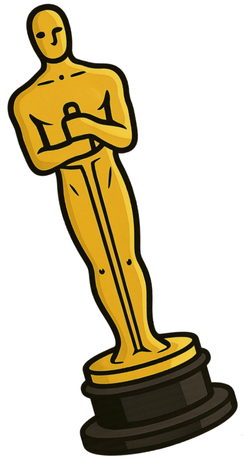
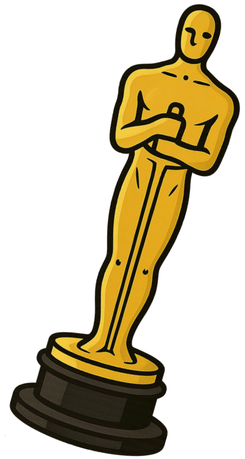


Analysis on 97th Academy Awards



Data Management Project 2024-2025

by Chiara Genuardi and Giovanni Noè

Table of contents

Data Acquisition - API

We acquired general data about movies and their ratings through conventional API queries from TMDB and OMDb.

TMDB

Purpose:
Acquire general information about movies.

Process:

- retrieve internal IDs from */discover/movie* (with *vote_count>24* and *year=2024*)
- manually get IDs for Oscar-nominated movies with few votes
- individual queries from */movie/{movie_id}*

OMDb

Purpose:
Acquire multiple rating sources in one shot.

Process:

- use IMDb IDs from TMDB data
- individual queries from the only endpoint

Then we collected the text of comments from three relevant posts in the subreddits *r/oscars*, *r/movies* and *r/popculture*, retrieving approximately 35 thousands comments.

Reddit

Purpose:
Get comments from Reddit discussions about Oscars to create a mention count for nominated movies.

Process:

- implement a code with the *PRAW* (Python Reddit API Wrapper) library to collect comments
- save these comments as a string list in a *.pkl* file to access them in following steps

Lastly we acquired data about Oscars nominations and winners by scraping the 97th Academy Awards page on Wikipedia.

Wikipedia

Purpose:
Collect data about nominations, winners and people involved in the Awards (e.g. actors and directors).

Process:

- develop a script that automatically scrapes the data using the *BeautifulSoup* library
- save data in a list of dictionaries, where each dictionary corresponds to the nomination of a specific film for a specific Award category

WIKIPEDIA

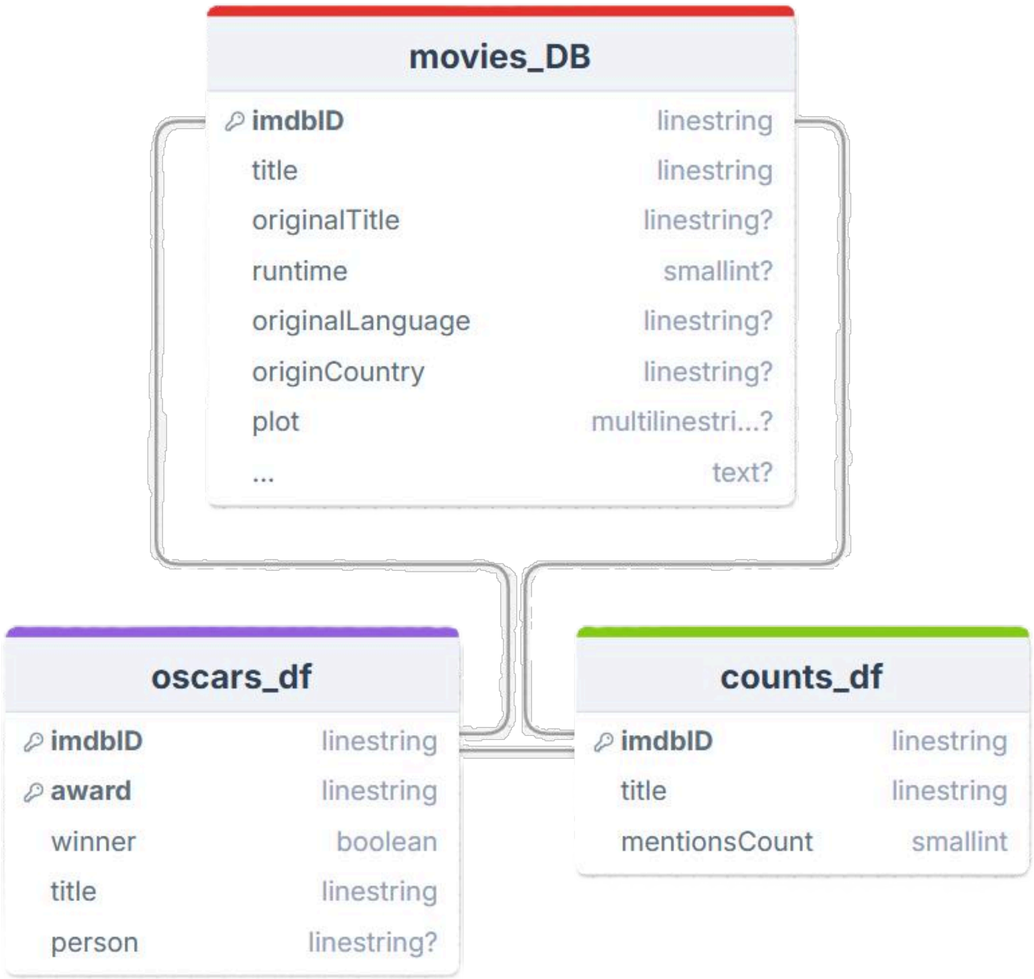
Data Modelling

For our project we chose the **relational model** for many reasons:

- **clear structure** and **simplicity**
- **ease of data integration**
- strong support for **structured queries** using SQL
- no need of different data structures that fasten the queries since we had **less than 1000 rows**

We designed a schema with three tables:

- ***movies_DB*** table: general information about movies
- ***oscars_df*** table: Oscar nominations and winners
- ***counts_df*** table: count of mentions in Reddit posts for each nominated movie



TMDB

TMDB is our first source of information; we keep most of information from there and then enrich it with OMDB.

- **delete** non-useful **columns**:
  → *adult*, *backdrop_path*, *belong_collection*, *production_companies*, *production_countries*, *poster_path*, *spoken_languages*, *status*, *tagline*, *video* for non-relevance
  → *id* keep IMDbID as key
  → *genres* keep it from OMDB for storage reasons
- **replace** missing values and zeros with ***nan*** type of NumPy
- check for **consistency** in data type
- **convert** to format date *release_date*
- rearrange column positions and change columns names

OMDB

We extract from *Ratings* the rating from Rotten Tomatoes and scale it to 0-10.

- **delete** non-useful **columns**:

  → *Year* since it is in *release_date*
  → *Runtime*, *Released*, *Plot*, *Language*, *BoxOffice* keep these from TMDB
  → *Writer*, *Country*, *Poster*, *DVD*, *Type*, *Production*, *Website*, *Response*, *Season*, *Episode*, *seriesID* for non-relevance
  → *Ratings*
- **replace** missing values and zeros with ***nan*** type of NumPy
- check for **consistency** in data type
- **convert** *IMDbRating*, *IMDbVotes* and *Metascore* to float and scale the latter in the range 0-10
- rearrange column positions and change columns names

Oscars

- **add** column *ID*

Reddit

Since Reddit comments serve a different purpose (count how many comments mention each nominated movie), the processing of this data differs as well.

We proceeded as follows:

1. **construct a dictionary** where the keys are the titles of the nominated movies, and the values are lists of words associated with each movie
2. **load** the Reddit **comments** from the *.pkl* file and iterate through them
3. for each comment, check whether any of the words associated with each movie appear, using a **regular expression** composed for each movie
4. **create the *counts_df* DataFrame**, which contains a row for each movie and *title*, the count as *mentionsCount* and *IMDbID* as attributes.

Data Quality Assessment

Data Quality is taken into account throughout all the Data Science pipeline:

→ **usability, readability** and **minimality** were considered during data processing.

We actively checked some specific dimensions:

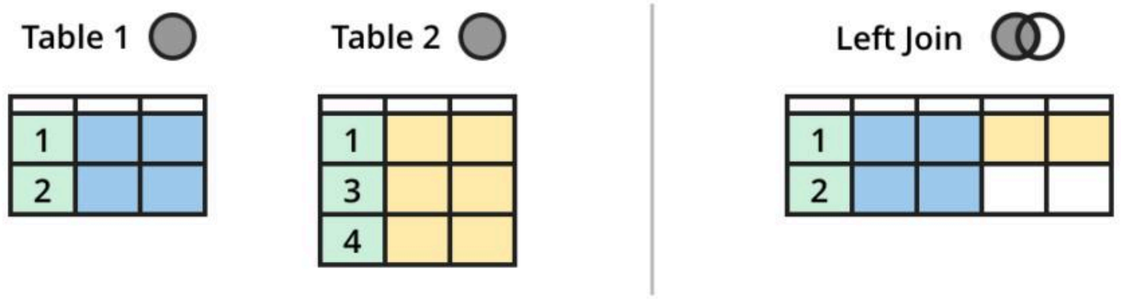
- **accuracy -** identify missing values
  → referred to minor movies
- **minimality -** check for duplicate rows on the combination of *title* and *IMDb_ID*
  → ‘Anuja’ is duplicated
- **consistency -** analyze numerical variables to detect outliers or values that fall outside a plausible range
- **completeness -** check whether all Oscar-nominated films were included in the TMDB and OMDb datasets and identify missing values for key variables across these sources
  → ‘Yuck!’ and ‘I am ready, Warden’ are not present

Data Enrichment

Data Enrichment involves supplementing existing data with additional information from external sources—in this case, the OMDb database—creating more complete profiles for each movie.

We **merged** the TMDB dataframe with the OMDb dataframe using the ***IMDbID*** field as the common key, using the first as our primary source to be enriched.

We haven't joined the Oscars since this would lead to a lot of **duplicates**; therefore the *IMDbID* field will allow us to perform comparisons.



Data Quality Improvement and Assessment

Data Quality Improvements are performed after Data Enrichment to avoid unnecessary work on data we may have ended up not storing.

- **accuracy**
  → delete rows with **null *IMDbID***
- **minimality**
  → delete **duplicates** on the pair (*title*, *IMDbID*)
- **completeness**
  → **add** missing nominees and information found on them

We then repeated previous Data Quality Assessments on **accuracy, minimality, consistency** and **completeness.**

Data Exploration - Statistics

Basic statistics findings:

- the longest movie has a **duration** of 247 minutes, the average runtime is 101 minutes
- the average **budget** is 36 million dollars, the average **revenue** 64 million dollars
- Rotten Tomatoes is the platform that shows the highest **variability in user ratings**, with a standard deviation of 2.54 and ratings ranging from 0 to 10

Linear correlation:

- **predictable strong correlations** between some variables
- interestingly, no significant correlation between ***ratings*** *and* either ***budget*** or ***revenue***

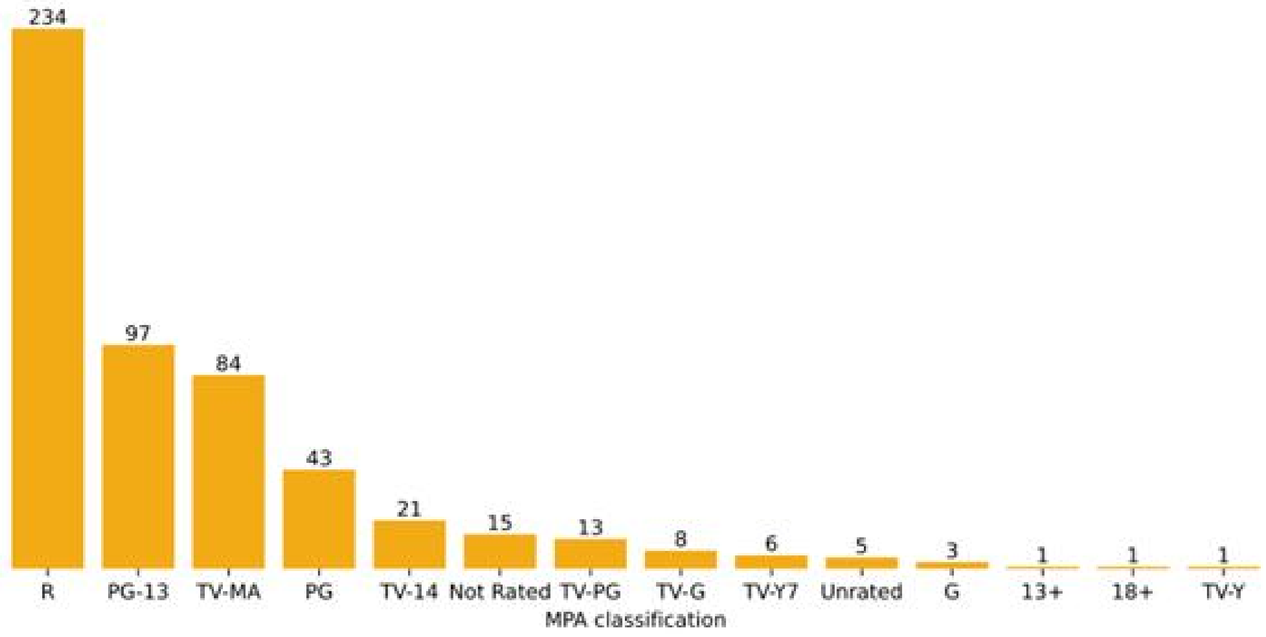
Distribution of some categorical variables:

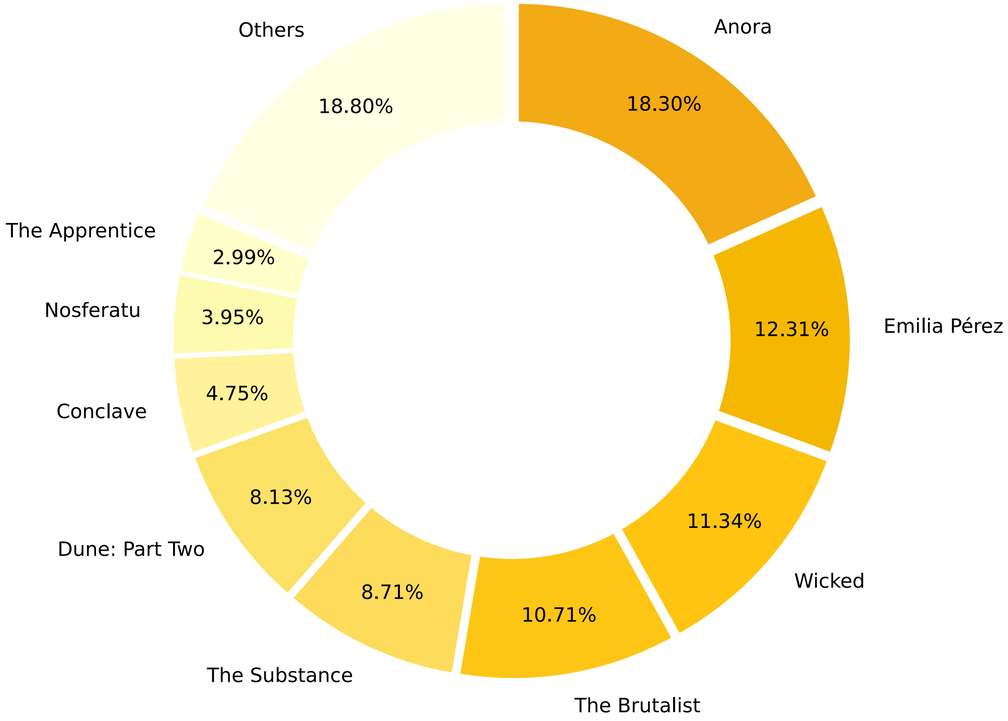
- The most frequent **original language** is English, followed by French, Spanish, and Italian.
- The three most frequent **genres**, in order, are Drama, Comedy and Action.

Data Exploration - Visualizations

Mentions Podium:

1. Anora, 18.3% (winner)
2. Emilia Pérez, 12.3% (very criticized)
3. Wicked 11.3% (very appriciated leading duo)





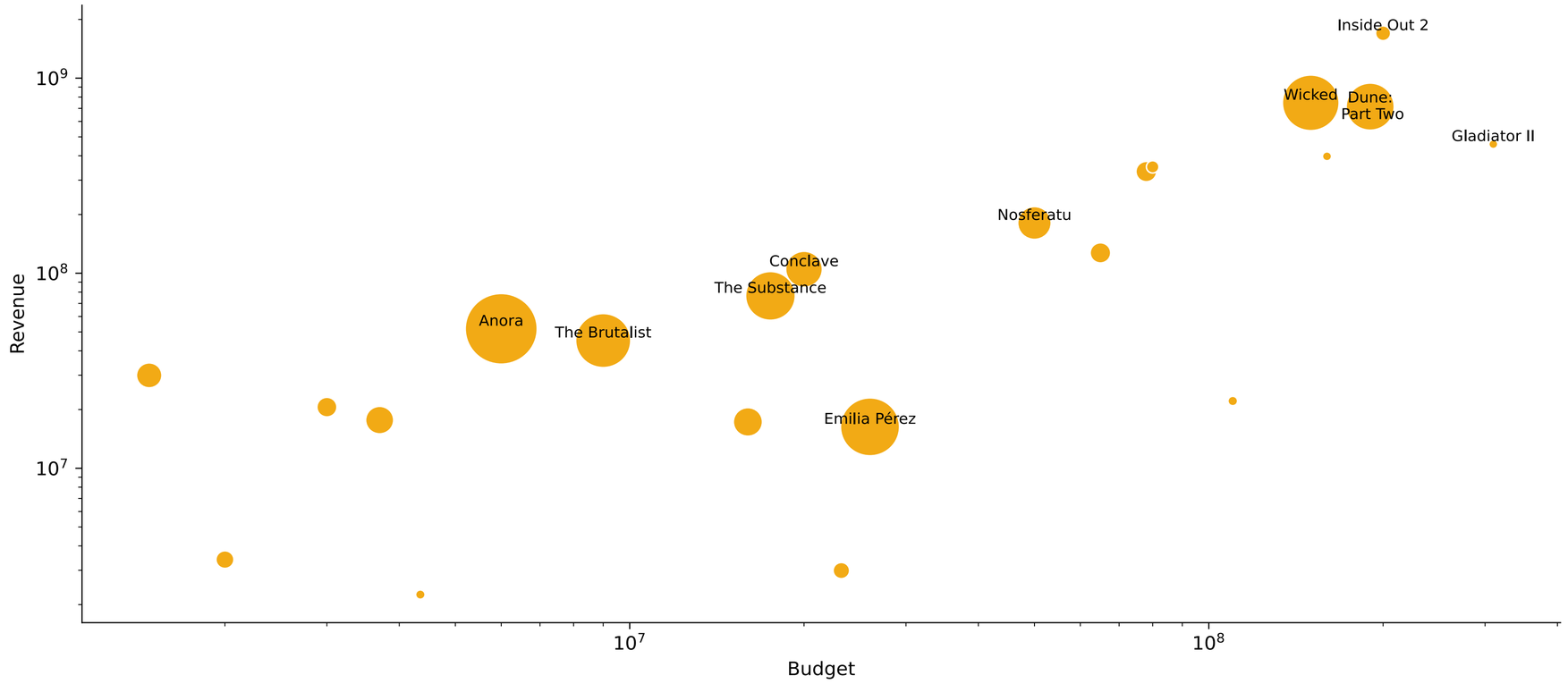
MPA ratings:

1. Restricted (Adults Only)
2. Parents Strongly Cautioned (children over 13)
3. MAture Audience (Adults Only)

Data Exploration - Visualizations

Budget vs revenue:

- Inside Out 2: highest revenue
- Wicked and Dune: Part Two: high budget and high revenue
- Anora and The Brutalist: relatively low budgets but decent revenue
- Gladiator II: high budget but relatively low revenue



Data Storage

Since we have worked with pandas DataFrames throughout the project, we were able to:

- easily **export** these **tables** using the DataFrame.to_csv() function
- insert them into a **SQLite database**
- now they are **ready to be queried**

Queries

At first we computed a new index ***ratingAverage*** and added it as a new column.
This column wasn't added in Python to ensure **minimality** since it is a composition of other columns.

Since we used a relational model using **SQLite** was a natural choice.

1. analyses on **nominations**: which movies have most nominations and which have most Oscars

1	Emilia Pérez	13
2	Wicked	10
3	The Brutalist	10

1	Anora	5
2	The Brutalist	3
3	Wicked	2

Queries

2. analyses on **ratings**: analyze variable *ratingAverage* on all the movies of TMDB dataset, considering that Anora has a score of 8.325

1	Attack on Titan: THE LAST ATTACK	8.9
2	No Other Land	8.8745
3	Lost Ladies	8.7
4	Look Back	8.7
5	Flow	8.65

3. analyses on **ratings for category**: show which has the highest *ratingAverage* and compare them to actual winner → in 7 out of 23 category movies with the highest rating are also winners
4. analyses on **mentions for category**: show which has the highest *mentionsCount* and compare them to actual winner → in 15 out of 23 category movies with the highest rating are also winners

Conclusions and Future Developments

A comprehensive data pipeline was applied to analyze the 97th Academy Awards and 2024 movies, enriching diverse data sources for deep insights.

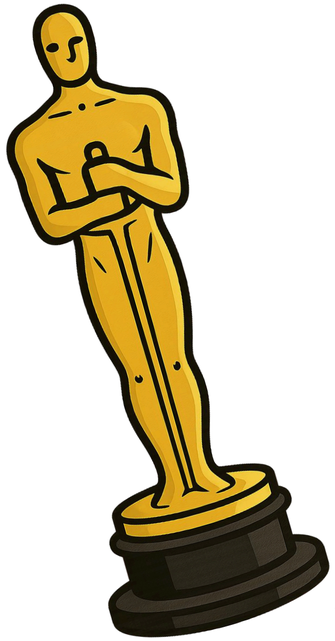
Key findings revealed **gaps** between nominations and wins and how **social media engagement** reflects and sometimes predicts award outcomes.

This project can be broadened in various ways:

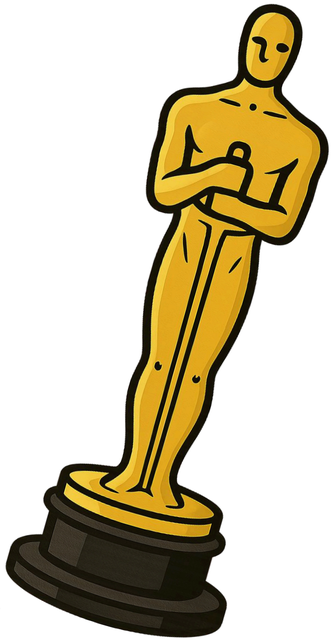
- broaden analysis to different **editions** of the Academy Awards
- deepen analysis on Reddit comments using **NLP** and in particular **sentiment analysis**
- broaden analysis using different **social networks**
- create a new **index** based on number of ratings and actual rating
- this analysis could also evolve into a **predictive study**

Main references

- *The Academy of Motion Picture Arts and Sciences. Oscars Official Website. https://www.oscars.org/oscars*
- *Wikipedia. Academy Awards. https://en.wikipedia.org/wiki/Academy_Awards*
- *Wikipedia. 97th Academy Awards. https://en.wikipedia.org/wiki/97th_Academy_Awards*
- *The Movie Database (TMDb). https://www.themoviedb.org/*
- *OMDb - The Open Movie Database. https://www.OMDb.org/*
- *Reddit. https://www.reddit.com/*



Thanks for your Attention



by Chiara Genuardi and Giovanni Noè