CS544Final\_Aggarwal.R

Adetya Aggarwal

2020-06-25

library(quantmod)

## Loading required package: xts

## Loading required package: zoo

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

## Loading required package: TTR

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

## Version 0.4-0 included new data defaults. See ?getSymbols.

library(gridExtra)  
library(Hmisc)

## Loading required package: lattice

## Loading required package: survival

## Loading required package: Formula

## Loading required package: ggplot2

##   
## Attaching package: 'Hmisc'

## The following object is masked from 'package:quantmod':  
##   
## Lag

## The following objects are masked from 'package:base':  
##   
## format.pval, units

library(plotly)

##   
## Attaching package: 'plotly'

## The following object is masked from 'package:Hmisc':  
##   
## subplot

## 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

library(sampling)

##   
## Attaching package: 'sampling'

## The following objects are masked from 'package:survival':  
##   
## cluster, strata

library(ggplotlyExtra)  
library(ggplot2)  
library(tidyverse)

## -- Attaching packages -------------------------------------------------------------------- tidyverse 1.3.0 --

## v tibble 3.0.1 v dplyr 1.0.0  
## v tidyr 1.1.0 v stringr 1.4.0  
## v readr 1.3.1 v forcats 0.5.0  
## v purrr 0.3.4

## -- Conflicts ----------------------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::combine() masks gridExtra::combine()  
## x dplyr::filter() masks plotly::filter(), stats::filter()  
## x dplyr::first() masks xts::first()  
## x dplyr::lag() masks stats::lag()  
## x dplyr::last() masks xts::last()  
## x dplyr::src() masks Hmisc::src()  
## x dplyr::summarize() masks Hmisc::summarize()

options(scipen=999)  
  
cd = read.csv("C:/Users/adety/Desktop/Courses/544/Project/consolidated\_coin\_data.csv")  
  
# Initial view of the data  
  
head(cd)

## Currency Date Open High Low Close Volume Market.Cap  
## 1 tezos Dec 04, 2019 1.29 1.32 1.25 1.25 46,048,752 824,588,509  
## 2 tezos Dec 03, 2019 1.24 1.32 1.21 1.29 41,462,224 853,213,342  
## 3 tezos Dec 02, 2019 1.25 1.26 1.20 1.24 27,574,097 817,872,179  
## 4 tezos Dec 01, 2019 1.33 1.34 1.25 1.25 24,127,567 828,296,390  
## 5 tezos Nov 30, 2019 1.31 1.37 1.31 1.33 28,706,667 879,181,680  
## 6 tezos Nov 29, 2019 1.28 1.34 1.28 1.31 32,270,224 867,085,098

tail(cd)

## Currency Date Open High Low Close Volume Market.Cap  
## 28939 bitcoin-sv May 03, 2013 3.39 3.45 2.40 3.04 0 52,694,847  
## 28940 bitcoin-sv May 02, 2013 3.78 4.04 3.01 3.37 0 58,287,979  
## 28941 bitcoin-sv May 01, 2013 4.29 4.36 3.52 3.80 0 65,604,596  
## 28942 bitcoin-sv Apr 30, 2013 4.40 4.57 4.17 4.30 0 74,020,918  
## 28943 bitcoin-sv Apr 29, 2013 4.37 4.57 4.23 4.38 0 75,388,964  
## 28944 bitcoin-sv Apr 28, 2013 4.30 4.40 4.18 4.35 0 74,636,938

summary(cd)

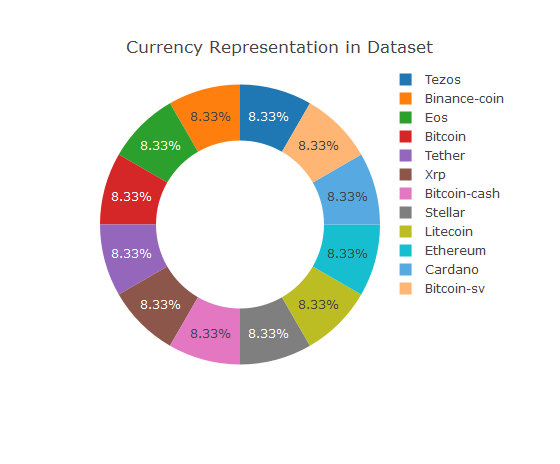
## Currency Date Open High   
## Length:28944 Length:28944 Length:28944 Length:28944   
## Class :character Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character Mode :character   
## Low Close Volume Market.Cap   
## Length:28944 Length:28944 Length:28944 Length:28944   
## Class :character Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character Mode :character

# summary function revels that all features are saved as of character type.   
# converting each feature to its appropriate format.  
# removing comma's and then converting to numeric as otherwise numeric introduces   
# NA's.  
# Add new feature year  
  
cd$Date = as.Date(cd$Date, '%b %d, %Y')  
cd$Open = as.numeric(gsub(pattern = ',','',cd$Open))  
cd$High = as.numeric(gsub(pattern = ',','',cd$High))  
cd$Low = as.numeric(gsub(pattern = ',','',cd$Low))  
cd$Close = as.numeric(gsub(pattern = ',','',cd$Close))  
cd$Volume = as.numeric(gsub(pattern = ',','',cd$Volume))  
cd$Market.Cap = as.numeric(gsub(pattern = ',','',cd$Market.Cap))  
cd$Currency = capitalize(cd$Currency)  
cd$year = as.numeric(format(cd$Date,"%Y"))  
  
# Checking data again  
  
summary(cd)

## Currency Date Open   
## Length:28944 Min. :2013-04-28 Min. : 0.001   
## Class :character 1st Qu.:2014-12-21 1st Qu.: 0.205   
## Mode :character Median :2016-08-15 Median : 2.995   
## Mean :2016-08-15 Mean : 300.720   
## 3rd Qu.:2018-04-10 3rd Qu.: 24.430   
## Max. :2019-12-04 Max. :19475.800   
## High Low Close   
## Min. : 0.002 Min. : 0.001 Min. : 0.001   
## 1st Qu.: 0.212 1st Qu.: 0.197 1st Qu.: 0.205   
## Median : 3.090 Median : 2.880 Median : 2.980   
## Mean : 309.833 Mean : 290.859 Mean : 300.948   
## 3rd Qu.: 25.530 3rd Qu.: 23.270 3rd Qu.: 24.430   
## Max. :20089.000 Max. :18974.100 Max. :19497.400   
## Volume Market.Cap year   
## Min. : 0 Min. : 0 Min. :2013   
## 1st Qu.: 241870 1st Qu.: 63451426 1st Qu.:2014   
## Median : 5212684 Median : 345367261 Median :2016   
## Mean : 813305774 Mean : 7194826310 Mean :2016   
## 3rd Qu.: 155476420 3rd Qu.: 3422402834 3rd Qu.:2018   
## Max. :53509128965 Max. :326502485530 Max. :2019

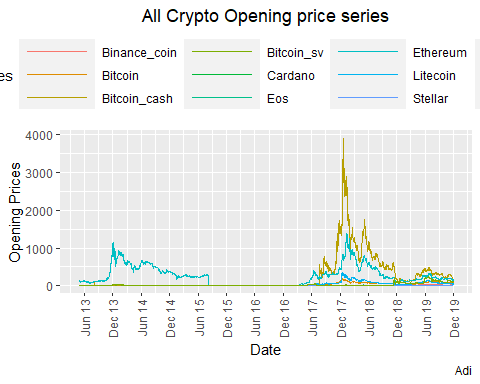
# checking which currencies are in the data set.  
  
plot\_ly(as.data.frame(cd$Currency),labels = cd$Currency,type = 'pie',hole = 0.6) %>%  
 layout(title = "Currency Representation in Dataset",  
 xaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE),  
 yaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE))

## Warning: `arrange\_()` is deprecated as of dplyr 0.7.0.  
## Please use `arrange()` instead.  
## See vignette('programming') for more help  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_warnings()` to see where this warning was generated.

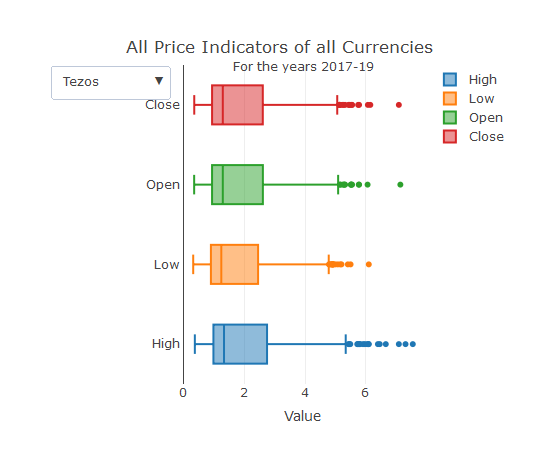


# Creating subsets of each currency for easier analyses  
# Also dropping the name cols for individual currencies and Changing Names  
  
Bitcoin\_sv = subset(cd,cd$Currency == 'Bitcoin-sv')  
Bitcoin\_sv$Currency = NULL  
colnames(Bitcoin\_sv) = c('Bitcoin\_svDate','Bitcoin\_svOpen','Bitcoin\_svHigh',  
 'Bitcoin\_svLow','Bitcoin\_svClose','Bitcoin\_svVolume',  
 'Bitcoin\_svMarket.Cap')  
Bitcoin\_sv = xts(Bitcoin\_sv[,-1], order.by=Bitcoin\_sv[,1])  
  
Bitcoin\_cash = subset(cd,cd$Currency == 'Bitcoin-cash')  
Bitcoin\_cash$Currency = NULL  
colnames(Bitcoin\_cash) = c('Bitcoin\_cashDate','Bitcoin\_cashOpen','Bitcoin\_cashHigh',  
 'Bitcoin\_cashLow','Bitcoin\_cashClose','Bitcoin\_cashVolume',  
 'Bitcoin\_cashMarket.Cap')  
Bitcoin\_cash = xts(Bitcoin\_cash[,-1], order.by=Bitcoin\_cash[,1])  
  
Bitcoin = subset(cd,cd$Currency == 'Bitcoin')  
Bitcoin$Currency = NULL  
colnames(Bitcoin) = c('BitcoinDate','BitcoinOpen','BitcoinHigh',  
 'BitcoinLow','BitcoinClose','BitcoinVolume',  
 'BitcoinMarket.Cap')  
Bitcoin = xts(Bitcoin[,-1], order.by=Bitcoin[,1])  
  
Cardano = subset(cd,cd$Currency == 'Cardano')  
Cardano$Currency = NULL  
colnames(Cardano) = c('CardanoDate','CardanoOpen','CardanoHigh',  
 'CardanoLow','CardanoClose','CardanoVolume',  
 'CardanoMarket.Cap')  
Cardano = xts(Cardano[,-1], order.by=Cardano[,1])  
  
Eos = subset(cd,cd$Currency == 'Eos')  
Eos$Currency = NULL  
colnames(Eos) = c('EosDate','EosOpen','EosHigh',  
 'EosLow','EosClose','EosVolume',  
 'EosMarket.Cap')  
Eos = xts(Eos[,-1], order.by=Eos[,1])  
  
Ethereum = subset(cd,cd$Currency == 'Ethereum')  
Ethereum$Currency = NULL  
colnames(Ethereum) = c('EthereumDate','EthereumOpen','EthereumHigh',  
 'EthereumLow','EthereumClose','EthereumVolume',  
 'EthereumMarket.Cap')  
Ethereum = xts(Ethereum[,-1], order.by=Ethereum[,1])  
  
Litecoin = subset(cd,cd$Currency == 'Litecoin')  
Litecoin$Currency = NULL  
colnames(Litecoin) = c('LitecoinDate','LitecoinOpen','LitecoinHigh',  
 'LitecoinLow','LitecoinClose','LitecoinVolume',  
 'LitecoinMarket.Cap')  
Litecoin = xts(Litecoin[,-1], order.by=Litecoin[,1])  
  
Stellar = subset(cd,cd$Currency == 'Stellar')  
Stellar$Currency = NULL  
colnames(Stellar) = c('StellarDate','StellarOpen','StellarHigh',  
 'StellarLow','StellarClose','StellarVolume',  
 'StellarMarket.Cap')  
Stellar = xts(Stellar[,-1], order.by=Stellar[,1])  
  
Tether = subset(cd,cd$Currency == 'Tether')  
Tether$Currency = NULL  
colnames(Tether) = c('TetherDate','TetherOpen','TetherHigh',  
 'TetherLow','TetherClose','TetherVolume',  
 'TetherMarket.Cap')  
Tether = xts(Tether[,-1], order.by=Tether[,1])  
  
Tezos = subset(cd,cd$Currency == 'Tezos')  
Tezos$Currency = NULL  
colnames(Tezos) = c('TezosDate','TezosOpen','TezosHigh',  
 'TezosLow','TezosClose','TezosVolume',  
 'TezosMarket.Cap')  
Tezos = xts(Tezos[,-1], order.by=Tezos[,1])  
  
Xrp = subset(cd,cd$Currency == 'Xrp')  
Xrp$Currency = NULL  
colnames(Xrp) = c('XrpDate','XrpOpen','XrpHigh',  
 'XrpLow','XrpClose','XrpVolume',  
 'XrpMarket.Cap')  
Xrp = xts(Xrp[,-1], order.by=Xrp[,1])  
  
Binance\_coin = subset(cd,cd$Currency == 'Binance-coin')  
Binance\_coin$Currency = NULL  
colnames(Binance\_coin) = c('Binance\_coinDate','Binance\_coinOpen','Binance\_coinHigh',  
 'Binance\_coinLow','Binance\_coinClose','Binance\_coinVolume',  
 'Binance\_coinMarket.Cap')  
Binance\_coin = xts(Binance\_coin[,-1], order.by=Binance\_coin[,1])  
  
# Plot opening prices of all the currencies.  
  
ggplot(Bitcoin, aes(x = index(Bitcoin),y = Bitcoin[,1]/100,color = 'Bitcoin')) +   
 geom\_line() +   
 ggtitle("All Crypto Opening price series") +   
 theme(legend.position = "top") +   
 theme(axis.text.x = element\_text(angle = 90, vjust = 0.5, hjust=1)) +  
 labs(  
 x = "Date",  
 y = "Opening Prices",  
 color = "Currencies",  
 caption = "Adi") +  
 scale\_x\_date(date\_labels = "%b %y", date\_breaks = "6 months") +  
 theme(plot.title = element\_text(hjust = 0.5)) +  
 theme(plot.subtitle = element\_text(hjust = 0.5)) +  
 theme(legend.key.width = unit(2,"cm")) +  
 geom\_line(aes(y = Xrp[,1],color = 'Xrp')) +  
 geom\_line(aes(y = Cardano[,1],,color = 'Cardano')) +   
 geom\_line(aes(y = Tezos[,1],color = 'Tezos')) +  
 geom\_line(aes(y = Binance\_coin[,1],color = 'Binance\_coin')) +  
 geom\_line(aes(y = Eos[,1],color = 'Eos')) +  
 geom\_line(aes(y = Tether[,1],color = 'Tether')) +  
 geom\_line(aes(y = Bitcoin\_cash[,1],color = 'Bitcoin\_cash')) +  
 geom\_line(aes(y = Stellar[,1],color = 'Stellar')) +  
 geom\_line(aes(y = Litecoin[,1],color = 'Litecoin')) +   
 geom\_line(aes(y = Ethereum[,1],color = 'Ethereum')) +  
 geom\_line(aes(y = Bitcoin\_sv[,1],color = 'Bitcoin\_sv'))

## Don't know how to automatically pick scale for object of type xts/zoo. Defaulting to continuous.

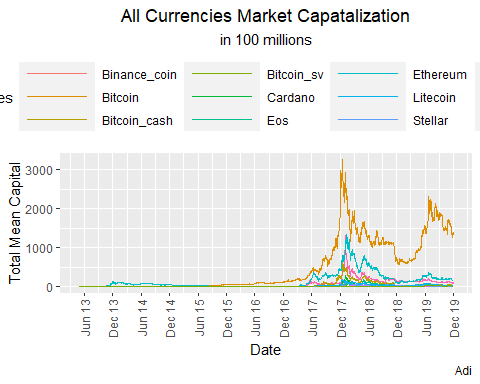


# Boxplots all prices all currencies  
  
cd %>%  
 filter(year == '2018'| year == '2019' | year == '2019') %>%  
 plot\_ly(type = 'box',x = ~High,transforms = list(list(type = 'filter',  
 target = ~Currency,  
 operation = '=',  
 value = unique(cd$Currency)[1])),name="High") %>%  
 add\_boxplot(x = ~Low,name = 'Low') %>%  
 add\_boxplot(x = ~Open,name = 'Open') %>%  
 add\_boxplot(x = ~Close,name = 'Close') %>%  
   
 layout(title = 'All Price Indicators of all Currencies',  
 xaxis = list(title = 'Value'),   
 updatemenus = list(list(type = 'dropdown',buttons = list(  
 list(method = "restyle",args = list("transforms[0].value", unique(cd$Currency)[1]),  
 label = unique(cd$Currency)[1]),  
 list(method = "restyle",args = list("transforms[0].value", unique(cd$Currency)[2]),  
 label = unique(cd$Currency)[2]),  
 list(method = "restyle",args = list("transforms[0].value", unique(cd$Currency)[3]),  
 label = unique(cd$Currency)[3]),  
 list(method = "restyle",args = list("transforms[0].value", unique(cd$Currency)[4]),  
 label = unique(cd$Currency)[4]),  
 list(method = "restyle",args = list("transforms[0].value", unique(cd$Currency)[5]),  
 label = unique(cd$Currency)[5]),  
 list(method = "restyle",args = list("transforms[0].value", unique(cd$Currency)[6]),  
 label = unique(cd$Currency)[6]),  
 list(method = "restyle",args = list("transforms[0].value", unique(cd$Currency)[7]),  
 label = unique(cd$Currency)[7]),  
 list(method = "restyle",args = list("transforms[0].value", unique(cd$Currency)[8]),  
 label = unique(cd$Currency)[8]),  
 list(method = "restyle",args = list("transforms[0].value", unique(cd$Currency)[9]),  
 label = unique(cd$Currency)[9]),  
 list(method = "restyle",args = list("transforms[0].value", unique(cd$Currency)[10]),  
 label = unique(cd$Currency)[10]),  
 list(method = "restyle",args = list("transforms[0].value", unique(cd$Currency)[11]),  
 label = unique(cd$Currency)[11]),list(method = "restyle",  
 args = list("transforms[0].value", unique(cd$Currency)[12]),  
 label = unique(cd$Currency)[12]))))) %>%  
 layout(annotations = list(list(text = "For the years 2017-19", xref = "paper", yref = "paper",  
 yanchor = "bottom",xanchor = "center", align = "center",  
 x = 0.5, y = .97, showarrow = FALSE)))

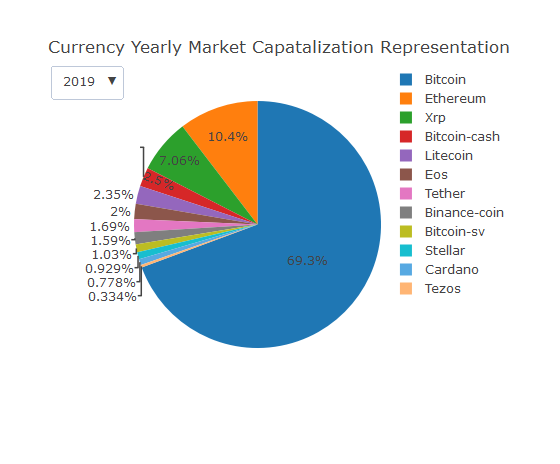


# Plot market capitalization   
  
ggplot(Bitcoin, aes(x = index(Bitcoin),y = Bitcoin[,6]/100000000,color = 'Bitcoin')) +   
 geom\_line() +   
 ggtitle("All Crypto Market Cap") +   
 theme(legend.position = "top") +   
 theme(axis.text.x = element\_text(angle = 90, vjust = 0.5, hjust=1)) +  
 labs(  
 x = "Date",  
 y = "Total Mean Capital",  
 color = "Currencies",  
 title = "All Currencies Market Capatalization",  
 subtitle = "in 100 millions",  
 caption = "Adi") +  
 scale\_x\_date(date\_labels = "%b %y", date\_breaks = "6 months") +  
 theme(plot.title = element\_text(hjust = 0.5)) +  
 theme(plot.subtitle = element\_text(hjust = 0.5)) +  
 theme(legend.key.width = unit(2,"cm")) +  
 geom\_line(aes(y = Xrp[,6]/100000000,color = 'Xrp')) +  
 geom\_line(aes(y = Cardano[,6]/100000000,,color = 'Cardano')) +   
 geom\_line(aes(y = Tezos[,6]/100000000,color = 'Tezos')) +  
 geom\_line(aes(y = Binance\_coin[,6]/100000000,color = 'Binance\_coin')) +  
 geom\_line(aes(y = Eos[,6]/100000000,color = 'Eos')) +  
 geom\_line(aes(y = Tether[,6]/100000000,color = 'Tether')) +  
 geom\_line(aes(y = Bitcoin\_cash[,6]/100000000,color = 'Bitcoin\_cash')) +  
 geom\_line(aes(y = Stellar[,6]/100000000,color = 'Stellar')) +  
 geom\_line(aes(y = Litecoin[,6]/100000000,color = 'Litecoin')) +   
 geom\_line(aes(y = Ethereum[,6]/100000000,color = 'Ethereum')) +  
 geom\_line(aes(y = Bitcoin\_sv[,6]/100000000,color = 'Bitcoin\_sv'))

## Don't know how to automatically pick scale for object of type xts/zoo. Defaulting to continuous.

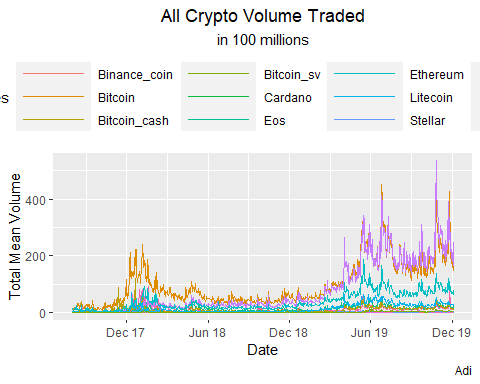


# Plot market cap yearly all coins  
  
cd %>%  
 plot\_ly(labels = ~Currency,values = ~Market.Cap, type = 'pie',  
 transforms = list(list(type = 'filter',target = ~year,operation = '=',  
 value = unique(cd$year)[1])),name="2013") %>%  
 layout(title = "Currency Yearly Market Capatalization Representation",  
 xaxis = list(title = 'Value'),   
updatemenus = list(list(type = 'dropdown',buttons = list(  
 list(method = "restyle",args = list("transforms[0].value", unique(cd$year)[1]),  
 label = unique(cd$year)[1]),  
 list(method = "restyle",args = list("transforms[0].value", unique(cd$year)[2]),  
 label = unique(cd$year)[2]),  
 list(method = "restyle",args = list("transforms[0].value", unique(cd$year)[3]),  
 label = unique(cd$year)[3]),  
 list(method = "restyle",args = list("transforms[0].value", unique(cd$year)[4]),  
 label = unique(cd$year)[4]),  
 list(method = "restyle",args = list("transforms[0].value", unique(cd$year)[5]),  
 label = unique(cd$year)[5]),  
 list(method = "restyle",args = list("transforms[0].value", unique(cd$year)[6]),  
 label = unique(cd$year)[6]),  
 list(method = "restyle",args = list("transforms[0].value", unique(cd$year)[7]),  
 label = unique(cd$year)[7])))))

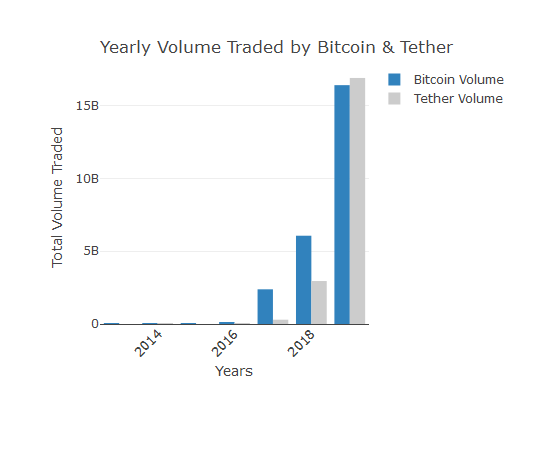


# All Currencies Volume Traded.  
  
ggplot(Bitcoin['2017-08/'], aes(x = index(Bitcoin['2017-08/']),  
 y = Bitcoin['2017-08/'][,5]/100000000,  
 color = 'Bitcoin')) +   
 geom\_line() +   
 ggtitle("All Crypto Volume Traded") +   
 theme(legend.position = "top") +   
 labs(  
 x = "Date",  
 y = "Total Mean Volume",  
 color = "Currencies",  
 subtitle = "in 100 millions",  
 caption = "Adi") +  
 theme(plot.title = element\_text(hjust = 0.5)) +  
 theme(plot.subtitle = element\_text(hjust = 0.5)) +  
 scale\_x\_date(date\_labels = "%b %y", date\_breaks = "6 months") +  
 theme(legend.key.width = unit(2,"cm")) +  
 geom\_line(aes(y = Xrp['2017-08/'][,5]/100000000,color = 'Xrp')) +  
 geom\_line(aes(y = Cardano['2017-08/'][,5]/100000000,color = 'Cardano')) +   
 geom\_line(aes(y = Tezos['2017-08/'][,5]/100000000,color = 'Tezos')) +  
 geom\_line(aes(y = Binance\_coin['2017-08/'][,5]/100000000,color = 'Binance\_coin')) +  
 geom\_line(aes(y = Eos['2017-08/'][,5]/100000000,color = 'Eos')) +  
 geom\_line(aes(y = Tether['2017-08/'][,5]/100000000,color = 'Tether')) +  
 geom\_line(aes(y = Bitcoin\_cash['2017-08/'][,5]/100000000,color = 'Bitcoin\_cash')) +  
 geom\_line(aes(y = Stellar['2017-08/'][,5]/100000000,color = 'Stellar')) +  
 geom\_line(aes(y = Litecoin['2017-08/'][,5]/100000000,color = 'Litecoin')) +   
 geom\_line(aes(y = Ethereum['2017-08/'][,5]/100000000,color = 'Ethereum')) +  
 geom\_line(aes(y = Bitcoin\_sv['2017-08/'][,5]/100000000,color = 'Bitcoin\_sv'))

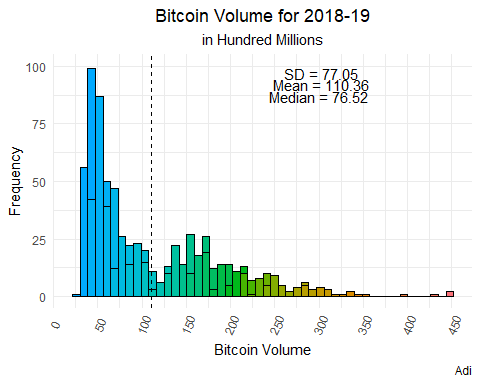
## Don't know how to automatically pick scale for object of type xts/zoo. Defaulting to continuous.



# Tether & Bitcoin volume in the last two years.  
  
Bitcoin$year = as.numeric(format(index(Bitcoin),"%Y"))  
Tether$year = as.numeric(format(index(Tether),"%Y"))  
  
Vol\_yrly\_BT = aggregate(BitcoinVolume ~ year, data=Bitcoin , FUN= mean)  
Vol\_yrly\_BT$TetherVolume = aggregate(TetherVolume ~ year, data=Tether , FUN= mean)[2]  
  
Vol\_yrly\_BT$TetherVolume = as.numeric(unlist(Vol\_yrly\_BT$TetherVolume))  
  
plot\_ly(Vol\_yrly\_BT,x =~year ,y = ~BitcoinVolume, type = 'bar',name = 'Bitcoin Volume',  
 marker = list(color = 'rgb(49,130,189)')) %>%  
 add\_trace(y = ~TetherVolume, name = 'Tether Volume',  
 marker = list(color = 'rgb(204,204,204)')) %>%  
 layout(title = 'Yearly Volume Traded by Bitcoin & Tether',  
 xaxis = list(title = 'Years',tickangle = -45),  
 yaxis = list(title = 'Total Volume Traded'),  
 margin = list(b=100),  
 barmode = 'group')

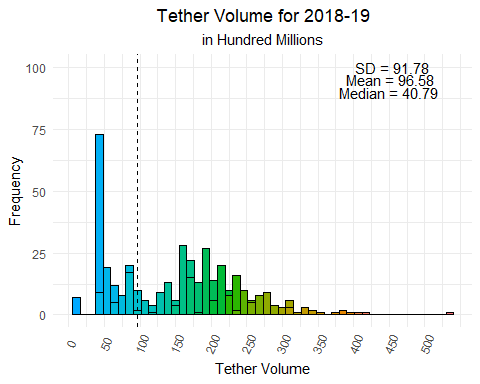


# The distribution of the data.  
# Volume of last two years of Bitcoin and Tether  
# Correlation between volume of trading and market capitalization  
  
Bitcoin1819 = subset(Bitcoin,year == '2018' | year == '2019')  
Tether1819 = subset(Tether,year == '2018' | year == '2019')  
  
# Bitcoin Volume Chart  
  
ggplot(Bitcoin1819,aes(x = BitcoinVolume/100000000, fill = cut(BitcoinVolume,30))) +  
 geom\_histogram(show.legend = FALSE,bins = 50,color = 'black') +   
 scale\_fill\_discrete(h = c(240,10)) +   
 theme\_minimal() +  
 labs(x = 'Bitcoin Volume', y = 'Frequency',title = 'Bitcoin Volume for 2018-19',  
 subtitle = 'in Hundred Millions',caption = "Adi" ) +  
 theme(axis.text.x = element\_text(angle=65, vjust=0.6)) +  
 theme(plot.title = element\_text(hjust = 0.5)) +  
 theme(plot.subtitle = element\_text(hjust = 0.5)) +  
 annotate("text", x=303, y=92, label= paste("Mean =",round(mean(Bitcoin1819$BitcoinVolume/100000000),2))) +  
 annotate("text", x=303, y=97, label= paste("SD =",round(sd(Bitcoin1819$BitcoinVolume/100000000),2))) +  
 annotate("text", x=300, y=87, label= paste("Median =",round(median(Bitcoin1819$BitcoinVolume/100000000),2))) +  
 scale\_x\_continuous(breaks = seq(0, 500, 50)) +  
 scale\_y\_continuous(limits = c(0, 100))+  
 geom\_vline(aes(xintercept = mean(Bitcoin1819$BitcoinVolume/100000000)), linetype = "dashed")



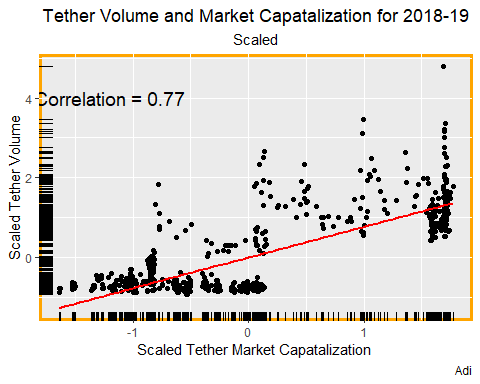
# Tether Volume Chart   
  
ggplot(Tether1819,aes(TetherVolume/100000000, fill = cut(TetherVolume,30))) +   
 geom\_histogram(show.legend = FALSE,bins = 50,color = 'black',size = .001) +   
 scale\_fill\_discrete(h = c(240,10)) +   
 theme\_minimal() +  
 labs(x = 'Tether Volume', y = 'Frequency',title = 'Tether Volume for 2018-19',  
 subtitle = 'in Hundred Millions' ) +  
 theme(axis.text.x = element\_text(angle=65, vjust=0.6)) +  
 theme(plot.title = element\_text(hjust = 0.5)) +  
 theme(plot.subtitle = element\_text(hjust = 0.5)) +  
 scale\_x\_continuous(breaks = seq(0, 500, 50)) +  
 annotate("text", x=452, y=95, label= paste("Mean =",round(mean(Tether1819$TetherVolume/100000000),2))) +  
 annotate("text", x=454, y=100, label= paste("SD =",round(sd(Tether1819$TetherVolume/100000000),2))) +  
 annotate("text", x=450, y=90, label= paste("Median =",round(median(Tether1819$TetherVolume/100000000),2))) +  
 scale\_y\_continuous(limits = c(0, 100)) +  
 geom\_vline(aes(xintercept = mean(Tether1819$TetherVolume/100000000)), linetype = "dashed")

## Warning: Removed 3 rows containing missing values (geom\_bar).



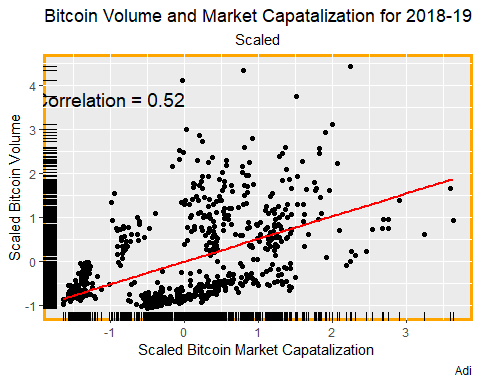
# Correlation Plots  
  
ggplot(Tether1819, aes(x=scale(TetherMarket.Cap), y=scale(TetherVolume))) +   
 geom\_point() +  
 geom\_smooth(method=lm, se=FALSE, fullrange=TRUE,color = 'red') +   
 geom\_rug()+  
 theme(plot.title = element\_text(hjust = 0.5)) +  
 theme(plot.subtitle = element\_text(hjust = 0.5)) +  
 labs(x = 'Scaled Tether Market Capatalization ', y = 'Scaled Tether Volume',  
 title = 'Tether Volume and Market Capatalization for 2018-19',  
 subtitle = 'Scaled',caption = "Adi" ) +  
 theme(panel.background = element\_rect(colour = "orange",size = 2,linetype = "solid")) +  
 annotate("text", x=-1.20, y=4,size = 5, label= paste("Correlation =",  
 round(cor(Tether1819$TetherVolume,Tether1819$TetherMarket.Cap),2)))

## `geom\_smooth()` using formula 'y ~ x'

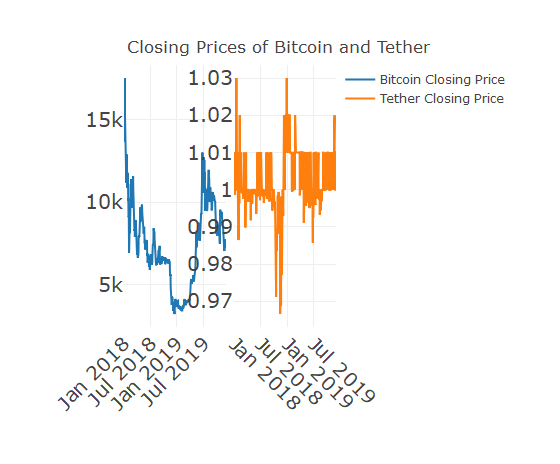


ggplot(Bitcoin1819, aes(x=scale(BitcoinMarket.Cap), y=scale(BitcoinVolume))) +   
 geom\_point() +  
 geom\_smooth(method=lm, se=FALSE, fullrange=TRUE,color = 'red') +   
 geom\_rug()+  
 theme(plot.title = element\_text(hjust = 0.5)) +  
 theme(plot.subtitle = element\_text(hjust = 0.5)) +  
 labs(x = 'Scaled Bitcoin Market Capatalization ', y = 'Scaled Bitcoin Volume',  
 title = 'Bitcoin Volume and Market Capatalization for 2018-19',  
 subtitle = 'Scaled',caption = "Adi" ) +  
 theme(panel.background = element\_rect(colour = 'orange',size = 2,   
 linetype = "solid"))+  
 annotate("text", x=-1, y=3.7,size = 5, label= paste("Correlation =",  
 round(cor(Bitcoin1819$BitcoinVolume,Bitcoin1819$BitcoinMarket.Cap),2)))

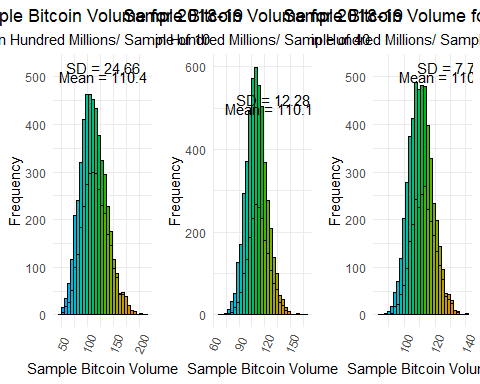
## `geom\_smooth()` using formula 'y ~ x'



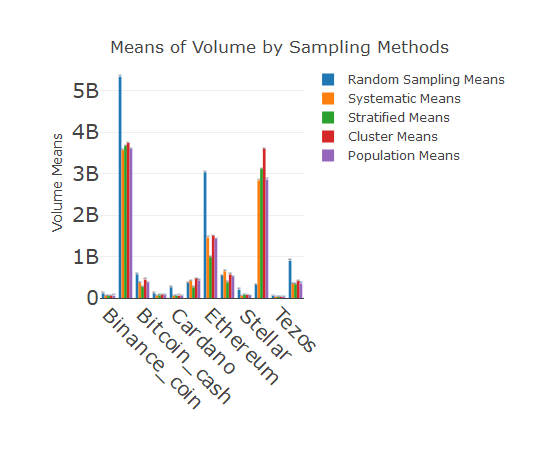
# Plot closing prices Tether & Bitcoin  
  
bitcoin\_close = plot\_ly(as.data.frame(Bitcoin1819),y = ~BitcoinClose, x = ~index(Bitcoin1819),   
 type = 'scatter',mode = 'lines',name = 'Bitcoin Closing Price') %>%  
layout(xaxis = list(title = 'Years',tickangle = -45,tickfont = list(size = 20)),  
 yaxis = list(title = 'Price Range',tickfont = list(size = 20)))  
  
tether\_close = plot\_ly(as.data.frame(Tether1819),y = ~TetherClose, x = ~index(Tether1819),   
 type = 'scatter',mode = 'lines',name = 'Tether Closing Price') %>%  
 layout(xaxis = list(title = 'Years',tickangle = 45,tickfont = list(size = 20)),  
 yaxis = list(title = 'Price Range',tickfont = list(size = 20)))  
  
subplot(bitcoin\_close, tether\_close) %>% layout(title="Closing Prices of Bitcoin and Tether",font = 20)



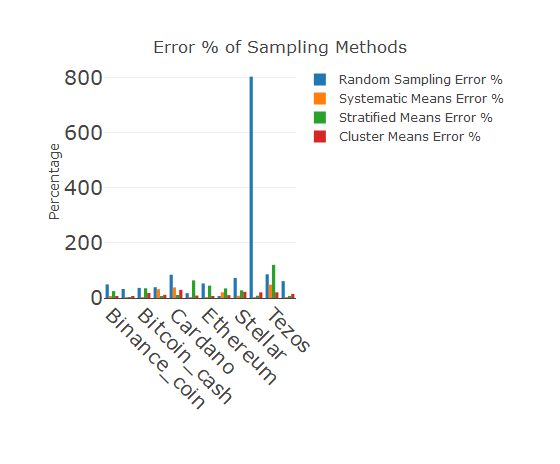
# Central Limit Theorem  
  
# Sample of 10  
xbar10 = c()  
  
for (i in 1:5000) {  
 xbar10[i] = mean(sample(x = Bitcoin1819$BitcoinVolume,size = 10,replace = TRUE))  
}  
xbar10 = as.numeric(unlist(xbar10))  
  
# Plot sample 10  
  
p1 = ggplot(mapping = aes(xbar10/100000000, fill = cut(xbar10,30))) +   
 geom\_histogram(show.legend = FALSE,color = 'black',bins = 30,size = .001) +   
 scale\_fill\_discrete(h = c(240,10)) +   
 theme\_minimal() +  
 labs(x = 'Sample Bitcoin Volume', y = 'Frequency',title = 'Sample Bitcoin Volume for 2018-19',  
 subtitle = 'in Hundred Millions/ Sample of 10') +  
 theme(axis.text.x = element\_text(angle=65, vjust=0.6)) +  
 theme(plot.title = element\_text(hjust = 0.5)) +  
 theme(plot.subtitle = element\_text(hjust = 0.5)) +  
 annotate("text", x=130, y=500, label= paste("Mean =",round(mean(xbar10/100000000),2))) +  
 annotate("text", x=130, y=520, label= paste("SD =",round(sd(xbar10/100000000),2)))  
  
# Sample of 40  
  
xbar40 = c()  
  
for (i in 1:5000) {  
 xbar40[i] = mean(sample(x = Bitcoin1819$BitcoinVolume,size = 40,replace = TRUE))  
}  
xbar40 = as.numeric(unlist(xbar40))  
  
# Plot sample 40  
  
p2 = ggplot(mapping = aes(xbar40/100000000, fill = cut(xbar40,30))) +   
 geom\_histogram(show.legend = FALSE,,bins = 30,color = 'black',size = .001) +   
 scale\_fill\_discrete(h = c(240,10)) +   
 theme\_minimal() +  
 labs(x = 'Sample Bitcoin Volume', y = 'Frequency',title = 'Sample Bitcoin Volume for 2018-19',  
 subtitle = 'in Hundred Millions/ Sample of 40' ) +  
 theme(axis.text.x = element\_text(angle=65, vjust=0.6)) +  
 theme(plot.title = element\_text(hjust = 0.5)) +  
 theme(plot.subtitle = element\_text(hjust = 0.5)) +  
 annotate("text", x=130, y=500, label= paste("Mean =",round(mean(xbar40/100000000),2))) +  
 annotate("text", x=130, y=520, label= paste("SD =",round(sd(xbar40/100000000),2)))  
  
# Sample of 100  
  
xbar100 = c()  
  
for (i in 1:5000) {  
 xbar100[i] = mean(sample(x = Bitcoin1819$BitcoinVolume,size = 100,replace = TRUE))  
}  
xbar100 = as.numeric(unlist(xbar100))  
  
# Plot sample 100  
  
p3 = ggplot(mapping = aes(xbar100/100000000, fill = cut(xbar100,30))) +   
 geom\_histogram(show.legend = FALSE,,bins = 30,color = 'black',size = .001) +  
 scale\_fill\_discrete(h = c(240,10)) +   
 theme\_minimal() +  
 labs(x = 'Sample Bitcoin Volume', y = 'Frequency',title = 'Sample Bitcoin Volume for 2018-19',  
 subtitle = 'in Hundred Millions/ Sample of 100' ) +  
 theme(axis.text.x = element\_text(angle=65, vjust=0.6)) +  
 theme(plot.title = element\_text(hjust = 0.5)) +  
 theme(plot.subtitle = element\_text(hjust = 0.5)) +  
 annotate("text", x=128, y=500, label= paste("Mean =",round(mean(xbar100/100000000),2))) +  
 annotate("text", x=130, y=520, label= paste("SD =",round(sd(xbar100/100000000),2)))  
  
# Plot together  
  
grid.arrange(p1, p2, p3, nrow = 1)



################################################################################  
  
# Sampling Methods  
  
# random sampling  
  
set.seed(100)  
  
Currency\_list = c("Tezos","Binance\_coin","Eos","Bitcoin","Tether","Xrp","Bitcoin\_cash",  
 "Stellar","Litecoin","Ethereum","Cardano","Bitcoin\_sv")  
  
random\_sampling\_means = c()  
count = 1  
  
for (i in Currency\_list){  
 random\_sampling\_means[count] = mean(get(i)[(srswor(50,nrow(get(i)[,5])))!=0,])  
 count = count + 1  
}  
  
sampling\_DF = as.data.frame(random\_sampling\_means,Currency\_list)  
  
# Systematic Sampling  
  
set.seed(100)  
  
N = nrow(Bitcoin)  
n = 50  
k = floor(N / n)  
r = sample(k, 1)  
s = seq(r, by = k, length = n)  
  
systematic\_sampling\_means = c()  
count = 1  
  
for (i in Currency\_list){  
 systematic\_sampling\_means[count] = mean(get(i)[s,][,5])  
 count = count + 1  
}  
sampling\_DF$systematic\_sampling\_means = systematic\_sampling\_means  
  
# Stratified Sampling  
  
# Pick 'year' Strata from each currency  
  
freq = table(cd$Currency, cd$year)  
st.sizes =as.vector(t(round((595 \* freq / sum(freq))))) # 595 so as to get 50 samples for each currency  
st.sizes = st.sizes[st.sizes != 0]  
  
st.3 = strata(cd,stratanames = c('Currency','year'),  
 size = st.sizes, method = 'srswor',  
 description = FALSE)  
  
stratified\_sampling\_means = getdata(cd, st.3)  
stratified\_sampling\_means = tapply(stratified\_sampling\_means$Volume, stratified\_sampling\_means$Currency, mean)  
stratified\_sampling\_means = as.data.frame(stratified\_sampling\_means)  
sampling\_DF$stratified\_sampling\_means = stratified\_sampling\_means[c(11,1,6,2,10,12,3,9,8,7,5,4),]  
  
# Cluster  
  
cluster\_sampling\_means = c()  
  
# Assign random letters to create clusters  
cd$category = sample(LETTERS[1:12],nrow(cd),replace = TRUE)  
cluster\_data = getdata(cd,(cluster(cd,c('category'),6,'srswor')))  
  
data\_Tezos = cluster\_data[cluster\_data$Currency == 'Tezos',]  
sample\_Tezos = srswor(50,nrow(data\_Tezos))  
cluster\_sampling\_means[1] = mean(data\_Tezos[sample\_Tezos != 0,][,7])  
  
data\_Binance\_coin = cluster\_data[cluster\_data$Currency == 'Binance-coin',]  
sample\_Binance\_coin = srswor(50,nrow(data\_Binance\_coin))  
cluster\_sampling\_means[2] = mean(data\_Binance\_coin[sample\_Binance\_coin != 0,][,7])  
  
data\_Eos = cluster\_data[cluster\_data$Currency == 'Eos',]  
sample\_Eos = srswor(50,nrow(data\_Eos))  
cluster\_sampling\_means[3] = mean(data\_Eos[sample\_Eos != 0,][,7])  
  
data\_Bitcoin = cluster\_data[cluster\_data$Currency == 'Bitcoin',]  
sample\_Bitcoin = srswor(50,nrow(data\_Bitcoin))  
cluster\_sampling\_means[4] = mean(data\_Bitcoin[sample\_Bitcoin != 0,][,7])  
  
data\_Tether = cluster\_data[cluster\_data$Currency == 'Tether',]  
sample\_Tether = srswor(50,nrow(data\_Tether))  
cluster\_sampling\_means[5] = mean(data\_Tether[sample\_Tether != 0,][,7])  
  
data\_Xrp = cluster\_data[cluster\_data$Currency == 'Xrp',]  
sample\_Xrp = srswor(50,nrow(data\_Xrp))  
cluster\_sampling\_means[6] = mean(data\_Xrp[sample\_Xrp != 0,][,7])  
  
data\_Bitcoin\_cash = cluster\_data[cluster\_data$Currency == 'Bitcoin-cash',]  
sample\_Bitcoin\_cash = srswor(50,nrow(data\_Bitcoin\_cash))  
cluster\_sampling\_means[7] = mean(data\_Bitcoin\_cash[sample\_Bitcoin\_cash != 0,][,7])  
  
data\_Stellar = cluster\_data[cluster\_data$Currency == 'Stellar',]  
sample\_Stellar = srswor(50,nrow(data\_Stellar))  
cluster\_sampling\_means[8] = mean(data\_Stellar[sample\_Stellar != 0,][,7])  
  
data\_Litecoin = cluster\_data[cluster\_data$Currency == 'Litecoin',]  
sample\_Litecoin = srswor(50,nrow(data\_Litecoin))  
cluster\_sampling\_means[9] = mean(data\_Litecoin[sample\_Litecoin != 0,][,7])  
  
data\_Ethereum = cluster\_data[cluster\_data$Currency == 'Ethereum',]  
sample\_Ethereum = srswor(50,nrow(data\_Ethereum))  
cluster\_sampling\_means[10] = mean(data\_Ethereum[sample\_Ethereum != 0,][,7])  
  
data\_Cardano = cluster\_data[cluster\_data$Currency == 'Cardano',]  
sample\_Cardano = srswor(50,nrow(data\_Cardano))  
cluster\_sampling\_means[11] = mean(data\_Cardano[sample\_Cardano != 0,][,7])  
  
data\_Bitcoin\_sv = cluster\_data[cluster\_data$Currency == 'Bitcoin-sv',]  
sample\_Bitcoin\_sv = srswor(50,nrow(data\_Bitcoin\_sv))  
cluster\_sampling\_means[12] = mean(data\_Bitcoin\_sv[sample\_Bitcoin\_sv != 0,][,7])  
  
sampling\_DF$cluster\_sampling\_means = cluster\_sampling\_means  
  
# Population  
  
population\_means = c()  
count = 1  
  
for (i in Currency\_list){  
 population\_means[count] = mean(get(i)[,5])  
 count = count + 1  
}  
  
sampling\_DF$population\_means = population\_means  
  
# Plot  
  
plot\_ly(sampling\_DF, x = Currency\_list, y = ~random\_sampling\_means, type = 'bar',   
 name = 'Random Sampling Means',texttemplate = '%{y:.2s}', textposition = 'outside') %>%   
 add\_trace(y = ~systematic\_sampling\_means, name = 'Systematic Means') %>%  
 add\_trace(y = ~stratified\_sampling\_means, name = 'Stratified Means') %>%  
 add\_trace(y = ~cluster\_sampling\_means, name = 'Cluster Means') %>%  
 add\_trace(y = ~population\_means, name = 'Population Means') %>%  
 layout(title = 'Means of Volume by Sampling Methods',xaxis = list(tickangle = 45,  
 tickfont = list(size = 20)),  
 yaxis = list(title = 'Volume Means',tickfont = list(size = 20)))



# Calculate sampling error  
  
sampling\_DF$percentage\_error\_rs = round((abs(sampling\_DF$population\_means -   
 sampling\_DF$random\_sampling\_means) / sampling\_DF$random\_sampling\_means),4) \* 100  
  
sampling\_DF$percentage\_error\_systematic = round((abs(sampling\_DF$population\_means -   
 sampling\_DF$systematic\_sampling\_means) / sampling\_DF$systematic\_sampling\_means),4) \* 100  
  
sampling\_DF$percentage\_error\_stratified = round((abs(sampling\_DF$population\_means -   
 sampling\_DF$stratified\_sampling\_means) / sampling\_DF$stratified\_sampling\_means),4) \* 100  
  
sampling\_DF$percentage\_error\_cluster = round((abs(sampling\_DF$population\_means -   
 sampling\_DF$cluster\_sampling\_means) / sampling\_DF$cluster\_sampling\_means),4) \* 100  
  
# Plot Sampling errors  
  
plot\_ly(sampling\_DF, x = Currency\_list, y = ~percentage\_error\_rs, type = 'bar',   
 name = 'Random Sampling Error %') %>%   
 add\_trace(y = ~percentage\_error\_systematic, name = 'Systematic Means Error %') %>%  
 add\_trace(y = ~percentage\_error\_stratified, name = 'Stratified Means Error %') %>%  
 add\_trace(y = ~percentage\_error\_cluster, name = 'Cluster Means Error %') %>%  
 layout(title = 'Error % of Sampling Methods',xaxis = list(tickangle = 45,tickfont = list(size = 20)),  
 yaxis = list(title = 'Percentage',tickfont = list(size = 20)))



References

En.wikipedia.org. 2020. *Bitcoin*. [online] Available at: <https://en.wikipedia.org/wiki/Bitcoin> [Accessed 25 June 2020].En.wikipedia.org. 2020. *Cryptocurrency*. [online] Available at: <https://en.wikipedia.org/wiki/Cryptocurrency> [Accessed 25 June 2020].

En.wikipedia.org. 2020. *Cryptocurrency*. [online] Available at: <https://en.wikipedia.org/wiki/Cryptocurrency> [Accessed 25 June 2020].

2020. [online] Available at: <https://www.quora.com/> [Accessed 25 June 2020].

Investopedia. 2020. *Cryptocurrency*. [online] Available at: <https://www.investopedia.com/terms/c/cryptocurrency.asp> [Accessed 25 June 2020].