Top 250 Movies on IMDB

Joana Godinho 81478
João Pedro Crespo 81811
Pedro Caldeira 81888

INTRODUCTION

IMDB is one of the most well-known movie related websites and is used by many as a source for movie content.

The website offers a wide variety of lists for different purposes of research the users may have. One of the most important ones is the top 250 rated movies. This is a list of movies rated by the public, ranging from the beginning of the 1920's to the present day. Therefore, in the past years, this top has revealed cinematographic tastes and preferences in a worldwide scale.

However, the top movies list is ranked only by rating and other differences and correlations between them like the budget/gross and the awards/rating ratios are not as explicit in the website nor in any other tool.

Consequently, this visualization will seek to address a multitude of aspects about the top movies that may be of value not only to the user but also for someone in the cinematographic business.

Having that in mind, the visualization will work around the data offered by IMDB about those 250 movies and it aims to support the following three tasks:

- Task 1: **Search -> Explore**: Find correlations between years/genres/awards and top rated movies.
- Task 2: **Search -> Browse/Locate:** Browse movies by genre/rating/gross and locate a movie either by its title or the actors who starred in it;
- Task 3: **Query -> Compare**: Compare movies by certain attributes (awards won, gross, rating, budget, etc);

With the chosen tasks, the visualization not only allows the user to see the attributes of a certain movie such as gross, budget and rating, something that IMDB already offered. It allows the user to compare those attributes from different movies in a faster and more expedited way. It offers a tool to visualize correlations that may be relevant to formulate interesting conclusions.

The following tasks provide a wide set of unanswered questions that seem to be quite interesting:

- Question 1: Which is the decade with more awards from top rated movies? (**task 1**)
- Question 2: Which actor participated in a greater number of top movies? (**task 2**)
- Question 3: Which movie won more awards? (**task 3**)
- Question 4: Which movie had higher gross or rating? (**task 3**)
- Question 5: How are the top rated movies distributed over the past century? Is there a year that has a greater number of movies? (**task 1**)
- Question 6: Do the movies that have higher ratings, have more nominations/awards? (**task 1**)

The first and fifth questions, before the implementation phase, were related to the genre and to the decade, respectively. However, changing the questions to approach the decade and the year (respectively) seem to be more important and suitable to the visualization.

RELATED WORK

The motivation for this Vis emerged from another website's visualization made by Shivaraj[1]. This visualization consists in multiple heatmaps where in each heatmap are represented the 250 top-rated IMDB's movies. Each heatmap has a selectable attribute through a dropdown. For instance, the user can select the title of the movie. When the attribute is selected, the correspondent movie is highlighted in the heatmap.

While this visualization allows for the user to collect some information (e.g. what actor participated in what movies), it lacks the functionality to visualize what movies two certain actors participated together in an easy and intuitive fashion. It also lacks a feature to analyze trends and correlations between different attributes of a movie. Lastly, this visualization doesn't allow for the user to be aware of trends over time (e.g. the gross difference for different decades).

In order to address these matters, the group settled with the creation of a new visualization. This Vis would allow for the user not only to acquire the aforementioned information, but also to gather correlations by mixing multiple attributes (e.g. movies with the biggest gross have the most number of awards) and visualize the relations between the actors (what actors have ever participated on a movie together and on what movies). Finally, this Vis would let the user visualize overall trends of attributes over decades or for a certain year.

THE DATA

Initially, to obtain the necessary data it was developed a Python script that generated a JSON file with the extracted data. The script would first collect IMDb's top 250 movies ids via a request [2]. Afterwards, each ID was used as a parameter in a request to a third-party API [3], which collects each movie's cast, rating, budget, year, amongst other attributes.

However, this API did not return any information concerning each movie's awards nor its actual total worldwide gross. Therefore, the group needed to resort to web scrapping techniques to access two different web pages [4] [5] and retrieve the missing data.

After extracting the data, the group realized that in order to compare monetary values such as the budget and gross of two different movies from two different years, they had to be adjusted to inflation. This way, the comparison would be fair and accurate. As a result, two derived measures were created: *worldWide_Gross_Inflated* (movie's worldwide gross adjusted for inflation) and *budget_inflated* (movie's budget adjusted for inflation), derived from attributes *worldwide gross* and *budget*, respectively.

However, some movies had no data referring to the budget or gross. Seeing that those cases were very rare, the missing data was completed by "hand"**,** after researching for it.

Subsequently, the data needed to be cleaned, since each movie had several unnecessary attributes such as *rating_count*, *description*, *trailer* among others. During this phase, another problem emerged. The fields *image* in cast and *poster* contained an URL for the actor/actress's image and for the movie's poster, respectively. Since the project needed to be stand-alone, it was created a folder with the actors/actresses' image and the movie's poster. The image and poster fields were then changed to the file path containing the respective image. The cast was also filtered in order to contain only the stars.

After all this process, a static dataset was generated containing all the filtered and completed data. The dataset consisted in only one JSON file, that had for each movie it's budget, gross, rating, number of awards etc.

Nevertheless, during the implementation phase, the group realised that using just one file would result in scalability and complexity problems. The visualization would take longer fetching the data, during loading and during interactions between the different idioms.

Therefore, three new JSON files were created. The first one gathers for each year, from 1920 till 2017, the movies released in that year. The second one contains for each decade, from the same period as the years, genre attributes such as gross, budget, number of awards etc. Finally, the third one, retains the stars of all 250 movies and their relations (i.e. the actors/actresses that stared together in at least one movie and the correspondent movie).

VISUALIZATION

Overall Description

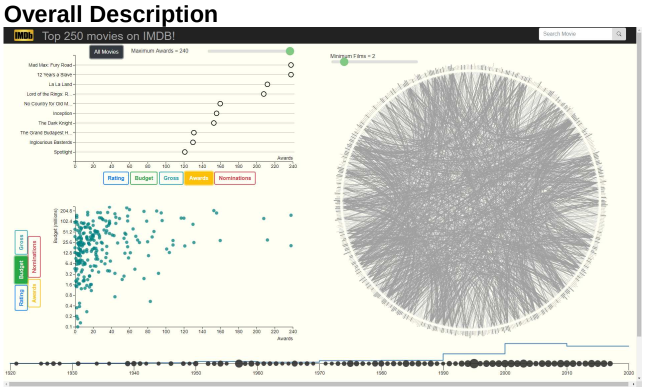

Figure 1. Vis Layout

The Vis consists in a static display of four idioms in the same window distributed as showcased in Figure 1. Depending on the idiom, the data is filtered and differentiated in distinct ways, but all of them in the end work together. However, the visualization offers a search box where the user can input the name of a movie (see Figure 2), allowing him/her to see the information of that particular movie. This functionality is related to the task 2 described in the introduction, since the user can locate a movie by its title.

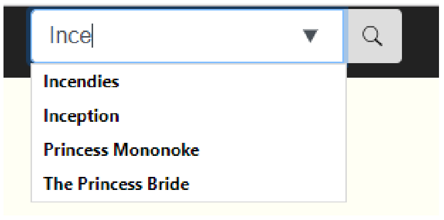

Figure 2. Detail search box in the right corner

This section will now present in detail both the idioms and the way they interact with each other.

Movies' Cleveland
This idiom has one main goal: representing movie details. Consequently, the user can compare several movies.

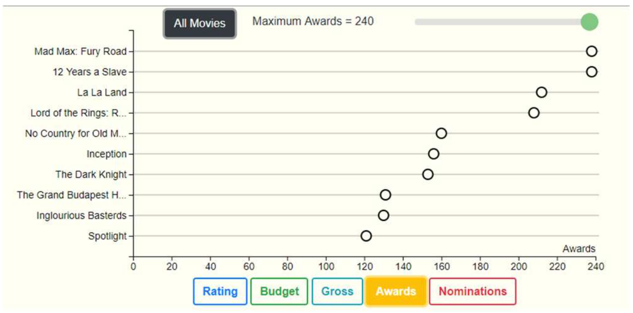

Figure 3. Movies' Cleveland overall

There are five buttons along the x axis labelled as rating, gross, budget, awards and nominations (see Figure 3). There is also a slider to establish a maximum value for the x axis attribute displayed (see Figure 3). It's "default" value is the highest value for the x axis attribute in the dataset.

In the y axis the user can observe the names of ten movies. Those ten movies will depend on the attribute selected. Along the x axis the user can see the value of the attribute selected.

The names in the y axis will be displayed in descending order depending on the button selected. These names will also depend on the value established in the slider. The items displayed will correspond to the ten items with higher value for the attribute selected, within the slider range.

For instance, if the user selects the button budget, the visualization will display the budget for the ten movies with higher budget of all the 250, in descending order (the first movie has the higher budget of all top movies). If then the user drags the slider to a certain value, the ten movies displayed will correspond to the ones with higher budget, considering only the movies with a budget smaller or equal than the slider value (see Figure 4).

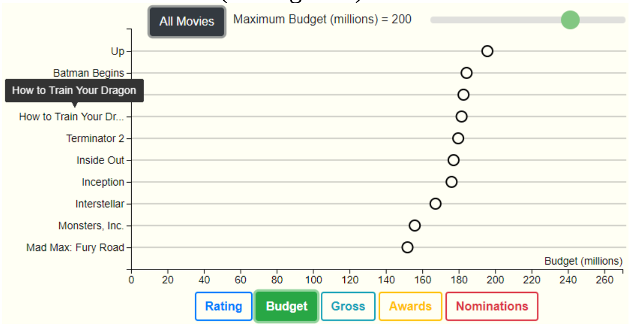

Figure 4. Up is the movie with highest budget of all the movies with budget smaller or equal to 200 million.

The interactivity for this idiom is not only in the buttons and the slider, as explained before, but also in hovering the circles in the *Cleveland*. If the user hovers one, the visualization will display, for the corresponding movie, the actual value for the attribute selected in the x axis. It will also highlight in orange in the *Actors' Circos* the actors who starred in the movie, in the Timeline the movie's year and in the Scatter the correspondent dot (see Figure 13). Then, if the user clicks, the information sticks and is now coloured in blue (see Figure 14).

If he/she hovers a title in the y axis the visualization will display the item's full title, since in the axis the names are sliced (see Figure 4).

Movies' Scatter Plot
This idiom will display the correlations between rating, budget, awards, nominations and gross, or the absence of them.

The user will be able to choose which attributes to visualize in both the x and y axis. For the y axis the user selects one of the buttons alongside it. For the x axis the attribute selected will depend on the button selected in the *Cleveland Dot Plot* (see Figure 5).

Each dot corresponds to a movie and the dot's position is related to the attributes selected for both axis.

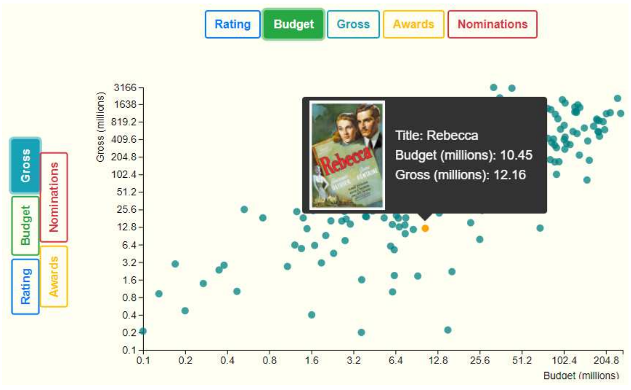

Figure 5. Scatter Plot overall with movie window shown when hovering a dot.

The interactivity is in the mentioned buttons and in hovering and clicking a dot. If the user hovers a point, the visualization will display a small window with the movie's name, poster and the actual values for the attributes selected in each axis (see Figure 5). It will also highlight in orange in the *Actors' Circos* the actors who starred in the movie and in the *Timeline* the movie's year, similarly to the hover in the *Cleveland* (see Figure 13).

By clicking in a dot, the visualization displays the movie information in the other idioms, just like when you click in the Cleveland dot plot (see Figure 14).

Timeline
This idiom will display the distribution of the top movies between the years 1920 and 2017. It consists in a horizontal axis with a circle for each year when at least on top movie was released. The size of the dot will encode the number of movies released in that year (see Figure 6).

If the user hovers a point it is shown a window with the posters of the movies released in that year (see Figure 15**)**. By clicking in a dot, the visualization displays for all the year's movies its information in the other idioms, similarly to when the user clicks in a dot from the scatter plot.

The timeline also has a mix of a bar chart with a line chart where you can see the total amount of whichever attribute is selected in the *Cleveland* buttons, for each decade (see Figure 6).

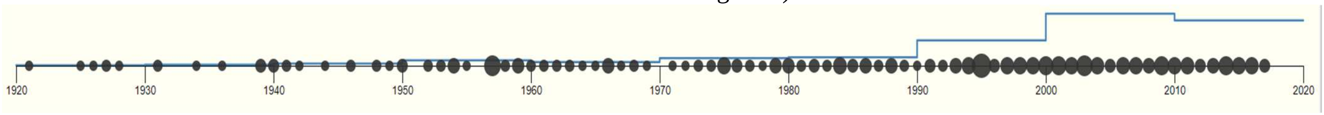

Figure 6. Timeline with the line-bar chart showcasing the awards.

Actors' Circos

The *Circos* displays the actors who starred in at least one top movie. It also presents the connections between them. If two actors participated together in one top movie they have a link between them (see Figure 7).

If the user hovers a connection it is shown a window with the movie(s) where the correspondent actors participated. Also, the connection's and respective actors' colours change to red. The same happens when the user hovers an actor, except for the window, since all his/her connections are highlighted (see Figure 7).

There is a slider to set a minimum number of movies to filter the actors. The ones who didn't star in a number of movies higher or equal than the minimum will be underemphasized (see Figure 7).

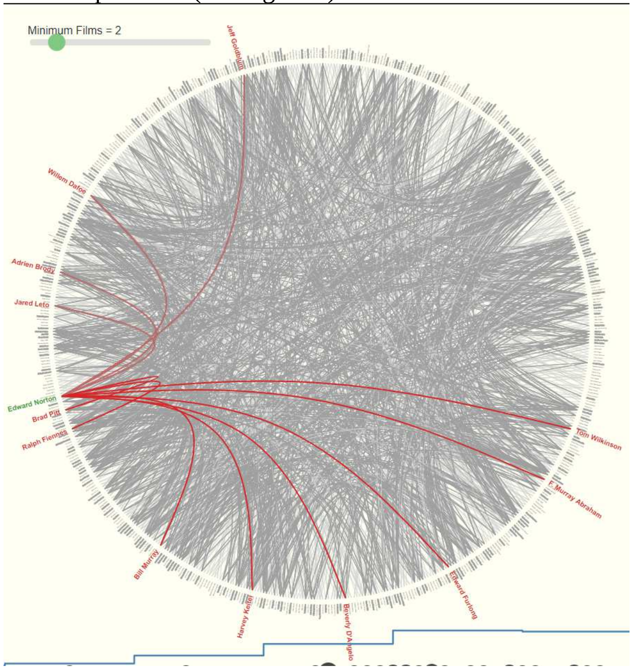

Figure 7. Hover over an actor in the Actors' Circos

Rationale

Since our dataset is small and very rich in information, there were a lot of ideas in how to transmit it in a consistent and interesting way.

Therefore, during the implementation phase, the Vis suffered a few changes. In the next subsections, it will be detailed the evolution of every idiom as well as the issues related to them.

Movies' Cleveland

Since one of the main tasks set for the visualization was to compare movies, there were to possible idioms that would allow that: a bar chart and a Cleveland dot plot. The Cleveland was chosen, due to aesthetic and consistency reasons. Since the other idioms were based on a set of circles, encoding the movies as circles and not bars would fit much better with the other idioms.

In the Cleveland, each dot represents a movie. However, initially, the dot would represent a movie or a genre depending on the context. The reason why the encode change will be explainer further down.

To implement the Cleveland buttons functionality, the Cleveland had to be scalable not only to the dataset, but also to the dynamic change of domain in x the axis. As a result, the maximum and minimum in the ticks' domain changes accordingly to the range of values for the attribute selected, provided by the dataset. This way if in the future we would like to present more attributes, the code is robust to handle it.

The idiom suffered some changes since the sketch phase (see Figure 8). The first change was replacing the dropdowns by a set of buttons, in order to be easier for the user to see all the options available immediately (see Figure 9).

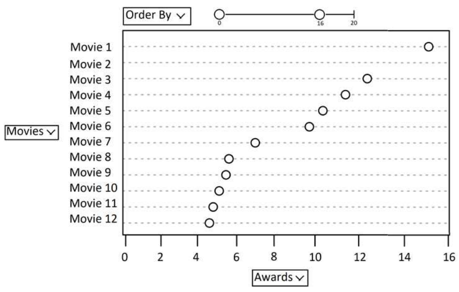

Figure 8. Cleveland's first sketch.

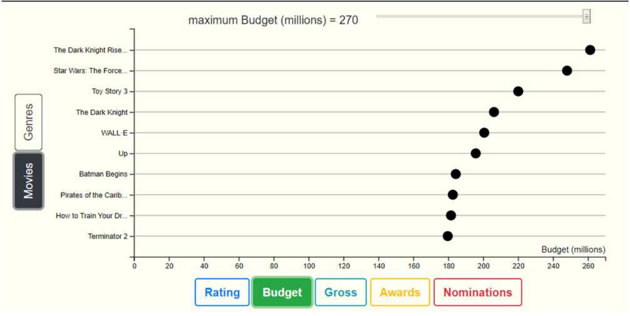

Figure 9. Cleveland's first functional version.

Then, the option *genre* was removed, since it required a lot of work to fit the genres with the desired interactivity between idioms. Considering that the main reason we implemented the genre functionality was to answer one question, we opted to change it and to remove the functionality (leading to the change on the encode).

Movies' Scatter Plot

In an early phase, the group set that one of the visualization's main goals would be to provide means to find interesting correlations, like if the most expensive movies are the ones with higher rating or gross. As a result, it was decided that the scatter plot would do the work since it's the best technique to visualize correlations.

In this idiom each movie is represented by a circle and the color is used to distinguish the movies that were clicked. We considered using one color for each movie selected to distinguish in the other idioms which actor/year was related to which movie. However, that would be too complex since the user can select an arbitrary number of movies. As an alternative, the user can see the actor and year of a certain movie by hovering it.

Considering the implementation explained in the Overall Description section, the scatter had to be scalable to the dataset and to the dynamic change of domain in both axis. Consequently, we adopted the same strategy as we used in the Cleveland.

We were also able to reduce the complexity of the data by using a logarithmic scale, when the budget or the gross attribute was selected. Without this scale the points would be too close to each other, which would affect the interactivities and the readability.

The idiom did not change considerably since the sketch phase. First the dropdowns were replaced by sets of buttons, like in the Cleveland (see Figure 10). Afterwards, during the implementation, the x axis buttons were removed. The x attribute was then selected by the Cleveland buttons.

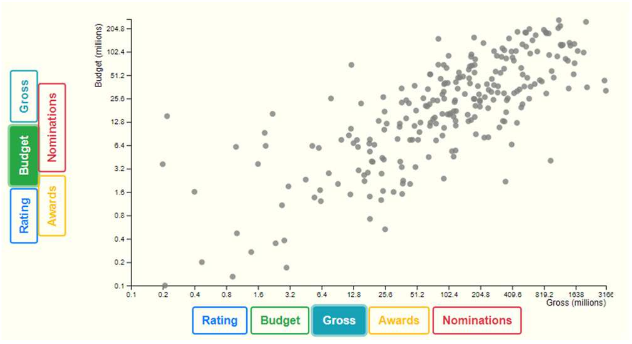

Figure 10. Scatter Plot's first functional version.

Timeline

Initially, the *Timeline* was only composed by a horizontal axis and several circles. Each circle visually encoded a year and the size of the circle reflected the number of movies released in that year. This encode is still used in the last version.

This seemed to be a simple and appropriate technique to visualize the distribution of the top movies throughout the years. The scatter could have been used to provide this feature. Nevertheless, we considered that the horizontal axis with the circles would give a more chronological feel to the distribution.

After the sketch phase, the idiom only suffered one extension, the decades line chart. We thought it would be a valuable addition seeing that it showcases information that the user would have to research and gather by himself. It would also fit and connect well with the other idioms.

In the beginning, there were a few complexity and scalability issues, because we were only working with one JSON file that had for each movie the attributes mentioned before. However, that organization led to an unnecessary waste of computational power, since the movies needed to be grouped by years. So, if in the future, the visualization would consider more years, that waste would affect the loading time. Consequently, we created two JSON files one for the years and another one for the decades, as mentioned in The Data section.

Actors' Circos

In the *Circos*, each connection encodes a pairwise relationship between two points (actors). It connects two actors who starred together in a certain movie. The colour intensity reflects the actors who participated in a minimum number of movies (the number inputted in the slider) and their connections. The green highlights an actor when he is hovered and the red highlights his connections and actors he is connected. When a link is hovered, the red encodes that link and the actors connected by it. The blue highlights the actors/actresses who starred a selected movie and their connections.

This idiom was the only one which basically did not change since the sketch phase. The only difference between the last version and the sketch was the use of a slider instead of a dropdown to filter the actors by the number of movies.

In terms of complexity and scalability we found the same issues as in the timeline, the file's organization. Therefore, the same solution was applied, create a new JSON file with the actors and their connections.

Demonstrate the Potential

To demonstrate how the Vis works, this section will provide answers to three different questions: Which top movie had the lowest gross? Which is the decade with more awards from top rated movies? Was the movie Jaws profitable?

In the end, there will be some insights that were not planned to be discovered, yet the Vis brought them up.

Which top movie had the lowest gross?

This question is not only related to the task of comparing movies, task 3, but also to task 2 since to answer this question the user needs to browse movies by gross.

To answer the question the Cleveland dot plot is more than enough. First the user needs to click on the button labelled "gross". The Vis will then present the ten movies with higher gross in descending order of gross value. By dragging the slider to its minimum, the user can see the movies that had the lowest gross in all the 250 movies (see figure 11). Then, the answer is the last movie, "Memories of Murder", since they are sorted, as mentioned.

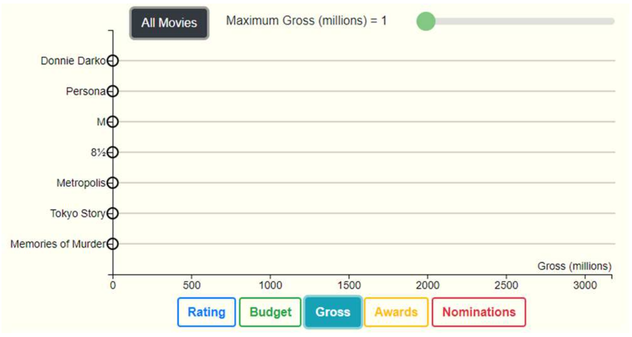

Figure 11. Illustrates the lowest grossing movies.

Which is the decade with more awards from top rated movies?

This question is one of many that demonstrates how the user can find correlations between years and certain movie attributes. As a result, it's part of the task 1 domain.

To answer the question the user needs to resort to the *Cleveland dot plot* and to the *Timeline*. First the user needs to click on the button labelled "awards", in the *Cleveland*. Then, the *Timeline* will adjust the line bar chart to demonstrate the distribution of awards along the decades. The answer is the decade with the highest step, in this case is the 2000 – 2010 decade.

Was the movie Jaws profitable?

For this question the user needs to locate the movie in the scatter by its title, something that converges with the task 2.

The first step to answer this question is to input the title "Jaws" in the search box located in the screen top right corner (see Figure 2). As a result, all the information related to Jaws is highlighted in almost all the idioms. Then, the user needs to click on the budget button in the *Cleveland* and on the gross button in the *Scatter*. Afterwards, by hovering the only red dot in the scatter the user can see if the Jaws' gross was higher than its budget, answering the question (see Figure 12)

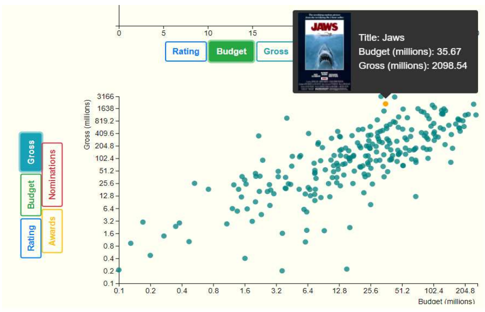

Figure 12. Detail Jaws Budget and Gross in the Scatter

Golden Insights

One insight that the group noticed was that, when selecting gross in the Cleveland, the movie with highest gross of all the 250 movies was "Gone with the Wind". A movie released in 1939. We were expecting that the first place would go to one of the most recent blockbusters like "Star Wars: The Force Awakens".

Another one is the actor who started in more top rated movies, John Ratzenberger, a not so well known actor. The group was expecting a more famous and awarded actor. To see this the user only needs to drag the slider to its maximum and the only actor emphasized is John Ratzenberger.

IMPLEMENTATION DETAILS

The *Movies' Cleveland* was adapted from an example of the D3 page. Building the *Cleveland* was not very difficult, however, there were some challenges in implementing the slider and buttons functionalities, since the code had to be abstract enough to deal with the constant changes.

This idiom also required that the data would be sorted in descending order by a certain attribute. In order to that we used a d3 function called *d3.descending* which does exactly what was intended.

The *Movies' Scatter Plot* was adapted from the classes D3 Tutorial. Since we had done the tutorial, building the scatter was not complicated.

The difficult part was in the dynamic change of the axis attribute, because not all the attributes had the same domain and scale. As a result, we created a dictionary that keeps for each attribute certain variables (like its scale and minimum).

The *Timeline* was built from scratch, however for the line-bar chart it was necessary a certain help from a line chart example in the D3 page. Initially, the dataset revealed to be a little complex to build all the timeline, because it required a lot of work during loading time. However, after extending the dataset, as mentioned in The Data section, that was no longer an issue. Afterwards, only the line-bar chart was challenging, since it had to change every time an attribute was selected in the *Cleveland*.

The *Circos* implementation was inspired in another D3 based visualization. This idiom was particularly challenging since the structure of the data we gathered beforehand wasn't very suitable for the implementation, as mentioned before. Besides, the inspiring visualization was meant to be used with a hierarchy-based data. Since the data for the circle did not share the same structure of the used on other idioms, the interaction with other visualizations had to be carefully done, making a map of the attributes.

The group had different ideas on how the filtering by the minimum number of movies participated would work. On the one hand, there was the option to redo the circle every time with less or more links/actors. On the other hand, there was the option to just highlight the links/actors that fulfil the requirements. The group tried both implementation. The first one ended up taking too much computational power, making the visualization more clunky with load times way higher than the believed acceptable. The latter option turned out being computationally efficient, so it was decided that it would be the approach applied to the idiom.

The connections between all the idioms did not require any complex technique. They were easily done with the manipulation of classes and with event listeners.

CONCLUSION

The group, by doing this project, was exposed to the challenges and intricacies of creating a new visualization, from data gathering to idiom interaction. The project was supported by the JavaScript D3 library, so there was also a challenging learning process with the software.

The project was also very educational in the sense that incited healthy discussions between the group members (e.g. on whether something should be done/look this or that way) due to freedom on implementation and appearance of the visualization.

It also demonstrated to the group members that sometimes the change of small details has a big difference on the appeal of what the user sees.

The group concluded that the idioms used, namely the *Circos* and the *Timeline*, worked well on the visualization since the database wasn't too big. If the project managed data of a larger scale the idioms would have to suffer some adaptions (which ended up happening in the circle anyway, due to the enormous amount of links it created).
Overall, the group really enjoyed the creation of this visualization and all the inherent challenges. It was a fun iterative journey. Merry Christmas.

REFERENCES

1. https://public.tableau.com/profile/shivarajc#!/
2. request to page http://akas.imdb.com/chart/top
3. http://www.theimdbapi.org/
4. http://www.imdb.com/title/{MovieID}/
5. http://www.imdb.com/title/{MovieID}/business

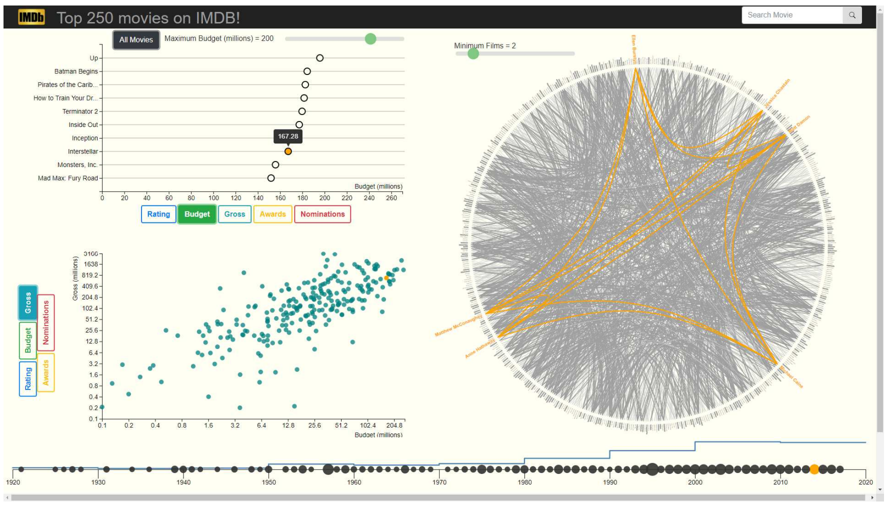

Figure 13. Hover in a Cleveland dot. The hover in the scatter also highlights actors in the circos and a circle in the timeline.

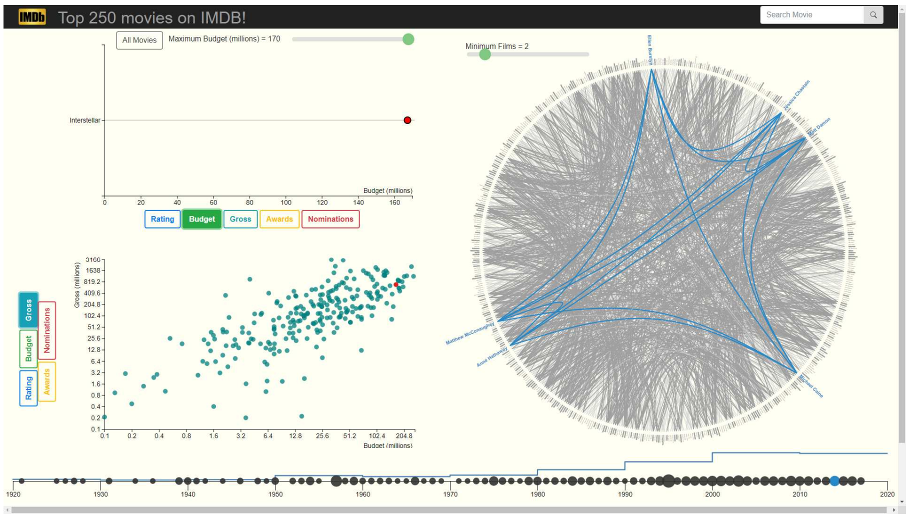

Figure 14. Click in a Cleveland dot. The click in the scatter does the same thing.

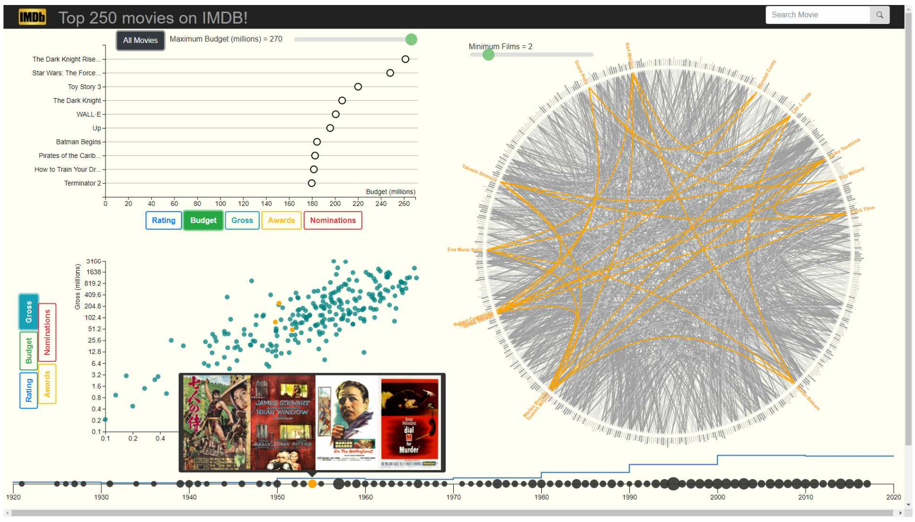

Figure 15. Hover in a Timeline circle.