NTNU – Trondheim
Norwegian University of
Science and Technology

Detecting Contract Cheating by using Stylometry and Keystroke Dynamics

Nils Folvik Danielsen & Per Kristian Gravdal

Submission date: July 2020
Supervisor: Patrick Bours, IIK
Co-supervisor: Nancy Agarwal, IIK

NTNU – Norwegian University of Science and Technology
Department of Information Security and Communication Technology

Title: Detecting Contract Cheating by using Stylometry and Keystroke Dynamics

Students: Nils Folvik Danielsen & Per Kristian Gravdal

Problem description:

The transformation of education as a societal sector through digitalisation has provided us with new forms of schooling, such as online courses and interactive classes streamed over the web. Along with these new forms of teaching, comes new ways of evaluating students. With these new ways of evaluation, in addition to an increased number of online resources, comes new possibilities of cheating. One of these new evaluation methods is where the students are to take written exams remotely or at home over a longer period of time. With this type of exam, an individual could receive payment for taking the exam tasks for the student that is to be evaluated. There exist several websites where people are offering to write academic texts for money, but except for plagiarism detection that tests for similarities against sources on the internet, no common methods are implemented to detect cheating on exams that are submitted online. A study from 2015 shows that 5,78% of Australian students admit having engaged in contract cheating, which refers to the practice of paying a third party to complete their course work [BHB+19].

In this project, we aim to develop a solution for detecting the cheating method where a home exam is written by a different person than the intended student. The detection method will be based on verifying authorship of exam submissions, by analyzing stylometric patterns and keystroke dynamics of individual students, using statistical methods and machine learning techniques. The objective of this project is to explore if the unique way of how people type and the way individuals express themselves textually, can be successfully used for exam cheat detection. The goal is that this research could prove useful in further development of cheat detection in education of today.

Responsible professor: Patrick Bours, IIK
Supervisor: Patrick Bours, IIK
Co-supervisor: Nancy Agarwal, IIK

Abstract

As the education sector is transitioning into new, digitalized forms of teaching and conducting classes, so comes new forms of evaluating students. The evolution of technology opens up for examining students remotely, either by online home exams or longer written assessments done away from the classroom. With these new, digitalized evaluation methods, traditional measures to counter cheating on exams can not always be applied, such as exam proctoring or exam aid controls. This transition also opens up for new ways to conduct academic dishonesty, such as *contract cheating* on remote exams or assessments. Contract cheating refers to when a student gets an obligatory exam, essay, or other assessment work completed by a third party on their behalf, which will then be submitted as if they have completed the work themselves.

This project aimed to investigate the feasibility of detecting if contract cheating has taken place in an online exam. Three different approaches for contract cheating detection were developed; one approach using stylometry, another approach using keystroke dynamics, and a third approach where stylometry and keystroke dynamics were combined. Three different datasets were used in this research: one dataset containing only text data, another dataset containing keystroke data, and a third dataset that contained both text and keystroke data. The stylometry approach was applied to the two datasets containing text, while the keystroke dynamics approach was applied to the two datasets consisting of keystroke data. The fusion approach was tested on the dataset consisting of both text and keystrokes. The keystroke dynamics method showed the best results, where the system was able to detect 98.4% of the cheating cases, and wrongfully classifying only 1.7% of the non-cheating cases. The best results from the stylometry approach showed a detection rate of 95.1%, with a 5.3% wrongful accusation rate of non-cheaters. Experiments were also conducted to see how many cheaters the methods could detect without wrongfully accusing any genuine exam attempts. The best results from these experiments came from an *Aggregated Scores Fusion* that was able to detect 97.4% of the cheating cases without wrongfully classifying any non-cheating attempts.

Sammendrag

Utdanningssektoren er midt i en overgang til nye, digitaliserte måter å holde undervisning på. Med disse nye læringsformene, kommer nye måter å evaluere studentene på, for eksempel hjemmeeksamener over nettet eller større, skriftlige oppgaver. Tradisjonelle måter for å oppdage og forhindre juks, som eksamensvakter og hjelpemiddelkontroll, kan dermed ikke lenger benyttes. Når evalueringsmetodene av studentene endres, åpner dette også opp for nye måter studenter kan jukse på, eksempelvis *kontraktjuksing*. Kontraktjuksing referer til når en student får en tredjepart til å utføre arbeid på egne vegne, slik at studenten dermed blir vurdert basert på tredjepartens arbeid.

Dette prosjektet har undersøkt mulighetene for å oppdage om kontraktjuksing har funnet sted på en netteksamen. Til dette har prosjektet benyttet seg av tre forskjellige fremgangsmåter; bruk av stylometry, bruk av keystroke dynamics, og en tredje fremgangsmåte hvor en fusjon av stylometry og keystroke dynamics ble tatt i bruk. Tre ulike datasett har blitt brukt: ett datasett som kun inneholdt tekstdata, ett datasett som kun inneholdt keystroke-data, og et tredje sett som inneholdt både stylometry- og keystroke-data. Fremgangsmåten med stylometry ble benyttet på de to datasettene som inneholdt tekst og metoden med keystroke dynamics ble brukt på de to datasettene som inneholdt keystroke-data. Fremgangsmåten hvor keystroke-dynamics og stylometry ble kombinert ble benyttet på datasettet som inneholdt både tekst og keystroke-data. Systemet som tok for seg keystroke dynamics viste de beste resultatene. Her klarte systemet å oppdage 98.4% av juksetilfellene, hvor bare 1.7% av tilfellene ble feilklassifisert som juks. De beste resultatene fra stylometry-systemet viste en detekteringsrate på 95.1%, hvor 5.3% av ikke-jukserne ble feilaktig klassifisert. Det ble også gjennomført tester for å undersøke hvor mange tilfeller av juks det var mulig å oppdage uten å feilaktig beskylde noen studenter for juks. De beste resultatene fra disse testene kom fra en *Aggregated Scores Fusion* som klarte å oppdage 97.4% av juksetilfellene uten å feilaktig klassifisere noen ikke-juksere.

Preface

This project was written as the final part of a Master of Science degree within Communications Technology at the Norwegian University of Science and Technology, in the faculty of Information Technology and Electrical Engineering. The work for this project was conducted from January to late June 2020. Patrick Bours has been the supervisor and responsible professor for this project, with Nancy Agarwal as co-supervisor.

Nils Folvik Danielsen & Per Kristian Gravdal
Trondheim, Wednesday 17th June, 2020

Acknowledgments

We would like to thank our supervisors Patrick Bours and Nancy Agarwal at the faculty of Information Technology and Electrical Engineering. We are grateful for their availability for discussions, tips and other help with this thesis. We would also like to thank all our co-students that participated in the Data Collection Experiment, that turned out to be very useful in this research.

Our gratitude also goes out to our flatmates Alexander Walde, Bendik Markussen, Eivind Høydal and Jakob Stenersen Kok. Thank you for your support and for facilitating a positive work environment, considering great parts of our work were done from home. Lastly, we want to thank Morten Gabrielsen for his early wake-up calls and motivational speeches.

Contents

List of Figures

List of Tables

List of Acronyms

BoW Bag-of-Words.

DCE Data Collection Experiment.

EER Equal Error Rate.

FN False Negatives.

FNR False Negative Rate.

FP False Positives.

FPR False Positive Rate.

KD Keystroke Dynamics.

LogReg Logistic Regression.

NB Naive Bayes.

NLP Natural Language Processing.

NTNU Norwegian University of Science and Technology.

SVM Support Vector Machine.

TF-IDF Term Frequency-Inverse Document Frequency.

TN True Negatives.

TNR True Negative Rate.

TP True Positives.

TPR True Positive Rate.

Chapter 1
Introduction

1.1 Motivation

Like many other societal sectors, the education sector is profoundly altered and transformed by the digitalization era. In the last decades, new forms of teaching and conducting classes have been introduced due to the evolution of technology. These new methods within education refer to for example online courses, or interactive classes streamed through the web. These new forms have proven valuable, not only increasing the affordability of universities, but also expanding the accessibility for students [YDJP19]. Along with new ways of teaching students, comes new forms of evaluating them. This can, for instance, be exams conducted digitally with personal computers, either at a dedicated examination room or from home. Even though these new evaluation methods might benefit the education institutes in substituting older, traditional examination forms, they could also make room for new ways of cheating.

In recent years, the downsides of evaluating the participants in a course by making them take written exams in classrooms over a few hours, have been recognized. New forms of evaluation, like written reports or remote home exams conducted outside of a designated examination area, are increasingly used. Digital solutions, like the systems for home examination developed by Inspera Assessment [ins], enable online evaluation methods, such as written reports or exam tasks done over several days. These examination methods could sometimes provide a more in-depth and thorough evaluation of the examined person. However, the traditional way of preventing the individual students being evaluated from cheating on the exam using exam proctors and monitors can no longer be applied. New methods within cheat detection need to be in place. An example of a cheat prevention method already in use in universities, is plagiarism control programs, like the Urkund system [urk], used at the Norwegian University of Science and Technology (NTNU).

In mid-March 2020, the prime minister of Norway and the Norwegian Department of Education declared that all kindergartens, schools, and universities would be shut

down [udi]. This was one of the measures taken by the government to decrease the transmission of the Coronavirus, which had been spreading across the world since early January 2020. This shutdown had large consequences also for the universities as they had to either substitute classes with remote teaching or other online education methods, or simply cancel the classes. This also meant that every exam that required a physical appearance on campus would be canceled. However, every student that is signed up for a course and has met all its evaluation requirements is entitled to be evaluated in said course [lova]. This meant that the universities had to find a substitution for the traditional exams held in classrooms and examination halls. Shortly after the shutdown, NTNU informed its students that all exams would be held remotely [inn], either as a written exam or as an oral exam conducted digitally. Although the pandemic is a unique situation, it has demonstrated the possibilities of conducting traditional four hour exams as remote home exams. This is an additional example of a situation where schools, colleges, and universities are in need of tools to detect and prevent cheating on home exams.

The use of plagiarism control can be an efficient tool to prevent cheating on home exams, but it only covers one specific type of academic dishonesty. One form of cheating that is not covered by plagiarisms control is when a student gets someone else to write his or her exam for them. This could, for example, happen when an individual receives payment for doing an exam for a different student. In this case, plagiarism control would not detect anything, as the work is not taken from any online sources. The student would then be evaluated and graded on work done by someone else. This type of cheating is called *contract cheating*, and is proven to already take place at several universities and colleges. An article in NRK [BiB], describes how students at BI Business School would pay several thousand Norwegian kroner to get other students to do their home exams for them by establishing networks within social media sites. A study by Bretag et al. from 2015 [BHB+19], shows that 5,78% of Australian students admit to having engaged in contract cheating.

1.2 Scope

This project will investigate a solution for the problem of contract cheating, by exploring methods for verifying that the individual that is to be evaluated has actually written the needed work. The main task of this thesis will be to examine the possibility of detecting if a home exam is written by the student that is intended for evaluation. This problem can be coined as *authorship verification*, which is the action of verifying that a text is written by the individual claiming authorship, based on earlier works from that person. The goal of this project is to investigate if authorship verification methods using **machine learning** are viable for use in future contract cheating prevention solutions. Two approaches will be used for developing a method for verifying the author of a text. The first approach involves the use of **stylometry**,

which is based on the observation that an individual will textually write in a relatively consistent and unique manner. The second approach in author verification will be in analyzing **keystroke dynamics**, which refers to the unique typing patterns that can be examined in order to verify an individual's identity [YDJP19]. The following two-part research question was developed in adherence to our project:

- *[RQ1a] To what degree can stylometric analysis verify that the correct student has written the exam?*
- *[RQ1b] To what degree can keystroke dynamics verify that the correct student has written the exam?*

In this project, both textual data and keystroke data will be used. The fact that there are two different approaches with two different types of data opens up the possibility of comparing them to each other, as well as combining the two approaches for an optimized author verification method. From this, a second research question will be addressed:

- *[RQ2]How can stylometry and keystroke dynamics be combined to improve the author verification of the separate systems?*

By analysing the two approaches individually, as well as a combination of both approaches, we will examine which strategy gives the strongest results.

The scope of this thesis is restricted to exploring the theoretical potential of using stylometry and keystroke dynamics to detect cheating on home exams. The thesis will not focus on privacy related issues regarding the storage of text and keystrokes, or other administrative issues that is not related to the design of the detection method. The thesis is also restricted to exploring contract cheating only. This excludes other types of cheating, like fabrication of data or using illegal aids. However, the problem of contract cheating is related to the problem of plagiarism. Both are forms of cheating where students submit work they have not written themselves. The proposed method will thus also be relevant as an alternative method for detecting plagiarized work, but in this project the focus will be on contract cheating.

1.3 Outline

This thesis is is structured in the following way:

Chapter 2 (Background) presents the background for this project. The background includes information regarding the issue of remote examination and

contract cheating, followed by explanations of the concepts of stylometry, keystroke dynamics and authorship verification. In addition, earlier work related to stylometry and Keystroke Dynamics (KD), as well as technical information about machine learning methods and feature extraction used later in the thesis are included in this chapter.

Chapter 3 (Methodology) outlines the methodology that has been used when writing this thesis.

Chapter 4 (Datasets) presents the datasets that are used to perform the experiments in this project.

Chapter 5 (Treatment Design) presents the design of the cheat detection method developed for this thesis. The chapter includes the overall method, as well as specific technical details regarding stylometry and keystroke dynamics.

Chapter 6 (Treatment Validation) describes the method used to evaluate the performance of the cheat detection method, called biometric performance evaluation.

Chapter 7 (Results) presents the results of from the experiments.

Chapter 8 (Discussion) discusses the results from the previous chapter and the research questions of this project.

Chapter 9 (Conclusion and future work) includes the conclusion and the final remarks of this project, as well as suggestions for future work.

Chapter 2
Background

This chapter presents background information relevant for the problem area of our thesis. First, literature related to remote home examination and contract cheating is presented. Then the definitions of stylometry and keystroke dynamics and an explanation of authorship verification are given. The chapter proceeds by presenting earlier work on keystroke dynamics and stylometry that are relevant for this project. Finally, the technical background regarding machine learning, natural language processing, and classification methods covering our approach are presented.

2.1 Remote E-examination and Contract Cheating

Related work regarding contract cheating were partly researched in the project preceding this thesis [Dan19].

In the history of summative assessment, cheating has long been a major concern. By transitioning from examinations on pen and paper to evaluations done remotely, online, or with electronic devices, there is a concern that cheating increases as it makes cheating easier [CSND20]. In a poll conducted in 2019 by the Norwegian research company Sentio [sen], they asked 1000 Norwegian students *How many times have you cheated on exams?* Of the participants in the poll, 16% answered that they have cheated, where 12% said they have cheated once, 2% answered they have cheated twice, and 2% said they have cheated more than twice. When a similar poll was conducted nine years earlier, the number of people admitting to cheating on exams was only 5% [sen]. Arve Østgård from Sentio thinks that on top of the increase in cheaters, there are more cases of cheating than what the research says. He states that it is easier for a student to answer that they have never cheated on the poll, rather than admitting to it [sen]. One must also consider that less than 0.1% of Norwegian students get caught cheating on exams, which again indicates that most of the cheaters do not get caught [CSND20]. Compared to other countries, the penalties for cheating on exams are very strict in Norway. According to the

Norwegian laws of universities and higher education, the convicted student risks being expelled for one or two semesters, in addition to the annulment of the taken exam [lovb]. Norwegian universities might also take a reluctant stance in raising cases of cheating against their students if they are not crystal clear, because Norwegian law requires the universities themselves to cover the legal costs, no matter what the outcome of the case would be [CSND20].

One of the cheating methods that has been raising concerns in the sector for higher education in recent years is so-called *contract cheating*. Contract cheating is a form of *academic dishonesty*, where a student gets obligatory exams, essays or other assessment work completed on their behalf, which then they will submit as if they have done the work themselves [MLU+16]. This means that candidates that are up for evaluation will be graded on work done by someone else. The contract cheating provider; those who do the work for the student, ranges from freelancers, online companies, other students, and other third party contributors. According to a study from 2006, over 12% of the bid requests on the then-popular outsourcing website RentACoder were students seeking contract cheating services [CL06]. The same study also revealed that the contract cheaters posted on average 4-7 requests each and that some users had posted over 50 bid requests, including examples from multiple educational institutions. The latter indicates that these users belong to agencies that subcontract work [CL06].

Reports from the media that exposes this type of cheating suggests that it is increasing and that the hire of third parties for cheating purposes is going undetected by the universities, due to perceptions that is it very hard to identify, and that there are no effective detection and prevention methods in place [HBR20]. However, credible evidence that there is an increase of contract cheating is lacking. Still, the damage that a contract cheating scandal could do to the reputation of a university has prompted the call for solutions for mitigation or prevention. Some researchers propose the increased use of traditional exams where the students are guarded by invigilators, due to the belief that this examination form is the most secure assessment method [mor][Lin16]. Other researchers suggest forming new assessment designs that can combat contract cheating [MLU+16], although there is not much evidence to back up the relationship with assessment design and contract cheating [HBR20]. However, the increase of third parties like essay-writing companies and exam stand-ins is evidence enough that contract cheating is an evolving issue in education [NL16].

As contract cheating has been a challenge within academic evaluation for some time, certain measures have been proposed to combat this issue. According to the *Institutional Toolkit to Combat Contract Cheating* by the International Center for Academic Integrity, the first step in challenging this issue is to sharpen the focus on academic integrity at universities in order to counter a contract cheating

culture [MLU+16]. This entails educating students in making ethically right choices in completing academic work and raising awareness of the importance of the institution's values, as well as their own values. It also involves educating and informing faculty on how to cultivate ethical practice in academics, how to communicate to their students about academic integrity, and how to create a culture of integrity in the class. Beyond the sharpened focus on ethical practice and academic integrity, the toolkit also proposes specific approaches that can be used for prevention of contract cheating.

The toolkit proposes that assessment design that ensures good pedagogy in conjunction with reducing contract cheating should be prioritized. Such designs could entail assessments with integrity, where, for example, a student is required to complete multiple drafts of an assessment. The toolkit states that students that are required to submit multiple iterations of an assignment are less likely willing to pay a third party to do their work. Another assessment design that could reduce contract cheating is providing more personalized and authentic assignments that are more specific to the course or class, and avoid using assignments that are provided with textbook publisher's content [MLU+16]. Measures like limitations on non-substantive requirements (requirements on page counts, word counts, etc.) are also suggested to be reevaluated. The toolkit argues that limitations can often compel students to plagiarize or contract cheat [MLU+16]. It also states that allowing late submissions could also help prevent and mitigate contract cheating. Cheating is not always a result of bad character but often comes from desperation or the feeling that the student is unable to do what is academically required from them. A preoccupied student with strict deadlines on work submissions will often resort to cheating [MLU+16]. The toolkit suggests that the setting of later submission dates could help provide students with the breathing room in such situations.

Although the toolkit proposed by the International Center for Academic Integrity proposes several measures to reduce contract cheating, it does not focus much on actually detecting contract cheating cases. As traditional written exams are transitioning into becoming *e-exams,* examinations done online or remotely, new anti-cheating measures that catch cheaters must also be considered. Several methods to mitigate and counter contract cheating on e-exams have been researched. Bawarith et al. [BBFGD17] investigates several methods used for cheat detection in online exams through continuous authentication and online proctors. They investigated methods like eye tribe-tracking to continuously guarantee the identification of the examinee, and the use of fingerprints for authentication of students during the exams, as well as using an E-proctor to monitor the student during the examination. The method classified the students status as non-cheating or cheating with the help of two parameters; the total time the examinee was off-screen, and the number of times the examinee went off-screen. Cluskey Jr et al. [CJER11] investigated the control issues

related to online exams and asserted that the cost of enabling e-proctors exceed potential benefits. The researchers proposed non-proctor alternatives to promote academic honesty instead.

2.2 Stylometry and Keystroke Dynamics

This project will use two approaches to create a contract cheating detection system: stylometry and keystroke dynamics. This section outlines the two different approaches and explains the principles related to authorship verification that is relevant for this project.

2.2.1 Behavioral biometrics

Stylometry and keystroke dynamics goes under a common category in studying metrics related to human characteristics, known as *biometrics.* A system for detecting exam cheating using stylometry and keystroke dynamics is thus a special case of a *biometric system.* A biometric system is defined as a *" system that allows the recognition of a certain characteristic of an individual using mathematical algorithms and biometric data."* [GMA19]. More specifically, stylometry and keystroke dynamics are categorized as *behavioral biometrics*, which refers to every human behavior that can be used in either authenticating or identifying an individual's identity.

Behavioral biometrics differs from other biometrics where physiological (or biological) attributes, such as fingerprints, are used. In authentication or identification of a person looking at behavior, behavioral patterns that consist of several *semi-behaviors* reflecting unique habits are observed in activities that an individual undertakes. Other examples of these activity patterns are *vocal behavior* or speech, unique mouse dynamics, movement, and signature dynamics [Wan09]. One can say that the biometric factor in behavioral biometrics is *something you do*, and that behavioral biometrics focuses on *how* an individual conducts an activity, rather than what the outcome of an activity is.

2.2.2 Stylometry

Stylometry refers to the quantitative study of literary- and writing styles and is based on the observation that an individual writer tends to write in a relatively unique and consistent way. While style-based text recognition can be used to distinguish between different topics and genres (quantification of topic), stylometry is based on analyzing the personal writing style of authors (quantification of style) [PS19]. For example, each writer has a unique vocabulary, sometimes broad, other times limited. Another example is that authors have their own way of structuring sentences; some write them short, while others tend to write long blocks of text. Also, no writer

uses punctuation like colons, semicolons, and periods in the exact same way as other authors. The unique and recognizable manner that every individual writer structure their texts and formulate sentences, opens up for many applications regrading textual data analysis. Stylometry is a well-researched field, but there is no consensus in the literature on what characteristics are most efficient to quantify style. Different characteristics of a text are generally extracted from a document and called *features* when applied in a stylometric method. Some of the most used stylometry features are lexical and syntatic, which refers to the characters and words, and sentence structure, respectively. As a behavioral biometric, stylometry is considered less effective than e.g., keystrokes and mouse movement, as stylometry operates at a higher cognitive level [BMCT12].

2.2.3 Keystroke Dynamics

Keystroke dynamics (KD) is a behavioral biometric that refers to the automatic method of identifying or verifying a person based on the rhythm, speed, and manner the individual is typing on a keyboard. This approach for verification and identification is based on the observation that people have a unique way of typing. By recording and analyzing sets of measurements collected from the way an individual types, a profile of an individual, often referred to as a *keyprint*, can be established. This profile represents the typing behavior pattern of the person. One of the two prime features of the measurements taken in order to construct these individual profiles is *duration*, which refers to the amount of time a key on a keyboard is held down. The other feature is *latency*, which is the time between the previous key is released and the next key is pressed down. Keystroke dynamics is considered a reliable behavioral biometric, as it operates at an automatic motor control level [BMCT12], i.e., processes outside of our consciousness. One of the benefits of using keystroke dynamics for identification or verification, is its ability to capture unique, low-level human processes involved in typing behavior. The ballistic and semi-autonomous nature of these behaviors makes typing patterns very hard to duplicate. Keystroke dynamics is also advantageous as a behavioral biometric because it can be recorded and collected without the knowledge of the user, and because of its inexpensiveness. [BW12].

2.2.4 Authorship verification

In this subsection, the words "document" and "sample" are used interchangeably to describe both the text that a document is made up of and the keystroke data sample corresponding to that text.

Authorship analysis using stylometry and keystroke dynamics can be carried out from three different perspectives, namely *authorship attribution, authorship*

verification, and authorship profiling [BTSW13] [Pla18]. Authorship attribution consists of determining who the correct author of a given document is, given a list of possible authors. Authorship verification refers to verifying whether a given document is written by a specific author or not. Authorship profiling consists of determining different characteristics of the author, like age, gender etc. The problem of detecting contract cheating using stylometry and keystroke dynamics can be viewed as a particular case of authorship verification.

The problem of authorship verification is closely linked with the problem of authorship attribution. The difference between them, as defined by PAN [SDV+15], are:

Definition 2.1. Authorship Verification Given a set of documents by a single author and a questioned document, determine if the questioned document was written by that particular author or not

Definition 2.2. Authorship Attribution Given a sample of reference documents from a restricted and finite set of candidate authors, determine the most likely author of a previously unseen document of unknown authorship.

While the majority of research related to keystroke dynamics have been on authorship verification, this is not the case for stylometry. Traditionally, most research in the field of stylometry has been on author attribution in a closed-world setting, which is the procedure defined in definition 2.2. Closed-world attribution means that all the authors that are to be identified are included in the training set. The closed-world setting is the main characteristic that distinguishes authorship attribution from authorship verification. Despite the two problems being similar, verification is significantly more difficult than closed-world attribution due to the open-world setting [KS04]. For example, if we wanted to attribute an unknown document to either author A or author B, it would be enough to create a model based on author A and B's known texts (stylometry) or keystrokes (keystroke dynamics) and test the unknown document against that model.

On the other hand, if we want to determine if an unknown document is written by author A or not, there is no way to find a perfect representative sample of "not author A" texts or keystrokes. The problem of contract cheating detection corresponds to the latter example, as the real author of an exam will not be known in the case of contract cheating. Author verification can thus be approached as a one-class problem, a classification problem that tries to identify an outlier from a target class, in this case, the set of known textual or keystroke features from a specific author. It is, however, common to make use of negative samples to create a binary classification problem in an efficient way, as demonstrated by Koppel & Winter [KW14] (Stylometry) and Antal & Szabó [Pla18] (Keystroke Dynamics).

The two approaches of authorship verification represent an important distinction on how to implement a method for verifying authorship, namely as a one class-problem or a two-class(binary) problem. The two approaches to authorship verification are often called the *intrinsic* and *extrinsic* model. The intrinsic verification model approaches the task as a one-class classification problem, and utilize only the given text or keystrokes from an author and the unknown document to determine if the unknown document is written by the author or not. This approach does not compare the unknown document to any external features but only attempts to determine if the unknown document belongs in the set of known documents or not. The extrinsic model, on the other hand, uses text or keystrokes from external authors to create a binary classification problem [SDV+15]. The extrinsic model attempts to use external text or keystrokes to represent a general writing style or keystroke pattern. The difference between the two models is illustrated in Figure 2.1 [PS19].

Figure 2.1: The intrinsic model using only the known documents from an author (left) vs. the extrinsic model using external samples to create a binary classification problem (right)

Another important distinction to be aware of in authorship verification is the difference between the *instance-based* and the *profile-based* method. The difference between the two methods is based on whether they treat each training sample individually or cumulatively (per author) [Sta09]. In the instance-based method, each training sample is individually represented as a separate instance of authorial style. Each document from the known author contributes separately to the training model and is considered an instance of the problem in question. The profile-based method, on the other hand, concatenates all the known documents from an author into one single sample. For stylometry, this means concatenating all texts into one large text file, while for KD, this means combining keystrokes corresponding to different documents to represent one single instance of the authors typing style. A

disadvantage of the profile-based method is that it disregards any differences in style between the individual samples from the same author. The advantage of the profile-based method is that long passages of text and keystrokes from a person can give a more accurate representation of an individual's writing style. The instance-based method is the most common of the two, *"but the profile-based method is generally more robust when few texts (in quantity or length) of known authorship are available"* [SDV+15]. The distinction between instance-based and profile-based is mainly drawn for stylometry, as KD ordinarily are instance-based. However, the profile-based method can be useful also for KD when few keystrokes are available per sample.

2.3 Related Work: Stylometry and Keystoke Dynamics

Stylometry and keystroke dynamics are well researched fields with several areas of application. Although little research has been done directly related to contract cheating, the techniques and technologies used in other areas can be applicable in this project. This section presents the state of the art within stylometry and keystroke dynamics.

2.3.1 Stylometry

Koppel and Winter [KW14] performed a study on authorship verification using stylometry by using a set of impostor documents to create a classification problem, called The Impostors method. Koppel and Winter compare it to a police lineup, where the goal is to determine if the questioned document is sufficiently more similar to the questioned authors' documents than the impostors. The corpus[1] used in the study is blog posts obtained from blogger.com. The Impostor method compares the similarity of the unknown document and the known document to the similarity between the external documents (impostors) and the known and unknown documents. If the similarity of (known document, unknown document)^2 is higher than the similarity of (known document, external document) $\times$ (unknown document, external document), the unknown document is classified as positive (same-author). To evaluate the Impostor method's results, they developed two simple baseline methods: a similarity-based baseline method that uses the cosine and min-max similarity measures and a supervised baseline method using Support Vector Machine (SVM). The similarity-based baseline method simply measures the similarity of two documents and label them same-author or different-author based on a threshold. In the supervised baseline method, the SVM is trained on 1,000 pairs of documents labeled either same-author or different-author. The study used various feature sets, including word and character n-grams, function words, and others, and documents of 500 words in length are used. The results show that the Impostor method outperforms the baseline methods in

[1] all the writings or works of a particular kind or on a particular subject

recall, precision, and accuracy. The most efficient version of the Impostor method uses impostor texts from other bloggers in the same genre as the given document pair. The Impostor method obtained an accuracy of 87,4%, while the best baseline method(supervised SVM) obtain an accuracy of 80%. Koppel and Winter discuss the importance of choosing a balanced set of impostors, in regards to similarity and number of impostors. The best results are obtained when the impostors are selected from the same genre as the input documents.

Feng et al. [FBC12] used four different datasets to investigate how stylometry could be used for deception detection. One dataset contained both truthful and deceptive essays, and the three other datasets contained truthful and deceptive reviews from Tripadvisor and Yelp. The paper uses lexical and syntactic features and SVM for classifying the essays and reviews as either truthful or deceptive, by using 80% of the data as training data and 20% as test data. The experiments using the essays obtained an 85% accuracy, while the best results from the reviews achieved a 91.2% accuracy.

Schroeder, Kũppers, and Opgen-Rhein [ORKS18] performed a study that uses stylometry to detect cheating on programming exams. The study used Deep Neural Networks (DNNs), RandomForests (RFs), and Support Vector Machines (SVMs) for author verification on a dataset containing 12 assignments from 13 different students taking a basic first-semester course in Java. An extrinsic method is used, creating a negative class from all reference material that doesn't stem from the supposed author. The study extracted features based on the layout of the code, such as number of spaces used, and syntactic features that describe the code's inner logic. The study conducted two different experiments: one where all assignments were used for training, and the accuracy was computed using 5-fold cross-validation, and one where all assignments were used for training except one, that was used for testing. In the experiment where one single assignment was used as a test set, which is the test that bears most resemblance to an actual exam cheat detection scenario, RandomForests performed best with an accuracy of 71.43%. Deep Neural Networks performed worst, with an accuracy of 23.81%, which the authors argue is caused by insufficient training data. The study concludes that it is plausible to use this method in a real exam scenario, but that more work can be done to improve the accuracy.

Howedi and Mohd [HM14] addresses the problem of authorship attribution using stylometry when faced with limited training data. The paper states that traditionally, author attribution using stylometry has focused on long texts, and 10,000 words per author have been considered to be a reliable minimum. In this study, short texts between 290 and 800 words per text are used. The data is collected from 10 different Arabic authors writing about their travels, and three different texts is collected from each author. The primary classifier used in the study is Naive

Bayes, as this tends to be more accurate on a small amount of training data. SVM is used for comparison, as SVM traditionally provides good accuracy when given a large dataset. Two separate experiments were conducted, one using character n-grams and one using word n-grams. Chi-Squared and Information Gain were used as feature selection methods, and three-fold cross-validation was used instead of separate training and testing set due to the small amount of data. The results from the experiments show that the best accuracy was achieved from using Naive Bayes on word unigrams, with an accuracy of 96,67%. The next best accuracy was achieved from using character tetragrams, with an accuracy of 93,33% on both Naive Bayes and SVM. The average accuracy from character and word n-grams combined, with n ranging from 1-4, was 71,85% using NB and 62,95% using SVM. Character n-grams performed better than word n-grams on average. The experiments also showed that the inclusion of punctuation improves the accuracy.

Plechac [Ple19] performed a study on the play Henry VIII to determine who was the authors of specific pieces of texts. The play Henry VIII is recognized as a collaborative work between Shakespeare and Fletcher. However, there are different opinions on which parts are written by whom, and if other authors were involved as well. Based on the opinions of James Spedding and other experts, the study considered Shakespeare, Fletcher, and Massinger as candidates for the play's authorship. Plechac performed an experiment using SVM as classifier, and the 500 most frequent rhythmic types and 500 most frequent words as features. Individual plays from the authors were used as training data. The results correspond to a great extent to the attribution from James Spedding. Except for two occurrences, all the scenes are attributed to the same author as Spedding proposed. Plechac concludes that Henry VIII is highly likely a collaboration between Shakespeare and Fletcher, while the participation of Massinger is unlikely.

Stamatos et al. [JS13] [SDV$^+$14] [SDV$^+$15] gives an overview of the author identification competition PAN 2013, 2014 and 2015 that focuses on author verification using stylometry. The rules for each year's competitions were the same: given a set of documents by a single author and a questioned document, determine if the questioned document was written by that particular author or not. The corpus included English, Spanish and Greek documents in 2013, supplemented with Dutch for 2014 and 2015. The winner of the 2013 competition, Seidman [Sei13], used a modified version of the impostor method. Seidman used unigrams, unigrams-Term Frequency-Inverse Document Frequency (TF-IDF) and character 4-grams as features, and tested several different distance/similarity measures. The winner of the 2014 competition, Khonji & Iraqi [KI14], also used a modified version of the Impostors method, using documents by other authors to create a binary classification task. Khonji & Iraqi used a diverse set of features, namely: letter-level, word-level word shape-level, and part-of-speech tag-level. The main modification to the original

impostor method was in regards to the scoring measure. Instead of measuring *whether* two input vectors are similar, Khonji & Iraqi measured *how* similar they were. The winner of the 2015 competition, Bagnall [Bag15], used a recurrent neural network (RNN) adapted to perform well on a smaller corpus than RNNs typically need. Bagnall performed a rather complex text preprocessing on a character level, before training the RNN. As the winners of PAN 2013 and 2014, Bagnall also used an extrinsic verification model. It is worth mentioning that several of the high ranking submissions in the 2015 competition applied variations of the Impostor method. The key takeaways from the three author verification competitions from PAN, are that extrinsic verification models seem to perform better than intrinsic models. Two out of three winners used a variation of the impostor method, highlighting the method's effectiveness. The most common features used are simple character and word n-grams (including unigrams), punctuation marks and stop words, while for the more advanced features, part-of-speech (POS) is the most popular. Almost all submissions attempted to combine different types of features, while a few approaches focused on only one type.

2.3.2 Keystroke Dynamics

Young et al. [YDJP19] explored the potential of using keystroke dynamics data to create *keyprints*, which can be described as typing fingerprints, for authentication of individuals in online courses. They explored the best practices of implementing keyprint signatures in contexts other than simple password verification. Their study invited university students taking a Management Information Systems course to provide data. The researchers managed to gather keystroke data of 84 students by tracking duration- and latency times for specific keys and key combinations while the subjects completed four different typing tasks. Attempting to correctly identify individuals through keystroke dynamics can be difficult, however the results of this study suggest that keyprints can reliably indicate negative cases, in other words where a typing sample was not the intended student.

Tappert et al. [TVC10] developed a system for experimenting on free-text input keystrokes used for authentication of users. The system developed consists of a Java application used to capture raw keystroke data over the Internet, a component used for feature extraction, and pattern classifiers used to make decisions for identification and authentication. This system was used on experiments with 100 different subjects giving inputs on two different modes - copy typing and free-text input, as well as taking input from both desktop and laptop keyboards. Duration and latency features were used as input in a Nearest Neighbor classifier to identify users by comparing the feature vector of the test sample in question against those of the samples in the training set. The best results under optimal conditions was 97.4% accuracy when using the full dataset. The system could accurately identify or authenticate subjects

if the individuals used the same type of keyboard, and if sufficient enrollment samples were provided. The input texts that were evaluated contained up to 650 keystrokes, but additional experiments with the system showed that input of 300 keystrokes can also obtain a reasonable accuracy.

Banerjee et al. [BFKC14] explored how keystrokes as a means to access the writing process of online authors can distinguish between truthful and deceptive writing. Their research showed that the varied keystroke patterns, like editing maneuvers, backspace trends, and pause durations, could help in telling if the writing is truthful or not. In this study, the empirical results showed how analyzing and incorporating keystroke-based features can lead to improved deception detection in online reviews and essays. All the experiments conducted in this research were performed with SVM-classifiers, using 5-fold cross validation with a 80/20 division for training and testing. The best results from the research were an accuracy of $\approx$ 84%.

Monaco et al. [MPT+15] presented their results of the One-Handed Keystroke Biometric Identification Competition (OhKBIC) held at the 8th IAPR Internation Conference of Biometrics. A dataset of keystrokes that included freely typed long-text samples from 64 subjects was collected. The participants designed classification models trained on normally-typed samples in an attempt to correctly classify an unlabeled dataset consisting of both normally-typed and onehanded-typed samples. Duration and latency features were used in the experiments as input in two different classifiers; one that computed the normalized distance between feature vectors, and one using SVM. A combined version of the two classifiers showed the best performance. The best results for the analysis of the normally-typed samples, which is most relevant for this project, showed an 83% prediction accuracy. This result came from a system using duration - and latency keypress features, and Random Forest for classification.

Deng et al. [DZ13]introduced two new algorithms to the domain of authenticating users through keystroke dynamics. Their experiments were conducted on a benchmark dataset called the CMU Keystroke Dynamics Benchmark set, where algorithms such as Neural Networks, SVM, Manhattan, and Gaussian Mixture model had already been tested and compared from earlier research [KM09]. The two new algorithms introduced were the *Gaussian Mixture model with the Universal Background Model (GMM-UBM*), and the *Deep Belief Nets (DBN)*. These two new algorithms, unlike most existing approaches in user authentication, did not only use genuine user's data at training time. GMM-UBM and DBN also leveraged data from background users to increase the model's discriminative capability without seeing the impostor data at training time [DZ13]. Their research showed that these two new additions showed at best a 3.5% Equal Error Rate (Equal Error Rate (EER)), which was a 58% reduction in EER from the best earlier published approaches for the CMU Keystroke dynamics

benchmark set. An EER of 3.5% corresponds to an accuracy of 96.5%.

2.3.3 Combined Stylometry and Keystroke Dynamics

Monaco et al. [MSCT13] investigated the possibility of developing a system for authenticating students taking online exams. This authentication was based on analyzing both KD and stylometry as behavioral biometrics. The data used in the research were collected from 30 university students that were invited to answer 40 test questions. Two separate systems for authentication were developed, one for stylometry and one for KD. Both systems consisted of a data collector, a feature extractor, and a pattern classifier. The features extracted for the KD system were mainly based on different *duration* times on key presses and *latency* times on different key transitions an individual performed while answering the test questions. The stylometry system employed a set of linguistic features. These were either word-based-, character-based- or syntax-based features the individual used in the text they wrote for the exam. The pattern classifier in the systems is based on a vector-difference model, where a multi-class problem is transformed into a two-class problem. The two classes the researchers wanted to determine were *within-person* ("authentication successful") and *between-person* ("authentication unsuccessful"). Two separate experiments were conducted on both the developed authentication systems; in the first experiment, 8 samples per student where used. In the second experiment, only 4 samples per student were used, but the samples were twice as long as in the first experiment. The performance of their stylometry system was 74% and 78% on the first and second experiment respectively, while the KD system was considerably stronger with a performance of 99,96% and 100%. The data used in [MSCT13], which consists of both textual- and KD data, is also used extensively in this project. This dataset is coined the *Stewart* dataset, and is described in Chapter 4.

Li et al. [LBBB19] investigated how chat data could be used to predict the actual gender of an online subject by examining features in both keystroke dynamics and stylometry. The researchers acquired the data by capturing the keystrokes and textual data where the participants were chatting remotely via Skype for only 15 minutes. Their research proposed a method for gender prediction using a Random Forest approach by analyzing features such as length of each message and the average number of letters in a word for stylometry, and duration and latency for KD. The participants were asked to chat freely on a topic of choice in order to be better acquainted. The experimental results with the Random Forest approach achieved a 84% prediction accuracy for stylometry and a 76% prediction accuracy for KD. These results were obtained from using the training data which consisted only of messages constructed from more than 40 keys. Stylometry and KD was combined with a score-level fusion and a majority voting mechanism to predict the user's gender, and achieved 72% prediction accuracy when applied to the complete dataset

of participants chatting freely.

Fridman et al. [KGJ+13] investigated a fusion of keystroke dynamics, mouse movement, stylometry, and web browsing for authenticating individuals from an office environment. For both stylometry and keystroke dynamics, SVM was used for classification. The features used for keystroke dynamics were duration and latency features, while the feature set for stylometry included function words, grammar and n-grams. The decision rules for the fusion is based on Bayesian cost, and uses the probabilities of error for the individual detection systems to make the decisions. The results show that keystroke dynamics and mouse movement are the most efficient for authenticating users. However, using stylometry in addition to these approaches improves the results marginally.

2.4 Technical Background

This project will use machine learning algorithms to create a method for verifying students and detect contract cheating. This section will explain the technical background for the machine learning algorithms used later in this project. Also, the Natural Language Processing (NLP) techniques used in stylometry to transform the text into feature vectors are described.

The relevant technical background were researched in the project preceding this thesis [Dan19]. This is amended with more material that has been studied after the project.

2.4.1 Machine Learning Classifiers

Verifying students taking a home exam can be seen as a classification problem, as a student delivering an exam can be classified as either a cheater or a non-cheater. A machine learning classifier is an algorithm that maps input data to specific categories by combining fundamental principles in computer science with statistics, probability, and optimization [MRT12]. The input data used in this project are text and keystroke data. No encoding process is needed for the keystroke data, as the raw keystroke information is already in number form. However, in order for text to be used as input data in machine learning, features need to be extracted from the text and encoded as numbers. This will be further explained in the next subsection. In this subsection, the three machine learning classifiers used in this project are described.

Support Vector Machine

Support Vector Machine is a classifier formally defined by a separating hyperplane [Alp20] [svm]. A hyperplane is a subspace of a V-dimensional vector space, where the subspace is V-1 dimensional [Bis06]. When having a two-dimensional plane, the hyperplane is a one-dimensional line, shown in Figure 2.2.

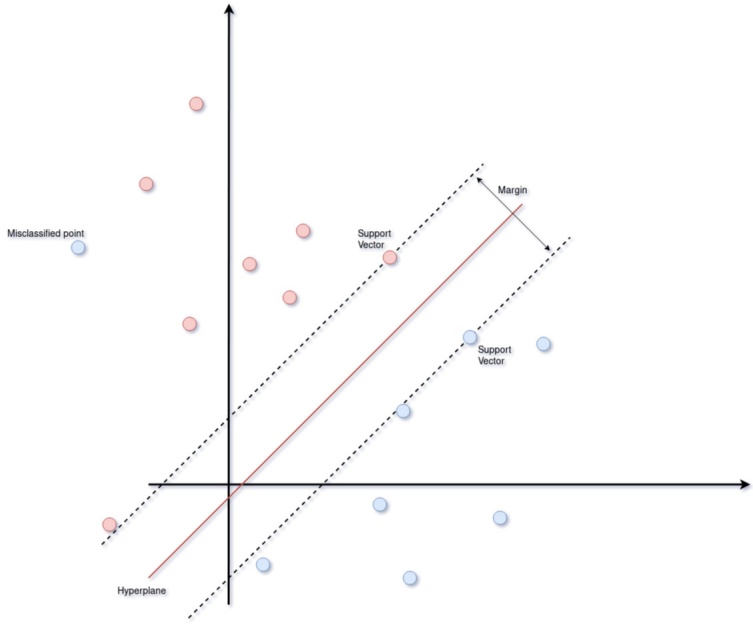

Figure 2.2: Linear SVM

SVM can also be applied for non linearly data using kernel transformation, which refers to that non linearly separable data in a dimensional space can be linearly separable in a high dimensional space. With a linear kernel, the learning of the hyperplane is performed by transforming the problem using linear algebra. For the prediction of a new input using the dot product of the input (x) and each support vector (sv_i) is calculated as

$$f(x) = B_0 + \sum a_i * (x, sv_i) \tag{2.1}$$

This involves calculating the inner products of a new input vector (x) with all the support vectors of the training data. Both B_0 and a_i are coefficients estimated from the training data [Alp20][svm].

SVM contains a regularization parameter, coined as C in the Python scikit-learn library, which decides how much each training input can be misclassified. If the regularization parameter is large, the hyperplane margin will be smaller if the hyperplane is effective in classifying the input correctly. The margin is a separation of the line to the closest class points. If C is small, some misclassifications are ignored,

and a larger margin is trying to be achieved for the hyperplane. A gamma parameter in SVM defines the influence reach of a single training input, with low value meaning that points further away from the separation line are considered and high values meaning only the closest points are considered [Alp20][svm].

A good margin is necessary for SVM classifiers to be optimal. The margin is the separation of a line to the closest class points, illustrated in Figure 2.2. A margin is good when there is an equal distance between the separation line and the closest class point of each class. SVM is proven to be memory efficient and is considered by many as the best classifier for two-class classification tasks.

Naive Bayes

NB refers to different variants of supervised learning algorithms based on Bayes' theorem, which is stated mathematically as

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{2.2}$$

where A and B are events and $P(B) \neq 0$. P(A|B) is the likelihood of A occurring given that B is true, and P(B|A) is the likelihood of B occurring, given that A is true. P(A) and P(B) are the probabilities of observing A and B separately and independently of each other. This theorem describes the probability of an event by using prior knowledge of conditions that could be related to the event [MMS99]. The "naive" expression refers to the assumption of conditional independence between every pair of features given the value of the class variable [MMS99]. These classifiers have shown good performance in real-world situations, like document classification. NB can be very fast compared to other methods. Because it requires few necessary parameters, NB can often perform well with a small amount of training data.

Logistic regression

LogReg is a supervised machine learning classifier that uses linear regression and the Sigmoid function to determine the probability of observations belonging to a discrete set of classes. While standard linear regression can give predictions that exceed the range of [0,1], logistic regression uses the Sigmoid function to map all values into the range of [0,1] [Mis19].

Linear regression:

$$y = \beta_0 + \beta_1 X \tag{2.3}$$

Sigmoid function:

$$f(x) = \frac{1}{1 + e^{-(x)}} \tag{2.4}$$

Logistic regression:

$$p = \frac{1}{1 + e^{-(y)}} \tag{2.5}$$

In a binary classification problem, the probability score returned from logistic regression is the predicted probability of an observation belonging to class 0 or 1. A value of 0.90 represents that an observation belongs to class 1 with 90% probability. Logistic regression can be illustrated by plotting the sigmoid function, as in figure 2.3. All values above the threshold of 0.5 are predicted to belong in class 1, while the values below are predicted to belong in class 0.

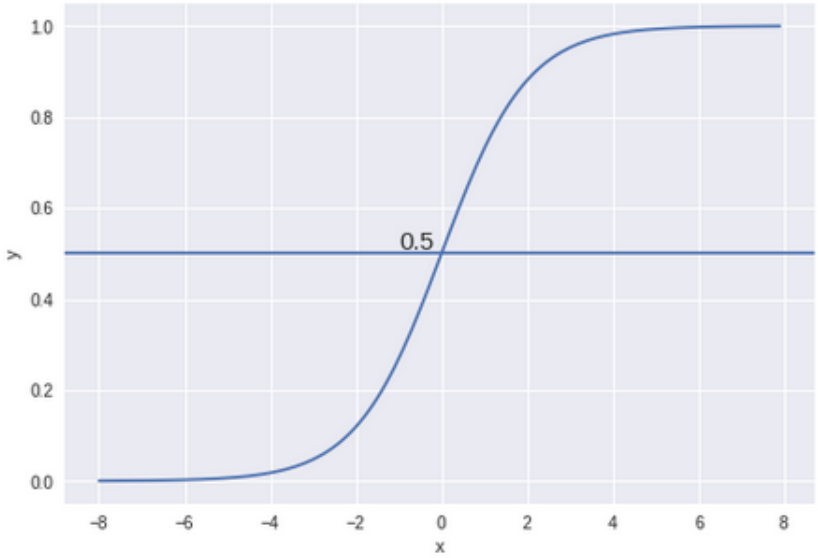

Figure 2.3: Logistic regression [Pan]

Some of the advantages of using LogReg are that it does not require too many computational resources, it is easy to interpret and implement, it does not require input features to be scaled, and it does not require hyper-parameters for tuning [MCH18]. Disadvantages of LogReg are that it is vulnerable to over-fitting, and that it is sensitive to correlated predictive values and outliers. [MCH18].

2.4.2 Natural Language Processing Techniques

NLP is a broad term that encompasses many different techniques that allow computers to understand human speech and text. The ultimate objective of NLP is to read, decipher, understand, and make sense of the human languages in a manner that is valuable [Bha19]. In order for a text to be used as input in the machine learning classifiers used in this project, NLP techniques needs to be applied. By extracting features from a text and encoding them as numbers, machine learning classifiers are able to interpret the data for training and classification.

This section will present the technical background of the NLP techniques that are used in this project.

Bag-of-Words and n-grams

Bag-of-Words (BoW) is a model that counts the occurrences of every word in a given text, which again is used to create a dictionary of the words and their count, measuring their presence of known words in a text. BoW is essentially a special occurrence of counting *word n-grams*, with n = 1, and is called *unigrams.* Bigrams and trigrams are the names of word n-grams with n = 2 and n = 3, respectively. Unigrams does not register in what order or structure the words in a text occur, as they are independent of adjacent words. When n > 1, the model is extended to store multi-word expressions of adjacent words, like, for example, "[contract, cheat, detection]" for trigrams. The method of counting n-grams is used to extract features from a text that can be applied in modeling. Its main idea is that texts with similar content are similar texts and that we can learn something about the meaning of the text based on its content [MMS99]. When applied in stylometry, n-grams are proven to be effective in quantifying personal writing style. Standard n-gram counting does not take the length of the document into consideration.

N-grams can also be used on a character level, called *character n-grams.* While BoW and word n-grams are the most common use of n-grams, character n-grams have proven to be an effective technique that, in some cases, outperform word n-grams [LRGL13]. Conceptually, word and character n-grams work the same way, counting the occurrences of n words or characters adjacent to each other in a text. Examples of different n-grams are presented in Table 2.1.

Term Frequency-Inverse Document Frequency

Word 1-gram	[Hello] [world!]
Word 2-gram	[Hello, world!]
Character 1-gram	[H] [e] [l] [l] [o] [-] [w] [o] [r] [l] [d] [!]
Character 2-gram	[H,e] [e,l] [l,l] [l,o] [o ,-] [- ,w] [w ,o] [o ,r] [r ,l] [l ,d] [d ,!]

Table 2.1: N-grams for sentence "Hello world!"

TF-IDF is an extension of the n-gram model, which in addition to counting the occurrences of all n-grams, also focuses on the total frequencies of n-grams in a *text corpus* and helps penalize too frequent n-grams across all documents. TF-IDF can also remove n-grams that occur less than a specified count from the feature space [MMS99]. An apparent weakness with exclusively counting n-grams in a text is the bias for longer documents, and lack of scoring when highly frequent n-grams occur. To avoid a bias for longer text, it is possible to normalize the counts of each n-gram, by dividing each count on the total number of n-grams in the text. This is called the **Term Frequency**. In addition to the Term Frequency, **Inverse Document Frequency** can be used to decide the rarity of the n-gram, and scoring it thusly [mlma]. IDF decreases the weight of frequent n-grams and increases the weight of rare occurrences. The scikit-learn library used in Python contains a model of TF-IDF that will prove useful in this project [mlmb].

Term frequency is given by

$$tf_{id} = \frac{n_{i,j}}{\sum kn_{i,j}} \tag{2.6}$$

where the numerator is number of occurrences for term t_i in document d_j and the denominator is the sum of all terms in document d_j. IDF is given by

$$idf_i = Log\frac{N}{|\{j : t_i \in d_j\}|} \tag{2.7}$$

where the numerator is the total number of documents in the corpus, and the denominator is the number of documents containing term t_i. TF-IDF is computed as the product of TF and IDF:

$$tfidf_{t,d} = tf_{i,d} * idf_i \tag{2.8}$$

2.4.3 Technical Tools

In this project, a set of technical tools, software packages, and frameworks are used in order to create a method for authorship verification. The implementation has been realized with the programming language Python, using the interactive development environment *Jupyter Notebook*[2]. Jupyter is used due its ability to perform data visualization and being effective when working with larger datasets. The *pandas*[3] library has been used extensively for effective manipulation and representation of the data in this thesis. For handling the classification tasks in both stylometry and keystroke dynamics, the machine learning library *scikit-learn*[4] has been used. Scikit-learn, as well as the as well as the *Natural Language Toolkit*[5] library has been used for feature extraction methods in the stylometry implementation.

[2]https://jupyter.org/
[3]https://pandas.pydata.org/
[4]https://scikit-learn.org/stable/
[5]https://www.nltk.org/

Chapter 3
Methodology

3.1 Design Science

The chosen methodology for this project is Design Science, based on the book "Design Science Methodology for Information Systems and Software Engineering" [Wie14]. Design science is a specific methodology created for research in the field of technology and is fundamentally different from the methodology used in natural science. While natural science aims to gain knowledge about the world without changing it, design science is driven by the intention of solving or mitigating a problem in a specific context. A key feature of design science is to investigate and design an *artifact* in a context. The artifacts should be designed to interact with a problem context in order to improve something in that context and to contribute to the stakeholders' goal. An artifact is broadly defined as *"something created by people for some practical purpose"* [Wie14]. Examples of artifacts can be methods, techniques, notations, or algorithms used in software and information systems. For this project, the artifact will be the method used for verifying students taking an exam. The stakeholder of a problem is defined as *"a person, group of persons, or institution affected by treating the problem."* [Wie14]. In this project, the stakeholders are represented by the administration of higher education, as the purpose of the designed artifact is to determine if stylometry and keystroke dynamics can be used to detect contract cheating. It is the stakeholders' goals and needs that are the source of the artifact requirements, so it is important to identify the stakeholder's goals.

Design science is a process composed of three tasks: Problem investigation, treatment design, and treatment validation. In problem investigation, the researcher should ask him- or herself the questions: *"What phenomena must be improved? Why?"* [Wie14]. In treatment design, the goal is to *"design one or more artifacts that could treat the problem"* [Wie14] In treatment validation, the task is to answer the question: *"Would these designs treat the problem?"* [Wie14]. This set of tasks is called the *design cycle* because the researcher iterates over the three tasks several times during a design science research project. For this project, the following is done

in each step of the design cycle:

- **Problem investigation**
  - Literature review
  - Semi-structured interview
  - Email interview
- **Treatment design**
  - Define requirements
  - Design artifact
- **Treatment validation**
  - Biometric performance evaluation

3.2 Problem investigation

The problem investigation stage of design science is all about doing the groundwork before the artifact is designed. Problem investigation does not consider any requirements for an artifact, but instead aims to *"identify, describe, explain, and evaluate the problem to be treated."* [Wie14]

There are many ways to investigate implementations and problems. For example: by reading the scientific, professional, and technical literature or by interviewing experts [Wie14]. For this project, a literature review combined with a semi-structured interview and an e-mail interview will be the chosen method to investigate the problem. The literature review will mainly focus on the theoretical and technical aspects in regards to the artifact. The semi-structured interview, as well as the e-mail interview, will investigate the context and stakeholders' goals by gaining qualitative insight into the potential real-world application of the artifact.

3.2.1 Literature review

A literature review is an analysis of available research literature or other published sources on a specific topic. The main goals of a literature review are to identify state of the art, provide an overview of key terms and concepts, and identify how one can contribute to the research. Hart [Har18] states the following on literature reviews:

> *"The review forms the foundation for the research. The researcher needs to know about the contributions others have made to the knowledge pool relevant to their topic. It is the ideas and work of others that will provide the researcher with the framework for their own work; this includes*

methodological assumptions, data-collection techniques, key concepts and structuring the research into a conventional academic thesis. "

The following paragraphs outlines the details of how the literature review is performed.

Searching the literature was done mainly through the search engine for academic literature, Google Scholar [1]. In addition, Microsoft Academic[2], Oria[3] and ScienceDirect[4] were used. When searching for relevant literature, the following keywords were used: *contract cheating, exam cheating, stylometry, author verification, author identification, NLP, machine learning, keystroke dynamics.* The keywords were used both alone and in combination with each other.

Reading and analyzing the literature is a time consuming and comprehensive task. To manage the substantial amount of research papers, articles, and websites found when searching for literature, a separate document was made to maintain an overview and group the different literature into sub-genres. The references related to the information were further investigated when relevant information was discovered in a paper or article.

Findings from the literature review are presented in Chapter 2. Many papers and articles were read to gain knowledge and understanding about the topic, but only the literature used directly in the thesis was added to the references.

3.2.2 Semi-structured interview and email interview

Design science emphasizes the importance of understanding the stakeholders' goals when treating a problem. While the literature review gave insight into the theoretical and technical aspects regarding stylometry and keystroke dynamics in authorship verification, little information was found regarding the specific use case of implementing a cheat detection solution using these methods. To better understand the stakeholders' goals, a semi-structured interview and an email interview of persons representing the administration of higher education institutions were performed. In a semi-structured interview, a set of questions are prepared, but there is room for improvisation [MN07]. The goal of a semi-structured interview is to create a structure that allows the interviewee to address issues that are not necessarily covered by the set of questions but is still relevant for the stakeholders' goals. The semi-structured interview was chosen due to the flexibility and opportunity to explore new aspects of the topic that was not initially considered.

[1] https://scholar.google.com/
[2] https://academic.microsoft.com/home
[3] https://bibsys-almaprimo.hosted.exlibrisgroup.com/primo-explore/search?vid=NTNU_UB&lang=no_NO
[4] https://www.sciencedirect.com/

Initially, the intention was to conduct three separate semi-structured interviews. However, due to the Corona-virus outbreak, we were only able to schedule one semi-structured interview conducted through a video conference. The candidate is a person working with digital evaluation within higher education in Norway. The candidate works directly with the technical aspects of digital exams and has insight into the current status of different cheat detection methods. The goal of the interview was to acquire insight that can be used when designing the requirements and implementing a method for detecting contract cheating. The prepared questions can be found in Appendix A. In addition to the semi-structured interview, an email interview with a person working directly with the processing of cheating cases within higher education in Norway was conducted. The candidate is an expert in the legal concerns related to cheating. The goal of the email interview was to investigate how the requirements of the cheat detection method could be adapted to facilitate the legal prosecution of cheaters. The questions and answers in this email exchange can be found in Appendix A.

3.2.3 Key findings from the interviews

Semi-structured interview

The first part of the interview was dedicated to gain an overview of how the problem of academic dishonesty is viewed within higher education in Norway, as well as how cases of cheating and plagiarism are processed. The interviewee did not consider cheating an increasing challenge considering the transition to digital evaluation, based on the amount of cheating and plagiarism cases in recent years. Nor could the interviewee report of any cases where cheating through third party assistance was provided. However, the subject of contract cheating was known to the interviewee through media reports from different educational communities, and that there is a good reason to believe that such cheating forms are being conducted at Norwegian universities and colleges as well.

Regarding the processes in the handling of cheat cases, the interviewee explained that in case of plagiarism, the sensor of the examination would be notified with a report containing a percentage of plagiarized work in the exam submission. These percentages are divided into categories ranging from minimal to severe. Based on the severity of the plagiarism, the report will be discussed with the relevant institute whether or not a case should be created. This case can be forwarded to the exam office, where a jurist will decide the outcome. The interviewee stated the importance of a score to indicate the confidence of a prediction for a cheat detection method using authorship verification, similar to how plagiarism detection works.

In the interview, we wanted to ask if any particular methods were used to mitigate or prevent cheating on home exams today. We pointed out to the interviewee that

many exams and evaluations are being transitioned from classrooms to remote and digital assessments due to the Corona outbreak. We learned that solutions like online proctors, examination proctors guarding the examinee through live video streams, had been reviewed and researched. Online proctors have, however, never been applied in exams as of yet, as the resources demanded to implement online proctors did not weigh up in the amount of cheat prevention value. As of today, there is no method or system other than plagiarism control for preventing cheating on home exams.

The interviewee stated that implementing a system based on stylometry and keystroke dynamics could be demanding, but possible. Regarding the privacy challenges in capturing keystrokes, the interviewee did not see any boundaries in making it possible by extending the Inspera Assessment tools like Safe Exam Browser, considering that all data going through their evaluation software is anonymized. The interviewee stated that implementing keystroke capture software in a home examination program should not be challenging on a technical level. We discussed the importance of accuracy of our proposed system, where the interviewee stated that it was natural to design a method where more importance is put on classifying as few non-cheaters as cheaters as possible, while still detecting real cases of contract cheating.

Email interview

Due to the severe penalty for cheating, the interviewee explained that educational institutions will only pursue cheating cases if it is highly probable that a student has cheated. In practice, a case of suspected cheating will not be pursued if there is any ambiguity. The interviewee stated that the presentation of concrete evidence is crucial in a cheating case. If there is evidence enough to say that there have objectively been detected cheating, a review must be done to determine if the requirement of fault is compliant with the Norwegian law of Universities and Colleges §4-7 [uhl]. In other words, if the student has acted willfully or if the student has acted negligently.

The interviewee emphasized that a system using stylometry and keystroke dynamics to detect cheating needs to be almost flawless for a cheating case to be based entirely on that system. The interviewee stated that the system could be interesting to consider along with other factors, in a case where there is other evidence to support that a student has cheated. The interviewee provided an example of a third party claiming that he or she has written an exam for a student. In that scenario, where it is word against word, the system could be used as evidence to support the student's innocence or guilt.

3.3 Treatment design

In treatment design, the decision on how to design the artifact is made. The goal of design science is to create an artifact to treat a problem in its context. The word *treatment* is deliberately used instead of solution, *"because the word solution may blind us for the possibility that an artifact may solve a problem only partially or maybe not at all"* [Wie14]. This fits well into this project, as a system that completely eliminates the problem of contract cheating through stylometry and keystroke dynamics is probably impossible to design. While the literature review indicates it should be feasible to design an artifact that successfully detects a high number of cheaters, it is not realistic to expect to solve the problem of contract cheating for good.

When designing the treatment to the problem, a key element is to create artifact requirements. In design science, *the requirements are defined as the treatment properties*. The requirements should be specified in order to contribute to the stakeholders' goal in the problem context. For this project, the stakeholders are the administration of universities, and the stakeholders' goals are formulated from the literature review and the interview in the problem investigation phase. To justify a requirement, one should use the **contribution argument**, which is *"the argument that an artifact that satisfies the requirements, would contribute to a stakeholder goal in the problem context"* [Wie14]. The contribution argument has the form [Wie14]:

(Artifact Requirements) x (Context Assumptions) contribute to (Stakeholder Goal)

The requirements will be designed based on the information gathered from the problem investigation phase and will be used as a basis for the design of the cheat detection method. To best capture the isolated differences of stylometry and keystroke dynamics for cheat detection and to facilitate the fusion of the two approaches, a common method applicable for both approaches will be designed. In design science, the design of artifacts is a dynamic process, and several different approaches will be considered before a final treatment is proposed. The final cheat detection method, including the technical details, is presented in Chapter 5.

3.4 Treatment validation

The last step of the design cycle is treatment validation. In this step, the goal is to validate if the treatment design would treat the problem. *"To validate a treatment is to justify that it would contribute to stakeholder goals if implemented."* [Wie14]. The treatment validation should tell us what would happen if the artifact were implemented in the problem context, in this case, how well would the contract cheating detection method perform if implemented in a real exam scenario.

As explained in Chapter 2, stylometry and keystroke dynamics are examples of behavioral biometrics. A system that utilizes these features to verify whether a student has cheated or not is a specific case of a biometric system. This project will thus use a procedure called *biometric performance evaluation*. Biometric performance evaluation is defined as [RK11]:

> *"A procedure of quantifying the performance of a biometric recognition system under given conditions. The goal of such an evaluation is a comparison to another system, or a prediction of the system's performance using unseen biometric data, collected under similar operating conditions"*

Biometric performance evaluation is standardized by ISO/IEC [5], and is a recognized method for determining how a biometric system will perform in a real-life scenario. The basic principle of biometric performance evaluation is to utilize the full dataset to perform *genuine* and *impostor* attempts, in order to measure how well the system can differentiate between them. A genuine attempt is defined as *"A single attempt by a user to match his/her own stored template."*, and an impostor attempt is the opposite, *"A single attempt by a user to match someone else's template."* [Bio14]. When applying the biometric performance evaluation in this project, genuine attempts will represent a student delivering an exam they have written themselves (non-cheater), while impostor attempts will represent a student delivering an exam they have not written themselves (cheater).

The results from the biometric performance evaluation will be presented in Chapter 7. The details of how the biometric performance evaluation is implemented and measured in this project are described in Chapter 6. To validate the treatment, the biometric performance evaluation will be performed on three different datasets: One for stylometry and keystroke dynamics individually, and one where both approaches will be tested on the same dataset.

[5] https://www.iso.org/standard/41447.html

Chapter 4
Datasets

This chapter contains information about the datasets used in this project. In total, three different datasets were used. Two of them were gathered from external sources, while one dataset came from our own data collection.

- **PAN 2013:** Texts from the authorship verification task from the PAN 2013 competition [JS13].
- **Data Collection Experiment (DCE):** Keystrokes and text answers collected from students at NTNU, Norway.
- **Stewart:** Keystrokes and text answers collected from students taking a test at Pace University, USA [SMCT11]

The cheat detection method presented in Chapter 5 is tested on all three datasets individually. The PAN 2013 dataset contains only text, and will thus only be tested using stylometry. The Data Collection Experiment (DCE) dataset contains both text and keystrokes, but due to incomplete text data, only the keystrokes will be tested. For the Stewart dataset, both stylometry and keystroke dynamics will be tested.

This chapter describes the original structure of the datasets. Modifications will be done to the datasets in order to be usable in the biometric performance evaluation. These modifications are presented in the introduction of the results in Chapter 7.

4.1 PAN

The first dataset was gathered from the PAN competition on author verification from 2013 [JS13]. *"PAN is a series of scientific events and shared tasks on digital text forensics and stylometry."*[1] On their website, several different tasks on authorship

[1] https://pan.webis.de/shared-tasks.html

analysis is available. PAN has had three different competitions on author verification, in 2013, 2014 and 2015, in addition to one this year. The format of each year's competition has been the same every year: *"given a set of documents by a single author and a questioned document, the problem was to determine if that particular author wrote the questioned document or not."* [JS13]. In this project, only the English part of the dataset is used, as this is the most relevant in the context of exam cheating in a Norwegian university. The other languages covered in the datasets are Spanish, Greek, and Dutch.

The datasets from all the different authorship verification tasks were considered, and the dataset from 2013 was chosen due to the characteristics of the corpus. The corpus in the 2013 task was more in line with the objective of this project, as the texts are collected from different textbooks in the genre of computer science. Due to the academic writing style, this can be compared to a home exam for a computer science class, where the students are asked to write about a topic related to computer science.

4.1.1 Dataset description

The dataset is structured as a set of folders, each representing an author. Each folder contains a set of *known* documents: texts that are indisputably written by the same author. The number of known documents per author ranges from one to five. Each folder also contains exactly one *unknown* document, which is the document in question. In the PAN competition, the goal was to decide whether the unknown document in each folder were written by the same person as the known documents. The structure of the dataset is illustrated in Figure 4.1. The dataset consists of 30 different folders. The distribution of true and false are: 14 true and 16 false, i.e., 14 of the folders contain an unknown document labeled *same-author*, and 16 folders contain an unknown document labeled *different-author*. Folders, where the unknown document is labeled different-author, are comprised of documents from the folders where the unknown document is same-author. For example, folder one can contain the same documents as folder two, except for the unknown documents. In that case, only one of the folders will contain an unknown document from the same author. This structure has relevance for the biometric performance evaluation used in this project and will be described further in Chapter 7.

The corpus is constructed from the works of a total of 16 different authors. All the texts are collected from the genre *textbooks regarding IT or computer science*, but the topic of each text is varied inside that genre. Examples of different topics in the corpus are Java programming, cybercrime, and digital system design. This variation is introduced to make the verification task more difficult by not making it possible to guess authorship based on the topics. Each document is around 1,000

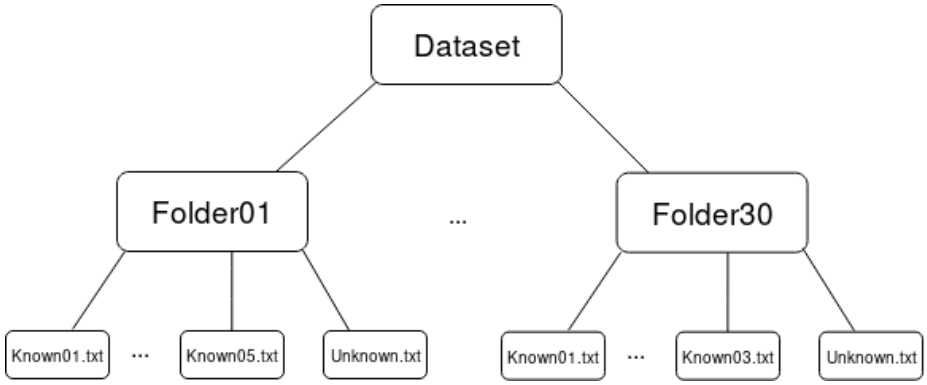

Figure 4.1: Structure of the dataset used in the PAN-2013 competition

words that are extracted from longer passages of text in the books.

4.2 Data Collection Experiment

In order to gather data to be used in this project, a DCE was initiated. This data was to be collected by having participants answering 11 questions, where both the text and the subject's keystrokes were captured. These questions were to be answered on a web application developed by researchers at NTNU Gjøvik. The questions were related to information security and were chosen specifically to mimic a home exam. The participants had to sign a consent form explaining what the data was to be used for, and they were handed usernames and passwords for logging themselves into the web application. The experiment could be performed from any place through the web application, but a meeting where the participants could collectively answer the questions was planned nonetheless. The meetup was planned to gain an oversight of the participants, as well as to give the subjects an incentive to participate. Unlike the data collection for the Stewart dataset, where every participant was assigned a desktop computer with similar keyboards, the test subjects were to bring their laptops. The participants mainly consisted of students attending their last years of the Communications Technology master's program. Other participants were acquired through other acquaintances and friends from the university. 40 participants agreed to contribute to the experiment, either remotely or during the meeting. Despite the excellent turn-up for the experiment, only 23 of the participants completed all 11 questions. Many of the test subjects found the experiment too long and demanding since the subjects had exams and projects to focus on. Not all test subjects were supervised at the same time, and almost half of them were to finish the questions remotely, which made it hard to gain any oversight of the participants answering

the questions. The average amount of keystrokes per question answered in the experiment was about 942 keystrokes. However, many questions consisted of answers of keystrokes less than 500, while some answers consisted of over 1300 keystrokes.

The experiment encountered some problems during the capture of the text data. On some occasions, the texts submitted by the participants were incomplete or abruptly cut off in the middle of a sentence. This occurred when a test subject went on to answer the next question before correctly submitting the previous question. During the meetup, these problems were, for the most part, avoided, as the participants were guided on the proper protocol on how to answer the questions. Although this problem had little to no effect on the quality of the keystroke data, this affected the quality of the text data that could be used for the stylometry analysis.

4.2.1 Dataset Description

The keystroke data from the DCE was stored in a *.csv*-file, represented as a Pandas dataframe[2] in figure 4.2.

	user	session	keystroke_in_session	keycode	duration	latency
0	1	1	1	16	878	-48
1	1	1	2	83	55	433
2	1	1	3	8	55	129
3	1	1	4	16	152	-96
4	1	1	5	80	80	56
...	...	...	...	...	...	...
302594	40	1	345	65	88	39
302595	40	1	346	66	73	84
302596	40	1	347	76	98	-32
302597	40	1	348	69	114	850
302598	40	1	349	190	101	-1000

302599 rows × 6 columns

Figure 4.2: Excerpt of the keystrokes data from the DCE

Each row in the dataframe shows from which *user* and at what question (session) the keystroke was captured. *Keystroke in session* tells the index of the keystroke within that given session, and *keycode* is the ASCII-number of the corresponding key pressed on the keyboard. The duration values in the dataframe were calculated by subtracting the key-release time with the key-press time for every row, while the latency values were calculated by finding the difference of the key-release time and

[2]https://pandas.pydata.org/

the key-press time of the following key-press. The last keystroke captured for every session did not get a latency value, because there were no following keystroke to use in the latency calculation. The latency values for these keystrokes were therefore stored with the value -1000.

4.3 Stewart

The data that is coined the *Stewart dataset* for this project, was used in the publications [SMCT11] and [MSCT13]. The data was collected from students attending a spreadsheet modeling course from a liberal arts college at Pace University. 40 students met at a laboratory where each student was assigned a desktop computer. These were Dell desktop computers with associated Dell keyboards. Earlier research suggests that the subjects should use the same type of keyboard for optimal results; it is shown that keystroke dynamics can vary for different types of keyboard and different types of environment [TVC10]. Even though the experiment was carried out through an online application, data acquisition and capture were carried out in a computer laboratory to gain experimental control. The participants were asked to answer four tests with short answers, where each test consisted of 10 questions each. During this experiment, the keystrokes, as well as the textual inputs, were recorded from the participants without them being aware that their data was captured for later analysis. The researchers encountered several problems with the collection, including students forgetting usernames and passwords, the weakness of the server hosting the web application, and general slowness of the system response. Despite the problems with the data collection system, the experiment resulted in a complete data set from 30 students. The average lengths of the answers are 96 words. With an average word length of five characters plus one counting the space, the results yielded about 600 keystrokes captured per answer. In the dataset, one answer to a question is called a *session*.

4.3.1 Dataset Description

The dataset is divided into two different .csv-files: one for stylometry and one for keystrokes. Excerpts of the two files represented in a Pandas dataframe is shown in Figure 4.3 and Figure 4.4 respectively. There were some problems with the representation of the user and session IDs in the Stewart dataset that we acquired. The .csv-file containing the stylometry data included 30 users with 40 texts each. However, the .csv-file with the keystroke data consisted of the keystrokes from 43 different users. These users had session counts varying from 10, 20, 30, and 40. As the data was meant to represent 30 individual users, some users in dataset are represented under several user IDs. How this obstacle was handled is explained in Chapter 7. In addition, the stylometry and keystroke data did not have matching user and session IDs. There was, however, a time start and a time end for each

session, that made it possible to create a script and map the sessions in the stylometry .csv-file to the corresponding user and sessions from the keystroke .csv-file. This was done in order to make it possible to compare the result from the stylometry and the keystroke dynamics methods. For example, in the stylometry dataframe shown in Figure 4.3, user 169 and session 2004 corresponds to user 1 and session 1 in the keystroke dataframe from Figure 4.4, due to timestart = timepress for the first key pressed in this particular session.

	user	session	timestart	timeend	text
1073	169	2004	1304433167859	1304433435103	Coefficient of determination measures are used...
1074	169	2024	1300198269586	1300198613338	Optimization is when you seek out for the best...
1075	169	2056	1301581759499	1301582060280	Data mining is just as it sounds we are mining...
1076	169	2069	1301583726832	1301584019190	Bayes Therom is when you obtain probabilities ...
1077	169	2085	1303395069492	1303395303373	The decision that you face when managing inven...

Figure 4.3: Excerpt from the stylometry data in the Stewart dataset

	user	session	timepress	timerelease	keycode	keyname
0	1	1	1304433167859	1304433168307	16	shift
1	1	1	1304433168227	1304433168371	67	c
2	1	1	1304433168291	1304433168451	79	o
3	1	1	1304433170051	1304433170179	69	e
4	1	1	1304433170451	1304433170531	70	f

Figure 4.4: Excerpt from the keystrokes data in the Stewart dataset

Chapter 5
Treatment design

This chapter presents a detailed explanation of the methods created to detect contract cheating. In the following sections, the requirements of the artifact and the overall detection method used for both stylometry and keystroke dynamics (KD) are described. Then the specific technical implementation choices made in stylometry and KD respectively are presented. Finally two different methods for combining the results from stylometry and KD are described.

5.1 Artifact Requirements

Based on the information from the problem investigation and the research questions for this project, two requirements for the method has been defined:

- **R1:** The method must be designed to focus on avoiding false accusations while still detecting cheating cases
- **R2:** The method must provide a probability-score to indicate the confidence in a prediction

In the process of designing the cheat detection method, these requirements have influenced the choices made and the technical details of the method.

5.2 Detection method

Determining if a student has cheated or not based on stylometry and KD can be approached from many different angles. In this project, a general method is developed in order to easily compare the effect of stylometry vs. KD, in addition to facilitate the fusion of the two approaches. When describing the method, the word *features* will be used as a general term, covering both the textual features and keystroke

features. *Pandas* is used in order to manipulate and structure the data, [1]. Pandas is a Python library that offers the possibility of structuring data in *dataframes*, which is a 2-dimensional labeled data structure. A detailed description of how the actual feature extraction was performed for stylometry and KD will be explained in the sections covering the specifics of each approach. For this project, as for all authorship verification tasks, the foundation of the method revolves around the assumption that existing samples of text/keystrokes that indisputably originate from the students, are available.

5.2.1 Binary extrinsic model

An introduction of the extrinsic and intrinsic model was presented in Chapter 2. From the literature review, it is clear that the extrinsic model generally performs better than the intrinsic model in authorship verification tasks. The extrinsic model uses features (textual or keystrokes) from other authors to create a binary classification task. In contrast, the intrinsic model only uses features from the given author to determine if an unknown sample is written by the same person or not, i.e., determining if it is an outlier in the class of known samples. Due to the findings in the literature review, an extrinsic model was used when creating a model for detecting contract cheating. The extrinsic model is exemplified in the "Impostor method" [KW14] reviewed in Chapter 2, which has had a significant influence on the topic of author verification.

The extrinsic model's basic idea is to use external features to represent the negative class in a binary classification problem, i.e., as a representation of all features not belonging to the person in the positive class. In simple terms, the method will take a sample from an unknown author as input, and determine if it is more similar to the positive class (samples from a specific person) or the negative class (external samples). The idea behind the extrinsic model is fundamentally fallible. A scenario where a cheating student delivers an exam that by chance contains features more similar to his or her style than the negative sample class is entirely plausible. It is however, less likely that an exam from a non-cheating student will be more similar to the negative class, assuming the classifier successfully can recognize similarities in style. The decision to use an extrinsic model is thus in line with requirement R1, as this will reduce false accusations at the cost of accepting more cheaters.

When testing the method, the external samples that form the representations of negative features will be chosen randomly from the pool of authors/users for the given dataset. The idea is to simulate an exam scenario where the number of unique authors in the dataset represents the students from a university class. All the datasets used in this project contain samples that belong to similar topics or genres.

[1] https://pandas.pydata.org/

This could be compared to a dataset containing real samples from students in the same university class.

5.2.2 Instance-based method using machine learning

In authorship verification, there exist two main approaches for using the features to represent an author, namely profile-based and instance-based. In the profile-based method, the features from all samples are combined to represent the author's style. While this is more common for stylometry-based authorship verification, where all the texts are combined into one large text file, it can also be done for KD, as explained in Chapter 2. In the instance-based method, all the known samples from a given author will be treated individually as a separate instance of the authorial style. The advantages of this method are that individual differences between the samples will be regarded, while this is not the case for the profile-based method. In this project, each sample from an author will be used as input into a machine learning classifier, which will treat all samples as separate instances from the class it belongs to. The method created in this project is thus instance-based. However, a combination of instance-based and profile-based will also be tested. This is done by merging several samples and treating the merged samples as one single sample, or instance. There are many different ways to implement an instance-based method. In this project, a machine learning classifier is used to predict the probability of a sample belonging to a given class. The instance-based method created for this project is illustrated in Figure 5.1.

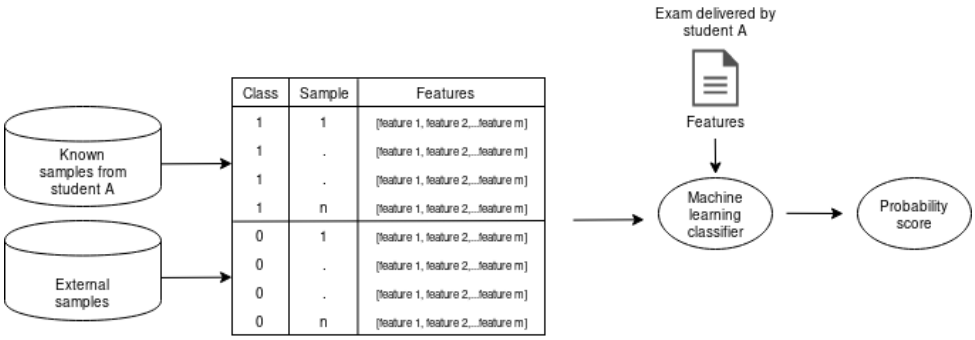

Figure 5.1: The method created to verify if a student has cheated or not

Figure 5.1 is an illustration of how the method would work in a real-life scenario for a specific student and a specific exam. In accordance with requirement R2, the method outputs a score that indicates the probability of the exam being written by the student who delivered it, in this case, "Student A." The selection of samples belonging to class 0 is randomly chosen from the pool of external samples, while the

samples belonging to class 1 are all the samples available from Student A. This creates a binary classification problem, where the machine learning classifier will train on the samples from class 1 and class 0. The exam in question will then be tested against the trained model, and ideally produce a high probability-score when a student who delivered the exam has written the exam him- or herself (non-cheater), and a low probability-score if the student has not written the exam him- or herself (cheater). In order to reduce the influence of randomness, the average probability-score of 10 separate iterations is calculated. This is done due to the random selection of samples for class 0. Class 1 will remain the same, but new samples for class 0 are randomly picked for every iteration. As the negative samples is meant to represent the general features of "not student A", taking the average of 10 iterations will better capture the overall negative sample pool.

The effectiveness of this method will be measured using the biometric performance evaluation, described in Chapter 6.

5.2.3 Classifiers

Three different machine learning classifiers are tested in order to compare their performances. The classifiers chosen are Support Vector Machine (SVM), Naive Bayes (NB), and Logistic Regression (LogReg). The technical details explaining how they work are presented in Chapter 2. These classifiers were chosen due to being recognized as useful for binary classification tasks. All classifiers are implemented using the *scikit-learn* library in python[2]. LogReg and NB have the advantage of being naturally probabilistic classifiers, meaning they produce a probability-score for a given input. SVM, on the other hand, only produces the class that the input most likely belongs to. In order to obtain a probability-score from SVM, CalibratedClassifierCV from scikit-learn is used. The CalibratedClassifierCV is a class that transforms a prediction from SVM into a probability-score using probability calibration.

5.3 Stylometry

Features

In machine learning, features are the actual input that the model uses to create some output. The features are the properties of the data that the machine learning model uses to predict the results. In the stylometry part of this project, Term Frequency-Inverse Document Frequency (TF-IDF) is used to transform the texts into vectors that are used as features. TF-IDF calculates the term frequency (TF), meaning it measures how often an n-gram occurs in the text and normalizes the vector by dividing the total number of n-grams in the document. In addition to the term

[2]https://scikit-learn.org/stable/

frequency, TF-IDF also uses inverse document frequency (IDF), meaning it weights the n-grams based on how often they occur in the training data, i.e., the total number of documents in the classification. This means n-grams present in several samples of the training data, e.g., words like "if" and "that" will be given less importance in classification. The technical formulas for calculating the TF-IDF is provided in subsection 2.4.2. TF-IDF is implemented using scikit-learns TfidfVectorizer [3].

When using TF-IDF to extract features, either character level or word level n-grams can be used. The most common practice is to use word-level 1-grams (unigram). However, the literature study revealed that some experiments had obtained better results by using character-level n-grams and word level n-grams with $n > 1$. In this project, in order to explore the difference in performance, both character and word level n-grams with different n-values are used, in addition to traditional unigrams. All features used in testing are presented in Table 5.1. The parentheses represent the range of n-values used, for example, (1,2)-gram means 1-gram and 2-gram used together.

Word level	1-gram, 2-gram, (1,2)-gram
Character level	3-gram, 4-gram, (3,4)-gram

Table 5.1: The different features being used in the stylometry part of this project

TF-IDF is chosen as the feature representation in this project due to being consistent in its performance, as was learned from the literature review.

Pre-processing

Pre-processing is often used in different text-classification tasks to filter out words and symbols that are not relevant or important for the specific task. For tasks related to stylometry and personal writing style, pre-processing needs to be done carefully. A common technique in pre-processing is to remove "stop-words," which are common words for a specific language. In English, words like "is," "a," and "the" are examples of stop-words. While removing these words can improve the results in a model that aims to distinguish texts based on topic, an individual's use of stop-words can be essential in capturing his or her writing style.

The pre-processing applied to the text in this project was inspired by the paper "Improving cross-topic authorship attribution: The role of pre-processing" [MSS17]. The paper presents the effect of different pre-processing techniques when tested on

[3] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

authorship attribution. Two different pre-processing techniques are used in this project:

- **Replacing digits:** Actual numbers do not reveal any information about the style of an author, but how they are represented do. In order to capture the stylistic information from numbers, all digits are replaced by 0. For example, 12,000 is replaced by 00,000, and 12000 is replaced by 00000.
- **Splitting punctuation marks:** When applying different variations of n-grams, stylistic information about the frequencies of punctuation marks can get lost. For example, when applying character 3-grams to the word "exam!", the only information captured about the exclamation mark will be combined with the adjacent characters, e.g. "am!". By adding a space between all punctuation marks, a character 3-gram would be able to capture the single use of the exclamation mark by producing the 3-gram "-!-". This will allow the n-grams to better capture the differences in frequency of punctuation marks, which can be a characteristic that helps identify a person's writing style.

In this project the testing was performed both with and without pre-processing in order to review the effect.

5.4 Keystroke Dynamics

Data filtering and pre-processing

The raw data used for developing the method for verification through KD came in the form of a table where every single keystroke was a single row. Every row included a *user identification* and a *session identification*, that indicated which user had pressed the captured key - and at which question/sample the subject had pressed the key on. Each row also came with the ASCII-keycode[4] corresponding to the pressed key.

Although there are several features to consider when studying behavioral typing patterns of a subject, only two keystroke features where studied in this project. These two typing features are *duration*; which refers to the time between a single key on a keyboard is pressed down until it is released and *latency*; which refers to the time between one key is released, and the next key is pressed down. The duration and latency metrics are illustrated in Figure 5.2.

In the datasets used for the KD analysis, every keystroke was logged with a timestamp telling the time when the key was pressed down, as well as a timestamp

[4]https://www.ascii-code.com/

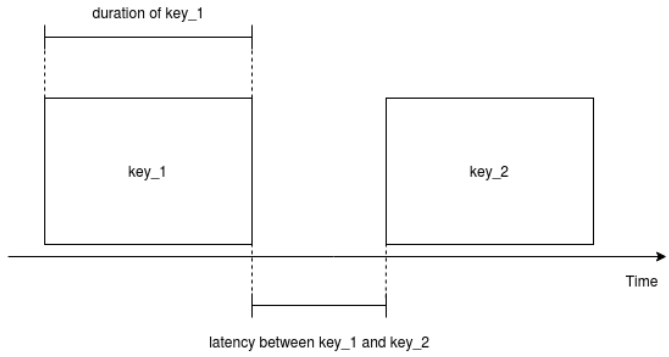

Figure 5.2: Duration and latency as metrics when keys are pressed on a keyboard. Latency can also be negative; where a second key is pressed before the first key is released.

when the key was released. These timestamps are logged down to the millisecond. In order to find the duration for every keystroke captured in the data, every key-release timestamp was subtracted with every key-press timestamp for every row. In order to find the latency of the transitions of keystrokes, the timestamp for key-press was subtracted with the timestamp for key-release with the previous keystroke. The latency computation was completed for every row except the last keystroke in every answered question, where the absence of a following keystroke do not generate a latency value. These keystroke rows were stored with a latency value of -1000.

After selecting which users and samples to be included in the model, *outlier* keystrokes in the datasets were removed. These outlying values were latency - or duration values that were either caused by writing pauses during the data collection activity or small errors in the capture software. In [MSCT13], the researchers stated that coffee breaks and general pauses taken by the test subjects in the data collection, made some of the keystroke capture abnormally long and inconsistent. Similar breaks and pauses were observed during our DCE. These outlier latency values were therefore filtered out by first calculating the mean and standard deviation over all the latency values, and then removing the latency values greater than two standard deviations from the mean, over the whole dataset. This outlier removal process were iterated multiple times until no changes occurred. The same outlier removal process was conducted in this project.

Features

Two different sets of features were extracted from the filtered and processed set of keystrokes; one set for the durations on key-presses done by the users - and one

set of features containing the latencies calculated from the transitions between two different keys. Figure 5.3 depicts the different duration features extracted for every session done for every user. The employed duration features consist of the means and standard deviations of the key-press durations depicted in the ellipses in the figure. One mean value and one standard deviation value for every ellipse gives us 31 duration feature pairs. The final choice for which key durations to be used in the feature extraction were primarily based on testing with a larger array of duration features beforehand. This larger set included the durations on keys that are not widely used in the English alphabet; such as the letters **x**, **z**, **q** and so on. It also included other non-letter keys that did not occur enough times in the keystroke datasets to give any analytical value in the calculations of the means and standard deviations.

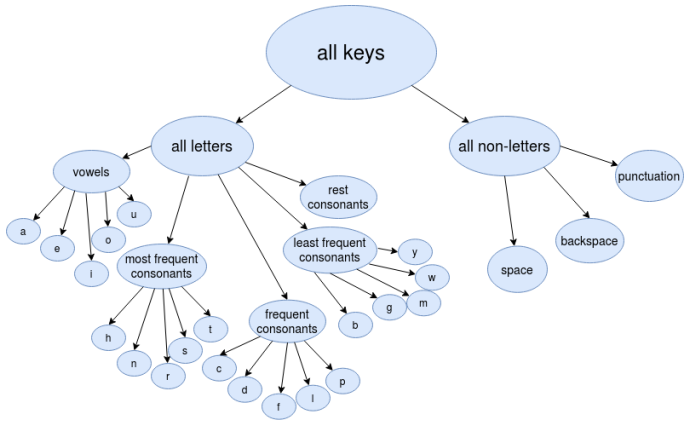

Figure 5.3: Tree structure with every duration feature employed in the model. Every ellipse correspond to a feature pair; the mean and standard deviation of the key duration. This tree structure is based on Figure 1 in [MSCT13].

Figure 5.4 depicts the 20 latency feature pairs extracted. As for Figure 5.3, every ellipse corresponds to one mean - and one standard deviation value, but for a specific *key-to-key* transition. Also, testing for which latency values to include in the final feature set was conducted. Originally, the plan was to include latency features representing several key transitions between non-letters like **space**, **punctuation**, **shift**, as well as the transitions from non-letters to letters and opposite. However, results from the feature extraction showed that these latency values were either very sporadic and uneven over several sessions of the same user, or simply not available because of the removal of too high latency values in the pre-processing of the data. The calculated latencies of several non-letter transitions indicated a relationship between the pressing of a non-letter, followed by pauses in writing over a longer

period.

Theoretically, one could construct a feature set with the mean and standard deviations of every key-transition latency possible on a keyboard. For a standard Windows keyboard with 104 keys, this would result in 10816 different key-to-key transitions. The feature extraction in this model was constructed such that only the latencies of the transitions with at least three occurences in a session were extracted.

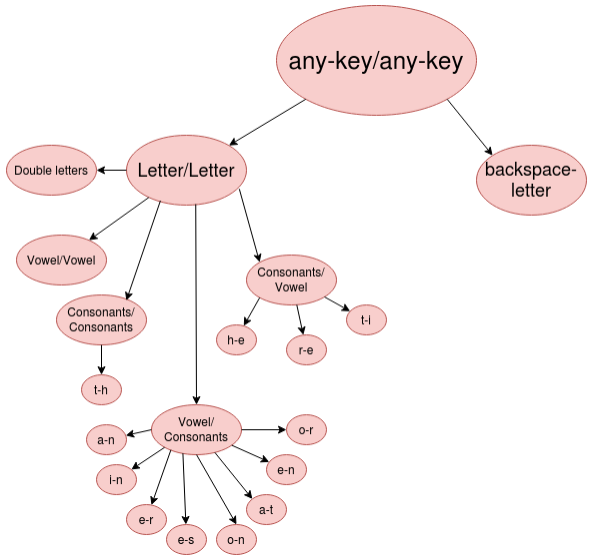

Figure 5.4: Tree structure with the 20 latency feature pairs employed. The tree structure is based on Figure 2 in [MSCT13].

Finally, the features were standardized into the fixed range of 0 to 1 by using *Min-max scaling*. This normalization process was done because two of the three classification methods used in this thesis would not function optimally otherwise. Standardization of the feature values is advantageous in using SVM, because it helps in avoiding features with greater values dominating features with smaller values. For NB, the classifier implementation fails if there are negative values in the data. The latency values can on many occasions be negative; where a second key is pressed before the first key in a transition, is released. Min-max scaling was also used to bypass this. The min-max scaling gave the features roughly an equal weight, and the formula is given by

$$m = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{5.1}$$

where m is the new feature value, x_{min} is the lowest value of all the values within a feature, and x_{max} is the highest value of all the values within a feature.

5.5 Combining stylometry and keystroke dynamics

To answer research question RQ2: *"How can stylometry and keystroke dynamics be combined to improve the author verification of the separate systems?"*, two different fusions of the two approaches are created. The fusion of the two approaches will be tested using the Stewart dataset, as this contains both stylometry and KD data. In order to accomplish the fusion of the two approaches, a fixed subset of the dataset was created for the Stewart dataset, which is explained further in the results in Chapter 7. By using the same samples and user IDs, it was possible to look at every sample and use results from the stylometry and KD classifications to produce a joint decision. The results from the fusions will be compared to the results from the individual results from stylometry and KD, in order to determine if the results can be improved by combining both approaches.

5.5.1 Unanimous decision

In accordance with requirement R1, this fusion of the two approaches focus on reducing the number of false accusations. The fusion is based on the principle of Unanimous Decision: Unless both the method using stylometry and KD classifies a student as a cheater, the student will not be accused of cheating. Pseudo-code for the unanimous decision fusion is presented in algorithm 5.1.

This fusion will not take the aggregated values of the two probability-scores into account. Each classification is performed individually, and the resulting classification from stylometry and KD is used as individual votes in the final decision. This fusion does not fully utilize the combined information from both probability-scores. For example, a scenario can occur where KD have a very low probability-score for an attempt, meaning it is very likely a cheating student. If the probability-score for the stylometry approach is only slightly above the threshold for classifying an attempt as a cheater, the unanimous decision fusion would classify this attempt as a non-cheater. This characteristic is both the strength and weakness of this fusion. On the one hand, it minimizes consequences (given that false accusations are more serious) of either stylometry or KD being wrong. On the other hand it will not utilize the cases of highly confident probability-scores.

5.5.2 Aggregated scores fusion

The aggregated scores fusion will use the aggregated probability-scores from stylometry and KD for each sample to determine if a student has cheated. The pseudo-code for the aggregated scores fusion is presented in algorithm 5.2. A characteristic of this fusion is that highly confident probability-scores from either stylometry or KD will influence the final decision more than less confident probability-scores. Assuming very high or very low scores are only given in obvious cases and is always correct,

this will be an advantage. On the other hand, this fusion will be more vulnerable to outliers that are wrongly classified in either stylometry or KD.

Algorithm 5.1 Unanimous decision

```
1: procedure DETECT CHEATING(a, b, c, d)
2:     Score_KD ← a
3:     Score_Style ← b
4:     Threshold_KD ← c
5:     Threshold_Style ← d
6:     Cheater ← False
7:     if Score_KD < Threshold_KD and Score_Style < Threshold_Style then
8:         Cheater ← True
9:     end if
        return Cheater
10: end procedure
```

Algorithm 5.2 Aggregated scores

,

```
1: procedure DETECT CHEATING(a, b, c)
2:     Score_KD ← a
3:     Score_Style ← b
4:     Threshold ← c
5:     Cheater ← False
6:     if Score_KD + Score_Style < Threshold  then
7:         Cheater ← True
8:     end if
        return Cheater
9: end procedure
```

Chapter 6
Treatment Validation

As explained in Chapter 3, biometric performance evaluation will be used to validate the treatment in this project. This chapter will first introduce the fundamentals of biometric performance evaluation, followed by the specific details of how it is implemented in this project. Finally, the metrics used to measure the results will be explained. The chapter is based on the papers "Biometric Systems Evaluation" [RK11] and "Understanding biometric performance evaluation" [Bio14].

6.1 Fundamentals of biometric performance evaluation

The purpose of most biometric systems is to verify a person's identity based on biometric characteristics. When a person tries to be verified by the system, it is called an **attempt**. The necessary procedure for an attempt is: A user produces a sample, and the features from the sample are extracted. Then, the extracted features are compared to the claimed user's **template** that is stored in a database. This comparison produces a similarity- or dissimilarity-score, depending on the underlying comparison method. Based on this score, the user's identity will be verified or not.

This procedure applies to the majority of biometric systems, e.g., a system verifying persons' identities based on fingerprints, as well the contract cheating detection method in this project, where the biometric samples are text and keystrokes. The template used for comparison in this project is the trained machine learning model that is illustrated in Figure 5.1. A specific person's template consists of his or her known samples and randomly selected external samples.

Biometric performance evaluation is an evaluation method that utilizes the full dataset to simulate genuine and impostor attempts, and aims to answer two main questions [Bio14]:

– How could you measure the accuracy of a biometric system (or components thereof)?

– How to compare different systems with each other?

When performing biometric performance evaluation, every sample in the testing set will be used to represent an attempt. When a specific user's test sample is compared to his or her template, it is a genuine attempt, and when it is compared to someone else's template, it is an impostor attempt. All possible comparisons are made to capture the actual performance of a method, and the score for every genuine and impostor attempt is saved. The distribution of the scores from the genuine and impostor attempts can then be plotted, as illustrated in Figure 6.1.

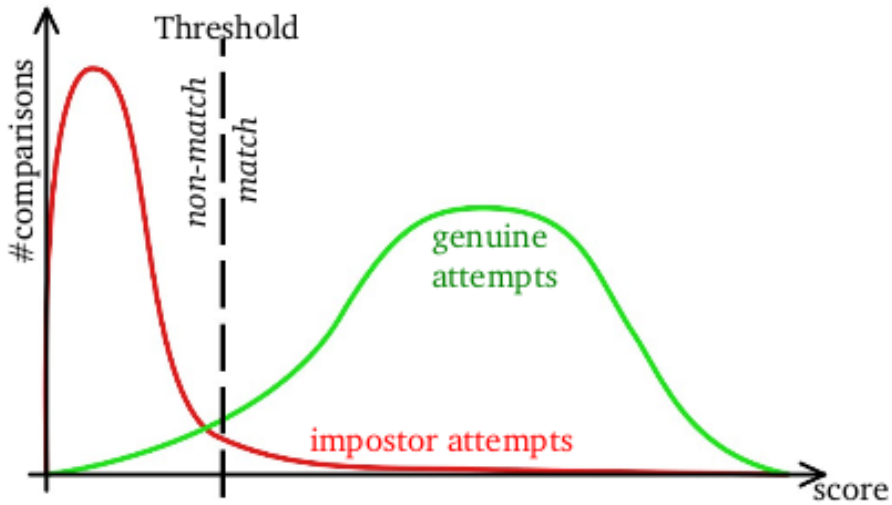

Figure 6.1: Distribution of impostor and genuine comparisons[Bio14]

Based on the plotted distributions, a threshold is set to separate the impostors and genuine users, i.e., cheaters and non-cheaters on the exam. Unless the system works perfectly, the two distributions will always overlap. The distribution plots will depend on the underlying method used to calculate the score for the genuine and impostor attempts, as well as the quality of the data. By increasing the threshold, fewer impostor attempts will be accepted, but at the expense of more genuine attempts being rejected. One of the advantages of biometric performance evaluation is the opportunity to adjust the threshold based on the purpose of the system. If the detection method were to be implemented as a real contract cheating detection

system, it would be natural to set a low threshold, and thus "accept" more cheaters to avoid false accusations.

6.2 Comparison scheme

A set of rules on how to utilize the dataset to simulate the genuine and impostor attempts needs to be in place in biometric performance evaluation. This set of rules is called the comparison scheme. For this project, the comparison scheme *leave-one-out cross validation* is used and consists of three main steps. The method explained in Chapter 5 and illustrated in Figure 5.1 is used as the underlying method for calculating the scores for each comparison/attempt. For an example dataset consisting of 10 users with 5 different samples each, the following steps would be carried out:

- **Step 1:** Extract one sample from each user. These samples will be the test data, i.e., used for the attempts (In total 10 samples)
- **Step 2:** Create a template for each user, consisting of the 4 remaining samples and 4 randomly selected samples from the other users (In total 10 templates)
- **Step 3:** Compare all of the 10 extracted samples from step 1 against all templates from step 2, and store the probability-score
- **Step 4:** Repeat step 1-3 five times. For every iteration, new test samples will be extracted in step 1.

For this example, one iteration would result in $10 \times 1 = 10$ genuine and $10 \times 9 = 90$ impostor comparisons. After five iterations, a total of $10 * 5 = 50$ genuine comparisons and $10 \times 9 \times 5 = 450$ impostor comparisons will be made. In general, for n users with m texts each, the comparison scheme will produce $n \times m$ genuine comparisons and $n \times (n\text{-}1) \times m$ impostor comparisons. In practice, each of the extracted samples that are tested against all the users' templates represents an exam, and each comparison simulates the users attempting to verify that he or she is the true author of that exam. If an extracted sample from user A is compared to user B's template, this will simulate user B trying to be verified as the true author of an exam he or she has not written and is thus an impostor attempt.

6.2.1 Performance Measurements

For measuring the system's performance, the saved scores for the genuine and impostor attempts are used. By applying a varying score threshold, one can calculate how many genuine and impostor attempts are classified correctly for different thresholds. In this

project, **the non-cheaters (genuine attempts) are defined as the "Positive" class while the cheaters (impostor attempts) are defined as the "Negative" class.** This means the genuine attempts with a score higher than the threshold will be classified as True Positives (TP), while the genuine attempts with a score lower than the threshold will be classified as False Negatives (FN). For the impostor attempts, scores lower than the threshold will be True Negatives (TN), while scores higher than the threshold are False Positives (FP).

Typically in biometric performance evaluation, the results are presented in pairs of False Non-match Rate (FNMR) and False Match Rate (FMR). However, due to the purpose of a cheat detection system, the results in this project will be presented in different terms. For an exam cheat detection method, it is most relevant to look at the probability of a cheater being caught (True Negative Rate) and the probability of a non-cheating student being falsely accused of cheating (False Negative Rate). The information contained in the True Negative Rate (TNR) and False Negative Rate (FNR) are identical to FNMR and FMR, as FNMR = FNR and FMR = 1 - TNR. The AUROC-score will also be presented for every classification.

The performance measurements presented in the results are:

- **True Negative Rate (TNR):** Proportion of correct predictions from all the cases that are actually negative. The True Negative Rate is also known as Specificity, and is the opposite of the False Positive Rate (FPR). A high TNR is obtained by increasing the threshold. A high TNR is desirable, as the TNR tells us the probability of a randomly chosen negative sample being classified correctly, i.e. the probability of a cheating student being caught.

$$TNR = \frac{TN}{TN + FP} \tag{6.1}$$

- **False Negative Rate (FNR):** Proportion of incorrect predictions from all the cases that are actually positive. The FNR are the opposite of the True Positive Rate (TPR). Similar to TNR, a high FNR is obtained by increasing the threshold. A low FNR is desirable, as the FNR tells us the probability of a randomly chosen positive sample being classified incorrectly, i.e. the probability of a non-cheating student being falsely accused of cheating.

$$FNR = \frac{FN}{FN + TP} \tag{6.2}$$

- **AUROC:** The receiver operating characteristic (ROC) curve is created by plotting the TPR against the FPR at various threshold settings. The area under the ROC curve (AUROC) is a measure that calculates the entire area under the

ROC curve. AUROC provides a way to measure the quality of a classification model across all possible thresholds. The AUROC is useful to compare the overall performance of different classifiers to each other.

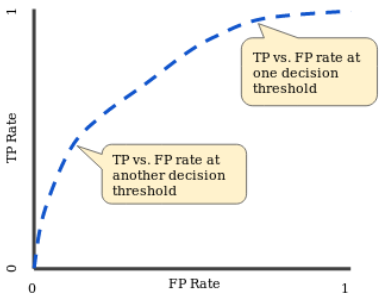

Figure 6.2: ROC curve [Don]

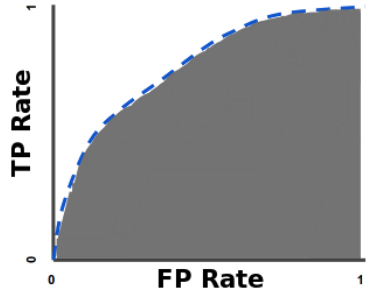

Figure 6.3: Area under the ROC curve (AUROC) [Don]

In order to determine the threshold to calculate the TNR and FNR, the optimal cut-off point needs to be found. The goal is to find the optimal trade-off between getting the TNR as high as possible and the FNR as low as possible. Following the standard for biometric performance evaluation, the cut-off points are decided from the **Equal Error Rate (EER)** in this project. The EER is the point where the proportion of errors for the positive and negative class intersect, as illustrated in Figure 6.4. The EER is often used as a measure of overall performance of a biometric system, as EER $\approx$ FPR $\approx$ FNR for the specific threshold. The results in this project will be presented in pairs of FNR and TNR and not the EER, but the information contained in these measurements are identical, as **EER $\approx$ FNR**.

In addition to using the EER to find the optimal threshold, **the TNR when the FNR is minimized will be presented for the best classifications.** Minimizing FNR means getting FNR $= 0$, i.e. zero false accusations of cheating. By using the highest possible threshold where FNR $= 0$, the TNR for this same threshold will show the proportion of cheaters that would be caught when the system has zero false accusations. The confusion matrixes showing the TP, FP, TN, and FN will be presented for both the threshold given by the EER and the threshold for minimizing false accusations.

Traditional performance metrics like accuracy, recall, precision, and F-score will not be presented in this project, in accordance with the standard for biometric performance evaluation. These metrics can however be calculated based on the confusion matrixes presented in Chapter 7.

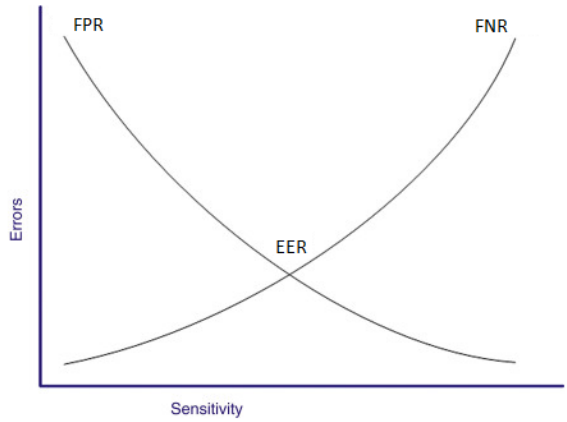

Figure 6.4: Equal error rate (EER)[CMF12]

Chapter 7
Results

This chapter presents the results from the biometric performance evaluation for the three different datasets. The False Negative Rate and True Negative Rate presented in the tables are based on the given threshold. As explained in Chapter 6, this threshold is obtained from the Equal Error Rate (EER). The results from the PAN-2013 dataset will be presented first, followed by the results from the NTNU data collection. Finally, the results from the Stewart dataset, where both stylometry and KD are described. In the section presenting the Stewart dataset, the results from the different fusion approaches are also presented.

The performance of the classifications is judged based on the best False Negative Rate (FNR) and True Negative Rate (TNR) pairs, i.e., classifications with the lowest Equal Error Rate (EER).

7.1 PAN-13 dataset

The results from the biometric performance evaluation using the PAN-13 dataset are presented in this section. This dataset does not include keystrokes and is thus only used for the stylometry part of this project. For this dataset, the best performing classifications for the word level and character level features are presented. The complete collection of the results can be found in Appendix B.

7.1.1 Preparing the data

In order to use the biometric performance evaluation, some changes were made to the dataset presented in Chapter 4. From the 30 folders representing each author in the PAN-2013 dataset, only 14 of the folders represent a problem where the answer is true. This means the known documents and the unknown document in each of these 14 folders are written by the same author. For the remaining 16 folders, the correct answer to the problems is false, i.e., the unknown documents are not written by the same author as the known documents. The known documents in the 16 false

folders are constructed from the content of the 14 true folders. For example, if folder one is a true folder and folder two is a false folder, folder two can contain the same known documents as folder one, but a different unknown document. For this reason, all the false folders from the PAN-2013 dataset are removed from the dataset in this project. This modification is necessary to perform the performance evaluation, as this method requires only true samples from each author and no duplicate samples across the different authors. Exactly four documents from the 14 true authors were used. An excerpt of how the data is structured is shown in Figure 7.1.

	user	text
0	0	This chapter gives a very broad overview of•...
1	0	In this chapter we look at the four fundament...
2	0	In Chapter 1 I defined a database to be "... an...
3	0	I should make it clear right away that to mos...
4	1	There's an old joke, well known in database c...
5	1	The main purpose of this chapter is to explai...
6	1	In order to understand the TR approach to imp...
7	1	Now (at last) I can begin to explain the TR m...
8	2	While there is a study guide (available from ...
9	2	The aim of Chapter Two is to take the simple ...

Figure 7.1: Excerpt of the dataframe consisting of texts corresponding to each author/user in the modified PAN-2013 dataset

After the modifications to the testing set, a total of 14 different authors with 4 documents each were left to perform the analysis, leading to $14 \times 4 = 56$ genuine attempts, and $14 \times 4 \times 13 = 728$ impostor attempts. The average word count for each document is around 1000 words.

7.1.2 Results from biometric performance evaluation using word-based features

The testing with word-based features was conducted using word 1-grams and 2-grams separately, as well as combined. There were only small differences in performance between the different word-based features, but the best overall results from testing

with word n-grams were obtained by combining 1-grams and 2-grams. However, the best individual FNR and TNR pair were obtained using 1-grams separately.

	Word 1-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.071	0.071	0.071	0.053	**0.053**	0.071
True Negative Rate	0.936	0.932	0.934	0.945	**0.951**	0.940
Threshold	0.523	0.530	0.503	0.504	0.545	0.503
AUROC	0.978	0.979	0.984	0.979	0.983	0.979

Table 7.1: Results from classification on the PAN-2013 dataset using word 1-grams

Table 7.1 shows that there are small differences in the results between the classifiers. Both the AUROC-score and the FNR and TNR are very similar across all classifiers. The best pair of False Negative and True Negative rates from the word-based features was obtained using NB with 1-grams on the pre-processed text. The distribution plot of genuine and impostor attempts for this specific classification are presented in Figure 7.2.

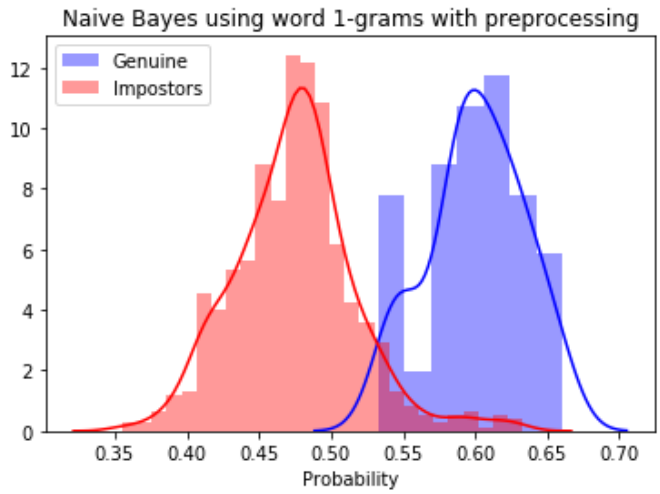

Figure 7.2: Distribution plot of genuine and impostor attempts using Naive Bayes and word 1-grams with pre-processed text

As shown in Table 7.1, by using the threshold = 0.545, NB using pre-processed text got a FNR = 0.053 and a TNR = 0.951. For an exam scenario, these numbers tell us the probability of a student being falsely accused of cheating, and the probability of a cheating student being caught, respectively. The results show that for this classification, **5.3% of non-cheaters will be accused of cheating**, while **95.1% of cheaters will get caught.** The threshold of 0.545 results in the confusion matrix presented in Figure 7.3, which shows the number of TP, TN, FP and FN.

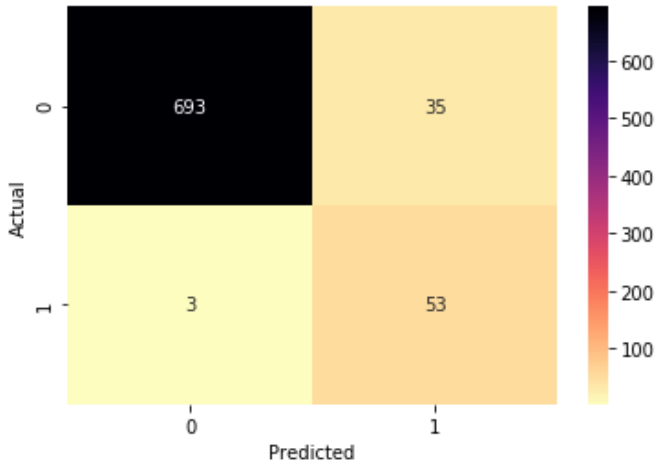

Figure 7.3: Confusion matrix for the Naive Bayes classifier using word 1-grams pre-processed text and threshold = 0.545

By decreasing the threshold for the same classification, it is possible to obtain an FNR = 0. This would remove all instances of false accusations. In a real-life application of the cheat detection system, setting a lower threshold to minimize false accusations would be the most likely choice. By setting the threshold = 0.532, the same classification would result in **0% false accusations,** while still **catching 92.5% of the cheaters.** The confusion matrix for this threshold is presented in Figure 7.4. As mentioned, the complete results from all classifications can be found in Appendix B.

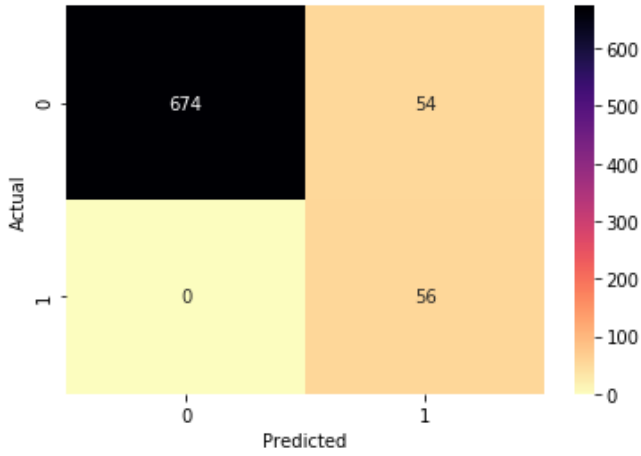

Figure 7.4: Confusion matrix for the Naive Bayes classifier using word 1-grams pre-processed text and threshold = 0.532

7.1.3 Results from biometric performance evaluation using character-based features

The testing with character n-grams was performed using character 3-grams and 4-grams separately, as well as combined. The best overall and individual results from the character-based n-grams were obtained using character 4-grams separately.

	Character 4-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.071	0.071	0.071	0.071	0.071	**0.053**
True Negative Rate	0.932	0.920	0.923	0.924	0.929	**0.942**
Threshold	0.477	0.513	0.501	0.470	0.516	0.505
AUROC	0.974	0.973	0.979	0.974	0.978	0.979

Table 7.2: Results from classification on the PAN-2013 dataset using 4-grams

Table 7.2 shows that the best pair of FNR and TNR was obtained from using LogReg on pre-processed text. With an FNR = 0.053 and a TNR = 0.942, the best performing classification using character-based features got a slightly worse result

than the best classification using word-based features. For both kinds of features, the best results were obtained when the text was pre-processed. The distribution plots for the genuine and impostor attempts are presented in Figure 7.5.

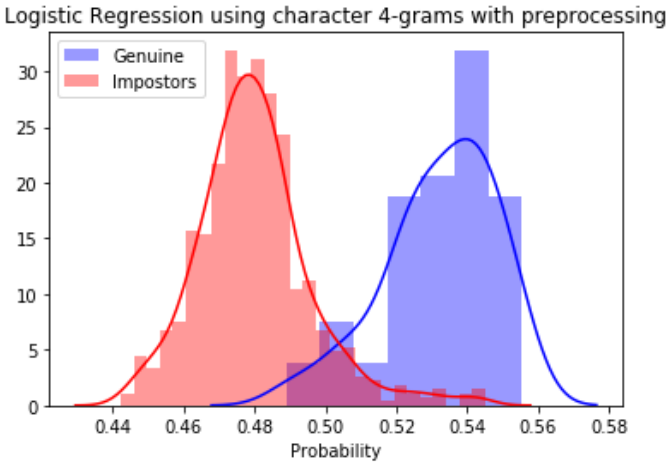

Figure 7.5: Distribution plot of genuine and impostor attempts using LogReg and character 4-grams with pre-processed text

In an exam scenario, the results from this classification would translate to **5.3% of non-cheaters being accused of cheating**, while **94.2% of cheaters would get caught**, using the threshold = 0.505. The confusion matrix for this classification is presented in Figure 7.6.

By setting the threshold = 0.488, the same classification would result in **0% false accusations** while still catching **78.3% of the cheaters**. The resulting confusion matrix from using this threshold is presented in Figure 7.7. Compared to the best word-based classification (word 1-gram using NB), the proportion of cheaters being caught when the threshold is decreased to avoid any false accusations is significantly lower for this classification.

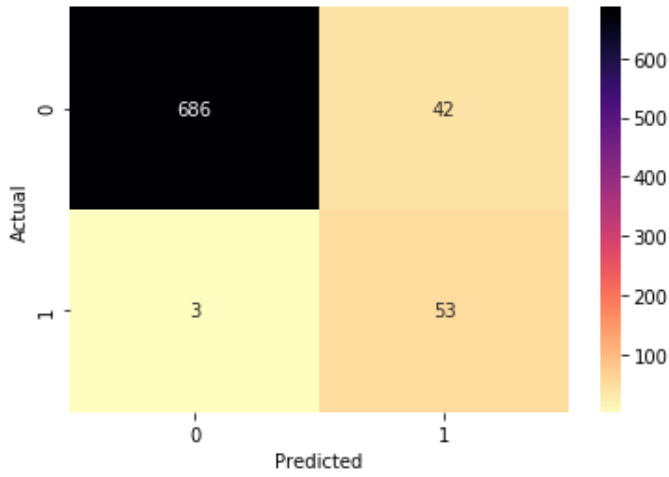

Figure 7.6: Confusion matrix for the Logistic Regression classifier using character 4-grams with pre-processed text and threshold = 0.503

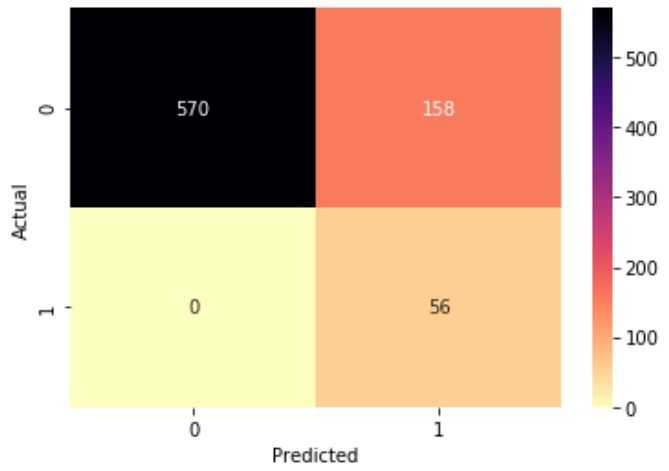

Figure 7.7: Confusion matrix for the Logistic Regression classifier using character 4-grams with pre-processed text and threshold = 0.488

7.1.4 Comparing the method to the PAN-2013 competition

In order to evaluate the performance of the method used in this project, a comparison to the results from the PAN-2013 competition is made. For comparison, the original dataset and evaluation metrics employed in the PAN-2013 competition are used. The original dataset structure is presented in Chapter 4. The rules of the competition were: *"given a set of documents by a single author and a questioned document, determine if the questioned document is written by that particular author or not."* [JS13]. The results from the biometric performance evaluation are not necessarily representative for the performance in the competition, as biometric performance evaluation uses the ground truth (true or false) about each attempt to determine the performance of a system given different thresholds. In the PAN-2013 competition, the participants did not have this information, and could not experiment with different thresholds.

The best classification from the biometric performance evaluation (Naive Bayes, word 1-gram, and pre-processed text) is used to "compete" against the PAN-2013 participants. It is an advantage to have tested different classifiers on the same data in advance, but the participants in the competition are given a training dataset before the actual competition, which level out this advantage to some degree. In order to not get an even more significant advantage, this same training dataset was used to obtain a threshold that was used on the testing dataset. This was done to not get the advantage of using the threshold (0.545) that yielded the best results in the biometric performance evaluation on the same dataset. In addition, the documents from the training dataset are used to represent the negative class in each classification. Using the documents in the testing dataset as negative samples would be inaccurate, as some of the folders contain the same documents. Using the documents from the testing dataset to create the negative class could result in duplicate samples across the positive and negative classes for some classifications.

The evaluation metrics used in the competition were [JS13]:

$$Recall = \frac{\#Correct\_answers}{\#Problems} \tag{7.1}$$

$$Precision = \frac{\#Correct\_answers}{\#Answers} \tag{7.2}$$

In the PAN-2013 competition, it was possible to leave problems unanswered. In case all the problems were answered, the recall and precision scores used in the competition were identical. The ranking in the competition was based on the best F_1-score, which is the harmonic mean between recall and precision. The AUROC

was also measured for the submissions that included a score for each answer. The winner of the competition was Seidman [Sei13]. Seidman won the competition based on all the languages in the corpus and got the best F_1-score for the English corpus. The F_1-scores, precision and recall for the submission on the English corpus are presented in Figure 7.8, and the AUROC-scores are presented in Figure 7.9.

Table 2. Results on the English part of the evaluation corpus.

Submission	F_1	Precision	Recall
Seidman [31]	0.800	0.800	0.800
Veenman&Li [35]	0.800	0.800	0.800
Layton *et al.* [25]	0.767	0.767	0.767
Moreau&Vogel [27]	0.767	0.767	0.767
Jankowska *et al.* [13]	0.733	0.733	0.733
Vilariño *et al.* [36]	0.733	0.733	0.733
Halvani *et al.* [12]	0.700	0.700	0.700
Feng&Hirst [7]	0.700	0.700	0.700
Ghaeini [9]	0.691	0.760	0.633
Petmanson [29]	0.667	0.667	0.667
Bobicev [2]	0.644	0.655	0.633
Sorin	0.633	0.633	0.633
van Dam [5]	0.600	0.600	0.600
Jayapal&Goswami [14]	0.600	0.600	0.600
Kern [19]	0.533	0.533	0.533
BASELINE	0.500	0.500	0.500
Vartapetiance&Gillam [34]	0.500	0.500	0.500
Ledesma *et al.* [26]	0.467	0.467	0.467
Grozea	0.400	0.400	0.400

Figure 7.8: The F_1-scores, recall and precision for the English corpus in the PAN-2013 competition

Table 5. Evaluation of real scores (AUC) for the whole corpus and per language.

Rank	Submission	Overall	English	Greek	Spanish
1	Jankowska, *et al.* [13]	**0.777**	**0.842**	0.711	0.804
2	Seidman [31]	0.735	0.792	**0.824**	0.583
3	Ghaeini [9]	0.729	0.837	0.527	**0.926**
4	Feng&Hirst [7]	0.697	0.750	0.580	0.772
5	Petmanson [29]	0.651	0.672	0.513	0.788
6	Bobicev [2]	0.642	0.585	0.667	0.654
7	Grozea	0.552	0.342	0.642	0.689
	BASELINE	0.500	0.500	0.500	0.500
8	Kern [19]	0.426	0.384	0.502	0.372
9	Layton *et al.* [25]	0.388	0.277	0.456	0.429

Figure 7.9: The AUROC-scores for the whole corpus per language in the PAN-2013 competition

As the method used in this project did not leave any problems unanswered, the recall, precision, and F_1-score was identical. The method got a recall, precision and **F_1-score = 0.833**. This score would have put the method created for this project in first place in the competition. The method also obtained a **AUROC = 0.933**, which is better than all of the submissions that included a probability-score for the classifications.

The winner of the competition, Seidman [Sei13], used a modified version of the Impostor method, as explained in related work in Chapter 2. The method created for this project and the method developed by Seidman are similar in that both use external documents for comparison (extrinsic method) and TF-IDF unigrams as features, but differ a lot in the way the comparison is made. The method created by Seidman uses the min-max similarity function to calculate the similarity score for individual comparisons and uses the aggregated results to classify the unknown documents, while the method used in this project have an entirely different approach by utilizing machine learning algorithms.

7.2 Dataset from the Data Collection Experiment

This section presents the results from the analysis of the keystroke dataset collected from the Data Collection Experiment. The analysis was performed with three different feature sets; the first set consisting of only duration features, the second set with only latency features, and the last feature set comprised of both duration- and latency features. For every feature set, three different machine learning classifiers were used; the Support Vector Machine (SVM), Logistic Regression (LogReg) and Naive-Bayes (NB). The best results are presented and highlighted in this section. The complete collection of the results using this dataset can be found in Appendix C.

7.2.1 Preparing the data

Although 40 students participated in the data collection experiment, only 23 of the participants completed all of the 11 questions. After filtering the data, all but one of the keystroke feature pairs depicted in Figure 5.3 and Figure 5.4 were extracted. The feature pair left out of this analysis was the mean - and standard deviation values for **punctuation**. The punctuation feature was based on the use of periods, commas, and hyphens. The keystroke capture software used in the experiment did not record the use of these keys during the data collection. Every sample averaged about 942 keystrokes each.

The extracted values were structured in a new Pandas dataframe, where every row in the dataframe represented the features extracted for every session and its corresponding user, as illustrated in Figure 7.10.

	user	session	mean_A	std_A	mean_B	std_B	mean_C	std_C	mean_D	std_D	...
0	2	1	89.103448	10.015505	81.666667	9.165151	86.750000	13.876894	88.315789	16.003655	...
1	2	2	92.615385	15.961492	74.666667	11.707547	94.333333	31.113672	83.250000	28.667054	...
2	2	4	81.952381	13.488843	76.833333	8.768055	87.117647	23.042470	83.500000	19.077405	...
3	2	5	79.333333	14.816044	76.333333	8.504901	90.545455	18.123817	81.750000	5.257647	...
4	2	6	93.125000	16.187909	81.571429	9.880235	86.428571	15.006226	84.086957	15.045123	...
...	...	...	...	...	...	...	...	...	...	...	...
248	37	8	122.960000	24.717538	105.909091	29.931437	105.533333	18.321988	121.100000	30.344137	...
249	37	9	115.500000	31.194031	108.000000	30.408332	118.214286	25.641918	138.250000	28.566112	...
250	37	11	123.433333	35.708503	108.538462	25.095203	113.809524	30.635958	125.913043	29.901221	...
251	37	12	124.610169	30.858977	89.500000	9.146948	119.480000	23.876627	116.200000	25.963115	...
252	37	13	118.135135	34.418983	89.000000	31.151244	110.360000	28.816778	111.363636	23.102784	...

Figure 7.10: Dataframe consisting of the features for every session per user from the Data Collection Experiment.

With 23 users having 11 sessions each, the rows amounted to 253; where each row had 100 columns of features, corresponding to the 30 duration feature pairs (with the punctuation feature pair left out) and the 20 latency pairs from figures 5.3 and 5.4. As explained in Chapter 5, some of the features representing key presses and key transitions are less common than others. For example, the frequency of key **v** is lower than the frequency of key **a**, or the transition from key **a** to key **t** has less occurrences than the transition from key **t** to key **h**. On a few occasions, the mean - and standard deviation values corresponding to these less common key-press occurrences were registered as *None*, simply because these key presses did not occur enough to calculate the values. This would obstruct the classification of the data samples, as the *scikit-learn*-classifiers can not perform on datasets that have empty values. To bypass this obstacle, the None values were replaced with the mean of the values corresponding to the same class and the same feature column. For example, if a session from a user's template is in class 1 and is missing a value in the column for "mean_V," then that None value would be replaced with the mean value of that entire column for class 1. The same procedure would happen if the session with the None value belonged to class 0, only that the mean from that column for class 0 would take the None value's place instead.

7.2.2 Results from biometric performance evlaluation

Table 7.3 and Table 7.4 shows the results from using the three different feature sets.

Keystroke dynamics from the Data Collection Experiment						
	duration features			latency features		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.040	0.067	0.032	0.130	0.209	0.115
True Negative Rate	0.960	0.933	0.972	0.873	0.789	0.889
Threshold	0.551	0.561	0.758	0.543	0.510	0.650
AUROC	0.992	0.976	0.991	0.947	0.871	0.954

Table 7.3: Results from classification using duration- and latency features separately.

Combined features			
	SVM	NB	LogReg
False Negative Rate	**0.028**	0.087	0.075
True Negative Rate	**0.971**	0.913	0.927
Threshold	0.570	0.617	0.656
AUROC	0.995	0.970	0.979

Table 7.4: Results from classification using a combination of duration- and latency features.

By inspecting Table 7.3 and 7.4, we can see that using the combined set of features with SVM gives the best results. The distribution plot for this classifications is presented in Figure 7.11. This specific approach achieves an FNR of 0.028 and a TNR of 0.971 with a threshold of 0.570. Translated into a real exam scenario, this would mean that **2.8% of the non-cheaters would be wrongfully accused, and 97.1% of the cheaters being caught.** The confusion matrix belonging to this classification can be seen in Figure 7.12.

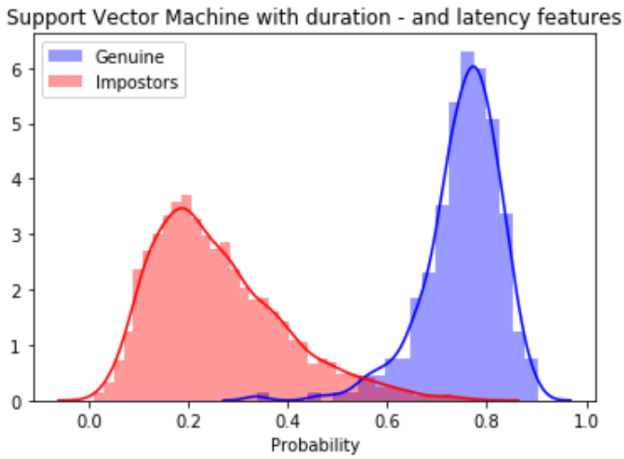

Figure 7.11: Distribution plot of genuine and impostor attempts using a combination of duration - and latency features and SVM for classification.

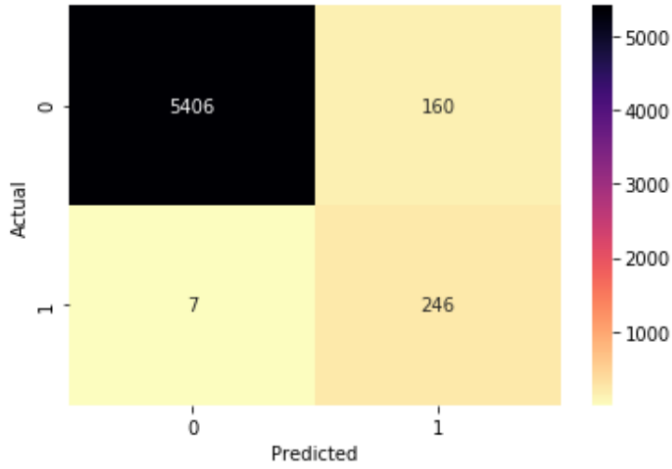

Figure 7.12: Confusion matrix for the SVM classifier using a combination of duration- and latency features with threshold = 0.570.

Testing for avoiding the wrongful accusations while still detecting impostor cases was conducted. By investigating the distribution plot in Figure 7.11 and adjusting the threshold to 0.337, the system managed to **catch 71.9% of the cheaters** without

wrongfully accusing any non-cheaters. The distribution of these classifications is illustrated by the confusion matrix in Figure 7.13

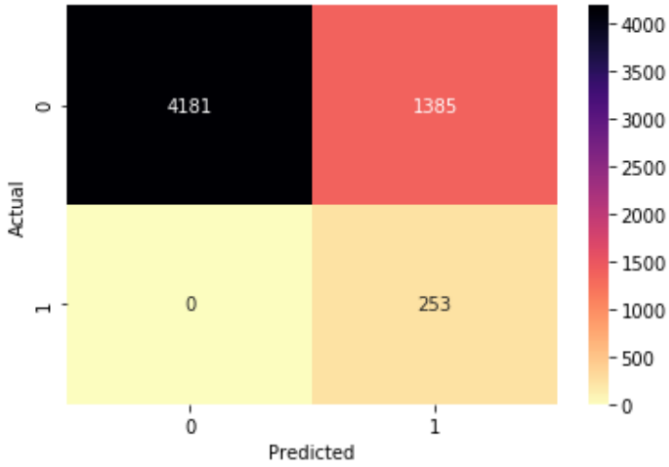

Figure 7.13: Confusion matrix for the Support Vector Machine classifier using a combination of duration- and latency features with threshold = 0.337.

7.3 Stewart dataset

This section presents the results from both the stylometry and keystroke dynamics (KD) on the Stewart dataset, as well as the two different methods for combining the results of stylometry and KD. For both stylometry and KD, the biometric performance evaluation is performed using two approaches: Single sessions and merging 5 sessions. The best results from both of these approaches are presented in this section. The complete collection of the results from the stylometry part can be found in Appendix D, while the complete collection of the results from the keystroke dynamics part can be found in Appendix E.

7.3.1 Preparing the data

Due to problems in the data collection, the original dataset was incomplete. The stylometry data came as 30 users with 40 texts each. However, the keystroke data came with 43 users, with session counts varying from 10, 20, 30, and 40. As the data was meant to represent 30 individual users, this meant that some users had participated in the Stewart data collection under several user IDs. In order to make the data usable for this project, users with less than 30 sessions were removed. This was necessary in order to avoid representing a single user with two separate user IDs, as this would be inaccurate for the biometric performance evaluation. From the remaining user pool, there were in total 26 users that had 30 or more sessions. From these 26 users, the 20 longest answers were used. This was done to remove short sessions that did not contain sufficient information and to make results more comparable to the results from Monaco et al. [MSCT13]. After this filtering, a total of 26 users with 20 different sessions each were left to perform the analysis. The average word count for the sessions was 123 words, while the average keystroke count for each session was 860.

In order to investigate the difference in performance between short and longer sessions, two different versions of the dataset were created and tested with the biometric performance evaluation:

- **Version 1:** The initial sessions were used individually for testing. In this version, $26 \times 20 = 520$ genuine attempts and $26 \times 20 \times 25 = 13\,000$ impostor attempts were performed.
- **Version 2:** Sessions were merged in groups of 5. For the stylometry, this meant concatenating the texts, while the KD treated all keystrokes for the 5 merged sessions as a single session. This Version had $26 \times 4 = 104$ genuine attempts and $26 \times 4 \times 25 = 2600$ impostor attempts. The average length for the merged sessions is 615 words and 4300 keystrokes.

7.4 Results from Version 1

7.4.1 Stylometry

As for the PAN-2013 dataset, the testing was conducted using word 1-grams and 2-grams both separately and combined, in addition to character 3- and 4-grams both separately and combined. The best results from testing on Version 1 of the Stewart dataset were obtained from using word 1- and 2-grams combined. For all classifiers, combining the 1- and 2-grams resulted in better pairs of FNR and TNR than for any of the other features. The tables containing results from all tests can be found in Appendix D.

	Word 1- and 2-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	**0.215**	0.240	0.226	0.225	0.244	0.226
True Negative Rate	**0.783**	0.760	0.772	0.774	0.754	0.774
Threshold	0.488	0.502	0.498	0.484	0.501	0.498
AUROC	0.861	0.836	0.851	0.864	0.838	0.852

Table 7.5: Results from classification on Version 1 of the Stewart dataset using word 1- and 2-grams

Table 7.5 shows that the best result was obtained from using SVM on the unprocessed(raw) text. The distribution plot of genuine and impostor attempts for this classification are presented in Figure 7.14.

By using the threshold = 0.488, the classification gets an FNR = 0.215 and a TNR = 0.783. In an real exam scenario, this would translate into **21.5% of non-cheaters being accused of cheating**, while **78.3% of cheaters would get caught.** As expected, the results are significantly worse than the results from the PAN-2013 dataset, due to the average word count being only 123 compared to 1,000. The confusion matrix for this classification is presented in Figure 7.15.

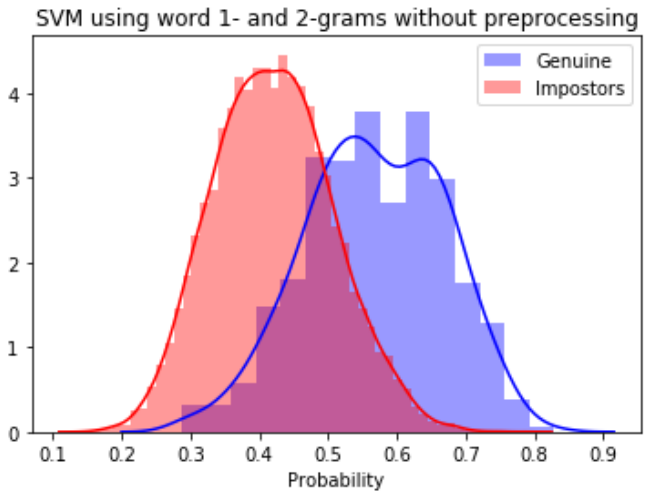

Figure 7.14: Distribution plot of genuine and impostor attempts using SVM, word 1- and 2-grams without pre-processed text

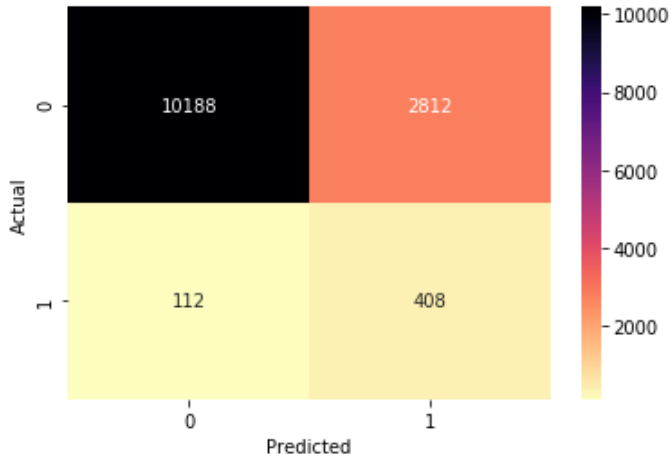

Figure 7.15: Confusion matrix for the SVM classifier using word 1- and 2-grams, unprocessed text and threshold = 0.488

By setting the threshold = 0.286, the same classification would result in **0% false accusations**, but only **5.5% of the cheaters would be caught.** The confusion matrix for this threshold is presented in Figure 7.16. The low proportion of cheaters being caught when the threshold is decreased to avoid any false accusations is

illustrated in the huge overlap of genuine and impostor attempts in Figure 7.14.

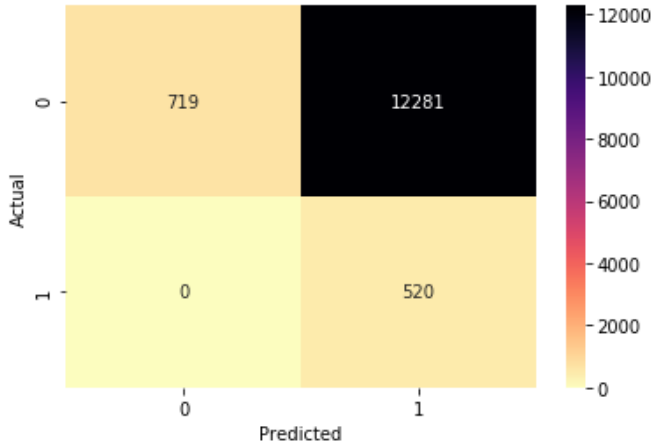

Figure 7.16: Confusion matrix for the SVM classifier using word 1- and 2-grams, unprocessed text and threshold = 0.286

7.4.2 Keystroke Dynamics

The keystroke features consisting of mean- and standard deviation values depicted in figures 5.3 and 5.4 were extracted from the filtered Stewart dataset. These values were, as done in section 7.2, structured in a Pandas dataframe with every session and its corresponding users as rows, illustrated in Figure 7.17.

user	session	sessionCounter	mean_A	std_A	mean_B	std_B	mean_C	std_C	...
2.0	1.0	1	106.258065	15.769100	92.800000	13.386560	114.823529	22.793188	...
2.0	2.0	2	108.235294	22.292177	104.000000	9.237604	89.600000	18.242807	...
2.0	3.0	3	109.461538	17.918104	86.000000	25.567837	112.000000	15.673757	...
2.0	4.0	4	103.787879	18.802855	101.666667	9.814955	104.772727	11.783491	...
2.0	5.0	5	119.074074	27.819371	114.666667	21.266562	128.000000	22.627417	...
...	...	...	...	...	...	...	...	...	...
36.0	32.0	16	83.064516	14.530280	74.666667	9.237604	79.285714	11.777703	...
36.0	33.0	17	90.578947	28.010260	70.857143	8.552360	80.575758	14.790263	...
36.0	36.0	18	83.765957	14.667641	72.000000	12.094863	83.478261	10.740195	...
36.0	39.0	19	76.509091	9.112207	76.000000	8.000000	77.333333	11.110773	...
36.0	40.0	20	75.636364	15.982057	67.555556	7.055337	72.476190	9.708857	...

Figure 7.17: Dataframe consisting of the features for every session per user in Version 1.

This dataframe consisted of 520 rows, corresponding to 20 sessions times 26 users, where every row had 102 columns of features, corresponding to the 31 duration feature pairs and 20 latency feature pairs from figures 5.3 and 5.4. The same procedure for replacing None values in the templates done for the analysis of the dataset of section 7.2, were performed here.

The analysis of the Stewart KD data was also conducted with three different feature sets with testing of LogReg, SVM and NB for each of the three sets. One feature set with only the *duration* features, one consisting of only the *latency* features, and one set comprised of both duration - and latency features. Table 7.6 and Table 7.7 shows the results from all three feature sets. The distribution plots corresponding to the tables can be found in Appendix E.

	Stewart keystroke data, no concatenation					
	duration features			latency features		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.033	0.188	**0.017**	0.096	0.186	0.082
True Negative Rate	0.967	0.812	**0.984**	0.907	0.814	0.918
Threshold	0.588	0.607	0.770	0.554	0.511	0.633
AUROC	0.995	0.883	0.998	0.964	0.894	0.973

Table 7.6: Results from classification using duration- and latency features separately.

	Combined features		
	SVM	NB	LogReg
False Negative Rate	0.029	0.127	0.0654
True Negative Rate	0.971	0.872	0.934
Threshold	0.597	0.640	0.649
AUROC	0.997	0.935	0.982

Table 7.7: Results from classification using a combination of duration- and latency features.

Table 7.6 shows that the best results came from the feature set where only the duration features were used, with the feature set using the combined duration- and latency features coming on a close second. The tables show that using the latency

feature set gives the weakest results. The classifier giving the best results from the duration feature set was LogReg. Figure 7.18 depicts the distribution plot of genuine and impostor attempts for this classification.

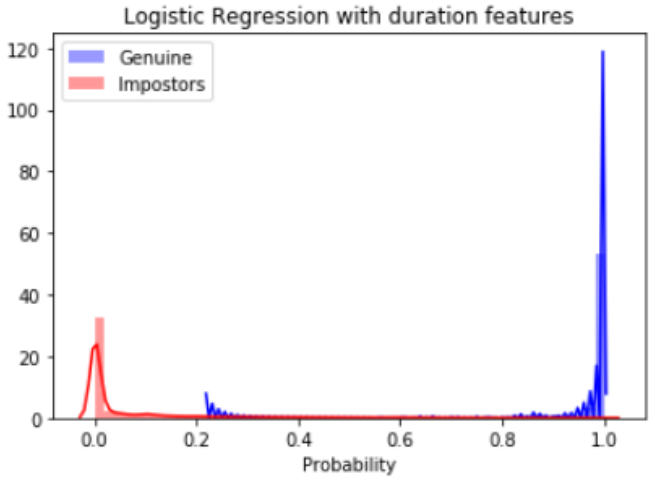

Figure 7.18: Distribution plot of genuine and impostor attempts using duration features and LogReg for classification, Version 1.

This classification achieves an FNR of 0.017 and a TNR of 0.984 when the threshold is set as 0.770. This would translate into **1.7% of non-cheaters being accused and 98.4% of cheaters being caught**, in a real exam scenario. The confusion matrix belonging to this classification can be seen in Figure 7.19.

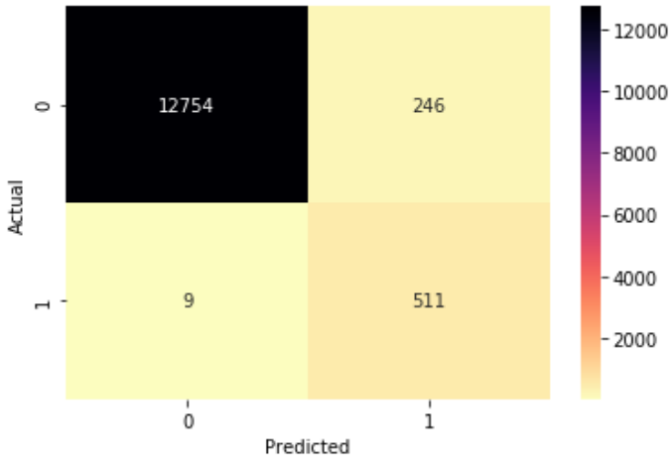

Figure 7.19: Confusion matrix representing the results from the LogReg classifier using duration features and threshold = 0.77, Version 1.

In order to minimize the accusations of non-cheaters while still detecting impostor cases, the threshold is adjusted to the highest possible value that still achieve FNR = 0. Considering the LogReg classification from the plot in Figure 7.18, the threshold can be moved to 0.222. Due to low overlap, the system is still able to catch about **86,1 % of the cheaters** without wrongfully classifying any genuine attempts. The results of this classification is illustrated by the confusion matrix in Figure 7.20.

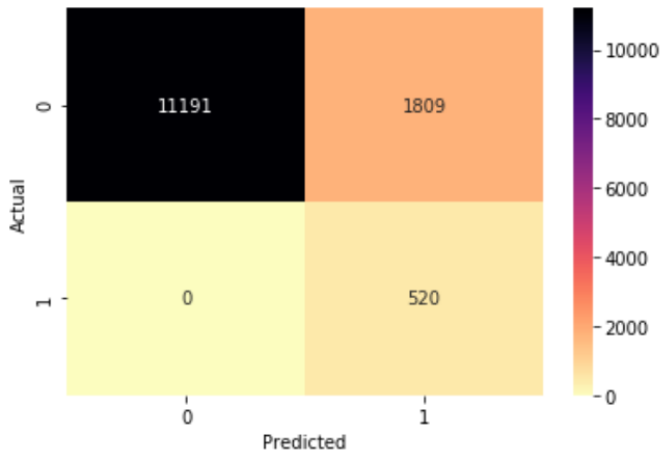

Figure 7.20: Confusion matrix representing the results from the LogReg classifier using duration features for Version 1 with threshold = 0.222

7.5 Results from Version 2

This section presents the result of the biometric performance evaluation of Version 2 of the dataset. The same users and sessions are used, but each sample is concatenated from 5 different sessions from the same user. This version of the dataset has 4 samples per student, with an average length of 615 words and about 4300 keystrokes per sample.

7.5.1 Stylometry

Out of all features, using word 2-grams obtained the best results from testing on Version 2 of the Stewart dataset. The best individual result was obtained using LogReg. The tables and distribution plots from all tests can be found in Appendix D.

	Word 2-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.163	0.153	0.144	0.182	0.144	**0.125**
True Negative Rate	0.841	0.850	0.832	0.821	0.851	**0.870**
Threshold	0.524	0.512	0.502	0.514	0.513	0.502
AUROC	0.914	0.935	0.932	0.921	0.933	0.943

Table 7.8: Results from classification on the Stewart dataset using word 2-grams with concatenated samples

Table 7.8 shows that LogReg with pre-processed text got an FNR = 0.125 and TNR = 0.870 for threshold = 0.502. In an real exam scenario, this would translate into **12.5% of non-cheaters being accused of cheating**, while **87.0% of cheaters would get caught.** This is a big improvement compared to the best result from Version 1 of the same dataset. The EER is decreased by about 10% compared to Version 1. The distribution plot of genuine and impostor attempts for this classification are presented in Figure 7.21. The confusion matrix is presented in Figure 7.22

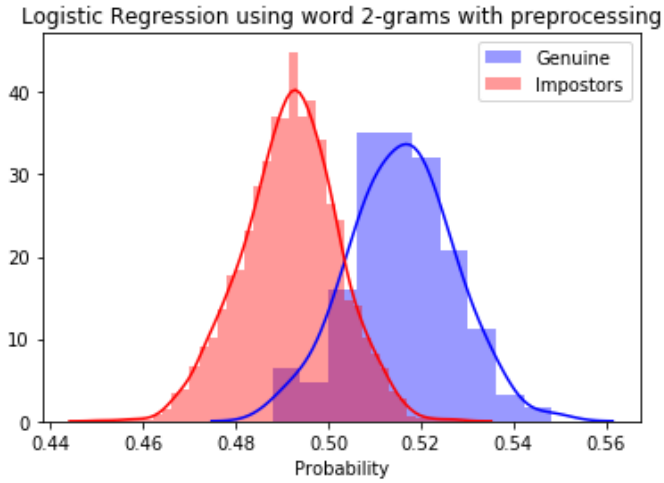

Figure 7.21: Distribution plot of genuine and impostor attempts using logreg, word 2-grams and pre-processed text

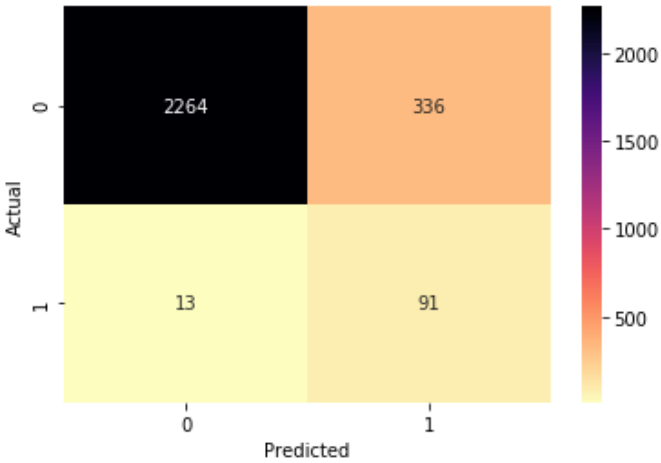

Figure 7.22: Confusion matrix for the logistic regression classifier using word 2-grams, pre-processed text and threshold = 0.502

In order get FNR = 0, the threshold = 0.488 needs to be used. For this threshold, the same classification would result in **0% of non-cheaters being falsely accused of cheating**, while **34.0% of cheaters being caught.** The confusion matrix for

this classification is presented in Figure 7.23. Compared to the best result from using stylometry on Version 1, this is a big improvement. In order to get 0% false accusations from the classification in Version 1, only 5.5% of cheaters was caught (TNR = 0.055). However, compared to the best results from the PAN dataset, the results from Version 2 of the Stewart dataset are significantly worse.

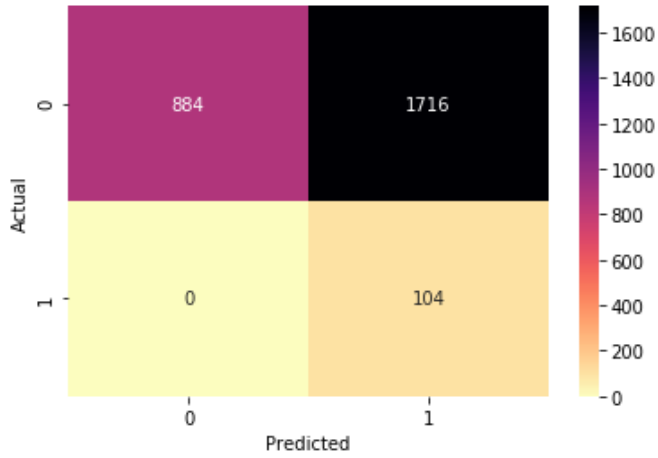

Figure 7.23: Confusion matrix for the logistic regression classifier using word 2-grams, pre-processed text and threshold = 0.488

7.5.2 Keystroke dynamics

As for Version 1, the same process of extracting the feature sets for each session was conducted. The resulting Pandas dataframe, illustrated in Figure 7.24, consisted of 104 rows, corresponding to Version 2's 26 users with 4 sessions each, along with 102 columns of duration and latency features.

	user	session	mean_A	std_A	mean_B	std_B	mean_C	std_C	mean_D	std_D	...
0	2.0	1.0	109.104478	21.018527	98.500000	21.104028	109.446429	19.442147	112.761905	21.116182	...
1	2.0	2.0	93.932292	22.520743	91.962963	19.156203	96.451613	19.417482	96.272727	21.255620	...
2	2.0	3.0	105.474453	21.309029	93.400000	19.487175	102.788732	21.390863	106.425000	18.894628	...
3	2.0	4.0	102.267241	26.052347	95.314286	22.290556	92.802469	23.701487	102.531746	23.625643	...
4	3.0	1.0	85.590551	13.994531	77.314286	10.546177	84.000000	10.728764	79.837500	12.601430	...
...	...	...	...	...	...	...	...	...	...	...	...
99	33.0	4.0	82.891892	13.535050	86.956522	18.076328	82.125000	16.546896	84.480620	15.837604	...
100	36.0	1.0	80.647059	17.008694	69.805556	10.906929	76.262500	13.063374	76.981132	13.999307	...
101	36.0	2.0	86.670996	17.662167	69.117647	10.931646	85.023256	24.445366	80.439024	14.553545	...
102	36.0	3.0	71.866337	12.146164	71.968750	9.930352	73.764706	11.071894	70.291139	11.852893	...
103	36.0	4.0	81.656863	17.737270	71.225806	9.087389	78.868852	12.378301	76.008929	12.046079	...

Figure 7.24: Dataframe consisting of the features for every session per user in Version 2.

Table 7.9 shows the results from the feature sets using duration and latency features separately, while Table 7.10 shows the results from the feature set where duration and latency features were combined.

Stewart keystroke data, five samples concatenated	duration features			latency features		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.087	0.087	**0.038**	0.135	0.192	0.135
True Negative Rate	0.918	0.912	**0.965**	0.868	0.813	0.860
Threshold	0.529	0.813	0.820	0.524	0.532	0.688
AUROC	0.976	0.968	0.995	0.950	0.896	0.942

Table 7.9: Results from classification using duration- and latency features separately, where each profile have 4 samples each, where five and five single samples are concatenated.

Combined features			
	SVM	NB	LogReg
False Negative Rate	0.048	0.125	0.106
True Negative Rate	0.951	0.876	0.894
Threshold	0.582	0.721	0.741
AUROC	0.985	0.945	0.966

Table 7.10: Results from classification using a combination of duration- and latency features on the concatenated data.

Once again, using duration features with LogReg for classification shows the best results. This approach achieves and FNR of 0.038 and a TNR of 0.965 with a threshold of 0.820, which would translate to **3.8% of non-cheaters being wrongfully accused** and **96.5% of cheaters being caught** in a contract cheating system. The distribution plot and confusion matrix belonging to this classification is illustrated in Figure 7.25 and Figure 7.26 respectively. This result is slightly worse than for Version 1, with an increase in the EER of about 2.1%.

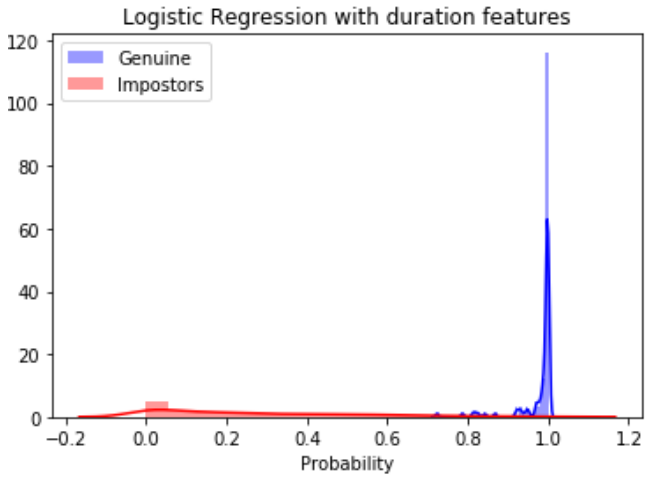

Figure 7.25: Distribution plot of genuine and impostor attempts using duration features and LogReg for Version 2.

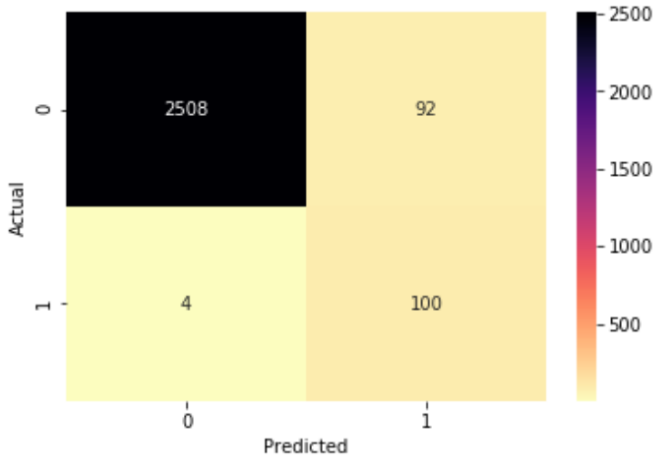

Figure 7.26: Confusion matrix representing the results from the LogReg classifier using duration features for Version 2 and threshold = 0.820

Testing for minimizing the accusation of non-cheaters and still catching as many cheaters as possible was also conducted for this version. By manipulating the threshold on the classification from Figure 7.25, the system was able to **catch 94.0% of the cheaters** without wrongfully accusing any non-cheaters, using threshold = 0.722. The confusion matrix for this classification is presented in Figure 7.27. Version 2 shows better results in detecting cheaters when the FNR = 0, compared to Version 1; where the percentage of cheaters caught without wrongful accusations was 86.1%.

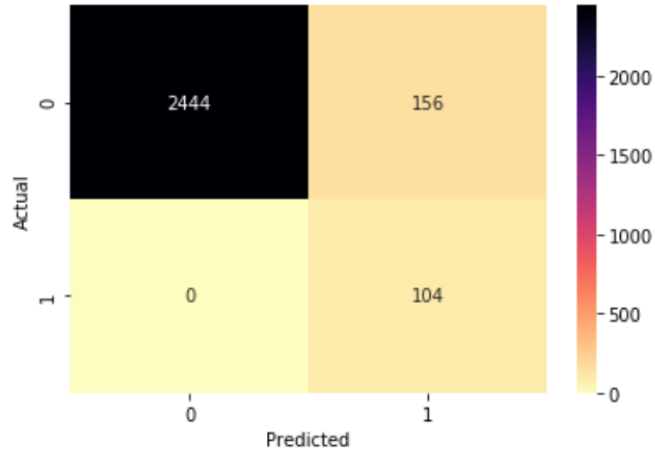

Figure 7.27: Confusion matrix representing the results from the LogReg classifier using duration features for Version 2 with threshold = 0.722.

7.6 Comparing the results to previous work

In this section, the methods and results from the paper written by Monaco et al. [MSCT13] is compared to the results of the method used in this project. Both [SMCT11] and [MSCT13] are written by Monaco et al. and use the Stewart dataset in their research, but we will compare our results only to the most recent of these papers.

Monaco et al. developed two separate systems, one using stylometry and one using keystroke dynamics. The first difference from their systems against the systems developed for this project, were the features extracted. Monaco et al. employed 239 keystroke features, including the means and standard deviations of the key-press duration - and latency timings, percentage use of specific non-letter keys, mouse clicks, and keystroke input rates. Their stylometry system employed a set of 288 linguistic features - 13 word-based, 49 character-based, and 166 syntax-based.

The classification procedure used by Monaco et al., which is the same for both their stylometry system and their KD system, differs from the one used in this project. Whereas this projects procedure is illustrated by Figure 5.1 in Chapter 5, their authentication process is based on *differences of difference vectors.* In their process, a claimed user's sample that is to be authenticated is first converted into a feature vector. A computation is done to find the differences between this feature vector and all the earlier-obtained feature vectors from the same user. With the resulting query-difference vector, a classification of either *authenticated* or *not authenticated* is done, based on comparing it to the difference vectors computed previously on the same user. In order to classify the unknown difference vectors, a k-nearest neighbor algorithm with Euclidean distance is used. This is done by using a reference set made up of the differences between all combinations of the claimed user's vectors, and the differences between the claimed user and every other user [MSCT13].

Similar to our performance evaluation method, Monaco et al. use a *leave-one-out* procedure, where for n users supplying m samples each, $m \times n$ genuine tests and $m \times n \times (n-1)$ impostor tests can be performed. The EER is used to present the results. As explained in subsection 6.2.1, the FNR and TNR pair used to present the results in this paper are based on the EER, and contains the same information. The relationship between them are EER $\approx$ FNR $\approx 1-$ TNR.

Monaco et al. performed two experiments for each system, where they used data from 30 students with 40 question answers each. To simulate longer samples, the researchers combined several answers. In their first experiment, they combined five test samples, while in the second experiment, they combined 10 samples. In the experiments of this project, the data came from the same dataset, although the 20

longest samples from 26 different users were used. The results from [MSCT13], along with the results from this project, are summarized in Table 7.11. The results are presented by the Equal Error Rate (EER).

Experiment	**Sample scheme**	**KD EER**	**Stylometry EER**
[MSCT13]	5 answers combined, 8 samples per user	0.04%	∼26%
[MSCT13]	10 answers combined, 4 samples per user	0%	∼22%
This project (Version 1)	no combined answers, 20 samples per user	∼1.70%	∼21.5%
This project (Version 2)	5 answers combined, 4 samples per user	∼3.80%	∼12.5%

Table 7.11: Results from the paper [MSCT13] compared to the results of this thesis

As the experiments performed by Monaco et. al uses different sample schemes than the experiments in this project, the results are not perfectly comparable. However, the best EER for stylometry in this project is clearly better than the best EER obtained by Monaco et. al. The best EER using stylometry in this project (∼12.5%) is obtained from Version 2, using the sample scheme of 5 answers combined and 4 samples per user. The best EER for Monaco et al. (∼22%) is obtained from using 10 answers combined and 4 samples per user. The number of samples is thus the same, but Monaco et al. used twice as many answers to construct each sample. Although using longer text samples in stylometry is usually considered an advantage, the method in this project got a 9.5% lower EER than Monaco et al. for the same number, but shorter samples. This suggests that the method created for this project is an improvement compared to Monaco et al. 's work for the stylometry part of the experiments. In addition to using a completely different verification method, a possible explanation for this can be that this project uses more simplistic features than Monaco et al. This will be further discussed in Chapter 8.

Monaco et al. obtained an EER = 0% for the sample scheme using 10 answers combined and 4 samples per user when using KD. This is a perfect result and would translate to 0% false accusations and 100% detection of cheaters for a contract cheating detection system. The best result from this project was an EER of 1.70% when using Version 1 with no answers combined and 20 samples per user. An unexpected result in this project was that the sample scheme using 20 samples

with no answers combined (Version 1) got a better EER than the sample scheme combining 5 answers with 4 samples per user (Version 2). This is the opposite effect than Monaco et al. accomplished in their experiments, where combining more answers improved the results. However, Version 2 got a significantly better TNR for the testing where FNR is minimized than Version 1. The possible explanations for these results will be discussed further in Chapter 8.

Monaco et al also conducted an experiment with a fusion of stylometry and KD. The method used for combining stylometry and KD was simply to concatenate the features from both approaches and perform the same verification method on the combined feature set. The results from this fusion are not presented in the paper, as the performance of the KD system obtained a perfect score, and the stylometry features did not improve the performance.

7.7 Fusion of stylometry and keystroke dynamics

The two methods for combining stylometry and KD presented in subsection 7.7 are tested using best results from both approaches on Version 2 of the Stewart dataset. For stylometry, the best result was obtained from using LogReg, word 2-grams, and pre-processed text, while the best result for KD was obtained from using LogReg and duration features. The fusions are tested using Version 2 of the Stewart dataset due to the overall result from both approaches being better for this version than for Version 1.

As the results from KD are far better than for stylometry, the research question RQ2: *"How can stylometry and keystroke dynamics be combined to improve the author verification of the separate systems?"* can in practice be reformulated to "Can stylometry be used to improve the author verification of keystroke dynamics?".

Thus, the goal of combining the two approaches is to investigate whether a fusion can get better results than the results obtained from using KD individually. Two separate results using KD were presented for the best classification of Version 2 of the Stewart dataset: The results based on the EER and the results when minimizing the false accusations. These results were:

Equal Error Rate: FNR = 0.038, TNR = 0.965

Minimizing false accusations: FNR = 0, TNR = 0.940

For both fusions, two different results are presented. One where the results are optimized for minimizing false accusations, i.e. getting FNR = 0, while still getting a TNR as high as possible. The other results are based on general classification rules, i.e., using generic thresholds of 0.50 and 1.00.

7.7.1 Unanimous Decision fusion

This fusion uses a simple logic where both stylometry and KD has to classify the attempt as negative for a student to be accused of cheating, as explained in Chapter 5. As the optimized fusion aims to maximize the TNR (cheaters caught) and minimize the FNR (false accusations), the thresholds for both the individual classifications are set as high as possible, while still getting a FNR = 0 for the fusion.

The optimized Unanimous Decision fusion obtained a **TNR = 0.949** while still having an **FNR = 0.** For stylometry, the threshold = 0.510 is used, while KD used the threshold = 0.915. Despite the thresholds being higher than the thresholds used for the individual classifications, zero false accusations are obtained. The reason for this is that the same attempts that are below the threshold (0.512) for stylometry, are above the threshold (0.959) for the KD approach, and vice versa. This is illustrated by plotting both scores against each other in a scatterplot[1], illustrated in Figure 7.28. The stylometry scores are plotted on the x-axis, and KD scores are plotted on the y-axis. The blue line is the threshold for stylometry, while the red line is the threshold for KD. For an attempt to be classified as an impostor, it needs to be below the red line **and** to the left of the blue line (in the bottom left square). The confusion matrix for the results is presented in Figure 7.29. The results for using the generic thresholds of 0.50 for both stylometry and KD are **TNR = 0.638** and **FNR = 0.00**. This classification is illustrated in Figure 7.30, and the confusion matrix is presented in Figure 7.31. The same rules for an attempt to be classified as an impostor apply for this classification.

The results from using the Unanimous Decision fusion shows that stylometry indeed can improve the results compared to using KD individually. The optimized results presented in this experiment are based on the best thresholds possible for the calculated scores to show the potential of the method. By decreasing the thresholds, only the most evident cases of cheating are classified as cheaters. This increases the confidence in the accusations, at the expense of accepting more cheaters. Classifying an attempt as negative when using a low threshold for both stylometry and KD, in addition to using the unanimous decision fusion, can be considered a strong indication that a student has cheated. This can be observed in the Figure 7.30, as the distance from the bottom left square in the scatterplot to the closest genuine attempt (orange dot), is substantial.

[1] https://seaborn.pydata.org/generated/seaborn.scatterplot.html

Figure 7.28: Scatterplot for the stylometry and KD scores with optimized thersholds

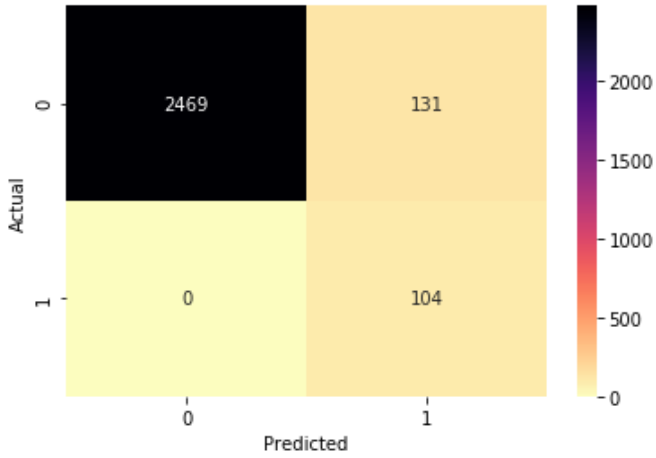

Figure 7.29: Confusion matrix of the results from the optimized unanimous decision fusion

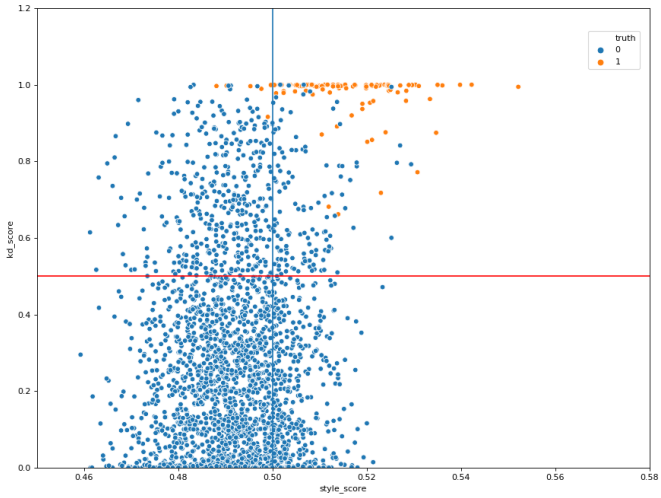

Figure 7.30: Scatterplot for the stylometry and KD scores generic thresholds

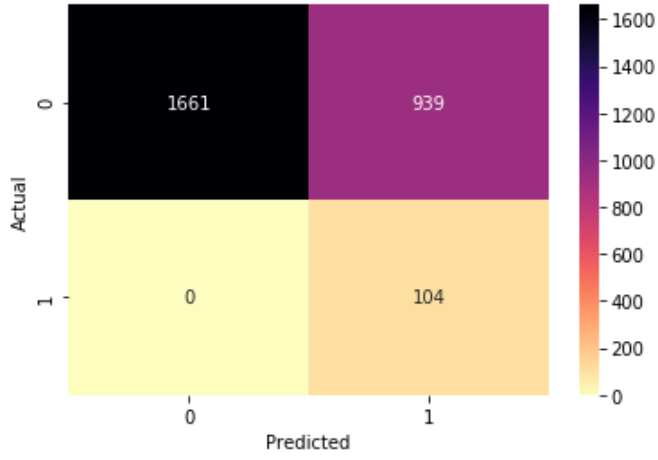

Figure 7.31: Confusion matrix of the results from the generic unanimous decision fusion

7.7.2 Aggregated Scores fusion

The Aggregated Scores fusion uses the aggregated scores from stylometry and KD to determine if a student has cheated, as explained in Chapter 5. As the range of probability-scores from KD are [0,1] while the stylometry scores range only from [0.44, 0.56], the stylometry scores were normalized to [0,1] using min-max scaling[2]. In addition to simply aggregating the scores, different weighting on the stylometry and KD scores were tested, in order to optimize the results from the fusion.

By plotting the stylometry and KD scores against each other in a scatterplot, it was possible to draw a single line to separate impostor and genuine attempts. Figure 7.32 shows the scatterplot with two different linear functions plotted as separating lines. The stylometry scores are plotted on the x-axis, and KD scores are plotted on the y-axis. The blue line illustrates the generic classification rules for this fusion: a classification where no weighting and threshold = 1.0 is used. The generic rules simply take the normalized stylometry score + the KD score for every attempt, and classify it as an impostor if the aggregated score is below 1.00 and as a genuine if the score is above 1.00. This classification results in **FNR = 0** and **TNR = 0.879.** The confusion matrix is presented in Figure 7.33.

The red line in the scatterplot illustrates an optimized classification for this fusion. For this classification, the stylometry scores are weighted with 1.34, and the threshold = 1.41 is used. This results in **FNR = 0** and **TNR = 0.974**. The confusion matrix showing the results of this classification is presented in Figure 7.34

The results show that also this fusion is able to improve the results of the individual classifiers. Of all classifications performed in this project, the optimized Aggregated Scores fusion obtained the best results when minimizing false accusations. As for the Unanimous Decision fusion, the best result (red line) presented is optimized for the specific probability-scores. However, the blue line shows that the method is still effective when using generic classifications rules. Although the number of cheaters caught decreases from 97.4% to 87.9% for the generic classification rule, the majority of cheaters still get caught. At the same time, the risk of falsely accusing a student is significantly lower. The blue line's distance to the closest genuine attempt is significant, implying that the risk of false accusations is low.

[2]https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html

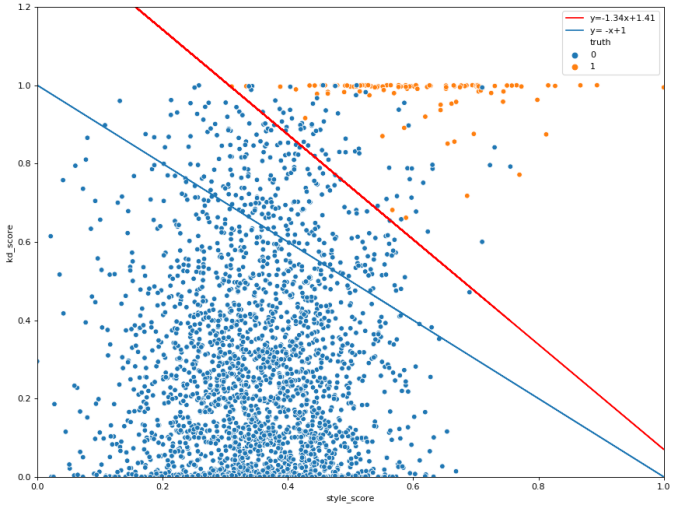

Figure 7.32: Scatterplot for the stylometry and KD scores with two different linear functions as separators

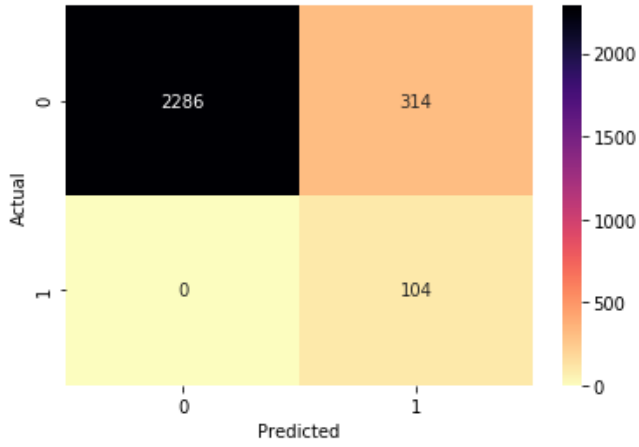

Figure 7.33: Confusion matrix of the results from the the aggregated score fusion with no weighting and threshold = 1.00

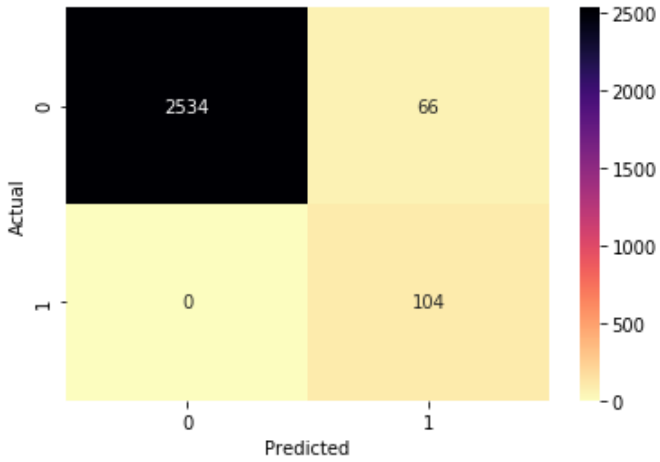

Figure 7.34: Confusion matrix of the results from the the aggregated score fusion with stylometry scores weighted 1.34 and threshold = 1.41

7.8 Summary of results

The best results from all datasets are summarized in Table 7.12. The table includes the results from the Equal Error Rate (EER) and the result where the threshold is decreased to avoid any false accusations.

	Equal Error Rate		Minimizing accusations	
	FNR	TNR	FNR	TNR
PAN-2013 dataset	0.053	0.951	0.00	0.925
NTNU data collection	0.028	0.971	0.00	0.719
Stewart Version 1 - Stylometry	0.215	0.783	0.00	0.055
Stewart Version 1 - KD	0.017	0.984	0.00	0.861
Stewart Version 2 - Stylometry	0.125	0.870	0.00	0.340
Stewart Version 2 - KD	0.038	0.965	0.00	0.940
Unanimous decision fusion	-	-	0.00	0.949
Aggregated scores fusion	-	-	0.00	0.974

Table 7.12: The best results obtained from all testing on all datasets

7.8.1 Stylometry

The PAN-2013 dataset obtained the best results from all tests using stylometry, with an EER = $\sim$ 5.3%, resulting in a FNR = 0.053 and TNR = 0.951. When false accusations were minimized, the same classification got an TNR = 0.925. For the Stewart dataset, using Version 2 with concatenated samples got a far better result than Version 1, with an EER = $\sim$12.5% compared to $\sim$21.5%. The difference was even more evident when false accusations was minimized, with a TNR = 0.340 for Version 2 compared to TNR = 0.055 for Version 1.

7.8.2 Keystroke Dynamics

Based on the EER, the best result from using KD was obtained on Version 1 of the Stewart dataset, with an EER = $\sim$1.70, resulting in a FNR = 0.017 and a TNR = 0.984. However, Version 1 got a lower TNR than Version 2 when the threshold was decreased to minimize false accusations (FNR), with a TNR = 0.861 for Version 1 and TNR = 0.940 for Version 2. The dataset from the NTNU data collection had an EER = $\sim$2.80%, which places this result in the middle of Version 1 and Version 2 of

the Stewart dataset. For the results minimizing false accusations, the NTNU dataset got the worst score of the KD experiments, with a TNR = 0.719.

7.8.3 Fusion

The two fusions in this project are based on the results from stylometry and KD for Version 2 of the Stewart dataset. The last four rows in Table 7.12 can thus be compared to each other. The Aggregated Scores fusion obtained the best result, by getting a TNR = 0.974 with no false accusations, while the Unanimous Decision fusion got a TNR = 0.949. Both fusions were able to get a better result than KD individually, confirming that a combination of stylometry and KD can improve the results. The results using generic classification rules are not presented in Table 7.12, but the Aggregated Scores and Unanimous Decision fusion obtained a TNR = 0.879 and TNR = 0.683 for these rules, respectively.

Chapter 8
Discussion

In this chapter, a discussion about the results from Chapter 7 is presented. The results from stylometry, KD, and the fusions of the two approaches are discussed separately. The discussion will highlight key findings and properties of the datasets and discuss the research questions. Finally, a general discussion about the results in light of the context and the stakeholders' goals is presented.

8.1 Stylometry

In this section, the different factors that affected the results for the stylometry part of this project will be discussed. The discussion includes limitations of the project and implications for a potential implementation of the system. Finally, a discussion about the results in light of RQ1a is presented.

8.1.1 Length of texts

As expected, the length of the texts used in classification was the single most differentiating factor affecting the results. This is clearly shown in the varying results from Version 1 and Version 2 of the Stewart dataset, which obtained a significantly better result when the answers were concatenated in groups of five, compared to being used separately. This can also explain that the best results in the stylometry part of this project came from the PAN-2013 dataset, which had an average length of 1000 words per text, compared to 615 for Version 2 of the Stewart dataset.

The number of samples used for training in a single classification is clearly of less importance than the length of each sample. This is in accordance with the information gathered in the literature review. Using several instances of texts from a single person to capture his or her writing style is generally state of the art in authorship verification using stylometry. However, the profile-based method of concatenating texts is proven to be more effective when only short texts are available. In Version 2 of the Stewart dataset, a combination of both approaches is used.

Although the results are as expected, the results confirm the importance of text length when working with stylometry and authorship verification. For a potential implementation of a contract cheating detection system using stylometry, the results in this project indicate that texts with at least 1000 words should be used.

8.1.2 Effect of pre-processing

All testing was performed both with and without pre-processing on the texts before classification, as explained in subsection 5.3. In total, 18 different combinations of features and classifiers were tested on all datasets, both with and without pre-processing.

The results show that the pre-processing had a positive effect on the results, although the difference from using unprocessed texts was, in many cases, marginal. For the PAN-2013 dataset and Version 2 of the Stewart dataset, the best results were obtained from using pre-processing. However, Version 1 of the Stewart dataset obtained its best result from using unprocessed text. When reviewing the complete set of results that are presented in Appendix A and Appendix B, it can be observed that pre-processing had an overall positive effect on the results, although many classifications also had a negative or insignificant effect of pre-processing. The effect is most noticeable for the character n-grams. Overall, the results in this project indicate that pre-processing should be considered for a real-life implementation of a contract cheating detection system. A limitation of the pre-processing performed in this project is that it is simplistic and chosen based on a paper about improving authorship attribution and not specifically for authorship verification for a contract cheating detection system. An example of pre-processing that could be considered is the filtering of topic words. For the Stewart dataset, where all participants answered the same set of questions, many words are directly related to each question's specific topic. For a system detecting contract cheating on exams, topic words related to the specific task of an exam will not necessarily provide any valuable information about a person's writing style. They could, in some cases, influence the results negatively. By removing topic words, the results in this project might be improved.

8.1.3 Features

As explained in subsection 5.3, six different variations of features were tested on all datasets. The results show that in this project, word-based n-grams outperforms character n-grams by a clear margin. The best results for all datasets were obtained from using word n-grams. For the PAN-2013 dataset, word 1-grams obtained the best results, while Version 1 and Version 2 from the Stewart dataset obtained the best results from 1- and 2-grams combined and 2-grams separately. In the appendices containing all the results, the superiority of the word-based features compared to

character-based features in this project is confirmed. However, the overall best results from the character n-grams are obtained from using character 4-grams, which is the highest n-value used in this project. Using character 3-grams, in combination with 4-grams, did not improve the performance. This can imply that using an even higher n-value could improve the results from character n-grams.

The results in this project are limited from the restricted number of features used. Using a more extensive range of features like higher n-values for the character n-grams, or other types of features like sentence structure could improve the results. However, the results from the stylometry part of this project are an improvement compared to the work done by Monaco et al. [MSCT13], despite using more simplistic features. This may imply that an increased feature pool will not necessarily improve the results. TF-IDF and n-grams were chosen in this project due to being recognized as effective in authorship verification tasks. Providing the machine learning classifiers with additional features that are less representative of a person's writing style may hurt the results. This effect is exemplified by the decreased performance when using character 3- and 4-grams combined compared to using 4-grams separately in this project.

8.1.4 Classifiers

Three different machine learning classifiers are tested in this project. The results indicate that different classifiers will perform differently relative to each other, based on the length of the texts used or the number of samples provided. This becomes evident when looking at the results of Version 1 and Version 2 of the Stewart dataset. For Version 1 of the dataset, SVM outperforms the other classifiers for all features. However, for Version 2, when texts are concatenated, both NB and LogReg outperforms SVM in all the tests. For Version 2, LogReg is, without a doubt, the best classifier of the three classifiers used in this project.

The results are particularly noteworthy given that the texts used in Version 1 and Version 2 are from the same dataset. The results indicate that for a cheat detection system, the choice of machine learning classifiers can be an important decision that should be based on the number of available texts from the students. Generally, SVM seems to outperform NB and LogReg when a high number of short texts are available, while LogReg is better for fewer and longer texts. However, the results from the PAN-2013 dataset does not reveal any distinct differences across the different classifiers. The PAN-2013 dataset contains the same number of texts per author as Version 2 of the Stewart dataset. This indicates that SVM can perform as good as NB and LogReg when using fewer texts, as long as the texts are of sufficient length.

The results in this project are limited by only using three different classifiers. It

is possible that other machine learning algorithms, or even other types of statistical methods for measuring similarity, would obtain better results.

8.1.5 Research question Q1a

Research question RQ1a is formulated: *"To what degree can stylometric analysis verify that the correct student has written the exam?"*.

Based on the results from the three different tests performed using stylometry in this project, it is difficult to provide a definite answer to the research question RQ1a. There are significant variations in the results despite the same method being used on all datasets. This indicates a low degree of generalizability. Also, none of the datasets contains data from real exams. In order to answer the research question, the limiting factors of the two datasets used in the stylometry part of this project need to be addressed. The following paragraphs discuss the datasets in the context of contract cheating detection, and what the properties of the respective datasets imply for the generalizability of the results.

PAN-2013 dataset

The PAN-2013 dataset was chosen specifically for the property of containing texts taken from textbooks in the genre of computer science. The topics of the individual texts were varied inside the genre of computer science and included topics ranging from cybercrime to Java programming. The idea was that the academic writing style of the authors of the books could resemble the style of exams about topics related to computer science.

However, two specific limitations of using the texts in this dataset to mimic exams are evident. The first limitation is that the authors behind the textbooks can be assumed to have years of experience in writing and have developed a personalized and unique writing style. For students enrolled in a university, this is more likely not the case. Most students have little experience with writing academic texts and will most likely develop their writing style and vocabulary gradually during university years. Using texts written by students in their first year in university to determine if a student has cheated in later years, may result in students being wrongly classified as cheaters, even when the method successfully can identify differences in style.

The second limitation of this dataset is that all texts in the dataset are from the same genre. While this can also be considered a strength, the results do not reflect a scenario where a student delivers an exam in a different genre than the training texts available for this student. Although students in most cases have exams related to their specific field of study, e.g., computer science students mostly write about computer science-related topics, it is not uncommon for them to participate

in courses outside their field of study. The results from the PAN-2013 dataset does not reflect how well the method would perform if all available texts from a student are from a specific genre, while the exam in question is from a completely different genre. Based on information from the literature review, it can be assumed that this would negatively impact the accuracy of the method.

Stewart dataset

Compared to the PAN-2013 dataset, the results from the Stewart dataset are more representative of this project's purpose. The main reason for this is that the Stewart dataset consists of answers from actual students enrolled in a university. In addition, all the students answered the same set of questions, which reduces the potential bias of the method, detecting differences in topics instead of personal writing style.

However, similar to the PAN-2013 dataset, all the questions are related to the same genre in this dataset. The data was collected from students attending a spreadsheet modeling course, and all the questions were related to that specific course. Similar to the PAN-2013 dataset, this limits the generalizability of the results, as it is not possible to say anything about how the method would perform in an exam scenario where the texts used for training are in a different genre than the questioned exam.

QA1

As discussed, there are several limitations associated with the results in this project given the specific datasets used. The answer to RQ1a can not be deducted directly from the results but needs to be understood from a holistic viewpoint. The result from testing on the PAN-2013 dataset showed that it was possible to detect 92.5% of cheaters while avoiding any false accusations. The result from Version 2 of the Stewart dataset, on the other hand, only detected 34.0% of the cheaters while avoiding any false accusations. For Version 1 of the Stewart dataset, the same number was only 5.5%. Although these results are strongly associated with the length of the texts, the complete set of factors influencing authorship verification using stylometry can not be fully asserted. Stylometry is not an exact science as it operates at a higher cognitive level [BMCT12], and the stylometric profile of a person is dynamic. Thus, the honest answer to RQ1a is: it depends. There is, however, strong indications for assuming that stylometry could indeed be used effectively in a contract cheating detection system, with an EER of approximately 5.3% as the best result. The results in this project show that the potential of stylometry to verify students taking exams exists, but that more research is necessary before stylometry can be utilized in a real contract cheating detection system.

8.2 Keystroke Dynamics

In this section, the results from the keystroke dynamics analysis from the DCE and the Stewart dataset are discussed. This includes interesting findings and limitations of the approach in the project. A discussion of a real-world implementation of the system is also included in this section. Finally, the system is discussed in light of *RQ1b* from Chapter 1.

8.2.1 Length of keystroke samples

As explained in chapter 7, two versions of the Stewart dataset analysis were performed; one version where there was no combination of samples and one version where the samples were combined in groups of five. Monaco et al. [MSCT13] conducted two similar experiments; one version that combined five samples and a second version with 10 samples combined. Their keystroke system improved its results from a 0.04% EER to a 0% EER by increasing the number of samples combined. Based on these results, it was expected that the EER would also decrease for the experiment using combined samples in this project; however, this was not the case. The system in this project saw an increase from an EER of 1.7% to an EER of 3.8% between Version 1 and Version 2, respectively. One could speculate that if the combination of samples in this project was done in the same manner as in [MSCT13]; that is where 10 samples are combined instead of 5, the EER would be lower in Version 2 than in Version 1. This is based on the fact that Monaco et al. saw a decrease of EER when they combined 10 samples. Another reason could be in the manner of how the keystroke samples were combined. In this project, the samples were combined such that the keystroke features were extracted from the keystrokes of sample 1 to sample 5, sample 6 to sample 10 and so on. For this approach, one row in the feature dataframe represented features from five samples. Since the Monaco et al. did not describe how the samples in their study are combined, it is possible that they had a different and more efficient process for sample combination.

Although Version 1 showed better overall performance, one noteworthy finding occurred for the testing of both Version 1 and Version 2 where the threshold was decreased to get zero false accusations. For Version 1 the system detected 86.1% of the cheaters, while for Version 2 with the combined samples, the system detected 94% of the cheaters. The difference in TNR between the two versions can be explained by the decreased amount of *outlier* probability scores. In the analysis where the templates consisted of single, non-concatenated samples, the distribution of scores for the genuine classifications was much more spread out, ranging from a probability score of about 0.222 to 1. This meant that the threshold had to be significantly adjusted to achieve FNR = 0, affecting the TNR in the process. In Version 2, these outlying probability scores for genuine classifications are more even, resulting in

a narrower range of probability scores. As a result, the threshold adjustment for Version 2 was minimal in order to achieve FNR = 0, meaning the TNR was affected less than for Version 1. These results suggest that using longer samples by combining them is useful for a system that wants to avoid false accusations of non-cheaters, although using shorter and more samples gives better overall accuracy in this project.

The results from the Stewart dataset were more promising compared to the results from the DCE. At best, the results from the DCE showed an FNR = 0.028 and a TNR = 0.971, while the best results from the Stewart dataset gave an FNR = 0.017 and a TNR = 0.984. In addition, DCE got a lower TNR than both Version 1 and Version 2 of the Stewart dataset when minimizing the FNR. This small difference in performance could also be due to the length differences of the keystroke samples between the two datasets. From the literature review, we learned that the keystroke samples containing 300 raw keystrokes are sufficient for training a model [TVC10]. After investigating the lengths of every sample used, it shows that the DCE dataset contains a higher number of samples with less than 300 keystrokes than the Stewart dataset. Although this is true only for a small subset of the samples, it could still be a reason for the difference of results between the two experiments.

8.2.2 Features

Evidently, there are choices and approaches to how features are handled that produce better results than others when working with keystroke data. In all cases, both for the results with the DCE and with the Stewart dataset, the use of duration features shows better performance than the results produced from the use of latency features. This could primarily be a cause of the number of features employed in the different feature sets. Whereas the duration feature set consisted of 31 feature pairs, the latency feature set only consisted of 21 feature pairs. However, it could also mean that *keystroke durations* works better as a behavioral biometric feature for author verification than *keystroke latencies*.

For the analysis of the dataset from the DCE, the combined feature set gave the best results, with an EER of 2.8%. For the Stewart dataset, the combination of duration - and latency features set produced almost as good results as only using the duration feature set, but **not as good.** This suggests that the quantity of features used does not necessarily give better results, but that the quality of the features; the capability of distinguishing a feature from one class to another, is what matters. It is not clear why the combined feature set produced the best results for the DCE, and not for the Stewart dataset, but a possible explanation can be how the data collection were performed. The main difference between the two data collections is the keyboards used by the participants. For the Stewart dataset, every user was provided with a Dell desktop along with a Dell keyboard. For the DCE, every participant

used their own laptops. Earlier studies have shown that keystroke dynamics can vary depending on different types of keyboards used [TVC10]. It is plausible that the quality of the latency features extracted could be affected by the type of keyboard the participants in the different data collections were using. One argument could be that a student would show a more natural typing rhythm on the keyboard they are most comfortable with or most used to, i.e., their laptops. If that would be the case, one could expect that keystroke features like latency and duration would increase in quality as behavioral metrics, since the keystroke biometric essentially operates at an automatic motor control level [BMCT12]. One could also point out that, in a remote examination scenario, students will be expected to use their own laptops. Considering this, the data extracted from the DCE would be the most realistic in a contract cheat detection situation.

To again compare the methods of this project to the approaches created by Monaco et al, it is plausible that the results could have improved if a more extensive and more varied feature set were used for training and testing. In [MSCT13], a set of 239 keystroke features were used to achieve at best an EER of 0%, in other words, a perfect performance, contrary to the system in this project that produced at best an EER of 1.7%. This project shows that both durations and latencies in keystroke dynamics work as viable features in verification problems. However, using additional features that were not employed in this project, like keystroke input rates or percentage measurements from the use of special keys, could be the addition needed to achieve even better results.

8.2.3 Classifiers

As for all the approaches in this project, three different machine learning classifiers were used. The results from the KD analysis suggests that there are classifiers more fitted than others when working with KD data. For the DCE analysis and both versions of the Stewart dataset analysis, the NB classifier has given the worst results for all the feature sets. For the tests with the duration feature set and latency feature set individually, the LogReg classifier has shown the best results on both datasets. On the other hand, while the feature set consisting of both the duration - and latency features were used, the SVM classifier performed best for both datasets. This could imply that SVM works better when more features are provided. Given that the machine learning classifiers produce different results, and that the performance of these classifiers is seemingly based on what type of feature set is used for classification, the right choice of classifier can be a significant decision in implementing a viable contract cheating detection method.

8.2.4 Research question RQ1b

RQ1b from Chapter 1 is formulated as follows: *"To what degree can keystroke dynamics verify that the correct student has written the exam?"*. Based on the research in this project, analyzing the keystrokes of individual users can be of great use in order to verify the identity of a student taking a written home exam. This is mainly based on the assumption of the data used in this project being representative of written home exams, and that the best results show only a 1.70% EER. Compared to stylometry, the results from KD can be considered more generalizable due to KD operating on a lower cognitive level [BMCT12]. One must, however, consider some aspects that could play a vital role in a contract cheat detection system using keystroke dynamics, such as how much data is used and available for training the verification model, what features to extract from the raw keystroke data, or on what type of keyboards the students conduct the exams. The research done in this project shows that such factors could affect such a system's performance and that it is not without flaws.

8.3 Fusion

Research question RQ2 is formulated: *"How can stylometry and keystroke dynamics be combined to improve the author verification of the separate systems?"*

To answer RQ2, the results from the two different methods of combining stylometry and KD presented in Chapter 7 needs to be discussed. Two different fusions were created for this project: the Unanimous Decision fusion and the Aggregated Scores fusion. For both these fusions, two different sets of rules for the classifications was tested: One where the results were optimized based on the given probability-scores to show the potential of the fusions, and one where generic classification rules were used.

Although the optimized classifications for both fusions were able to obtain a TNR = 0.949 and 0.974 while still having zero false accusations, these classification rules are not necessarily optimal for combining stylometry and KD in a real implementation of the system. The results show that fusion has the potential of improving the individual results of the separate systems. It is, however, important to take into consideration that these results are based on thresholds and weighting that is tailored to the specific probability-scores given by the stylometry and KD classifications. It is unlikely that the results would be equally good by using the same rules for an unseen dataset.

The generic classification rules, however, were not chosen based on the given probability-scores. The Unanimous Decision and Aggregated Scores fusions obtained a TNR = 0.638 and 0.879 and zero false accusations by using the generic rules,

respectively. The thresholds used for these tests are chosen independently of any prior information about the classification. It is therefore not unlikely that the results would be equally good by using the same rules on new data, given that the new data has similar properties.

Although the generic classification rules will detect a lower proportion of the cheaters, it can be observed in the figures 7.30 and 7.32 that the distance from the negative classifications to the closest genuine attempt is considerable. This indicates that fusions using generic rules can increase the confidence in the predictions of cheating students. By interpreting RQ2 in the context of a contract cheating detection system and what was learned in the problem investigating phase, it could be argued that the best way to combine stylometry and KD is to use the generic rules. From the interviews conducted in problem investigation, it became clear that to implement the use of stylometry and KD in a contract cheating detection system, it needs to consistently minimize false accusations. By using the combined results from stylometry, KD, and low thresholds, a fusion is less vulnerable to outliers in one of the two systems, and the risk of false accusations would be decreased. With the generic classification rules in the aggregated scores, fusion can detect 87.9% of cheaters, and at the same time, increase the confidence in accusations significantly. Considering the risk of being expelled from the university, an 87.9% chance of being caught would most likely have a preventive effect on most students. Due to KD being overall more reliable than stylometry, other classification rules that give more weight to the KD results could also be considered. There are endless possible combinations of thresholds and weighting that can be applied for the two fusions presented in this project. By doing more research, other classification rules that further increase the confidence in predictions of cheating, while not necessarily decreasing the number of cheaters caught, can be found.

Research question RQ2 can thus be answered in two ways. Stylometry and KD can be combined in order to increase the proportion of cheaters caught. Still, it can also be combined to decrease the probability of false accusations and increase the confidence in negative predictions, i.e., accusations of cheating. From the two fusions investigated in this project, the Aggregated Scores fusion seems to perform best. However, this fusion is more vulnerable to outliers, as a very high or low score in only one of the approaches will affect the combined decision more. A limitation to the answer of RQ2 is that only two different fusions were tested in this project. Other methods using more advanced decision rules could combine stylometry and KD in an even more efficient way.

8.4 Context and stakeholders' goals

The results show that the methods used in this project have the potential of detecting cheating cases. However, several factors come in to play when a real-life implementation of such a system is brought up. The question that will be discussed in this section is: could the methods from this project be used as a real contract cheating detection method?

Accusing a student of cheating is a severe action. The consequences for cheating are high and ranges from an annulled exam to the possibility of being expelled from the university. For this reason, the administrations of universities have a cautious approach when a student is suspected of cheating on an exam and requires substantial evidence to proceed with a cheating case. The email interview with the person working directly with the legal processing of cheating cases revealed that it is probably unrealistic to use any form of behavioral biometrics as the primary evidence against a cheating student. Although the results from this project indicate that stylometry, KD, and especially the proposed fusions can predict cheaters with relatively high confidence, there are strict requirements for concrete evidence. Expelling a student based on cheating requires evidence that is valid in a potential lawsuit, as the student has the right to sue the university if he or she does not agree with a decision. As explained in Chapter 2, Norwegian law requires the universities themselves to cover the legal costs in such cases, which gives reason to have a cautious approach unless the available evidence is very strong.

Due to the probability-score provided by the methods in this project, a real implementation could be used in a similar manner as plagiarism detection. The semi-structured interview revealed that the plagiarism detection used in Norwegian universities today produces a score that will decide whether someone in the administration will investigate the case further. The same process could be applied using the scores from the stylometry and KD methods. As the scores will not be strong enough evidence in itself, a student that is flagged by the system could, e.g., be called in for an interview with someone in the administration, where he or she will be confronted with the suspicion. Based on the interview, a student might admit to cheating, or the administration can decide to pursue the case further by looking for other evidence. The probability-scores can then be used in addition to other types of evidence, e.g., in a case including a witness and word against word.

In the motivation for this project, several media reports are mentioned that suggest the problem of contract cheating being an increasing problem. A reason for this might be that students know that there is no way for universities to detect this type of cheating, while the use of plagiarism control is common knowledge. Implementing a system designed to detect contract cheating might have a preventive

effect on students looking to cheat. The ultimate goal for universities is not to catch as many cheaters as possible, but to reduce the amount of cheating. By implementing this system, students might think twice before they pay a person online or a fellow student to write his or her exam, knowing that there is a possibility of being caught.

For a system using stylometry and KD to be implemented, the first thing that needs to be in place are sufficient data from the students. The collection of text and keystroke data must be carried out in a controlled environment to ensure the correctness of the data. A possible solution could be to arrange mandatory data collections that will collect data used as templates for the students. The collection of these data raises some legal concerns related to privacy, but as mentioned in the scope of this project, this will not be further discussed. The semi-structured interview did, however, indicate that the collection of these data would not necessarily be a privacy issue, as long as the data was anonymized.

In addition to a data collection, a necessary factor for realizing the KD component of such a system are the implementation of a keystroke capture software in Inspera or similar e-assessment providers. The semi-structured interview revealed that the technical implementation of a keystroke capture software should not be a challenge. However, the application of KD for cheat detection on home exams would require all students to write their exams directly into Inspera or similar. Depending on the format of the exam, this could be a challenge for many students. Realistically, the use of KD for cheat detection can thus only be applied on selected exams, as some exam formats require the use of online text editors like, e.g., Overleaf [1].

[1] https://www.overleaf.com/

Chapter 9
Conclusion and future work

This project has investigated the potential of using the behavioral biometrics stylometry and keystroke dynamics (KD) as tools to detect contract cheating within education. A common method for stylometry and KD was designed to verify a student's delivery of an exam. The method uses known texts and keystrokes from a student and external features to create a binary classification problem. The questioned exam is given a probability-score representing the probability that the exam was written by the intended student or not. Three different machine learning classifiers were used to predict the probability-score, namely SVM, NB and LogReg.

The performance of the method was measured using biometric performance evaluation, which is a procedure of quantifying the performance of biometric recognition systems. In total three different datasets were used to evaluate the performance of the designed method, two for stylometry and two for KD, where one of them was used for both approaches. A leave-one-out cross validation was used on all datasets.

The results from stylometry are an improvement compared to earlier work on the same datasets, which indicates the effectiveness of the designed method. The best result obtained from stylometry was an EER of approximately 5.3%, which would result in 95.1% of cheaters being caught and 5.3% of non-cheaters being falsely accused of cheating in a real contract cheating detection system. The same classification would detect 92.5% of cheaters if the threshold are reduced to avoid any false accusations of non-cheaters. The results are promising and demonstrates that stylometry has the potential of being applied in a real contract cheating detection system in the future. However, the results varied between the different datasets. The main factor influencing the results are the length of the texts available for training, but more research are necessary before stylometry can be reliably used in a real contract cheating detecting system.

For KD, the best result was an EER of approximately 1.70%, which would result in 98.4% of cheaters being caught and only 1.7% of non-cheaters being falsely accused

of cheating in a real contract cheating detection system. The best results when reducing the threshold to avoid any false accusations where 94.0%. The results from KD are more stable across all tests than for stylometry. KD are less dependent on the topic of an exam and other factors than stylometry, due to operating on a subconscious cognitive level. Implementing KD in a contract cheating detection system could be considered highly feasible, given that administrative and practical concerns allows it. Using a high number of keystrokes to represent a sample by concatenating several samples was better for minimizing false accusations, although using more and shorter samples had an overall higher accuracy. Compared to earlier work using KD, the results are promising. However, one research paper using the same dataset as used in this project was able to get 100% accuracy. This can be due to the differences in the actual verification method, but other factors such as the length of each sample and number of features used are also possible explanations.

Two methods combining the results from stylometry and KD were also designed, called the Unanimous Decision and Aggregated Scores fusion. The Aggregated Scores fusion performed best, and obtained a result of 97.4% correct predictions of cheaters, while having 0% false accusations when optimized for the probability-scores. In addition, the fusion was able to detect 87.9% of cheaters when generic classification rules were applied. The work of combining stylometry and KD shows that a fusion of both approaches is able to increase the performance over both modalities separately. It also shows that a fusion can be used to decrease the risk of false accusations.

To summarize, both stylometry and KD have the potential to be used in a real system for detecting contract cheating on home exams. The results in this project suggests that online exams that is written directly into an e-assessment provider like Inspera would benefit greatly from KD. Preferably in combination with stylometry, but when the answers are short, stylometry alone would not be very efficient. On the other hand, reports that are written over a longer period of time are not necessarily suitable for KD, as it can be difficult to capture keystrokes from the students. However, longer reports contains more text, which makes it possible to implement an efficient system using onlys stylometry.

Future work

There are several aspects that would be valuable to research further. The first thing is to explore how the method would perform when using real exams from students enrolled in a university. The results from such a study would to a higher degree be able to say something about the performance of a system in a real world implementation.

Factors such as varying genres and topics, and development of personal writing style and keystroke typing patterns over time could be important to investigate

further for real world contract cheating detecting systems. Different topics of the exams are particularly interesting for stylometry, and research using real exams would be valuable for this purpose.

In addition, exploring additional features, such as sentence structure and misspelled words for stylometry, or keystroke input rates for KD, could be valuable. Also, exploring the performance of other machine learning classifiers such as Random Forest or Long-Short Term Memory for the same method used in this project, could be a suggestion for further research. For stylometry, additional pre-processing that removes topic words of the exam can be further explored as well.

For the fusion of stylometry and KD, further research on how these can be combined even more efficiently could be conducted. Both by exploring different thresholds and weighting on the two fusion methods created in this project, but also by using more advanced forms of decision rules. Examples could be to use statistical methods like Bayesian sum and product rules and the Dempster–Shafer rule.

References

[Alp20] Ethem Alpaydin. *Introduction to machine learning*. MIT press, 2020.

[Bag15] Douglas Bagnall. Author identification using multi-headed recurrent neural networks. *arXiv preprint arXiv:1506.04891*, 2015.

[BBFGD17] Razan Bawarith, Abdullah Basuhail, Anas Fattouh, and Shehab Gamalel-Din. E-exam cheating detection system. *Int. J. Adv. Comput. Sci. Appl*, 8:176–181, 2017.

[BFKC14] Ritwik Banerjee, Song Feng, Jun Seok Kang, and Yejin Choi. Keystroke patterns as prosody in digital writings: A case study with deceptive reviews and essays. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1469–1473, 2014.

[Bha19] Raghav Bharadwaj. What is natural language processing? – a definition for business leaders. 2019.

[BHB+19] Tracey Bretag, Rowena Harper, Michael Burton, Cath Ellis, Philip Newton, Pearl Rozenberg, Sonia Saddiqui, and Karen van Haeringen. Contract cheating: A survey of australian university students. *Studies in Higher Education*, 44(11):1837–1856, 2019.

[BiB] Bi-studentar pungar ut for å sleppe å ta eksamen sjølve – kjøper seg inn hos andre. https://www.nrk.no/vestland/bi-studentar-pungar-ut-tusenvis-for-a-kjope-seg-inn-pa-heimeeksamenar-1.14817213. Accessed: 2020-02-07.

[Bio14] Precise Biometrics. Understanding biometric performance evaluation. *URL: https://precisebiometrics. com/wp-content/uploads/2014/11/White-Paper-Understanding-Biometric-Performance-Evaluation. pdf (pristupljeno: srpanj 2018.)[10]*, 2014.

[Bis06] Christopher M Bishop. *Pattern recognition and machine learning*. Springer Science+ Business Media, 2006.

[BMCT12] Ned Bakelman, John V Monaco, Sung-Hyuk Cha, and Charles C Tappert. Continual keystroke biometric authentication on short bursts of keyboard input. *Proceedings of Student-Faculty Research Day, CSIS, Pace University*, 2012.

[BTSW13] Marcelo Luiz Brocardo, Issa Traore, Sherif Saad, and Isaac Woungang. Authorship verification for short messages using stylometry. In *2013 International Conference on Computer, Information and Telecommunication Systems (CITS)*, pages 1–6. IEEE, 2013.

[BW12] Salil P Banerjee and Damon L Woodard. Biometric authentication and identification using keystroke dynamics: A survey. *Journal of Pattern Recognition Research*, 7(1):116–139, 2012.

[CJER11] GR Cluskey Jr, Craig R Ehlen, and Mitchell H Raiborn. Thwarting online exam cheating without proctor supervision. *Journal of Academic and Business Ethics*, 4(1), 2011.

[CL06] Robert Clarke and Thomas Lancaster. Eliminating the successor to plagiarism? identifying the usage of contract cheating sites. In *proceedings of 2nd international plagiarism conference*, pages 1–13. Citeseer, 2006.

[CMF12] Eric Conrad, Seth Misenar, and Joshua Feldman. *CISSP study guide*. Newnes, 2012.

[CSND20] Aparna Chirumamilla, Guttorm Sindre, and Anh Nguyen-Duc. Cheating in e-exams and paper exams: the perceptions of engineering students and teachers in norway. *Assessment & Evaluation in Higher Education*, pages 1–18, 2020.

[Dan19] Nils Folvik Danielsen. Exam cheat detection. Project report in TTM4502, Department of Information Security and Communication Technology, NTNU – Norwegian University of Science and Technology, Dec. 2019.

[Don] Niklas Donges. Classification: Roc curve and auc.

[DZ13] Yunbin Deng and Yu Zhong. Keystroke dynamics user authentication based on gaussian mixture model and deep belief nets. *ISRN Signal Processing*, 2013, 2013.

[FBC12] Song Feng, Ritwik Banerjee, and Yejin Choi. Syntactic stylometry for deception detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 171–175. Association for Computational Linguistics, 2012.

[GMA19] Souhail Guennouni, Anass Mansouri, and Ali Ahaitouf. Biometric systems and their applications. In *Eye Tracking and New Trends*. IntechOpen, 2019.

[Har18] Chris Hart. *Doing a literature review: Releasing the research imagination*. Sage, 2018.

[HBR20] Rowena Harper, Tracey Bretag, and Kiata Rundle. Detecting contract cheating: examining the role of assessment type. *Higher Education Research & Development*, pages 1–16, 2020.

[HM14] Fatma Howedi and Masnizah Mohd. Text classification for authorship attribution using naive bayes classifier with limited training data. *IISTE*, 2014.

[inn] Ingen eksamener med fysisk oppmøte på campus våren 2020|no exams with physical attendance on campus in spring 2020. https://innsida.ntnu.no/start#/feed/d41af31e-8040-39fe-96a5-c96c0e64e977/35ec22b9-82f0-3bdb-b85a-642f255e3686.

[ins] Inspera assessment product roadmap 2019. https://www.inspera.com/2019/roadmap.

[JS13] Patrick Juola and Efstathios Stamatatos. Overview of the author identification task at pan 2013. *CLEF (Working Notes)*, 1179, 2013.

[KGJ+13] M. Kam, R. Greenstadt, P. Juola, P. Brennan, S. Acharya, A. Stolerman, and A. Fridman. Decision fusion for multimodal active authentication. *IT Professional*, 15(04):29–33, jul 2013.

[KI14] Mahmoud Khonji and Youssef Iraqi. A slightly-modified gi-based author-verifier with lots of features (asgalf). *CLEF (Working Notes)*, 1180:977–983, 2014.

[KM09] Kevin S Killourhy and Roy A Maxion. Comparing anomaly-detection algorithms for keystroke dynamics. In *2009 IEEE/IFIP International Conference on Dependable Systems & Networks*, pages 125–134. IEEE, 2009.

[KS04] Moshe Koppel and Jonathan Schler. Authorship verification as a one-class classification problem. In *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML '04, page 62, New York, NY, USA, 2004. Association for Computing Machinery.

[KW14] Moshe Koppel and Yaron Winter. Determining if two documents are written by the same author. *Journal of the Association for Information Science and Technology*, 65(1):178–187, 2014.

[LBBB19] Guoqiang Li, Parisa Rezaee Borj, Loic Bergeron, and Patrick Bours. Exploring keystroke dynamics and stylometry features for gender prediction on chat data. In *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1049–1054. IEEE, 2019.

[Lin16] Lisa Lines. Ghostwriters guaranteeing grades? the quality of online ghostwriting services available to tertiary students in australia. *Teaching in Higher Education*, 21(8):889–914, 2016.

[lova] Forskrift om studier ved norges teknisk-naturvitenskapelige universitet (ntnu). https://lovdata.no/dokument/SF/forskrift/2015-12-08-1449#KAPITTEL_5.

[lovb] Lov om universiteter og høyskoler (universitets- og høyskoleloven). https://lovdata.no/dokument/NL/lov/2005-04-01-15#KAPITTEL_1-4.

[LRGL13] Charlotte Lecluze, LoÃ¯s Rigouste, Emmanuel Giguet, and Nadine Lucas. Which granularity to bootstrap a multilingual method of document alignment: character n-grams or word n-grams? *Procedia - Social and Behavioral Sciences*, 95:473–481, 2013.

[MCH18] João Moreira, Andre Carvalho, and Tomás Horvath. *A General Introduction to Data Analytics*. Wiley, 2018.

[Mis19] Abhishek Mishra. *Machine Learning in the AWS Cloud: Add Intelligence to Applications with Amazon SageMaker and Amazon Rekognition*. Wiley, 2019.

[mlma] A gentle introduction to the bag-of-words model. https://machinelearningmastery.com/gentle-introduction-bag-words-model/.

[mlmb] How to prepare text data for machine learning with scikit-learn. https://machinelearningmastery.com/prepare-text-data-machine-learning-scikit-learn/.

[MLU+16] Christian Moriarty, Christopher Lang, Margaret Usdansky, Maya Kanani, Megan Jamieson, Tricia Bertram Gallant, and Valerie George. Institutional toolkit to combat contract cheating, 2016.

[MMS99] Christopher D Manning, Christopher D Manning, and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

[MN07] Michael D. Myers and Michael Newman. *The qualitative interview in IS research: Examining the craft*. 2007.

[mor] More exams the way to beat cheats buying contract essays. http://theconversation.com/more-exams-the-way-to-beat-cheats-buying-contract-essays-32399.

[MPT+15] John V Monaco, Gonzalo Perez, Charles C Tappert, Patrick Bours, Soumik Mondal, Sudalai Rajkumar, Aythami Morales, Julian Fierrez, and Javier Ortega-Garcia. One-handed keystroke biometric identification competition. In *2015 International Conference on Biometrics (ICB)*, pages 58–64. IEEE, 2015.

[MRT12] Mohri Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning (Adaptive Computation and Machine Learning Series)*. MIT press, 2012.

[MSCT13] John V Monaco, John C Stewart, Sung-Hyuk Cha, and Charles C Tappert. Behavioral biometric verification of student identity in online course assessment and authentication of authors in literary works. In *2013 IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS)*, pages 1–8. IEEE, 2013.

[MSS17] Ilia Markov, Efstathios Stamatatos, and Grigori Sidorov. Improving cross-topic authorship attribution: The role of pre-processing. In *International Conference on Computational Linguistics and Intelligent Text Processing*, pages 289–302. Springer, 2017.

[NL16] Philip M Newton and Christopher Lang. Custom essay writers, freelancers, and other paid third parties. *Handbook of academic integrity*, pages 249–271, 2016.

[ORKS18] Julia Opgen-Rhein, Bastian Küppers, and Ulrik Schroeder. An application to discover cheating in digital exams. *IT Professional*, 2018.

[Pan] Ayush Pant. Introduction to logistic regression.

[Pla18] Barbara Plank. Predicting authorship and author traits from keystroke dynamics. In *Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media*, pages 98–104, New Orleans, Louisiana, USA, June 2018. Association for Computational Linguistics.

[Ple19] Petr Plechac. Relative contributions of shakespeare and fletcher in henry viii: An analysis based on most frequent words and most frequent rhythmic patterns. 11 2019.

[PS19] Nektaria Potha and Efstathios Stamatatos. Improved algorithms for extrinsic author verification. *Knowledge and Information Systems*, pages 1–19, 2019.

[RK11] Jonas Richiardi and Krzysztof Kryszczuk. *Biometric Systems Evaluation*, pages 117–124. Springer US, Boston, MA, 2011.

[SDV+14] Efstathios Stamatatos, Walter Daelemans, Ben Verhoeven, Martin Potthast, Benno Stein, Patrick Juola, Miguel A Sanchez-Perez, and Alberto Barrón-Cedeño. Overview of the author identification task at pan 2014. In *CLEF 2014 Evaluation Labs and Workshop Working Notes Papers, Sheffield, UK, 2014*, pages 1–21, 2014.

[SDV+15] Efstathios Stamatatos, Walter Daelemans, Ben Verhoeven, Patrick Juola, Aurelio LÃ³pez-LÃ³pez, Martin Potthast, and Benno Stein. Overview of the author identification task at pan 2015. *CLEF 2015 Evaluation Labs and Workshop â€“ Working Notes Papers*, pages 1–17, 2015.

[Sei13] Shachar Seidman. Authorship verification using the impostors method. In *CLEF 2013 Evaluation labs and workshop–Working notes papers*, pages 23–26. Citeseer, 2013.

[sen] Studentundersøkelsen: 1 av 6 har jukset på eksamen. https://universitas.no/sak/65773/studentundersokelsen-1-av-6-har-jukset-pa-eksamen/.

[SMCT11] John C Stewart, John V Monaco, Sung-Hyuk Cha, and Charles C Tappert. An investigation of keystroke and stylometry traits for authenticating online test takers. In *2011 International Joint Conference on Biometrics (IJCB)*, pages 1–7. IEEE, 2011.

[Sta09] Efstathios Stamatatos. A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, 60(3):538–556, 2009.

[svm] Chapter 2: Svm (support vector machine) theory. https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72.

[TVC10] Charles C Tappert, Mary Villani, and Sung-Hyuk Cha. Keystroke biometric identification and authentication on long-text input. In *Behavioral biometrics for human identification: Intelligent applications*, pages 342–367. IGI global, 2010.

[udi] Alle barnehager og skoler er stengt på grunn av koronavirus. https://www.udir.no/kvalitet-og-kompetanse/sikkerhet-og-beredskap/informasjon-om-koronaviruset/.

[uhl] Retningslinjer ved behandling av fusk eller forsøk på fusk til eksamen ved norges teknisk-naturvitenskapelige universitet (ntnu). https://innsida.ntnu.no/c/wiki/get_page_attachment?p_l_id=22780&nodeId=24647&title=Generelle%20lover%20og%20regler%20-%20studier&fileName=Fusk%20retningslinjer%20at%20NTNU%20vedtatt%20250619.pdf.

[urk] Urkund: The solution to plagiarism. https://www.urkund.com/about-urkund/.

[Wan09] Liang Wang. *Behavioral Biometrics for Human Identification: Intelligent Applications: Intelligent Applications*. IGI Global, 2009.

[Wie14] Roel J Wieringa. *Design science methodology for information systems and software engineering*. Springer, 2014.

[YDJP19] Jay R Young, Randall S Davies, Jeffrey L Jenkins, and Isaac Pfleger. Keystroke dynamics: establishing keyprints to verify users in online courses. *Computers in the Schools*, 36(1):48–68, 2019.

Appendix

Appendix A

The questions from the Semi-structured interview, as well as the questions and written answers from the Email interview, can be found in this appendix.

A.1 Semi-structured interview

- How is a case treated if there is cheating suspected, i.e. when a delivery is flagged as plagiarized?
  1. Is there a "plagiarism threshold" that tells you how much plagiarism is tolerated?
  2. Do you have a conversation with the student?
- In general, how serious do you consider cheating on home exams as a problem?
  1. Do you have any experience with contract cheating?
  2. Do you expect an increase in cheating today, considering a significant amount of evaluations have been moved to remote - or home examinations due to the corona outbreak?
- What counter measures do you have today against cheating on remote- or home exams?
- How realistic is the implementation of keystroke capture software in existing assessment programs, like Inspera or Safe Exam Browser?
  1. Would it be technically challenging?
  2. Would it be challenges related to privacy?
  3. Storage challenges?
- Should a cheat detection method provide a *probability score*, indicating how accurate a prediction (cheating/non-cheating) is?

- What level of accuracy must the system show in order for it to be useful?
- How important is it to incorrectly blame someone for cheating versus discovering as many cheating individuals as possible?
- In general, do you have any thoughts on what sort of requirements the cheat detection system/method should follow?
    1. Both in terms of keystrokes and stylometry
- What type of data or further analysis from a potential cheating case would you think is valuable when such a case occurs/cheating is suspected?

A.2 Email interview

The first paragraph in this section represents the introduction in the email sent.

Vi holder for øyeblikket på å skrive en masteroppgave som handler om å bruke tekstgjenkjenning og keystrokes til å detektere juks på hjemmeeksamen. Kort forklart har vi laget en metode som bruker maskinlæring til å kunne avdekke om en person har skrevet en tekst selv, eller om noen andre har skrevet det for han/henne, basert på både hvordan de skriver (f.eks ordvalg) og hvordan de taster (f.eks hvor lenge holder en person inne en tast i gjennomsnitt). Vi begynner å nærme oss slutten og har fått en del resultater, men for å drøfte resultatene hadde det vært nyttig med litt informasjon for å diskutere hvordan dette evt. kunne blitt brukt i reell eksamenssitusjon. Håper derfor du har mulighet til å besvare noen korte spørsmål.

- ***Question 1**: Om en slik metode som beskrevet skulle bli tatt i bruk, hvor treffsikkert måtte det vært?*
    - **Answer 1**: Hvis man skal basere seg på at dette systemet skal avdekke fusk så må det helt unntaksvis være feil. Spørsmålet blir også hvor lett det vil være å fange opp slike feil i ettertid og hvilke andre faktorer det kan være som støtter den tekniske gjennomgangen. F.eks flere skriveprøver. Hvordan regelverket rundt bruken utformes vil også ha betydning. Systemet kan også være aktuelt å vurdere sammen med andre faktorer. Hvis det kan være andre forhold som støtter opp om at vedkommende ikke har skrevet dette selv, f.eks en som i ettertid står frem og påstår at han skrev dette for vedkommende. Her kan det bli ord mot ord og da kan et slikt system støtte/ikke støtte en av partene.
- ***Question 2**: Hvor sikker er man vanligvis før man går videre med en juksesak?*

- **Answer 2**: Vedtak om annullering og utestenging på grunn av fusk er inngripende vedtak og alvorlig for den det gjelder. Vi må derfor kunne si at det objektivt sett foreligger fusk. Med objektivt sett menes at vi kan konstatere at det er avskrift uten kildehenvisning, at vedkommende hadde ulovlige hjelpemidler tilgjengelig, at vedkommende ikke har skrevet dette selv. Hvis det er konstatert at det objektivt sett er fusk må det vurderes om skyldkravet etter uh-loven § 4-7 er oppfylt, dvs. om studenten har opptrådt forsettlig (gjort dette med bevissthet/vilje) eller om hen har vært grovt uaktsom (har opptrådt grovt klanderverdig). Et godt eksempel på grov uaktsomhet er en student som sier vedkommende ved et uhell hadde tatt med seg lapper med pensumtekst og ikke var klar over at de lå på pulten. Uansett er det grovt uaktsomt å ikke sørge for at dette ble fjernet før eksamen. Samlet sett må det være overveiende sannsynlig at vedkommende har fusket. I praksis vil vi ikke reise sak om fusk dersom det er en del tvil.

– ***Question 3**: Hvor stor er konsekvensen av å anklage noen feilaktig for juks?*

  - **Answer 3**: Det er alvorlig. Vedkommende kan da urettmessig bli utestengt og institusjonen kan ved opphevelse av vedtaket i en domstol bli pålagt å betale erstatning for studentens økonomiske tap, hvis en utestenging har ført til forsinkelse i studiet hvilket det gjerne vil medføre.

– ***Question 4**: Hvor viktig er det med helt konkrete bevis?*

  - **Answer 4**: Det er helt avgjørende.

– ***Question 5**: Hva gjør man om man "vet" at noen har jukset, men kan ikke bevise det?*

  - **Answer 5**: At man vet må underbygges av bevis. Vitner kan være et aktuelt bevis. Det kan være flere forhold som må vurderes; dokumentbevis, forklaringer, tekniske bevis.

– ***Question 6**: Om en person er mistenkt for juks, men ingen konkrete bevis er tilgjengelig, blir vedkommende "flagget" for fremtidige eksamener på noen måte?*

  - **Answer 6**: Nei.

Appendix B

Appendix B

All results and plots from the biometric performance evaluation using the PAN-2013 dataset are contained in this Appendix.

	Word 1-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.071	0.071	0.071	0.053	0.053	0.071
True Negative Rate	0.936	0.932	0.934	0.945	0.951	0.940
Threshold	0.523	0.530	0.503	0.504	0.545	0.503
AUROC	0.978	0.979	0.984	0.979	0.983	0.979

Table B.1: Results from classification on the PAN-2013 dataset using word 1-grams

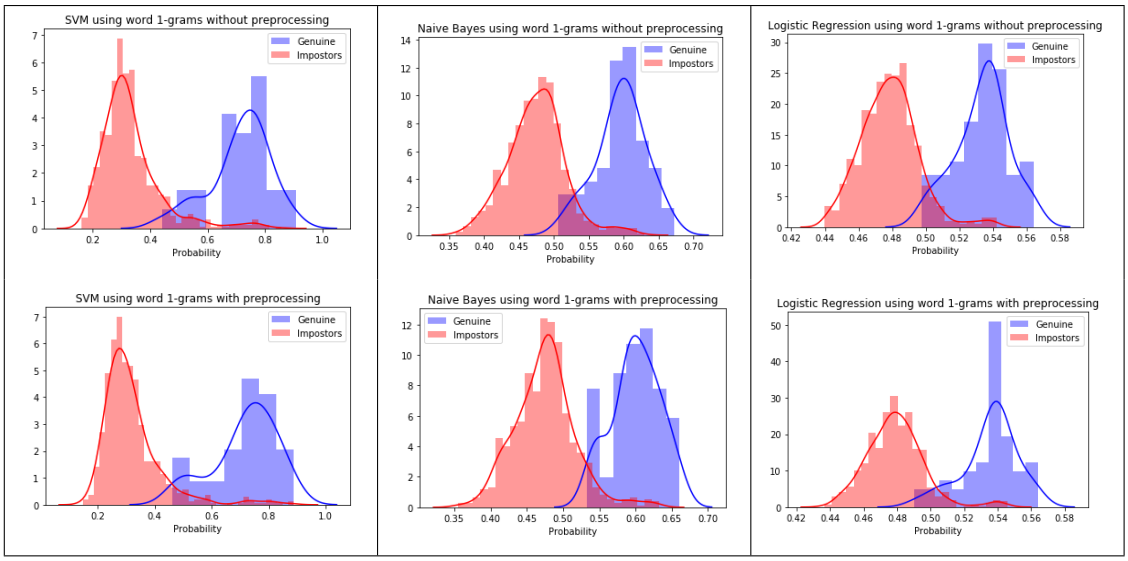

Figure B.1: The distribution plots from table B.1

	Word 2-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.053	0.053	0.071	0.071	0.071	0.071
True Negative Rate	0.949	0.942	0.928	0.943	0.929	0.934
Threshold	0.538	0.524	0.502	0.550	0.518	0.503
AUROC	0.974	0.977	0.977	0.967	0.973	0.978

Table B.2: Results from classification on the PAN-2013 dataset using word 2-grams

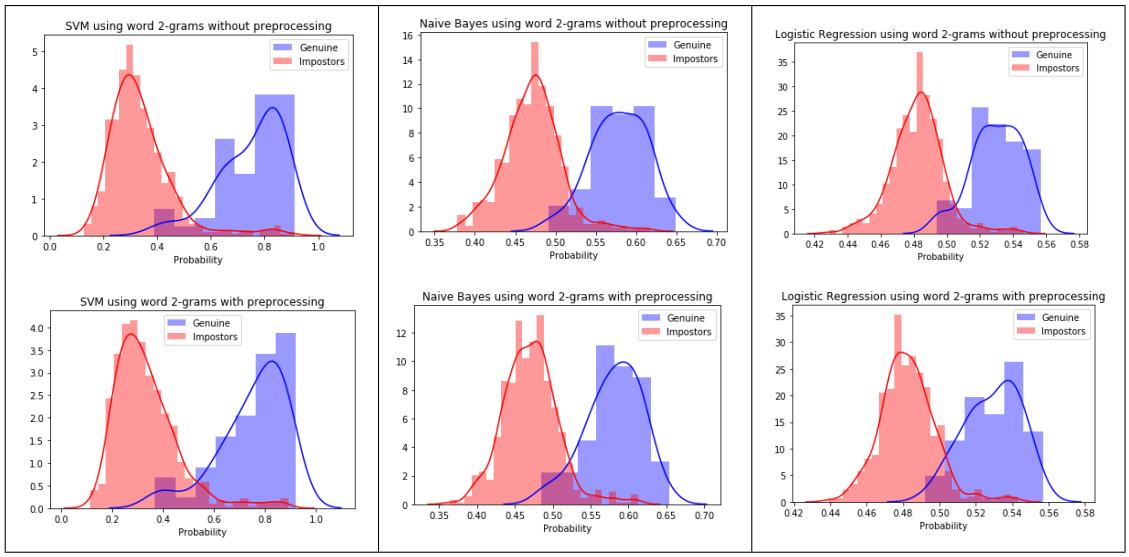

Figure B.2: The distribution plots from table B.2

	Word 1- and 2-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.071	0.053	0.071	0.071	0.053	0.053
True Negative Rate	0.935	0.938	0.940	0.928	0.946	0.947
Threshold	0.493	0.519	0.505	0.474	0.521	0.502
AUROC	0.977	0.982	0.983	0.978	0.984	0.984

Table B.3: Results from classification on the PAN-2013 dataset using word 1- and 2-grams

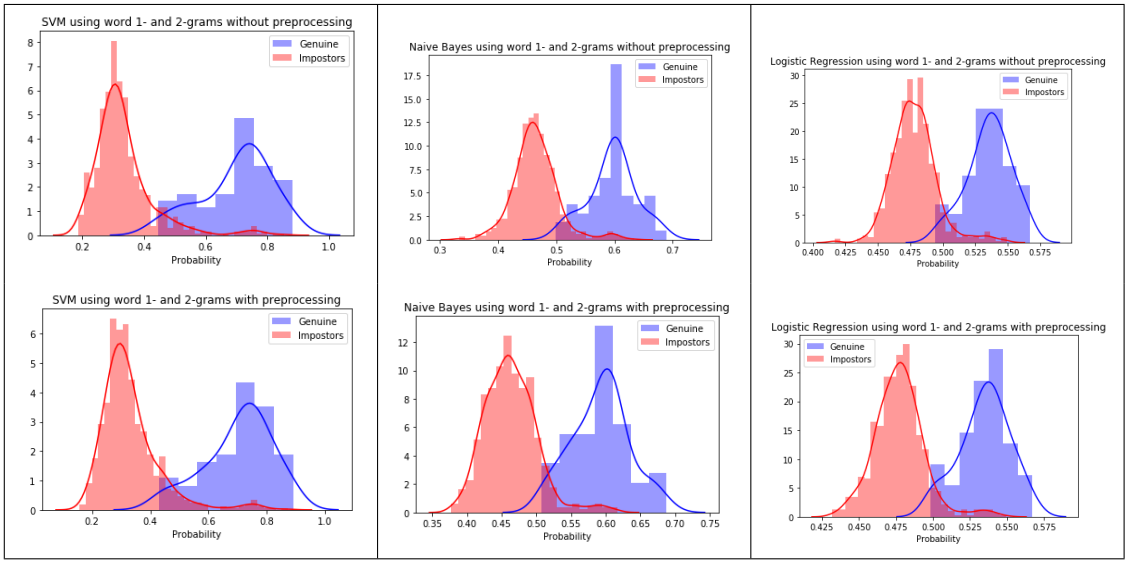

Figure B.3: The distribution plots from table B.3

	Character 3-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.089	0.071	0.089	0.071	0.071	0.089
True Negative Rate	0.912	0.938	0.902	0.925	0.932	0.917
Threshold	0.457	0.512	0.498	0.460	0.511	0.500
AUROC	0.972	0.975	0.978	0.976	0.972	0.976

Table B.4: Results from classification on the PAN-2013 dataset using word 1- and 2-grams

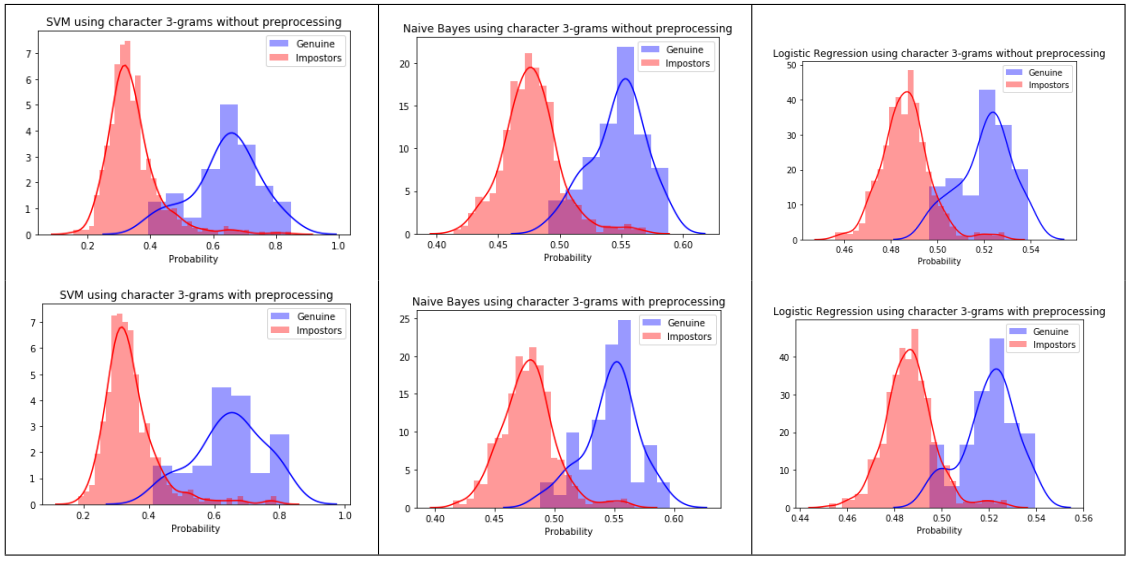

Figure B.4: The distribution plots from table B.4

	Character 4-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.071	0.071	0.071	0.071	0.071	0.053
True Negative Rate	0.932	0.920	0.923	0.924	0.929	0.942
Threshold	0.477	0.513	0.501	0.470	0.516	0.505
AUROC	0.974	0.973	0.979	0.974	0.978	0.979

Table B.5: Results from classification on the PAN-2013 dataset using character 4-grams

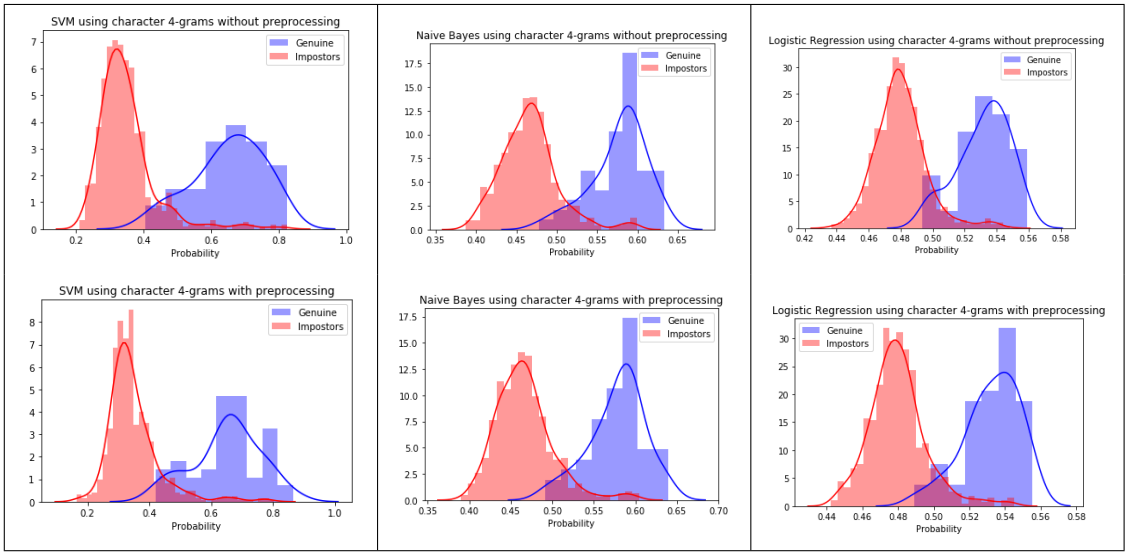

Figure B.5: The distribution plots from table B.5

	Character 3- and 4-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.089	0.071	0.089	0.071	0.089	0.089
True Negative Rate	0.906	0.918	0.912	0.928	0.910	0.907
Threshold	0.459	0.507	0.498	0.473	0.501	0.498
AUROC	0.967	0.966	0.970	0.965	0.965	0.971

Table B.6: Results from classification on the PAN-2013 dataset using character 3- and 4-grams

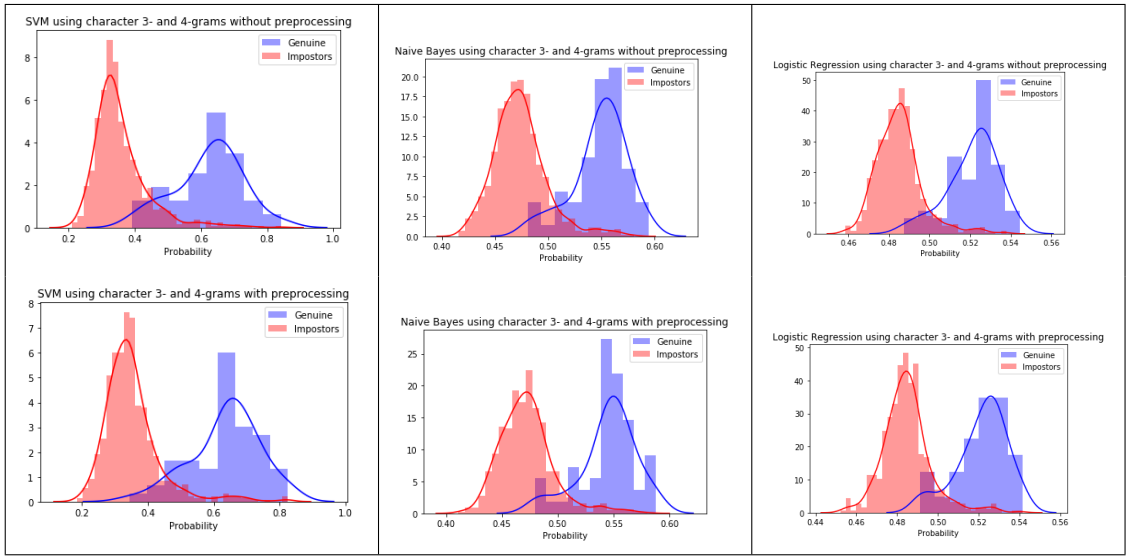

Figure B.6: The distribution plots from table B.6

Appendix

Appendix C

This appendix consists of tables and distribution plots from evaluation of the results, testing our models on the Data Collection Experiment dataset. The tables illustrate the results from testing with three different feature sets; the duration features, the latency features, and a combination of both.

Keystroke dynamics from the Data Collection Experiment						
	duration features			latency features		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.040	0.067	0.032	0.130	0.209	0.115
True Negative Rate	0.960	0.933	0.972	0.873	0.789	0.889
Threshold	0.551	0.561	0.758	0.543	0.510	0.650
AUROC	0.992	0.976	0.991	0.947	0.871	0.954

Table C.1: Results from classification using duration- and latency features separately.

Combined features			
	SVM	NB	LogReg
False Negative Rate	0.028	0.087	0.075
True Negative Rate	0.971	0.913	0.927
Threshold	0.570	0.617	0.656
AUROC	0.995	0.970	0.979

Table C.2: Results from classification using a combination of duration- and latency features.

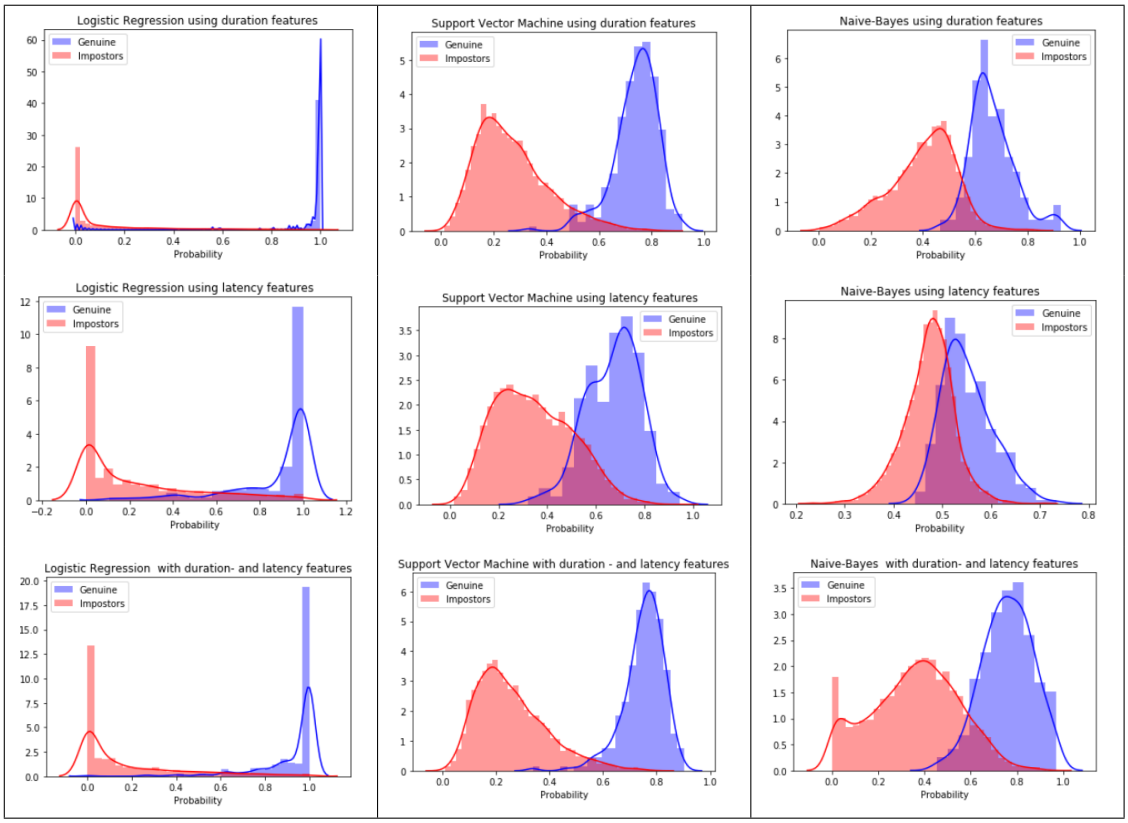

Figure C.1: The distribution plots from tables C.1 and C.2

Appendix D

Appendix D

The results from the biometric performance evaluation on the Stewart dataset using stylometry.

D.1 Version 1

	Word 1-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.228	0.267	0.238	0.230	0.263	0.236
True Negative Rate	0.769	0.731	0.759	0.767	0.737	0.763
Threshold	0.489	0.515	0.499	0.489	0.516	0.499
AUROC	0.852	0.810	0.840	0.855	0.812	0.842

Table D.1: Results from classification on the Stewart dataset using word 1-grams

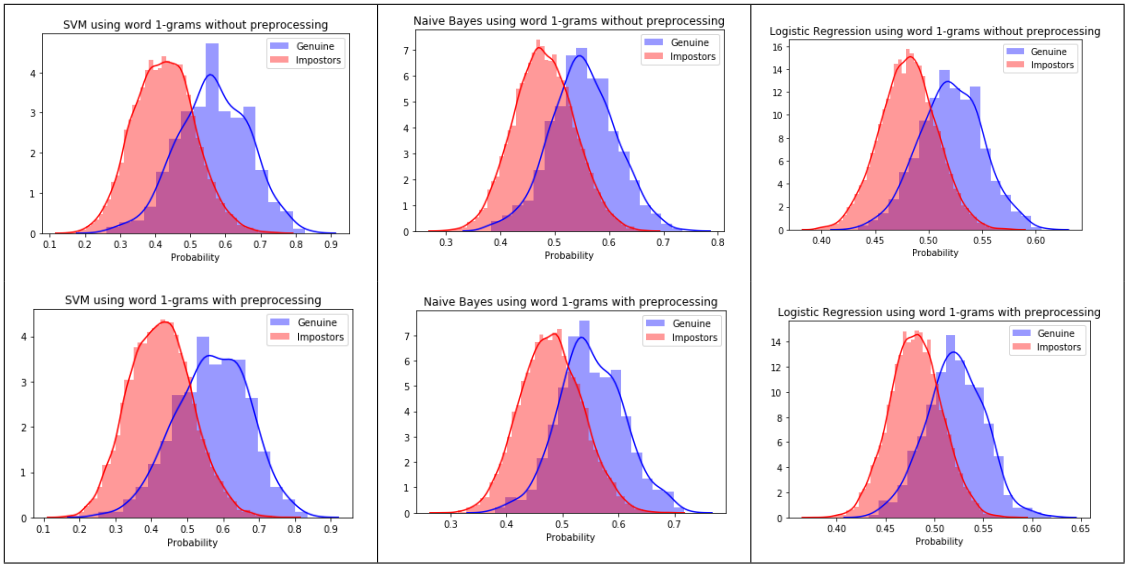

Figure D.1: The distribution plots from table D.1

	Word 2-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.246	0.255	0.253	0.244	0.248	0.246
True Negative Rate	0.754	0.744	0.745	0.752	0.752	0.751
Threshold	0.494	0.510	0.499	0.495	0.511	0.500
AUROC	0.841	0.835	0.839	0.841	0.839	0.842

Table D.2: Results from classification on the Stewart dataset using word 2-grams

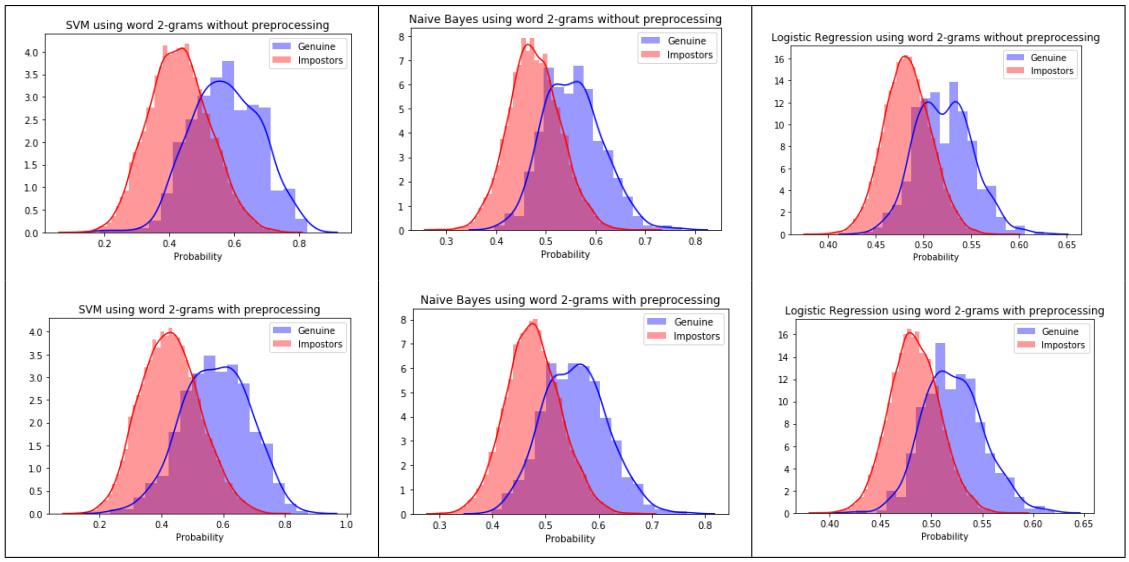

Figure D.2: The distribution plots from table D.2

	Word 1- and 2-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.215	0.240	0.226	0.225	0.244	0.226
True Negative Rate	0.783	0.760	0.772	0.774	0.754	0.774
Threshold	0.488	0.502	0.498	0.484	0.501	0.498
AUROC	0.861	0.836	0.851	0.864	0.838	0.852

Table D.3: Results from classification on the Stewart dataset using word 1- and 2-grams

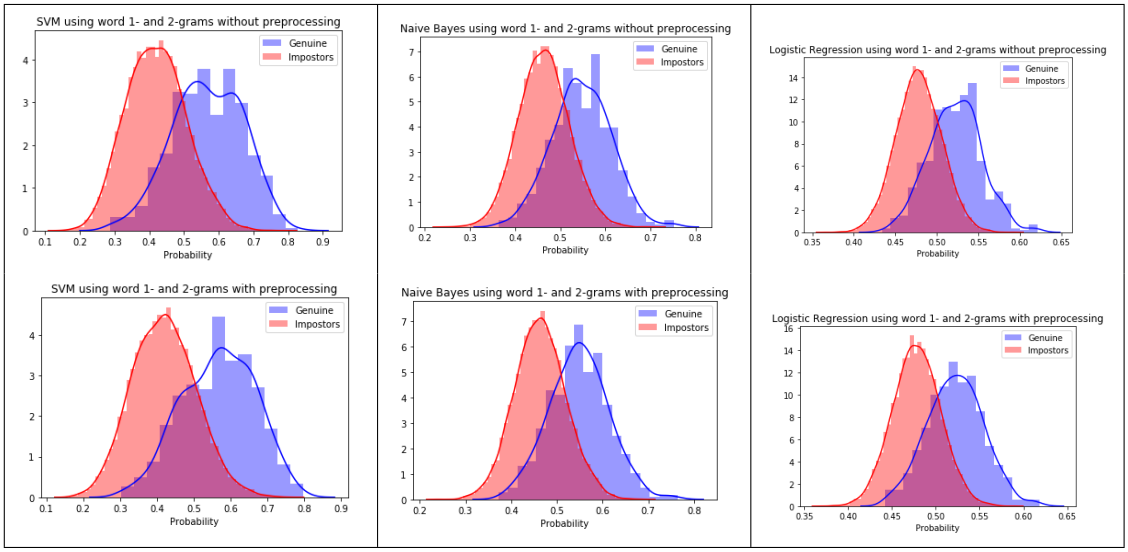

Figure D.3: The distribution plots from table D.3

	Character 3-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.248	0.296	0.265	0.240	0.282	0.257
True Negative Rate	0.751	0.703	0.733	0.758	0.716	0.774
Threshold	0.492	0.505	0.498	0.490	0.505	0.499
AUROC	0.832	0.779	0.809	0.841	0.787	0.817

Table D.4: Results from classification on the Stewart dataset using character 3-grams

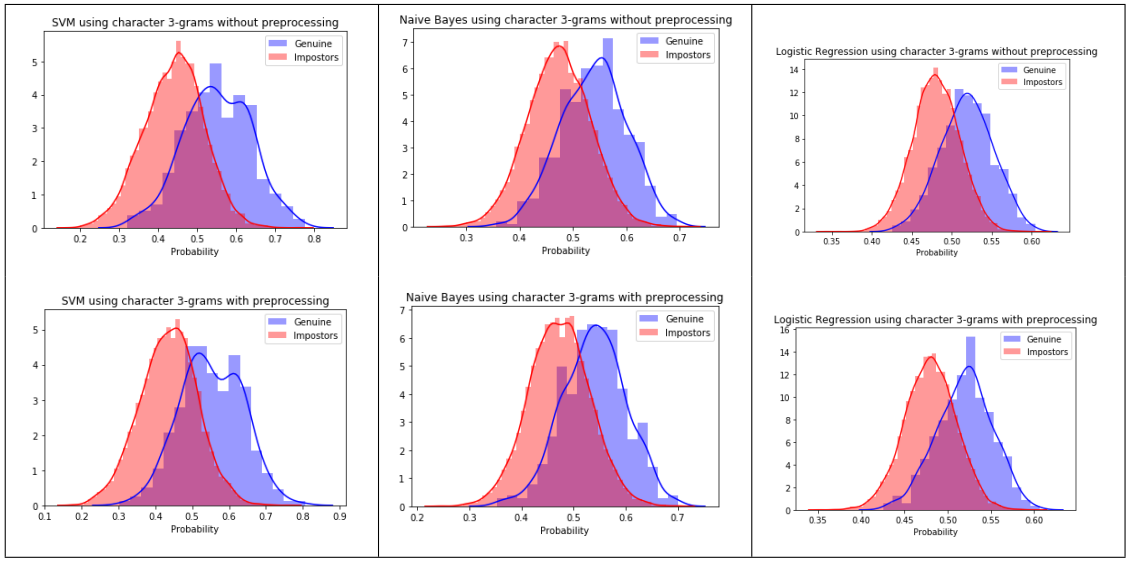

Figure D.4: The distribution plots from table D.4

	Character 4-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.248	0.298	0.263	0.238	0.284	0.257
True Negative Rate	0.752	0.700	0.738	0.761	0.714	0.741
Threshold	0.491	0.502	0.499	0.491	0.504	0.499
AUROC	0.830	0.777	0.808	0.839	0.788	0.820

Table D.5: Results from classification on the Stewart dataset using character 4-grams

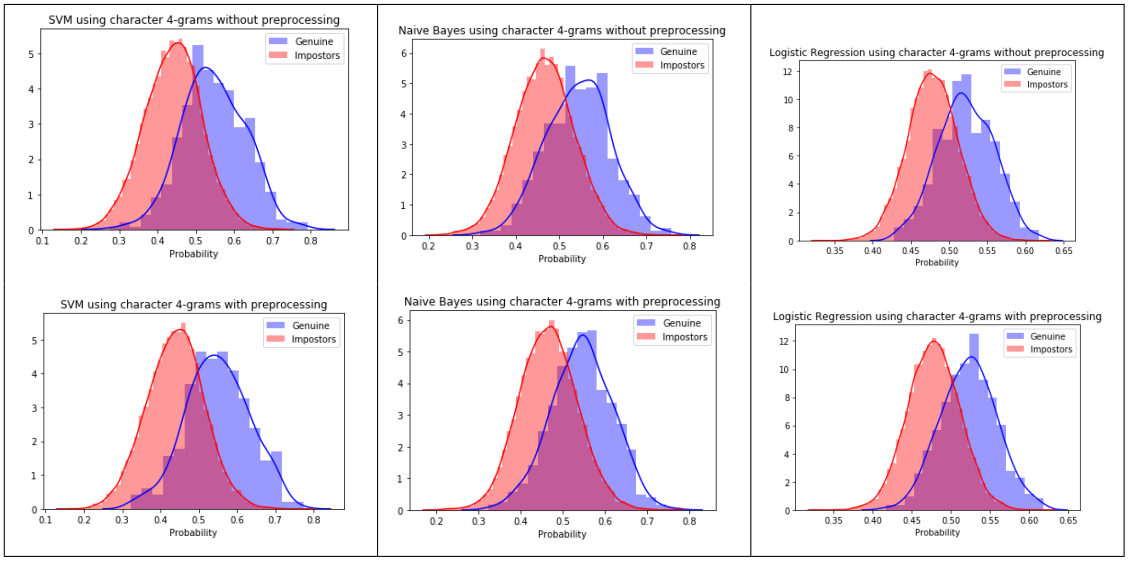

Figure D.5: The distribution plots from table D.5

	Character 3- and 4-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.253	0.301	0.276	0.255	0.301	0.263
True Negative Rate	0.743	0.702	0.722	0.741	0.698	0.735
Threshold	0.494	0.505	0.498	0.491	0.502	0.500
AUROC	0.823	0.767	0.799	0.827	0.768	0.811

Table D.6: Results from classification on the Stewart dataset using character 4-grams

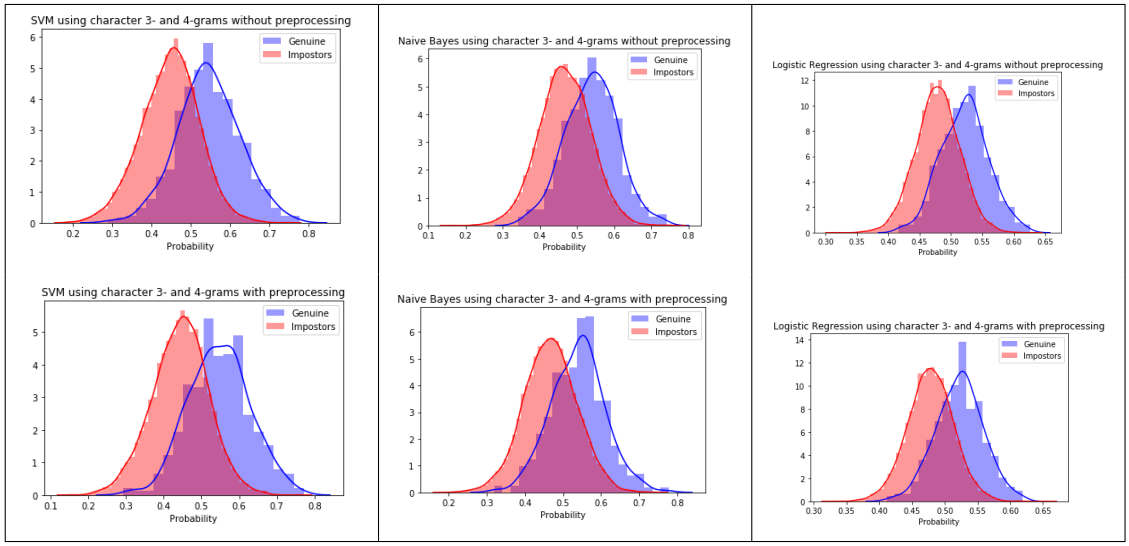

Figure D.6: The distribution plots from table D.6

D.2 Version 2

	Word 1-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.221	0.192	0.182	0.230	0.173	0.173
True Negative Rate	0.776	0.804	0.821	0.769	0.826	0.825
Threshold	0.501	0.512	0.501	0.497	0.513	0.501
AUROC	0.868	0.890	0.912	0.866	0.893	0.913

Table D.7: Results from classification on the Stewart dataset using word 1-grams with concatenated samples

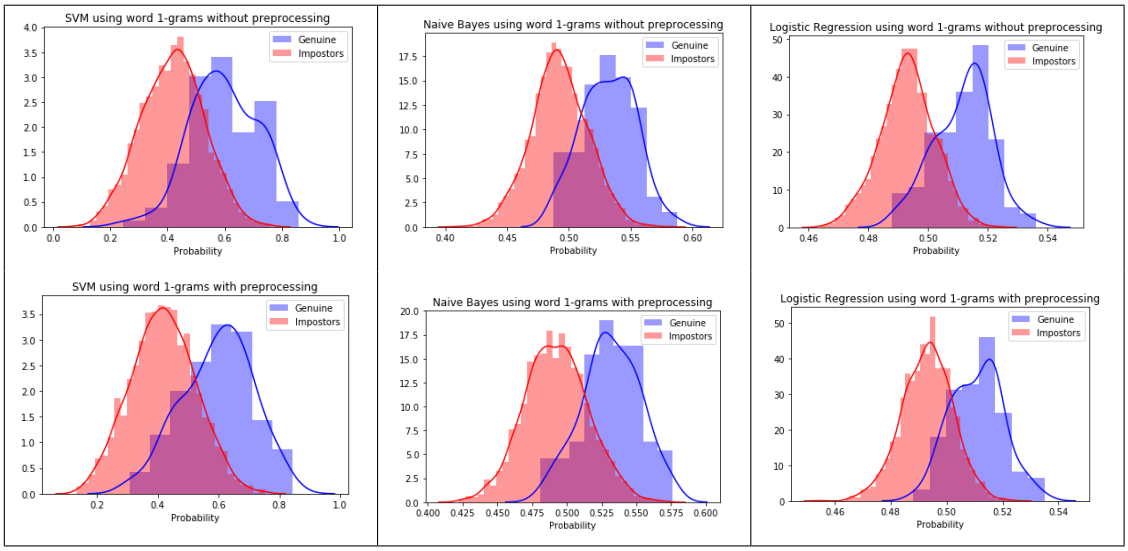

Figure D.7: The distribution plots from table D.7

	Word 2-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.163	0.153	0.144	0.182	0.144	0.125
True Negative Rate	0.841	0.850	0.832	0.821	0.851	0.870
Threshold	0.524	0.512	0.502	0.514	0.513	0.502
AUROC	0.914	0.935	0.932	0.921	0.933	0.943

Table D.8: Results from classification on the Stewart dataset using word 2-grams with concatenated samples

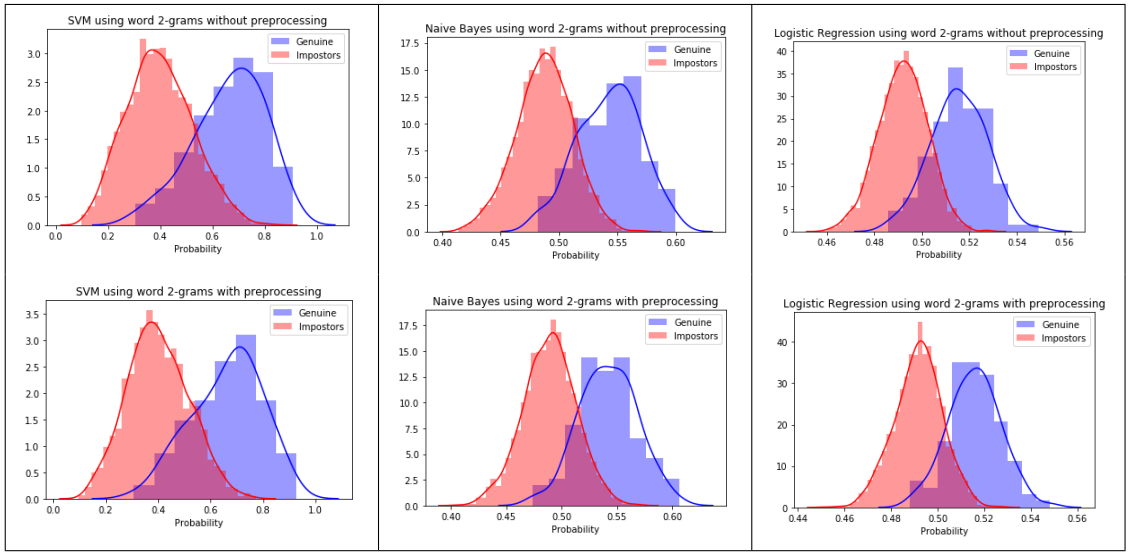

Figure D.8: The distribution plots from table D.8

	Word 1- and 2-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.173	0.173	0.144	0.192	0.163	0.134
True Negative Rate	0.825	0.826	0.855	0.800	0.838	0.870
Threshold	0.499	0.507	0.500	0.490	0.508	0.501
AUROC	0.894	0.920	0.936	0.908	0.926	0.946

Table D.9: Results from classification on the Stewart dataset using word 1- and 2-grams with concatenated samples

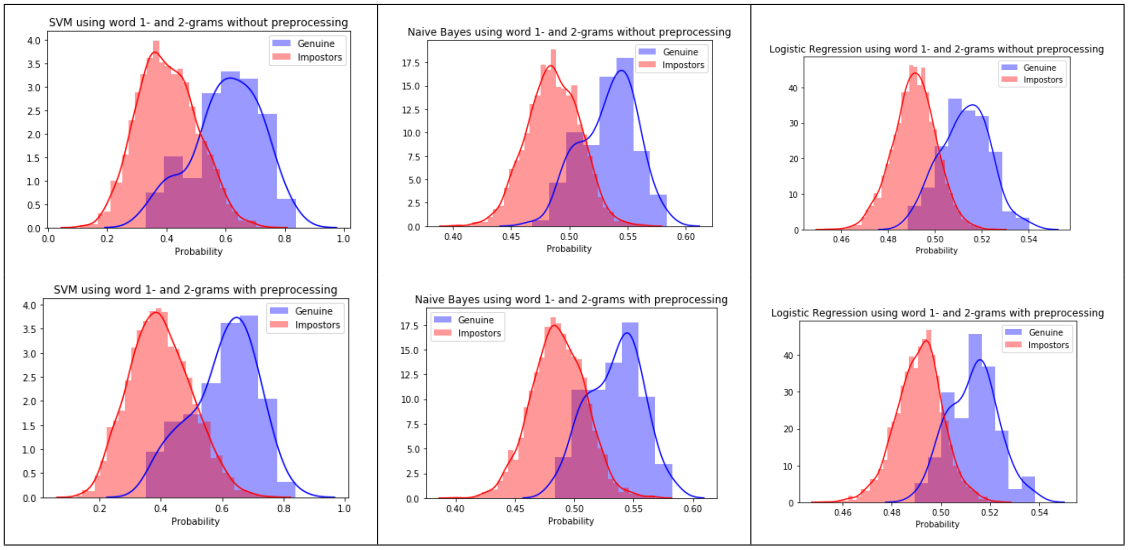

Figure D.9: The distribution plots from table D.9

	Character 3-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.250	0.250	0.221	0.259	0.250	0.201
True Negative Rate	0.755	0.748	0.773	0.748	0.754	0.776
Threshold	0.506	0.503	0.500	0.500	0.504	0.500
AUROC	0.832	0.845	0.874	0.841	0.842	0.883

Table D.10: Results from classification on the Stewart dataset using character 3-grams with concatenated samples

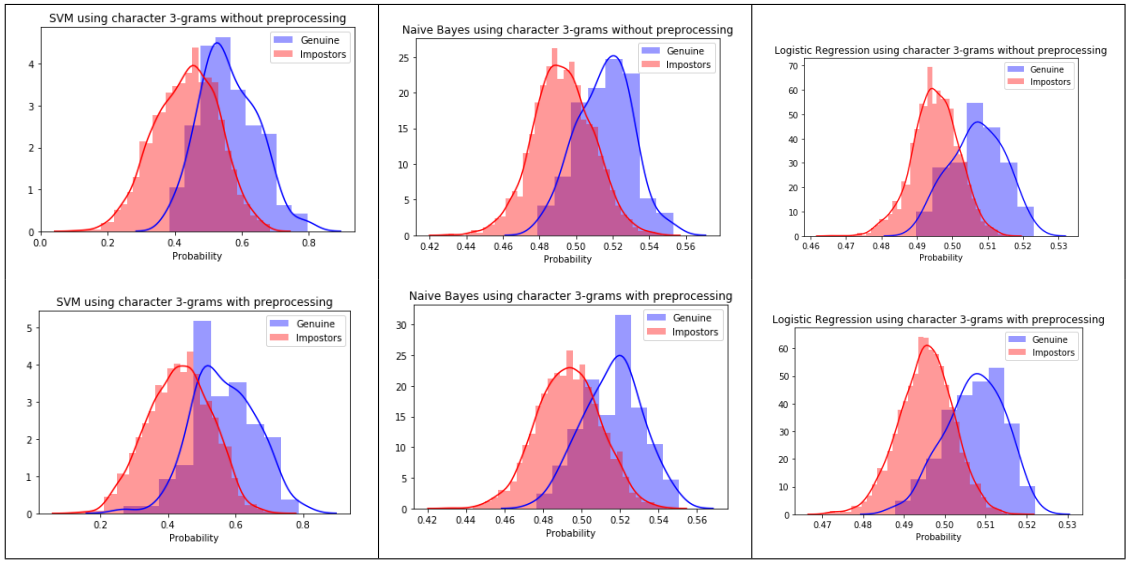

Figure D.10: The distribution plots from table D.10

	Character 4-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.259	0.250	0.211	0.259	0.230	0.211
True Negative Rate	0.737	0.752	0.801	0.742	0.767	0.772
Threshold	0.505	0.502	0.501	0.501	0.502	0.500
AUROC	0.827	0.860	0.878	0.825	0.872	0.884

Table D.11: Results from classification on the Stewart dataset using character 4-grams with concatenated samples

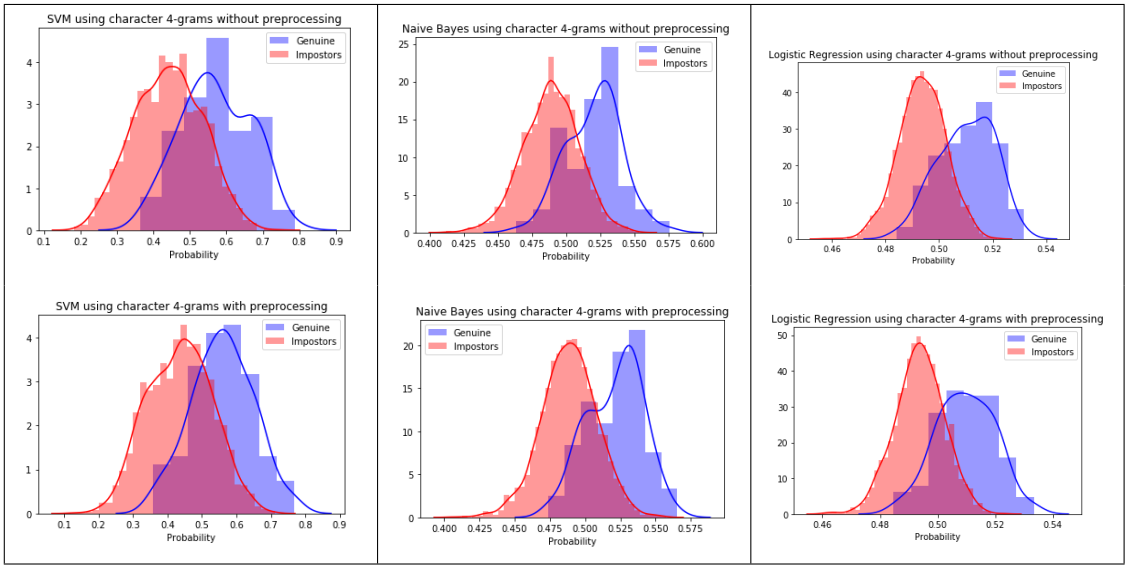

Figure D.11: The distribution plots from table D.11

	Character 3- and 4-gram					
	Raw text			Pre-processed text		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.307	0.240	0.240	0.317	0.240	0.230
True Negative Rate	0.688	0.758	0.762	0.683	0.768	0.768
Threshold	0.499	0.503	0.500	0.491	0.503	0.501
AUROC	0.755	0.830	0.845	0.779	0.835	0.841

Table D.12: Results from classification on the Stewart dataset using character 3- and 4-grams with concatenated samples

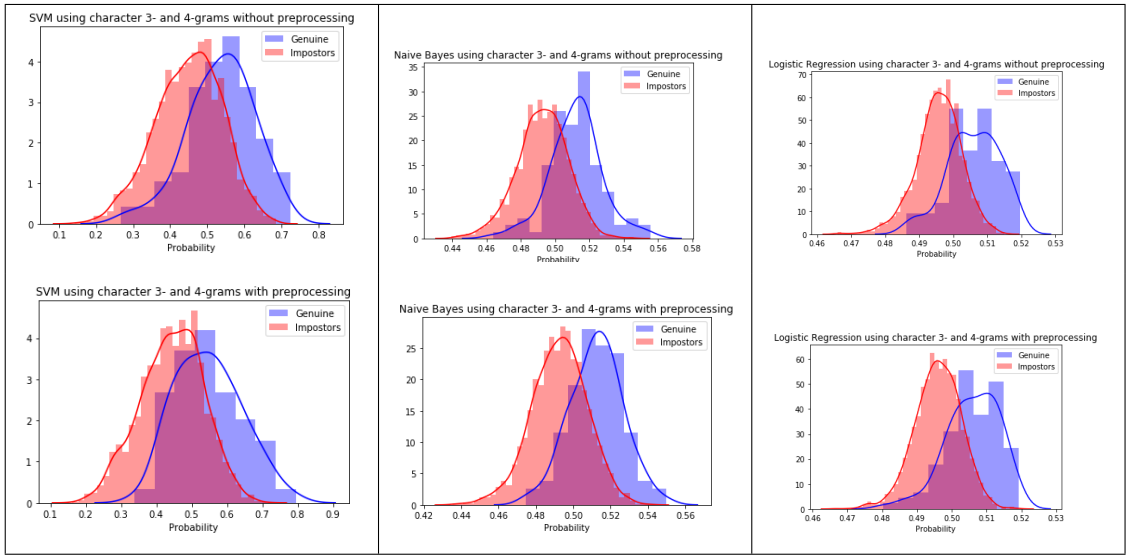

Figure D.12: The distribution plots from table D.12

Appendix

Appendix E

The results from the biometric performance evaluation on the Stewart dataset using keystroke dynamics.

E.1 Version 1

	Stewart keystroke data, no concatenation					
	duration features			latency features		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.033	0.188	0.017	0.096	0.186	0.082
True Negative Rate	0.967	0.812	0.984	0.907	0.814	0.918
Threshold	0.588	0.607	0.770	0.554	0.511	0.633
AUROC	0.995	0.883	0.998	0.964	0.894	0.973

Table E.1: Results from classification using duration- and latency features separately.

Combined features			
	SVM	NB	LogReg
False Negative Rate	0.029	0.127	0.0654
True Negative Rate	0.971	0.872	0.934
Threshold	0.597	0.640	0.649
AUROC	0.997	0.935	0.982

Table E.2: Results from classification using a combination of duration- and latency features.

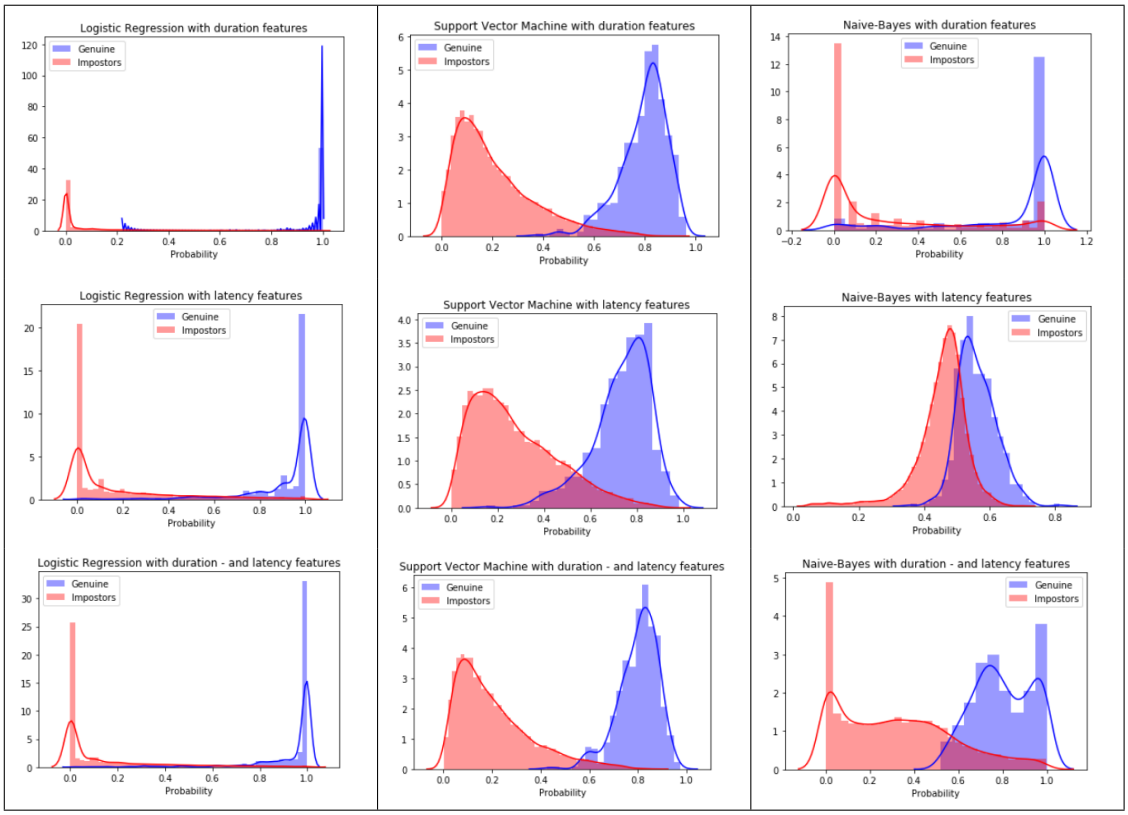

Figure E.1: The distribution plots from tables E.1 and E.2.

E.2 Version 2

Stewart keystroke data, five samples concatenated						
	duration features			latency features		
	SVM	NB	LogReg	SVM	NB	LogReg
False Negative Rate	0.087	0.087	0.038	0.135	0.192	0.135
True Negative Rate	0.918	0.912	0.965	0.868	0.813	0.860
Threshold	0.529	0.813	0.820	0.524	0.532	0.688
AUROC	0.976	0.968	0.995	0.950	0.896	0.942

Table E.3: Results from classification using duration- and latency features separately, where each profile have 4 samples each, where five and five single samples are concatenated.

Combined features			
	SVM	NB	LogReg
False Negative Rate	0.048	0.125	0.106
True Negative Rate	0.951	0.876	0.894
Threshold	0.582	0.721	0.741
AUROC	0.985	0.945	0.966

Table E.4: Results from classification using a combination of duration- and latency features on the concatenated data.

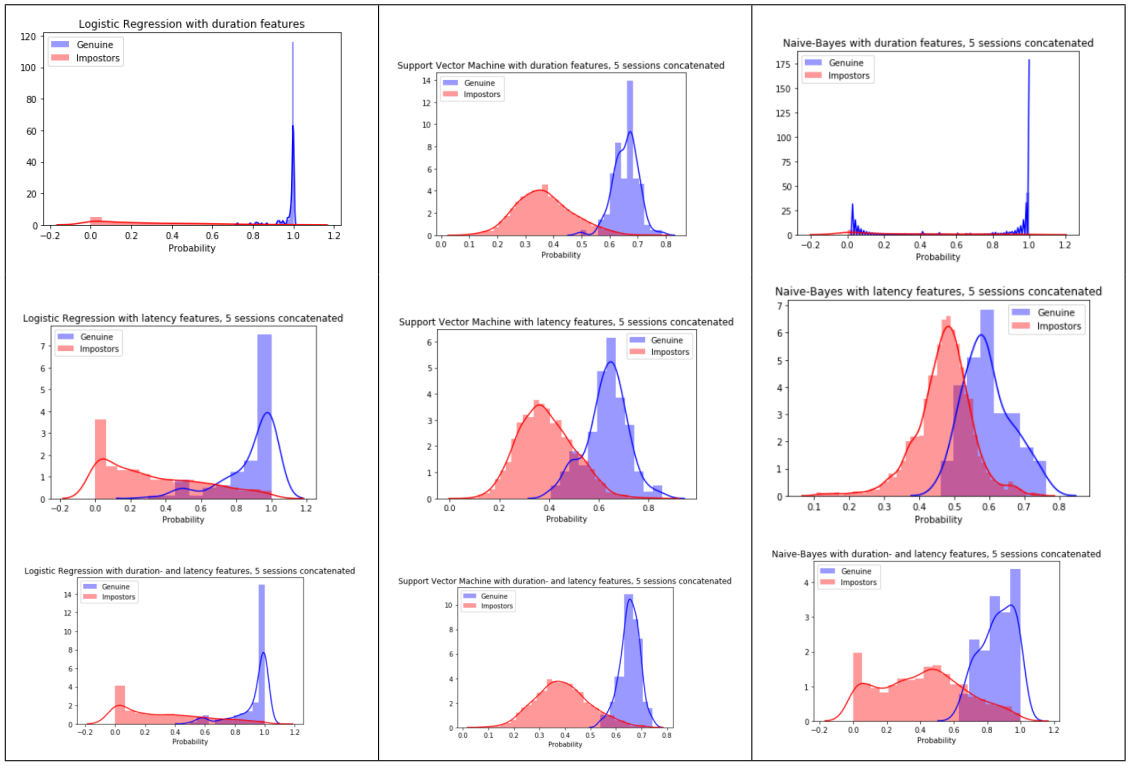

Figure E.2: The distribution plots from tables E.3 and E.4.