**Predicting IMDB Score**

**Pranav Srivastava**

**Abstract** The main objective of this project is to follow steps in a machine learning lifecycle to build a model, based on the IMDB dataset, that predicts IMDB score of a movie.

Summary of Results : Binary classification, multi-class classification and regression model.

**1** **Introduction**

I have chosen IMDB dataset from Kaggle for this study. It contains 28 features and 5043 instances. In the table below you can see the list of features. “imdb\_score” at index 26 is our target class. Rest of the features are the predictors.

|  |  |
| --- | --- |
| index | Features |
| 1 | color |
| 2 | director\_name |
| 3 | num\_critic\_for\_reviews |
| 4 | duration |
| 5 | director\_facebook\_likes |
| 6 | actor\_3\_facebook\_likes |
| 7 | actor\_2\_name |
| 8 | actor\_1\_facebook\_likes |
| 9 | gross |
| 10 | genres |
| 11 | actor\_1\_name |
| 12 | movie\_title |
| 13 | num\_voted\_users |
| 14 | cast\_total\_facebook\_likes |
| 15 | actor\_3\_name |
| 16 | facenumber\_in\_poster |
| 17 | plot\_keywords |
| 18 | movie\_imdb\_link |
| 19 | num\_user\_for\_reviews |
| 20 | language |
| 21 | country |
| 22 | content\_rating |
| 23 | budget |
| 24 | title\_year |
| 25 | actor\_2\_facebook\_likes |
| 26 | imdb\_score |
| 27 | aspect\_ratio |
| 28 | movie\_facebook\_likes |

I have focused on following predictions:

**binary classification** – predict if the movie is good or bad

**multi-class classification** – predict if the movie if bad, average, good, excellent

[0,4,6,8,10]

**regression** – predict the movie rating

Many people believe that movies are reflection of the society. The motivation to build models using this dataset is to predict class of the movie using classification or exact rating of the movie using regression. Knowing the score helps a user to have an idea about the movie before they start watching it or buying the tickets. Preprocessing of data, feature selection, model selection, model evaluation, hyperparameter optimization for tuning best models are the important phases in machine learning lifecycle and are implemented using scikit-learn in this project.

There are many academic papers available on internet that used this dataset. I found an interesting paper from Jackson State University with name “Predicting Movie Success Using Machine Learning Algorithms”. Paper is available here/reference: <http://www.laccei.org/LACCEI2017-BocaRaton/student_Papers/SP499.pdf>

This paper proposes a way to predict how successful a movie will be prior to its arrival at the box office. It is a multi-class classification problem with five categories for the “success rate” of movie. The interesting part is that they used two datasets (IMDB and YouTube) and merged it into one, in order to build a model that can achieve a better prediction.

**2 Research**

Feature Selection is one of the most important steps in a machine learning lifecycle. In a dataset every column is a feature. Few features are correlated with the target class and many irrelevant features are not correlated with the target class. In other words, only few features will have an impact on the output variable. If the irrelevant features are not removed, then it is highly probable that they will make the model perform worst. Hence, there comes a need to carry out feature selection.

I have implemented

Univariate feature selection:

Selected features=['num\_voted\_users', 'num\_user\_for\_reviews', 'movie\_facebook\_likes', 'num\_critic\_for\_reviews', 'duration', 'title\_year', 'director\_facebook\_likes', 'gross', 'color', 'language', 'aspect\_ratio', 'actor\_2\_facebook\_likes']

Tree based feature selection:

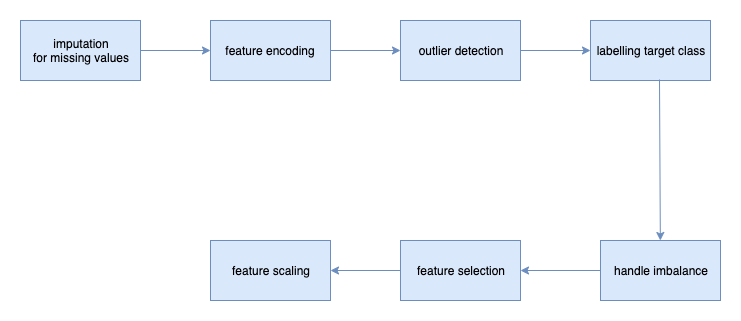
Selected features = ['num\_voted\_users', 'duration', 'genres', 'num\_critic\_for\_reviews', 'title\_year', 'num\_user\_for\_reviews', 'gross', 'budget', 'director\_facebook\_likes', 'movie\_facebook\_likes', 'cast\_total\_facebook\_likes', 'actor\_3\_facebook\_likes']

Greedy feature selection:

Selected features = ['director\_name', 'num\_critic\_for\_reviews', 'duration', 'director\_facebook\_likes', 'actor\_3\_facebook\_likes', 'actor\_2\_name', 'actor\_1\_facebook\_likes', 'genres', 'num\_user\_for\_reviews', 'title\_year', 'actor\_2\_facebook\_likes']

**3 Methodology**

**3.1 Sequence of preprocessing steps**



I undertook the preprocessing steps mentioned in the figure above. Brief explanation for choosing them -

1. Imputation – as there were a lot of missing values but were less than 30% in each row or column I chose not to drop them but impute them. I used SimpleImputer class in combination with ‘most\_frequent’ and ‘mean’ strategy parameter for string and integer feature columns respectively.
2. Feature encoding – it was required because few feature columns had string values
3. Outlier detection – I used it to check if there were major outlier in the dataset. However, I did not implement and handling techniques because I did not see major outliers in the dataset.
4. Labeling target class – this was necessary as I was aiming for multi-class classification problem. I binned the ‘imdb\_score’ column into 4 quartiles/labels (bad, average, good, excellent).
5. handle imbalance – after labeling the target class I detected imbalance in the target class and handled it using SMOTE
6. Feature selection – this is also my research topic in section 2. In order to reduce the complexity of model, reduce overfitting I applied this step.
7. Feature scaling – as my dataset was having huge variation in the feature column so I used this step to normalize the feature columns. This helped in having correct evaluation of models which are heavily dependent on distance metric like kNN.

3.2 Range of models and hyper-parameter optimization techniques

I used following models

**4 Evaluation**

Before SMOTE is applied

Confusion matrix before the scaling

[[ 3 15 18 0]

[ 4 156 139 0]

[ 3 85 531 4]

[ 0 0 28 22]]

Accuracy = 70.3373015873016

5 Conclusions and Future Work