Automated Essay Grading Using Machine Learning

Manvi Mahana, Mishel Johns, Ashwin Apte
CS229 Machine Learning - Autumn 2012
Stanford University
Final Report Submitted on: 14-Dec-2012

Abstract

The project aims to build an automated essay scoring system using a data set of ≈13000 essays from kaggle.com. These essays were divided into 8 different sets based on context. We extracted features such as total word count per essay, sentence count, number of long words, part of speech counts etc from the training set essays. We used a linear regression model to learn from these features and generate parameters for testing and validation. We used 5-fold cross validation to train and test our model rigorously. Further, we used a forward feature selection algorithm to arrive at a combination of features that gives the best score prediction. Quadratic Weighted Kappa, which measures agreement between predicted scores and human scores, was used as an error metric. Our final model was able to achieve a kappa score of 0.73 across all 8 essay sets. We also got a good insight into what kind of features could improve our model, for example N-Grams and content testing features.

1 Introduction

Essays are crucial testing tools for assessing academic achievement, integration of ideas and ability to recall, but are expensive and time consuming to grade manually. Manual grading of essays takes up a significant amount of instructors' valuable time, and hence is an expensive process. Automated grading, if proven to match or exceed the reliability of human graders, will significantly reduce costs. The purpose of this project is to implement and train machine learning algorithms to automatically assess and grade essay responses. These grades from the automatic grading system should match the human grades consistently. Currently, automated grading is used instead of second graders in some high-stakes applications, and as the only grading scheme in low stakes evaluation.

2 Data

We used a data set from a competition on kaggle.com, by the William and Flora Hewlett Foundation. The data consists of 8 sets of essays in ASCII text, written by students from Grade 7 to Grade 10. Essays in each of the 8 sets have unique characteristics that are used for grading. This ensures that the automated grader is trained effectively across different types of essays. Each essay has one or more human scores and a final resolved score. Each essay set has a different grading rubric, holistic in most cases, trait based in one set. Each essay is approximately 150 to 550 words in length. Some essays are more dependent upon source materials than others.

3 Methodology

A basic block diagram level implementation methodology is shown in Figure 1. We extracted a set of features from each essay. We chose features that may serve as proxies for what a human grader might look for while grading the essay. Linear Regression was then used as our learning model to learn parameters based on the features. Scores were predicted for a distinct set of test essays. These scores were compared against human graded scores to arrive at an error metric.

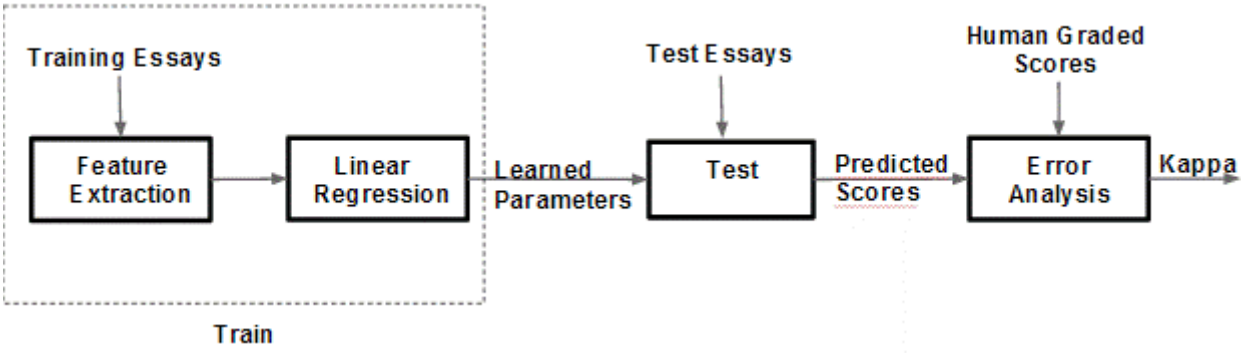

Figure 1: Implementation Methodology.

3.1 Hypothesis

We hypothesize that a good prediction for an essay score would involve a range of feature types such as language fluency and dexterity, diction and vocabulary, structure and organization, orthography and content. A good model would incorporate features from each of these areas to arrive at a good prediction. Unfortunately, we do not have source content available for all the essay sets, and hence will be unable to test for content.

3.2 Software Tools & Packages

We chose to implement our model in Python 2.7.x, since there exist a diverse set of libraries for working with natural language processing. We have used the *Natural Language Toolkit* (NLTK) and *textmining* for most NLP tasks. We also used *scikit-learn* to implement regularized linear regression (in addition to our own implementation). Other libraries (*numpy, scipy, xlrd, xlwt, re*) have been used for various tasks.

4 Feature Extraction

We used *textmining* and *NLTK* libraries for text preprocessing and feature extraction. This mainly involved removing placeholders for proper nouns, preparing the text, tokenizing and stripping essays of punctuation. Intrinsic variables in an essay like the style and fluency cannot be directly measured - they have to be approximated with measurable quantities like the sentence and word length, length of essay etc.

We extracted features across following categories to train our model :

1) Bag of Words (BOW) : We used this as a basis to select features (words) that are good predictors of the essay score. The *textmining* library has a utility called *TermDocumentMatrix* for creating a bag of words. This is a measure of the presence of specific content and concepts in each essay set that the examiner looks for.

For each essay set, the top words were collected after removing stop words like *the, of, is, at*, then compared to the frequency of words in common language. We collected words that are far more frequent than in average writing, and assigned a weight to them according to how frequent they were in the essay set. Each essay was assigned a value based on the number of these top words present and the word weights.

2) Numerical features : Features such as total word count per essay, average word length per essay, sentence count indicate language fluency and dexterity. For feature extraction, the essays were tokenized and split using python utilities. The individual tokens were then used to compute word count, sentences count and character count values.

3) Parts of speech count: Various parts-of-speech such as nouns, adjectives adverbs and verbs are good proxies to test vocabulary. This feature can also be taken as a rudimentary proxy for diction. These features were extracted using python NLTK-part-of-speech tagger. Essays were tokenized into sentences before the tagging process.

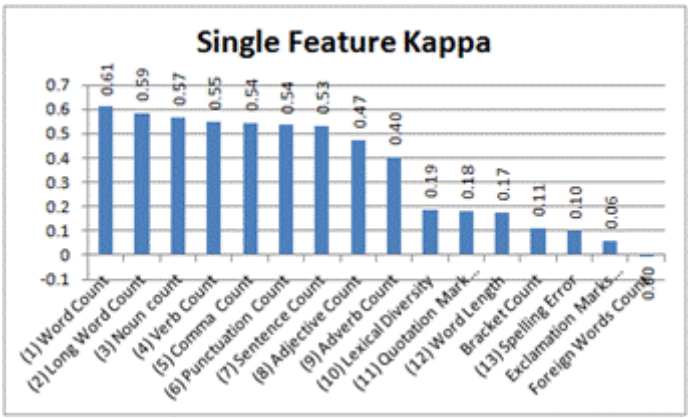

Figure 2: Single Feature Kappa values.

4) Orthography : Correct word spelling indicates command over language and facility of use. We extracted the number of spelling errors per essay to test for these characteristics. The essays were stripped of punctuation and tokenized. We ran the *PyEnchant 1.6.5* spell checker with the *aspell* dictionary to obtain the count of misspelt words per essay.

5) Structure and Organization : Punctuation is a good indicator of a well structured and organized essay. Punctuation based features were extracted using regular expressions matching from individual essays after tokenizing.

5 Learning Model

An overview of related prior work (see references) indicates that linear regression works well for essay grading applications, We chose linear regression as our learning model. We did not have a separate validation test essay set, so we used 5-fold cross validation to train and test our learning model across the whole range of essays at our disposal. This was done to guard against overfitting.

5.1 Evaluation

The error metric for the performance of our learning system is the Quadratic Weighted Kappa. This is a robust error metric, since it takes into account as the baseline the possibility of agreement occurring by chance. This metric typically varies from 0 (only random agreement) to 1 (complete agreement). In the event that there is less agreement between the raters than expected by chance, this metric may go below 0. The quadratic weighted kappa is calculated between the automated scores for the essays and the resolved score for human raters on each set of essays. The mean of the quadratic weighted kappa is then taken across all sets of essays. This mean is calculated after applying the Fisher Transformation to the kappa values.

6 Feature Selection

After implementing a set of features, we proceeded to Forward Feature selection. To decide the order in which to add features, we regressed each feature individually against the human graded scores, and calculated kappa values. We repeated this across all 8 essay sets and averaged the kappa scores. The results were as follows:

We then proceeded to add one feature at a time to our learning model in the descending order of kappa values calculated above. We discarded features which did not improve resulting model's kappa value. We repeated this step across all essay sets and averaged the resulting kappa values. The resulting final list of features and the resulting model kappa is as below. This was the final set of features to train our learning model :

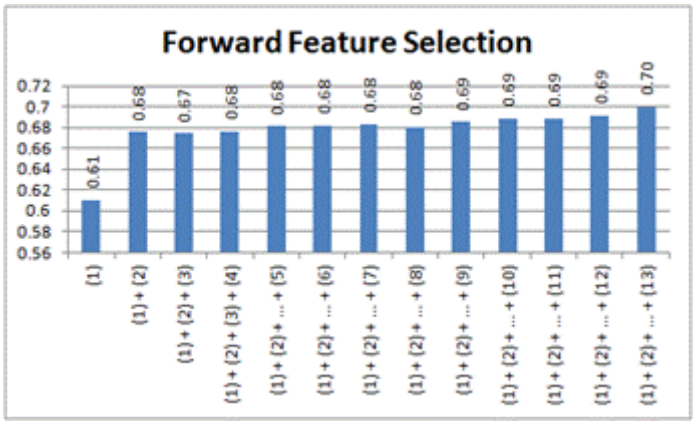

Figure 3: Forward Feature Selection

Note: One feature added at each step, numbering from figure above.

7 Results and Analysis

We used the learning model as described above to train and test across each essay set. 5-fold cross validation was carried out within each essay set. The results are as follows:

Essay Set	**Average Kappa***
1	0.80
2	0.73
3	0.69
4	0.69
5	0.76
6	0.70
7	0.71
8	0.68
Quadratic Weighted Kappa	**0.72**

* Average over 5-fold cross validation, using Fisher Transformation

Also, we normalized scores across all 8 essay sets, and used our learning system to train and test using the entire set of 12978 essays. We again used 5-fold cross validation. The average kappa value for this was 0.73**.** Thus, we do not see a significant improvement in training and testing all essay sets over training and testing individual essay sets**.**

The essay prompt description gives us some insight into why our model works better on some test sets than on others. Sets 1 and 2 are persuasive essays, and hence relatively free from contextual text. Sets 3 to 6 expect a critical response after reading a piece of text (essay, story, book), and is therefore expected to have more context specific content. Sets 7 and 8 are narratives based on personal experience or imagination, and should thus feature rich lexical content and complex sentence structure.

As expected, our model performs relatively well on the persuasive essays. It suffers on sets where context is central to the essay. This suggests that counting the presence of top words of the essay set in the essay is not a sufficiently complete measure of content. Surprisingly, it does well on set 5, which is expected to have context based content. Our model does not work well with narrative essays. This indicates we may need to incorporate features for testing complex sentence structure, like N-grams.

8 Conclusions

As we hypothesized, features from each category contribute towards a good prediction. Our final model contains features across all the categories from section 3.1 we tried to test for. Our model works relatively better on non-context specific essays. Performance on content specific and richer essays can be improved by incorporating content and advanced NLP features.

8.1 Future Work

Even though linear regression worked as a good predictor for essay scores we are did not test if this is the best model for text assessment machine learning problems. There is scope for further exploration and evaluation of alternative models in this area (for example logistic model trees). We could also further improve the model by using more complex features. This would be particularly constructive for context specific essays in the data set. We believe that more advanced NLP features (N-grams, k-nearest neighbors in bag of words) and features that are grammar and usage specific can further enhance the prediction model.

9.0 References

[1] Valenti, S., Neri, F., & Cucchiarelli, A. (2003). An Overview of Current Research on Automated Essay Grading, 2.

[2] Attali, Y., & Burstein, J. (2006). Automated essay scoring with e-raterï¿½ V. 2. *The Journal of Technology, Learning and ...*, 4(3). Retrieved from https://escholarship.bc.edu/ojs/index.php/jtla/article/view/1650

[3] Kaggle (2012). *The Hewlett Foundation: Automated Essay Scoring.* Retrieved 17 October 2012 from Kaggle: http://www.kaggle.com/c/asap-aes

[4] Ng, Andrew (). *CS229 Machine Learning Autumn 2012 Lecture Notes.* Retrieved 16 November 2012 from Stanford University: http://cs229.stanford.edu/materials.html

[5] Lukic, A., & Acuna, V. (n.d.). Automated Essay Scoring, Rice University

[6] Preston, D., & Goodman, D. (2012). Automated Essay Scoring and The Repair of Electronics, 1–18.

[7] Bird, Steven, Edward Loper and Ewan Klein (2009). Natural Language Processing with Python. O'Reilly Media Inc.

[8] Peccei, Christian. "Textmining." .<http://www.christianpeccei.com/projects/textmining>.

[9] Machin, John. "xlrd0.8.0." . <http://pypi.python.org/pypi/xlrd>.

[10] Machin, John. "xlwt 0.7.4." . <http://pypi.python.org/pypi/xlrd>.

[11] Jones, Eric, Travis Oliphant, Pearu Peterson, and et al. "SciPy: Open source scientific tools for Python." . <http://www.scipy.org/>.

[12] Kelly, Ryan. "PyEnchant." <http://packages.python.org/pyenchant/>.

[13] Pedregosa, F., Weiss, R., & Brucher, M. (2011). Scikit-learn : Machine Learning in Python, 12, 2825–2830.