SENG 474 Final Project- IMDb Rating Prediction System

Nolan Kurylo	Jayden Chan	Kahvi Patel	Ahnaf Ahmed
V00893175	V00898517	V00895917	V00889954
nolankurylo@uvic.ca	jaydenc7@uvic.ca	iamkahvi@uvic.ca	ahnafa@uvic.ca

Abstract

IMDb is one of the most prominent resources for finding information about movies and the people involved in their production. Additionally, the IMDb database contains user and critic ratings for each movie along with various other metrics that contribute to the movie's IMDb rating. [1]

The focus of this project is to apply various machine learning models to the IMDb database in order to predict its rating. The goal of applying these models is to predict the ratings of unreleased movies. The predicted rating for a movie will coincide with IMDb's 0-10 scale to one decimal point. To create the prediction, it is assumed that a user will supply all characteristics regarding the movie.

1 Dataset

The dataset used for this project is sourced from Kaggle and contains all movies in the IMDb database with more than 100 votes, as of January 1st, 2020. The dataset is split into four CSV files: movies, ratings, names of actors/actresses, and other miscellaneous information. This dataset was chosen for its size (85,855 movie examples) and the diversity of its attributes. The column chosen to determine the output of model training, also known as the label vector, is the column `weighted_average_vote`. This column serves as the label vector for each of the supervised machine learning algorithms applied in experimentation. [2]

2 Data Munging

This section discusses the methods and implementation techniques used to combine and clean the dataset before building the machine learning models. After EDA on the input CSV files, the cleansed dataset was outputted into `combined_data.csv`, ready to be used for building the various models.

2.1 Data Merging

After merging the names and movies datasets, there are 17 features. Ignoring our label vector and the `movie_title` column, there are 10 categorical features and 5 continuous features remaining. Additionally, after removing all rows with any `NaN` values, there are 15,688 examples remaining from our original 85,855. Next, the continuous features are *scaled* and the categorical features are *encoded*.

2.2 Scaling Continuous Features

The continuous features `duration`, `budget`, `height` and `divorces` were scaled by their mean and to unit variance.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

2.3 Encoding Categorical Features

The categorical features to encode in the dataset are `genre`, `country`, `language`, `director`, `writer`, `production_company`, `top_actor`, `top_actor_gender` and `year`.

For the `top_actor_gender` and `year` columns, it's obvious that the number of columns added to the dataset will be reasonable since there are a small, finite number of genders (2 for our purposes) and years (around 80) for each movie. Therefore, a normal one hot encoding of all unique values will suffice.

2.3.1 Analyzing Large Categorical Features

Unfortunately, the other categorical features are not as well behaved. Analysis was done to describe the diversity or specificity of each large categorical column. To describe each column, every unique category either contains only 1 movie, between 1 and 10 movies, between 10 and 100 movies or over 100 movies. The following pie charts describes that distribution.

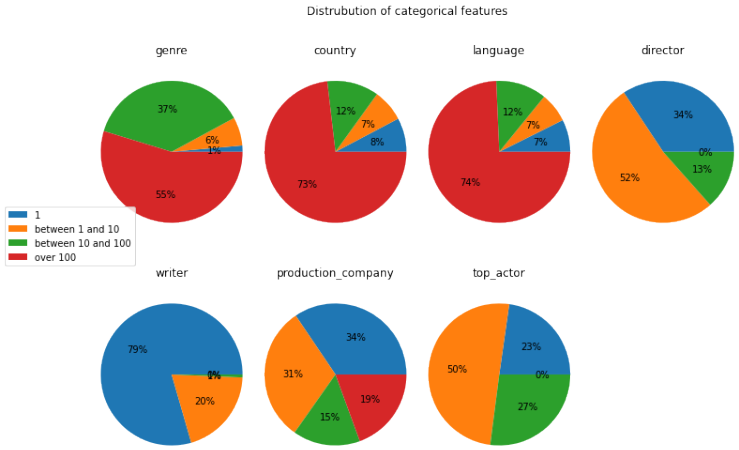


Figure 1: Pie charts describing the distribution of the larger categorical features.

As shown above, some columns have too many unique values and would take excessive amounts of time to perform the full one hot encoding conversion. Due to this, it was decided to only one hot encode some values instead of all of them. By looking at the frequencies of the unique values for a specific column, only the top N frequent values were one hot encoded.

Five different datasets were constructed, each with the same number of rows, but with differing levels of encoding. First no features were encoded, resulting in only continuous variables. Then only the smaller categorical features were encoded: `top_actor_gender` and `year`. Finally, the smaller categorical features and only the top 10, 100 and 500 most frequently occurring categories of the larger categorical features, like `genre` and `top_actor`, were encoded.

3 Machine Learning Models

Running a simple neural network on the 5 different datasets indicated that encoding is helpful, but the degree to which the categorical features are encoded is not significant. The mean absolute error (MAE) metric seen below was used to analyze the how well each of these models performed as well as the other models in this section. The difference in MAE between the top 10 versus the top 500 most frequently occurring categories is less than 0.05. Importantly, training the neural network on the top 500 dataset takes much longer compared to the top 10 dataset given the increased number of columns (198 vs 828). Therefore, to save processing time, only the top 10 and top 100 most frequently occurring categories were encoded.

$$MAE = \frac{1}{N}\sum_{i}|y_i - \hat{y}_i| \tag{1}$$

3.1 Simple Neural Network

In attempt to configure the rating prediction of the IMDb dataset, a supervised learning neural network was constructed on the `combined_data_all_list_encoded_100.csv` preprocessed dataset. The neural networks in this section were basic in that they involved a single input layer, hidden layer and output layer using `Keras`[5]. Both classification and regression techniques are explored below.

3.1.1 Regressor

The regressor neural network is designed to output a rating result in the range of 0-10, to one decimal point. During training, the goal is to find the optimal number of hidden neurons to be used in the hidden layer by applying various researched methods. The following methods were tested by Jinchuan and Xinzhe (n_{h1}), Trenn (n_{h2}), and Shibata and Ikeda (n_{h3}) respectively:

$$n_{h1} = \frac{n_i + \sqrt{N}}{h_n} \qquad n_{h2} = n_i + n_o - \frac{1}{2} \qquad n_{h3} = \sqrt{n_i n_o} \tag{2}$$

where n_{hi} is the number of neurons in the hidden layer for method i, n_i is the number of neurons in the input layer (n_i = number of dimensions in the training set), N is the number of samples in the training set, n_o is the number of neurons in the output layer ($n_o = 1$ for regression) and h_n is the number of hidden layers ($h_n = 1$ for this single layer neural network). [3]

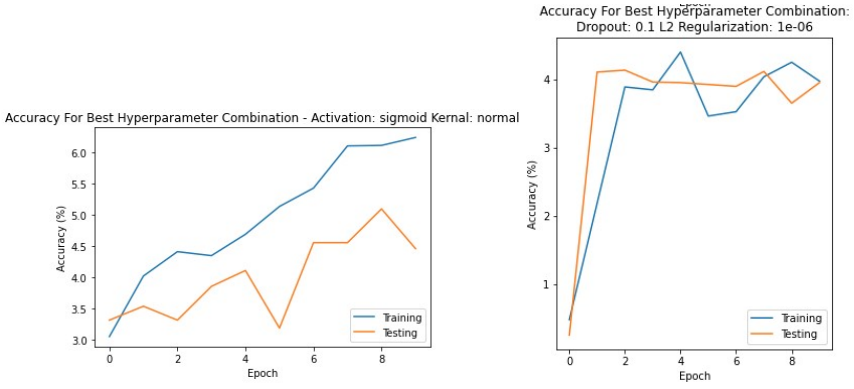
It was determined that Trenn's formula provided model that achieved the best MAE with 748 hidden layer neurons; the resulting validation MAE of 0.72 was an acceptable output.

3.1.2 Overfitting Classifier

The prediction rating output range of 0-10 can be converted into 100 discrete classes for each single decimal point value, followed by descaling after prediction. In this way, the same neural network can be tested as a classifier instead of a regressor.

The goal of this model was to identify the best combination of activation function (values: Sigmoid, ReLU) and kernel initializer (values: Normal, He Normal) hyperparameters that would result in the model's highest validation accuracy.

The result of the classifier led to an understanding that this type of model should not be used for this problem. With low validation accuracy (5%), extensive overfitting and outlier fluctuations in resulting plots (Figure 2), this model was problematic.



(a) (b)

Figure 2: Best hyperparameters for validation accuracy after 5-fold CV (a) and the accuracies of the overfitting improvement (b).

3.2 Deep Neural Network

To properly decide how to construct a `Keras` deep neural network there was a lot of hyperparameter tuning that needed to be done. The hyperparameters chosen to adjust were the activation functions, dropout values, batch sizes, and regularizations values. There was a prior preliminary tuning done on the number of hidden layers and number of nodes per layer. This was performed with GridSearchCV and came out to 2 hidden layers and dim*2, dim, 50, and 1 node(s) for each layer respectively.

Using the basic activation functions of ReLU, Sigmoid, tanh, eLU, along with 3 each of regularization, dropout, and batch size represented a large number of models to be created. Initially this was much larger, in the 16,000 range for number of models needed, however it was trimmed after a single run on one dataset took 8 hours. It was deemed more beneficial to work on testing multiple preprocessed datasets than doing such extensive training on the hyperparameters. Even so there seemed to be a hard lower limit reached. As can be seen in figure 3 even with 1200 models, they never went very far lower than 0.75 for testing error.

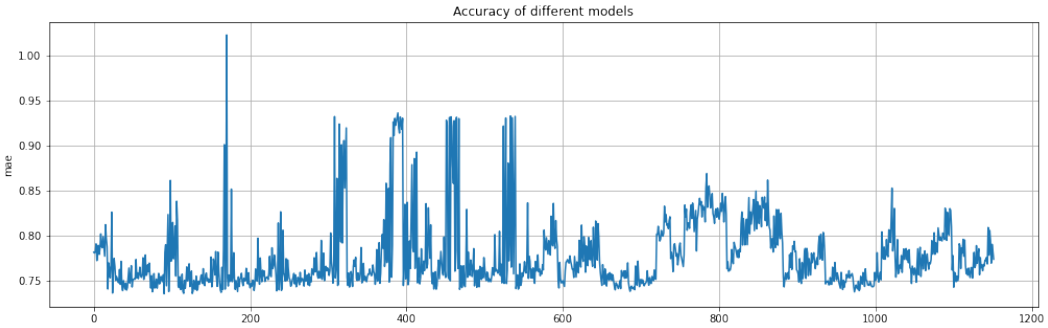


Figure 3: Subset of MAE values of various models.

In the end a model with ReLU, Sigmoid, tanh, and eLU produced the most consistent results which can be seen in Figure 3. Interestingly, for the most part dropout values at best did not affect the model and usually made it perform worse. L2 regularization of 0.01 on just the Sigmoid layer also increased consistency.

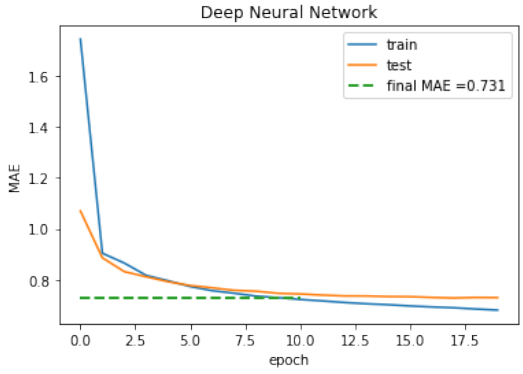


Figure 4: A plot of the MAE values of the deep neural network.

3.3 Decision Tree

The third model tested was a decision tree classifier which was inferred from the training data using the ID3 algorithm. The previously-written code from Assignment 1 [4] was used to construct a decision tree for predicting the rating of a movie given a set of categorical features. The algorithm needed to be modified to support outputting any number of classes instead of the two true/false output

classes that were used in the assignment. This involved removing any references to true/false or 0/1 in the code, as well as rewriting the feature Entropy calculation to work with any number of output classes.

For the initial test, the training data set ratings were rounded to the nearest whole number, giving 10 output classes for the decision tree. The testing data set was not rounded. After training on a 70/30 split of the data, the resulting decision tree achieved an MAE of `0.96` with the following features enabled: `top_actor_gender`, `height`, and `divorces`. The features were selected by the algorithm automatically based on the number of unique values for the feature (in this case the limit was `80` unique values). Although the selected features seem surprising, increasing this limit resulted in a worse MAE overall regardless of which features were selected.

Rounding the scores to the nearest half-number gave 20 output classes for the tree. This change resulted in slightly worse scores, achieving an MAE of `0.97`.

3.3.1 Random Forest

After the simple decision tree was tested, the data set was used to train both single decision trees and random forests using the `sklearn` library[6]. The `sklearn` decision tree achieved an MAE of `1.32` when trained on the same feature set as the ID3 tree. Including additional features in the training resulted in worse scores.

500 random forests were generated using between 50 and 100 trees for each forest. For each forest, a random subset of the available training features were selected to be used. The results of the training are shown below:

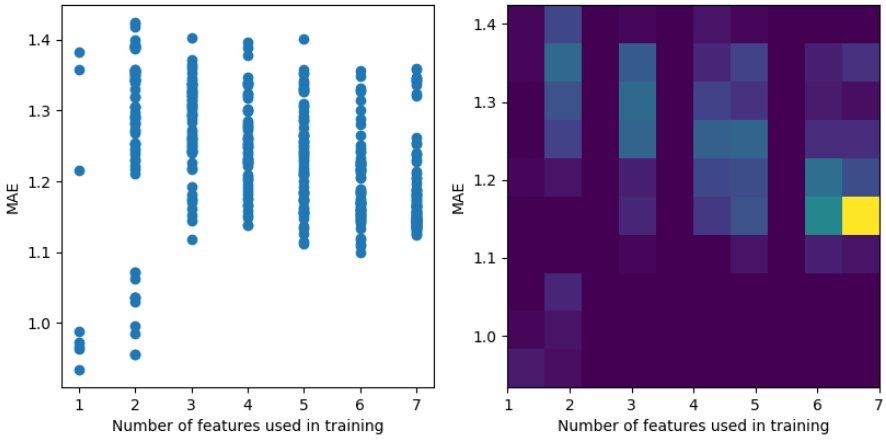


Figure 5: Number of features used for training vs MAE

The best MAEs are achieved with only one or two features. Adding more features results in a sharp increase in MAE which slowly decreases as more features are added. However, even when including nearly all of the features the results are worse than simply using one or two.

4 Results

4.1 Neural Networks

After testing the Simple Neural Network regressor with 5-fold cross validation (CV) and analyzing the results of each method, the Trenn and Jinchuan and Xinzhe methods proved to be almost identically effective in determining the number of neurons needed in the hidden layer, as seen in Figure 6 below. This led to a 0.72-best MAE for the Trenn method, with slight overfitting occuring. This is seen as the validation curve for both Trenn and Jinchuan and Xinzhe methods flatten out quickly, meaning that the MAE has reached a bottleneck and will not improve much more. It is also worth noting that

each trial of these models were trained with the Sigmoid activation function for the hidden layer and HeNormal initialization for the weights.

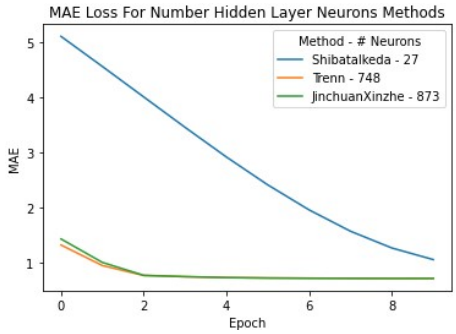


Figure 6: Best Validation MAE for each method after 5-fold CV.

After training the Simple Neural Network classifier, it is evident that some overfitting has occurred on the validation set. Seen in Figure 2a, The accuracy for the training curve separates away from the testing curve implying that the data is being learned "too well" on many outliers by the model during the training phase. The outcome of tuning an L2 regularizer (values: 0.0001, 0.000001) and Dropout layer (values: 0.7, 0.3, 0.1) onto the hidden layer helped control the overfitting (Figure 2b). While this does yield less overfitting, the validation results turn out to be highly fluctuating due to outliers, meaning that the classification model will be problematic in production. Based on overfitting of outliers and low accuracy of the classification model, the regressor will be the better choice.

As can be seen in Figure 4 the deep learning model worked out to be within 0.736 of the true value on average. This MAE itself could vary on different trainings of the model due to the randomness of the train test split function. The general range would go from as low as 0.71 to 0.74. However, in some more rare cases it could reach an MAE of 6. The model would only return values under 1 or above 6, leading to the possibility that there are a large number of outliers in the dataset. This can be solved by acquiring more data to improve the model or more advanced feature engineering.

4.2 Decision Trees

The results of the decision tree methods suggest that the number of unique values for the categorical features in the data set is simply too high. The decision tree-based models performed significantly better when trained with features that had the fewest unique values. This is reflected most obviously in the results of the ID3 algorithm, which selected a highly unexpected feature set to train on. Despite this, it still achieved the best MAE of any decision tree-based method.

5 Conclusion

After exploring the various types of supervised machine learning models, 5-fold cross validation and 70/30 training/testing splits from the dataset were applied. The best achieving MAE was found to be from the The Simple Neural Network from Table 1, and is thus the choice for the Prediction System.

It should be noted that while this model had the best error rate, future work could explore different approaches to improve the error. The results show that the way categorical data was dealt with was troublesome; alternatives could include hash encoding, binary encoding or embedding layers.

Table 1: Model Validation Results.

Model	Best MAE
Simple Neural Network	0.72
Deep Neural Network	0.73
Decision Tree	0.95
Random Forest	0.96

References

[1] "IMDb: Ratings, Reviews, and Where to Watch the Best Movies & TV Shows", IMDb, 2021. [Online]. Available: https://www.imdb.com/. [Accessed: 13- Aug- 2021]

[2] "IMDb movies extensive dataset", Kaggle.com, 2021. [Online]. Available: https://www.kaggle.com/stefanoleone992/imdb-extensive-dataset?select=IMDb+names.csv. [Accessed: 13- Aug- 2021]

[3] K. Sheela and S. Deepa, "Review on Methods to Fix Number of Hidden Neurons in Neural Networks", Mathematical Problems in Engineering, vol. 2013, pp. 1-11, 2013. [Accessed: 13- Aug- 2021]

[4] J. Chan, “ID3 Algorithm Implementation.” Victoria, BC, 06-Jun-2021. [Accessed: 13- Aug- 2021]

[5] K. Team, "Keras: the Python deep learning API", Keras.io, 2021. [Online]. Available: https://keras.io/. [Accessed: 13- Aug- 2021]

[6] "scikit-learn: machine learning in Python — scikit-learn 0.24.2 documentation", Scikit-learn.org, 2021. [Online]. Available: https://scikit-learn.org/stable/. [Accessed: 13- Aug- 2021]