Stock Forecasting Project

Libraries

```
library(tidyverse)
library(lubridate)
library(quantmod)
library(tseries)
library(forecast)
```

Data Gathering

User Inputs (Stock and Date)

```
symbol <- "AAPL"
start_date <- as.Date("2018-01-01")
end_date <- Sys.Date()
```

Stock Data Collection

```
getSymbols(symbol,
           src  = "yahoo",
           from = start_date,
           to   = end_date,
           auto.assign = TRUE)
```

```
## [1] "AAPL"
```

```
stock_data <- get(symbol)

head(stock_data)
```

```
##            AAPL.Open AAPL.High AAPL.Low AAPL.Close AAPL.Volume AAPL.Adjusted
## 2018-01-02   42.5400   43.0750  42.3150    43.0650   102223600      40.47983
## 2018-01-03   43.1325   43.6375  42.9900    43.0575   118071600      40.47280
## 2018-01-04   43.1350   43.3675  43.0200    43.2575    89738400      40.66079
## 2018-01-05   43.3600   43.8425  43.2625    43.7500    94640000      41.12372
## 2018-01-08   43.5875   43.9025  43.4825    43.5875    82271200      40.97097
## 2018-01-09   43.6375   43.7650  43.3525    43.5825    86336000      40.96627
```

Data Exploration & Feature Engineering

EDA

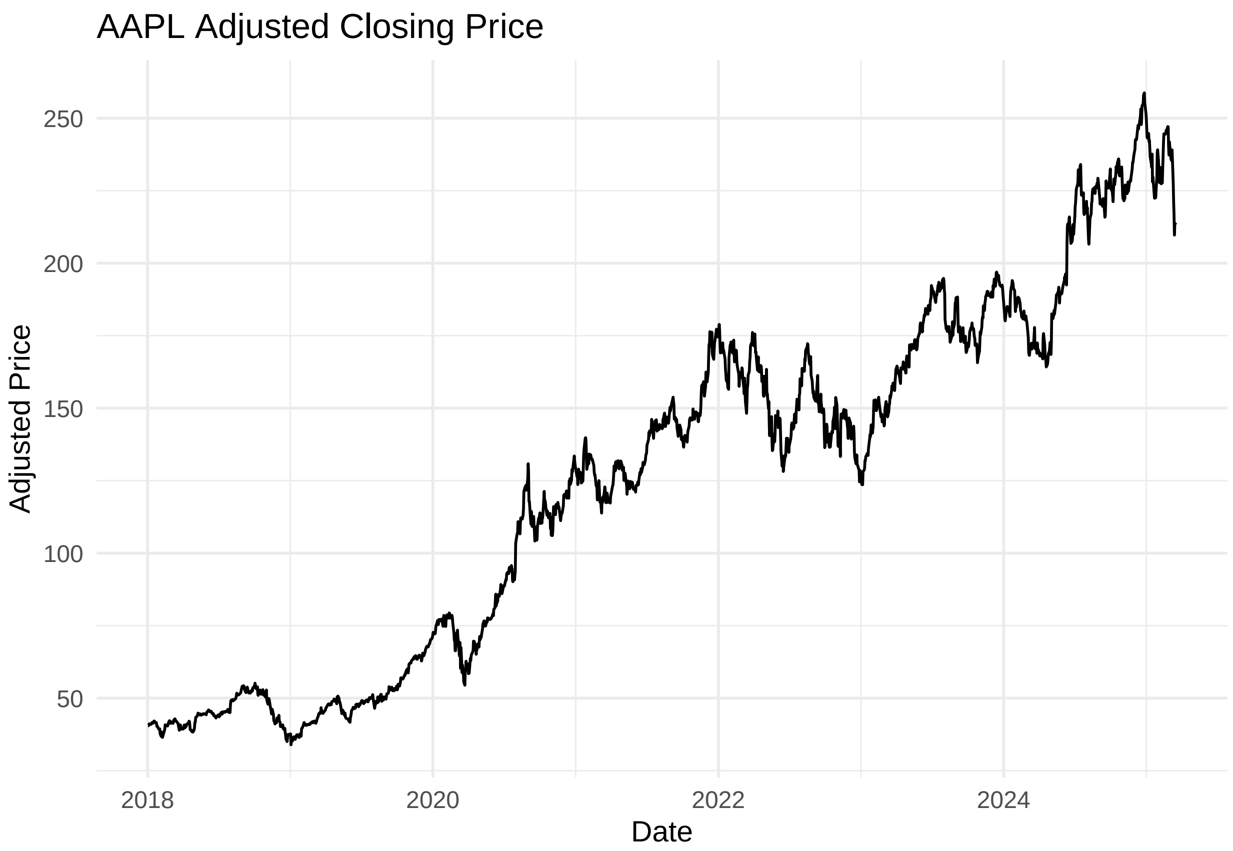
```
# Convert time-series data (xts object) to a regular tibble
df_stock <- tibble(
  date     = index(stock_data),
  open     = as.numeric(stock_data[, paste0(symbol, ".Open")]),
  high     = as.numeric(stock_data[, paste0(symbol, ".High")]),
  low      = as.numeric(stock_data[, paste0(symbol, ".Low")]),
  close    = as.numeric(stock_data[, paste0(symbol, ".Close")]),
  volume   = as.numeric(stock_data[, paste0(symbol, ".Volume")]),
  adjusted = as.numeric(stock_data[, paste0(symbol, ".Adjusted")])
)

glimpse(df_stock)
```

```
## Rows: 1,811
## Columns: 7
## $ date     <date> 2018-01-02, 2018-01-03, 2018-01-04, 2018-01-05, 2018-01-08, ~
## $ open     <dbl> 42.5400, 43.1325, 43.1350, 43.3600, 43.5875, 43.6375, 43.2900~
## $ high     <dbl> 43.0750, 43.6375, 43.3675, 43.8425, 43.9025, 43.7650, 43.5750~
## $ low      <dbl> 42.3150, 42.9900, 43.0200, 43.2625, 43.4825, 43.3525, 43.2500~
## $ close    <dbl> 43.0650, 43.0575, 43.2575, 43.7500, 43.5875, 43.5825, 43.5725~
## $ volume   <dbl> 102223600, 118071600, 89738400, 94640000, 82271200, 86336000,~
## $ adjusted <dbl> 40.47983, 40.47280, 40.66079, 41.12372, 40.97097, 40.96627, 4~
```

```
summary(df_stock)
```

```
##       date                 open             high             low        
##  Min.   :2018-01-02   Min.   : 35.99   Min.   : 36.43   Min.   : 35.50  
##  1st Qu.:2019-10-19   1st Qu.: 58.93   1st Qu.: 59.47   1st Qu.: 58.57  
##  Median :2021-08-06   Median :136.82   Median :138.59   Median :134.92  
##  Mean   :2021-08-07   Mean   :127.20   Mean   :128.59   Mean   :125.89  
##  3rd Qu.:2023-05-24   3rd Qu.:172.74   3rd Qu.:174.18   3rd Qu.:171.19  
##  Max.   :2025-03-17   Max.   :258.19   Max.   :260.10   Max.   :257.63  
##      close            volume             adjusted     
##  Min.   : 35.55   Min.   : 23234700   Min.   : 33.92  
##  1st Qu.: 59.01   1st Qu.: 60953250   1st Qu.: 56.98  
##  Median :136.96   Median : 85671900   Median :134.38  
##  Mean   :127.31   Mean   : 98860006   Mean   :125.35  
##  3rd Qu.:173.00   3rd Qu.:119555100   3rd Qu.:171.49  
##  Max.   :259.02   Max.   :426510000   Max.   :258.74  
```

```
# Plot Adjusted Closing Price over time with a dynamic title
ggplot(df_stock, aes(x = date, y = adjusted)) +
  geom_line() +
  labs(title = paste(symbol, "Adjusted Closing Price"),
       x = "Date",
       y = "Adjusted Price") +
  theme_minimal()
```



```r
# Check if there is any NA (rare to have NA)
df_stock %>%
  summarize(across(everything(), ~ sum(is.na(.))))
```

```
## # A tibble: 1 x 7
##    date  open  high   low close volume adjusted
##   <int> <int> <int> <int> <int>  <int>    <int>
## 1     0     0     0     0     0      0        0
```

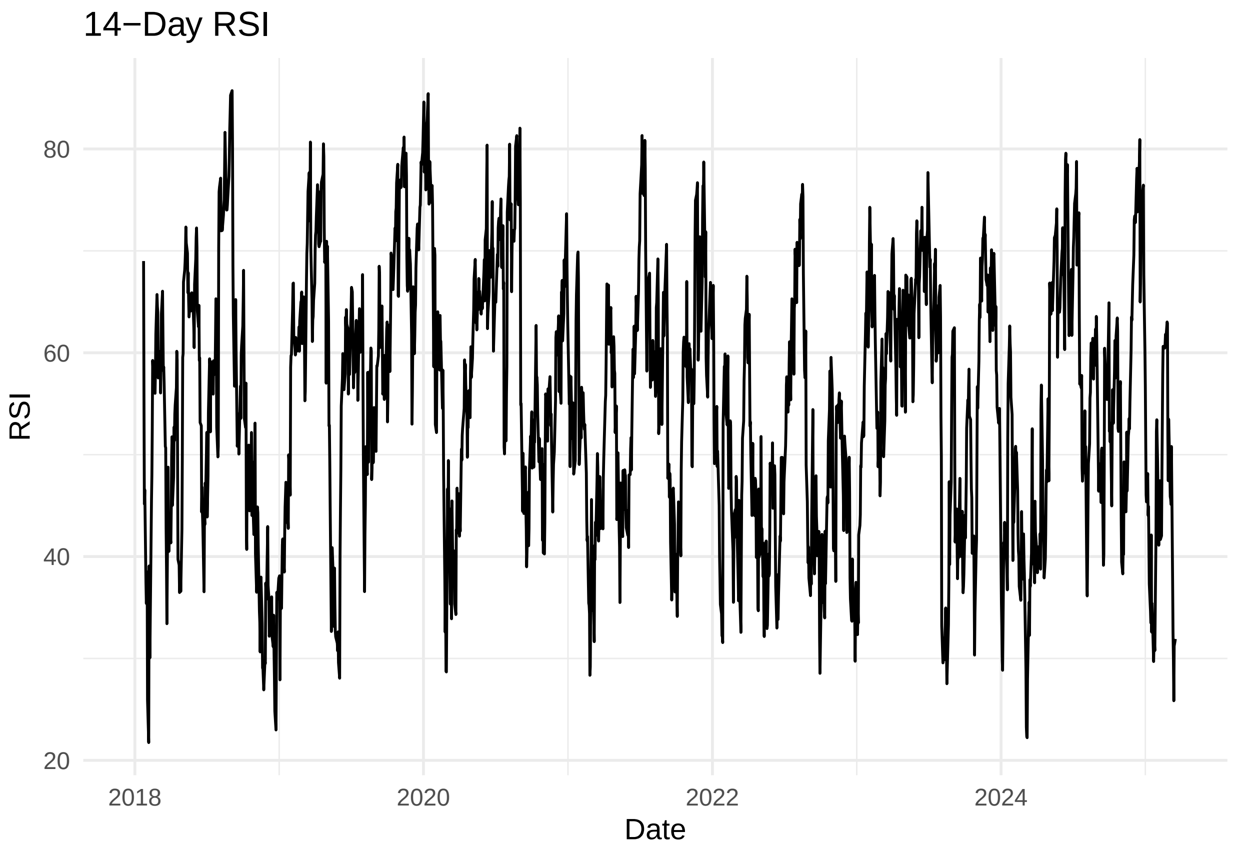
Feature Engineering

Add new variables

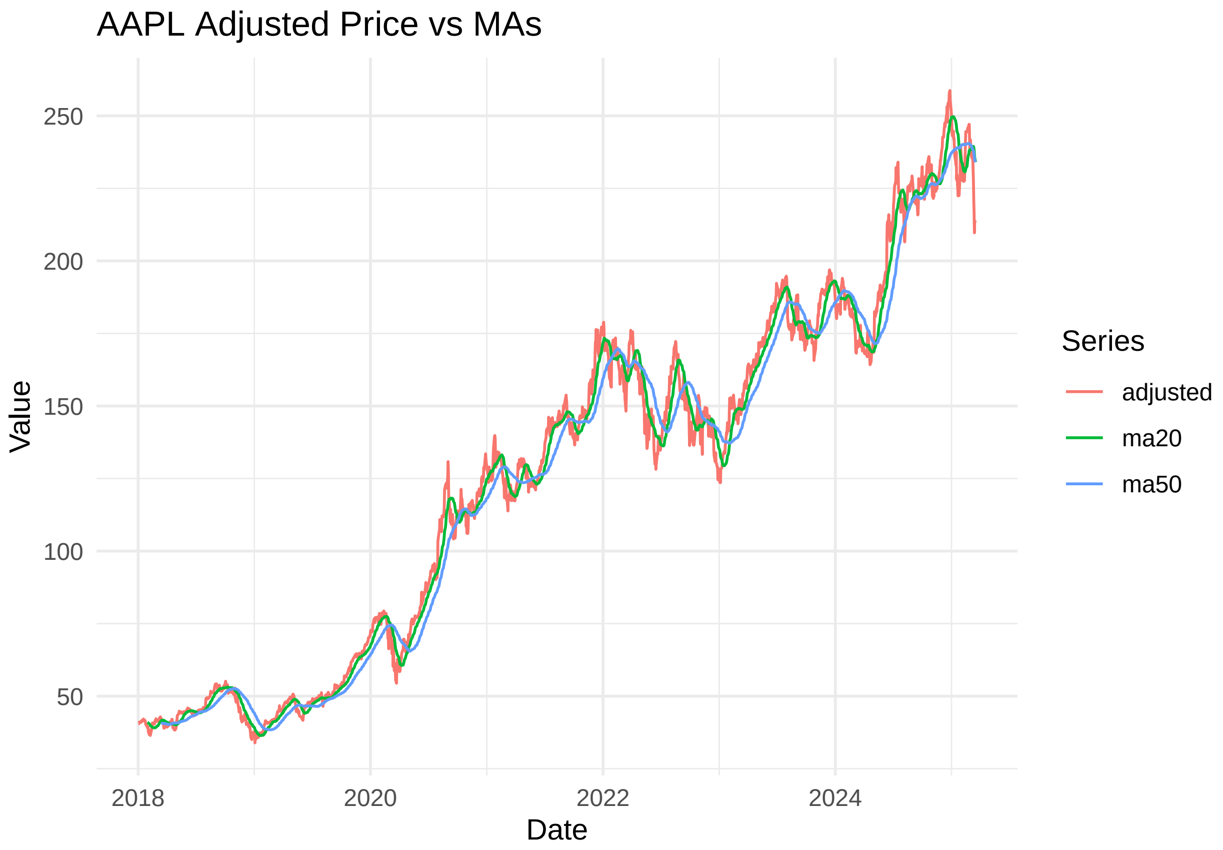
```r
# Calculate Bollinger Bands using high, low, and close
bb <- BBands(HLC = df_stock %>% select(high, low, close),
             n = 20, maType = "SMA", sd = 2)

# Calculate MACD based on the adjusted closing price
macd_values <- MACD(df_stock$adjusted, nFast = 12, nSlow = 26, nSig = 9)

# Add all new variables
df_stock <- df_stock %>%
  arrange(date) %>%
  mutate(
    daily_return = (adjusted - lag(adjusted)) / lag(adjusted) * 100,
    lag1_close = lag(adjusted, 1),
    lag2_close = lag(adjusted, 2),
    ma20 = rollmean(adjusted, k = 20, fill = NA, align = "right"),
    ma50 = rollmean(adjusted, k = 50, fill = NA, align = "right"),
    rsi14 = RSI(adjusted, n = 14),
    bb_dn   = bb[, "dn"],
    bb_mavg = bb[, "mavg"],
    bb_up   = bb[, "up"],
    bb_pctB = bb[, "pctB"],
    macd    = macd_values[, "macd"],
    macdSig = macd_values[, "signal"],
    rolling_sd_20 = rollapply(daily_return, width = 20,
                              FUN = sd, fill = NA, align = "right"),
    wday = wday(date, label = TRUE),
    sin_wday = sin(2 * pi * wday(date) / 7),
    cos_wday = cos(2 * pi * wday(date) / 7)
  )
```

Visualize new variables

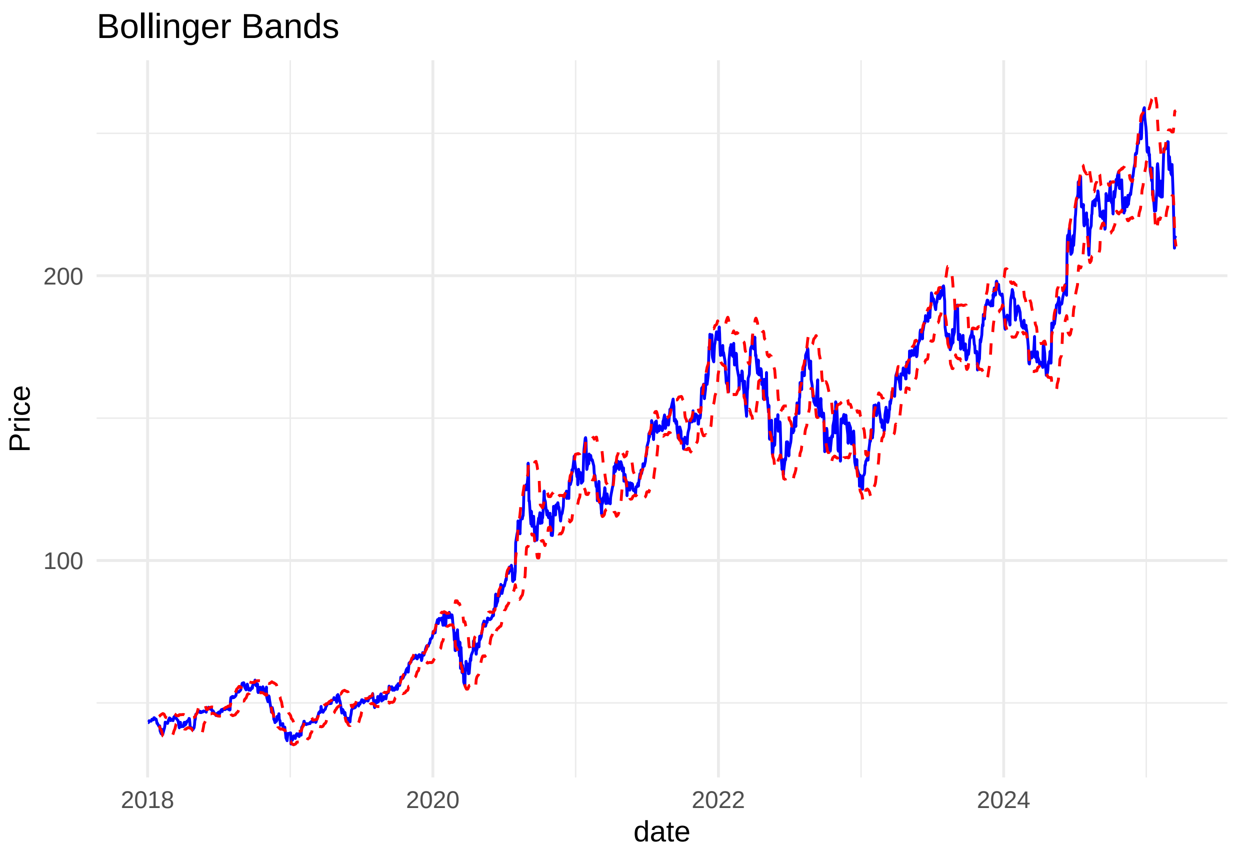
```
# Plot 14-Day RSI
ggplot(df_stock, aes(x = date, y = rsi14)) +
  geom_line() +
  labs(title = "14-Day RSI", x = "Date", y = "RSI") +
  theme_minimal()
```



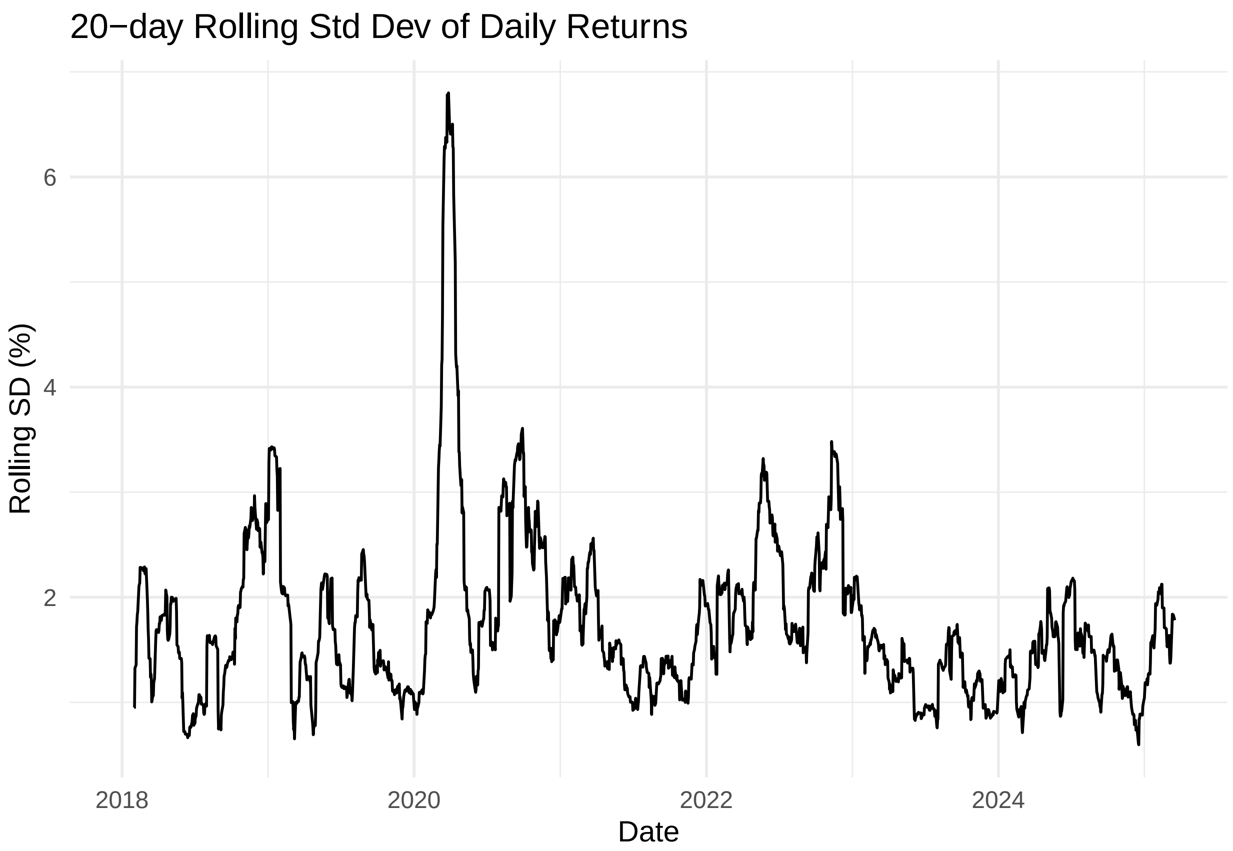
```
# Plot Adjusted Price vs. Moving Averages using dynamic title
df_stock_long <- df_stock %>%
  select(date, adjusted, ma20, ma50) %>%
  pivot_longer(cols = c("adjusted", "ma20", "ma50"),
               names_to = "variable", values_to = "value")

ggplot(df_stock_long, aes(x = date, y = value, color = variable)) +
  geom_line() +
  labs(title = paste(symbol, "Adjusted Price vs MAs"),
       x = "Date", y = "Value", color = "Series") +
  theme_minimal()
```



```
# Plot Bollinger Bands with the closing price
ggplot(df_stock, aes(x = date)) +
  geom_line(aes(y = close), color = "blue") +
  geom_line(aes(y = bb_dn), color = "red", linetype = "dashed") +
  geom_line(aes(y = bb_up), color = "red", linetype = "dashed") +
  labs(title = "Bollinger Bands", y = "Price") +
  theme_minimal()
```



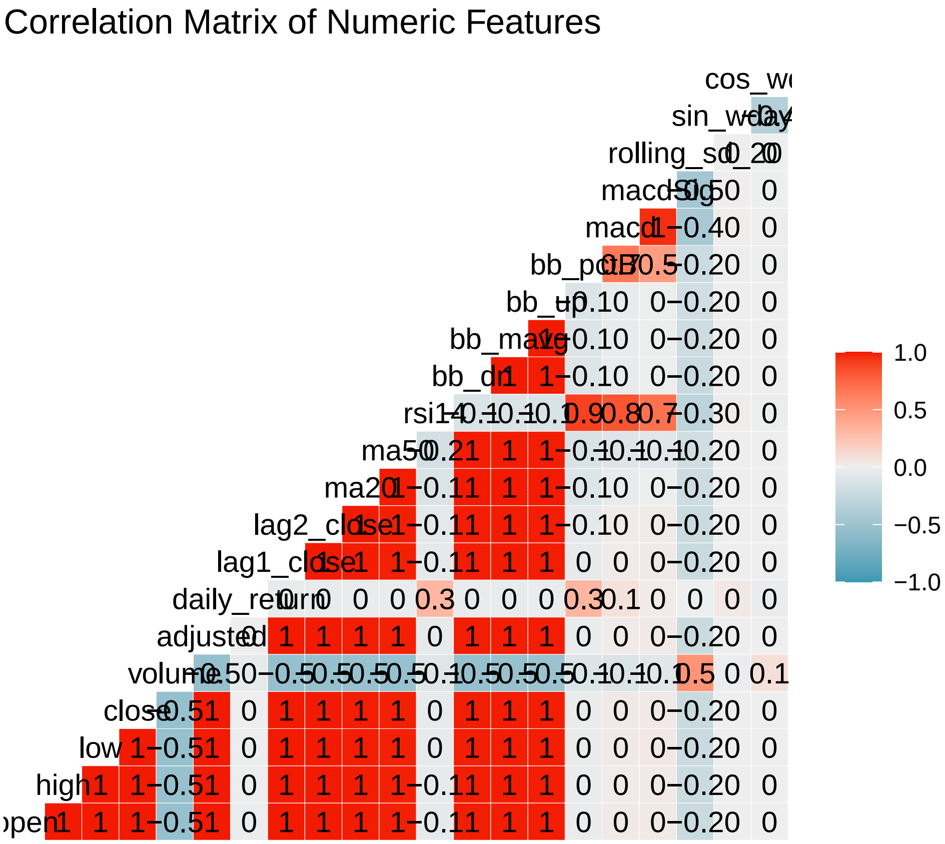
```
# Plot 20-day Rolling Std Dev of Daily Returns
ggplot(df_stock, aes(x = date, y = rolling_sd_20)) +
  geom_line() +
  labs(title = "20-day Rolling Std Dev of Daily Returns",
       x = "Date", y = "Rolling SD (%)") +
  theme_minimal()
```



Other tests and analysis

```
# Correlation Matrix of Numeric Features
df_numerics <- df_stock %>%
  select(where(is.numeric)) %>%
  drop_na()  # Remove rows with NA for accurate correlation

GGally::ggcorr(df_numerics,
               method = c("pairwise.complete.obs", "pearson"),
               label  = TRUE) +
  ggtitle("Correlation Matrix of Numeric Features")
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```


Correlation Matrix of Numeric Features

```r
# Stationarity Tests: Adjusted Price and Daily Returns
adf_result <- adf.test(df_stock$adjusted, alternative = "stationary")
print(adf_result)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  df_stock$adjusted
## Dickey-Fuller = -3.2358, Lag order = 12, p-value = 0.08174
## alternative hypothesis: stationary
```

```r
adf_returns <- adf.test(na.omit(df_stock$daily_return), alternative = "stationary")
```

```
## Warning in adf.test(na.omit(df_stock$daily_return), alternative =
## "stationary"): p-value smaller than printed p-value
```

```r
print(adf_returns)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  na.omit(df_stock$daily_return)
## Dickey-Fuller = -11.491, Lag order = 12, p-value = 0.01
## alternative hypothesis: stationary
```

Model 1: ARIMA with Exogenous Regressors (ARIMAX)