ChatGPT Detection in Essay Authorship: Stylometric Analysis

s2152592, s2279407, & s2277508

November 2024

Executive Summary

ChatGPT, an AI chatbot, has increasingly been requested to write texts for its users, which has particularly become an issue with academics using ChatGPT to author their essays. To combat this issue, we want to determine if stylometry can identify AI chatbot, specifically ChatGPT, generated texts from human-authored essays. First, it was investigated whether there is any trend in ChatGPT authorship resembling human writing by essay topic. Then, the difference between training our 4 classification methods (K-Nearest Neighbours, Discriminant Analysis, Random Forests, and Support Vector Machines) using topic-specific data compared to our whole dataset was studied. Using Leave-one-out cross-validation, the accuracies of the different stylometric methods were tested to investigate the importance of different aspects of our data sets on this determination specifically regarding: essay length, and reducing the number of function words.

It was firstly found that ChatGPT may write society-related essays most similarly to a human, however, it was determined the difference was large enough to not pay extra care to these topics in the following analysis. Regarding training our methods using the topic-specific dataset compared to the whole dataset, it was found that the former method performed worse, especially Random Forests and K-Nearest Neighbours (KNN). However, as training using the topic-specific data was a lot computationally faster, this method was used for the rest of the research. Although our results may have been limited by the training data, significant evidence suggesting that shortening essay length has an adverse effect on determination accuracy for all methods, with the worst impact on random forests, was still found. Additionally, when reducing function words, the function words with the highest coefficient of variation had the highest (negative) impact on the determination accuracy of KNN and discriminant analysis when removed. Only these two methods were used as they performed best in the essay-length analysis. Removing words with the highest standard deviation was also found to have a similarly strong impact on the methodical accuracy when removed. Removing any function words did reduce accuracy on average, especially for KNN, therefore having a large bank of function words, especially those with high coefficient of variations or standard deviations, would most likely give you the most information on each essay, hence the best determination. Overall, though we found that all methods performed well in determining ChatGPT authorship when using the whole unaltered data set, DA was more resistant to decreases in information of modifying the dataset and would be recommended for usage on smaller datasets.

The recommendations for improving this report's analysis included expanding the dataset (e.g. using different text types, having more essays on different topics, and sourcing the AI-generated essays using different prompts to reflect real user behaviour) and bettering the stylometric analysis (e.g. using the whole data set for training, considering different stylometry methods like lexical features, using more statistical methods, and finding more about the function words' relationships to go more in-depth about their impact on the accuracy). Future research on this topic should consider studying different AIs, and also considering how ChatGPT has improved throughout different versions to see what areas may be worth focusing on for detection.

1 Introduction

Launched in 2022, ChatGPT is a generative artificial intelligence (AI) chatbot based on a large language model (LLM) capable of using its large data set to predict possible combinations of words to write answers to its users' questions. Trained on a large variety of texts, such as books, essays, and blogs, ChatGPT has become an increasingly popular tool for aiding with a myriad of tasks such as writing emails, language translation, and coding. While many organisations have found benefits in using artificial intelligence with increased work speed and decreased workload, many are also concerned about the increase in the tool's usage due to plagiarism, misinformation, and copyright issues [4]. To investigate whether stylometry techniques can be used to determine whether the author of an essay is human or AI, specifically ChatGPT, we will analyse the occurrence of 70 chosen common words to categorise the human and AI writing styles. We will use Leave-One-Out Cross-Validation (LOOCV) to find the accuracies of authorship determination for each of the following four different statistical methods: K-nearest neighbours (KNN), discriminant analysis (DA), random forests (RF), and Support Vector Machines (SVM). We will find the effects of training our methods using the dataset where the essays are all grouped together by their topic compared to the essays being individual. Another topic of investigation will be whether the accuracy of authorship determination is genre-specific, i.e. whether stylometry is more significantly more, or less, accurate for specific topics or genre types. Additionally, we will explore whether text length affects the accuracy of determining the author type of an essay and the effects of not including specific function words in our analysis.

2 Dataset Overview

2.1 Description of our Dataset

To perform our stylometry, we analysed the pattern of usage of common words in a text, where the authorship determination is done by considering each author's style as the pattern of word occurrence. We first started by choosing 70 function words to perform our stylometric analysis. These 70 function words were chosen based on two factors: their prevalence in the English language's usage, and being words that are used context-free, such as to not create a bias in our dataset from certain words having a higher occurrence due to a text's topic rather than a text's author's style.

Our corpus consisted of 2000 human-written essays of varied lengths from the Aeon.co [1] and 1000 word long ChatGPT essays written about the title of each human essay. Although there are 1500 authors, we treated our dataset as if there is only one author for all 2000 essays to perform our analysis. The essays were written about 110 largely varied topics such as Politics, Physics, Sports, etc. The AI essays were generated using GPT4o-mini based on the following prompt: "Hi chatgpt I am going to give you a title for an essay, and I would like you to write a 1000 word essay on this subject." Our dataset consisted of 71-entry long vectors for each essay, which we refer to as our function vectors, where the first 70 entries represented the count of each of the function words used in the text and the last entry was the total of all the other remaining words.

As the real distances between the 71 dimensional function vectors representing the authors/essays do not scale down appropriately to 2-D, we will use Multi-Dimensional Scaling (MDS), which minimises the graphing, for accurate visualisation of these vectors.

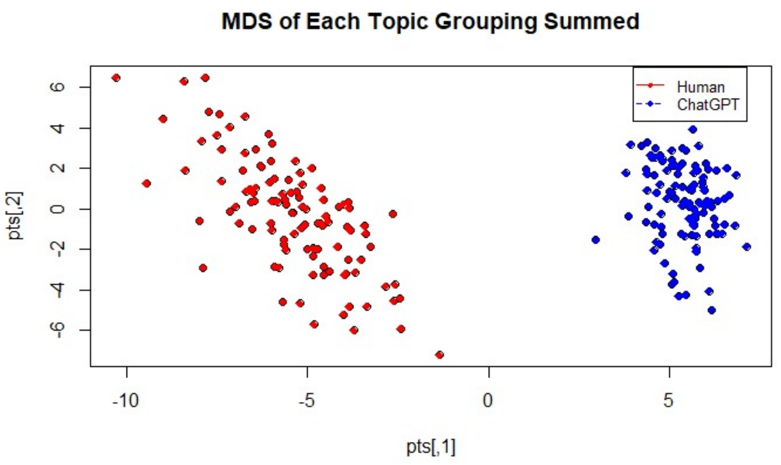

Figure 1: MDS plot of the 110 topics

2.2 Trends in Topic type

Although we found that the human authored essays were quite separate from the ChatGPT essays, clear in Figure 1, we wanted to determine whether specific types of topics are written more human-like by ChatGPT, which could indicate the need for extra care regarding these topics. To do this, we found the 10 closest and 10 furthest topics in terms of the Euclidean distance from the human-written essays regarding each topic to the ChatGPT-written essays on the same topic. The farthest topic pairs seemed to have no relation (e.g. Engineering, Spirituality, and Home Making seemed unassociated), thus we determined that it seemed most likely that there was no relation between the pairings. The topics we found where ChatGPT and human essays were most similar, twice closer to each other on average compared to the farthest group, were: Religion, Nations and Empires, History, Global History, Subcultures, History of Technology, War and Peace, Politics and Government, Economics, and Economic History. There seemed to be a relation between these topics, with all of them relating closely to culture, either regarding history, economics, or culture. We believe this may have been caused by ChatGPT having more content relating to society in its training dataset, especially since throughout time, many people have documented history. Extra care may be needed in the authorship determination of essays relating to similar topics due to the larger similarity in ChatGPT and human styles. However, as these are not all society-related topics, and the ChatGPT essays were still quite different from the human ones, this theory is not conclusive.

3 Breakdown of Classification Methods

In this report, we use these four different statistical tools to analyse our dataset stylometrically: Discriminant Analysis, K Nearest Neighbours, Random Forests, & Support Vector Machines. These are classification methods to best attribute a class to an essay of unknown classification, this corresponds to one of our pairings (x_i, y_i) where x_i is the function vector for the i^{th} essay grouping and y_i is the associated binary classification of whether the author is ChatGPT or that author is not ChatGPT for the i^{th} essay grouping. The function vector for an essay grouping is the sum of some chosen group of essay's function vectors.

For the purpose of explaining our statistical methods, these i groupings are arbitrary. However, in the context of this report, they can be grouped by each essay, by each topic, or by any other statistically relevant grouping. With the initial groundwork laid, we can now begin the discussion of the statistical methods employed in this report.

Discriminant Analysis

The aim of each of our statistical methods in this report is to predict if a given grouping of essays was written by ChatGPT or written by a human. Discriminant analysis approaches this problem by calculating the probability that a grouping of essays is or is not written by ChatGPT, then classifying our essay group according to the higher probability.

We do this by using Bayes Theorem $p(y_i = q \mid x_i) \propto p(x_i \mid y_i = q)p(y_i = q)$, where each x_i is an essay group of unknown ChatGPT authorship classification containing N_i words and we have classes $q = -1$ and $q = 1$ corresponding to the classifications "not written by ChatGPT" and "written by ChatGPT" respectively. Our class conditional densities are given by the multinomial $p(x_i|y_i = q) = \text{Multinomial}(x_i; N_i, \theta_q)$.

Our parameter vector is given by $\theta_q = (\theta_{1,q}, \theta_{2,q}, \ldots, \theta_{71,q})$, where $\theta_{i,q}$ is the probability of a random word from the class q being the word corresponding to the k^{th} entry of the function vector. Given, for class q, that $k_{i,q}$ is the total word count for the corresponding i^{th} entry of the function vector and that N_q is the total word count for the given classification, we can estimate our parameter by using the Maximum Likelihood Estimator, giving us $\hat{\theta}_{i,q} = \frac{k_{i,q}}{N_q}$.

If we take that all essay groupings are equally likely to have been "written by ChatGPT" or "not written by ChatGPT", $q = 1$ or $q = -1$, we can take our Bayesian prior to be $p(y_i = q) = \frac{1}{2}$, $\forall q \in \{-1, 1\}$. With this, we have all the constituent components required to calculate our proportional probabilities given by Bayes Theorem as above. Thus, for a given x_i essay grouping of unknown ChatGPT authorship, we calculate the class conditional density for each q and multiply by the corresponding prior, that being $\frac{1}{2}$ in this case. These two probabilities do not sum to 1 as they are supposed to, so we rescale them appropriately such that they do. Then finally, we assign the classification with the largest probability to our given essay grouping.

K-Nearest Neighbours

This technique uses less statistically involved mathematics, but it is still a surprisingly accurate technique nonetheless. This classification technique calculates the K closest distances from our $\tilde{x}$ essay grouping's function vector of unknown ChatGPT authorship to all other essay grouping's function vectors in the (x_i, y_i) pairings training set for each $y_i = -1$ & $y_i = 1$. Subsequently, our $\tilde{x}$ is assigned class as the majority classification y_i of these K closest x_i vectors, in the case of a tie either one is randomly chosen. More rigorously, we define the previous as follows:

Given a distance function $d(x, y)$ for two vectors x & y, we define $d_i = d(x_i, \tilde{x})$. For N different (x_i, y_i) pairings, we define the set $\mathcal{D} = \{d_1, d_2, \ldots, d_N\}$. If we let the least element of $\mathcal{D}$ be δ_1, then let the least element of $\mathcal{D} \setminus \{\delta_1\}$ be δ_2, then let the least element of $\mathcal{D} \setminus \{\delta_1, \delta_2\}$ be δ_3 and so on, we can take the a set $\Delta_K = \{\delta_1, \delta_2, \ldots, \delta_K\}$ = which contains only the K closest distances to our $\tilde{x}$ as each δ_j corresponds to some d_i. Taking a set containing the indices of the smallest distances, $I = \{i : d_i \in \Delta_K\}$, we can then define our classification choice for $\tilde{y}$ as below,

$$\tilde{y} = \begin{cases} -1, & \text{if } -1 \le \frac{\sum_{i \in I} y_i}{K} < 0 \\ 1, & \text{if } 0 < \frac{\sum_{i \in I} y_i}{K} \le 1 \\ -1 \text{ or } 1 \text{ randomly}, & \text{if } \frac{\sum_{i \in I} y_i}{K} = 0 \end{cases}$$

In either level of mathematical complexity, we take our distance function d to be the Euclidean distance, the L_2 norm. For two K-dimensional vectors α & β, it is defined as:

$$d(\alpha, \beta) = ||\alpha - \beta||_2 = \left(\sum_{j=1}^{K} (\alpha_j - \beta_j)^2 \right)^{\frac{1}{2}}$$

Random Forests

The way in which this technique assigns classification to a given essay grouping is similar to KNN in the sense that we decide classification based on a majority output class of lots of differently fitted decision trees, constructed through algorithms discussed in this section. The $C4.5$ algorithm, which constructs the decision trees, and the (modified) bagging algorithm, creating lots of "pseudo-datasets" which we can train decision trees on. Below we explain how each of these algorithms function and their contribution to the whole technique:

Decision Trees: A decision tree is basically a flow chart where a feature is considered at each node and at the end of this tree, we reach one of the binary classes. To construct the decision tree for a training set, we employ the $C4.5$ algorithm, which we now set out to explain. We have our normalised training set, the pairings (x_i, y_i) where we normalise each x_i, and at each node of the tree the $C4.5$ algorithm chooses the feature that most effectively splits the remaining parings into subsets which predominantly contain one of the binary classes. The criteria to decide which feature to split on is to maximise the information gain, which is defined as the decrease in entropy, a measure of uncertainty, as a result of splitting on a some feature.

Intuitively speaking, in so maximising the decrease in entropy by choice of feature to split on, we are choosing the feature that best reduces uncertainty about the target class, allowing for more reliable predictions. For example, imagine predicting if someone will buy a product based on age and income. If splitting by income creates groups where one mostly contains buyers and the other mostly non-buyers, this reduces uncertainty about who will buy. By repeatedly choosing the most informative splits, i.e the maximum information gain, the algorithm builds a tree that predicts outcomes by asking the most relevant questions.

Decision trees on their own are not usually a suitable classification method due to their instability, small changes in the training data can produce very different trees, and thus high variance. This can be mitigated to produce an accurate classification method through use of the algorithm in the next section.

Bagging: To start, we introduce a statistical technique which allows us to create additional copies of our training data set using resampling called bootstrapping. So, considering some N number of (x_i, y_i) pairings, we create a new dataset by choosing N of these pairings *with replacement*; this new dataset is a bootstrapped dataset. Carrying out this process a vast array of times, we would have $M \gg 1$ bootstrapped datasets.

We introduce bootstrapping as a concept because bagging is the name given to the general bootstrap aggregation algorithm for some given classification technique. We proceed with bagging as follows:

- Create M bootstrap training sets by applying the bootstrap to the original training set
- For each j^{th} bootstrapped training set, train a classifier using only this data
- Given a new observation $\tilde{x}$, let $f^j(\tilde{x})$ be the prediction from the j^{th} bootstrapped training set's classifier
- Our final classification decision for $\tilde{x}$ is the majorative binary classification from $f^j(\tilde{x})\ \forall j$.

The main reason bagging is effective is that, although each individual classifier trained on a subset of the data may perform worse than a single classifier trained on the entire dataset, the combined average performance of the classifiers trained on each bootstrapped training set tends to improve due to a reduction in variance.

The above two algorithms come into play when considering their application to the random forests classification technique as described initially. Bagging works particularly well with decision trees because they are a classifier with high variance and thus each bootstrap can produce a much different decision tree, giving us large variety in classifiers; thus great reduction in variance.

In order to further reduce the overall variance, we want to further increase variety in trained classifier. In the typical bagging algorithm, the trees that we learn may be relatively similar due to the fact that there is significant overlap in all of the bootstrapped datasets. If however, for each bootstrapped dataset, we take only some P features to incorporate from the original K features dataset, conventionally $P = \sqrt{K}$, the decision trees we learn on each bootstrapped dataset will be very different, further increasing variance.

This algorithm, using decision trees and the modified version of bagging, is known as Random Forests. We will now outline the steps for it below:

- Create M bootstrap training sets by applying the bootstrap to the original training set
- For each bootstrapped training set, choose P features at random and discard all others from that set
- For each such j^{th} bootstrapped training set, train a classifier using only this data
- Given a new observation $\tilde{x}$, let $f^j(\tilde{x})$ be the prediction from the j^{th} bootstrapped training set's classifier
- Our final classification decision for $\tilde{x}$ is the majoritive binary classification from $f^j(\tilde{x})\ \forall j$.

Support Vector Machines

The key intuition behind SVMs is that, given our data is binary classified, we can find the best possible decision boundary with which we can assign such classification to our data, with function vector $x_i = (x_{i,1}, \ldots, x_{i,K})$ and binary classification $y_i = -1$ or $y_i = 1$ as before. Support vector machines has a few different approaches to tackling this problem to correctly define such a boundary, which we will go through for each such case below.

Linearly Separable Data: If the data is linearly separable, that means that there exists a hyperplane, corresponding to a line in 2D and a plane in 3D and a hyperplane for higher dimensions, that perfectly separates the data between the two classes. We define this hyperplane through the following optimisation problem:

$$\begin{aligned} &\text{Maximise } M \\ &\text{subject to } \sum_{k=0}^{K} \theta_k^2 = 1 \\ &\qquad y_i(\theta_0 + \theta_1 x_{i,1} + \ldots + \theta_K x_{i,K}) \geq M \quad \forall i \end{aligned}$$

The techniques required to solve this optimization problem are slightly too tedious to include in this report. However, it boils down to an equivalent problem given below:

$$\max_{\lambda_i \geq 0} \left[\sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j (\mathbf{x}_i^T \mathbf{x}_j) \right]$$

where the λ parameters are functions of Lagrange multipliers and the θs. Most importantly, the above only needs to consider our data's function vectors with regards to the given inner product. We care about this because, to solve the optimization problem and determine the optimal hyperplane that maximizes the margin on the training set, it is sufficient to compute the inner product (dot product) between the observations in the training set, which is much easier to compute than the initial problem.

Non-Linearly Separable Data In some real-world scenarios, the data may not be perfectly separable. A soft-margin SVM introduces slack variables to allow some misclassifications, adjusting the previous optimisation problem as follows:

$$\begin{aligned}
&\text{Maximise } M \\
&\text{subject to } \sum_{k=0}^{K} \theta_k^2 = 1 \\
&\qquad\qquad y_i(\theta_0 + \theta_1 x_{i,1} + \ldots + \theta_K x_{i,K}) \geq M(1 - \varepsilon_i) \;\; \forall i \\
&\qquad\qquad \varepsilon \geq 0, \;\; \sum_{i=1}^{n} \varepsilon_i \leq C
\end{aligned}$$

This can also be solved by computation of kernels in a similar fashion as before.

Kernel Trick for Non-Linear Data To motivate this, basically for data that is non-linear we can typically construct some mapping to some higher dimensional space such that our data is linear in the sense that we can fit a hyperplane through it. Coming up with such mappings is, in practice, really quite difficult. Luckily for us, we can use standard kernels, $\kappa(\mathbf{x_i}, \mathbf{x_j})$, which are equivalent to mapping our data to some higher dimensional space, by some function $\phi(\mathbf{x})$, but are significantly computationally easier as the kernels are some quite straight forward function to compute without the arduous need to determine the implicit higher dimensional mapping $\phi(\mathbf{x})$ that it represents. So, Mercers theorem gives us this, $\kappa(\mathbf{x_i}, \mathbf{x_j}) = \phi(\mathbf{x_i})^T \phi(\mathbf{x_j})$

We saw before that the problem of fitting our classifier to our training data is equivalent to the following problem, which we add our explicit higher dimensional mapping onto, making it linear, as follows:

$$\max_{\lambda_i \geq 0} \left[\sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j (\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)) \right]$$

Thus, by the kernel trick, as we want to rather now implicitly use this higher dimensional mapping, we can write the above using our $\kappa(\mathbf{x_i}, \mathbf{x_j})$ to show that we can solve this problem instead:

$$\max_{\lambda_i \geq 0} \left[\sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j (\kappa(\mathbf{x_i}, \mathbf{x_j})) \right]$$

Some common forms of kernel used are,

- **Linear kernel**: $\kappa(x_i, x_j) = x_i^T x_j$
- **Polynomial kernel**: $\kappa(x_i, x_j) = (x_i^T x_j + c)^d$
- **Radial Basis Function (RBF) kernel**: $\kappa(x_i, x_j) = \exp(-\frac{||x_i - x_j||^2}{2\sigma^2})$

As is easily spotted, upon choice of the linear kernel, we simply get the form of the equivalent problem for the linearly separable data. The reference [3], specifically recommends the linear kernel for text classification as it is "often linearly separable", which we can see is vastly majoritively the case by Figure 1. Consequently, we proceed in all subsequent analysis by sole use of the linear kernel with reference to SVMs.

4 Statistical Techniques

4.1 Performance Metrics

We calculated some performance metrics using the known classifications (human or ChatGPT-written) within our dataset to evaluate how well our classification methods performed under various conditions.

To perform leave one out cross validation, LOO-CV, we iterated through each essay in some grouping mentioned in Section 3 and split the whole dataset into a test dataset (containing only the essay we wished to classify), "leaving one out", and the training dataset (containing the rest of the essays in our dataset). Then, we trained each of our classification methods using the training data and used those methods to classify the test data. After iterating through every essay in the dataset, we calculated the accuracy of each model; that is to say, the proportion of classifications that were correct.

We were also interested in assessing if topic-specific data or the whole dataset was better used as training data for the classification models. To assess the accuracy using topic-specific data, we applied LOO-CV within a topic-specific essay grouping. In the case of the whole dataset, we again applied LOO-CV using topic-specific data but, at each iteration, we included the rest of the data from the whole dataset in the training data. Accuracies for both of these training datasets were then compared for several topics.

4.2 Optimising Hyper-parameter Selection

In order to tune our classification methods for this particular dataset, we selected the optimal model hyper-parameters that produced the maximum accuracy in LOO-CV using each of the essay lengths we were interested in. "Hyper-parameters" referred to K in K-Nearest Neighbours and P in Random Forests, as well as the choice of kernel, cost, gamma, coef0, and degree in Support Vector Machines.

Due to computational limitations, we did not use the whole dataset to tune these hyper-parameters; instead, we grouped a representative random sample of 10 topics together and applied LOO-CV to that dataset. K in K-Nearest Neighbours and P in Random Forests both took integer values, so we performed LOO-CV with P and K taking integer values from 1 to 70 and selected the best-performing parameter.

As stated in Section 3 for Support Vector Machines we have used the linear kernel which only has the cost hyper-parameter. Cost took continuous values, so a different approach was taken to find its optimal value. A selection of 5 cost values was tested with LOO-CV; if the accuracy varied too much between the parameters, a closer second selection of 5 values centered around the cost with the highest accuracy was then tested. This was iterated until the variation around some cost was reduced to a desired level.

5 Classification Performance Assessment

Given our dataset of 4000 essays, we estimate the relevant hyper-parameters, as in Section 4.2, and subsequently perform LOO-CV on each method to obtain their classification accuracy for this dataset. The graphs in Figure 2, Figure 3, & Figure 4 show how the accuracy of each classification method varies with choice of hyper-parameter, we thus choose the hyper-parameter value corresponding to the highest accuracy.

	DA	KNN	RF	SVM
Accuracy:	97.69%	98.49%	100%	99.76%
Error:	2.31%	1.51%	0%	0.24%
Hyper-parameters:	N/A	K = 60	P = 8	C = 0.1

As we can see above, our classification methods ordered by accuracy are Random Forests, Support Vector Machines, K-Nearest-Neighbours, and then Discriminant Analysis. Whilst we can order them "best to worst", even the technical worst of these classification methods still have a really high accuracy for this dataset. Here, it is easy to say that these methods are all really good at detecting ChatGPT authorship as they are all close to 100%, but these vary quite a lot in terms of their error. For example, the error of DA is roughly 10× worse than the error of SVM. Even still, all methods are high performing, even if we can rank them compared to each other.

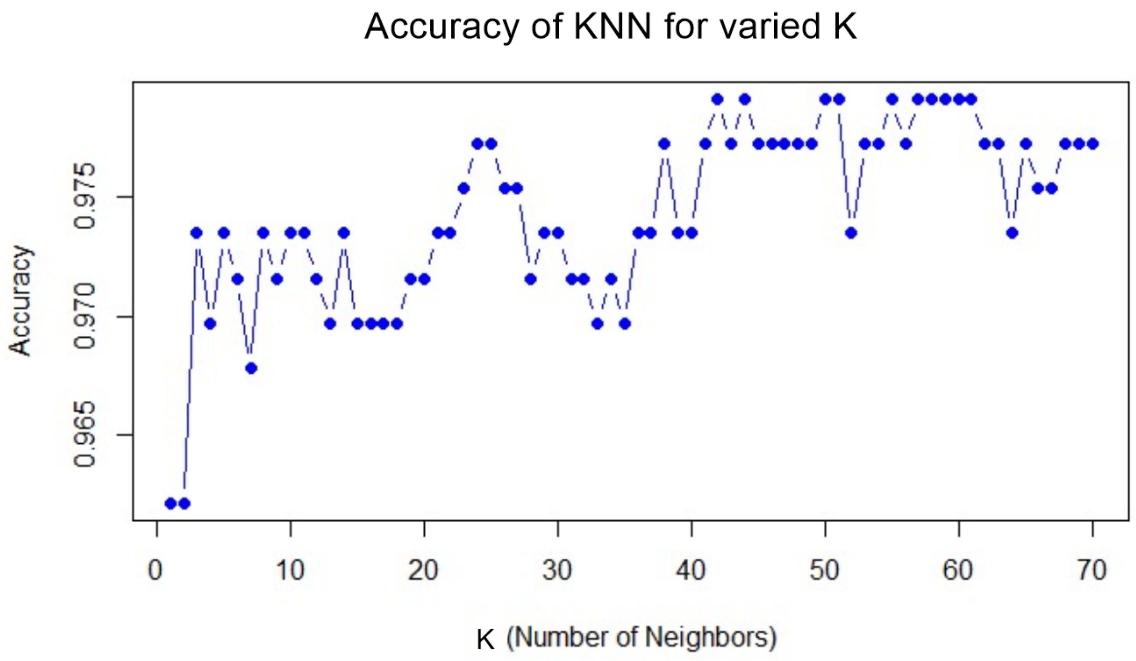

Figure 2: Graph to determine the optimal value of K to maximize classification accuracy

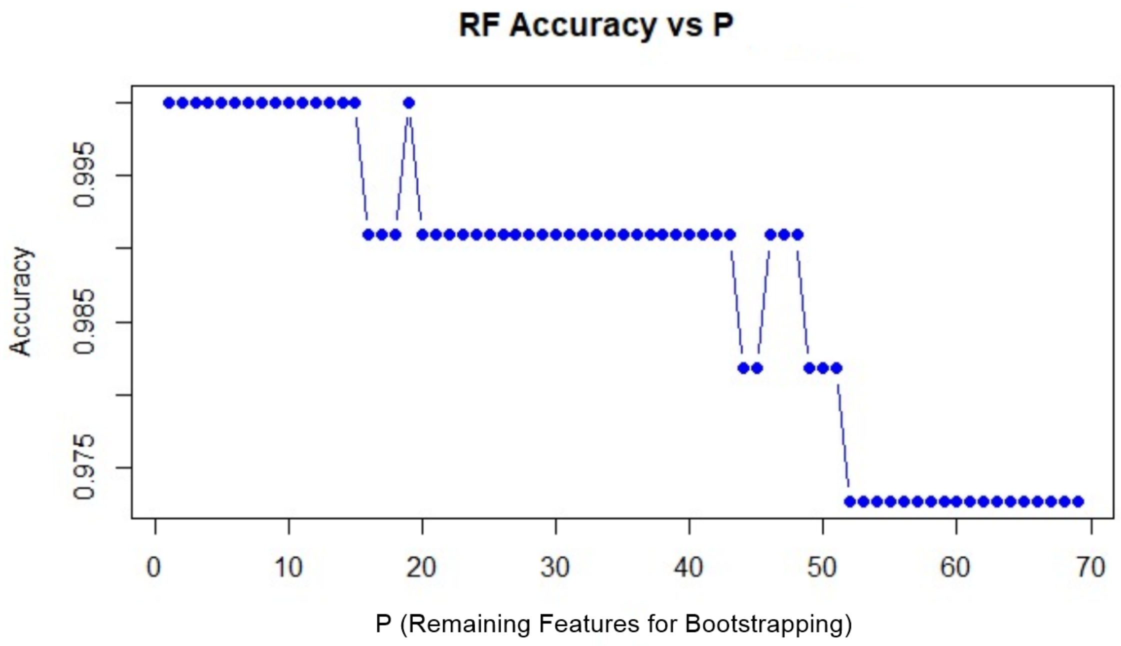

Figure 3: Graph to determine the optimal value of P to maximize classification accuracy

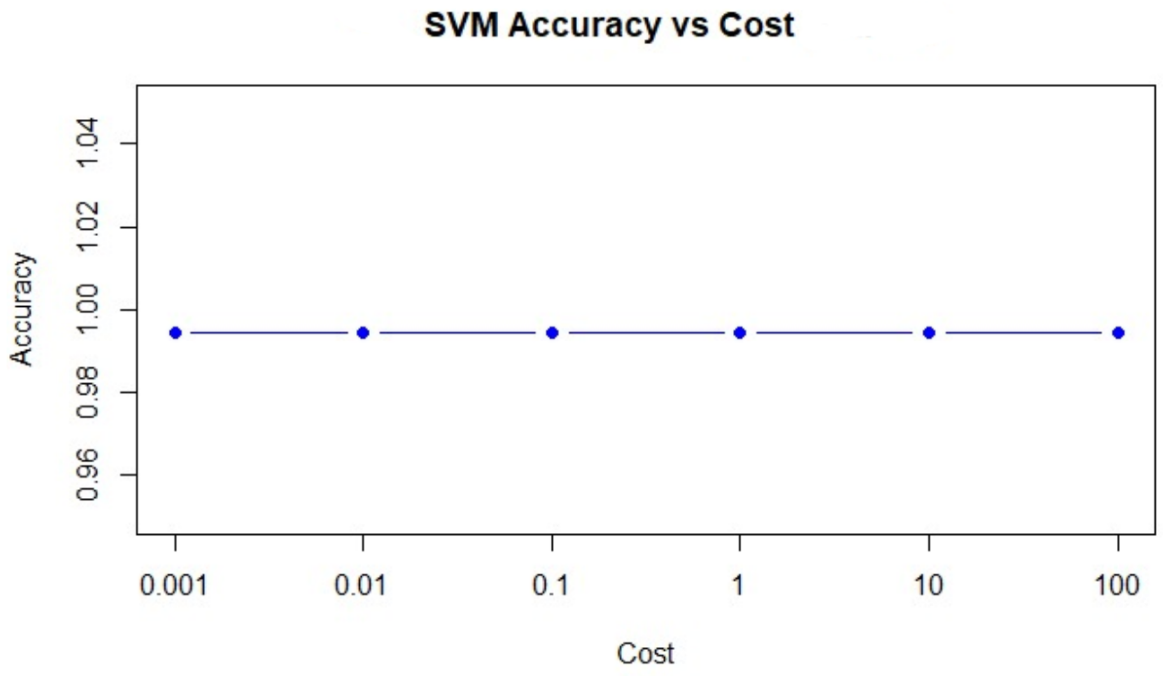

Figure 4: Graph to determine the optimal Cost value to maximize classification accuracy

6 Effects of Dataset Modification on Classification

In the previous section, we saw that all of our classification methods perform into the high 90's for our dataset. However it begs the question, due to the sheer size of our data set, which classification method is the most versatile and resilient to a meaningful reduction in information during classification, through modification of our dataset in some manner.

For instance, the accuracies of the classification methods may change if less data is available, if the essays are shorter, or if we use less function words. To test this, the ways in which we seek to modify our dataset in this section is threefold:

- Making our training data topic specific:
  This meaning that for a given essay of a specific topic, we only consider training data of that specific topic instead of the whole dataset consisting of all other topics too
- Reducing essay lengths:
  To achieve this, we cut down the essay we are testing to some length using the reducewords() function, which essentially randomly samples, with replacement, the specified number of words from the original essay to make the new one. For some chosen length, we perform LOO-CV five times and take the average to reduce the randomness introduced by sampling. We assume this is statistically equivalent to an actual shorter essay. We take the percentage error as the ratio of the error difference of length vs no length to that of the no length restriction error.
- Reducing our function words:
  We remove the count of some chosen function words and add the tallies of those words to the non-function word total.

In the rest of this section, we investigate the impact of reducing information from our dataset in these meaningful ways, as above, on the accuracy and percentage error of our classification methods.

6.1 Whole vs Topic Specific Training Data

In the interest of finding out the effects of training our statistical methods with our datum grouped by topic rather than having each essay as an individual data point, we found the accuracies of all 4 methods using both training sets:

	Classification Methods			
Error (%, 2d.p.)	DA	KNN	RF	SVM
Whole	2.31	1.51	0	0.24
Topic	2.98	5.43	1.25	0.42
Difference	0.57	3.92	1.25	0.18
Difference/Whole error	24.68	258.94	N/A	75.00

In regards to the accuracies of the methods compared to each other. RF is by far the best when using the whole data set, closely followed by SVM, KNN then DA. However for the topic specific training set classifications, the order is instead: SVM, RF, DA, then KNN.

We found differences in determination accuracy for using topic specific training set, specifically where the essays are individually used to train the determination metric performed better than where the combinations of essays regarding the same topic were used instead. KNN and RF performed significantly worse when using the topic specific data set, with KNN having over 250% more error and RF no longer having 0% error. DA, in contrast, only has 25% more error when using the topic specific data. Thus we found that compared to the whole dataset classification, when using the topic specific DA was the had the lowest percent loss in accuracy, then SVM, KNN and finally RF.

Even though the topic specific trained methods were found to perform worse, they were significantly less computationally expensive. Therefore for the sake of investigating the variety of issues discussed in this report, we proceeded to use the topic training data, although it will be a limitation of our results. As certain methods (KNN and RF) were found to have performed significantly worse with the topic specific counterpart, extra care is needed for the results regarding them, as they may not be accurate to the methods using the whole data set.

6.2 Impact of Essay Length

How we further meaningfully removed information from our dataset in this section was by simulating shorter essay lengths, done using the reducewords function. The training data used here is the corresponding topic-specific data as outlined in the previous section. We now proceed by restricting the essay lengths to 1000, 500, 100, 50, & 10, and plot the accuracies of each method against the essay lengths in Figure 5.

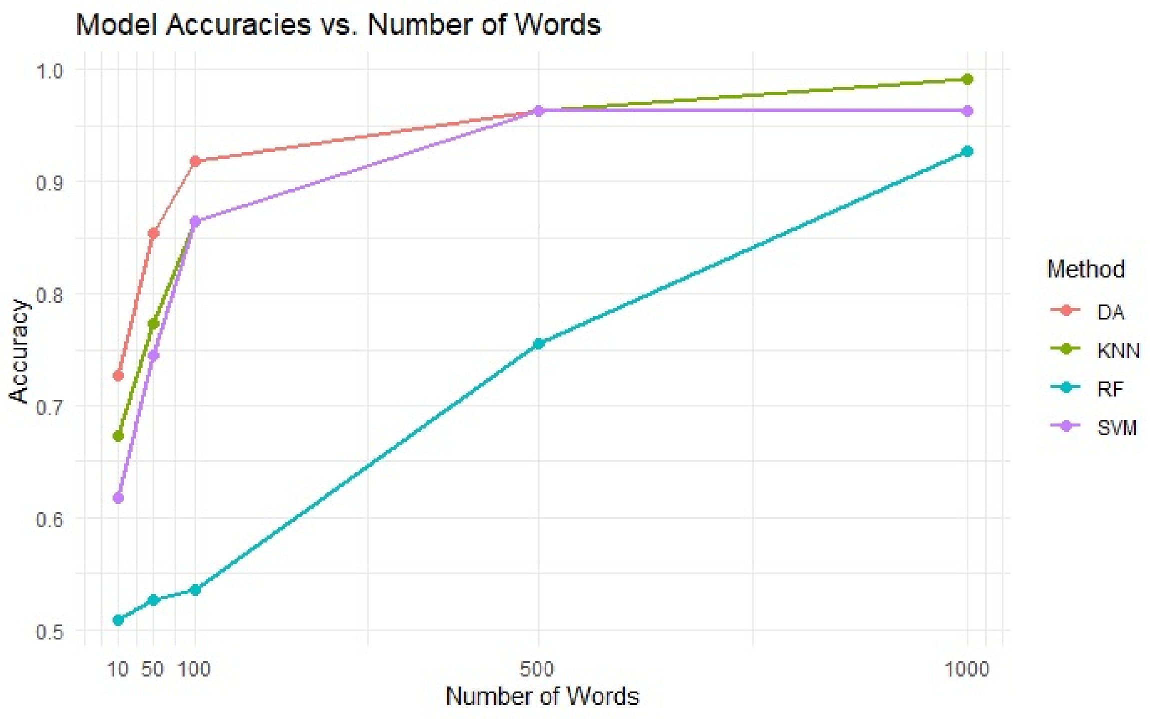

Figure 5: Plot of the change in accuracy for each classification model through variation in essay length.

From Figure 5, we notice immediately that as the number of words in a given essay decreases, the accuracy of all four classification methods also decreases. This could be due to the fact that shorter essays lose key nuances in the function word counts leading to a reduction of the distinction between each of the binary classes. Further, when we use the reducewords() function to sample with replacement to construct our pseudo shorter essays, the function word frequency distributions will tend to become noisier and therefore less representative of the original essay it was sampled from, leading to an increase in misclassifications. Having understood the general movement of classification methods, we now move on to a granular analysis of the particular behaviours of each classification method, justifying claims with appropriate reasoning as follows.

We notice that the accuracy of RF decreases at a much more alarming rate than the other classification techniques. This is because the performance of Random Forests is particularly sensitive to changes in the function words, i.e. the features. As said previously, when the lengths of the essays decrease, the distributions of the function words become noisier, making it much more difficult for the decision trees to consistently split on specific features. Thus, because RF is an ensemble classification technique, this weakens its ability to generalise, which explains the much more pronounced decline in performance when compared with the other methods as its once previous strength is leveraged to a disadvantage when introducing all of this noise.

In the case of SVM, we see that it initially does not change its accuracy from 1000 words to 500 words. We know that using the reducewords() function introduces variability/noise into the data but it could be the case that because SVM's decision boundary is defined by the support vectors, which are small in numbers compared to other data points, it is resilient to this variability in the data. Further, when essays are reduced to 500 words, the distributions of the function words for most essays may remain sufficiently consistent to maintain the original set of support vectors and decision boundaries. It then tapers off with decreasing accuracy as expected, this could be indicative of some threshold level of information in the dataset such that the function vectors for human and ChatGPT essays could begin to overlap more in this reduced feature space. This overlap would make it harder for SVM to maintain a clear separation between the two classes, leading to misclassifications.

Moving to KNN, we can very plainly see that as essay length decreases, the accuracy of this classification method also decreases. This is most likely as KNN calculates distances between the test data vector and all other function vectors to classify it. When essay lengths decrease, the function vectors of shorter essays become noisier, leading to less reliable distance calculations and thusly an increase in misclassification, which gets progressively worse with increase in noise.

Finally discussing DA, we see that it performs the best for all lengths of essays, achieving a greater than 90% accuracy for only 100 words, proving quite impressive given the lack of information. This could most probably boil down to DA being a probabilistic classification model and that it focuses on the holistic distributions of the classes, thus potentially being less prone to significantly impact accuracy, versus any kind of higher dimensional geometrical calculations that may falter due to reasons listed in prior discussions of the classification methods.

Having analysed the graph displaying the relationship between each classification method's accuracy and the pseudo shorter essays, we will now give the specific values of the required hyper-parameters for each length also giving the accuracies of each method and the percentage error.

Length 1000:

Length 1000 Essays	Classification Methods			
	DA	**KNN**	**RF**	**SVM**
Accuracy (%)	99.09	99.09	92.73	96.36
Error (%)	0.91	0.91	7.27	3.64
Rel Error = $\frac{\Delta \text{ Error}}{\text{Topic Error}}$(%)	-69.46	-83.24	481.60	766.67
Hyper-parameters	N/A	K = 16	P = 1	C = 0.001

In the above table, we can see that the percentage error for DA and KNN is negative, meaning that the errors for both are less than the errors for the topic-specific accuracy. This could be because before we set the essay length to be fixed at 1000, the ChatGPT essays are all of length 1000 regardless, but the human-written essays aren't restricted as such, so this affixing of the essay length may, in some capacity, improve the performance of each method. This is a stark contrast to the other two classification methods methods RF and SVM whose percentage error being positive shows a marked decrease in performance. For RF, this could be attributed to, upon fixing length 1000, a decrease in the variety of classifiers and consequently increasing variance, leading to comparatively worse performance. For SVM, because the reducewords() function samples with replacement, it may be the case that the two classifications could become slightly more intertwined with each other at the boundary, making marginal essays near the hyperplane more likely to fall on the wrong side of the hyperplane, reducing accuracy. It seems that this level of accuracy stays stable, as discussed before, up until 500 essay length though.

Length 500:

Length 500 Essays	Classification Methods			
	DA	**KNN**	**RF**	**SVM**
Accuracy (%)	96.36	96.36	75.45	96.36
Error (%)	3.64	3.64	24.55	3.64
Rel Error = $\frac{\Delta \text{ Error}}{\text{Topic Error}}$(%)	22.15	-32.97	1864.00	766.67
Hyper-parameters	N/A	K = 31	P = 1	C = 0.001

We can see that the accuracy for DA is now worse than before using reducewords(), as expected, and that KNN's accuracy has not yet decreased. This apparent discrepancy could be due in part to our usage of the topic-specific data as the training data here and that the accuracy for KNN is quite substantially changed as a result to be anomalous. The RF method has an increased percentage error from the last length, which is also to be expected, and the percentage error of SVM is the same as last time, as discussed previously.

Length 100:

Length 100 Essays	Classification Methods			
	DA	**KNN**	**RF**	**SVM**
Accuracy (%)	91.82	86.36	53.64	86.36
Error (%)	8.18	13.64	46.36	13.64
Rel. Error = $\frac{\Delta \text{ Error}}{\text{Topic Error}}$(%)	174.50	151.20	3608.80	3147.62
Hyper-parameters	N/A	K = 51	P = 57	C = 0.1

It is now true that the percentage error of all four methods is strictly positive, meaning an increase in error in so using the pseudo shorter essays of length 100, thus a lower accuracy. We can see that this continues to be the case for the following essay lengths, so we make no further comment.

Length 50:

Length 50 Essays	Classification Methods			
	DA	**KNN**	**RF**	**SVM**
Accuracy (%)	85.45	77.27	52.72	74.55
Error (%)	14.55	22.73	47.28	25.45
Rel. Error = $\frac{\Delta \text{ Error}}{\text{Topic Error}}$(%)	388.26	318.60	3682.40	5959.52
Hyper-parameters	N/A	K = 53	P = 70	C = 100

Length 10:

Length 10 Essays	Classification Methods			
	DA	**KNN**	**RF**	**SVM**
Accuracy (%)	72.73	67.27	50.91	61.82
Error (%)	27.27	32.73	49.09	38.18
Rel. Error = $\frac{\Delta \text{ Error}}{\text{Topic Error}}$(%)	815.10	502.76	3827.20	8990.48
Hyper-parameters	N/A	K = 66	P = 69	C = 0.1

All Tables: It is to be noted that, while we have commented on the errors and accuracies of the methods, we refrained from observing the hyper-parameters until now, to allow us to discuss how the changes table by table.

Regarding KNN, we see that for high pseudo essay lengths, the value of the hyper-parameter K is closer to 0, whereas, for low pseudo essay lengths, K is much closer to 70. This is interesting because it could point towards KNN attempting to account for the increased variability in the data by taking more and more of the function vectors into consideration to attempt to even out misclassifications, however, it inevitably falls short and experiences performance decline.

We can notice a similar pattern emerge for RF in that optimal hyper-parameter choice P is 1 for both 1000 and 500. However, when there appears to be too much noise from 100 onwards, it seems to want to take into account more and more of the function words from the function vector in the modified bagging process to generate the trees, assumedly attempting to assuage the noise generated by the reducewords() function.

It seems a pattern, or common choice, of Cost, exists for SVM, with the length 50 essay table being an outlier. Apart from essay length 50, the classification for SVM is performed most optimally when the Cost is low for all lengths. This means that the hyperplane decision boundary allows for a non-negligible amount of misclassifications, indicating that the decision boundary may be, at these points, intermingled between a non linearly separable dataset. Looking at essay length 50 and seeing that its Cost is 100 seems anomalous, this may indicate towards overfitting of the hyperplane due to variability in our reducewords() function.

6.3 Function Word Importance

In Section 6.2 we found the methods that are least resilient against changes in essay length are clearly Random Forests and less clearly Support Vector Machines, we also found these methods to be the most computationally expensive and thus limiting. Because of the variable human essay length in the data set, we fixed the length to 1000 words using reducewords() as seen in Section 6. Hence, we moved forward with an investigation of Discriminant Analysis and K-Nearest Neighbours at a fixed essay length of 1000 in this section.

We wanted to explore the impact of reducing our set of function words on our statistical analysis. To find the effect of removing 1 to 50 function words on our statistical accuracy, we first had to determine which function words should be removed. We explored 3 different paths of selecting function words to remove: choosing randomly, removing those with larger standard deviations (sd), and using the coefficient of variations (cv).

For the latter two methods, we found the standard deviations of percentage usage of each function word in the texts, and the cv's were calculated by dividing the standard deviation by the mean percentage usage of the corresponding function word. Assuming independence between function word usage, a smaller standard deviation should represent whether a specific function word is used similarly by different authors, with most using a similar percent of it; these function words would provide less information on an author's style then. A larger standard deviation would imply the opposite, where that specific function word should strongly affect our statistical analysis. A larger cv should similarly represent more variation in the word's usage by different authors, while also factoring in its total usage.

Firstly, we found the results of removing the function words randomly on methodical accuracy for DA and KNN. As portrayed in the following graph, the accuracy of both methods fluctuates greatly with the random selection of function words to remove. This resonates with the idea that not all function words have equal importance in authorship classification. However, while both graphs seem to indicate a shallow decrease in quality, the effect is more prominent in KNN which starts with around 5% error which falls to around 10% when the number of removed function words approaches 50, doubling in error. DA in comparison seems to go from 3% to 5%, implying that KNN is more impacted by the removal of function words, at least when they are picked randomly.

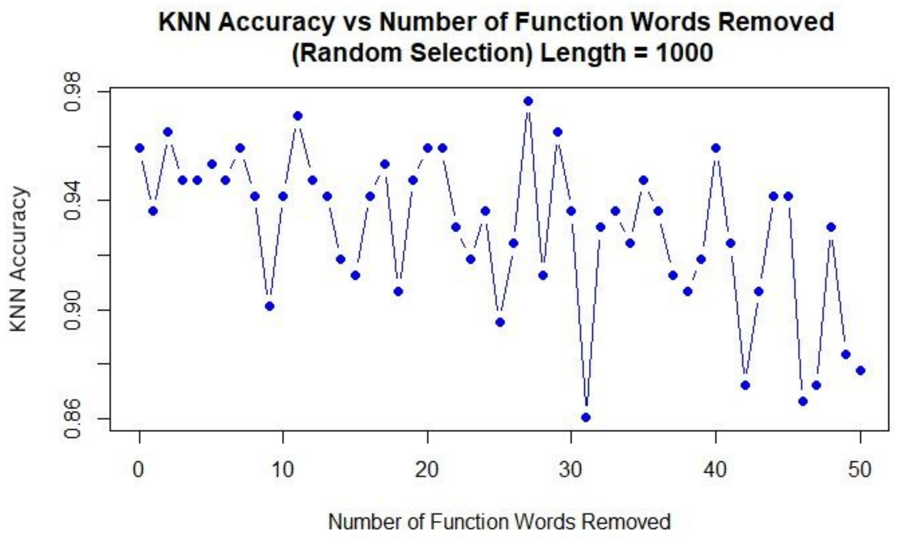

Figure 6: Plot of the change in accuracy for KNN removing random function words.

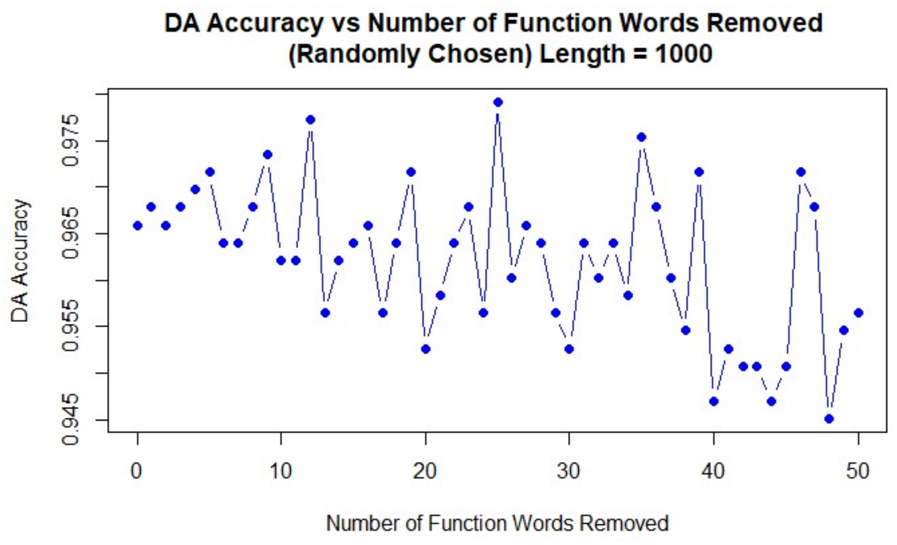

Figure 7: Plot of the change in accuracy for DA removing random function words.

Then we found the effects of removing function words based on those with the highest sd. Compared to when the function words are picked randomly, there is less fluctuation in the accuracy when the function words with higher sd are removed, indicating that words with sd's close in value have a similar effect on authorship determination. In the KNN case, we observe a relatively steady decrease in accuracy with the decrease in function words, with the error going from around 4% to 16%. However, for DA, the accuracy seems to slightly rise and then drop off past the removal of 30 function words. This rise in accuracy is very minor, therefore it could be attributed to randomness, but it is possibly caused by DA's tendency to overfit instead. The accuracy near 50 function words removed is around 6% compared to an initial 3%, once again increasing less proportionally than KNN, reinforcing that DA is less affected by function word removal. Both methods had their accuracies more impacted by the removal of these high sd function words compared to the random ones, implying that high sd function words provide more information to the authorship determination.

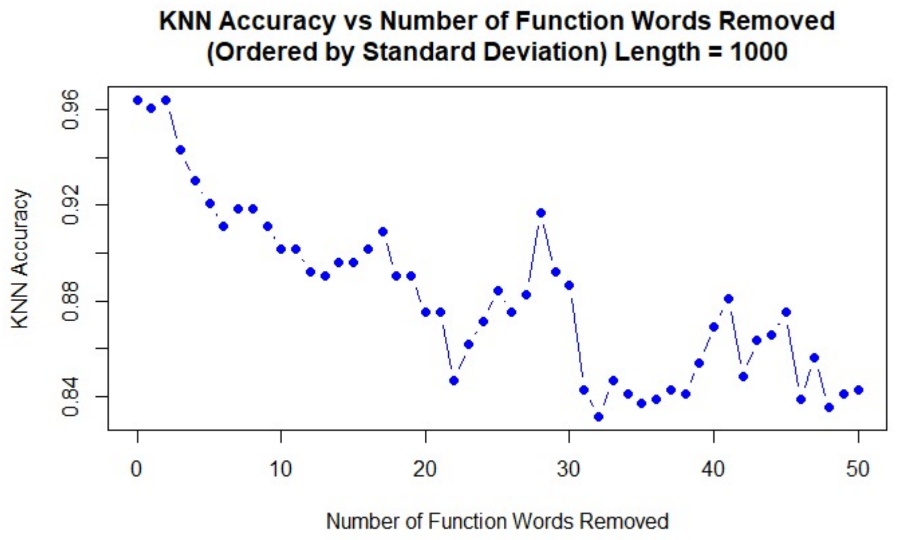

Figure 8: Plot of the change in accuracy for KNN removing function words ordered by largest sd.

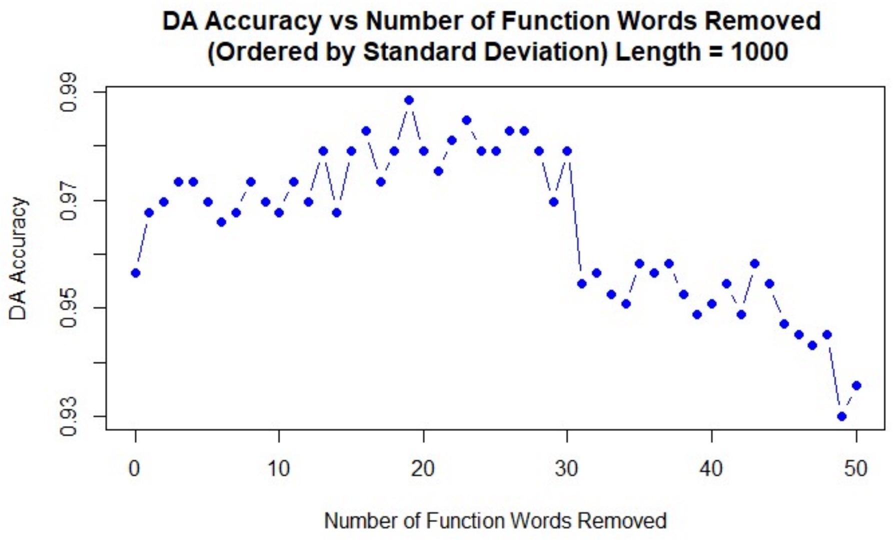

Figure 9: Plot of the change in accuracy for DA removing function words ordered by largest sd.

Finally, we found the impact of removing function words based on the highest cv's. Here, the change in accuracy seemed to have even fewer fluctuations than the sd case, meaning that the relationship between a function word's cv and its effect on methodical accuracy is likely more consistent. While for KNN we saw a similar, though slightly lower, decrease in accuracy, the decrease in DA's accuracy went from around 3.5% with most of the function words to 15%. Here, it is hard to determine whether one method is more affected by the function word removal, seeming to have a similar trend in accuracy decrease. All in all, the accuracy decline suggests that function words with higher variability, especially when regarding the cv, across texts are critical indicators of authorial style.

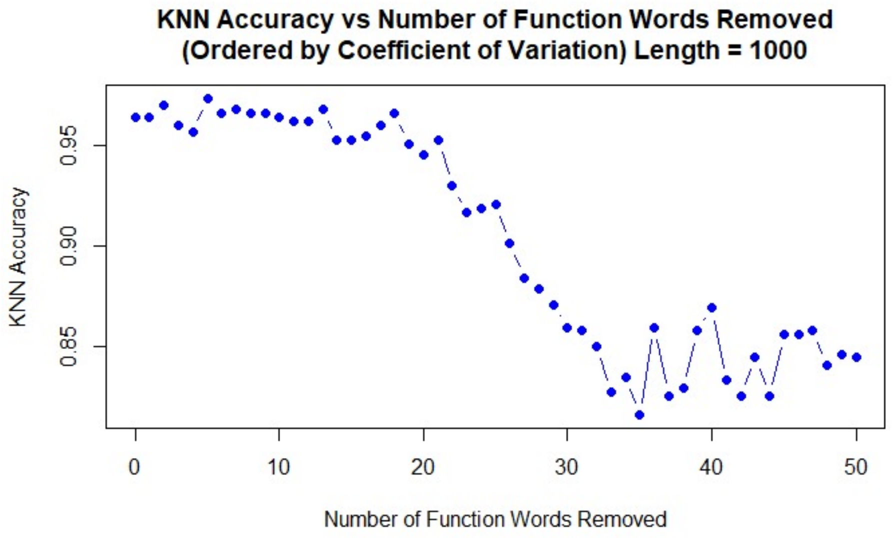

Figure 10: Plot of the change in accuracy for KNN removing function words ordered by largest cv.

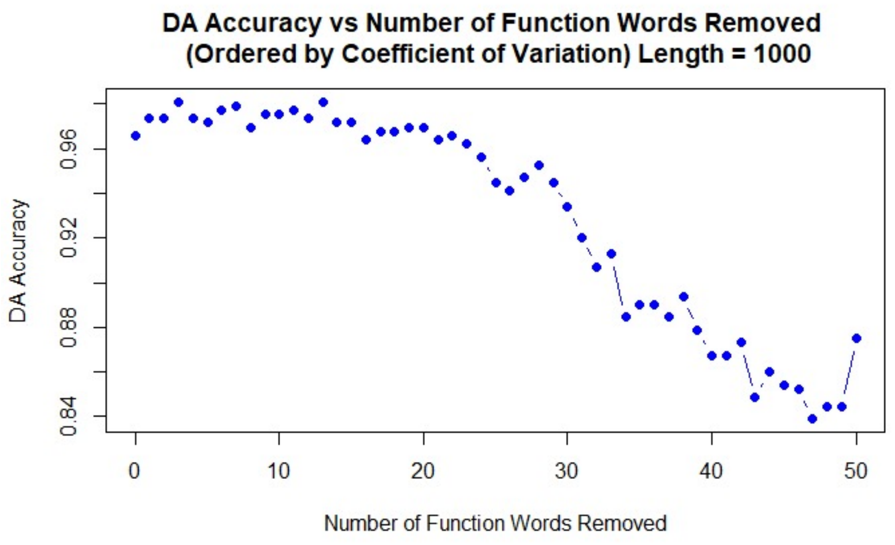

Figure 11: Plot of the change in accuracy for DA removing function words ordered by largest cv.

7 Extensions and Further Research

To improve the applicability of the data set used to learn whether stylometric analysis can be used to analyse the problem of ChatGPT authored papers, the following suggestions are recommended.

- Increasing the dataset size, sourcing more human and ChatGPT essays, and adding essays of more topics.
- Using different texts types, e.g. internet posts written by humans and AI bots, preferably based on ChatGPT.
- Using essays from various sources, with particular attention to the diversity and population representation of the authors. [7]
- Requesting ChatGPT generates the essays by the average essay length of the specific topic, rather than a general 1000 word length. [7]
- Instead of using a single prompt and only providing the AI with the essay's titles, using a variety of essay generation prompts and providing more details on the essay, such as the thesis, if applicable. Aiming to replicate real user behaviour when using ChatGPT for essay, or text writing requests, specifically in cases where avoiding detection would be a user's concern, in the data studied would best increase result accuracy.

The suggestions to refine the stylometric analysis done are following.

- Using different stylometry methods, e.g. considering grammatical, structural and lexical features [5].
- Analysing whether other statistical methods, such as cluster analysis and correspondence analysis [2], are better predictors for ChatGPT authorship.
- If studying the impact of reducing function words again, it would be recommended to find the relationship and correlation between usage of the function words to find out how removing them is best.
- Use the whole dataset instead of the topic specific one for the method training data.
- Test the impact of reducing function words using the other methods, not just DA and KNN.
- Research more about relations in topics ChatGPT writes more similarly to humans.

Future research should expand from solely focusing on ChatGPT; other AI chatbots, like Microsoft Copilot or Gemini, are frequently used as a tool for text generation and may outdo ChatGPT in replicating human writing style in different essay topics [6]. An additional consideration would be exploring how ChatGPT, and the detection of its authorship, has changed throughout its versions, to focus on improving the detection of traits newer models improved on [8].

8 Conclusion

Throughout this report, we investigated the performance fluctuation of each of our four classification methods given worthwhile modification of our dataset to decrease information available in classification, allowing us to make comparison between each method. We found that, without modification, our classification methods perform with a high degree of accuracy, all above 97.5%, with order of performance given by RF, SVM, KNN, & DA. We also saw that there is a potential link between topics being related to society and ChatGPT writing them more human-like, which could possibly be explained by ChatGPT having more information regarding these societal topics in the its corresponding LLM dataset.

Considering our varied approaches towards decreasing information available for classification, we found that the method most resilient to this was DA. When only using topic specific training data, we see that the percentage error of DA is the lowest of all classification methods, and KNN has the highest percentage error. In our essay length reduction, we saw that DA consistently performed higher than all other classification methods and that the accuracy of RF dropped off the fastest. Lastly, through function word reduction by importance, we saw that DA still has majoritively greater, or only once equal to, accuracy when compared to KNN.

In conclusion, we have found that for a dataset consisting of a vast quantity of essay's function vectors with a reasonable choice of function words, Random Forests is the best classifier of ChatGPT authorship. However, if the dataset has a significantly lower amount of information with which to train classifiers, Discriminant Analysis is the most resistant to this lack of information whilst still maintaining a notably high accuracy at ChatGPT authorship classification.

References

[1] Mann Acharya. ‘Aeon Essays Dataset’. In: *Kaggle* (Jan. 2024). URL: https://shorturl.at/vPLoo.

[2] David I Holmes and Judit Kardos. ‘Who was the author? An introduction to stylometry’. In: *Chance (N. Y.)* 16.2 (Mar. 2003), pp. 5–8. URL: https://shorturl.at/DezCh.

[3] T. Joachims. ‘Text Categorization with Support Vector Machines: Learning with Many Relevant Features’. In: *European Conference on Machine Learning (ECML)* (1998), pp. 137–142. URL: https://tinyurl.com/4r5udsfb.

[4] Dmitry Kobak et al. ‘Delving into ChatGPT usage in academic writing through excess vocabulary’. In: *Cornell University: arXiv* (June 2024). eprint: 2406.07016 (cs.CL). URL: https://arxiv.org/html/2406.07016v1.

[5] Berriche Lamia and Larabi-Marie-Sainte Souad. ‘Unveiling ChatGPT text using writing style’. en. In: *Heliyon* 10.12 (June 2024), e32976. URL: https://shorturl.at/1VxyO.

[6] Sabrina Ortiz. ‘The best ai chatbots of 2024: Chatgpt, copilot, and worthy alternatives’. In: *ZDNET* (Aug. 2024). URL: https://tinyurl.com/4fktnshp.

[7] Nello Cristianini Sheng Wang and Bruce Hood. ‘Stylometric Comparison between ChatGPT and Human Essays’. In: *Workshop Proceedings of the 18th International AAAI Conference on Web and Social Media* (2024). URL: https://shorturl.at/bsNmC.

[8] Zaitsu Wataru and Jin Mingzhe. ‘Distinguishing ChatGPT(-3.5, -4)-generated and human-written papers through Japanese stylometric analysis’. In: *PLoS One* 18.8 (Aug. 2023), e0288453. URL: https://shorturl.at/HxITL.