Is adding the release year of movies beneficial for predicting movie ratings using text features?

Anonymous

1 Introduction

Movie-watching remains a popular leisure activity worldwide and contributes to the thriving global film industry, which is characterized by significant profits and diverse audiences [1]. A movie's rating serves as a determinant for viewers, guiding their decisions on which films to invest their time and money in. Therefore, it's crucial to accurately predict movie ratings for audiences. In this study, I aim to use the text features of movies - title, tagline, overview, and production companies to predict their rating categories and assess the impact of including the release year on the performance of the machine learning models. The dataset utilised in this analysis is from TMDB, and detailed column information will be provided in the subsequent section.

2 Related work

Previous research done by Bhadrashetty, A. and Patil, S. (2024) has explored the prediction of movie success and ratings based on textual data, such as movie titles, descriptions, and comments under movie teasers on YouTube [2]. After collecting data using the data mining approach, they trained random forest and naïve Bayes models on these features. The result revealed that accurately predicting the movie ratings solely based on these text features is challenging.

Abarja, R. A., and Wibowo, A. (2020) pursued an alternative strategy to forecast movie ratings [3]. They involved more diverse features, including topic features, such as overview and keywords, categorical features like release year, and aggregated numerical features derived from historical records, for instance, average ratings of related movies directed by the same director. They employed a generative convolutional neural network (CNN) and dropout to avoid overfitting. This study underscored the effectiveness of neural network models and the significance of aggregated features as the CNN model made accurate rating predictions using derived numeric features rather than text features and release year.

3 Method

3.1 Features and Label

Table 1 outlines the columns within the TMDB dataset to build the models. *title*, *tagline* and *overview* were selected since they provide brief but informative outline of movies. For *production_companies* and *release_year*, the former may be related to the quality of movies, and the latter may imply the level of special effects and filmmaking techniques of movies. All the above factors contribute to movie ratings. The label here is *rate_category*, which ranges from 0 to 5, representing scores shown in Table 2.

Name	Description	Type
title	Title of the movies	text
release_year	Release year of the movies	categorical
overview	Overview of the movies	text
tagline	Tagline of the movies	text
Production_ companies	Production company of the movies	text
rate_category (label)	Level of movie rating	categorical

Table 1 - *Features and Label in the TMDB Dataset.*

Rating range	Level
Rating < 4	0
4 <= Rating < 5	1
5 <= Rating < 6	2
6 <= Rating < 7	3
7 <= Rating < 8	4
Rating >= 8	5

Table 2 - *Rating bins with the corresponding scores.*

To further inspect the label distribution, Figure 1 illustrates the distribution of *rate_category* within the training and evaluation sets. While both sets share a similar distribution, the class imbalance exists. For instance, class 3 accounts for over 25%, while class 5 only accounts for less than 10% of all instances.

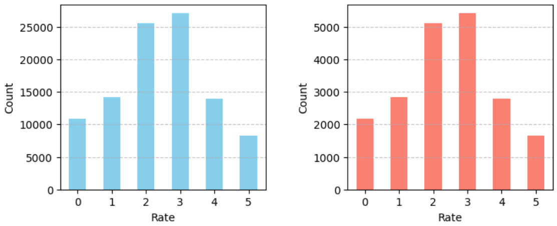

Figure 1 - *rate_category distribution across the training set (left) and evaluation set (right).*

3.2 Feature Engineering

3.2.1 Text Features

- **Cleaning and filtering**

The text features - *title*, *overview*, *tagline*, and *production_companies* were concatenated into a sentence. I removed English stop words, eliminated punctuations and lemmatized the words to extract the original meaning. Given the enormous vocabulary across the dataset with over 15,000 unique words, the concatenated sentences were filtered to retain 20% of the top frequent words.

- **Vectorization**

I use Term Frequency-Inverse Document Frequency (TF-IDF) to vectorize the words. Unlike Bag-of-Words (BoW), which gives the same weight to each word, TF-IDF considers the frequency of each word in a sentence, as well as how rare each word is across all sentences. It penalizes common words that appear frequently across all documents, thus providing potentially more meaningful representations.

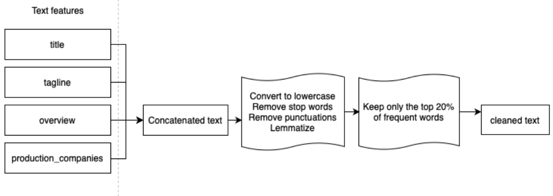

Figure 2 - *The steps of pre-processing text features.*

3.2.2 Categorical Feature

release_year was encoded with one-hot encoder, which converts categorical variables into numeric format, representing each category with 0s and 1s.

3.3 Models

Two models were employed:

3.3.1 Logistic Regression

Logistic regression imposes no assumption on independent features and is particularly suited for frequency-based features, making it a popular choice in NLP. However, logistic regression still assumes independence between instances. Moreover, it is constrained to linear-separable classification problems and may suffer from overfitting when the number of features is larger than the number of observations. Given the 100,000 training instances, this number outweighed the number of features after vectorizing and encoding. To mitigate overfitting further, I utilize L2 regularization, which adds a penalty to the loss function, encouraging smaller but non-zero weights to avoid unnecessary complexity. The maximum number of iterations is set to 100 for time efficiency despite the optimal accuracy happening when the iteration equals 500 and the accuracy difference is minor.

3.3.2 Multi-layer Perceptron (MLP)

MLP model is able to handle non-linear separable and more complicated problems compared to linear classifiers. Additionally, MLPs can capture the intermediate relationships between features. I applied a relatively simple architecture with only two hidden layers and the rectified linear unit (ReLU) activation function to balance computation efficiency and complexity. The first hidden layer consists of 64 nodes, followed by 32 nodes in the second hidden layer. The learning rate was set to be 0.001 only to prevent overfitting.

3.3 Evaluation methods

Four indicators and one evaluation metrics are applied to evaluate the results.

- **Accuracy**

The proportion of instances that the model correctly classifies out of all instances. While accuracy is straightforward for assessing performance, it needs to be interpreted carefully, especially for the existing imbalanced class distributions here.

$$Accuracy = \frac{Correctly\ classified\ instances}{Number\ of\ all\ instances}$$

- **Precision**

The proportion of instances predicted to belong to a specific rating category and actually in that category. Let's say a viewer wants to watch a movie in the highest rating category. If the model classifies several low-rating movies to the top category, the viewer will have a high chance of wasting time on undesirable low-scored movies.

$$Precision = \frac{TP}{TP + FP}$$

- **Recall**

The proportion of instances with a rating in reality that are accurately predicted to that category by the model. Considering viewers' limited time and preference of watching top-rated movies, recall ensures that high-quality movies are not overlooked due to misclassification.

$$Recall = \frac{TP}{TP + FN}$$

- **F1-score**

Derived from precision and recall. An ideal model should have a high F1-score since it has high precision and recall simultaneously.

$$F1 = \frac{2 * Precission * Recall}{Precision + Recall}$$

- **Confusion metrics**

Visualise the result with colours and enable inspection of the overall TP, FP, TN and FN.

3 Result

Hypothesis 1: *release_year* is a good feature and adding it to the feature set can improve the performance of the logistic regression model.

Table 3, Figures 3 and 5 are the results of training the logistic regression model only on text features. While Table 4, Figures 4 and 6 depict the results of adding *release_year* to the feature set and training a new model.

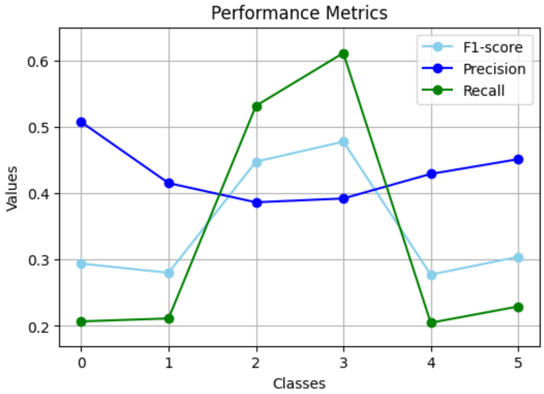

Figure 3 - *Accuracy, precision, recall and F1-score of the logistic regression model trained on the text features.*

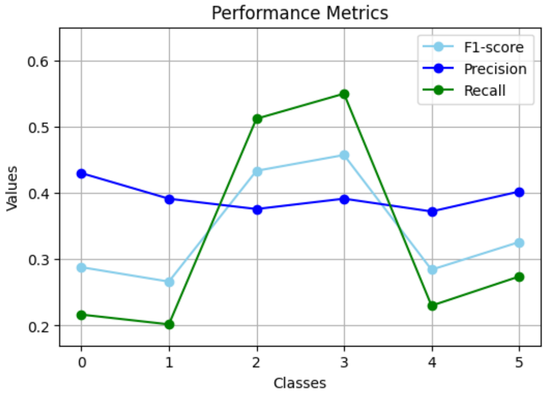

Figure 4 - *Accuracy, precision, recall and F1-score of the logistic regression model trained on the text features and release_year.*

Metric	0	1	2	3	4	5	Overall accuracy
Precision	0.51	0.42	0.39	0.39	0.43	0.45	0.40185
Recall	0.21	0.21	0.53	0.61	0.20	0.23	-
F1-score	0.29	0.28	0.45	0.48	0.28	0.30	-
Number of instances	2184	2829	5119	5420	2791	1657	20000

Table 3 - *Accuracy, precision, recall and F1-score of the logistic regression model trained on the text features.*

Metric	0	1	2	3	4	5	Overall accuracy
Precision	0.43	0.39	0.38	0.39	0.37	0.40	0.38705
Recall	0.22	0.20	0.51	0.55	0.23	0.27	-
F1-score	0.29	0.27	0.43	0.46	0.28	0.33	-
Number of instances	2184	2829	5119	5420	2791	1657	20000

Table 4 - *Accuracy, precision, recall and F1-score of the logistic regression model trained on the text features and release_year.*

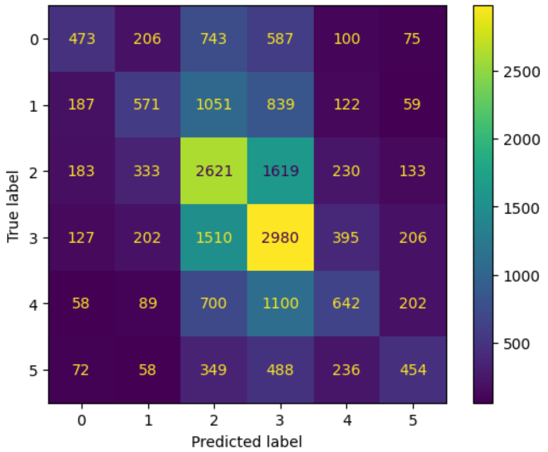

Figure 5 – *Confusion metrics of the logistic regression model trained on the text features.*

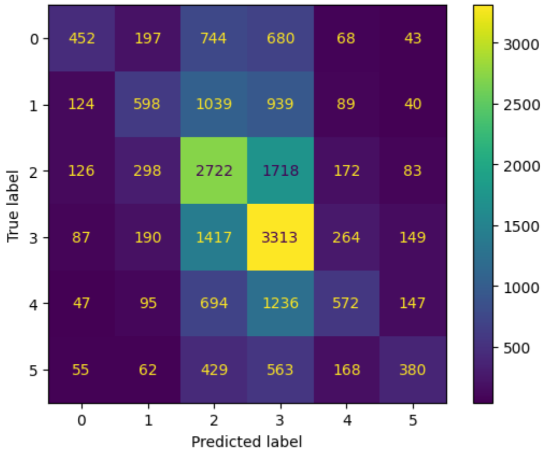

Figure 6 – *Confusion metrics of the logistic regression model trained on the text features and release_year.*

When including *release_year*, the changes in performance weren't significant. The accuracy declined by less than 2%; the precision and recall fluctuated slightly – neither the increase nor decrease exceeded 8%. There are three potential explanations for these observations:

1. release_year is an irrelevant feature:

The results might namely suggest that *release_year* didn't impact the model's performance positively but introduced noise, resulting in a slight drop in performance. This aligned with the findings of Abarja, R. A., and Wibowo, A. (2020), who didn't include the release year of movies in the optimal model probably due to its irreverence [3].

2. Multicollinearity between the text features and *release_year*:

Logistic regression assumes no significant multicollinearity between features. The decrease in accuracy might indicate that *release_year* and text features have multicollinearity. Given the TMDB dataset that spans over 100 years of movie data, it is plausible that the language usage in titles, taglines, and overviews has evolved; some words frequently used decades ago are obsolete. As a result, this strong multicollinearity affected the model's ability to converge.

3. Non-linear decision boundary between features and targets:

Logistic regression is constrained by its ability to capture only linear relationships between features and labels. The relatively poor performance, only 40% accuracy and unstable F1-score, of the first logistic regression model might imply that the relationship between the text and label was already too complex for this linear classifier. Including *release_year* further increased this relationship's complexity, surpassing logistic regression's capabilities. Addressing this issue may require more complex models, as noted by Bhadrashetty, A. and Patil, S. (2024, January), who deployed linear classifiers and concluded that predicting movie ratings based on text is challenging [2].

To further explore the possible non-linear relationship between the features and label or multicollinearity between the text features and *release_year*, I opted to train a MLP classifier.

Hypothesis 2: *release_year* is a good feature and adding it to the text features can increase the performance of the MLP model.

Table 5, Figures 7 and 9 are the results of training the MLP model only on text features. Table 6, Figures 8 and 10 illustrate the results of adding *release_year* to the feature set and training a new MLP model.

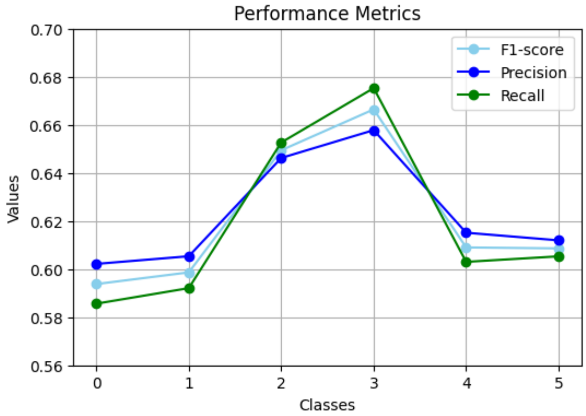

Figure 7 - *Accuracy, precision, recall and F1-score of the MLP model trained on the text features.*

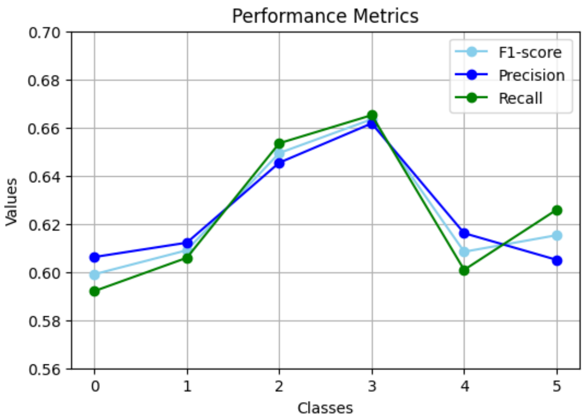

Figure 8 - *Accuracy, precision, recall and F1-score of the logistic regression model trained on the text features and release_year.*

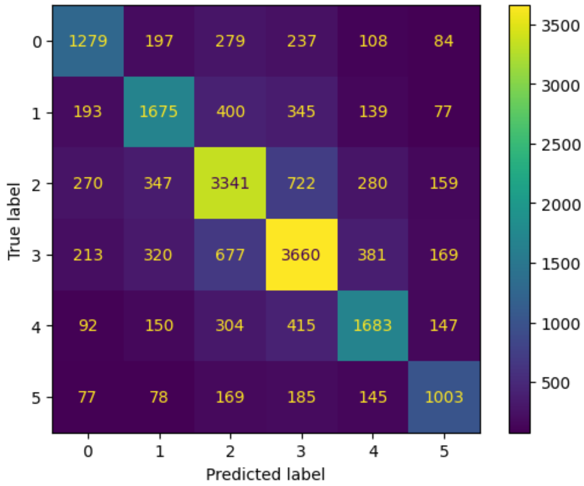

Figure 9 – *Confusion metrics of the MLP model trained on the text features.*

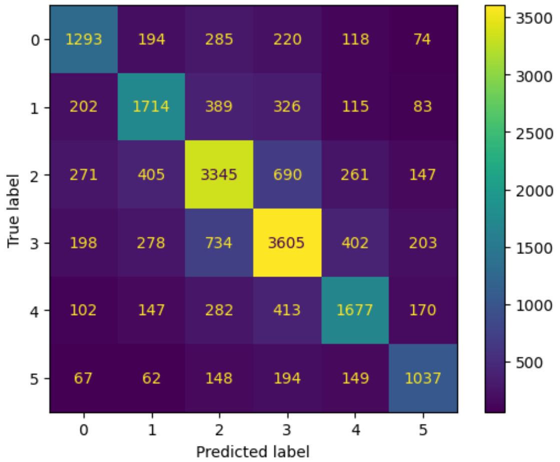

Figure 10 – *Confusion metrics of the MLP model trained on the text features and release_year.*

Metric	0	1	2	3	4	5	Overall accuracy
Precision	0.60	0.61	0.65	0.66	0.62	0.61	0.63205
Recall	0.59	0.59	0.65	0.68	0.60	0.61	-
F1-score	0.59	0.60	0.65	0.67	0.61	0.61	-
Number of instances	2184	2829	5119	5420	2791	1657	20000

Table 5 - *Accuracy, precision, recall and F1-score of the MLP model trained on the text features.*

Metric	0	1	2	3	4	5	Overall accuracy
Precision	0.61	0.61	0.65	0.66	0.62	0.61	0.63355
Recall	0.59	0.61	0.65	0.67	0.60	0.63	-
F1-score	0.60	0.61	0.65	0.66	0.61	0.62	-
Number of instances	2184	2829	5119	5420	2791	1657	20000

Table 6 - *Accuracy, precision, recall and F1-score of the MLP model trained on the text features and release_year.*

The accuracy, precision, recall, and F1-score improved less than 1% after adding *release_year*. Combining this observation with the previous findings, I could infer:

1. ***release_year* is an irrelevant feature:**

Similar to what the logistic regression models suggested, the marginal increase in performance depicts, again, *release_year* might be redundant and did not significantly contribute, either positively or negatively, to the model's performance.

2. **Medium generalizability due to data imbalance:**

Figure 10 reveals imbalances in the distribution of *release_year* across classes; over 70% of movies produced in the last 30 years. The data on movies released before 1988 is relatively inadequate. This fact limited both types of models as they had less information for earlier movies and could be one of the reasons for medium generalizability.

Additionally, regardless *release_year*, the differences between precision and recall of each class in the MLP models are smaller. This might show that the dramatic differences in these indicators in the logistic regression models were caused by their ability to handle linear-separable problems only.

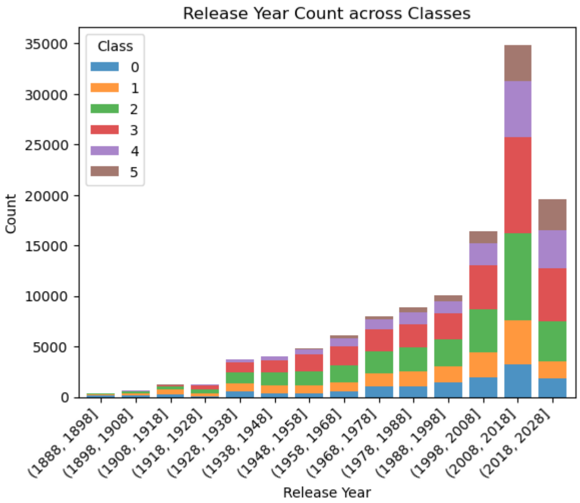

Figure 10 - Distribution of *release_year* in each class.

4 Conclusions

Through these two experiments, it's hard to tell that the release year of movies truly benefits the prediction of movie ratings based on the text features - title, tagline, overview, and production companies. Adding *release_year* may introduce noise, undermining the performances of logistic regression and MLP models. Moreover, the number of training instances indeed affects the results since we saw from Figure 1 that more instances belong to class 2 and 3. This pattern led to higher precision and recall for these two classes.

It's worth noting that *release_year* may still be a relevant feature, but the applied models failed to effectively capture its relationship with the classes, possibly due to the need for further hyperparameter tuning. Several future steps can be considered:

1. Alternative encoders for *release_year*: For example, given the inherent chronological order of years, an ordinal encoder may be a better choice.
2. Hyperparameter Tuning: The performance improvement observed when utilising the MLP model suggested that non-linear classifiers might be more suitable for this dataset. Fine-tuning the hyperparameters, including learning rate, activation function, number of hidden layers, and nodes per layer, could optimise the model performance.

References

1. C. Zhan, J. Li and W. Jiang, "An Empirical Investigation on Movie Industry from 1980 to 2018," *2018 IEEE Symposium on Product Compliance Engineering - Asia (ISPCE-CN)*, Shenzhen, China, 2018, pp. 1-4, doi: 10.1109/ISPCE-CN.2018.8805813. https://ieeexplore.ieee.org/abstract/document/8805813
2. Bhadrashetty, A., & Patil, S. (2024). Movie Success and Rating Prediction Using Data Mining. *Journal of Scientific Research and Technology,* *2*(1), 1–4. https://doi.org/10.61808/jsrt78
3. Aditya, R. A., & Wibowo, A. (2020). Movie Rating Prediction using Convolutional Neural Network based on Historical Values. *International Journal of Emerging Trends in Engineering Research*, 8(5), 2156-2164. http://www.warse.org/IJETER/static/pdf/file/ijeter109852020.pdf