brtracy / capstone_project Public

0 stars 0 forks

Star Unwatch

Code Issues Pull requests Actions Projects Wiki Security Insig

main

brtracy Merge branch 'main' of https://github.com/brtracy/capstone_... ... 15 seconds ago 13

View code

README.md

Detecting Movie Spoilers with NLP

The Background

Nobody likes a spoiled movie (or TV show or book for that matter). Studies show that about 2/3 of U.S. adults utilize reviews to influence their decision to see a movie. When those reviews contain spoilers, they affect the anticipation and suspension of disbelief that are integral to a pleasureable movie-going experience. Users tend to avoid resources that do not censor spoiler content; they can knowingly seek out such content from other sources if they so wished.

Business Problem

IMDb, the premier site for movie information, is no stranger to dealing with the effects of hosting spoiler content. Their current solution mainly relies on self-flagging of spoiler content by the user themselves, or for other users to report a review that contains spoilers. By employing automated flagging sytems to hide user reviews containing spoilers, IMDb can provide the users with a confidence that their browsing will not present them with spoiler content. We are tasked with developing a model to detect reviews containing spoilers so they can be censored.

The Data

The dataset can be downloaded from Kaggle, which is also available here. It contains 573,913 review records that correspond to 1572 movies. There is additional film and review metadata, like film release date, critic and user rating, review post date, movie genre, etc. Primarily we are interested in the user reviews text and the target, a feature called 'is_spoiler' which indicates if the review contains a spoiler or not. There is an imbalance to our target, about 73.6% of the reviews do not contain a spoiler.

The Process

After removing non-English reviews, the review text was cleaned and lemmatized. A simple baseline model was developed, predicting that if the word 'spoiler' appeared in the review then the review was classified as a spoiler. The baseline scores allowed us a comparative benchmark as we iterated through different vectorization encoding types/settings and modeled with Multinomial Naive Bayes, Logistic Regression, and Support-vector Machines. Evaluation of model performance was done using classification reports and confusion matrices resulting in selection of a best performing model.

Conclusions

We found that bag of words modeling struggled in picking up on what makes a spoiler review a spoiler. Our best model utilized TF-IDF text encoding, adding in bi- and tri-grams with a minimum count of 8, trained using a Support-vector machine. We achieved an overall accuracy of 77% while maintaing recall of above 90% on non-spoiler reviews but only a 40% on spoiler reviews. Seeing improvement over iterations of the bag-of-words modeling we feel that turning to cutting edge tools like BERT modeling would see even greater success. Another interesting avenue is cosine similarity, comparing the review to the synopsis of the film being reviewed, which has a more narrow scope but could prove to be successful.

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

- **Jupyter Notebook** 100.0%