Basic Information

Project Title
Exploring Movies (tentative)

Name
David Lu

Email
dlu1@andrew.cmu.edu

Project Repository
https://github.com/dylu/movie-vis

Overview and Motivation

There are a lot of movies.

There is also a lot of data associated with each movie.

I thought it would be cool to have some kind of visualization to tie together all these aspects of 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.

Questions

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

Data

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

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

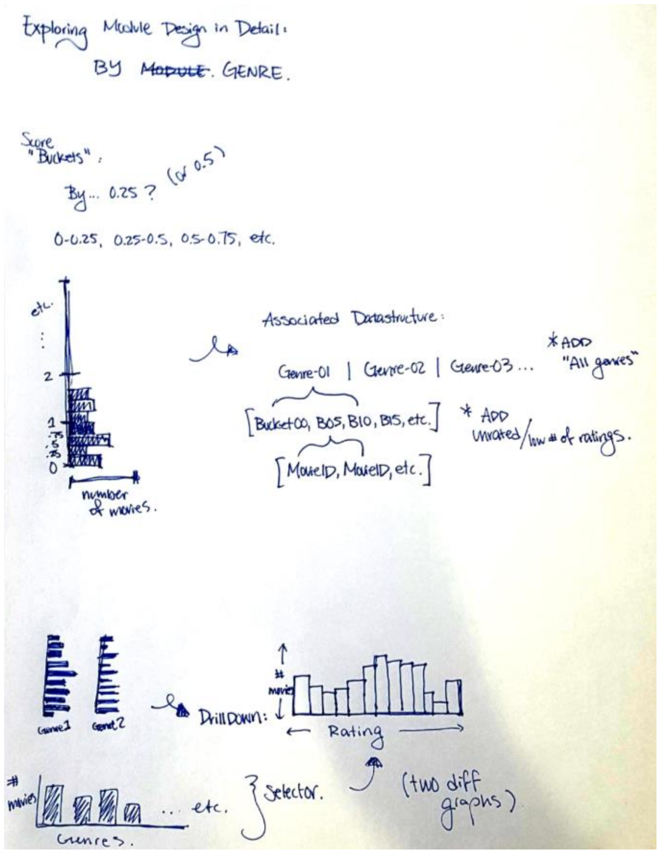
I cross-referenced movies between both datasets; IMDb provides solid metadata information, however MovieLens provides user-specific data as well as individual ratings (and timestamps of said ratings). Because IMDb contains a lot of data on not just movies (TV shows and series), I used the MovieLens database as a base, augmenting it with data from IMDb.

There is a lot of information.

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. Javascript and d3 interact with only my 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.

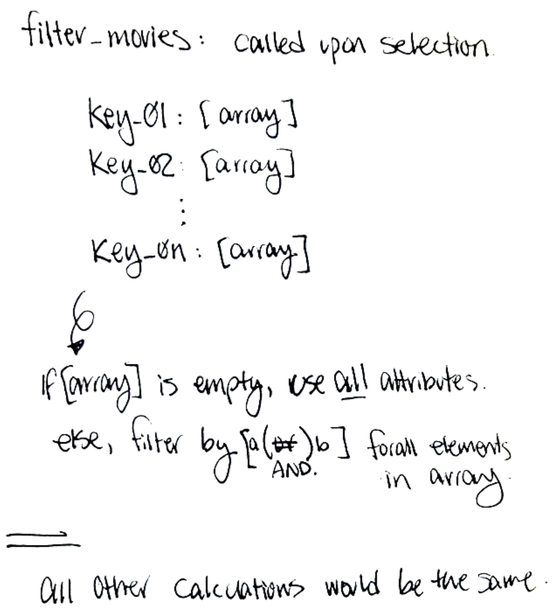
There were a couple edge cases I needed to clean up (such as certain movies that used to be in the IMDb database, but for undocumented reasons, have been deleted), however the majority of movies did allow me to reliably link the two data sources together.

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)



Designing how a specific module would represent its datastructure for ease of access for d3.

This specific example was for my original 'genre' graph, which I'll have the result for in my 'design evolution' section.



filter_movies: called upon selection

Key-01 : [array]
Key-02 : [array]
:
Key-0n : [array]

If [array] is empty, use all attributes.
else, filter by [a (~~or~~ AND) b] forall elements in array.

all other calculations would be the same.

`filter_movies` was designed to be used as a secondary datastructure, to allow 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.)

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.

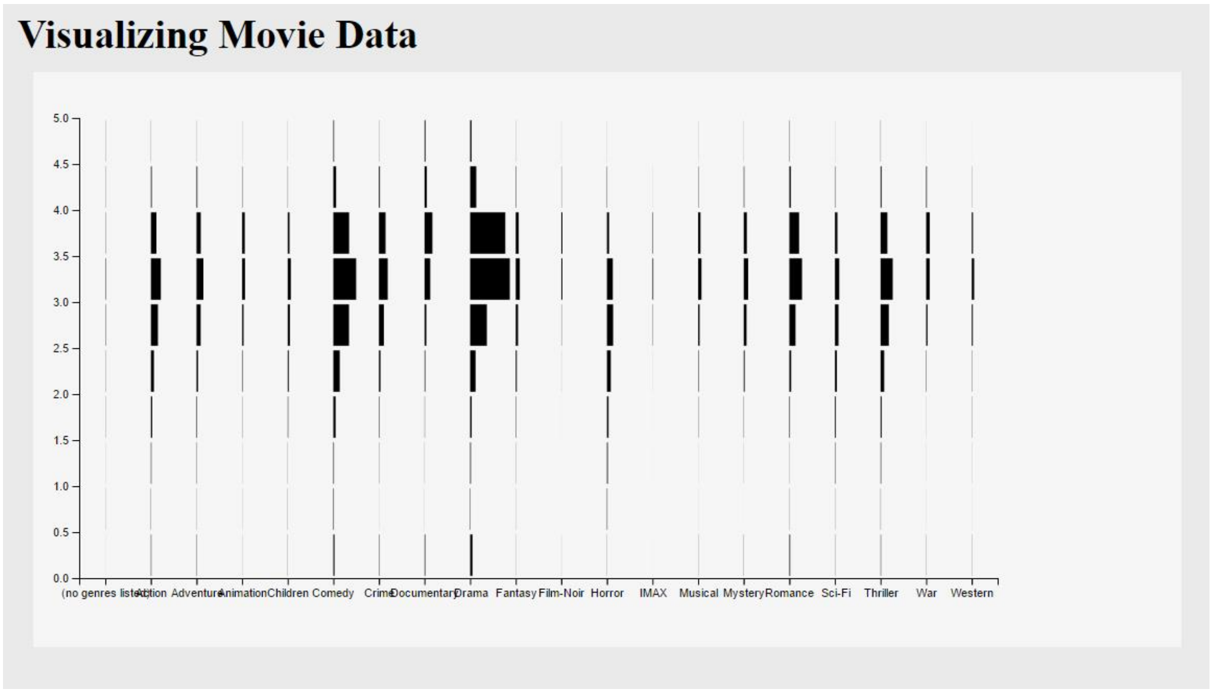
Thus my "exploratory data analysis" revolves less around which data points are interesting, and more around which data attributes are (potentially) interesting in combination with other ones. This starts to tie in with design evolution, as different attributes means for potentially different charts or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

Design Evolution

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.

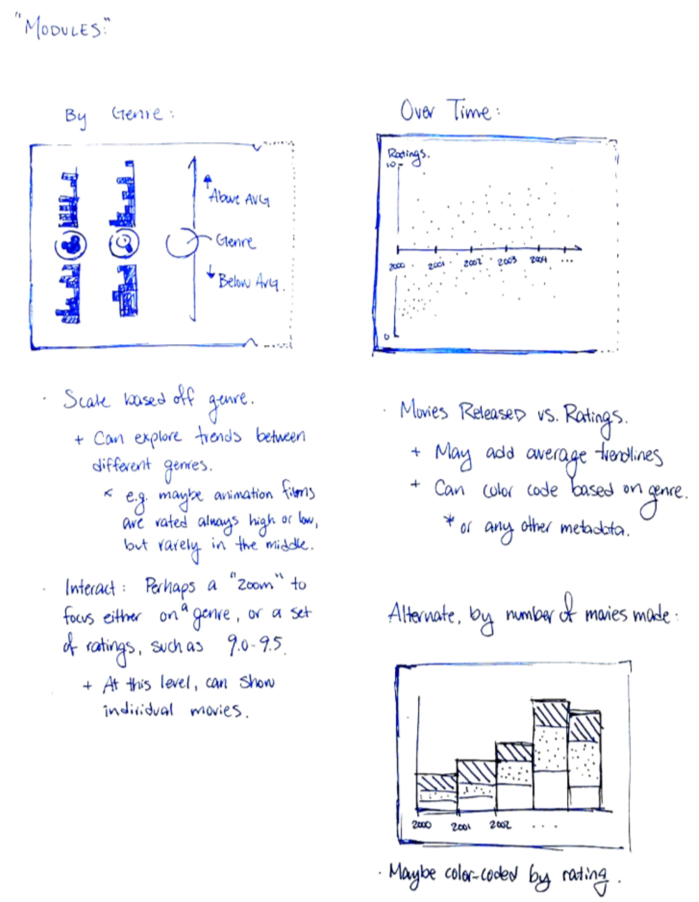
Since these data entries should be obtained from a single source, it should be easy to link the data-points between different modules if applicable, providing an easier exploration experience for the user.

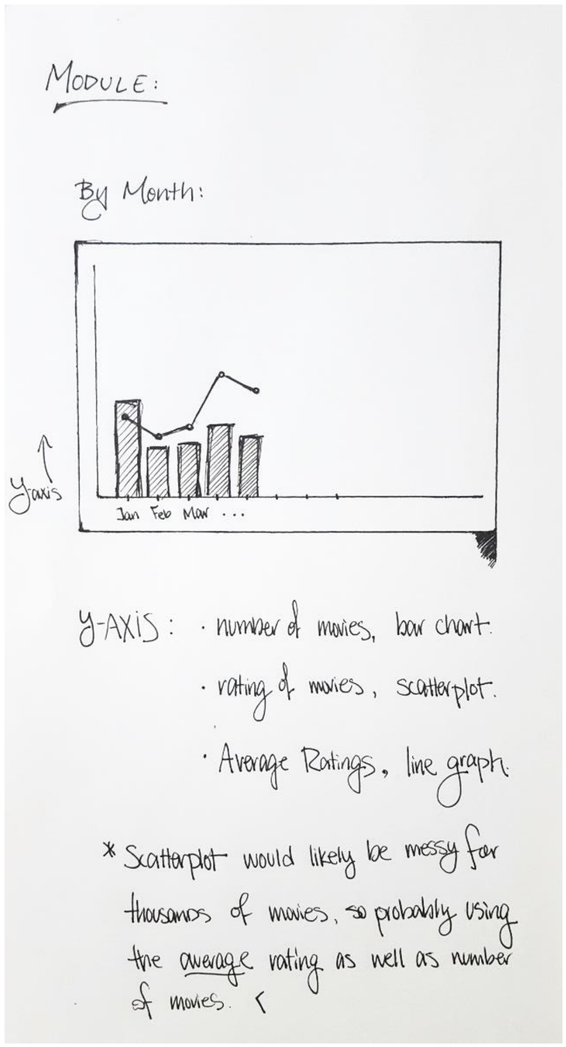
A screenshot of my first implementation of a module is shown below.



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.


"Modules:"
By Genre:
Above Avg
Genre
Below Avg
· Scale based off genre.
+ Can explore trends between different genres.
* e.g. maybe animation films are rated always high or low, but rarely in the middle.
· Interact: Perhaps a "zoom" to focus either on a genre, or a set of ratings, such as 9.0-9.5
+ At this level, can show individual movies.
Over Time:
Ratings.
10
0
2000 2001 2002 2003 2004 ...
· Movies Released vs. Ratings.
+ May add average trendlines
+ Can color code based on genre.
* or any other metadata.
Alternate, by number of movies made:
2000 2001 2002 ...
· Maybe color-coded by rating.


Module:
By Month:
y axis
Jan Feb Mar ...
y-AXIS: · number of movies, bar chart.
· rating of movies, scatterplot.
· Average Ratings, line graph.
* Scatterplot would likely be messy for thousands of movies, so probably using the average rating as well as number of movies.

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)

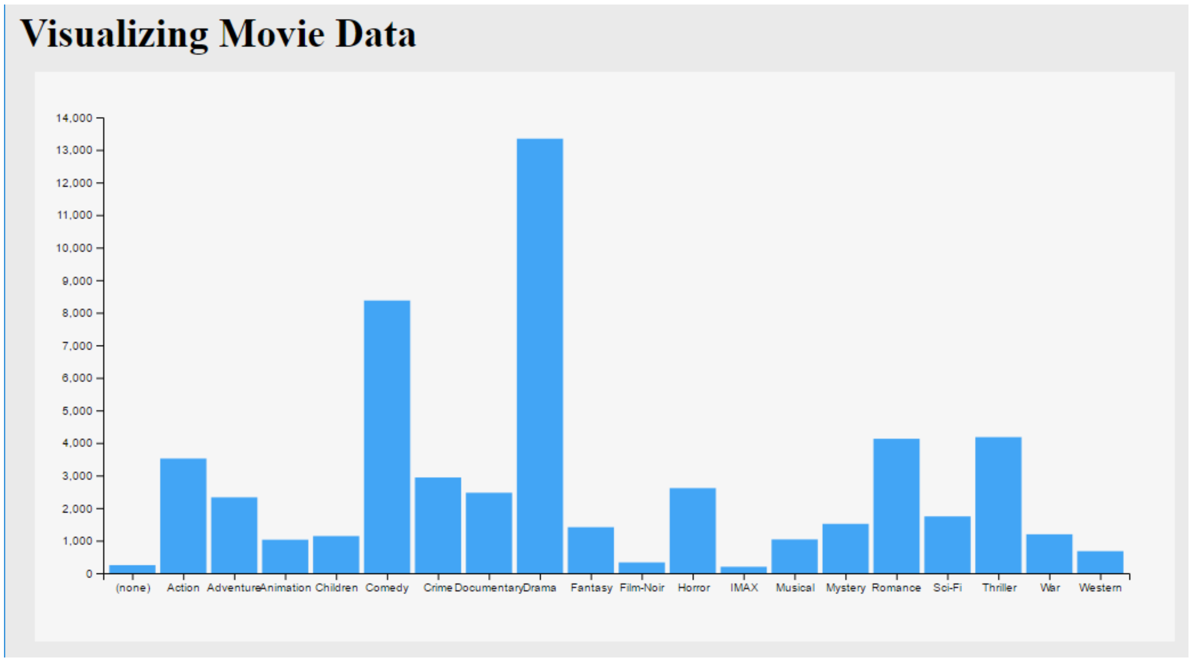
However, I ultimately decided on a redesign making the overall graph much simpler, and moving rating data to a separate graph altogether (new graph idea above, on the right)

This made my initial graph much simpler, as shown in the next section:

Implementation

I do heavy preprocessing in Java, leaving little computation needing 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

TODO. It may be difficult to complete this section before finishing the actual project itself.