VIRGINIA COMMONWEALTH UNIVERSITY

Statistical analysis and modelling (SCMA 632)

A7a-Basics of Time Series Analysis (Univariate)

ALEENA MARY ABRAHAM

V01150203

Date of Submission: 17-07-2025

CONTENTS

INTRODUCTION

INTRODUCTION

In today's rapidly evolving financial markets, accurate stock price forecasting plays a critical role in guiding investment decisions, risk management, and strategic financial planning. Apple Inc. (AAPL), one of the most influential technology companies in the world, is closely followed by investors globally due to its strong market capitalization, innovation pipeline, and shareholder value generation.

This project focuses on time series forecasting of Apple's stock price using both univariate statistical models (like Holt-Winters and ARIMA) and multivariate machine learning models (such as Random Forest, and Decision Trees). By comparing traditional forecasting techniques with advanced machine learning approaches, the goal is to identify which models best capture the trend, seasonality, and structure of Apple's stock data.

OBJECTIVES

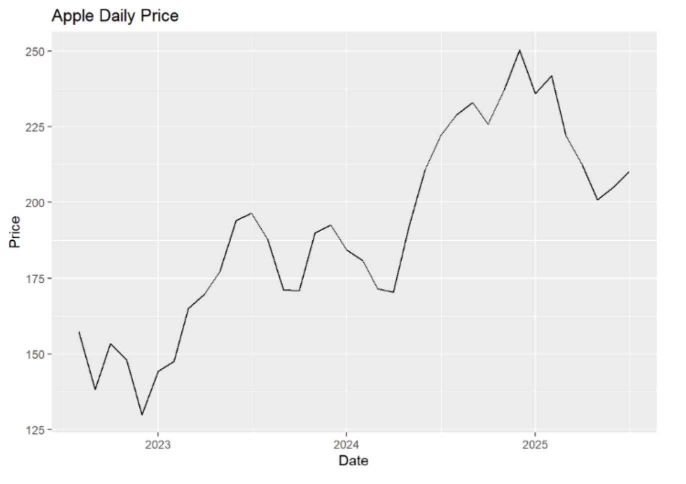
- Clean the data, check for outliers and missing values, interpolate the data if there are any missing values, and plot a line graph of the data neatly named
- Convert the data to monthly and decompose time series into the components using additive and multiplicative models.
- Apply Holt-Winters Exponential Smoothing to model seasonality and trend in the monthly data.
- Fit ARIMA/SARIMA models on both daily and monthly series to assess their forecasting power and perform diagnostics.
- Apply Decision Tree and Random Forest Regressors to predict price based on temporal trends

BUSINESS SIGNIFICANCE

1. Predicting future stock prices empowers individual and institutional investors to make informed buy, hold, or sell decisions, optimizing returns and managing risks.
2. Forecasting volatile price movements helps portfolio managers create hedging strategies to mitigate potential losses.
3. Understanding future trends in Apple's stock can assist businesses and analysts in benchmarking, valuation modeling, and strategic allocation of resources.
4. Comparing conventional models with AI-driven approaches (LSTM and Random Forest) provides insight into how machine learning enhances financial forecasting accuracy and efficiency.

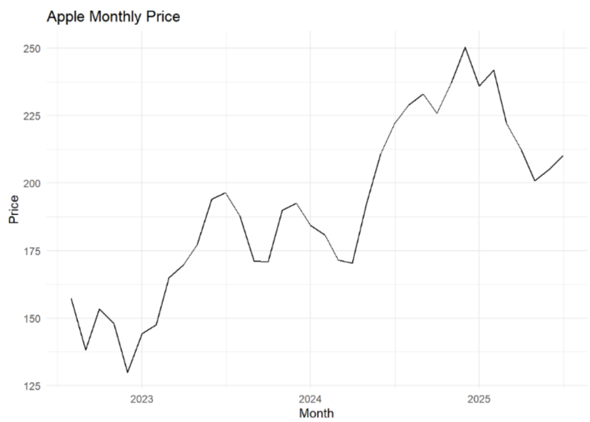
RESULTS AND INTERPRETATION

1. Daily Aggregation



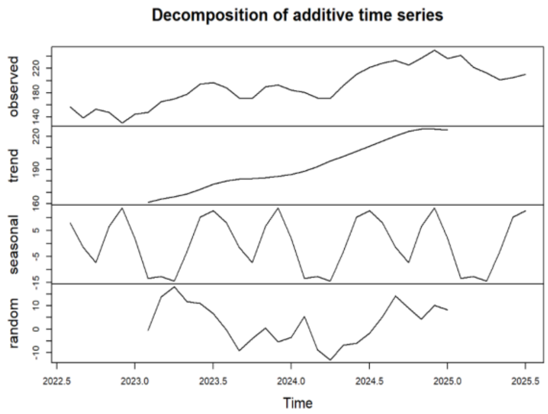
- The graph reflects typical stock market behavior — growth, correction, and recovery.
- The stock price showed a steady and strong upward trend starting from around $140 in early 2023 to nearly $250 by mid-2024.
- This visual helps in selecting the appropriate forecasting model — ARIMA for capturing trend and volatility, and Holt-Winters/SARIMA if seasonality emerges in decomposed components.

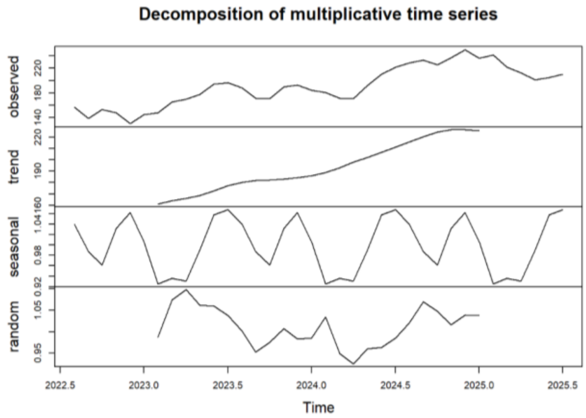
2. Monthly Aggregation



- 2023: A gradual upward trend begins from $140 to $190.
- 2024: Mid-year dip observed, then a strong rally towards end-2024 to $250.
- 2025: A sharp drop followed by a minor recovery.

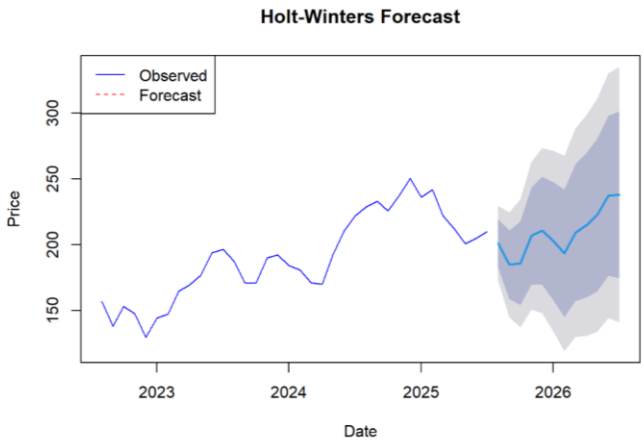
3. Time Series Decomposition (Additive and Multiplicative)





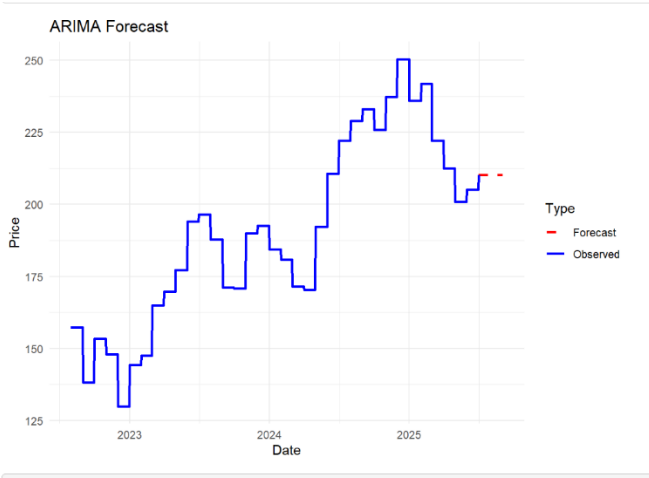
- The additive model (Observed = Trend + Seasonal + Random) seems to be a reasonable fit for this data, as the components effectively separate distinct aspects of the series behavior.
- The multiplicative model (Observed = Trend × Seasonal × Random) is used when the magnitude of the seasonal fluctuations increases or decreases in proportion to the level of the series.

4. Holt-Winters Forecast



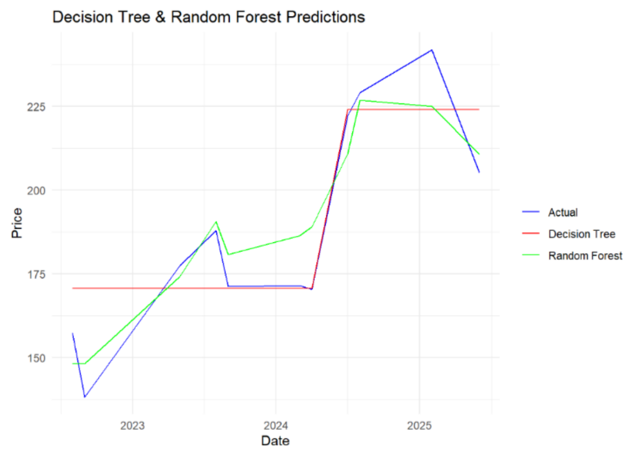
- The plot shows that the Holt-Winters model was applied to forecast the time series.
- The historical data exhibits a clear upward trend with some short-term volatility.
- The forecast attempts to capture and extend these patterns into the future. It predicts a short-term dip followed by a general recovery, but with a less aggressive upward slope compared to the historical peak.

5. ARIMA Forecast



- The observed price shows volatility with periods of growth and decline.
- There was a significant upward trend starting in mid-2023, peaking in early 2025 (~250).
- This was followed by a sharp decline in 2025, bottoming near ~200.
- The ARIMA model forecast predicts a slight upward movement beyond the latest observed point, with forecasted prices around 205–210.

6. Decision Tree and Randon Forest Predictions



- The Random Forest is the superior model here, though it still does not fully capture the volatility present in the actual prices.
- The Decision Tree performs poorly, likely due to its piecewise constant nature and tendency to underfit.

RECOMMENDATIONS

This project conducted a detailed statistical analysis of IPL player performance data, with a special focus on Aiden (AK) Markram, integrating both performance metrics and salary information using R and Python.

Summary of Key Findings:

1. Apple's stock has shown a strong upward trajectory from 2023 to 2024, with short-term dips and recoveries, suggesting both trend and possible seasonality—suitable for decomposition and smoothing techniques like Holt-Winters.
2. ARIMA forecasts were moderately accurate, capturing trend reversals but not sharp fluctuations. Its linear nature makes it more suited for gradual changes.
3. The Holt-Winters model highlighted seasonal patterns and predicted a short-term correction followed by a gentle upward trend, aligning with market cyclicality.
4. Machine Learning Models Vary in Performance
    - Random Forest outperformed Decision Trees, showing better predictive accuracy, especially on capturing non-linear relationships.
    - Decision Tree underfitted the data, failing to represent volatility and trends.

Recommendations:

1. Based on the recent dip in early 2025 and forecasts suggesting mild recovery, investors may hold or accumulate gradually, anticipating a long-term rebound.
2. Sudden corrections (e.g., in 2025) underline the need for risk-adjusted strategies and possibly hedging tools like options or stop-loss orders.
3. Businesses linked to Apple's supply chain or retail ecosystem can use these insights for inventory planning, budgeting, or market entry timing.

This study demonstrates that while traditional time series models in R (like ARIMA and Holt-Winters) effectively capture trends and seasonality in Apple stock prices, machine learning models implemented in Python (such as Random Forest and LSTM) offer improved performance in handling non-linear patterns and volatility. Integrating both approaches can enhance forecasting accuracy and provide robust insights for informed investment decisions.

CODES

R Codes:

```
#### 1. Auto-install and Load Required Packages ####
install_and_load <- function(packages) {
  for (package in packages) {
    if (!require(package, character.only = TRUE)) {
      install.packages(package, dependencies = TRUE)
    }
    library(package, character.only = TRUE)
  }
}

packages <- c("quantmod", "zoo", "forecast", "ggplot2", "dplyr", "tidyr", "TTR", "rpart", "randomForest")
install_and_load(packages)

#### 2. Load and Prepare the Data ####
# Load CSV
aapl_data <- read.csv("C:/Users/Aleena Mary Abraham/OneDrive/Desktop/SCMA632_2025/DATA/Apple Stock Price History.csv")

# Clean column names
colnames(aapl_data) <- make.names(colnames(aapl_data))

# Convert Date column
aapl_data$Date <- as.Date(aapl_data$Date)

# Use Price and Vol. columns
aapl_data <- aapl_data %>% select(Date, Price, Volume = Vol.)

# Preview data
```

```r
head(aapl_data)

#### 3. Handle Missing Values ####
# Check NA count before interpolation
print("Missing values before interpolation:")
print(sum(is.na(aapl_data$Price)))

# Interpolate missing values in Price
aapl_data$Price <- na.interp(aapl_data$Price)

# Check NA count after interpolation
print("Missing values after interpolation:")
print(sum(is.na(aapl_data$Price)))

#### 4. Plot Daily Price Data ####
ggplot(aapl_data, aes(x = Date, y = Price)) +
  geom_line() +
  labs(title = "Apple Daily Price", x = "Date", y = "Price")

#### 5. Split into Training and Testing Data ####
set.seed(123)
train_index <- sample(nrow(aapl_data), 0.7 * nrow(aapl_data))
train_data <- aapl_data[train_index, ]
test_data <- aapl_data[-train_index, ]

# Show sizes
cat("Training size:", nrow(train_data), "\n")
cat("Testing size:", nrow(test_data), "\n")
```

```
#### 6. Monthly Aggregation ####
monthly_data <- aggregate(Price ~ format(Date, "%Y-%m"), aapl_data, mean)
colnames(monthly_data) <- c("Month", "Price")
monthly_data$Month <- as.Date(paste0(monthly_data$Month, "-01"))

# Plot monthly price
ggplot(monthly_data, aes(x = Month, y = Price)) +
  geom_line() +
  labs(title = "Apple Monthly Price", x = "Month", y = "Price") +
  theme_minimal()

head(monthly_data)
str(monthly_data)
monthly_data$Month <- as.Date(monthly_data$Month)

# Recalculate the start year and month correctly
start_year <- as.numeric(format(min(monthly_data$Month), "%Y"))
start_month <- as.numeric(format(min(monthly_data$Month), "%m"))

# Create a time series object from all 36 values
monthly_ts <- ts(monthly_data$Price,
                 start = c(start_year, start_month),
                 frequency = 12)

# Check again
length(monthly_ts)
frequency(monthly_ts)


#### 7. Time Series Decomposition ####
```

```r
monthly_ts <- ts(monthly_data$Price,
                 start = c(as.numeric(format(min(monthly_data$Month), "%Y")),
                           as.numeric(format(min(monthly_data$Month), "%m"))),
                 frequency = 12)

# Additive decomposition
plot(decompose(monthly_ts, type = "additive"))

# Multiplicative decomposition
plot(decompose(monthly_ts, type = "multiplicative"))

#### 8. Holt-Winters Forecasting ####
hw_model <- HoltWinters(monthly_ts, seasonal = "additive")
hw_forecast <- forecast(hw_model, h = 12)

# Plot forecast
plot(hw_forecast, main = "Holt-Winters Forecast", xlab = "Date", ylab = "Price")
lines(monthly_ts, col = "blue")
legend("topleft", legend = c("Observed", "Forecast"), col = c("blue", "red"), lty = 1:2)



#### 9. Daily Time Series & ARIMA ####
# Complete daily sequence
daily_data <- aapl_data %>%
  complete(Date = seq.Date(min(Date), max(Date), by = "day")) %>%
  fill(Price, .direction = "downup")

# Interpolate again just in case
daily_data$Price <- na.approx(daily_data$Price)
```

```r
# Convert to time series
daily_ts <- ts(daily_data$Price, frequency = 365,
               start = c(as.numeric(format(min(daily_data$Date), "%Y")),
                         as.numeric(format(min(daily_data$Date), "%j"))))

# ARIMA model
arima_model <- auto.arima(daily_ts)
summary(arima_model)

# Forecast next 63 days
arima_forecast <- forecast(arima_model, h = 63)

# Forecast dataframe
forecast_df <- data.frame(Date = seq(max(daily_data$Date) + 1, by = "day", length.out = 63),
                          Price = as.numeric(arima_forecast$mean),
                          Type = "Forecast")

# Combine with observed
daily_data_plot <- daily_data %>% select(Date, Price) %>% mutate(Type = "Observed")
plot_data <- rbind(daily_data_plot, forecast_df)

# Plot ARIMA
ggplot() +
  geom_line(data = plot_data, aes(x = Date, y = Price, color = Type, linetype = Type), size =
1) +
  labs(title = "ARIMA Forecast", x = "Date", y = "Price") +
  scale_color_manual(values = c("Observed" = "blue", "Forecast" = "red")) +
  scale_linetype_manual(values = c("Observed" = "solid", "Forecast" = "dashed")) +
  theme_minimal()
```

```
#### 10. Decision Tree & Random Forest ####

# Create numeric version of Date
train_data$Date_num <- as.numeric(train_data$Date)
test_data$Date_num <- as.numeric(test_data$Date)

# Model: Decision Tree
model_dt <- rpart(Price ~ Date_num, data = train_data, method = "anova")
pred_dt <- predict(model_dt, test_data)

# Model: Random Forest
model_rf <- randomForest(Price ~ Date_num, data = train_data)
pred_rf <- predict(model_rf, test_data)

# Store predictions
test_data$Pred_DT <- pred_dt
test_data$Pred_RF <- pred_rf

# Plot actual vs predictions
ggplot(test_data, aes(x = Date)) +
  geom_line(aes(y = Price, color = "Actual")) +
  geom_line(aes(y = Pred_DT, color = "Decision Tree")) +
  geom_line(aes(y = Pred_RF, color = "Random Forest")) +
  labs(title = "Decision Tree & Random Forest Predictions", x = "Date", y = "Price") +
  scale_color_manual("", values = c("Actual" = "blue", "Decision Tree" = "red", "Random Forest" = "green")) +
  theme_minimal()
```