Machine Learning Engineer Nanodegree

Capstone Proposal

Jude Chen   January 25, 2018

Proposal

Domain Background

Movie database such as IMDb has been well-known of collecting information from films including cast, storyline, production company, budget and technical specs. The audiences can share their reviews online and rate the film from 1 to 10 and because anyone registered in the website can contribute to the final rating which is considered relatively objective. The filmmakers may want to know what is the reason to make a movie successful? In some cases, a film with small budget and unknown actors may surprisingly create a big success. We, the movie fans, want to judge which movie should I spend a great time watching? Sometimes, the movie genre or plot summary may be interesting to us, but it doesn't mean we will like this movie.

There were some paper used machine learning to study in this area. [1] is to create a movie recommender system based on Decision Trees. A decision tree is trained for the user content preference of different movie story like comedy, action and horror. The recommender also takes credit information such as genre, director and starring into consideration for their output rating. Another movie rating research is that [2] using Regression Tree and multilayer feedforward neural network separately trained to represent a user's rating with the features of genres, actors and directors. Given an enormous amount of movie data each requires only the basic movie information for the training model, it seems possible for us to predict whether it is a good movie, even we have not yet watched them. The production company seems can find out the key to make a successful movie, and it is what we are willing to see.

Problem Statement

The movie rating is the target we want to predict. We want to train a model to learn from the movie data, to dig out the features of the movie basic information which lead to a successful movie and the result will be represented as the movie rating. How to define a successful movie? We want to use 7.0 as the boundary of successful and ordinary movies which split the good movies and general movies into 20% and 80%. In addition to the Decision Tree and neural network introduced from the previous research, we plan to use more supervised learning model such as Naive Bayes, Support Vector Machine to solve this problem. This problem is quantifiable and measurable because the target rating can be represented as a continuous number and we are going to transfer it into a binary classification problem as successful movie or not. The replicability can be proved by testing the datasets multiple times to get the rating. New datasets can also be added into testing to improve the reliability of model.

Datasets and Inputs

The datasets are from Kaggle, [3] collected 5000 movies from TMDb. Due to the copyright concern, Kaggle replaced their movie datasets from IMDb to The Movie Database (TMDb) with their terms of use declaration [4]. There are 2 parts in the datasets - credits and movies. The ‘credits’ part includes movie id, title, cast and crew. We will focus on the main actors in the cast data and also the director in the crew data. The ‘movies’ part includes title, budget, genres, homepage, keywords, spoken language, overview, popularity, production companies, release date, revenue, vote average, vote count and so on. Some less related columns like ‘homepage’ will be excluded from training.

Preprocessing for the datasets is needed. Some columns like ‘cast’ and ‘crew’ are in json format, we should do preprocessing to flatten the input data. The training sets and testing sets will be extracted from these 5000 datasets in some portion of 75% v.s 25%. The ‘vote average’ will be treated as our target value and be turned into a classification value 1/0 representing whether the vote is >=7.0 or <7.0. All the mentioned movie information may in some extent affect on the movie rating, what we need to do is to figure out the relationship hidden in these large data.

Solution Statement

We want to utilize supervised learning to solve the problem of predicting the movie rating. Ensemble methods like AdaBoost and Random Forest will be used to train the datasets including the feature importance attribute to analyze the most relevant features which affect the movie rating. Besides, we will use other supervised learning model like Decision Tree and XGBoost to train the datasets, the testing results of the trained models will be compared. Dealing with the data preprocessing, we will use feature scaling to solve the data skewing problem, some input column in json format will be transformed by one-hot encoding and use ‘Word Cloud’ to pick the most relevant words into training.

Benchmark Model

Random predict: We choose a random number between 0 and 1 as the predict rating for each movie. This benchmark is a naive and simple one without data training which may result to a lower testing accuracy, our training model should outperform this benchmark.

Decision Tree: According to the paper mentioned, decision tree is a commonly used training model in predicting movie rating. Within data training, the decision tree model should perform a better testing accuracy. We will use other model like Random Forest or XGBoost to see if we can train a better testing accuracy than decision tree.

Evaluation Metrics

Accuracy: The percentage of movies correctly predicted as a successful movie.

F-beta score: We choose the beta = 0.5 which means we put more weights to precision. Because we care more about the correctly predicted successful movie which is rating over 7.0, we put more emphasis on the precision criteria, that is what we predicted as successful movie were actually rating > 7.0.

$$F_\beta = (1+\beta^2) \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{(\beta^2 \cdot \mathrm{precision}) + \mathrm{recall}}$$

F-beta score formula [5]

Project Design

Data Exploration: Use scatter matrix and heatmap to observe the data distribution and feature relationship.

Data Preprocessing:

- **Input data handling:** Includes feature and target data separation; target data encoding into 1 : rating >= 7.0 and 0 : rating < 7.0; null input handling
- **Json input handling:** Some columns given as json format will be flattening and using WordCloud to pick top most keywords for feature selection. These features will apply one-hot encoding as input.
- **Feature Scaling:** In order to solve data skewing problem, Log transformed and MinMaxScaler will be applied to format better feature distribution.
- **Training / Testing data separation:** Since we have 5000 movie datasets, we can split the data into Training data(70%) and Testing data (30%).

Learning Models: Besides the random predict benchmark, we also apply the referenced model - Decision Tree and the popular method in Kaggle competition - XGBoost.

- Decision Tree
- XGBoost

Feature Importance: Use Ensemble methods to train and utilize the attribute of feature importance to look into the feature relevant.

- AdaBoost
- Random Forest

Model Evaluation:

- Test Accuracy
- F-beta score

Reference

[1] P. Li and S. Yamada, “A movie recommender system based on inductive learning” in Cybernetics and Intelligent Systems, 2004 IEEE Conference on.
[2] M. Marovic, M. Mihokovic, M. Miksa, S. Pribil, and A. Tus, “Automatic movie ratings prediction using machine learning” in MIPRO, 2011 Proceedings of the 34th International Convention.
[3] https://www.kaggle.com/tmdb/tmdb-movie-metadata
[4] https://www.themoviedb.org/documentation/api/terms-of-use
[5] https://en.wikipedia.org/wiki/F1_score