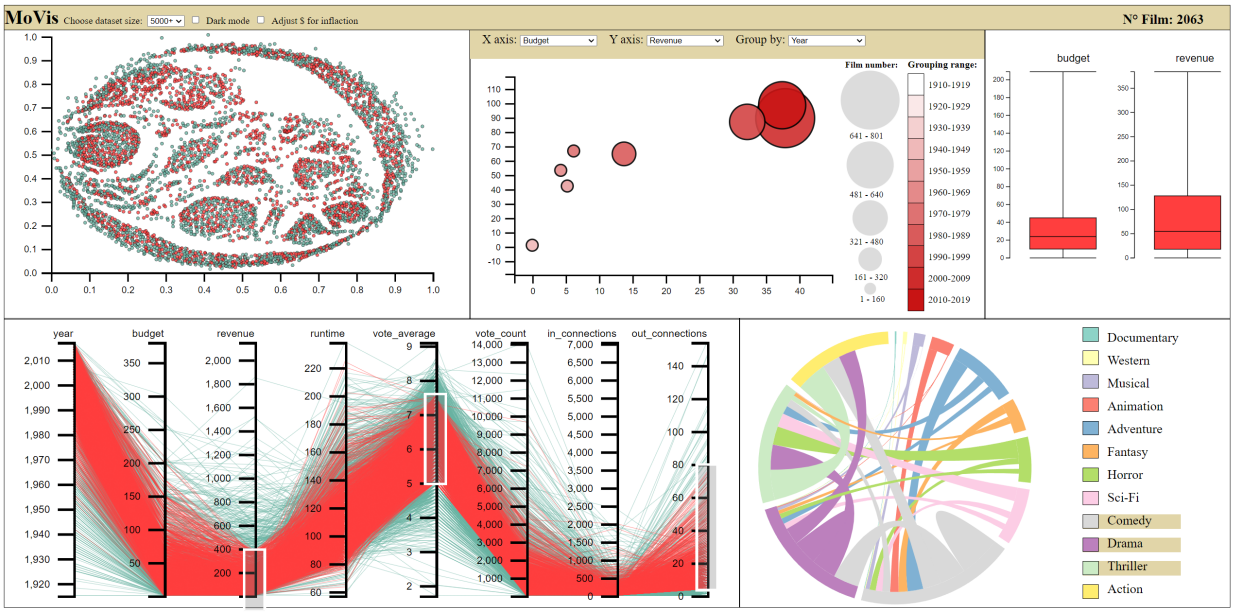
Sapienza University of Rome

Engineering in Computer Science

MoVis

Visual Analytics
Final project

A.Y. 2021/22



Professors:
Marco Angelini
Giuseppe Santucci

Students (alphabetical order):
Giovanni Pecorelli 1799865
Jacopo Rossi 1801667
Giacomo Venneri 1810169

Introduction

The movie industry has been producing movies for more than a century already, and it has put together a very extensive catalogue. The whole collection of movies can be found on the IMDB.com website, with comprehensive information about nearly every movie ever made. Unfortunately IMDB lacks a tool to visualize all of this catalogue entirely, and it is more useful to browse the movies' pages one at a time.

Our project, **MoVis**, aims to solve this shortcoming, providing a tool to explore visually the IMDB dataset, and more generally the whole history of the cinema industry production. On top of this, our visualization integrate information from different sources, expanding the original datasets already available on the web.

The final tool is an interactive visualization where different parts work together to give the user an accurate overview of the whole dataset, which can be used to highlight patterns, find outliers and more.

This tool is directed to movies enthusiasts, and can be used in different ways:

- to discover new movies based on personal preferences. It is possible to apply filters to different movies' characteristics, and then explore the dataset to find a new movie to watch;
- to find movies similar to the ones you like: a user can find his favourite films and thanks to a dimensionality reduction visualization ideally it will be able to find out similar ones, which will be visually close to it;
- to explore the current state and the history of the movies industry as a whole, and find trends, insights, statistics and curiosities.

Related works

Before the development of the project, to have an idea of what we could do we analyzed the literature, studying papers with topics similar to ours.
We could not find any interactive visualization in our research, but nonetheless found out interesting studies published over the years, discussed in the following paragraphs.

In *Correlations Between User Voting Data, Budget, and BoxOffice for Films in the Internet Movie Database (2015)* **[1]** the authors base their work on the IMDB dataset, the same starting point of our project. The paper is a study about the correlation between user voting data and economic film characteristics. The conclusion of this study is that the number of user votes is a better indicator of a film's prominence compared to the average rating of the film; therefore a film's budget is overwhelmingly the most relevant factor in determining a film's ultimate prominence.
To support this thesis, the authors consider movies metadata such as the year of release, country of production, primary language, user voting statistics, and several types of financial information, including the film's budget and box office gross in the United States.

This paper was an inspiration for us, and in our visualization we decided to give the user the possibility to interact with the aforementioned features, but with the proper changes, along with others. In **[1]**, movies from outside the U.S.A. and not in English language are excluded from the study; this is not the case in our work.
Also, in **[1]** financial data are preprocessed to take into account the effect of inflation throughout the decades,

and normalized using statistical heuristics. We took note of this problem and considered it when building our dataset, adjusting the financial data in our own way.
Another interesting feature used in this paper is the notion of a "movie connection". We found this particularly interesting and further researched the topic.

Eventually *Quantitative approaches for evaluating the influence of films using the IMDB database* **[2]** was found. This paper focuses on the concept of a movie connection, and uses it as a starting point to analyze what are the distinctive characteristics of an influential movie, and when to start considering it as such.
A movie connection is defined in two ways, depending on its direction. On the one hand, the references made to a film in subsequent movies; and on the other, the references made in this film to previous ones: the former is called an "incoming connection" and measures the influence that a film exerts on the future, while the latter is an "outgoing connection" and measures the influence that the cinematic past has exerted on it.

Our project integrates the notions presented by **[2]** and expands them, considering not only the number of incoming and outgoing connections, but also the connected movies themselves. The website of IMDB made this possible since offers for every film a specific section related to movie connections, where it presents a list of all the movies that are considered connected to it by the website users.

In a later chapter we will try to replicate some of the experiments presented in the above-mentioned publications, comparing the results and verifying their compatibility.

During the development phase we came up with the idea of using a chord diagram in our visualization. To make the most of this element we researched and studied technical papers that implemented it, even in different context than ours.
Eventually we found *Visualization of Enrollment Data using Chord Diagrams (2015)* **[3]**, a paper where there was mention of the Shneiderman's principle. This mantra, enunciated for the first time in *The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations (1996)* **[4]** states that: "Overview first, zoom and filter, then details-on-demand".
We applied these three rules when designing our chord diagram, as we will explain in a later section.

Data

Creating the dataset

The main dataset used for the project is the **Kaggle Movie Dataset** (https://www.kaggle.com/rounakbanik/the-movies-dataset?select=movies_metadata.csv). It's a collection of movies taken from the IMDB website updated to 2017, containing more than 45000 entries, with 24 different attributes. The *AS index* would be way too high if we were to use all of those features, and the visualization would be too heavy and totally unresponsive. For this reason we selected only some of them:

- *imdb_id*: the unique id of the movie, used as a primary key for the whole dataset and used in many ways, from joining purposes to data mining. It's a string formatted like "tt0123456";
- *title*: the title of the movie, a string;
- *release_date*: the release date of the movie. It's a string formatted like "1970-01-01";
- *budget*: the budget of the movie in dollars, an integer;
- *revenue*: the revenue of the movie in dollars, an integer;
- *runtime*: the movie duration expressed in minutes, as a float (ex. 120.0);
- *vote_average*: the average rating of the movie by the users of (https://movielens.org/);
- *vote_count*: the number of ratings of the movie by the users of MovieLens;
- *popularity*: a float value usually between 0 and 30, with some outliers that reach values of hundreds. It is computed by IMDB with an "in-house formula", that considers the rating and the number of votes, among other factors;

On top of this we added another attribute, *director*, the director of the movie. We got this by parsing and joining the contents of the *credits.csv* file, which contains full information about the cast and crew of every IMDB movie. We decided to consider only one director for each movie, in case there were multiple.
This data manipulation (as well as all of the following) is done in Python, using libraries such as `numpy` and `pandas`.

Another column that was added is the *genres* column. The movies' genres are taken from another source, the **MovieLens 25M Dataset** (https://grouplens.org/datasets/movielens/). Since the genres were too many and too sparse, we choose only 12 of them, incorporating them into more inclusive categories ("noir" and "crime" movies go into "thriller") and eliminating others completely ("romantic" movies). The final categories are: *Documentary, Western, Musical, Animation, Adventure, Fantasy, Horror, Sci-Fi, Comedy, Drama, Thriller, Action*. These are saved in the dataset as strings separated by "|" (for example: `action|adventure|sci-fi`).

Then, we added 4 more columns with data about the relationship between different movies. This was done by doing a web scraping of the IMDB website (using Python library `Beautiful Soup`) to obtain information directly from the HTML code of every `imdb.com/title/imdbID/movieconnections` page. In particular, we wanted to know the number of "incoming" and "outgoing" connections, as well as the total number of them. A connection is defined as a citation, an homage, a reference or a direct sequel/remake. In addition

to this, we also wanted to know which were these connected movies. However this turned out to be quite useless, as sometimes movies have a number of connections in the order of the thousands.
So, we narrowed our definition of a "strict connection": we consider two movies strictly connected when one is a reboot, remake, sequel or prequel of the other. In this way we assure that the list of connected movies does not grow beyond a dozen movies.

Finally, we noticed a problem in the dataset regarding the columns "budget" and "revenue" and decided to take care of it. The problem is the following: because of the long-term effects of inflation on the value of the american dollar, the values of earnings unadjusted for inflation gives far more weight to later films. Therefore it is quite meaningless to compare films widely separated in time. We must consider for example that 1$ during the 1910s is worth almost 30$ in today's standards.
So we created another two columns, *actual_budget* and *actual_revenue*, representing the value of the previous columns adjusted for inflation, expressed in today's dollar value.

As an example, this would be a row in our final dataset, out of the 5205 total, saved in a .csv file:

```
imdb_id, title, year, budget, actual_budget, revenue, actual_revenue, runtime,
vote_average, vote_count, popularity, movielens_id, tmdb_id, genres, director,
in_connections, out_connections, tot_connections, connected_movies

0076759, Star Wars, 1977, 11.0, 50.6, 775.4, 3566.84, 121, 8.1, 6778, 42.149697,
260, 11.0, Action|Adventure|Sci-Fi, George Lucas, 7037, 28, 7065,
0080684|0086190|0120915|0121765|0121766|2488496|2527336|2527338|2199603|1490713|0076759
```

Later in development phase we decided to create a second, smaller dataset composed of only the 250 top movies. This was done because using the full dataset, with real-time animations on thousands of elements, can be challenging on low-end PCs because of the amount of processing power required by the browser page.
To filter the movies we used the *popularity* attribute, and took the first 250 movies with highest value.
Finally we also created an intermediate version of it, containing the top 1000 movies.

Dimensionality reduction

The dimensionality reduction approach used in our project is to do a preprocessing phase where we compute a dissimilarity matrix between all the movies, and then integrate the results of this algorithm into the dataset. The result of the dimensionality reduction are two arrays of numbers, that represent X and Y coordinates: these will be used to plot each movie as a point in a scatterplot.

The reason why this procedure is not done at runtime is that it's a very heavy task that takes nearly an hour to complete, using Python libraries `pandas, numpy, matplotlib and sklearn`.

For our project we will use MDS as a dimensionality reduction technique because it allows to define and use different distance functions for different types of attributes, such as Jaccard distance for categorical data and euclidean distance for numerical data.

The attributes used for our MDS are the following:

genres, connected_ movies, titles, release_ date, budget, revenue, runtime, vote_ average, vote_ count, popularity, in_ connections, out_ connections.

Each attributes computes its own dissimilarity matrix, in a different way.

With numerical data we chose to use the euclidean distance between the two values: $(n_1 - n_2)^2$

Instead when the data are categorical (ex.: genres, connected movies, titles, directors) we used custom distance functions, that returned values between 0 and 1:

- the *director* attribute generates a boolean matrix filled with 0 when two movies share the same director, 1 otherwise;
- the *titles* attributes is considered in an attempt to plot closer together movies that have very similar titles, meaning that the two are likely connected to each other. The values range is in $[0, 1]$ as before, only this time the value is a float;
- the *connected_ movies* and *genres* attributes use the jaccard distance $J(A, B) = \frac{|A \cap B|}{|A \cup B|}$ between the two lists.

All of these matrices are then summed into one final matrix, with different weights to give more importance to certain attributes, and to cancel the differences between the attribute domains (for example average votes go up to 10, whereas the number of votes goes up to 14000).

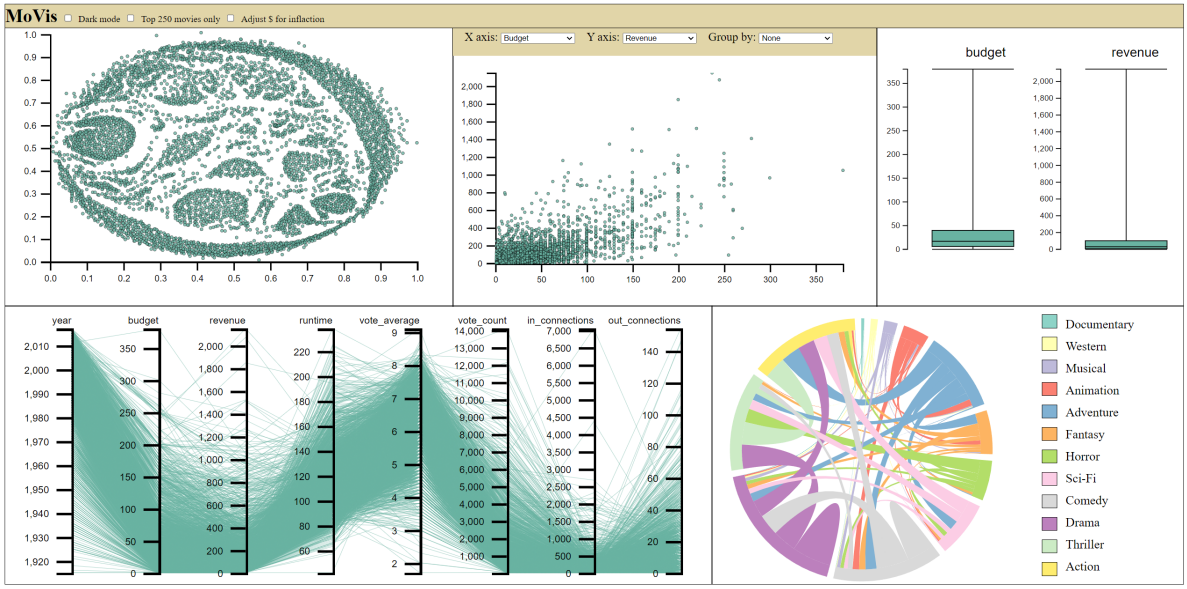
This matrix is then used to compute the MDS components using `sklearn.MDS` with parameters `n_components=2, max_iter=1000, eps=1e-9, dissimilarity="precomputed", random_state=0`.

These components are then joined into two new columns in the main dataset of the project.

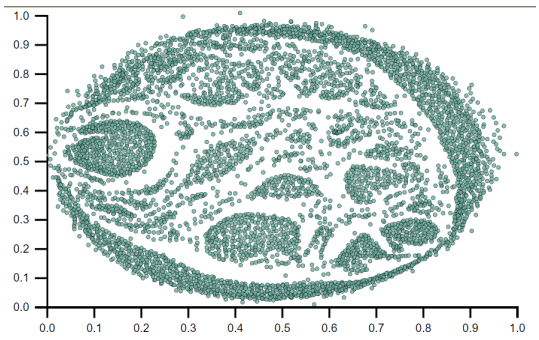
Visualizations and Interactions

Visualizations

This is the MoVis webpage. It has been built with the use of the `D3.js` Javascript library and CSS. The development was done in VSCode using the Live Share extension.



The Scatterplot



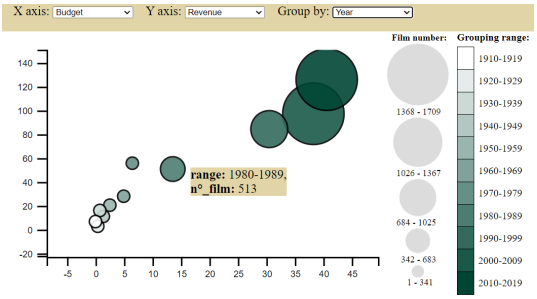
In the scatterplot we visualize the results of the MDS. With the use of functions such as `d3.axis`, and `d3.circle` we plotted every movie from the dataset as a small circle, that has its *cx* and *cy* attributes taken from the MDS X and Y columns in the dataset. The closer two movies are, the more similar they are. The most important attributes taken in consideration to do this are the genres, the connected movies and the average vote. The final result is shown in the figure.

Note that the axes' domains have no particular meaning.

When the user hovers on a specific circle a tooltip appears, giving info about the movie (specifically the title, the year and the director). The chart can also be zoomed in and out scrolling with the mouse wheel or double-clicking, and can be moved with the mouse.

The Bubbleplot

In the bubbleplot we can visualize nearly every attribute present in the dataset. It's possible to choose the attribute to use on both the X and Y axes, as well as the attribute used to aggregate the data. For example, choosing a grouping by year, the bubbleplot shows a bubble representing each decade, from 1910s to 2010s. The number of movies that compose each bubble can be visualized in the size of the bubble.

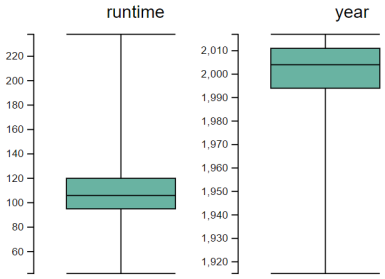


It's also possible to ignore the aggregation. In this case, when choosing "None" as the grouping attribute, all the movies will be represented as a small circle, like in the MDS scatterplot.

In this chart too it is possible for the user to hover on the circles. If the circle represent a single movie, its title will be displayed in the tooltip; otherwise, if the circle is a bubble and thus represents a collection of movies, the exact number of movies is displayed in the tooltip, along with the range that the bubble represents (for example: the decade, or the votes range, or the min and max values of the other attributes).

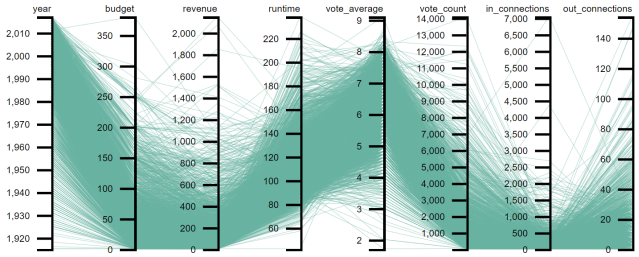
The Boxplots

The two boxplots are used to show the distribution of the two features momentarily selected on the X and Y axes. They show a five-number summary of the data: the minimum, the maximum, the sample median, and the first and third quartiles.



The Parallel Coordinates

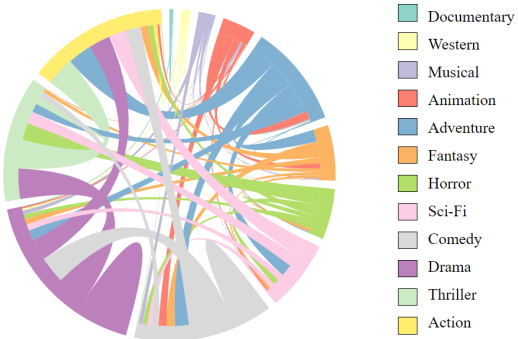
To represent our high-dimensional dataset we used a parallel coordinates visualization. It shows 8 of the features of our dataset: Year, Budget (in million of dollars), Revenue (in million of dollars), Runtime (in minutes), Average vote, Votes number, Incoming connections and Outgoing connections. These are all the continuous (not categorical) data present in the dataset. Each movie is represented as a polyline that goes through the columns at the appropriate height.



The user can interact with this visualization by brushing one or more columns at the same time. This results in the brushed lines being selected and changing color.

The Chord Diagram

A chord diagram is one of the most suitable ways of displaying the inter-relationships between data in a matrix. The data are arranged radially around a circle with the relationships between the data drawn as arcs connecting the data.
We decided to use it to display visually the movies' genres and their occurrences. Each genre is represented by a fragment along the circumference of the circle; arcs are drawn between fragment to show flows: the thickness of the arc is proportional to the significance of the flow. This means that genres that occur often together will have more relevance in the visualization.



Movies that have only a single genre will contribute in forming hills (self-connected arcs).

The colors used for this elements have been chosen thanks to this website: https://colorbrewer2.org/#type=qualitative&scheme=Set3&n=12.

This visualization can be interacted with in two different ways: clicking on one or more genres in the legend will highlight them and the chord will update to display only movies that include those genres.
Instead, if the user wants to focus on a specific couple of genres, it is possible to click on a single arc: this will highlight it; furthermore, when hovering the mouse over arcs, a tooltip will display which two categories are selected, and the number of movies complying with the request over the total number of movies that include both of those categories, but not exclusively those two.

These two levels of interaction (click on genres on the legend, and hovering on arcs) follow Shneiderman's mantra **[4]**. The overview consists of the chord diagram as it is created, without any interaction from the user; the filter happens when the user selects one or more genres from the legend, and the zoom of those genres is a consequence of the user actions; finally, the details-on-demand are provided by the tooltip that appears only if the user hovers on an arc, giving extra information about those genres.

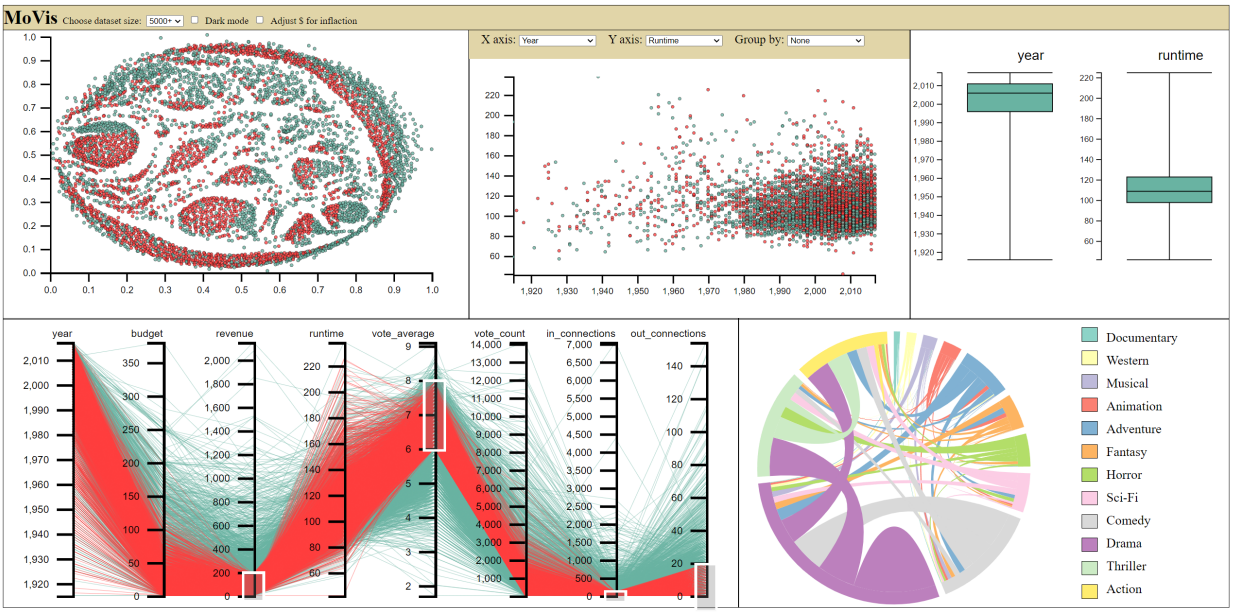
The Toolbar

The toolbar on the top of the page gives the user three options, as described in the previous chapters: it can choose to change the dataset dimension (250, 1000, or full size); it can switch to the dark mode; it can choose to the actual budget and revenues of movies, adjusting their values for the inflaction. This last input switches the two columns 'budget' and 'revenue' of the parallel coordinates with the alternative "actual_budget" and "actual_revenue".

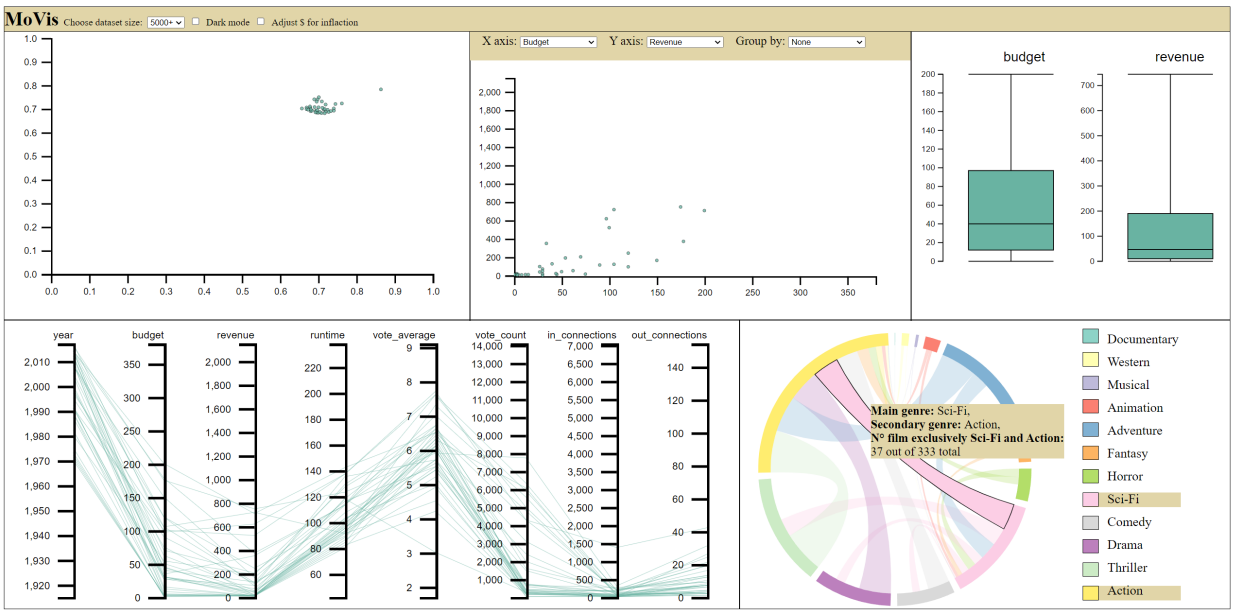
Inter-visualizations interactions and coordination

The user interactions with this visualizations are not limited to the single parts already described until now: as a matter of fact, many of them change more than one element at a time, and bring together the project by creating an interactive and responsive experience.

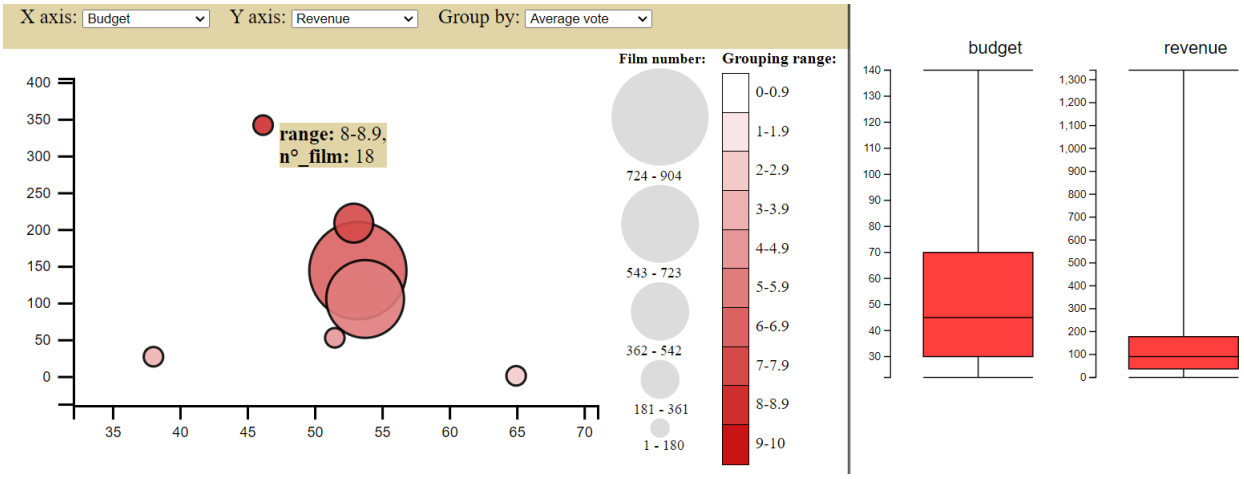
The following interactions modify more than one element at once:



- **brushing the parallel coordinates**: when brushing the parallel coordinates, aside from coloring the lines in red, we also color the corresponding circles in the MDS and bubbleplot (both circles and bubbles); also, the chord diagram and the boxplots will be recompute only for the brushed movies, and color them red to indicate it to the user.



- **selecting genres in the chord diagram**: when selecting genres, both from the legend or directly from the arcs of the chord diagram, the movies that do not belong to those genres are removed from all the visualization (MDS, bubbleplot and parallel coordinatesand from the chord diagram itself, which is updated). Again, the boxplots are updated to include only movies of the relevant genres.



- **changing bubbleplot axes**: when changing the attribute used on an axis, one of the two boxplots will change, to display the distribution of that feature;
- **selecting a bubble**: when clicking on a bubble, its color will change, to indicate that it has been selected, and the two boxplots will be updated and recomputed only with the selected data.

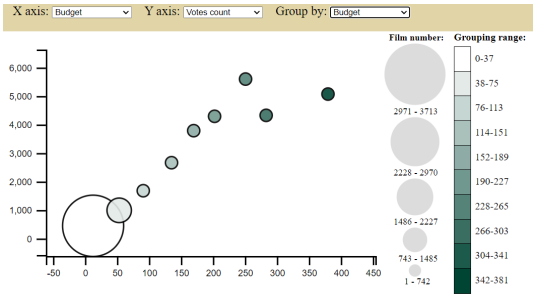
All of the previous interactions can be combined with each other, in any order.

Insights

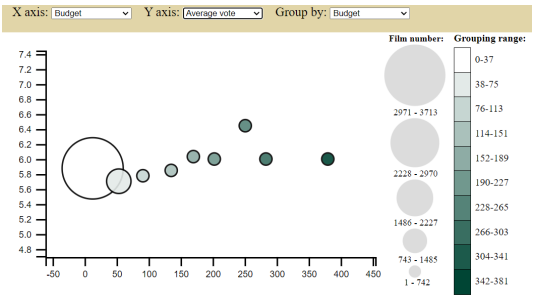
In this chapter we present only some of the insights that can be found using the MoVis visualization.

In the paper **[1]** the authors conclude that: *"[...] we find a strong correlation between number of user votes and the economic statistics, particularly budget. Remarkably, we find no evidence for a correlation between number of votes and average user rating. Our results suggest that total user votes is an indicator of a film's prominence or notability, which can be quantified by its promotional costs..."*

With our visualization, in particular with the bubbleplot, it is possible to confirm this thesis.



(a) X: budget, Y: votes count



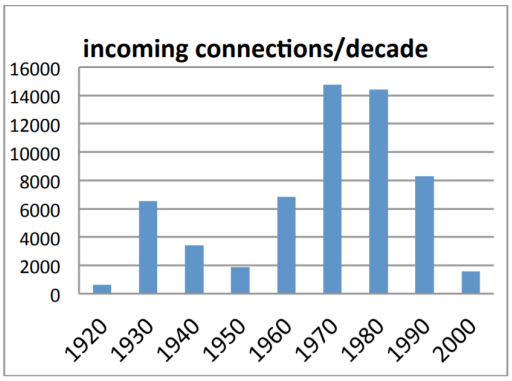
(b) X: budget, Y: average vote

It is clear that in the first graph there is a clear direct proportionality between the two feature, which is missing in the second graph. Thus, the thesis sustained in **[1]** can be visually confirmed.

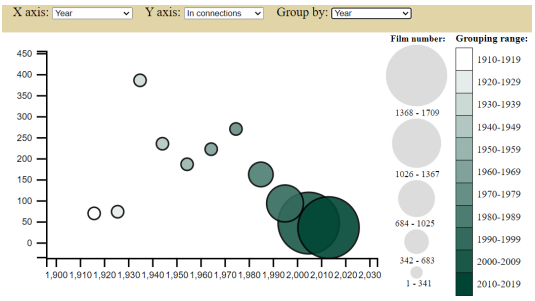
* * *

In the paper **[2]** graph *(b)* can be found, showing which decades where the ones receiving the greatest amount of incoming connections. The same shape can be found in our visualization when grouping by year and plotting incoming connections over years.

The slight differences in the vertical scale can be explained by the fact that in **[2]** it represents the sum of the connections, while in our case it is the average of the movies of that decade. Despite this, in both graphs there are peaks in the 1930s and 1970s, and lows in the 1950s and in the most recent decades. It makes sense: recent movies have had less occasions to be referenced.



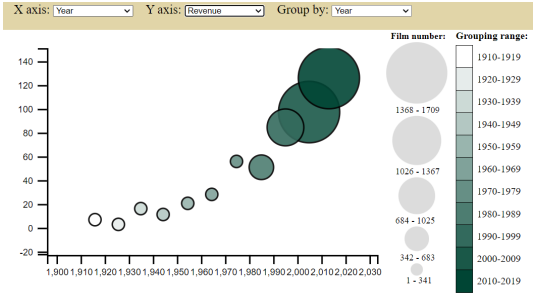
(a)



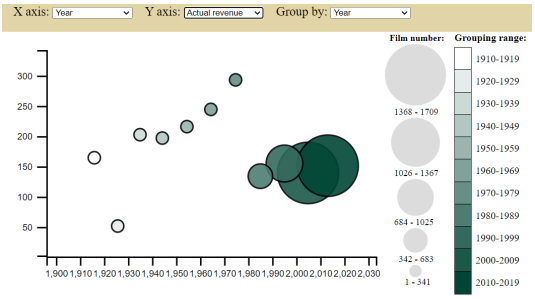
(b)

* * *

A visual confirmation of the correctness of adjusting for inflation can be found by looking at the difference between plotting the revenues (unadjusted) over the years (*fig. a*) versus the adjusted revenues (*fig. b*). The former shows an exponential growth of the feature, whereas the latter doesn't, confirming that the conclusion that movies nowadays earn a lot more than the past is in great part influenced by the effects of inflation, among other historic factors.



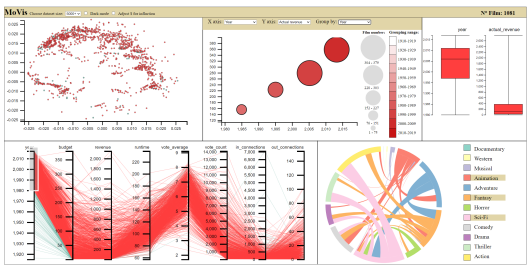
(a) X: year, Y: revenue



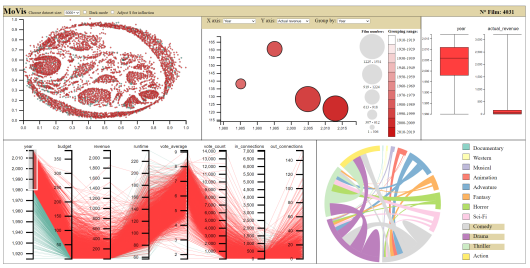
(b) X: year, Y: actual revenue

* * *

The bubbleplot of these two very different collection of genres shows opposite trends in revenues. The second group (*fig. b*) are movie genres whose quality is less dependent from the technological possibilities available in that years, and for this reason their earnings are quite stable (varying by no more than 20%). Instead, movie categories in *fig. a* are more associated with visual effects, so the technological advancements in the last four decades brought these movies to earn many times more than before.



(a) Animation, Fantasy and Sci-Fi



(b) Comedy, Drama and Thriller

Conclusions

The final visualization allows for an interactive visual analytic of the cinema industry history. We are satisfied with the final result. It allowed to answer most of the question we came up with during development. Every part of the visualization is useful to a specific need, and together they give a comprehensive and visually pleasing overview about an extensive catalog of movies.

Even if most of the data has been taken from already existing datasets, we compiled them together and re-elaborated them, in some cases resulting in new data not retrievable directly anywhere on the web. Most importantly, the visualization of these data in an interactive web page produces a more comprehensible, immediate and pleasing experience for the average user.

Future works

While working on the project we came up with many new ideas on the future development of the tool. In particular it would be interesting to:

- implement a search functionality that allows to type a movie title and highlight it in the visualization;
- a visualization that analyzes the relationships between directors and actors, in a graph structure (this would require a lot of effort but could be interesting);
- a visualization that focuses on the movies' languages and/or production countries, in this case with a geo-referenced visualization. This was considered but then discarded since more than 90% of the movies are in English language and are produced in the U.S.A.;
- implement user-selectable presets that change all the visualization to answer a specific need, acting as a sort of tutorial on the many different possible uses of the tool;
- a slider that allows to visualize single years, and maybe automatically scrolls through the timeline after some delay.

Bibliography

[1] Max Wasserman, Satyam Mukherjee, Konner Scott, Xiao Han T. Zeng, Filippo Radicchi and Luís A. N. Amaral.
Correlations Between User Voting Data, Budget, and BoxOffice for Films in the Internet Movie Database.
66(4):858–868, 2015.
URL: https://asistdl.onlinelibrary.wiley.com/doi/epdf/10.1002/asi.23213

[2] Canet, F., Valero, M.A. & Codina, L.
Quantitative approaches for evaluating the influence of films using the IMDb database.
Communication & Society 29(2), 151-172 (2016).
URL: https://www.semanticscholar.org/paper/Quantitative-approaches-for-evaluating-the-of-films-Canet/c747930515ee690a22b62e04cf19c0fa9c6e0d61

[3] Laia Blasco-Soplon, Josep Grau-Valldosera, Julia Minguillon
Visualization of Enrollment Data using Chord Diagrams
978-989-758-087-1 (2015).
URL: https://www.scitepress.org/papers/2015/53605/53605.pdf

[4] Ben Shneiderman
The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations
0-8186-7469-5 (1996).
URL: https://www.cs.umd.edu/ ben/papers/Shneiderman1996eyes.pdf

[5] https://colorbrewer2.org/

[6] https://learnui.design/tools/data-color-picker.html

[7] https://www.officialdata.org/us/inflation/1915?amount=1

[8] https://www.kaggle.com/rounakbanik/the-movies-dataset?select=movies_metadata.csv

[9] https://grouplens.org/datasets/movielens/

[10] D3.js documentation
URL: https://github.com/d3/d3/wiki
URL: https://observablehq.com/@d3

[11] M. Angelini, G. Santucci
Material of Visual analytics course, 2021/2022