Investment Performance Tracker

Nathan Kim

June 17, 2025

Contents

Introduction

This project builds an Investment Performance Tracker in R using real-time and historical financial data. The goal is to enable data-driven investment decisions through trend visualization and price forecasting. The tracker collects stock data, calculates return metrics, visualizes price trends with moving averages, and generates short-term forecasts using ARIMA models.

```r
knitr::opts_chunk$set(echo = TRUE, fig.width=7, fig.height=5)
options(digits = 4)
library(lubridate)
library(quantmod)
library(ggplot2)
library(dplyr)
library(zoo)
library(forecast)
library(tseries)
```

Data Collection and Preparation

I use the `quantmod` package to pull historical stock data from Yahoo Finance. Stocks included in this analysis are AAPL, MSFT, and GOOGL.

- Pulling historical data from Yahoo Finance, putting into tibble

```r
tickers <- c("AAPL", "MSFT", "GOOGL")

stock_data <- lapply(tickers, function(ticker) {
  getSymbols(ticker, src = "yahoo", from = Sys.Date() - 365, auto.assign = FALSE) %>%
    fortify.zoo() %>%
    as_tibble() %>%
    rename(date = Index) %>%  # Rename the date column explicitly
    rename_with(
      ~ c("open", "high", "low", "close", "volume", "adjusted"),
      .cols = 2:7
    ) %>%
    mutate(ticker = ticker)
}) %>%
  bind_rows()
```

Return Analysis

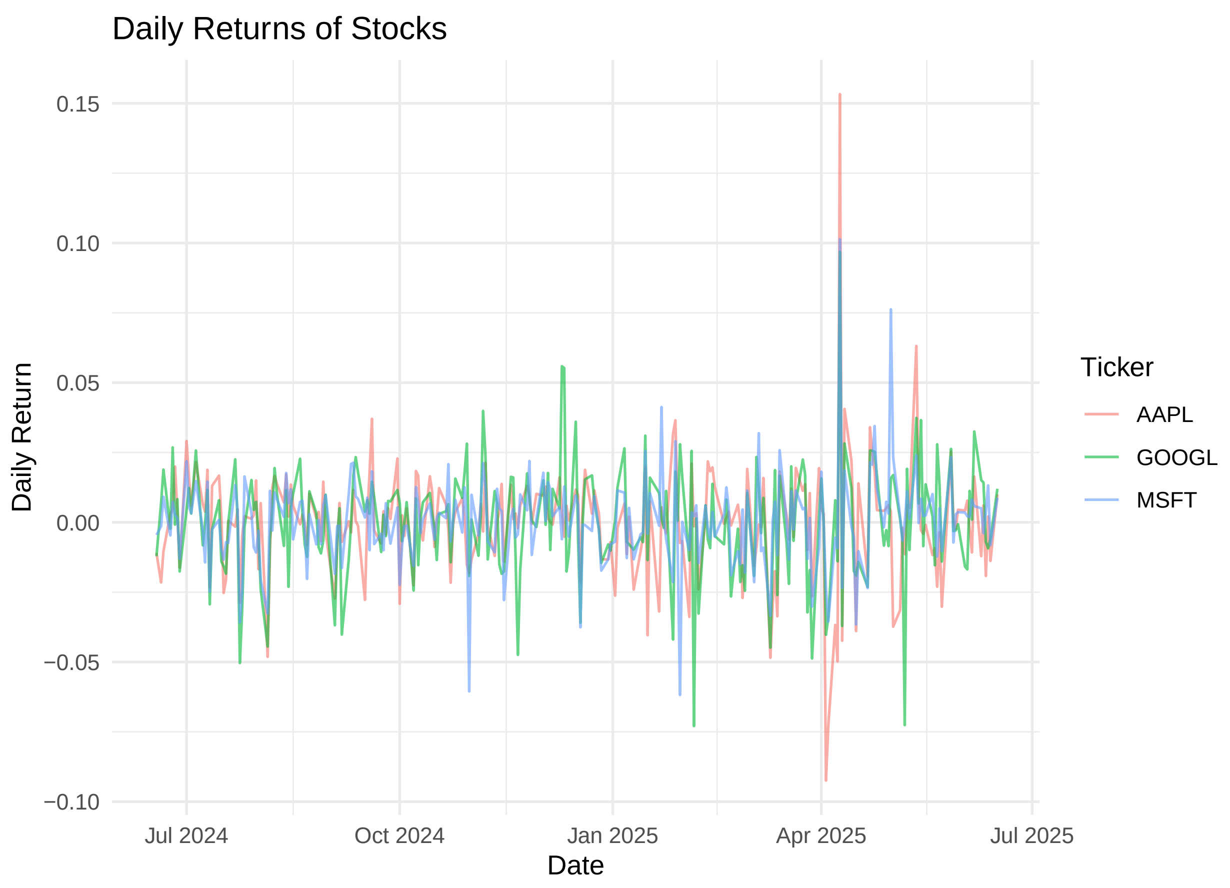
I calculate daily returns for each stock to assess short-term performance and volatility.

- Calculating daily returns for each stock using adjusted close price

```r
stock_returns <- stock_data %>%
  group_by(ticker) %>%
  arrange(date) %>%
  mutate(
    daily_return = (adjusted / lag(adjusted)) - 1
  ) %>%
  ungroup()
```

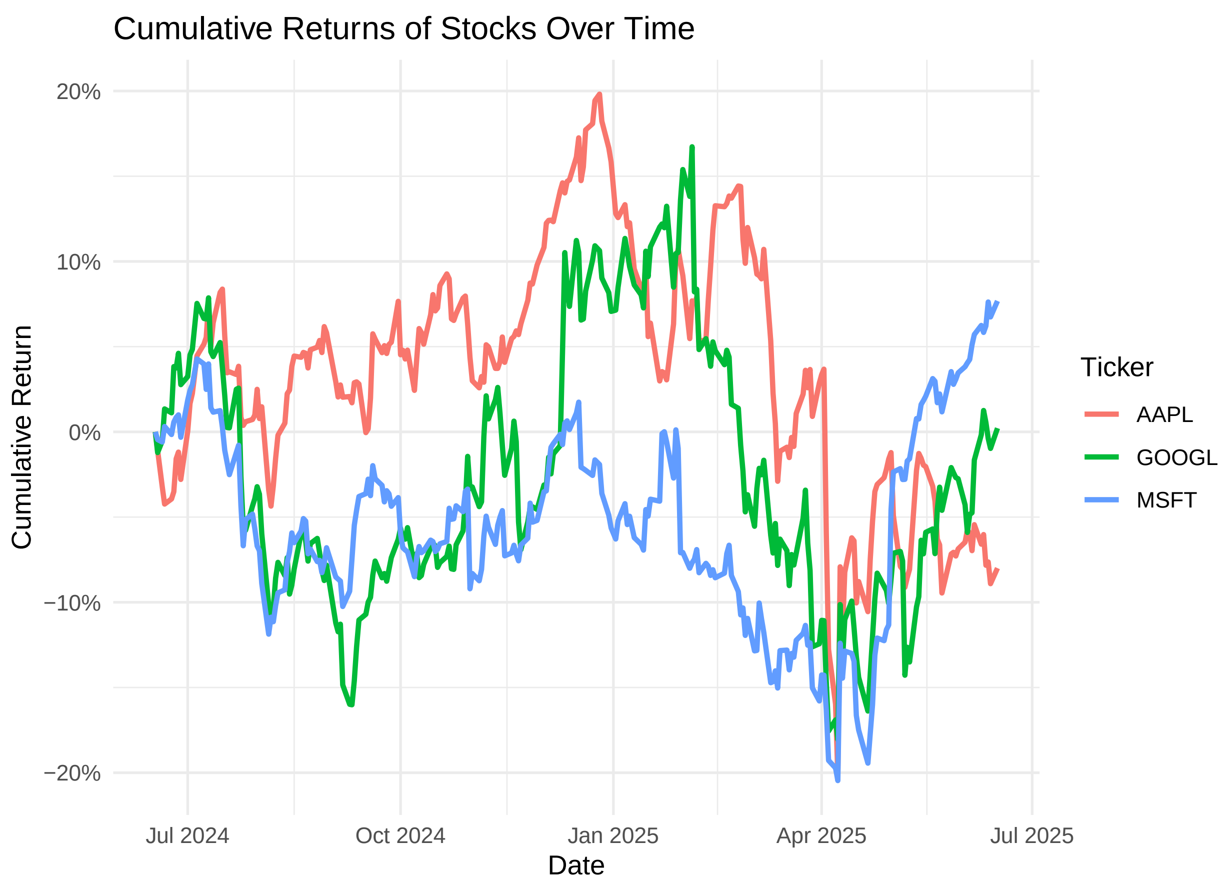
- This plot shows the daily returns of AAPL, MSFT, and GOOGL over time. Each company is represented by a colored line. This shows short term performance and volatility.

```r
ggplot(stock_returns, aes(x = date, y = daily_return, color = ticker)) +
  geom_line(alpha = 0.6) +
  labs(title = "Daily Returns of Stocks",
       x = "Date", y = "Daily Return",
       color = "Ticker") +
  theme_minimal()
```



- The next plot shows the cumulative returns of stocks for the 3 companies over time. This shows how much my investment would have grown during the time period.

```
stock_cumulative <- stock_returns %>%
  group_by(ticker) %>%
  arrange(date) %>%
  mutate(cumulative_return = cumprod(1 + coalesce(daily_return, 0)) - 1) %>%
  ungroup()

ggplot(stock_cumulative, aes(x = date, y = cumulative_return, color = ticker)) +
  geom_line(linewidth = 1) +
  labs(title = "Cumulative Returns of Stocks Over Time",
       x = "Date", y = "Cumulative Return",
       color = "Ticker") +
  scale_y_continuous(labels = scales::percent) +
  theme_minimal()
```



Trend Visualization

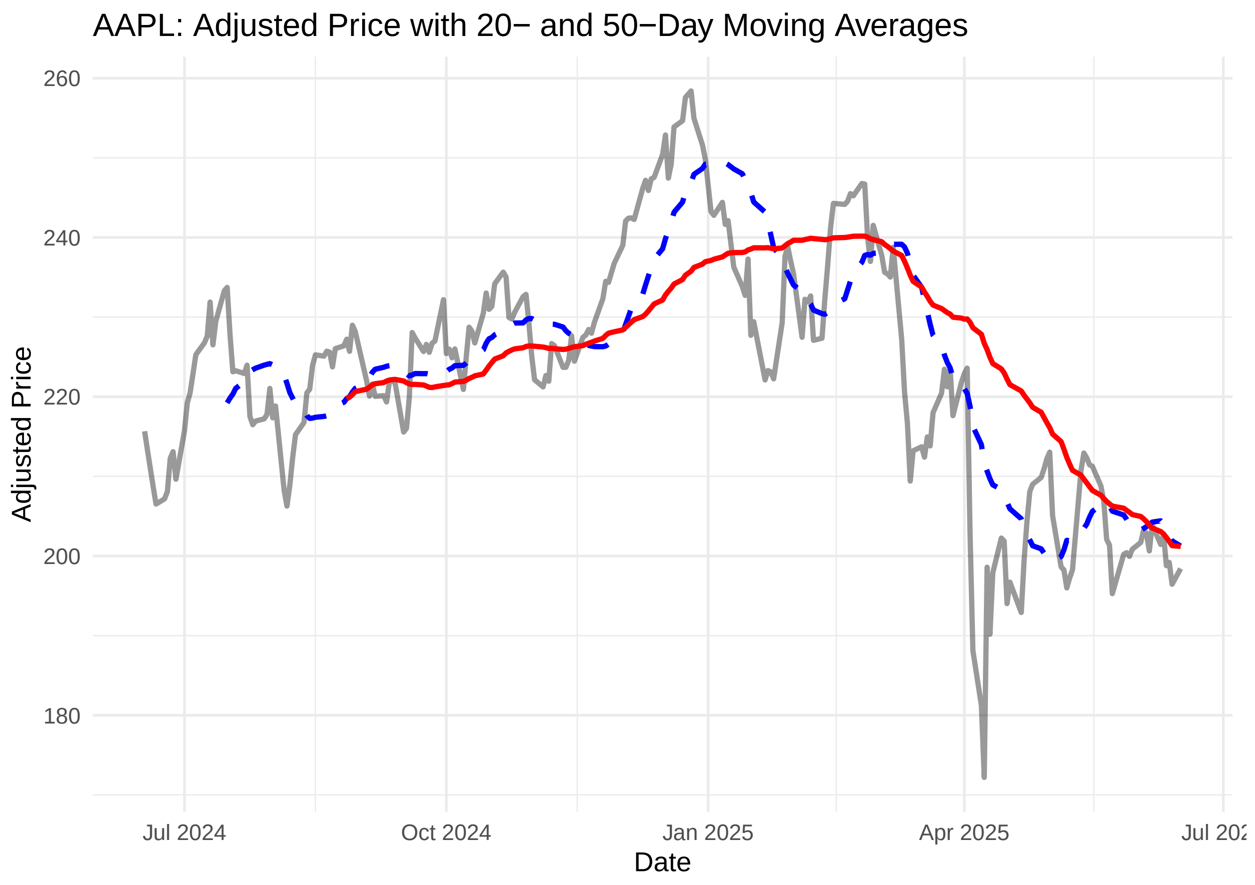
We plot the adjusted prices for each stock along with 20-day and 50-day moving averages to identify short- and medium-term trends.

The following plot shows AAPL's adjusted closing price over the past year, with two moving averages applied:

- A 20-day moving average (dashed blue) captures short-term trends.
- A 50-day moving average (solid red) reflects broader, medium-term trends.

These visualizations help identify momentum shifts through moving average crossovers.

```r
stock_ma <- stock_returns %>%
  group_by(ticker) %>%
  arrange(date) %>%
  mutate(
    ma_20 = rollmean(adjusted, k = 20, fill = NA, align = "right"),
    ma_50 = rollmean(adjusted, k = 50, fill = NA, align = "right")
  ) %>%
  ungroup()

ggplot(filter(stock_ma, ticker == "AAPL"), aes(x = date)) +
  geom_line(aes(y = adjusted), color = "black", alpha = 0.4, size = 1) +
  geom_line(aes(y = ma_20), color = "blue", size = 1, linetype = "dashed") +
  geom_line(aes(y = ma_50), color = "red", size = 1, linetype = "solid") +
  labs(title = "AAPL: Adjusted Price with 20- and 50-Day Moving Averages",
       x = "Date", y = "Adjusted Price") +
  theme_minimal()
```



Forecasting

We fit an ARIMA model to historical adjusted prices and forecast the next 30–90 trading days.

The forecast shows expected price trends and confidence intervals based on recent patterns.

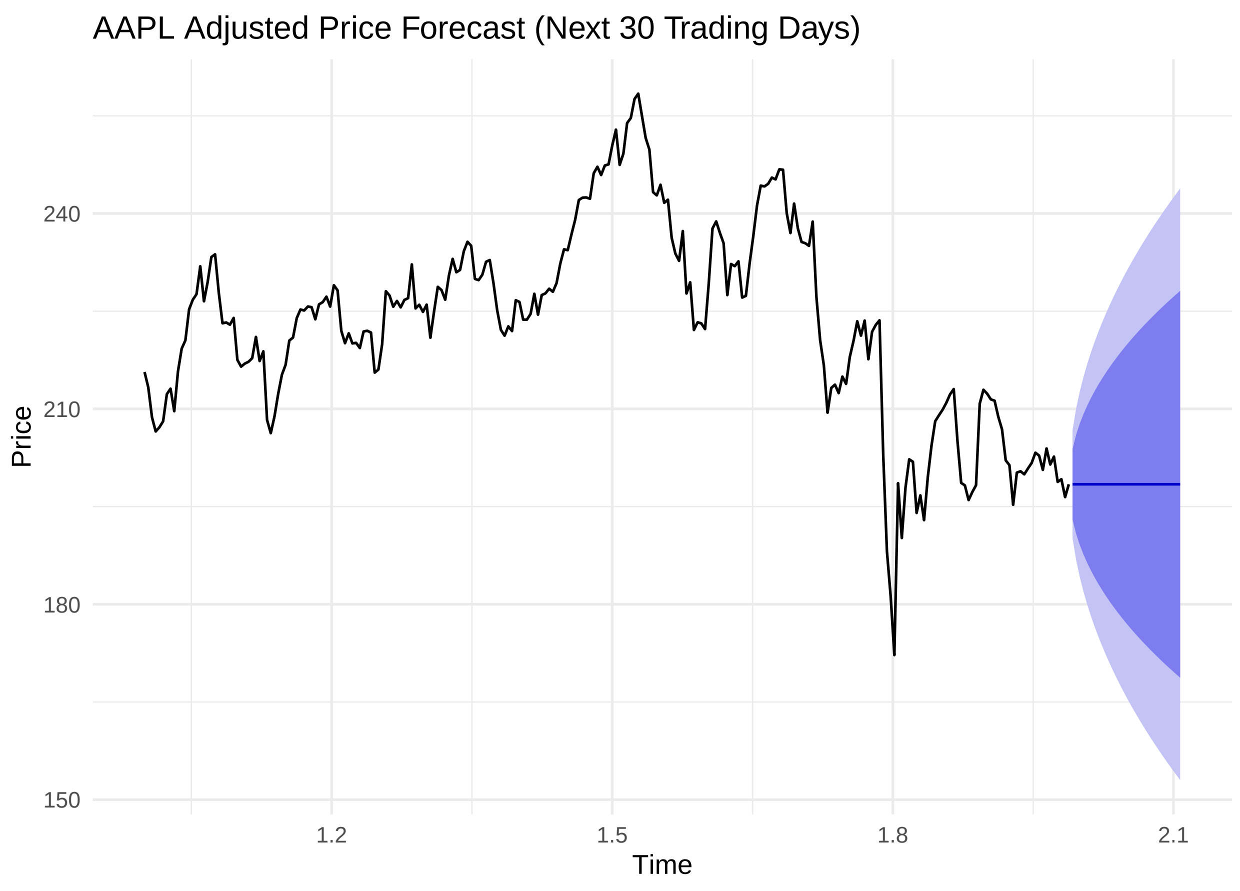
Continuing to focus on AAPL, I fit an ARIMA model.

```
aapl_ts <- stock_returns %>%
  filter(ticker == "AAPL") %>%
  arrange(date) %>%
  select(date, adjusted)

aapl_ts_obj <- ts(aapl_ts$adjusted, frequency = 252)

arima_fit <- auto.arima(aapl_ts_obj)
summary(arima_fit)
```

```
## Series: aapl_ts_obj
## ARIMA(0,1,0)
##
## sigma^2 = 17.9:  log likelihood = -712.5
## AIC=1427   AICc=1427   BIC=1430
##
## Training set error measures:
##                   ME  RMSE   MAE      MPE MAPE MASE    ACF1
## Training set -0.0681 4.222 2.851 -0.05314 1.31  NaN 0.03239
```

This plot shows the adjusted price forecast for AAPL in the next 30 trading days. The black line is the historical data while the blue line and shaded area are the forecasted path along with the 80% and 95% confidence intervals.

```
forecast_aapl <- forecast(arima_fit, h = 30)

autoplot(forecast_aapl) +
  labs(title = "AAPL Adjusted Price Forecast (Next 30 Trading Days)",
       x = "Time", y = "Price") +
  theme_minimal()
```



Model Evaluation

We split the data into training and test sets to evaluate model performance using RMSE and MAE.

The actual prices during the test period are compared with model forecasts to assess predictive accuracy.

- Evaluating the ARIMA forecasting accuracy using train and test approach.

```
n <- nrow(aapl_ts)

train_size <- floor(0.8 * n)
train_data <- aapl_ts$adjusted[1:train_size]
test_data <- aapl_ts$adjusted[(train_size + 1):n]

train_ts <- ts(train_data, frequency = 252)
test_ts <- ts(test_data, frequency = 252)

model <- auto.arima(train_ts)
summary(model)
```

```
## Series: train_ts
## ARIMA(0,1,0)
##
## sigma^2 = 13.4:  log likelihood = -540.6
## AIC=1083   AICc=1083   BIC=1087
##
## Training set error measures:
##                     ME  RMSE   MAE       MPE  MAPE MASE   ACF1
## Training set -0.06261 3.653 2.604 -0.04324 1.151  NaN 0.0433
```

```
h <- length(test_data)
forecast_test <- forecast(model, h = h)

rmse <- sqrt(mean((forecast_test$mean - test_data)^2, na.rm = TRUE))
mae <- mean(abs(forecast_test$mean - test_data), na.rm = TRUE)

cat("RMSE:", round(rmse, 4), "\n")
```

```
## RMSE: 8.026
```

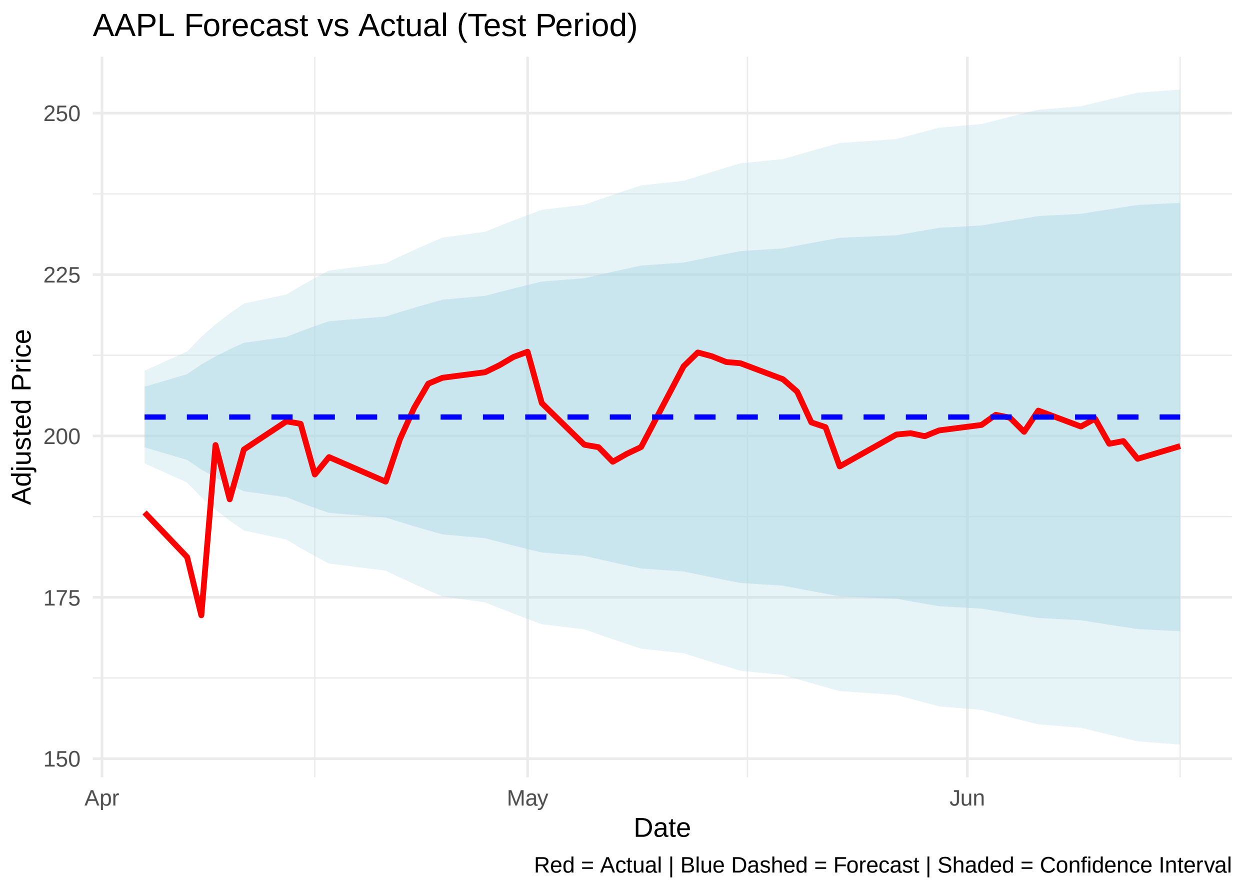
```
cat("MAE:", round(mae, 4), "\n")
```

```
## MAE: 5.897
```

Plot to visualize the Predicted Forecasting Values vs. Actual Values : - blue dashed line : forecasted path - shaded regions : 80% and 90% confidence intervals - red line : actual data

```
test_dates <- aapl_ts$date[(train_size + 1):n]

forecast_df <- data.frame(
  date = test_dates,
  actual = as.numeric(test_data),
  forecast = as.numeric(forecast_test$mean),
  lower_80 = forecast_test$lower[, 1],
  upper_80 = forecast_test$upper[, 1],
  lower_95 = forecast_test$lower[, 2],
  upper_95 = forecast_test$upper[, 2]
```

```r
)

ggplot(forecast_df, aes(x = date)) +
  geom_ribbon(aes(ymin = lower_95, ymax = upper_95), fill = "lightblue", alpha = 0.3) +
  geom_ribbon(aes(ymin = lower_80, ymax = upper_80), fill = "lightblue", alpha = 0.5) +
  geom_line(aes(y = actual), color = "red", size = 1.1) +
  geom_line(aes(y = forecast), color = "blue", linetype = "dashed", size = 1) +
  labs(title = "AAPL Forecast vs Actual (Test Period)",
       x = "Date", y = "Adjusted Price",
       caption = "Red = Actual | Blue Dashed = Forecast | Shaded = Confidence Interval") +
  theme_minimal()
```



Conclusion

This Investment Performance Tracker provides an end-to-end solution for real-time data ingestion, trend analysis, and price forecasting. It supports data-driven investment strategies by automating data collection and enabling rapid insights into stock behavior.

Future improvements could include fundamental data integration (e.g., P/E ratios), portfolio optimization, and a Shiny-based interactive dashboard.