DESIGN CHALLENGE 2

Arpit Jain (ajain74@wisc.edu)

Goal

What?

To let sellers take data-driven decisions to improve their sales.

How?

The application visualizes the customer review data of a seller's products. Sellers can thus identify the best and worst-performing categories, best and worst-performing products in each category, and how are these products doing overtime to find trends and patterns.

Try it out!

- **Website** (hosted on Heroku): https://cs765-dc2-visualizer.herokuapp.com/myapp
- **Source Code**
- Run it locally
- Checkout out the ***video***

Note: The website uses an unpaid Heroku account and thus the load time of the application is **approximately 30 seconds**. Please don't give up as the page loads.

Step by Step Walkthrough

The application has 2 pre-built sample datasets and allows users to upload custom data as well (more on this later). The application has various tabs for specifics tasks and we will go through them step by step, trying to adhere to our intended goal.

1. Landing Page Tab

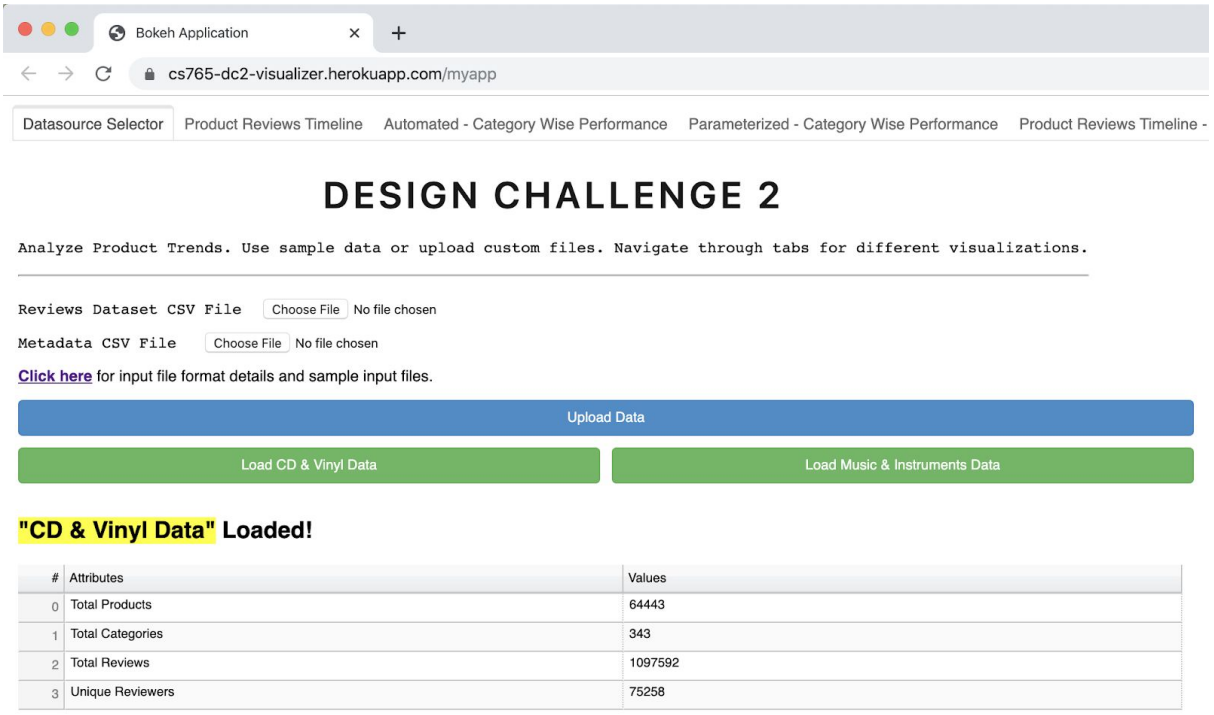
Description

The landing page provides an overview of the uploaded data as a simple tabular visualization. The content comprises of 1) Total Products, 2) Total Categories, 3) Total Reviews, and 4) Unique Reviewers.

Designs and Intents

The intent is for the user to get a high-level overview of the uploaded data. The design should present these attributes and consider the scale of each attribute as they can

significantly differ from each other. For instance, there are 343 categories, while 1000 times more, 1,097,592 reviews.



Use Case Evaluation

Let us take a running example by loading “CD & Vinyl Data”. The loaded data tells the user 1) Total Products, 2) Total Categories, 3) Total Reviews, and 4) Unique Reviewers.

Scalability

This design scales well to any number of data points in the dataset as we are going to represent them with just a handful of attributes.

2. Automated - Category Wise Performance Tab

Description

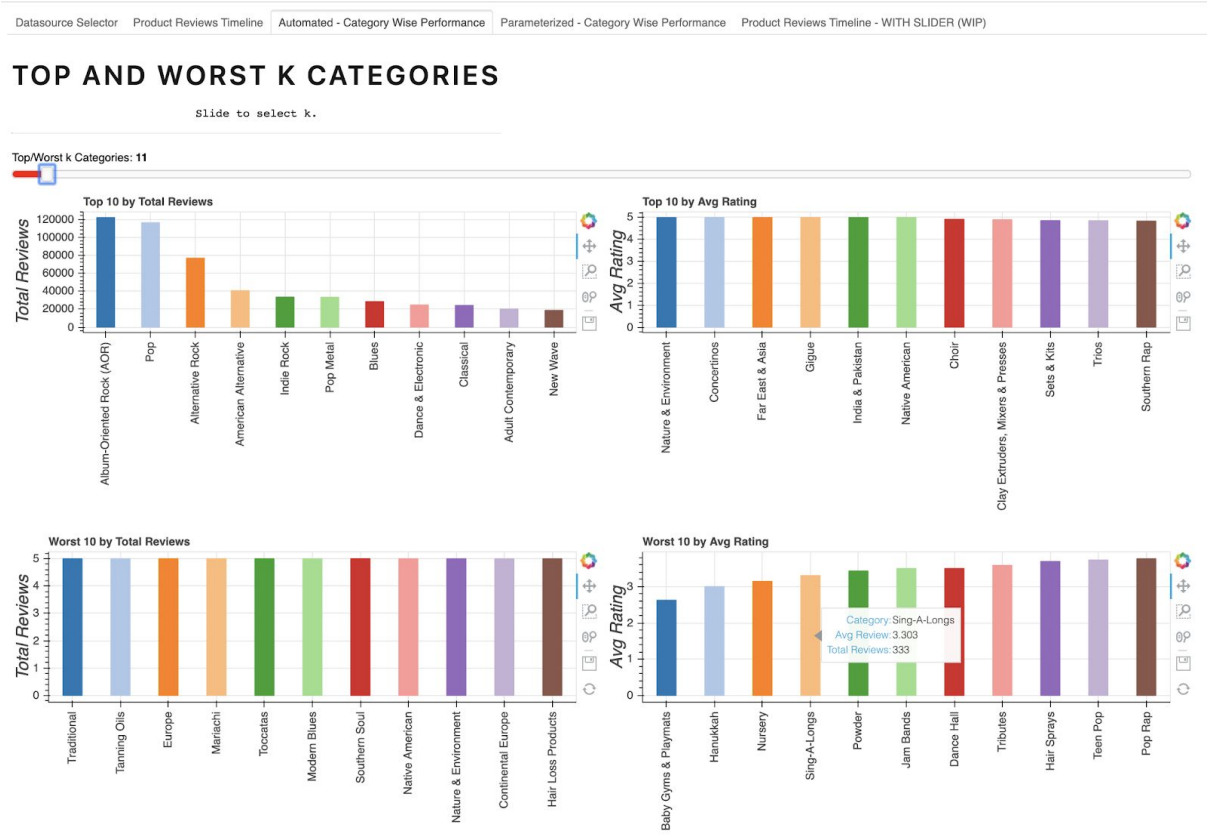
This tab allows the user to get the Top ‘k’ and Worst ‘k’ performing categories. The value of ‘k’ is controlled using a slider.

Designs and Intents

We have three data points, 1) categories, 2) the total number of reviews, and 3) average rating. The visualization comprises two bar plots, one for each attribute, the total number of reviews and the average rating. Categories are encoded as position along the x-axis and other attributes are encoded as position along the y-axis. The categories are also color encoded just to make the distinction evident. The position encodings help get the exact

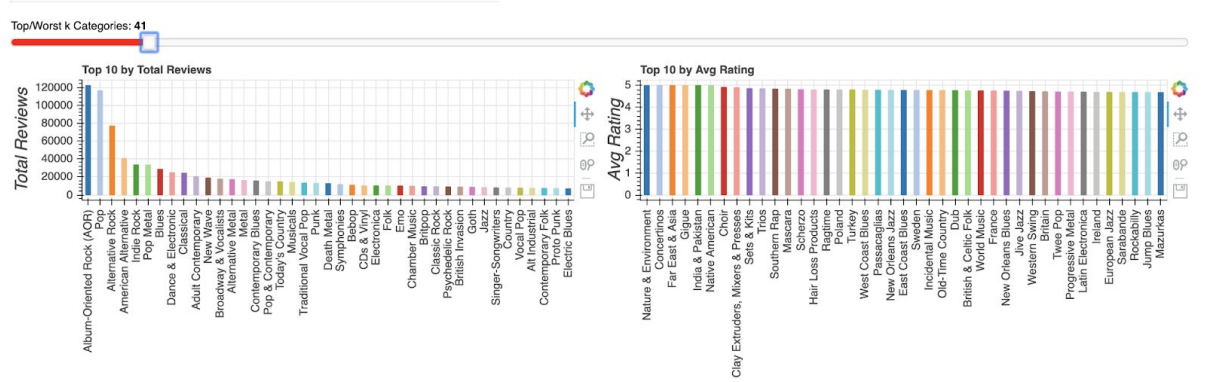
values and thus make a precise decision about the performance of the category. The visualization also uses data on demand via a hover tool.

Use Case Evaluation



Continuing with our running example, the user can navigate through the visualization to identify which categories have high total reviews and poor average rating. In this case, “Baby Gyms & Playmats” is an evident poor performer with an average rating of less than 3. Upon navigation, the user can also identify “Sing-A-Longs” to be another such category with an average rating of 3.3 and a total review count of 333.

Scalability



Scalability is a challenge as we increase k. And hence, I considered the next design.

3. Manual - Category Wise Performance Tab

Description

The problem with automated detection of category performance is the normalization factor. How many total reviews and what average rating threshold makes a category good or bad performer. For instance, it is possible that the average review of a category is 2, but it only has 3 ratings (1 star, 2 stars, and 3 stars). Also, the generic visualization of “Top k Performers” does not scale well.

I addressed these issues by giving the user more control over what they want to search. Users can feed in values for “Average Rating Less/More Than”, and “Total Reviews More Than” parameters and the visualization will display only filtered results.

Designs and Intents

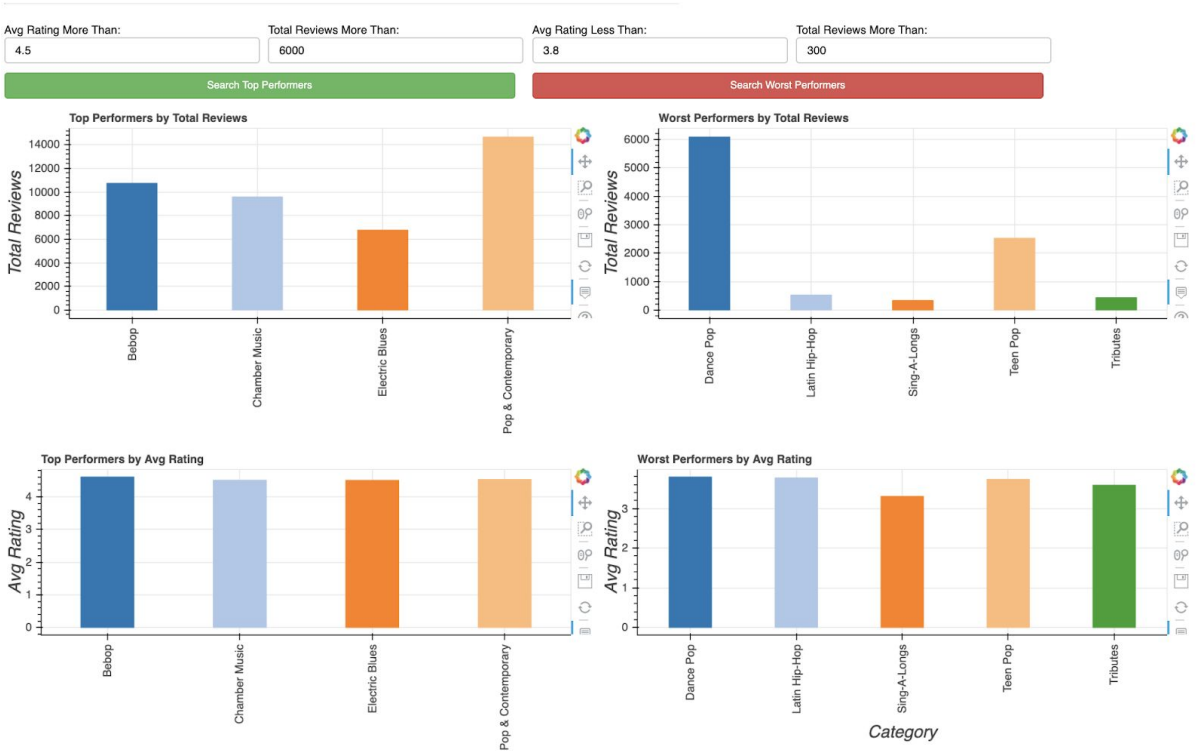
The design and encodings remain the same as in automated visualization. The intent of a parameterized search is for the user to take more control over what they are looking for in the visualization. The visualization also uses data on demand via a hover tool.

Use Case Evaluation

Consider this case where the user specifically searched for products with an average rating of less than 3.5 and a total review count of more than 100. In this case, only “Sing-A-Longs” category is identified as worst-performing.

Scalability

This design also addresses the scalability challenge of automated visualization to some extent. This is because there are fewer results for each query given the visualization is now more constraint. Though one should note that the scalability challenge will surface again if there are more results even with constraint search.



4. Product Review Timeline Tab

Description

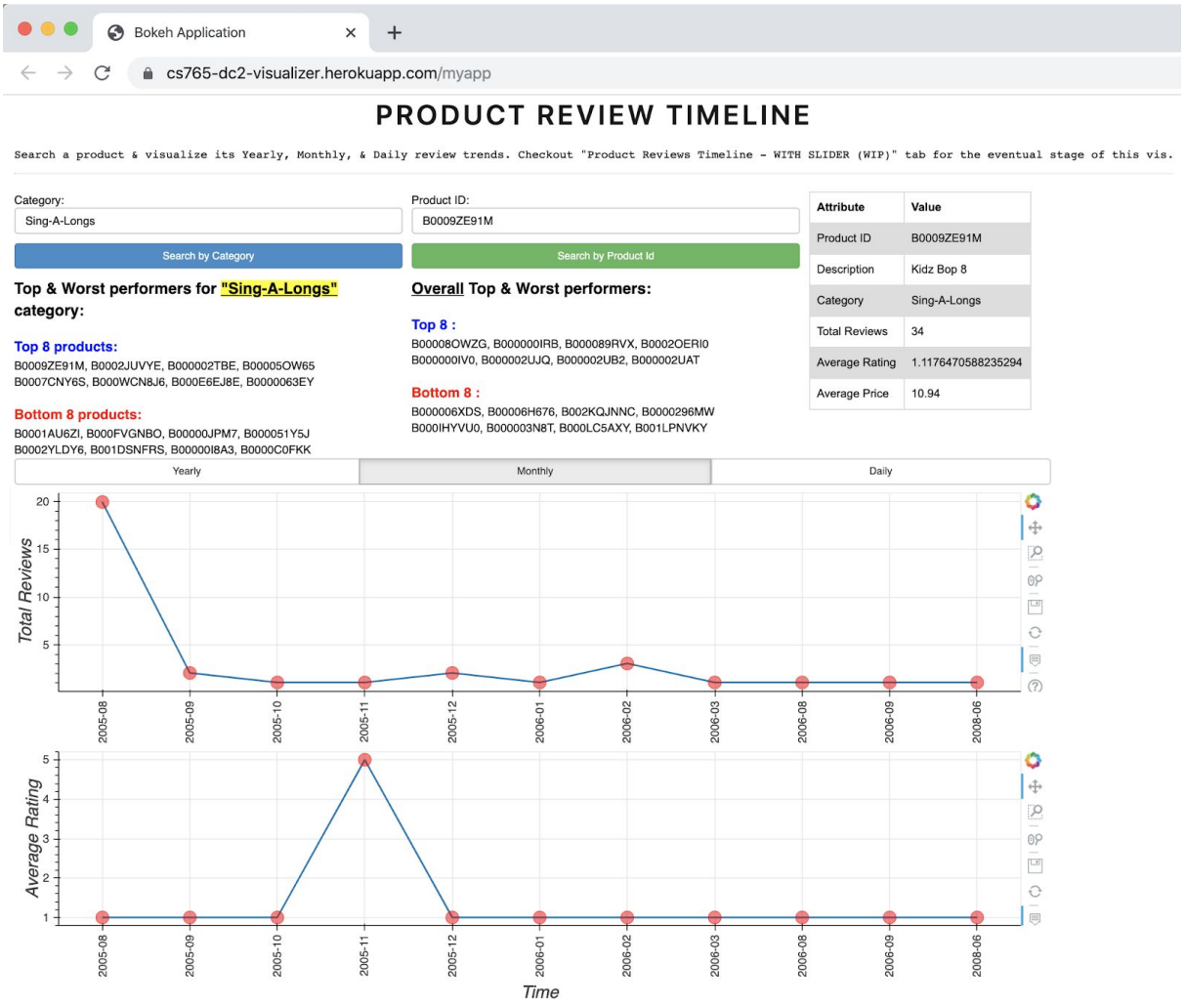
Now that the user has some idea of the best and worst-performing categories, she can now analyze the products within each category. To facilitate the same, I developer the “Product Timeline” tab.

Designs and Intents

The product timeline is a heavily parameterized tab as the user is attempting to analyze low-level details to find trends and patterns. It allows the user to search for a product, provide a glyph of the product’s information, and two graphs for the product’s total reviews and average rating over time. The time scale can also be switched between yearly, monthly, and daily scales. As the product data is vivid, hence a glyph design choice enables

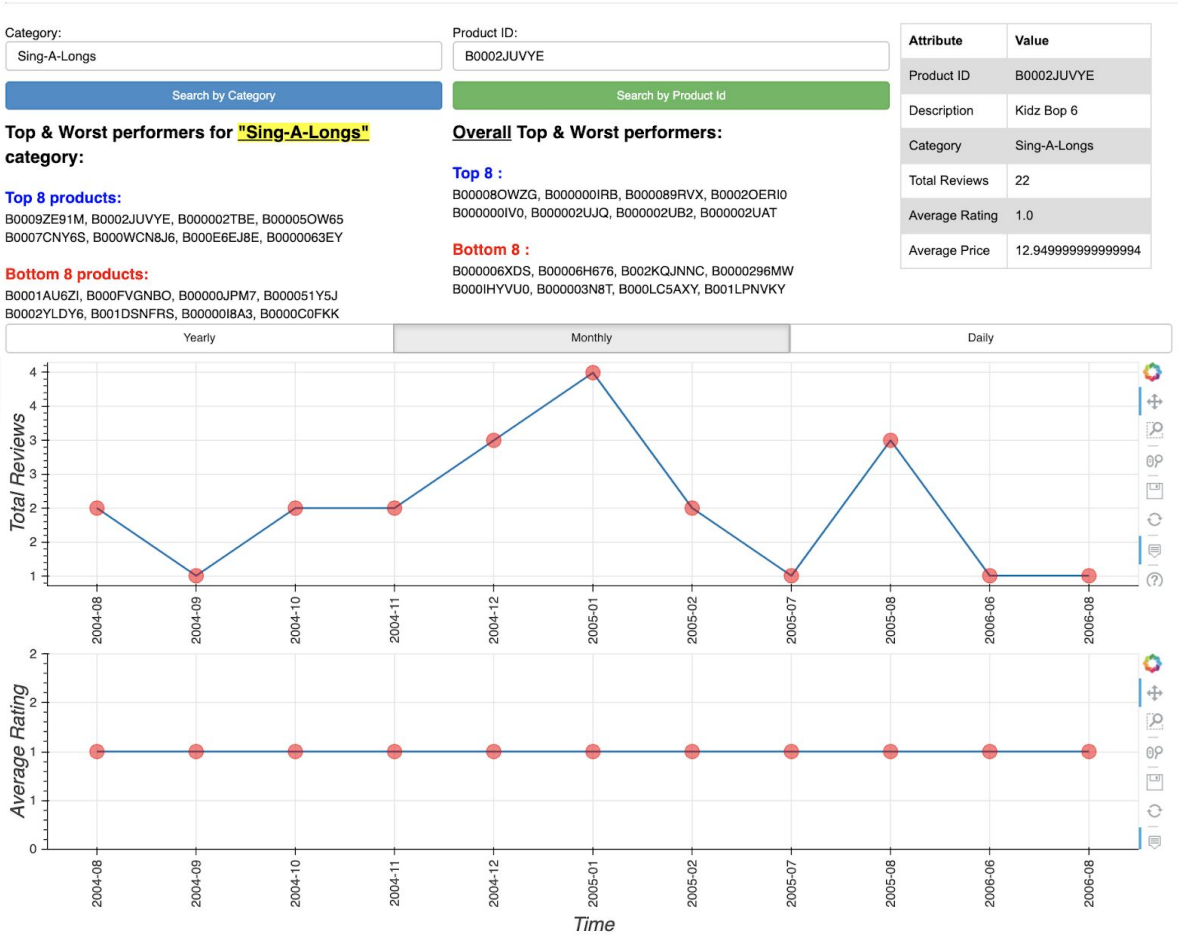
showcasing all the attributes. The total ratings, average ratings, and time are position encoded for the user to extract exact values during the analysis. The visualization also uses data on demand via a hover tool.

Use Case Evaluation



As per the running example, we have established that “Sing-A-Longs” category is a potential poor performer. So I searched for the same category and got the top & worst performers of the category as per the total number of reviews. Now I picked the top result, B0009ZE91M and searched for its timeline. It is evident that while this product has 34 reviews, most of them are 1 rated.

To further establish the hypothesis, I continued searching with the products next in this top list. As expected, almost all of them had a high total review count but an average rating of less than 2.



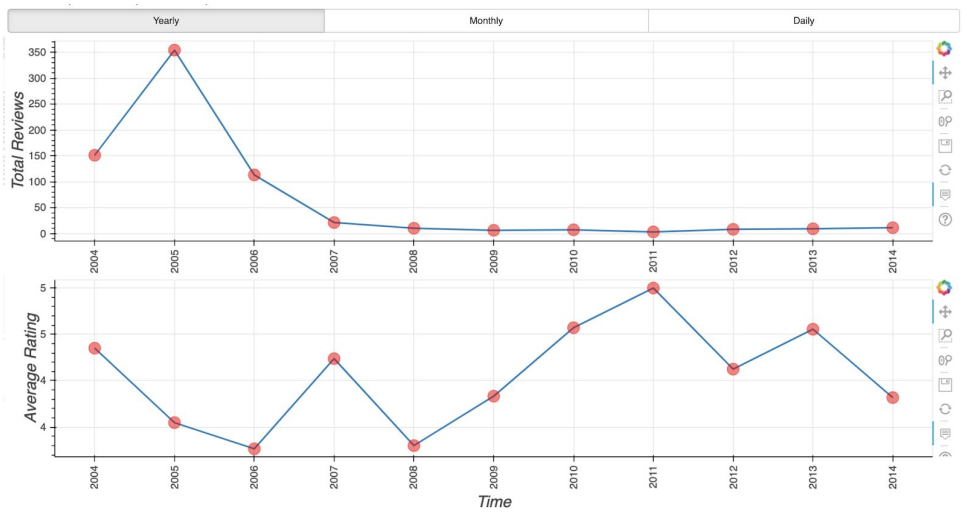
Thus, as a seller, I now have information about which of my products is not performing well and I can take measures to fix the same.

Scalability

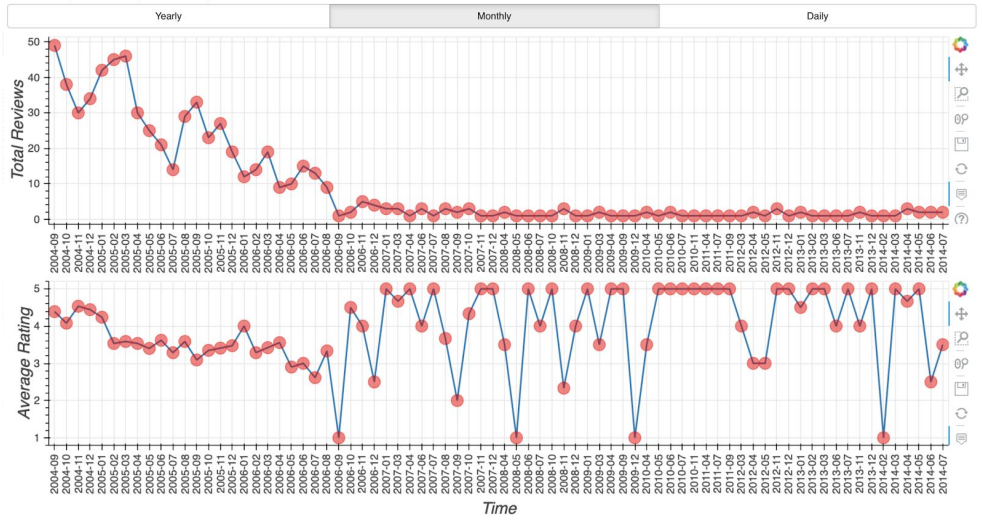
With respect to the number of products, this design scales efficiently. We can search for any products from 10k or 100k or 1M products and the design works. Data analysis might not be as straightforward as the data scales.

One issue which I faced with this design is the data cluttering, particularly in monthly and daily views, if there are a lot of reviews. This is evident in the images provided on the next page.

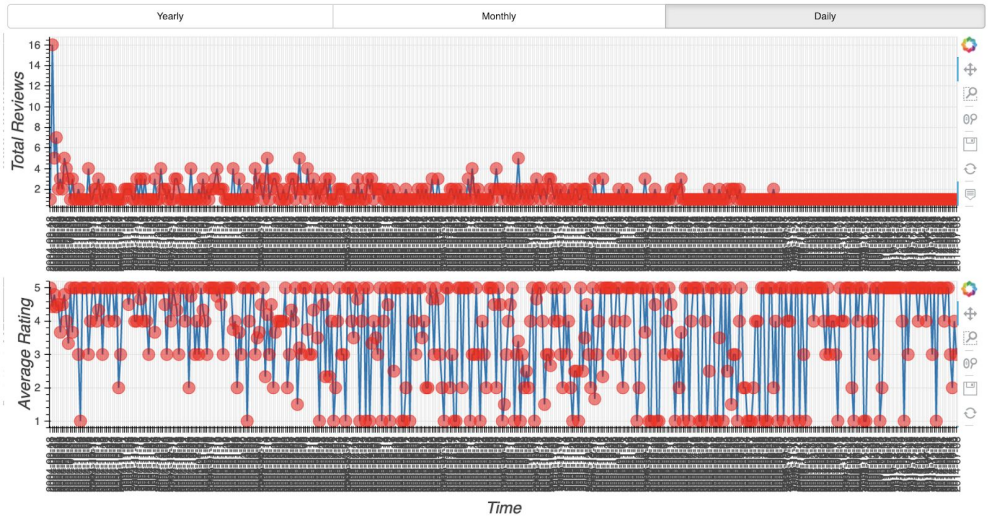
I attempted to fix this issue by creating a panning tool that lets you focus on a specific segment of the graph at any point in time. I was able to successfully implement this tool but was not able to properly switch between yearly, monthly, and daily tabs. A brief overview of this tool is provided in the next section.



Yearly



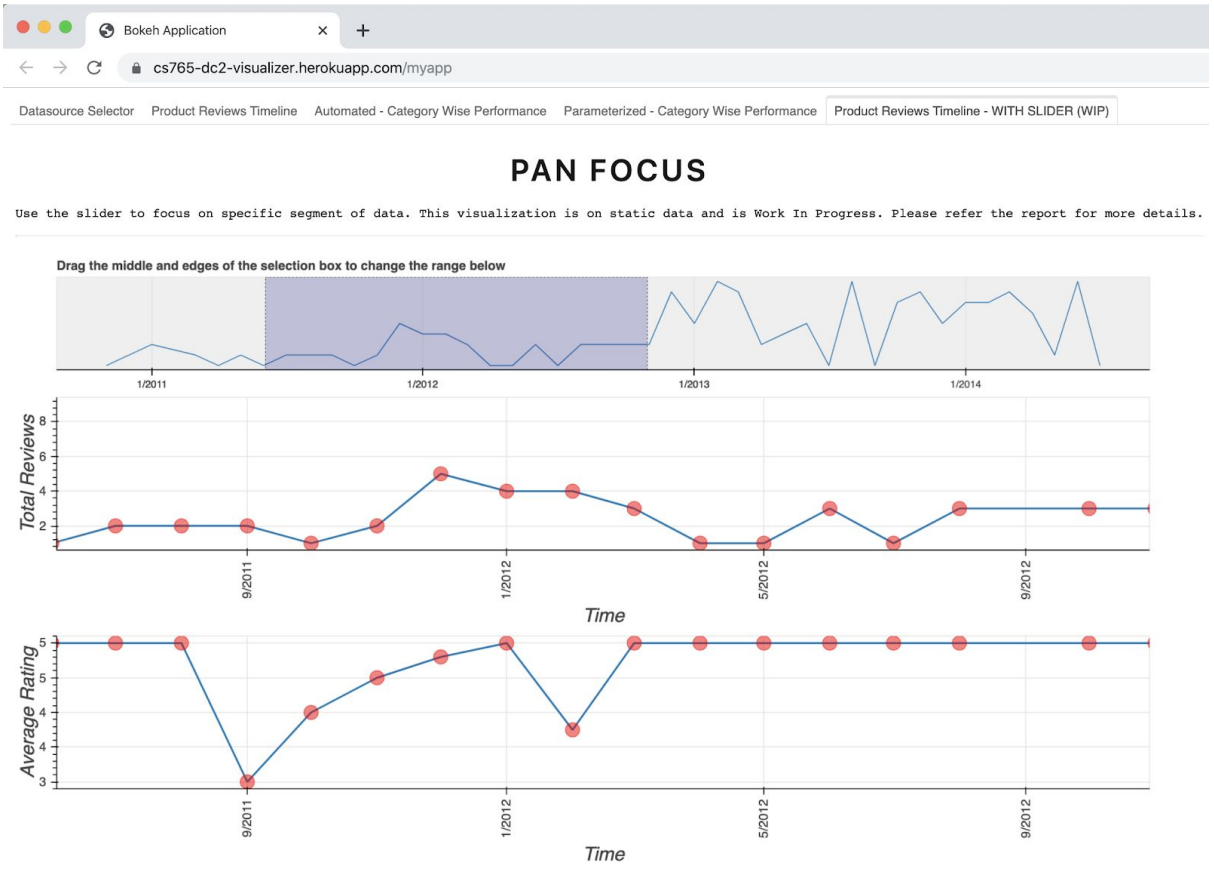
Monthly



Daily

5. Product Review Timeline (With Pan Tool) Tab

As discussed in the previous section, here is a snippet of the pan tool I created. I have added the working demo of this tool on the website using the product timeline of a particular product with a high number of reviews.



Uploading Custom User Data

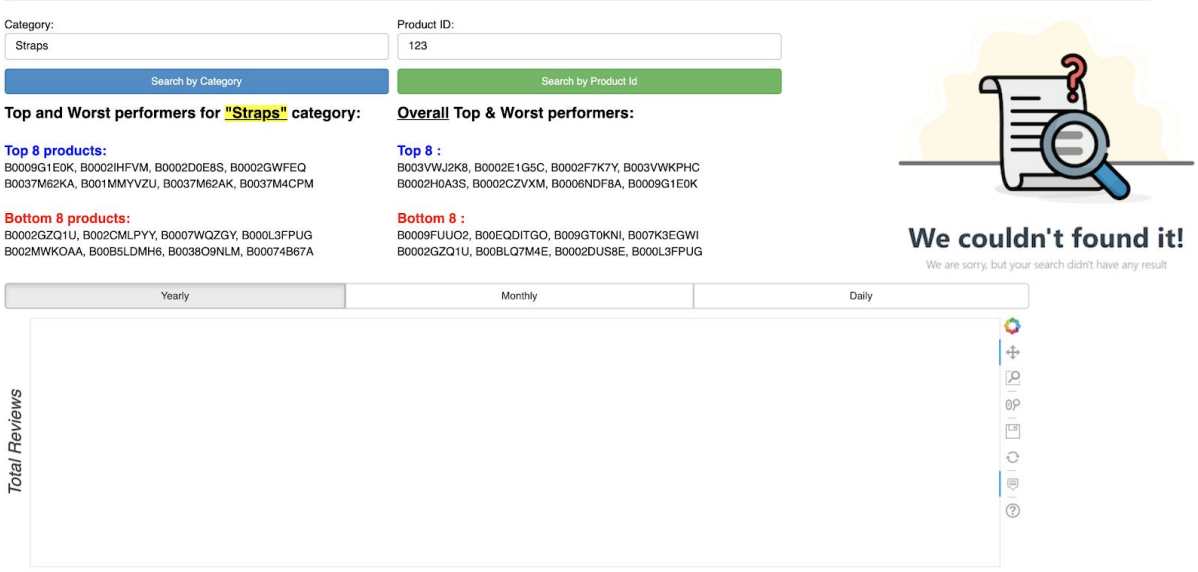
The application allows users to upload custom data and perform analysis on the same. Users need to upload 2 CSV files, review.csv, and metadata.csv

The column position of the CSV files is restricted. For instance, the first column of review.csv should refer to product id only. Not that column name has no restriction. Thus, the first column of review.csv can be named "ProductId", "ASIN", "Product Number", etc.

Here are some sample custom files which can be referenced to test custom uploader or generate your own data files.

Limitations

- As we discussed, the bar charts for “Top k Category” tab will not scale well, making automated analysis limited.
- With more review counts, the product timeline will not scale well.
- The application might perform slowly on larger data sets for two reasons.
    - It is not heavily optimized for large datasets.
    - I am using an unpaid Heroku account to run my application and hence have limited scheduling and resources available for the application.
- While I tried to make the UI as user-friendly as I can, there is a significant scope of improvement.

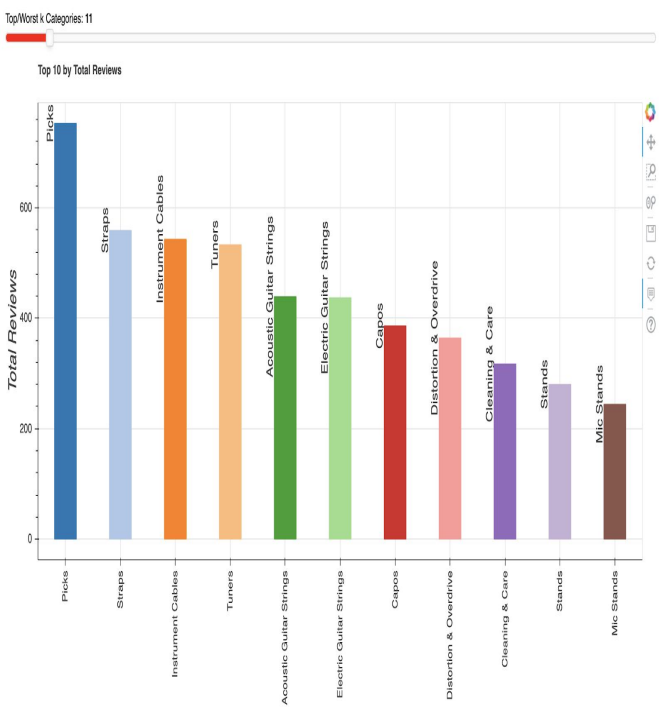


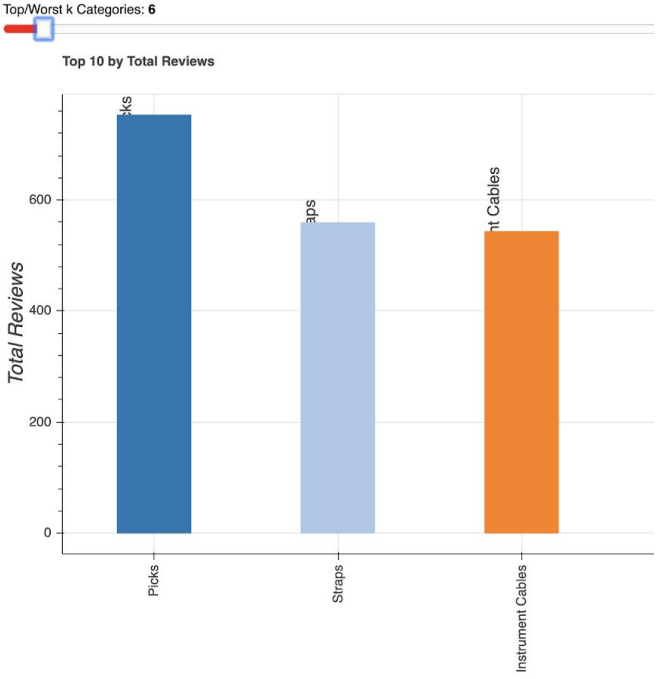
Graceful Error Handling (Product Not Found)

Challenges & Learnings

- I am not a regular Python developer and hence spent a good amount of time getting accustomed to Pandas Dataframe APIs.
- I spent over a day researching what all Python libraries can I use for this project. I shortlisted Plotly, Bokeh, and Dash.
- I finally decided to go with Bokeh as it is completely open-source and thus I will have a better support system if I get stuck.
- Also, I wanted to create a web server, which seemed the most feasible with Bokeh.
- As I worked on this project and discussed with my friends who were using Plotly, I figured out the following:
    - Bokeh is a low-level library, in the sense that it provides you building blocks and you can create visualizations the way you want.
    - Plotly, on the other hand, is a bit high-level library, as it provides many specific implementations like Scatter Plot, Linking, etc.

- I started off by creating a small visualization in Bokeh which lets you select multiple categories and plot them on a bar chart. This was just for a proof of concept that I will be able to work with this low-level library.
- I also spent some time setting up my application on Heroku.
- With respect to implementation, the integration of the pan tool was a rock-bottom point for me.
- I also tried to put labels on bars in the graphs, but could not display them properly at scale. I finally ended up removing them.





Future Work

1. Integrate the work in progress pan tool implementation with the timeline switch functionality.
2. Enhance product timeline visualization to scale with respect to automated analysis.

References

1. Bokeh: an interactive visualization library for modern web browsers
2. Deploy Bokeh server plots using Heroku