Behavioural Biometrics for PIN Authentication

V. Wassenaar

Delft University of Technology

TU Delft

Behavioural Biometrics for PIN Authentication

by

V. Wassenaar

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Thursday May 1st, 2025 at 15:00.

Student number:	4251490	
Project duration:	September 1, 2024 – May 1, 2025	
Thesis committee:	Dr. D. de Laat,	TU Delft, supervisor
	Dr. A. Heinlein,	TU Delft
	L. de Kroon,	Ubiqu, company supervisor

This thesis is confidential and cannot be made public until December 31, 2025.

Style: TU Delft Report Style

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft

Preface

The research presented in this thesis was conducted as final part of my studies in the Master of Science in Applied Mathematics program at TU Delft, and serves as the final requirement for the completion of the degree. The work presented in this thesis was carried out at Ubiqu, from September 2024 to April 2025, where I had the opportunity to apply my academic knowledge in a professional setting and gain valuable industry experience. I am grateful for this opportunity and all the help I received from my colleagues, as well as for taking me in.

Originally the thesis project consisted of two parts; Behavioural Biometrics for PIN authentication and Server Traffic Analysis. During the project it became clear that the scope of the former is substantial enough and would take up all available time. Hence this thesis will solely focus on Behavioural Biometrics for PIN Authentication. Besides researching Behavioural Biometrics through literature studies and developing methods to grasp the subject, I also spend a lot of time writing code both for analysis and data collection.

Not included in this report is the signification amount of time, during November/December 2024 and January 2025, I spend on researching and implementing Pairing-based Cryptography services at Ubiqu. In particular I worked on the most recent IETF draft for BBS signature schemes. Both as a mathematician and as a developer this was challenging, and provided valuable experience. This secondary project seems fitting given my specialisation Discrete Mathematics and Optimisation. Covering both Discrete Maths as well as Optimisation during my thesis project feels satisfying.

First, I would like to express my sincere gratitude to my academic supervisor, David de Laat, for his invaluable guidance, continuous support, and insightful feedback throughout the course of this thesis. His expertise and encouragement, as well as our mutual discourses, were essential to the success of this work. In particular helping me overcome points I got stuck. I am also sincerely thankful to my daily supervisor at Ubiqu, Lars de Kroon, for their practical guidance, technical input, and for making my time at the company both educational and enjoyable. The lack of hand-holding allowed me to rapidly familiarize myself with Ubiqu's code stack. His support helped bridge the gap between academic theory and real-world application and prioritise actions. I would also like to thank another member of my thesis committee, Alexander Heinlein, for taking the time to review my work and for their valuable comments and suggestions. Another expression of gratitude goes out to David Tax, who was willing to sit with me and share his extensive knowledge on One-Class Classifiers. His insights proved very valuable, in particular with regards to SVDD kernels.

A special thank you goes to my partner, Maaike Mossinkoff, for her unwavering support, patience, and encouragement throughout this journey. Not just during the duration of this research project, but since the moment I set out on starting a whole new studies. Your belief in me was been essential. Last but certainly not least, I want to thank my family and friends who have been there throughout my academic journey, cheering me on in both good times and challenging moments.

Finally, it would delight me if this thesis will contribute to further research and development in the field of Applied Mathematics. In particular contributing to the novel field of behavioural biometrics for authentication feels satisfactory.

V. Wassenaar
Delft, April 2025

Abstract

A PIN pad is a common way to authenticate, in particular for mobile applications. To strengthen PIN authentication we utilize behavioural biometrics in the form of keystroke dynamics. For authentication we require new PIN entries from the actual user to be accepted, while entries from an adversary to be rejected. Hence we strive to capture a user profile as compact as possible, such that false acceptance rates (FAR) are low, while the true acceptance rate (TAR) remains high. We estimate a user profile P by training one-class classification methods on user data U. We compare three one-class classification methods, namely Multivariate Gaussian Estimation (MGE), Support Vector Data Description (SVDD) and k-Minimal Enclosing Ball (k-MEB). We collected PIN data and analysed nine users which entered their PIN correctly at least 50 times. We find all methods outperform dummy classifiers. In most cases, a TAR score of over 0.8 combined with a maximum FAR score of at most 0.1 is achieved after tuning. Therefore, we conclude adding a behavioural biometric check does increase security of PIN authentication substantially.

Contents

Nomenclature

Abbreviations

Abbreviation	Definition
PIN	Personal Identification Number
MEB	Minimum Enclosing Ball
SVDD	Support Vector Data Description
MGE	Multivariate Gaussian Estimation
TAR	True Acceptance Rate
FAR	False Acceptance Rate
EER	Equal error rate for type I and type II errors
FRR	False Rejection Rate
ITI	Inter-tap Interval
KPT	Keypress Time
BB	Behavioural Biometrics
MFA	Multi-Factor Authentication
PCA	Principal Component Analysis
SVD	Singular Value Decomposition
UUID	Universally unique identifier

1 Introduction

Accessing and using (secure) systems requires authentication and authorisation processes. Ideally, secure systems remain easy to use, yet hard to compromise. The coming of digital identities will mark a new era in which systems will become easier to use whilst also harder to abuse externally. As more institutions and companies will adopt digital identities, both the variety and gravity of (legal) actions users can undertake will increase. Actions such as buying a house, applying for subsidies or sharing medical files are already worked on to be less cumbersome. Similarly, consequences of these actions increase if used maliciously by adversaries (i.e. hackers). Thus, with great power comes great responsibility, which implies that authentication processes need to as secure as possible.

This research focusses on ways to strengthen authentication processes using behavioural traits. Hence, we reiterate why the introduction of digital identities motivates research into behavioural biometrics for authentication processes. Then, authentication methods will be covered and in particular inherence based authentication (i.e. Biometrics). Finally, we converge to PIN authentication processes.

1.1. Digital Identities

The digital realm is expanding and strives to simplify our lives. A new chapter herein are Digital Identities, opening the way to more impactful actions for citizens to undertake. The European Union recently introduced Digital identity eIDAS 2.0 [60], which provides rules, regulation and standardisation which will boost this development. Consider filing for a new study, a subsidy or buying a house. All of these actions require additional documents, which themselves can only be obtained by verification of a person's (digital) identity. A digital identity can be stored in an E-Wallet, similar to a physical passport (card) in a physical wallet. Already digital keys, passwords and loyalty cards could be stored in E-wallets, such as Google's Wallet or Ubiqu's Wallet, among many. With the addition of the digital identity, more personal documents could be securely stored and ready for verification when filing for a service. Most notably, identity documents such as driver's licences or passports will be supported. Other examples are medical files, certificates and diplomas. To stress this change in administrative processes we compare the traditional process with the new process made possible by self-ownership of Digital Identities. To do so, we use the following example; we want to register with a 'Protected Profession Registry'.

The efficiency of the new process is appealing. As mentioned before, the stakes are higher, hence more severe ramifications when hacked or locked-out. Therefore, storing a digital identity requires strict security regulations. To that end, a concept defined by the EU is Qualified Trust Service Provider (QTSP) [21].

> "A Qualified Trust Service Provider (QTSP) is an entity that offers electronic trust services in accordance with the standards set forth by the European Union's eIDAS Regulation. These services include electronic signatures, seals, timestamps, electronic delivery services, and more. QTSPs play a crucial role in ensuring the authenticity, integrity, and legal validity of electronic transactions, thereby fostering trust in digital interactions." - Ubiqu [61]

Thus, necessity is born to provide strong security options for smartphone use. We first lay some security concept foundations. Access to a protected environment or resource generally requires two

steps [59]:

- *Authentication*; Are you who you say your are
- *Authorisation*; Given who you are, are you eligible to this environment/resource

If an adversary is able to pass the authentication step of a person with a high level of access (i.e. crucial authorisation rights) the ramification can be disastrous. While there are ways to limit the risks of authorisation breaches, such as compartmentalisation [48], this thesis shall focus on the authentication step.

1.2. Authentication methods

There are three different major authentication types to distinguish:

1. Knowledge based
2. Object based (possession)
3. Inherence based (biometrics)

These types are also known as factors. A combination of these types creates Multi-Factor Authentication (MFA), see Figure 1.1. Most people are familiar with some version of MFA, most likely '2-step verification', which consists of a knowledge based step and an object based step usually. Additionally aspects like location and time can be included to strengthen MFA further.

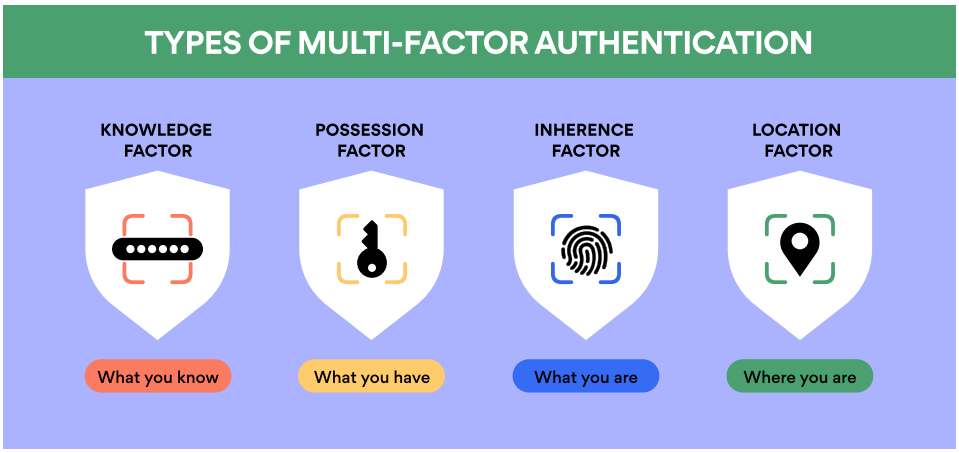


Figure 1.1: Authentication factors which make up Multi-Factor Authentication. Source: NordPass [42]

For knowledge based authentication one can think of a password, combined with username, email or ID number. Other knowledge based authentication methods are PIN codes, security questions or images (graphical secrets) [33]. Human behaviour is not random and for easy remembrance passwords or PIN codes are often chosen random either [36]. This lowers the respective information entropy, as a lot of possible options become 'highly unlikely' and weakens the practical security of the authentication[43].

Object based authentication is simply an object in your possession, a so called 'token', which verifies you are indeed who you say you are. A token can be a physical key, number generator, or custom device. While generally more secure than knowledge based authentication, it still has to be stored and remains vulnerable to theft. [43]

The third type, inherence based authentication, leverages unique features of 'who you are'. Prime examples are DNA, voice, retina and iris and fingerprint [7]. Scans to utilise these physical biometrics have long been reserved for highly secure facilities. The last few decades however, it has become widely available and affordable. For example, besides the fingerprint scan on (smart)phones [17], there are also 'smart guns' who can only be used after authentication by fingerprint [51]. Furthermore, combining inherence based authentication with the previous types of authentication strengthens the security. For example, Microsoft announced development of a Token Ring, which combines object authentication with inherence based authentication [20]. In other words the ring only works on you and stealing it would not benefit an adversary. Sadly, proven techniques for stealing fingerprints can be found in literature, and likewise for DNA samples [68].

1.2.1. Behavioural Biometrics

One possible route to overcome this hurdle is to include **behavioural biometrics** (BB), in stead of only acknowledging one's physical biometrics (PHB). There are various behaviours which are useful for authentication, such as keystroke dynamics, pattern recognition or (sub)conscious preferences [47]. A categorised overview of behavioural biometrics is shown in Figure 1.2. Within the scope of this thesis we focus on context limited natural behaviour, in particular PIN code typing.

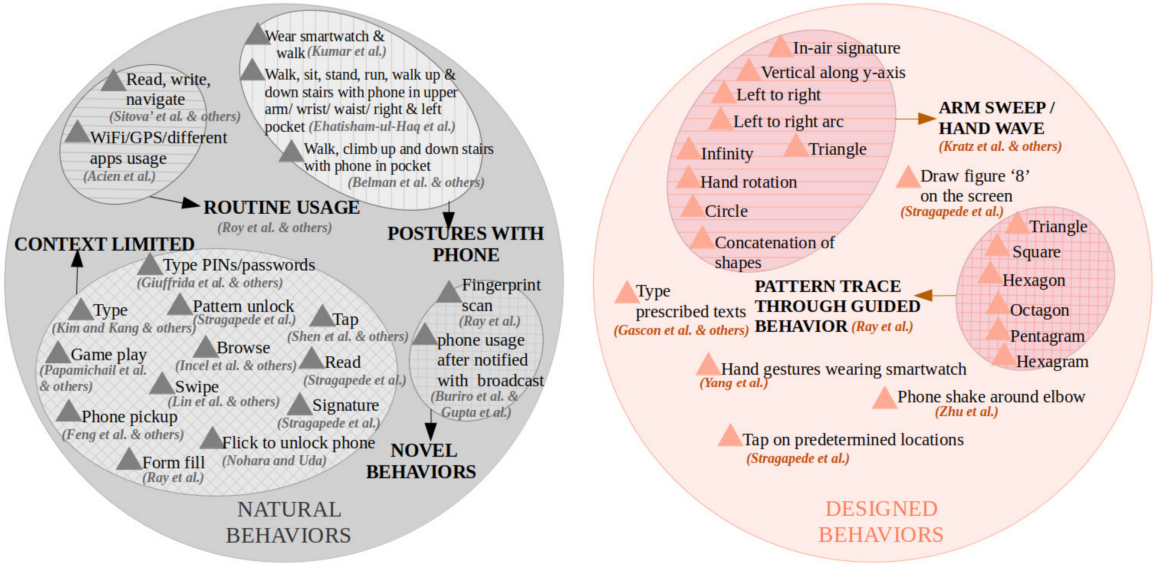


Figure 1.2: Subcategories of different user behaviours within natural and designed behaviour sets. Source: Ray-Dowling et al. (2023) [47]

The theoretical advantage of behavioural biometrics is that the 'learnt' behaviour is difficult to mimic or copy [2]. Even if seen, they corresponding association is not 'observed'. In some sense, this behaviour is also secret to the user itself, as it is often hard to explain or teach to another.

Nevertheless, attacks to behavioural biometric authentication methods have been designed and tested [41, 67]. One such attack, proposed by [27], consists of adding adversarial noise to fool the classifier, known as an Evasion Attack [25, 39, 64]. Therefore, in practice, behavioural biometrics could serve as an authentication layer on top of other authentication types. For example, the way you scan your face, or they way you type your password, or the pattern in which you carry your token. As a result, we recall for authentication, a low *false acceptance rate* (FAR) is usually considered more important than *false rejection rate* (FRR) [47]. When observed behaviour differs substantially to the expected behaviour, the authentication manager can trigger a retry or forces the user to try another authentication method, if available [44]. Another reason to do so are practical difficulties of behavioural biometrics. For example, how to deal with sudden change of user behaviour; Imagine a user being under influence, suffering from fatigue, or feeling severely stressed [52]. Also injury can jeopardise the user's default behaviour. Interesting to note, novel research also tries to link biometric markers to these states, which is out of the scope of this thesis [14]. Moreover, ethical questions like "should a person under influence be allowed to login to a service, like a bank", while very interesting, are also far outside our scope.

To classify correctly and reduce the false acceptance rate as much as possible, mathematical models are required to quantify when an observed behaviour *differs substantially* from the stored, known user behaviour. This default stored user behaviour is known as a *user profile* or 'golden standard'.

Literature provides multiple methods in quantifying this difference between behaviours.

We recall that combining multiple different authentication types allows for secure Multi-Factor Authentication.

1.3. PIN code authentication

PIN authentication is a traditional form of knowledge based authentication. PIN codes are designed to be single number, consisting of generally 4 to 6 digits, mostly entered via a PIN pad. PIN pads can consist of either physical (i.e. door lock) or digital buttons (i.e. smartphone) [36]. Most people will be familiar with an ATM PIN pad, like the one in Figure 1.3. Even though a PIN code should not be written down, saved or stored otherwise (both physically or digitally), still users continue to do so, for various reasons. A legitimate reason could be a person suffering from dementia wanting to withdraw cash from an ATM, for example.

Figure 1.3: Image of a traditional ATM PIN pad.

Even if not saved or stored, the PIN chosen is oft related to a personal event or memory and rarely drawn randomly [13]. Analysis of over 3.4 million leaked PIN codes illustrates this as well, see Figure 1.4. This fact, by definition, makes the PIN less secure [36]. PIN code selection/generation policies are proven to be effective, but limit the user's freedom of choice [29]. Moreover, if the generation process is publicly known this could be exploited.

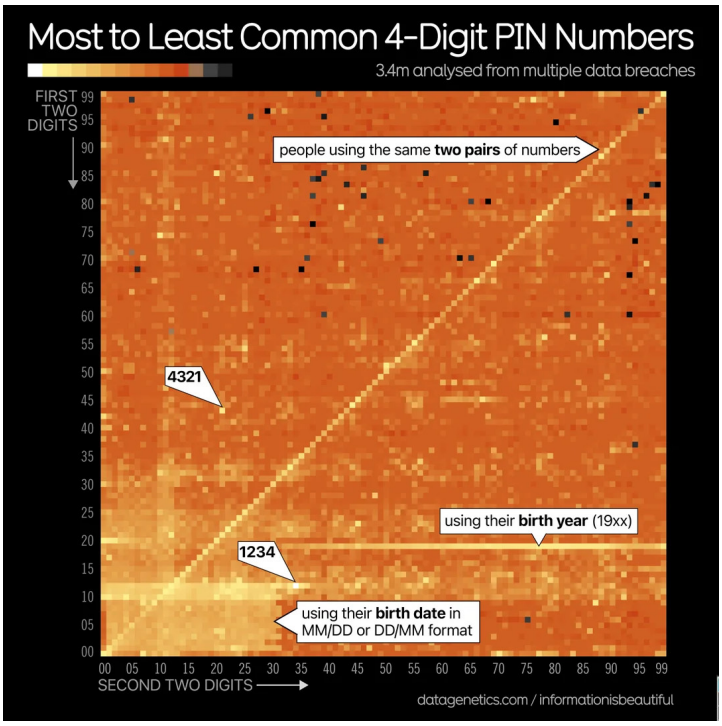


Figure 1.4: Frequency heatmap of used PIN codes for ∼3.4 million PIN codes [6]

.

During the authentication process the PIN code is at risk of being observed. Observations by a human adversary can happen both on-site or remote. Remote options include audio/visual instruments, such as (video) recorders, as well as key loggers [13]. Interestingly, even direct Brain-Computer interfaces remain vulnerable to such attacks [38].

A year of birth is a prime example of a PIN code which severely weakens the level of security of PIN authentication. A more in-depth PIN analysis shows how educated guesses weaken the security further [6]. Especially on smartphones, smudge attacks can be dangerous on their own, but combined with some basic user information even more so. Thus the total amount of 'attacks' on the PIN security and be summarised by

- Bruteforce Attack
- Direct Observation Attack (during authentication process by man or machine)
- Indirect Observation Attack (finding the code on a storage medium)
- Smudge Attack
- Educated Guess Attack (version of the bruteforce attack given additional information on the target)

Several strategies could be employed to parry attacks. For example, warding-off smudge attacks can be achieved by permuting the digits on the PIN pad, either for each digit or just per transactions. Bultel et al. [13] describe several common PIN code authentication methods. The four most of common out the fix mentioned by Bultel et al. are:

1. PIN Code with a Deterministic Placement (PDP). A traditional ATM uses this (knowledge based) PDP PIN authentication method in unison with a credit/debit card (object based).
2. PIN Code with a non Deterministic Placement (PnDP); An example of this authentication method be found in the online game 'Runescape' when entering a PIN code to enter an item storage, see Figure 1.5. This was added to combat key logger as user had to click on the screen using their mouse and the location of the clicks could not be used to recover the PIN without the actual visual footage. The permutation of the digits changed for each and every digit.
3. PIN code Substring (PS) ; Sequential request for specific digits of the PIN string. Assuming a PIN string of length p, then k times, a position from $\{1, \ldots, p\}$ is drawn at random and the user has to enter the digit corresponding to the requested position. Note that some PIN codes like 333333 would not be suitable for this method. Moreover, this method is more efficient is the code to be remembered is at least six digits long.
4. PIN Code Modulo a Random Number (PR); Enter result of addition of the PIN code C with a given (random) value A modulo a random value R. For example, given PIN code $C = 1956$, addition of $A = 3948$ and modulo $R = 124$ the result to enter in the PIN pad is 76. This method is too cumbersome for practical use. Moreover it does not really increase security, since the PIN can be recovered quickly after a only a handful of modulo-addition-result triples using the Chinese Remainder Theorem.

While these methods do combat both smudge attacks and key loggers efficiently. None of them strengthen the PIN authentication process for a 'discovered' PIN code. Prevention of leaking a PIN code is desirable of course, but the question remains how to strengthen the PIN authentication given the PIN code is discovered by an adversary. In the following chapter we motivate why behavioural biometrics does provide this additional layer of security for PIN authentication.

Figure 1.5: Example of a PIN Code with a non Deterministic Placement (PnDP) from the online game 'Runescape. Copyright: ©Jagex Ltd [40]

2

Keystrokes dynamics for smartphone authentication

To strengthen knowledge based authentication protocols, behavioural biometrics (BB) is suggested to provide another layer of security, by detecting if the correct PIN was also entered in the 'right' way. The goal of this chapter is to quantify to which extend this concept holds in practice.

In Section 2.2 we investigate to what extend keystrokes dynamics, a specific behavioural biometric, can strengthen a 'default' PIN authentication. In Subsection 2.2.1 we explain how we abstract behavioural log data to vectors. Then, in Subsection 2.2.2, we analyse features and *stability* of learnt behaviour.

Section 2.3 will explore PIN authentication for customisable PIN pads, which allow for a 'personal' digit configuration layout. The layout can be modified both by the interchanging digit boxes as well as changing the (relative) box sizes.

Methods for building user profiles based on the behavioural vectors can be found in Chapter 3.

2.1. Keystrokes dynamics

Physical keyboards mainly provide two behavioural traits to observe. The keypress time [1] and inter-key times [2]. Keypress time is defined as the time duration between the initial activation of the key and the release of that key. The inter-key time is defined as the duration between the release of a key and the activation of the subsequent key press. Typing behaviour on a traditional, physical keyboard might also include tracking mistakes, how often the 'Backspace' key has been utilised.

A touchscreen keyboard offers more traits to be observed than a traditional keyboard. In this research we shall focus on

- Keypress time (KPT), also known as Event Time Span
- Inter-key time (IKT), also known as Inter-tap interval (ITI) or Inter Event Time
- Touch pressure (TP)
- Touch area (TA)
- (Relative) keypress/touch location (RKPL)

If besides keypresses (i.e. taps) other motion movement is tracked, like swiping even more features can be used to build a user's keystroke profile.

2.2. 5-digit PIN authentication

Ubiqu's Authenticate application contains a 5-digit PIN pad, used as a primary authentication method. An opt-in fingerprint scan is also available. For each digit touch event we collect behavioural traits. The PIN pad used for data collection is illustrated in Figure 2.1. We shall refer to entering a PIN sequence

[1] also known as 'dwell time' or 'hold time'
[2] also known as 'latency' or 'flight time'

for user authentication as a `transaction`. Collecting keystroke dynamics data on many transactions will allow us to build a profile of the user's behaviour based on behavioural traits, such as Keypress Time or Touch Area.

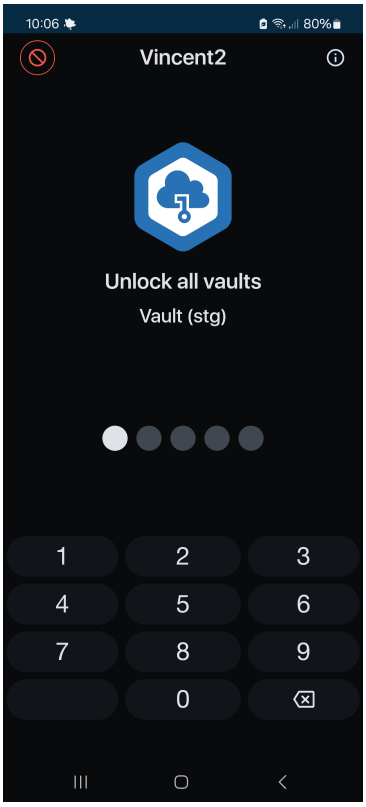


Figure 2.1: Ubiqu's PIN pad authentication screen.

2.2.1. Transaction data structure

Before we explain how we detect outliers for a given user profile, we explain the data structure used to build these profiles. For this thesis we collected three data sets: Baseline, Phase 1 and Phase 2.

Phase	PIN Pad Layout	Rules	PIN sequence
Baseline	Default	Strict	35753
Phase 1	Default	None	30194
Phase 2	Custom	None	30194

Table 2.1: Summary of Data Sets properties

Each data set consists of PIN entries which in the context of Ubiqu also refer to as `transactions`. While entering the PIN (i.e. completing a transaction) keystrokes dynamics are collected and logged. Logging occurs both on device, by writing the behavioural data to a logfile, and via a dedicated behavioural service (server).

We quantify behavioural biometrics for PIN entries by creating a vector containing the floating point values obtained by logging a transaction. For each digit we obtain six behavioural traits (features), except for the first digit, which has only five, as inter-key time (IKT) is not defined. This leaves us with $5 + 4 \cdot 6 = 29$ features in total.

For example, a logged transaction in JSON format, see Listing 1, is reduced to a 29 dimensional data table, see Table 2.2. We can further abstract the data (table) by casting it to a 29-dimension vector

as seen in Equation 2.1.

$$\begin{aligned}\mathbf{v} = \big[&0.0235294 \quad 1.0 \quad 131.879 \quad 55.8984 \quad 70.0 \quad 0.0 \quad 0.0313725 \quad 1.0 \quad 239.635 \quad 63.4844 \\ &92.0 \quad 216.0 \quad 0.027451 \quad 1.0 \quad 171.914 \quad 97.0234 \quad 65.0 \quad 266.0 \quad 0.0235294 \quad 1.0 \\ &136.539 \quad 85.1641 \quad 58.0 \quad 267.0 \quad 0.0372549 \quad 1.0 \quad 165.893 \quad 75.8203 \quad 49.0 \quad 232.0\big]\end{aligned} \tag{2.1}$$

Feature	TA	TP	RKPL-x	RKPL-y	KPT	IKT
Digit 1	0.027450982	1	202.01562	70.546875	64	-
Digit 2	0.03137255	1	144.71289	88.09375	66	108
Digit 3	0.027450982	1	224.1211	58.9375	58	133
Digit 4	0.03137255	1	134.95703	61.140625	49	125
Digit 5	0.027450982	1	166.41992	86.36719	58	50

Table 2.2: Keystroke dynamics log example

For some mobile devices, such as Google Pixel smartphones, `touch pressure` data is supported, hence informative. Samsung smartphones in data sets used in this thesis did not support it, so the (informative) dimension is reduced to 24. The vector from the example we reduce accordingly, see Equation 2.2.

$$\begin{aligned}\mathbf{v} = \big[&0.0235294 \quad 131.879 \quad 55.8984 \quad 70.0 \quad 0.0 \quad 0.0313725 \quad 239.635 \quad 63.4844 \\ &92.0 \quad 216.0 \quad 0.027451 \quad 171.914 \quad 97.0234 \quad 65.0 \quad 266.0 \quad 0.0235294 \\ &136.539 \quad 85.1641 \quad 58.0 \quad 267.0 \quad 0.0372549 \quad 165.893 \quad 75.8203 \quad 49.0 \quad 232.0\big]\end{aligned} \tag{2.2}$$

The example we used depicts a valid transaction. A PIN entry (i.e. `transaction`) is considered **valid** if the correct PIN is entered without use of backspace. Due to technical issues sometimes not all five digits have been logged correctly for every transaction, and therefore have been excluded from data analysis. Reasons for excluding these data points are dimension mismatch, abusability and irregularity. For the latter reason, we argue no user purposely enters their PIN in a fashion, like $3, 5, 4$, backspace, $7, 5, 3$ for example as supposed to $3, 5, 7, 5, 3$.

2.2.2. Behavioural data analysis

It is worthwhile to familiarise ourselves with the baseline data set in order to better understand the behavioural patterns. This data set was created for the sole purpose of this thesis under strict rules and provides us with insights in behaviour data of a single user. The baseline data set, also referred to as Data set A, is generated with PIN sequence $3, 5, 7, 5, 3$. The PIN code was entered by myself (i.e. author) periodically, sitting in a desk chair, with the mobile device stationary on the desk directly in front. The mobile device used is a Samsung Galaxy A52, with operating system Android 14 and One UI version 6.1. The PIN code was entered as follows

- 3; Right hand - Index Finger
- 5; Left hand - Index Finger
- 7; Left hand - Index Finger
- 5; Left hand - Index Finger
- 3; Right hand - Index Finger

Due to the movement of the left hand index finger between buttons for digits 5 and 7, and back, there is a inter-key time lower bound. In a single, continuous session, the PIN code was repeatedly entered on the mobile device by the same person. The PIN code $3, 5, 7, 5, 3$ was entered 164 times of which 161 were initially correct. For simplicity only initial successful authentication attempts are included in the data set analysis. Furthermore, there are physical limitations on the inter-key times [37], which also reduces the practical feature space. However, inter-tap intervals (ITI) are different between one finger taps and multi-finger taps [3]. Also the order of the tapping fingers plays a role for ITI [4]. It is possible to distinguish between people categorically, for example with or without a motor-impacting illness [30, 5], but identifying a *unique* profile is much more challenging [4].

```
{
      "TouchEventLogUUID": "8f7048f1-87bf-4261-aec4-1634c1da8f6e",
      "logItems": [
        {
          "logItemId": "08ae0865-9b59-4970-baba-5deff06c38bb",
          "touchArea": 0.027450982,
          "touchPressure": 1,
          "touchLocation": {
            "first": 202.01562,
            "second": 70.546875
          },
          "eventTimeSpan": 64
        },
        {
          "logItemId": "d2c30e4e-253f-4b94-87b2-93fc8252f475",
          "touchArea": 0.03137255,
          "touchPressure": 1,
          "touchLocation": {
            "first": 144.71289,
            "second": 88.09375
          },
          "eventTimeSpan": 66,
          "interEventTime": 108
        },
        {
          "logItemId": "23b4da54-bd23-457c-b04d-51f8c5ad522e",
          "touchArea": 0.027450982,
          "touchPressure": 1,
          "touchLocation": {
            "first": 224.1211,
            "second": 58.9375
          },
          "eventTimeSpan": 58,
          "interEventTime": 133
        },
        {
          "logItemId": "411f2766-9f56-4913-a026-57a60d1f934c",
          "touchArea": 0.03137255,
          "touchPressure": 1,
          "touchLocation": {
            "first": 134.95703,
            "second": 61.140625
          },
          "eventTimeSpan": 49,
          "interEventTime": 125
        },
        {
          "logItemId": "b2f64c36-29ce-460d-913d-3874fcd989af",
          "touchArea": 0.027450982,
          "touchPressure": 1,
          "touchLocation": {
            "first": 166.41992,
            "second": 86.36719
          },
          "eventTimeSpan": 58,
          "interEventTime": 50
        }
      ]
    },
```


Listing 1: JSON log example of PIN entry keystroke dynamics

While learning a behaviour is never finished, the initial phase of the learning curve is expected to be more volatile [12]. Identifying a feature which would display an 'elbow' characteristic could help us to pinpoint the transition from a learning phase to solidified/stable behaviour.

The first features we examine are keypress times for each respective digit. Shown in Figure 2.2 is the evolution of keypress times per digit. A moving average of 10 consecutive data points is chosen to smooth out noise in the data. While we see that, at least for this data sample, digit 1 is held down longer than the other digits. We could not distinguish a transition towards stable behaviour for the keypress time features. The combined (i.e. summed) keypress time of all five digits we also consider

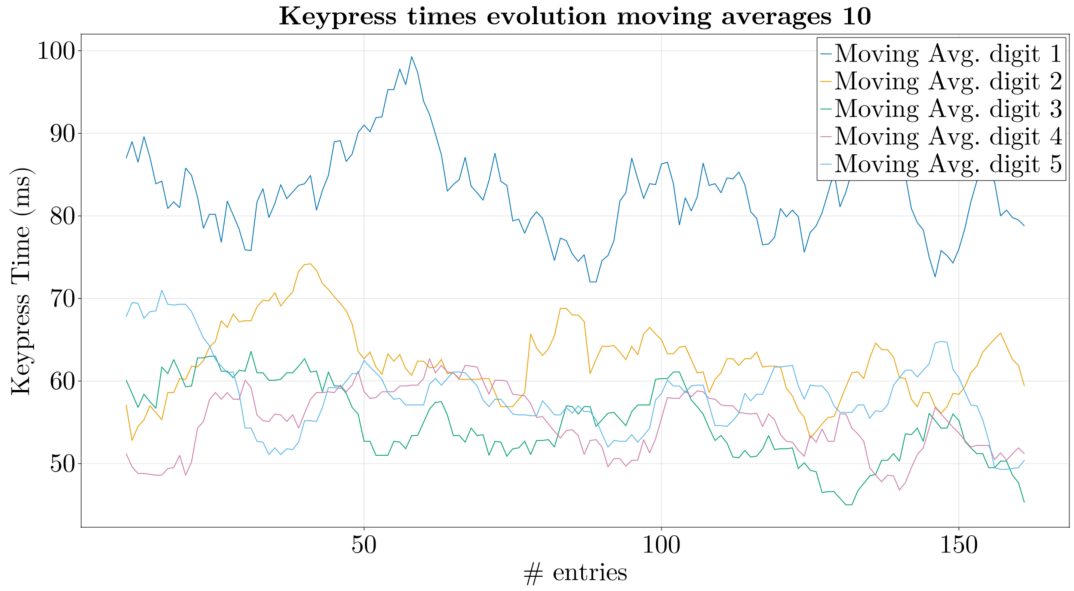


Figure 2.2: Moving average of 10 consecutive data points for feature `Keypress time` for all five digits.

as a derived feature. Again we use a moving average to find any trends, see Figure 2.3. While the trend appears downwards, no obvious infliction point could be observed. Contrary to derived feature

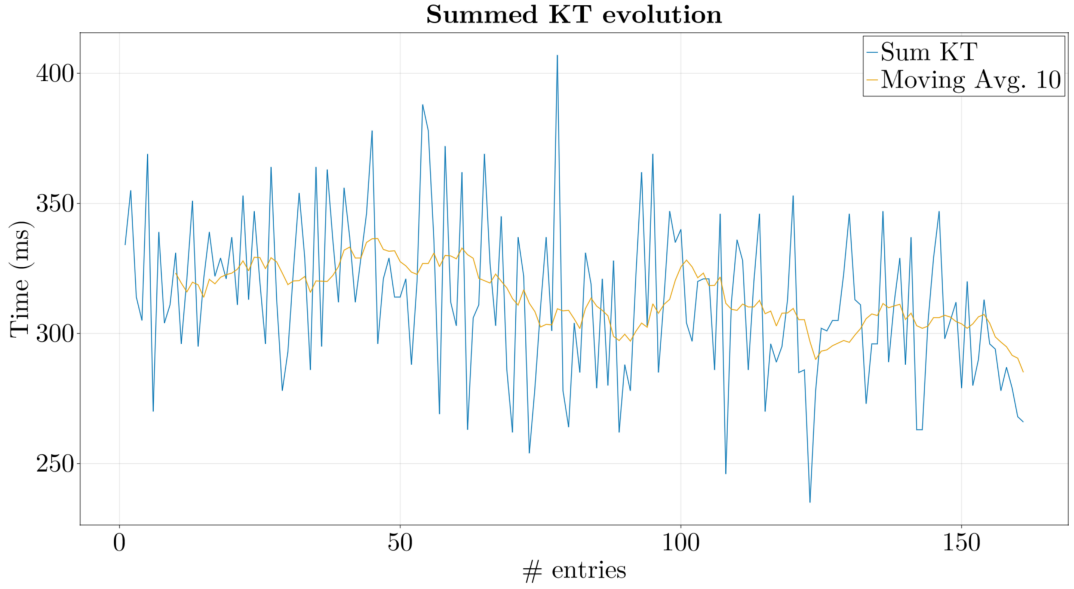


Figure 2.3: Evolution of derived feature `summed keypress time`. A moving average of 10 consecutive data points is also visible (yellow line).

`summed keypress time`, the derived feature `inter-key time` (ITI) does provide us with a transition to

stable behaviour. To quantify this transition point we iteratively computed a linear fit whilst removing the first data point. In other words, if the linear fit is not within the stability threshold, we remove the first data point and try again. If the slope of linear fit is less than our chosen threshold of 0.045 we stop and have found our transition point. In Figure 2.4 we show the result of this elimination algorithm, which finds us a linear fit after removing the first 30 data points from the data set. Based on these findings we argue to mark all entries after the 30th transaction as part of the stable user profile. This leaves us with 131 of the 161 data points for outlier analysis. Now we have obtained a stable behaviour subset of

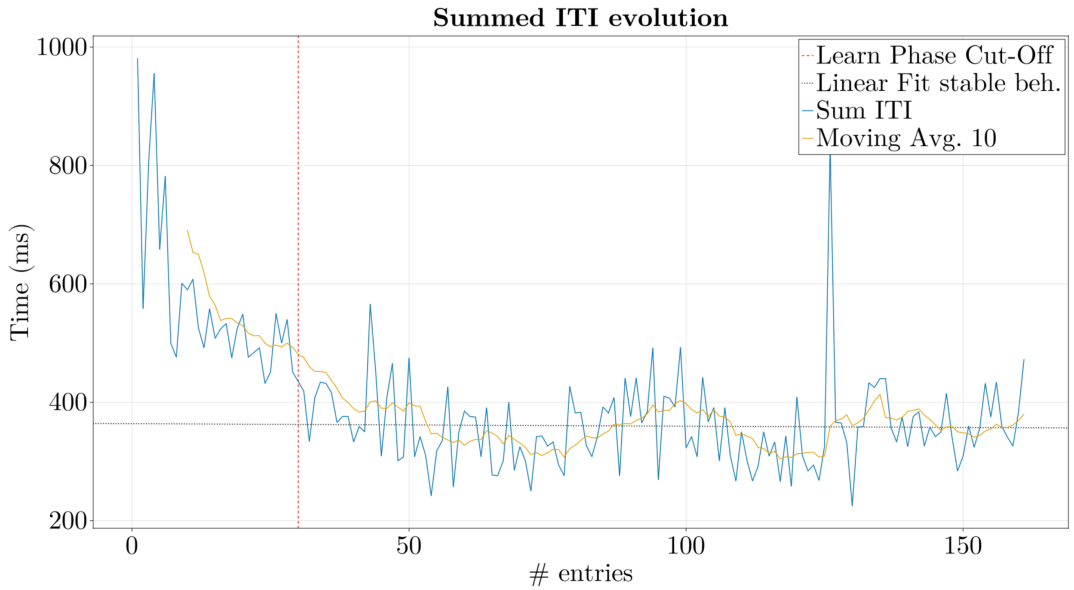


Figure 2.4: Caption

the data, we want to capture its essence. In other words, given a keystroke profile of a user U, which is a trained model on user input log data, how accurately can we predict if a PIN code is provided by user U. Since we only desire to know if the input was provided by user U, this problem can be viewed as a binary classification problem. In this case however, the classes are unbalanced. A priori, data points for the 'not user U' class can be (almost) anywhere in the feature space. So observing only data points for class 'user U'. Therefore, this problem can also be framed as detecting outliers. One way to detect outlier is to assume the user keystroke profile can be modelled as a cloud of points in the high dimension of the feature space. The underlying assumption here is the user behaviour has a **unique** profile.

In order to better grasp the interplay between features, a Principal Component Analysis (PCA) is performed on the user data. Mathematically speaking this effectively requires us to compute a Singular Value Decomposition (SVD). In Figure 2.5 the singular values are shown in descending order. Dimension reduction is achieved by only taking the most dominant singular values and disregarding others. In this case, dimension reduction is hard to defend as, likely, too much information is lost on the relation between features. In other words, the relative difference between de singular values is quite small, which makes it hard to argue which are to be labelled 'dominant'. Another reason to use PCA is to decorrelate features. This could help us to more accurately estimate a user profile. Hence, we shall apply methods to build user profiles both directly to the data and to the PCA representation of the data. When stating the results in Chapter 4 we shall compare both representations.

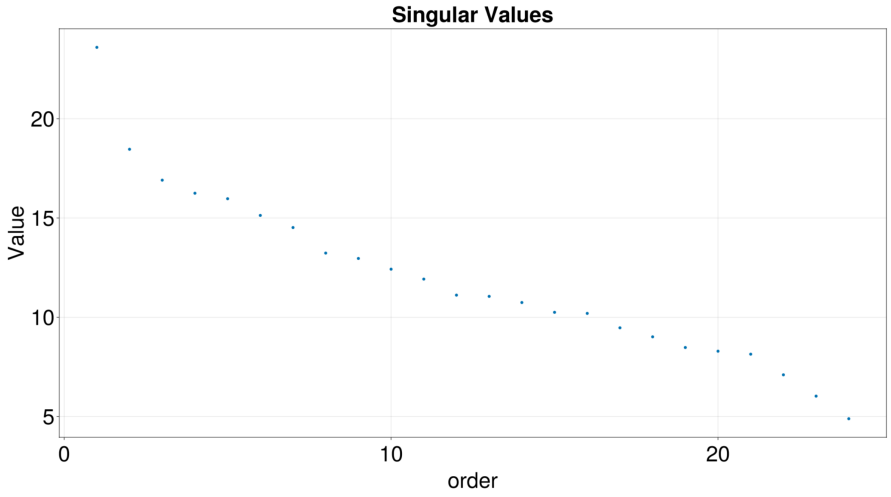


Figure 2.5: Singular values (in descending order) plotted for Data Set C

2.3. Customized PIN pad layout

To increase the distinctiveness of the learnt behaviour a customized PIN pad can be used. In this section we explain in detail how we map a user's unique *Asset* identifier to one of the 3628800 $(= 10!)$ different PIN pad permutations.

The Asset identifier is a Universally unique identifier (UUID) ensuring uniqueness [19]. Unfortunately the PIN pad has only $10! = 3628800$ permutations. The default sequence of digits is $(1, 2, 3, 4, 5, 6, 7, 8, 9, 0)$, split over four rows of the PIN pad, as seen in Figure 2.1.

First we convert the Asset UUID, which is 128-bit long, to an 32-bit integer for software purposes. Effectively we only consider the 32 least-significant bits of the UUID. Let us denote this 32-bit integer by a. Then, the 32-bit integer a is reduced to its modular representative $b \in \{0, \ldots, 10! - 1\}$. So we have

$$b \equiv a \pmod{10!} \tag{2.3}$$

Next we convert this integer b into an positional array, by using a factorial basis. For example $5 \to [2, 1, 0]$, since $5 = 2 \cdot 2! + 1 \cdot 1!$. Similarly $43 \to [1, 3, 0, 1, 0]$ because $42 = 1 \cdot 4! + 3 \cdot 3! + 0 \cdot 2! + 1 \cdot 1!$. The zeros at the front of the positional array have been omitted here for readability, but Table 2.3 states them fully. The 0 at the end is reserved for 0, meaning 0 would map to positional array $[0]$.

Integer	Positional Array
0	[0,0,0,0,0,0,0,0,0,1]
1	[0,0,0,0,0,0,0,0,1,0]
2	[0,0,0,0,0,0,0,1,0,0]
5	[0,0,0,0,0,0,0,2,1,0]
43	[0,0,0,0,0,1,3,0,1,0]
1001	[0,0,0,1,2,1,2,2,1,0]
$10! - 1$	[9,8,7,6,5,4,3,2,1,0]

Table 2.3: Tables of various integers represented as positional array in factorial basis notation, modulo $10!$.

Using the positional array representation we can construct the desired permutation in the following way. From left to right we read out the positional array and set corresponding 'basis' digit to the index, of the remaining slots to be filled in the permutation, prescribed by the factor. To illustrate this construction consider the case of integer 1001, with positional array $[0, 0, 0, 1, 2, 1, 2, 2, 1, 0]$. The permutation will be denoted brackets $(,)$, as per usual. The permutation is derived by the following steps. We start out

with an empty permutation and begin reading our positional array at the left most entry.

$$(9,\cdot,\cdot,\cdot,\cdot,\cdot,\cdot,\cdot,\cdot,\cdot) \quad [\underset{\uparrow}{0},0,0,1,2,1,2,2,1,0]$$
$$(9,8,\cdot,\cdot,\cdot,\cdot,\cdot,\cdot,\cdot,\cdot) \quad [0,\underset{\uparrow}{0},0,1,2,1,2,2,1,0]$$
$$(9,8,7,\cdot,\cdot,\cdot,\cdot,\cdot,\cdot,\cdot) \quad [0,0,\underset{\uparrow}{0},1,2,1,2,2,1,0]$$

Since the first zero represents the position of the digit 9 and all slots are available in the permutation we put a 9 in the first slot[3]. Then, the second left most entry in the positional array represents the position of the digit 8. Since the first slot is taken, the first available slot is directly right of the 9 already in place. Similarly we can place 7 into its slot next to 8. Now for placing digit 6 we have to be careful as the positional array indicates we need to fill the available slot which has index 1, resulting in

$$(9,8,7,\cdot,6,\cdot,\cdot,\cdot,\cdot,\cdot) \quad [0,0,0,\underset{\uparrow}{1},2,1,2,2,1,0]$$
$$(9,8,7,\cdot,6,\cdot,5,\cdot,\cdot,\cdot) \quad [0,0,0,1,\underset{\uparrow}{2},1,2,2,1,0]$$
$$(9,8,7,\cdot,6,4,5,\cdot,\cdot,\cdot) \quad [0,0,0,1,2,\underset{\uparrow}{1},2,2,1,0]$$
$$(9,8,7,\cdot,6,4,5,\cdot,3,\cdot) \quad [0,0,0,1,2,1,\underset{\uparrow}{2},2,1,0]$$
$$(9,8,7,\cdot,6,4,5,\cdot,3,2) \quad [0,0,0,1,2,1,2,\underset{\uparrow}{2},1,0]$$
$$(9,8,7,\cdot,6,4,5,1,3,2) \quad [0,0,0,1,2,1,2,2,\underset{\uparrow}{1},0]$$
$$(9,8,7,0,6,4,5,1,3,2) \quad [0,0,0,1,2,1,2,2,1,\underset{\uparrow}{0}]$$

To place digit 5 we need to consider the third empty slot as indicated by index 2. For completeness we added the remaining steps of the algorithm. The mapping from positional array to permutation is bijective. Note that mathematically speaking filling the permutation slots from the right is an equally valid strategy and results in a mirror version of the permutation we obtained here. The choice to index from the left is just to adhere to convention.

An example of the custom PIN pad configuration is shown in Figure 2.6. The Asset UUID in the example, 'f80b944b-1c2a-459e-a55f-6004**42b9b3ed**' is displayed in hexadecimal representation (i.e. base-16), where the least-signifcant 32-bits have been boldfaced. Since the permutation is linked to the Asset UUID, the layout remains the same across multiple transactions for the same Asset.

Customization can be extended even further. The relative size between the buttons can be modified. Another extension is to consider different shapes for buttons, like an 'L' shape for example. This can allow for more specific behaviour to be learnt, which ideally translates to a more distinct keystroke profile. In practice, deviating too much from the standard will result in confusion or people not taking the application seriously. Thus, in this thesis we investigate both the default PIN pad and custom PIN pad permutation derived from the Asset UUID.

[3] By default we consider index 0 the first slot as is common practice in many computer languages

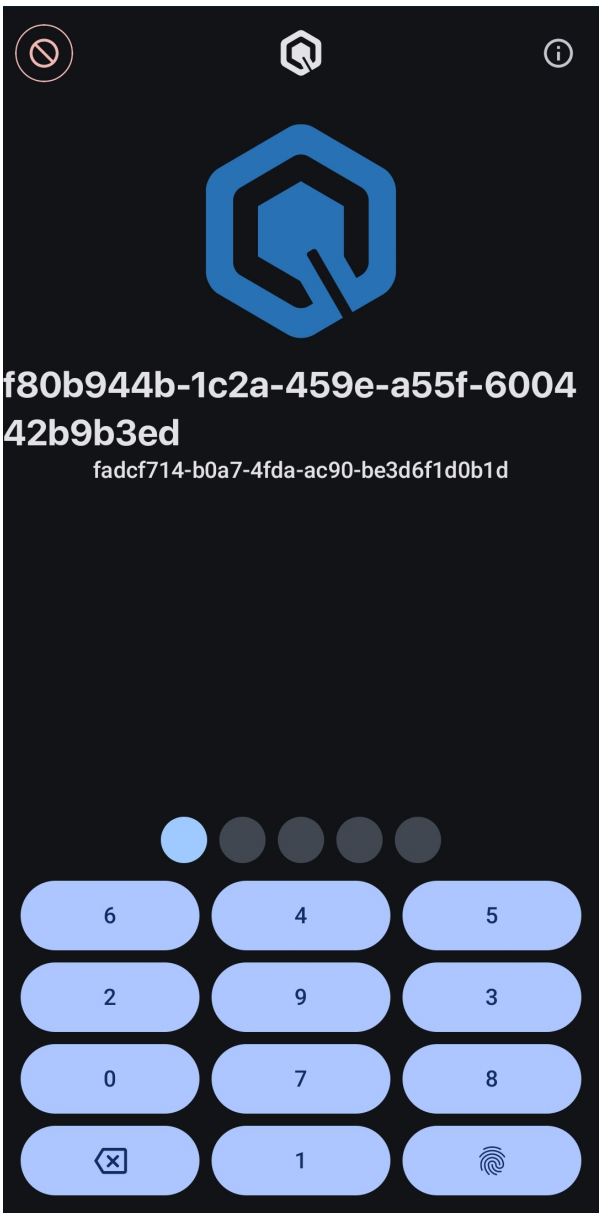


Figure 2.6: Ubiqu's customized PIN pad authentication screen corresponding to the Asset UUID ('f80b944b-1c2a-459e-a55f-600442b9b3ed'). Since the permuation is linked to the Asset UUID, the layout remains the same across multiple transactions for the same Asset.

3

User profile modelling

In the previous chapter we have quantified the behaviour of PIN entries by means of keystroke dynamics. Furthermore we abstracted the behavioural log data into (behavioural) vectors[1]. Since we cannot make assumptions on other user data our problem is a so-called one-class classification problem [35]. In this chapter we focus on building tunable one-class classification (OCC) methods to capture user keystroke profiles using that behavioural data. In general this is a hard task, in the sense that a priori it is not clear which model is most suitable.

We define a user keystroke profile P to be the mathematical representation of the learnt behaviour. For convenience we shall often omit the 'keystroke' adjective and write *user profile* or simply *profile*. More specifically P can be seen as some unknown distribution from which the PIN entry data U is sampled. Using one-class classification methods we construct an user profile approximation $\hat{P}(U)$. Since we assume to be dealing with learnt stable behaviour (i.e. P), we expect data samples of large enough size to be approximately normally distributed, regardless of the true distribution of the behaviour. This important fact is a consequence of the Central Limit Theory for our data samples [22]. This fact also motivates the choices for the three OCC methods used in this thesis to model user keystroke profiles.

Let us assume user data U to be a sample of some unknown stable behaviour distribution P. For authentication purposes we want to minimise false acceptance rates (FAR), hence model the user profile minimally. On the other hand we do not want to burden users with high false rejection rates. Thus we need to strike a balance between classifying a user keystroke profile fully (i.e. maximising true acceptance rates) and minimising false acceptance rates (FAR). One-class classification methods are inherently tuneable which allows us to strike such a balance.

First we motivate our selection of one-class classification methods. In the subsequent three section we elaborate on each of the chosen classification methods in turn.

3.1. One-class classification methods

Let us define a classifier, which is a function assigning a label to any data point provided.

Definition 3.1.1: Classifier [56]

Let $f : \mathcal{X} \to \mathcal{Y}$ be a classification function, i.e. classifier, such that for a given feature vector **x** an estimate of the label is obtained. For binary classifiers we usually either take $\mathcal{Y} = \{-1, 1\}$ or $\mathcal{Y} = \{0, 1\}$.

Where multi-class classifiers are generally a form of supervised learning, one-class classifiers fall into the unsupervised learning category [9]. In contrast to multi-class classification task, one-class classification methods need to split data in two groups; 'interior' and 'exterior'. The analogy here to user profile is such: 'interior' data points are considered correct, while 'exterior' data points are considered incorrect outliers [32]. Even though these "outliers" are part of the user data set, they are data points

[1] Throughout this chapter we use the terms *behavioural vectors* and *data points* interchangeably

considered too far deviating from the stable behaviour and thus inflating the profile unnecessarily. One-class classification models are tunable, meaning that the ratio between interior and exterior points, and thus the volume of $\hat{P}(U)$, can be regulated using a (hyper)parameter. While the general OCC model $f(\mathbf{x}, \mathbf{w})$ allows for higher dimensional hyperparameters [56], in this research we only consider one-dimensional hyperparameters for tuning the one-class classifiers.

One-class classification models can be categorized into three groups; boundary methods, density estimation methods and reconstruction methods [57]. In this research we analyse three different methods:

- k-Minimal Enclosing Ball (k-MEB)
- Support Vector Data Description (SVDD)
- Multivariate Gaussian Estimation (MGE)

The first method we will describe in Section 3.2 is k-Minimal Enclosing Ball (k-MEB) and belongs to the boundary method category. The second method we investigate, Support Vector Data Description (SVDD), also belongs to the boundary method category. The third and final method is Multivariate Gaussian Estimation (MGE) and falls into the density estimation category of OCC models. SVDD and MGE are explained in Section 3.3 and Section 3.4 respectively. First we formalise the boundary method one-class classifiers.

3.1.1. Boundary one-class classifiers

In the case of boundary one-class classifiers a data set U is partitioned into two sets; C and $U \setminus C$. We refer to C as the *core* and $U \setminus C$ as *outliers*. Then a decision boundary separates these sets and defines a closed set $\varphi(U, \gamma)$ which must include C, known as a decision region. The shape and definition of the decision region is dependent on the boundary OCC of choice and hyperparameter $\gamma \in \mathbb{R}$.

Definition 3.1.2: Boundary one-class classifier

Let $U \subset \mathcal{X}$ be a data set. Then $f_U : \mathcal{X} \times \mathbb{R} \to \mathcal{Y} = \{0, 1\}$ is a boundary one-class classifier given by

$$f_U(\mathbf{x}, \gamma) := \begin{cases} 1 & \text{if} \quad \mathbf{x} \in \varphi_{\mathcal{M}}(U, \gamma) \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

with $\varphi_{\mathcal{M}}(U, \gamma) \subset \mathcal{X}$ the decision region for a given boundary method $\mathcal{M}$.

In this research we shall solely focus on hyperellipsoidal decision regions. This implies we construct a hyperellipsoid[2] $\mathcal{E}_U$ which estimates the true (unknown) user profile P. In other words, we take $\hat{P}(U) = \mathcal{E}_U$. To construct $\mathcal{E}_U$ from U we first normalise the user data set U, using affine transformation T, to obtain $Z = T(U)$. On this normalised data we apply $\varphi(Z, \gamma)$ to obtain a decision region. In our case, both for SVDD and k-MEB, we obtain a spherical decision region described by a centre-radius pair (c, r). From this sphere we can derive matrix $\mathbf{B}$ representing a orthogonal basis of choice. A common choice is to take standard basis which correspond to the features. Another choice would be to take the basis obtained through principal component analysis. The final step to construct the (hyperellipsoid) decision region is to revert the normalisation by applying T^{-1}. The described mathematical formalisation is visualised in Figure 3.1.

More accurately, we have $U \subset \mathbb{R}^d$ a finite set of n data points (i.e. feature vectors) and take $X \in \mathbb{R}^{n \times d}$ as the corresponding user data matrix. The standardised user data matrix $Z = T(X)$ we obtain by standardising for each feature (column) x_i with

$$T(x_i) = \frac{x_i - \overline{x_i}}{\sigma(x_i)} \tag{3.2}$$

Consider a one-class classifier method $\varphi : \mathbb{R}^{n \times d} \times \mathbb{R} \to \mathbb{R}^d \times \mathbb{R}$, given by $\varphi(Z, \gamma) = (c, r)$ where $c \in \mathbb{R}^d$ is the hypersphere centre and $r \in \mathbb{R}$ the radius. Then the user profile hyperellipsoid is described by the matrix

$$A = T^{-1}(rI + \mathbf{1}c^T) \tag{3.3}$$

The matrix A can be written in turn as a hyperellipsoid in its quadric form, which we denoted $\mathcal{E}_U$.

[2] See Definition 4.1.2.

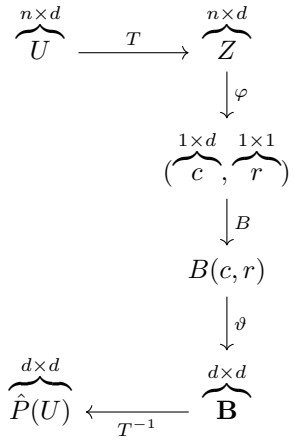


Figure 3.1: Diagram for building a user profile $\hat{P}(U)$, from a user data sample U, using boundary one-class classification methods in order to obtain an ellipsoidal decision region description.

3.1.2. Density estimation one-class classifiers

For the density estimation methods a decision boundary follows from the condition whether the probability of a feature vector **x** exceeds a certain threshold or not.

Definition 3.1.3: Density estimation one-class classifier

Let $U \subset \mathcal{X}$ be a data set. Then $f_U : \mathcal{X} \times \mathbb{R} \to \mathcal{Y} = \{0, 1\}$ is a density estimation one-class classifier given by

$$f_U(\mathbf{x}, t) := \begin{cases} 1 & \text{if} \quad p(\mathbf{x}|U) \geq t \\ 0 & \text{otherwise} \end{cases} \tag{3.4}$$

with $p(\cdot|U) \sim D$ for some distribution D.

3.2. k-Minimum Enclosing Ball Classification

First we will cover the Minimum Enclosing Ball method (MEB), also known as 'Bounding Sphere' method. We apply MEB both directly on the data as well as after standardisation. Then, we consider a tuneable extension known as the k-Minimum Enclosing Ball (k-MEB). The k-MEB method is a decision boundary one-class classifier. An alternative to such bounding (hyper)sphere approach is the (hyper)rectangle learner [34], which is less sensitive to magnitude differences between features. Moreover, since user data is a sample, creating a convex polytope as convex hull of all data points is undesirable as it is too rigid. k-MEB does naturally encompass the Gaussian assumption for sampling [16].

3.2.1. Minimum Enclosing Ball

First we shall provide a mathematical definition for a Minimum Enclosing Ball. Let U be a sample of profile data consisting of n points in a d-dimensional feature space, i.e. $U = \{u_1, \ldots, u_n\} \subset \mathbb{R}^d$. A d-dimensional ball is defined as

Definition 3.2.1: d-Dimensional Ball

Let $c \in \mathbb{R}^d$ be the centre of the ball and $r \in \mathbb{R}_{\geq 0}$ the radius, then the **ball** is given by

$$B(c, r) := \{x \in \mathbb{R}^d : \|x - c\|_2 \leq r\} \tag{3.5}$$

Then, an enclosing ball is given by

Definition 3.2.2: Enclosing Ball

Let $U = \{u_1, \ldots, u_n\}$ a finite set of points with $u_i \in \mathbb{R}^d \, \forall i \in [n]$ and let $B(c, r)$ be a ball. Then the ball is an **enclosing ball** if

$$U \subseteq B(c, r) \tag{3.6}$$

Moreover, the collection of Enclosing Balls given U, denoted as $\mathcal{B}$, is defined as

$$\mathcal{B}(U) := \{B(c, r) : U \subseteq B(c, r)\} \tag{3.7}$$

We remark that we rightfully have written *the* collection of enclosing balls as it is unique [31]. From this collection of enclosing balls the definition of a minimum enclosing ball simply follows

Definition 3.2.3: Minimum Enclosing Ball

Let $U \subset \mathbb{R}^d$ be a finite set of points and $\mathcal{B}_U$ a collection of enclosing balls, then we define the Minimum Enclosing Ball given U by

$$\mathrm{MEB}(U) := \arg\min_{\mathrm{vol}} \mathcal{B}_U = \arg\min\{\mathrm{vol}\,(B(c, r)) : U \subseteq B(c, r)\} \tag{3.8}$$

Now, by definition a minimum enclosing ball (MEB) captures the data sample of the stable behaviour fully [31]. Characterised by points on the boundary of the enclosing ball with minimum radius, we could safely write $\arg\min$ in stead of $\arg\inf$, as the minimum is always attained. Moreover, a MEB is unique [65], hence one can refer to *the* MEB. In other words, all data points of the data set are correctly classified as part of user U's behaviour profile, since they lie inside, or on the boundary/surface of the enclosing ball. A minimum enclosing ball is also referred to as smallest bounding sphere and various other terms [63]. Let us formalise this point categorisation

Definition 3.2.4: Interior and boundary points

Let $u_1 \in U$ be a point, then u_1 is an **interior point** of $\mathrm{MEB}(U)$ if

$$u_1 \in \mathrm{Int}(\mathrm{MEB}(U)) \quad \text{or equivalently} \tag{3.9}$$
$$\|u_1 - c\| < r \tag{3.10}$$

where c is the centre and r the radius of $\mathrm{MEB}(U)$.
For a point u_2 to be a **boundary point** of $\mathrm{MEB}(U)$, it must hold

$$u_2 \in \delta(\mathrm{MEB}(U)) \quad \text{or equivalently} \tag{3.11}$$
$$\|u_2 - c\| = r \tag{3.12}$$

We shall refer to the set of all boundary points of an enclosing ball as the boundary (support) set, denoted by T. In this thesis we can safely assume the data samples to be large enough, i.e. $n > d$. Since $|T| \approx d$ we conclude T to be a proper subset of U.

Computing the MEB for an arbitrary set of points $P \subset \mathbb{R}^d$ is no trivial task. In 1991, Welzl proposed a recursive algorithm which randomly adds points to the boundary support set and inflates the ball accordingly, dropping points from the support set if possible [65]. In 2003, Fischer, Gärtner and Kutz proposed an iterative algorithm which starts with an initial enclosing ball and shrinks the radius whilst moving the ball's centre towards centre of the affine support plane [23]. Alternatively, MEB can be formulated as a convex optimisation problem in an intersection of n half-spaces in $d + 1$ [16]. For our MEB classification we developed a Julia implementation of the algorithm by Fischer, Gärtner and Kutz. Similar to Fischer et al and Welzl we shall keep track of the boundary support set, denoted T, at each iteration. Before we analyse the results of the MEB classifier, applied to the user profile data (see Subsection 3.2.1), we shall discuss this algorithm.

MEB algorithm

Let $U = \{u_1, \ldots, u_n\}$ be user profile data. First we compute the initial enclosing ball $B(c_0, r_0)$ starting ball, where $c_0 = \frac{1}{n}\sum_{i=1}^{n} u_i$, i.e. the average of our set of points U. The initial radius r_0 is found by computing

$$r_0 := \max_{u \in U}\{\|c_0 - u\|\} \tag{3.13}$$

The point(s) resulting in this maximum distance to the initial centre are added to the boundary set. Given the dimension-feature ratio and floating point values, with high probability we assume only one point is now added to the boundary set. Now the algorithm iterates between two phases until a termination condition is met

1. A *dropping phase*
2. A *walking phase*

Before we explain both phases, we provide the termination condition of the algorithm. We terminate when the current centre c lies in the convex hull of boundary support set T. If this condition is met, no points can be dropped in the dropping phase, due to the definition of convex combination. Neither is walking possible since c coincides with its projection on $\mathrm{aff}(T)$.

More details for the dropping phase will now follow. For each iteration i first the dropping phase commences. Given the boundary point set $T = \{t_1, \ldots, t_m\} \subseteq U$ at some iteration i. First we check if the centre of the ball at this iteration lies on the affine plane spanned by T. Let us provide a notion for the supporting hyperplane of a boundary point set T.

Definition 3.2.5: Supporting Hyperplane

Let $T = \{t_1, \ldots, t_m\}$ be the set of boundary points of an enclosing ball for user profile $U = \{u_1, \ldots, u_n\} \subset \mathbb{R}^d$. We assume the points of T are not collinear. Then, if $m \leq d$, the **supporting hyperplane**, denoted $\mathrm{aff}(T)$ can be described by the parametric plane equation

$$\mathrm{aff}(T) : t_1 + x_1(t_2 - t_1) + x_2(t_3 - t_1) + \ldots + x_{m-1}(t_m - t_1) \tag{3.14}$$

with parameters $x_1, \ldots, x_{m-1}$ under the condition $\sum_{k=1}^{m-1} x_k = 1$.

Let c be the current centre of the ball. Then, if $c \in \mathrm{aff}(T)$, we find the affine combination of vectors $(t_2 - t_1), \ldots, (t_m - t_1)$ that make up $c - t_1$. To keep the notation simpler, take $v_{j-1} = (t_j - t_1)\, \forall j \in \{2, \ldots, m\}$ and then the parametric supporting hyperplane equation 3.14 becomes

$$t_1 + x_1 v_1 + x_2 v_2 \ldots x_{m-1} v_{m-1} \tag{3.15}$$

and the affine combination is expressed as the system of equations

$$\begin{cases} x_1 v_1 + x_2 v_2 \ldots x_{m-1} v_{m-1} = (c - t_1) \\ x_1 + \ldots + x_{m-1} = 1 \end{cases} \tag{3.16}$$

It is reasonable to assume these vectors v_j are linearly independent since the features are floating point values are points are unlikely to coincide. This assume allows for a unique solution of the affine combination to express centre c. Writing this in matrix-vector notation we obtain the following system

$$\begin{bmatrix} (v_1)_1 & (v_2)_1 & \cdots & (v_{m-1})_1 \\ (v_1)_2 & (v_2)_2 & \cdots & (v_{m-1})_2 \\ \vdots & \vdots & \ddots & \vdots \\ (v_1)_d & (v_2)_d & \cdots & (v_{m-1})_d \\ 1 & 1 & \cdots & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{m-1} \end{bmatrix} = \begin{bmatrix} (c - t_1)_1 \\ (c - t_1)_2 \\ \vdots \\ (c - t_1)_d \\ 1 \end{bmatrix} \tag{3.17}$$

where $(v_j)_b$ marks the b-th coordinate of the j-th support plane vector and d the feature space dimension. Now the dropping is removing all points t_j from T which have a negative coefficient in the affine combination to the centre. Therefor, we can update T as follows

$$T \leftarrow T \setminus \{t_j \in T : x_{j-1} < 0\} \tag{3.18}$$

Algorithm 1 (Adaptation of) Smallest Enclosing Ball Algorithm by Fischer, Gärtner and Kutz [23]

Require: $|U| > 0$
$n \leftarrow |U|$
$c \leftarrow \frac{1}{n}\sum_{i=1}^{n} u_i$
$T \leftarrow \{\arg\max_{u\in U}\{\|c-u\|\}\}$
$r \leftarrow \max_{u\in U}\{\|c-u\|\}$

while $c \notin \mathrm{conv}(T)$ **do**
  if $c \in \mathrm{aff}(T)$ **then** ▷ Condition for dropping phase
    $m \leftarrow |T|$
    $v_j \leftarrow t_{j+1} - t_1 \quad \forall j \in [m-1]$
    $(c - t_1) = \sum_{j=1}^{m-1} x_j v_j$
    $T \leftarrow T \setminus \{t_{j+1} : j \in \{j : x_j < 0\}\}$
  end if
  $\varepsilon_j \leftarrow \max\left\{\frac{r^2 - \|\vec{s}_j\|^2}{2(r\|\vec{d}\| - \vec{s}_j\cdot\vec{d})}, 0\right\} \quad \forall j \in (U \setminus T)$ ▷ Start walking phase
  $\varepsilon \leftarrow \min\{1, \min_{j\in(U\setminus T)}\{\varepsilon_j\}\}$
  $w \leftarrow \arg\min_{j\in(U\setminus T)}\{\varepsilon_j : \varepsilon_j > 0\}\}$
  $T \leftarrow T \cup \{w\}$
  $\vec{n} \leftarrow \mathrm{proj}_{\mathrm{aff}(T)}(c) - c$
  $c \leftarrow c + \varepsilon \cdot \vec{n}$
  $r \leftarrow \|c - w\|$
end while
return c, r, T

Now during the *walking phase* the current centre c_i is moved towards the supporting hyperplane $\mathrm{aff}(T)$ orthogonally, until an interior point becomes a boundary point. In other words, c_i is moving towards $\mathrm{proj}_{\mathrm{aff}(T)}(c_i)$[3] (i.e. its projection on the supporting hyperplane).

Since the projection is orthogonal we have to solve the following system to find $\mathrm{proj}_{\mathrm{aff}(T)}(c_i)$

$$[t_1 + x_1 v_1 + x_2 v_2 \ldots x_{m-1} v_{m-1} - c_i] \cdot v_j = 0 \quad \forall j \in [m-1] \tag{3.19}$$

which can also be written as

$$\begin{aligned} t_1 v_j + x_1 v_1 \cdot v_j + \ldots + x_{m-1} v_{m-1} \cdot v_j - c_i v_j = 0 \quad \forall j \in [m-1] \quad \text{or} \\ x_1 v_1 \cdot v_j + \ldots + x_{m-1} v_{m-1} \cdot v_j = (c_i - t_1) \cdot v_j \quad \forall j \in [m-1] \end{aligned}$$

and in matrix-vector notation

$$\begin{bmatrix} \langle v_1 \mid v_1\rangle & \ldots & \langle v_{m-1} \mid v_1\rangle \\ \vdots & \ddots & \vdots \\ \langle v_1 \mid v_{m-1}\rangle & \ldots & \langle v_{m-1} \mid v_{m-1}\rangle \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_{m-1} \end{bmatrix} = \begin{bmatrix} (c_i - t_1)\cdot v_1 \\ \vdots \\ (c_i - t_1)\cdot v_{m-1} \end{bmatrix} \tag{3.20}$$

where $\langle v \mid v'\rangle$ denotes the dot product of vectors v and v'. The vectors v_j are real-valued, hence $\langle v \mid v'\rangle = \langle v' \mid v\rangle$, which implies this matrix of dot products is a real symmetric matrix. Acknowledging this fact, we can also rewrite the equation as

$$\begin{bmatrix} v_1 & \ldots & v_{m-1} \end{bmatrix}^\top \begin{bmatrix} v_1 & \ldots & v_{m-1} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_{m-1} \end{bmatrix} = \begin{bmatrix} v_1 & \ldots & v_{m-1} \end{bmatrix}^\top w \tag{3.21}$$

where we recall v_j to be a column vector and take $w = (c_i - t_1)$. Solving for $x_1, \ldots, x_{m-1}$, and substituting them in the parametric hyperplane equation, provides us with the desired coordinates of the projection of c_i. Hence the direction to 'walk' towards is given by the vector $\mathrm{proj}_{\mathrm{aff}(T)}(c_i) - c_i$.

[3] In Fischer et al this is denoted by $\mathrm{cc}(T)$

With the direction to the affine hyperplane known, we need to know how far we can walk before a interior point becomes a boundary point. So for each point u_j in the set $U \setminus T$ we calculate the scalar until it turns into a boundary point. We quickly derive this scalar, denoted ε, using a 3D example shown in Figure 3.2.

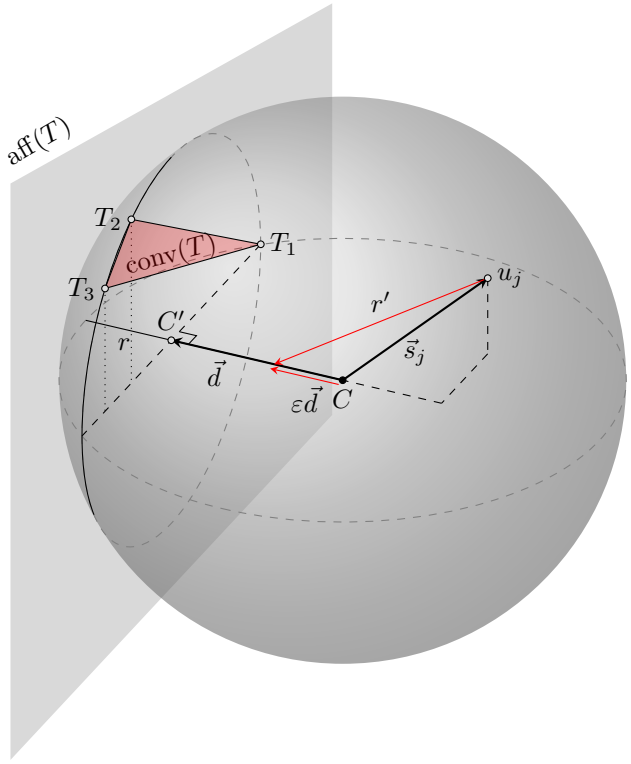


Figure 3.2: Visualisation of the *walking phase* at some iteration i of the MEB algorithm by Fischer et al. Assume at this iteration point C is the center of ball and r the radius. Moreover, the boundary support set is $T = \{T_1, T_2, T_3\}$. Consider the point $u_j \in U \setminus T$ and determine its limitation on scalar ε to move C towards C'.

Assume the MEB algorithm is at some iteration i, with point C is the centre of ball and r the radius. Moreover, the current boundary support set is given by $T = \{T_1, T_2, T_3\}$. Now consider the point $u_j \in U \setminus T$ and we shall determine its limitation on scalar ε to move C towards C'. First we denote $u_j - C$ by vector notation

$$\vec{s}_j = u_j - C \tag{3.22}$$

Now, from Figure 3.2 we observe the following relation

$$r' = \varepsilon\vec{d} - \vec{s}_j \tag{3.23}$$

This implies, taking the inner product of both sides with respect to themselves, we have

$$r' \cdot r' = (\varepsilon\vec{d} - \vec{s}_j) \cdot (\varepsilon\vec{d} - \vec{s}_j) \tag{3.24}$$

When moving, the radius (vector) shrinks depending on ε according to

$$r(\varepsilon) = r - \varepsilon\|\vec{d}\| \tag{3.25}$$

Thus, when $r' = r(\varepsilon)$, point u_j has become a boundary point and the corresponding value for ε is the restriction.

Solving for ε gives

$$\begin{aligned}
r' \cdot r' &= (\varepsilon\vec{d} - \vec{s}_j) \cdot (\varepsilon\vec{d} - \vec{s}_j) \\
\left(r - \varepsilon\|\vec{d}\|\right) \cdot \left(r - \varepsilon\|\vec{d}\|\right) &= (\vec{d} - \vec{s}_j) \cdot (\vec{d} - \vec{s}_j) \\
r^2 - 2r\|\vec{d}\|\varepsilon + \varepsilon^2\|\vec{d}\|^2 &= \varepsilon^2\|\vec{d}\|^2 - 2\left(\vec{s}_j \cdot \vec{d}\right)\varepsilon + \|\vec{s}_j\|^2 \\
r^2 - \|\vec{s}_j\|^2 &= 2\left(r\|\vec{d} - \vec{s}_j \cdot \vec{d}\right)\varepsilon
\end{aligned}$$

We noticed the $\varepsilon^2\|\vec{d}\|^2$ term cancels on both sides and then simply rearranged terms to find

$$\varepsilon = \frac{r^2 - \|\vec{s}_j\|^2}{2\left(r\|\vec{d}\| - \vec{s}_j \cdot \vec{d}\right)} \tag{3.26}$$

To keep track of which ε limits our moment due to point u_j at iteration i we use the notation ε_j^i. This allows us to denote the minimum of ε^i for all points $u_j \in (U \setminus T)$, which yields us the maximum walking distance

$$\varepsilon^i := \min\{1, \min_j\{\varepsilon_j^i\}\} \tag{3.27}$$

The additional 1 is to make sure we never walk further than our desired new centre, which we recall as the projection of the current centre on the affine plane of the boundary set. Meaning, for the example in Figure 3.2, if $\varepsilon^i = 1$ we could walk all the way to from C to C' without interior points hindering this. In any case, the new centre is given by

$$c_{i+i} := c_i + \varepsilon^i \vec{d} \tag{3.28}$$

Thus, we find the new ball $B^{i+1}(U)$, after the walking phase of iteration i, to be

$$B^{i+1}(U) := B\left(c_{i+1}, r^i - \varepsilon^i\|\vec{d}^i\|\right) \tag{3.29}$$

where r^i denotes the radius at iteration i before the walking phase and $\vec{d^i}$ the direction vector from c_i to its projection on $\mathrm{aff}(T)$. If $c_{i+1} \in \mathrm{conv}(T)$ we terminate the algorithm, otherwise we move to the dropping phase of iteration $i+1$. This concludes the in-depth explanation of the MEB algorithm as proposed by Fischer et al [23].

MEB feature scaling

Computing the MEB directly for the data is not desirable, since due to the difference in order of magnitude of the features, this unnecessary enlarges the classification region, increasing FAR. For example, computing the bounding sphere, for features[4] `keypress-hold-time`, `relative-touch-coordinate-y` and `touch-pressure` for the first digit from the data set A suffers from this, as depicted in Figure 3.3a.

One solution is to prevent an unnecessary large classification region is using a Minimum-Volume Enclosing Ellipsoid method directly [31]. Another solution is to standardise the features. This implies first centring the data around to origin by translation and then scaling the data columns (features) to unit vectors. Only centring the user data is not sufficient, because features with largest variance will then define the diameter of the bounding sphere. These two actions combined essentially transform a (hyper)ellipsoid into a (hyper)sphere. To inverse the transformation we need to store both the translation, also known as off-set, and scalars.

With the mathematical foundation of MEB secure, we can expand this method to k-MEB. After all, MEB itself is not an OCC but can be used as a building block. One example would be; drawing samples from the user data, computing MEB for all such samples and then taking the intersection of these MEBs. A deterministic variant of this is known in literature as Minimum k-Enclosing Ball (k-MEB).

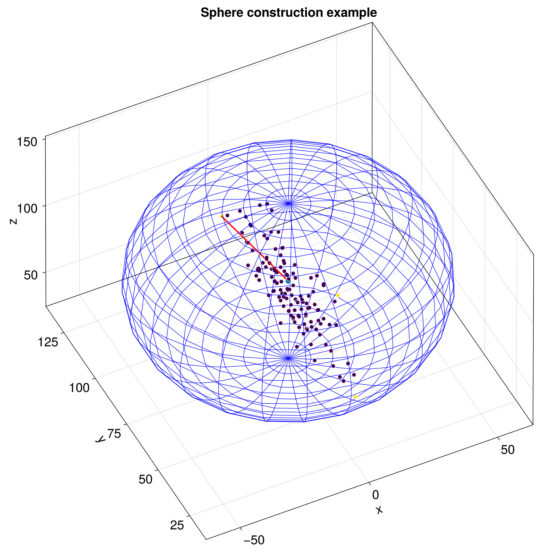
3.2.2. Minimum k-Enclosing Ball

To build an OCC from the minimum enclosing ball method, we can use k-minimum enclosing ball, often denoted k-MEB[5] or MB_k [15]. The k-MEB method finds the smallest MEB of all subsets of U with size k. In other words, given n points in $\mathbb{R}^d$, $(n-k)$ points are disregarded in finding the bounding sphere. These disregarded points will be referred to as **exterior point**s. This of course raises the question which value for k, also known as a hyper-parameter, is most suitable. Often k is chosen as a fixed percentage of n.

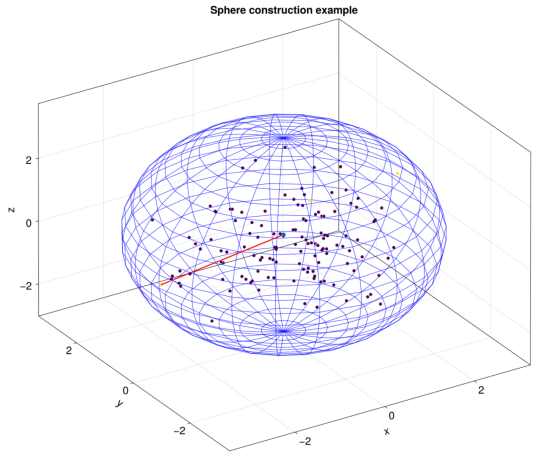
While k-MEB is known to be NP-hard in the strong sense [53], there are several methods which efficiently deal with small to medium data sets [15]. Then, given n points in $\mathbb{R}^d$, $(n-k)$ exceptions are

[4] Limiting to three features for the sake of visualising the sphere.

[5] Not to be confused with K-MEB, which is an abbreviation for the problem of finding K enclosing balls to cover a data set [54]



(a) Minimum Enclosing Ball without standardising features. Since touch pressure is a value in $[0, 1]$, while touch pressure in milliseconds ranges between 50 and 150, the Minimum Enclosing Ball could not shrink further.



(b) Minimum Enclosing Ball with standardising features. Due to standardisation, the volume of the Minimum Enclosing Ball $\mathrm{MEB}(U)$ relates more closely to $\mathrm{conv}(U)$

Figure 3.3: Comparing the MEB on data set A based on three selected features both before and after standardisation on data set. In both visuals we have; x = touch pressure, y = relative touch y-coordinate, z = keypress hold time

disregarded in finding the bounding sphere. This of course raises the question which value for k, also known as a hyper-parameter, is most suitable. Often k is chosen as a fixed percentage of n.

Definition 3.2.6: Minimum k-Enclosing Ball

Let $U \subset \mathbb{R}^d$ be a finite set of points with $|U| = n$, then a Minimum k-Enclosing Ball is given by, with $1 \leq k \leq n$

$$\mathrm{MEB}_k(U) := \arg\min_r \{B(c, r) : W \subseteq B(c, r) \cap U \wedge |W| \geq k\} \tag{3.30}$$

In order to compute the k-MEB we developed an adaptation to the branch-and-bound algorithm proposed by Cavaleiro and Alizadeh [15]. First compute the initial MEB, with the centre as average of all points and radius as distance from average centre to furthest point. Then sort the points from furthest to nearest to the MEB centre and only take k nearest to compute a MEB as a base line radius. To speed up the search, we build a minimum solution search tree in order of distance so pruning is most likely to occur and less branches have to be investigated.

Let us apply this principle to our Data Set C. Even after standardising, still some extreme deviations in the feature space might still severely inflate the MEB. To illustrate this, we computed the volume ratio between MEB and k-MEB on Data Set C for various values of k. In Figure 3.4 it can be seen that extreme deviations can blow up the MEB's volume substantially, which we consider undesirable. The sharp decrease in volume by only decreasing k slightly motivates our claim, this is seen in the steep part of the graph at the right-most section of Figure 3.4.

Moreover, in Figure 3.5, the k-MEB is visualised for four different thresholds, where k is taken to be a percentage of total available data points in the stable behaviour user data set. Meaning $k = \mathrm{ceil}(T \cdot |U_{\text{stable}}|)$ for some threshold value $T \in (0, 1]$. When $T = 1$ the original MEB is obtained. The threshold allows us to 'tune' the strictness of the classifier by reducing/enlarging the classification volume, as desired.

In order to compare profiles the profile defining hyperspheres needs to be transformed back into

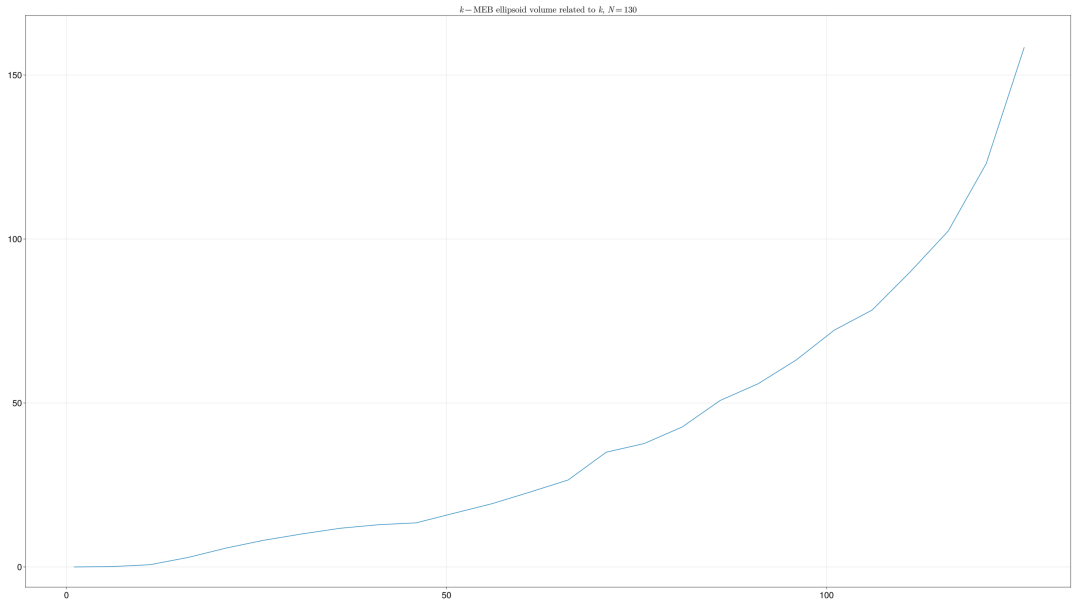
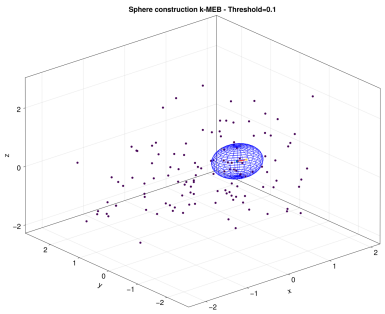
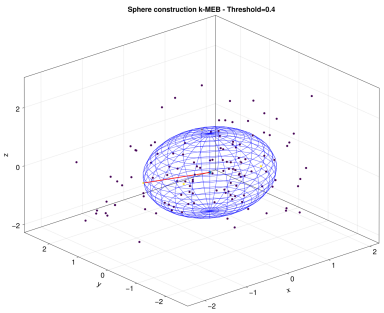


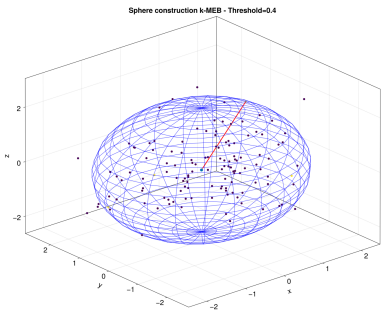
Figure 3.4: Relation between ellipsoidal volume of k-MEB method and k (i.e. number of points included)



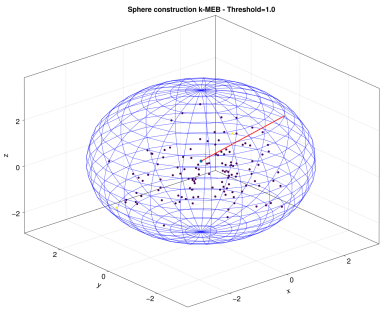
(a) $k = 0.1 \cdot |U_{\text{stable}}| = 13$



(b) $k = 52$



(c) $k = 111$



(d) $k = 130$, remark this is precisely MEB

Figure 3.5: Visualisation of k-MEB applied to the standardised Data Set C for different thresholds $\{0.1, 0.4, 0.85, 1\}$ for k as a percentage of the stable user profile.

ellipsoids. In Figure 3.6 we see how the spherical boundary morphs into a ellipsoid upon inverse transformation. We have formally defined this in Definition 3.1.1. The hyperellipsoid $\mathcal{E}_U$ estimates the unknown user profile distribution P.

While k-MEB allows us to control exactly how many data points are excluded from the sample to build a user profile, it is computationally expensive. Moreover, depending on the variation of the sample data we would like to exclude more/less data points to capture the user profile more accurately. To provide a more general notion of exclusion we turn to another outlier One-Class Classification method

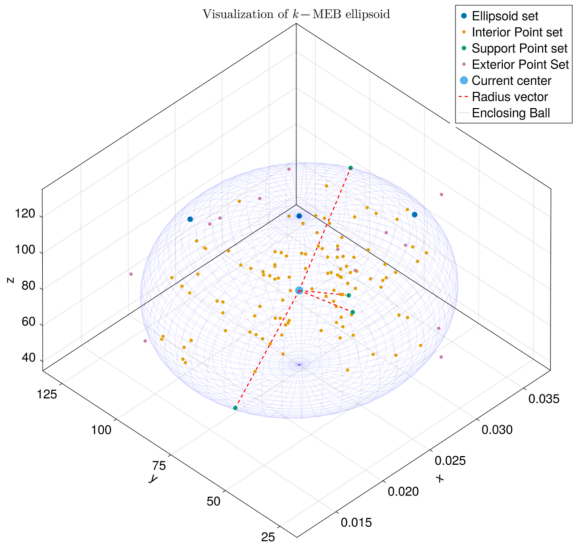


Figure 3.6: 3D visualisation of the k-MEB decision boundary after inverse transformation for data set C.

called ‘SVDD’.

3.3. One-class Support Vector Machines

Another way of creating a tuneable 'hard'-boundary classifier is using One-Class Support Vector Machines (OC-SVM) [50]. When using a Gaussian kernel, OCSVM can equivalently be solved as a Support Vector Data Description (SVDD) problem [10]. Tax and Duin, proposed in 2004 a method to obtain a spherically shaped (decision) boundary around a data set [58]. Similarly to regular SVM, minimizing the volume is rewarded, but excluding points decreases the objective value, which results in a balance only discarding outliers. In [58], this minimization problem is formulated as

$$\begin{aligned} \text{minimize } & F(c,r) = r^2 + C\sum_{i=1}^{n}\xi_i \\ \text{subject to } & \|u_i - c\|^2 \leq r^2 + \xi_i \quad \forall i \in [n] \\ & \xi_i \geq 0 \quad \forall i \in [n] \end{aligned} \tag{3.31}$$

Tax and Duin showed that

$$\begin{aligned} \text{maximize } & L(\boldsymbol{\alpha}) = \sum_{i=1}^{n}\alpha_i\,(u_i \cdot u_i) - \sum_{i,j}^{n}\alpha_i\alpha_j\,(u_i \cdot u_j) \\ \text{subject to } & \\ & 0 \leq \alpha_i \leq C \quad \forall i \in [n] \\ & \sum \alpha_i = 1 \end{aligned} \tag{3.32}$$

is the Lagrangian dual formulation to Problem (3.31), where $c = \sum_{i=1}^{n}\alpha_i u_i$ and $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$ are the decision variables. The radius can be derived by computing $r^2 = \|u_i - c\|^2$ where $i \in \{i : 0 < \alpha_i < C\}$ [58]. The dual formulation, Equation 3.32, explicately shows a kernel function can be used in stead of the regular inner product.

Due to the size difference between features the regular inner product (i.e. Euclidean kernel) does not yield the desired result. Similarly as when applying the k-MEB method standardising the features is required. For k-MEB this behaviour was portrayed in Figure 3.3, whereas the SVDD equivalent is Figure

Analogous to k-MEB, the data points are divided into three groups:

1. Interior Points; $\{u_i \in U : i \in \{i : \alpha_i = 0\}\}$
2. Boundary Points; $\{u_i \in U : i \in \{i : 0 < \alpha_i < C\}\}$
3. Exterior Points; $\{u_i \in U : i \in \{i : \alpha_i = C\}\}$

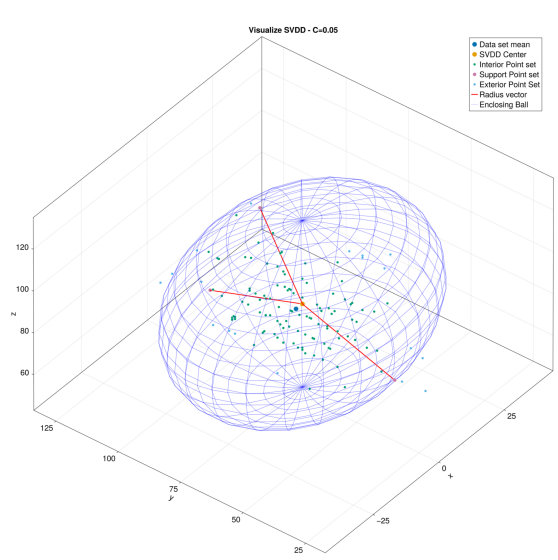
including their respective conditions.

We remark that the SVDD problem is a convex quadratic (i.e. nonlinear) optimisation problem. Due to the number of data points being rather small (i.e. between 100 and 700 points), most solvers find an optimal solution within a short amount of time. Using an interior point method solver for Julia, Ipopt, we find optimal solutions for Equation 3.32. Ipopt is well-supported in Julia and therefore chosen for ease of use. For Ubiqu's Python webservice we use Scipy's `COBYQA` optimisation algorithm, specialised for solving (multivariate) minimisation problems [18].

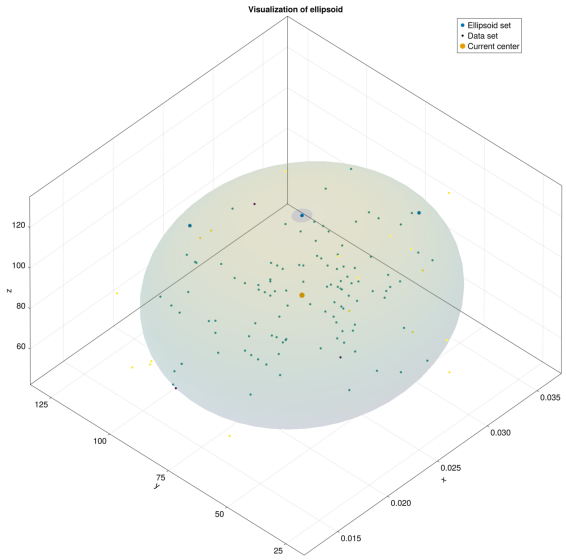
Applying the SVDD method, with Euclidean kernel, on our data set C, we find a sphere shaped boundary as excepted. However, as seen in Figure 3.7, if we do not standardize the features before applying SVDD the sphere radius is dominated by the feature with biggest order. This effect is similarly to what we observed in the previous section with k-MEB. Notice that in Figure 3.7b, where standardisation is applied, the x-axis has a different step-size compared to Figure 3.7b. Even though the shape in Figure 3.7b appears to a sphere, it is actually an ellipsoid.

Instead of maximizing $L(\boldsymbol{\alpha})$, we could also minimise $-L(\boldsymbol{\alpha})$. Then the formulation, with kernel notation, becomes

$$\begin{aligned} \text{minimise } & -\sum_{i=1}^{n}\alpha_i k\,(u_i, u_i) + \sum_{i,j}^{n}\alpha_i\alpha_j k\,(u_i u_j) \\ \text{subject to } & 0 \leq \alpha_i \leq C \quad \forall i \in [n] \\ & \sum \alpha_i = 1 \end{aligned} \tag{3.33}$$



(a) SVDD without standardising features. Since touch pressure is a value in $[0, 1]$, while touch pressure in milliseconds ranges between 50 and 150, the SVDD could not shrink any further.



(b) Due to standardisation, the SVDD fits the data more compactly. Green, yellow and purple points respectively mark interior, exterior and boundary points.

Figure 3.7: Comparing the SVDD, with Euclidean kernel, on data set A based on three selected features both before and after standardisation on data set. In both visuals we have; x = touch pressure, y = relative touch y-coordinate, z = keypress hold time

Under our assumption of a stable behaviour which for large entries we view as being drawn from a multivariate normal distribution, the kernel of choice is the linear kernel [58]. In the case of more exotic boundary shapes radial basis function (RBF) kernel could be considered. The RBF kernel is independent of the position of the data set with respect to the origin [58], since it only utilizes the distances between objects.

Definition 3.3.1: Radial Basis Function Kernel

Let $U \subset \mathbb{R}^d$ be a finite set of points, then we define the RBF kernel as

$$k(u, u') = e^{-\frac{\|u-u'\|^2}{2\sigma^2}} \tag{3.34}$$

for $u, u' \in U$ with a free parameter σ.

The parameter σ originates from the Gaussian Distribution and can flatten or steepen the distribution [49]. For smaller values of σ, the decision boundary is more strict. While for larger values the decision boundary is more flexible. This requires additional hyper-parameter tuning besides our value for C.

Other kernels such as the polynomial kernel $k(u, u') = (u \cdot u')^d$ also provide a hyperparameter to tune, making the SVDD method more flexible. When $d = 1$ we obtain the Euclidean kernel, hence the alias 'linear kernel'. While the flexibility of larger degrees d allows us to fit the data samples more snugly, it is prone to overfit the data.

Given our problem the kernel of choice is the linear kernel for the PIN authentication user profile problem. Pursuing more complex kernels could be considered, but does not match with the nature of the PIN authentication problem. Moreover, given the accuracy of performance achieved by the linear kernel, combined with the shape of the decision boundary in question, pursuing kernels with more degrees of freedom will require more effort to train properly (especially with limited data) and, most likely, will yield little performance gain.

Solving techniques

To actually find a minimizer to 3.33, denoted $(\alpha_1^\star, \ldots, \alpha_n^\star)$, several methods can be employed, such as (stochastic) gradient descent, coordinate descent [1]. Newton's method is a second order method, requiring to compute the inverse of the Hessian matrix, and is therefore computationally expensive [11]. Since we are dealing with small to medium size data sets this is viable. Moreover, we remark our problem given in Equation 3.33 is a convex optimization problem. In particular, Newton's method is well-suited for convex optimization problems [11].

In case of a large data set, coordinate descent would be more suitable [66]. For coordinate descent, we start with some initial values for $\boldsymbol{\alpha}^0 = (\alpha_1^0, \ldots, \alpha_n^0)$ and iteratively update values coordinate-wise, in ordered fashion. The update rule is defined as

$$\alpha_i^{k+1} = \max\{0, \min\{C, \alpha_i^k - \lambda \frac{\partial(-L)}{\partial \alpha_i}(\alpha_1^{k+1}, \ldots, \alpha_{i-1}^{k+1}, \alpha_i^k, \alpha_{i+1}^k, \ldots, \alpha_n^k)\}\} \tag{3.35}$$

where the minimum and maximum ensure the boundary conditions are satisfied. To work out the partial derivate we first rewrite the objective function, given in Equation 3.32, as follows

$$\begin{aligned}
L(\boldsymbol{\alpha}) &= \sum_{i=1}^{n} \alpha_i \left(u_i \cdot u_i\right) - \sum_{i,j}^{n} \alpha_i \alpha_j \left(u_i \cdot u_j\right) \\
&= \sum_{i=1}^{n} \alpha_i \left(u_i \cdot u_i\right) - \sum_{i=j=1}^{n} \alpha_i \alpha_j \left(u_i \cdot u_j\right) - \sum_{i \neq j}^{n} \alpha_i \alpha_j \left(u_i \cdot u_j\right) \\
&= \sum_{i=1}^{n} \alpha_i \left(u_i \cdot u_i\right) - \sum_{i=1}^{n} \alpha_i \alpha_i \left(u_i \cdot u_i\right) - \sum_{i>j}^{n} \alpha_i \alpha_j \left(u_i \cdot u_j\right) - \sum_{i<j}^{n} \alpha_i \alpha_j \left(u_i \cdot u_j\right) \\
&= \sum_{i=1}^{n} (\alpha_i - \alpha_i^2) \left(u_i \cdot u_i\right) - 2 \sum_{i>j}^{n} \alpha_i \alpha_j \left(u_i \cdot u_j\right)
\end{aligned}$$

We first split the summation and then use the symmetric property of the regular inner product, which also holds for the RBF kernel. Then the partial derivate with respect to some coordinate α_p is given by

$$\begin{aligned}
\frac{\partial L}{\partial \alpha_p}(\boldsymbol{\alpha}) &= \frac{\partial}{\partial \alpha_p} \left[\sum_{i=1}^{n} (\alpha_i - \alpha_i^2) \left(u_i \cdot u_i\right) - 2 \sum_{i>j}^{n} \alpha_i \alpha_j \left(u_i \cdot u_j\right) \right] \\
&= (1 - 2\alpha_p) \left(u_p \cdot u_p\right) - 2 \frac{\partial}{\partial \alpha_p} \left[\sum_{i=2}^{n} \sum_{j=1}^{i-1} \alpha_i \alpha_j \left(u_i \cdot u_j\right) \right] \\
&= (1 - 2\alpha_i) \left(u_i \cdot u_i\right) \\
&\quad - 2 \frac{\partial}{\partial \alpha_p} \left[\mathbb{1}_{i<p} \sum_{i=2}^{n} \sum_{j=1}^{i-1} \alpha_i \alpha_j \left(u_i \cdot u_j\right) + \mathbb{1}_{p=i} \sum_{i=2}^{n} \sum_{j=1}^{i-1} \alpha_i \alpha_j \left(u_i \cdot u_j\right) \alpha + \mathbb{1}_{i>p} \sum_{i=2}^{n} \sum_{j=1}^{i-1} \alpha_i \alpha_j \left(u_i \cdot u_j\right) \right] \\
&= (1 - 2\alpha_i) \left(u_i \cdot u_i\right) - 2 \left(\mathbb{1}_{i<p} 0 + \frac{\partial}{\partial \alpha_p} \left[\sum_{j=1}^{p-1} \alpha_p \alpha_j \left(u_p \cdot u_j\right) \alpha + \sum_{i=p+1}^{n} \alpha_i \alpha_p \left(u_i \cdot u_p\right) \right] \right) \\
&= (1 - 2\alpha_i) \left(u_i \cdot u_i\right) - 2 \left(0 + \sum_{j=1}^{p-1} \alpha_j \left(u_p \cdot u_j\right) + \sum_{i=p+1}^{n} \alpha_i \left(u_i \cdot u_p\right) \right) \\
&= (1 - 2\alpha_i) \left(u_i \cdot u_i\right) - 2 \sum_{\substack{j=1 \\ j \neq p}}^{n} \alpha_j \left(u_p \cdot u_j\right)
\end{aligned}$$

Hence, Equation 3.35, can be written as

$$\alpha_i^{k+1} = \max \left\{ 0, \min \left\{ C, \alpha_i^k + \lambda \left[(1 - 2\alpha_i^k) \left(u_i \cdot u_i\right) - 2 \sum_{j=1}^{i-1} \alpha_j^{k+1} \left(u_i \cdot u_j\right) - 2 \sum_{j=i+1}^{n} \alpha_j^k \left(u_i \cdot u_j\right) \right] \right\} \right\} \tag{3.36}$$

When each coordinate is updated we have completed an *epoch* and start updating the first coordinate again, until the stopping condition is met. In Equation 3.35, λ is a hyper-parameter usually referred to as *step size*. Tuning this hyper-parameter is necessary to converge properly to a maximum. Alternatively, a more computationally costly update rule called *line search* could be used to find a local maximum [1] faster.

Negative samples

In this last part on the SVDD method we cover the capability of including negative examples. Tax and Duin extend SVDD to incorporate negative examples [58]. On top of our positive examples $U = (u_1, \ldots, u_n)$ also negative examples, we denote by $W = (w_1, \ldots, w_m)$, are available to us. In practice these would be PIN entries from other users. The extended SVDD formulation then becomes

$$\text{minimize } F(c, r) = r^2 + C_1 \sum_{i=1}^{n} \xi_i + C_2 \sum_{j=1}^{m} \psi_i \tag{3.37}$$

$$\begin{aligned} \text{subject to } & \|u_i - c\|^2 \leq r^2 + \xi_i \quad \forall i \in [n] \\ & \|w_j - c\|^2 \geq r^2 - \psi_j \quad \forall j \in [m] \\ & \xi_i \geq 0 \quad \forall i \in [n] \\ & \psi_j \geq 0 \quad \forall i \in [m] \end{aligned} \tag{3.38}$$

which is quite similar to Problem 3.31.

This extension allows us to reduce the user profile volume further when it is overlapping with other user profiles. However, secure data sharing (server sided) would be required to test for overlap. So assuming the profile is constructed locally on-device, we cannot use this SVDD extension. Moreover, including positive examples in this setting could have undesirable effects on the decision region. The profile could explode in volume when forced in a certain direction by negative examples. To avoid this effect, multi-stage classification can be used. These reasons motivate why this SVDD extension falls outside the scope of this research.

Hyperparameter tuning and learning rates

Since we are dealing with user behaviour data sets of limited size, we want to know how the SVDD learning rates develop with increasing data set size. The train:test split ratio is chosen to be 80:20 in all cases. To reduce sample variance, 50 times a subset of user data set U is drawn and randomly split into training and test sets. Using our data set C once more, we can compute true acceptance rates (TAR) for the SVDD method. We desire training TAR to fall between 0.9 and 1. Moreover, the test TAR should not stray far from the training TAR. Clearly the choice of C influences both train and test rates. In Figure 3.8a training and test rates are shown for fixed value, i.e. $C = 0.05$, of our hyperparameter. Assuming a training TAR of 0.95 is desirable, we see that such rate is never achieved for $C = 0.05$. In contrast Figure 3.8b illustrates training and test rates for C dependent on the sample data size N. Taking $C = \frac{4.5}{N}$ based on various performance tests, we find that training TARs exceed 0.95 and also test TARs have decent performance. For the whole data set used here, $N = 132$, we have C similar to 0.05 from Figure 3.8a, as

$$C = \frac{4.5}{N} = \frac{4.5}{132 \cdot 0.8} \approx 0.0426 \tag{3.39}$$

given the 80:20 split.

From hyperparameter tuning for C, using linear search illustrated in Figure 3.8, we obtained $C \approx 0.16$ to be ideal for this data set containing 132 data points. Comparing $C = 0.05, C = 0.16, C = 0.25$ we observe that $C = 0.25$ overfits on the data, inflating classification region too much, while $C = 0.05$ is too strict and results in a poor TAR. Moreover, for hyperparameter tuning, cross-validation is used to increase the reliability of the test scores [34].

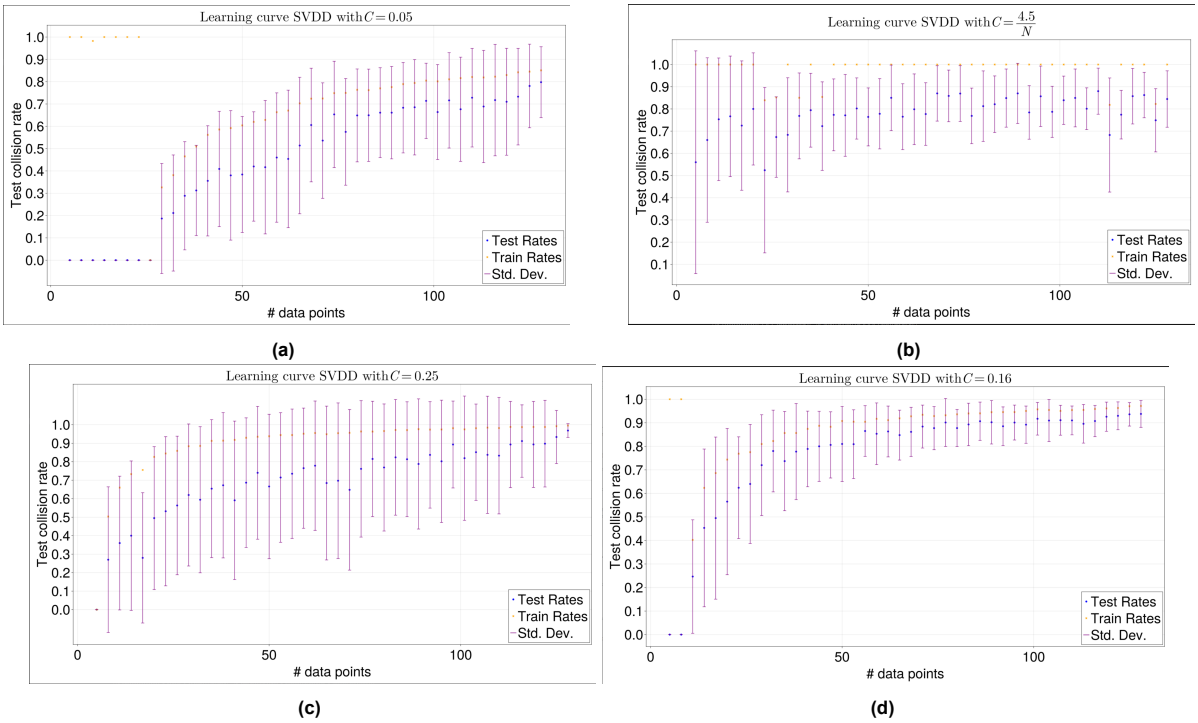


Figure 3.8: SVDD training and test rates comparison between a fixed value of hyperparameter C and a data set size dependent variant $C(n)$.

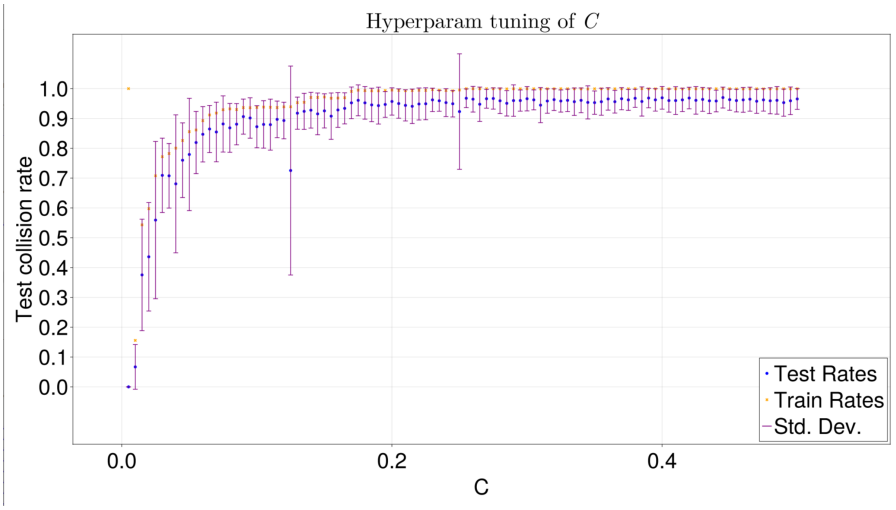


Figure 3.9: Hyperparameter tuning of C; computing training and test TAR on data set I for various values of C. The desired threshold for `Test collision rate` is 0.95.

3.4. Multivariate Gaussian Estimation

A third method to produce a tuneable boundary classifier is fitting a probability distribution to the data set. Now clearly there are many distributions to draw from. Assuming however the behaviour is stable with some natural deviation to it, we argue a Multivariate Gaussian distribution to be suitable. Multivariate Gaussian Estimation (MGE) can be categorized as a density estimation one-class classification method [8]. Following Definition 3.1.2, our MGE classifier on user data U is given by

$$f_U(\mathbf{x}, t) = \begin{cases} 1 & \text{if} \quad p(\mathbf{x}|U) \geq t \\ 0 & \text{otherwise} \end{cases} \tag{3.40}$$

where $p(x|U) \sim \mathcal{N}(\mu_U, \text{Cov}(U))$. To determine t we need to perform hypertuning using cross-validation. In order to estimate the unknown distribution $\hat{P}(U)$ we classify only feature vectors which exceed the threshold t for the given density function. To efficiently compute $p(x|U) \sim \mathcal{N}(\mu_U, \text{Cov}(U))$, we first calculate the sample mean μ_U of the feature vectors. Then we centre the data around the origin by subtracting the sample mean. Based on the centred data we compute the sample covariance matrix. We now write this out more formally. Let us define data matrix $X \in \mathbb{R}^{n \times d}$ where the rows X represent the data points (i.e. feature vectors) of a profile $U = u_1, \ldots, u_n$ with $u_i \in \mathbb{R}^d$, i.e.

$$X = \begin{bmatrix} u_1^T \\ \vdots \\ u_n^T \end{bmatrix} = \begin{bmatrix} u_1 \ldots u_n \end{bmatrix}^T \tag{3.41}$$

Then, the data sample mean we denote by μ_X. We refer to $\overline{X} = X - \mu_X$ as the centred version of X. The Gaussian Covariance Matrix can thus be written as

$$\Sigma = (\mathbf{X} - \mu_X)^T (\mathbf{X} - \mu_X) = \overline{X}^T \overline{X} \tag{3.42}$$

The density function for the estimated Multivariate Gaussian distribution is given by

$$p_{\mathbf{X}}(w) = \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} e^{-\frac{1}{2}(w-\mu_X)^T \Sigma^{-1} (w-\mu_X)} \tag{3.43}$$

Computing the inverse and determinant of the covariance matrix directly is computationally expensive. Using a Singular Value Decomposition (SVD) of the covariance matrix we can more efficiently compute it in stead. Given a SVD of $\overline{X} = UDV^T$ we can write the Gaussian Covariance Matrix as

$$\Sigma = \left(UDV^T\right)^T UDV^T = VDU^T UDV^T = VD^2V^T \tag{3.44}$$

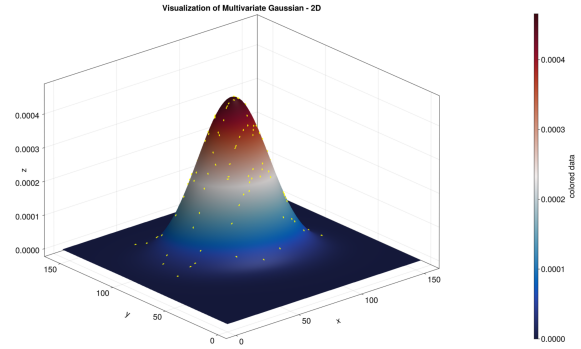
since U is orthogonal. With the SVD formulation of the covariance matrix, its inverse can be written efficiently knowing that V is an orthogonal matrix and D a diagonal matrix

$$\Sigma^{-1} = \left(VD^2V^T\right)^{-1} = \left(V^T\right)^{-1} D^{-2} V^{-1} = VD^{-2}V^T$$

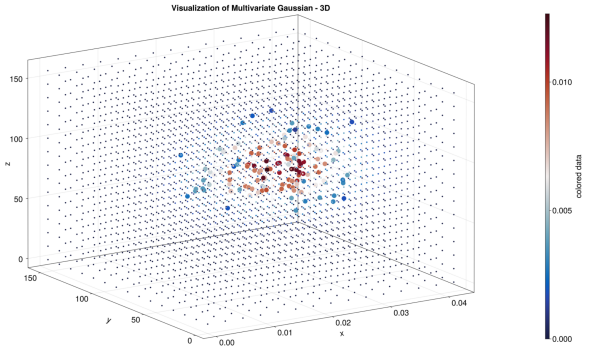
Likewise the determinant of the covariance matrix can be reduced to

$$|\Sigma| = |VD^2V^T| = |VV^T D^2| = |D^2| = |D|^2$$

Again, visualizing all features simultaneously poses a problem. If we restrict to only two features of the data set we obtain a classical visualisation, see Figure 3.10a. For three features we can circumvent rendering in 4D space. Instead, we show MGE densities with colour coding in Figure 3.10b.



(a) 2D visualisation of the MGE probability density function, with x = `relative touch y-coordinate`, y = `keypress hold time`. The z-axis value corresponds to the probability.



(b) 3D visualisation of the MGE probability density function, with x = `touch pressure`, x = `relative touch y-coordinate`, y = `keypress hold time`. The colour of the data points corresponds to the probability. Additional grid points are added as reference.

Figure 3.10: Visualisations of the Multivariate Gaussian Estimation methods applied to Data Set C.

4

Results

4.1. User profiles uniqueness

In the previous section we employed three methods to capture a user's profile; k-MEB, SVDD and MGE. All three methods can be tuned by a (hyper)parameter to eliminate outliers. Let us now assume hyperparameters have been tuned properly to capture the profile as best as possible. Then, one way to quantify a method's additional security strength for PIN authentication is user profile uniqueness. Given two users A and B using the same 5-digit PIN code, how much overlap is there between their keystroke profiles in the feature space. Or stated more strongly, given a profile of user A, do PIN entries of any other user lie inside it's profile interior. In this section we quantify and compare the uniqueness for all three methods.

Important to note is that, no matter the amount of user data collected, we can never guarantee a user profile to be unique, as the set of users using an application changes over time. This begs the question: **To what extend does behavioural biometrics for PIN code authentication allow for unique user profiles.** If not sufficient, other authentication methods need to be considered to capture user's behaviour more accurately. Given the limited variation in features for PIN authentication, combined with the small number of features, we would expect at least some overlap between users. We propose two methods for quantifying user profile uniqueness: Collision Matrices and Overlap Measures.

Finally, we stress the profiles are build using one-class classification (OCC) methods. Hence, dividing up the features space by clustering labelled data of multiple users is undesirable. User profiles will be easier to exploit since too large chunks of space will be allocated to a user. Another reason is privacy. When building a profile locally (on device) data of other users is not available. Therefore we build profiles on a user's data sample solely and only afterwards compute uniqueness. In other words, we do not allow 'negative samples' to help construct a user profile.

4.1.1. Collision Matrix

One way to quantify profile uniqueness is by estimating it using TAR and FAR rates [55]. We now formalise the terms TAR and FAR. Let us recall U to be a data sample of some unknown stable behaviour distribution P. Then using one-class classification methods we construct an user profile approximation $\hat{P}(U)$. We like $\hat{P}(U)$ to be as small as possible to adhere to low FAR, but large enough to be meaningful for authentication in terms of true acceptance rates (TAR). In this research we compute the TAR as follows

$$\text{TAR}(U_{\text{train}}, U_{\text{test}}, \gamma) = \frac{\sum_{\mathbf{x} \in U_{\text{test}}} f_{U_{\text{train}}}(\mathbf{x}, \gamma)}{|U_{\text{test}}|} \tag{4.1}$$

for each partition (i.e. split) of U into U_{train} and U_{test}. Then, for sufficient k-fold cross-validation and split value $\alpha \in [0, 1]$ a valid estimation for TAR of a user profile U is the average of all TAR scores for each respective split, for some one-class classifier method $\mathcal{M}$.

For a given split of U_i we compute the FAR of user U_j on the estimated profile of user U_i by

$$\text{FAR}((U_i)_{\text{train}}, U_j) = \frac{\sum_{\mathbf{x} \in U_j} f_{(U_i)_{\text{train}}}(\mathbf{x}, \gamma)}{|U_j|} \tag{4.2}$$

We prefer True/False acceptance rates (TAR/FAR) over Equal Error Rates (EER), since EER varies widely between user profiles. In a full pairwise comparison less accurate user profiles were "tanking" EER rates, skewing results. To visualize profile uniqueness we create a TAR/FAR matrix, a multi-class error/confusion matrix, we shall refer to as *collision matrix* [45].

The term 'collision' is motivated by a data point of one user lying inside the estimated profile of another. Each cell of the collision matrix contains a (rounded) percentage based on how many data point of User A, column indicated, collide with the estimated profile of User B, row indicated. This implies we have True Acceptance Rates (TAR) on the diagonal. To estimate TAR values properly, we split a User's behaviour data into training and test sets [9], with ratio 80:20 (a common standard). The training portion is used to build (i.e. estimate) a user profile, while the testing portion is checked for collisions on the user profile. In other words, the diagonal of the collision matrix, the TAR values, are estimated by using the user's test set on the profile build using the training set. On the off-diagonal of the collision matrix, one finds FAR of User A, column indicated, with respect to the estimated profile of User B, row indicated. For example, one of many collision matrices, is represented by Table 4.2.

4.1.2. Overlap Measures

Another approach to quantify profile uniqueness is computing user profile overlap. Where the collision matrix can be viewed as approximating the overlap of two ellipsoids by sampling, this approach us with a exact notion. First let us define a hyperellipsoid in its quadratic form

Definition 4.1.1: Hyperellipsoid [11, 62]

Let $A \in S^d_{++}$ and $a \in \mathbb{R}^d$. Then the[a] hyperellipsoid (including its interior), denoted $\mathcal{E}(a, A)$ in its quadratic form is given by

$$\mathcal{E}(a, A) = \{x \in \mathbb{R}^d : (x - a)^T A(x - a) \leq 1\} \tag{4.3}$$

[a] justified by uniqueness

This allows us to define a shorthand notation for our user profile which assumed to be shaped like a hyperellipsoid.

Definition 4.1.2: User profile hyperellipsoid

Let U be a user data sample from an unknown distribution P. Moreover, let $\hat{P}(U)$ be a matrix describing an ellipsoidal decision region obtained by appropriate boundary one-class classification method $\mathcal{M}$. We denote the ellipsoid offset with respect the origin with $\mu_{\hat{P}(U)}$ and denote the centred version of $\hat{P}(U)$ by $\overline{\hat{P}(U)}$. Then, we define a user profile hyperellipsoid by

$$\mathcal{E}_U := \mathcal{E}\left(\mu_{\hat{P}(U)}, \overline{\hat{P}(U)}\right) \tag{4.4}$$

Given two ellipsoids $\mathcal{E}_A$ and $\mathcal{E}_B$ the (mutual) overlap is simply their intersection $\mathcal{E}_A \cap \mathcal{E}_B$. Hughes and Chraibi provide us with a general algorithm for regular (2D) ellipses [28]. Unfortunately, in general it is not known how to analytically find overlap between hyperellipsoids [46]. Gilitschenski and Hanebeck do provide us, in their 2012 paper, with a fast way to check whether two ellipsoids overlap or not [24]. They define a convex scalar function $K : (0, 1) \to \mathbb{R}$ for two (hyper)ellipsoids to do so, given by

$$K(s) := 1 - (b - a)^T \left(\frac{1}{1 - s}A^{-1} + \frac{1}{s}B^{-1}\right)^{-1} (b - a) \tag{4.5}$$

for $a, b \in \mathbb{R}^d$ and $A, B \in S^d_{++}$[1].

[1] S_{++} denotes the collection of all symmetric positive definite matrixes in $\mathbb{R}^{d \times d}$

Theorem 4.1.1: Ellipsoid intersection theorem [26]

Let $\mathcal{E}_A$ and $\mathcal{E}_B$ hyperellipsoids. Given the convex scalar function $K(s)$ it holds that

$$\mathcal{E}_A \cap \mathcal{E}_B = \emptyset \iff \exists s \in (0,1) : K(s) < 0 \tag{4.6}$$

In our context consider some user profile hyperellipsoids $\mathcal{E}_A$ and $\mathcal{E}_B$. Given (estimated) Ellipsoidal user profiles regions, we can calculate the pairwise overlap of these Ellipsoids, as well as the total overall overlap. Note that these overlap measures do not have to the same. Figure 4.1 illustrates that for three user profiles A, B and C the total overlap for A is equal to segment 1 plus segment 3 minus segment 2. Ideally, we would like the total overlap to be as low as possible to get close to profile 'uniqueness'. Since total overlap is computationally hard to compute we can instead use 'summed pairwise overlap'. In fact, 'summed pairwise overlap' is an upper bound for total overlap.

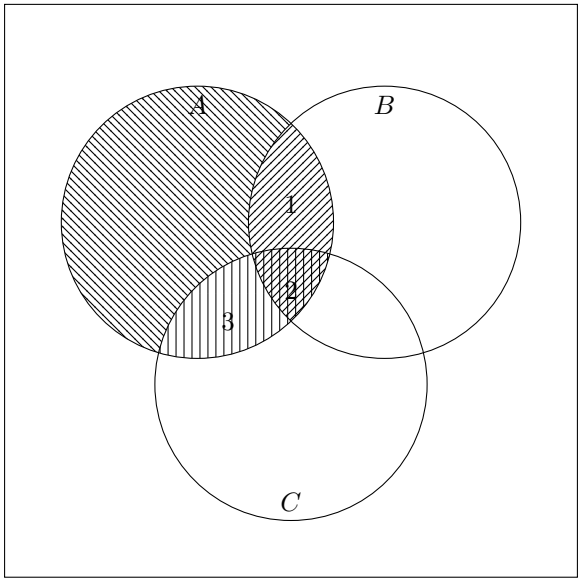


Figure 4.1: Venn diagram illustrating the distinction between pairwise and total overlap for three fictive User profiles A, B and C.

If want to quantify the performance methods and compare between them, there are multiple objectives to minimize. Let $\mathcal{U} = \{U_1, U_2, \ldots, U_t\}$ be a collection of users. One such objective function $g(\mathcal{U})$ would be the overall summed pairwise overlap, see Equation (4.7), which simply adds all summed pairwise overlap, and divides by two to account for double counting.

$$g(\mathcal{U}) = \frac{1}{2} \sum_{(U,W) \in \binom{\mathcal{U}}{2}} \mathbf{Vol}(\mathcal{E}_U \cap \mathcal{E}_W) \tag{4.7}$$

Another measure would be the average summed pairwise overlap. For a large amount of users, it raises the question if sacrificing the overlap measure of a single profile is acceptable to reduce the global objective. Hence there might remain a few users that would have had a relatively large summed pairwise overlap, but their profile is contracted tremendously to avoid it. Similarly to EER, these objectives leaves us with little control on how to balance the TAR/FAR trade-off.

4.2. Default PIN pad results

In the following Subsections (4.2.1, 4.2.2, 4.2.3) the results are discussed of three methods to capture a user profile based on its behavioural biometrics traits. All results presented in this Section are derived from data set 'Phase 1'. For each of these profile building methods we computed collision matrices and list them as tables. The methods are trained on data obtained using a default PIN pad configuration. Out of all 18 participants of 'Phase 1', nine users completed over 50 valid transactions, of which five

completed over 100 valid transactions. Table 4.1 show the number of valid transaction per participant as referred to in these results of 'Phase 1' and the corresponding collision matrices. In addition we

	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9
$\lvert U\rvert$	250	62	238	89	71	63	559	197	104
$\lvert U_{\text{train}}\rvert$	200	49	190	71	56	50	447	157	83

Table 4.1: User data sample sizes (Phase 1) and respective training set sizes for a 80:20 split ratio.

also had a bot complete 488 valid transactions. We did not randomize the bot's clicks so its relative keypress locations are identical for each transaction. All participants used the same PIN sequence, i.e. $3, 0, 1, 9, 4$, without prescribed rules on how to enter this PIN sequence. In contrast, in Section 4.3 the same methods are trained on behavioural data for customized PIN pads.

4.2.1. k-MEB method

The number of points to exclude, denoted by $k \in \mathbb{N}$, can both be a fixed number or relative to the size of the user data sample U. Since we split U into U_{train} and U_{test} to obtain more reliable TAR/FAR estimates it makes sense to let k be a fraction of the size of U_{train}. We estimate a user profile on U_{train}. Hence, for computing collision matrices using k-MEB we use

$$k = \left\lfloor (1-\kappa) \cdot |U_{\text{train}}| + \frac{1}{2} \right\rfloor \qquad \kappa \in [0,1] \tag{4.8}$$

which ensures $0 \leq k \leq |U_{\text{train}}|$. Since our implementation of k-MEB is more computationally expensive than SVDD or MGE we did 25 folds instead of 100 when computing TAR/FAR scores. Our first result is a collision matrix found in Table 4.2 using the k-MEB method with (kappa) $\kappa = 0.95$. Taking $\kappa = 0.95$ implies that five percent of the data point in U_{train} will be considered outliers and the estimated user profile will not include these.

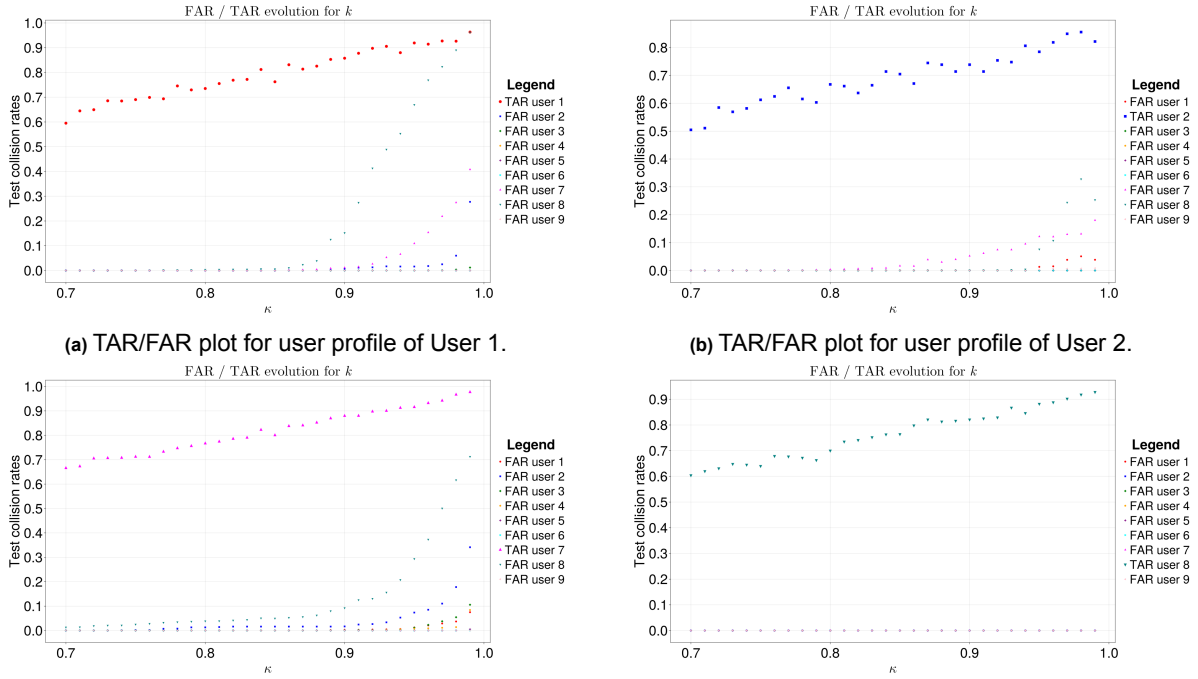
In the Table we see User 1 obtained an estimated TAR score of 0.908. Given the estimated user profile of User 1, collisions were found with User 2, 7 and 8 according to estimated FAR scores of 0.016, 0.109 and 0.689 respectively. Furthermore, we not that the collision matrix is asymmetric. The reason for the asymmetry lies in the fact a profile is constructed as concise as possible, not using all points from the user data sample. If a user profile is more spread out, i.e. harder to compress, it is more likeli to have collision. User 7 fits this description in Table 4.2, where its estimated user profile collides with many other user data samples. Only User 5, 6 and 9 do not have collision with User 7's profile and hence display FAR scores of 0.0. For $\kappa = 0.95$ we expected TAR scores to be also close to 0.95. While smaller data sets like User 5 and 6 have TAR scores of 0.768 and 0.788, User 2 (the smallest of all) has a TAR 0.865. Meaning this TAR discrepancy cannot be solely explained by the data set size. Furthermore we notice User 8's data colliding with profiles of User 1, 2 and 7 resulting in FAR scores of $0.689, 0.117$ and 0.299 respectively. Since User 1,2 and 7 do not collide on the profile of User 8, this could imply profiles of User 1,2 and 7 are less compact.

	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9
User 1	0.908	0.016	0.0	0.0	0.0	0.0	0.109	0.689	0.0
User 2	0.017	0.865	0.0	0.0	0.0	0.0	0.126	0.117	0.007
User 3	0.0	0.0	0.914	0.0	0.011	0.0	0.0	0.0	0.0
User 4	0.0	0.0	0.0	0.840	0.0	0.134	0.0	0.0	0.0
User 5	0.0	0.0	0.002	0.0	0.768	0.0	0.0	0.0	0.0
User 6	0.0	0.0	0.0	0.023	0.0	0.788	0.0	0.0	0.0
User 7	0.013	0.076	0.011	0.008	0.0	0.0	0.916	0.299	0.0
User 8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.870	0.0
User 9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.817

Table 4.2: User Profiles collision matrix using k-MEB with $k = 0.05 \cdot |U_{\text{train}}|$ for all users.

To gain insight in the relation of κ, and k by extension, to TAR and FAR scores we compute these for various values of κ. Plotting these allows to better understand how to tune the user profile and balance FAR/TAR scores. These plots are provided in Figure 4.2 for User 1, 2, 7 and 8. For User 1 we see in Figure 4.2a that FAR score by User 8 strongly increases in the range $\kappa \in [0.85, 1]$. Moreover, the

TAR score appears to be linearly related to κ. Depending on the security risk of an application $\kappa = 0.85$ might be considered, especially for a high-risk transaction. For User profiles of User 2, 7 and 8 setting $\kappa = 0.95$ might be appropriate for low-risk transactions to improve user experience.



(a) TAR/FAR plot for user profile of User 1.

(b) TAR/FAR plot for user profile of User 2.

(c) TAR/FAR plot for user profile of User 7.

(d) TAR/FAR plot for user profile of User 8.

Figure 4.2: Plots relating TAR/FAR to various values of κ for User 1,2,7 and 8, based on 25-folds and a split ratio of 0.8

4.2.2. SVDD method

In the previous subsection we listed results obtained by the one-class classifier method k-MEB. In this subsection we shall state results for the SVDD method. Unless specified otherwise, we used 100 folds and a train-test split ratio of 0.8 to estimate FAR/TAR scores. Initially we used a fixed value for C for all user profiles, namely $C = 0.05$ based on our findings in Subsection 3.3. The collision matrix for $C = 0.05$ is shown in Table 4.3. On the diagonal we immediately see how fixing $C = 0.05$ is way to punishing in terms of TAR for User 2,4,5 and 6. In particular, for User 2, which completed 62 valid transactions, the chosen value of $C = 0.05$ is too strict given the limited number of completed transactions. For User 5 taking $C = 0.05$ results in a TAR score of 0.56. On the other hand, we only find three FAR scores above 0.1 overall. Increasing C would work for small data sets, but not for the larger ones such as user 1 and 7, who are responsible for the remaining FAR scores over 0.1. To overcome this, we computed

	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9
User 1	0.88	0.02	0.0	0.0	0.0	0.0	0.06	0.62	0.0
User 2	0.0	0.57	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 3	0.0	0.0	0.86	0.0	0.0	0.0	0.0	0.0	0.0
User 4	0.0	0.0	0.0	0.62	0.0	0.04	0.0	0.0	0.0
User 5	0.0	0.0	0.0	0.0	0.56	0.0	0.0	0.0	0.0
User 6	0.0	0.0	0.0	0.09	0.0	0.56	0.0	0.0	0.0
User 7	0.03	0.14	0.0	0.01	0.0	0.0	0.94	0.33	0.0
User 8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.85	0.0
User 9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.71

Table 4.3: User profile collision matrix for SVDD, with $C = 0.05$ fixed for all profiles. A 100-fold cross-validation is used with 80:20 splits.

a collision matrix using C values dependent on the size of the user data sample U. In particular, we relate C to a scaled inverse of the training set size. Thus, one possible relation is given by

$$C = \frac{16}{|U_{\text{train}}|} \tag{4.9}$$

where we derived the scaling factor from the baseline data set analysis, such as Figure 3.8b. The corresponding collision matrix based on Equation (4.9) is displayed in Table 4.4. When compared to

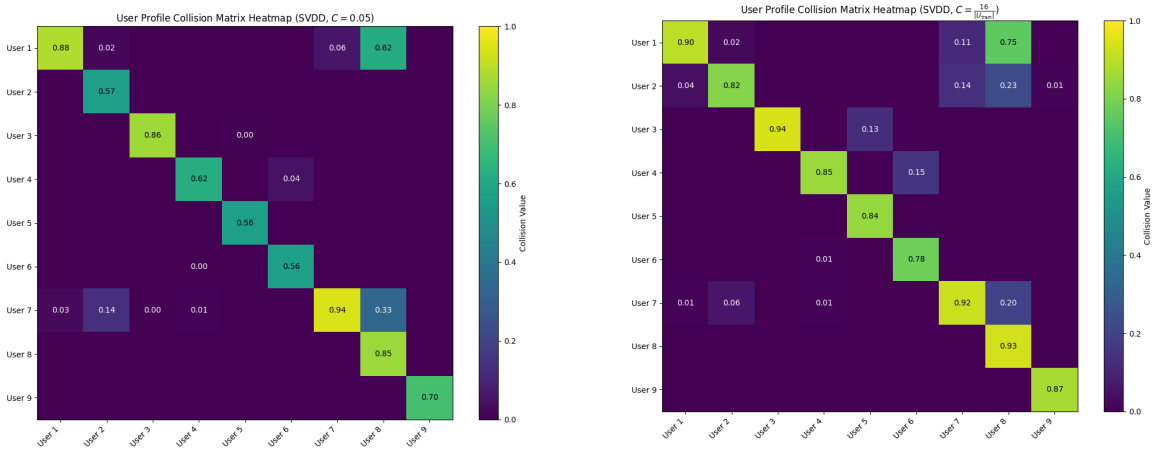
	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9
User 1 ($C = 0.08$)	0.9	0.02	0.0	0.0	0.0	0.0	0.11	0.75	0.0
User 2 ($C = 0.327$)	0.04	0.82	0.0	0.0	0.0	0.0	0.14	0.23	0.01
User 3 ($C = 0.084$)	0.0	0.0	0.94	0.0	0.13	0.0	0.0	0.0	0.0
User 4 ($C = 0.225$)	0.0	0.0	0.0	0.85	0.0	0.15	0.0	0.0	0.0
User 5 ($C = 0.286$)	0.0	0.0	0.0	0.0	0.84	0.0	0.0	0.0	0.0
User 6 ($C = 0.32$)	0.0	0.0	0.0	0.01	0.0	0.78	0.0	0.0	0.0
User 7 ($C = 0.036$)	0.01	0.06	0.0	0.01	0.0	0.0	0.92	0.2	0.0
User 8 ($C = 0.102$)	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.93	0.0
User 9 ($C = 0.193$)	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.87

Table 4.4: User profile collision matrix for SVDD, with $C = \frac{16}{|U_{\text{train}}|}$. A 100-fold cross-validation is used on 80:20 splits.

Table 4.3 we see a clear increase in most TAR scores, with a notable exception of user 1 and user 8. Moreover, we see user 3's profile is now described too loosely, indicated by a FAR of 0.437 for user 5, which was 0 in the collision table for $C = 0.05$. To visually compare collision matrices more easily we can create corresponding heatmaps, see Figure 4.3. The TAR/FAR trade-off becomes apparent for user 2 in particular. While TAR for user 2 increases from 0.57 in Figure 4.3a to 0.82 in Figure 4.3b, the FARs also increase, most notably for user 7 and 8 with respect to user 2's profile.

In order to tune hyperparameter C value for a user we can also view the evolution of TAR and FAR for various C. Plots relating TAR/FAR scores to C for users 1,2,7 and 8 are shown in Figure 4.4. User 2 does not reach TAR scores greater than 0.9 for any value of C listed, while user 1,7 and 8 reach this threshold for $C = 0.07$, $C = 0.03$ and 0.1 respectively. Figure 4.4a shows that user 1's profile collides with user 8, just was we observed in the heat maps 4.3. For user 1, with $C = 0.02$ a TAR of 0.74 can be achieved with a low FAR of 0.04 (user 8). However, with $C = 0.05$ a TAR of 0.88 is most likely at the cost of a FAR of 0.62 (user 8). Furthermore, we observe that user 8's profile does not collide with any of the other eight users, see Figure 4.4d. This suggests user 8's profile is distinctive.

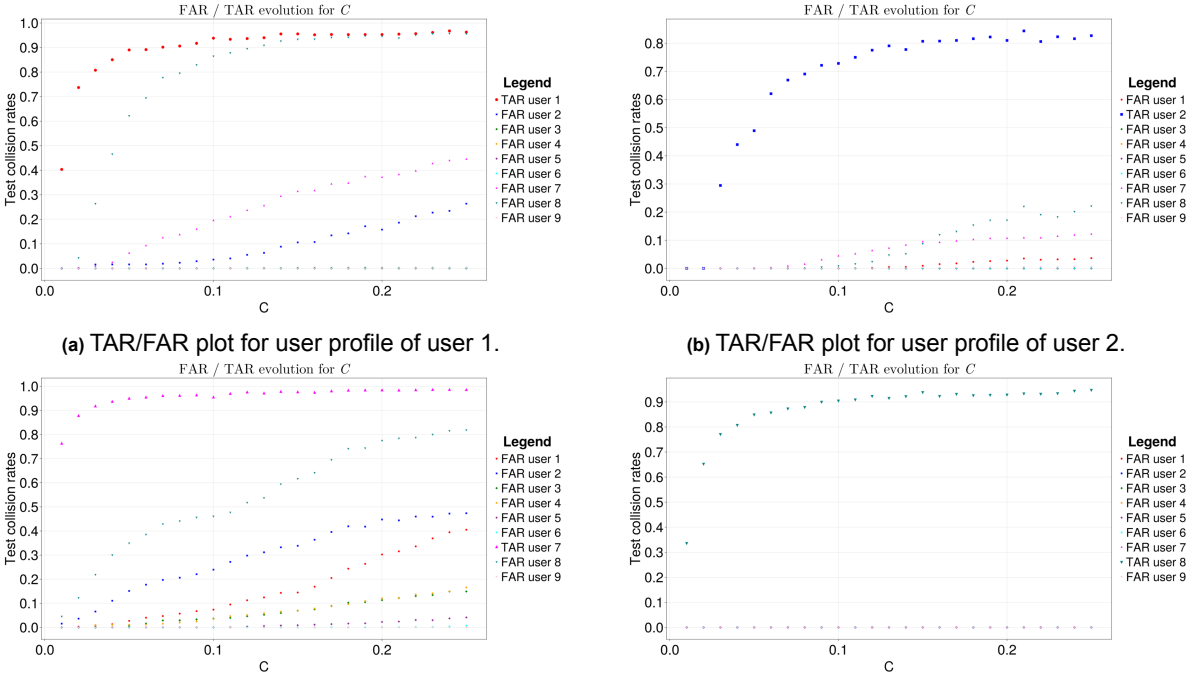
While insightful for this research, for on-device implementations FAR scores are not available to tune C. An example of a tuned version is visualised as a heatmap in Figure 4.5. In this tuned version we used C values $[0.02, 0.15, 0.08, 0.16, 0.25, 0.25, 0.02, 0.1, 0.25]$ for User 1 till 9 respectively. We have



(a) Heatmap of SVDD collision matrix with $C = 0.05$ for all user profiles.

(b) Heatmap of SVDD collision matrix with $C = \frac{16}{|U_{\text{train}}|}$ for all user profiles.

Figure 4.3: Heatmaps of SVDD collision matrices. Left $C = 0.05$ and right $C = \frac{16}{|U_{\text{train}}|}$.



(a) TAR/FAR plot for user profile of user 1.

(b) TAR/FAR plot for user profile of user 2.

(c) TAR/FAR plot for user profile of user 7.

(d) TAR/FAR plot for user profile of user 8.

Figure 4.4: Plots relating TAR/FAR to various values of C for User 1,2,7 and 8.

chosen to prioritise reducing FAR scores, but also limited C to at most 0.25. User 3 reaches the highest TAR score of 0.91. The lowest TAR score is belongs to User 1 with 0.73. The highest FAR for User 1 however is reduced from 0.62, in Figure 4.3a, to only 0.04. After boundary OCC methods such as k-MEB and SVDD, we provide results of a density method in the next subsection.

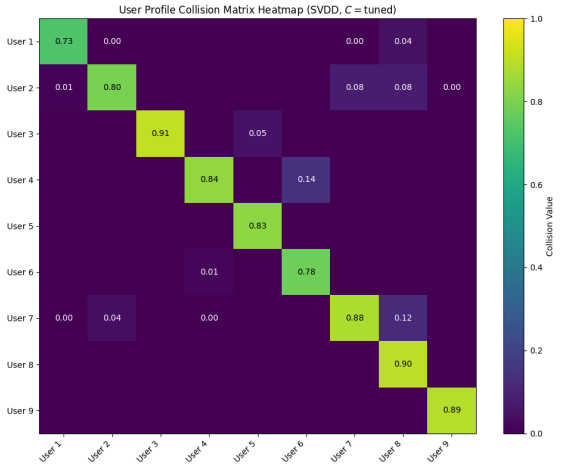


Figure 4.5: Heatmap of SVDD collision matrix with tuned values for C. Respective values for C for User 1 to 9 are $[0.02, 0.15, 0.08, 0.16, 0.25, 0.25, 0.02, 0.1, 0.25]$

4.2.3. Multivariate Gaussian Estimation method

In this last subsection, for authentication using a default PIN pad layout, we test uniqueness for user profiles obtained through Multivariate-Gaussian Estimation (MGE). Multivariate-Gaussian Estimation is a density one-class classification method. This allows to consider probabilities and match that for a threshold given the risk of a certain transaction. Alternatively, by setting a predetermined threshold a classification boundary is obtained similar to k-MEB and SVDD. In other words, given both the user profile distribution and the boundary threshold probability, we can iterate over all possible data points from other users to see *collisions* occur. A collision is a data point which probability exceeds the boundary threshold value. This implies the PIN entry corresponding to the data point would have been accepted according to the MGE model. The threshold is set based on the α-percentile of the training data probabilities. In other words α percent of the training data set is considered an anomaly. A FAR/TAR trade-off is a result of tuning this α. A lower value of α should result in higher TAR scores and higher TAR scores.

In Table 4.5 a collision matrix is displayed for user profiles using the MGE method with $\alpha = 0.05$ (i.e. 5-percentile). The collision matrix shows better TAR scores for users with a larger user data sample, notably User 1, 3 and 7. In order to reduce the FAR score of User 8 on User 1's estimated user profile,

	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9
User 1	0.91	0.02	0.0	0.0	0.0	0.0	0.02	0.53	0.0
User 2	0.0	0.55	0.0	0.0	0.0	0.0	0.01	0.0	0.0
User 3	0.0	0.0	0.92	0.0	0.12	0.0	0.0	0.0	0.0
User 4	0.0	0.0	0.0	0.58	0.0	0.0	0.0	0.0	0.0
User 5	0.0	0.0	0.0	0.0	0.48	0.0	0.0	0.0	0.0
User 6	0.0	0.0	0.0	0.01	0.0	0.39	0.0	0.0	0.0
User 7	0.0	0.06	0.02	0.0	0.0	0.0	0.93	0.04	0.0
User 8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.84	0.0
User 9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.61

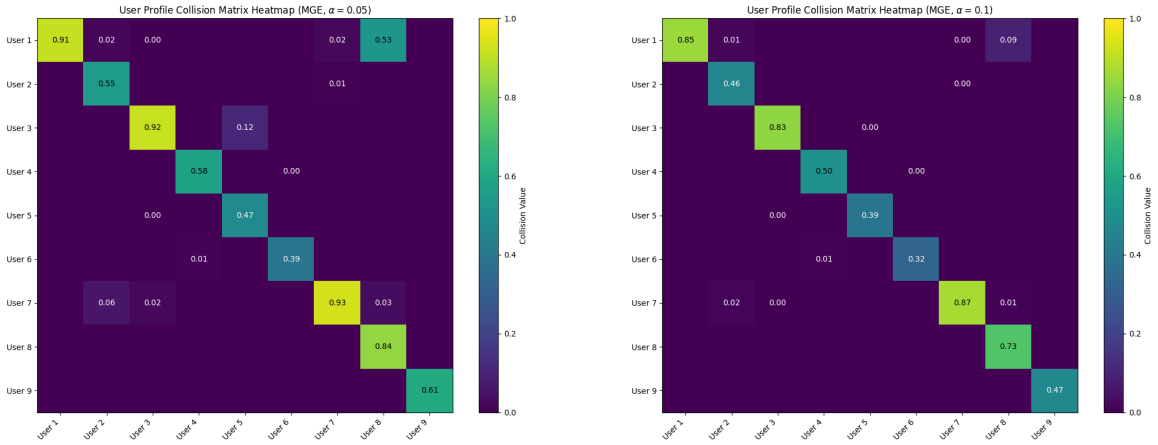
Table 4.5: User Profiles collision matrix for 5-percentile MGE thresholds based on 100 folds, with train/test split ratio 0.8. Each cell contains a (rounded) percentage of total possible collision is given where the row indicates the MGE of a user and the column user data points used to compute a collision.

we also provide a 10-percentile collision matrix ($\alpha = 0.1$) in Table 4.6. As expected, TAR scores have decreased. For example, User 9's TAR score drops from 0.61 at $\alpha = 0.05$ to 0.47 for $\alpha = 0.1$. On the other hand, all FAR scores have dropped below 0.1, as intended. To compare collision matrices side-by-side for both $\alpha = 0.05$ and $\alpha = 0.1$, Figure 4.6 provides heat maps representing the collision matrices. In real-world applications TAR scores greater than 0.9 are desirable to not hinder user experience. For $\alpha = 0.1$ none of the user reach such TAR score of 0.9 while for $\alpha = 0.05$ TAR scores of user 1,3 and 7 do exceed the 0.9 threshold. FAR scores indicated with 0.00 signify collision have been observed but rounded to the nearest second decimal place result in 0.00.

Similarly to Figure 4.4 we plot TAR/FAR trade-off for different values of α. Figure 4.7 shows these plots for Users 1,2,7 and 8. For ease of comparing TAR/FAR plots of different methods the percentile

	User 1	User 2	User 3	User 4	User 5	User 6	User 7	User 8	User 9
User 1	0.85	0.01	0.0	0.0	0.0	0.0	0.0	0.09	0.0
User 2	0.0	0.46	0.0	0.0	0.0	0.0	0.0	0.0	0.0
User 3	0.0	0.0	0.83	0.0	0.0	0.0	0.0	0.0	0.0
User 4	0.0	0.0	0.0	0.50	0.0	0.0	0.0	0.0	0.0
User 5	0.0	0.0	0.0	0.0	0.39	0.0	0.0	0.0	0.0
User 6	0.0	0.0	0.0	0.01	0.0	0.322	0.0	0.0	0.0
User 7	0.0	0.02	0.0	0.0	0.0	0.0	0.87	0.01	0.0
User 8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.73	0.0
User 9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.47

Table 4.6: User Profiles collision matrix for 10-percentile MGE thresholds based on 100 folds, with train/test split ratio 0.8. Each cell contains a (rounded) percentage of total possible collision is given where the row indicates the MGE of a user and the column user data points used to compute a collision.

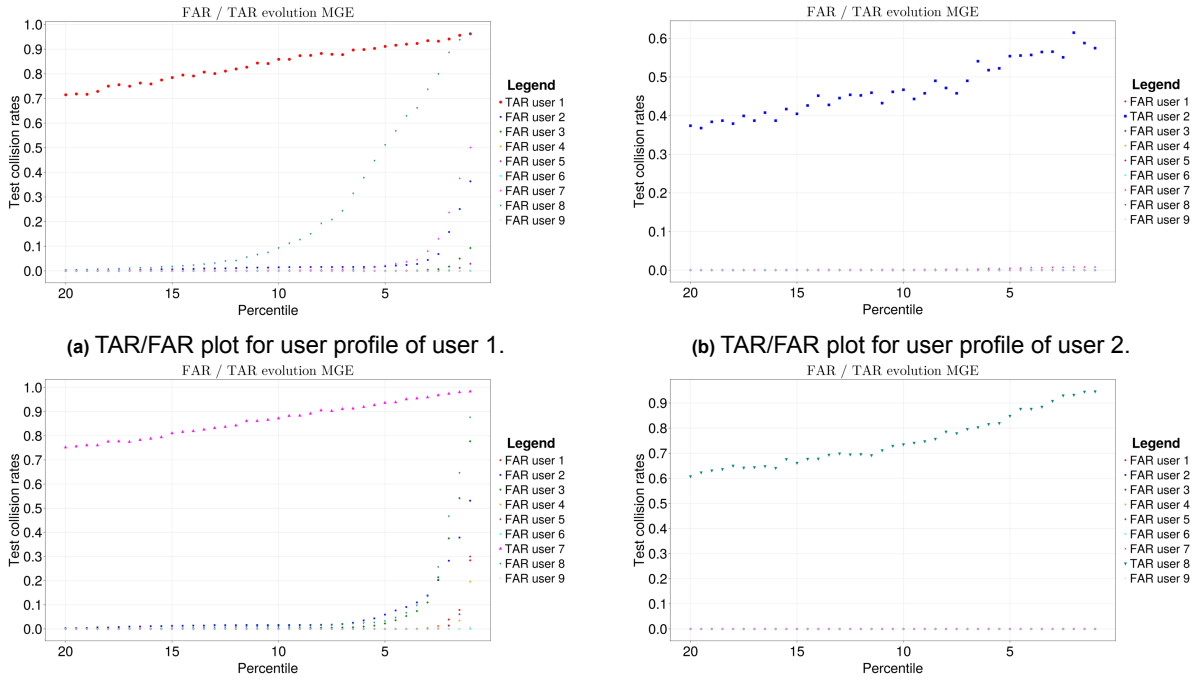


(a) Heatmap of MGE collision matrix with $\alpha = 0.05$ for all user profiles.

(b) Heatmap of MGE collision matrix with $\alpha = 0.1$ for all user profiles.

Figure 4.6: Heatmaps of MGE collision matrices. Left $\alpha = 0.05$ and right $\alpha = 0.1$.

axis direction has been reversed. User profiles by MGE for User 1 and User 7 see FAR scores increase around $\alpha = 0.15$ and $\alpha = 0.07$ respectively. The relation between the percentile α and TAR score appears to be linear, similar to k for the k-MEB method as seen in Figure 4.2. The TAR score for user profile of User 2 (Figure 4.7b) is estimated to be 0.56 for $\alpha = 0.05$ and barely reaches 0.6 for the lowest percentiles.



(a) TAR/FAR plot for user profile of user 1.

(b) TAR/FAR plot for user profile of user 2.

(c) TAR/FAR plot for user profile of user 7.

(d) TAR/FAR plot for user profile of user 8.

Figure 4.7: Plots relating TAR/FAR to various values of α for User 1,2,7 and 8 for MGE.

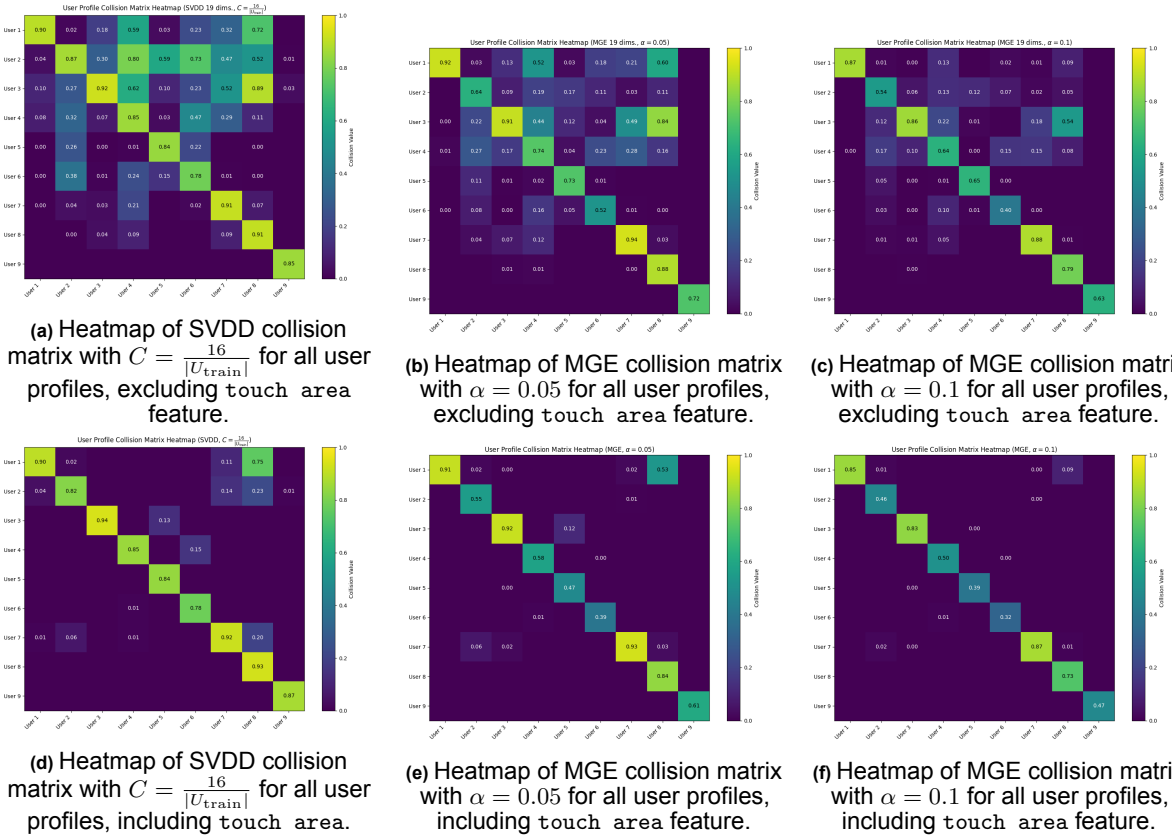
4.2.4. Impact on excluding physical biometric related features

The results in the previous subsections did not include the feature `touch pressure`, because not all participants used mobile devices which support collecting it. The results did include the feature `touch area` for each digit. To contribute any effect to learnt behaviour, we must understand the contribution physical biometric dependent features. Simply put, participants having smaller/bigger fingers might determine the range for values of `touch area` to fall into. While certainly useful in practice, its role hinders us to rightfully attribute good results to us having properly estimated behavioural biometrics. Excluding touch area as a feature our feature space shrinks from 24 to 19 in dimension size.

Using Phase 1 data, we compare heatmaps of collision matrices both with and without touch area features. These heatmaps are displayed in Figure 4.8. When comparing the collision matrix results obtain by SVDD with $C = \frac{16}{|U_{\text{train}}|}$ we immediately see more, and higher, FAR scores for the 19-dimensional feature space, see Subfigure 4.8a and 4.8d respectively.

Furthermore, in the MGE case, especially for $\alpha = 0.05$, the FAR scores actually increase when excluding touch area. For example, user 5 in Figure 4.8b has an estimated TAR score of 0.47, but in Figure 4.8e is believed to score 0.73. The FAR score for User 2 on User 5's profile did increase from 0 to 0.11 though. A similar effect we observe for User 2, 4, 6 and 9. The increase in TAR scores is also noticeable for $\alpha = 0.1$, but its effect is less strong. On the other hand the FAR scores also increased less.

Let us put these scores in perspective. One example of a dummy classifiers suitable for this research is a stratified classifier which would result in an equal TAR-FAR score, for example 50-50 or 95-95. In all cases the methods outperform dummy classifiers. In other words, even without the feature `touch area` we can still estimate a user's profile using OCC methods, albeit at a big performance loss for Users 1,2,3 and 4. Thus, we must conclude touch area does play a significant role in building user profiles which are more distinctive.



(a) Heatmap of SVDD collision matrix with $C = \frac{16}{|U_{\text{train}}|}$ for all user profiles, excluding `touch area` feature.

(b) Heatmap of MGE collision matrix with $\alpha = 0.05$ for all user profiles, excluding `touch area` feature.

(c) Heatmap of MGE collision matrix with $\alpha = 0.1$ for all user profiles, excluding `touch area` feature.

(d) Heatmap of SVDD collision matrix with $C = \frac{16}{|U_{\text{train}}|}$ for all user profiles, including `touch area`.

(e) Heatmap of MGE collision matrix with $\alpha = 0.05$ for all user profiles, including `touch area` feature.

(f) Heatmap of MGE collision matrix with $\alpha = 0.1$ for all user profiles, including `touch area` feature.

Figure 4.8: Heatmap comparison between

4.3. Custom PIN pad configurations

To increase the distinctiveness of the learnt behaviour a customized PIN pad can be used. Desirable are adaptions of the PIN pad which are least disruptive to the user and feels reasonably familiar. Hence we chose permuting the PIN pad digits to achieve customization.

Every user has an *Asset*, which has a unique identifier (i.e. UUID). In Section 2.3 we explained how we mapped a user's unique *Asset* identifier to one of the $3.628.800$ $(= 10!)$ different PIN pad permutations. While with high probability different between users, the permutation is fixed across transactions per user. However, with many users, it is possible for distinct users to still obtain the same layout, resulting in a similar analysis of the previous results (Section 4.2.2). Nevertheless, in the following results no pair of users used the same layout in this second phase.

In similar fashion to Phase 1 we shall analyse the user data and compare method performances. Phase 2 of data collection ran for a total of 10 weeks, from 17-January-2025 until 31-March-2025. Out of twenty people partaking in 'Phase 2', nine completed over 50 valid transactions. A transaction is considered valid if the correct PIN is entered without use of backspace. All results presented are derived from the data set 'Phase 2'. Similar to 'Phase 1', all participants used the same PIN sequence, i.e. $3, 0, 1, 9, 4$, without prescribed rules on how to enter this PIN sequence. During this 10 week period, out of twenty participants, seven unique participants completed over 100 valid transactions. Data sets of these seven participants, which reached the 100 valid transaction threshold, are used to obtain the results in this section. The exact number of data points for each user can be found in Table 4.7.

User	1	2-a	3-a	4	5	2-b	6	3-b	7
$\lvert U \rvert$	434	118	258	128	469	201	111	137	202
$\lvert U_{\text{train}} \rvert$	347	94	206	102	375	160	88	109	161

Table 4.7: User data sample sizes (Phase 2) and respective training set sizes for a 80:20 split ratio.

Due to technical difficulties with their phone two participants, User '2' and '3' both had to switch to

another asset during the data collection phase. For clarity, we keep these profiles separate and refer to them as '2-a', '2-b', '3-a' and '3-b'. We refrained from merging the data sets into one for a user because in this second phase the layout of the PIN pad differed. Whilst unfortunate, it also provides us with insight in how persistent their behaviour is across both assets. The exact number of data points for each user can be found in Table 4.7. Since we use a 80:20 split ratio, corresponding training set sizes are also listed.

4.3.1. k-MEB method

Using the k-MEB one-class classification method we constructed user profiles for phase 2 (i.e. where users had different PIN pad layouts). These results are displayed in Table 4.8 and corresponding heat map in Figure 4.9. The TAR scores exceed are generally better than in phase 1. We count a similar number of significant FAR scores, whilst accounting for the fact that we have less unique users in these results of phase 2.

Moreover, we find user 3-b's data collide, with a FAR score of 0.4, on the profile of User 3-a. This indicates that some behaviour transcends the layout. The collision matrix is not symmetric for FAR scores, even for user 3-a and 3-b. The FAR score obtained by testing data of 3-b on the estimated profile of 3-a is 0.4, while the FAR score obtained from 3-a on 3b is 0.12. We also note that User 4's profile has numerous collision by both User 2-a and User 7, resulting in FAR scores of 0.38 and 0.26 respectively, but not with User 2-b. User 1 has no collision with anyone, except for some negligible exceptions.

User	1	2-a	3-a	4	5	2-b	6	3-b	7
1	0.92	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2-a	0.0	0.86	0.0	0.01	0.0	0.0	0.0	0.0	0.0
3-a	0.0	0.0	0.91	0.0	0.0	0.0	0.14	0.4	0.0
4	0.0	0.38	0.0	0.87	0.0	0.0	0.0	0.0	0.26
5	0.0	0.0	0.0	0.0	0.91	0.0	0.0	0.0	0.0
2-b	0.0	0.18	0.0	0.03	0.18	0.91	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	0.84	0.03	0.0
3-b	0.0	0.0	0.12	0.0	0.0	0.0	0.31	0.84	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.9

Table 4.8: User Profiles collision matrix of phase 2 using k-MEB with $k = 0.05 \cdot |U_{\text{train}}|$ for all users (i.e. $\kappa = 0.95$).

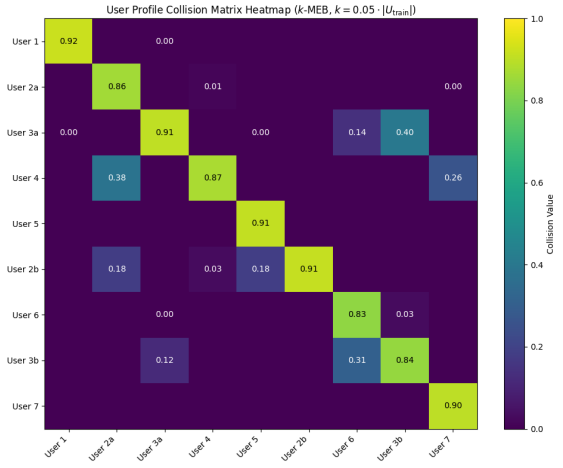


Figure 4.9: Heatmap of k-MEB collision matrix with $k = 0.05 \cdot |U_{\text{train}}|$ for all users (i.e. $\kappa = 0.95$).

4.3.2. SVDD method

Similar to phase 1, we computed the collision matrix for a fixed value, $C = 0.05$, for all user profile estimates. In Table 4.9 these results for $C = 0.05$ are listed. Furthermore, we compute the collision matrix for values of C dependent on the training set size, i.e. $C = \frac{16}{|U_{\text{train}}|}$. These results are given in Table 4.10. In Figure 4.10 we visualise both collision matrix tables using heatmaps. Comparing both

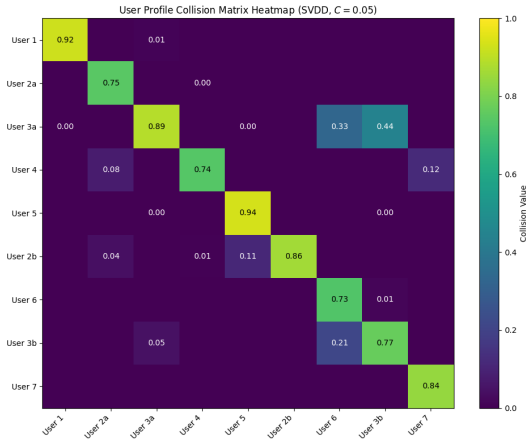
User	1	2-a	3-a	4	5	2-b	6	3-b	7
1	0.92	0.0	0.01	0.0	0.0	0.0	0.0	0.0	0.0
2-a	0.0	0.75	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3-a	0.0	0.0	0.89	0.0	0.0	0.0	0.33	0.44	0.0
4	0.0	0.08	0.0	0.74	0.0	0.0	0.0	0.0	0.12
5	0.0	0.0	0.0	0.0	0.94	0.0	0.0	0.0	0.0
2-b	0.0	0.04	0.0	0.01	0.11	0.86	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	0.73	0.01	0.0
3-b	0.0	0.0	0.05	0.0	0.0	0.0	0.21	0.77	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.84

Table 4.9: User Profiles collision matrix using SVDD with $C = 0.05$ for all users.

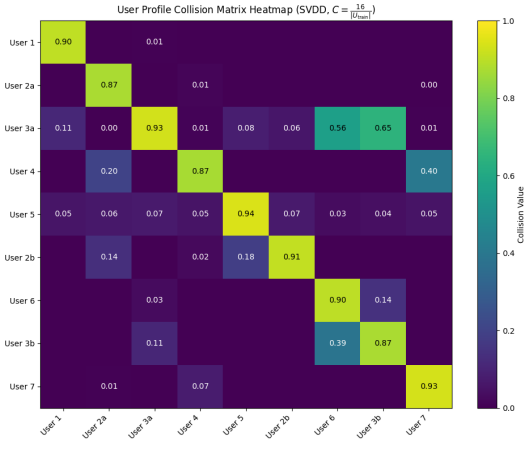
User	1	2-a	3-a	4	5	2-b	6	3-b	7
User 1	0.90	0.0	0.01	0.0	0.0	0.0	0.0	0.0	0.0
User 2-a	0.0	0.87	0.0	0.01	0.0	0.0	0.0	0.0	0.0
User 3-a	0.09	0.0	0.94	0.01	0.05	0.03	0.59	0.67	0.0
User 4	0.0	0.20	0.0	0.88	0.0	0.0	0.0	0.0	0.40
User 5	0.05	0.06	0.06	0.05	0.94	0.06	0.03	0.03	0.04
User 2-b	0.0	0.13	0.0	0.02	0.18	0.91	0.0	0.0	0.0
User 6	0.0	0.0	0.03	0.0	0.0	0.0	0.90	0.13	0.0
User 3-b	0.0	0.0	0.11	0.0	0.0	0.0	0.39	0.87	0.0
User 7	0.0	0.02	0.0	0.10	0.0	0.0	0.0	0.0	0.94

Table 4.10: User Profiles collision matrix using SVDD with $C = \frac{16}{|U_{\text{train}}|}$.

heatmaps we see that for $C = 0.05$, Figure 4.10a, User 1 and User 5 profile estimates score better, while for Users 2a, 3a, 4, 2b, 6, 3b and 7 taking $C = \frac{16}{|U_{\text{train}}|}$ is advantageous. For User 3a in particular this increased TAR score comes at a cost, i.e. higher FAR scores, see Figure 4.11b.



(a) Heatmap of SVDD collision matrix with $C = 0.05$ for all user profiles.



(b) Heatmap of SVDD collision matrix with $C = \frac{16}{|U_{\text{train}}|}$ for all user profiles.

Figure 4.10: Heatmaps of SVDD collision matrices for Phase 2. Left $C = 0.05$ and right $C = \frac{16}{|U_{\text{train}}|}$.

4.3.3. MGE method

Since the sizes of the training set per user are (mostly) larger than in the first phase, it seemed worthwhile to also compute the collision matrices using the one-class classification density method Multivariate-Gaussian Estimation. In Table 4.11 and Table 4.12 collision matrices are listed for hyper-parameter $\alpha = 0.05$ and $\alpha = 0.1$ respectively. To compare the TAR/FAR scores more easily, we also

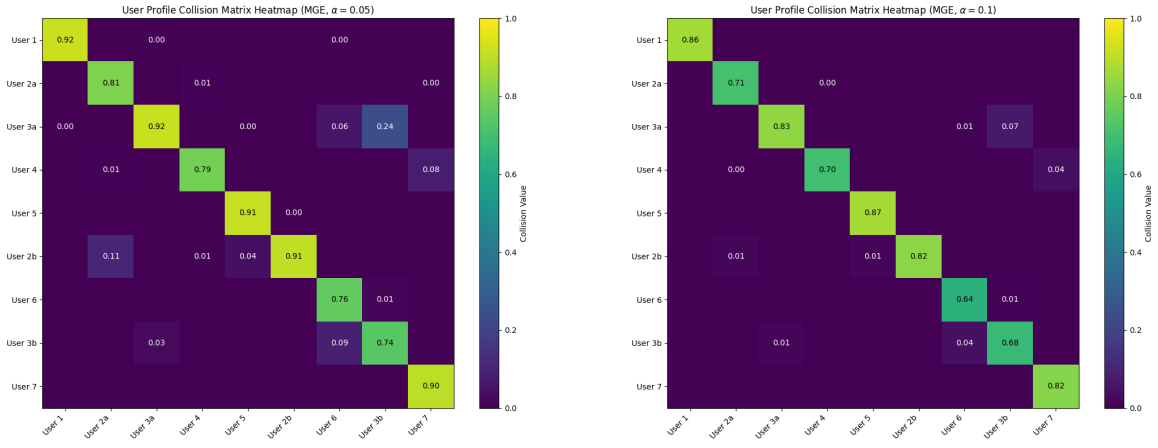
User	1	2-a	3-a	4	5	2-b	6	3-b	7
1	0.92	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2-a	0.0	0.81	0.0	0.01	0.0	0.0	0.0	0.0	0.0
3-a	0.0	0.0	0.91	0.0	0.0	0.0	0.06	0.24	0.0
4	0.0	0.01	0.0	0.79	0.0	0.0	0.0	0.0	0.08
5	0.0	0.0	0.0	0.0	0.91	0.0	0.0	0.0	0.0
2-b	0.0	0.11	0.0	0.01	0.04	0.91	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	0.76	0.01	0.0
3-b	0.0	0.0	0.03	0.0	0.0	0.0	0.09	0.74	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.9

Table 4.11: User Profiles collision matrix for 5-percentile MGE thresholds (i.e. $\alpha = 0.05$). Each cell contains a (rounded) percentage of total possible collision is given where the row indicates the MGE of a user and the column user data points used to compute a collision.

User	1	2-a	3-a	4	5	2-b	6	3-b	7
1	0.86	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2-a	0.0	0.71	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3-a	0.0	0.0	0.83	0.0	0.0	0.0	0.01	0.07	0.0
4	0.0	0.0	0.0	0.7	0.0	0.0	0.0	0.0	0.04
5	0.0	0.0	0.0	0.0	0.87	0.0	0.0	0.0	0.0
2-b	0.0	0.01	0.0	0.0	0.01	0.83	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	0.64	0.01	0.0
3-b	0.0	0.0	0.01	0.0	0.0	0.0	0.04	0.68	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.82

Table 4.12: User Profiles collision table for 10-percentile MGE thresholds. Each cell contains a (rounded) percentage of total possible collision is given where the row indicates the MGE of a user and the column user data points used to compute a collision.

provide the collision matrix data as heatmaps, see Figure 4.11. For both $\alpha = 0.05$ and $\alpha = 0.1$ we see low FAR scores. In the case of $\alpha = 0.1$ none of the TAR scores exceed 0.9, which indicates too many user data points are cut-off when estimating the profile.



(a) Heatmap of MGE collision matrix with $\alpha = 0.05$ for all user profiles.

(b) Heatmap of MGE collision matrix with $\alpha = 0.1$ for all user profiles.

Figure 4.11: Heatmaps of MGE collision matrices. Left $\alpha = 0.05$ and right $\alpha = 0.1$.

4.3.4. Increased dimensionality for supporting devices

Upon further analysis of the data, we found that three participants of 'Phase 2' of data collection did have mobile devices which support touch pressure data collection. Due to technical issues, one of these three user had to switch to another Asset during Phase 2, denoted with '3-a' and '3-b'. Contrary to phase 1, we could now compare multiple user profiles in both 29 and 24 dimensions. Generally, we would expect better performance in terms of TAR-FAR trade-off.

Table 4.13 and Table 4.15 show TAR (on diagonal) and FAR (on off-diagonal) scores for MGE method with 5-percentile threshold considering 29 and 24 dimensions respectively. Comparing these tables we indeed see indeed a reduction in FAR scores. For example the FAR score of User 3-a on User 1's profile becomes negligible if all 29 dimensions are used in building the user profile. Let us now address the exceptions.

For user 3-a's profile we actually see a worse FAR score on User 6's data in 29 dimensions, compared to 24 dimension. With a FAR score increase from roughly 0.081 to 0.306 we exceed any acceptable FAR error margin. We can however increase to 10-percentile, see Table 4.14 to bring the FAR score back down, at the cost of a lower TAR score.

Moreover, substantial FAR rates for User 3-b data on User 3-a could be an indication, that some behaviour unique to User Ni is invariant under PIN sequence, since 3-a and 3-b had a distinct custom PIN layout. If so, that would of course be a very useful result indeed.

Nevertheless, it remains worthwhile to explain this counter-intuitive behaviour more in-depth. Within the scope of the thesis further analysis on this matter was not feasible due to time constraints. It does provide us with a cautionary tale that simply increasing dimensions does not guarantee better TAR/FAR rates for user profiles.

	User 1	User 3-a	User 6	User 3-b
1	0.949	0.0	0.0	0.0
3-a	0.0	0.95	0.306	0.489
6	0.0	0.0	0.946	0.0146
3-b	0.0	0.0233	0.0901	0.949

Table 4.13: MGE 29 Dimensions ($\alpha = 0.05$)

	User 1	User 3-a	User 6	User 3-b
1	0.899	0.0	0.0	0.0
3-a	0.0	0.899	0.00901	0.102
6	0.0	0.0	0.892	0.0073
3-b	0.0	0.00775	0.036	0.898

Table 4.14: MGE 29 Dimensions ($\alpha = 0.1$)

	User 1	User 3-a	User 6	User 3-b
1	0.949	0.00388	0.0	0.0
3-a	0.00461	0.95	0.0811	0.321
6	0.0	0.0	0.946	0.0146
3-b	0.0	0.031	0.126	0.949

Table 4.15: MGE 24 Dimensions ($\alpha = 0.05$)

	User 1	User 3-a	User 6	User 3-b
1	0.899	0.0	0.0	0.0
3-a	0.0	0.899	0.00901	0.073
6	0.0	0.0	0.892	0.0146
3-b	0.0	0.0155	0.0721	0.898

Table 4.16: MGE 24 Dimensions ($\alpha = 0.1$)

4.4. Method comparison

In this section we compare the collision matrices between three one-class classification methods; k-MEB, SVDD and MGE. Since the collected user data sets, in both phases, are limited in size, all methods are viable computation wise. Nevertheless it is good to note that with our used implementations that profiles are faster to estimate with MGE than SVDD. In turn, SVDD profile estimation are much faster to compute than k-MEB, which is not surprising as it considers all possible subsets of certain size of the user data training set. So, while k-MEB allows for very precise exclusion of anomalies, it still remains to be seen if it outperforms SVDD and MGE. To that end, for each phase we list which method performs 'best' for a given user. In this comparison we assume the maximum FAR score allowed for a user profile estimation is 0.1 at most. The method which achieves the highest TAR score under such condition is listed as 'best performing'. In Table 4.17 we list the 'best performing' method for each user. For reference we also note the user training data set size as well as the method's tuned hyperparameter value and the respective TAR and FAR scores. In five out of all nine users the SVDD method came out on top. In addition, for User 6 and User 7 the SVDD method as good as tied with the best performing method. Only for User 1 the MGE method resulted in the 'best' TAR/FAR trade-off. For User 4 the k-MEB was leading with a TAR score of 0.82 and TAR score of 0.09, given tuned hyperparameter $\kappa = 0.09$. In all cases, the best performing method at least produced a TAR score of 0.8. Where SVDD

User	$\lvert U_{\text{train}}\rvert$	Method	Hyperparam.	TAR	max(FAR)
1	200	MGE	$\alpha = 0.1$	0.85	0.09
2	49	SVDD	$C = 0.15$	0.8	0.08
3	190	SVDD	$C = 0.09$	0.93	0.08
4	71	k-MEB	$\kappa = 0.93$	0.82	0.09
5	56	SVDD	$C = 0.286$	0.84	0.0
6	50	k-MEB[2]	$\kappa = 0.95$	0.80	0.03
7	447	MGE[3]	$\alpha = 0.05$	0.93	0.06
8	157	SVDD	$C = 0.12$	0.95	0.0
9	83	SVDD	$C = 0.21$	0.89	0.0

Table 4.17: Table of best performing method after tuning, for each user of phase 1.

was best performing for most users in phase 1, this is not the case for phase 2. The 'best performing' methods for phase 2 per user are listed in Table 4.18. In phase 2, the MGE method is best performing for all but one user, although for user 1 and user 7 it is tied with k-MEB. It seems that for MGE taking $\alpha = 0.05$ is suitable choice in most cases, which results in low FAR scores with decent TAR scores. For User 5 all three methods perform the same when tuned properly. With exception of User 3b the TAR scores are around 0.9 with User 5 even reaching a TAR score of 0.97. TAR scores above 0.97 are, in essence, not even desirable as the user profile would also accept anomalies which we try to avoid.

User	$\lvert U_{\text{train}}\rvert$	Method	Hyperparam.	TAR	max(FAR)
1	347	MGE [4]	$\alpha = 0.03$	0.94	0.02
2a	94	MGE	$\alpha = 0.02$	0.91	0.09
3a	206	MGE	$\alpha = 0.05$	0.91	0.06
4	102	MGE	$\alpha = 0.05$	0.89	0.08
5	375	MGE[5]	$\alpha = 0.02$	0.97	0.01
2b	160	MGE	$\alpha = 0.04$	0.9	0.06
6	88	k-MEB	$\kappa = 0.97$	0.87	0.05
3b	109	MGE	$\alpha = 0.05$	0.73	0.08
7	161	MGE [6]	$\alpha = 0.03$	0.95	0.03

Table 4.18: Table of best performing method after tuning, for each user of phase 2.

[2] SVDD got very close with TAR score of 0.78 for a slightly lower FAR of 0.02.
[3] SVDD got very close with TAR score of 0.92 for the same max FAR.
[4] Tied with k-MEB and SVDD with same TAR/FAR scores
[5] Tied with k-MEB
[6] Tied with k-MEB and SVDD with same TAR/FAR scores

5

Conclusion

In this research we investigated how (well) behavioural biometrics can strengthen PIN authentication, in particular keystroke dynamics. We implemented functionality to collected and extract behavioural data via Ubiqu's Authenticate App[1] on mobile (android) devices of participating users. With the collected user data we want to estimate its profile. Initial feature analysis showed that the time between digit taps on the PIN pad summed is an informative derived feature. Moreover, it allowed to mark a point of transition into stable behaviour.

The next step was to estimate the user profile as a whole, based on the data assumed to part of stable behaviour. Given limited variation in features for PIN authentication, combined with the small number of features, we would expect at least some overlap between users. We constructed user profiles via three different one-class classifier methods; Minimum k-Enclosing Ball (k-MEB), Support Vector Data Description (SVDD) and Multivariate-Gaussian Estimation (MGE). All three methods are tunable with their respective one-dimensional hyperparameter k, C and α.

Ideally, we would want to find user profiles to be fully unique, meaning they do not have overlap at all. After analysing data from phase 1 we found this was not the case. Our results show that estimated user profiles, using any of our three OCC methods, are generally not disjoint. In phase 2, custom user layouts did reduce overlap, as expected, but overlap still persisted. In other words, based on our research we did not find evidence to claim a user profile can always be uniquely defined.

However, while disjointness is sufficient for ensuring secure authentication, it is not a necessary condition. The number of users which completed over 100 valid transactions is not a large enough sample to back up any claims on the likelihood of user profiles having overlap. Even so, it is interesting to have identified ample users who appear to hardly have any overlap (or none at all) with other users. Further (empirical) research on many users could help to determine a potential security risk for an arbitrary user. Meaning, if overlap between user profiles is adequately sparse it could compensate for such overlap in practice.

All method which yielded roughly the same amount of profile collisions ($\sim$13 collisions) in phase 1, of which around three above the 0.1 collision rate (FAR). Comparing the methods we found all three yielded promising results. Since we are interested in the TAR/FAR trade-off, when comparing methods we look at the biggest difference between the TAR and FAR score for an estimated user profile. Meaning the method with the largest value of $\mathrm{TAR} - \max(\mathrm{FAR})$ we call 'best performing'. For the user data samples of phase 1 we noticed (tuned) SVDD often offered the greatest FAR-TAR trade-off of the methods considered. In both phase 1 and 2 the k-MEB method did better on user data sets of smaller size. In phase 2, after tuning, MGE provided the highest TAR scores, when allowing the maximum FAR score to at most 0.1. In both phases we managed to tune the OCC methods to obtain TAR scores greater than 0.8, whilst keeping FAR scores below 0.1. The only exception herein is user profile 3-b in phase 2 with 0.73.

All results in this research have been obtained by constructing profiles in the standard (feature) basis. For k-MEB and SVDD, the sphere has been oriented according to the feature axes. Alternatively, one can use another basis to describe sphere before transforming it back to an ellipsoid. A common basis is

[1] a variant of the app was build accessible to company staff only

the eigenvector basis of the covariance matrix, also known as principal axes. Due to time constraints, this way of estimating the user profile we only investigated briefly. The estimated profiles obtained through an eigenvector basis resulted in collision matrices with higher FAR scores, compared to the standard basis. Further analysis is required to postulate any claims regarding this matter.

Strengthening PIN authentication using Behavioural Biometrics we argue to be effective based on our results. It outperforms dummy classifiers and matches or exceeds previous results in this area of research. Depending on the specific security requirements different measures can be taken when a user is, possibly falsely, rejected by this augmented PIN authentication process. In such cases, ‘proof-stacking’ (a.o. multifactor-authentication) is commonly used to allow users to continue the authentication process.

6

Discussion

Behavioural biometrics is maturing field of research. Nevertheless there are still many interesting questions to pursue. In this thesis we look into behavioural biometrics for PIN authentication. A comparison between different authentication methods in terms of their ability to capture learnt behaviour more uniquely appears to be a valid way forward.

To estimate user profiles, we both boundary and density one-class classification method. Building user profiles using reconstruction methods remains unknown. For computing overlap between profiles, we resorted to sampling using training and test splits on the user data set. Using k-fold cross-validation we reduced the variance of our overlap estimations. However, other overlap measures might provide a more exact estimate herein.

Moreover, variants such as a moving profile should be investigated, where over time older data points are disregarded in favour of newest point. This assumes that (stable) behaviour can evolve over longer time periods. Theoretically a moving profile should be more accurately, but requires periodic recalibration of the classifier model. This implies shaping time-dependent user profile model.

Tying into changing behaviour states, progress could be made on capturing a user profile in terms of separate states. Similarly to drunk-driving tests, different 'moods' can be identified and correctly asserted. Possibly, by separating clusters to obtain a tighter user profile for each mood, both the false acceptance rates can be reduced whilst improving true acceptance rates. One can think of clustering a given data sample with K-MEB, under the assumption of K 'moods', with some minimisation condition, for example smallest volume sum. Yet, even without accounting for different behavioural states, it seems a worthwhile effort either rule out or confirm that for authentication methods multiple small clusters can also capture a user profile.

Lastly, more in-depth analysis of user profile shapes would be desirable to increase the modelling. In this thesis we limited ourselves to a Gaussian assumption due to constructing a profile from a sample. More elaborate methods might yield far better results. A challenging task given the vast amount of different types of boundaries, kernels and parameters available for state-of-the-art one-class classifiers. Moreover, if the negative examples (data from other users) can be utilized multi-class classifiers definitely will reduce the FAR. It might also provide a way forward to develop authentication flows which inherently minimize FAR, while maintaining decent TAR scores.

References

[1] Peter Adby. *Introduction to optimization methods*. Springer Science & Business Media, 2013.

[2] Israa Majeed Alsaadi. “Study on most popular behavioral biometrics, advantages, disadvantages and recent applications: A review”. In: *Int. J. Sci. Technol. Res* 10.1 (2021).

[3] Tomoko Aoki and Hiroshi Kinoshita. “Temporal and force characteristics of fast double-finger, single-finger and hand tapping”. In: *Ergonomics* 44 (Jan. 2002), pp. 1368–83. DOI: 10.1080/00140130110107452.

[4] Ramamurthy Arunachalam, Vajira S Weerasinghe, and Kerry R Mills. “Motor control of rapid sequential finger tapping in humans”. In: *Journal of neurophysiology* 94.3 (2005), pp. 2162–2170.

[5] Çağatay Barut et al. “Advanced Analysis of Finger-Tapping Performance: A Preliminary Study”. In: *Balkan Medical Journal* 2013 (2013), pp. 167–171. DOI: 10.5152/balkanmedj.2012.105.

[6] Nick Berry. *DataGenetics - PIN analysis*. Accessed on Sept 20th, 2024. 2012. URL: http://www.datagenetics.com/blog/september32012/.

[7] Debnath Bhattacharyya et al. “Biometric authentication: A review”. In: *International Journal of u-and e-Service, Science and Technology* 2.3 (2009), pp. 13–28.

[8] Christopher M. Bishop. *Neural networks for pattern recognition*. Oxford university press, 1995.

[9] Christopher M. Bishop and Nasser M. Nasrabadi. *Pattern recognition and machine learning*. Vol. 4. 4. Springer, 2006.

[10] Abdenour Bounsiar and Michael G. Madden. “One-Class Support Vector Machines Revisited”. In: *2014 International Conference on Information Science & Applications (ICISA)*. 2014, pp. 1–4. DOI: 10.1109/ICISA.2014.6847442.

[11] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[12] Daniel A Braun et al. “Motor task variation induces structural learning”. In: *Current Biology* 19.4 (2009), pp. 352–357.

[13] Xavier Bultel et al. “Security analysis and psychological study of authentication methods with PIN codes”. In: *2018 12th International Conference on Research Challenges in Information Science (RCIS)*. 2018, pp. 1–11. DOI: 10.1109/RCIS.2018.8406648.

[14] Leonardo Causa et al. “Analysis of behavioural curves to classify iris images under the influence of alcohol, drugs, and sleepiness conditions”. In: *Expert Systems with Applications* 242 (2024), p. 122808. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2023.122808. URL: https://www.sciencedirect.com/science/article/pii/S0957417423033109.

[15] Marta Cavaleiro and Farid Alizadeh. “A branch-and-bound method for the minimum k- enclosing ball problem”. In: *Operations Research Letters* 50.3 (2022), pp. 274–280.

[16] Timothy M. Chan. “Improved Deterministic Algorithms for Linear Programming in Low Dimensions”. In: *ACM Trans. Algorithms* 14.3 (June 2018). ISSN: 1549-6325. DOI: 10.1145/3155312. URL: https://doi-org.tudelft.idm.oclc.org/10.1145/3155312.

[17] Nathan L Clarke, Steven M Furnell, and Paul L Reynolds. “Biometric authentication for mobile devices”. In: *Proceeding of the 3rd Australian information warfare and security conference*. 2002, pp. 61–69.

[18] SciPy Community. *Constrained Minimization — SciPy v1.15.2 (Stable) Manual*. Accessed: 2025-04-10. 2025. URL: https://docs.scipy.org/doc/scipy/tutorial/optimize.html#constrained-minimization.

[19] Kyzer R. Davis, Brad Peabody, and P. Leach. *Universally Unique IDentifiers (UUIDs)*. RFC 9562. May 2024. DOI: 10.17487/RFC9562. URL: https://www.rfc-editor.org/info/rfc9562.

[20] Will Dixon. *Windows Hello in Token, a biometric ring*. 2017. URL: https://blogs.windows.com/windowsexperience/2017/06/29/windows-hello-token-biometric-ring/ (visited on 09/23/2024).

[21] EU. *Becoming a (qualified) trust service provider*. 2025. URL: https://eidas.ec.europa.eu/efda/discover/becoming-qtsp (visited on 10/02/2024).

[22] Hans Fischer. *A history of the central limit theorem: from classical to modern probability theory*. Vol. 4. Springer, 2011.

[23] Kaspar Fischer, Bernd Gärtner, and Martin Kutz. *Fast Smallest-Enclosing-Ball Computation in High Dimensions*. Ed. by Giuseppe Di Battista and Uri Zwick. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 630–641. ISBN: 978-3-540-39658-1.

[24] Igor Gilitschenski and Uwe D. Hanebeck. "A robust computational test for overlap of two arbitrary-dimensional ellipsoids in fault-detection of kalman filters". In: *2012 15th International Conference on Information Fusion*. IEEE. 2012, pp. 396–401.

[25] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples". In: *arXiv preprint arXiv:1412.6572* (2014).

[26] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*. Vol. 2. Springer Science & Business Media, 2012.

[27] Honghao Guo et al. "Fooling A Deep-Learning Based Gait Behavioral Biometric System". In: *2020 IEEE Security and Privacy Workshops (SPW)*. 2020, pp. 221–227. DOI: 10.1109/SPW50608.2020.00052.

[28] Gary B. Hughes and Mohcine Chraibi. *Calculating ellipse overlap areas*. 2011. arXiv: 1106.3787 [physics.comp-ph]. URL: https://arxiv.org/abs/1106.3787.

[29] Emin Huseynov. "User Behaviour in Choosing PIN Codes for FIDO2 Security Keys: An Empirical Study and a Market Research". In: *International Communication Engineering and Cloud Computing Conference (CECCC)* 6 (2024).

[30] Ákos Jobbágy et al. "Analysis of finger-tapping movement". In: *Journal of Neuroscience Methods* 141.1 (2005), pp. 29–39. ISSN: 0165-0270. DOI: https://doi.org/10.1016/j.jneumeth.2004.05.009. URL: https://www.sciencedirect.com/science/article/pii/S016502700400202X.

[31] Linus Källberg. "Minimum enclosing balls and ellipsoids in general dimensions". PhD thesis. Mälardalen University, 2019.

[32] Seokho Kang. "Using binary classifiers for one-class classification". In: *Expert Systems with Applications* 187 (2022), p. 115920. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2021.115920. URL: https://www.sciencedirect.com/science/article/pii/S0957417421012744.

[33] Christina Katsini et al. "Security and Usability in Knowledge-based User Authentication: A Review". In: *Proceedings of the 20th Pan-Hellenic Conference on Informatics*. PCI '16. Patras, Greece: Association for Computing Machinery, 2016. ISBN: 9781450347891. DOI: 10.1145/3003733.3003764. URL: https://doi-org.tudelft.idm.oclc.org/10.1145/3003733.3003764.

[34] Michael J Kearns and Umesh Vazirani. *An introduction to computational learning theory*. MIT press, 1994.

[35] Shehroz S. Khan and Michael G. Madden. "One-class classification: taxonomy of study and review of techniques". In: *The Knowledge Engineering Review* 29.3 (2014), pp. 345–374. DOI: 10.1017/S026988891300043X.

[36] Hyoungshick Kim and Jun Ho Huh. "PIN selection policies: Are they really effective?" In: *computers & security* 31.4 (2012), pp. 484–496.

[37] Jun-ichi Kitamura, Hiroshi Shibasaki, and Tohru Kondo. "A cortical slow potential is larger before an isolated movement of a single finger than simultaneous movement of two fingers". In: *Electroencephalography and Clinical Neurophysiology* 86.4 (1993), pp. 252–258. ISSN: 0013-4694. DOI: https://doi.org/10.1016/0013-4694(93)90106-6. URL: https://www.sciencedirect.com/science/article/pii/0013469493901066.

[38] Joseph Lange et al. “Side-Channel Attacks Against the Human Brain: The PIN Code Case Study”. In: *Constructive Side-Channel Analysis and Secure Design*. Ed. by Sylvain Guilley. Cham: Springer International Publishing, 2017, pp. 171–189. ISBN: 978-3-319-64647-3.

[39] Mingting Liu et al. “Explanation-Guided Minimum Adversarial Attack”. In: *Machine Learning for Cyber Security*. Ed. by Yuan Xu et al. Cham: Springer Nature Switzerland, 2023, pp. 257–270. ISBN: 978-3-031-20096-0.

[40] Jagex Ltd. *Bank PIN interface [Screenshot]*. Copyrighted material used under fair use for illustrative/educational purposes. 2025. URL: https://runescape.wiki/w/Bank_PIN.

[41] Tempestt Neal and Damon Woodard. “Presentation Attacks in Mobile and Continuous Behavioral Biometric Systems”. In: *Securing Social Identity in Mobile Platforms: Technologies for Security, Privacy and Identity Management*. Ed. by Thirimachos Bourlai, Panagiotis Karampelas, and Vishal M. Patel. Cham: Springer International Publishing, 2020, pp. 21–40. ISBN: 978-3-030-39489-9. DOI: 10.1007/978-3-030-39489-9_2. URL: https://doi.org/10.1007/978-3-030-39489-9_2.

[42] NordPass. *What is Multi-Factor Authentication?* 2024. URL: https://nordpass.com/blog/what-is-multi-factor-authentication/ (visited on 09/20/2024).

[43] L. O’Gorman. “Comparing passwords, tokens, and biometrics for user authentication”. In: *Proceedings of the IEEE* 91.12 (2003), pp. 2021–2040. DOI: 10.1109/JPROC.2003.819611.

[44] Aleksandr Ometov et al. “Multi-factor authentication: A survey”. In: *Cryptography* 2.1 (2018), p. 1.

[45] Juri Opitz. “A Closer Look at Classification Evaluation Metrics and a Critical Reflection of Common Evaluation Practice”. In: *Transactions of the Association for Computational Linguistics* 12 (June 2024), pp. 820–836. ISSN: 2307-387X. DOI: 10.1162/tacl_a_00675. eprint: https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl_a_00675/2465598/tacl_a_00675.pdf. URL: https://doi.org/10.1162/tacl%5C_a%5C_00675.

[46] Nima Rabiei and Elias G Saleeby. “On intersection volumes of confidence hyper-ellipsoids and two geometric Monte Carlo methods”. In: *Monte Carlo Methods and Applications* 27.2 (2021), pp. 153–167.

[47] Aratrika Ray-Dowling, Daqing Hou, and Stephanie Schuckers. “Stationary mobile behavioral biometrics: A survey”. In: *Computers & Security* 128 (2023), p. 103184. ISSN: 0167-4048. DOI: https://doi.org/10.1016/j.cose.2023.103184. URL: https://www.sciencedirect.com/science/article/pii/S0167404823000949.

[48] Peter Y. A. Ryan. “Mathematical Models of Computer Security”. In: *Foundations of Security Analysis and Design*. Ed. by Riccardo Focardi and Roberto Gorrieri. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 1–62. ISBN: 978-3-540-45608-7.

[49] Bernhard Schölkopf, Koji Tsuda, and Jean-Philippe Vert. *Kernel methods in computational biology*. MIT press, 2004.

[50] Bernhard Schölkopf et al. “Support vector method for novelty detection”. In: *Advances in neural information processing systems* 12 (1999).

[51] X. Shang et al. “Algorithm Design for Grip-Pattern Verification in Smart Gun”. English. In: *ProRISC 2005, 16th Workshop on Circuits, Systems and Signal Processing*. 16th Workshop on Circuits, Systems and Signal Processing, ProRISC 2005 ; Conference date: 17-11-2005 Through 18-11-2005. STW, 2005, pp. 674–678. ISBN: 90-73461-50-2.

[52] Heather Shaw et al. “Digital Behavioural Biometrics: A Review of Reviews”. In: (Sept. 2024). DOI: 10.31234/osf.io/vs2ut. URL: https://osf.io/preprints/psyarxiv/vs2ut.

[53] V.V. Shenmaier. “The problem of a minimal ball enclosing k points”. In: *Journal of Applied and Industrial Mathematics* 7.3 (2013), pp. 444–448.

[54] Daniel Staps, Thomas Villmann, and Benjamin Paaßen. “K Minimum Enclosing Balls for Outlier Detection”. In: *International Workshop on Self-Organizing Maps, Learning Vector Quantization & Beyond*. Springer. 2024, pp. 174–184.

[55] Ioannis Stylios et al. “Behavioral biometrics & continuous user authentication on mobile devices: A survey”. In: *Information Fusion* 66 (2021), pp. 76–99. ISSN: 1566-2535. DOI: https://doi.org/10.1016/j.inffus.2020.08.021. URL: https://www.sciencedirect.com/science/article/pii/S1566253520303493.

[56] David M.J. Tax. “One-class classification; concept-learning in the absence of counter-examples”. English. ASCI Dissertation Series 65. Dissertation (TU Delft). Delft University of Technology, 2001. ISBN: 90-75691-05-X.

[57] David M.J. Tax. “One-class classification: Concept learning in the absence of counter-examples.” PhD thesis. TU Delft, 2002.

[58] David M.J. Tax and Robert P.W. Duin. “Support vector data description”. In: *Machine learning* 54 (2004), pp. 45–66.

[59] Michal Trnka et al. “Systematic review of authentication and authorization advancements for the Internet of Things”. In: *Sensors* 22.4 (2022), p. 1361.

[60] Ubiqu. *eIDAS 2.0 – Ubiqu*. 2025. URL: https://ubiqu.com/eidas-2-0/ (visited on 12/20/2024).

[61] Ubiqu. *QTSP – Ubiqu*. 2025. URL: https://ubiqu.com/what-is-an-qtsp/ (visited on 12/20/2024).

[62] Moscow State University. *Ellipsoidal Calculus*. 2024. URL: https://systemanalysisdpt-cmc-msu.github.io/ellipsoids/doc/chap_ellcalc.html (visited on 10/20/2024).

[63] Michael N. Vrahatis. *Towards the mathematical foundation of the minimum enclosing ball and related problems*. 2024. arXiv: 2402.06629 [cs.CG]. URL: https://arxiv.org/abs/2402.06629.

[64] Shunyao Wang et al. “Evasion Attack and Defense on Machine Learning Models in Cyber-Physical Systems: A Survey”. In: *IEEE Communications Surveys & Tutorials* 26.2 (2024), pp. 930–966. DOI: 10.1109/COMST.2023.3344808.

[65] Emo Welzl. “Smallest enclosing disks (balls and ellipsoids)”. In: *New Results and New Trends in Computer Science*. Ed. by Hermann Maurer. Berlin, Heidelberg: Springer Berlin Heidelberg, 1991, pp. 359–370. ISBN: 978-3-540-46457-0.

[66] Stephen J Wright. “Coordinate descent algorithms”. In: *Mathematical programming* 151.1 (2015), pp. 3–34.

[67] Roman V. Yampolskiy. “Mimicry Attack on Strategy-Based Behavioral Biometric”. In: *Fifth International Conference on Information Technology: New Generations (itng 2008)*. 2008, pp. 916–921. DOI: 10.1109/ITNG.2008.78.

[68] Yang Zhang et al. “Fingerprint attack against touch-enabled devices”. In: *Proceedings of the Second ACM Workshop on Security and Privacy in Smartphones and Mobile Devices*. SPSM ’12. Raleigh, North Carolina, USA: Association for Computing Machinery, 2012, pp. 57–68. ISBN: 9781450316668. DOI: 10.1145/2381934.2381947. URL: https://doi-org.tudelft.idm.oclc.org/10.1145/2381934.2381947.

A

Data Set I

0.023529414	131.8789	55.898438	70.0	0.03137255	239.63477	63.484375	92.0	216.0	0.027450982	171.91406	97.02344	65.0	266.0	0.023529414	136.53906	85.16406	58.0	267.0	0.037254903	165.89258	75.82031	49.0	232.0
0.033333335	212.29883	56.484375	107.0	0.037254903	145.5039	105.671875	50.0	92.0	0.027450982	213.04688	65.38281	66.0	182.0	0.037254903	167.125	73.44531	49.0	167.0	0.04117647	201.75195	52.382812	83.0	117.0
0.033333335	216.51758	66.44531	73.0	0.043137256	142.07617	80.47656	50.0	307.0	0.033333335	262.08984	63.039062	50.0	149.0	0.037254903	153.67773	63.484375	49.0	258.0	0.025490198	219.68164	44.765625	92.0	91.0
0.023529414	214.67188	37.734375	115.0	0.037254903	152.0957	77.546875	33.0	390.0	0.03137255	220.42969	46.046875	66.0	208.0	0.050980397	139.43945	76.375	33.0	192.0	0.037254903	212.5625	24.257812	58.0	166.0
0.04117647	217.57227	61.171875	97.0	0.027450982	149.98633	55.867188	74.0	167.0	0.023529414	240.73242	11.4765625	74.0	150.0	0.037254903	152.62305	45.320312	50.0	184.0	0.050980397	206.76172	39.492188	74.0	157.0
0.037254903	194.63281	71.71875	72.0	0.03529412	277.07617	71.10156	66.0	292.0	0.023529414	211.72852	58.351562	25.0	107.0	0.023529414	299.75195	75.78906	58.0	108.0	0.027450982	185.66797	37.734375	49.0	275.0
0.033333335	213.08984	47.695312	107.0	0.03529412	144.97656	56.453125	41.0	100.0	0.027450982	255.76172	2.6875	66.0	150.0	0.04117647	138.38477	88.09375	50.0	142.0	0.03137255	189.35938	50.625	75.0	108.0
0.027450982	190.9414	44.179688	82.0	0.03529412	160.00586	76.375	58.0	100.0	0.027450982	229.39453	35.5	49.0	133.0	0.054901965	168.70703	44.734375	57.0	134.0	0.021568628	194.63281	37.148438	58.0	109.0
0.033333335	222.8457	36.5625	81.0	0.03137255	136.53906	75.203125	49.0	125.0	0.027450982	224.38477	19.09375	65.0	168.0	0.050980397	126.7832	68.75781	42.0	167.0	0.037254903	216.2539	47.695312	74.0	141.0
0.027450982	238.66602	38.320312	66.0	0.05294118	172.13477	68.75781	58.0	133.0	0.023529414	247.58789	19.679688	75.0	149.0	0.037254903	149.98633	53.523438	66.0	150.0	0.050980397	203.33398	39.492188	66.0	158.0
0.025490198	211.77148	43.007812	90.0	0.029411767	180.83594	58.210938	49.0	117.0	0.027450982	217.79297	63.039062	49.0	208.0	0.03529412	154.20508	63.484375	42.0	175.0	0.023529414	216.51758	45.351562	66.0	108.0
0.021568628	198.32422	98.671875	82.0	0.029411767	160.5332	102.15625	67.0	58.0	0.03137255	186.41602	87.64844	49.0	183.0	0.060784318	159.74219	39.460938	41.0	167.0	0.029411767	225.21875	27.773438	82.0	117.0
0.03137255	196.21484	74.0625	104.0	0.037254903	166.86133	69.34375	58.0	75.0	0.027450982	217.79297	40.1875	66.0	133.0	0.021568628	116.5	58.210938	49.0	142.0	0.03529412	187.77734	68.78906	74.0	142.0
0.023529414	204.65234	64.6875	90.0	0.023529414	183.47266	20.710938	50.0	132.0	0.023529414	250.22461	74.171875	57.0	150.0	0.033333335	173.18945	76.96094	32.0	151.0	0.021568628	217.04492	79.921875	66.0	125.0
0.029411767	218.36328	91.05469	65.0	0.03137255	156.31445	92.19531	66.0	141.0	0.023529414	267.0996	42.53125	66.0	117.0	0.03529412	144.71289	86.33594	49.0	142.0	0.021568628	187.77734	41.835938	75.0	108.0
0.029411767	215.19922	67.03125	75.0	0.033333335	200.61133	90.4375	57.0	158.0	0.015686275	270.26367	33.15625	75.0	133.0	0.029411767	115.18164	75.78906	58.0	125.0	0.033333335	224.95508	54.726562	74.0	108.0
0.027450982	188.56836	71.71875	74.0	0.033333335	176.35352	88.09375	74.0	175.0	0.019607844	272.10938	49.5625	58.0	109.0	0.027450982	166.07031	65.24219	58.0	141.0	0.03137255	203.07031	75.82031	58.0	108.0
0.027450982	200.69727	112.734375	90.0	0.03137255	169.76172	78.71875	58.0	100.0	0.03137255	224.1211	58.351562	66.0	125.0	0.03529412	153.15039	56.453125	58.0	141.0	0.039215688	165.6289	43.59375	57.0	109.0
0.03137255	177.23047	53.554688	74.0	0.037254903	187.95508	93.36719	66.0	91.0	0.021568628	242.05078	29.054688	49.0	167.0	0.021568628	201.13867	99.8125	57.0	167.0	0.019607844	164.04688	45.9375	75.0	100.0
0.023529414	193.05078	57.65625	114.0	0.03529412	158.16016	77.546875	57.0	108.0	0.023529414	223.85742	78.859375	58.0	134.0	0.029411767	150.51367	99.8125	42.0	141.0	0.019607844	186.98633	65.859375	66.0	166.0
0.03137255	161.41016	51.796875	81.0	0.03137255	162.11523	48.835938	65.0	84.0	0.021568628	239.9414	29.054688	50.0	142.0	0.029411767	109.11719	68.75781	58.0	141.0	0.027450982	171.69336	48.867188	57.0	109.0
0.033333335	222.31836	112.14844	56.0	0.027450982	148.66797	94.53906	66.0	159.0	0.023529414	211.20117	47.21875	83.0	108.0	0.027450982	89.3418	114.46094	82.0	100.0	0.029411767	221.0	74.0625	66.0	117.0
0.027450982	178.28516	89.88281	66.0	0.019607844	131.5293	98.05469	66.0	133.0	0.03529412	222.53906	67.140625	66.0	125.0	0.037254903	115.18164	76.96094	57.0	133.0	0.021568628	176.9668	39.492188	58.0	101.0
0.03137255	183.03125	67.03125	107.0	0.027450982	133.63867	86.33594	66.0	83.0	0.027450982	231.24023	34.328125	59.0	133.0	0.03529412	142.33984	54.695312	58.0	149.0	0.019607844	157.98242	61.171875	57.0	67.0
0.029411767	212.03516	69.375	65.0	0.03529412	158.95117	66.41406	73.0	109.0	0.027450982	228.07617	84.71875	66.0	109.0	0.027450982	124.146484	125.59375	58.0	125.0	0.03529412	197.26953	64.10156	58.0	108.0
0.021568628	191.20508	94.57031	41.0	0.027450982	148.66797	87.50781	82.0	175.0	0.027450982	254.17969	29.054688	57.0	134.0	0.027450982	102.78906	71.6875	50.0	142.0	0.019607844	184.61328	69.375	66.0	99.0
0.027450982	204.65234	70.546875	124.0	0.033333335	192.70117	71.10156	66.0	117.0	0.021568628	234.93164	33.742188	58.0	125.0	0.015686275	110.43555	88.09375	66.0	133.0	0.033333335	160.35547	50.625	50.0	125.0
0.027450982	225.7461	71.13281	73.0	0.021568628	155.52344	98.640625	74.0	108.0	0.027450982	238.35938	43.117188	58.0	142.0	0.027450982	103.31641	96.296875	50.0	158.0	0.029411767	190.41406	36.5625	58.0	132.0
0.033333335	231.2832	106.875	57.0	0.03529412	160.5332	71.10156	57.0	100.0	0.033333335	194.58984	47.804688	58.0	150.0	0.03529412	138.38477	79.890625	57.0	117.0	0.027450982	202.2793	52.382812	49.0	84.0
0.027450982	165.36523	72.890625	89.0	0.033333335	172.39844	75.78906	58.0	108.0	0.021568628	231.5039	34.328125	57.0	134.0	0.03529412	139.43945	64.65625	65.0	126.0	0.033333335	185.31836	47.078125	24.0	67.0
0.039215688	212.29883	28.359375	80.0	0.027450982	205.35742	96.88281	65.0	67.0	0.03137255	214.89258	122.21875	74.0	142.0	0.043137256	181.09961	47.078125	49.0	134.0	0.037254903	156.92773	15.46875	57.0	76.0
0.023529414	197.00586	57.65625	115.0	0.021568628	196.65625	110.94531	83.0	50.0	0.03137255	254.17969	50.734375	57.0	125.0	0.03529412	141.02148	72.859375	49.0	109.0	0.033333335	155.3457	58.828125	50.0	50.0
0.019607844	213.35352	61.757812	82.0	0.03137255	188.48242	86.33594	74.0	100.0	0.027450982	237.56836	23.195312	66.0	125.0	0.033333335	148.14062	39.460938	50.0	133.0	0.027450982	186.19531	46.523438	57.0	50.0
0.025490198	215.19922	34.804688	72.0	0.03137255	156.8418	75.203125	65.0	108.0	0.021568628	224.91211	34.328125	50.0	142.0	0.019607844	105.68945	57.039062	57.0	117.0	0.03529412	181.18555	75.82031	42.0	67.0
0.029411767	215.19922	57.65625	82.0	0.029411767	175.29883	64.07031	83.0	108.0	0.021568628	236.25	40.773438	66.0	125.0	0.033333335	116.5	55.867188	67.0	133.0	0.027450982	179.33984	45.351562	66.0	66.0
0.027450982	202.01562	70.546875	64.0	0.03137255	144.71289	88.09375	66.0	108.0	0.027450982	224.1211	58.9375	58.0	133.0	0.03137255	134.95703	61.140625	49.0	125.0	0.027450982	166.41992	86.36719	58.0	50.0
0.021568628	228.64648	37.734375	107.0	0.027450982	172.66211	74.61719	75.0	58.0	0.027450982	220.16602	45.460938	66.0	125.0	0.027450982	131.5293	58.210938	58.0	108.0	0.027450982	174.85742	67.03125	57.0	75.0
0.03529412	183.29492	76.40625	81.0	0.015686275	181.36328	80.47656	82.0	76.0	0.021568628	278.17383	34.328125	58.0	117.0	0.023529414	48.208984	99.8125	58.0	116.0	0.033333335	170.11133	28.359375	57.0	67.0
0.033333335	209.39844	73.47656	65.0	0.03137255	154.9961	89.265625	82.0	84.0	0.027450982	247.06055	87.0625	66.0	109.0	0.037254903	175.03516	43.5625	41.0	125.0	0.027450982	174.59375	51.796875	58.0	58.0
0.027450982	211.77148	76.99219	91.0	0.029411767	180.83594	72.27344	66.0	91.0	0.03137255	240.20508	54.25	66.0	117.0	0.015686275	118.3457	98.640625	83.0	83.0	0.011764707	161.41016	66.44531	50.0	42.0
0.033333335	210.7168	81.67969	90.0	0.027450982	149.19531	95.125	66.0	83.0	0.027450982	226.75781	70.07031	58.0	117.0	0.029411767	111.490234	92.19531	66.0	100.0	0.023529414	164.04688	46.523438	57.0	59.0
0.019607844	191.9961	70.546875	73.0	0.015686275	126.7832	42.976562	75.0	66.0	0.023529414	220.42969	59.523438	58.0	125.0	0.033333335	112.01758	57.625	57.0	117.0	0.03137255	173.01172	61.171875	49.0	42.0
0.019607844	189.0957	80.50781	106.0	0.033333335	171.34375	94.53906	58.0	116.0	0.029411767	263.14453	77.6875	41.0	167.0	0.03529412	151.04102	59.382812	50.0	158.0	0.019607844	190.67773	68.203125	74.0	125.0
0.019607844	190.67773	71.71875	90.0	0.033333335	147.87695	41.804688	58.0	116.0	0.023529414	228.60352	61.28125	74.0	117.0	0.037254903	123.88281	64.07031	58.0	133.0	0.03137255	174.59375	53.554688	66.0	84.0
0.023529414	188.04102	52.96875	123.0	0.03137255	157.36914	78.13281	74.0	25.0	0.027450982	238.88672	48.390625	49.0	134.0	0.03529412	141.02148	68.75781	66.0	109.0	0.027450982	174.59375	43.007812	66.0	41.0
0.013725491	246.3125	57.65625	65.0	0.03529412	147.87695	77.546875	58.0	108.0	0.03137255	228.60352	81.203125	58.0	125.0	0.03137255	132.84766	90.4375	58.0	108.0	0.03529412	187.77734	60.585938	57.0	67.0
0.027450982	224.95508	68.203125	82.0	0.029411767	115.18164	88.67969	66.0	41.0	0.029411767	206.71875	48.390625	57.0	134.0	0.029411767	113.33594	92.78125	50.0	134.0	0.027450982	227.06445	70.546875	66.0	157.0
0.023529414	180.6582	75.234375	90.0	0.03137255	132.84766	91.02344	66.0	42.0	0.023529414	208.03711	48.390625	41.0	125.0	0.03529412	79.322266	71.6875	66.0	108.0	0.03137255	171.69336	57.65625	66.0	26.0
0.027450982	186.72266	78.75	91.0	0.03137255	153.15039	86.921875	49.0	58.0	0.027450982	232.03125	84.13281	58.0	117.0	0.03137255	144.18555	95.125	58.0	108.0	0.027450982	166.6836	75.82031	58.0	25.0
0.021568628	196.21484	43.007812	100.0	0.03529412	142.86719	76.375	57.0	83.0	0.027450982	250.75195	19.679688	33.0	159.0	0.03137255	137.0664	74.03125	58.0	141.0	0.03529412	182.24023	76.40625	66.0	92.0
0.027450982	203.07031	41.835938	82.0	0.021568628	142.86719	95.125	74.0	41.0	0.03137255	253.125	52.492188	58.0	117.0	0.03529412	148.14062	71.10156	58.0	109.0	0.03529412	186.98633	61.171875	49.0	41.0
0.021568628	208.08008	109.21875	90.0	0.03529412	146.82227	79.30469	57.0	58.0	0.027450982	238.35938	42.53125	41.0	134.0	0.03529412	148.14062	81.64844	58.0	100.0	0.033333335	179.86719	43.007812	42.0	50.0
0.029411767	213.61719	70.546875	107.0	0.021568628	156.8418	108.015625	49.0	25.0	0.027450982	230.44922	52.492188	41.0	142.0	0.027450982	176.35352	60.554688	57.0	109.0	0.027450982	170.90234	69.375	66.0	34.0
0.023529414	170.11133	88.71094	123.0	0.021568628	155.78711	100.984375	83.0	25.0	0.019607844	261.03516	43.703125	74.0	83.0	0.033333335	118.08203	61.140625	58.0	84.0	0.023529414	165.36523	87.53906	50.0	50.0
0.027450982	200.69727	47.695312	123.0	0.027450982	142.86719	99.22656	65.0	59.0	0.019607844	220.16602	53.664062	66.0	109.0	0.033333335	119.66406	88.67969	66.0	92.0	0.03137255	166.41992	74.0625	58.0	58.0
0.03529412	171.42969	71.71875	90.0	0.021568628	151.04102	103.91406	66.0	58.0	0.021568628	230.71289	88.82031	57.0	109.0	0.027450982	137.0664	48.835938	65.0	109.0	0.03137255	165.10156	48.28125	58.0	59.0
0.015686275	214.93555	99.84375	63.0	0.03137255	135.48438	91.02344	49.0	101.0	0.027450982	213.04688	68.3125	49.0	125.0	0.04705883	117.81836	104.5	50.0	117.0	0.03529412	180.39453	53.554688	58.0	83.0
0.033333335	215.19922	47.695312	124.0	0.033333335	174.77148	87.50781	58.0	7.0	0.033333335	262.35352	44.289062	57.0	117.0	0.037254903	112.01758	100.984375	67.0	92.0	0.03137255	172.2207	49.453125	66.0	41.0
0.023529414	189.0957	60.585938	72.0	0.027450982	157.10547	76.96094	66.0	75.0	0.033333335	279.4922	36.671875	58.0	117.0	0.027450982	123.88281	76.96094	58.0	108.0	0.027450982	184.34961	98.08594	58.0	50.0
0.03529412	181.97656	67.03125	65.0	0.03137255	137.33008	83.99219	57.0	84.0	0.027450982	214.36523	53.078125	49.0	125.0	0.029411767	117.02734	72.859375	66.0	109.0	0.021568628	154.02734	56.484375	66.0	67.0
0.027450982	208.60742	71.71875	66.0	0.029411767	148.14062	100.984375	66.0	100.0	0.03137255	217.26562	57.765625	74.0	116.0	0.015686275	110.171875	106.25781	82.0	101.0	0.03529412	181.97656	79.33594	74.0	59.0
0.023529414	195.6875	75.234375	66.0	0.021568628	164.75195	109.1875	58.0	58.0	0.021568628	245.21484	45.460938	49.0	141.0	0.027450982	147.87695	41.21875	41.0	117.0	0.027450982	174.85742	23.085938	49.0	59.0
0.023529414	197.26953	50.625	82.0	0.029411767	177.4082	109.1875	58.0	25.0	0.021568628	247.06055	37.84375	42.0	125.0	0.027450982	126.51953	93.36719	66.0	108.0	0.019607844	187.77734	58.242188	58.0	50.0
0.015686275	226.27344	57.070312	79.0	0.03137255	139.70312	88.09375	66.0	83.0	0.023529414	214.89258	70.07031	58.0	117.0	0.033333335	120.98242	85.75	50.0	117.0	0.029411767	181.71289	37.148438	58.0	74.0
0.021568628	195.42383	36.5625	131.0	0.029411767	136.80273	100.39844	66.0	25.0	0.021568628	207.2461	26.710938	41.0	117.0	0.04705883	123.88281	102.74219	65.0	101.0	0.03529412	176.70312	55.898438	66.0	34.0
0.021568628	197.26953	94.57031	96.0	0.033333335	177.4082	85.16406	58.0	17.0	0.027450982	258.92578	71.828125	57.0	117.0	0.027450982	138.64844	105.671875	58.0	92.0	0.03529412	187.77734	58.828125	58.0	50.0
0.027450982	205.17969	65.859375	90.0	0.029411767	118.08203	89.265625	49.0	50.0	0.03529412	233.08594	139.07031	41.0	133.0	0.03137255	142.33984	77.546875	66.0	84.0	0.03529412	191.9961	76.99219	57.0	34.0
0.019607844	199.3789	64.6875	89.0	0.03529412	142.07617	52.9375	58.0	50.0	0.03529412	230.44922	57.765625	74.0	108.0	0.015686275	103.84375	83.99219	66.0	117.0	0.037254903	202.2793	68.78906	58.0	125.0
0.019607844	211.24414	87.53906	64.0	0.03137255	168.9707	69.92969	66.0	42.0	0.027450982	265.2539	42.53125	49.0	117.0	0.029411767	118.87305	67.0	57.0	101.0	0.019607844	198.58789	51.210938	50.0	25.0
0.023529414	220.73633	91.05469	56.0	0.027450982	171.34375	70.515625	58.0	83.0	0.021568628	280.01953	29.640625	50.0	116.0	0.029411767	159.74219	33.601562	49.0	100.0	0.03529412	189.88672	81.09375	49.0	26.0
0.033333335	205.9707	77.578125	90.0	0.015686275	119.13672	95.125	66.0	42.0	0.03137255	213.04688	45.460938	49.0	117.0	0.021568628	90.66016	103.328125	66.0	75.0	0.021568628	180.6582	41.25	66.0	67.0
0.023529414	203.07031	54.726562	99.0	0.027450982	155.52344	91.609375	41.0	8.0	0.029411767	262.08984	56.007812	66.0	100.0	0.03529412	149.98633	56.453125	58.0	92.0	0.03137255	186.98633	93.984375	58.0	50.0
0.027450982	196.74219	68.78906	48.0	0.03137255	158.42383	88.67969	41.0	58.0	0.021568628	292.4121	23.78125	41.0	142.0	0.021568628	118.3457	68.75781	66.0	100.0	0.029411767	149.54492	69.375	58.0	42.0
0.027450982	190.41406	93.39844	74.0	0.03529412	149.98633	75.203125	66.0	50.0	0.03137255	233.87695	62.453125	41.0	125.0	0.037254903	124.67383	86.33594	49.0	101.0	0.037254903	163.7832	68.78906	49.0	67.0
0.027450982	201.75195	64.10156	88.0	0.043137256	115.708984	104.5	66.0	26.0	0.033333335	207.50977	0.9296875	50.0	108.0	0.03529412	145.76758	68.75781	49.0	141.0	0.03137255	202.2793	81.67969	57.0	51.0
0.027450982	193.05078	47.695312	98.0	0.027450982	131.79297	81.0625	67.0	16.0	0.027450982	235.45898	80.03125	58.0	116.0	0.023529414	83.01367	80.47656	57.0	117.0	0.033333335	186.19531	98.08594	57.0	84.0
0.015686275	214.67188	86.36719	73.0	0.03529412	164.75195	62.3125	57.0	25.0	0.029411767	265.2539	0.9296875	49.0	117.0	0.021568628	137.0664	105.08594	65.0	101.0	0.033333335	204.91602	47.695312	57.0	51.0
0.033333335	173.45312	89.265625	106.0	0.033333335	173.45312	89.265625	129.0	0.0	0.03137255	243.36914	86.47656	58.0	117.0	0.03529412	150.51367	61.726562	57.0	100.0	0.037254903	150.07227	48.28125	57.0	59.0
0.019607844	218.89062	107.46094	73.0	0.029411767	125.46484	103.91406	49.0	109.0	0.029411767	199.59961	70.07031	66.0	133.0	0.033333335	175.03516	70.515625	40.0	134.0	0.03137255	176.70312	50.625	50.0	51.0
0.03137255	195.42383	81.67969	48.0	0.033333335	189.80078	81.0625	49.0	58.0	0.021568628	256.0254	30.8125	50.0	134.0	0.03529412	151.04102	78.13281	50.0	149.0	0.027450982	182.24023	56.484375	67.0	41.0
0.015686275	197.5332	99.25781	65.0	0.019607844	150.51367	74.03125	74.0	66.0	0.027450982	222.53906	42.53125	50.0	125.0	0.03529412	160.79688	74.03125	57.0	125.0	0.023529414	173.53906	88.125	58.0	67.0
0.019607844	231.54688	73.47656	73.0	0.027450982	160.00586	85.16406	57.0	25.0	0.03137255	250.75195	54.25	57.0	126.0	0.027450982	161.06055	69.34375	49.0	117.0	0.03529412	197.5332	68.78906	49.0	59.0
0.023529414	175.64844	78.16406	75.0	0.023529414	131.5293	83.40625	74.0	33.0	0.027450982	216.73828	46.046875	66.0	117.0	0.027450982	101.734375	79.890625	58.0	100.0	0.027450982	177.23047	85.78125	58.0	58.0
0.027450982	212.29883	123.86719	71.0	0.019607844	148.66797	74.03125	66.0	25.0	0.023529414	225.43945	61.28125	66.0	134.0	0.033333335	110.171875	86.921875	58.0	125.0	0.013725491	171.69336	76.40625	58.0	58.0
0.019607844	194.10547	103.94531	72.0	0.027450982	144.71289	88.67969	58.0	76.0	0.027450982	237.56836	75.92969	49.0	141.0	0.033333335	123.0918	89.85156	50.0	117.0	0.03529412	196.74219	69.96094	50.0	58.0
0.019607844	205.70703	60.0	89.0	0.023529414	135.74805	82.234375	67.0	41.0	0.027450982	268.41797	47.804688	49.0	133.0	0.015686275	125.728516	78.13281	50.0	125.0	0.019607844	194.63281	47.695312	66.0	83.0
0.027450982	214.67188	86.36719	81.0	0.03529412	144.71289	82.82031	49.0	92.0	0.03137255	247.06055	57.179688	58.0	125.0	0.03529412	148.66797	79.30469	42.0	108.0	0.03137255	167.21094	43.007812	50.0	83.0
0.027450982	199.90625	31.875	73.0	0.021568628	149.45898	105.08594	66.0	42.0	0.027450982	228.07617	44.875	58.0	108.0	0.037254903	119.66406	112.11719	74.0	83.0	0.027450982	173.01172	98.08594	57.0	43.0
0.013725491	198.32422	81.67969	73.0	0.019607844	161.85156	83.40625	66.0	49.0	0.021568628	237.04102	35.5	42.0	150.0	0.03137255	135.48438	75.203125	41.0	158.0	0.03137255	202.54297	100.42969	40.0	84.0
0.027450982	189.88672	84.609375	74.0	0.03137255	170.28906	79.890625	65.0	75.0	0.027450982	251.80664	89.40625	58.0	109.0	0.037254903	132.84766	71.6875	41.0	117.0	0.019607844	191.20508	75.234375	50.0	75.0
0.03137255	194.10547	79.33594	71.0	0.027450982	120.45508	78.13281	74.0	33.0	0.027450982	222.53906	64.796875	58.0	167.0	0.027450982	107.53516	20.125	33.0	157.0	0.021568628	179.60352	31.875	42.0	84.0
0.019607844	216.2539	58.828125	91.0	0.037254903	114.390625	110.94531	58.0	99.0	0.03137255	245.21484	57.765625	58.0	125.0	0.027450982	89.86914	67.58594	59.0	108.0	0.03137255	168.26562	50.039062	57.0	33.0
0.027450982	208.34375	38.90625	123.0	0.019607844	158.6875	69.92969	66.0	59.0	0.03137255	245.74219	50.734375	58.0	142.0	0.037254903	101.20703	65.828125	58.0	116.0	0.03529412	174.59375	75.234375	57.0	67.0
0.023529414	217.3086	45.9375	81.0	0.03137255	139.70312	58.796875	57.0	133.0	0.03529412	235.72266	70.65625	58.0	142.0	0.037254903	128.36523	64.65625	49.0	125.0	0.03529412	191.20508	74.0625	58.0	92.0
0.023529414	224.42773	98.08594	114.0	0.03137255	131.00195	86.921875	74.0	34.0	0.023529414	228.07617	62.453125	66.0	92.0	0.033333335	104.634766	86.921875	57.0	100.0	0.03529412	180.6582	59.414062	58.0	43.0
0.023529414	202.2793	79.33594	64.0	0.027450982	175.03516	83.40625	57.0	67.0	0.027450982	226.23047	88.234375	57.0	134.0	0.027450982	120.98242	119.14844	50.0	109.0	0.03529412	181.18555	72.890625	57.0	100.0
0.03137255	198.32422	54.726562	57.0	0.027450982	157.10547	85.16406	74.0	99.0	0.027450982	257.08008	63.039062	58.0	126.0	0.033333335	125.46484	86.33594	67.0	116.0	0.03529412	173.01172	80.50781	58.0	66.0
0.029411767	215.99023	76.40625	91.0	0.029411767	121.509766	77.546875	74.0	42.0	0.03137255	227.8125	61.28125	58.0	125.0	0.037254903	123.35547	92.78125	58.0	125.0	0.029411767	208.08008	60.585938	66.0	100.0
0.013725491	192.25977	80.50781	72.0	0.03137255	138.1211	83.40625	58.0	125.0	0.019607844	220.42969	53.664062	73.0	117.0	0.021568628	92.76953	115.63281	58.0	101.0	0.03529412	178.54883	56.484375	74.0	150.0
0.021568628	227.5918	71.71875	99.0	0.029411767	141.02148	108.015625	58.0	66.0	0.019607844	219.11133	37.84375	59.0	116.0	0.033333335	110.171875	68.171875	66.0	108.0	0.03137255	178.54883	50.625	58.0	33.0
0.029411767	219.68164	76.99219	73.0	0.03137255	166.33398	82.82031	57.0	34.0	0.027450982	238.0957	19.09375	58.0	125.0	0.037254903	111.75391	97.46875	58.0	125.0	0.03529412	191.9961	71.13281	58.0	58.0
0.03137255	165.36523	86.953125	65.0	0.019607844	141.02148	93.36719	58.0	50.0	0.021568628	248.3789	29.054688	66.0	108.0	0.027450982	168.17969	68.171875	58.0	92.0	0.029411767	184.61328	35.390625	50.0	58.0
0.03529412	180.6582	62.929688	73.0	0.015686275	118.3457	95.71094	74.0	50.0	0.029411767	203.55469	23.195312	58.0	125.0	0.027450982	109.11719	77.546875	58.0	167.0	0.03529412	179.07617	65.859375	57.0	100.0
0.029411767	218.62695	83.4375	114.0	0.015686275	130.47461	83.99219	58.0	26.0	0.019607844	223.0664	45.460938	41.0	150.0	0.03529412	86.70508	79.890625	58.0	108.0	0.033333335	173.27539	42.421875	50.0	83.0
0.03137255	227.5918	52.382812	98.0	0.03529412	165.01562	68.75781	58.0	66.0	0.023529414	213.04688	74.171875	49.0	125.0	0.027450982	111.75391	80.47656	50.0	117.0	0.027450982	189.88672	89.88281	66.0	83.0
0.027450982	208.60742	72.890625	81.0	0.027450982	134.16602	64.07031	41.0	33.0	0.03137255	245.74219	54.25	49.0	133.0	0.037254903	118.08203	96.296875	57.0	93.0	0.027450982	168.00195	78.16406	58.0	42.0
0.027450982	193.05078	85.78125	98.0	0.03529412	154.73242	67.58594	75.0	66.0	0.019607844	239.9414	63.625	49.0	125.0	0.033333335	122.828125	76.375	66.0	117.0	0.033333335	204.65234	81.67969	58.0	83.0
0.027450982	209.92578	128.55469	64.0	0.021568628	131.79297	97.46875	49.0	42.0	0.021568628	266.57227	0.34375	41.0	142.0	0.033333335	174.77148	45.320312	50.0	100.0	0.03529412	192.25977	65.859375	42.0	25.0
0.019607844	200.69727	112.14844	74.0	0.03137255	165.2793	85.75	74.0	8.0	0.027450982	230.44922	49.5625	58.0	108.0	0.03529412	139.9668	83.99219	57.0	92.0	0.029411767	196.21484	33.046875	49.0	59.0
0.015686275	164.31055	113.90625	88.0	0.015686275	139.17578	90.4375	66.0	83.0	0.029411767	199.86328	63.039062	65.0	109.0	0.04705883	107.53516	100.39844	59.0	92.0	0.019607844	161.9375	44.179688	58.0	66.0
0.021568628	182.24023	28.359375	91.0	0.019607844	132.05664	91.02344	74.0	58.0	0.021568628	256.8164	29.054688	49.0	117.0	0.033333335	119.40039	58.210938	57.0	84.0	0.03137255	193.05078	86.36719	57.0	43.0
0.027450982	195.42383	98.671875	64.0	0.015686275	117.291016	101.57031	58.0	16.0	0.027450982	225.70312	40.1875	49.0	117.0	0.037254903	119.66406	83.99219	58.0	100.0	0.03529412	197.00586	58.242188	57.0	34.0
0.027450982	206.23438	61.757812	81.0	0.03137255	125.20117	117.390625	66.0	42.0	0.03137255	248.3789	47.804688	57.0	117.0	0.033333335	125.99219	93.953125	50.0	92.0	0.019607844	192.25977	70.546875	66.0	41.0
0.015686275	222.8457	59.414062	99.0	0.021568628	151.30469	102.15625	66.0	41.0	0.027450982	234.93164	46.046875	50.0	125.0	0.019607844	107.79883	69.92969	57.0	92.0	0.03137255	170.90234	84.609375	74.0	92.0
0.027450982	231.01953	81.67969	65.0	0.027450982	139.70312	88.09375	58.0	49.0	0.027450982	282.39258	44.875	41.0	125.0	0.029411767	191.11914	60.554688	49.0	100.0	0.029411767	217.04492	40.078125	57.0	35.0
0.021568628	195.6875	40.078125	73.0	0.03529412	151.83203	67.0	49.0	58.0	0.027450982	239.15039	13.8203125	50.0	125.0	0.019607844	140.49414	70.515625	58.0	100.0	0.021568628	204.91602	37.734375	66.0	50.0
0.029411767	218.62695	82.85156	66.0	0.019607844	169.49805	81.64844	57.0	33.0	0.029411767	271.8457	24.953125	58.0	109.0	0.03137255	81.16797	117.97656	50.0	91.0	0.03529412	179.60352	66.44531	58.0	33.0
0.027450982	234.71094	98.671875	65.0	0.037254903	184.52734	52.9375	50.0	58.0	0.021568628	235.45898	27.296875	57.0	125.0	0.033333335	120.19141	51.179688	65.0	101.0	0.03137255	200.4336	52.96875	58.0	59.0
0.023529414	202.2793	53.554688	82.0	0.03137255	200.08398	79.30469	74.0	17.0	0.03137255	309.81445	53.664062	41.0	108.0	0.033333335	172.13477	61.726562	50.0	100.0	0.013725491	194.89648	79.33594	66.0	33.0
0.027450982	197.5332	92.22656	123.0	0.019607844	134.69336	91.02344	65.0	67.0	0.021568628	223.85742	100.53906	66.0	117.0	0.023529414	94.87891	89.85156	41.0	92.0	0.03529412	194.36914	61.171875	58.0	133.0
0.027450982	199.64258	65.27344	81.0	0.03529412	146.29492	79.30469	48.0	17.0	0.015686275	196.43555	40.1875	49.0	126.0	0.033333335	124.67383	83.99219	49.0	117.0	0.03529412	188.04102	61.171875	58.0	50.0
0.019607844	194.63281	61.171875	72.0	0.027450982	147.34961	111.53125	49.0	25.0	0.029411767	255.23438	39.015625	50.0	117.0	0.03529412	154.46875	71.10156	57.0	92.0	0.03529412	174.33008	82.85156	58.0	50.0
0.015686275	214.67188	86.36719	73.0	0.027450982	160.79688	115.63281	49.0	17.0	0.03137255	247.58789	45.460938	32.0	125.0	0.033333335	166.07031	39.460938	40.0	101.0	0.029411767	176.43945	32.460938	41.0	51.0
0.015686275	212.03516	107.46094	56.0	0.03137255	166.86133	62.898438	57.0	9.0	0.029411767	264.4629	81.78906	57.0	134.0	0.027450982	125.728516	75.203125	50.0	92.0	0.027450982	176.70312	50.625	58.0	33.0
0.021568628	199.3789	105.703125	89.0	0.027450982	134.42969	89.85156	33.0	33.0	0.03137255	217.79297	55.421875	32.0	142.0	0.03137255	90.66016	126.765625	74.0	92.0	0.019607844	180.6582	18.398438	74.0	59.0
0.023529414	170.11133	92.8125	81.0	0.021568628	123.88281	107.42969	57.0	9.0	0.021568628	229.39453	31.398438	48.0	601.0	0.019607844	129.15625	79.30469	50.0	157.0	0.03137255	201.75195	80.50781	65.0	67.0
0.03137255	202.01562	92.8125	82.0	0.03529412	153.41406	76.375	66.0	50.0	0.027450982	230.44922	133.79688	33.0	141.0	0.03529412	164.75195	64.07031	66.0	100.0	0.027450982	189.35938	61.171875	58.0	75.0
0.021568628	191.9961	105.11719	89.0	0.03529412	152.35938	71.6875	58.0	66.0	0.015686275	267.36328	80.03125	58.0	125.0	0.015686275	131.00195	85.75	50.0	116.0	0.027450982	184.87695	44.179688	50.0	58.0
0.027450982	184.08594	79.921875	107.0	0.03529412	140.49414	64.07031	75.0	41.0	0.015686275	268.41797	25.539062	41.0	133.0	0.033333335	174.50781	75.78906	50.0	108.0	0.03529412	177.75781	56.484375	50.0	50.0
0.03137255	207.28906	90.46875	81.0	0.027450982	153.15039	92.19531	83.0	25.0	0.03137255	257.8711	42.53125	58.0	100.0	0.021568628	92.76953	105.08594	74.0	75.0	0.03529412	187.77734	79.33594	50.0	25.0
0.023529414	205.70703	81.67969	99.0	0.027450982	147.87695	86.33594	66.0	41.0	0.021568628	254.17969	26.710938	41.0	142.0	0.03529412	144.71289	61.140625	49.0	125.0	0.03137255	175.38477	65.859375	58.0	50.0
0.021568628	220.20898	78.75	97.0	0.019607844	144.71289	87.50781	66.0	33.0	0.023529414	233.34961	71.24219	50.0	134.0	0.03529412	145.5039	68.171875	40.0	133.0	0.03529412	178.54883	69.96094	58.0	59.0
0.029411767	218.09961	51.796875	91.0	0.027450982	159.47852	80.47656	50.0	66.0	0.021568628	254.17969	34.328125	49.0	150.0	0.03137255	163.69727	69.92969	33.0	134.0	0.03529412	177.23047	85.19531	50.0	83.0
0.027450982	195.42383	84.609375	82.0	0.021568628	160.5332	108.015625	49.0	50.0	0.021568628	217.79297	102.296875	66.0	142.0	0.027450982	166.07031	102.15625	41.0	125.0	0.03137255	192.25977	91.640625	58.0	108.0
0.027450982	189.62305	92.22656	97.0	0.03529412	145.5039	75.78906	58.0	66.0	0.03137255	245.21484	57.179688	41.0	142.0	0.027450982	138.1211	88.09375	42.0	141.0	0.03529412	184.61328	64.10156	58.0	91.0
0.027450982	249.21289	55.898438	107.0	0.027450982	143.13086	92.19531	75.0	66.0	0.023529414	243.10547	10.3046875	50.0	141.0	0.03137255	147.34961	82.234375	41.0	158.0	0.023529414	200.96094	84.02344	74.0	75.0
0.027450982	203.33398	75.234375	66.0	0.027450982	157.10547	110.359375	58.0	16.0	0.019607844	217.5293	40.773438	50.0	133.0	0.021568628	114.12695	86.33594	58.0	124.0	0.03529412	198.85156	74.0625	57.0	84.0
0.013725491	199.3789	81.09375	80.0	0.03529412	152.0957	70.515625	58.0	67.0	0.025490198	277.11914	26.710938	58.0	133.0	0.03529412	151.30469	78.71875	58.0	100.0	0.019607844	171.95703	73.47656	58.0	33.0
0.027450982	186.98633	50.039062	91.0	0.03137255	172.13477	85.75	66.0	49.0	0.021568628	236.51367	25.539062	49.0	125.0	0.03137255	167.38867	85.75	49.0	126.0	0.011764707	173.53906	66.44531	74.0	75.0
0.027450982	222.8457	63.515625	65.0	0.021568628	148.14062	90.4375	58.0	32.0	0.027450982	215.15625	62.453125	49.0	125.0	0.029411767	144.97656	100.984375	57.0	109.0	0.027450982	204.65234	56.484375	59.0	59.0
0.019607844	193.57812	88.71094	73.0	0.029411767	122.828125	100.39844	65.0	67.0	0.027450982	217.5293	71.24219	74.0	109.0	0.015686275	107.79883	105.671875	58.0	117.0	0.015686275	161.9375	98.08594	67.0	83.0
0.027450982	197.26953	98.671875	57.0	0.023529414	137.0664	60.554688	41.0	58.0	0.015686275	208.82812	33.15625	49.0	150.0	0.027450982	81.16797	115.63281	58.0	101.0	0.03137255	170.11133	53.554688	58.0	75.0
0.027450982	237.34766	22.5	74.0	0.027450982	147.34961	90.4375	41.0	58.0	0.015686275	212.25586	82.375	57.0	133.0	0.03529412	141.28516	77.546875	49.0	101.0	0.027450982	210.7168	35.390625	42.0	34.0
0.027450982	205.17969	67.61719	73.0	0.03529412	80.37695	86.33594	66.0	42.0	0.021568628	248.3789	30.8125	50.0	116.0	0.03529412	152.62305	56.453125	49.0	142.0	0.03529412	194.63281	62.929688	66.0	58.0
0.021568628	203.59766	99.84375	64.0	0.03529412	131.00195	64.65625	58.0	59.0	0.03137255	254.17969	88.82031	75.0	108.0	0.027450982	108.32617	85.16406	66.0	100.0	0.03529412	193.31445	57.65625	66.0	75.0
0.023529414	203.59766	74.64844	83.0	0.019607844	139.9668	76.375	58.0	25.0	0.03529412	230.71289	133.79688	41.0	150.0	0.019607844	124.67383	66.41406	66.0	116.0	0.021568628	184.61328	28.359375	99.0	59.0
0.027450982	207.55273	79.33594	98.0	0.037254903	181.36328	96.88281	50.0	83.0	0.027450982	252.07031	68.3125	41.0	150.0	0.027450982	141.02148	119.14844	50.0	116.0	0.027450982	160.88281	77.578125	59.0	66.0
0.027450982	186.19531	95.74219	74.0	0.027450982	141.02148	97.46875	66.0	33.0	0.03529412	242.05078	71.24219	58.0	125.0	0.027450982	119.40039	61.140625	50.0	108.0	0.03137255	187.77734	92.8125	57.0	75.0
0.027450982	208.08008	68.78906	82.0	0.015686275	118.08203	83.99219	82.0	25.0	0.03137255	237.56836	70.07031	66.0	92.0	0.03529412	139.9668	58.210938	41.0	100.0	0.03529412	183.29492	57.070312	41.0	67.0
0.023529414	223.90039	86.953125	82.0	0.013725491	142.86719	86.33594	57.0	17.0	0.023529414	258.92578	16.75	41.0	142.0	0.033333335	120.98242	63.484375	50.0	108.0	0.027450982	205.9707	60.585938	49.0	42.0
0.027450982	202.2793	69.96094	98.0	0.027450982	115.97266	96.88281	74.0	50.0	0.027450982	229.39453	71.828125	49.0	125.0	0.037254903	115.708984	90.4375	49.0	101.0	0.015686275	164.57422	87.53906	50.0	84.0
0.027450982	180.92188	23.671875	89.0	0.019607844	131.00195	78.71875	58.0	33.0	0.027450982	252.07031	22.609375	41.0	150.0	0.03137255	132.84766	71.6875	50.0	108.0	0.027450982	171.95703	71.71875	42.0	33.0
0.023529414	198.85156	51.796875	99.0	0.027450982	155.78711	74.61719	49.0	41.0	0.027450982	247.85156	13.8203125	50.0	133.0	0.027450982	96.46094	100.39844	51.0	116.0	0.03137255	192.25977	81.67969	41.0	66.0
0.027450982	203.86133	69.96094	82.0	0.021568628	124.67383	104.5	83.0	74.0	0.021568628	213.83789	31.984375	50.0	116.0	0.03529412	139.43945	75.78906	49.0	133.0	0.033333335	202.2793	44.765625	49.0	109.0
0.03137255	185.66797	71.71875	65.0	0.023529414	128.6289	76.375	66.0	66.0	0.027450982	223.0664	50.734375	58.0	142.0	0.03137255	99.88867	82.82031	66.0	100.0	0.03529412	181.71289	72.890625	41.0	67.0
0.021568628	188.04102	108.63281	72.0	0.023529414	123.0918	95.71094	66.0	67.0	0.019607844	234.93164	78.859375	41.0	142.0	0.023529414	136.01172	81.0625	49.0	150.0	0.027450982	183.5586	88.125	66.0	75.0
0.027450982	189.35938	101.015625	57.0	0.027450982	151.04102	97.46875	57.0	58.0	0.03529412	237.83203	68.3125	49.0	142.0	0.015686275	111.75391	117.390625	58.0	92.0	0.03529412	184.08594	67.61719	57.0	67.0
0.029411767	216.2539	75.82031	81.0	0.033333335	173.7168	82.234375	50.0	8.0	0.027450982	214.10156	29.640625	58.0	133.0	0.04705883	112.01758	104.5	41.0	125.0	0.03529412	189.88672	68.203125	57.0	75.0
0.019607844	229.17383	67.03125	73.0	0.033333335	178.19922	98.640625	66.0	50.0	0.025490198	268.68164	23.78125	49.0	117.0	0.037254903	170.02539	56.453125	49.0	125.0	0.033333335	222.8457	59.414062	42.0	34.0
0.019607844	194.10547	88.71094	79.0	0.019607844	144.18555	61.726562	50.0	49.0	0.015686275	270.52734	50.734375	32.0	150.0	0.027450982	144.97656	94.53906	57.0	126.0	0.013725491	171.69336	77.578125	50.0	59.0
0.027450982	190.67773	61.171875	91.0	0.03137255	159.47852	85.75	49.0	58.0	0.013725491	239.15039	55.421875	25.0	150.0	0.037254903	166.33398	64.07031	42.0	166.0	0.027450982	195.6875	88.71094	59.0	99.0

Table A.1: Table containing baseline data set, containing 161 transactions in feature dimension 24 (area pressure features are omitted. Each row is a completed transactions. Rows are ordered chronologically on completion date-time.