Big Data:
Design of a new interactive data analysis tool

Máster Universitario en Ingeniería Informática
Universidad Politécnica de Madrid

Group 22:
José Domínguez Pérez
Ismael Muñoz Aztout
Jonatan Ruedas Mora

January 2021

Contents

1 Introduction

We have selected a dataset based on films due the fact that we really love films and we think that the topic could be quite interesting, specially if we are able to show the evolution of the film industry in the last decades.

We have chosen The Movie Database dataset in order to carry out our practical work.

1.1 Dataset description

Our dataset consists of table with items and attributes. More specifically, it is a Flat Table with an explicit index (ID) of categorical type.

Variable	**Type**	**Description**
Budget	Integer	Budget of the movie
Genres	JSON	Genres of the movie
Keywords	JSON	Keywords related to the movie
Production_companies	JSON	Companies that produced the movie
Revenue	Integer	Revenue of the movie
Runtime	Integer	Duration of the movie in minutes
Vote_average	Integer	Rating of the movie given by the users
Tittle	String	Name of the movie
Release_date	Date	Release date of the movie
Production_countries	JSON	Countries that have produced the movie
Homepage	String	Webpage of the movie
Id	Integer	Movie identifier
Original_language	String	Original language of the movie
Original_title	String	Original title of the movie
Overview	String	Description of the movie
Popularity	Integer	Popularity of the movie
Spoken_languages	JSON	Different spoken languages in the movie
Status	String	Status of the movie
Tagline	String	Brief description of the movie
Vote_count	Integer	Number of votes that the movie has received

The columns that we use are:

Variable	**Type**	**Description**
Budget	Integer	Budget of the movie
Genres	JSON	Genres of the movie
Keywords	JSON	Keywords related to the movie

Production_companies	JSON	Companies that produced the movie
Revenue	Integer	Revenue of the movie
Vote_average	Integer	Rating of the movie given by the users
Release_date	Date	Release date of the movie
Production_countries	JSON	Countries that have produced the movie

The columns that we have produced are:

Variable	**Type**	**Description**
Year	Integer	Year of the release
Month	Integer	Month of the release
Day	Integer	Day of the release
Decade	Integer	Decade of the release
Earnings	Integer	Earnings of a movie

We had to parse the columns that contain JSON values to String in order to have a better structure to be able to retrieve the required information.

2 Problem characterization

The domain in which we have worked on is the one related to the movie industry, so terms like budget, revenue, ratings are used everywhere. This domain does not have many fuzzy words, as a consequence, it has been quite easy to understand its vocabulary.

Apart from that, as we do not have access to experts in the topic, we have tried put ourselves into a movie analyst shoes, in order to think what kind of questions we would want to explain to the public to ilustrate how the industry has evolved over the last 50 years.

The questions that we have raised for this practical work are the following:

1. When is more profitable to release a film?
2. How the popularity of the companies has evolved over time?
3. Which are the most remarkable genres and keywords in the movie industry of a country?

3 Data and Task abstractions

3.1 First Question

3.1.1 Target

The target of this question the *Period of the year in which it is more profitable to release a film.* This target can be classified as an Atribute $\rightarrow$ One $\rightarrow$ Distribution.

3.1.2 Abstract Data

First of all, we had to think what kind of data do we need to answer the first question properly. We realized that we need to produce some data in order to know the *Decade*, *Month* and the *Earnings* of the different films.

We have decided to use the *Decade* instead of the *Year* because there were many years and as the film industry usually categorize films by decades we thought that it could be a good option.

The attributes that we use for this question are the following:

- **Consumed attributes**
  - *Budget*: Ordered $\rightarrow$ Quantitative $\rightarrow$ Sequential.
  - *Revenue*: Ordered $\rightarrow$ Quantitative $\rightarrow$ Sequential.
  - *Release_date*: Ordered $\rightarrow$ Ordinal $\rightarrow$ Secuential.
- **Produced attributes** (these attributes have been produced on the EDA)
  - *Year*: Ordered $\rightarrow$ Ordinal $\rightarrow$ Sequential.
  - *Month*: Ordered $\rightarrow$ Ordinal $\rightarrow$ Cyclic.
  - *Decade*: Ordered $\rightarrow$ Ordinal $\rightarrow$ Sequential.
  - *Earnings*: Ordered $\rightarrow$ Quantitative $\rightarrow$ Diverging.

3.1.3 Abstract Tasks

The abstract tasks that we have defined for this question are the following:

- **Produce** $\rightarrow$ **Derive**. We combine the variables Revenue and Budget to produce the Earnings.
- **Consume** $\rightarrow$ **Discover** $\rightarrow$ **Generate**. We are generating an hypothesis about when is more profitable to release a movie.
- **Search** $\rightarrow$ **Explore**. We do not know the period of the year in which it is better to release a film.

- **Query** → **Identify**. We are identifying the most profitable time to release a film.

3.2 Second Question

3.2.1 Target

We have defined the next target to this question: *Evolution of the popularity of the companies over the years.* The defined target can be classified as All data → Trend.

3.2.2 Abstract Data

In order to answer this question we realized that we will need the following data:

- **Consumed attributes**
  - *Vote_average*: Ordered → Quantitative → Sequential.
  - *Production_companies*: Categorical.
  - *Year*: Ordered → Ordinal → Sequential.

3.2.3 Abstract Tasks

The abstract tasks that we have defined for this question are the following:

- **Consume** → **Present** →. We want to present the evolution (trend) of the popularity of the companies over time. We already have this data so we do not need to generate more data..
- **Search** → **Explore**. We do not know the evolution of the popularity of the companies and we do not know where to search for it.
- **Query** → **Compare**. We want to compare the evolution of different companies

3.3 Third Question

3.3.1 Target

The target for this question is: *Most remarkable genres of films and keywords of a country.* It can be classified as All data → Outliers.

3.3.2 Abstract Data

To tackle this question we need this data:

- **Consumed attributes**

  - *Genres*: Categorical.
  - *Production_ countries*: Categorical.

- **Produced attributes**
  - *Frequency of movies in a genre by country*: Ordered → Quantitative → Sequential.
  - *Frequency of keywords by country*: Ordered → Quantitative → Sequential.

3.3.3 Abstract Tasks

We have selected the next abstract tasks for this question:

- **Consume** → **Enjoy**. We feel curiosity to know which are the most important genres and keywords by country.
- **Search** → **Lookup**. We know the number of movies in a genre and the number of keywords by country, and also we do know where to find them.
- **Query** → **Identify**. We want to identify the outliers.

4 Interaction and visual encoding

4.1 First Question

The methods that we have selected to interact with the visualization are the following:

- **Filter**: We filter by decades to reduce the size of the dataset in order to have a clear view of the data.

To encode the data we have used a Heatmap. We have selected this idiom because we have two ordered variables, *Year* and *Month*, that can act as categorical, and a quantitative variable, so we can use the categorical variables in the axis and let the quantitative for the cells of the heatmap.

The X-axis represents the years of the selected *Decade* while the Y-axis encodes the different months. Finally, the cells encode the *Earnings* of the movies that have been released in a specific month of a year.

We have used a green tone and we vary its saturation to obtain a green scale that represents the *Earnings*.

The figure 1 ilustrates the idiom that we have selected.

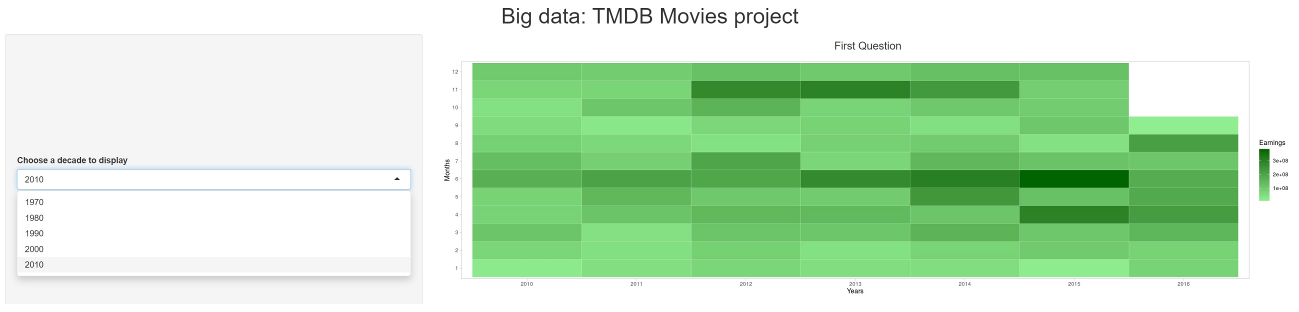


Figure 1: First question: 2010 decade

4.2 Second Question

We have selected the following methods to interact with the visualization:

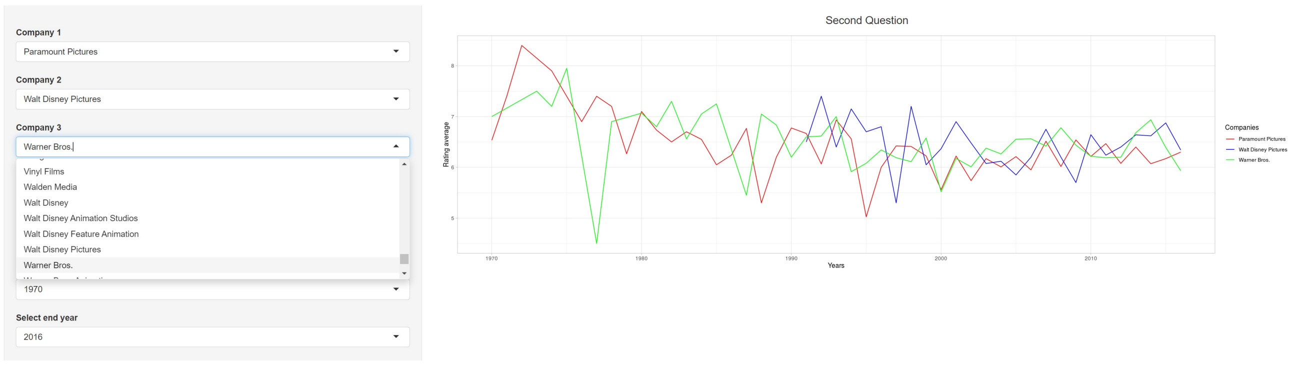
- **Filter**: We filter can filter by a group of years.
- **Facet $\rightarrow$ Superimpose**: We overlap the evolution of different companies in the same visualization.

In order to encode the data we have selected a Line chart because these plots are perfect to show changes over certain periods of time. In our case, these changes will be the variation of the average rating and we will be able to overlap different companies to compare them.

The X-axis represents the *years* that the user have selected, the Y-axis shows the *average rating* of a given company based on the ratings of their movies. Apart from that, every *country* has a different color to improve the visual perception.

We only allow to compare 5 companies at the same time because because there are too much data and it wont be understandable. In addition, we do not allow to select the same year as starting and ending year with that we can assure that the minimum evolution that can be displayed is a 1 year evolution.

The images 2 and 3 shows an example of the idiom.



Company 1
Paramount Pictures

Company 2
Walt Disney Pictures

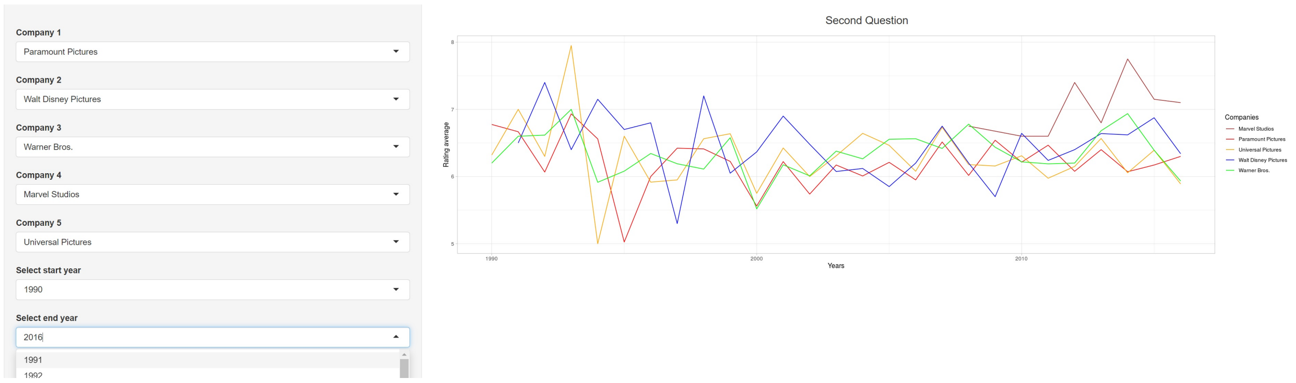
Company 3
Warner Bros.

Vinyl Films
Walden Media
Walt Disney
Walt Disney Animation Studios
Walt Disney Feature Animation
Walt Disney Pictures
Warner Bros.

1970

Select end year
2016

Figure 2: Second question: Default companies: 1970-2016



Company 1
Paramount Pictures

Company 2
Walt Disney Pictures

Company 3
Warner Bros.

Company 4
Marvel Studios

Company 5
Universal Pictures

Select start year
1990

Select end year
2016

1991

Figure 3: Second question: Maximum companies. 1988-2016

4.3 Third Question

The interaction with the visualization will be carried out by the following methods:

- **Order**: We can order by frequency.
- **Filter**: We filter the data by country and by the number of keywords.
- **Juxtapose**: We have 2 graphics that interact at the same time when the country is selected.

A lollipop chart and a word cloud has been selected for the encoding. The first one encodes the prevalent genres in a country and the second one the most frequent keywords in a country.

The X-axis of the Lollipop chart encodes the *Frequency* while the Y-axis encodes the *Genre*.

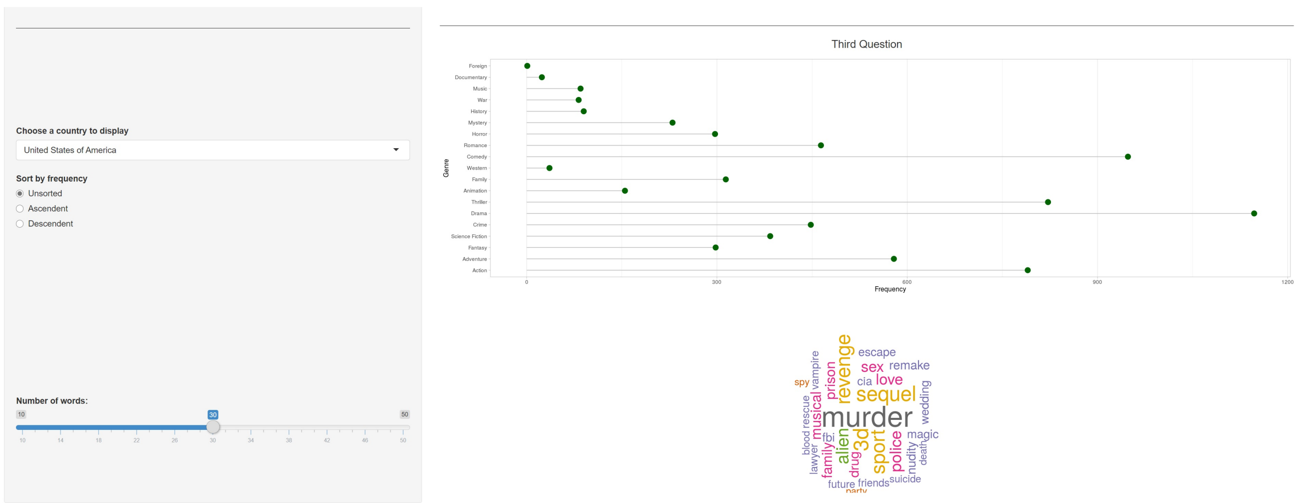
We consider that both graphics are suitable to show the frequency of the data.

It is believed that choosing the Word Cloud is not a good decision because the area is a poor metaphor and the longer words appear bigger. We agree with those arguments, however, as we are not interested in the specific value of the frequency, we think that it could be a right choice because it highlights the most important keywords.

In addition, we have set a max number of characters for the keywords to avoid the problem related with the long strings mentioned before.

Apart from that, we allow the user to select the number of words that he wants to display. However, we have set a range for the number of words displayed in the word cloud, the minimum is 10 and the maximum is 50.

The figures 4, 5, 6, 7 ilustrate the explanation about the idioms:

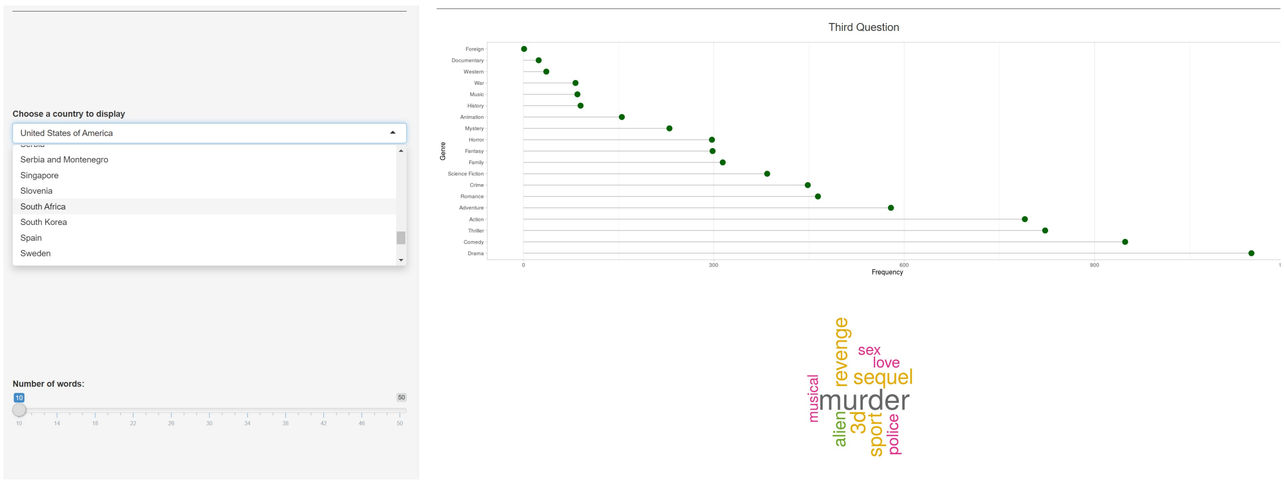


Choose a country to display: United States of America

Sort by frequency: [✓ Unsorted] Ascendent Descendent

Number of words: 30

Figure 4: Third question: USA unsorted. 30 words. Default


Choose a country to display

United States of America

Serbia and Montenegro

Singapore

Slovenia

South Africa

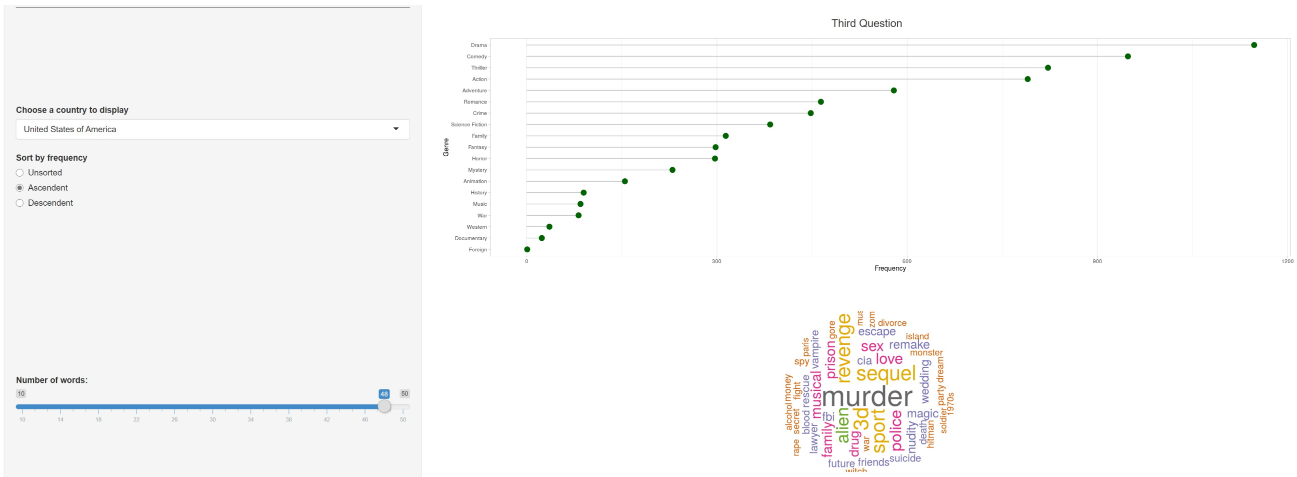
South Korea

Spain

Sweden

Number of words: 10

Figure 5: Third question: USA descendent. 10 words


Choose a country to display

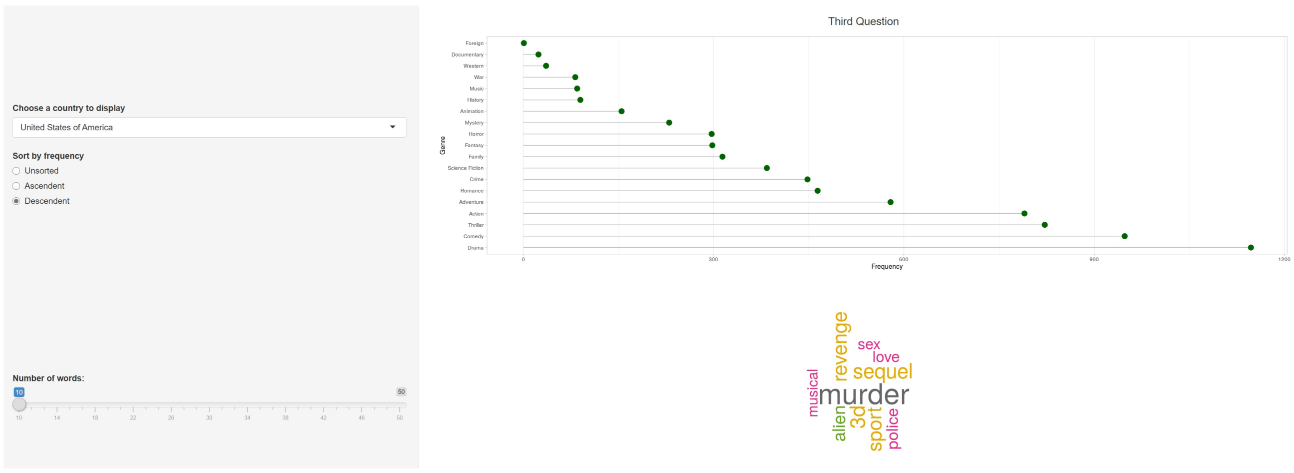
United States of America

Sort by frequency

Unsorted [✓ Ascendent] Descendent

Number of words: 48

Figure 6: Third question: USA ascendent. 48 words


Choose a country to display

United States of America

Sort by frequency

- Unsorted
- Ascendent
- [✓ Descendent]

Number of words: 10

Figure 7: Third question: USA descendent. 10 words

5 Algorithmic implementation

Before implementing our application, we have performed a exploratory data anaylysis in order to inspect the data in more depth. Thanks to this, we have been able to realize which data is worth it and which one can be removed.

In this exploratory data analysis we have carried out the following tasks:

- Keep the columns we are going to use in our application
- Remove missing values from the dataset
- Parse date to new columns
- Create new columns such as decade and earnings
- Parse JSON columns

Also, this makes the execution of our shiny app more efficient due to the fact that we simplify our dataset and the application has to deal with much less data.

5.1 First Question

In the first question, we aggregate by year and month from a selected decade and we show the result of this operation in a heatmap. In order to have the heatmap properly printed we had to factorize the years and months.

5.2 Second Question

In this question, we first do the following tasks outside the server function so that they are executed once in the execution of our application:

1. Calculate the minimum and maximum year a company has released a film and plot the evolution of three well-known companies in this range of time by default. Then the user can select another range of years to be shown.

2. Keep the companies that have published five or more movies in different years so that we can see an evolution in the line chart. There is no sense in making a line chart for a company that has only published very few movies as we are not going to be able to appreciate the evolution. Additionally, we remove repetitions and whitespaces that may exist.

3. Sort the list of companies we are going to provide by name so that the user can easily identify a company. Furthermore, we add to the beginning of this list, the option to leave a company blank (choose a company). This way, the corresponding company will not be shown in the line chart. This is the default value for the fourth and fifth companies in the line chart. Then, the user can try to add more companies in order to see their evolution.

In the server function, we associate a color with a company such that when a change in the selected companies is made, the color of the rest of the companies do not change, since this is the fault operation of the *ggplot* function. In order to solve this problem, we map a color to every button in the UI, ie, the first company will be shown in red, the second one in blue, and so on. Later on, we filter the companies by the selected range of years, calculate the rating of the films of these companies between the selected range of years and plot the results in the line chart.

5.3 Third Question

In the third question, we first do the following tasks outside the server function so that they are only executed once in the execution of our application:

1. Expanded the production countries, genres and keywords of every film as they were contained in a single cell separated by commas per every film. This way, we get a new row for every production country, genre and keyword and we can make queries to this new data. Additionally, we remove the whitespaces that may exist and the keywords that have more that seven characters so that they can be displayed properly.

2. Gest a list of countries without repetitions to show to the user and they can select a country to be shown.

In the server function, we do the following tasks:

1. Get the frequency of the different genres of the selected country and sort the genres by name in an ascendent or descendent way if the user have selected the corresponding sorting button. By default, the genres is not sorted. Then, we display the genres and their frequency in a lollipop chart.

2. Get the keywords related to the selected country and display them in a wordcloud allowing the user to control the number of words displayed in every moment.

6 Execution instructions

6.1 Dependencies

The dependencies required are the following R packages:

- Data exploration (*dataset_preparation.R*):
  - *rjson*
- Shiny app:
  - *shiny*
  - *ggplot2*
  - *dplyr*
  - *tidyr*
  - *stringr*
  - *wordcloud*

6.2 Project structure and input data

The app folder is called *movies*.

Download the dataset file **tmdb_5000_movies.csv** from: The Movie Database dataset and place it into the *movies/data* folder. The name of the dataset file must be: **tmdb_5000_movies.csv**.

Apart from that, the **dataset_preparation.R** must be executed in order to clean the data and generate the new dataset, **cleaned_tmdb_5000_movies.csv** inside the *movies/data* folder.

6.3 Execution

The Shiny app can be executed from *RStudio* or from the command line interface.

In order to run the app from the CLI you must be in the folder that contains the *movies* folder and execute the following command into an R session:

```
$ R
> install.packages("shiny")
> library(shiny)
> runApp('movies')
```

7 Application deployment

The application has been deployed to https://www.shinyapps.io/. It can be found here Big data: TMDB Movies project.

8 Final conclusions

We found this project very interesting, we really enjoyed the R programming and Shiny. The Shiny documentation was really well written so it was easy to find out how to solve the problems. The main problem was the data preparation aspect because some columns were in JSON format so we had to parse them, and the Tasks abstractions and econding tasks, even though, this project has been educative and interesting.