Untitled

Idriss

2024-07-14

Contents

PORTFOLIO FINANCIAL ANALYSIS

In portfolio financial analysis, several models and techniques are commonly used to assess risk, return, and overall performance. Here are some key models:

1. Modern Portfolio Theory (MPT)

- **Developed by:** Harry Markowitz
- **Concept:** Investors can construct portfolios to maximize expected return based on a given level of market risk, emphasizing diversification.
- **Key Metric:** Efficient Frontier, which represents the set of optimal portfolios offering the highest expected return for a defined level of risk.

2. Capital Asset Pricing Model (CAPM)

- **Concept:** Describes the relationship between systematic risk and expected return for assets, particularly stocks.
- **Formula:**

$$E(R_i) = R_f + \beta_i(E(R_m) - R_f)$$

  Where $E(R_i)$ is the expected return on the capital asset, R_f is the risk-free rate, β_i is the beta of the security, and $E(R_m)$ is the expected market return.

3. Arbitrage Pricing Theory (APT)

- **Concept:** An alternative to CAPM, APT considers multiple factors that might affect an asset's return, instead of just market risk.
- **Formula:**

$$E(R_i) = R_f + \beta_{i1}F_1 + \beta_{i2}F_2 + ... + \beta_{in}F_n$$

  Where F represents various factors.

4. Fama-French Three-Factor Model

- **Developed by:** Eugene Fama and Kenneth French
- **Concept:** Expands on CAPM by adding size risk (SMB, Small Minus Big) and value risk (HML, High Minus Low) factors.
- **Formula:**

$$E(R_i) = R_f + \beta_{i1}(E(R_m) - R_f) + \beta_{i2}SMB + \beta_{i3}HML$$

5. Black-Litterman Model

- **Developed by:** Fischer Black and Robert Litterman
- **Concept:** Combines investor views with market equilibrium to estimate the expected returns of assets.
- **Use:** Often used in portfolio optimization to create a more stable and intuitive estimation of expected returns.

6. Value at Risk (VaR)

- **Concept:** Measures the potential loss in value of a portfolio over a defined period for a given confidence interval.
- **Usage:** Helps in risk management to understand the maximum potential loss.

7. Monte Carlo Simulation

- **Concept:** Uses repeated random sampling to simulate the performance of a portfolio under various conditions.
- **Usage:** Useful for estimating the impact of risk and uncertainty in financial and investment decisions.

8. Sharpe Ratio

- **Concept:** Measures the performance of an investment compared to a risk-free asset, after adjusting for its risk.
- **Formula:**

$$\text{Sharpe Ratio} = \frac{E(R_p) - R_f}{\sigma_p}$$

  Where $E(R_p)$ is the expected portfolio return, R_f is the risk-free rate, and σ_p is the portfolio standard deviation.

These models and tools are essential for constructing, analyzing, and managing portfolios to achieve desired risk-return profiles.

getting the data

Installing and importing thelibrary

```
#install.packages("quantmod")
library(quantmod)
```

```
## Loading required package: xts
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
## Loading required package: TTR
```

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

importing data from yahoo

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

# Get S&P 500 (SPY) data
getSymbols("SPY", src = "yahoo", from = "2000-01-01", to = Sys.Date())
```

```
## [1] "SPY"
```

```r
spy_data <- SPY

# Get Dow Jones Industrial Average (DJI) data
getSymbols("^DJI", src = "yahoo", from = "2000-01-01", to = Sys.Date())
```

```
## [1] "DJI"
```

```r
dji_data <- DJI
colnames(spy_data)
```

```
## [1] "SPY.Open"     "SPY.High"     "SPY.Low"      "SPY.Close"    "SPY.Volume"
## [6] "SPY.Adjusted"
```

```r
colnames(dji_data)
```

```
## [1] "DJI.Open"     "DJI.High"     "DJI.Low"      "DJI.Close"    "DJI.Volume"
## [6] "DJI.Adjusted"
```

```r
#View(spy_data)
```

Calculating returns

```r
# Calculate daily returns for SPY
spy_returns <- dailyReturn(spy_data[, "SPY.Adjusted"])

# Calculate daily returns for DJI
dji_returns <- dailyReturn(dji_data[, "DJI.Adjusted"])
```

Returns Summary

```r
# Summary statistics for SPY returns
summary(spy_returns)
```

```
##       Index            daily.returns
##  Min.   :2000-01-03   Min.   :-0.1094237
##  1st Qu.:2006-02-22   1st Qu.:-0.0046919
##  Median :2012-04-07   Median : 0.0006732
##  Mean   :2012-04-07   Mean   : 0.0003658
##  3rd Qu.:2018-05-23   3rd Qu.: 0.0060115
##  Max.   :2024-07-12   Max.   : 0.1451975
```

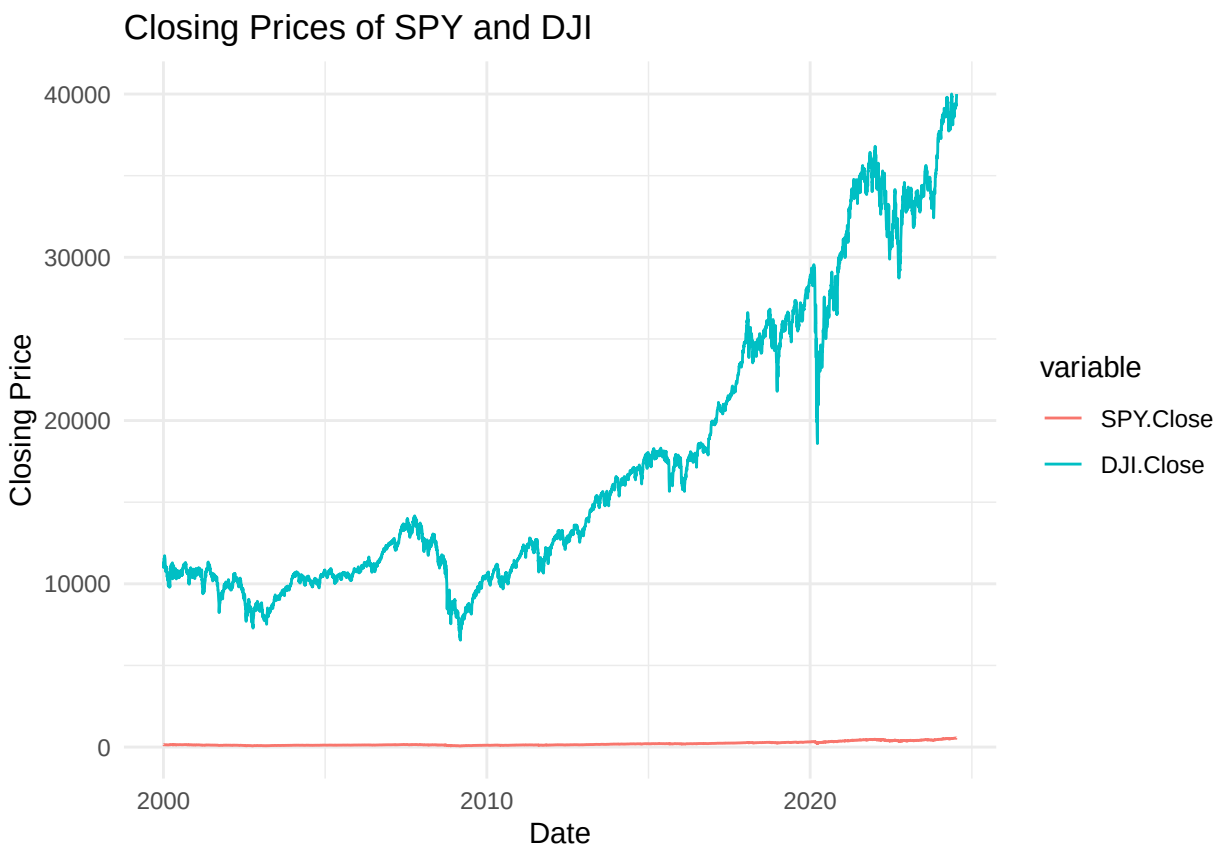
```r
# Summary statistics for DJI returns
summary(dji_returns)
```

```
##       Index             daily.returns
##  Min.   :2000-01-03   Min.   :-0.1292655
##  1st Qu.:2006-02-22   1st Qu.:-0.0045844
##  Median :2012-04-07   Median : 0.0004982
##  Mean   :2012-04-07   Mean   : 0.0002723
##  3rd Qu.:2018-05-23   3rd Qu.: 0.0055937
##  Max.   :2024-07-12   Max.   : 0.1136504
```

Vizualization

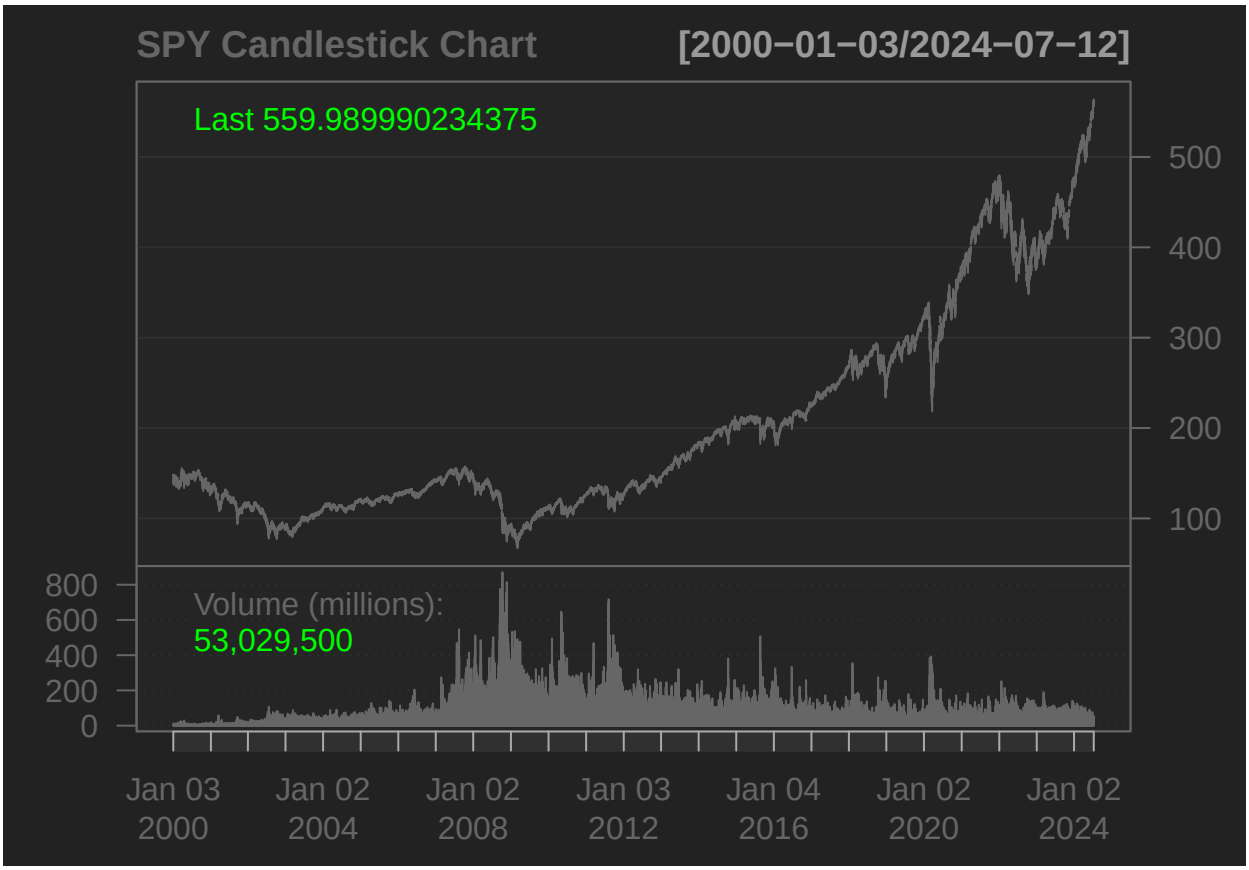
Closing prices

```r
# Load necessary libraries
library(ggplot2)
library(reshape2)

# Combine data into one data frame for plotting
spy_close <- data.frame(Date = index(spy_data), SPY.Close = spy_data$SPY.Close)
dji_close <- data.frame(Date = index(dji_data), DJI.Close = dji_data$DJI.Close)

# Merge data frames by Date
combined_close <- merge(spy_close, dji_close, by = "Date", all = TRUE)

# Melt the data for ggplot2
combined_close_melt <- melt(combined_close, id.vars = "Date")

# Line plot
ggplot(combined_close_melt, aes(x = Date, y = value, color = variable)) +
  geom_line() +
  labs(title = "Closing Prices of SPY and DJI", x = "Date", y = "Closing Price") +
  theme_minimal()
```

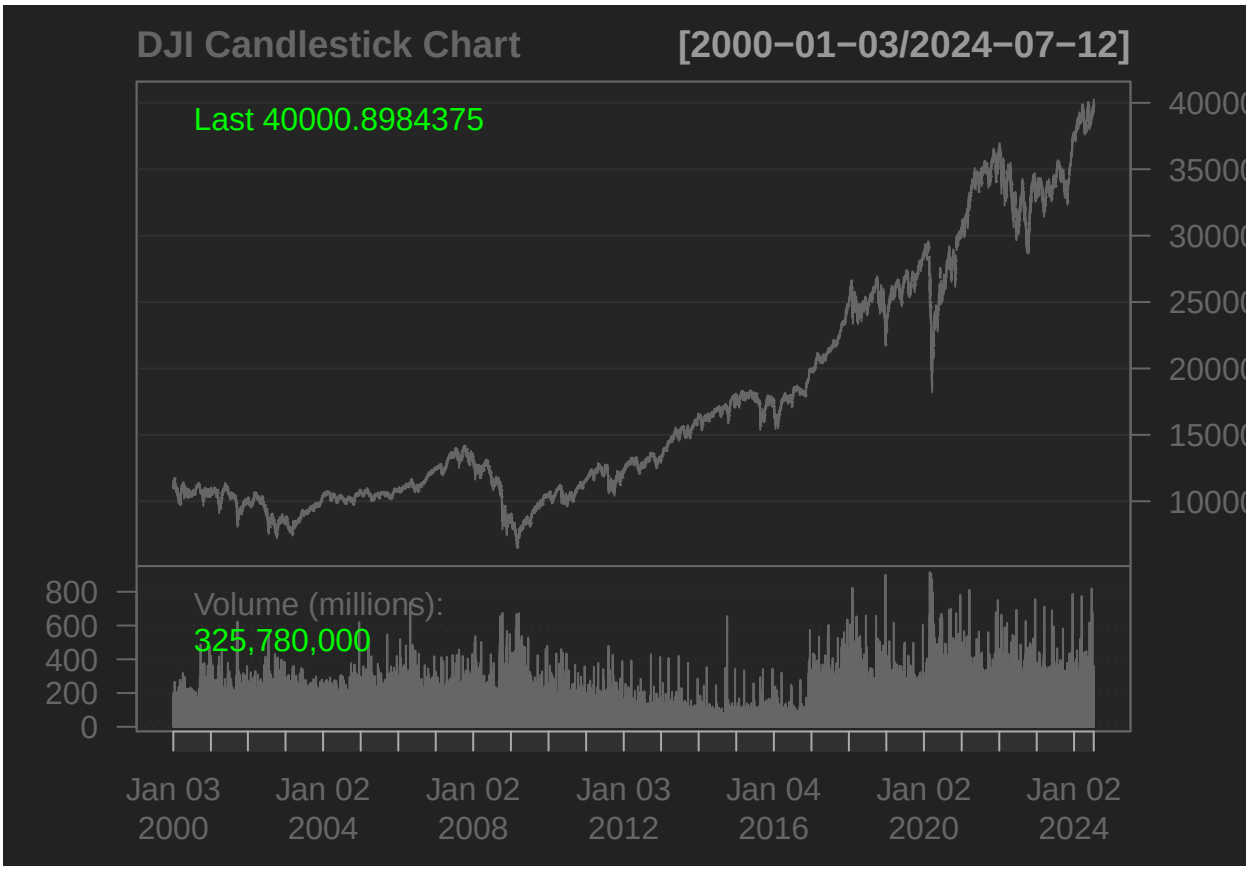


candlestick vizualization

```
# Load necessary library
library(quantmod)

# Candlestick chart for SPY
chartSeries(spy_data, type = "candlesticks", name = "SPY Candlestick Chart")
```



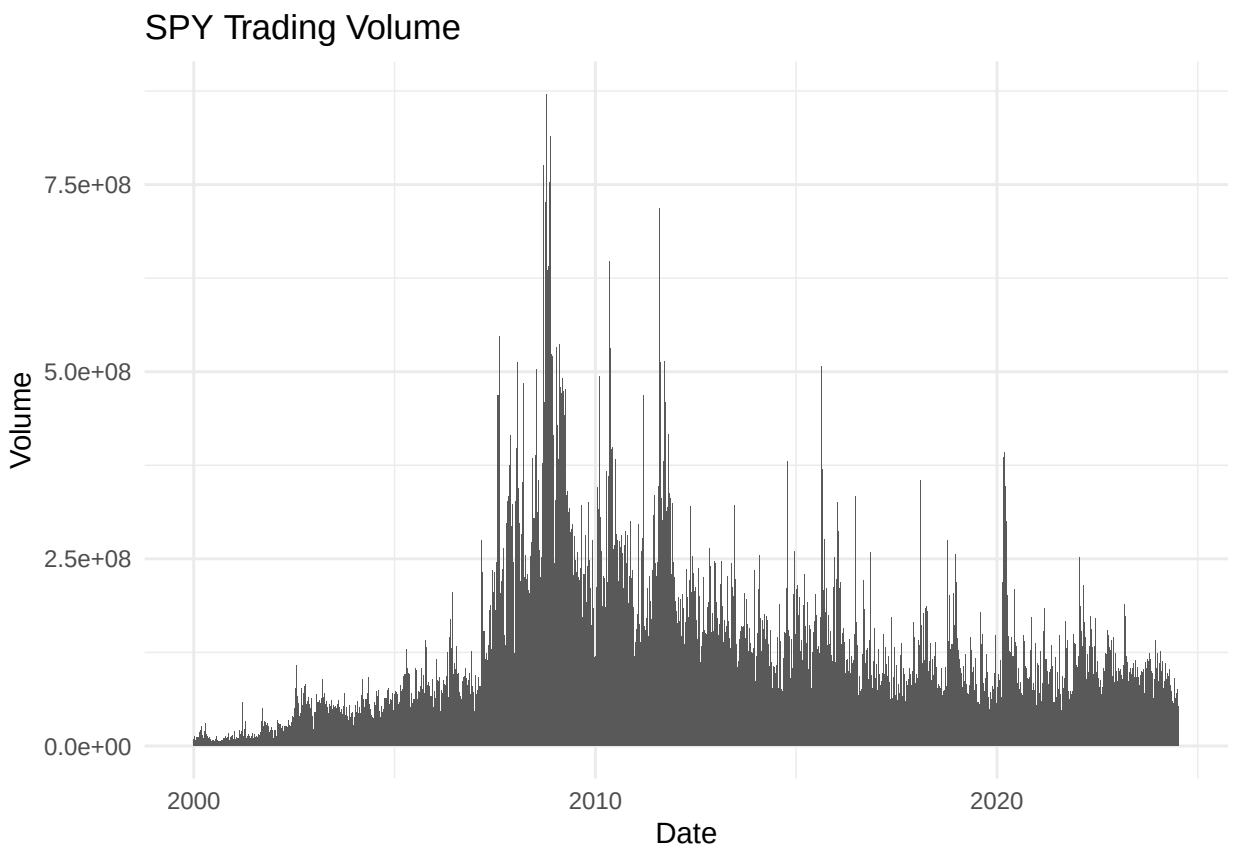

```r
# Candlestick chart for DJI
chartSeries(dji_data, type = "candlesticks", name = "DJI Candlestick Chart")
```



Volume plot

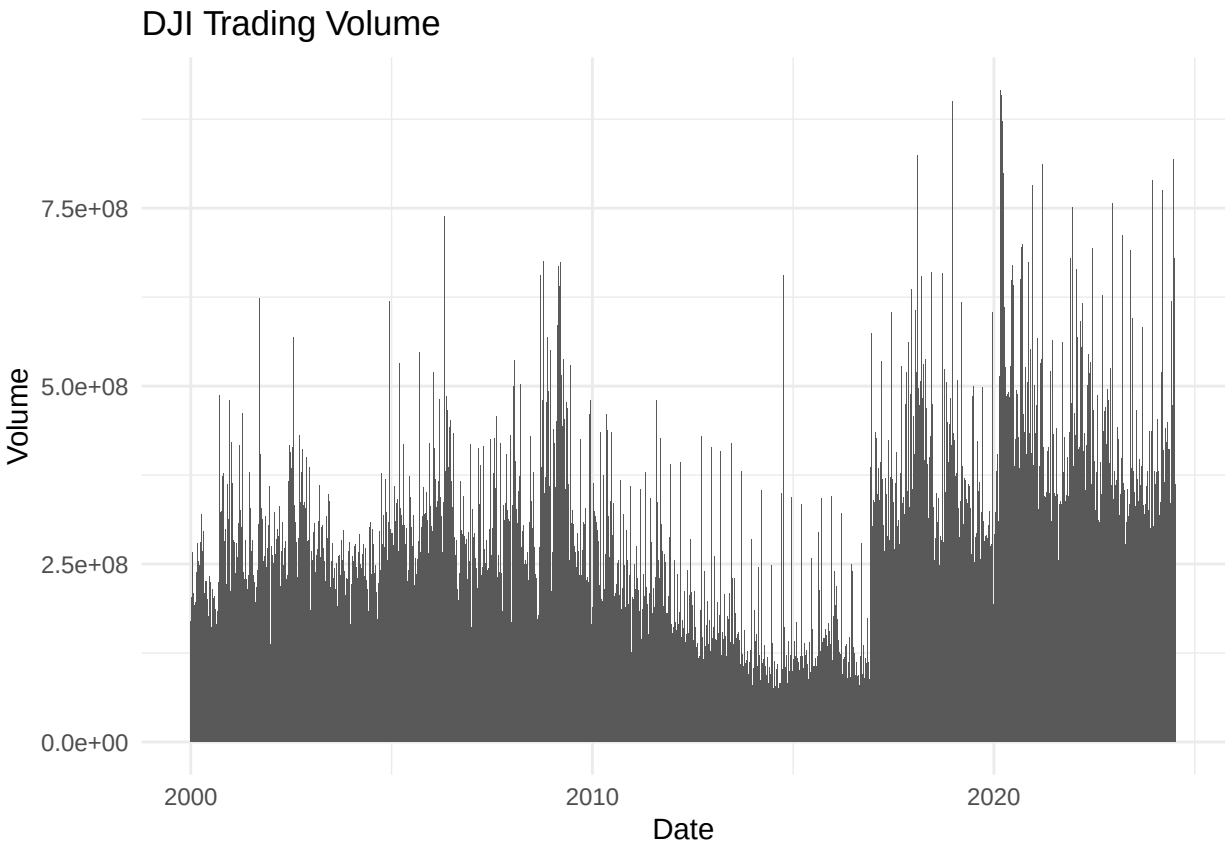
```
# SPY Volume Plot
ggplot(spy_close, aes(x = Date, y = spy_data$SPY.Volume)) +
  geom_bar(stat = "identity") +
  labs(title = "SPY Trading Volume", x = "Date", y = "Volume") +
  theme_minimal()
```

```
## Don't know how to automatically pick scale for object of type <xts/zoo>.
## Defaulting to continuous.
```



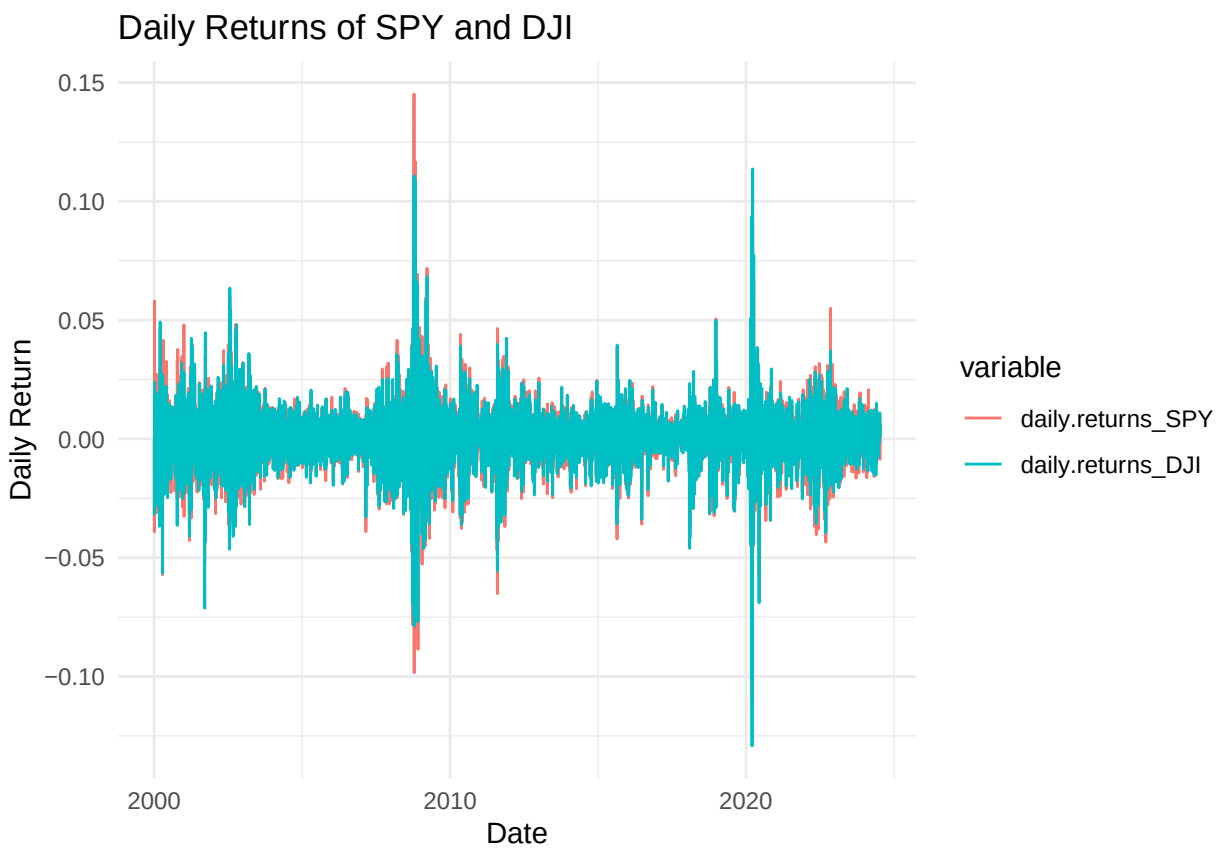

```
# DJI Volume Plot
ggplot(dji_close, aes(x = Date, y = dji_data$DJI.Volume)) +
  geom_bar(stat = "identity") +
  labs(title = "DJI Trading Volume", x = "Date", y = "Volume") +
  theme_minimal()
```

```
## Don't know how to automatically pick scale for object of type <xts/zoo>.
## Defaulting to continuous.
```



Daily returns plot

```
# Calculate daily returns for SPY and DJI
spy_returns <- dailyReturn(spy_data[, "SPY.Adjusted"])
dji_returns <- dailyReturn(dji_data[, "DJI.Adjusted"])

# Convert to data frames
spy_returns_df <- data.frame(Date = index(spy_returns), Return = coredata(spy_returns))
dji_returns_df <- data.frame(Date = index(dji_returns), Return = coredata(dji_returns))

# Combine returns for plotting
combined_returns <- merge(spy_returns_df, dji_returns_df, by = "Date", suffixes = c("_SPY", "_DJI"))
combined_returns_melt <- melt(combined_returns, id.vars = "Date")

# Line plot of daily returns
ggplot(combined_returns_melt, aes(x = Date, y = value, color = variable)) +
  geom_line() +
  labs(title = "Daily Returns of SPY and DJI", x = "Date", y = "Daily Return") +
  theme_minimal()
```

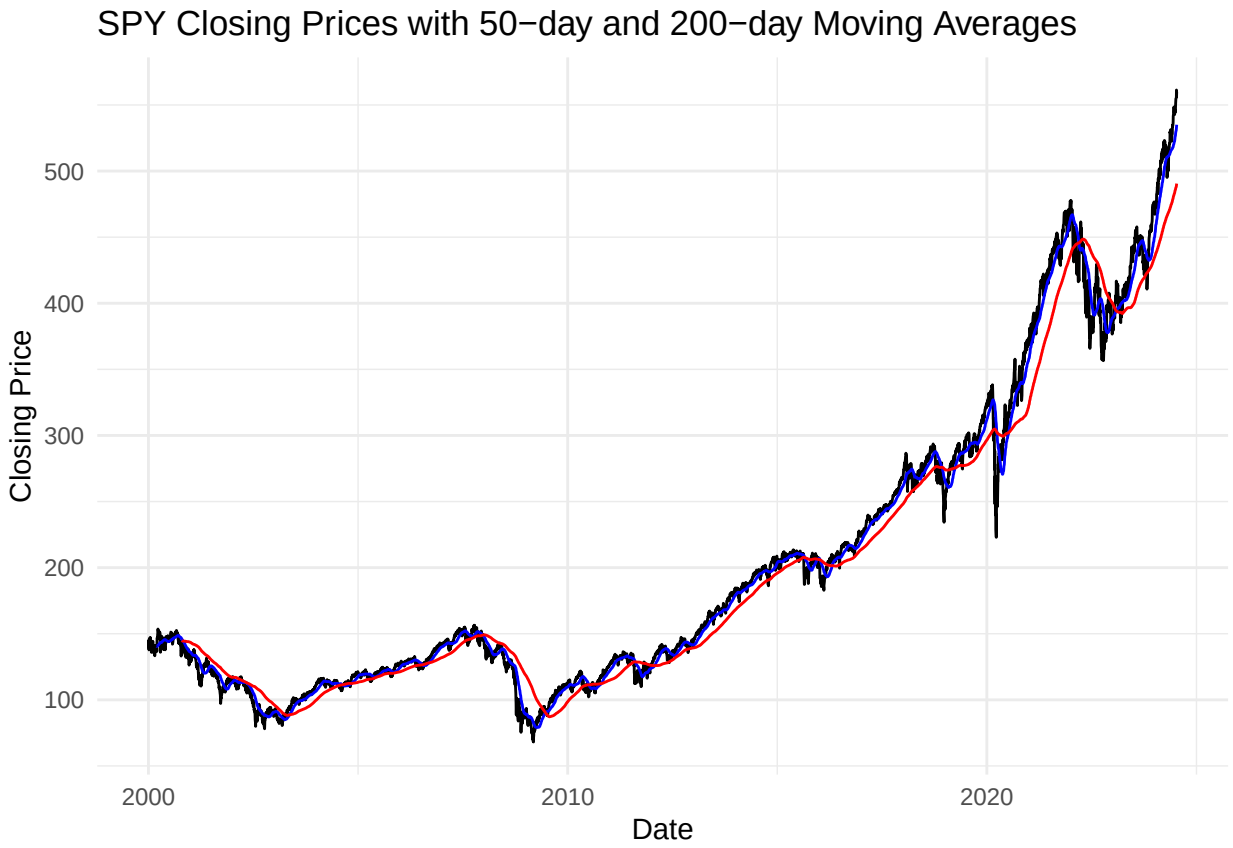


Moving averages plot

```
# Calculate 50-day and 200-day moving averages for SPY
spy_data$SPY.MA50 <- SMA(spy_data[, "SPY.Close"], n = 50)
spy_data$SPY.MA200 <- SMA(spy_data[, "SPY.Close"], n = 200)

# Plot SPY with moving averages
ggplot(spy_close, aes(x = Date, y = SPY.Close)) +
  geom_line() +
  geom_line(aes(y = spy_data$SPY.MA50), color = "blue") +
  geom_line(aes(y = spy_data$SPY.MA200), color = "red") +
  labs(title = "SPY Closing Prices with 50-day and 200-day Moving Averages", x = "Date", y = "Closing Pr
  theme_minimal()
```

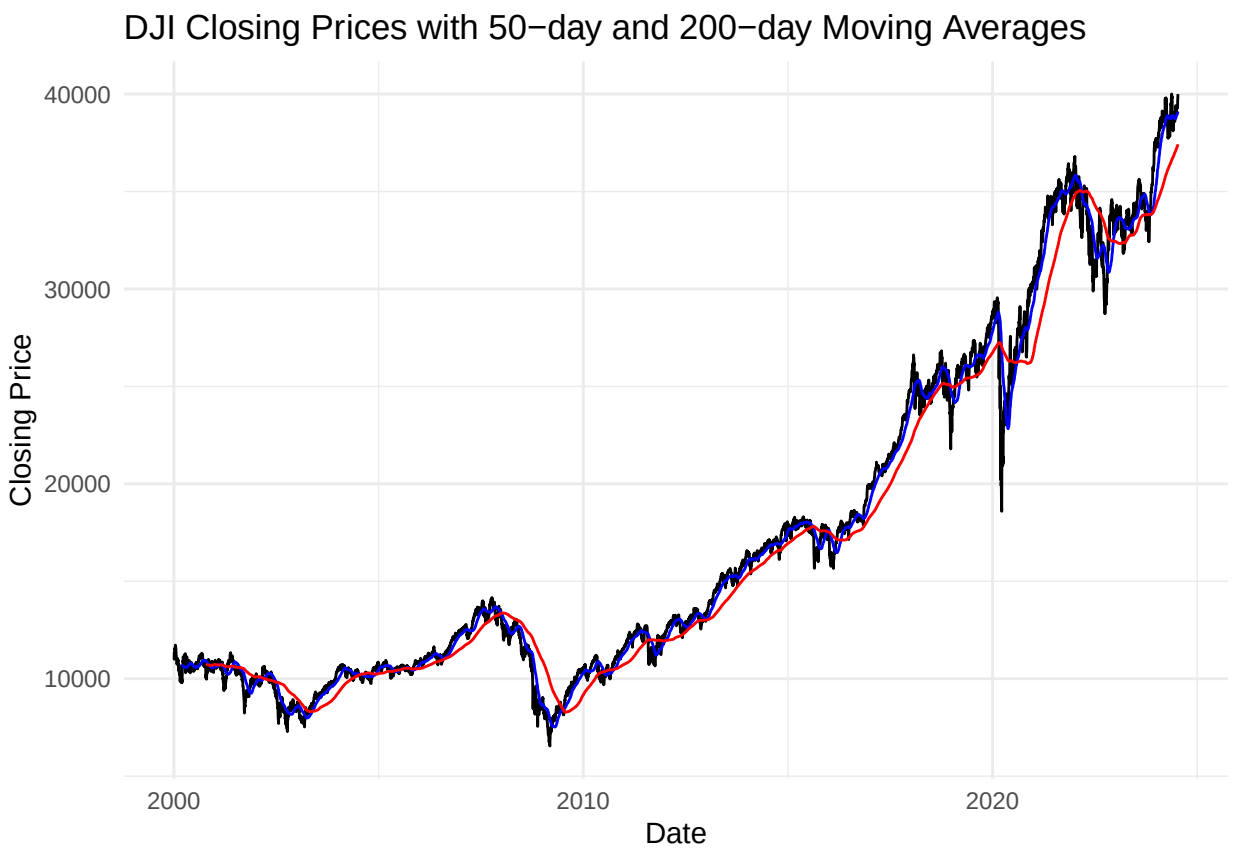
```
## Warning: Removed 49 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

```
## Warning: Removed 199 rows containing missing values or values outside the scale range
## (`geom_line()`).
```



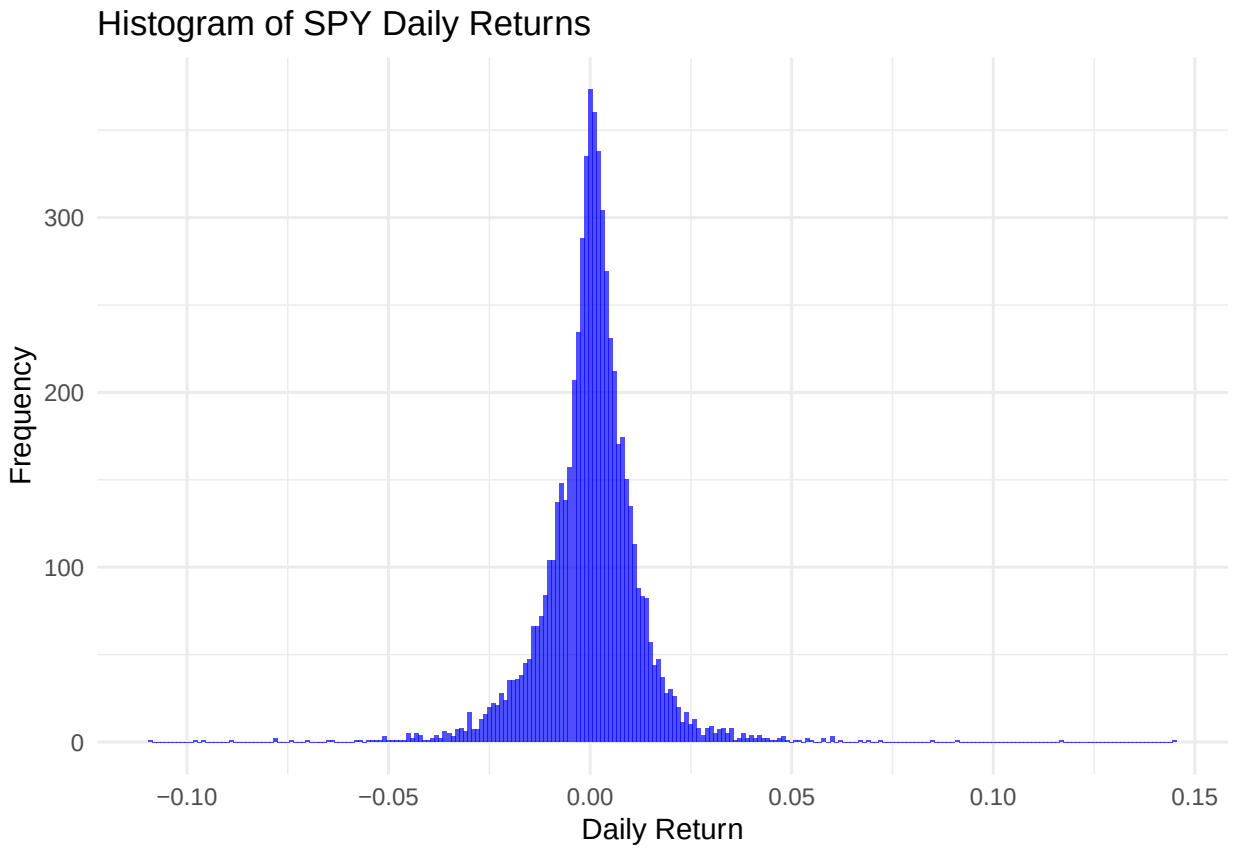

```
# Calculate 50-day and 200-day moving averages for DJI
dji_data$DJI.MA50 <- SMA(dji_data[, "DJI.Close"], n = 50)
dji_data$DJI.MA200 <- SMA(dji_data[, "DJI.Close"], n = 200)

# Plot DJI with moving averages
ggplot(dji_close, aes(x = Date, y = DJI.Close)) +
  geom_line() +
  geom_line(aes(y = dji_data$DJI.MA50), color = "blue") +
  geom_line(aes(y = dji_data$DJI.MA200), color = "red") +
  labs(title = "DJI Closing Prices with 50-day and 200-day Moving Averages", x = "Date", y = "Closing Pr
  theme_minimal()
```

```
## Warning: Removed 49 rows containing missing values or values outside the scale range
## (`geom_line()`).
## Removed 199 rows containing missing values or values outside the scale range
## (`geom_line()`).
```

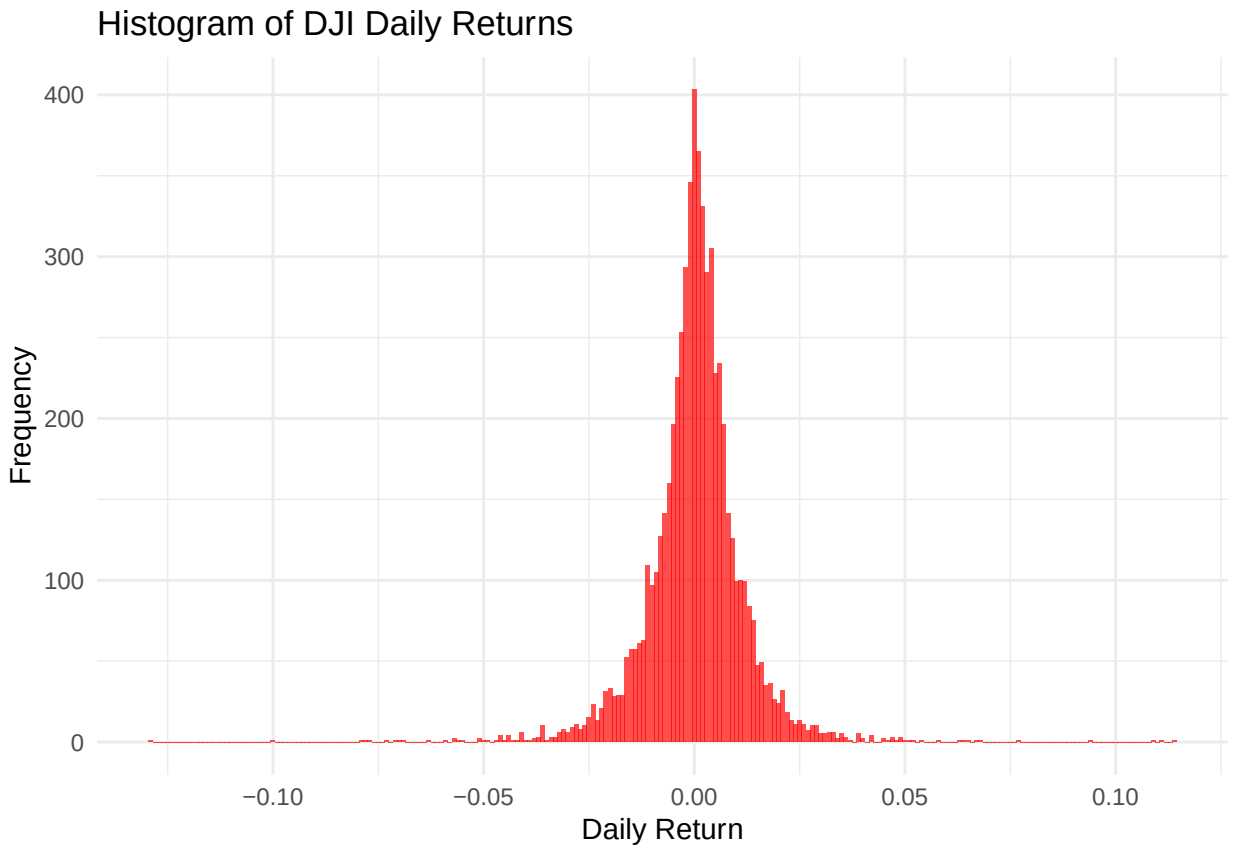


Returns Histogram

```
# Histogram of SPY returns
ggplot(spy_returns_df, aes(x = daily.returns)) +
  geom_histogram(binwidth = 0.001, fill = "blue", alpha = 0.7) +
  labs(title = "Histogram of SPY Daily Returns", x = "Daily Return", y = "Frequency") +
  theme_minimal()
```




```
# Histogram of DJI returns
ggplot(dji_returns_df, aes(x = daily.returns)) +
  geom_histogram(binwidth = 0.001, fill = "red", alpha = 0.7) +
  labs(title = "Histogram of DJI Daily Returns", x = "Daily Return", y = "Frequency") +
  theme_minimal()
```



Summary Statistics

```
# Summary statistics for SPY data
print("Summary statistics for SPY data:")
```

```
## [1] "Summary statistics for SPY data:"
```

```
summary(spy_data)
```

```
##       Index               SPY.Open        SPY.High        SPY.Low
##  Min.   :2000-01-03   Min.   : 67.95   Min.   : 70.0   Min.   : 67.1
##  1st Qu.:2006-02-22   1st Qu.:119.70   1st Qu.:120.5   1st Qu.:119.1
##  Median :2012-04-07   Median :147.48   Median :148.4   Median :146.4
##  Mean   :2012-04-07   Mean   :204.11   Mean   :205.3   Mean   :202.8
##  3rd Qu.:2018-05-23   3rd Qu.:268.08   3rd Qu.:270.2   3rd Qu.:266.8
##  Max.   :2024-07-12   Max.   :561.44   Max.   :563.7   Max.   :557.1
##
##    SPY.Close        SPY.Volume         SPY.Adjusted       SPY.MA50
##  Min.   : 68.11   Min.   :  1436600   Min.   : 50.86   Min.   : 79.1
##  1st Qu.:119.84   1st Qu.: 51146925   1st Qu.: 85.13   1st Qu.:119.6
##  Median :147.42   Median : 80209450   Median :110.01   Median :147.2
##  Mean   :204.13   Mean   :107186836   Mean   :175.60   Mean   :203.0
##  3rd Qu.:268.35   3rd Qu.:135968775   3rd Qu.:243.14   3rd Qu.:268.8
```

```
##  Max.   :561.32   Max.   :871026300   Max.   :561.32   Max.   :535.0
##                                                        NA's   :49
##    SPY.MA200
##  Min.   : 87.21
##  1st Qu.:118.94
##  Median :145.31
##  Mean   :200.00
##  3rd Qu.:269.90
##  Max.   :490.52
##  NA's   :199
```

```
# Summary statistics for DJI data
print("Summary statistics for DJI data:")
```

```
## [1] "Summary statistics for DJI data:"
```

```
summary(dji_data)
```

```
##      Index               DJI.Open        DJI.High        DJI.Low
##  Min.   :2000-01-03   Min.   : 6547   Min.   : 6710   Min.   : 6470
##  1st Qu.:2006-02-22   1st Qu.:10588   1st Qu.:10641   1st Qu.:10522
##  Median :2012-04-07   Median :13239   Median :13303   Median :13168
##  Mean   :2012-04-07   Mean   :17495   Mean   :17596   Mean   :17389
##  3rd Qu.:2018-05-23   3rd Qu.:24463   3rd Qu.:24603   3rd Qu.:24247
##  Max.   :2024-07-12   Max.   :39990   Max.   :40257   Max.   :39865
##
##    DJI.Close       DJI.Volume         DJI.Adjusted      DJI.MA50
##  Min.   : 6547   Min.   :  8410000   Min.   : 6547   Min.   : 7522
##  1st Qu.:10588   1st Qu.:163295000   1st Qu.:10588   1st Qu.:10570
##  Median :13240   Median :236575000   Median :13240   Median :13245
##  Mean   :17498   Mean   :244573917   Mean   :17498   Mean   :17437
##  3rd Qu.:24414   3rd Qu.:307667500   3rd Qu.:24414   3rd Qu.:24478
##  Max.   :40004   Max.   :915990000   Max.   :40004   Max.   :39106
##                                                      NA's   :49
##    DJI.MA200
##  Min.   : 8280
##  1st Qu.:10529
##  Median :13126
##  Mean   :17260
##  3rd Qu.:24625
##  Max.   :37424
##  NA's   :199
```

```
# Calculate daily returns for SPY
spy_returns <- dailyReturn(spy_data[, "SPY.Adjusted"])
# Convert to data frame
spy_returns_df <- data.frame(Date = index(spy_returns), Return = coredata(spy_returns))

# Summary statistics for SPY returns
print("Summary statistics for SPY returns:")
```

```
## [1] "Summary statistics for SPY returns:"
```

```r
summary(spy_returns_df)
```

```
##       Date             daily.returns
##  Min.   :2000-01-03   Min.   :-0.1094237
##  1st Qu.:2006-02-22   1st Qu.:-0.0046919
##  Median :2012-04-07   Median : 0.0006732
##  Mean   :2012-04-07   Mean   : 0.0003658
##  3rd Qu.:2018-05-23   3rd Qu.: 0.0060115
##  Max.   :2024-07-12   Max.   : 0.1451975
```

```r
# Calculate daily returns for DJI
dji_returns <- dailyReturn(dji_data[, "DJI.Adjusted"])
# Convert to data frame
dji_returns_df <- data.frame(Date = index(dji_returns), Return = coredata(dji_returns))

# Summary statistics for DJI returns
print("Summary statistics for DJI returns:")
```

```
## [1] "Summary statistics for DJI returns:"
```

```r
summary(dji_returns_df)
```

```
##       Date             daily.returns
##  Min.   :2000-01-03   Min.   :-0.1292655
##  1st Qu.:2006-02-22   1st Qu.:-0.0045844
##  Median :2012-04-07   Median : 0.0004982
##  Mean   :2012-04-07   Mean   : 0.0002723
##  3rd Qu.:2018-05-23   3rd Qu.: 0.0055937
##  Max.   :2024-07-12   Max.   : 0.1136504
```

MODELs

Modern Portfolio Theory (MPT) Explanation

Modern Portfolio Theory (MPT) is a framework for constructing a portfolio of assets that maximizes expected return for a given level of risk. It was introduced by Harry Markowitz in his paper "Portfolio Selection" published in 1952. The key concept behind MPT is diversification, which helps reduce the overall risk of the portfolio.

Mathematical Explanation

1. **Expected Return of a Portfolio ($E(R_p)$):**

$$E(R_p) = \sum_{i=1}^{n} w_i E(R_i)$$

   Where:

   - $E(R_p)$ is the expected return of the portfolio.
   - w_i is the weight of asset i in the portfolio.

- $E(R_i)$ is the expected return of asset i.

2. **Variance of a Portfolio (σ_p^2):**

$$\sigma_p^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} w_i w_j \sigma_{ij}$$

   Where:

   - σ_p^2 is the variance of the portfolio.
   - σ_{ij} is the covariance between the returns of assets i and j.

3. **Efficient Frontier:** The efficient frontier is the set of portfolios that offers the highest expected return for a defined level of risk (or the lowest risk for a given level of expected return).

Steps to Perform MPT in R

To perform MPT on your data frames and visualize the efficient frontier, we can use the `PortfolioAnalytics` package. Below are the steps:

```
#install.packages("quantmod")
#install.packages("PortfolioAnalytics")
#install.packages("PerformanceAnalytics")
#install.packages("ROI")
#install.packages("ROI.plugin.quadprog")
#install.packages("ROI.plugin.glpk")
library(quantmod)
library(PortfolioAnalytics)
```

1. Install and Load Necessary Packages

```
## Loading required package: foreach

## Loading required package: PerformanceAnalytics

##
## Attaching package: 'PerformanceAnalytics'

## The following object is masked from 'package:graphics':
##
##     legend

## Registered S3 method overwritten by 'PortfolioAnalytics':
##   method           from
##   print.constraint ROI
```

```
library(PerformanceAnalytics)
library(ROI)
```

```
## ROI: R Optimization Infrastructure
```

```
## Registered solver plugins: nlminb, symphony, glpk, quadprog.

## Default solver: auto.

##
## Attaching package: 'ROI'

## The following objects are masked from 'package:PortfolioAnalytics':
##
##     is.constraint, objective
```

```
library(ROI.plugin.quadprog)
library(ROI.plugin.glpk)
```

```
# Load data for SPY and DJI
getSymbols("SPY", src = "yahoo", from = "2000-01-01", to = Sys.Date())
```

2. Get Data and Calculate Returns

```
## [1] "SPY"
```

```
getSymbols("^DJI", src = "yahoo", from = "2000-01-01", to = Sys.Date())
```

```
## [1] "DJI"
```

```
# Calculate daily returns
spy_returns <- dailyReturn(Ad(SPY))
dji_returns <- dailyReturn(Ad(DJI))

# Combine returns into one data frame
returns_data <- na.omit(merge(spy_returns, dji_returns))
colnames(returns_data) <- c("SPY", "DJI")
```

```
# Define portfolio
portfolio <- portfolio.spec(assets = colnames(returns_data))

# Add constraints
portfolio <- add.constraint(portfolio, type = "full_investment")
portfolio <- add.constraint(portfolio, type = "box", min = 0.01, max = 0.99)

# Add objectives
portfolio <- add.objective(portfolio, type = "return", name = "mean")
portfolio <- add.objective(portfolio, type = "risk", name = "StdDev")
```

3. Define Portfolio and Constraints

```r
# Optimize portfolio with tracing enabled
opt_portfolio <- optimize.portfolio(returns_data, portfolio, optimize_method = "ROI", trace = TRUE)
```

4. Optimize Portfolio

```r
# Check optimization results
print(opt_portfolio)
```

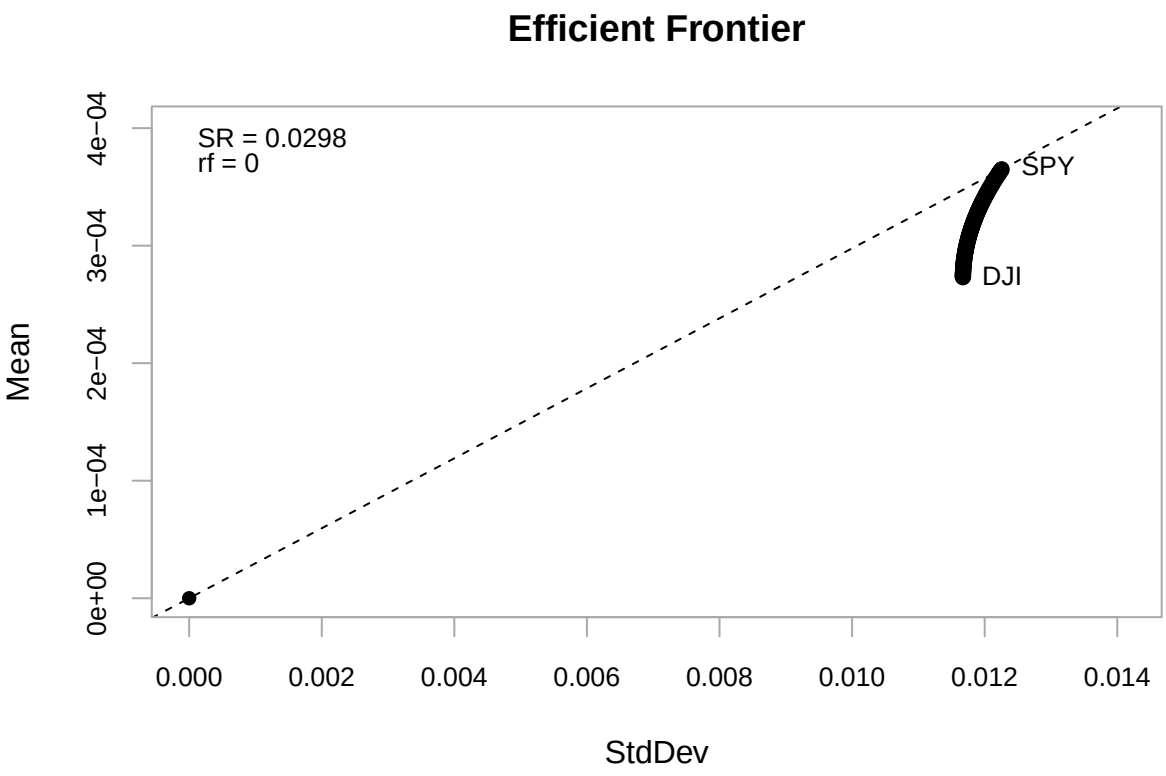
5. Plot Efficient Frontier

```
## ***********************************
## PortfolioAnalytics Optimization
## ***********************************
##
## Call:
## optimize.portfolio(R = returns_data, portfolio = portfolio, optimize_method = "ROI",
##     trace = TRUE)
##
## Optimal Weights:
##  SPY  DJI
## 0.99 0.01
##
## Objective Measure:
##      mean
## 0.0003649
##
##
##  StdDev
## 0.01225
```

```r
# Efficient frontier
ef <- extractEfficientFrontier(opt_portfolio, match.col = "StdDev", n.portfolios = 100)
```

```
## Warning: executing %dopar% sequentially: no parallel backend registered
```

```r
# Plot efficient frontier
chart.EfficientFrontier(ef, match.col = "StdDev", main = "Efficient Frontier")
```



Interpretation of Results

1. **Expected Return ($E(R_p)$):** This is the weighted average of the expected returns of the assets in the portfolio. It shows the return you can expect from the portfolio based on historical data.
2. **Portfolio Variance (σ_p^2):** This measures the risk of the portfolio. It considers not just the individual volatilities of the assets but also how they move together (covariance).
3. **Efficient Frontier:** The efficient frontier plot shows the set of optimal portfolios. Portfolios on the efficient frontier offer the highest expected return for a given level of risk. Portfolios below the frontier are sub-optimal because they do not provide enough return for their level of risk.

Capital Asset Pricing Model (CAPM) Explanation and Implementation

CAPM is a model that describes the relationship between the risk of an asset and its expected return. The model is used to estimate an asset's expected return based on its risk compared to the market.

Mathematical Explanation

The CAPM formula is given by:

$$E(R_i) = R_f + \beta_i(E(R_m) - R_f)$$

Where: - $E(R_i)$ = Expected return of the asset i - R_f = Risk-free rate - β_i = Beta of the asset i, which measures the asset's sensitivity to market returns - $E(R_m)$ = Expected return of the market

Beta (β_i) is calculated as:

$$\beta_i = \frac{\text{Cov}(R_i, R_m)}{\text{Var}(R_m)}$$

Where: - $\text{Cov}(R_i, R_m)$ = Covariance between the returns of the asset i and the market - $\text{Var}(R_m)$ = Variance of the market returns

Steps to Implement CAPM in R

1. **Calculate Daily Returns**
2. **Obtain Risk-Free Rate**
3. **Calculate Beta (β_i)**
4. **Estimate Expected Returns Using CAPM**

Interpretation of Results:

- **Expected Return for SPY:** The estimated return based on its sensitivity (beta) to the DJI index.
- **Expected Return for DJI:** Based on the CAPM, the DJI index itself should match the market return, and its beta is assumed to be 1.

This script calculates and displays the expected returns for SPY and DJI based on the CAPM formula. Adjust the risk-free rate and market return as needed for more accurate and current estimates.

```
# Install and load necessary packages
library(quantmod)
library(PerformanceAnalytics)
library(ggplot2)

# Load data for SPY and DJI
getSymbols("SPY", src = "yahoo", from = "2000-01-01", to = Sys.Date())
```

```
## [1] "SPY"
```

```
getSymbols("^DJI", src = "yahoo", from = "2000-01-01", to = Sys.Date())
```

```
## [1] "DJI"
```

```
# Calculate daily returns
spy_returns <- dailyReturn(Ad(SPY))
dji_returns <- dailyReturn(Ad(DJI))

# Combine returns into one data frame
returns_data <- na.omit(merge(spy_returns, dji_returns))
colnames(returns_data) <- c("SPY", "DJI")

# Risk-free rate (annualized)
R_f_annual <- 0.02
# Convert to daily risk-free rate
R_f <- (1 + R_f_annual)^(1/252) - 1

# Calculate beta for SPY relative to DJI
```

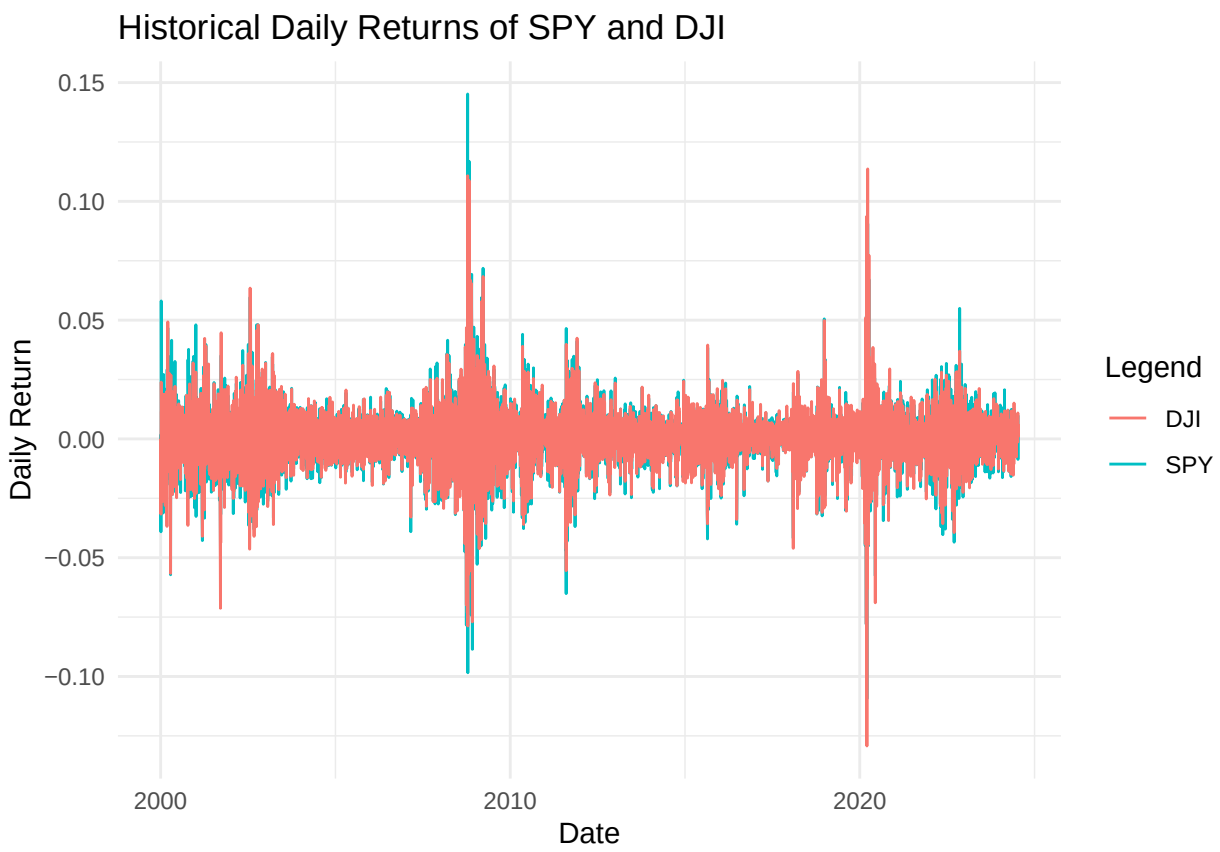
```r
covariance <- cov(returns_data$SPY, returns_data$DJI)
variance_dji <- var(returns_data$DJI)
beta_spy <- covariance / variance_dji

# Calculate beta for DJI relative to itself (which should be 1)
beta_dji <- 1

# Calculate expected returns using CAPM
expected_return_spy <- R_f + beta_spy * (mean(returns_data$DJI) - R_f)
expected_return_dji <- R_f + beta_dji * (mean(returns_data$DJI) - R_f)

# Display the results
cat("Expected Return for SPY:", expected_return_spy, "\n")
```

```
## Expected Return for SPY: 0.0002724756
```

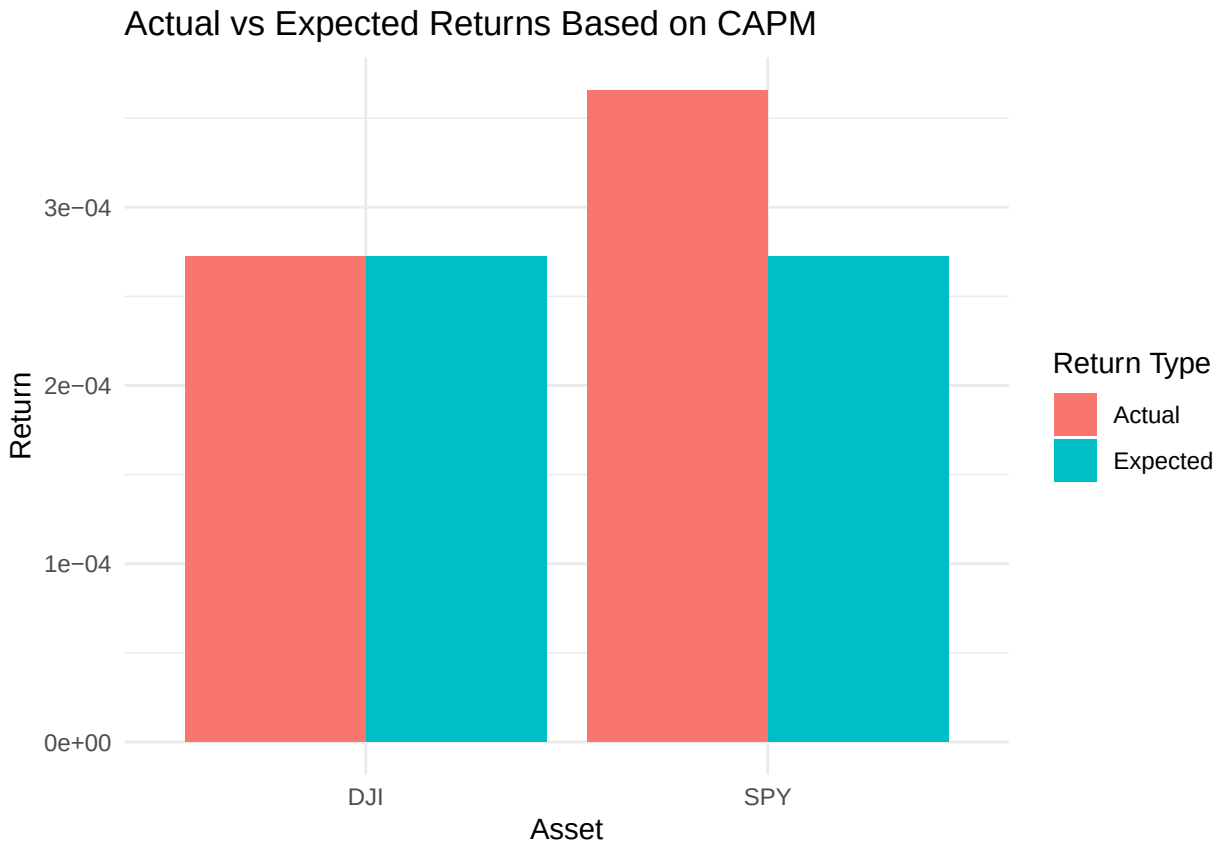
```r
cat("Expected Return for DJI:", expected_return_dji, "\n")
```

```
## Expected Return for DJI: 0.0002722884
```

```r
# Prepare data for plotting

# Plot 1: Historical Returns of SPY and DJI
ggplot(data = returns_data, aes(x = index(returns_data))) +
  geom_line(aes(y = SPY, color = "SPY")) +
  geom_line(aes(y = DJI, color = "DJI")) +
  labs(title = "Historical Daily Returns of SPY and DJI",
       x = "Date",
       y = "Daily Return",
       color = "Legend") +
  theme_minimal()
```




```r
# Plot 2: CAPM Expected Returns vs Actual Returns
actual_mean_returns <- colMeans(returns_data)
expected_returns <- c(SPY = expected_return_spy, DJI = expected_return_dji)

returns_comparison <- data.frame(
  Asset = c("SPY", "DJI"),
  Actual = actual_mean_returns,
  Expected = expected_returns
)

# Reshape data for ggplot
returns_comparison_melted <- reshape2::melt(returns_comparison, id.vars = "Asset")

ggplot(returns_comparison_melted, aes(x = Asset, y = value, fill = variable)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Actual vs Expected Returns Based on CAPM",
       x = "Asset",
       y = "Return",
       fill = "Return Type") +
  theme_minimal()
```




```r
# Plot 3: Beta of SPY Relative to DJI
beta_values <- data.frame(
  Asset = c("SPY"),
  Beta = beta_spy
)
```