Viral articles(P#3)

Hyunpyo Kim

Introduction

This is a model for expecting viral articles, which share more than 1,400. The data set has 39,644 obersvations and 36 variables(except 'URL') which might explain the shares of articles.

```r
library(mosaic)
library(tidyverse)
library(class)
library(foreach)
library(knitr)
library(kableExtra)

news = read.csv("../data/online_news.csv")
head(news, 5)
```

```
##                                                              url
## 1   http://mashable.com/2013/01/07/amazon-instant-video-browser/
## 2    http://mashable.com/2013/01/07/ap-samsung-sponsored-tweets/
## 3 http://mashable.com/2013/01/07/apple-40-billion-app-downloads/
## 4       http://mashable.com/2013/01/07/astronaut-notre-dame-bcs/
## 5               http://mashable.com/2013/01/07/att-u-verse-apps/
##   n_tokens_title n_tokens_content num_hrefs num_self_hrefs num_imgs
## 1             12              219         4              2        1
## 2              9              255         3              1        1
## 3              9              211         3              1        1
## 4              9              531         9              0        1
## 5             13             1072        19             19       20
##   num_videos average_token_length num_keywords data_channel_is_lifestyle
## 1          0             4.680365            5                         0
## 2          0             4.913725            4                         0
## 3          0             4.393365            6                         0
## 4          0             4.404896            7                         0
## 5          0             4.682836            7                         0
##   data_channel_is_entertainment data_channel_is_bus data_channel_is_socmed
## 1                             1                   0                      0
## 2                             0                   1                      0
## 3                             0                   1                      0
## 4                             1                   0                      0
## 5                             0                   0                      0
##   data_channel_is_tech data_channel_is_world self_reference_min_shares
## 1                    0                     0                       496
## 2                    0                     0                         0
## 3                    0                     0                       918
## 4                    0                     0                         0
## 5                    1                     0                       545
##   self_reference_max_shares self_reference_avg_sharess weekday_is_monday
## 1                       496                    496.000                 1
## 2                         0                      0.000                 1
```

```
## 3                       918                  918.000                1
## 4                         0                    0.000                1
## 5                     16000                 3151.158                1
##   weekday_is_tuesday weekday_is_wednesday weekday_is_thursday
## 1                  0                    0                   0
## 2                  0                    0                   0
## 3                  0                    0                   0
## 4                  0                    0                   0
## 5                  0                    0                   0
##   weekday_is_friday weekday_is_saturday weekday_is_sunday is_weekend
## 1                 0                   0                 0          0
## 2                 0                   0                 0          0
## 3                 0                   0                 0          0
## 4                 0                   0                 0          0
## 5                 0                   0                 0          0
##   global_rate_positive_words global_rate_negative_words
## 1                 0.04566210                0.013698630
## 2                 0.04313725                0.015686275
## 3                 0.05687204                0.009478673
## 4                 0.04143126                0.020715631
## 5                 0.07462687                0.012126866
##   avg_positive_polarity min_positive_polarity max_positive_polarity
## 1             0.3786364            0.10000000                   0.7
## 2             0.2869146            0.03333333                   0.7
## 3             0.4958333            0.10000000                   1.0
## 4             0.3859652            0.13636364                   0.8
## 5             0.4111274            0.03333333                   1.0
##   avg_negative_polarity min_negative_polarity max_negative_polarity
## 1            -0.3500000                -0.600            -0.2000000
## 2            -0.1187500                -0.125            -0.1000000
## 3            -0.4666667                -0.800            -0.1333333
## 4            -0.3696970                -0.600            -0.1666667
## 5            -0.2201923                -0.500            -0.0500000
##   title_subjectivity title_sentiment_polarity abs_title_sentiment_polarity
## 1          0.5000000               -0.1875000                    0.1875000
## 2          0.0000000                0.0000000                    0.0000000
## 3          0.0000000                0.0000000                    0.0000000
## 4          0.0000000                0.0000000                    0.0000000
## 5          0.4545455                0.1363636                    0.1363636
##   shares
## 1    593
## 2    711
## 3   1500
## 4   1200
## 5    505
```

```r
ncol(news)
```

```
## [1] 38
```

First, I make a linear regression model by expecting the number of shares and determining whether the artile is viral or not.

Second, I make a logit model by rogit regression with a “viral” variable, then show which model is better in expecting viral articles.

Data modifying

First, I delete the "URL" variable which is not related to the modelling. Some variables, such as "n_tokens_content"(Number of words in the content), "average_token_length"(Average length of the words in the content), should not be zero, because there must be some words in the article. Thus, I think the data with zero "n_tokens_content" are imperfect, and those data also are zero in some variables such as "average_token_length". Thus, I delete those data, then there are 38,463 observations.

```r
news = subset(news, select = -c(url))
summary(news$n_tokens_content)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0   246.0   409.0   546.5   716.0  8474.0
```

```r
news_0_content = news[which(news$n_tokens_content==0),]
summary(news_0_content)
```

```
##  n_tokens_title  n_tokens_content   num_hrefs num_self_hrefs
##  Min.   : 5.00   Min.   :0        Min.   :0   Min.   :0
##  1st Qu.:10.00   1st Qu.:0        1st Qu.:0   1st Qu.:0
##  Median :11.00   Median :0        Median :0   Median :0
##  Mean   :10.93   Mean   :0        Mean   :0   Mean   :0
##  3rd Qu.:12.00   3rd Qu.:0        3rd Qu.:0   3rd Qu.:0
##  Max.   :17.00   Max.   :0        Max.   :0   Max.   :0
##     num_imgs         num_videos      average_token_length  num_keywords
##  Min.   :  0.000   Min.   : 0.0000   Min.   :0            Min.   : 1.000
##  1st Qu.:  0.000   1st Qu.: 0.0000   1st Qu.:0            1st Qu.: 6.000
##  Median :  0.000   Median : 1.0000   Median :0            Median : 7.000
##  Mean   :  3.928   Mean   : 0.7968   Mean   :0            Mean   : 7.509
##  3rd Qu.:  1.000   3rd Qu.: 1.0000   3rd Qu.:0            3rd Qu.: 9.000
##  Max.   :100.000   Max.   :24.0000   Max.   :0            Max.   :10.000
##  data_channel_is_lifestyle data_channel_is_entertainment
##  Min.   :0.00000           Min.   :0.0000
##  1st Qu.:0.00000           1st Qu.:0.0000
##  Median :0.00000           Median :0.0000
##  Mean   :0.01863           Mean   :0.1702
##  3rd Qu.:0.00000           3rd Qu.:0.0000
##  Max.   :1.00000           Max.   :1.0000
##  data_channel_is_bus data_channel_is_socmed data_channel_is_tech
##  Min.   :0.00000     Min.   :0.00000       Min.   :0.00000
##  1st Qu.:0.00000     1st Qu.:0.00000       1st Qu.:0.00000
##  Median :0.00000     Median :0.00000       Median :0.00000
##  Mean   :0.01947     Mean   :0.01016       Mean   :0.01778
##  3rd Qu.:0.00000     3rd Qu.:0.00000       3rd Qu.:0.00000
##  Max.   :1.00000     Max.   :1.00000       Max.   :1.00000
##  data_channel_is_world self_reference_min_shares self_reference_max_shares
##  Min.   :0.0000        Min.   :0                 Min.   :0
##  1st Qu.:0.0000        1st Qu.:0                 1st Qu.:0
##  Median :0.0000        Median :0                 Median :0
##  Mean   :0.2193        Mean   :0                 Mean   :0
##  3rd Qu.:0.0000        3rd Qu.:0                 3rd Qu.:0
##  Max.   :1.0000        Max.   :0                 Max.   :0
##  self_reference_avg_sharess weekday_is_monday weekday_is_tuesday
```

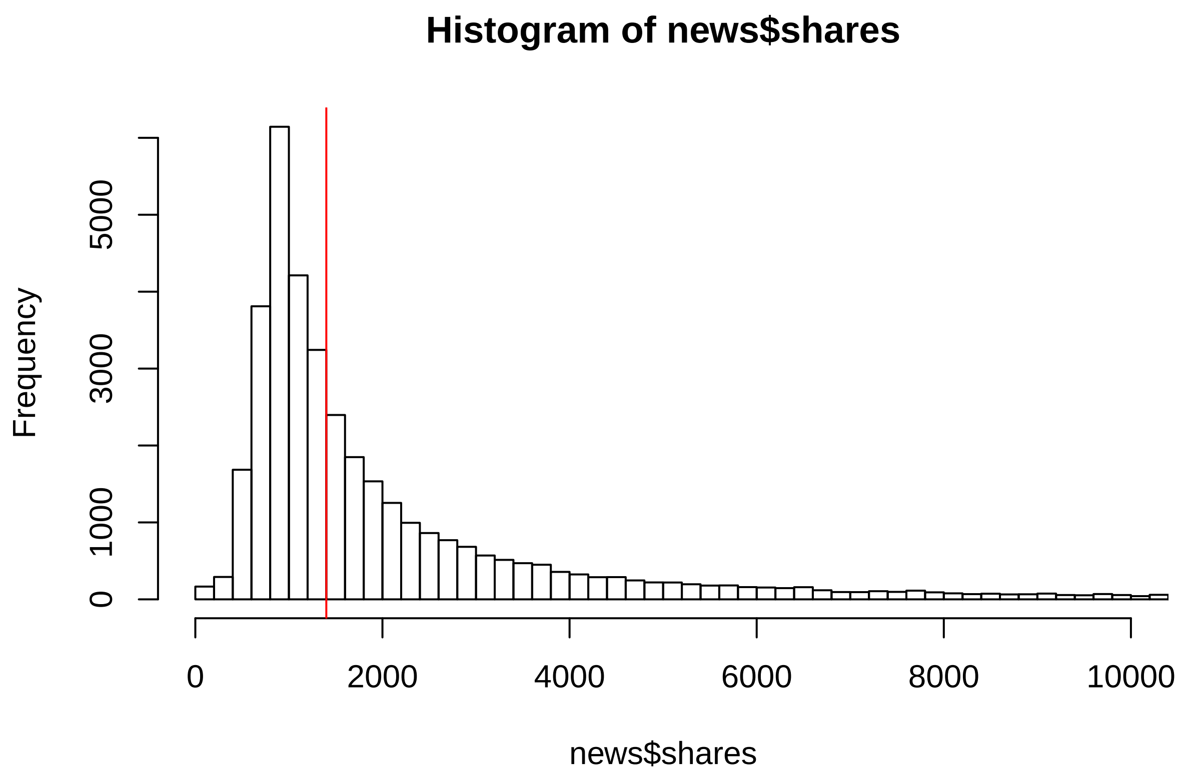
```
##  Min.   :0                   Min.   :0.0000    Min.   :0.0000
##  1st Qu.:0                   1st Qu.:0.0000    1st Qu.:0.0000
##  Median :0                   Median :0.0000    Median :0.0000
##  Mean   :0                   Mean   :0.1609    Mean   :0.1854
##  3rd Qu.:0                   3rd Qu.:0.0000    3rd Qu.:0.0000
##  Max.   :0                   Max.   :1.0000    Max.   :1.0000
##  weekday_is_wednesday weekday_is_thursday weekday_is_friday
##  Min.   :0.0000       Min.   :0.000       Min.   :0.000
##  1st Qu.:0.0000       1st Qu.:0.000       1st Qu.:0.000
##  Median :0.0000       Median :0.000       Median :0.000
##  Mean   :0.1948       Mean   :0.182       Mean   :0.138
##  3rd Qu.:0.0000       3rd Qu.:0.000       3rd Qu.:0.000
##  Max.   :1.0000       Max.   :1.000       Max.   :1.000
##  weekday_is_saturday weekday_is_sunday   is_weekend
##  Min.   :0.00000     Min.   :0.00000   Min.   :0.0000
##  1st Qu.:0.00000     1st Qu.:0.00000   1st Qu.:0.0000
##  Median :0.00000     Median :0.00000   Median :0.0000
##  Mean   :0.07113     Mean   :0.06774   Mean   :0.1389
##  3rd Qu.:0.00000     3rd Qu.:0.00000   3rd Qu.:0.0000
##  Max.   :1.00000     Max.   :1.00000   Max.   :1.0000
##  global_rate_positive_words global_rate_negative_words
##  Min.   :0                   Min.   :0
##  1st Qu.:0                   1st Qu.:0
##  Median :0                   Median :0
##  Mean   :0                   Mean   :0
##  3rd Qu.:0                   3rd Qu.:0
##  Max.   :0                   Max.   :0
##  avg_positive_polarity min_positive_polarity max_positive_polarity
##  Min.   :0             Min.   :0             Min.   :0
##  1st Qu.:0             1st Qu.:0             1st Qu.:0
##  Median :0             Median :0             Median :0
##  Mean   :0             Mean   :0             Mean   :0
##  3rd Qu.:0             3rd Qu.:0             3rd Qu.:0
##  Max.   :0             Max.   :0             Max.   :0
##  avg_negative_polarity min_negative_polarity max_negative_polarity
##  Min.   :0             Min.   :0             Min.   :0
##  1st Qu.:0             1st Qu.:0             1st Qu.:0
##  Median :0             Median :0             Median :0
##  Mean   :0             Mean   :0             Mean   :0
##  3rd Qu.:0             3rd Qu.:0             3rd Qu.:0
##  Max.   :0             Max.   :0             Max.   :0
##  title_subjectivity title_sentiment_polarity abs_title_sentiment_polarity
##  Min.   :0.0000     Min.   :-1.00000         Min.   :0.0000
##  1st Qu.:0.0000     1st Qu.: 0.00000         1st Qu.:0.0000
##  Median :0.3000     Median : 0.00000         Median :0.1111
##  Mean   :0.3403     Mean   : 0.08539         Mean   :0.1930
##  3rd Qu.:0.5667     3rd Qu.: 0.25000         3rd Qu.:0.3000
##  Max.   :1.0000     Max.   : 1.00000         Max.   :1.0000
##      shares
##  Min.   :     4
##  1st Qu.:  1000
##  Median :  1600
##  Mean   :  4699
##  3rd Qu.:  3800
```

```
## Max. :211600
```

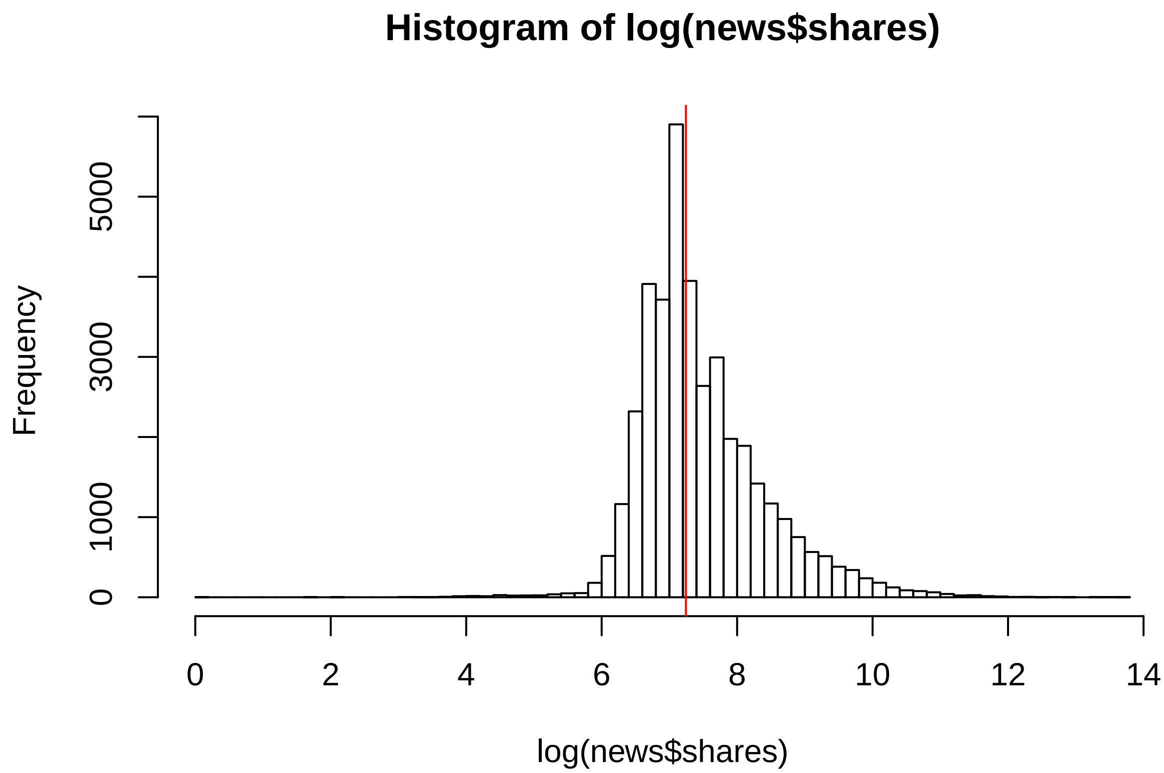
```r
news = news[-which(news$n_tokens_content==0),]
```

Next, the number of shares has a long right tail, and log(shares) looks well distributed. So, I use log(shares) for the linear regression, and make a new variable of logshares.

```r
hist(news$shares, xlim=c(1,10000), breaks=5000) ; abline(v=1400, col='red')
```




```r
hist(log(news$shares), breaks=50) ; abline(v=log(1400), col='red')
```



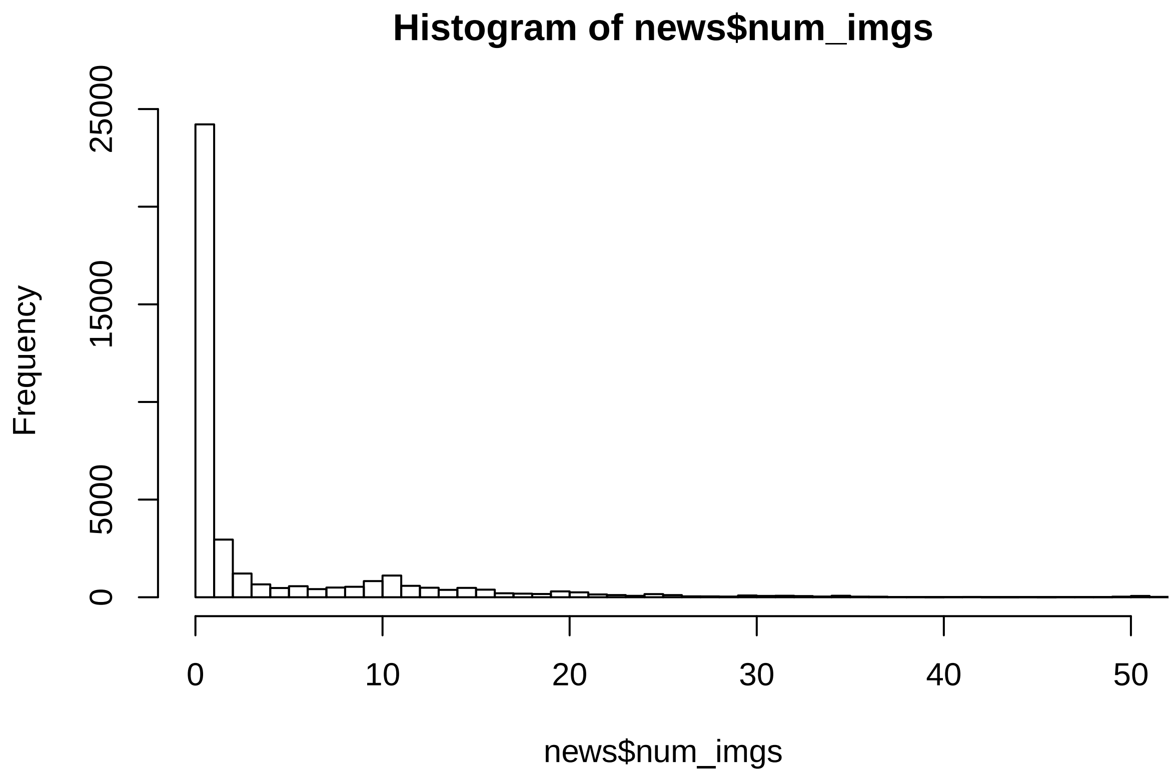

```
summary(news$shares)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1     945    1400    3355    2700  843300
```

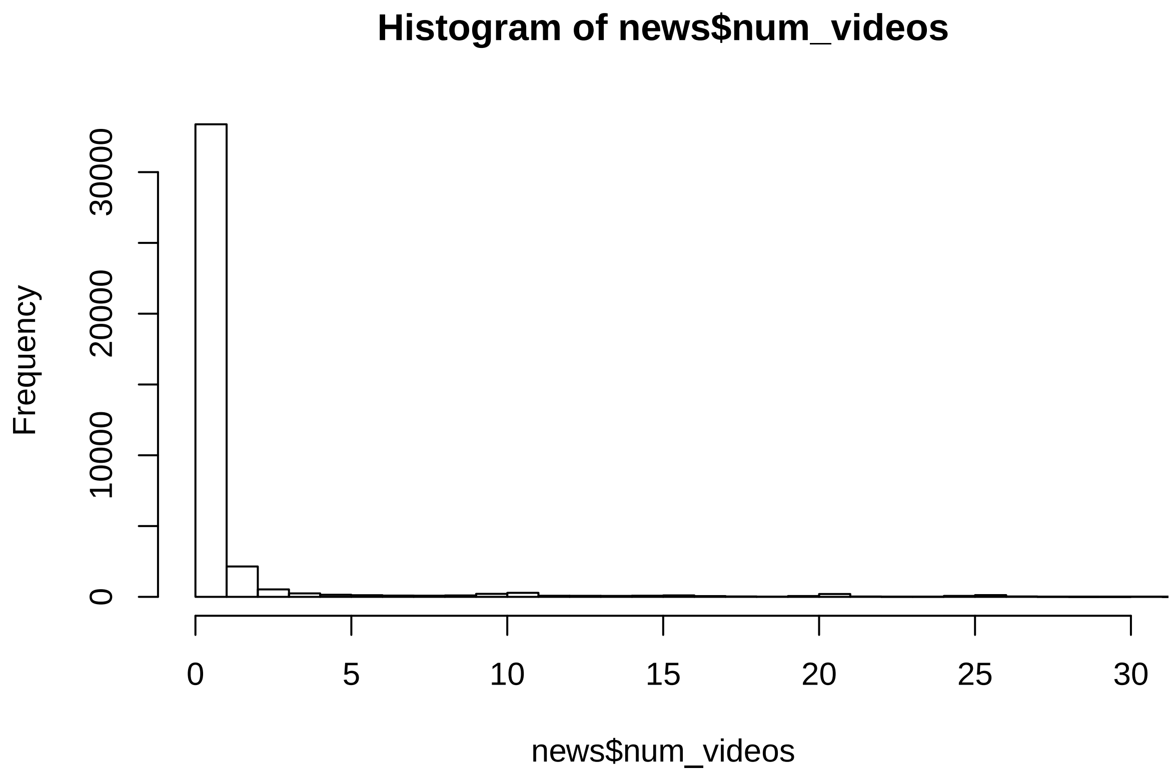
```
news = mutate(news, logshares = log(shares))
```

Last, we can see that most data of "num_imgs"(Number of images) and "num_videos"(Number of videos) are zero or one. Thus, I make dummy variables for those variables.

```
hist(news$num_imgs, xlim = c(0,50), breaks = 128)
```




```r
hist(news$num_videos, xlim = c(0,30), breaks = 91)
```

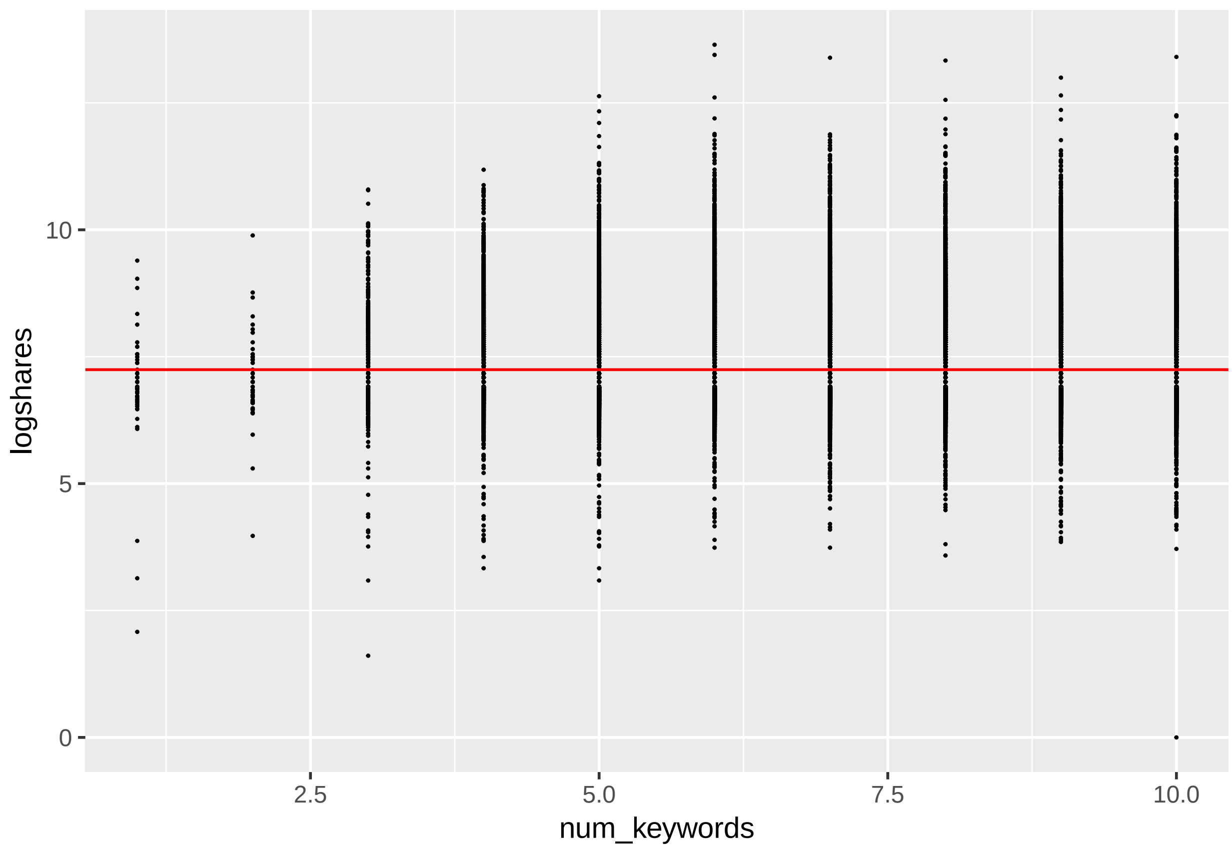



```
news = mutate(news, img = ifelse(num_imgs==0,0,1))
news = mutate(news, video = ifelse(num_videos==0,0,1))
```

Data analysis

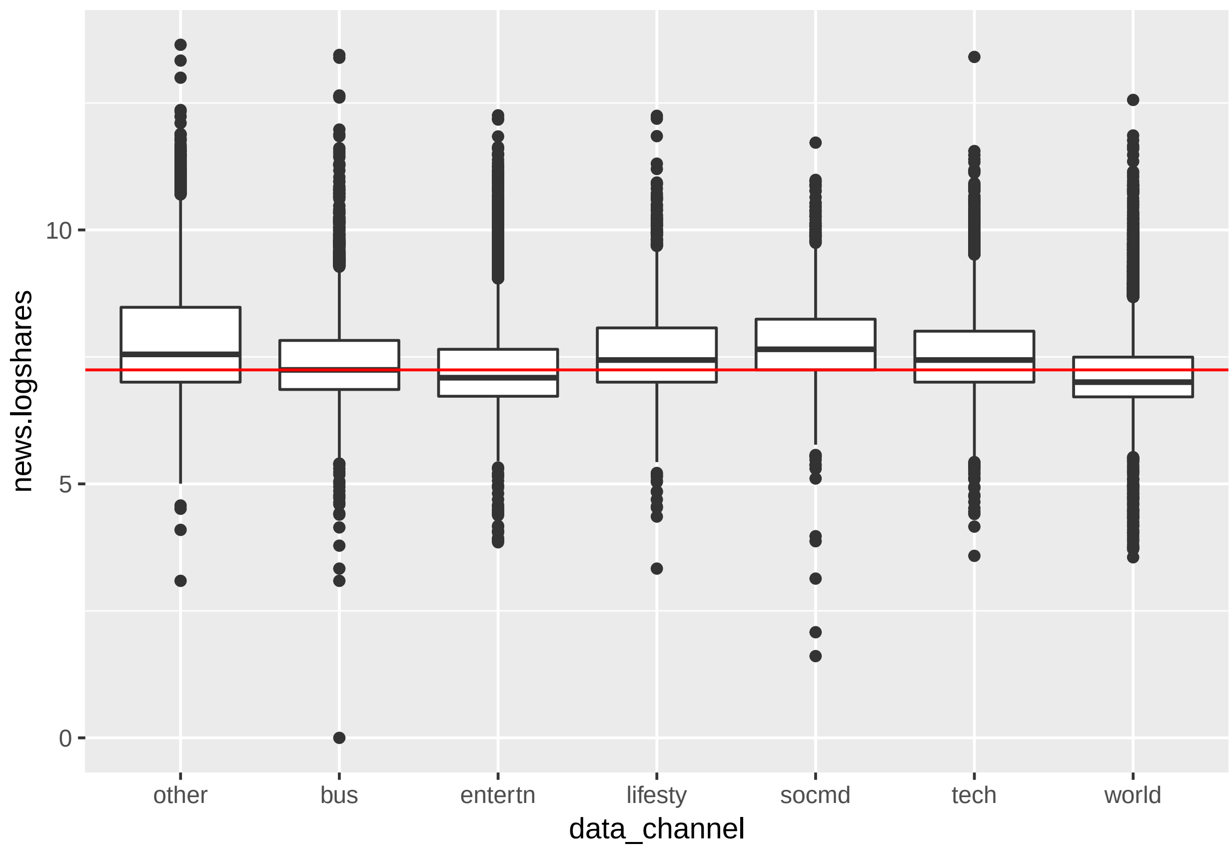
It is difficult to find a strong relationship between log(shares) and other variables. Most variables show the following relationship.

```
ggplot(data=news) + geom_point(aes(x=num_keywords, y=logshares), size=0.1) + geom_hline(yintercept = log
```

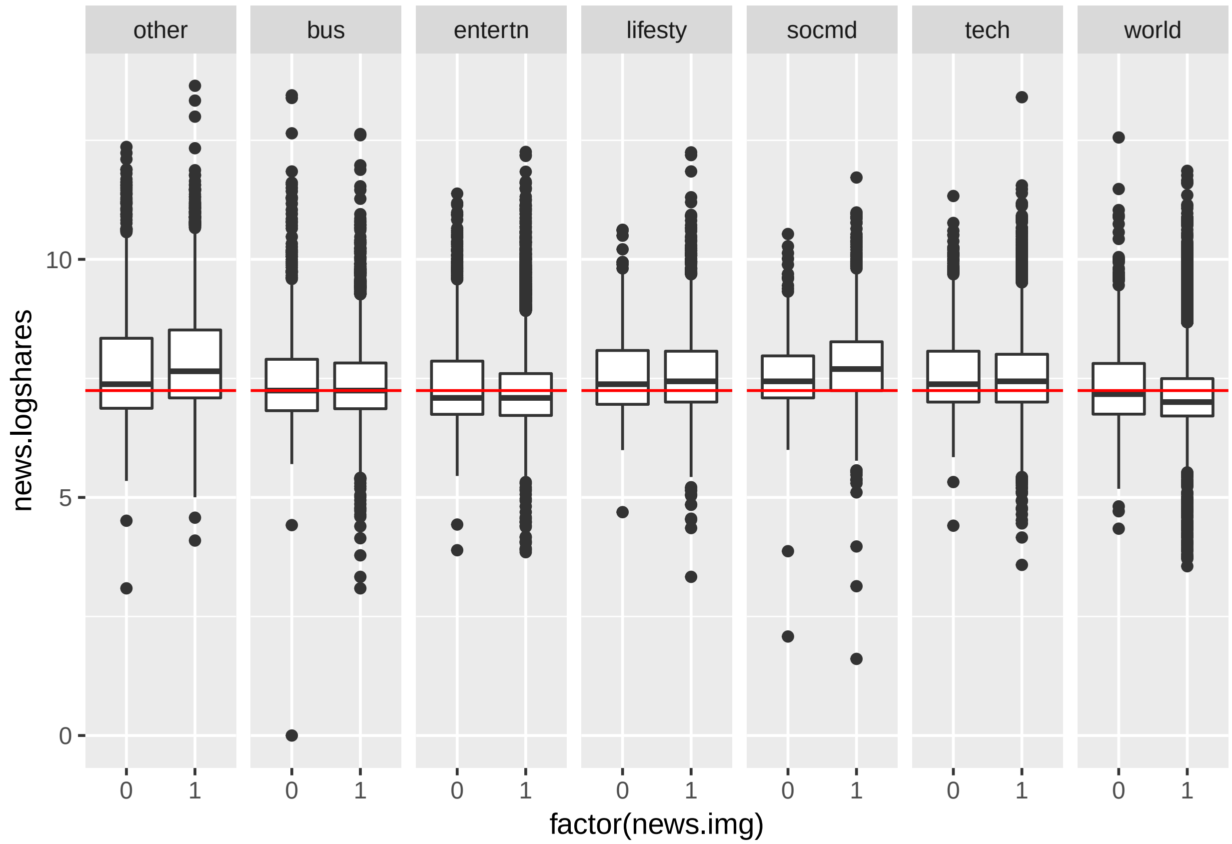


However, we can find a relationship with the 'data channel' dummy variables and 'weekday' dummy variables. Data channel variables show an intercation relationship with image and video variables. Weekday variables show a relationship when it is weekend. I will use a "is_weekend" variable not "weekday_is_" variables.

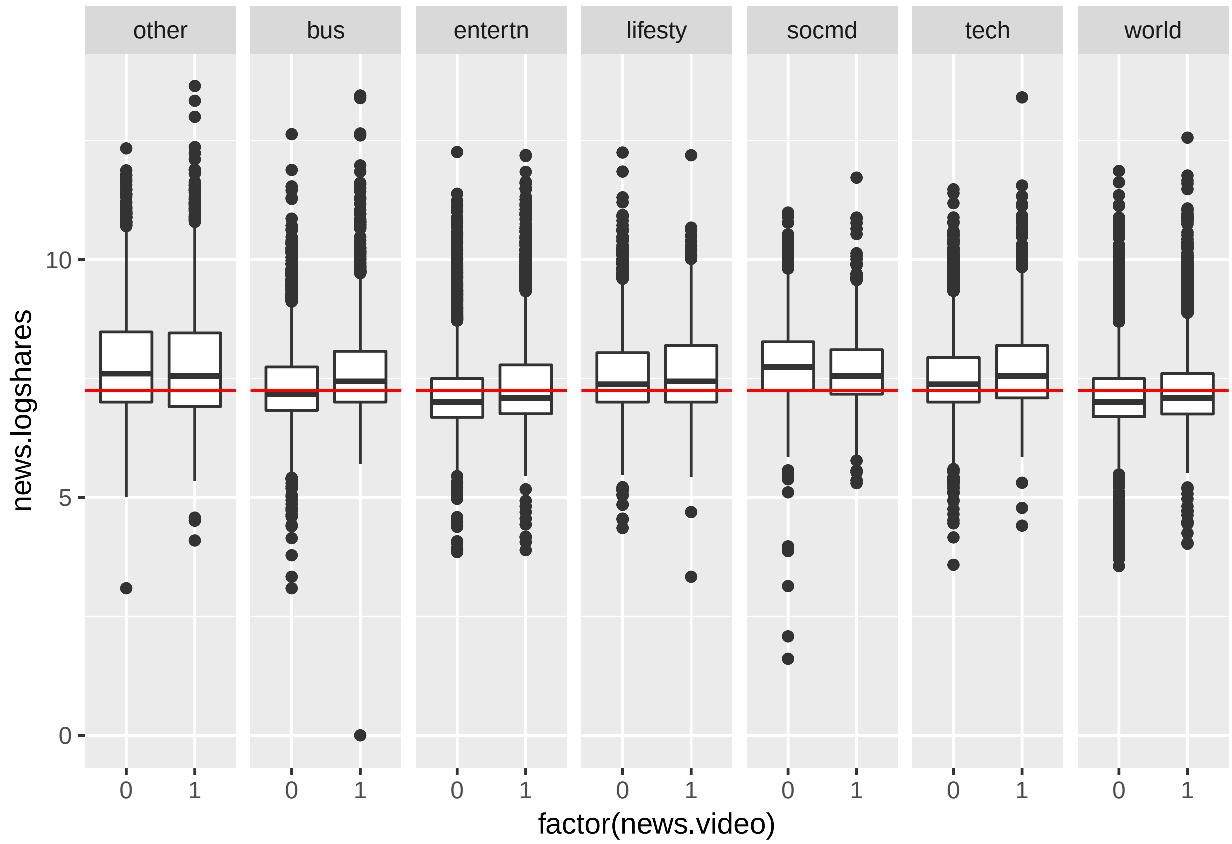
```
# data channel
news_data_ch = select(news, data_channel_is_bus, data_channel_is_entertainment,
                      data_channel_is_lifestyle, data_channel_is_socmed,
                      data_channel_is_tech, data_channel_is_world)
data_channel = factor(data.matrix(news_data_ch) %*% 1:ncol(news_data_ch),
                      labels = c("other", "bus", "entertn", "lifesty", "socmd", "tech", "world"))
news_data_ch = data.frame(news_data_ch, data_channel, news$logshares, news$img, news$video)
ggplot(data = news_data_ch) + geom_boxplot((aes(x=data_channel, y=news.logshares))) +
  geom_hline(yintercept = log(1400), col="red")
```



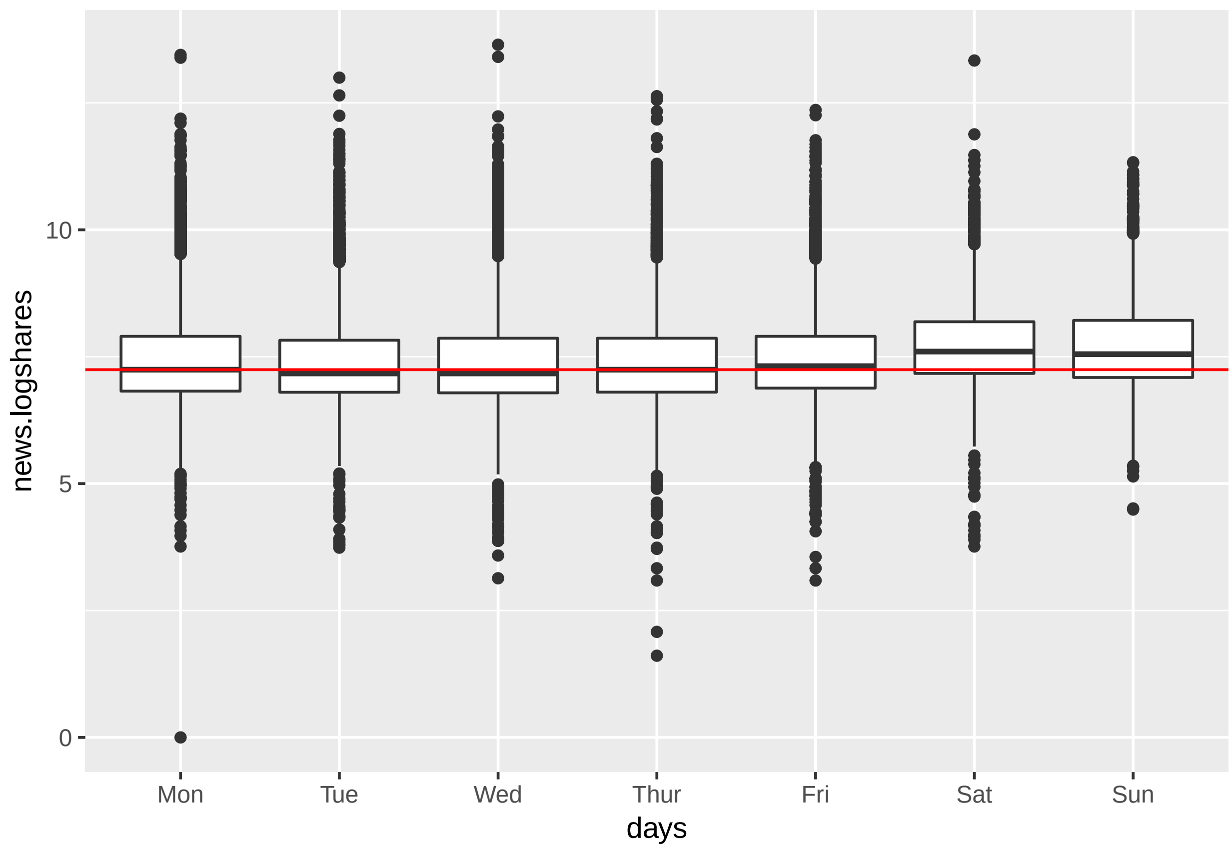

```
ggplot(data = news_data_ch) + geom_boxplot((aes(x=factor(news.img), y=news.logshares))) +
  facet_wrap( ~ data_channel, nrow = 1) +
  geom_hline(yintercept = log(1400), col="red")
```




```r
ggplot(data = news_data_ch) + geom_boxplot((aes(x=factor(news.video), y=news.logshares))) +
  facet_wrap( ~ data_channel, nrow = 1) +
  geom_hline(yintercept = log(1400), col="red")
```




```
# weekday
news_days = select(news, weekday_is_monday, weekday_is_tuesday, weekday_is_wednesday,
                   weekday_is_thursday, weekday_is_friday, weekday_is_saturday, weekday_is_sunday)
days = factor(data.matrix(news_days) %*% 1:ncol(news_days),
              labels = c("Mon", "Tue", "Wed", "Thur", "Fri", "Sat", "Sun"))
news_days = data.frame(news_days, days, news$logshares)
ggplot(data = news_days) + geom_boxplot((aes(x=days, y=news.logshares))) + geom_hline(yintercept = log(
```



Modelling

I split the data for training and testing. I use 80% data to build a model by training, and use other 20% data for test. The train and test data are selected randomly.

```
## Split into training and testing sets
n = nrow(news)
n_train = round(0.8*n)   # round to nearest integer
n_test = n - n_train
```

I will test the model by using confusion table. For this purpose, I define some functions. Because of random sampling, the model and the test result changes at each randoming sampling. So, I will repeat 100 times and average the test results.

```
# define function
## confidence table
conf_table = function(y, yhat) {
  y_test = ifelse(y>log(1400), 1, 0)
  yh_t = ifelse(yhat>log(1400), 1, 0)
  table(y_test, yh_t)
}

## conf_rate
conf_rate = function(y, yhat) {
  y_test = ifelse(y>log(1400), 1, 0)
```

```
  yh_t = ifelse(yhat>log(1400), 1, 0)
  sum(yh_t != y_test)/length(y)
}
```

Linear model

First, I build a linear regresstion model by hand bulid and trial. I use overall error rate to find the best linear model. Model 3 is the initial model to check the additional performance of my hand-build model. By assumming the relationship of variables, buinding the model and testing it, I make the model 2. Then, I make a model 1 by deleting the variables which have low p-values. The test resule shows that model 1 has the least overall error rate, and this is the best linear model. I will do the performance check of my best linear model later, with the logit model in order to use the same sampling train and test data.

```
err_vals = do(100)*{

  train_cases = sample.int(n, n_train, replace=FALSE)
  test_cases = setdiff(1:n, train_cases)
  news_train = news[train_cases,]
  news_test = news[test_cases,]

  lm1 = lm(logshares ~ n_tokens_content + num_hrefs + num_self_hrefs + average_token_length +
             num_keywords + data_channel_is_lifestyle +
             num_imgs * (data_channel_is_bus + data_channel_is_socmed + data_channel_is_world) +
             video * (data_channel_is_entertainment + data_channel_is_bus +
                        data_channel_is_socmed + data_channel_is_tech + data_channel_is_world) +
             self_reference_avg_sharess * self_reference_max_shares +
             is_weekend + global_rate_positive_words * avg_positive_polarity +
             avg_negative_polarity + title_subjectivity + title_sentiment_polarity,
           data = news_train)
  lm2 =  lm(logshares ~ n_tokens_title + n_tokens_content + num_hrefs + num_self_hrefs +
              average_token_length + num_keywords + (video + num_imgs) *
              (data_channel_is_lifestyle + data_channel_is_entertainment +
                 data_channel_is_bus + data_channel_is_socmed +
                 data_channel_is_tech + data_channel_is_world) +
              self_reference_avg_sharess * (self_reference_min_shares +
                                              self_reference_max_shares) + is_weekend +
              global_rate_positive_words * avg_positive_polarity +
              global_rate_negative_words * avg_negative_polarity + title_subjectivity +
              title_sentiment_polarity, data = news_train)
  lm3 = lm(logshares ~ .-shares- weekday_is_sunday - is_weekend - img - video, data = news_train)

  yhat_test1 = predict(lm1, news_test)
  yhat_test2 = predict(lm2, news_test)
  yhat_test3 = predict(lm3, news_test)

  # confusion rate
  c(conf_rate(news_test$logshares, yhat_test1), conf_rate(news_test$logshares, yhat_test2),
    conf_rate(news_test$logshares, yhat_test3))

}
colMeans(err_vals) %>% round(3)
```

```
##    V1    V2    V3
```

```
## 0.396 0.396 0.411
```

Classification model

I use a logit model for the classification, because y is a binimial(viral or not) variable. I build a logit model by using the same variables with the linear model. In order to show the averages of "Overall Error Rate", "True Positivie Rate", and "False Positive Rate" of a logit model and a linear model, I repeat 100 times and make averages of the results. As you can see, the logit model is better in terms of Overall Error Rate and False Positive Rate, but the linear model is better in terms of True Positive Rate.

```
news = mutate(news, viral = ifelse(shares > 1400,1,0))

errs_vals = do(100) * {
  train_cases = sample.int(n, n_train, replace=FALSE)
  test_cases = setdiff(1:n, train_cases)
  news_train = news[train_cases,]
  news_test = news[test_cases,]

  logit_m = glm(viral ~  n_tokens_content + num_hrefs + num_self_hrefs + average_token_length +
                  num_keywords + data_channel_is_lifestyle + num_imgs *
                  (data_channel_is_bus + data_channel_is_socmed + data_channel_is_world) + video *
                  (data_channel_is_entertainment + data_channel_is_bus + data_channel_is_socmed +
                     data_channel_is_tech + data_channel_is_world) +  self_reference_avg_sharess *
                  self_reference_max_shares + is_weekend + global_rate_positive_words *
                  avg_positive_polarity + avg_negative_polarity +
                  title_subjectivity + title_sentiment_polarity, data = news_train, family = 'binomial')
  phat_logit = predict(logit_m, news_test, type = 'response')
  yhat_logit = ifelse(phat_logit>0.5, 1, 0)
  ct_lg = table(news_test$viral, yhat_logit)

  # linear
  lmF = lm(logshares ~ n_tokens_content + num_hrefs + num_self_hrefs + average_token_length +
             num_keywords + data_channel_is_lifestyle + num_imgs *
             (data_channel_is_bus + data_channel_is_socmed + data_channel_is_world) + video *
             (data_channel_is_entertainment + data_channel_is_bus + data_channel_is_socmed +
                data_channel_is_tech + data_channel_is_world) +  self_reference_avg_sharess *
             self_reference_max_shares + is_weekend + global_rate_positive_words *
             avg_positive_polarity + avg_negative_polarity +
             title_subjectivity + title_sentiment_polarity, data = news_train)
  yhatF = predict(lmF, news_test)
  ct_lm = conf_table(news_test$logshares, yhatF)

  # result
  c((1-sum(diag(ct_lg))/sum(ct_lg)), (1-sum(diag(ct_lm))/sum(ct_lm)),
    ct_lg[2,2]/sum(ct_lg[2,]), ct_lm[2,2]/sum(ct_lm[2,]),
    ct_lg[1,2]/sum(ct_lg[1,]), ct_lm[1,2]/sum(ct_lm[1,]))
}
errMean = colMeans(errs_vals) %>% round(3)
err = matrix(errMean, nrow = 2, dimnames =
               list(c("Classification Model", "Numerical Model"),
               c("OverallErrorRate", "TruePositiveRate", "FalsePositiveRate")))
kable(err) %>% kable_styling("striped")
```

	OverallErrorRate	TruePositiveRate	FalsePositiveRate
Classification Model	0.367	0.626	0.361
Numerical Model	0.397	0.842	0.628