Automated Essay Scoring

Tyler Schmalz

Introduction

Many states' department of education are developing new forms of testing and grading methods in order to assess the implementation of common core standards. These standards emphasize the need for students to be able to read and write in all subject matters. However, state testing has predominantly relied on multiple choice testing for the speed and efficiency of scoring. This conflicts with the overall intent of common core and writing. While multiple choice questions undoubtedly still have a role in assessing student learning, written responses provide a different and equally important look at a student's academic achievement. Very few educators would argue otherwise, but they are expensive and time consuming to grade them by hand, leading to many standardized tests and individual teachers relying on multiple choice testing for efficiency.

This issue led me to my analysis into automated essay scoring. It is my purpose through this study to create a model for automatically scoring an essay using Natural Language Processing techniques.

Motivation

As mentioned in the introduction, there is a heightened interest in US public education around student's abilities to write in all subject matters. While virtually all educators would agree this is an important endeavour, scoring written assignments takes substantially longer and can depend heavily on the grader themselves. On a larger level, state departments of education would be interested in the model's capability to increase efficiency and reduce cost of grading short and long essay answers, especially on the new standardized testing designed to assess student achievement with common core standards. I also envision individual districts, schools, and teachers interested in a tool like this. It could help with standardization of grading within a department as well as hopefully reduce the load on individual teachers, freeing up time for more professional development and collaboration within and across departments.

Data

Source

The essay data for my analysis comes from a Kaggle competition in 2012, which can be easily accessed at the link: Automated Essay Scoring. The dataset includes eight essay sets, each one including student responses to a different prompt. The individual essays range from an average length of 150 to 550 words per response. All responses were written by students ranging in grade levels from Grade 7 to Grade 10, and were hand graded and double-scored. The following information is recorded for each observation:

- **essay_id**: A unique identifier for each individual student essay
- **essay_set**: 1-8, an id for each set of essays
- **essay**: The ascii text of a student's response
- **rater1_domain1**: Rater 1's domain 1 score; all essays have this
- **rater2_domain1**: Rater 2's domain 1 score; all essays have this
- **rater3_domain1**: Rater 3's domain 1 score; only some essays in set 8 have this.
- **domain1_score**: Resolved score between the raters; all essays have this
- **rater1_domain2**: Rater 1's domain 2 score; only essays in set 2 have this
- **rater2_domain2**: Rater 2's domain 2 score; only essays in set 2 have this
- **domain2_score**: Resolved score between the raters; only essays in set 2 have this
- **rater1_trait1 score - rater3_trait6 score**: trait scores for sets 7-8

Anonymization

In an effort to remove personally identifying information from the essays used the Named Entity Recognizer (NER) from the Stanford Natural Language Processing group and a variety of other approaches. Identifying information such as names, locations, dates, times, email, and phone numbers are replaced with generic name preceded by the “@” symbol. For example, the statement “I attend Springfield School…” is converted to “I attend @ORGANIZATION1…”. This anonymization could have an impact on the training of the overall model, and the method of dealing with it will be discussed later.

Dropped Essays

The following essays were dropped from the dataset:

- Essay 10534: Missing all scores
- Essays 6123, 6239, 6669, 7177: All had a rater2 score of zero when the rater1/domain1 scores were both relatively high. Examination of the essay text revealed a score of zero is highly unlikely, as they were on topic and thoughtfully written.

Text Pre-processing

I took the following steps to preprocess the essay text in preparation for creating features:

1. Strip text of accented characters and replace with typical English letters (use ***n*** in place of ***ń***.
2. Remove anonymized identifying information (all words such as @LOCATION or @PERSON).
3. Expand all contractions to ensure they aren't interpreted as different words (change don't into do not).
4. Remove all special characters (except for periods).
5. Lemmatize the text (replace all tenses of verbs with the root word: change swimming into swim)
6. Remove stopwords (common words that don't add to the understanding of the text such as the, is, are, etc.

The end result is that it converts an essay text such as:

```
“I think That The author concludes the story with This paragraph to show that she does
not want to give up on fighting her homesickness. The first time she bought that
plant, it brought back memories of her old home and it showed how much she missed it
there. By saying shell do the test again shows that she doesnt want to let ths defeat
her in her new home and her new life.”
```

To a simplified version:

```
“think author conclude story paragraph show want give fight homesickness first time buy
plant bring back memory old home show much miss say shell test show want let th defeat
new home new life”
```

This particular example highlights some of the shortcomings of my current text preprocessing techniques. For example, it struggles to deal with misspelled words such as not expanding “shell” to “she will” because of the missing apostrophe, or failing to drop “th” because it isn’t spelled correctly as “the”. I was able to deal with this particular issue later in the feature creation, specifically the tf-idf vectors.

Features

Rating Discrepancy

The dataset provides the individual assessments for at least two different raters for each essay (‘rater1_domain1’, ‘rater2_domain1’, ‘rater3_domain1’) as well as the final, resolved score (‘domain1_score’). Using all of these ratings, the “rating discrepancy” column assigns a value between 0 and 1 to describe the level of disagreement in scoring that particular essay. A score of zero means the raters and final score agreed perfectly while a one means they disagreed as much as possible. This is calculated by finding the largest difference between the raters and final score, and dividing by the overall maximum score for that essay_set.

For example, here are the scores for essay_id = 10433:

rater1_domain1	rater2_domain1	rater3_domain1	domain1_score	rating_ discrepancy
3	2	NaN	3	0.25

In this situation, the rating_discrepancy of 0.25 shows that the graders mostly agree with each other, but there is a slight disagreement somewhere, as can be seen by rater2 scoring slightly lower than the others. The motivation for calculating this particular feature is that an essay with a higher discrepancy will most likely be more difficult for the model to predict as well.

tf-idf Vectors

The other main feature calculated were term frequency-inverse document frequency (AKA tf-idf) vectors, which are used to reflect how important a word is to a document in a collection of documents. The tf-idf value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word. This helps to adjust for the fact that some words appear more frequently in general, and are not necessarily important.

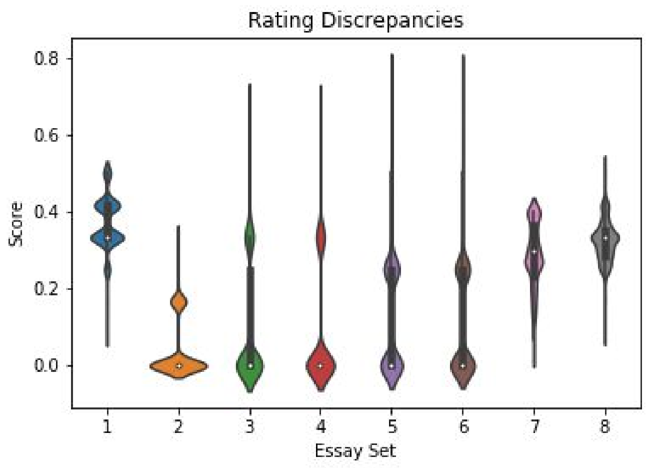
I decided to exclude the 5% of most common and least common words in the corpus. By eliminating the least common words, I was able to eliminate many of the random spelling errors because each new misspelling was registered as a distinct word in the vocabulary. Eliminating the most common words helps to avoid any common, but irrelevant words that were missed by the stopword removal in the text preprocessing steps.

Essay Set

Lastly, I included a feature for each essay set (essay_set1, essay_set2, etc.) which has the value of 0 or 1 if the particular essay is a part of that set or not. This was included to help offset the overall difficulty of the particular essay set (i.e. if an essay set was scored more harshly, this could be identified by the essay set feature).

Exploratory Data Analysis

Because this analysis deals primarily with text and tf-idf vectors which don't necessarily translate well into exploratory data analysis, there isn't a lot to explore before creating our model. With that being said, there are some interesting insights that can be drawn from the rating discrepancies, especially when broken down by essay set.



In the figure to the right you can see the distribution of the rating discrepancies separated by the essay set that they came from. At first glance, it appears there are two groups with differing characteristics. Essay sets 1, 7, and 8 have a median value around 0.35 suggesting that it is fairly common for raters to disagree with each other by about 35% of the maximum score. Essay sets 2, 3, 4, 5, and 6 have a median value of 0, suggesting that it is mostly common that the raters agree with each other's scores completely. However, these essay sets have a much higher spread in that they also have very high discrepancies, with some close to 1, suggesting the raters strongly disagreed with each other. During more in-depth analysis, this grouping of rating discrepancies could suggest that a model may be more successful at predicting one group of essay sets versus the other.

In-Depth Analysis

Scaling

In order to prepare the data for applying machine learning models, it’s crucial to apply some form of a scaler to minimize the impact of the different ranges and magnitudes of features and target variables. I used sklearn’s MinMaxScaler to scale all of the features to between zero and one. However, the target feature domain1_score had an interesting situation where the scores had a different range depending on which essay set they came from.

My first option was to apply the same MinMaxScaler to the all essay’s scores, regardless of their original range. This type of scaler is simpler to implement and save for reverse transformations, as well as be more likely to lead to a model that generalizes well to other essays, regardless of their potential range of values. However, with some essays being scored from 1-4, while others were scored from 1-60, I had concerns about the impact this may have on the predictions themselves.

My second option was to apply a separate MinMaxScaler to each essay set separately. By doing so, a score of 3/4 from on essay set should be scaled roughly equivalent to a score of 45/60 on a different essay set. This way, similar features in each of the essay sets should lead to a prediction of a similar quality of score on an essay, regardless of what the overall range was. However, this method was more difficult to implement and more calculation heavy, as it required fitting a separate instance of MinMaxScaler to each essay set, as well as saving all of them for later reverse transformations. In the end, I decided to train models using both scalers, and determining which was most effective at the end.

Model Selection and Training

With the ultimate goal of predicting essay scores based on the features (tf-idf vectors, rating discrepancy, and to which essay set the document belongs), I tried various models at first using the essay dependent scaled data in order to identify the possible best performer, such as Ridge linear regression, random forest regression, and support vector machine regression. However, while these models were much quicker to train, they all struggled to accurately predict the final essay score, as evidence by them all having a coefficient of determination 0.4 or lower. The most successful model seemed to be the Gradient Boosting Regressor from scikit-learn’s library which had a coefficient of determination of about 0.55.

The next step was to use the grid search cross validation function to tune the following hyperparameters of the Gradient Boosted model:

- N_estimators: The number of boosting stages to perform
- Max_depth: Maximum depth of the individual regression estimators
- Min_samples_split: The minimum number of samples required to split an internal node
- Learning_rate: Shrinks the contribution of each tree by this rate

The results were the following parameters for both the single scaler and essay-dependent scaler models.

	Single Scaler	**Essay Dependent Scaler**
n_estimators	500	1000
max_depth	2	2
min_samples_split	2	2
learning_rate	0.1	0.1

Evaluating Models

After cross-validation to determine the best hyperparameters listed above, there were interesting results in the predictions and evaluatory statistics from both the single scaler model and the essay-dependent scaler model. At first glance, the single-scaler model seemed to do substantially better at predicting essay scores with a coefficient of determination 0.95 versus the essay dependent scaler with a coefficient of determination 0.60. However, while the coefficient of determination can provide useful insight into the effectiveness of a model at predicting, it can be easily swayed by other factors, requiring a more in-depth exploration of the actual predictions from both models.

Comparing Predictions
In order to compare the effectiveness of each model at predicting the final score for essays, I looked into how closely each model was predicting values to the actual essay scores, as can be seen in the table below (S = single scaler, E = essay dependent scaler). The top row indicates which essay cohort is being considered.

Within		1	2	3	4	5	6	7	8	Total
5%	S	44.0	27.6	19.3	20.0	20.9	28.2	33.8	44.1	28.6
	E	43.5	37.7	16.9	18.1	25.2	28.9	32.2	47.1	29.9
10%	S	74.2	51.0	36.0	37.0	43.3	49.4	62.9	80.2	51.9
	E	73.7	67.7	34.6	40.3	44.5	49.6	61.9	73.8	54.2
20%	S	97.1	84.6	63.1	62.0	75.2	76.9	91.6	99.5	79.5
	E	91.1	95.1	65.4	68.9	80.2	93.1	92.6	96.5	83.7

At first glance I noted that that the essay dependent scaler model is performing slightly better at predicting closer to the actual original scores, especially as we increase the tolerance from within 5% to 20%. This ability could be useful if the model's ultimate goal is to predict a score within a range of values rather than the numeric score.

Another interesting thing I noted is the difficulty that both models are having at predicting scores for certain essay sets, most notably essay sets 3 and 4. This difference is also particularly noticeable at the higher tolerance level (within 20%) where the other essays are predicting about 80-90% of scores within that tolerance level, but still struggling to closely predict essay set 3, 4, 5, and 6.

What could it be about these essay sets that make it difficult for the model to accurately predict a score? Upon closer inspection of the individual essay topics, the prompts from essay sets 3, 4, 5, and 6 are classified as "source dependent responses" in the original description of the essay prompt. These essay prompts involve the writer reading a story, and analyzing literature elements from the story (i.e. setting, mood, climax, etc). However, the model appears to do substantially better at predicting the score of essays from set 1, 2, 7, and 8, which are classified as persuasive, narrative, or expository writing.

Conclusion

Firstly, the model trained using the essay dependent scaled data is able to more closely predict the final score, particularly when trying to predict within 20% of the original score. While this may not be traditionally useful in most machine learning applications, I believe in the case of standardized testing it could play a large role. Typically, the results of standardized testing are grouped into bands or categories such as: way below standard, below standard, met standard, exceeded standard. In this case, it is less important to exactly predict an essay's score, but rather to get within a certain range of it. Therefore, this model could be an effective tool to achieve this much more quickly and cheaply.

Secondly, the essay dependent scaled model was able to predict the scores of persuasive/expository type essays substantially more accurately as compared to source dependent type essays. For instance, predicting approximately 40% of expository type versus predicting approximately 15% of source dependent type essay within 5% of the actual score.

Future Work/Considerations

For future work on this project, I would strongly consider separating the essays into these different types, and training a model separately for each one. Because of the substantial difference in the accuracy of predictions between the two types mentioned earlier, I believe that this could have substantial results in improving accuracy for both types.

I would also like to look into creating additional features to build a model on. While the tf-idf vectors were effective at predicting some essay types, I have seen other approaches for creating features based on counting other aspects of the essay such as the number of sentences, words, lemmas, etc. or the length of sentences and words. I think that these values could improve the accuracy of my model's predictions, especially for the source dependent essay types.