DATA440 Final Project: MoVIZ Visualization Tool

M.R. Sanford

Background

The rise of streaming and non-traditional media distribution models has led to an increasingly diverse film landscape. The shift has created a growing need for more accurate tracking of audience trends and preferences to better predict future project outlooks for the film industry.

MoVIZ provides a graphical user interface (GUI) enabling users to explore connections between various fil m characteristics and to uncover macrotrends. The project prioritizes up-to-date data on released movies available for exploration with a focus on quality over quantity to ensure meaningful analysis.

For data scientists and financiers: an everyday tool to visualize relationships and help prioritize film projects and marketing campaigns. For the everyday movie buff: a new way to discover new movies to add to their watchlists or a new way to uncover their viewing habits.

Data and Metadata

Data Acquisition

My pipeline ingests data from three sources which specialize in providing rating and financial data and movie metadata for the film industry: TMDB, IMDb, and The-Numbers.com.

The version of the TMDB dataset used for MoVIZ is available on Kaggle and is collected and regularly updated by u/asaniczka. TMDB is more comprehensive in its film metadata and foreign title offerings, which made it an optimal primary dataset with standard columns such as movie title, release date (in YYYY -MM-DD), runtime, genres, movie summary, vote average and vote count (format like IMDb's voting system), and general budget and revenue data. The raw CSV contains roughly ~1.2 million film titles. An advantage to TDMB was its similar column structure to IMDb, especially pertaining to the voting system because voting standardization did not require extra pre-processing steps. In addition, TMDB provides key features including a Boolean 'adult' column where 'True' indicates pornography. The qualitative 'status' column, another key feature, indicates the status of a film ranging from a value such as 'Rumored' to various production statuses ('In Production' or 'Postproduction') to the completed 'Released' status. The inclusion of these key variables was helpful in the data processing stage. The TMDB dataset also provided an 'imdb_id' column, which contained indices in the identical formatting convention as the IMDb dataset. This was a key column because it was used later for primary merging during the database construction stage.

The IMDb Dataset by Genre was collected by u/Chidambara Raju G on Kaggle; however, the initial data was collected by IMDb, a traditional film rating company. IMDb differs from TMDB on the number of available title offerings, which tend to favor American audiences and

filmmaking productions. The level of accessible data outside of IMDb's pay-to-access API limits is less than the available open-source movie metadata offered by TMDB. The total number of rows in the aggregated raw genre CSVs is roughly 368000 films; however, the dataset has similar columns including movie_id, which follows identical formatting as TMDB's 'imdb_id' column; as well as movie title, year (in YYYY format), genre, runtime, movie description, revenue, rating columns (average rating and number of votes). Thus, the IMDb dataset is intended to supplement the TMDB with key variables such as 'certificate', 'director' , and 'star' columns. The 'certificate' column contains age-based suitability ratings (e.g. G, PG, PG-13, R, NC-17). The 'director' and 'star' columns relate to a film's director and stars but exclude other roles such as production crew.

The-Numbers.com tracks box office revenue and film production financial data. The budget dataset for MoVIZ was self-collected directly from the website by web scraping financial data from all available table and totaling roughly 6200 movies. The table is used supplement with a more comprehensive breakdown with key features including Production Budget and film revenue indicators (e.g. "Domestic Gross" and "Worldwide Gross") in addition to movie title, release date (Month, DD, YYY format).

Pre-Processing

The variation of available titles and movie metadata column for each of the datasets demanded a significant amount of pre-processing to normalize observations before ingesting into the database. For example, the TMDB format was YYYY -MM-DD, the IMDb release year column was YYYY, and the budgets dataset is initially MON, DD, YYYY. The creation of the numerical 'year' column was a natural step in handling datetime format discrepancies between datasets. I was interested in exploring decade-by-decade trends, so all datasets have an added 'decade' column of string types based on the observation value of 'year.'

A 'normalized_title' column was implemented in all three datasets, which removed whitespace and lowercased the entire string. A combination of 'normalized_title' and 'year' resulted in the 'normalized_title_year' column, which was created as a fallback ID system for merging the movies on the 'movie_id' in IMDb and its TMDB 'imdb_id' equivalent. The combined 'normalized_title_year' was also used for all three datasets in dropping duplicate rows. The final step for the three datasets included standardizing column names for continuity and database creation, and remapping for easier CSV handling upon outputting.

Regarding more specific dataset cleaning steps in TMDB, unused columns containing movie posters and official movie homepage links were immediately dropped because of lack of relevance. Any row where the movie title was blank was removed before the movie title and year normalization and the creation of the fallback merge key, outlined above.

In the normalized numerical 'year' column, there were instances of typos (e.g. a film's release year was 1837 instead of 1937) that both predated the first recorded motion picture, *The Roundhay Garden Scene* (1887)[1], and postdated the current date (e.g. a movie's year listed

[1] Wikipedia Contributors. "List of Cinematic Firsts." Wikipedia. Wikimedia Foundation, November 19, 2019. https://en.wikipedia.org/wiki/List_of_cinematic_firsts.

at 2057). I implemented a floor limit to eliminate any rows where the year preceded 1880, which was the same decade as the first motion picture; on the other end, I placed a ceiling to prevent any movies released after 2025. This justification was implemented in the IMDb 'year' column too. Future implementations may change the floor or ceiling values to include a wider range of movie release years, but users must be aware of an increased probability of raw data typos at the range's extreme values.

Within TMDB, I added a conditional filtering clause to remove rows where 'runtime', 'revenue', and 'budget' observations in the row were 0.0 or NaN. The justification for the move: an observation lacking a runtime, revenue, and budget is unlikely to contribute meaningful insight to visualization without some of the core quantitative indicators for an individual film. Outlined in Section 2A, TMDB's 'status' column contains qualitative data regarding the release status of a film; the dataset was filtered to only include films where 'status' == 'Released' to avoid noise from speculative or incomplete productions. MoVIZ's goal is to provide visualization support for released movies only. Also mentioned earlier, pornography in the 'adult' column was filtered out of the dataframe; however, future instances of this project could visualize with the 'adult' characteristics included.

The IMDb dataset had similar unnecessary columns as TMDB including movie poster and official movie links, and IMDb's identification system for unique directors and stars, all of which were dropped immediately. I followed similar pre-processing steps to normalize the movie titles and drop duplicate rows on the movie_id (TMDB's imdb_id equivalent) and then falling back on combination normalized movie title and year. Reformatting a release date column was unnecessary since a 'year' column was included within IMDb; however, I implemented the floor/ceiling filter from TMDB for valid years.

IMDb's 'certificate' column required equivalency matching foreign and television age certifications to the MPAA standards (e.g. G, PG, PG-13, R, NC-17) [2]for uniformity. Rows with television ratings were not automatically dropped because some films were aired on TV rather than a traditional theater, but I opted not to disqualify their inclusion. Some rows with obscure or irrelevant ratings (such as 'T', a videogame rating) were dropped to maintain the integrity of a film dataset.

The budgets dataset underwent the same pre-processing steps as listed above related to normalizing titles and years, grouping decades, dropping duplicates based on normalized title and year, implementing the 'normalized_title_year' id conventions for database merging. The-Numbers dataset included several movies with release dates postdated after April 2025; as a result, I implemented a dynamic fix which handles available movies in relation to today's date. Thus, the code should be able to handle future added film financial data seamlessly.

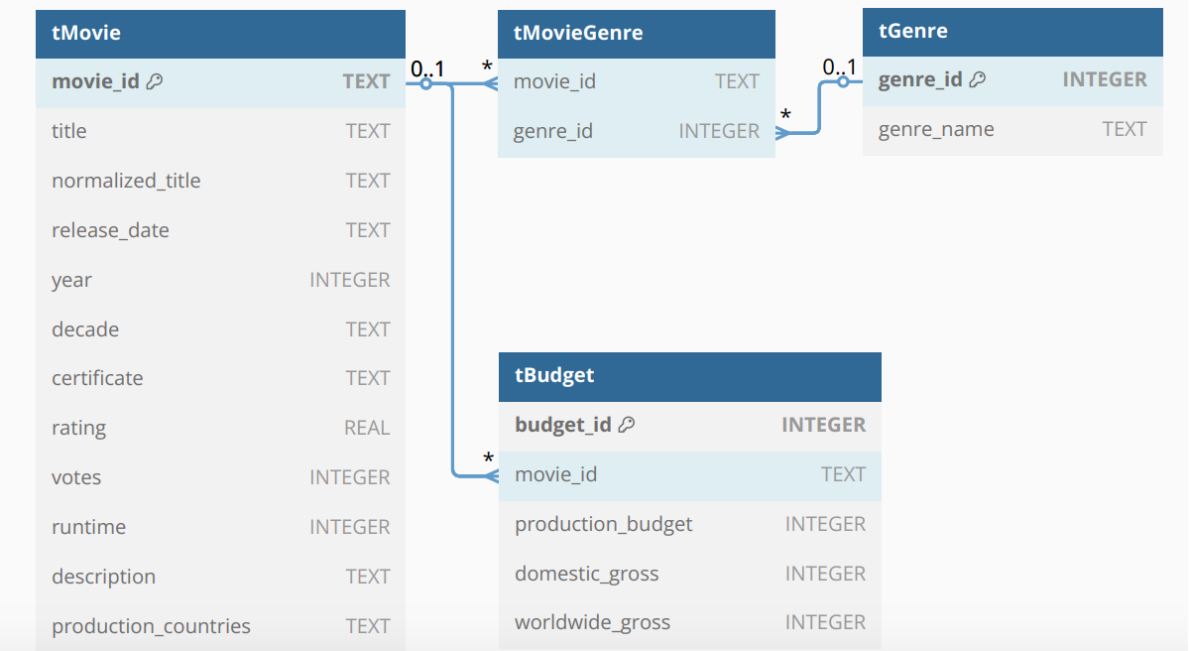
Merging

To ensure comprehensive data availability and streamline integration into the database, the merging process involved joining the movie_ids for both TMDB and genre

[2] Rating System Wiki. "Motion Picture Association," n.d. https://rating-system.fandom.com/wiki/Motion_Picture_Association.

datasets. Additional merging steps included creating a genre_id system and constructing a movie-genre pivot table to manage many-to-many relationships between movies and genres. The merging was completed in multiple steps: preliminary merging for movie_id matches in both datasets, secondary 'merging' on normalized_title and year for movie_id values where availability was limited to one category (e.g. only TMDB but not genres). In the secondary step, if there was a movie id in either of the columns that did not exist in the other column, I created the normalization and added the movie_id to the missing observation. A similar framework was implemented for the budget's dataset, where I populated movie_ids based on the merged datasets.

Relational Database Diagram



Above is a diagram of the MoVIZ relational database, which stores and organizes related movie information for querying and visualization.

Libraries and Tools

The third-party libraries used included the kagglehub package[3], which allows public dataset hosted on Kaggle to be downloaded without the use of a traditional API. I opted for kagglehub instead of a traditional API for the ease of integration for the downloading portion of my pipeline. I employed tqdm [4] to visually track the progress of the Kaggle dataset downloads and the web scraping portions, which I thought was an important library for improved user functionality and progress tracking.

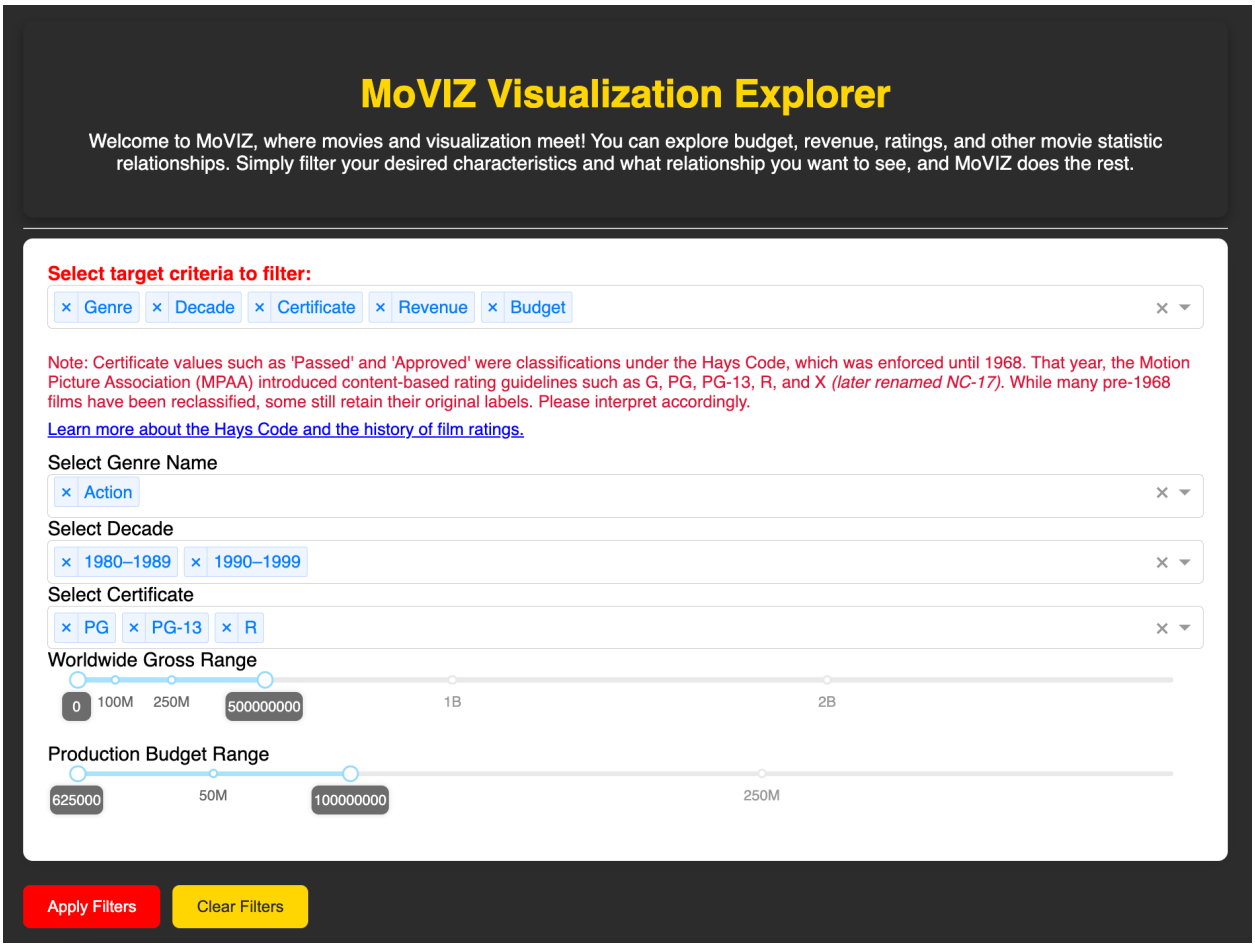
Regarding tools, I implemented a logging system using Python's built-in logging module where I created a setup_logger() function to create custom-named loggers that write timestamped entries to .log files in the dedicated logs directory. Each logger outputs messages in a systematized format and log level related to the portion of the pipeline being executed. This system allows for easy review; additionally, I implemented a clear logs() functionality to allow the reset of all .log files' contents to allow for resetting between runs. I opted for this system instead of less integrated print statements to better track progress within each stage of my pipeline.

Demonstration

I used Dash for the visualization deployment because the plotly capabilities were important to my product (the GUI application). I have used the tool to visualize various trends such as certificate rating over time– have the number of PG-13 and rated R movies increased in quantity over time? Another example was examining production budget vs worldwide gross of romance movies from the 1990s and 2000s. I expect other users of this tool may do similar analyses with the dashboard. MoVIZ is designed to allow users to explore and analyze my cleaned film data via GUI application implemented in Dash. They can filter movies by characteristics including genre, decade, certificate, rating, revenue (based on worldwide gross), and budget.

First, the user selects their desired filter fields, then each selected filter's respective dropdown (for qualitative data) or dual slider (for quantitative data) appears below the dropdown section for further filter refinement. The user may select as many or as few filters as desired; however, any unselected filters selected in the first dropdown 'stage' will automatically default to include all available data. For an example, I will explore 1980s and 1990s action movies rated PG, PG-13, and R with a budget between $0-100 million, and a worldwide gross between $0 and $500 million. I will not set a required Rating range.

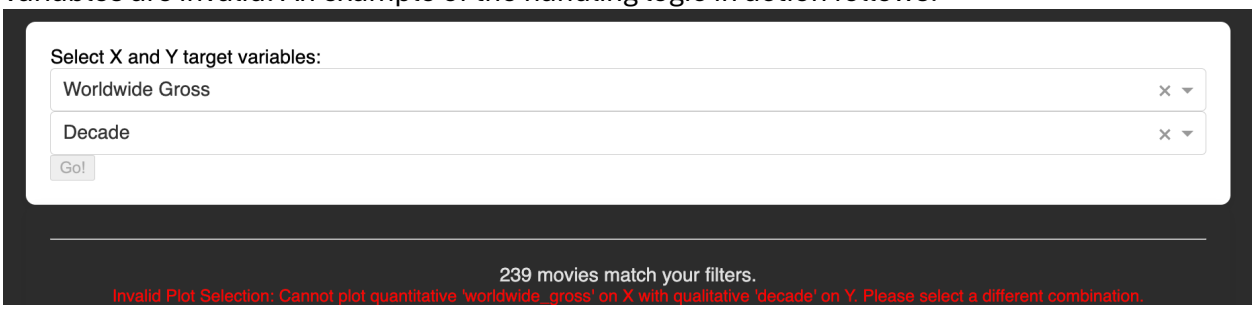
[3] PyPI. "Kagglehub," April 23, 2025. https://pypi.org/project/kagglehub/.
[4] Costa-Luis, Casper da. "Tqdm Documentation." tqdm.github.io, October 11, 2015. https://tqdm.github.io/.



The user has the option to 'Apply Filters' to apply their changes or to 'Clear Filters,' which will do the opposite. After selecting 'Apply Filters', the user would receive a message if the filters applied successfully. The number of filtered movies will also appear as text above the visualization as "{n} movies match your filters. {n} plotted on the " Then the user will be able to select which features to target for visualization as x and y variables.

The application is intuitive in automatically adjusting and creating a plot based on the type of data selected. For example, quantitative vs quantitative will return a scatter plot, quantitative vs qualitative will return a bar chart, and qualitative vs qualitative will return a stacked bar chart. I have implemented handling to alert a user when the ordering of x and y

variables are invalid. An example of the handling logic in action follows:



The application recognizes the invalid combination and ‘greys out’ the Go! button intuitively to prevent error.

Examples of Visualization Outputs