Università degli Studi di Padova

Dipartimento di Matematica "Tullio Levi-Civita"

Master's Degree in
Cybersecurity

Video Gamers Profiling Based on Controller Usage and Behavioral Data

Supervisor:
Prof. Mauro Conti

Graduating:
Riccardo Zin
2029016

Co-supervisor:
Pier Paolo Tricomi

Academic Year 2022/2023

Abstract

This master's thesis explores the profiling of video gamers based on data obtained exclusively from the controllers they use during gameplay. The study focuses on four popular games: Battlefield V, PUBG, Elden Ring, and Dark Souls 3, representing two distinct genres: first-person shooters (FPS) and role-playing games (RPG). The objective is to investigate the viability of using data gathered from two different controllers, the Nintendo Switch Pro Controller and the DualShock 4, to train an AI model utilizing machine learning algorithms. The experimental procedure involved participants playing each game for a duration of 15 minutes using each controller, resulting in a total of 30 minutes of gameplay per game. Throughout the gameplay sessions, data from various input elements such as buttons, triggers, and sensors was collected to construct the training dataset for the AI model. The central objective of this research is to classify video gamers in six specific scenarios. These scenarios involve training and testing the AI model using various combinations of games and controllers. Initially, the experiments are conducted by training and testing on a single game. Subsequently, the process is repeated with training on one game and testing on another game within the same genre. And finally, the experiments involve training on one game and testing on another game from a different genre. In all cases, the experiments are executed first using one controller for both training and testing, and then by utilizing one controller for training and another for testing. We conducted the experiments with 23 participants, collecting almost 50 hours worth of data. The highest results were obtained in the most correlated scenario which reached an F1-score as high as 87%. The outcomes have implications for personalized gaming experiences, adaptive difficulty adjustments, and targeted game recommendations. Furthermore, the study will provide insights into the effectiveness of employing controller data to differentiate player behaviors and preferences within specific genres as well as across different genres.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years, the video game industry has become one of the biggest in the economic world. In 2022, the global gaming market size reached USD 202.7 Billion and is expected to grow by 9.03% in the next five years [1]. The market has been driven by various factors, including the widespread use of smart devices, easy access to free-to-play online games, and the rising popularity of competitive multiplayer gaming events, i.e., Electronic Sports (esports) tournaments. In particular, esports contests are streamed online and supported by corporate sponsorships as they count millions of enthusiastic spectators worldwide. To quantify their impact, we report that they produced more than 1.38 billion in revenue and totaled 532 million viewers in 2022 [2]. Gaming includes the act of playing video games on dedicated gaming consoles, PCs, or smartphones. It offers entertainment and relaxation through mental and physical stimulation for players and spectators alike [3]. Moreover, it also involves competitive tournaments streamed online (one of the most famous platforms is "Twitch"[1]) and supported by corporate sponsorships [4]. Hence, internet connectivity is crucial for players' communication and streaming gaming events.

Gaming does not only involve a single platform, as it includes the act of playing video games on dedicated gaming consoles, PCs, or smartphones. It offers a vast choice in both the physical means and kind of entertainment proposed when playing. Indeed, a wide variety of genres are available, including adventure, role-playing, puzzles, social, strategic, and simulation games, which feature high-quality visuals and captivating storytelling to engage and entertain gamers [5]. Gaming can be categorized as casual, hardcore, or professional, with individuals pursuing careers in professional gaming and earning income through competitions.

[1]`https://www.twitch.tv/`

Gaming serves purposes beyond entertainment, such as socializing, skill development, and educational supplementation. It improves response time, encourages teamwork, enhances visual memory, and fosters critical thinking. Additionally, gaming can enhance manual dexterity and multitasking abilities, increasing global demand for gaming experiences [6].

When the COVID-19 pandemic struck, it substantially changed people's daily lives worldwide, restricting their ability to socialize and venture outside. This resulted in a sense of boredom and limited travel opportunities due to lockdown measures. Therefore, individuals started spending more time at their computers, either for work-related reasons or seeking entertainment, leading to an increase in the number of online scams and crimes [7, 8]. This situation also profoundly impacted the gaming community, suddenly becoming more numerous and active. Indeed, gaming provided a captivating diversion for those desiring social interaction from the comfort of their homes, guiding to a significant increase in the demand for games [9]. However, this situation created numerous opportunities for fraudulent activities versus gamers. In-game purchases have become increasingly prevalent, where players can usually buy loot-boxes, liveries (also called skins) for their characters and equipment, and downloadable contents (DLCs). Thus, users' payment information is easily accessible through their accounts. Consequently, hackers primarily target account takeovers to exploit this situation. According to the Global Digital Fraud Trends Report Worldwide, the rate of suspected digital fraud in the gaming industry, which includes video games and online role-playing, experienced a significant increase of 68.6% between 2019 and 2021. The following year, from 2020 to 2021, the growth rate was slightly lower at 32.6% [10]. One of the strategies employed by scammers against innocent players is to add them to their friend list after a game played together, engage in conversation, and either send harmful links or propose advantageous trades. While scammers can be reported and banned, they can easily create new accounts and persist with malicious activities.

Scammers and hackers are not the only problems that affect the gaming community. One other major illicit behavior is cyberbullying. The term comprises hate speech, racism, toxic chats and other forms of written demeanors. Over half of the global gaming community, comprising more than three billion active players out of a population of nearly eight billion, have participated in some manner of cyberbullying or online harassment. This indicates that approximately 20% of individuals worldwide have engaged in these detrimental behaviors at least once in their lives [11]. One possible solution to address these issues is to develop a method to identify players based on their unique play-style. This particular solution would be able to locate gamers regardless of the account or game they are currently playing in. Then, if the profiling is accurate, we could find malicious users even if they create other accounts, effectively making this solution biometric.

Because of the broad variety of data that circulates around games and gamers, we focus on one specific type: controller's data. The way people hold and use their controllers while gaming could be unique to themselves, whether is the frequency with which they press buttons, the position of the controller in their hands, or the subconsciously sudden movements they do while trying to make certain actions in-game. Therefore, we decided to use this kind of data due to its possible uniqueness and its transversal nature across consoles and games.

Contributions In this thesis, we are the first to study the possibility of profiling gamers by solely employing data extracted from their controllers. Hopefully, this will open up paths to new research horizons ranging from biometric authentication, malicious intent counteractions, and gamers' safety enhancement.

Organization Chapter 2 overviews related work, then Chapter 3 provides a background to the gaming world, the cybersecurity related topics, and the machine learning techniques used to elaborate the data. Chapter 4 explains the data collection procedure, Chapter 5 describes the profiling aspect of the project. In the end, Chapter 6 regards discussions and Chapter 7 the final remarks.

Chapter 2

Related Works

This section describes the importance of security and privacy in video games. We explore three distinct aspects within this domain. We present the connections between video games and three key areas: privacy, profiling, and machine learning.

2.1 Privacy

With a vast user base spanning various age demographics and income brackets, video games have emerged as the foremost global entertainment sector. Beneath the enjoyable veneer they offer, the fact that contemporary gaming devices pose a substantial risk to consumer privacy often escapes widespread attention.

Referencing patents and literature across various fields, Kröger et al. in [12] also delve into how patterns and associations within accumulated gameplay data might inadvertently reveal supplementary information that is not readily grasped or predicted by the user. This envelops deductions related to a user's biometric identity, age, gender, emotions, skills, interests, consumption habits, and personality traits. There are many projects that study the correlation between video games and gamers' personalities like [13] which explores these relationships with the famous MOBA (Multiplayer Online Battle Arena) called "League of Legends". A similar approach is taken by [14] where they evaluate personality traits from the game "Guild Wars". In this large literature of personal traits correlation with games there are also cybersecurity threats like in [15] where Tricomi et al. propose an attribute inference attack on players of "DOTA 2".

Modern gaming systems log interactions in time-stamped files, creating a history of user actions and in-game events. This covers attributes like action duration, frequency, strength, and accuracy. Gameplay data includes actions, progression, puzzle-solving, dishonest behavior, interactions, purchases, successes, failures, and settings like map choice or difficulty. Sensors now capture voice, gestures, heart rate, and more. Gaming systems gather hardware and software details, often using tracking technologies like tags or cookies. Games request access to other app data or social profiles. Temporal patterns (logins, session duration, frequency), game preferences, genre choices, and features (multiplayer/single-player) are also recorded. Speaking of sensors, a similar study to ours, which involves applying artificial intelligence to sensor data for player profiling, can be found in the work by Tricomi et al. [16]. In their research, they utilized machine learning on behavioral data to distinguish users and predict their individual characteristics. The study examined eleven common actions with varying levels of mental load, drawn from two distinct scenarios: an augmented reality (AR) everyday application and a virtual reality (VR) robot teleoperation. Their results showed that they could identify users with an F1-score of 97% in VR and 80% in AR. Regarding profiling, they successfully inferred gender in VR with an F1-score of 82% and age with an F1-score of 90%.

In [17], Newman et al. demonstrated the methods employed by companies to accumulate player data from gaming consoles using diverse sensors. Notably, player attributes like voice, physical appearance, geographical location, and social network details are commonly gathered. The authors further argued that in games characterized by extensive player interaction with the game environment or fellow players, such as conversations or decision-making, these data types are captured and processed to formulate a player's psychographic insights. Such insights can be harnessed for crafting personalized gaming experiences, real-time adjustments of in-game difficulty, or augmenting gameplay mechanics to sustain player engagement.

2.2 Profiling

In terms of profiling gamers, those who deal with this argument usually rely on statistical analysis of users' account data. In [18], Baumann et al. uncover distinct behavioral classifications among dedicated gamers by employing an unsupervised machine learning technique. Then, they extract individual profiles within these identified categories, such as those who blend strategy-action games or frequently switch between games. Other studies like [19] approach profiling from a behavioral standpoint, with this case specific to the female genre and the psychological effect of video games and the gamer community on mental health. The study proposed in [20] by Williams et al. presents a profiling framework that considers demographic, psychometric, and psychographic aspects. In addition, Conti and Tricomi [21] demonstrated that is possible to recognize players based on their in-game data (with very high accuracy) and acknowledge the potential benefits and risks associated with this approach, including privacy concerns and the possibility of it being exploited for harmful purposes [22].

Another means of use for machine learning in this field is to profile gamers like in [23], where Gosztonyi studies adult gamers in a remote region of Eastern Europe. These findings indicate that in a semi-peripheral country, the proportion of adults engaged in video gaming closely resembles that of central countries.

2.3 Machine Learning

Artificial intelligence has been used in the video game context for quite some time now. Its application differentiates between varied objectives, such as creating NPCs/adversaries in strategy games, recommendation systems, dynamic challenges for players, victory predictions and even pursuing demographic studies outside of the games. Focusing on recommender systems, in [24], De Simone et al. use machine learning to propose a novel approach to design a recommender system for video games. This method relies on profiling the player's in-game behavior and utilizes a new classification system for game activities. These recommender systems aim to suggest a customized collection of additional items or products to users, which are likely to align with their preferences and interests. A similar approach to model a recommender system to buy items in-game has been studied by [25] where they employ machine learning for studying in-game data produced by players, to predict the rating of an item or product for a particular gamer. A very similar procedure is described also in [26] for "DOTA 2".

To make another example, studies like [27], use a data-driven method for identifying patterns in combat which lead to successful game outcomes.

Chapter 3

Background

3.1 Cybersecurity & gaming

A gaming software often handles sensitive personal information and facilitates transactions involving real money or cryptocurrencies for in-game items. This makes gamers attractive targets for hackers seeking to steal such valuable data. These malicious individuals employ various methods to intercept and exploit the player's information, either by selling it online or diverting transactions to their own accounts. Some hackers may focus on finding and exploiting security vulnerabilities within gaming systems, aiming to disrupt gameplay. These disruptions not only cause inconvenience for players but also result in financial losses and harm the reputation of game developers or companies.

To counter these threats, robust cybersecurity protocols are essential. These protocols help prevent data and currency theft associated with in-game transactions, thwart attacks on gaming software, and protect users' devices from malware infections. By implementing effective security measures, the gaming industry can safeguard sensitive information, maintain the integrity of in-game transactions, and ensure a safe and enjoyable gaming experience for players.

3.2 Malicious players

Cyber threats in the gaming world can manifest in various forms, targeting different aspects of gaming software and exploiting vulnerabilities. Below are some common cyber threats that gamers should be aware of:

- **Cheats**. Cheating can occur in various forms, but in every case, the cheater seeks to gain an unfair advantage that is exclusive to them, often at the expense of other players or the intended gameplay experience of the game. Cheating in games can take on various forms, each tailored to exploit specific aspects of the game. For instance, players may cheat by enhancing their own character's abilities, speed, or strength compared to the game or other players, using techniques like speed hacking (a cheating tool that modifies the game's internal clock speed, allowing characters to move much faster than intended) or memory editing (scanning the game's memory to change stored values and achieve almost any cheat). Some cheats involve acquiring in-game skills, assets, ammunition, or currency without making legitimate purchases, bypassing in-app payment systems. Automated tools such as auto-clickers, click bots, or auto macros may provide players with superhuman advantages. Cheaters may also utilize methods to progress in the game without investing the required time and effort, such as modifying the game's internal concept of time or paying third-party companies to farm resources for them [28]. In some cases, cheaters aim to disable license checkers to access premium versions of the game for free, or simply disable in-game advertising. The tactics employed by cheaters are diverse and reflect their desire to gain an unfair advantage or manipulate the game to suit their preferences.

- **Mods**. A mod, short for "modification", refers to changes made by players to a video game that alter various aspects of the game's functionality or behavior. This can involve modifying game values to gain an unfair advantage. Mods are a form of cheating, and within this category, there are numerous different types of modding. Game mods can range from small additions of new features to extensive overhauls of the entire game. Some mods may reveal hidden user interfaces that unlock additional features or enable players to acquire game objects or value without making in-app purchases, effectively bypassing the intended monetization mechanisms.

- **Personally Identifiable Information (PII) leaks**
  PII leaks refer to cyber attacks that involve the unauthorized collection and potential exploitation or sale of valuable personal information. These attacks can occur through various means, such as manipulating game forms to gather personal data, targeting data repositories that store information of game users, or capitalizing on developer mistakes that expose sensitive data. The collected information can encompass a range of data, including

email addresses, passwords, credit card details, device information, and other personally identifiable and sensitive data. PII leaks pose a significant threat to individuals' privacy and security, requiring robust measures to safeguard personal information and mitigate the risks associated with such attacks.

- **Phishing attacks**. Phishing attacks aim to deceive individuals and obtain their personal information or payments. Attackers impersonate trusted individuals or services through messages, requesting personal data. Once acquired, the collected information can be sold or used for extortion purposes. Gamers are at particular risk of phishing attacks. Within a year, a single security solution [29] identified over 3.1 million instances of phishing in online games. These attacks primarily focus on acquiring user credentials to gain control of gaming accounts. Even popular titles like Grand Theft Auto have been targeted, where attackers set up websites offering in-game rewards to lure players into providing their credentials.

- **Distributed Denial of Service attacks**. A Distributed Denial-of-Service (DDoS) cyberattack is designed to disrupt regular server traffic by flooding it with excessive requests, resulting in a slowdown or complete blockage of legitimate connections. These attacks can be directed at game servers, impacting multiple users by blocking their connections, or targeted at individual devices to disrupt the online gaming experience of a single user [30]. The motivations behind these attacks may vary, and the data required differs accordingly. In the case of DDoS attacks on individuals, the objective is to deliberately render the victim's online gaming system slow and unplayable. This is typically done with the intention of gaining a competitive advantage over the targeted user. To execute such attacks, the attacker needs to obtain the IP address of the individual, which can be obtained through various means, including the use of malware.

- **Malicious payloads and malware**. Certain PC and mobile games present a heightened risk to users' online and personal security due to the presence of hackers or inadequate security measures by developers. These games can lead to device infections with malware, which aims to steal sensitive data. Users may unwittingly download the wrong file or an infected program, making their devices vulnerable to such attacks. In some cases, downloaded games can be compromised with malware when hackers inject malicious code into an otherwise legitimate game. Additionally, malicious actors may create counterfeit applications that masquerade as legitimate games but are, in reality, concealed viruses. This type of threat is particularly prevalent when downloading games from torrent sites, although it is also possible with mobile games. It is essential for users to exercise caution and employ robust security measures to mitigate the risks associated with these compromised games.

3.3 Player profiling

Game Analytics is an interdisciplinary field that leverages statistics, data mining, machine learning, and data visualization to derive valuable insights about player behaviors and preferences. These insights are used to guide game designers and developers in making informed decisions. By analyzing telemetry data from gamified systems, valuable information can be extracted, including players' preferences, strategies, behavioral patterns over time, and even predictions about player rate. The acquired information plays a vital role in shaping the customization and adaptation of the user experience, ensuring that games are tailored to meet the needs and preferences of the players. Game Analytics serves as a powerful tool in understanding player engagement, enhancing game design, and ultimately optimizing the overall gaming experience.

By closely monitoring the interactions between users and a gamified system, it becomes possible to identify and rectify any design flaws within the game. This monitoring process involves assessing whether the intended gamification objectives are being pursued effectively. Additionally, it allows for the detection and moderation of abnormal behaviors to ensure fairness among players, such as identifying and blocking cheaters or individuals engaging in unfair practices. Furthermore, monitoring user interactions enables the profiling of players, which in turn facilitates the creation of personalized experiences. This personalized approach takes into account individual preferences, behaviors, and engagement patterns to tailor the game experience to each player. By leveraging monitoring mechanisms, game developers can address design issues, maintain fairness, and deliver personalized experiences that enhance overall user satisfaction and engagement [31].

3.3.1 Definition

Player profiling is a branch of player modeling that concentrates on gathering information that is not specific to a particular game and remains relatively stable over extended periods. This includes data such as personality traits, cultural background, gender, and age, which form the basis of a player's profile [32]. Unlike game-specific player models, player profiles can be applied across different games. Models created for one game can be utilized in another, and player data from various games or sources outside of games can be combined to develop comprehensive profiles. Such portability of player profiles makes them more suitable for predictive purposes rather than adaptive use, unlike player models that are tailored to a specific game and focus on immediate gameplay adjustments. Player profiling allows for insights and predictions that can be beneficial in game design, personalization, and targeted player experiences.

3.3.2 Uses of player profiling

Player profiles offer several similar applications as player models, along with some distinct advantages and disadvantages. They can be utilized in the following ways:

- **Dynamic difficulty adjustment**. Player profiles have the capability to fine-tune the game difficulty according to the specific abilities of each player. This is achieved by making adjustments to various aspects, such as numerical values and the placement or removal of obstacles. The goal is to provide a level of challenge that keeps players engaged and interested without causing excessive frustration. It is worth noting that player profiles can be imported from other games, allowing developers to incorporate the preferred difficulty settings or gameplay preferences of players right from their initial playthrough. This ensures that players have the most enjoyable and personalized difficulty level tailored to their individual skills and preferences.

- **Content generation**. Whether through procedural generation or manual design, the use of player profiles enables game developers to create content that resonates with players on a more personal level. By aligning the challenges and gameplay mechanics with the player's profile, developers can provide a more enjoyable and fulfilling gaming experience.

- **Monetisation**. With player profiles, developers can analyze player preferences, purchasing history, and behavior patterns to identify the types of content that are most appealing to individual players. This opens up opportunities for personalized offers, targeted advertising, and optimizing the presentation of content in the in-game marketplace.

3.4 Machine Learning models

Based on the specific domain at hand, there are different models to choose from, each with its own strengths and weaknesses. In this context, we describe three of the models used: Decision Tree, Random Forest and Logistic Regression. Then we define the loss functions and the metrics used to score the models' performance.

3.4.1 Supervised and Unsupervised Learning

Before diving into the models' description, we need to define the difference between learning techniques. Supervised and unsupervised learning are two distinct approaches in machine learning. They diverge in their training methods and data requirements. Each approach possesses its unique advantages, making them suitable for varying types of tasks or problems.

Supervised machine learning relies on having both input and output data labeled during the model's training phase. Typically, data scientists annotate or mark the data during the preparation phase, and this labeled information is then used to train and validate the model. Once the model comprehends the connections between input and output data, it becomes capable of categorizing fresh, unobserved datasets and making predictions. The term "supervised" is applied because human involvement is essential in this process. A significant portion of available data exists in its raw, unlabeled form. To prepare it for supervised learning, human effort is needed to meticulously label the data, which can be a resource-intensive undertaking due to the requirement for a substantial amount of accurately annotated training data. Supervised machine learning is valuable for classifying unobserved data into well-defined categories and making predictions about future trends and changes using a predictive model. Models developed through supervised machine learning can understand and distinguish objects along with their defining characteristics. These predictive models are frequently employed to forecast various outcomes, such as predicting shifts in property prices or customer purchasing behavior.

Unsupervised machine learning involves training models using raw, unlabeled training data. Its primary purpose is to uncover inherent patterns and trends within unprocessed datasets or to organize similar data into distinct groups. This approach is frequently employed during the initial exploration phase to gain a deeper understanding of the datasets. In accordance with its name, unsupervised machine learning is a more autonomous approach when compared to supervised machine learning. While a human may configure model hyperparameters, such as the number of cluster points, the model can efficiently process vast volumes of data without continuous human supervision. Consequently, unsupervised machine learning is well-suited for addressing questions related to hidden patterns and relationships within the data itself. However, due to the reduced human intervention, it's important to carefully consider the interpretability and explainability of unsupervised machine learning results. Since the majority of available data is in its raw, unlabeled form, unsupervised learning serves as a valuable tool for extracting insights from such datasets by either grouping data based on shared characteristics or unveiling underlying patterns. In contrast, supervised machine learning can be resource-intensive because of its reliance on labeled data.

3.4.2 Models

Decision Tree A decision tree is a supervised machine learning technique employed for categorization and prediction, relying on the sequence of questions and answers from previous data. This method falls under supervised learning, in which the model learns and is evaluated using a dataset containing the desired categorizations. Here is the terminology for decision trees:

- **Root node** - the starting point or foundation of the decision tree;
- **Decision node** - occurs when a sub-node is further divided into additional sub-nodes due to a decision;
- **Leaf node** - a sub-node that does not split further, indicating potential outcomes;
- **Branch** - a segment of the decision tree including multiple nodes;
- **Splitting** - the action of dividing a node into multiple smaller sub-nodes;
- **Pruning** - the process of eliminating sub-nodes within a decision tree.

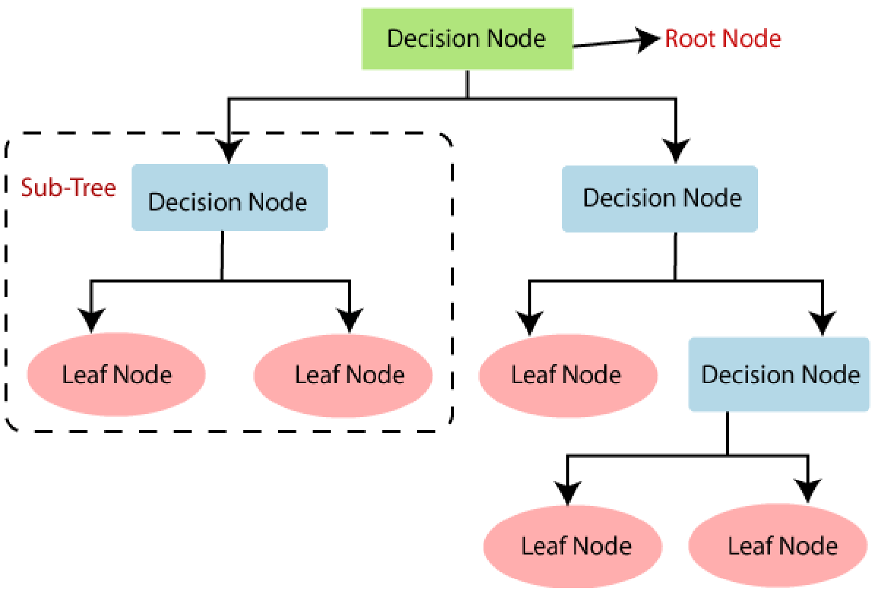

Figure 3.1: Stylized example of a decision tree

A decision tree (fig. 3.1) is similar to a tree in nature. At its core lies the root node, serving as the starting point. As you progress from the root, you encounter decision nodes, analogous to forks in the tree, where choices are made. These decision nodes lead to leaf nodes, much like the leaves on the branches of a tree, signifying the potential outcomes of those choices. Each decision node represents a query or a point of division, and the resulting leaf nodes symbolize the possible answers or consequences. In essence, leaf nodes sprout from decision nodes, similar to how leaves grow from branches, thus giving rise to the term “branch” for each section of a decision tree.

Creating a decision tree entails a two-step process. First, there's the construction phase, during which you choose the attributes and criteria that will shape the tree. Following this, there is the pruning phase, which involves eliminating unnecessary branches that might hinder accuracy. Pruning includes identifying outliers, which are data points significantly different from the majority, and could disrupt the calculations by giving undue importance to rare occurrences in the data. Depending on the type of outcome variables that are used, decision trees can either be continuous or categorical. In our case, we employed categorical decision trees, as we are dealing with that kind of data for our classes.

Random Forest A random forest is a supervised machine learning algorithm derived from decision trees. It is a technique used for solving regression and classification problems, employing ensemble learning, which combines multiple classifiers to address complex issues. A random forest comprises numerous decision trees. This "forest" is trained through *bagging* (bootstrap aggregating), an *ensemble* learning technique that enhances the accuracy of machine learning models. In the *bagging* method we select a random subset of data from the training set with a technique known as sampling with replacements (i.e. it is possible to select the same data points more than once). Once we have generated several of these data subsets, we proceed to train individual models independently. Depending on the nature of the task, whether it is regression or classification, we calculate the average (for regression) or majority (for classification) of the predictions from these models to determine the final output of the *ensemble*. This process leads to a more precise estimate. A key difference between random forests and decision trees is *feature bagging*. It involves creating decision trees with a degree of unpredictability in feature selection. In contrast to traditional decision trees that examine all possible features for splitting data, feature randomness selects only a random subset of features to make decisions. The random forest algorithm determines outcomes by aggregating predictions from these decision trees, typically using an average or mean approach. Increasing the number of trees enhances prediction accuracy. In essence, a random forest overcomes decision tree limitations, reducing overfitting and improving precision. It provides predictions without requiring extensive parameter tuning, but with the downside of a generally longer training time.

Logistic Regression Logistic regression is a type of regression analysis. It is a statistical technique used for examining a dataset containing one or more independent variables influencing an outcome. While typically employed to forecast binary target variables, logistic regression can be adapted for multinomial problems, like in our case study. The primary objective of employing logistic regression is to identify the most suitable model for characterizing the connection between the dependent and independent variables, and it works by estimating probabilities of belonging to one of the target classes. When employing logistic regression for binary classification tasks, the activation function of choice is usually the sigmoid (Equation 3.1), which has its signature S-shape curve that maps inputs into values between the range of 0 and 1.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{3.1}$$

In the case of multiclass classification, the sigmoid is superseded by the softmax (Equation 3.2) activation function, which converts the initial outputs from the model into a probability vector, essentially creating a distribution of probabilities across the input classes. In the softmax's formula, z_i is the i-th element of the vector $\vec{z}$, which is the probability vector, and “K” refers to the number of input classes.

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad for\ i = 1, 2, \ldots, K \tag{3.2}$$

3.4.3 Loss Function

In supervised learning (which is our case), the loss function serves as the mechanism for calculating the disparity between the algorithm's current output and the expected output. It is a means of assessing how well the algorithm represents the data. These functions can be divided into two main categories: one for classification tasks, dealing with discrete values, and the other for regression tasks, which involve continuous values. In many machine learning problems, the objective is to reduce the loss function to its minimum value, signifying that the model's predictions align as closely as achievable with the actual data points. The selection of a particular loss function depends on the problem's characteristics, like whether it involves regression or classification, and the intended performance of the model throughout the training process. In our study, we deal with a classification task, and therefore, we have to refer to the available loss functions for the selected models. For the decision tree and the random forest models, we tested the Entropy (or Log-Loss) and Gini losses.

The Entropy loss function serves as a metric for assessing the effectiveness of a classification model that produces probability values ranging from 0 to 1. The log-loss quantifies how much the predicted probability deviates from the true label. When there is a significant disparity between the predicted probability and the actual label, the log-loss increases. For instance, in binary classification tasks there are two classes denoted 0 and 1. The model generates a probability indicating the likelihood of a data point belonging to class 1. If such probability exceeds 0.5, the data point is classified as belonging to class 1; otherwise, it is classified as belonging to class 0. In the case of a log-loss function for binary classification, if the model predicts a probability of 0.01, whereas the real label is 1, this would yield a high loss value, indicating poor performance. In an ideal scenario, a perfect model would achieve a log-loss of 0. In equation 3.3 we consider log as the natural logarithm, k indicates the class of which we are computing the loss, m is the sample we are currently considering, y_{mk} is the binary indicator (0 or 1) that m is in class k, and p_{mk} is the predicted probability that m is in class k. If we were to compute the loss on a binary classification problem, we would remove the sum, and calculate the loss on the only two classes present.

$$LogLoss = -\sum_{k} y_{mk} \log(p_{mk}) \tag{3.3}$$

The Gini impurity serves as a metric for assessing how often an element within a dataset would be incorrectly classified if it were randomly assigned a class. When the Gini Index reaches its minimum value of 0, it signifies that a node is pure. This purity indicates that all elements within the node belong to a single, unique class. Consequently, such a node will not undergo further splitting. Therefore, the selection of the optimal split is determined by features with lower Gini Index values. Conversely, the Gini Index reaches its maximum value when the probabilities of the two classes are equal. The equation 3.4 has, again, k as the class it is computing the loss for, and p_{mk} is the probability of m being classified as part of k

$$GiniLoss = 1 - \sum_{k} (p_{mk})^2 \tag{3.4}$$

Differently from Random Forests and Decision Trees which employ the use of pruning and tree depth limitations, the Logistic Regression, in the optimization problem, uses the *Regularized Loss Minimization.* It is a learning paradigm that tries to reduce both the *Empirical Risk* and the *Regularization Function.* The *Empirical Risk* is the average loss derived over the training data and the *Regularization Function* is a component employed to fine-tune machine learning models to minimize the adjusted loss function, thereby mitigating overfitting or underfitting. The optimization problem of logistic regression for classification tasks can be modeled as shown in equation 3.5:

$$min_W \sum_{i=1}^{n} \sum_{k=0}^{K-1} [y_i = k] \log(\hat{p}_k(X_i)) + \lambda \cdot r(W) \tag{3.5}$$

where $[y_i = k]$ represents the Iverson bracket which evaluates to 0 if is false, otherwise it evaluates to 1, and $r(W)$ indicates the regularization function[1]. The objective function used to train a logistic regression has two components, the

[1]`https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression`

loss function and the regularization parameter. The tradeoff between them is regulated via the parameter λ which is the regularization strength (the higher the value, the higher the regularization power).

We chose to evaluate two of the possible regularization functions: L1 and L2. L1 or Lasso regression (Equation 3.6), short for *Least Absolute Shrinkage and Selection Operator*, introduces a penalty term into the cost function. This penalty term (λ) is the sum of the absolute values of the coefficients. As the coefficients increase from zero, this term penalizes them, leading the model to decrease their values to minimize loss.

$$Loss = Error(Y - \widehat{Y}) + \lambda \sum_{1}^{n} |w_i| \tag{3.6}$$

L2 or Ridge regression (Equation 3.7) introduces a penalty term in the form of the square of the coefficients, represented as the L2 term. This term accounts for the squared magnitude of the coefficients. Additionally, a parameter, denoted as λ(lambda), is included to control the strength of this penalty. When λ is set to zero, the equation simplifies to ordinary least squares (OLS). However, for $\lambda > 0$, a constraint is imposed on the coefficients. As λ increases, this constraint drives the coefficients towards zero. This results in a tradeoff, where higher values of λ lead to higher bias (some coefficients tend to become exactly zero while others become very large), but lower variance, making the model less flexible.

$$Loss = Error(Y - \widehat{Y}) + \lambda \sum_{1}^{n} w_i^2 \tag{3.7}$$

One key distinction between ridge and lasso regression is that lasso tends to drive some coefficients to exactly zero, while ridge never forces coefficients to reach absolute zero.

3.4.4 Metrics

To measure the efficacy of our models we considered the most common performance metrics, which are *precision*, *recall*, *accuracy*, and *F1-score*. The base notions needed to introduce such metrics are reported in Table 3.1 as a confusion matrix for a binary classification problem. The two target classes are indicated as “positive” and “negative” respectively.

Table 3.1: Confusion matrix of a binary classification problem

	Actual Positive	**Actual Negative**
Predicted Positive	True Positive	False Positive
Predicted Negative	False Negative	True Negative

Precision Quantifies the model's capability to accurately pinpoint and classify positive data points. To put it in mathematical terms, precision is calculated as the number of true positives (TP) divided by the total number of positive samples, i.e., by the sum of true positives and false positives (FP):

$$Precision = \frac{\text{TP}}{\text{TP+FP}}$$

Recall This is a metric that measures the model's capacity to correctly recognise all the positive instances of the dataset. Mathematically, it is defined as the number of true positives divided by the sum of true positives and false negatives (FN):

$$Recall = \frac{\text{TP}}{\text{TP+FN}}$$

Accuracy Provides a broad overview of a model's performance across all the predictive outcomes and is particularly suited when samples are equally distributed among classes. Besides that, it is the only considered metric that takes into account the quantity of True Negatives (TN). It is defined as the ratio between the number of accurate predictions over the total number of predictions:

$$Accuracy = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}}$$

F1-score Like *accuracy*, supplies a comprehensive evaluation of the model's behavior but, in addition, is also reliable when classes are unbalanced. In other words, F1-Score also accounts for data distribution. It is computed as the harmonic mean of precision and recall:

$$F1\text{-}Score = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * \text{TP}}{2 * \text{TP+FP+FN}}$$

These metrics can also be adapted to the multiclass case, merely by alternately accounting for each class as positive and evaluating all the other classes as negative. Then, it is enough to average the various scores, to obtain the final measure provided by the chosen metric. It is important to mention that there are different possibilities to compute the mean, especially when it comes to the *F1-Score*. In particular, we chose to utilize the "macro" option provided by the *metrics* module of *scikit-learn*[2], which returns the unweighted mean of the metric computed for each label.

[2]`https://scikit-learn.org/stable/modules/model_evaluation.html`

Chapter 4

Data Collection

In this chapter, we describe the data collection process. First, we explain the setup of the experiment and the choices in games and equipment, then the actual data collection method.

4.1 Video Games

Since there is an enormous amount of possible video games in existence, we had to choose between a vast variety of titles in order to have a list that comprehended diverse genres and games. Moreover, we had to select games known enough to have a high probability of finding participants with some degree of expertise in playing them. Another requirement was to have different genres that were similar to one another to a certain extent. For instance, if we happened to choose a racing simulator as the first genre, we would then select other genres closely related to driving, such as space exploration or arcade driving games. This decision was made with the testing phase in mind, where we needed to train the model on one genre and then test it on another. If the selected genres were too dissimilar, the model would have been disadvantaged from the beginning. One more crucial criterion for selecting the games was the presence of a repeatable tutorial phase and/or the option to play against computer-controlled bots rather than human opponents. This decision was made to ensure the testing process's consistency and allow participants to engage with the same challenges in a controlled environment. Many of these games offer a multiplayer mode that relies on matchmaking algorithms to pair players for online matches. However, this can lead to discrepancies in the gameplay experience. For example, if an experienced player is matched with newcomers due to factors like a fresh account or the prior participant's inexperience, the gameplay becomes too easy and does not reflect their true skill level. Conversely, if a participant with little to no gaming experience is placed in a lobby with highly skilled players, they cannot showcase their abilities because the game ends quickly, or they must play cautiously due to the challenging environment. An additional issue with online play is the occasional loss of connectivity, which can disrupt the testing session and create problems. Therefore, by choosing games with a single-player mode against bots, we could ensure a controlled and consistent testing environment for all participants. By keeping all of these considerations in mind, we had to choose two genres among the list of possible ones:

- Fighting Games;
- Roguelike/Roguelite;
- First Person Shooter;
- Role Playing Games;
- Hack'n'Slash.

The common point of all of these genres is that they all involve the player controlling a character's movements and actions, often featuring combat stages. However, we had to exclude some genres from our selection due to compatibility issues between certain games and the selected controllers. To keep the testing phase manageable in terms of duration, we ultimately decided to pick two games from both First Person Shooters (FPS) and Role Playing Games (RPG).

4.1.1 FPS

Battlefield V

Battlefield V (Figure 4.1), also referred to as Battlefield 5, BFV, or BF5 (with the development code-name 'Casablanca'), is the sixteenth installment in the Battlefield Series. This game is developed by DICE and published by EA. The open beta for the game started on September 4th, 2018, exclusively for EA Access and Origin Access members. It was later made available worldwide from September 6th, 2018, until September 11th, 2018. The official release dates varied, with an early release for EA Access and Origin Access members on November 9th, 2018, and the Early Enlister Deluxe Edition owners on November 15th, 2018. The

Figure 4.1: Cover art of Battlefield V

worldwide release date for all players was November 20th, 2018. Battlefield V revisits the significant backdrop of World War II, emphasizing the often-overlooked battles of this global conflict. The game's primary objective is to provide players with an exceptionally immersive experience within the series. This immersion is achieved through engaging story-lines that traverse the major battles of the war. In terms of gameplay, Battlefield V introduces comprehensive redesigns of core mechanics, including how players move and how health regeneration functions, among others. Players are granted a more interactive role in the game world, requiring them to physically perform actions like opening doors, entering vehicles, and picking up health or ammunition items. This heightened level of realism and engagement enhances the overall gaming experience. The primary campaign of Battlefield V comprises a series of distinct narratives known as War Stories. These narratives chronicle the lesser-known wartime experiences of numerous soldiers in different World War II settings. As seen in previous game iterations, players can seek collectibles within each War Story, including Letters and War Story Challenges[1].

PlayerUnknown's Battlegrounds

PlayerUnknown's Battlegrounds (Figure 4.2), also known as PUBG: Battlegrounds, is a battle royale game that was developed by PUBG Studios and published by Krafton. This game draws its inspiration from the Japanese film "Battle Royale"

[1]`https://battlefield.fandom.com/wiki/Battlefield_V`

Figure 4.2: Cover art of PUBG

(2000) and is rooted in the earlier mods created by Brendan "PlayerUnknown" Greene for various games. Under Greene's creative leadership, these concepts were expanded into a standalone game, marking the inception of the PUBG Universe series. Players engage with the game from either a third-person or first-person perspective. In PUBG, up to one hundred players are airdropped onto an island where they must scavenge for weapons and equipment, all while aiming to eliminate other players and avoid being eliminated themselves. As time progresses, the safe area on the game map gradually shrinks, compelling remaining players to confront one another in a steadily diminishing play area. Victory in each round is claimed by the last player or team standing[2].

4.1.2 RPG

Dark Souls 3

Dark Souls III (Figure 4.3) is an action RPG developed by From Software. Its official announcement was made on June 15, 2015, during E3. The game was initially launched in Japan on March 24, 2016, for Microsoft Windows, PlayStation 4, and Xbox One, followed by a global release on April 12 of the same year. As a bonus, users who pre-ordered the game received a code allowing them to play Dark Souls on Xbox One. The game unfolds in the Kingdom of Lothric, where players are faced with the daunting task of surviving an impending apocalypse triggered by the ongoing conflict between the Age of Fire and individuals marked with the Darksign, a theme consistent with previous installments in the series. To endure this catastrophic event, the player's character must confront the Lords of Cinder, former heroes who previously linked the fire. This battle is situated

[2]`https://pubg.fandom.com/wiki/PLAYERUNKNOWNS_BATTLEGROUNDS_Wiki`

Figure 4.3: Cover art of Dark Souls 3

Figure 4.4: Cover art of Elden Ring

within the recurring cycle of Light and Dark, emphasizing that, regardless of the player's actions, darkness will inevitably return, perpetuating an eternal cycle[3].

Elden Ring

Elden Ring (Figure 4.4), released in 2022, is an action RPG developed by FromSoftware. It was a collaborative effort, with direction from Hidetaka Miyazaki and worldbuilding contributions by renowned fantasy writer George R. R. Martin. The game saw a release on multiple platforms, including PlayStation 4, PlayStation 5, Windows, Xbox One, and Xbox Series X/S. Its Japanese release date was February 25, and it was distributed internationally by Bandai Namco Entertainment. In Elden Ring, players assume the role of a customizable character embarking on a quest to restore the Elden Ring and claim the title of Elden Lord. The game is presented from a third-person perspective, granting players

[3]`https://darksouls.fandom.com/wiki/Dark_Souls_III`

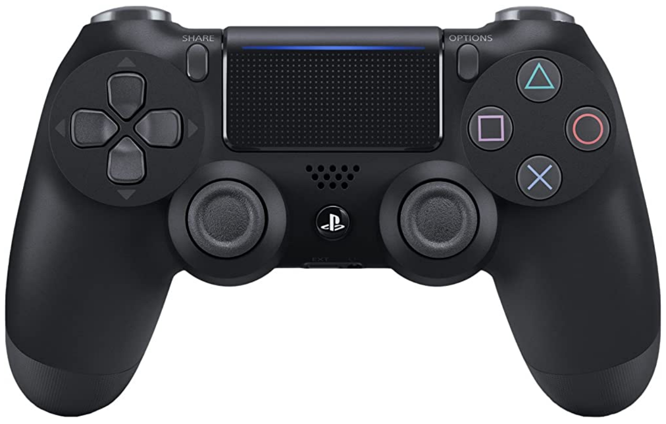

Figure 4.5: Dualshock 4

the freedom to explore a rich and interactive open world. Travel is facilitated by the player character's steed, Torrent, which serves as the primary means of transportation. Hidden dungeons are scattered throughout the world, offering rewards to those who venture within. Players have access to various weapons, magic spells, and even stealth mechanics for non-direct engagement with adversaries. Fast travel checkpoints are strategically placed across the game's expansive world, allowing players to enhance their attributes using an in-game currency known as Runes. Elden Ring also includes an online multiplayer mode, enabling cooperative gameplay and player-versus-player combat experiences[4].

4.2 Equipment

We faced the task of selecting from a plethora of gaming devices, each associated with different consoles. Among the viable options were controllers such as the PS3's Dualshock 3, PS4's Dualshock 4, PS5's Dualsense 5, Nintendo Switch's Pro Controller, Nintendo Switch's Joycons, Xbox 360's Controller, Xbox One's Controller, and Xbox Series X's Controller. In the process of elimination, we ruled out controllers from the Xbox series due to their lack of gyroscopic and accelerometric sensors. Ultimately, our choice centered on utilizing the Dualshock 4 and the Nintendo Switch's Pro Controller.

4.2.1 Dualshock 4

The DualShock 4 (Figure 4.5) serves as the controller for the PlayStation 4 (PS4). It introduces several enhancements over the DualShock 3. Notably, it features

[4]`https://eldenring.wiki.fextralife.com/Elden+Ring+Wiki`

a built-in two-point capacitive touch pad on the controller's front, which can be clicked. The controller supports motion detection through a three-axis gyroscope and three-axis accelerometer, along with vibration feedback. It is equipped with a non-removable, rechargeable 3.7 V, 1000 mAh lithium-ion battery that can be charged while the system is in rest mode. The controller weighs 210 g (7.4 oz) and measures 162 mm × 52 mm × 98 mm (6.4 in × 2.0 in × 3.9 in).

The front of the DualShock 4 includes a light bar with three LEDs that can illuminate in various colors. Originally developed for PlayStation VR, it serves multiple purposes, such as identifying players by matching the colors of their in-game characters or providing feedback by changing colors or patterns in response to gameplay. Additionally, the light bar works alongside the PlayStation Camera to track the positions and movements of multiple players. The controller features several input and output ports, including a stereo 3.5mm headset jack, a micro-USB port, and an extension port. Charging can be done via the console, a dedicated charging station, or a standalone charger using microUSB. The DualShock 4 also includes a mono speaker. The DualShock 4 introduces various buttons: PS button, SHARE button, OPTIONS button, directional buttons, action buttons (triangle, circle, cross, square), shoulder buttons (R1/L1), triggers (R2/L2), analog stick click buttons (L3/R3), and a touch pad click button. Some notable changes from the DualShock 3 include merging the START and SELECT buttons into a single OPTIONS button and introducing a dedicated SHARE button for easy sharing of screenshots and videos from gameplay. Only the L2 and R2 triggers are pressure-sensitive, which differs from the DualShock 2 and 3. This change can be attributed to the limited utilization of these buttons in most games and to maintain parity with competitor controllers.

4.2.2 Nintendo Switch: Pro Controller

The Nintendo Switch Pro Controller (Figure 4.6), designed by Nintendo for the Nintendo Switch console, offers an alternative to the Joy-Con controllers. Its button layout is reminiscent of the Wii Classic Controller Pro, with a staggered analog stick arrangement related to the GameCube controller and Microsoft's Xbox controllers. In terms of design, it has similarities to the Xbox Controller. The Nintendo Switch supports the use of up to eight Pro Controllers simultaneously. This Pro Controller has a range of features, including near-field communication for compatibility with Nintendo's Amiibo toys, HD Rumble, and motion controls. Its rechargeable battery is the same as that found in the 3DS/2DS handheld game

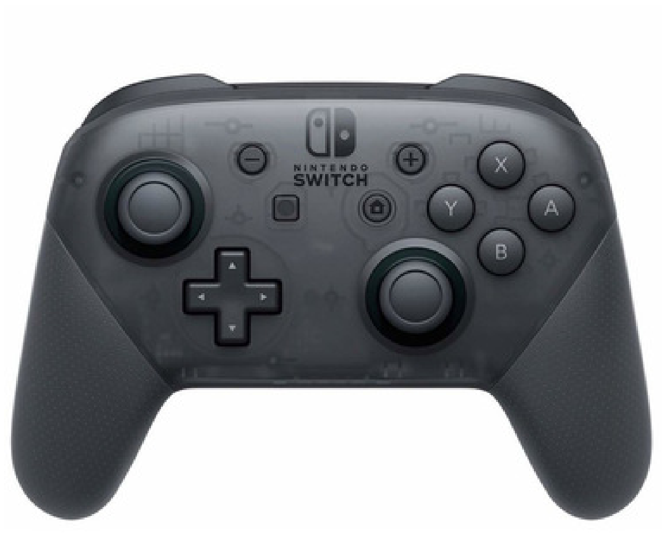

Figure 4.6: Nintendo Switch: Pro Controller

consoles and Wii U Pro Controllers. To recharge the controller, a USB-C connector is used, and it comes with a USB-C to USB Type-A charging cable. This cable can be connected to one of the USB-A 2.0 ports on the Nintendo Switch dock for convenient charging. The controller supports motion detection through a three-axis gyroscope and three-axis accelerometer, along with vibration feedback. The Pro Controller introduces various buttons: Home button, Screenshot button, Plus button, Minus button, directional buttons, action buttons (A, B, X, Y), shoulder buttons (R/L), triggers (RZ, LZ), analog click buttons (L3/R3). The LZ and RZ triggers are not pressure sensitive, in fact, they are buttons in the same way as R and L.

4.2.3 Computer

Because there is virtually no distinction between a gaming console and a computer in terms of the gaming experience, we opted to utilize a computer from the Department of Mathematics, which was at our disposal, to execute the games and operate the Python script responsible for capturing the inputs from the controllers. The specific computer we used is equipped with an AMD Ryzen 5 3600X processor, an NVIDIA RTX 3090 12GB graphics card, 32GB of DDR4 RAM, a 1TB HDD, and runs on Windows 10 Pro.

Figure 4.7: Steam Logo

4.3 Experiment Setup

First, we had to prepare the computer to be able to run the games and record the activity of the participants. This involved installing the games through Launchers, installing Anaconda to run it on the Spyder IDE, and coding the Python Script to record the inputs of the DualShock 4 and Switch Pro Controller.

4.3.1 Steam Launcher

Steam[5] (Figure 4.7), made by Valve Corporation, is one of the most popular digital platforms designed for the distribution of PC games. This platform lets users have the ability to acquire PC games online and install them directly to their personal computers upon purchase. In addition to buying games, Steam offers a range of features such as user reviews, the uploading of user-generated content, the purchase of DLCs (Downloadable Content), and more. Steam's client offers various functionalities, like game updates, a friends list, in-game voice chat, and the ability to share games among friends. Initially launched in 2003, Steam introduced the Steam Community four years later, during its beta testing phase. It features a user-friendly interface that enables users to discover a wide selection of games, from Action and Adventure to Indie and 3rd-person shooters. In addition to purchasing games, users can also trade or gift collectible items within the Steam community. Steam's community section is a forum where gamers can write reviews, discuss news about new games, share memes, offer tips and tricks, uncover Easter eggs, and engage in various discussions. One of Steam's strengths is its comprehensive game information. Users can access game trailers, preview screenshots, and see user tags that indicate whether a game offers single-player, co-op, or multiplayer modes. Steam also provides information on

[5]`https://store.steampowered.com/`

Origin™
Powered by EA

Figure 4.8: Origin Logo

system requirements, user reviews, recent news, and similar titles. Users can add games to their wish lists or shopping carts for easy checkout. Once users have entered their personal information, buying subsequent games becomes a seamless process, with no need to re-enter credit card details. Steam is compatible with Windows, macOS, Linux, TV, and mobile devices. A reliable internet connection and a modern laptop are all that's required to enjoy bandwidth-intensive games.

4.3.2 Origin Launcher

Origin[6] (Figure 4.8) is a digital platform created by Electronic Arts, initially for Windows and later for macOS, designed for purchasing and playing video games. It offers various social features such as profile management, connecting with friends for chat and joining games directly. Additionally, it includes an in-game overlay, supports streaming via Twitch, and facilitates sharing game libraries and community engagement through platforms like Facebook, Xbox Live, PlayStation Network, Nintendo Network, and Nintendo Account. In 2011, Electronic Arts expressed its ambition for Origin to rival Valve's Steam service, its primary competitor. This involved incorporating features like cloud game saves, automated game updates, achievement systems, and the release of games across multiple platforms. By 2013, Origin had amassed a user base exceeding 50 million registered users. The Origin store offers a platform for users to explore and buy games featured in Electronic Arts' collection. Rather than receiving a physical box, disc, or CD key, the software purchased is instantly linked to the user's Origin account, and it can be downloaded using the corresponding Origin client.

4.3.3 Python script for controllers

To capture buttons and sensor data from the controllers, we connected the controllers and the computer via their proprietary USB cables. To interpret and

[6]`https://www.ea.com/en-gb`

gather these input data, we created a Python script that utilized the libusb[7] library. Such library is C-based and designed to offer a universal way to interact with USB devices, making it easier for developers to create applications that communicate with USB hardware. Its key features include:

1. **Portability**, this library utilizes a single cross-platform API, allowing access to USB devices on various operating systems like Linux, macOS, and Windows.

2. **User-mode**, applications can communicate with devices without needing special privileges or elevated permissions.

3. **Version Compatibility**, libusb supports all versions of the USB protocol, from 1.0 to the latest 3.1, ensuring compatibility with a wide range of USB devices.

We mapped all controller inputs manually, which allowed us to read and record them effectively. Both controllers transmit data in the form of 64-element arrays, with each element representing a specific input. To correctly identify the controllers and discern the associated input IDs, we relied on the hidapi[8] library, which provided us with the necessary Vendor ID and Product ID for identification. Each element in these arrays was a byte of data and corresponded to either a set of buttons or sensor readings. For instance, the action buttons (such as A, B, X, and Y on the Pro Controller) were all encapsulated within a single byte. By employing a straightforward bitmasking technique, we were able to determine which button was currently being pressed. In contrast, the directional buttons were represented by two axes. To handle the gyroscope and accelerometer data, we needed to transform it into a range that fell within 2^{16}.

4.4 Data Gathering

The data collection phase of the project can be divided into two distinct parts: location and recording. We recruited participants through various channels, including our circle of friends, social media group chats, class announcements, and word-of-mouth referrals. The data recording phase of the project commenced in early June 2023 and spanned approximately 90 days. Participants who agreed to

[7]`https://pypi.org/project/libusb/`
[8]`https://pypi.org/project/hidapi/`

assist us with the project were given invitations to a Spritz party as a token of appreciation once the recording phase concluded.

The location designated for our experiment setup was within the HIT Centre (Human Inspired Technology) at Via Luzzatti 4, Padova. Inside this facility, we were provided with a room where we set up our equipment. This arrangement included connecting the computer to a 55" Samsung television featuring 4K resolution and a curved screen. Participants were seated in a stationary chair positioned at a distance of 1.80 meters from the screen, and the audio was transmitted through the TV's built-in speakers.

Prior to commencing the gaming session, each participant was required to provide their consent by signing a disclaimer for the Ethical Committee, which is overseen by the HIT organization itself. By signing, the participants would give us the approval to use their gaming data for our research. Following this, participants were asked about their familiarity with the four games they were about to play, their experience using the controllers, and their general gaming background. Subsequently, we assigned them a randomly generated alphanumeric identifier, as stipulated in the disclaimer, to guarantee anonymity in the processing and potential publication of their data. Alongside this random ID, we recorded their gender and whether or not they possessed prior gaming expertise.

After completing the preliminary steps, we randomly allocated participants to one of the two controllers and selected a permutation of the four games from the available options to eliminate session bias. Each game had a duration of 15 consecutive minutes. Participants played all four games with one controller, and then repeated the process with the other controller, using a different order of the games. Just before starting each game, we initiated the Python script to record their actions, along with Windows' built-in screen recording feature. This allowed us to review and document every action they performed during the gameplay. The purpose was to label the data and test whether the model would exhibit an advantage when handling specific types of scenarios with respect to the unlabeled data. In the following, we report the list of elements and scenarios presented to the participants for each of the considered games:

- **Battlefield V** - In this scenario, the participant is instructed to engage in one of the single-player campaign missions titled "Under No Flag." In this mission, the player assumes the role of Billy Bridger, a reformed London-based criminal who is given a second chance and recruited into the Special Boat Service under the leadership of George Mason. Together with a team of similarly unconventional individuals, their mission is to sabotage Luftwaffe air bases deep within North Africa, a daunting 500 miles behind enemy lines. Players are presented with the choice of either executing precise surgical strikes on key targets and making a quick exit or opting for a more chaotic approach by pilfering enemy Stuka dive bombers and utilizing them to eliminate the designated objectives. The gaming session continues until the player reaches the first significant checkpoint in the mission, which typically requires around 15 minutes of gameplay when following the intended path designed by the developers.

- **Dark Souls 3** - In this game session, participants commence the main storyline from the very start. They enjoy a considerable degree of freedom in deciding their actions, except for the necessity of attempting to defeat the initial enemy boss (named "Index Gundyr") at least once. They are required to try and beat the enemy. If the player has not previously encountered the mini boss known as the "Ravenous Crystal Lizard" in the preceding area, they will be prompted to attempt to defeat it. However, this should only be done after successfully defeating the main boss. However, they are not obligated to do so and can opt to proceed with their exploration in the subsequent area.
- **Elden Ring** - Similar to the Dark Souls 3 scenario, participants commence the game's narrative and navigate through the tutorial segment, leading to an unavoidable encounter with the first boss known as the "Soldier of Godrick". Upon successfully overcoming this challenge, they proceed to the initial primary game area, where they are encouraged to confront with the first mini-boss they encounter, the so called "Erdtree Knight," to fill the entire 15-minute recording session with engaging gameplay.
- **PUBG: Battlegrounds** - During the initial ten matches on a new account, players find themselves in lobbies populated solely by AI-controlled bots, creating a unique experience where they engage in PUBG matches but within bot-filled servers. Unlike the other games where there is flexibility regarding the number of deaths, in this case, the primary objective is to avoid dying. Even a single death results in the termination of the game, prompting participants to initiate a new match to continue playing. The 15-minute gameplay duration is specific to active gameplay, and in the event of a player's demise, the timer halts, only resuming with the start of

the subsequent match. Typically, players begin by airdropping into a city, gathering equipment, and subsequently striving to survive for 15 minutes, as PUBG matches typically extend beyond the 20-minute mark.

Chapter 5

Player Profiling

This chapter provides an overview of how we generated the dataset and a summary of the models we developed for participant identification based on their controller and screen recordings.

5.1 Dataset Creation

To construct our dataset, we followed a step-by-step process. Initially, the collected data was organized into files based on participants, games, and controllers during the recording session. Subsequently, we proceeded through several phases: *sequencing*, *identifying and combining*, *feature extraction*, and *variance analysis*. Upon completing these stages, our dataset was prepared for use with the AI model. All of these stages are motivated and explained in the present section.

5.1.1 Starting Features

Here are the starting features collected from the Dualshock 4 and the Switch Pro Controller during the gaming sessions (Table 5.1):

5.1.2 Sequencing

The sampling frequency of the controller data varied depending on the game. For instance, Battlefield V had roughly 110 samples per second, whereas Elden Ring had 40 samples per second. Considering that this data is in the form of time series, we opted to generate seven datasets with different sequence lengths (also done in order to better standardize the sampling rate): 1-second, 3-second, 5-second, 10-second, 20-second, 30-second, and 60-second sequences. Collectively,

Table 5.1: List of features taken from the controllers. All the controls are mapped with respect to their equivalent of the other controller. The buttons without a match are reported beside a white cell.

DS4	**Switch**
Cross	B
Square	Y
Circle	A
Triangle	X
R1	R
R2	RZ
L1	L
L2	LZ
L3	L3
R3	R3
Home	Home
Share	Minus
Options	Plus
Hat	Hat
Gyroscope X-Axis	Gyroscope X-Axis
Gyroscope Y-Axis	Gyroscope Y-Axis
Gyroscope Z-Axis	Gyroscope Z-Axis
Accelerometer X-Axis	Accelerometer X-Axis
Accelerometer Y-Axis	Accelerometer Y-Axis
Accelerometer Z-Axis	Accelerometer Z-Axis
Stick L X-Axis	Stick L X-Axis
Stick L Y-Axis	Stick L Y-Axis
	Stick L Z-Axis
Stick R X-Axis	Stick R X-Axis
Stick R Y-Axis	Stick R Y-Axis
	Stick R Z-Axis
Time Instant	Time Instant

these sequenced datasets amount to a total stored size of 17 gigabytes.

5.1.3 Identifying and Combining

In this stage, our objective was to formulate the ultimate identification feature to include in the dataset. This identification number is constructed from multiple components, which include:

- 2 digits $\in [10, 32]$ to indicate the participant code;
- 1 digit $\in [1, 4]$ to indicate the game of reference. Where "1" indicates Battlefield V, "2" refers to Dark Souls 3, "3" stands for Elden Ring, and

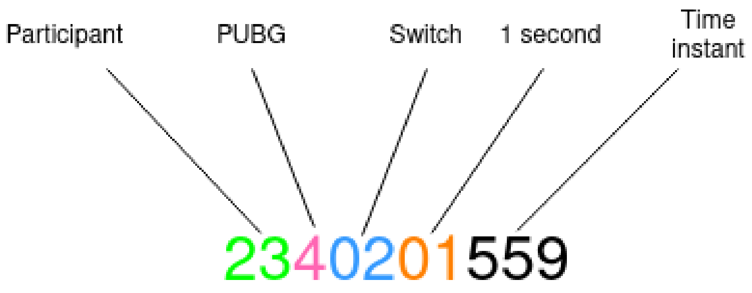

Figure 5.1: Identification Number Example

"4" is linked to PUBG;

- 2 digits $\in \{01, 02\}$ to indicate the controller used. "01" for the Dualshock 4 and "02" for the Nintendo Switch Pro Controller;
- 2 digits $\in \{01, 03, 05, 10, 20, 30, 60\}$ referring to the corresponding time interval;
- 1-4 digits to keep track of the starting instant of the sequence.

Where the order of the list is reflected in the order of the elements for the IDs (Figure 5.1). The "participant code" is an identification number that begins at "10" and increments automatically for each participant when constructing the dataset, in order to have always two digits and thus make parsing easier. For the sake of simplicity, we ordered alphabetically the participants from their alphanumeric codes. The last digits indicate the integer part of the "time" column where the interval would start. So, for example, the first element of every game recording has "0" as the "start of the sequence". Following dataset identification, we combined all the sequences, restoring the original format where each file denoted the participant, the game, and the controller.

5.1.4 Feature Extraction

For the feature extraction phase, we used the tsfresh[1] library. Tsfresh serves as a tool for structured feature engineering from sequential data, including time-series and other data types that share the characteristic of being organized based on an independent variable. This independent variable, as in our case, is time. However, it is important to note that other types of sequential data, such as reflectance and absorption spectra organized by wavelength, fall under this category. The features

[1]`https://tsfresh.readthedocs.io/en/latest/index.html`

extracted by tsfresh serve two primary purposes. First, they provide a descriptive representation of time series data, offering new perspectives and insights into the characteristics and dynamics of the data. Second, these features can be employed in various applications, including clustering time series data and training machine learning models for tasks such as classification or regression involving time series.

For our specific use case, we needed to keep each data file in the format we initially saved it, with individual files representing one participant playing one game with one controller. This was necessary to effectively utilize the *extract_features* function, provided by the library. We made this choice to avoid introducing any unwanted bias or contaminating the rest of the data while using the feature extraction function, and also maintained under control the RAM usage, which would otherwise fill up and interrupt the calculations. The *extract_features* function computed for every column of the file a long series of mathematical characteristics, spanning from the easiest mean and standard deviation to more complex features such as the Fourier coefficients of the one-dimensional discrete Fourier Transform for real input by fast Fourier transformation algorithm.

The whole process, for all the sequences of data, took at least three weeks of uninterrupted computation, divided on two separate computers, one with an Intel i5-10400F processor and 16GB of DDR4 RAM, and the other with a AMD Ryzen 5 3600X processor and 32GB of DDR4 RAM. It produced upwards of 20 thousand calculated features for every file, bringing the load of data from 17GB to 100GB.

5.1.5 Variance Analysis

Because the data obtained from feature extraction was extensive and contained many unusable features, we opted to filter it using sklearn's *VarianceThreshold* function. We set this filter to 0.10 (i.e., data with variance lower than 10%) to remove stagnant data. However, before applying the variance function, we eliminated every column that had NaN values, as these could interfere with the process. This endeavor took several days to finish but resulted in a much more manageable dataset, totaling 27GB. This dataset includes both labeled and unlabeled sets for each of the seven time interval sequences. We passed from the more than 20 thousand features to a few thousand.

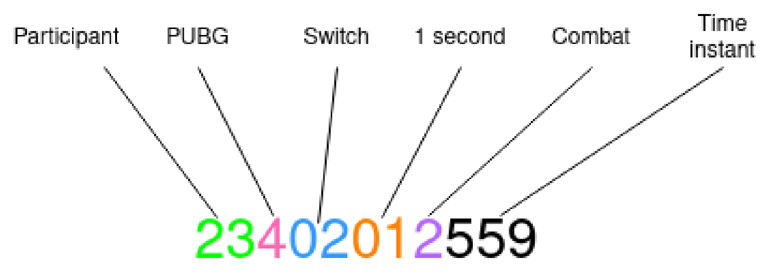

Figure 5.2: Identification Number with Labels

5.1.6 Labeling

We decided to create a version of our database adding labels because we wanted to explore if training and testing would gain an advantage if done on specific actions or scenarios. To construct a supplementary dataset, we relied on the screen recordings we had made. The data collected from 1080p/30fps videos occupied 252GB of space with almost 50 hours of gameplay. We reviewed and scrutinized each video depicting the participants' gameplay sessions. During this process, we meticulously documented the timestamps corresponding to specific actions or events. The labels we employed are:

- Exploration
- Combat
- Death
- Cutscene
- Airdrop (only present in PUBG)
- Vehicle (only present in PUBG)

To obtain sequences of sufficient length, ensuring that we also capture continuous actions within the 60-second timeframe, we employed a moderately relaxed timestamp criteria. For instance, in Elden Ring and Dark Souls 3, the "exploration" label relates to periods when the character is actively engaged in the game but not in combat with a boss or mini-boss. In Battlefield V, the "combat" label applies to situations where the player enters an area filled with enemies and initiates combat, or is detected by enemies while attempting to hide themselves. We added, then, another digit (from 0 to 5) to the identification number right before the last element which specifies the time instant 5.2. Concerning the "participant code", as we generated the two final datasets (one with labels and one without) we needed to ensure that all individuals were assigned the same values. To achieve this, we organized the alphanumeric identifiers in alphabetical order before associating them with their respective codes.

5.2 Machine Learning Models

We opted to experiment with various models to determine which one performed best in different scenarios. We trained, validated, and tested these models using sklearn's *StratifiedKFold* function in conjunction with *GridSearchCV*. The models we employed (also taken from sklearn) included *DecisionTreeClassifier*, *RandomForestClassifer*, *LogisticRegression*, *GaussianNB*, and three *DummyClassifier* instances, which served as baselines for comparison with our results.

5.2.1 Nested Cross Validation

Before defining the models' hyperparameters, we need to explain the procedure used to tune them which is a *nested* version of k-fold cross validation. Cross-validation is a statistical technique widely used in machine learning to assess the performance of models. This method is commonly employed in practical machine learning tasks for model selection because of its simplicity and effectiveness in providing unbiased skill estimates compared to other techniques. It is essentially a resampling procedure used to evaluate machine learning models when working with a limited data sample. The key parameter in this procedure is "k", which represents the number of groups the data sample is divided into. This is why it is often referred to as k-fold cross-validation, where "k" can be any chosen value (e.g., k=10 for 10-fold cross-validation).

In applied machine learning, cross-validation is primarily used to estimate how well a model will perform on unseen data. It accomplishes this by using a portion of the available data to evaluate the model's performance, allowing us to gauge its general predictive capabilities beyond the training data. The division of the dataset is done in "k" equal parts, one of them is left out for the validation procedure and the other remaining are used for training. This method is favored for its simplicity and its tendency to provide a more realistic and less overly optimistic estimate of a model's performance compared to simpler techniques like a basic train/test split. The *nested k-fold cross-validation* has the same concept of the normal cross-validation but has two levels, one for hyperparameter tuning and the other to see the generalization perfomance. This method is valuable for two purposes: fine-tuning a model's hyperparameters on a dataset and choosing the best model for a particular task. If the same cross-validation process and dataset are used for both hyperparameter tuning and model selection, it can result in an overly optimistic evaluation of the model's performance. To address this bias, one approach is to embed the hyperparameter optimization step within the model selection process, as happens in the *nested cross-validation.*

In this project, we used this method with both the "k" parameters set to 5.

5.2.2 DecisionTreeClassifier

For this model, we chose to tune a specific set of hyperparameters that we report in the following, together with the possible values tested:

- criterion: gini, entropy
- max_depth: 3, 5, 7
- min_samples_leaf: 1, 3, 5

"Criterion" specifies the classification criteria, also known as the loss function. As already described in Chapter 3, the loss functions we took into consideration for the test are Gini and Entropy. The "max_depth" parameter refers to the maximum depth of the decision tree. If set to "None", the tree nodes are expanded until they all become leaf. This can happen either if all leaves become pure (contain only one class) or until all leaves have fewer samples than the specified "min_samples_split" parameter, which is the minimum number of samples required to split an internal node and is set to the default value of 2. We also specified the parameter "class_weight", which indicates the weight associated with the elements of the classification. By default is set to "None", where all classes are supposed to have weight one. We chose the "balanced" version, where sklearn uses the values of the class set to automatically adjust weights inversely proportional to class frequencies in the input data. "Min_samples_leaf" is the parameter that indicates the minimum number of samples required to be at a leaf node. The tree will consider a split only if there are at least that number of samples in the present leaf.

5.2.3 RandomForestClassifier

For the Random Forest model, we decided to tune the same hyperparameters described for the Decision Tree one (Subsection 5.2.2), with the addition of the number of trees to include in the ensemble. Therefore, we tested:

- n_estimators: 25, 50, 100
- criterion: gini, entropy
- max_depth: 3, 5, 7
- min_samples_leaf: 1, 3, 5

Being very similar to the DecisionTreeClassifier, we decided to maintain the same possible values for tuning. The parameter "n_estimators" accounts for the number of trees in the forest, as reported above. Again we tuned the "class_weight" parameter to "balanced" for the same reason as before.

5.2.4 LogisticRegression

The hyperparameters that we decided to tune for the Logistic Regression model, and their respective possible values, are:

- penalty: l1, l2
- C: 0.1, 1, 10

The parameter "penalty" indicates the regularization function to apply to the loss function and "C" is the inverse of the regularization strength (a smaller value means a stronger regularization). Both concepts were presented and explained in Subsection 3.4.3. "max_iter" is the parameter that specifies the maximum number of iterations the solver has to try to converge. We set it to 1000. "solver" is the algorithm to use in the optimization problem. The default is "lbfgs" but "liblinear" is the best choice for small datasets. It allows the estimation of predictive linear models for classification and regression. From our tests it is the fastest and most accurate of the solvers by a great margin of 20/30% in every situation. "tol" is the tolerance parameter that stops the logistic regression when the difference between the error at the n-th step and the (n-1)-th step is lower than the tolerance value. This speeds up the computation in case the model reaches in a few steps the final accuracy and does not improve much from there to "max_iter". By default, the tolerance value is 0.0001 but we needed to manipulate it to greatly reduce the computation time, so we chose 0.5 and from testing we saw that the results didn't improve once reached past our tolerance value.

5.2.5 GaussianNB

Gaussian Naive Bayes is a Machine Learning classification method that relies on probability and follows a Gaussian distribution. It operates under the assumption that each parameter (often referred to as features) can independently contribute to predicting the output variable. GaussianNB calculates the probability of the output variable belonging to each group based on these independent predictions for each parameter. The final classification is then assigned to the group with the highest probability. For this model we decided to stick with the default parameters because there were limited customization options. Out of the two parameters available, the first one is "priors", which requires prior distribution probabilities of the classes to identify and that we couldn't provide due to the experimental setup. The second parameter, "var_smoothing," determines the portion of the largest variance among all features that is added to variances for calculation stability. We opted to leave this parameter at its default value, due to it not being much effective anyway. We selected this model primarily to establish an additional baseline for comparison with our results. Additionally, we aimed to explore whether individual features had the potential to contribute to participant identification.

5.2.6 DummyClassifier

The "DummyClassifier" is designed to generate predictions that do not consider the input features provided. It serves as a basic benchmark for evaluating the performance of more advanced classifiers. The behavior of this baseline can be customized using the "strategy" parameter. Regardless of the strategy chosen, the predictions made by the "DummyClassifiers" are primarily based on the values of the labels observed during the training phase, while the input features are essentially disregarded. We chose to test three different strategies:

- "most_frequent", the dummy classifier always predicts the most frequent class label among the training samples;
- "stratified", in this case, the dummy classifier generates predictions according to the distribution of the class labels in the training data;
- "uniform", the predictions of the dummy classifier are uniformly random, i.e., each class has an equal probability to be returned by the classifier.

5.3 Experiments and Results

Our experiment had a total of 23 participants of which 85% males and 15% females. In addition, 65% of them have expertise in video games and had little to no problems understanding the games and commands. The remaining had no expertise, which means that they had much more trouble memorizing the buttons and navigating through the games.

Our objective was to assess a broad spectrum of potential combinations of training and testing sets, spanning from the most straightforward to the least correlated ones. In other words, since we wanted to assess the suitability of information retrievable from controllers for player profiling, we had to test data obtained from different games, genres, and controllers. Therefore, we had to perform many experiments to evaluate all the possible cases and obtain transversal results.

5.3.1 Train Test Combinations

Our initial step involved utilizing the unlabeled dataset, allowing us to observe how the models respond to the data including various actions and scenarios without prior knowledge of their labels. This approach aimed to establish a more generalized framework for the classification task. We conducted a series of experiments for each dataset we created, across all the machine learning models and game/genre combinations. Such studies are reported in Table 5.2, in which the train and test columns report the controller used to gather data for that set ("DS4" for Dualshock 4 and "Switch" for Nintendo Switch Pro Controller) and the games column indicates whether those sets refer to the same game, different games belonging to the same genre, or different games of a different genre.

In total, just for the unlabeled dataset, we performed 448 experiments. Every experiment tested one train/test combination of controllers and games for every time interval (1-second to 60-second sequences) and for every one of the 7 models. The same experiments were done for each one of the labeled datasets, as we explain 5.3.3

5.3.2 Unlabeled Dataset Results

In this subsection, we analyze the results achieved by our models as we address the various questions (RQ) we posed ourselves before embarking on this project. We start with the experiment involving the most interconnected elements and

Table 5.2: Experiments conducted for each of the datasets.

TRAIN	**TEST**	**GAMES**
DS4	DS4	Same for train/test
Switch	Switch	Same for train/test
DS4	Switch	Same for train/test
Switch	DS4	Same for train/test
DS4	DS4	Different game Same genre
Switch	Switch	Different game Same genre
DS4	Switch	Different game Same genre
Switch	DS4	Different game Same genre
DS4	DS4	Different game Different genre
Switch	Switch	Different game Different genre
DS4	Switch	Different game Different genre
Switch	DS4	Different game Different genre

proceed sequentially through those with decreasing degrees of correlation, i.e., according to the order reported in Table 5.2.

RQ1: Can we identify players by training and testing on the same controllers and on the same games? In Table 5.3, we reported the top results in terms of average F1-score (and their standard deviation) reached by our models. As the best scoring one is always the random forest classifier, we omit that information from the table. By analysing such outomes it becomes evident that in the experiment involving the most correlated factors, we have achieved acceptable F1-score values.These scores enable us to state that, by employing data gathered from the same controller and the same game, it is possible to identify players with an F1-score that ranges between 77.7% and 87.7%. As anticipated, the results show that longer time intervals yield better outcomes. This implies that the model faces challenges in accurately identifying a player when analyzing short-time sequences. Short intervals contain less information per data point and accumulate more noise compared to longer intervals. The standard deviations added to the F1-scores have a low significance, meaning that the F1 values taken from the outer folds of the nested cross validation are all closely grouped together and therefore more cohesive.

For every case presented in the table, the best model is always the Random Forest Classifier. This suggests that our problem may involve complex, non-linear patterns in the data. As a result, Random Forests excel in capturing these intricate relationships among the features in such situations. Besides that, the dataset's high dimensionality in terms of the number of features could pose

Table 5.3: F1-scores of random forest model for RQ1.

Model	Controller	Game	Interval	F1-score
1	DS4	BF5	30s	0.761±0.024
2	DS4	DS3	60s	0.877±0.031
3	DS4	Elden	60s	0.772±0.094
4	DS4	PUBG	60s	0.824±0.062
5	Switch	BF5	60s	0.758±0.048
6	Switch	DS3	60s	0.864±0.06
7	Switch	Elden	30s	0.777±0.041
8	Switch	PUBG	60s	0.845±0.032

challenges for the Logistic Regression model. This is due to the fact that such a model tends to be prone to overfitting in high-dimensional settings, making it susceptible to being outperformed by the Random Forest model.

In terms of selected hyperparameters, the models in Table 5.3 do not differ much from each other. Since the F1-score values are obtained through a mean of the 5 folds, we chose to highlight only the hyperparameters of the best model out of such folds. Therefore, in the following, when reporting the hyperparameters of a certain model, we always refer to those with the higher score among the folds. Taking in such a sense Model 2, we obtained the following hyperparameters:

- criterion: entropy
- max_depth: 5
- min_samples_leaf: 1
- n_estimators: 100

Which are the same criterion and number of estimators obtained by all the other seven models.

In order to provide a more extensive understanding of the results, we also investigated which features were considered important and, therefore, most used by the various models for their predictions. The results presented in Table 5.4 display the three principal occurrences in terms of feature importance (out of 20 recorded for every experiment and for every outer cross validation fold) for the Decision Tree, Random Forest and Logistic Regression models. Initially, we anticipated that the gyroscope and accelerometer data would significantly contribute to player classification, due to the constant stream of data (something that buttons and triggers do not provide with such diversity) and the uniqueness

Table 5.4: Most frequently used features for our models in the context of RQ1. "Count" shows the number of occurrences while the last column shows the percentage over the total number of features used.

Algorithm	Feature	Count	Percentage
Decision Tree	Gyro Z	1522	27.18%
	Gyro X	1367	24.41%
	Gyro Y	1053	18.80%
Random Forest	Gyro Z	1522	27.18%
	Gyro X	1367	24.41%
	Gyro Y	1053	18.80%
Logistic Regression	Gyro Z	1522	27.18%
	Gyro X	1367	24.41%
	Gyro Y	1053	18.80%

in how people used the controllers. However, what surprised us was the remarkable equality in feature selection across the models. This might appear to be an anomaly, but it is important to note that we not only recorded a list of the most crucial features but also documented the specific importance scores assigned to them by the models. These importance scores varied across different models, indicating that this phenomenon is not due to an error but rather a consistent observation applicable to all models.

RQ2: Can we identify players by training and testing on different controllers but on the same games? To enable training on one controller and testing on the other, we undertook a feature remapping process for the Switch Pro Controller to align its features with those of the Dualshock 4 controller. For instance, the action buttons were transformed from A, B, X, Y to Circle, Cross, Triangle, Square, respectively. However, despite the buttons' translation, we did not achieve valuable results for this experiment. By examining the results for this setting (reported in Table 5.5) it is evident that we are unable to successfully classify players in this scenario as the model that achieves the best performance is always a dummy classifier.

This indicates that the data derived from the controllers exhibits an extremely low level of correlation, to the extent that it completely confuses the algorithms. Investigating the feature importance, we once again encounter the exact same features, with nearly identical percentages: "Gyro Z" at 28.02% occurrence, "Gyro X" at 27.28%, and "Gyro Y" at 17.84%, and this pattern persists across all three algorithms.

Table 5.5: F1-scores of dummy classifiers for RQ2.

Model	**Controller Train**	**Controller Test**	**Game**	**Interval**	**F1-score**
9	DS4	Switch	BF5	30s	0.048±0.012
10	DS4	Switch	DS3	10s	0.046±0.007
11	DS4	Switch	Elden	10s	0.047±0.004
12	DS4	Switch	PUBG	20s	0.045±0.003
13	Switch	DS4	BF5	60s	0.047±0.007
14	Switch	DS4	DS3	10s	0.047±0.006
15	Switch	DS4	Elden	30s	0.047±0.005
16	Switch	DS4	PUBG	20s	0.045±0.004

RQ3: Can we identify players by training and testing on the same controllers and on different games of the same genre? In this scenario, we observe a noticeable enhancement in the model's performance with respect to the previous case, although it still falls short of being a robust player classification system. As can be seen in Table 5.6, the F1-score values range around 20% to 30% accuracy. This signifies a more discernible correlation between different games within the same genre compared to the previous test, where changing the controllers significantly hindered the algorithms.

Table 5.6: F1-scores of the best models for RQ3.

Model	**Controller**	**Game Train**	**Game Test**	**Interval**	**Model Type**	**F1-score**
17	DS4	BF5	PUBG	60s	LR	0.244±0.032
18	DS4	DS3	Elden	60s	LR	0.186±0.012
19	DS4	Elden	DS3	30s	LR	0.231±0.019
20	DS4	PUBG	BF5	10s	LR	0.304±0.008
21	Switch	BF5	PUBG	20s	RF	0.328±0.018
22	Switch	DS3	Elden	60s	RF	0.286±0.014
23	Switch	Elden	DS3	30s	LR	0.29±0.011
24	Switch	PUBG	BF5	10s	RF	0.261±0.009

In contrast to the previous two sets of experiments, in this scenario, Logistic Regression (LR) emerges as the most frequently used algorithm. Random Forest (RF) surpasses it only for Models 21 and 22. Again, longer time intervals are favored over shorter ones, and the standard deviation is sufficiently low, indicating stability across different folds.

While for the Random Forest the hyperparameters have the same variance as the ones in Table 5.3, for the Logistic Regression, the only element that changes through experiments is “C”, which correlates to the strength of the regularization. The only other hyperparameter that is not fixed from the start is the “penalty” which remains constant throughout experiments on the L1 (or LASSO) regularization.

Concerning the features’ importance, we once again found gyroscope-related features marked as the most relevant ones, having occurrences of 27.18% for “Gyro Z”, 24.41% for “Gyro X”, and 18.80% for “Gyro Y”. This means that no matter the question we are trying to answer, models would still use the same features to classify players.

RQ4: Can we identify players by training and testing different controllers and on different games of the same genre? It is interesting to see that the statement made for Table 5.5 seems to be valid also for the set of experiments displayed in Table 5.7. Thus, training and testing on different controllers hinder the capability of the algorithms to reach an acceptable classification rate, making all the selected models score worse than dummy classifiers.

Table 5.7: F1-scores of best models for RQ4.

Model	Controller Train	Game Train	Controller Test	Game Test	Interval	F1-score
25	DS4	BF5	Switch	PUBG	1s	0.055±0.002
26	DS4	DS3	Switch	Elden	20s	0.049±0.004
27	DS4	Elden	Switch	DS3	60s	0.049±0.014
28	DS4	PUBG	Switch	BF5	30s	0.049±0.009
29	Switch	BF5	DS4	PUBG	60s	0.052±0.01
30	Switch	DS3	DS4	Elden	60s	0.05±0.008
31	Switch	Elden	DS4	DS3	30s	0.048±0.005
32	Switch	PUBG	DS4	BF5	20s	0.048±0.007

In this scenario, all the models once again indicate the Dummy Classifier as the best algorithm, with the exception of Model 25, where the Random Forest is ranked as the top-performing algorithm. However, when analyzing the scores, there is no discernible pattern or meaningful reason why this particular experiment deviates from the others.

RQ5: Can we identify players by training and testing on the same controller and on different games of different genres? In this section, the outcomes are similar to those of Table 5.6 but are generally less promising. Nevertheless, they do manage to outperform the basic dummy classifiers. The results in Table 5.8 are lower with respect to the precedent cases as we are now training on one genre of games and testing on another. These F1-score values highlight the existing relationship between different game genres. Hence, as expected, it is empirically evident that player classification is influenced not only by the controller used but also by the specific types of games being played.

Table 5.8: F1-scores of best models for RQ5

Model	**Controller**	**Game Train**	**Game Test**	**Interval**	**Model Type**	**F1-score**
33	DS4	BF5	DS3	3s	LR	0.227±0.008
34	DS4	DS3	BF5	3s	LR	0.233±0.009
35	DS4	Elden	PUBG	10s	LR	0.131±0.011
36	DS4	PUBG	Elden	3s	RF	0.14±0.003
37	DS4	BF5	Elden	3s	RF	0.212±0.008
38	DS4	DS3	PUBG	3s	RF	0.167±0.002
39	DS4	Elden	BF5	20s	RF	0.153±0.016
40	DS4	PUBG	DS3	60s	RF	0.198±0.035
41	Switch	BF5	DS3	30s	RF	0.307±0.011
42	Switch	DS3	BF5	10s	RF	0.238±0.002
43	Switch	Elden	PUBG	20s	LR	0.101±0.013
44	Switch	PUBG	Elden	5s	RF	0.13±0.006
45	Switch	BF5	Elden	10s	RF	0.257±0.008
46	Switch	DS3	PUBG	5s	RF	0.269±0.01
47	Switch	Elden	BF5	20s	LR	0.156±0.007
48	Switch	PUBG	DS3	10s	LR	0.169±0.021

Again, similarly to the other set of experiments cited above, between the best algorithms there is a mixture of Random Forests and Logistic Regressions. And, same as before, the hyperparameters slightly change depending on the model (like "C" for Logistic Regression and "max_depth" for Random Forest). A noteworthy trend in the data analysis is the significant prevalence of short time intervals, particularly the 3-second ones. This suggests that there might be substantial distinctions between game genres when focusing on quick actions and swift shifts in player behavior. Shorter intervals offer a finer-grained perspective of the data, but it's important to acknowledge that they tend to introduce more noise compared to longer intervals. However, we should approach this observation cautiously as the F1-scores remain low and are thus inadequate for a reliable classification. The trend we are noticing might be a misleading pattern in the data. Additionally, even in the case of the best-performing model, i.e., Model 41, it still favors longer time intervals.

Lastly, once again we have that the top-feature occurrences concern gyroscope data, reaching 27.18% for "Gyro Z", 24.41% or "Gyro X", and 18.80% for "Gyro Y".

RQ6: Can we identify players by training and testing on different controllers and on different games of different genres? In this scenario, we're dealing with the most uncorrelated data and, as anticipated based on the patterns observed in the previous sets, it is no surprise that we are obtaining similarly low values, primarily from the dummy classifiers. All the insights and trends discussed in the earlier questions are consistent with the results obtained in this final set of experiments for the unlabeled dataset.

Table 5.9: F1-scores of best results for RQ6.

Model	Controller Train	Game Train	Controller Test	Game Test	Interval	F1-score
49	DS4	BF5	Switch	DS3	20s	0.05±0.007
50	DS4	DS3	Switch	BF5	5s	0.044±0.004
51	DS4	Elden	Switch	PUBG	5s	0.044±0.003
52	DS4	PUBG	Switch	Elden	60s	0.055±0.011
53	DS4	BF5	Switch	Elden	1s	0.055±0.002
54	DS4	DS3	Switch	PUBG	20s	0.045±0.008
55	DS4	Elden	Switch	BF5	60s	0.049±0.014
56	DS4	PUBG	Switch	DS3	30s	0.049±0.009
57	Switch	BF5	DS4	DS3	60s	0.05±0.014
58	Switch	DS3	DS4	BF5	30s	0.046±0.003
59	Switch	Elden	DS4	PUBG	60s	0.048±0.009
60	Switch	PUBG	DS4	Elden	5s	0.044±0.003
61	Switch	BF5	DS4	Elden	60s	0.052±0.01
62	Switch	DS3	DS4	PUBG	30s	0.049±0.009
63	Switch	Elden	DS4	BF5	30s	0.048±0.005
64	Switch	PUBG	DS4	DS3	20s	0.048±0.007

Features' Relevance

In order to further investigate the features selected by our models, we present a consolidated display that combines all the features marked as relevant by each of the tested models, providing an overview of how each feature from both controllers has been assessed in terms of importance. Considering the fact that all algorithms choose the features in the same way, we present only the occurrences for the

Random Forest (which is the one that usually performs best), which are reported in Table 5.10.

Table 5.10: A display of all features' importance combined.

Feature	**Count**	%
Gyro Z	12347	27.56%
Gyro X	11570	25.83%
Gyro Y	8195	18.29%
Acc X	4465	9.97%
R1	1671	3.73%
Acc Z	932	2.08%
Acc Y	673	1.50%
Stick RX	438	0.98%
Stick RZ	412	0.92%
Hat	400	0.89%
Stick RY	282	0.63%
L2	280	0.62%
L3	276	0.62%
Stick LX	248	0.55%
LZ	216	0.48%
Square	196	0.44%
R3	176	0.39%
Plus	168	0.38%
Triangle	168	0.38%
Circle	165	0.37%
R	152	0.34%
Hat Left	140	0.31%
Stick LY	132	0.29%
Hat Down	132	0.29%
Y	120	0.27%
L1	115	0.26%
Cross	114	0.25%
Minus	100	0.22%
Hat Up	96	0.21%
R2	93	0.21%
A	72	0.16%
Hat Right	64	0.14%
X	60	0.13%
RZ	48	0.11%
L	44	0.10%
B	24	0.05%
Stick LZ	12	0.03%
Options	4	0.01%

Apart from the top three features, which are evidently the most influential, there are only a handful of others that significantly impact the models. Among these, we find the other three expected features: “Acc X”, “Acc Z”, and “Acc Y”, which correspond to the three axes of the accelerometer. Notably, the feature R1, representing the right shoulder button on the Dualshock 4 controller, ranks high in importance. This button is particularly crucial for combat actions in games like Elden Ring and Dark Souls 3, specifically associated with the Light Attack action. Beyond these features, all others have a minimal impact on the models, contributing with less than 1% occurrence, rendering them nearly irrelevant. Interestingly, we had anticipated the Stick Axes to hold higher positions in terms of importance, as our observations during gameplay suggested that both experienced and inexperienced players used the sticks in unique and distinctive ways that we expected to influence the results more significantly.

5.3.3 Labeled Dataset Results

We replicated all the previous experiments for two out of the six labels we identified during the screen recording analysis, which are “Exploration” and “Combat”. In this section, we aim to investigate whether training and testing on specific segments of the dataset yield different results compared to the previous approach, where we trained on the entire dataset without distinguishing between labels. Specifically, we wanted to see if this method would produce better results. The research questions are the same as in the unlabeled case and, hence, also the experiments to be conducted. In other words, for each datasets derived from the above-mentioned labels, we repeated the experiments reported in Table 5.2. It is necessary to specify that we could not utilize the 60-second sequences for training and testing due to the limited number of data instances where the same action persisted consecutively for a full 60 seconds, particularly in combat sequences. Hence, for each label, we derived six datasets.

Exploration Label

RQ1 The F1-sores obtained in this case (Table 5.11) result to be comparable with the ones of its unlabeled counterpart (Table 5.3).

One thing we want to highlight is the fact that, by employing labeled data, models are able to achieve more or less the same performance using shorter sequences with respect to the unlabeled case. Nevertheless, such “shorter” sequences are well-crafted and separated for meaningful and complete actions. The

Table 5.11: F1-scores of random forest model with "Exploration" label relative to RQ1.

Model	Controller	Game	Interval	F1-score
1	DS4	BF5	20s	0.785±0.07
2	DS4	DS3	20s	0.848±0.039
3	DS4	Elden	20s	0.814±0.037
4	DS4	PUBG	20s	0.808±0.03
5	Switch	BF5	10s	0.734±0.033
6	Switch	DS3	30s	0.781±0.047
7	Switch	Elden	10s	0.783±0.007
8	Switch	PUBG	30s	0.86±0.055

key point to note here is that even with the constraint of using shorter time intervals, the model still achieved excellent results. This suggests that if we had a sufficient amount of data for the 60-second sequences, we could potentially achieve even higher performance.

In terms of the algorithms, Random Forest continues to outperform the others in all cases. However, there are some variations in the hyperparameters compared to previous experiments. Specifically, in addition to "min_sample_leaf" and "max_depth," we now observe changes in the "n_estimators" parameter, which varies between two values (50 and 100) instead of being fixed at 100 as in the unlabeled experiments. Furthermore, the "criterion" parameter now favors the "Gini" loss in models like 2 and 6, whereas it was different in previous experiments.

Coming to analyze the features' importance, as expected "Gyro Z" and "Gyro X" have still the highest occurrence (respectively 38.18% and 20.57%), but now instead of "Gyro Y" in third place we have "L3" at 12.88% and refers to the button on the Left Stick of the controllers, which in all games is the command to start sprinting. The occurrences are still equal for all three algorithms.

RQ2 In the same way as the unlabeled dataset (Table 5.5), all the tested models are overperformed by dummy classifiers, and the results are no better or worse.

The features' importance this time returns to the previously seen ranking with "Gyro Z" at 28.81%, "Gyro Y" at 25.62% and "Gyro X" at 19.67%.

RQ3 In this case, we came upon a change in the pattern from a labeled dataset with respect to the unlabeled one. Indeed, as can be seen in Table 5.12, the F1-scores obtained in this case are overall lower than the ones of its unlabeled equivalent (Table 5.6). This, could be due to the absence of the 60-seconds sequences, but making reference to Model 19, both experiments prefer the 30-second interval while there is a clear difference between scores (0.231 against 0.179). Alternatively, the performance worsening could be because labeled data points of inferior value.

Table 5.12: F1-scores of best models with "Exploration" label relative to RQ3.

Model	**Controller**	**Game Train**	**Game Test**	**Interval**	**Model Type**	**F1-score**
17	DS4	BF5	PUBG	10s	LR	0.231±0.018
18	DS4	DS3	Elden	1s	RF	0.135±0.004
19	DS4	Elden	DS3	30s	LR	0.179±0.019
20	DS4	PUBG	BF5	10s	LR	0.253±0.009
21	Switch	BF5	PUBG	10s	RF	0.28±0.01
22	Switch	DS3	Elden	20s	RF	0.278±0.015
23	Switch	Elden	DS3	10s	RF	0.226±0.02
24	Switch	PUBG	BF5	5s	RF	0.245±0.006

Regarding the best algorithms, we have a mixture of Random Forest and Logistic Regression. In terms of features' importance we have, similar to the first question with this label, "Gyro X" at 33.52%, "Gyro Z" at 27.18% and "L3" at 11.55%.

RQ4 Virtually, there is no difference from Table 5.7. Dummy classifiers represent the upper bound of the performance of the whole set, achieving an F1-score of 5%. In this scenario, features' importance gives much more power to one axis of the gyroscope: "Gyro X" at 50.83%, then "Gyro Z" at 23.52%, and "Gyro Y" at 6.58%.

RQ5 During the experiments for this question (Table 5.13), something unexpected occurred with the *select_features* function. This function, which is designed to eliminate irrelevant features and retain only those essential for classification, ended up removing all features except for two: "Acc X" and the "A" button. "Acc X" was utilized exclusively by the models trained on Dualshock 4 data, while the "A" button was used for the Switch Pro Controller. Consequently, the features' importance is both 50%. The achieved results are not easily comparable (to Table 5.8) because, while overall the general scores are similar, in some cases like Model 35 we have an 11% increase in value, or in Model 46 we have a decrement of 16%. Maybe, re-iterating the experiments without the use of *select_features* could change the results and make them more comparable, with the expectation to see the three axes of the gyroscope taking the lead in feature importance.

Table 5.13: F1-scores for best models with "Exploration" label relative to RQ5.

Model	Controller	Game Train	Game Test	Interval	Model Type	F1-score
33	DS4	BF5	DS3	3s	LR	0.213±0.003
34	DS4	BF5	Elden	1s	RF	0.249±0.005
35	DS4	DS3	BF5	3s	LR	0.245±0.007
36	DS4	DS3	PUBG	1s	RF	0.159±0.007
37	DS4	Elden	BF5	30s	DT	0.15±0.029
38	DS4	Elden	PUBG	20s	DT	0.119±0.03
39	DS4	PUBG	DS3	10s	RF	0.192±0.01
40	DS4	PUBG	Elden	1s	RF	0.134±0.003
41	Switch	BF5	DS3	20s	RF	0.237±0.019
42	Switch	BF5	Elden	5s	RF	0.243±0.006
43	Switch	DS3	BF5	5s	RF	0.245±0.01
44	Switch	DS3	PUBG	3s	RF	0.269±0.006
45	Switch	Elden	BF5	1s	DT	0.153±0.014
46	Switch	Elden	PUBG	1s	DT	0.109±0.017
47	Switch	PUBG	DS3	1s	LR	0.17±0.003
48	Switch	PUBG	Elden	30s	RF	0.141±0.011

For the first time, in this instance, we have between the list of best algorithms, the Decision Tree (DT) for models 37, 38, 45, 46. Its hyperparameters are set as the same as the Random Forest's and the only element that varies is "min_samples_leaf". The rest of the algorithms is again a mixture between Random Forest and Logistic Regression.

RQ6 The results of the final experiment for the "Exploration" label do not bear any differences from its counterpart (Table 5.9), while the frequency of features is strangely changes: "Acc Z" at 23.33%, "Acc X" at 16.67% and "Triangle" button at 6.67%. It is important to know that the "Triangle" button refers both to the Dualshock 4 and to the "X" button on the Switch Pro Controller, since this being a set of experiments with mixed controllers for train/test we have the buttons of the Switch Pro Controller remapped to those of the Dualshock 4. In all four games the "Triangle" button is used to change weapons or rearrange their hold in hand (double-wielded or single-hand combat for DS3 and Elden Ring).

Combat Label

RQ1 Following the pattern of the “Exploration” dataset (Table 5.11) and the unlabeled dataset (Table 5.3), we have the best outcomes on the question with the most correlated elements. Such favorable results can be seen in Table 5.14. Comparing the three sets of experiments there is no clear winner in terms of F1-scores. One trend we can appreciate is the progressive decrease in interval length. Here, half of the experiments have some of the shortest time intervals.

Table 5.14: F1-scores for random forest models with “Combat” label relative to RQ1.

Model	Controller	Game	Interval	F1-score
1	DS4	BF5	10s	0.83±0.024
2	DS4	DS3	10s	0.825±0.041
3	DS4	Elden	3s	0.779±0.017
4	DS4	PUBG	1s	0.824±0.021
5	Switch	BF5	10s	0.818±0.013
6	Switch	DS3	10s	0.838±0.008
7	Switch	Elden	3s	0.732±0.023
8	Switch	PUBG	1s	0.833±0.01

As expected the algorithm chosen on all the experiments is the Random Forest and all the hyperparameters have the same behavior as in the other cases, apart from Model 2 in which the preferred loss is “Gini”.

The feature importance seems more spread out, with the highest three features being: “Gyro Z” at 17.61%, “Gyro X” at 17.44%, and “Acc X” at 13.13%.

RQ2 With no surprise, even here the algorithms can’t overtake the Dummy Classifiers. In the end, the classification for this particular problem can be deemed negative.

The features’ importance are: “Gyro Y” at 25.42%, “Gyro Z” at 22.90%, and “Gyro X” at 21.29%.

RQ3 As with Table 5.6 and Table 5.12 the results hover around 20% to 30%. There are no significant differences, apart from Model 21 which has a surprisingly high F1-score, compared to the others. The time intervals are still on the longer side (considering the absence of the 60-seconds sequences).

The algorithms that we encounter in this set are predominantly Random Forests, but there are a few instances of Logistic Regression and Decision Tree.

The features’ importance a similar pattern to what happened for Table 5.13 where the function *select_features* removed most of the features leaving only a

Table 5.15: F1-scores of best models with “Combat” label relative to RQ3.

Model	Controller	Game Train	Game Test	Interval	Model Type	F1-score
17	DS4	1	4	20s	LR	0.215±0.036
18	DS4	2	3	1s	RF	0.197±0.019
19	DS4	3	2	10s	RF	0.172±0.009
20	DS4	4	1	10s	DT	0.198±0.017
21	Switch	1	4	20s	RF	0.373±0.035
22	Switch	2	3	10s	RF	0.236±0.013
23	Switch	3	2	20s	RF	0.25±0.036
24	Switch	4	1	1s	LR	0.182±0.002

few left. This time the occurrences are: “A” button at 50%, “Acc X” at 39.58%, and “Gyro Z” at 3%.

RQ4 We cannot find any relevant differences between this set with the “combat” label, and the other two sets of experiments (“exploration” and unlabeled). There are two cases in which the algorithms (one Random Forest and one Decision Tree) pass the Dummy Classifiers but the F1-scores remain under 10%, which is definitely too low to be considered a classifier.

The features’ importance are: “Gyro Z” at 19.08%, “Gyro X” at 15.71%, and “Stick RX” at 10.22%. The “Stick RX” is the right stick on the controller and for all games is used to look around and move the camera.

RQ5 This is the same situation of Table 5.13 where during the experiments the *select_features* function removed all but the same two features: “Acc X” and “A” buttons. It would be ideal to try and repeat the whole set of experiments without the aid of that function. The results in this case are overall slightly worse than both (Tables 5.8 and 5.13) its counterparts.

The algorithms used in this case are, for the major part, Random Forests, and then a few instances of Logistic Regression and Decision Tree. The features’ importance, as already stated, are: “Acc X” at 53.33% and “A” button at 46.47%.

RQ6 In this final set of experiments, we reinstate that being the question with least correlated items, it is not possible, at this stage of project development, to produce a player classifier. Its results are consistent with its counterpart (Table 5.9) and also with the corresponding question with the “Exploration” label. Every algorithm falls short of the Dummy Classifiers.

Table 5.16: F1-scores of best models with "Combat" label relative to RQ5.

Model	Controller	Game Train	Game Test	Interval	Model Type	F1-score
33	DS4	1	2	1s	LR	0.161±0.006
34	DS4	1	3	1s	RF	0.155±0.008
35	DS4	2	1	1s	LR	0.153±0.005
36	DS4	2	4	1s	RF	0.149±0.015
37	DS4	3	1	5s	DT	0.15±0.017
38	DS4	3	4	5s	DT	0.121±0.012
39	DS4	4	2	3s	RF	0.155±0.007
40	DS4	4	3	1s	RF	0.12±0.003
41	Switch	1	2	10s	RF	0.209±0.005
42	Switch	1	3	1s	RF	0.181±0.005
43	Switch	2	1	1s	RF	0.221±0.004
44	Switch	2	4	5s	RF	0.221±0.01
45	Switch	3	1	5s	RF	0.153±0.008
46	Switch	3	4	1s	DT	0.088±0.024
47	Switch	4	2	10s	DT	0.116±0.023
48	Switch	4	3	1s	RF	0.102±0.006

The features' importance are: "Stick LY" at 14.17%, "Gyro Y" at 12.50%, and "Gyro X" at 10.83%. The "Stick LY" feature refers to the left stick which in all games is used for character movement.

Features' Relevance

In this section, we present the features' importance of all the experiments combined, divided into the two labels we tested. Firstly, we display the occurrences with the anomalies of the Fifth Question then, for a more fitting scenario, we present the lists without those cases.

In the same manner, as the unlabeled case, we combined all the features of the Random Forest Classifier, since for all three algorithms the features' importance is exactly the same. It is worth noticing that the features "Acc X" and "A" buttons are on the top positions (Table 5.17), which is expected because of the case discussed for the Fifth Questions (Paragraphs 5.3.3 and 5.3.3). While it could be possible that the "Acc X" feature would be still fairly important, our analysis of the unlabeled version (Table 5.10), reveals that buttons are not the most prominent of features. Thus seeing one so high definitely catches the eye. While for Table 5.18, as predicted, has "Acc X" still a fairly high importance but the "A" button has lost many positions in the ranking. It is interesting to notice

Table 5.17: Top ten of the features' importance from labeled datasets combined with anomalies.

Exploration			**Combat**		
Feature	**Count**	**%**	**Feature**	**Count**	**%**
Gyro X	7922	19.09%	Acc X	9203	22.92%
Acc X	7886	19.00%	A	7726	19.24%
A	5686	13.70%	Gyro X	4054	10.10%
Gyro Z	5406	13.03%	Gyro Z	3873	9.65%
Gyro Y	2734	6.59%	Gyro Y	3346	8.33%
Acc Z	2444	5.89%	Stick LY	1584	3.95%
Stick RX	1367	3.29%	Stick RX	1564	3.90%
L3	1356	3.27%	R2	1078	2.68%
Triangle	742	1.79%	Circle	1069	2.66%
Circle	665	1.60%	Stick LX	956	2.38%

that the gyroscope's axes have the top three spots, as for the unlabeled dataset.

The key distinction between the two labels lies in the distribution of feature importance. For "Combat", the importance values are more evenly spread out across features, with even the less important ones having some significance. Conversely, for "Exploration", the initial features hold more prominence compared to the rest, indicating their greater relevance to this label. When we compare these lists to the unlabeled dataset version (Table 5.10), we observe some interesting distinctions. In the "Combat" list, besides the gyroscope and accelerometer, both the Left and Right Sticks hold greater importance. On the other hand, the "Exploration" list emphasizes the Right Stick, which is typically used for looking around, and certain action buttons that see more usage during exploration, such as "Triangle", "L3", and "Circle".

Table 5.18: Features' importance of labeled datasets combined without anomalies.

Exploration			**Combat**		
Feature	**Count**	**%**	**Feature**	**Count**	**%**
Gyro X	7922	26.14%	Gyro X	4054	15.99%
Gyro Z	5406	17.84%	Gyro Z	3873	15.28%
Gyro Y	2734	9.02%	Gyro Y	3346	13.20%
Acc Z	2444	8.07%	Acc X	1703	6.72%
Acc X	2286	7.54%	Stick LY	1584	6.25%
Stick RX	1367	4.51%	Stick RX	1564	6.17%
L3	1356	4.47%	R2	1078	4.25%
Triangle	742	2.45%	Circle	1069	4.22%
Circle	665	2.19%	Stick LX	956	3.77%
R2	642	2.12%	Stick RY	853	3.37%
L1	582	1.92%	R1	817	3.22%
Stick RY	564	1.86%	L2	581	2.29%
Stick LY	556	1.83%	Acc Z	515	2.03%
Square	499	1.65%	Acc Y	478	1.89%
R1	433	1.43%	L3	465	1.83%
Acc Y	416	1.37%	L1	454	1.79%
Stick LX	319	1.05%	A	426	1.68%
Cross	315	1.04%	Square	342	1.35%
R3	114	0.38%	Triangle	324	1.28%
L2	102	0.34%	L	237	0.93%
Options	98	0.32%	Cross	169	0.67%
A	86	0.28%	Hat	82	0.32%
Hat Up	86	0.28%	Hat Right	77	0.30%
Touch Pad	78	0.26%	Touch Pad	35	0.14%
Share	69	0.23%	Stick RZ	33	0.13%
Label	65	0.21%	Hat Up	29	0.11%
R	52	0.17%	Hat Left	28	0.11%
Stick RZ	49	0.16%	Home	25	0.10%
L	44	0.15%	Hat Down	24	0.09%
RZ	44	0.15%	R3	22	0.09%
Hat Left	40	0.13%	Label	19	0.07%
Home	38	0.13%	Share	17	0.07%
Hat	22	0.07%	Y	17	0.07%
Hat Right	14	0.05%	RZ	15	0.06%
Stick LZ	10	0.03%	Options	10	0.04%
Plus	10	0.03%	R	9	0.04%
Minus	8	0.03%	LZ	8	0.03%
Y	6	0.02%	B	8	0.03%
Hat Down	6	0.02%	Stick LZ	2	0.01%
X	6	0.02%	X	1	0.00%
B	4	0.01%			
LZ	4	0.01%			

Chapter 6

Discussion

6.1 Applications

The results we have obtained demonstrate the feasibility of identifying players based on their behavioral data from game controllers. This suggests that an enhanced version of our system could be effectively applied across various video game genres. In practice, this system could find broad utility within the gaming industry, particularly for video game companies that produce games within the same genre. Additionally, since the controllers are typically used on their respective gaming consoles, it is uncommon for an individual to switch between different controllers, unless they are playing on a computer.
The ability to recognize individuals solely through their controller data has significant implications for the gaming community. While there are potential commercial applications such as game companies profiling gamers and offering personalized advertising, our primary focus is not on such uses. Instead, a more critical and impactful application lies in enhancing the battle against cheating, harassment, scams, and other malicious player behaviors. With our system in place, if a player is banned for violating game rules or engaging in misconduct, they can be identified even if they create a new account, leading to a more effective deterrent against unlawful activities. This permanent restriction from playing the game can serve as a substantial deterrent for ill-minded individuals who engage in disruptive behaviors. Moreover, this kind of player profiling can help identify less grave misbehaviors that still negatively impact other players' gaming experience, like cheating, smurfing, and boosting. Cheating, as already explained in 3.2, is the act of using unauthorized methods, tools, or practices to gain an unfair advantage or manipulate the outcome of a video game. Smurfing,

is gaming term that refers to the practice of experienced or highly skilled players creating new or secondary accounts with lower skill levels or ranks in order to compete against less-experienced opponents. Boosting is the practice of using external assistance or services to improve a player's in-game performance, rank, or achievements. This can involve paying another player or service provider to play on one's behalf. Another natural consequence of successful player classification is biometric authentication, where if, for example, someone hacked into an account they could be identified as not the person who originally owns the profile, just by playing, and notify the real creator that a breach might have occurred.

6.2 Limitations

In this subsection, we aim to report all the limitations encountered during our research study. Firstly, our sample size has not a great amplitude and might have influenced our findings. The minimum number of participants normally would be 30 people.

In addition to the sample size, the demographic composition of our participants, who were exclusively young adults aged 18 to 30, might not be representative of the full gamers population. Moreover, it is important to note that the real gender distribution among gamers was not captured by our sample, as indicated by [33], showing that at least 43% of online gamers are female.

Additional limitations stem from the substantial computational resources required for both feature extraction and model training. Despite employing two computers equipped with mid-range processors, the extensive computational demands meant that these machines were occupied for more than a month, rendering them unavailable even for routine tasks during this period.

Moreover, given the novelty of this research, determining the optimal approach for feature extraction and model selection was challenging. Consequently, we adopted a more generalized strategy, experimenting with various well-known algorithms and libraries to assess our direction. While results for simpler tasks were reasonably acceptable, we encountered significantly lower accuracies in more uncorrelated scenarios.

In this study, our experimentation was limited to four games from two genres, and we focused on just two labels separately. Perhaps the restricted selection of games in our study posed a limitation, or it is possible that the chosen games did not exhibit strong correlations. Another factor could be the dissimilarity in game genres. Moreover, when creating our datasets, we limited ourselves to employing seven specific time intervals. It is worth noting that we achieved higher performance with longer sequences, ranging from 30 seconds to 60 seconds. This suggests that considering additional durations, such as 40 seconds or 50 seconds, or even sessions lasting over a minute, could have been beneficial. Furthermore, due to the extended duration required for testing with participants, we opted for 15-minute gameplay sessions. Exploring longer sessions, such as 30 minutes or even an hour of continuous gameplay, might have resulted in more promising outcomes.

Chapter 7

Final Remarks

7.1 Future Works

Also accounting for the ideas outlined when explaining the limitations of this work (§6.2), we now expose a list of engaging research directions and objectives to pursue in the future:

- Expand the selection of tested games to understand the system's limitations across a broader range of titles.
- Enhance the feature extraction process to make it more comprehensive and possibly tailor it for different game types.
- Wide the selection of algorithms and tuning parameters, even exploring unsupervised deep learning methods.
- Explore the additional labels not experimented with in this study, and test scenarios that involve interactions between them.
- Investigate how the system performs in the same game but in different matches or sections.
- Increase the number of participants to enhance the robustness of the system and obtain a more representative sample of the population.
- Conduct experiments that exclusively utilize the gyroscope and accelerometer sensors, to lower the computational complexity.
- Run tests without the gyroscope and accelerometer sensors data to gauge their impact.

- Evaluate the system's performance on gaming sessions of varying durations, both shorter and longer than the 15-minute intervals used in this study.

In essence, the goal is to identify the system's optimal environment and continually improve its capabilities, considering the vast array of potential scenarios beyond what we have explored thus far.

7.2 Conclusions

In this thesis, we proposed a system that employs machine learning (ML) to identify video gamers by their playstyle based solely on controllers' data. The experiment was set up to make every participant play for 15 minutes at each of the four selected games with one controller and then replay them with the second controller. We registered the data while they were playing and also recorded their screen for additional tests. The dataset retrieved from these controllers was then used to create seven datasets consisting of time sequences of different lengths, from 1-second to 60-second intervals. Furthermore, we performed a variance analysis on the datasets to eliminate low-variance features. Finally, we fed this elaborated dataset to our ML models and tested various scenarios to understand how it would behave in different environments. We then repeated the same tests on datasets created with two labels manually extrapolated from the screen recordings. The labels refer to a specific action or situation in the gameplay, like exploration, combat, or death. This procedure allowed us to evaluate whether the system's performance improved or deteriorated when training and testing these more specific datasets. Our findings revealed that the system achieved reasonable classification accuracy only in scenarios where training and testing were conducted on the same controllers and games, indicating a high degree of correlation. However, in scenarios with lower correlation, the system struggled to provide valid classification scores, falling short of being a reliable classifier. Even with these shortcomings in mind, we can confidently state that this system has the potential to be hugely improved in the future.

Bibliography

[1] IMARC Group. Gaming market: Global industry trends, share, size, growth, opportunity and forecast 2023-2028. https://www.imarcgroup.com/gaming-market, 2022. Accessed: August, 2023.

[2] Christina Gough. Revenue of the global esports market 2020-2025. https://www.statista.com/statistics/490522/global-esports-market-revenue, 2023. Accessed: August 2023.

[3] Warburton. The health benefits of active gaming: Separating the myths from the virtual reality. https://link.springer.com/article/10.1007/s12170-013-0322-0, 2013.

[4] Mark R Johnson and Jamie Woodcock. The impacts of live streaming and twitch.tv on the video game industry. *Media, Culture & Society*, 41(5):670–688, 2019.

[5] Alaa Qaffas. An operational study of video games' genres. 09 2020.

[6] Aleksandar Klasnja, Natasa Milenovic, Sonja Lukac, Aleksandar Knezevic, Jelena Klasnja, and Vedrana Karan Rakic. The effects of regular physical activity and playing video games on reaction time in adolescents. *International Journal of Environmental Research and Public Health*, 19(15), 2022.

[7] Mohamed Chawki. Cybercrime in the context of covid-19. In Kohei Arai, editor, *Intelligent Computing*, pages 986–1002, Cham, 2021. Springer International Publishing.

[8] Harjinder Singh Lallie, Lynsay A. Shepherd, Jason R.C. Nurse, Arnau Erola, Gregory Epiphaniou, Carsten Maple, and Xavier Bellekens. Cyber security in the age of covid-19: A timeline and analysis of cyber-crime and cyber-attacks during the pandemic. *Computers Security*, 105:102248, 2021.

[9] Adam Epstein. The pandemic has turned everyone into gamers. `https://qz.com/1904276/everyone-is-playing-video-games-during-the-pandemic`, 2020. Accessed: August 2023.

[10] TransUnion. 2022 global digital fraud trends report. `https://content.transunion.com/v/2022-global-digital-fraud-trends-report`, 2022. Accessed: August, 2023.

[11] Matt Zajechowski. Study: Majority of gamers say they've witnessed racism and hate speech. `https://preply.com/en/blog/hate-speech-and-bullying-in-video-games/`, 2022. Accessed: July, 2023.

[12] Jacob Leon Kröger, Philip Raschke, Jessica Percy Campbell, and Stefan Ullrich. Surveilling the gamers: Privacy impacts of the video game industry. *Entertainment Computing*, 44:100537, 2023.

[13] Zhao Wang, Anna Sapienza, Aron Culotta, and Emilio Ferrara. Personality and behavior in role-based online games, 2019.

[14] Carl Symborski, Gary M. Jackson, Meg Barton, Geoffrey Cranmer, Byron Raines, and Mary Magee Quinn. The use of social science methods to predict player characteristics from avatar observations. In Muhammad Aurangzeb Ahmad, Cuihua Shen, Jaideep Srivastava, and Noshir Contractor, editors, *Predicting Real World Behaviors from Virtual World Data*, pages 19–37, Cham, 2014. Springer International Publishing.

[15] Pier Paolo Tricomi, Lisa Facciolo, Giovanni Apruzzese, and Mauro Conti. Attribute inference attacks in online multiplayer video games: A case study on dota2. In *Proceedings of the Thirteenth ACM Conference on Data and Application Security and Privacy*, CODASPY '23, page 27–38, New York, NY, USA, 2023. Association for Computing Machinery.

[16] Pier Paolo Tricomi, Federica Nenna, Luca Pajola, Mauro Conti, and Luciano Gamberini. You can't hide behind your headset: User profiling in augmented and virtual reality. *IEEE Access*, 11:9859–9875, 2023.

[17] Joseph Newman, Joseph W. Jerome, and Christopher J. Hazard. Press start to track?: Privacy and the new questions posed by modern videogame technology. 2014.

[18] Florian Baumann, Dominik Emmert, Hermann Baumgartl, and Ricardo Buettner. Hardcore gamer profiling: Results from an unsupervised learning approach to playing behavior on the steam platform. *Procedia Computer Science*, 126:1289–1297, 2018. Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 22nd International Conference, KES-2018, Belgrade, Serbia.

[19] Olatz Lopez-Fernandez, A. Jess Williams, and Daria J. Kuss. Measuring female gaming: Gamer profile, predictors, prevalence, and characteristics from psychological and gender perspectives. *Frontiers in Psychology*, 10, 2019.

[20] Jessica Williams, Rhyse Bendell, Stephen M. Fiore, and Florian Jentsch. Towards a conceptual framework of comprehensive video game player profiles: Player models, mental models, and behavior models. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 65(1):807–811, 2021.

[21] Mauro Conti and Pier Paolo Tricomi. Pvp: Profiling versus player! exploiting gaming data for player recognition. In Willy Susilo, Robert H. Deng, Fuchun Guo, Yannan Li, and Rolly Intan, editors, *Information Security*, pages 393–408, Cham, 2020. Springer International Publishing.

[22] Marçal Mora-Cantallops and Miguel Ángel Sicilia. Moba games: A literature review. *Entertainment Computing*, 26:128–138, 2018.

[23] Márton Gosztonyi. Who are the gamers? profiling adult gamers using machine learning approaches. *Telematics and Informatics Reports*, 11:100074, 2023.

[24] Lorenzo De Simone, Davide Gadia, Dario Maggiorini, and Laura Anna Ripamonti. Design of a recommender system for video games based on in-game player profiling and activities. In *CHItaly 2021: 14th Biannual Conference of the Italian SIGCHI Chapter*, CHItaly '21, New York, NY, USA, 2021. Association for Computing Machinery.

[25] Paul Bertens, Anna Guitart, Pei Pei Chen, and Africa Perianez. A machine-learning item recommendation system for video games. In *2018 IEEE Conference on Computational Intelligence and Games (CIG)*, pages 1–4, 2018.

[26] Alexander Dallmann, Johannes Kohlmann, Daniel Zoller, and Andreas Hotho. Sequential item recommendation in the moba game dota 2, 2022.

[27] Pu Yang, Brent E. Harrison, and D. Roberts. Identifying patterns in combat that are predictive of success in moba games. In *International Conference on Foundations of Digital Games*, 2014.

[28] Julian Dibbell. The life of the chinese gold farmer. `https://www.nytimes.com/2007/06/17/magazine/17lootfarmers-t.html`, 2007. Accessed: August 2023.

[29] Kaspersky. Good game, well played: an overview of gaming-related cyberthreats in 2022. `https://securelist.com/gaming-related-cyberthreats-2021-2022/107346/`, 2022. Accessed: August, 2023.

[30] Andrej Hadji-Vasilev. What is ddos in gaming? cyber attacks on gamers in 2023. `https://www.cloudwards.net/what-is-ddos-in-gaming/`, 2022. Accessed: August 2023.

[31] Enrica Loria and Annapaola Marconi. Player types and player behaviors: Analyzing correlations in an on-the-field gamified system. pages 531–538, 10 2018.

[32] Kevin Hutchinson. Player profiling in practice. `https://www.gamedeveloper.com/business/player-profiling-in-practice`, 2016. Accessed: August, 2023.

[33] Terlecki et al. Sex differences and similarities in video game experience, preferences, and self-efficacy: Implications for the gaming industry. https://link.springer.com/article/10.1007/s12144-010-9095-5, 2010.

Acknowledgments

I am very grateful to each and everyone that supported me during this master's degree and overall during my studies. Firstly, I would like to thank my supervisor, Prof. Mauro Conti, who has been always supportive and a great source of inspiration and knowledge. Then, my deep thanks go obviously to my great co-supervisors, Pier Paolo Tricomi and Lisa Facciolo, who guided me through thick and thin even if they were on the other side of the world.

I want to express my gratitude to all of my friends who supported me and listened to all of my complaints. A special thanks, goes to Alberto Zanon, Egle, Mauro, and the Asahi Dojo which has become an important part of my life.

My warmest thanks go to every participant who agreed to let me torture them, helped me share my experiment with their friends in order to find new victims, and lightened my long days in front of that huge TV screen. Especially, Marco, Matteo, Blerina and Giuseppe.

Last but not least, I would like to thank my mom, my dad and my cat who spent endless days and nights by my side, encouraging me during my studies. In addition, I thank my grandparents, uncles, aunts and cousins and also those who are not here anymore: Nonna Rosa, Nonno Paul, Nonna Haimanot, and my godfather Stefano.