GOOG

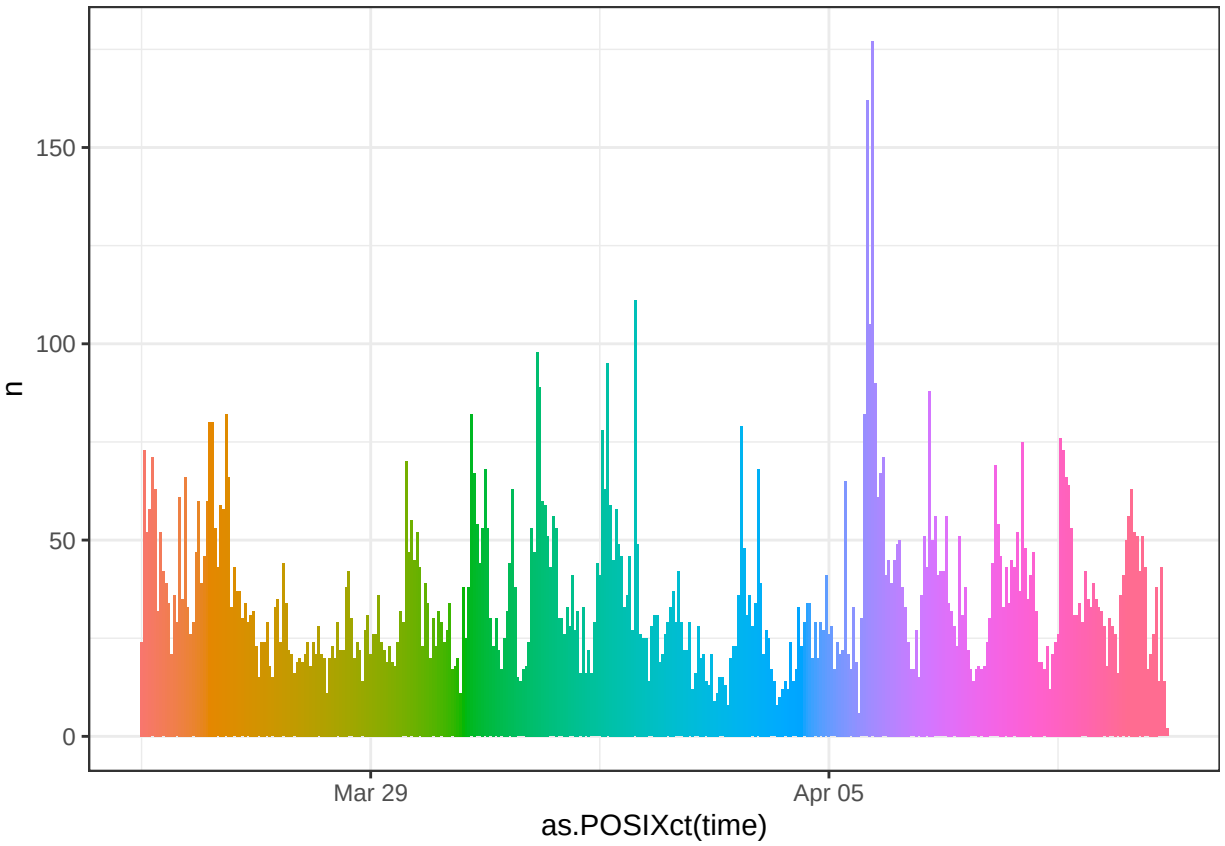
Evan Day

2023-05-08

GOOG

Read Text file and Text Cleanning

The following table shows the tweet number per hour with a barplot.



paste all the text together group by hour, the following table shows an example of the text dataframe.

```
## # A tibble: 6 x 3
## # Groups:   date [1]
```

```
##   date       time                text
##   <date>     <chr>               <chr>
## 1 2021-03-25 2021-03-25 12:00:00 " U S China Tensions Are Getting Real for Inve~
## 2 2021-03-25 2021-03-25 13:00:00 " GOOG Twitter Sentiment on Mar was Positive N~
## 3 2021-03-25 2021-03-25 14:00:00 " GAFAM GOOG AMZN FB AAPL MSFT   Main Tweeted ~
## 4 2021-03-25 2021-03-25 15:00:00 " GAFAM GOOG AMZN FB AAPL MSFT   Pegatron sees~
## 5 2021-03-25 2021-03-25 16:00:00 " Alphabet Inc GOOG surprised the market with ~
## 6 2021-03-25 2021-03-25 17:00:00 " Alphabet Inc GOOG surprised the market with ~

## [1] "there are total 377 observation"
```

Sentiment Data frame with bing, afinn, and nrc

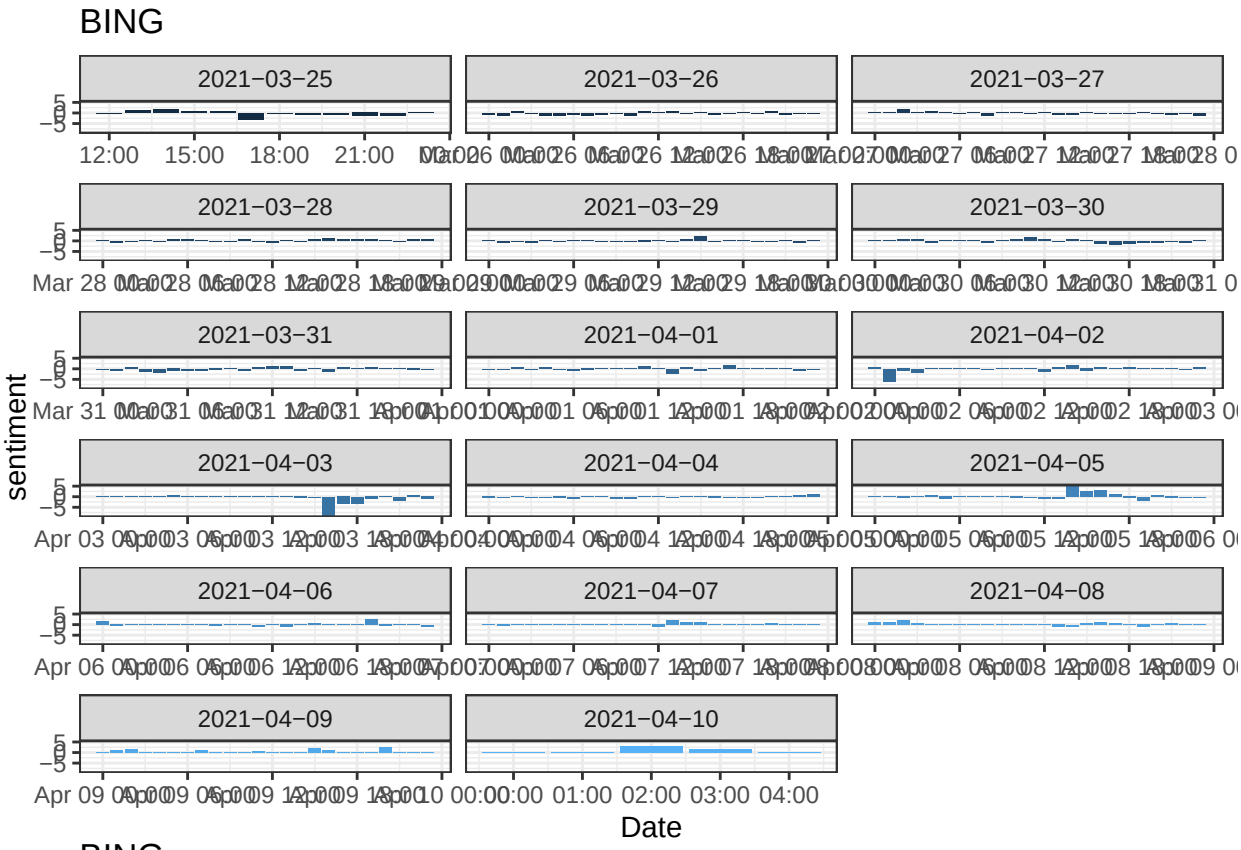
We start with the bing data frame

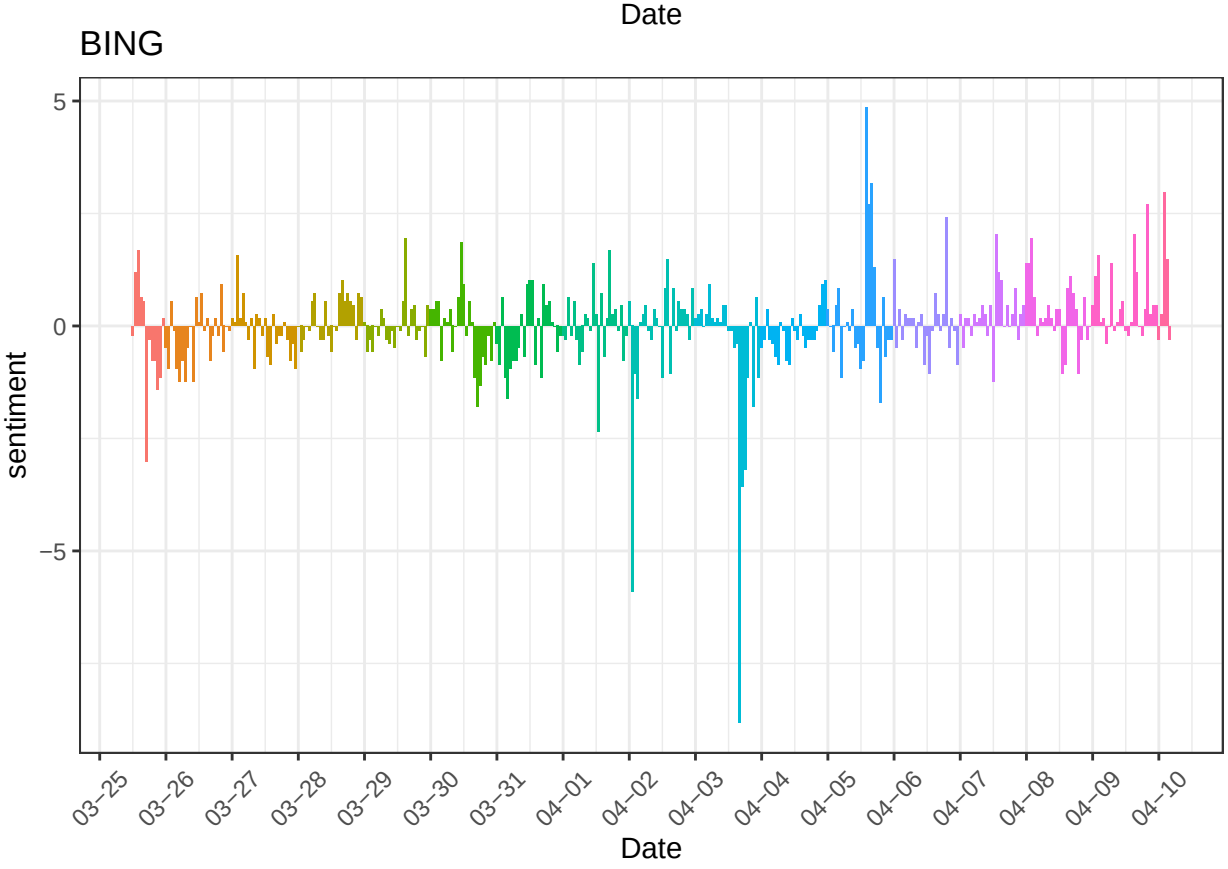
```
## # A tibble: 6 x 3
## # Groups:   date [1]
##   date       time                sentiment
##   <date>     <chr>                   <dbl>
## 1 2021-03-25 2021-03-25 12:00:00        -1
## 2 2021-03-25 2021-03-25 13:00:00        14
## 3 2021-03-25 2021-03-25 14:00:00        19
## 4 2021-03-25 2021-03-25 15:00:00         8
## 5 2021-03-25 2021-03-25 16:00:00         7
## 6 2021-03-25 2021-03-25 17:00:00       -31
```

then, we normalize the sentiment, normalized data has mean = 0 // aother way is rescale to c(-3,3)

```
## # A tibble: 6 x 3
## # Groups:   date [1]
##   date       time                sentiment
##   <date>     <chr>                   <dbl>
## 1 2021-03-25 2021-03-25 12:00:00    -0.204
## 2 2021-03-25 2021-03-25 13:00:00     1.20
## 3 2021-03-25 2021-03-25 14:00:00     1.67
## 4 2021-03-25 2021-03-25 15:00:00     0.638
## 5 2021-03-25 2021-03-25 16:00:00     0.544
## 6 2021-03-25 2021-03-25 17:00:00    -3.01
```

and then, we plot the normalized sentiment against the time.

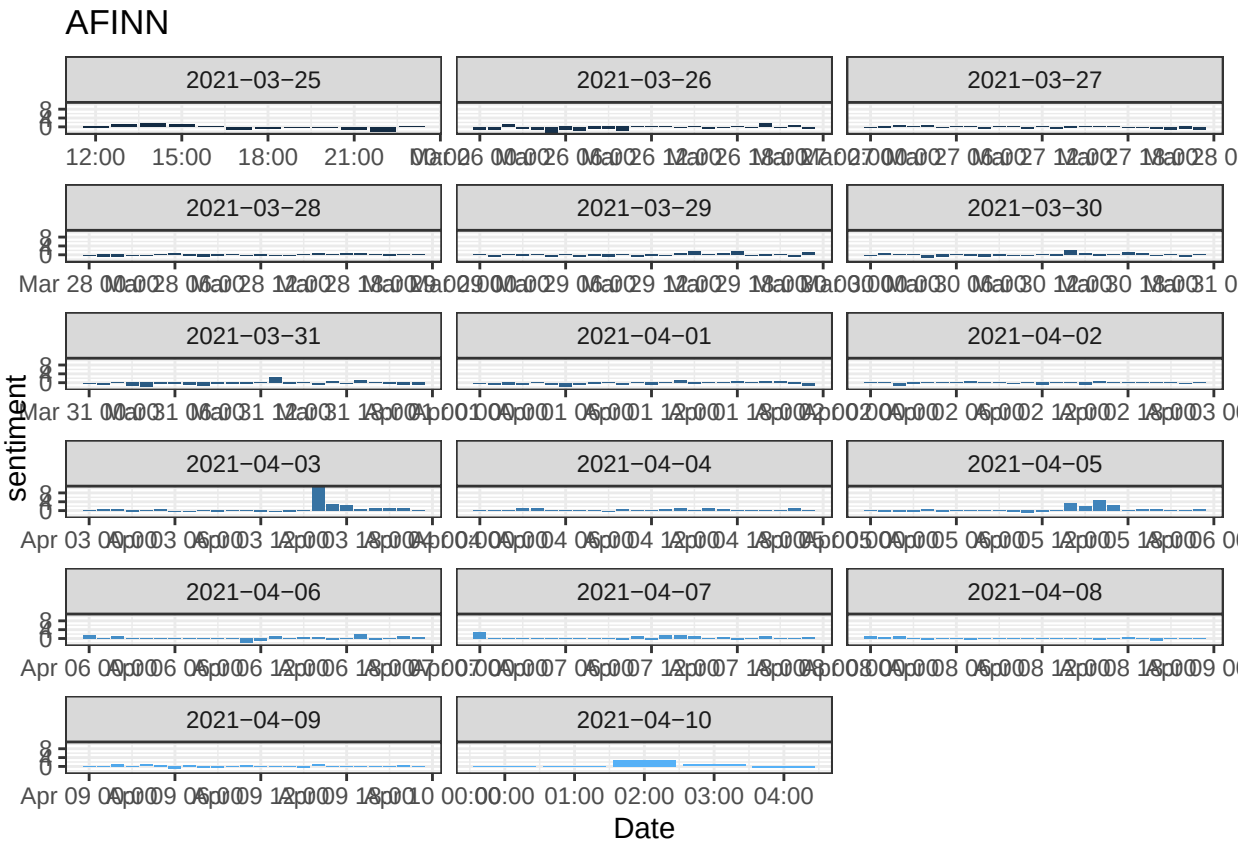


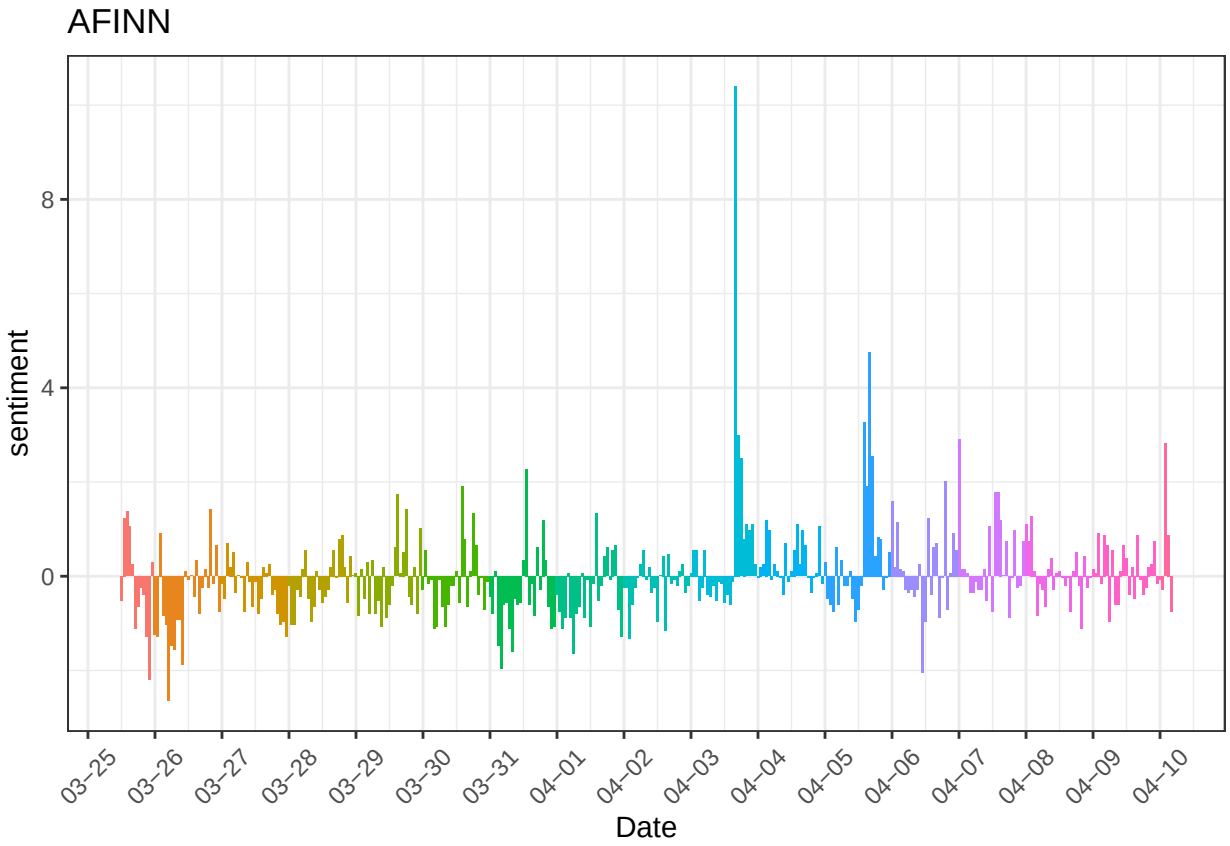


And then, we deal with the afinn sentiment dataframe

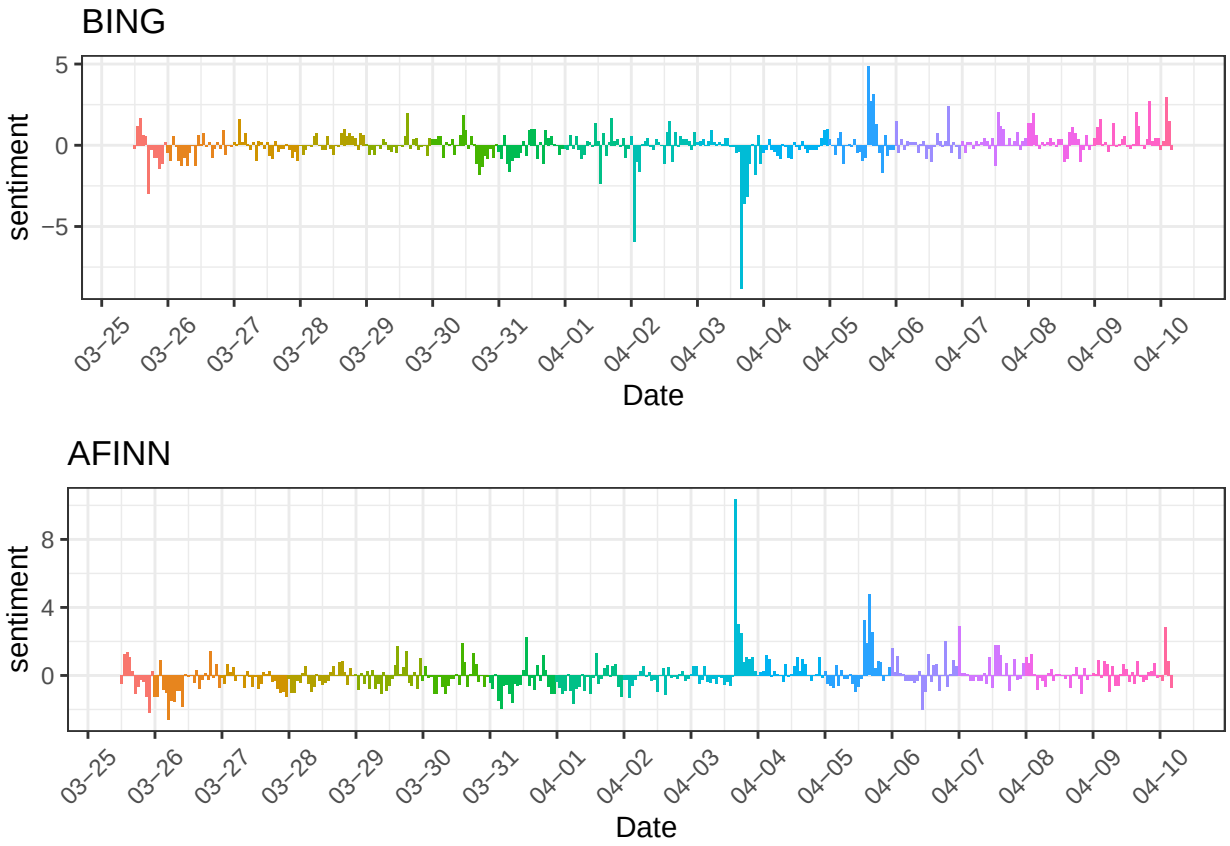
```
## # A tibble: 6 x 3
## # Groups:   date [1]
##   date       time                sentiment
##   <date>     <chr>                   <dbl>
## 1 2021-03-25 2021-03-25 12:00:00    -0.522
## 2 2021-03-25 2021-03-25 13:00:00     1.24
## 3 2021-03-25 2021-03-25 14:00:00     1.37
## 4 2021-03-25 2021-03-25 15:00:00     1.06
## 5 2021-03-25 2021-03-25 16:00:00     0.245
## 6 2021-03-25 2021-03-25 17:00:00    -1.11
```

and then, we plot the normalized sentiment against the time. // Aother method is rescale to c(-3,3)





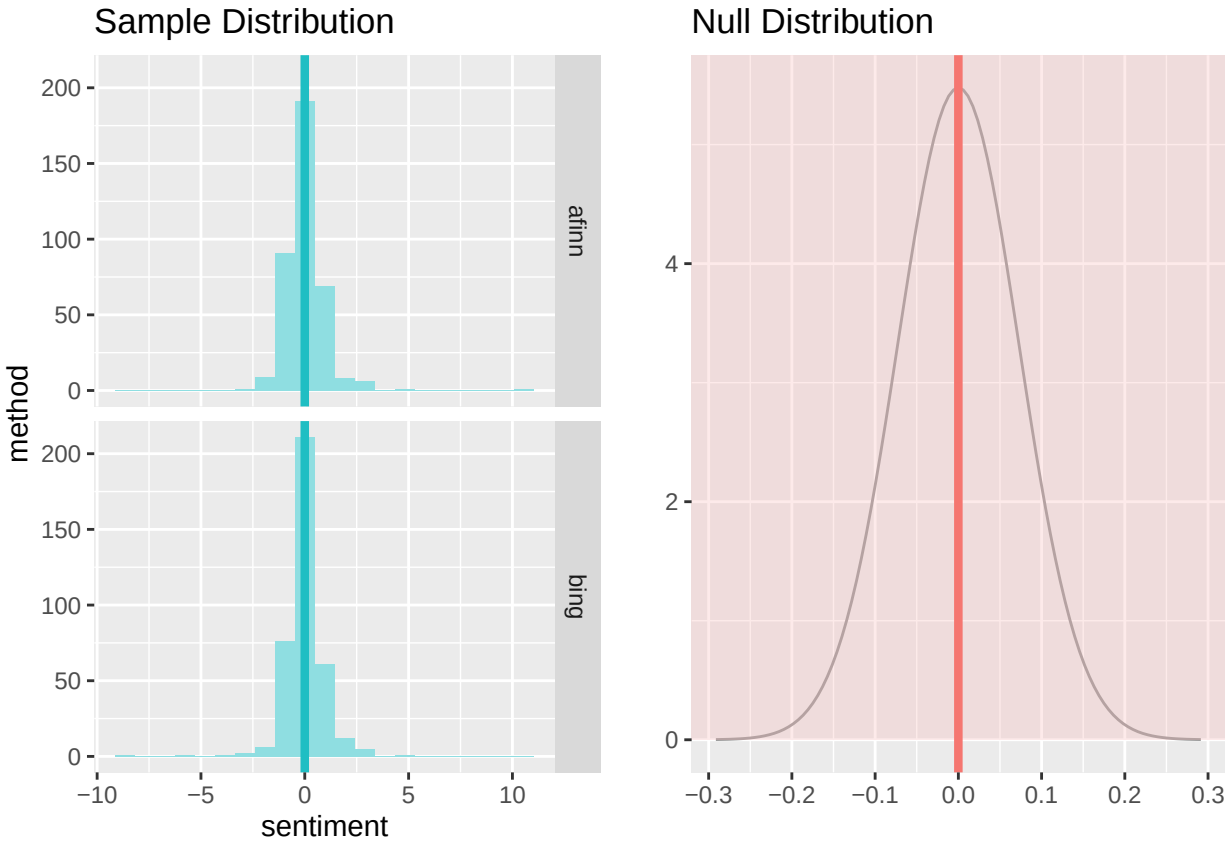
we compare the two sentiment plot together



using t-test to check the whether there is a difference between bing lexicon and afinn lexicon, however the distriibution must be similar. (this is meaningless, because we have already normalize the data, the distributio will be almost the same

```
## Loading required package: BayesFactor

## Loading required package: coda

## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

## ************
## Welcome to BayesFactor 0.9.12-4.3. If you have questions, please contact Richard Morey (richarddmorey
##
## Type BFManual() to open the manual.
## ************

## Warning: Missing null value, set to 0
```

```
## Response variable: numerical
## Explanatory variable: categorical (2 levels)
## n_afinn = 377, y_bar_afinn = 0, s_afinn = 1
## n_bing = 377, y_bar_bing = 0, s_bing = 1
## H0: mu_afinn =  mu_bing
## HA: mu_afinn != mu_bing
## t = 0, df = 376
## p_value = 1
```



we should use the KS-test to check the distribution: as a result, reject the null h0, the distribution are different.

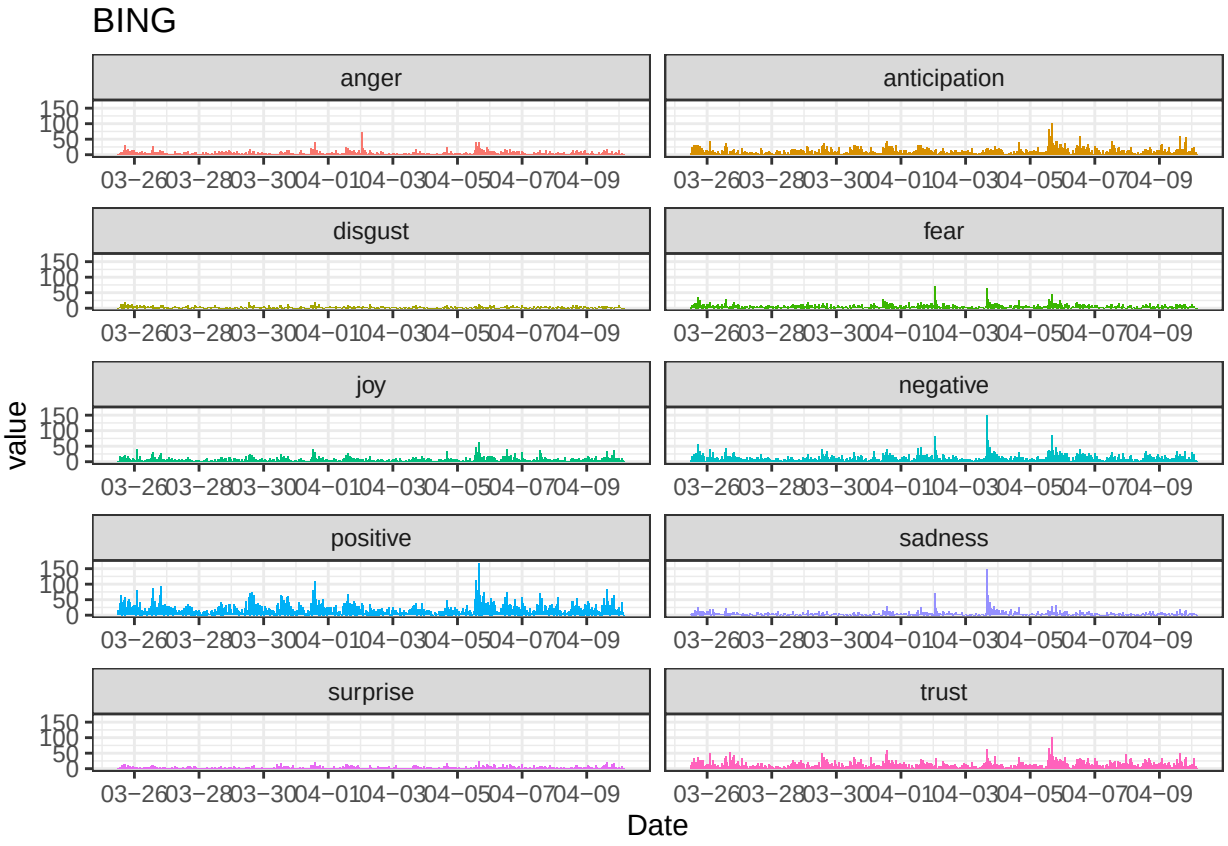
```
## Warning in ks.test(bing_afinn$bing, bing_afinn$afinn, alternative =
## "two.sided"): p-value will be approximate in the presence of ties
```

```
##
##  Two-sample Kolmogorov-Smirnov test
##
## data:  bing_afinn$bing and bing_afinn$afinn
## D = 0.13263, p-value = 0.002637
## alternative hypothesis: two-sided
```

Then, here is the method with nrc lexicon

```
## # A tibble: 6 x 12
## # Groups:   date [1]
```

```
##   date       time       anger anticipation disgust  fear   joy negative positive
##   <date>     <chr>      <dbl>        <dbl>   <dbl> <dbl> <dbl>    <dbl>    <dbl>
## 1 2021-03-25 2021-03-2~     2            5       0     4     2        5       16
## 2 2021-03-25 2021-03-2~     6           23       2     8    17       18       37
## 3 2021-03-25 2021-03-2~     8           32      12    12    15       24       64
## 4 2021-03-25 2021-03-2~     7           23      13    16    14       20       42
## 5 2021-03-25 2021-03-2~    16           30      14    17    17       29       48
## 6 2021-03-25 2021-03-2~    30           31      19    36    20       56       58
## # ... with 3 more variables: sadness <dbl>, surprise <dbl>, trust <dbl>

##
## Attaching package: 'reshape2'

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

## No id variables; using all as measure variables
```



GOOG

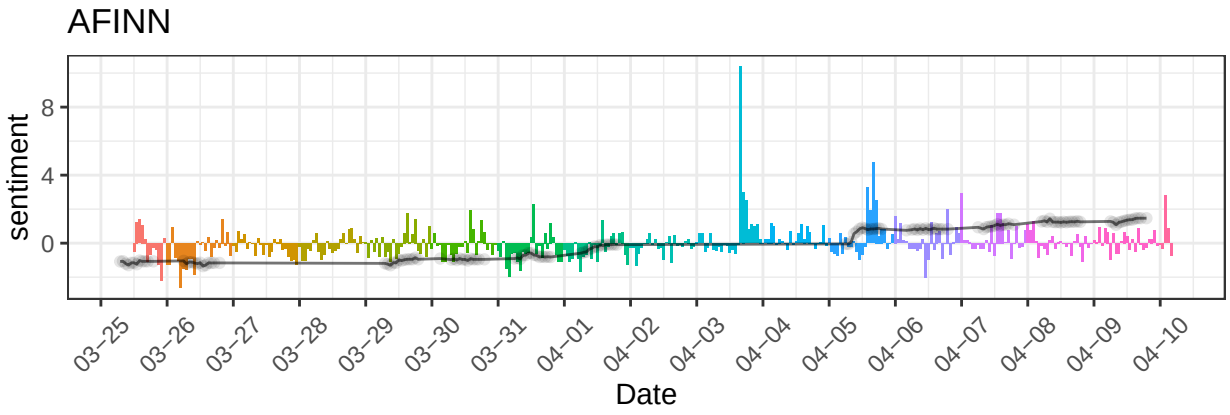
Stock Information

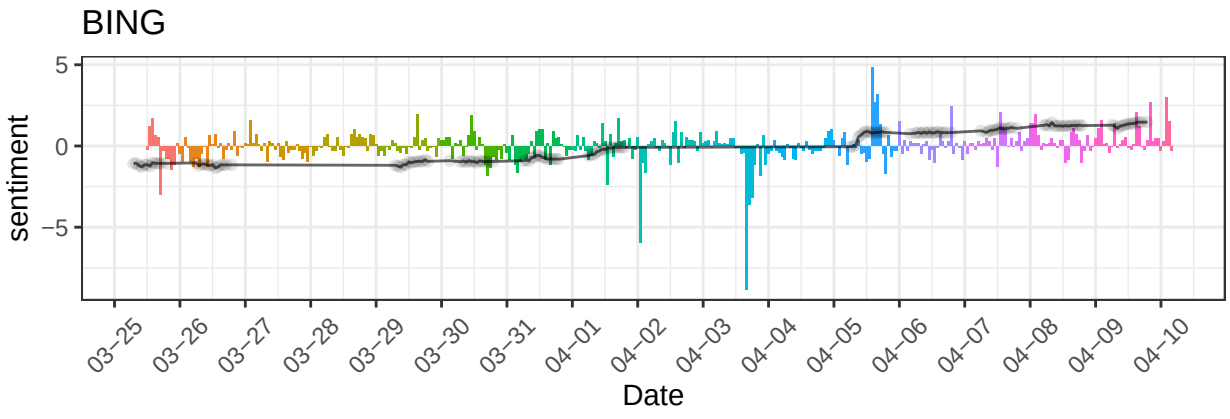
```
## # A tibble: 6 x 2
```

```
##   time                price
##   <chr>               <dbl>
## 1 2021-03-25 07:00:00 2044.
## 2 2021-03-25 08:00:00 2045
## 3 2021-03-25 09:00:00 2032.
## 4 2021-03-25 10:00:00 2023.
## 5 2021-03-25 11:00:00 2036
## 6 2021-03-25 12:00:00 2036.
```

normalize the price data:

```
## # A tibble: 6 x 2
##   time                price
##   <chr>               <dbl>
## 1 2021-03-25 07:00:00 -1.08
## 2 2021-03-25 08:00:00 -1.06
## 3 2021-03-25 09:00:00 -1.20
## 4 2021-03-25 10:00:00 -1.29
## 5 2021-03-25 11:00:00 -1.16
## 6 2021-03-25 12:00:00 -1.15
```





2. Build the model dataframe:

```
## Joining, by = c("datetime", "date")
```

Here we need to deal with several questions: 1. Stock maket open at 9 am and close at 4 pm 2. At the open time, stock market record the XX:30, which is not consistent with sentimennt XX::00 3. At close time, stock market also record some stock price

Separate the dataframe into close data_frame and open data_frame

```
## # A tibble: 6 x 15
##   datetime            price date       time_stock anger anticipation disgust
##   <dttm>              <dbl> <date>     <chr>      <dbl>        <dbl>   <dbl>
## 1 2021-03-25 17:00:00 -1.07 2021-03-25 17:00         30           31      19
## 2 2021-03-25 19:00:00 -1.07 2021-03-25 19:00         18           26      13
## 3 2021-03-26 06:00:00 -1.04 2021-03-26 06:00         10            9       6
## 4 2021-03-26 07:00:00 -1.21 2021-03-26 07:00          6            4       8
## 5 2021-03-26 08:00:00 -1.12 2021-03-26 08:00          7           11       7
## 6 2021-03-26 17:00:00 -1.16 2021-03-26 17:00          8           11       4
## # ... with 8 more variables: fear <dbl>, joy <dbl>, negative <dbl>,
## #   positive <dbl>, sadness <dbl>, surprise <dbl>, trust <dbl>, state <chr>
```

```
## # A tibble: 6 x 15
##   datetime            price date       time_stock anger anticipation disgust
##   <dttm>              <dbl> <date>     <chr>      <dbl>        <dbl>   <dbl>
## 1 2021-03-25 12:00:00 -1.15 2021-03-25 12:00          2            5       0
## 2 2021-03-25 13:00:00 -1.24 2021-03-25 13:00          6           23       2
## 3 2021-03-25 14:00:00 -1.02 2021-03-25 14:00          8           32      12
## 4 2021-03-25 15:00:00 -1.07 2021-03-25 15:00          7           23      13
## 5 2021-03-25 16:00:00 -1.07 2021-03-25 16:00         16           30      14
## 6 2021-03-26 09:00:00 -1.11 2021-03-26 09:00          6            5       2
## # ... with 8 more variables: fear <dbl>, joy <dbl>, negative <dbl>,
## #   positive <dbl>, sadness <dbl>, surprise <dbl>, trust <dbl>, state <chr>
```

GOOG NRC Regression Model result

1. this is the model for total recording

```
##
## Call:
## lm(formula = price ~ anger + anticipation + disgust + fear +
##     joy + negative + positive + sadness + surprise + trust, data = full_nrc)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.92886 -0.80996 -0.00017  0.76922  2.05514
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.04082    0.07923   0.515  0.60727
## anger         0.21053    0.19724   1.067  0.28775
## anticipation  0.57979    0.24598   2.357  0.01990 *
## disgust      -0.20216    0.12124  -1.667  0.09781 .
## fear         -0.37494    0.17701  -2.118  0.03605 *
## joy           0.15078    0.21567   0.699  0.48569
```

```
## negative      0.26257    0.21603   1.215  0.22638
## positive     -0.88148    0.27748  -3.177  0.00186 **
## sadness      -0.09724    0.15358  -0.633  0.52774
## surprise      0.04155    0.12104   0.343  0.73195
## trust         0.25489    0.20165   1.264  0.20848
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9441 on 131 degrees of freedom
## Multiple R-squared:   0.16,  Adjusted R-squared:  0.09586
## F-statistic: 2.495 on 10 and 131 DF,  p-value: 0.008981
```

```
## randomForest 4.7-1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:gridExtra':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

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

```
##
## Attaching package: 'xgboost'
```

```
## The following object is masked from 'package:dplyr':
##
##     slice
```

```
## [1]  train-rmse:0.960872
## [2]  train-rmse:0.841710
## [3]  train-rmse:0.753869
## [4]  train-rmse:0.677950
## [5]  train-rmse:0.615848
## [6]  train-rmse:0.570059
## [7]  train-rmse:0.521409
## [8]  train-rmse:0.488368
## [9]  train-rmse:0.458562
## [10] train-rmse:0.418885
## [11] train-rmse:0.372401
## [12] train-rmse:0.322024
## [13] train-rmse:0.294043
## [14] train-rmse:0.270112
```

```
## [15] train-rmse:0.245364
## [16] train-rmse:0.220303
## [17] train-rmse:0.192379
## [18] train-rmse:0.174749
## [19] train-rmse:0.167681
## [20] train-rmse:0.151293
## [21] train-rmse:0.140558
## [22] train-rmse:0.127906
## [23] train-rmse:0.116733
## [24] train-rmse:0.106982
## [25] train-rmse:0.094697
## [26] train-rmse:0.081497
## [27] train-rmse:0.076820
## [28] train-rmse:0.071709
## [29] train-rmse:0.067276
## [30] train-rmse:0.062575
## [31] train-rmse:0.059739
## [32] train-rmse:0.051959
## [33] train-rmse:0.049990
## [34] train-rmse:0.047158
## [35] train-rmse:0.042242
## [36] train-rmse:0.039307
## [37] train-rmse:0.034730
## [38] train-rmse:0.032091
## [39] train-rmse:0.028531
## [40] train-rmse:0.025453
## [41] train-rmse:0.022097
## [42] train-rmse:0.019859
## [43] train-rmse:0.019382
## [44] train-rmse:0.016639
## [45] train-rmse:0.015927
## [46] train-rmse:0.014035
## [47] train-rmse:0.011939
## [48] train-rmse:0.011220
## [49] train-rmse:0.010181
## [50] train-rmse:0.009171
## [51] train-rmse:0.008270
## [52] train-rmse:0.007619
## [53] train-rmse:0.007113
## [54] train-rmse:0.006503
## [55] train-rmse:0.006208
## [56] train-rmse:0.005690
## [57] train-rmse:0.005118
## [58] train-rmse:0.004818
## [59] train-rmse:0.004382
## [60] train-rmse:0.003819
## [61] train-rmse:0.003583
## [62] train-rmse:0.003497
## [63] train-rmse:0.003052
## [64] train-rmse:0.002766
## [65] train-rmse:0.002428
## [66] train-rmse:0.002295
## [67] train-rmse:0.002035
## [68] train-rmse:0.001974
```

```
## [69] train-rmse:0.001802
## [70] train-rmse:0.001682
## [71] train-rmse:0.001400
## [72] train-rmse:0.001369
## [73] train-rmse:0.001314
## [74] train-rmse:0.001280
## [75] train-rmse:0.001126
## [76] train-rmse:0.000986
## [77] train-rmse:0.000945
## [78] train-rmse:0.000877
## [79] train-rmse:0.000877
## [80] train-rmse:0.000877
## [81] train-rmse:0.000877
## [82] train-rmse:0.000877
## [83] train-rmse:0.000877
## [84] train-rmse:0.000877
## [85] train-rmse:0.000877
## [86] train-rmse:0.000877
## [87] train-rmse:0.000877
## [88] train-rmse:0.000877
## [89] train-rmse:0.000877
## [90] train-rmse:0.000877
## [91] train-rmse:0.000877
## [92] train-rmse:0.000877
## [93] train-rmse:0.000877
## [94] train-rmse:0.000877
## [95] train-rmse:0.000877
## [96] train-rmse:0.000877
## [97] train-rmse:0.000877
## [98] train-rmse:0.000877
## [99] train-rmse:0.000877
## [100]    train-rmse:0.000877
```

2. this is the model for close recording

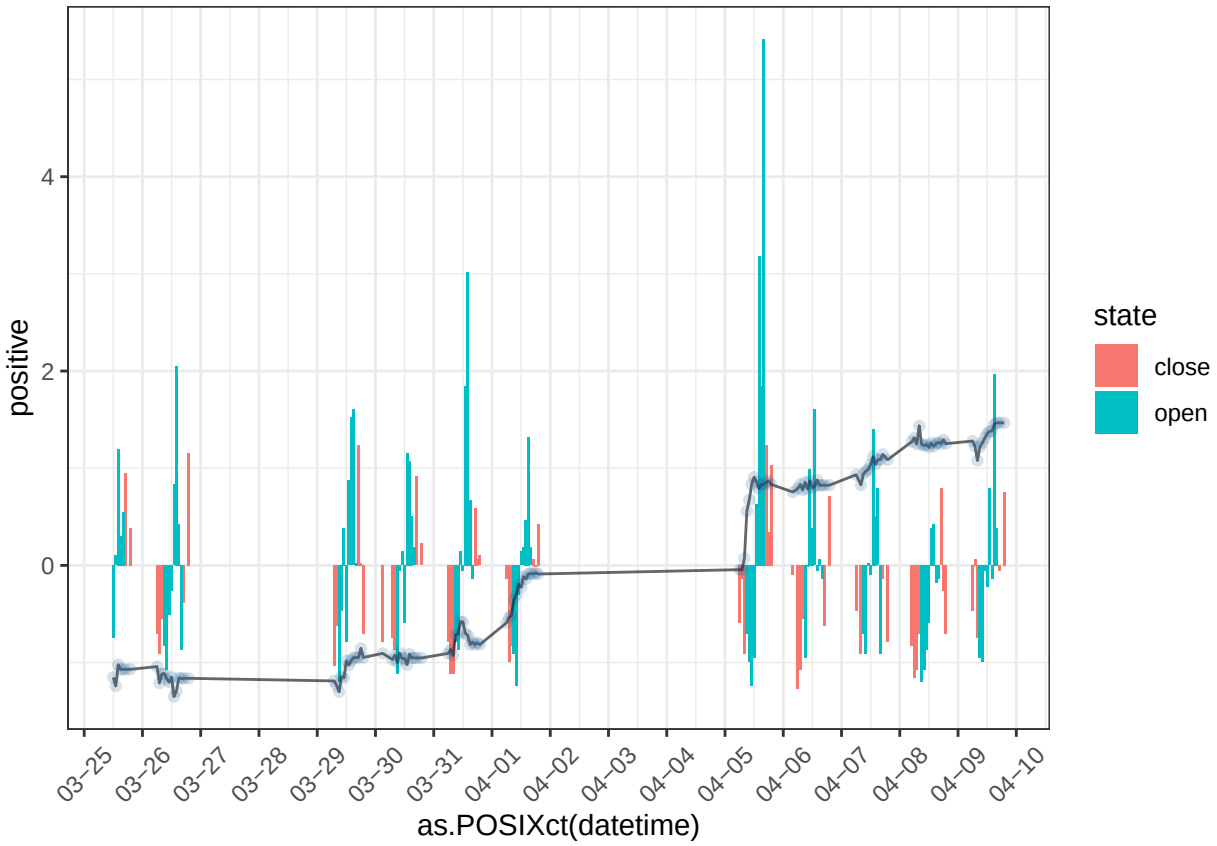
```
##
## Call:
## lm(formula = price ~ anger + anticipation + disgust + fear +
##     joy + negative + positive + sadness + surprise + trust, data = full_nrc[which(full_nrc$state ==
##     "close"), ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.30640 -0.72169 -0.05894  0.61686  1.57350
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.19417    0.13230   1.468   0.1490
## anger         0.45499    0.37220   1.222   0.2278
## anticipation  0.97351    0.38945   2.500   0.0161 *
## disgust      -0.02478    0.21923  -0.113   0.9105
## fear         -0.66733    0.32378  -2.061   0.0450 *
## joy           0.32844    0.38613   0.851   0.3994
## negative     -0.34521    0.43027  -0.802   0.4265
```

```
## positive     -0.72690    0.40344  -1.802   0.0781 .
## sadness       0.03905    0.24702   0.158   0.8751
## surprise      0.28237    0.23794   1.187   0.2414
## trust        -0.01997    0.29098  -0.069   0.9456
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8758 on 46 degrees of freedom
## Multiple R-squared:  0.3512, Adjusted R-squared:  0.2101
## F-statistic:  2.49 on 10 and 46 DF,  p-value: 0.01768
```

3. this is the model for open recording

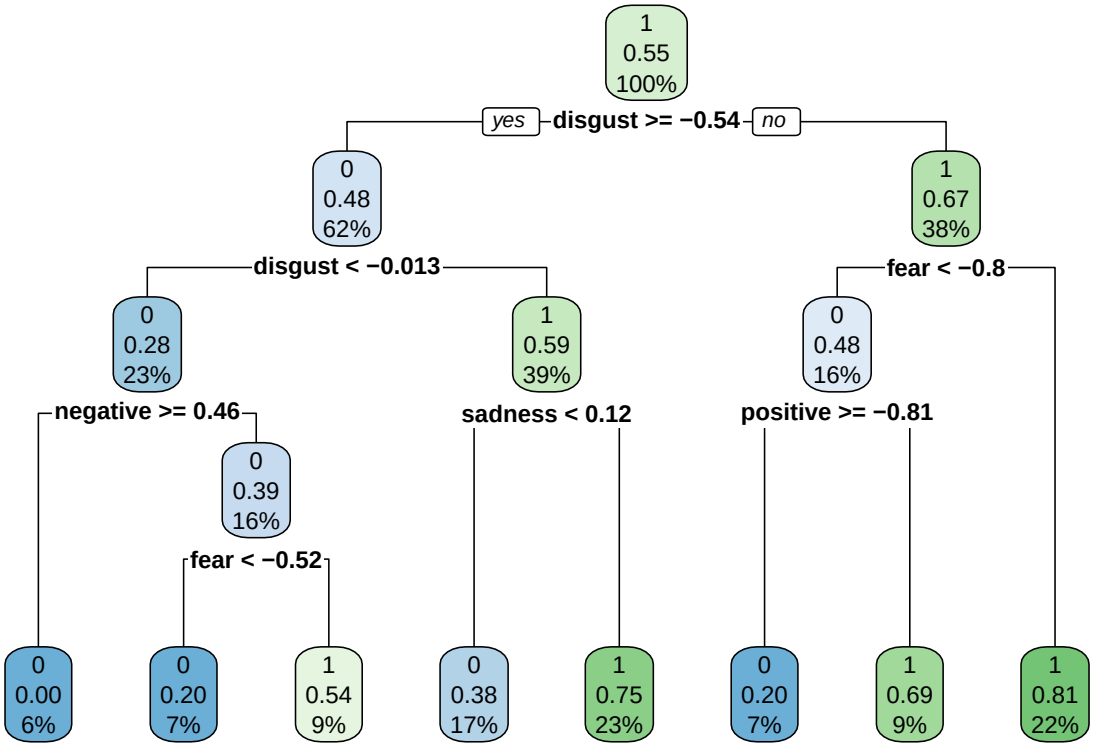
```
##
## Call:
## lm(formula = price ~ anger + anticipation + disgust + fear +
##     joy + negative + positive + sadness + surprise + trust, data = full_nrc[which(full_nrc$state ==
##     "open"), ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.57202 -0.80612  0.05917  0.81955  1.52094
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.03382    0.11006   0.307    0.759
## anger         0.22482    0.25175   0.893    0.375
## anticipation  0.26358    0.35755   0.737    0.463
## disgust      -0.25588    0.16905  -1.514    0.134
## fear         -0.26718    0.24878  -1.074    0.286
## joy           0.21918    0.29300   0.748    0.457
## negative      0.37743    0.26960   1.400    0.166
## positive     -0.94853    0.41171  -2.304    0.024 *
## sadness      -0.13979    0.22049  -0.634    0.528
## surprise     -0.01262    0.14906  -0.085    0.933
## trust         0.46287    0.30235   1.531    0.130
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9929 on 74 degrees of freedom
## Multiple R-squared:  0.1379, Adjusted R-squared:  0.02143
## F-statistic: 1.184 on 10 and 74 DF,  p-value: 0.3155
```

the most relative variable is the trust sentiment, plotting its plot and stock price



NRC Decision Tree

maximum Tree




```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 40 13
##          1 24 65
##
##                Accuracy : 0.7394
##                  95% CI : (0.6592, 0.8094)
##     No Information Rate : 0.5493
##     P-Value [Acc > NIR] : 2.314e-06
##
##                   Kappa : 0.4655
##
##  Mcnemar's Test P-Value : 0.1002
##
##             Sensitivity : 0.6250
##             Specificity : 0.8333
##          Pos Pred Value : 0.7547
##          Neg Pred Value : 0.7303
##              Prevalence : 0.4507
##          Detection Rate : 0.2817
##    Detection Prevalence : 0.3732
##       Balanced Accuracy : 0.7292
##
##        'Positive' Class : 0
##


## [1]  train-logloss:0.662553
```

```
## [2]  train-logloss:0.639739
## [3]  train-logloss:0.619974
## [4]  train-logloss:0.595694
## [5]  train-logloss:0.577895
## [6]  train-logloss:0.560501
## [7]  train-logloss:0.550860
## [8]  train-logloss:0.542462
## [9]  train-logloss:0.527537
## [10] train-logloss:0.521734
## [11] train-logloss:0.512148
## [12] train-logloss:0.503838
## [13] train-logloss:0.496695
## [14] train-logloss:0.489024
## [15] train-logloss:0.483822
## [16] train-logloss:0.480159
## [17] train-logloss:0.476414
## [18] train-logloss:0.474293
## [19] train-logloss:0.468820
## [20] train-logloss:0.465986
## [21] train-logloss:0.461783
## [22] train-logloss:0.460018
## [23] train-logloss:0.456438
## [24] train-logloss:0.453526
## [25] train-logloss:0.451150
## [26] train-logloss:0.448370
## [27] train-logloss:0.445741
## [28] train-logloss:0.443645
## [29] train-logloss:0.442278
## [30] train-logloss:0.441271
## [31] train-logloss:0.440036
## [32] train-logloss:0.438799
## [33] train-logloss:0.437575
## [34] train-logloss:0.436679
## [35] train-logloss:0.435737
## [36] train-logloss:0.434813
## [37] train-logloss:0.433739
## [38] train-logloss:0.432612
## [39] train-logloss:0.431508
## [40] train-logloss:0.430875
## [41] train-logloss:0.429958
## [42] train-logloss:0.428533
## [43] train-logloss:0.427841
## [44] train-logloss:0.427291
## [45] train-logloss:0.426625
## [46] train-logloss:0.425617
## [47] train-logloss:0.425013
## [48] train-logloss:0.424513
## [49] train-logloss:0.423970
## [50] train-logloss:0.423435
## [51] train-logloss:0.422658
## [52] train-logloss:0.421809
## [53] train-logloss:0.421373
## [54] train-logloss:0.420844
## [55] train-logloss:0.420409
```

```
## [56] train-logloss:0.420078
## [57] train-logloss:0.419687
## [58] train-logloss:0.419379
## [59] train-logloss:0.419048
## [60] train-logloss:0.418782
## [61] train-logloss:0.418498
## [62] train-logloss:0.418036
## [63] train-logloss:0.417768
## [64] train-logloss:0.417562
## [65] train-logloss:0.417380
## [66] train-logloss:0.417031
## [67] train-logloss:0.416719
## [68] train-logloss:0.416505
## [69] train-logloss:0.416219
## [70] train-logloss:0.416077
## [71] train-logloss:0.415871
## [72] train-logloss:0.415587
## [73] train-logloss:0.415409
## [74] train-logloss:0.415242
## [75] train-logloss:0.415012
## [76] train-logloss:0.414750
## [77] train-logloss:0.414551
## [78] train-logloss:0.414385
## [79] train-logloss:0.414274
## [80] train-logloss:0.414096
## [81] train-logloss:0.413959
## [82] train-logloss:0.413756
## [83] train-logloss:0.413573
## [84] train-logloss:0.413443
## [85] train-logloss:0.413231
## [86] train-logloss:0.413088
## [87] train-logloss:0.412974
## [88] train-logloss:0.412823
## [89] train-logloss:0.412719
## [90] train-logloss:0.412610
## [91] train-logloss:0.412526
## [92] train-logloss:0.412415
## [93] train-logloss:0.412262
## [94] train-logloss:0.412163
## [95] train-logloss:0.412032
## [96] train-logloss:0.411945
## [97] train-logloss:0.411857
## [98] train-logloss:0.411773
## [99] train-logloss:0.411684
## [100]    train-logloss:0.411583
```

bing and Afinn regression

```
## Joining, by = "word"
## Joining, by = c("datetime", "date")

## Warning in log(price): NaNs produced

##
```

```
## Call:
## lm(formula = log(price) ~ negative + positive, data = full_bing)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.62678 -0.17104  0.07464  0.23398  0.39376
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.007818   0.076561   0.102    0.919
## negative    -0.002780   0.006310  -0.441    0.661
## positive     0.001294   0.004455   0.291    0.772
##
## Residual standard error: 0.4044 on 63 degrees of freedom
##   (76 observations deleted due to missingness)
## Multiple R-squared:  0.003161,   Adjusted R-squared:  -0.02848
## F-statistic: 0.09989 on 2 and 63 DF,  p-value: 0.9051
```

```
##
## Call:
## lm(formula = price ~ negative + positive, data = full_bing_close)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.3344 -0.8276 -0.1432  0.9194  1.5819
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.09806    0.22708   0.432  0.66760
## negative    -0.03840    0.01422  -2.701  0.00921 **
## positive     0.03223    0.01781   1.810  0.07590 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9419 on 54 degrees of freedom
## Multiple R-squared:  0.1191, Adjusted R-squared:  0.08642
## F-statistic: 3.649 on 2 and 54 DF,  p-value: 0.03263
```

```
##
## Call:
## lm(formula = price ~ negative + positive, data = full_bing_open)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.3958 -1.0385 -0.1054  0.9578  1.4124
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.076487   0.186697   0.410    0.683
## negative    -0.012294   0.013003  -0.945    0.347
## positive     0.008240   0.009935   0.829    0.409
##
## Residual standard error: 1.01 on 82 degrees of freedom
## Multiple R-squared:  0.01118,    Adjusted R-squared:  -0.01294
```

```
## F-statistic: 0.4636 on 2 and 82 DF,  p-value: 0.6306
```

```
## Joining, by = c("datetime", "date")
```

```
## Warning in log(price): NaNs produced
```

```
##
## Call:
## lm(formula = log(price) ~ sentiment, data = full_afinn)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.62394 -0.17043  0.07208  0.23020  0.40258
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.003193   0.050370  -0.063    0.950
## sentiment   -0.018030   0.047181  -0.382    0.704
##
## Residual standard error: 0.4014 on 64 degrees of freedom
##   (76 observations deleted due to missingness)
## Multiple R-squared:  0.002277,   Adjusted R-squared:  -0.01331
## F-statistic: 0.146 on 1 and 64 DF,  p-value: 0.7036
```

```
##
## Call:
## lm(formula = price ~ sentiment, data = full_afinn_close)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.4335 -0.8305 -0.1529  0.8843  1.5283
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.08883    0.12815   0.693   0.4911
## sentiment    0.34655    0.15567   2.226   0.0301 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9524 on 55 degrees of freedom
## Multiple R-squared:  0.08266,    Adjusted R-squared:  0.06598
## F-statistic: 4.956 on 1 and 55 DF,  p-value: 0.03012
```

```
##
## Call:
## lm(formula = price ~ sentiment, data = full_afinn_open)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.4321 -0.9995 -0.1129  0.9364  1.4890
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)  0.02773    0.10913   0.254    0.800
## sentiment    0.13109    0.10479   1.251    0.214
##
## Residual standard error: 1 on 83 degrees of freedom
## Multiple R-squared:  0.0185, Adjusted R-squared:  0.006678
## F-statistic: 1.565 on 1 and 83 DF,  p-value: 0.2145
```

Predict the following days

```
## # A tibble: 6 x 3
## # Groups:   date [1]
##   date       time                text
##   <date>     <chr>               <chr>
## 1 2021-04-09 2021-04-09 03:00:00 " REQUEST SixTONES FridayLivestream Request NA~
## 2 2021-04-09 2021-04-09 04:00:00 " Alphabet Inc GOOG surprised the market with ~
## 3 2021-04-09 2021-04-09 05:00:00 " What is it ahead for US Tech Giants Check wh~
## 4 2021-04-09 2021-04-09 06:00:00 " AMZN GOOG FB NFLX AAPL took market share fro~
## 5 2021-04-09 2021-04-09 07:00:00 " on the goog   They know how to play LONG gam~
## 6 2021-04-09 2021-04-09 08:00:00 " Alphabet Inc GOOG surprised the market with ~
```

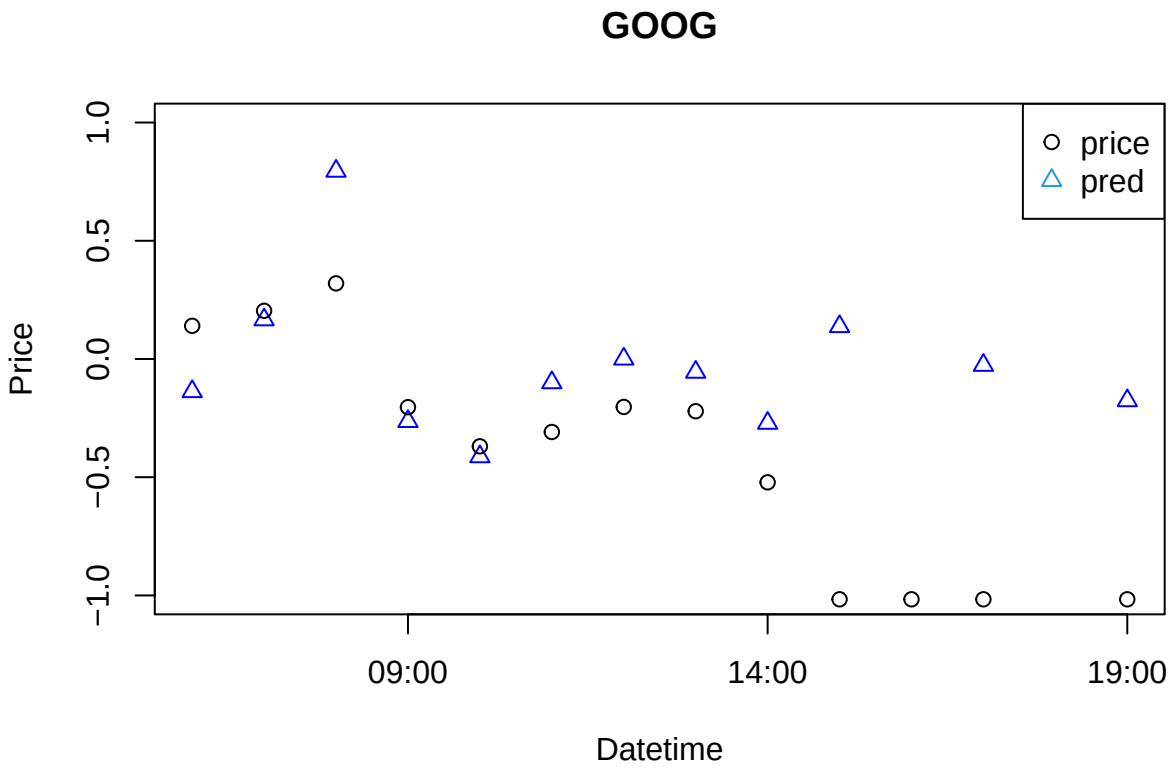
```
## [1] "there are total 194 observation"
```

```
## Joining, by = "word"
## Joining, by = "word"
## `summarise()` has grouped output by 'date'. You can override using the
## `.groups` argument.
## Joining, by = "word"
## `summarise()` has grouped output by 'date'. You can override using the
## `.groups` argument.
## Joining, by = "word"
## Joining, by = c("datetime", "date")
```

```
## # A tibble: 6 x 15
##   datetime             price date       time_stock anger anticipation disgust
##   <dttm>               <dbl> <date>     <chr>      <dbl>        <dbl>   <dbl>
## 1 2021-04-09 06:00:00 -0.178 2021-04-09 06:00          2           11       2
## 2 2021-04-09 07:00:00 -0.509 2021-04-09 07:00          0           22       0
## 3 2021-04-09 08:00:00 -1.39  2021-04-09 08:00          2            7       1
## 4 2021-04-09 17:00:00  0.960 2021-04-09 17:00          6           17       3
## 5 2021-04-09 19:00:00  0.960 2021-04-09 19:00         11           30       3
## 6 2021-04-12 06:00:00  0.210 2021-04-12 06:00          2           11       2
## # ... with 8 more variables: fear <dbl>, joy <dbl>, negative <dbl>,
## #   positive <dbl>, sadness <dbl>, surprise <dbl>, trust <dbl>, state <chr>
```

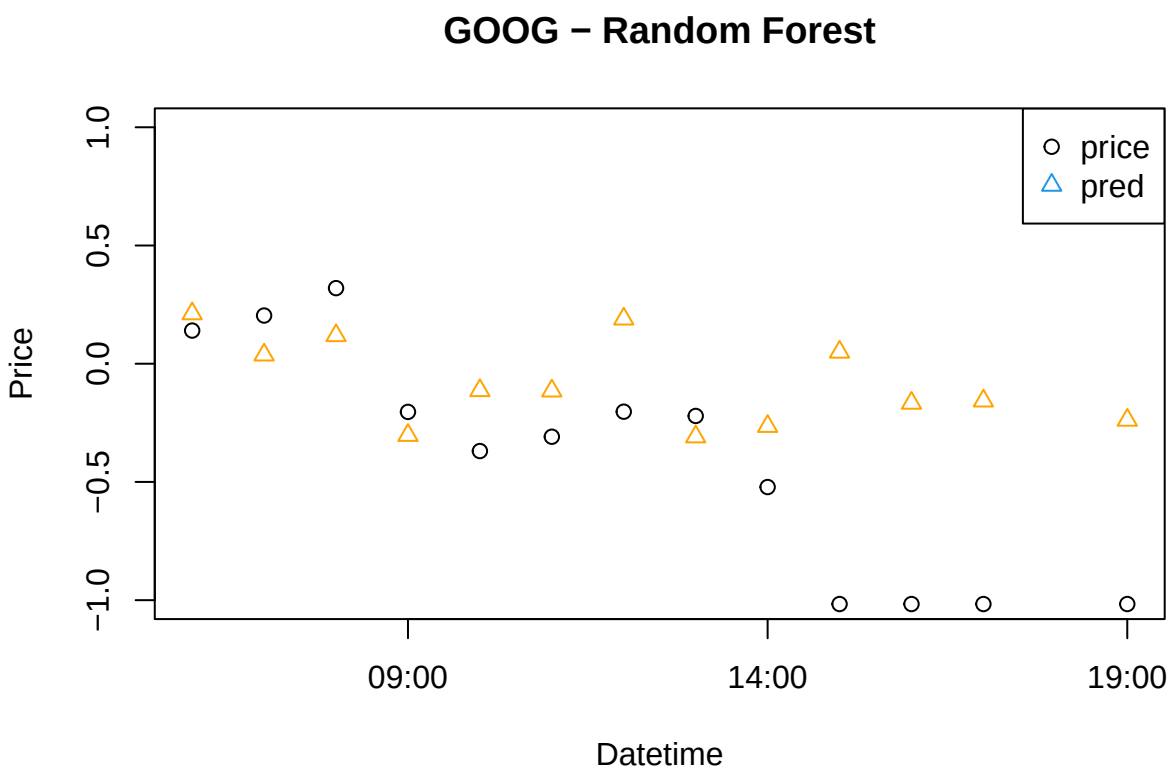
```
## # A tibble: 6 x 15
##   datetime               price date       time_stock anger anticipation disgust
##   <dttm>                 <dbl> <date>     <chr>      <dbl>        <dbl>   <dbl>
## 1 2021-04-09 09:00:00 -0.536    2021-04-09 09:00          8            5       6
## 2 2021-04-09 10:00:00 -0.296    2021-04-09 10:00          3            4       2
## 3 2021-04-09 11:00:00  0.000988 2021-04-09 11:00          5            8       5
## 4 2021-04-09 12:00:00  0.282    2021-04-09 12:00          8           16       4
## 5 2021-04-09 13:00:00  0.425    2021-04-09 13:00         16           18       5
```

```
## 6 2021-04-09 14:00:00  0.448    2021-04-09 14:00         11           18       7
## # ... with 8 more variables: fear <dbl>, joy <dbl>, negative <dbl>,
## #   positive <dbl>, sadness <dbl>, surprise <dbl>, trust <dbl>, state <chr>

## Joining, by = "word"
## Joining, by = c("datetime", "date")
## Joining, by = c("datetime", "date")
```



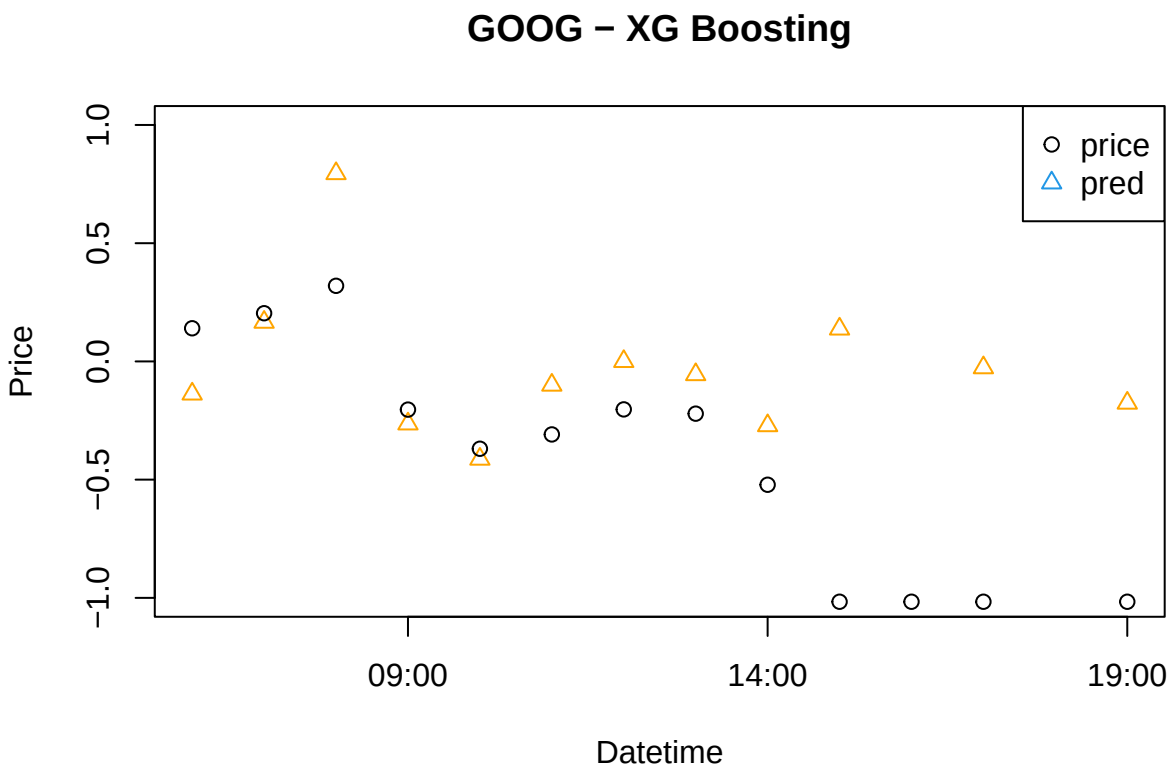

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction 0 1
##          0 4 2
##          1 5 2
##
##                Accuracy : 0.4615
##                  95% CI : (0.1922, 0.7487)
##     No Information Rate : 0.6923
##     P-Value [Acc > NIR] : 0.9787
##
##                   Kappa : -0.046
##
##  Mcnemar's Test P-Value : 0.4497
##
##             Sensitivity : 0.4444
##             Specificity : 0.5000
##          Pos Pred Value : 0.6667
##          Neg Pred Value : 0.2857
##              Prevalence : 0.6923
```

```
##          Detection Rate : 0.3077
##    Detection Prevalence : 0.4615
##       Balanced Accuracy : 0.4722
##
##        'Positive' Class : 0
##
```




```
## Warning in confusionMatrix.default(factor(round(abs(pred_cart))),
## full_nrc2$trend): Levels are not in the same order for reference and data.
## Refactoring data to match.

## Confusion Matrix and Statistics
##
##           Reference
## Prediction 0 1
##          0 9 4
##          1 0 0
##
##                Accuracy : 0.6923
##                  95% CI : (0.3857, 0.9091)
##     No Information Rate : 0.6923
##     P-Value [Acc > NIR] : 0.6310
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : 0.1336
##
##             Sensitivity : 1.0000
##             Specificity : 0.0000
```

```
##          Pos Pred Value : 0.6923
##          Neg Pred Value :    NaN
##              Prevalence : 0.6923
##          Detection Rate : 0.6923
##    Detection Prevalence : 1.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : 0
##

##  [1] 1 1 0 0 1 1 0 0 0 0 0 0 0 0
## Levels: 0 1
```




```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction 0 1
##          0 6 2
##          1 3 2
##
##                Accuracy : 0.6154
##                  95% CI : (0.3158, 0.8614)
##     No Information Rate : 0.6923
##     P-Value [Acc > NIR] : 0.8184
##
##                   Kappa : 0.1558
##
##  Mcnemar's Test P-Value : 1.0000
##
```

```
##             Sensitivity : 0.6667
##             Specificity : 0.5000
##          Pos Pred Value : 0.7500
##          Neg Pred Value : 0.4000
##              Prevalence : 0.6923
##          Detection Rate : 0.4615
##    Detection Prevalence : 0.6154
##       Balanced Accuracy : 0.5833
##
##        'Positive' Class : 0
##
```