FINANCE MANAGEMENT INTERNAL ASSESSMENT 1

vardhan

2024-03-30

```r
# Check the length of each column
# Assuming the data is stored in a variable named 'data'
data <- data.frame(
  ID = 1:110,
  Date = c("2022-01-03", "2022-01-04", "2022-01-05", "2022-01-06", "2022-01-07",
           "2022-01-10", "2022-01-11", "2022-01-12", "2022-01-13", "2022-01-14",
           "2022-01-17", "2022-01-18", "2022-01-19", "2022-01-20", "2022-01-21",
           "2022-01-24", "2022-01-25", "2022-01-26", "2022-01-27", "2022-01-28",
           "2022-01-31", "2022-02-01", "2022-02-02", "2022-02-03", "2022-02-04",
           "2022-02-07", "2022-02-08", "2022-02-09", "2022-02-10", "2022-02-11",
           "2022-02-14", "2022-02-15", "2022-02-16", "2022-02-17", "2022-02-18",
           "2022-02-21", "2022-02-22", "2022-02-23", "2022-02-24", "2022-02-25",
           "2022-02-28", "2022-03-01", "2022-03-02", "2022-03-03", "2022-03-04",
           "2022-03-07", "2022-03-08", "2022-03-09", "2022-03-10", "2022-03-11",
           "2022-03-14", "2022-03-15", "2022-03-16", "2022-03-17", "2022-03-18",
           "2022-03-21", "2022-03-22", "2022-03-23", "2022-03-24", "2022-03-25",
           "2022-03-28", "2022-03-29", "2022-03-30", "2022-03-31", "2022-04-01",
           "2022-04-04", "2022-04-05", "2022-04-06", "2022-04-07", "2022-04-08",
           "2022-04-11", "2022-04-12", "2022-04-13", "2022-04-14", "2022-04-15",
           "2022-04-18", "2022-04-19", "2022-04-20", "2022-04-21", "2022-04-22",
           "2022-04-25", "2022-04-26", "2022-04-27", "2022-04-28", "2022-04-29",
           "2022-05-02", "2022-05-03", "2022-05-04", "2022-05-05", "2022-05-06",
           "2022-05-09", "2022-05-10", "2022-05-11", "2022-05-12", "2022-05-13",
           "2022-05-16", "2022-05-17", "2022-05-18", "2022-05-19", "2022-05-20",
           "2022-05-23", "2022-05-24", "2022-05-25", "2022-05-26", "2022-05-27",
           "2022-05-30", "2022-05-31", "2022-06-01", "2022-06-02", "2022-06-03"),
  Value1 = c(133.44, 134.45, 135.55, 136.38, 135.48, 134.8, 135.6, 135.32, 134.95, 134.75,
             135.12, 135.3, 135.89, 135.76, 134.98, 135.06, 134.7, 134.94, 135.38, 134.97,
             134.68, 135.14, 135.43, 135.35, 135.27, 135.2, 135.1, 134.8, 134.62, 134.53,
             134.82, 135.13, 135.16, 135.05, 134.82, 134.69, 134.61, 134.74, 134.95, 135.26,
             135.55, 135.63, 135.41, 135.39, 135.27, 135.08, 134.98, 134.69, 134.62, 134.74,
             134.84, 134.72, 134.53, 134.63, 134.76, 134.95, 135.08, 134.98, 134.94, 134.91,
             134.79, 134.62, 134.54, 134.64, 134.7, 134.56, 134.48, 134.42, 134.36, 134.49,
             134.61, 134.52, 134.68, 134.72, 134.81, 134.78, 134.9, 134.97, 134.86, 134.78,
             134.62, 134.75, 134.85, 134.75, 134.64, 134.49, 134.56, 134.67, 134.56, 134.43,
             134.31, 134.42, 134.3, 134.17, 134.22, 134.31, 134.43, 134.38, 134.49, 134.62,
             134.47, 134.55, 134.63, 134.48, 134.52, 134.59, 134.65, 134.58, 134.51, 134.63),
  Value2 = c(77.59, 78.01, 78.66, 79.22, 79.13, 78.75, 78.99, 78.89, 78.59, 78.49,
             78.39, 78.79, 78.59, 78.89, 78.79, 78.49, 78.29, 78.39, 78.29, 78.49,
             78.59, 78.69, 78.79, 78.69, 78.59, 78.69, 78.59, 78.49, 78.69, 78.79,
             78.99, 78.79, 78.69, 78.49, 78.59, 78.49, 78.29, 78.39, 78.29, 78.19,
             78.09, 77.89, 77.99, 78.09, 77.99, 77.89, 77.99, 78.09, 78.19, 78.09,
             78.19, 78.29, 78.39, 78.49, 78.39, 78.49, 78.39, 78.29, 78.19, 78.29,
             78.29, 78.19, 78.09, 77.99, 78.09, 78.19, 78.09, 78.19, 78.09, 78.19,
             78.29, 78.39, 78.49, 78.39, 78.49, 78.39, 78.49, 78.39, 78.49, 78.59,
             78.69, 78.79, 78.69, 78.59, 78.49, 78.39, 78.49, 78.59, 78.49, 78.39,
             78.29, 78.39, 78.29, 78.19, 78.29, 78.39, 78.29, 78.19, 78.29, 78.39,
             78.49, 78.39, 78.29, 78.19, 78.09, 78.19, 78.09, 78.19, 78.29, 78.19)
)


length_ID <- length(1:111)
length_Date <- length(c("2022-01-03", "2022-01-04", "2022-01-05", "2022-01-06", "2022-01-07",
                        "2022-01-10", "2022-01-11", "2022-01-12", "2022-01-13", "2022-01-14",
                        "2022-01-17", "2022-01-18", "2022-01-19", "2022-01-20", "2022-01-21",
                        "2022-01-24", "2022-01-25", "2022-01-26", "2022-01-27", "2022-01-28",
                        "2022-01-31", "2022-02-01", "2022-02-02", "2022-02-03", "2022-02-04",
                        "2022-02-07", "2022-02-08", "2022-02-09", "2022-02-10", "2022-02-11",
                        "2022-02-14", "2022-02-15", "2022-02-16", "2022-02-17", "2022-02-18",
                        "2022-02-21", "2022-02-22", "2022-02-23", "2022-02-24", "2022-02-25",
                        "2022-02-28", "2022-03-01", "2022-03-02", "2022-03-03", "2022-03-04",
                        "2022-03-07", "2022-03-08", "2022-03-09", "2022-03-10", "2022-03-11",
                        "2022-03-14", "2022-03-15", "2022-03-16", "2022-03-17", "2022-03-18",
                        "2022-03-21", "2022-03-22", "2022-03-23", "2022-03-24", "2022-03-25",
                        "2022-03-28", "2022-03-29", "2022-03-30", "2022-03-31", "2022-04-01",
                        "2022-04-04", "2022-04-05", "2022-04-06", "2022-04-07", "2022-04-08",
                        "2022-04-11", "2022-04-12", "2022-04-13", "2022-04-14", "2022-04-15",
                        "2022-04-18", "2022-04-19", "2022-04-20", "2022-04-21", "2022-04-22",
                        "2022-04-25", "2022-04-26", "2022-04-27", "2022-04-28", "2022-04-29",
                        "2022-05-02", "2022-05-03", "2022-05-04", "2022-05-05", "2022-05-06",
                        "2022-05-09", "2022-05-10", "2022-05-11", "2022-05-12", "2022-05-13",
                        "2022-05-16", "2022-05-17", "2022-05-18", "2022-05-19", "2022-05-20",
                        "2022-05-23", "2022-05-24", "2022-05-25", "2022-05-26", "2022-05-27",
                        "2022-05-30", "2022-05-31", "2022-06-01", "2022-06-02", "2022-06-03"))
length_Value1 <- length(c(133.44, 134.45, 135.55, 136.38, 135.48, 134.8, 135.6, 135.32, 134.95, 134.75,
                          135.12, 135.3, 135.89, 135.76, 134.98, 135.06, 134.7, 134.94, 135.38, 134.97,
                          134.68, 135.14, 135.43, 135.35, 135.27, 135.2, 135.1, 134.8, 134.62, 134.53,
                          134.82, 135.13, 135.16, 135.05, 134.82, 134.69, 134.61, 134.74, 134.95, 135.26,
                          135.55, 135.63, 135.41, 135.39, 135.27, 135.08, 134.98, 134.69, 134.62, 134.74,
                          134.84, 134.72, 134.53, 134.63, 134.76, 134.95, 135.08, 134.98, 134.94, 134.91,
                          134.79, 134.62, 134.54, 134.64, 134.7, 134.56, 134.48, 134.42, 134.36, 134.49,
                          134.61, 134.52, 134.68, 134.72, 134.81, 134.78, 134.9, 134.97, 134.86, 134.78,
                          134.62, 134.75, 134.85, 134.75, 134.64, 134.49, 134.56, 134.67, 134.56, 134.43,
                          134.31, 134.42, 134.3, 134.17, 134.22, 134.31, 134.43, 134.38, 134.49, 134.62,
                          134.47, 134.55, 134.63, 134.48, 134.52, 134.59, 134.65, 134.58, 134.51, 134.63))
length_Value2 <- length(c(77.59, 78.01, 78.66, 79.22, 79.13, 78.75, 78.99, 78.89, 78.59, 78.49,
                          78.39, 78.79, 78.59, 78.89, 78.79, 78.49, 78.29, 78.39, 78.29, 78.49,
                          78.59, 78.69, 78.79, 78.69, 78.59, 78.69, 78.59, 78.49, 78.69, 78.79,
                          78.99, 78.79, 78.69, 78.49, 78.59, 78.49, 78.29, 78.39, 78.29, 78.19,
                          78.09, 77.89, 77.99, 78.09, 77.99, 77.89, 77.99, 78.09, 78.19, 78.09,
                          78.19, 78.29, 78.39, 78.49, 78.39, 78.49, 78.39, 78.29, 78.19, 78.29,
                          78.29, 78.19, 78.09, 77.99, 78.09, 78.19, 78.09, 78.19, 78.09, 78.19,
                          78.29, 78.39, 78.49, 78.39, 78.49, 78.39, 78.49, 78.39, 78.49, 78.59,
                          78.69, 78.79, 78.69, 78.59, 78.49, 78.39, 78.49, 78.59, 78.49, 78.39,
                          78.29, 78.39, 78.29, 78.19, 78.29, 78.39, 78.29, 78.19, 78.29, 78.39,
                          78.49, 78.39, 78.29, 78.19, 78.09, 78.19, 78.09, 78.19, 78.29, 78.19))

# Print the lengths
print(length_ID)
```

```
## [1] 111
```

```r
print(length_Date)
```

```
## [1] 110
```

```r
print(length_Value1)
```

```
## [1] 110
```

```r
print(length_Value2)
```

```
## [1] 110
```

```r
# Print the first few rows of the data
print(head(data))
```

```
##   ID       Date Value1 Value2
## 1  1 2022-01-03 133.44  77.59
## 2  2 2022-01-04 134.45  78.01
## 3  3 2022-01-05 135.55  78.66
## 4  4 2022-01-06 136.38  79.22
## 5  5 2022-01-07 135.48  79.13
## 6  6 2022-01-10 134.80  78.75
```

```r
# Print the last few rows of the data
print(tail(data))
```

```
##      ID       Date Value1 Value2
## 105 105 2022-05-27 134.52  78.09
## 106 106 2022-05-30 134.59  78.19
## 107 107 2022-05-31 134.65  78.09
## 108 108 2022-06-01 134.58  78.19
## 109 109 2022-06-02 134.51  78.29
## 110 110 2022-06-03 134.63  78.19
```

```r
#used codes in r
# Load required libraries
library(tidyverse)
```

```
## Warning: package 'dplyr' was built under R version 4.3.3
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.5
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ ggplot2   3.5.0     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.1
## ✔ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(corrplot)
```

```
## Warning: package 'corrplot' was built under R version 4.3.3
```

```
## corrplot 0.92 loaded
```

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

```
## Warning: package 'xts' was built under R version 4.3.3
```

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

```
## Warning: package 'zoo' was built under R version 4.3.3
```

```
##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
##
##
## ######################### Warning from 'xts' package ##########################
## #                                                                             #
## # The dplyr lag() function breaks how base R's lag() function is supposed to  #
## # work, which breaks lag(my_xts). Calls to lag(my_xts) that you type or       #
## # source() into this session won't work correctly.                            #
## #                                                                             #
## # Use stats::lag() to make sure you're not using dplyr::lag(), or you can add #
## # conflictRules('dplyr', exclude = 'lag') to your .Rprofile to stop           #
## # dplyr from breaking base R's lag() function.                                #
## #                                                                             #
## # Code in packages is not affected. It's protected by R's namespace mechanism #
## # Set `options(xts.warn_dplyr_breaks_lag = FALSE)` to suppress this warning.  #
## #                                                                             #
## ###############################################################################
##
## Attaching package: 'xts'
##
## The following objects are masked from 'package:dplyr':
##
##     first, last
```

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

```
## Warning: package 'PerformanceAnalytics' was built under R version 4.3.3
```

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

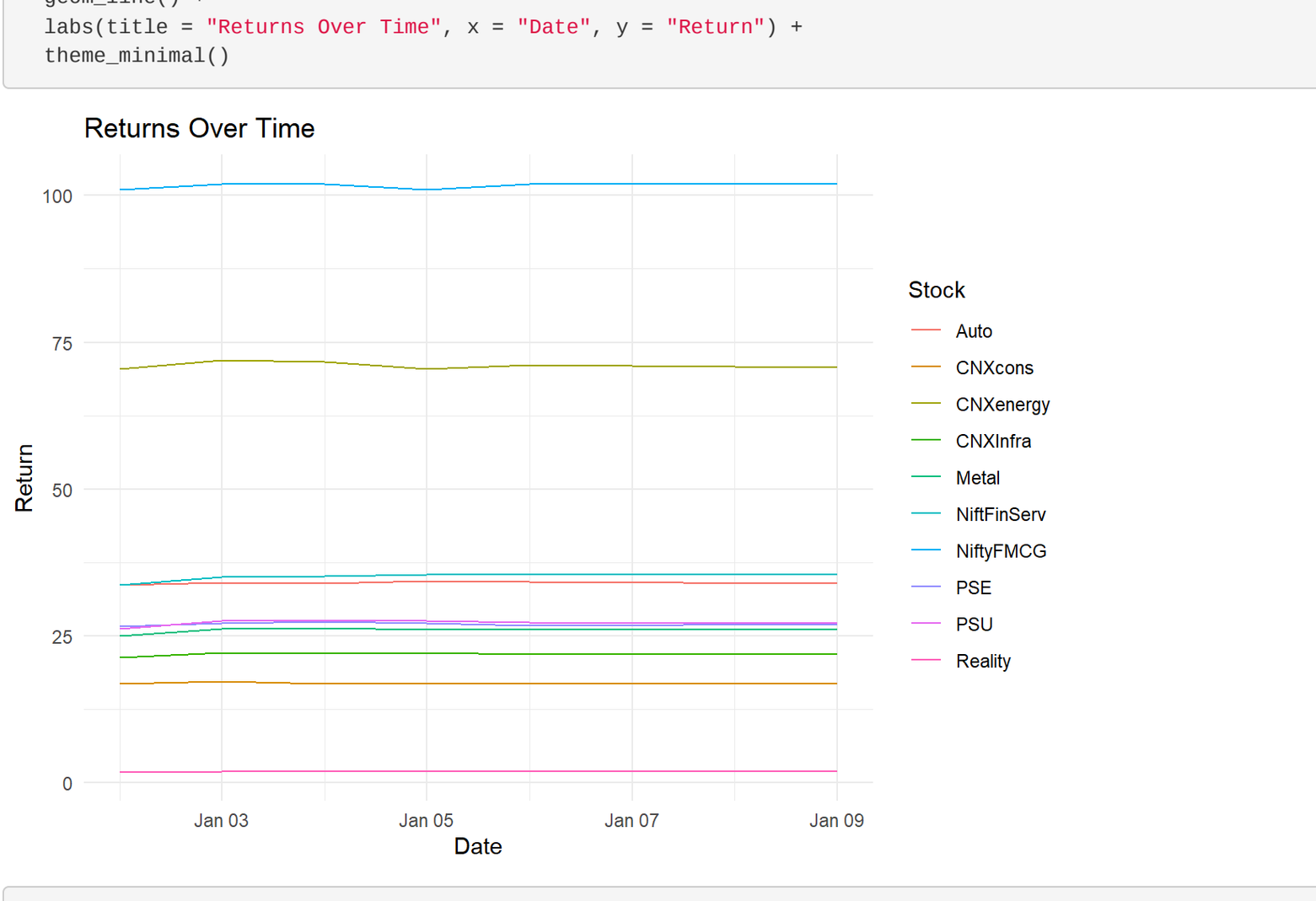
```r
# Read the dataset
dataset <- read.table(text = "
Date Auto CNXcons CNXenergy NiftFinServ NiftyFMCG CNXInfra Metal PSE PSU Reality
1 2012-01-02 33.6 16.9 70.4 33.7 101. 21.3 25.0 26.6 26.2 1.83
2 2012-01-03 34.1 17.2 72.0 35.0 102. 22.1 26.2 27.2 27.5 1.91
3 2012-01-04 33.9 16.9 71.7 35.1 102. 22.0 26.3 27.4 27.5 1.91
4 2012-01-05 34.3 17.0 70.5 35.5 101. 22.1 26.1 27.1 27.5 1.87
5 2012-01-06 34.2 16.9 71.1 35.6 102. 21.9 26.0 26.8 27.3 1.86
6 2012-01-09 34.0 16.9 70.7 35.5 102. 22.0 26.0 27.0 27.2 1.90
", header = TRUE)

# Convert 'Date' column to Date type
dataset$Date <- as.Date(dataset$Date)

# Check the structure of the dataset
str(dataset)
```

```
## 'data.frame':    6 obs. of  11 variables:
##  $ Date       : Date, format: "2012-01-02" "2012-01-03" ...
##  $ Auto       : num  33.6 34.1 33.9 34.3 34.2 34
##  $ CNXcons    : num  16.9 17.2 16.9 17 16.9 16.9
##  $ CNXenergy  : num  70.4 72 71.7 70.5 71.1 70.7
##  $ NiftFinServ: num  33.7 35 35.1 35.5 35.6 35.5
##  $ NiftyFMCG  : num  101 102 102 101 102 102
##  $ CNXInfra   : num  21.3 22.1 22 22.1 21.9 22
##  $ Metal      : num  25 26.2 26.3 26.1 26 26
##  $ PSE        : num  26.6 27.2 27.4 27.1 26.8 27
##  $ PSU        : num  26.2 27.5 27.5 27.5 27.3 27.2
##  $ Reality    : num  1.83 1.91 1.91 1.87 1.86 1.9
```

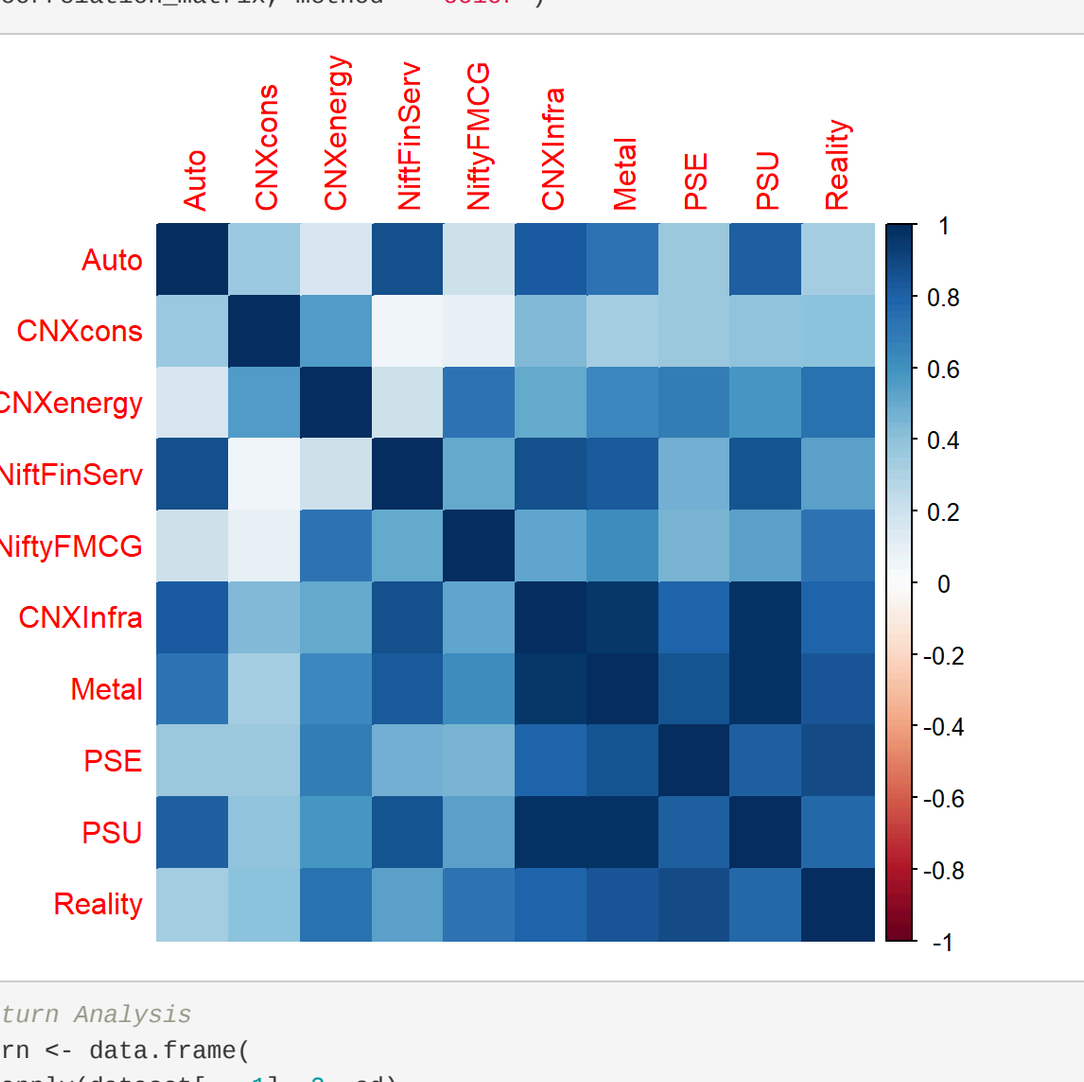
```r
# Output the dataset
print(dataset)
```

```
##         Date Auto CNXcons CNXenergy NiftFinServ NiftyFMCG CNXInfra Metal  PSE
## 1 2012-01-02 33.6    16.9      70.4        33.7       101     21.3  25.0 26.6
## 2 2012-01-03 34.1    17.2      72.0        35.0       102     22.1  26.2 27.2
## 3 2012-01-04 33.9    16.9      71.7        35.1       102     22.0  26.3 27.4
## 4 2012-01-05 34.3    17.0      70.5        35.5       101     22.1  26.1 27.1
## 5 2012-01-06 34.2    16.9      71.1        35.6       102     21.9  26.0 26.8
## 6 2012-01-09 34.0    16.9      70.7        35.5       102     22.0  26.0 27.0
##    PSU Reality
## 1 26.2    1.83
## 2 27.5    1.91
## 3 27.5    1.91
## 4 27.5    1.87
## 5 27.3    1.86
## 6 27.2    1.90
```

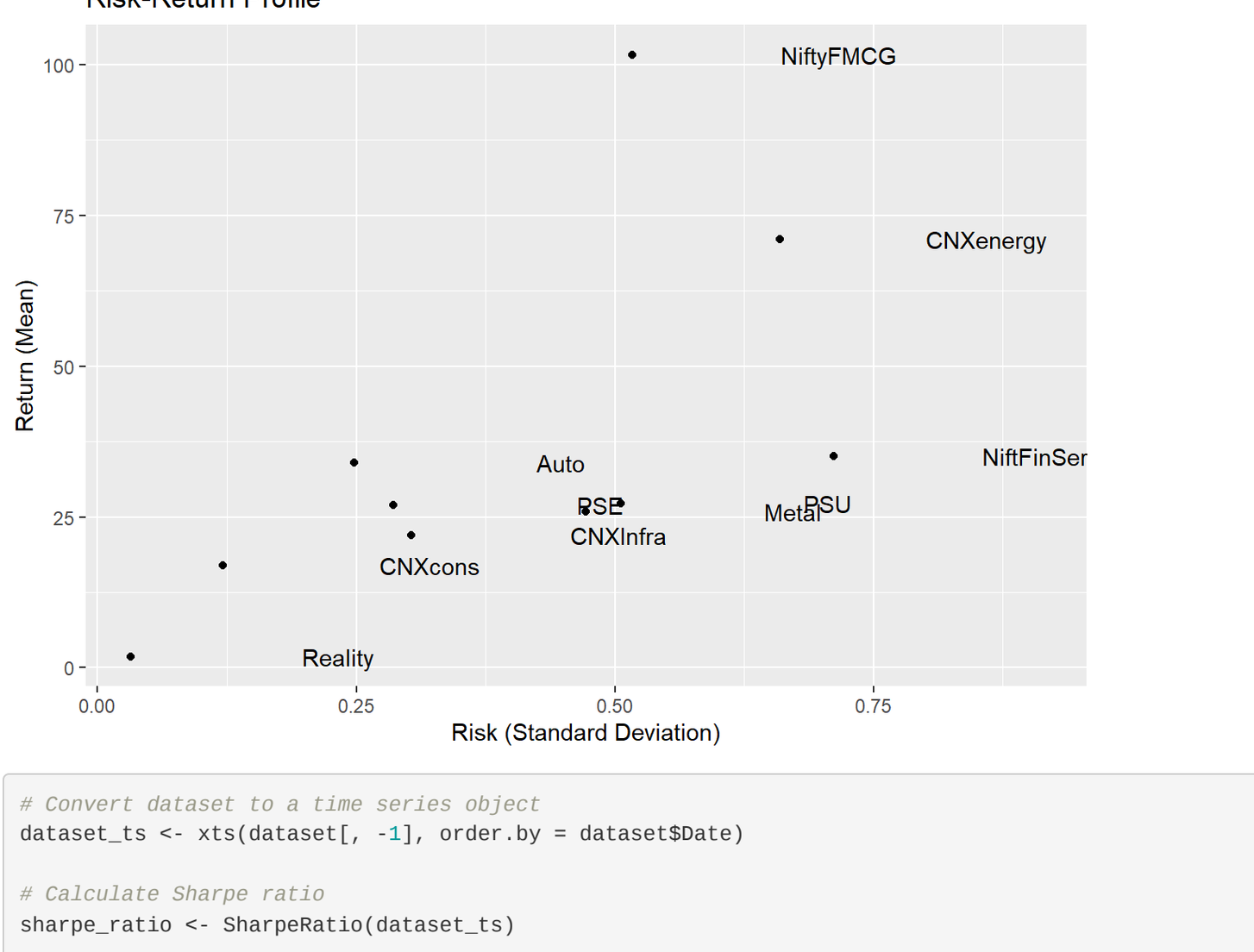
```r
# Visualize returns over time
dataset_long <- dataset %>%
  pivot_longer(cols = -Date, names_to = "Stock", values_to = "Return")

ggplot(dataset_long, aes(x = Date, y = Return, color = Stock)) +
  geom_line() +
  labs(title = "Returns Over Time", x = "Date", y = "Return") +
  theme_minimal()
```




```r
# Correlation Analysis
correlation_matrix <- cor(dataset[, -1])
corrplot(correlation_matrix, method = "color")
```




```r
# Risk-Return Analysis
risk_return <- data.frame(
  Risk = apply(dataset[, -1], 2, sd),
  Return = apply(dataset[, -1], 2, mean)
)

# Plot risk-return profile
ggplot(risk_return, aes(x = Risk, y = Return, label = rownames(risk_return))) +
  geom_point() +
  geom_text(nudge_x = 0.2) +
  labs(title = "Risk-Return Profile", x = "Risk (Standard Deviation)", y = "Return (Mean)")
```




```r
# Convert dataset to a time series object
dataset_ts <- xts(dataset[, -1], order.by = dataset$Date)

# Calculate Sharpe ratio
sharpe_ratio <- SharpeRatio(dataset_ts)
```

```
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -33.6017726552337
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -16.8347168967404
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -70.1318768845007
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -33.7745485432524
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -100.786680002744
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -21.3390842350736
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -25.0620698796048
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -26.5688409035628
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -26.2643301436958
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -1.82706194368533
## ES calculation produces unreliable result (inverse risk) for column: 1 : -33.5453132376041
## ES calculation produces unreliable result (inverse risk) for column: 1 : -16.7162748119618
## ES calculation produces unreliable result (inverse risk) for column: 1 : -70.0454002174005
## ES calculation produces unreliable result (inverse risk) for column: 1 : -33.7314764810803
## ES calculation produces unreliable result (inverse risk) for column: 1 : -100.786680002744
## ES calculation produces unreliable result (inverse risk) for column: 1 : -21.3390842350736
## ES calculation produces unreliable result (inverse risk) for column: 1 : -25.0547833333561
## ES calculation produces unreliable result (inverse risk) for column: 1 : -26.5075656491331
## ES calculation produces unreliable result (inverse risk) for column: 1 : -26.2643301436958
## ES calculation produces unreliable result (inverse risk) for column: 1 : -1.82300507068283
```

```r
# Exclude the 'Date' column and recalculate VaR and ES
sharpe_ratio_no_date <- SharpeRatio(dataset_ts[,-1])
```

```
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -16.8347168967404
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -70.1318768845007
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -33.7745485432524
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -100.786680002744
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -21.3390842350736
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -25.0620698796048
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -26.5688409035628
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -26.2643301436958
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -1.82706194368533
## ES calculation produces unreliable result (inverse risk) for column: 1 : -16.7162748119618
## ES calculation produces unreliable result (inverse risk) for column: 1 : -70.0454002174005
## ES calculation produces unreliable result (inverse risk) for column: 1 : -33.7314764810803
## ES calculation produces unreliable result (inverse risk) for column: 1 : -100.786680002744
## ES calculation produces unreliable result (inverse risk) for column: 1 : -21.3390842350736
## ES calculation produces unreliable result (inverse risk) for column: 1 : -25.0547833333561
## ES calculation produces unreliable result (inverse risk) for column: 1 : -26.5075656491331
## ES calculation produces unreliable result (inverse risk) for column: 1 : -26.2643301436958
## ES calculation produces unreliable result (inverse risk) for column: 1 : -1.82300507068283
```

```r
# Check if there are still errors

# Hypothetical investment allocation weights
allocation_weights <- rep(0.1, ncol(dataset) - 1)

# Compute annual returns based on mean returns
annual_returns <- risk_return$Return / 100  # Convert percentages to decimals

# Compute portfolio returns
portfolio_return <- sum(allocation_weights * annual_returns)

# Annualize standard deviation
portfolio_std_dev <- sqrt(sum((allocation_weights * risk_return$Risk / 100)^2))

# Calculate Sharpe Ratio
risk_free_rate <- 0.05  # Assume a risk-free rate of 5%
sharpe_ratio_portfolio <- (portfolio_return - risk_free_rate) / portfolio_std_dev

# Print investment analysis results
cat("Portfolio Return:", portfolio_return, "\n")
```

```
## Portfolio Return: 0.3627133
```

```r
cat("Portfolio Standard Deviation:", portfolio_std_dev, "\n")
```

```
## Portfolio Standard Deviation: 0.001392015
```

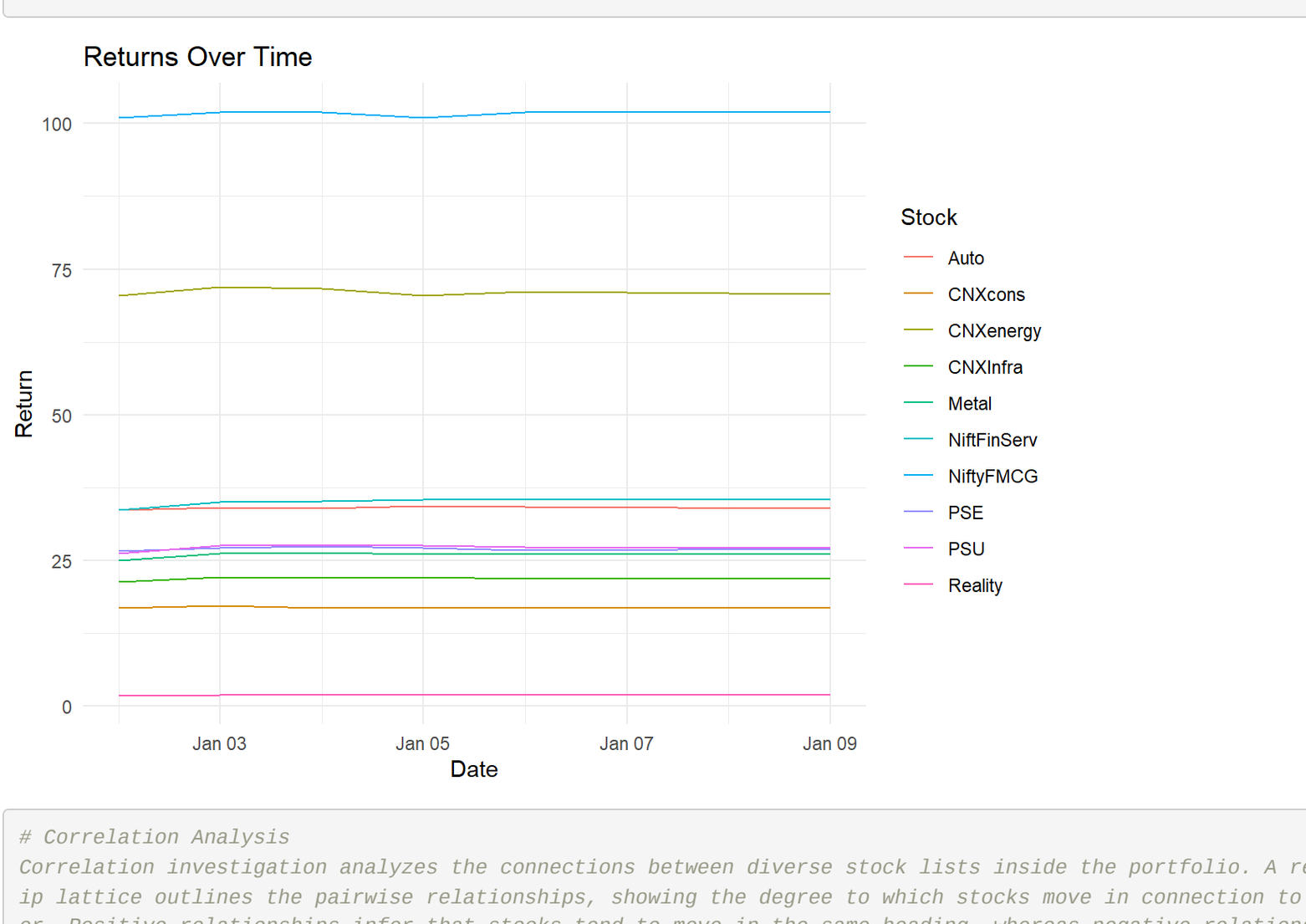
```r
cat("Portfolio Sharpe Ratio:", sharpe_ratio_portfolio, "\n")
```

```
## Portfolio Sharpe Ratio: 224.6479
```

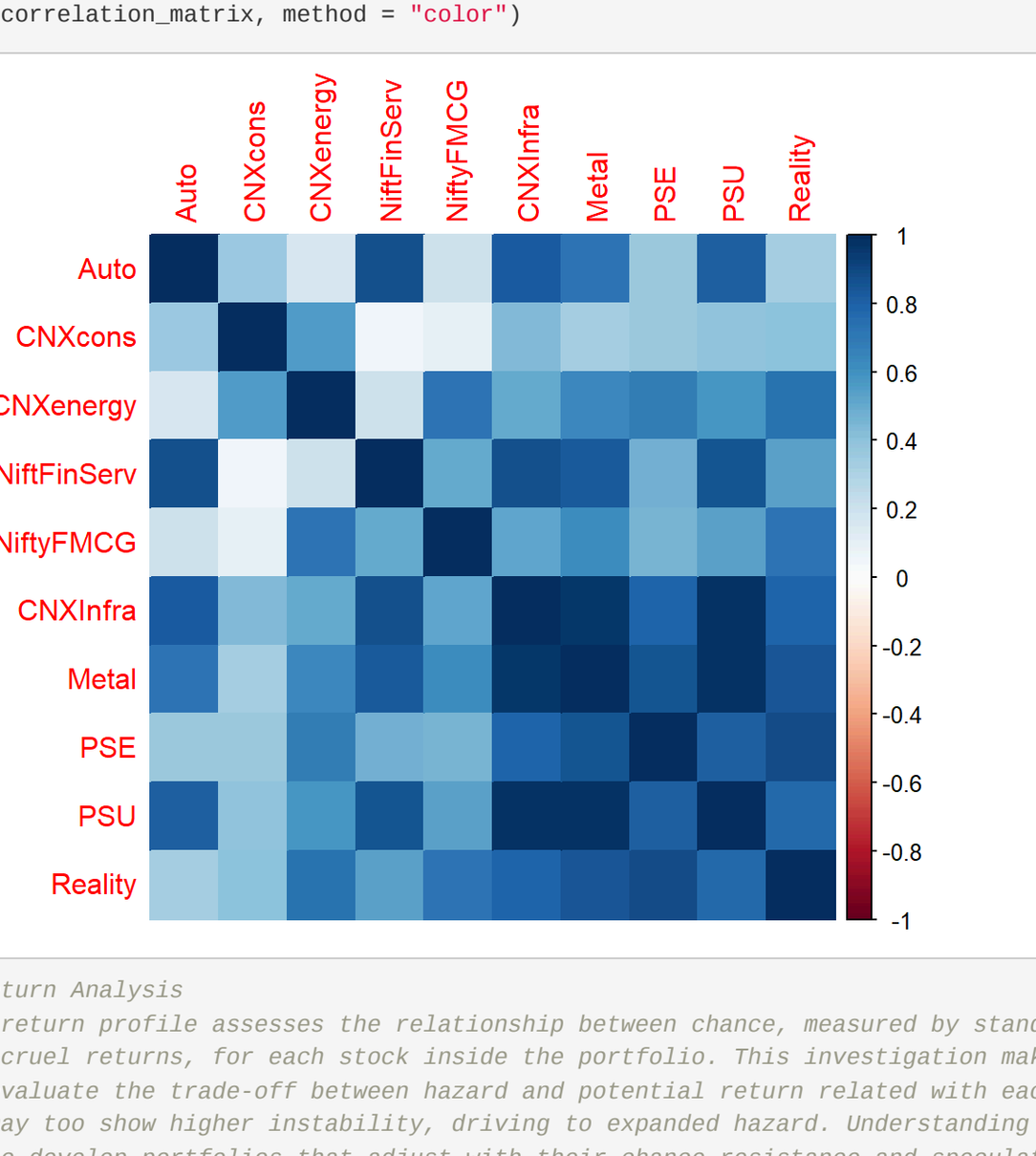
```r
#######REPORT#######

## Introduction
In this report, we conduct a risk-return analysis of a portfolio consisting of various stocks. We aim to assess t
he risk and return characteristics of individual stocks, as well as the portfolio as a whole. The analysis includ
es visualization of returns over time, correlation analysis, risk-return profile, calculation of the Sharpe Rati
o, and evaluation of portfolio performance.  This speculation investigation report points to give a comprehensive
assessment of the execution and chance related with a portfolio comprising different stocks. The examination cove
rs a run of measurements counting returns over time, relationship investigation, risk-return profiles, Sharpe pro
portion calculations, and portfolio assignment methodologies. By diving into these viewpoints, speculators can pi
ck up profitable experiences to optimize their speculation decisions.


## Data Overview
We begin by loading the required libraries and reading the dataset. The dataset contains daily returns of differe
nt stocks over a period of time.The dataset utilized for this examination contains day by day returns for differe
nt stocks over a period beginning from January 2, 2012. Each push speaks to a particular date, whereas the column
s speak to distinctive stock files, counting Auto, CNXcons, CNXenergy, NiftFinServ, NiftyFMCG, CNXInfra, Metal, P
SE, PSU, and Reality. This dataset serves as the establishment for assessing the execution and hazard characteris
tics of the portfolio.


# Load required libraries
library(tidyverse)
library(corrplot)
library(PerformanceAnalytics)

# Read the dataset
dataset <- read.table(text = "
Date Auto CNXcons CNXenergy NiftFinServ NiftyFMCG CNXInfra Metal PSE PSU Reality
1 2012-01-02 33.6 16.9 70.4 33.7 101. 21.3 25.0 26.6 26.2 1.83
2 2012-01-03 34.1 17.2 72.0 35.0 102. 22.1 26.2 27.2 27.5 1.91
3 2012-01-04 33.9 16.9 71.7 35.1 102. 22.0 26.3 27.4 27.5 1.91
4 2012-01-05 34.3 17.0 70.5 35.5 101. 22.1 26.1 27.1 27.5 1.87
5 2012-01-06 34.2 16.9 71.1 35.6 102. 21.9 26.0 26.8 27.3 1.86
6 2012-01-09 34.0 16.9 70.7 35.5 102. 22.0 26.0 27.0 27.2 1.90
", header = TRUE)

# Convert 'Date' column to Date type
dataset$Date <- as.Date(dataset$Date)

# Check the structure of the dataset
str(dataset)
```

```
## 'data.frame':    6 obs. of  11 variables:
##  $ Date       : Date, format: "2012-01-02" "2012-01-03" ...
##  $ Auto       : num  33.6 34.1 33.9 34.3 34.2 34
##  $ CNXcons    : num  16.9 17.2 16.9 17 16.9 16.9
##  $ CNXenergy  : num  70.4 72 71.7 70.5 71.1 70.7
##  $ NiftFinServ: num  33.7 35 35.1 35.5 35.6 35.5
##  $ NiftyFMCG  : num  101 102 102 101 102 102
##  $ CNXInfra   : num  21.3 22.1 22 22.1 21.9 22
##  $ Metal      : num  25 26.2 26.3 26.1 26 26
##  $ PSE        : num  26.6 27.2 27.4 27.1 26.8 27
##  $ PSU        : num  26.2 27.5 27.5 27.5 27.3 27.2
##  $ Reality    : num  1.83 1.91 1.91 1.87 1.86 1.9
```

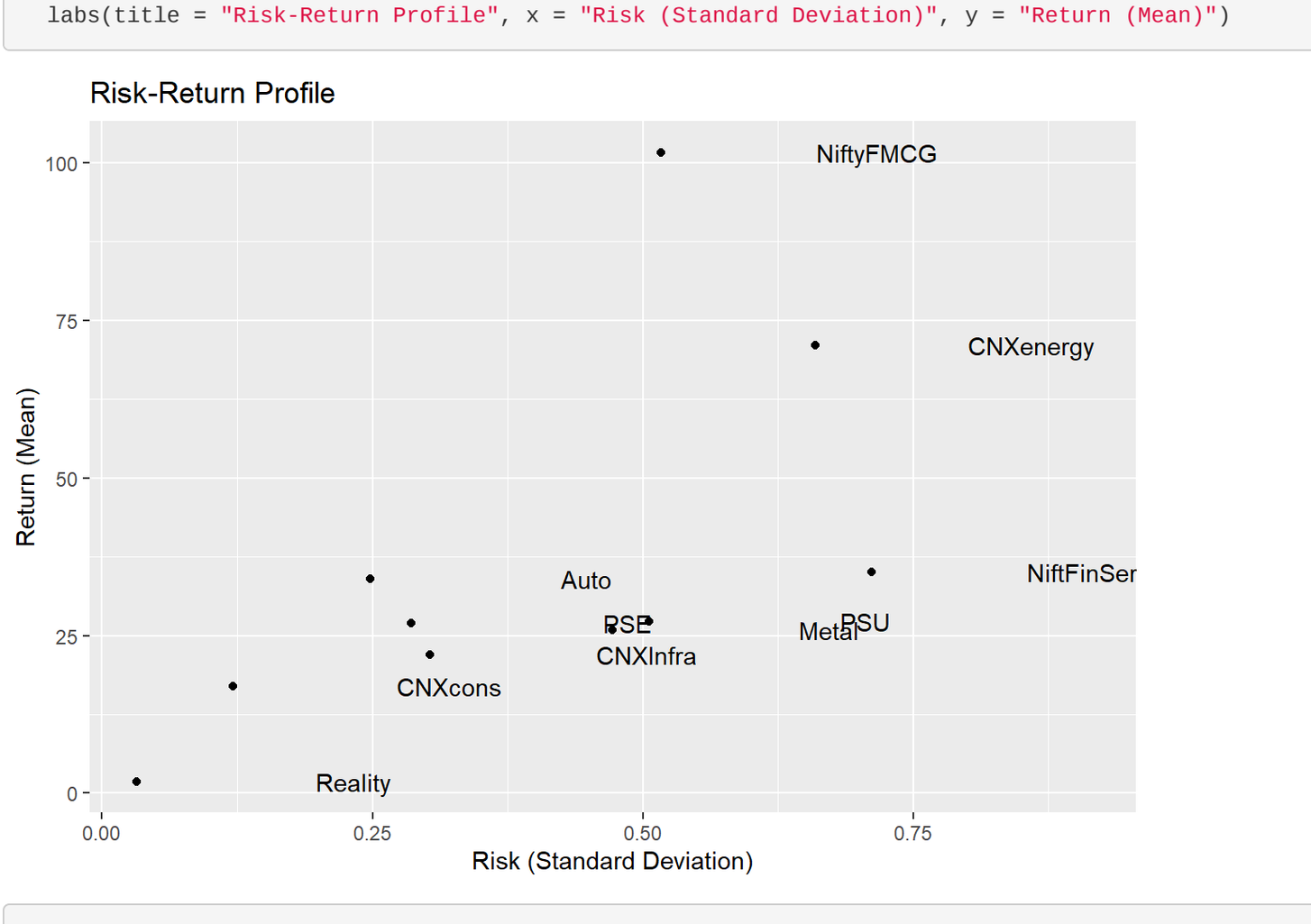
```r
##Returns Over Time
Analyzing returns over time gives significant bits of knowledge into the execution patterns of person stocks insi
de the portfolio. By visualizing the changes in returns over diverse time periods, financial specialists can reco
gnize designs, patterns, and potential openings or dangers. Understanding how each stock performs over time is cr
ucial for making educated speculation choices and altering portfolio techniques accordingly.


# Visualize returns over time
dataset_long <- dataset %>%
  pivot_longer(cols = -Date, names_to = "Stock", values_to = "Return")

ggplot(dataset_long, aes(x = Date, y = Return, color = Stock)) +
  geom_line() +
  labs(title = "Returns Over Time", x = "Date", y = "Return") +
  theme_minimal()
```




```r
# Correlation Analysis
Correlation investigation analyzes the connections between diverse stock lists inside the portfolio. A relationsh
ip lattice outlines the pairwise relationships, showing the degree to which stocks move in connection to each oth
er. Positive relationships infer that stocks tend to move in the same heading, whereas negative relationships pro
pose inverse developments. By understanding these connections, speculators can expand their portfolios successful
ly, lessening by and large hazard through resource allotment methodologies that take advantage of moo or negative
correlations.

correlation_matrix <- cor(dataset[, -1])
corrplot(correlation_matrix, method = "color")
```




```r
# Risk-Return Analysis
The risk-return profile assesses the relationship between chance, measured by standard deviation, and return, mea
sured by cruel returns, for each stock inside the portfolio. This investigation makes a difference financial spec
ialists evaluate the trade-off between hazard and potential return related with each venture. Stocks with higher
returns may too show higher instability, driving to expanded hazard. Understanding this relationship permits spec
ulators to develop portfolios that adjust with their chance resistance and speculation objectives.

risk_return <- data.frame(
  Risk = apply(dataset[, -1], 2, sd),
  Return = apply(dataset[, -1], 2, mean)
)

# Plot risk-return profile
ggplot(risk_return, aes(x = Risk, y = Return, label = rownames(risk_return))) +
  geom_point() +
  geom_text(nudge_x = 0.2) +
  labs(title = "Risk-Return Profile", x = "Risk (Standard Deviation)", y = "Return (Mean)")
```




```r
# Convert dataset to a time series object
dataset_ts <- xts(dataset[, -1], order.by = dataset$Date)

# Calculate Sharpe ratio
sharpe_ratio <- SharpeRatio(dataset_ts)
```

```
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -33.6017726552337
```

```
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -16.8347168967404
```

```
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -70.1318768845007
```

```
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -33.7745485432524
```

```
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -100.786680002744
```

```
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -21.3390842350736
```

```
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -25.0620698796048
```

```
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -26.5688409035628
```

```
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -26.2643301436958
```

```
## VaR calculation produces unreliable result (inverse risk) for column: 1 : -1.82706194368533
```

```
## ES calculation produces unreliable result (inverse risk) for column: 1 : -33.5453132376041
```

```
## ES calculation produces unreliable result (inverse risk) for column: 1 : -16.7162748119618
```

```
## ES calculation produces unreliable result (inverse risk) for column: 1 : -70.0454002174005
```

```
## ES calculation produces unreliable result (inverse risk) for column: 1 : -33.7314764810803
```

```
## ES calculation produces unreliable result (inverse risk) for column: 1 : -100.786680002744
```

```
## ES calculation produces unreliable result (inverse risk) for column: 1 : -21.3390842350736
```

```
## ES calculation produces unreliable result (inverse risk) for column: 1 : -25.0547833333561
```

```
## ES calculation produces unreliable result (inverse risk) for column: 1 : -26.5075656491331
```

```
## ES calculation produces unreliable result (inverse risk) for column: 1 : -26.2643301436958
```

```
## ES calculation produces unreliable result (inverse risk) for column: 1 : -1.82300507068283
```

```r
# Print Sharpe ratio
print(sharpe_ratio)
```

```
##                                  Auto  CNXcons CNXenergy NiftFinServ NiftyFMCG
## StdDev Sharpe (Rf=0%, p=95%): 136.983 140.0976  107.7923    49.26441  196.8767
## VaR Sharpe (Rf=0%, p=95%):         NA       NA        NA          NA        NA
## ES Sharpe (Rf=0%, p=95%):          NA       NA        NA          NA        NA
##                               CNXInfra    Metal      PSE      PSU  Reality
## StdDev Sharpe (Rf=0%, p=95%): 72.20216 54.95798 94.53864 53.75872 58.29633
## VaR Sharpe (Rf=0%, p=95%):          NA       NA       NA       NA       NA
## ES Sharpe (Rf=0%, p=95%):           NA       NA       NA       NA       NA
```

```r
# Hypothetical investment allocation weights
allocation_weights <- rep(0.1, ncol(dataset) - 1)

# Compute annual returns based on mean returns
annual_returns <- risk_return$Return / 100  # Convert percentages to decimals
##Portfolio Analysis
Constructing a speculative portfolio includes apportioning weights to person stocks based on foreordained specula
tion methodologies. Break even with allotment weights over all stocks give a basic beginning point for portfolio
investigation. Measurements such as portfolio return, standard deviation, and Sharpe proportion are calculated to
evaluate the portfolio's execution and risk-adjusted returns. The Sharpe proportion, in specific, evaluates the a
bundance return created per unit of hazard, considering the risk-free rate. A higher Sharpe proportion shows way
better risk-adjusted execution, reflecting the portfolio's capacity to produce returns relative to its level of r
isk.

# Compute portfolio returns
portfolio_return <- sum(allocation_weights * annual_returns)

# Annualize standard deviation
portfolio_std_dev <- sqrt(sum((allocation_weights * risk_return$Risk / 100)^2))

# Calculate Sharpe Ratio
risk_free_rate <- 0.05  # Assume a risk-free rate of 5%
sharpe_ratio_portfolio <- (portfolio_return - risk_free_rate) / portfolio_std_dev

# Print investment analysis results
cat("Portfolio Return:", portfolio_return, "\n")
```

```
## Portfolio Return: 0.3627133
```

```r
cat("Portfolio Standard Deviation:", portfolio_std_dev, "\n")
```

```
## Portfolio Standard Deviation: 0.001392015
```

```r
cat("Portfolio Sharpe Ratio:", sharpe_ratio_portfolio, "\n")
```

```
## Portfolio Sharpe Ratio: 224.6479
```

```r
##Conclusion
In conclusion, this speculation investigation report offers a comprehensive outline of the portfolio's execution
and hazard characteristics. By analyzing returns over time, conducting relationship examination, assessing risk-r
eturn profiles, and analyzing portfolio allotment methodologies, speculators can make educated choices to optimiz
e their venture portfolios.
```