VISUALIZING MOVIE METADATA

Making sense of and exploring movie metadata information

Process Book

David Lu

05-499: Visualization in HCI, Spring 2017

Carnegie Mellon University

Table of Contents

Project Plan

Basic Information

Project Name:	Movie Metadata Visualizer
Team:	David Lu davidylu@cmu.edu
Project URL:	movie-vis.davidylu.com

Overview and Motivation

There are a lot of movies.

There exists a lot of data associated with each movie.

I thought it would be cool to have some kind of visualization to tie together all the metadata from a set of movies. Someone could use this tool to explore the relationships between different movies, as well as see how different genres of movies stack up against each other, a sort of "big picture view" of the dataset as a whole.

It would also help with finding a new movie to watch: if given a number of various preferences or parameters, this tool could help narrow down the list to a manageable number of movies, from which it should prove much easier to choose a specific one.

Related Work

FlowingData has a collection of movie-inspired visualizations, available here:
https://flowingdata.com/tag/movies/

There are also a couple websites that go through data movie-by-movie, such as these:
http://www.informationisbeautiful.net/visualizations/based-on-a-true-true-story/
http://cinemetrics.fredericbrodbeck.de/

Many of the projects I've found compare movies on an extremely detailed level, for example their color profile over the entire course of a movie, or in the "information is beautiful" example, truthfulness of a movie by section. When movies are compared on a metadata level they are often static charts or graphs that, while beautiful, allow only for the interpretation that the author was going for.

I intend to take this one step "further" by taking one step back – the data I will be examining is more general than most of these visualizations (metadata, per movie, instead of by minute of film), but may allow the viewer more flexibility of what they want to see. This will be less of me telling a story with data, but allowing users to explore the data themselves.

Questions

As noted before, I did not have a specific story to tell upfront – rather, I wanted to see if the data had some kind of story through its data already that just needed to be 'discovered.' Because of this, the questions I had were tied in closely with my data exploration.

Initially, I wanted to see if various genres had different distributions of ratings across their respective sets of movies. However, I discovered soon that most movies follow the same pattern (see design evolution) of a sort of bell-curve distribution, making that question, while answered, quite mundane (read: boring) as a visualization.

Because of this, I refocused my efforts on designing something where the users could ask their own questions – if they wanted to see data from movies of a particular genre, for example, I should design something where they could select whichever genre they wished, intuitively, instead of having a few preset options that I personally selected to visualize.

The questions I was tackling evolved into "could this be a useful metric to use, either as a visualization or a selector?", or "Which attributes could be both visualized and used as a filter of sorts?"

Data

I had two primary sources for data: IMDb and MovieLens.
Both datasets are freely available for non-commercial use.

IMDb provides its information in a raw data format, available here:
http://www.imdb.com/interfaces/

MovieLens provides its data across 20 million ratings and tags, available here:
https://grouplens.org/datasets/movielens/

Gathering Data

I cross-referenced movies between both datasets; IMDb provides solid metadata information, however MovieLens provides user-specific data as well as individual ratings (and timestamps of said ratings). Because IMDb contains a lot of data on not just movies (TV shows and series), I used the MovieLens database as a base, augmenting it with data from IMDb. This allowed me to have a manageable dataset without having to worry about filtering out non-movie data.

There is still a lot of information.

Feeling adventurous, I decided to load the original MovieLens dataset directly, using Javascript. Two minutes of waiting later and significantly less adventurous, I abandoned the concept and set about designing how to pre-process this data first.

As a result, part of my project is in Java, as I had to pre-process the information to make it easier for a browser to handle within a reasonable load time. I had another, separate Java program handling scraping and parsing web data from IMDb, based off the MovieLens list.

Javascript and d3 interact with only these pre-processed datasets, enabling me to have both the data precision I need for my visualizations as well as reasonably fast load times for a web-based tool. The makeshift "database" and its data doesn't update unless I re-run my backend Java code again, but that was a reasonable sacrifice for performance and loading times.

There were a couple edge cases I needed to clean up (such as certain movies that used to be in the IMDb database, but for undocumented reasons, have been deleted), however the majority of movies did allow me to reliably link the two data sources together.

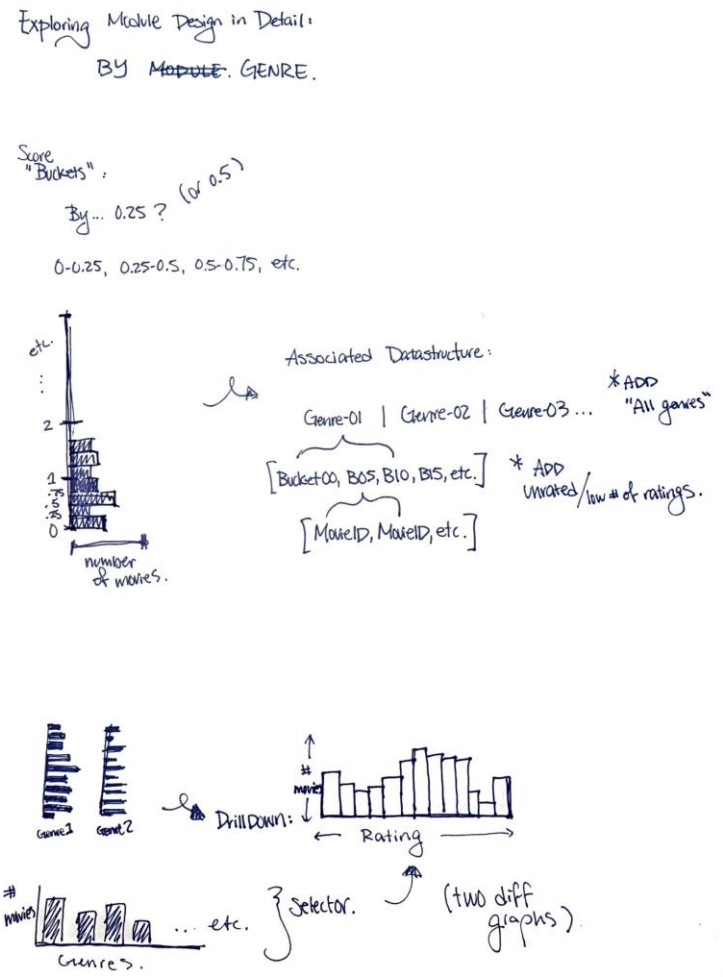
I also had to handle cases where IMDb had deleted a previously-existing entry.

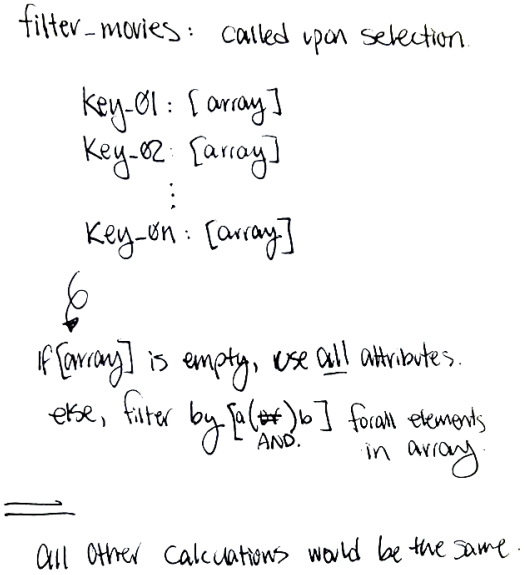
On the next page I have a few images from my backend data structure brainstorming. (How this data was to be represented so d3 could easily handle it)

Representing Data

In order to represent data, I needed to have a good idea of how I wanted the data to be graphed out, in order to ensure an easy way to recall the data with minimal transformations to the data. This is to help with performance, as web dashboard-like applications are generally frustrating to use if the user is made to wait long periods of time.

The image of the right depicts one of my brainstorm sessions, an early design of my "by genre" module. It features a three-tiered array, to make my original genre module easy to retrieve. This original genre is shown later on in the book.





I decided to go with something akin to `filter_movies` (original concept on the right), which was designed to be used as a secondary datastructure. This allows for dynamic charts and graphs that let the user filter / drilldown to more specific data. (e.g. view data only from movies that are labeled as the 'Adventure' genre.)

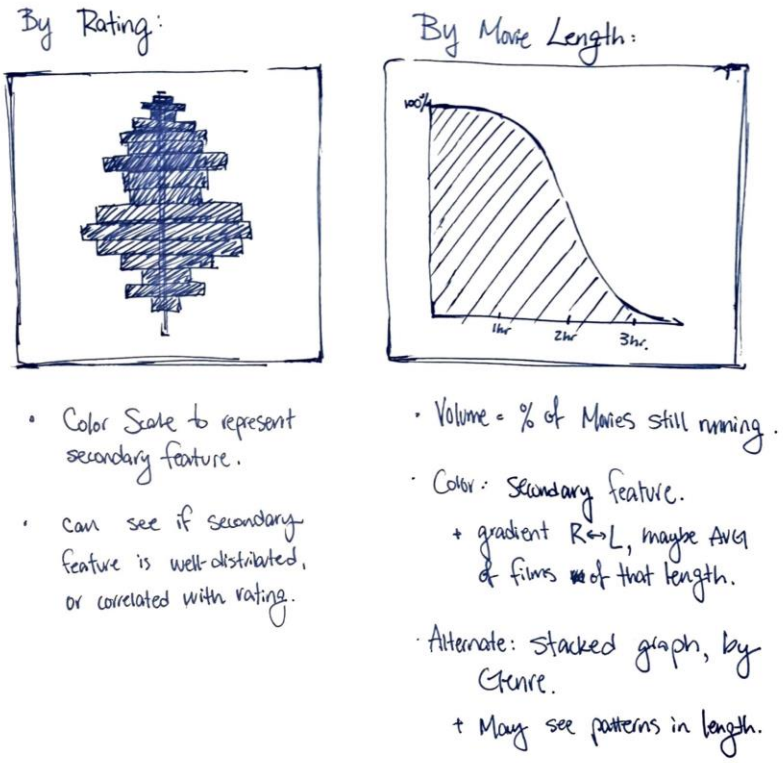
In addition, since this structure is global, it easily allows updates to filters across different charts.

Exploratory Data Analysis

There is a lot of data.

Much of this data however, is either "obvious" or "irrelevant." For example, we already assume that not all movies from any established genre would be rated consistently below 20%, as if this were the case, if nobody wished to watch these types of movies, it would likely cease being a genre to produce.

However, as my project aims to help construct a tool that helps users explore this data themselves, I do not need to find a heart-wrenching narrative for any subset of this industry. I simply need to allow the user to explore this data in a meaningful way.



Certain ideas, such as 'by rating,' were interesting designs on paper but when implemented, showed that most movies' average ratings hovered around the 70% mark, across all genres.

This makes for a relatively boring and static chart to view, taking up precious screen space better dedicated to other information.

Thus my "exploratory data analysis" revolves less around which data points are interesting, and more around which data attributes are (potentially) interesting in combination with other ones. This starts to tie in with design evolution, as different attributes means for potentially different charts or at least different axes.

Design Evolution

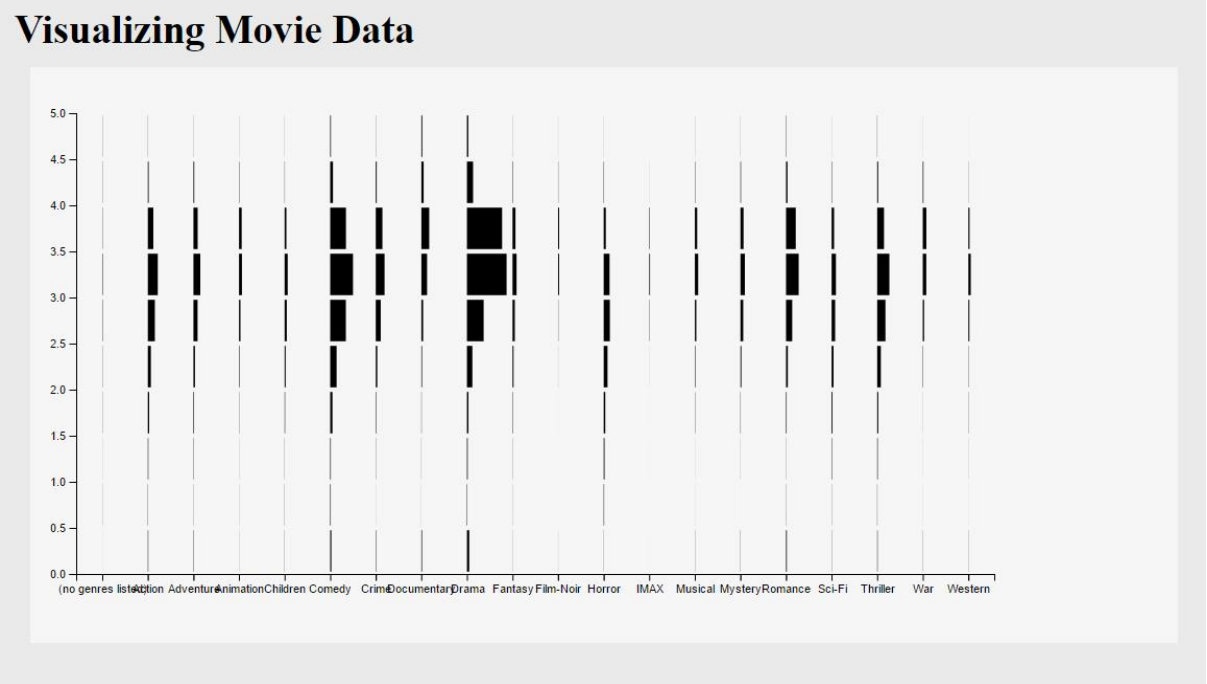
Individual Module – By Genre

My general idea was to design various modules, which I can then combine to create an overall visualization. For example, one module may display ratings over time, while another one may separate data-points into their respective genres.

Since these data entries should be obtained from a single source, it should be easy to link the data-points between different modules if applicable, providing an easier exploration experience for the user.

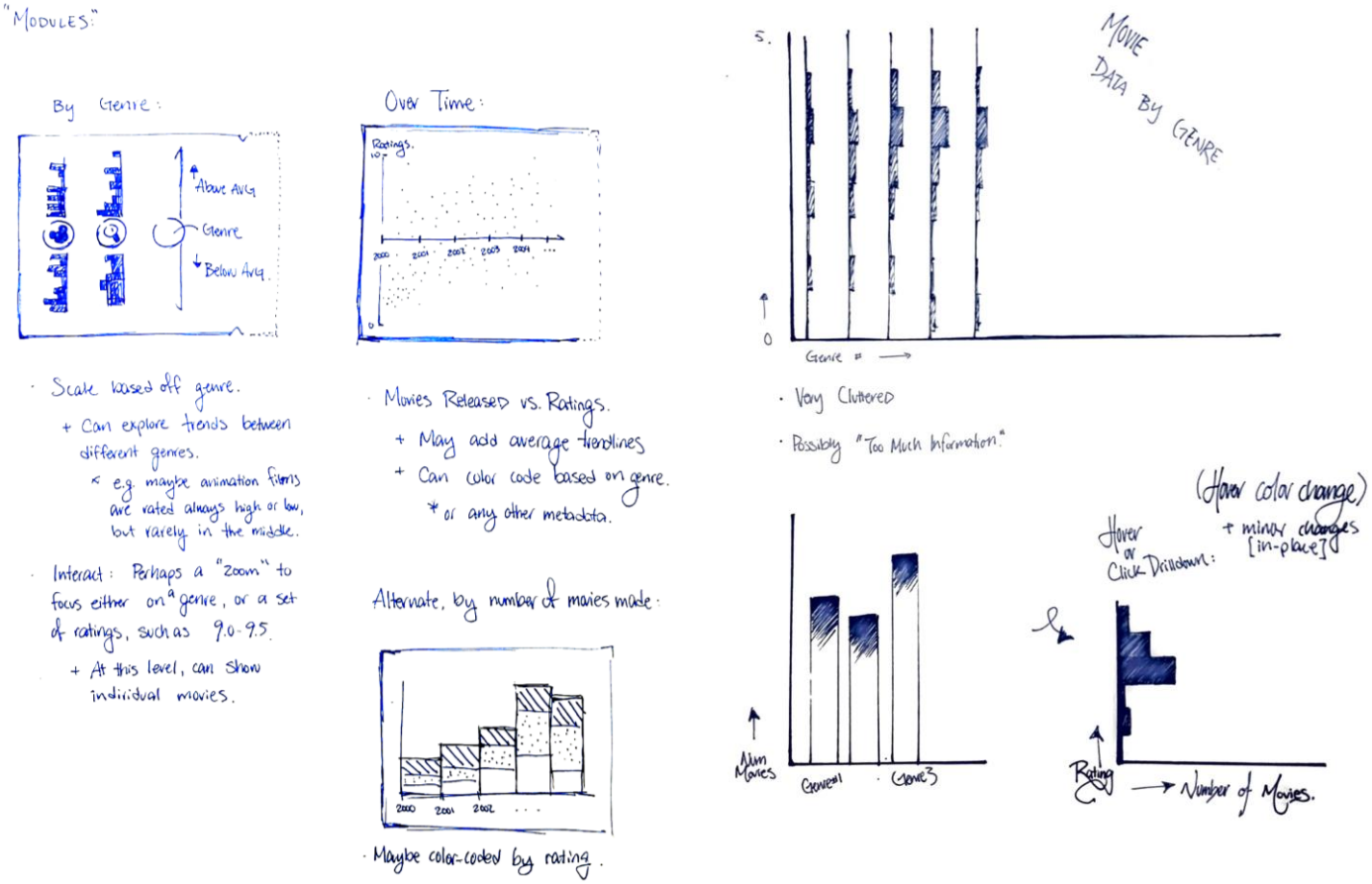
One of the first modules I designed was visualizing data by genre, separated by rating.

A screenshot of my first implementation of this module is shown below.



It was painfully obvious that this was not only a cluttered graph, but also one that did not necessarily convey very much interesting information, despite its information density. Most movies followed a bell-curve for distribution of movies, landing on average around 3-3.5 on a 5 point scale, roughly 60-70%. Based off this visualization, after a (relatively long) period of time, one can conclude that there is roughly no difference between genres.

I decided to redesign this chart, focusing more-so on the individual genres as opposed to the absolute number of ratings each movie genre fell into – this way, I could separate the rating data later into a different chart, making everything easier to read.



The image on the top left shows the original design idea, as well as possible "improvements" I could make to the idea (if I were to still keep that many number of elements in one chart)

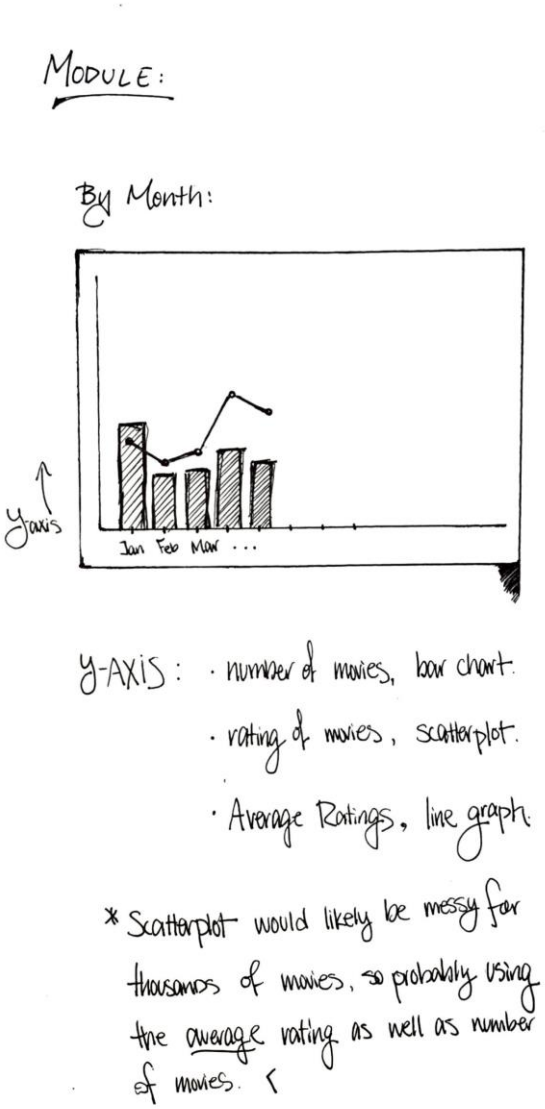
However, I ultimately decided on a redesign making the overall graph much simpler, and moving rating data to a separate graph altogether. This is seen in the second brainstorm, on the right. This made my initial graph much simpler and easy to process.

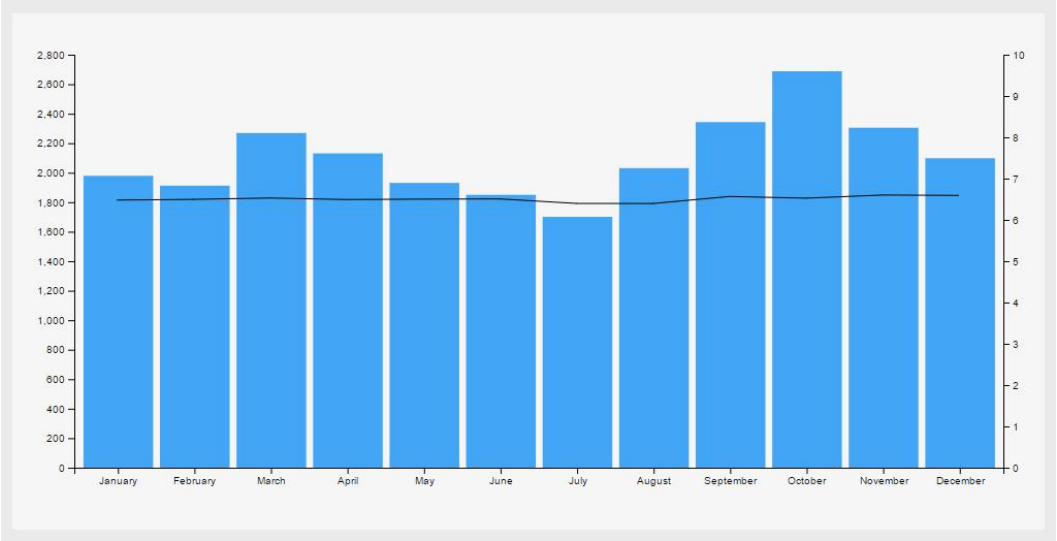
Individual Module – By Month.

The original brainstorm for 'by month' is on the right – it featured two y axes, to display both number of movies as well as their average rating each month. Average rating was represented using a line graph, while number of movies was featured using a bar graph.

The first implemented version is featured below – the result was lackluster, to say the least. Most movies averaged between the 60-70% marks for all months, with little variation between different months.

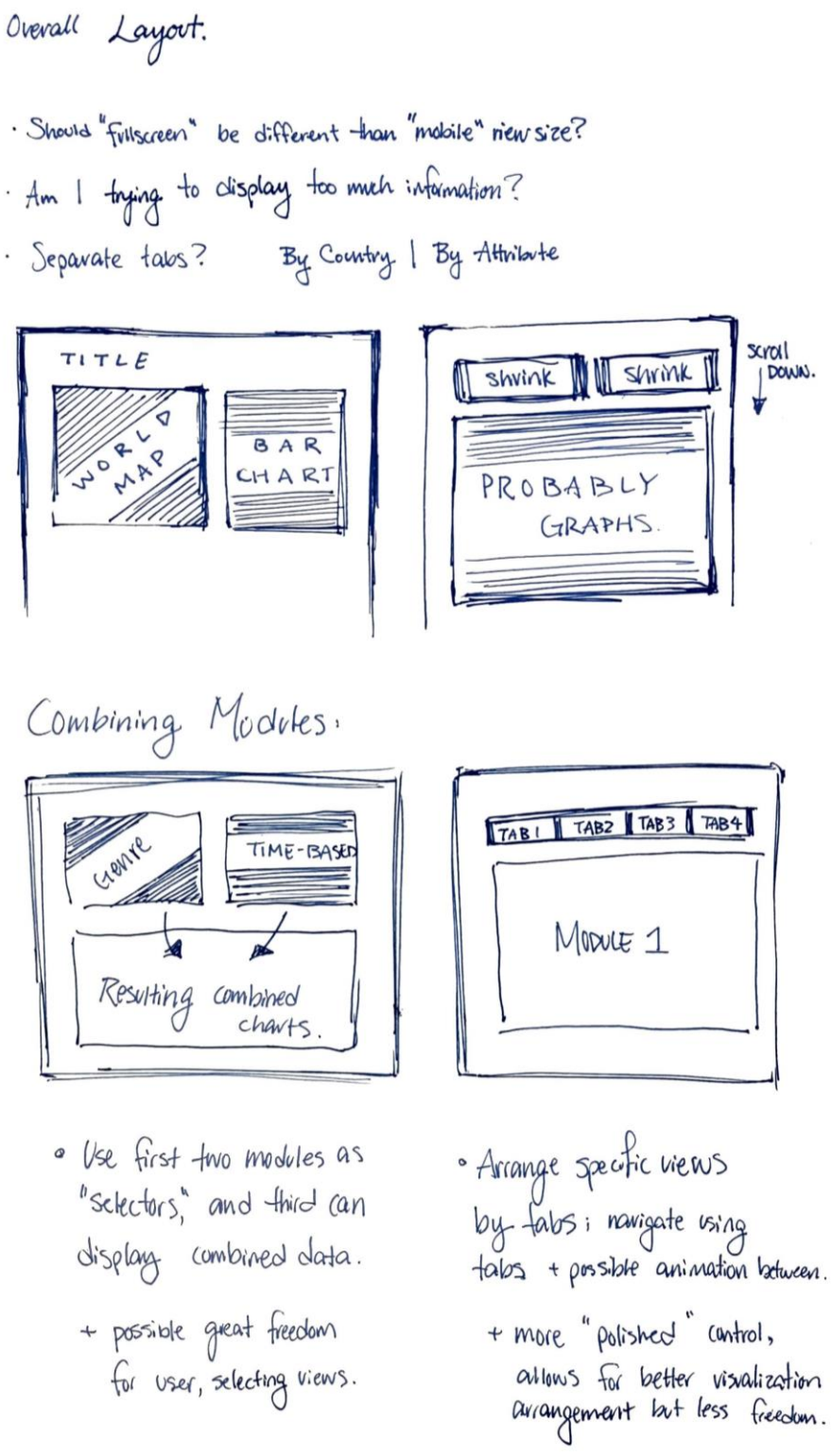
In addition, the line made very little sense, design-wise, as this was not data that was continuous over time. Because of these considerations, I ultimately removed the line graph for ratings in the final design.





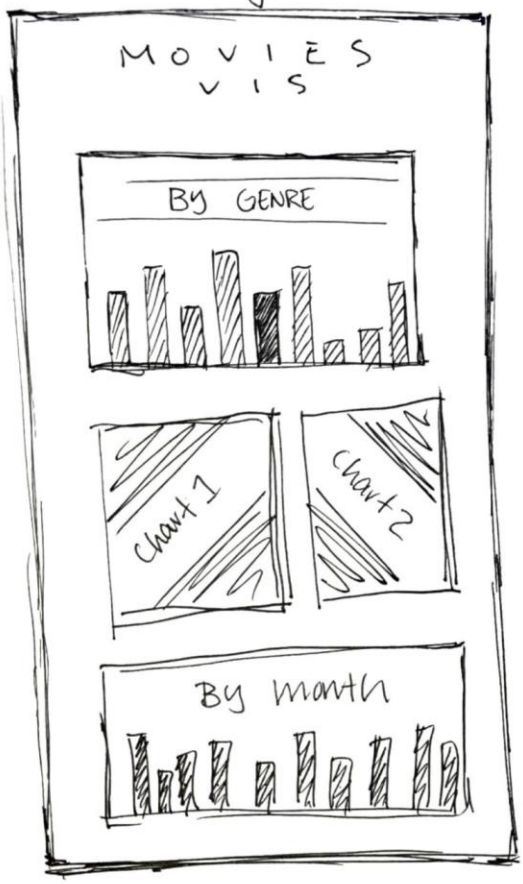
Overall Layout Design

Images below refer to designing overall layouts. The most basic decision revolved around a portrait vs. landscape layouts – I ended up going for a single-page, landscape layout.

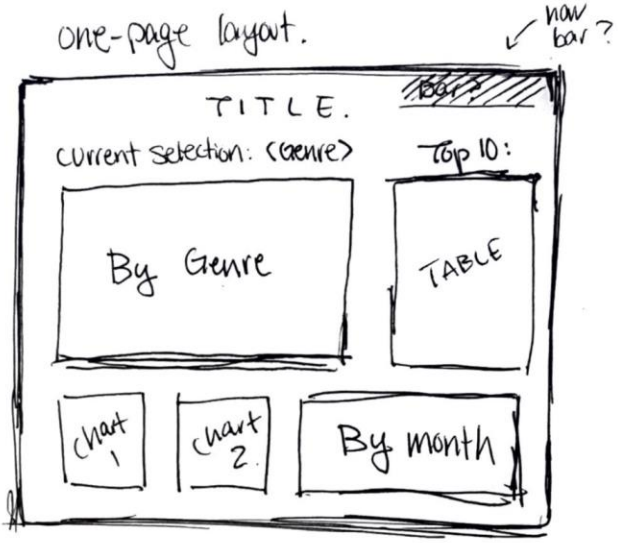


LAYOUT Sketches.

Scroll-based layout.



one-page layout.



Single-Screen Layout.

- Cons: . may be difficult to adapt to different screen sizes.

* Due to modular implementation, different screen sizes can be adapted / flexible layout.

Cons: · Selection not intuitive?

· not all on one page.

· Add a "top 10" type of (only chart 1) <table> to maybe a side panel

· expands Down but screens (↕) are typically wide. (↔)

current selection: ☒ Action

↑ click to "unselect".

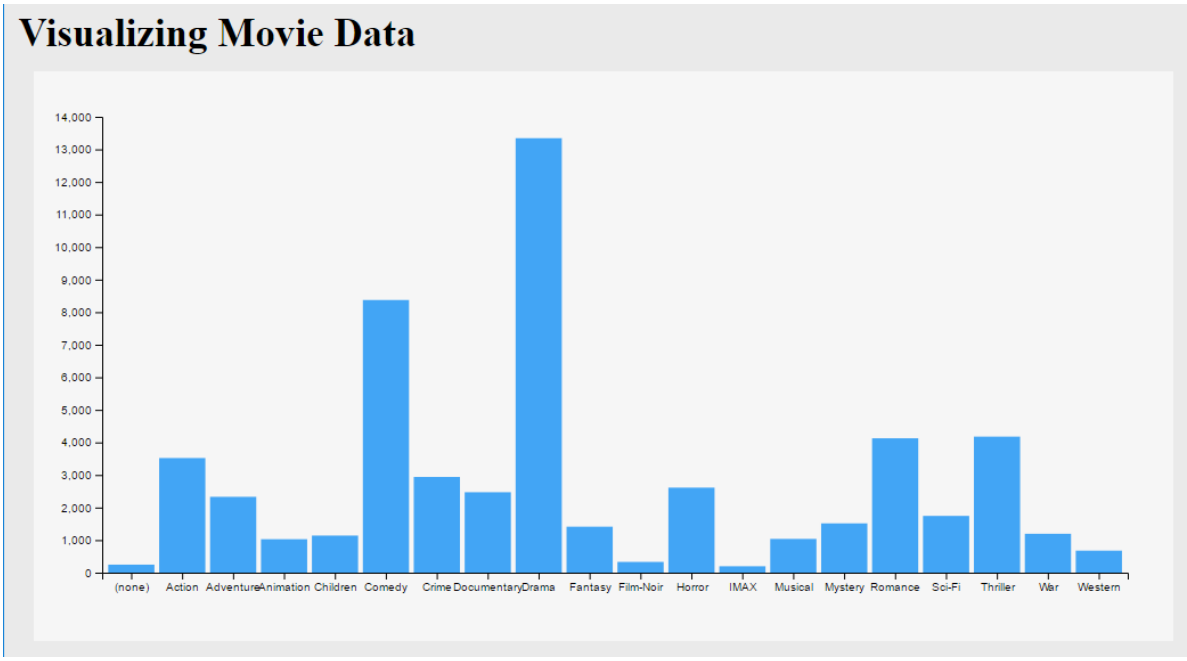
movies, filtered movies.

↑ allows dynamic changes.

Implementation

I do heavy preprocessing in Java, leaving little computation that needs to done on frontend with Javascript, aside from organizing the data into its relevant datatypes for d3.

A screenshot of the resulting first module (number of movies organized by genre) has been implemented as follows:



The chart will be interactive – while not terribly exciting by itself, it will serve as two features – the main one being a chart, however if the user were to click on any genre it will filter all other charts by that genre.

For example, clicking 'Action' will filter all the other charts (to be implemented) by the Action genre, showing results for only that specific subset of data.

This should allow for a great amount of freedom on the part of the user, without heavily influencing how the charts would react under different variables (limiting the amount of edge cases I have to account for in the coding process)

Evaluation

Personally, I'm quite pleased of how my visualization turned out. It offers most of the basic features I wanted to implement, namely interactivity between charts showing metadata information for all the movies (or subset of) in the database.

Improvements

There are a few things I wish I did better however, being overall layout (and how good the website looks aesthetically) and somehow squeezing in more parameters. In addition, its current state is only designed to be functioning properly in 1920x1080: sufficient for this project, but definitely not ideal.

I think the color scheme, while safe, could use some innovation – the entire dashboard looks very plain and one-dimensional as it is right now. There is a little variation for hover or selection, but everything is the same shade of light blue and/or lighter blue, with a dash of violet.

The fonts can be optimized more for ease of viewing, and I need to find some way of explaining what each chart represents, be that a discrete title or hover-help-text. On that note of explanations, the 'filtered by' buttons should have more of an indicator that they are clickable for de-selection, making the entire dashboard more usable without me being there to explain every feature.

As noted during my presentation, I may need to tweak my movies rating chart – sometimes the number of people who have voted on a movie may be a good indicator of how popular it is. While I did filter out movies with extremely-low number of votes, I may need to work more on the sorting of which movies get to be in the chart and which do not.

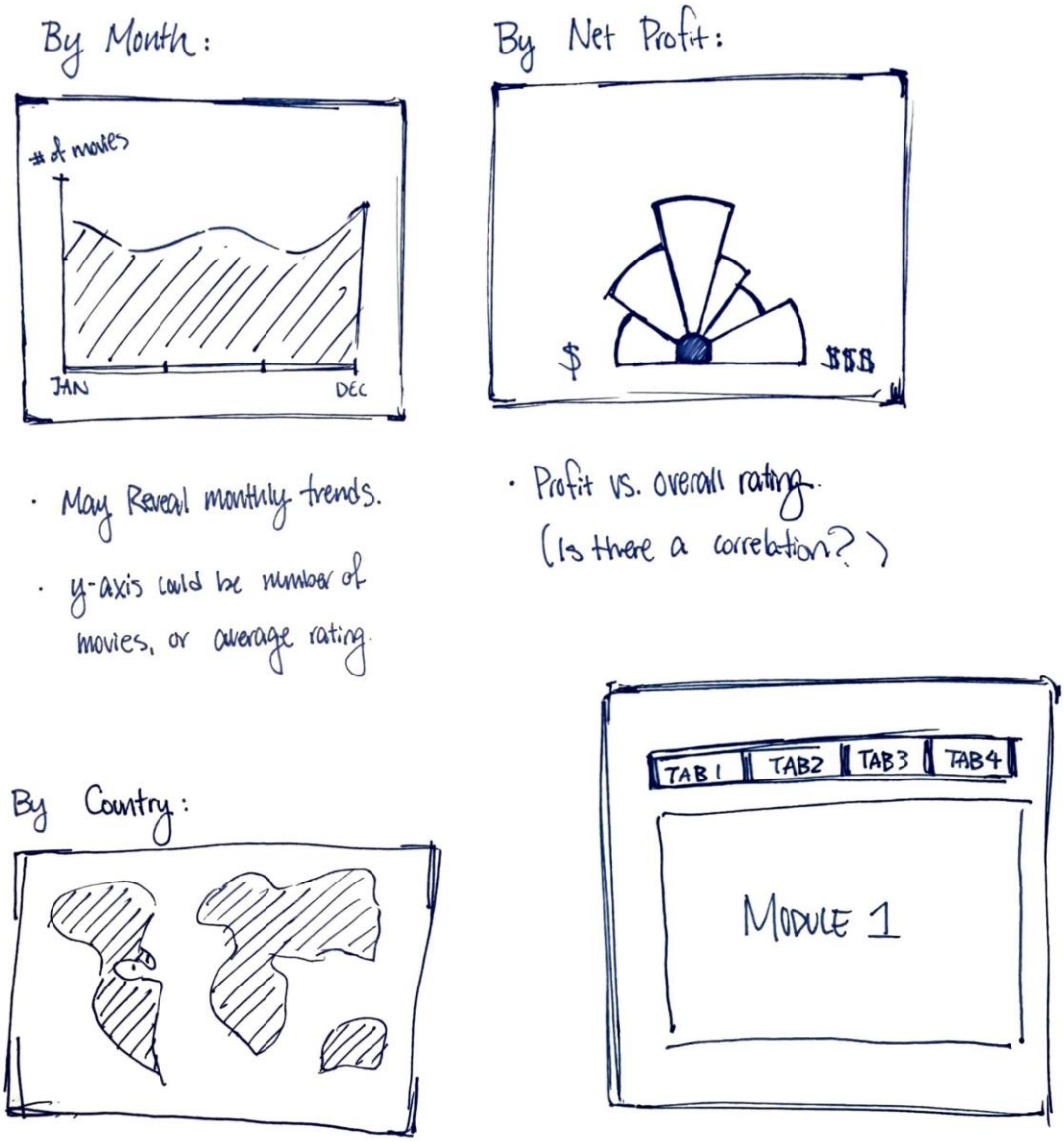
Accomplishments

I am really proud of the level of responsiveness I'm able to provide, especially with this amount of data and the level of interactivity present in the dashboard. I especially like the 'filtered by' button idea as well as the html table; I think these were implemented very close to what I had in mind and provide very seamless functionality.

I'm also very satisfied with the little backend I've built up. The only library I've used is d3, and I've formed a little system where it became relatively easy to add in a new module, once I've decided where it's going. With this kind of modular system, while the first module took a while to plan and design and fit in, the other ones that followed were significantly faster, and it makes it very easy to extend and maintain in the future. I do plan on extending this project, so this should be an important aspect moving forward.

Future Plans

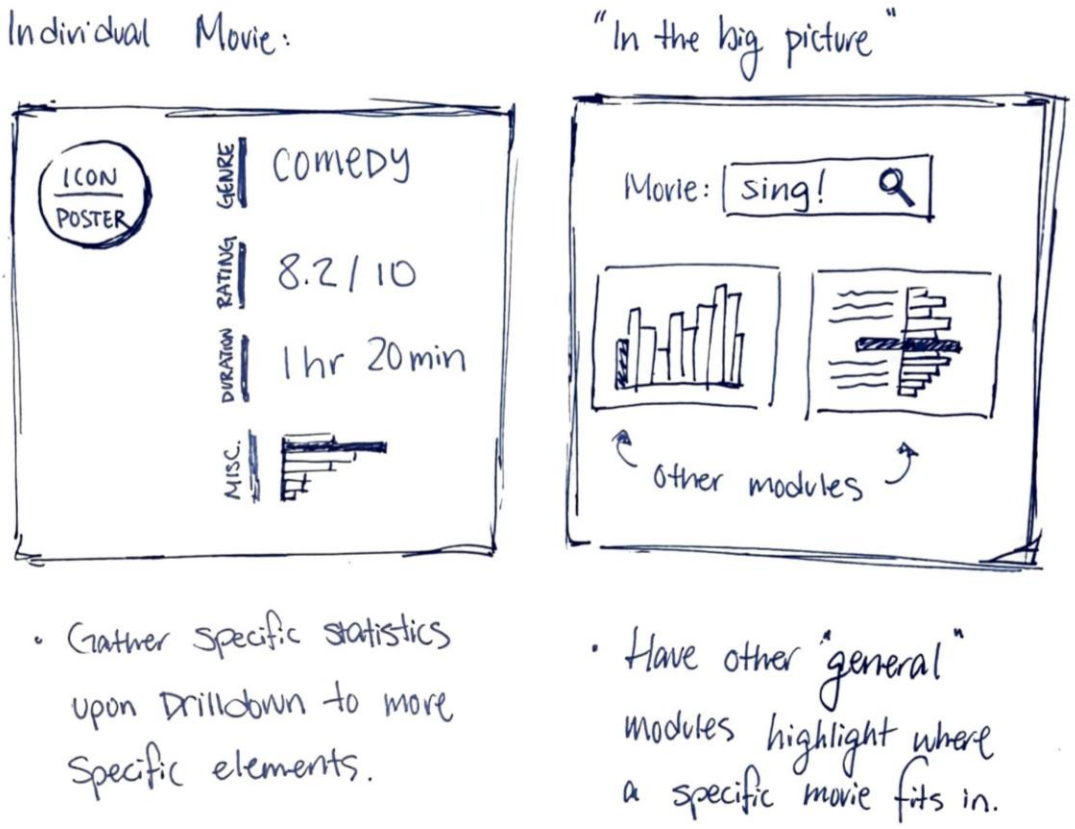
I have a couple module ideas I had been exploring which didn't make it into the final visualization. Here's a few of the main ones (a form of the 'by month' one did make it):



I plan on exploring some of these, as well as different chart ideas (global location-based graphs, or that weird burst circle for net profit) for these or existing modules. In addition, I want to experiment with different ways to try to fit everything still into one dashboard. Some of the ideas I had been exploring to this end included zoom on hover for selection, and tabbed pages.

Each however, have their own drawbacks – zoom on hover would make charts potentially too small to view comfortably while not zoomed, and tabbed views don't allow the user to see an overall big picture view. I'll have to experiment more to find a happy compromise.

One of the biggest "features" I wish I had to implement this time around however, was individual movie pages/popups. I have included an initial concept design below:



These could be either separate pages, or in a mini popup-type box when selecting a movie, compared to sending the user directly to the associated IMDb page. This helps create a space where I can visualize how this movie in particular does compared to other movies, either all or perhaps in the same categories/genres as it. This also can be extended to allow users to search up a specific movie of their choice to see how it stacks up against the competition.

Overall, I think the current dashboard is functional, but not pretty or elegant to use – it's responsive, but feels like some corporate dashboard using very standard charts. This presents the data in a very mundane way that may not seem very exciting to use.