man\_coef)  
arima\_var7\_drift\_coef\_df <- as.data.frame(arima\_var7\_drift\_coef)  
ets\_var7\_params\_df <- as.data.frame(ets\_var7\_params)  
  
# Rename columns to "Coefficient" (assuming they are single-column data frames)  
names(arima\_var7\_coef\_df) <- "Coefficient"  
names(arima\_var7\_man\_coef\_df) <- "Coefficient"  
names(arima\_var7\_drift\_coef\_df) <- "Coefficient"  
names(ets\_var7\_params\_df) <- "Coefficient"  
  
# Add model names  
arima\_var7\_coef\_df <- arima\_var7\_coef\_df %>%  
 mutate(Model = "AUTO ARIMA")  
  
arima\_var7\_man\_coef\_df <- arima\_var7\_man\_coef\_df %>%  
 mutate(Model = "ARIMA (1, 1, 0)")  
  
arima\_var7\_drift\_coef\_df <- arima\_var7\_drift\_coef\_df %>%  
 mutate(Model = "ARIMA (0, 1, 0) with Drift")  
  
ets\_var7\_params\_df <- ets\_var7\_params\_df %>%  
 mutate(Model = "ETS")  
  
# Combine data frames  
coef\_df <- rbind(arima\_var7\_coef\_df, arima\_var7\_man\_coef\_df, arima\_var7\_drift\_coef\_df, ets\_var7\_params\_df)  
  
# Print table  
kable(coef\_df, caption = "Model Coefficients for S03 Var05")  
# Create a data frame to compare the models  
model\_comparison\_var7 <- data.frame(  
 Model = c("AUTO.ARIMA", "ARIMA(1,0,0)", "ARIMA(0,1,0) with drift", "ETS"),  
 AIC = c(arima\_var7$aic, arima\_var7\_man$aic, arima\_var7\_drift$aic, ets\_var7$aic),  
 BIC = c(arima\_var7$bic, arima\_var7\_man$bic, arima\_var7\_drift$bic, ets\_var7$bic),  
 Log\_Likelihood = c(arima\_var7$loglik, arima\_var7\_man$loglik, arima\_var7\_drift$loglik, ets\_var7$loglik)  
)  
  
# Print the model comparison table  
print(model\_comparison\_var7)  
# Residuals for ARIMA(0,1,0) with drift  
checkresiduals(arima\_var7\_drift)  
# Generate predictions for the next 200 periods  
arima\_var7\_forecast <- forecast(arima\_var7\_drift, h = 200)  
arima\_var7\_forecast$mean  
plot(arima\_var7\_forecast)  
# Add the forecasts to the original data  
# Ensure the forecast has the same time index as the original series  
arima\_var7\_forecast$mean <- ts(arima\_var7\_forecast$mean, start = start(var7), frequency = frequency(var7))  
  
# Now you can add them together  
var7\_complete <- c(var7, arima\_var7\_forecast$mean)  
length(var7)  
length(arima\_var7\_forecast$mean)  
length(var7\_complete)  
# Convert the forecasts to a dataframe to bind with the category\_three dataframe  
var7\_complete\_df <- data.frame(  
 SeriesInd = time(var7\_complete, origin = "1899-12-30"),  
 Var07 = as.numeric(var7\_complete)  
)  
  
var7\_complete\_df$SeriesInd <- as.numeric(var7\_complete\_df$SeriesInd) + 40668  
  
# Replace Var05 in category\_three with Var05 from var5\_complete  
category\_three\_test2 <- category\_three\_test |>  
 left\_join(var7\_complete\_df, by = "SeriesInd") |>  
 select(-Var07.x) |>  
 rename(Var07 = Var07.y)  
  
# Check the updated category\_three  
tail(category\_three\_test2)  
write\_xlsx(category\_three\_test2, here("Project\_1", "Prompt", "category\_three\_forecasted.xlsx"))  
# Extract category three  
category\_three <- data\_list$`S03` |>  
 select(-category) # drop the category column  
  
summary(category\_five)  
kable(head(category\_five), caption = "First 6 rows of the S03 data")  
cat5\_var3 <- category\_five |>  
 select(SeriesInd, Var03)  
  
# Convert to tsibble  
cat5\_var3 <- as\_tsibble(cat5\_var3, index = SeriesInd)   
  
# Check the tsibble object  
#length(cat5\_var3$SeriesInd)  
  
# Generate summary statistics  
summary\_stats <- summary(cat5\_var3)  
  
# Convert the summary table to a data frame  
summary\_df <- as.data.frame(as.table(summary\_stats))  
  
# Clean up the data frame  
summary\_df <- summary\_df %>%  
 rename(Statistic = Var1, Variable = Var2, Value = Freq) %>%  
 mutate(Value = trimws(Value))  
  
# Split the Value column into separate statistic and value columns  
summary\_df <- summary\_df %>%  
 separate(Value, into = c("Statistic", "Value"), sep = ":", convert = TRUE) %>%  
 mutate(Statistic = trimws(Statistic), Value = as.numeric(trimws(Value)))  
  
# Remove the extra 'NA' column, if present  
summary\_df <- summary\_df %>%  
 filter(!Statistic %in% NA)  
  
# Pivot the summary data frame wider  
summary\_wide <- summary\_df %>%  
 pivot\_wider(names\_from = Statistic, values\_from = Value)  
  
# Print the summary table  
kable(summary\_wide, caption = "Summary Statistics for Variable 'Var03'")  
# Remove last 140 rows  
n <- nrow(cat5\_var3)  
cat5\_train <- cat5\_var3[-c((n-139):n), ]  
  
# Print the rows of missing values of the entire data set  
cat5\_train |>  
 filter(is.na(Var03))  
  
# Impute missing values  
cat5\_train$Var03 <- na.approx(cat5\_train$Var03)  
  
# Check for missing values to make sure they were filled in  
cat5\_train |>  
 filter(is.na(Var03))  
# Add the implicitly missing rows  
cat5\_train <- cat5\_train |>   
 fill\_gaps()  
  
# Fill NAs with the last value  
cat5\_train$Var03 <- na.locf(cat5\_train$Var03)  
  
# Check for missing values  
cat5\_train |>  
 filter(is.na(Var03))  
 # Plot the time series  
cat5\_train |>  
 ggplot(aes(x = SeriesInd, y = Var03)) +  
 geom\_point(color = "red") +  
 labs(title = "Time Series Plot", x = "Date", y = "Value") +  
 theme\_minimal()  
findfrequency(cat5\_train$Var03)  
  
adf.test(cat5\_train$Var03)  
  
ndiffs(cat5\_train$Var03)  
  
# Take the first difference  
var3\_diff <- diff(cat5\_train$Var03)  
  
# Plot the differenced data  
ggplot() +  
 geom\_line(aes(x = cat5\_train$SeriesInd[-1], y = var3\_diff), color = "blue", alpha = 0.5) +  
 labs(title = "Time Series Plot", x = "Date", y = "Value", color = "Variable") +  
 theme\_minimal()  
  
adf.test(var3\_diff)  
# ACF and PACF plots  
ggtsdisplay(var3\_diff)  
# Fit ARIMA model  
arima\_model\_diff <- auto.arima(var3\_diff)  
  
# Summary of ARIMA model  
summary\_arima <- summary(arima\_model\_diff)  
# Fit ARIMA model to original data  
arima\_model <- Arima(cat5\_train$Var03, order = c(0,1,1))  
  
# Summary of ARIMA model  
summary\_arima <- summary(arima\_model)  
# Fit ARIMA model with drift  
arima\_drift\_model <- Arima(cat5\_train$Var03, order = c(0,1,1), include.drift = T)  
  
# Summary of ARIMA model with drift  
summary\_arima\_drift <- summary(arima\_drift\_model)  
# Fit ETS model  
ets\_model <- ets(cat5\_train$Var03)  
  
# Summary of ETS model  
summary\_ets <- summary(ets\_model)  
# Convert data to Prophet format  
cat5\_train\_prophet <- cat5\_train |>  
 rename(ds = SeriesInd, y = Var03) |>  
 mutate(ds = as.Date(ds, origin = "1899-12-30"))  
  
head(cat5\_train\_prophet)  
# Fit Prophet model  
prophet\_model <- prophet(cat5\_train\_prophet, daily.seasonality = F, weekly.seasonality = F, yearly.seasonality = F)  
  
# Summary of Prophet model  
summary\_prophet <- summary(prophet\_model)  
# Check residuals  
checkresiduals(arima\_model)  
checkresiduals(arima\_drift\_model)  
checkresiduals(ets\_model)  
# Predict on the training data  
df\_train\_pred <- predict(prophet\_model, cat5\_train\_prophet)  
  
# Calculate residuals  
residuals <- cat5\_train\_prophet$y - df\_train\_pred$yhat  
  
# Create a time series object  
residuals\_ts <- ts(residuals, start = c(year(min(cat5\_train\_prophet$ds)), month(min(cat5\_train\_prophet$ds))), frequency = 365)  
  
# Use checkresiduals() function from the forecast package  
checkresiduals(residuals\_ts)  
# Generate predictions for the next 200 periods  
arima\_model\_forecast <- forecast(arima\_model, h = 200)  
arima\_drift\_model\_forecast <- forecast(arima\_drift\_model, h = 200)  
ets\_model\_forecast <- forecast(ets\_model, h = 200)  
# Create a dataframe for future dates (e.g., forecasting the next 200 days)  
future <- make\_future\_dataframe(prophet\_model, periods = 200)  
  
# Generate forecasts  
prophet\_forecast <- predict(prophet\_model, future)  
# Convert forecasts to ggplot objects  
gg\_arima <- autoplot(arima\_model\_forecast) +   
 labs(title = "ARIMA Model Forecast")  
  
gg\_arima\_drift <- autoplot(arima\_drift\_model\_forecast) +   
 labs(title = "ARIMA with Drift Model Forecast")  
  
gg\_ets <- autoplot(ets\_model\_forecast) +   
 labs(title = "ETS Model Forecast")  
  
prophet\_plot <- plot(prophet\_model, prophet\_forecast)  
gg\_prophet <- as.ggplot(prophet\_plot)  
  
# Arrange plots in a 2x2 grid  
grid.arrange(gg\_arima, gg\_arima\_drift, gg\_ets, gg\_prophet, ncol = 2)  
# Extract the mean forecasted values  
forecasted\_values <- prophet\_forecast$yhat  
  
# Combine the actual values with the forecasted values  
actual\_values <- cat5\_train$Var03  
  
# Create the combined vector  
var\_3\_complete <- c(actual\_values, forecasted\_values[(length(actual\_values) + 1):length(forecasted\_values)])  
# Select last 200 entries from prophet\_forecast$yhat  
future\_forecast <- tail(prophet\_forecast$yhat, 200)  
  
# Combine actual data points and future forecasts  
var\_3\_complete <- c(cat5\_train\_prophet$y, future\_forecast)  
# convert to a dataframe and add the SeriesInd column with the correct dates  
var\_3\_complete\_df <- data.frame(  
 SeriesInd = prophet\_forecast$ds,  
 Var03 = as.numeric(var\_3\_complete)  
)  
  
# View  
head(var\_3\_complete\_df)  
# Convert the SeriesInd column to a numeric value  
var\_3\_complete\_df$SeriesInd <- time(var\_3\_complete\_df$SeriesInd)  
  
# Add 40668 to the SeriesInd column to convert it back to Excel date serials  
var\_3\_complete\_df$SeriesInd <- as.numeric(var\_3\_complete\_df$SeriesInd) + 40668  
  
# View  
head(var\_3\_complete\_df)  
# Replace Var03 in category\_five with Var03 from var\_3\_complete  
category\_five\_test <- category\_five |>  
 left\_join(var\_3\_complete\_df, by = "SeriesInd") |>  
 select(-Var03.x) |>  
 rename(Var03 = Var03.y)  
  
# Check the updated category\_five  
head(category\_five\_test)  
tail(category\_five\_test)  
# Save the updated data  
write\_xlsx(category\_five\_test, here("Project\_1", "Prompt", "category\_five\_forecasted.xlsx"))