PLAYER AUTHENTICATION BASED ON MOUSE AND KEYBOARD DYNAMICS

Lauri Vanamo

Master's Thesis
May 2016

DEPARTMENT OF INFORMATION TECHNOLOGY
UNIVERSITY OF TURKU

UNIVERSITY OF TURKU
Department of Information Technology

VANAMO, LAURI: Player Authentication Based on Mouse and Keyboard Dynamics

Master's Thesis, 55 p., 0 appendix p.
Computer Science
May 2016

Often in online games it is necessary to be able to recognize or authenticate a known player (for example owner of the game account) from some other player. The other player might be, for example, hacker, gold farmer, or even some third party to whom the original owner has sold her account. This kind of authentication can be done by biometric means, using machine learning algorithms for classification. Suspected user's biometric data, that is, data collected from user's mouse and keyboard usage, can be classified towards known data. If mouse and keyboard usage patterns of the game account owner and the suspected user differ enough, foul play can be suspected, and more thorough investigation by other means can be started

From player's point of view this kind of data collection and analysis is inconspicuous and non-intrusive. It does not disrupt the gaming experience nor is threat to player's privacy or information security. The data collection can be done in background without user noticing, and only the mouse and keyboard usage inside game can be collected because of restrictions enforced by the operating system. It is also noticeable, that any game that uses mouse and keyboard, uses the input data anyway, so now only one more usage for that data is proposed. Player also gets an extra layer of game account security in form of continuous biometric authentication.

In master's thesis, mouse and keyboard data is collected from two players, that are in a slightly different skill level (experienced gamer, less experienced gamer). Granular raw data was then transformed into a dataset, prepared, and analyzed. Classification was carried out with k-Nearest Neighbor classification with leave-one-out cross-validation.

Keywords: computer games, continuous biometric authentication, mouse and keyboard dynamics, k-nearest neighbor classification

The originality of this thesis has been checked in accordance with the University of Turku quality assurance system using the Turnitin OriginalityCheck service.

Table of Contents

1 Introduction

1.1 Research Question

In some situations, it is necessary to distinguish two different computer users from each other. One good example would be a case, when there is a suspicion that online gaming account has ended up in wrong hands. Often also selling online gaming account is forbidden in the game's terms of service (TOS), often also called as terms of use (Blizzard Entertainment 2010). This causes a justified need for the game company to be able to make sure, that the original owner of the gaming account is still in control, and no foul play exists. Usernames and passwords are not necessarily enough to ensure account safety or TOS-rules, as they can be given out, sold or stolen. For this reason, there needs to be a way to recognize owner by some feature she has. Recognition can be based virtually on any feature that is inherent for the player and can be measured. These biometric features could include, for example, fingerprints, iris-scans, speech etc., but the recognition can also be based on some behavioral features.

In this research I'm studying the possibility to distinguish two computer game players based on their mouse and keyboard usage patterns. Behavioral mouse and keyboard data is also called a mouse and keyboard dynamics. Because in research of this nature, a dataset consisting mouse and keyboard dynamics from a computer game was needed, but none was readily available. For that reason, an experiment was conducted, in which two players played same quest chain, and the mouse and keyboard usage was recorded. Dataset was then formed from this raw data, and then by means of machine learning algorithm, k-Nearest Neighbors, the classification was performed.

Goal is to explore possibility of differentiating players, so the research question is formulated as follows: Can two players, playing the same computer game, be distinguished from each other based on differences of how they use mouse and keyboard? In practice, answer to the research question could also help to provide an answer to the question: Has the mouse and keyboard behavior in certain gaming account changed so much in a short period of time, that the account is suspect to be

compromised? Can this kind of change be detected, and how much confidence can be put into results?

In traditional statistical research setting, the research question can also be formulated as statistical hypothesis testing, where data is seen as it is produced by some invisible or unknown process, i.e. statistical model, that needs to be studied. This formulation is done by forming a null hypothesis, in this case the null hypothesis would be "there is no difference between players", and the alternative hypothesis would be "there is statistically significant difference between players". Now there are two hypotheses, from which the another one is the correct one:

$$H_0 : there\ is\ no\ difference\ between\ players$$
$$H_v : there\ is\ statistically\ significant\ difference\ between\ players$$

There are two problematic words in above formulations: what does "statistically significant" mean in context of classification? What threshold in classification performance can be seen as significant enough? In this research there is no test statistic p-value related to statistical model that can give the answer, and it need to be determined some other way.

To narrow the scope of the thesis, only two players participated in experiment, and they had somewhat different gaming skills and experience. Both of them are long time computer users with touch typing skills and familiarity with computer gaming. Main characteristic difference between players is, that another player has played game used in experiment earlier, as another has only tried it shortly couple of times before. Because of the experience difference, only starting area quests in the game was played; only limited game character skill set, movements, etc., were available at that point in game. Quests offered at starting area are also easy to comprehend and execute ("See those wild boars over there? Go kill ten of them and return to me."), so no extra complications or hidden features were present in experiment. Quest texts (quest's textual representation) also offered very little of game story, lore, thus minimizing idle mouse and keyboard input while reading.

Research about differentiating computer game players is also loosely related to the game bot detection. Game bots are scripts made to control the avatar (playable game character) without player's presence, and automate repetitive tasks, called farming. Bots are usually used for online gold farming, where in-game currencies are accumulated, and often sold for real money to other players. Using game bots ("botting") is tied with game account theft; hackers seldom start playing with stolen account, but more likely set up a bot for gold collecting. Collected gold can then be transferred to safety, away from stolen account. Using game bots is forbidden in most online games and leads, if detected, to a banned game account. For this reason, bots are rarely used in legitimate gaming accounts. Game bots could also be detected with same kind of mechanism as proposed earlier.

As many problems relating to user authentication in computer games have much to do with having unfair advantage in game, or formation of black market around bot farmed gold and selling stolen accounts, and because of these phenomena can have big impact to honest player's gameplay, it might be necessary to reinforce existing systems that enforces fair play and anti-hacking rules in online games. One such way could be the topic of the thesis: mouse and keyboard dynamics.

1.2 Branch of Science and Related Fields

Research is strongly rooted to the computer science, and its many areas consisting of, but not limited to, computer games research, machine learning, data analysis, programming and operating systems. Computer games research gave a solid framework and meaning for research, machine learning and data analysis provided algorithms and methods to be implemented for the analysis, and some amount of programming and operating systems knowledge were required to implement the mouse and keyboard data collection and the machine learning algorithms. Basic knowledge from algorithmics was also helpful when designing the software implementations to keep their running times reasonable.

Other related fields concerning the thesis are biometric methods in user authentication, and statistical methods and tools. Also, a large role was played by the framework for

statistical research (i.e., how statistical research is usually done), which together with Cross Industry Standard Process for Data Mining (CRISP-DM) laid sound groundwork for structure of research in the thesis. Overall structure based on both of these two were matter of the utmost importance, because lack of experience doing experimental/data mining research I had before this project. So for this reason, although they are not in the center of the topics discussed in thesis, they were brought up for discussion because of their importance to successful execution of the research.

Although there are multiple different things to be taken into consideration, the main focus is at all times in the data, and how it should be transformed to a dataset, from which signal needed for classification can be found. I constantly tried to keep things as simple as possible, because in this kind of research things getting complicated can easily lead to an endless swamp of possibilities in which non-seasoned researcher as myself would end up lost. In other words: keep it simple, focus on basic principles, and try to produce best possible outcome from this perspective.

Because of many different, yet related fields, I tried to pick up the most important aspects considering the thesis and aggregate them to comprehendible whole. Many of the theoretical subjects discussed in chapter 2 may seem unrelated at first, but all of them has their place in the work.

1.3 Used Materials

Materials used in this thesis are mostly computer science textbooks (total of 10), as well as mouse and keyboard dynamics, and machine learning articles from last five or so years (total of 13). In some places also other, non-scientific original online resources are used, such as game company's legal document, etc. Mainly the primary sources of information were used (if one exists), and materials has been chosen by perceived reliability and relevance. Even the documents mentioned latter above are trustworthy and original sources for certain information.

In other words: source criticism is in focus. I cannot, of course, fully evaluate reliability of research papers made by researchers from complicated topics, and for this reason,

some things must be taken as given, if the source is otherwise perceived as reliable (e.g., the source is a scientific publication from a recognized institution, and no game changing criticism from other researchers were made later). Same qualification processes were conducted with textbooks used, and perceived reliability and overall recognition the book has gotten were the deciding factors. My key point here is the following: some things must be trusted if they seem trustworthy. This thesis is not a quest for absolute truth, as this is not thesis in philosophy (or mathematics for that matter), but merely a quest for search of increased knowledge in area of mouse and keyboard dynamics based on computer games.

1.4 Research Methods

Everything is, of course, based on tradition of the scientific method, where focus is in investigating the matter while trying to obtain new knowledge and integrate existing research. Data collection was made in controlled experiment setting, in which both players had same hardware (especially keyboard and mouse). Behavior was observed and recorded, and the research question was answered based on the data. Research hypothesis and research question are the basis for methods used. Assumption, that there must be a difference in two players mouse and keyboard dynamics, because no two humans use mouse and keyboard exactly the same way and each should have their own own usage patterns, is in focus. Proving this to be correct is the way to go, because it is easier to prove that the difference between players exists, than it does not exist. If the difference does not appear, it might as well be because of researcher's skills, not the data.

Methodology in this research can be seen also from totally different perspective; it has been based on typical steps of conducting statistical research. 1) Planning phase, in which data collection and research is planned, and decisions how to proceed further were made, 2) data collection phase, i.e., the experiment, 3) getting to know and understand the data, 4) statistical analysis, and finally, 5) reporting the results. Methods for phases 1 and 2 are mainly borrowed from other similar kinds of research studies made, and are proven to work well. Phases 3 and 4 utilizes common statistical

and machine learning methods, and in phase 5 methodology was to compare research and achieved results to existing research done by others.

In addition to research setting based to the scientific method and tradition of statistical research, there is no other specific research methods used, mainly because they did not seem to fit very well, or were necessary at all. As would be the case with research made entirely based on literature, where different, often multiple research methods need to be used for reasonable outcome, is this not the case in experimental research setting like one in thesis. In thesis, the framework of statistical research, with constraints and rules forced by used tools and methods (e.g., k-Nearest Neighbor algorithm), and logical choices made based on existing knowledge, are enough to provide sound, reliable and reasoned outcome.

1.5 Contribution

Every scientific research done should give its contribution to the science it depends on, and this is the case with this research and thesis as well. My main proposed contribution is the new raw dataset collected, that is set to be public and for everyone to download and use in future research. Thesis also offers a one perspective for continuous or partially continuous player authentication, and has some proposed ideas of securely implementing it without being, at least seemingly not, easily to be tampered with. These could potentially provoke thoughts and ideas in future research in related fields.

In such unlikely case, that mouse and keyboard dynamics related research has not yet been done in realm of computer games, thesis also offers some initial results for mouse and keyboard dynamics in this application area. I have no absolute certainty whether or not this is the case, as I have not found any research where mouse and keyboard dynamics is directly related to computer games, but surely it should exist. Even if not in public domain, but inside gaming companies or security research firms.

1.6 Structure of the Thesis

In chapter 2 I take a look to the theoretical background of methods used, and form the basis to the thesis. Topics discussed ranges from cheating in computer games and biometric authentication, to a look to the existing research in the field, and to machine learning. Second chapter was designed to give the reader theoretical background needed for understanding research made, and also parts of rest of the thesis. A goal is also to discuss shortly the ties between theory and research conducted, in effort to give the reader better understanding why certain topics are relevant.

Chapter 3 has its focus in main subject: the data. This chapter discusses how it became to be, what it is like, and what is to be done with it. Third chapter exists to give the reader all information necessary to understand nature of the data used in research and thesis, as this understanding will become much needed when advancing forward to constructing the dataset and conducting analysis.

Chapter 4 consists of the actual analysis, i.e., classification and the results. Fourth chapter can be seen as the main course, as it reveals partial answer to the research question. This chapter, although not complicated or large in size, is fundamental core of the thesis.

Results are revealed, and then analyzed as well as discussed in chapter 5. Finally, in chapter 6 conclusions from the research are made, and thoughts and ideas based on results are discussed.

2 Theoretical Background

2.1 Cheating in Computer Games

There exist multiple reasons for cheating in computer games. Especially in multiplayer online games, greed has become more and more driving factor to encourage cheating. In other words, instead of cheating in actual gameplay, cheating is often done because of the financial gains from selling virtual assets, such as virtual currencies or gear. Virtual economies in online games has grown to a degree, that they can have gross national products comparable to real world countries. (Smed & Hakonen 2006, 213.)

As known from human behavior, when somewhere is market opportunities and demand, shortly there is also people providing the supply and selling them for money. This is the case online games as well, where whole industry has risen to sell virtual goods for real world money. Real money transactions to virtual assets are often prohibited by the game maker, because of negative side effects; cheaters hack other players' game accounts, steal the goods and sell them to others with real money. Gamers themselves might also, after playing for a while, choose to sell their accounts, and possibly start playing with new account. One who buys readily more powerful game account has unfair advantage in eyes of other players, who themselves worked for their prestige. In gaming world, this is often referred as "pay to win".

Game bots are also part of the problem. A game bot is a script or machine learning based AI construct, which is developed automate parts of gameplay such way, that the gamer does not need to be present all the time. Problematic side effects for using game bots exist, as Pao et al. states: "This phenomenon often annoys honest users as it erodes the balance and order of the game world, where bot users and the customers of gold farmers can monopolize scarce resources and easily outperform honest users in terms of economics and military force" (2010, 162). They also note, that usage of game bots is the second most impactful form of cheating that affects games, right after account theft (Pao et al. 2010, 162).

Robles et al. (2008, 145) in their paper about online games and security issues comes to conclusion, that the cheating in online games are activities that modify the game experience to give one player an advantage over another player. Whereas Robles et al. quite nicely identifies and provides basic taxonomy to types of cheating in online games, and mentioning hacking, farming and use of bots, their view for preventing the foul play is – dare it be said – typical to most research papers. In addition to widely used means, like anti-cheat software, patching the game to prevent exploits and banning suspected cheaters, they propose the use of CAPTCHAs (Robles et al. 2008, 147). CAPTCHAs are textual messages only supposedly readable to humans that can be prompted to user if needed. The problem here is, that the CAPTCHAs are conceptually notoriously disruptive if used mid-game, and useless, if used only at the start of the game, because bot can be started after prompting the user with CAPTCHA. CAPTCHAs also does not help in case of sold or hacked account, when there is a real human playing with non-legitimate account.

Anti-cheat software is not real answer either, because their use is limited into observing other programs in player's computer, that no known hacking or game bot software is present, and preventing any other program to interfere with the game client (game bots need to read and write game's data directly to game's memory space). In other words, anti-cheat programs do not measure players' behaviors, at least not for current knowledge. Some game companies have rumored to been tracking behaviors of user account in server side; login times, IP-addresses, and such. If there is to be sudden changes in these behaviors, suspicion about account ending in wrong hands can be raised. But again these tracking practices, although plausible and even possible, have no official sources backing them up, and are just rumors. This actually illustrates nicely hardships of doing even mundane research in cheating prevention from this perspective (i.e., what is done to prevent cheating in real world online games): methods are often trade secrets of the companies, and are not revealed to public in fear that hackers could use the information against them. This has led to situation, in which only bits and pieces of knowledge surfaces every now and then, and that information is usually from hackers themselves discussing these topics in the Internet.

Cheating, either in-game, or stealing/selling game accounts and assets, causes real demand for solution to prevent account thefts and and black market. Solution to also

prevent using game bots is direly needed. It is important to note, that cheating in its various forms can cause unsatisfied customers and monetary losses to game companies. Any methods, that are not perceived as intrusive or disruptive from player's viewpoint should be considered. Thus, when adding extra layer security and cheating prevention, one possible way could be to factor in the mouse and keyboard dynamics, as an addition to the existing security framework.

2.2 Biometric Authentication

Authentication is often understood as preventing unauthorized access to the computing system. As Andrew S. Tanenbaum (2009, 693) notes in his widely recognized textbook *Modern Operating Systems* regarding authentication: "Every *secured* computer system must require all users to be authenticated at login time." Tanenbaum (2009, 640) also lays out three principles in which the user authentication can be based: on (1) something user knows (e.g., a password), or on (2) something user has in her possession (e.g., a keycard, or an authenticator device, which gives newly generated one-time password), or on (3) some property or feature the user has. Type 1 authentication, especially passwords, are commonly utilized in all kinds of usage-scenarios ranging from mobile phone PIN-numbers and personal computer passwords to large web-services and online games. Despite the common acceptance of the password-based authentication, type 1 authentication cannot be seen as entirely secure (National Institute of Standards and Technology 2011). Username (account name), which is usually coupled with the password, is often more or less public information, or at least it often can be guessed from publicly available information. When this notion of public username is combined with the fact of users being lazy and choosing easy to remember passwords, is the security at risk (Imperva 2014). Type 2 authentication is more secure, especially in combination with username and password, but it requires every user to have the token needed for the authentication. These tokens can be costly, and they may not be readily available to the user. Both of type 1 and 2 authentications also suffers from the fact that the authentication is usually done only once in the login-phase. Authentication could be redone in the middle of the computing or gaming session, but it can easily become annoyance to users, disrupting workflow in office, or

immersion in computer games. For this reason, something entirely different is needed to ensure hassle-free, strong security. This is when type 3 authentication comes to play.

Type 3 authentication, also called a biometric authentication, has its focus in authenticating users by some of their biological feature. Fingerprints, iris-scans, DNA, and such are probably the most recognizable, because of their constant appearances in entertainment industry (books, movies and TV-shows), but in latest years increasingly in real world applications too. But there is also an another way of using a biometric authentication: not based on physical feature, but based on the way user does something (movements, gestures, speech, etc.). In thesis, the focus is in the mouse movements and keyboard strokes, i.e., the way user uses mouse and keyboard. This approach is called a mouse and keyboard dynamics.

2.3 Mouse and Keyboard Dynamics

Mouse and keyboard dynamics are the gestures, movements and usage patterns that a computer user produces in her everyday computing sessions. Shen et al. describes mouse dynamics as "…a behavioral biometric for analyzing behavior data from pointing devices…" (2013, 18). Keyboard dynamics can be characterized in same manner, as it is seen as behavioral biometric that has its focus in identifying users based on the analysis of their keyboard typing dynamics (Ahmed & Traore 2014, 458). Even though the idea in both is similar, differences between the mouse and keyboard dynamics data is quite remarkable. Intuitively mouse can be seen as a device with lots of freedom in movements, while keyboard has fixed locations for buttons.

Interplay between mouse and keyboard in computer games is deep and profound, because there is often possibility to use either one for same task. For example, game character abilities can have shortcuts for keyboard (e.g., number keys 1-10 are by default used by certain abilities), but user can also decide to click them with a mouse. Another example could be moving forward, which is usually tied to key "w", but same effect can be achieved by clicking and holding both mouse buttons down simultaneously. These features could possibly be ones that allows differentiation of

players, in addition to others, such as how fast player pushes buttons, or what direction mouse movements usually are.

Continuous biometric authentication means that the authentication is done in ongoing basis, in oppose to traditional login-time authentication. Continuous authentication collects authentication data from whole duration of computing session, or in certain intervals. Ongoing authentication can prevent situations, where account is taken from the original owner after she has logged herself in (session hijacking), or when owner logs into a system, but then gives control to third party. (Ahmed & Traore 2014, 458.) Many continuous authentication methods, such as password prompt or request for some other code can be easily disruptive to the user. Continuous biometric authentication based on mouse and keyboard dynamics can be done without user noticing anything special is going on. Mouse and keyboard dynamics based authentication should be implemented such way, that the security is maximized while intrusiveness to player is at the same time minimized.

One major advantage in using mouse and keyboard dynamics for biometric authentication is that no special hardware is needed (Shen et al. 2013, 16). Every personal computer these days has a mouse and keyboard, so only software collecting and analyzing behavior data is necessary. In other words, there is no cost associated in special hardware (e.g., fingerprint scanner), and the extra layer of security that mouse and keyboard dynamics could provide can be distributed via software download. That is why mouse and keyboard dynamics can be appealing and easy solution for the end user and to the gaming company alike.

Based on existing research, mouse and keyboard dynamics is not alone sufficient enough (see chapter 2.4) to provide full player authentication in online games, and it should be used in addition to existing authentication methods. After player has logged in with username and password, continuous authentication could start, either in intervals or full time. Mouse and keyboard behavior data can then be used to make decisions if the user seems to be who she says she is, or if there is suspicion that the account has ended up in wrong hands.

2.4 Existing Research

Fairly large amount of research has been done in last decade or two in field of biometric user authentication via mouse and keyboard dynamics. Need for continuous user authentication has risen to new importance with advanced Internet and cloud computing applications, such as Google Apps, Office 365, Dropbox, etc. Most papers indeed focus in online banking, email, or other crucial everyday user authentication situations, in which authentication must be trustworthy, non-disruptive and fast. Below is listed and shortly described a sample of research papers published in last ten years.

Shen et al. (2013, 16-30) released research paper about user authentication based on mouse dynamics, in which they achieved 8.74% false-acceptance rate (FAR) and a 7.69% false-rejection rate (FRR) with a corresponding authentication time of 11.8 seconds (sample size 32 mouse operations). With longer authentication times, results did get considerably better, and for example with authentication time 588.62 seconds and 800 mouse operations, FAR was 0.87% and FRR 0.69%. They used a one-class Support Vector Machine for classification, but tested also 3-nearest neighbor classifier, which yielded FAR 15.67% and FRR 14.53% performance for the smallest authentication time (11.8 seconds). Short authentication times are of course important in online applications, but more interesting for continuous authentication in computer games are longer authentication times and very good results derived from them. Computer gaming session can last multiple hours, and authentication is not time critical, i.e., authentication can be done over a long period, because risks of unauthorized use of game-account are not as severe as those, for example, with banking-account or email.

Rybnik et al. (2013, 245-250) explored keystroke dynamics authentication using non-fixed text of various lengths. With sample length of eight keystrokes, misclassification rate was 26.6%, but similarly decreased to 6.1% when sample length was 200 keystrokes. A nearest neighbor classifier with Manhattan distance metric was used for classification. Although otherwise trustworthy, the article caused minor questions regarding performance: "It is worth noting that the two compared text parts were selected at random, and most likely were different" (Rybnik et. al. 2013, 248). This

could cause overall classification perform better than it normally should be. But again, this kind of results suggest that the mouse and keyboard dynamics approach could be suitable for computer games that makes long authentication times possible.

Continuous authentication based on mouse dynamics were studied by Mondal and Bours (2013, 1-12). In their study various machine learning algorithms were tested, including k-nearest neighbors and support vector machines among others. They reported probability to impostor not to be detected (FAR) with k-NN to be 12% (Mondal & Bours 2013, 3). But they somehow did not report FRR value also, which is strange: FAR value can be set to zero by classifying all to impostors – and this is the reason for reporting also the false rejection rate. Their main focus was in SVM with same kind of feature set as in other papers mentioned earlier, and results were similar as well. Mondal & Bours used entirely different, obscure scheme for measuring performance based on their own previous work, and it made their research bit hard to get grasp on.

Traore et al. (2012, 138) combined mouse and keyboard dynamics approaches and achieved equal error rate (ERR) of 8.21% with Bayesian network. Their study shows combination of mouse and keyboard dynamics yielding promising results for authentication in web-environments. Results and used features are similar to other studies mentioned earlier.

Something little bit different is the article "Keystroke Analysis of Free Text" by Gunetti and Picardi (2005, 312-347). The research focuses into the text itself, not the dynamics, and so neither the research, nor the results are comparable with this research. Their data consists of typed letters, and dynamic timing information between letter being typed. Interesting part in their paper is how well the security environment, biometrics and application areas are discussed (not mentioning computer games though), and it has steered my thinking to certain direction even when article does not contain anything directly useful for this the research. Mainly the idea, that a typed letter (or a key) tied together with timing information will not work with computer games (discussed in chapter 3.3) emerged and became obvious while reading this article.

Main difference in existing research and the thesis are the following: usually research concentrates either in mouse or keyboard dynamics, not both, and in research goal is to get the authentication times as small as possible, which is not needed in case of computer games. Biometric authentication research based on mouse and keyboard dynamics is also conducted in other application areas than computer games, probably because computer games are not viewed to have a critical need for safety and privacy when compared, for example, to a financial or healthcare applications. While this is of course true, also opposing view can be taken into account: account thefts and using game bots can have a huge negative impact to fairness and enjoyment of the game for honest player (Pao et al. 2010, 162). Online games are also multibillion dollar business, and customer satisfaction is crucial for long term success. One could argue, that any measures that can be taken to ensure happiness of average honest gamer is a step to right direction revenue wise.

Another observation considering the existing research is, that keystroke dynamics is often understood as analysis of text, produced by humans, and from which the classification is made. It thesis, keyboard dynamics means similar as mouse dynamics: timing and distance information of key presses as they occur. Text-patterns or words produced are not the matter of interest.

2.5 Conducting Statistical Research

Well known, typical phases in statistical study are (1) planning and designing the study, (2) collecting the data, (3) getting to know the data, (4) statistical analysis, and (5) reporting the results. This research, and even this thesis-document, follows these steps closely. These steps are present, because research made had also an experimental side in which the data was produced. But as there is also lightweight data mining tasks, as well as machine learning that need to be tied to the research, also another model for data analysis part of research was needed.

Cross Industry Standard for Data Mining (CRISP-DM) defines a process model that is independent from application area and used carrying out data mining related projects. CRISP-DM is a collection of iterative phases, in which the process advances

and refines itself. Process start from business understanding, where the application area is explored. Second phase is data understanding, where knowledge discovery is in the focus. Third and fourth phases are data preparation and modeling, often tied tightly together. Fifth phase is evaluation, in which the success of the process is observed. Sixth, and the last phase is deployment, where system developed in process is deployed to use. (Wirth & Hipp 2000, 1-7.)

In first steps of this research, planning the study was made. Concurrently with planning, both application area (computer games) and research topics (mouse and keyboard dynamics) were explored to gain more knowledge for research. In next phase, the experiment was made and the raw data was collected. After this, started the iterative process of data understanding, preparation and modeling. When results satisfying enough were produced, evaluation and deployment (this thesis document) was started.

CRISP-DM in addition to traditional steps of statistical research worked very well for the research made, and for the thesis itself; thesis has the structure based on research phases mentioned earlier.

2.6 Data Preparation

Any modification or editing done to the dataset before actual analysis can be called data preparation. Data preparation is often also referred as data preprocessing. Goal in data preparation is to modify the dataset such way, that 1) actual errors and qualities that harms the analysis are removed, and that 2) data can be somehow transformed to a form in which the analysis is performed better. Group number one could include cases with missing values, values that are out-of-range, possibly erroneous values (outliers), etc. Group two can include cases such as standardizing or normalizing the dataset, or selecting subset of features for analysis (feature selection), etc. Problem is, that there are no formal guidelines for doing data preparation, because of varying datasets have different needs for data preparation. And to make the matter much more interesting: data preparation phase can often have major effects to generalization performance of machine learning algorithms (Kotsiantis et al. 2006, 111).

Data preparation is a step of crucial importance in data analysis. As Dorian Pyle states in preface of his book's Data Preparation for Data Mining: "In order to use the techniques, or make the predictions, industry professionals almost universally agree that one of the most important parts of any such project, and one of the most time-consuming and difficult, is data preparation" (1999, xvii). Especially interesting topic related to this research, Pyle (1999, 100-101) discusses the need for adapt the data preparation to modeling tool used, because many modeling techniques and methods can have certain sensitivities and needs for a dataset. In case of k-Nearest Neighbor algorithm when using Euclidean distance, importance of normalizing the data and being aware of "curse of dimensionality" are probably the main concerns. Both of these are directly related how the distance measure is calculated, and need to be taken into account.

2.7 k-Nearest Neighbor Algorithm

The k-Nearest Neighbor algorithm (k-NN) is a supervised, non-parametric machine learning method, that can be used with classification and regression tasks. Supervised learning means, that in a dataset exists some variable that acts as a supervisor (e.g., in this thesis supervisor is class label for the players), and based on this supervisor learning and prediction is made (Abbot 2014, 5). Non-parametric algorithm is good for use with data of unknown properties because there is no need for assumptions about distribution of the data. Non-parametric algorithms can be seen as more flexible than parametric algorithms that need known distribution, and thus are suitable for machine learning applications (Abbot 2014, 6).

The main idea behind algorithm can be described following way: when classifying new sample x, it should be seen as a data vector, and its variable values as coordinates in $|x|$-dimensional feature space. When x is classified, around it is formed $|x|$-dimensional hyper sphere that has inside itself exactly k closest data vectors to x. From classes of these k closest data vectors, the class of vector x is determined based on k-nearest neighbor classification rule. Rule says, that to minimize the probability of

misclassifying the new sample, it should be assigned to a class that has most members in its k nearest neighbors. (Bishop 1994, 57.)

The old and commonly used distance metric to calculate distances of data vectors in k-NN algorithm is Euclidean distance (Short & Fukunaga 1981, 622). Over the time, multiple distance metrics for k-NN has been developed, including Mahalanobis distance, City block metric (Manhattan distance), and Minkowski metric to name a few. Euclidean distance is probably the easiest one to comprehend, as it has its basis in basic school mathematics, namely in Pythagorean formula. Distance $d_{v,w}$ between two vectors v and w in n-dimensional space, also referred as Euclidean space, is:

$$d_{v,w} = \sqrt{(v_1 - w_1)^2 + (v_2 - w_2)^2 + \cdots + (v_n - w_n)^2}$$

where v and w are two the data vectors.

As a side note from purely mathematical standpoint; as the Euclidean distance only works in Euclidean space, i.e., one that conforms Euclid's postulates, it is not necessarily the best metric used in every case of a problem that arises from real physical world. There seems to exist quite much evidence for universe to actually be overall hyperbolic in its geometry, and in hyperbolic space Euclid's parallel postulate does not hold (Penrose 2005, 46-49). There are of course other, more pressing reasons not to use Euclidean distance, for example, that many datasets may have better performance with different metrics.

Pseudo-code of a full k-NN algorithm and its operation is described below:

```
k-NN(training data, test sample) {
   1. Go through training data and calculate distances from
      training samples to test sample
   2. Search the training samples for k smallest distances
   3. Return class label that has most members within those k
      training samples
}
```

Figure 1: Pseudo-code for k-NN algorithm.

As can be seen from pseudo-code listing, k-NN algorithm is quite easy to understand and most importantly easy to implement. But something else is also needed for k-NN to be really useful: when exploring the suitability of k-Nearest Neighbor algorithm for a given task, it is often best used with cross-validation technique, which in combination with k-NN gives better estimates for k-NN prediction performance in generalized data. Cross validation is discussed in next chapter 2.8.

Curse of dimensionality is one potential problem when using k-NN algorithm. Jerome H. Friedman addresses the bias-variance tradeoff and the curse of dimensionality with k-NN classification in his article:

> For K -nearest neighbor procedures the bias-variance trade-off associated with estimation error generally is driven by the bias in high dimensional settings (many inputs). This is due to the geometry of Euclidean spaces; the radius of a region varies as the nth root of its volume, whereas the number of training points in the region K varies roughly linearly with the volume. Thus, even the smallest possible volume (K = 1) gives rise to large regions in terms of radius. This can already produce high bias even for the largest variance (K = 1). This phenomenon is referred to as the "curse-of-dimensionality." (Friedman 1997, 65.)

In other word, this means that when having high amount of features in a dataset, k-NN algorithm tends to provide higher bias (classifier does not accurately capture patterns in training data) even in cases when bias should be small, i.e., with small k-values. Essentially traditional bias-variance tradeoff is then broken, and only high bias, high variance results are produced; classifier does not capture patterns very well, and it also does not generalize to unseen data well because of high variance.

2.8 Cross Validation

Cross-Validation is a statistical concept, resampling technique, that can be used to provide information for model selection, and to provide estimates of the prediction error in classification and regression (Berthold & Hand 2007, 58). Multiple different cross-validation schemes exist, and the general K-fold cross-validation can be

described as follows: 1) First split the data into K equal-sized parts, 2) use one of these parts as testing set, and other $K - 1$ parts for training, record the prediction error, and 3) proceed with different testing and training sets until every one of K-parts is once used as testing set, and then finally combine prediction errors (Berthold & Hand 2007, 58). When the K-value is same as size of the dataset, the K-fold cross validation is called leave-one-out cross-validation, which is the one used in this research.

How many fold in generally should then be chosen for cross-validation? With larger K-values, all the way up to $K = N$ (leave-one-out), the cross-validation has less bias for the expected prediction performance, but can have higher variance because the N training sets are so close to each other. Also computational load can be an issue for larger K-values; the learning method must be applied N-times. Smaller N-values on the contrary have less variance, but might produce biased results, if the performance of the learning method varies with the size of the training set. (Hastie et al. 2009, 242-243.)

Comparing to a single classification task, in which data is first split into the training and testing sets, and then machine learning algorithm is trained with the training set, and its performance is measured with testing set, cross-validation will give much more robust results when considering generalizing to a new, independent dataset. In other words, performance estimate given by single classification is often less accurate than one provided by cross-validation, and thus will not be a good measure when looking for how well the model will generalize to new data. This can be reasoned the following way; multiple, K-number of tests with different test data will probably give more accurate result to a new data, than a single test with single test data. In single test the data can somehow get sampled such a way, that the results are skewed. Skewness can be reduced by sampling and testing the data multiple times, e.g., with cross-validation.

Cross-validation can also be used in model selection. In case of k-NN algorithm, cross-validation can be used to determine the k-value which gives best results for the data. Running cross-validation with different k-values, $k = 1, 2, 3, \ldots$, will give quite good estimates for classification performance with different number of neighbors. It is then easy to pick k-value with highest performance, or smallest error.

2.9 Performance Measure

Performance measure for classification, is a measure how correctly the classification was performed, or how large different errors made in classification were. Two often used performance measures in binary classification are Accuracy (ACC) and Area Under the ROC Curve (AUC or sometimes referred as c-statistic). The Receiver Operating Characteristic Curve is commonly used summary for tradeoff between False Acceptance Rate (FAR) and true positive rate (Hastie et al. 2009, 317). False acceptance rate is also called a false positive (type I error), where true negative condition is predicted to be positive instead. False rejection rate (FRR) in contrary is the false negative (type II error), where true positive condition is predicted to be negative instead. FAR and FRR are terms used in authentication, as in case of type I error where user with no legitimate privileges will get authenticated, and in case of type II error the legitimate user will be denied the authentication. Classifier accuracy, classifier error rate, false acceptance rate and false rejection rate are the ones used in this research to report the results.

Confusion matrix, that illustrates error types is below:

		True condition	
		Positive condition	Negative condition
Predicted condition	Positive condition	Correct prediction	**Type I error**
	Negative condition	**Type II error**	Correct prediction

Table 1: Confusion matrix illustrating error types.

Accuracy (ratio of correctly classified cases) for classifier can be calculated from confusion matrix with formula:

$$ACC = \frac{Number\ of\ correct\ predictions}{All\ predictions\ made}$$

And an error rate (ratio of misclassified cases) is then:

$$ERR = 1 - ACC = \frac{FAR + FRR}{All\ predictions\ made}$$

FAR and FRR are calculated with the following formulae:

$$FAR = \frac{Type\ I\ error}{\#\ of\ negative\ conditions}$$

$$FRR = \frac{Type\ II\ error}{\#\ of\ positive\ conditions}$$

FAR and FRR in context of this study can be thought the following way: Player1-class is the positive condition, and Player2-class is the negative condition. FAR is situation where Player2-class is classified as Player1-class, and FRR is similarly situation where Player1-class is classified as Player2-class.

3 Experiment and Data

3.1 Data Collection

Data collection was performed in a personal computer running World of Warcraft game with patch 5.1.0A, version 5.1.0.16357 (but any game version from patch 4.0.3A onwards to at least to spring 2016 will have same quest-chain available). Two players, "Player1" and "Player2", played same quest-chain that begun from Human starting area onwards to quest which included first visit into the Stormwind city, where experiment was completed. "Player1" played first and logged off, and then "Player2" played the same route. At the beginning, both players got time they needed to get used to controls and movements, and recording of mouse and keyboard data started when players were ready to start questing.

Experiment setting (data collection) was set to be as controlled and similar as possible. In real world level similarity was based on using the same hardware (computer, keyboard and mouse), time of the day and network traffic (latency ranging from 12-20 milliseconds) were similar, and no lag or disturbances were observed. In game world similarity was based on same quests in same areas, same character race and class used, and only default character skill key-bindings were allowed. Inside the game, there is of course some amount of uncertainty, for example all quest goals are not necessarily in exact same places, but might wander around a bit. This kind of dynamical game world cannot, and is not even necessary to be circumvented. Goal were to set similar test-setting up for more stable player comparisons, but in-game setting is not the important thing here; the way player uses mouse and keyboard inside game is. Features were chosen to reflect this idea; they should be logically as independent as possible from decision made in-game, and also in-game contents.

There was no research nor consideration made when planning what features to collect from mouse and keyboard in data collection phase. Mainly, because it seemed unnecessary: better collect all data that is collectable, and later on weed out the parts deemed not to be useful. This is why the data collection included everything that could

become even relatively helpful for analysis. It is always easy to discard extraneous information later.

3.2 Raw data

The raw dataset consists of total *77 037* samples with variable dimensionality between five and eight features. *4*8 929 is belongs to class 1, and rest 28 108 to class 2. Data is very granular, that is, it's samples are the smallest pieces of information about mouse- and keyboard usage, that is possible to get out from Mac OS X –operating system via existing application programming interfaces (APIs). Granular nature is also derived from the fact that added calculations were not seen as ideal in collection phase, as complex calculations could have congested or otherwise affected the experiment. All calculations from this very basic form of collected data was decided to be left in dataset construction phase.

Raw data is stored in comma separated value (CSV) file format, meaning that every row in the file denotes one sample, and each numerical or textual part separated by comma "," denotes distinct variable value. Different lines can have different amount of variables, because certain value combinations have different feature sets from each other. Below is an example of random part of the raw data:

```
Player1,6711.205571,MOUSE,MouseMovement,(1054.000000,690.000000)
Player1,6711.222199,MOUSE,MouseMovement,(1053.000000,690.000000)
Player1,6711.222772,KEY,KeyDown,0,a,1.900000
Player1,6711.410658,KEY,KeyUp,0,a
Player1,6713.885761,KEY,KeyDown,2,d,1.900000
Player1,6714.060674,KEY,KeyUp,2,d
Player1,6715.047860,KEY,KeyDown,2,d,1.900000
Player1,6715.210535,KEY,KeyUp,2,d
Player1,6716.230251,MOUSE,MouseMovement,(1052.000000,690.000000)
Player1,6716.247258,MOUSE,MouseMovement,(1050.000000,690.000000)
Player1,6716.263466,MOUSE,MouseMovement,(1045.000000,692.000000)
Player1,6716.280761,MOUSE,MouseMovement,(1041.000000,693.000000)
Player1,6716.297123,MOUSE,MouseMovement,(1039.000000,694.000000)
Player1,6716.314197,MOUSE,MouseMovement,(1037.000000,694.000000)
Player1,6716.330303,MOUSE,MouseMovement,(1035.000000,694.000000)
```

Figure 2: Example of the raw data.

First variable of raw data is always the class label: `Player1`, or `Player2`, and it is common to whole dataset. Also the second variable is common for all data, and includes time information in form of a timestamp. Timestamp is continuous time from beginning of the game session, to the end of the session, and gets unique values per player. This way both players have their own (different from each other) timing information, i.e., timing between classes is not directly comparable, but time ratios are.

From variable 3 onwards variables differ quire much sample to sample basis. Variable three has two possibilities, `MOUSE` tells that sample belongs to a mouse-event, and `KEY` correspondingly means keyboard-event. Variable number four tells the sub-type of the event. For mouse-events, `LeftMouseDown` means that left mouse button has been clicked, `LeftMouseUp` means that earlier clicked left mouse button has been released. `RightMouseDown` and `RightMouseUp` can be understood similarly. `MouseMovement` is sign of a general mouse movement, whereas `LeftMouseDragged` and `RightMouseDragged` are mouse movements with either left- or right mouse button down. Keyboard events in variable number four are `KeyDown` and `KeyUp`, which are quite self-explanatory.

Variable 5 for mouse-events are the following: For mouse movement and dragging, it is the coordinates to which mouse has moved, and for clicks, variable tells coordinate location of the click. For keyboard, variable 5 and onward consists of character of the key (e.g., "w", "a", "s", "d", …), information if it is a special button ("Enter", "Tab", "Space", "Backspace", "Esc"), distance information (distance from button "s" in centimeters). Each `KeyDown` and `KeyUp` -event also includes an integer value that is a unique key code value for each specific key. `KeyDown`-events have special variable number eight, which tells that a key is kept pushed down ("Repeat") without release (`KeyUp`-event).

Mouse scroll wheel data was removed from the raw dataset, because only three instances from all *77* 000+ were scroll events. But in larger study it will probably be good data to keep, because in some games, and with some players, scrolling can be much used and distinctive factor.

In following array, the variables from V1 to V8 are listed with all possible feature combinations:

V1	V2	V3	V4	V5	V6	V7	V8
Class	Time	Mouse	Movement	Coordinates	-	-	-
Class	Time	Mouse	Click	Coordinates	-	-	-
Class	Time	Mouse	Click AND Movement	Coordinates			
Class	Time	Key	Key down	Code	Key	Distance	Is repeat?
Class	Time	Key	Key up	Code	Key	-	-

Table 2: Raw data variables and feature combinations.

3.3 Constructing the Dataset

With a granular raw data like one collected in this research, number of possibilities constructing a dataset for analysis is virtually unlimited. For example, with mouse data, dataset could include data based on movements, directions, speed, clicks and click locations, double clicks, scroll wheel usage, etc. Variables then could be for example, frequency and mean length of movements in certain sized time windows, and frequency of right clicks per screen quarter, etc. Only limit that exists with construction of variables is imagination; from simple frequencies to more complex aggregates and even obstruct constructs – everything is possible. But one thing is for sure: there exists a need to do this construction as well as possible, to ensure best available performance in classification.

Furthermore, whole dataset can be constructed either by creating all the variables that comes to mind, and later via feature selection etc., choose best ones for the classification, or, try to figure out good set of features beforehand. Because the focus in this research was to keep it simple, and decision was made to choose a reasonable number of variables that should provide good results. How to choose said variables?

In and of itself, this is easily topic of such extent, that whole study could be written around this exact question.

Having spent considerable time finding solutions to this problem, the only viable outcome seems to be consider application area (e.g., computer game), and try to think features that most probably can be used to differentiate players. After this consideration, features should be merged and fit with ones used in existing research, and has been proved to be working well.

Thinking two players, what kind of differences they could have in their mouse and keyboard behavior? Behaviors related to in-game content should not affect the features too much: for example, if another player plays hasty fighting scene, pushing buttons and clicking mouse like a madman – while other player sits on a riverbank fishing and clicking fishing bobber every fifteen seconds or so. In other words, some in-game play scenarios might include certain type of behavior, but because that is not what we're trying to identify, it must be taken into account when creating the features. Of course situation like this will have some effect on variables, were they virtually anything. But key idea is to not let frequencies and timing have too deep effect (e.g., no feature with "total buttons pushed" or similar frequencies), but instead go deeper into a single event. Mean button push time (and its standard deviation) for example, will not have considerable effect if player pushes button once, or ten, or hundred times. It just measures how player tends to push buttons. This is also the reason why there is no multiple features measuring times between certain events in constructed dataset, e.g., key presses: only one overall "time between all events (mean and standard deviation)", which alone will not have too large effect overall if game scenarios are very different from each other.

Before choosing features for the dataset, it is good idea to take a look for features previously used in research studies. Mondal and Bours (2013, 5) chose their mouse dynamics features based on type of action (mouse movement, point and click etc.), direction of movement, speed, acceleration and distance travelled. Shen et al. (2013, 20) had single- and double-click statistics, movements and movement times, and speed and acceleration information for their mouse dynamics features. Shen et al. (2012, 63) also had mouse dynamics feature set similar to previously mentioned, but feature set

was more granular in nature, e.g., mouse movement speeds and accelerations were constructed per axis basis (x-speed, y-speed, x-acceleration, …). Traore et al. (2012, 141) featured in their keyboard dynamics data average typing speed, and dwell and flight[1] time based variants (mean of dwell time, standard deviation of dwell time, etc.). Ahmed and Traore (2014, 459) had bit different take on features for keystroke-dynamics: they did collect dwell and flight times, and actual key codes representing what letter or special button (such as enter) were pushed.

Features chosen for the dataset should reflect those mentioned in paragraph before, but also must be ensured, that the game content does not have too large effect on results. Best way should be to construct features that captures behavior of a player, but at same time those features should not be too tied to other happenings and behaviors in-game. This kind of features could be, for example, mean time and standard deviation of a mouse clicks that happened in certain segment of raw data. Speed of a mouse click seems to be something, that is inherent for a player, and does not necessarily vary much between different game content. Same thing goes with keystrokes, or means and standard deviations of lengths of mouse movements; different players can have different ways of moving the mouse, and mean of those movements tells how lengthy they are on average, while standard deviation tells how much variation there is in case-to-case basis.

Final consideration before constructing the dataset, was about how many continuous rows in raw data should be used to form one sample to the dataset. Smaller amount of raw data used would cause the final dataset to have more samples. But with smaller amount of raw data, does the features have enough accuracy and information in them to be useful in classification? There is tradeoff between amount of data, and how well the final dataset samples represent the raw data. Tradeoff were solved by trying to construct different sized datasets as long as best possible tradeoff point were found: 400 row in raw data preserved good amount of information, and produced total of 192 samples to the final dataset, which is enough for serious testing and classification.

[1] Dwell time is time between key down and key up event with the same key, i.e., how long button is being pressed. Flight time between two keys (same or different one) is time from last key up event to next key down event.

It is not necessary to construct the dataset from the raw data on row basis as mentioned earlier. Other non-row based methods were also explored, for example time based windowing, or windowing based on certain types of activities. All these got complicated quite fast, and did not, at least not logically, provide necessary independence in relation to in-game content same way as choosing by rows. Choosing 400 consecutive rows ensures, that events are indeed continuous in nature, and random in sense, that no certain activities were strictly put into a sample (i.e., certain activities or patterns might be there fully, or partially, or not at all).

3.4 The Dataset

After thoughtful consideration, testing and consulting existing research, the following feature set was extracted from the raw data:

Feature	**Event type**	**Based on**	**Measures**
Time between events	All	Time	Mean
			Standard deviation
Distance travelled	Mouse	Distance	Mean
			Standard deviation
Distance from default position	Keyboard	Distance	Mean
			Standard deviation
Time for single key operation	Keyboard	Time	Mean
			Standard deviation
Time for single click operation	Mouse	Time	Mean
			Standard deviation
Number of repeats divided by number of samples	Key	Time	Frequency

Table 3: Features of the Final Dataset

Final dataset consists of total eleven features and the twelfth feature that has class labels. Dataset size is now 192 samples, from which to Player1-class belongs 122

samples, and Player2-class is size of 70 samples. Example of four random rows from the final dataset is below:

```
0.043903, 0.071798, 523.822704, 243.776584, 3.850000, 1.341901,
0.223481, 0.458212, 0.108947, 0.016816, 0.010000, 0

0.140439, 0.406236, 506.427858, 265.204638, 2.371667, 1.114632,
1.257845, 5.677072, 0.101036, 0.013979, 0.045000, 0

0.029288, 0.072128, 375.153090, 168.306381, 2.545000, 1.116609,
0.980375, 1.389499, 0.258841, 0.195129, 0.030000, 1

0.034982, 0.092296, 343.245252, 122.328346, 2.147222, 0.923248,
0.865635, 1.765458, 1.180236, 1.312168, 0.160000, 1
```

Figure 3: Four example rows from the final dataset.

First two features are the mean and standard deviation of times between all the events. These two features are most prone to measure game content in addition to measuring the players. For example, fast-paced game content will provide different results in these features than more tranquil content. But when two players play content of similar pace, these two features give more information of reaction times and speed patterns of players. Features three and four are the mean and standard deviation of distance travelled with a mouse. These features measure player's mouse movements, i.e., how long arcs player typically makes with a mouse, and have little to no dependence to game content. Features five and six are similar, but measurement considers keyboard: mean and standard deviation of key distances. All keys have been assigned distance in centimeters from default position, the key "s". This key was chosen to the default position, because default position for left hand in most computer games is with buttons "w", "a", "s", and "d", as these buttons function similar to arrow keys ("w" is forward, "a" is left, and so on). Features measure how long reaching certain player is in keyboard wise, or does she only pushes closest buttons and resort to using mouse to click farther abilities. Features from 7-10 are means and standard deviations of times spent keyboard and mouse buttons down, i.e., how fast players clicks and button presses tend to be. Last feature, number eleven is concerned about frequency of repeats. Repeat is situation, where a keyboard key is kept pushed down longer than it would count to be just a key press.

It is important to iterate, that these features are only one possible combination from uncounted possibilities, and there are two levels of possibilities built on top of each other in feature construction: Every single feature can be chosen to be virtually anything. Possibilities are nearly endless, and only imagination sets limit for what kind of feature is to be constructed. Also, combinations to which of different features that have been constructed can be formed is large. With this plethora of choices, in what is really important phase considering success of classification, choices were made by educated guess with help of 1) existing research, 2) using logic to what kind of features should separate players, 3) feature should measure player, not game or content, and 4) simplicity; keep it simple. With these ideas, the dataset was formed.

In sake of simplicity mentioned, some possible features that could provide good separation between players were left out of feature construction process. Some mouse direction and acceleration related features were deliberately left out to keep process simple. But these are indeed features that should be explored in some other situation.

4 Analysis of the Data

4.1 Preparing the Data

Preparing the data for the analysis was actually quite straightforward process, because the data quality question was accessed in earlier dataset construction phase, i.e., high dataset quality was a byproduct of dataset construction. Also much other work, such as feature selection, and feature extraction were also made in that phase. Because of this, dataset was already quite ready for the analysis, and only some scaling for variables were made in actual data preparation phase.

The scaling method, often referred as standardization or normalization (or z-score standardization, or standard score), can be calculated with formula

$$Z = \frac{X - \mu}{\sigma}$$

where Z is the new standardized variable value (Dodge 2008, 387). X is the current variable value, μ is arithmetic mean of variable vector, and finally σ is the sample standard deviation of the variable vector.

It is important to notice, that while this z-score standardization formula ensures that random variable that is normally distributed $N(\mu, \sigma^2)$ will after standardization follow normal distribution with mean and standard deviation of $N(0,1)$ (Dodge 2008, 387), this is not necessarily the case with non-normally distributed random variable. Distribution of variables in the dataset is unknown, but can be assumed to be non-normal: for example, there are cases of features where standard deviations are larger than the respective means in strictly non-negative scale of values (variables number 6, 8, 9, 10 and 11). This is very strong sign of skewness and non-normality in a data. What is trying to be said here, is that performed z-score standardization does not transform used dataset to be normally distributed, and thus many statistical analysis methods, such as analysis of variance (ANOVA) among others, are not usable with current data.

k-NN algorithm on the contrary does not make any assumptions of normality, and has no problems with skewed data.

Use of z-score standardization for the dataset is justified, because when taking a look to variable values, big differences in value ranges between variables can be observed:

Variable	**Minimum value**	**Maximum value**
V1	0.023305	0.184183
V2	0.035586	0.855367
V3	98.23511	1545.416
V4	43.17527	745.0535
V5	0	6.268
V6	0	7.505553
V7	0	2.240685
V8	0	9.006641
V9	0	5.228984
V10	0	4.473114
V11	0	0.375

Table 4: Value ranges for variables.

Differences in value ranges is especially harmful to k-NN algorithm when using Euclidean distance as distance metric, because if one or more of the values in vectors that works as coordinates in this case in 11-dimensional space, is several orders of magnitudes larger than the others, and will overshadow smaller coordinates and render them nearly useless.

As a side note, it could be mentioned that also rescaling (min-max-normalization)

$$x' = \frac{x - \min x}{\max x - \min x}$$

was explored in the process, but it did produce slightly smaller performance for $\forall k$, than z-score.

4.2 Cross-Validation and Classification

Classification was made with k-Nearest Neighbor algorithm using leave-one-out cross validation. Optimal k-value was searched repeating cross-validation process with different k-values $k = 1, 2, \ldots, n$, where n is size of the dataset. Best performing number of neighbors was $k = 3$, which was then chosen to be the parameter value for classification model constructed.

Decision to use leave-one-out cross-validation was based on fact that dataset is not quite large in size, and idea of reaching smaller bias with probably higher variance seemed to be good approach. Variance of cross-validation results were observed just in case by randomly sampling partial datasets for cross-validation, and comparing the results. There was some variation, but not enough to raise concerns about suitability of leave-one-out cross-validation.

Also effects of z-score normalization was tested, by running cross-validation with data without z-score normalization, and then with normalization. Performance differences were remarkable, even tens of percentage points better results for z-score normalized data. Also min-max-normalized data was tested, but it performed couple of percentage points worse than z-score over all k-values.

Cross-validation and classification of the data was quite fast and straightforward process, as most of the work with data and k-NN implementation were done in earlier phases. Both k-NN regression with concordance index performance measure (AUC) and regular k-NN classification with accuracy performance measure (ACC) were performed, and results were comparable. Results obtained with self-implemented software were also compared to tested and trustworthy existing software, R-program and its built-in functions, and results were exactly the same.

4.3 Constructing a "Model"

When considering how to construct actual model for player classification with the dataset, two different schemes comes up. One possibility would be construct a model that get new sample, that is known to be produced by either Player1 or Player2, and model then classifies and predicts to which of these classes the sample belongs. This model would probably be best, when constructed by randomly selecting 70 samples from Player1-class and choosing all 70 samples from Player2-class. Doing so should reduce bias towards classifying new samples to Player1-class, which is larger in size. Another variation could be, that instead of random selection, 70 somehow measured as best samples from Player1-class gets chosen. But this approach will reduce classifiers ability predict Player1-class in future: if certain types of samples were to be pruned systematically away from the dataset, consequences are most probably not predictable. Third, and probably most robust way would be to use full data (122 + 70) and take possible bias into account when classifying new samples.

Another possible scheme with the current dataset could be the case, where new data sample is compared to full data of Player1-class, and model classifies and predicts if new sample belongs to Player1-class or not. In this case model could be used to continuous authentication for Player1-class.

Term "model" may be little misleading in case of k-NN, as there is no actual learning happening in k-NN training phase, and there is no real model structured; the model in k-NN is the training data which acts as a lookup table for testing data values (Abbot 2014, 254).

5 Results of the Analysis

5.1 Results and Performance

Analysis with k-Nearest Neighbor classifier, leave-one-out cross-validation and k-value three, yielded the following results:

		True class	
		Player1	Player2
Predicted class	Player1	115	12
	Player2	7	58

Table 5: Confusion matrix for the analysis results.

From now on, Player1-class should be seen as a positive condition, and Player2-class as negative a condition. Confusion matrix above shows how classes are predicted in classification: 115 from 122 classes are correctly predicted to belong to Player1-class, and 7 out of 122 are falsely predicted to belong to Player2-class. Similarly, from Player2-class 58 samples out of 70 is predicted correctly, and 12 samples out of 70 is predicted to falsely belong to Player1-class.

False positive rate (false acceptance rate):

$$FAR = \frac{12}{70} \approx 0.171$$

False negative rate (false rejection rate):

$$FRR = \frac{7}{122} \approx 0.057$$

Overall accuracy of the classifier:

$$ACC = \frac{115 + 58}{192} \approx 0.901$$

Error rate of the classifier:

$$ERR = 1 - ACC = \frac{7 + 12}{192} \approx 0.099$$

Overall performance for classifier is 0.901 (ACC), and classifier error rate similarly 0.099 (ERR), i.e., 90.1% accuracy and 9.9% error rate. False acceptance rate is 0.171 is 17.1% (FAR) and false rejection rate 0.057 is 5.7% (FRR), as these are often reported as percentage values:

Classifier Accuracy (ACC)	Classifier Error rate (ERR)	False acceptance rate (FAR)	False rejection rate (FRR)
90.1%	**9.9%**	**17.1%**	**5.7%**

Table 6: Results.

5.2 Analysis of the Results

The results described in chapter 5.1 are promising. Overall accuracy slightly over 0.90, while FAR and FRR being 0.171 and 0.057 respectively gives strong evidence, that there indeed exists a signal in the data collected. This means that Player2 has a probability of 17.1% to be falsely classified as Player1, and Player1 has a probability of 5.7% to be falsely classified as Player2. In other words, probability that Player1 gets classified correctly is 94,3%, and Player2 gets correct classification with probability of 82,9%. This difference in correctly classified cases between players could partially be caused by imbalance in number of samples for classes, and some more data from Player2 could help ease the situation a bit. Now there is bias towards Player1-class, because it is over-represented in the model when comparing to Player2-class. Still, results are good and comparable with existing research, where for example Shen et al. (2013, 24) got FAR 15.67% and FRR 14.53% from 3-Nearest Neighbor, although here must be remembered that their average authentication time was quite low, only 11.8 seconds.

There should not be effects of the curse-of-dimensionality in results, because the dataset had moderate dimensionality of 11 features. Bias-variance tradeoff can be seen the following way: k-value three means, that bias should be moderately low, as choosing three closest samples is still quite close to training data, model wise. Variance on contrary is higher, because only three nearest neighbors are evaluated. Small bias and high variance means, that patterns in training data are accurately captured, but generalization to unseen data can be somewhat problematic, i.e., there is risk of over-fitting.

The research question for the thesis was: Can two players, playing the same computer game, be distinguished from each other based on differences of how they use mouse and keyboard? Based on research made, and results produced, the answer is yes. Two players indeed can be distinguished from each other with probability of 0.901. Probability is high enough, that answer to the research question is trustworthy, and no real doubt is left that the contrary would be true. 9.9% error rate of the classifier of course means, that not every sample is correctly classified, and recognized to belong to true class. But with large enough authentication time, i.e., with enough samples, the class membership for player who produced data can be applied with high confidence.

The answer to the research question itself already includes the answer to the hypothesis:

$$H_0 : there\ is\ no\ difference\ between\ players$$
$$H_v : there\ is\ statistically\ significant\ difference\ between\ players$$

But it can be viewed also another way; a little thought experiment. By following the definition of p-value, we could in this case ask a question: What is the probability that 0.901 classification accuracy is reached by coincidence? This probability is low. If there is no difference between players, classification results are quite stubbornly around 0.50 (random classification; although we have bit imbalanced classes). Based on this logic, we can reject the null hypothesis H_0 and conclude that the alternative hypothesis holds.

Quite large difference between false acceptance rate (17.1%) and false rejection rate (5.7%) is interesting. This means that probability of Player2-class samples to be accepted (classified) as Player1-class is 17.1%, and probability of Player1-class samples to be falsely classified to Player2-class (rejected) is 5.7%. Difference is probably based on bias towards Player1-class, that has lots more samples. More samples from Player1-class means that there is more density for Player1-class in the model, and thus probability of those being neighbors to Player2-class test sample is higher. Same way Player1-class samples have a smaller probability to be falsely classified as Player2-class, because neighboring samples in the model have higher probability to belong to class Player1. Even though this bias exists, FAR and FRR are not alarmingly high, and overall results are quite trustworthy.

When considering the results in a larger scope, for example the EU has certain standards for biometric authentication. The EU standard for commercial biometric technology requires 0.001% false acceptance rate together with one percent false rejection rate (Shen et al. 2013, 16). Results achieved are far from those, as are many other research studies made mentioned in chapter 2.4. This shows that despite the promising results, research in the matter is still in early stages, and no solutions to real world usage can yet to be made based on mouse and keyboard dynamics.

Comparing results based on behavioral biometrics to biometrics based on physiological features, such as fingerprint scanners that are able to reach needed standardization in the EU, behavioral characteristics are less successful. Reason for this could be, that physiological features are very stable, and they normally do not vary much over time. Behaviors on the other hand can vary quite much. For example, stress or illness can change behavioral characteristics, and behaviors also vary from time to time without any apparent reason. (Gunetti & Picardi 2005, 312.)

Additional limitations to interpretation of these results will arise, if these methods used would be taken out of safe experimental context, and set out in the wild, i.e., real world use. Then additional variability caused by different environments must be taken into account, and it probably will have more or less negative impact to classification performance. Causes of real world-variability are for example, hardware-level factors (different keyboards, mice, computer types), software-level factors (screen resolutions,

operating systems, speed and sensitivity settings for mice and keyboards), environmental factors (different setups of mice, keyboards, screens, tables, chairs), and physiological or psychological states of users, such as fatigue, distraction, stress (Shen et al. 2013, 28). One could also argue, that because data is from certain user, and this certain user often has only one computer she plays on, then those factors mentioned are mingled in the data such way, that they become part of mouse and keyboard dynamics identity for that user. Change in this established environment then can cause uncertainty in classification.

6 Conclusions

6.1 Meaning of the Results

Mouse and keyboard dynamics is relatively new field of biometric research, that can be used to authenticate users of a computing system based on their mouse and keyboard usage behaviors. Although it does not yet provide results comparable to biometrics based on physical features of user, research done in the matter does indeed provide promising visions for future. It also offers possibility to continuous authentication, that fingerprint scanners or plain passwords does not. In this research, possibility of differentiating two computer game players was explored, and many of the topics and challenges related to mouse and keyboard dynamics present in research were considered. Feature construction for analysis were probably the most challenging and time consuming process that were encountered, because of endless possibilities for features and feature sets to be constructed. Feature construction were also challenging because certain possible feature-types are not suitable for computer games (e.g., those that measures the game or in-game content instead of player).

Positive outcome to the research question gives hopeful tone to possible future exploration in field of mouse and keyboard dynamics in computer games, as it is now shown, that mouse and keyboard dynamics methods used in other application areas can be applied to computer games as well. Overall results of the research, and the fact that results were in line with existing research in related fields gives a good signal that, if studied more with more general research setting, approach proposed in this thesis could lead to an actual methods implemented in real world usage scenarios. This research should not be seen as complete in a sense that there is still a lot of questions waiting for answer, e.g., how well these preliminary results reached will generalize to larger population, and where are the limits of biometric authentication based on mouse and keyboard dynamics in computer games; could every player be provided with biometric fingerprint based on dynamics data, that is unique and able to be identified? These are hard questions, and unfortunately out of the reach for the thesis.

Mouse and keyboard dynamics in any application area is not yet a robust method, or some magical answer to life, the universe, and everything. These dynamics in their existing stance are merely an addition for providing an extra layer of security to authentication in cases of dire need. Unfortunately, almost the whole networked world would seemingly actually need all the security that can be provided. And from this standpoint, it might not be a bad idea to explore these ideas further.

Based on results achieved, it is possible to take a look for possibilities mouse and keyboard dynamics in computer games could provide. These are the continuous player authentication, and possibly even bot detection. Chapters 6.3-6.5 tries to explain why these matters should be studied more in future, and chapter 6.6 explores some ideas for possible future research. But before that, some of shortcomings and proposed improvements to the research just conducted should be reflected.

6.2 Improvements

After the raw dataset of size 77 000+ samples were collected, it indeed felt that there were plenty of data for analysis. What I did not took into account then, was how much the amount of data shrunk during the dataset construction phase. Drop from 77 000 samples to 192 was far more than I predicted. It would have been helpful to have more data, for example final dataset of size 1000 would have given much more possibilities for sampling etc.

Another thing that would have been very useful, is couple more test subjects. Then one could have been chosen as the "owner of game account", and other players could have been seen as "adversaries" trying to authenticate as owner. More test subject would also provide more behaviors. Classification could have been done in two different ways; binary (owner/not owner), and multi-class (Player1, Player2, Player3, …).

Third thing that could have been done otherwise were the methods used in analysis. k-Nearest Neighbor algorithm did provide successful results, but maybe some other method could have performed better. Many research studies in mouse and keyboard dynamics often uses support vector machines for classification tasks. k-NN was perfect

for this research in a sense that it was easy to understand, easy to set up and performed in a sufficient level, thus releasing more resources to other topics, namely feature construction.

6.3 Security and Privacy Aspects

Collection of user's mouse and keyboard data can potentially be danger to the information security and privacy. While this seems like a real problem, other kind of view to the matter should be taken. When player plays a computer game, operating system gives mouse and keyboard commands directly to the game, i.e., game already has access to mouse and keyboard information, and uses them to move the game avatar etc. This same information inside the game can be used for continuous authentication based on mouse and keyboard dynamics, and there is no need for data collection to be done outside the game. As long as data collection indeed happens inside game program, no other input data apart from those given to game can be collected. This is enforced by operating system; inputs go only to the program that is in active state regarding the mouse and keyboard inputs. For example, game that collect the dynamics data, cannot collect what user types into web-browser despite the fact that game is open in the background and is collecting the dynamics data. This is very basic operating system security feature that is present in every modern operating system. These rules of course can be loosened from operating system's settings to gain direct access to all input data from all sources coming into computer, but these changes need to be approved by computer's owner (this is how the data for this research was collected).

Here, when following the logic, we're in situation, where input would anyway get into the game. And, as is the case with any game, if user does not want to give her inputs to game, then she can't play it anyway. Now it is matter of collecting and sending the data forward, because game client in a computer must be seen as it resides in a possibly hostile environment, as of yet we cannot be sure that player is who she says she is. A game client at this moment can be in hacker's computer, who has full control on game client and its network traffic. This is the reason why data collected cannot be analyzed inside game client, and possible performance issues caused by the analysis to an honest player are also important factor here. Collected data must be sent to the game

company's servers for analysis. If data would be analyzed inside game client, hacker could halt game client's attempt to send negative results to the game company, and send "all is OK here" -message instead.

Sending raw mouse and keyboard dynamics data over the network has minor security and privacy issue even if it is encrypted with SSL or other technology. Player might, for example have chat with a friend in the game, and does not want to think that those letters typed in conversation end up collected by the game company. Key idea is to send constructed dataset instead. As can be seen from the dataset formed for this thesis, the raw data cannot be re-constructed based on it; dataset construction is from this perspective a one-way function. When constructed dataset has features such as how long was average key press (any key, because keys were not specified in the dataset), there is no real threat to private information leakage.

As seen in previous paragraphs, there exists no real privacy nor security risks for player. It is all different story, how this idea can be sold to the player without raising concerns of trust and privacy, but it is marketing and out of the scope of this thesis.

6.4 Circumventing the Continuous Authentication

Possibility of circumvent the continuous authentication based on mouse and keyboard dynamics is also an important topic that should be discussed. In online games such as World of Warcraft, avatar (player controlled game character) movements are send to the game server in quite high frequency, so that the game server knows location and movement direction of the avatar, and can redistribute the data to other players nearby. It is important to notice, that the actual movement commands from players input-devices are not send, only state information of the avatar. Game client itself only takes mouse and keyboard inputs, move the avatar based on those (and restrictions forced by the game, e.g., walls and fences), and send the avatar state to the game server.

With mouse and keyboard dynamics based continuous authentication, it is necessary send also the mouse and keyboard dynamics data to the game company's servers. As discussed earlier, the analysis cannot be made in player's computer (a hacker with

stolen game account), or the analysis can cause performance troubles for player conforming to the rules. Now again, a minor problem has risen: continuously sending extraneous information via network that can possibly be very slow, can cause headache to the player. For this reason, data should only be collected and sent when suspicion has risen, and company's server sends a request to the game client to conduct the collection. Approach can be seen as a partial continuous authentication, that is an extra security measure in situation of need.

If a hacker, when noticing that data collection is asked, sends her own input data, there is risk of getting caught. Her dynamics does not necessarily match ones collected earlier from the account owner. Could hacker have in her possession some amount of mouse and keyboard dynamics data, that is somehow collected from accounts legitimate or previous owner? This is very unlikely, but always possible. Hacker could send requested amount of that data to game company, and all is clear. What have to be done now, is that in game company must record their view (server side) of suspected accounts game avatar movements for same exact time period that the request for dynamics data was made. Comparing these two, can be determined if only a dataset with no realization in game world was sent. Hacker could of course make the game avatar move similarly as in some random segment of stole user dynamics data from account's owner and get clean this way, but it will render game character useless, as it only does some random previously recorded movements, and thus defeats hacker's goals to do anything productive with it.

6.5 Continuous Authentication in Game Bot Detection

Similar methods as previously mentioned could possibly be used in game bot detection. Idea can be based on the fact that game bots do not produce mouse and keyboard dynamics data. Game bots are often moving avatar by directly writing over game's memory-space and this way producing actions and movements. In case of game bots, bot user can record his own mouse and keyboard dynamics as much as she wants, and send her previous data when asked. Because data sent must match with observed avatar movements, bot must obey movements in the dynamics data, and thus reducing its usefulness. Because game worlds are dynamic, i.e., once collected mineral vein does

not necessarily later re-spawn in exact same location, it is hard to predict could a bot and dynamics data be constructed such way, that it stays undetected at the same time that it actually can do something useful in-game (the gold farming etc.). There is also quite much effort for bot user to play the game to produce the dynamics data needed together with the bot (data cannot be produced synthetically, as it would not include dynamics the player has). In situation like this, it is probably easier to just play the game, and do farming by yourself.

This idea of human generated data versus bot generated "synthetic" data has been explored in research last couple of years. Most often research is based on behaviors of game avatar, such as avatar trajectory analysis, where movements visible to server-side are analyzed and bots are detected by certain movement patterns that differ from those of humans. Mouse and keyboard dynamics would take research one step lower level, where the actual data that creates the movements that are then realized in-game is researched.

6.6 Future Research

The idea behind this research was to investigate the possibility of doing continuous authentication in computer games based mouse and keyboard dynamics. Now, as seen in the thesis, idea seems to be possible with good confidence. How well this task could be done in future research if enough time and resources were to put into effort?

Main improvements I would like to see to be made to research would be: More test subjects in controlled environment (computer classroom or similar), where players would play multiple gaming sessions in several distinct sessions. Amount of players could range somewhere around 30-50, with all kinds of different gaming backgrounds (or lack of gaming experience for that matter). Even more effort should be put into dataset construction and choosing best possible features, and support vector machine or other more sophisticated classifier should be used.

Focus in such research could be two-fold: Easier version would be to learn better how to distinguish owner of the game account from others, based on mouse and keyboard

dynamics data collected from game account owner's previous gaming sessions. More challenging task would be identifying certain player based only on previously collected dynamics data. Based on existing knowledge, the first task seems to be quite possible. Second one raises more questions. It could be possible, if biometrics based on mouse and keyboard dynamics really differ between players, and are such inherent features for player that separation can be made. But when thinking the fact that there are tens of millions online game players around the world, differences between some of them must be miniscule.

References

Research Articles and Conference Papers

Ahmed A. Ahmed & Issa Traore (2014) "Biometric Recognition Based on Free-Text Keystroke Dynamics", *IEEE Transaction on Cybernetics*, vol. 44, no. 4, April 2014, pp. 458-472.

Friedman, Jerome H. (1997) "On bias, variance, 0/1-loss, and the curse-of-dimensionality." *Data mining and knowledge discovery*, vol. 1, no. 1, pp. 55-77.

Gunetti Daniele & Claudia Picardi (2005) "Keystroke Analysis of Free Text." *ACM Transaction on Information and System Security*, vol. 8, no. 3, August 2005, pp. 312-347.

Kotsiantis, S. B., D. Kanellopoulos & P. E. Pintelas (2006) "Data Preprocessing for Supervised Learning." *International Journal of Computer Science*, vol. 1, no. 2, pp. 111-117.

Mondal, Soumik & Patrick Bours (2013) "Continuous Authentication Using Mouse Dynamics." *BIOSIG 2013 Proceedings of the 12th International Conference of the Biometrics Special Interest Group*, September 2013.

Pao, Hsing-Kuo, Kuan-Ta Chen & Hong-Chung Chang (2010) "Game Bot Detection via Avatar Trajecotry Analysis." *IEEE Transaction on Computational Intelligence and AI in Games*, vol. 2, no. 3, September 2010, pp. 162-175.

Robles, Rosslin John, Sang-Soo Yeo, Young-Deuk Moon, Gilcheol Park & Seoksoo Kim (2008) "Online Games and Security Issues", *Second International Conference on Future Generation Communication and Networking, FGCN'08*, pp. 145-148.

Rybnik, Mariusz, Marek Tabedzki, Marcind Adamski & Khalid Saeed (2013) "An Exploration of Keystroke Dynamics Authentication using Non-fixed Text of Various Length." *2013 International Conference on Biometrics and Kansei Engineering (ICBAKE)*, pp. 245-250. IEEE.

Shen, Chao, Zhongmin Cai, Xiaohong Guan, Youtian Du & Roy A. Maxion (2013) "User Authentication Through Mouse Dynamics." *IEEE Transaction on Information Forensics and Security*, vol. 8, no. 1, pp. 16-30, January 2013.

Shen, Chao, Roay A. Maxion, Guang Xiang & Xiaohong Guan (2012) "Comparing Classification Algorithm for Mouse Dynamics Based User Identification." *2012 IEEE Fifth International Conference on Biometrics: Theory, Applications and Systems (BTAS),* pp. 61-66. IEEE.

Short Robert D. & Keinosuke Fukunaga (1981) "The Optimal Distance Measure for Nearest Neighbor Classification." *IEEE Transactions on Information Theory*, vol. 27, no. 5, pp. 622-627.

Traore, Issa, Isaac Woungang, Mohammad S. Obaidat, Youssef Nakkabi & Iris Lai (2012) "Combining Mouse and Keystroke Dynamics Biometrics for Risk Based Authentication in Web Environments." *2012 Fourth International Conference on Digital Home (ICDH)*, pp. 138-145. IEEE.

Wirth, Rüdiger & Jochen Hipp (2000) "CRISP-DM: Towards a standard process model for data mining." *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*.

Original Sources

Blizzard Entertainment (2012) "World of Warcraft Terms of Use Agreement." Online resource. Referenced 20.4.2016. Available: http://eu.blizzard.com/en-gb/company/legal/wow_tou.html

Imperva (2014) "Comsumer Password Worst Practices." Online resource. Referred 20.4.2016. Available: http://www.imperva.com/docs/wp_consumer_password_worst_practices.pdf

National Institute of Standards and Technology (2011) "National Strategy for Trusted Identities in Cyberspace: Why We Need It." Online resource. Referenced 20.4.2016. Available: http://www.nist.gov/nstic/NSTIC-Why-We-Need-It.pdf

Textbooks

Abbot, Dean (2014) *Applied Predictive Analytics: Principles and Techniques for the Professional Data Analyst*. Indianapolis: John Wiley & Sons.

Berthold, Michael & David J. Hand (2007) *Intelligent Data Analysis, An Introduction*, 2nd ed. Heidelberg: Springer.

Bishop, Christopher M. (2000) Neural Networks for Pattern Recognition. Midsomer Norton: Oxford University Press.

Dodge, Yadolah (2008) *The Concise Encyclopedia of Statistics*. Heidelberg: Springer.

Hastie, Trevor, Robert Tibshirani & Jerome Friedman (2009) *The Elements of Statistical Learning, Data Mining, Inference, and Prediction*, 2nd ed. Springer.

James, Gareth, Daniela Witten, Trevor Hastie & Robert Tibshirani (2014) *An Introduction to Statistical Learning, with Applications in R*. New York: Springer.

Penrose, Roger (2005*) The Road to Reality, A Complete Guide to the Laws of the Universe*. London: Vintage.

Pyle, Dorian (1999) *Data Preparation for Data Mining*. San Francisco: Morgan Kaufmann Publishers.

Smed, Jouni & Harri Hakonen (2006) *Algorithms and Networking for Computer Games*. Chichester: John Wiley & Sons.

Tanenbaum, Andrew S. (2009) *Modern Operating Systems*, 3rd ed. New Jersey: Pearson Education.

Tables and Figures