Machine Learning Engineer Nanodegree
Capstone Project
Automated Essay Checking using Machine Learning

Vasu Mistry

May 29, 2017

1 Definition

1.1 Domain Background

Automated essay scoring (AES) has been in the research area of computer science since the early 1966 [1]. Predicting the score of an essay so that the score might seem like it has come from a human reader is a bit daunting task because there are numerous quantified features that have to be extracted from the essay as well as many unquantifiable properties like the perceptions of the writer while writing the essay and his thoughts that he is trying to inscribe on the paper. Therefore, the behavior of the essay inherently noisy, non-stationary and deterministically chaotic. The quantifiable data that can be extracted from an essay is relatively easy for the computer to process rather than processing the ideas or thoughts of the writer in the essay, which may or may not affect the scoring of an essay by a computer.

1.2 Motivation

Personally, I came across this problem during my internship at a start-up. For recruiting content-writers for their website, during the interview phases, a candidate must write a 250 word long essay to demonstrate his skills. At present, this grading is being done manually by the start-up. My goal is to make grading of these essays automatically and I plan to begin my approach through this Capstone Project.

1.3 Problem Statement

The Hewlett Foundation sponsored the Automated Student Assessment Prize on Kaggle - AES Challenge, challenging teams to produce essay evaluation models that best approximate human graders. Many competitive exams try to include maximum number of multiple choice questions since written essays are being graded manually and take up a lot of time to evaluate. While it is a known fact that written essays provide opportunities to challenge the students with more sophisticated measures of ability, the ease of bubbled-answers checking promises a faster evaluation of the student.

The project aims to build a regression model that can take in an essay and automatically output the grade of that essay. Using feature-extraction and Machine Learning algorithms, this task can be automated. The output value will be a continuous value between 2-12.

1.4 Evaluation Metrics

The competition's evaluation metric *Quadratic weighted Kappa* will be used as the evaluation metric for this project. This metric is a statistic which measures inter-rater agreement for qualitative (categorical) items. It is generally thought to be a more robust measure than simple percent agreement calculation, since κ takes into account the possibility of the agreement occurring by chance [1]. The metric will also allow us to compare our results with the scores obtained by the participating teams as well as with our benchmark.

The quadratic weighted kappa score is a measure of agreement of our scores and the human annotator's gold-standard. 0 represents only random agreement between the raters and 1 is full agreement. For N possible essay ratings, an N X N matrix O is constructed where $O_{i,j}$ represents the number of essays receiving grade i from the first grader and j from the second rater. Additionally, a matrix E is constructed the same way, but assuming there is no correlation. The matrices are normalized so that they have the same sum. An N X N matrix w is also calculated where:

$$w_{i,j} = \frac{(i-j)^2}{(N-1)^2} \tag{1}$$

The quadratic weighted kappa is calculated by:

$$\kappa = 1 - \frac{\sum_{i,j} w_{i,j} O_{i,j}}{\sum_{i,j} w_{i,j} E_{i,j}} \tag{2}$$

The Quadratic Weighted Kappa metric typically varies from 0 - only random agreement between raters - to 1 (complete agreement between raters). In the event that there is less agreement between the raters than expected by chance, this metric may go below 0. [2]

2 Analysis

2.1 Datasets

The dataset was provided as a part of the Kaggle - AES Challenge [3]. For this competition, there are eight essay sets with an average length of 150-550 words. The students writing these essays are from Grade 7 to Grade 10. Each essay was graded by 2 or 3 graders and each generated from a single prompt.

For the purpose of this project, Essay sets [1-3] were used to train and test the learning model. The model is capable of acknowledging the difference in scale and outputting the corresponding grade.

Essays had been anonymized before being released to the public using the Named Entity Recognizer (NER) developed by the Stanford Natural Language Processing group [2]. Replacement IDs of the @ sign followed by words in all capitals were used instead. Name Entities of People, Organizations, Locations, Times/Dates, Numbers, Percents, E-mail Addresses, and Money were replaced.

[1] https://en.wikipedia.org/wiki/Cohen%27s_kappaWeighted_kappa

[2] Evaluation function present in : utils/metrics.py

[3] https://www.kaggle.com/c/asap-aesm

Essay Set	Essay Type	Domain	Grade Level	Score Range	Average Length of words	Total
1	Persuassive/Narrative /Expository	-Letter Writing	8	2-12	350	1783
2	Persuassive/Narrative /Expository	- Writing Applications - Language Conventions	10	- 1-6 - 1-4	350	1800
3	Source Dependent Responses	-	10	0-3	150	1726

Table 1: Data-set information and statistics

2.2 Inputs

The data set contains following parameters:

- essay_id: A unique identifier for each individual student essay
- essay_set: The set of a given essay.
- essay: The ascii text of a student's response
- rater1_domain1: Rater 1's domain 1 score
- rater2_domain1: Rater 2's domain 1 score
- domain1_score: Resolved score between the raters.

The following example contains an essay from the data set:

Essay Id	Essay Set	Essay	Rater1_score	Rater2_score	Domain1_score
27	1	Reference: Table 3	4	4	8

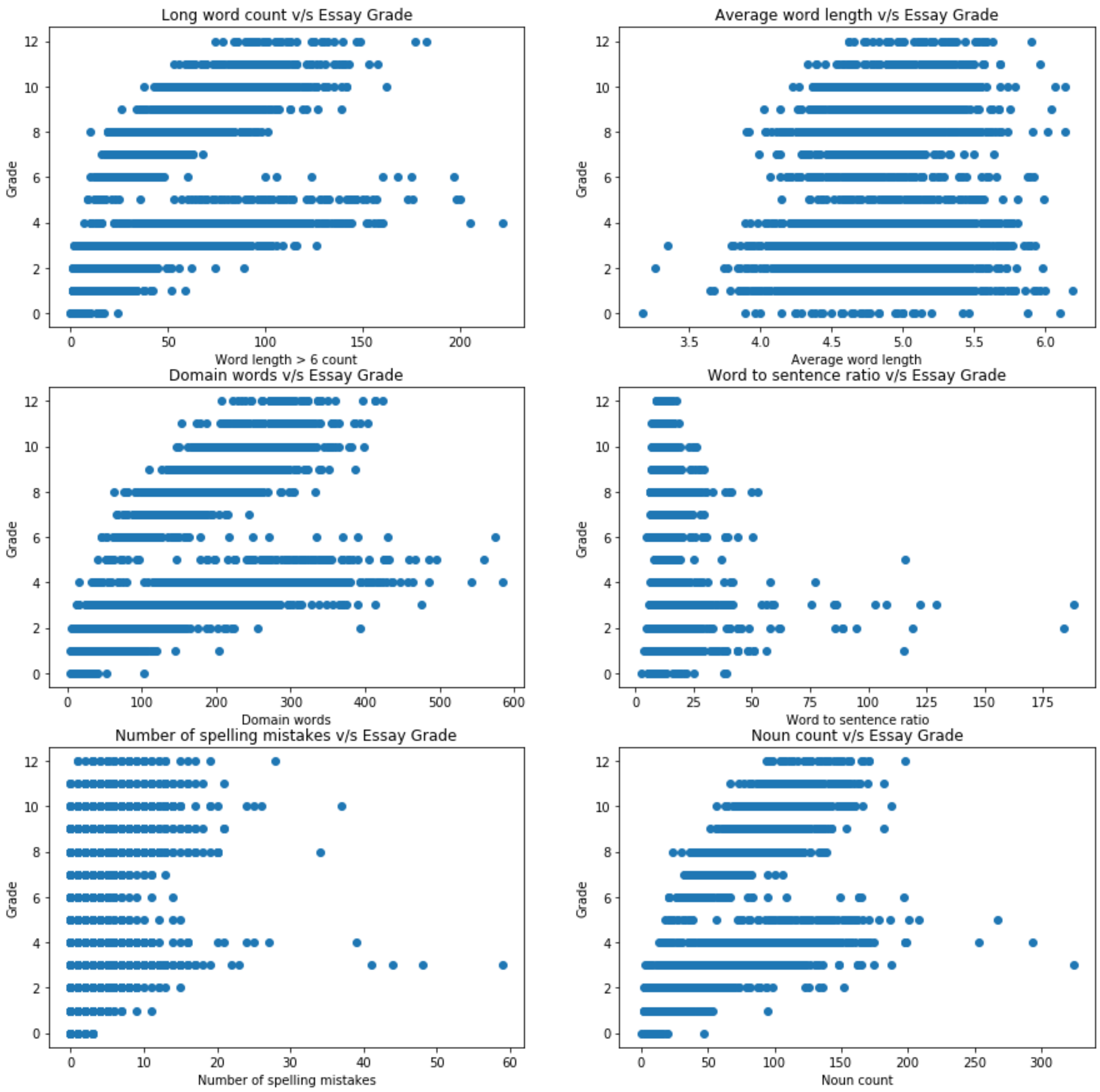
Table 2: Sample data entry

Computers a good because you can get information, you can play games, you can get pictures, But when you on the computer you might find something or someone that is bad or is viris. If ther is a vris you might want shut off the computers so it does not get worse. The are websites for kids, like games, there are teen games, there are adult games. Also pictures are bad for kids because most of the time they lead to inapropreit pictures. You should only look up information that you need not things like wepons or knifes. Also there are differnt kinds of companies like @CAPS1&t @CAPS2. @CAPS2 is a good place to get computers @CAPS1 so is @CAPS1&t.

Table 3: Sample essay from data set

2.3 Explanatory Visualization

Essay Sets - [1,2,3]



It is inevitable for a good essay to follow a good sentence structure. From the custom features generated [4], we can clearly see this general trend being followed. Interestingly, it can be observed here that essays with large number of spelling mistakes can receive a higher grade. I would argue that with the advancement of technology in modern spell checker's, the importance for a student to spell a word correctly is given less importance now. The graders clearly consider that the theme/message the student wants to convey in the essay is more important unlike these specific errors.

2.4 Algorithms and Techniques

2.4.1 Linear Regression

An overview of related prior work [3][4][5] indicates that linear regression works well for essay grading applications. In Linear Regression, an output vector denoted y is generated, based on features x extracted from a given essay. Given an input feature vector [5] $x \ \epsilon \ R^m$, an output vector [6] $\hat{y} \ \epsilon \ R$ using a linear model with a weight of β:

$$\hat{y} = \ \beta_0 + \ x^t \beta \tag{3}$$

To learn values for the parameters $\theta = (\beta_0, \beta)$ the sum of squared errors for a training set containing n pairs of essays is minimized and scores,(x_i, y_i) where $x_i \ \epsilon \ R^m$ and $y_i \ \epsilon R$ for $1 \leq i \leq n$:

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}=(\beta_0,\boldsymbol{\beta})}{\arg\min} \frac{1}{2n} \sum_{i=1}^{n} \left(y_i - (\beta_0 + \boldsymbol{x}_i^\intercal \boldsymbol{\beta})\right)^2 \tag{4}$$

2.4.2 Word2vec model

Word2vec is a group of related models that are used to produce word embeddings. These models are shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words. Word2vec takes as its input a large corpus of text and produces a vector space, typically of several hundred dimensions, with each unique word in the corpus being assigned a corresponding vector in the space [7]. Word vectors are positioned in the vector space such that words that share common contexts in the corpus are located in close proximity to one another in the space [6]

The model will generate a word vector for each unique word in the training set. This vector will be combined with custom heuristic features to create a final vector for each word in the training set.

2.4.2.1 The skip gram model

The training objective of the Skip-gram model is to find word representations that are useful for predicting the surrounding words in a sentence or a document. More formally, given a sequence of training words $w_1, w_2, w_3, ..., w_t$, the objective of the Skip-gram model is to maximize the average log probability:

$$\frac{1}{T} \sum_{t=1}^{T} \sum_{-c \leq j \leq c, \ j \neq 0} log \ p \left(w_{t+j} | w_t\right) \tag{5}$$

[4] Python Notebook : Data_Exploration.ipynb

[5] For building this vector, Google's word2vec model and a few custom generated heuristic features will be used. They are explained in the later sections of this report.

[6] The scores predicted by the learning model

[7] https://en.wikipedia.org/wiki/Word2vec

where c is the size of the training context (which can be a function of the center word w_t). Larger c results in more training examples and thus can lead to a higher accuracy, at the expense of the training time.

2.4.3 Forward feature selection

It is possible that all of the custom heuristic features generated during the feature extraction will not contribute positively towards grading the essay. To ensure that all features contribute positively towards the learning and to eliminate the poor performing features, the greedy forward feature [8] [9] algorithm will be used.

The forward feature selection algorithm works by making changes to the set of features and only keeping the new set if there is an increase in accuracy. Greedy Forward Search works by starting with just one feature and incrementally adding in all the other features. As each feature is added in, the model is evaluated with the feature set and the new feature is only kept if there is a notable increase in accuracy. This is a greedy solution and may not find the absolute optimum feature set, however by looking at which features cause an increase in accuracy it will pick out useful features for the learning model. Also, because the accuracy of the model is evaluated with all the features in a set, this method will pick out features which work well together to achieve a higher accuracy score. Features are not assumed to be independent and so advantages may be gained from looking at their combined effect. [7]

2.5 Benchmark Model

The automated reader developed by the Educational Testing Service, e-Rater, used hundreds of manually defined features. It was trained on 64 different prompts and more than 25,000 essays. Evaluated on the quadratic weighted kappa calculated between the automated scores for the essays and the resolved score for human raters on each set of essays, e-rater could only achieve the kappa score below 0.5 [8]. Hence, I would desire to develop a learning model that has at least a kappa score of 0.5.

3 Methodology

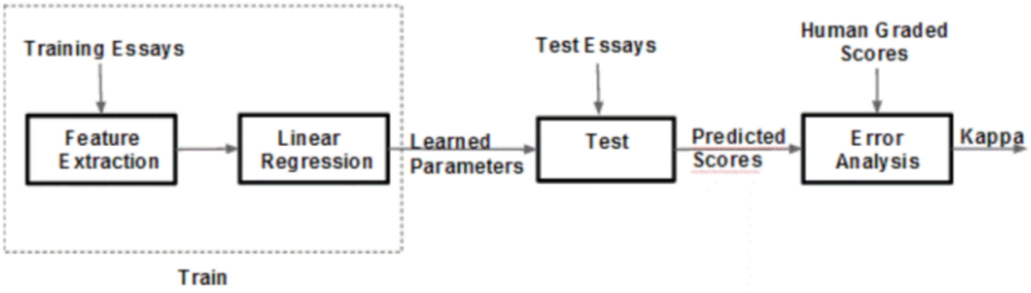


Figure 1: Implementation Methodology

[8] https://en.wikipedia.org/wiki/Feature$_{s}election$

[9] https://www.cs.cmu.edu/ kdeng/thesis/feature.pdf

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 [3].

3.1 Data pre-processing

The data had already been pre-processes by using the Stanford Natural Language Processing group [2]. Replacement IDs of the @ sign followed by words in all capitals were used instead. Name Entities of People, Organizations, Locations, Times/Dates, Numbers, Percents, E-mail Addresses, and Money were replaced. (Reference: Table 3 - Sample data from data set)

There were no features available for the training set, so some custom heuristic features were generated. The *GenerateFeatures()* and *FeatureSet2()*[10] functions were utilized to accomplish this task.

3.1.1 Feature extraction:

Total 16 features were generated using open source libraries [11]. The words already marked by the *"@Text"* in the essay were not considered for generating the features.

- Heuristic features
  Several heuristic features that are likely to contribute to a good essay were generated. Some of the heuristic features are: word count, long word count (words with $length > 6$), average word length per essay, quotation mark count, number of characters, sentence count and comma count.

- Spelling and Grammatical features
  It is likely for a student to make grammatical and spelling errors, number of spelling mistakes and grammatical mistakes were generated using Grammar-Check.

- Part of Speech (POS) tags
  A **count** for most regular POS tags that helps to identify a good sentence structure were used, for example, count of Nouns, Verb, Adjective and Adverb for a given essay using NLTK

- Other features
  Other important features like wrong words, domain words (number of words that relevant to the domain of the essay), punctuation count, word to sentence ratio etc were also generated using NLTK's WordNet library.

3.1.2 Feature generation issue

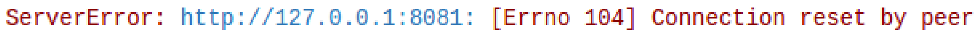
The Grammar-Check library that was used for generating number of spelling mistakes throwed an error after generating features for approximately 900 essays. The error was

To avoid this error, features were generated in a batch of 800 essays. After generating a batch of essays, I waited for 2-3 minutes and called the feature generation function again to generate features for the next batch. After generating the features for all the essays, they were stored [12] so that the features can directly be imported by other notebooks.

[10] File location : utils/helperfunctions.py

[11] List of libraries used : Libraries.txt

[12] Features : *model_and_visualization/features_set_1.csv*, Output : *model_and_visualization/target_set_1.csv*



```
ServerError: http://127.0.0.1:8081: [Errno 104] Connection reset by peer
```

Figure 2: Grammar-Check library server error

3.2 Implementation

Single feature Kappa: After generating the heuristic features, each feature was evaluated individually to check their contribution towards the learning model. The *Evaluate()* function from the *helperfunctions.py* file was utilized. The function uses LinearRegression as learning model with a 5 fold cross validation strategy with quadratic weighted kappa as the evaluation metric to guard against over fitting.

Single feature Kappa obtained :

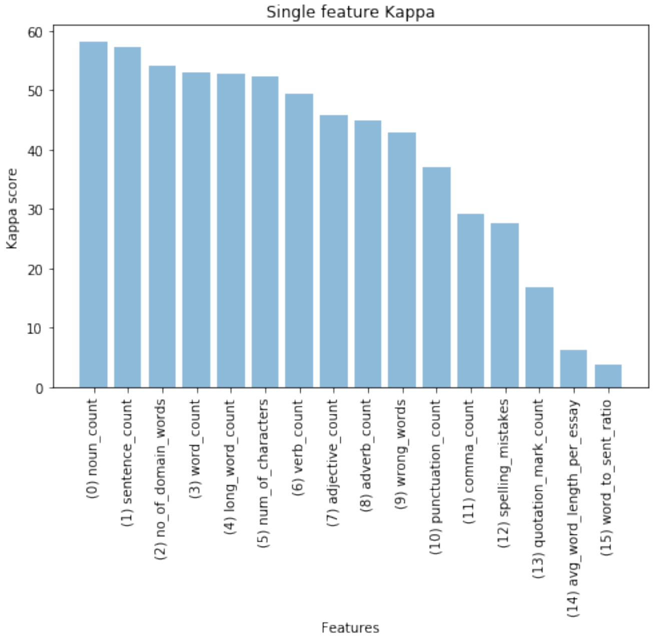


Figure 3: Single feature Kappa scores

Greedy Forward Selection : The features were sorted in decreasing order based on the single kappa scores obtained. The forward selection algorithm was applied and the feature set with highest Kappa score (Set [1-15]) was selected. The highest Kappa score obtained was 0.6740. This score was above our set benchmark of 0.5. Only the last feature (*word_to_sent_ratio*) was not used in the feature set with the highest Kappa score, we can say that almost all the features contributed towards increasing the Kappa score for the model.

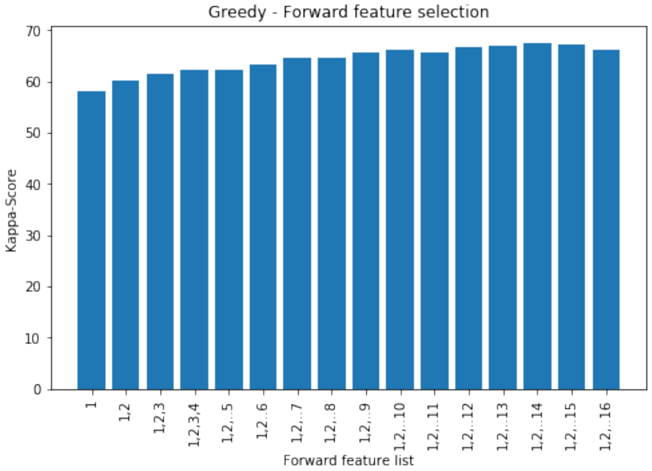


Figure 4: Forward Feature Kappa score (Indexing as in Figure 3)

3.3 Refinement : Word2vec model

The Word2vec model has shown promising applications in the Natural Language Processing (NLP) domain since its introduction in 2013. To increase the accuracy of the model, I used the word2vec model from the gensim library and combine the word vectors generated with the heuristic feature vectors implemented earlier. Any reasonable increase in accuracy was welcomed since using only the heuristic features are no longer preferred for NLP tasks.

Library functions : *essay_to_wordlist(), essay_to_sentences(), makeFeatureVec()* and *getAvgFeatureVecs()* [13]

Sentences were extracted from all the essays in the training set using the *essay_to_sentences()* function. NLTK's punkt sentence tokenizer was used for this purpose. This function called the *essay_to_wordlist()* function to perform word tokenization on each word in the given sentence. Stopwords [14] ('yourself', 'but', 'again', 'there', 'about', 'once') and already tagged words ("@Text") were not considered for this.

A word2vec model consisting of 300 features (300 dimension word vector) was generated using the *sentences* list generated above. Training vectors and Testing vectors were generated using the *getAvgFeatureVecs()* function. These vectors were then concatenated with the heuristic features vectors generated earlier.

- Dummy Regressor: I ran the model using sklearn's Dummy Regressor [15] to see if the learned model actually performs better than a model guessing only one answer always. It scored a Kappa value of 0.0, indicating no agreement between the predicted values and the human graders.

- Linear Regression: A Linear Regression model with 5 fold cross validation was then trained using the combined vectors of word2vec model and the heuristic features. The model was

[13] Location: utils/helperfunctions.py
[14] https://en.wikipedia.org/wiki/Stop_words
[15] http://scikit-learn.org/stable/modules/generated/sklearn.dummy.DummyRegressor.html

then evaluated on using the test vectors and after cross validation an average Kappa score of **0.9359** was obtained.

4 Results

4.1 Model evaluation and Validation

Id	Model	Kappa Score
1	Dummy Regressor.	0.0
2	Linear Regression with heuristic features.	0.6714
3	Linear Regression with word2vec model.	0.9035
4	Linear Regression using Word2vec and heuristic features.	0.9359
5	Competition's winning team. [16]	0.8141

Table 4: Results

- The heuristic features were generated based on the hypothesis presented in Section 3. However, the advancing research in the NLP domain allowed us to use the word2vec model which proved to be the difference between a good model and an excellent model for this project. The use of forward feature selection algorithm ensured us that the selected heuristic features contributed positively in the model.

- The 5 fold cross validation allowed the model learn the underlying trend of data sets, learning different sentence structures, most commonly used part of speech tags etc and generate word vectors for different kind of words used by students in Grade 8 and 10. Similarly, it also allowed us to test the learning model on some good testing points i.e. testing on essays containing less scores and essays containing high scores. Iterating through the training and testing process multiple time ensured that the learning model was tested on all possible inputs in the data set. The mean average Kappa score obtained is highly encouraging.

- Only essay set 1,2 and 3 were used for the part of this project. Adding the other 5 essay sets to the model should not decrease the efficiency of the model since the essays in these 5 sets contain essays similar to essays in essay set 3, generated by a single prompt. The heuristic features were generated by considering the general necessary features for a good essay, these should be applicable on any essay in any domain. Similarly, the word2vec model was also trained on a large corpus. This should ensure that the model is applicable for any input essay domain that it has been trained on.

4.2 Justification

- The final model consisting of the word vectors generated by the Word2vec model and heuristic features performed exceptionally well. The score improved by 0.2643 when to compared to using only heuristic features, an approach which was followed by many participants during the competition. The final score of 0.9359 surpasses the benchmark set by a convincing margin of 0.4357 and the model performed even better than the score of the winning team of the competition, confirming the usefulness of using word2vec model for NLP tasks.

- With the current research in the NLP domain, we believe that the adaptability of the word2vec model ensures that the model after training on any given data set can be used for the essay grading projects.

5 Conclusion

5.1 Free-Form Visualization

Id	Essay id [17]	Essay snapshot	Predicted Score	Actual Score
1	3	Dear Local Newspaper, @CAPS1 I have found that many experts say that computers do not benifit our society. .... Computers help people reaserch subjects for school reports, and they make the current economy get better everyday. I n moderation computers are the most useful tool out there.	10.36	10
2	26	Computers a good because you can get infermation, you can play games, you can get pictures .... also there are diffent kinds of companies like @CAPS1&t @CAPS2. @CAPS2 is a good place to get computers @CAPS1 so is @CAPS1&t.	5.04	4
3	1118	In my opinion i think that computers help people. It helps them with imformation that they @MONTH1 need. It can also help learn a lot of things.	4.49	2

Table 5: Examle of predicted scores on selected essays

On visualizing the predicted scores by the model for a few selected essays, we can observe that that model prediction for high scoring essays is more accurate than the prediction for essays with poor grade. The model was also able to identify the different score ranges of the various essay types and adapt accordingly.

A principal component analysis (PCA) for visualizing good essays ($grade > 8$) and poor essays ($grade < 4$) was also done. 100 good essays and 100 poor essays were extracted from the data-set, the PCA visualization obtained is :

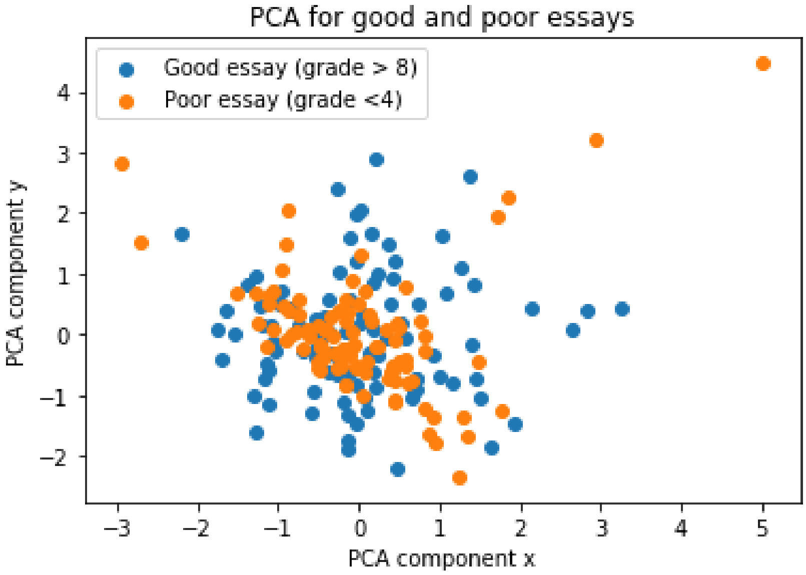


Figure 5: PCA on good and poor essays

Approximately, here, the poor essays are confined (excluding the outliers) to a region and are highly dense in this region compared to the good essays which are spread over the graph. This may be leading us to the fact that the general mistakes that students make in their essays which lead to a poor grade are similar in nature (i.e. similar types of mistakes) whereas for an essay to obtain a good grade consists of several components eg. strength of logical progression of the essay and optimal essay length. Essays with poor grade are likely to follow the common trends like less essay length compared to the average essay length, limited use of vocabulary, less number of domain words etc. The heuristic features and the Word2vec model somewhat captured this idea and the result is visualized in the PCA.

5.2 Reflection

The goal of the project was to find a reliable automated approach for grading of essays. Due to lack of practical hands-on experience with Deep Learning, a relatively simple model of Linear regression was chose for this project [18]. The model uses simple features that concentrate on the sentence structures of the essay and the similarities of the words used in the essay. The end results shows us that an approach without involving Deep Learning, a model with a high accuracy can be generated for this project.

Analyzing the sentence structure that serves as a benchmark for a good essay, exploring the latest researches in the NLP domain, the immense usefulness of the open source NLP libraries proved to be a great learning curve for me.

Due to a confined data-set, essays similar to the training essays can be evaluated using this model directly with appreciable results. However, for essays in different domain,the model can serve as a benchmark for future tasks in the field of Automated Essay Grading.

[18] Most used model in previous approaches

Using simple features, the model performed better than the benchmark set for this project. However, the introduction of Word2vec demonstrated the importance of ongoing research in the NLP domain, how simple tasks that take up majority of a professionals work day can be automated using Machine Learning reliably.

Researching for relevant libraries and reading their documentations were time consuming but very exciting and fascinating. This project being the first individual project that I worked on from scratch, it was difficult in the beginning to handle the data and manage them efficiently. The project provided hands-on experience to work on a problem with a real life application and produce highly satisfying results.

With the personal motivation involved, the model has helped me analyze the inputs required to develop an essay grader that can work for a given domain. The finding reported in this project will prove to be very helpful in the future work carried out at the start-up that I pursued my internship in.

5.3 Improvement

Deep Learning: The Deep Learning approach is very popular at present, by using TensorFlow and it's awesome visualizing tools, the deep learning algorithms provided a great result for this task. I would like to continue this project and follow the approach as publsihed by Huyen Nguyen and Lucio Dery [9], to follow an approach that does not require any featue engineering and automatically learns the representations required for the task.

As reported by Huygen and Lucio in their paper [9], the Deep Learning techniques achieved an average Kappa score of 0.9447 using 2 layer neural network that trains word vectors together which is better than this model.

References

[1] Md. Haider Ali Annajiat Alim Rasel Arshad Arafat, Mohammed Raihanuzzaman. *Automated Essay Grading with Recommendation.* 04 2016.

[2] Trond Grenager Jenny Rose Finkel and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. *Proc. of ACL*, pages 363–370, 2005.

[3] Ashwin Apte Manvi Mahana, Mishel Johns. *Automated Essay Grading Using Machine Learning.* 12 2012.

[4] Neri F. Cucchiarelli A. Valenti, S. *An Overview of Current Research on Automated Essay Grading,.* 2003.

[5] Burstein J. Attali, Y. Automated essay scoring with e-rater® v.2. *The Journal of Technology, Learning, and Assessment*, 2006.

[6] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013. URL http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf.

[7] Matthew Shardlow. *An analysis of feature selection techniques*. URL https://studentnet.cs.manchester.ac.uk/pgt/COMP61011/goodProjects/Shardlow.pdf.

[8] et al. Foltz, PeterW. *Implementation and applications of the Intelligent Essay Assessor*. Routledge Handbooks, 2013.

[9] Lucio Dery Huyen Nguyen. *Huyen Nguyen and Lucio Dery*. URL https://cs224d.stanford.edu/reports/huyenn.pdf.