CIS 600 Final Project - Digit Recognition

Gafei, Greg Shin, Ashraf Elnashar

Nov. 11th, 2019

## Introduction

The ultimate goal in creating a stock portfolio is to gather a group of stocks that will increase in value overtime. However, it is often difficult to pick which stocks to invest in from all the ones that are available.Therefore, we intend to explore the use of clustering analysis in identifying groups of similarly performingstocks.

## Goal

To use clustering techniques with historical stock prices to group different stocks together, in orderto form an investment strategy.

## Load Libraries

#Function that loads libraries  
EnsurePackage <- function(x) {  
 x <- as.character(x)  
 if (!require(x,character.only = T))  
 install.packages(x,repos = "https://cran.rstudio.com/")  
 require(x,character.only = T)  
}  
  
EnsurePackage("caret") # set of functions that attempt to streamline the process for creating predictive models

## Loading required package: caret

## Loading required package: lattice

## Loading required package: ggplot2

EnsurePackage("rpart") #Recursive Partitioning And Regression Trees

## Loading required package: rpart

EnsurePackage("DMwR") #Smote

## Loading required package: DMwR

## Loading required package: grid

## Registered S3 method overwritten by 'xts':  
## method from  
## as.zoo.xts zoo

## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

EnsurePackage("rattle") #graphical user interface to many other R packages that provide functionality for data mining

## Loading required package: rattle

## Rattle: A free graphical interface for data science with R.  
## Version 5.2.0 Copyright (c) 2006-2018 Togaware Pty Ltd.  
## Type 'rattle()' to shake, rattle, and roll your data.

EnsurePackage("tidyverse") #Manipulating dataset

## Loading required package: tidyverse

## -- Attaching packages ------------------------------------------------------------------------------------------------------------------------------------------------------------------ tidyverse 1.2.1 --

## v tibble 2.1.3 v purrr 0.3.2  
## v tidyr 1.0.0 v dplyr 0.8.3  
## v readr 1.3.1 v stringr 1.4.0  
## v tibble 2.1.3 v forcats 0.4.0

## -- Conflicts --------------------------------------------------------------------------------------------------------------------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()  
## x purrr::lift() masks caret::lift()

EnsurePackage("ggplot2")  
EnsurePackage("readr")  
EnsurePackage("dplyr") #selecting data  
EnsurePackage("magrittr") #using pipe operators

## Loading required package: magrittr

##   
## Attaching package: 'magrittr'

## The following object is masked from 'package:purrr':  
##   
## set\_names

## The following object is masked from 'package:tidyr':  
##   
## extract

EnsurePackage("corrplot")

## Loading required package: corrplot

## corrplot 0.84 loaded

EnsurePackage("knitr")

## Loading required package: knitr

EnsurePackage("sm")

## Loading required package: sm

## Package 'sm', version 2.2-5.6: type help(sm) for summary information

##   
## Attaching package: 'sm'

## The following object is masked from 'package:rattle':  
##   
## binning

EnsurePackage("gmodels")

## Loading required package: gmodels

EnsurePackage("rpart") #Recursive Partitioning and Regression Trees  
EnsurePackage("rpart.plot")

## Loading required package: rpart.plot

EnsurePackage("plotly")

## Loading required package: plotly

##   
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':  
##   
## last\_plot

## The following object is masked from 'package:stats':  
##   
## filter

## The following object is masked from 'package:graphics':  
##   
## layout

EnsurePackage("e1071") #deals with Probability group theory functions

## Loading required package: e1071

EnsurePackage("RColorBrewer") #coloring of graphs

## Loading required package: RColorBrewer

EnsurePackage("plotly")  
EnsurePackage("cluster") # clustering algorithms

## Loading required package: cluster

EnsurePackage("dendextend") # for comparing two dendrograms

## Loading required package: dendextend

##   
## ---------------------  
## Welcome to dendextend version 1.13.2  
## Type citation('dendextend') for how to cite the package.  
##   
## Type browseVignettes(package = 'dendextend') for the package vignette.  
## The github page is: https://github.com/talgalili/dendextend/  
##   
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues  
## Or contact: <tal.galili@gmail.com>  
##   
## To suppress this message use: suppressPackageStartupMessages(library(dendextend))  
## ---------------------

##   
## Attaching package: 'dendextend'

## The following object is masked from 'package:rpart':  
##   
## prune

## The following object is masked from 'package:stats':  
##   
## cutree

EnsurePackage("stats")  
EnsurePackage("pacman")

## Loading required package: pacman

EnsurePackage("factoextra")

## Loading required package: factoextra

## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa

EnsurePackage("reshape2")

## Loading required package: reshape2

##   
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':  
##   
## smiths

EnsurePackage("tidyr")  
EnsurePackage("textshape")

## Loading required package: textshape

##   
## Attaching package: 'textshape'

## The following object is masked from 'package:dplyr':  
##   
## combine

## The following object is masked from 'package:purrr':  
##   
## flatten

## The following object is masked from 'package:tibble':  
##   
## column\_to\_rownames

## Load Data

nRowsRead = 1000 # specify 'None' if want to read whole file  
 # dataset\_summary.csv has 7091 rows in reality, but we are only loading/previewing the first 1000 rows  
  
 path <- 'dataset\_summary.csv'  
 dataSetReader\_Summary <-read.csv(path, nrows = nRowsRead)  
 #Summary of Stock Prices.  
 str(dataSetReader\_Summary)

## 'data.frame': 1000 obs. of 7 variables:  
## $ symbol : Factor w/ 1000 levels "A","AA","AAAP",..: 1 2 3 4 5 6 7 8 9 10 ...  
## $ total\_prices : int 4962 697 574 5434 1222 3489 1675 5434 5436 1476 ...  
## $ stock\_from\_date : Factor w/ 593 levels "1998-01-02","1998-01-20",..: 37 458 416 1 365 145 299 1 1 322 ...  
## $ stock\_to\_date : Factor w/ 71 levels "2018-01-30","2018-02-12",..: 69 69 16 67 69 67 69 67 69 67 ...  
## $ total\_earnings : int 42 11 0 14 21 23 23 39 41 24 ...  
## $ earnings\_from\_date: Factor w/ 351 levels "2009-04-16","2009-04-23",..: 13 281 351 166 216 166 147 13 52 157 ...  
## $ earnings\_to\_date : Factor w/ 149 levels "2015-05-11","2015-07-16",..: 143 123 149 7 83 129 35 39 129 138 ...

#Head of Summary of Stock.  
 head(dataSetReader\_Summary, 5)

## symbol total\_prices stock\_from\_date stock\_to\_date total\_earnings  
## 1 A 4962 1999-11-18 2019-08-09 42  
## 2 AA 697 2016-11-01 2019-08-09 11  
## 3 AAAP 574 2015-11-11 2018-07-18 0  
## 4 AABA 5434 1998-01-02 2019-08-07 14  
## 5 AAC 1222 2014-10-02 2019-08-09 21  
## earnings\_from\_date earnings\_to\_date  
## 1 2009-05-14 2019-08-14  
## 2 2017-01-24 2019-07-17  
## 3 NULL NULL  
## 4 2014-01-28 2017-04-18  
## 5 2014-11-05 2019-04-16

#Divid ends of Stock Prices.  
 path <- 'dividends\_latest.csv'  
 dataSetReader\_Dividends <-read.csv(path, nrows = nRowsRead)  
 #Dividends of Stock Prices.  
 str(dataSetReader\_Dividends)

## 'data.frame': 1000 obs. of 3 variables:  
## $ symbol : Factor w/ 33 levels "AAL","AAME","AAON",..: 33 33 33 33 33 33 33 33 33 33 ...  
## $ date : Factor w/ 704 levels "2000-01-10","2000-02-28",..: 633 376 264 368 401 385 178 516 542 282 ...  
## $ dividend: num 0.39 0.16 0.11 0.16 0.2 0.16 0.09 0.31 0.31 0.13 ...

#Head of Divid ends of Stock.  
 head(dataSetReader\_Dividends, 5)

## symbol date dividend  
## 1 MSFT 2016-11-15 0.39  
## 2 MSFT 2011-05-17 0.16  
## 3 MSFT 2008-05-13 0.11  
## 4 MSFT 2011-02-15 0.16  
## 5 MSFT 2012-02-14 0.20

#Earnings of Stock Prices.  
 path <- 'earnings\_latest.csv'  
 dataSetReader\_Earnings <-read.csv(path, nrows = nRowsRead)  
 #Dividends of Stock Prices.  
 str(dataSetReader\_Earnings)

## 'data.frame': 1000 obs. of 6 variables:  
## $ symbol : Factor w/ 32 levels "A","AA","AABA",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ date : Factor w/ 667 levels "2009-05-05","2009-05-06",..: 5 17 31 40 59 72 84 93 111 128 ...  
## $ qtr : Factor w/ 88 levels "01/2010","01/2011",..: 22 44 65 1 23 45 66 2 24 46 ...  
## $ eps\_est : Factor w/ 253 levels "-0.0200","-0.0300",..: 253 253 253 253 253 253 253 253 253 253 ...  
## $ eps : Factor w/ 261 levels "-0.0100","-0.0200",..: 261 261 261 261 261 261 261 261 261 261 ...  
## $ release\_time: Factor w/ 3 levels "NULL","post",..: 2 2 3 3 2 2 3 2 1 2 ...

#Head of Earnings of Stock.  
 head(dataSetReader\_Earnings, 5)

## symbol date qtr eps\_est eps release\_time  
## 1 A 2009-05-14 04/2009 NULL NULL post  
## 2 A 2009-08-17 07/2009 NULL NULL post  
## 3 A 2009-11-13 10/2009 NULL NULL pre  
## 4 A 2010-02-12 01/2010 NULL NULL pre  
## 5 A 2010-05-17 04/2010 NULL NULL post

#Stock Prices.  
 path <- 'stock\_prices\_latest\_Simplified.csv'  
 dataSetReader\_Prices <-read.csv(path, nrows = nRowsRead)  
 #Stock Prices.  
 str(dataSetReader\_Prices)

## 'data.frame': 1000 obs. of 9 variables:  
## $ symbol : Factor w/ 924 levels "AABA","AAME",..: 855 789 630 174 544 84 577 871 311 873 ...  
## $ date : Factor w/ 912 levels "1/10/2012","1/11/2005",..: 697 425 651 91 153 35 154 298 448 253 ...  
## $ open : num 0.51 6.37 9.6 19 27.12 ...  
## $ high : num 0.51 6.37 9.95 19 27.41 ...  
## $ low : num 0.51 6.37 9.52 18.73 26.99 ...  
## $ close : num 0.51 6.37 9.9 18.74 27.35 ...  
## $ close\_adjusted : num 0.17 5.16 11404.8 8.89 27.15 ...  
## $ volume : num 0 0 147735 5400 1028741 ...  
## $ split\_coefficient: int 1 1 1 1 1 1 1 1 1 1 ...

#Head of Stock Prices.  
 head(dataSetReader\_Prices, 5)

## symbol date open high low close close\_adjusted volume  
## 1 TXMD 7/10/2009 0.51 0.510 0.51 0.51 0.1700 0  
## 2 SPA 3/3/1999 6.37 6.370 6.37 6.37 5.1619 0  
## 3 NURO 6/25/2007 9.60 9.950 9.52 9.90 11404.8000 147735  
## 4 CEA 10/14/2004 19.00 19.000 18.73 18.74 8.8906 5400  
## 5 MDU 10/31/2017 27.12 27.405 26.99 27.35 27.1524 1028741  
## split\_coefficient  
## 1 1  
## 2 1  
## 3 1  
## 4 1  
## 5 1

## Data preprocessing

### stock dataset summary

#find NA across all  
 missing = dataSetReader\_Summary[, sapply(dataSetReader\_Summary, anyNA), drop = FALSE]  
  
 cat("Missing data found in ",ncol(missing),"Columns, which is",  
 ncol(missing)/ncol(dataSetReader\_Summary)\*100, "% of features")

## Missing data found in 0 Columns, which is 0 % of features

#The missing columns and how many missing value it has  
 missingData <- sapply(dataSetReader\_Summary,function(x) {sum(is.na(x))})  
 Position(function(x) x > 0, missingData)

## [1] NA

MissingNames <- names(dataSetReader\_Summary[, sapply(dataSetReader\_Summary, anyNA), drop = FALSE])  
  
 for (i in MissingNames){  
 dataSetReader\_Summary[is.na(dataSetReader\_Summary[,i]),i] <- median(dataSetReader\_Summary[,i],na.rm = T)}  
  
#Find number of missing values/check ranges  
sum(is.na(dataSetReader\_Summary))

## [1] 0

# Check Duplicate Data Record   
 nrow(dataSetReader\_Summary)

## [1] 1000

nrow(dataSetReader\_Summary[!duplicated(dataSetReader\_Summary),])

## [1] 1000

### stock dataset Divid ends

#find NA across all  
 missing = dataSetReader\_Dividends[, sapply(dataSetReader\_Dividends, anyNA), drop = FALSE]  
  
 cat("Missing data found in ",ncol(missing),"Columns, which is",  
 ncol(missing)/ncol(dataSetReader\_Dividends)\*100, "% of features")

## Missing data found in 0 Columns, which is 0 % of features

#The missing columns and how many missing value it has  
 missingData <- sapply(dataSetReader\_Dividends,function(x) {sum(is.na(x))})  
 Position(function(x) x > 0, missingData)

## [1] NA

MissingNames <- names(dataSetReader\_Dividends[, sapply(dataSetReader\_Dividends, anyNA), drop = FALSE])  
  
 for (i in MissingNames){  
 dataSetReader\_Dividends[is.na(dataSetReader\_Dividends[,i]),i] <- median(dataSetReader\_Dividends[,i],na.rm = T)}  
  
#Find number of missing values/check ranges  
sum(is.na(dataSetReader\_Dividends))

## [1] 0

# Check Duplicate Data Record  
 nrow(dataSetReader\_Dividends)

## [1] 1000

nrow(dataSetReader\_Dividends[!duplicated(dataSetReader\_Dividends),])

## [1] 1000

### stock dataset Earnings

#find NA across all  
 missing = dataSetReader\_Earnings[, sapply(dataSetReader\_Earnings, anyNA), drop = FALSE]  
  
 cat("Missing data found in ",ncol(missing),"Columns, which is",  
 ncol(missing)/ncol(dataSetReader\_Earnings)\*100, "% of features")

## Missing data found in 0 Columns, which is 0 % of features

#The missing columns and how many missing value it has  
 missingData <- sapply(dataSetReader\_Earnings,function(x) {sum(is.na(x))})  
 Position(function(x) x > 0, missingData)

## [1] NA

MissingNames <- names(dataSetReader\_Earnings[, sapply(dataSetReader\_Earnings, anyNA), drop = FALSE])  
  
 for (i in MissingNames){  
 dataSetReader\_Earnings[is.na(dataSetReader\_Earnings[,i]),i] <- median(dataSetReader\_Earnings[,i],na.rm = T)}  
  
#Find number of missing values/check ranges  
sum(is.na(dataSetReader\_Earnings))

## [1] 0

# Check Duplicate Data Record  
 nrow(dataSetReader\_Earnings)

## [1] 1000

nrow(dataSetReader\_Earnings[!duplicated(dataSetReader\_Earnings),])

## [1] 1000

### stock dataset Stock Prices.

#find NA across all  
 missing = dataSetReader\_Prices[, sapply(dataSetReader\_Prices, anyNA), drop = FALSE]  
  
 cat("Missing data found in ",ncol(missing),"Columns, which is",  
 ncol(missing)/ncol(dataSetReader\_Prices)\*100, "% of features")

## Missing data found in 0 Columns, which is 0 % of features

#The missing columns and how many missing value it has  
 missingData <- sapply(dataSetReader\_Prices,function(x) {sum(is.na(x))})  
 Position(function(x) x > 0, missingData)

## [1] NA

MissingNames <- names(dataSetReader\_Prices[, sapply(dataSetReader\_Prices, anyNA), drop = FALSE])  
  
 for (i in MissingNames){  
 dataSetReader\_Prices[is.na(dataSetReader\_Prices[,i]),i] <- median(dataSetReader\_Prices[,i],na.rm = T)}  
  
#Find number of missing values/check ranges  
sum(is.na(dataSetReader\_Prices))

## [1] 0

# Check Duplicate Data Record  
 nrow(dataSetReader\_Prices)

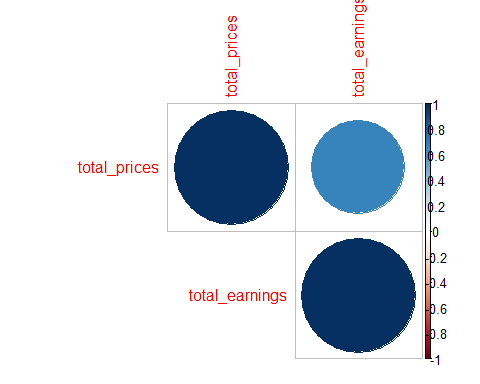
## [1] 1000

nrow(dataSetReader\_Prices[!duplicated(dataSetReader\_Prices),])

## [1] 1000

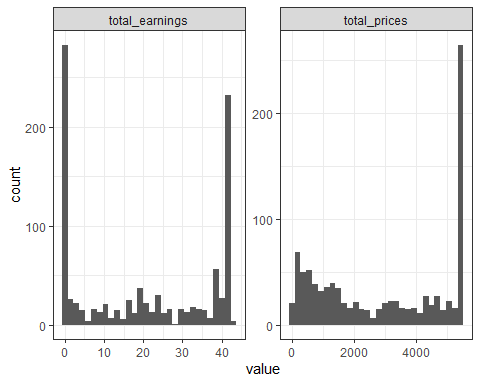
## Stock Summary Data Exploration

# Removing outliers   
dataSetReader\_Summary$total\_earnings[dataSetReader\_Summary$total\_earnings %in% boxplot.stats(dataSetReader\_Summary$total\_earnings)$out] <- median(dataSetReader\_Summary$total\_earnings, na.rm = T)  
  
# Removing outliers   
dataSetReader\_Summary$total\_prices[dataSetReader\_Summary$total\_prices %in% boxplot.stats(dataSetReader\_Summary$total\_prices)$out] <- median(dataSetReader\_Summary$total\_prices, na.rm = T)  
  
  
#Correlation between total\_prices and total\_earnings variables  
cor\_matrix <- cor(dataSetReader\_Summary[complete.cases(dataSetReader\_Summary), sapply(dataSetReader\_Summary, is.numeric)], method = "pearson")  
corrplot(cor\_matrix, type = "upper")



# a graphical way of representing the relationship between total\_prices and total\_earnings field.  
theme\_set(theme\_bw())  
  
# ggplot(dataSetReader\_Summary, aes(x = total\_earnings, y = total\_prices, group = 2)) +  
# geom\_boxplot() +  
# theme(panel.grid.major.x = element\_blank())  
  
dataSetReader\_Summary %>%  
 keep(is.numeric) %>%   
 gather() %>%   
 ggplot(aes(value)) +  
 facet\_wrap(~ key, scales = "free") +  
 geom\_histogram()

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



d <- dataSetReader\_Summary  
d$vs <- factor(d$total\_earnings)  
d$am <- factor(d$total\_prices)  
  
d %>% str()

## 'data.frame': 1000 obs. of 9 variables:  
## $ symbol : Factor w/ 1000 levels "A","AA","AAAP",..: 1 2 3 4 5 6 7 8 9 10 ...  
## $ total\_prices : num 4962 697 574 5434 1222 ...  
## $ stock\_from\_date : Factor w/ 593 levels "1998-01-02","1998-01-20",..: 37 458 416 1 365 145 299 1 1 322 ...  
## $ stock\_to\_date : Factor w/ 71 levels "2018-01-30","2018-02-12",..: 69 69 16 67 69 67 69 67 69 67 ...  
## $ total\_earnings : num 42 11 0 14 21 23 23 39 41 24 ...  
## $ earnings\_from\_date: Factor w/ 351 levels "2009-04-16","2009-04-23",..: 13 281 351 166 216 166 147 13 52 157 ...  
## $ earnings\_to\_date : Factor w/ 149 levels "2015-05-11","2015-07-16",..: 143 123 149 7 83 129 35 39 129 138 ...  
## $ vs : Factor w/ 44 levels "0","1","2","3",..: 43 12 1 15 22 24 24 40 42 25 ...  
## $ am : Factor w/ 672 levels "16","32","33",..: 619 165 144 670 258 490 332 670 672 307 ...

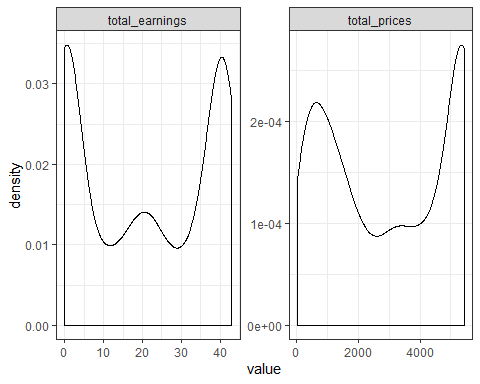
library(purrr)  
d %>% keep(is.numeric) %>% head()

## total\_prices total\_earnings  
## 1 4962 42  
## 2 697 11  
## 3 574 0  
## 4 5434 14  
## 5 1222 21  
## 6 3489 23

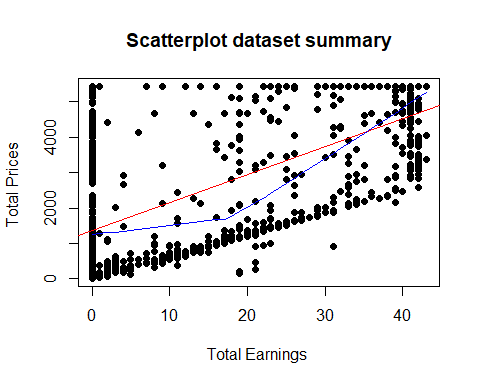
library(tidyr)  
d %>%  
 keep(is.numeric) %>%   
 gather() %>%  
 head()

## key value  
## 1 total\_prices 4962  
## 2 total\_prices 697  
## 3 total\_prices 574  
## 4 total\_prices 5434  
## 5 total\_prices 1222  
## 6 total\_prices 3489

library(ggplot2)  
d %>%  
 keep(is.numeric) %>% # Keep only numeric columns  
 gather() %>% # Convert to key-value pairs  
 ggplot(aes(value)) + # Plot the values  
 facet\_wrap(~ key, scales = "free") + # In separate panels  
 geom\_density() # as density



plot(dataSetReader\_Summary$total\_earnings, dataSetReader\_Summary$total\_prices, main="Scatterplot dataset summary",  
 xlab="Total Earnings ", ylab="Total Prices ", pch=19)  
  
  
# Add fit lines  
abline(lm(dataSetReader\_Summary$total\_prices~dataSetReader\_Summary$total\_earnings), col="red") # regression line (y~x)  
lines(lowess(dataSetReader\_Summary$total\_earnings,dataSetReader\_Summary$total\_prices), col="blue") # lowess line (x,y)

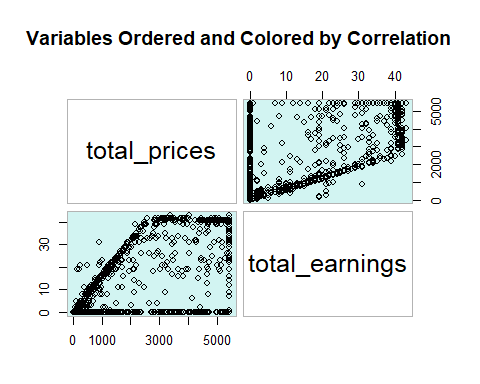


# Scatterplot Matrices from the glus Package  
library(gclus)

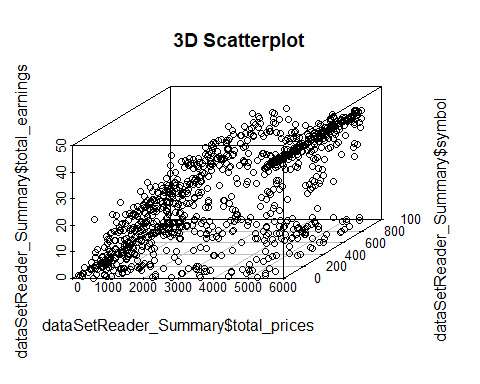
##   
## Attaching package: 'gclus'

## The following object is masked from 'package:dendextend':  
##   
## order.hclust

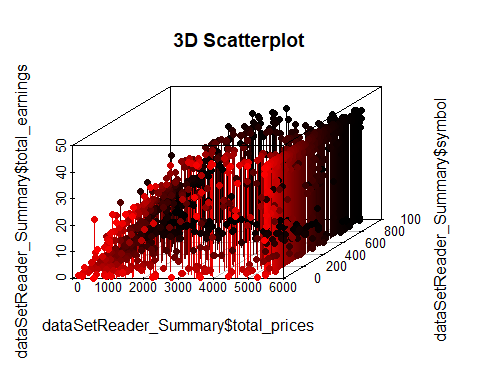
dta <- dataSetReader\_Summary[c(2,5)] # get data  
dta.r <- abs(cor(dta)) # get correlations  
dta.col <- dmat.color(dta.r) # get colors  
# reorder variables so those with highest correlation  
# are closest to the diagonal  
dta.o <- order.single(dta.r)  
cpairs(dta, dta.o, panel.colors=dta.col, gap=.5,  
main="Variables Ordered and Colored by Correlation" )



# 3D Scatterplot  
library(scatterplot3d)  
scatterplot3d(dataSetReader\_Summary$total\_prices,dataSetReader\_Summary$symbol,dataSetReader\_Summary$total\_earnings, main="3D Scatterplot")

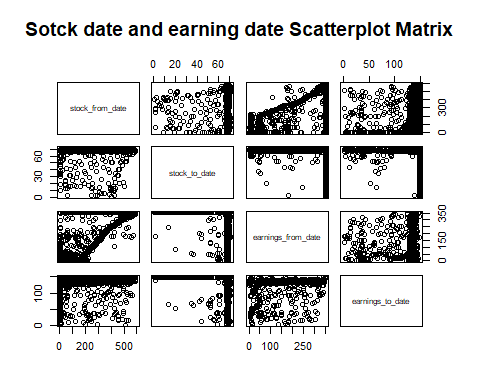


# 3D Scatterplot with Coloring and Vertical Drop Lines  
library(scatterplot3d)  
scatterplot3d(dataSetReader\_Summary$total\_prices,dataSetReader\_Summary$symbol,dataSetReader\_Summary$total\_earnings, pch=16, highlight.3d=TRUE,  
 type="h", main="3D Scatterplot")



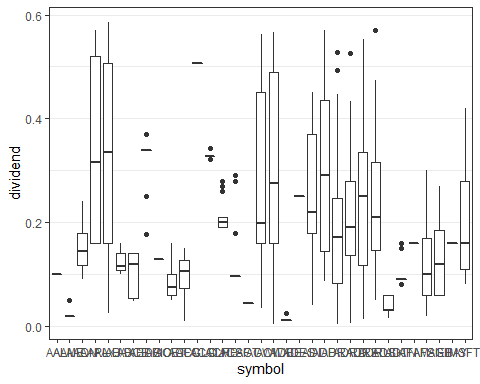
# Spinning 3d Scatterplot  
library(rgl)  
  
plot3d(dataSetReader\_Summary$total\_prices,dataSetReader\_Summary$symbol,dataSetReader\_Summary$total\_earnings, col="red", size=3)

# Basic Scatterplot Matrix  
pairs(~stock\_from\_date+stock\_to\_date+earnings\_from\_date+earnings\_to\_date,data=dataSetReader\_Summary,  
 main="Sotck date and earning date Scatterplot Matrix")



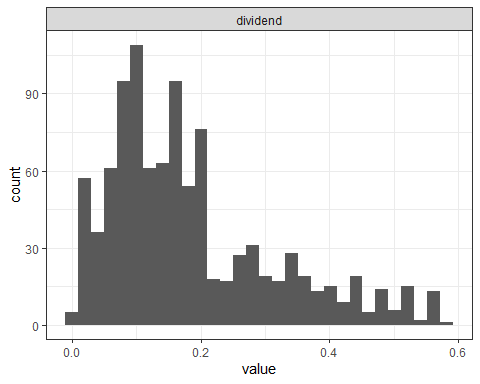
## Stock Dividends Data Exploration

# Removing outliers   
dataSetReader\_Dividends$dividend[dataSetReader\_Dividends$dividend %in% boxplot.stats(dataSetReader\_Dividends$dividend)$out] <- median(dataSetReader\_Dividends$dividend, na.rm = T)  
  
ggplot(dataSetReader\_Dividends, aes(x = symbol, y = dividend)) +  
 geom\_boxplot() +  
 theme(panel.grid.major.x = element\_blank())



dataSetReader\_Dividends %>%  
 keep(is.numeric) %>%   
 gather() %>%   
 ggplot(aes(value)) +  
 facet\_wrap(~ key, scales = "free") +  
 geom\_histogram()

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



d <- dataSetReader\_Dividends  
d$vs <- factor(d$symbol)  
d$am <- factor(d$dividend)  
  
d %>% str()

## 'data.frame': 1000 obs. of 5 variables:  
## $ symbol : Factor w/ 33 levels "AAL","AAME","AAON",..: 33 33 33 33 33 33 33 33 33 33 ...  
## $ date : Factor w/ 704 levels "2000-01-10","2000-02-28",..: 633 376 264 368 401 385 178 516 542 282 ...  
## $ dividend: num 0.39 0.16 0.11 0.16 0.2 0.16 0.09 0.31 0.31 0.13 ...  
## $ vs : Factor w/ 33 levels "AAL","AAME","AAON",..: 33 33 33 33 33 33 33 33 33 33 ...  
## $ am : Factor w/ 331 levels "0.0041","0.0043",..: 269 121 73 121 158 121 62 233 233 92 ...

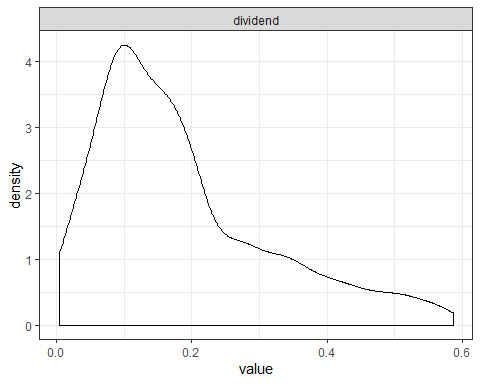
library(purrr)  
d %>% keep(is.numeric) %>% head()

## dividend  
## 1 0.39  
## 2 0.16  
## 3 0.11  
## 4 0.16  
## 5 0.20  
## 6 0.16

library(tidyr)  
d %>%  
 keep(is.numeric) %>%   
 gather() %>%  
 head()

## key value  
## 1 dividend 0.39  
## 2 dividend 0.16  
## 3 dividend 0.11  
## 4 dividend 0.16  
## 5 dividend 0.20  
## 6 dividend 0.16

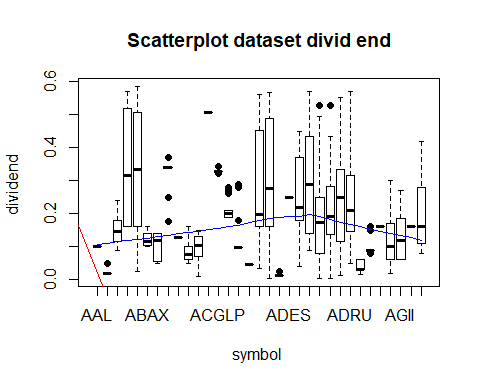
library(ggplot2)  
d %>%  
 keep(is.numeric) %>% # Keep only numeric columns  
 gather() %>% # Convert to key-value pairs  
 ggplot(aes(value)) + # Plot the values  
 facet\_wrap(~ key, scales = "free") + # In separate panels  
 geom\_density() # as density



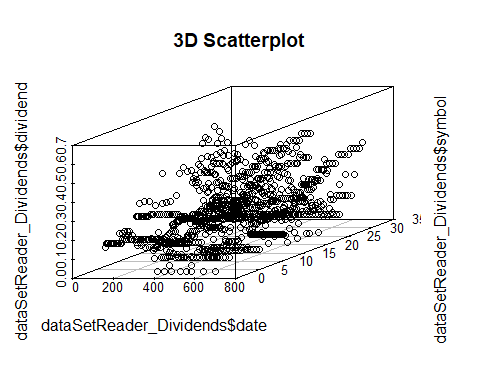
plot(dataSetReader\_Dividends$symbol, dataSetReader\_Dividends$dividend, main="Scatterplot dataset divid end",  
 xlab="symbol ", ylab="dividend ", pch=19)  
  
# Add fit lines  
abline(lm(dataSetReader\_Dividends$dividend~dataSetReader\_Dividends$symbol), col="red") # regression line (y~x)

## Warning in abline(lm(dataSetReader\_Dividends$dividend ~  
## dataSetReader\_Dividends$symbol), : only using the first two of 33  
## regression coefficients

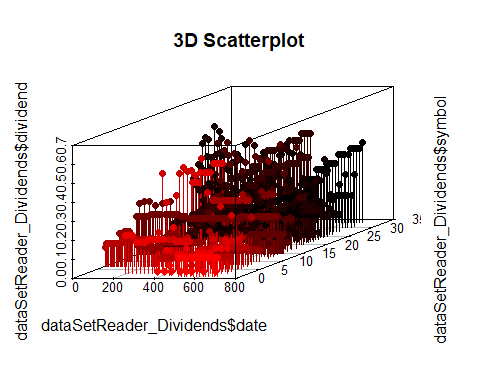
lines(lowess(dataSetReader\_Dividends$symbol,dataSetReader\_Dividends$dividend), col="blue") # lowess line (x,y)



# 3D Scatterplot  
library(scatterplot3d)  
scatterplot3d(dataSetReader\_Dividends$date,dataSetReader\_Dividends$symbol,dataSetReader\_Dividends$dividend, main="3D Scatterplot")



# 3D Scatterplot with Coloring and Vertical Drop Lines  
library(scatterplot3d)  
scatterplot3d(dataSetReader\_Dividends$date,dataSetReader\_Dividends$symbol,dataSetReader\_Dividends$dividend, pch=16, highlight.3d=TRUE,  
 type="h", main="3D Scatterplot")

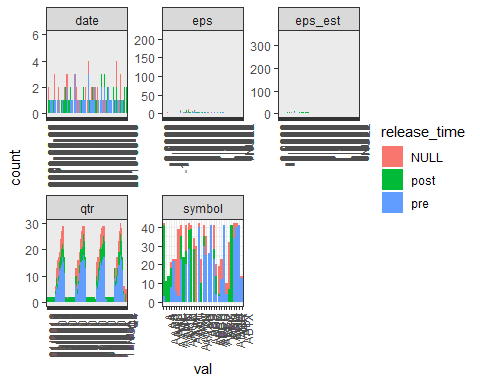


# Spinning 3d Scatterplot  
library(rgl)  
  
plot3d(dataSetReader\_Dividends$date,dataSetReader\_Dividends$symbol,dataSetReader\_Dividends$dividend, col="red", size=3)

## Stock Earnings Data Exploration

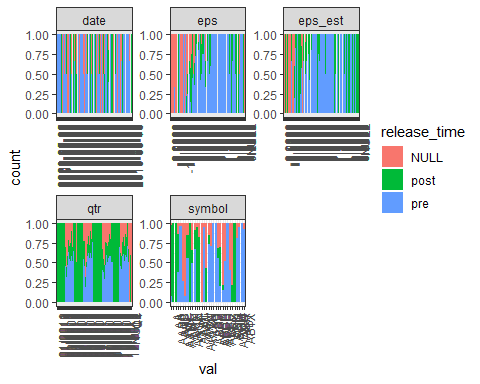
dataSetReader\_EarningsFactor <- dataSetReader\_Earnings %>% select\_if(is.factor)  
  
# Exploration of all factor variables  
# absolute bar chart  
dataSetReader\_EarningsFactor %>%gather("key","val",setdiff(names(.), "release\_time")) %>%   
 ggplot(aes(val,fill=release\_time)) +  
 facet\_wrap(~ key, scales = "free") +  
 geom\_bar(stat = 'count',position = "stack") + theme(axis.text.x = element\_text(angle = 90, hjust = 1))

## Warning: attributes are not identical across measure variables;  
## they will be dropped

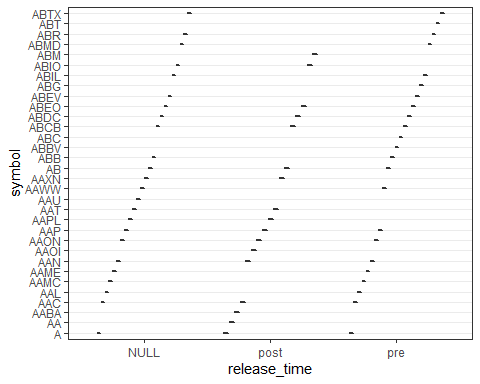


# Exploration of all factor variables  
# Relative bar chart  
dataSetReader\_EarningsFactor %>%gather("key","val",setdiff(names(.), "release\_time")) %>%   
 ggplot(aes(val,fill=release\_time)) +  
 facet\_wrap(~ key, scales = "free") +  
 geom\_bar(stat = 'count',position = "fill") + theme(axis.text.x = element\_text(angle = 90, hjust = 1))

## Warning: attributes are not identical across measure variables;  
## they will be dropped

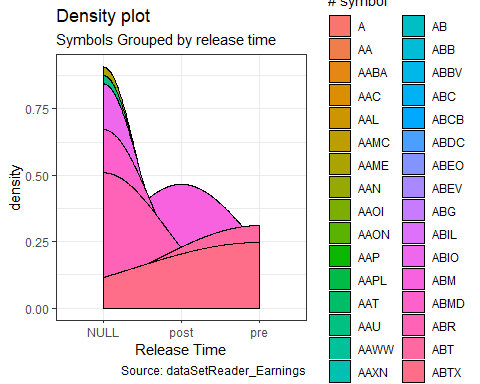


# a graphical way of representing the Min, 1st Qu, Median, Mean 3rd Qu, and Max relationship between daily rate and educational field.  
theme\_set(theme\_bw())  
  
ggplot(dataSetReader\_Earnings, aes(x = release\_time, y = symbol)) +  
 geom\_boxplot() +  
 theme(panel.grid.major.x = element\_blank())



# The density plot is a basic tool in the data science toolkit.  
# density plots are usually a much more effective way to view the distribution of a variable. Create the plot using plot(density(x)) where x is a numeric vector.  
  
ggplot(dataSetReader\_Earnings, aes(release\_time)) +  
 geom\_density(aes(fill=factor(symbol))) +  
 labs(title="Density plot",  
 subtitle="Symbols Grouped by release time",  
 caption="Source: dataSetReader\_Earnings",  
 x="Release Time",  
 fill="# symbol")

## Warning: Groups with fewer than two data points have been dropped.  
  
## Warning: Groups with fewer than two data points have been dropped.  
  
## Warning: Groups with fewer than two data points have been dropped.  
  
## Warning: Groups with fewer than two data points have been dropped.



# Categorical variable(release time) vs Categorical variable(symbol)  
  
  
# compare two categorical variable education field and attrition.  
# as we see in the graph the technical people and marketing are the most people that they leave the company.  
# this is an important attribute for prediction based on the p-value result (p=0.008471793 < 0.05).  
xtabs(~symbol+release\_time,dataSetReader\_Earnings)

## release\_time  
## symbol NULL post pre  
## A 1 38 3  
## AA 0 11 0  
## AABA 0 14 0  
## AAC 3 10 8  
## AAL 1 0 22  
## AAMC 17 0 6  
## AAME 36 0 3  
## AAN 6 12 23  
## AAOI 0 24 0  
## AAON 14 7 20  
## AAP 4 10 28  
## AAPL 2 39 0  
## AAT 6 28 0  
## AAU 30 0 0  
## AAWW 2 0 40  
## AAXN 13 10 0  
## AB 6 5 30  
## ABB 10 0 25  
## ABBV 0 0 26  
## ABC 0 0 41  
## ABCB 13 5 23  
## ABDC 6 10 4  
## ABEO 15 1 3  
## ABEV 11 0 12  
## ABG 0 0 41  
## ABIL 6 0 4  
## ABIO 25 7 0  
## ABM 0 41 0  
## ABMD 8 0 34  
## ABR 2 0 40  
## ABT 0 0 41  
## ABTX 1 0 13

# convert eps and eps\_est to numeric   
  
dataSetReader\_Earnings$eps <- as.numeric(as.character(dataSetReader\_Earnings$eps))

## Warning: NAs introduced by coercion

dataSetReader\_Earnings$eps\_est <- as.numeric(as.character(dataSetReader\_Earnings$eps\_est))

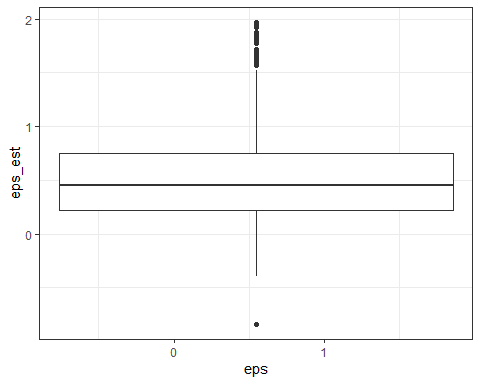
## Warning: NAs introduced by coercion

# Removing outliers   
dataSetReader\_Earnings$eps[dataSetReader\_Earnings$eps %in% boxplot.stats(dataSetReader\_Earnings$eps)$out] <- median(dataSetReader\_Earnings$eps, na.rm = T)  
  
dataSetReader\_Earnings$eps\_est[dataSetReader\_Earnings$eps\_est %in% boxplot.stats(dataSetReader\_Earnings$eps\_est)$out] <- median(dataSetReader\_Earnings$eps\_est, na.rm = T)

# a graphical way of representing the relationship between eps and eps\_est field.  
theme\_set(theme\_bw())  
  
ggplot(dataSetReader\_Earnings, aes(x = eps, y = eps\_est, group = 2)) +  
 geom\_boxplot() +  
 theme(panel.grid.major.x = element\_blank())

## Warning: Removed 211 rows containing missing values (stat\_boxplot).

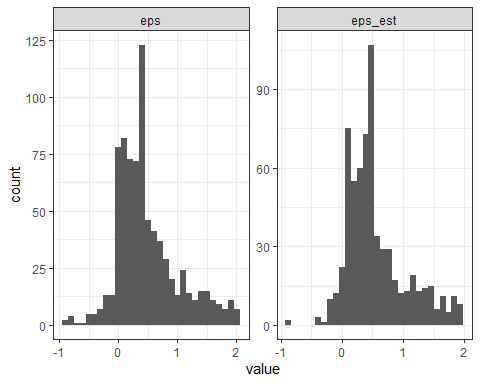
## Warning: Removed 135 rows containing non-finite values (stat\_boxplot).



dataSetReader\_Earnings %>%  
 keep(is.numeric) %>%   
 gather() %>%   
 ggplot(aes(value)) +  
 facet\_wrap(~ key, scales = "free") +  
 geom\_histogram()

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 555 rows containing non-finite values (stat\_bin).



d <- dataSetReader\_Earnings  
d$vs <- factor(d$eps)  
d$am <- factor(d$eps\_est)  
  
d %>% str()

## 'data.frame': 1000 obs. of 8 variables:  
## $ symbol : Factor w/ 32 levels "A","AA","AABA",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ date : Factor w/ 667 levels "2009-05-05","2009-05-06",..: 5 17 31 40 59 72 84 93 111 128 ...  
## $ qtr : Factor w/ 88 levels "01/2010","01/2011",..: 22 44 65 1 23 45 66 2 24 46 ...  
## $ eps\_est : num NA NA NA NA NA NA NA NA NA NA ...  
## $ eps : num NA NA NA NA NA NA NA NA NA NA ...  
## $ release\_time: Factor w/ 3 levels "NULL","post",..: 2 2 3 3 2 2 3 2 1 2 ...  
## $ vs : Factor w/ 209 levels "-0.91","-0.85",..: NA NA NA NA NA NA NA NA NA NA ...  
## $ am : Factor w/ 215 levels "-0.84","-0.398",..: NA NA NA NA NA NA NA NA NA NA ...

library(purrr)  
d %>% keep(is.numeric) %>% head()

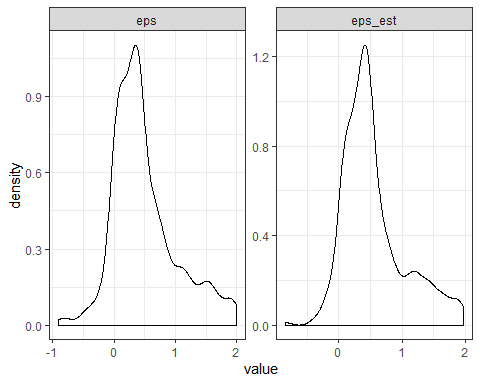
## eps\_est eps  
## 1 NA NA  
## 2 NA NA  
## 3 NA NA  
## 4 NA NA  
## 5 NA NA  
## 6 NA NA

library(tidyr)  
d %>%  
 keep(is.numeric) %>%   
 gather() %>%  
 head()

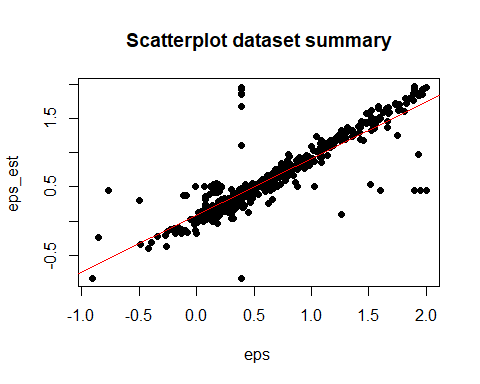
## key value  
## 1 eps\_est NA  
## 2 eps\_est NA  
## 3 eps\_est NA  
## 4 eps\_est NA  
## 5 eps\_est NA  
## 6 eps\_est NA

library(ggplot2)  
d %>%  
 keep(is.numeric) %>% # Keep only numeric columns  
 gather() %>% # Convert to key-value pairs  
 ggplot(aes(value)) + # Plot the values  
 facet\_wrap(~ key, scales = "free") + # In separate panels  
 geom\_density() # as density

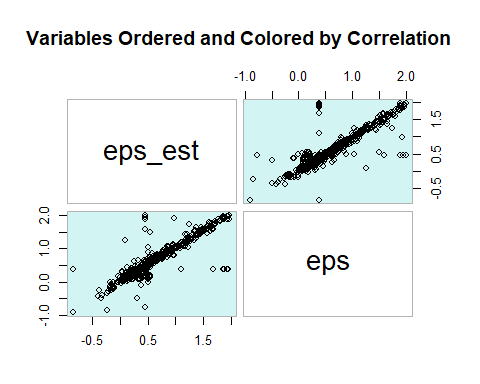
## Warning: Removed 555 rows containing non-finite values (stat\_density).



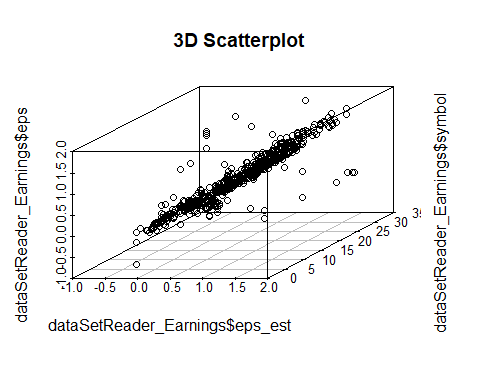
plot(dataSetReader\_Earnings$eps, dataSetReader\_Earnings$eps\_est, main="Scatterplot dataset summary",  
 xlab="eps ", ylab="eps\_est ", pch=19)  
  
  
# Add fit lines  
abline(lm(dataSetReader\_Earnings$eps\_est~dataSetReader\_Earnings$eps), col="red") # regression line (y~x)



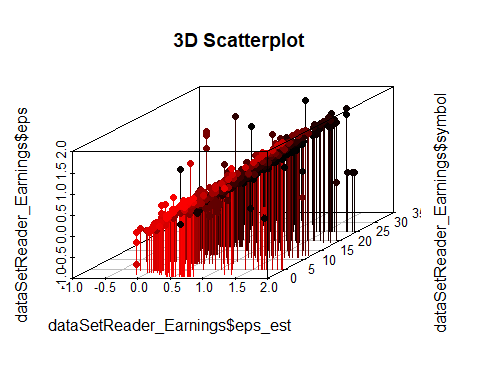
# Scatterplot Matrices from the glus Package  
library(gclus)  
dta <- dataSetReader\_Earnings[c(4,5)] # get data  
dta.r <- abs(cor(dta)) # get correlations  
dta.col <- dmat.color(dta.r) # get colors  
# reorder variables so those with highest correlation  
# are closest to the diagonal  
dta.o <- order.single(dta.r)  
cpairs(dta, dta.o, panel.colors=dta.col, gap=.5,  
main="Variables Ordered and Colored by Correlation" )



# 3D Scatterplot  
library(scatterplot3d)  
scatterplot3d(dataSetReader\_Earnings$eps\_est,dataSetReader\_Earnings$symbol,dataSetReader\_Earnings$eps, main="3D Scatterplot")

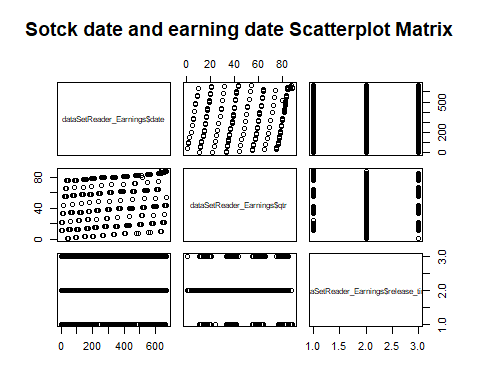


# 3D Scatterplot with Coloring and Vertical Drop Lines  
library(scatterplot3d)  
scatterplot3d(dataSetReader\_Earnings$eps\_est,dataSetReader\_Earnings$symbol,dataSetReader\_Earnings$eps, pch=16, highlight.3d=TRUE,  
 type="h", main="3D Scatterplot")



# Spinning 3d Scatterplot  
library(rgl)  
  
plot3d(dataSetReader\_Earnings$eps\_est,dataSetReader\_Earnings$symbol,dataSetReader\_Earnings$eps, col="red", size=3)

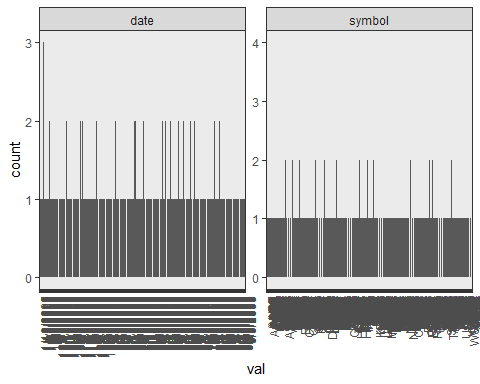
# Basic Scatterplot Matrix  
pairs(~dataSetReader\_Earnings$date+dataSetReader\_Earnings$qtr+dataSetReader\_Earnings$release\_time,data=dataSetReader\_Earnings,  
 main="Sotck date and earning date Scatterplot Matrix")



## Stock Prices Data Exploration

dataSetReader\_PricesFactor <- dataSetReader\_Prices %>% select\_if(is.factor)  
  
# Exploration of all factor variables  
# absolute bar chart  
dataSetReader\_PricesFactor %>%gather("key","val",setdiff(names(.), "release\_time")) %>%   
 ggplot(aes(val,fill=dataSetReader\_PricesFactor$close\_adjusted)) +  
 facet\_wrap(~ key, scales = "free") +  
 geom\_bar(stat = 'count',position = "stack") + theme(axis.text.x = element\_text(angle = 90, hjust = 1))

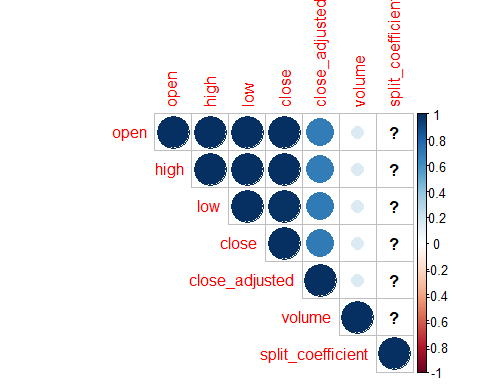
## Warning: attributes are not identical across measure variables;  
## they will be dropped



# Removing outliers   
dataSetReader\_Prices$high[dataSetReader\_Prices$high %in% boxplot.stats(dataSetReader\_Prices$high)$out] <- median(dataSetReader\_Prices$high, na.rm = T)  
dataSetReader\_Prices$low[dataSetReader\_Prices$low %in% boxplot.stats(dataSetReader\_Prices$low)$out] <- median(dataSetReader\_Prices$low, na.rm = T)  
dataSetReader\_Prices$close[dataSetReader\_Prices$close %in% boxplot.stats(dataSetReader\_Prices$close)$out] <- median(dataSetReader\_Prices$close, na.rm = T)  
dataSetReader\_Prices$open[dataSetReader\_Prices$open %in% boxplot.stats(dataSetReader\_Prices$open)$out] <- median(dataSetReader\_Prices$open, na.rm = T)  
dataSetReader\_Prices$close\_adjusted[dataSetReader\_Prices$close\_adjusted %in% boxplot.stats(dataSetReader\_Prices$close\_adjusted)$out] <- median(dataSetReader\_Prices$close\_adjusted, na.rm = T)  
dataSetReader\_Prices$split\_coefficient[dataSetReader\_Prices$split\_coefficient %in% boxplot.stats(dataSetReader\_Prices$split\_coefficient)$out] <- median(dataSetReader\_Prices$split\_coefficient, na.rm = T)  
dataSetReader\_Prices$volume[dataSetReader\_Prices$volume %in% boxplot.stats(dataSetReader\_Prices$volume)$out] <- median(dataSetReader\_Prices$volume, na.rm = T)  
  
  
#Correlation between total\_prices and total\_earnings variables  
cor\_matrix <- cor(dataSetReader\_Prices[complete.cases(dataSetReader\_Prices), sapply(dataSetReader\_Prices, is.numeric)], method = "pearson")

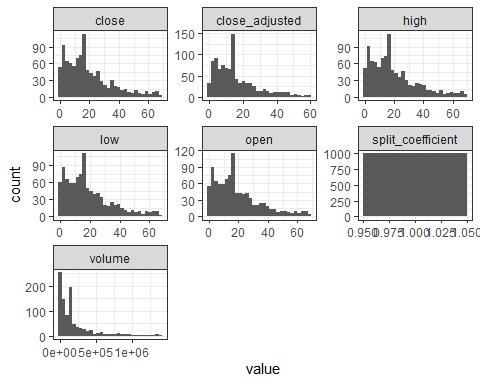
## Warning in cor(dataSetReader\_Prices[complete.cases(dataSetReader\_Prices), :  
## the standard deviation is zero

corrplot(cor\_matrix, type = "upper")



# a graphical way of representing the relationship between total\_prices and total\_earnings field.  
theme\_set(theme\_bw())  
  
dataSetReader\_Prices %>%  
 keep(is.numeric) %>%   
 gather() %>%   
 ggplot(aes(value)) +  
 facet\_wrap(~ key, scales = "free") +  
 geom\_histogram()

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



d <- dataSetReader\_Prices  
d$vs <- factor(d$close)  
d$am <- factor(d$open)  
  
d %>% str()

## 'data.frame': 1000 obs. of 11 variables:  
## $ symbol : Factor w/ 924 levels "AABA","AAME",..: 855 789 630 174 544 84 577 871 311 873 ...  
## $ date : Factor w/ 912 levels "1/10/2012","1/11/2005",..: 697 425 651 91 153 35 154 298 448 253 ...  
## $ open : num 0.51 6.37 9.6 19 27.12 ...  
## $ high : num 0.51 6.37 9.95 19 27.41 ...  
## $ low : num 0.51 6.37 9.52 18.73 26.99 ...  
## $ close : num 0.51 6.37 9.9 18.74 27.35 ...  
## $ close\_adjusted : num 0.17 5.16 13.91 8.89 27.15 ...  
## $ volume : num 0 0 147735 5400 1028741 ...  
## $ split\_coefficient: num 1 1 1 1 1 1 1 1 1 1 ...  
## $ vs : Factor w/ 851 levels "0.007","0.011",..: 24 190 270 483 622 245 442 532 55 834 ...  
## $ am : Factor w/ 838 levels "0.0068","0.011",..: 24 192 266 479 603 246 432 525 58 821 ...

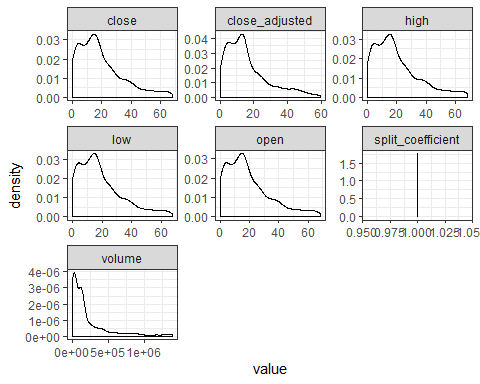
library(purrr)  
d %>% keep(is.numeric) %>% head()

## open high low close close\_adjusted volume split\_coefficient  
## 1 0.51 0.510 0.51 0.51 0.17000 0 1  
## 2 6.37 6.370 6.37 6.37 5.16190 0 1  
## 3 9.60 9.950 9.52 9.90 13.90595 147735 1  
## 4 19.00 19.000 18.73 18.74 8.89060 5400 1  
## 5 27.12 27.405 26.99 27.35 27.15240 1028741 1  
## 6 8.80 8.800 8.65 8.80 8.23970 1672 1

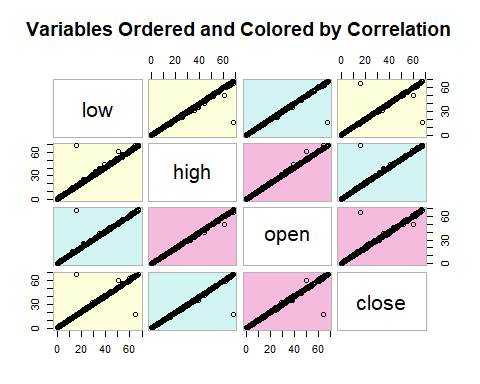
library(tidyr)  
d %>%  
 keep(is.numeric) %>%   
 gather() %>%  
 head()

## key value  
## 1 open 0.51  
## 2 open 6.37  
## 3 open 9.60  
## 4 open 19.00  
## 5 open 27.12  
## 6 open 8.80

library(ggplot2)  
d %>%  
 keep(is.numeric) %>% # Keep only numeric columns  
 gather() %>% # Convert to key-value pairs  
 ggplot(aes(value)) + # Plot the values  
 facet\_wrap(~ key, scales = "free") + # In separate panels  
 geom\_density() # as density



# Scatterplot Matrices from the glus Package  
library(gclus)  
dta <- dataSetReader\_Prices[c(3,6,4,5)] # get data  
dta.r <- abs(cor(dta)) # get correlations  
dta.col <- dmat.color(dta.r) # get colors  
# reorder variables so those with highest correlation  
# are closest to the diagonal  
dta.o <- order.single(dta.r)  
cpairs(dta, dta.o, panel.colors=dta.col, gap=.5,  
main="Variables Ordered and Colored by Correlation" )



## Attempt our own cleaning up of data here.

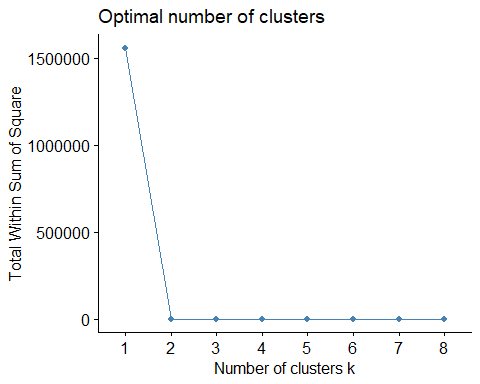
# lets look at the data briefly  
str(stockTradingDataKmeans)

## 'data.frame': 21170358 obs. of 9 variables:  
## $ symbol : Factor w/ 7091 levels "A","AA","AAAP",..: 4292 4292 4292 4292 4292 4292 4292 4292 4292 4292 ...  
## $ date : Factor w/ 5440 levels "1998-01-02","1998-01-05",..: 4622 1015 932 2470 4148 3485 3506 592 4868 3471 ...  
## $ open : num 50.8 68.8 53.4 36 41.6 ...  
## $ high : num 52 69.8 55 36 42.3 ...  
## $ low : num 50.8 67.8 53.2 34.6 41.5 ...  
## $ close : num 51.8 67.9 54.3 35 42.2 ...  
## $ close\_adjusted : num 49.7 22.6 18.1 27.2 38.7 ...  
## $ volume : num 2.00e+07 3.10e+07 4.16e+07 2.88e+08 7.46e+07 ...  
## $ split\_coefficient: num 1 1 1 1 1 1 1 1 1 1 ...

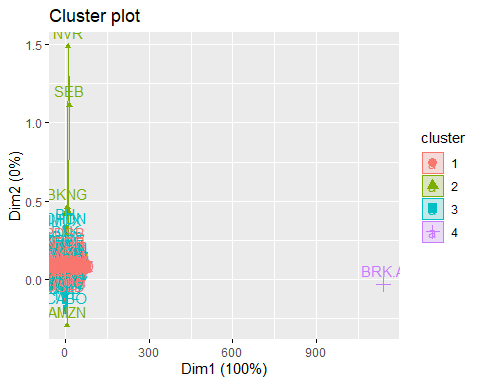
# remove the split coefficient column  
stockTradingDataKmeans <-stockTradingDataKmeans[, -c(9)]

# change the date column to be in "date" format  
stockTradingDataKmeans$date <- as.Date(stockTradingDataKmeans$date)

# lets figure out the optimal amount of clusters for testna1 (symbols)  
fviz\_nbclust(testna1, kmeans, method = "wss", k.max = 8)



# shows an aggregate of stock movement, based on stock symbol for center of 4  
fviz\_cluster(kmeansStocks2018\_s4, data = testna1)



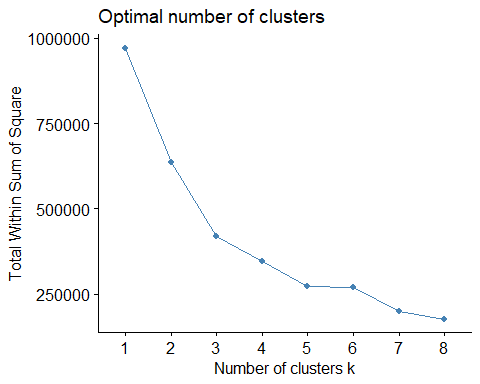
# results say, remove brk.a, bkng, nvr, seb since it skews data

testna1\_mod<-testna1[-c(687, 614, 3381, 4148), ]

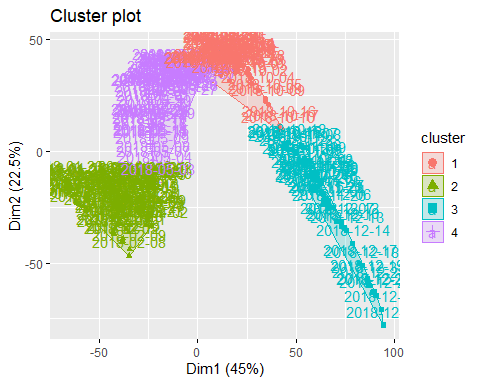
# rerun with testna1 modified

testna1\_mod<-testna1[-c(687, 614, 3381, 4148), ] fviz\_nbclust(testna1\_mod, kmeans, method = “wss”, k.max = 8) kmeansStocks2018\_s4 <-kmeans(testna1\_mod, centers = 4, iter.max = 10000) fviz\_cluster(kmeansStocks2018\_s4, data = testna1\_mod)

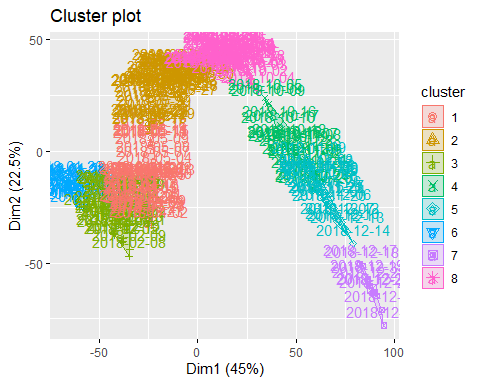
# lets figure out the optimal amount of clusters for testna2 (date)  
fviz\_nbclust(testna2, kmeans, method = "wss", k.max = 8)



# shows an aggregate of stock movement, based on trading day for center of 4  
fviz\_cluster(kmeansStocks2018\_4, data = testna2)

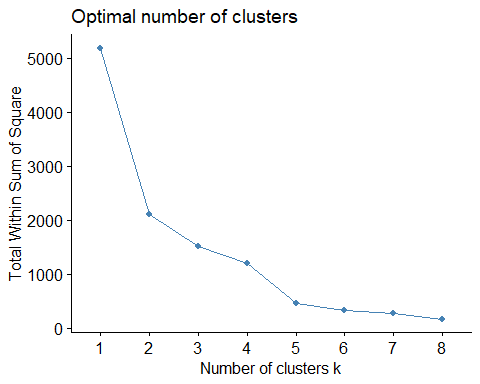


# shows an aggregate of stock movement, based on trading day for center of 8  
fviz\_cluster(kmeansStocks2018\_8, data = testna2)



## Lets calcuate annual returns here? How should we select stocks for annual returns?

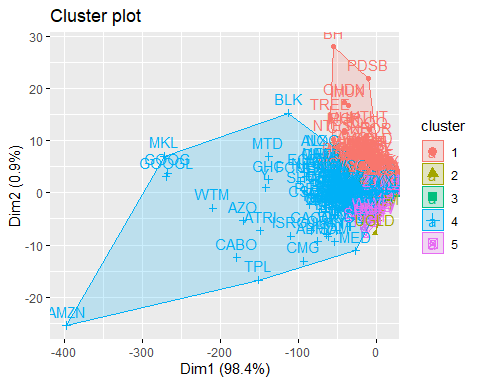
# lets figure out the optimal amount of clusters for annual returns  
fviz\_nbclust(annual\_returns\_scaled, kmeans, method = "wss", k.max = 8)



# graph says 5 clusters should be fine.

#k means analysis by annaul returns, but omit anything that is na, with center of 5  
kmeansannualreturns <-kmeans(annual\_returns\_scaled, centers = 5, iter.max = 10000)

# shows an aggregate of stock based on annual returns,   
fviz\_cluster(kmeansannualreturns, data = reshapeStockssymbolrowNONA)



## Calculate returns based on earnings data? Generally, there’s higher volatility during “earnings” season. Q1, Q2, Q3, Q4?

## Data Preprocessing

To prepare the data for cluster analysis, the Earnings data was first restricted to all earnings within the year 2018. From there, the difference between the reported Earnings per Share and the Estimated Earnings per Share was calculated. Outliers were removed, and the data was then scaled and centered.

#import data  
initial <- read\_csv("../earnings\_latest.csv")

## Parsed with column specification:  
## cols(  
## symbol = col\_character(),  
## date = col\_date(format = ""),  
## qtr = col\_character(),  
## eps\_est = col\_character(),  
## eps = col\_character(),  
## release\_time = col\_character()  
## )

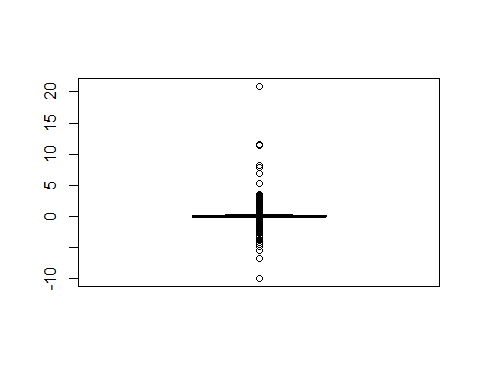
head(initial)

## # A tibble: 6 x 6  
## symbol date qtr eps\_est eps release\_time  
## <chr> <date> <chr> <chr> <chr> <chr>   
## 1 A 2009-05-14 04/2009 NULL NULL post   
## 2 A 2009-08-17 07/2009 NULL NULL post   
## 3 A 2009-11-13 10/2009 NULL NULL pre   
## 4 A 2010-02-12 01/2010 NULL NULL pre   
## 5 A 2010-05-17 04/2010 NULL NULL post   
## 6 A 2010-08-16 07/2010 NULL NULL post

#restrict data to 2018  
stocks2018 <- initial[initial$date >= "2018-01-01" & initial$date <= "2018-12-31",]  
  
#replace NULL values with NA  
stocks2018$eps\_est <- gsub("NULL", NA, stocks2018$eps\_est)  
  
#drop unnecessary columns  
stocks2018$release\_time <- NULL  
stocks2018$qtr <- NULL  
stocks2018$date <- NULL  
  
#drop all incomplete cases  
stocks2018 <- stocks2018[complete.cases(stocks2018),]  
  
#cast numeric data as.numeric  
stocks2018$eps <- as.numeric(stocks2018$eps)  
stocks2018$eps\_est <- as.numeric(stocks2018$eps\_est)  
  
#create column to represent difference between estimate and actual earnings per share  
stocks2018$diff <- stocks2018$eps - stocks2018$eps\_est  
  
str(stocks2018)

## Classes 'tbl\_df', 'tbl' and 'data.frame': 13048 obs. of 4 variables:  
## $ symbol : chr "A" "A" "A" "A" ...  
## $ eps\_est: num 0.58 0.65 0.63 0.73 1.23 0.6 1.33 0.25 0.04 0.1 ...  
## $ eps : num 0.66 0.65 0.67 0.81 1.04 0.77 1.52 0.63 0.1 0.13 ...  
## $ diff : num 0.08 0 0.04 0.08 -0.19 0.17 0.19 0.38 0.06 0.03 ...

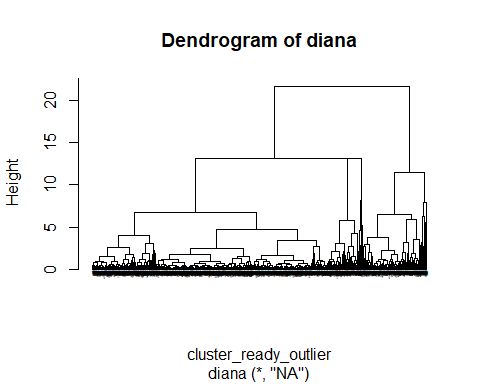
#remove outliers  
mean\_outlier <- boxplot(stocks2018$diff)$out



stocks2018\_mean\_noOutlier <- stocks2018[-which(stocks2018$diff %in% mean\_outlier),]  
  
#get mean of all data by symbol  
stocks2018\_mean\_noOutlier <- stocks2018\_mean\_noOutlier %>% group\_by(symbol) %>% summarise\_all(mean)  
  
#change row names to be stock symbols  
stocks2018\_mean\_noOutlier <- column\_to\_rownames(stocks2018\_mean\_noOutlier, loc=1)  
  
#scale and center all data  
cluster\_ready\_outlier <- scale(stocks2018\_mean\_noOutlier)

## Hierarchical Clustering: Divisive Method

divMeanOutlier <- diana(cluster\_ready\_outlier)  
  
#display dendrogram of DIANA algorithm  
pltree(divMeanOutlier, cex = 0.1, hang = -1, main = "Dendrogram of diana")



## Hierarchical Clustering: Agglomerative Method

To determine the ideal agglomerative clustering method, models were created using all four different types of distance algorithms: complete, single, average, and Ward’s. Ward’s distance was found to have the largest agglomerative coefficient (0.999), which indicated a strong tendency towards clustering.

#run agglomerative clustering with four different measures of distance: complete, single, average, ward  
agglComplete <- agnes(cluster\_ready\_outlier, method="complete")  
agglSingle <- agnes(cluster\_ready\_outlier, method="single")  
agglAverage <- agnes(cluster\_ready\_outlier, method="average")  
agglWard <- agnes(cluster\_ready\_outlier, method="ward")  
  
#display agglomerative coefficient  
agglComplete$ac

## [1] 0.9970553

agglSingle$ac

## [1] 0.9858992

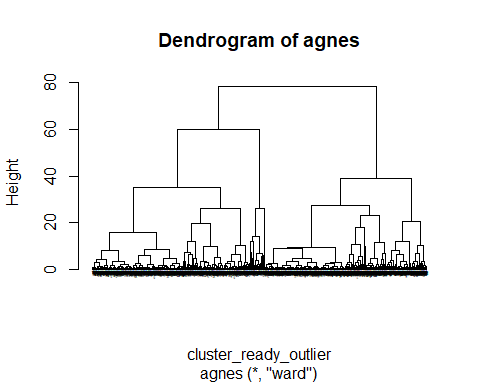
agglAverage$ac

## [1] 0.9939901

agglWard$ac

## [1] 0.9991861

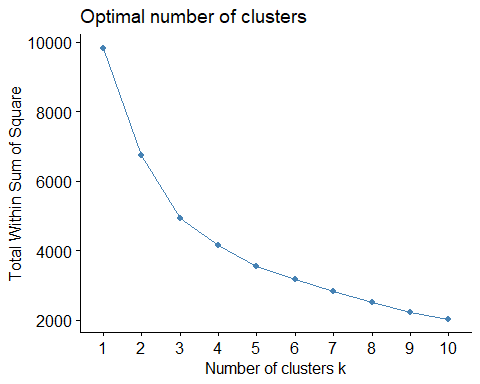
#display dendrogram of clustering using Ward distance  
pltree(agglWard, cex = 0.1, hang = -1, main = "Dendrogram of agnes")



## Optimal Clustering

An elbow diagram was created to determine that the optimal number of clusters was three.

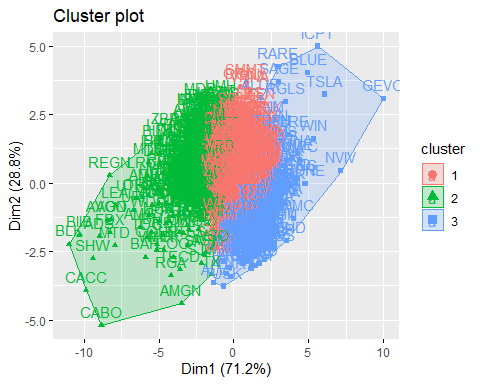
#create elbow diagram  
fviz\_nbclust(cluster\_ready\_outlier, FUN = hcut, method = "wss")



#create k=3 clusters  
sub\_grp\_diana <- cutree(as.hclust(divMeanOutlier), k=3)  
sub\_grp\_agnes <- cutree(as.hclust(agglWard), k=3)

## Final Clustering

#divisive  
fviz\_cluster(list(data = cluster\_ready\_outlier, cluster = sub\_grp\_diana))



#agglomerative  
fviz\_cluster(list(data = cluster\_ready\_outlier, cluster = sub\_grp\_agnes))

