FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

RECHTS- UND WIRTSCHAFTS-
WISSENSCHAFTLICHE FAKULTÄT

Linguistic Consumer Profiling – Identifying Big-Five Personality Traits on the Facebook Statuses.

Handed in by:	Dustin Nguyen	Vishweshwara Keekan	Dmitrij Petrov
Matr.Nr.			
Email	{dustin.nguyen, vishweshwara.keekan, dmitrij.petrov} @ fau.de		

Professor: Prof. Dr. Freimut Bodendorf

Supervisor: Alexander Piazza

Seminar: Case Solving Seminar – Final Project Report

Nuremberg, 13 March 2016

Eidesstattliche Erklärung

Wir versichern, dass wir die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt haben. Die Arbeit wurde in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegt und von dieser als Teil einer Prüfungsleistung angenommen. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Nürnberg, den 13. March 2016

Table of Contents

Table of figures and tables

List of abbreviations

AUC	Area under (ROC) curve
CPU	Central Processing Unit
DRC	Dream of the Red Chamber
FB	Facebook
GPGPU	General-Purpose GPU
GPU	Graphics Processing Unit
kNN	k-Nearest Neighbors
Linear SVC	Support Vector Classifier with a Linear Kernel
ML	Machine Learning
MNB	Multinomial Naïve-Bayes
NB	Naïve-Bayes
NLP	Natural Language Processing
ODS	Original dataset
SVMs	Support Vector Machines
TF-IDF	Term frequency – Inverse document frequency
VPS	Virtual Private Server

1 Introduction

In the following academic paper, we are going to introduce our reader to the profiling of consumers using linguistic analysis, popularly known as stylometry. Without going into the details of this technique at this stage, as it will be further explained in the subsequent chapters, one can say that each person has a unique writing style. Therefore, for example, in the case of disputed or anonymous documents the content can be analysed, compared and its style characteristics might be attributed to its most probable author(s).

This paper consists of two parts. In the first part, we are going to give the reader insights into this field, starting with a definition of the term stylometry and then a literary review. Consequently, we outline several historical examples which are all mainly related to the concepts of the authorship attribution. However, in this part, we also give several modern-era examples which rather deal with the profiling of users, e.g. on the social media.

In the second part, our goal will be more practical as we are going to conduct supervised machine learning (ML) on our dataset in Python[1]. The dataset, which is available on the *myPersonality.org* project, consists of almost 10 000 Facebook (henceforth referred as FB) status updates with many different network properties, e.g. size, density, betweenness etc. (Kosinski et al, 2015; Celli, 2013). Our goal here is to try to analyse the FB data and be able to attribute (predict) each status update of the user to one of the *Big Five personality trait*, at the highest possible confidence.

2 Stylometry – The Discipline

The term *stylometry* has emerged in the second half of 19th century, particularly the Polish scholar named *Wincenty Lutosławski* is credited of coining it in his manuscript *The Origin and Growth of Plato's Logic* from 1897. However, as Grzybek (2014, p. 2) writes, by the same year "*neither the term nor the discipline, stylometry, was new*".

[1] https://www.python.org

In fact, there have been many forerunners such as Augustus De Morgan (1806-1871), Thomas Mendenhall (1841-1924), Robin Collingwood (1889-1943) or Lewis Campbell (1830-1908) who was one of the pioneers of the whole discipline and has contributed significantly "*to the question of Plato's chronological development*" in 1867 (Craik, 2014). Indeed, basic concepts and methodologies of different authors which dealt with analysis of written text have converged during the end of 19th and beginning of 20th century "*into one common field of research*" – the stylometry (Grzybek, 2014, p. 3).

As Holmes & Kardos (2003, p. 5) write in their journal article, stylometry is "*the statistical analysis of literary style*" which according to Knight (1993, p. 45) further deals with "*the study of individual or group characteristics in written language*" (e.g. sentence length, word frequencies etc.). Holmes & Kardos (2003, p. 5) also continue by saying that authors have "*unconscious aspect[s] to their style*", also known as *stylometric features*. These are "*behavioral aspect(s) that a person exhibits during writing*" (Brocardo et al., 2015). Given that they cannot be manipulated, are distinctive and can be even quantifiable they allow to "*uncover 'characteristics' of an author*" (Holmes & Kardos, 2003; Holmes 1998).

There are many purposes of stylometry such as genre and gender classification, sentiment analysis, historical study of language changes or forensic linguistics (Wang, 2013, p. 2). An interesting but less well known approach of stylometry has been studied by Wincenty Lutosławski. Based on earlier works of Lewis Campbell, he tried to understand the chronological order of Plato's dialogues as opposed to have it "*supplied by our historical tradition*" (Grzybek, 2014, p. 4; Lutosławski, 1897, p. 11).

There are two major use-cases of *stylometry* and both of these will be explored in this paper. The first one is authorship attribution and its verification. Scholars have always been interested in identifying authorship of anonymous texts (e.g. Beowulf), detecting plagiarism (e.g. Karl-Theodor zu Guttenberg's Dissertation Affair), resolving issues of disputed authors (e.g. see later section 2.2.1.2) and recently also in attributing authorship of the (malicious) software source code (Wang, 2013, p. 3; Burrows, 2010; Abou-Assaleh et al, 2004; Spafford et al., 1993).

The second use-case of stylometry is authorship profiling. In recent years, this arose due to the unlimited access to very large volumes of free behaviour data about users of different online services. In particular e-commerce websites and social media networks with their users offer a great and powerful resource to explore. Such a profiling can also apply both on purely anonymous texts (e.g. on aforementioned Beowulf) as well as on the content of which it is known by whom it had been written – for example on the social media where some form of a nickname is usually required. Resulting examples of profile dimensions can be gender, age, language (incl. e.g. geographic position), mood or neuroticism[2] — a personality type (Koppel et al., 2009).

2.1 Stylometric Analysis

Analysis of stylometry according to Abbasi et al. (2008) is the "*statistical analysis of writing style*". Zheng et al. (2006) further explain the "*four important characteristics of stylometric analysis are the tasks, stylistic features, classification techniques, and parameters (i.e., factors influencing authorship analysis performance, such as number of classes, amount of text, noise)*". Brief explanation on each of the individual characteristics is provided below.

2.1.1 Stylometric Tasks

Two types of stylometric tasks exist. These are *identification* and *similarity detection*. While the first one involves matching anonymous with previously identified texts and later developing conclusions, the similarity detection involves matching multiple anonymous texts and accessing the degree of similarity (Vel et al, 2001; Gray et al., 1997).

2.1.2 Stylometric Features

Along with the evolution of stylometry it is also evident that the features of stylometry have evolved over time. Some of the initial instances of stylometry date to usage of frequency of words which can be considered as a very basic feature (Mendenhall, 1901).

[2] Neuroticism is one of Big-Five psychological traits, which refers to the "*individual differences in negative emotional response to threat, frustration, or loss*" and in text this can be observed by using words such as *feel*, *worry* or *hurt* etc. (Lahey, 2009; Koppel et al, 2009).

Today stylometric features are plenty and their type to be used for any study is often determined by the volume and quality of the data available for training and testing purposes. The stylometric features can be broadly classified as following.

2.1.2.1 Quantitative

This feature is based on the quantitative attributes of the text and they are primarily based on characteristics such as *Input Algorithm* (algorithm used for comparing writing samples), *Textual Measurements* (word-length, sentence-length, vocabulary richness, word frequency, punctuation frequency, etc.), *Corpus Compilation* and finally A*lgorithmic Evaluation* (Grieve, 2007).

2.1.2.2 Lexical

Iqbal et al. (2008) describe that these features focus on an individual's usage of isolated characters and words which are unique to his particular writing style. Since this feature can contain a large range of characteristics, it is divided into two types:

- *Character-based*: Considering features such as character count (N), ratio of digits to N, ratio of letters to N etc.
- *Word-based*: considering features such as average sentence length, ratio of words in characters to N, vocabulary richness etc.

2.1.2.3 Syntactic

Syntactic features focus on function words which are typically used for all purposes in writing. Some of the common function words are "thought", "where", "your" etc. Punctuations such "," and "!" are also considered in this stylometric feature. The occurrence of function words and punctuations provide the approach in syntactic feature. Function words are very effective and this has been proved previously by Mosteller & Wallace (1964) and Iqbal et al. (2008).

2.1.2.4 Structural

This features focuses on the overall structure of text than the characters or words associated with it. Every individual is considered to have unique arrangement to his text layout and organization of content. This is also applicable to the way people write subjects or use greetings in emails (Iqbal et al., 2008).

2.1.2.5 Semantic

This feature focuses on the synonyms, polysemantic words (words having several meanings) and also identification of parts of speech and words thesaurus (Zurini, 2015).

2.1.2.6 Content Specific

Here the focus is on the keywords used in the content. Typically, it is observable behaviour that people involved in cybercrimes would use street keywords however those participating in forums/discussions on the internet would use different ones. This feature requires a term taxonomy to be built using which the investigations can be conducted to match the analysis with the text keywords (Zheng et al., 2006; Iqbal et al., 2008).

2.1.2.7 Idiosyncratic

Lastly, idiosyncratic features include consistent grammatical or punctuation errors and mistakes that an individual commits while writing the text (Iqbal et al., 2008).

2.1.3 Stylometric Techniques

Implementation of stylometric features on a text or a dataset are usually done using special techniques. Stylometry techniques are usually selected based on the existing data and it may also result in creation of new framework to overcome existing analysis techniques (Sun et al., 2012).

The popular techniques as given by Iqbal et al. (2008) are:

- Principal component analysis (PCA)
- N-gram models
- Markov models
- Cross entropy
- K–L similarity

2.1.4 Stylometric Parameters

The two parameters given by Zheng et al. (2006) are *scalability* and *robustness*. While scalability refers to the impact of the number of author classes on classification performance, robustness represents the ability to overcome techniques used by intentional stylistic alteration and copycatting/message forging (Iqbal et al., 2008).

2.2 Examples

In this section, we present several examples where methods of stylometry have been successfully applied in order to attribute authorship of disputed works. At first, we are going to look at some historical examples.

William Shakespeare is not only considered the greatest English playwright but is also one of the most disputed authors of all times. Some of his plays are said to be written by Christopher Marlowe, Edward de Vere, William Stanley or even a group of authors.

Then, we are going to continue with one of China's *Four Great Classical Novels* – the *Dream of the Red Chamber* from 18th century. Here the theory says that the first 80 chapters have been written by one author, whereas 40 others come from two different authors. This resulting into a prime example where stylometry can be applied.

In the modern era, we are going to look at stylometry from today's 21st century perspective. It is especially relevant, because of the latest advancements in internet technologies where the majority of communication is now taking place. Therefore, often times it has become important to identify anonymous posts/emails/statuses and stylometry has turned into the ideal discipline to address this new age problem.

2.2.1 Historical Examples

2.2.1.1 Controversies of William Shakespeare

Ever since the late 19th century extraction of statistical tendencies has been significantly used in author identification (Mendenhall, 1887; Mascol, 1888a/b). The method employed was comparing the extracted tendencies and finding some statistically quantifiable set of characteristics inherent in single author's use of written language with a group of works to identify the uniqueness of the document (Fox et al., 2012).

These techniques and their relevancy have been checked even on the works by William Shakespeare who is considered to be one of the greatest dramatist and writer (Wells, 1997). The extent of which is such that the individuals, who contest Shakespearean authorship, are specifically known as *anti-Stratfordians*. Individuals who are pro Shakespearean authorship are known as *Stratfordians* (Fox et al., 2012).

The anti-Stratfordians cite various pieces of evidence including that Shakespeare may have been illiterate or even barely literate (Bethell, 1991; Nelson, 2004). This combined with other pieces of evidence does lead us to the debate about the authorship of his works (Fox et al., 2012).

Principles of stylometry can be considered as the ideal method for investigating the authorship authenticity. Not surprisingly there has been a substantial amount of work that has been done around it. Some of the famous studies include the examination of word length frequency which is a unique writing trait exhibited by authors. If every author has even distribution of word length frequency across authors, it can be argued that the author of all works are same. But there are also instances of individual authors that have the same word length frequency, hence this theory fails (Mendenhall, 1901; Williams, 1975; Fox et al., 2012).

Similarly, many other techniques including modern methods such as machine learning and neural networks were employed on both Shakespeare's works, as well as on authors associated with him around the same time. The results did cast a doubt and showed anomaly in many of Shakespeare canon works (Merriam, 1996/1998; Matthews & Merriam, 1993; Merriam & Matthews, 1994; Fox et al., 2012).

The most famous however remains the "Marvolian Theory" which hypothesis that influential dramatist Christopher Marlowe escaped death and in exile wrote with Shakespeare's name as a front. Thus, all of these works are the entirety of Shakespeare Canon (Schoenbaum, 1991; Fox et al., 2012). The most detailed validation testing was indeed conducted by Fox et al. (2012) who compared samples of five authors – Georg Chapman, Ben Jonson, Christopher Marlowe, Thomas Middleton and Shakespeare. A total of 77 works were chosen, with the total word count of 1 945 800. The testing was based using two different models:

- *General Vocabulary*: With this model, distribution of all words appearing in the works of an author are examined and considering the frequency and probabilities of function word occurrences, similar works are statistically determined.
- *Generative Model*: With this model, not just the vocabulary of words but also frequencies of function words, parts-of-speech and bigrams (two words that repeat together consequently) are considered. The testing is based on the probabilities between these words.

Also a third experiment using clustering was employed to account for unknown authoring or overlap in some of the works. Here proven works were taken into account to calculate a total variance. This was done to ensure "*the unlabelled vectors of data for each work cluster together in the way that minimizes the overall differences in the distributions*" (Fox et al., 2012).

The results of the testing showed that Shakespeare and Marlowe had similar authoring styles, but their works were also unique. Though there are limitations to the experiment, the results obtained were best explained by the assumption that Marlowe is not Shakespeare.

2.2.1.2 Dream of the Red Chamber

The novel by Cao Xueqin from the 18th century forms a part of China's most-influential classical works, widely known as *Four Great Classical Novels* (Antizio, 2013). The *Dream of the Red Chamber* (hereinafter DRC) is the oldest from the set and considered by many as China's greatest classic novel (Tu & Hsiang, 2013). It has even a special field – redology – devoted entirely to studying this novel as it describes the life of two aristocratic families during the Qing Dynasty (1644-1912) in 18th century and "*psychological affairs*" of over 400 characters (Hu et al., 2014).

The novel first saw its light in 1759, just five years before Cao died. Indeed, he didn't publish his piece and what had been circulated in the society were only some hand-copied first 80 chapters of his novel (Hu et al., 2014). The first printed edition was published in 1791 and this is often called *Cheng-Gao edition*, after two scholars Cheng Weiyuan and Gao E. The dispute came on the stage when readers of this version saw that it had 40 additional chapters, which both scholars claimed were unpublished Cao's papers "*obtained through different channels*" (Hu et al., 2014). Since then, there was a lot of questioning of who wrote those 40 chapters and many believed the publishers – Cheng and Gao.

Certainly one of the most recent and profound studies about application of stylometry in the DRC has been conducted by Xianfeng Hu, Yang Wang, and Qiang Wu in 2014. They have presented a new mathematical method "*of authorship by testing for a so-called chrono-divide in writing styles*" which also incorporated newest techniques of the ML, particularly the S*upport Vector Machines* (Hu et al, 2014). Their results proved that first 80 chapters, with one exception of the chapter 67, have been written by Cao Xueqin, whereas the last 40 chapters were written by two different authors. This is also in line with several other studies including Zhao and Chen (1975), Yu (1998), and Yang (2003) which all "*observed significant differences between the first 80 and the last 40 chapters*" (Tu & Hsiang, 2013).

The method of Hu et al. (2014, p. 5) is based on the premise that there is a change of authors at some point in the novel. Therefore, if one divides the story in *n* chapters ("*chronologically ordered samples*"), at some point there will be a stylistic discontinuity what they call a *chrono-divide*. This is precisely what their method detects and tries to achieve using a classifier that separates the book into two different classes given "*a group of features that characterize the difference of their [authors] respective styles*" (p.6). The mere "*existence of such a classifier will provide strong support for the two-author hypothesis*" they write (p. 6).

For the selection of stylistic features, Hu et al. (2014) used for their analysis *n + m + 4* characteristics. The first two (*n+m*) are content independent and they are most frequently used *function characters* and *words*, meaning "*a class of words [and Chinese characters] that in general have little content meaning, but instead serve to express grammatical relationships with other words within a sentence*" (p. 2). In English, this would be for example articles, pronouns, prepositions, conjunctions etc. (Pylkkanen, 2003). The other *four* features are chapter specific and they are "*the mean and variance of* *sentence length as well as the frequencies of direct speeches and exclamations*" (Hu et al., 2014, p. 7). Given fluctuations in number of features for each sample of the book, only a subset of all *n* + *m* + *4* possible ones that achieve "*the highest discriminative power*" will be applied (p. 7).

As already mentioned, researchers have used *Support Vector Machines* with *Recursive Feature Elimination* algorithm (SVMs-RFE) to "*build classifier for the whole book and look for the existence of chrono-divide*" (p. 10). SVMs is "*feature ranking method*", which "*ranks the importance of the features according to their weights*" (p. 8). However, the classical linear SVC can be inaccurate, and therefore researchers decided for the "*RFE step*" where the least important features are removed and classifier retrained so that it is more reliable and provides better outcomes (p. 8).

This has initially resulted into 196 most important variables – 144 characters and 48 words, in addition to the mean and variance of sentence length and frequencies of direct speeches and exclamations (p. 10). However, due to randomness in fluctuation of a number of features and their relevance, authors had to introduce a new and "*a more appropriate metric*" called *relative* (instead of absolute) *frequency* of features (p. 11f.). This allowed them to reduce a number of features to between 10 and 50 which they consider to be "*enough to tell the style difference between the two parts*" (p. 12).

Given that the novel is suspected to have a *chrono-devide* between 80th and 81st chapter, authors have also decided to use "*standard technique of separating the whole data into samples [chapters] consisting of training data [1-80; used for training the model] and test data [81-120; used for the validation of the model]*" (p. 7). Once the model has been trained and subsequently applied on the test data, "*results provide an extremely convincing if not irrefutable evidence that there exist clear stylometric differences between the writings of the first 80 chapters and the last 40 chapters. This difference strongly supports the two-author hypothesis for Dream of the Red Chamber*" (p. 13).

This is also further strengthened by testing their method on other three *Great Classical* novels from the set, where authors are not disputed. Hu et al. (2014) confirm this, and thereby they conclude that approach and methodology seems to be correct, reliable and effective (p. 16f).

2.2.2 Modern-Era Examples

2.2.2.1 Authorship Identification in Greek Tweets

Twitter has been a phenomenal social media platform and since its launch in 2006, it has expanded rapidly over the previous decade and today boasts of 320 million active users every month. With support of over 35 languages, Twitter has drastically changed the way information from different users are distributed across the world with internet.

Each tweet is very unique since it has a limit of 140 characters and forces users to come up with short forms and colloquiums to express their thoughts. Naturally this presents an ideal platform for obtaining training data and performing tests on validating stylometric features.

A particular example of special interest is the stylometric analysis is conducted on tweets written in Greek language. Mikros & Perifanos (2013) conducted this experiment with a first Modern Greek Twitter corpus consisting of 12,973 tweets retrieved from 10 Greek popular users. Their objective was to achieve automatic authorship identification, which is not new to stylometry and has wide range of applications such as authorship attribution, verification and profiling (Mendenhall, 1887; Stamatatos, 2009).

In this particular study of authorship attribution in Greek the focus was to develop author's multilevel n-gram profile. Character and word n-grams have already been used successfully previously for automatic authorship identification, and therefore a combined vector of both character and word n-grams resulted into 40 000 features (Bennett, 1976). Based on this vector, researchers were able to create the author's multilevel n-gram profile which represented a document that could capture character and word representations (Mikros & Perifanos, 2013).

Using the n-gram profile they performed the test validation on Twitter's Greek corpus and found that author attribution is indeed feasible and a tweet's linguistic character can strongly indicate the author. By this experiment, they confirmed that using n-grams is definitely one of best features for stylometric analysis.

2.2.2.2 Forensic Stylometry for Anonymous Emails

In recent years there has been alarming increase in cybercrimes, most of which are usually conducted via anonymous or phishing emails. Hence, it is vital today to identify the most probable author of an anonymous email from a set of potential suspects as email has become one of the most widely used tool for committing a range of cybercrimes (Zheng et al., 2006; Iqbal et al., 2008).

In particular interest, it is the experiment conducted by Iqbal et al. in 2008 on an actual company dataset. They employed the frequent pattern technique using which they were able to determine the possible author of an email with "*similar patterns of vocabulary usage, structural and/or stylometric features*" (Iqbal et al., 2008; Agrawal et al., 1993). These characteristics together make the overall pattern or a write-print of an author.

They first started with pre-processing the emails where they removed all blank lines, punctuations, spaces, and then used common discretization techniques to detect presence or absence of patterns. Next, they identified the frequent patterns and finally proceeded towards defining the write-prints of the author. In defining the write-print of the author they considered two feature items, firstly a unique embedded pattern that is present in every email and secondly a piece of writing that is unique to an author in an email.

The conclusion of this experiment was done on a company email dataset containing 200,399 real-life emails from 158 employees and they found that the techniques employed by them were able to achieve an accuracy of 86 to 90 % on a random sample set of emails. The experiment evaluation concluded that they could identify write-prints, determine the author, and finally provide forensic evidence as proof of the identification.

3 Experimental Approach

In our second part of the paper, we are going to examine how one can use stylometry and FB status updates in order to recognize and predict personality type of a person. Indeed, such a *personality recognition* has been described by Celli et al. (2013, p. 1) as "*automatic classification of authors' personality traits, that can be compared against gold standard labels, obtained by means of personality tests*". And as one can assume, there are great number of different tests – usually in the form of questionnaire – which try to assess the personality of a human. For example, the *Myers Briggs Type Indicator* (MBTI) and *Minnesota Multiphasic Personality Inventory* (MMPI) are among them.

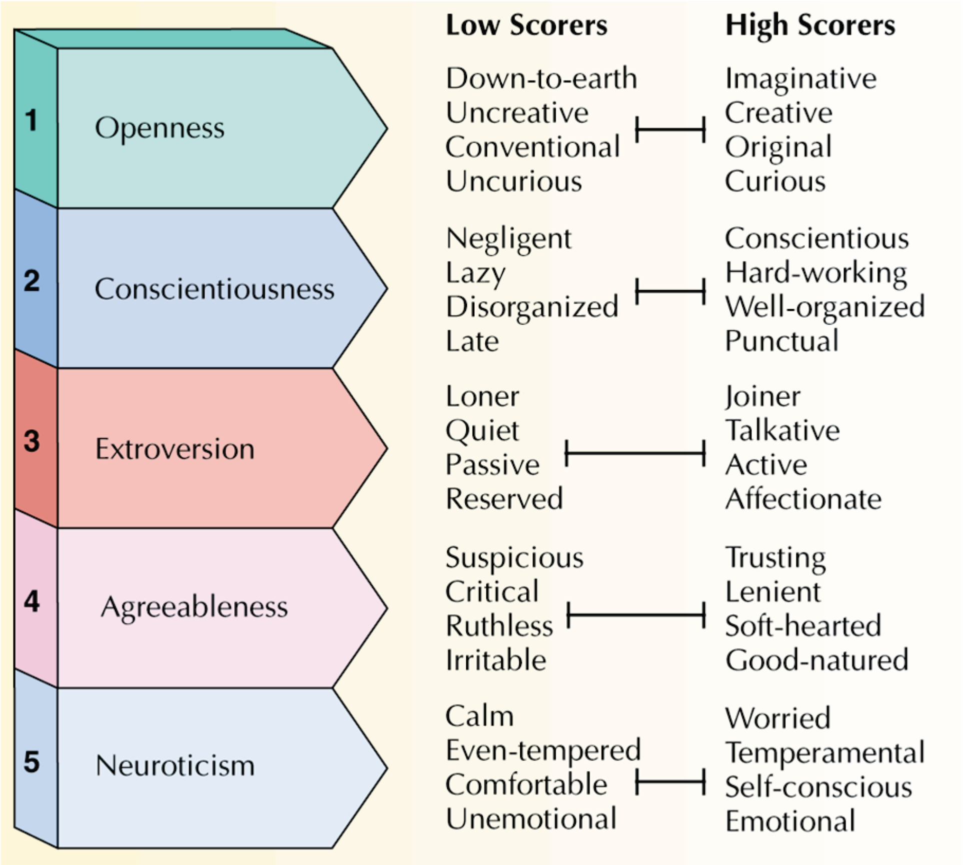


Figure 1 illustrates five personality types and their meaning in terms of scores [ASBECO].

However, for us the relevant group of tests is based on the *Five Factor Model* developed in 1985, usually also found under a name of *Big Five personality traits*. At this point we spare the reader explaining each of five traits individually by saying that (s)he can exercise different scores on each of them. Subsequently, such a score describes his (her) psyche, see figure 1.

One of the reasons for choosing this approach of assessing personality – as Mairesse et al. (2007, p. 2) write – is the fact that it become "*over the last 50 years (...) a standard in psychology and experiments using [it] have shown that personality traits influence many aspects of task-related individual behavior*". These include and are not limited to the leadership and sales ability or the job performance and teacher effectiveness.

In the next chapters we are going to dive into the detail of each section and our approach can be summarized in several steps below:

1. Introduce and describe the dataset (unit 3.1).
2. Present our technology stack and tools used (unit 3.2).
3. Review publically available papers which have conducted the same task using the same dataset (unit 3.3).
4. Talk about our own methodology, conduct various data experiments and examine its results (unit 3.4). Specifically, we follow these iterative steps:
    I. Extract relevant stylometric features from our STATUS column.
    II. Split dataset into the training and testing one using a *cross-validation*.
    III. Train statistical model on training part.
    IV. Test that model on the unseen FB's updates – on the testing part.
    V. Validate the performance of our classification task.
5. Summarize our results and give a research outlook of what are further (interesting) possibilities to study and apply in stylometry (chapter 4 and 5).

(Thornton, 2012; Mairesse et al., 2007)

3.1 The Dataset

The dataset, which Kosinski et al. (2015) kindly provide on the *myPersonality.org Project* website, "*has been collected by David Stillwell and Michal Kosinski by means of a Facebook application that implements the Big5 test (...), among other psychological tests. (...) The application obtained the consent from its users to record their data and use it for the research purposes*" (for more see Celli et al., 2013, p. 2). Consequently, the data consists of 9917 status updates from 250 Facebook users where each one of them can have between 1 and 223 statuses (=rows). These updates "*have been anonymized manually. (…) For instance, each proper name of person [in the status] has been replaced with a fixed string (*PROPNAME*)*" (p. 2). However, publicly known personalities (e.g. "Mozart") or locations, such as "New York" and "Mexico", have not been changed. Besides the author IDs and respective raw strings, the set also includes many other features such as date and time of the update and network properties like network size, density, betweenness, brokerage and others. For each status, additionally, Celli et al. (2013, p. 2) "*included personality labels both as scores [a decimal number 1-5] and classes [binary of type 'y' or 'n']*". The later one, however, will be the only variable – originally coming directly from the dataset together with statuses – which we are going to use in our experiments. All other variables are going to be extracted from status updates.

Speaking about the content, our corpus contains various tokens including for example emoticons ("<3", "☺"), slang and abbreviations ("OMG!!!", "LOLZ"). Indeed, basically all typical internet expressions (for the social media networks) can be found here. In order to preserve their full meaning and consistency, we do not pre-process the statuses (either manually or automatically) before. Thus, also not grouping and joining statuses by users as this would give us a very small sample of just 250 users with their statuses joined all in one large string.

3.2 Our Working Approach, used Tools and Libraries

In our team we used GitHub[3] – the collaborative code sharing application — which for us was very beneficial not only due to source code management but also because of combining lightweight bug tracking system, wiki and many other features.

In addition, we also used *Jupyter Notebook*[4] (formerly *IPython*) technology which is supported by GitHub and allows to display "*and share documents that contain live code, equations, visualizations and explanatory text*" (Pérez & Granger, 2007). Such notebooks can include different code snippets that will be executed by various languages, including e.g. *Python*, *R* or *Haskell*. Therefore, these notebooks – among other reasons –

[3] https://github.com/dmpe/CaseSolvingSeminar

[4] https://jupyter.org/

are usually used in the field of *Data Science* where the researcher is creating a story-line (a "timeline") of how (s)he acquires, explores, pre-processes, analyses and finally visualizes data in order to gain valuable insights. All that by applying statistical or ML methods and briefly describing each step.

As a result, in such cases, there is usually no need to use third-party editors such as VIM, PyCharm or others because they cannot supplement the notebook functionality, e.g. rendering the markdown [PythonIDEs]. Nonetheless, we have also used them – most notably the abovementioned VIM – for bigger programming tasks where interactivity of code was neither required nor desired.

Besides that – as we already knew at the beginning of our project – we would be using two major Python packages. One of them is the *NLTK* which is the most advanced and open-source *natural language processing* library (Bird et al., 2009). Additionally, given our task, it was also clear to us that some ML libraries are going to be used as well. Even though *NLTK* library has out-of-the-box limited functionality for it, it has also developed a more robust (albeit small) wrapper[5] around *scikit-learn* library (Pedregosa et al., 2011). Fortunately, there is nothing that hinders us from accessing, using and combining *scikit-learn* with *NLTK* directly.

To conclude this section, in the figure 3 in the appendix the reader may find our software environment. In particular, the names and versions of all Python 3 extensions that we have used in order to run and reproduce our results.

3.3 Review of Literature

The *myPersonality.org* (Celli et al., 2013) dataset has already been utilized by other researchers and on the next page, we examine two of them and shortly present their outcomes.

Farnadi et al. (2013) used the same corpus of 250 FB users and 9917 status updates, however they treated users' statuses as one big string (for more see end of ch. 3.1).

[5] http://www.nltk.org/api/nltk.classify.html#module-nltk.classify.scikitlearn

They obtained their results while using four groups of stylometric features by using *Weka* (Witten & Frank 2005) and were able to compare outcomes of three ML algorithms: *Support Vector Machines* with a linear kernel, *Nearest Neighbor* with k=1 (kNN) and *Naive Bayes* (NB). They concluded that "*there is no single kind of features [from the set of four] that gives the best results for all personality traits*" and generally were able to achieve a precision lying between 0.40 and 0.71 (see ch. 3.4.4; Farnadi et al., 2013).

Another set of researchers, Alam et al. (2013), also used the same corpus (albeit without joining statuses by users) and they made a comparative study between three classification methods: SMO (*Sequential Minimal Optimization* for *Support Vector Machines*), Bayesian Logistic Regression (BLR) and *multinomial* Naïve Bayes (MNB) *sparse model*. Regarding the features, they took only "*bag-of-words approach and used tokens (unigrams) as feature*", later applying TF-IDF algorithm (Alam et al., 2013). As a result, they were able to find that the *MNB sparse modelling* performs better than SMO or BLR method – by achieving a precision of around 58.5 % ± 0.9.

3.4 Experiments

3.4.1 Extract Relevant Stylometric Features

As briefly mentioned in the section 3.1, we are going to use only one variable from the original dataset (ODS), see table 1. Therefore, it is naturally necessary to extract additional stylometric features from the statuses themselves. In order to accomplish that we combine the strengths of R-language[6] with Python's NLTK resulting into new features being added to our dataset.

Our task is to conduct a *supervised* machine learning in order to *classify* users' statuses to one of five *binary* traits which are the labels (five output variables; e.g. "cNEU" that can be 'yes' or 'no'). Therefore, we construct a statistical model that should be "*able to predict the label of an object given the set of [our 14] features*", the input variables (VanderPlas, 2013).

[6] https://www.r-project.org/

<table>
<tr><th rowspan="2">Labels (5)*</th><th rowspan="2">Features used from ODS (1)*</th><th colspan="2">Extracted from statuses</th></tr>
<tr><th>Lexical (6)*</th><th>Character (8)*</th></tr>
<tr><td>cNEU</td><td>STATUS</td><td># functional words
|| smileys</td><td>string length
|| # commas</td></tr>
<tr><td>cAGR</td><td></td><td>lexical diversity [0-1]</td><td># words || # dots</td></tr>
<tr><td>cOPN</td><td></td><td># personal pronouns</td><td># semicolons || # colons</td></tr>
<tr><td>cCON</td><td></td><td>Bag-of-words (n-grams)</td><td># *PROPNAME*</td></tr>
<tr><td>cEXT</td><td></td><td>Parts-of-speech Tags</td><td>average word length</td></tr>
</table>

*Table 1 shows features (variables) that we are going to use in our experiments for the supervised learning. '#' denotes 'number of …'; * (star) denotes sum of number of features in the column; brackets next to the features denote what values they can take on. We have been inspired by the work of Gupta et al. (2014) and Anuraag et al. (2014) who did the same task of predicting the psychological traits from the same dataset and publically published their reports with the code associated.*

3.4.2 Split Dataset using Cross-Validation

Once we had our features extracted and given that at first stage we wanted to predict only one label from our features at a time, we divided our whole dataset into five parts according to the binary labels. Practically, at the end we had five different datasets (with all 9917 rows) containing all 14 features with the corresponding one binary trait variable (e.g. "cNEU"). We call this a *trait dataset*.

Subsequently, we proceeded to the next step of splitting each of *trait datasets* into the training and testing set – a usual ML procedure in order not to run into the *overfitting* situation due to the learning and testing the model on the same data [CVEEP16]. Indeed, then, such a model would result into an excellent accuracy but would fail to predict labels on the similar, yet unseen data.

As described, to avoid that problem, we create the training and testing set from the *trait dataset*. As a result, we train our classifier on the training data and then we would examine its quality on the hold-out test data. However, methods such as Naïve Bayes or SVMs usually offer many additional *hyperparameters* for further fine-tuning the performance. In fact, for example, just by changing the default value of the *loss* and/or *penalty* parameter, it could result into a (slightly) better metrics of the whole *Linear SVC* classifier, for more see section 3.4.4.

Although two datasets can suffice for many cases, one may again run into the overfitting issue due to specific tuning of hyperparameters to "*perform(s) best on the testing set*" (Head, 2015). A solution to this problem is again to split data but this time into three sets, having additionally a *validation* dataset – so-called *dev* set.

On the one hand, this is able to overcome the issue of hyperparameters that "*can be tweaked until the (...) [classifier] performs optimally*", but also on the other hand – and in our particular case – creating three sets of data would make our samples small [CVEEP16]. Even though, Celli et al. (2013) write that researchers "*were free to split the training and test sets as they wish*", the same authors also "*suggested to use Weka (Witten & Frank 2005) with 66% training and 33% test splitting*"[7]. We are going to follow their suggestion only partially and rather instead we will use a particular form of statistical sampling called *stratified k-fold cross-validation*.

A regular k-fold cross-validation splits data into *k* samples and trains the model *k* times, "*treating a different chunk as the holdout set each time*" (Thornton, 2012). Then, it averages the performance measure from each iteration, allowing us to compute the *standard deviation* of our performance metrics (Head, 2015; Astroml.org, 2012). The stratification then additionally "*rearrange[s] the data (...) to ensure each fold is a good representative of the whole*" dataset (Liu & Zsu, 2009).

From a practical point of view, for example, our binary label "cNEU" has 3717 (37.5%) cases of 'yes' and 6200 (62.5%) cases of 'no'. If only *k-fold cross-validation* was applied, there could have been folds where one of them would contain 90% of 'yes' cases and only 10% of 'no' cases. Whereas in a different fold, it would be the opposite. As this is clearly undesirable behavior and would result into having an unbalanced chunk of data, *stratification* ensures that each fold is representative of the whole dataset and class labels (here of "cNEU" label) are also distributed according to their proportion in it. Meaning that each of our ten folds should then contain approximately a ratio of 1 'yes' class to 1.66 'no' classes (Shams, 2014).

[7] http://www.cs.waikato.ac.nz/ml/weka/index.html

3.4.3 Train and Test the Model

In order to determine and predict the personality trait (i.e. a document's label) from a sentence or status post, we used several (supervised) ML algorithms for our binary text classification task. The general usage of *classifiers* includes two steps: the *training* process with labelled data and *classifying* with test ones.

In many cases, the initial labelling of data used for training a classifier has to be done manually by humans. This data can be then used to train a model to match documents to different classes of labels and such a result of a training is a classifier which is capable of matching yet unknown documents. Given the rules it derived during its training process, now – in our case — it should be able to say if a status can be classified to a certain personality trait (i.e. for example 'yes' to the notion that a person behind that status was neurotic at the time of writing it) (Manning et al, 2008, pp. 253-257).

As already known, all data provided by *myPersonality.org* have been already labelled, and thus we didn't have to deal with it and could create a train and test datasets as mentioned already in the previous chapter. Therefore, once, we have created our *trait datasets* with appropriate *stratification* of class labels, we would continue conducting different data experiments. In fact, we tested four different classes of learning algorithms from the *scikit-learn* library. There were the following:

- two *Support Vector Classifiers* (*Linear SVC* and *SVC* with *rbf kernel*)
- two *Naïve Bayes* methods (*multinomial-* and *bernoulli-NB*)
- *k-Nearest Neighbors* algorithm and
- finally, two *ensemble methods* (*Random Forests* and *AdaBoost*)

Even though our results will be presented in next section 3.4.4, here in this chapter, we are going to give a brief overview of a theory behind two classifiers which produced the best results during our experiments and further introduce the reader to terms such as *TF-IDF*, *pipeline* and *grid search*. Lastly, we briefly introduce our *best-results final* model.

3.4.3.1 Text Feature Extraction

Most machine learning classifiers require their input documents in a specific form. A common form are vectors which give information about frequency of certain words within a document. Especially important are *term frequency–inverse document frequency*-vectors (TF-IDF). Their special characteristic is to weight each word of a document based on its frequency within the document and all documents known to a classifier. As a result, words appearing very frequently or throughout all documents will be rated as less important by TF-IDF while more unique words are believed to have a higher significance (Manning et al, 2008, pp. 117-119).

TF-IDF also counts towards a group of *bag-of-words* implementations. This describes a concept of ignoring the order of components (e.g. words) within a document and focusing on the number of occurrences instead. Even if this abstraction removes all meaning from a document which was built by its syntax structure, one can assume that documents consisting of the same words revolve around the same topic as shown in table 2 (Manning et al, 2008, pp. 117.).

Sentences	You like thunder, don't you	You don't like thunder
Bag of Words (frequency)	{'you': 2, 'like': 1, 'thunder': 1, 'don\'t': 1}	{'you': 1, 'don\'t': 1, 'like': 1', 'thunder': 1}

Table 2 illustrates an example of bag-of-words

Another concept tightly connected to TF-IDF are n-grams. Because plain TF-IDF creates a bag-of-words common combinations of words such as "Mozilla Firefox", "smart home" will be separated and their special meaning will get lost. To prevent this meaningful connection of words getting split up, n-grams had been introduced. N-grams create collections of *n* successive words (whereas $n \in \mathbb{N} \setminus \{0\}$) which will be then used to create TF-IDF vectors rather than single words, see table 3 (Perkins, 2010, pp. 25-27).

Sentence	I hate Microsoft Word
n-gram	('I', 'hate'), ('hate', 'Microsoft'), ('Microsoft', 'Word')

Table 3 shows n-gram example with n = 2.

3.4.3.2 Support Vector Machines

Talking about SVMs, it is necessary to say that they can be used for the classification as well as regression tasks. What they basically do is they "*draw a boundary between clusters of [training] data*", mathematically trying to "*find(ing) the best hyperplane that separates all data points of one class from those of the other class*" (Vanderplas, 2015; Matlab, 2016). Given the variability of data in the real world, SVMs also allow to use different *kernel* methods (equations) such as *radial basis function (rbf)* or *Gaussian*. For our purposes, we used the *linear* and *rbf kernel* equations which in our results proved to be the best.

3.4.3.3 Naïve Bayes

"*The Naïve Bayes classifier is based on the Bayes rule of conditional probability. It makes use of all the attributes contained in the data, and analyses them individually as though they are equally important and independent of each other*" (Padhye, 2016).

During the training phase, *multinomial* NB gets labelled documents where a label represents a class (e.g. 'yes') and a document consists of terms (statuses). For every such term in a document, NB calculates how likely this term indicates the label. When querying the NB classifier to which class a certain document belongs, it calculates for each class a probability and returns the one with the highest value (Manning et al, 2008, pp. 258). Nonetheless, we have also used *bernoulli* NB which works in a similar fashion.

3.4.3.4 Pipeline and FeatureUnion

When working with the *scikit-learn* library, we also stumbled across two concepts called *Pipeline* and *FeatureUnion* [Pipe16]. Both of these allow to chain multiple classes (e.g. estimators or transformers) into one single unit, an interface, which can then be used as whole for training and classifying the data.

Due to the use of many different algorithms, this has allowed us to build a complex model, resulting into improving (to some degree) our results. Additionally, they provide a great assistance for keeping the source code short and clean. Since *Pipeline* and

FeatureUnion are a mere construct for managing dataflow within a model, we are not going to discuss them here more in detail as the reader can see their documentation.

In the *business intelligence*, for example, there is a process called the *extraction, transformation* and *loading* (ETL). And precisely this can be portrayed in such a pipeline as three different, distinct and sequential steps to execute on the data in order to store them in database.

3.4.3.5 Grid Search

As already described, we tested multiple classifier algorithms and different methods of features extraction and transformation on our dataset. Naturally, most of these functions can receive multiple parameters which can have great influence on the performance and results. As an example, *multinomial* NB delivered better results when TF-IDF was set to use n-grams ranging from to 2-3, instead of 1-2.

Since we used various classifiers and transformers, it would have been very hard and time-consuming to manually execute each combination of classifier and transformer for each of our labels and keep track of all results in order to filter out the best ones. Therefore, we utilized *scikit-learn's grid search*[8] which exhaustively tries all parameters provided to find the combination returning best performance. How we differentiated between good and bad results is further explained in chapter 3.4.4.

3.4.3.6 Best Performing Statistical Model

Our model, which achieved best results, combined several different aspects, see figure 4. Besides using the abovementioned *grid search* with just one parameter – the n-gram range – we had to append, in each iteration, one of the several classifiers we decided to use. Indeed, for the final results, we choose to compare only four models, namely the *linear* SVC, SVC with *rbf kernel*, *multinomial* and *bernoulli* Naïve Bayes. All of these showed us the largest potential for achieving best outcomes.

Our pipeline consisted of four distinct and independent steps to accomplish. The first one was to take the status column and convert its raw statuses "*to a matrix of TF-IDF*

[8] http://scikit-learn.org/stable/modules/grid_search.html

features" [TfIDF16]. The second one was to take the status column again, but this time first to convert it to *part-of-speech* tags and then later to a matrix of TF-IDF features. The third one was to use statuses (again as TF-IDF features) but without smileys which we would have extracted before. The last step was to aggregate extracted 11 numeric features such as number of words, dots or lexical diversity and standardize them to a range between 0 and 1.

By using *FeatureUnion*, we "*combine[d] several transformer objects [our four pipelines] into a new transformer that combine[d] their output*" [Pipe16]. Then, we were ready to fit and predict the data as each of four pipelines would "*fit to the data independently*". They would be applied in parallel "*and the sample vectors they output are concatenated end-to-end into larger vectors*" [Pipe16]. With this simplification, it allowed us to create a single large pipeline on which we would be calling the classical *fit* and *predict* functions.

3.4.4 Examine the Results and Classifiers' Performance

Once we finished with the iterative process of training and testing the model on our dataset, we could check the performance of the used classifier. For that, Celli et al. (2013, p. 3) "*suggest[ed] to use Precision, Recall and F1-measure to evaluate predictions over [the binary] classes [e.g. "cEXT"]* ".

We additionally added an *accuracy* to our mix which is a "*proportion of the total number of predictions that were correct*" and its value is between 0 and 1, i.e. 1 ~ 100% of all predicted values through the classifier were correct (and thus same to the truth values of the ODS) (Sayad, 2011). In order to visualize this and many other classification metrics described later, let us now introduce the reader to a *confusion matrix*, see an example of such in table 4. This classical (ML) evaluation method shows a 2x2 matrix where two (grey) columns represent the number of cases in the original dataset while two (grey) rows represent the number of cases that have been correctly or incorrectly predicted by different classifiers.

Going back to our first metric, which is close to 60% (1) showing us relatively high accuracy of prediction, it can also be very unreliable and sometimes even misleading.

Confusion Matrix for “cEXT” label using *Bernoulli-NB*		Target cases (the reference from ODS)		
		Positive ‘yes’	Negative ‘no’	
Classified by model (predicted values)	Positive	256 (True positive)	226 (False Positive = Type 1 error)	**Precision**
	Negative	1175 (False Negative = Type 2 error)	1715 (True Negative)	
		Sensitivity/Recall	**Specificity**	**Accuracy**
		= 1431	= 1941	Total: 3372

Table 4 shows an example of a confusion matrix applying Bernoulli-NB classifier while using only 13 stylometric features, which could have been observed in ‘derived_columns_only’ notebook in /src/ folder. This matrix is adapted from Sayad’11.

$$\text{Accuracy} = \frac{1715\ (TN) + 256\ (TP)}{1715\ (TN) + 226\ (FP) + 1175\ (FN) + 256\ (TP)} = 0.58 \qquad (1)$$

$$\text{Precision} = \frac{256\ (TP)}{256\ (TP) + 226\ (FP)} = 0.53 \qquad (2)$$

$$\text{Sensitivity/Recall} = \frac{256\ (TP)}{256\ (TP) + 1175\ (FN)} = 0.18 \qquad (3)$$

$$\text{F1-score} = 2 * \frac{0.53\ (precision) * 0.18\ (recall)}{0.53\ (precision) + 0.18\ (recall)} = 0.27 \qquad (4)$$

The problem occurs “*when the number of negative cases is much greater than the number of positive cases*” (Hamilton, 2012; Kubat et al., 1998; Brownlee, 2014). As Hamilton (2012) further writes “*suppose there are 1000 cases, 995 of which are negative cases and 5 of which are positive cases. If the system classifies them all as negative, the accuracy would be 99.5%, even though the classifier missed all positive cases*” (for more see Descoins, 2013). Due to this *accuracy paradox*, a model with a lower accuracy can be actually much more useful and have higher predictive power. As a result, this measure should be applied only with a great caution and fortunately there are other metrics which are recommended, namely the *precision* and *recall* (also called *sensitivity*).

Both of these two metrics go hand in hand and higher their number, the better the classifier. Indeed, “i*f the classifier does not make [any] mistakes, then precision = recall = 1.0*” (Descoins, 2013). This is, however, extremely rare to achieve in real world. Hence, it is usually the case to have one of these metrics relatively high whereas the other one is lower. In the example stated above by Hamilton (2012), the value of 995 is now not important at all.

Given the recommendation of using them over the *accuracy*, we begin with a *precision* which is defined as "*the proportion of positive cases that were correctly identified*" (Sayad, 2011). It basically tries to answer the following question: "*out of all the examples the classifier labelled as positive, what fraction were correct*" (Descoins, 2013)? In our case, in table 4, the precision of the *bernoulli* Naïve Bayes is relatively high of 53% (2), i.e. that about a half of the cases it did label was indeed correct (Zembowicz, 2016). In fact, the model is relatively relevant and precise.

Secondly, there is also a *recall* which is the opposite measure and it tells "*the proportion of actual positive cases which are correctly identified*" (Sayad, 2011; Manning, 2016). Descoins (2013) paraphrases this as "*out of all the positive examples there were [from ODS], what fraction did the classifier pick up*" right? As a matter of fact, between both abovementioned metrics there is usually an inverse relationship where if the recall increases, the precision commonly decreases and vice-versa. In our example, this can be observed as the recall is very low, just around 18% (3) meaning that we have high number of 'yes' cases in the ODS which "*remained unidentified*" correctly by the model (*there are 1431 true cases but we have only identified 256*; Zembowicz, 2016; Castro Da Silva, 2014). Here, one can argue that our model is not very sensitive.

As a result, it is necessary to balance out this relationship as there is a clear trade-off between a more conservative (higher precision but reduced recall) vs. more a liberal (lower precision but higher recall) model (Zembowicz, 2016). Choosing the right one is very application specific and for example in fraud detection done in banks it is better to have higher recall rate (i.e. all fraudulent transactions are identified) maybe at a loss that some of which won't be identified as fraudulent (Descoins, 2013). On the other hand, for our purposes of identification of personality traits, we would be considering the precision to be more relevant as we want to know what the quality of our model is.

Trait Dataset	Achieved Results with CV = 2				Best Algorithm
	Accuracy Mean	Recall	Precision	F1-score	
NEU	0.62	0.32	0.54	0.39	Linear-SVC
OPN	0.74	1.0	0.87	0.85	Bernoulli-NB
AGR	0.53	1.0	0.76	0.69	SVC
EXT	0.60	0.46	0.61	0.49	Linear-SVC
CON	0.59	0.53	0.65	0.54	Linear-SVC
Average	**0.61**	**0.66**	**0.69**	**0.59**	--

Table 5 displays our final results, which could have been observed in 'derived_columns' notebook in /src/ folder, after applying different learning algorithms on each of 5 trait datasets with all 14 stylometric features.

Lastly, we want to talk about the *F1-score* which is defined as a *harmonic mean* of both measurements – in our case 27% (4) and it precisely tries to "*convey(s) the balance between the precision and the recall*" (Brownlee, 2014; Castro Da Silva, 2014). Because of that we have considered it as our evaluation metric, to a smaller extent with accuracy, for the individual results of different classifiers on our *trait datasets*, see table 5.

And as one can see, the final results vary greatly as predicting for example the "NEU" trait proved to be the most challenging task with F1-score of only 0.39. On the other side, *bernoulli* NB performs best when predicting the "OPN" trait. Especially this trait is very easily and correctly predictable, achieving very good results where none of values is less than 80% for both F1-score, recall and precision.

3.4.4.1 Graphical Visualization of Other Metrics

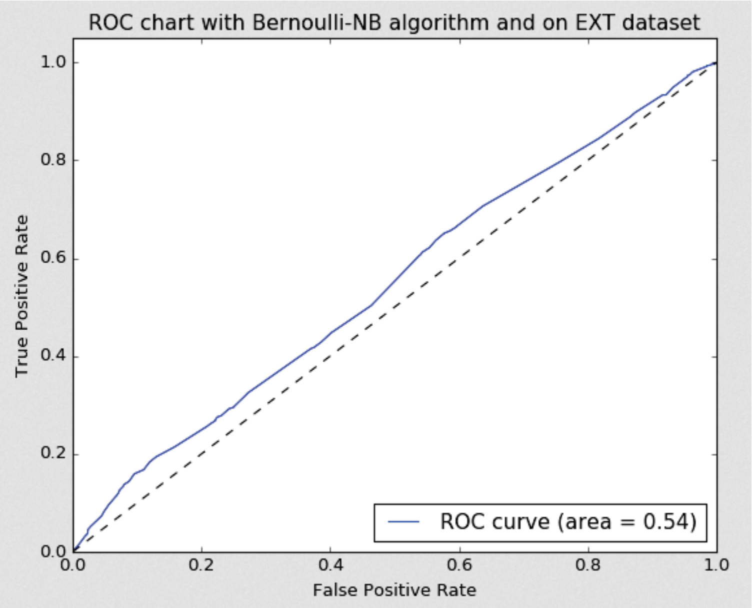
Up until now, we have largely considered numeric-only metrics due to the use of the confusion matrix. Therefore, in this section we want to take a look on another set of metrics which can be easily visualized through a chart.

The main metric we are going to consider here is so-called *Receiver Operating Characteristic* (ROC) graph, see an example for such in figure 2. This is used to "*evaluate classifier output quality*" and it "*is a plot with the false positive rate on the X-axis [calculated as 1 - specificity] and the true positive rate on the Y-axis [the sensitivity]*" (Hamilton, 2012b; ROC16). Hamilton (2012b) further continues by explaining that "*the* *point (x=0, y=1) is the perfect classifier: it classifies all positive cases and negative cases correctly. It is (0,1) because the false positive rate is 0 (none), and the true positive rate is 1 (all) (…) [On the other hand] point (1,0) is the classifier that is incorrect for all classifications*".

In order of being able to analyse the ROC curve, is necessary to tell that "*it is ideal to maximize the true positive rate while minimizing the false positive rate (…) This means that [the curve touching] the top left corner of the plot is the 'ideal' point - a false positive rate of zero, and a true positive rate of one*" (ROC16). And basically, it "*shows the trade-off between sensitivity and specificity (any increase in sensitivity will be accompanied by a decrease in specificity). The closer the curve follows the left-hand border and then the top border of the ROC space, the more accurate the test*", our prediction (Tape, 2003).

From the top picture in figure 2, we can clearly observe that the curve is rather at close to the dotted line, i.e. that our prediction performance using *Bernoulli-NB* on the *extroversion* trait is rather very poor and "*[very similar to a] (…) random*" (Vogler, 2015). As a result, we ought to change our model for predicting "EXT" trait as it is almost not any better than assigning personality traits to statuses at random. This compares somewhat sharply to the picture below as this begins to show preferred bending of the curve using the final pipeline as our classifier.

Moreover, ROC chart has one additional metric which we can derive from it − namely the *area under curve* (AUC). This "*quantifies the overall ability of the test [prediction] to discriminate between those*" traits which have true ('yes') and those which have false ('no') values (GraphPad, 2015). Same authors further continue, "*a truly useless test [prediction] (one no better at identifying true positives than flipping a coin) has an area of 0.5 [or under]. A perfect test [prediction] (one that has zero false positives and zero false negatives) has an area of 1.00*". And as we can see in the first image, our AUC is indeed just 0.54 confirming that 13 features alone with *Bernoulli-NB* classifier are not well suited for the prediction of extraversion for Facebook's statuses. However, when additionally adding a bag-of-words (n-grams) feature while still applying the same classifier, we can clearly see the improvement of this metric (Schoonjans, 2016).



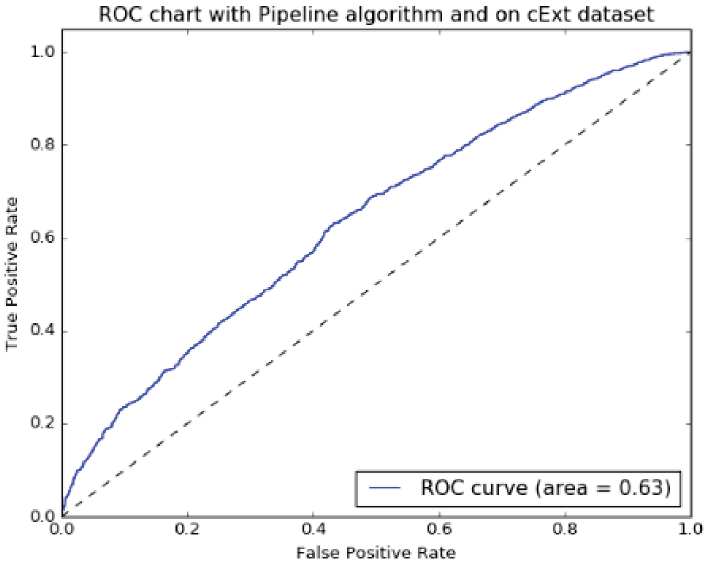


Figure 2 shows two ROC charts for "EXT" trait, one predicted using solely 13 features and applying Bernoulli-NB from table 2, while another one with the pipeline we developed later (this is to be found in 'derived_columns' notebook in /src/ folder).

4 Future Research

Throughout our work on the project we came across several interesting aspects. Firstly, in many ML tasks — be it a training a model or manipulating data – they proved to be very CPU intensive, and thus also very slow.

A potential solution for that is to apply NVIDIA GPUs and its CUDA processors for *general-purpose* computing (GPGPU). In fact, nowadays, not only modern GPUs can perform 3D rendering in computer games with millions of polygons but also can run compute-intensive jobs such as for example in the *artificial intelligence* field.

In order to use GPGPU in ML, one can use many freely available open-source projects such as *PyCUDA*, *Numba* and other *deep-learning* frameworks like *Theano* and *Cafee* [GPUlibs]. All of them allow to train models combining both the power of GPUs and CPUs and indeed, when looking at today's commonly used two-, four- or eight-core CPUs, NVIDIA TESLA K80 delivers as many as 4992 specially-optimized CUDA cores. As a result, these can offer faster performance up to 10 times in computing (NVIDIA, 2015). A disadvantage, however, is very high price ranging between 4000-5000 USD.

Next, one can also look into using cluster computing systems, among which the most popular seems to be *Apache Spark*. By using many cheap VPS from cloud providers such as *Amazon AWS*, a researcher is able to run many distributed machine learning tasks on a cheap *commodity hardware*, costing smaller amount of money.

5 Summary

5.1 Milestones and results

After thoroughly researching and understanding about the theoretical aspects of this project (e.g. the vast field of language processing and machine learning), we started out with the practical implementation. While working on our project, we have conducted many experiments on *myPersonality.org*'s data and continued to improve our initial results.

Our first experiments were rather of a primitive nature. We created different models using *scikit-learn's* transformers and estimators and used them on our split data. When analyzing our first results, we realized that each personality trait is very unique and that it would be unlikely to find a classifier which is capable to deliver good estimations for all labels at once.

We continued using *NLTK* library for creating stylometric features including for example lexical diversity, average word length or part-of-speech tags. In contrast to our initial belief, we could hardly achieve any progress by using our classifiers on our derived features. As a matter of fact, the best prediction results for any personality trait were slightly worse than our first attempt.

On the other hand, our final model included using both statuses and derived values in a pipeline, see section 3.4.3.6. For some personality traits, we managed to produce better results (in comparison to our first try), however, others could not have been improved.

With status posts consisting of ~80 characters on average, it (was and it) is troublesome to retrieve useful features. Therefore, we also tried to aggregate all status posts from each person into a single corpus in order to have larger text. We expected to get better results by applying classifiers on lexical features derived from such a larger corpus. However, the expected improvement has not been achieved. So we did not continue with this approach any further.

5.2 Problems

During our research we encountered some hardships as we discovered that the dataset which we worked with had some flaws. For one, there are some statuses which had been written in a language other than English. Since we had no reliable way of finding non-English phrases, we had no other choice but to accept that there are sentences that will cause some natural language algorithms fail. This, for example, had also influence on TF-IDF vectors. Indeed, when assuming that there is a single short Greek sentence in the whole dataset, TF-IDF will consider Greek words to be more important and rank them consequently very high.

Moreover, as already mentioned, the average length of statuses was very short, just around 80 characters. Also the distribution of posts per person was very odd (the average count of posts per person was about 40 by ~10,000 posts in total). We assume that in order to achieve better results, it might have been better to have much larger dataset for such experimentations.

Lastly, due to our lack of powerful computer hardware, searching for example for a models' best parameters with *grid search* was extremely time consuming task. In fact, it took us many hours to calculate a few parameters, and therefore we ended up with using only one. As described in chapter 0, using more sophisticated hardware or a cluster of computers could have been a great assistance for us.

6 Appendix

```
> sudo -H pip3 list.txt

https://gist.github.com/dmpe/3a8987e9197b86b636ba
```

Figure 3 shows a command which has been used to create a list of all Python 3 installed packages and their versions on our OS. The reader is strongly encouraged to install them (all) from the included link in order to reproduce our code. For development purposes Ubuntu OS (see picture in GitHub link) has been used.

```
base_pipeline = sklearn.pipeline.Pipeline([
    ('features', sklearn.pipeline.FeatureUnion(
          transformer_list=[
           ('status', sklearn.pipeline.Pipeline([
              ('tf_idf_vect', sklearn.feature_extraction.text.TfidfVectorizer()),
            ])),

            ('derived_string', sklearn.pipeline.Pipeline([
                ('part_of_speech', csstransformer.PartOfSpeech()),
                ('tf_idf_vect',
sklearn.feature_extraction.text.TfidfVectorizer()),
            ])),

            ('derived_string (smileys)', sklearn.pipeline.Pipeline([
                ('smileys', csstransformer.Smileys()),
                ('tf_idf_vect',
sklearn.feature_extraction.text.TfidfVectorizer(vocabulary=csstransformer.Smileys
.smileys, stop_words=None)),
            ])),

            ('derived_numeric', sklearn.pipeline.Pipeline([
              ('numeric_aggregator', csstransformer.Aggregator([
                    csstransformer.SentenceLength(),
                    csstransformer.NumberOfWords(),
                    csstransformer.NumberOfCommas(),
                    csstransformer.NumberOfDots(),
                    csstransformer.NumberOfSemicolons(),
                    csstransformer.NumberOfColons(),
                    csstransformer.LexicalDiversity(),
                    csstransformer.AverageWordLength(),
                    csstransformer.NumberOfFunctionalWords(),
                    csstransformer.NumberOfPronouns(),
                    csstransformer.NumberOfPropnames(),
                ])),
              ('scaler', sklearn.preprocessing.MinMaxScaler()),
            ])),
          ],
    )),
])
```

Figure 4 shows our pipeline which performed best (without different classifiers where each of them has been later appended to it). For more see 'derived_columns' notebook.

7 Reference list

[ASBECO] – Source of image: Drenth, Hans. "*MBTI Gebruiken Bij Marketing; Kan Het Ook Anders?*" 25 Mar. 2015. Web. 13 Mar. 2016. http://www.canicas.nl/visie/canicas-sex-and-the-city-model/mbti-gebruiken-bij-marketing-kan-het-ook-anders/

[CVEEP16] – "*3.1. Cross-validation: Evaluating Estimator Performance.*" scikit-learn. Web. 13 Mar. 16. http://scikit-learn.org/stable/modules/cross_validation.html

[GPUlibs] – https://github.com/inducer/pycuda; https://github.com/numba/numba; https://github.com/Theano/Theano; https://github.com/BVLC/caffe

[Pipe16] – "*4.1. Pipeline and FeatureUnion: Combining Estimators.*" scikit-learn.Web. 10 Mar. 2016. http://scikit-learn.org/stable/modules/pipeline.html

[PythonIDEs] – http://www.vim.org; https://www.jetbrains.com/pycharm; https://en.wikipedia.org/wiki/Markdown

[ROC16] – "*Receiver Operating Characteristic (ROC).*" Web. 13 Mar. 16. http://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html

[TfIDF16] – "*Sklearn.feature_extraction.text.TfidfVectorizer*". scikit-learn. Web. 10 Mar. 2016. http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

Abbasi, A., Hsinchun, C., & Nunamaker Jr., J. F. (2008). *Stylometric Identification in Electronic Markets: Scalability and Robustness*. Journal Of Management Information Systems, 25*(1), 49-78.*

Abou-Assaleh, T., Cercone, N., Keselj, V., & Sweidan, R. (October 2004). *Detection of New Malicious Code Using N-grams Signatures*. In *PST* (pp. 193-196). http://index-of.co.uk/virii/D/Detection%20of%20New%20Malicious%20Code%20Using%20N-grams%20Signatures.pdf

Agrawal, R., Imieliński, T., & Swami, A. (1993). *Mining association rules between sets of items in large databases*. ACM SIGMOD Record SIGMOD Rec., 22(2), 207-216.

Alam, F., Stepanov, E. A., & Riccardi, G. (2013). Personality traits recognition on social network-facebook. In *Proceedings of the Workshop on Computational Personality Recognition* (pp. 6-9). http://clic.cimec.unitn.it/fabio/wcpr13/alam_wcpr13.pdf

Albert C.-C Yang, C.-K Peng, H.-W Yien, Ary L Goldberger, *Information categorization approach to literary authorship disputes*, Physica A: Statistical Mechanics and its Applications, Volume 329, Issues 3–4, 15 November 2003, Pages 473-483, ISSN 0378-4371, http://dx.doi.org/10.1016/S0378-4371(03)00622-8

Anchiêta, R. T., Neto, F. A. R., de Sousa, R. F., & Moura, R. S. (2015). *Using Stylometric Features for Sentiment Classification*. In *Computational Linguistics and Intelligent Text Processing* (pp. 189-200). Springer International Publishing. http://link.springer.com/chapter/10.1007/978-3-319-18117-2_15

Anuraag A., Reddy N., Bharadwaj N. , Balwani, S., "*Personality Detection Group: 52*", 17.4.2014, https://github.com/anuraagvak/IRE-PersonalityRecognition-Final/blob/master/ire_report.pdf ; GitHub: https://github.com/anuraagvak/IRE-PersonalityRecognition-Final

Argamon, S., Koppel, M., Pennebaker, J. W., & Schler, J. (2009). Automatically profiling the author of an anonymous text. *Communications of the ACM*, *52*(2), 119-123. https://dl.acm.org/citation.cfm?id=1461959

Astroml.org (2012). "*2. Machine Learning 101: General Concepts.*", 16 Nov. 2012. Web. 13 Mar. 16 http://www.astroml.org/sklearn_tutorial/general_concepts.html

Bethell, T. (1991). *The Case for Oxford. Atlantic Monthly*, 268 (4), 45–61.

Bird, Steven, Edward Loper and Ewan Klein (2009), *Natural Language Processing with Python*. O'Reilly Media Inc. https://www.nltk.org

Brownlee, Jason (2014). "*Machine Learning Mastery - Classification Accuracy Is Not Enough: More Performance Measures You Can Use.*" 21 Mar. 2014. Web. 13 Mar. 16. http://machinelearningmastery.com/classification-accuracy-is-not-enough-more-performance-measures-you-can-use/

Burrows, S. (2010). *Source code authorship attribution*. http://researchbank.rmit.edu.au/view/rmit:10828

Castro Da Silva, Bruno (2014). "*What Is an Intuitive Explanation of F-score? - Quora.*" 1 Aug. 2014. Web. 13 Feb. 2016. https://www.quora.com/What-is-an-intuitive-explanation-of-F-score

Celli F., Pianesi F., Stillwell D., Kosinski M. (2013). *Workshop on Computational Personality Recognition (Shared Task)*. In *Proceedings of WCPR13, in conjunction with ICWSM-13*, http://mypersonality.org/wiki/doku.php?id=wcpr13

Craik, E.M. (2014, August 18). *Lewis Campbell*. The Gifford Lectures. 2014. Web. Retrieved 13 Mar. 16, from http://www.giffordlectures.org/lecturers/lewis-campbell

Descoins, Alan (2013). "Why Accuracy Alone Is a Bad Measure for Classification Tasks, and What We Can Do about It." 23 Mar. 2013. Web. 13 Mar. 16 http://blog.tryolabs.com/2013/03/25/why-accuracy-alone-bad-measure-classification-tasks-and-what-we-can-do-about-it/

Dream of the Red Chamber. (n.d.). In *Wikipedia*. Retrieved 13 Mar. 16, from https://en.wikipedia.org/wiki/Dream_of_the_Red_Chamber

Ed. Knight B., *Chapter 3 Forensic stylistics*, Forensic Science International, Volume 58, Issues 1–2, March 1993, Pages 45-55, ISSN 0379-0738, http://dx.doi.org/10.1016/0379-0738(93)90168-A

Farkhund Iqbal, Rachid Hadjidj, Benjamin C.M. Fung, Mourad Debbabi (2008). *A novel approach of mining write-prints for authorship attribution in e-mail forensics*, Digital

Investigation, Volume 5, Supplement, September 2008, Pages S42-S51, ISSN 1742-2876, *http://dx.doi.org/10.1016/j.diin.2008.05.001*.

Farnadi, G., Zoghbi, S., Moens, M. F., & De Cock, M. (January, 2013). *Recognising personality traits using facebook status updates*. InProceedings of the workshop on computational personality recognition (WCPR13) at the 7th international AAAI conference on weblogs and social media (ICWSM13). AAAI.

Fernanda López-Escobedo, Carlos-Francisco Méndez-Cruz, Gerardo Sierra, Julián Solórzano-Soto, *Analysis of Stylometric Variables in Long and Short Texts*, Procedia - Social and Behavioral Sciences, Volume 95, 25 October 2013, Pages 604-611, ISSN 1877-0428, http://dx.doi.org/10.1016/j.sbspro.2013.10.688

Fernando Pérez, Brian E. Granger, *IPython: A System for Interactive Scientific Computing*, Computing in Science and Engineering, vol. 9, no. 3, pp. 21-29, May/June 2007, doi:10.1109/MCSE.2007.53. URL: http://ipython.org

Fox, N., Ehmoda, O., & Charniak, E. (2012). Statistical Stylometrics and the Marlowe-Shakespeare Authorship Debate. Proceedings of the Georgetown University Roundtable on Language and Linguistics (GURT), 363-371.
http://cs.brown.edu/research/pubs/theses/masters/2012/ehmoda.pdf

GraphPad (2015). “*GraphPad Statistics Guide - Interpreting Results: ROC Curves*.” 1 Dec. 2015. Web. 13 Mar. 16
http://www.graphpad.com/guides/prism/6/statistics/index.htm?sensitivity_and_specificity.htm

Gray, A. R., Sallis, P. J., & MacDonell, S. G. (1997). *Software forensics: Extending authorship analysis techniques to computer programs*. Dunedin, N.Z.: Dept. of Information Science, University of Otago.

Grieve, Jack (2007). *Quantitative Authorship Attribution: An Evaluation of Techniques Lit Linguist Computing* 22 (3): 251-270 first published online July 26, 2007

Grzybek, Peter. *The Emergence of Stylometry: Prolegomena to the History of Term and Concept.* Published as a chapter in: Kroó, Katalin; Torop, Peeter (Eds.), *Text within Text - Culture within Culture*. Budapest, Tartu: L'Harmattan, 58-75. http://www.peter-grzybek.eu/science/publications/2014/grzybek_2014_stylometry.pdf

Gupta D., Sharma A., Sindhusha Y., Pachorkar Ch., “*Personality Recognition*”, 17.4.2016, https://github.com/Charudatt89/Personality_Recognition/blob/master/22-9-PersonalityRecognition/Report/Report.pdf ; GitHub:
https://github.com/Charudatt89/Personality_Recognition

Hamilton, Howard (2012). “*Confusion Matrix*.” 8 June 2012. Web. 13 Mar. 16.
http://www2.cs.uregina.ca/~dbd/cs831/notes/confusion_matrix/confusion_matrix.html

Hamilton, Howard (2012b). “*ROC Graph*.” 8 June 2012. Web. 13 Mar. 16.
http://www2.cs.uregina.ca/~dbd/cs831/notes/ROC/ROC.html

Head, Tim (2015). "*Advanced Scikit-learn for TMVA Users.*" 11 Mar. 2015. Web. 13 Mar. 16. https://betatim.github.io/posts/advanced-sklearn-for-TMVA/

Holmes, D. I. (1998). *The evolution of stylometry in humanities scholarship*. Literary and linguistic computing, 13(3), 111-117. http://llc.oxfordjournals.org/content/13/3/111.short

Holmes, D. I., & Kardos, J. (2003). *Who was the author? An introduction to stylometry. Chance*, *16*(2), 5-8. http://www.tandfonline.com/doi/abs/10.1080/09332480.2003.10554842

Hu, X., Wang, Y., & Wu, Q. (2014). *Multiple Authors Detection: A Quantitative Analysis of Dream of the Red Chamber*. Advances in Adaptive Data Analysis, Article ID: 1450012. http://arxiv.org/abs/1412.6211

Juola, Patrick (2008). *Authorship Attribution*, Foundations and Trends in Information Retrieval: Vol. 1: No. 3, pp 233-334. http://dx.doi.org/10.1561/1500000005

Koppel, M., Schler, J. and Argamon, S. (2009), Computational methods in authorship attribution. J. Am. Soc. Inf. Sci., 60: 9–26. http://onlinelibrary.wiley.com/doi/10.1002/asi.20961/full

Kosinski, M., Matz, S., Gosling, S., Popov, V. & Stillwell, D. (2015). *Facebook as a Social Science Research Tool: Opportunities, Challenges, Ethical Considerations and Practical Guidelines.* American Psychologist. Vol 70(6), Sep 2015, 543-556. http://dx.doi.org/10.1037/a0039210

Kubat, M., Holte, R. C., & Matwin, S. (1998). Machine learning for the detection of oil spills in satellite radar images. Machine learning, 30(2-3), 195-215. http://link.springer.com/article/10.1023/A:1007452223027

Lahey, B. B. (2009). Public Health Significance of Neuroticism. *The American Psychologist*, *64*(4), 241–256. http://doi.org/10.1037/a0015309

Liu, L., & Zsu, M. T. (2009). *Encyclopedia of database systems*. Springer Publishing Company, Incorporated. https://dl.acm.org/citation.cfm?id=1804422 (Based on https://stats.stackexchange.com/questions/49540/understanding-stratified-cross-validation)
Lutosławski, W. (1897). *The origin and growth of Plato's logic: with an account of Plato's style and of the chronology of his writings*. Longmans, Green and Company. https://archive.org/details/origingrowthofpl00luto

Mairesse, F., Walker, M. A., Mehl, M. R., & Moore, R. K. (2007). *Using linguistic cues for the automatic recognition of personality in conversation and text.* Journal of artificial intelligence research, 457-500. http://dx.doi.org/10.1613/jair.2349

Manning, Ch. (2016). *Coursera*. Web. 13 Feb. 2016. https://class.coursera.org/nlp/lecture/142

Manning, Ch., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval* (Vol. 1, No. 1, p. 496). Cambridge: Cambridge university press.
Marcelo Luiz Brocardo, Issa Traore, Isaac Woungang, *Authorship verification of e-mail and tweet messages applied for continuous authentication*, Journal of Computer and System Sciences,

Volume 81, Issue 8, December 2015, Pages 1429-1440, ISSN 0022-0000, http://dx.doi.org/10.1016/j.jcss.2014.12.019

Mascol, C. (1888a). *Curves of pauline and pseudo-pauline style I*. Unitarian Review, 30, 452–460

Mascol, C. (1888b). *Curves of pauline and pseudo-pauline style II*. Unitarian Review, 30, 539-546.

Matlab. "*Support Vector Machines for Binary Classification*." Matlab. Web. 10 Mar. 2016. https://www.mathworks.com/help/stats/support-vector-machines-for-binary-classification.html

Matthews, R. A., & Merriam, T. V. (1993). Neural computation in stylometry I: An application to the works of Shakespeare and Fletcher. *Literary and Linguistic Computing*, *8*(4), 203-209.

Mendenhall, T. C. (1887). The Characteristic Curves of Composition. *Science*, *9*(214), 237–249. Retrieved from http://www.jstor.org/stable/1764604

Mendenhall, T. C. (1901). *A mechanical solution to a literary problem*. Popular Science Monthly, 9, 97–110.

Meng, X., Bradley, J., Yavuz, B., Sparks, E., Venkataraman, S., Liu, D., ... & Xin, D. (2015). *MLlib: Machine Learning in Apache Spark. arXiv preprint,* http://arxiv.org/abs/1505.06807

Merriam, T. (1996). *Marlowe's hand in Edward III revisited*. Literary and Linguistic Computing, 11 (1), 19–22.

Merriam, T. (1998). *Heterogeneous authorship in early Shakespeare and the problem of Henry V*. Literary and Linguistic Computing, 13, 15–28.

Merriam, T. V., & Matthews, R. A. (1994). *Neural computation in stylometry II: An application to the works of Shakespeare and Marlowe*. Literary and Linguistic Computing, *9*(1), 1-6.

Mikros, G. K., & Perifanos, K. (2013). *Authorship Attribution in Greek Tweets Using Author's Multilevel N-Gram Profiles. In AAAI Spring Symposium: Analyzing Microtext.* http://www.aaai.org/ocs/index.php/SSS/SSS13/paper/view/5714

Mosteller, F., & Wallace, D. L. (1964). *Inference and disputed authorship: The Federalist. Reading*, MA: Addison-Wesley.

Nelson, A. H. (2004). *Stratford Si! Essex No. Tennessee Law Review*, 72 (1), 149–69.

NVIDIA (2015). *Three Reasons to Deploy Tesla K80 in Your Data Center.* http://www.nvidia.com/object/tesla-k80.html, 11 Dec. 2015. Web. 13 Mar. 16 http://images.nvidia.com/content/pdf/tesla/k80-three-reasons-flyer.pdf

Padhye, Apurva. "*CLASSIFICATION METHODS.*" Web. 11 Mar. 2016. http://www.d.umn.edu/~padhy005/Chapter5.html

Pawłowski, A., & Pacewicz, A. (2004). *Wincenty Lutosławski (1863–1954): Philosophe, helléniste ou fondateur sous-estimé de la stylométrie?* Historiographia Linguistica, 31(2-3), 423-447. http://dx.doi.org/10.1075/hl.31.2.10paw

Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, V. and Thirion, B. and Grisel, O. and Blondel, M. and Prettenhofer, P. and Weiss, R. and Dubourg, V. and Vanderplas, J. and Passos, A. and Cournapeau, D. and Brucher, M. and Perrot, M. and Duchesnay, E., *Scikit-learn: Machine Learning in Python*, Pedregosa *et al*., Journal of Machine Learning Research 12, pp. 2825-2830, 2011. http://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html

Perkins, J. (2010). *Python text processing with NLTK 2.0 cookbook*. Packt Publishing Ltd. https://dl.acm.org/citation.cfm?id=1952104

Pylkkanen, Liina (March 2003). Part of the neurolinguistics lecture (neural bases): *Function Words* [PDF document]. Retrieved from web site on 13 Mar. 16: http://www.psych.nyu.edu/pylkkanen/Neural_Bases/13_Function_Words.pdf

Sayad, Saed (2011). “Model Evaluation - Classification.” 21 Apr. 2011. Web. 13 Mar. 16. http://www.saedsayad.com/model_evaluation_c.htm

Schoenbaum, S. (1991). *Shakespeare's Lives. Oxford*, UK: Oxford University Press.

Schoonjans, Frank (2016). "*ROC Curve Analysis in MedCalc.*" 4 Feb. 2016. Web. 13 Mar. 16. https://www.medcalc.org/manual/roc-curves.php

Shams, Rushdi (2014). “*Weka Tutorial 34: Generating Stratified Folds (Data Preprocessing).*” *YouTube*. 6 Jan. 2014. Web. 13 Mar. 16. https://www.youtube.com/watch?v=tyHpHOn7R8Y

Spafford, Eugene H., Weeber, Stephen A., *Software forensics: Can we track code to its authors?*, Computers & Security, Volume 12, Issue 6, 1993, Pages 585-595, ISSN 0167-4048, http://dx.doi.org/10.1016/0167-4048(93)90055-A

Stamatatos, E. (2009). *A survey of modern authorship attribution methods*. Journal of the American Society for information Science and Technology, *60*(3), 538-556. http://onlinelibrary.wiley.com/doi/10.1002/asi.21001/full

Sun, J., Yang, Z., Liu, S., & Wang, P. (2012). *Applying stylometric analysis techniques to counter anonymity in cyberspace*. Journal of Networks, 7(2), 259-266. http://www.ojs.academypublisher.com/index.php/jnw/article/view/jnw0702259266

Tape, Thomas (2003). “*Plotting and Interpreting an ROC Curve.*”, 2 Dec. 2003. Web. 13 Mar. 16. http://gim.unmc.edu/dxtests/roc2.htm

Thornton, Chris (2012). "*Machine Learning - Lecture 5: Cross-validation.*", 23 Jan. 2012. Web. 13 Mar. 16. http://users.sussex.ac.uk/~christ/crs/ml/lec03a.html

Tu, H. C., & Hsiang, J. (2013). *A Text-Mining Approach to the Authorship Attribution Problem of Dream of the Red Chamber*. http://dh2013.unl.edu/abstracts/ab-162.html

User: Antizio (2013). "*Stylometry: Tools and Examples for Authorship Attribution.*" *DH101*. Ed. Frédéric Kaplan. 16 Oct. 2013. Web. 13 Mar. 16

VanderPlas, Jacob (2013). "*Tutorial: Machine Learning for Astronomy with Scikit-learn.*" Web. 13 Mar. 16. http://www.astroml.org/sklearn_tutorial/index.html

Vanderplas, Jake. “*Introduction to Support Vector Machines.*” *O'Reilly Media*. 6 May 2015. Web. 10 Mar. 2016. https://www.oreilly.com/learning/intro-to-svm

Vel, O. D., Anderson, A., Corney, M., & Mohay, G. (2001). *Mining e-mail content for author identification forensics*. ACM SIGMOD Record SIGMOD Rec., 30(4), 55.

Vogler, Raffael (2015). “*Illustrated Guide to ROC and AUC.*” 23 June 2015. Web. 13 Mar. 16. http://www.joyofdata.de/blog/illustrated-guide-to-roc-and-auc/

Walton, Steven (2015). "*Graphics Card Battle 2015: Nvidia Versus AMD At Every Price Point.*", 31 Oct. 2015. Web. 13 Mar. 16. http://kotaku.com/graphics-card-battle-2015-nvidia-versus-amd-at-every-p-1739764338

Wang, Yang (August 2013). *A Mathematical Study of Authorship Attribution* [PDF document]. Retrieved from web site on 13 Mar. 16: http://mate.dm.uba.ar/~hafg/cimpa-2013/cimpa-talks/Yang-Wang-slides.pdf

Wells, S. W. (1997). *Shakespeare: A Life in Drama. New York*, NY: W. W. Norton & Company.

Williams, C. B. (1975). Mendenhall's Studies of Word-Length Distribution in the Works of Shakespeare and Bacon. *Biometrika*, *62*(1), 207–212. http://doi.org/10.2307/2334505

Witten, I. H., & Frank, E. (2005). *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann. https://dl.acm.org/citation.cfm?id=1205860

Yu, Q.-X. (1998). Applications of Statistical methods to Dream of the Red Chamber, *Journal of National Cheng-Chi University*, 76: 303-327.

Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S., & Stoica, I. (2010, June). *Spark: cluster computing with working sets*. In *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing* (Vol. 10, p. 10). https://amplab.cs.berkeley.edu/publication/spark-cluster-computing-with-working-sets-paper/ || Documentation https://spark.apache.org/docs/latest/index.html

Zembowicz, Filip (2016). “*Visual Algorithms: Precision and Recall.*” Web. 13 Mar. 16. http://www.filosophy.org/post/7/visual_algorithms_precision_and_recall

Zhao, G., and Z. Chen (1975). «紅樓夢研究新編» *A New Compilation on the research of The Dream of The Red Chamber*, Taipei: Linking Publishing.

Zheng, R.; Qin, Y.; Huang, Z.; and Chen, H (2006). A *framework for authorship analysis of online messages: Writing-style features and techniques. Journal of the American Society for Information Science and Technology, 57, 3 (2006), 378–393.*

Zurini, M. (2015). *Stylometry Metrics Selection for Creating a Model for Evaluating the Writing Style of Authors According to Their Cultural Orientation*. Informatica Economica, 19*(3), 107-119. doi:10.12948/issn14531305/19.3.2015.10*