TMDB box office prediction-Text mining process and more EDA

Mianchun Lu, Yuting Gong, Cijun Sun, Guangxu Luo

April 19, 2019

libraries

```
#Sys.setenv(JAVA_HOME='C:/Program Files/Java/jdk-11.0.2')
library(rJava)
```

```
## Warning: package 'rJava' was built under R version 3.5.2
```

```
library(qdap)
```

```
## Warning: package 'qdap' was built under R version 3.5.2
```

```
## Warning: package 'qdapDictionaries' was built under R version 3.5.2
```

```
## Warning: package 'qdapRegex' was built under R version 3.5.2
```

```
## Warning: package 'qdapTools' was built under R version 3.5.2
```

```
library(gtools)
library(tidytext)
```

```
## Warning: package 'tidytext' was built under R version 3.5.2
```

```
library(tm)
```

```
## Warning: package 'tm' was built under R version 3.5.2
```

```
## Warning: package 'NLP' was built under R version 3.5.2
```

```
library(stringr)
library(syuzhet)
```

```
## Warning: package 'syuzhet' was built under R version 3.5.3
```

```
library(dplyr)
library(gridExtra)
library(wordcloud)
```

```
## Warning: package 'wordcloud' was built under R version 3.5.2
```

```
library(xgboost)
library(readr)
library(stringr)
library(caret)
library(car)
library(gbm)          # basic implementation
library(ggplot2)      # model visualization
library(tidyr)
library(plyr)
library(randomForest)
library(cluster)
```

```
## Warning: package 'cluster' was built under R version 3.5.2
```

I load the datasets cleaned in the Deliverable 1. The main purpose of the following code is to do the text mining of the variables that have not been processed before(e.g. overview, keywords). Besides, I also calculate the log of budget and revenue to minimize the error caused by the high variance.

```
setwd('D:/personal/columbia university/AA method 2/final project/tmdb-box-office-prediction')
train=read.csv("train_clean.csv", stringsAsFactors = FALSE, na.strings = c("","#N/A","[]"))
test=read.csv("test_clean.csv", stringsAsFactors = FALSE,na.strings = c("","#N/A","[]"))
```

variable: Keywords

new variable: the number of keywords

```
keywordsCount_train <- str_count(train$Keywords, "\\}")
train$numberOfKeywords <- keywordsCount_train
train$numberOfKeywords[is.na(train$Keywords)] <- 0
keywordsCount_test <- str_count(test$Keywords, "\\}")  # each Keyword is followed by a "}"
test$numberOfKeywords <- keywordsCount_test
test$numberOfKeywords[is.na(test$Keywords)] <- 0
```

Log Revenue

convert budget and revenue to log

```
train$log.budget=log(train$budget+1)
train$log.revenue=log(train$revenue+1)
test$log.budget=log(test$budget+1)
```

text mining

Data Preparation One

Firstly, I counted the overview length in words and in sentences seperately.

```
# in words
train$overviewLengthInWords=str_count(string = train$overview,pattern = '\\S+')
test$overviewLengthInWords=str_count(string = test$overview,pattern = '\\S+')

# in sentence
train$overviewLengthInSentence=str_count(string = train$overview,pattern = "[A-Za-z,;'\"\\s]+[^.!?]*[.?
test$overviewLengthInSentence=str_count(string = test$overview,pattern = "[A-Za-z,;'\"\\s]+[^.!?]*[.?!]

# For overview, there are 0 missing value in the train data and test data.
sum(is.na(train$overview))
```

```
## [1] 0
```

```
sum(is.na(test$overview))
```

```
## [1] 0
```

```
# let's see the coorelation between the revenue and these two new variables
cor(train$revenue,train$overviewLengthInWords)
```

```
## [1] -0.004841359
```

```
cor(train$overviewLengthInSentence,train$revenue)
```

```
## [1] -0.005592217
```

Let's use qdap package to have a look at the words appeared in the overview frequently. (An interesting thing is that the following result has a little difference with the xdtm)

```
overview.all=smartbind(train,test)
freq_terms(text.var=overview.all$overview,top=25,stopwords = Top100Words)
```

```
##    WORD    FREQ
## 1  life    1257
## 2  after   1165
## 3  new     1066
## 4  young    931
## 5  world    818
## 6  man      782
## 7  love     772
## 8  family   730
## 9  story    686
## 10 must     613
## 11 film     585
## 12 only     573
## 13 while    558
## 14 finds    548
## 15 years    525
## 16 where    507
## 17 father   476
## 18 help     468
## 19 woman    464
## 20 back     461
## 21 friends  452
## 22 war      429
## 23 lives    425
## 24 own      423
## 25 home     411
```

Then, I counted the number of positive and negative words in every row with the help of "nrc", which can represent the sentiment level of each review. I also did the same work to test data.

```
df = data.frame()
df.new = data.frame()
library(syuzhet)
for (i in 1:nrow(train))
  {
  sentiment1 <- get_nrc_sentiment(train$overview[i])
  p1 = sum(sentiment1$positive)
  n1 = sum(sentiment1$negative)
  anger1=sum(sentiment1$anger)
  anticipation1=sum(sentiment1$anticipation)
  disgust1=sum(sentiment1$disgust)
  fear1=sum(sentiment1$fear)
  joy1=sum(sentiment1$joy)
  sadness1=sum(sentiment1$sadness)
  surprise1=sum(sentiment1$surprise)
  trust1=sum(sentiment1$trust)
  df.new = cbind(p1,n1,anger1,anticipation1,disgust1,fear1,joy1,sadness1,surprise1,trust1)
  df = rbind(df,df.new)
}
```

```
train= cbind(train,df)
```

```
df = data.frame()
df.new = data.frame()
for (i in 1:nrow(test))
  {
  sentiment1 <- get_nrc_sentiment(test$overview[i])
  p1 = sum(sentiment1$positive)
  n1 = sum(sentiment1$negative)
  anger1=sum(sentiment1$anger)
  anticipation1=sum(sentiment1$anticipation)
  disgust1=sum(sentiment1$disgust)
  fear1=sum(sentiment1$fear)
  joy1=sum(sentiment1$joy)
  sadness1=sum(sentiment1$sadness)
  surprise1=sum(sentiment1$surprise)
  trust1=sum(sentiment1$trust)
  df.new = cbind(p1,n1,anger1,anticipation1,disgust1,fear1,joy1,sadness1,surprise1,trust1)
  df = rbind(df,df.new)
}

test= cbind(test,df)
```

Sentiment Analysis One

```
#sentiment analysis: p1, n1, anger1, etc
senti_plot<-function(senti, label){
  ggplot(data=train, aes(x = senti,
              y = revenue, fill=senti,
              stat = "summary", fun.y = "mean")) +
  geom_bar(stat = "identity") +
  theme_light() +
  xlab(label) +
  ylab("Average Revenue")
}

head(train)
```

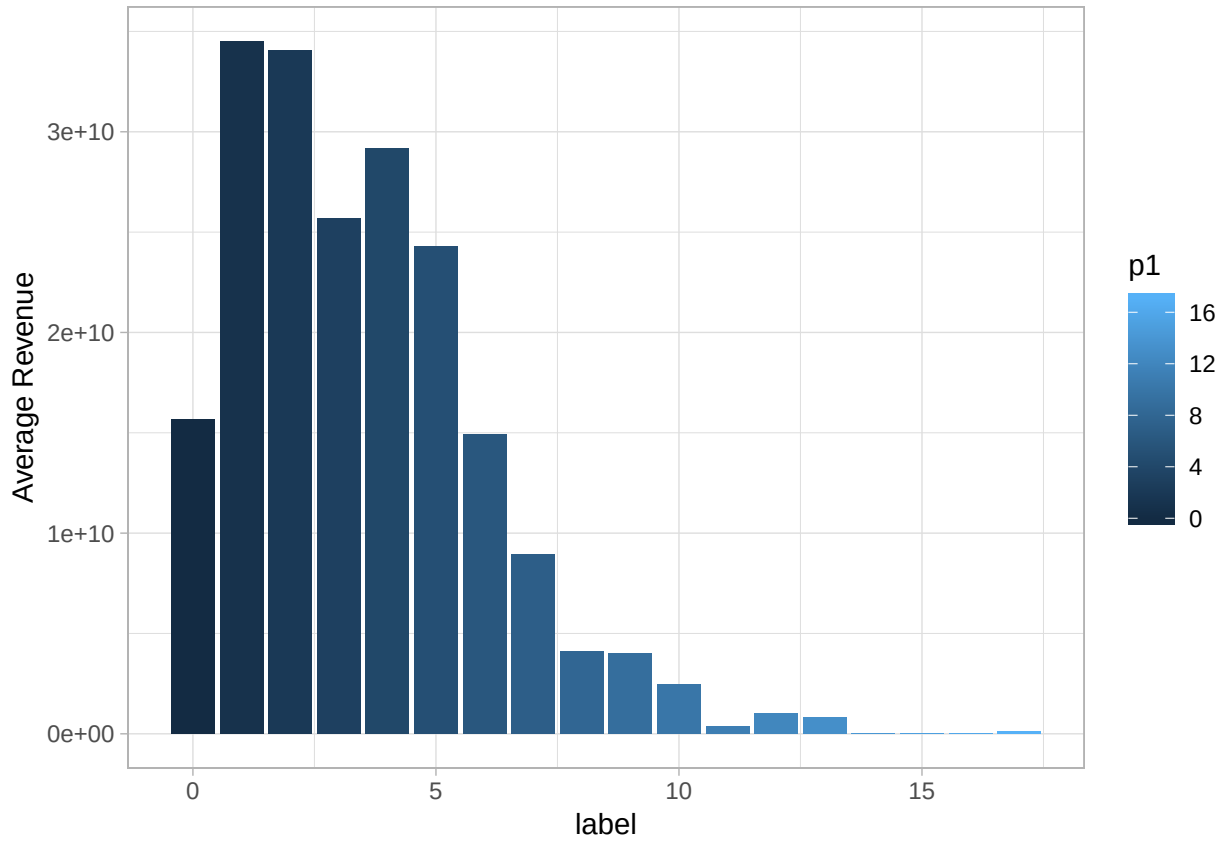
```
##   X id belongs_to_collection  budget original_language
## 1 1  1                     1 1.4e+07                en
## 2 2  2                     1 4.0e+07                en
## 3 3  3                     0 3.3e+06                en
## 4 4  4                     0 1.2e+06                hi
## 5 5  5                     0 1.6e+07                ko
## 6 6  6                     0 8.0e+06                en
##                            original_title
## 1                   Hot Tub Time Machine 2
## 2 The Princess Diaries 2: Royal Engagement
## 3                                Whiplash
## 4                                 Kahaani
## 5                              ë§\210ë¦°ë³´ì\235´
## 6   Pinocchio and the Emperor of the Night
##
## 1
```

```
## 2
## 3
## 4 Vidya Bagchi (Vidya Balan) arrives in Kolkata from London to find her missing husband Arnab Bagchi
## 5
## 6
##   popularity month year runtime
## 1   6.575393     2 2015      93
## 2   8.248895     8 2004     113
## 3  64.299990    10 2014     105
## 4   3.174936     3 2012     122
## 5   1.148070     2 2009     118
## 6   0.743274     8 1987      83
##                                                      tagline
## 1         The Laws of Space and Time are About to be Violated.
## 2 It can take a lifetime to find true love; she's got 30 days!
## 3              The road to greatness can take you to the edge.
## 4                                                         NA
## 5                                                         NA
## 6                                                         NA
##                                     title
## 1                   Hot Tub Time Machine 2
## 2 The Princess Diaries 2: Royal Engagement
## 3                                Whiplash
## 4                                 Kahaani
## 5                              Marine Boy
## 6   Pinocchio and the Emperor of the Night
##
## 1
## 2
## 3 [{'id': 1416, 'name': 'jazz'}, {'id': 1523, 'name': 'obsession'}, {'id': 1640, 'name': 'conservato
## 4
## 5
## 6
##    revenue release_date week_day numberOfGenres Action Adventure Animation
## 1 12314651   2015-02-20   Friday              1      0         0         0
## 2 95149435   2004-08-06   Friday              4      0         0         0
## 3 13092000   2014-10-10   Friday              1      0         0         0
## 4 16000000   2012-03-09   Friday              2      0         0         0
## 5  3923970   2009-02-05 Thursday              2      1         0         0
## 6  3261638   1987-08-06 Thursday              3      0         1         1
##   Comedy Crime Documentary Drama Family Fantasy Foreign History Horror
## 1      1     0           0     0      0       0       0       0      0
## 2      1     0           0     1      1       0       0       0      0
## 3      0     0           0     1      0       0       0       0      0
## 4      0     0           0     1      0       0       0       0      0
## 5      0     0           0     0      0       0       0       0      0
## 6      0     0           0     0      1       0       0       0      0
##   Music Mystery Romance Science.Fiction Thriller TV.Movie War Western
## 1     0       0       0               0        0        0   0       0
## 2     0       0       1               0        0        0   0       0
## 3     0       0       0               0        0        0   0       0
## 4     0       0       0               0        1        0   0       0
## 5     0       0       0               0        1        0   0       0
## 6     0       0       0               0        0        0   0       0
```

```
##   numberOfcasts numberOfcrews numberOfcompanies Columbia.Pictures
## 1            24            72                 3                 0
## 2            20             9                 1                 0
## 3            51            64                 3                 0
## 4             7             3                 0                 0
## 5             4             2                 0                 0
## 6             4            11                 0                 0
##   Columbia.Pictures.Corporation Metro.Goldwyn.Mayer..MGM. New.Line.Cinema
## 1                             0                         0              0
## 2                             0                         0              0
## 3                             0                         0              0
## 4                             0                         0              0
## 5                             0                         0              0
## 6                             0                         0              0
##   Paramount.Pictures Touchstone.Pictures TriStar.Pictures
## 1                  1                   0                0
## 2                  0                   0                0
## 3                  0                   0                0
## 4                  0                   0                0
## 5                  0                   0                0
## 6                  0                   0                0
##   Twentieth.Century.Fox.Film.Corporation Universal.Pictures
## 1                                      0                  0
## 2                                      0                  0
## 3                                      0                  0
## 4                                      0                  0
## 5                                      0                  0
## 6                                      0                  0
##   Walt.Disney.Pictures Warner.Bros. numberOfcoun numberOflang is_released
## 1                    0           0           1            1           1
## 2                    1           0           1            1           1
## 3                    0           0           1            1           1
## 4                    0           0           1            2           1
## 5                    0           0           1            1           1
## 6                    0           0           0            1           1
##      budget2 popularity2   runtime2 numberOfGenres2 numberOfcasts2
## 1 -0.3672958 -0.15597168 -0.7272781      -1.3461401     0.20423378
## 2  0.3751774 -0.01771143  0.2252800       1.3395750    -0.03630066
## 3 -0.6728520  4.61307961 -0.1557432      -1.3461401     1.82784125
## 4 -0.7328210 -0.43690831  0.6539311      -0.4509017    -0.81803759
## 5 -0.3101825 -0.60436254  0.4634195      -0.4509017    -0.99843842
## 6 -0.5386357 -0.63780570 -1.2035571       0.4443366    -0.99843842
##   numberOfcrews2 numberOfcompanies2 numberOflang2 numberOfcoun2   revenue2
## 1      1.6972003          0.1497759    -0.5095632    -0.4337528 -0.3956248
## 2     -0.5474751         -0.8432132    -0.5095632    -0.4337528  0.2066684
## 3      1.4121621          0.1497759    -0.5095632    -0.4337528 -0.3899727
## 4     -0.7612538         -1.3397078     0.6169583    -0.4337528 -0.3688286
## 5     -0.7968835         -1.3397078    -0.5095632    -0.4337528 -0.4566336
## 6     -0.4762156         -1.3397078    -0.5095632    -1.7629237 -0.4614494
##   numberOfKeywords log.budget log.revenue overviewLengthInWords
## 1                4   16.45457    16.32630                   29
## 2                4   17.50439    18.37096                   69
## 3               12   15.00943    16.38751                   21
## 4                7   13.99783    16.58810                  106
```

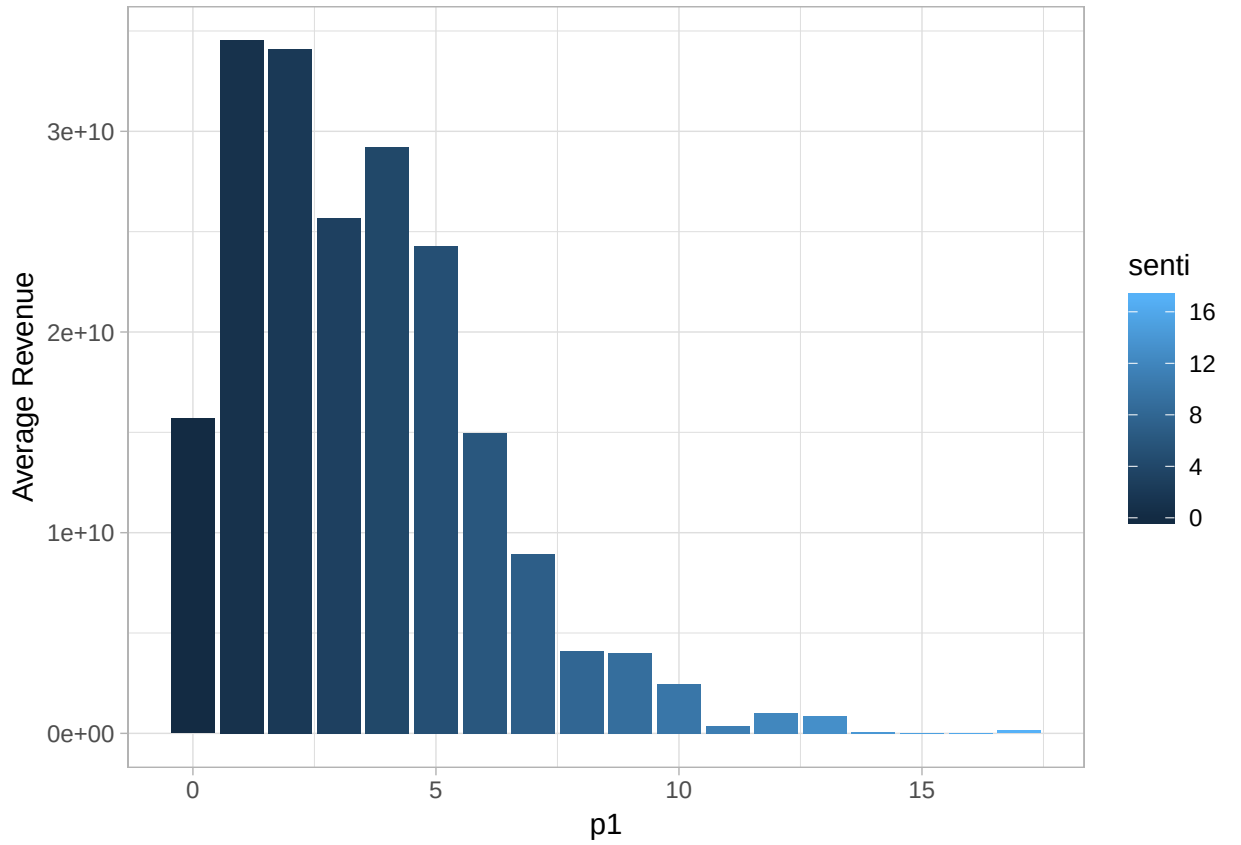
```
## 5                0   16.58810    15.18261                    32
## 6                0   15.89495    14.99774                    28
##   overviewLengthInSentence p1 n1 anger1 anticipation1 disgust1 fear1 joy1
## 1                        1  2  3      2             2        0     4    2
## 2                        4  2  1      0             1        0     0    1
## 3                        1  4  1      1             3        1     1    3
## 4                        5  5  2      0             2        0     1    2
## 5                        1  1  4      1             0        1     0    0
## 6                        2  2  2      1             2        1     1    2
##   sadness1 surprise1 trust1
## 1        2         1      4
## 2        0         0      1
## 3        0         2      3
## 4        1         0      2
## 5        1         0      2
## 6        2         1      1
```

```
ggplot(data=train, aes(x = p1,
             y = revenue, fill=p1,
             stat = "summary", fun.y = "mean")) +
  geom_bar(stat = "identity") +
  theme_light() +
  xlab("label") +
  ylab("Average Revenue")
```

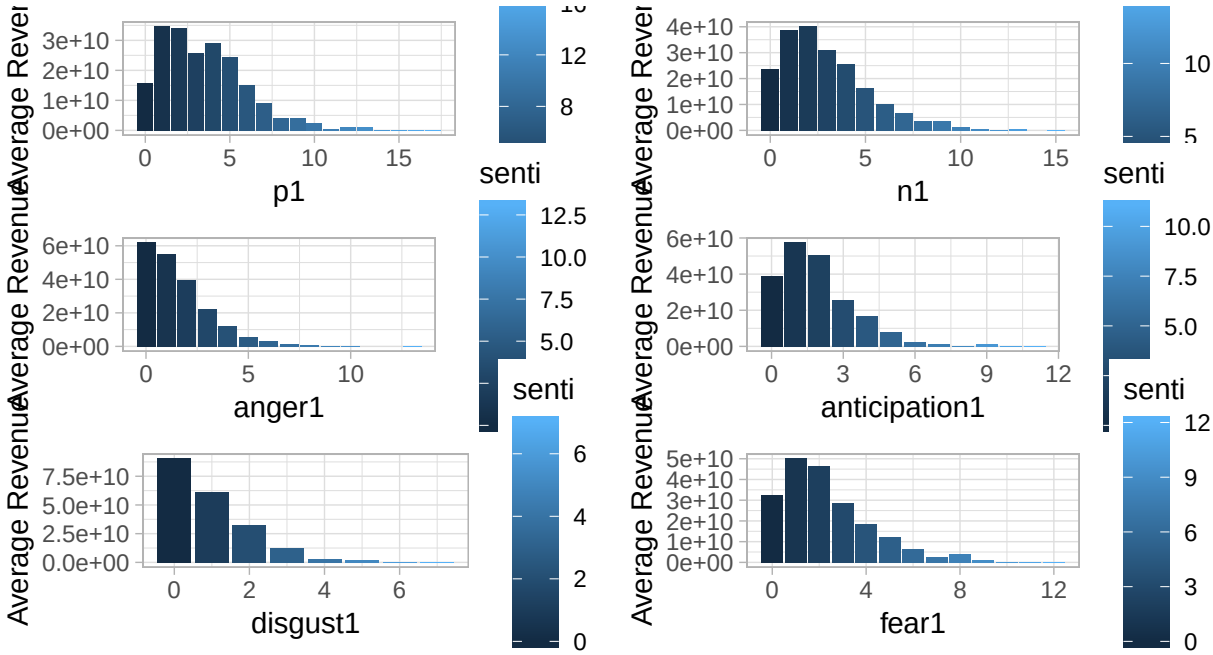



```
#for nrc
p01<-senti_plot(train$p1,"p1")
```

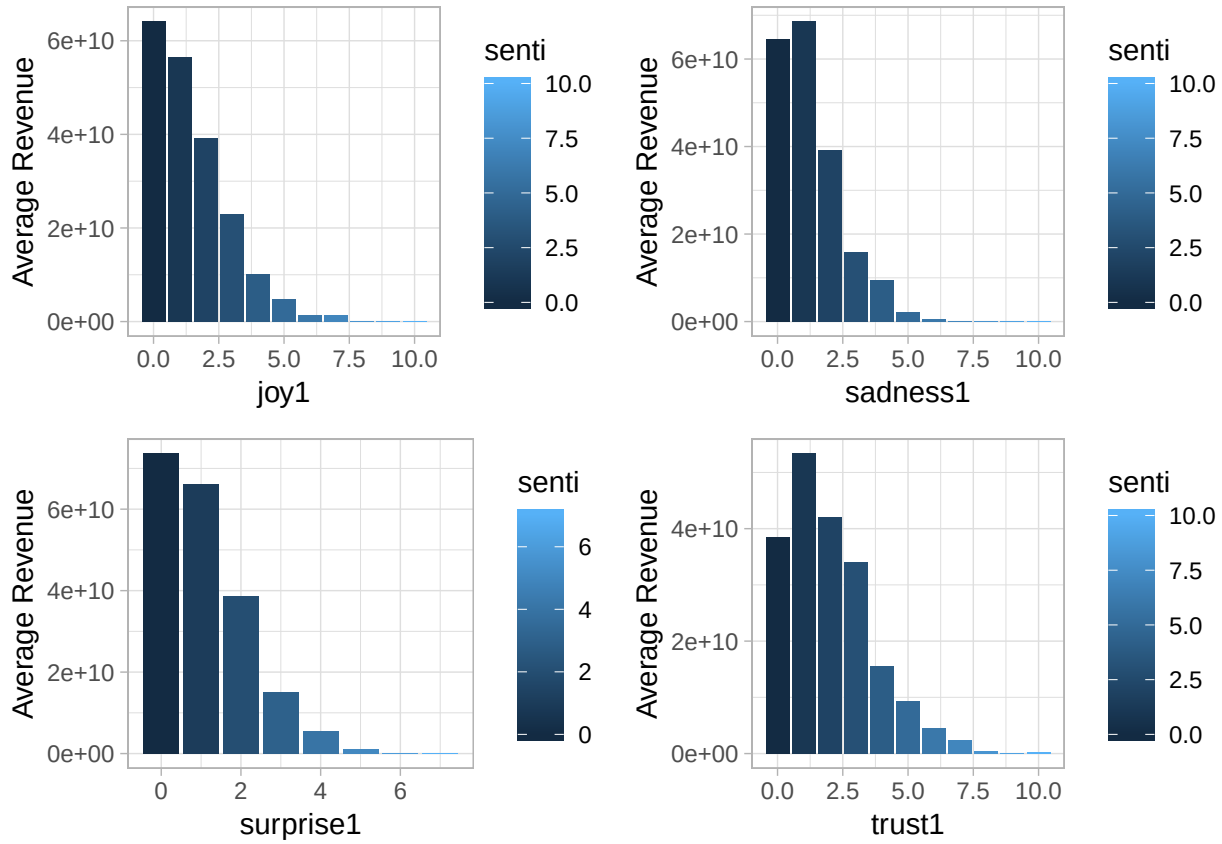
```
p2<-senti_plot(train$n1,"n1")
p3<-senti_plot(train$anger1,"anger1")
p4<-senti_plot(train$anticipation1,"anticipation1")
p5<-senti_plot(train$disgust1,"disgust1")
p6<-senti_plot(train$fear1,"fear1")
p7<-senti_plot(train$joy1,"joy1")
p8<-senti_plot(train$sadness1,"sadness1")
p9<-senti_plot(train$surprise1,"surprise1")
p10<-senti_plot(train$trust1,"trust1")
p01
```




```
grid.arrange(p01,p2,p3,p4,p5,p6,nrow=4)
```




```
grid.arrange(p7,p8,p9,p10,nrow=2)
```



Data Preparation Two

Then, let's try "afinn" lexicon, which scores the sentiment of words. (In this part, we faced with a problem that if all of the words in an overview do not belong to afinn lexion, then it will be deleted from the sentiment=mean(score), which means we cannot bind the ungrouped sentiment with original data. Our solution is to write a for loop and keep every records in a dataframe.)

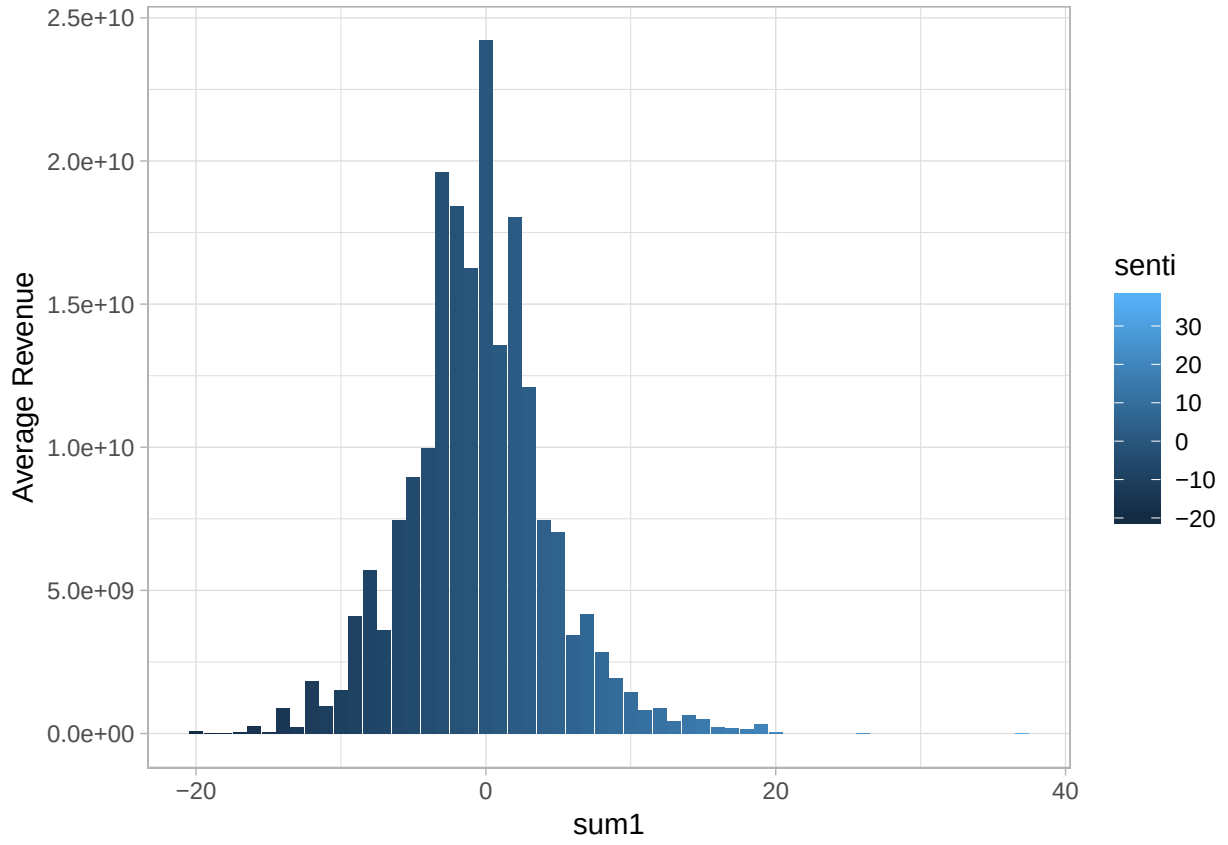
```
# train data
df1 = data.frame()
df.new1 = data.frame()
for (i in 1:nrow(train)){
  senti<-train[i,] %>%
    select(overview)%>%
    unnest_tokens(output=word,input=overview)%>%
    inner_join(get_sentiments('afinn'))
  sum1=sum(senti$score)
  mean1=mean(senti$score)
  df.new1=cbind(sum1,mean1)
  df1=rbind(df1,df.new1)
}

train=cbind(train,df1)

# test data
df1 = data.frame()
df.new1 = data.frame()
```

```
for (i in 1:nrow(test)){
  senti<-test[i,] %>%
    select(overview)%>%
    unnest_tokens(output=word,input=overview)%>%
    inner_join(get_sentiments('afinn'))
  sum1=sum(senti$score)
  mean1=mean(senti$score)
  df.new1=cbind(sum1,mean1)
  df1=rbind(df1,df.new1)
}

test=cbind(test,df1)
```

Sentiment Analysis Two

I draw 12 charts(10 charts above) with the sentiment related variables in order to find out the relationship between them and revenue, which is the explanatory variable. The charts show that the overviews of movies with high revenue tend to include less sentimental words. The emotion of their overview is relatively neutral.

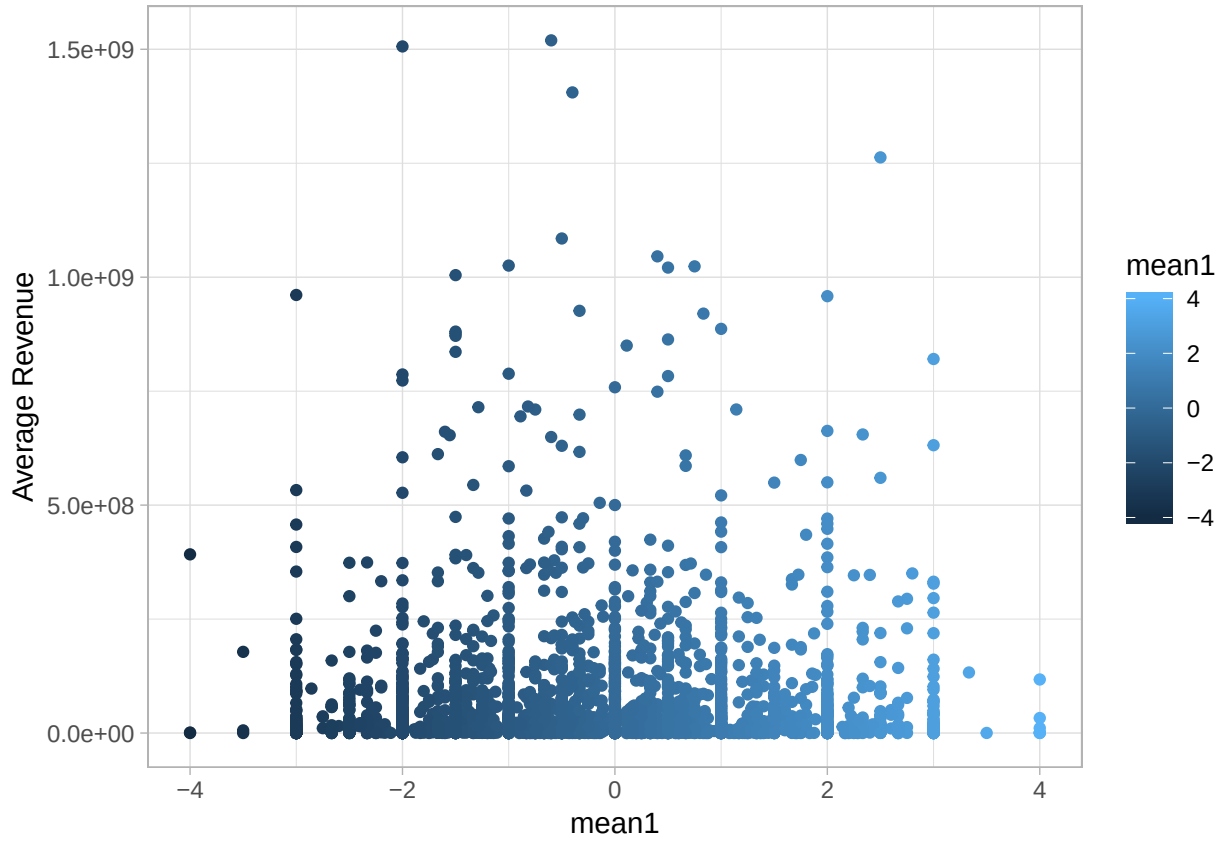
```
#for afinn
senti_plot(train$sum1,"sum1")
```




```
ggplot(data=train, aes(x = mean1,y = revenue, color=mean1,
                       stat = "summary", fun.y = "mean")) +
  geom_point()+
  theme_light() +
```

```
  xlab("mean1") +
  ylab("Average Revenue")
```

```
## Warning: Removed 229 rows containing missing values (geom_point).
```



Data Preparation Three

The next step is to create a corpus which contains all the overviews of train and test. Then, I will use afin to evaluate the emotion score of each row of data.

```
#create a corpus
corpus = Corpus(VectorSource(overview.all$overview))
#convert to lower case
corpus = tm_map(corpus,FUN = content_transformer(tolower))
```

```
## Warning in tm_map.SimpleCorpus(corpus, FUN = content_transformer(tolower)):
## transformation drops documents
```

```
#remove puntuation
corpus = tm_map(corpus,FUN=removePunctuation)
```

```
## Warning in tm_map.SimpleCorpus(corpus, FUN = removePunctuation):
## transformation drops documents
```

```
#Remove stopwords
corpus = tm_map(corpus, FUN=removeWords, c(stopwords('english')))
```

```
## Warning in tm_map.SimpleCorpus(corpus, FUN = removeWords,
```

```
## c(stopwords("english"))): transformation drops documents
```

```
#Strip whitespace
corpus = tm_map(corpus, FUN=stripWhitespace)
```

```
## Warning in tm_map.SimpleCorpus(corpus, FUN = stripWhitespace):
## transformation drops documents
```

```
#Create a dictionary
dict = findFreqTerms(DocumentTermMatrix(Corpus(VectorSource(overview.all$overview))),lowfreq = 0)
dict_corpus = Corpus(VectorSource(dict))

#Stem document
corpus = tm_map(corpus,FUN = stemDocument)
```

```
## Warning in tm_map.SimpleCorpus(corpus, FUN = stemDocument): transformation
## drops documents
```

```
#Create a document term matrix:
dtm = DocumentTermMatrix(corpus)

#Each review is represented as a document in the document term matrix. Let's see how many times the wor
inspect(dtm[1094,'crime'])
```

```
## <<DocumentTermMatrix (documents: 1, terms: 1)>>
## Non-/sparse entries: 1/0
## Sparsity           : 0%
## Maximal term length: 5
## Weighting          : term frequency (tf)
## Sample             :
##       Terms
## Docs   crime
##   1094     2
```

Our matrix is very sparse. We want to remove Sparse Term. Remove sparse term:

```
xdtm = removeSparseTerms(dtm, sparse=0.95)
```

Complete Stems

```
xdtm = as.data.frame(as.matrix(xdtm))
colnames(xdtm) = stemCompletion(x = colnames(xdtm),dictionary = dict_corpus,type='prevalent')
colnames(xdtm) = make.names(colnames(xdtm))
```

Browse tokens

```
sort(colSums(xdtm),decreasing = T)
```

```
##     life     find      new      one    young    world     love      get
##     1268     1258     1066     1064      933      916      904      894
##     live      man     take familial   friend      two   become  storied
##      870      865      860      841      833      828      783      763
##     year     will     film     must     make tri.star   father     help
##      719      691      676      613      612      567      564      557
##     time     come    force      way     turn    woman      set      can
##      529      509      505      504      500      498      498      493
##     work discover      day     back     meet   murder     girl      war
##      492      488      482      475      468      465      462      459
##      son    begin     home     fall     wife
```

```
##      442      431      430      420      413
```

Document Term Matrix-tfidf

```
dtm_tfidf = DocumentTermMatrix(x=corpus,control = list(weighting=function(x) weightTfIdf(x,normalize=F)
xdtm_tfidf = removeSparseTerms(dtm_tfidf,sparse = 0.95)
xdtm_tfidf = as.data.frame(as.matrix(xdtm_tfidf))
colnames(xdtm_tfidf) = stemCompletion(x = colnames(xdtm_tfidf),dictionary = dict_corpus,type='prevalent
colnames(xdtm_tfidf) = make.names(colnames(xdtm_tfidf))
sort(colSums(xdtm_tfidf),decreasing = T)
```

```
##     life     find      new      one     love    world      get      man
## 3506.599 3415.027 3159.812 3155.504 2969.607 2913.333 2881.865 2868.082
##    young familial     live     take   friend      two   become     film
## 2865.765 2860.837 2817.242 2788.030 2739.545 2732.607 2644.337 2627.703
##  storied     will     year     make     must   father tri.star     help
## 2606.567 2538.050 2502.159 2271.017 2237.938 2232.369 2142.567 2121.349
##     time     come    force    woman      way      can      day     turn
## 2084.276 2002.435 1998.841 1994.140 1993.357 1983.436 1980.097 1979.051
##     work      set     girl      war discover     back   murder     meet
## 1971.655 1963.627 1946.256 1945.338 1941.984 1932.435 1922.888 1905.490
##      son    begin     home     wife     fall
## 1881.475 1815.663 1808.348 1765.778 1758.785
```

Combine xdtm_tfidf with our train data and test data:

```
train = cbind(train,xdtm_tfidf[1:3000,])
test = cbind(test,xdtm_tfidf[3001:7398,])
```

I didn't drop any columns during these parts so that you can decide which variables you want to use in your model.

```
write.csv(train,file="train_clean2.csv")
write.csv(test,file="test_clean2.csv")
```

reload

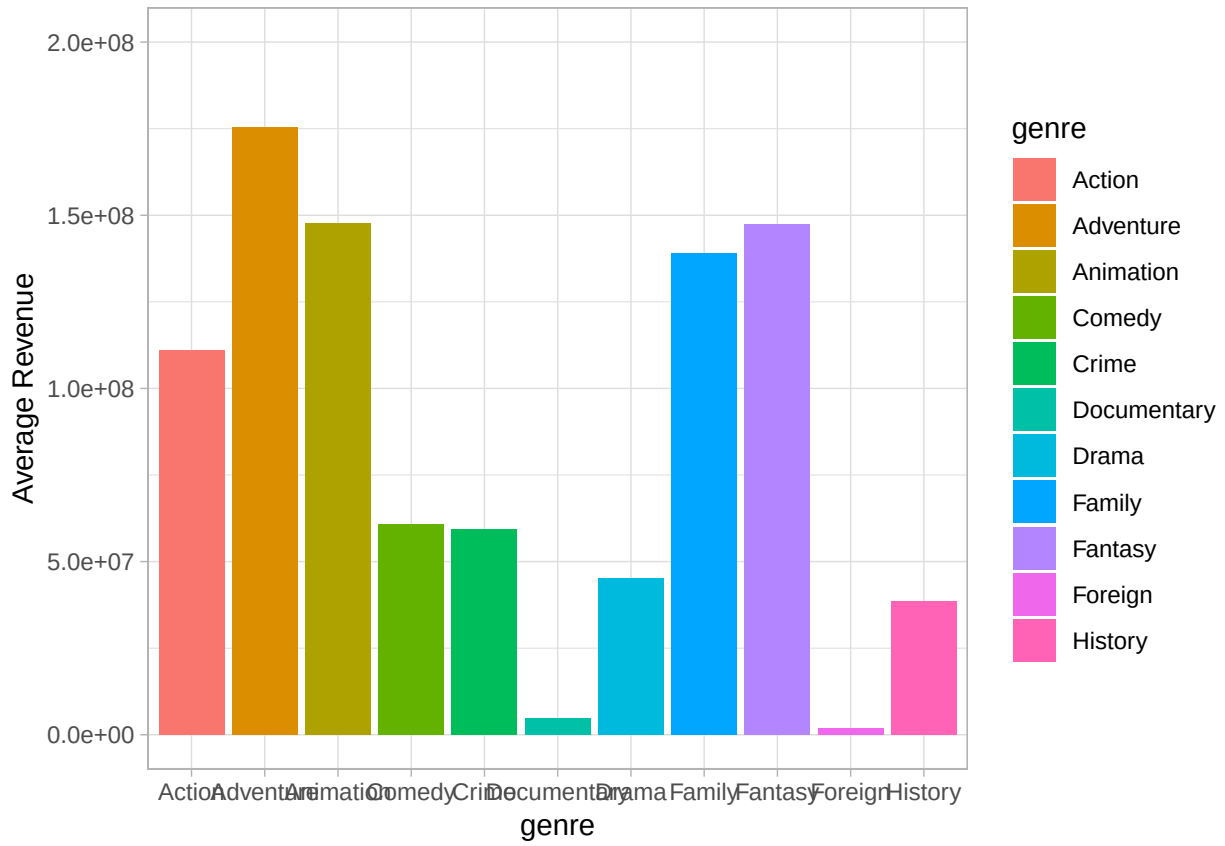
```
train<- read.csv("train_clean2.csv", stringsAsFactors = FALSE, na.strings = c("","#N/A","[]"))
test<- read.csv("test_clean2.csv", stringsAsFactors = FALSE, na.strings = c("","#N/A","[]"))
```

More EDA

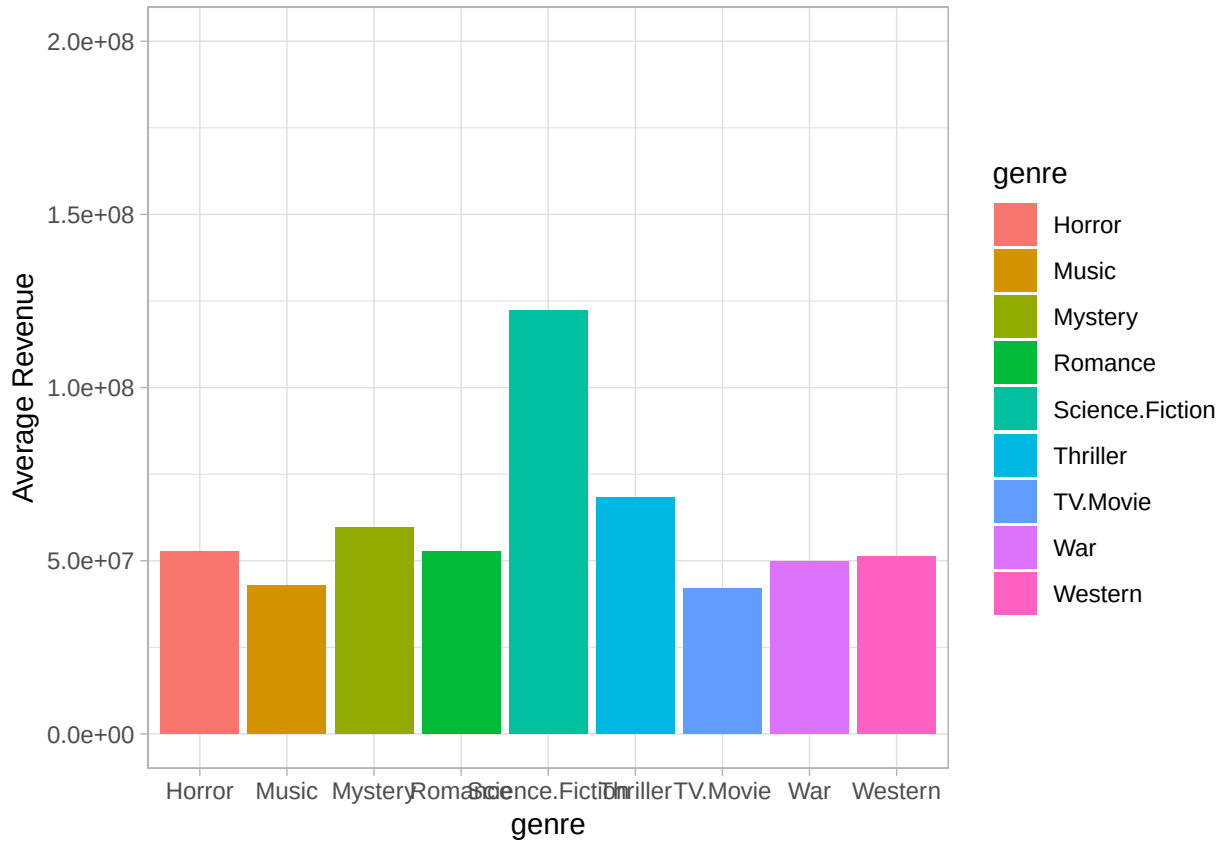
The relationship between genres and revenue:

```
#create a new data frame which contains the average revenue of each movie genre.
genre_mean=data.frame(matrix(ncol=2,nrow=0))
x=c("genre","mean.revenue")
colnames(genre_mean)=x
for (i in 20:30){
  gerne.mean=mean(train$revenue[train[i]==1])
  gerne=colnames(train[i])
  y=data.frame(gerne,gerne.mean)
  names(y)=x
  genre_mean=rbind(genre_mean,y)
}

genre_mean1=data.frame(matrix(ncol=2,nrow=0))
```

```r
colnames(genre_mean)=x
for (i in 31:39){
  gerne.mean=mean(train$revenue[train[i]==1])
  gerne=colnames(train[i])
  y=data.frame(gerne,gerne.mean)
  names(y)=x
  genre_mean1=rbind(genre_mean1,y)
}

ggplot(data=genre_mean, aes(x = genre,
             y = mean.revenue, fill=genre)) +
  geom_bar(stat = "identity") +
  theme_light() +
  ylim(0,200000000)+
  ylab("Average Revenue")
```

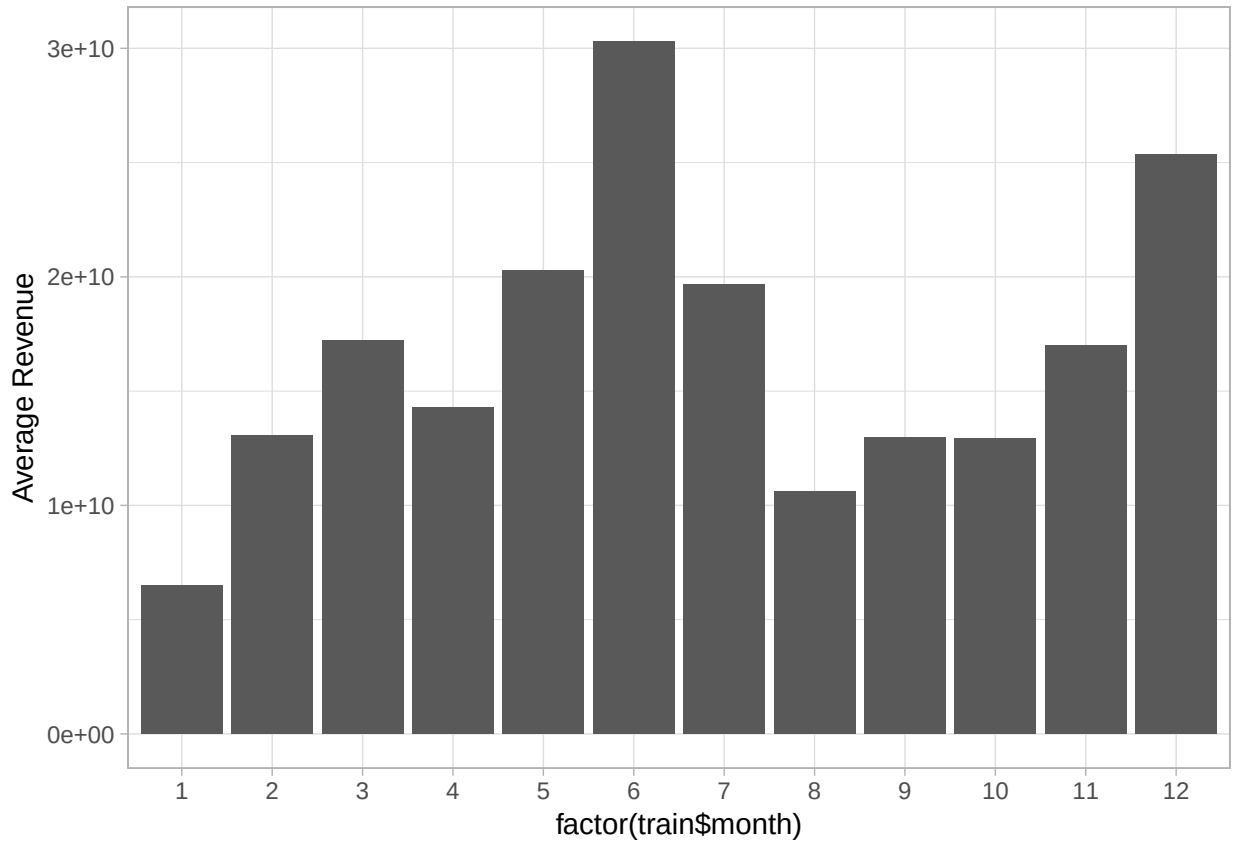



```r
ggplot(data=genre_mean1, aes(x = genre,
             y = mean.revenue, fill=genre)) +
  geom_bar(stat = "identity") +
  theme_light() +
  ylim(0,200000000)+
  ylab("Average Revenue")
```

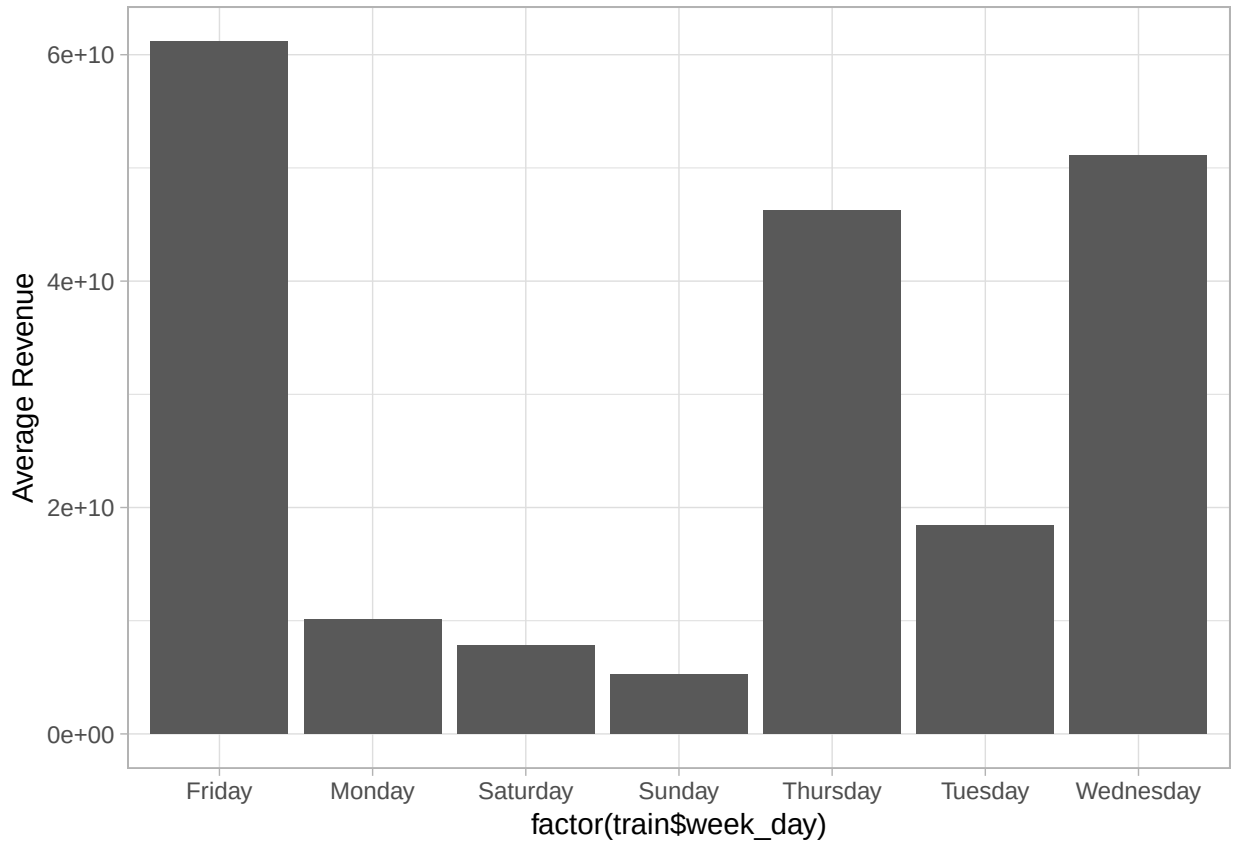


The relationship between release date and revenue:

```
#release month
ggplot(data=train, aes(x = factor(train$month),
             y = revenue,
             stat = "summary", fun.y = "mean")) +
  geom_bar(stat = "identity")+
  theme_light() +
  ylab("Average Revenue")
```



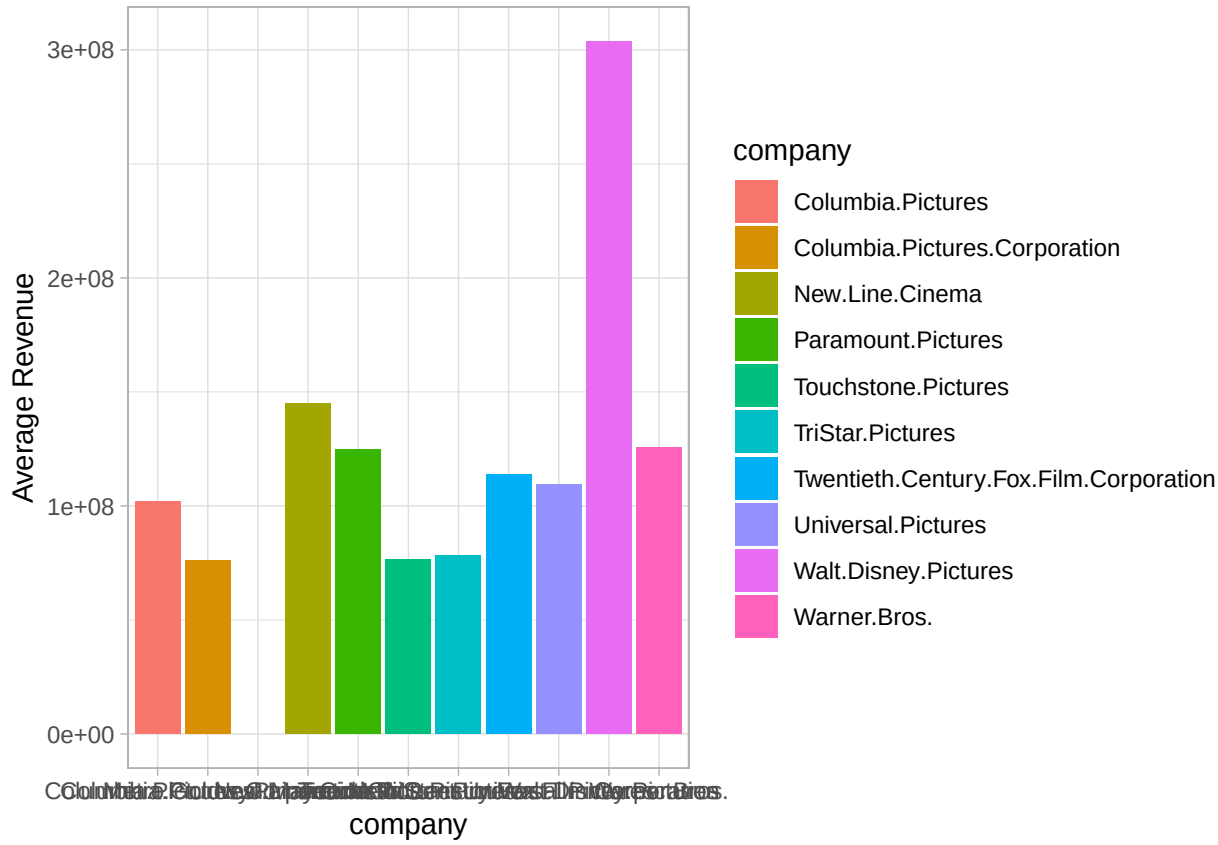

```
#release week_day
ggplot(data=train, aes(x = factor(train$week_day),
             y = revenue,
             stat = "summary", fun.y = "mean")) +
  geom_bar(stat = "identity")+
  theme_light() +
  ylab("Average Revenue")
```



The relationship between production company and revenue:

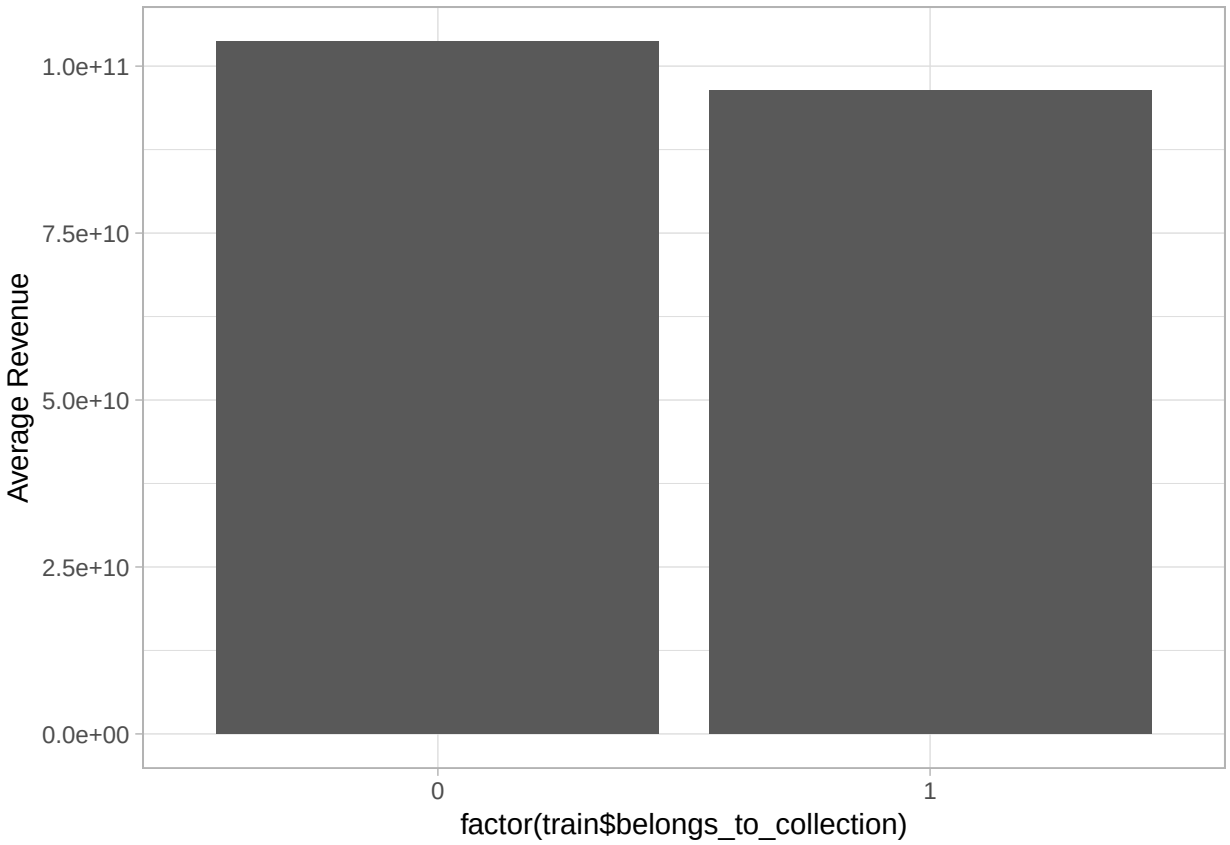
```
#create a new data frame which contains the average revenue of each movie producing company.
company_mean=data.frame(matrix(ncol=2,nrow=0))
x=c("company","mean.revenue")
colnames(genre_mean)=x
for (i in 43:53){
  company.mean=mean(train$revenue[train[i]==1])
  company=colnames(train[i])
  y=data.frame(company,company.mean)
  names(y)=x
  company_mean=rbind(company_mean,y)
}

#"Metro.Goldwyn.Mayer..MGM."
ggplot(data=company_mean, aes(x = company,
             y = mean.revenue, fill=company)) +
  geom_bar(stat = "identity") +
  theme_light() +
  ylab("Average Revenue")
```

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



The relationship between whether belongs to a collection and revenue:

```
ggplot(data=train, aes(x = factor(train$belongs_to_collection),
             y = revenue,
             stat = "summary", fun.y = "mean")) +
  geom_bar(stat = "identity")+
  theme_light() +
  ylab("Average Revenue")
```


1.0e+11
7.5e+10
5.0e+10
2.5e+10
0.0e+00
Average Revenue
0
1
factor(train$belongs_to_collection)