Exploratory Data Analysis

Grizzlies

2022-11-07

I. Introduction

The purpose of this study is to investigate how the sentiments (people's general feelings and emotions), which are measured from Twitter, StockTwits, and Reddit, can affect the prices, volumes, and market caps of different cryptocurrencies and stocks, or tickers. We can also learn about the fluctuation of the changes in sentiment scores as well as ticker prices over time. With the information in hand, we expect to find a correlation between sentiment scores and other information about different tickers like prices or volumes, which can help leverage one's trading experience and help traders make profits with accurate data.

II. Background

Using Utradea's Social Sentiment data, we can identify and track popular stocks and cryptocurrencies on social media platform like Twitter, StockTwits, and Reddit. The sentiment scores are inferred from datapoints for stocks and cryptocurrencies mentioned on those social networks, for example posts, likes and comments. Data is sourced and provided over a 24-hour or 72-hour period, which keeps track of the change in price and volume over the period. The change in posts, comments, and impressions over that given time period can also be used to identify hot stocks or cryptocurrencies, which can lead to high sentiment scores.

The units of observations are tickers (stocks/cryptocurrencies). In order to understand the EAD, one only need to understand about tickers, which have price, volume, market cap, and sentiments, which are people's general feelings and emotions towards a specific ticker. The list of variables that are important for the analysis is as belows:

- ticker: the ticker code
- sentiment: the sentiment score of the ticker
- lastSentiment: the last sentiment score measured of the ticker
- sentimentChange: the sentiment score change in percentage
- price: the price of the ticker
- previousClose: the closing price of the ticker previously measured
- change: the change in price
- changePercent: the price change in percentage
- volume: the volume of the ticker
- previousVolume: the volume of the ticker previously measured
- marketCap: the market cap of the ticker

III. Data Wrangling

```
library(readr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)

bullish_data <- read_csv("social_sentiment_twitter_chginsentiment_bullish_03-16-2022.csv")
```

```
## Rows: 200 Columns: 13

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (2): ticker, name
## dbl (11): sentiment, lastSentiment, sentimentChange, rank, price, change, ch...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
bullish_data2 <- read_csv("social_sentiment_stocktwits_chginsentiment_bullish_03-16-2022.csv")
```

```
## Rows: 200 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (2): ticker, name
## dbl (11): sentiment, lastSentiment, sentimentChange, rank, price, change, ch...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
bearish_data <- read_csv("social_sentiment_twitter_chginsentiment_bearish_03-16-2022.csv")
```

```
## Rows: 200 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (2): ticker, name
## dbl (11): sentiment, lastSentiment, sentimentChange, rank, price, change, ch...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
bearish_data2 <- read_csv("social_sentiment_stocktwits_chginsentiment_bearish_03-16-2022.csv")
```

```
## Rows: 200 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (2): ticker, name
## dbl (11): sentiment, lastSentiment, sentimentChange, rank, price, change, ch...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# binding both data for bullish and bearish tickers together
data <- rbind(bullish_data, bearish_data, bullish_data2, bearish_data2)

data <- data %>%
  # filter out all dummy data
  filter(lastSentiment > 0) %>%
  rename(sentimentChangePercent="sentimentChange",
         priceChange="change",
         priceChangePercent="changePercent") %>%
  # adding new variable volumeChangePercent
  mutate(priceChangePercent = priceChangePercent * 100,
         volumeChangePercent= (volume - previousVolume) / previousVolume * 100)

glimpse(data)
```

```
## Rows: 670
## Columns: 14
## $ ticker                <chr> "ANY", "NOV", "ATOS", "WBEV", "ACA", "KALV", "O~
## $ sentiment             <dbl> 2.844205, 3.331944, 10.966667, 31.875000, 16.42~
## $ lastSentiment         <dbl> 0.06968775, 0.09479167, 0.31250000, 1.42857143,~
## $ sentimentChangePercent <dbl> 3981.3553, 3415.0183, 3409.3333, 2131.2500, 122~
## $ rank                  <dbl> 3, 6, 7, 10, 14, 16, 17, 19, 20, 22, 23, 26, 27~
## $ name                  <chr> "Sphere 3D Corp.", "NOV Inc.", "Atossa Therapeu~
## $ price                 <dbl> 1.830, 18.750, 1.270, 2.980, 58.260, 14.440, 14~
## $ priceChange           <dbl> 0.10000002, -0.57000000, 0.09000003, -0.2400000~
## $ priceChangePercent    <dbl> 5.78034830, -2.95031000, 7.62712200, -7.4534200~
## $ volume                <dbl> 2607003, 6887001, 1858733, 32008, 481282, 21820~
## $ marketCap             <dbl> 66444014, 7362474769, 160812620, 39214327, 2836~
## $ previousVolume        <dbl> 2155014, 5881926, 1689995, 28808, 241042, 32666~
## $ previousClose         <dbl> 1.730, 19.320, 1.180, 3.220, 57.060, 14.500, 14~
## $ volumeChangePercent   <dbl> 20.973831, 17.087515, 9.984527, 11.108026, 99.6~
```

We introduce new variable volumeChangePercent, which keeps track of the change in volume in percentage for better reference. For readability, we also change some variables name, for example change and changePercentage, to distinguish from other changes. For priceChangePercentage column, we multiply all values by 100 to match the data format of other percentage changes.

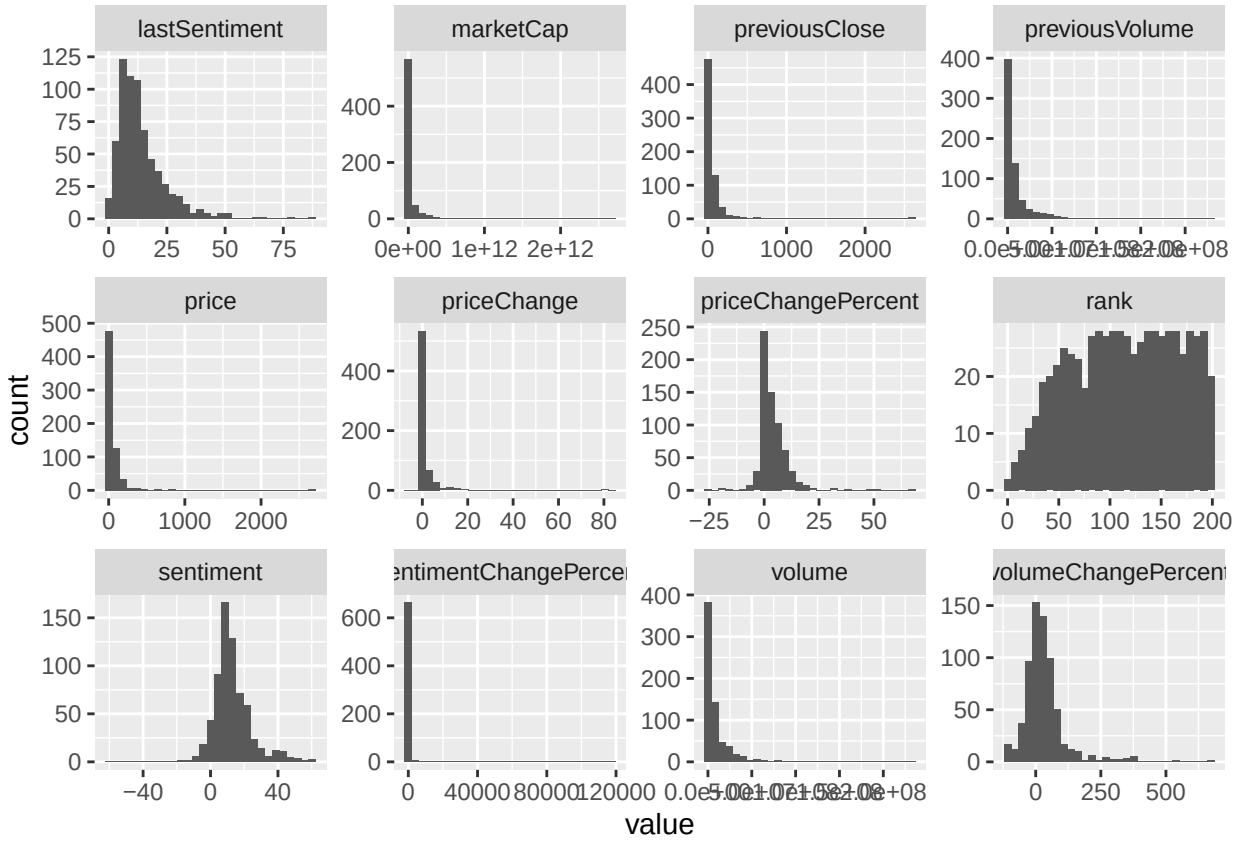
IV. Exploratory Analysis

A histogram of all featured variables in the datasets:

```
library(purrr)
library(tidyr)
data %>%
  keep(is.numeric) %>%
  gather() %>%
  ggplot(aes(value)) +
    facet_wrap(~ key, scales = "free") +
    geom_histogram()
```

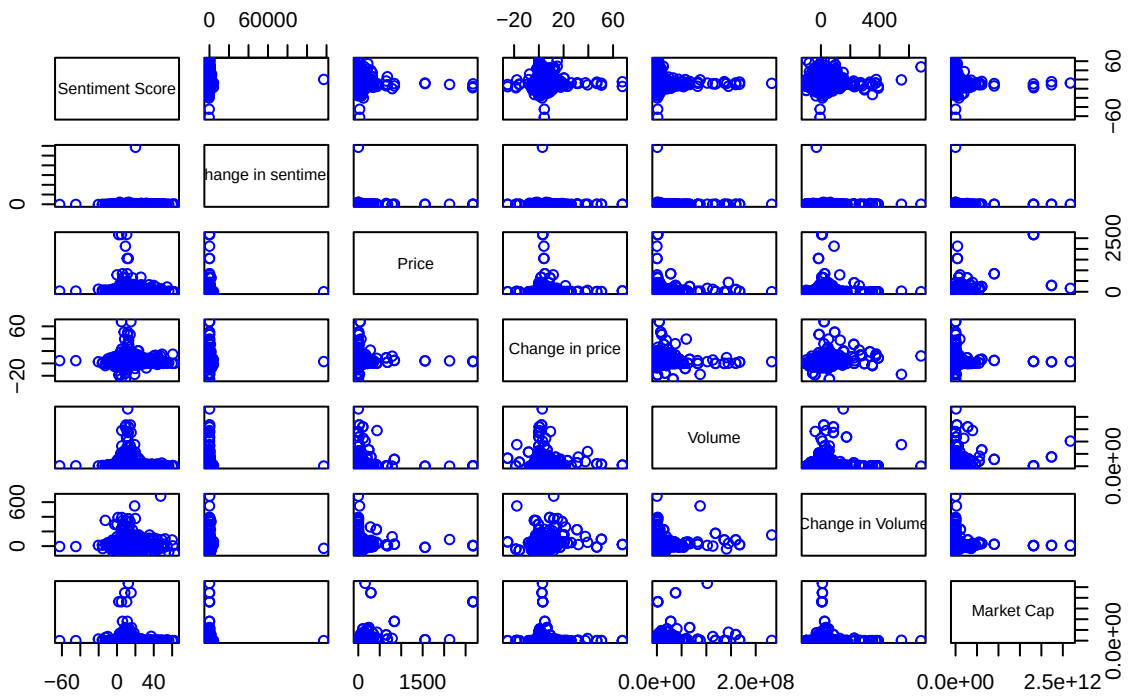
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 3 rows containing non-finite values (stat_bin).
```



The pairs plot of the variables that we use, since these are all important features as discussed in section II. They are sentiment, price, volume, market cap of each ticker together with their changes in percentage.

```
pairs(data[,c("sentiment","sentimentChangePercent","price", "priceChangePercent",
              "volume", "volumeChangePercent", "marketCap")],
      col="blue",
      labels = c('Sentiment Score', 'Change in sentiment', 'Price', "Change in price",
                 "Volume", "Change in Volume", "Market Cap"),
      main="Important variables")
```



Look at the correlations between the variables:

```
res = cor(data[,c("sentiment","sentimentChangePercent","price", "priceChangePercent",
              "volume", "volumeChangePercent", "marketCap")], use="complete.obs")
res
```

```
##                          sentiment sentimentChangePercent       price
## sentiment                1.00000000             0.03832636 -0.02898016
## sentimentChangePercent   0.03832636             1.00000000 -0.01326746
## price                   -0.02898016            -0.01326746  1.00000000
## priceChangePercent       0.02264758            -0.00582294 -0.01705073
## volume                  -0.05835371            -0.01869624 -0.01402218
## volumeChangePercent     -0.01499749            -0.02722581 -0.01978711
## marketCap               -0.05036620            -0.01097214  0.56075678
##                         priceChangePercent      volume volumeChangePercent
## sentiment                      0.02264758 -0.05835371         -0.01499749
## sentimentChangePercent        -0.00582294 -0.01869624         -0.02722581
## price                         -0.01705073 -0.01402218         -0.01978711
## priceChangePercent             1.00000000 -0.05611939          0.18621690
## volume                        -0.05611939  1.00000000          0.09069845
## volumeChangePercent            0.18621690  0.09069845          1.00000000
## marketCap                     -0.04622797  0.19137405         -0.03912905
##                           marketCap
## sentiment               -0.05036620
## sentimentChangePercent -0.01097214
## price                   0.56075678
```

```
## priceChangePercent     -0.04622797
## volume                  0.19137405
## volumeChangePercent    -0.03912905
## marketCap               1.00000000
```

Find out if there are missing values:

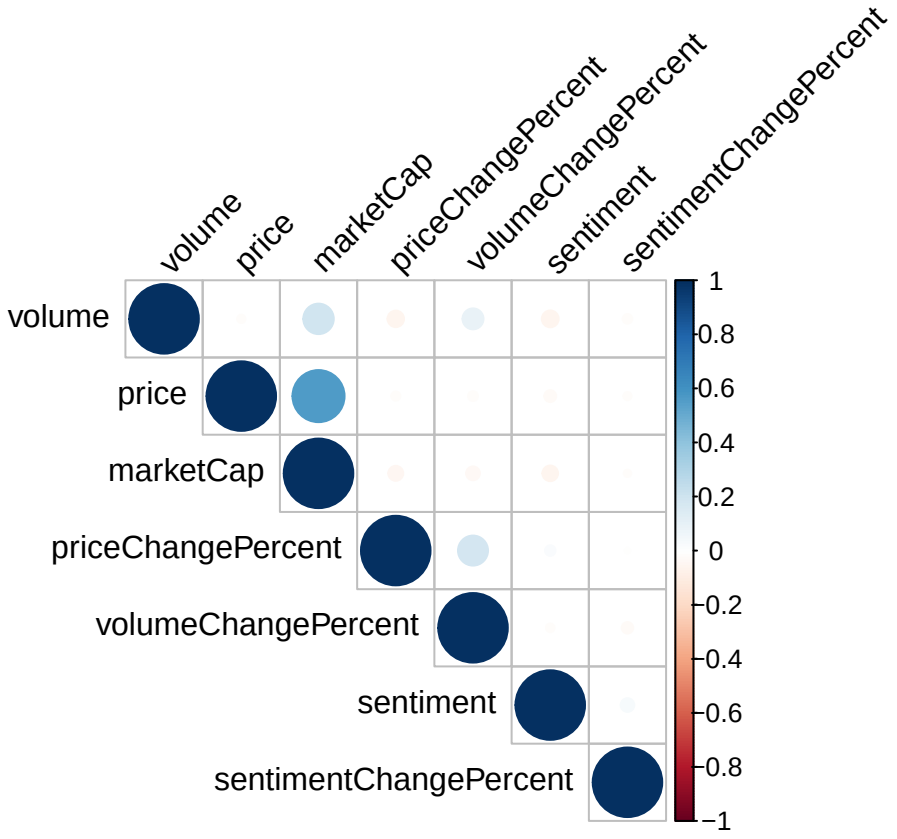
```
sapply(data, function(x) sum(is.na(x)))
```

```
##                  ticker               sentiment           lastSentiment
##                       0                       0                       0
## sentimentChangePercent                    rank                    name
##                       0                       0                       0
##                   price             priceChange     priceChangePercent
##                       0                       0                       0
##                  volume               marketCap         previousVolume
##                       0                       1                       1
##           previousClose     volumeChangePercent
##                       0                       1
```

```
library(corrplot)
```

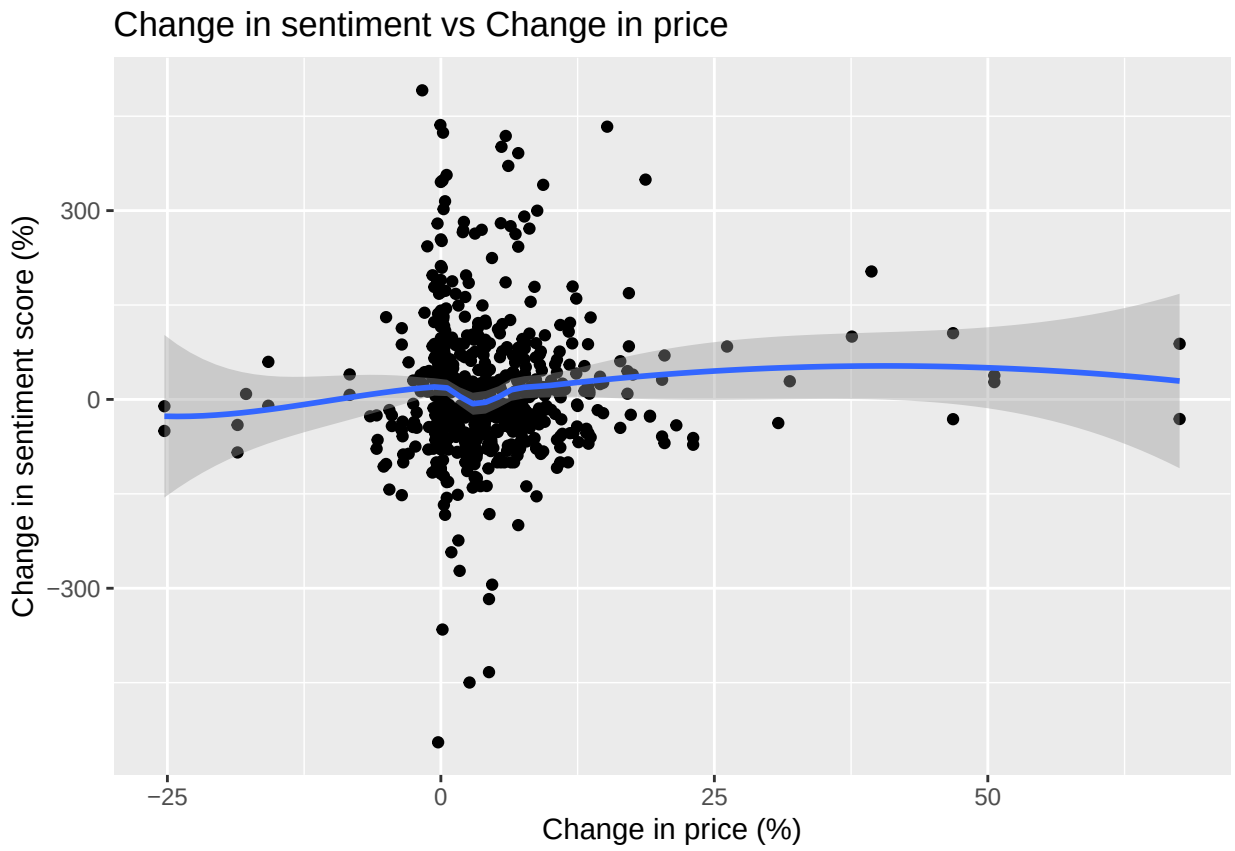
```
## corrplot 0.92 loaded
```

```
corrplot(res, type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45)
```



```
plotdata <- data %>%
  filter(sentimentChangePercent < 500)
ggplot(plotdata, aes(y=sentimentChangePercent, x = priceChangePercent)) +
  geom_point() + geom_smooth() +
  labs(
    title = "Change in sentiment vs Change in price",
    y = "Change in sentiment score (%)",
    x = "Change in price (%)"
  )
```

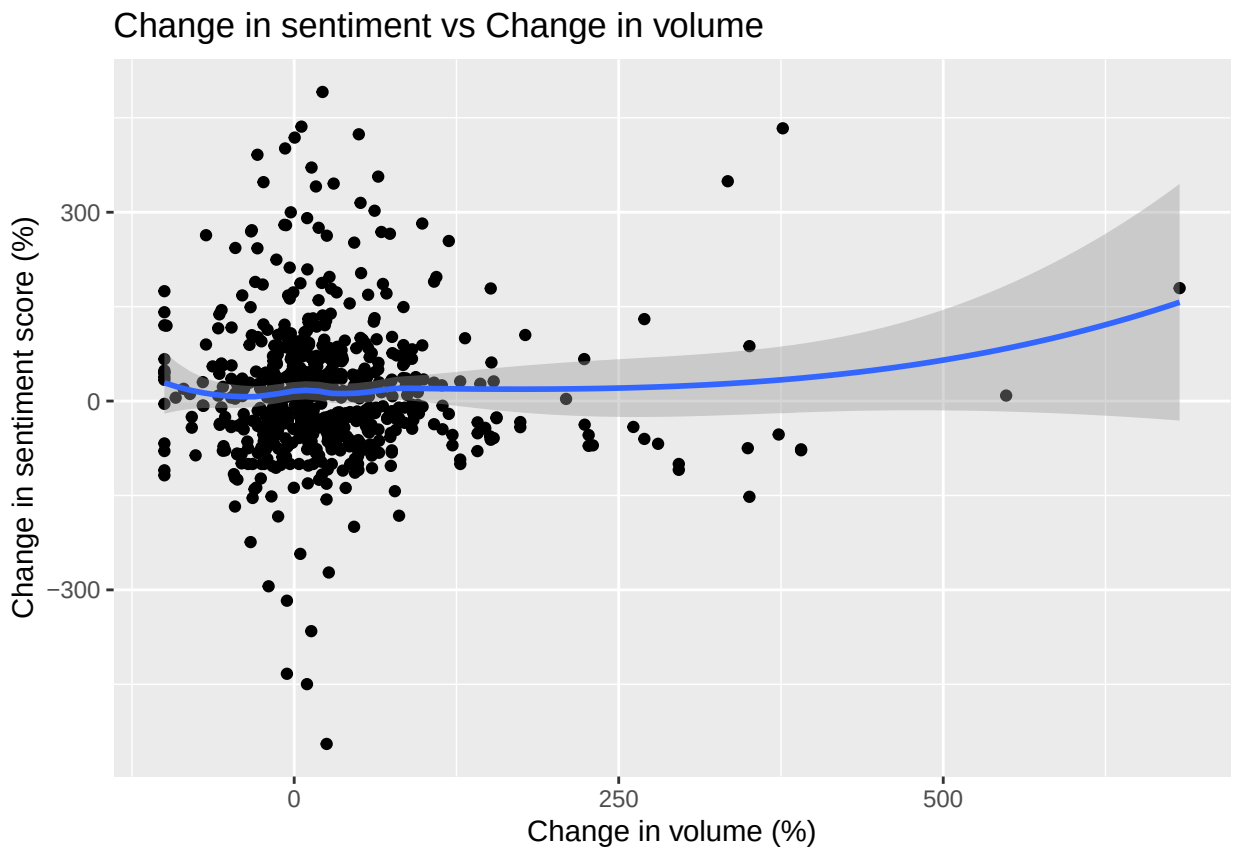
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
plotdata1 <- data %>%
  filter(sentimentChangePercent < 500)
ggplot(plotdata1, aes(y=sentimentChangePercent, x = volumeChangePercent)) +
  geom_point() + geom_smooth() +
  labs(
    title = "Change in sentiment vs Change in volume",
    y = "Change in sentiment score (%)",
    x = "Change in volume (%)"
  )
```

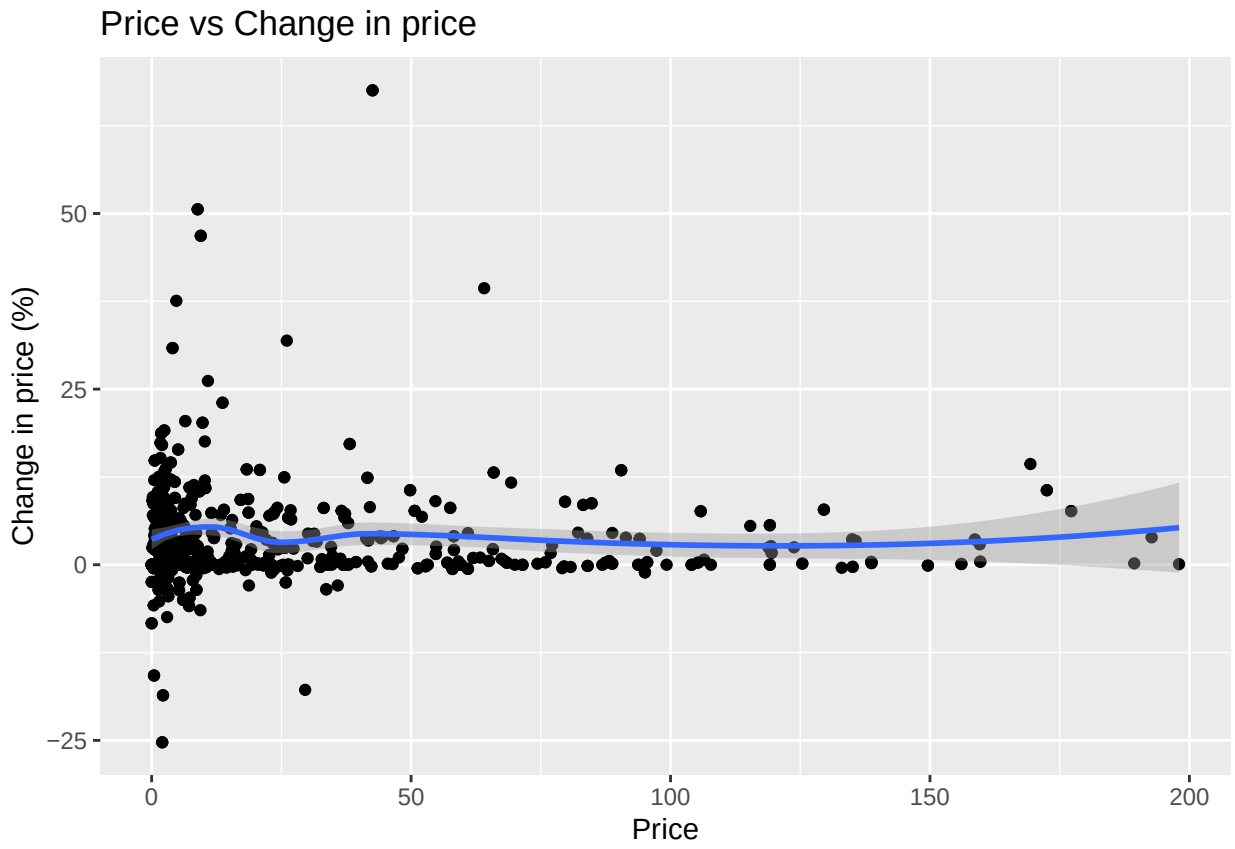
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'

## Warning: Removed 1 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```



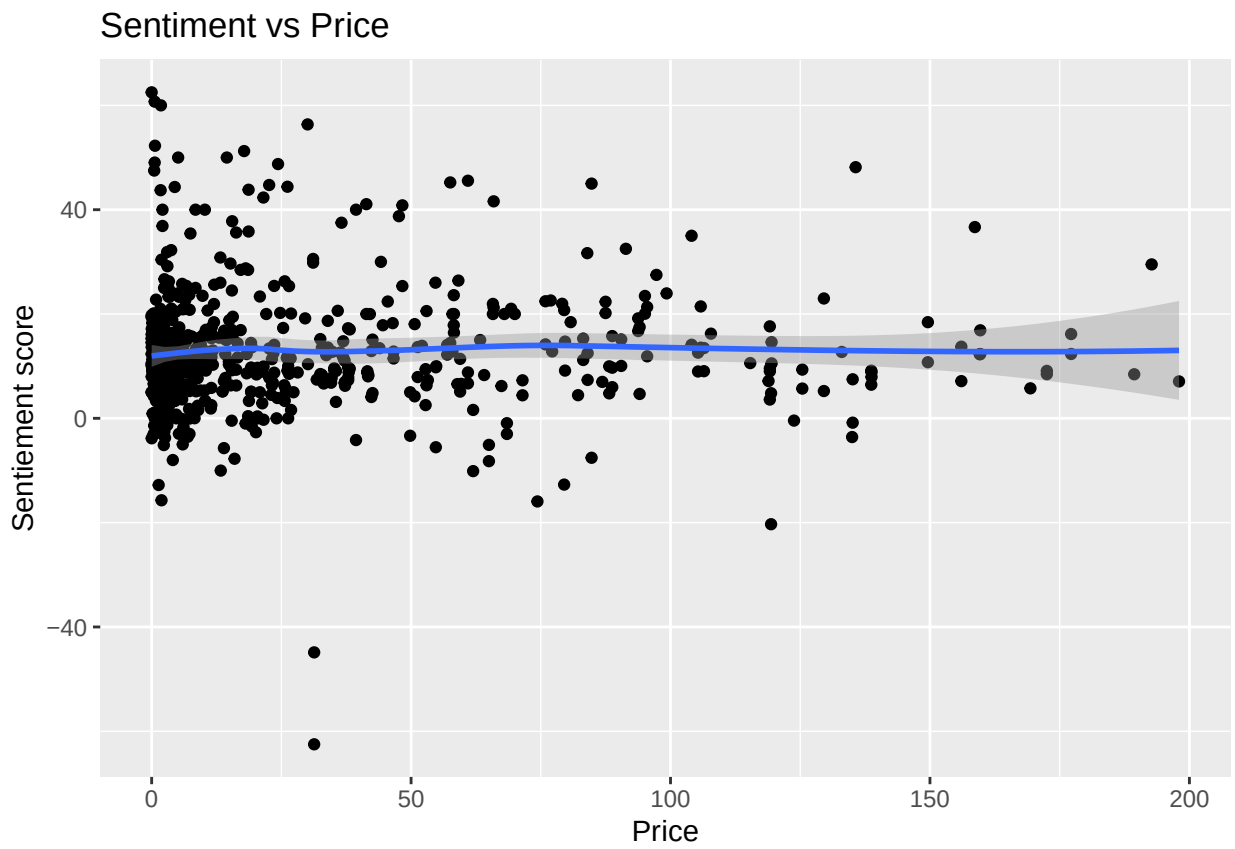

```
plotdata1 <- data %>%
  filter(price < 200)
ggplot(plotdata1, aes(x=price, y = priceChangePercent)) +
  geom_point() + geom_smooth() +
  labs(
    title = "Price vs Change in price",
    x = "Price",
    y = "Change in price (%)"
  )
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



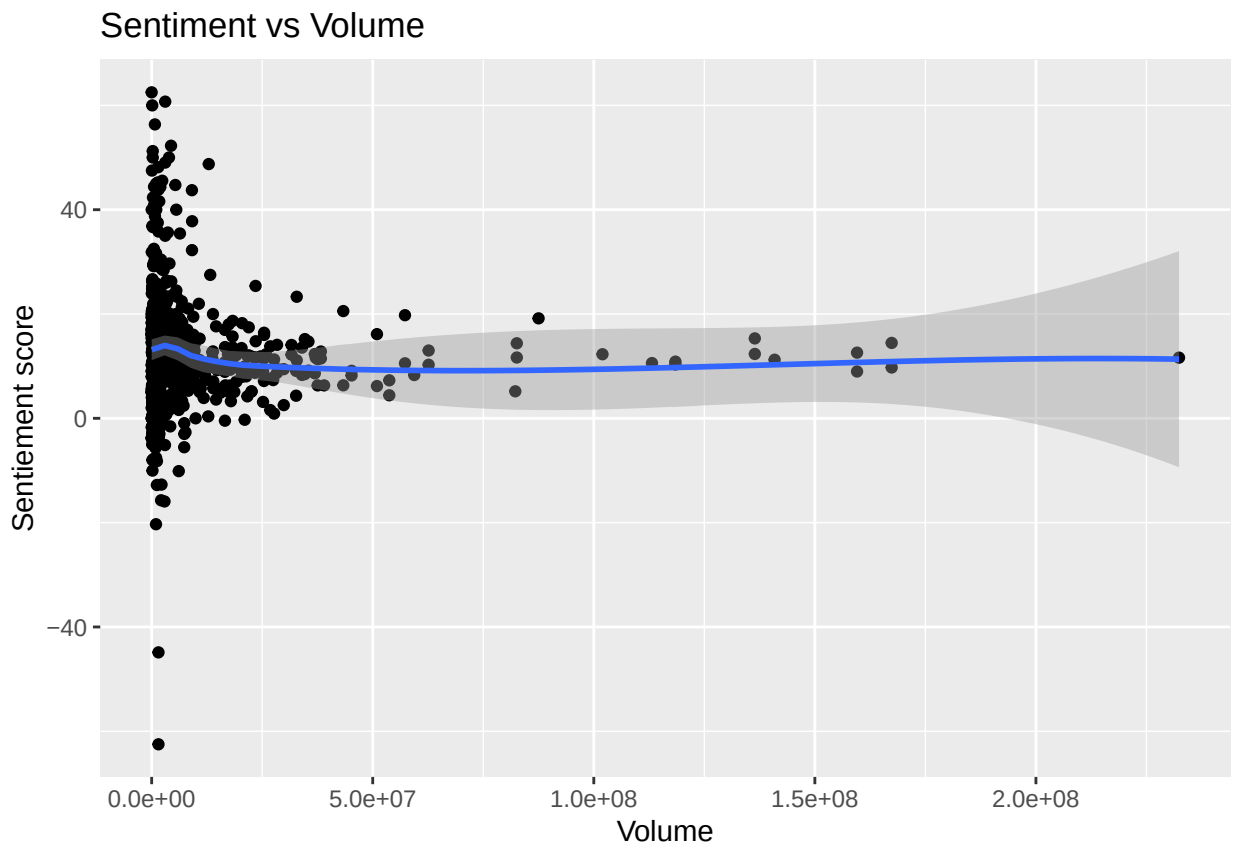

```
plotdata1 <- data %>%
  filter(price < 200)
ggplot(plotdata1, aes(y=sentiment, x = price)) +
  geom_point() + geom_smooth() +
  labs(
    title = "Sentiment vs Price",
    y = "Sentiement score",
    x = "Price"
  )
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



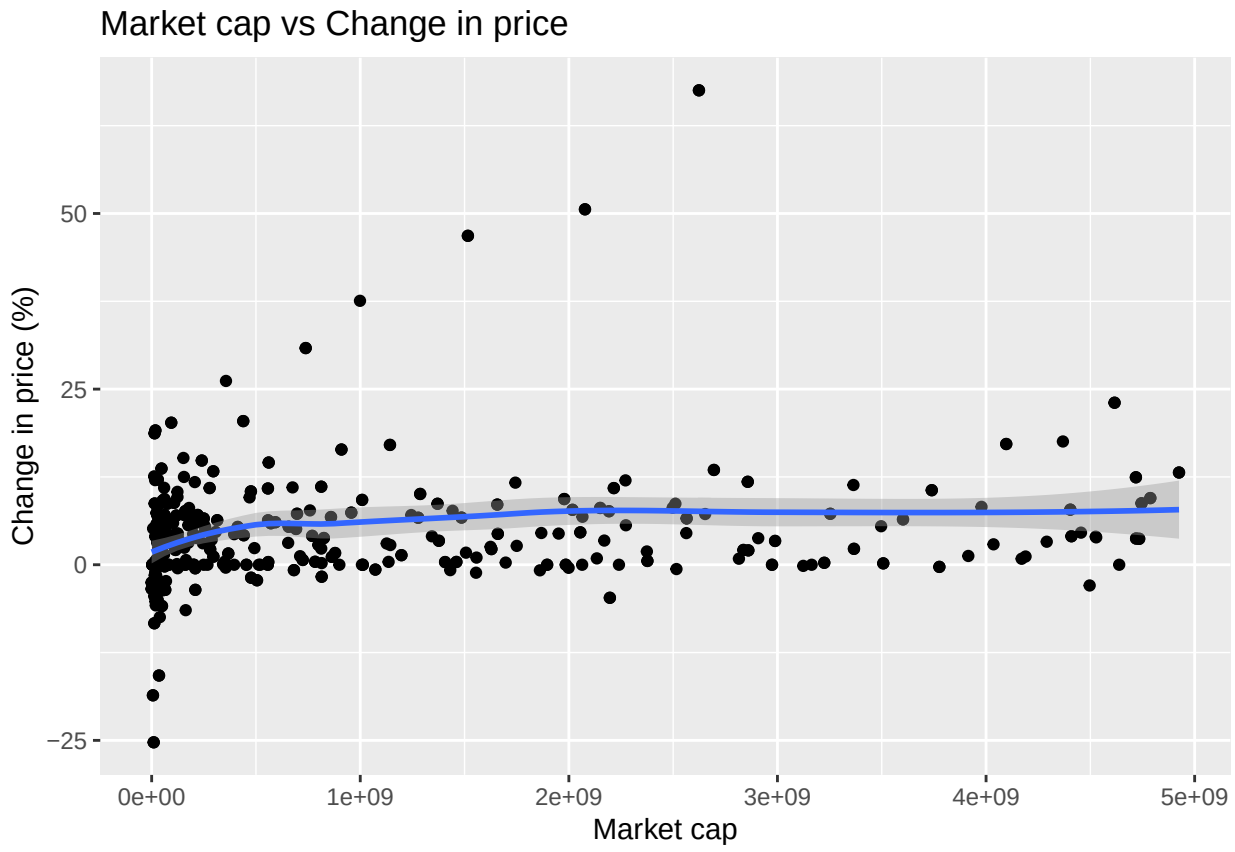

```
ggplot(plotdata1, aes(y=sentiment, x = volume)) +
  geom_point() + geom_smooth() +
  labs(
    title = "Sentiment vs Volume",
    y = "Sentiement score",
    x = "Volume"
  )
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



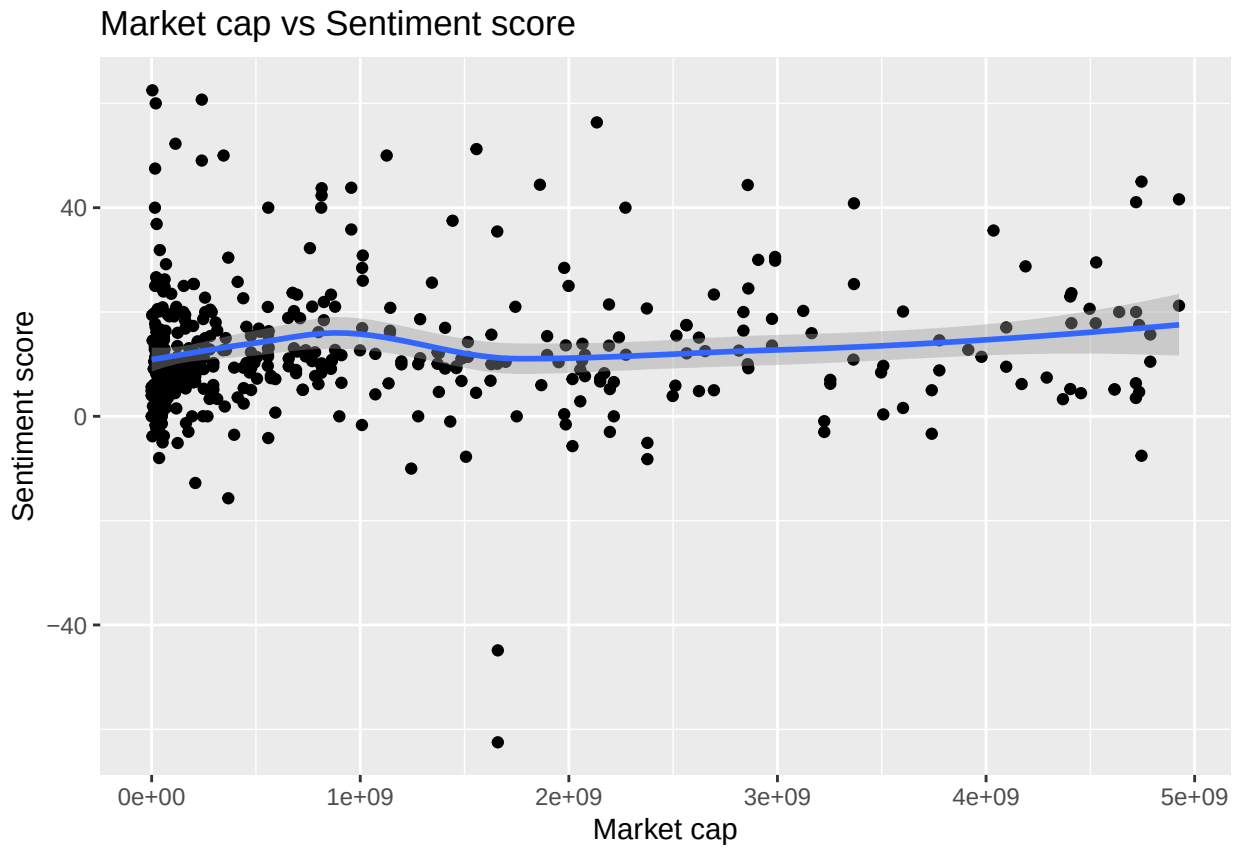

```
plotdata2 <- data %>%
  filter(marketCap < 5000000000)
ggplot(plotdata2, aes(x=marketCap, y = priceChangePercent)) +
  geom_point() + geom_smooth() +
  labs(
    title = "Market cap vs Change in price",
    x = "Market cap",
    y = "Change in price (%)"
  )
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



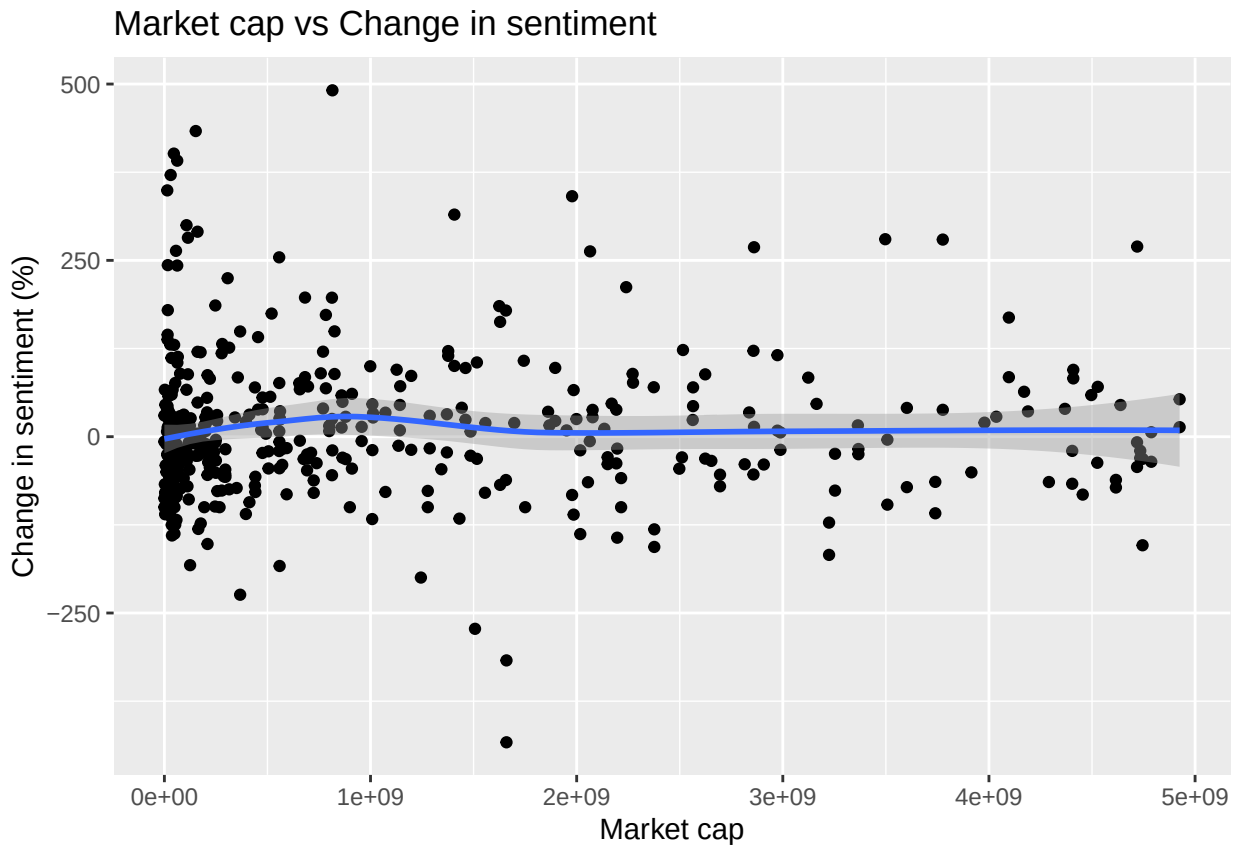

```
ggplot(plotdata2, aes(x=marketCap, y = sentiment)) +
  geom_point() + geom_smooth() +
  labs(
    title = "Market cap vs Sentiment score",
    x = "Market cap",
    y = "Sentiment score"
  )
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
plotdata3 <- data %>%
  filter(marketCap < 5000000000, sentimentChangePercent < 500)
ggplot(plotdata3, aes(x=marketCap, y = sentimentChangePercent)) +
  geom_point() + geom_smooth() +
  labs(
    title = "Market cap vs Change in sentiment",
    x = "Market cap",
    y = "Change in sentiment (%)"
  )
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```


Market cap vs Change in sentiment
Change in sentiment (%)
500
250
0
−250
0e+00
1e+09
2e+09
3e+09
4e+09
5e+09
Market cap