IMDB Rating Prediction

Teuku Mahfuzh Aufar Kari

1. Abstract

We built machine learning (ML) pipelines to predict the IMDB rating of IMDB titles, leveraging both the numerical and sparse text data as features. We achieved mean absolute error of 0.79 and r-squared score of 0.37 on test set, outperforming the baseline model. The sparse text data features are listed among the most important features. Leveraging the sparse text data and addressing the issue of dimensionality can be the focus for future works.

2. Data Description

In this work we are using the IMDB database to predict the average ratings of a given title. The data is available here.

3. Technical Approach

We split the technical work into 3 components:

1. Data wrangling, can be found in this notebook.
2. Exploratory data analysis, can be found in this notebook.
3. ML Modelling, can be found in this notebook.

3.1 Main Hypothesis and Data Wrangling

We hypothesized that the average rating of a given IMDB title depends primarily on 3 factors:

1. ***The nature of the title***, such as the type (tv show vs movie), the genre (horror vs comedy), etc.
2. ***The production value of the given title***. Since we don't have information on the production budget, we try to infer the production value from the production crew information (e.g. director and writer).
3. For a TV episode, ***the performance of the past episode and the longevity of the show***.

Building on top of these hypothesis, we calculated the following additional features as the predictors, as listed on Table 1.

Table 1 Description of calculated features

Column	**Type of data**	**Description**
parentTconst	String	Alphanumeric identifier of the parent TV Series of the given title
directors_writers	String	Text data containing the identifier of the director(s) & writer(s) of the given title
max_episodeNumber	Numerical	Maximum number of episodeNumber of the given title
max_seasonNumber	Numerical	Maximum number of seasons of the given title
Column with prefix 'unique_'	Dummy indicator	Number of unique {language, ordering, region, title, types} the given title has
Column with prefix 'listed_'	Dummy indicator	Number of listed {actor, actress, archive_footage, archive_sound, cinematographer, composer, director, editor, producer, production_designer, self, writer,total} for the given title
Column with prefix 'type_'	Dummy indicator	One-hot encoding whether the given title is of type {short, tv series, tv episode, movie, video, tv special, tv movie, video game, tv short, tv mini-series}
Column with prefix 'genre_'	Dummy indicator	One-hot encoding whether the given title is considered to be of {Adventure, Drama, Short, Comedy, Family, Game-Show, Music, Documentary, War, Reality-TV, Action, Animation, History, Sci-Fi, Fantasy, Horror, Mystery, Adult, Romance, Crime, Film-Noir, no_genre, Talk-Show, Western, Thriller, Sport, Biography, Musical, News} genre
total_genres	Numerical	Total number of genres the given title is considered
number_past_episode	Numerical	For TV episode, the number of past episodes before the given title is released
average_show_past_rating	Numerical	For TV episode, the average rating of the past episodes before the given title
director_median_past_rating	Numerical	The median rating of the past contents that the directors has worked on before directing the given title

director_number_past_movies	Numerical	The number of past contents that the directors has worked on before directing the given title

3.2 Exploratory Data Analysis

Our exploratory data analysis identified the following noteworthy insights:

1. TV Episode is the most common title type (47%). It also has the highest average rating compared to other title types.
2. As a given title can be classified into more than 1 genre, we performed difference in means analysis to study the relationship between a title genre and its rating. For a given genre, for example horror, we calculate the difference in means of average rating between titles classified as horror (genre_Horror ==1) and titles classified as non-horror (genre_Horror ==0). We found that documentary, history, and crime are the genres with the most positive difference in average rating. We also found that horror, thriller, and adult are the genres with the most negative difference in average rating.
3. We found that ‘average_show_past_rating’ and ‘director_median_past_rating’ are the features with the highest correlation with rating.

3.3 ML Modelling

Our ML modelling work consist of the following steps:

1. We split the data into training and testing set.
2. For the numerical features, we performed the following pre-processing:
    a. We removed redundancy in the form of sparse features, features with zero variance, and highly correlated features.
    b. We performed k-Nearest Neighbor (k-NN) imputation to impute missing values to enable downstream modelling.
    c. We performed Recursive Feature Elimination (RFE) to select the best numerical features. We first performed RFE with Cross Validation (RFECV) to study the relationship between number of features on model performance. We then used the insight from RFECV to decide the number of features to be selected by the RFE procedure.
3. For the sparse text data (‘parentTconst’ and ‘directors_writers’), we performed custom tokenization and dimensionality reduction. We used count vectorization for text tokenization followed by truncated SVD to reduce the number of dimensions. As the standard sklearn implementation does not support the treatment of missing values, we wrote our own custom implementation for this step.
4. We performed standard scaling on the processed numerical features and the tokenized sparse text features.
5. We trained the ML model on the pre-processed data and tuned the parameters using randomized search.
6. We analyzed the relative importance of each feature using permutation importance analysis.

4. Modelling Results

There are 4 models that we used in this work:

1. Baseline Model: Naïve prediction model which uses the mean of the average rating on the training set as the prediction.
2. Linear Model: Elastic Net model using on only the kNN-imputed numerical features.
3. XGB Model: XGBoost model using only the unimputed numerical features.
4. XGB+Tokenization Model: XGBoost model using the unimputed numerical features and the sparse text features.

The performance of the model is given on Table 2. All the three ML model perform better than the baseline model. Both XGB and XGB+Tokenization models performed better than the Linear Model. As both XGB-based model uses the unimputed numerical features, this suggests that the Sparsity-aware Split Finding algorithm of XGBoost can be a better imputation method than k-NN imputation which requires additional computing resources. We also find that including the sparse text features as predictors improves the model performance.

Table 2 The training and testing performance the tested model

Model	**Training Performance**	**Testing Performance**		
	R2 - Score	**Mean Absolute Error**	**Mean Squared Error**	**R2 - Score**
Baseline Model	0.000	1.074	1.933	0.000
Linear Model	0.223	0.917	1.512	0.218
XGB Model	0.347	0.821	1.271	0.342
XGB+Tokenization	0.365	0.792	1.215	0.371

We performed permutation importance test to study the importance of each feature on model performance. The result of the permutation importance in presented on Table 3. Based on the analysis, the most important feature is 'parentTconst', with permutation importance score significantly higher than the rest of the features. Additionally, the other sparse text feature 'directors_writers' is also listed among the top 10 features. The past performance features, 'average_show_past_rating' and 'director_median_past_rating' are also listed among the top 10 list.

Table 3 Top 10 features with the highest Permutation Importance Score

Feature	Permutation Importance Score	Rank
parentTconst	0.363	1
average_show_past_rating	0.145	2
numVotes	0.056	3
startYear	0.049	4
type_movie	0.045	5
genre_Documentary	0.039	6
directors_writers	0.039	7
runtimeMinutes	0.036	8
director_median_past_rating	0.022	9
number_past_episode	0.022	10

5. Potential Future Works

Currently the best performing model has an r-squared score of 0.37. To improve the performance of the model, potential future works includes:

1. **Optimizing the sparse text data as feature**. Our analysis shows that the sparse text data has significant influence on model performance. Due to resource limitation, we only explored two of them. We also didn't spend much time to work on optimizing the dimensionality reduction parameters (number of dimension and explained variance trade-off). In the future, we can invest more resources to explore more (sparse) text data and to optimize the modelling parameters. The use of more modern text tokenization method beyond count vectorization is worth exploring.
2. **Network clustering of production personnel to reduce dimensionality.** There are more than 48,000s directors and writers on the 100,000 title samples that we used for this work. The high number of dimensions of production personnel makes it hard to use them as features. One way to address this is to build a network model of production personnel, where each node is a person and if two persons ever worked together on a title, their nodes are connected with an edge. The network model can be clustered using community detection algorithm. As a high-quality actor tends to work with other high-quality actors, the community cluster information of a given production personnel will carry information about the production value of the given personnel. Hence instead of working with more than 48,000 columns, we can potentially work with much smaller dimensions.