DATA ANALYTICS
Group- 20

Business Understanding:

Problem Domain: Stock market.

Target audience: Investors,Traders, Companies in stock market.

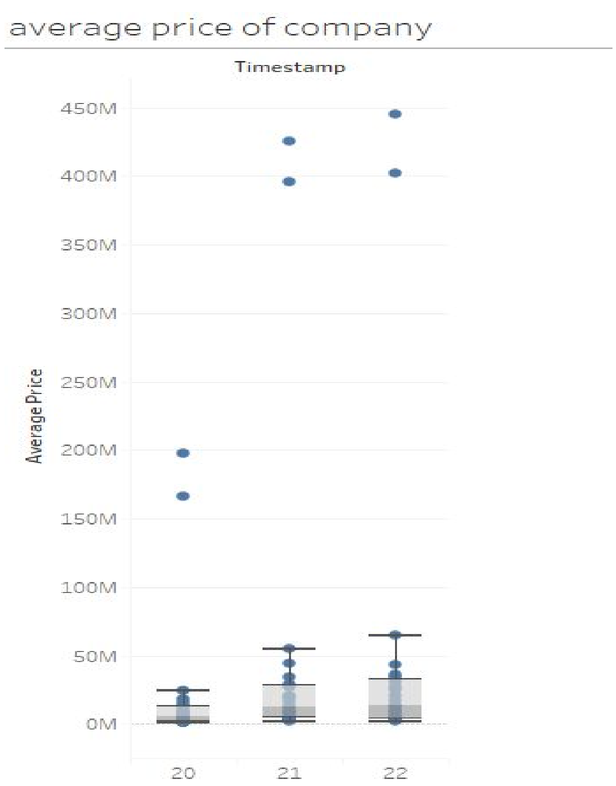
Goal: By analysing the given data we help investors, traders, companies to make buying and selling decisions in a way to gain an edge in the market.

Current (Existing) Solution:

Currently, to make any investment decision we are relying on stock brokers who provide us the current trends happening in the market which include their opinions. Based on their inputs we invest/buy or sell shares. With our approach we can minimise the role of stock brokers. This helps the users to know the trends in the market and also make unbiased decisions.

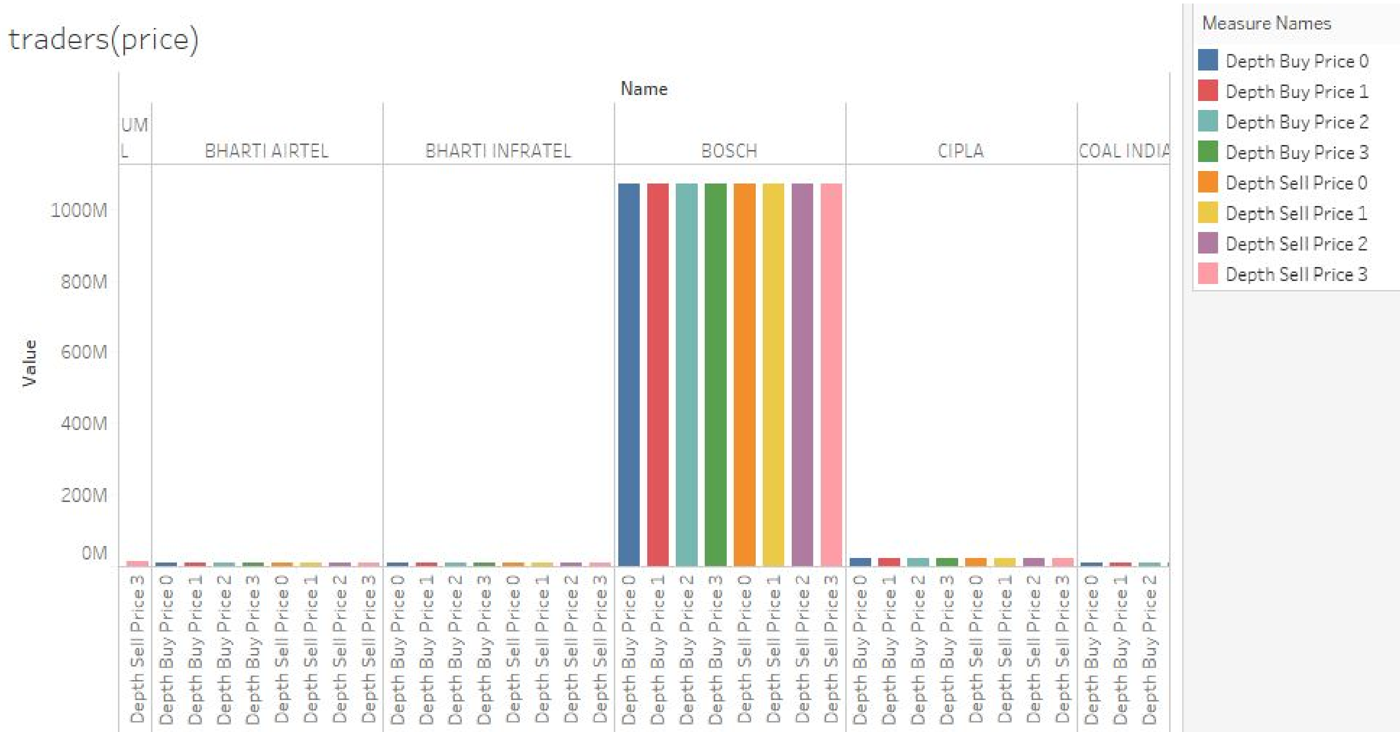
Data Understanding:

1. **Average price for different companies across various days. [Niharika]**



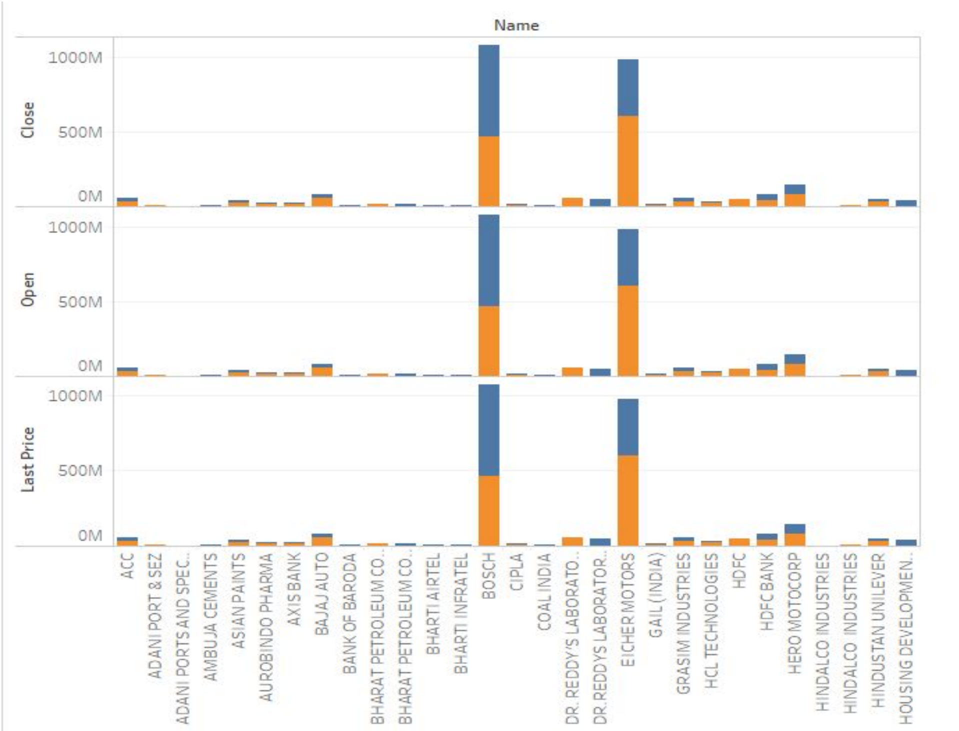
Target Users: Investors, companies
Explanation: BOSCHLTD, EICHERMOT has highest average price, BANK OF BARODA has least average price.

2. **Different depth (buy and sell) level prices: [Abhigna]**



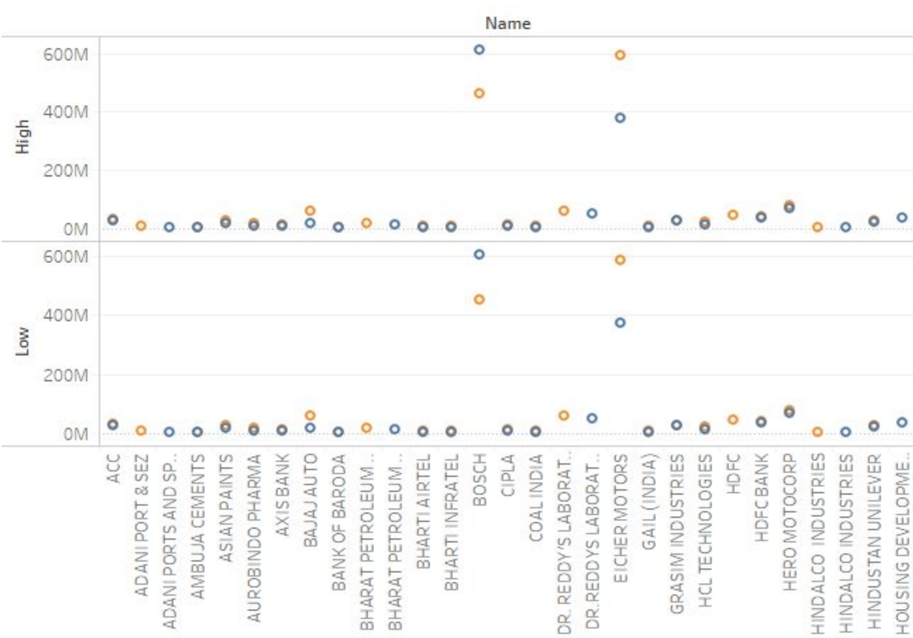
Target Users: Investors, companies
Explanation: BOSCH has highest depth level buy and depth level sell prices.

3. Open, close and last price of companies. [Nikunj]



Target users: Traders, companies
Explanation: Bosch has highest last price in all the three days.

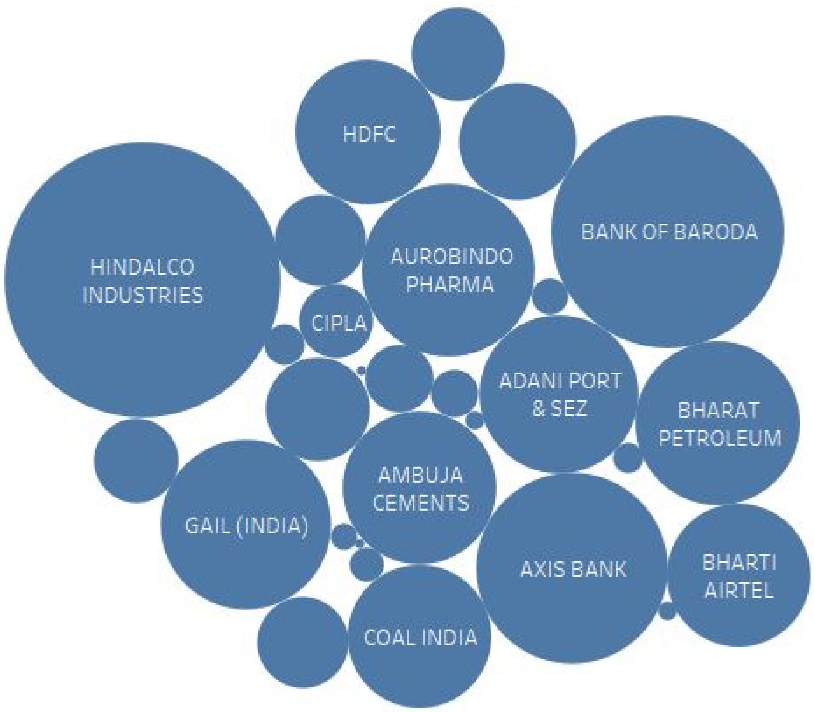
4. High and Low prices of companies [Mahati]



Target users: Investors

Explanation: Bosch, Eicher motors has high low and high prices. This can be used for analysing support and resistance values of a stock.

5. Volume of stocks for each company [Indu]



Target users: Traders, companies
Explanation: HINDALCO has highest volume traded.

Data Quality

Testing the data for the following Data Quality issues:

1. Missing data
2. Data errors
3. Measurement errors
4. Coding inconsistencies
5. Bad metadata

Data errors, measurement errors and bad metadata were not found in our dataset.
Some particular issues were found in the following:

1. Nulls: The dataset has some columns having nulls. The following table shows the percentage of nulls in all columns. The column 'name' has the maximum nulls.

2. Coding inconsistencies:
    a. The column ‘name’, expected to have the names of the stock’s company, was found to have nearly 80% nulls. So, it cannot be used in our data analysis. The column- ‘tradesymbol’ can be used to obtain company names, but the name of the stock companies were concatenated with the date, strike and instrument_type making it a data quality issue.

        Name: 73.98525 %
        Last_price: 37.8181 %
        Expiry: 25.82413 %
        Strike: 26.71046 %
        Lot_size: 23.13176 %

    b. Also, there is inconsistency in this pattern (the same pattern) is not found for all rows of a column the file.

Data Preparation

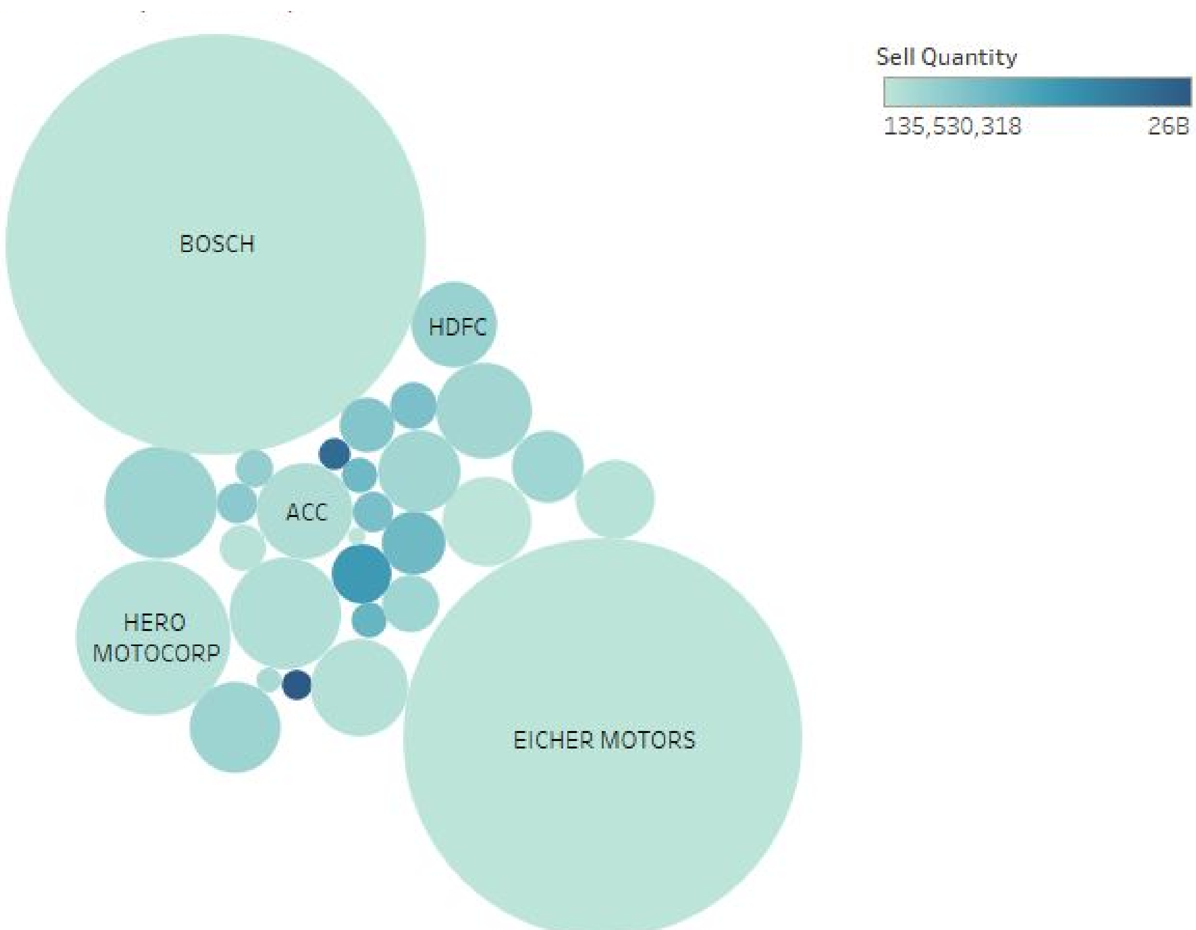
To address the above mentioned data quality issues we have written these R scripts:

1. For null values, there are 2 ways to go about. One way is to ignore the attribute having unacceptable number of null values. Another way is to fill these missing values.
2. For extracting company names from ‘tradesymbol’ the following R code was used. Now, we can use these company names.

```
> data <- read.csv("instruments.csv")
> library(stringr)
> library(dplyr)

Attaching package: ‘dplyr’

The following objects are masked from ‘package:stats’:

    filter, lag

The following objects are masked from ‘package:base’:

    intersect, setdiff, setequal, union

> write.csv(data %>% mutate(Year = str_extract(tradingsymbol, "[0-9]+"), NameOfCompany = str_extract(tradingsymbol, "[aA-zZ]+")), 'InstrumentsUpdated.csv')
> 
```

EXPLORATORY AND DESCRIPTIVE ANALYTICS

1. Companies success rate [Mahati, Indu]

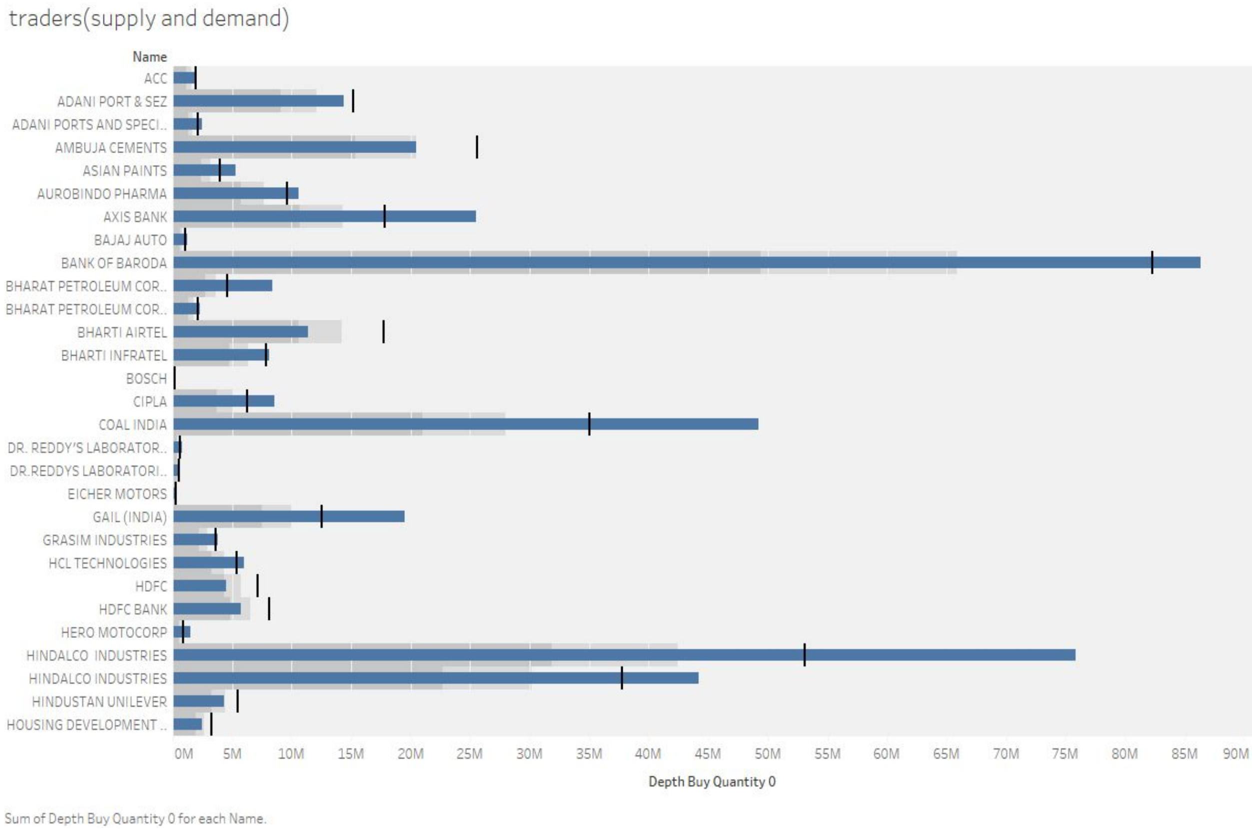


Name. Color shows sum of Sell Quantity. Size shows sum of Average Price. The marks are labeled by Name.

Target Users: Companies, Investors.

Explanation: The bigger circles (Bosch and Eicher Motors) have higher sum(average price). The companies with darkest color shade (HindAlCo Industries and Bank of Baroda) have higher sum(sell quantity). This useful to analyse the success of the companies.

2. Traders supply and demand quantity: [Niharika]



Target users - Traders

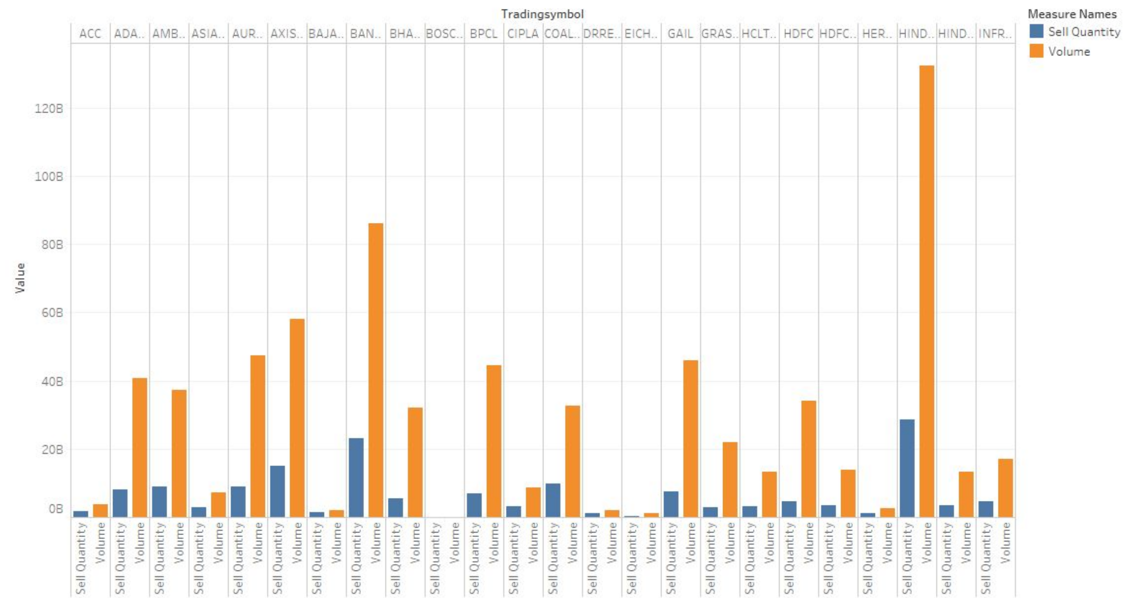
Explanation - The blue color coding shows depth buy quantity of a company.

Black color coding shows average depth sell quantity.

If depth buy quantity > depth sell quantity that means there is more demand in stocks of a company than the number of stocks available for investment. If depth buy quantity < depth sell quantity that means there is large set of people selling their existing stocks but, there is less demand for them.

3) Comparison of volume and sell quantities across various trade symbols. [Nikunj]

Sheet 1



Sell Quantity and Volume for each Tradingsymbol. Color shows details about Sell Quantity and Volume.

Target users: Investors, traders

Explanation:

Volume: quantity actually bought and sold during the day.

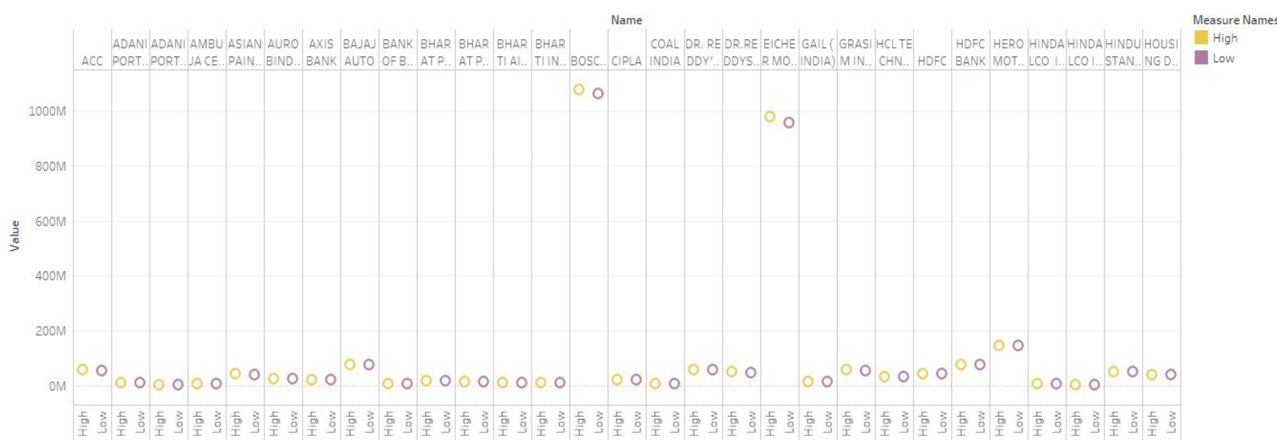
Sell quantity: quantity offered for sale by sellers but has not matched with a buyer at the end of day.

There is high volume and sell quantity for HINDALCO compared with other trade symbols

If sell quantity > volume that means there is no enough buying of stocks that is there is less demand for stocks. .

4) Resistance and support levels of various companies. [Abhigna, Indu]

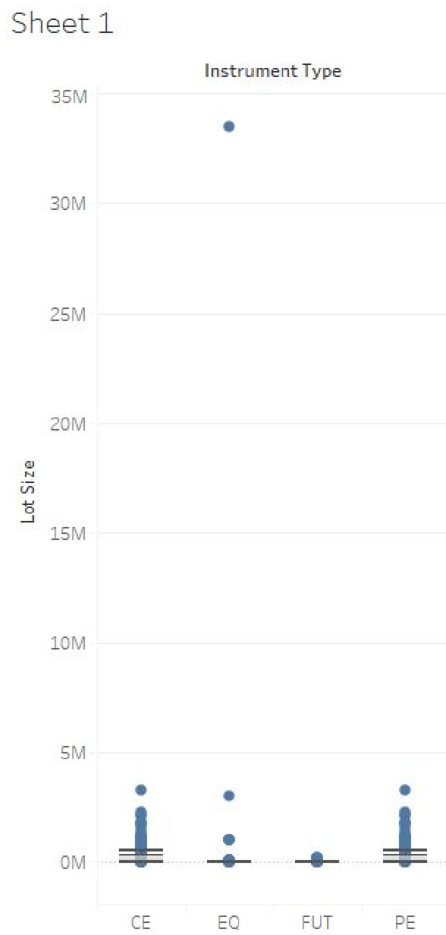
Sheet 4



High and Low for each Name. Color shows details about High and Low. The view is filtered on Name and sum of Low. The Name filter excludes Null. The sum of Low filter keeps non-Null values only.

Target users: companies
Explanation: We can check the resistance and support levels of various companies
Resistance level is basically highest price of a stock in a time period(here, taken as one day)
Low level is basically is basically lowest price of stock in a time period.

5) Long term and Short term investment suggestions [Abhigna]



Sum of Lot Size for each Instrument Type. Details are shown for Name Of Company.

Target Users: Investors
Explanation: We can tell the company with maximum shares during long term and short term investment.

6. Frequency of increase/ decrease in stock prices based on demand in the market. [Niharika]

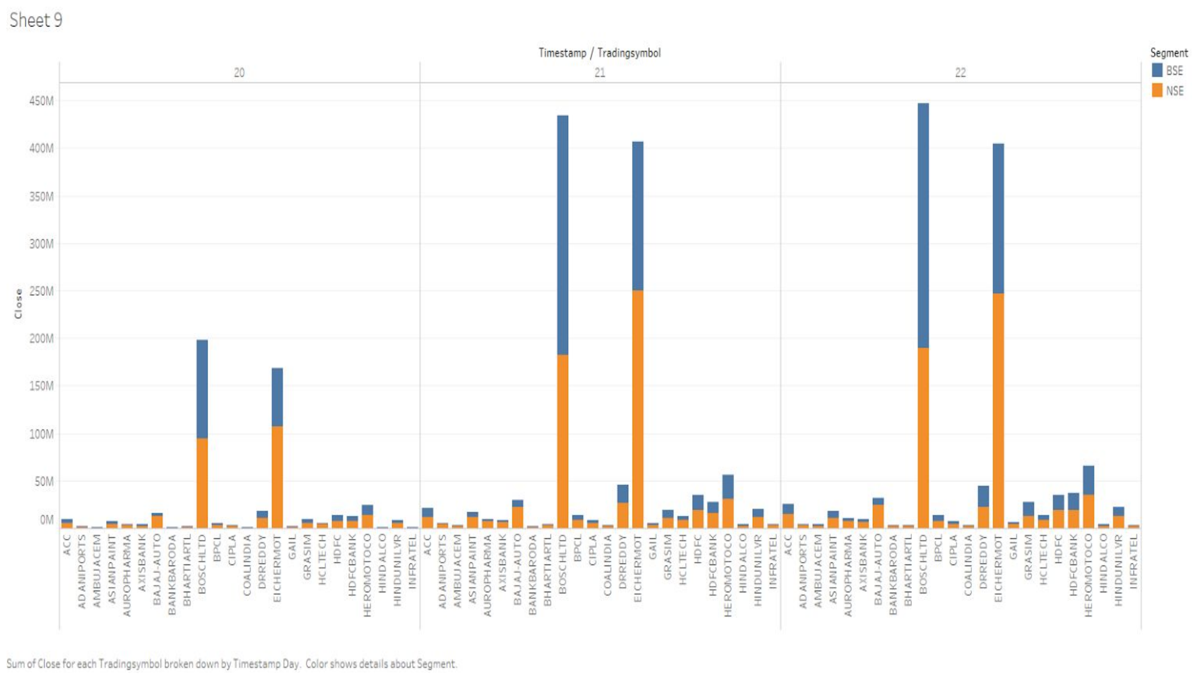
```
setwd("C:/Users/mahathi/Desktop/DAProject")
seeds <- read.csv("log_inf.csv")
seeds_1 <- read.csv("instrumentsUpdated.csv")
attach(seeds)
diff <- change
total <- merge(seeds,seeds_1,by="instrument_token")
col_1<-total[total$volume-total$sell_quantity>0,]
col_2<-total[total$volume-total$sell_quantity<0,]
unique_companies1 <- unique(col_1[c("NameOfCompany")])
unique_companies2 <- unique(col_2[c("NameOfCompany")])
freq1 <- aggregate(total$change>0,by=list(Nameofcompany=total$NameOfCompany),FUN=sum)
freq2 <- aggregate(total$change<0,by=list(Nameofcompany=total$NameOfCompany),FUN=sum)
colnames(freq1)[2] <- "Positive"
colnames(freq2)[2] <- "Negative"
merged_data <- merge(freq1,freq2,by = "Nameofcompany")
```

Output:

	Nameofcompany	Positive	Negative
1	ACC	24619	9612
2	ADANIPORTS	16724	14873
3	AMBUJACEM	20521	10824
4	ASIANPAINT	29963	6122
5	AUROPHARMA	22433	14970
6	AXISBANK	17468	25175
7	BAJAJ	13622	13623
8	BANKBARODA	23152	12993
9	BHARTIARTL	15112	12062
10	BOSCHLTD	18001	25895
11	BPCL	383	47867
12	CIPLA	13950	22757
13	COALINDIA	2765	27636
14	DRREDDY	14228	25884
15	EICHERMOT	14436	20731
16	GAIL	1	36585
17	GRASIM	32268	16654
18	HCLTECH	17495	20292
19	HDFC	21829	29277
20	HDFCBANK	30444	15324
21	HEROMOTOCO	1525	37201
22	HINDALCO	10982	35605
23	HINDUNILVR	20552	25411
24	INFRATEL	12993	10393

Inference: In the above table, for a company “positive” indicates number of time there is increase in stock price based on increase in demand of company, “Negative” indicates number of time there is decrease in stock price based on decrease in demand of company. So, max(positive-negative) indicates there is high demand for company across the period. ASIANPAINT, has highest number of positive-negative fluctuations in price, this means there was considerable amount of positive demand for that company.

7) Based on segment, opening and closing price of various companies across different days. [Nikunj]



Target: Investors, Traders
Explanation: In day1, day2, day3, BOSCHLTD has highest opening and closing price in both NSE and BSE segments.

CLASSIFICATION

1. **Classification for selling-buying trend**

```
library(rpart)
setwd("C:/Users/mahathi/Desktop/DAProject")
seeds <- read.csv("log_inf.csv",sep=",")
library(caret)
set.seed(789)
bolval <- seeds$volume >seeds$sell_quantity
Lab = factor(bolval, levels = c(FALSE, TRUE), labels = c("BUY", "SELL"))
```

```
seeds$Labels <- Lab
ind <- sample(2, nrow(seeds), replace = TRUE, prob=c(0.7,0.30))
train.data <- seeds[ind==1,]
test.data <- seeds[ind==2,]
library(e1071)
#col <- seeds[,13:41]
model <- naiveBayes(train.data$Labels ~ (depth_buy_quantity_0 + average_price), data =
train.data)
preds <- predict(model, newdata = test.data)
#library("caret")
conf_matrix <- confusionMatrix(preds, test.data$Labels)
print(conf_matrix)
```

Results:

```
          Reference
Prediction    BUY    SELL
      BUY   8151  15331
      SELL  64678 186724

              Accuracy : 0.7089
                95% CI : (0.7072, 0.7106)
```

2. Classification:

```
> seedsc = read.csv("~/Desktop/Acads/7thsem/DA/Project-data/1-m-real-time-stock-market-data-nse-bse/log_inf.csv")
> bolvalc <- seedsc$volume >seedsc$sell_quantity
> Labc = factor(bolvalc, levels = c(FALSE, TRUE), labels = c("SELL", "BUY"))
> seedsc$Labels <- Labc
> indc <- sample(2, nrow(seedsc), replace = TRUE, prob=c(0.7,0.30))
> train.datac <- seedsc[indc==1,]
> test.datac <- seedsc[indc==2,]
> myf <- Labels ~ depth_buy_quantity_0+depth_sell_quantity_0
> seeds_ctree <- ctree(myf, data=train.datac)
> table(predict(seeds_ctree), train.datac$Labels)

> testpred <- predict(seeds_ctree, newdata=test.datac)
> table(testpred, test.datac$Labels)

testpred   SELL    BUY
    SELL   4513   2627
    BUY   68782 199269
```

Accuracy = 74.1%

3. Classification for increase/decrease in stocks demand [Niharika]

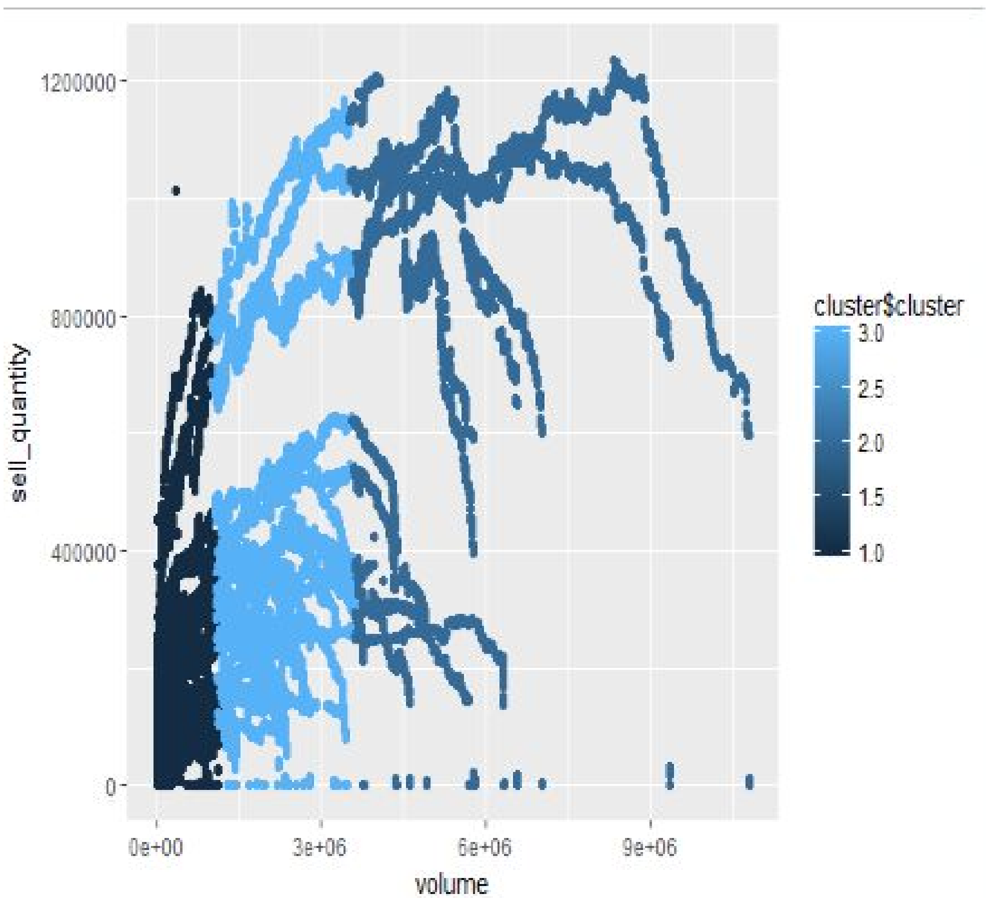
```
library(e1071)
library(party)
set.seed(123)
setwd("C://Users/iiitb/Desktop")
seeds <- read.csv("log_inf.csv",sep=",")
trainSize <- round(nrow(seeds) * 0.7)
testSize <- nrow(seeds) - trainSize
set.seed(123)
training_indices <- sample(seq_len(nrow(seeds)),
                           size=trainSize)

test_indices = -training_indices

attach(seeds)
diff <- depth_buy_quantity_0 - depth_sell_quantity_0
high_1 <- ifelse(diff>0,"high_demand","low_demand")

seeds = data.frame(seeds,high_1)
#trainSet = data.frame(trainSet,high_1)
#testSet = data.frame(testSet,high_1)
trainSet <- seeds[training_indices, ]
testSet <- seeds[-training_indices, ]
test_high = high_1[training_indices]
tree_2 = ctree(high_1~change,data=trainSet)
testpred <- predict(tree_2,newdata=testSet)
table(testpred,testSet$high_1)

testpred      high_demand low_demand
  high_demand      111539      98391
  low_demand        31030      34057
```

Explanation: If there is increase in stock demand (by increase in buy quantity) then stock price increases which is represented via positive change(current_stock_price - prev_stock_price) attribute. Similarly if there is decrease in stock demand, then it’s represented via negative change (decrease in stock price)

Accuracy: 52.98%

2. Data Clustering
K-means clustering:

1.[Mahati]



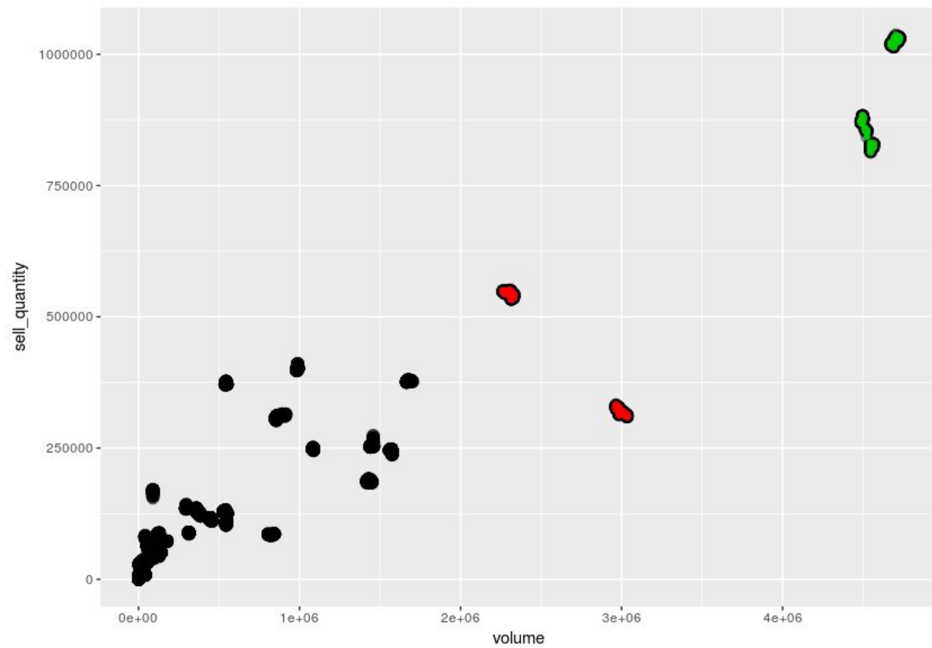

```
setwd("C:/Users/mahathi/Desktop/DAProject")
seeds <- read.csv("log_inf.csv",sep=",")
# Determine number of clusters
#set.seed(20)
cluster <- kmeans(seeds[,3:12], 3, nstart = 20)
cluster
library(ggplot2)
ggplot(seeds, aes(volume, sell_quantity, color = cluster$cluster)) + geom_point()
```

```
K-means clustering with 3 clusters of sizes 178317, 38361, 700044

Cluster means:
  last_price    volume sell_quantity last_quantity     change average_price
1   560.1771 1967640.1     376302.62      82.29863 -0.1300327      560.3834
2   238.4758 5170073.7     804516.07     145.68525 -0.8132469      239.2190
3  4058.9931  213664.8      83248.86      42.80991 -0.1303272     4021.0493
       open      high       low     close
1  559.5936  565.6492  554.5389  559.4278
2  240.3279  241.6394  236.9339  240.4798
3 4082.0922 4059.0492 3993.2631 4074.8552


Within cluster sum of squares by cluster:
[1] 8.192556e+16 7.991612e+16 6.076112e+16
 (between_SS / total_SS =  84.8 %)
```

K-means clustering was done on the basis of quantity and price attributes (column 3:12 in log_inf.csv). There are three clusters indicating less demand, intermediate demand and more demand.

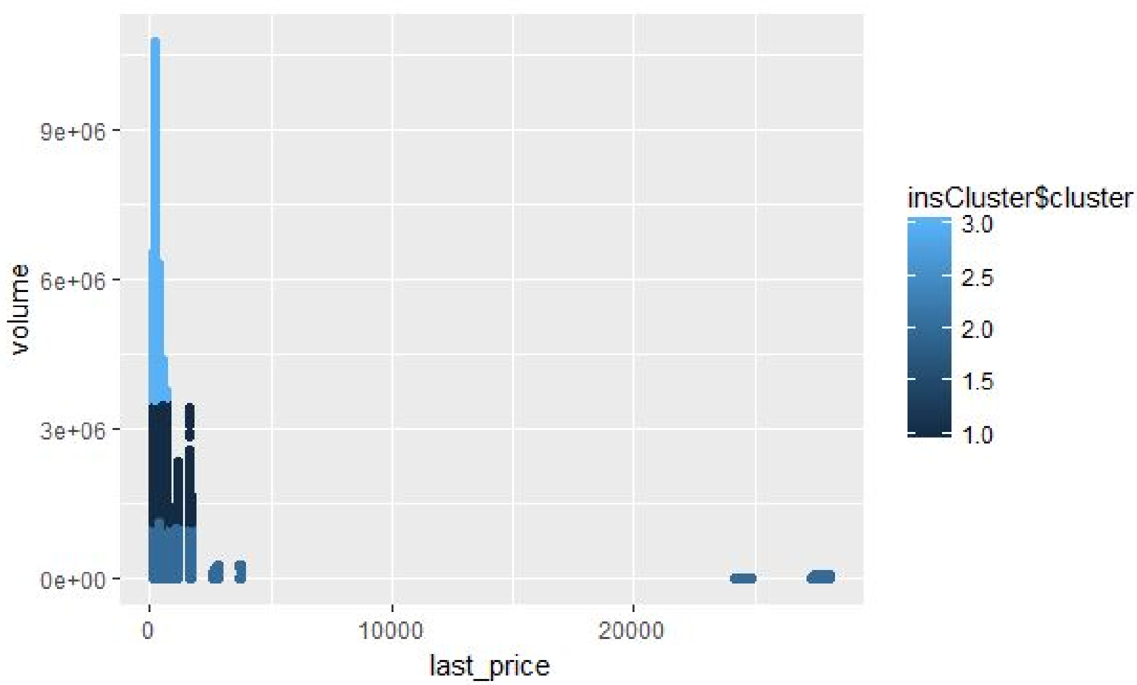
2. Hierarchical Clustering

```r
library(ggplot2)
seeds_log = read.csv("~/IIITB/Semesters/Sem 7/Data Analytics/Project/1-m-real-time-stock-market-data-nse-bse/log_inf.csv")
temp <- seeds_log[0:20000, 1:41]
hclusters <- hclust(dist(temp[, 4:5]))
clusterCut <- cutree(hclusters, 3)
ggplot(temp, aes(volume, sell_quantity)) + geom_point(alpha = 0.4, size = 3.5) + geom_point(col = clusterCut) + scale_color_manual(value


```



This is obtained using hierarchical clustering. In this method, we need not give the number of clusters before hand, unlike in kmeans. The method itself comes up with an appropriate number of clusters. (It is a “bottom-up” approach). We used clusterCut to cut the number of clusters formed to 3. The clusters are based on the volume and sell_quantity of a given stock.

3. K-means clustering



Code:

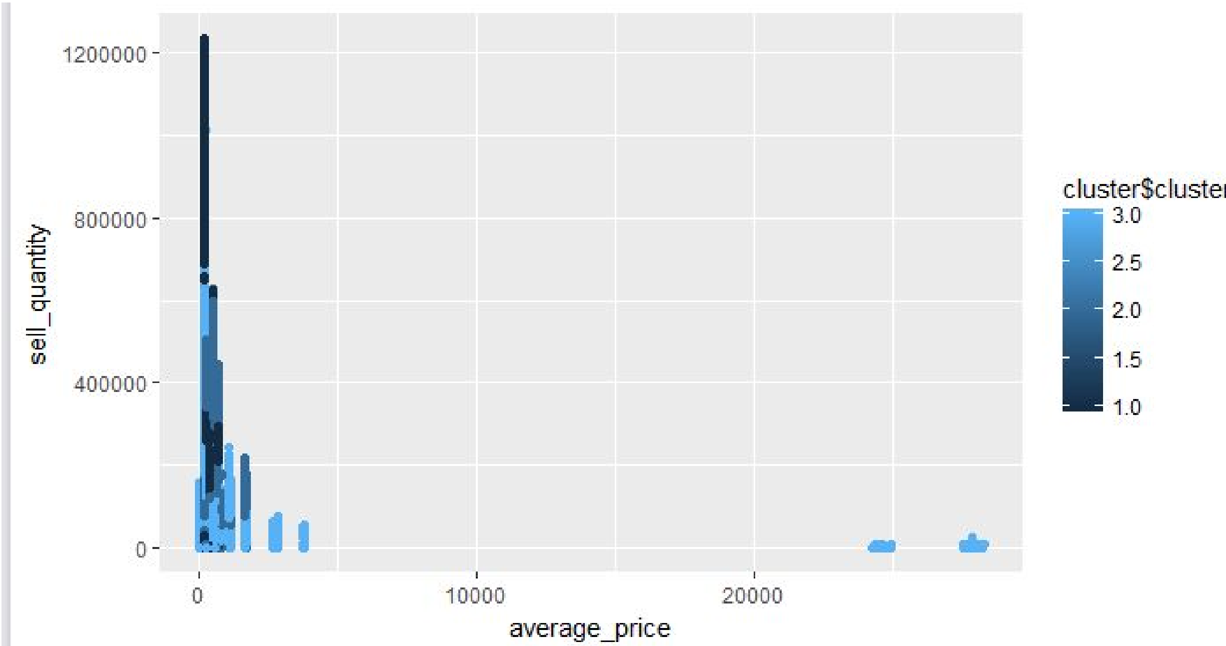
```
insCluster <- kmeans(seeds[3:7],2,nstart=20)
insCluster$cluster <- as.factor(insCluster$cluster)
ggplot(seeds,aes(last_price, volume, color=insCluster$cluster)) + geom_point()
```

Output:

```
K-means clustering with 2 clusters of sizes 134292, 782430

Cluster means:
  last_price    volume sell_quantity last_quantity     change
1   429.8724 3218963.0      548498.3     102.06163 -0.2993010
2  3697.1776  340588.5      105545.7      46.68358 -0.1347406

Clustering vector:
   [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 1 2 2
  [49] 2 1 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2
  [97] 1 1 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 1 2 1
 [145] 1 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 1 2
 [193] 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 1
 [241] 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2
 [289] 2 2 2 2 2 1 2 2 2 2 1 2 2 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 1 2 2 1 2 2 2 2 2 2 2
 [337] 1 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 1 1 1 1
 [385] 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 1 2 2 2 2 2 1 2 2 2 2 2 2
 [433] 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 1 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 1
 [481] 1 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 1 2 1 2 2 1 2 2 2 2 2 2 2 2 2 1 2 2 2 2
 [529] 2 1 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2
 [577] 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 1 2 2 2 2
 [625] 1 2 2 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 2 2 1 2 2
 [673] 1 2 2 1 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2
 [721] 1 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 [769] 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 1 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 1 2 1 2 2 2 2 2 2 1 2 2
 [817] 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 1 2 2 2 2 2 2 2 1 2 2 2 1 2 2 2 2 2 1 1 2 2 2 2 2 1 2 2 2 2 2
 [865] 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 1 2 1 2 2 2 2 2 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2
 [913] 2 1 1 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1
 [961] 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2
 [ reached getOption("max.print") -- omitted 915722 entries ]
Levels: 1 2

Within cluster sum of squares by cluster:
[1] 3.173527e+17 1.762961e+17
 (between_SS / total_SS =  66.3 %)
```

Clustering is done on the basis of 8 attributes from the dataset, this plot represents the clustering of different data points based on their volume and last_price attributes.

4. K-means clustering - average price and sell_quantity across various clusters [Niharika]



Clustering is done on the basis of 8 attributes from the dataset, this plot represents the clustering of different data points based on their average_price and sell_quantity attributes.

Code:

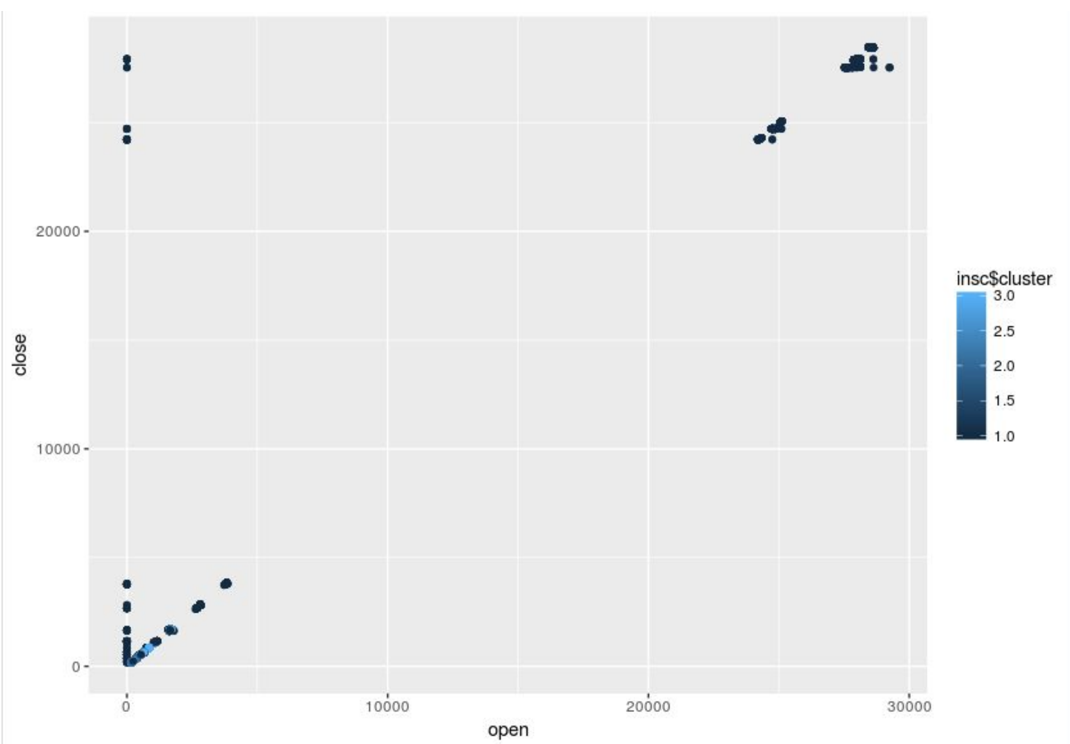
```
setwd("C://Users/iiitb/Desktop")
seeds <- read.csv("log_inf.csv",sep=",")
# Determine number of clusters
#set.seed(20)
cluster <- kmeans(seeds[,3:12], 3, nstart = 20)
cluster
library(ggplot2)
ggplot(seeds, aes(average_price, sell_quantity, color = cluster$cluster)) + geom_point()
```

Output:

```
K-means clustering with 3 clusters of sizes 38361, 700044, 178317

Cluster means:
  last_price    volume sell_quantity last_quantity     change average_price      open      high       low     close
1   238.4758 5170073.7     804516.07     145.68525 -0.8132469      239.2190  240.3279  241.6394  236.9339  240.4798
2  4058.9931  213664.8      83248.86      42.80991 -0.1303272     4021.0493 4082.0922 4059.0492 3993.2631 4074.8552
3   560.1771 1967640.1     376302.62      82.29863 -0.1300327      560.3834  559.5936  565.6492  554.5389  559.4278

Clustering vector:
   [1] 2 3 2 2 2 2 2 2 2 2 3 2 2 2 3 3 2 2 2 2 2 2 3 2 2 2 2 2 3 2 2 2 2 2 2 1 2 2 2 2 2 2 2 1 2 2 2 1 3 2 2 2 2 2 3 3 3 2 2 2 2 2
  [65] 2 3 2 2 2 3 3 2 3 2 3 2 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 3 2 3 3 3 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 3 2 2 2 1 2 2 3 2
 [129] 2 2 2 3 2 2 3 2 2 3 2 2 2 1 2 3 3 2 2 2 3 2 2 3 2 2 1 2 2 2 2 2 3 2 2 3 2 2 2 2 3 2 2 2 1 2 2 2 2 2 3 3 3 2 2 2 3 2 2 2 2 2 1 3
 [193] 2 2 2 2 3 2 2 2 2 3 2 2 2 2 2 3 2 2 2 1 2 2 2 2 2 2 2 3 2 3 2 2 2 2 2 2 3 2 2 2 3 3 2 2 3 2 2 1 3 2 2 2 2 2 3 2 2 2 3 2 2 3 3 2
 [257] 2 2 2 3 2 2 2 2 2 2 3 2 2 2 3 1 2 2 2 2 3 2 3 3 2 3 2 2 3 2 2 2 3 2 2 3 2 1 2 2 2 2 1 2 2 2 3 3 2 3 2 2 1 2 2 2 2 2 3 2 3 2 2 2
 [321] 2 2 3 2 2 1 2 2 1 2 2 3 2 2 2 2 3 2 2 2 2 2 2 2 3 3 2 2 2 3 2 2 2 3 2 2 2 2 3 3 2 2 2 2 2 2 2 3 2 2 2 2 1 3 2 2 2 2 2 2 3 3 3 3
 [385] 2 2 2 2 2 3 2 1 2 2 2 2 2 3 2 3 2 2 2 2 2 3 2 2 2 2 1 2 2 3 2 2 2 2 2 1 2 2 2 3 2 3 2 2 3 2 3 2 2 2 2 2 2 3 2 3 2 2 2 1 2 2 2
 [449] 3 2 2 3 3 2 2 2 2 3 3 2 2 3 2 2 3 2 2 2 1 2 2 2 3 2 2 2 2 2 2 1 1 2 2 2 3 2 2 3 2 3 2 3 2 2 2 3 3 2 2 2 3 2 2 3 2 2 2 2 1 2 1 2
 [513] 2 3 3 2 3 2 2 2 2 3 2 1 2 3 2 2 2 3 2 2 2 2 2 3 2 3 3 2 2 2 3 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 1 2 2 3 3 2 2 3 2 2 2 3 2 2 3 2 2
 [577] 3 3 2 2 2 3 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 3 3 2 2 2 2 1 2 2 3 2 2 2 2 3 2 2 3 2 2 2 2 2 3 2 2 2 2 3 2 2 2 3 2 2 2 2 3 2 2 2 1 1 2
 [641] 2 2 2 2 2 2 2 2 3 2 2 2 2 3 2 2 2 3 3 2 2 2 2 2 2 2 2 2 3 2 1 2 2 1 2 2 3 2 2 2 2 3 2 3 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 3 2 2 1 2 2 2
 [705] 3 2 2 2 3 2 2 2 2 2 2 2 2 2 1 2 2 3 2 2 2 2 2 2 2 2 2 3 2 2 2 3 2 2 2 3 2 2 2 2 2 2 1 3 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 3 2 2 3 3 2 2
 [769] 2 2 2 2 2 2 2 2 2 3 3 2 2 2 3 2 2 2 2 2 1 2 2 3 2 2 2 2 3 2 2 2 3 2 2 2 2 3 2 1 2 3 2 2 2 2 3 2 2 3 3 2 2 2 3 3 2 2 2 2 2 2 2 2 1
 [833] 2 3 2 2 2 2 3 2 2 3 2 2 2 3 3 3 2 2 2 1 3 2 2 2 3 2 1 2 2 2 2 3 3 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 3 3 2 1 2 3 2 2 2 2 2 1 2 2 2 2 3 2
 [897] 2 2 2 2 2 2 2 2 2 2 2 1 2 3 2 2 2 1 3 2 2 2 3 2 1 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 1 2 2 2 3 2 2 2 3 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 3
 [961] 2 2 2 2 3 2 2 2 2 3 3 1 2 2 2 2 3 2 2 3 2 2 3 2 2 3 2 2 2 3 2 3 2 2 2 2 1 2 2 2 2
 [ reached getOption("max.print") -- omitted 915722 entries ]

Within cluster sum of squares by cluster:
[1] 7.991612e+16 6.076112e+16 8.192556e+16
 (between_SS / total_SS =  84.8 %)
```

6. K means clustering across various clusters. [Niharika]

Opening and closing prices across various clusters



Code:

```
insc <- kmeans(seeds[,3:12],3,nstart=20)
ggplot(seeds,aes(open,close,color=insc$cluster)) + geom_point()
```

Output:

```
K-means clustering with 3 clusters of sizes 38361, 700044, 178317

Cluster means:
  last_price    volume sell_quantity last_quantity     change average_price      open      high       low     close
1   238.4758 5170073.7     804516.07     145.68525 -0.8132469      239.2190  240.3279  241.6394  236.9339  240.4798
2  4058.9931  213664.8      83248.86      42.80991 -0.1303272     4021.0493 4082.0922 4059.0492 3993.2631 4074.8552
3   560.1771 1967640.1     376302.62      82.29863 -0.1300327      560.3834  559.5936  565.6492  554.5389  559.4278

Clustering vector:
   [1] 2 3 2 2 2 2 2 2 2 2 3 2 2 2 3 3 2 2 2 2 2 2 2 3 2 2 2 2 2 2 3 2 2 2 2 2 2 1 2 2 2 2 2 2 2 1 2 2 2 1 3 2 2 2 2 2 3 3 3 2 2 2 2 2
  [65] 2 3 2 2 2 3 3 2 3 2 3 2 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2 2 3 2 3 3 3 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 3 2 2 2 1 2 2 3 2
 [129] 2 2 2 3 2 2 3 2 2 3 2 2 2 1 2 3 3 2 2 2 3 2 2 3 2 2 1 2 2 2 2 2 3 2 2 3 2 2 2 2 2 3 2 2 2 1 2 2 2 2 2 3 3 3 2 2 2 3 2 2 2 2 2 1 3
 [193] 2 2 2 2 3 2 2 2 2 2 3 2 2 2 2 2 3 2 2 2 1 2 2 2 2 2 2 2 3 2 3 2 2 2 2 2 2 3 2 2 2 3 3 2 2 3 2 2 1 3 2 2 2 2 2 3 2 2 2 3 2 2 3 3 2
 [257] 2 2 2 3 2 2 2 2 2 2 3 2 2 2 3 1 2 2 2 2 3 2 3 3 2 3 2 2 3 2 2 2 3 2 2 3 2 1 2 2 2 2 1 2 2 2 3 3 2 3 2 2 1 2 2 2 2 2 3 2 3 2 2 2
 [321] 2 2 3 2 2 1 2 2 1 2 2 3 2 2 2 2 3 2 2 2 2 2 2 2 3 3 2 2 2 3 2 2 2 3 2 2 2 2 3 3 2 2 2 2 2 2 2 2 3 2 2 2 2 1 3 2 2 2 2 2 2 3 3 3 3
 [385] 2 2 2 2 2 3 2 1 2 2 2 2 2 3 2 3 2 2 2 2 2 3 2 2 2 2 1 2 2 3 2 2 2 2 2 1 2 2 2 3 2 3 2 2 3 2 3 2 2 2 2 2 2 2 3 2 3 2 2 2 1 2 2 2
 [449] 3 2 2 3 3 2 2 2 2 3 3 2 2 3 2 2 3 2 2 2 1 2 2 2 3 2 2 2 2 2 2 1 1 2 2 2 3 2 2 3 2 3 2 3 2 2 2 3 3 2 2 2 3 2 2 3 2 2 2 2 1 2 1 2
 [513] 2 3 3 2 3 2 2 2 2 3 2 1 2 3 2 2 2 3 2 2 2 2 2 3 2 3 3 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 1 2 2 3 3 2 2 3 2 2 2 3 2 2 3 2 2
 [577] 3 3 2 2 2 3 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 3 3 2 2 2 2 1 2 2 3 2 2 2 2 3 2 2 3 2 2 2 2 2 3 2 2 2 2 3 2 2 2 3 2 2 2 2 3 2 2 2 1 1 2
 [641] 2 2 2 2 2 2 2 2 3 2 2 2 2 3 2 2 2 3 3 2 2 2 2 2 2 2 2 3 2 1 2 2 1 2 2 3 2 2 2 3 2 3 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 3 2 2 1 2 2 2
 [705] 3 2 2 2 3 2 2 2 2 2 2 2 2 1 2 2 3 2 2 2 2 2 2 2 2 3 2 2 2 3 2 2 2 3 2 2 2 2 2 2 1 3 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 3 2 2 3 3 2 2
 [769] 2 2 2 2 2 2 2 2 2 3 3 2 2 2 3 2 2 2 2 2 1 2 2 3 2 2 2 2 3 2 2 2 3 2 2 2 3 2 1 2 3 2 2 2 2 3 2 2 3 3 2 2 2 3 3 2 2 2 2 2 2 2 2 2 1
 [833] 2 3 2 2 2 2 3 2 2 3 2 2 2 3 3 3 2 2 2 1 3 2 2 2 3 2 1 2 2 2 2 3 3 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 3 3 2 1 2 3 2 2 2 2 2 1 2 2 2 2 3 2
 [897] 2 2 2 2 2 2 2 2 2 2 2 1 2 3 2 2 2 1 3 2 2 2 3 2 1 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 1 2 2 2 3 2 2 2 3 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 3
 [961] 2 2 2 2 3 2 2 2 2 2 3 3 1 2 2 2 2 3 2 2 3 2 2 3 2 2 3 2 2 2 3 2 3 2 2 2 1 2 2 2 2
 [ reached getOption("max.print") -- omitted 915722 entries ]

Within cluster sum of squares by cluster:
[1] 7.991612e+16 6.076112e+16 8.192556e+16
 (between_SS / total_SS =  84.8 %)
```

8) CODE:

```
> logdataCluster <- kmeans(seeds[, 13:41], 2, nstart = 3)
> table(logdataCluster$cluster, seeds$Labels)

      SELL    BUY
  1  11322      0
  2 232442 672958
```

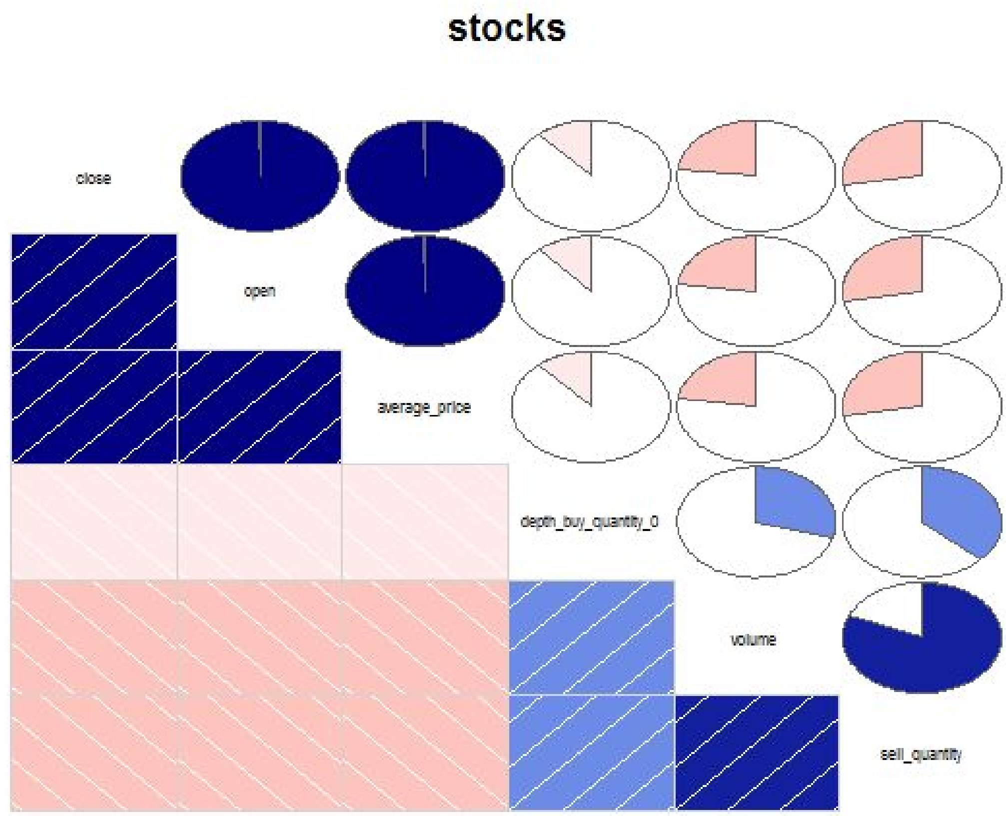
Accuracy for clusters formed is 74.6%.

```
> logdataCluster
K-means clustering with 2 clusters of sizes 11322, 905400

Cluster means:
  depth_buy_price_0 depth_buy_orders_0 depth_buy_quantity_0 depth_sell_price_0 depth_sell_orders_0
1          2536.654           2.007419             307.0905           2461.001            1.774333
2          3222.118           2.873454             466.5350           3226.997            3.158062
  depth_sell_quantity_0 depth_buy_price_1 depth_buy_orders_1 depth_buy_quantity_1 depth_sell_price_1
1              393.9172          2494.682           1.529147             158.8192           2469.036
2              405.8936          3221.094           3.315412             669.0426           3224.252
  depth_sell_orders_1 depth_sell_quantity_1 depth_buy_price_2 depth_buy_orders_2 depth_buy_quantity_2
1            1.596096              128.0077          2481.549           1.612436             170.2604
2            2.853247              567.9881          3220.581           4.439011             975.3496
  depth_sell_price_2 depth_sell_orders_2 depth_sell_quantity_2 depth_buy_price_3 depth_buy_orders_3
1           2477.032            1.288730              254.5117          2476.924           1.751899
2           3224.458            3.213095              838.4526          3220.192           4.865287
  depth_buy_quantity_3 depth_sell_price_3 depth_sell_orders_3 depth_sell_quantity_3 depth_buy_price_4
1             177.5196           2479.503            1.370253              189.7255       39550723.39
2            1251.9312           3224.805            3.435652             1100.2419           3217.88
  depth_buy_orders_4 depth_buy_quantity_4 depth_sell_price_4 depth_sell_orders_4
1           3.557410             364.2063        37889183.309            4.541954
2           5.086757            1400.5342            3223.336            3.579004

Clustering vector:
   [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

```
 Within cluster sum of squares by cluster:
 [1] 3.692886e+18 5.083246e+14
  (between_SS / total_SS =  90.1 %)
```

ASSOCIATION RULE MINING

```r
setwd("C:/Users/mahathi/Desktop/DAProject")
seeds <- read.csv("log_inf.csv")
library(corrgram)
cols <- c(4,5,8,9,12,15)
corrgram(seeds[,cols], order=TRUE, lower.panel=panel.shade,
         upper.panel=panel.pie, text.panel=panel.txt,
         main="stocks")
```



Explanation: Blue shade represents positive correlation and red shade represents negative correlation. The percentage of confidence is depicted from the pie charts. This gives a brief overview of how the attributes are related.

Association Code:

```
seeds = read.csv("~/Desktop/Acads/7thsem/DA/Project-data/1-m-real-time-stock-market-data-nse-bse/log_inf.csv")
set.seed(121)
price0 = seeds$depth_buy_price_0 > seeds$depth_sell_price_0
order0 = seeds$depth_buy_orders_0 > seeds$depth_sell_orders_0
quantity0 = seeds$depth_buy_quantity_0 > seeds$depth_sell_quantity_0
price0 = factor(price0, levels = c(FALSE, TRUE), labels = c("-ve", "+ve"))
seeds$price0 <- price0
order0 = factor(order0, levels = c(FALSE, TRUE), labels = c("-ve", "+ve"))
seeds$order0 <- order0
quantity0 = factor(quantity0, levels = c(FALSE, TRUE), labels = c("-ve", "+ve"))
seeds$quantity0 <- quantity0

price1 = seeds$depth_buy_price_1 > seeds$depth_sell_price_1
order1 = seeds$depth_buy_orders_1 > seeds$depth_sell_orders_1
quantity1 = seeds$depth_buy_quantity_1 > seeds$depth_sell_quantity_1
price1 = factor(price1, levels = c(FALSE, TRUE), labels = c("-ve", "+ve"))
seeds$price1 <- price1
order1 = factor(order1, levels = c(FALSE, TRUE), labels = c("-ve", "+ve"))
seeds$order1 <- order1
quantity1 = factor(quantity1, levels = c(FALSE, TRUE), labels = c("-ve", "+ve"))
seeds$quantity1 <- quantity1

price2 = seeds$depth_buy_price_2 > seeds$depth_sell_price_2
order2 = seeds$depth_buy_orders_2 > seeds$depth_sell_orders_2
quantity2 = seeds$depth_buy_quantity_2 > seeds$depth_sell_quantity_2
price2 = factor(price2, levels = c(FALSE, TRUE), labels = c("-ve", "+ve"))
seeds$price2 <- price2
```

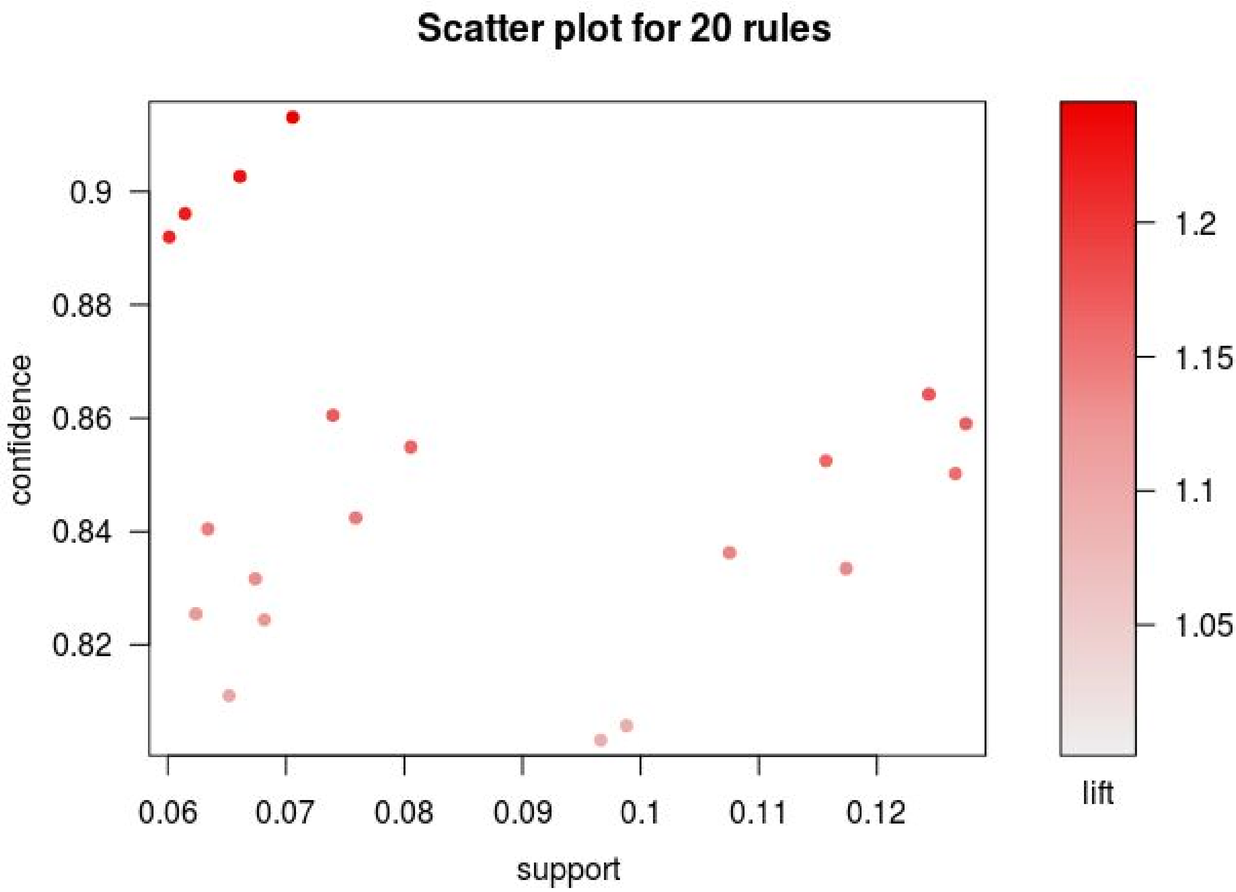
```
seeds$price2 <- price2
order2 = factor(order2, levels = c(FALSE, TRUE), labels = c("-ve", "+ve"))
seeds$order2 <- order2
quantity2 = factor(quantity2, levels = c(FALSE, TRUE), labels = c("-ve", "+ve"))
seeds$quantity2 <- quantity2

price3 = seeds$depth_buy_price_3 > seeds$depth_sell_price_3
order3 = seeds$depth_buy_orders_3 > seeds$depth_sell_orders_3
quantity3 = seeds$depth_buy_quantity_3 > seeds$depth_sell_quantity_3
price3 = factor(price3, levels = c(FALSE, TRUE), labels = c("-ve", "+ve"))
seeds$price3 <- price3
order3 = factor(order3, levels = c(FALSE, TRUE), labels = c("-ve", "+ve"))
seeds$order3 <- order3
quantity3 = factor(quantity3, levels = c(FALSE, TRUE), labels = c("-ve", "+ve"))
seeds$quantity3 <- quantity3

price4 = seeds$depth_buy_price_4 > seeds$depth_sell_price_4
order4 = seeds$depth_buy_orders_4 > seeds$depth_sell_orders_4
quantity4 = seeds$depth_buy_quantity_4 > seeds$depth_sell_quantity_4
price4 = factor(price4, levels = c(FALSE, TRUE), labels = c("-ve", "+ve"))
seeds$price4 <- price4
order4 = factor(order4, levels = c(FALSE, TRUE), labels = c("-ve", "+ve"))
seeds$order4 <- order4
quantity4 = factor(quantity4, levels = c(FALSE, TRUE), labels = c("-ve", "+ve"))
seeds$quantity4 <- quantity4

bolval <- seeds$volume >seeds$sell_quantity
```

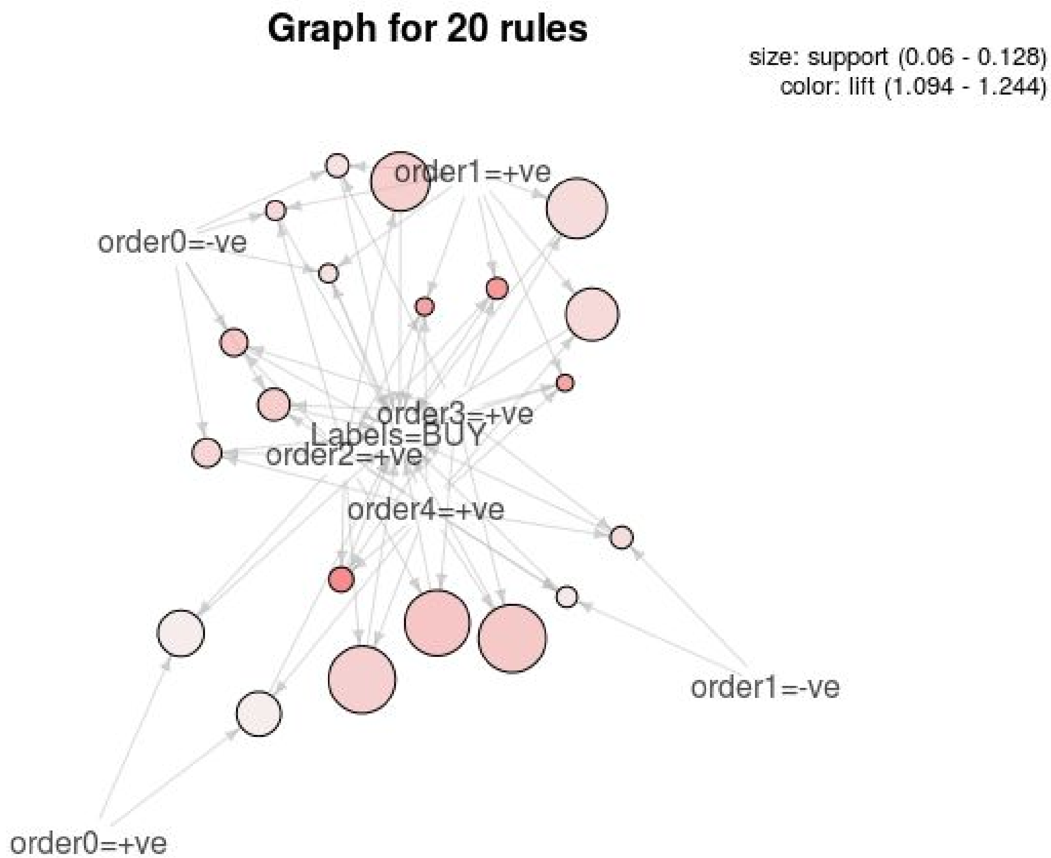
```r
Lab = factor(bolval, levels = c(FALSE, TRUE), labels = c("SELL", "BUY"))
seeds$Labels <- Lab
ind <- sample(2, nrow(seeds), replace = TRUE, prob=c(0.7,0.30))
train.data <- seeds[ind==1,]
test.data <- seeds[ind==2,]

library(arules)
#associate_data = seeds[43:58]
associate_data1 = seeds[c(44,47,50,53,56,58)]
associate_data2 = seeds[c(43,46,49,52,55,58)]
associate_data3 = seeds[c(45,48,51,54,57,58)]

#rules <- apriori(associate_data1, parameter = list(minlen=2, supp=0.005, conf=0.8), appearance = list(rhs=c("
#rules <- apriori(associate_data3, parameter = list(minlen=2, supp=0.07, conf=0.8), appearance =
#list(rhs=c("Labels=BUY", "Labels=SELL"), default="lhs"), control = list(verbose=F))

rules <- apriori(associate_data2, parameter = list(minlen=2, supp=0.07, conf=0.8), appearance = list(rhs=c("La

rules.sorted <- sort(rules, by="lift")
inspect(rules.sorted)
subset.matrix <- is.subset(rules.sorted, rules.sorted)
subset.matrix[lower.tri(subset.matrix, diag=T)] <- NA
redundant <- colSums(subset.matrix, na.rm=T) >= 1
rules.pruned <- rules.sorted[!redundant]

library(grid)
library(arulesViz)
```

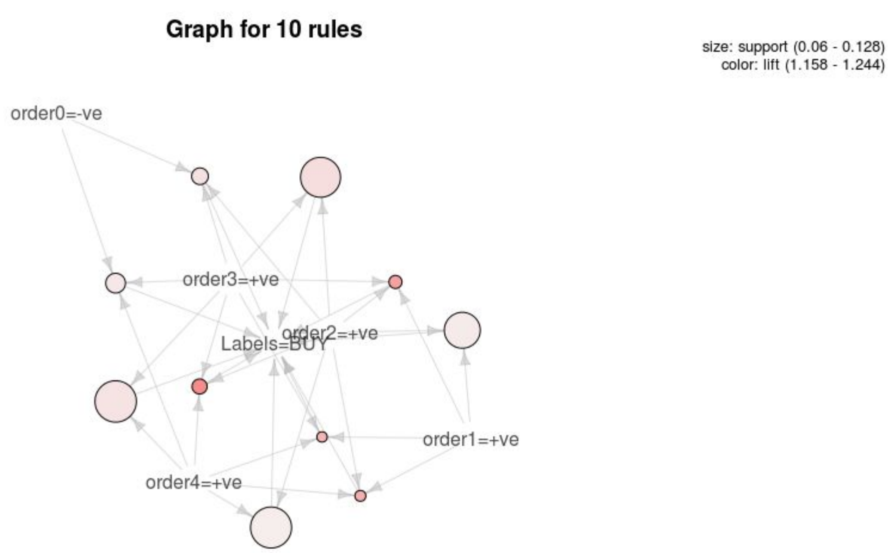
```r

library(grid)
library(arulesViz)
plot(rules)
plot(rules, method="graph", control=list(type="items"))


```

Results for Association Rules:

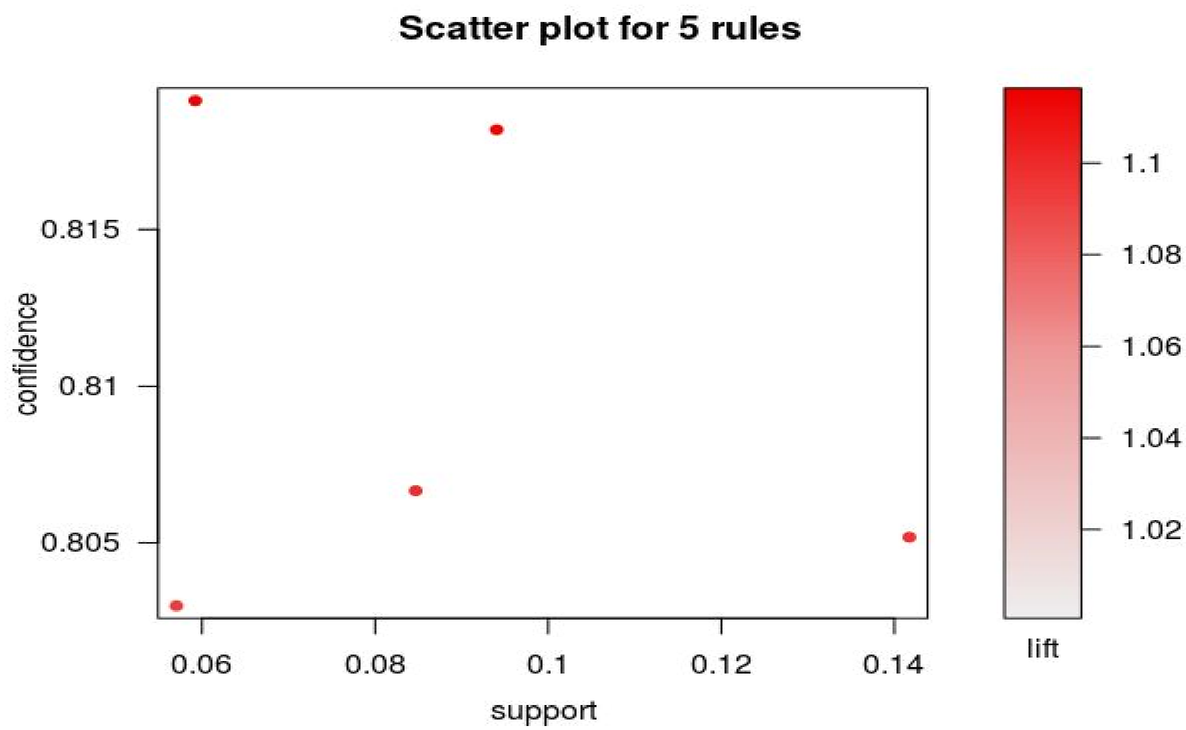
Results for how the order(+ve if BUY_ORDERS > SELL_ORDERS) attributes at different depth levels influences the Labels(Which tells the BUY trend or SELL trend):

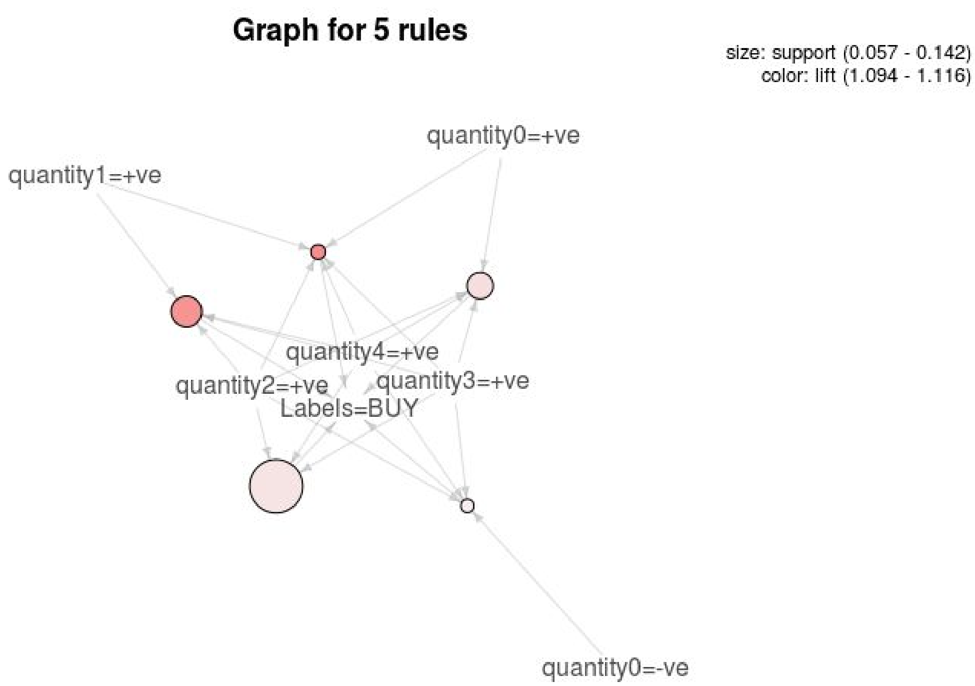

Scatter plot for 20 rules
confidence
0.9
0.88
0.86
0.84
0.82
support
0.06
0.07
0.08
0.09
0.1
0.11
0.12
1.2
1.15
1.1
1.05
lift


Graph for 20 rules
size: support (0.06 - 0.128)
color: lift (1.094 - 1.244)
order1=+ve
order0=-ve
order3=+ve
Labels=BUY
order2=+ve
order4=+ve
order1=-ve
order0=+ve

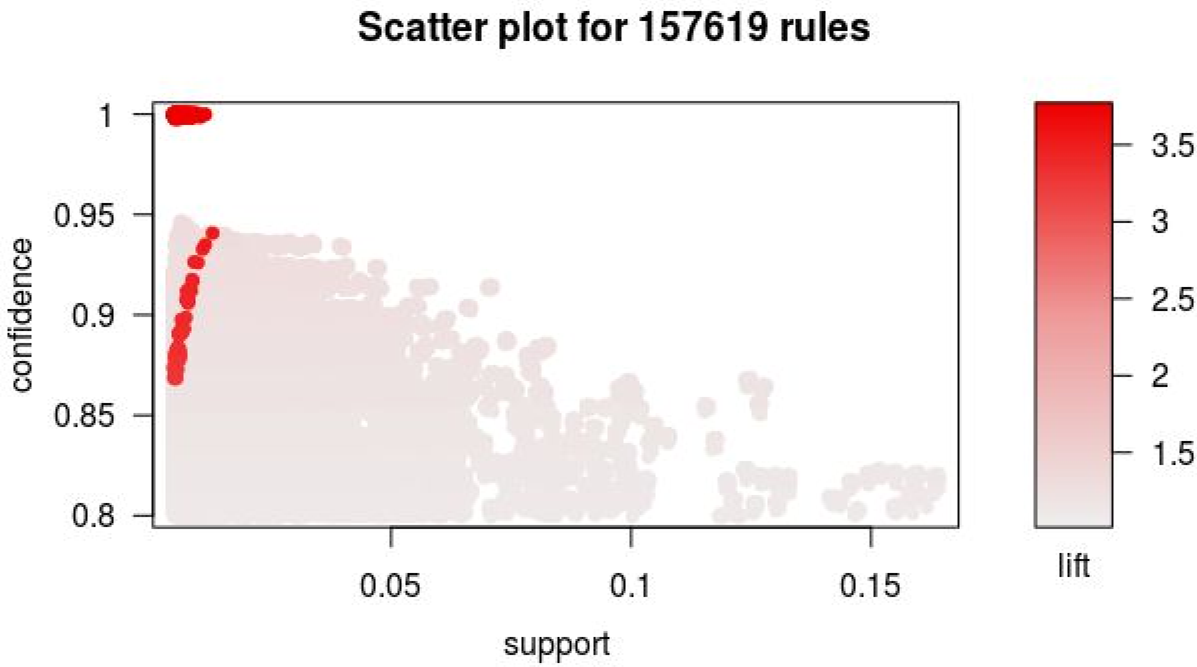

Graph for 10 rules
size: support (0.06 - 0.128)
color: lift (1.158 - 1.244)
order0=-ve
order3=+ve
order2=+ve
Labels=BUY
order1=+ve
order4=+ve

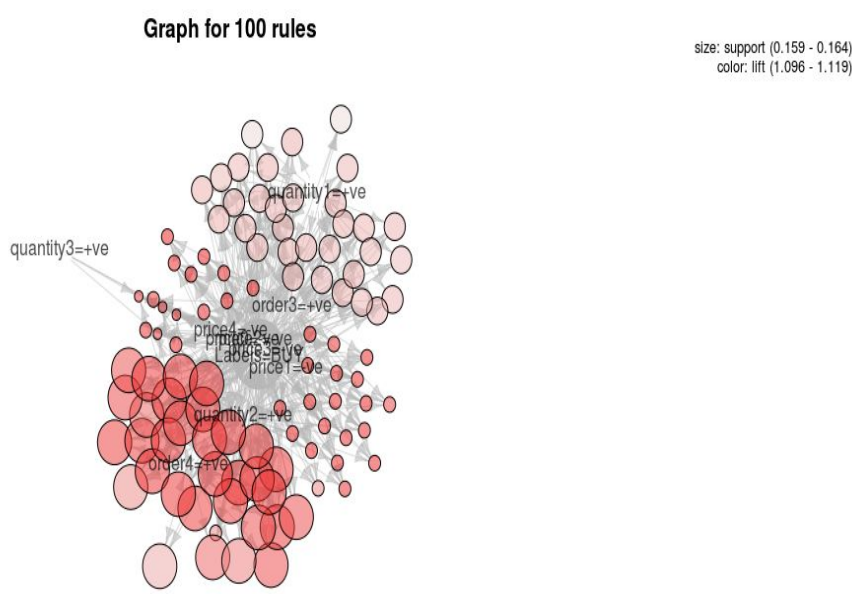
Results for how the quantity(+ve if BUY_QUANTITY > SELL_QUANTITY) attributes at different depth levels influences the Labels(Which tells the BUY trend or SELL trend):





Results for how the order(+ve if BUY_ORDERS > SELL_ORDERS) attributes at different depth levels influences the Labels(Which tells the BUY trend or SELL trend), similarly quantity and price at different depth levels are taken into consideration:





Relationship among stock prices:

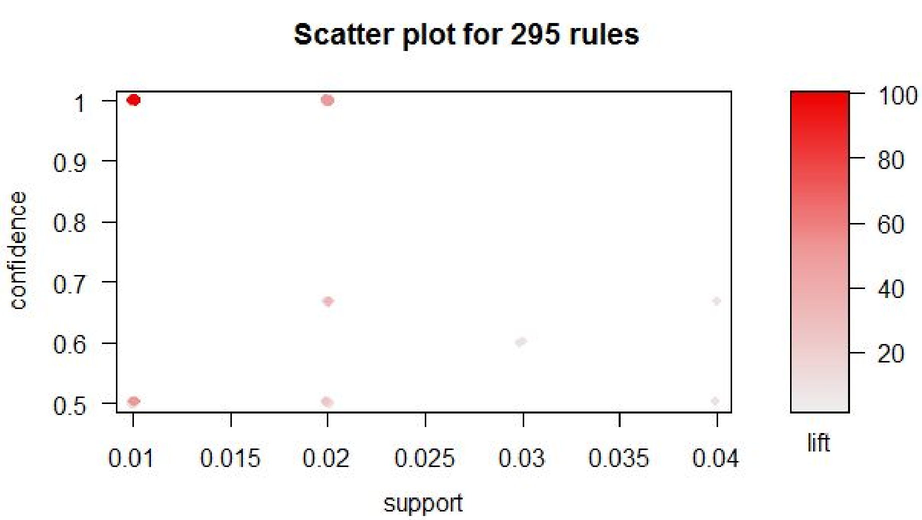
Script:

```
for(i in 1:100){
  vect <- c()
  for(j in 1:1000){
    if(sampleids[i] == tot_new$stamps[j]){
      vect <- c(vect,tot_new$newname[j])
    }
  }
  lst[[i]] <- vect
}

rules <- apriori(lst,parameter=list(supp=0.01, conf = 0.5))
```

Results:

```
lhs           rhs          support confidence lift      count
[1] {dACC}      => {iAUROPHARMA} 0.01    1.0        12.50000 1
[2] {iBPCL}     => {dEICHERMOT}  0.01    1.0        33.33333 1
[3] {iHCLTECH}  => {iHINDALCO}   0.01    1.0       100.00000 1
[4] {iHINDALCO} => {iHCLTECH}    0.01    1.0       100.00000 1
[5] {iHCLTECH}  => {dHDFCBANK}   0.01    1.0        50.00000 1
[6] {dHDFCBANK} => {iHCLTECH}    0.01    0.5        50.00000 1
```

Plots:



Inference:

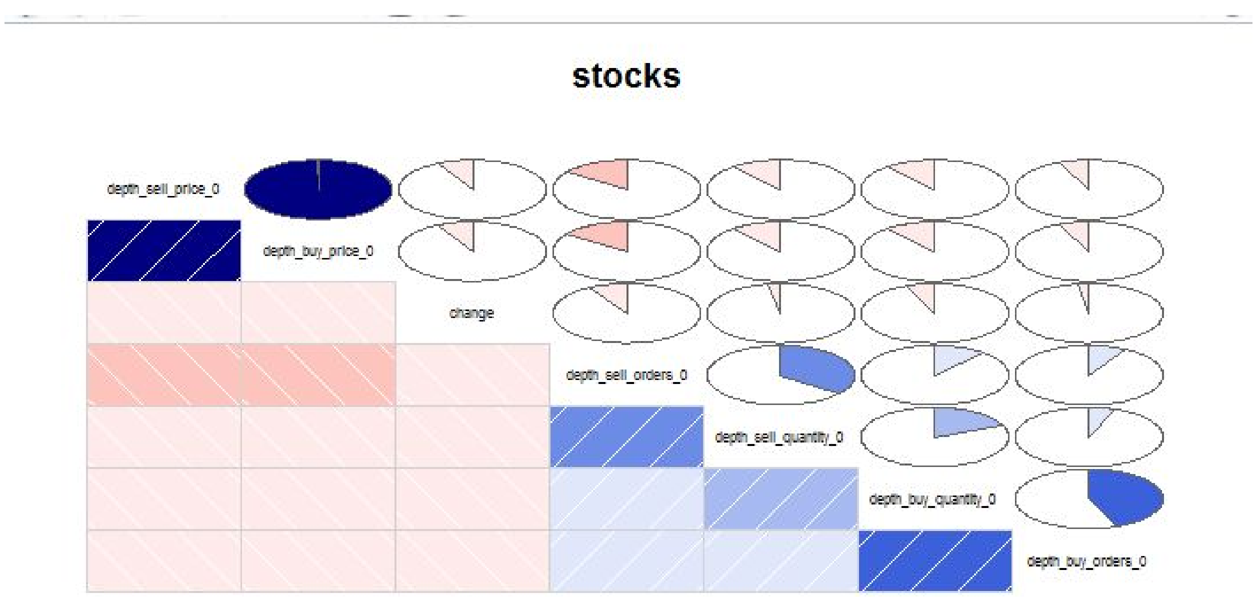
Here, in our lhs and rhs, every company string is prefixed either with “i” or “d”. This means increasing trend or decreasing trend respectively. The predictor used for this is the “change” percentage attribute which is dependant on previous close_price and current close_price. If change is positive, company’s name is prefixed as “i” else it is prefixed as “d”.

After giving the minSupport = 0.01, confidence = 0.5, Six most relevant rules are retrieved.

For example:

1) **One of the valid inference could be the relation between the share prices of BPCL(Petroleum sector) and EICHERMOT(Automobile sector).**

4. Variation of one level depth attributes with each other, across change attribute



Explanation: Blue shade represents positive correlation and red shade represents negative correlation. The percentage of confidence is depicted from the pie charts. This gives a brief overview of how the attributes are related.