Research Project

On

Stock Market Analysis

By

Bhakti Sangoi

Sagar Chettiyar

DA5020 35665 Collect/Store/Retrieve Data

Spring 2018

Point of Contact

Bhakti Sangoi

Email: sangoi.b@husky.neu.edu

Phone: +1 (857) 389-5956

Sagar Chettiyar

Email: chettiyar.s@husky.neu.edu

Phone: +1 (617) 407-9130

Submitted on: April 22, 2018

Instructor: Prof. Kathleen Durant

Table of Contents

1. Introduction

Stocks:

A stock is a type of ownership in a corporation. It represents the claim on the company's earnings and assets. More you buy stocks of a company, more ownership stake in the company becomes greater.

Stock Market:

Stock Market is a place where bonds, securities and shares/stocks of public listed companies are traded. The main function of the stock market is to process the transaction between buyers and sellers. Buyers/Sellers can track the price change of stocks through the stock market.

One of the many things that people believe is that there is only one stock market as the name suggests but that's not true, because there are multiple other stock markets.

How do Stock Market works?

The stock market is consisting of different exchanges such as New York Stock Exchange(NYSE), National Association of Securities Dealers Automated Quotation System (Nasdaq) and American Stock Exchange (AMEX). Stocks/Shares are entered on one of these exchanges. Buyers and Sellers come together to the market to exchange stocks. Individuals typically buy and sell between one another and there is an auction occurring; i.e. the highest bidding price is matched with the lowest asking price.

NYSE and Nasdaq:

New York Stock Exchange (NYSE) is the world's largest and the oldest stock exchange and is located on Wall Street in New York City. It was founded in 1792. 2800 companies are listed under NYSE with trading 1.46 billion shares each day. Nasdaq is 46 years young. There are 3800 listings with the market capitalization of $11 trillion. NYSE and Nasdaq do the maximum business in the stock market.

Trading starts at 9:30 am to 4:00 pm EST from Monday to Friday for both the exchanges. The only difference between NYSE and Nasdaq is the way stocks are exchanged between buyers and sellers. Nasdaq is Dealer's market whereas NYSE is auction market.

2. Aim

Seemingly endless data is available for Stock market. It is difficult to collect a large dataset of stock prices which is structured, cleaned and has high granularity. Analysing stock prices helps an intelligent trader to invest wisely in the stock market. Thereby we provide a dataset which is trying to cover the 6V's of big data: Volume, Volatility, Veracity, Variety, Velocity, and Validity.

2.1 Objective

The Objective of this project is:

- **Collect** and clean the high-volume dataset for the year 2013-2016 on both daily and annual basis.
- Once the data is cleaned, create a schema with a variety of data and store it in the database. We are using SQL database to **store** the data.
- After the data is in the structured format, **retrieve** data using SQL queries.
- Perform Data Analysis using R programming

- Additionally, perform Machine Learning Algorithm on highly volatile data
- Create a Web App using R Shiny

2.2 Motivation

We both come from an IT domain with experience and a strong interest in Finance industry and Investment Management. Also, we plan to deal in the stock market as we all know that stock market has been on a bit of a tear. Is it a great time to buy stocks? Let's have a look.

3. Data Collection and Data Cleaning

We are collecting data from two sources. We followed the extraction with the manipulation phase on both the sources.

Source 1: S&P Dow Jones Indices

- S&P Dow Jones Indices is maintained by S&P Global company and we are using this data because it covers almost 80% of American equity market by capitalization.
- This data source proved to be a wise choice as it helped in exhibiting interpretation from raw data and extract the required data for changing and to improve on.

Source 2: Kaggle NYSE and NASDAQ data

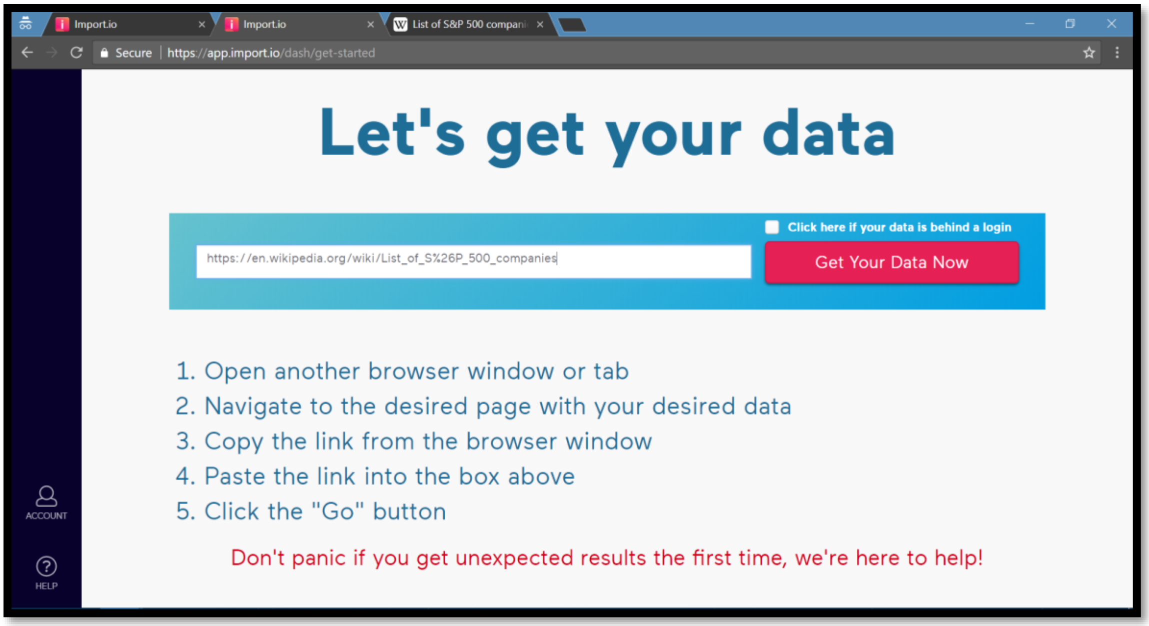
- This dataset has raw, as-in daily prices of data spans from 2010 to the end of 2016.
- This dataset consists of high level of granularity which is enough to derive some other fundamental indicators.

Source 3: Yahoo! Finance

- Data from Yahoo Finance is used to generate Dynamic Time Series analysis. Yahoo Finance provides live and raw data to perform analysis and further explain interpretations.

Step 1: Scraping the S&P Dow Jones Indices website

i) We used scraping techniques for Dow Jones Index data. Using Import.io data scrapping tool, we collected the necessary data for 500 companies required for this project.



https://app.import.io/dash/get-started

Let's get your data

□ Click here if your data is behind a login

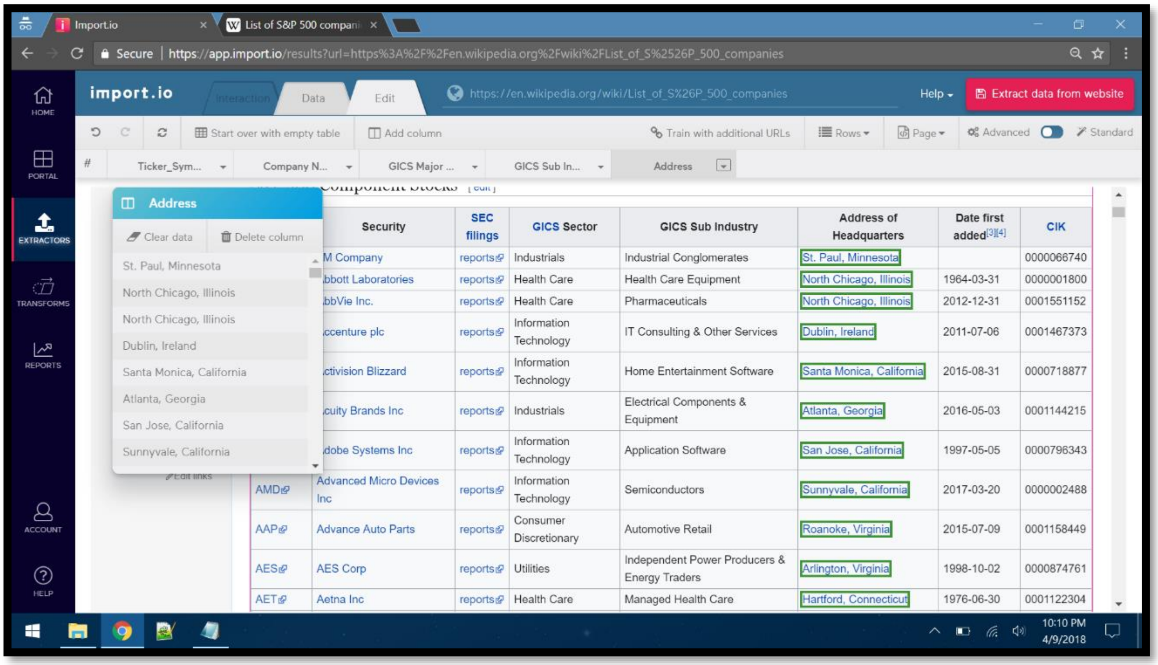
https://en.wikipedia.org/wiki/List_of_S%26P_500_companies

Get Your Data Now

1. Open another browser window or tab
2. Navigate to the desired page with your desired data
3. Copy the link from the browser window
4. Paste the link into the box above
5. Click the "Go" button

Don't panic if you get unexpected results the first time, we're here to help!

ii) Selected the required columns needed for storing in the schema. It helps to reduce the process time for loading the data into the database system.



https://app.import.io/results?url=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FList_of_S%2526P_500_companies

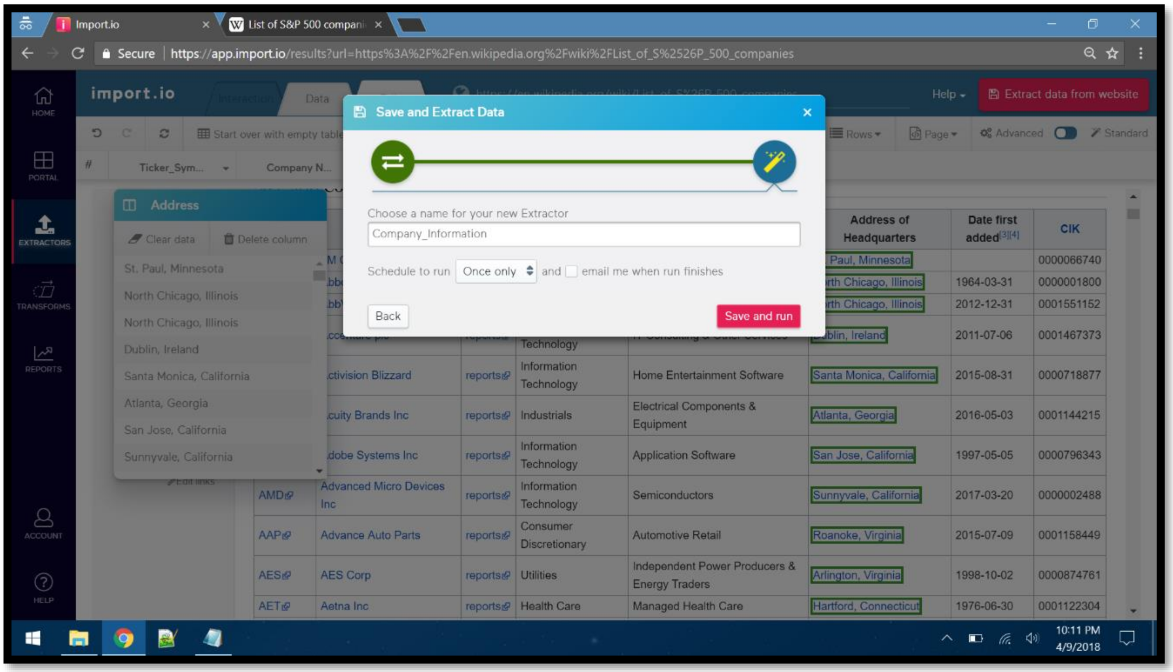
https://en.wikipedia.org/wiki/List_of_S%26P_500_companies

Columns: # | Ticker_Sym... | Company N... | GICS Major ... | GICS Sub In... | Address

Address: St. Paul, Minnesota; North Chicago, Illinois; North Chicago, Illinois; Dublin, Ireland; Santa Monica, California; Atlanta, Georgia; San Jose, California; Sunnyvale, California

	Security	SEC filings	GICS Sector	GICS Sub Industry	Address of Headquarters	Date first added[3][4]	CIK
	M Company	reports	Industrials	Industrial Conglomerates	St. Paul, Minnesota		0000066740
	bbott Laboratories	reports	Health Care	Health Care Equipment	North Chicago, Illinois	1964-03-31	0000001800
	bbVie Inc.	reports	Health Care	Pharmaceuticals	North Chicago, Illinois	2012-12-31	0001551152
	ccenture plc	reports	Information Technology	IT Consulting & Other Services	Dublin, Ireland	2011-07-06	0001467373
	ctivision Blizzard	reports	Information Technology	Home Entertainment Software	Santa Monica, California	2015-08-31	0000718877
	cuity Brands Inc	reports	Industrials	Electrical Components & Equipment	Atlanta, Georgia	2016-05-03	0001144215
	dobe Systems Inc	reports	Information Technology	Application Software	San Jose, California	1997-05-05	0000796343
AMD	Advanced Micro Devices Inc	reports	Information Technology	Semiconductors	Sunnyvale, California	2017-03-20	0000002488
AAP	Advance Auto Parts	reports	Consumer Discretionary	Automotive Retail	Roanoke, Virginia	2015-07-09	0001158449
AES	AES Corp	reports	Utilities	Independent Power Producers & Energy Traders	Arlington, Virginia	1998-10-02	0000874761
AET	Aetna Inc	reports	Health Care	Managed Health Care	Hartford, Connecticut	1976-06-30	0001122304

iii) Saving the data in the .csv format.

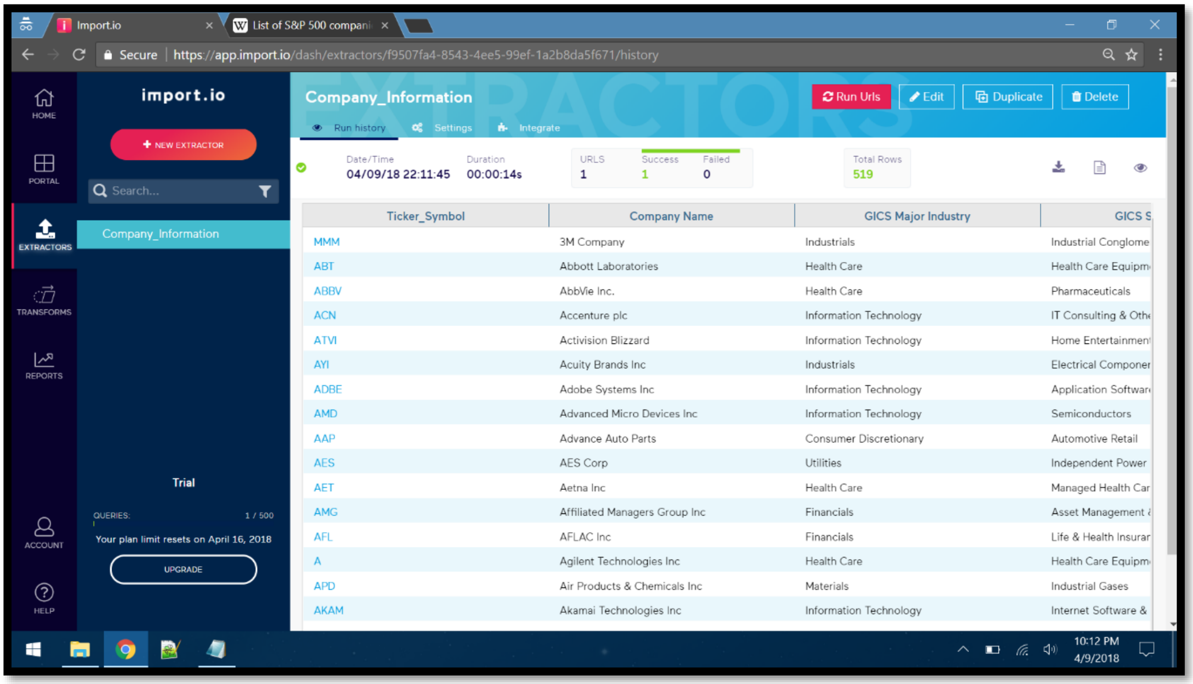


Save and Extract Data

Choose a name for your new Extractor: Company_Information

Schedule to run: Once only and □ email me when run finishes

iv) The final output of the Scrapping. This dataset has 519 rows and 7 columns.



https://app.import.io/dash/extractors/f9507fa4-8543-4ee5-99ef-1a2b8da5f671/history

Company_Information

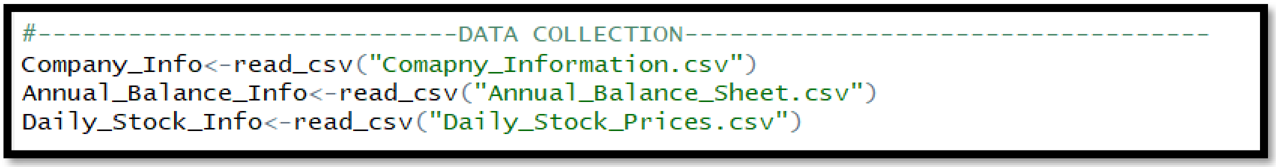
Date/Time	Duration	URLS	Success	Failed	Total Rows
04/09/18 22:11:45	00:00:14s	1	1	0	519

Ticker_Symbol	Company Name	GICS Major Industry	GICS S
MMM	3M Company	Industrials	Industrial Conglome
ABT	Abbott Laboratories	Health Care	Health Care Equipm
ABBV	AbbVie Inc.	Health Care	Pharmaceuticals
ACN	Accenture plc	Information Technology	IT Consulting & Othe
ATVI	Activision Blizzard	Information Technology	Home Entertainment
AYI	Acuity Brands Inc	Industrials	Electrical Componer
ADBE	Adobe Systems Inc	Information Technology	Application Softwar
AMD	Advanced Micro Devices Inc	Information Technology	Semiconductors
AAP	Advance Auto Parts	Consumer Discretionary	Automotive Retail
AES	AES Corp	Utilities	Independent Power
AET	Aetna Inc	Health Care	Managed Health Car
AMG	Affiliated Managers Group Inc	Financials	Asset Management
AFL	AFLAC Inc	Financials	Life & Health Insurar
A	Agilent Technologies Inc	Health Care	Health Care Equipm
APD	Air Products & Chemicals Inc	Materials	Industrial Gases
AKAM	Akamai Technologies Inc	Information Technology	Internet Software &

Step 2: Collecting Stocks data

We have collected the stocks data for the listed 500 companies from source 1 for both daily basis and annual basis. We planned to work with stocks coming from both the exchanges i.e. NYSE and NASDAQ because it will help in performing analysis on a huge dataset. Also, this will help us give an accurate result for forecasting and prediction algorithms.

The daily stocks dataset file contains 851013 rows and annual stocks file contains 1781.



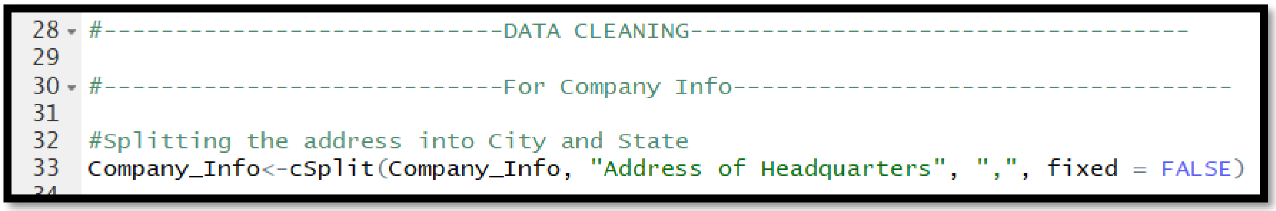
```
#----------------------------DATA COLLECTION-----------------------------------
Company_Info<-read_csv("Comapny_Information.csv")
Annual_Balance_Info<-read_csv("Annual_Balance_Sheet.csv")
Daily_Stock_Info<-read_csv("Daily_Stock_Prices.csv")
```

Step 3: Data Cleaning

A lot of time and efforts was put on data cleaning.

i) Splitting the address into City and State

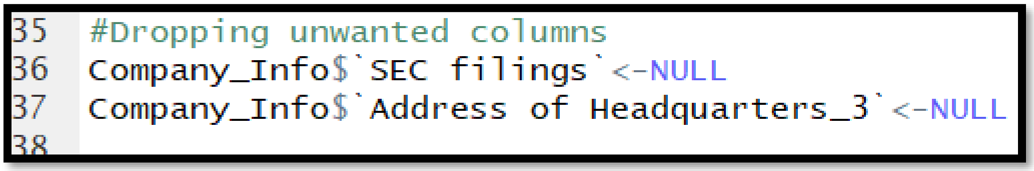
Splitting the City and State from Headquarters address for better analysis using the cSplit function



```
#----------------------------DATA CLEANING-----------------------------------

#----------------------------For Company Info-----------------------------------

#Splitting the address into City and State
Company_Info<-cSplit(Company_Info, "Address of Headquarters", ",", fixed = FALSE)
```

ii) Removing unwanted columns

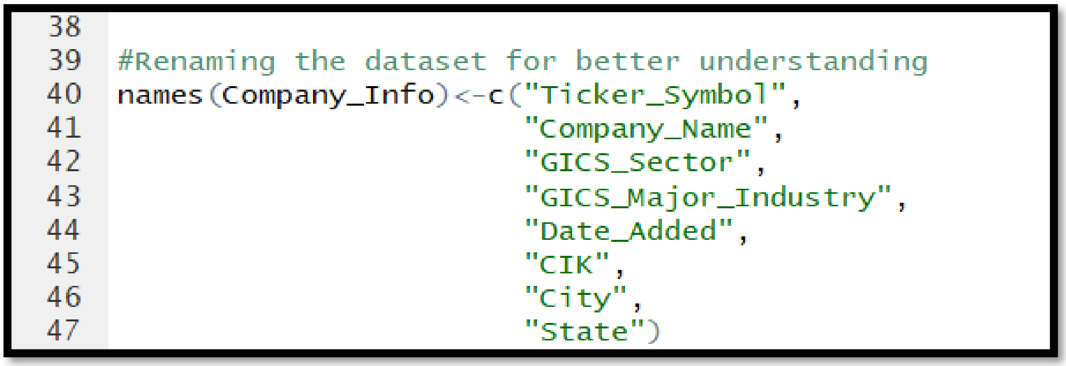
Removing all the unwanted columns such as SEC filings.



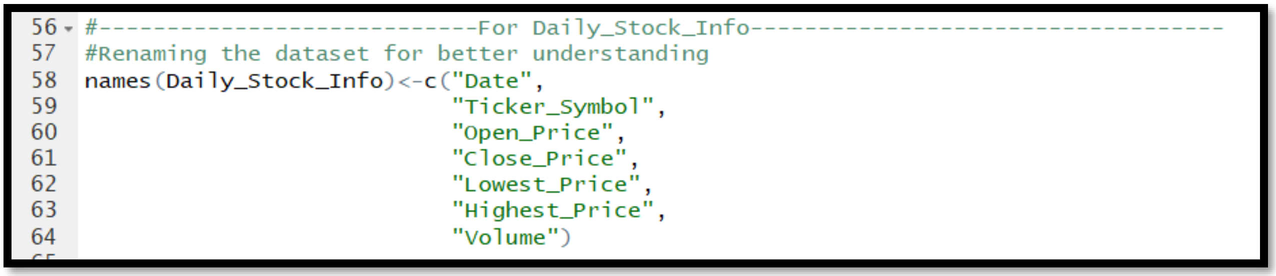
```
#Dropping unwanted columns
Company_Info$`SEC filings`<-NULL
Company_Info$`Address of Headquarters_3`<-NULL
```

iii) Renaming the column names

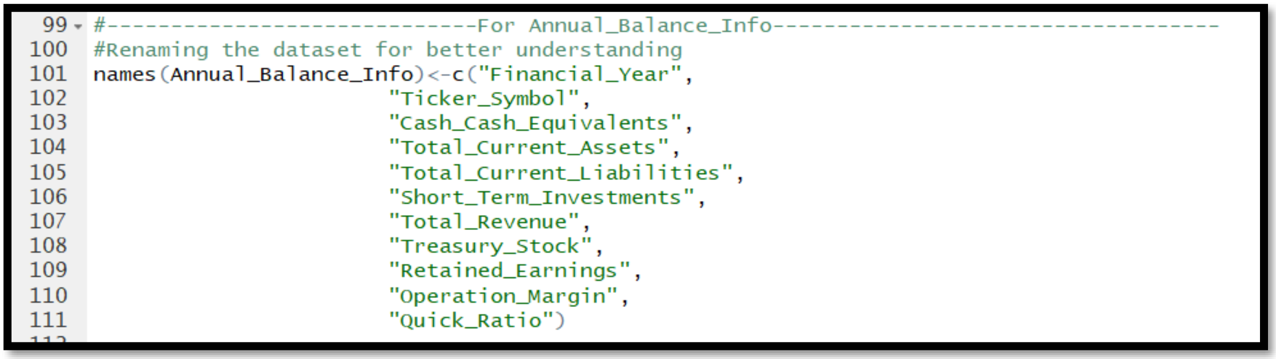
Renaming the columns names as per database naming schema for easy understanding.



```

#Renaming the dataset for better understanding
names(Company_Info)<-c("Ticker_Symbol",
                       "Company_Name",
                       "GICS_Sector",
                       "GICS_Major_Industry",
                       "Date_Added",
                       "CIK",
                       "City",
                       "State")
```



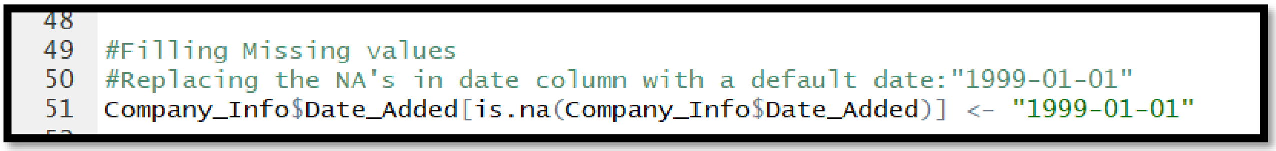
```
#----------------------------For Daily_Stock_Info-----------------------------------
#Renaming the dataset for better understanding
names(Daily_Stock_Info)<-c("Date",
                           "Ticker_Symbol",
                           "Open_Price",
                           "Close_Price",
                           "Lowest_Price",
                           "Highest_Price",
                           "Volume")
```



```
#----------------------------For Annual_Balance_Info---------------------------------
#Renaming the dataset for better understanding
names(Annual_Balance_Info)<-c("Financial_Year",
                              "Ticker_Symbol",
                              "Cash_Cash_Equivalents",
                              "Total_Current_Assets",
                              "Total_Current_Liabilities",
                              "Short_Term_Investments",
                              "Total_Revenue",
                              "Treasury_Stock",
                              "Retained_Earnings",
                              "Operation_Margin",
                              "Quick_Ratio")
```

iv) Filling Missing Values in date column by defaulting a date to all the NA's

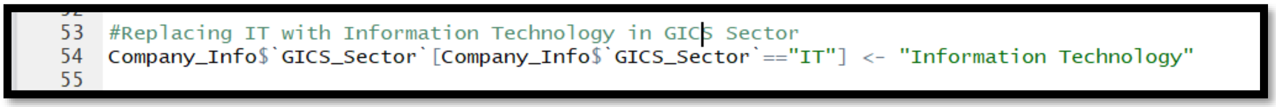
There were some missing dates of company information being added to the database. We thought of giving a default date value to NA's in the Date_Added field.



```

#Filling Missing values
#Replacing the NA's in date column with a default date:"1999-01-01"
Company_Info$Date_Added[is.na(Company_Info$Date_Added)] <- "1999-01-01"
```

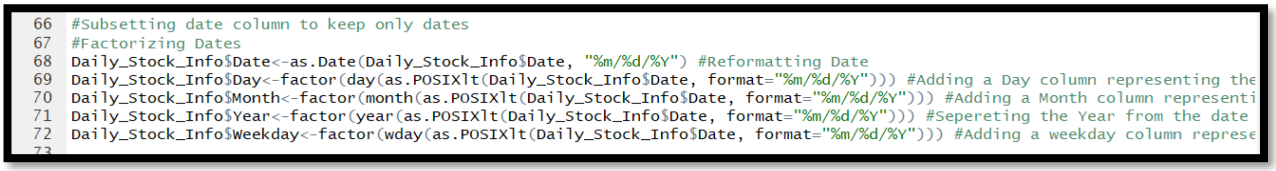
v) Following one standard format for Major Sectors

There are rows which have Major Sector as IT and some have Information Technology. Following one format and converting all of them to Information Technology to get an accurate analysis.



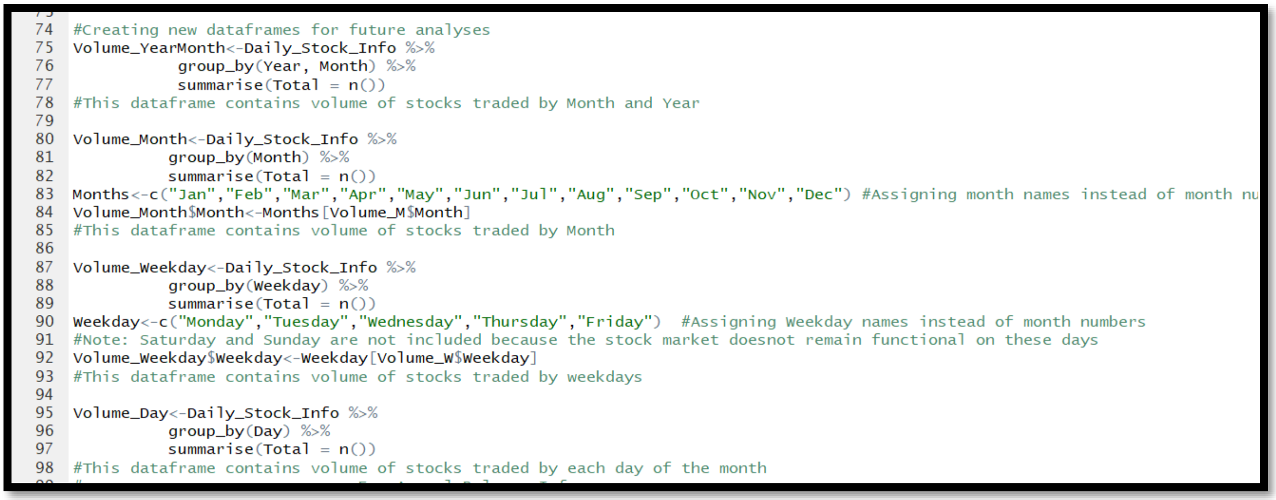
```
#Replacing IT with Information Technology in GICS Sector
Company_Info$`GICS_Sector`[Company_Info$`GICS_Sector`=="IT"] <- "Information Technology"

```

vi) Cleaning the date column, formatting it and factorizing it into different columns for analysis purpose.



```
#Subsetting date column to keep only dates
#Factorizing Dates
Daily_Stock_Info$Date<-as.Date(Daily_Stock_Info$Date, "%m/%d/%Y") #Reformatting Date
Daily_Stock_Info$Day<-factor(day(as.POSIXlt(Daily_Stock_Info$Date, format="%m/%d/%Y"))) #Adding a Day column representing the
Daily_Stock_Info$Month<-factor(month(as.POSIXlt(Daily_Stock_Info$Date, format="%m/%d/%Y"))) #Adding a Month column representi
Daily_Stock_Info$Year<-factor(year(as.POSIXlt(Daily_Stock_Info$Date, format="%m/%d/%Y"))) #Sepereting the Year from the date
Daily_Stock_Info$Weekday<-factor(wday(as.POSIXlt(Daily_Stock_Info$Date, format="%m/%d/%Y"))) #Adding a weekday column represe
```

vii) Creating new data frames volume of stocks traded by month, Year, weekdays and each day of the month respectively.



```
#Creating new dataframes for future analyses
Volume_YearMonth<-Daily_Stock_Info %>%
           group_by(Year, Month) %>%
           summarise(Total = n())
#This dataframe contains volume of stocks traded by Month and Year

Volume_Month<-Daily_Stock_Info %>%
          group_by(Month) %>%
          summarise(Total = n())
Months<-c("Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec") #Assigning month names instead of month nu
Volume_Month$Month<-Months[Volume_M$Month]
#This dataframe contains volume of stocks traded by Month

Volume_Weekday<-Daily_Stock_Info %>%
          group_by(Weekday) %>%
          summarise(Total = n())
Weekday<-c("Monday","Tuesday","Wednesday","Thursday","Friday")  #Assigning Weekday names instead of month numbers
#Note: Saturday and Sunday are not included because the stock market doesnot remain functional on these days
Volume_Weekday$Weekday<-Weekday[Volume_W$Weekday]
#This dataframe contains volume of stocks traded by weekdays

Volume_Day<-Daily_Stock_Info %>%
          group_by(Day) %>%
          summarise(Total = n())
#This dataframe contains volume of stocks traded by each day of the month
```

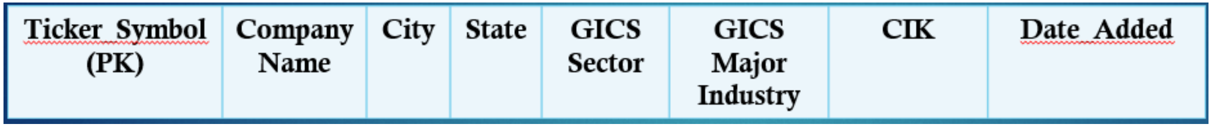
4. Data Storage

Once the data was collected and cleaned, next step was to create the data structure which follows 3rd Normal Form(3NF). 3NF is used in normalizing the database design to reduce the duplication of data and ensuring referential integrity.

We created 3 tables by establishing the connection with the RSQLite package.

Step 1: Creating schema design

1. Company Information
   This table has company information



Ticker_Symbol (PK)	Company Name	City	State	GICS Sector	GICS Major Industry	CIK	Date_Added

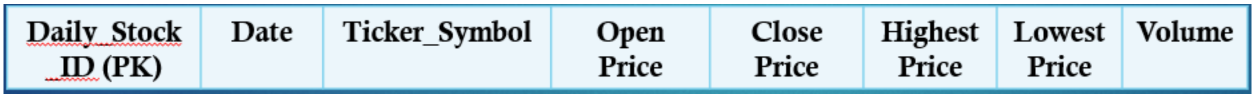
- ○ Ticker_Symbol
  - ▪ Ticker_Symbol is the primary key for Company Information table
  - ▪ It contains companies Stock's Name
  - ▪ Eg: Apple – AAPL, Microsoft – MSFT
- ○ Company Name
  - ▪ It shows the Name of the company a stock belongs to.
- ○ City
  - ▪ It provides the City in which the Headquarter of the company is located.
- ○ State
  - ▪ Contains state in which headquarter of the company is located
- ○ GICS Sector
  - ▪ Provides the main domain of a company.
  - ▪ Eg: Healthcare Industry, Information Technology
- ○ GICS Major Industry
  - ▪ It gives a more detailed information about the GICS Major industry

        - Eg: Application Software, Automotive Retail, Health Care Equipment or Pharmaceuticals
    - Date_Added
        - Date at which the company information was added to the stock market

2. Daily Stock Information

   Step 1: Creating Schema

   This schema has stock information on a daily basis

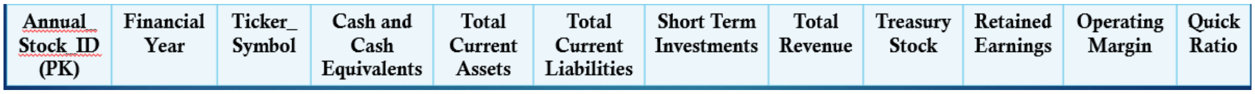


Daily_Stock_ID (PK)	Date	Ticker_Symbol	Open Price	Close Price	Highest Price	Lowest Price	Volume

- Daily_Stock_ID
    - It is a primary key to the Daily Stock Information table
- Date
    - The daily date for which prices are displayed.
- Ticker_Symbol
    - It is a foreign key from the Company Information table (schema 1)
- Open Price
    - The price at which the stock market opens for a day
    - It need not be the same as previous day's closing price
- Close Price
    - The price at which the stock market closes for the day
- Highest Price
    - The highest price of a stock for one day
- Lowest Price
    - The lowest price of a stock for one day
- Volume
    - Total number of stocks traded in one day

3. Balance Sheet

   This table has stock information on an annual basis



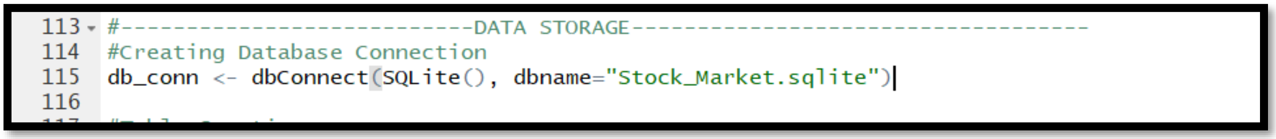
Annual Stock_ID (PK)	Financial Year	Ticker_ Symbol	Cash and Cash Equivalents	Total Current Assets	Total Current Liabilities	Short Term Investments	Total Revenue	Treasury Stock	Retained Earnings	Operating Margin	Quick Ratio

- Annual_Stock_ID
    - It is the primary key for Balance Sheet
- Financial Year
    - Since the balance sheet is calculated annually, this specifies the year for which it was calculated
- Ticker_Symbol:
    - It is a foreign key from Company Information Table (schema 1)
- Cash and Cash Equivalents
    - Value of a company's assets that are cash or can be converted into cash immediately.
    - These include bank accounts, marketable securities, commercial paper, Treasury bills and short-term government bonds with a maturity date of three months or less
- Total Current Assets

  - The total sum of all the available assets of the company for a particular financial year
  - Total Assets and Total Current Assets are different
- Total Current Liabilities
  - The total sum of all the available liabilities of the company for a particular financial year
  - Total liabilities and Total Current liabilities are different
- Short-Term Investments
  - Debt incurred by a company that is due within one year
  - The value of the short-term debt account is very important when determining a company's performance.
  - If the account is larger than the company's cash and cash equivalents, this suggests that the company may be in poor financial health and does not have enough cash to pay off its short-term debts.
- Total Revenue
  - It is the top line or gross income figure from which costs are subtracted to determine net income.
  - Revenue is calculated by multiplying the price at which goods or services are sold by the number of units or amount sold.
- Treasury Stock
  - A portion of shares that a company keeps in its own treasury.
  - Treasury stock may have come from a repurchase or buyback from shareholders, or it may have never been issued to the public in the first place.
  - These shares don't pay dividends, have no voting rights and should not be included in shares outstanding calculations.
- Retained Earnings
  - Retained earnings refer to the percentage of net earnings not paid out as dividends, but retained by the company to be reinvested in its core business, or to pay the debt
  - They can show a positive earnings accumulation or can turn negative and have a deficit if a current period's net loss exceeds the period's beginning retained earnings.
- Operating Margin
  - Operating margin is a measurement of what proportion of a company's revenue is left over after paying for variable costs of production such as wages, raw materials
  - Operating margin gives analysts an idea of how much a company makes on each dollar of sales.
- Quick Ratio
  - The quick ratio is an indicator of a company’s short-term liquidity and measures a company’s ability to meet its short-term obligations with its most liquid assets.
  - Because we're only concerned with the most liquid assets, the ratio excludes inventories from current assets.
  - Quick assets are current assets that can be converted to cash within 90 days or in the short-term
  - If greater than 1, liquid assets can cover for short-term investments
  - If less than 1, the company may not be able to pay off their debts

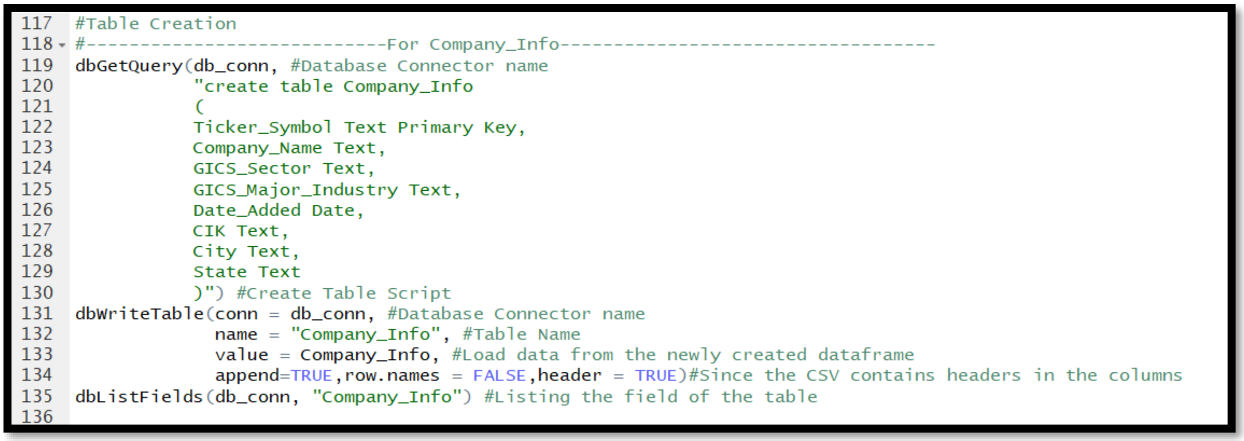
Step 2: Setting up Connection

Initializing Database connection using SQLite.

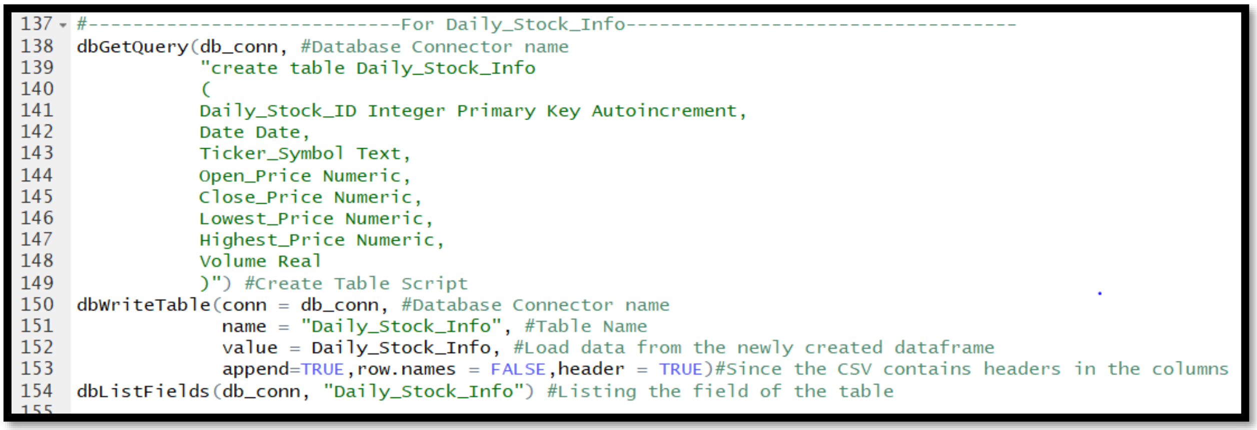


```
#----------------------------DATA STORAGE-----------------------------------
#Creating Database Connection
db_conn <- dbConnect(SQLite(), dbname="Stock_Market.sqlite")

```

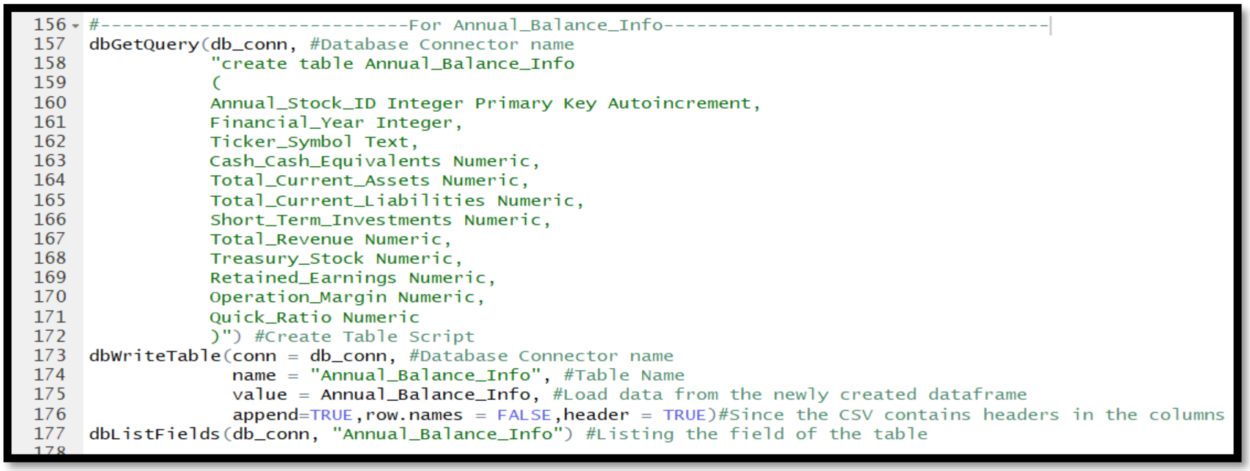
Step 3: Creating Tables in the database



```
#Table Creation
#----------------------------For Company_Info-----------------------------------
dbGetQuery(db_conn, #Database Connector name
           "create table Company_Info
           (
           Ticker_Symbol Text Primary Key,
           Company_Name Text,
           GICS_Sector Text,
           GICS_Major_Industry Text,
           Date_Added Date,
           CIK Text,
           City Text,
           State Text
           )") #Create Table Script
dbWriteTable(conn = db_conn, #Database Connector name
             name = "Company_Info", #Table Name
             value = Company_Info, #Load data from the newly created dataframe
             append=TRUE,row.names = FALSE,header = TRUE)#Since the CSV contains headers in the columns
dbListFields(db_conn, "Company_Info") #Listing the field of the table

```



```
#----------------------------For Daily_Stock_Info-----------------------------------
dbGetQuery(db_conn, #Database Connector name
           "create table Daily_Stock_Info
           (
           Daily_Stock_ID Integer Primary Key Autoincrement,
           Date Date,
           Ticker_Symbol Text,
           Open_Price Numeric,
           Close_Price Numeric,
           Lowest_Price Numeric,
           Highest_Price Numeric,
           Volume Real
           )") #Create Table Script
dbWriteTable(conn = db_conn, #Database Connector name
             name = "Daily_Stock_Info", #Table Name
             value = Daily_Stock_Info, #Load data from the newly created dataframe
             append=TRUE,row.names = FALSE,header = TRUE)#Since the CSV contains headers in the columns
dbListFields(db_conn, "Daily_Stock_Info") #Listing the field of the table
```



```
#----------------------------For Annual_Balance_Info-----------------------------------
dbGetQuery(db_conn, #Database Connector name
           "create table Annual_Balance_Info
           (
           Annual_Stock_ID Integer Primary Key Autoincrement,
           Financial_Year Integer,
           Ticker_Symbol Text,
           Cash_Cash_Equivalents Numeric,
           Total_Current_Assets Numeric,
           Total_Current_Liabilities Numeric,
           Short_Term_Investments Numeric,
           Total_Revenue Numeric,
           Treasury_Stock Numeric,
           Retained_Earnings Numeric,
           Operation_Margin Numeric,
           Quick_Ratio Numeric
           )") #Create Table Script
dbWriteTable(conn = db_conn, #Database Connector name
             name = "Annual_Balance_Info", #Table Name
             value = Annual_Balance_Info, #Load data from the newly created dataframe
             append=TRUE,row.names = FALSE,header = TRUE)#Since the CSV contains headers in the columns
dbListFields(db_conn, "Annual_Balance_Info") #Listing the field of the table
```

5. Calculation

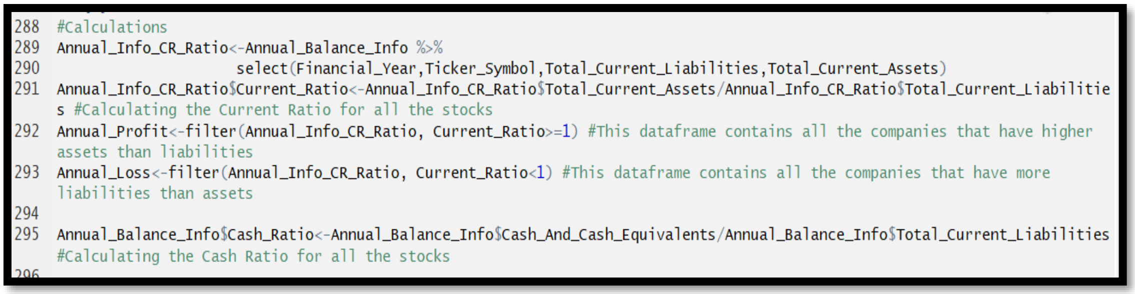
We have done few calculations on Current Ratio, Cash Ratio, %Change in volume/week and %Change in price/week was not present in our dataset. This calculation helps in knowing fluctuation of stocks in the stock market per week and the liability of a company. This additional calculation makes the analysis even more interesting and helps an intelligent trader invest wisely.

- Current Ratio
  - It is a ratio of Total Current Assets to Total Current Liabilities
  - If greater than 1, company has more assets than liabilities
  - If less than 1, company has more liabilities which is not good
  - If equal to 1, company has zero assets left

$$\textit{Current Ratio} = \frac{\textit{Total Current Assets}}{\textit{Total Current Liabilities}}$$

- Cash Ratio
  - The ratio of company's cash and cash equivalents to its current liabilities
  - If the company is forced to pay all current liabilities immediately, this metric shows the company's ability to do so without having to sell or liquidate other assets
  - If a company's cash ratio is equal to 1, the company has the same amount of current liabilities as it does cash and cash equivalents to pay off those debts.
  - If a company's cash ratio is less than 1, there are more current liabilities than cash and cash equivalents. In this situation, there is insufficient cash on hand to pay off short-term debt.
  - If a company's cash ratio is greater than 1, the company has more cash and cash equivalents than current liabilities. In this situation, the company can cover all short-term debt and still have cash remaining

$$\textit{Cash Ratio} = \frac{\textit{Cash \& Cash Equivalent}}{\textit{Total Current Liabilities}}$$



```
#Calculations
Annual_Info_CR_Ratio<-Annual_Balance_Info %>%
                    select(Financial_Year,Ticker_Symbol,Total_Current_Liabilities,Total_Current_Assets)
Annual_Info_CR_Ratio$Current_Ratio<-Annual_Info_CR_Ratio$Total_Current_Assets/Annual_Info_CR_Ratio$Total_Current_Liabilities #Calculating the Current Ratio for all the stocks
Annual_Profit<-filter(Annual_Info_CR_Ratio, Current_Ratio>=1) #This dataframe contains all the companies that have higher assets than liabilities
Annual_Loss<-filter(Annual_Info_CR_Ratio, Current_Ratio<1) #This dataframe contains all the companies that have more liabilities than assets

Annual_Balance_Info$Cash_Ratio<-Annual_Balance_Info$Cash_And_Cash_Equivalents/Annual_Balance_Info$Total_Current_Liabilities #Calculating the Cash Ratio for all the stocks
```

6. Data Retrieving

6.1 Analysis using SQL queries

To transform structured data into meaningful information. We performed analysis by retrieving data using SQL queries.

1) Display the Companies having a current ratio greater than one.

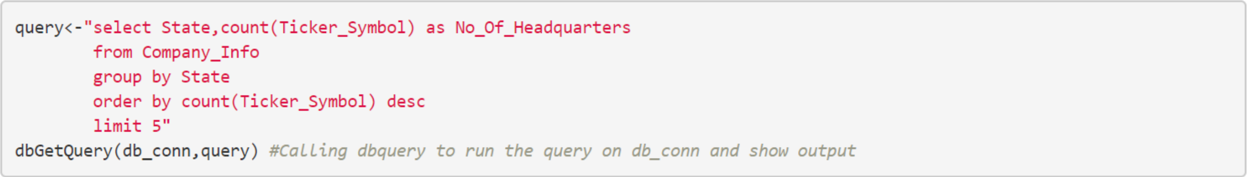
```
query<-"select a.Ticker_Symbol,c.Company_Name, (a.Total_Current_Assets/a.Total_Current_Liabilities) as Current_Ratio
        from Annual_Balance_Info a
        join Company_Info c on a.Ticker_Symbol=c.Ticker_Symbol
        group by a.Ticker_Symbol,c.Company_Name
        having (a.Total_Current_Assets/a.Total_Current_Liabilities)>1
        order by (a.Total_Current_Assets/a.Total_Current_Liabilities) desc
        limit 10"
dbGetQuery(db_conn,query) #Calling dbquery to run the query on db_conn and show output
```

```
##    Ticker_Symbol            Company_Name Current_Ratio
## 1             FB                Facebook            11
## 2           LLTC Linear Technology Corp.             9
## 3           SWKS      Skyworks Solutions             9
## 4           MCHP    Microchip Technology             8
## 5            ADI     Analog Devices, Inc.            6
## 6           FAST             Fastenal Co             6
## 7           MNST         Monster Beverage            6
## 8            FTR Frontier Communications             5
## 9           ISRG  Intuitive Surgical Inc.            5
## 10           WAT       Waters Corporation            5
```

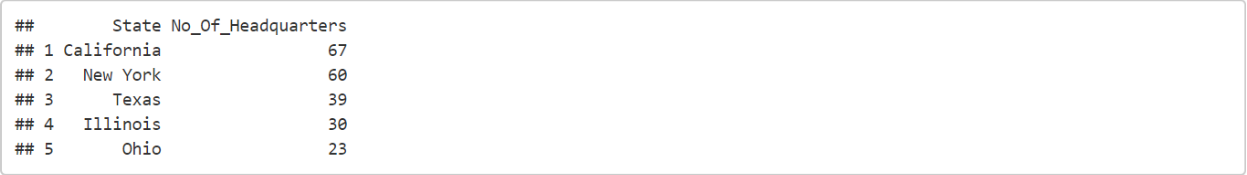
Result:

- Current ratio of Facebook is 11. Facebook has the highest current ratio followed by Linear Technology Corp and Skyworks Solutions with a current ratio of 8.
- Microchip Technology, Analog Devices, Inc., Fastenal Co, Monster Beverage, Frontier Communications, Intuitive Surgical Inc. and Waters Corporation having the current ratio of **greater than one**. It means these 10 companies have **more assets than liabilities**.

2) Which state has the highest number of headquarters?



```
query<-"select State,count(Ticker_Symbol) as No_Of_Headquarters
        from Company_Info
        group by State
        order by count(Ticker_Symbol) desc
        limit 5"
dbGetQuery(db_conn,query) #Calling dbquery to run the query on db_conn and show output
```



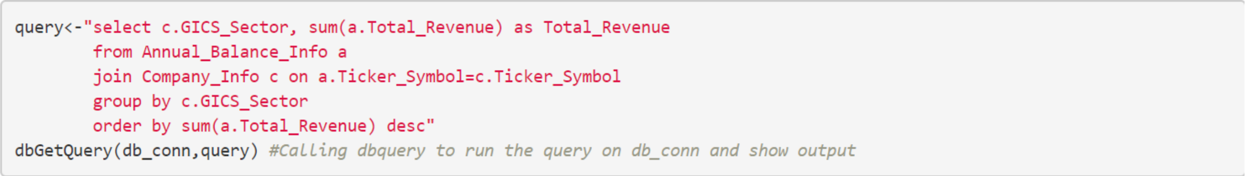
```
##        State No_Of_Headquarters
## 1 California                 67
## 2   New York                 60
## 3      Texas                 39
## 4   Illinois                 30
## 5       Ohio                 23
```

Result:

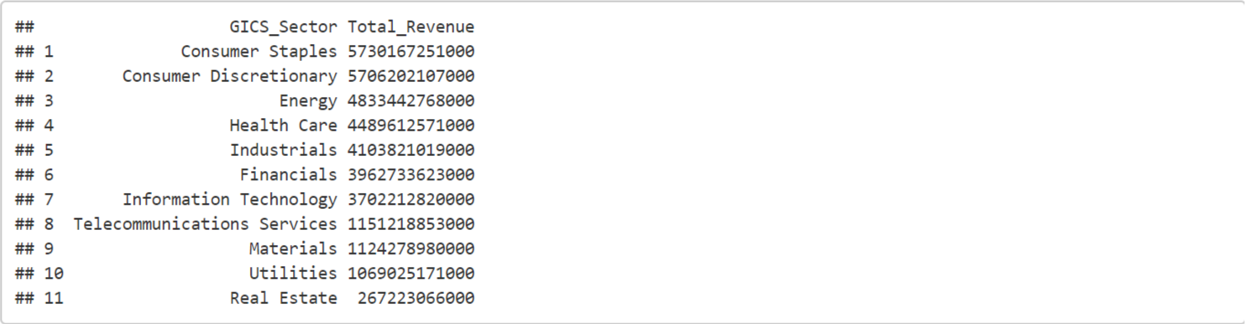
- **California** has the highest number of Headquarters i.e. 67, which means **maximum money flow** from there.

- **New York** is the second highest state with 60 no. of headquarters followed by Texas, Illinois, and Ohio.

3) Display all the GICS Sector along with their Total revenue.



```
query<-"select c.GICS_Sector, sum(a.Total_Revenue) as Total_Revenue
        from Annual_Balance_Info a
        join Company_Info c on a.Ticker_Symbol=c.Ticker_Symbol
        group by c.GICS_Sector
        order by sum(a.Total_Revenue) desc"
dbGetQuery(db_conn,query) #Calling dbquery to run the query on db_conn and show output
```



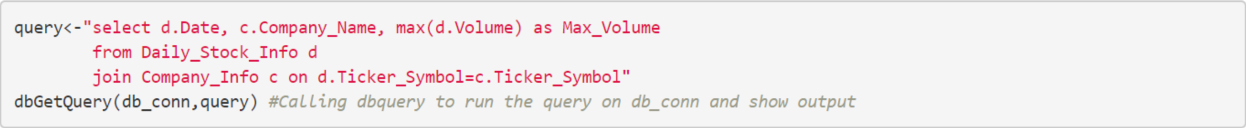
```
##                   GICS_Sector Total_Revenue
## 1            Consumer Staples 5730167251000
## 2      Consumer Discretionary 5706202107000
## 3                      Energy 4833442768000
## 4                 Health Care 4489612571000
## 5                 Industrials 4103821019000
## 6                  Financials 3962733623000
## 7      Information Technology 3702212820000
## 8  Telecommunications Services 1151218853000
## 9                   Materials 1124278980000
## 10                  Utilities 1069025171000
## 11                Real Estate  267223066000
```

Result:

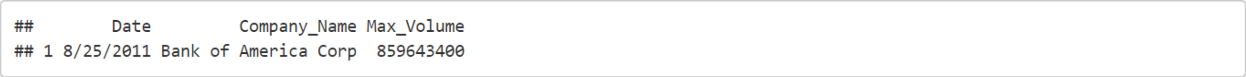
-Among GICS_Sector, **Consumer Staples** has the **highest** total revenue.

- Consumer Discretionary is very near to Consumer Discretionary. Energy Sector has the third highest Revenue. **Real Estate** has the **lowest** revenue.

4) When was the highest volume of stocks traded? For which company? How much?



```
query<-"select d.Date, c.Company_Name, max(d.Volume) as Max_Volume
        from Daily_Stock_Info d
        join Company_Info c on d.Ticker_Symbol=c.Ticker_Symbol"
dbGetQuery(db_conn,query) #Calling dbquery to run the query on db_conn and show output
```

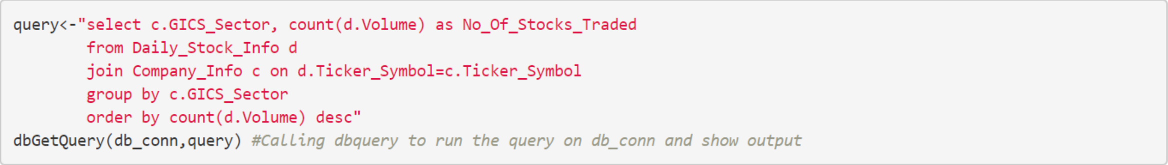


```
##        Date         Company_Name Max_Volume
## 1 8/25/2011 Bank of America Corp  859643400
```

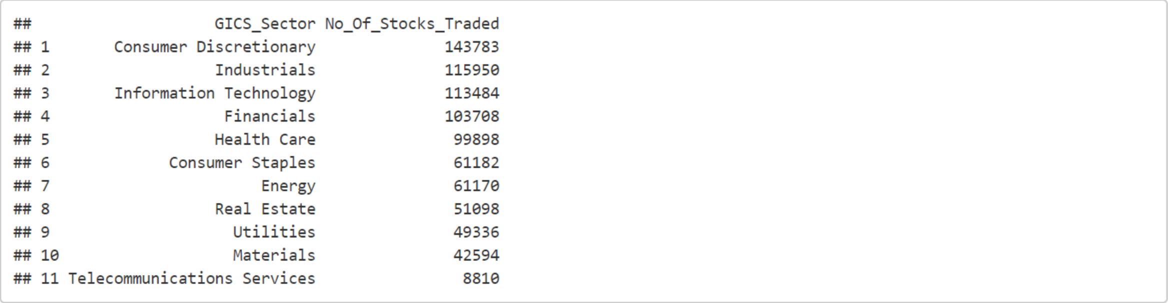
Result:

- Bank of America Corp has the **highest volume** of stocks traded on **08-25-2011** with the volume of 859643400. It means there was a huge demand for the Bank of America stock on 08-25-2011.

5) Which Industry has the highest number of stocks traded?



```
query<-"select c.GICS_Sector, count(d.Volume) as No_Of_Stocks_Traded
       from Daily_Stock_Info d
       join Company_Info c on d.Ticker_Symbol=c.Ticker_Symbol
       group by c.GICS_Sector
       order by count(d.Volume) desc"
dbGetQuery(db_conn,query) #Calling dbquery to run the query on db_conn and show output
```

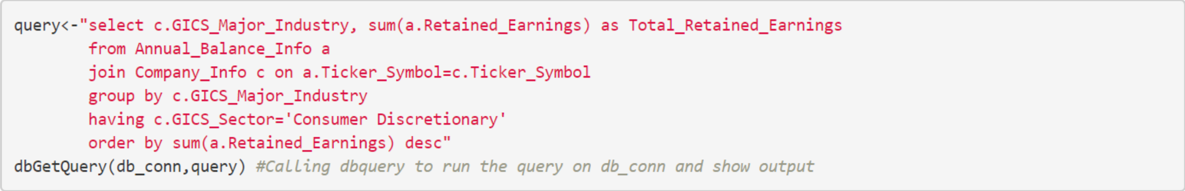


```
##                     GICS_Sector No_Of_Stocks_Traded
## 1        Consumer Discretionary              143783
## 2                   Industrials              115950
## 3        Information Technology              113484
## 4                    Financials              103708
## 5                   Health Care               99898
## 6              Consumer Staples               61182
## 7                        Energy               61170
## 8                   Real Estate               51098
## 9                     Utilities               49336
## 10                    Materials               42594
## 11 Telecommunications Services                 8810
```

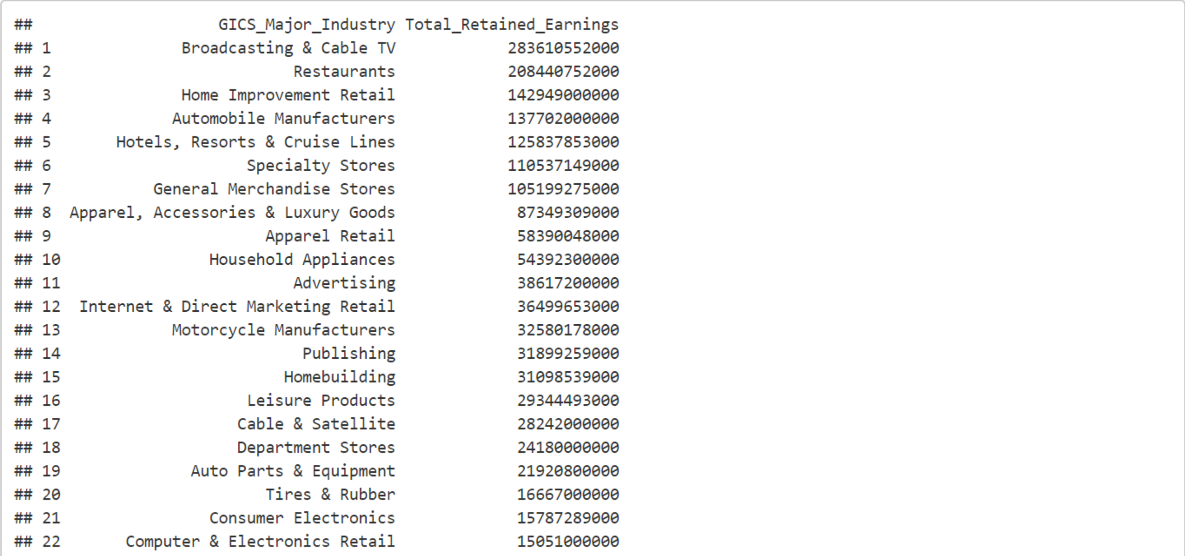
Result:

- **Consumer Discretionary** sector has the highest number of stock traded. Consumer Discretionary sector has 143783 stocks traded from 2010 to 2016. Consumer Discretionary stocks are in **demand**.
- Industrials sector has 115950 stocks traded and Information Technology has 113484 stocks traded.

6) Display the Retained earnings for Industries under Telecommunications Sector?



```
query<-"select c.GICS_Major_Industry, sum(a.Retained_Earnings) as Total_Retained_Earnings
       from Annual_Balance_Info a
       join Company_Info c on a.Ticker_Symbol=c.Ticker_Symbol
       group by c.GICS_Major_Industry
       having c.GICS_Sector='Consumer Discretionary'
       order by sum(a.Retained_Earnings) desc"
dbGetQuery(db_conn,query) #Calling dbquery to run the query on db_conn and show output
```



```
##                   GICS_Major_Industry Total_Retained_Earnings
## 1                Broadcasting & Cable TV            283610552000
## 2                            Restaurants            208440752000
## 3                Home Improvement Retail            142949000000
## 4               Automobile Manufacturers            137702000000
## 5          Hotels, Resorts & Cruise Lines            125837853000
## 6                       Specialty Stores            110537149000
## 7             General Merchandise Stores            105199275000
## 8   Apparel, Accessories & Luxury Goods             87349309000
## 9                         Apparel Retail             58390048000
## 10                  Household Appliances             54392300000
## 11                           Advertising             38617200000
## 12   Internet & Direct Marketing Retail             36499653000
## 13              Motorcycle Manufacturers             32580178000
## 14                            Publishing             31899259000
## 15                          Homebuilding             31098539000
## 16                      Leisure Products             29344493000
## 17                     Cable & Satellite             28242000000
## 18                     Department Stores             24180000000
## 19                 Auto Parts & Equipment             21920800000
## 20                        Tires & Rubber             16667000000
## 21                  Consumer Electronics             15787289000
## 22        Computer & Electronics Retail             15051000000
## 23              Home Furnishings          13148618000
## 24        Housewares & Specialties         8739100000
## 25               Specialty Retail          7147124000
## 26                   Distributors          6161206000
## 27              Automotive Retail          5390430000
## 28               Casinos & Gaming           330724000
```

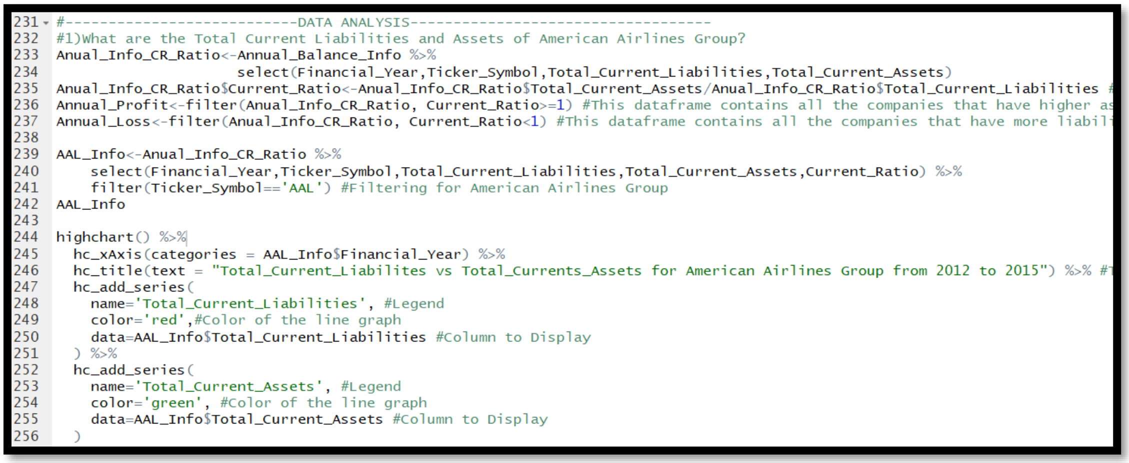
Result:

- Under Telecommunication Sector, there are **28 GICS Major Industry**.
- Broadcasting & Cable TV has retained earnings of 283610552000. Broadcasting & Cable TV has the highest net earning which is not paid out as dividends but retained by the company to be **reinvested** in its core business whereas Casinos and Gaming have the lowest retained earnings in the Telecommunication Sector.

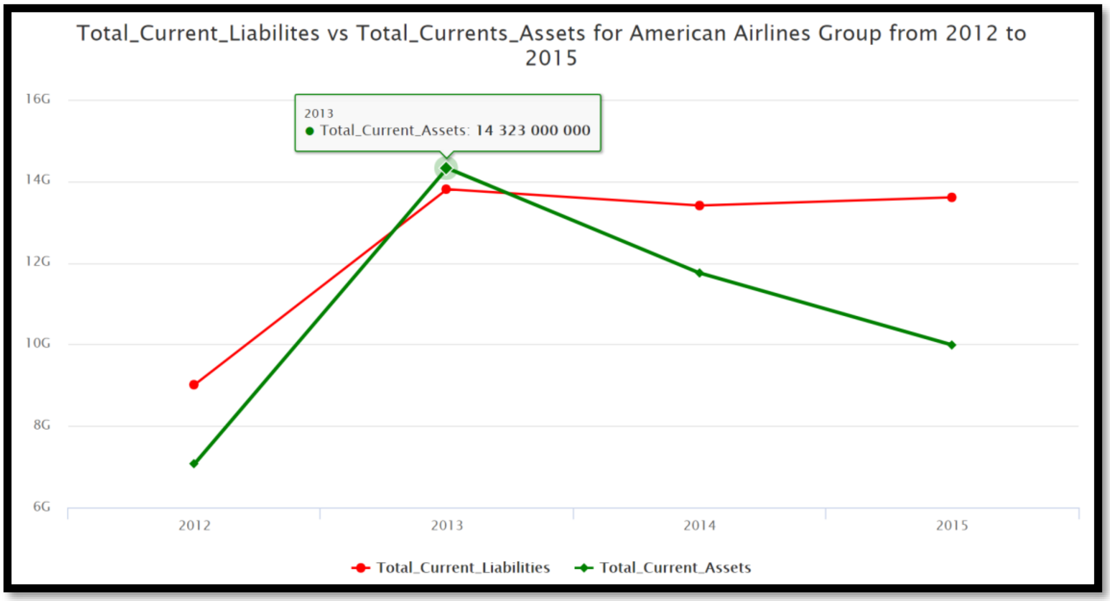
6.2. Data Visualization using R libraries

To perform advanced analysis, which gives more clarity in understanding the data. We have performed Data Visualization using R Libraries.

1) What are the Total Current Liabilities and Assets of American Airlines Group?



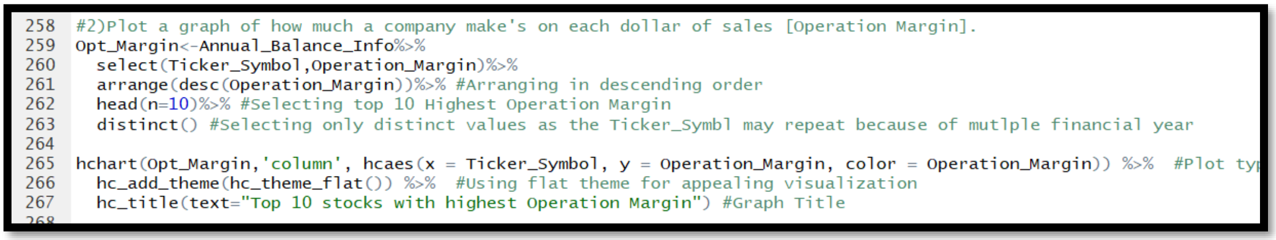
```
#---------------------------DATA ANALYSIS-----------------------------------
#1)What are the Total Current Liabilities and Assets of American Airlines Group?
Anual_Info_CR_Ratio<-Annual_Balance_Info %>%
                     select(Financial_Year,Ticker_Symbol,Total_Current_Liabilities,Total_Current_Assets)
Anual_Info_CR_Ratio$Current_Ratio<-Anual_Info_CR_Ratio$Total_Current_Assets/Anual_Info_CR_Ratio$Total_Current_Liabilities #
Annual_Profit<-filter(Anual_Info_CR_Ratio, Current_Ratio>=1) #This dataframe contains all the companies that have higher as
Annual_Loss<-filter(Anual_Info_CR_Ratio, Current_Ratio<1) #This dataframe contains all the companies that have more liabili

AAL_Info<-Anual_Info_CR_Ratio %>%
    select(Financial_Year,Ticker_Symbol,Total_Current_Liabilities,Total_Current_Assets,Current_Ratio) %>%
    filter(Ticker_Symbol=='AAL') #Filtering for American Airlines Group
AAL_Info

highchart() %>%
  hc_xAxis(categories = AAL_Info$Financial_Year) %>%
  hc_title(text = "Total_Current_Liabilites vs Total_Currents_Assets for American Airlines Group from 2012 to 2015") %>% #T
  hc_add_series(
    name='Total_Current_Liabilities', #Legend
    color='red',#Color of the line graph
    data=AAL_Info$Total_Current_Liabilities #Column to Display
  ) %>%
  hc_add_series(
    name='Total_Current_Assets', #Legend
    color='green', #Color of the line graph
    data=AAL_Info$Total_Current_Assets #Column to Display
  )
```



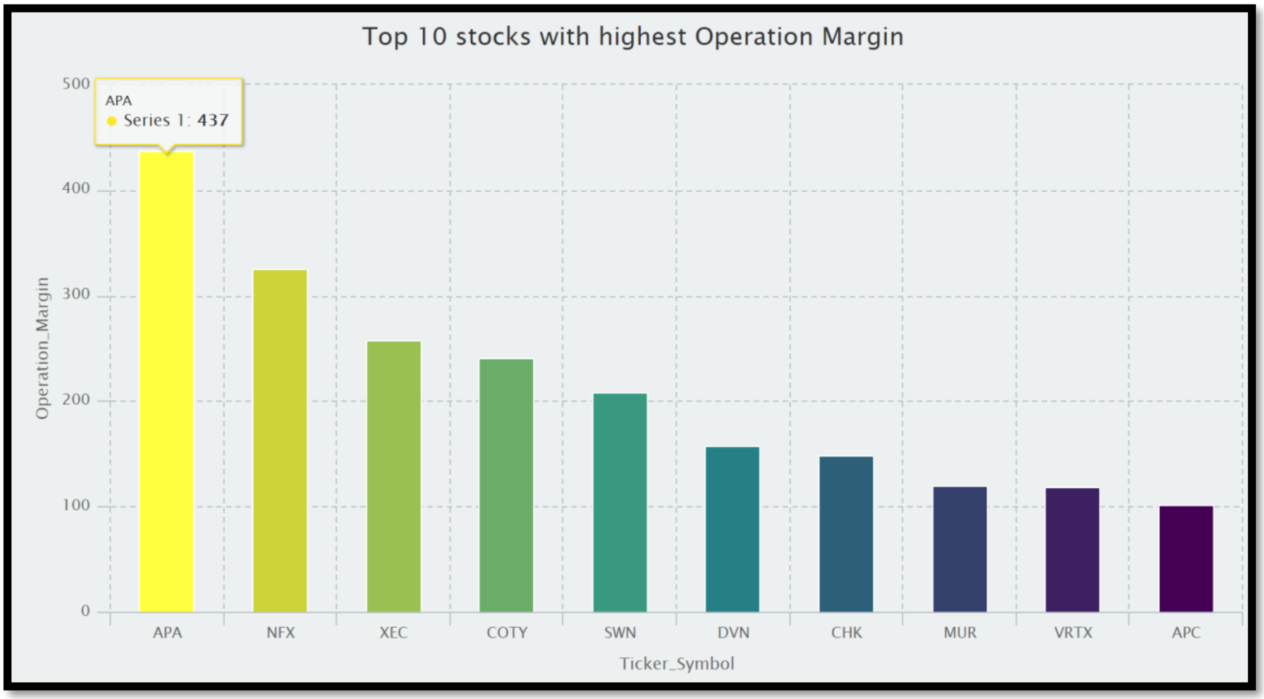
Result:

- Only 2013 has a Current Ratio **greater than one** which is a **good sign** otherwise current ratio is less than one in the year of 2012, 2014 and 2015 which means total liabilities are greater than total assets.

2) Plot a graph of how much a company makes on each dollar of sales [Operation Margin]



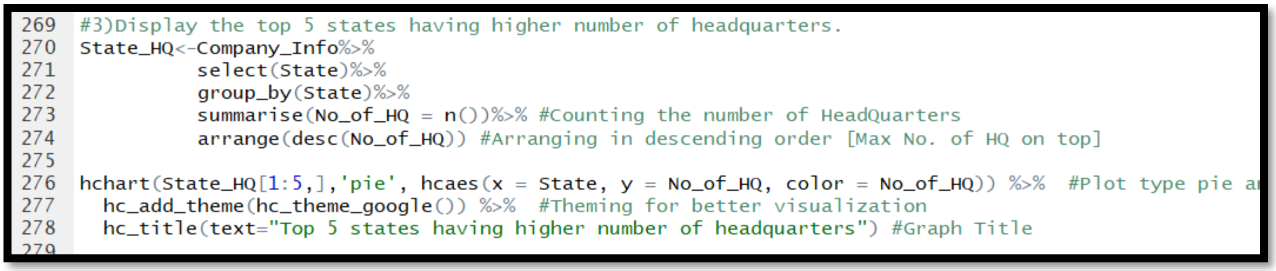
```r
#2)Plot a graph of how much a company make's on each dollar of sales [Operation Margin].
Opt_Margin<-Annual_Balance_Info%>%
  select(Ticker_Symbol,Operation_Margin)%>%
  arrange(desc(Operation_Margin))%>% #Arranging in descending order
  head(n=10)%>% #Selecting top 10 Highest Operation Margin
  distinct() #Selecting only distinct values as the Ticker_Symbl may repeat because of mutlple financial year

hchart(Opt_Margin,'column', hcaes(x = Ticker_Symbol, y = Operation_Margin, color = Operation_Margin)) %>%  #Plot ty
  hc_add_theme(hc_theme_flat()) %>%  #Using flat theme for appealing visualization
  hc_title(text="Top 10 stocks with highest Operation Margin") #Graph Title
```



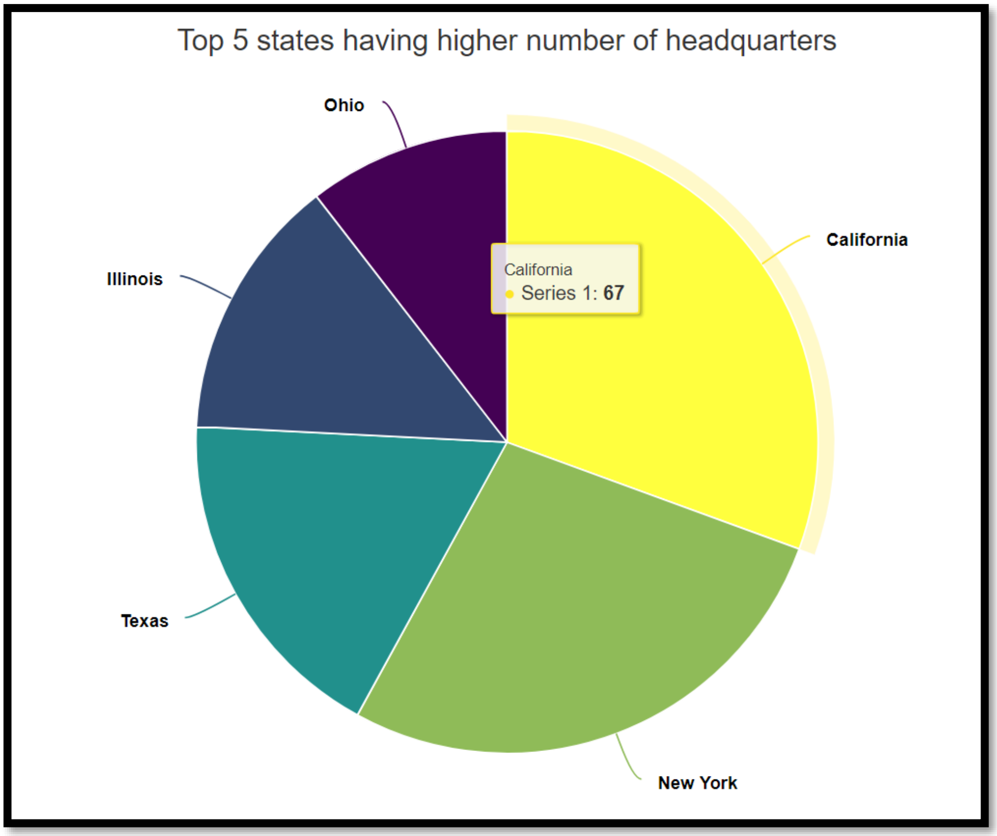
Result:

- Operation Margin measures **profitability**. APA stocks have the highest Operation Margin which says they earn the highest profit on each dollar of revenue.
- APA stocks belong to Apache Corporation. Investing in APA stocks seems to be a safe bet for investors. The bar graph displays top 10 stocks having highest Operation Margin.

3) Display the top 5 states having a higher number of headquarters.



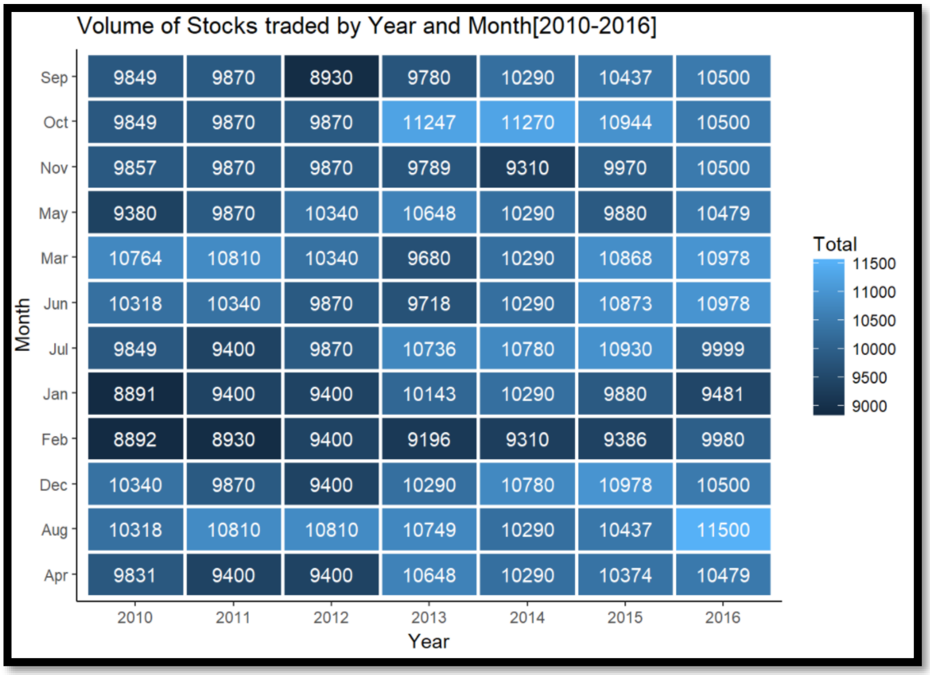
```
#3)Display the top 5 states having higher number of headquarters.
State_HQ<-Company_Info%>%
          select(State)%>%
          group_by(State)%>%
          summarise(No_of_HQ = n())%>% #Counting the number of HeadQuarters
          arrange(desc(No_of_HQ)) #Arranging in descending order [Max No. of HQ on top]

hchart(State_HQ[1:5,],'pie', hcaes(x = State, y = No_of_HQ, color = No_of_HQ)) %>%  #Plot type pie a
  hc_add_theme(hc_theme_google()) %>%  #Theming for better visualization
  hc_title(text="Top 5 states having higher number of headquarters") #Graph Title
```



Result:

- California has the 67 no. of headquarters from top 500 companies. Followed by New York, Texas, Illinois and Ohio. These are the top 5 states having highest no of headquarters.

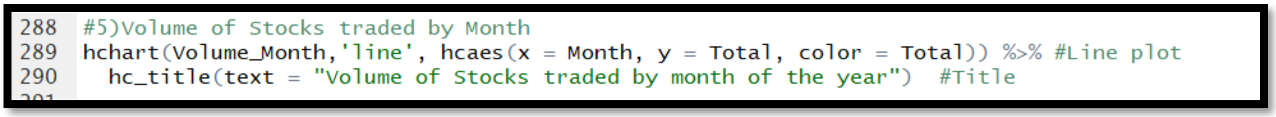
4) The volume of Stocks traded by Year and Month from 2010 to 2016.



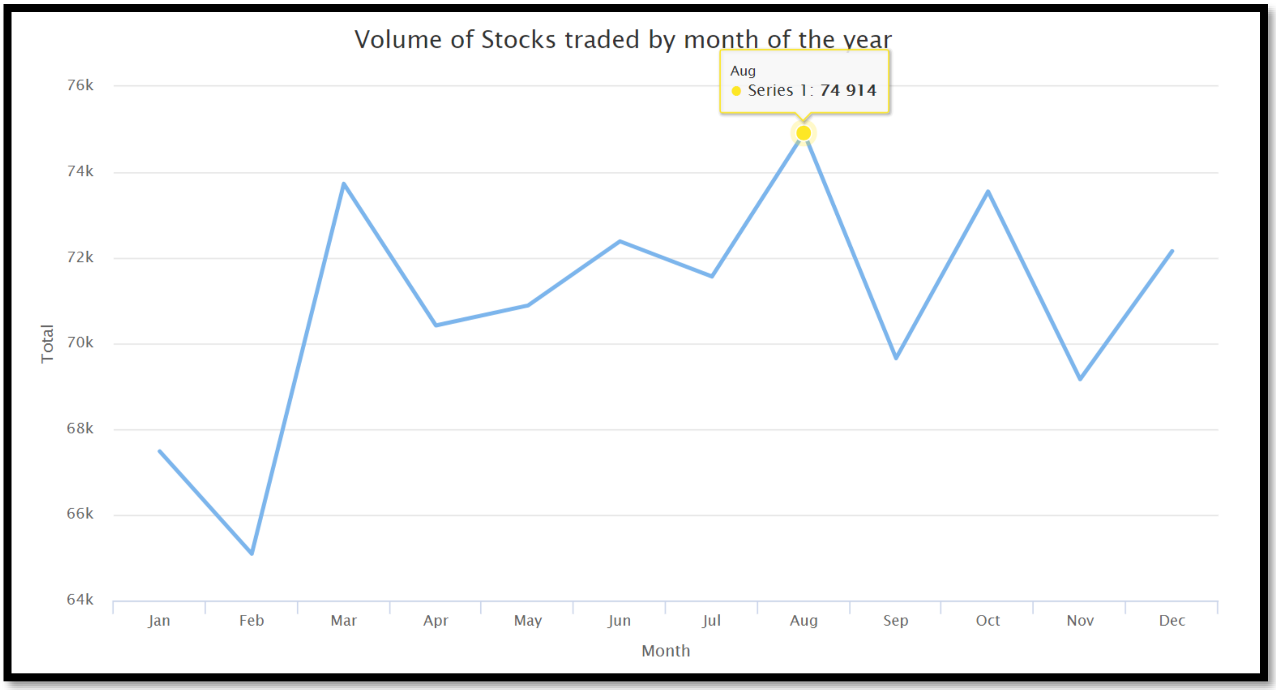
Result:

- Through this analysis, we can infer that, from 2010 to 2016, the highest volume of stocks was traded in **August 2016** and lowest volume of stocks were traded in **September 2012**.
- It is also observed that more than 10K volume of stocks is traded in the month of August from 2010 to 2016. August month has never seen a downfall in the volume of stocks traded. Investors usually **prefer investing** in the month of **August** because it's the middle of the year and maximum product launches at that time and hence the market is always in the boom.
- Also, there is another interesting insight that **less than 10K volume** of stocks were traded in the month of **February** always. This could be because it is the start of a year and budgets are usually not allocated thus it is always a risk.

5) The volume of Stocks traded by Month



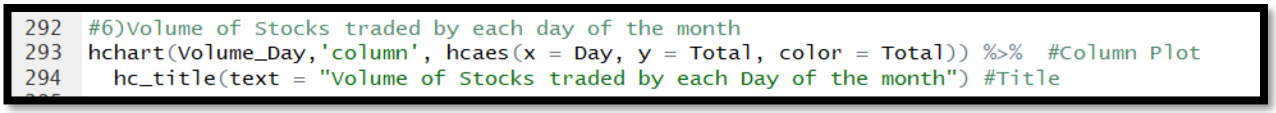
```
#5)Volume of Stocks traded by Month
hchart(Volume_Month,'line', hcaes(x = Month, y = Total, color = Total)) %>% #Line plot
  hc_title(text = "Volume of Stocks traded by month of the year")  #Title
```



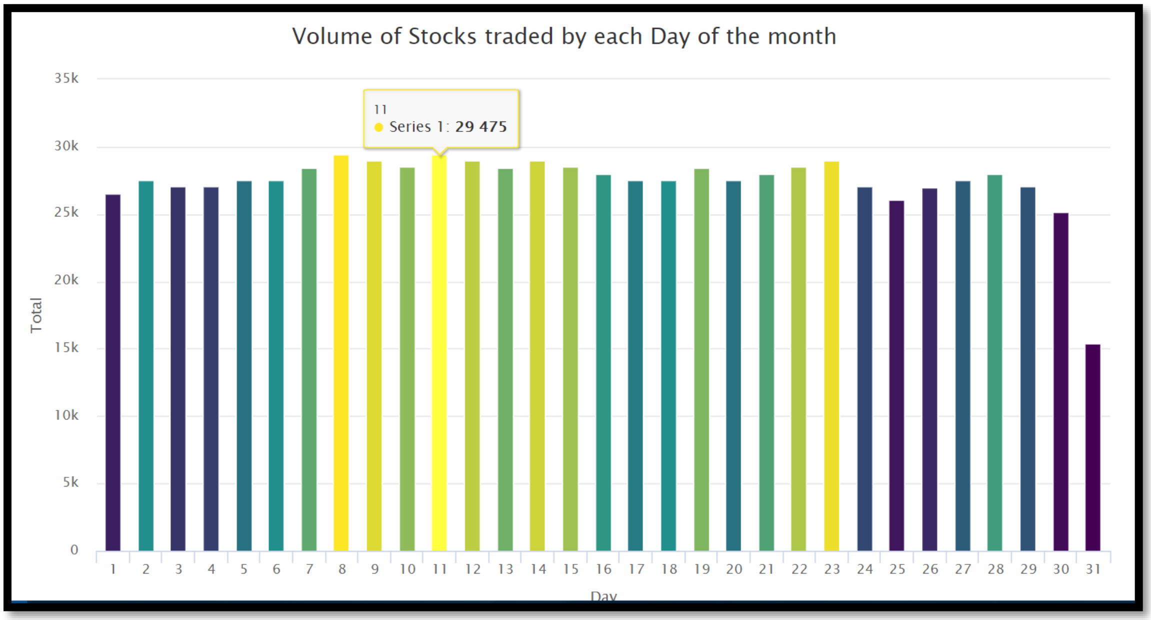
Result:

- Line graph clearly shows that **August has the highest** volume of stocks i.e. 74914 whereas February has the lowest number of stocks i.e. 65095.
- **January, February doesn't** seem to be a good time at Stock Market being the start of the year.
- The market seems to be very **uncertain** from **September to December**.

6) The volume of Stocks traded by each day of the month



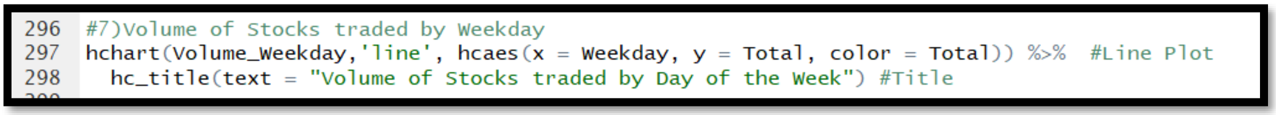
```
#6)Volume of Stocks traded by each day of the month
hchart(Volume_Day,'column', hcaes(x = Day, y = Total, color = Total)) %>%  #Column Plot
  hc_title(text = "Volume of Stocks traded by each Day of the month") #Title
```



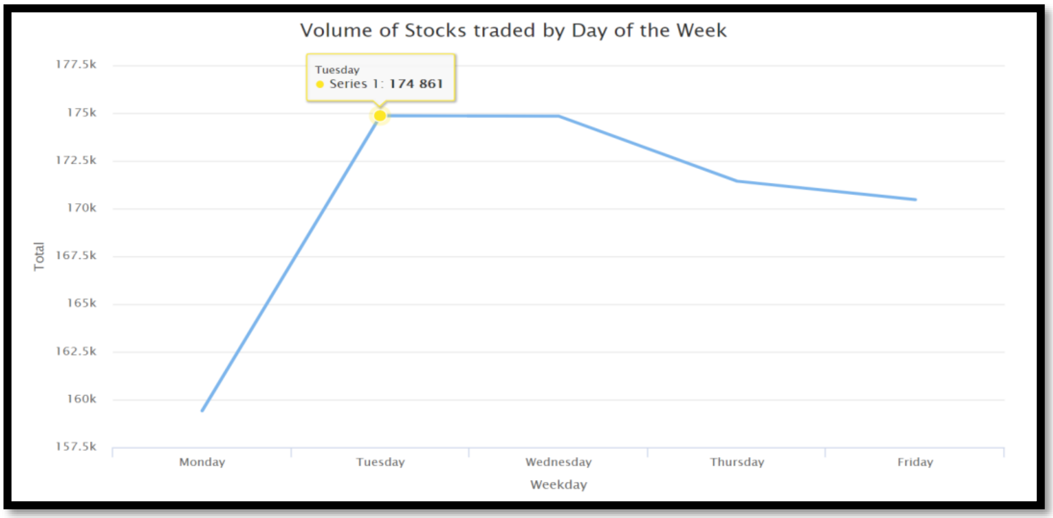
Result:

- There is the **highest peak in the middle** of each month. There has been a record that on 11th of every month, the highest volume of stocks is traded.
- Bars with yellow to green color says high volume of stocks has been traded, which is seen in the middle of every month. Whereas dark green to purple and blue says the **low volume** of stocks have been traded, which is seen in **start and end** of every month.
- People get their salary at the start of every month and the first week usually they spend on clearing their dues which helps them give a clear idea on how much to invest. Thus, middle of the month is the time when people seem to invest in stock market.

7) The volume of Stocks traded by Weekday



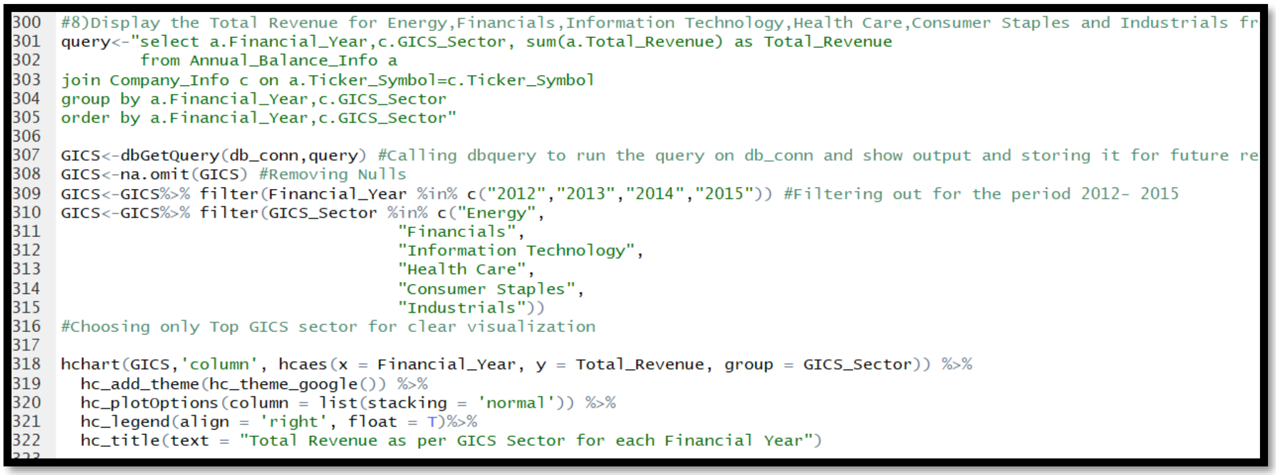
```
#7)Volume of Stocks traded by Weekday
hchart(Volume_Weekday,'line', hcaes(x = Weekday, y = Total, color = Total)) %>%  #Line Plot
  hc_title(text = "Volume of Stocks traded by Day of the Week") #Title
```



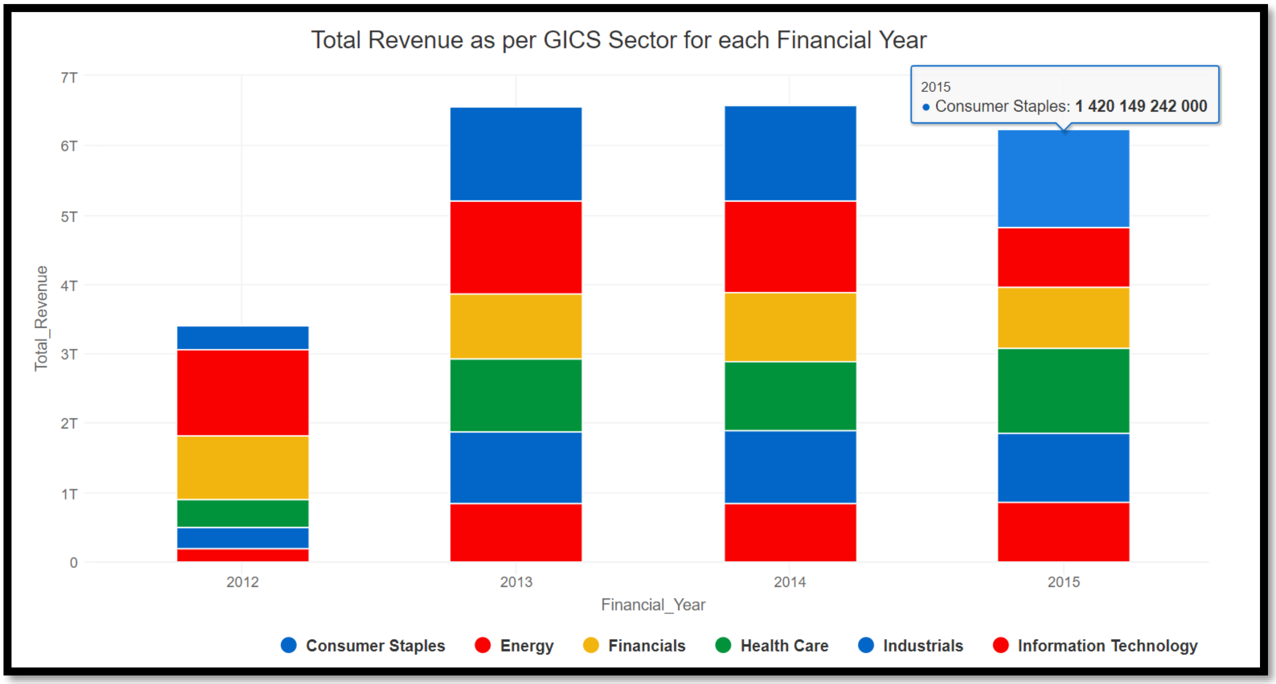
Result:

- **Lowest volume** of stocks are traded on **Monday**. As Monday is the first day of the week and it is also the busiest day of the week as well as Monday being the first day of the week for the stock market, the market is highly volatile thus making it unpredictable for investments.
- **The maximum volume** of stocks is traded on **Tuesdays and Wednesdays**. Tuesdays and Wednesdays are comparatively more predictable. Hence it is the preferred choice for the investors.

8) Display the Total Revenue for Energy, Financials, Information Technology, Health Care, Consumer Staples and Industrials from 2012 to 2015



```
#8)Display the Total Revenue for Energy,Financials,Information Technology,Health Care,Consumer Staples and Industrials fr
query<-"select a.Financial_Year,c.GICS_Sector, sum(a.Total_Revenue) as Total_Revenue
        from Annual_Balance_Info a
join Company_Info c on a.Ticker_Symbol=c.Ticker_Symbol
group by a.Financial_Year,c.GICS_Sector
order by a.Financial_Year,c.GICS_Sector"

GICS<-dbGetQuery(db_conn,query) #Calling dbquery to run the query on db_conn and show output and storing it for future re
GICS<-na.omit(GICS) #Removing Nulls
GICS<-GICS%>% filter(Financial_Year %in% c("2012","2013","2014","2015")) #Filtering out for the period 2012- 2015
GICS<-GICS%>% filter(GICS_Sector %in% c("Energy",
                                  "Financials",
                                  "Information Technology",
                                  "Health Care",
                                  "Consumer Staples",
                                  "Industrials"))
#Choosing only Top GICS sector for clear visualization

hchart(GICS,'column', hcaes(x = Financial_Year, y = Total_Revenue, group = GICS_Sector)) %>%
  hc_add_theme(hc_theme_google()) %>%
  hc_plotOptions(column = list(stacking = 'normal')) %>%
  hc_legend(align = 'right', float = T)%>%
  hc_title(text = "Total Revenue as per GICS Sector for each Financial Year")
```



Result:

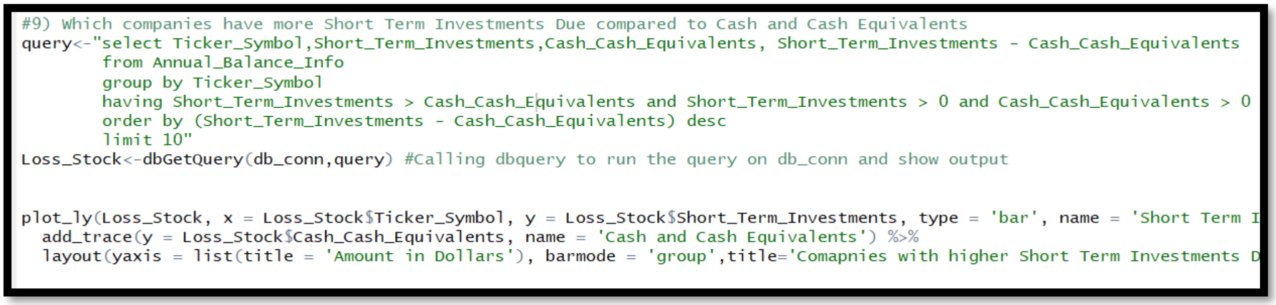
- From Data Retrieval Question 5 we found that Consumer Staples, Energy, Financials, Health Care, Industrials and Information Technology are top 5 GICS Sector having the highest number of stocks traded.

- To perform further analysis, we created a stacked graph to look for total revenue of top 5 GICS Sectors.

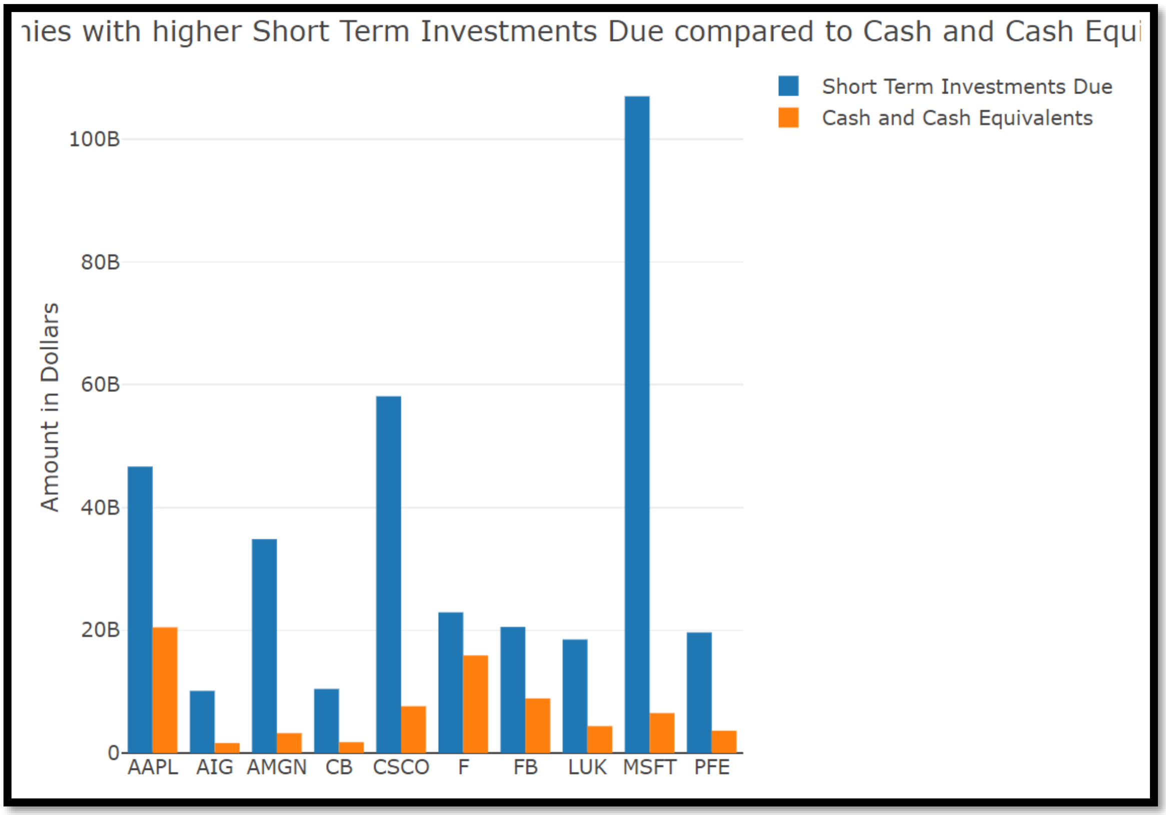
- Above graph shows that Total Revenue for Consumer Staples has increased every year from 2012 to 2015. **Consumer Staples** are basic necessities and what so ever be the individuals' financial conditions, consumption cannot be compromised. Hence it has an **ever-growing market**.

- Total Revenue is decreased in 2015 as compared to 2014 and 2013 because of a decrease in revenue from Energy, Financials, Industrials and Information Technology sectors.

9) Which companies have more Short-Term Investments Due compared to Cash and Cash Equivalent.



```
#9) Which companies have more Short Term Investments Due compared to Cash and Cash Equivalents
query<-"select Ticker_Symbol,Short_Term_Investments,Cash_Cash_Equivalents, Short_Term_Investments - Cash_Cash_Equivalents
        from Annual_Balance_Info
        group by Ticker_Symbol
        having Short_Term_Investments > Cash_Cash_Equivalents and Short_Term_Investments > 0 and Cash_Cash_Equivalents > 0
        order by (Short_Term_Investments - Cash_Cash_Equivalents) desc
        limit 10"
Loss_Stock<-dbGetQuery(db_conn,query) #Calling dbquery to run the query on db_conn and show output


plot_ly(Loss_Stock, x = Loss_Stock$Ticker_Symbol, y = Loss_Stock$Short_Term_Investments, type = 'bar', name = 'Short Term I
  add_trace(y = Loss_Stock$Cash_Cash_Equivalents, name = 'Cash and Cash Equivalents') %>%
  layout(yaxis = list(title = 'Amount in Dollars'), barmode = 'group',title='Comapnies with higher Short Term Investments D
```



Result:

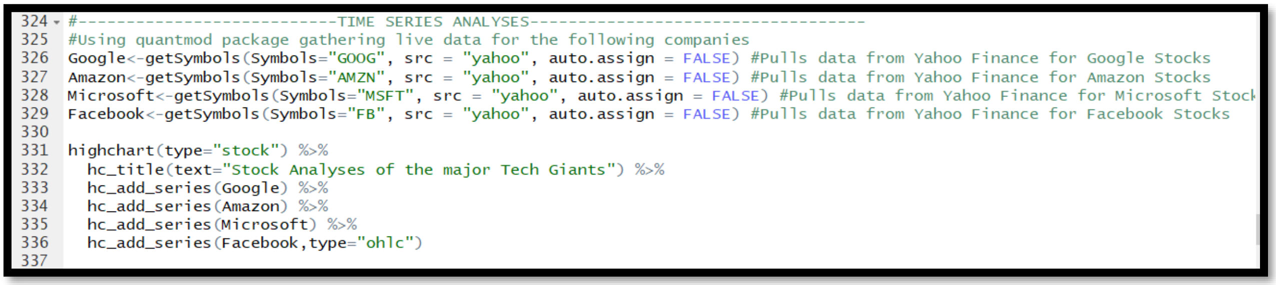
- Short-Term Investments Due are the **debts** incurred by the company which is due within one year. This value helps in determining a company's performance.
- Cash and Cash Equivalents are company's assets that are **cash or can be converted** into cash immediately. Companies with a high amount of cash and cash equivalents are better and show higher liquidity.
- Thus, having less short-term investments due and higher cash & cash equivalents makes company stable and reliable to buy stocks.
- The above bar graph shows **top 10 companies** which have more short-term investments due and less cash & cash equivalent, which makes them **less reliable**. This analysis can make investors think twice before buying their stocks.

7. Additional

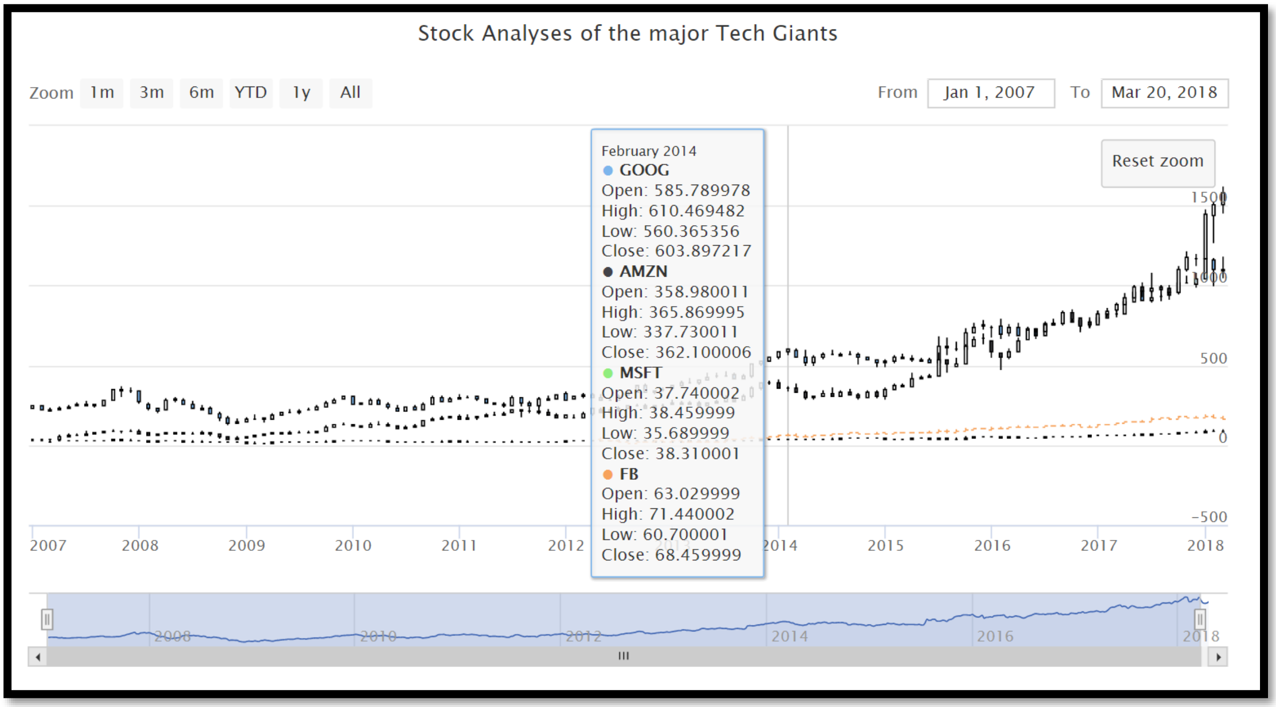
7.1 Data Mining – Time Series Analysis

Quantmod Package is a financial modeling R package which has a lot of financial trading options such as adjust OHLC (Open, High, Low, Close) stock price. Getsymbols is a wrapper to load data from different sources and it defaults to yahoo source. It helps in creating interesting and intuitive financial charts.

To generate Time Series Analysis, we are using Yahoo! Finance to get livestock market data from January 1, 2007, till now. Google, Amazon, Microsoft, and Facebook run the Information Technology sector, hence we decided to perform Time Series Analysis on top 4 Information Technology companies.



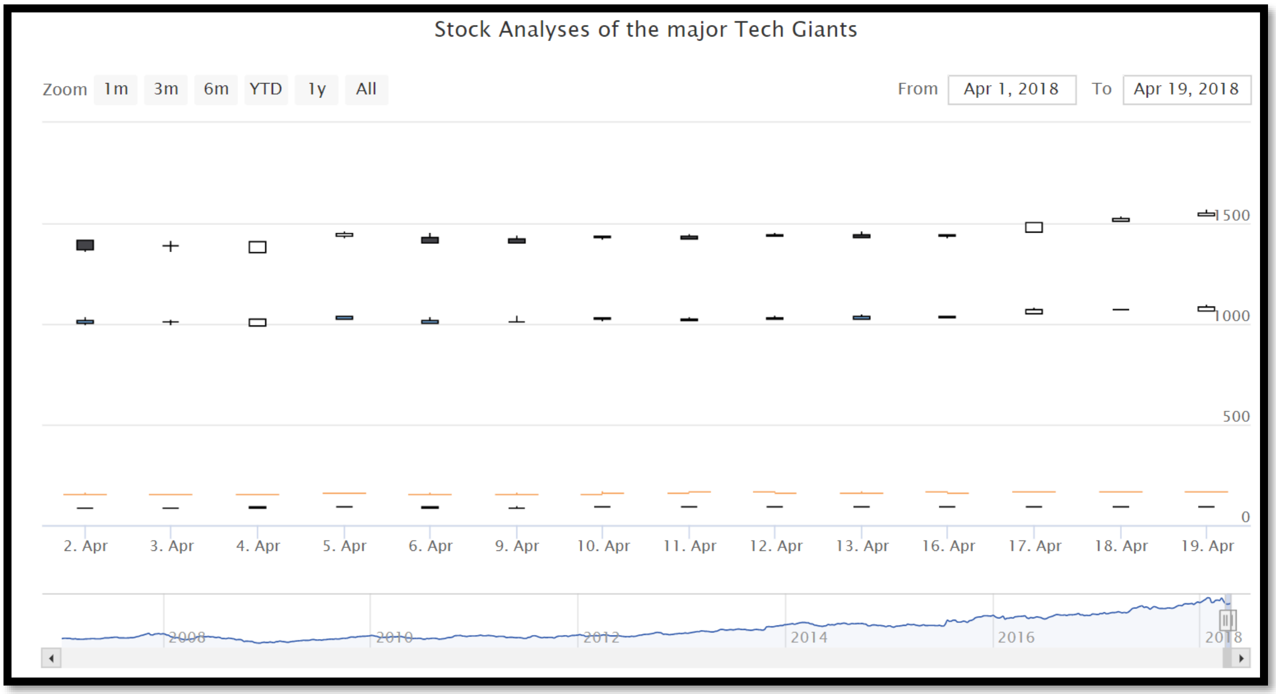
```r
#---------------------------TIME SERIES ANALYSES-----------------------------------
#Using quantmod package gathering live data for the following companies
Google<-getSymbols(Symbols="GOOG", src = "yahoo", auto.assign = FALSE) #Pulls data from Yahoo Finance for Google Stocks
Amazon<-getSymbols(Symbols="AMZN", src = "yahoo", auto.assign = FALSE) #Pulls data from Yahoo Finance for Amazon Stocks
Microsoft<-getSymbols(Symbols="MSFT", src = "yahoo", auto.assign = FALSE) #Pulls data from Yahoo Finance for Microsoft Stock
Facebook<-getSymbols(Symbols="FB", src = "yahoo", auto.assign = FALSE) #Pulls data from Yahoo Finance for Facebook Stocks

highchart(type="stock") %>%
  hc_title(text="Stock Analyses of the major Tech Giants") %>%
  hc_add_series(Google) %>%
  hc_add_series(Amazon) %>%
  hc_add_series(Microsoft) %>%
  hc_add_series(Facebook,type="ohlc")
```



Stock Analyses of the major Tech Giants

- Above Time series analysis takes the **live data** containing Open, Close, Highest and Lowest price of Google, Amazon, Microsoft and Facebook stocks from **Yahoo Finance source**. It helps you in comparing the price of 4 stocks. This analysis is very useful for investors who are keen on investing in these stocks and looking for the right time to gain maximum profit.

- Refer the following Link for **Dynamic Time Series**, as above graph is understood better when seen through the HTML format: (Download the HTML to view it)
- https://drive.google.com/open?id=140j85Tjm6BuBsgcVJMVwtXCJotdakrR2

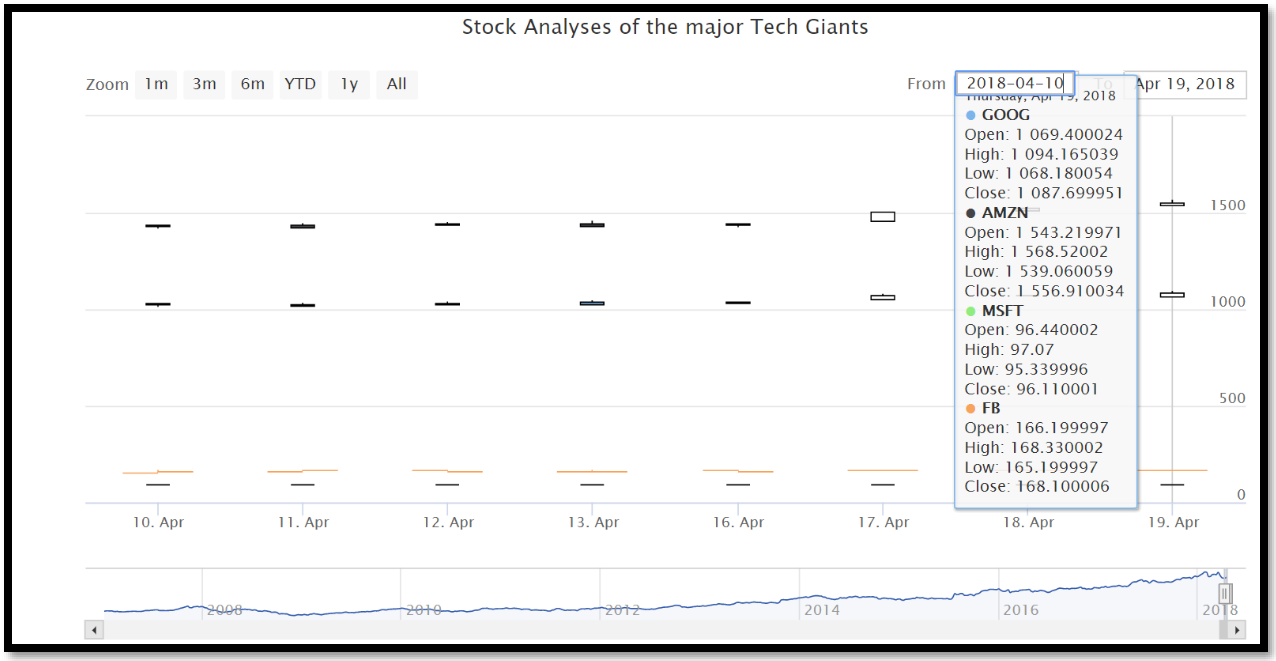


Stock Analyses of the major Tech Giants

Zoom 1m 3m 6m YTD 1y All

From Apr 1, 2018 To Apr 19, 2018

- Dynamic Time Series graph helps you change and look for the desired Start and End date of the time series graph.
- In the above screenshot, we are looking for the latest stock prices of Amazon, Google, Facebook and Microsoft from April 1, 2018, to April 19, 2018.



Stock Analyses of the major Tech Giants

Zoom 1m 3m 6m YTD 1y All

From 2018-04-10 To Apr 19, 2018

Thursday, Apr 19, 2018

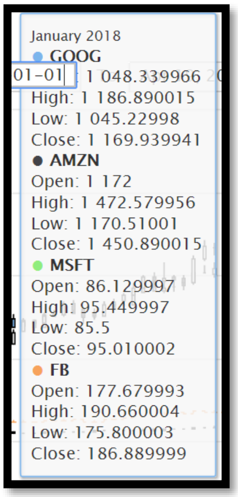
● GOOG
Open: 1 069.400024
High: 1 094.165039
Low: 1 068.180054
Close: 1 087.699951

● AMZN
Open: 1 543.219971
High: 1 568.52002
Low: 1 539.060059
Close: 1 556.910034

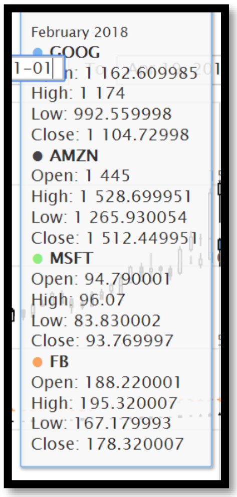
● MSFT
Open: 96.440002
High: 97.07
Low: 95.339996
Close: 96.110001

● FB
Open: 166.199997
High: 168.330002
Low: 165.199997
Close: 168.100006

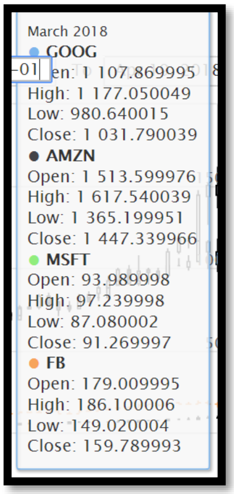
- On April 21, Google stocks open at 1069 which is the highest while comparing with Amazon, Microsoft, and Facebook.
- High and Low stock price of Microsoft and Facebook is not fluctuating. It is very stable and not risky to invest.



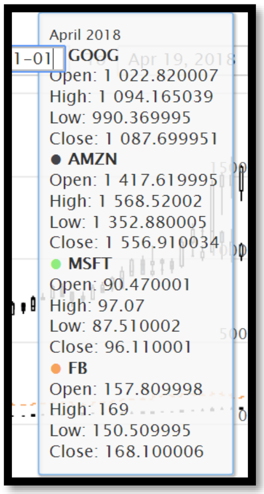
January 2018
● GOOG
01-01 1 048.339966
High: 1 186.890015
Low: 1 045.22998
Close: 1 169.939941
● AMZN
Open: 1 172
High: 1 472.579956
Low: 1 170.51001
Close: 1 450.890015
● MSFT
Open: 86.129997
High: 95.449997
Low: 85.5
Close: 95.010002
● FB
Open: 177.679993
High: 190.660004
Low: 175.800003
Close: 186.889999



February 2018
● GOOG
1-01 n: 1 162.609985
High: 1 174
Low: 992.559998
Close: 1 104.72998
● AMZN
Open: 1 445
High: 1 528.699951
Low: 1 265.930054
Close: 1 512.449951
● MSFT
Open: 94.790001
High: 96.07
Low: 83.830002
Close: 93.769997
● FB
Open: 188.220001
High: 195.320007
Low: 167.179993
Close: 178.320007



March 2018
● GOOG
-01 en: 1 107.869995
High: 1 177.050049
Low: 980.640015
Close: 1 031.790039
● AMZN
Open: 1 513.599976
High: 1 617.540039
Low: 1 365.199951
Close: 1 447.339966
● MSFT
Open: 93.989998
High: 97.239998
Low: 87.080002
Close: 91.269997
● FB
Open: 179.009995
High: 186.100006
Low: 149.020004
Close: 159.789993

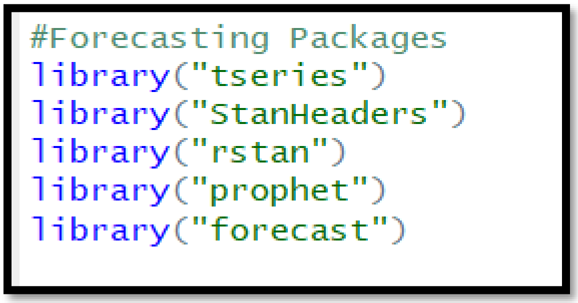


April 2018
1-01 GOOG
Open: 1 022.820007
High: 1 094.165039
Low: 990.369995
Close: 1 087.699951
● AMZN
Open: 1 417.619995
High: 1 568.52002
Low: 1 352.880005
Close: 1 556.910034
● MSFT
Open: 90.470001
High: 97.07
Low: 87.510002
Close: 96.110001
● FB
Open: 157.809998
High: 169
Low: 150.509995
Close: 168.100006

- We have taken a screenshot from the Dynamic Time Series graph to compare the stock prices for Amazon, Google, Microsoft, and Facebook for the month of January, February, March and April (until now) which will help to analyse the stocks before investing in them.
- As seen above, Google's stock prices are highly fluctuating since February. Hence, due to such a volatile nature of the stock, an investor should consider other parameters too before investing.

7.2 Forecasting/Prediction Algorithm

Using the historical data, we are forecasting the volume of stocks traded for the future years. We are predicting the data using two packages. One is Prophet package, and another is forecast and tries to build the ARIMA model.

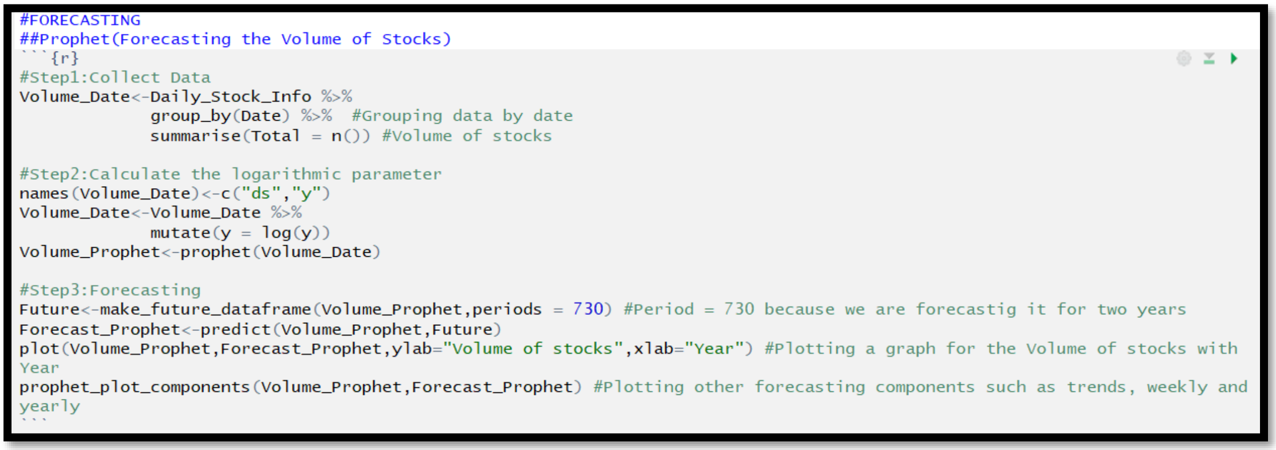


```
#Forecasting Packages
library("tseries")
library("StanHeaders")
library("rstan")
library("prophet")
library("forecast")
```

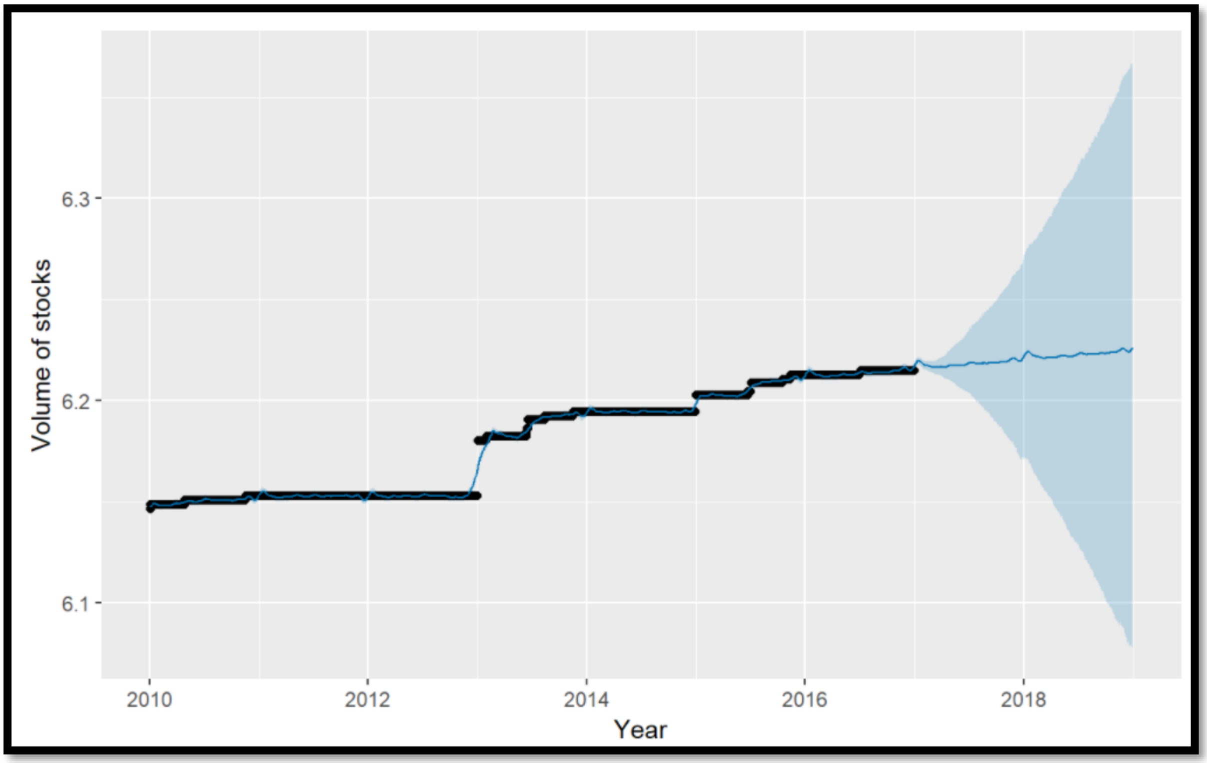
7.2.1 Prophet – Facebook's Forecasting R Package

Prophet, Facebook's R package is a procedure to forecast time series data. It is an open source software released by Facebook's Data Science core team. It is used for non-linear trends and best used for daily periodic values having historical data.

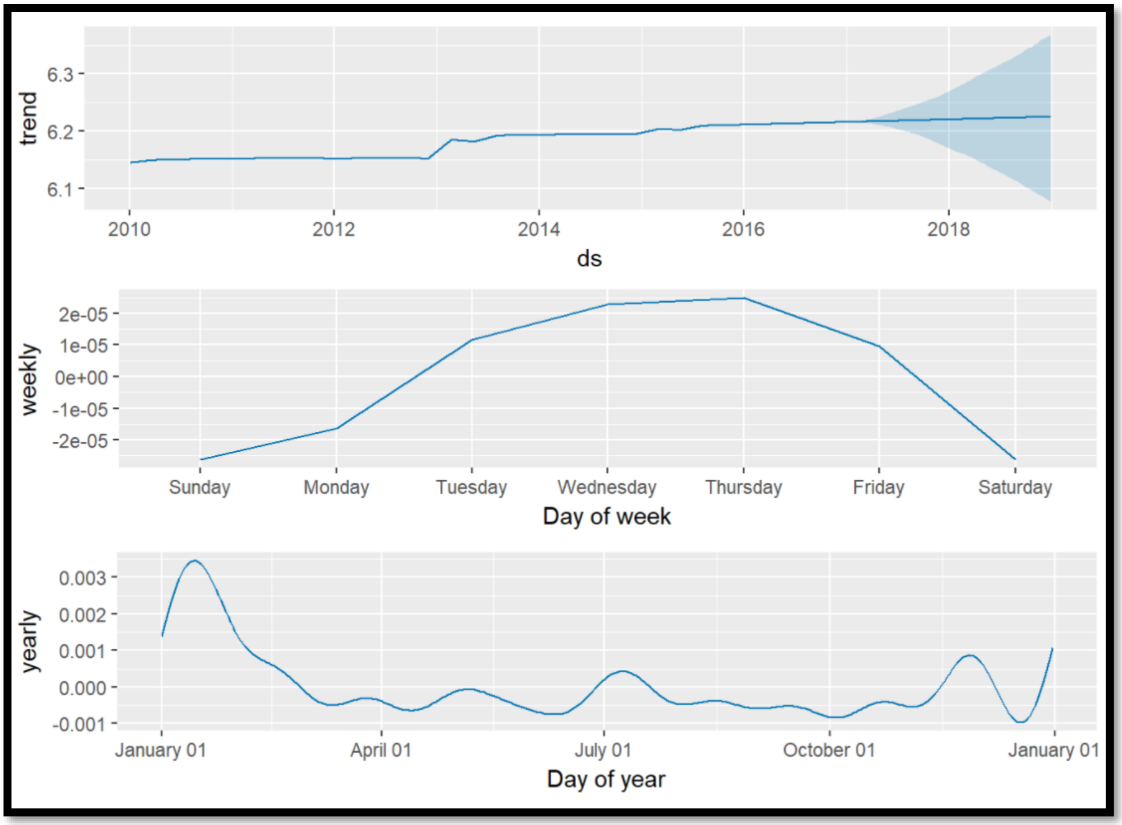
Using the historical data from 2010 and assuming to have all unpredictable factors (such as geo political factors like government changing policies, civil instabilities or natural calamities), we are predicting the volume of stocks traded for the year 2017 and 2018.



~~~
#FORECASTING
##Prophet(Forecasting the Volume of Stocks)
```{r}
#Step1:Collect Data
Volume_Date<-Daily_Stock_Info %>%
             group_by(Date) %>%  #Grouping data by date
             summarise(Total = n()) #Volume of stocks

#Step2:Calculate the logarithmic parameter
names(Volume_Date)<-c("ds","y")
Volume_Date<-Volume_Date %>%
             mutate(y = log(y))
Volume_Prophet<-prophet(Volume_Date)

#Step3:Forecasting
Future<-make_future_dataframe(Volume_Prophet,periods = 730) #Period = 730 because we are forecastig it for two years
Forecast_Prophet<-predict(Volume_Prophet,Future)
plot(Volume_Prophet,Forecast_Prophet,ylab="Volume of stocks",xlab="Year") #Plotting a graph for the Volume of stocks with
Year
prophet_plot_components(Volume_Prophet,Forecast_Prophet) #Plotting other forecasting components such as trends, weekly and
yearly
```
~~~



- As per the historical data, it can be forecasted that there is a minor increase in the volume of stock traded. The volume of stocks increased to 6.23 in 2018 to 6.21 in 2016.



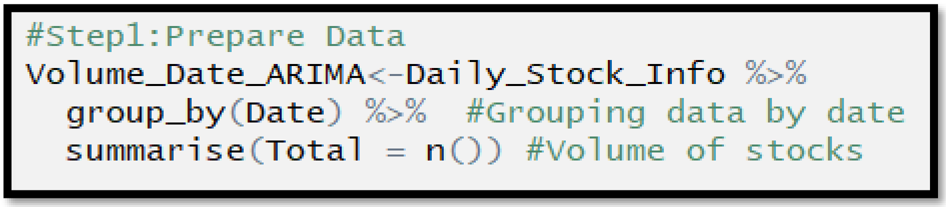
- There has been a change in trend which defines peoples trading pattern, the gradual shift from Tuesday and Wednesday which was seen in Data Visualization section to Wednesday and Thursday in the year 2017 and 2018.
- August always has been a crucial year for trading until 2016, but it has been forecasted that January will supersede August in the year 2018.

7.2.2 ARIMA Model

- ARIMA model is the Autoregressive integrated moving average. ARMIA model is applied in cases where data show evidence of non-stationarity, where an initial differencing step (corresponding to the "integrated" part of the model) can be applied one or more times to eliminate the non-stationarity.
- ARIMA model is used to forecasting model that utilize historical information to make predictions.

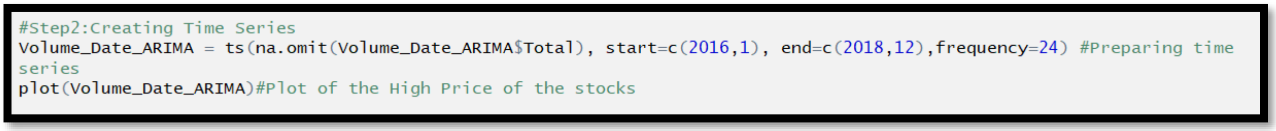
Step 1: Collecting data

Collecting data and limiting it to few companies for better understanding and clear analysis.

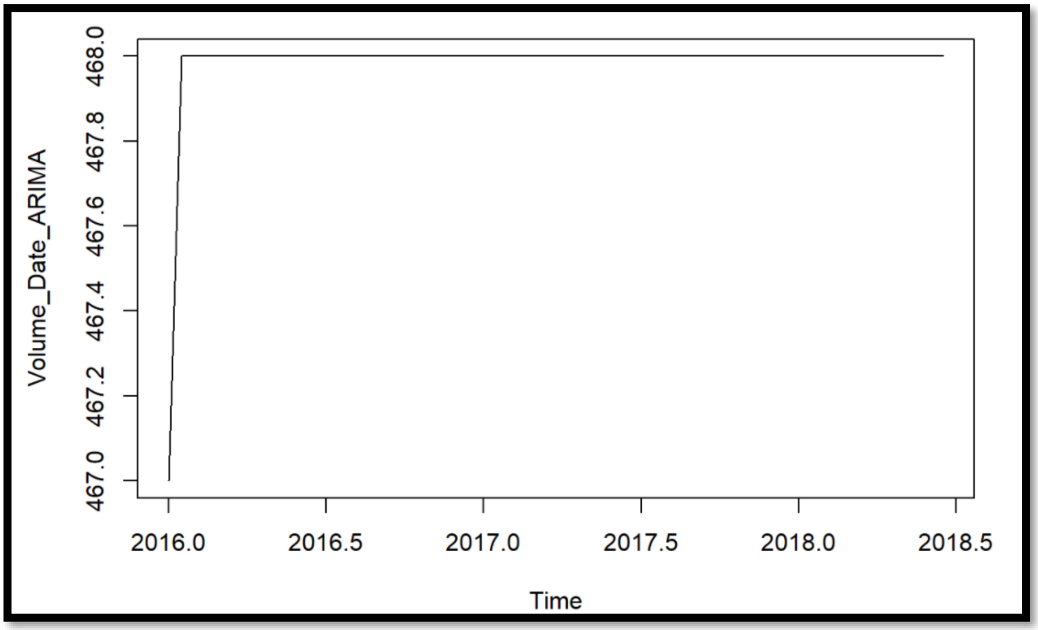


```
#Step1:Prepare Data
Volume_Date_ARIMA<-Daily_Stock_Info %>%
  group_by(Date) %>%  #Grouping data by date
  summarise(Total = n()) #Volume of stocks
```

Step 2: Creating Time Series

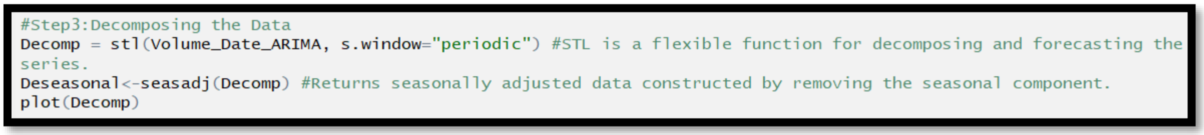


```
#Step2:Creating Time Series
Volume_Date_ARIMA = ts(na.omit(Volume_Date_ARIMA$Total), start=c(2016,1), end=c(2018,12),frequency=24) #Preparing time
series
plot(Volume_Date_ARIMA)#Plot of the High Price of the stocks
```

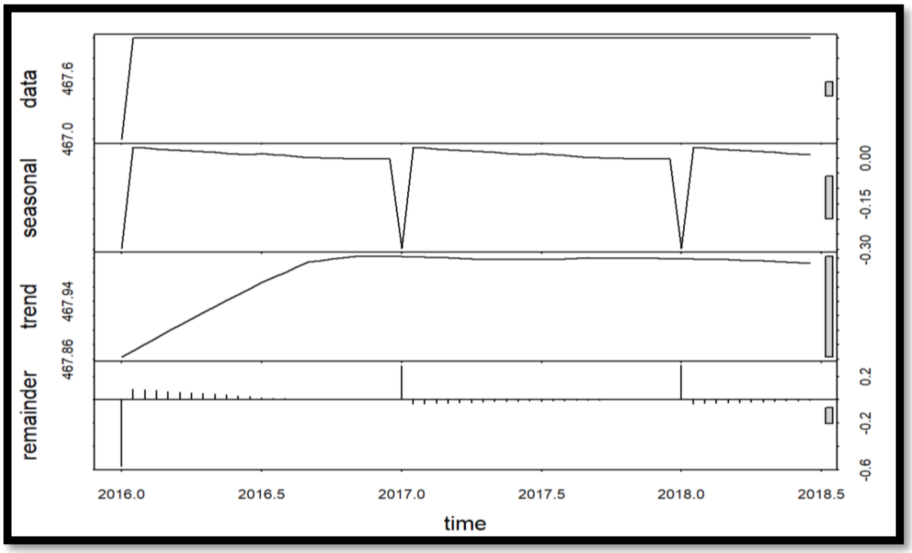


Step 3: Decomposing data

Decomposing the data to remove seasonality trends and outliers.

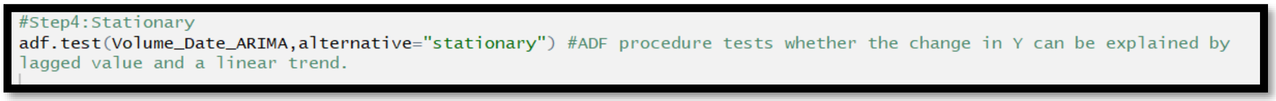


```
#Step3:Decomposing the Data
Decomp = stl(Volume_Date_ARIMA, s.window="periodic") #STL is a flexible function for decomposing and forecasting the
series.
Deseasonal<-seasadj(Decomp) #Returns seasonally adjusted data constructed by removing the seasonal component.
plot(Decomp)
```



Step 4: Stationary

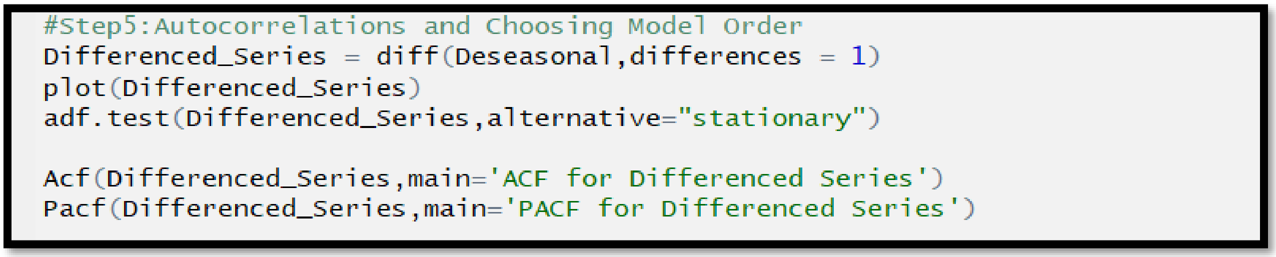
ARIMA model requires series to be stationary. ADF (Augmented Dickey-Fuller) test is a statistical test for stationarity. The test checks whether the change in Y can be explained by lagged value and a linear trend.



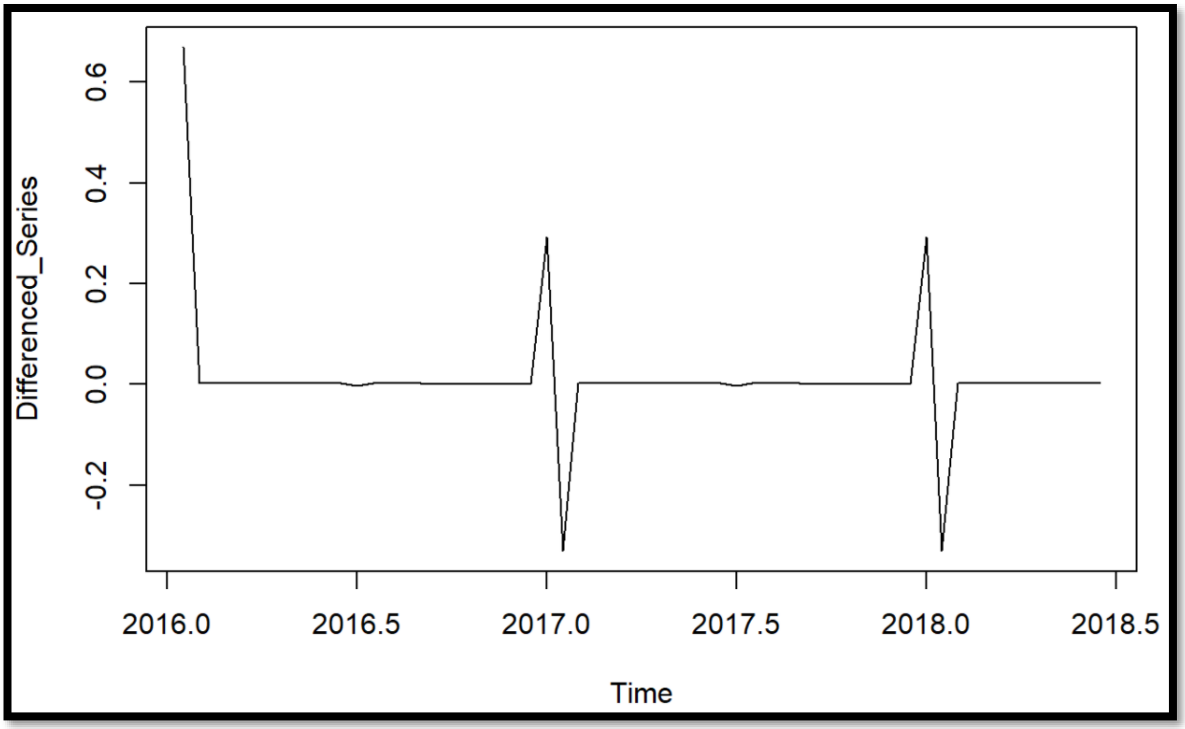
```
#Step4:Stationary
adf.test(Volume_Date_ARIMA,alternative="stationary") #ADF procedure tests whether the change in Y can be explained by
lagged value and a linear trend.
```

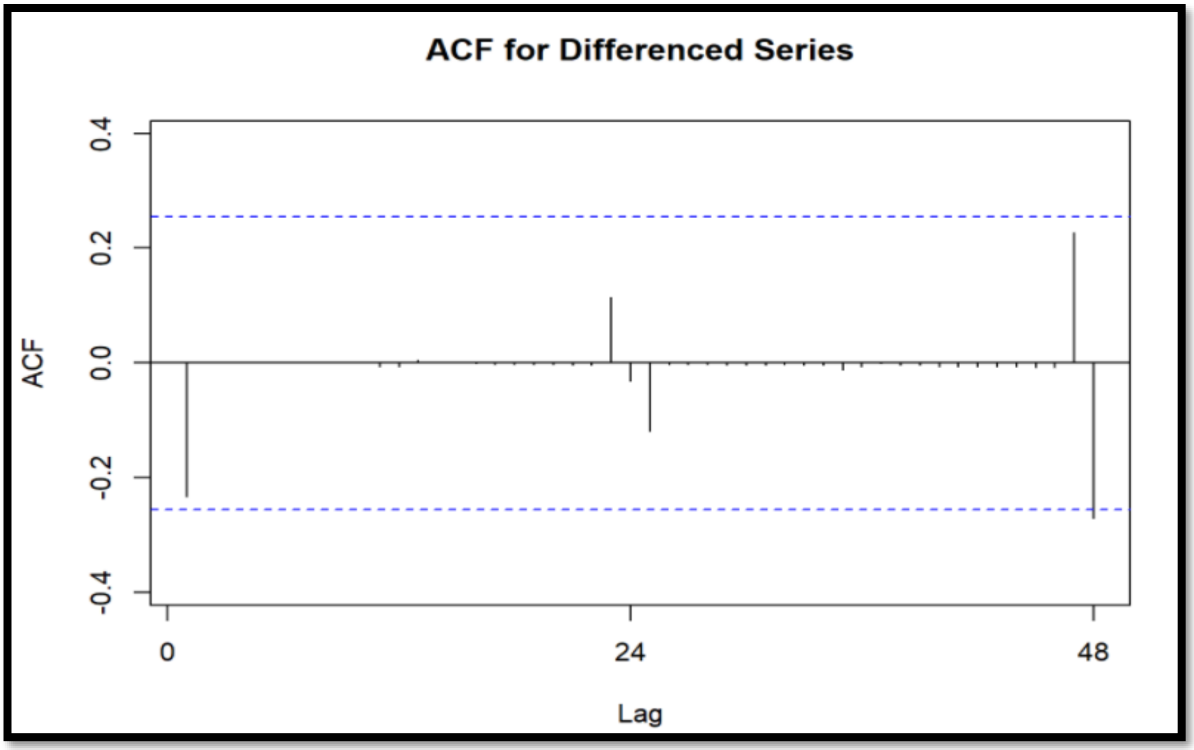
Step 5: Choosing Model Order

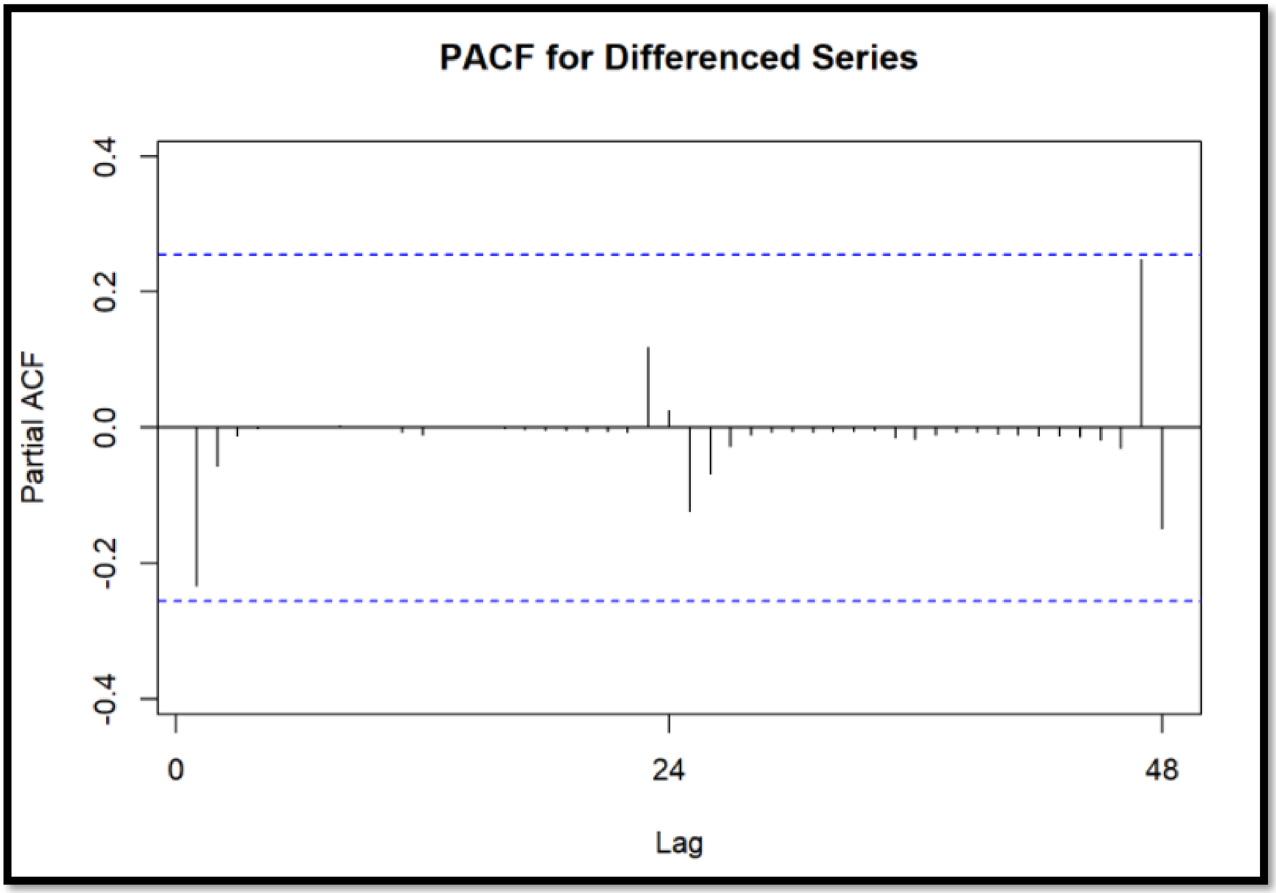
ACF() plots correlation between a series and its lags. PACF () at K lag is autocorrelation function to plot the correlation between all data points that are exactly k steps apart



```
#Step5:Autocorrelations and Choosing Model Order
Differenced_Series = diff(Deseasonal,differences = 1)
plot(Differenced_Series)
adf.test(Differenced_Series,alternative="stationary")

Acf(Differenced_Series,main='ACF for Differenced Series')
Pacf(Differenced_Series,main='PACF for Differenced Series')
```



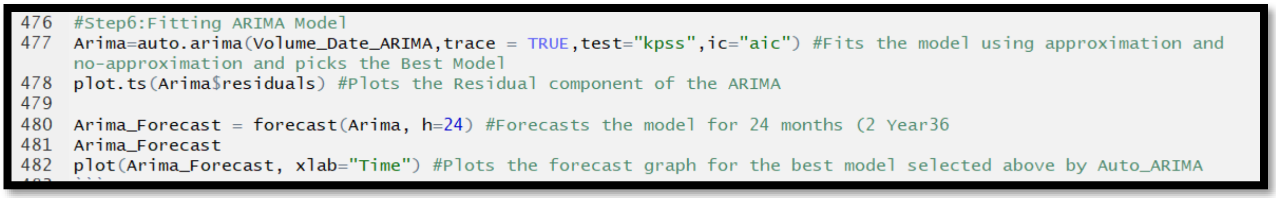




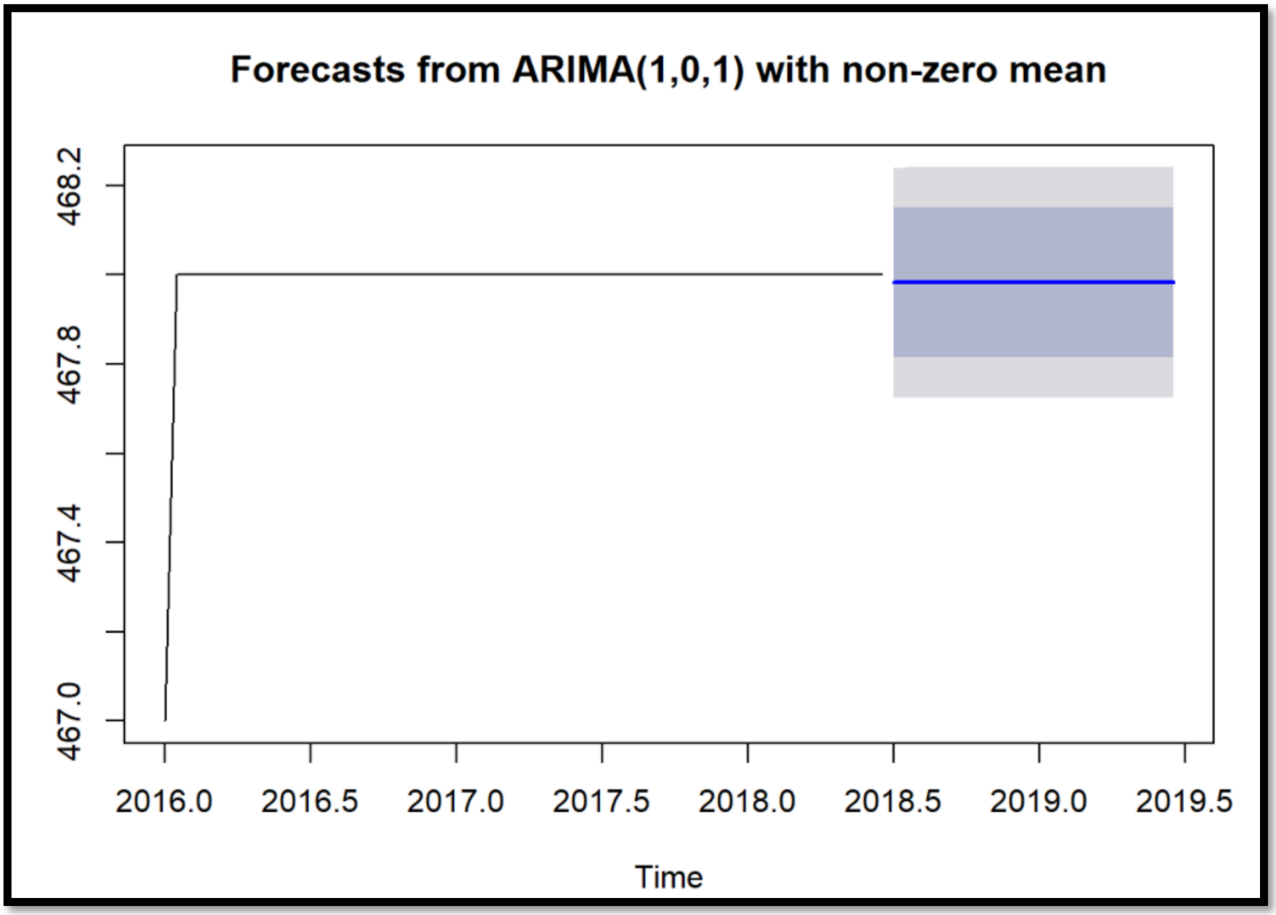
ACF and PACF residuals show no significant autocorrelations.

Step 6: Fitting ARIMA Model

An auto.arima function automatically selects an appropriate lag value using statistical tests and trains a linear regression model.



```
#Step6:Fitting ARIMA Model
Arima=auto.arima(Volume_Date_ARIMA,trace = TRUE,test="kpss",ic="aic") #Fits the model using approximation and
no-approximation and picks the Best Model
plot.ts(Arima$residuals) #Plots the Residual component of the ARIMA

Arima_Forecast = forecast(Arima, h=24) #Forecasts the model for 24 months (2 Year36
Arima_Forecast
plot(Arima_Forecast, xlab="Time") #Plots the forecast graph for the best model selected above by Auto_ARIMA
```



- Using ARIMA model it is predicted that Volume of Stocks traded remains same in 2018 and 2019 compared to previous year
- One can find minimal difference stocks traded from 2016 to 2019.

7.2.3 Comparison between Prophet and ARIMA

- Prophet utilizes a Bayesian-based curve fitting method to forecast the time series data while
- ARIMA uses the traditional method for forecasting.
- The cool thing about Prophet is that it doesn't require much prior knowledge or experience of forecasting time series data since it automatically finds seasonal trends beneath the data and offers a set of 'easy to understand' parameters.

- Hence, it allows non-statisticians to start using it and get reasonably good results that are often equal or sometimes even better than the ones produced by the experts.
- But this is not always the case that Prophet is better ARIMA always.

7.3. Stock Market Web Application – RShiny

R lets us create easy and interactive web applications using the R Shiny Package. We can host standalone apps on a webpage or embed them in R Markdown documents or dashboards. To make the project more interactive, we decided to build few R Shiny applications.

All the data frames including input files are on the cloud which makes the R Shiny Application accessible from any device including mobile phones.

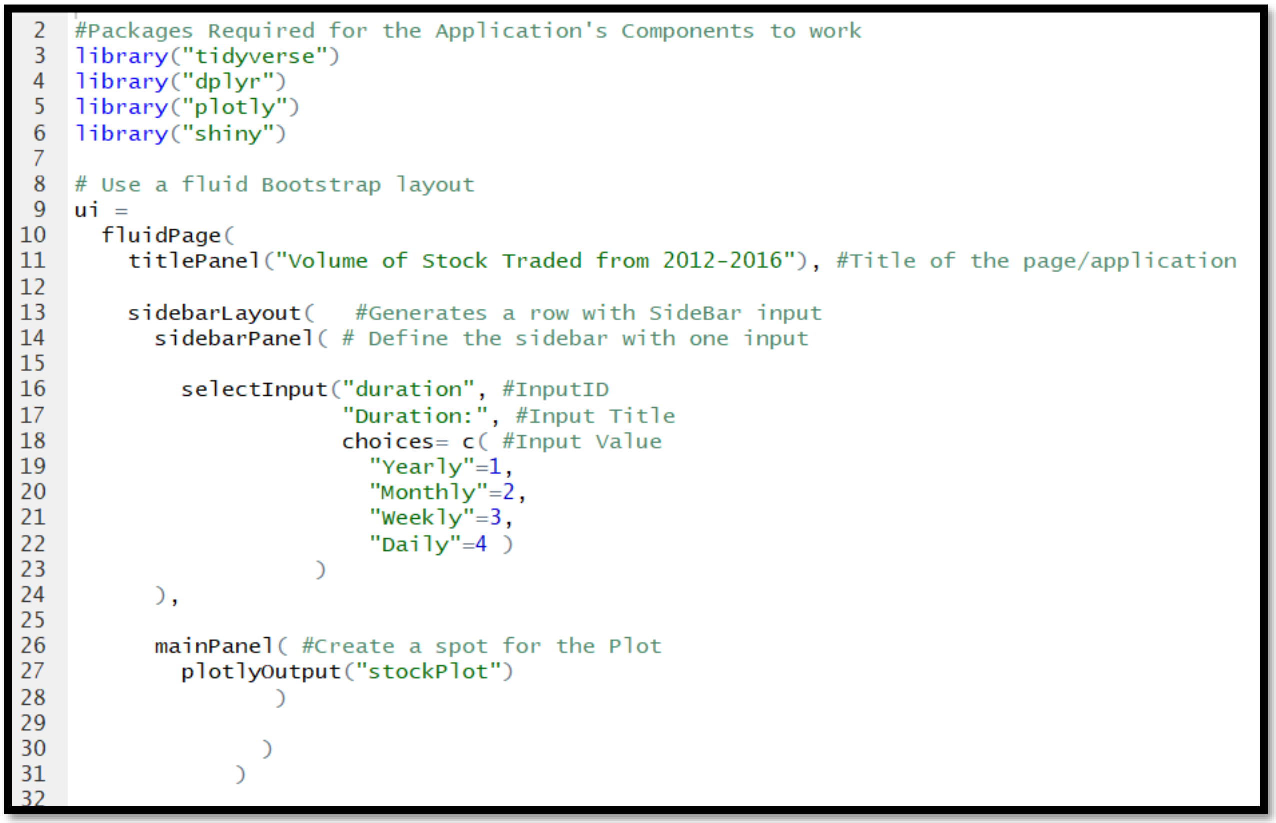
Application 1: Volume of Stocks Traded

Link: https://stockmarket.shinyapps.io/Volume_Of_Stocks_Traded/

Description:

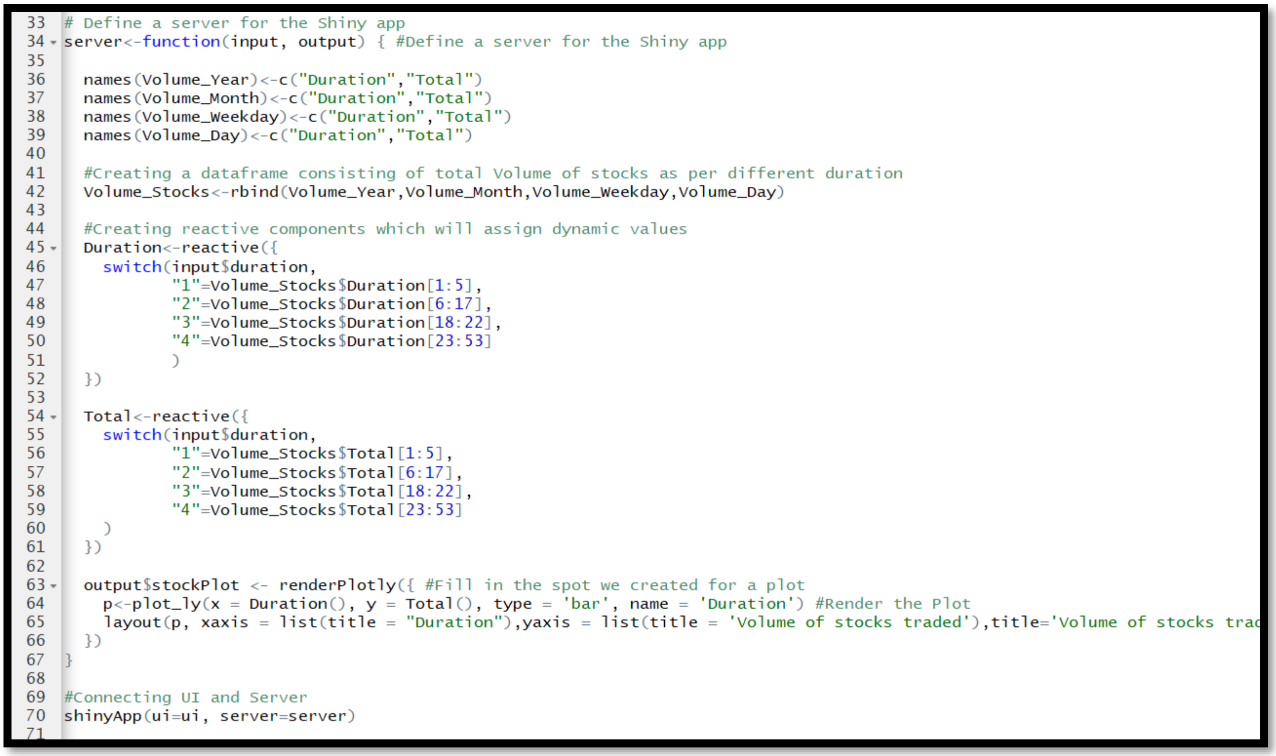
- This app displays the volume of stocks traded between 2012 to 2016 by all the S&P 500 Companies.
- The user can choose the duration from a SliderInput for which he/she wants to view the volume of stocks. Eg: Monthly
- This will display a plot for all the volume of stocks for the period of 2012 to 2016 by month.
- The graph is displayed using plotly so it has additional functions of downloading the graph as .png or taking a snapshot, variable zoom options and reset axes.

Code:



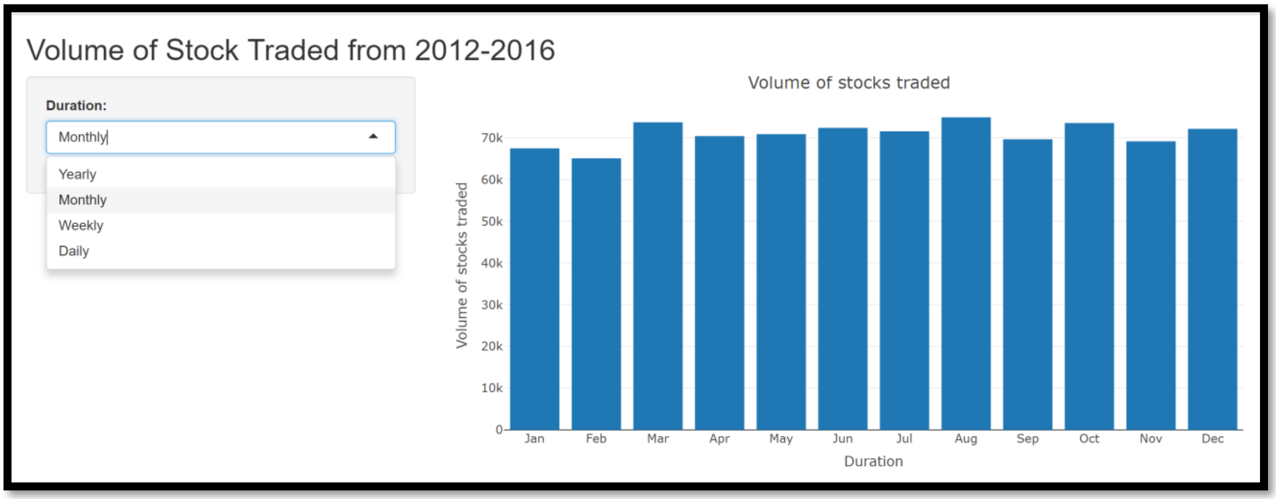
```r
#Packages Required for the Application's Components to work
library("tidyverse")
library("dplyr")
library("plotly")
library("shiny")

# Use a fluid Bootstrap layout
ui =
  fluidPage(
    titlePanel("Volume of Stock Traded from 2012-2016"), #Title of the page/application

    sidebarLayout(   #Generates a row with SideBar input
      sidebarPanel( # Define the sidebar with one input

        selectInput("duration", #InputID
                    "Duration:", #Input Title
                    choices= c( #Input Value
                      "Yearly"=1,
                      "Monthly"=2,
                      "Weekly"=3,
                      "Daily"=4 )
                  )
      ),

      mainPanel( #Create a spot for the Plot
        plotlyOutput("stockPlot")
               )

              )
            )
```

User Interface R Code for Volume of Stock Traded Analysis



```r
# Define a server for the Shiny app
server<-function(input, output) { #Define a server for the Shiny app

  names(Volume_Year)<-c("Duration","Total")
  names(Volume_Month)<-c("Duration","Total")
  names(Volume_Weekday)<-c("Duration","Total")
  names(Volume_Day)<-c("Duration","Total")

  #Creating a dataframe consisting of total Volume of stocks as per different duration
  Volume_Stocks<-rbind(Volume_Year,Volume_Month,Volume_Weekday,Volume_Day)

  #Creating reactive components which will assign dynamic values
  Duration<-reactive({
    switch(input$duration,
           "1"=Volume_Stocks$Duration[1:5],
           "2"=Volume_Stocks$Duration[6:17],
           "3"=Volume_Stocks$Duration[18:22],
           "4"=Volume_Stocks$Duration[23:53]
           )
  })

  Total<-reactive({
    switch(input$duration,
           "1"=Volume_Stocks$Total[1:5],
           "2"=Volume_Stocks$Total[6:17],
           "3"=Volume_Stocks$Total[18:22],
           "4"=Volume_Stocks$Total[23:53]
    )
  })

  output$stockPlot <- renderPlotly({ #Fill in the spot we created for a plot
    p<-plot_ly(x = Duration(), y = Total(), type = 'bar', name = 'Duration') #Render the Plot
    layout(p, xaxis = list(title = "Duration"),yaxis = list(title = 'Volume of stocks traded'),title='Volume of stocks trad
  })
}

#Connecting UI and Server
shinyApp(ui=ui, server=server)
```

Server R Code for RShiny Application 1.



Volume of Stock Traded from 2012-2016

Duration: Monthly

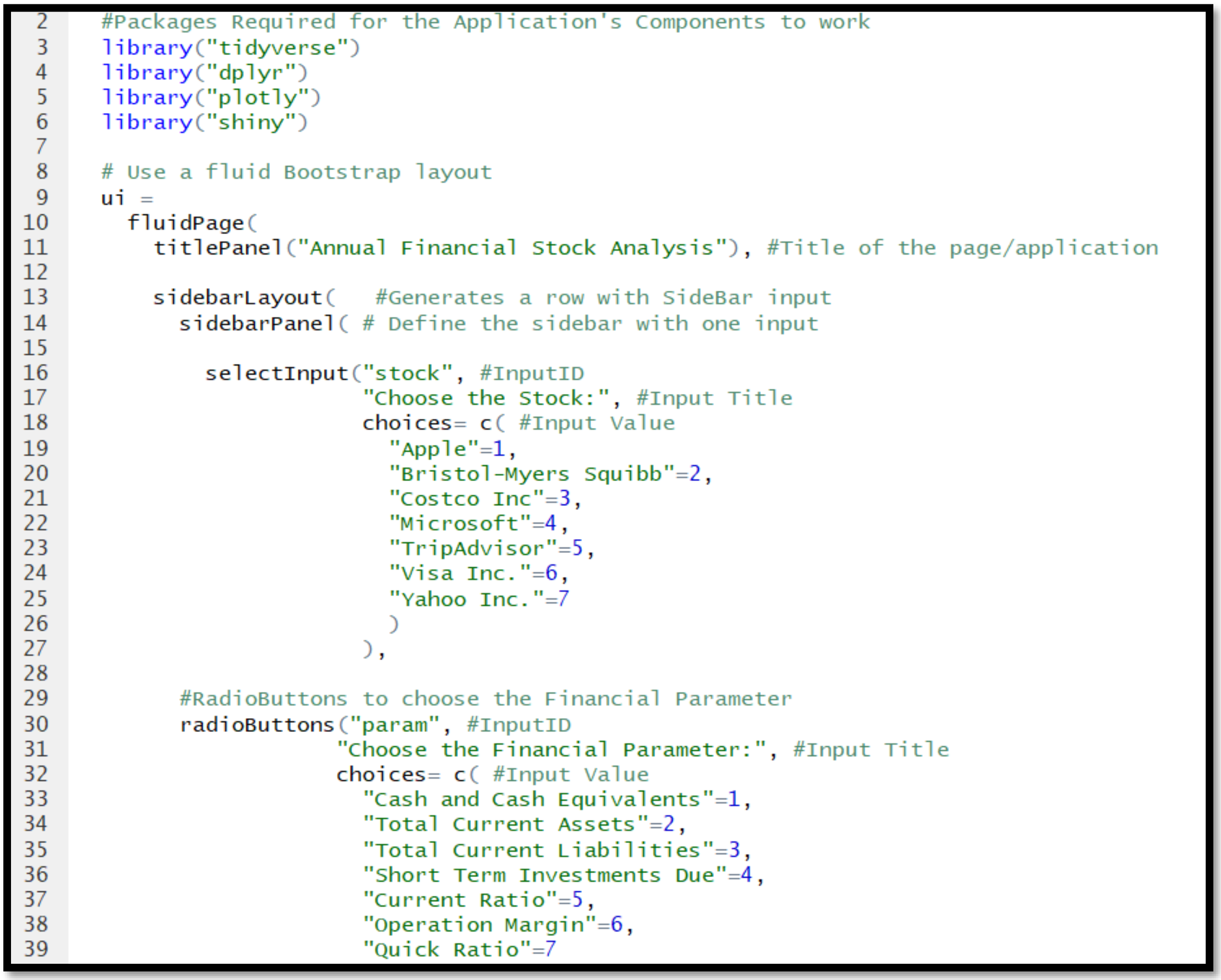
Shiny Application 1 displaying Volume of Stocks Traded with Scroll Down options to select Duration

Application 2: Annual Financial Stock Analysis
Link: https://stockmarket.shinyapps.io/Annual_Financial_Stock_Analysis/
Description:

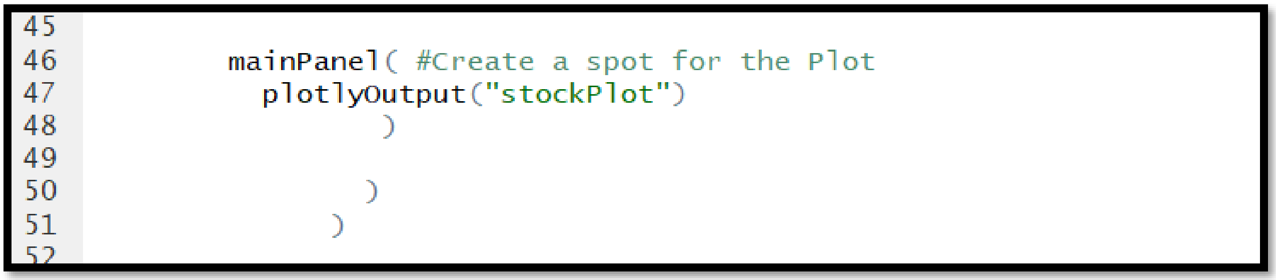
- This app displays the different financial parameters to choose for analysis.
- Using the Radio Button, the user can choose the Financial parameters such as Operation Margin, Current Ratio, Total Current Assets and Liabilities and so on.
- A Slider Input allows the user to pick the Stock Company in particular for which he/she wants to perform the analysis.

- The graph has a fixed X-Axis representing the financial year ranging from 2013 to 2016.
- This graph is also displayed using plot function of R.

Code:

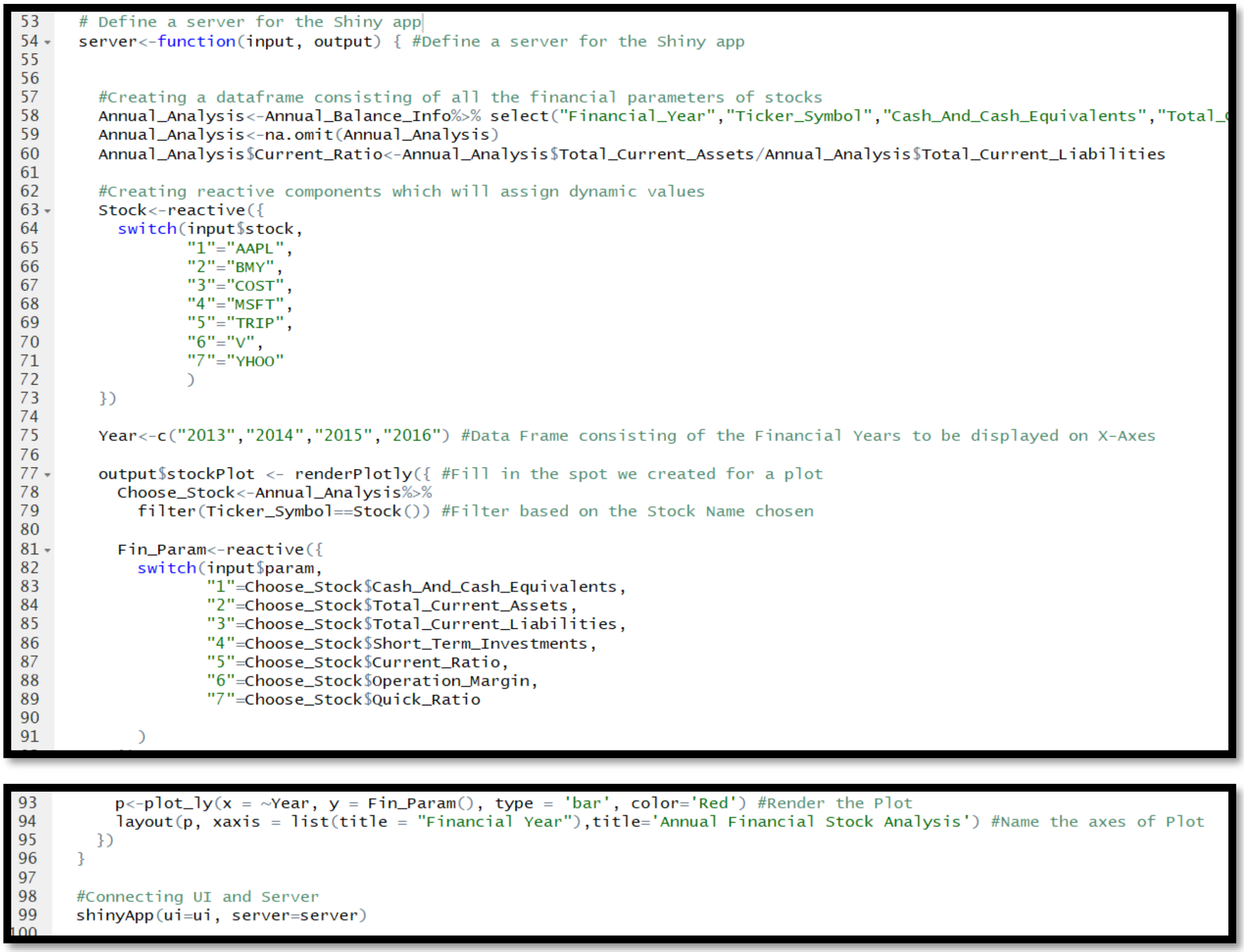


```
#Packages Required for the Application's Components to work
library("tidyverse")
library("dplyr")
library("plotly")
library("shiny")

# Use a fluid Bootstrap layout
ui =
  fluidPage(
    titlePanel("Annual Financial Stock Analysis"), #Title of the page/application

    sidebarLayout(   #Generates a row with SideBar input
      sidebarPanel( # Define the sidebar with one input

        selectInput("stock", #InputID
                    "Choose the Stock:", #Input Title
                    choices= c( #Input Value
                      "Apple"=1,
                      "Bristol-Myers Squibb"=2,
                      "Costco Inc"=3,
                      "Microsoft"=4,
                      "TripAdvisor"=5,
                      "Visa Inc."=6,
                      "Yahoo Inc."=7
                      )
                    ),

      #RadioButtons to choose the Financial Parameter
      radioButtons("param", #InputID
                  "Choose the Financial Parameter:", #Input Title
                  choices= c( #Input Value
                    "Cash and Cash Equivalents"=1,
                    "Total Current Assets"=2,
                    "Total Current Liabilities"=3,
                    "Short Term Investments Due"=4,
                    "Current Ratio"=5,
                    "Operation Margin"=6,
                    "Quick Ratio"=7
```



```

        mainPanel( #Create a spot for the Plot
          plotlyOutput("stockPlot")
                 )

                )
              )
```

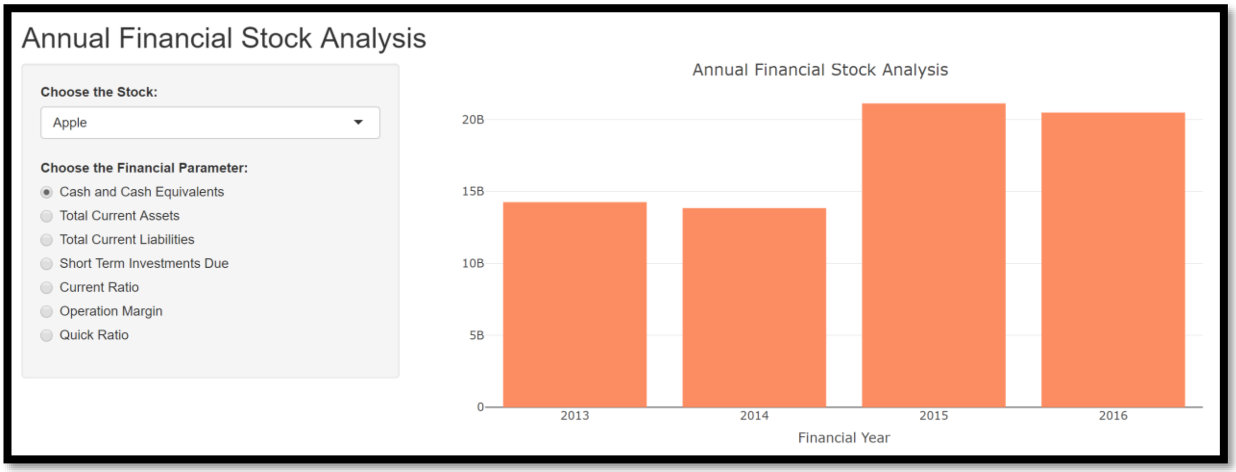
User Interface Code for Annual Stock Financial Analysis



```
# Define a server for the Shiny app
server<-function(input, output) { #Define a server for the Shiny app


  #Creating a dataframe consisting of all the financial parameters of stocks
  Annual_Analysis<-Annual_Balance_Info%>% select("Financial_Year","Ticker_Symbol","Cash_And_Cash_Equivalents","Total_C
  Annual_Analysis<-na.omit(Annual_Analysis)
  Annual_Analysis$Current_Ratio<-Annual_Analysis$Total_Current_Assets/Annual_Analysis$Total_Current_Liabilities

  #Creating reactive components which will assign dynamic values
  Stock<-reactive({
    switch(input$stock,
           "1"="AAPL",
           "2"="BMY",
           "3"="COST",
           "4"="MSFT",
           "5"="TRIP",
           "6"="V",
           "7"="YHOO"
           )
  })

  Year<-c("2013","2014","2015","2016") #Data Frame consisting of the Financial Years to be displayed on X-Axes

  output$stockPlot <- renderPlotly({ #Fill in the spot we created for a plot
    Choose_Stock<-Annual_Analysis%>%
      filter(Ticker_Symbol==Stock()) #Filter based on the Stock Name chosen

    Fin_Param<-reactive({
      switch(input$param,
             "1"=Choose_Stock$Cash_And_Cash_Equivalents,
             "2"=Choose_Stock$Total_Current_Assets,
             "3"=Choose_Stock$Total_Current_Liabilities,
             "4"=Choose_Stock$Short_Term_Investments,
             "5"=Choose_Stock$Current_Ratio,
             "6"=Choose_Stock$Operation_Margin,
             "7"=Choose_Stock$Quick_Ratio

      )
```

```
    p<-plot_ly(x = ~Year, y = Fin_Param(), type = 'bar', color='Red') #Render the Plot
    layout(p, xaxis = list(title = "Financial Year"),title='Annual Financial Stock Analysis') #Name the axes of Plot
  })
}

#Connecting UI and Server
shinyApp(ui=ui, server=server)
```

Server Code for Annual Financial Stock Analysis



Annual Financial Stock Analysis

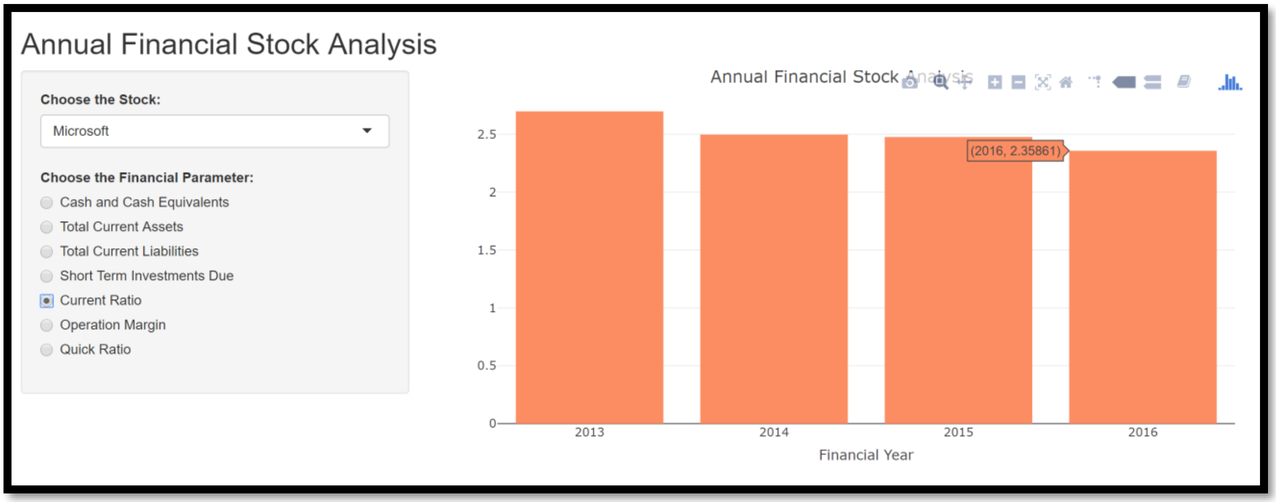
Choose the Stock:

Apple ▾

Choose the Financial Parameter:

- ● Cash and Cash Equivalents
- ○ Total Current Assets
- ○ Total Current Liabilities
- ○ Short Term Investments Due
- ○ Current Ratio
- ○ Operation Margin
- ○ Quick Ratio

Shiny App 2 for Annual Financial Stock Analysis



Annual Financial Stock Analysis

Choose the Stock:

Microsoft ▾

Choose the Financial Parameter:

- ○ Cash and Cash Equivalents
- ○ Total Current Assets
- ○ Total Current Liabilities
- ○ Short Term Investments Due
- ● Current Ratio
- ○ Operation Margin
- ○ Quick Ratio

Shiny Application 2: Selecting the Stocks to visualize different Annual Financial Stock Analysis.

8. Challenges Faced

1) Collecting data from heterogeneous sources and transforming them into a common format to work with.
2) Storing data in SQLite database. Since the data was collected from different sources each of them had different formats and types. So, while creating tables in the database it was critical to managing these different data types properly.
3) Deriving new columns by performing additional calculations. Since there was a high amount of missing and dirty data, the data first needed to be cleaned properly to carry out further calculations.
4) Working with Quantmod package to plot time series graph. Since, it was the first time we used the finance Quantmod package, to understand the components and plot it over a time series took a lot of effort.
5) Forecasting for additional 2 years using ARIMA model. Understanding how to evaluate the model to plot ACF and PACF.
6) It was extremely difficult building the R shiny app. Since this was the first time we were building the app from scratch. Tutorials from R Shiny website helped us a lot. The main challenge was to make the application more interactive and reactive.

9. Learning Outcomes

1) Scrapping, gathering, cleaning and consolidating techniques on the huge volume of data to fit 6 V's of big data is learned.
2) Creating meaningful and structured schema in its 3NF and then establishing an SQLite connection is skilled
3) Data retrieval is done using SQL queries and data visualization techniques are performed using ggplot2, plotly, ggthemes, highcharter packages
4) Data Mining Technique for Dynamic Time Series Analysis is performed.
5) Implemented forecasting technique using Prophet package and learned how to build ARIMA model to predict the data for future years.
6) Constructed a web application to perform parameter selection tool using RShiny.

10. Future Scope

1) Expanding the dataset by adding Stock Indexes, International Market Overview, Interest Rates etc. can increase the complexity in this project for which faster and efficient methods to collect, store and retrieve needs to be found.
2) Training the ARIMA model with more data to increase the accuracy. Also implementing other machine learning algorithms,
3) R Shiny app can be developed which more features and technicality. The user interface can be made more attractive.

11. References

- https://ntguardian.wordpress.com/2017/03/27/introduction-stock-market-data-r-1/
- http://www.visualcapitalist.com/difference-nyse-nasdaq/
- https://www.investopedia.com/terms/c/cashandcashequivalents.asp
- https://www.nerdwallet.com/blog/investing/stock-market-basics-everything-beginner-investors-know/
- http://www.visualcapitalist.com/difference-nyse-nasdaq/
- https://cran.r-project.org/web/packages/quantmod/quantmod.pdf
- https://dantonnoriega.github.io/ultinomics.org/post/2017-04-05-highcharter-explainer.html
- http://jkunst.com/highcharter/highstock.html
- https://www.datascience.com/blog/introduction-to-forecasting-with-arima-in-r-learn-data-science-tutorials
- http://ucanalytics.com/blogs/step-by-step-graphic-guide-to-forecasting-through-arima-modeling-in-r-manufacturing-case-study-example/
  https://shiny.rstudio.com/tutorial/