Capstone Report

Color-tone and ranking relationships of Chinese Film Industry, 2017

Tian Wang

12/4/2018

Introduction

1.1 Introduction

The idea of this project is to explore the relationship between film color-tone and its ranking. Films are cultural artifacts created by specific cultures. In this project, the topic concentrated on Chinese film industry in year of 2017. Film is a source of entertainment, but it is more a form art. Art can affect people in a psychological way. Just as show below, film affect people not only because of the story it told. But also, how the post production processed. If ranks do have a strong relationship with color-tone, then the productors of films could modulate the color tone of films into the best attractive tone. Then eventually, the box-office will be affected.

1.2 Problems and limitations

First of all, there were no ready-to-use datasets. And I had to face the threat to be blocked when crawling the data.

At the very beginning, the topic was to explore the relationship between color-tone and its box-office. But, after crawled the data, the problem arose. The box-office had a great discrepancy between the maximum and the minimum. Therefore, it would be hard to build levels based on box-office.

The amount of raw data was huge. With a huge amount, it was hard to control the qualities that were needed in the project. And there were even more missing values that had to be dropped.

Although the raw data had a significant amount. The final data were only 203 merged pictures. This leaded to the limitation to build models. At least 1000 films should be collected.

Process

2.1 Build clawer programs.

The main purpose of this step is to generate the data from website due to there is no any resource can be downloaded and used straightly.

When building the clawers, it was hard to generate all the data from websites once. Therefore, the crawler I built generated data from several pages. I found data from different website to find the best one that could give me all I need with a unity format. While going over the web pages and extracting data, even I pretended to be not a crawler, I still be blocked after extracting hundreds of data. Technically, the urllib package with beautifulsoap package worked perfectly together. A 'time.sleep()' was the way helped me out of being blocked by the website. With a detailed website, it is good to find everything in one website, more attention should be paid to peel the 'class' and seize the embedded information needed in the webpage body. And in this step, I had a sense about how to traverse in a more proper and efficient way and how to grab only the useful information from website pages.

2.2 Download

After crawled data, I used a month to download all the films by its torrent. But, in vain. Torrents are not that trustful. And the VPN limited the speed to download as well as the number of resources. The way out was to crawl prevue of films. It's easy to find the '.mp4' data. And could be downloaded legally and efficiently by python. To implement the download process is to use the urllib package. To request the materials given in the link with '.mp4' suffix. All the prevue were downloaded in one folder with name start at 0. The order followed the films' titles. To generate all the data in the same order make things easier and more efficiently.

2.3 Create empty folders

In this step, the main purpose was to classify the prevue by the film list. The folders' names followed the index of the films.

2.4 Copy and move

I generated the prevue size of each film. So in this step, there will be another traversal program built. If the index equals to the folder's name, then copy the number of the size of the prevue of the film into the folder. Shutil package worked good in this step for the copy-paste processes.

2.5 Cut prevue

In this step, prevue were cut into pictures and saved. But, when the number of videos reached 700 and in 200 more folders, things went crazy. Another traverse function built here to make things more efficient and convenient. But, there were eventually more than 100000 screenshots. All the pictures stored twice. One was in the folder they should be, the other was a duplicate backup in the root folder.

In the function I built to cut the prevue, I used every two seconds as interval. Which eventually lead to some chaos in some folders. And had to be fixed in process 1.8.

2.6 Rename all the pictures

In this step, I built a function to rename all the pictures in all the folders. The names of pictures in each folder began from 0. With this process, it became easy to traverse all the data in the following steps.

2.7 Crop and filters

The screenshots had black edges, almost black pieces and almost white pieces(As showed below). First, I had a function to cut all the black edges of all the screenshots. Then some of the darker pieces were cut into some other shapes that were not good for the merge step. So, I manually took an example in each folder to initialize and store the shape and size to cut. And changed the conditions of the remove black picture function to save those horror movies with almost every scene black(As showed bottom right corner).

(Before processing)

(After processing)

2.8 Merge

The trouble that being caused in step 1.5 showed up. Some of the movies had only one prevue with even less then 1 min, and some of the movies had more than 10 prevues with even more than 5 mins each. This step had to be done manually. So, I had to go into each folder to see whether the pictures were qualified for merging purpose.

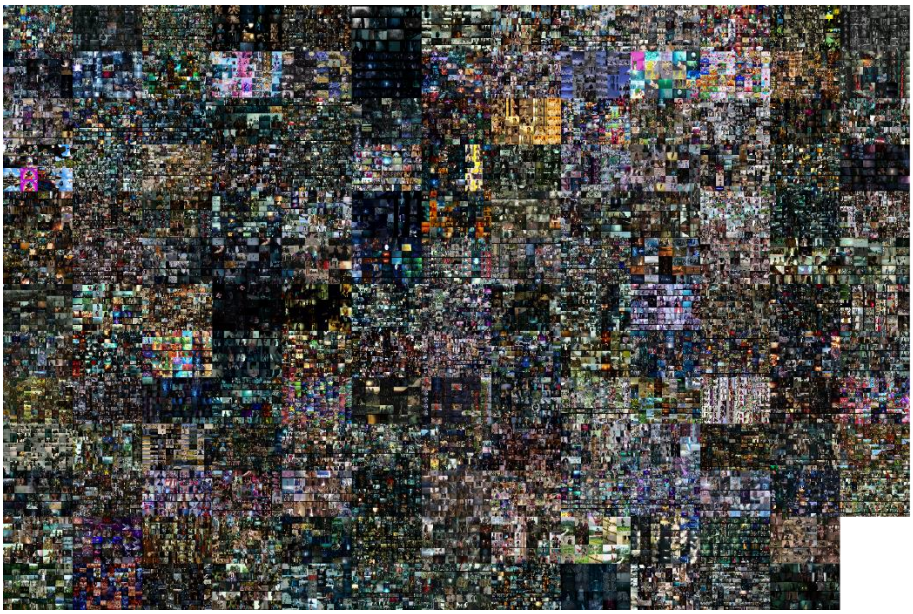
I wrote a function to cut the videos only in one folder. And redo the step 1.7 to save the qualified pictures.

The merging method had a condition that the total number of pictured being merged should not exceed 150 pictures to form a new picture. But the fundamental reason was the new picture with more than 150 pictures exceed the maximum computation space. Because I didn't resize the pictures before merging them.

2.9 Resize

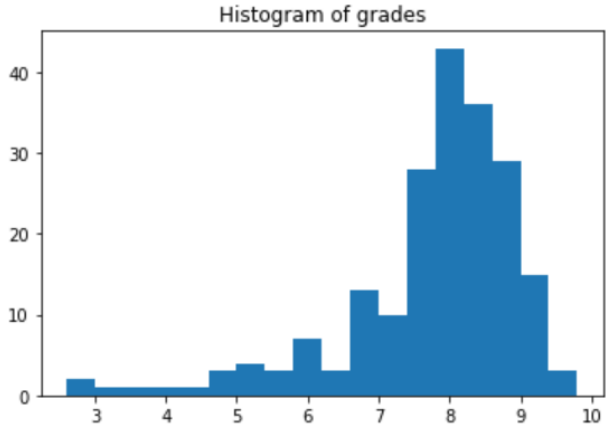
The original merged pictures are in random sizes and random shapes. Most of the pictures exceed 30mbs which were too big to use. I reshaped all the pictures into 600*400, the size were only around hundreds of kbs.

So that I merged all the reshaped and resized merged photos into one. To see the color changes.(As showed below) It obvious that, different films had different color-tones used.

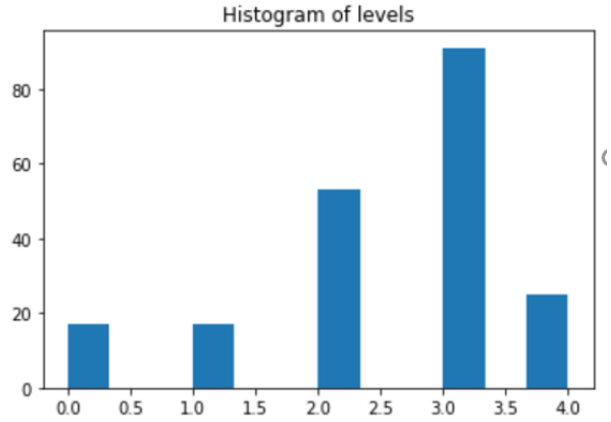


2.10 Build KNN model

Before built the model, I built a histogram based on rankings.



The ranking data are skewed. And the ranking unit was 0.1. It was too much detailed for model building. Then, I split the rankings into levels by conditions. And redo the histogram.

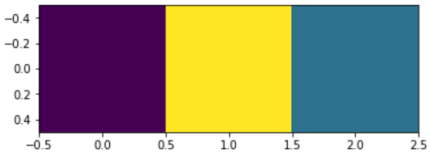


The left tail became better. But I didn't split the interval 8~9 into two levels. The amount of dataset was too limited. It would be not wise to split the small dataset into a lot of groups.

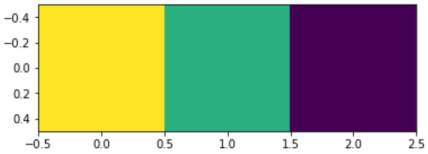
The first KNN model was built by 2D data. The build in function didn't support 3D data. After converting the 3D data into 2D data, I got the percentage of accuracies showed in the table. Then, because I want to explore the color-tone of the whole movie. So test all the details of color in the model were not necessary. So I convert the data into average color of the whole picture. And got higher accuracies. And also I printed the average color of each picture to see the visual results.

If a film director or a film productor could know the tendency of the color-tone used in the film, they can tone the film into the best fit color-tone, then they could have a higher ranking of the film released. But in this step, I had another idea. If the color-tone could be compared with genre, the results of the model will be more meaningful. I did visualizations on genre, there are more than 10 kinds of different genres. If to generate a meaningful data, more films should be trained.

2D data		Average data	
Brute	0.33497536945812806	brute	0.5517241379310345
kd_tree	0.4433497536945813	kd_tree	0.5566502463054187
ball_tree	0.46798029556650245	ball_tree	0.5467980295566502



(color of highest ranking)

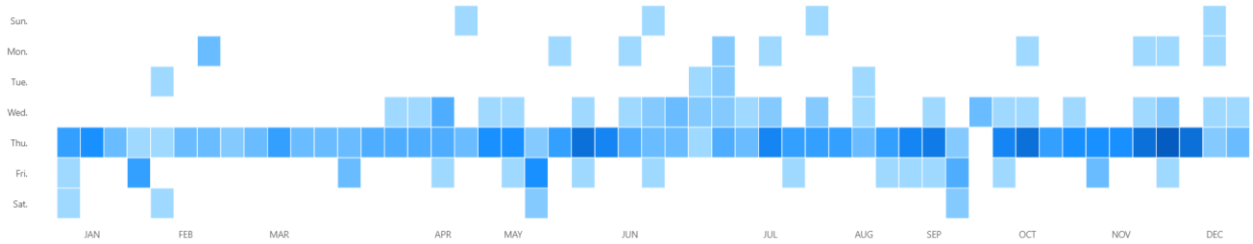


(color of lowest ranking)

2.11 Data visualizations

I used react.js & bizchart for this step. The work had to be done was to analyze the data structure of the bizchart and do the data processing. Use python to process the data into the correct format that the package using. And output into a '.json' file that could be used in 'react.js'.

2.11.1 Release number by day

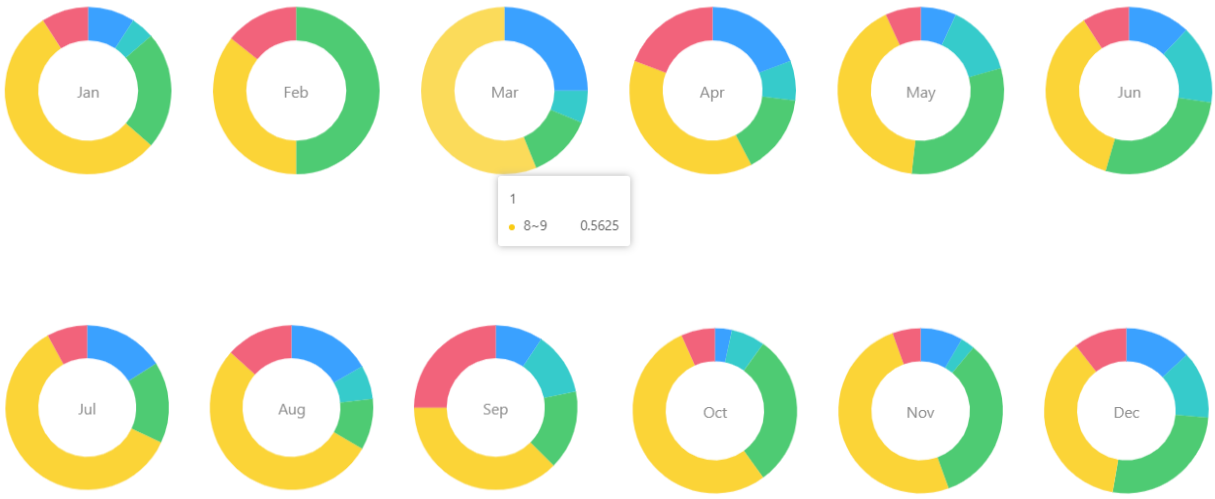


As showed in this graph, most of the films in 2017 launched on Thursday. The darker the color the more released films on that day.

2.11.2 Box-office by day

As showed in the graph, each month had some dark houses. Most of the films had the box-office around ￥500,000,000(around $71,430,000).

2.11.3 Ranking by month(rankings modified in levels)

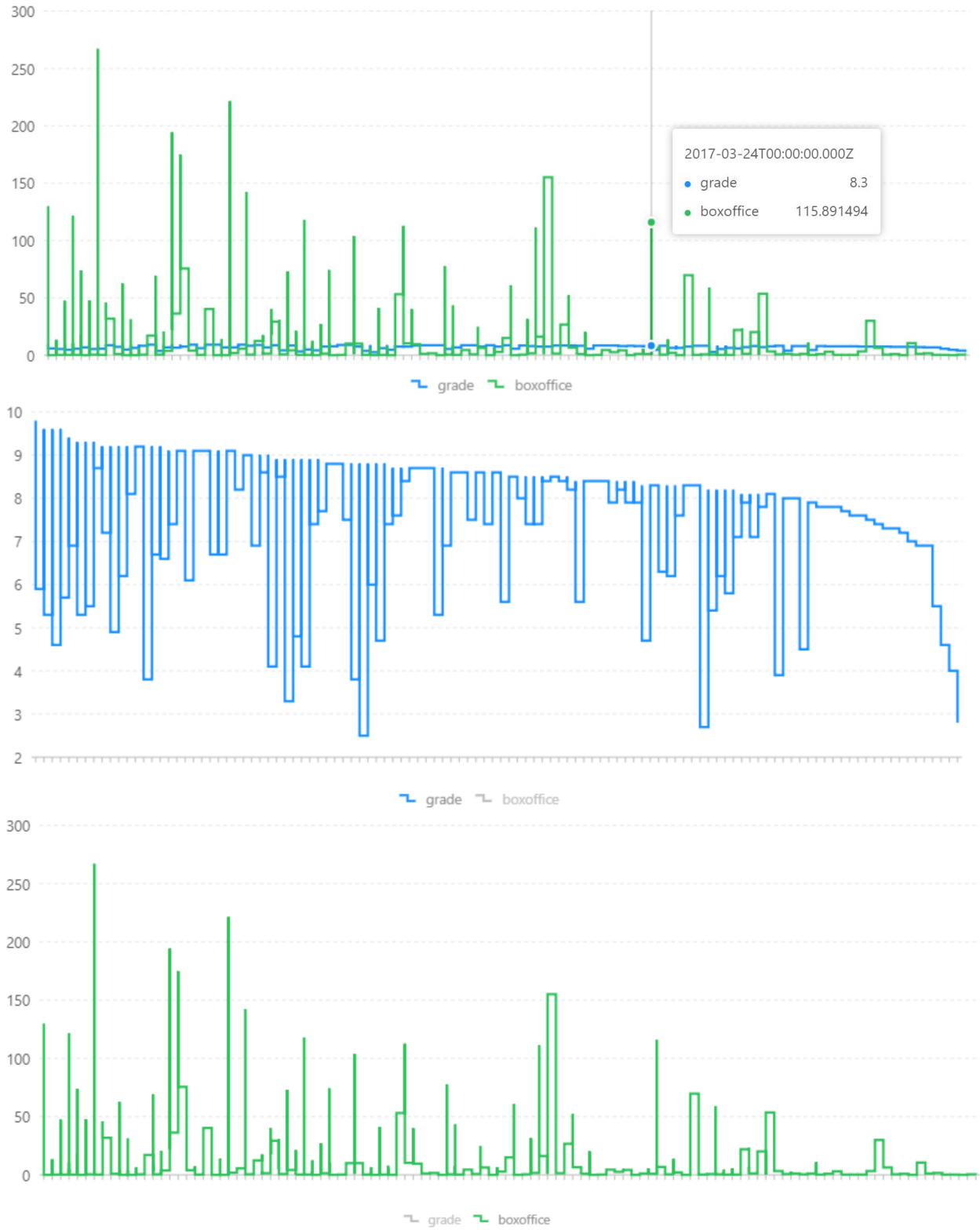


The visualization of this graph matched the histogram in 1.10. Most of the film got their rankings in 8~9. The funny thing is all the films in Feb had a ranking exceeded score 7. Spring Festival was in that month, it is called Lunar New Year Movie Season, similar to the Christmas Movie Season in the US, maybe the entertainment companies had to think twice before release that time. Big companies with best movie would try their best to earn more money. Small companies might not even have a chance to scheduled their film in the cinema.

2.11.4 Word cloud of genre

The most released during 2017 was comedy movie. Everyone wants to be happy. So comedy movies always have a big share of the market. Action movie and affectional movie are also always the red hot topics.

2.11.5 Box-office & ranking



Based on the graphs, there should be relationships between grade(ranking) and box-office. The higher ranking the films has, leads to a better box-office.