Did Mary Shelley Write 'Frankenstein'?
A Stylometric Analysis

s1714840

Year 4 Project
School of Mathematics
University of Edinburgh
2021

Abstract

Mary Shelley's authorship of the novel 'Frankenstein' has been questioned since its then-anonymous publication in 1818. Many have claimed her husband and famous poet Percy Bysshe Shelley was a major collaborator, or even the author himself. However, these claims have not gained significant traction within mainstream literature. All studies of Mary Shelley's work thus far have been qualitative in nature (textual analysis). We apply stylometry - a method for quantifying an authors' styles using textual features - to the 'Frankenstein' authorship question

We analyse Mary Shelley apply , a method of quantifying an author's style using textual features have never through statistical analyses and perform stylometric authorship attribution for 'Frankenstein' using her works and the works of her closest friends and family.

Declaration

I declare that this thesis was composed by myself and that the work contained therein is my own, except where explicitly stated otherwise in the text.

(s1714840)

*First of all, thank you to Gordon Ross, my supervisor,
for your sound advice great ideas
This thesis is dedicated to the my family, and Liv, and
my friends
Shout out to Stephen Dunne who moved my keyboard
here haha
And, finally, thank you to Edinburgh, my home for the
last four years*

Contents

Chapter 1

Introduction

We begin by motivating our key research question - ‘Did Mary Shelley Write “Frankenstein?”’. We first introduce the background of the novel - its origins, influences, publication, and the disputes of its authorship. We also examine claims that she received a lot of help from her husband Percy Bysshe Shelley, and even claims that she did not write it.

Next, we introduce the field of stylometry - the motivation behind its use, its history and origins, several historical examples of its use, the technologies today’s stylometrists use, and, above all, its suitability for answering the ‘Frankenstein’ authorship question. Also, throughout this Chapters 3 and 4, we explore the stylometry literature.

Then, we introduce a step-by-step framework for performing stylometric analyses which we apply in a simple case study to validate our methodology: with a dataset of 9 writers from the 19th/20th centuries we perform a test authorship attribution task for the book ‘Agnes Grey’ written by Anne Brontë. Through this case study we introduce the type of preprocessing required for our text datasets, many of the features stylometrists use, and the algorithms we use for performing classification and visualisation of our data. We also validate methods/ perform instance-based classification using k-NN for short author chunks.

Finally, we apply all we have learned in answering the ‘Frankenstein’ authorship question. We begin by introducing the dataset we have chosen for the task, then we apply our stylometric framework. We then discuss our results and suggest directions for further study.

Chapter 2

Background and Motivation

In this Chapter, we investigate the background of the novel ‘Frankenstein’ - who was involved in its composition, and why its authorship is disputed. This will motivate our key research question: ‘Did Mary Shelley write “Frankenstein”?’. This discussion gives us the historical context that is vital when performing stylometric studies like this.

2.1 The Novel

‘Frankenstein; or, The Modern Prometheus’ is an 1818 novel that follows the story of a young, unorthodox scientist, Victor Frankenstein, who creates a sapient creature - Frankenstein’s monster. Published anonymously, but later attributed to Mary Shelley, it went on to achieve widespread popularity, becoming a classic novel of Romantic, Gothic, and science fiction literature, and it continues to influence literature and popular culture to this day.

2.2 Who Was Involved?

Mary Shelley (née Godwin) is the supposed author of the book. Born in 1797 as Mary Wollstonecraft Godwin, she was the daughter philosopher and feminist activist Mary Wollstonecraft, and writer and political philosopher William Godwin.

Percy Bysshe Shelley was Mary Shelley’s husband, and is one of the most renowned poets of all time. He has often been credited with writing much if not all of ‘Frankenstein’, but is typically regarded as having had the same influence as that of a publisher’s editor.

Lord Byron was a friend of the Shelleys, another widely acclaimed poet, and a politician. He joined up with the Shelleys on their tour of Europe, the time the novel was conceived.

William Godwin was Mary Shelley’s father, a writer, and political philosopher. His work greatly influenced that of Percy Bysshe Shelley and Lord Byron, and no doubt that of Mary Shelley herself due to the rigourous education he gave her growing up. However, his relationship with his daughter broke down when she eloped with Percy Bysshe Shelley at the age of 16.

Figure 2.1: From left - Mary Shelley, Percy Bysshe Shelley, Lord Byron, and William Godwin
Source: Richard Rothwell; Alfred Clint; Thomas Phillips; Henry William Pickersgill

Their portraits are shown in Figure 2.1

2.3 Origins of the Novel

Mary Godwin and Percy Bysshe Shelley completed several tours of Europe together. On one particular trip to Geneva in the summer of 1816, they visited Lord Byron, and his personal physician and fellow writer John Polidori. Bad weather, due to the eruption of Mount Tambora in 1815 that led to the 'Year Without a Summer', confined the group indoors where they read German ghost stories. Conversations of Galvanism and the occult ensued, and Byron soon suggested they all write their own ghost stories. Mary Shelley, perhaps inspired by Frankenstein Castle (shown in Figure 2.2) which she passed nearby months previous, conceived the idea for 'Frankenstein'. She was only 18.

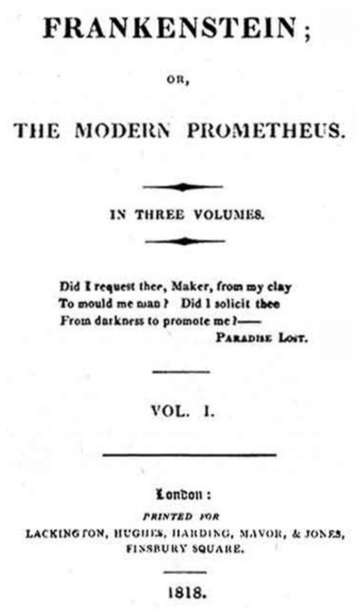
Less than three years later, 'Frankenstein' was published, and the lovers had been married. In fact, two seminal works were conceived that summer in Geneva, as Byron's 'Fragment of a Novel' was expanded by Polidori into the short story 'The Vampyre', one of the first published modern vampire stories written in English, and one that was a major influence on Bram Stoker's 'Dracula'.

Figure 2.2: Frankenstein Castle
Source: tinyurl.com/24b48u8f

2.4 Publication and Disputes

'Frankenstein' was published on 1st January 1818. It was published anonymously, as can be seen in title page shown in Figure 2.3. Featuring a dedication to William Godwin, an anonymous preface written by Percy Bysshe Shelley, as well as quotes from Percy Shelley's 1816 poem 'Mutability', there was much confusion as to the novel's true author. Many critics suggested Percy Shelley was the novel's author, recognising themes he had focused on in his other works, and as he was a loyal supporter of Godwin, however he denied this at the time. One such critic was Walter Scott, who said of the novel 'it is said to be written by Mr Percy Bysshe Shelley', in the Blackwood's Edinburgh Magazine [16].

Women publishing their works anonymously or under male pen names names was a common practice at the time, often to avoid prejudice. Decades later, the Brontës published their books under pseudonymous male names (for example Acton Bell for Anne Brontë).



FRANKENSTEIN;

OR,

THE MODERN PROMETHEUS.

IN THREE VOLUMES.

Did I request thee, Maker, from my clay
To mould me man? Did I solicit thee
From darkness to promote me?——
PARADISE LOST.

VOL. I.

London:
PRINTED FOR
LACKINGTON, HUGHES, HARDING, MAVOR, & JONES,
FINSBURY SQUARE.

1818.

Figure 2.3: Title page of the 1818 edition of 'Frankenstein'
Source: https://tinyurl.com/n8jsved3

The 2nd edition of 'Frankenstein was published in 1823 under the name Mary Wollstonecraft Godwin, however Percy Bysshe Shelley was not credited for his preface and poem. The edition was supervised by William Godwin.

In 1824, Mary faced further criticism and doubt, this time from an anonymous writer in the literary magazine 'Knight's Quarterly Review'.

The 3rd edition published in 1831 saw Percy Shelley's contribution finally acknowledged.

In 1974, original drafts of the novel were released that showed sections written in Percy Shelley's handwriting. This led James Rieger, a professor and editor, to argue in 1982 he should be credited as a minor collaborator of the novel, saying 'We know that [Shelley] was more than an editor. Should we grant him the status of minor collaborator?'[14].

Later, in 1996, Charles E. Robinson published a transcribed edition of the manuscripts of 'Frankenstein' together with a chronology of the work's composition. His conclusion was that Shelley's contribution amounted to no more than that of an editor today. Duncan Wu, in 2015 concurred with this conclusion from Charles E. Robinson [19].

The most recent claims of Percy Shelley's complete authorship have come from a series of spurious publication by John Lauritsen, notably his book 'The True Author of Frankenstein'[7]. In it he argues, for reasons unknown, that Percy Shelley did not want his authorship of 'Frankenstein' to be known, that the handwritten drafts are not conclusive proof as they are incomplete, and that sections of the novel were dictated by Percy Shelley.

2.5 How Do We Proceed?

We eschew a textual analysis of the work - analysing things like themes, imagery, and tone associated with specific authors. Instead, we want to use statistics, namely stylometric methods, to investigate the 'Frankenstein' authorship question, something that has not yet been attempted.

We first need to understand where stylometry has come from and how it works in order to apply its methods in the best way. Once

After, we will

Chapter 3

What is Stylometry?

In this section we will give an overview of the field of stylometry - why we use it, where it came from, then outline its basic methods, and examine some use cases and historical examples.

Stylometry, coming from ‘stylus’ (Latin) for stake or pole, and ‘meter’ (Greek) meaning measure, is the statistical or computational analysis of style, and is most often used to determine the author of disputed anonymous works. Stylometry works by measuring a set of textual features, for example the average sentence length of a document, and comparing this to the works of all possible authors, assigning the unknown work to the most similar candidate author. The basic methods of stylometry were set out by Polish philosopher Wincenty Lutosławski in his 1890 work ‘Principes de stylométrie’[8].

While it is usually focused on linguistic style, it has also been applied to artistic style (e.g. fine-art paintings), musical style, and even computer code.

One key assumption behind much of stylometry is that each individual author unconsciously writes with a unique distribution of stylometric properties, for example their use of function words (grammatical words like ‘the’, ‘and’, and ‘or’), and that this distribution is consistent across their works (that are long enough to be statistically significant?) and is unlike any other author’s. This unique set of features is known as the **author invariant**, and can be thought of as a sort of autograph or signature of an author.

One crucial point about stylometry is that it cannot definitively make claims about authorship, but only offer statistical probabilities and evidence. Like all statistical methods, it must be used in context. In summary, ‘Stylometric analysis is highly accurate, but ultimately, it must be supported by historical and circumstantial evidence to solidify any conclusions about authorship.’[1]

3.1 Why Use Stylometry?

The typical use case for stylometry is **Authorship Attribution** (also called Authorship Identification). This is where we are given a set of works of known

[1]Quoted from ’https://www.pbs.org/opb/historydetectives/blog/how-we-solved-it-stylometric-analysis/’

authorship from several candidate authors, and we use these to identify the author of a disputed or anonymous work written by one of the given authors.

A similar problem is Authorship Verification, where we are given a set of works by a single author and are asked if a separate text is written by that author or not. It can be argued, the fundamental stylometric task is taking two works and asking if they are written by the same author[6].

The other most common linguistic applications of stylometry are:

- Author Profiling or Characterisation: Trying to determine the gender, age, race, etc. of an author given some of their work. This could also be extended to profiling the time period or genre of a work, or the author’s level of proficiency with a language.
- Detecting Plagiarism: Trying to determine the similarity between documents.
- Detecting Stylistic Inconsistencies: Trying to find sections of writing that were written by a different author. This could potentially be used in the ‘Frankenstein’ authorship question to find sections greatly influenced by either Percy Bysshe Shelley or William Godwin.
- Identifying Vandalism on Wikipedia.
- Detecting Forgeries: Trying to detect if a work was not written by the supposed author.

3.2 How Does it Work?

To perform stylometric studies, stylometrists try to extract features that represent writing style from authors’ texts.

Possible features include:

- Average word or sentence length of the authors’ works.
- Character frequencies: the distribution of the use of each character (for example one author could favour words with ‘z’ far more than another).
- Word frequencies.
- Vocabulary richness: how wide an author’s vocabulary is. This can be measured crudely by the number of unique words as a proportion of all words in a text.

These features are then collected in vectors that represent either the style of a particular piece of writing (a short passage or a whole book) or the author’s overall style.

Then, these features are used to classify the unknown text using a variety of statistical methods. The methods we use will be developed in Chapter 4.

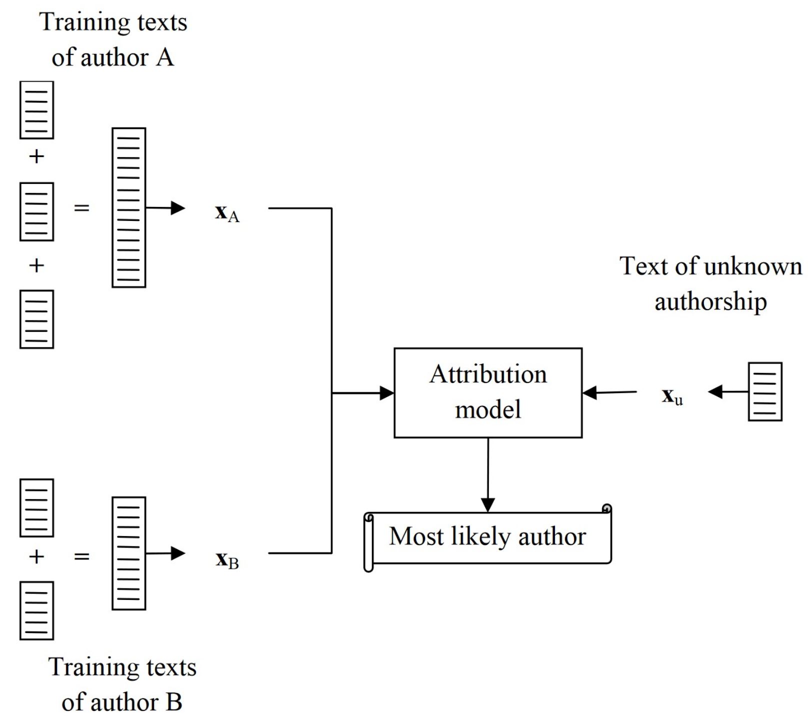

Training texts of author A
+
=
+
x_A
Text of unknown authorship
Attribution model
x_u
Most likely author
+
=
x_B
Training texts of author B

Figure 3.1: Typical architecture of profile-based approaches
Source: [17]

3.2.1 Profile-Based vs Instance-Based Methods

There are two general methodologies for authorship attribution based on whether we treat the texts of each author cumulatively or individually. These, in a sense, define the philosophy of the analysis - using the texts cumulatively means we use generative models, whereas using the texts individually means we use discriminative models. Of course, this means our two approaches differ in what algorithms we will use to classify texts.

Profile-based methods concatenate all an author's texts together, and use the combined texts to extract features that represent an author's *profile*, which consists of the stylistic tendencies across their work. The typical architecture for a profile-based approach is shown in Figure 3.1, where the training texts of author A are combined to yield a single style vector $\mathbf{x}_A$, and the same for author B, giving $\mathbf{x}_B$. The text of unknown authorship gives a style vector $\mathbf{x}_u$. The attribution model then gives us the most likely author for this text.

In contrast, **instance-based methods** treat each instance of an author's writing (which could be whole novels, short stories, chapters, paragraphs, or even sentences) as separate examples of how the author writes. The typical architecture for a profile-based approach is shown in Figure 3.2, where the training texts for authors A and B are kept separate, giving style vectors for each text $(\mathbf{x}_A, 1$, $\mathbf{x}_A, 2$, $\mathbf{x}_A, 3$, $\mathbf{x}_B, 1$, and $\mathbf{x}_B, 2)$. Another step is usually involved - that of training the classifier. This added training step accounts for processes like construction of decision boundaries. In the case we do not have lots of short training examples, we can create them by segmenting longer texts.

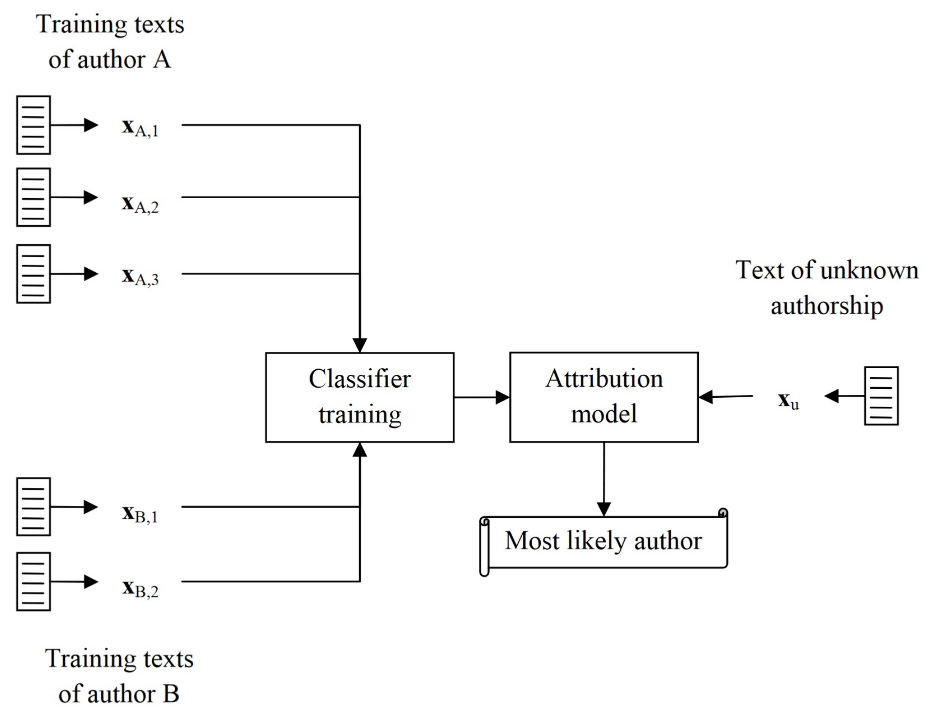


Figure 3.2: Typical architecture of instance-based approaches
Source: [17]

There are benefits to both approaches:

- With a profile-based approach, we get a consistent look at how an author writes, there is less computation involved, and classification methods are simpler.

- On the other hand, an instance-based approach captures the deviations and changes in author's style, and it challenges the assumption that authors have a consistent style. Intuitively, an author is not going to write with the same style in every book or even chapter, especially as they write from different viewpoints/narrators, and about different characters and stories. There is also the potential to use very sophisticated machine learning algorithms.

For both approaches, more training examples is pretty much always a good thing as long as the quantity of text is balanced between the candidate authors. This will yield more accurate average styles for profile-based methods, and a classifier with a bigger training set for instance-based methods.

3.3 History of Stylometry

We now explore the history of stylometry - how it came into being, the definitive pieces of research that established it, and the modern technology at the cutting edge.

3.3.1 In the Past

Before stylometry was developed, attributing the authorship of anonymous or disputed texts was originally done by hand. The focus was on qualitative differences, or small sets of distinguishing features. It relied on experts, and often used rare elements of text - for example, two authors might be distinguished by one's preference for 'while' and another's for 'whilst'.

Donation of Constantine

An early example of research resembling stylometry is that of Lorenzo Valla in the 15th century. He showed the Donation of Constantine, a Roman imperial decree, was actually a forgery probably made in the 8th century. The decree was supposedly issued in the 4th century by Roman emperor Constantine the Great that transferred power to the papacy, and in the 13th century was used to support many Popes' claims of power [18]. Valla used the presence of anachronisms, showing the language of the document could not have been dated to the 4th century, but more likely the 8th century, and argued the document was so obviously forged that many Popes used it knowing it was inauthentic [13]. His argument was based on **philological** arguments - using textual analysis and linguistics/historical-based arguments, and it is an early example of modern diplomatics, a field concentrating on the critical analysis of historical documents.

The Work of Mendenhall

An early attempt to quantitatively measure writing style was suggested by famous mathematician and logician Augustus de Morgan who in 1851 "in a letter to a friend of his, brought forth the idea of studying word length as an indicator of individual style, and as a possible factor in determining authorship" [5]. His idea was to measure the average length of the words in each of Paul's Epistles, and use them to determine which have been falsely attributed to him.

This idea inspired Thomas Corwin Mendenhall, often regarded as the father of author profiling, in 1887 to analyse not just the average length of words in a text, but to graph the overall use of words of different lengths [11]. This yields what he called 'characteristic curves', shown in Figure 3.3, which shows two curves of 1000 word sections of Charles Dickens' 'Oliver Twist', which are remarkably similar. Also referring to them as 'word spectrums', he compared these curves to the frequency spectrums generated in spectroscopy - in spectroscopy, the mass spectrum for elements are unique and are used to classify unknown substances. Mendenhall believed this same principle could be used to find the unique features of an author's writing.

Mendenhall foresaw the potential of this kind of analysis, writing:

> "It is hardly necessary to say that the method is not necessarily confined to the analysis of a composition by means of its mean word-length: it may equally well be applied to the study of syllables, of words in sentences, and in various other ways."

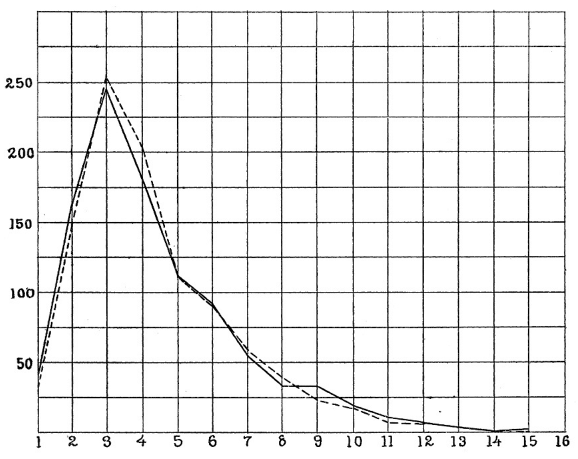


Figure 3.3: Characteristic curve for two 1000 word sections from 'Oliver Twist'
Source: [11]

3.3.2 Case Study - The Federalist Papers

'The Federalist Papers' were a collection of 85 essays written in the late 18th century in support of the United States Constitution. Three authors were involved - Alexander Hamilton, James Madison, and John Jay - however the Papers were published under a collective pseudonym 'Publius'. There was a problem; while the authorship of 73 of the essays was well established, 12 remained that were contested between Hamilton and Madison. The problem was how to identify who wrote each essay.

Frederick Mosteller and David L. Wallace, in 1964, published a seminal study that seemingly solved the authorship of the disputed essays [12]. They popularised and pioneered the use of statistical methods for measuring style, essentially founding the field of stylometry, and firmly establishing it as a central tool in determining authorship. They were even featured in TIME magazine for their work.

They used the frequencies of function words (grammatical words like 'the', 'and', and 'with') as discriminators between the author's styles as seen in the known essays. For example, the word 'upon' has 3.24 appearances per 1,000 words in Hamilton's work but only 0.23 in Madison's. They then expressed degrees of belief about claims like 'Madison wrote Paper x', and then used Bayes Theorem to adjust for the evidence.

They found all 12 papers were written by Madison which was the consensus at the time, however, since then, it has been argued that the collaborative nature of the Papers composition (the authors worked broadly together) means that authorship attribution is a naïve solution [2].

3.3.3 Since Then

Since Mosteller & Wallace's pioneering work, there have been many problems addressed with stylometric techniques. Some examples include:

- The Shakespeare authorship question: Modern neural network techniques applied to it?

- The field of forensic linguistics which has been helpful in:
  - Uncovering the identity of the Unabomber (Ted Kaczynski) by comparing his manifesto 'Industrial Society and Its Future' and letters written to his brother David Kaczynski [3].
  - Detecting malicious code. This is a field called code stylometry which usees features like how a given piece of code solves a computational problem (overall approach), the way it is implemented in the source code of the given language, and, of course, the actual formatting of the code itself (e.g. borders/tab length, spacing, variable naming styles). An example is the work of Stephen MacDonell [9].
- The Beatles song 'In My Life' (disputed between Lennon and McCartney) was analysed by researchers who used the 'melodic notes, chords, melodic note pairs, chord change pairs, and four-note melody contours' of the song [4] - they used not text but musical forms. They found the likelihood that Paul McCartney wrote the 'middle eight' (the bridge) was almost 50%, but that Lennon very likely wrote the rest of the song.
- In 2020, the classic novel 'Wuthering Heights' was analysed by Rachel McCarthey and James O'Sullivan in a very similar authorship attribution task to our own [10]. They found Emily Brontë is the novel's true author after some critics claimed it was written by either Branwell or Charlotte Brontë.

3.3.4 Stylometry Today

Today, modern computers and access to vast amounts of text data from the internet have changed things dramatically: we have unprecedented computational power, access to documents online in machine readable forms (like .txt). Modern computer science has given us powerful technologies like:

- Natural language processing (NLP) tools that allow us to analyse and edit text efficiently to quickly retrieve features, and the potential to create higher level features through automated analyses like grammatical parsing.
- Machine learning (ML) algorithms that allow us to handle large amounts of high-dimensional data, offer a wide variety of sophisticated classification algorithms, and the ability to generate more abstract feature representations.
- Information Retrieval (IR) techniques that make representing and classifying large quantities of text data easy and quick.

Stylometry packages streamline analysis and make it accessible to non-technical users. A popular example is the 'stylo' package for the R programming language[2].

[2]https://cran.r-project.org/web/packages/stylo/index.html

Chapter 4

How Does Stylometry Work?

We now introduce our step-by-step framework for performing stylometric analyses, namely authorship attribution.

Stylometry Framework

1. Choose Dataset and Preprocess Data: We first choose our dataset which depends on the task at hand. Preprocessing involves reading in our files, possibly editing them, and finally putting them into data structures for use in the next step.

2. Extracting Features: We use our preprocessed data to extract the features that represent authors' styles. At this stage we need to choose our methodology - whether we use profile- or instance-based approach.

3. Choosing Algorithms: Once we have our features, we then need to choose the classification algorithms we will run on our dataset.

4. Evaluating Results: Finally, we apply our algorithms to the dataset, and evaluate the results.

We see how to apply this framework in a baseline task - attributing the novel 'Agnes Grey' by Anne Brontë. This acts as a control for our classification methods, as this is a dataset of works of a similar time period to 'Frankenstein'. We also introduce and validate the use of all of the methods we will be using later for attributing 'Frankenstein'.

4.1 The Dataset

Of course, we first need a dataset and a goal. For a validation of our methods we use a dataset of nine authors from the 19th/20th centuries. We take 2/3 texts of each author's works (novels, plays) which we detail in Table A.1 in Appendix A. This comprises our **corpus**, what linguists call a large, structured set of texts. Our **task** is to identify the author of 'Agnes Grey', which we assume for the purposes (of most) of our experiments to be disputed between the authors of our corpus. We also use the corpus to test our methods using a train/test split where we include 'Agnes Grey' in Anne Brontë's works.

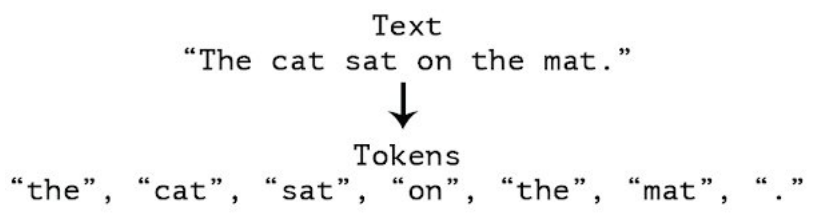

Text
"The cat sat on the mat."
Tokens
"the", "cat", "sat", "on", "the", "mat", "."

Figure 4.1: An example of tokenising the sentence 'The cat sat on the mat.'
Source: freecontent.manning.com/deep-learning-for-text/

4.2 Preprocessing

Our first step is to collect the texts. As our texts are in the public domain, they are easily accessible on public library websites like Project Gutenberg[1], and are widely available in .txt format. This is a plaintext format that is easy to work with.

4.2.1 Basic Preprocessing

Once we have downloaded our texts we may need to remove unwanted sections like licensing/publishing details, prefaces, introductions, footnotes, etc. This is a time-intensive task that is difficult to automate. The key point is that we need to be using only the author's words.

4.2.2 Tokenisation

The next step in preprocessing our data is that of **tokenisation**. Tokenisation is the process of turning unstructured text into a structured data structure that we can run our algorithms on. This process is shown in Figure 4.1. The tokeniser takes an input sentence and breaks it up into its constituent tokens (individual words or punctuation) which allows much faster processing of the data of the text (e.g. counting the occurences of specific words. It is regarded as an NLP task.

We might want our tokeniser to format the tokens it creates. Some operations we may perform on the input text could be to change all words to lowercase, or to remove punctuation entirely.

4.2.3 Stemming

We may also want to stem the words of our texts. **Stemming** is when we remove the inflections of words to reduce them to their lexical stem. An example can be seen in Figure 4.2.

Stemming may not be appropriate for every task. One good use case for stemming is content analysis; this is where we try to categorise or classify

[1]www.gutenberg.org/

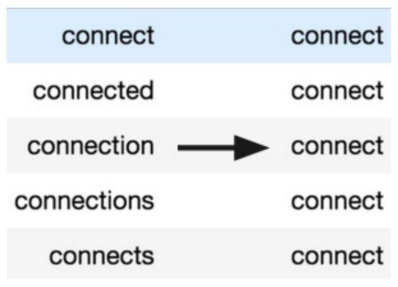


Figure 4.2: Using a Porter Stemmer on various forms of the word 'Connect'

documents by their content. For example, if we were trying to categorise if a document was discussing swimming or cycling, we could convert every inflection of 'swim' (e.g. 'swimming', 'swimmer', 'swims') to 'swim'. After stemming we would simply be able to count every instance of 'swim' and compare it to, for example, the number of instances of 'cycl' (the stem of 'cycle'); if 'swim' appeared more often we would classify this document as discussing swimming, and vice versa.

Luckily, there are many pre-built stemmers, that automate this task for us, available for most programming languages, such as the Porter Stemmer[2]. However it is a nontrivial task, and mistakes are often made by stemming algorithms, so careful use is required.

4.3 Extracting Features

The next step in our process is to extract the desired textual features from our structured, preprocessed data. These features are also aptly called **style markers**.

The purpose of feature extraction is to convert our texts into mathematical representations, which we can then use in statistical analyses. Figure 4.3 gives us an interpretation of feature extraction; 'Jane Eyre' is converted into a vector, where each value is the frequency of a different function word like 'at' or 'be'. (Note: these values aren't the correct values). This vector then represents the style of 'Jane Eyre'. After feature extraction, we can use vectors like these in our machine learning algorithms.

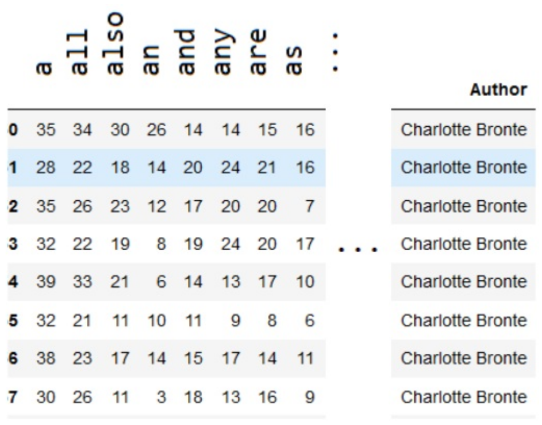
at	403
be	1043
been	759
but	327
by	935
can	366
do	755
down	928
even	571
every	220
for	856
from	740
...	...

Figure 4.3: Turning Jane Eyre into a vector
Source: Adapted from tinyurl.com/k3pu8c6f

Our goal is to create a dataset that looks like the one in Figure **??**, where we have a series of short chunks of Charlotteë's writing (the rows), and the column values are the frequencies of each function word in the specific chunk. At the end of our vectors we have the author label.

Next, we discuss what kind of features we might extract from our texts. Remember, our goal is to attain features that will allow us to differentiate between authors' styles. Of

[2]tartarus.org/martin/PorterStemmer/



	a	all	also	an	and	any	are	as	...	Author
0	35	34	30	26	14	14	15	16		Charlotte Bronte
1	28	22	18	14	20	24	21	16		Charlotte Bronte
2	35	26	23	12	17	20	20	7		Charlotte Bronte
3	32	22	19	8	19	24	20	17	...	Charlotte Bronte
4	39	33	21	6	14	13	17	10		Charlotte Bronte
5	32	21	11	10	11	9	8	6		Charlotte Bronte
6	38	23	17	14	15	17	14	11		Charlotte Bronte
7	30	26	11	3	18	13	16	9		Charlotte Bronte

Figure 4.4: A dataset of chunks of Charlotte Brontë's work
Source: Adapted from tinyurl.com/k3pu8c6f

course, the domain of our texts affect what are good choices for features, think novels vs text messages.

4.3.1 Lexical features

Lexical features are those which use the words (or tokens) of our text. Using lexical features has often been the classic approach to stylometry. They are easily to process (for example, simply use spaces to split up every line of text into words), require minimal preprocessing vs other features, they are often used unconsciously, give easily interpretable datasets. They are also often used in conjunction with other features, as they are rarely a bad choice. They can also generally be applied to any language, however languages like Chinese are a problem due to the difficulty in tokenisation.

- **Sentence length** features include the average number of words in a sentence, or the frequency distribution of sentence lengths similar to the 'characteristic curves' seen in Section 3.3.1.
- **Word length** features include the average word length in a document or section of a document, the frequency distribution of word lengths (the 'characteristic curves'), the average number of syllables in a word, or indeed the frequency distributions of syllabes per word.
- **Word frequency** features often use the frequency of a set of function words, also known as stopwords, that are grammatical or commonly used words like 'the' or 'and'. For other tasks like content or sentiment analysis (analysing what a document is about, or if it is negative or positive respectively), much larger content-specific sets of words can be used. Using word frequencies is known as a 'bag-of-words' model.
- **N-grams** are *n*-long sequences, traditionally sequences of words. For n=2, these are things like 'I will', 'and the', or 'hot dog'. 1-grams, 2-grams, and 3-grams are called 'unigrams', 'bigrams', and 'trigrams' respectively, however

1-grams are the same as our word frequency measures. When we increase n, we capture more context (for example, a sentence with the words 'hot' and 'dog' has a very different meaning when these words are used together vs separately). While capturing more context is often desirable, that data we generate will be more sparse.

- **Vocabulary richness** features attempt to capture how wide an author's vocabulary is. This is usually normalised in some way with respect to the length of the particular document.

4.3.2 Character Features

Character features use the individual letters, or characters, of our texts.

- **Character frequency** features (for English) measure the distribution of use of the 26 letters of the alphabet. This could be used to distinguish an author who uses far more words with 'z' in them than an author who doesn't. However, other languages may have different alphabets, and some don't have an alphabet at all. In general, unfortunately, features for English are the best explored in the literature, and stylometry for e.g. Asian languages is underdeveloped.
- **Character n-grams** are simply n-long sequences of characters. These will be common short words, parts of common words, and common prefixes/suffixes (e.g. 'th', 'he', 'an', 'nd' for 2-grams, and similarly 'the', 'and', ' th', and 'ing' for 3-grams).
- **Punctuation use** features could be used to determine if, for example, one of the candidate authors uses a lot of semicolons, while one never uses them.

4.3.3 Syntactic Features

Syntactic features capture the grammatical aspects of writing. The goal of using syntactic features is to differentiate between authors based on the way they structure their ideas/sentences, and are probably used more unconsciously than e.g. words. The success of function words implies these features will be successful as function words are often dependent on grammatical structures.

Generating syntactic features requires sophisticated NLP tools, and it is a far more computationally intensive task. It is known as **parsing** - generating the grammatical structure of a sentence given only the words of the sentence. We need to be able to trust our NLP tools as accuracy is difficult to achieve. The use of **treebanks**, large corpora of human-annotated texts, allows the large scale training of parsers, and, luckily, many pre-built parsers are freely available to use.

We will explain the syntactical features we could possibly use with a sample parse tree. We look at potential parses of sentence 'I shot an elephant in my pyjamas' in Figure 4.5. The two interpretations we see are:

1. 'I shot an elephant while I was wearing my pyjamas'

2. 'I shot an elephant that was wearing my pyjamas'

This shows ambiguity is a big problem for NLP tasks like parsing. An important note is that there are many theories of grammar, and the parse trees generated for each can look quite different. We only discuss features for the *phrase structure grammar* (PSG) that gives us *constituency-based parse trees* as seen in the figure. A common alternative is dependency-based grammars and parse trees

Possible syntactic features are summarised below:

- The **part of speech** (POS) of a word captures its grammatical role in the sentence or phrase. These are the symbols (called POS *tags*) immediately above the words of the sentence in Figure 4.5. The ones seen here are N = noun, V = verb, Det = determiner, P = preposition. Others include adjectives, pronouns, adverbs, however these are sometimes language-specific. The NLP tool that creates these tags is called a *POS tagger*.

- We can also use the different **types of phrases** seen in an author's work. Phrases are groups of words that act as a single unit in the syntax of a sentence; they can also be called a **constituent**. We note that POSs can be thought of as phrases, and, also, phrases can be combined as we see in 4.5. In our example we have the phrases S = sentence, NP = noun phrase like 'an elephant', VP = verb phrase like 'shot an elephant', PP = preposition phrase like 'in my pyjamas'.

- Another possible feature is the **phrases per sentence** of an author, which would be roughly proportional to mean sentence length.

- Similarly, the average **length of phrases** could be used. This could be the mean length of the different phrase types as seen in the text, or the frequency distribution of the lengths of the different phrase types.

- **L**astly, we could look at the ordering of phrases. In natural language there is often some choice in the order we put our phrases. For instance, we could rephrase our example sentence as 'In my pyjamas I shot an elephant', which has the same meaning as Sentence 1 but moves the prepositional phrase into the NP before 'I'. This is very much a part of an author's style (think Yoda in 'Star Wars). This behaviour is captured in **Phrasal n-grams**; these are n-long sequences of phrases, e.g. 'NP VP' or 'Det N'.

4.3.4 Semantic, Pragmatic, and Discourse Features

Semantic features of text are about meaning that is 'coded' in the language, e.g. two words are synonyms, while discourse concerns semantics beyond individual sentences. On the other hand, pragmatic features combine the understanding of meaning from a structural and grammatical perspective with an understanding from the context - general knowledge about the world, and the motivation of the text itself.

We could:

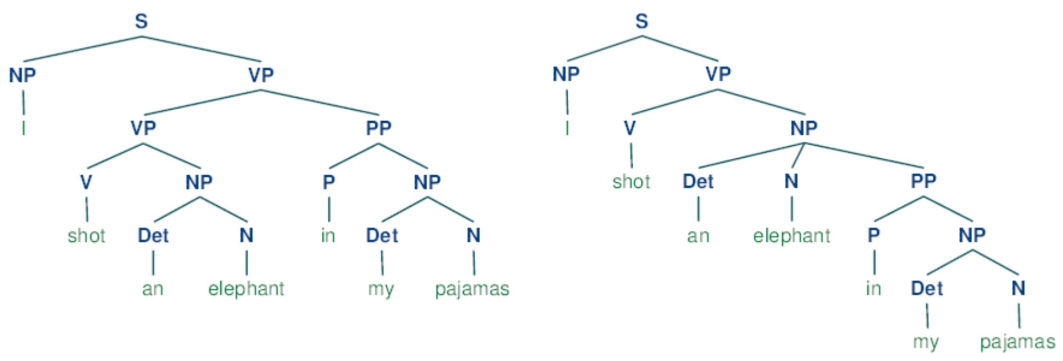


Figure 4.5: Two conflicting parse trees for the sentence 'I shot an elephant in my pyjamas'
Source: Adapted from www.nltk.org/book/ch08.html[1]

- Look at the meanings of words (e.g. use of synonymns, hypernyms, etc.).
- Perform sentiment or content analyses.
- Examine the use of dialogue in texts.
- Measure how often characters or places are referred to.

Higher level features like this are even harder to generate accurately, and often humans will get these meanings wrong. These are very hard tasks that remain generally unsolved. For word meanings, **word embeddings** have become popular - these are vectors that represent the meaning of words - especially in tasks like state-of-the-art machine translation. However, things representing phrase and sentence embeddings don't really exist yet.

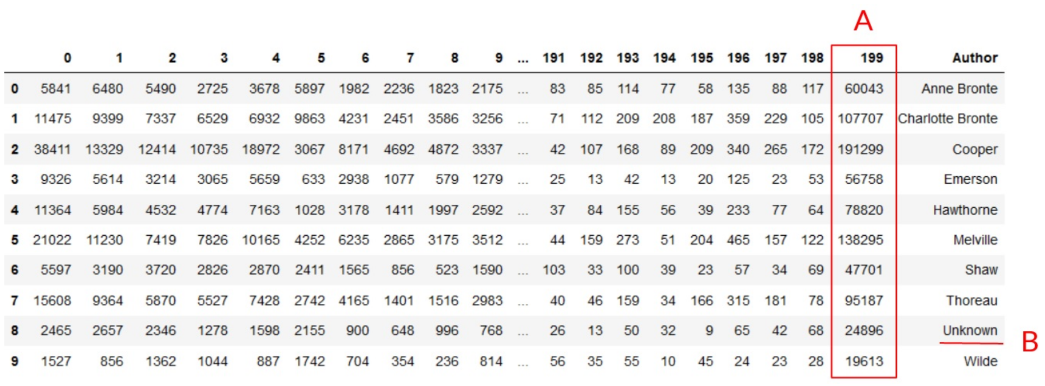
4.3.5 Feature Selection

There are many possible feature sets to choose from, in fact the number of style markers that have been suggested was approximately 1,000 back in 1997 [15]. We have the option of combining features (e.g. lexical and character). A common practice in stylometry is to trim our feature sets to leave only those that best differentiate the authors of our dataset.

4.3.6 What Next?

There are lots of style markers to choose from, but for our attempt at classifying 'Agnes Grey', we will use two sets of features:

1. The 200 most common words in the dataset. This are mostly function words. Simple feature sets are often the best, and, as we will see, we get surprisingly good results.
2. The 200 most common character 3-grams. These are things like 'the', ' th', 'and', 'nd ', and 'ing'. These are things like common short words, parts of common words, and common prefixes/suffixes.



	0	1	2	3	4	5	6	7	8	9	...	191	192	193	194	195	196	197	198	199	Author
0	5841	6480	5490	2725	3678	5897	1982	2236	1823	2175	...	83	85	114	77	58	135	88	117	60043	Anne Bronte
1	11475	9399	7337	6529	6932	9863	4231	2451	3586	3256	...	71	112	209	208	187	359	229	105	107707	Charlotte Bronte
2	38411	13329	12414	10735	18972	3067	8171	4692	4872	3337	...	42	107	168	89	209	340	265	172	191299	Cooper
3	9326	5614	3214	3065	5659	633	2938	1077	579	1279	...	25	13	42	13	20	125	23	53	56758	Emerson
4	11364	5984	4532	4774	7163	1028	3178	1411	1997	2592	...	37	84	155	56	39	233	77	64	78820	Hawthorne
5	21022	11230	7419	7826	10165	4252	6235	2865	3175	3512	...	44	159	273	51	204	465	157	122	138295	Melville
6	5597	3190	3720	2826	2870	2411	1565	856	523	1590	...	103	33	100	39	23	57	34	69	47701	Shaw
7	15608	9364	5870	5527	7428	2742	4165	1401	1516	2983	...	40	46	159	34	166	315	181	78	95187	Thoreau
8	2465	2657	2346	1278	1598	2155	900	648	996	768	...	26	13	50	32	9	65	42	68	24896	Unknown
9	1527	856	1362	1044	887	1742	704	354	236	814	...	56	35	55	10	45	24	23	28	19613	Wilde

Figure 4.6: 200 function word style vectors for the 'Agnes Grey' dataset's authors

Remember, our task is to find the author of 'Agnes Grey'. Firstly, we will try a profile-based (one vector per author) approach using the 200 most common function words. Until now, we have:

1. Read in (loaded) the preprocessed authors' files.
2. Found the 200 most common words across all the texts.
3. Counted the occurrences of each function word in every author's work which we save in a table.

The end result of this is shown in Figure 4.6. We have a dataset where the rows are our different authors' style vectors, and the columns give the frequency of a particular function word in author's texts. The column shown by label A is the total frequency of words that are not in the previous function words - this allows us to normalise by the length of the document. The author shown by the label B is that of 'Agnes Grey'.

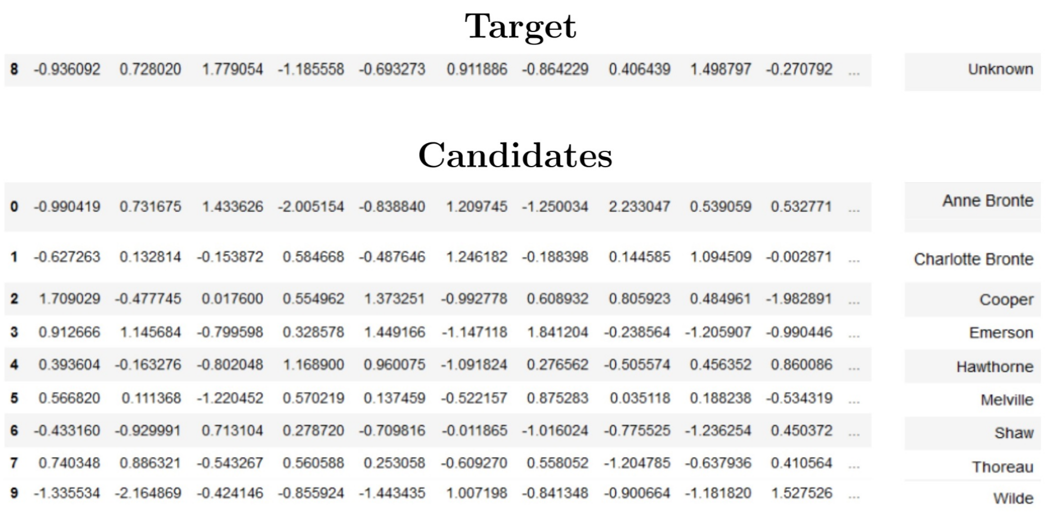
After we have completed our analysis, we will repeat it with a 200 character 3-gram feature set. Following this, we will use the 3-gram dataset for a train/test validation of our instance-based methods. Then we will do some instance-based classification with short chunks (approximately 500 words) of writing.

4.3.7 Scaling and Normalising Data

The final step of feature extraction is scaling and normalising our data, luckily programming libraries (like 'numpy' for Python) allow easy vector and matrix manipulation. Our goal is to account for length of authors texts and to improve overall performance. We also get a dataset that is more statistically interpretable. We let i and j index the rows and columns respectively, x_{ij} be the entries in our dataset, and finally μ_j and σ_j be the mean and standard deviation of column j respectively.

By dividing each row by their sum, we normalise by length of document. Scaling:

$$x_{ij} = \frac{x_{ij}}{\sum_{j=1}^{n} x_{ij}} \tag{4.1}$$



Target

8	-0.936092	0.728020	1.779054	-1.185558	-0.693273	0.911886	-0.864229	0.406439	1.498797	-0.270792	...	Unknown

Candidates

0	-0.990419	0.731675	1.433626	-2.005154	-0.838840	1.209745	-1.250034	2.233047	0.539059	0.532771	...	Anne Bronte
1	-0.627263	0.132814	-0.153872	0.584668	-0.487646	1.246182	-0.188398	0.144585	1.094509	-0.002871	...	Charlotte Bronte
2	1.709029	-0.477745	0.017600	0.554962	1.373251	-0.992778	0.608932	0.805923	0.484961	-1.982891	...	Cooper
3	0.912666	1.145684	-0.799598	0.328578	1.449166	-1.147118	1.841204	-0.238564	-1.205907	-0.990446	...	Emerson
4	0.393604	-0.163276	-0.802048	1.168900	0.960075	-1.091824	0.276562	-0.505574	0.456352	0.860086	...	Hawthorne
5	0.566820	0.111368	-1.220452	0.570219	0.137459	-0.522157	0.875283	0.035118	0.188238	-0.534319	...	Melville
6	-0.433160	-0.929991	0.713104	0.278720	-0.709816	-0.011865	-1.016024	-0.775525	-1.236254	0.450372	...	Shaw
7	0.740348	0.886321	-0.543267	0.560588	0.253058	-0.609270	0.558052	-1.204785	-0.637936	0.410564	...	Thoreau
9	-1.335534	-2.164869	-0.424146	-0.855924	-1.443435	1.007198	-0.841348	-0.900664	-1.181820	1.527526	...	Wilde

Figure 4.7: Normalised target and candidate vectors for the 200 function word 'Agnes Grey' dataset

or in vector notation:

$$\mathbf{x_i} = \frac{\mathbf{x_i}}{||\mathbf{x_i}||} \tag{4.2}$$

We standardise columns by subtracting their mean and dividing by the standard deviation. This is a standard ML practice that improves the performance of ML algorithms. Normalisation:

$$z_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j} \tag{4.3}$$

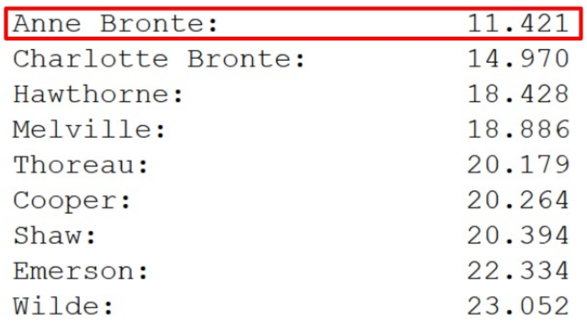
We finally have our profile-based function word dataset, which is shown in Figure 4.7. This consists of z-scores of each author's use of the function words. This is interpretable on the table, for example we notice James Fenimore Cooper uses the word 'the' (the leftmost column as it is the most frequent word) 1.7 standard deviations above the mean, while Oscar Wilde uses it 1.3 below.

4.4 Basic Profile-Based Algorithm

We now have our features. For our profile-based setting, our approach to classification is very simple. We have a target vector - the representation of the author of 'Agnes Grey'. We also have a set of candidate vectors - these are our potential authors. Both are shown in Figure 4.7.

Of the eight candidate authors, who is most likely to be our target author? As a mathematician, the intuitive solution could be to calculate which candidate's style vector is closest to the target's style vector. In a sense, this will calculate their similarity, as distance is a measure of the difference between vectors. We have effectively created a *vector space* of style vectors.

We summarise a basic solution below, and present the results in Figure 4.8:



```
Anne Bronte:                 11.421
Charlotte Bronte:            14.970
Hawthorne:                   18.428
Melville:                    18.886
Thoreau:                     20.179
Cooper:                      20.264
Shaw:                        20.394
Emerson:                     22.334
Wilde:                       23.052
```

Figure 4.8: The Euclidean distances between the target vector and each candidate vector in the 200 function word 'Agnes Grey' dataset

Basic Solution

1. Calculate the Euclidean distance between the target vector **t** and each candidate vector **c**. Euclidean distance is discussed in Section 4.5.1, and its equation is (4.5).

2. Rank the candidates from closest to furthest.

3. Choose the closest candidate vector as the author of 'Agnes Grey'. We see this is Anne Brontë.

Anne Brontë is, perhaps unsurprisingly, the closest author, which is the correct answer, and in her sister Charlotte is the second closest.

We can extend this idea to a general solution for profile-based authorship attribution, where we need only to choose some measure of distance:

General Solution

1. Calculate the **distance** between the target vector **t** and each candidate vector **c**.

2. Rank the candidates from closest to furthest.

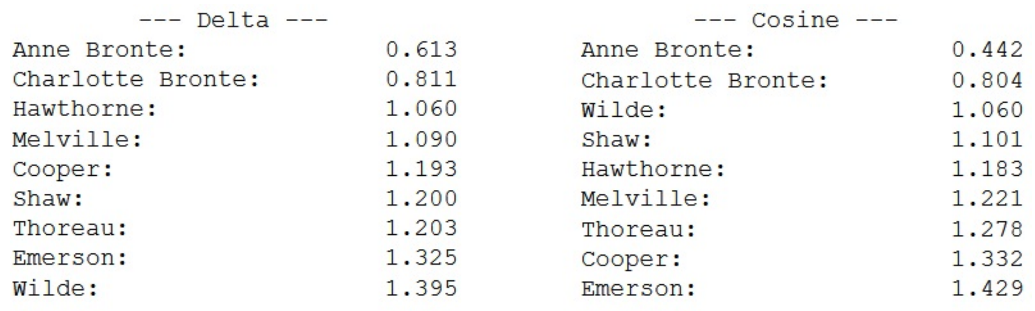
3. Choose the closest candidate vector as the author of our unseen/disputed work.

This approach is equivalent to choosing the nearest neighbour to the target vector. This is the k-nearest neighbours algorithm (discussed in 4.7.1) for k = 1.

We summarise this approach in the following equation:

$$author(\mathbf{t}) = \arg\min_{\mathbf{c}\in C} d(\mathbf{c}, \mathbf{t}) \tag{4.4}$$

Of course, Euclidean distance has several problems, notably that it is sensitive to extreme values, however we find that is works well. It is good to try different methods too. Burrow's Delta (cite), discussed in Section 4.5.1 and defined in Equation 4.10, is a very popular and respected distance measure in stylometry.



```
         --- Delta ---                            --- Cosine ---
Anne Bronte:                0.613        Anne Bronte:               0.442
Charlotte Bronte:           0.811        Charlotte Bronte:          0.804
Hawthorne:                  1.060        Wilde:                     1.060
Melville:                   1.090        Shaw:                      1.101
Cooper:                     1.193        Hawthorne:                 1.183
Shaw:                       1.200        Melville:                  1.221
Thoreau:                    1.203        Thoreau:                   1.278
Emerson:                    1.325        Cooper:                    1.332
Wilde:                      1.395        Emerson:                   1.429
```

Figure 4.9: Delta and Cosine distances from the target vector to each candidate vector in the 200 function word 'Agnes Grey' dataset

The cosine similarity (the cosine angle between vectors) is another commonly used similarity measure.

We summarise the results of using Delta and Cosine measures in Figure 4.9, which again show Anne Brontë as the true author.

4.5 Profile-Based Algorithms

We see we can attribute authorship using profile vectors purely by choosing the nearest candidate. However there are many ways to calculate distance, and these can vastly change the results.

We summarise some common methods of measuring distance and similarity in the following section. We let $d(\mathbf{c}, \mathbf{t})$ measure the distance between a candidate vector $\mathbf{c}$ and target vector $\mathbf{t}$.

4.5.1 Measuring Distance and Similarity

Euclidean Distance

This is the 'straight line' distance between two points. The Euclidean distance between candidate vector $\mathbf{c}$ and target vector $\mathbf{t}$ is

$$d(\mathbf{c}, \mathbf{t}) = \sqrt{\sum_{j=1}^{n} (c_j - t_j)^2} \qquad (4.5)$$

Manhattan Distance

The 'Manhattan', or 'taxicab', distance is the distance between two points measured along axes at right angles.

$$d(\mathbf{c}, \mathbf{t}) = \sum_{j=1}^{n} |c_j - t_j| \qquad (4.6)$$

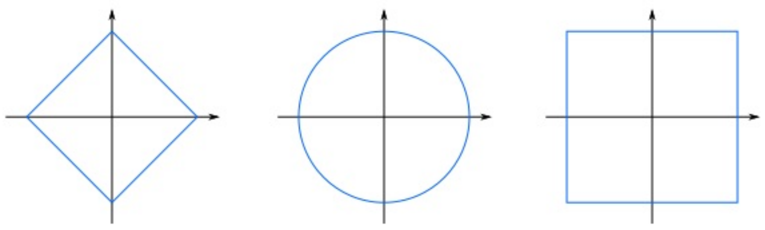


Figure 4.10: Unit circles based on Manhattan (1-Norm), Euclidean (2-Norm), and Chebyshev (∞-norm) distance measures
Source: Adapted from wikipedia.org/wiki/Minkowski_distance

Chebyshev

The Chebyshev distance is the greatest difference along any coordinate dimension for two points.

$$d(\mathbf{c},\mathbf{t}) = max_j\left|c_j - t_j\right| = max(\left|c_1 - t_1\right|, \left|c_2 - t_2\right|, ..., \left|c_n - t_n\right|) \tag{4.7}$$

Minkownski Distance

The Minkowski distance generalises all previous distance measures using a parameter p that is at least 1. It is also called the p-norm, and the Minkowski distance measure of order p is also called a p-norm: for example 1-norm is Manhattan, and 2-norm is Euclidean, while it can be shown that the ∞-norm is the Chebyshev distance (it equals equals $\lim_{p\to\infty}(\sum_{j=1}^{n}\left|c_j - t_j\right|^p)^{1/p}$). The three p-norms so far are shown in Figure 4.10.

$$d(\mathbf{c},\mathbf{t}) = (\sum_{j=1}^{n}\left|c_j - t_j\right|^p)^{1/p} \tag{4.8}$$

Cosine Similarity

While cosine similarity is not strictly a measure of distance, it can be used for similar purposes. It is derived from the Euclidean dot product formula.

$$similarity(\mathbf{c},\mathbf{t}) = cos(\Theta) = \frac{\mathbf{c}\cdot\mathbf{t}}{\|\mathbf{c}\|\,\|\mathbf{t}\|} = \frac{\sum_{j=1}^{n} c_j t_j}{\sqrt{\sum_{j=1}^{n} c_j^2}\sqrt{\sum_{j=1}^{n} t_j^2}} \tag{4.9}$$

Delta

Burrow's Delta is a very common distance measure in stylometry. It effectively calculates the average difference in z-scores for each column value.

$$\Delta_c = \sum_{j=1}^{n}\frac{(c_j - t_j)}{n} \tag{4.10}$$

We now discuss how we will visualise this high dimensional data.

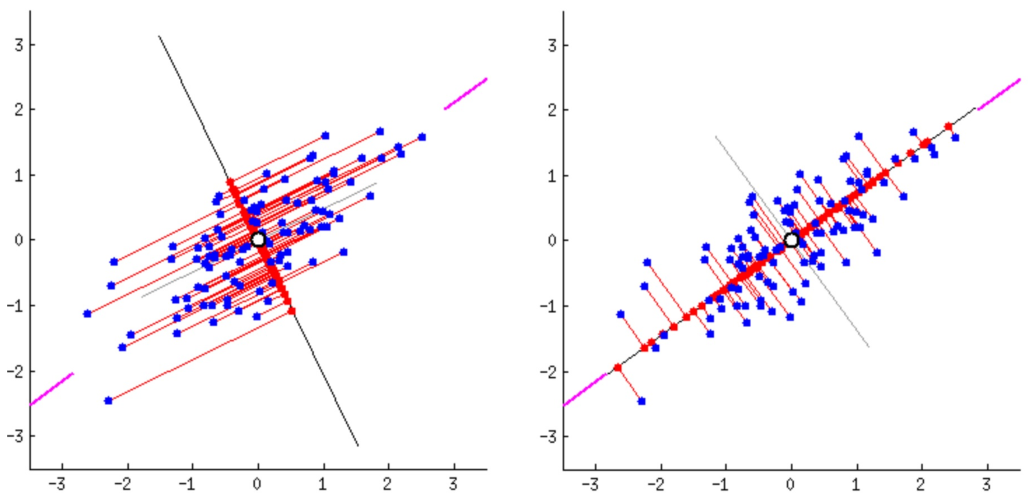


Figure 4.11: Projection of 2-d dataset to 1-d using two different principal components
Source: tinyurl.com/aauuyp75

4.5.2 Visualisation with Principal Component Analysis

Principal component analysis (PCA) is a dimensionality reduction technique. It works by choosing axes that maximise the variability of data when projected; it captures most of the information in the data but using fewer dimensions. For example, we can use PCA to reduce our high-dimensional feature sets down to two dimensions without losing lots of variability, which allows us to plot the data on a graph, giving us a visual interpretation of our data which is very useful.

Figure 4.11 gives us an insight into how PCA works for projecting 2-dimensional data down to a single dimension, where we project the data points in blue onto the black line. The projection is shown in red. We observe the line on the right graph captures far more variability than that on the left. The 1-dimensional subspace that captures most variability passes through the purple markers. As we are in two dimensions, the second principal component would be on the grey line orthogonal to our black line - we have no choice here.

However, in higher dimensions, the next principal component is a vector orthogonal to all previous components, and the one that maximises the remaining variability. In Figure 4.12 the given line is our first principal component, and our second will be the vector orthogonal to this line (on the plane) that maximises variance when the points are projected onto the plane defined by these two principal components.

How do we calculate the principal components for a dataset? First, we need to discuss variance and covariance. We let X, Y be random variables, and μ_X, μ_Y be the mean, or expected value, of X and Y respectively.

Variance is a measure of how far spread the values of a random variable are from its mean value. It is defined in Equation 4.11 as the expected value of the squared deviation from the mean.

Covariance, on the other hand, measures the joint variability of two random variables. It is a measure of how well correlated they are - if they tend to increase

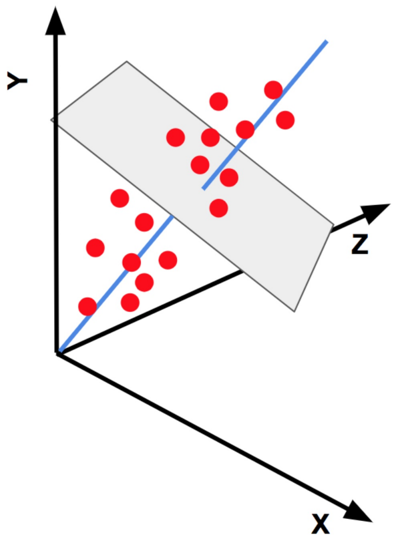


Figure 4.12: Choosing the 2nd principal component for a 3-d dataset
Source: learnopencv.com/principal-component-analysis/

or decrease together, the covariance will be positive, if one tends to increase when the other decreases, the covariance will be negative, and if they are uncorrelated, the covariance will be zero. It is defined in Equation 4.12 as the expected value of the product of the two random variables' deviations from their expected value. Variance can also be thought of as the covariance of a random variable with itself.

Covariance matrices generalise the idea of variance to multiple dimensions. They are square matrices where the diagonal elements X_{ii} are the variances of the ith dimension (in our dataset this is the ith column of function word frequencies), and the non-diagonal elements X_{ij} are the covariances of the ith and jth dimensions. Thus they are also symmetric matrices as $cov(X, Y) = cov(Y, X)$.

$$Var(X) = E[(X - \mu_X)^2] \tag{4.11}$$

$$cov(X, Y) = E[(X - E[X])(Y - E[Y])] \tag{4.12}$$

The principal components are actually the eigenvectors of the covariance matrix. As the covariance matrix is a square symmetric matrix, it can always be diagonalised due to the Spectral Theorem (from Honours Algebra). This diagonalised matrix will have the eigenvalues along its diagonal and zeroes in every other entry. We can rearrange the diagonal entries from largest to smallest, and, in fact, these values are the variances captured by each principal component.

Also, **scree plots** show the variance in the data captured by successive principal components. This lets allows us to see if we are adequately capturing the distribution of our dataset. The proportion of total variance captured by a principal component is its corresponding eigenvalue divided by the sum of all eigenvalues. An example scree plot is 4.13, however we omit them and instead include the variance captured by each principal component on its axis in the corresponding graph.

In summary we:

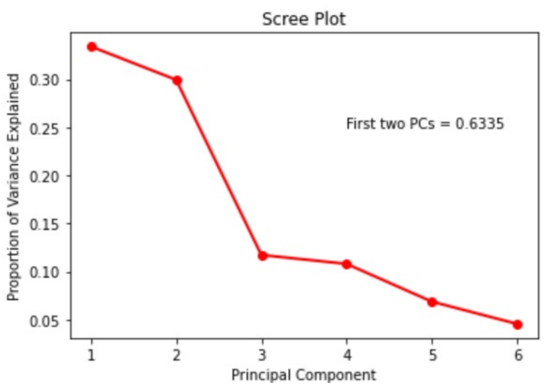


Figure 4.13: An example scree plot including the first 6 principal components

1. Calculate principal components using the covariance matrix of the dataset.
2. Project the data onto our new axes.
3. Plot the results.

We show the results of our PCA in the following section.

4.6 Evaluating Profile-Based Results

We use PCA for both the 200 function word dataset, and the 200 character 3-gram dataset.

4.6.1 200 Function Word PCA

We see in Figure 4.14 that although the ‘Agnes Grey’ style vector is closest the Anne Brontë, it is also quite close to Charlotte Brontë (which we would expect given our distance results). We also notice we are capturing over 50% of the variance of our dataset. This is good considering it is 70-dimensional. We can get a clearer picture by removing some authors (Cooper, Emerson, Shaw, and Thoreau) and by performing PCA again. This is shown in Figure 4.15. This shows Anne Bronte still very close, while Charlotte has been moved further, and we are getting almost 70% variance.

4.6.2 200 character 3-gram Distances

We now summarise the distances between the ‘Agnes Grey’ style vector and all candidate style vectors using the 200 most common 3-grams in the dataset. Reminder: this is the 1-NN algorithm as discussed in Section 4.7.1. We summarise the distance results in Figure 4.16, which again shows Anne Bronte as consistently the closest, and an especially stark similarity using the Cosine measure.

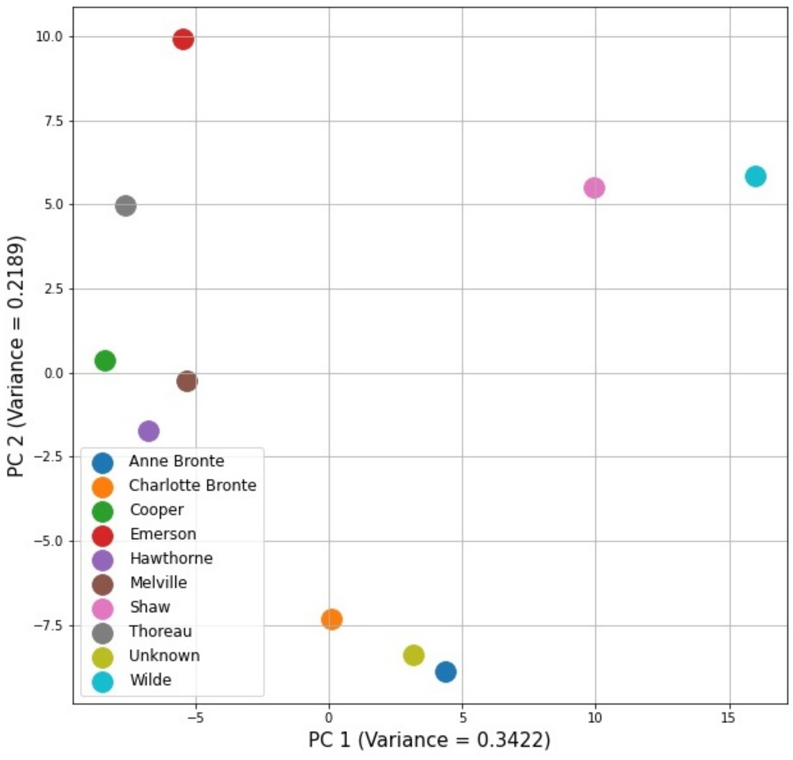


Figure 4.14: PCA with all authors using 200 function words

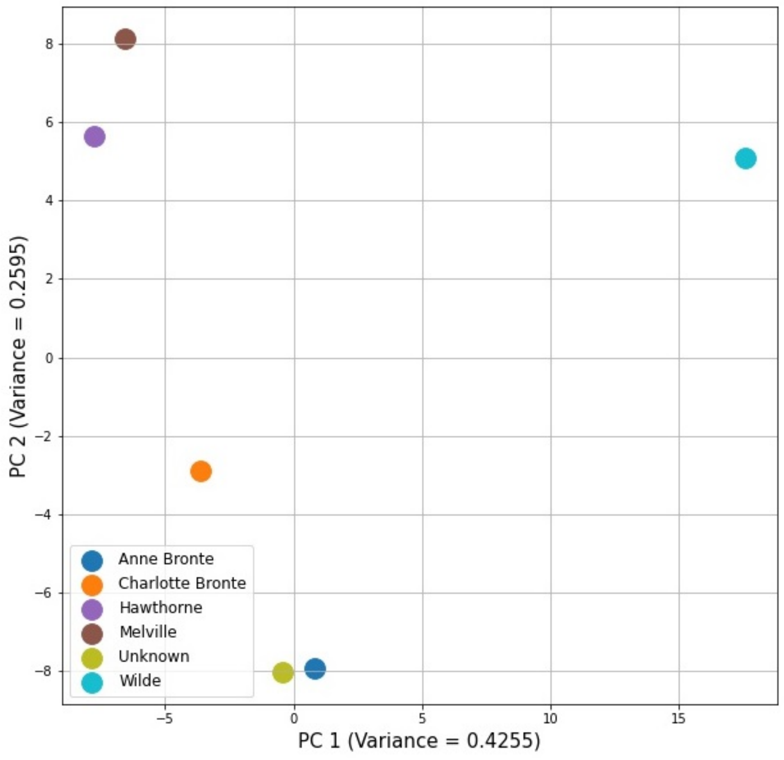
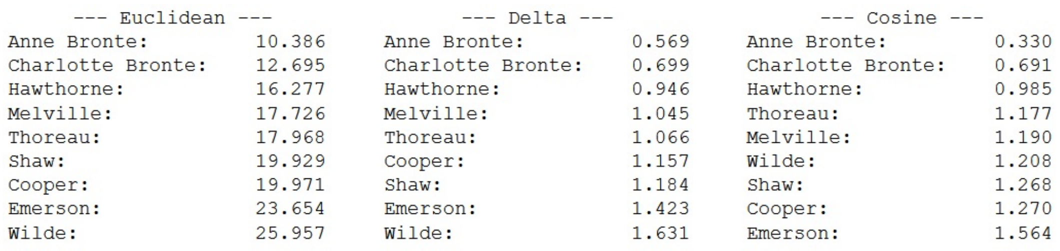


Figure 4.15: PCA (with four authors removed) using 200 function words



```
      --- Euclidean ---                     --- Delta ---                       --- Cosine ---
 Anne Bronte:          10.386    Anne Bronte:          0.569    Anne Bronte:          0.330
 Charlotte Bronte:     12.695    Charlotte Bronte:     0.699    Charlotte Bronte:     0.691
 Hawthorne:            16.277    Hawthorne:            0.946    Hawthorne:            0.985
 Melville:             17.726    Melville:             1.045    Thoreau:              1.177
 Thoreau:              17.968    Thoreau:              1.066    Melville:             1.190
 Shaw:                 19.929    Cooper:               1.157    Wilde:                1.208
 Cooper:               19.971    Shaw:                 1.184    Shaw:                 1.268
 Emerson:              23.654    Emerson:              1.423    Cooper:               1.270
 Wilde:                25.957    Wilde:                1.631    Emerson:              1.564
```

Figure 4.16: Distance results for the 'Agnes Grey' dataset using 200 character 3-grams

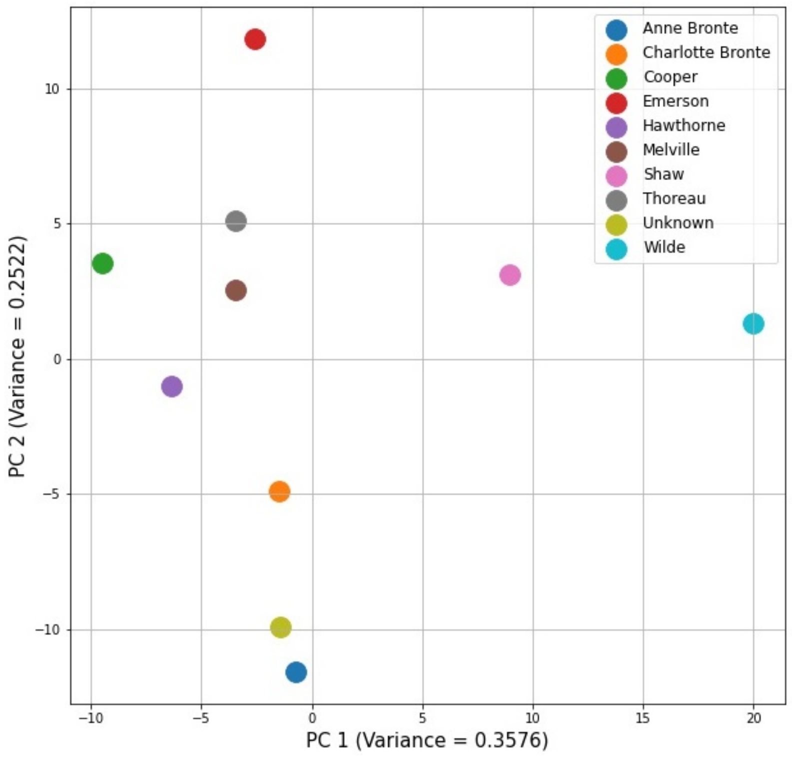


Figure 4.17: PCA with all authors using 200 character 3-grams

4.6.3 200 character 3-gram PCA

We again perform PCA however using the 200 character 3-gram vectors. This is shown in Figure 4.17. We see it is very similar to the function word PCA, and for this reason we omit the PCA graph using less authors.

4.6.4 Discussion

We now have very strong evidence that Anne Bronte is the author of 'Agnes Grey' (phew). We see the success of our three distance measures, and also that PCA works very well for these kinds of datasets.

Now that we have successfully found the correct author, we turn to the problem of instance-based methods.

We introduce our algorithms - k-nearest neighbours, and support vector ma-

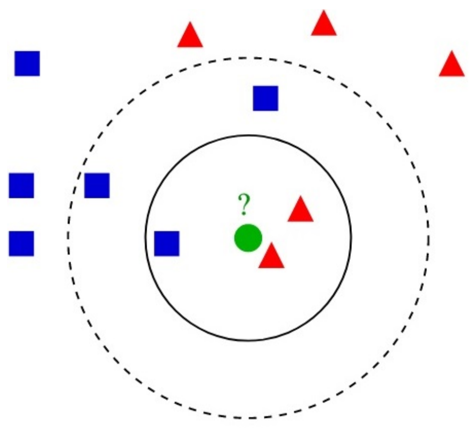


Figure 4.18: A simple k-NN classification using k = 3 (unbroken line) and k = 5 (dotted line)
Source: https://tinyurl.com/p7fhbap4

chines. We then split the authors' writing up into short chunks (500 words). First, we segment the entire dataset (including 'Agnes Grey' as an Anne Bronte work) into a train/test split (80%/20%) in order to validate our methods. We then see how well our new classification algorithms work for attributing 'Agnes Grey'.

4.7 Instance-Based Algorithms

We use two different instance-based classification algorithms - k-nearest neighbours, and support vector machines - which we now introduce.

4.7.1 K-Nearest Neighbours

The k-nearest neighbours (k-NN) algorithm is a very popular and well established technique for performing classification with lots of training examples, and it is actually quite simple to understand. It can be used for regression too, but our focus is on classification.

We show how k-NN works using a simple example. Imagine we are trying to classify the green circle in Figure 4.18, where the possible classes are the blue squares and the red triangles. We find the k closest neighbours and choose the class that is most common among them. For example, if k = 3 (bounded by the unbroken circle in the figure), then the green circle would be classified as a red triangle, however if k = 5 (the dotted circle), then it would be classified as a blue square. Trouble arises when k = 4, as we would have a tie. In cases like this we have a number of options, however the most common is to randomly choose between the classes that are tied for the most neighbours.

No training is required for k-nn, we only need to load in our data. Different measures of distance are of course possible, however, in early tests, we found the results using both the 'Manhattan' and 'Delta' distances were very similar to those with the standard Euclidean distance, and chose to only use Euclidean.

Normalising our data also greatly improved the performance of our k-nn algorithm. One interesting technique we use is the distance weighting scheme.

Distance Weighting

We can weight the neighbours of a point by their distances away. Rather than each of the k neighbours contributing equally to the decision of the classifier, they can be weighted according to a weighting scheme - typically each is given a weight 1/d, where d is the distance from the neighbour to the point being classified. Then, the point is classified to the class with the highest combined weights. We compare all our standard k-nn results to distance weighted results

4.7.2 Support Vector Machines

Support vector machines (SVMs) are a very popular supervised[3] machine learning model for both classification and regression, however they can also be used for unsupervised learning. SVMs work by constructing hyperplanes that best separate two classes - it is a binary linear classifier - however this process can be extended to cover as many classes as needed. When a new point is to be classified, the SVM simply checks which side of the hyperplane it is on and assigns it the associated class.

Figure 4.19 shows data points belonging to two classes, white and black circles, and three possible hyperplanes (the lines H1, H2, H3 here) to separate them. We notice H1 fails to separate the classes, whereas H2 and H3 do, however H3 separates the classes with a far bigger **margin** (the space between the hyperplane and the nearest point from either class), and is likely to classify unseen data more accurately. This is the separating hyperplane an SVM would choose for this dataset - it is the hyperplane that maximises the margin, or distance to the closest point on both sides. In fact the SVM boundary is defined purely by these closest points from both classes - this is shown by the grey lines from both H2 and H3 to the closest points from both classes. These are known as the **support vectors**, from where the support vector machine gets its name. When the data is not linearly separable, which is probable in high dimensional spaces with lots of data points, SVMs can still work - points on the wrong side of the separating hyperplane are punished proportional to their distance to the hyperplane and become support vectors. However, the 'kernel trick' is often used to effectively construct non-linear decision boundaries.

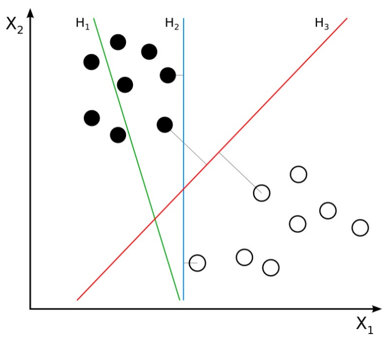


Figure 4.19: Three possible separating hyperplanes (H1, H2, H3) for a 2-d dataset with two classes
Source: https://tinyurl.com/yvk62bxe

[3]meaning it requires labelled training data like our labelled author profiles

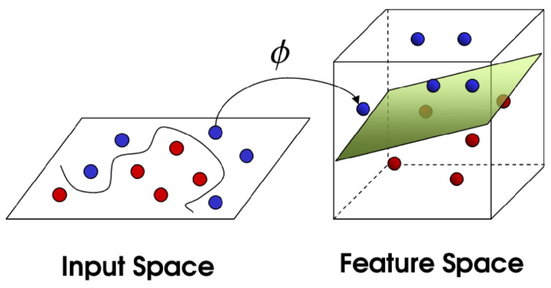


Figure 4.20: A illustration of the 'kernel trick' for a 2-d input space
Source: https://tinyurl.com/2fmxxvr3

Kernel Trick

The kernel trick is a ingenious method of using a higher dimensional space that an SVM can use to construct a better separating hyperplane. The **kernel function** calculates the inner product of images of points from a dataset but in a higher dimensional space - an *implicit* feature space that is potentially infinite-dimensional - however it does not ever *explicitly* calculate the coordinates in the higher dimensional space. This is shown in Figure 4.20 for a simple example. The *kernel function* ϕ allows the construction of a separating hyperplane (here, a plane) in the feature space for a non-linearly separable dataset (in the input space).

Luckily, in machine learning libraries, the choice of kernel is simply a parameter we choose. We look at four kernels - 'linear', 'rbf' (radial basis function), 'poly' (polynomial), and 'sigmoid'.

Next, we validate our methods.

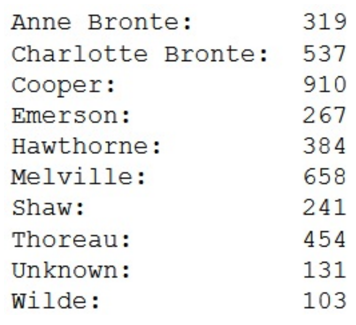
4.8 Evaluating Instance-Based Algorithms' Accuracies

We now split the entire dataset (including 'Agnes Grey' as an Anne Brontë text) into training and testing (80%/20%) splits. We can evaluate the performance of our algorithms on the whole dataset as a form of validation for our methods before we perform classification.

We show the number of chunks per author in Figure 4.21. This is important to consider as unbalanced classes can make our algorithms, k-nn especially, perform much worse. We notice that Cooper has the most instances by far, whereas Wilde has barely any (roughly 9 to 1 difference).

4.8.1 K-Nearest Neighbours Results

The accuracy for both the standard k-nn and distance weight k-nn models over the test set is shown in Figure 4.22. We notice that both achieve extremely high accuracy with k=1,2,3 and that performance drops, but remains high (80%), as we increase the value of k.



```
Anne Bronte:        319
Charlotte Bronte:   537
Cooper:             910
Emerson:            267
Hawthorne:          384
Melville:           658
Shaw:               241
Thoreau:            454
Unknown:            131
Wilde:              103
```

Figure 4.21: Number of chunks belonging to each author in 'Agnes Grey' dataset

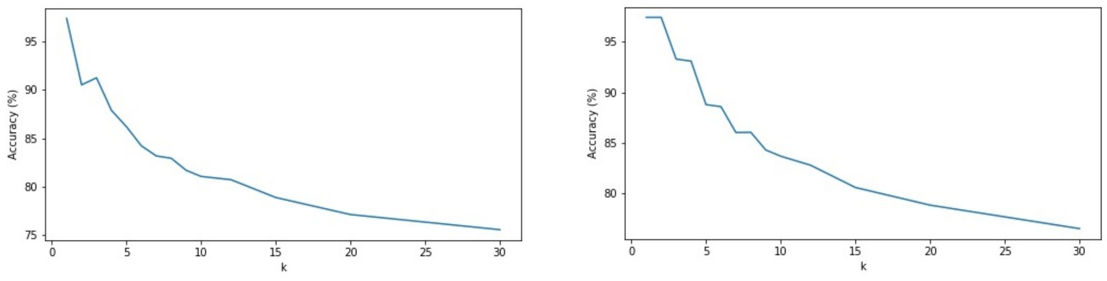


Figure 4.22: Accuracies for standard k-nn (left) and distance weighted k-nn (right) classification of 'Agnes Grey' 200 character 3-gram dataset by value of k

4.8.2 Support Vector Machine Results

The accuracy results by kernel is shown in Table 4.1. We see very impressive results (90% accuracy) for all kernel types except the polynomial ('poly') type. No tuning of parameters was performed beforehand, which perhaps explains this discrepancy.

4.9 Evaluating Instance-Based Classification Results

We return to classifying 'Agnes Grey' using our 200 character 3-gram dataset, but this time we use our instance-based classifiers and measure the proportion of chunks of 'Agnes Grey' attributed to each author.

Kernel Type	**Accuracy (%)**
Linear ('linear')	88.9
Radial Basis Function ('rbf)	91
Polynomial ('poly')	72.8
Sigmoid ('sigmoid')	91.1

Table 4.1: Accuracies for SVM classification of 'Agnes Grey' 200 character 3-gram dataset by kernel type

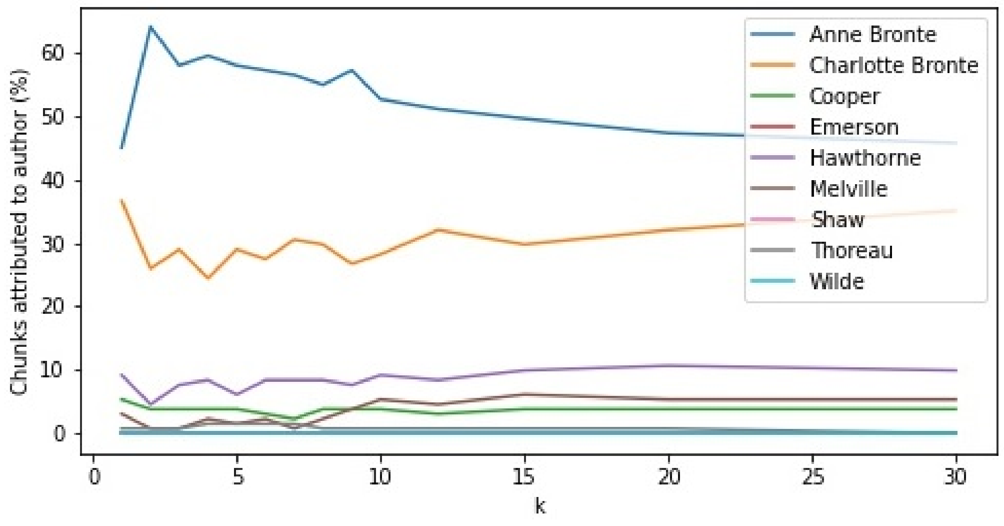


Figure 4.23: Proportion of chunks of 'Agnes Grey' attributed to each author in 'Agnes Grey' dataset for k-nn classifier by value of k

4.9.1 K-Nearest Neighbours Results

First, we look at k-nn. We first use the standard k-nn model, and then using distance weighting. The results are shown in Figures 4.23 and 4.24 respectively. We find comparable results to our profile-based distance measures - Anne Bronte is the clear favourite, and Charlotte Bronte shares quite a bit of similarity, with Hawthorne next. The standard k-nn performs best with small k (less than 10) with around 60% attributed to Anne, however distance weighted k-nn gives between 60% and 70% for all k less than 10.

At this point we tried using 'Manhattan' and 'Delta' distances as discussed in Section 4.7.1, however our results were very similar.

4.9.2 Support Vector Machine Results

We also tried classifying all the 'Agnes Grey' chunks using four different kernel types. These give us our most convincing results, with linear, rbf, and sigmoid kernels giving between 70% and 80% chunks attributed to Anne Bronte. The polynomial kernel, on the other hand, attributes quite a lot of chunks to Cooper (who was ignored by the other SVMs), suggesting it is being affected by the class imbalance. This may also explain why it attributes more writing to Charlotte Bronte than Anne. We also note Melville is the consistent third choice which is unexpected.

Again, these SVMs were used with the standard parameter presets (no tuning).

4.10 Discussion

We have shown that both the function word and 3-gram datasets have given convincing results in our attribution of 'Agnes Grey' - we will use these style

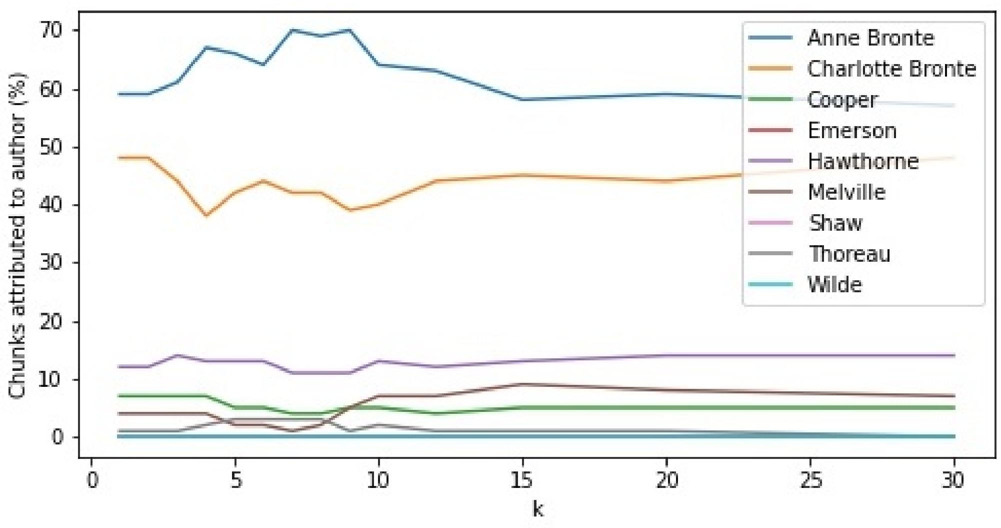


Figure 4.24: Proportion of chunks of 'Agnes Grey' attributed to each author in 'Agnes Grey' dataset for k-nn classifier with distance weights by value of k

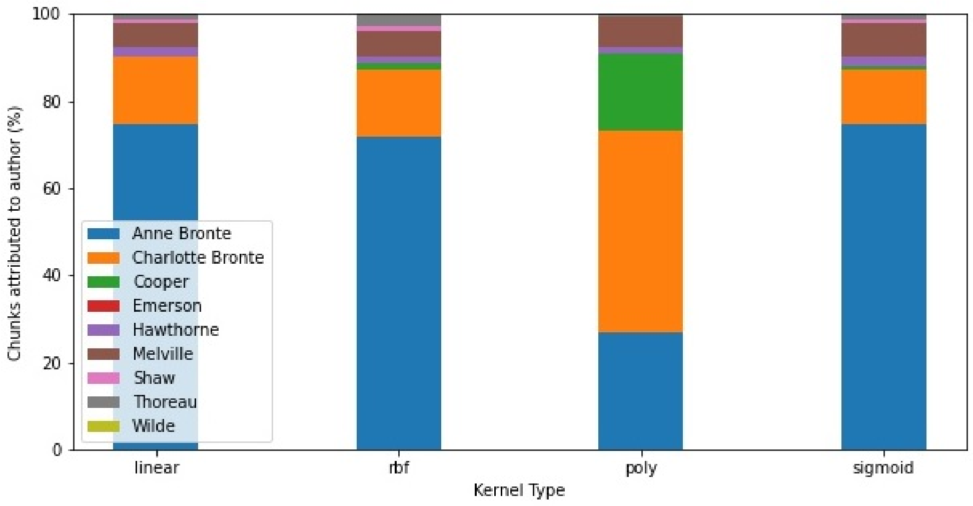


Figure 4.25: Proportion of chunks of 'Agnes Grey' attributed to each author in 'Agnes Grey' dataset for SVM classifier by kernel type

markers to try to attribute 'Frankenstein'. We have given strong evidence as to the effectiveness of our methodologies - the profile-based approach using 1-nn and PCA, and the instance-based approach using k-nn and SVMs.

We now apply all we have learned in our attribution of 'Frankenstein'.

Chapter 5

Who is the Author of 'Frankenstein'?

We follow our stylometric framework for performing authorship attribution as set out in Section 4 to answer the Frankenstein authorship question.

5.1 Dataset

For our studies we use original 1818 edition of 'Frankenstein'. We include texts from Mary Shelley, Percy Bysshe Shelley (both his poetry and prose), and William Godwin. We also include Byron (both his poetry and prose) as another writer from this time closely related to our other authors. We include all the texts used for this dataset in Table A.2 in the Appendix.

We also split Mary Shelley's work into an Early and Late period consisting of our her earliest novels (around the time of 'Frankenstein') and novels she wrote later in her life. This also allowed the classes to remain balanced.

We also note here that many of the prose works for Godwin especially, and also for Percy Bysshe Shelley and Lord Byron, were not available online.

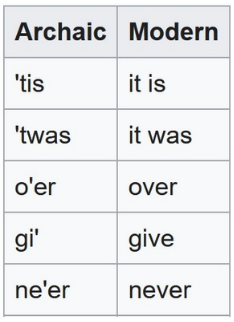
5.2 Preprocessing

The preprocessing for this dataset followed a similar pattern to the 'Agnes Grey' dataset, however there were far more footnotes present throughout the works, which was very time-consuming to remove.

Some extra steps were taken to improve the cohesiveness of our dataset.

5.2.1 Archaic Words

We standardise the spellings of common archaic words. Words like 'ere', which means 'before', were sometimes used at the time 'Frankenstein' was published. We find and replace these in all our documents with their standard forms nowadays for consistent word use across our dataset.



Archaic	Modern
'tis	it is
'twas	it was
o'er	over
gi'	give
ne'er	never

Figure 5.1: Some common poetic contractions
Source: Adapted from en.wikipedia.org/wiki/Poetic_contraction

5.2.2 Poetic Contractions

We must also consider **poetic contractions**, often called elisions. They are commonly used in poetry, usually to reduce the number of syllables in a line to fit the meter of a poem without changing the meaning. Some poetic contractions are shown in Figure 5.1. We again find and replace these across our documents to ensure we do not overlook the use of function words in particular.

5.3 Extracting Features

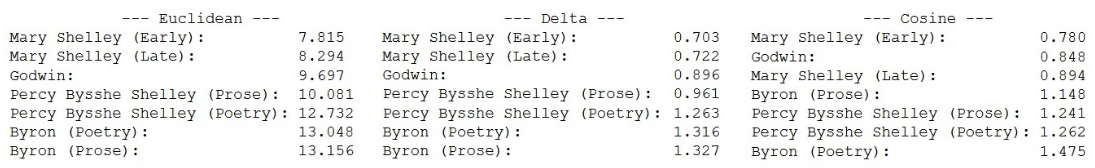
For our feature extraction, we use three feature sets to describe the authors' styles. We use 70 function words, 250 function words, and the 200 most common character 3-grams.

We also scale and normalise our dataset in the same way as the 'Agnes Grey' dataset.

5.4 Choosing Algorithms

We complete three analyses that are set out below.

1. Profile-based 70 & 250 function word, and 200 character 3-gram analysis using:
   (a) 1-nn
   (b) PCA visualisation
2. Instance-based (by texts) 70 & 250 function word analysis using:
   (a) PCA visualisation
3. Instance-based (by chunks) character 200 character 3-gram analysis
   (a) k-NN
   (b) SVM



```
         --- Euclidean ---                          --- Delta ---                              --- Cosine ---
Mary Shelley (Early):         7.815    Mary Shelley (Early):         0.703    Mary Shelley (Early):         0.780
Mary Shelley (Late):          8.294    Mary Shelley (Late):          0.722    Godwin:                       0.848
Godwin:                       9.697    Godwin:                       0.896    Mary Shelley (Late):          0.894
Percy Bysshe Shelley (Prose): 10.081   Percy Bysshe Shelley (Prose):  0.961   Byron (Prose):                1.148
Percy Bysshe Shelley (Poetry): 12.732  Percy Bysshe Shelley (Poetry): 1.263   Percy Bysshe Shelley (Prose):  1.241
Byron (Poetry):               13.048   Byron (Poetry):               1.316    Percy Bysshe Shelley (Poetry): 1.262
Byron (Prose):                13.156   Byron (Prose):                1.327    Byron (Poetry):               1.475
```

Figure 5.2: Distance results for the 'Frankenstein' dataset using 70 function words

5.5 Evaluating Profile-Based Results

We begin with an analysis of the profile-based k-nn and PCA results.

5.5.1 70 Function Word Analysis

First, we use the 70 function word dataset.

Distances

Figure 5.2 clearly shows Mary Shelley (Early) as the closest in style. The Cosine results differ quite a bit from the Euclidean and Delta results - Godwin is placed 2nd ahead of Mary Shelley (Late), and Byron (Prose) jumps to 4th from 6th. We notice Percy Bysshe Shelley's Prose style, the author some have claimed is 'Frankenstein's' true author, never makes it in the top 3.

PCA

We examine the PCA projections for the 70 function word dataset using all of the authors, shown in Figure 5.3. This shows 'Frankenstein' closest to Mary Shelley's work, but still close to both Godwin and Percy Bysshe Shelley (Prose).

We then remove the poetry vectors and perform PCA again, to get a clearer picture. This is shown in Figure 5.4. Comprising over 67% of the variance, the graph clearly shows Frankenstein close in style to both Mary Shelley's Early and Late works, however now Percy Bysshe Shelley appears to be next closest, in disagreement with our distance functions.

5.5.2 250 Function Word Analysis

Now, we repeat our analyses but using the 250 function word dataset.

Distances

The distance results, shown in Figure 5.5, give even stronger evidence for Mary Shelley's claim of authorship, as she is the top 2 across the measures, with her Early style notably even closer. Godwin again claims the 3rd spot.

PCA

We again examine the PCA projections but now for the 250 function word dataset. First we use all of the authors, as shown in Figure 5.6, and after this using only the

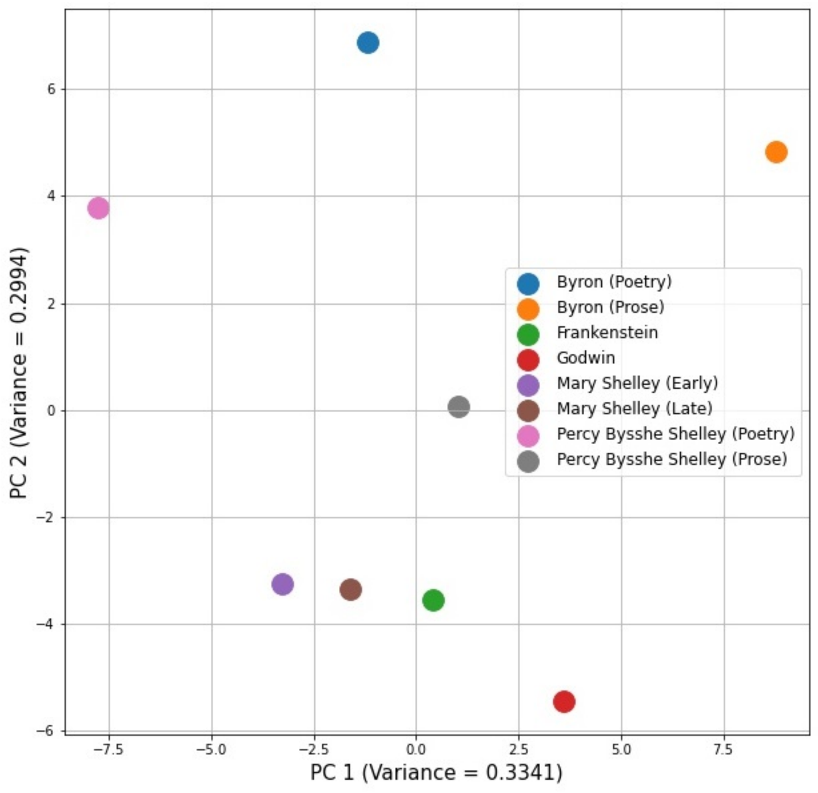


Figure 5.3: PCA with all authors using 70 function words

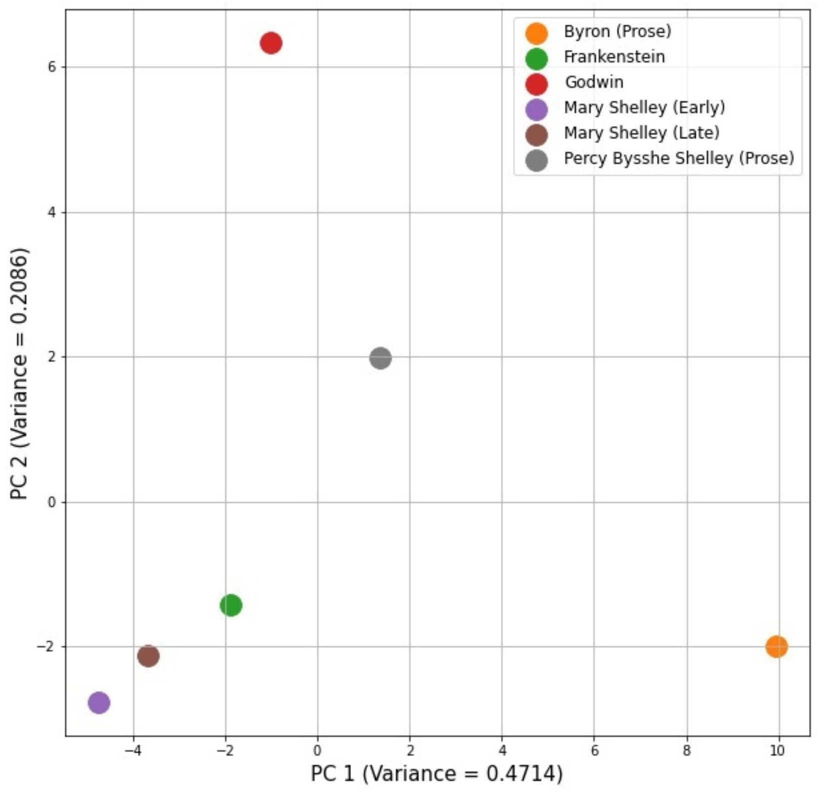
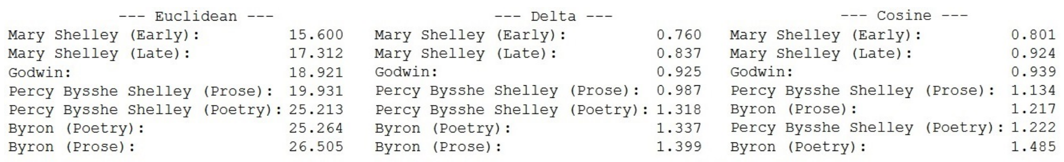


Figure 5.4: PCA with no poetry using 70 function words



```
        --- Euclidean ---                           --- Delta ---                               --- Cosine ---
Mary Shelley (Early):          15.600   Mary Shelley (Early):          0.760   Mary Shelley (Early):          0.801
Mary Shelley (Late):           17.312   Mary Shelley (Late):           0.837   Mary Shelley (Late):           0.924
Godwin:                        18.921   Godwin:                        0.925   Godwin:                        0.939
Percy Bysshe Shelley (Prose):  19.931   Percy Bysshe Shelley (Prose):  0.987   Percy Bysshe Shelley (Prose):  1.134
Percy Bysshe Shelley (Poetry): 25.213   Percy Bysshe Shelley (Poetry): 1.318   Byron (Prose):                 1.217
Byron (Poetry):                25.264   Byron (Poetry):                1.337   Percy Bysshe Shelley (Poetry): 1.222
Byron (Prose):                 26.505   Byron (Prose):                 1.399   Byron (Poetry):                1.485
```

Figure 5.5: Distance results for the 'Frankenstein' dataset using 250 function words

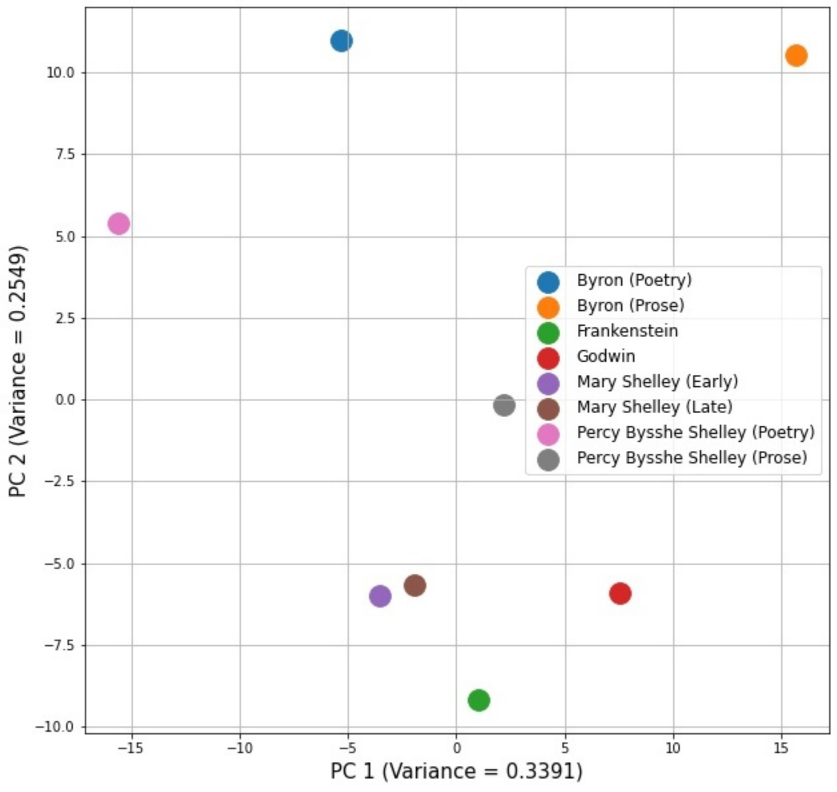


Figure 5.6: PCA with all authors using 250 function words

poetry vectors, shown in Figure 5.7. Like the 70 function word PCA these show 'Frankenstein' closest to Mary Shelley's work, but still close to both Godwin and Percy Bysshe Shelley (Prose), however when poetry is removed the next closest style vector is that of Godwin, not Percy Shelley as before.

5.5.3 200 character 3-gram Analysis

Finally, we repeat our analyses using the 200 character 3-gram dataset.

Distances

The distance results for this dataset, shown in Figure 5.8, tell a slightly different story - Godwin consistently bests Mary Shelley (Late), and even convincingly tops the results for the Cosine similarity measure, however she remains top for the other two measures. Percy Bysshe Shelley (Prose) also makes a consistent 4th place.

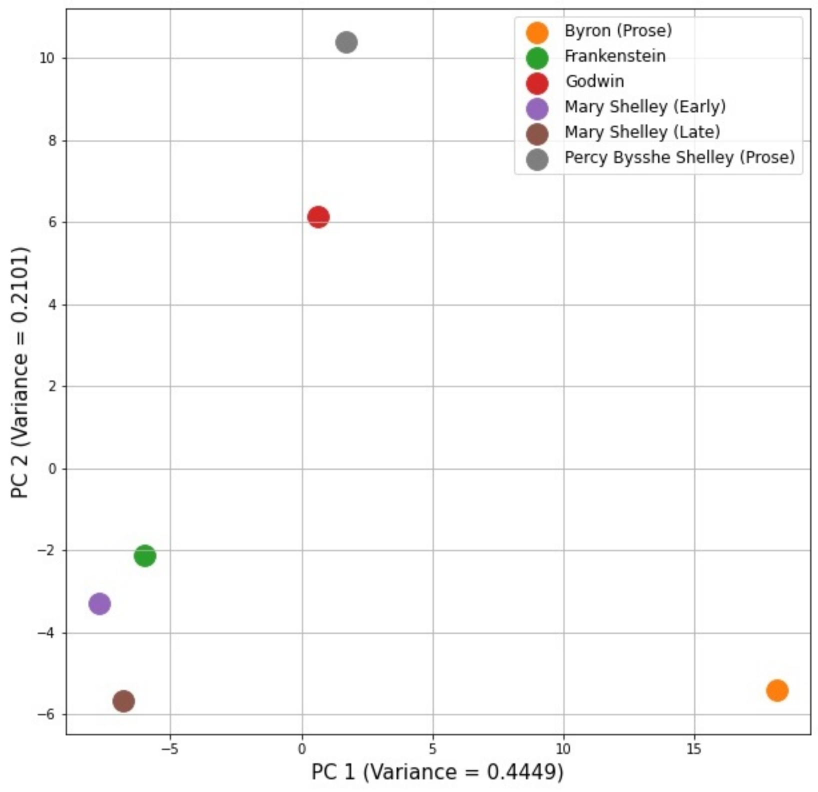
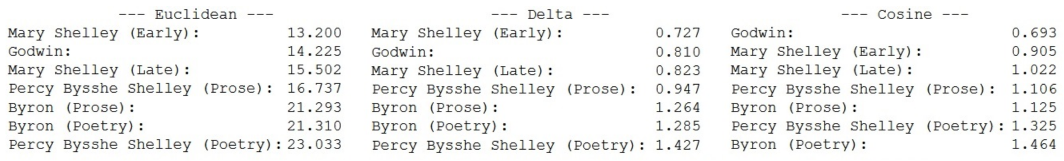


Figure 5.7: PCA with no poetry using 250 function words



```
           --- Euclidean ---                              --- Delta ---                              --- Cosine ---
Mary Shelley (Early):         13.200   Mary Shelley (Early):           0.727   Godwin:                          0.693
Godwin:                       14.225   Godwin:                         0.810   Mary Shelley (Early):            0.905
Mary Shelley (Late):          15.502   Mary Shelley (Late):            0.823   Mary Shelley (Late):             1.022
Percy Bysshe Shelley (Prose): 16.737   Percy Bysshe Shelley (Prose):   0.947   Percy Bysshe Shelley (Prose):    1.106
Byron (Prose):                21.293   Byron (Prose):                  1.264   Byron (Prose):                   1.125
Byron (Poetry):               21.310   Byron (Poetry):                 1.285   Percy Bysshe Shelley (Poetry):   1.325
Percy Bysshe Shelley (Poetry):23.033   Percy Bysshe Shelley (Poetry):  1.427   Byron (Poetry):                  1.464
```

Figure 5.8: Distance results for the 'Frankenstein' dataset using 200 character 3-grams

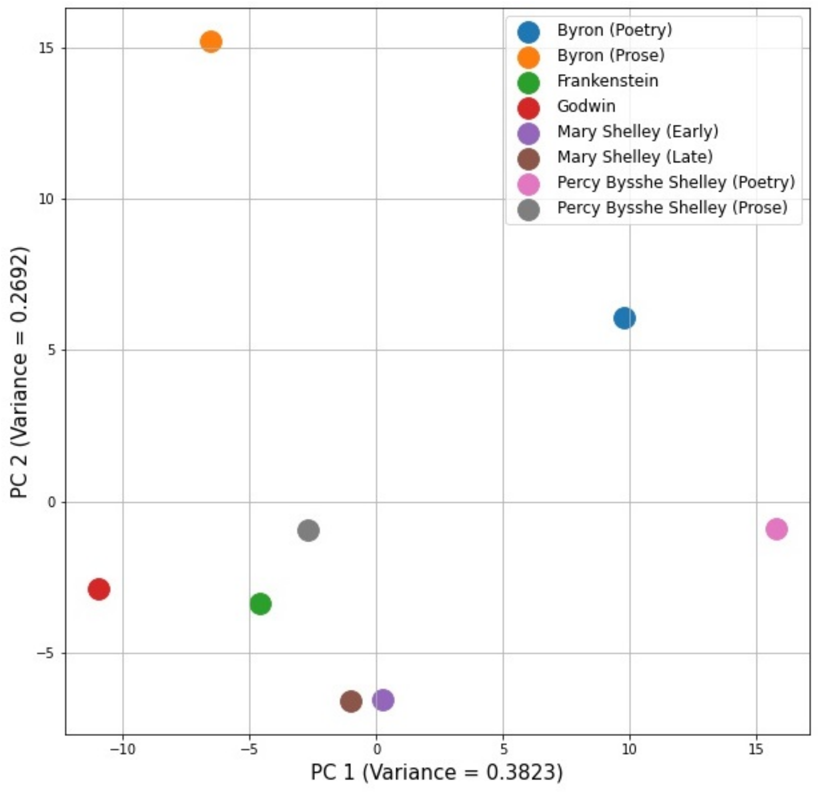


Figure 5.9: PCA with all authors using 200 character 3-grams

PCA

The PCA graphs for the 3-gram dataset are less conclusive than that of our other two datasets. Both the PCA using all the authors, shown in Figure 5.9, and when removing the poetry, shown in 5.10 put 'Frankenstein' near both Mary Shelley vectors, Godwin, and Percy Shelley (Figure 5.9 only)

5.6 Evaluating Instance-Based (by texts) Results

We also examine the style vectors produced for each text in our corpus using the 70 function words in order to see how 'Frankenstein' fits in with the works of each author. We use PCA to graph these style vectors.

We also repeated this analysis for the 250 function word dataset but the results were very similar and so have been omitted.

PCA

We first plot every text in our corpus, shown in Figure 5.11. This shows 'Frankenstein' in the middle of lots of Mary Shelley (Early and Late) works, as well as Percy Bysshe Shelley, and also relatively close to Godwin's works. Byron's Poetry and Prose, and Percy Bysshe Shelley's Poetry, are all on the outskirts of the

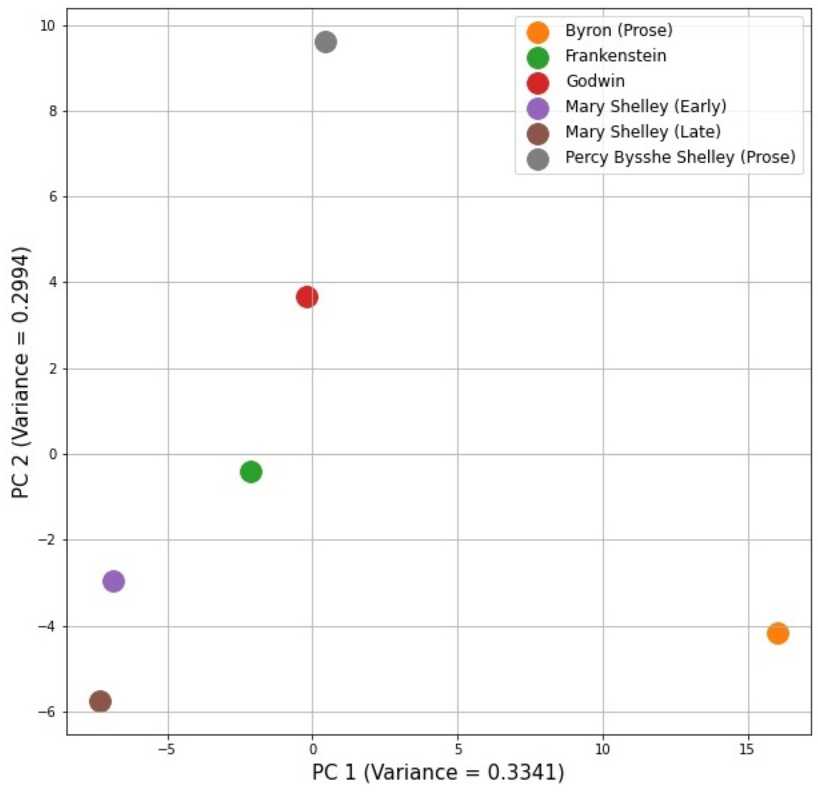


Figure 5.10: PCA with all authors using 200 character 3-grams

graph fairly well clustered. Important to note is that only 40% of the variance is maintained.

Figure 5.12 shows the PCA for our text style vectors but with only Mary Shelley, Godwin, and Percy Shelley's works. This clearly shows Frankenstein surrounded by the works of Mary Shelley, away from the other authors' works. This is perhaps our most convincing evidence yet.

5.7 Instance-Based (by chunks) Character 3-Gram Validation

Before performing instance-based classification using the 200 character 3-gram dataset, we repeat a validation of our classifications algorithms (k-nn and SVMs) that we used for the 'Agnes Grey' dataset, this time for our 'Frankenstein' dataset where we omit 'Frankenstein' itself. The parameters we see that give us the best results for this particular dataset will validate our findings when we turn to classifying 'Frankenstein'.

We include the number of chunks of each authors writing in Figure 5.13. We see the number of chunks is significantly greater than in the 'Agnes Grey' dataset. This means our computation will be much slower, but perhaps we will get more accurate results.

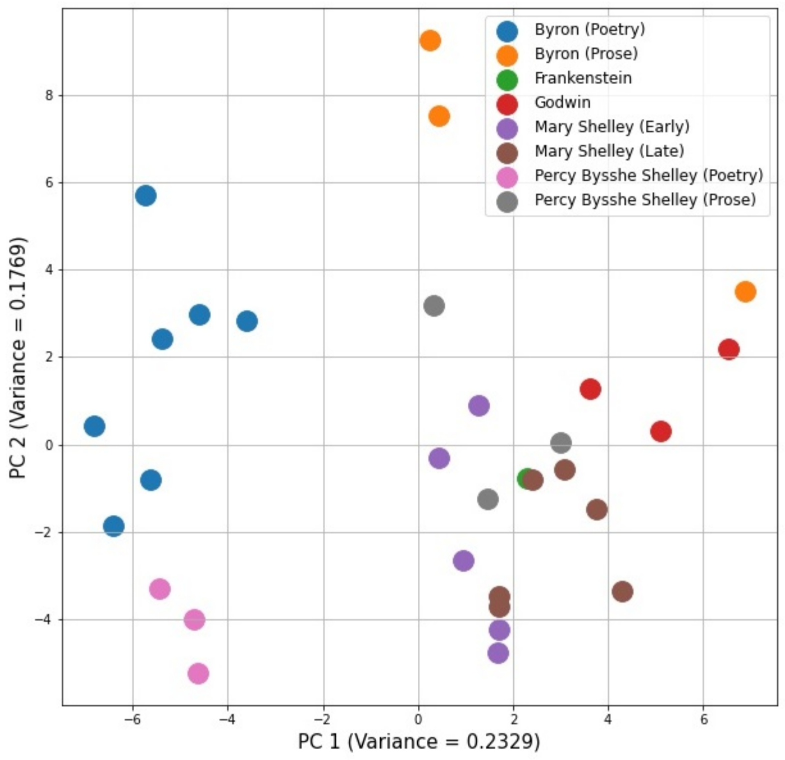


Figure 5.11: PCA with all authors' texts using 70 function words

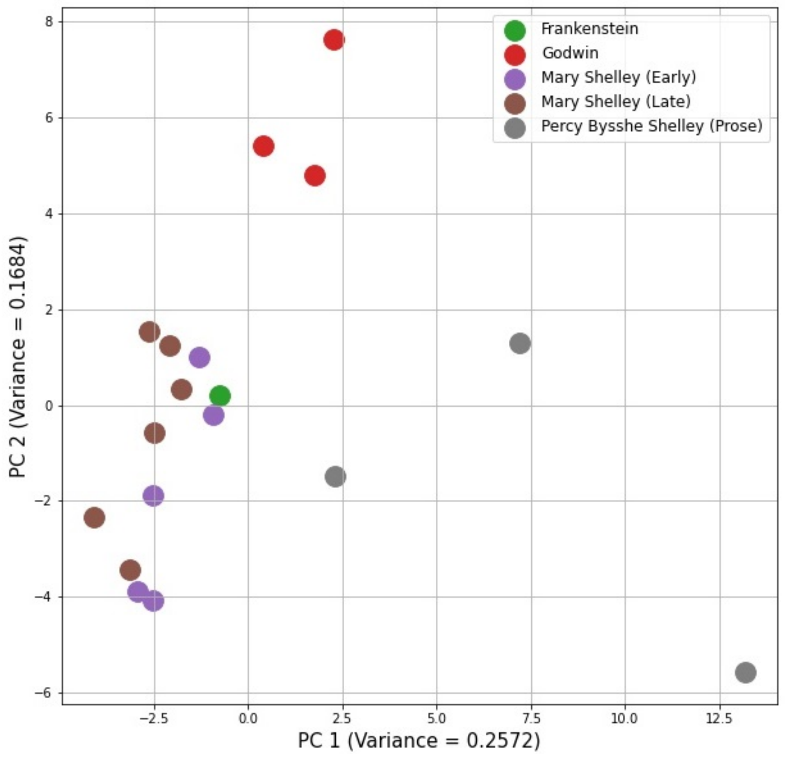


Figure 5.12: PCA with only Mary Shelley, Godwin, and Percy Bysshe Shelley's texts using 70 function words

```
Byron (Poetry) 4213
Byron (Prose) 1105
Frankenstein 581
Godwin 2886
Mary Shelley (Early) 2967
Mary Shelley (Late) 4943
Percy Bysshe Shelley (Poetry) 2132
Percy Bysshe Shelley (Prose) 1901
```

Figure 5.13: Number of chunks belonging to each author in 'Frankenstein' dataset

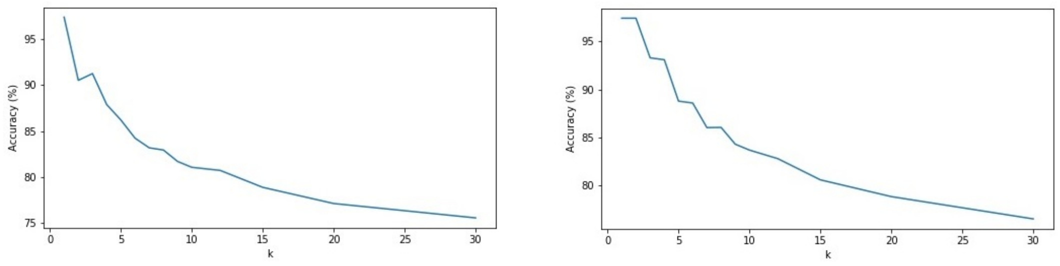


Figure 5.14: Accuracies for standard k-nn (left) and distance weighted k-nn (right) classification of 'Frankenstein' 200 character 3-gram dataset by value of k

K-NN

The k-nn accuracies for both standard k-nn and distance weighted k-nn for the 'Frankenstein' dataset are shown in Figure 5.14. This shows a very similar performance to the 'Agnes Grey' dataset, with very high performance for k=1,2,3 and a steady decline as k increases.

SVM

The SVM accuracy results, shown in Figure 5.1, for this dataset are again quite similar to the 'Agnes Grey' dataset, however the performance has decreased a decent bit. We now achieve between 76% and 86% accuracy, with the polynomial kernel performing worst, and the radial basis function kernel performing by far the best (6% better than any other kernel). This is still respectable performance.

Kernel Type	**Accuracy (%)**
Linear ('linear')	80
Radial Basis Function ('rbf')	85.9
Polynomial ('poly')	75.8
Sigmoid ('sigmoid')	78.1

Table 5.1: Accuracies for SVM classification of 'Frankenstein' 200 character 3-gram dataset by kernel type

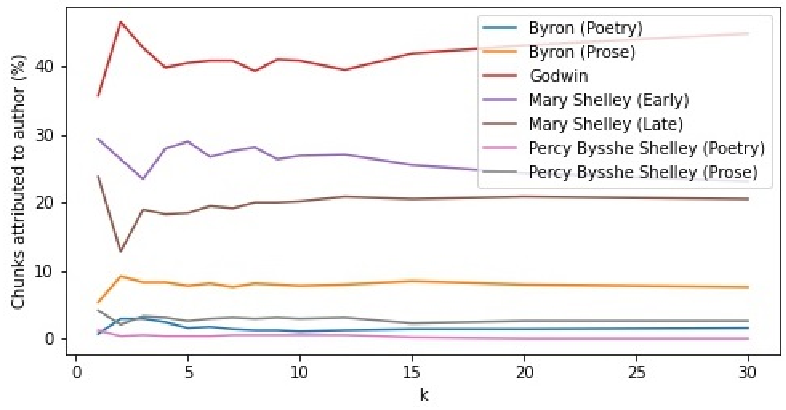


Figure 5.15: Proportion of chunks of 'Frankenstein' attributed to each author in 'Frankenstein' dataset for k-nn classifier by value of k

5.8 Instance-Based (by chunks) Character 3-Gram Analysis

We now use our instance-based classification methods (k-nn and SVMs) to attribute the chunks of 'Frankenstein'.

K-nn

We show the proportion of chunks of 'Frankenstein' attributed to each author of our dataset in Figure 5.15. We notice that Godwin consistently gets 40% of the chunks, the majority stake, however Mary Shelley Early and Late are in 2nd and 3rd place respectively. This is perhaps unsurprising given the inconclusive results for both the distance measures and PCA for our 3-gram dataset as in Section 5.5.3. Something surprising is that we do not see Percy Shelley's Prose being attributed many chunks at all, even though he was 4th in the distance measures and by PCA analysis, in fact it is Byron's Prose that gets the next most. Perhaps this is due to the class imbalance between Percy Shelley's Prose and, Mary Shelley and Godwin.

We combine the chunks for Mary Shelley Early and Late in Figure 5.16 where we see, for k below 20, Mary Shelley receives the most chunks. However, this is not a fair statistical k-nn analysis as it is not attributing the chunks to a 'Mary Shelley' class; it is combining the results of two classes treated as separate.

We also achieved very similar results for distance weights and even other distance measures (Manhattan and Delta).

SVM

Figure 5.17 shows the proportion of the chunks of 'Frankenstein' attributed to each author of our corpus. In contradiction to our k-nn results, we see Mary Shelley Early getting the most chunks (40%) while Godwin receives around 20%. Mary Shelley Late is in 3rd, while Percy Shelley Prose gets slightly more than Byron as we would expect from our profile-based results.

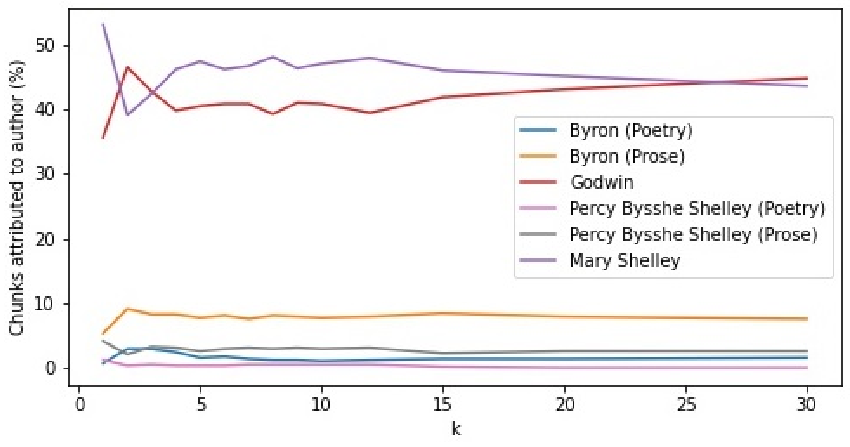


Figure 5.16: Proportion of chunks of 'Frankenstein' attributed to each author (with Mary Shelley (Early) and (Late) combined) in 'Frankenstein' dataset for k-nn classifier by value of k

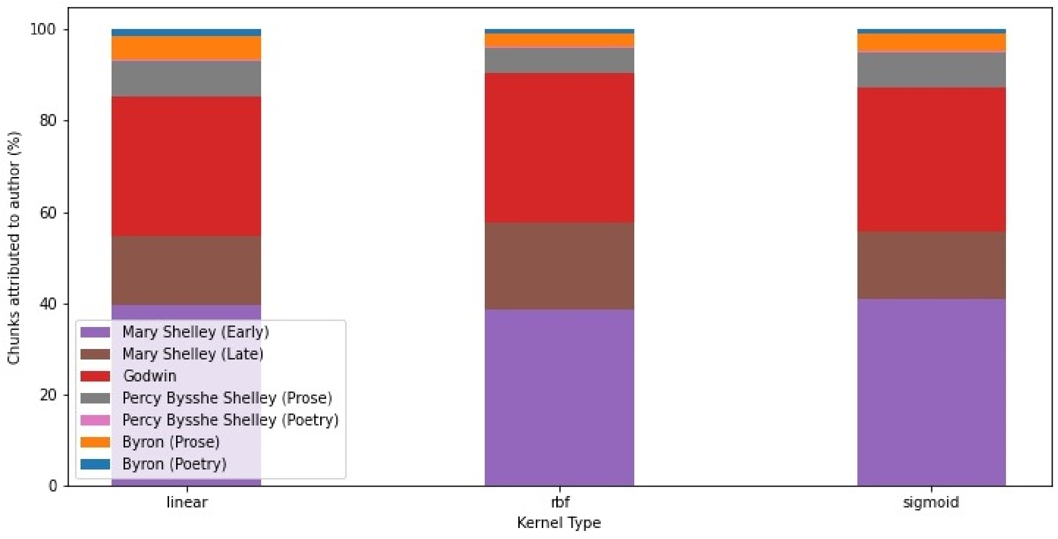


Figure 5.17: Proportion of chunks of 'Frankenstein' attributed to each author in 'Frankenstein' dataset for SVM classifier by kernel type

Chapter 6

Conclusion

Given the historical context, and the statistical analyses we have performed, it is highly likely ‘Frankenstein’ was written by Mary Shelley, and not Percy Bysshe Shelley. In terms of stylometric features, Percy Bysshe Shelley’s work shows little resemblance to ‘Frankenstein’, while Mary Shelley’s works do.

In fact, it is the works of her father, William Godwin, who show the most similarity to that of ‘Frankenstein’. Is this closeness in style to the work of Godwin as a result of his education of her and the influence her had on her upbringing as a single parent. Or, perhaps, did Godwin have a significant role in the writing of ‘Frankenstein’?

A second analysis could be carried out but with complete corpus of all our authors’ works. A suggestion would be to use smaller chunks of writing. This could be used to explore where there were collaborations in the text. Further study could also examine the differences between all three editions, especially comparing the 1st and 2nd to see influence of Godwin’s editing.

Further analysis could look at differences in Mary Shelley’s style as she grew up, from the early education from her father, his library, and his friends, to her more independent lifestyle later. Perhaps psychologist and/or philologists could analyse the text at a higher level, looking at themes and characters influenced from Mary Shelley’s life (like death of her mother, her often bad relationship with father, and the deaths of many of her children, her friends and her husband).

Appendix A

Lists of Works

Author	Work
Anne Brontë	Agnes Grey The Tenant
Charlotte Brontë	Jane Eyre The Professor
James Fenimore Cooper	Last of the Mohicans The Spy Water Witch
Ralph Walsdo Emerson	Conduct of Life English Traits
Nathaniel Hawthorne	House of Seven Gables The Scarlet Letter
Herman Melville	Moby Dick Redburn
George Bernard Shaw	Getting Married Misalliance Pygmalion
Henry David Thoreau	Concord Walden
Oscar Wilde	A Woman of No Importance Ideal Husband

Table A.1: Authors and Works of our ‘Anges Grey’ dataset

Author	**Work**
???	Frankenstein (1818 Edition)
Mary Shelley (Early Period)	Mathilda The Last Man Valperga
Mary Shelley (Late Period)	Falkner Lodore Tales and Stories The Fortunes of Perkin Warbeck
Percy Bysshe Shelley (Poetry)	Complete Poetical Works (Vol 1-3)
Percy Bysshe Shelley (Prose)	A Vindication of Natural Diet Prose (Vol 1-2)
Lord Byron (Poetry)	Poetry Works (Vol 1-7)
Lord Byron (Prose)	Fragment of a Novel Letters and Journals (Vol 1-2)
William Godwin	Caleb Williams Damon and Delia St. Leon

Table A.2: Authors and Works of our ‘Frankenstein’ dataset

Bibliography

[1] Steven Bird, Edward Loper, and Ewan Klein. *Natural Language Processing with Python*, chapter 8. O'Reilly Media Inc., 2009.

[2] Jeff Collins, David Kaufer, Pantelis Vlachos, Brian Butler, and Suguru Ishizaki. Detecting collaborations in text: Comparing the authors' rhetorical language choices in the federalist papers. *Computers and the Humanities*, pages 15–36, 2004.

[3] James R. Fitzgerald. Using a forensic linguistic approach to track the unabomber. *Profilers*, pages 193–221, 2004.

[4] Mark Glickman, Jason Brown, and Ryan Song. (a) data in the life: Authorship attribution in lennon-mccartney songs. *Harvard Data Science Review*, 1(1), 7 2019. https://hdsr.mitpress.mit.edu/pub/xcq8a1v1.

[5] Peter Grzybek. *History and Methodology of Word Length Studies*, pages 15–90. 01 1970.

[6] Moshe Koppel and Yaron Winter. Determining if two documents are written by the same author. *JASIST*, 65(1), 2014.

[7] John Lauritsen. *The Man Who Wrote Frankenstein.* Pagan Press, 2007.

[8] Wincenty Lutoslawski. Principes de stylométrie appliqués à la chronologie des œuvres de platon. 1898.

[9] Stephen MacDonell, Andrew Gray, Grant MacLennan, and Philip Sallis. Software forensics for discriminating between program authors using case-based reasoning, feedforward neural networks and multiple discriminant analysis. volume 1, pages 66–71, 02 1999.

[10] Rachel McCarthy and James O'Sullivan. Who wrote Wuthering Heights? *Digital Scholarship in the Humanities*, 06 2020.

[11] T. C. Mendenhall. The characteristic curves of composition. *Science*, 9(214):237–249, 1887.

[12] Frederick Mosteller and David L. Wallace. Inference in an authorship problem. *Journal of the American Statistical Association*, 58(302):275–309, 1963.

[13] Peter E. Prosser. Church history's biggest hoax: Renaissance scholarship proved fatal for one of the medieval papacy's favorite claims. *Christian History*, 20(4):35+, 2001.

[14] James Rieger. *Frankenstein, or, The modern Prometheus: The 1818 text.* University of Chicago Press, 1982.

[15] Joseph Rudman. The state of authorship attribution studies: Some problems and solutions. *Computers and the Humanities*, 31(4):351–365, 1997.

[16] Walter Scott. 'Remarks on Frankenstein'. *Blackwood's Edinburgh Magazine*, 2(XII):613–20, 1818.

[17] Efstathios Stamatatos. A survey of modern authorship attribution methods. *Journal of the American Society for Information Science and Technology*, 60(3):538–556, 2009.

[18] Andre Vauchez. *Encyclopedia of the Middle Ages*, page 445. Routledge, 2001.

[19] Duncan Wu. *Myth 25: Percy Bysshe Shelley wrote Frankenstein*, chapter 25, pages 212–219. John Wiley & Sons, Ltd, 2015.