AUTOMATED ESSAY GRADING

IST 664 / CIS 668: Natural Language Processing

Vijayalakshmi Girijala
Priyanka Bapat
Hrishikesh Gadkari
Jeet Patel

December 9, 2020

Table of Contents

Table of Contents **1**

Abstract **2**

1 Introduction **2**

2 Data **3**

2.1 Dataset Evaluation 3

2.2 Data Cleaning 3

3 Methodology **6**

3.1 Feature Selection 6

3.2 Feature Extraction 7

3.3 Feature Generation 8

3.4 Algorithms and Techniques 9

4 Implementation **10**

4.1 Single Feature Kappa 10

4.2 Greedy Forward Feature Selection 11

4.3 Refinement using Word2vec Model 12

5 Results **13**

5.1 Model Evaluation 13

5.2 Free Form Visualization 13

6 Conclusion **15**

6.1 Limitations 16

6.2 Future Work 16

7 References **16**

Abstract

This project aims to build an automated essay scoring system that can accurately match the scores given by human graders. Using a dataset of 12976 essays, we extracted features such as orthography, grammar, vocabulary complexity, sentence complexity, and part-of-speech count. We used a Linear Regression model to learn from these features and used 5-fold cross-validation to train and test our model rigorously. Additionally, we also used Forward Feature Selection and Word2vec model to further refine our results. We found that the combination of our heuristic features and word vectors generated by the Word2vec model provided the best score prediction, and our final model achieved a kappa score of 0.92.

1 Introduction

Essay writing is a foundational aspect of many primary and secondary school curriculums, where it is used as a tool to assess students' composition skills. Essays are also frequently used in standardized testing, where they are often the only portion of the examination that is scored by human graders, rather than by machines. As this is a time-consuming process, improvements in natural language technology have led to an increased interest in automated essay grading. Though automated grading systems have been employed in the past, the complexity of determining essay quality has limited widespread implementation. Essay grading is a complicated task, which must account for not only spelling, grammar, and punctuation, but also word choice and sentence flow. In addition, essays can be graded purely on content, on form and style, or on some combination of the two.

In January 2012, the William and Flora Hewlett Foundation hosted a research competition on Kaggle[1] that prompted "innovation for new solutions to automated student assessment." Their goal was to determine whether it was possible to create an automated grading system that could match the scores given by human graders. Three groups were able to create a model with a Weighted Mean Quadratic Weighted Kappa Score above 0.805, and were awarded a total of $100,000 when the competition closed in April 2012.

Later that year, three Stanford students[2] chose to attempt the task outside the bounds of the project, using the dataset provided by the Hewlett Foundation. Their model, published in December 2012, had a Quadratic Weighted Kappa score of 0.73, which would have placed them in the top 25 results in the Kaggle competition.

Inspired by the technological advances in the field of natural language processing over the last eight years, we chose to take on the same challenge. Though the ultimate goal was to create an automated grading system that perfectly matched the scores given by human graders, our primary goal was to create a model that beat the kappa score reached by Mahana et. al.

---

[1] The William and Flora Hewlett Foundation. *The Hewlett Foundation: Automated Essay Scoring*. January 19, 2012. Distributed by Kaggle. https://www.kaggle.com/c/asap-aes/data.

[2] Mahana, Manvi, Mishel Johns, and Ashwin Apte. Rep. *Automated Essay Grading Using Machine Learning*, December 14, 2012. http://cs229.stanford.edu/proj2012/MahanaJohnsApte-AutomatedEssayGradingUsingMachineLearning.pdf.

2 Data

As the basis for our project, we used the dataset obtained by the William and Flora Hewlett Foundation, which was made publicly available by their 2012 Kaggle competition. This dataset consists of eight different essay sets, generated from eight different prompts. The essays, written by students in grades 7-10, were manually graded and scored by at least two raters. The provided test dataset contains 4254 rows and only five columns, corresponding to the essay_id, essay_set, essay, domain1_predictionid, and domain2_predictionid. In contrast, the training dataset contains 12976 entries with 28 unique attributes, which encompass the five columns stated above while also including individual rater scores and rater trait scores, where provided.

2.1 Dataset Evaluation

Though we know that each essay set corresponds to a different essay prompt, the dataset provided on Kaggle contains neither the essay prompts nor the rubrics used by the human graders. As a result, we cannot take the expected essay length or writing style into account, or determine how either of those factors could impact the scores given by the human graders. We also know that the manual scores are inherently biased by virtue of being human-generated data - factors such as handwriting legibility, topic interest, and grader fatigue are difficult to quantify but potentially impactful.

Prior to providing this dataset to the public, the Hewlett Foundation[3] used Stanford's Named Entity Recognizer[4] to replace identifying information like names, dates, locations, and email addresses with anonymous placeholders such as @PERSON1. As this anonymization was applied somewhat haphazardly, the dataset contains sentences such as "To whom it @MONTH concern…".

2.2 Data Cleaning

As the test dataset contains only five columns - essay_ID, essay_set, essay, and the two domain scores - we chose to drop the eighteen rater trait columns and the five rater-specific domain score columns from our working version of the training dataset. In examining the two domain scores, we noticed that essay_set 2 was the only set that provided scores for domain2. As such, we chose to combine the two domain score columns to create a single resolved_score column, bringing our working training dataset to four columns:

[3] The Hewlett Foundation

[4] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pp. 363-370. http://nlp.stanford.edu/~manning/papers/gibbscrf3.pdf

```
working_train.head()
```

	essay_id	essay_set	essay	resolved_score
0	1	1	Dear local newspaper, I think effects computer...	8.0
1	2	1	Dear @CAPS1 @CAPS2, I believe that using compu...	9.0
2	3	1	Dear, @CAPS1 @CAPS2 @CAPS3 More and more peopl...	7.0
3	4	1	Dear Local Newspaper, @CAPS1 I have found that...	10.0
4	5	1	Dear @LOCATION1, I know having computers has a...	8.0

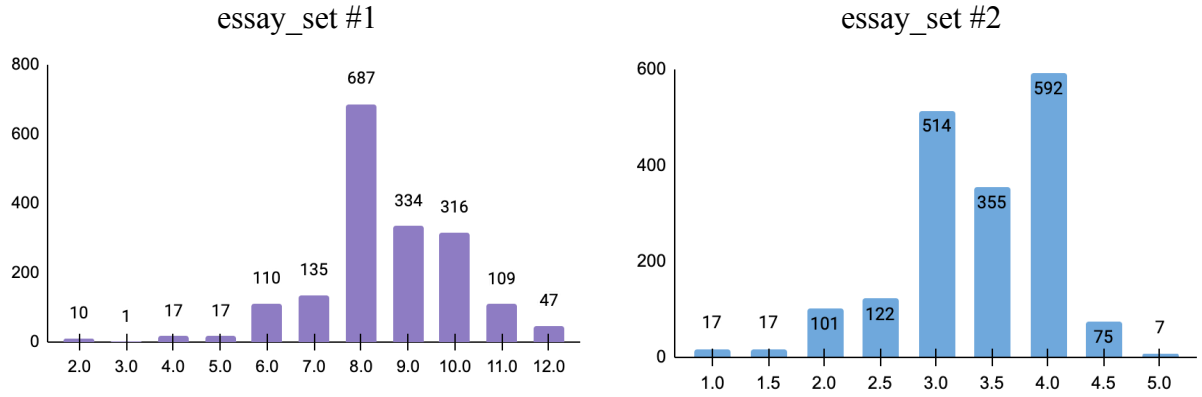
Figure 1: Sample Training Set

In examining the range of the resolved_score column, we were puzzled to see that while the range of essay scores spanned from 0.0 - 60.0, the vast majority were clustered between zero and five. To better understand this score distribution, we separated out the essay sets and plotted the score distribution for each. The resulting graphs showed that each essay set had been graded with a different rubric, with varying score ranges and grading increments. Though creating individual models for each essay set would produce a higher accuracy, we chose to create a single model to avoid overfitting our automated grading system to the training dataset.

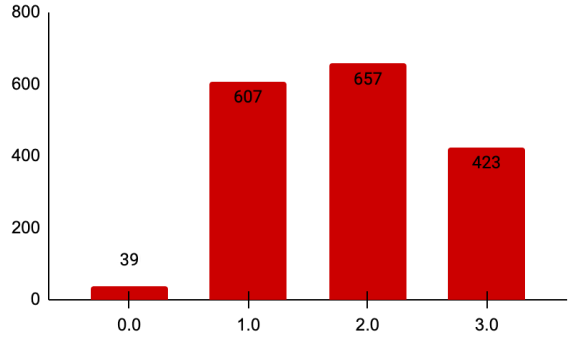
Table 1: Resolved Essay Scores By Essay Set

essay_set	1	2	3	4	5	6	7	8
score range	2.0 - 12.0	1.0 - 5.0	0.0 - 3.0	0.0 - 3.0	0.0 - 4.0	0.0 - 4.0	2.0 - 24.0	10.0 - 60.0
increments	1.0	0.5	1.0	1.0	1.0	1.0	1.0	1.0

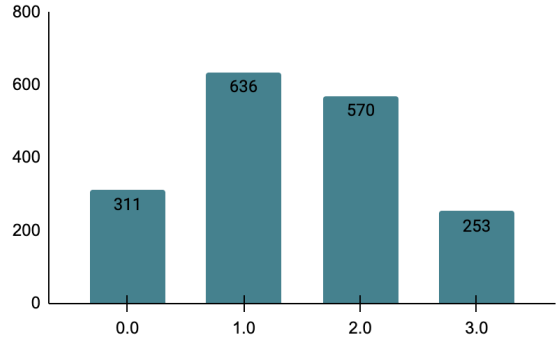
Fig 2: Resolved Essay Score Distributions By Essay Set



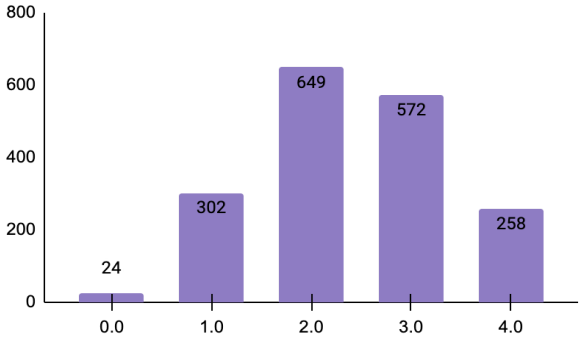
essay_set #3



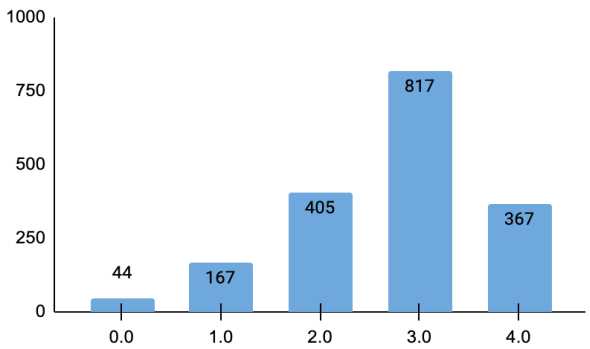
essay_set #4



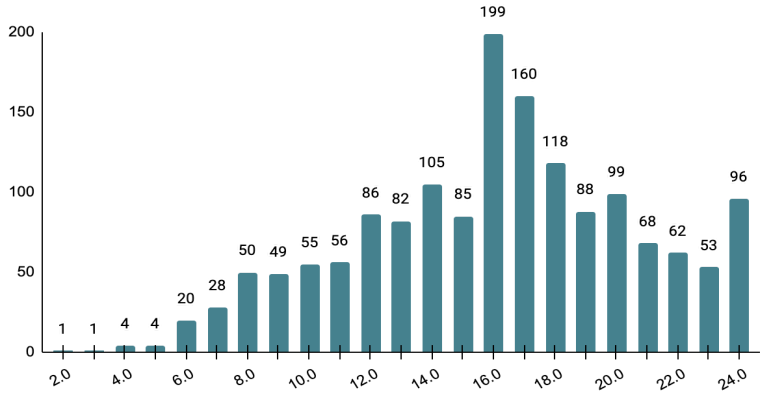
essay_set #5



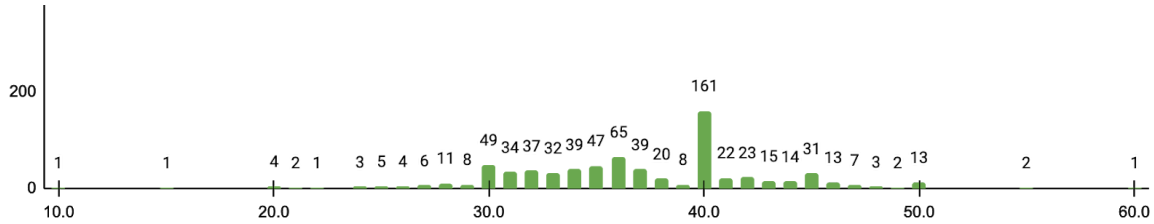
essay_set #6



essay_set #7



essay_set #8



The last step of our data cleaning measures was handling the essay text itself. We chose to make all of the text lowercase, change all the anonymized terms that had been substituted using the Stanford Named Entity Recognizer to the all-caps word 'ANON', and remove excess whitespace to make future word-splitting easier.

3 Methodology

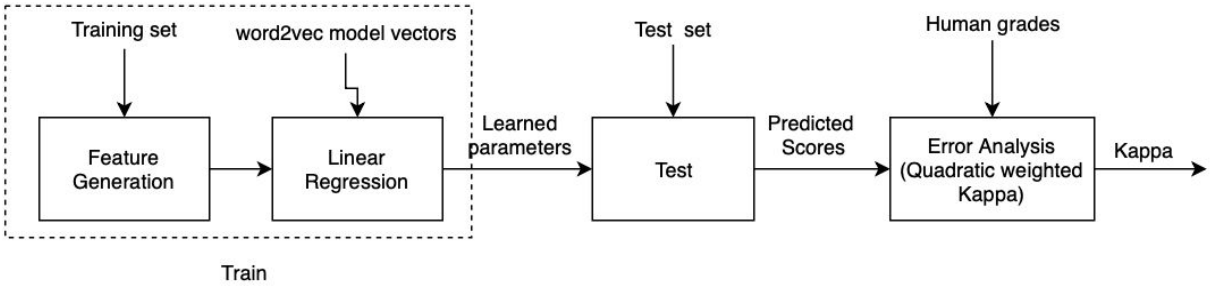


Figure 3: Implementation Methodology

Theoretically, the best way for an Automated Essay Scoring system to match the score given by human graders would be to utilize the same scoring metric. The SAT[5], a standardized test that is widely used for college admissions in the United States, grades the essay portion of the exam on a complex, multifaceted rubric. This scoring guide includes features such as reading comprehension, factual understanding of the subject material, insightful analysis, strategic selection of evidence, vocabulary precision, paragraph structure, sentence variety, and correctness of spelling and grammar. It is difficult to determine which metrics were used by the human graders that provided the scores for our dataset, as the lack of access to the prompts prevents us from grading the essays on content aspects such as comprehension or analysis. In addition, many essay features are fundamentally non-quantitative - grading vocabulary precision requires a deep understanding of word connotations. As such, we can only assume that fundamental aspects of essay writing, such as correct orthography and grammar and proper use of sentence and vocabulary complexity, had a significant impact on the scores assigned by the human graders. Each of these features must be incorporated into the model to achieve an accurate prediction.[6]

3.1 Feature Selection

After determining that our model would grade essays on the basis of orthography, grammar, sentence complexity, and vocabulary complexity, we had to choose how we would grade each of these aspects in a quantitative manner.

Mahana et. al.[7] chose similar features to train their model, grading orthography with a count of misspelled words, grammar and structure using punctuation based features, sentence complexity

[5] SAT Essay Scoring. (2018, March 12). Retrieved December 10, 2020, from https://collegereadiness.collegeboard.org/sat/scores/understanding-scores/essay
[6] Mahana et. al.
[7] Mahana et. al.

using heuristic data such as sentence count, total word count, and average word length per essay, and vocabulary complexity using part-of-speech counts and frequent word weighting.
With Mahana et. al's model as an example, we chose to discard the grammar feature, focusing instead on orthography, sentence complexity, and vocabulary complexity.

3.2 Feature Extraction

We extracted a total of seven features - spelling error percentage to represent orthography, average sentence length to represent sentence complexity, and average word length and part-of-speech tagging to represent vocabulary complexity.

Heuristic Features: Heuristic features contribute to a good essay. We generated several heuristic features like word count, long word count, average word length per essay, sentence count, etc. These features show language fluency and dexterity.

3.2.1 Orthography

Spelling errors are a significant reflection of an essay writer's grasp of the English language. In order to measure an essay's spelling error, we tokenized each essay using the NLTK's Whitespace Tokenizer, then compared it to the PyEnchant en_US dictionary. We divided the number of misspelled tokens by the total number of tokens in the essay to find the spelling error percentage.

3.2.2 Sentence Complexity

We used the average sentence length - the total word count divided by the total sentence count - as a measure of each essay's sentence complexity. While we could have followed Mahana et. al.'s example and used sentence count, we realized that the variety in essay type leads to vastly different sentence counts between the essay sets. Essay sets 3-5 contained essays that were only a few sentences long, while other essay sets contained multi-paragraph responses.

3.2.3 Vocabulary Complexity

We used two different methods to measure vocabulary complexity - average word length, and part-of-speech distribution. To measure the average word length, we used the total character count (not including whitespace or punctuation) divided by the total word count. We considered this to be a better measure of vocabulary complexity than of sentence complexity, as English academic language generally skews towards longer words, making a high average word length indicative of a more advanced vocabulary.
To measure the part-of-speech distribution, we decided to tokenize the essay text using NLTK's Regular Expression or regex tokenizer, to split the essays into a list of words. Next, we removed the stopwords using the NLTK's stopwords library as they are irrelevant for the analysis. We finally performed POS tagging using NLTK's part-of-speech library.
Part of Speech (POS) tags: Count of nouns, verbs, adverbs, and adjectives help to identify good sentence structure.

3.3 Feature Generation

We extracted these features from the given essays and assigned the value/count to the respective column. We generated the required features using the *GenerateFeatures()* function.

```
In [122]: X_all[:20]
Out[122]:
```

	essay	word_count	long_word_count	avg_word_length_per_essay	noun_count	verb_count	adjective_count	adverb_count
0	[dear, local, newspaper, think, effects, compu...	158	53	5.79	68	35	30	11
1	[dear, believe, using, computers, benefit, us,...	216	77	5.67	96	54	23	10
2	[dear, people, use, computers, everyone, agree...	132	52	6.05	72	36	17	2
3	[dear, local, newspaper, found, many, experts,...	259	109	6.25	123	56	40	13
4	[dear, know, computers, positive, effect, peop...	221	82	5.97	111	46	25	14
5	[dear, think, computers, negative, affect, us,...	101	35	5.62	41	26	18	9
6	[know, people, days, depending, computers, saf...	251	89	5.9	117	53	31	26
7	[people, agree, computers, make, life, less, c...	243	80	5.9	114	62	42	15
8	[dear, reader, dramatic, effect, human, life, ...	216	67	5.71	107	43	36	17
9	[technology, computer, say, computers, good, s...	228	54	5.43	106	52	40	19
10	[dear, people, acknowledge, great, advances, c...	175	72	6.38	83	36	27	17

Figure 4: Generated Feature Columns

3.3.1 Feature Generation Issues

Though we had originally planned on using an extant English grammar to check the grammar of each sentence, we ran into various technical difficulties when attempting to access their libraries. Most were older versions, with no update for the Python 3 version that we were using. We finally tried to implement the Grammar-check tool and upon running the library, the code threw an error for processing large datasets. We decided not to use correct grammar as a feature, focusing instead on features like vocabulary complexity and sentence complexity.

3.3.2 Splitting the data

After generating the features, they were stored in a CSV file so that other notebooks can directly import the features. The resultant dataset was first split into input(X_all) and output(y_all) columns. X_all consists of all the feature columns while y_all includes the target data. We then passed both arrays for further analysis and split the data appropriately into train and test subsets.

3.4 Algorithms and Techniques

3.4.1 Linear Regression

An overview of related prior work indicates that linear Regression works well for essay grading applications[8][9][10]. For this project, we evaluated a simple Linear Regression model using the equation

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

Here, the output vector y is generated based on features x extracted from essays. Given an input feature vector $x \in R^m$, an output vector $\hat{y} \in R$ using a linear model with a weight of β.

3.4.2 Word2vec Model

Gensim's word2vec model is a widely used algorithm that is based on neural networks and deep learning. It uses a large amount of unannotated text and learns relationships between words automatically. It generates an output of word vectors for each unique word with linear relationships in the training set. We will combine this vector with our heuristic features to create a final vector for each word in our train set.

3.4.3 Forward Feature Selection

The heuristic features that we generated during the feature extraction process may not entirely contribute positively to the essay grading. Thus, to eliminate features that might perform poorly and to ensure that the features assist positively for the learning model, we decided to use the greedy forward feature algorithm.

The forward feature selection makes changes to a set of features and keeps the new set for further analysis in order to increase accuracy. The greedy approach starts with one feature and keeps adding on other features incrementally. It will evaluate the model with the features and will keep the feature only if it makes a positive change in accuracy and in turn choose the most useful features for our learning model.

3.4.4 Benchmark model

We are using Quadratic Weighted Kappa as our evaluation metric. It is a statistical metric that measures the inter-rater agreement for qualitative items. This metric will allow us to compare our results with our benchmark scores. 0 represents random agreement between the raters and 1 is full agreement. Thus the quadratic weight Kappa score ranging from 0(random agreement) to

---

[8] Mahana et. al.

[9] Valenti, S., F. Neri and A. Cucchiarelli. "*An Overview of Current Research on Automated Essay Grading*." J. Inf. Technol. Educ. 2 (2003): 319-330.

[10] Attali, Yigal & Burstein, Jill. (2006). Attali, Y., & Burstein, J. (2006). *Automated essay scoring with e-rater*® V.2. Journal of Technology, Learning, and Assessment, 4(3).. Journal of Technology, Learning, and Assessment. 4.

1(full agreement). Mahana et al[11] achieved a kappa score of 0.73. We are hoping to exceed the score with our model.

4 Implementation

4.1 Single Feature Kappa

We evaluated each heuristic feature individually to check their performance for the learning model. We created the function *Evaluate()* that uses Linear Regression as the learning model with 5-fold cross-validation. The quadratic weighted kappa was used as an evaluation error metric to guard against overfitting. We obtained the following single feature kappa values for our heuristic features.

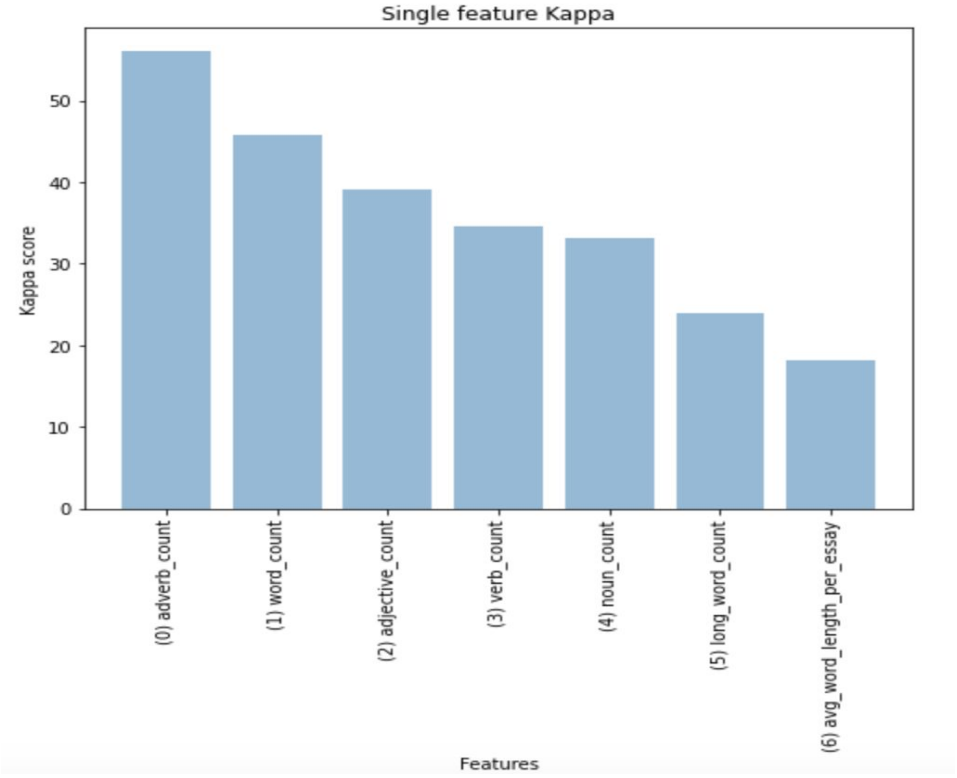


Figure 5: Single Feature Kappa Scores

4.2 Greedy Forward Feature Selection

We sorted the features in decreasing order so that a feature is selected only if the addition of the feature increases the accuracy higher than the accuracy of the best performing individual feature. After applying the forward feature selection algorithm, we obtained the highest performing feature set[1..7] with a kappa score of 0.76. This score was above our set benchmark of 0.73. Thus we can say that almost all our heuristic features assisted in increasing the Kappa score for our learning model.

[11] Mahana et. al.

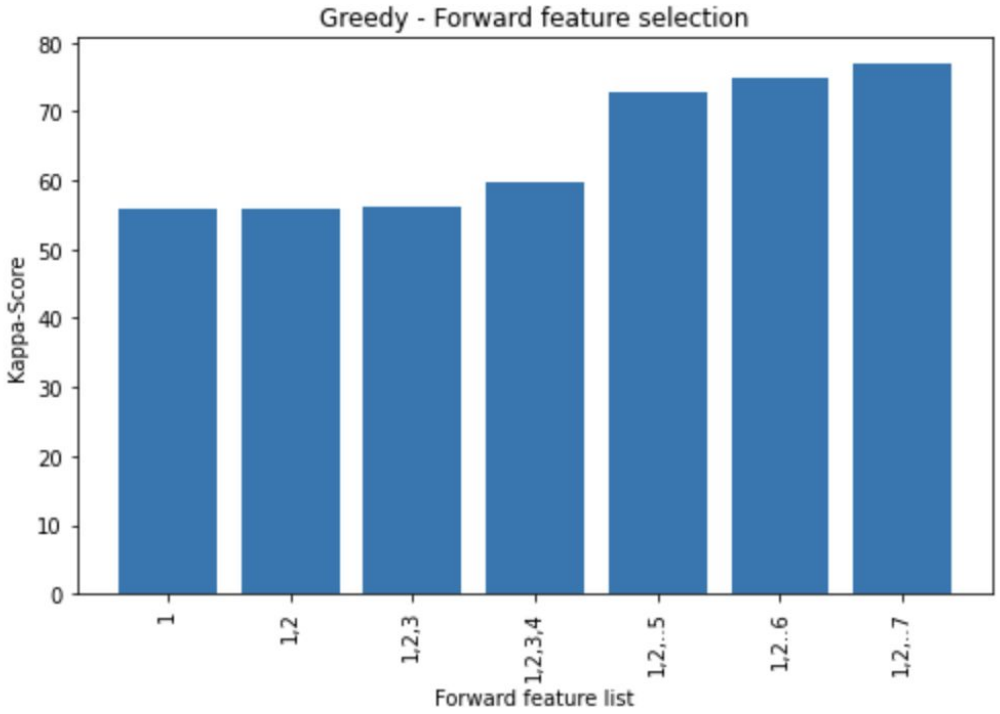


Figure 6: Forward Feature Kappa Score

4.3 Refinement using Word2vec Model

In order to increase the accuracy of the model, we decided to try implementing the word2vec model from the Gensim library. We merged the generated word vectors with our heuristic feature vectors. We used the following library functions:

1. *essay_to_sentences():* Extract the sentences from all the essays in the training set. Sentence tokenized the essays.
2. *essay_to_wordlist():* Removed the tag labels and word tokenized each word in the sentences.
3. *makeFeatureVec():* This is a helper function that generates word vectors and generates a 300 dimension word vector.
4. *getAvgFeatureVecs():* Generated the training and the testing vectors which were then combined with the heuristic features thus generating the main word vectors for the word2vec model.

Dummy Regressor

We ran our word2vec model against Sci-kit Learn's dummy regressor to check whether the refined model actually learns something. We observed a Kappa value of 0.0

Linear Regression

Since the dummy regressor indicated no learning, we then trained the Linear Regression model using a 5 fold cross-validation with the combined vectors of the word2vec model and our heuristic feature vectors. We initialized the learning model, fit the data, and predicted the results

for the testing data by evaluating the model on the evaluation metric, Quadratic Weighted Kappa. We obtained an average Kappa score of **0.92**.

5 Results

5.1 Model Evaluation

Model	Kappa Score
Dummy Regressor	0.0
Mahana et. al.	0.73
Linear Regression with Heuristic Features	0.76
Linear Regression with word2vec model and heuristic features	0.92

Table 2: Results

The use of the forward feature algorithm in our project ensured that the heuristic features contributed to the model positively. As we can see, using Linear Regression with forward feature on our heuristic features alone resulted in a Kappa score of 0.76 which is already higher than the score achieved by Mahana et. al.

The 5-fold cross-validation allowed the model to learn different sentence structures, most commonly used POS tags, and generate word vectors for the various words used in the essays. Running the model against our training and testing process multiple times ensured that our learning model tested all possible inputs in the data set.

The heuristic features we generated are general necessary features for a good essay. These features can be used for essays from any domain. The final model consisting of word vectors generated from the word2vec model and heuristic features performed the best out of all the models. The quadratic weighted Kappa score improved drastically by 0.16 when the model was trained with word2vec vectors and heuristic features as compared to, with just heuristic features. The final Kappa score of 0.92 surpassed our set benchmark of Mahana's Kappa score of 0.73 thus confirming the usefulness of our model with heuristic features combined with the word2vec model.

We believe, with current research on NLP techniques, the adaptability of the word2vec model ensures that the learning model after training on any data set can be used for similar essay grading projects.

5.2 Free Form Visualization

We performed a Principal Component Analysis(PCA) for visualizing good essays (grade>8) and bad essays(grade<4). We extracted 100 good and bad essays from the dataset and obtained the following PCA visualizations.

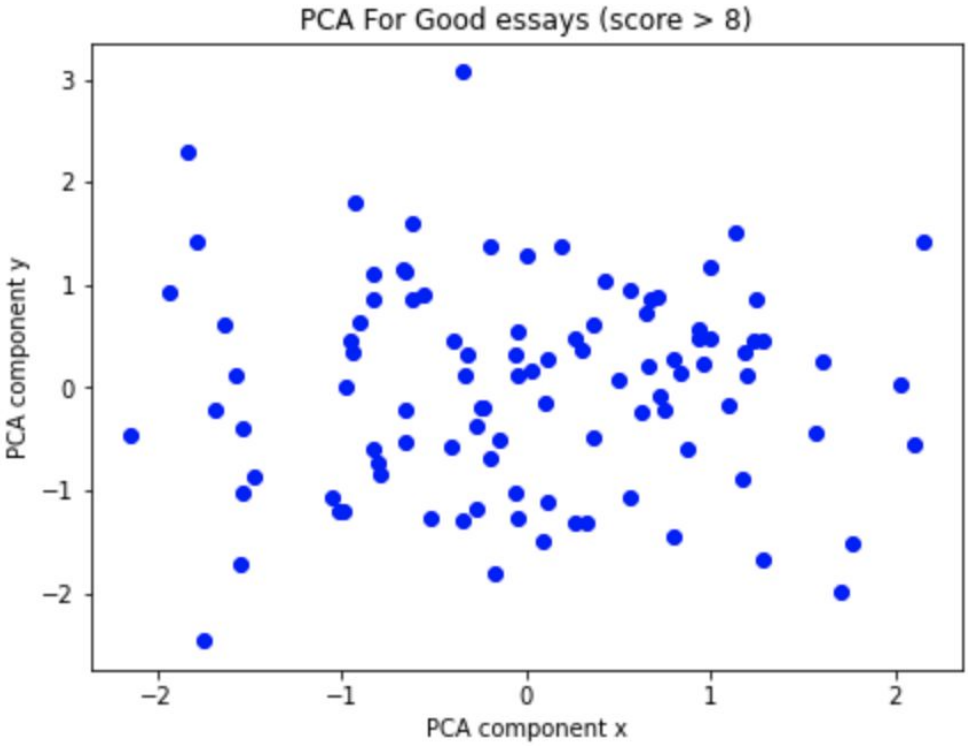


Figure 7: PCA Visualizations for Good Essays

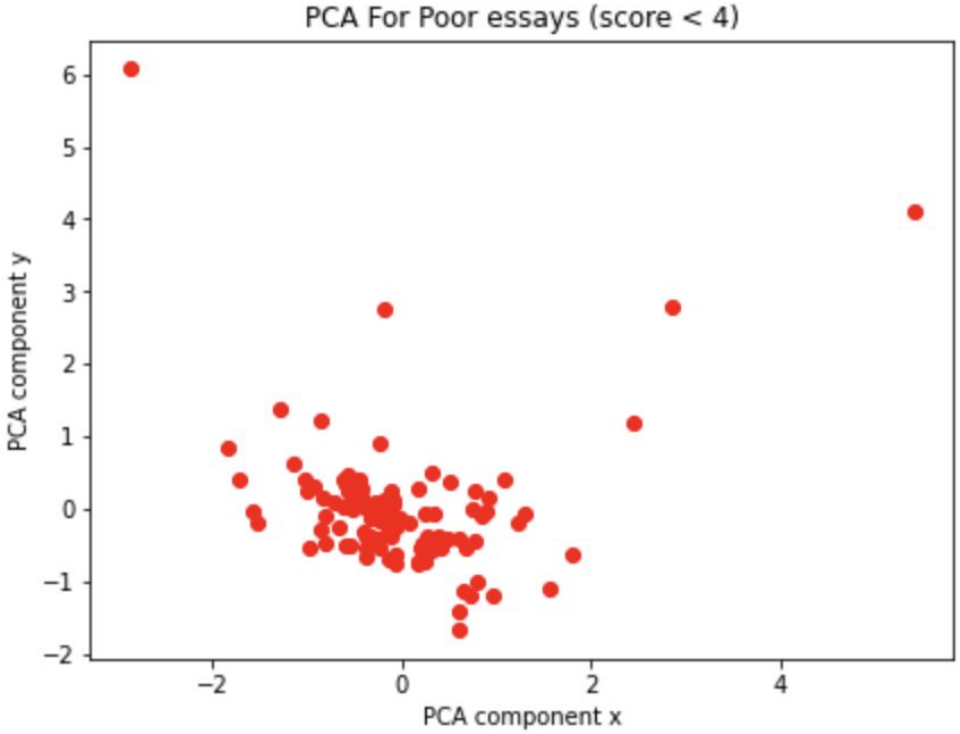


Figure 8: PCA Visualization for Poor Essays

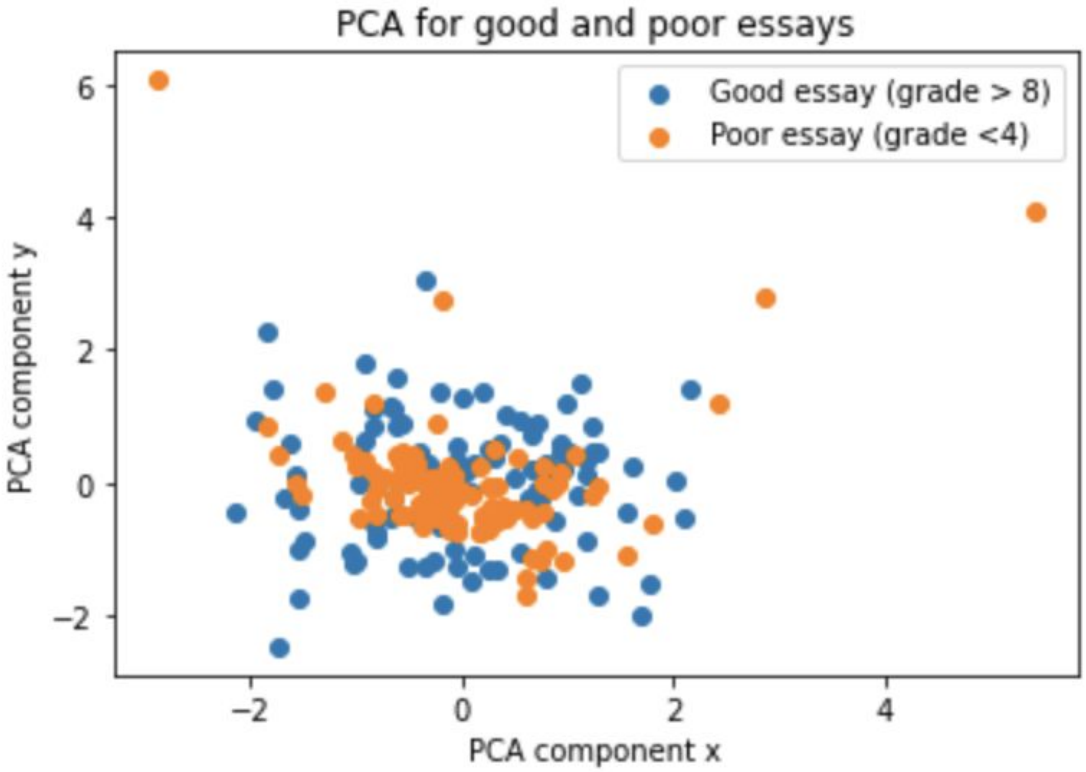


Figure 9: Combined PCA Visualizations

As we can see in Figure 6, with the exception of outliers, the poor essays are confined to a region and are highly dense while the good essays are spread over the graph. What we can infer from this is that students with poor grades make mistakes that are similar in nature. But for a student to achieve a good grade for their essay, several components are considered like the complexity of words or essay length. Essays with poor grades likely followed similar patterns like using less complex words or limited use of vocabulary. Our model somewhat captured this and we visualized it using PCA as shown above.

6 Conclusion

The goal of the project was to find a reliable model to accurately predict grades and thus provide an efficient automated approach to essay scoring. For our project, we decided to use a simple model of Linear Regression. We wanted to use general features that focus on sentence structure, similarity, and complexity of words used in the essays. Our model achieved an end result of high accuracy with a kappa score of 0.92 surpassing Mahana et al[12] kappa score of 0.73 which was set as our benchmark.

[12] Mahana et. al.

6.1 Limitations

While our project achieved a certain success by obtaining a good Kappa score, it is not without pitfalls. Most of our features are based on the structure and complexity of essay writing which for the purpose of our project turned out to be very useful. However, judging a paper by the features we used alone would be unfair. Most human graders also look at the writing style of the essays. Features like maturity in writing, emotive effectiveness, imagery, and meaningfulness would help in achieving a more human-like grade prediction.

However, using simple features, our model performed better than the benchmark set for this model. Combining heuristic features along with the word2vec model generated word vectors to train the learning model proved to be highly advantageous and thus, confirming the usefulness of using the word2vec model for NLP tasks.

6.2 Future Work

The scope for this project would be to follow an approach that does not require any feature engineering and automatically learns the parameters required for the task. Utilization of various and new Deep Learning and Neural Network techniques have great potential in solving NLP problems such as automated essay grading.

Additionally, as mentioned in the previous section, we hope to improve our model and our prediction scores by adding other features that indicate writing style like maturity and emotive effectiveness and thus, bringing our predictive scores closer to the human grades.

7 References

1. Mahana, Manvi, Mishel Johns, and Ashwin Apte. Rep. *Automated Essay Grading Using Machine Learning*, December 14, 2012. http://cs229.stanford.edu/proj2012/MahanaJohnsApte-AutomatedEssayGradingUsingMachineLearning.pdf.
2. The William and Flora Hewlett Foundation. *The Hewlett Foundation: Automated Essay Scoring*. January 19, 2012. Distributed by Kaggle. https://www.kaggle.com/c/asap-aes/data.
3. Valenti, S., F. Neri and A. Cucchiarelli. "*An Overview of Current Research on Automated Essay Grading*." J. Inf. Technol. Educ. 2 (2003): 319-330.
4. Attali, Yigal & Burstein, Jill. (2006). Attali, Y., & Burstein, J. (2006). *Automated essay scoring with e-rater*® V.2. Journal of Technology, Learning, and Assessment, 4(3).. Journal of Technology, Learning, and Assessment. 4.
5. Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. *Proceedings of the 43nd Annual Meeting of the Association for Computational*

Linguistics (ACL 2005), pp. 363-370. http://nlp.stanford.edu/~manning/papers/gibbscrf3.pdf

6. Nguyen, Huyen, and Lucio Dery. "*Neural networks for automated essay grading*." (2018).
7. SAT Essay Scoring. (2018, March 12). Retrieved December 10, 2020, from https://collegereadiness.collegeboard.org/sat/scores/understanding-scores/essay