Mean-Variance Portfolio Optimization

ISDS 570

Professor Dr. Pawel J. Kalczynski

Team 7

Bangad Purva

Girikumar Harikrishnan

Gudapati Venkata Vivek

McKay Jason

Sonawane Sameet

Table of Contents

ABSTRACT

The report concentrates on the Portfolio Optimization of 15 different tickers obtained from Quandl Wiki. The SP500TR data mentioned in the report is obtained from Yahoo Finance. The data is then imported into PostgreSQL for the further ETL process. A custom calendar is created and imported for recognizing market working days. Portfolio Optimization is conducted in two steps, the first one is assets classes' weight improvement, and for the second one weights are streamlined for resources in the related resource class.

The report explains which software is used, the ETL process, tickers used for evaluation, evaluation methodologies, and conclusion. Graphical content in the information includes optimized weights of the ticker, comparison between SP500TR and portfolio returns, portfolio ticker weights and cumulative returns of both SP500TR and the portfolio tickers.

INTRODUCTION

The stock market is where people can make their fortune by correctly predicting the stock price and making revenue out of it. So, the stock price prediction is very important for the investors. The stock market is highly volatile, and therefore, it is challenging to make predictions of a given stock. It is also known as stock market forecasting, where we determine its potential value for the next few months or years. There is no proper method or way to accurately predict the stock price as the stock price depends on the demand and supply ratio. There are two approaches we can use to predict stock price:

Fundamental Analysis

It is a method used to determine your stock's values by studying the conditions of the market. FA involves both macroeconomic and microeconomic. Macroeconomics considers the state of the economy and all other factors which can affect the stock financially. In comparison, microeconomic includes studying the company’s management.

Technical Analysis

Technical analysis is entirely in contrast to fundamental analysis as the analyst studies the charts of stock by visualizing its price and historical movements of the stock.

In this project we are going to focus on 15 stocks ticker which are taken from NYSE the tickers are 'B', 'BAC', 'BAH', 'GABC', 'GLAD', 'GPRE', 'GOOD', 'GOLD', 'GORO', 'MCK', 'MDC', 'MCY', 'SHOO', 'SON', 'STAR'. We have chosen these tickers based on the last name of all the teammates. We will predict three months of stocks based on past data from 2013 to 2108. Initially, we started by extracting data for the following tickers. And Analyze the future price for the stocks.

PROBLEM STATEMENT

The optimization of SP500TR ticker's weights so that it will be evident that which investment plans can give the maximum yearly ROI from the portfolio.

SOFTWARE

RStudio is used for creating the portfolio and to train and test the model and PostgreSQL is used for processing the data i.e., creating the tables from the extracted data and then importing the created CSV file to build the portfolio. Clear description of both the database and tool is given below.

PostgreSQL

PostgreSQL is an open-source database system used for creating and storing of the database and database tables. Here, we created a database named “stock_market” with the tables named ‘eod_quotes’, ‘eod_indices’, and custom calendar. The data extracted from different sources is processed into tables here using PostgreSQL and then a custom table is created using Microsoft Excel which contains all the trading dates. The table named 'eod_quotes', contains the information collected from the Quandl wiki data, which consists of various tickers and the table eod_indices contains the data that is collected from SP500TR from the yahoo finance website from January 1, 2013, to March 31, 2018. Now, Views are created for the tables to obtain data for the specified time period and then we extract the columns like year, month, day, day of the week, and trading (zero across a date implies that it is not a trading day, one suggests it is a trading day) using excel functions and save them in a CSV file which is further used to create a table, and then additional columns like prev_trading_day and eom are added to the custom table.

Before performing the calculations, we create a table named Exclusion which contains the data of the excluded tickers which are not complete i.e., less than 99% and then a View is created by combining both the tables and excluding the tickers which are incomplete. After the View has been executed, we calculate returns using the formula:

$$\text{daily returns} = \frac{\text{eod.adj\_close}}{prev\_eod.adj\_close} - 1$$

we then export the daily prices and daily_retruns data by joining the custom calendar and the materialized view for eod and save them and then a 'stockmarketproject' role is created for this database and provide read rights for existing and future tables.

RStudio

Initially all the required packages are installed and are used for transforming the data that is read using read.csv where we have imported the CSV data from the PostgreSQL. We can also read the data of the custom calendar, eod_quotes, and eod_indices for the period from December 31, 2012 to March 31, 2018 by connecting to PostgreSQL using the role name and 'RPostgreSQL', 'DBI' packages. We, then combine the data from eod_quotes and eod_indices tables and store it in a single data frame eod as they both contain same table format.

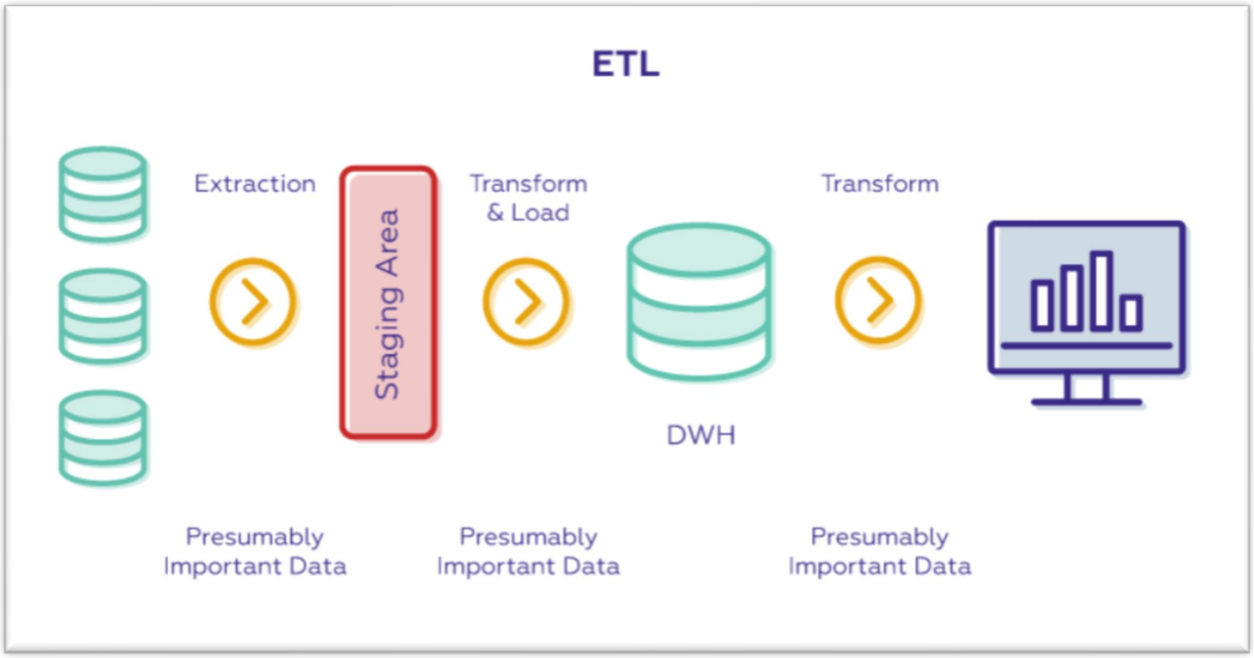
For the effective performance we subset our data using trading days where 99% or more complete data is filtered and stored in eod_complete table. We performing some preprocessing on the data before training the model by using some functions and installing the packages like 'reshape2', where 'dcast' tool is used to pivot the tables. Similarly, 'na.locf' function of 'zoo' package is used to replace NA and NAN values. Functions like 'CalculateReturns' in 'PerformanceAnalytics' package, are used to calculate the returns for every single day and maximum returns are calculated.

After completing all the preprocessing procedures and cleaning the data we create the portfolio by using 15 tickers which are created by using the first letter of our last names. And now, we train the model using the data from 2013 to 2017 as Ra_training and then test the model using the data of first 3 months of 2018 i.e., Jan, Feb, Mar as Rb_training and then we optimize our portfolio and extract the weights.

Now, the benchmark ticker, SP500TR is compared to calculate the annualized returns and then we use 'PerformanceAnalytics' package to convert these vectors to extensible time-series data, which is then used to calculate the annualized and cumulative returns. We then compare our graph with SP500TR, if both the trends are match then we specify our portfolio as the efficient one.

ETL

ETL process is used for collecting the data from different sources and combining them to form a new database which has different context from the original sources. Extract, Transform, Load (ETL) is a process that synthesizes, extracts or copies data from various sources, transforms the data which is loaded in a format different than the sources. A basic ETL structure can be found as shown below:


ETL
Extraction
Staging Area
Transform & Load
DWH
Transform
Presumably Important Data
Presumably Important Data
Presumably Important Data

Data extraction

Data Extraction is the process where the data from different sources is extracted and is processed, so that any corrupted data will not degrade the system performance. Here, we collected the data from the official Yahoo website from the datafile SP500TR which is in the CSV format containing the data from the year 2013 to the first three months of 2018(Jan, Feb and March only). The data is collected from sites like Yahoo Finance, NASDAQ company list and Quandl wiki dataset. Quandl Wiki dataset was utilized for the purpose of tickers. SP500TR was populated using the Quandl Wiki dataset for updated ticker information along with other parameters. The WHERE condition was utilized in order to retrieve accurate data from the tables. A custom calendar was created in order to filter the trading days particularly. The table 'custom_calendar' consisted of the values corresponding to the created custom calendar. Views, materialized view were created in order to fetch returns and reports. A portfolio analysis was created with the help of various data sources as required. A connection was established to access the created 'stock_market' database in PostgreSQL on to R Studios.

R packages PortfolioAnalytics and PerformanceAnalytics help to achieve the secure connection for further analysis.

Data transformation

Transformation is an important process in ETL and helps to receive the desired results. The extracted data from different sources is then sent to Data Transformation where several calculations and transformation functions are applied on the data in order to fill the missing values and form the pivot tables which are further combined to form the custom calendar. And several other transformations are performed to reshape the data and also filter the data to eliminate the extreme values which may cause drastic change in the results. R packages 'reshape2' and 'zoo' helped to deal with transformation and null or missing values respectively. Transformation includes formatting the data in a standard format, cleaning and removing duplicates, provide derived and calculated values. The tickers form the columns and the dates form the rows. The missing values are adjusted with possible next entry with the help of R packages. 'CalculateReturns' a function in PerformanceAnalytcs help to load data and perform optimization on the tickers chosen by the team. Cumulative and annualized return help to improve the portfolio analysis.

Data loading

The created database 'stock_market' undergoes lot of transactions. The extracted data is loaded after the required steps. The read.csv function help to insert values into the database. Before loading into the tables, the format of the CSV file the format of data and header values are checked. The 'company_list' table is joined with 'eod_quotes' and SP500TR with 'eod_indices'. The 'custom_calendar' is updated with the new values.

TICKERS

An abbreviation created to distinctly identify the shares traded of a stock are termed as tickers. Tickers were selected by the first letter of the last name of each team member. Since the team consists of 5 members, a total of 15 tickers were selected as follows:

Letter for ticker	**Last name**
B	Bangad
G	Girikumar
G	Gudapati
M	McKay
S	Sonawane

TICKER	**COMPANY**	**SECTOR**
B	BARNES GROUP INC	Producer Manufacturing Industrial Machinery
BAC	BANK OF AMERICA CORPORATION	Finance Major Banks
BAH	BOOZ ALLEN HAMILTON HLDG CORP	Commercial Services

		Miscellaneous Commercial Services
GLAD	GLADSTONE CAPITAL CORP	Finance Investment Managers
GOOD	GLADSTONE COMMERCIAL CORP	Finance Real Estate Investment Trusts
GOLD	BARRICK GOLD CORPORATION	CFDs Metals
GABC	GERMAN AMERN BANCORP	Finance Regional Banks
GORO	GOLD RESOURCE CORPORATION	Non-Energy Minerals Precious Metals
GPRE	GREEN PLAINS INC	Process Industries Chemicals: Specialty
MCK	MCKESSON CORPORATION	Distribution Services Medical Distributors
MCY	MERCURY GENERAL CORP	Finance Property/Casualty Insurance
MDC	M.D.C. HLDGS INC	Consumer Durables Homebuilding
SHOO	MADDEN(STEVEN)	Consumer Non-Durables Apparel/Footwear
SON	SONOCO PRODUCTS CO	Process Industries Containers/Packaging
STAR	ISTAR INC	Finance Real Estate Investment Trusts

EVALUATION

1. *Cumulative return chart for 2013-2017 for stock tickers selected by your team*

Figure 1 shows the tends of cumulative returns of all the tickers specified in the specified period. X- axis specifies the time period where as the y- axis specifies the cumulative return

2. *Weights of your optimized portfolio (four digits precision) and the sum of these weights*

Figure 2 shows the optimized weights of the portfolio for all the 15 tickers along with the sum of the weights which are optimized

3. *Cumulative return chart for your optimized portfolio and SP500TR index for available 2018 data (approximately 3 months)*

Figure 3 shows the trends of cumulative return of optimized portfolio and the SP500TR for the testing data. The data of the year 2018. Here X-axis refers to the time period and Y-axis to the cumulative return.

4. *Annualized returns for your portfolio and SP500TR index for available 2018 data (approximately 3 months ending on Mar. 27th, 2018)*

Figure 4 shows the table of the Annualized returns and cumulative returns that are calculated from the data given i.e., the data from 2013 to 2017 and the basic metrics are compared for SP500TR and ptf data.

5. *One paragraph comment on the performance of your portfolio against the index.*

Based on the chart in Figure 3, it is easy to see that the portfolio performed worse than the index for all of the available date range. Both curves shared a similar shape, with the index maintaining positive cumulative returns for most of the examined time frame. Unfortunately, the portfolio experienced the opposite, where most of the cumulative returns were negative. Based on Figure 1, most individual indexes at least broke even in cumulative returns, so perhaps if the portfolio dropped and replaced individual unsuccessful tickers, the portfolio could close its performance gap against the index. For example, in Figure 1, the 'GORO' performed the worst for the entirety of the examined four-year period and should be the first to get replaced. The portfolio needs many adjustments before outperforming the index, but matching it in the present doesn't seem unrealistic.

CONCLUSION

For our ETL and evaluation, we relied on standard applications RStudio and PostgreSQL and we prepared our datasets by relying on entries with less than 1% missing data. We chose 15 separate tickers based on the name requirement, but didn’t pre-examine all valid tickers for each name for performance. After examining the returns, most of the tickers returned neutral or positive results, but not enough to outperform the index. Significant adjustments are needed to first bring the portfolio out of consistently accumulating losses, then catch up in returns to the index.

APPENDIX

Appendix 1: Initialization and Query Creation and Execution

```
#StockMarket Prediction
rm(list=ls(all=T)) # this just removes everything from memory
#install.packages("RPostgreSQL")
require(RPostgreSQL)
pg = dbDriver("PostgreSQL")
conn = dbConnect(drv=pg
          ,user="stockmarketreader"
          ,password="read123"
          ,host="localhost"
          ,port=5432
          ,dbname="stock_market"
)
# Query for custom calendar
qry="SELECT * FROM custom_calendar where date BETWEEN '2012-12-30' AND '2018-03-
27' ORDER by date"
ccal<-dbGetQuery(conn,qry)
#eod prices and indices
qry1="SELECT symbol,date,adj_close FROM eod_indices WHERE date BETWEEN '2012-12-
30' AND '2018-03-27'"
qry2="SELECT ticker,date,adj_close FROM eod_quotes WHERE date BETWEEN '2012-12-
30' AND '2018-03-27'"
eod<-dbGetQuery(conn,paste(qry1,'UNION',qry2))
dbDisconnect(conn)
```

Appendix 2: Calendar Creation

```r
#For monthly we may need one more data item (for 2011-12-30)
#We can add it to the database (INSERT INTO) - but to practice:
eod_row<-data.frame(symbol='SP500TR',date=as.Date('2012-12-30'),adj_close=2505.44)
eod<-rbind(eod,eod_row)
tail(eod); dim(eod)
tdays<-ccal[which(ccal$trading==1),,drop=F]
head(tdays)
nrow(tdays)-1
pct<-table(eod$symbol)/(nrow(tdays)-1)
selected_symbols_daily<-names(pct)[which(pct>=0.99)]
eod_complete<-eod[which(eod$symbol %in% selected_symbols_daily),,drop=F]
require(reshape2)
eod_pvt<-
dcast(eod_complete, date ~ symbol,value.var='adj_close',fun.aggregate = mean, fill=NULL)
eod_pvt[1:10,1:5]
# Merge with Calendar
eod_pvt_complete<-merge.data.frame(x=tdays[,'date',drop=F],y=eod_pvt,by='date',all.x=T)
#use dates as row names and remove the date column
rownames(eod_pvt_complete)<-eod_pvt_complete$date
eod_pvt_complete$date<-NULL
```

Appendix 3: Calculating Returns for tickers

```
#Stock Market Forecasting and Optimization 14
eod_pvt_complete<-na.locf(eod_pvt_complete,na.rm=F,fromLast=F,maxgap=3)
require(PerformanceAnalytics)
eod_ret<-CalculateReturns(eod_pvt_complete)
eod_ret[1:10,1:4]
colMax <- function(data) sapply(data, max, na.rm = TRUE)
max_daily_ret<-colMax(eod_ret)
max_daily_ret[1:10] #first 10 max returns
selected_symbols_daily<-names(max_daily_ret)[which(max_daily_ret<=1.00)]
length(selected_symbols_daily)
eod_ret<-eod_ret[,which(colnames(eod_ret) %in% selected_symbols_daily)]
eod_ret[1:10,1:4]
#first 10 rows and first 4 columns
#Stocks selected according to the Last Name of Group Members
inds <-
 which(names(eod_ret) %in% c('B','BAC','BAH','GABC','GLAD','GPRE','GOOD','GOLD','GOR
O','MCK','MDC','MCY','SHOO','SON','STAR'))
Ra<-as.xts(eod_ret[,inds,drop=F])
Rb<-as.xts(eod_ret[,'SP500TR',drop=F]) #benchmark
table.AnnualizedReturns(cbind(Rb,Ra),scale=252)
acc_Ra<-Return.cumulative(Ra)
acc_Rb<-Return.cumulative(Rb)
chart.CumReturns(Ra,legend.loc = 'topleft')
chart.CumReturns(Rb,legend.loc = 'topleft')
chart.CumReturns(cbind(Ra,Rb,legend.loc),legend.loc = 'topleft')
# withold the last 61 trading days
Ra_training<-head(Ra,-59)
Rb_training<-head(Rb,-59)
chart.CumReturns(Ra_training,legend.loc='left')
```

Figure 1: Cumulative returns for the years from 2013 to 2017 for the selected tickers

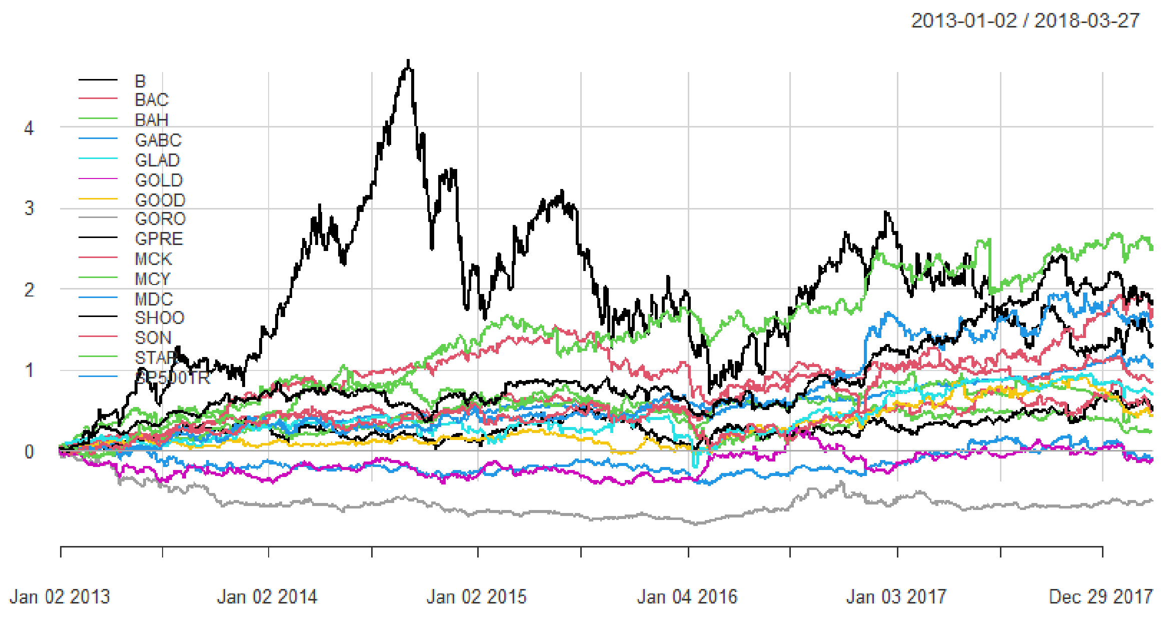
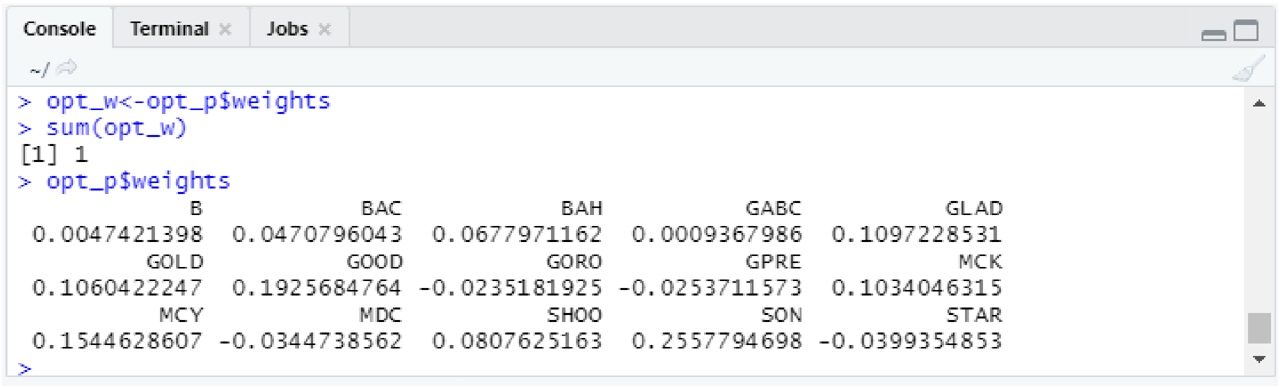


Figure 2: Optimized weights and the sum



```
> opt_w<-opt_p$weights
> sum(opt_w)
[1] 1
> opt_p$weights
            B           BAC           BAH          GABC          GLAD 
 0.0047421398  0.0470796043  0.0677971162  0.0009367986  0.1097228531 
         GOLD          GOOD          GORO          GPRE           MCK 
 0.1060422247  0.1925684764 -0.0235181925 -0.0253711573  0.1034046315 
          MCY           MDC          SHOO           SON          STAR 
 0.1544628607 -0.0344738562  0.0807625163  0.2557794698 -0.0399354853 
>
```

Appendix 4: Portfolio Optimization

```
#Stock Market Forecasting and Optimization 15
# use the last 61 trading days for testing
Ra_testing<-tail(Ra,59)
Rb_testing<-tail(Rb,59)
chart.CumReturns(cbind(Ra_testing, Rb_testing), legend.loc='left')
table.AnnualizedReturns(Rb_training)
mar<-mean(Rb_training) #we need daily minimum acceptabe return
#optimize the MV portfolio weights based on training
require(PortfolioAnalytics)
require(ROI.plugin.quadprog)
pspec<-portfolio.spec(assets=colnames(Ra_training))
pspec<-add.objective(portfolio=pspec,type="risk",name='StdDev')
pspec<-add.constraint(portfolio=pspec,type="full_investment")
pspec<-add.constraint(portfolio=pspec,type="return",return_target=mar)
#optimize portfolio
opt_p<-optimize.portfolio(R=Ra_training, portfolio=pspec, optimize_method = 'ROI')
#extract weights
opt_w<-opt_p$weights
sum(opt_w)
#apply weights to test returns
Rp<-Rb_testing # easier to apply the existing structure
#define new column that is the dot product of the two vectors
Rp$ptf<-Ra_testing %*% opt_w
#Stock Market Forecasting and Optimization 16
#check Rp
head(Rp)
tail(Rp)
#Compare basic metrics
table.AnnualizedReturns(Rp)
acc_Rp<- Return.cumulative(Rp)
acc_Rp #Cumulative Return
```

Figure 3: Cumulative returns of the optimized portfolio and SP500TR of Jan. Feb and March 2018

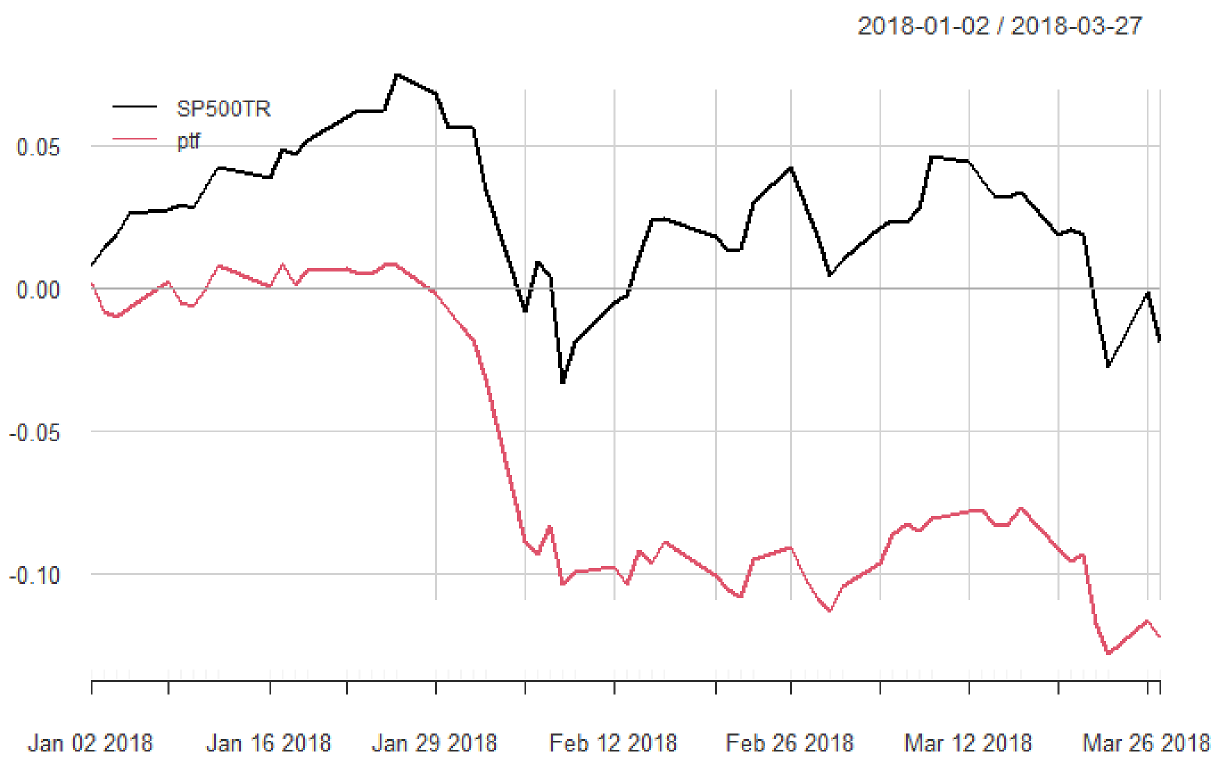
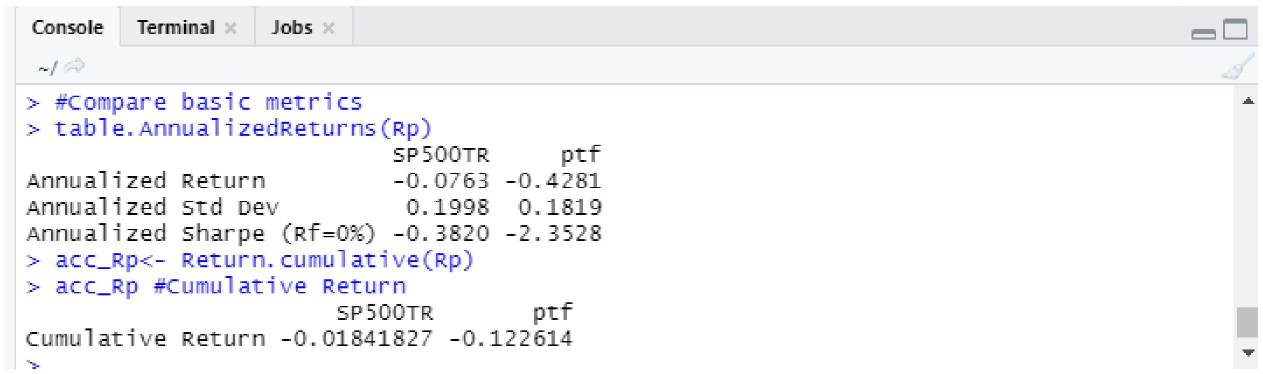


Figure 4: Annual returns of the portfolio and SP500TR for 2018 data



```
> #Compare basic metrics
> table.AnnualizedReturns(Rp)
                          SP500TR     ptf
Annualized Return         -0.0763 -0.4281
Annualized Std Dev         0.1998  0.1819
Annualized Sharpe (Rf=0%) -0.3820 -2.3528
> acc_Rp<- Return.cumulative(Rp)
> acc_Rp #Cumulative Return
                      SP500TR       ptf
Cumulative Return -0.01841827 -0.122614
>
```