Dataset

2025-01-23

```r
# Install the jsonlite package
install.packages("jsonlite")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)
```

```r
install.packages("dplyr")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)
```

```r
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
# Load the jsonlite package
library(jsonlite)

# Read the JSON file into R
amazon_tweet <- fromJSON("amazon.json")
apple_tweet <- fromJSON("apple.json")
facebook_tweet <- fromJSON("facebook.json")
google_tweet <- fromJSON("google.json")
```

```r
# Remove the 'id' column from all datasets
amazon_tweet <- amazon_tweet %>% select(-id)
apple_tweet <- apple_tweet %>% select(-id)
facebook_tweet <- facebook_tweet %>% select(-id)
google_tweet <- google_tweet %>% select(-id)
```

```r
# Check the structure of the loaded data
str(amazon_tweet)
```

```
## 'data.frame':    46 obs. of  5 variables:
##  $ date     : num  1.61e+12 1.60e+12 1.60e+12 1.60e+12 1.59e+12 ...
##  $ favorites: chr  "0" "0" "102013" "35724" ...
##  $ isRetweet: logi  TRUE TRUE FALSE FALSE FALSE FALSE ...
##  $ retweets : chr  "10413" "13580" "24645" "10935" ...
##  $ text     : chr  "RT @PATPmovie: Our new trailer! The Plot Against The President in 2mins &amp; 20
```

```r
# Convert the large numeric date (Unix timestamp in milliseconds) to a readable date format
amazon_tweet$date <- as.POSIXct(amazon_tweet$date / 1000, origin = "1970-01-01", tz = "UTC")
apple_tweet$date <- as.POSIXct(apple_tweet$date / 1000, origin = "1970-01-01", tz = "UTC")
facebook_tweet$date <- as.POSIXct(facebook_tweet$date / 1000, origin = "1970-01-01", tz = "UTC")
google_tweet$date <- as.POSIXct(google_tweet$date / 1000, origin = "1970-01-01", tz = "UTC")
```

```r
# Convert the 'date' column to a readable date format
amazon_tweet$date <- format(as.Date(amazon_tweet$date), "%Y-%m-%d")
apple_tweet$date <- format(as.Date(apple_tweet$date), "%Y-%m-%d")
facebook_tweet$date <- format(as.Date(facebook_tweet$date), "%Y-%m-%d")
google_tweet$date <- format(as.Date(google_tweet$date), "%Y-%m-%d")
```

```r
# Assign Sentiment variable


amazon_sentiment <- c("Neutral", "Neutral", "Negative", "Negative", "Positive", "Negative", "Negative",

"Neutral","Negative", "Negative", "Negative", "Negative", "Negative", "Negative", "Negative", "Neutral"

"Neutral", "Neutral", "Negative", "Negative", "Negative", "Negative", "Negative", "Positive", "Positive

"Negative", "Negative", "Negative", "Negative", "Negative", "Negative", "Negative", "Negative", "Negati

"Negative", "Negative", "Negative", "Negative", "Negative", "Negative")

apple_sentiment <- c("Neutral", "Neutral","Positive", "Negative", "Negative", "Positive", "Positive", "

  "Positive", "Negative", "Negative", "Neutral", "Neutral", "Neutral", "Negative", "Positive", "Neutral

  "Positive")

facebook_sentiment <- c("Neutral", "Neutral","Negative", "Negative", "Negative", "Negative", "Negative"

"Negative", "Neutral", "Positive", "Neutral", "Neutral", "Positive", "Negative", "Negative", "Neutral",

"Negative","Negative", "Negative", "Negative", "Negative", "Negative", "Negative", "Negative", "Neutral

"Negative", "Negative", "Negative", "Negative", "Negative", "Positive","Negative" )

google_sentiment <- c("Negative", "Negative", "Neutral", "Neutral", "Negative", "Negative", "Negative",

"Negative", "Negative", "Negative", "Negative", "Neutral", "Negative", "Negative", "Negative", "Neutral

"Negative", "Negative", "Positive", "Positive", "Negative", "Negative", "Negative", "Negative", "Neutra

"Negative", "Negative", "Negative", "Negative", "Negative")


# Convert Sentiment Values into Tweet data
amazon_tweet$Sentiment <- amazon_sentiment
apple_tweet$Sentiment <- apple_sentiment
facebook_tweet$Sentiment <- facebook_sentiment
google_tweet$Sentiment <- google_sentiment
```

```r
# Convert the sentiment labels to numerical values:
# 1 for Positive, 0 for Neutral, and -1 for Negative
amazon_tweet$Sentiment <- ifelse(amazon_tweet$Sentiment == "Positive", 1,
                                 ifelse(amazon_tweet$Sentiment == "Neutral", 0, -1))

apple_tweet$Sentiment <- ifelse(apple_tweet$Sentiment == "Positive", 1,
                                ifelse(apple_tweet$Sentiment == "Neutral", 0, -1))

facebook_tweet$Sentiment <- ifelse(facebook_tweet$Sentiment == "Positive", 1,
                                   ifelse(facebook_tweet$Sentiment == "Neutral", 0, -1))

google_tweet$Sentiment <- ifelse(google_tweet$Sentiment == "Positive", 1,
                                 ifelse(google_tweet$Sentiment == "Neutral", 0, -1))
```

Save the data frame to a CSV file

write.csv(amazon_data, "amazon_tweets.csv", row.names = FALSE) write.csv(apple_data, "apple_tweets.csv", row.names = FALSE)

```r
# Load necessary libraries
library(tidyquant)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo

## -- Attaching core tidyquant packages ----------------------- tidyquant 1.0.10 --
## v PerformanceAnalytics 2.0.8      v TTR                  0.24.4
## v quantmod             0.4.26     v xts                  0.14.1
## -- Conflicts ------------------------------------------ tidyquant_conflicts() --
## x zoo::as.Date()                 masks base::as.Date()
## x zoo::as.Date.numeric()         masks base::as.Date.numeric()
## x dplyr::filter()                masks stats::filter()
## x xts::first()                   masks dplyr::first()
## x dplyr::lag()                   masks stats::lag()
## x xts::last()                    masks dplyr::last()
## x PerformanceAnalytics::legend() masks graphics::legend()
## x quantmod::summary()            masks base::summary()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(dplyr)
library(tidyr)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
##
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```r
# Define a function to get stock data and process it
get_stock_data <- function(ticker, start_date, end_date) {
  stock_data <- tq_get(ticker, from = start_date, to = end_date) %>%
```

```r
    dplyr::select(symbol, date, adjusted) %>%
    tidyr::pivot_wider(names_from = symbol, values_from = adjusted) %>%
    group_by(week = floor_date(date, "week")) %>%
    summarize(price = mean(get(ticker), na.rm = TRUE))  # Calculate weekly average price

  # Convert time series data into a data frame for easier viewing
  return(data.frame(week = stock_data$week, price = stock_data$price))
}

# Set date range for Trump's presidency
start_date <- "2017-01-20"
end_date <- "2021-01-20"

# Get stock data for each company
apple_stock <- get_stock_data("AAPL", start_date, end_date)
amazon_stock <- get_stock_data("AMZN", start_date, end_date)
facebook_stock <- get_stock_data("META", start_date, end_date)
google_stock <- get_stock_data("GOOGL", start_date, end_date)
```

amazon, * 46 tweets/rt apple, * 21 tweets/rt

facebook ,* 37 tweets/rt

Google * 35 tweets/rt

```r
library(dplyr)
library(lubridate)

# Convert the date column in amazon_tweet to Date format
amazon_merged_data <- amazon_tweet %>%
  mutate(date = as.Date(date, format = "%Y-%m-%d")) %>%
  # Add a 'week' column for each tweet based on its 'date'
  mutate(week = floor_date(date, unit = "week")) %>%
  # Join the tweet data with the stock data by the week
  left_join(amazon_stock %>%
              mutate(week = as.Date(week, format = "%Y-%m-%d")),
             by = "week")

# View the result
head(amazon_merged_data)
```

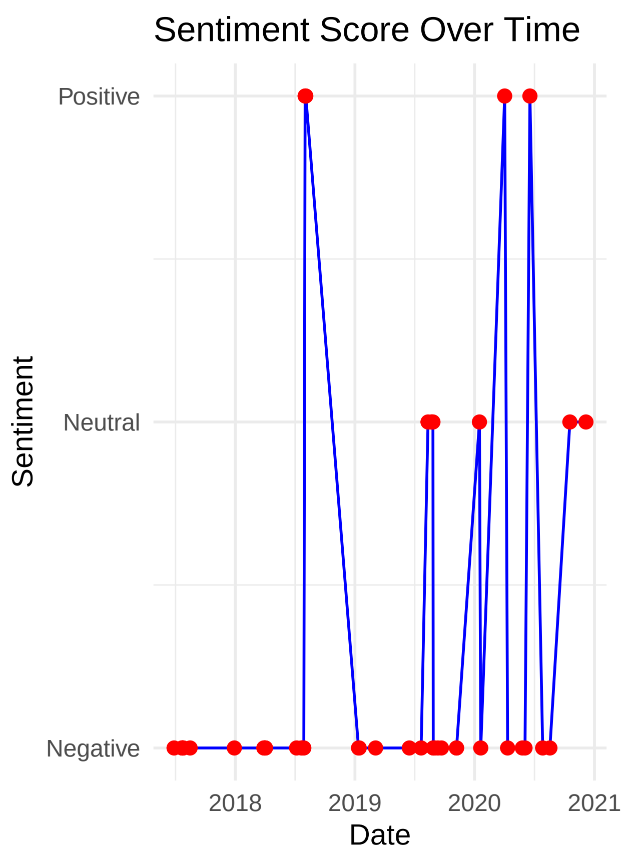
```
##         date favorites isRetweet retweets
## 1 2020-12-06         0      TRUE    10413
## 2 2020-10-18         0      TRUE    13580
## 3 2020-08-18    102013     FALSE    24645
## 4 2020-07-27     35724     FALSE    10935
## 5 2020-06-18     63840     FALSE    13623
## 6 2020-06-03    121188     FALSE    29957
##
## 1
## 2
## 3                               .@Amazon, and others in that business, should be charged (by the U.S.
## 4
## 5
## 6 Really sick to watch the Fake and totally Slanted News(?) coming out of MSDNC and CNN. It bears NO
```

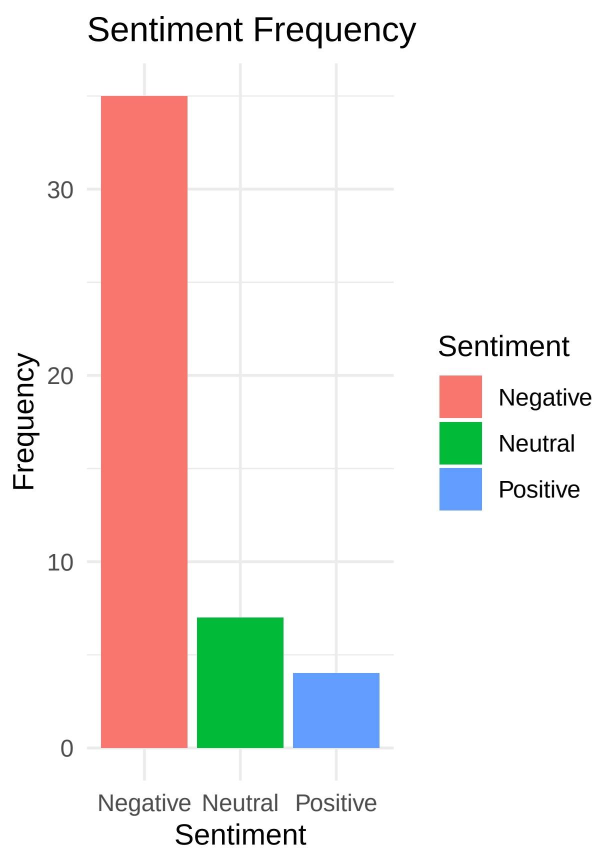
```
##   Sentiment       week    price
## 1         0 2020-12-06 156.5740
## 2         0 2020-10-18 159.8996
## 3        -1 2020-08-16 163.3747
## 4        -1 2020-07-26 153.0563
## 5         1 2020-06-14 131.5792
## 6        -1 2020-05-31 123.6545
```

```r
# Assuming amazon_tweet has the sentiment and date columns
library(ggplot2)

# Ensure that 'date' is in Date format (if it's not already)
amazon_tweet$date <- as.Date(amazon_tweet$date)

# Create the line plot
sentiment_plot <- ggplot(amazon_tweet, aes(x = date, y = Sentiment)) +
  geom_line(color = "blue") +  # Line plot of sentiment scores
  geom_point(color = "red", size = 2) +  # Points to highlight sentiment changes
  ggtitle("Sentiment Score Over Time") +
  xlab("Date") +  # Use 'Date' for clarity
  ylab("Sentiment") +
  scale_y_continuous(breaks = c(-1, 0, 1), labels = c("Negative", "Neutral", "Positive")) +  # Adjust l
  theme_minimal()  # Clean theme


# Frequency of Each Sentiment Plot
sentiment_frequency <- amazon_tweet %>%
  count(Sentiment) %>%
  mutate(Sentiment = factor(Sentiment, levels = c(-1, 0, 1), labels = c("Negative", "Neutral", "Positiv

frequency_plot <- ggplot(sentiment_frequency, aes(x = Sentiment, y = n, fill = Sentiment)) +
  geom_bar(stat = "identity") +  # Bar plot of sentiment frequencies
  ggtitle("Sentiment Frequency") +
  xlab("Sentiment") +
  ylab("Frequency") +
  theme_minimal()

# Display both plots side by side
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
grid.arrange(sentiment_plot, frequency_plot, ncol = 2)
```





Correlation

```r
# Correlation between sentiment and stock price
cor(amazon_merged_data$Sentiment, amazon_merged_data$price)
```

```
## [1] 0.2902291
```

VAR

```r
library(vars)
```

```
## Loading required package: MASS

##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select

## Loading required package: strucchange

## Loading required package: sandwich

## Loading required package: urca

## Loading required package: lmtest

##
## Attaching package: 'vars'

## The following object is masked from 'package:tidyquant':
##
```

```
##     VAR
```

```r
# Assuming weekly_data contains both sentiment and stock price columns
weekly_ts <- ts(amazon_merged_data[, c("Sentiment", "price")], start = c(2017, 1), frequency = 52)

# Fit VAR model with 2 lags
var_model <- VAR(weekly_ts, p = 2)
summary(var_model)
```

```
##
## VAR Estimation Results:
## =========================
## Endogenous variables: Sentiment, price
## Deterministic variables: const
## Sample size: 44
## Log Likelihood: -175.55
## Roots of the characteristic polynomial:
## 0.8887 0.3939 0.3939 0.002703
## Call:
## VAR(y = weekly_ts, p = 2)
##
##
## Estimation results for equation Sentiment:
## ==========================================
## Sentiment = Sentiment.l1 + price.l1 + Sentiment.l2 + price.l2 + const
##
##               Estimate Std. Error t value Pr(>|t|)
## Sentiment.l1  0.135049   0.158447   0.852   0.3992
## price.l1     -0.002026   0.016342  -0.124   0.9020
## Sentiment.l2 -0.098916   0.163805  -0.604   0.5494
## price.l2      0.006646   0.015356   0.433   0.6675
## const        -1.106626   0.440463  -2.512   0.0162 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Residual standard error: 0.6397 on 39 degrees of freedom
## Multiple R-Squared: 0.07003, Adjusted R-squared: -0.02535
## F-statistic: 0.7342 on 4 and 39 DF,  p-value: 0.5742
##
##
## Estimation results for equation price:
## ======================================
## price = Sentiment.l1 + price.l1 + Sentiment.l2 + price.l2 + const
##
##              Estimate Std. Error t value Pr(>|t|)
## Sentiment.l1   2.1130     1.3872   1.523  0.13577
## price.l1       1.0655     0.1431   7.447 5.26e-09 ***
## Sentiment.l2   2.7124     1.4341   1.891  0.06603 .
## price.l2      -0.1860     0.1344  -1.384  0.17436
## const         11.9303     3.8563   3.094  0.00365 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
```

```
## Residual standard error: 5.6 on 39 degrees of freedom
## Multiple R-Squared: 0.9537,  Adjusted R-squared: 0.9489
## F-statistic: 200.8 on 4 and 39 DF,  p-value: < 2.2e-16
##
##
##
## Covariance matrix of residuals:
##           Sentiment   price
## Sentiment    0.4092  0.2984
## price        0.2984 31.3631
##
## Correlation matrix of residuals:
##           Sentiment   price
## Sentiment   1.00000 0.08331
## price       0.08331 1.00000
```

```
# Check causality from sentiment to stock price
causality(var_model, cause = "Sentiment")
```

```
## $Granger
##
##  Granger causality H0: Sentiment do not Granger-cause price
##
## data:  VAR object var_model
## F-Test = 3.3115, df1 = 2, df2 = 78, p-value = 0.04165
##
##
## $Instant
##
##  H0: No instantaneous causality between: Sentiment and price
##
## data:  VAR object var_model
## Chi-squared = 0.30326, df = 1, p-value = 0.5818
```

```
amazon_merged_data = amazon_merged_data %>% mutate(price_change = price - lag(price, 1),
                                                   sent_2 = Sentiment - lag(Sentiment,1),
                                                   favorites = as.numeric(favorites))

# Fit the linear model with Sentiment as the predictor for price change
lm_model <- lm(price_change ~ Sentiment + sent_2 + week +favorites ,
               data = amazon_merged_data,
               na.action = na.omit)

# Display model summary
summary(lm_model)
```

```
##
## Call:
## lm(formula = price_change ~ Sentiment + sent_2 + week + favorites,
##     data = amazon_merged_data, na.action = na.omit)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -16.681  -2.091   1.516   3.616  10.836
##
```

```
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.518e+01  5.230e+01   1.055    0.298
## Sentiment    1.907e+00  2.205e+00   0.865    0.392
## sent_2      -2.114e+00  1.609e+00  -1.314    0.196
## week        -3.064e-03  2.902e-03  -1.056    0.297
## favorites   -1.881e-05  2.768e-05  -0.680    0.501
##
## Residual standard error: 6.484 on 40 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.07058,    Adjusted R-squared:  -0.02236
## F-statistic: 0.7594 on 4 and 40 DF,  p-value: 0.5579
```

```r
# Display model summary
summary(lm_model)
```

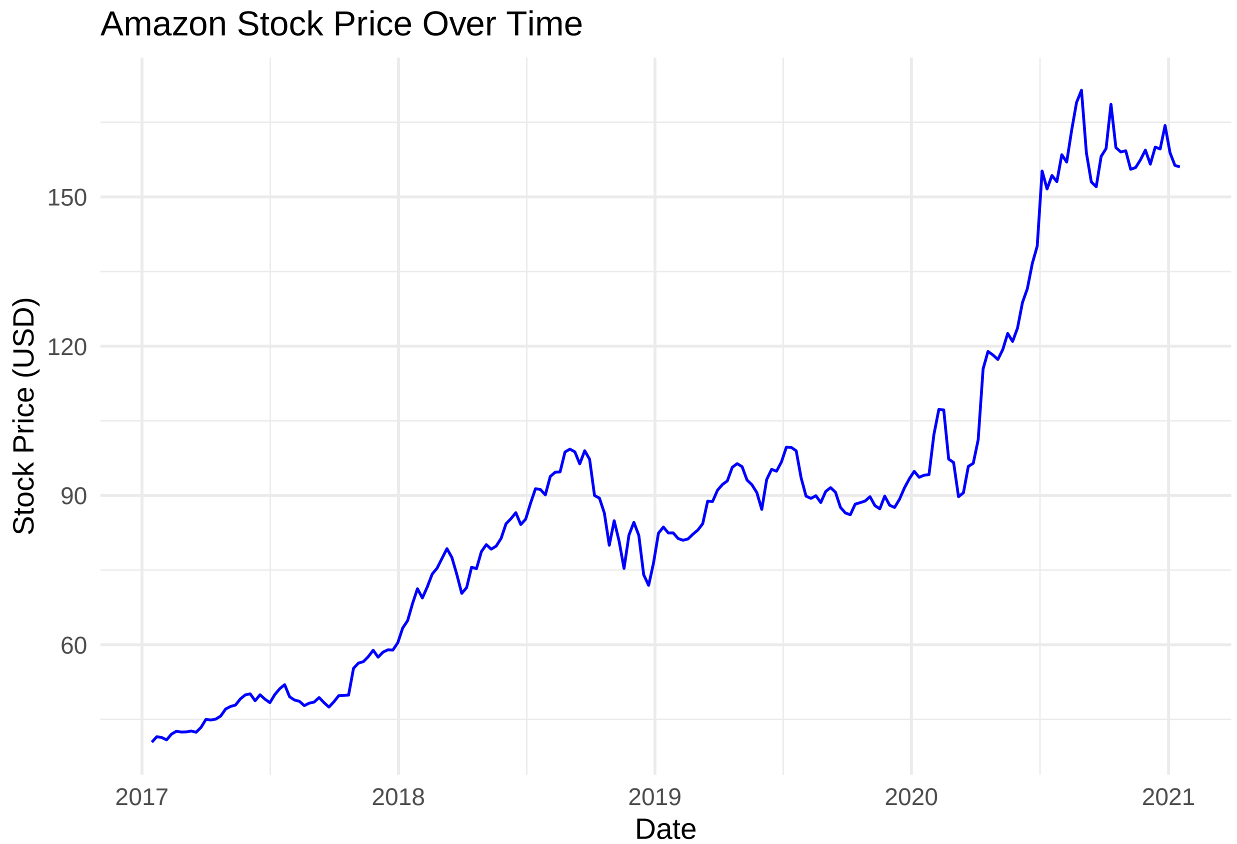
```
##
## Call:
## lm(formula = price_change ~ Sentiment + sent_2 + week + favorites,
##     data = amazon_merged_data, na.action = na.omit)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -16.681  -2.091   1.516   3.616  10.836
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.518e+01  5.230e+01   1.055    0.298
## Sentiment    1.907e+00  2.205e+00   0.865    0.392
## sent_2      -2.114e+00  1.609e+00  -1.314    0.196
## week        -3.064e-03  2.902e-03  -1.056    0.297
## favorites   -1.881e-05  2.768e-05  -0.680    0.501
##
## Residual standard error: 6.484 on 40 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.07058,    Adjusted R-squared:  -0.02236
## F-statistic: 0.7594 on 4 and 40 DF,  p-value: 0.5579
```

```r
forecast <- predict(var_model, n.ahead = 1)
print(forecast)
```

```
## $Sentiment
##                     fcst    lower    upper       CI
## Sentiment.fcst -0.902128 -2.15584 0.351584 1.253712
##
## $price
##                fcst   lower   upper       CI
## price.fcst 49.85085 38.8745 60.8272 10.97635
```

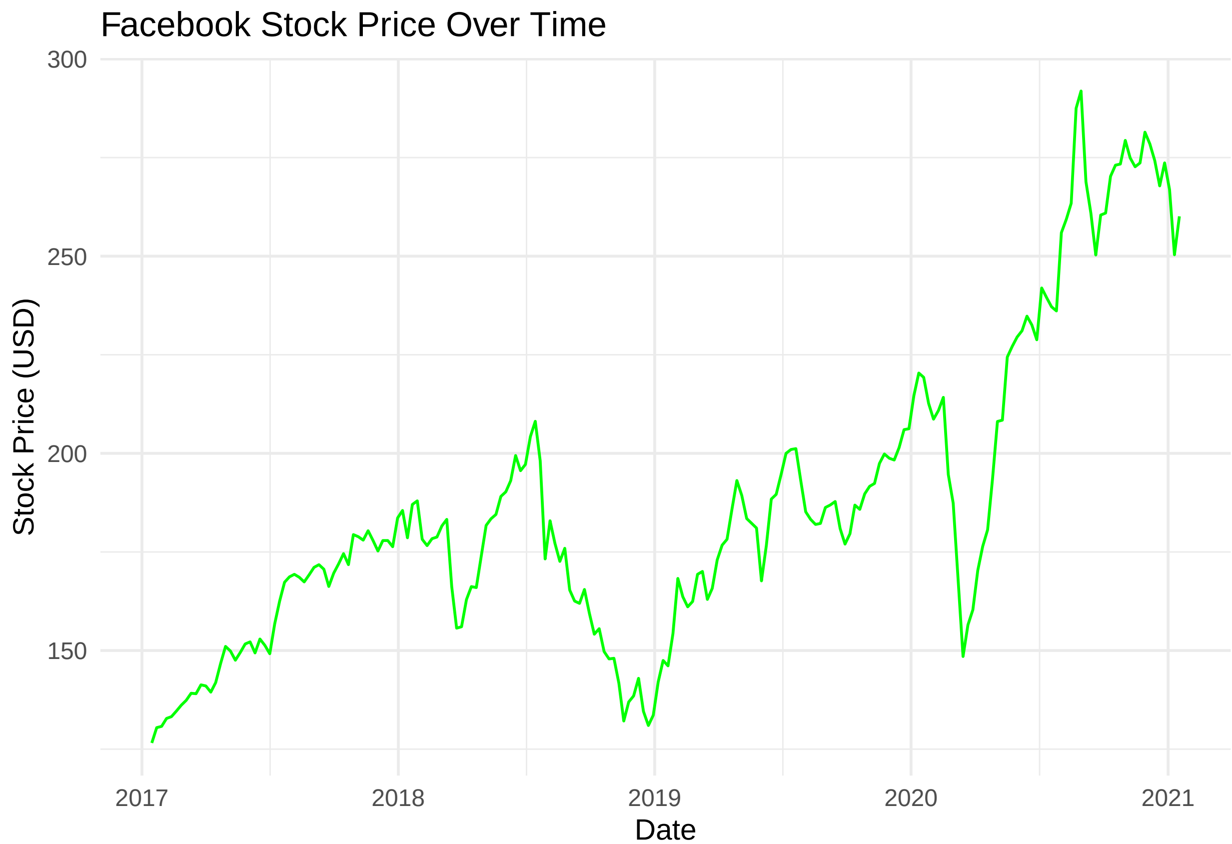
```r
library(ggplot2)

# Plot Amazon Stock Price
ggplot(amazon_stock, aes(x = week, y = price)) +
  geom_line(color = "blue") +
  labs(title = "Amazon Stock Price Over Time",
       x = "Date",
```

```r
       y = "Stock Price (USD)") +
  theme_minimal()
```




```r
# Plot Apple Stock Price
ggplot(apple_stock, aes(x = week, y = price)) +
  geom_line(color = "red") +
  labs(title = "Apple Stock Price Over Time",
       x = "Date",
       y = "Stock Price (USD)") +
  theme_minimal()
```


```r
# Plot Facebook Stock Price
ggplot(facebook_stock, aes(x = week, y = price)) +
  geom_line(color = "green") +
  labs(title = "Facebook Stock Price Over Time",
       x = "Date",
       y = "Stock Price (USD)") +
  theme_minimal()
```




```r
# Plot Google Stock Price
ggplot(google_stock, aes(x = week, y = price)) +
  geom_line(color = "purple") +
  labs(title = "Google Stock Price Over Time",
       x = "Date",
       y = "Stock Price (USD)") +
  theme_minimal()
```


Google Stock Price Over Time
Stock Price (USD)
90
80
70
60
50
40
2017
2018
2019
2020
2021
Date