Rotten or Fresh?

Joseph T. Tustin

/github.com/joetustin
/in/joseph-tustin/
joetustin67@gmail.com

Intro

The Rotten Tomatoes Movie Review Dataset consists of over 420,000 reviews which are in the form of simple sentences. These reviews are classified as either positive or negative. The purpose of this capstone project is to to build a model to analyze the text to predict the class of the review. With random guessing, we would have a baseline predictive ability of 50%.

Problem Statement

Natural Language Processing is hard! Words (unigrams) have meaning. Combination of words (bigrams and trigrams) can hold even more meaning However, despite all this information that words and combinations of words hold, there can still be double meanings, sarcasm, and contradictory text. It is estimated that given some random text only 65% of all people will agree on its given meaning.

Positive Review
"Manakamana doesn't answer any questions, yet makes its point: Nepal, like the rest of our planet, is a picturesque but far from peaceable kingdom."

Positive Review
"Wilfully offensive and powered by a chest-thumping machismo, but it's good clean fun."

Negative Review
"It would be difficult to imagine material more wrong for Spade than Lost & Found."

Negative Review
"Despite the gusto its star brings to the role, it's hard to ride shotgun on Hector's voyage of discovery."

Text Cleaning

Text cleaning is an important first step in natural language processing. After text cleaning, the simplified words in the reviews must be turned into a numerical matrix using count vectorization as well as tf-idf factorization. The typical work flow for text cleaning is show below:

Input Text Documents
↓
RegEx: to remove num, punc, char
↓
Tokenize: Sentence to list of words
↓
Stopwords: Remove common words
↓
Lemmatize to find common word basis

Once this process has been completed, the data from the previously listed reviews is shown below.

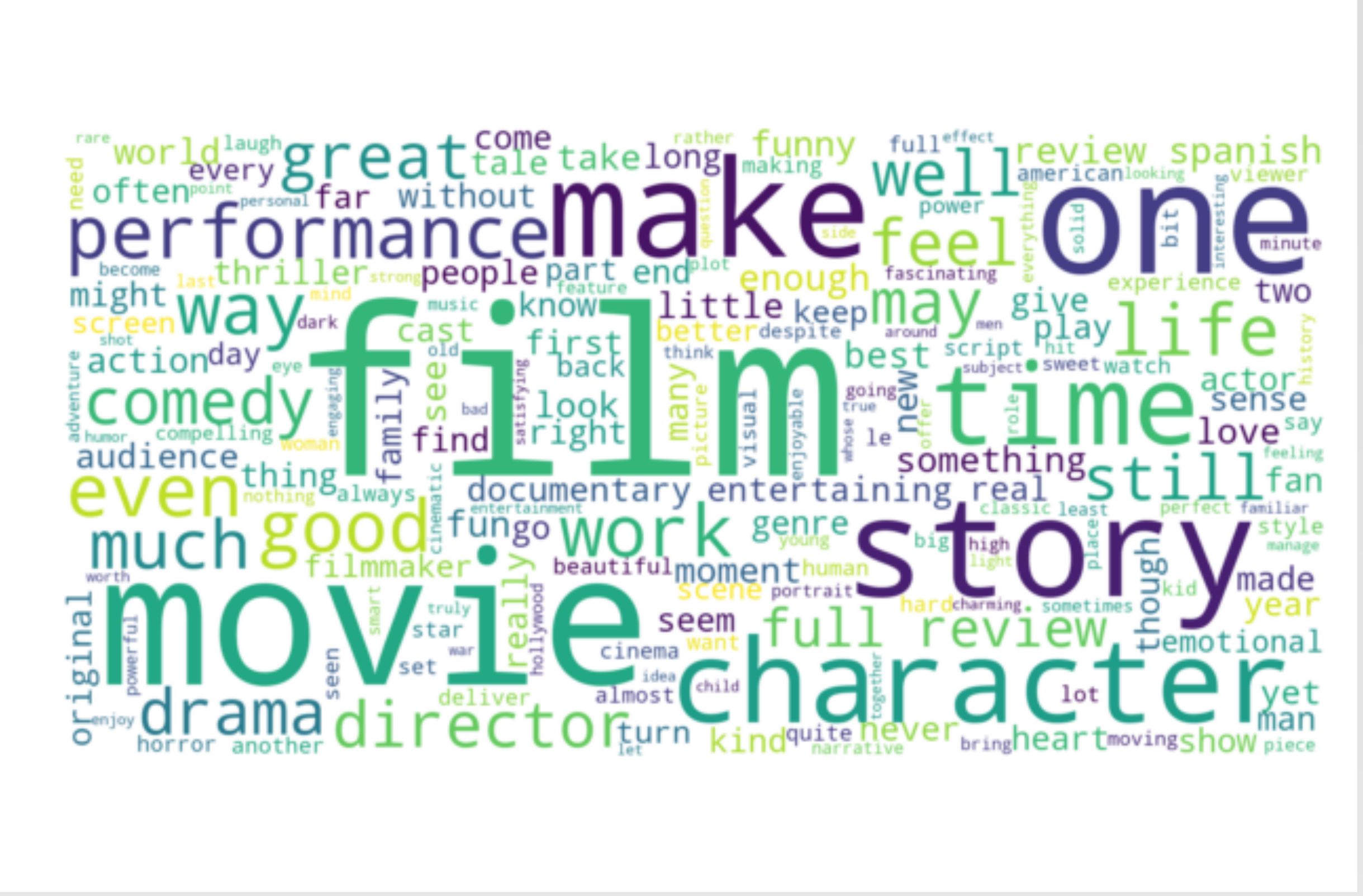
['manakamana', 'answer', 'question', 'yet', …]

['wilfully', 'offensive', 'powered', 'chest', …….]

['would', 'difficult', 'imagine', 'material', 'wrong',..]

['despite', 'gusto', 'star', 'brings', 'role', 'hard',..]

As a picture can be worth a thousand words, the word clouds shown represent the frequency of the most common words as seen in both the positive and negative reviews. Words, such as "great", "love", and "beautiful" really stand out as being associated with positive reviews. It is interesting to note that some of the words associated with the most value appear to have smaller frequency count sizes.



Word Cloud for Positive Reviews

This observation further validates the choice to use to use term frequency - inverse document frequency as a numerical vectorization method. Please note the use of "nothing", "bad", and "dull" in the word cloud of negative reviews shown below.

Word Cloud for Negative Reviews

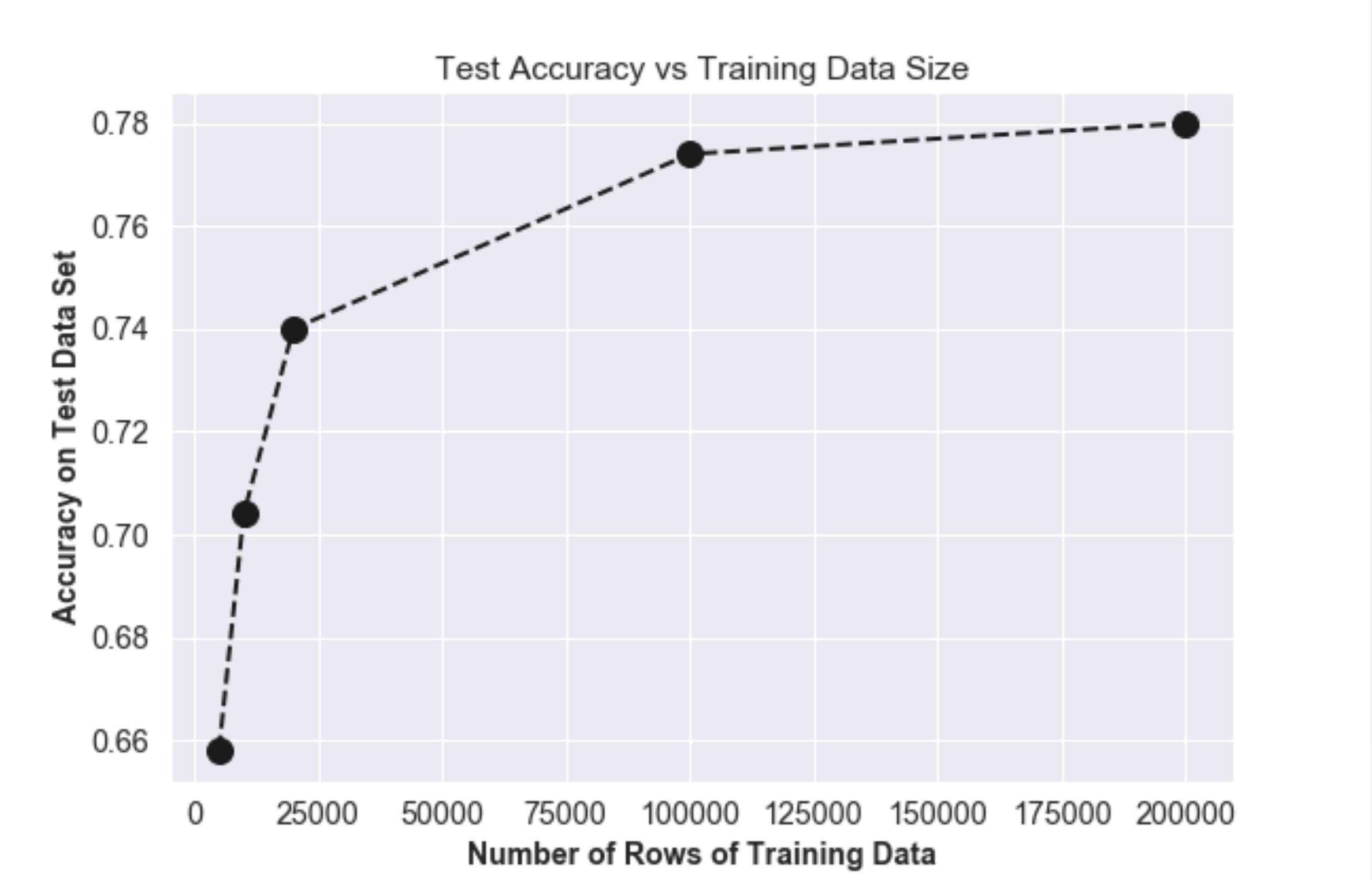
Methodology

With the normalized, numerical matrix representing the words in each of the reviews, models were created with a small subset of the data using Naive Bayes, Random Forest, and Logistic Regression to find the best predictive model while hyper tuning the model parameters using gridsearch with cross-validation.

	Training Accuracy Score	Test Accuracy Score
Naive Bayes	0.78	0.73
Loigistic Regression	0.80	0.70
Random Forest	0.65	0.62

Once the best model (Naive Bayes) was chosen, the model was scaled up to work on a larger dataset using an AWS m5 instance. With the finalized model, a flask app was created to allow users to interact with the model.

Results



Test Accuracy vs Training Data Size

The final model was based on a training data set of 100,000 reviews. The model was scored on a test set of 25,000 reviews. It has an accuracy, precision, and recall equal to 0.78. Not Bad for our exploratory purposes!!!

Thank you *for taking the time to checkout my Capstone project! Hope that you enjoyed it. Please feel free to play around with my Flask App. Special thank you to our instructors, dsr, fellow cohorts, and my wife!*