PORTFOLIO OPTIMIZATION

Team Quantifiers

ERIC SARGENT, DEBALEENA SAHA, ARKOJYOTI HORE, SAMARPITA CHAKRABARTY

Contents

Executive Summary

Portfolio Optimization is a complex topic made up of competing theories and complicated strategies for choosing the best proportion of the various assets to be held in a portfolio in order to maximize an investor's preferences with respect to return and risk. The goal for this project was to not only explore different portfolio optimization methods, but bring different approaches together, culminating in a straightforward, user friendly app, which could take a user created stock portfolio as input and output optimal stock allocations for maximum returns within the portfolio.
The output of the app is comprised of three main sections:

- Monthly Returns - Here multiple stocks that are selected can be analyzed and compared based on their performance over the last one year.
- Individual Stock Performance - Detailed performance of individual stocks can be viewed.
- Optimal Portfolio - Optimal Percentage of Asset Allocation for each stock, Volatility, and Expected Returns can be viewed for user selected portfolio of stocks.

This report provides a background of portfolio optimization, introduces the different tools and terminology that are used for it and summarizes the methods which have been used in this particular project for the specific goal of portfolio optimization.

Background and Optimization Methodology

The first step in the project was to gain domain knowledge on stocks, optimization methods, and typical return rates. Starting with the basics, a stock is essentially a "small piece of a company" that when traded publicly, can be owned by an individual (Money101, 2017). Companies typically offer publicly traded stock as a way to raise capital for investments instead of taking pursuing loans or other methods for securing financing (Fitzsimmons, 2009).

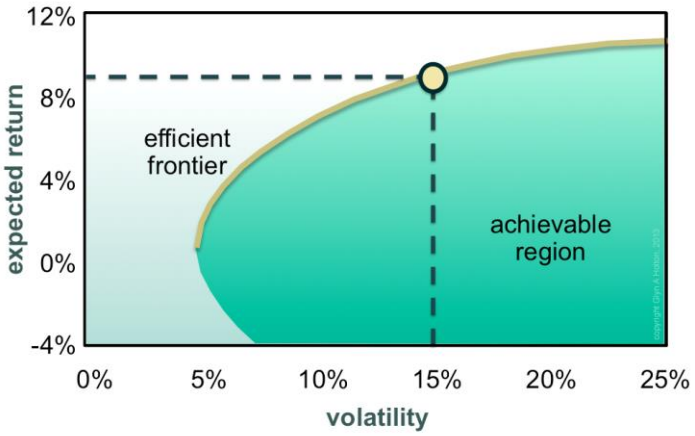
An investor makes money from a stock purchase in several different ways. Some stocks (typically more mature companies) will pay out dividends which are company profits that are paid back to shareholders. An investor may also realize returns when they sell a stock that has appreciated in value (Money101, 2017). Our project will focus on this second type of investing strategy.

When making an investment decision, there are two main metrics that must be considered: Risk and Return. Return is a straightforward concept and represents the amount of money an investor will receive, while risk refers to the overall volatility of an asset. Most investors set out with the goal of maximizing returns while minimizing the amount of risk they are forced to accept. For this project, both risk and return were two of the major inputs considered with optimizing portfolios.

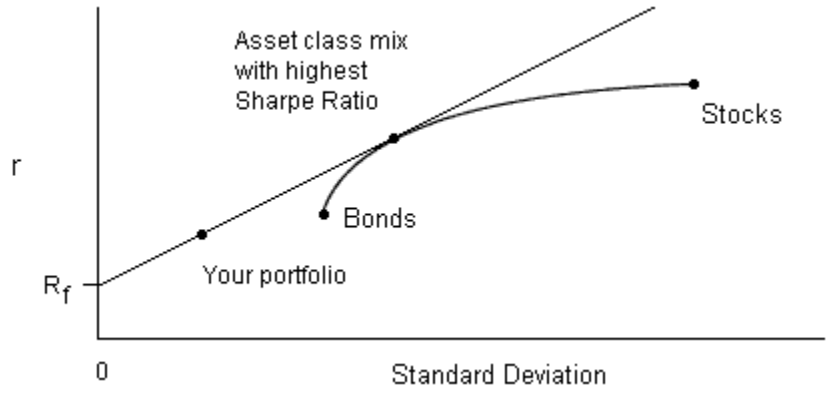
For this project, we have based our optimization on the Modern Portfolio Theory(MPT), a hypothesis put forth by Harry Markowitz in his paper "Portfolio Selection" published in 1952 by

the Journal of Finance. MPT is a theory on how risk-averse investors can construct portfolios to optimize or maximize expected return based on a given level of market risk, emphasizing that risk is an inherent part of higher reward. It suggests that it is not enough to look at the expected risk and return of one particular stock. By investing in more than one stock, an investor can reap the benefits of diversification, particularly a reduction in the riskiness of the portfolio.

Efficient Frontier Method: It provides the set of optimal portfolios that offers the highest expected return for a defined level of risk or the lowest risk for a given level of expected return. The x-axis of the graph represents the portfolio risk level (volatility) while the y-axis tracks the projected rates of return for each risk value. Portfolios that lie below the efficient frontier are sub-optimal, because they do not provide enough return for the level of risk. Portfolios that cluster to the right of the efficient frontier are also sub-optimal, because they have a higher level of risk for the defined rate of return.



Sharpe Ratio: After calculating the efficient frontier for a user selected portfolio, the Sharpe Ratio was then employed to determine where along the efficient frontier the optimal portfolio should fall. This point on the efficient frontier represents the optimal basket allocation for each selected fund that minimizes risk while maximizing expected return. The inputs to the Sharpe Ratio were the volatility and expected return of each stock.



R Packages Used

The Portfolio Analysis app was created using RShiny along with the following packages:

- stockPortfolio
- broom
- shinyjs
- magrittr
- shiny
- ggplot2
- quantmod
- tidyverse
- stringr
- corrplot
- plotly
- dplyr
- devtools
- shinysky
- xts
- lubridate
- shinythemes
- forcats
- rvest
- quadprog
- analytixware

Creating Shiny App

Rather than presenting investors with a simple RScript, our team created an RShiny app using the shiny package to allow for easier use and no coding from the user. The app is constructed by two script components:

- User Interface Script (UI): Used for designing all the buttons, graphics, and elements that will ultimate make up that app
- Server Script: Includes all the code required to make UI section functional

UI App Section

The initial step in creating the app was to design the app layout in the UI section. The major commands used were:

- fluidPage: Shiny ui.R scripts use the function fluidPage to create a display that automatically adjusts to the dimensions of the user's browser window
- TitlePanel: Establishes title for app
- ActionButton: Creates interactive button

```
ui <- fluidPage(
  useShinyjs(),
  #Displaying busy indicator while waiting to load data
  busyIndicator(text = "Loading, please wait...", wait = 4),
  theme = shinytheme("sandstone"),
  br(),
  titlePanel('Portfolio Optimization'),
  br(),
  #Displays sidebar panels and it's contents
  sidebarPanel(
    style ='overflow-y: auto; max-height: 600px;',

    helpText("Choose the Ticker Symbol, Start and End Dates"),
    selectInput("inputstocks","Select Stocks for Portfolio:",
                choices = as.factor(stockdetails[,1]), multiple = TRUE),
    verbatimTextOutput("dfStr"),
    actionButton("reset_input", "Clear"),
    HTML(paste0("<br><br><b>","Reference table to find stocks:","</b>")),
    DT::dataTableOutput('allstockdata'),
    br(),
    helpText("(Range between start date and end date should be minimum 60 working days)"),
    dateInput("startdate","Start Date:",value = Sys.Date()-365),
    dateInput("enddate","End Date:",value = Sys.Date()),
    actionButton("btnGenerate", "Generate Plots",icon("line-chart"))

  ),
```

The sidebarPanel takes inputs from the users, to display the results depending on the parameters.

One of the major challenges when designing the app layout was creating individual tabs that would display different information but would pull from the same data input source on the app home page. To solve this issue, tabset panels were created using the “tabsetPanel” and “tabPanel” commands to allow for multiple different plots to be rendered at the same time.

```
#Displays the tabs and contents of the tabs
mainPanel(
  style ='overflow-y: auto; ',
  tabsetPanel(type = 'tabs',
              tabPanel("Monthly Returns",
                       fluidRow(
                       column(8,helpText("The below plot shows monthly return of all the selected stocks based on their perform
                       column(8,helpText("For individual stock performance click on the next tab 'Individual Stock Performance'
                       column(8,helpText("For optimized portfolio click on the tab 'Optimized Portfolio'.")),
                       column(12,plotOutput("plt_mnthlyRtrn",width = "100%"))

                       )
              ),
              tabPanel("Individual Stock Performance",
                       fluidRow(
                       column(8,uiOutput("select_stocks")),
                       column(12,plotOutput("plt_individual",width = "100%"))
                       )

              ),
              tabPanel("Optimized Portfolio",
                       fluidRow(
                         column(8,helpText("Optimal Allocation Weights:")),
                         column(10,tableOutput("summary")),
                         column(8,plotlyOutput("plt_shrpRatio"))
                       )

              )

  )#tabsetPanel ends
)#mainPanel ends
```

Server App Section

In order for the app to come to life, all the buttons and displays in the UI portion of the app needed to be linked to the server coded section. The server portion of the app works by taking in all the components of the UI interface as inputs and outputs the required actions. The main syntax for the function command used to create these outputs is as follows:

```
server = function(input, output, session) {
```

Function used for Validating the input parameters taken from the user:

```
validation = function()
{
  #The Portfolio optimization is based on the premise that atleast 2 stocks need to
  #be considered for analysis
  if(length(stocks_input()) < 2)
    {
      showModal(modalDialog(
        title = "Error",
        "Select at least two stocks to optimize!!!",
        easyClose = FALSE
      ))
    }
      if(input$startdate>input$enddate)
      {
        #the end date should be greater than the start date
        showModal(modalDialog(
          title = "Error",
          "Start date should be less than end date!!! Resetting the dates!",
          easyClose = TRUE
        ))

      }
    if(input$enddate-input$startdate<60)
    {
      #the quadratic optimizer(used below) requires sufficient number of observations
      #to be able to run the optimizing algorithm
      showModal(modalDialog(
        title = "Error",
        "Too few observation for performing optimization.
        Please ensure date range for analysis to be at least 60 working days!",
        easyClose = TRUE
      ))

    }
```

Server Reactive Function

In order to allow the user to select and reselect stocks for analysis a reactive function needed to be included inside the server function. This reactive function changes output values every time new values are entered.

```
stocks_input <- reactive({
  if(input$gobutton == 0)
    return()
  return(input$inputstocks)
 })
```

Steps for Using the Portfolio Optimization App

1. Select stocks for analysis
2. Set stock data retrieval interval (monthly, daily, etc.)
3. Enter maximum allocation percentage of funds to a single stock in portfolio
4. Select if planning to short sell
5. Enter start and end dates to set range from where stock data will be retrieved

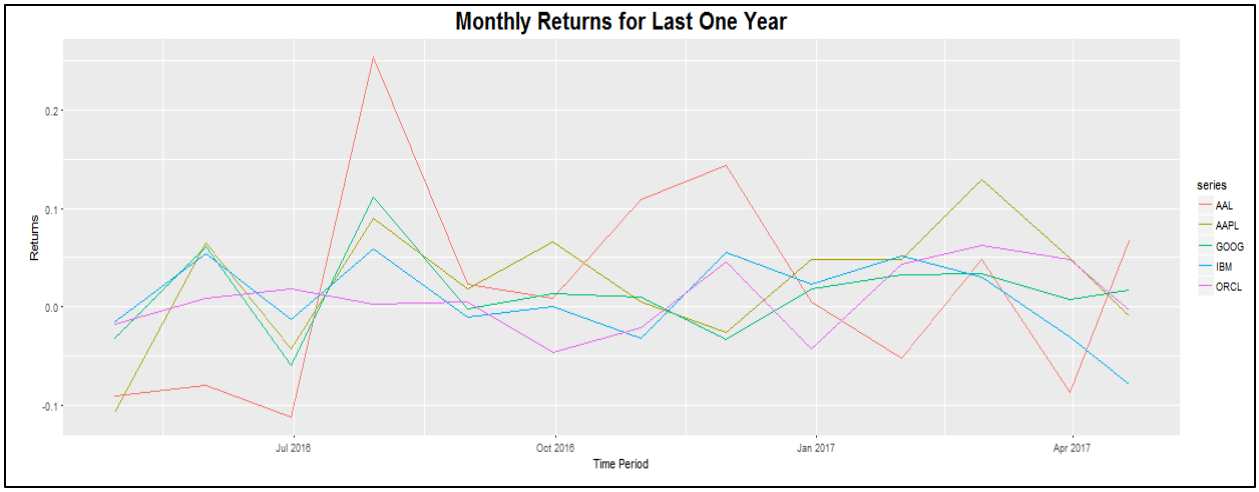
Monthly Returns Tab

The first analysis provided by the app is found on the Monthly Returns tab and computes the monthly returns of each selected stock for the last year (Note: The monthly returns chart excludes that start and end dates set by the user and only computes the monthly returns for a year from current date).

```
#Generating plot for monthly returns using the data loaded in data.plt_mnthlyRtrn function
output$plt_mnthlyRtrn <- renderPlot({
  #Checking validation errors(Start)
  if(input$btnGenerate == 0)
    return()
  if(length(stocks_input()) < 2)
    return()
  if(input$startdate>input$enddate)
    return()
  if(input$enddate-input$startdate<60)
    return()
  #Checking validation errors(End)
  isolate(tidy(data.plt_mnthlyRtrn()) %>% ggplot(aes(x=index,y=value, color=series)) +
    geom_line() +
    ggtitle("Monthly Returns for Last One Year") +
    theme(plot.title = element_text(face = "bold",
                                    color="black",
                                    size=20,
                                    hjust=0.5))+
    labs(x="Time Period", y = "Returns")
  )

 })
```

Using this plot, the potential investor gets an idea of the returns each selected stock has had for the past year as well as a general understanding of the stock volatility. This plot is meant to serve as a preliminary analysis tool to assist the user in future investment decision making.



Individual Stock Performance Tab

The second tab of the app, Plot 2, provides users with a more detailed view of stock performance. The app user can select stocks for individual examination from the drop-down option at the top of the screen to view the following information:

- Bollinger Bands
- Volume
- Moving Average Convergence/Divergence

These graphs were constructed by first plotting the performance for each selected stock over the designated time-period and then adding the following low level plot commands:

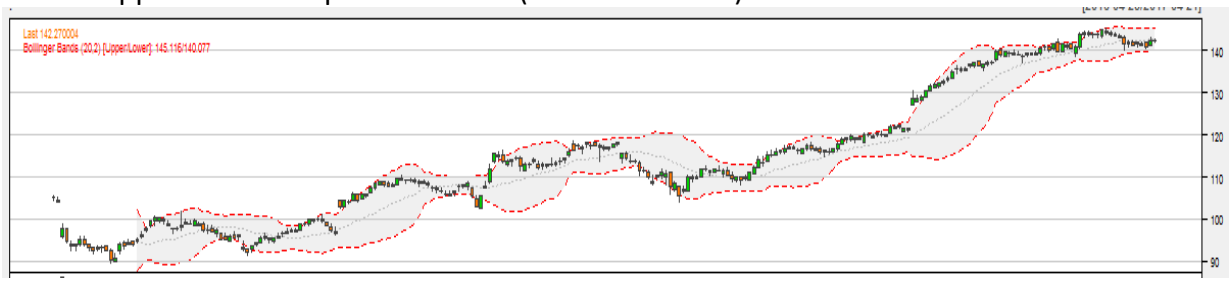
```
#Generating plots for the Adjeusted Returns of the individual stock selected
output$plt_individual <- renderPlot({
  if(is.null(data.plt_individual()))
    return()
  data.plt_individual() %>% chartSeries(TA='addVo();
                          addBBands();
                          addBBands(draw="p");
                          addMACD()',
                          theme="white"
              )

})
```

Bollinger Bands

The Bollinger Bands chart is made up of 3 components:

- Stock price (green line)
- Center line (dotted grey line)
- Upper and lower price channels (dotted red lines).

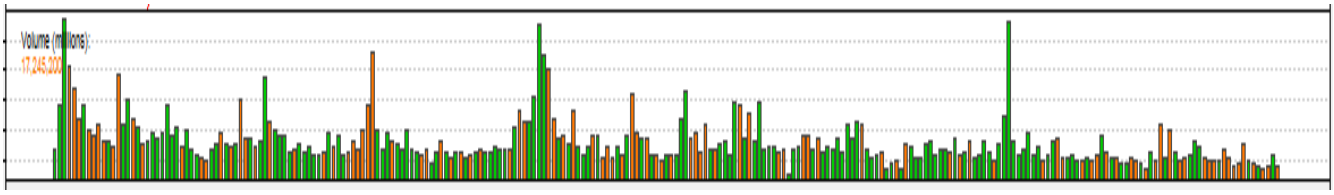


While initially convoluted, the Bollinger Bands graph is easy to decipher and incredibly useful to investors. The center line is simply an exponential moving average of the selected stock price. The upper and lower price channels are nothing more than one standard deviation above and below the moving average of the selected stock (Bollinger Bands, 2015).

When reviewing the sample Bollinger Band graph shown above, a user quickly notices that there are sections where the upper and lower channels expand and contract. These changes are due to stock volatility and can be used by investors as a way to predict future volatility. Areas of contraction (upper and lower channels close together) are typically followed by high volatility for a stock as well as the reverse scenario. Another reason for investors to use Bollinger Band Graphs is to identify times when assets are priced above or below the channels. When asset price falls below the lower channel this typically signals a good time to buy while a price above the upper channel could mean a good time to sell (Bollinger Bands, 2015). Bollinger Bands have become a very popular investment tool and our team felt it necessary to include this chart in our app to give users a more in depth analysis tool.

Volume

Put simply, volume is nothing more than "how much of a given financial asset has been traded in a given period of time" (Mitchell, 2017). Volume is important to an investor because of what it signals for the future of a stock price.



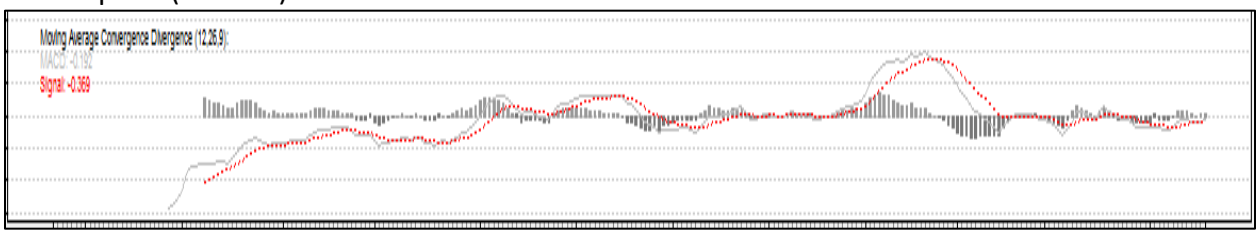
The first concept worth understanding is that volume is often associated with stock liquidity or the availability of stock in the market. If there is low volume of a stock, it is considered to be illiquid since there are not many traders buying or selling that specific stock. In contrast, a stock is considered to be liquid if there is a large number of stocks being bought or sold (Swing-Trade-Stocks, 2017).

While there are many different ways to interpret changes in volume, one of the more basic ideas is that large spikes in volume (as we can see in the volume chart above) typically represent "exhaustion moves" which tend to be followed volume decreases and subsequent price drops. Overall, volume can be used to assess the trend of a stock along with the Bollinger Bands Plot and is an important component to be included in the app (Mitchell, 2015).

Moving Average Convergence/Divergence

Like the other 2 plots, the Moving Average Convergence/Divergence (MACD) plot is initially overwhelming but quickly becomes clear with additional exploration. There are two lines that make up a MACD plot:

- MACD line: This is found by calculating a 26-day and 12-day exponential moving average of a stock price and subtracting them from one another (gray line)
- Signal line: This is nothing more than a 9-day exponential moving average of a stock price (red line)



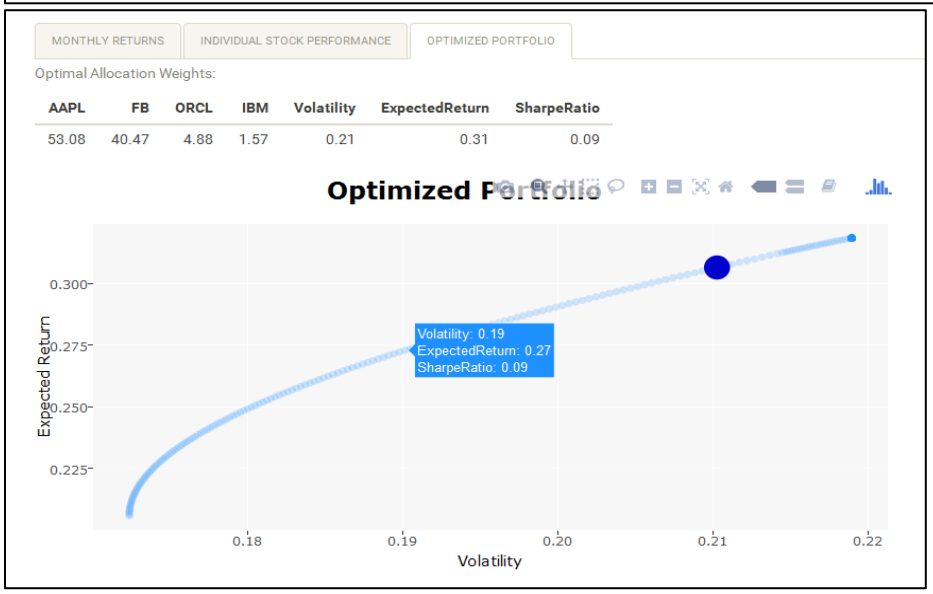
Once the plot is created, using a MACD plot is very simple. Typically, investors will consider buying a stock when the MACD line is above the signal line and sell when it is below the signal line. Because of its usefulness and simplicity to understand, our team felt it necessary to include this chart for the app users.

Optimized Portfolio Tab

The final portion of the app, is one of the key components of the app. After viewing stock performance on the previous plots, this plot shows users the percentage allocation of funds to each of the selected stocks, volatility of portfolio, Sharpe Ratio, and the expected return for the portfolio.

```
  #Displaying stock allocation and generating sharpe ratio plot
  optimal.allocation <- data.plt_shrpRatio()[data.plt_shrpRatio()$SharpeRatio==max(data.plt_shrpRatio()$SharpeRatio),]
  p <- ggplot(data.plt_shrpRatio(), aes(x=Volatility, y=ExpectedReturn,label = SharpeRatio))+
    geom_point(alpha=.2, color= "dodgerblue1")+
    geom_point(data=optimal.allocation, color="blue3", size=6)+
    ggtitle("Optimized Portfolio")+
    labs(x="Volatility", y="Expected Return")+
    theme(panel.background=element_rect(fill="gray97"), text=element_text(color="black"),
          plot.title=element_text(size=18,hjust = 0.5 ,color="black",face = "bold"))+
    theme(axis.title.y=element_text(hjust=0.4))
  return(p)

})
```



Quadratic Programming has been used to find the values of the different parameters used for computing the following values:

Volatility

Volatility depicts the risk an investor needs to consider while purchasing a particular stock. Portfolio variance considers the covariance/correlation coefficient for the stocks and is calculated by multiplying the squared weight of each stock by its corresponding variance and adding two times the weighted average weight multiplied by the covariance of all individual stocks

It has been assumed that there are 252 trading days in a year when annualizing volatility
The general rule of thumb for assessing volatility is as follows:

- Volatility =<1: The stock has volatility less than or equal to the market
- Volatility=0: The stock has no volatility
- Volatility>1: The stock has volatility greater than the market

In the example depicted above, the selected portfolio represents a relatively volatile basket of stocks.
The volatility has been calculated using:

```
efficient.frontier.matrix[counter,"Volatility"] <- sqrt(sum(quad.opt$solution *colSums((covar * quad.opt$solution)))*252)
```

Expected Return

The expected return is the weighted average of the profits of the assets in the portfolio calculated by the formula:

$$E(R) = \sum_{i=1}^{n} P_i \times R_i$$

, where Pi = Percentage Allocation, Ri = Adjusted Return for the stock. In our approach we have calculated the expected return as a cross-product of the weights vector with the returns covariance matrix as evident in the the following snippet. We are multiplying with 252 to annualize the return (assuming there are 252 trading days in a year)

```
efficient.frontier.matrix[counter,"ExpectedReturn"] <- as.numeric(quad.opt$solution %*% colMeans(data))*252
```

Sharpe Ratio

The Sharpe Ratio is a metric for measuring risk adjusted return on the investment and can be used to evaluate the overall performance of the portfolio. We can calculate the Sharpe ratio by dividing the Adjusted return (minus the risk free rate) with the volatility of the portfolio for that investment. As a rule of thumb, any Sharpe Ratio >1 is considered to be good by investors. For our analysis, we have considered the risk free rate to be zero but it can be incorporated in future analysis.

The Sharpe Ratio is found by:

```
efficient.frontier.matrix[counter,"SharpeRatio"] <- efficient.frontier.matrix[counter,"ExpectedReturn"]
/(efficient.frontier.matrix[counter,"Volatility"])
```

Using the Efficient Frontier Method, the range of possible returns is plotted against volatility to get the optimal possible outcomes for the portfolio. Then volatility and expected returns for the portfolio are calculated and used in the Sharpe Ratio calculation to output the best allocation of funds to each stock selected in the portfolio.

Limitations of the Approach

The efficient frontier and modern portfolio theory have many assumptions that may not properly represent reality. For example, one of the assumptions is that asset returns follow a normal distribution. In reality, securities may experience returns that are more than three standard deviations away from the mean more than 0.03% of the observed values. Additionally, it is assumed that investors are rational and avoid risk when possible, there are not large enough investors to influence market prices, and investors have unlimited access to borrowing and lending money at the risk-free interest rate. However, the market includes irrational and risk-seeking investors, large market participants who could influence market prices, and investors do not have unlimited access to borrowing and lending money. There are also ethical concerns when using Sharpe Ratio since there are cases where it is purposely manipulated by parties to report inaccurate numbers.
While not in the scope of this project, it is reasonable to expect other portfolio optimization techniques such as Capital Asset Pricing Model to be integrated into the app in later iterations.

Conclusion

The purpose of this project was to create an easy to use, portfolio analysis app in RShiny to help investors make smart, well informed decisions. MPT takes into account the returns of a security as well as its systematic and unsystematic risk. It then takes into account the overall return and risk of a portfolio comprised of specific securities, and it plots a line (Efficient Frontier) with all the available portfolio options. Also, Sharpe Ratio outputs an easy to understand value where negative is bad and positive is good. Typically, a Sharpe Ratio of 1 is considered a good investment while a 3 represents an amazing investment opportunity (Maverick, 2015)
Through the use of the Efficient Frontier Method and Sharpe Ratio, our app tells users the appropriate allocation percentages to each stock within a selected portfolio to maximize returns while also giving them details on stock historical performance.

References

- "What is a Stock?", Money. 2017. Retrieved from http://time.com/money/collection-post/2791998/what-is-a-stock/
- Fitzsimmons, C. 2009. "Why Do COmpanies Sell Stocks?". Sapling. Retrieved from https://www.sapling.com/5149719/do-companies-sell-stocks
- "Sharpe Ratio". Investing Answers. 2017. Retrieved from http://www.investinganswers.com/financial-dictionary/ratio-analysis/sharpe-ratio-4947
- "Sharpe Ratio". Investopedia. 2017. Retrieved from http://www.investopedia.com/terms/s/sharperatio.asp
- Aw, E. 2017. "Find The Highest Returns With The Sharpe Ratio". Investopedia. Retrieved from http://www.investopedia.com/articles/07/sharpe_mpt.asp
- "Efficient Frontier". Investopedia. 2017. Retrieved from http://www.investopedia.com/terms/e/efficientfrontier.asp
- Maverick, J. 2015. "What is a good Sharpe ratio?". Investopedia. Retrieved from
- http://www.investopedia.com/ask/answers/010815/what-good-sharpe-ratio.asp
- "Sharpe Ratio Range of Possible Values". Macroption. 2017. Retrieved from
- http://www.macroption.com/sharpe-ratio-range/
- "The Basics of Bollinger Bands". Investopedia. 2015. Retrieved from http://www.investopedia.com/articles/technical/102201.asp
- Mitchell, C. 2017. "How to Use Volume To Improve Your Trading". Investopedia. Retrieved from http://www.investopedia.com/articles/technical/02/010702.asp
- "How to Interpret Volume on a Stock Chart". Swing-Trade-Stocks. 2017. Retrieved from http://www.swing-trade-stocks.com/stock-chart-volume.html
- "Moving Average Convergence Divergence-MACD". Investopedia. 2017. Retrieved from http://www.investopedia.com/terms/m/macd.asp
- "Mean-Variance Portfolio Optimization with R and Quadratic Programming". 2012. Retrieved from http://www.wdiam.com/2012/06/10/mean-variance-portfolio-optimization-with-r-and-quadratic-programming/
- "Risk and Return: Expected Return". PreMBA finance. 2000. Retrieved from http://ci.columbia.edu/ci/premba_test/c0332/s6/s6_3.html

Code References

- https://www.r-bloggers.com/a-gentle-introduction-to-finance-using-r-efficient-frontier-and-capm-part-1/
- http://www.capitalspectator.com/portfolio-analysis-in-r-part-vi-risk-contribution-analysis/
- http://shiny.rstudio.com/gallery/tabsets.html
- https://www.showmeshiny.com/category/visualization/ggplot2/
- https://moderndata.plot.ly/portfolio-optimization-using-r-and-plotly/

Appendix

stockPortfolio	Download stock data, build single index, constant correlation, and multigroup models, and estimate optimal stock portfolios. Plotting functions for the portfolio possibilities curve and portfolio cloud are included. A function to test a portfolio on a data set is also provided.
broom	The broom package takes the messy output of built-in functions in R, such as lm, nls, or t.test, and turns them into tidy data frames.
shinyjs	It lets you perform common useful JavaScript operations in Shiny applications
magrittr	to decrease development time and to improve readability and maintainability of code
shiny	used for building interactive web applications with R
ggplot2	A graphing package used on top of the R statistical package to create elegant visualisations in R.
quantmod	An R package to manage the quantitative financial modelling workflow.
xts	Provide for uniform handling of R's different time-based data classes by extending the time package in R, maximizing native format information preservation and allowing for user level customization and extension, while simplifying cross-class interoperability.
lubridate	It is an R package that makes it easier to work with dates and times
forcats	It provides different tools for working with categorical variables
rvest	It is a package that makes it easy to scrape (or harvest) data from html web pages
tidyverse	It is a set of packages that work in harmony because they share common data representations and 'API' design. This package is designed to make it easy to install and load multiple 'tidyverse' packages in a single step
stringr	The four main functions achieved with this package are - Character manipulation: these functions allow you to manipulate the individual characters inside the strings inside character vectors. Whitespace tools to add, remove, and manipulation whitespace. Locale sensitive operation whose operation will vary for locale to locale. Pattern matching functions. These recognize four engines of pattern description
corrplot	The corrplot package is a graphical display of a correlation matrix, confidence interval. It also contains some algorithms to do matrix reordering.
plotly	Plotly is an R package for creating interactive web-based graphs via the open source JavaScript graphing library plotly.js
quadprog	This package contains routines and documentation for solving quadratic programming problems

DT	The R package DT provides an R interface to the JavaScript library DataTables
dplyr	It is focused on important data manipulation tools for working with data frames, with two other goals - providing blazing fast performance for in-memory data by writing key pieces in C++, and using the same interface to work with data no matter where it's stored, whether in a data frame, a data table or database
devtools	Collection of package development tools.
Shinysky	Contains a collection of Shiny UI components/widgets that are not a part of standard Shiny.