PREDICTING MOVIE SUCCESS FROM SCRIPT ANALYSIS USING NLTK TOOLKIT

Antwerp

Diana, Tije, Andrew, Adedoyin

Contents

Introduction…………………………………………………………………………………………1

Background review/research……………………………………………………………………..2

Project specification……………………………………………………………………………….3

Problem Analysis…………………………………………………………………………………..4

Solution design……………………………………………………………………………………..5

Implementation……………………………………………………………………………………..6

Evaluations………………………………………………………………………………………….7

Summary/Conclusions……………………………………………………………………………..8

INTRODUCTION

Every movie enthusiast has favourite movies. There are certain criterium that makes movies likable. This criterium is often based on individual bias. Movies are often rated as “outstanding” or “boring” depending on personal criterium. At sources like IMDB, Movie enthusiasts rate movies with scores ranging from one to ten. These scores vary. For example, popular movies like “The Shawshank Redemption” receive a huge number of votes between 9 and 10 while poor movies have ratings between 1 and 3. From this voting we can conclude that there are some general conditions that determine how successful a movie will do. These conditions may be independent of subjective human bias

In the 21st century where data is considered the sexiest job in the century, data analysis could prove useful in understanding why some movies are received better than others. In movie ratings, movie enthusiasts often talk about camera angles, acting performances, and image quality. However, these qualities are not enough. In addition, it is difficult to find patterns in the features mentioned above. For instance, experienced actors could argue that acting makes a film great, while photographers find the image and video quality more appealing. In order to concretize movie quality, the analysis of movie scripts could prove useful. Writing and storytelling are important parts of a movie and are often mentioned in user reviews.

Script analysis could provide us with useful information. Patterns and features that could help and explain the difference between a good and a bad movie. Programming languages like Python allow us to analyse data sets quickly, with our current technologies. Looking for patterns in bad/good movies is something that hasn’t been done, or at least documented on the internet. This reason, and our love for movies, made us decide to pick this topic for our group project. Features like lexical diversity are useful considerations and can be calculated with Python. We chose the topic from the perspective of movie enthusiasts, as well as starting Python programmers.

We used the NLTK toolkit in Python, as well as the website of the Internet Movie Script Database (IMSDB) as tools for our research. The IMSDB website (<https://www.imsdb.com>) has an enormous amount of movie scripts ready to be downloaded. We compared movies from the same genre, in order to find differences between movies. In order to get a general idea about the movies in the database, we calculated the Lexical Diversity from the entire database of scripts. In total, 1092 scripts were scanned, and a lexical diversity score was calculated. This was done with the stop-words function, the word\_tokenize function and the Porterstemmer function. The words in every script were tokenized as individual words. All words were “stemmed” to their base words. For example; walking becomes walk, sleeping becomes sleep e.t.c. The number of individual stems were then divided by the amount of unique words to generate a diversity score. The higher this score, the more diverse the vocabulary of the movie script.

We then wrote the scores into a text file, so it can be used for later analysis.

SOLUTION DESIGN

38 features were extracted for 1000 movie scripts: 32 frequencies of speech parts, lexical diversity, average sentence length and number of votes on imdb.com. The rating of the movie was taken as a response. The dataset was split either in halves or randomly in various ratios into the training and the testing sets. Various regression algorithms were tested in order to find the model and use it for predictions.

RESULTS

The linear regression model is characterized by an adjusted R-squared value of 0.392 showing relatively low correlation between the features and the response. The most significant features in this model were determined: number of votes on the imdb.com, lexical diversity, sentence length and parts of speech.

TYPICAL OUTPUT FOR A RANDOMIZED 50/50 DATASET

|  |
| --- |
| Mean squared error: 0.73 |
|  | Variance score: 0.23 |
|  | OLS Regression Results |
|  | ============================================================================== |
|  | Dep. Variable: y R-squared: 0.445 |
|  | Model: OLS Adj. R-squared: 0.392 |
|  | Method: Least Squares F-statistic: 8.338 |
|  | Date: Sun, 13 Jan 2019 Prob (F-statistic): 1.22e-31 |
|  | Time: 17:59:28 Log-Likelihood: -472.38 |
|  | No. Observations: 434 AIC: 1023. |
|  | Df Residuals: 395 BIC: 1182. |
|  | Df Model: 38 |
|  | Covariance Type: nonrobust |
|  | ============================================================================== |
|  | coef std err t P>|t| [0.025 0.975] |
|  | ------------------------------------------------------------------------------ |
|  | const 2.3063 1.843 1.251 0.212 -1.318 5.930 |
|  | x1 2.19e-06 1.6e-07 13.711 0.000 1.88e-06 2.5e-06 |
|  | x2 -3.6616 2.105 -1.740 0.083 -7.799 0.476 |
|  | x3 28.5565 23.140 1.234 0.218 -16.936 74.049 |
|  | x4 48.8130 24.922 1.959 0.051 -0.184 97.810 |
|  | x5 45.7409 41.848 1.093 0.275 -36.532 128.014 |
|  | x6 26.9791 61.544 0.438 0.661 -94.016 147.975 |
|  | x7 -17.6602 10.587 -1.668 0.096 -38.473 3.153 |
|  | x8 -7.6554 9.712 -0.788 0.431 -26.749 11.438 |
|  | x9 -0.0879 8.802 -0.010 0.992 -17.392 17.217 |
|  | x10 -26.6006 16.467 -1.615 0.107 -58.974 5.773 |
|  | x11 24.0727 6.032 3.991 0.000 12.214 35.931 |
|  | x12 -16.0172 7.119 -2.250 0.025 -30.013 -2.022 |
|  | x13 23.3205 9.390 2.483 0.013 4.859 41.782 |
|  | x14 810.6948 880.824 0.920 0.358 -920.995 2542.384 |
|  | x15 199.5895 111.860 1.784 0.075 -20.326 419.505 |
|  | x16 -27.2954 14.452 -1.889 0.060 -55.707 1.116 |
|  | x17 7.3603 87.542 0.084 0.933 -164.746 179.466 |
|  | x18 7.1944 10.166 0.708 0.480 -12.793 27.181 |
|  | x19 -28.2152 182.857 -0.154 0.877 -387.709 331.279 |
|  | x20 188.6613 454.525 0.415 0.678 -704.929 1082.252 |
|  | x21 -4.8249 16.746 -0.288 0.773 -37.747 28.098 |
|  | x22 2.8920 8.719 0.332 0.740 -14.250 20.034 |
|  | x23 -136.9678 249.835 -0.548 0.584 -628.140 354.204 |
|  | x24 -54.8599 27.516 -1.994 0.047 -108.956 -0.764 |
|  | x25 -61.3896 106.323 -0.577 0.564 -270.419 147.640 |
|  | x26 -15.5728 150.616 -0.103 0.918 -311.682 280.536 |
|  | x27 204.6259 194.262 1.053 0.293 -177.291 586.542 |
|  | x28 -18.0820 22.256 -0.812 0.417 -61.836 25.672 |
|  | x29 6.6691 21.029 0.317 0.751 -34.673 48.011 |
|  | x30 33.0199 62.270 0.530 0.596 -89.402 155.442 |
|  | x31 5.3331 2.795 1.908 0.057 -0.162 10.828 |
|  | x32 96.7838 85.821 1.128 0.260 -71.940 265.507 |
|  | x33 5.0469 10.150 0.497 0.619 -14.908 25.002 |
|  | x34 -589.6085 280.644 -2.101 0.036 -1141.351 -37.866 |
|  | x35 14.5846 9.949 1.466 0.143 -4.975 34.144 |
|  | x36 6.4065 5.455 1.174 0.241 -4.318 17.131 |
|  | x37 14.4948 18.907 0.767 0.444 -22.677 51.666 |
|  | x38 0.0370 0.019 1.906 0.057 -0.001 0.075 |
|  | ============================================================================== |
|  | Omnibus: 62.573 Durbin-Watson: 2.133 |
|  | Prob(Omnibus): 0.000 Jarque-Bera (JB): 128.379 |
|  | Skew: -0.791 Prob(JB): 1.33e-28 |
|  | Kurtosis: 5.143 Cond. No. 7.47e+09 |
|  | ============================================================================== |

|  |
| --- |
| Warnings: |
|  | [1] Standard Errors assume that the covariance matrix of the errors is correctly specified. |
|  | [2] The condition number is large, 7.47e+09. This might indicate that there are |
|  | strong multicollinearity or other numerical problems. |
|  | Mean squared error: 0.70 |
|  | Variance score: 0.26 |
|  | Mean squared error: 0.68 |
|  | Variance score: 0.28 |
|  | Mean squared error: 0.71 |
|  | Variance score: 0.24 |
|  | Mean squared error: 0.71 |
|  | Variance score: 0.25 |
|  | Mean squared error: 0.71 |
|  | Variance score: 0.25 |
|  | Mean squared error: 0.95 |
|  | Variance score: -0.00 |
|  | Mean squared error: 0.67 |
|  | Variance score: 0.29 |
|  | SVR |
|  | Mean squared error: 0.94 |
|  | Variance score: 0.00 |
|  | Mean squared error: 1.25 |
|  | Variance score: -0.32 |
|  | Mean squared error: 0.69 |
|  | Variance score: 0.27 |
|  | Mean squared error: 0.95 |
|  | Variance score: -0.00 |

PREDICTIONS

R-squared value for predictions varied significantly depending on the regression model and training/test set ratio. Naturally, the highest levels close to the above R2 of the model itself were achieved for small test sets (15% and less), up to 0.38-0.39. Bayesian ridge regression was always the best performing model. Of the data pre-processing methods, Dimensionality reduction (PCA) was the most successful. Coupled with simple linear regression, it delivered on average much better results than the full-dimensional modelling. This can be attributed to high number of features, many of the marginal or non-significant, that add some noise to the model.

CONCLUSIONS

The carried-out analysis showed that textual content of a movie (script analysis) cannot completely account for success among the audience as we hoped for. Nevertheless, some correlations were observed, despite its moderate values.

Besides features like actors' and director' talents, production budget, etc., possible reasons such as dialogs and scenery descriptions were not separated during script analysis; relatively low response variance (scripts are available only for the most significant movies which have ratings 5-9 out of 10). Nevertheless, movies have a unique advantage of popularity criteria, the ratings from imdb.com, being easily extractable. This sharply contrasts with other purely textual artworks, such as novels, where much more correlation between success and the text analysis could be expected, but the popularity criterium is much harder to mine.