STAR TREK
THE NEXT GENERATION

PREDICTING IMDB RATINGS WITH NLP AND MACHINE LEARNING

CAPSTONE PROJECT BY KATYA KOGAN

THE PROBLEM

Many tv shows face countless risks with their scriptwriting and casting choices. Sometimes, unknowingly, reducing or increasing a certain character's spoken parts or appearances can have a negative impact on the ratings for the show.

What if there was a way to determine from past productions to be confident in writing these scripts to guarantee a positive outcome?

BACKGROUND

This project explores the possibility of determining, through NLP - if certain characters (and recurring characters) play a role in affecting the IMDB rating. Having the capability to check against the past seasons and episodes can be incredibly useful in reducing costs of producing, casting and as well as scriptwriting.

In the case of Star Trek: TNG, it cost an average of $1.3 million to produce a single episode. Theoretically, scriptwriters could test the new script with the model, and learn about the impact that each character plays in contributing toward an optimal rating.

DATA

Star Trek: The Next Generation is a Science Fiction TV show that aired from the late '80s into the mid-'90s. At the time the show was live, it had an average of 20 million viewers, and still holds widespread popularity to this day. I decided to use this dataset acquired from Github, because it breaks down all of the scripts, line by line, per character.

In most NLP cases, there is a generalized view of using the text data to find a general outcome of "good" or "bad". However, there are too many factors at play to just blindly accept it as such, as well as considering the fact that generalizing ratings from 5-9 is too narrow of a window to split into two categories (even if I did, the dataset is too small).

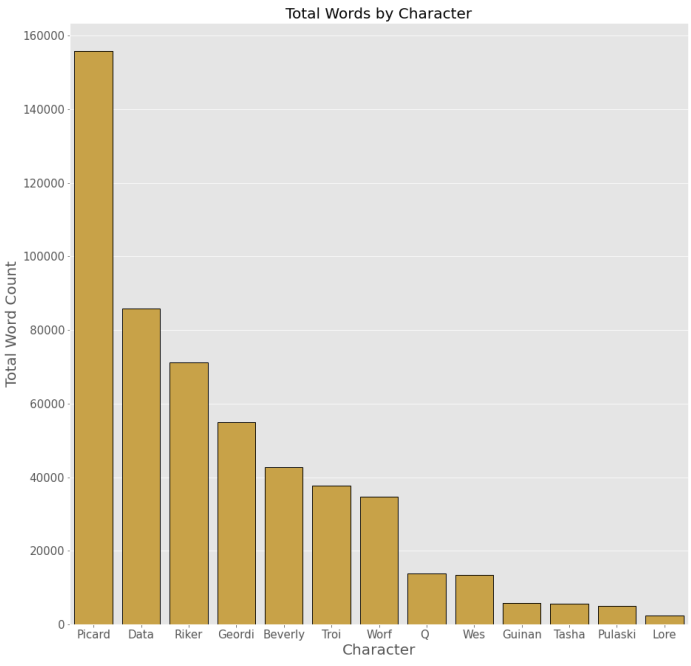
EDA & PREPARATION SUMMARY

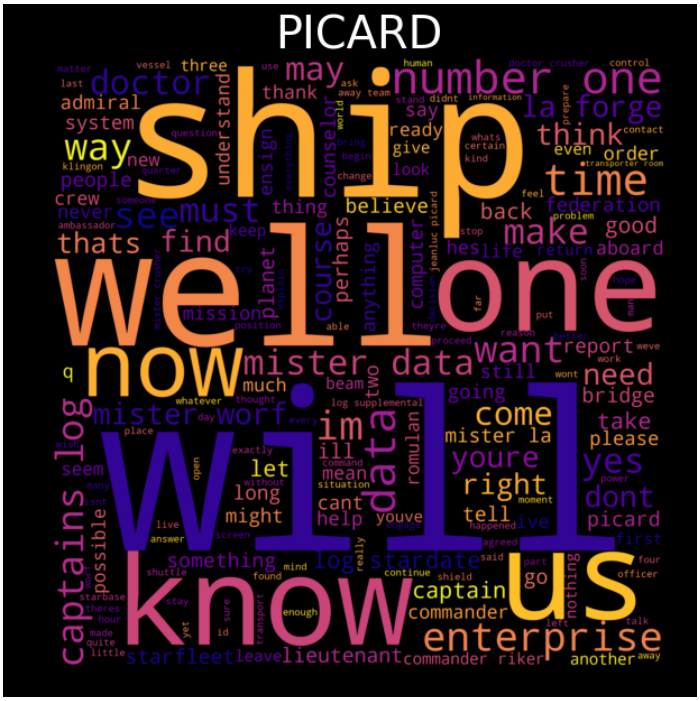
After filtering the dataset into spoken lines, I did the process of cleaning, imputing missing values, and removing duplicate rows. After the standard generalized cleaning procedure was finished, I focused on cleaning up the text and character data.

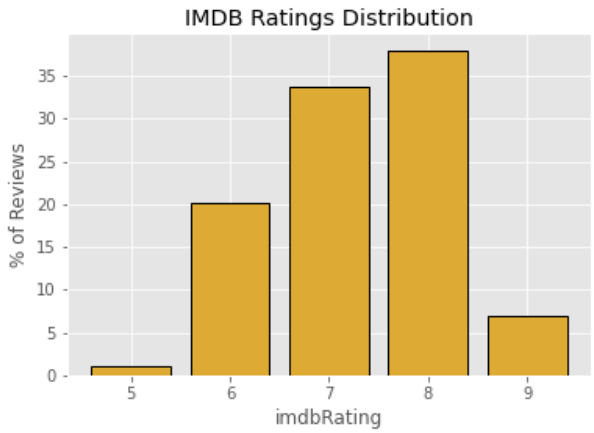
Firstly, the character column had more than just the main cast of characters, it contained recurring characters as well as guest roles. My first order of action was to clean this column so that it no longer contained strange brackets, quotations, or a combination of names, and a similar process was applied to the script text as well.

After the cleaning was finished (for both the characters and script data), I started to perform more heavy-duty EDA. I started off simply using WordCloud to have a look at the most recurring words per character and then expanded from there - exploring the most words across the show, ratings across the years, and the total lines vs IMDB ratings.

Through the process of EDA, I was able to acquire the total word counts for each line in the data frame. A loop was created in order to take the total words from the episode, and compute the total word counts for each character in the main cast (grouped on the episode level), which was then turned into relative percentages for each character's spoken parts.







MODELLING SUMMARY

With my engineered dataset, I approached this as a regression problem. I started with looking at a correlation heatmap of the character's spoken parts to have an idea of which characters correlate with each other and the rating score. From there, I performed a train test split of the data, and scaled it.

I tested a variety of models, such as Linear Regression, Decision Tree Regressor, and Random Forest Regressor - to name a few examples. Using a function, I was able to test the models at baseline to see which ones performed the best. With the Root Mean Squared Error as my guiding metric, I was able to narrow down a potential model, which was Random Forest.

After performing GridSearch, the baseline model performed the best with an RMSE of 0.84. Then, I explored the feature importances of the model and found interesting insights. Some characters that are perceived as beloved are expected to talk more or less - which more insights are shared in the notebook regarding this.

FINDINGS AND NEXT STEPS

Using this model to test by word count is the simplest way to gain a general idea of the proportions in each character's spoken parts that would determine the best result. There are more advanced techniques to gain better accuracies, but due to time constraints, I wasn't able to explore building a Neural Network or test other models such as SVM.

To take this further, I would expand the analysis to include more NLP-focused models, and see what words have the best impact (and which people should be saying these words). While building this, I would also attempt building an RNN to see if it'll bring a better result than I have currently.