TMDB 5000 Data Analyze Report

Yujing Song

May 03, 2019

Abstract

With the popularity of entertainment, film has become a part of life as an important entertainment item. Diversified market demand pushes film producers in to a dilemma because of the difficulty of making a popular movie. This paper tries to figure out what influence the quality of movies and what types of films are popular. The analysis will give a general guidance of shooting a good film.

Outline

1. Introduction

- Background
- Problems statement

2. Data exploration

- Data description
- Load data

3. Data cleaning
4. Analyzing and visualization

- Regional distribution of high-quality movies
- The change of movie types over time
- Audiences' key words preference
- Who is the most attention worthy high-quality movies' director
- Relation among budgets movie scores and product studios
- Factors influence the quality of a film

5. Conclusion
6. Future works
7. Bibliography

1. Introduction

1.1 Background

Film, as one of most popular art expressions, has become an important entertainment carrier and even an indispensable part of our daily life. The explosive development of film industry attracts increasing more attention from all groups of people. At the same time, film can also be regarded as a form of investment, which is extremely dependent on capital and industrial level. Therefore, analyzing film data can help to catch the trend of the development of film industry.

With the development of film commercialization, people gradually became tired with stylized films. This analysis attempts to understand the factors that make up a popular film and to visualize them. We will discuss the geographical distribution of high-quality films, what factors influence the popularity of a film, and trends of film types over time.

1.2 Problem statements

The biggest point in our research is to find what makes a good film. In order to understand the meaning of "good" for a film, the following perspectives were analyzed:

- What is the regional distribution of global high-quality films?
- How does the types of film change over time?
- What are the audience keyword preferences for a film?
- How do we judge whether a film is worth watching based upon its director?
- What are the relationship among budgets, movie scores and film studio?
- What factors are influence the quality and profit of a film most?

2. Data exploration

2.1 Dataset description

The data is based on the website: The Movie DB (the, 2018), and the datasets are available in Kaggle (kag, 2018).

The datasets originally contains 24 variables for 4803 movies, but with the process of transferring the data from JSON form to csv form, I deleted 7 unrelated or repetitive variables. The transfer processes are shown in "TMDB trans try.ipynb"(Attached file 1) . The result transfered file 'movie_data.csv' is constituted by 4803 movies with 17 variables, cross 100 years and thousands of actors. 'vote_average' will be the variable used to judge the quality of a movie.

Variable	Description
budget	Budget in the movie in dollars.
genres	Film categorization.
keywords	Keywords describing the movie plot.
original_language	Language used in the film
original_title	Title of the movie
popularity	Relative page views on the Movie Database
production_companies	Companies produced the movie
producion_countries	Countries produced the movie
release_date	Release date of the movie
revenue	The total revenue of the movie
runtime	Duriation in minutes
spoken_languages	Languages used in the movie
tagline	The tagline of the movie
vote_average	The average score for the movie in the Movie Database
vote_count	The amount of people who graded the movie

Variable	Description
cast	The actors/actresses in the movie
director	The director of the movie

2.2 Load data

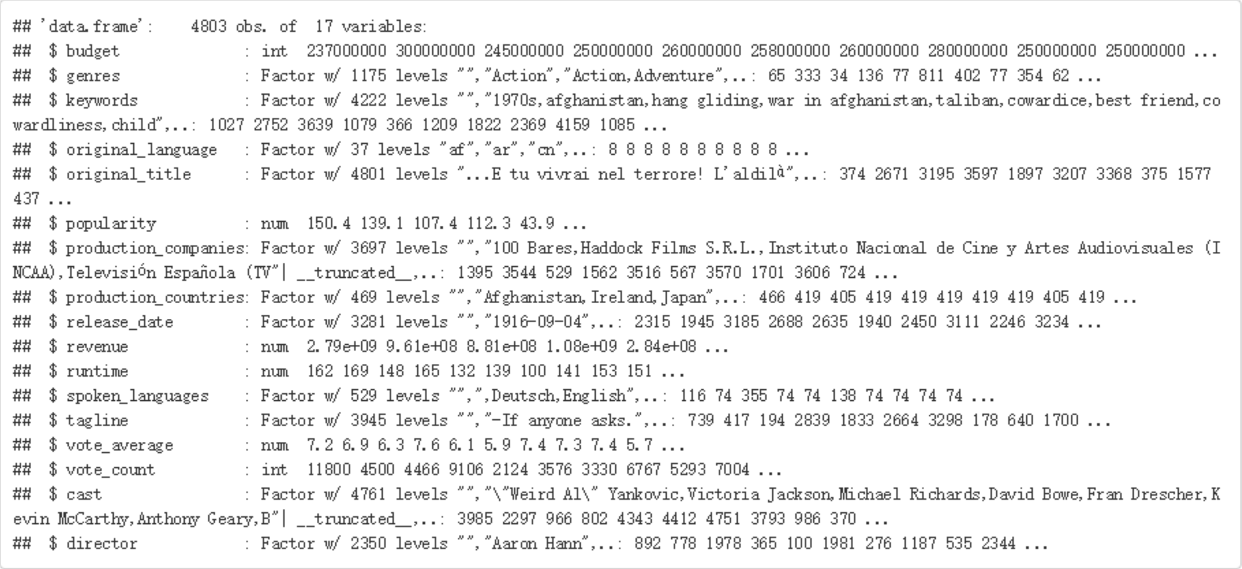
Load our packages first to get ready for analyzing.

```
# Load packages
library(ggplot2)
library(ggrepel)
library(ggthemes)
library(scales)
library(dplyr)
library(VIM)
library(data.table)
library(formattable)
library(plotly)
library(corrplot)
library(GGally)
library(caret)
library(car)
library(maps)
library(mapproj)
library(stringr)
library(tidyr)
library(rpart)
library(rpart.plot)
```

Take a look at the structure of our data:

```
TMDB = read.csv("movie_data.csv")
```

```
str(TMDB)
```



```
## 'data.frame':    4803 obs. of  17 variables:
##  $ budget              : int  237000000 300000000 245000000 250000000 260000000 258000000 260000000 280000000 250000000 250000000 ...
##  $ genres              : Factor w/ 1175 levels "","Action","Action,Adventure",..: 65 333 34 136 77 811 402 77 354 62 ...
##  $ keywords            : Factor w/ 4222 levels "","1970s,afghanistan,hang gliding,war in afghanistan,taliban,cowardice,best friend,co
wardliness,child",..: 1027 2752 3639 1079 366 1209 1822 2369 4159 1085 ...
##  $ original_language   : Factor w/ 37 levels "af","ar","cn",..: 8 8 8 8 8 8 8 8 8 8 ...
##  $ original_title      : Factor w/ 4801 levels "...E tu vivrai nel terrore! L'aldilà",..: 374 2671 3195 3597 1897 3207 3368 375 1577
437 ...
##  $ popularity          : num  150.4 139.1 107.4 112.3 43.9 ...
##  $ production_companies: Factor w/ 3697 levels "","100 Bares,Haddock Films S.R.L.,Instituto Nacional de Cine y Artes Audiovisuales (I
NCAA),Televisión Española (TV"| __truncated__,..: 1395 3544 529 1562 3516 567 3570 1701 3606 724 ...
##  $ production_countries: Factor w/ 469 levels "","Afghanistan,Ireland,Japan",..: 466 419 405 419 419 419 419 419 405 419 ...
##  $ release_date        : Factor w/ 3281 levels "","1916-09-04",..: 2315 1945 3185 2688 2635 1940 2450 3111 2246 3234 ...
##  $ revenue             : num  2.79e+09 9.61e+08 8.81e+08 1.08e+09 2.84e+08 ...
##  $ runtime             : num  162 169 148 165 132 139 100 141 153 151 ...
##  $ spoken_languages    : Factor w/ 529 levels "",",Deutsch,English",..: 116 74 355 74 74 138 74 74 74 74 ...
##  $ tagline             : Factor w/ 3945 levels "","-If anyone asks.",..: 739 417 194 2839 1833 2664 3298 178 640 1700 ...
##  $ vote_average        : num  7.2 6.9 6.3 7.6 6.1 5.9 7.4 7.3 7.4 5.7 ...
##  $ vote_count          : int  11800 4500 4466 9106 2124 3576 3330 6767 5293 7004 ...
##  $ cast                : Factor w/ 4761 levels "","\"Weird Al\" Yankovic,Victoria Jackson,Michael Richards,David Bowe,Fran Drescher,K
evin McCarthy,Anthony Geary,B"| __truncated__,..: 3985 2297 966 802 4343 4412 4751 3793 986 370 ...
##  $ director            : Factor w/ 2350 levels "","Aaron Hann",..: 892 778 1978 365 100 1981 276 1187 535 2344 ...
```

We imported 4803 movies with 17 variables, these variables in the types of factors or numbers or integers.

Number of levels of some factors are inaccurate, such as "keywords", "production_companies", "procuction_countries", and "cast". It is because these variables contains more than one element in a single table cell. But we do not deal with these variables until we need to use them to analyse related problems.

3. Data cleaning

Though we already transform our original data into a readable format, missing data or unreadable data may still exist. Cleaning useless data and trying to keep the integrity of the original data are the main goals in this section.

```r
sum(duplicated(TMDB))
```

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

Noticed that "spoken_languages" contains some unreadable code, and this is the column we will not use in later analyzing, we decide to remove that.

```r
TMDBset = TMDB[ , !names(TMDB) %in% c("spoken_languages")]
```

Some titles of movies take a special character "Â" in the original name and whitespaces, it may be caused by the different types of files when collecting data. It is better for us to remove these meaningless characters.

```r
TMDBset$original_title <- gsub("Â", "", as.character(factor(TMDBset$original_title)))
```

Then, let's check if there is any missing data.

```r
#try to find missing data
colSums(sapply(TMDBset, is.na))
```

```
##               budget               genres             keywords 
##                    0                    0                    0 
##    original_language       original_title           popularity 
##                    0                    0                    0 
## production_companies production_countries         release_date 
##                    0                    0                    0 
##              revenue              runtime              tagline 
##                    0                    2                    0 
##         vote_average           vote_count                 cast 
##                    0                    0                    0 
##             director 
##                    0
```

```
#get the position of missing data
which(is.na(TMDBset),arr.ind = T)
```

```
##       row col
## [1,] 2657  11
## [2,] 4141  11
```

The result above show that there are only 2 missing data in the column “runtime”. Actually, there are more missing data than what we have above. The 2 in the summary above indicates that there are 2 “NA” values in the column “runtime”, but there are other ways in which the data can be inappropriate. For example, the value “budget” takes some unusual low budgets, like 1 dollars or 2 dollars. These extremely low value are also counted as missing data, but we will deal with them latter when analyzing these data. Focus on “runtime” missing data, it is not a big missing amount, we can try to find the runtime online and fill them out.

```
#find the movie title of missing runtime
TMDBset[c(2657,4141),c(5,11)]
```

```
##                                   original_title runtime
## 2657 Chiamatemi Francesco - Il Papa della gente      NA
## 4141                  To Be Frank, Sinatra at 100      NA
```

```
#put online runtime info in the table
TMDBset[2657,11] <- 113
TMDBset[4141,11] <- 81
TMDBset[c(2657,4141),c(5,11)]
```

```
##                                   original_title runtime
## 2657 Chiamatemi Francesco - Il Papa della gente     113
## 4141                  To Be Frank, Sinatra at 100      81
```

In fact, besides “runtime”, we can still find that there are a lot of 0 in the column “revenue”. Check how many zero are there in total.

```
sum(TMDB$revenue==0)
```

```
## [1] 1427
```

1427 movies’ revenues are 0 for some reasons. There are two possible reasons. One, these films do not take enough revenue to record. Two, hard to get the revenue data because of the regions differences. Usually, since the amount of the zero revenue is not a small number, we can delete this column, or try to use the average value or medium to replace these missing values. (Wackerly *et al.*, 2008)

Some times we choose to delete the rows with missing data or delete the column if it has a large missing data. However, considering that "revenue" is an important potential variable and the other potential variables are not containing a lot missing data, we will not remove this column and any rows of it. In view of a huge gap between the revenue of different films, we will not replace these zero revenues with the average or a medium number of the revenue.

At this point, we have completed the overall cleaning of original data. We will partition variables and set many sub-tables when analyzing specific questions. Cleaning data in detail will happen at that time.

4. Analyzing and visualization

In the previous section the data was cleaned. In this section, we try to answer the questions we have in Section 1.2 by analyzing related variables.

4.1 Regional distribution of high-quality movies

First of all we need to define what is a high-quality movie. Here our "high-quality" based on the values in column "vote_average". We are trying to set an integer as the threshold value to define high-quality movies. Quartile is applied in this situation, and the integer that is closest to the third quartile is used as the threshold value.

Quartile is "one of the three points that divide a range of data or population into four equal parts" (Hyndman *et al.*, 1996). The third quartile is the number that greater or equal to 75% of movies' score, and we believe that higher than 75% means good enough to represent high-quality movies.

```
summary(TMDBset$vote_average)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   5.600   6.200   6.092   6.800  10.000
```

The third quartile of "vote_average" is 6.8, therefore, the closest integer is 7. So, decide to choose 7 as the threshold value.

Then we created a new table for filtered high-quality movies.

```
#create a table for high-quality movie:
high_quality= subset(TMDBset,vote_average >= 7)
length(high_quality$vote_average)
```

```
## [1] 988
```

Only 988 movies have a score greater than 7. Now we want to know the countries distribution of these movies, but noticed that some movies are made by more than one country, each countries is separated by a comma. Since we want to count high-quality movies for each country, and check if high-quality movies has relationship with regions, we save the average score and countries' information as a new frame.

```r
hq.country <- as.data.frame(high_quality[,c("production_countries","vote_average")])
#separate different countries into new columns
seperator <- function(table_name,col_length,sep_var){
  sapply(1:col_length, function(x) if (table_name[x,1] %like% sep_var) 1 else 0)
}

hq.country$Afghanistan<- seperator(hq.country,
                      length(hq.country$production_countries),"Afghanistan")
hq.country$Japan<- seperator(hq.country,
                      length(hq.country$production_countries),"Japan")
hq.country$Ireland<- seperator(hq.country,
                      length(hq.country$production_countries),"Ireland")
hq.country$Algeria<- seperator(hq.country,
                      length(hq.country$production_countries),"Algeria")
hq.country$USA<- seperator(hq.country,
      length(hq.country$production_countries),"United States of America")
hq.country$Argentina<- seperator(hq.country,
                    length(hq.country$production_countries),"Argentina")
hq.country$Denmark<- seperator(hq.country,
                    length(hq.country$production_countries),"Denmark")
hq.country$Finland<- seperator(hq.country,
                    length(hq.country$production_countries),"Finland")
hq.country$France<- seperator(hq.country,
                    length(hq.country$production_countries),"France")
hq.country$Germany<- seperator(hq.country,
                    length(hq.country$production_countries),"Germany")
hq.country$Iceland<- seperator(hq.country,
                    length(hq.country$production_countries),"Iceland")
hq.country$Italy<- seperator(hq.country,
                    length(hq.country$production_countries),"Italy")
hq.country$Netherlands<- seperator(hq.country,
                    length(hq.country$production_countries),"Netherlands")
hq.country$Norway<- seperator(hq.country,
                    length(hq.country$production_countries),"Norway")
hq.country$Sweden<- seperator(hq.country,
                    length(hq.country$production_countries),"Sweden")
hq.country$UK<- seperator(hq.country,
                    length(hq.country$production_countries),"United Kingdom")
hq.country$Spain<- seperator(hq.country,
                    length(hq.country$production_countries),"Spain")
hq.country$Aruba<- seperator(hq.country,
                    length(hq.country$production_countries),"Aruba")
hq.country$HK<- seperator(hq.country,
                    length(hq.country$production_countries),"Hong Kong")
hq.country$Australia<- seperator(hq.country,
                    length(hq.country$production_countries),"Australia")
hq.country$Belgium<- seperator(hq.country,
                    length(hq.country$production_countries),"Belgium")
hq.country$India<- seperator(hq.country,
                    length(hq.country$production_countries),"India")
hq.country$Canada<- seperator(hq.country,
                    length(hq.country$production_countries),"Canada")
hq.country$China<- seperator(hq.country,
                    length(hq.country$production_countries),"China")
hq.country$Taiwan<- seperator(hq.country,
                    length(hq.country$production_countries),"Taiwan")
hq.country$Thailand<- seperator(hq.country,
                    length(hq.country$production_countries),"Thailand")
hq.country$UAE<- seperator(hq.country,
                    length(hq.country$production_countries),"United Arab Emirates")
hq.country$Romania<- seperator(hq.country,
                    length(hq.country$production_countries),"Romania")
hq.country$SouthAfrica<- seperator(hq.country,
                    length(hq.country$production_countries),"South Africa")
hq.country$Russia<- seperator(hq.country,
                    length(hq.country$production_countries),"Russia")
hq.country$Nigeria<- seperator(hq.country,
                    length(hq.country$production_countries),"Nigeria")
hq.country$Morocco<- seperator(hq.country,
                    length(hq.country$production_countries),"Morocco")
hq.country$Mexico<- seperator(hq.country,
                    length(hq.country$production_countries),"Mexico")
hq.country$Malta<- seperator(hq.country,
                    length(hq.country$production_countries),"Malta")
hq.country$CzechRepublic<- seperator(hq.country,
                  length(hq.country$production_countries),"Czech Republic")
hq.country$Chile<- seperator(hq.country,
                  length(hq.country$production_countries),"Chile")
hq.country$Hungary<- seperator(hq.country,
                  length(hq.country$production_countries),"Hungary")
hq.country$Luxembourg<- seperator(hq.country,
                  length(hq.country$production_countries),"Luxembourg")
hq.country$Egypt<- seperator(hq.country,
                  length(hq.country$production_countries),"Egypt")
hq.country$Colombia<- seperator(hq.country,
                  length(hq.country$production_countries),"Colombia")
hq.country$Kazakhstan<- seperator(hq.country,
                  length(hq.country$production_countries),"Kazakhstan")
hq.country$Poland<- seperator(hq.country,
                  length(hq.country$production_countries),"Poland")
hq.country$Philippines<- seperator(hq.country,
                  length(hq.country$production_countries),"Philippines")
```

```
hq.country$Peru<- seperator(hq.country,
                  length(hq.country$production_countries),"Peru")
hq.country$Panama<- seperator(hq.country,
                  length(hq.country$production_countries),"Panama")
hq.country$NewZealand<- seperator(hq.country,
                  length(hq.country$production_countries),"New Zealand")
hq.country$SouthKorea<- seperator(hq.country,
                  length(hq.country$production_countries),"South Korea")
hq.country$Lebanon<- seperator(hq.country,
                  length(hq.country$production_countries),"Kazakhstan")
hq.country$CzechRepublic<- seperator(hq.country,
                  length(hq.country$production_countries),"Czech Republic")
hq.country$Dominica<- seperator(hq.country,
                  length(hq.country$production_countries),"Dominica")
hq.country$Jamaica<- seperator(hq.country,
                  length(hq.country$production_countries),"Jamaica")
hq.country$Indonesia<- seperator(hq.country,
                  length(hq.country$production_countries),"Indonesia")
hq.country$Guyana<- seperator(hq.country,
                  length(hq.country$production_countries),"Guyana")
hq.country$Greece<- seperator(hq.country,
                  length(hq.country$production_countries),"Greece")
hq.country$Finland<- seperator(hq.country,
                  length(hq.country$production_countries),"Finland")
hq.country$Brazil<- seperator(hq.country,
                  length(hq.country$production_countries),"Brazil")
hq.country$Bhutan<- seperator(hq.country,
                  length(hq.country$production_countries),"Bhutan")
```

In this new frame, every country is extracted as a new column, if a country participates in the production of related film, we set the value as 1, otherwise the value is 0.

Get the top 30 countries with the largest amount of high-quality movies.

```
country <- hq.country[,3:57]
country_sum <- colSums(country)
head(sort(country_sum,decreasing=T),30)
```

```
##           USA            UK       Germany        France        Canada
##           767           169            78            74            39
##         Italy         Spain     Australia         India         Japan
##            30            26            17            17            15
##         China    NewZealand        Mexico       Ireland            HK
##            14            13            12            10            10
##        Sweden       Belgium       Denmark        Norway    SouthKorea
##             9             9             8             7             6
##        Brazil        Russia CzechRepublic     Argentina   Netherlands
##             6             5             5             4             4
##        Taiwan       Romania   SouthAfrica        Poland       Iceland
##              3             3             3             3             2
```

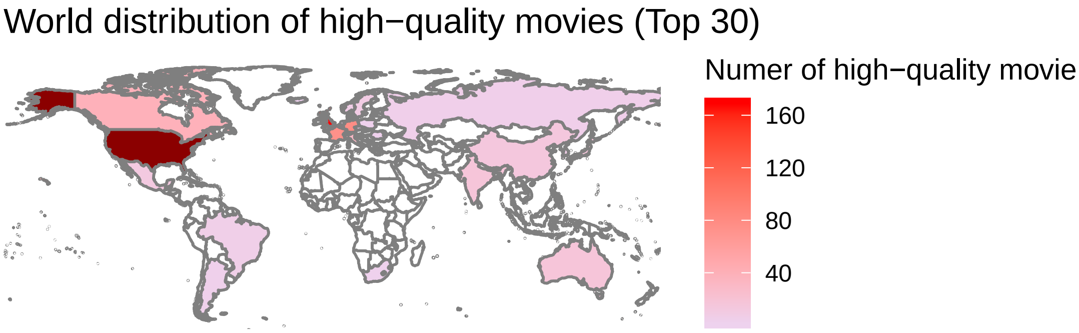
From the table above, USA take the first place by an overwhelming advantage. The overrepresentation of American films in the Internet reflects the universal existence of the American film industry (Wasserman *et al.*, 2015). UK, Germany and France also stand in an outstanding places. The amount for good movies from Canada, Italy, and Spain are impressive because they have over 25. The other countries seems didn't get a lot of recognition. In order to visualize their geographical distribution, we draw a high-quality movies map based on the frequency of their high-quality movies.

```
opts_chunk$set(eval=FALSE,message=FALSE,echo=TRUE)
WorldData <- map_data('world')
WorldData <- fortify(WorldData)


country_sum_df <- data.frame(region = c ('UK', 'Germany','France','Canada',
                                         'Italy','Spin','Australia','India',
                                         'Japan', 'China','NewZealand',
                                         'Mexico','Irelan','Hong Kong',
                                         'Sweden','Belgium','Denmark',
                                         'Norway', 'South Korea',
                                         'Brazil','Russia','Czech Republic',
                                         'Argentina','Netherlands','Taiwan',
                                         'Romania','South Africa',
                                         'Poland','Iceland'),
                             value = c (169,78,74,39,30,26,17,17,15,14,
                                        13,12,10,10,9,9,8,7,6,6,5,5,4,
                                        4,3,3,3,3,2))


p <- ggplot()
p <- p + geom_map(data=WorldData, map=WorldData,
                  aes(x=long, y=lat, group=group, map_id=region),
                  fill="white", colour="#7f7f7f", size=0.5)
```

```
## Warning: Ignoring unknown aesthetics: x, y
```

```
p <- p + geom_map(data=country_sum_df, map=WorldData,
                  aes(fill=value, map_id=region),
                  colour="#7f7f7f", size=0.5)

p <- p + coord_map("rectangular", lat0=0, xlim=c(-180,180), ylim=c(-60, 90))
p <- p + scale_fill_continuous(low="thistle2", high="red",
                               guide="colorbar")
p <- p + geom_map(data=WorldData, map=WorldData,
                  aes(map_id='USA'), fill = "darkred",
                  colour="#7f7f7f", size=0.5)

p <- p + scale_y_continuous(breaks=c())
p <- p + scale_x_continuous(breaks=c())
```

```
p <- p + labs(fill="Numer of high-quality movie",
      title="World distribution of high-quality movies (Top 30)", x="", y="")
p <- p + theme_bw()
p <- p + theme(panel.border = element_blank())
p
```


World distribution of high−quality movies (Top 30)
Numer of high−quality movie
160
120
80
40

The map shows the top 30 countries with the highest number of high-quality movies. The darker the shake of red in a region, the more high-quality movies were produced in that region. What we can see from the map is that the high-quality movies are mostly gathered in North America and Europe. Although not in large numbers, East Asia, Australia, and the south part of South America still have a place for high-quality movies.

In addition to the European and American countries film shooting technology is mature, the cause of above phenomenon may because the movie scores are rated by mostly English users, therefore movies from North America and Europe could be easier to hit their point of interests (Wasserman *et al.*, 2015). Besides, the database takes more American films than any other regions, which makes US movies have more possibilities to get higher grades.

4.2 The change of movie types over time

In this section, we will show how the types of movie change based on the time change. We will divided into three parts: split genres, re-organize the movie table, plot the changes of each type of movies.

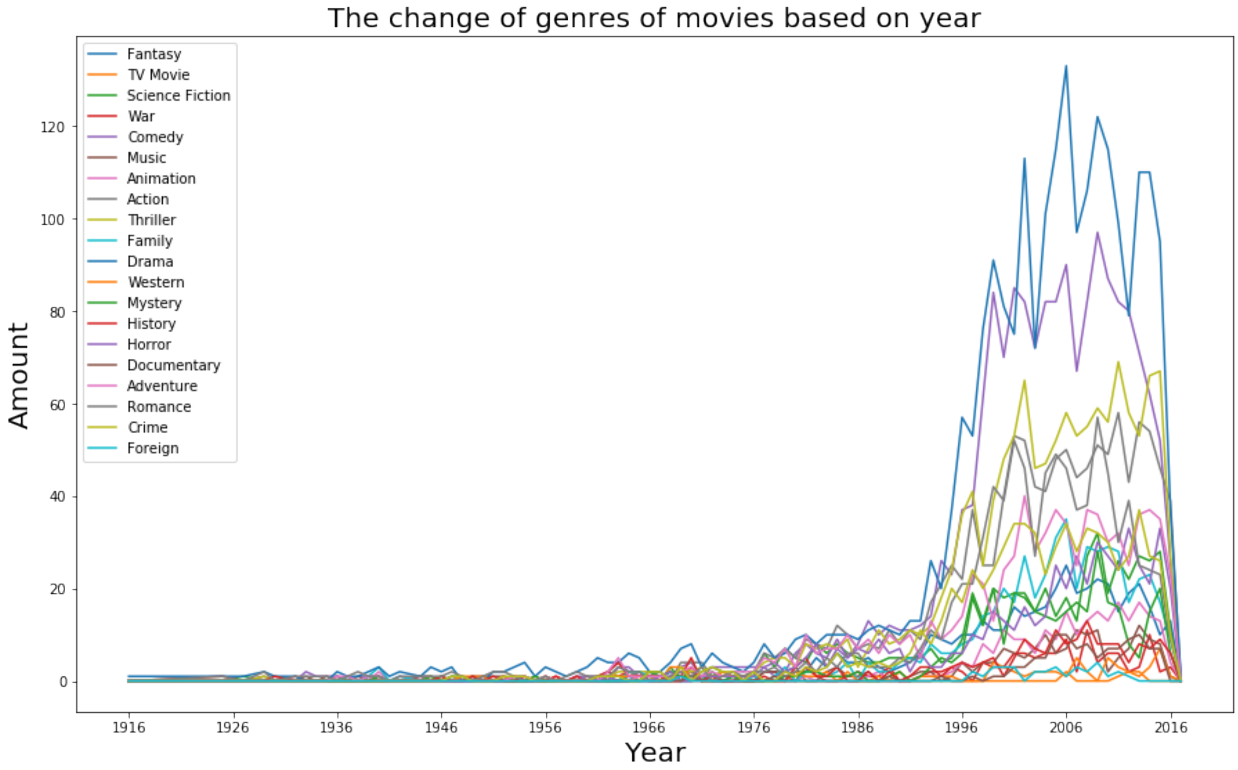
Load python package.

```
library(reticulate)
py_available()
use_virtualenv("r-reticulate")
```

The Python code below creates a time series genres plot. For more details about my python work in this part, please open “Movie types analyzing.ipynb” (Attached File 2).

```
# prepare the packages need to use later
import numpy as np
import pandas as pd

import warnings
warnings.filterwarnings('ignore')
```

```
import csv
import json
import matplotlib.pyplot as plt

import seaborn as sns
from PIL import Image

#read data from local directory
tmdbset = pd.read_csv("D:/undergraduate/2018fall/STAT senior sem/TMDBset.csv")
tmdbset.info()

#use split to split strings
tmdbset['genres'][1].split(',')
#but we need to fill out NAs
tmdbset['genres']=tmdbset['genres'].fillna('Unknown')


#use set() to create a set without elements repeat
genre = set()
for i in tmdbset['genres'].str.split(','):
    genre=set().union(i,genre)


#Then we got a list of all the genres
genre=list(genre)


#Transfer the date formate
tmdbset['release_year']=pd.to_datetime(tmdbset.release_date,
                        format='%Y-%m-%d').dt.year

# for loop: for each line,
#if contain the specific genre,
#return 1, otherwise return 0
for genr in genre:
    tmdbset[genr] = tmdbset['genres'].str.contains(genr).
                    apply(lambda x:1 if x else 0)


#create a table for genre and year
tmdbset_genre_year = tmdbset.loc[:, genre]
tmdbset_genre_year.index = tmdbset['release_year']
tmdbset_genre_year.head()


#count the amount of each genre for each year
tmdbset_genre_year_count = tmdbset_genre_year.groupby('release_year').sum()
tmdbset_genre_year_count = tmdbset_genre_year_count.drop(columns=['Unknown'])


# get the plot!
plt.figure(figsize=(15,9))
plt.plot(tmdbset_genre_year_count,label=tmdbset_genre_year_count.columns)
plt.legend(tmdbset_genre_year_count)
plt.xticks(range(1916, 2017,10))
plt.xlabel('Year',fontsize=20)
plt.ylabel('Amount',fontsize=20)
plt.title('The change of genres of movies based on year',fontsize=20)
plt.show()
```



In the big picture, various types of movies present blowout growth after 1996, and there is also a clear distinction in the number of different genres of films. From 1996 to 2006, the number of all types of movies fluctuated increasing, while after 2006, it showed a fluctuating declining trend.

The movie genre “Fantasy” has been in an outstanding popular place since 1946. Even over 120 movies were included in type “Fantasy” in 2006. “Comedy” experienced a rapid increase after 1996 and become the second popular type of movie. “Crime”, “Action”, “Romance” occupied a relatively important position in the film genres. “Animation”, “Thriller”, “Family”, “Mystery” performed normally. Other genres, although they had different levels of increase, they still could not become an important part of the film genres.

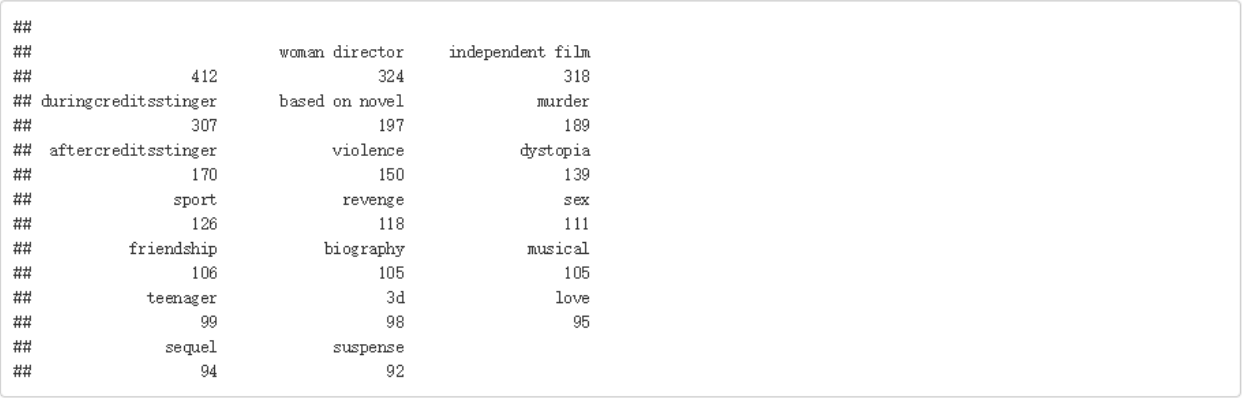
4.3 Audiences' key words preference

The key words are also a big part that the audiences pay attention to. This section counts the number of keywords people like so that we will know how to produce a popular movie. Firstly, we split keywords by its delimiter into several independent strings.

```
library(wordcloud2)
keywords_list = str_split(as.character(TMDBset$keywords),",")
```

Sort these keywords by a decreasing order of frequency to check what are the popular keywords.

```
keywords_table = table(unlist(keywords_list))
head(sort(keywords_table,decreasing = T),20)
```

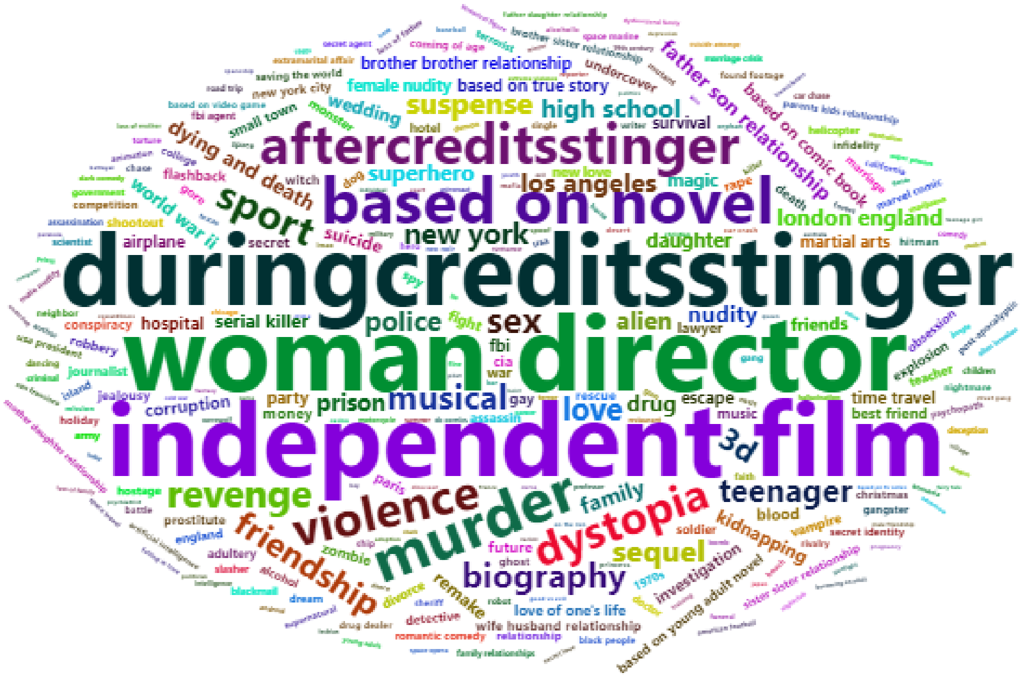


```
##
##                          woman director    independent film
##                  412               324                 318
## duringcreditsstinger    based on novel              murder
##                  307               197                 189
##  aftercreditsstinger          violence            dystopia
##                  170               150                 139
##                sport           revenge                 sex
##                  126               118                 111
##           friendship         biography             musical
##                  106               105                 105
##             teenager                3d                love
##                   99                98                  95
##               sequel          suspense
##                   94                92
```

Blank has 412 frequency means 412 missing data.

In order to intuitively understand the popular keywords, we create a world cloud.

```
keywords_table_top300 = head(sort(keywords_table,decreasing = T),300)
wordcloud2(keywords_table_top300, size = 0.4)
```



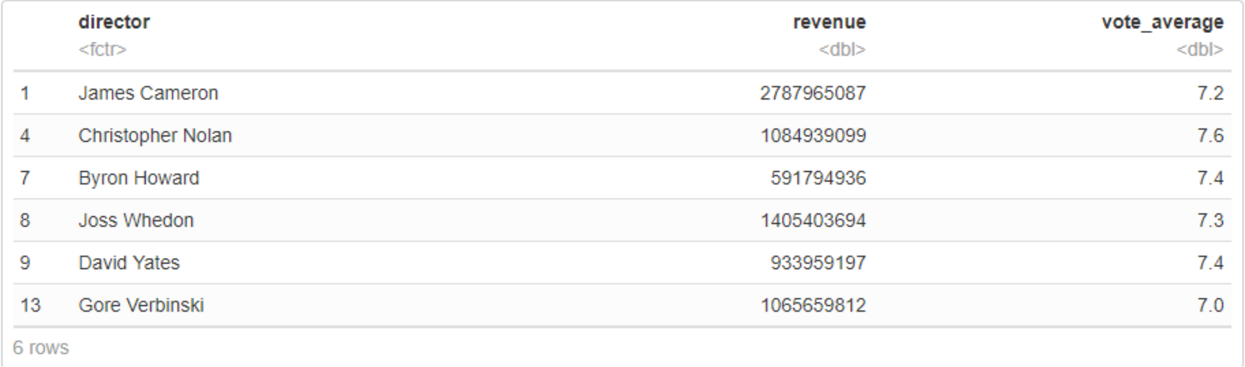
As we can see, three striking keywords in the word cloud "woman director", "independent film", and "credits stinger". With the rise of the women's movement, "woman director" has a compelling position. Keyword "independent films" is popular because it tend to express the director's personal ideas rather than the commercial market. Film watchers usually like to have extra information or surprises from movies, therefore "credits stinger" is also a significant keywords. Movies "based on novel" is a good strategy to attract a certain audiences who have read the novels before. More keywords like "murder","violence", "dystopia" are worthy of note as well, these keywords satisfy people's curiosity to some extent.

4.4 Who is the most attention worthy high-quality movies' director

The director is the organizer and leader of the artistic creation of films. The films presented by different film directors usually have their own unique styles. A good director greatly increases the chances of producing a good film, therefore, here we are trying to find directors that make high-quality movies.

Start with creating a table for high-quality movies' director and its revenue.

```
d_revenue_score = as.data.frame(high_quality[,c("director","revenue",
                                                "vote_average")])
head(d_revenue_score)
d_revenue_score = d_revenue_score[which(d_revenue_score$director != ""),]
d_revenue_score = d_revenue_score[which(d_revenue_score$revenue != 0),]
```

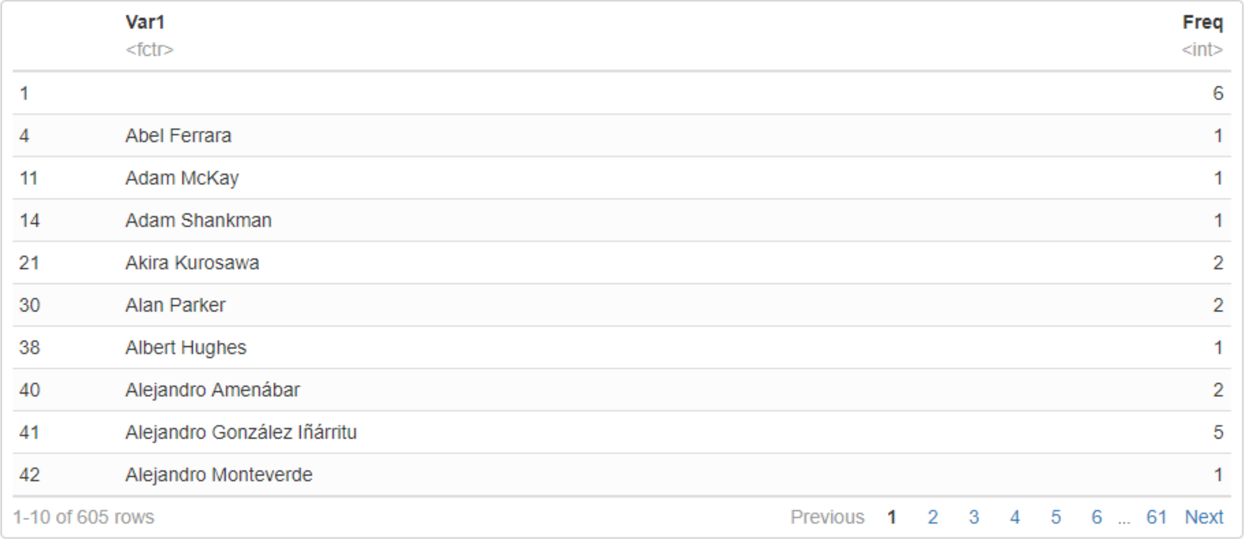


	director <fctr>	revenue <dbl>	vote_average <dbl>
1	James Cameron	2787965087	7.2
4	Christopher Nolan	1084939099	7.6
7	Byron Howard	591794936	7.4
8	Joss Whedon	1405403694	7.3
9	David Yates	933959197	7.4
13	Gore Verbinski	1065659812	7.0

6 rows

Check if there is any duplicated directors:

```
director_dup =as.data.frame(table(d_revenue_score$director))
director_dup[which(director_dup[,2]>0),]
```

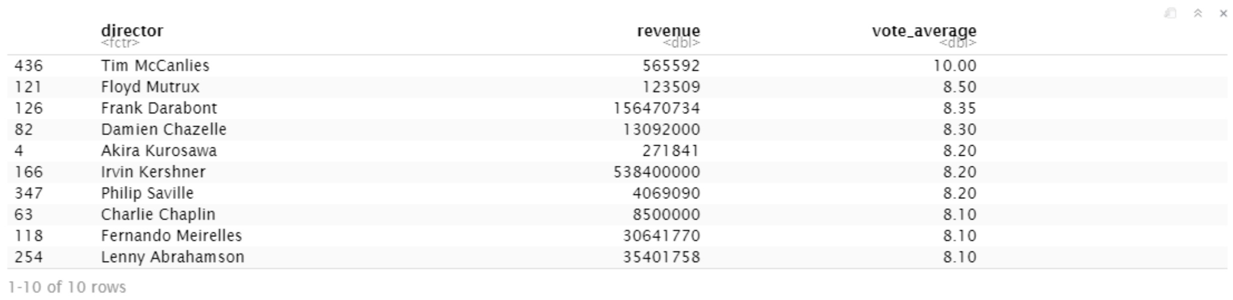


	Var1 <fctr>	Freq <int>
1		6
4	Abel Ferrara	1
11	Adam McKay	1
14	Adam Shankman	1
21	Akira Kurosawa	2
30	Alan Parker	2
38	Albert Hughes	1
40	Alejandro Amenábar	2
41	Alejandro González Iñárritu	5
42	Alejandro Monteverde	1

1-10 of 605 rows Previous 1 2 3 4 5 6 ... 61 Next

From the table above, it looks like we do have some directors directed more than one movies. We count the mean of the movies' revenue of each director, and list the top 10 directors with highest score.

```
quality_dir_top10 = head(good_dir[order(good_dir[,3],decreasing = T),],10)
quality_dir_top10
```

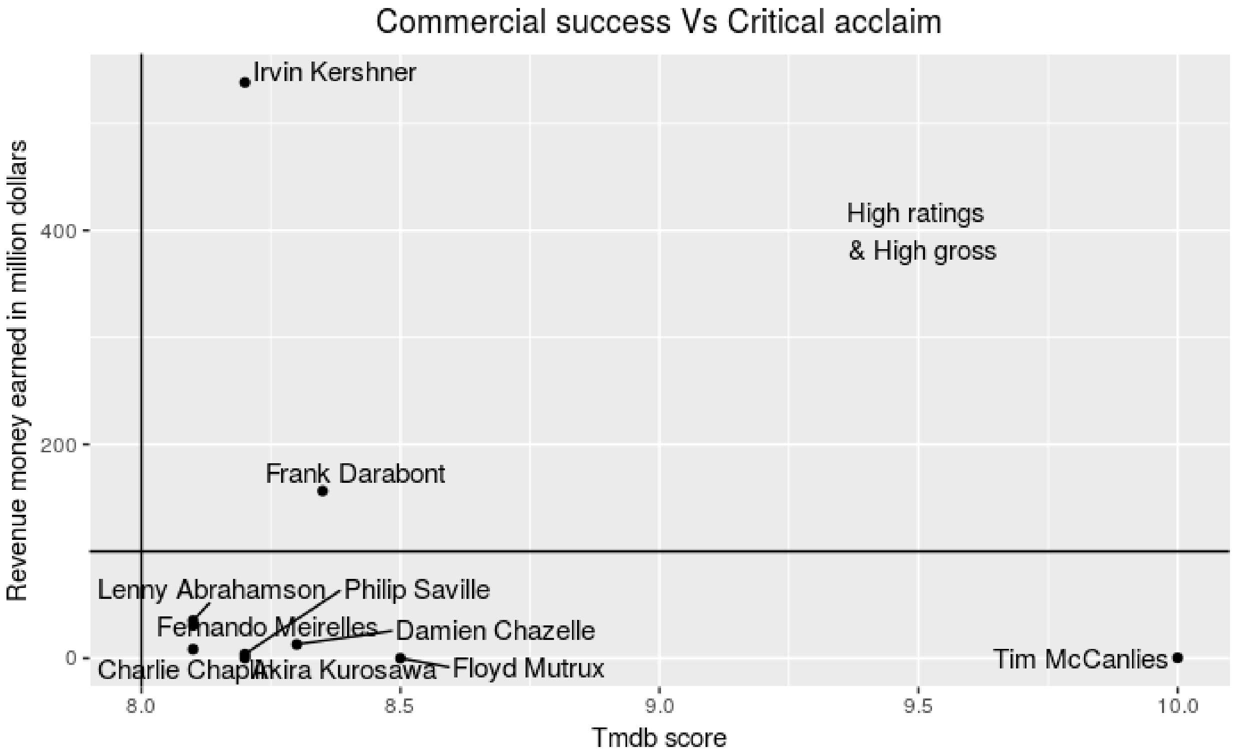


	director <fctr>	revenue <dbl>	vote_average <dbl>
436	Tim McCanlies	565592	10.00
121	Floyd Mutrux	123509	8.50
126	Frank Darabont	156470734	8.35
82	Damien Chazelle	13092000	8.30
4	Akira Kurosawa	271841	8.20
166	Irvin Kershner	538400000	8.20
347	Philip Saville	4069090	8.20
63	Charlie Chaplin	8500000	8.10
118	Fernando Meirelles	30641770	8.10
254	Lenny Abrahamson	35401758	8.10

1-10 of 10 rows

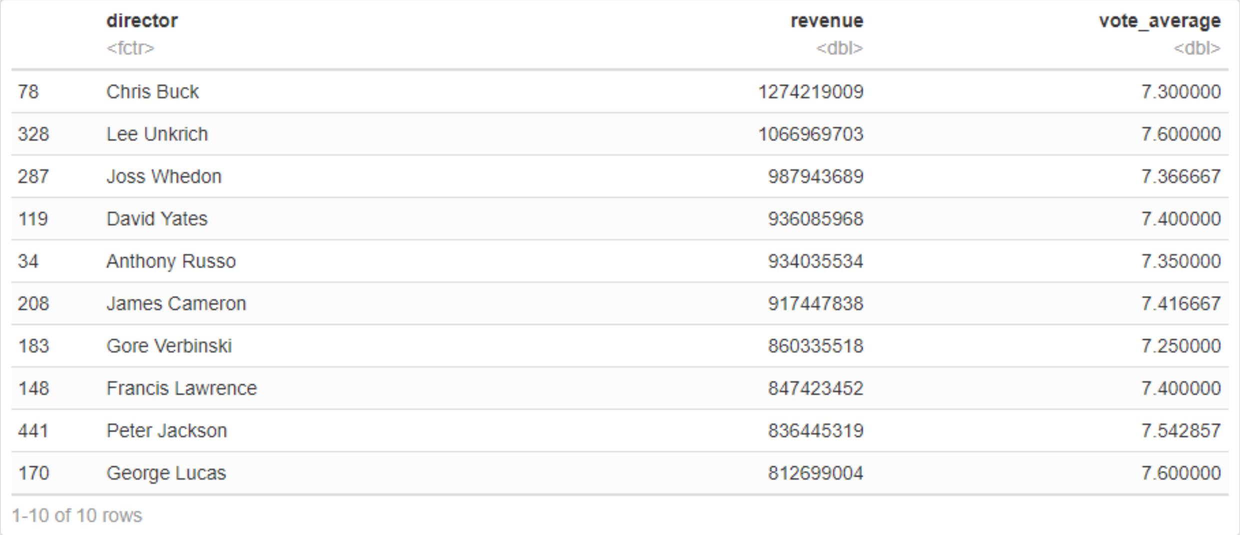
This 10 directors got the highest average score in TMDB 5000 database, but we noticed that their revenues are vary different. Therefore, we draw a plot for this ten directors to directly observe their differences.

```
quality_dir_plot = data.frame(director = quality_dir_top10$director,
                              revenue = quality_dir_top10$revenue,
                              vote_average = quality_dir_top10$vote_average)
ggplot(quality_dir_plot, aes(x = vote_average, y = revenue/10^6)) +
  geom_point() +
  geom_hline(aes(yintercept = 100)) +
  geom_vline(aes(xintercept = 8)) +
  geom_text_repel(aes(label = director), size = 4) +
  xlab("Tmdb score") +
  ylab("Revenue money earned in million dollars") +
  ggtitle("Commercial success Vs Critical acclaim") +
  annotate("text", x = 9.5, y = 400, label = "High ratings \n & High gross") +
  theme(plot.title = element_text(hjust = 0.5))
```



The plot above shows the relationship between rating and gross among top 10 high-quality movie directors. Based on a movie research by Wasserman M, we set the horizontal line that is 100 billion dollars, which stands for the high grossing line (Wasserman *et al.*, 2015). The vertical line is set as 8, if a movie has score over 8, we define it as a master work. Irvin Kershner and Frank Darabont are two noticeable directors that in the high-ratings and high gross area, but the other 8 directors are gathered in high-rating but not high-gross area which represent they have a master work but did not have a high revenue in their movies. Especially for Tim McCanlies, his movie get the full mark 10, but only have 500 thousand dollars' revenue. If you are an audience who is looking for a extremely high quality movie, movies directed by Tim McCanlies will be a good choice. If you are a movie producer who cares revenue more than quality but still want to make your movie stay in high-quality, following is the director list you may want to study from them.

```r
good_dir=aggregate(d_revenue_score[,c(2,3)],
                   by =list(director= d_revenue_score$director), mean)
good_dir_top10 = head(good_dir[order(good_dir[,2],decreasing = T),],10)
good_dir_top10
```



	director <fctr>	revenue <dbl>	vote_average <dbl>
78	Chris Buck	1274219009	7.300000
328	Lee Unkrich	1066969703	7.600000
287	Joss Whedon	987943689	7.366667
119	David Yates	936085968	7.400000
34	Anthony Russo	934035534	7.350000
208	James Cameron	917447838	7.416667
183	Gore Verbinski	860335518	7.250000
148	Francis Lawrence	847423452	7.400000
441	Peter Jackson	836445319	7.542857
170	George Lucas	812699004	7.600000

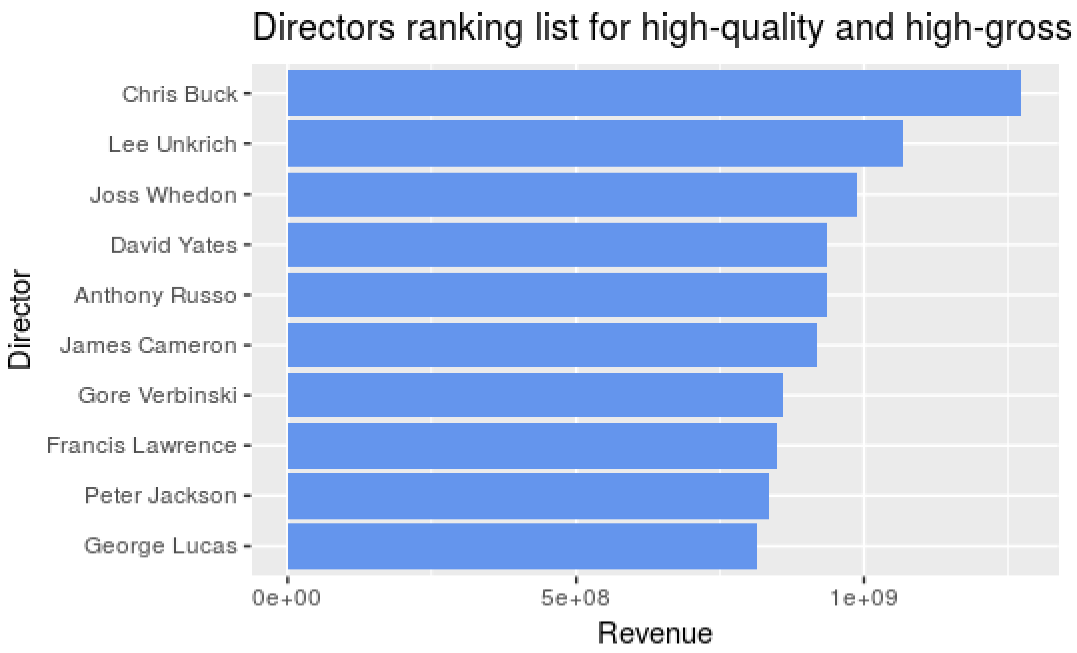
1-10 of 10 rows

```r
diff(range(good_dir_top10$vote_average))
```

```
## [1] 0.35
```

The average scores of directors are not showing big differences, since the range of "vote_average" is 0.35. That is to say there is a huge differences in movies' quality. Then, we get closer to the revenue:

```r
good_dir_plot = data.frame(director = good_dir_top10$director,
                           revenue = good_dir_top10$revenue)
ggplot(good_dir_plot,aes(x=reorder(director, revenue), y=revenue)) +
geom_bar(stat = "identity",fill = "cornflowerblue") +
coord_flip() +
xlab("Director") +
ylab("Revenue") +
ggtitle("Top 10 highest revenue directors in quality films")
```



From the point of view of the audience, it is a good choice to go to movie theaters and choose films made by directors with high-quality and high-grossing. 10 directors are selected from above.

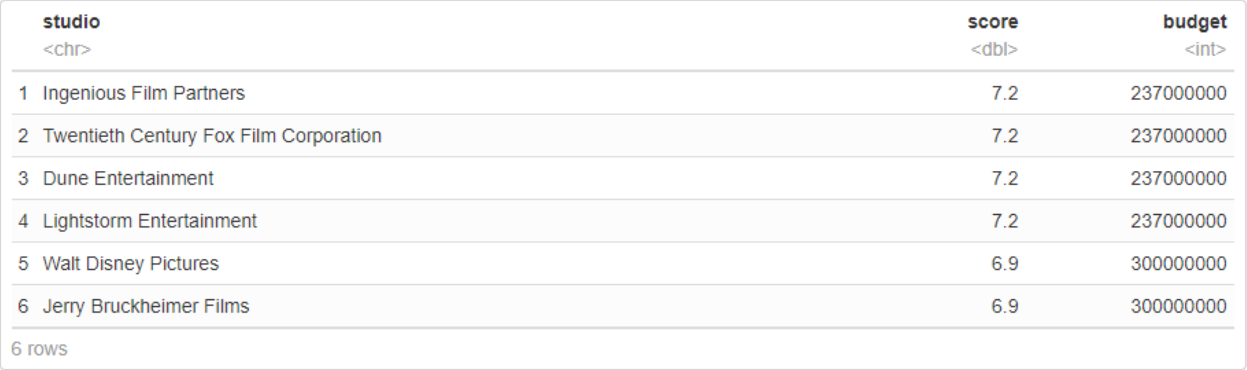
If new generation of directors want to rapidly enter into the film market, they can start with learning from Chris Buck, Lee Unkrich, and Joss Whedon because they are gross oriented directors, and Irvin Kershner and Frank Darabont are your goal. But if you are a quality oriented director, Tim McCanlies is a good teacher for you.

4.5 Relation among budgets, movie scores and product studios

There is a long-standing problem: does a high quality film have to have a high budget? The relationship between budgets and movie scores will be discussed in this section. Two parts are included. The first part is to reorganize the budget and score based on each product company. The second part draws a plot to explore the correlation between budget and scores.

We still start with splitting the product studios and change them into an appropriate form.

```
#create a table just for product studios, movie scores, and budgets
origion_studio_score_budget = data.frame(studio = TMDBset$production_companies,
      score = TMDBset$vote_average, budget = TMDBset$budget)
#separate a column with more than 1 studios into rows
studio_score_budget = separate_rows(origion_studio_score_budget,studio,sep=",")
head(studio_score_budget)
```



	studio <chr>	score <dbl>	budget <int>
1	Ingenious Film Partners	7.2	237000000
2	Twentieth Century Fox Film Corporation	7.2	237000000
3	Dune Entertainment	7.2	237000000
4	Lightstorm Entertainment	7.2	237000000
5	Walt Disney Pictures	6.9	300000000
6	Jerry Bruckheimer Films	6.9	300000000

6 rows

Then reorganize studios' information by creating a table contains the average of budget and score of each product studio.

```
summary_studio=aggregate(studio_score_budget[,c(2,3)],
           by =list(studio= studio_score_budget$studio),mean)
summary(summary_studio)
```

```
##     studio               score           budget
##  Length:5026        Min.   : 0.000   Min.   :0.00e+00
##  Class :character   1st Qu.: 5.700   1st Qu.:2.00e+06
##  Mode  :character   Median : 6.300   Median :1.30e+07
##                     Mean   : 6.193   Mean   :2.55e+07
##                     3rd Qu.: 6.800   3rd Qu.:3.50e+07
##                     Max.   :10.000   Max.   :2.80e+08
```

```
sum(summary_studio$score==0)
```

```
## [1] 27
```

```
sum(summary_studio$budget==0)
```

```
## [1] 829
```

Based on the summary above, we noticed that there are 27 movies with score 0 and 829 movies with budget 0. The movies with 0 score is not a lot, it may because some movies lack of people to judge it and give it a valuable score. But the number of 0-budget movies are minor, and the 1st quartile is larger than the minimum too much, so we need to count the outliers by quartile.

```
multi_diff_Q1_Q3 = (35000000 - 2000000)*3
minimum= 2000000- multi_diff_Q1_Q3
maximum= 35000000 + multi_diff_Q1_Q3
minimum
```

```
## [1] -9.7e+07
```

```
maximum
```

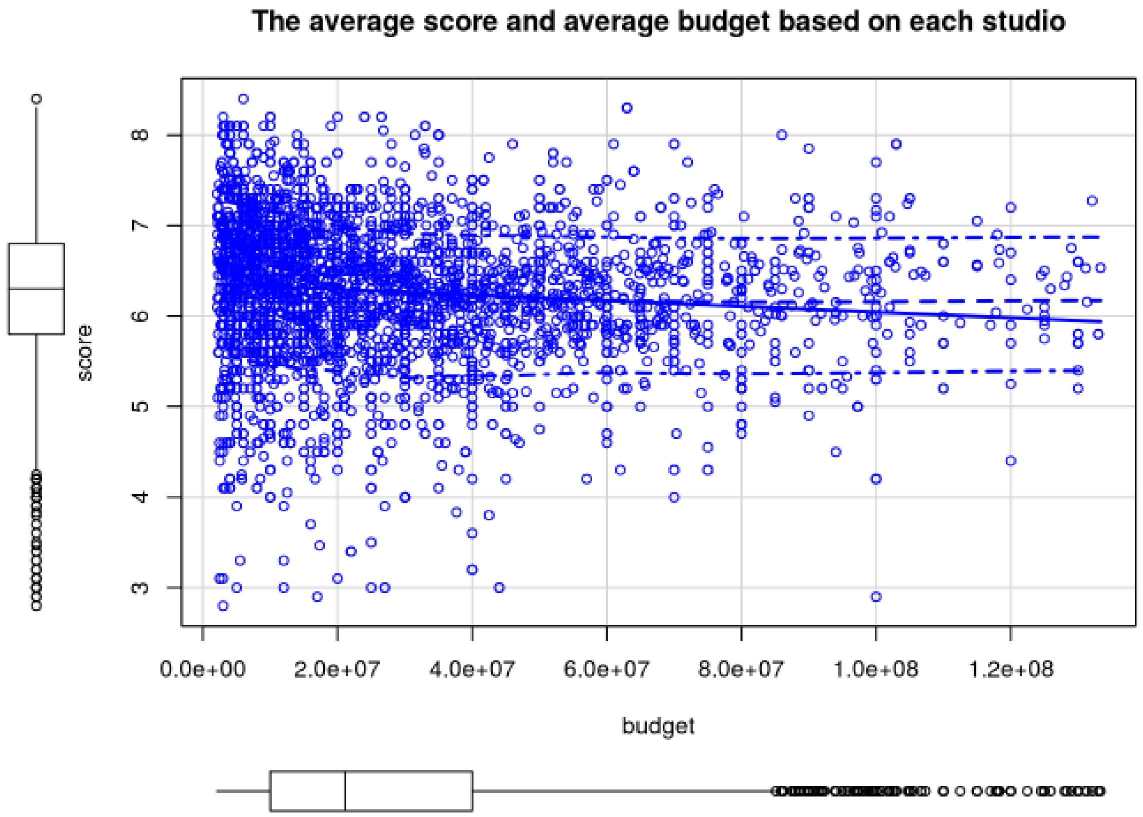
```
## [1] 1.34e+08
```

Though the 99% confidence interval include 0 dollar (Wackerly *et al.*, 2008), negative budgets are unreasonable. The distribution of the budget is right-skewed, which means there are a lot of movies gathered in low budgets, the number of movies is decreased with the increasing of budgets. Though there is no denying that very few low-budget films have done well, data like 1 dollar or 2 dollars are obviously incorrect for the budget. Combining with the average reasonable cost for a movie is 45 million dollars (Karniouchina and V., 2011), we set the lowest reasonable budget is our first quartile, 2 million dollars.

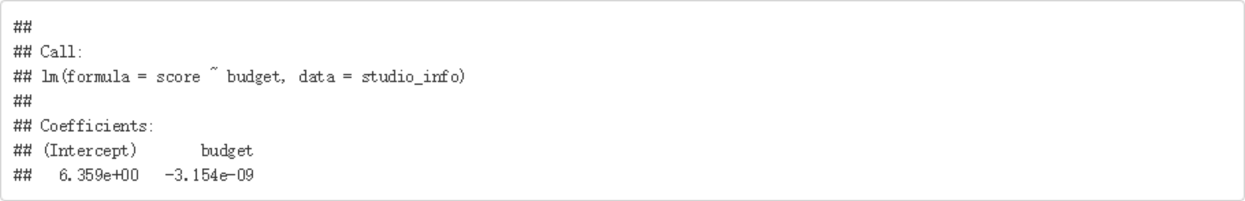
```r
sum(summary_studio$budget<2000000 & summary_studio$budget >0)
```

```
## [1] 389
```

```r
studio_info= summary_studio[which(summary_studio$score>0 ),]
studio_info= studio_info[which(studio_info$budget >2000000
                              & studio_info$budget < 134000000),]

#Draw a scatterplot
scatterplot(studio_info$score~studio_info$budget,
            main = "The average score and average budget based on each studio",
            xlab = "budget", ylab = "score")
```



```r
lm(score~budget, data = studio_info)
```

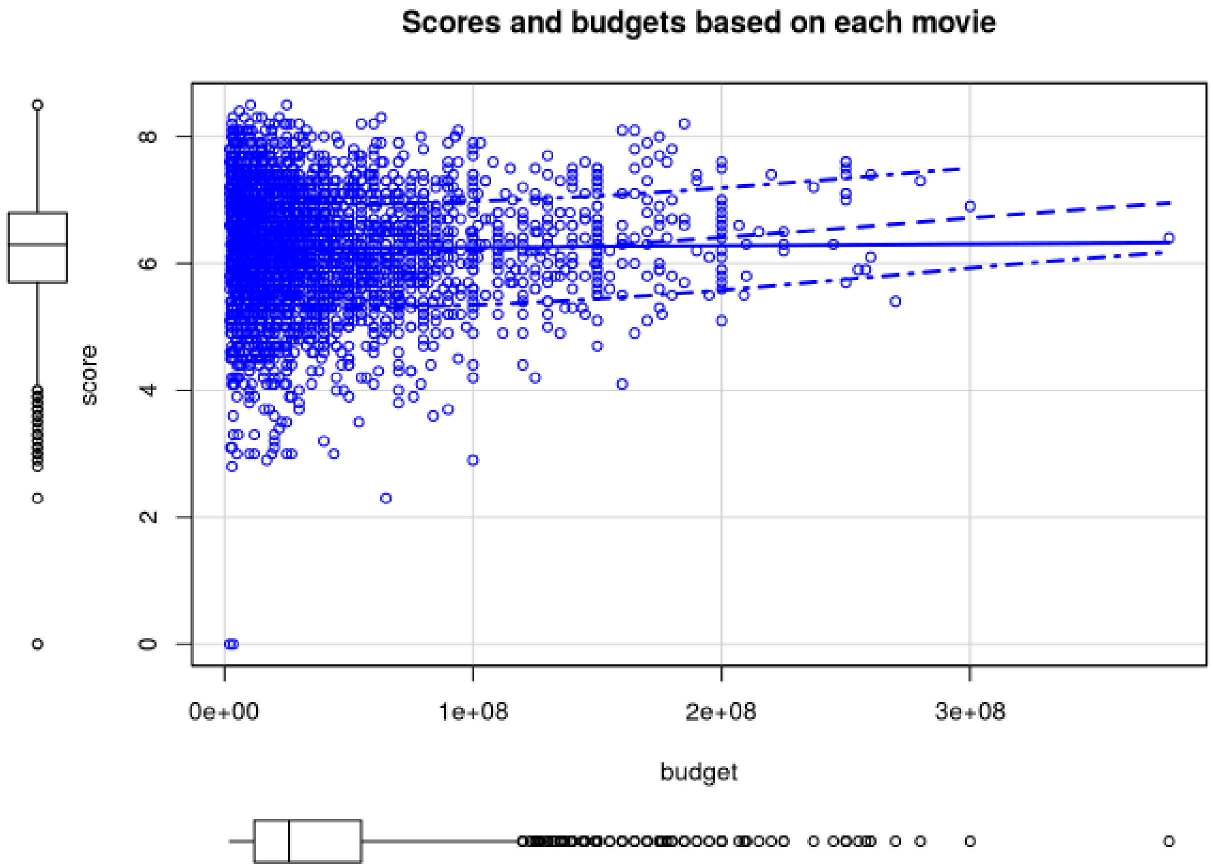


```
##
## Call:
## lm(formula = score ~ budget, data = studio_info)
##
## Coefficients:
## (Intercept)       budget
##   6.359e+00   -3.154e-09
```

The linear regression model of the score and the budget with the coefficient -0.000000000439. That is to say, score and budget have a very tiny negative relationship. But this relationship is too weak to call it a "relationship". In general from the scatter plot above, no matter whether the budget is extremely high or low, the score is stable around 6, so there is no obvious relationship between budgets and score. But the variance for the high budget is slightly smaller than the low budget, that is to say, though the difference is not large, higher budget movies are more concentrated in a stable score. When shooting a movie, reducing the cost as much as possible will not have a strong influence for the quality of the movie.

For some extra curiosity, the following lead us to get the scatterplot that shows the correlation of scores and budgets based on movies.

```
film_studio_score_budget = data.frame(score = TMDBset$vote_average,
                                      budget = TMDBset$budget)
film_studio_score_budget =
  film_studio_score_budget[which(film_studio_score_budget$budget >2000000),]

scatterplot(film_studio_score_budget$score~film_studio_score_budget$budget,
            main = "Scores and budgets based on each movie",
            xlab = "budget", ylab = "score")
```



```
linearMo = lm(score~budget, data = film_studio_score_budget)
linearMo
```

```
##
## Call:
## lm(formula = score ~ budget, data = film_studio_score_budget)
##
## Coefficients:
## (Intercept)       budget
##   6.223e+00    2.839e-10
```

Similar with what we observed based on produce studios, there is no obvious linear regression relationship of these two. But an interesting phenomenon is that even though higher budget cause fewer movies, movies with higher budget are more stable around score 6.

4.6 Other factors influence the quality of a film

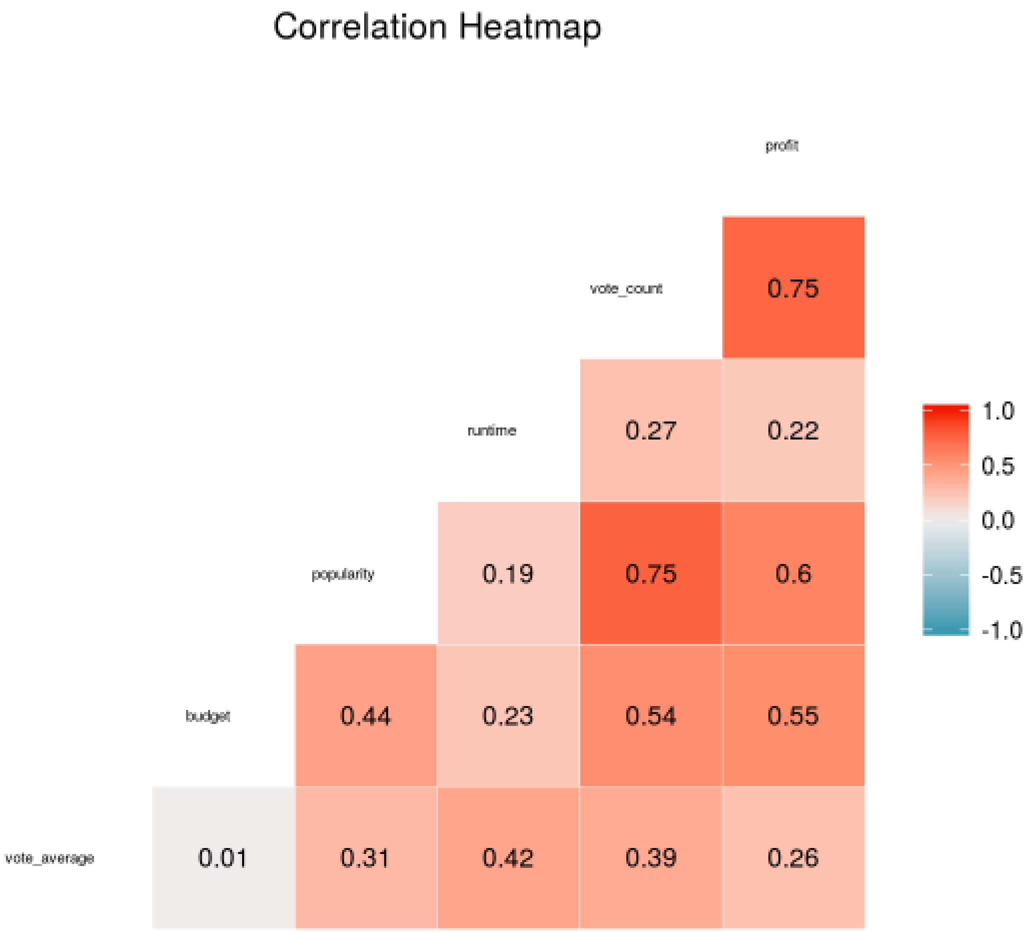
We already analyzed the relationship between “vote_average” and related discrete variables. Now in this section, we extract numerical variables in the “tmdbset” and try to figure out if there are more relationships between movies’ quality and these numerical variables. A classification tree will be created to see how these other numerical variables influence the “vote_average”.

Extract all the numerical variables.

```
tmdbNumSet = subset(TMDBset,select = c(13,1,6,10,11,14))
#delete the rows with budget less than 2000000 dollars
tmdbNumSet = tmdbNumSet[which(tmdbNumSet$budget >2000000),]
```

Draw a heatmap to see the correlation among variables. Heatmap is an analytical method to mark and present areas on a graph according to the degree of concern by using different symbols (Stanton, 2012). In our case, the color of the crossed area between two variables represents the degree of correlation between the two variables. Red represents positive relation whereas blue represents negative relation. The darker of the color it has, the stronger relationship between related two variables.

```
#creat a column for profit
tmdbNumSet$profit = tmdbNumSet$revenue - tmdbNumSet$budget
tmdbNumSet_modi = tmdbNumSet[,c(1,2,3,5,6,7)]
#draw heatpot
ggcorr(tmdbNumSet_modi, label = TRUE, label_round = 2, label_size = 3.5,
       size = 2, hjust = .85) +
  ggtitle("Correlation Heatmap") +
  theme(plot.title = element_text(hjust = 0.5))
```



The last row of heatmap shows the relevancy between "vote_average" and other variables. I did not see any high correlation (greater than 0.8) between predictors above, so we are going to use these variables as predictor.

For understanding simpler, “vote_average” is divided into 4 parts, 0~4, 4~6, 6~8, and 8~10. These small intervals represent movies are “bad”, “normal”, “good”, “masterwork”, respectively.

```
tmdbNumSet_modi$binned_score <- cut(tmdbNumSet_modi$vote_average,
                                    breaks = c(0,4,6,8,10))
```

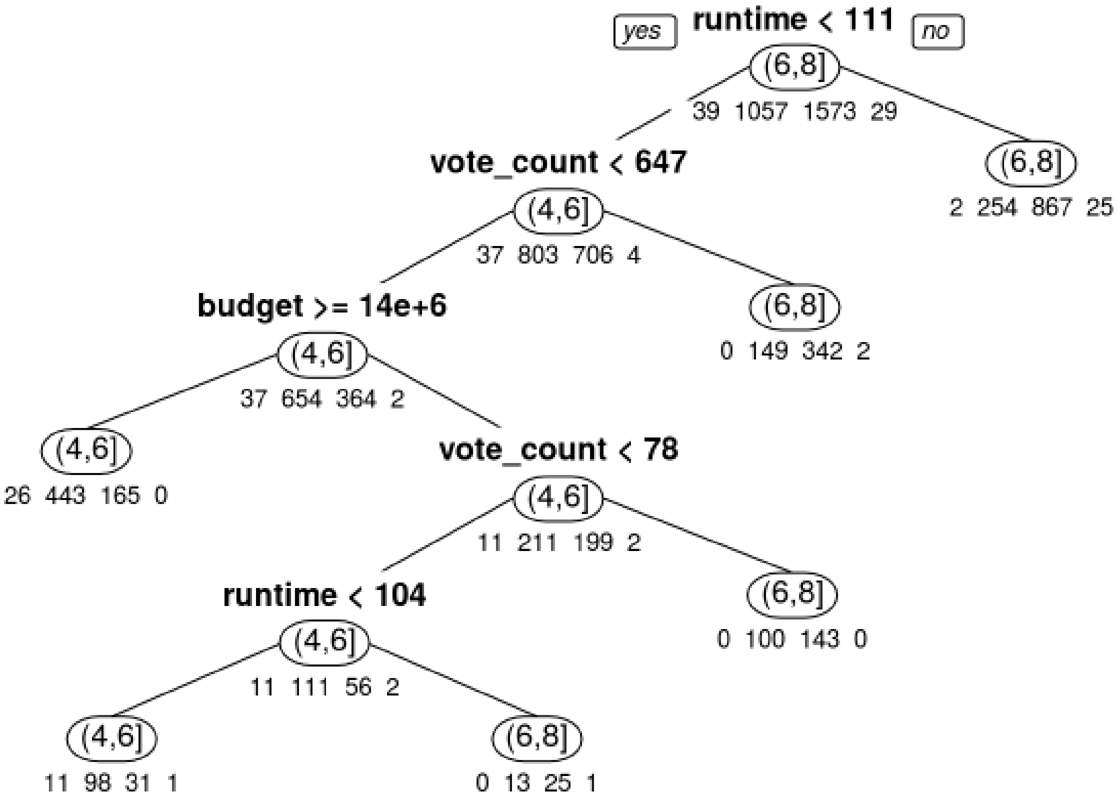
The training set is to create a model by training existed data. After the model training is completed, the parameters of the trained model are obtained. We need a testing set to test the accuracy of the model we have trained. Usually, when using a dataset to establish a model, about 80% of total data are put in training set and 20% of total data are put in testing set. (Goodfellow *et al.*, 2016)

In our project, we randomly picked up 2700 rows (79% of total data) from table “tmdbNumSet_modi” into training set, and the rest of data in that table are selected in testing set.

```
set.seed(5)
training_index = sample(nrow(tmdbNumSet_modi), 2700)
training = tmdbNumSet_modi[training_index,]
testing = tmdbNumSet_modi[-training_index,]
```

Plot a basic tree by using data in the training set:

```
# Full grown tree
class_tree <- rpart(binned_score ~ . -vote_average,
                    data = training, method = "class")
## plot tree
prp(class_tree, type = 1, extra = 1, under = TRUE, split.font = 2, varlen = 0)
```



The tree above provides a basic classification rules:

- If (runtime >= 111) then class = (6,8].
- If (runtime < 111) and (vote_count >= 647) then class = (6,8].
- If (runtime < 111) and (vote_count < 647) and (budget >= 14e+6) then class = (4,6].
- If (runtime < 111) and (78 <= vote_count < 649) and (budget < 14e+6) then class = (6,8].
- If (runtime < 104) and (vote_count < 78) and (budget < 14e+6) then class = (4,6].
- If (104 <= runtime < 111) and (vote_count < 78) and (budget < 14e+6) then class = (6,8].

According to these rules, we can conclude that within a certain time limit, if a movie has a longer runtime, it may get higher score. If it cannot last over 111 minutes, then the movies that vote a lot on the TMDB website tend to score higher, because many fans of popular movies will vote for them with high scores.

Our previous tree is not a tree that will be used as our model, instead, we are going to find a best-pruned tree by cross-validation procedure. The purpose of pruning is to avoid overfitting of the decision tree model. In the process of learning from training data, in order to classify training samples as accurately as possible, the decision tree algorithm keeps dividing nodes, which will lead to too many branches of the whole tree and lead to overfitting (Goodfellow *et al.*, 2016) . Cross validation is used to verify the performance of the classifier a statistical analysis method, the basic idea is to apart the original data as part of the training set, the other part as a Validation set, first with the training set to train a classifier in the Validation set is used to test the model obtained from the training, to be used as evaluation of performance of the classifier (Goodfellow *et al.*, 2016). The best-pruned tree will be find by finding the lowest cross-validation error.

For the coding in the chunk below, ‘cp’ is complexity parameter, which is set the threshold to use for pruning. ‘minsplit’ is minimum number of branch nodes, ‘xval’ is the number of cross-validation. Values that I used for these arguments are the optimum values that obtained after many trials.

```
set.seed(2)
cv_ct <- rpart(binned_score ~ . -vote_average,
               data = training, method = "class",
               cp = 0.00001, minsplit = 5, xval = 5)
printcp(cv_ct)
```

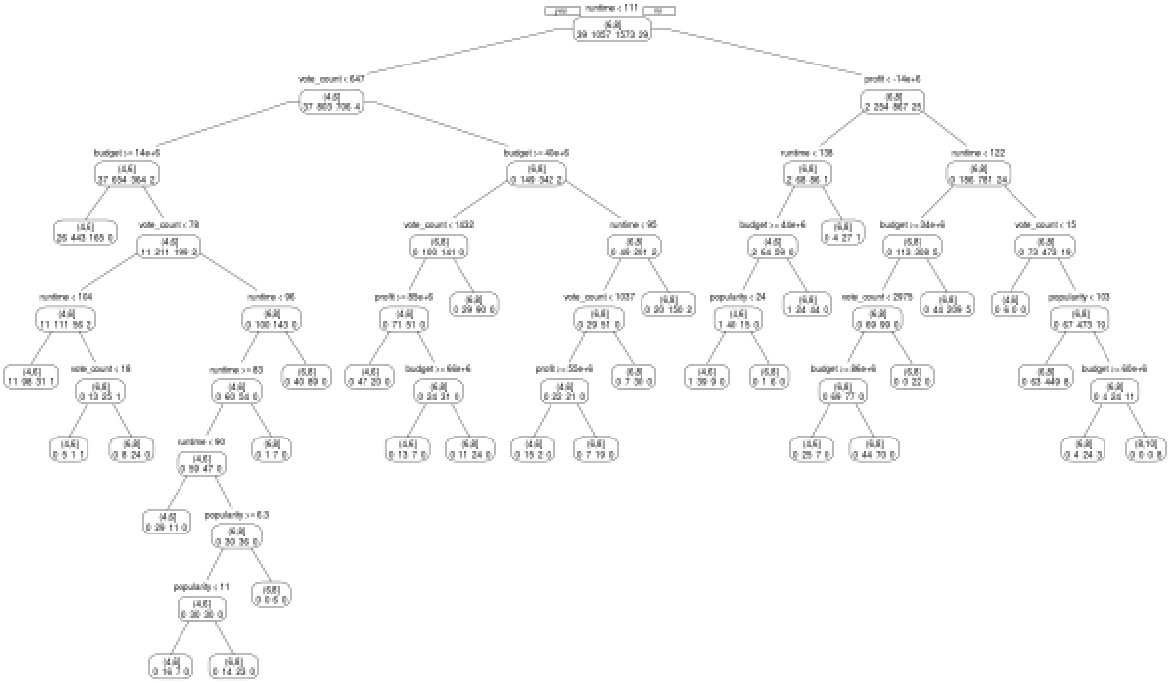
```
##
## Classification tree:
## rpart(formula = binned_score ~ . - vote_average, data = training,
##     method = "class", cp = 1e-05, minsplit = 5, xval = 5)
##
## Variables actually used in tree construction:
## [1] budget     popularity profit     runtime    vote_count
##
## Root node error: 1125/2698 = 0.41698
##
## n=2698 (2 observations deleted due to missingness)
##
##            CP nsplit rel error  xerror     xstd
## 1  0.12888889      0   1.00000 1.00000 0.022765
## 2  0.01911111      2   0.74222 0.75200 0.021421
## 3  0.01066667      4   0.70400 0.73600 0.021294
## 4  0.00888889      5   0.69333 0.72978 0.021244
## 5  0.00740741      7   0.67556 0.71378 0.021110
## 6  0.00622222     10   0.65333 0.71200 0.021095
## 7  0.00533333     11   0.64711 0.71022 0.021080
## 8  0.00444444     15   0.62578 0.71733 0.021140
## 9  0.00426667     16   0.62133 0.71467 0.021118
## 10 0.00400000     21   0.60000 0.71556 0.021125
## 11 0.00385185     23   0.59200 0.71556 0.021125
## 12 0.00355556     26   0.58044 0.70844 0.021064
## 13 0.00325926     29   0.56978 0.70489 0.021033
## 14 0.00266667     32   0.56000 0.70756 0.021057
## 15 0.00222222     42   0.53244 0.70844 0.021064
## 16 0.00207407     44   0.52800 0.71022 0.021080
## 17 0.00177778     51   0.51289 0.71733 0.021140
## 18 0.00160000     84   0.45156 0.71467 0.021118
## 19 0.00148148     99   0.42489 0.73156 0.021258
## 20 0.00133333    108   0.40978 0.73956 0.021323
## 21 0.00118519    146   0.35733 0.73778 0.021309
## 22 0.00111111    158   0.34311 0.73778 0.021309
## 23 0.00106667    162   0.33867 0.76000 0.021482
## 24 0.00088889    177   0.32000 0.76000 0.021482
## 25 0.00066667    277   0.22933 0.78044 0.021633
## 26 0.00059259    285   0.22400 0.79467 0.021733
## 27 0.00044444    326   0.19822 0.80178 0.021781
## 28 0.00029630    350   0.18756 0.81244 0.021852
## 29 0.00001000    368   0.18222 0.81156 0.021846
```

The 13th tree has the lowest cross-validation error (xerror): 0.70489. So we use that tree as our final model.

```
# prune by lowest cross-validation procedure
pruned_ct <- prune(cv_ct,
                   cp = cv_ct$cptable[which.min(cv_ct$cptable[,"xerror"]),"CP"])
length(pruned_ct$frame$var[pruned_ct$frame$var == "<leaf>"])
```

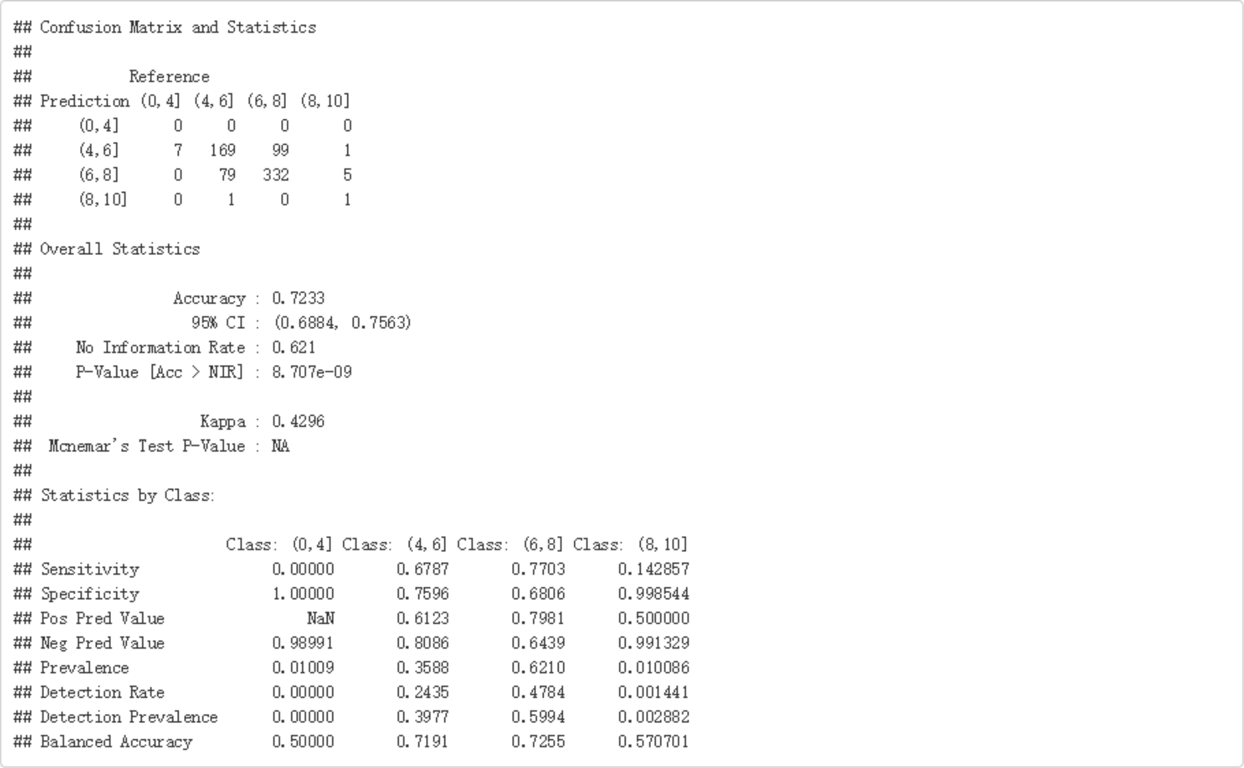
```
## [1] 30
```

```r
prp(pruned_ct, type = 1, extra = 1, split.font = 1, varlen = -10)
```



Apply our model to testing set:

```r
# apply model on test set
tree_pred_test <- predict(pruned_ct, testing, type = "class")
# generate confusion matrix for test data
confusionMatrix(tree_pred_test, testing$binned_score)
```



```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction (0,4] (4,6] (6,8] (8,10]
##     (0,4]      0     0     0      0
##     (4,6]      7   169    99      1
##     (6,8]      0    79   332      5
##     (8,10]     0     1     0      1
##
## Overall Statistics
##
##                Accuracy : 0.7233
##                  95% CI : (0.6884, 0.7563)
##     No Information Rate : 0.621
##     P-Value [Acc > NIR] : 8.707e-09
##
##                   Kappa : 0.4296
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: (0,4] Class: (4,6] Class: (6,8] Class: (8,10]
## Sensitivity               0.00000       0.6787       0.7703      0.142857
## Specificity               1.00000       0.7596       0.6806      0.998544
## Pos Pred Value                NaN       0.6123       0.7981      0.500000
## Neg Pred Value            0.98991       0.8086       0.6439      0.991329
## Prevalence                0.01009       0.3588       0.6210      0.010086
## Detection Rate            0.00000       0.2435       0.4784      0.001441
## Detection Prevalence      0.00000       0.3977       0.5994      0.002882
## Balanced Accuracy         0.50000       0.7191       0.7255      0.570701
```

The accuracy for our classification tree is 0.7233. It is a not bad accuracy.

5. Conclusion

High-quality films are mainly distributed in European and north American countries. Most popular movie genres are related with "Fantasy" and "Comedy" in the last 20 years. "Woman director", "independent film", and "credits stinger" are three of the most important keywords which have the highest frequency. Chris Buck, Lee Unkrich, and Joss Whedon, are the top three for having the most high-revenue movies, and Tim McCanlies, Floyd Mutrux and Frank Darabont are the top three high-quality movie directors. There is no obvious relationship between the movie's score and its budget, and holds regardless of whether the producer's studio is famous or not. Runtime and the number of people who vote the film will influence the score a lot.

That is to say, for the people who want to shoot a high quality movie, you can learn to produce a film from Europe or north American countries. Or learning from other high quality movie directors like Chris Buck, Lee Unkrich, and Joss Whedon. That will be great if your movie genre is related to fantasy or comedy. You do not need to worry too much about the movie-production companies and the budget since the quality of your movie will be not influenced much by these two factors, and independent films are actually popular. Do not forget to add some after credit scenes if it is possible. After your movie's publish, encourage people to vote for your movie might help you to get a better score online.

6. Future work

This project is based on analysing the relationship between high-quality movies and other factors, so we put “vote_average” in the center place. But during the research and analysing, we noticed many other interesting topics that may worthy to go deeper. Thus in this section, we list two points that is most worthy for the future work.

In section 4.1, we noticed that the high-quality movies are concentrated in Europe and the United States. But we cannot make sure if that is caused by some biases in metadata since TMDB is user edited. More work should do to explore the vote scores’ biases, like bias between English user group and non-English user group,or bias between English user American group and English user non-American group.

In section 4.2. We learn from the line chart about the movie genres’ trend. In general, after skyrocketing all types of movies in 1996, there is a decreasing trend after 2006. Two assumptions are trying to explain the phenomenon. One, the film industry has a close relationship with social economics. If we look carefully at the years that the amount of all types of movies rapidly decreasing, it coincided with the onset of financial crises. The rapid increasing is mostly caused by global economic prosperity, and the fluctuating decreasing trend is caused by the wake of the financial crisis in 2007. Two, the growth and reduce of total amount of movies is a combination result of economics and technology. The fast growth around 1996 may mostly caused by economics, but the decrease after 2006 may mostly caused by the impact of new technology, such as internet and smartphone. Both economics and technology effect on film industry, but the weights for economics and technology in each year is different. That will be great if we can find the factors that affecting the amount of films.

7. Bibliography

(2018). “Kaggle.” URL. https://www.kaggle.com/tmdb/tmdb-movie-metadata/home.

(2018). “The movie database.” URL. https://www.themoviedb.org/.

Goodfellow I, Bengio Y, Courville A (2016). *Deep Learning (Adaptive Computation and Machine Learning series)*. The MIT Press.

Hyndman, J R, Fan Y (1996). “Sample Quantiles in Statistical Packages.” *The American Statistician*, **50**(4), 361–365.

Karniouchina, V E (2011). “Impact of star and movie buzz on motion picture distribution and box office revenue.” *International Journal of Research*, **28**(1), 62–74.

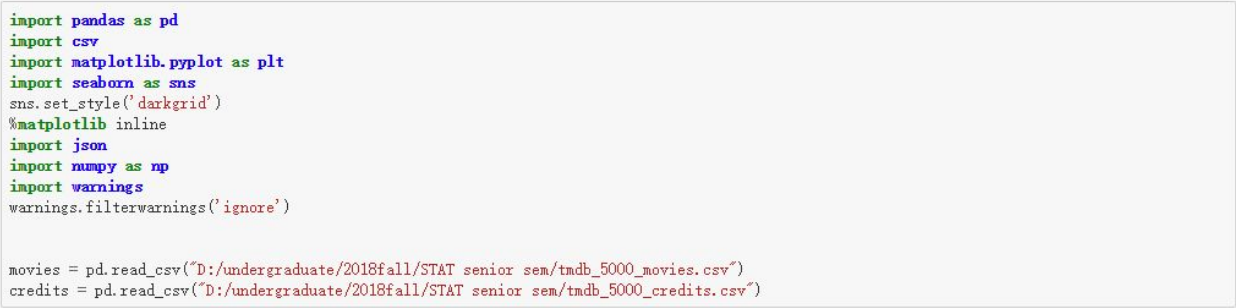
Stanton J (2012). *Introduction to Data Science*. 3th edition.

Wackerly D, Mendenhall W, Scheaffer RL (2008). *Mathematical Statistics with Applications*. Thomson Brooks, 7th edition.

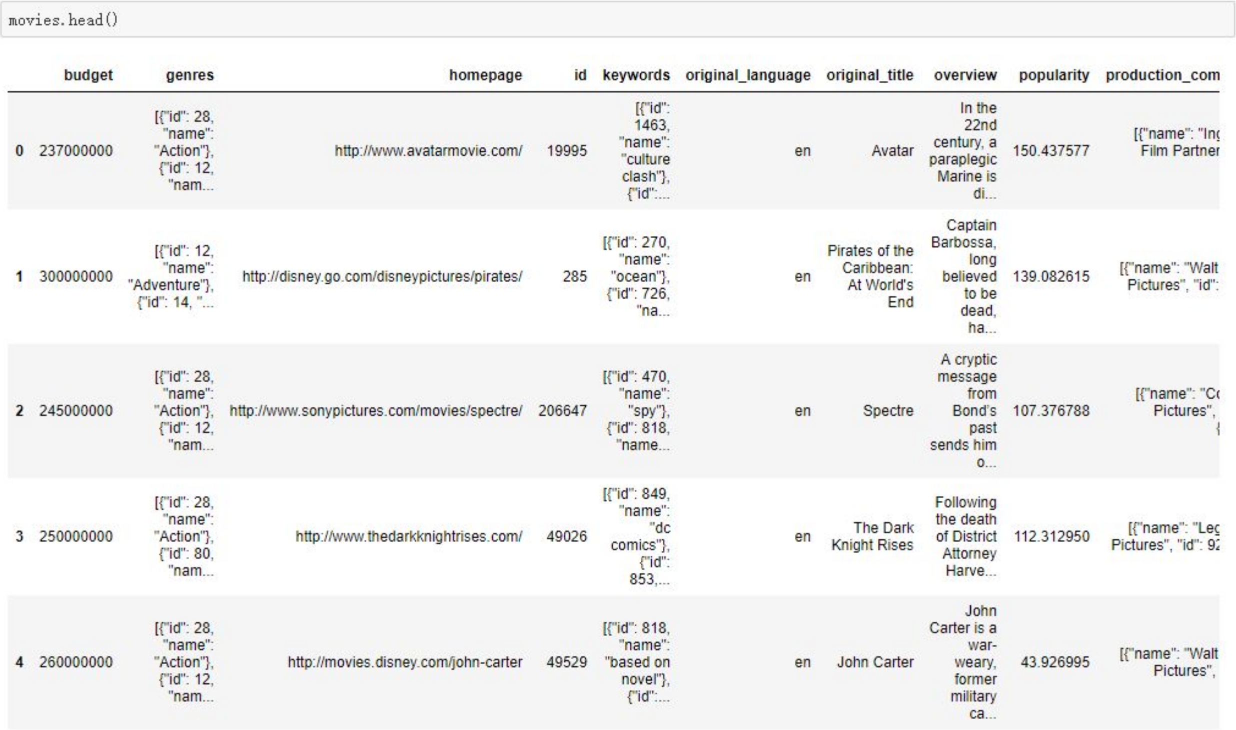
Wasserman M, Mukherjee S, Scott K, Zeng XHT, Radicchi F, Amaral LA (2015). “Correlations between user voting data, budget, and box office for films in the Internet Movie Database.” *Journal of the Association for Information Science and Technology*, **66**(4), 858–868.

Attached File 1

TMDB tans try.ipynb

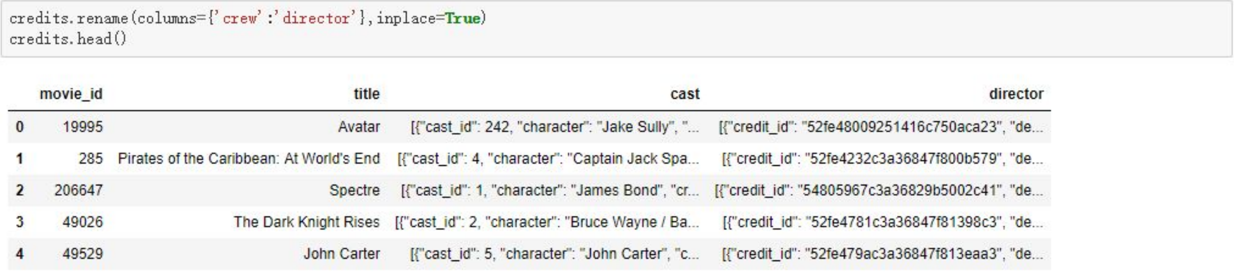


```
import pandas as pd
import csv
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style('darkgrid')
%matplotlib inline
import json
import numpy as np
import warnings
warnings.filterwarnings('ignore')


movies = pd.read_csv("D:/undergraduate/2018fall/STAT senior sem/tmdb_5000_movies.csv")
credits = pd.read_csv("D:/undergraduate/2018fall/STAT senior sem/tmdb_5000_credits.csv")
```



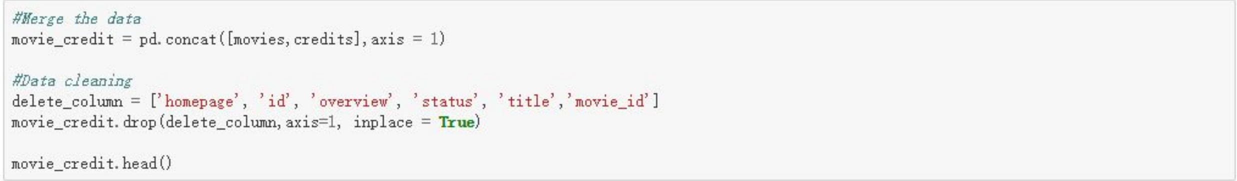
```
movies.head()
```

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	production_com
0	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.avatarmovie.com/	19995	[{"id": 1463, "name": "culture clash"}, {"id":...	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.437577	[{"name": "Ing Film Partner
1	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	http://disney.go.com/disneypictures/pirates/	285	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139.082615	[{"name": "Walt Pictures", "id":
2	245000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.sonypictures.com/movies/spectre/	206647	[{"id": 470, "name": "spy"}, {"id": 818, "name...	en	Spectre	A cryptic message from Bond's past sends him o...	107.376788	[{"name": "Co Pictures", {
3	250000000	[{"id": 28, "name": "Action"}, {"id": 80, "nam...	http://www.thedarkknightrises.com/	49026	[{"id": 849, "name": "dc comics"}, {"id": 853,...	en	The Dark Knight Rises	Following the death of District Attorney Harve...	112.312950	[{"name": "Leg Pictures", "id": 92
4	260000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://movies.disney.com/john-carter	49529	[{"id": 818, "name": "based on novel"}, {"id":...	en	John Carter	John Carter is a war-weary, former military ca...	43.926995	[{"name": "Walt Pictures",

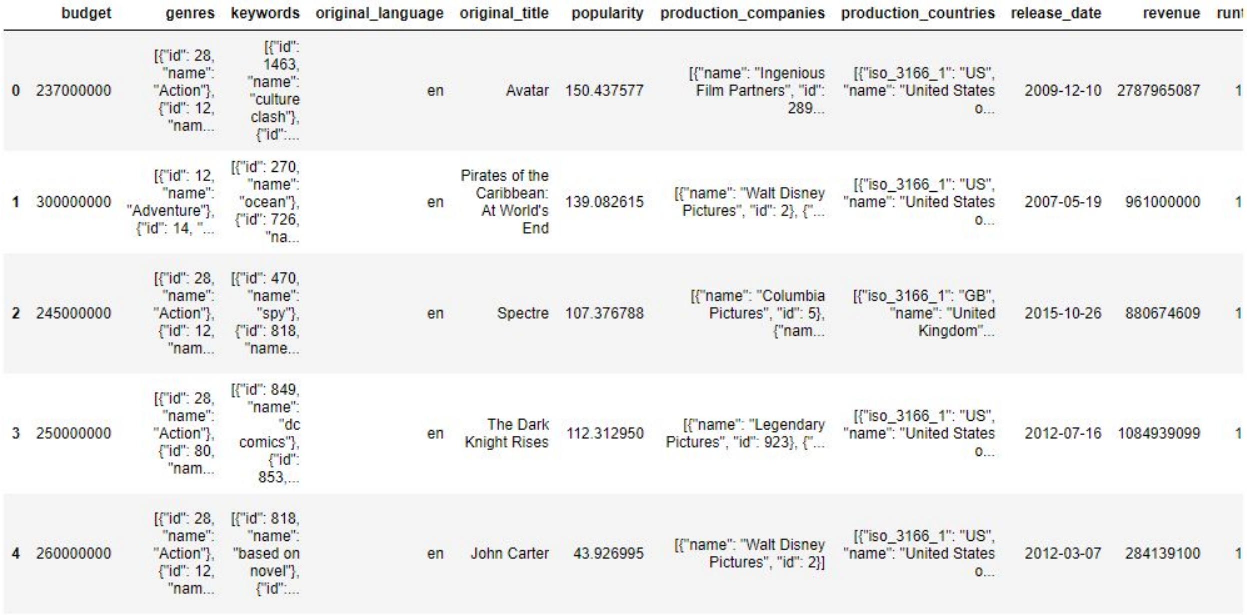


```
credits.rename(columns={'crew':'director'},inplace=True)
credits.head()
```

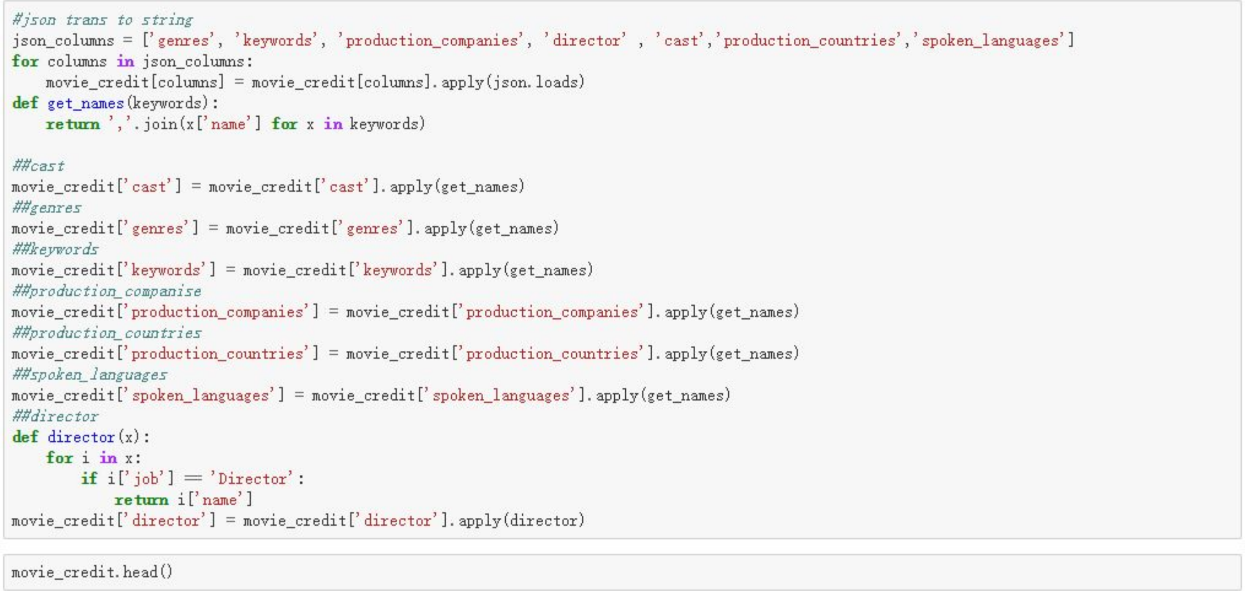
	movie_id	title	cast	director
0	19995	Avatar	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id": "52fe48009251416c750aca23", "de...
1	285	Pirates of the Caribbean: At World's End	[{"cast_id": 4, "character": "Captain Jack Spa...	[{"credit_id": "52fe4232c3a36847f800b579", "de...
2	206647	Spectre	[{"cast_id": 1, "character": "James Bond", "cr...	[{"credit_id": "54805967c3a36829b5002c41", "de...
3	49026	The Dark Knight Rises	[{"cast_id": 2, "character": "Bruce Wayne / Ba...	[{"credit_id": "52fe4781c3a36847f81398c3", "de...
4	49529	John Carter	[{"cast_id": 5, "character": "John Carter", "c...	[{"credit_id": "52fe479ac3a36847f813eaa3", "de...



```
#Merge the data
movie_credit = pd.concat([movies,credits],axis = 1)

#Data cleaning
delete_column = ['homepage', 'id', 'overview', 'status', 'title','movie_id']
movie_credit.drop(delete_column,axis=1, inplace = True)

movie_credit.head()
```

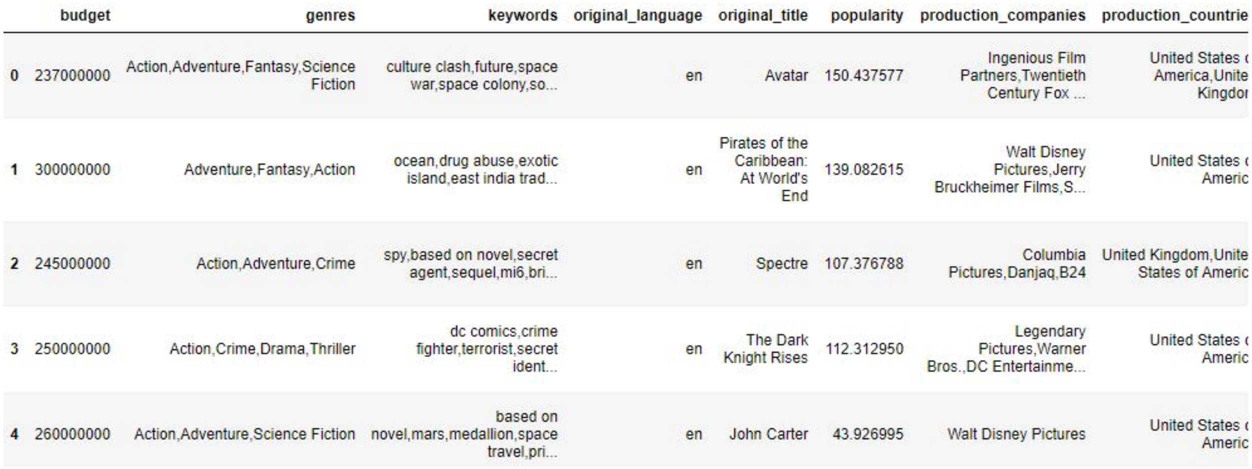


	budget	genres	keywords	original_language	original_title	popularity	production_companies	production_countries	release_date	revenue	run
0	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	[{"id": 1463, "name": "culture clash"}, {"id":...	en	Avatar	150.437577	[{"name": "Ingenious Film Partners", "id": 289...	[{"iso_3166_1": "US", "name": "United States o...	2009-12-10	2787965087	1
1	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	en	Pirates of the Caribbean: At World's End	139.082615	[{"name": "Walt Disney Pictures", "id": 2}, {"...	[{"iso_3166_1": "US", "name": "United States o...	2007-05-19	961000000	1
2	245000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	[{"id": 470, "name": "spy"}, {"id": 818, "name...	en	Spectre	107.376788	[{"name": "Columbia Pictures", "id": 5}, {"nam...	[{"iso_3166_1": "GB", "name": "United Kingdom"...	2015-10-26	880674609	1
3	250000000	[{"id": 28, "name": "Action"}, {"id": 80, "nam...	[{"id": 849, "name": "dc comics"}, {"id": 853,...	en	The Dark Knight Rises	112.312950	[{"name": "Legendary Pictures", "id": 923}, {"...	[{"iso_3166_1": "US", "name": "United States o...	2012-07-16	1084939099	1
4	260000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	[{"id": 818, "name": "based on novel"}, {"id":...	en	John Carter	43.926995	[{"name": "Walt Disney Pictures", "id": 2}]	[{"iso_3166_1": "US", "name": "United States o...	2012-03-07	284139100	1



```
#json trans to string
json_columns = ['genres', 'keywords', 'production_companies', 'director' , 'cast','production_countries','spoken_languages']
for columns in json_columns:
    movie_credit[columns] = movie_credit[columns].apply(json.loads)
def get_names(keywords):
    return ','.join(x['name'] for x in keywords)

##cast
movie_credit['cast'] = movie_credit['cast'].apply(get_names)
##genres
movie_credit['genres'] = movie_credit['genres'].apply(get_names)
##keywords
movie_credit['keywords'] = movie_credit['keywords'].apply(get_names)
##production_companise
movie_credit['production_companies'] = movie_credit['production_companies'].apply(get_names)
##production_countries
movie_credit['production_countries'] = movie_credit['production_countries'].apply(get_names)
##spoken_languages
movie_credit['spoken_languages'] = movie_credit['spoken_languages'].apply(get_names)
##director
def director(x):
    for i in x:
        if i['job'] == 'Director':
            return i['name']
movie_credit['director'] = movie_credit['director'].apply(director)
```

```
movie_credit.head()
```

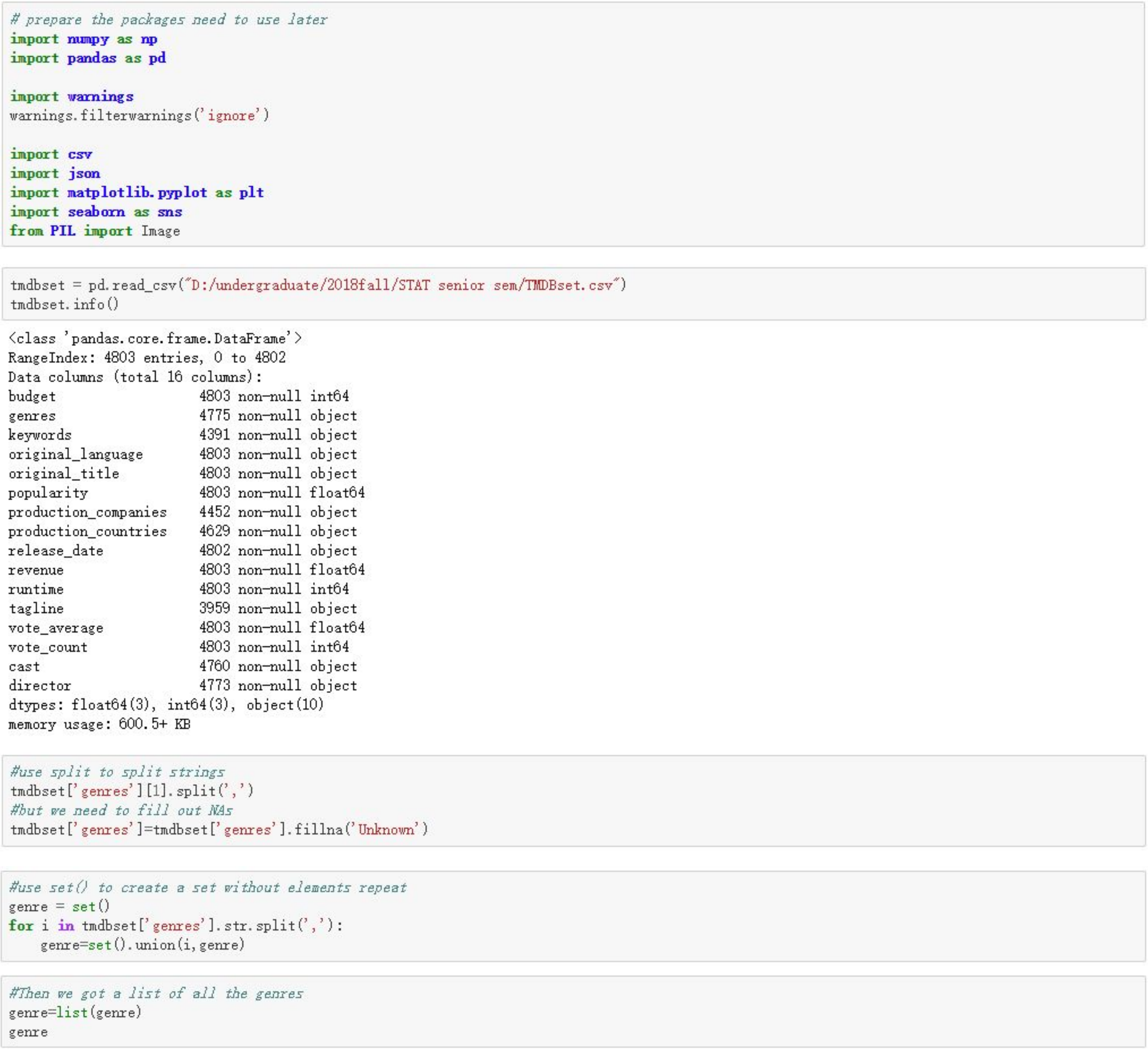


	budget	genres	keywords	original_language	original_title	popularity	production_companies	production_countrie
0	237000000	Action,Adventure,Fantasy,Science Fiction	culture clash,future,space war,space colony,so...	en	Avatar	150.437577	Ingenious Film Partners,Twentieth Century Fox ...	United States America,Unite Kingdo
1	300000000	Adventure,Fantasy,Action	ocean,drug abuse,exotic island,east india trad...	en	Pirates of the Caribbean: At World's End	139.082615	Walt Disney Pictures,Jerry Bruckheimer Films,S...	United States Americ
2	245000000	Action,Adventure,Crime	spy,based on novel,secret agent,sequel,mi6,bri...	en	Spectre	107.376788	Columbia Pictures,Danjaq,B24	United Kingdom,Unite States of Americ
3	250000000	Action,Crime,Drama,Thriller	dc comics,crime fighter,terrorist,secret ident...	en	The Dark Knight Rises	112.312950	Legendary Pictures,Warner Bros.,DC Entertainme...	United States Americ
4	260000000	Action,Adventure,Science Fiction	based on novel,mars,medallion,space travel,pri...	en	John Carter	43.926995	Walt Disney Pictures	United States Americ

```
movie_credit.to_csv("D:/undergraduate/2018fall/STAT senior sem/movie_data.csv",index=False)
```

Attached File 2

Movie types analyzing.ipynb



```
# prepare the packages need to use later
import numpy as np
import pandas as pd

import warnings
warnings.filterwarnings('ignore')

import csv
import json
import matplotlib.pyplot as plt
import seaborn as sns
from PIL import Image
```

```
tmdbset = pd.read_csv("D:/undergraduate/2018fall/STAT senior sem/TMDBset.csv")
tmdbset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4803 entries, 0 to 4802
Data columns (total 16 columns):
budget                  4803 non-null int64
genres                  4775 non-null object
keywords                4391 non-null object
original_language       4803 non-null object
original_title          4803 non-null object
popularity              4803 non-null float64
production_companies    4452 non-null object
production_countries    4629 non-null object
release_date            4802 non-null object
revenue                 4803 non-null float64
runtime                 4803 non-null int64
tagline                 3959 non-null object
vote_average            4803 non-null float64
vote_count              4803 non-null int64
cast                    4760 non-null object
director                4773 non-null object
dtypes: float64(3), int64(3), object(10)
memory usage: 600.5+ KB
```

```
#use split to split strings
tmdbset['genres'][1].split(',')
#but we need to fill out NAs
tmdbset['genres']=tmdbset['genres'].fillna('Unknown')
```

```
#use set() to create a set without elements repeat
genre = set()
for i in tmdbset['genres'].str.split(','):
    genre=set().union(i,genre)
```

```
#Then we got a list of all the genres
genre=list(genre)
genre
```

```
['Fantasy',
 'TV Movie',
 'Science Fiction',
 'War',
 'Comedy',
 'Music',
 'Animation',
 'Action',
 'Thriller',
 'Family',
 'Drama',
 'Western',
 'Mystery',
 'History',
 'Horror',
 'Documentary',
 'Unknown',
 'Adventure',
 'Romance',
 'Crime',
 'Foreign']
```

```python
#Transfer the date formate
tmdbset['release_year']=pd.to_datetime(tmdbset.release_date,format='%Y-%m-%d').dt.year
```

```python
# for loop: for each line, if contain the specific genre,return 1, otherwise return 0
for genr in genre:
    tmdbset[genr] = tmdbset['genres'].str.contains(genr).apply(lambda x:1 if x else 0)
```

```python
#create a table for genre and year
tmdbset_genre_year = tmdbset.loc[:, genre]
tmdbset_genre_year.index = tmdbset['release_year']
tmdbset_genre_year.head()
```

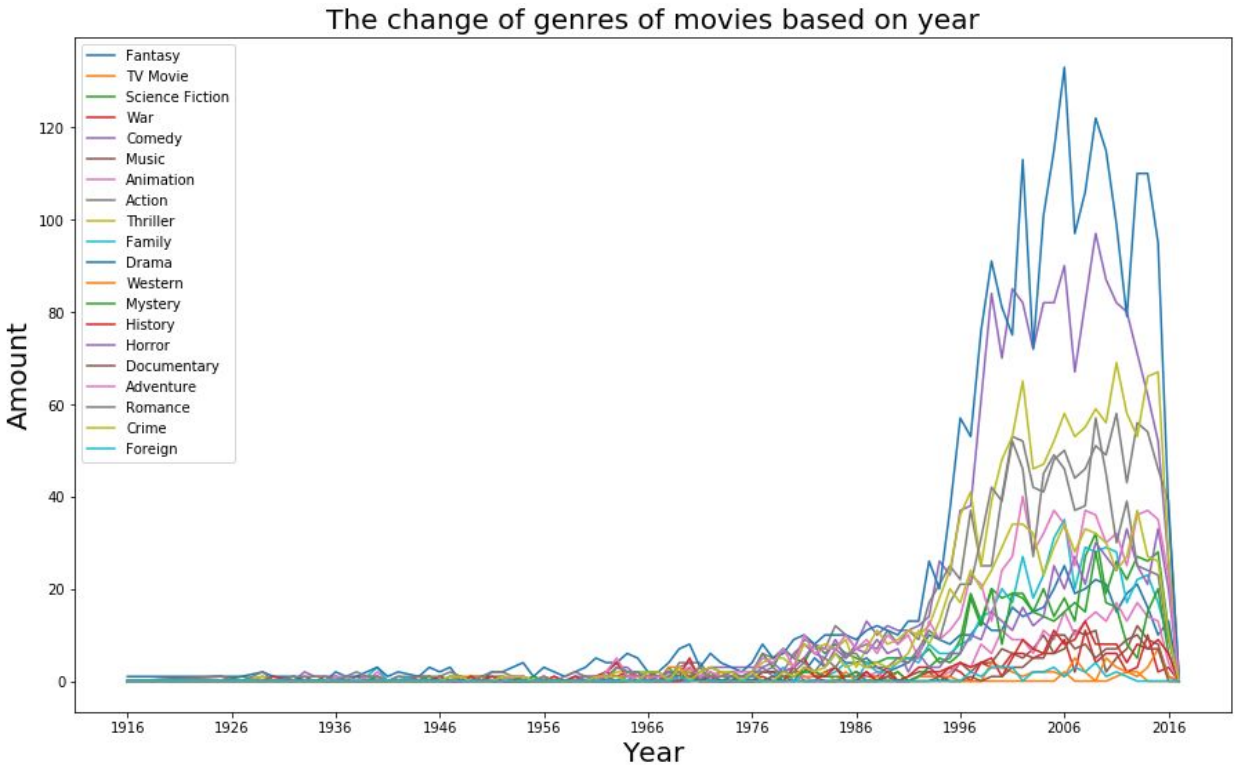
	Fantasy	TV Movie	Science Fiction	War	Comedy	Music	Animation	Action	Thriller	Family	...	Western	Mystery	History	Horror	Documentary	Unkno
release_year																	
2009.0	1	0	1	0	0	0	0	1	0	0	...	0	0	0	0	0	
2007.0	1	0	0	0	0	0	0	1	0	0	...	0	0	0	0	0	
2015.0	0	0	0	0	0	0	0	1	0	0	...	0	0	0	0	0	
2012.0	0	0	0	0	0	0	0	1	1	0	...	0	0	0	0	0	
2012.0	0	0	1	0	0	0	0	1	0	0	...	0	0	0	0	0	

5 rows × 21 columns

```python
#count the amount of each genre for each year
tmdbset_genre_year_count = tmdbset_genre_year.groupby('release_year').sum()
tmdbset_genre_year_count = tmdbset_genre_year_count.drop(columns=['Unknown'])
tmdbset_genre_year_count.head()
```

	Fantasy	TV Movie	Science Fiction	War	Comedy	Music	Animation	Action	Thriller	Family	Drama	Western	Mystery	History	Horror	Documentary	A
release_year																	
1916.0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	
1925.0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	
1927.0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	
1929.0	0	0	0	0	0	1	0	0	1	0	2	0	0	0	0	0	
1930.0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	0	

```python
# get the plot!
plt.figure(figsize=(15,9))
plt.plot(tmdbset_genre_year_count,label=tmdbset_genre_year_count.columns)
plt.legend(tmdbset_genre_year_count)
plt.xticks(range(1916, 2017,10))
plt.xlabel('Year',fontsize=20)
plt.ylabel('Amount',fontsize=20)
plt.title('The change of genres of movies based on year',fontsize=20)
plt.show()
```


The change of genres of movies based on year
Fantasy
TV Movie
Science Fiction
War
Comedy
Music
Animation
Action
Thriller
Family
Drama
Western
Mystery
History
Horror
Documentary
Adventure
Romance
Crime
Foreign
Amount
0
20
40
60
80
100
120
Year
1916
1926
1936
1946
1956
1966
1976
1986
1996
2006
2016