5-8-2018

Behavioural Based Biometrics Using Keystroke Dynamics for User Authentication

Emamuzo Cletus Ogemuno
University of Windsor

Behavioural Based Biometrics Using Keystroke Dynamics for User Authentication

By

Emamuzo Cletus Ogemuno

A Thesis
Submitted to the Faculty of Graduate Studies
through the School of Computer Science
in Partial Fulfillment of the Requirements for
the Degree of Master of Science
at the University of Windsor

Windsor, Ontario, Canada

2018

Behavioural Based Biometrics Using Keystroke Dynamics for User Authentication

by

Emamuzo Cletus Ogemuno

APPROVED BY:

H. Wu
Department of Electrical and Computer Engineering

S. Saad
School of Computer Science

L. Rueda, Advisor
School of Computer Science

May 1, 2018

DECLARATION OF ORIGINALITY

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office and that this thesis has not been submitted for a higher degree to any other University or Institution.

ABSTRACT

Security of data in recent times has become paramount, which has led to the development of many security systems. Among such systems is keystroke dynamics. Keystroke dynamics has become an active area of research in recent times. This is due, in part, to the increased importance of cyber-security, computer or network access control. Also known as typing dynamics, keystroke refers to a method that identifies users/individuals based on the manner of their typing pattern or rhythm on the keyboard, which could either mean a user is verified (identified) or authenticated. User identification is a critical factor before authentication. Now with a person already identified, the next step is to authenticate. Even if the user types in a correct password, that does not mean that the user is whom they say they are. The focus of this thesis is on the dynamic approach of keystrokes. In this work, we propose a method which improves our classification algorithm. We introduce a method that uses the minimum redundancy maximum relevance feature selection method which selects the best features based on their relevance and redundancy. We have also used several classifiers that include support vector machine, k-nearest neighbour, Naïve Bayes and grid search for optimizing the support vector machine. The results not only show the efficiency of our method but also show that the proposed method can be applied to other datasets to produce optimal results.

DEDICATION

This thesis is dedicated to my parents Mr. & Mrs. G.A Ogemuno, and to my siblings for their advice, support and tremendous sacrifice through the years.

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to God Almighty who has made it possible to complete my program successfully. I also want to express my profound gratitude to my supervisor Dr. Luis Rueda for his support, advice, encouragement and guidance through the course of my study and help in transitioning as an international student.

I also want to express my sincere thanks to Prof. Vinnie Monaco, for making available the dataset used in the experiments carried out in this thesis.

Special thanks to my external reader Dr. Huapeng Wu and internal reader Dr. Sherif Saad for their valuable input and suggestions to this research.

I want to express my gratitude to the faculty staff of the School of Computer Science for their support and help in my transition as an international student.

I also want to express my utmost gratitude to my pastor Grace Ugbeye, James Ugwogbo, Mr&Mrs Imoyera for continuous prayers and support. Also, to my brother Emmanuel Ogemuno, thanks for the support, prayers and encouragement. Finally, to my friends Castro, Nonso, Victor, Naveen, Roophesh, Sowndarya, and others too numerous to mention; thanks for the immerse support and encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

1.1 What Is Biometrics?

The use of biometric systems dates back to the early 1800s, when Alphonse Bertillon a Perisian anthropologist and police desk clerk, developed a method known as the Bertillonage used for criminal identification [1]. This technique employed anthropometry (the scientific study of the measurements and proportions of the human body), a system that measures the body for classification and comparison purposes. Requiring a separate and precise analysis of the bony parts of the human anatomy, also recording the shapes of the body as it relates to movements and differential markings (scars, tattoos, birthmarks) for identification. As a result, this system required time and effort for measurement and, overall it was not an accurate one. Due to this, it was quickly replaced by the fingerprinting method in the late 1870's as a means of identification and first looked at as a form of criminal identification by Dr. Henry Faulds. In recent times different biometric systems have been developed and, current implementations are found in mobile technologies such as in the high-end Samsung phones, which has a blend of facial recognition and fingerprinting, while some also incorporate voice recognition. Biometric systems are an automated method of verifying or recognizing the identity of a living person, usually based on the physiological or behavioural characteristics of an individual [1]. Biometrics in the context of this thesis

is the automated method of authentication using machines [6]. Biometric systems are categorized into 1). Physiological and 2). Behavourial Biometrics.

As it regards operational environment, based on the implementation of our system, it can be categorized as follows:

- Overt or Covert: Overt means the user is aware that a biometric identifier is measured, while covert implies the user is unaware of any such biometric identifier. Biometric identifier as it relates to our study is keystroke dynamics, which falls under the overt category.
- Habituated or Non-Habituated: Depending on the frequency of use it can be habituated, if frequently used or non-habituated otherwise. In this thesis, keystroke dynamics is intended for the habituated user.
- Open or Closed: A system is open when data collection, compression and format standards are required. A closed system, on the other hand, operates well entirely on proprietary formats.

1.1.1. Types of Biometric

1. Physiological Biometrics: Physiological biometrics (biological or chemical) is based on the physical attributes of an individual. Data are often derived from measurements of body parts of the human body [3,4]. Some of the human senses notably used for user identification, verification, or authentication, include touch (fingerprinting), sight (retina scan, facial recognition), taste (DNA from saliva), and smell (odour or scent).

2. Behavourial Biometrics: Behavourial biometrics (a reflection of an individual's psychology) is based on the behaviour of individuals, that is the manner in which they conduct themselves. It is also a field related to how measurements are uniquely done to identify individuals based on measurable patterns in human activities [3,4]. Examples include handwritten signatures, voice pattern, mouse movement dynamics, and keystroke dynamics. Figure 1.1 shows some examples of both physiological and behavioural biometrics.

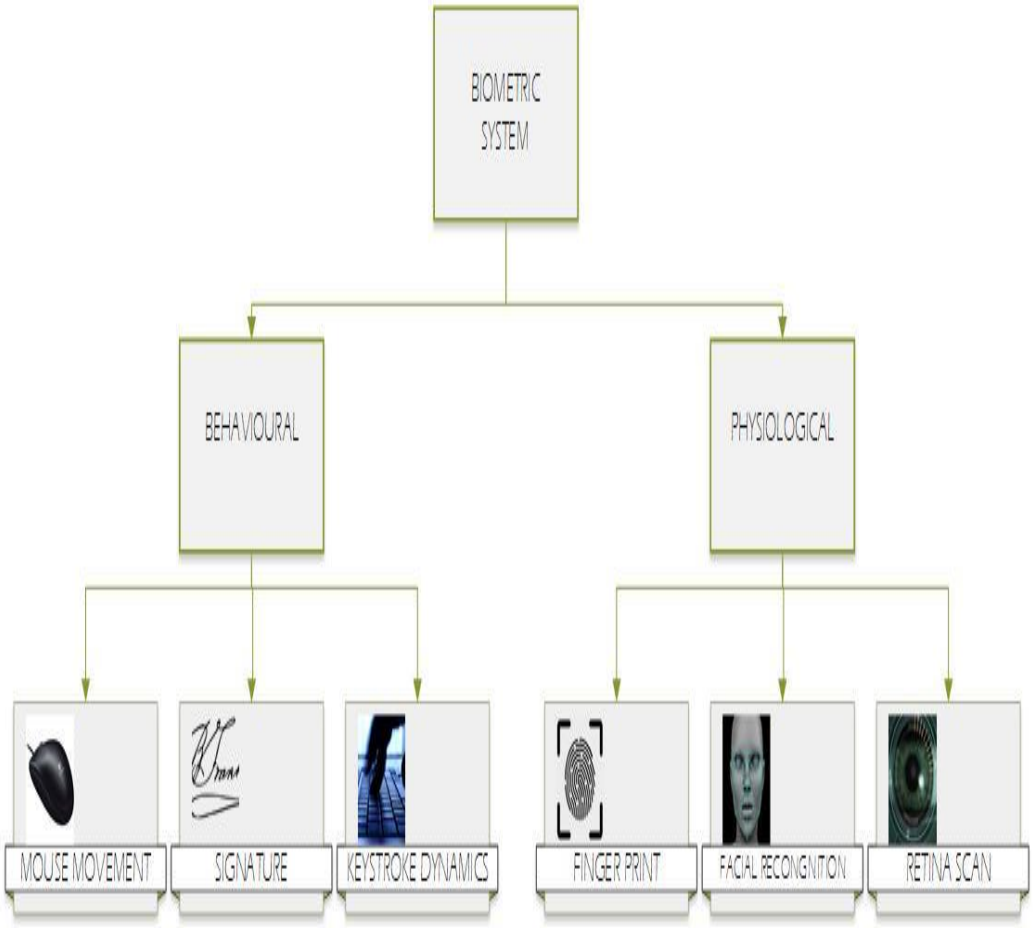

Figure 1.1: Different examples of biometric systems.

1.1.2. What Is Keystroke Dynamics?

Keystroke dynamics also are known as typing dynamics. It refers to an automated method that identifies or verifies a user or an individual based on the typing patterns or typing dynamics with the aid of a keyboard [5, 6]. It is also a process of analyzing the way an individual types, by monitoring the keyboard inputs thousands of times per second in an attempt to either identify or verify a user based on recurrent typing rhythm or patterns. It is also the timing information describing each key when it is pressed and released as a user types. As a means of user authentication, it authenticates users uniquely by their typing patterns. Figure 1.2 illustrates the measurement of a user's keystroke, with the measurement taken for every key pressed and released.

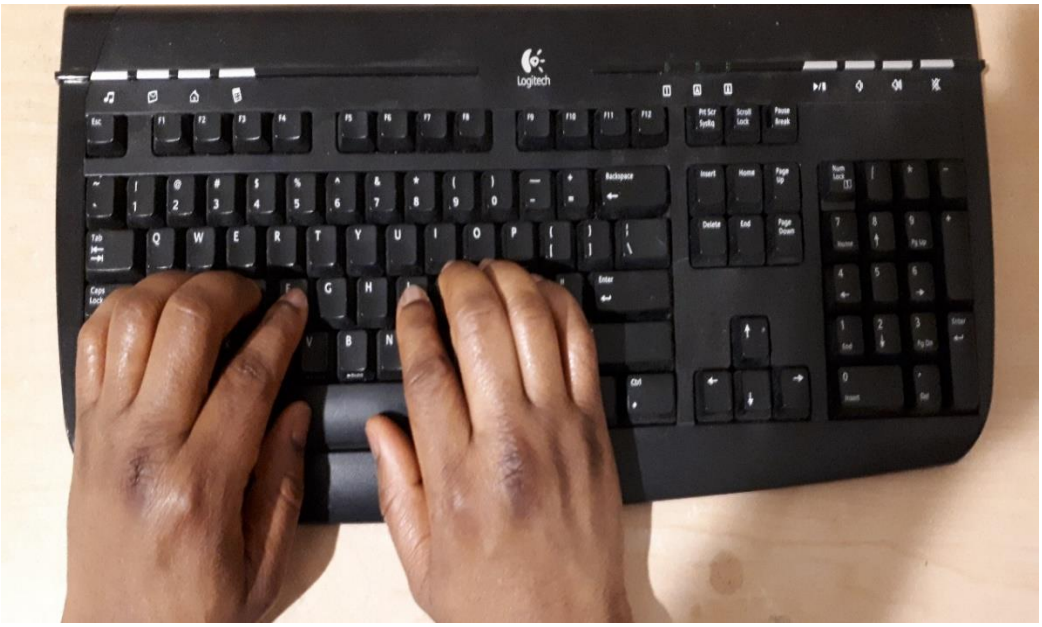

Figure 1.2: Illustration of a user's keystrokes.

1.1.3. Measurement of Keystroke Biometrics

In keystroke biometrics measurement, the two-timing information used is the flight and dwell time. Both are called timing information which describes when a key is pressed and released as a user types on the keyboard by measuring each key action in milliseconds [5, 7, 9]. Other timing information used include di-graph and tri-graph, as shown in Figure 1.3.

- Flight Time: It is the timing duration between the release of a key, and press of the next key.
- Dwell Time: It is the timing duration when a key is pressed down.
- Di-graph: Refers to the timing information between two keystrokes.
- Tri-graph: Refers to the keystroke timing information between three successive keys.

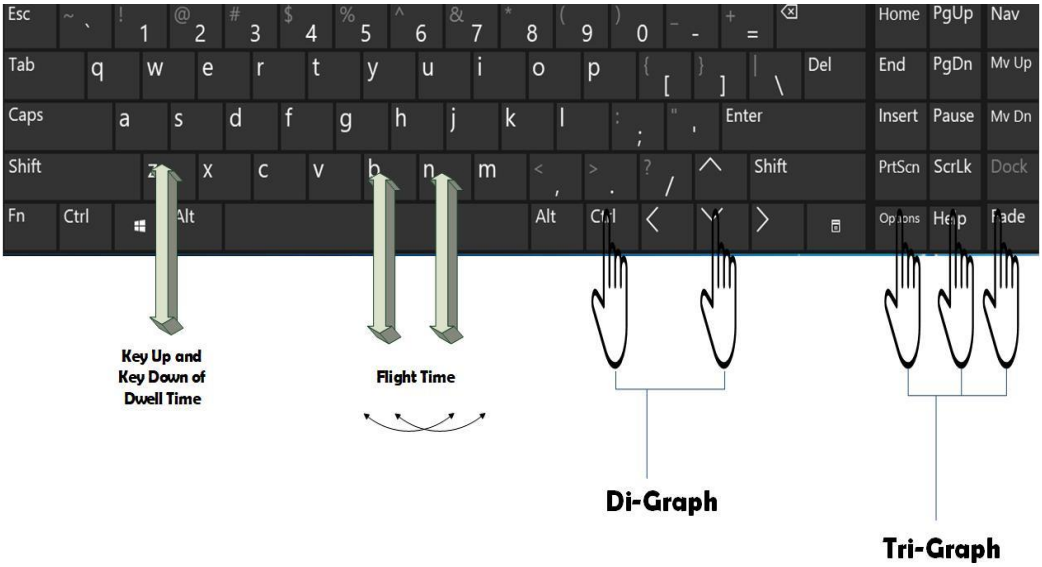

Figure 1.3: Description of keystroke timing features.

1.1.4. Biometric Templates for Identification

Biometric templates are digital references of distinct user characteristics extracted from a biometric sample during a user's enrollment, used during user verification/authentication. In user verification/authentication the biometric system performs one to one comparison with the captured user samples to ascertain the correct user. In user identification, the biometric system performs one to one comparison with the obtained user samples to determine the right user. In keystroke dynamics, three templates used for user identification often during enrollment are:

- The speed of typing of a user while using a keyboard.
- Typing pattern is a reliable sample of traits, acts, tendencies, or other observable characteristics of the user.
- Typing rhythm describes the habitual way a user types, i.e., the manner in which a user types on a keyboard.

1.1.5. Keystroke Dynamics Approaches

Keystroke dynamics has two approaches for user authentication, which can be either static or dynamic.

- Static or Fixed Text Verification: Static verification analyzes keystroke timing information only at specific times [2,3]. An example is a user's first interaction with the system at login, based solely on the password typing rhythm. The minimum length from various studies [3, 7, 8, 9, 11] shows that eight-character long text is the minimum requirement, which also is the minimum character length required for a password enabled system.

- Dynamic or Free-Text Verification: Dynamic verification or free text analyzes keystroke timing information of a user first at login and continuously or periodically during the user's session [2, 3]. The aim is to ensure a user is still whom they say they are after the initial login. The approach of this thesis follows the dynamic verification approach. The minimum length of characters used in dynamic authentication from [22] shows that the minimum number of characters for dynamic authentication is between 28 to 54 characters. Also, the research carried out by the authors of [19] illustrates the use of 380 long character texts for dynamic authentication.

1.1.6. Advantages and Disadvantages of Keystroke Dynamics

Advantages of keystroke dynamics:

- Non-intrusive and transparent, hence it is overt because the user is aware of what is happening during a session.
- Hardware required is the keyboard connected to a computer, making it inexpensive.
- With dynamic verification approach, the system continuously verifies a user during the user's session.
- When integrated into a security system it provides an extra layer of security.

Disadvantages of keystroke dynamics:

- Change of keyboard leads to a dynamic shift in typing patterns.
- Fatigue, injury to the user leads to a change in typing patterns.

A way of addressing these disadvantages is to collect timing information of the user round the clock when fatigued or when there is an injury to the user, and in the event when a keyboard is replaced. In other words, the advantages significantly outweigh the disadvantages of keystrokes.

1.2 Problem Statement

This thesis aims to provide a method that analyzes the typing patterns of users using known classification algorithms, which accurately authenticates them continuously during a session. While selecting the best features relevant for accurate classification of the users, it results in better performance with the given dataset, also bearing in mind that these features may vary from one keystroke dataset to another, but with the categories well remaining the same. This thesis extends the work previously done by the authors of [4]. Given the Four HCI tasks dataset with the obtained timing information features; we aim to:

- Identify users/individuals as either imposters or genuine on analysis of the timing features obtained.
- Determine the relevant features that improve the classification algorithms.
- Provide a method which reduces the number of features without affecting the performance of the system.

1.3 Motivation

The motivations behind this thesis are:

- It is economical and easily integrated into an existing computer security systems with minimal alteration and user intervention.
- It is non-intrusive and transparent.
- With dynamic or continuous authentication, it ensures there is round the clock authentication of every user.
- On account of data being the heart of every organization, having a secured system which guarantees the proper authentication of users, also ensures the data is available to an authorized user, preserves its integrity, and protects the confidentiality of the data.
- It provides an efficient way of reducing current security concerns.

1.4 Contribution of This Thesis

This thesis extends the work reported in [4, 5]. The main contributions of this study are:

- Provide an authentication system that selects relevant features for an improved user authentication.
- Provide a method that selects relevant features using Minimum Redundancy Maximum Relevance (mRMR) which improves the performance.
- Introduce authentication classification algorithms that improve the performance of the classifier evaluators.

1.5 Conclusion

In this chapter, we introduced the term biometrics, discussed some essential terminologies of keystroke dynamics, the two types of biometric systems, advantages of keystroke dynamics, the problem statement, motivation, and contributions of this thesis.

This thesis is organized as follows:

- In Chapter 2, we introduce the literature review.
- In Chapter 3, we introduce the datasets and methods used for this thesis.
- In Chapter 4, we provide details of the results obtained from the experiments we performed.
- Chapter 5 includes the conclusion of this thesis.

CHAPTER 2

Literature Review

Over the years various researchers on keystroke dynamics focused on either of the two approaches, with 83% focused on static authentication and about 10% on dynamic authentication [3], and with lots of strides achieved in the study of keystroke dynamics. This chapter provides a review of previous studies in this area, methods and performances accomplished. It is divided two sections, for static and dynamic verifications.

2.1 Review of Static Verification Approaches

Bergadano et al. compared other access control systems based on biometric features (keystroke analysis) and concluded that it has not led in providing an acceptable degree of accuracy [6]. The reason being the intrinsic variability of typing dynamics when compared with other biometric characteristics that are stable which include face or fingerprint patterns. The authors seek to address measures of keystroke dynamics that limit the instability of using this biometric feature. In their experiments, they implemented several techniques, one of which is the degree of disorder of an array. They assumed an array V contains given elements N, and V is taken against its disorder counterpart V, then computed as a sum of the distances between each of them. To obtain an ordered and an unordered array, by normalizing the degree of disorder dividing it by the maximum disorder of the array of N elements, obtaining values which lie between 0 (implies the ordered array) and 1 (unordered array). The distance between two typing samples was

another method. The aim was to find the similarity and differences between a given sample of the same text. The two primary measures here were, the duration of a key (how long a key is held down) and latency between two consecutively typed keys (the time between the release of the first key and depression of the next). In the experiment, they took into consideration the dwell time between the first and the third key of a trigraph (three keys typed one after the other). For example, a user is asked to type "America.". The outcome of the trigraph can be broken down into five possible sets of trigraphs, and each trigraph sorted with respect to its duration calculated in milliseconds. Comparison between different samples is done to obtain the degree of disorder between them. This study describes a biometric measure of the typing characteristics of individuals. This measure was tested on a set of users and was found to perform exceptionally well to authenticate legitimate users and rejects impostors. It obtained on the average a 4% False Acceptance Rate (FAR) and Impostor Pass Rate (IPR) of less than 0.01% (less than one successful attack out of 10,000 attempts). Compared to other methods or approaches, this technique allows typing errors, uses the same sampling text for all the individuals and requires a constrained number of samples which forms the model for typing used by the users. It was also found to work well without problems of overfitting and even when there is a remote connection.

Araujo et al. employed static keystroke dynamics for user authentication in their research [7]. The key down, key up and the key ASCII codes, were the inputs captured while the user typed a string. The key code, two keystroke latencies and key duration were the four features analyzed in seven

experiments carried out by combining these features. The result of this experiment was evaluated for three categories of users: the legitimate, the impostor, and the observer impostor. The False Rejection Rate (FRR) of 1.45% and FAR of 1.89% was obtained. This technique was used to improve the regular login password authentication process when all users know the password. In this study, a target string with at least ten characters was used. In the enrollment process, ten samples were collected from each user. The features analyzed were the key code, two keystroke latencies and the duration key. Whenever a user attempts to access a system, he specifies an account and types the target string, while typing the user's keystroke data is captured, and a sample is created containing the features calculated using that data. When it is a new account (enrollment), the training set is collected, and a template is created providing the pattern found, a sample will only be stored when it matches the key code feature. If it is an existing account (authentication), then a classifier will analyze the accounts template to decide if the sample belongs to the owner's account and if it passes this phase the user is permitted to access the system. Otherwise, it allows for a second try if it fails to authentiacte the user, the user is considered an impostor. The technique used does not deal with typographic errors, a session of verification starts when there is not one. The following are the methods being employed:

- Timing accuracy: In this work, the time stamp counter function was utilized to record the count of clock cycles. It keeps an accurate count of every cycle that happens in the processor. The accuracy received must be good with the range estimations of the gathered examples.

Since 98% of the accumulated sample values are somewhere around 10 and 900ms, a 1-ms accuracy is utilized.

- Keystroke data: It was assumed to contain m characters resulting in n keystrokes. Some characters needed more than one keystroke. Each keystroke comprised of the key downtime and the key up and the key code (ASCII code).
- Features: Calculated using the keystroke data. Four features analyzed are the key code feature and three timing features, down-down (DD), down-up (DU) and up-down (UD).
- Template: It contains the key code, the mean and standard deviation calculated for each i^{th} element for the feature (DD, DU, UD).
- Classifier: The classifier analyzes the sample of the account, and a comparison is made to determine if there is a match; if different the sample is considered false.
- Adaptation Mechanism: It consists of creating a new updated template, including the new sample and discarding the oldest one. This mechanism is performed after successful authentication with a sample if the majority elements are I of its time features.

This study presents a methodology through typing biometrics features, which improves the standard login password authentication. Experiments carried out using a statistical classifier based on distance and a combination of four features times (key code, DD, UD and DU), obtaining a FRR of 1.45% and 1.89% FAR. These rates were shown to be competitive when compared to previous studies, using one target string and ten samples in enrollment. The utilization of four features to verify users is novel since

earlier reviews used only a couple of features. This study demonstrates the influence of some liberal perspectives, which were tried and observed and proved that they have an applicable influence in the final results. These viewpoints are the similarity of the target string, the two-trial authentication, the adaptation mechanism, the timing accuracy, and the number of samples in enrollment.

Hu et al. followed a *k*-nearest neighbour approach for authentication, using as a distance measure for classification and clustering [8]. An input needs only to be verified against a limited pool of users which reduces the verification load significantly. The degree of disorder was also used in this case, the trigraph, two arrays of trigraphs sorted according to the duration of trigraphs with the distance between these two arrays counted and summed. The verification process in this study is within a cluster and not through the entire set of data. Also, for a given user only its representative profile is needed during the authentication process. In the experiment, 27 typing data were provided by impostors along with samples obtained from profiles from the database, with a FAR of 0.045%. This algorithm, gave an improved increase in the authentication time from 16s to 19.3s, making it entirely scalable.

Kevin et al. experimented and obtained a dataset employed in their research [9]. Typing data was collect from 51 users each typing 400 repetitions of a particular password. Each user was allowed to type the same password, similar in both length and structure. The idea of using the same password for the data-collection was that in letting users choose their password, it makes it easier to discriminate them because it stands out and

biases the result of the experiment designed to evaluate the performance of an arbitrary password. In selecting the password ".tie5Roanl", a password generator was employed. The enter key was also considered as part of the password. In this study, 14 detectors were evaluated using the password timing data. Each trained and tested using the procedure, and the scores obtained from the anomaly converted to standard measures of error. The ROC curve (visualization of the accuracy of the detector) was used to determine the performances of the detectors. The performance results of the detectors on the Equal Error Rate (EER) indicates that Manhattan (scaled) detector has an EER of 0.096%, nearest neighbor (Mahalanobis) 0.100%, Outlier Count (z-score) 0.102% as the top performers, showing a significant improvement when compared with previous studies. Beforehand, it was unrealistic to think about the execution of various anomaly detectors crosswise over reviews in the keystroke dynamic study. This study aimed at gathering a dataset, build up an assessment system and measure the execution of many anomaly detection calculations on a similar premise. Overall, they established which detectors had the minimal error rate on the data collected (e.g., the Nearest Neighbor (Mahalanobis) locator), and provides an information set and assessment technique that can be utilized by the group to evaluate new identifiers and report comparative results.

Sluganović et al. proposed a system that uses an artificial neural network to distinguish between a genuine user and an impostor with a relatively high success rate [11]. The system was implemented using data collected from various users, and features extracted using C#. The neural network classification and training process of the new data was executed in Matlab.

Features extracted are interkey (the time between successive keystrokes) and hold-times (duration of a given key press). The data is separated into two files, to train the neural network for a single user. The first contains the real user's input data, and the other includes data of non-users attempts to type the password. 10-fold cross validation was used in estimating the performance of the neural network classifier, where 90% was used for training, and 10% used for testing. The intent was for the network to accept all legitimate user inputs. The chosen topology of the artificial neural network is the Feedforward Neural Network. Due to the complexity of the model, a multilayered perceptron was found more suitable than the single layer perceptron network. MLP comprises a group of neurons organized in layers, the input and output layers connect it to the outside world. There is at least one or more hidden layer which is not directly accessible and is used strictly to store the model of the input data.

The testing demonstrated that when using three hidden layers, the neural system indicates poor speculation qualities, e.g. the decision-making process based on a comparison of total password times typed. Explicit storage and learning of the input data model are done using the backpropagation algorithm which works in two phases: forward and backwards. The rate of false negative detections is in most cases lower than the rate of false positive samples. The explanation behind this is, while the network tries to partition the feature space into two classes, genuine and impostor, just the feature vectors having a place with the valid user class, are confined in the feature space. The number of samples that belongs to impostors is quite smaller

than some the actual samples adding to better classification and more misses while classifying the impostor samples.

Alves et al. proposed an authentication system based on keystroke dynamics for computational environments [12]. The system was low-cost, non-intrusive and could be applied in controlled areas to increase security. The algorithm works by observing the typing of the user progressively catching split circumstances in which the key was pressed and released. Five characteristics used in this study are, the ASCII code (American Standard Code for Information Interchange), three splits related, and a duration associated with the key, which would be the time that it stayed pressed. From their data, it was conceivable to trace metrics that could recognize the user. Their study also follows the fixed text approach and a statistical method (Maximum Likelihood Estimation) was used, which produced satisfactory results. Further optimization using differential evolution as continuous learning capped FRR and FAR were below 4%, and the accuracy of high authentications increased by 13% (79% to 92%).

Roy et al. followed the approach that there are some common words a user types daily and that they are habituated to press it in the same rhythm, which is unique and can be used to distinguish them [13]. Their study takes a fixed-text approach and considers not only the rhythm of the common words or familiar characters. The data was obtained using from 15 users gotten from 270 fixed-text rhythms, from the experiment carried out 0.133% EER for the password "kolkata123" where 0.45% and 0.53% EER for the password "password" and "123456" respectively, which were the best values obtained.

Anusas-amornkul et al. study addresses the vulnerabilities from the weak password, by adding keystroke dynamics to a username or a password to improve the authentication process [15]. In their study, they employed the use of three keystroke dynamics methods; these are *k*-means clustering, trajectory dissimilarity, and confidence interval was implemented with the same dataset and results compared. The trajectory dissimilarity technique gives the best accuracy of 96% among these methods. It also provides an EERof 4% as was obtained from previous studies.

Darabseh et al. followed the keystroke dynamics approach based on feature selection [16]. In their study, they addressed the problem of user authentication when faced with the challenge of accurately detecting and reducing the delay time of detection, and the need to balance both. An approach to tackle this problem was to reduce the number of features that need to be learned by a classifier which increases the classification time. A wrapper-based feature selection method was used in this study with the objective of reducing the dimensionality of the user data through identifying a smaller subset of features that represent the most discriminating features in keystrokes dynamic. Feature selection techniques such as genetic and greedy algorithms, best first search algorithms, and particle swarm optimization (PSO) are used to find the best subset features. These selection techniques are integrated with different machine learning classifiers namely SVM, Naive Bayes, and *k*-NN for feature subset selection procedure that can automatically select the most appropriate and representative subset of features. The experimental results showed that the proposed Best Forward Selection (BFS) with *k*-NN reduces 62.25% of the total features (80 features)

while improving the accuracy from 90.47% to 92.85% with only 31 features used. Thus, this allowed for quick detection while maintaining excellent detection accuracy when using less relevant features.

Darabseh et al. aimed at advancing active user authentication using keystrokes dynamics [17]. In their study, the performance of various keystroke dynamics timing features, along with classification performances were assessed and compared. The keystroke features used were key duration, flight time latency, digraph time latency and world total time duration, and with a total of 28 users captured. SVM and k-NN were the classification algorithms used. The experiment carried out showed that key duration time offered the best performance results amongst other keystroke timing features; next was total world time. For both instances when classification was done using the SVM and k-NN, the accuracy of classification was 84%.

2.2 Review of Dynamic Verification Approaches

Rybnik et al. sought to address user authentication especially with remote access [10]. They proposed that due to the vast nature of the internet, it will be difficult to establish a reliable remote authentication, while the distributed nature makes it even more challenging. This approach follows the collection of the keystroke data, obtained with a web application located at http://www.kds.miszu.pl/. The collected data consists of scan code of every keystroke, information latencies, the timing information in milliseconds counted from the first keystroke. The web browser was a factor that influenced the raw data. A total of 1100 samples were gathered from 250 registered users. This study presents an authentication using a one-word phrase

"kaloryfer." The features extracted were dwell and flight, which may be negative if overlapping occurs. Their study aim was to compare fixed texts. Typing errors were also disregarded, with a corresponding flight time between removed letters. Implementing an algorithm, which examines only the dwell and flight times when comparing the matching keystrokes for the same text. The classification algorithm used is the k-nearest neighbour. Data normalization was performed by counting averages and standard deviations for dwell and flight time.

They randomly selected a reference database (known samples compared with unseen samples) and a generalization database (to be classified). The experiments showed a high percentage classification rate for the one-word phrase "kaloryfer" used as a user password. There is also a significant increase in classification accuracy with an increase of k value. Similar experimental setups for two other phrases, foreign language (English) phrase "After some consideration, I think the right answer is:" and native language (Polish) tongue twister "Stół z powyłamywanymi nogami" also obtained from the database. The overall tendencies in classification accuracies were similar, with the best results obtained for nine samples in the reference database (maximal number researched). An exciting observation could be made that shorter phrases tend to profit more from the flight (p in area 0.60.8), while longer ones from dwell (p in area 0.4-0.6). However, it was observed that longer phrases allow for significantly better classification accuracy. This study concludes that because the users typed in a foreign language, it made it difficult for them to type fast failing to represent much of the individual typing habits.

Monaco et al. used the long-text input to authenticate a user, but in earlier studies, passwords or short name strings were used [5]. The difference being that the former ensured that the system continuously authenticates a user even after login, while the later validates/ authorizes a user once during login. The focus was on developing a new classification algorithm which centred on online student test taking and intruder detection. The previous system applied involved a closed application where it is possible to train the system on all of the validated users (approach is also known as "all at once") often used in a multi-class problem. In this study the "one against all" approach is used, where a relationship with the other samples is tested against the claimed user sample. With this, it matches the asserted user's sample against all other samples rather than just one, used in previous studies. Further, the system improved when evaluated on closed populations of 14, 30 and 119 users. The performance metrics used was the ROC curve.

These include the methods used in this study, consists of frontend application (Data Capture and Feature Extraction) and backend (Authentication Classification and ROC Curve Derivation). The data capture involved a Java applet that captured the keystroke data and the PC windows-event clock which recorded the keystrokes data (key press and release times). In this study, a vector of 239 extracted features from the raw data. It is made up of averages and standard deviations of key press and digraph transition times. 78 duration features, 70 key-release-to-key-press and 70 key-press-to-key-press, were made up of mean and standard deviation. 19 percentage features and two keystroke input rates. Outlier removal (removing transition time which is more than two standard

deviations from the mean values) and feature standardization (standardized into 0-1 range). For classification a vector difference authentication was used in transforming from a multi-class problem to a two-class problem, turning the feature space into feature difference space obtaining the within-person distance and between-person distance. ROC curve derivation was used to evaluate the performance of the k-NN classification algorithm, using linear ranking weight to assign the values of k. The result obtained showed that on a sample of 300 or more keystrokes, performance was 98% for 30 users, and 94% of 119 users, while on the 755 keystrokes resulted in 98.3% for 30 users and 96.3% of 119 users.

Monaco et al. focused on reducing the frequency of independent authentication checks with the ability to reduce the false alarm rate [4]. Two time periods considered are the length of the pause which needs to be shorter than the entry time of the intruder. The second is the length of the data capture authentication window. With the intruder, the scenario needs to be sharp enough to catch the intruder before any significant harm is done and yet long enough to accurately detect and reduce the false alarm rates. The method used is similar to [5], where the results obtained from the experiment performed showed that as the number of keystrokes per test increased the EER decreased and EER increased with population size. With 14 participants the performance was 99%, and with 30 participants it reduced to 96%. It suffices to say that, given a closed system the set of user data was tested against the data in the poll to determine whether to authenticate or not.

Kaganov et al. proposed a new hybrid method which was a combination of a new keystroke data representation model based on functions and machine learning algorithms based on decision trees [14]. With the approach tested in static and dynamic authentication scenarios on the benchmark Si6 dataset collected by the authors. The new keystroke data representation was intended to retain information about the order, times and key values of keystroke actions in a single fixed size feature vector. The difference between previous models and the proposed model of the authors was that it was both simple and complex. Simple models (based on press-release) cannot store information about sequences of keystrokes. Complex representation (e.g., n-grams), on the other hand, can store information about sequences of a keystroke, making it computational expensive and over-trained. A decision tree algorithm was used to address this. For the dynamic authentication, the reward-punishment method was applied to test the viability of model combinations and algorithms. A few experiments were carried out on the dataset Si6 and the data gathered by the authors of this research. The experiments demonstrated the pertinence of the proposed method to deal with static and dynamic authentication. Specifically, on the Si6 dataset used as a part of many papers as a benchmark the approach in this study gives an EER of 6%. The approach used in this study ranks high regarding the outcome of an experiment carried out on the Si6 dataset. The experimental results confirm that the proposed hybrid method applies to real-life authentication systems.

Mondal et al. study analyzed the performance of continuous user authentication and identification system for a user using Pairwise User

Coupling (PUC) on the collected dataset for analysis [18]. The dataset combined both keystroke and mouse usage behaviour data, obtaining an identification accuracy of 62.2% for the closed set experiment, while the system required an average of 471 activities to recognize an impostor. 58.9% detection and identification rate (DIR) was obtained, needing 333 actions to detect an impostor in the case of an open set experiment.

2.3 Conclusion

In this chapter, we have reviewed literature categorized in the two approaches of keystroke dynamics, with focus on the methods used, and results obtained, the objective is to understand what was done in the past, and the grounds that has well been covered in the area keystroke dynamics.

CHAPTER 3

Materials and Methods

The methods used in this thesis follow some known machine learning approaches. Machine learning is a branch of artificial intelligence that provides a method for training different model algorithms while testing it on some sets of input with the performances evaluated based on defined evaluation metrics. We have employed several methods for classification which we further look in-depth. Figure 3.1 shows the workflow of our approach. First, we obtained the dataset, extracted new features using the feature extraction algorithm, selected relevant features, and evaluated them based on our classification methods.

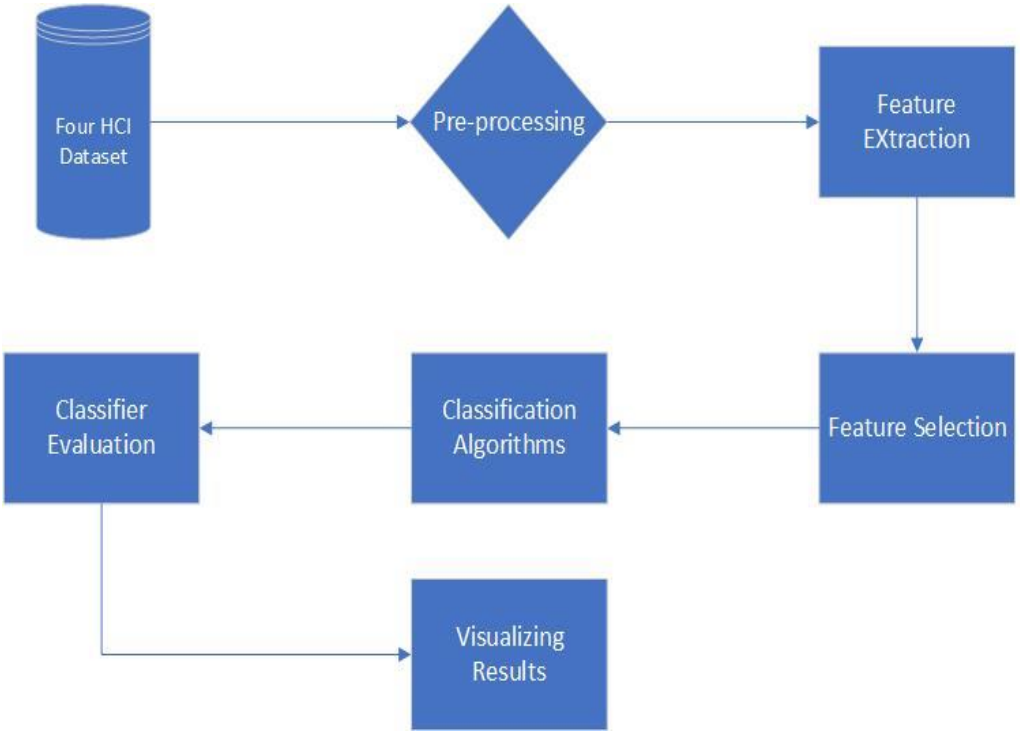

Figure 3.1: Workflow of our method.

3.1 The Datasets

We have used two datasets in this thesis; these are the Four HCI task dataset, and the Villani keystroke dataset [4,5,20].

3.1.1 The Four HCI Task Dataset

The dataset used for this thesis is the Four HCI Task dataset [4,5]. It is a privately available dataset, accessible upon request. It contains 366 samples of keystroke timing information (flight and dwell time) obtained when 57 users typed, using designated keyboards without changing them during the entire process of data collection. Before the raw data is transformed into a form which is more understandable, it contained information such as (user, session, time press, time-release, keycode, key name). The user describes the specific individual. Session is the active time during which the keystroke timing information of users is collected. Time-release and time-press are the timing information which are also known as flight and dwell time respectively this describe the information when specific keys are pressed and released, this timing information is measured in milliseconds. Implementation focuses on intruder detection and online test takers. The approach used for the data collection was the free text approach, while the following tasks were performed.

- Task 1: Browsing the web.
- Task 2: Editing a paragraph.
- Task 3: Playing Star Bubbles.
- Task 4: Playing Solitaire.
- Task 5: Question answering in an online quiz.

Figure 3.2 describes the contents of our dataset after performing feature extraction. We have obtained a total sample of 366 keystroke data, 218 feature vectors, from 57 users.

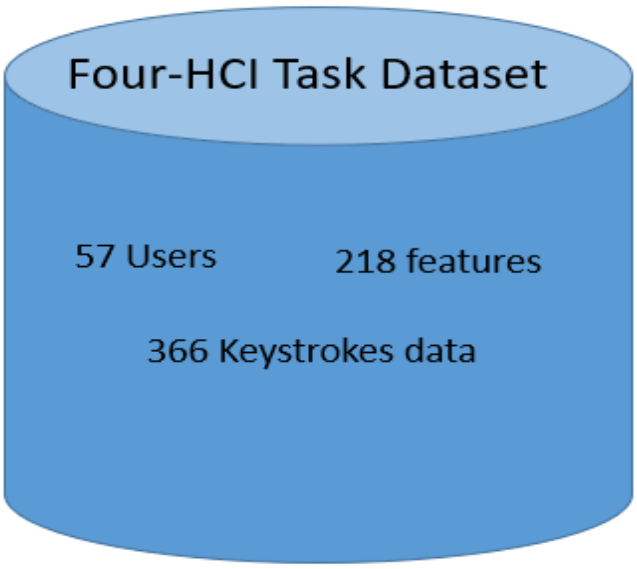

Figure 3.2: Makeup of the dataset.

3.1.2 The Villani Keystroke Dataset

The Villani keystroke dataset is the other dataset used in this thesis [4,20]. It is also a privately available dataset accessible upon request from the authors. It is made up of an average of 1,621 keystroke timing information collected from 143 users over specific periods. The uniqueness of this dataset is that some keystroke data were collected over the Internet, suitable for Internet security applications [20]. It focuses on long text, and applications using arbitrary text input. A Java applet collected keystroke data over the Internet with users required to type their names and specified text in the field box. The dataset contains long free text and fixed text input from users answering questions, and copies and pastes actions of fables. About 143

university students and faculty provided at least five samples with each in different recording sessions, with some as little as a sample from a user. The dataset contained information such as user, session, input type, task, platform, gender, age group, handedness, awareness, time press, time-release, key name, and location. Handedness and awareness provide information on which hand the individual uses to type, and whether they are aware of this. A platform describes the mode of typing either laptop or desktop computer. Input type and tasks explain the keystroke approaches used (free- text or fixed text).

3.2 Preprocessing and Feature Extraction

Data preprocessing is a technique that involves the transformation of the raw keystroke data into an understandable and readable format. With real-world data often incomplete, preprocessing a data resolves some of these issues [20]. Data during preprocessing goes through the following steps:

- Data Cleaning: Data cleaning involves processes that include smoothing the noise, filling in missing values, resolving inconsistency found in the data. This is relevant because during the process of data collection there can be pauses or moments when the user took breaks before continuing typing. We have taken this into account because our study is based on continuous authentication, which continuously authenticates a user after verification.
- Data Integration: Raw data with different representations are placed together, and conflicts resolved.

- Data Transformation: Here, the data is normalized, aggregated and generalized.
- Data Reduction: Aims at presenting a more compressed data after the data has gone through most of the above process.

3.2.1 Feature Extraction

Feature extraction is a part of data preprocessing which involves the transformation of our raw keystroke data into a format that is more understandable. Outlier removal and standardization were two other methods used at the stage of feature extraction [5,20]. Outlier removal is significant because a user could pause for a stretch, get a glass of water, long presses of the keys or for other reasons resulting in outliers, producing a data having a skewed feature measurement. In the case of our study, it involves removing timing information keys that are more than two standard deviations from the mean value. After the outliers are removed, averages and standard deviations are calculated recursively until no outliers are present. After this, feature standardization is performed standardizing the feature measurement into the range of 0 to 1 for each feature measurement at +1 and -2 standard deviations from the keystroke samples obtained from all users.

The feature extraction extracts a feature vector of 218 features from the timing information obtained for the raw data [4,5,20]. These features are statistical made up of means and standard deviations, comprising key presses and key releases transition times. The 218 features are grouped as follows:

- 78 duration features, made up of 39 means and 39 standard deviations of individual letter and non-letter keys, and a group of letter and non-letter keys. Grouping of letter or non-letter keys characterizes the concept of n-graph, which primarily involves the combination of two or more keys.
- 70 key-release-to-key-press transition features, comprising 35 means and 35 standard deviations of between letter or groups of letters, between letters and non-letters or groups, between non-letters and letters or group, and between non-letters and non-letters or groups.
- 70 key-press-to-key-press transition features comprising 35 means and 35 standard deviations similar to the key-release-to-key-press transition that has been described. Figures 3.4 and 3.5 describe the letters, non-letters, letter/letter and the various combinations from above. Figure 3.3, is a snippet of some features extracted from the raw keystroke data. These features are made up of means and standard deviations and are in three categories as explained above.

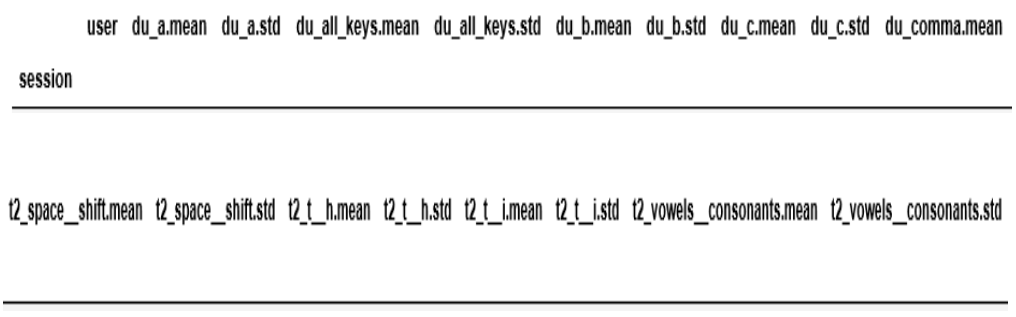

Figure 3.3: Snippet of features extracted.

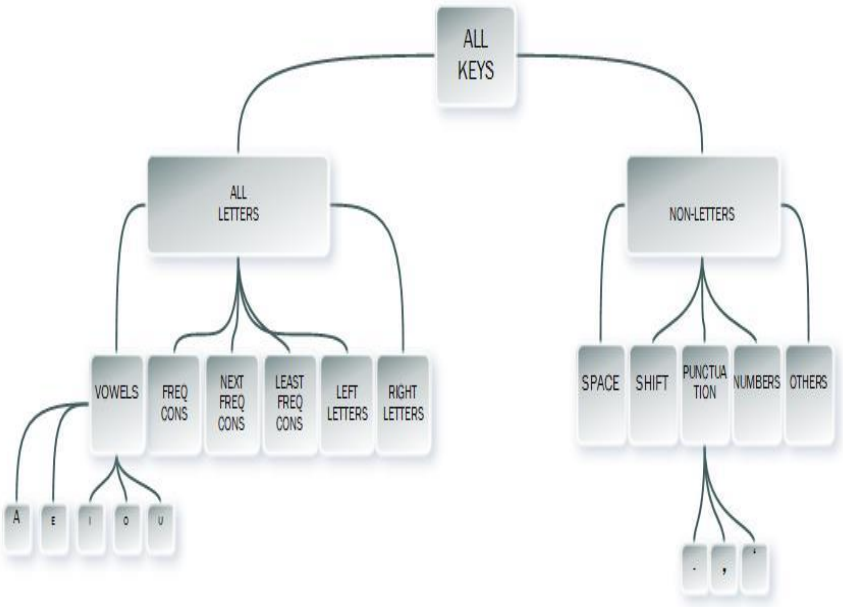

Figure 3.4: Description of the 78 duration features.

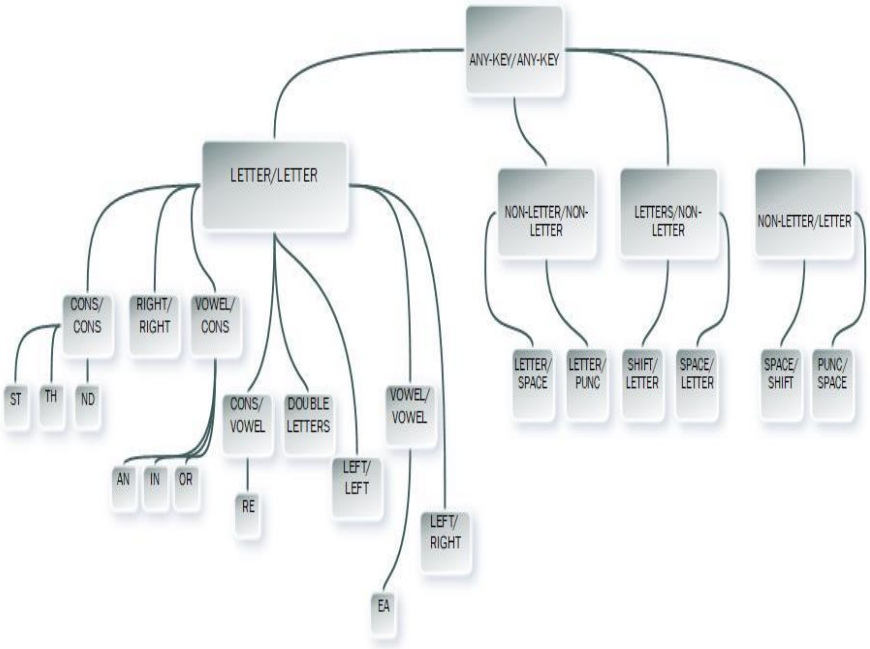

Figure 3.5: Description of the 70 key-release-to-key-press and key-press-to-key-press transition features.

The datasets used in this thesis, both contain a total of 218 similar categories of feature, described above. Figure 3.6 shows a breakdown of the total features extracted from the Villani Keystroke dataset. It contains a total sample of 1,621 keystroke data, with 218 feature vectors, obtained from 143 users.

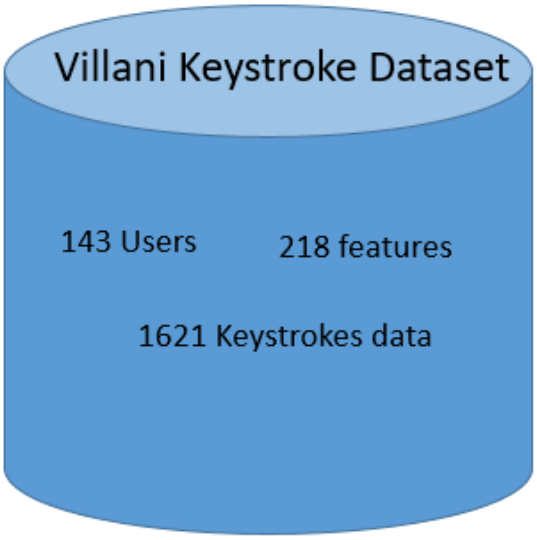

Figure 3.6: Makeup of the Villani keystroke dataset.

3.3 Feature Selection

After data preprocessing and feature extraction, we employed feature selection to select relevant features which improved the classification algorithms [24]. Feature selection is defined as the process of selecting a subset of relevant features for use in classification or regression. Given some features, we aim to choose the most important of them, reducing the number but at the same time retaining the discriminatory class information. Some reasons we worked with fewer features are:

- It decreases the computational resources (time/space) needed for training and classification.
- Also, interpretation of data can lead to exciting conclusions and reduce overfitting.
- Another reason for reducing the features of a subset is that although some features may carry good classification information, only when treated separately.

There exist various methods and techniques for selecting features. These techniques are grouped into wrapper methods and filter methods.

3.3.1 Filter Methods

Filter methods are based on statistical tests measuring intrinsic properties of the dataset features, here the variance of each feature is computed, and the subset of features is selected based on a specified threshold [24]. Thus, the features are ranked based on the scores, and the best scores are used in building the model, the others are kept in the dataset, but are not used in the analysis. Some filter-based feature selection methods include [24]:

- Pearson Correlation
- Mutual Information
- Kendall Correlation
- Spearman Correlation
- Chi-Squared
- Fisher Score
- Count Based
- Information Gain

The filter method approach is shown in Figure 3.7, first by selecting the sets of features, then selecting the best subset based on their scores obtained from an algorithm and evaluating their performances.

Figure 3.7: Flowchart of the filter method.

3.3.2 Wrapper Methods

Wrapper methods consider the selection of the set of features as some search problem, where different combinations are prepared, evaluated and compared with others [24]. The search is conducted within a subset of the feature space, with the goodness of the subset evaluated based on a classifier and performance metrics. Also, validation is performed on every step of the process. In the wrapper method, we take a subset of features and train a model using them. One disadvantage of the wrapper method is that it is often computationally expensive, but regardless very efficient. Common wrapper method includes recursive feature elimination, forward selection, and backward elimination [24].

In recursive feature elimination, it aims at finding the best performing subset of features. It creates a model, and recursively iterates the process by keeping aside the best and worst models. Then, with the remaining features, it constructs the model until the process becomes exhaustive, and the features are ranked based on elimination. Backward elimination begins with all features and removes the least significant features that do not

improve the performance of the algorithm. Forward selection starts with having no feature in the model. We begin by adding those features that best enhance the model, and we stop adding when there are no changes to model's performance. Figure 3.8 describes in generality the flow path of a wrapper model, from selecting the best features available, generating a subset, and developing the learning algorithm.

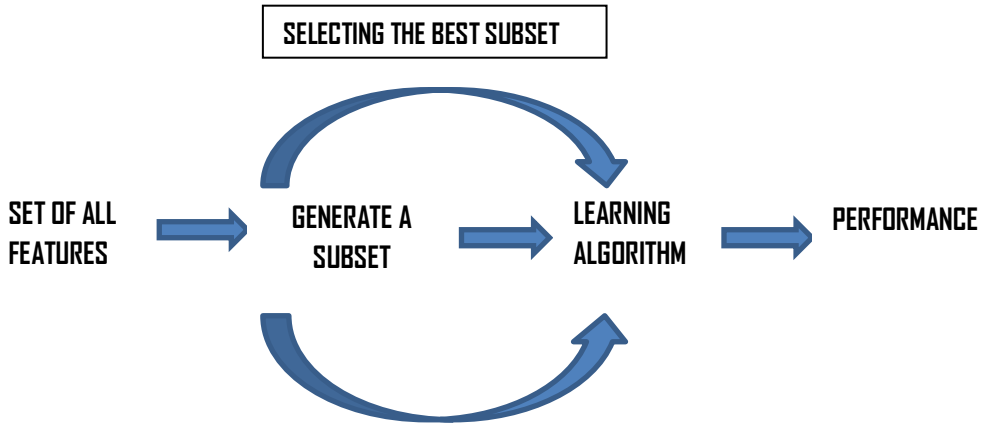

Figure 3.8: Work flow of a wrapper model.

In this thesis, we have used the Minimum Redundancy Maximum Relevance (mRMR) method for selecting the features used in the classification algorithms. We will further discuss this method.

3.3.3 Minimum Redundancy Maximum Relevance

Minimum redundancy maximum relevance was first developed as a robust filter method for selecting features but over the years its been combined with wrapper selection methods, and with this method producing promising results over different datasets from several research areas. mRMR is a method of feature selection, which selects subsets of features having

maximum relevance and removes the redundant features not selected, using mutual information to analyze relevance and redundancy [21]. There are also other approaches of mRMR, such as the correlation approach, which selects the features having a high correlation with the class and, low correlation between themselves. In this thesis, we have used the mutual information approach, which consists of two types MID (Mutual Information Difference) and MIQ (Mutual Information Quotient), both representing the difference and quotients of relevance and redundancy. Figure 3.9 shows the breakdown process of how a feature is selected based on the mutual information approach of mRMR.

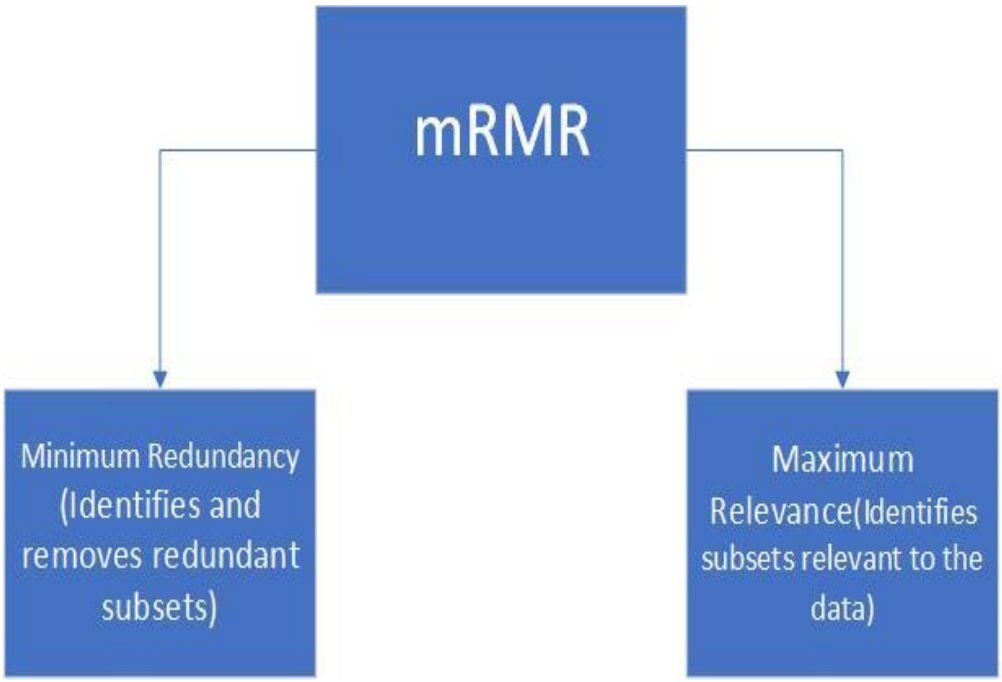

Figure 3.9: mRMR approach for selecting features.

In this thesis, we have used both types of mRMR (MID and MIQ) and was observed that similar features were selected from the Four HCI Task dataset. It was also found that the features selected from the Villani

keystroke dataset performed underwhelmingly with features from MIQ, against the results when the entire features were used, but with MID the results improved significantly. When the number of feature vectors was reduced, the performance metrics also reduced, but when increased to 50 features it even enhanced it. Figure 3.10 shows the 20 features selected from the Four HCI task dataset using the mRMR feature selection algorithm.

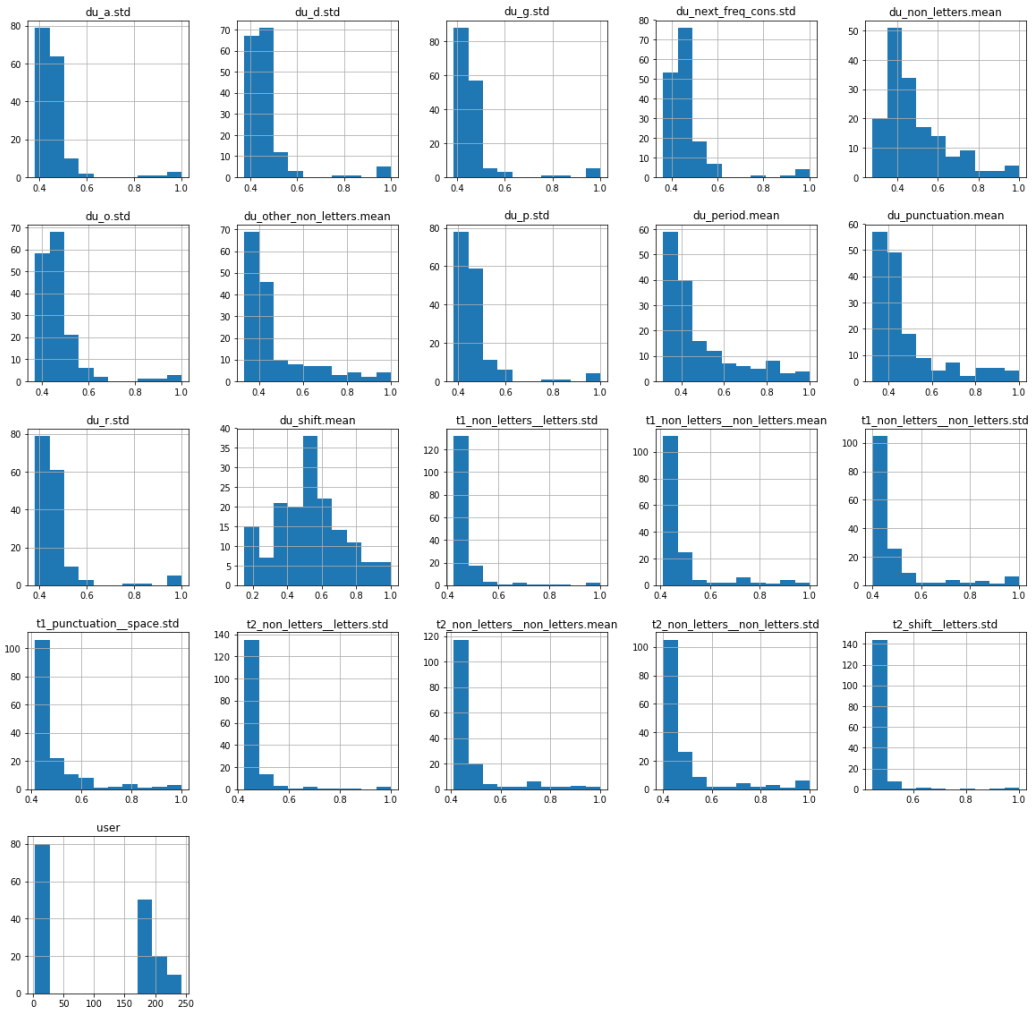

Figure 3.10: The 20 features selected using mRMR.

For discrete variables, mRMR for minimum redundancy is calculated as follows:

$$\min\ W_1,\ W_1 = \frac{1}{|s|^2} \quad \sum_{i,j \in s} I(i.j), \tag{3.1}$$

where S is a set of features and, I (i,j) is mutual information between features i and j

For maximum relevance:

$$\max\ V_1,\ \ V_1 = \frac{1}{|s|} \sum_{i \in s} I(h,i), \tag{3.2}$$

where h is the class, in this case, the users are the classes.

3.4 Classification Techniques

Classification is a data mining technique that assigns users in this case, to target categories, groups or classes. Classification aims at accurately predicting the target class for every given data. We have used the k-NN, SVM, Naïve Bayes and grid search algorithms in this thesis.

3.4.1 *k*-Nearest Neighbour

k-nearest neighbour is a non-parametric technique used for classification or regression. The input consists of k closest training examples, and the output is the class membership or class. An object is classified by the majority of the worth of its neighbour. Figure 3.11 illustrates the k-NN algorithm. Consider a two-class classification problem. We pick a random point in our sub-feature space, k, which is our start point or testing point

[22]. Thus the class with the highest number of elements in the sub-feature space becomes the class.

How k-NN works:

- Start with an empty region R (sub feature space) equivalent to x (feature space).
- Choose the number of k and a distance metric.
- Grow a region until it captures k samples of x.
- Find the nearest neighbours of the sample that we want to classify.
- Assign the class label by majority vote.

The distance function used in this thesis is the Minkowski distance [24]. Other common distance functions are the Euclidean and Manhattan distance. Minkowski distance is typically used when $p = 1 \; or \; 2$ corresponding to Manhattan and Euclidean distance respectively, as follows:

Minkowski distance:

$$d(x,y) = \left(\sum_{i=0}^{n-1}(x_i - y_i)^p\right)^{\frac{1}{p}} \tag{3.3}$$

Euclidean distance function:

$$d\ x,y\ = \left(\sum_{i=1}^{d}(x_i - y_i)^2\right)^{\frac{1}{2}} = \sqrt{\ x-y\ ^t\ \ x-y} \tag{3.4}$$

Manhattan distance (city block distance):

$$d\ x,y\ = \left(\sum_{i=1}^{d} x_i\ - y_i\right) \tag{3.5}$$

The choice of k is vital in finding a right balance between overfitting and underfitting. When Euclidean distance is used, it is essential to standardize the data, so that each feature equally contributes to the distance.

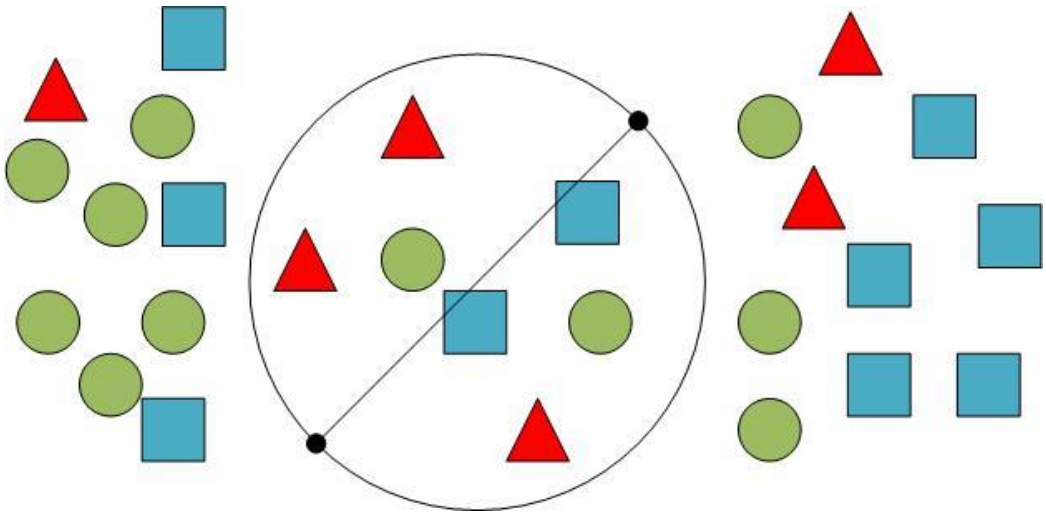

Figure 3.11: k-NN classification.

3.4.2 Support Vector Machine

SVM can be considered an extension of the perceptron algorithm, which minimizes misclassification errors [22,23]. However, in SVM, the goal is to optimize or maximize the margin. The margin is defined as the distance between the separating hyperplane called decision boundary, and the training samples closest to this decision boundary, which are called the support vectors as shown in Figure 3.12. The idea behind the decision boundaries having a large margin is that they lead to a lower generalization error, while models with small margins are more prone to overfitting. To understand margin maximization, we consider the positive and negative hyperplanes parallel to decision boundary:

$$\omega_0 + \omega^T x_{pos} = 1 \tag{3.6}$$

$$\omega_0 + \omega^T x_{neg} = -1 \tag{3.7}$$

After subtracting the positive and negative hyperplane, we further normalize the vector ω, hence we obtain the equation below:

$$\frac{\omega^T(x_{POS}-x_{neg})}{||\omega||} = \frac{2}{||\omega||} \tag{3.8}$$

The left-hand side of the above equation is defined as the distance between the positive and negative hyperplane, which is the margin we need to maximize. SVM aims at finding the plane of highest separation, according to some distance measure, using kernel functions [24]. The kernel function represents a dot product of data mapped to higher dimensions by transformation, where the data becomes separable. Kernels are useful when the data is not linearly separable and can be separated only in higher dimensions [23]. SVM has several kernel function types. These are:

- Linear Kernel: $K(x_i x_j) = x_i^t x_j$ *(3.9)*
- Polynomial Kernel: $K(x_i x_j) = (x_i x_j + 1\ )^{\ h}$ *(3.10)*
- Radial basis function kernel: $K(x_i x_j) = \ e^{-||x_i - x_j||^2/2\sigma^2}$ *(3.11)*

The parameters used for the kernel types mentioned above are: the cost, gamma, and degree parameters. The cost parameter is used for measuring the width of the margin. Small cost value means fewer misclassification errors, and large value also indicates large error penalties. The gamma parameter defines the point to which a single training set performs. Low gamma values mean far, and high gamma values imply close. The degree parameter controls the degree of freedom of the polynomial kernel. The linear kernel uses the cost parameter. The polynomial kernel uses the gamma and degree parameters. The RBF kernel also uses the gamma and cost parameters. Often, the performances of SVM does not provide optimal

performances. To optimize the performance of the SVM classifier further, we have used the grid search method, which exhaustively performs classification from the list of parameters specified, and outputs the best performing result.

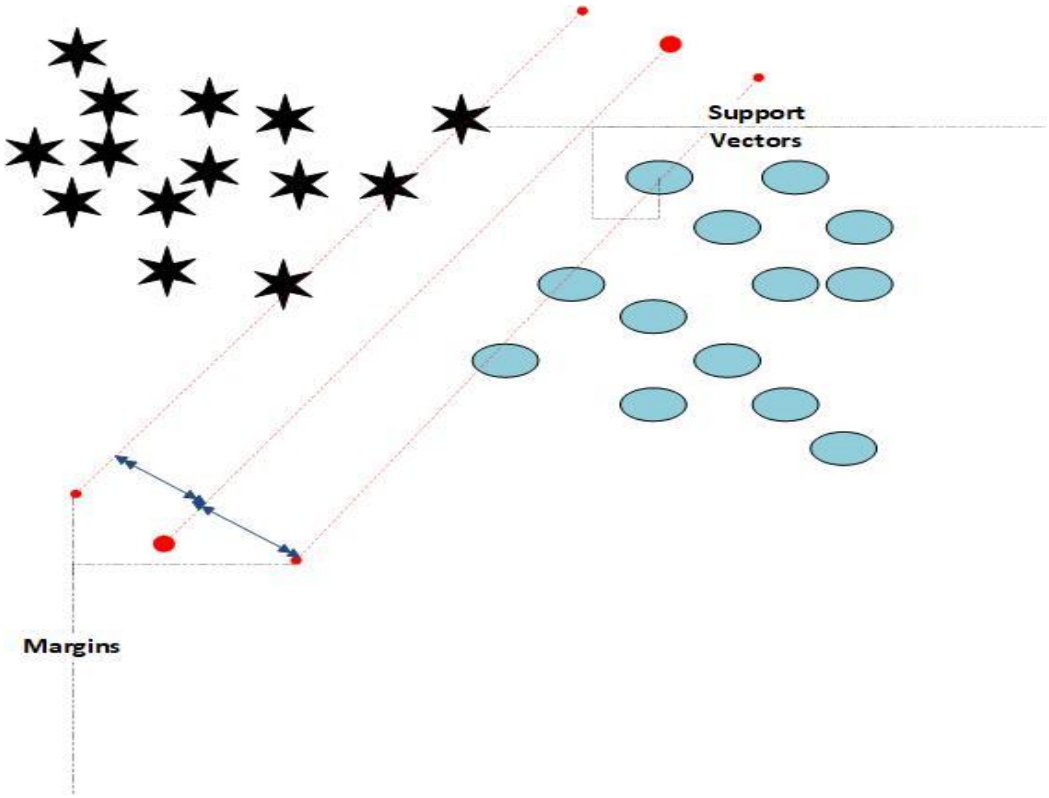

Figure 3.12: Support vector machine.

3.4.3 Naïve Bayes

Naïve Bayes algorithm is used for either a two class or multiclass classification problems. It hinges on quantifying tradeoffs between various classification decisions using probability, and the cost that accompanies such decisions [24]. Unlike other classifiers, it is capable of handling multiple class directly. Bayes classifier centres on the assumption that the decision problem

is posed in probabilistic terms and all relevant probability values are known. Given a state of nature ω, there exist different states ω_1 ,ω_2 ,$\omega_3 \ldots \omega_c$, in a multiclass problem, states are equivalent to the classes. If these are the only information we have for $p(\omega_1)$ and $p(\omega_2)$, then the decision will be:

$$\omega_1 \ if \ p \ \omega_1 \ > \ p \ \omega_2 \tag{3.12}$$

$$\omega_2 \ if \ p \ \omega_1 \ < \ p \ \omega_2 \tag{3.13}$$

And when $p(\omega_1) \neq p(\omega_2)$ there is a 50% chance of being correct. In other to improve these employ a method known as class conditional probability density function $p(x|\omega)$. Bayes formula is defined by the following equation:

$$p(\omega_j|x) = \frac{p(x|\omega_j)p(\omega_j)}{p(x)} \tag{3.14}$$

$$\text{where, } p(x) = \sum_{j=1}^{c} p(x|\omega_j) \ p(\omega_j) \tag{3.15}$$

Given the entire feature space x, taking action implies a risk called conditional risk. The aim is to minimize the expected loss by taking measures that reduce the risk.

Naive Bayes classifier is based on applying Bayes theorem with firm independence between features. It is a conditional probabilistic model, given an instance of a problem to be classified. The objective of this classifier is to classify new objects based on the already existing classes. In simple terms, naive Bayes classifier is:

$$posterior = \frac{prior * likelihood}{evidence} \tag{3.16}$$

Prior probability is based on previous experience for classifying objects and is used to predict the outcome before it occurs. Likelihood means how likely new cases will be classified to the class with the most samples. The evidence is the normalizing constant. Given a set of variables $X = x_1, x_2 \dots x_d$. The aim is to construct the posterior probability for C_j for possible outcomes of $C = c_1, c_2 \dots c_d$. Using Bayes' rule, we obtain:

$$p(C_j|x_1, x_2 \dots x_d) \alpha\ p(x_1, x_2 \dots x_d|C_j) pC_j \tag{3.17}$$

where $p(C_j|x_1, x_2 \dots x_d)$ is the posterior probability of class membership. Since it assumes that the conditional probability of the independent variable are statistically independent, the likelihood is therefore obtained as:

$$p(X|C_j) \alpha \prod_{k=1}^{d} p(x_k|C_j) \tag{3.18}$$

Therefore, the posterior probability becomes:

$$p(C_j|X) \propto p(C_j) \prod_{k=1}^{d} p(x_k|C_j) \tag{3.19}$$

Naïve Bayes assumes that all the predictors/features are independent from one another. Consider a multiclass problem with 2 or more classes, naïve Bayes considers the classes as predictors and considers each of these classes independently. It looks at each predictor independently and tries to calculate the probability that this unknown object will belong to a even class.

3.5 Other Methods

Other methods used in this thesis include grid search, which we have used to optimize the SVM classification algorithm, stratified m-fold cross-validation, one vs all approach for multiclass classification, principal

component analysis for visualizing our data, and confusion matrix for evaluating the performance of our classification algorithms.

3.5.1 Grid Search

Grid search is cross-validation, hyperparameter optimization technique for tuning sets of parameters of a learning algorithm which is optimized separately. It further helps improve the performance of a model by combining the hyperparameter values [23,24]. Grid search is an approach that utilizes an exhaustive search paradigm where a list of parameters is specified for different hyperparameters, and the computer evaluates the model performance for every combination of parameters to obtain a set of values. In this thesis, we have optimized the SVM algorithm using grid search. First, we specified a range of parameters, and our program made evaluations based on these parameters to produce the best values in the end.

3.5.2 Stratified *m*-Fold Cross-Validation

Cross-validation is a technique used in accessing the performance of a machine learning algorithm. We have used it to provide an estimation of the accuracy of our model. m-fold cross-validation partitions the feature space or our dataset into m equal parts. If it is divided into ten folds, it partitions the original sample randomly into m similar sized subsample retained as the validation data for testing the model, the remaining $m-1$ subsamples for training data, with this process iteratively implemented on the specified number of folds. Stratified m-fold CV is particularly vital in the event where

we have an imbalanced dataset. The folds are made to preserve percentages of each class. Here, the class proportions are retained in each fold ensuring that we have adequate representation of the class in the training dataset. In Figure 3.13, each fold is broken into one portion for testing, and the rest for training, with the number of folds determining the number of iterations that we will have.

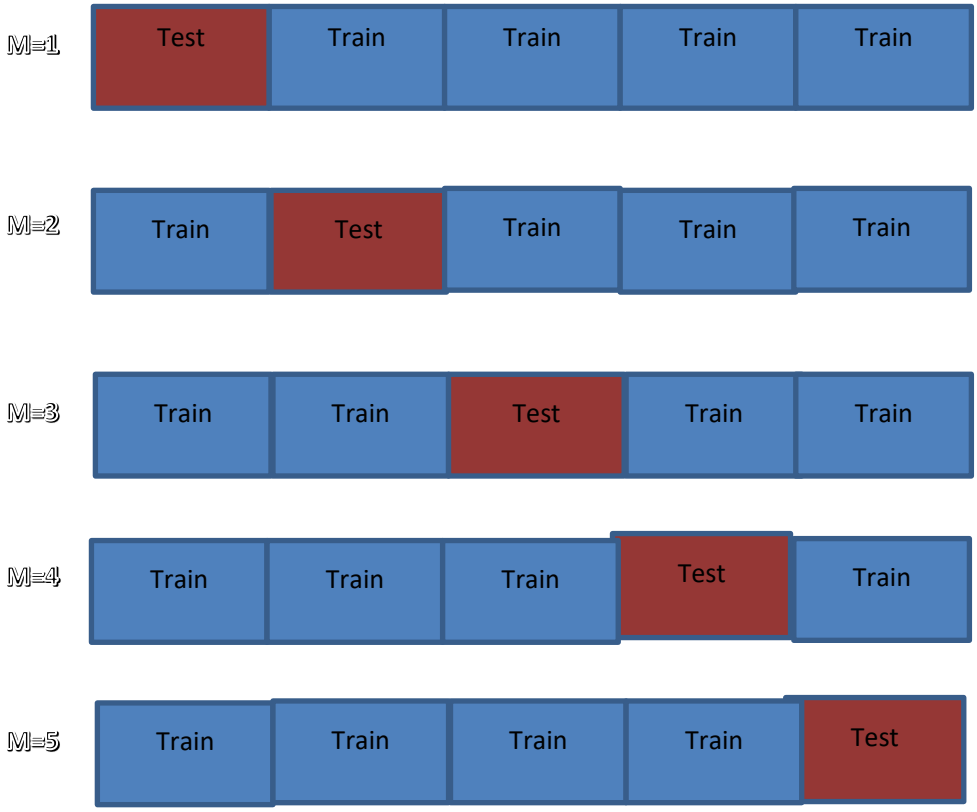

Figure 3.13: m-fold cross-validation.

3.5.3 One Vs All Approach

Multiclass classification is the problem of classifying instances involving two or more classes [23]. The problem involved in multiclass classification can be tackled with various approaches (One vs One, One vs All, and All vs All). In this thesis, we have used the One vs All approach, which consists of

fitting a classifier per class, with each classifier the class is fitted against the rest classes. It is also known as One Vs Rest. One favourable position of this approach is its interpretability. Since each class is well represented by one and one classifier alone, it is conceivable to pick up information about the class by assessing its relating classifier. As illustrated in Figure 3.14 OVA uses samples from all the class to train the binary classifier. The samples from one class are treated as class ω_1 and the other sample considered as the rest class ω_r. OVA is the most frequently utilized procedure for multiclass classification problems and is a reasonable default decision.

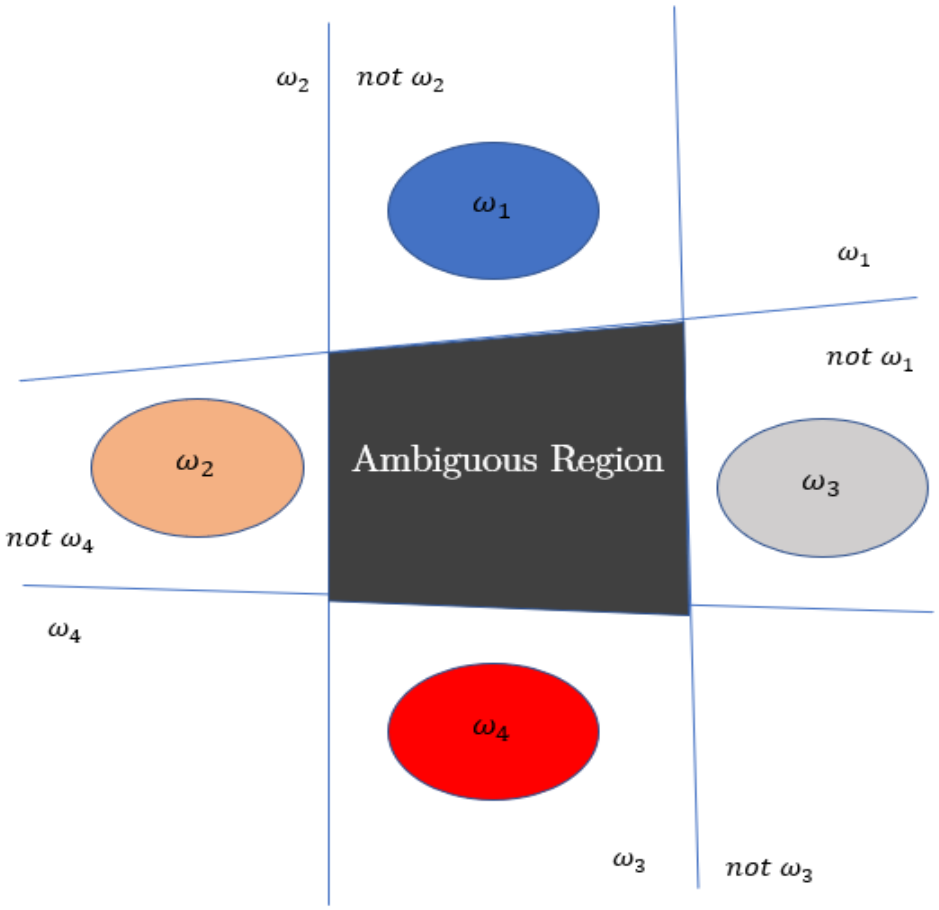

Figure 3.14: One Vs All approach of multiclass classification.

3.5.4 Principal Component Analysis

We have used principal component analysis (PCA) in this thesis. PCA was used in this regard to reduce the dimensionality of our dataset to lower dimensions. It finds the direction of the maximal variance of the dataset or finds directions that are mutually orthogonal [22]. The steps taken in PCA are as follows:

- Firstly, the d-dimensional mean vector $\boldsymbol{\mu}$ and d by d covariance matrix Σ are computed for the entire dataset.
- Next, the eigenvectors and eigenvalues are computed and sorted to decreasing eigenvalues, where λ_{i} is represented as eigenvalues and e_i the eigenvectors.
- The largest k eigenvalues are chosen from a spectrum of eigenvectors.

3.5.5 Performance Metrics

We have used the confusion matrix as a framework of evaluating the performances of our classification algorithm. Since the error rates give us an overall insight about the performance of the classifier but often provides no information about a particular class and its relationship to the other classes, a useful tool is the confusion matrix [22]. Some valuable components are seen Figure 3.15 and, defined as follows:

- TP = True Positive, positive classified as positive
- TN = True Negative, negative classified as negative
- FP = False Positive, negative classified as positive
- FN = False Negative, positive classified as negative

We have used Accuracy and F1-score for evaluating our classification algorithms both derived from the confusion matrix.

- $n = No\ of\ Samples$
- $Specificity = \frac{TN}{TN+FP} = 1 - FP\ rate$ *(3.20)*
- $Sensitivity\ (recall) = \frac{TP}{TP+FN} = TP\ rate$ *(3.21)*
- $precision = \frac{TP}{TP+FP}$ *(3.22)*

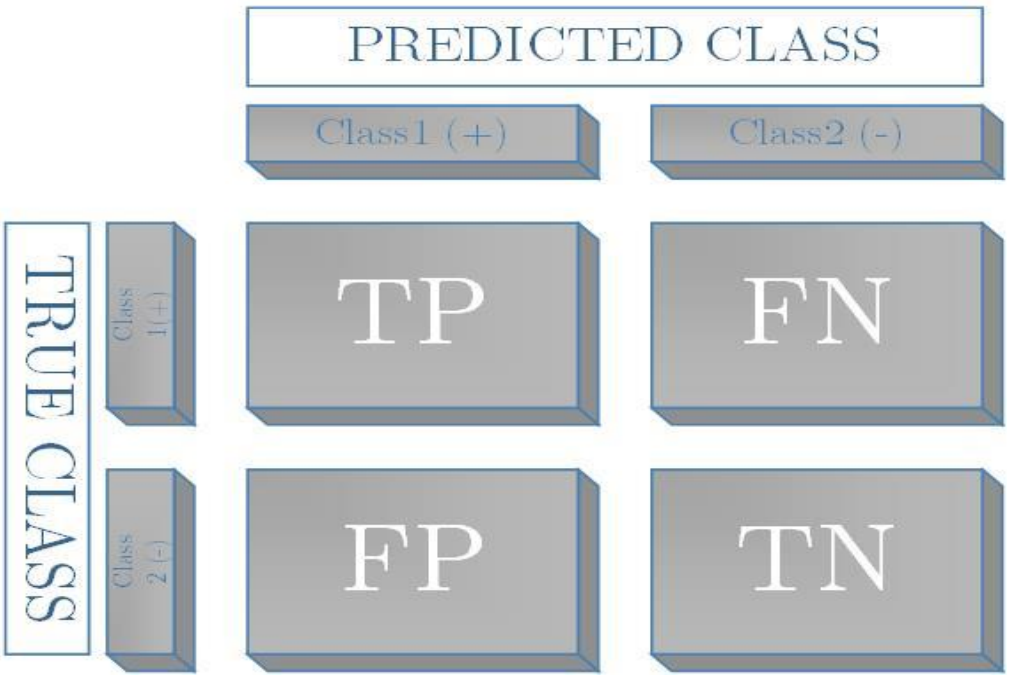

Figure 3.15: Confusion matrix.

- Accuracy is the ratio of correctly predicted observation to the total observations.

  $Accuracy = \frac{TP+TN}{n}$ *(3.23)*

- F1-score is the weighted average of Precision and Recall. F1-score is particularly useful when we have a class imbalance problem.

  $F1\ score = \frac{2(precision*recall)}{precision+recall}$ *(3.24)*

We have also used the Receiver Operating Characteristic (ROC) curve for multiclass classification, which is a plot of true positive rate against false positive rate [24]. TP rate is also known as recall, and FP rate is otherwise known as fall out or probability of false alarm. The Area Under Curve (AUC) gives a competitive comparison of the TP and FP rate. Given the ROC space, the best AUC is achieved when the ROC is closest to the top left corner of the plot and worst when close to the bottom right corner of the ROC space. We have obtained micro average and macro average AUCs in our experiments. Micro average AUC computes the sum of all true positives and false positives across all classes, while macro average AUC takes the average of the curves across all classes treating each class independently.

3.6 Conclusion

In this chapter, we have introduced and explained the contents of the Villani keystroke dataset and the four HCI task dataset used in our experiments, defined key concepts, introduce classifiers used in our proposed method. We also introduced other methods which include grid search for optimizing the SVM algorithm, stratified m-fold cross validation and methods for visualizing and measuring performances of the classification techniques.

CHAPTER 4

Experimental Results

We have used the Four HCI task dataset and the Villani keystroke dataset in our experiments. The implementation of all algorithms have been done using the Scikit-learn Python library. For classification purposes, our data went through a sequence of steps to present it in a useable format. The following steps were taken:

- First, we extracted the timing features with a feature extractor algorithm to obtain 218 feature vectors for the dataset used.
- On obtaining the new feature vectors, we further preprocessed the data by removing users with one sample data.
- Furthermore, we set a threshold of ten samples per user, with the users having less than ten samples removed in the case of both datasets. Also, we set a threshold of samples per user for the dataset used in the half sequence experiment.
- Finally, we applied classification algorithms on these datasets.

The nature of features extracted comprised of features consisting of means and standard deviations of 78 duration features, 70 key-to-release-to-key-press features, and 70 key-press-to-key-press features. These three categories are grouped into transition (key-to-release-to-key-press and key-press-to-key-press) represented with the prefix (*t1*) and (*t2*) respectively, and duration features represented with the prefix (du). Duration features in this case is the time spent on a key, *t1* transition features is the difference between

the press time of a second key from the release time of the first key, while *t2* is the difference between the press time of the second key from the press time of first key. The majority of features selected with the feature selection algorithm were made up of duration features and others making up the transition keys; this implies the relevance of this group of features in user authentication.

4.1 Experimental Results from The Four HCI Task Dataset

In Chapter 3, we explained the methods used in this thesis. An algorithm that has frequently been used in the study of keystroke dynamics is the *k*-NN algorithm. In most datasets, this algorithm was found to perform exceptionally well, and as a result, we have applied this algorithm to our datasets to see how well it performs. In our method, we extracted and selected relevant features as well as performed classification algorithms on the dataset. We classified our data using *k*-NN, SVM, Naïve Bayes, while the SVM algorithm was optimized using grid search optimization. Phase 1 of our experiment involved using all 218 features in our experiments. In Phase 2, we performed mRMR feature selection to reduce our feature vectors to 20 features, and in Phase 3 we selected 10 feature vectors.

Using the *k*-NN algorithm we selected *k* from 1 to 10 as shown in Figures 4.1 and 4.2; we chose this based on the number of samples in each class. This thesis expands on the work done by [4,5], and their study was conducted used the *k*-NN algorithm.

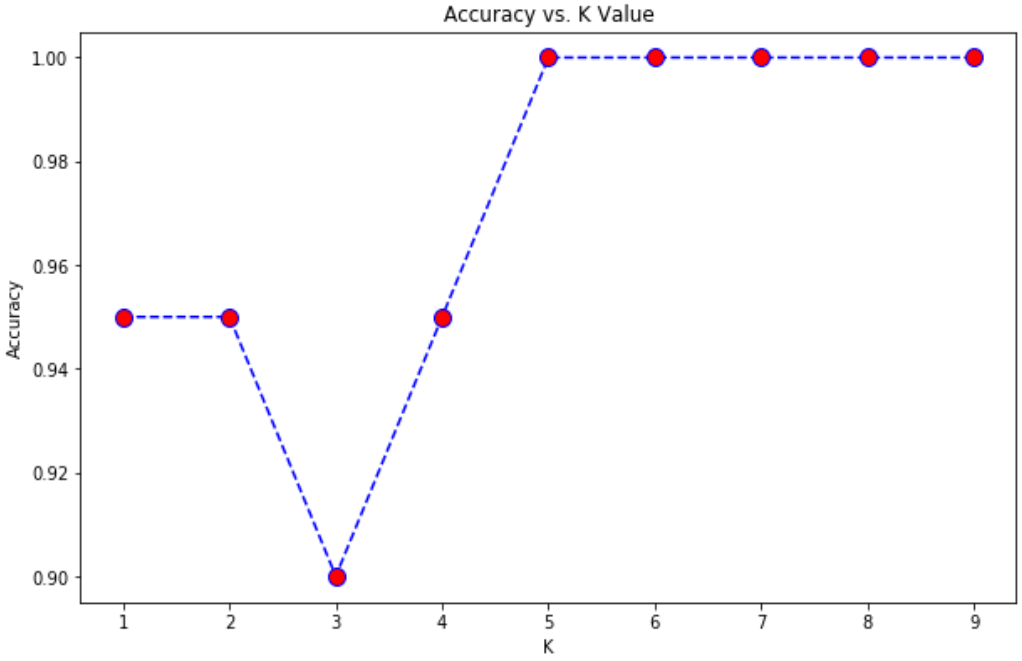

Figure 4.1: Accuracy vs k value with 218 features.

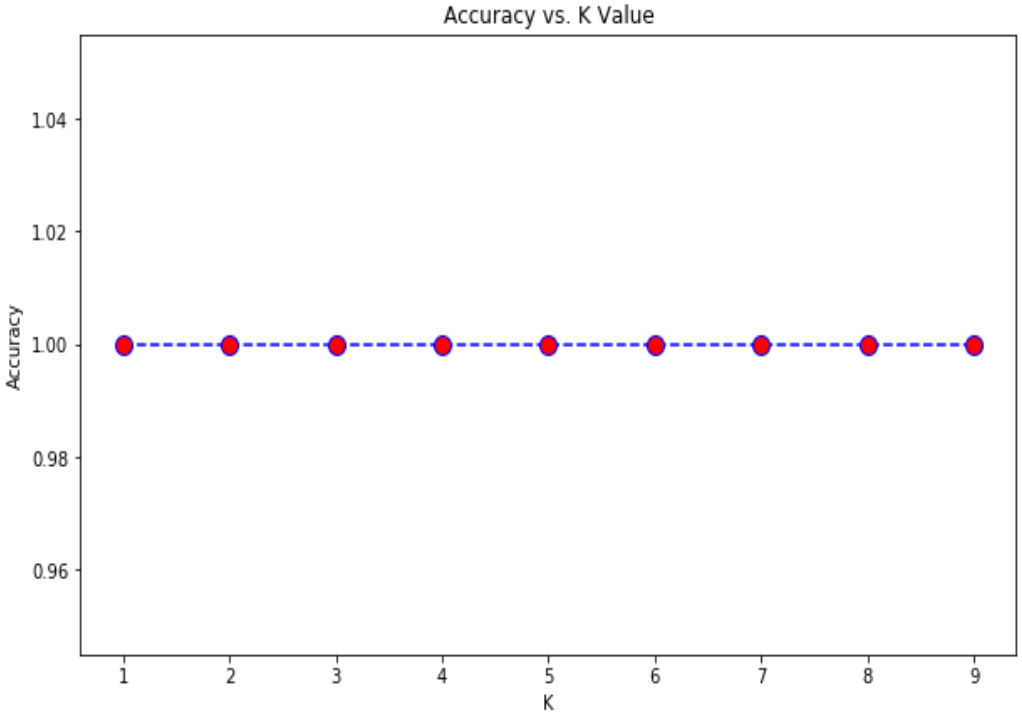

Figure 4.2: Accuracy vs k value with 20 features.

.

Figures 4.1 and 4.2 show accuracy vs k-value plot for the Four HCI Task dataset. This gives a better picture of the performance of this classifier at various k values as well as the progression of the accuracy performance metrics. As k increases the accuracy increases also for experiments without feature selection, but after feature selection the accuracy peaked at 100% for $k = 1\ to\ 10$. Experimental results showing accuracy and F1-scores are represented in bar plot format in Figure 4.3. These experiments were conducted in three phases. In Phase 1, all 218 features were used in the experiments, while the features were reduced to 20 and 10 for Phases 2 and 3 respectively.

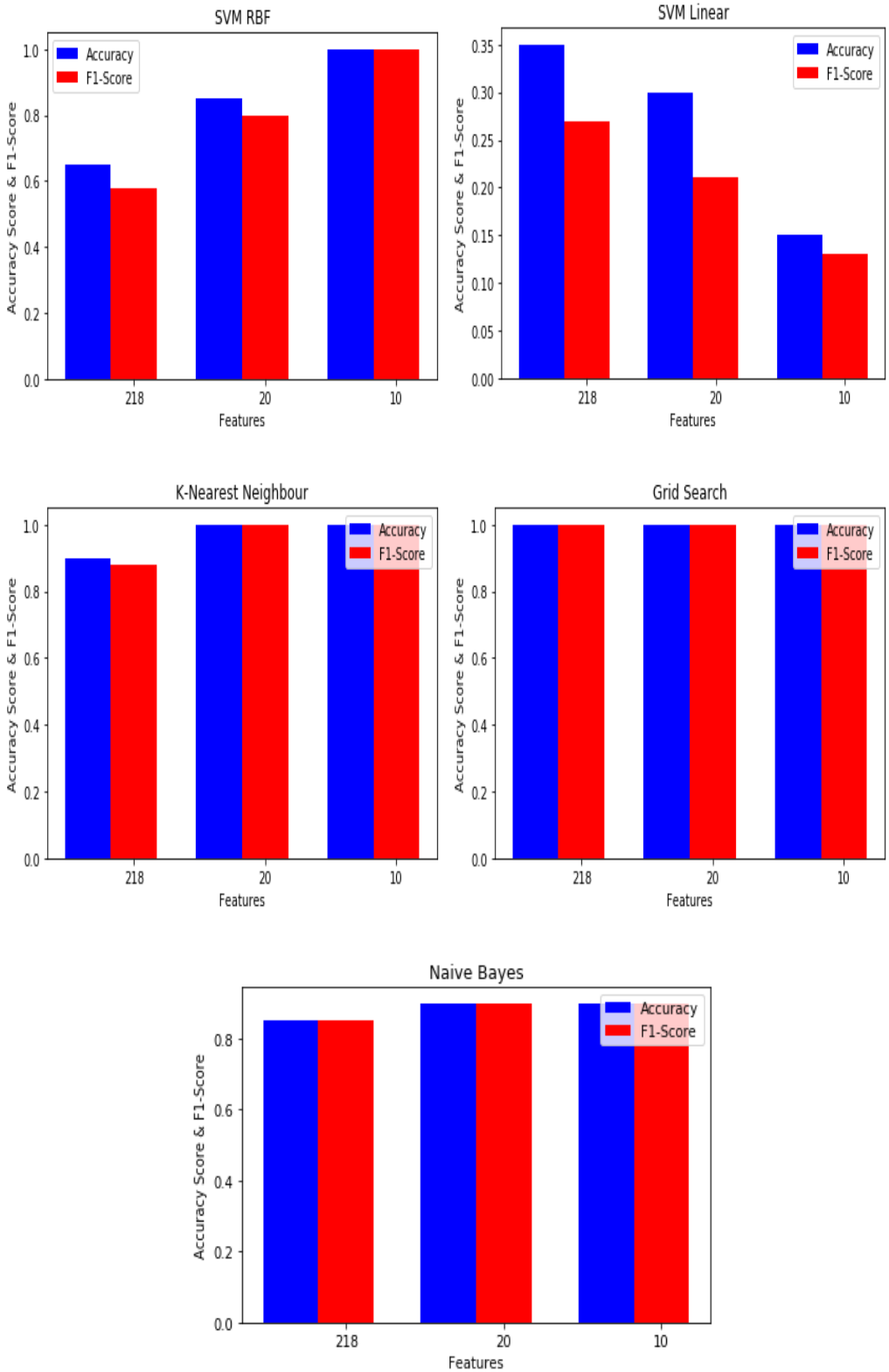

Figure 4.3: Bar plots of accuracy and F1-scores for the four-HCI task dataset.

From Figure 4.3, it is seen that linear SVM performs poorly with this dataset; it is also observed that as the features are reduced its performance also declined. This is because the data points of this dataset are clustered together and thus are not linearly separable. Hence, RBF performed better than its linear counterpart with other classifiers performing well because of the non-linear nature of the dataset. Figure 4.4 shows the visualization of the dataset and how the data points are clustered.

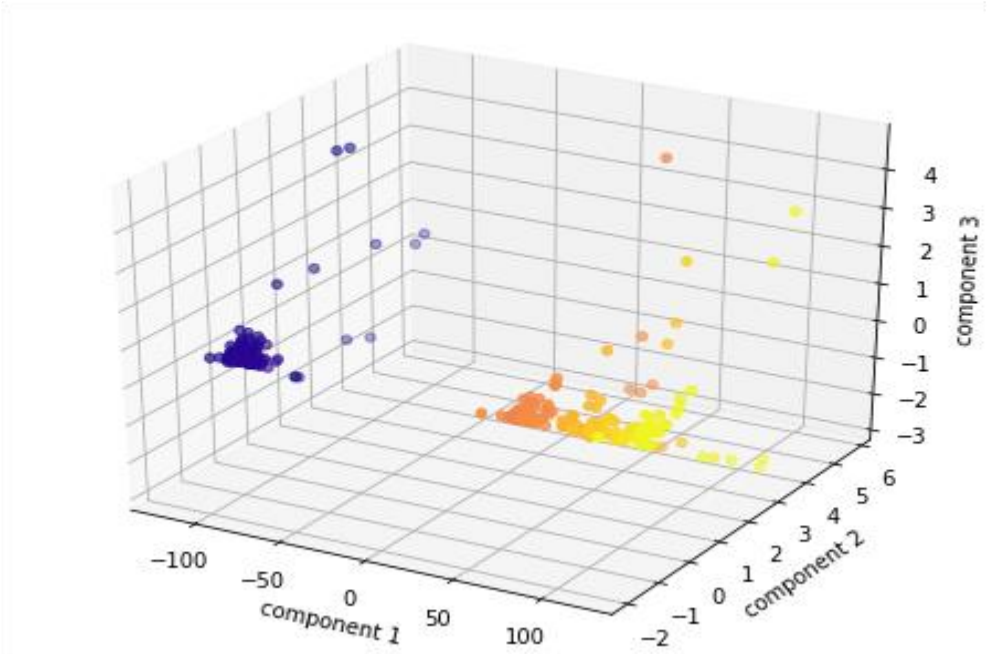

Figure 4.4: 3D scatter plot for the Four HCI task dataset.

Figure 4.4 shows the dataset projected onto a 3D space using principal component analysis for visualization. The colors blue, orange and yellow are clearly noticed in this plot. Different tones of these colors represent the classes found. The plot shows the distribution of our data points, and in relation

to classification due to how sparse it is distributed, it suggests that some classifiers will be more suitable for it than others.

Before the preprocessing phase of our experiments, we had a total of 57 users or classes, these classes varied in extremes with some classes having as little as one sample. Our preprocessing phase involved removing classes with only one keystroke sample, after which the samples were down to 43 users or classes. The reason for this cleaning process is to avoid a class imbalance problem when the classification task was performed.

After removing users with only one sample, users with less than 10 samples were screened out because we performed a 10-fold cross validation, and hence each class must be 10 or more samples

The first stage of our experiments was performed without feature selection using the k-NN algorithm; the results are shown in Table 4.1. The rate of classification accuracy of a user increases as the value of k increase, notably at $k = 4$. An accuracy of 100% means a high rate of accurately classifying a user. Furthermore, SVM, Naive Bayes, and gird search algorithms were applied to the dataset with high accuracy and F1-scores obtained. F1-score reached its best and worst values at 1 and 0 respectively, the closer to 1 the better. The F1-score performance metrics was used to tackle the class imbalance problem.

In the next stage of the experiments we performed feature selection on the dataset used in Phase 1 of our experiments. The first 20 features are shown in Figure 3.10, while the results from this experiment is shown in Figure 4.3,

with optimal results achieved when $k = 1\ to\ 10$. Lastly, features were further reduced to 10 with the result shown in Figure 4.3 and Table 4.1.

In comparison with the results obtained from the study of [4,5], our method used in this thesis incorporates both feature selection and stratified *m*-fold cross validation. Table 4.1 shows the complete results obtained from our experiment using this dataset, along with the parameters used for the classification algorithms. In our experiments, we have used 10-folds for stratified *m*-fold cross validation. We set our grid search parameters within this range: for cost (*C*= 0.1, 1, 10, 100, 1000), for gamma (Gamma= 1, 0.1 ,0.01, 0.001, 0.0001), and for kernels (RBF, linear, poly), evaluations were done based on these parameters, and for *k*-NN, *k* value was set at $k = 10$.

Experi-mental-Phase No.	No of Fea-tures	Classifier	No of Users	Accuracy (%)	F1-Score
1	218	*k*-NN	16	*K* (1to2) = 95.0 *K* (3) = 90.0 *K* (4) = 95.0 *K* (5to10) = 100	0.88
1	218	SVM	16	RBF = 65 Linear = 35	0.58 0.27
1	218	Grid search	16	(*C;* 0.1, degree; 3, gamma; 1, Poly) = 100	1.00
1	218	Naive Bayes	16	85	0.85
2	20	*k*-NN	16	*K* (1to10) = 100	1.00
2	20	SVM	16	RBF = 85 Linear = 30	0.80 0.21
2	20	Grid search	16	(*C;* 0.1, degree; 3, gamma; 1, Poly) = 100	1.00
2	20	Naive Bayes	16	90	0.90
3	10	*k*-NN	16	*K*=1to10; 100	1.00
3	10	SVM	16	RBF = 100 Linear = 15	1.00 0.13
3	10	Grid search	16	(*C;* 0.1, degree; 3, gamma; 1, Poly) = 100	1.00
3	10	Naive Bayes	16	90	0.90

Table 4.1: Results from experiments.

Table 4.1 shows the complete results of this experiment, as well as the parameters used for classification. The experimental phases correspond to experiments done without feature selection, 20 features and 10 features. The *k*-NN and grid search algorithms performed exceptionally well from the

result obtained. While we used accuracy and F1-score as performance measures we have also used ROC curve for measuring the performance of the SVM algorithm. Figure 4.5 shows the ROC curve derivation when 20 features were used.

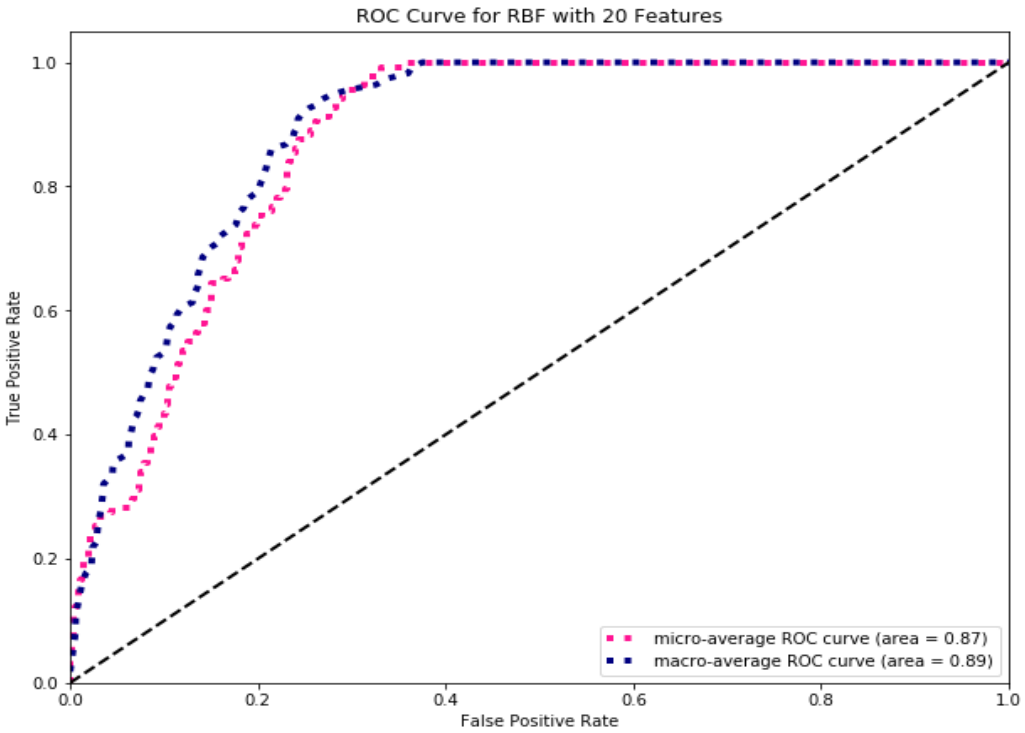

Figure 4.5: ROC curve RBF with 20 features.

Micro average calculates the true positive rates and false positive rates for each class and derives the AUC. Macro average sums up the individual AUCs to obtain the average. Figure 4.6 shows the ROC curve for each user, all well above the threshold. Index 0 represents the first user, and so on up until the last user making a total of 16 users.

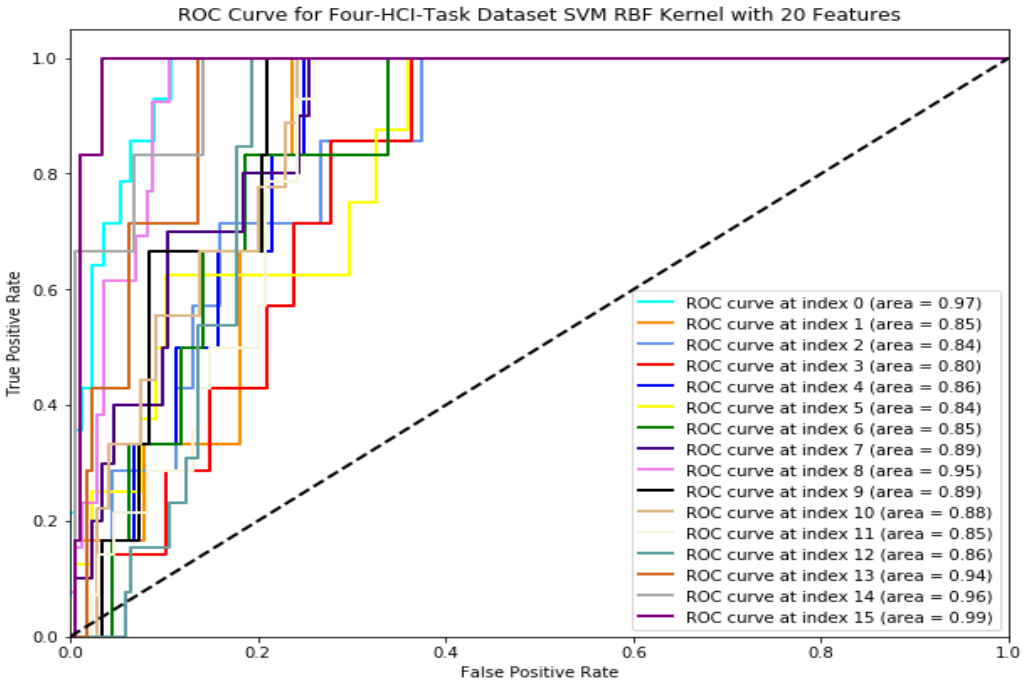

Figure 4.6: ROC curve derivation for each user with 20 features.

Figure 4.7 shows a box plot of AUC. The accuracy is used to maximizes or optimizes the authentication of each user, the highest accuracy was 99% and lowest at 84%, and the standard deviation from all users at 5.60803.

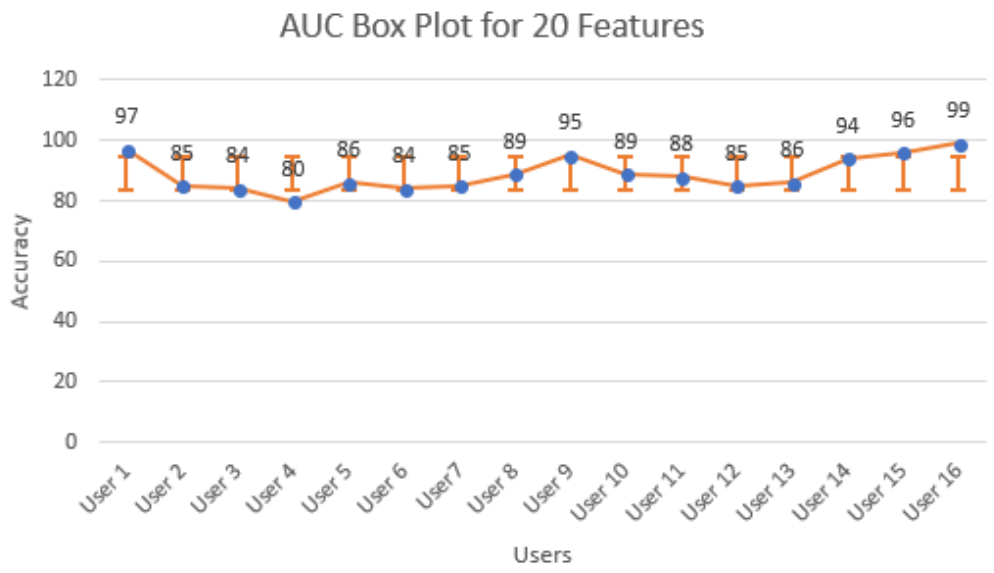

Figure 4.7: Box plot of AUC for 20 features.

4.2 Half Keystroke Sequence Experiment

The idea behind reducing the keystroke sequence is to provide a scenario where the authentication process takes less time before identifying an intruder. With more keystroke samples the efficiency of timely identifying an intruder may prove costly to the system. In this experiment, we have taken the following steps in reducing the keystroke sequence, to carry out the classification task on the new dataset. These are:

- For users with 500 keystrokes, we retained 90% of the sequence.
- For users with 1000 or more keystrokes, we retained half of the keystroke sequence.
- In the next step, we ran the feature extraction algorithm on the newly obtained keystroke data.
- Obtained a new keystroke dataset.
- Applied our method to the new dataset.

After extracting the features, we had a total of 16 users with at least three samples per class. Table 4.3 shows the results of this experiment. We have used 3-fold for stratified m-fold cross validation as well as set our grid search parameters within the range, for cost (C= 0.1, 1, 10, 100, 1000), for gamma (Gamma= 1, 0.1 ,0.01, 0.001, 0.0001), and for kernels (RBF, linear, poly), evaluations were done based on these parameters, and for k-NN, k was set at $k = 4$. We have also carried this experiment in three phases, Phase 1 without feature selection (218 features), Phases 2 and 3 with 20 and 10 features respectively.

Experimental-Phase No.	No of Features	Classifier	No of Users	Accuracy (%)	F1-Score
1	218	*k*-NN	16	*K* (1to4) = 96.55	0.97
1	218	SVM	16	RBF = 75.86 Linear = 24.13	0.70 0.17
1	218	Gridsearch	16	(*C* ; 0.1, degree ; 3, gamma ; 1, Poly) = 100	1.00
1	218	Naive Bayes	16	34.48	0.36
2	20	*k*-NN	16	*K* (1to4) = 96.55	0.97
2	20	SVM	16	RBF = 79.31 Linear = 27.58	0.74 0.23
2	20	Gridsearch	16	(*C* ; 0.1, degree ; 3, gamma ; 1, Poly) = 100	1.00
2	20	Naive Bayes	16	72.41	0.69
3	10	*k*-NN	16	*K* (1to4) = 100	1.00
3	10	SVM	16	RBF = 89.65 Linear = 17.24	0.86 0.16
3	10	Gridsearch	16	(*C* ; 0.1, degree ; 3, gamma ; 1, Poly) = 100	1.00
3	10	Naive Bayes	16	79.31	0.76

Table 4.2: Half-sequence experiment results.

Grid search and *k*-NN algorithms were the top performers in this experiment. As in the four HCI task experiment, the linear SVM performed poorly too due to the nature of the dataset, and because it is not linearly separable. Figure 4.8 shows the scatter plot for the dataset used in this experiment. This plot was done using principal component analysis the colors blue, orange and yellow are clearly noticed in this plot. Different tones of these

colors represent the classes found. The plot shows the distribution of our data points

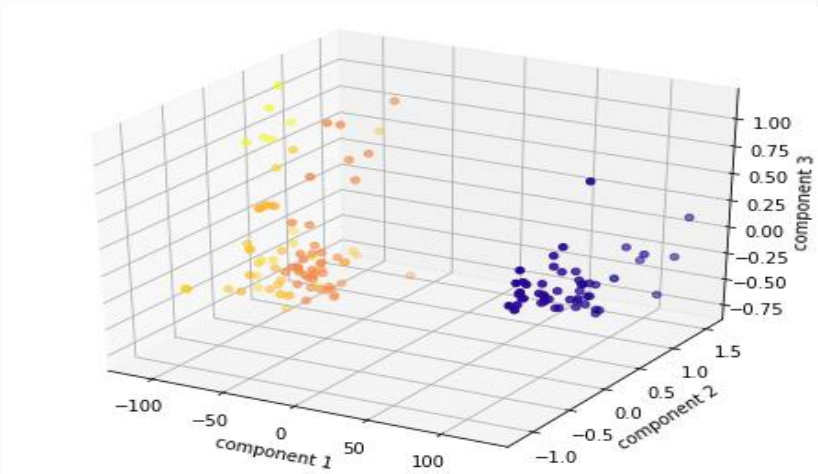

Figure 4.8: 3D scatter for plot half sequence experiment.

Figure 4.9 shows the ROC AUC for the RBF kernel when 20 features were used. It shows improvement in performance with reduction of the feature vectors.

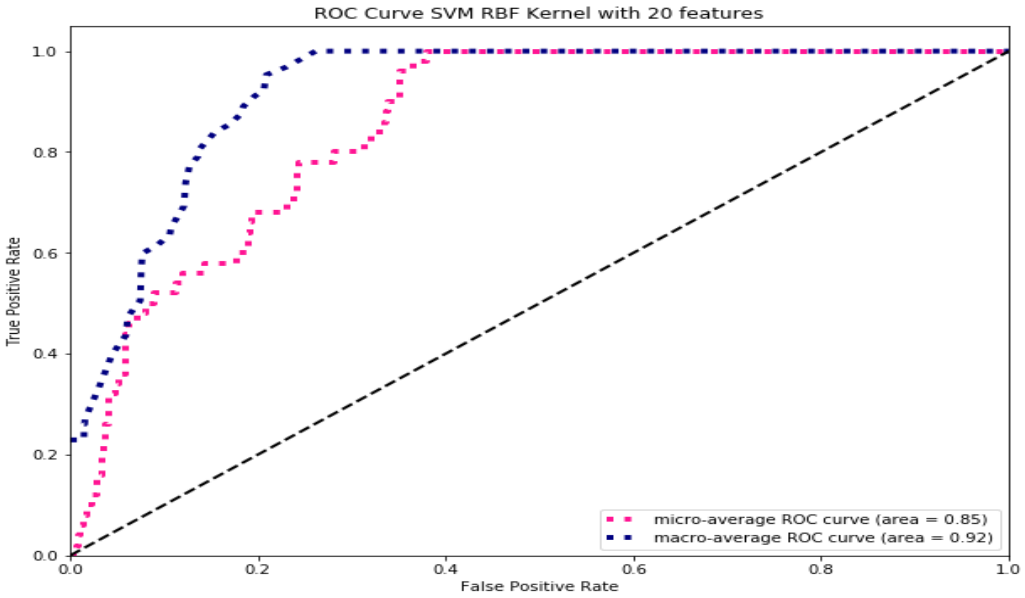

Figure 4.9: ROC curve derivation RBF with 20 features.

4.3 Experimental Results from The Villani Keystroke Dataset

Our method was also tested on the Villani keystroke dataset; the results obtained are shown in Table 4.3. As in our previous experiments we have used the same algorithms in this experiment. The experiment was divided into 3 phases. Phase 1 without feature selection, Phase 2 with 50 features, and Phase 3 with 20 features. We also reduced to 10 feature vectors but there was reduction in performance of the classification algorithms. Parameters for grid search remained the same, while we chose k values from 1 to 25. We also set the threshold for minimum samples per class at 10.

Experimental-Phase No.	No of Features	Classifier	No of Users	Accuracy (%)	F1-Score
1	218	k-NN	143	98.93	1.00
1	218	SVM	143	97.87	0.92
1	218	Grid search	143	100.00	1.00
1	218	Naive Bayes	143	92.55	0.91
2	50	k-NN	143	98.93	0.99
2	50	SVM	143	98.93	0.99
2	50	Grid search	143	100.00	1.00
2	50	Naive Bayes	143	84.04	0.80
3	20	k-NN	143	98.93	0.99
3	20	SVM	143	97.87	0.97
3	20	Grid search	143	100.00	1.00
3	20	Naive Bayes	143	70.21	0.64

Table 4.3: Results from Villani keystroke dataset experiments.

It can be observed that the performance metrics decreased when we reduced the number of features below 50; this lead to a reduction in accuracy and F1-score. ROC AUC increased as shown in Figure 4.10 with SVM RBF. Grid search optimization obtained the best performance in this experiment.

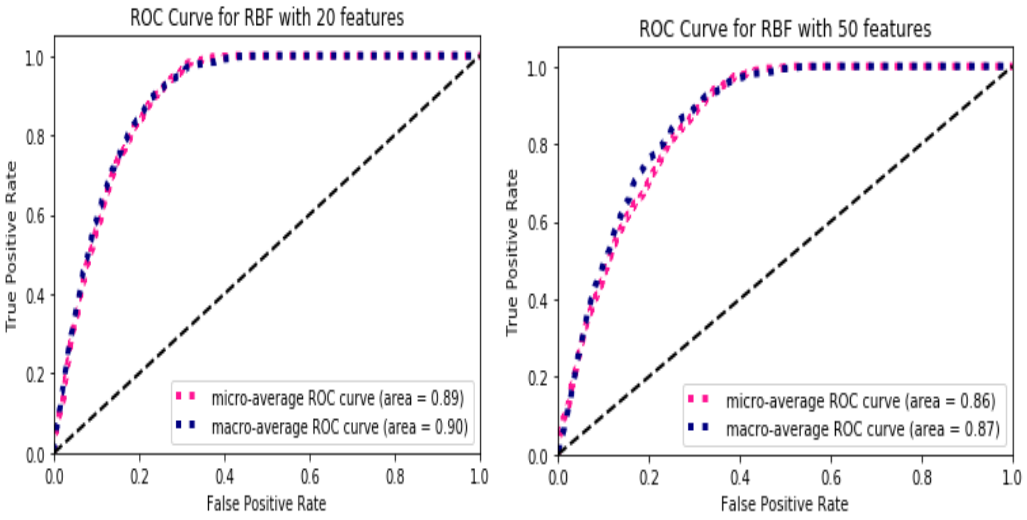

Figure 4.10: ROC curve derivation for SVM RBF with 20 and 50 features.

4.4 Conclusion

Our method incorporates stratified m-fold cross validation, and one vs all approach for multiclass classification. We also applied the same method to the Villani keystroke dataset and the experiments further shows the importance of our method. The results obtained from our experiments show the importance of selecting relevant features for classifying users. This is important because a security system should be able to detect treats accurately and within a short time frame.

CHAPTER 5

Conclusion and Future Work

In this thesis, we proposed a system for user authentication that reduces the computational time of authenticating each user by selecting relevant features. Our approach used mRMR feature selection for selecting relevant features. This algorithm selects a feature set by choosing the features that have maximum relevance and removes redundant features by using mutual information to analyze relevance and redundancy. This method improved the performance metrics of the classification algorithms. Our method also included stratified m-fold cross validation. The best performing classifier was the k-nearest neighbour with 100% accuracy recorded in some experiments. The results have not only shown the efficiency of our method but also show that the proposed method can be applied to other datasets to produce optimal results, and with selecting relevant features while reducing the computational time and still maintaining or improving the system.

5.1 Contributions

The main contributions of this thesis can be summarized as follows:

- The user authentication system provides a method which reduces the number of feature vectors without affecting the performance of the system.
- Determined the relevant features that improves the classification algorithm.

- Incorporated stratified m-fold cross validation, which solves the problem of class imbalance, because it equally distributes percentages of the classes or users into the available fold during cross validation.

5.2 Future Work

The performance achieved from applying our method indicates that our method can be implemented into a security system for authentication or verification of users. The goal is for this system to accurately authenticate a user within a short time frame when programmed. However, it would be a good idea if the system is able to learn and make intelligent decisions on its own without constantly being monitored; considering this provides areas that we explore in the future:

- Apply this method to other keystroke dynamic datasets.
- Try deep learning algorithms to see how well it performs, and how it makes intelligent decisions in authenticating users.
- Reduce the keystroke sequence to improve the authentication accuracy.

REFERENCES

[1] Jain, A. K., Bolle, R., & Pankanti, S. (Eds.). (2006). Biometrics: personal identification in networked society (Vol. 479). Springer Science & Business Media.

[2] Monrose, F., & Rubin, A. D. (2000). Keystroke dynamics as a biometric for authentication. Future Generation computer systems, 16(4), 351-359.

[3] Teh, P. S., Teoh, A. B. J., & Yue, S. (2013). A survey of keystroke dynamics biometrics. The Scientific World Journal, 2013.http://dx.doi.org/10.1155/2013/408280

[4] Monaco, J. V., Bakelman, N., Cha, S. H., & Tappert, C. C. (2012, August). Developing a keystroke biometric system for continual authentication of computer users. In Intelligence and Security Informatics Conference (EISIC), 2012 European (pp. 210-216). IEEE

[5] Monaco, J. V., Bakelman, N., Cha, S. H., & Tappert, C. C. (2013, August). Recent advances in the development of a long-text-input keystroke biometric authentication system for arbitrary text input. In Intelligence and Security Informatics Conference (EISIC), 2013 European (pp. 60-66). IEEE

[6] Bergadano, F., Gunetti, D., & Picardi, C. 2002. User authentication through keystroke dynamics. ACM Transactions on Information and System Security (TISSEC),5(4), 367-397.

[7] Araújo, L. C., Sucupira, L. H., Lizarraga, M. G., Ling, L. L., & Yabu-Uti, J. B. T. 2005. User authentication through typing biometrics features. IEEE Transactions on Signal Processing, 53(2), 851-855.

[8] Hu, J., Gingrich, D., & Sentosa, A. 2008. A k-nearest neighbour approach for user authentication through biometric keystroke dynamics. IEEE International Conference on Communications, 1556-1560.

[9] Killourhy, K. S., & Maxion, R. A. 2009. Comparing anomaly-detection algorithms for keystroke dynamics. IEEE/IFIP International Conference on Dependable Systems & Networks, 125-134.

[10] Rybnik, M., Panasiuk, P., & Saeed, K. 2009. User authentication with keystroke dynamics using fixed text. In Biometrics and Kansei Engineering. ICBAKE 2009. International Conference IEEE.70-75.

[11] Sluganović, I., Karlović, A., Bosilj, P., Šare, M., & Horvat, S. (2012, May). User authentication based on keystroke dynamics analysis. In MIPRO, 2012 Proceedings of the 35th International Convention.1719-1724.

[12] Alves, D. D., Cruz, G., & Vinhal, C. (2014, December). Authentication system using behavioral biometrics through keystroke dynamics. IEEE Symposium on Computational Intelligence in Biometrics and Identity Management (CIBIM).181-184.

[13] Roy, S., Roy, U., & Sinha, D. D. (2014, December). Free-text user authentication technique through Keystroke Dynamics. In High Performance Computing and Applications (ICHPCA), 2014 International Conference.1-6.

[14] Kaganov, V., Korolyov, A., Krylov, M., Mashechkin, I., & Petrovskiy, M. (2014, December). Hybrid method for active authentication using keystroke dynamics. In Hybrid Intelligent Systems (HIS), 2014 14th International Conference. 61-66.

[15] Anusas-amornkul, T., & Wangsuk, K. (2015, November). A comparison of keystroke dynamics techniques for user authentication. International Computer Science and Engineering Conference (ICSEC).1-5.

[16] Darabseh, A., & Namin, A. S. (2015, December). Effective User Authentications Using Keystroke Dynamics Based on Feature Selections. IEEE 14th International Conference on Machine Learning and Applications (ICMLA).307-312.

[17] Darabseh, A., & Namin, A. S. (2015, October). On Accuracy of Classification-based Keystroke Dynamics for Continuous User Authentication. International Conference on Cyberworlds (CW).321-324.

[18] Mondal, S., & Bours, P. (2016, February). Combining keystroke and mouse dynamics for continuous user authentication and identification.

IEEE International Conference on Identity, Security and Behavior Analysis (ISBA).1-8.

[19] Alsultan, A., & Warwick, K. (2013, October). User-friendly free-text keystroke dynamics authentication for practical applications. IEEE International Conference on Systems, Man, and Cybernetics.4658-4663.

[20] Tappert, C. C., Villani, M., & Cha, S. (2009). Keystroke biometric identification and authentication on long-text input. Behavioral biometrics for human identification: Intelligent applications, 342-367

[21] Radovic, M., Ghalwash, M., Filipovic, N., & Obradovic, Z. (2017). Minimum redundancy maximum relevance feature selection approach for temporal gene expression data. BMC bioinformatics, 18(1), https://doi.org/10.1186/s12859-016-1423-9

[22] Raschka, S. (2015). Python machine learning. Packt Publishing Ltd.

[23] Géron, A. (2017). Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems. " O'Reilly Media, Inc.".

[24] Han, J., Pei, J., & Kamber, M. (2011). Data mining: concepts and techniques. Elsevier.

APPENDIX A

The figures in this section show other visualized results for SVM and *k*-NN algorithms.

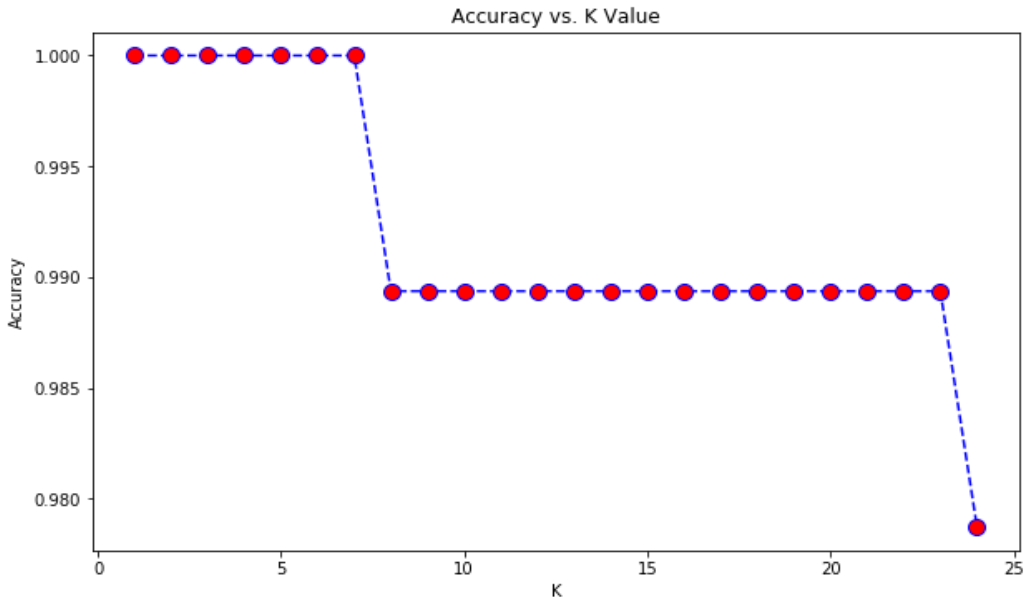

Figure A.1: Accuracy vs k value Villani keystroke dataset with 218 features.

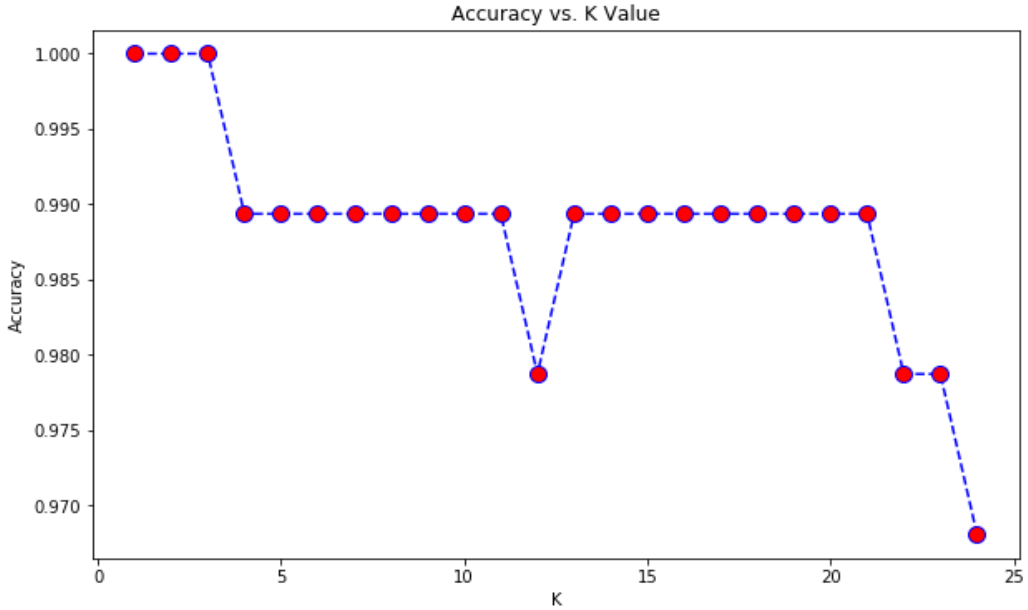

Figure A.2: Accuracy vs k value Villani keystroke dataset with 50 features.

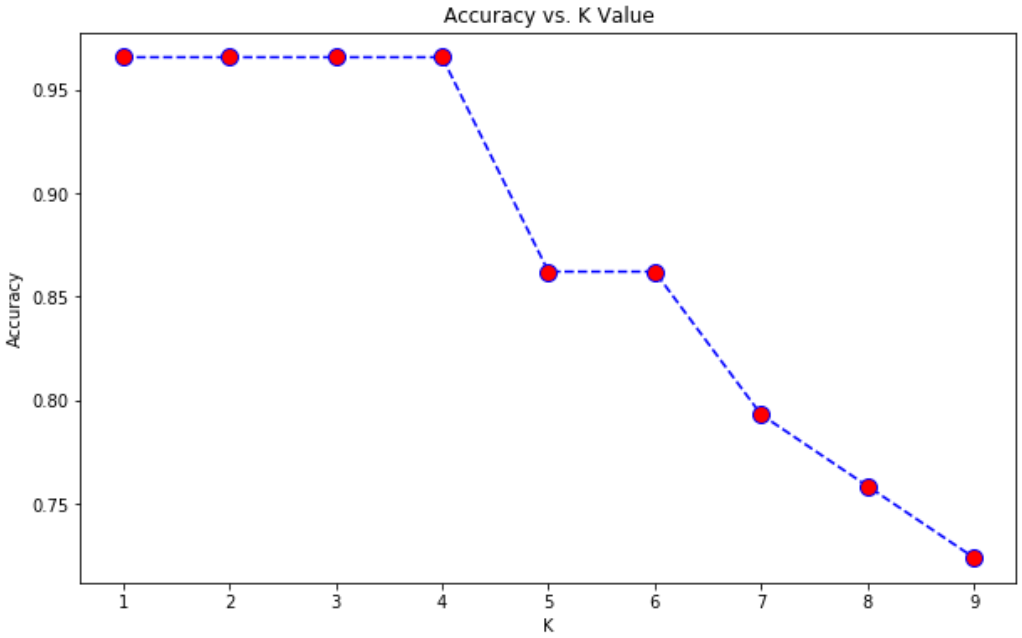

Figure A.3: Accuracy vs k value half sequence experiment with 218 features.

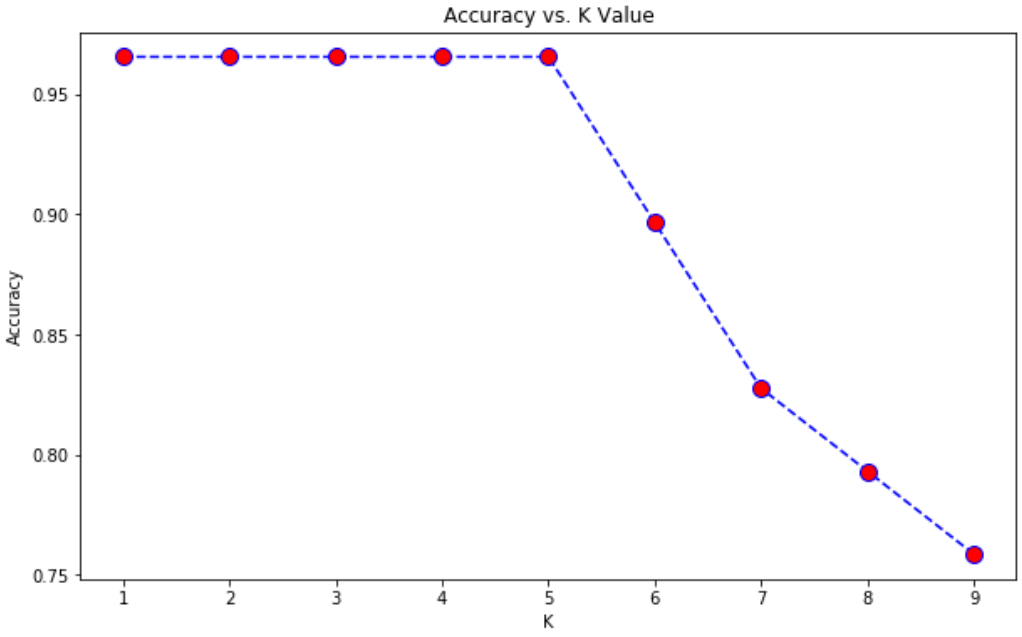

Figure A.4: Accuracy vs k value half sequence experiment with 20 features.

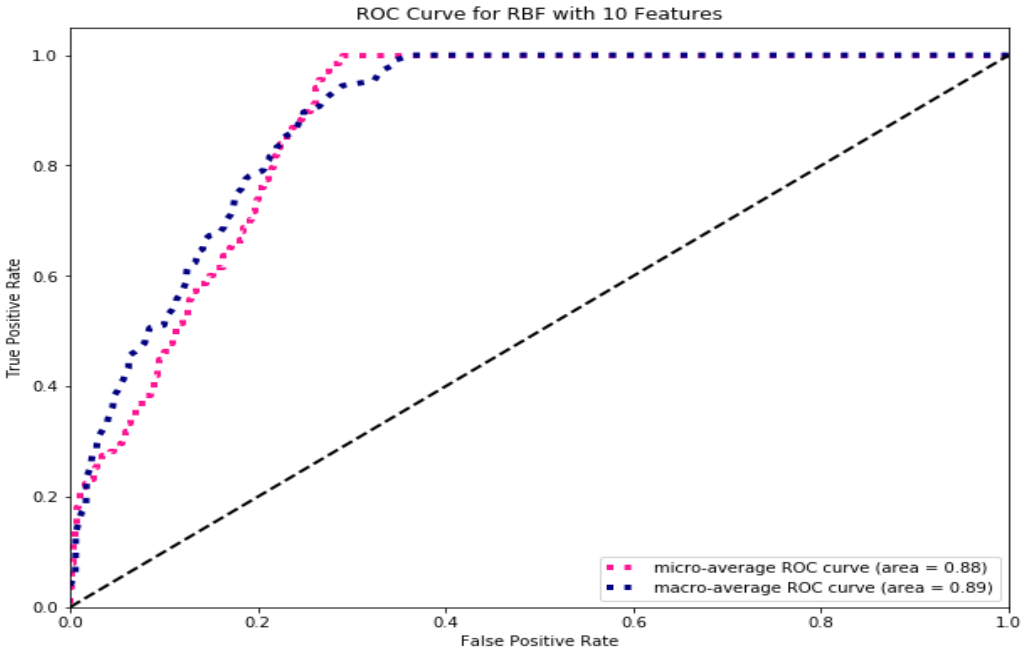

Figure A.5: ROC curve for SVM RBF with 10 features for four HCI dataset.

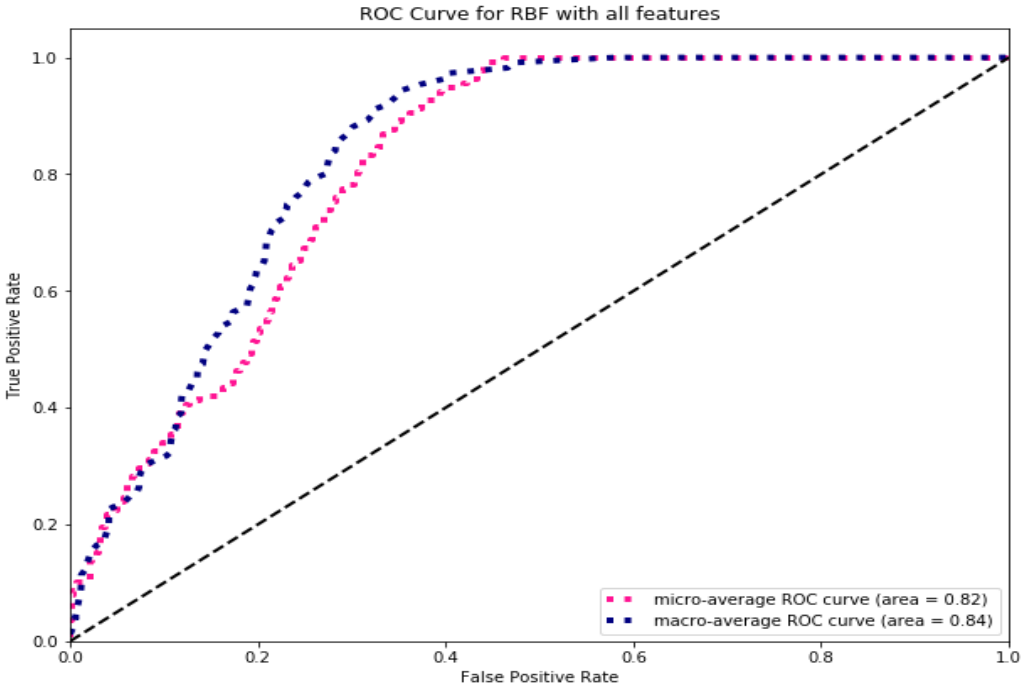

Figure A.6: ROC curve for SVM RBF with all features for four HCI dataset.

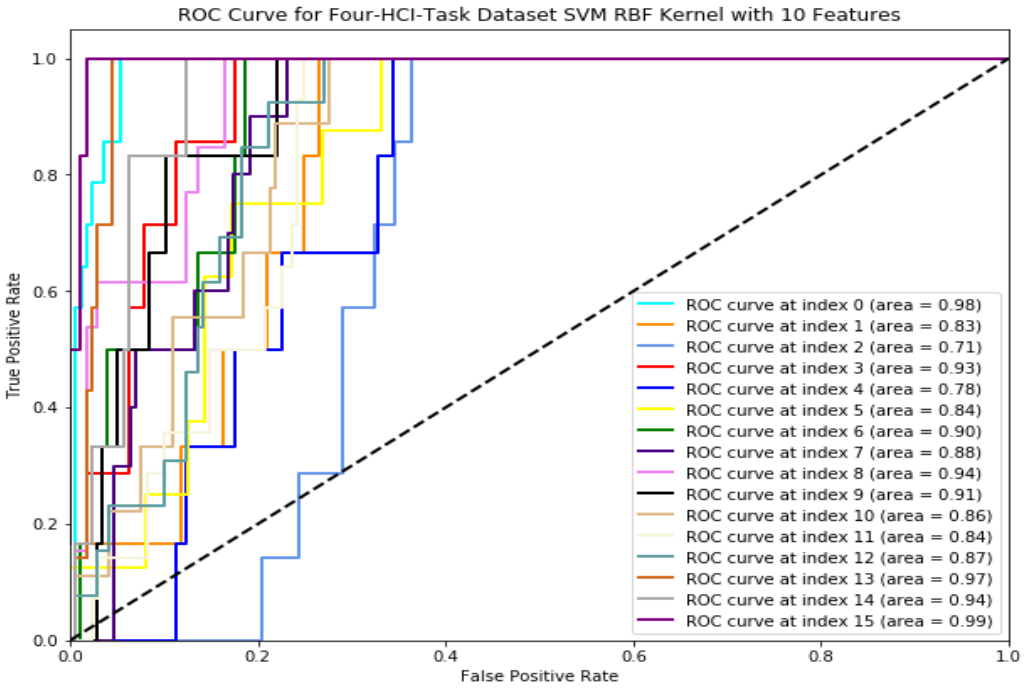

Figure A.7: ROC curve for each user with 10 features.

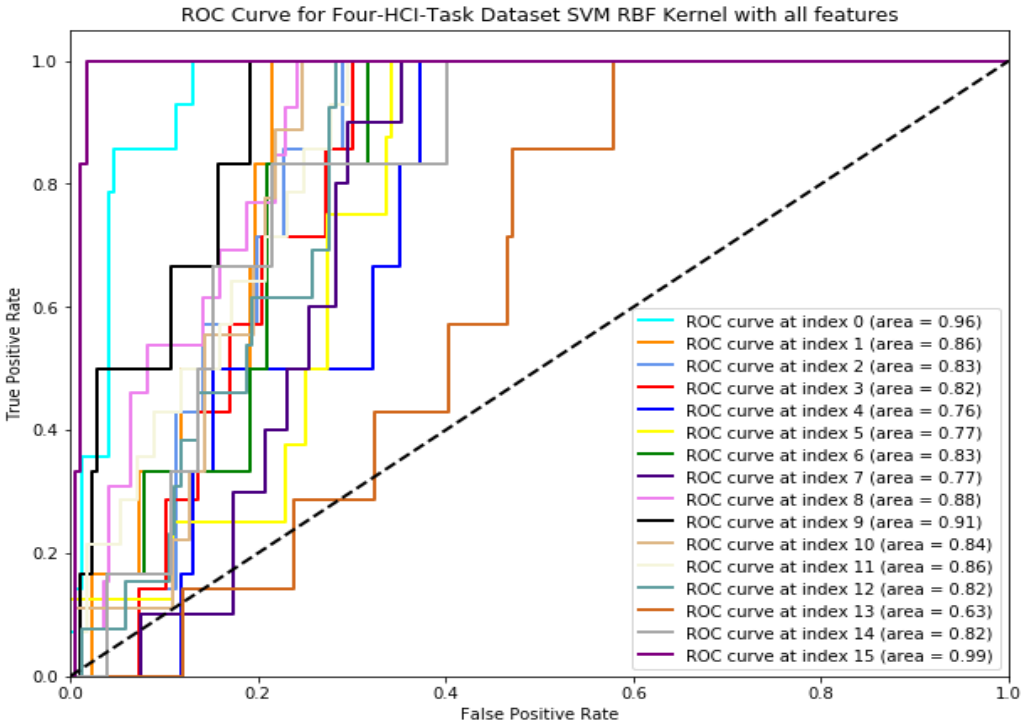

Figure A.8: ROC curve for each user with all features.

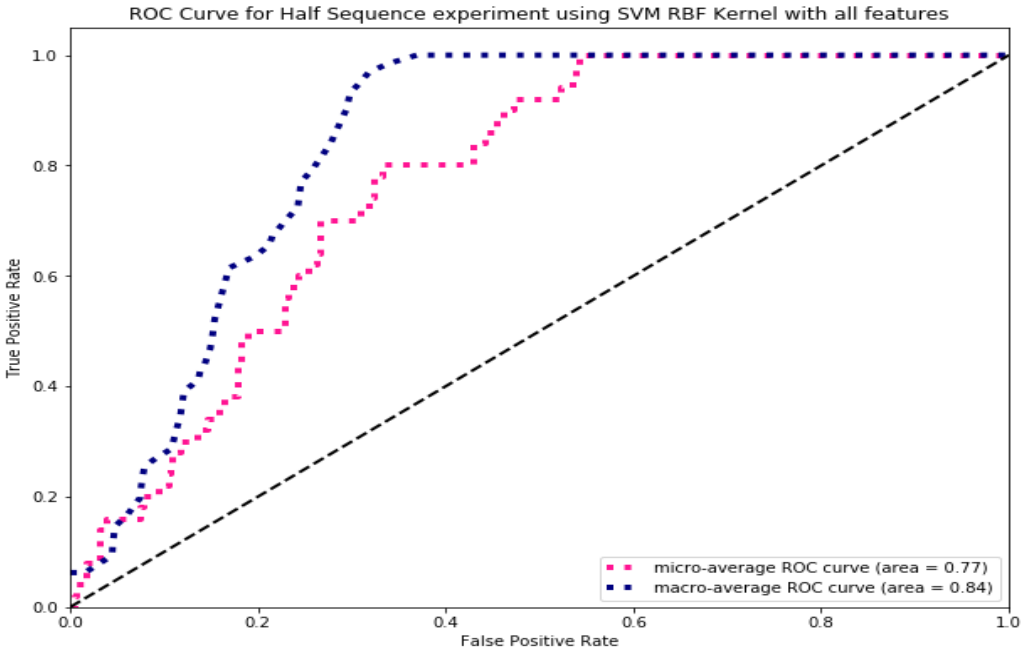

Figure A.9: ROC curve for SVM RBF with all features for half sequence experiment.

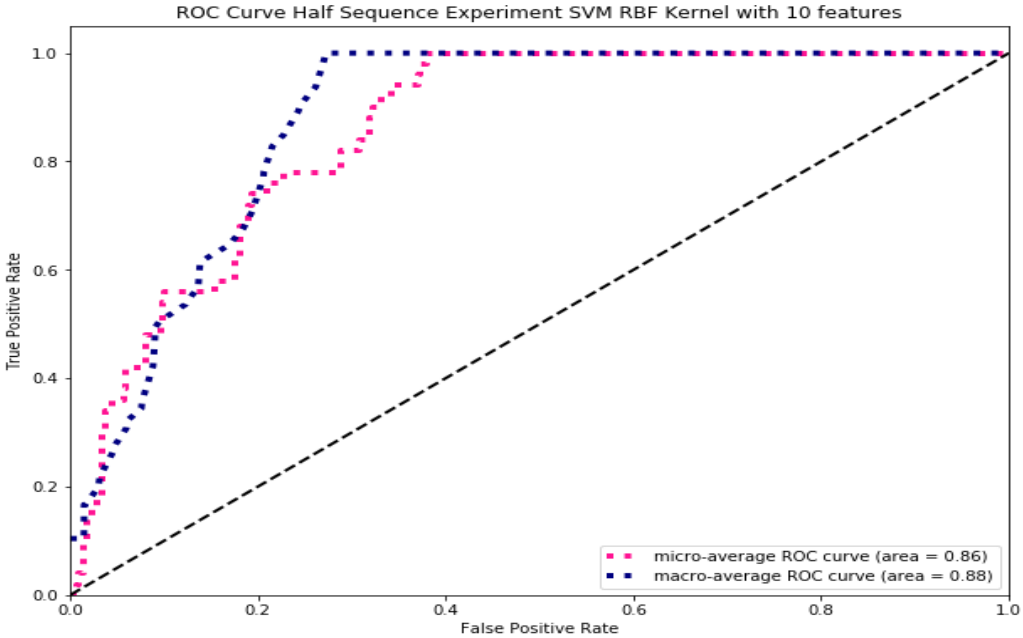

Figure A.10: ROC curve for SVM RBF with 10 features for half sequence experiment.

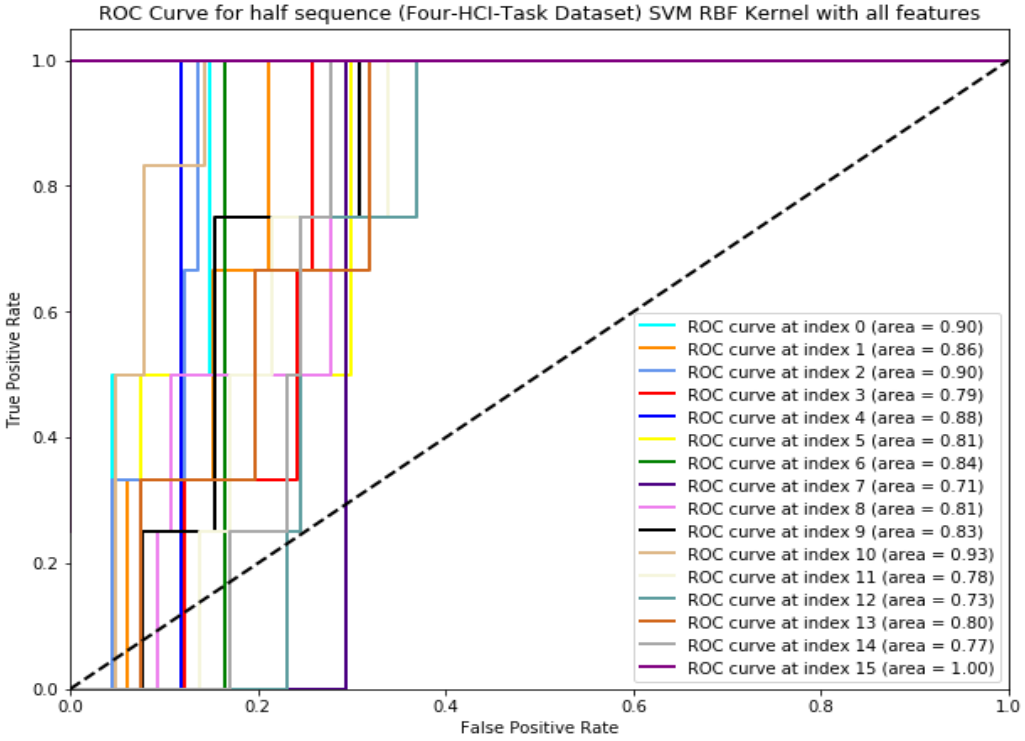

Figure A.11: ROC curve for each user with all features.

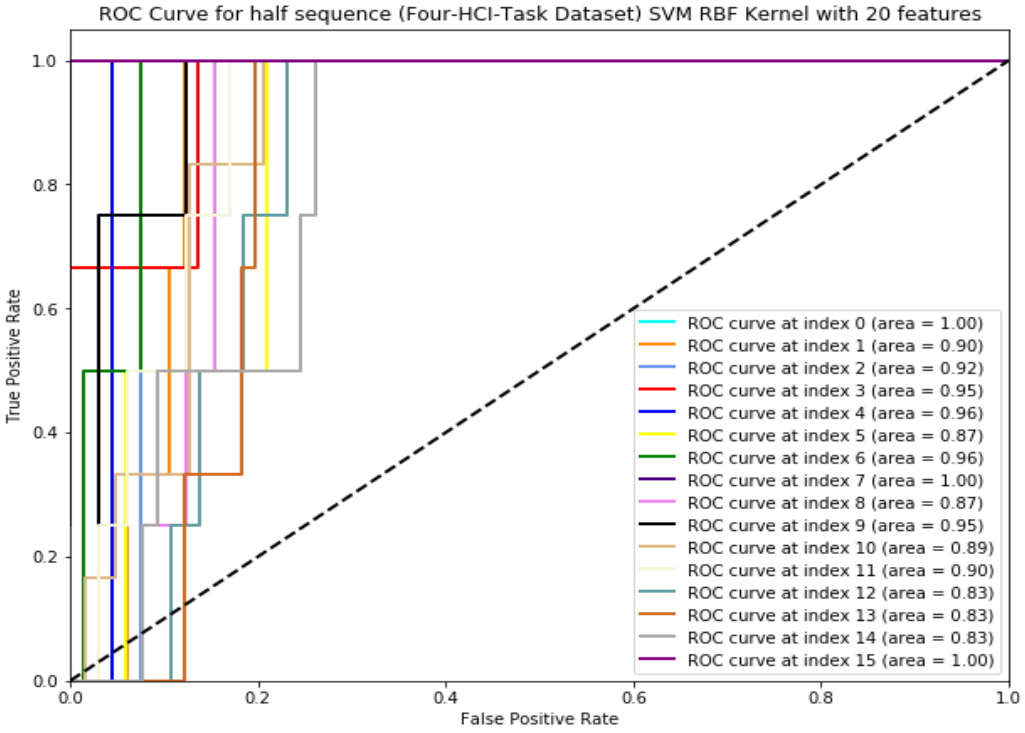

Figure A.12: ROC curve for each user with 20 features.

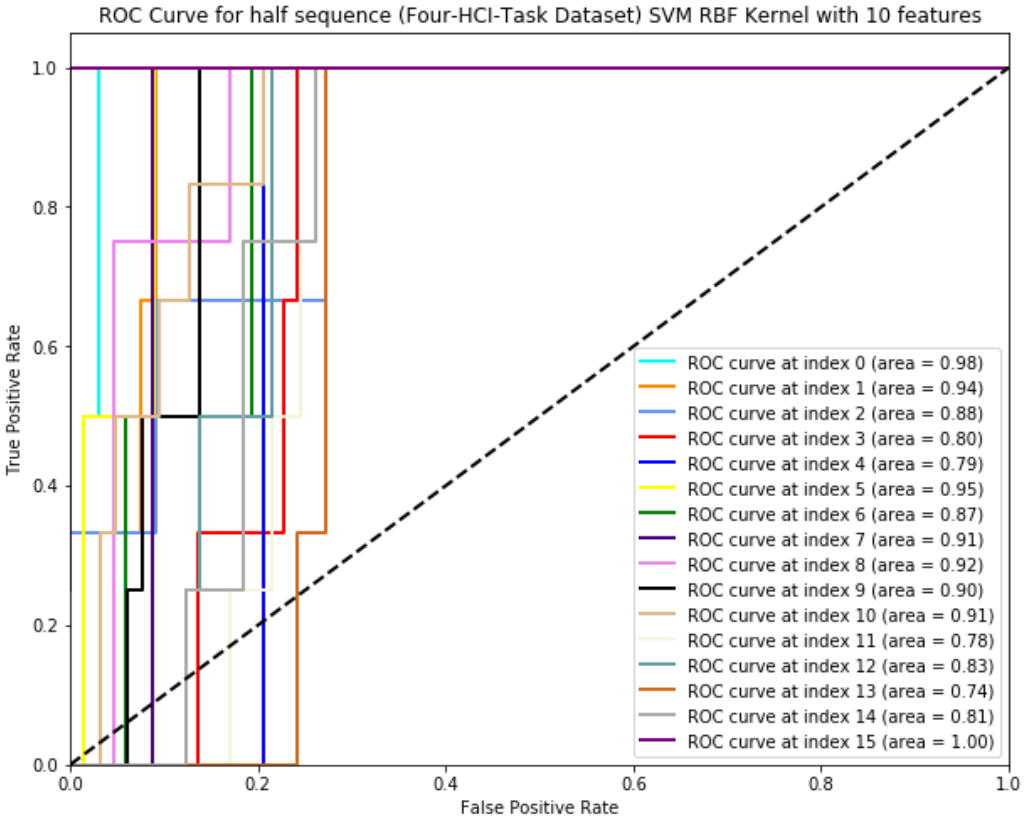

Figure A.13: ROC curve for each user with 10 features.

APPENDIX B

The figures in this section show the features selected using mRMR feature selection and 3D scatter plot before and after feature selection.

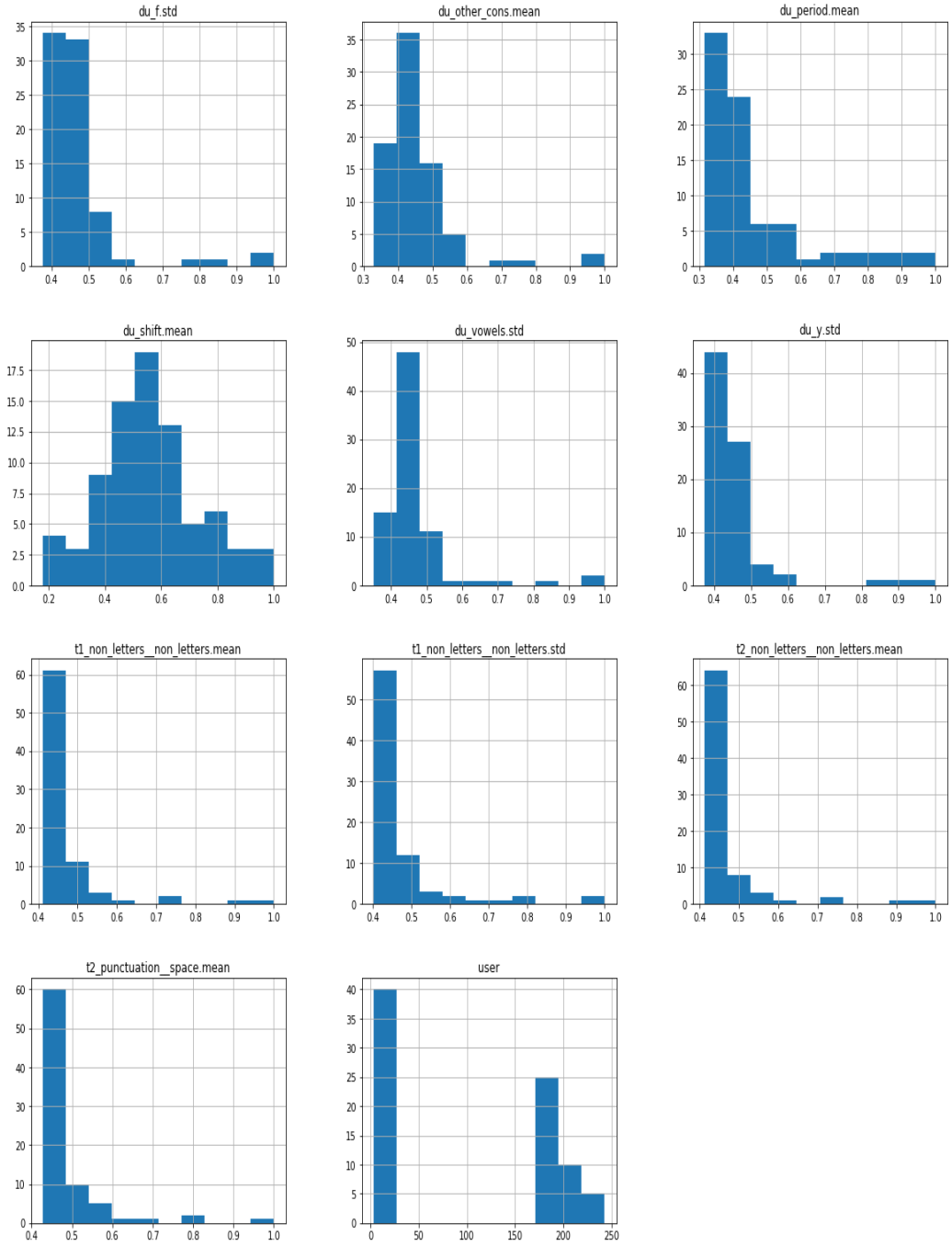

Figure B.1: 20 Features selected from the four HCI task dataset.

Figure B.2: 50 Features selected from the Villani keystroke dataset.

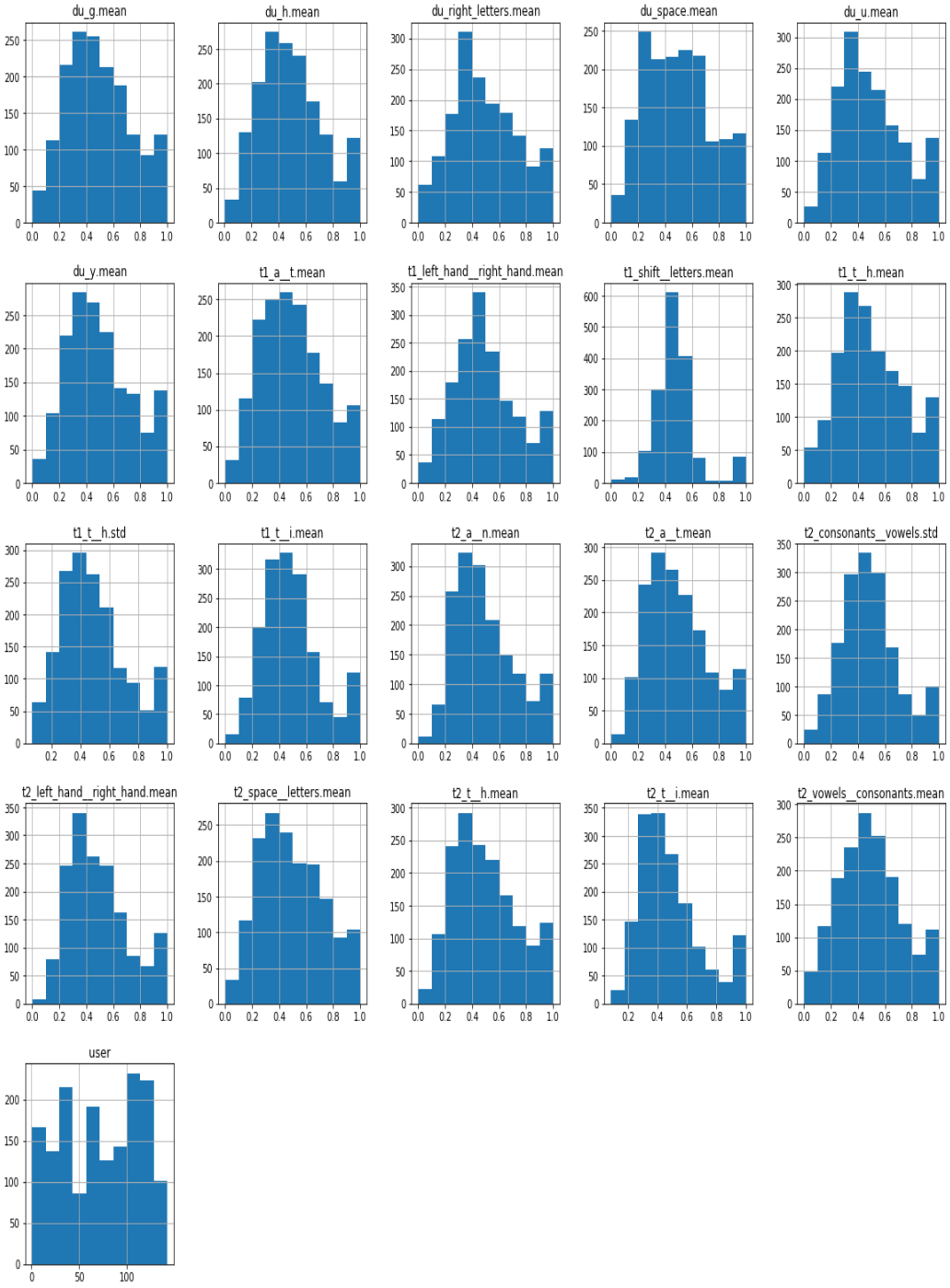

Figure B.3: 20 Features selected from the Villani keystroke dataset.

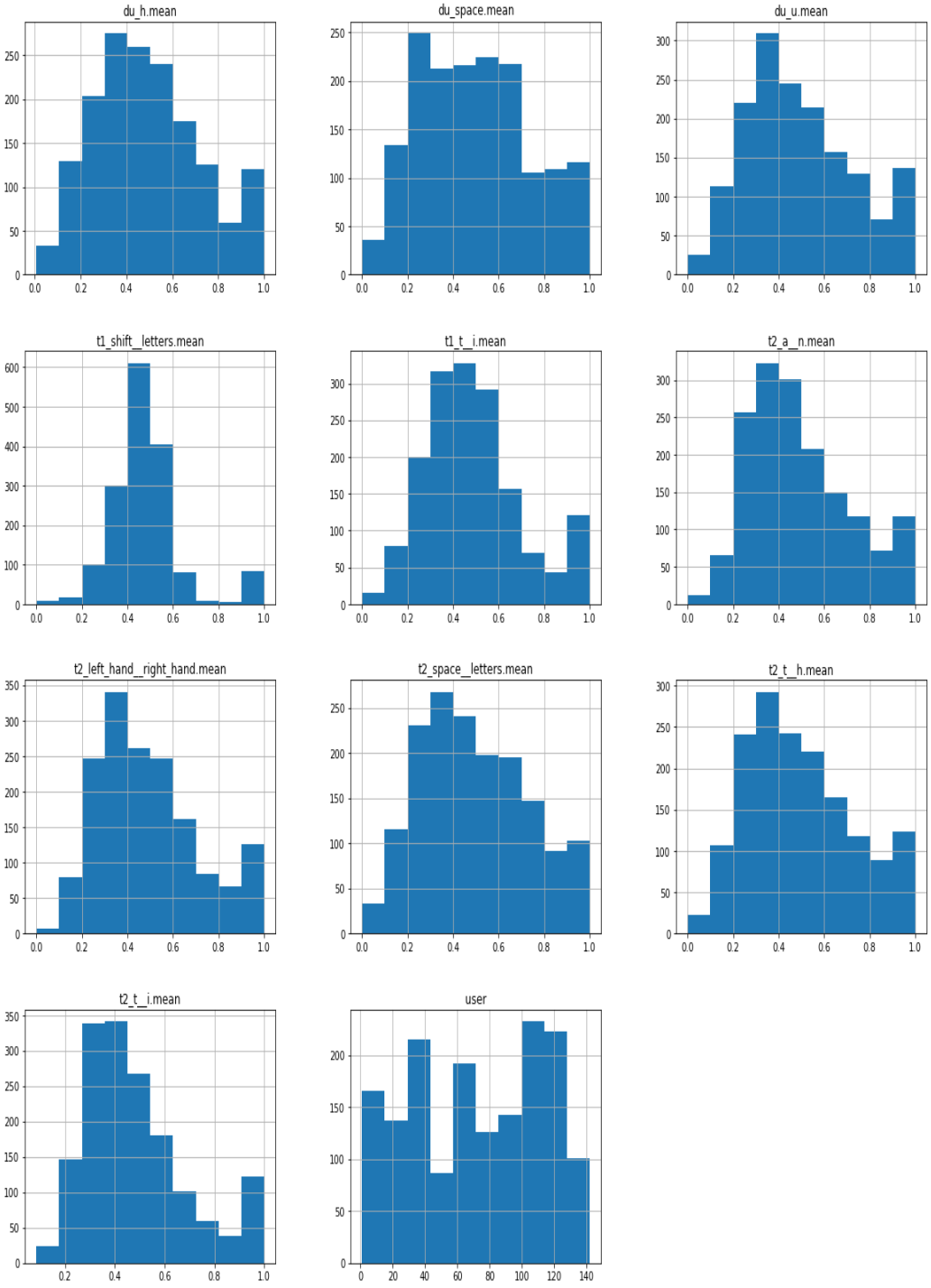

Figure B.4: 10 Features selected from the Villani keystroke dataset.

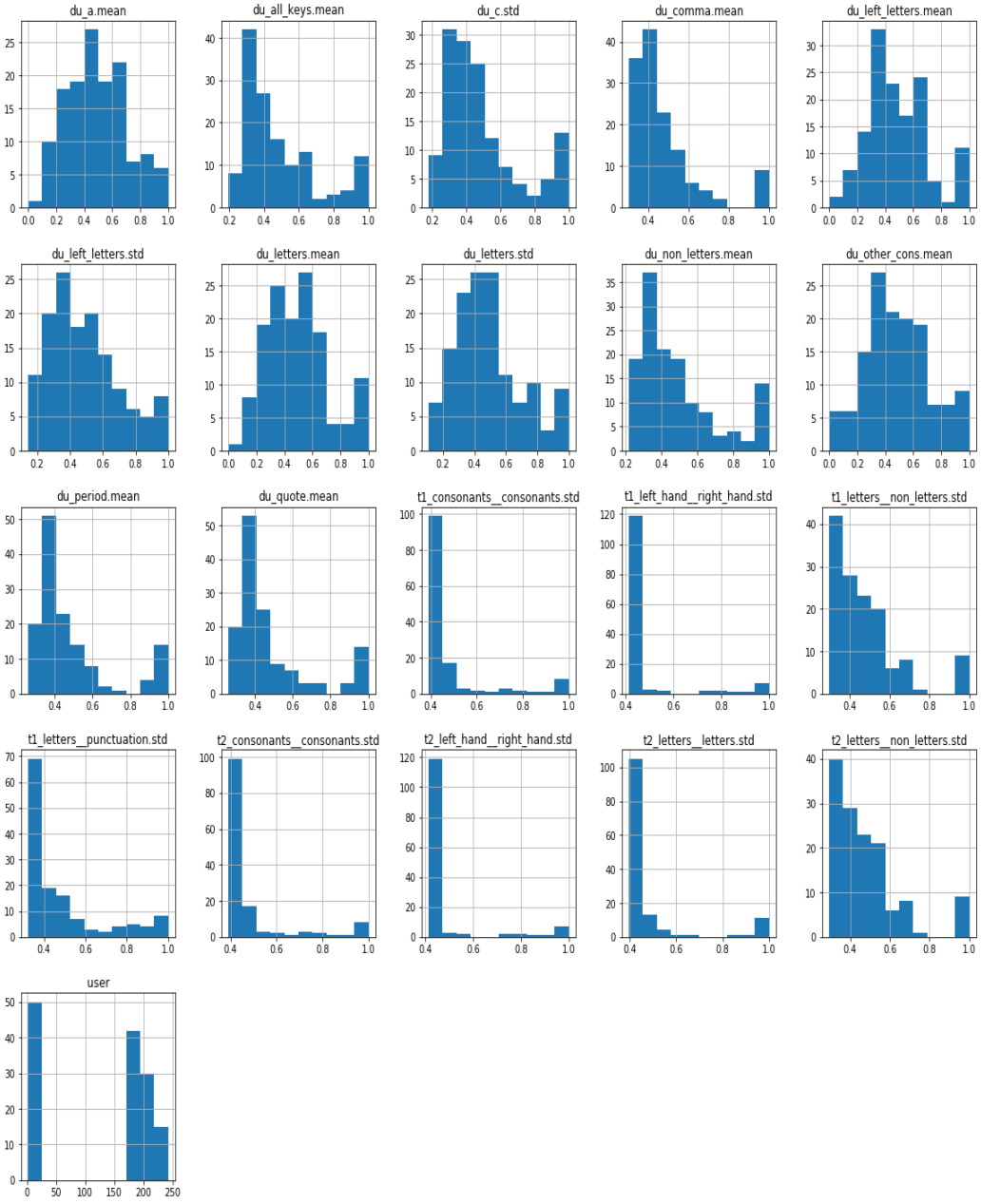

Figure B.5: 20 Features selected for the half sequence experiment.

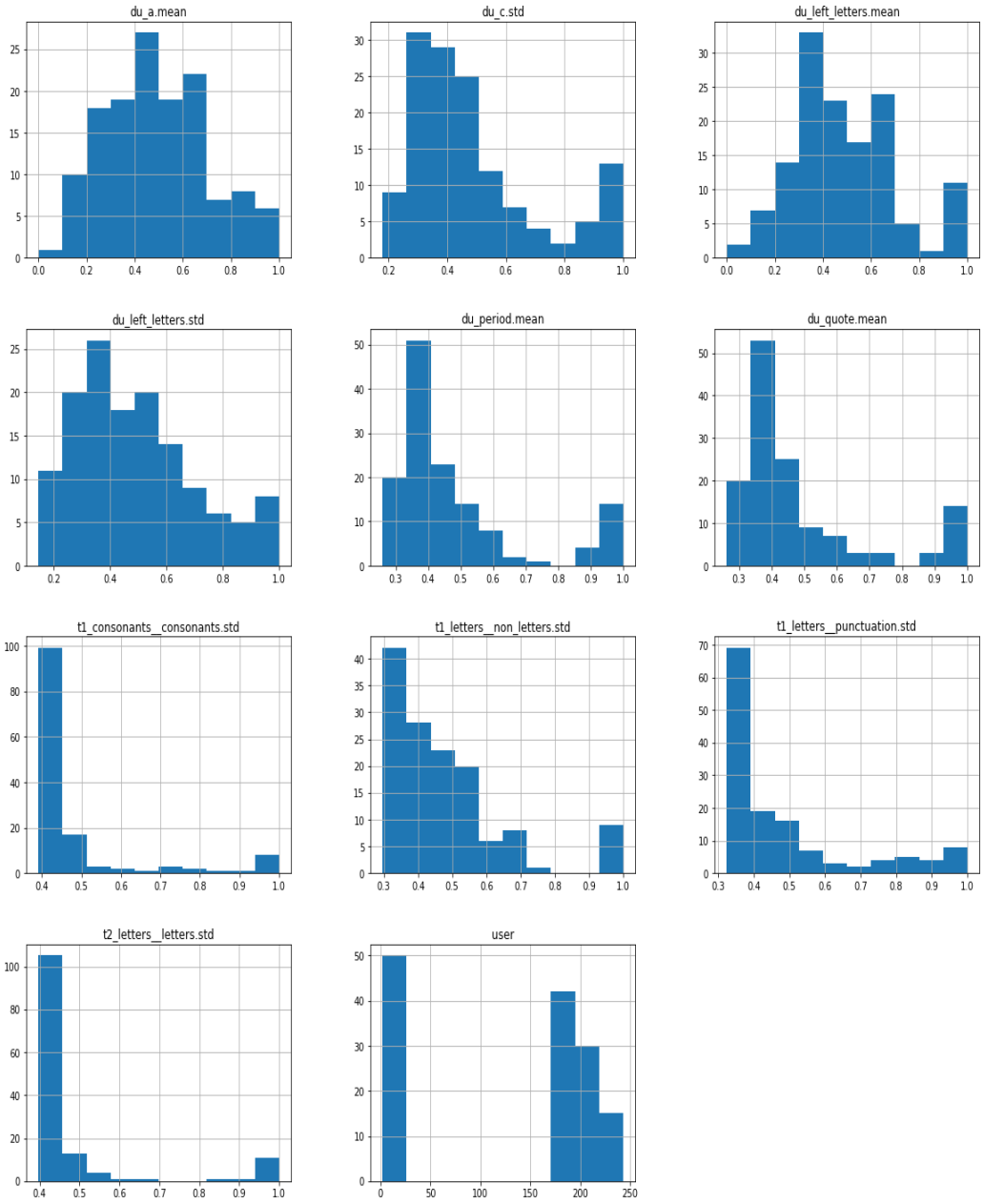

Figure B.6: 10 Features selected for the half sequence experiment.

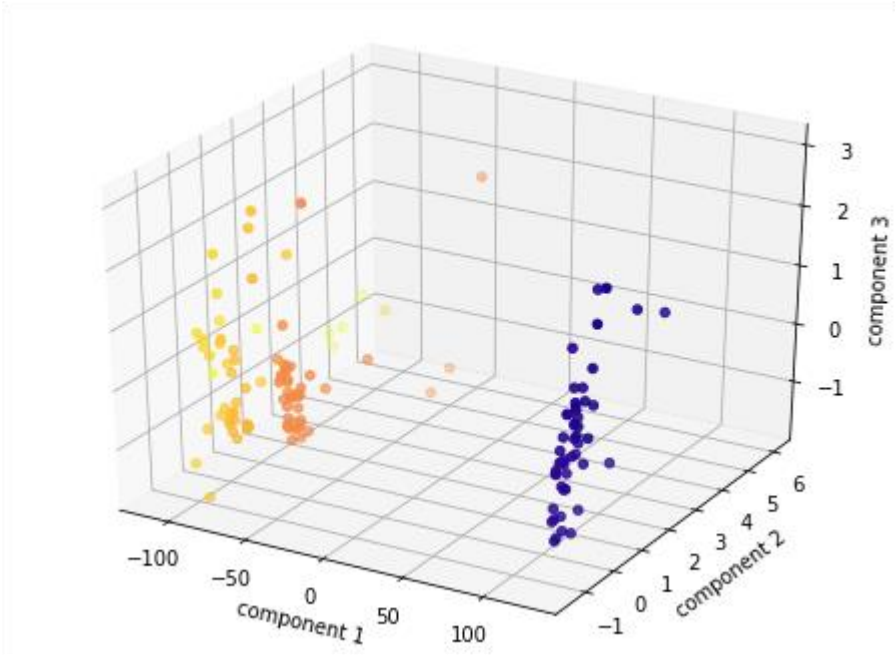

Figure B.7: 3D scatter plot before feature selection for the half sequence experiment.

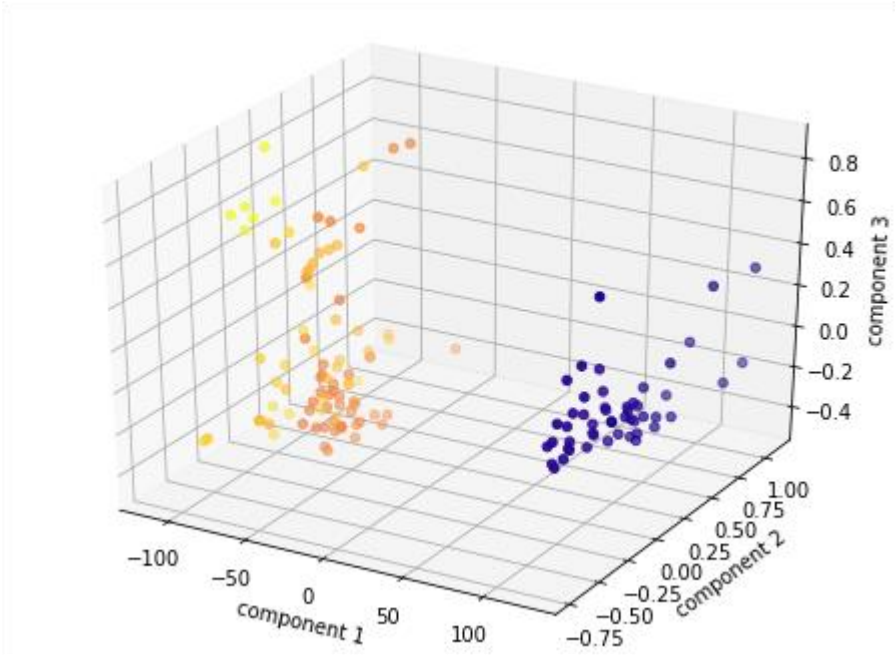

Figure B.8: 3D scatter plot with 10 features for the half sequence experiment.

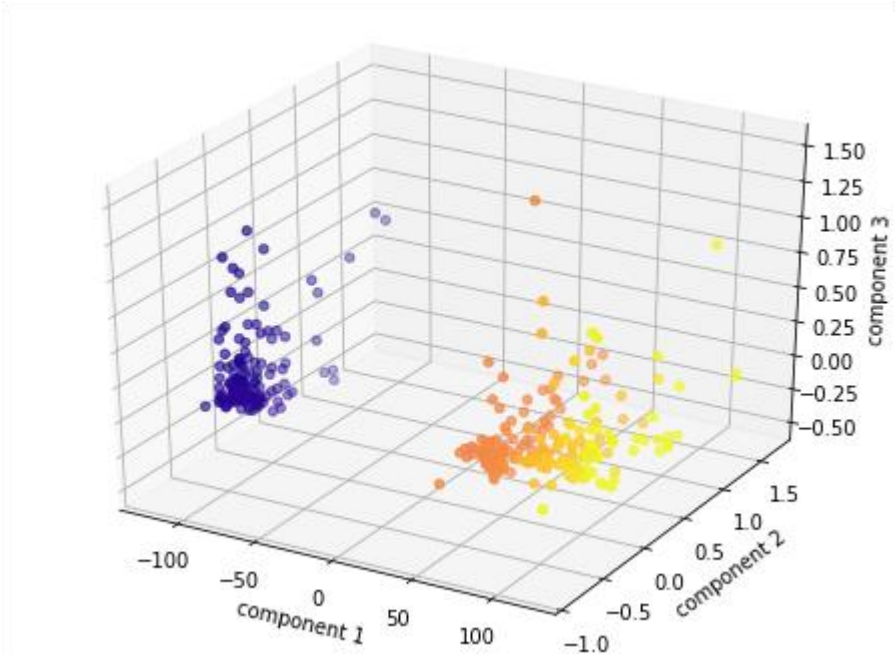

Figure B.9: 3D scatter plot with 20 features for the four HCI task dataset.

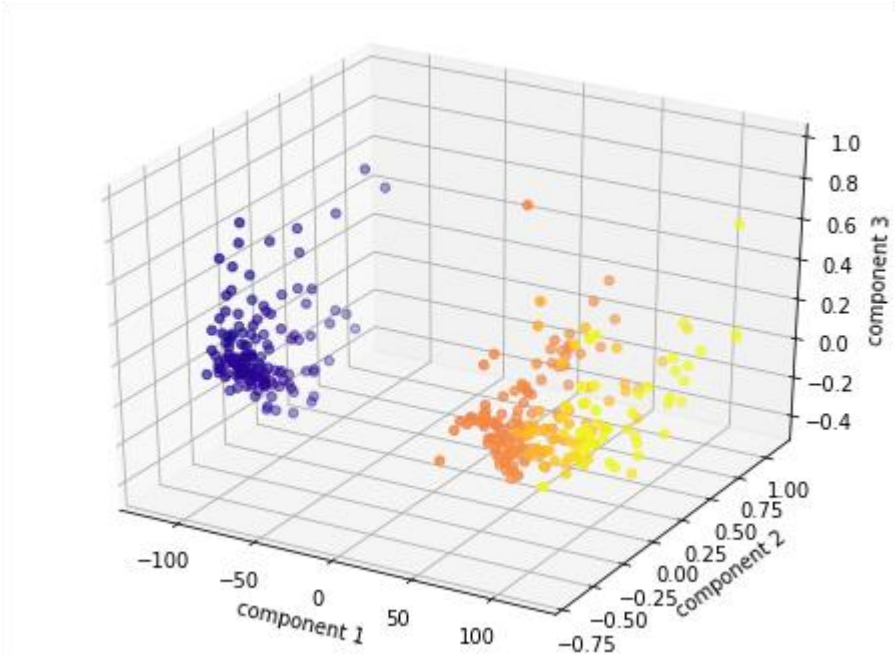

Figure B.10: 3D scatter plot with 20 features for the four HCI task dataset.

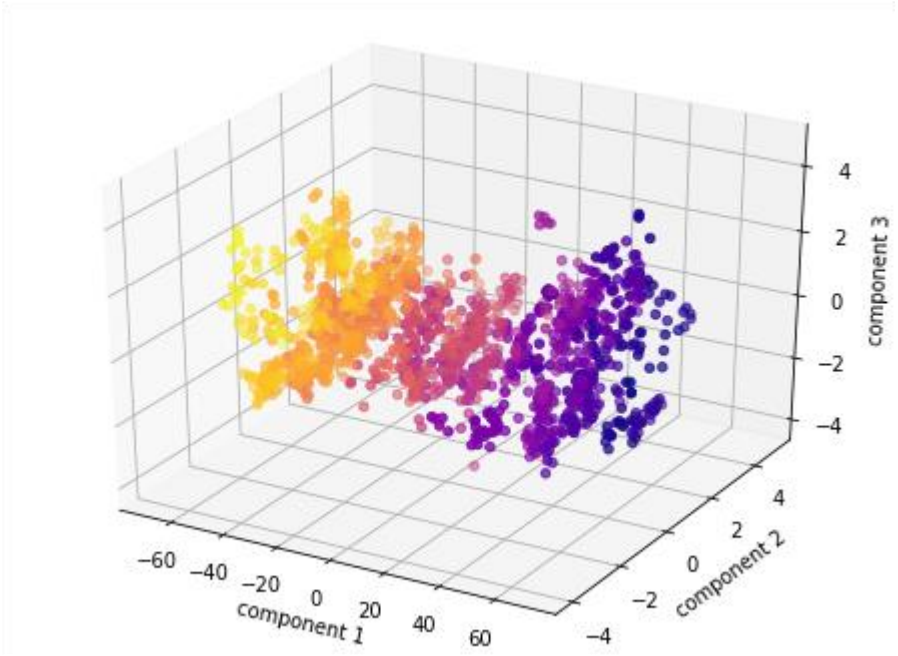

Figure B.11: 3D scatter plot with all features for the Villani keystroke dataset.

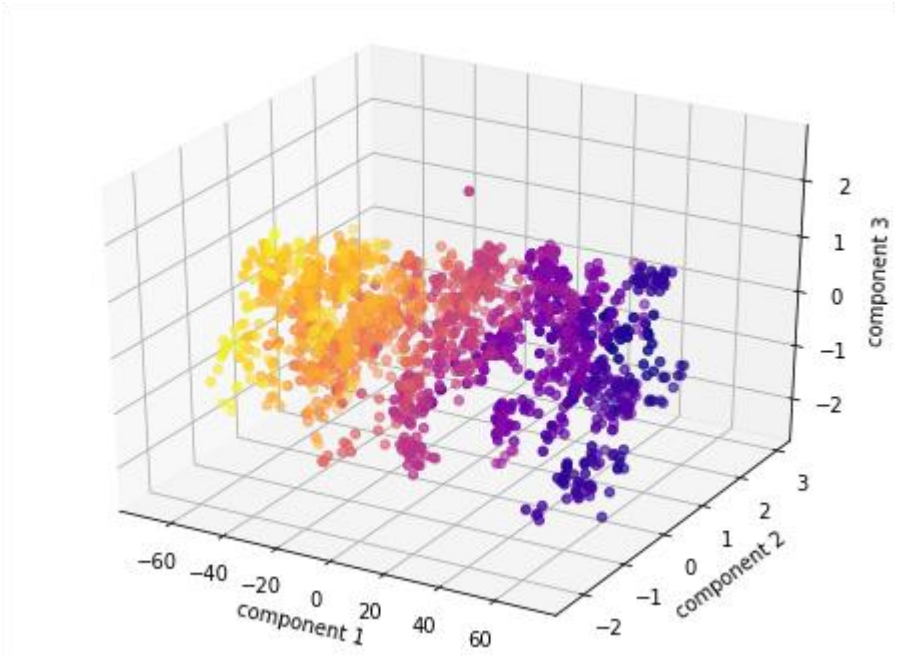

Figure B.12: 3D scatter plot with 50 features for the Villani keystroke dataset.

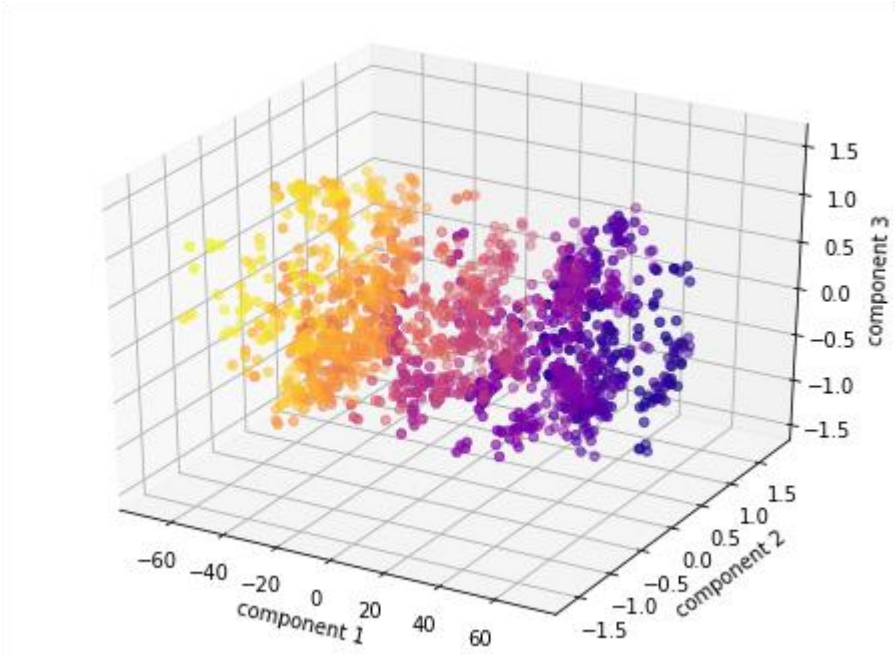

Figure B.13: 3D scatter plot with all features for the Villani keystroke dataset.

VITA AUCTORIS

NAME:	Emamuzo Cletus Ogemuno
PLACE OF BIRTH:	Warri, Delta State, Nigeria
EDUCATION:	Bachelor of Science in Computer and Information System, Achievers University Owo, Ondo State, Nigeria, 2011
	Master of Science in Computer Science, University of Windsor, Windsor, Ontario, Canada, 2018