Analysis on 97th Academy Awards

Chiara Genuardi, Giovanni Noè

Abstract

Each year, thousands of films are released worldwide. Only a portion of them makes it to a wider audience, and just a select few are honoured with an Oscar nomination.
The Oscars recognize their winners' outstanding achievements in cinematography and are the most prestigious awards in the film industry.
Our project aims to create a multi-purpose dataset containing information about movies released in 2024 while also providing insight into Oscar nominations and winners. Then we perform some queries on the dataset to answer questions about the popularity of the movies with reference to reviews, Oscar nominations, Oscars won and number of mentions in some relevant Reddit posts.

Introduction

The Academy Awards, commonly known as “the Oscars” [1], are the most prestigious awards for artistic and technical merit in filmmaking. Each year, between February and April, the Awards are assigned by the Academy of Motion Picture Arts and Sciences (AMPAS) in the USA, recognising excellence in films released during the previous year [2].
Oscar nominations are determined by members of the AMPAS, with each branch (e.g., directors, actors, writers) voting for their respective categories. Final winners are selected through a second round of voting, in which all Academy members can vote across most major categories. Some special Awards—such as the Honorary Award or certain short film and documentary categories—are decided by dedicated committees within the Academy.
The nominations for the 97th Academy Awards were officialised on January 23, 2025, while the award ceremony took place on March 2, 2025, in Los Angeles and assigned the Oscars for the year 2024 in 23 categories [3].
In 2024, thousands of movies were released worldwide, 50 of them received at least an Oscar nomination and only 14 actually won an Award.
Our project follows every step of the typical Data Science pipeline to gradually build a structured and versatile dataset focused on movies and Oscars for the year 2024. The dataset is designed to provide both simple movie-related information and enable more advanced statistical analysis.
The data acquisition phase combines automated API calls (e.g., TMDB API) with web scraping techniques (from Wikipedia). These diverse data sources are then cleaned, processed, and integrated into a single relational schema.
Most of the process, from data acquisition to storage, is carried out using the Python programming language within a single Jupyter Notebook, making use of various Python libraries (listed in the References section [4]-[15]).
The final part of our project focuses on querying the dataset to extract insights about films and Oscar nominations. Since the data model chosen is the relational one, the queries are written in SQL and executed using DB Browser for SQLite [16].
Ultimately, our project demonstrates how even a cultural event like the Oscars can serve as a case study for building a complete and analysable data product, from raw sources to structured relational storage and querying, using standard Data Management techniques.

1. Data Acquisition

The first step of our work consists in selecting appropriate data sources and gathering the necessary information. The types of data we aim to collect include:

- general information about movies released in 2024 (such as title, genres, actors, directors, etc.)
- indicators of popularity and public reception (e.g., ratings, reviews, and social media discussions)
- the complete list of Oscar nominations and winners for each award category

To collect this data, we identify four main sources: TMDB, OMDb, Reddit, and Wikipedia. The first three provide APIs for structured data access, while for Wikipedia we use web scraping to extract the relevant information from the 97th Academy Awards page.

1.1 API

An API (Application Programming Interface) is a set of rules and protocols that allows different software applications to communicate with each other. In the context of data acquisition, APIs enable programs to request and receive specific pieces of data from external services with a quick and efficient data transfer. We collect data from TMDB, OMDb and Reddit through API requests.

TMDB

TMDB (The Movie DataBase [17]) is a popular online database that provides detailed information about films, TV shows, and people in the audiovisual industry and is our main source for general information about movies. It offers a free and well-documented API [18] that allows developers to access such data.

The TMDB API includes several endpoints for accessing movie-related information. The one that returns the most comprehensive data for each individual movie is `/movie/{movie_id}`. However, this endpoint only works when the internal TMDB ID of the movie is already known.
To retrieve these IDs, we first query another endpoint: `/discover/movie`, which supports advanced filtering options and returns a list of movies along with their internal IDs. Specifically, we filter by release year (selecting only movies released in 2024) and by the number of votes received on the platform (greater than 24). The vote threshold is necessary due to restrictions in the OMDb API, which will be used later. However, after further inspection, we find that some of the movies nominated for the Oscars, such as documentaries and short films with a low `vote_count`, are missing from the results. To include them, we have to manually retrieve their IDs by inspecting the HTML source code of their respective TMDB web pages.

Once the internal IDs are retrieved, we perform individual queries to the `/movie/{movie_id}` endpoint for each film to obtain detailed data. The responses are returned in JSON format and include the following attributes:

- *adult*: true if the movie is for an adult public
- *backdrop_path*: path to the background image
- *belongs_to_collection*: the collection the movie belongs to
- *budget*: production budget
- *genres*: associated genres
- *homepage*: official website
- *id*: internal id we used to perform the query
- *IMDB_id*: ID of the movie on IMDb (used as universal ID)
- *original_language*: language in which the movie was originally produced
- *original_title*: title in the original language
- *overview*: short summary of the plot
- *popularity*: metric based on user interactions on the TMDB website (e.g., page views, votes, watchlists, reviews)
- *poster_path*: path to the official poster image
- *production_companies*: companies involved in production
- *production_countries*: countries where the production took place
- *release_date*: date
- *revenue*: total income generated by the movie
- *runtime*: duration of the movie in minutes
- *spoken_languages*: languages spoken in the movie
- *status*: release status
- *tagline*: catchphrase associated with the movie
- *title*: English title
- *video*: official video or trailer if available
- *vote_average*: average user rating on TMDB (scale 0-10)
- *vote_count*: number of votes received (used in the initial filtering)

OMDb

The second data source of our project is OMDb (Open Movie Database [19]), an online database that provides data about movies and TV shows, including ratings, plot summaries, cast, and production details. It offers a free API [20] that allows access to data one movie at a time, with a limit of 1000 requests per day. This limitation is the reason for the *vote_count* constraint added in the TMDB query: although TMDB returned data for tens of thousands of movies released in 2024, we needed to restrict the selection to a manageable number of titles (under 1,000) to ensure compatibility with OMDb's daily API quota. By applying this filter, we reduced the results to approximately 1000 movies.

To ensure consistency between the two dataframes originating from the two data sources, we performed the query to the OMDb API using IMDb IDs from the TMDB data, obtaining a response in JSON format and the following attributes for each movie:

- *Title*: the title of the movie
- *Year*: the release year
- *Rated*: the movie's rating by the Motion Picture Association (e.g., PG-13, R, etc.)
- *Released*: the release date
- *Runtime*: the duration of the movie in minutes
- *Genre*: associated genres
- *Director*: the director(s) of the movie
- *Writer*: the writer(s) of the movie
- *Actors*: the main actors
- *Plot*: a brief description of the plot
- *Language*: the language in which the movie was produced
- *Country*: the country of production
- *Awards*: awards won by the movie (not only Oscars)
- *Poster*: URL to the official poster
- *Ratings*: ratings from Metacritic, Rotten Tomatoes and IMDb in a dictionary
- *Metascore*: the Metacritic score
- *IMDbRating*: the IMDb rating (on a scale from 0 to 10)
- *IMDbVotes*: the number of votes received on IMDb
- *IMDbID*: the unique IMDb ID
- *Type*: the type of content (e.g., movie, series, episode)
- *DVD*: the DVD release date
- *BoxOffice*: earnings at the box office in the US
- *Production*: production companies involved
- *Website*: the official website of the movie or show

Naturally, there are several attributes in common with TMDB, but this is an issue we will address in the Data Processing section. Additionally, some of the movies retrieved from TMDB cannot be found in OMDb; however, upon manual inspection, we observe that these titles were very little-known and not particularly relevant to our case study.

Reddit

Reddit is a social media platform and online community where users can share content of any kind, ask questions, and discuss topics within themed forums called “subreddits”. It covers a wide range of interests, including movies, and offers public user-generated data, such as comments and post scores, that can be accessed via API for data analysis.
In our case, the data acquisition procedure for Reddit differs significantly from those used for TMDB and OMDb. Instead of making conventional API requests manually, we employ a Python library called PRAW (Python Reddit API Wrapper) [9], which simplifies and automates API interactions with Reddit.
We collect only the text of comments from three relevant posts found in the subreddits r/oscars [21], r/popculture [22] and r/movies [23], retrieving approximately 35000 comments.
Since we aimed to extract all levels of conversation (comments, replies, replies to replies, etc.), the code for this part is computationally intensive and takes a long time to run. For this reason, we execute it once (15th of March 2025) and store all the collected comments as a list of strings in a *.pkl* file using Python’s pickle library [11].
Therefore, this data will remain formatted as a list of strings, ready for further processing in the next steps of the pipeline.

1.2 Web Scraping

Web scraping is the process of automatically extracting data from websites by parsing their HTML structure. This technique is often used when information is available on a public web page but not accessible through an official API and that's exactly the case of data about 2024 Oscar nominations and winners.

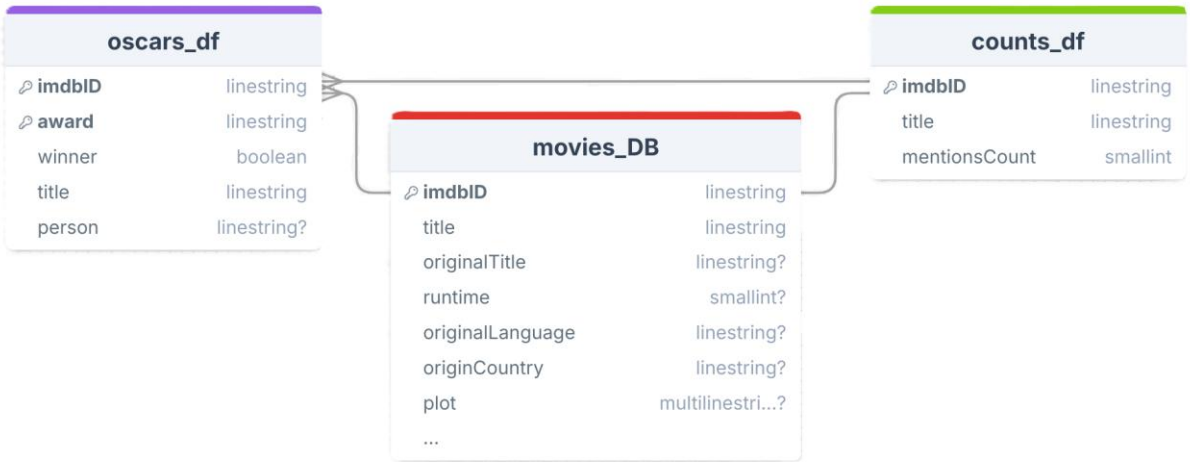
Wikipedia - 97th Academy Awards (Oscars)

The data we need about Oscars can be easily accessed in the 97th Academy Awards page on Wikipedia [2], which represents our last data source. By inspecting the HTML source code of the web page and the BeautifulSoup Python library [7]**,** we develop a script that automatically scrapes data about nominations, winners and people involved in the Awards (e.g. actors nominated for "Best Actor in a Leading Role").
Then we store the extracted data as a list of dictionaries, where each dictionary corresponds to the nomination of a specific film for a specific Award category.

2. Data Modelling

A crucial step in every Data Science pipeline is Data Modelling, which involves organizing and structuring the collected data in a way that supports efficient analysis and querying.
After gathering data from multiple heterogeneous sources, it becomes essential to define a coherent and unified schema.
In our project we choose the relational model due to its clear structure, simplicity, ease of data integration and strong support for structured queries using SQL. Additionally, since each of our datasets have less than 1000 rows, there is no need to use different data structures that fasten the queries, SQL queries (at this scale) are fast enough.
Finally, adopting a relational model allows us to use the Pandas library [8], which is very well-documented, widely used, and introduces the DataFrame data type, a data structure that closely resembles relational tables.
With this in mind, we can now design the schema for our dataset:

- the first and most important table will be *movies_DB*, containing general information about movies. It will be created by enriching data from TMDB with data from OMDb (one row per movie)
- the second table, *oscars_df***,** will contain Oscar nominations and winners, with one row for each nominated film in each category
- the third table, called *counts_df***,** will store the number of times each nominated film is mentioned in Reddit comments (more details in the Data Processing section)



At this point, it is important to note that we are not yet building the actual relational tables. For now, we create three separate Pandas DataFrames: one fromTMDB data, one from OMDb data, and another for Oscar-related data. These will be processed and integrated in the next steps of the pipeline.

3. Data Processing

One of the most relevant stages in Data Science pipelines is Data Processing; it is a step which includes cleaning, analysing and organizing data to make it useful and meaningful, transforming raw data into actionable insights. This also ensures minimality, usability and readability.

TMDB

The TMDB dataframe is the first source of information for our project, supported by the OMDb dataframe; we aim to keep most of its information and then enrich it with OMDb.

At first we reduce the dimensionality of the data set by deleting non-useful columns, both because we keep some information from another data set or because some aspects are not analyzed; in particular we delete:

- *adult*, *backdrop_path*, *belong_collection*, *production_companies*, *production_countries*, *poster_path*, *spoken_languages*, *status*, *tagline*, *video* for non-relevance
- *id* since we want to keep as a key value of movies the id from IMDb
- *genres* because we hold this information from the OMDb dataframe. Even though TMDB dataframe is our first source, the reason behind this choice lies in the way this information is stored: in this one *genre* is stored in a dictionary using a numerical classification, while in OMDb data is easily converted into a list type and the classification is more explicit

We replace all missing values with the *nan* type of NumPy [13] for consistency; for numerical variables (*budget*, *revenue* and *runtime*) another inconsistent value is 0, which we replace with the *nan* type as well.
All variables have consistent data types except for *release_date*, which we convert to format date time with the pandas' function.
We now rearrange column positions based on the importance of such variables and at last change column names to ensure usability and consistency along all dataframes.

OMDb

From the *Ratings* column we extract the information related to Rotten Tomatoes and rescale it to the range 0-10 as all other ratings to enable comparison.
We delete non-useful columns following the same principles used for TMDB, in particular:

- *Year* since we already have this information from *release_date*
- *Runtime*, *Released*, *Plot*, *Language*, *BoxOffice* because we hold this information from the TMDB dataframe
- *Writer*, *Country*, *Poster*, *DVD*, *Type*, *Production*, *Website*, *Response*, *Season*, *Episode*, *seriesID* for non-relevance
- *Ratings* since we already extracted relevant information from it

We replace all missing values with the *nan* type of NumPy.
We check all variables' data types and find out all of them are objects, so we convert *IMDbRating*, *IMDbVotes* and *Metascore* to float and moreover scale *Metascore* to the range 0-10 to enable comparison. We convert *Genre* to a list to make it usable. The variable related to MPA classification holds information both about USA classification and TV classification, which we decide to keep because of some different shades, although we merged non-rated films of both classification in just one value '*Unrated*'.
Then we rearrange column positions and change their names accordingly.

Reddit

Since Reddit comments serve a different purpose, the processing of this data differs as well. Our goal is to create the *counts_df* DataFrame, as mentioned in the Data Modelling section. To achieve this, we need to count how many comments mention each nominated movie.
To do so, we construct a dictionary where the keys are the titles of the nominated movies, and the values are lists of words associated with each movie. These include:

- titles and original titles from TMDB DataFrame

- full names of actors and directors from the OMDb DataFrame
- full names of Oscar-nominated individuals (e.g. Best Actor/Actress nominees) if not already present in the OMDb data
- additional terms such as abbreviations of titles, titles with common typos, standalone first or last names (for actors commonly referred to by just one), and context-specific words (e.g., “Latvia” for the film Flow)

Once the dictionary is complete, we load the Reddit comments from the *.pkl* file and iterate through them. For each comment, we check whether any of the words associated with each movie appear, using a regular expression [12] composed for each movie. If a match is found, the movie’s mention count is incremented by one.
Then we create the *counts_df* DataFrame, which contains a row for each movie and title, the count as *mentionsCount* and *IMDbID* as attributes.
It is worth noting that our dictionary covers only a subset of possible terms related to each movie, so some relevant comments might not be captured. However, we took care to improve accuracy by making the regular expressions case-insensitive and ensuring that matched terms appear as whole words, not as parts of other words (e.g., “Alien” would not match “alienating”).
This approach allowed us to associate a measurable level of public attention to each nominated film based on Reddit discussions. While the method is not exhaustive, it provides a useful approximation of audience engagement and serves as a valuable additional feature in our dataset. Moreover, once the initial extraction is complete, the counting process is lightweight and computationally inexpensive, making it easily reproducible or extendable in future analyses.

Oscar

Since we created the Oscars dataframe there’s not much processing to do; we add a column with the ID of IMDb to be used as a key value for possible comparisons with other dataframes.

4. Data Quality Assessment - Before Enrichment

At this point of the project we want to evaluate the quality of our data. It is a crucial step, since it affects the accuracy and reliability of the products of the project, directly influencing decision making and operational efficiency.
As we said we took care about the usability, readability and minimality of our data during data processing, by keeping the same information only from one source, fixing ratings variables in the same range and converting variables to the correct data type.
The most relevant dimensions to be analyzed now are accuracy, minimality, consistency and completeness.
We runned our code various time from when we started our project (5th of March 2025) to the 6th of April 2025 and we noticed that around ten days after the ceremony data stabilized and hasn’t had any major changes; this to say, temporal dimensions are not relevant to be analyzed considering how we decided to conduct this project.
Also, we won’t perform improvements in this session but after enriching our dataframe.

4.1 Accuracy

Accuracy refers to the correctness and reliability of the data values. In this project, accuracy is assessed by identifying missing values and evaluating their impact on the overall analysis to determine whether the corresponding records should be retained or removed.
We check if there is any missing value in the TMDB and OMDb dataframe; we already know that the Oscar data frame has some missing values for the variable *person* for construction.
The TMDB dataframe has 1 missing value for *IMDbID* variable, 728 for *budget* and 625 for *revenue*.
The OMDb dataframe has more missing values, 3 for *director*, 11 for *actors*, 459 for *awards*, 25 for *IMDbVotes*, 140 for *IMDbRating*, 641 for *metacriticRating*, 420 for *rottenTomatoesRating*, 436 for *mpaFilmRating* and 1 for *genre*.
We are not surprised by those high numbers: since we are now analyzing all films released in 2024, we are considering a lot of minor films, and for this reason the source may have decided not to spend time by adding some information (such as *budget*, *revenue*, *actor*, *genre*), few people may have watched them which result in few or no ratings / reviews and at last missing values for the variable awards may be simply the result of no nominations.

4.2 Minimality

Minimality ensures that the dataset does not contain unnecessary or redundant information. This is verified by checking for duplicate rows based on the combination of *title* and *IMDb_ID*, which uniquely identifies a film. For minimality we address the problem of duplicates and check if for the pair *(title, IMDbID)* we have some duplicates both in TMDB and OMDb; we find out that in both dataframes the film 'Anuja' appears two times.

4.3 Consistency

Consistency measures whether data values follow expected formats and ranges. In this context, consistency is evaluated by analyzing numerical variables to detect outliers or values that fall outside a plausible range. Since we are analyzing a lot of numerical variables we want to make sure that they don't span in an improbable range, in particular it would be problematic to have negative values, a runtime too high or ratings which don't span in a 0-10 range. For this reason for every numerical variable in TMDB and OMDb dataframes we print both the minimum and maximum value and we don't obtain any improbable value.

4.4 Completeness

Completeness represents the extent to which all required data is present. It is assessed by checking whether all Oscar-nominated films were included in the TMDB and OMDb datasets, and by identifying missing values for key variables across these sources.
We check if all nominee films appear in TMDB and OMDb dataframes, in TMDB we have all of them while in OMDb 'I am ready, Warden' and 'Yuck!' are missing. After another check we find out that none of them has won any award. We also define a variable which stores how many missing variables we have for any nominee in both DataFrames and can see that some movies have some missing values, for TMDB generally *budget* and *revenue* as expected from the analysis of missing values while for OMDb we have missing values for *actors*, *awards* but mostly for *ratings* and *votes*.

5. Data Enrichment

To enhance and expand the information available in our main dataset derived from TMDB, we implemented a Data Enrichment phase. This process involves supplementing the existing data with additional information from external sources—in this case, the OMDb database. The goal of data enrichment is to increase both the quality and utility of the dataset by creating more complete and informative profiles for each movie. We performed this enrichment by merging the TMDB dataframe with the OMDb dataframe using the *IMDbID* field as the common key. Specifically, we used the merge function with a left join, keeping the TMDB dataframe as the base and appending the corresponding data from OMDb wherever a match was found. This resulted in a new, enriched dataframe with a shape of (998, 23), which retains all the original TMDB entries while adding valuable contextual details.
Regarding the Oscars dataset, we chose not to merge it directly into the main dataframe. A full merge would have led to numerous duplicate entries and a significant number of missing values, due to differences in granularity and data completeness. Instead, in earlier stages of processing, we ensured that the IMDbID field was included in the Oscars dataset. This allows us to perform targeted joins or comparisons when needed, without compromising the integrity of the main dataset.

6. Data Quality Improvement

We decided to perform data quality improvements after data enrichment to avoid unnecessary work on data we may have ended up not storing.
Taking into account the results of the Data Quality Assessment done before Data Enrichment we improve the quality of the *movies* dataframe.

Accuracy

We delete rows which have a null *IMDbID*, as we know that none of them are nominees.

Minimality

We delete all duplicates on the pair (*title*, *IMDbID*).

Completeness

We saw that in the OMDb dataframe were missing two nominees, 'I am ready, Warden' and 'Yuck!' so we searched and added by hand the information found, in particular we were able to find for 'Yuck!' the number of wins and nominations, *IMDbVotes* and *IMDbRating*.

7. Data Quality Assessment - After Integration and Enrichment

After having performed Data Quality Assessment, Data Enrichment and Data Quality Improvement, we carry out another audit on Data Quality, following the same procedure of Section 4 on our merged dataframe.

7.1 Accuracy

The movies dataframe has 16 missing values for *runtime*, 740 for *budget*, 8 for *director*, 16 for *actors*, 470 for *awards*, 28 for *IMDbVotes*, 150 for *IMDbRating*, 661 for *metacriticRating*, 433 for *rottenTomatoesRating*, 451 for *mpaFilmRating* and 6 for *genre*.
We expected this result from previous analysis.

7.2 Minimality

Using a for cycle we find out that we don't have any duplicates.

7.3 Consistency

Here we print the ranges in which every numerical variable spans to see if it is improbable; all ranges turn out to be realistic.

7.4 Completeness

At last, we observe that all nominee movies are present in the movies dataframe; for each of them, we also checked the number of missing values. We can see that most of the films with missing data are nominees in the categories of Best Short Film (Live Action), Best Animated Short Film, Best Documentary Short Film, and Best Documentary Feature Film. This makes sense, since these types of films often have limited releases, lower budgets, and less mainstream exposure, which typically results in less comprehensive metadata being available through public databases like TMDB and OMDb. Consequently, information such as box office figures or full cast lists data may be incomplete or entirely unavailable for these entries.

8. Data Exploration

Data Exploration is the process of examining and analyzing a dataset in order to gain a deeper understanding of its structure, content, and underlying patterns. This phase typically involves generating descriptive statistics, visualizations, and summary tables to identify relationships between variables, spot potential outliers, and reveal trends or distributions of interest.
Since the purpose of our project is to build a structured dataset rather than perform complex data analysis or machine learning, the Data Exploration phase focuses on providing basic statistics and creating visualizations that offer meaningful insights into the data we have collected and processed.

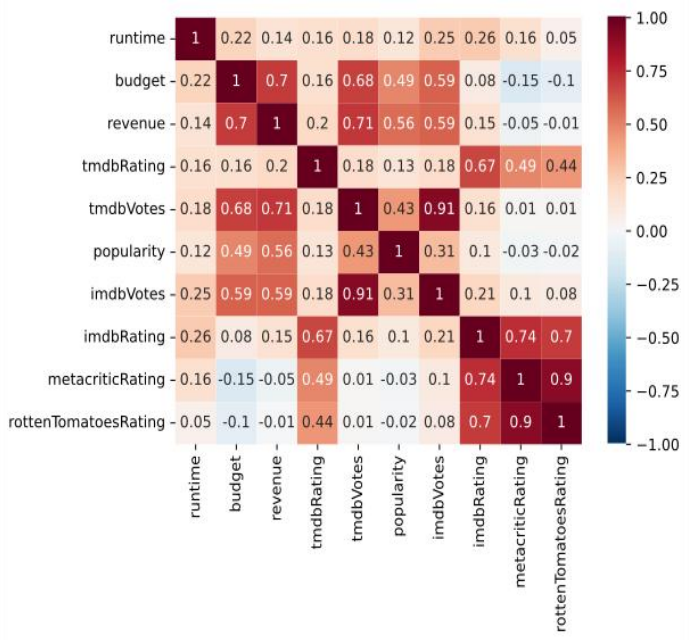
8.1 Statistics

In this subsection, we present some basic statistics computed on the dataset to summarize key aspects of the collected data. These metrics help highlight general trends and provide a first overview of the dataset's structure and content.
We begin by computing descriptive statistics for numeric variables, including the mean, standard deviation, and selected quantiles. Here are some interesting insights that emerge from this analysis:

- the longest movie in the dataset has a duration of 247 minutes, while the average runtime is 101 minutes
- the median movie budget is around 15 million dollars, whereas the median revenue is less than 8 million dollars, so the expenditure appears to be higher than the income

- in contrast, the average budget is approximately 36 million dollars, and the average revenue is around 64 million dollars
- among the platforms considered, Rotten Tomatoes shows the highest variability in user ratings, with a standard deviation of 2.54 and ratings ranging from 0 to 10, the extremes of the scale
- Metacritic and IMDb ratings share similar distributions, with comparable medians, minimums, and maximums. However, Metacritic ratings appear to be more evenly distributed across the scale
- TMDB is the platform with the lowest rating variability, as shown by its smaller standard deviation and less extreme minimum and maximum values

Next, we compute the correlation coefficients between the numeric variables and visualize them using a heatmap (from the Seaborn library [15]) of the correlation matrix.

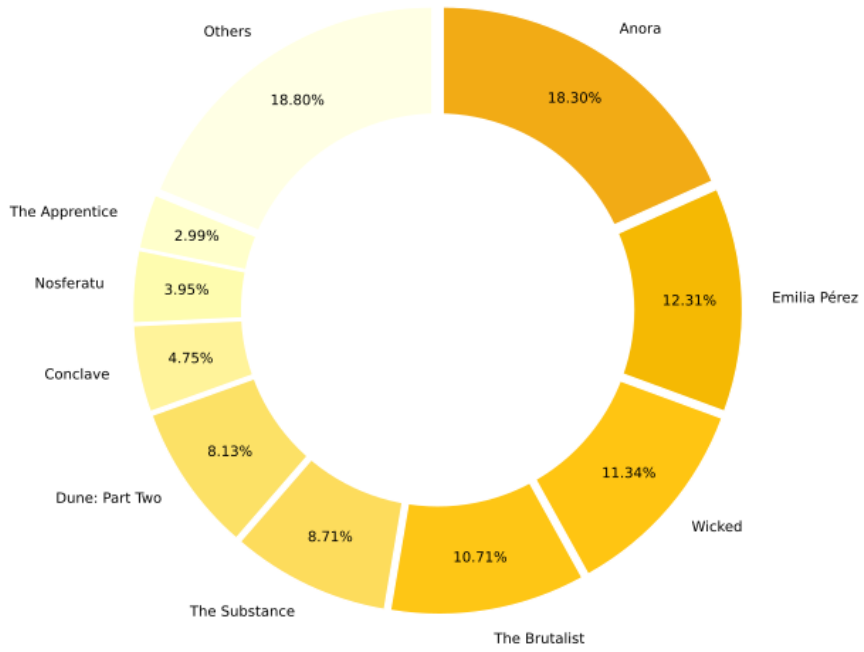


From the matrix, we observe strong positive correlations among various variables, such as ratings across different platforms, budget and revenue, and popularity-related metrics (e.g., *popularity*, *tmdbVotes*, and *IMDbVotes*). Interestingly, we also note that ratings show no significant correlation with either *budget* or *revenue*, suggesting that critical reception does not directly reflect commercial success.
Finally, we analyse the frequency distribution of some categorical variables, such as *language*, *country*, and *genre*. Here are some findings:

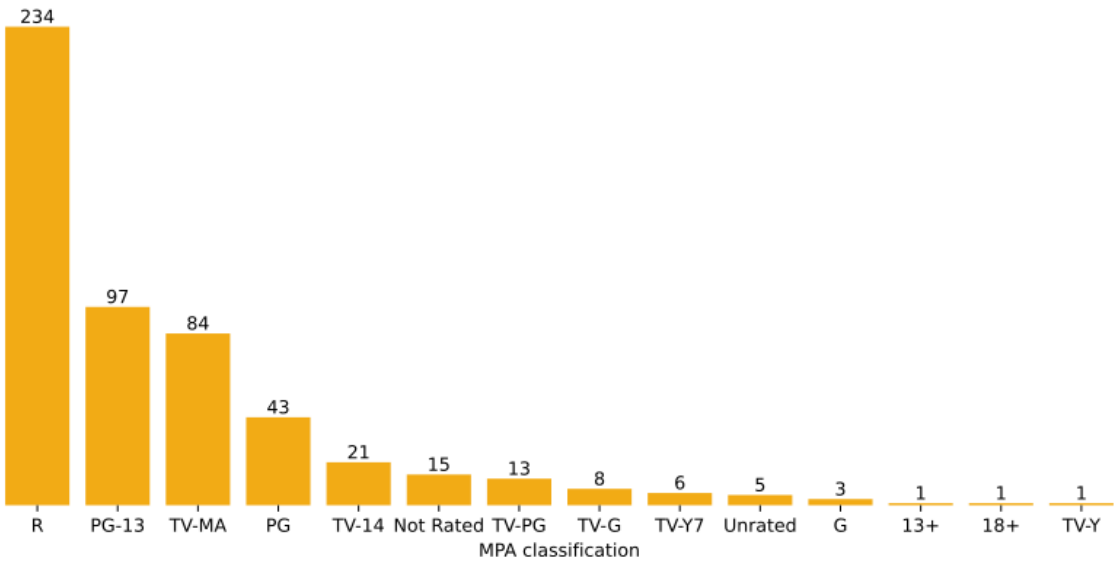
- The most frequent original language is, unsurprisingly, English, followed by French, Spanish, and then Italian.
- The United States produced the highest number of movies in the dataset, followed by France, Great Britain, and again Italy in fourth place.
- The three most frequent genres, in order, are Drama, Comedy and Action.

8.2 Visualizations

In this subsection, we present a set of visualizations designed to highlight interesting aspects of the dataset in a more intuitive and accessible way with the use of the Matplotlib library [14].
The first two visualizations are univariate representations of the distributions of two variables not previously considered: a pie chart showing the relative frequency of *mentionCount*, and a bar chart displaying the absolute frequencies of *mpaRating*.

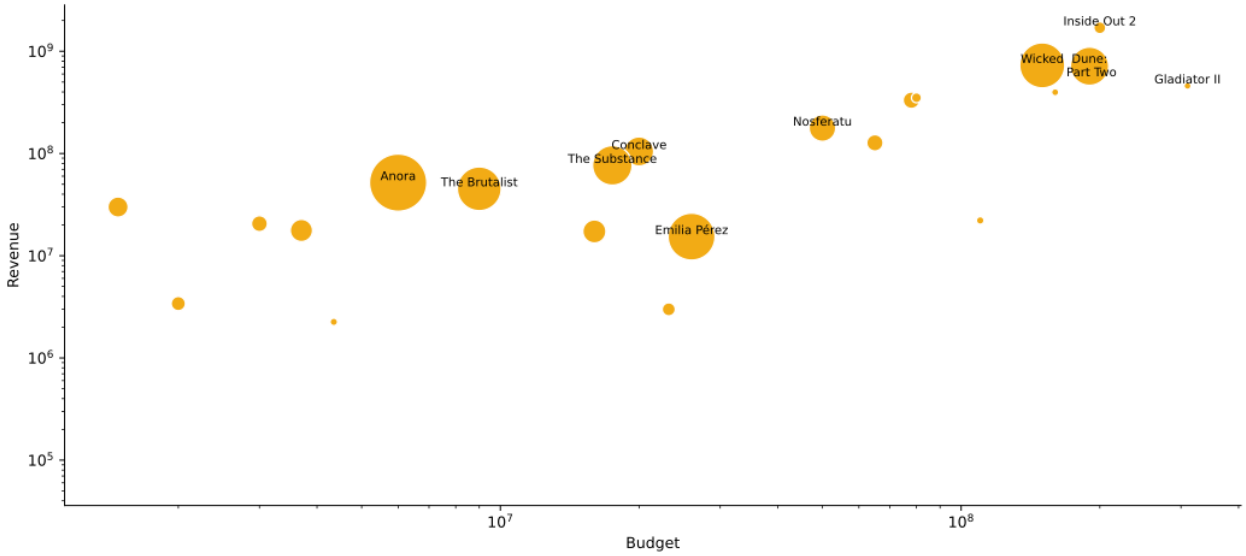


From the pie chart, we observe that Anora is by far the most mentioned movie in Reddit comments about the Oscars, accounting for 18.3% of the mentions, probably because it has a lot of nominations, won many Awards and, in particular, is the winner of the Best Picture Award; the second most mentioned is Emilia Pérez which had received a lot of criticism both from the Mexican and LGBTQ+ community because of how they have been represented in the movie; the third is Wicked which is probably mentioned a lot since it is taken from the famous “The wonderful wizard of Oz” and also the leading duo of Ariana Grande and Cynthia Erivo was highly appreciated by the audience.



The bar chart shows that the most common rating among the movies (234 out of about 1000) is R (Restricted: under 17 requires accompanying parent or adult guardian), followed by PG-13 (Parents Strongly Cautioned: some material may be inappropriate for children under 13), and TV-MA (Mature Audience: suitable for adults only; not appropriate for children under 17).

The last visualization is a bubble chart that displays *budgets* and *revenue* on logarithmic axes, with the size of each bubble representing the number of mentions in Reddit comments.



Here are some key observations about the plot:

- Inside Out 2 stands out with the highest revenue, suggesting strong commercial success
- Wicked and Dune: Part Two also show high budget and high revenue, indicating large productions that performed well
- Anora and The Brutalist have relatively low budgets but still achieved decent revenue, and their large bubbles suggest high nomination counts despite smaller scale
- Gladiator II has one of the highest budgets, but its revenue appears lower compared to others in the same range

Overall, there's a general positive correlation between *budget* and *revenue*, but with notable exceptions. Some films with moderate budgets (e.g., Emilia Pérez) didn't achieve comparable revenue.

9. Data Storage

The Data Storage phase focuses on organizing and saving the processed dataset in a structured and accessible format. This represents the final step before querying our data using SQL.
Since we have worked with pandas DataFrames throughout the project, we can easily export these tables using the DataFrame.to_csv() function. We apply this process to the three DataFrames we created and processed, and subsequently insert them into a SQLite database, ready to be queried.

10. Queries

We're now approaching the last step of our case study. All the work done from Data Acquisition to Data Processing to Data Storage was a necessary path to follow to describe the nominees of the Academy Awards in a structured way, considering some particular variables; this enables us to analyse different aspects of those movies. The analysis is executed using SQLite [16].
The first step is to add a new column to the movies dataframe: *ratingAverage*. This new index is intended to provide a general overview of the public reception of each film by aggregating the various rating sources available. To compute it, we calculate the average of the four rating variables present in the dataset. Since some of these ratings are missing (represented as *nan* in NumPy), we handle this by computing the numerator as the sum of the available ratings, using the coalesce approach, replacing missing values with 0. For the denominator, we count the number of non-null rating fields using conditional logic using the *case* function. In this way, the average is calculated only over the ratings that are actually available for each movie. Once the new column is added, we are able to perform various queries and analyses based on this overall rating metric. We decided not to add this column in the Python Notebook to ensure minimality in the output database, since the new column is obtained as a composition of some others.

The first two queries aim to analyse from nominees which movie received the most nominations from the 23 categories and then which one won the most Oscars. The top 5 of the first question returns

- Emilia Pérez with 13 nominations
- Wicked and The Brutalist with 10
- Conclave and A Complete Unknown with 8

On the other hand, the movies that have won the most Awards are

- Anora with 5 Awards
- The Brutalist with 3
- Wicked, Emilia Pérez and Dune: Part Two with 2 Awards

Those are interesting results, we can notice that Anora won 5 Awards out of the 6 nominations received while Emilia Pérez on which we had high expectations only got awarded of 2 prizes; another interesting aspect concerns the unsuccessful outcome of film with various nominations which ended up not winning any award, such as A Complete Unknown with 8 nominations, Nosferatu with 4 and The Wild Robot and Sing Sing with 3. This is a great first way to show that even though a film had many nominations, it ends up with few or no Awards.

The following queries are aimed at identifying the best-rated movies based on the computed *ratingAverage*. We begin by highlighting the Oscars' Best Picture winner, Anora, which holds an impressive average rating of 8.325. This serves as a benchmark for comparison.
Next, we display the full list of movies from the dataset - nearly 1,000 entries - sorted in descending order by their average rating. Interestingly, Anora ranks 30th in this overall list. This is still a very strong placement, especially when considering that many of the higher-ranked titles tend to be niche or have limited audiences, which often skews ratings upward due to selection bias (only highly interested viewers rate them, and they rate positively).
At the top of the overall ranking is Attack on Titan: The Last Attack, with an outstanding average rating of 8.9. However, it's worth noting that this type of content (anime and episodic releases) often garners extremely passionate fan ratings, which may not be fully representative of broader public opinion.

Now we analyse which results to be the best movie for each Oscars category based on average rating. At first, we show for each category which movies have the highest rating, and we compare it to the actual winners. In just 7 out of 23 categories movies with the highest rating are also winners.
Showing the obtained list in a descending order on *ratingAverage* we see the top 5 movies are

- No Other Land with 8.8745
- Flow and The Wild Robot with 8.65
- Anora with 8.325
- Conclave with 7.95

This view gives a more balanced comparison among critically recognized films and highlights how some lesser-known nominees actually achieve very high average ratings, suggesting a strong reception from smaller but enthusiastic audiences.

Lastly, we analyse wins in each category compared to the number of mentions on the selected Reddit's posts; this gives us a first hint on if the audience is at a certain level aligned with the results and from a broader perspective on how much every movie made people talk about them. In 15 out of 23 categories winners and most mentioned movie are aligned; thus this gives just a hint on if audience and commission appreciated or at least had a great impression on the same movies, since the mentions were counted after the Ceremony happened, and those data can be biased by the fact that a lot of people may have decided to watch winners' movies after they've been awarded or may have commented on movies out of disappointment on the awarding, for example.

Conclusions and Future Developments

This project demonstrated the application of a complete data management pipeline on a real-world cultural event: the 97th Academy Awards and movies in general. Starting from data acquisition through APIs and web scraping, we constructed a rich, integrated dataset combining metadata, ratings, social media mentions, and award information for movies released in 2024.
By applying rigorous processing, modelling, enrichment and quality assessment techniques, we ensured the dataset's usability and reliability. The subsequent data exploration and SQL-based querying enabled us to derive insightful analyses regarding movie popularity, audience reception, and correlations between Oscar wins and social engagement.
Key findings include the identification of strong discrepancies between nominations and actual wins, the confirmation that critical acclaim does not always align with commercial success, and the observation that online discussions often reflect award outcomes. The project highlights how combining structured and unstructured data sources can lead to more comprehensive and nuanced insights.

This project can be extended in various ways, and in particular some possible future developments are:

- broaden analysis to different edition of the Academy Awards;
- deepen analysis on reddit comments using NLP and in particular techniques of sentiment analysis and such;
- use the column *awards* to analyze different awards (Cannes, Venice ecc);
- study the relationship between release date and wins;
- analysis on the relationships between genre of a movie and its nominations and wins;
- broaden analysis using different social networks, for example search a pattern between popularity of actors (considering for example followers on IG) and wins in film in which they appeared;
- create a new index based on number of ratings and actual rating (a movie rated 8.3 with hundreds of votes has a more solid rating than a movie with a rating of 9 and just a few votes);
- this analysis could also evolve into a predictive study by leveraging the information available 10 days prior to the ceremony to forecast potential outcomes, and subsequently comparing those predictions with the actual results and the data collected 10 days after the event.

Overall, this case study not only provides a robust example of data management in practice but also emphasizes the power of interdisciplinary analysis across cinema, technology, and public discourse.

References

[1] The Academy of Motion Picture Arts and Sciences. *Oscars Official Website*. https://www.oscars.org/oscars

[2] Wikipedia. *Academy Awards*. https://en.wikipedia.org/wiki/Academy_Awards

[3] Wikipedia. *97th Academy Awards*. https://en.wikipedia.org/wiki/97th_Academy_Awards

[4] *Requests: HTTP for Humans*. https://requests.readthedocs.io/en/latest/

[5] Python Software Foundation. *json — JSON encoder and decoder*. https://docs.python.org/3/library/json.html

[6] Python Software Foundation. *pprint — Data pretty printer*. https://docs.python.org/3/library/pprint.html

[7] *BeautifulSoup Documentation*. https://www.crummy.com/software/BeautifulSoup/bs4/doc/

[8] *pandas Documentation*. https://pandas.pydata.org/docs/

[9] *PRAW: The Python Reddit API Wrapper*. https://praw.readthedocs.io/en/stable/

[10] Python Software Foundation. *time — Time access and conversions*. https://docs.python.org/3/library/time.html

[11] Python Software Foundation. *pickle — Python object serialization*. https://docs.python.org/3/library/pickle.html

[12] Python Software Foundation. *re — Regular expression operations*. https://docs.python.org/3/library/re.html

[13] *NumPy Documentation*. https://numpy.org/doc/stable/

[14] *Matplotlib Documentation*. https://matplotlib.org/

[15] *Seaborn Documentation*. https://seaborn.pydata.org/

[16] *DB Browser for SQLite*. https://sqlitebrowser.org/

[17] *The Movie Database (TMDb)*. https://www.themoviedb.org/

[18] *TMDb Developer API*. https://developer.themoviedb.org/

[19] *OMDb - The Open Movie Database*. https://www.OMDb.org/

[20] *OMDb API*. https://www.OMDbapi.com/

[21] Reddit. *The 97th Annual Academy Awards - Official Thread*. https://www.reddit.com/r/Oscars/comments/1j24nvr/the_97th_annual_academy_awards_official/

[22] Reddit. *2025 Oscars Megathread*. https://www.reddit.com/r/popculturechat/comments/1j1xpis/2025_oscars_megathread/

[23] Reddit. *2025 Oscar Nominations - Full List of Nominees*. https://www.reddit.com/r/movies/comments/1i836g7/2025_oscar_nominations_full_list_of_nominees/

[24] OpenAI. *ChatGPT (GPT-4)*. 2025. https://chat.openai.com (Used for correcting the report and writing the References)