Essay Grade Prediction Using a MaxEnt Classifier

Naomi Zarrilli and Alexander Ravan
COMP150-04: Natural Language Processing
Professor Yamangil

Abstract

We took on the task of predicting essay grades. We attempted to solve this problem using MaxEnt. Our data is from Kaggle (https://www.kaggle.com/c/asap-aes/data). It is labelled with numerical scores that vary depending on the essay set. There are eight sets and 11178 essay total. For our most accurate model, we used part of speech tags and tfidf as features. Our baseline was bag of words with Naive Bayes classification. Our baseline accuracy was 45.9%. We were able to increase our accuracy by nearly 20% using part of speech tags, bigrams, and tfidf as features for MaxEnt. Our resulting accuracy was 62.1%.

Introduction

We chose to use NLP to predict essay grades. As an English and Computer Science major, I (Naomi) am constantly interested in finding a meaningful intersection between these two fields. This is particularly because English is is the study of art, which is often not calculated but fluid, while Computer Science revolves around algorithmic concreteness. I think writing itself is an art and the evaluation of it is often entirely subjective. Trying to reframe this evaluation as algorithmic unites two seemingly disparate processes and allows us insights into why an essay succeeds. The factors that go into writing a good essay, to me, are not formulaic. There is no algorithm to writing a A+ paper because each topic requires something different and the evaluation of these paper comes from humans, who are inherently biased. This task seemed challenging and interesting because we attempted are attempting to simulate a human's process of evaluating and reasoning about a paper. This problem can be tackled, not necessarily solved, using NLP because MaxEnt will take information about the paper and determine how this information relates to the paper's grade. Rather than reading the paper itself, the information is abstracted as feature vectors.

Dataset

We used a kaggle dataset of tagged essays. These are essay responses from 6th through 10th graders and they are assigned numerical scores. They were graded by multiple graders and then a composite final score was given. We used these composite scores to calculate a letter grade. There are eight essay sets and 11178 essays total. Essay set 1 is graded on a scale of 1-6.. Essay set 2 was ignored because the scores were extremely difficult to translate into letter grades. Essay sets 3 and 4 were scored from 0 through 3. Essay sets 5 and 6 were from 0-4. Essay set 7 was out of 30 and essay set 8 was out of 60. We predicted letter grades A-F. We scaled all of the essays to letter grades based on which set they belonged to. This can be seen in our letter_grade() function in the get_data file. The data was initially sorted by set and set 8 had the longest, most complex essays because they were from 10th graders.

Each line of the training data contains the essay id number, the essay set number, the essay, and scoring information. We first randomized the data and then picked the last 10% as the test set and the other 90% as the training set. This was our training and test set. We then split up this information to extract the essay, the

grade, and the essay set number. In our data preprocessing, we added unknown tokens, tokenized the data, and created a parallel file of parts of speech tags for the essays. This data also tagged named entities, such as an organization was replaced with the string @ORGANIZATION1. We eliminated these and replaced them with our own tags, similar to the unknown tag. Using the process described in the previous paragraph, we used the essay set to determine the essay's letter grade and then created a parallel list of letter grades to correspond to the essays.

Here is an examples of our data are before pre-processing.

1 1 "Dear local newspaper, I think effects computers have on people are great learning skills/affects because they give us time to chat with friends/new people, helps us learn about the globe(astronomy) and keeps us out of troble! Thing about! Dont you think so? How would you feel if your teenager is always on the phone with friends! Do you ever time to chat with your friends or buisness partner about things. Well now - there's a new way to chat the computer, theirs plenty of sites on the internet to do so: @ORGANIZATION1, @ORGANIZATION2, @CAPS1, facebook, myspace ect. Just think now while your setting up meeting with your boss on the computer, your teenager is having fun on the phone not rushing to get off cause you want to use it. How did you learn about other countrys/states outside of yours? Well I have by computer/internet, it's a new way to learn about what going on in our time! You might think your child spends a lot of time on the computer, but ask them so question about the economy, sea floor spreading or even about the @DATE1's you'll be surprise at how much he/she knows. Believe it or not the computer is much interesting then in class all day reading out of books. If your child is home on your computer or at a local library, it's better than being out with friends being fresh, or being perpressured to doing something they know isnt right. You might not know where your child is, @CAPS2 forbidde in a hospital bed because of a drive-by. Rather than your child on the computer learning, chatting or just playing games, safe and sound in your home or community place. Now I hope you have reached a point to understand and agree with me, because computers can have great effects on you or child because it gives us time to chat with friends/new people, helps us learn about the globe and believe or not keeps us out of troble. Thank you for listening."
4 4 8

What is interesting about this example is the number of misspelled words. We imagine the occurrence of these misspelled words in other essays is low. By using bag of words of bigrams, the low occurrence of these misspelled words may correlate to a lower essay score. You can also see how named entities have been replaced with "@tags".

Experimental method

As this is a classification problem, we decided to use MaxEnt to classify essays as letter grades. Our main method is using bag of bigrams, POS tags and tfidf as features for logistic regression. We implemented

cosine similarities but omitted it because it did not increase the model's accuracy. We included named entity recognition as a feature for this model; however, excluding it causes no performance changes. We believe this is because bigram counts produces hundred of features and comparatively, named entities produces a negligible amount (only 9 features), therefore this vector does not impact the model's performance. We also used cosine similarities but it ultimately worsened our model's performance and slowed it down so we did not include it. Logistic regression determines how to weigh these features in relation to the Y axis (grades) that correspond to the features.

We implemented the creation of some feature vectors and used scikit-learn. We implemented bag of words, bag of bigrams, bag of named entities, and bag of tags. By this, we mean that we converted bigrams, named entities, and part of speech tags into feature vectors ourselves. We used scikit-learn's tfidf and cosine similarity functions to generate feature vectors. We also use scikit-learn's logistic regression to fit our model. The features can be seen in the features_matrix() function, which calls on both functions we wrote and scikit-learn functions to create features. These vectors are concatenated together and eventually given to the logistic regression model
We used NLTK to preprocess input, which includes tokenization and replacing unknown words with tokens. The Stanford Part of Speech tagger was used to assign POS tags to all the data, which is used as a feature in the MaxEnt classifier. Additionally, from scikit-learn, we used the Naive Bayes MultinomialNB function as part of the baseline algorithm.

Logistic regression doesn't have a fixed time complexity. However, for our baseline of using just bag of words for n essays with k words each, our feature vector would be size n by k. Logistical regression would have to work through this matrix and in the worst case, n=k. I don't believe this model will scale well to large amounts of data because we are using bigram counts as a feature. For large datasets, the number of bigrams will grow massively, which will severely slow down the model's performance and run the risk of running out of memory. We ran into memory issues that prevented us from concatenating the bag of words vector with the bag of bigrams vector.

Evaluation

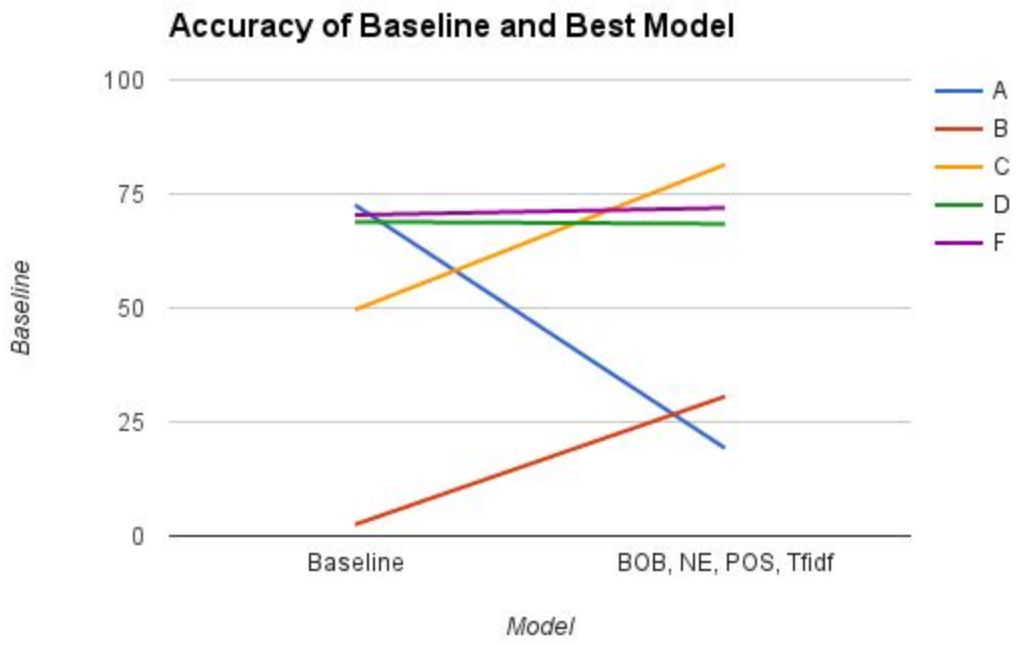
We evaluate our model by dividing the number of correctly predicted essays by the total number of essays to get an overall prediction accuracy. We do this same process for individual letter grades to get the model's accuracy predicting A, B, C, D, and F correctly.
Our baseline is using Naive Bayes classification with bag of words. Our baseline performance was 45.9% accuracy. These are the model's accuracies for predicting specific grades:

Model	A	B	C	D	F
Baseline	72.6	2.5	49.6	68.9	70.4

Our most accurate model resulted used POS tags, bigrams counts, and tf-idf. This model produced 62.1% prediction accuracy. These are the model's accuracies for predicting specific grades:

Model	A	B	C	D	F
BOB, NE, POS, Tfidf	19.3	30.6	81.4	68.4	71.9

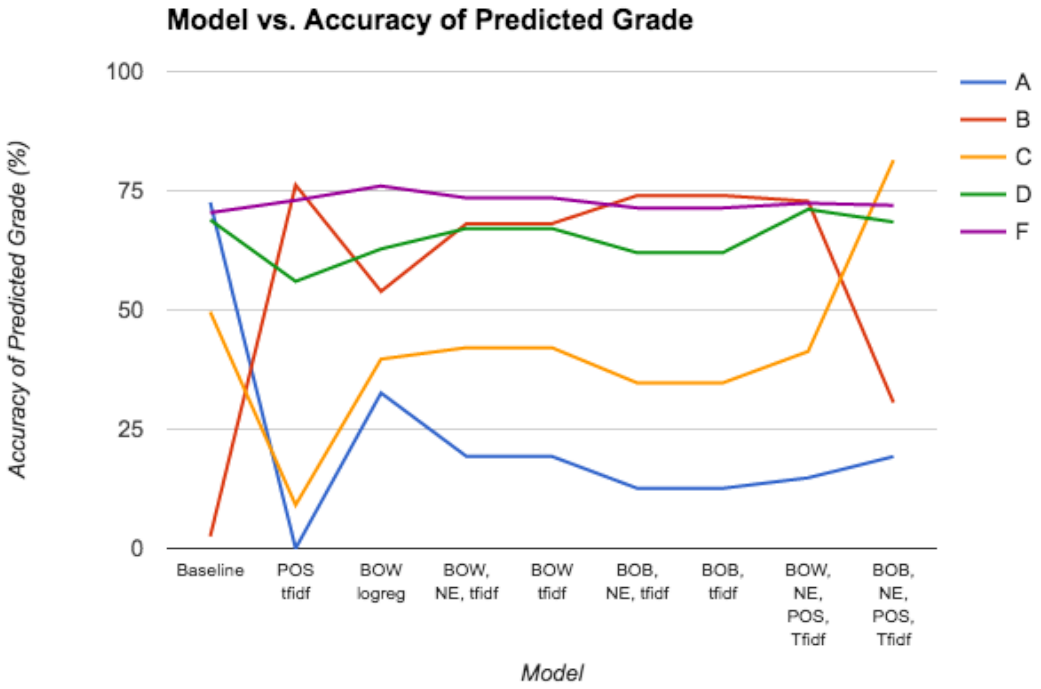
This graph provides a visual of our baseline performance versus our best model's performance:



We tried a variety of different combinations of features to produce the most accurate model. Our greatest improvement came from adding POS tags. A small performance difference came from using bigram counts over bag of words. Here are the differences in overall prediction accuracies based on the features we gave our model:

Model	Total Accuracy
Baseline	45.9

POS tfidf	52.6
BOW logreg	55.4
BOW, NE, tfidf	59.2
BOW tfidf	59.2
BOB, NE, tfidf	59.6
BOB, tfidf	59.6
BOW, NE, POS, Tfidf	60.7
BOB, NE, POS, Tfidf	62.1

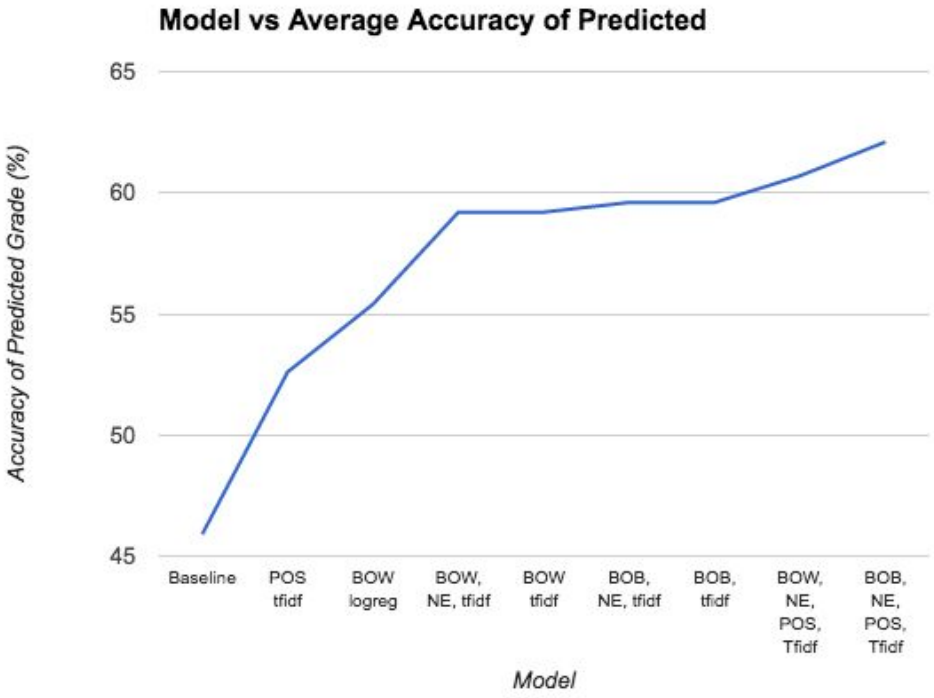


As seen in the chart above, the Baseline model had a very low (2.5%) accuracy when tagging essays of grade B. However, in our best model, the accuracy of predicting Bs increased to 30%. For example the following essay:

> *The author concludes the story with this paragraph because saeng was taking a test. In the story there are some quotes to support my response i-i failed the test saeng said. This shows that she was taking a test about gardening. Another reason why the author concludes the story with this paragraph is because the story was about spring, gardening, and the life of it to them. The*

paragraph talks about spring, and flowers. the daughter and mother live off of the plants by planting <UNK>, and fruits.

was classified as a D using the baseline model. When processed with the model using Bag of Bigrams, Named Entities, Parts of Speech tags, and Tf-idf, it was correctly classified as a B.



Conclusions

In the future, we would want to use a convolutional neural network instead of logistic regression. We now very little about CNNs and they are beyond the scope of this class; however, we think these may provide us with higher accuracies than MaxEnt. We would also want more complex essays, like college-level essays, because many of the essays in this dataset are short and overly simplistic. Essay set 8 contained the most advanced essays and when we used all of essay set 8 as test set, our model's accuracy dropped to ~45%. We would definitely want to further explore why our model struggled to predict grades for more "advanced" essays and part of dealing with this problem would be working with longer, more complex essays. We also would want to suggest to a user ways to improve their essays and identify which areas of their essays could be improved. It would also be interesting to learn information about the graders based on the way they evaluated an essay. We would probably need information on the graders but this idea is extremely interesting to consider.

One of the many things that makes this task extremely difficult for a computer is that essays are graded using the judgement of a human being, and human beings are inherently biased. A part of writing papers is catering to an audience and knowing what biases this audience may hold. For example, a student may write a paper for Professor X on a topic and intentionally quote Author Y because the student knows that Professor X particularly admires Author Y. Because of this, the paper could receive a higher grade. Another example is if a student is writing about Topic X and knows the professor will have little background on the topic, the student may choose to provide more background information on the topic; however, if this paper were submitted to a professor who knows a great deal about the topic, the paper may appear reductive or generalizing about the topic. The almost arbitrary nature in which papers are graded seems difficult to capture in an algorithm. It is difficult to capture the human aspect of grading papers because how do you programmatically represent the somewhat random human factors that go into essay grading? I personally think that intentionally catering a paper towards an audience is a meaningful skill, however, if a computer were grading essays without bias (if that is even possible), how would this change the way we write? Is it a positive to eliminate these human factors by assigning a computer to the task of essay grading?

These are questions that this project raised in our minds and like we mentioned before, an extremely interesting direction to take with this project is learning about the graders. Particularly, learning about how individual biases affect how a paper is graded. Ultimately, a greater-than-binary classification task is extremely difficult. We were really hoping to get our model to correctly grade over 50% of essays and we were thrilled with our model's 62.1% accuracy. We are overall happy with the accuracy we achieved especially in the context of the scope of this project and time limit.