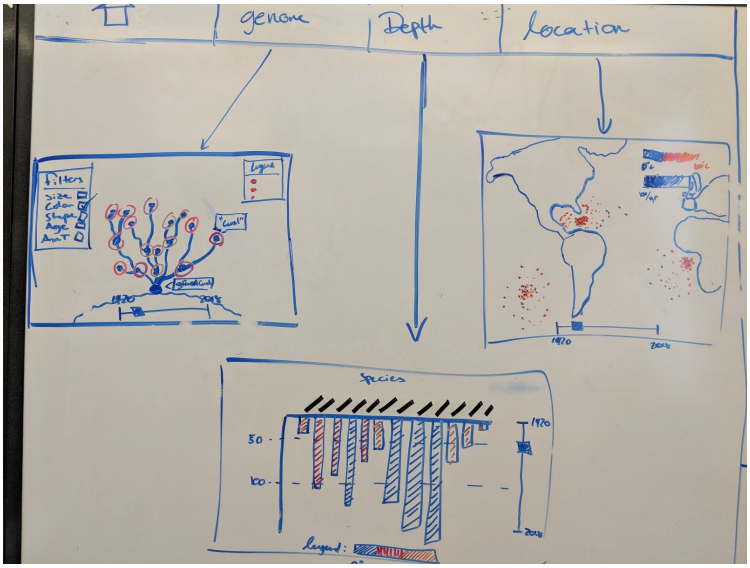
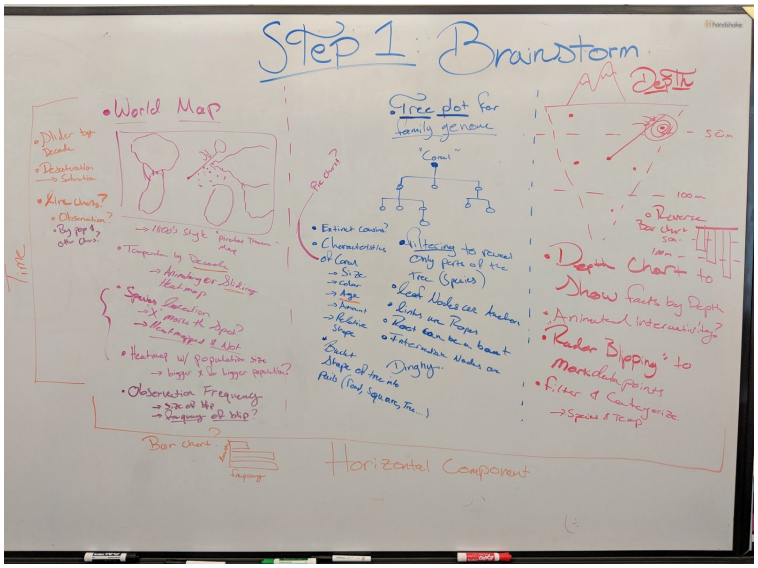
Dylan Fontana
Kenedi Heather
Aura Velarde

The Making of Movie Night.

Initial Project Proposal

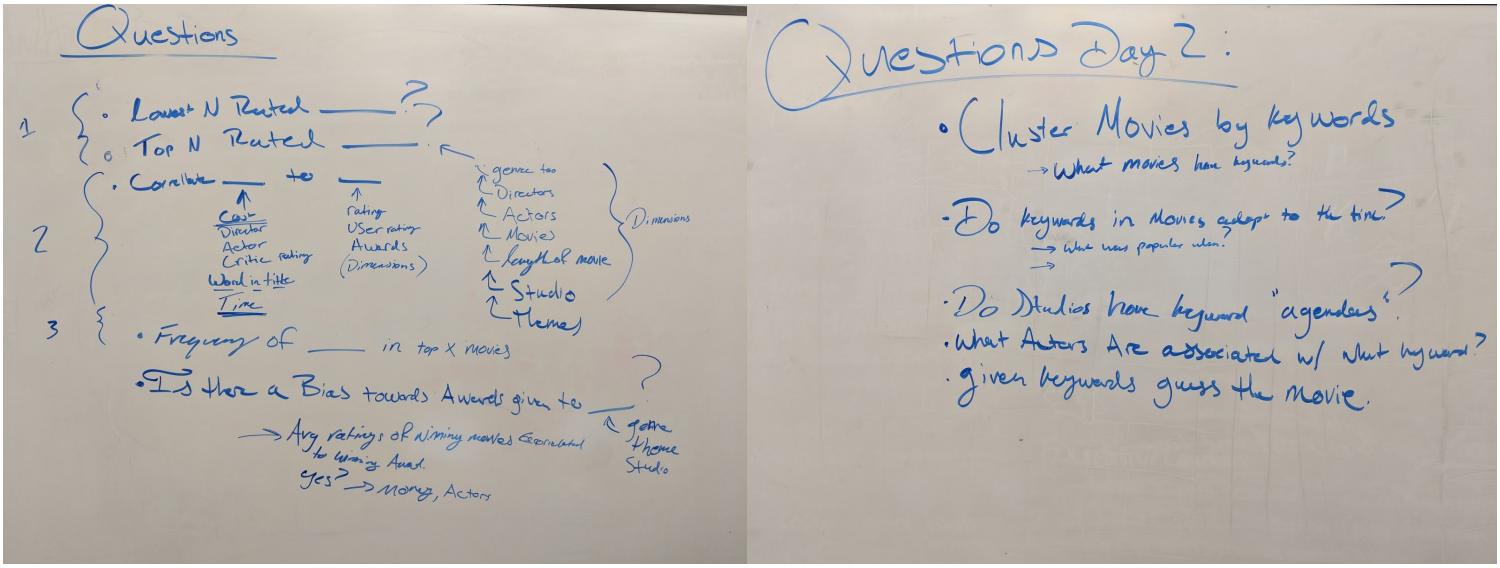




Our originally project aimed at exploring the impact we have had on Deep Sea Coral over the past few decades. We wanted to explore how it is being conserved, where it is endangered most, and where the coral lives. We had envisioned an interactive depth chart and map to help the user navigate. What we found, however, was that by the end of our designing we made the mistake of having too many assumptions about our data that didn't exist. We were short-sighted by what we could not find on Coral, and needed to try again.

Revised Project Proposal

After researching further with a strong focus on finding data with valuable questions and a means to answer them, we found "The Movies" dataset provided by Kaggle. This dataset is an aggregate of



Movies throughout the past century, with their cast, crews, plots, keywords, and more. Attached are also specific user ratings - which can be used in machine learning to predict movies users might like - but for our purposes we won't be needing those specifics. Overall, this means we have 45,000+ movies with about 26 dimensions after joining Keywords and Cast/Crew to the main metadata. After cleaning data, we found the true number of complete movie data was closer to 28,000 data points - still plenty to work with and certainly enough to have us asking many, many questions.

Overview, Motivation and Questions

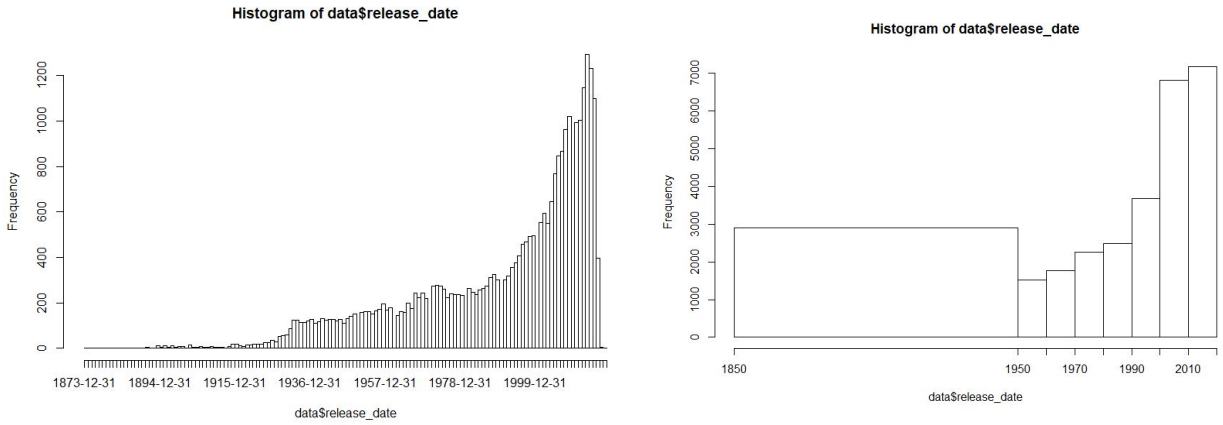
There's few mediums existent in society that reflects our cultures and ideals such as Cinema. Decades of "movie magic" and cinematography have brought to life some of humanity's deepest thoughts and funniest aspirations. What's more, movies are evolving from a Big Screen treat to an on demand fix for boredom through streaming services like Netflix and Hulu. With the increasing availability of movies, and their rich cultural history, we beg to ask - are movies truly a reflection of us, of our desires? And what impact has time had on telling that story?

Our final projects aims to explore the relationship between the keywords that describe a movie, the actors who brought them alive, and the studios that produced them. Using this data, we intend to learn more about what themes successful movies follow, what actors bring those themes most to life, and what studios are just the best at working those themes to their advantage. We'll aim at seeing how this changes with time - if perhaps movies reflect current day events or if there are simply common themes that work

better than others. To do this, we'll need to have some working definitions; for example - "success" should be a measure of revenue and user rating. "Best" should follow a similar function, albeit perhaps also inspecting frequency of the keyword. The resulting outcome, then, should be able to answer: does purely having a keyword or theme that is descriptive for that time-period guarantee that it will bring attention? Who are the major influences of those keywords?

Motivation for this theme came two fold. After our first dataset failed to follow through (see our proposal), we learned we needed more than just a question to answer but a means to answer it. This changed the way we sought our topic. Our fault came from looking for a dataset that reflected a general theme we wanted to pursue, rather than asking ourselves if there was enough sustenance to really approach the topic. When we realized we couldn't provide the missing sustenance, we needed to start over. That's when we shifted gears and took another perspective, browsing datasets and thinking of what topics might be interesting to us - but this time when filtering we carefully asked does this data carry meaning? Can we fill in what its missing? What potential does it hold? That is where the Movies topic shined. The topic is relatable, engaging, and integrated deeply with our culture - and data is boundless on it. We saw a lot of potential in what could be said, and when we dug deeper we realized there was something worth exploring to the high-dimensionality of this dataset. Instead of trying to find questions to ask about the data, the data was presenting questions to answer.

Binning:



Given the large amount of data present in the dataset, we wanted to bin the movies such that they were roughly balanced. This way we had discernable chunks to work with that wouldn't be too large of a burden on the browser at any time (think of this as a latency move). We brought the data into R and evaluated the valid movie data (~28,000 entries), and then made some quick histograms/binning decisions. The result was a large bin with movies released before 1950, followed by each decade after. This was a good compromise between bins that made sense and bins that were balanced - we now had a sense of the data distribution to bring forward.

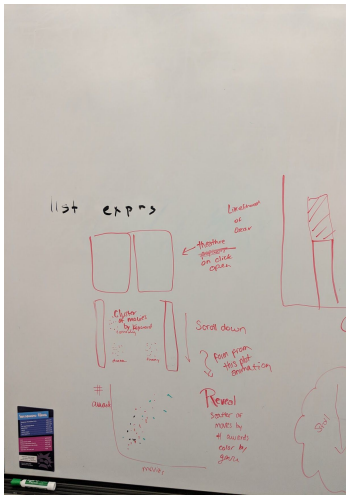
Project Objectives

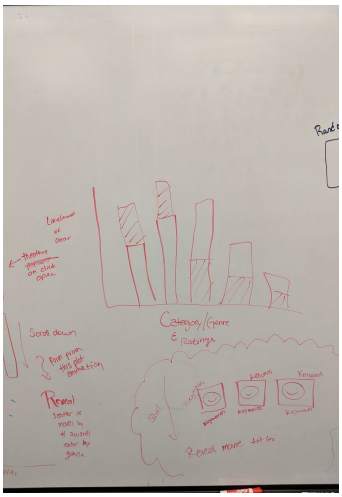
For our Vis, we wanted to inspect how related movies might have built themes for a decade via a network graph. We wanted to be able to then toggle the nodes of this graph to focus on varying interests - for example, individual movies, actors, or studios. Doing so would allow us to answer:

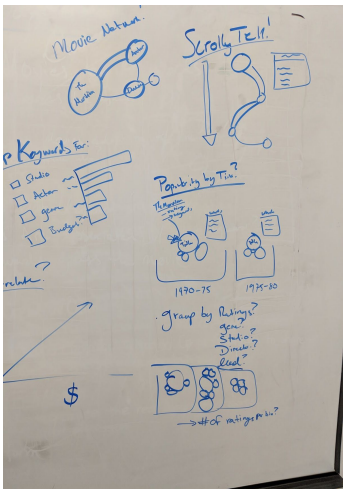
- Does the majority or vocal minority set a theme for a year? How are major players related
- What are the most popular themes (defined by top 3 keywords) by decade over the past 100 years of cinema?
- What actors, movies, and studios align themselves with what keywords?
- What studios/actors are strongest/successful in which top themes?
- How have a decade's themes evolved from the start until now?

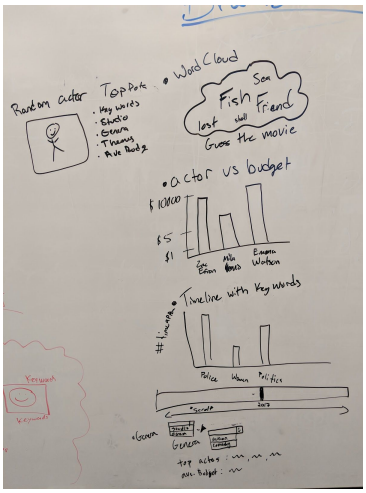
Through answering these questions we can gain insight towards the sociological impacts these themes have upon ourselves, as well as understanding of how influential specific entities are on our culture. For example, if we discovered the keyword "superhero" was a big theme in 1970, then we could explore when that theme started (by what movie?) and watch its rise each decade. We can then learn what actors and studio have had an influence on that keyword the most as time progressed, leading to its popularity. Thus, as a tangible outcome, we've now gained some sociological insight as to how "things are today" in the realm of cinema. Of course, alongside the sociological benefits, there's the "DataVis for fun" benefit where the common person can gain insight on a topic they can relate to while being exposed to the potential DataVis carries.

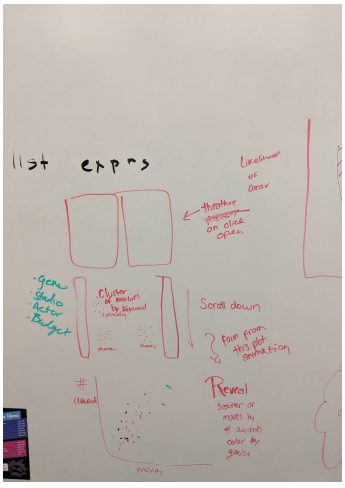
Brainstorming With Five Design Sheet

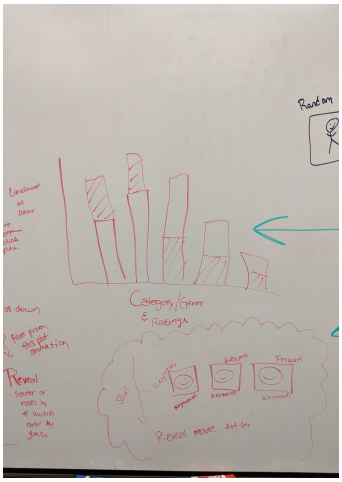


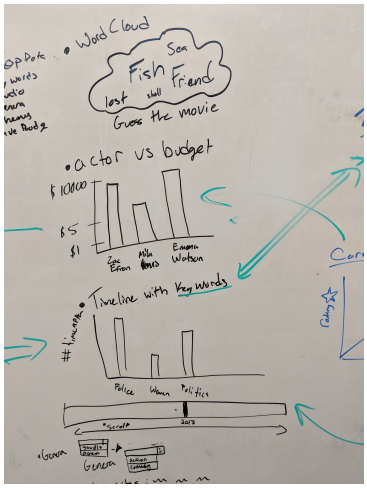


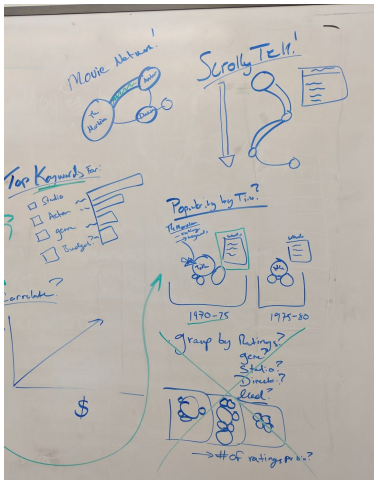


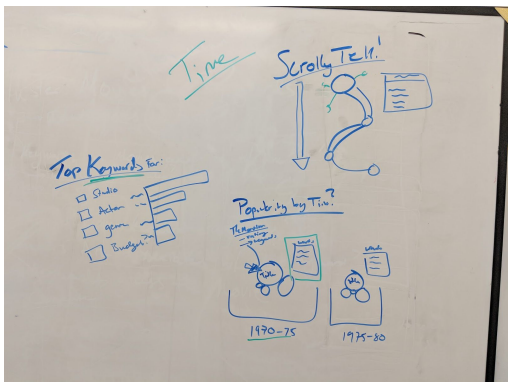


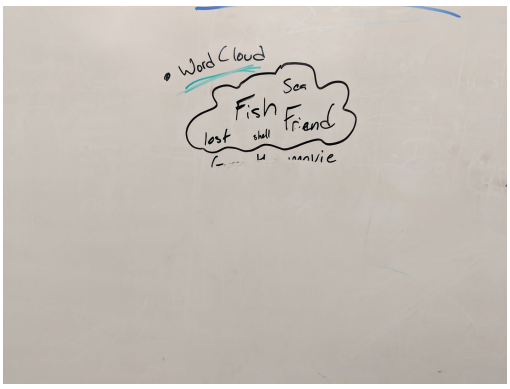


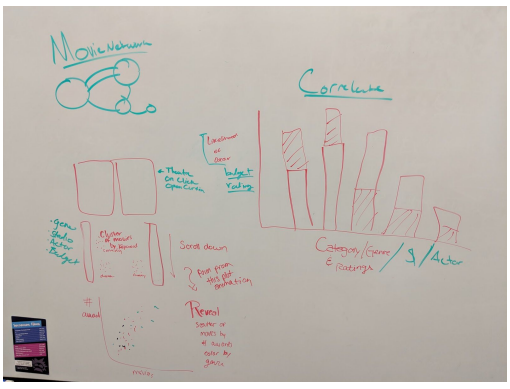




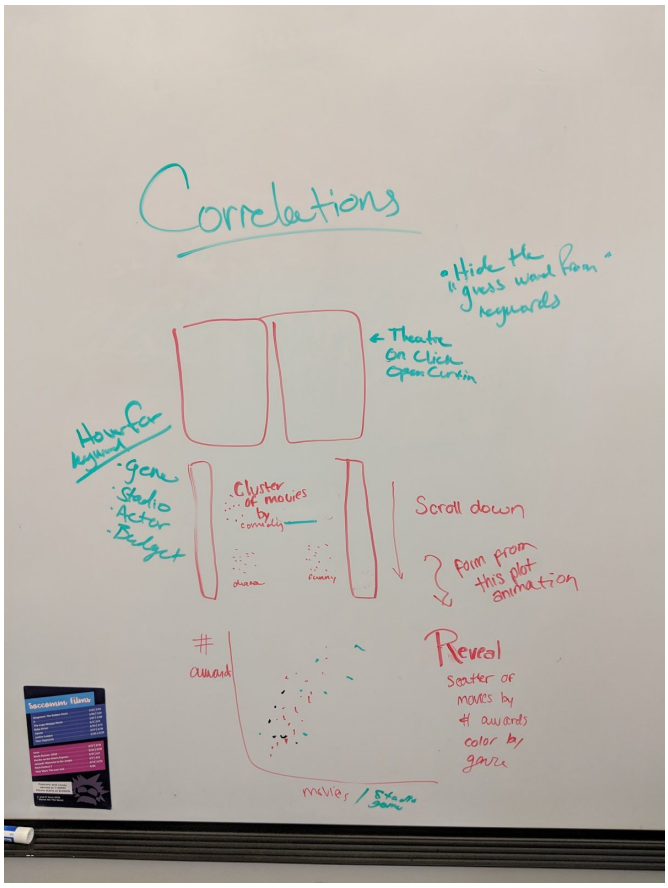


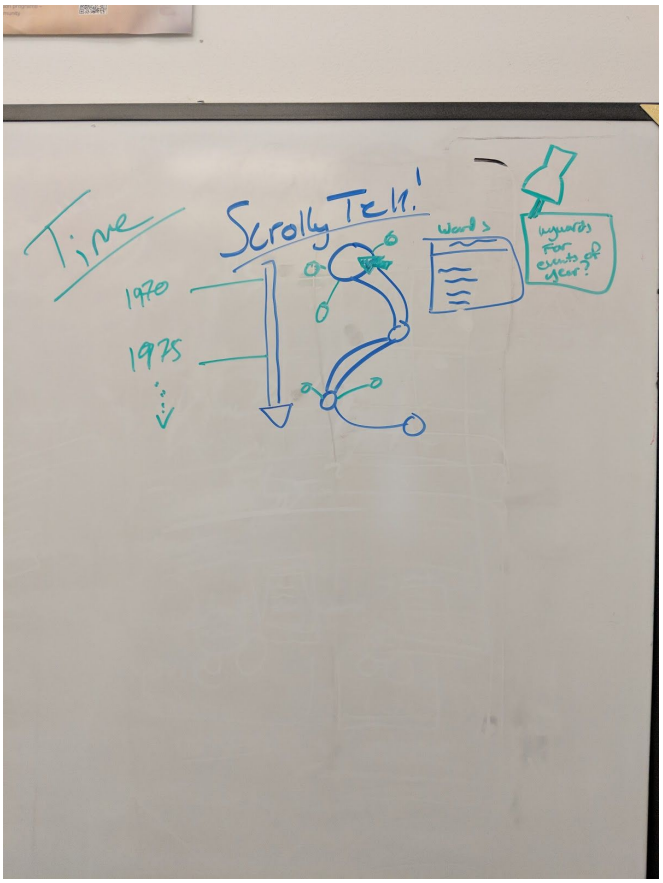






Final Step: Combine & Refine

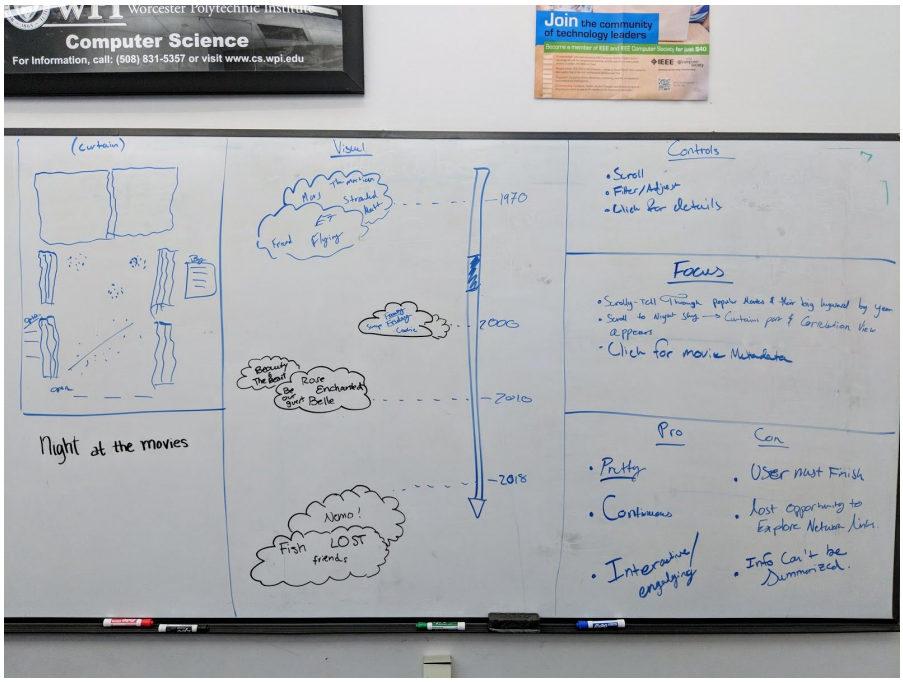




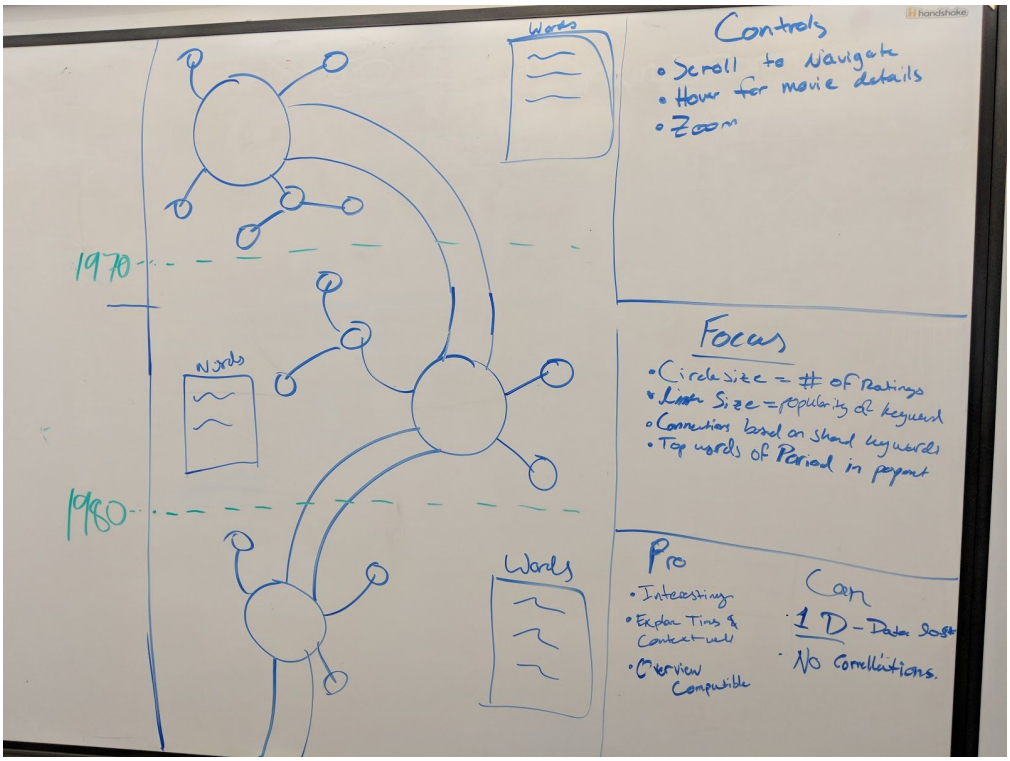
The resulting ideas we had centered on two concepts: the first, a series of correlations examining larger trends in the data. This wanted to answer higher level questions inspecting relationships that cause one thing or another. The latter wanted to explore how data changed with time, perhaps inspecting the way themes shifted and evolved through one larger network graph.

Design Evolution

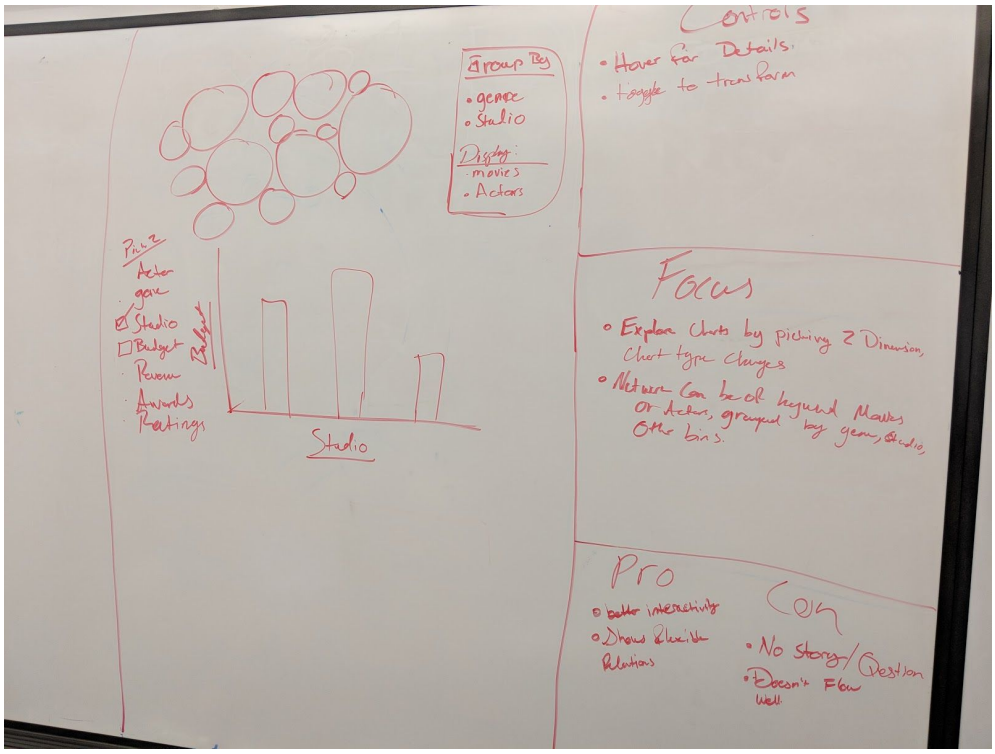
Design 1:



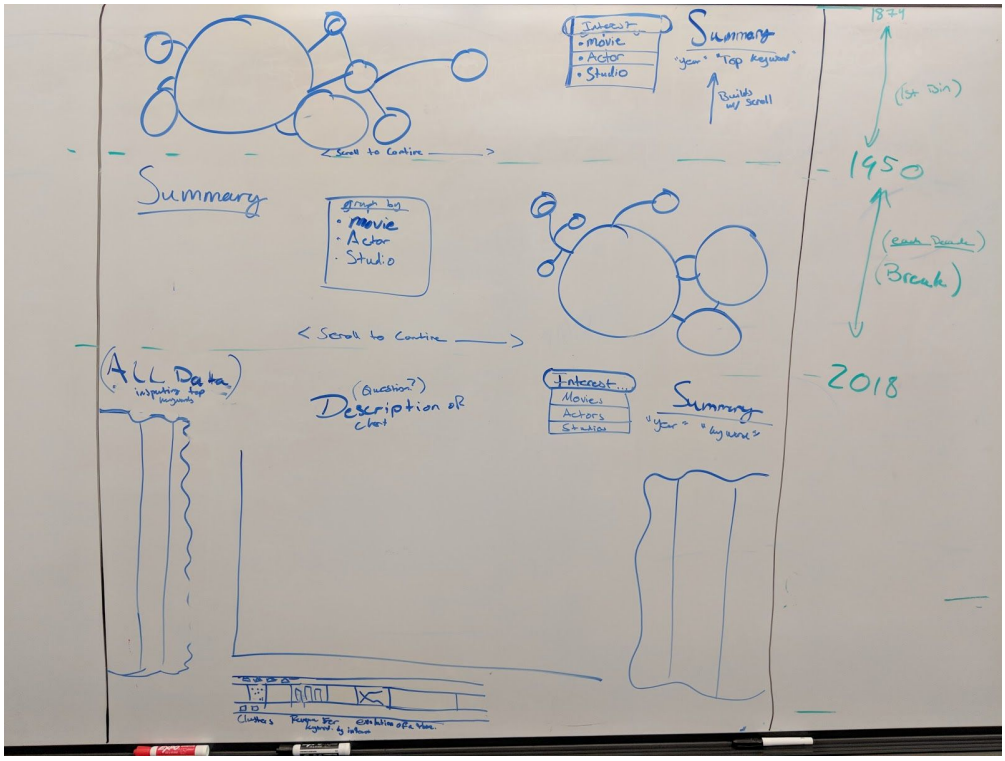
Design 2



Design 3

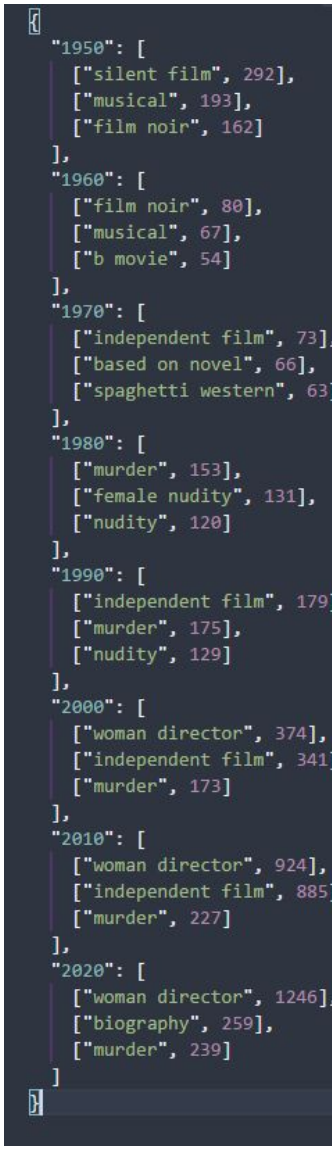


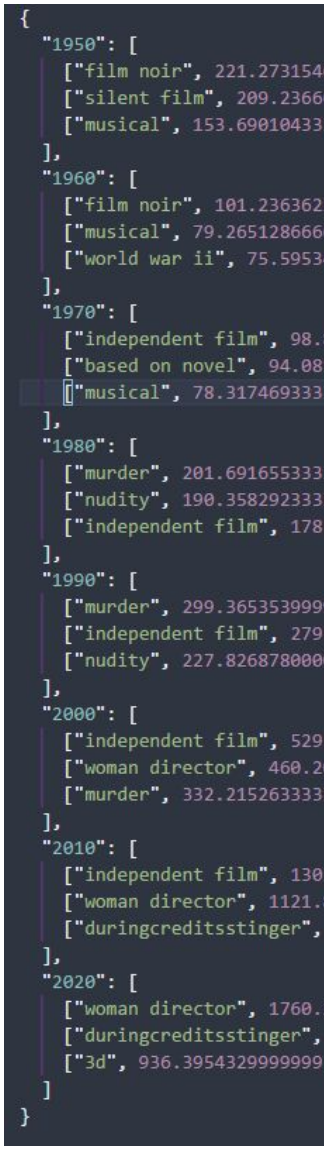
Final Design

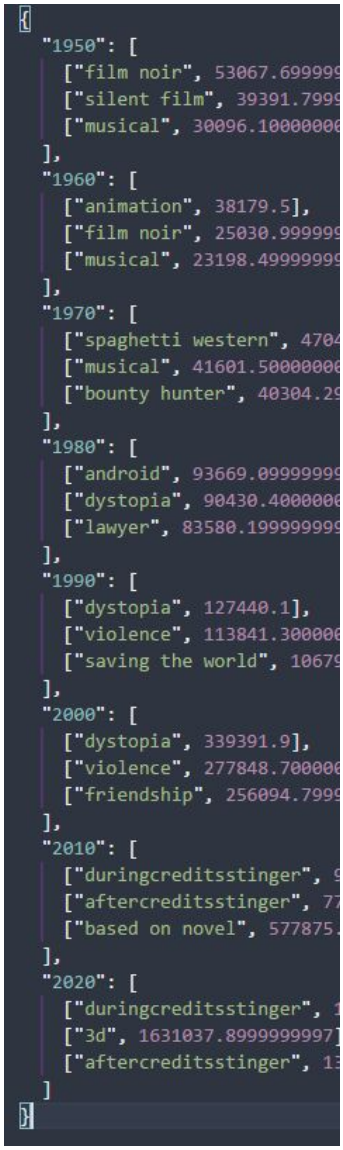


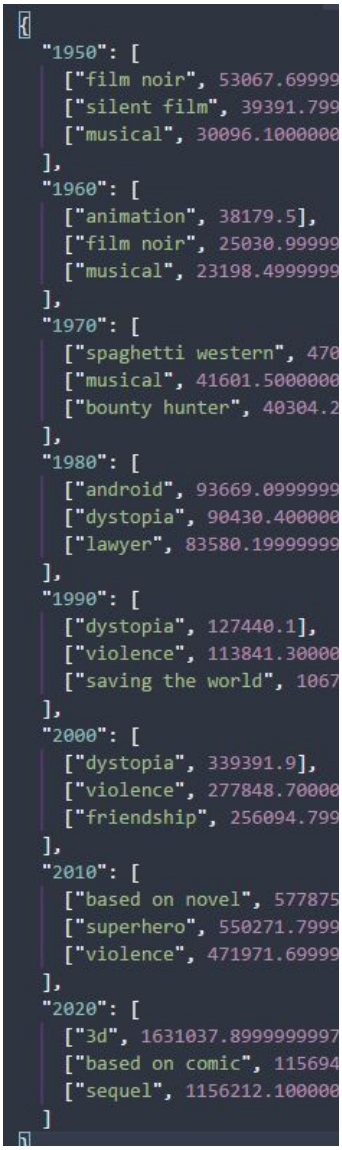
The final design was a marriage of our brainstorms ideas: a top part of the visualization where the user could scroll through the decades, exploring networks of nodes (based on interest - movie, actor, or studio). As they explored, we'd build our theme list by the decade, thus when they reached the bottom half they'd have the much needed context to make larger inquiries into the data. It was down here where we'd explore clusters, change over time, and success through bubble charts, multi-line graphs, and stacked bar charts (respectively).

Data: Defining Popularity







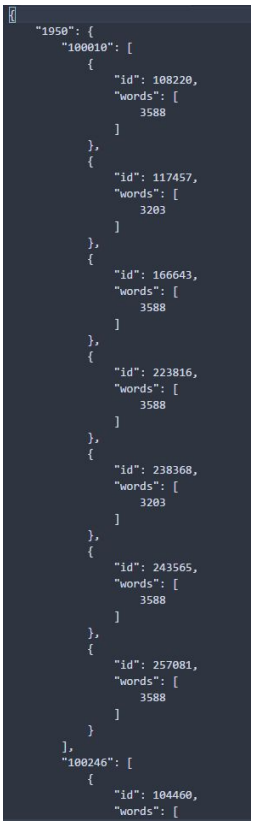


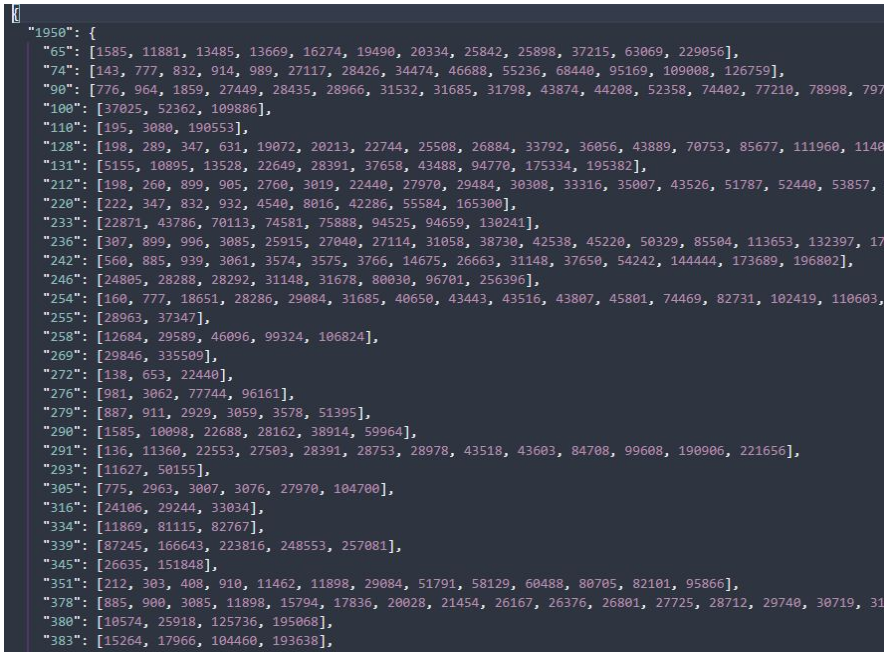
Raw Frequency → Popularity → Rating * Votes + Popularity → Rating * Votes Filtered

With a design in hand, we now turned to formatting our data. The first stipulation of that, however, was how to define the term “popularity.” Several aspects of our project aimed to convey what was popular and what was not, and as such, discussions took place as to how to define it. We looked at raw frequency of each keyword in the bin (related to its interest), the “popularity” field (which later we learned was tied to the week the data was collected), a weighted rating with popularity added, and a weighted rating with “stinger” keywords filtered out. We resolved to the final solution, as we felt it let each movie have the “voice” it deserved in the vote with the exception that “stingers” aren’t necessarily thematic - so we filtered those out.

Data: The Qualms of Networks

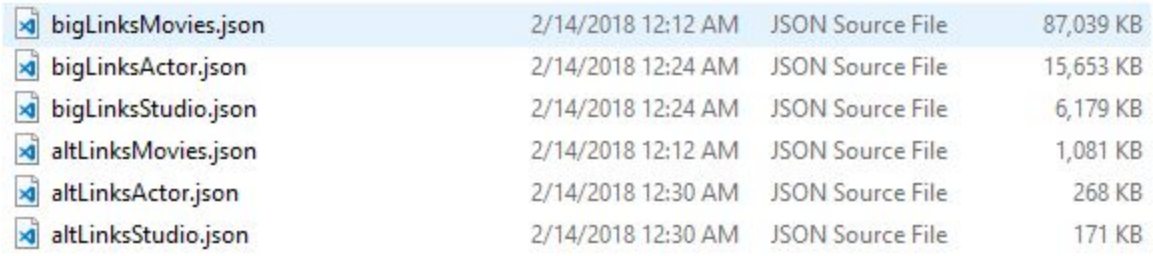
Data processing was relatively smooth, with the pseudocode outlined in our proposal. The issue, however, rose with our network chart’s links - while the novel algorithmic implementation worked well, it created a datashape that was far too vast in size, resulting in an 100+ Mb JSON file for just one of the three interests across all bins.





The expressive version (left) stored a bin’s nodes, and a list of all the neighbors of that node with the words they shared. We only stored links one way (as an optimization) and we replaced the full word strings with their IDs (to further reduce the size). The compact version (right), however, took a new

approach: Instead each bin mapped keywords found within that bin to a list of nodes it connected. This vastly reduced data sizes (compact is know an “alt” in the following)



bigLinksMovies.json	2/14/2018 12:12 AM	JSON Source File	87,039 KB
bigLinksActor.json	2/14/2018 12:24 AM	JSON Source File	15,653 KB
bigLinksStudio.json	2/14/2018 12:24 AM	JSON Source File	6,179 KB
altLinksMovies.json	2/14/2018 12:12 AM	JSON Source File	1,081 KB
altLinksActor.json	2/14/2018 12:30 AM	JSON Source File	268 KB
altLinksStudio.json	2/14/2018 12:30 AM	JSON Source File	171 KB

```
/**
 * Alternate means to build the links within the graph, storing node
 * to node links and the words that connect them. This is inefficient
 * with space.
 */
function findLinksExpressive(frontier){
  let output = {}
  while(frontier.length !== 0){
    let node = frontier.shift()
    frontier.map((other) => {

      // Find shared words between node and other.
      let sharedWords = node.keywords.reduce((acc, wordID) => {
        if(other.keywords.filter(kID => kID === wordID).length === 1){
          acc.push(wordID)
        }
        return acc
      }, [])

      // Add Neighbor if it's a neighbor
      if(sharedWords.length > 0){
        output[node.id] = (output[node.id] || [])
        output[node.id].push({
          id: other.id,
          words: sharedWords
        })
      }
    })
  }

  return output;
}
```

```
/**
 * Builds a mapping of ID to a list of {ID, SharedWords}
 * representing its neighbors. Uses the given frontier list
 * as the nodes to traverse.
 */
function findLinksCompact(frontier){
  let output = {}
  while(frontier.length !== 0){
    let node = frontier.shift()
    node.keywords.map((kID) => {
      output[kID] = output[kID] || []
      output[kID].push(node.id)
    })
  }

  return Object.keys(output).reduce((acc, kID) => {
    if(output[kID].length > 1){
      acc[kID] = output[kID]
    }
    return acc
  }, {})
}
```

Algorithmically, sometimes simpler is better.

To validate we were truly sure the two maps showed the same data, we performed a double difference computation to see each map translated to the same links being made. Naturally, the downside to the compact implementation was a need to reform the links. With that said, however, our diffs (found in the github repo) showed both maps contained no fewer or excess links than the other.

Visualizing Networks: A Tale of tSNE

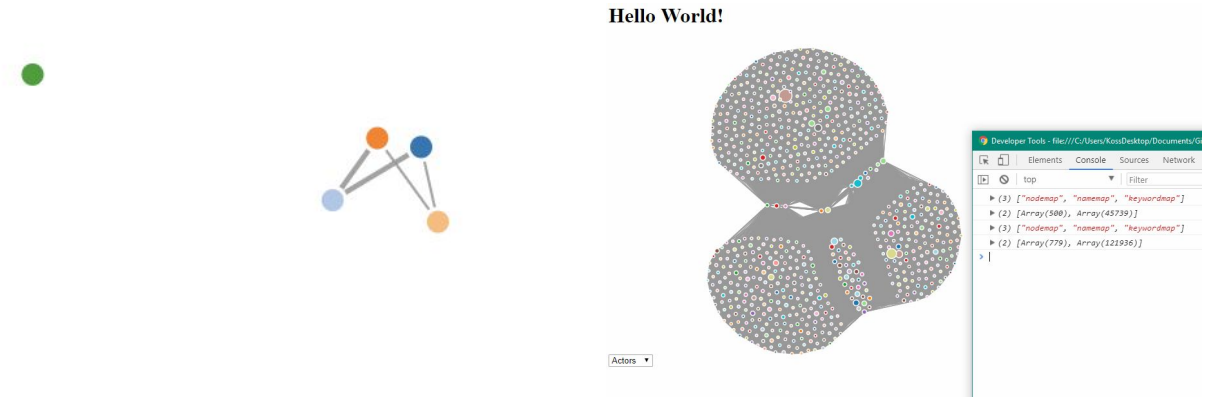


Figure 1.1: Network Graph Prototype versus Reality

While the data processing occurred, dummy JSONs were used to remain productive in preparing the D3 visualizations. This initial prototype looked quite neat. As seen in Figure 1.1, however, the real data had so many links it over-burdened the entire visualization. The latency of recreating the links in the frontend, as well, was far too strong of a delay to be acceptable. Between the thickness of the links and the performance issues they carried, we needed to reconsider if a network truly showed anything of meaning.

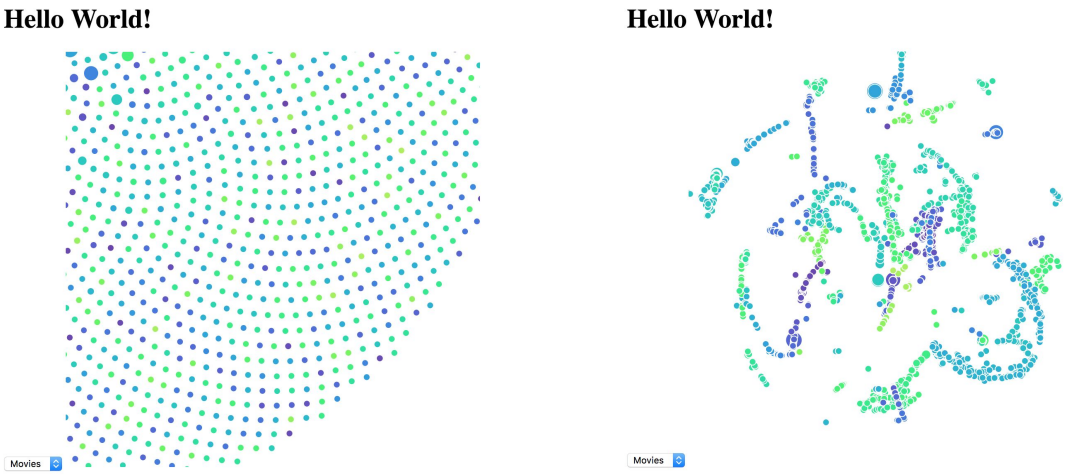


Figure 1.2: tSNE with Big Separation Values versus Pure tSNE

One of the most natural reactions from observing Figure 1.1, was that our network was closely resembling natural clusters. This led us to investigate some clustering algorithms that appeared interesting. tSNE is prize-winning technique to visualize high-dimensional data. We build a feature set

from our data recording director, studio, top three actors, and top three keywords of each decade. In Figure 1.2 shows our first attempts at building a tSNE graph, where overlapping nodes were resolved using d3-force collision resolution.

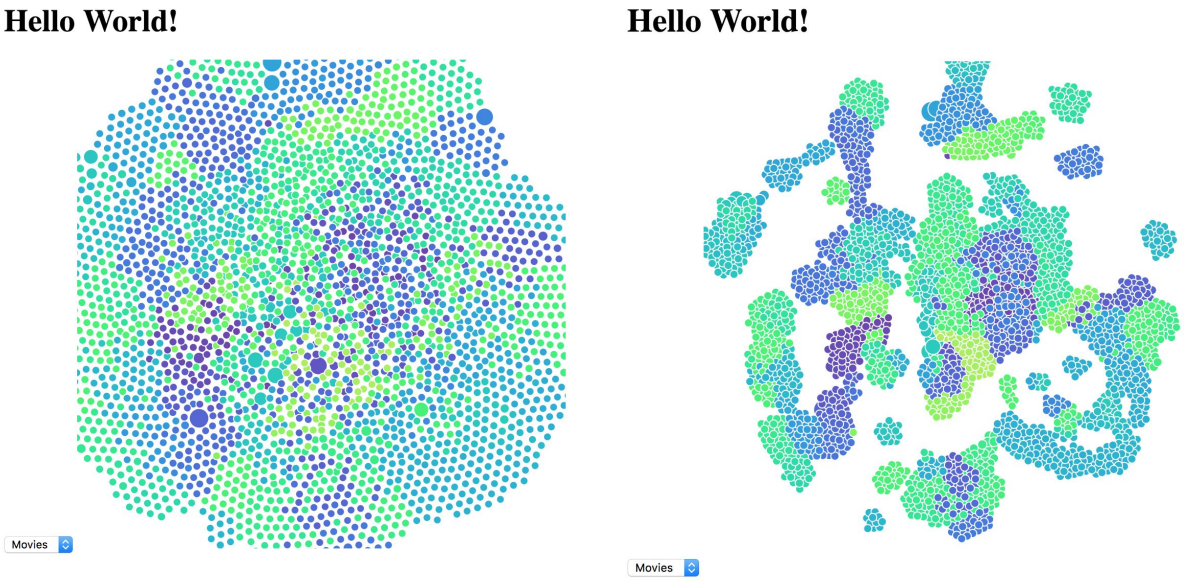


Figure 1.3: Picking the Correct Separation Values

After playing with the tSNE and force layout parameters, we were able to start spreading our nodes so they could still be inspected while maintaining their original clustered locations (as shown in Figure 1.3). One downside of tSNE, however, is that it doesn't distinctly label nodes with a cluster - thus if we wanted to draw any conclusions on these we needed to find a way to identify the clusters. Thus, we applied a K-Means clustering after tSNE to identify points within a locality, resulting in groups that could be operated on. In this initial attempt, coloring and clustering was done based on ***strictly*** the third dimension of our tSNE analysis - more on the emphasis in the next section.

This process led us to rethink the difference between what tSNE is describing and what our network graph was originally narrating. To ensure that we were focusing on narrating a story, we generated questions that would help us rescope our graph based on our new approach.

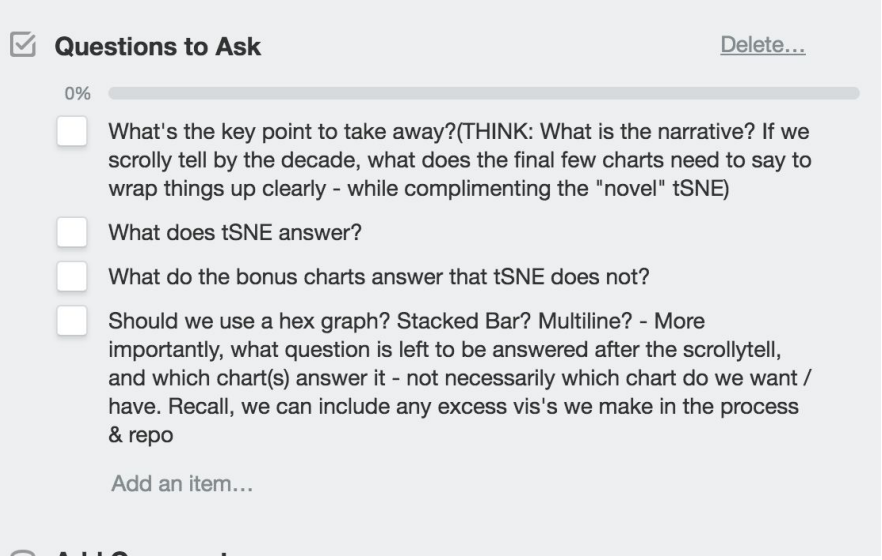
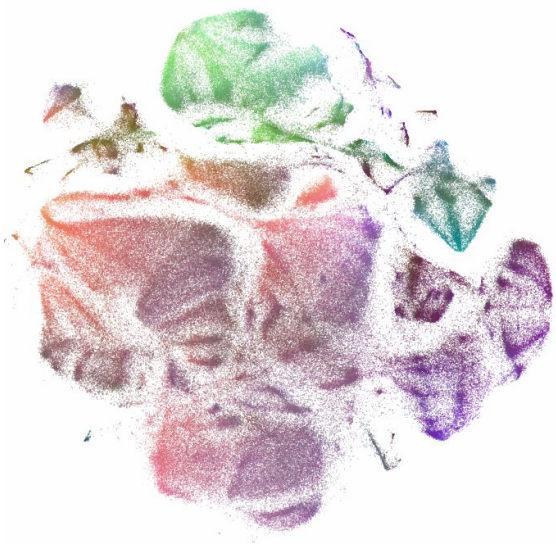


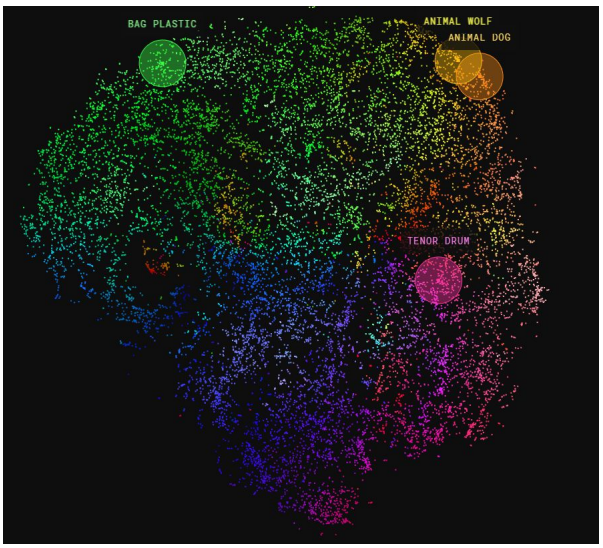
Figure 1.4 Asking the important questions

The questions above illustrate the overall process we took to decide on how to structure our scrolly-telly. After analyzing carefully how tSNE was clustering movies, we decided that we would highlight interesting clusters in each decade. Whether it encompassed the most popular movies in this year, or if they had an interesting cluster -- our main goal was to describe how the themes in movies developed through the year and define how human behavior historically approached the entertainment industry. *This meant, to keep the project in scope, we removed the connotation of interests from the visualization - focusing strictly on movies.*

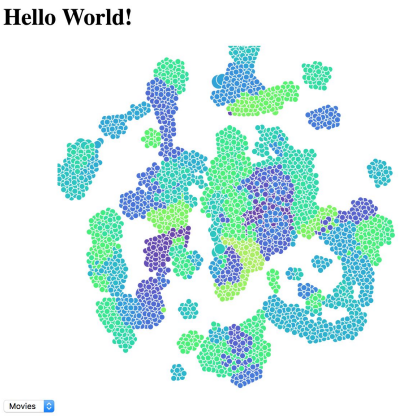
In addition, we decided to reinvent our conclusion due to the fact that revenue data was not as readily available as we believed. Instead, we wanted to illustrate a glimpse of what the user had just observed in a different manner. We created a hex map for each decade as a means to compactly represent the tSNE coordinate data in a smaller spacing - creating a snapshot of what the user had just seen.

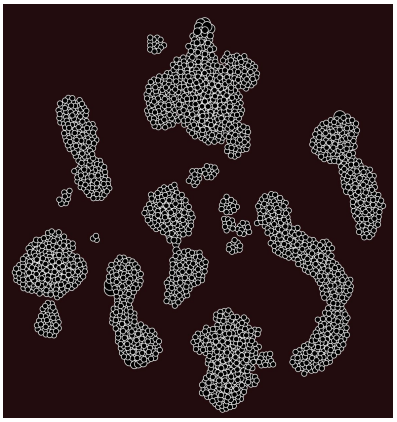
Color and Space, the tSNE Story

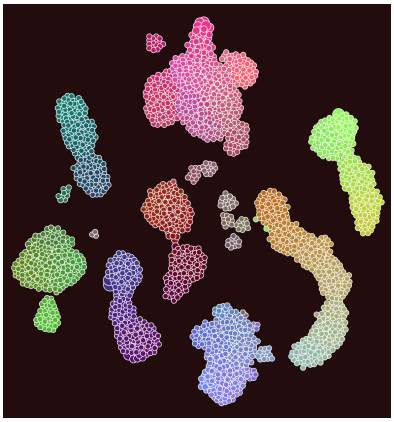




Understanding how to color tSNE was a different story. In the previous section we remarked explicit coloring our tSNE by it's third dimension after reading the work of Kyle McDonald (left). What we didn't realize was that we were coloring the space wrong. It wasn't until after we found a related work done by the Google Experiments team, Drum Machine, that we realized space and color were more related than just the third dimension. In the previous iteration, we coloring by only the third dimension meant we really were only conveying *depth* of a point in 3D space but everything was still plotting on a two dimensional plane. While this works, it doesn't quite make full sense - how deep is deep? And how can we make sense of points that are across the page but similarly colored and clustered?







Thus we introduced the faster BH-tSNE implementation to our data pipeline, and created both a 2D and 3D tSNE clustering of our features. We ran kMeans on the 2D representation, plotting that in our space. This created tighter clusters that could be better perceived compared to the 3D alternative. To then color the data we exploited the 3D tSNE, but instead of just scaling over the Z-Axis this time, we

normalized the entire coordinate into an RGB triplet. Using this value, we then colored each point to achieve a result more akin to the two inspiring projects. This, then, should not be perceived as a mapping of depth but an idea of locality - in addition to the locality in space, color can now tell the story of how clusters change within themselves.

Optimizations: The Random Post-Processing

- Searching was originally performed by adding attributes to the DOM and using CSS selectors. It was then moved to a D3-Selection filter such that all data backing the visualization could be substring searched
- Upon searching the opacity was originally set, which for high particle counts caused severe latency in the visualization due to the CSS processing. As a more performant alternative (but more verbose to encode), the alpha channel of particle colors was set to convey opacity.
- The Force went through several alternatives to find ways of improving performance. One of which attempted to use workers to render a static force layout in the background, however this failed due to too tight of coupling between the homing force and DOM elements. Rather than doing a regular static force layout (that would still result in UI latency while it loaded), the AlphaDecay was dropped thus making the Force layout settle faster in a local minutia. This turned out to be a good trade off between latency and speed in the high particle decades.
- To combat latency while moving into the conclusion or between decades, a dispatch was added that will stop the current Force simulation (if it wasn't already) while transitioning. This frees the brower's resources to smoothly render the transition.

Implementation: Welcome to the Theatre

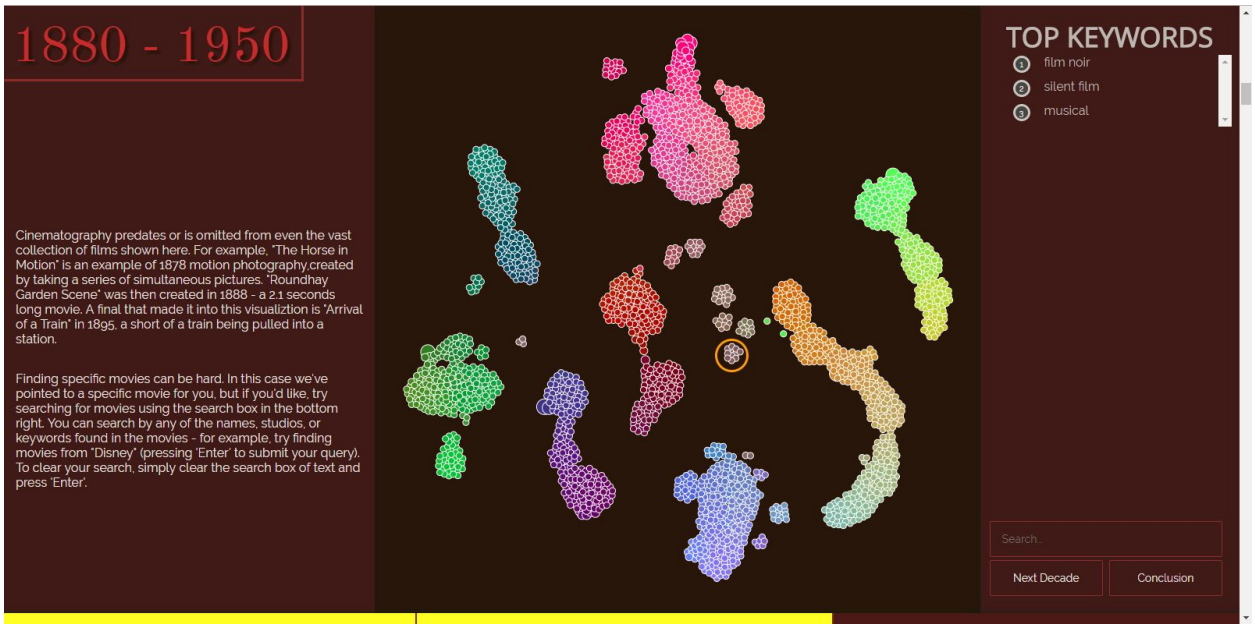
The layout's purpose is simple: *convey the story*. As such we've kept it simple and to the point. The splash screen introduces the project and its ideas, guiding the user to scroll onwards with two jumping arrows. The original implementation utilized a start button, but after further deliberation it was recognized being consistent with scrolling would avoid confusion. Finally the background has a simple gradient animation, moving from blue to red to simulate a movie theater lighting.

Figure 2.1 Splash Screen introducing the visualization and its purpose.

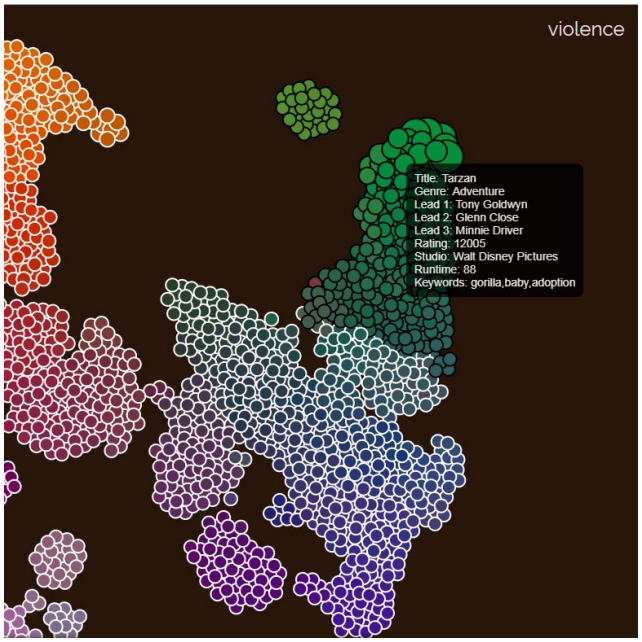
Upon scrolling, we transition to the first decade with a blank screen. The left describes a little about what the user should expect to do while navigating, just to get them warmed up for what is to come. The layout remains static from here on, showing narration on the left, the tSNE analysis for each bin in the center, and the top keyword list on the right. The center has a darkened background in order to separate the components and allow our graph to distinctly contrast. The right has two buttons to allow the user to skip to the next decade or directly to the conclusion with smooth scrolling. In addition, a search box sits ready for the user to help navigate the page. At the bottom is a progress bar, giving the user context of where they are at within the decade. The colors used mimic that of movie theater curtains - the deep red both a means to reminisce on the thought, while also helping make important elements pop with contrast.

Figure 2.2 First Page of the tSNE analysis, used as a means to orient the user

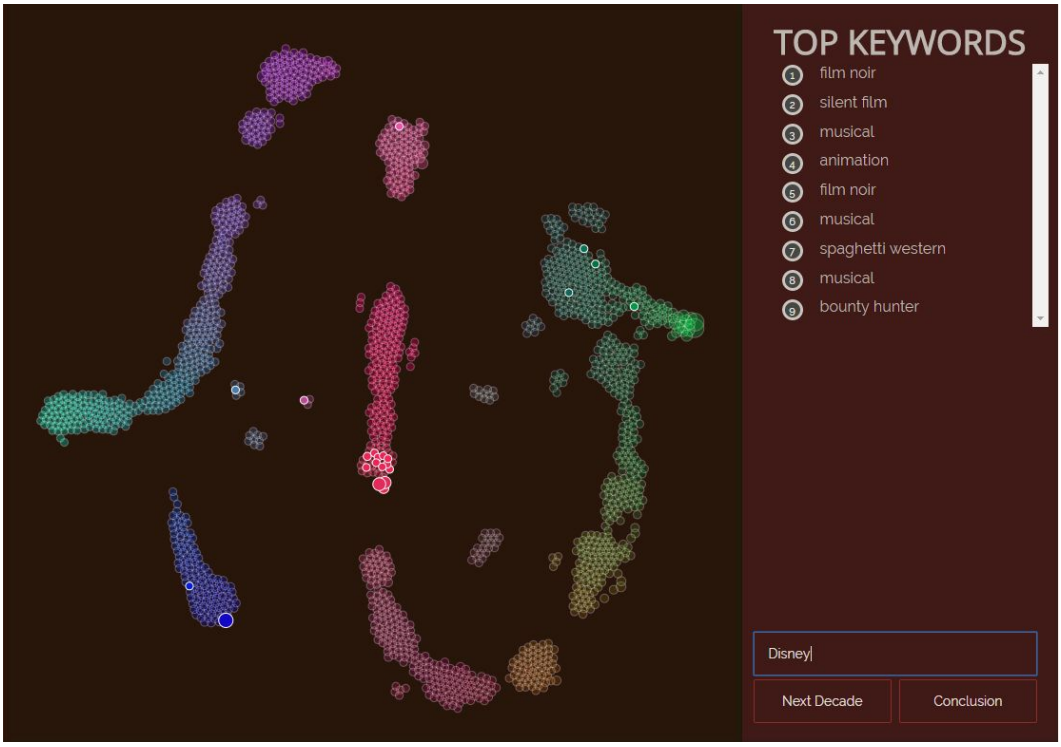
The main form of navigation within the Vis comes from scrolling. As the user scrolls, narration transitions while the progress bar at the bottom progresses. When the bar fills, the user will have scrolled enough such that the next narration to appear will transition the tSNE chart to the next bin. As the user scrolls, the narration will point to (or filter) specific areas of the Vis for them - helping draw their attention to what is being talked about. This is done through the usage of d3-dispatch coordinated with when the corresponding narration appears on the DOM.



Interactions with the tSNE charts include hovering over bubbles to display a tooltip describing it. This will show the fields in which tSNE was computed over so the user can try to discern why certain movies appeared in certain areas. The tooltip has been made such that it will never leave the svg. Additionally, while hovering, all movies that were clustered with that one will have their border change to black so the user can see the precomputed groupings kMeans found. If the user would like to get a closer look at a cluster, since they can be packed fairly tight, the chart has been d3-zoom enabled. Thus, they can zoom and pan around to get a closer look.



Since there can be a lot to wade through within a bin, a search box has been implemented that will allow the user to search the string based fields of the movies. This includes top 3 actors, the title, director, studio, and keywords. The search is a case-insensitive, substring based search - so partial matches will occur. The search box will not execute until the ‘Enter’ key is pressed, to prevent latency form constant DOM updates while the user types. Upon searching, all irrelevant movies will become translucent while relevant movies stay opaque.



To conclude the visualization, as hinted before, we originally envision several graphs comparing time and revenue for keywords. It was discovered, however, that the amount of revenue data was highly lacking for movies and needed an alternative. With a goal of still summarizing the user's experience, we took inspiration from a brewery visualization by the Pudding. We used a flex grid to display each binned decade as a hexbin plot, allowing the user to click to fast travel back to the tSNE of that bin should they wish to revisit it.

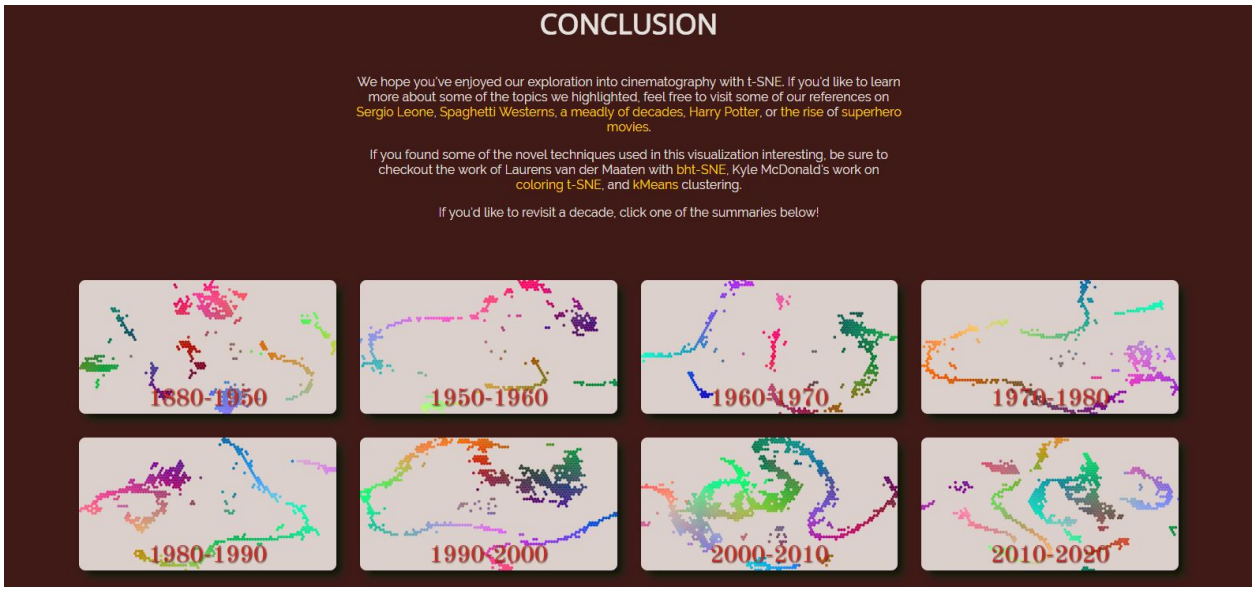


Figure 2.3 Conclusion Page

Evaluation: The Oscars

- Tech-wise we learned a lot in how t-SNE is plotted and the various ways it can be encoded with color. We explored how to obtain clustered information from this scatter using k-Means and got to test varying parameters on the t-SNE to see what made the best results. We also learned about alternate implementations that enabled generating t-SNE plots quicker, albeit still not fast enough to necessarily show in real-time (for this use case).
- From the data, we found a lot of insight in how cinema has evolved with time and where its most important aspects contributed the most. New genres such as Spaghetti Western was brought to our attention, and was interesting to learn about how it came to be only to then see its surge in popularity. In general, seeing those keywords appear and them being reflective of the time was truly rewarding - as it shows the visualization did its job well.
- The clustering worked pretty well, often clustering the big blockbusters together or pushing studios near one another. There's more than can be said about rendering the t-SNE is browser via

a server and potentially showing different ways the clusters can form - perhaps it might show differing conclusions based on the “perspective.” There’s also so more tuning that can be done with the K-Mean clustering to get better results per-decade (at times you’ll notice multiple clusters might get grouped together).

- In terms of the visualization, the polish looks pretty good on this main part - next steps might look at integrating some of the other parts of our original idea to summarize across decades better. It could also be interesting to see how one might compare t-SNE plots across decades - perhaps by overlaying them and using the search feature to compare keyword evolution. This would certainly require a swap to canvas, however, with an increase in particle count.