IJCAT - International Journal of Computing and Technology, Volume 5, Issue 2, February 2018
ISSN (Online) : 2348-6090
www.IJCAT.org

Automated Essay Grading using Natural Language Processing and Support Vector Machine

[1] Abhishek Suresh, [2] Manuj Jha

[1] Department of Computer Science and Department of Linguistics
University of Colorado Boulder, Colorado, United States

[2] Rawls College of Business
Texas Tech University, Texas, United States

Abstract - This paper proposes to grade various essays and literary materials automatically using the feature extraction techniques from Natural Language Processing (NLP) and Support Vector Machine (SVM), a powerful machine learning algorithm for classification, modelled around Education Testing Service's GRE Analytical Writing scoring guidelines. We extracted various features like word count, TF-IDF score, number of paragraphs, part of speech tagging and number of spelling mistakes on the essay dataset sourced from Kaggle [1]. After extracting the features using NLP, there were two possible approaches to tackle the problem; a regression model or a model based on classification. We used a classification-based approach to train our model with training essays, normalized to a scale of 1 to 6. Upon predicting the grades for the essays in testing set, we found that the accuracy of our model stood at 0.52, and 0.89 with a tolerance of one point, as permissible by ETS that uses automatic essay grading [2]. Individual essay sets can be graded with an automatic grading framework, a lot of human effort could be saved and literary pieces can be graded with transparency.

Keywords - *Natural Language Processing, Essay Grading, Machine Learning, Support Vector Machine*

1. Introduction

In today's education system, essays and literary pieces form an integral part of assessing one's academic understanding, ability to integrate and express ideas meaningfully. It is observed that grading such written pieces is a time-taking process and take up a huge chunk of instructor's time. Also, cases of manual grading sometimes being non-transparent are not unheard of. So, we plan to tackle this challenge by introducing a method to automatically grade written pieces. Automated essay scoring is a measurement technology in which computers evaluate written work [3]. With a model that can grade with good accuracy, a lot of instructor's time can be saved and an impartial scoring for students can be achieved. We have used a classification-based approach as with lower number of possible grades (1 to 6), it makes more sense to classify the essays. In a case with grading on a higher scale; let's say 0 to 100, a regression-based method would have been more suitable. We have decided to use SVM because despite the huge time it takes once to train the model, predictions are made with relatively ease and great speed.

2. Related Work

There are various testing services and commercial applications which utilize automated grading.

2.1 Bayesian Essay Test Scoring System (BETSY)

BETSY is a program that classifies text based on trained material into a four-point nominal scale (e.g. extensive, essential, partial, unsatisfactory) using a large set of features including both content and style specific issues [2]. It uses Multivariate Bernoulli Model (MBM) and Bernoulli Model (BM), which are considered naïve Bayes because they assume conditional-independence. BETSY is a Windows-based program written in Power Basic and is computationally intensive. It is one of the very few applications in the field which are freely available and usable. For a naïve Bayes model, it provides a good accuracy of around 80%.

2.2 Educational Testing Services (ETS)

ETS is a testing agency which conducts standardized tests like GRE and TOEFL. There are analytical writing segments in these exams which are usually graded by their proprietary grading framework. This system was developed in early nineties and works on a sentence fragment of length between 15 to 20 words [2]. According to ETS, this technology extracts certain features representative of writing quality that not only predict scores accurately, but also have a 'logical correspondence to the features that readers are instructed to consider when they award scores'. It uses Microsoft Natural Language Processing (MsNLP) tool to parse the essay data, suffixes and stop words are removed. The technique uses lexical-semantic technique for scoring and builds domain specific, concept-based lexicon from training data [4]. Grammar rules are constructed manually for each category of answer using syntactic parses of sentences from the training data along with the lexicon [4]. New essays are then parsed to get part of

speech for different words and phrases. The score for every GRE essay is the average of the scores of one human grader and the scoring engine; if these two scores differ by more than one point (out of six), the essay is scored by another human grader and that score is used to compute the average instead of the scoring engine's score. The features currently included in the scoring engine are [5]:

- content analysis based on vocabulary measures
- lexical complexity
- proportion of grammar errors
- proportion of usage errors
- proportion of mechanics errors
- proportion of style comments
- organization and development score
- features rewarding idiomatic phraseology

2.3 Conceptual Rater (C-Rater)

C-Rater is a NLP based prototype aimed at the assessment of short answers related to content-based questions. It is aimed to categorize a response as being either correct or incorrect which is achieved by evaluating whether response contains information related to specific domain concepts. It classifies the essay based on the content rather than style of writing, which is the critical factor in E-Rater. C-Rater achieved over 80% agreement with the score assigned by the instructor [2].

3. Methodology

The data consists of eight essay sets with between 1000-3000 essays in each set, and the rubrics for grading the essays. Each essay is about 100-550 words long. Some of the essays are more dependent on a particular source text than others, and the rubrics reflect this in their grade brackets.

The data for each essay set consists of the essays themselves, one or more scores for each essay (the score ranges are different across the essay sets) and a resolved score if there is more than one score.

This is the data that will be used to train this essay grader algorithm. The input data for the grader would be entire essays (and their corresponding question prompts) and the output would be scores. The score ranges for the essay sets vary, and for the sake of uniformity they have been normalized to a scale of 1-6, to mimic the Analytical Writing scoring scale used in GRE.

The workflow for this project is first extracting the relevant features from the essay, using a Support Vector Machine model for training and then comparing the predicted scores against the actual scores graded by evaluators to get an error metric.

Features extracted

3.1 TF-IDF prompt-essay correspondence score

TF or term frequency is the number of times a word appears in the document. DF or document frequency is number of times a word occurs in all the documents [6]. TF multiplied by the inverse of DF gives TF-IDF. It is a metric that reflects how significant a particular word is in a document in the collection of documents. Term frequency is proportional to the weight of a term that occurs in a document, e.g. count or frequency of a word in the document, and inverse document frequency is an inverse function of count or frequency of documents in which it occurs, e.g. the logarithm of the inverse of the ratio of documents in which the word occurs to the number of documents in the corpus.

By generating TF-IDF scores for all words in every essay and in its relevant prompt, we obtain two vectors of unigrams and their TF-IDF scores, with which we calculate the cosine similarity between these vectors. This value acts as a correspondence score between the prompt and the essay.

3.2 TF-IDF scores of bag of words

Generating TF-IDF scores of n-grams and using these as features helps in determining which words are good predictors of final essay score, as opposed to simply using a bag of words, with no measure of how important each word is in the document.

3.3 POS tags and lexical density

POS tags act as a good proxy for semantic difficulty, so we use counts of the various POS tags and their bigrams as well. We also calculate lexical density as an overall metric for semantic difficulty; this is simply a ratio of the count of all lexically important POS tags such as nouns, adjectives and adverbs to the count of all tags.

3.4 Statistical metrics

These are other features that may reflect syntactic complexity and semantic difficulty, such as word length, word count, sentence count and paragraph count. These are converted to z-scores on standardized scales which work better with the SVM model [7].

3.5 Spelling errors

By checking every word against a corpus of English words from Python's NLTK library, we obtain a count of spelling errors for each essay.

3.6 Type-token ratio

A metric for lexical diversity; essentially a ratio of the number of unique words in each essay to the overall number of tokens.

3.7 Support Vector Machine

The Support Vector Machine (SVM) is a supervised classification algorithm which was first proposed by Vladimir N. Vapnik in 1963. In 1992, he suggested a way to create non-linear classifiers by applying the kernel to maximum-margin hyperplanes and has since attracted a high degree of interest in the machine learning research community [8]. SVMs have been deployed in a wide range of real world problems such as text categorization, handwritten digit recognition, tone recognition, image classification, object detection and data classification. It has extensive support for kernels to deal with non-linearity of data and it needs to be finely tuned to get the best results [8]. NLP tasks typically represent instances by very high dimensional but very sparse feature vectors, which leads to positive and negative examples being distributed into two distinctly different areas of the feature space. This is particularly helpful for the SVM to search a classification hyperplane in feature space and for the generalisation capability of the classifier as well. That is a main reason why the SVM can achieve very good results in a variety of NLP tasks [9]. The algorithm takes longer time to train compared to other classifiers because of vector calculations, which are mathematically intensive. The time taken to predict is very low, which is desirable for our project since after training the model once, it can score the essays very fast.

4. Result

We used exact one-to-one correspondence percentages and percentage accuracy allowing for an error of one point from the actual scores to evaluate our model. The results for each essay set are in the Table 4.1.

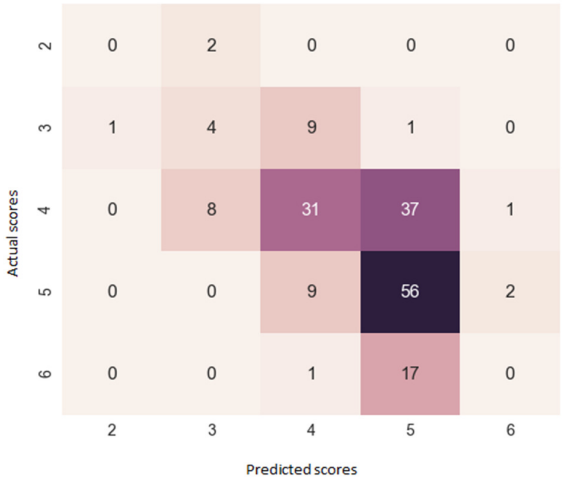

Figure 4.1

Essay Set	Exact Accuracy	Accuracy allowing for one-point Error
1	0.51	0.98
2	0.51	0.96
3	0.53	0.70
4	0.60	0.85
5	0.55	0.87
6	0.49	0.88
7	0.45	0.92
8	0.40	0.92

Table 4.1

Figure 4.1 is a confusion matrix of the results obtained for essay set 1. For the essays belonging to every actual human-graded score, this confusion matrix shows their distribution across the predicted scores. As can be seen, the accuracy of the algorithm is less than 60% when using a strict one-to-one method of evaluating the predicted classes against the actual grades, but allowing for an error of 1 point gives a high accuracy. The confusion matrix shows that most essays were either rated accurately or very close to the gold class.

5. Conclusion

In this paper, we identified a classification-based approach to solve the problem of grading literary materials manually. We used Natural Language Processing to extract various features which are characteristic of a good writing. The accuracy of model could be further improved if a metric for similarity between the essay topic/ problem statement were added. The topic on which essay was written was not described in the dataset we used, but this is mostly known in most of the exams/ standardized tests. The model we designed performed reasonably well with an allowance of one point in marking and could definitely be used to grade written essays/ literary materials.

References

[1] Kaggle. *The Hewlett Foundation: Automated Essay Scoring*. Available from: https://www.kaggle.com/c/asap-aes.

[2] Valenti, S., F. Neri, and A. Cucchiarelli, *An overview of current research on automated essay grading*. Journal of Information Technology Education: Research, 2003. **2**(1): p. 319-330.

[3] Shermis, M.D., et al., *Automated essay scoring: Writing assessment and instruction*. International encyclopedia of education, 2010. **4**(1): p. 20-26.

[4] Drolia, S., et al., *Automated Essay Rater using Natural Language Processing.* International Journal of Computer Applications, 2017. **163**(10).

[5] Ramineni, C., et al., *Evaluation of the e□rater® Scoring Engine for the GRE® Issue and Argument Prompts.* ETS Research Report Series, 2012. **2012**(1).

[6] Drucker, H., D. Wu, and V.N. Vapnik, *Support vector machines for spam categorization.* IEEE Transactions on Neural networks, 1999. **10**(5): p. 1048-1054.

[7] Abhishek Suresh and Manuj Jha, *Automated Essay Grading Python Code*. Available from: https://github.com/absu5530/AES/blob/master/AES.py

[8] Durgesh, K.S. and B. Lekha, *Data classification using support vector machine.* Journal of Theoretical and Applied Information Technology, 2010. **12**(1): p. 1-7.

[9] Li, Y., K. Bontcheva, and H. Cunningham, *Adapting SVM for data sparseness and imbalance: a case study in information extraction.* Natural Language Engineering, 2009. **15**(2): p. 241-271.

Abhishek Suresh received his BTech in Mechanical Engineering from Manipal Institute of Technology, Karnataka, India. He worked as a Trainee Decision Scientist at Mu Sigma Business Solutions Pvt. Ltd., Bengaluru, India after graduation. Currently he is pursuing Master's in Computational Linguistics, Analytics, Search and Informatics at the University of Colorado Boulder. His interests include computational linguistics, NLP, machine learning and text analytics.

Manuj Jha received his BE in Telecommunication from R.V. College of Engineering, Bengaluru, India. He worked as a Trainee Decision Scientist at Mu Sigma Business Solutions Pvt. Ltd., Bengaluru, India after graduation. Currently he is pursuing Master's in Data Science at Texas Tech University. His interests include NLP, machine learning and descriptive analytics.