CYO Cryptocurrency Report

Donal Medina

August 23, 2021

Introduction

For this project we will predict the close price of cryptocurrency historical prices data set(obtained from kaggle site), we will analyse de volatility of every crytocurrency and select one(Bitcoin) to apply machine learning algorithms.
We will develop a k-nearest neighbors (KNN) and random forest model and compare the performance using RMSE (residual mean squared error) and MAPE (Mean Absolute Percentage Error) in the test set.

Analysis

Install and load Packages

```
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: tidyverse

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

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.1.0     v dplyr   1.0.5
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: caret

## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```
if(!require(data.table)) install.packages("data.table", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: data.table

##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
## The following object is masked from 'package:purrr':
##
##     transpose
```

```
if(!require(randomForest)) install.packages("randomForest", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: randomForest
## randomForest 4.6-14
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
## The following object is masked from 'package:dplyr':
##
##     combine
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(tidyverse)
library(caret)
library(data.table)
library(randomForest)
```

Data Loading

Create crypto_hist set. Cryptocurrency Historical Prices can found: https://www.kaggle.com/sudalairajkumar/cryptocurrencypricehistory The Dataset can be download on my GitHub repository : https://github.com/dmedina-lab/cryptocurrency/raw/main/cryptocurrency_price.zip

```
# Note: this process could take a couple of minutes
dlf <- tempfile()
download.file("https://github.com/dmedina-lab/cryptocurrency/raw/main/cryptocurrency_price.zip", dlf)

unzip(dlf, exdir = "crypto-price-hist")

crypto_hist <-
  list.files(path = "crypto-price-hist", pattern = "*.csv", full.names = T) %>%
  map_df(~read_csv(.))
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
```

```
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
```

```
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
```

```
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
```

```
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
```

```
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
```

```
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
##
##
## -- Column specification --------------------------------------------------------
## cols(
##   SNo = col_double(),
##   Name = col_character(),
```

```
##   Symbol = col_character(),
##   Date = col_datetime(format = ""),
##   High = col_double(),
##   Low = col_double(),
##   Open = col_double(),
##   Close = col_double(),
##   Volume = col_double(),
##   Marketcap = col_double()
## )
```

```r
rm(dlf)
```

Data Exploration

First we check for any NA value.

```r
anyNA(crypto_hist)
```

```
## [1] FALSE
```

General overview of dataset:

```r
str(crypto_hist)
```

```
## spec_tbl_df[,10] [37,082 x 10] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ SNo      : num [1:37082] 1 2 3 4 5 6 7 8 9 10 ...
##  $ Name     : chr [1:37082] "Aave" "Aave" "Aave" "Aave" ...
##  $ Symbol   : chr [1:37082] "AAVE" "AAVE" "AAVE" "AAVE" ...
##  $ Date     : POSIXct[1:37082], format: "2020-10-05 23:59:59" "2020-10-06 23:59:59" ...
##  $ High     : num [1:37082] 55.1 53.4 42.4 44.9 47.6 ...
##  $ Low      : num [1:37082] 49.8 40.7 36 36.7 43.3 ...
##  $ Open     : num [1:37082] 52.7 53.3 42.4 39.9 43.8 ...
##  $ Close    : num [1:37082] 53.2 42.4 40.1 43.8 46.8 ...
##  $ Volume   : num [1:37082] 0 583091 682834 1658817 815538 ...
##  $ Marketcap: num [1:37082] 8.91e+07 7.10e+07 6.71e+07 2.20e+08 2.36e+08 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   SNo = col_double(),
##   ..   Name = col_character(),
##   ..   Symbol = col_character(),
##   ..   Date = col_datetime(format = ""),
##   ..   High = col_double(),
##   ..   Low = col_double(),
##   ..   Open = col_double(),
##   ..   Close = col_double(),
##   ..   Volume = col_double(),
##   ..   Marketcap = col_double()
##   .. )
```

```r
head(crypto_hist) %>% knitr::kable()
```

SNo	Name	Symbol	Date	High	Low	Open	Close	Volume	Marketcap
1	Aave	AAVE	2020-10-05 23:59:59	55.11236	49.78790	52.67504	53.21924	0.0	89128129
2	Aave	AAVE	2020-10-06 23:59:59	53.40227	40.73458	53.29197	42.40160	583091.5	71011441

SNo	Name	Symbol	Date	High	Low	Open	Close	Volume	Marketcap
3	Aave	AAVE	2020-10-07 23:59:59	42.40831	35.97069	42.39995	40.08398	682834.2	67130037
4	Aave	AAVE	2020-10-08 23:59:59	44.90251	36.69606	39.88526	43.76446	1658816.9	220265142
5	Aave	AAVE	2020-10-09 23:59:59	47.56953	43.29178	43.76446	46.81774	815537.7	235632208
6	Aave	AAVE	2020-10-10 23:59:59	51.40565	46.70333	46.81815	49.13372	1074627.0	247288429

After loading the data set we start by looking at the data structure and type we can see that there is 37,082 observations and 10 variables.
SNo: Incremental row number for each coin.
Name: Name of cryptcurrency.
Date : date of observation.
Open : Opening price on the given day.
High : Highest price on the given day.
Low : Lowest price on the given day.
Close : Closing price on the given day.
Volume : Volume of transactions on the given day.
Market Cap : Market capitalization in USD.

```r
summary(crypto_hist) %>% knitr::kable()
```

	SNo	Name	Symbol	Date	High	Low	Open	Close	Volume	Marketcap
	Min. : 1	Length:37082	Length:37082	Min. :2013-04-29 23:59:59	Min. : 0.00	Min. : 0.00	Min. : 0.00	Min. : 0.00	Min. :0.000e+00	Min. :0.000e+00
	1st Qu.: 420	Class :character	Class :character	1st Qu.:2017-03-05 23:59:59	1st Qu.: 0.08	1st Qu.: 0.07	1st Qu.: 0.07	1st Qu.: 0.07	1st Qu.:4.937e+06	1st Qu.:2.396e+08
	Median : 910	Mode :character	Mode :character	Median :2019-01-09 23:59:59	Median : 1.01	Median : 1.00	Median : 1.00	Median : 1.00	Median :8.513e+07	Median :1.405e+09
	Mean :1057	NA	NA	Mean :2018-08-16 07:12:30	Mean : 1016.06	Mean : 952.99	Mean : 985.32	Mean : 987.12	Mean :3.023e+09	Mean :1.543e+10
	3rd Qu.:1585	NA	NA	3rd Qu.:2020-05-13 23:59:59	3rd Qu.: 31.92	3rd Qu.: 29.00	3rd Qu.: 30.46	3rd Qu.: 30.51	3rd Qu.:9.388e+08	3rd Qu.:5.159e+09
	Max. :2991	NA	NA	Max. :2021-07-06 23:59:59	Max. :64863.10	Max. :62208.96	Max. :63523.75	Max. :63503.46	Max. :3.510e+11	Max. :1.186e+12

We can see the date range date from 2013-04-29 to 2021-07-06, the max Close price it is 63,503.46 usd and the average 987.12 usd.

```r
crypto_hist %>% group_by(Name) %>%
  summarize(count = n(), date_from= min(Date), date_to= max(Date) ) %>%
  arrange(desc(count)) %>% knitr::kable()
```

Name	count	date_from	date_to
Bitcoin	2991	2013-04-29 23:59:59	2021-07-06 23:59:59
Litecoin	2991	2013-04-29 23:59:59	2021-07-06 23:59:59
XRP	2893	2013-08-05 23:59:59	2021-07-06 23:59:59
Dogecoin	2760	2013-12-16 23:59:59	2021-07-06 23:59:59
Monero	2602	2014-05-22 23:59:59	2021-07-06 23:59:59
Stellar	2527	2014-08-06 23:59:59	2021-07-06 23:59:59
Tether	2318	2015-02-26 23:59:59	2021-07-06 23:59:59
NEM	2288	2015-04-02 23:59:59	2021-07-06 23:59:59
Ethereum	2160	2015-08-08 23:59:59	2021-07-06 23:59:59
IOTA	1484	2017-06-14 23:59:59	2021-07-06 23:59:59
EOS	1466	2017-07-02 23:59:59	2021-07-06 23:59:59
Binance Coin	1442	2017-07-26 23:59:59	2021-07-06 23:59:59
TRON	1392	2017-09-14 23:59:59	2021-07-06 23:59:59
Chainlink	1385	2017-09-21 23:59:59	2021-07-06 23:59:59
Cardano	1374	2017-10-02 23:59:59	2021-07-06 23:59:59
USD Coin	1002	2018-10-09 23:59:59	2021-07-06 23:59:59
Crypto.com Coin	935	2018-12-15 23:59:59	2021-07-06 23:59:59
Wrapped Bitcoin	888	2019-01-31 23:59:59	2021-07-06 23:59:59
Cosmos	845	2019-03-15 23:59:59	2021-07-06 23:59:59
Solana	452	2020-04-11 23:59:59	2021-07-06 23:59:59
Polkadot	320	2020-08-21 23:59:59	2021-07-06 23:59:59
Uniswap	292	2020-09-18 23:59:59	2021-07-06 23:59:59
Aave	275	2020-10-05 23:59:59	2021-07-06 23:59:59

We can see there are 23 coins. Bitcoin, Litecoin, XRP, Dogecoin are the most old, since 2013.

Top 5 newest cryptocurrency

```
crypto_hist %>% group_by(Name) %>%
  summarize(date_from= min(Date), date_to= max(Date) ) %>%
  arrange(desc(date_from)) %>%
  head(5) %>% knitr::kable()
```

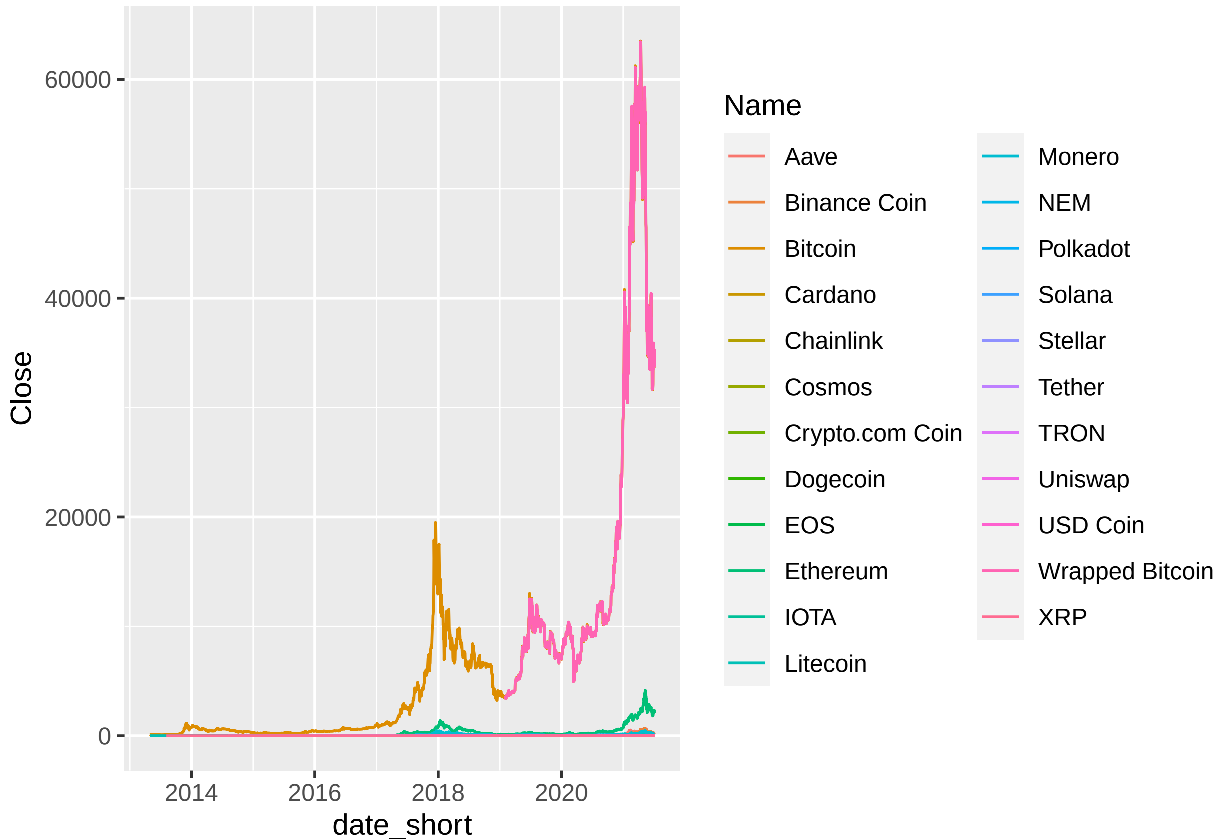
Name	date_from	date_to
Aave	2020-10-05 23:59:59	2021-07-06 23:59:59
Uniswap	2020-09-18 23:59:59	2021-07-06 23:59:59
Polkadot	2020-08-21 23:59:59	2021-07-06 23:59:59
Solana	2020-04-11 23:59:59	2021-07-06 23:59:59
Cosmos	2019-03-15 23:59:59	2021-07-06 23:59:59

Aave, Uniswap, Polkadot, Solana adn Cosmos are the newest coin, since 2019.

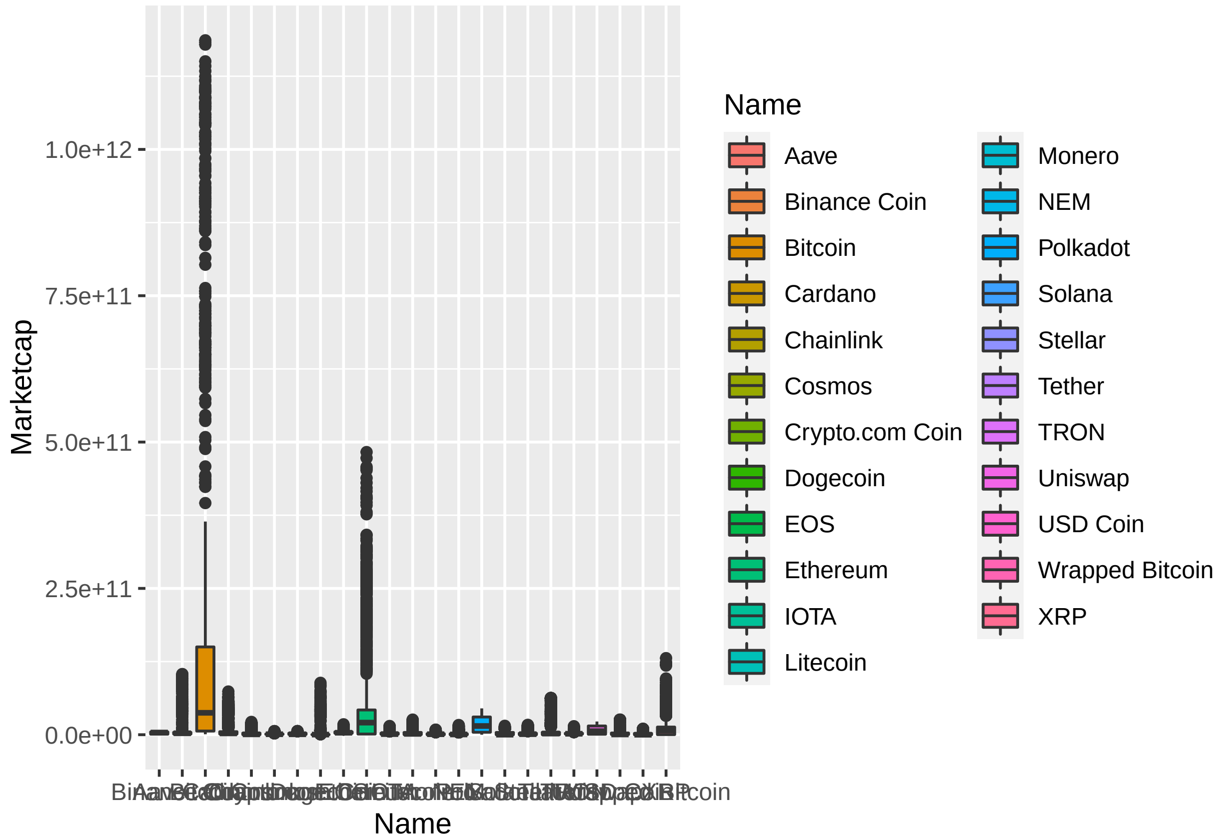
Transformation of date formats to short date, year, month, day and price market relationship

```
crypto_hist_p <- crypto_hist %>%
  mutate(year = as.integer(format(Date, format="%Y"))
         , month = as.integer(format(Date, format="%m"))
         , day = as.integer(format(Date, format="%d"))
         , date_short = as.Date(format(Date, format="%Y-%m-%d"))
         , Price_Market = as.numeric(Close / Marketcap)
         )
```

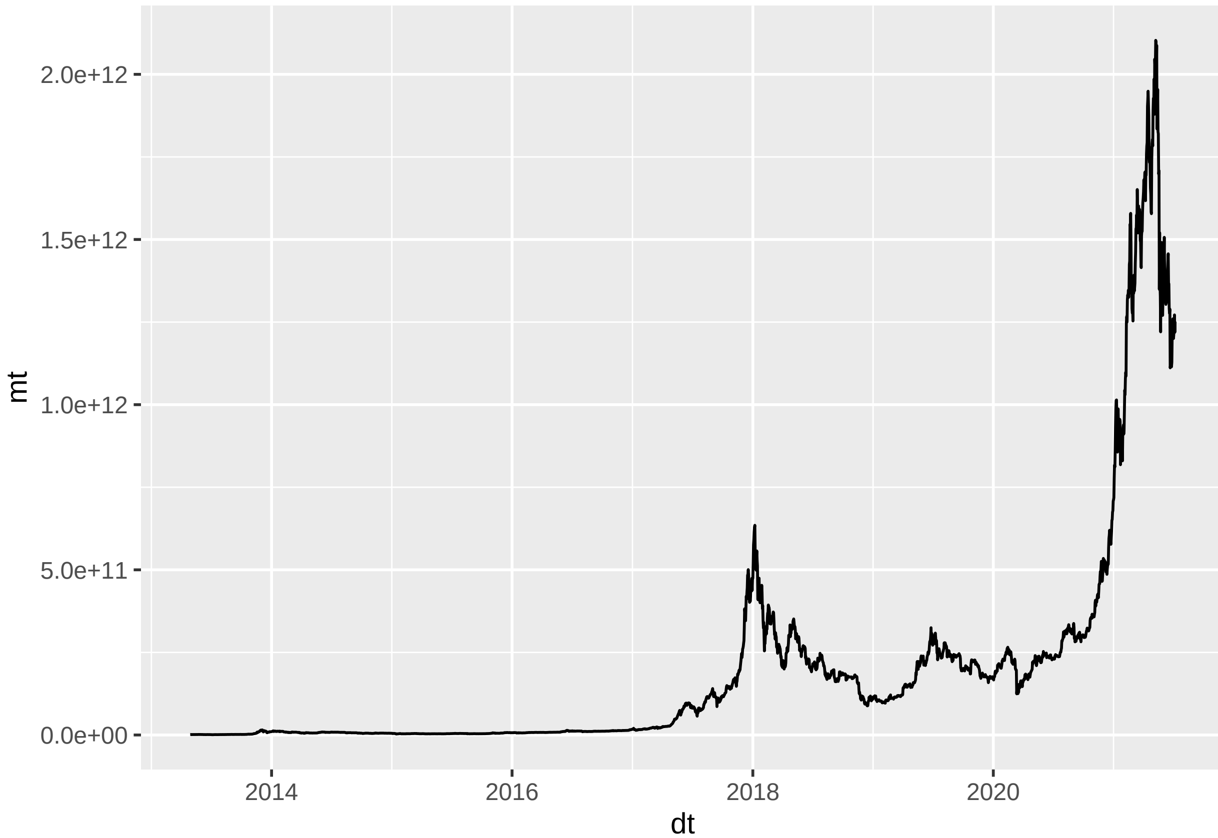
Trend historical Close price for each Coin Name.



Boxplot historical Close price for each Coin Name.



Trend hisotorical Market Capitalization.



Correlation between Close price and Market Capitalization.

```r
crypto_hist_p %>% summarize(r = cor(Close, Marketcap)) %>% pull(r)
```

```
## [1] 0.6884597
```

It is about 68.8%.

Correlation between Close price and Volume.

```r
crypto_hist_p %>% summarize(r = cor(Close, Volume)) %>% pull(r)
```

```
## [1] 0.2857774
```

It is very low, about 28.5%.

Now we will calculate the volatility of the cryptocurrencies en our dataset. Volatility is best defined as the 30-day standard deviation of daily log returns annualized. With this in mind, high volatility is at 100% or above, medium volatility is between 50% and 100%, while low volatility is below 50%.

Top 10 least Volatility (Anual).

```r
crypto_hist_p %>% group_by(Name) %>%
  summarize(avg_close = mean(Close),
            sd_close = sd(Close),
            volatility = sqrt(365) * sd(diff(log(Close))) * 100) %>%
  arrange(volatility) %>%
  head(10) %>% knitr::kable()
```

Name	avg_close	sd_close	volatility
USD Coin	1.003791e+00	6.821700e-03	8.763689
Tether	1.000696e+00	1.495060e-02	35.107711
Bitcoin	6.711290e+03	1.129814e+04	81.689556
Wrapped Bitcoin	1.708657e+04	1.579849e+04	82.815170
Ethereum	3.839107e+02	6.010788e+02	118.639144
Litecoin	4.927901e+01	6.324046e+01	121.786965
Monero	7.413477e+01	9.118056e+01	130.126362
Crypto.com Coin	8.191250e-02	5.369330e-02	135.306466
XRP	2.347898e-01	3.386292e-01	139.816135
IOTA	7.293697e-01	7.976106e-01	140.114964

Top 10 Most Volatility (Anual).

```
crypto_hist_p %>% group_by(Name) %>%
  summarize(avg_close = mean(Close),
            sd_close = sd(Close),
            volatility = sqrt(365) * sd(diff(log(Close))) * 100) %>%
  arrange(desc(volatility))%>%
  top_n(10)%>% knitr::kable()
```

```
## Selecting by volatility
```

Name	avg_close	sd_close	volatility
Solana	10.4713883	14.1144436	177.9985
Uniswap	17.0772562	12.7757893	169.1999
Aave	255.5258454	161.6499166	164.7774
TRON	0.0325849	0.0275621	161.6429
Polkadot	18.1430801	13.7357072	160.6633
Dogecoin	0.0137626	0.0625587	160.0191
Chainlink	6.3085826	9.8985171	150.6476
NEM	0.1246622	0.1979898	150.1648
Binance Coin	52.2503075	115.3909166	143.1800
Cardano	0.2563126	0.4096914	142.5422

We observe that Bitcoin es the third least volatility coin, The first and second least volatility are USD COin and Theter respectible, Both based on dollar.

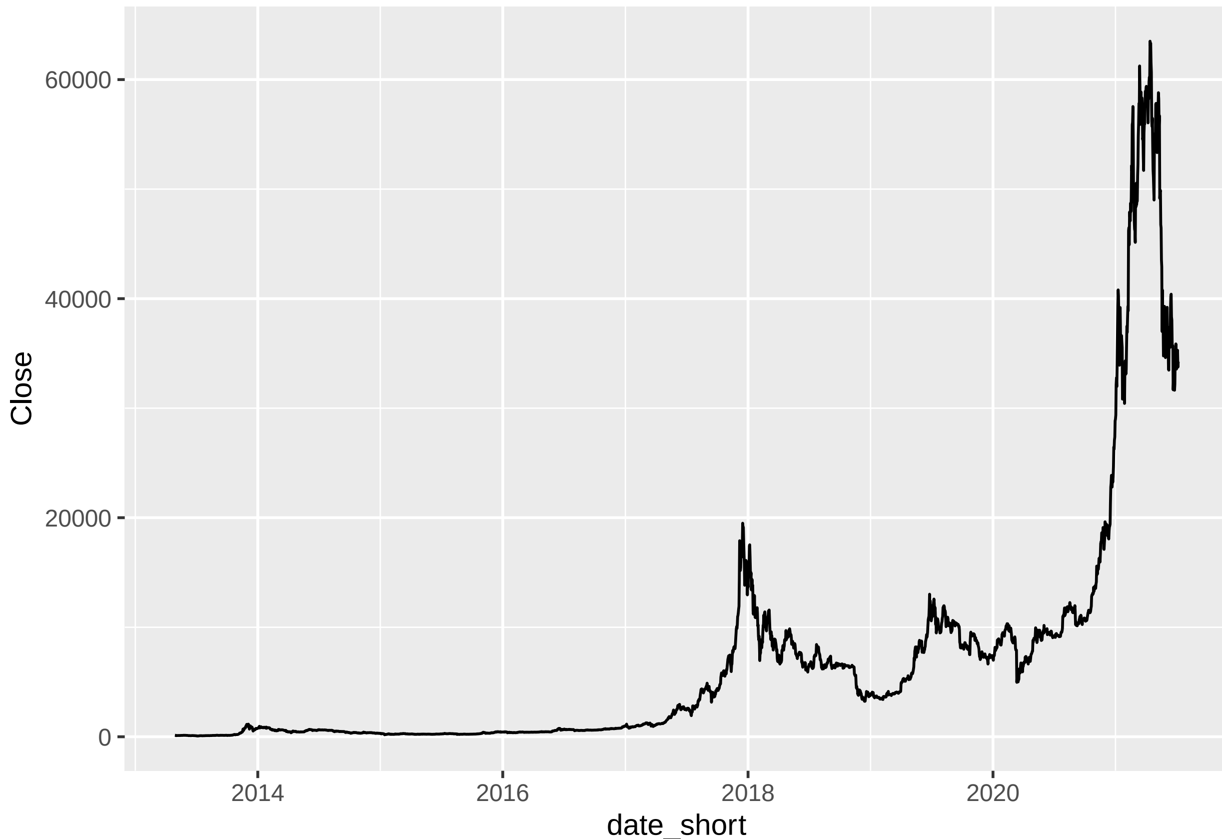
We will use Bitcoin in ours models and We will show the volatility for year.

```
crypto_hist_p %>% filter(Name=="Bitcoin") %>%group_by(year) %>%
  summarize(avg_close = mean(Close),
            sd_close = sd(Close),
            volatility = sqrt(365) * sd(diff(log(Close))) * 100) %>%
  arrange(volatility) %>% knitr::kable()
```

year	avg_close	sd_close	volatility
2016	568.4924	139.25573	48.29731
2019	7395.2463	2638.63505	67.54509
2015	272.4534	59.33786	70.28227
2014	527.2365	148.63774	75.27859
2020	11116.3781	4305.85884	76.66323

year	avg_close	sd_close	volatility
2018	7572.2989	2455.45547	82.00037
2021	45539.2759	10061.70811	92.97636
2017	4006.0336	4053.19220	94.27888
2013	257.9735	274.73719	128.39296

Trend of Bitcoin.



Data Wrangling

We will create a subset filtered by Bitcoin.

```
btc <- crypto_hist_p %>% filter(Name == "Bitcoin")
```

Set seed.

```
set.seed(1, sample.kind="Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

Split the btc data set to train set(80%) and test set(20%).

```
test_index_btc <- createDataPartition(y = btc$Close, times = 1, p = 0.2, list = FALSE)
train_set_btc <- btc[-test_index_btc,]
test_set_btc <- btc[test_index_btc,]
```

Results

MAPE (Mean Absolute Percentage Error)

```
MAPE <- function(actual_price,pred_price){
  mape <- mean(abs((actual_price - pred_price)/actual_price))*100
  return (mape)
}
```

RMSE (residual mean squared error) calculation funtion

```
RMSE <- function(true_ratings, predicted_ratings){
  sqrt(mean((true_ratings - predicted_ratings)^2, na.rm = TRUE))
}
```

First model : Random Forest

Fit the model.

```
set.seed(1, sample.kind="Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```
fit_rf <- randomForest(Close ~ Marketcap+Volume+Open+Low+High+date_short+year+month+day
                       , data=train_set_btc)

pred_rf <- predict(fit_rf, test_set_btc)
```

Evaluate the model with MAPE.

```
MAPE(test_set_btc$Close, pred_rf)
```

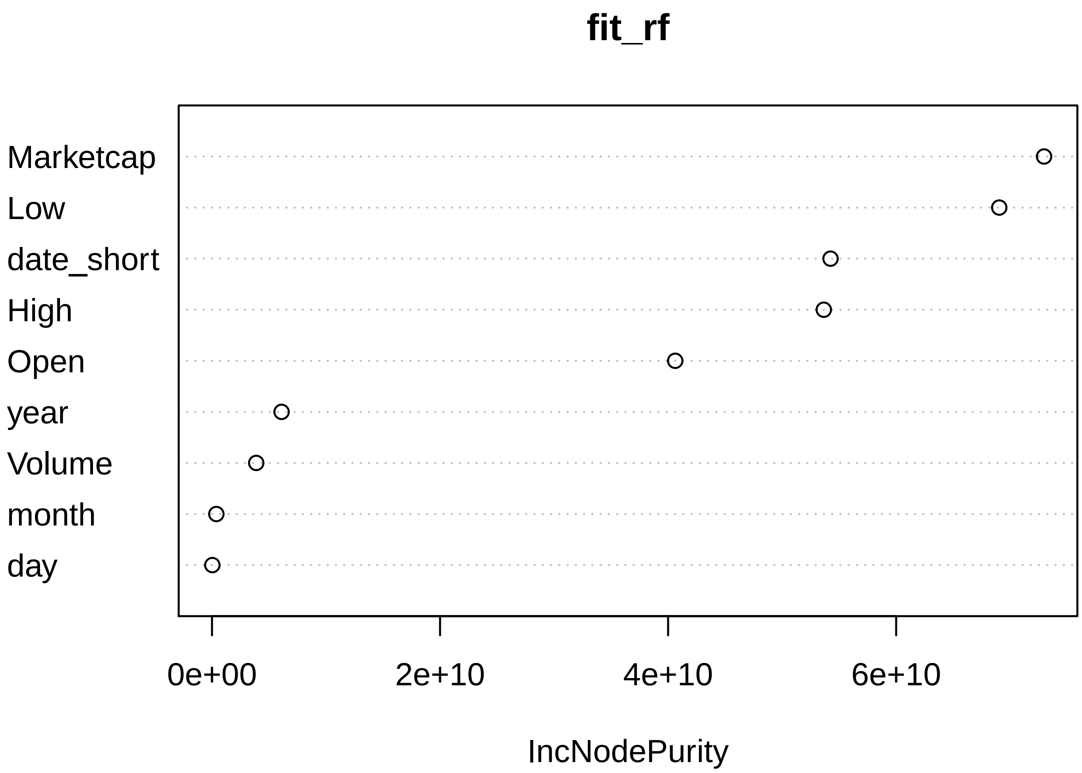
```
## [1] 1.18496
```

Now with RMSE.

```
RMSE(test_set_btc$Close, pred_rf)
```

```
## [1] 251.2234
```

Plot the importance variable.



The most important variables are Marketcap, Low, High, date_short and Open.

	IncNodePurity
Marketcap	72971061252
Low	69041883329
date_short	54242381338
High	53653603293
Open	40623726804
year	6102999143
Volume	3879565632
month	382907537
day	32171016

To avoid overtraing we will use only the most important variables to build the model.

```
rf_revised <- randomForest(Close ~ Marketcap+Low+High+date_short+Open
                           , data = train_set_btc)
print(rf_revised)
```

```
##
## Call:
##  randomForest(formula = Close ~ Marketcap + Low + High + date_short +      Open, data = train_set_btc
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 1
##
```

```
##           Mean of squared residuals: 73505.74
##                     % Var explained: 99.94
```

```r
pred_rf_r <- predict(rf_revised, test_set_btc)
```

aguain evaluate the model with MAPE.

```r
MAPE(test_set_btc$Close, pred_rf_r)
```

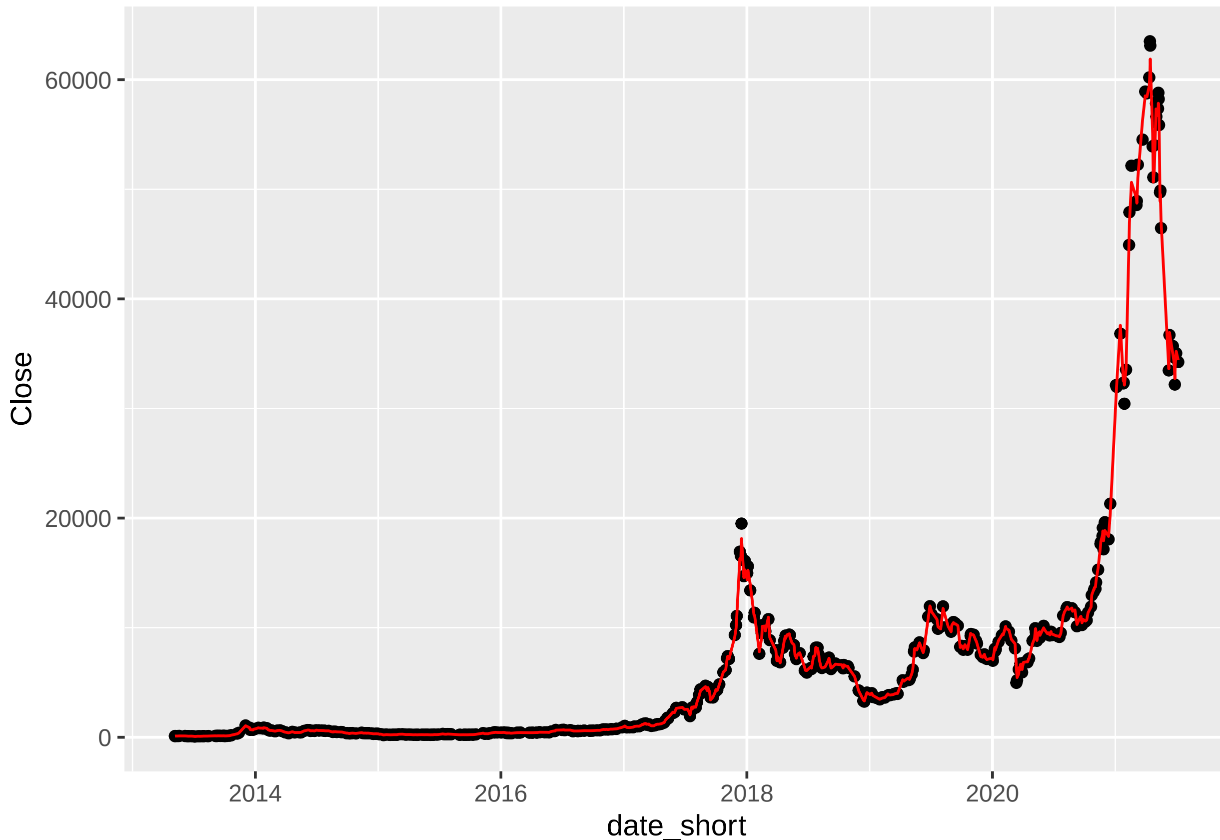
```
## [1] 1.269292
```

Now with RMSE

```r
RMSE(test_set_btc$Close, pred_rf_r)
```

```
## [1] 276.9303
```

Plot the test results:



Second model : k-nearest neighbors (KNN)

Fit the model.

```r
set.seed(1, sample.kind="Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```r
train_knn <- train(Close ~ Marketcap+Low+High+date_short+Open,
                   method = "knn",
                   data = train_set_btc,
```

```
                    tuneGrid = data.frame(k = seq(1, 100, 2)))
ped_knn <- predict(train_knn, test_set_btc)
```

Evaluate the model with MAPE.

```
MAPE(test_set_btc$Close, ped_knn)
```

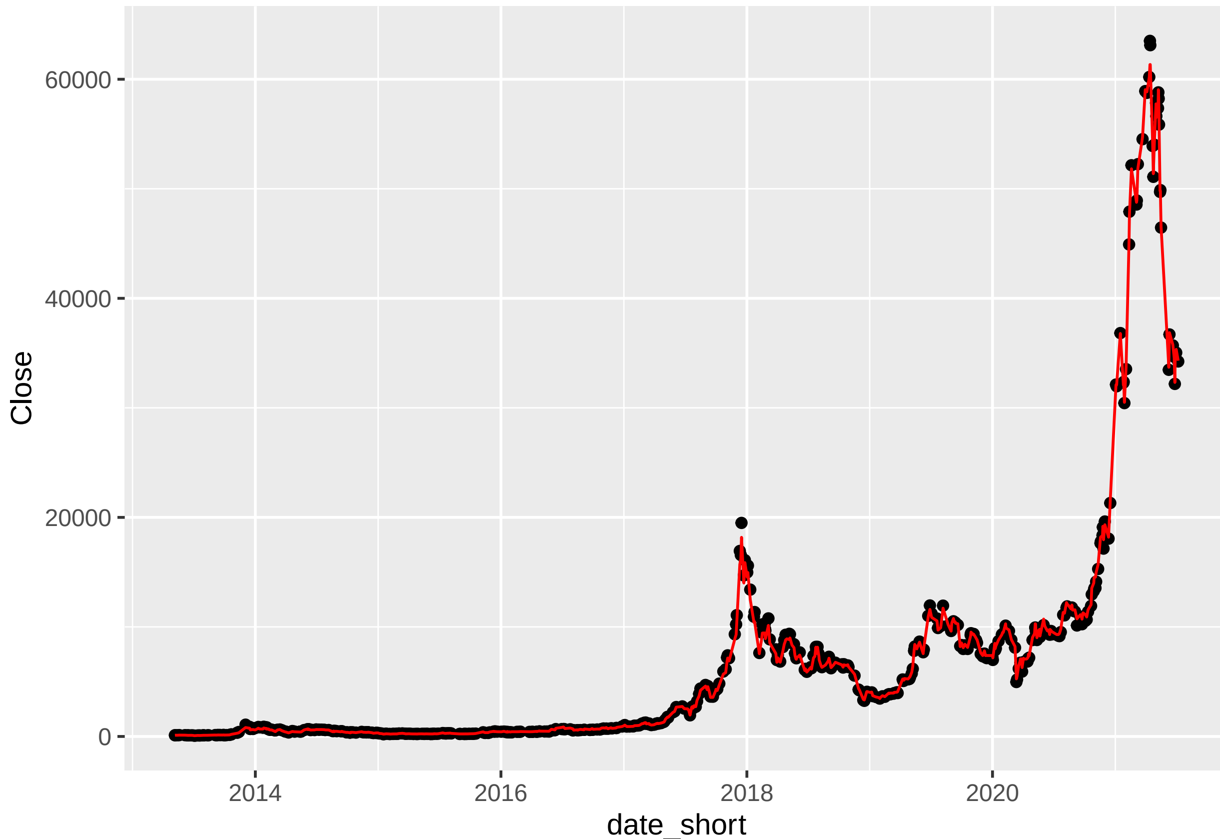
```
## [1] 3.270416
```

Now with RMSE.

```
RMSE(test_set_btc$Close, ped_knn)
```

```
## [1] 246.3319
```

Plot the test results:



Now We will create a tibble for the MAPE and RMSE results from each method to compare.

```
tibble(method = c("KNN","Random Forest")
       , MAPE = c(MAPE(test_set_btc$Close, ped_knn), MAPE(test_set_btc$Close, pred_rf_r))
       , RMSE = c(RMSE(test_set_btc$Close, ped_knn),RMSE(test_set_btc$Close, pred_rf_r))) %>%
  knitr::kable()
```

method	MAPE	RMSE
KNN	3.270416	246.3319
Random Forest	1.269292	276.9303

Conclusion

I have developed a k-nearest neighbors (KNN) and random forest model, the best MAPE was given by the random forest model and similarty results with RMSE.
The variability of the criptocurrurncy its mostly high, in a future we can consider tunning this algorithms and use deep learning techniques like Multilayer Perceptrons (MLPs), Convolutional Neural Networks (CNNs) and Long Short-Term Memory Networks (LSTMs).
Also consider other more complex aspects such as twitter data that could influence the trending.