Crypto

BAKENGA William

2024-10-16

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(ggthemes)
library(caret)
```

```
## Le chargement a nécessité le package : lattice
##
## Attachement du package : 'caret'
##
## L'objet suivant est masqué depuis 'package:purrr':
##
##     lift
```

```r
library(timeDate)
library(ggrepel)
library(readr)
library(askpass)
library(class)
library(forcats)
library(lubridate)
library(stringr)
library(corrplot)
```

```
## corrplot 0.95 loaded
```

```r
library(extrafont)
```

```
## Registering fonts with R
```

```r
library(systemfonts)
#fonts()
#loadfonts()
```

```r
data = read.csv('C://Users//HP//Documents//william_project_solo//bi-perso//marketprice.csv')
head(data)
```

```
##   X            timestamp    price  volume_24h   market_cap        name
## 1 0 2023-10-19T00:00:00Z 28497.38 10408090970 556216437429 btc-bitcoin
## 2 1 2023-10-20T00:00:00Z 29436.80 17416159256 574576473068 btc-bitcoin
## 3 2 2023-10-21T00:00:00Z 29805.58 13954479389 581800798688 btc-bitcoin
## 4 3 2023-10-22T00:00:00Z 29923.96 11141548512 584139969263 btc-bitcoin
## 5 4 2023-10-23T00:00:00Z 30903.85 16559018229 603298245173 btc-bitcoin
## 6 5 2023-10-24T00:00:00Z 34084.20 39316464490 665414809955 btc-bitcoin
```

```r
dim(data)
```

```
## [1] 18072     6
```

```r
colnames(data)
```

```
## [1] "X"          "timestamp"  "price"      "volume_24h" "market_cap"
## [6] "name"
```

```r
data$timestamp = str_replace(data$timestamp, "T00:00:00Z", "")
data$timestamp = as.Date(data$timestamp, tryFormats = "%Y-%m-%d")
head(data)
```

```
##   X  timestamp    price  volume_24h   market_cap        name
## 1 0 2023-10-19 28497.38 10408090970 556216437429 btc-bitcoin
## 2 1 2023-10-20 29436.80 17416159256 574576473068 btc-bitcoin
## 3 2 2023-10-21 29805.58 13954479389 581800798688 btc-bitcoin
## 4 3 2023-10-22 29923.96 11141548512 584139969263 btc-bitcoin
## 5 4 2023-10-23 30903.85 16559018229 603298245173 btc-bitcoin
## 6 5 2023-10-24 34084.20 39316464490 665414809955 btc-bitcoin
```

The greatest prices

```r
data %>%
  dplyr::select(name, price) %>%
  group_by(name) %>%
  summarise(prix_moyen = round(mean(price), 3)) %>%
  ungroup() %>%
  mutate(rang = as.integer(rank(desc(prix_moyen)))) %>%
  dplyr::filter(rang <= 25) %>%
  arrange(rang)
```

```
## # A tibble: 25 x 3
##    name                      prix_moyen  rang
##    <chr>                          <dbl> <int>
##  1 btcb-binance-bitcoin          55740.     1
##  2 btc-bitcoin                   55733.     2
##  3 wbtc-wrapped-bitcoin          55699.     3
##  4 eth-ethereum                   2807.     4
##  5 weth-weth                      2807.     5
##  6 steth-lido-staked-ether        2805.     6
##  7 bnb-binance-coin                460.     7
##  8 tao-bittensor                   388.     8
##  9 bch-bitcoin-cash                352.     9
## 10 xmr-monero                      153.    10
```

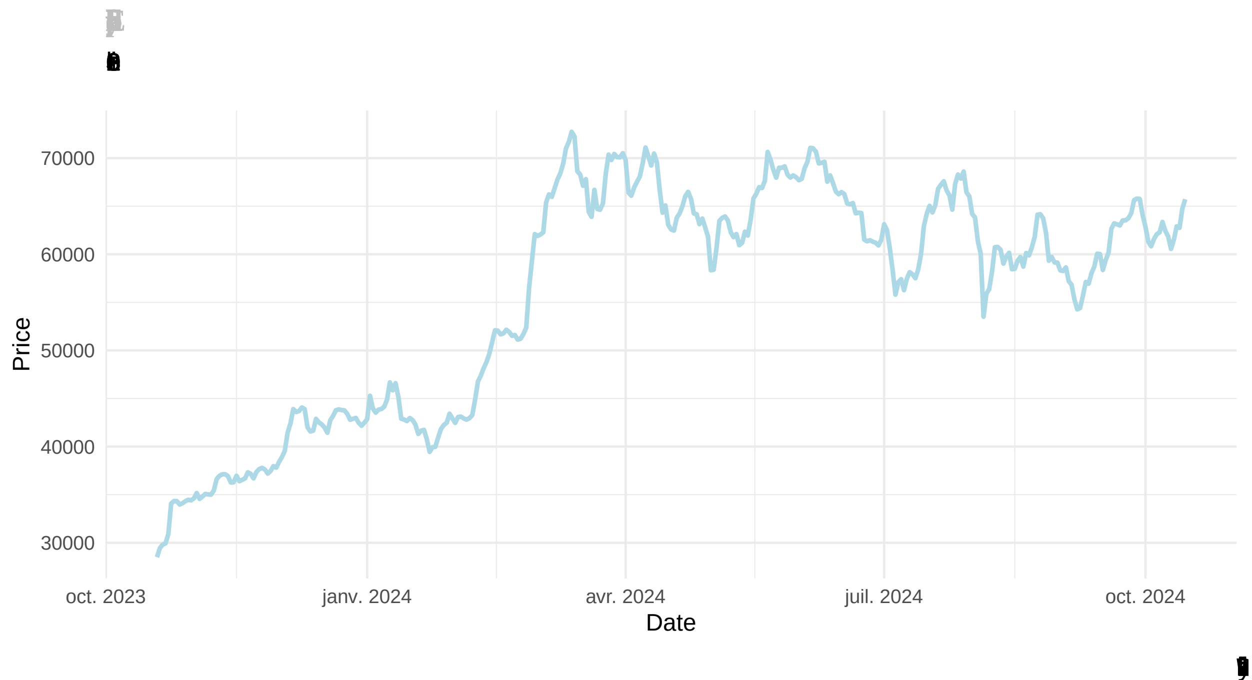
```
## # i 15 more rows
```

We have some interesting result but some data wrangling needs to be done at this point since BTC and ETH have each multiple values;

```r
labels = read.csv('C://Users//HP//Documents//william_project_solo//bi-perso//cryptocurrencies.csv')
head(labels)
```

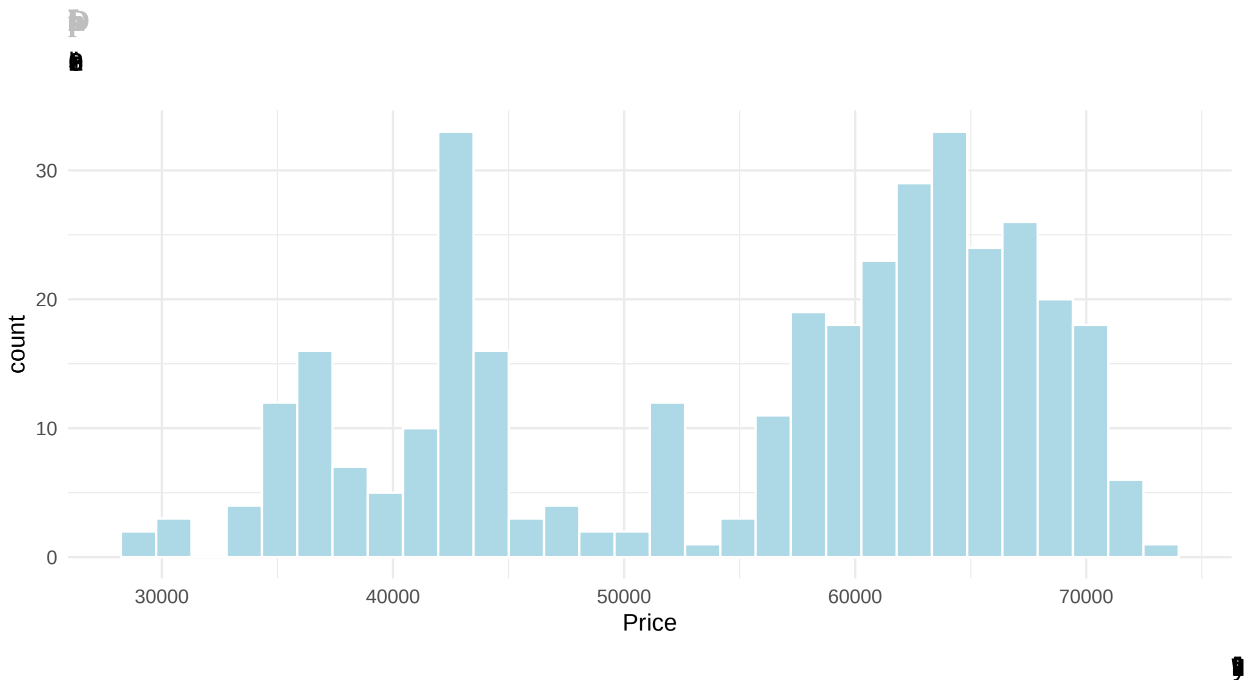
```
##   X                id         name symbol rank is_new is_active  type
## 1 0       btc-bitcoin      Bitcoin    BTC    1  False      True  coin
## 2 1      eth-ethereum     Ethereum    ETH    2  False      True  coin
## 3 2       usdt-tether       Tether   USDT    3  False      True token
## 4 3 bnb-binance-coin Binance Coin    BNB    4  False      True  coin
## 5 4        sol-solana       Solana    SOL    5  False      True  coin
## 6 5     usdc-usd-coin         USDC   USDC    6  False      True token
```

```r
data %>% dplyr::select(price, timestamp, name) %>%
  dplyr::filter(name == 'btc-bitcoin') %>%
  ggplot(aes(x = timestamp, y = price)) +
  geom_line(linewidth = 1
            , col = 'lightblue'
            ) +
  theme_minimal() +
  labs(title = "Evolution of the bitcoin price per day"
      , subtitle = "Since October 2023\n"
      , caption = "Done by William Bak"
      , x = "Date"
      , y = "Price"
      ) +
  theme(plot.title = element_text(family = "Times New Roman"
                                  , vjust = .5
                                  , face = "bold"
                                  , color = "grey"
                                  , size = 14
                                  )
        , plot.subtitle = element_text(family = 'Calibri Light'
                                       , size = 12
                                       )
        , plot.caption = element_text(family = 'Calibri Light'
                                       , size = 12
                                       )
        )
```




```r
data %>% dplyr::select(price, timestamp, name) %>%
  dplyr::filter(name == 'btc-bitcoin') %>%
  ggplot(aes(price)) +
  geom_histogram(fill = 'lightblue'
                 , col = "white"
                 ) +
  theme_minimal() +
  labs(title = "Distribution of the bitcoin price"
       , subtitle = "Since October 2023\n"
       , caption = "Done by William Bak"
       , x = "Price"
       ) +
  theme(plot.title = element_text(family = "Times New Roman"
                                  , vjust = .5
                                  , face = "bold"
                                  , color = "grey"
                                  , size = 14
                                  )
        , plot.subtitle = element_text(family = 'Calibri Light'
                                       , size = 12
                                       )
        , plot.caption = element_text(family = 'Calibri Light'
                                      , size = 12
                                      )
        )
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
tbl_ = aggregate(data$price, by = list(data$name), FUN=mean)
tbl_$x = round(tbl_$x, 2)
tbl_
```

```
##                         Group.1        x
## 1                      aave-new   105.57
## 2                   ada-cardano     0.46
## 3                    apt-aptos     8.67
## 4                avax-avalanche    31.74
## 5              bch-bitcoin-cash   351.73
## 6              bnb-binance-coin   460.23
## 7                   btc-bitcoin 55732.92
## 8          btcb-binance-bitcoin 55739.86
## 9             cro-cryptocom-chain     0.10
## 10                      dai-dai     1.00
## 11                doge-dogecoin     0.12
## 12                 dot-polkadot     6.44
## 13         etc-ethereum-classic    23.76
## 14                 eth-ethereum  2807.35
## 15      fdusd-first-digital-usd     1.00
## 16                     fetch-ai     1.34
## 17                 fil-filecoin     5.41
## 18                   ftm-fantom     0.54
## 19        hbar-hedera-hashgraph     0.08
## 20        icp-internet-computer    10.00
## 21              imx-immutable-x     1.90
## 22                    kas-kaspa     0.14
## 23                leo-leo-token     5.14
## 24              link-chainlink    14.59
## 25                 ltc-litecoin    74.10
## 26                  mnt-mantle     0.75
```

```
## 27           near-near-protocol     4.56
## 28                      okb-okb    48.99
## 29                  op-optimism     2.38
## 30                    pepe-pepe     0.00
## 31 pol-polygon-ecosystem-token     0.70
## 32            rndr-render-token     6.31
## 33               shib-shiba-inu     0.00
## 34                   sol-solana   125.88
## 35     steth-lido-staked-ether  2805.06
## 36                  stx-stacks     1.81
## 37                      sui-sui     1.09
## 38              tao-bittensor   387.73
## 39    toncoin-the-open-network     4.59
## 40                    trx-tron     0.12
## 41                 uni-uniswap     7.77
## 42               usdc-usd-coin     1.00
## 43                 usdt-tether     1.00
## 44                wbt-whitebit     8.52
## 45        wbtc-wrapped-bitcoin 55699.10
## 46                   weth-weth  2807.00
## 47              wif-dogwifcoin     1.85
## 48                 xlm-stellar     0.11
## 49                  xmr-monero   153.16
## 50                     xrp-xrp     0.56
```

```r
tbl_1 = aggregate(data$price, by = list(data$name), FUN=median)
tbl_1$x = round(tbl_1$x, 2)
tbl_1
```

```
##                       Group.1        x
## 1                    aave-new    98.90
## 2                 ada-cardano     0.44
## 3                   apt-aptos     8.34
## 4              avax-avalanche    32.16
## 5            bch-bitcoin-cash   338.22
## 6            bnb-binance-coin   533.44
## 7                 btc-bitcoin 60156.17
## 8        btcb-binance-bitcoin 60156.76
## 9          cro-cryptocom-chain     0.09
## 10                    dai-dai     1.00
## 11              doge-dogecoin     0.11
## 12               dot-polkadot     6.50
## 13        etc-ethereum-classic    22.96
## 14               eth-ethereum  2662.59
## 15     fdusd-first-digital-usd     1.00
## 16                   fetch-ai     1.22
## 17               fil-filecoin     5.06
## 18                 ftm-fantom     0.49
## 19      hbar-hedera-hashgraph     0.08
## 20       icp-internet-computer     9.77
## 21             imx-immutable-x     1.86
## 22                  kas-kaspa     0.14
## 23              leo-leo-token     5.74
## 24             link-chainlink    14.36
## 25               ltc-litecoin    71.55
```

```
## 26                   mnt-mantle     0.66
## 27           near-near-protocol     4.45
## 28                      okb-okb    49.66
## 29                  op-optimism     2.16
## 30                    pepe-pepe     0.00
## 31 pol-polygon-ecosystem-token     0.72
## 32            rndr-render-token     5.57
## 33               shib-shiba-inu     0.00
## 34                   sol-solana   138.15
## 35      steth-lido-staked-ether  2660.99
## 36                   stx-stacks     1.72
## 37                      sui-sui     1.01
## 38                tao-bittensor   340.36
## 39    toncoin-the-open-network     5.23
## 40                     trx-tron     0.12
## 41                  uni-uniswap     7.11
## 42                usdc-usd-coin     1.00
## 43                  usdt-tether     1.00
## 44                 wbt-whitebit     9.27
## 45         wbtc-wrapped-bitcoin 60089.87
## 46                    weth-weth  2662.23
## 47               wif-dogwifcoin     1.92
## 48                  xlm-stellar     0.11
## 49                  xmr-monero    157.21
## 50                      xrp-xrp     0.57
```

We don't have a clear probability law for the distribution of the bitcoin for this year. We would love to see other aspect like the pourcentage variation within the year and the difference between the mean and each values

```r
head(data, 10)
```

```
##    X  timestamp    price  volume_24h   market_cap        name
## 1  0 2023-10-19 28497.38 10408090970 556216437429 btc-bitcoin
## 2  1 2023-10-20 29436.80 17416159256 574576473068 btc-bitcoin
## 3  2 2023-10-21 29805.58 13954479389 581800798688 btc-bitcoin
## 4  3 2023-10-22 29923.96 11141548512 584139969263 btc-bitcoin
## 5  4 2023-10-23 30903.85 16559018229 603298245173 btc-bitcoin
## 6  5 2023-10-24 34084.20 39316464490 665414809955 btc-bitcoin
## 7  6 2023-10-25 34339.76 27089124613 670433978210 btc-bitcoin
## 8  7 2023-10-26 34332.81 19520008719 670329263962 btc-bitcoin
## 9  8 2023-10-27 33975.26 15867149140 663378526977 btc-bitcoin
## 10 9 2023-10-28 34109.35 12519323280 666032444786 btc-bitcoin
```

```r
percentage = function(price) {
  pourcentage =  1- (mean(price) - sd(price)) / mean(price)
}
```

```r
numbers = data %>%
  dplyr::select(name, price) %>%
  group_by(name) %>%
  summarise(ecart_type = round(sd(price), 3)
            , moyenne = round(mean(price), 3)
            , mediane = round(median(price), 3)
            , p = percentage(price) * 100
            )
```

```r
numbers
```

```
## # A tibble: 50 x 5
##    name                 ecart_type   moyenne   mediane       p
##    <chr>                     <dbl>     <dbl>     <dbl>   <dbl>
##  1 aave-new                 21.2      106.        98.9   20.0
##  2 ada-cardano               0.116      0.462      0.441 25.1
##  3 apt-aptos                 2.52       8.67       8.34  29.0
##  4 avax-avalanche           10.4       31.7       32.2   32.9
##  5 bch-bitcoin-cash        107.       352.       338.    30.5
##  6 bnb-binance-coin        145.       460.       533.    31.5
##  7 btc-bitcoin           11732.     55733.     60156.    21.1
##  8 btcb-binance-bitcoin  11734      55740.     60157.    21.1
##  9 cro-cryptocom-chain       0.024      0.1        0.092 23.9
## 10 dai-dai                   0.001      1.00       1.00   0.0523
## # i 40 more rows
```

```r
btc = data[which(data$name == "btc-bitcoin"),]
```

```r
btc_price = btc$price
moy = mean(btc_price)
```

```r
sd(btc_price)
```

```
## [1] 11732.15
```

```r
numbers = numbers %>%
  mutate(rang = row_number(desc(p))) %>%
  arrange(desc(rang))
```

We want to look at the coin with the weakest standard deviation pourcentage which can be use as a criteria to see if the coin has been consistent through the last year. We gonna select the lowest ranked "standard deviation pourcentage". Let's say $p < 20\%$.

```r
lowest_sd = numbers %>%
  dplyr::filter(p <= 20 & rang < 49)

lowest_sd
```

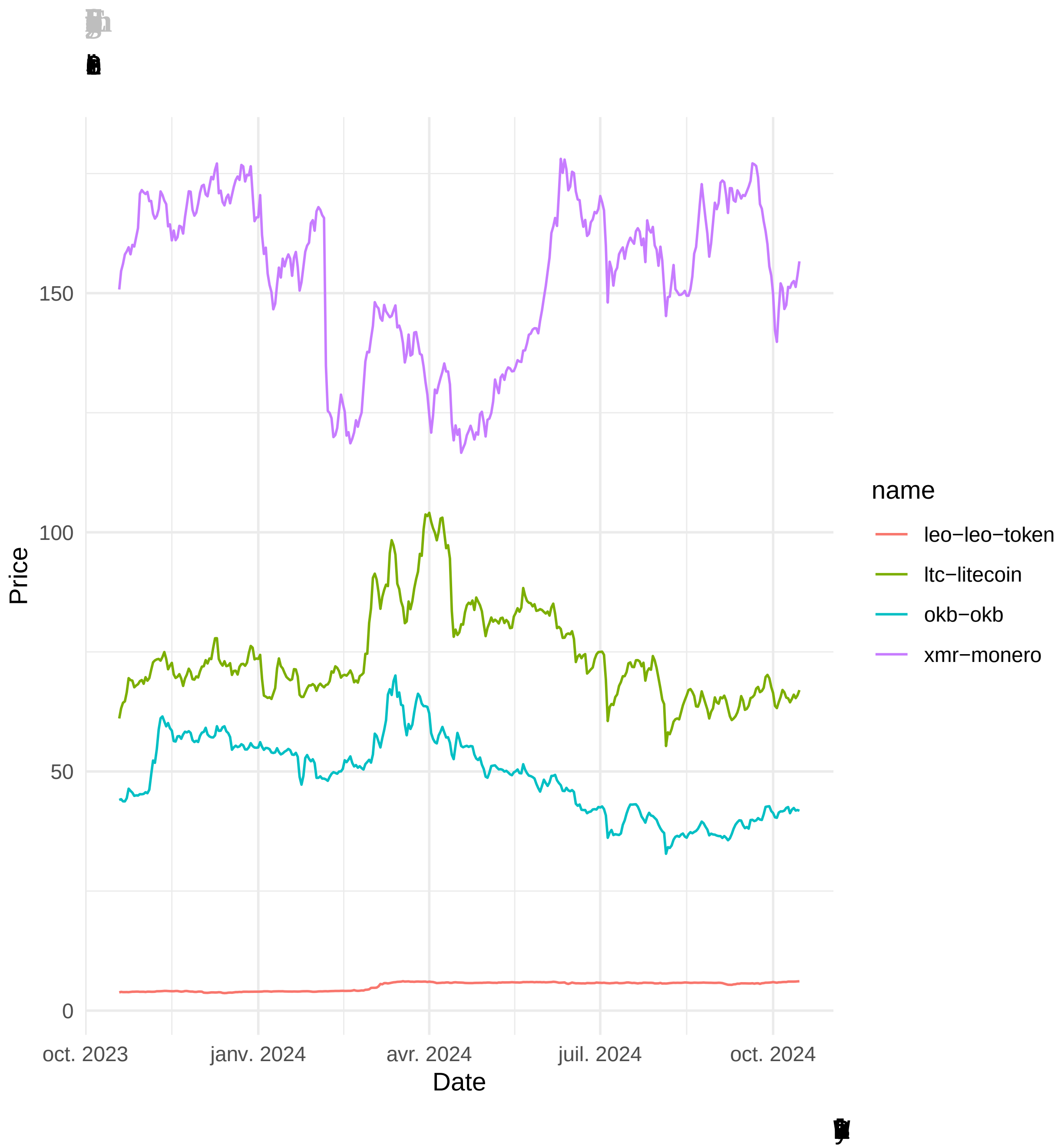
```
## # A tibble: 9 x 6
##   name                       ecart_type moyenne mediane       p  rang
##   <chr>                           <dbl>   <dbl>   <dbl>   <dbl> <int>
## 1 dai-dai                         0.001   1.00    1.00   0.0523    48
## 2 fdusd-first-digital-usd         0.001   1       1      0.129     47
## 3 xrp-xrp                         0.054   0.565   0.567  9.64      46
## 4 xmr-monero                     16.8   153.    157.    11.0       45
## 5 ltc-litecoin                    9.66   74.1    71.6   13.0       44
## 6 xlm-stellar                     0.015   0.111   0.111 13.1       43
## 7 trx-tron                        0.018   0.123   0.121 14.5       42
## 8 okb-okb                         8.18   49.0    49.7   16.7       41
## 9 leo-leo-token                   0.898   5.14    5.74  17.5       40
```

```r
data2 = data[data$name %in% lowest_sd$name,]
head(data2)
```

```
##      X  timestamp    price volume_24h  market_cap    name
## 2179 0 2023-10-19 0.488009  692174303 26079727969 xrp-xrp
## 2180 1 2023-10-20 0.518275 1336047140 27697173481 xrp-xrp
```

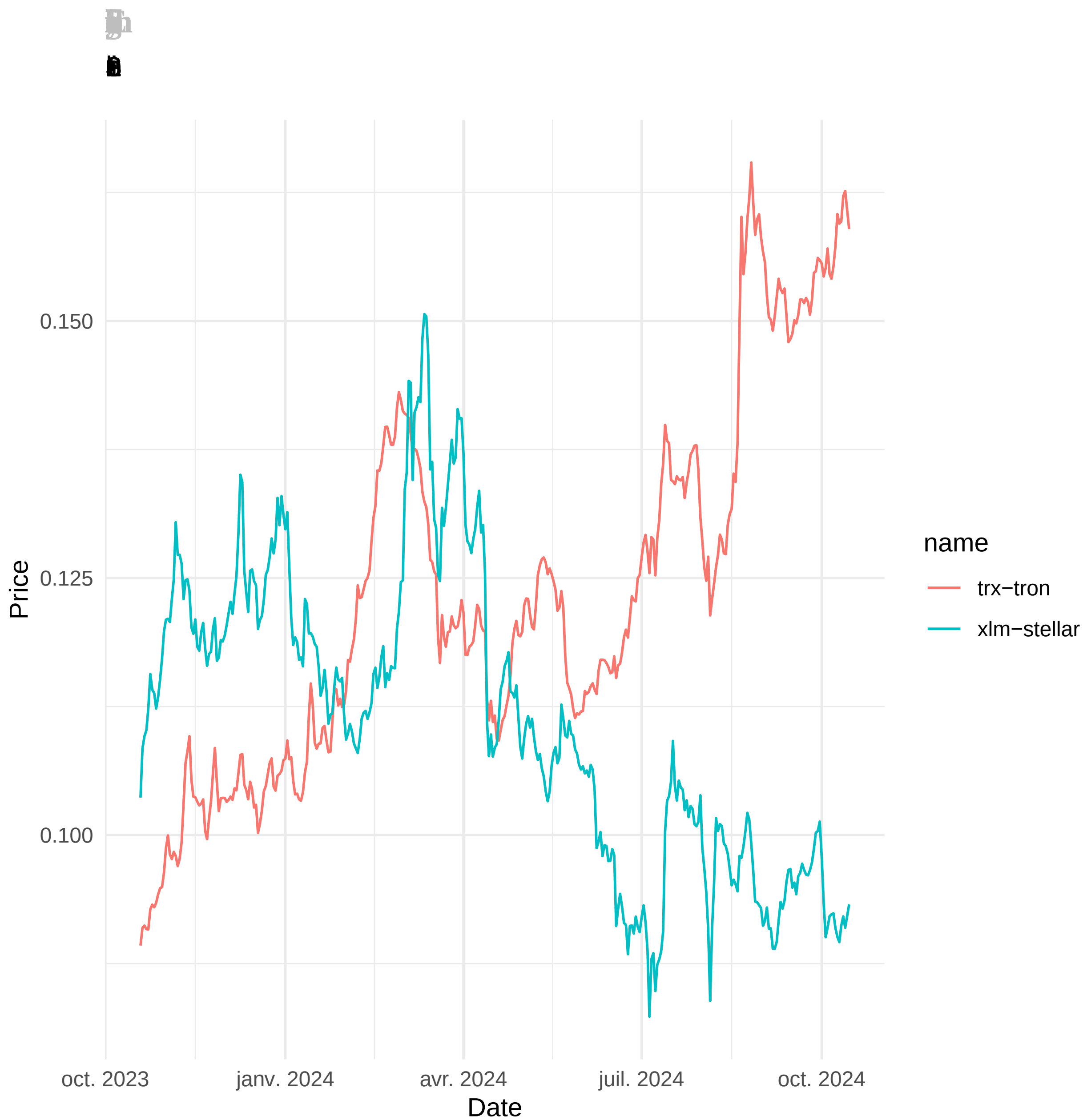
```
## 2181 2 2023-10-21 0.518605  677489103 27714792259 xrp-xrp
## 2182 3 2023-10-22 0.519916  507022951 27784852869 xrp-xrp
## 2183 4 2023-10-23 0.530073  789497105 28327624058 xrp-xrp
## 2184 5 2023-10-24 0.554511 1771040825 29633615168 xrp-xrp
```

```r
data2 %>%
  dplyr::filter(name %in% c("ltc-litecoin", "xmr-monero", "okb-okb","leo-leo-token")) %>%
  ggplot(aes(timestamp, price, col = name)) +
  geom_line() +
  theme_minimal() +
  labs(title = "Evolution of the litecoin, the okb and the xmr coins throught the year"
       , subtitle = "Since october 2023\n"
       , caption = "By William Bak"
       , x = "Date"
       , y = "Price"
       ) +
  theme(plot.title = element_text(family = "Times New Roman"
                                  , vjust = .5
                                  , face = "bold"
                                  , color = "grey"
                                  , size = 14
                                  )
        , plot.subtitle = element_text(family = 'Calibri Light'
                                       , size = 12
                                       )
        , plot.caption = element_text(family = 'Calibri Light'
                                      , size = 12
                                      )
        )
```




```
data2 %>%
  dplyr::filter(!(name %in% c("ltc-litecoin", "xmr-monero", "okb-okb", "leo-leo-token", "fdusd-first-di
  ggplot(aes(timestamp, price, col = name)) +
  geom_line() +
  theme_minimal() +
  labs(title = "Evolution of the trx and xmr coins throught the year"
       , subtitle = "Since october 2023\n"
       , caption = "By William Bak"
       , x = "Date"
       , y = "Price"
       ) +
```

```
  theme(plot.title = element_text(family = "Times New Roman"
                                  , vjust = .5
                                  , face = "bold"
                                  , color = "grey"
                                  , size = 14
                                  )
        , plot.subtitle = element_text(family = 'Calibri Light'
                                       , size = 12
                                       )
        , plot.caption = element_text(family = 'Calibri Light'
                                       , size = 12
                                       )
        )
```



TRX seems like a great coin to invest in going forward we gonna try to predict the future values, just like XMR

```
library(timeSeries)
```

```
##
## Attachement du package : 'timeSeries'

## L'objet suivant est masqué depuis 'package:dplyr':
##
##     lag

## Les objets suivants sont masqués depuis 'package:graphics':
```

```
##
##     lines, points
```

```
trx_data = data[data["name"] == 'trx-tron',]
```

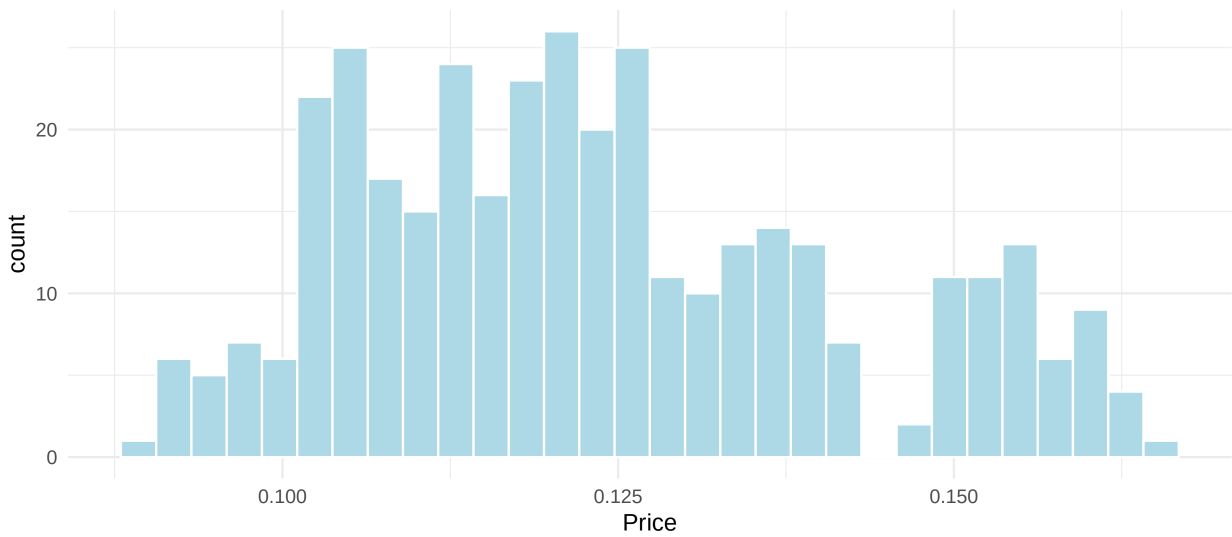
```
library(tseries)
```

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

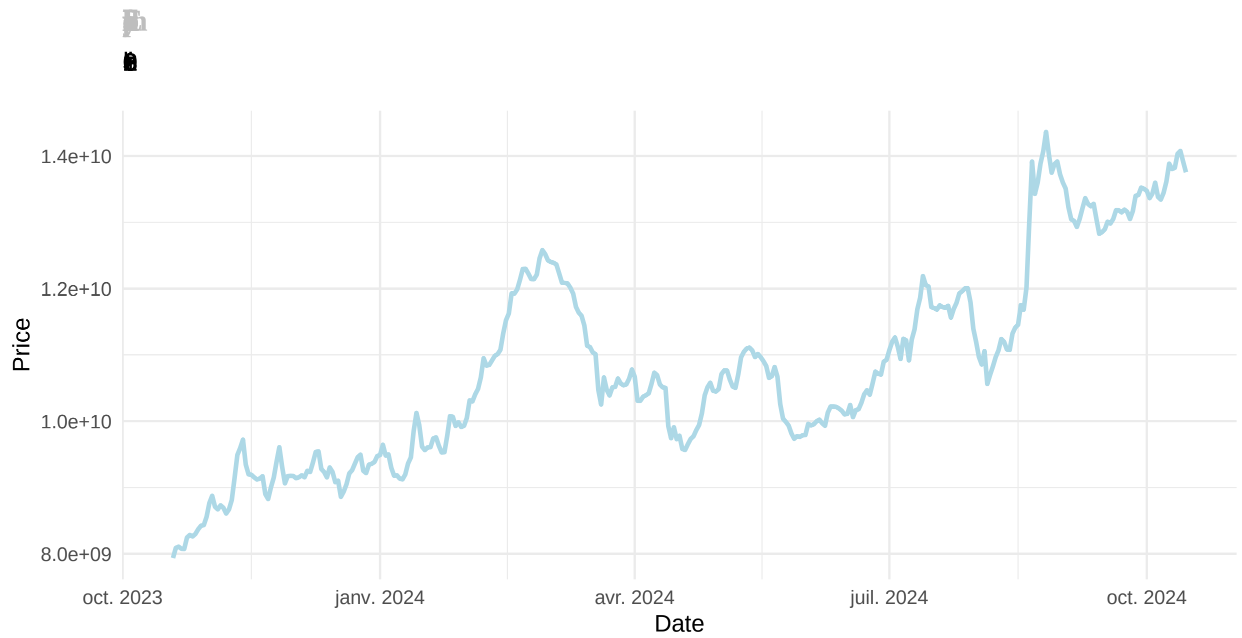
```
library(timeSeries)
```

```
data %>% dplyr::select(price, timestamp, name) %>%
  dplyr::filter(name == 'trx-tron') %>%
  ggplot(aes(price)) +
  geom_histogram(fill = 'lightblue'
                 , col = "white"
                 ) +
  theme_minimal() +
  labs(title = "Distribution of trx price"
       , subtitle = "Since October 2023\n"
       , caption = "Done by William Bak"
       , x = "Price"
       ) +
  theme(plot.title = element_text(family = "Times New Roman"
                                  , vjust = .5
                                  , face = "bold"
                                  , color = "grey"
                                  , size = 14
                                  )
        , plot.subtitle = element_text(family = 'Calibri Light'
                                       , size = 12
                                       )
        , plot.caption = element_text(family = 'Calibri Light'
                                      , size = 12
                                      )
        )
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
data %>% dplyr::select(market_cap, timestamp, name) %>%
  dplyr::filter(name == 'trx-tron') %>%
  ggplot(aes(x = timestamp, y = market_cap)) +
  geom_line(linewidth = 1
            , col = 'lightblue'
            ) +
  theme_minimal() +
  labs(title = "Evolution of the trx marketcap per day"
       , subtitle = "Since October 2023\n"
       , caption = "Done by William Bak"
       , x = "Date"
       , y = "Price"
       ) +
  theme(plot.title = element_text(family = "Times New Roman"
                                  , vjust = .5
                                  , face = "bold"
                                  , color = "grey"
                                  , size = 14
                                  )
        , plot.subtitle = element_text(family = 'Calibri Light'
                                       , size = 12
                                       )
        , plot.caption = element_text(family = 'Calibri Light'
                                      , size = 12
                                      )
        )
```




```r
model1 =lm(price ~ timestamp + market_cap, data = trx_data)
summary(model1)
```

```
##
## Call:
## lm(formula = price ~ timestamp + market_cap, data = trx_data)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -2.926e-04 -8.710e-05 -4.283e-05  1.032e-04  2.885e-04
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.698e-01  2.228e-03  -76.22   <2e-16 ***
## timestamp    8.546e-06  1.159e-07   73.74   <2e-16 ***
## market_cap   1.145e-11  8.084e-15 1416.12   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.000135 on 360 degrees of freedom
## Multiple R-squared:  0.9999, Adjusted R-squared:  0.9999
## F-statistic: 3.202e+06 on 2 and 360 DF,  p-value: < 2.2e-16
```

That means that if we can predict the market cap in an efficient way we can predict the trx coin with a linear model. Let's try a generalized linear model in order to find the market cap

```r
model2 = glm(market_cap ~ timestamp, data = trx_data,  family = "poisson")
summary(model2)
```

```
##
## Call:
## glm(formula = market_cap ~ timestamp, family = "poisson", data = trx_data)
##
```

```
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) 1.688e+00  9.595e-05   17592   <2e-16 ***
## timestamp   1.080e-03  4.836e-09  223271   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 7.4671e+10  on 362  degrees of freedom
## Residual deviance: 2.4630e+10  on 361  degrees of freedom
## AIC: 2.463e+10
##
## Number of Fisher Scoring iterations: 3
```

```r
library(Metrics)
```

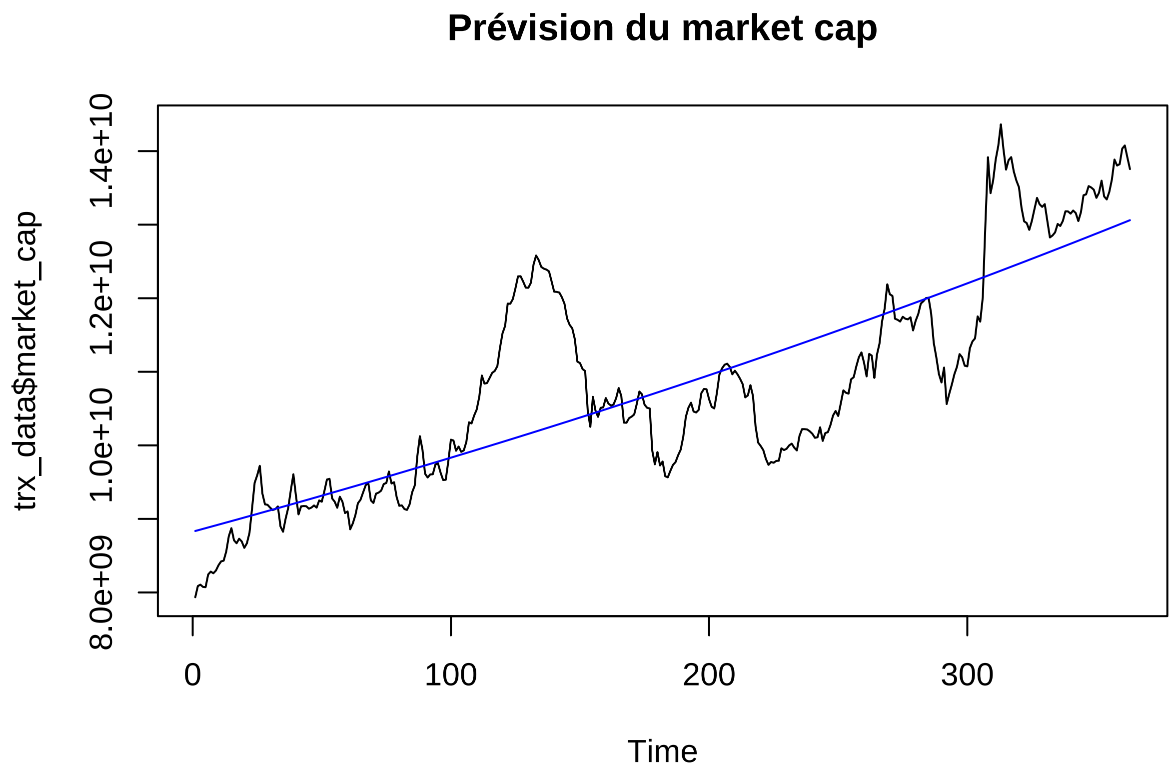
```
##
## Attachement du package : 'Metrics'

## Les objets suivants sont masqués depuis 'package:caret':
##
##     precision, recall
```

```r
mae1 = mae(trx_data$market_cap, model2$fitted.values)
mae1
```

```
## [1] 654138360
```

```r
pred_actual = cbind(trx_data$market_cap, model2$fitted.values)
```

```r
ts.plot(trx_data$market_cap, col = "black", main = "Prévision du market cap")
lines(model2$fitted.values, col = "blue")
```



Essaie des KNN par regression

```
library(MASS)
```

```
##
## Attachement du package : 'MASS'

## L'objet suivant est masqué depuis 'package:dplyr':
##
##     select
```

```
library(FNN)
```

```
##
## Attachement du package : 'FNN'

## Les objets suivants sont masqués depuis 'package:class':
##
##     knn, knn.cv
```

```
marketcap1 = trx_data[c("market_cap", "timestamp")]
marketcap1$timestamp = as.numeric(marketcap1$timestamp)
```

```
indice_train_x = sample(1:nrow(marketcap1), nrow(marketcap1) * 2 / 3, replace = T)
train_x = marketcap1[indice_train_x, "timestamp"]
test_x = marketcap1[-indice_train_x, "timestamp"]
train_y = marketcap1[indice_train_x, "market_cap"]
```

```r
model3 = knn.reg(train = as.data.frame(train_x), test = as.data.frame(test_x), y = train_y, k = 12)
```

```r
df_pred1 = marketcap1[-indice_train_x, ]
df_pred1$prediction = model3$pred
head(df_pred1, 20)
```

```
##      market_cap timestamp prediction
## 3268 7935038451     19649 8548235200
## 3270 8106124533     19651 8548235200
## 3271 8075040990     19652 8548235200
## 3274 8283669610     19655 8548235200
## 3275 8260646952     19656 8548235200
## 3276 8297899626     19657 8548235200
## 3277 8369739072     19658 8548235200
## 3279 8431984807     19660 8548235200
## 3282 8873967320     19663 8684396833
## 3285 8730543038     19666 8913555069
## 3286 8693848849     19667 8913555069
## 3287 8606464708     19668 8990437175
## 3288 8668655502     19669 8990437175
## 3290 9143834127     19671 9069564362
## 3291 9491023731     19672 9103227433
## 3292 9594109355     19673 9108177616
## 3295 9197807641     19676 9209262032
## 3296 9191418736     19677 9209262032
## 3299 9133385784     19680 9237484868
## 3301 8898506044     19682 9237484868
```

Next days predictions

```r
max(data$timestamp)
```

```
## [1] "2024-10-15"
```

```r
trx_data[trx_data$timestamp == max(trx_data$timestamp), ]
```

```
##        X  timestamp   price volume_24h  market_cap     name
## 3630 362 2024-10-15 0.15894  674041963 13755217238 trx-tron
```

```r
new_timestamp = as.Date(c("2024-10-16","2024-10-17","2024-10-18","2024-10-19","2024-10-20","2024-10-21"
new_timestamp
```

```
## [1] "2024-10-16" "2024-10-17" "2024-10-18" "2024-10-19" "2024-10-20"
## [6] "2024-10-21" "2024-10-22" "2024-10-23" "2024-10-24"
```

```r
new_data = data.frame(as.numeric(new_timestamp))
```

Since predict could not work with a `knnReg` object we will use a naive method and using all the dataset as a training set

```r
train_x =  marketcap1[, "timestamp"]
train_y =  marketcap1[, "market_cap"]

model4 = knn.reg(train = as.data.frame(train_x), test = new_data, y = train_y, k = 12)
model4$pred
```

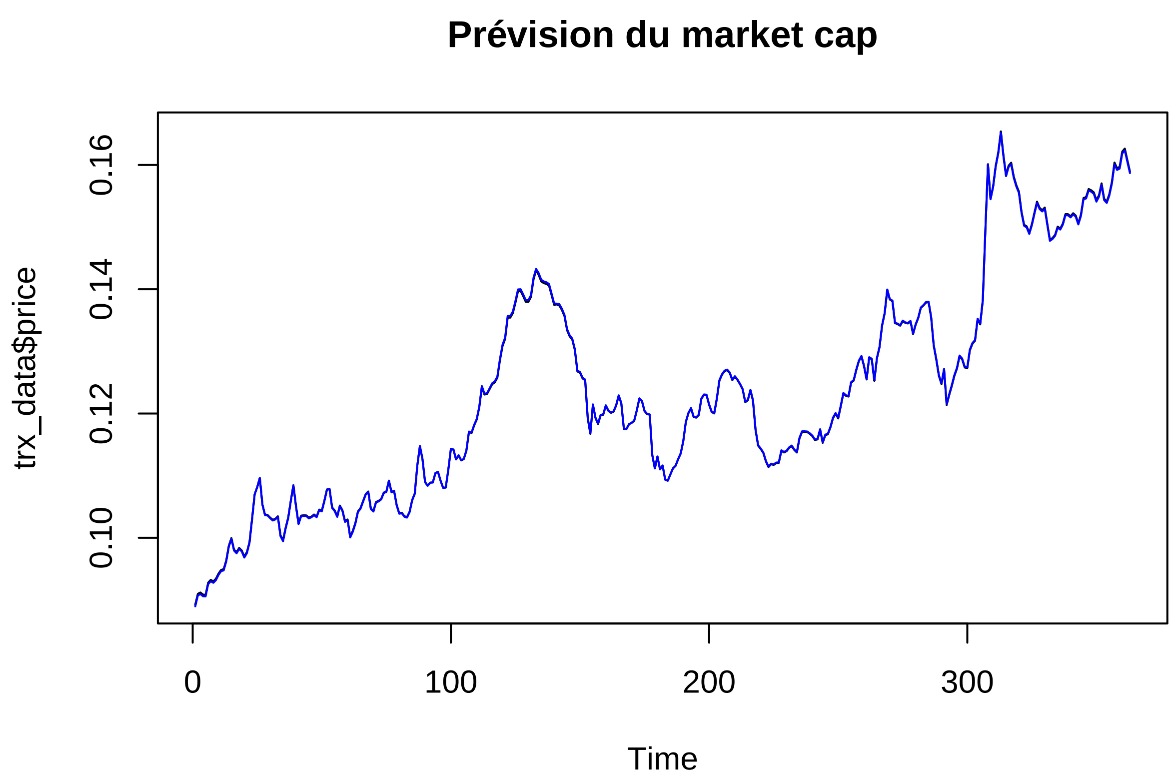
```
## [1] 13722923289 13722923289 13722923289 13722923289 13722923289 13722923289
```

```
## [7] 13722923289 13722923289 13722923289
```

```r
lm_model = lm(price ~ timestamp + market_cap, data = trx_data)
summary(lm_model)
```

```
##
## Call:
## lm(formula = price ~ timestamp + market_cap, data = trx_data)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -2.926e-04 -8.710e-05 -4.283e-05  1.032e-04  2.885e-04
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.698e-01  2.228e-03  -76.22   <2e-16 ***
## timestamp    8.546e-06  1.159e-07   73.74   <2e-16 ***
## market_cap   1.145e-11  8.084e-15 1416.12   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.000135 on 360 degrees of freedom
## Multiple R-squared:  0.9999, Adjusted R-squared:  0.9999
## F-statistic: 3.202e+06 on 2 and 360 DF,  p-value: < 2.2e-16
```

```r
ts.plot(trx_data$price, col = "black", main = "Prévision du market cap")
lines(lm_model$fitted.values, col = "blue")
```



```r
new_timestamp = as.Date(c("2024-10-16","2024-10-17","2024-10-18","2024-10-19","2024-10-20","2024-10-21"
new_timestamp
```

```
## [1] "2024-10-16" "2024-10-17" "2024-10-18" "2024-10-19" "2024-10-20"
## [6] "2024-10-21" "2024-10-22" "2024-10-23" "2024-10-24"
```

```r
new_data = data.frame(new_timestamp)
```

```r
new_data$market_cap = model4$pred
colnames(new_data)[1] = "timestamp"
predict(lm_model, new_data)
```

```
##         1         2         3         4         5         6         7         8
## 0.1583264 0.1583350 0.1583435 0.1583521 0.1583606 0.1583692 0.1583777 0.1583863
##         9
## 0.1583948
```

With the glm response

```r
prediction = predict(model2, new_data, type = "response")
```

```r
new_data2 = data.frame(new_timestamp)
colnames(new_data2)[1] = "timestamp"
new_data2$market_cap = prediction
predict(lm_model, new_data2)
```

```
##         1         2         3         4         5         6         7         8
## 0.1509109 0.1510812 0.1512516 0.1514221 0.1515929 0.1517638 0.1519349 0.1521062
##         9
## 0.1522777
```