FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Injustice and Balance in Pervasive Video Games

João Álvaro Cardoso Soares Ferreira

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: João Jacob

Second Supervisor: Zafeiris Kokkinogenis

July 22, 2021

Injustice and Balance in Pervasive Video Games

João Álvaro Cardoso Soares Ferreira

Mestrado Integrado em Engenharia Informática e Computação

July 22, 2021

Abstract

Being an art medium closely tied with the advancement of technology, some of the most interesting video games get made by expanding the technical limits of play in many new ways. Just like the jump from 2D graphics to 3D, how the internet shaped multiplayer games and inter-connectivity between players, and the way that now Virtual Reality and Augmented Reality are changing what is possible in a virtual world, another game genre has grown in popularity with the advent of new, exciting technology – pervasive games.

The genre of pervasive games encompasses games that merge the game's virtual world and the real world together by taking advantage of the player's location data and contextual information, using the new leaps in technology regarding mobile internet. The phenomenon of Pokémon GO spearheaded the genre's break into the mainstream, allowing many other similar games to thrive and carving for itself a really large audience worldwide.

A big issue with pervasive games is central to its claim-to-fame: due to being so context and location dependent, pervasive games are bound to have biases that make some contexts much better than others. A prime example of this is how players in rural areas tend to have very few game elements to interact with, while players in cities have a constant stream of new activities to engage in.

In this work, we tackled the issues described by building a data analytics platform dedicated solely to pervasive games, with the goal of helping game developers ensure that their games are balanced and just towards their players. This platform includes a variety of analysis methods that range from common graphs and statistics to innovative techniques that rely on cutting edge machine learning models and dynamic interactive maps, all of which were designed - unlike other available tools - uniquely with pervasive games in mind.

Furthermore, to facilitate the testing and showcase of this platform, systems that generate reliable artificial player profiles and a realistic simulation of their in-game behaviors were developed, allowing developers to change their games and test the impact that the new or modified features had on their player-base. This generated data is all based on each location's demographics, ensuring a close link between the simulated players and their real life counterparts.

This work is a step forward in not only providing open and universal tools for game developers, but also helping solve significant issues that hold pervasive games back from adventuring in more complex and deep experiences.

Keywords: Game Adaptivity, Pervasive Games, Data Analytics, Machine Learning, Game Design, Mobile Games, Data Platform

Category: Human-centered computing , Collaborative and social computing, Information systems , Information systems applications , Multimedia information systems, Contextual software domains

Resumo

Sendo um meio artistico intimamente relacionado com o avanço da tecnologia, alguns dos videojogos mais interessantes são feitos através da expansão dos limites técnicos do que é considerado um jogo. Tal como o salto de gráficos em 2D para 3D, o impacto da internet em jogos multiplayer e a inter-conectividade entre jogadores, ou a forma como a realidade virtual e a realidade aumentada têm mudado o que é possível dentro de um mundo virtual, há um outro género que tem aumentado em popularidade com a chegada de tecnologia nova - jogos pervasivos.

O género de jogos pervasivos engloba jogos que misturam o mundo virtual do jogo e o mundo real através dos dados contextuais e de localização do jogador, servindo-se dos novos avanços tecnológicos acerca de tecnologia móvel. O fenómeno de Pokémon GO liderou a quebra do género para o mainstream, permitindo a que muitos outros jogos semelhantes tenham sucesso e obtendo para si uma audiência mundial considerável.

Um grande problema presente nos jogos pervasivos é central ao aspeto que os torna únicos: devido a serem tão dependentes do contexto e localização do jogador, jogos pervasivos têm maior tendência para perconceitos que tornam alguns contextos superiores a outros. Um exemplo prevalente é jogadores em zonas rurais têm uma quantidade reduzida ou nula de elementos de jogo, enquanto jogadores em grandes cidades têm uma enchente constante de novas atividades.

Neste trabalho, procuramos lidar com os problemas descritos através da criação de uma plataforma de análise de dados dedicada unicamente a jogos pervasivos, com o objetivo de ajudar desenvolvedores de jogos a garantirem que os seus jogos são balançados e justos para com o seus jogadores. A plataforma inclui uma enorme variedade de metodos de análise, indo de gráficos e estatísticas comuns, passando por mapas dinâmicos e interativos, até técnicas inovadoras que se servem de tecnologia de ponta no ramo do Machine Learning, todos os quais foram desenhados - ao contrário de outras ferramentas disponíveis - unicamente com jogos pervasivos em mente.

De modo a facilitar o teste e demonstração desta plataforma, foram desenvolvidos sistemas capazes de gerar perfis de jogadores artificiais fidedignos e uma simulação realista, permitindo a que os desenvolvedores alterem os seus jogos e testem o impacto dos elementos novos ou alterados na sua base de jogadores. Este conjunto de dados gerados é todo baseado na demografia de cada região, garantindo um laço próximo entre os jogadores simulados e os seus equivalentes reais.

Este trabalho é um passo em frente não só em oferecer ferramentas abertas e universais a desenvolvedores, mas também em auxiliar na resolução de problemas significativos que têm impedido jgoos pervasivos de se aventurarem em experiências mais complexas e profundas.

Keywords: Game Adaptivity, Pervasive Games, Data Analytics, Machine Learning, Game Design, Mobile Games, Data Platform

Category: Human-centered computing , Collaborative and social computing, Information systems , Information systems applications , Multimedia information systems, Contextual software domains

Acknowledgements

My utmost gratitude goes out to my family, who were always incredibly supportive of me during my life and studies. I want to thank all of them, but in particular, I must thank my parents, grandparents Américo and Conceição, my uncle Luís and my aunt Rosa, for constantly offering their help during the writing of this dissertation. A special thanks to my parents, who endured months of hearing my keyboard during the frequent all-nighters and writing marathons. I thank them for their patience most of all.

I want to thank my colleagues João Lima, João Carlos Maduro, and Fernando, who are great friends and were crucial in their role during the development of the preliminary stages of this work.

I can't help but mention the rest of my friends who, although having no involvement in this work, are very important to me and helped keep my motivation high. My girlfriend Cezara, my best friend Matilde, and everyone else whose names I won't list for fear of forgetting someone. You know who you are; thank you for your friendship!

Lastly, I must show appreciation for the support offered by my supervisors, Zafeiris and João. Their role in the development of this work was invaluable.

João Álvaro Ferreira

"Start with what is right,
rather than with what is acceptable."

Franz Kafka

Contents

List of Figures

List of Tables

Abbreviations

AI	Artificial Intelligence
APM	Actions Per Minute
AR	Augmented Reality
ARG	Alternate Reality Game
GUI	Graphical User Interface
LARP	Live Action Role Playing [Game]
KPI	Key Performance Indicator
ML	Machine Learning
MAE	Mean Absolute Error
MSE	Mean Squared Error
RMSE	Root Mean Square Error
RL	Reinforcement Learning
SVM	Support Vector Machine
VR	Virtual Reality

Chapter 1

Introduction

Contents

1.1 Context

Modern, mobile pervasive games are a game genre very reliant on using the real world as a medium for the player to interact with its systems. Pervasive games have exploded in popularity recently due to the high technological requirements needed to produce an experience on par with other types of games. Some base requirements are widespread internet access and mobile phones powerful enough to be used as interfaces for games with quality graphical output from a client perspective. Significant camera and software requirements were also needed to use modern AR systems, a staple in many of these games. From a developer stand-point, the most significant barrier to creating these games is the quality of the infrastructure required and a framework to interface with it. Players and the real world they operate in need to be entirely in sync with the virtual world presented in the device they play the game with, so it is imperative that there are no connection issues and that the game's servers are capable of withstanding the load from a high player-base, which might spike during specific in-game events.

The "Pokémon GO" boom led to several other pervasive games getting released, shining a spotlight on the genre (although these were all much less popular). It would also be wrong to ignore pervasive games released before "Pokémon GO", such as Geocaching - which, although often more straightforward in its pervasive aspects, are still very relevant today.

While Niantic is the originator of this specific implementation of location-based pervasive games with "Ingress" and having produced others such as Harry Potter: Wizards Unite and Catan: World Explorers, they are not alone. Developers Ludia applied the same concept to the Jurassic

Park intellectual property, making "Jurassic World Alive". A common issue with these titles is the lack of balance in implementing the handling of geographic and context-specific user data.

Despite immense popularity, the "Niantic format" does not have a monopoly on modern, mobile pervasive games. Other pervasive games are also relevant, such as Minecraft Earth or Zombies, Run!. While not following the same format, the scope of this work also applies to them. A more in-depth exploration of pervasive games as a whole and their modern context is present in the Background section of the following chapter.

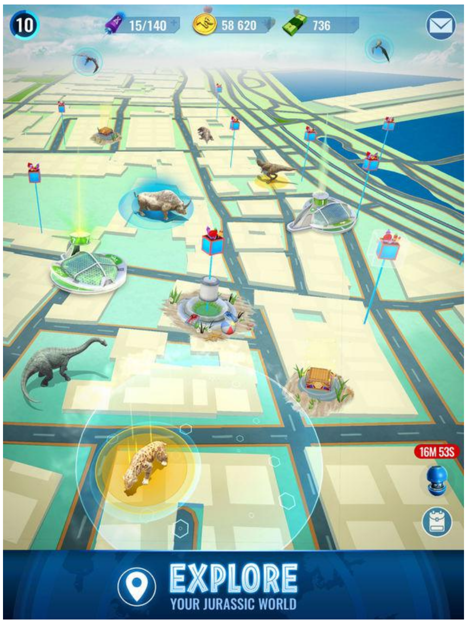

Figure 1.1: Jurassic World Alive promotional image

Part of the goal of this work is to aid in developing any games that include pervasive elements. After all, the genre's recent explosion has only produced relatively simple games centered around their concepts; there is a long history of experimentation with pervasive games and a vast universe of possibilities regarding integration of pervasive elements within every other game genre. This thesis aims to support any project that is developed with those characteristics.

1.2 Motivation

Pervasive games have risen in the public consciousness and are now a staple of mobile gaming. The genre as a whole and its flagship now have player bases in the millions and substantial revenues. Those are motivating factors towards the development of this project - helping fix some of the problems common in implementations of pervasive game elements would surely help improve the player experience, leading to better player retention and higher profits.

The pervasive flagship game, Pokémon Go, has broken revenue records in 2020 [20], despite struggles from the Coronavirus pandemic (which keeps players indoors and is an issue for this game genre for obvious reasons). From a player-base standpoint, Pokémon Go also reached an all-time peak of 28.5 million active daily players [29] in 2016, although this number leveled out to around 5 million soon after. Despite this, the game bounced back to around 11 million players by 2020, resulting in record profits. Pokémon Go is the focus whenever discussing Pervasive Games due to its dominance over the genre - Pokémon GO claimed an 84% market share of downloads of location-based games and a 92% share in terms of revenue in December 2018.

With how relevant Pokémon Go and some of its contemporaries have gotten, it is essential to analyze what kind of experience they offer to players and how that experience might be improved. Some of the most glaring flaws in modern pervasive games regard issues balancing the game around context-specific data, with a large portion of the player-base having a significantly sub-par experience.

The lack of balance pertains to how heavily the game promotes social interactions, using population density to generate many of its gameplay elements without considering how less populated areas might fare. Even without considering the usage of location data regarding population density, a lack of in-game connections, and real-life player groups for individual users to insert themselves in rends many of the game elements (such as raids) unusable.

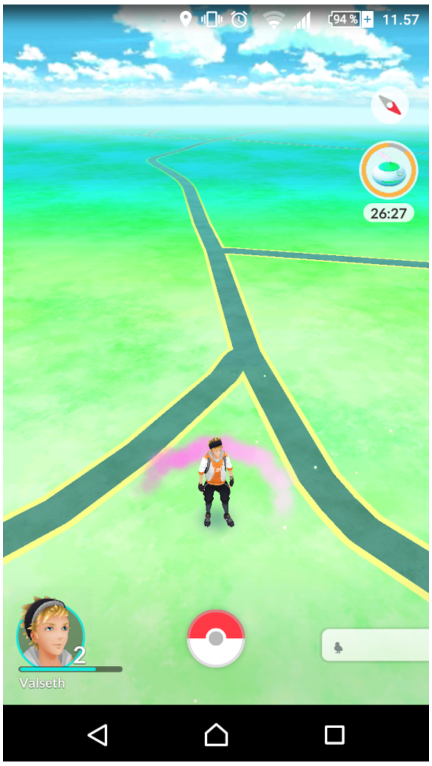

Figure 1.2: Pokémon Go in a rural area, with no spawned game elements

These factors lead to rural players being disenfranchised, effectively being unable to play the game to any significant extent, and making the game too easy for city players, providing an overwhelming amount of content and resources. Population density and the generation of content is not the only problem, however, as event and Pokémon spawns are also dictated by the biome in which the game is played. Players that live in the centers of large countries with very uniform geology will thus only be exposed to a tiny portion of the game content.

Furthermore, players are incentivized to travel to big cities to take advantage of in-game events, allowing city areas to further develop their "scene" for each game and even benefiting local businesses. This incentive is not only wrong from an injustice standpoint - catering to players that live in cities, forcing others to travel to be a part of a community, placing events within city centers bolstering their businesses only. It is also bad for variety, as players will end up in the same place for every big event (which, given the focus these games have on location, can lead to the events becoming repetitive and monotonous).

As a side note, this focus on location leads to players frequently playing while driving or in other dangerous contexts, which should be avoided as much as possible.

Figure 1.3: Pokémon Go in an urban area, with many Pokémon, PokéStops and a gym surrounding the player

This project's motivation is to provide a solution to the problems described in this section, as they affect such a large number of players. Fixing these issues would also certainly improve player attraction and retention, which is vital given the revenue at stake with games of this scale.

Despite the overwhelming popularity of some types of pervasive games, this work not only solves issues present in location-based mobile pervasive games but issues that might be common to any pervasive game.

1.3 Objectives

This dissertation's primary goal is to serve as a foundation for identifying and mitigating balancing issues and injustices within the scope of pervasive games. A data analytics platform will be developed to provide developers with statistics regarding their game's usage of contextual data and its impact on the game experience for their player-base. Addressing issues such as:

- Favouring players in specific areas too much
- Hurting players in some areas to the point of their game experience being negatively affected or made impossible
- Use of location data in a way that makes players unable to access a large portion of the in-game content
- Use of context data that leads to social injustice (directly and indirectly)
- Severe game imbalance that leads to an unpleasant meta-game
- Over-reliance on social aspects, making the game unplayable without friends
- Over-reliance on paid content for the meta-game, leading to a "pay-to-win" environment
- Promoting possibly criminal or predatory behavior (eg. playing while driving or

Of course, it is acknowledged that the goal of games is not necessarily to be optimized for perfect balance but to provide a satisfying and engaging experience to its players. More than just a balanced experience, a game is expected to provide exciting and varied experiences to its users, especially in pervasive games that rely so heavily on the player's context. Some imbalance is expected for a satisfying experience, which will be accounted for in the analytical process. This data analytics platform will use machine learning techniques to provide the developers with more profound conclusions regarding their game's data.

After providing developers with an analytical perspective, the platform also provides suggestions on what to do to fix the problems it finds. These suggestions can range from fundamental tips, obtained from frameworks such as GeoPGD [16], to specific concerns such as demographics that are disadvantaged. Topics covered by the suggestion can range from socio-economic factors to temporal or geographic ones. The goal is to use the information obtained from the world, be that from nature (location, weather) or human society, and balance the game's systems with it in mind.

1.4 Document Structure

This document begins with an abstract, providing an overview of the subjects it will cover, and an introduction that goes more into detail regarding the context for this work, the motivation for the work itself, and the objectives we aim to achieve. Following the introductory chapter, this document contains three more chapters, a list of references, and an appendix section.

The chapter after the introduction is the State of the Art. In this section, we explore the history of pervasive games and what the term means today, the most recent data analytics platforms for games, personality categorization for players, relevant machine learning techniques, specific data analytics that are used in pervasive games, how modern pervasive games handle their geographic and context data and related work.

Following the state of the art, the third chapter provides a strict problem definition and an outline of the intended planning for the data analytics platform and its development.

The fourth chapter entails the actual development of the project itself, with an in-depth description of the platform's structure and the implementation of its main components.

The fifth chapter is an analysis on the results obtained from the work developed, putting forth a discussion on the outcomes and what they mean for the project as a whole and for exploration of the issues that it tackles.

Finally, the sixth chapter is a conclusion, a culmination of all the work described in the previous chapters, and an overview of the work done, the contributions it provides and how it can be expanded upon in the future.

Chapter 2

State of the Art

Contents

In this chapter, there will be an exploration of subjects covered in this dissertation and of the techniques required for the development of the proposed solution. As such, this chapter can be split into two major sections. Section 2.1 provides background information on a lot of the elements and tools that will be at the center of this project, such as an in-depth look at pervasive games - the main theme of this dissertation. Section 2.2 is about the tools used to develop and manage these pervasive games that are relevant for the development of this work.

2.1 A Background on Pervasive Games

Pervasive games, a game genre that has grown in popularity in recent years, are a type of game that does not have one strict definition, covering a wide range of games and characteristics. The first definition for the term "Pervasive Games" came from "How to Host a Pervasive Game - Supporting Face-to-Face Interactions in Live-Action Role-Playing" [66], where they are described as games that are "augmented with computing and communication technology in a way that combines the physical and digital space together".

Although there are common elements to all games described as "pervasive", such as the game allowing the players to interface with it via actions in the real world, or using the player's location and time as a reference for the virtual environment, the simple definition provided by Schneider and Kortuem does not cover all games described as pervasive. According to Marcus Montola, in "Pervasive Games - Theory and Design", pervasive games "exist in the intersection of phenomena such as city culture, mobile technology, network communication, performing arts", and "some games use high-end technology, while others can be realized with no technology at all" [50] This definition ends up being the most accepted and used throughout works on pervasive games, and thus it will be the one used for this dissertation.

In this section, a comprehensive description of the history and definition of pervasive games will be outlined. Pervasive games aim to blur the line of the game world by using actions in the real world's space as a medium for inputs, blending the game context's abstraction into a real environment. There are pervasive games that mostly exist in a virtual world but then require the players to interface with the real world for in-game actions to take place or become available, and others that do not have any virtual elements at all and function based on an agreement by all players to follow the game rules during all of their actions in the real world space.

Due to the shifting definitions throughout the years and what developers had in mind with their gamification of the real world, the "pervasive" game genre is used as an umbrella term, covering games with non-standard input methods and even often games that do not require any technology, but that fall under the label of pervasive due to blending the imaginary game world with the real world. This broad scope has led to some research that has argued for the genre's re-definition, making the term more useful and descriptive of computer games with mixed reality elements [54]. These games can take advantage of spatial, temporal, social, or functional contexts to enhance the player experience. To fully understand the scope of games considered pervasive, it is essential to look at the history of the genre, live-action role-playing games, and the concept of the magic circle.

2.1.1 The Magic Circle

At the core of the magic circle concept is the question of what it truly means to enter a game's systems and begin play. The artificiality of the conflict within games, how they are separated from the life outside of that game, the boundaries between "game" and "non-game", these concepts are something that games often do not represent explicitly. Does a soccer game only begin when the referee blows the whistle or is the picking of the starting line-up, tactics, and formation part of the game as well? All participants accept a tackle in rugby during active play as an element of the game and not an aggression, but it is because everyone involved understands it to happen under the frame of the game and not in "the real world". Without this frame, the same action would be perceived as a violent assault. The conditions for a game being met, and the frame under which a spectator or player understands that a game has or has not started, are often only perceived passively and not outright stated. A core component of this frame is a feeling of safety due to the game's artificiality, as the game world is considered separate and understood under different contexts and rules.

Defined by Zimmerman and Salen in their book Rules of Play [65], the magic circle is a concept referring to how a player perceives a formal game's frame and the boundaries between the real world and the game world. The magic circle was defined, expanding upon the limits of play defined by Johan Huizinga's definition of play as "something outside ordinary life" and inspired by a passage where he describes every game as requiring a marked-off play area, with one of the examples being a magic circle.

This concept was developed further by Niewdorp in "The Pervasive Interface: Tracing the Magic Circle"[53], in which its relation to pervasive games are explored. Niewdorp clarifies that when it comes to pervasive games, the player must be able to interface with the game through a new layer of context that is applied to everyday reality and blends with it seamlessly during play. The edge of the magic circle is blurred, as the player does not stop living ordinary life while playing but simply adds a new meaning onto entities that he or she might encounter. According to Nieuwdorp:

> "We should not, however, view the magic circle as merely a rigid sphere that can be placed as an overlay on top of everyday reality. Instead, it can be seen as an almost organic entity that changes, develops, and interacts with its surroundings as the pervasive game comes into being for a player. In this sense, the magic circle becomes almost a permeable membrane through which conventional meaning, psychical artifacts and environments, and players alike can slide in and out of the game"

This perspective regarding pervasive games, which characterizes them more regarding their game design rather than their reliance on technology, was expanded upon by Montola in "Exploring the Edge of the Magic Circle: Defining Pervasive Games" [49]. In this work, Montola defines a pervasive game as "a game that has one or more salient features that expand the contractual magic circle of play socially, spatially or temporally.". As such, for a game to be pervasive, it must comply with at least one of the three mentioned forms of expansion of the magic circle:

- **Spatial Expansion** - The play area is undefined or unlimited. This is typically done by relying on GPS.
- **Temporal Expansion** - Play is not constrained to limited "play sessions", having a continuous "game session" that interlaces with everyday life.

- **Social Expansion** - A less common expansion in technology-driven pervasive games, the social expansion implies that people who have not entered the pervasive game's "contract" (that is, people who do not know the game is happening or do not think of themselves as active players) are also considered as game elements or players. The bounds of who is or is not a player or the roles a person might be taking become unclear.

Such as with many other game genres, the most exciting part of a pervasive game is what goes on inside the player's head. Like a chess player thinking of possible strategies, or a Poker player guessing cards and bluffs, a pervasive game player interprets the world differently than non-players.

During the time-span in which some of these works got released (2001-2005), we got some of the earliest examples of games that were designed with the "pervasive game" concept in mind. One of the first examples is "Pirates!" [27], a game where each player is considered the captain of a ship and uses a mobile computer connected to WLAN. In this way, they can explore a virtual environment by walking around with their computer in the physical world, doing missions, interacting with other players, and upgrading their ships.

Killer (or Assassin) [39] is a pervasive game designed before this time that is typically played on campuses by college students without any digital component. This game is a notable example as it has a social and temporal expansion of the magic circle. In Killer, players sign up anonymously and are given a target they have to "kill" according to the pre-agreed rules (e. g. hitting them with a nerf gun or a toy lightsaber). The game runs continuously until an established deadline, usually a week, and players must collect points by taking out targets and completing events (players who have been killed come back at set intervals of time, and new targets are given).

2.1.2 Live Action Role Playing Games

Live-action role-playing games, also known as LARPs, are role-playing games where the game space is the real world, and players act as their characters. LARPs originated in the 1970s and 1980s by table-top RPG enthusiasts. While not inherently pervasive themselves, the integration of technology in LARPs led to the inception of pervasive games and their first definition [66]. Integrating technology is not necessary for LARPs to be pervasive, however. Pervasive LARPs that expand the magic circle by including by-standers as a part of the setting, for example, in LARPs played in renaissance fairs where visitors are symbolic of peasants or royalty in the imaginary game world.

LARPs often have a vast scale, sometimes re-enacting old battles or being played in expansive areas to get a feeling of adventure typically associated with RPGs. It is common to use technology to track player locations, track player scores, calculate battle outcomes, and other features that are staples of RPGs but would be cumbersome and break immersion if done manually by the players. As described in Rules of Play: Game Design Fundamentals [28], LARPs have a very heavy social component, with participants dressing and acting as their characters and following an etiquette that fits the game's context and rules. Due to both of these factors, the introduction of pervasive elements in LARPs is a natural fit.

2.1.3 Alternate Reality Games

Alternate reality games, also known as ARGs, are a set of narrative-driven games that, similarly to pervasive games, blend the line between the real world and the game world. Aside from possibly also expanding the magic circle in a spatial, temporal, and social sense, ARGs typically expand it from a narrative sense - events that occur in-game are depicted as ordinary, real-life occurrences, and part of playing the game is recognizing that the event was even a part of the experience. These characteristics are most easily depicted by two common phrases associated with ARGs - "The Curtain" (referring to the division between the players and the game management) and "This Is Not A Game" - both relating to the need to treat the game and its narrative as if it is entirely real. The latter sentence is frequently seen in more mysterious ARGs that entice players with significant threats or rewards.

A common structure for ARGs is a treasure hunt, where a "puppet-master" drops hints to players that they must follow to locate further hints in real life and proceed in the game. A famous form of treasure hunt is "geo-caching" [24], in which players use GPS-enabled devices to locate boxes with hints and rewards. The findings can be accumulated and left in different "caches" according to players' wills, which are all logged on a global database. The competitive aspect and the stories help bring a social aspect to these kinds of games. Geo-caching, and many other ARGs, are considered pervasive games.

While not ARGs themselves, a very similar but more self-contained type of experience is the Escape Room. Escape rooms commonly have pervasive elements in their integration of technology [57] with puzzles. While typically, the only expansion of the magic circle that occurs in escape rooms is through technology that forces players to think of objects differently than they usually would, some escape rooms can be considered pervasive games if they integrate pervasive elements.

2.1.4 Ubiquitous Computing Games.

In the 1990s, Mark Wiser wrote a paper[75] describing his belief that hardware and software elements would become ubiquitous and integrated into normal, every day objects to the point where we would forget they were even there. Given the paradigm we live in with smartphones' omnipresence, his belief was proved to be correct. Despite the 90s being far too early for consumer products with ubiquitous computing to be relevant, the idea was adopted by institutions as a goal to strive for.

The Disappearing Computer Initiative [9] was started in 2001 by the EU, an initiative that funded research into "ubiquitous computing". This initiative funded sixteen projects that followed ubiquitous computing pioneer Norbert A. Streitz's philosophy of approaching this issue with real-world affordances in mind first [69].

Soon there was a push to use of this technology to design pervasive games, by then still in its conceptual infancy. These "ubiquitous computing games" were designed with the future in mind, imagining what kids ten years later would be playing with after school.[19]. Commercial ubiquitous computing games were released in this era to some enthusiasm, as new technology facilitated new forms of play in ways that it had not in previous attempts (for example, with the virtual reality push of the 90s).

One such example is "Boktai: The Sun Is in Your Hand" [43], a Game Boy Advance game that came with a light sensor. This game featured a main character with a light gun that got recharged whenever the player was in the real world's sunlight. The interactions with the outside world and the relevancy of the time of day leads to new strategies that come up during night time (players have to find in-game charging stations if they cannot use the sun). As such, Boktai can be retroactively considered one of the first commercial pervasive video games. Due to its pervasive elements, it was seen as an experimental gimmick by Nintendo. When asked of doubts about the game's feasability, Director Hideo Kojima said "There was the matter of the sensor itself, and also being able to furnish enough sensor parts for cart production. But the cost was a biggie too. It was hard to see it working as a commercial product."[55]. The game was not a huge success, but it is considered a cult classic to this day.

Ubiquitous computing games are spiritual predecessors to both augmented reality games and location-based mobile pervasive games from a technology standpoint.

2.1.5 Location-Based Mobile Pervasive Games

Location-based mobile pervasive games are the pervasive games that currently get the most focus due to the unprecedented success of Niantic Games' Pokémon Go. However, this sub-genre of pervasive games has a long history, as using GPS data to manage and interact with players has been at the center of many pervasive games. One of the pioneer commercial pervasive games, released in 2001, is Botfighters[47] by Swedish developer "It's Alive!". The basic concept of Botfighters is quite simple: players earn credits and a high score by locating and destroying other players that get represented in-game as bots. Using these credits, a player can build and upgrade their robot (done at a webpage). The game is played via text message, receiving the address of new targets and exchanging messages depending on the actions they want to take. Botfighters was a pioneer in the pervasive genre, as with meager technology requirements, it managed to create a very innovative pervasive experience.

Another relevant example is Mythical: The Mobile Awakening [35], a 2008 research project by Nokia. Mythical's gameplay loop consists of performing collaborative rituals to progress in the game and then use the skills obtained to find magical beings and fight other players. This game used many contextual elements, such as the moon phase, the weather, the time of day, the sun's location, nearby Bluetooth-enabled devices, and other elements to affect gameplay. The part of the gameplay loop where players would find new creatures and other players to fight relies heavily on location data and the overall experience depends entirely on contextual data and other players for content generation.

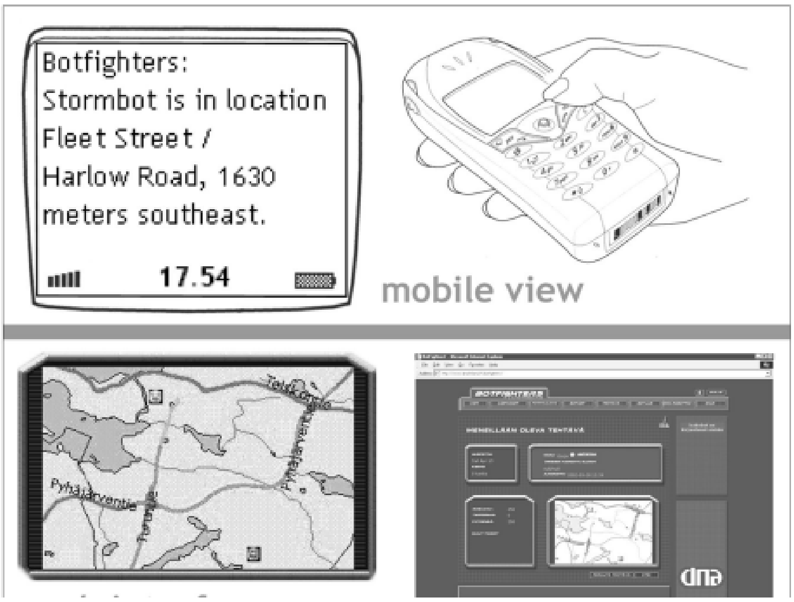

Figure 2.1: Botfighters gameplay - Credit Olli Sotamaa / It's Alive Mobile Games AB!

A more recent game that focuses solely on the location aspect is "Zombies, Run!"[68] by Six to Start, a narrative-heavy game that puts the player at the center of a zombie apocalypse. As the game's goal is to be a jogging companion, it uses location data and audio (players must use headphones, they listen to the narrative while running) to design both a balanced game and a well-paced running experience with highs and lows to match the narrative. The game encourages the player to go on further runs by having more missions, allowing them to progress in the narrative, and gameplay incentives that allow the player to improve their base (which might also get attacked by zombies).

Finally, there is a wealth of literature and research on Pokémon Go's success [20] (the most successful game within the "Niantic model"). Pokémon Go puts the player at the center of an adventure that allows them to meet and catch hundreds of new creatures called Pokémon, raise them and pit them against the Pokémon of other players in battle. There are numerous in-game activities such as raids (events where many players come together to fight a stronger version of a Pokémon for great rewards) and gyms (locations where players can leave Pokémon to collect rewards, at least until a new player arrives and defeats them, taking their place). In Pokémon Go, the magic circle is expanded spatially and temporally, as the game can be played any time and anywhere. The game has a heavy social component, and players not only depend on each other for raids and gyms but affect each other directly by placing lures that attract more Pokémon. Non-players only affect play very indirectly, with their phones' radio signal changing spawn rates slightly (more people detected in one location will lead to more Pokémon spawns), so we cannot claim that it expands the magic circle socially. Data obtained shows that Pokémon Go depends a lot on outdoor activity, ease of use, challenge, and nostalgia for player engagement and that these same factors, along with socializing, are key for players to want to spend money on the game. [32]

2.1.6 Augmented Reality Games

Augmented reality is a field that has been growing steadily throughout recent years and is now more accessible than ever by being easily integrated into most modern smartphones[18]. Augmented Reality and Virtual Reality have been many wished-for technologies that fundamentally change how we both see and interact with the world. Unlike VR, however, that creates a complete virtual world of its own, AR uses contextual clues to enhance the real world. Due to this, it is a perfect match for most mobile pervasive games [31], allowing users to see and interact with pervasive elements that they could not before. The AR elements in Pokémon Go, while not an essential part of gameplay, bring much of the charm and immersion that give the game widespread appeal.

That said, not all AR games are pervasive. While it expands the traditional bounds of how to interact with a video game, a game like "Invizimals" still operates under the magic circle's normal bounds, using the trading cards as a small extension of its play area. That is insufficient to consider a game like "Invizimals" pervasive, given that the trading cards are an element exclusive to the game and not a part of real-life that the players would interact with outside of playing the game.

2.1.7 Related Work

As mentioned previously, there is no unified platform that provides the solutions aimed for in this project. That said, there are projects have either similar goals or methodology that must be mentioned, as elements from their implementation are relevant for the development of this dissertation's proposed solution.

The first related project covers the use of learning analytics for location-based serious games[58]. In this paper, the authors added support for standards for location-based serious games into the xAPI [64] data analytics platform. The Experience API (xAPI), also known as Project Tin Can, facilitates the collection of data on user experiences with a background application that sends statements on user actions to a centralized database. The project in question then added two profiles to this specification, one for serious games and another for pervasive games. It then takes the data collected and analyzes it, obtaining assessments and visualizations that infer conclusions on the game experience.

A platform for analyzing pervasive games was developed that instead of optimizing for game experience, has safety in mind[59]. In this work, Nunes uses context data obtained while playing a pervasive game and, cross-referencing with third party information, calculates threats and dangers that might occur while playing a game. Some of these threats include: inattention to dangerous surroundings, personal physical safety, weather conditions, unexpected billings, restrictive locations and timed challenges. Nunes' platform then proposes ways that warnings for these dangers could be implemented, such as parental controls, and measures the impact it has on player experience so that developers can change their games accordingly.

Finally, a relevant work is the one of Johannes Henriksson[34], which investigated validation of game design choices with data analytics in location-based pervasive games. While not a platform, this project thesis adequately combined game analytics with pervasive game design. The player data was inspected for specific routes for each player and classified as different kinds of behavior, with notes being taken on what game mechanics led to the most engagement and interest.

2.1.8 Summary

This section aimed to define pervasive games, cover its history and obtain an overview of what game genres are related to them or contain pervasive elements. To do this, the definition of the magic circle was explored along with multiple conflicting definitions of pervasive games. The theoretical aspects behind the genre were initially defined by Schneider [66] in relation to LARP games, but then got expanded by extensive research in the following years by Montola [49] [50], Zimmerman and Salen [65], Falk [28] and many others, dictated in part by the technological advances of the time.

A few common trends can be concluded from this section. A clear conclusion is how reliant on technology the genre is and advances within it are, with many pervasive games also being experimental uses of new technology or research. As is stated in "Issues in Development of Location-Based Games"[40] by Jacob, this reliance is both a positive and a big limitation, as the games developed are designed to focus too much around this new technology and its newness means often it has not been polished enough for a quality experience. In fact, it can be inferred that pervasive elements have been very hard to integrate well within successful, large scale commercial pervasive games, and that the bigger successes come from recent implementations within very simple, location-based games. Deep implementations of pervasive elements have been reserved for more niche titles like Boktai [43]. An argument can be made that games that relied on a real-time clock, such as 2001's Animal Crossing[26], expand the magic circle temporally. This implementation is important to the game but is very light, reinforcing the idea that while pervasive elements can help a game be successful, it is hard to develop a complex experience centered around pervasive elements. As such, while Animal Crossing has pervasive elements, it is not considered to be a pervasive game due to how the core gameplay loop is not based on "pervasiveness".

Lastly, this section explored some of the elements that make location-based pervasive games fun and what measures to use to predict player enjoyment and instances of money spent [32], along with evaluating the importance of AR and its integration with location-based pervasive games to the genre.

2.2 Existing Technologies

This section covers the research done relating to the relevant technologies that might be useful to the project's development. The following subsections will be about technologies not only related to the topic of pervasive games, but to game analysis as a whole and for the specific techniques and concerns that must be taken into account (such as analysing the concept of fairness in machine learning).

The technologies, frameworks and other utilities chosen in this chapter were picked based on their predicted utility and importance in the project's implementation. As one of the issues that has been identified is the lack of good information and guidelines regarding pervasive games and their development, some of the following sections will cover the search for frameworks and tools. Some research into analysis techniques is also required, be them game-specific or more advanced concepts in the realm of machine learning and player profiling. All in all, while the cohesive thread that ran through the last section was the history of pervasive games and its adjacent genres, the thread that unites the following sections is the research of technical elements that are critical to the development of this work.

2.2.1 Data Analytics for Games

Player data analytics are a potent apparatus for researching play, game cultures, experiences, and past and present practices. Egliston[25] divides approaches to data analytics with large amounts of data into two methods - the use of analytics-based paratexts (objects surrounding the video game) directly leveraging the game's API to answer specific questions. The first is the use of analytics-based paratexts, objects surrounding the videogame proper, as a form of ad hoc research device. The second is the practice of directly leveraging the game's API and how this can be used to create custom data-tracking applications. While the first practice allows us to obtain general trends in players - what strategies or entities they search for the most, the more popular ways and locations to play the games - the second allows us to obtain usage statistics for the game itself and answer more specific questions. It is essential to use both approaches and ensure they're not wildly disparate. For example, if players have shown much interest in playing with a specific character, making that character too hard to obtain might make players frustrated and lead to a sub-optimal experience. Cross-referencing data and exploring the datasets thoroughly is vital to understand what issues lie with a game and how to allocate resources better.

One of the most important aspects of doing effective data analytics is to focus on asking the right questions and articulating a goal clearly[30], rather than just using complex analysis without concrete questions to answer. The "right questions" are often centered around what's referred to as a Key Performance Indicator (KPI). Some common KPIs are:

- **Cost of User Acquisition:** The average cost to obtain an install for a single player
- **Lifetime Value:** The average revenue for a single player over their lifetime. This value can be empirical (looking at historical data over a specific interval) or predicted, which extrapolates from the empirical with machine learning techniques.
- **Conversion Rate:** The percentage of players who've done something who do something else ("sign up" to "install", "install" to "in-app purchase", among others). This measure can also be done with thresholds.
- **Churn Rate:** The percentage of players who've left the game a specific time after install.

- **Active Users:** Number of active players over a specific range of time.

- **Playtime:** The number of hours played.

However, appropriate techniques are still important, which is why the following section will describe some machine learning techniques relevant for the development of this project.

2.2.2 Machine Learning Techniques

In developing a data analytics platform, employing effective machine learning techniques is imperative. When it comes to reinforcement learning uses in video games, it is more commonly used to enhance AI enemies or bots' performance. One of the most prominent reinforcement learning approaches is Stable Baselines3[63], which makes use of the Proximal Policy Optimization[67] algorithm.

Multi-output regression is a relatively recent machine learning technique with much promise in determining optimal game characteristics (explored in the preliminary work 3.2). There are two main approaches to multi-output regression [48], which include: building a single-target output regression model for each of the outputs and building a multi-target output that takes into account targets correlation and deliveries the multiple output variables needed[74]. The topic of multi-output regression is still recent in terms of machine learning, and the application in video games is still open to results and experiments.

2.2.3 Fairness in Machine Learning

Along with ensuring that the most appropriate and efficient techniques are used. A strict definition of fairness is a subject of much contention among researchers, with a large number of conflicting definitions being commonly used. There are organizations dedicated to finding answers regarding fairness and establishing solid parameters, such as the Fairness, Accountability, and Transparency in Machine Learning group (FATML) [2].

Taking into account prominent research [73], it is possible to classify some of those definitions according to the following statistical measures, taking into account the results for different groups where the variables are independent from the group's characteristics:

- Definitions based on predicted outcome - group fairness or statistical parity

- Definitions based on predicted and actual outcome - predictive parity (outcome test), false positive error balance, false negative error balance, equalized odds, overall accuracy equality, among others

- Definitions based on predicted probabilities and actual outcome - different from the last measure, as this focuses on calibration, test-fairness, and balancing for positive or negative classes

- Similarity-based measures - causal discrimination or fairness through awareness/unawareness
- Causal Reasoning - definitions based on causal reasoning assume a given causal graph, which is used for building fair classifiers (examples include counterfactual fairness, fair inference and resolving blatant discrimination within the graph).

With this variety of definitions, it is understandable that algorithmic decision making is increasingly controversial due to lack of fairness and accountability. There are many ways to measure fairness, and as such it is important that the measure chosen fits the context of its use. It is impossible to balance for all measures, as guaranteeing group fairness and false positive balance (for example) is not feasible in most situations[33]. As such, each context must be carefully analyzed for which type of fairness to strive for.

Even within the same context, different actors might desire different types of fairness to be optimized for by a machine learning algorithm. For example, an honest factory manager might be more concerned about the overall accuracy of a ML sorting app, while a regulator may be more concerned about bad products being approved rather than good products being rejected.

A notable example of lack of fairness within a machine learning algorithm having severe consequences is COMPAS, a recidivism algorithm that calculated how likely to become recidivist a criminal defendant was. This algorithm was used to determine probation lengths and parole values, so its accuracy is paramount. COMPAS was found to be biased, having overall good predictive parity, but having terrible group fairness and statistical parity, leading to racial bias in its results [45].

2.2.4 Player profiling

Players display different traits and preferences while playing games, and when it comes to pervasive games, it is imperative to denote that players from different regions might have radically different needs and wants. One of the steps in balancing a game and improving enjoyment is identifying the player profile. Although people are very complex, there are emotional responses that can be correlated with game satisfaction metrics[51].

Through their actions in-game, profiling player behaviors can facilitate personalization, adapting gameplay for different types of personalities to enhance enjoyment and reduce churn rate. Using behavlets [21], data-features that encode short activity sequences ('atoms' of play) which represent an aspect of playing style or player personality traits, we can predict what would lead to different reactions by different types of players within each game we analyze.

The suggestions obtained regarding player satisfaction must be implemented within the parameters of pervasive game guidelines, analyzed in a following section. A standard way of obtaining a player's satisfaction towards a game is the Game Experience Questionnaire [38], which is used in the preliminary work 3.2.

2.2.5 Pervasive Game Data Sources and Tools

Due to the popularity of Pokémon Go, Niantic, and the accessible API and GPS data, there is a large number of tools for Niantic games, mostly Pokémon Go but also some in Wizards Unite. The tools allow users to obtain information about predicted spawns, swarms, nests, shiny rates, among other things. These tools are either collaborative, reliant on players reporting spawns local to their areas (such as pokemap.net), while others use GPS and the Pokémon Go API to collect data (such as pogomap.info).

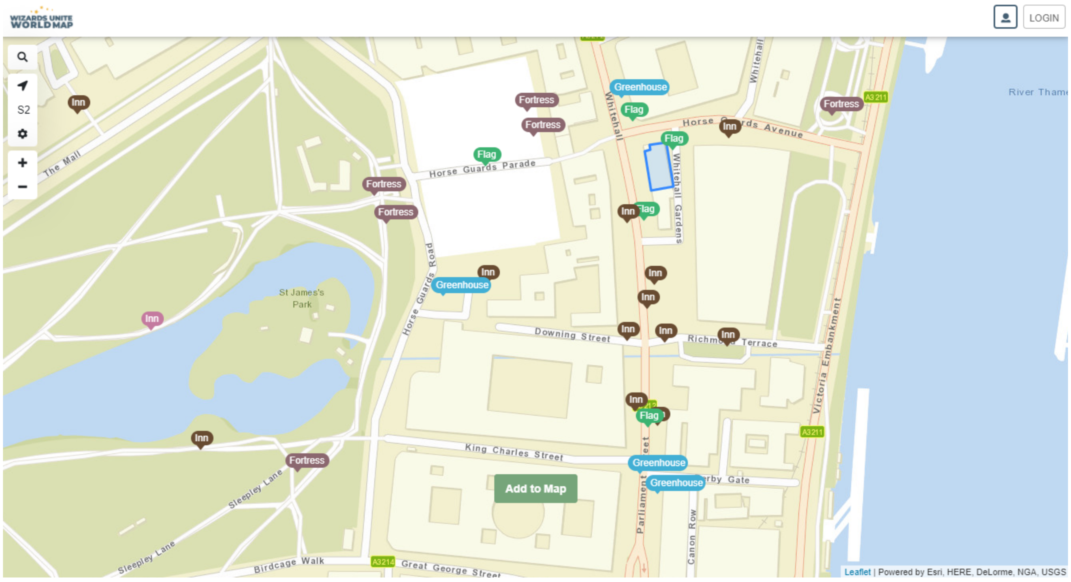

Figure 2.2: A screenshot from Wizards Unite World, detailing locations of inns, greenhouses, fortresses and flags

Harry Potter: Wizards Unite has an official Niantic community map (at harrypotterwizardsunite.com/map) [52], but there are still fan-made ones, such as Wizards Unite World (wizardsuniteworld.com/) [76]. Sadly, due to either popularity or how the APIs are structured, not many tools for other pervasive games are available. The largest and most prominent source for data is Silph Road [11], a community based on studying Pokémon Go and reporting their findings.

An essential piece of data for this project is what kinds of Pokémon spawn in which biomes, which, cross-referenced with the other map data, should predict what kinds of Pokémon will spawn where. Table 2.1, listing the common and uncommon spawns in specific biomes, was obtained from Silph Road.

2.2.6 Handling of geographic data

To the best of the author's knowledge, there is no unified framework on how to manage geographic data for pervasive games. Developers of each location-based mobile pervasive game develop their proprietary system that uses GPS data and other map information about objects and terrain to manage content generation. These systems are unique to each application and are all kept private,

Major Biome Name	Variant Biome Name	Alternative Names	Common and Uncommon Spawns
Bug Biome		Bush Biome	Spinarak, Eevee, Venonat, Ledyba, Exeggcute
Desert Biome	Fire Biome	Desert-2/Fire Biome	Ekans, Sandshrew, Geodude, Ponyta
Desert Biome	Arid Biome	Meowth/Desert-1 Biome	Ekans, Snubbull, Growlithe, Mankey, Meowth
Electric Biome		Commercial/Coast Biome	Magnemite, Voltorb
Grass Biome	Standard Grass Biome		Paras, Oddish, Bellsprout, Chikorita, Sunkern
Grass Biome	Grass+ Biome		Paras, Oddish, Bellsprout, Chikorita, Sunkern, Bulbasaur, Pinsir, Tangela, Dodrio
Mountain Biome		Clefairy Biome	Clefairy, Nidoran
Neutral Biome		Residential/Rural/Base Biome	Aipom, Murkrow, Natu
Oceanic Biome	Sea/Ocean Biome		Shellder, Seel
Oceanic Biome	Ocean+/Ice Biome	Psy/Swinub/Drowzee Biome	Shellder, Seel, Drowzee, Swinub, Jynx
Swamp Biome	Water2 Biome	Forest/River/Swamp/Canal Biome	Poliwag, Staryu, Horsea, Goldeen, Krabby, Marill, Wooper
Swamp Biome	Water2+ Biome	Totodile Biome	Poliwag, Staryu, Horsea, Goldeen, Krabby, Marill, Wooper, Totodile
Water Biome	Water1-slowpoke Biome	Magikarp/Slowpoke Biome	Magikarp, Psyduck, Slowpoke
Water Biome	Water1-tentacool Biome	Magikarp/Tentacool Biome	Magikarp, Psyduck, Tentacool

Table 2.1: Biome spawn distribution in Pokémon Go

so all of the information obtained from them is from the results players see in-game and from API mining, as described in the previous section. Each game developing its content-generation manager without any guidelines can often lead to problems due to biases within the system. In the specific case of Ingress and other Niantic games, they rely on the Historic Marker Database[10] for the USA, for which it bought the rights in 2011. The games do not rely solely on this database, using crowd-sourced suggestions for new markers [5] and allowing businesses to use Pokémon Go gyms stops to attract attention[8]. Due to Ingress having a player-base made up of mostly white young men, these crowd-sourced suggestions led to a higher rate of markers in predominantly white neighborhoods. Ingress's database was then re-used for all of Niantic's next games, leading to its biases being propagated. Its data generation also takes into account population density and regional socio-economic indicators into account[37].

Pokémon Go's success resulted in a ton of data being generated, which put their system to the test and showed the negative consequences of those biases. One of the first and more obvious ones was the rampant location spoofing that occurred [78], as many players could not engage with the game meaningful due to being in a location with a small or null amount of content generated.

This issue is at the core of the project, and the impact from this will be described in detail in the problem definition section of the next chapter.

2.2.7 Pervasive Game Guidelines

Pervasive games have evolved a lot since their inception and definition by Schneider in 2001 [66], and naturally, what is expected of an engaging pervasive game has also changed drastically. The GeoPGD methodology [16] is optimized for pervasive game development and has been designed with the medium's flaws and strong points in mind.

GeoPGD is focused on designing a pervasive narrative present in player's minds even when not playing, the game world and its evolution as it is integrated with reality, the definition of the game rules, and the pervasive dynamics that then expand upon the narrative. Unlike in traditional games, players expect pervasive games to be dynamic and change with the world around them, and the game's narrative and rules must fit this. The development process is cyclical, as the ideas behind these elements coalesce into a finished cohesive product.

Figure 2.3: The GeoPGD development cycle

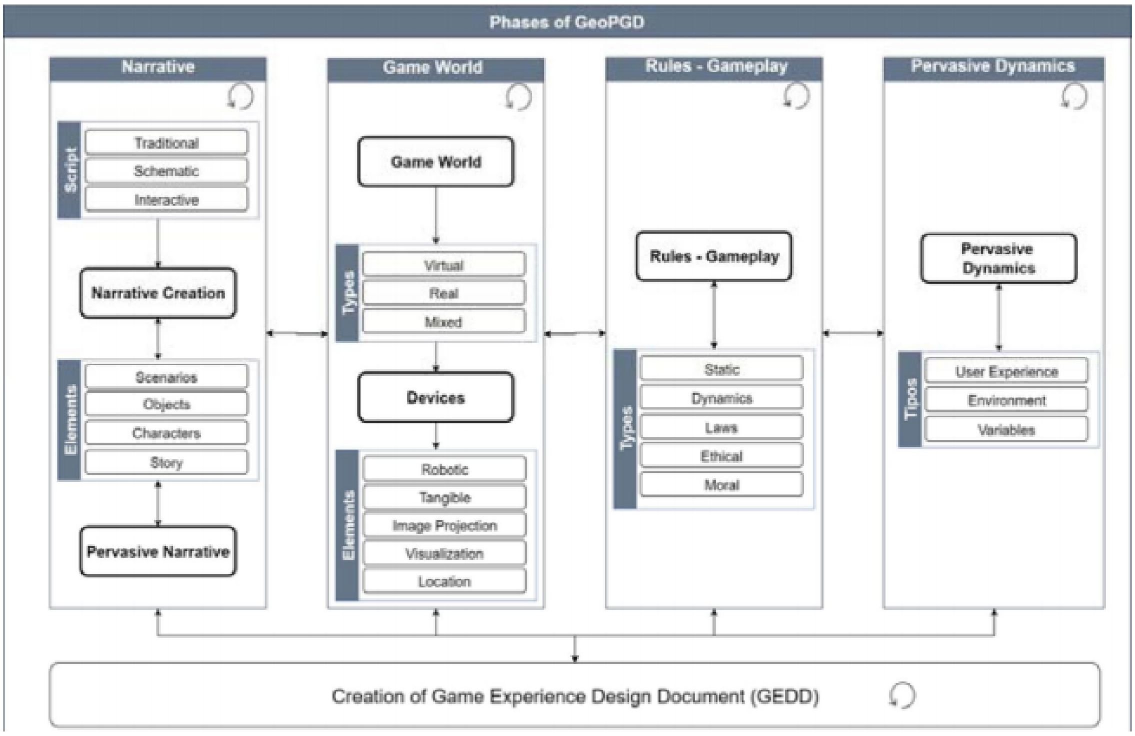

Figure 2.4: Detailing the specific phases of development in GeoPGD

There is a large body of work around how to improve game flow in traditional games, and while some of those principles also apply to pervasive games, these have their model[41] to proposed by Kalle Jeggers. This model can be summarized with the following guidelines:

- Concentration - The game should allow the player to easily switch concentration between in-game tasks and surrounding factors of importance, along with providing appropriate amounts of stimuli from different sources

- Challenge - The game should support the players in their own creation of game scenarios and pacing while helping players keep a balance of development in the game world.

- Player Skills - The game should be flexible and enable the players' skills to be developed at a pace set by themselves. Learning the game should be fun, and a player should be rewarded for their skill.

- Control - The game should enable the players to easily start play in a constantly ongoing game and quickly understand the current status of the game world

- Clear goals - The game should support the players in forming their own goals and providing new ones

- Feedback - players should receive appropriate feedback on their actions and on progress towards goals

- Immersion - The game should support a seamless transition between the game world and real-world, not require player actions that break social norms and allow for the attention to be shifted between the real and virtual aspects of The game without losing immersion or too much information
- Social Interaction - the game should support social interaction within the gaming system, integrating structures that motivate communication and social activity

2.2.8 Summary

This section described many of the principal research fields cutting edge technological knowledge required for the development of the project proposal at the core of this dissertation. The concrete programming languages and more basic frameworks weren't covered as they don't require deep investigation and might change with the experimentation done during the initial segments of the development. The next chapter, the project proposal itself, has a section detailing which technology was planned for use.

In this section, we went over relevant data analytics methods [25] for games and outlined the more relevant aspects of the development of a data analytics platform [30], listing some KPIs. We then mentioned some relevant machine learning techniques and algorithms that seem suitable for the project and are tested in preliminary work 3.2, namely the Stable Baselines [63] approach for Reinforcement Learning and multi-output regression [48]. The following subsection covered how to determine fairness in machine learning,describing some of the definitions[73] and the need for individualizing definitions to the project at hand. The best ways to enact player profiling were also detailed, covering satisfaction metrics [51] and the behavlets method [21].

After those sections describing general machine learning and analytics technologies, the following subsections covered relevant tools for pervasive games. Namely, these paragraphs went over game specific resources such as Silph Road, a description of the methods games used to generate and handle in-game geographic data [5] for these games and some guidelines for Pervasive Game development such as GeoPGD [16] or the model proposed by Jeggers [41]. Ideally, tools allowing an analysis of player data and behavior would be available, but given data protection laws it is understandable how they are not.

Chapter 3

Data Analytics Platform for Balancing Location-Based Pervasive Games

Contents

This chapter can be split into four sections that detail the problem at hand, previous relevant work done by the author, the proposed solution, and an overview. The first section delivers more information about the issue at the core of this dissertation, analyzing its causes and significant consequences. Section 3.2 describes a preliminary project developed in the lead-up to this dissertation that covers relevant topics to it. The third section describes the proposed solution and how it will be structured. This third section also includes a description of the system requirements, the system's architecture, and the technologies employed. The fourth and final section is comprised of a summary of this chapter.

3.1 Problem Description

As was mentioned previously, popular mobile location-based pervasive games often have issues with balancing their content generation. Given how central the handling of contextual and location data is to this kind of game, it is vital that balancing is good to ensure a game's health and longevity. It is inevitable, then, that since most of these games use in-house solutions and do not take advantage of any external framework to ensure their quality, problems will arise.

Balancing issues were noticed very early on in the lifespan of Pokémon Go. Released in the Summer of 2016 and reaching a peak of 28.5 million daily active players by mid-July in the United States of America alone, its player base's scale led to noticeable bias being apparent. As mentioned previously, Pokémon Go's marker database is based on Ingress's, and thus it kept some biases (in this case, in favor of a specific race), which immediately garnered attention [44] [37].

The issues with balancing were awful for the game as due to its simplicity and over-reliance on (then rudimentary) generated content, players with no spawns or markers nearby were left without anything to do in-game. Using population density for spawn rates without a minimum limit led to an urban/rural divide, where urban areas were swarmed with new spawns constantly, and rural players [72] were unable to play. Even for the players who were not affected by this, some of the biomes that define Pokémon habitats tended to have very low population density - meaning that Pokémon exclusive to those habitats ended up being much rarer than expected.

Other balance issues, related to exclusive events and Pokémon availability [71], also arose. A game that thrives on its social aspects turned its community against itself and drove massive amounts of players off, losing nearly 80% [46] of its player base just two months after its peak.

It is essential to have a closer look into these issues of social injustice. A study about Pokémon Go's content generation in urban areas in Florida and Boston [42] reinforced the racial bias findings, with black and Hispanic neighborhoods being disadvantaged when it comes to crowdsourced data coverage. Other than that, it is also apparent that PokéStops occur more frequently in areas with a lot of economic activity (commerce, tourism, recreation), and that PokéStops tend to cluster around gyms. A study product of the collaboration between five countries [22] gave us information on some of the distribution issues within the areas that do get content, indicating that Pokémon Go's content generation system favors areas with a high average socioeconomic status. The game advantages urban areas and neighborhoods with smaller minority populations, favoring neighborhoods with a majority non-Hispanic white populations.

Furthermore, this study shows that the surge in activity from Pokémon Go may have instigated a shift in human mobility patterns. Another study contends that Pokémon Go, due to its pervasive elements, can shape ethnographic encounters and fieldwork research [77]. Both of these factors combined with the economic boost that Pokémon Go players and events bring to any host location, and it is clear that the consequences for imbalance go far beyond a sub-optimal game experience.

3.2 Preliminary Work - Exploring Multi-Output Regression and Reinforcement Learning for Game Adaptivity

With this dissertation in mind, a project was developed exploring how machine learning techniques can change a game to improve balance and player satisfaction. The following subsections detail research into adaptive games, using reinforcement learning and multi-output regression. The game used for this experience was Breakout. The research developed produced promising results for the multi-output regression, along with showing reinforcement learning's effectiveness.

3.2.1 Methodology

This experience was developed using Unity and Python, with the connection between them being done using MLAgents. In Unity, a prototype of the game Breakout was developed that allowed for the system to change some of its parameters between each game round. Changing those parameters was how game adaptivity was achieved, by altering the ball's speed and size, the paddle's speed and size, and that the bricks were placed at.

Ideally, an experiment that relies so much on the human feedback aspect would get data from real players. This was impossible due to time constraints, so a player simulation model was developed in its place. Each player is generated based on six different generic personality templates, which dictate the heuristic the player will use for decision-making in-game, some of their physical attributes such as actions-per-minute (APM) or reaction time (with these values being randomized within a range) and their satisfaction, which is based on an implementation of the Game Experience Questionnaire [38] and whose parameters vary per personality. Satisfaction values could go from 1 (worst) to 20 (best). The reinforcement learning algorithm used was Stable Baselines3[63], and the multi-output regression models were the ones described in this dissertation's state of the art.

The user can determine how many rounds they want for the system to be trained with. Once that is done, and the system is running, each round works the following way: a player is generated from a personality template, and the system evaluates its characteristics, deciding from the data collected in previous rounds what parameters would suit a player with those traits best. For example, a competitive player would prefer a faster, more challenging game, while a newbie would expect the game to give them some leniency. Once the round is over, all of the game data is collected and saved, and a new round starts.

After all rounds have been finished and the training is over, a testing session begins for reinforcement learning with the complete model, following the same rules as before. Once that is finished, multi-output regression is ran over the testing data to predict the in-game parameters of each satisfaction. It is important to note that only data with a satisfaction of over 70% is used to train the multi-output regression model, ensuring it only learns from the rounds where RL's predictions were correct.

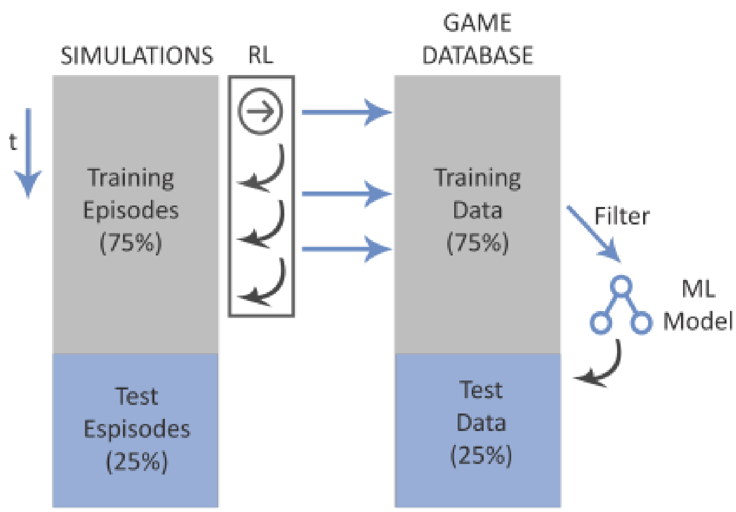

Figure 3.1: The structure of the rounds and data pipeline

3.2.2 Results and Conclusions

This experience's results were very positive, showing an average player satisfaction improvement of 20% using reinforcement learning. From an analysis of multi-output regression's results, it is clear that its root mean square error (RMSE) value is smaller than RL's standard deviation for each parameter, and thus the predictions are accurate and comparable to RL's.

The techniques used in this work for artificial player generation, gameplay simulation and machine learning analysis are extremely useful for the project at the core of this dissertation. The platform developed may implement adapted versions of these techniques, expanding upon their depth and complexity while re-tooling them to better fit the topic at hand.

From the rest of the results, it is also possible to conclude that personality differences were not accentuated enough for either machine learning algorithm to associate radically different parameters to each personality type, preferring instead to find parameters that maximize the average experience across all personalities (even if some suffer from this). Still, personality differences are slightly felt in the parameter choosing, with the speed fields showing to be especially relevant.

Analysis show that multi-output regression and combined individual regression have comparable results, with multi-output regression being preferable due to being less computationally demanding. This project shows the viability of multi-output regression for game adaptivity, confirming that reinforcement learning is also a standard way of achieving a positive effect on player's satisfaction.

	Training		Test	
	Average	Deviation	Average	Deviation
Brick Height	2.9	1.5	2.2	1.3
Paddle Speed	17.6	7.8	16.5	3.5
Paddle Length	28.2	4.7	28.5	4.9
Ball Speed	4	3.4	2.8	2.6
Ball Size	4.9	1.3	5.3	1.1
Wins	105/7004		878/1593	
Time	23.7	24.7	69.9	48.5
Satisfaction	9.6	3.1	13.5	3.3

Table 3.1: Reinforcement Learning overall results

	Multi Output RMSE	Individual Regression RMSE
Brick Height	1.293	1.277
Paddle Speed	2.836	2.799
Paddle Length	1.092	1.076
Ball Speed	1.415	1.575
Ball Size	1.039	1.038
Global	1.67	1.68

Table 3.2: Multi Output and Individual Regression Results

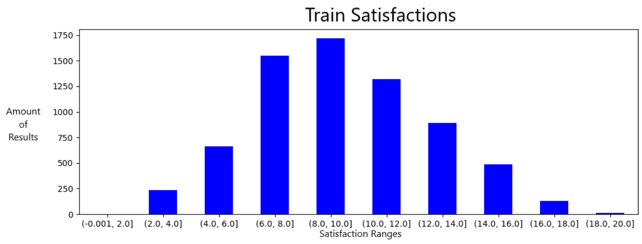

Figure 3.2: The RL training satisfaction distribution (frequency)

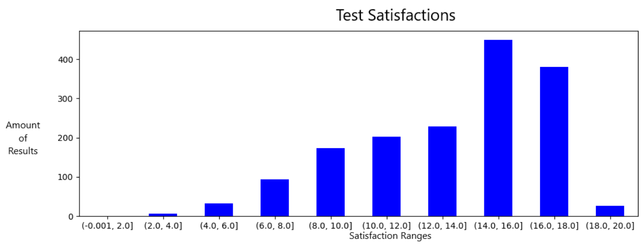

Figure 3.3: The RL testing satisfaction distribution (frequency)

3.3 Proposed Solution

This section covers the initial theorized solution that was developed in this dissertation. Although most its central concepts were implemented in the final version of the project, many of its technological proposals were not met due to a variety of constraints found during development.

Due to the changes, this section can be seen as a first draft that then suffered heavy alterations during development, with some of its secondary components getting discarded and replaced by more pragmatic and effective alternatives. The full list of changes and their respective justifications, along with a conceptualization of the new components, is present in the following section (and especially in section 4.2). Regardless, many of the concepts put forward in this section that ended up not being implemented are excellent candidates for future implementations that do not have the same constraints as this dissertation, and thus their elaboration is still important.

3.3.1 Overview

The solution explored within the scope of this dissertation is a data analytics platform designed with location-based pervasive games in mind. This platform receives player and game data from the game itself and cross-references it with geographic data, obtaining a clear image of the player's experience and how the game's contextual data handling had influenced it.

Each player's profiles are taken into account to balance the game for different personalities and play styles, maximizing satisfaction. These player profiles would be measured by player clustering techniques as outlined in the previous chapter.

All of this data is analyzed to provide developers with information on what oversights and biases are in their content generating system. Using the previously mentioned frameworks, the platform gives some suggestions about what changes should be implemented to improve balance and KPIs.

Ideally, developers would be able to insert their own game's structure in the platform in a way that allows the in-game elements to be quantified meaningfully. In an attempt to accommodate as many location-based pervasive games as possible, a general model was designed to abstract many of the popular concepts in pervasive games. The structure of this model can be seen in Figure 3.4, which displays its UML diagram.

To fit as many games as possible (and taking into account all the diverse structures that could entail), some concessions had to be made in the model that would mandate developers then add their own constraints to each object introduced in the database, in order to make the analysis more accurate to their own game. Taking the example of Pokémon Go: PokéStops, Pokémons and Missions are considered by the model to be ChallengeTypes with their own respective Challenges, and thus discretion should be applied in the analysis. Furthermore, the more specific the requirements are for an game concept, the harder it is to integrate them into the abstracted model and the more likely it is that there would have to be some customization of the platform to fit each respective game.

As much as the goal of the project is for the platform to be universal in its analysis of pervasive games, the thesis has a limited scope; as such, developing this structure mandated that we'd take popular existing examples into account and then allow future developers to customize the platform as necessary. Other location based games can choose to map these concepts in their own game, or leave some elements outside of the analysis.

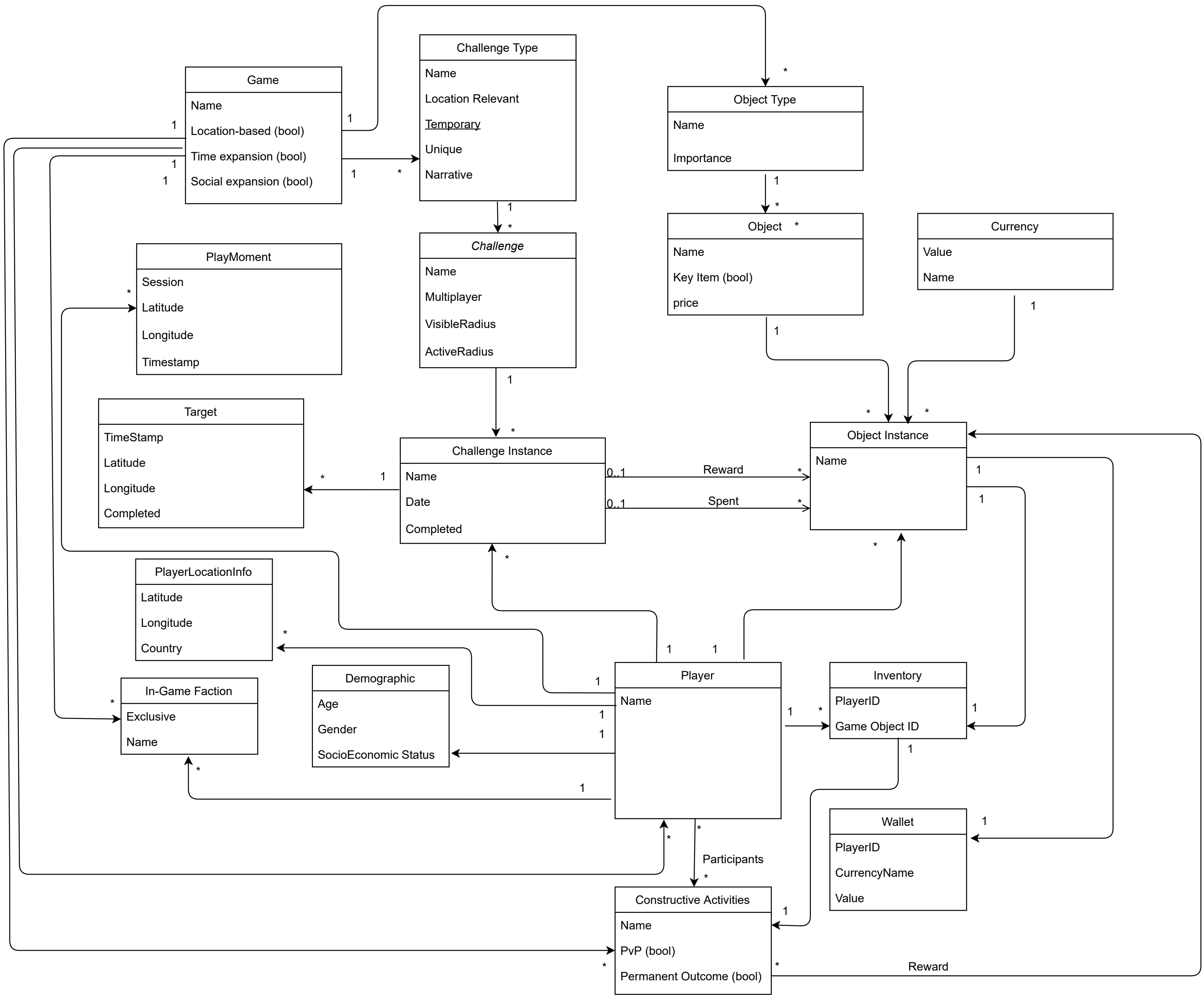

Figure 3.4: UML diagram for the conceptual model

Games have their features defined regarding what pervasive elements are present (location, time and social elements as described in the previous chapter). The central concept behind the structure is the Challenge, which can be abstracted into a variety of Challenge Types. The ChallengeType allows for the abstraction of big concepts such as events, missions, and smaller ones like finding a marker that re-stocks player items all in one entity, while still allowing their differentiation. Challenges of each type are then associated with them, still able to keep their uniqueness while being of the same type. Once a challenge is attempted by a player, a challenge instance is generated that is unique to that player, location and date. Each challenge instance can have multiple targets, "steps" that have to be completed for the challenge to be considered a success. A similar abstraction to Challenges occurs with Objects, which again can have be of many different types, from consumables to cosmetic items, and which become "Game Objects" once associated with a player. Game Objects can be objects or a form of currency, which then gets added to the wallet instead of the regular inventory.

From the player side, they're associated with all their items via the inventory and their in-game currency via their wallet. A record is kept of their attempted challenges (with or without completion) and a detailed profile containing their demographic data and KPIs is kept. Lastly, players have the option to engage in "constructive activities" - player-initiated gameplay moments that allow for the usage of in-game resources. These can be building a house in Minecraft Earth, or initiating a battle with a friend in Pokémon Go. These constructive activities are entirely distinct from any type of Challenge due to being created by the player, while Challenges are all necessarily generated by the game itself.

3.3.2 Balance and Injustice

Core to the premise of this dissertation are the questions of what is game balance and what is injustice in a video game, how can these concepts be detected and measured, and most of all what are some good solutions that can be proposed? Do the concepts ever become at odds, or can the search for a fairer experience lead to a less interesting and varied one?

Video game balance is a much talked about subject, but exploring this same subject regarding pervasive video games is a notorious gap that hasn't been filled by many other academic works (as was denoted in Chapter 2). This leaves the genre of pervasive games without easy ways to ensure their games are not committing mistakes that other genres have avoided for decades.

Following the cue given by our definitions of fairness in machine learning (subsection 2.2.3, there are variety of approaches that are viable and the proposal can feasible take multiple while not compromising on its goals. Basing our outlook on injustice on a combination of predictive parity and test-fairness allows us to identify injustice on different levels; these standard are easier for an analytic platform than causal reasoning or similarity-based measures, which require awareness of many contextual outside factors the platform may not be privy to or that do not translate directly into the numbers. The unfairness will be treated as an anomaly of the system that must be detected and corrected.

Identifying if a game is balanced or imbalanced is usually easier than determining fairness. Calculating that an item is over or under-used, or that a challenge might not be have a high enough player success rate, are usually trivial matters. Fairness, in the scope of this work, pertains specifically to injustice towards specific groups - that is, when players are disadvantaged based on factors that are in general terms outside of their control, such as the place where they live or their specific consumer demographic. Often, these issues don't come up in general balance analysis, as these focus more on the success of game components rather than on the different experiences that some demographics might have. This, along with the definitions of fairness as they pertain to Machine Learning defined in subsection 2.2.3, are at the core of this work.

Pervasive games have specific traits that require this kind of analysis to be applied to other factors, usually of a geographic nature. Ensuring that certain regions or types of landscapes aren't significantly favoured by the algorithm is important - in pervasive games the threat of imbalance has deeper consequences than in other game genres, as "the metagame" can form around specific areas or dates, which is much harder to counter-act as a player.

Given that pervasive games are very often multiplayer, it is important to keep a degree of variance in player experiences while maintaining balance and fairness. Not only is the unique experience valuable on its own merits, as players with different desires and backgrounds should have the capability to express themselves differently within the game's systems, but it also serves the purpose of player interactions with one another do not becoming repetitive. After all, using the example of Pokémon Go!: If everyone has the same Pokémon, not only does the game become less appealing, but player vs player content loses a lot of its luster.

In a concrete sense, the platform will ensure player KPIs do not have any major statistical discrepancies based on their respective demographics or regions, will analyse player behavior to detect outlier challenges or items and will have a trigger system in place the statistics reveal worrying trends - be them of a balance or an ethical nature (for example, young players who have unusually high lifetime value indicators).

3.3.3 System Architecture

The proposed solution is composed of the integration between two modules, one comprised of the game and integration service that reports on data, and the other containing a web API and analytics platform that cross-references data from the first module, the database, and third party sources (GPS info, weather info, among others) to deliver conclusions to the game designer.

The components are as follows:

- **Phone component** - Composed of the game and a shallow integration service, it collects superficial data such as player locations, session duration, session frequency, and others.
- **Web API** - A web API that works as an interface between the integration service, the database, and the analytics platform.
- **Database** - Where all the data collected from the integration service, collected by the third-party sources, and cross-referenced and analyzed by the analytics platform is saved.
- **Analytics** - An analytics server capable of handling the data, processing it, running some of the referenced ML techniques, and providing conclusions based on mentioned frameworks.
- **Third Party Sources** - Various sources such as GPS data, weather APIs, or population data APIs that can be cross-referenced with the existing data

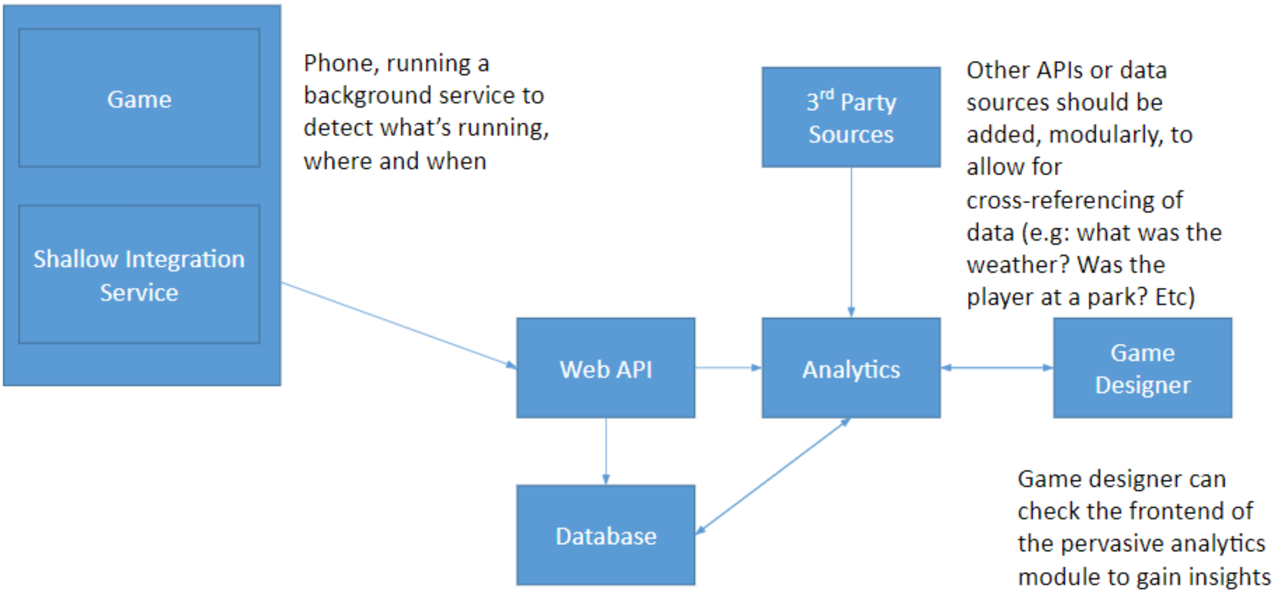

Figure 3.5: System architecture and modules

- **Game Designer** - The end-user, who interfaces with the analytics platform and integrates the service into their game.

3.3.4 Proposed Features

The following list of system requirements outlines what features should be functional in the final version of the project.

- **Shallow information tracking** - An android app capable of measuring usage for the pervasive game must be able to track play sessions, play session time, and player location.
- **Data communication** - The shallow integration service must communicate with the web API and have it store its data.
- **Data storage** - Session data and previous analysis must be stored in the database for future use and analysis.
- **Developer Access** - The developer should be able to access the analytics platform via a browser or through local requests.
- **Cross-Referencing with Third-Party Data** - The analytics platform must be able to request information to third party sources to cross-reference it with other data (can be modular according to user requests).
- **Statistics calculation** - The analytics platform must provide various relevant statistics and KPIs to be requested by the user.

- **Trend finding** - The analytics platform must be able to find trends, both positive and negative, that the developer should take into consideration.

- **Player profiling** - The platform should be able to profile players and calculate what features would better satisfy them (using KPIs as a reference).

- **Suggestions** - The analytics platform should be able to provide the developer with suggestions, based on some of the previously mentioned frameworks, the statistics, and the trends it found; these suggestions can be based on errors and imbalances that must be fixed or on overall features that would improve the game.

A shallow information tracking service is essential to track player activity and as a source for location data. The data communication and storage features are critical in the functioning of a complex system with so many moving parts. Collecting and analyzing data is necessary to obtain conclusions regarding the game elements and offer suggestions to the user. The third-party sources are vital to have as references, allowing the system to determine what outside factors may be relevant to player behavior. The statistic calculation, trend finding, player profiling, and suggestion features are various tools that help find imbalances and injustices by analyzing the player data. As such, they're the core of this project.

3.3.5 Technologies Used

The following is a list of the technologies that are intended for use in the development of this project:

- **REST API** - REST is a type of API architecture designed to handle HTTP requests, which will be the structure used for this project's Web API.

- **Javascript** - A very flexible programming language that will be used to create both the API and the web front. A Javascript framework such as React or NodeJS may be used.

- **HTML, CSS** - Two vital technologies that may be used to build a web front.

- **Flutter** - Flutter is a framework to develop android apps, which could be used in the shallow integration service.

- **SQLite** - A relational database system that can be used to build the database and is easy to integrate with all of the other chosen technologies

- **Python** - An advantageous programming language to easily implement complex machine learning techniques.

3.4 Summary

The first section of this chapter is an in-depth explanation of the problem this thesis is aiming at solving. While imbalance that leads to social injustice is common in many types of media, its effects particularly notable in pervasive games as they often deal with the movement of people in real space. Despite there probably being no malicious intent on the developer's part (especially as they ended up alleviating or outright solving some of these issues with their internal data handling system later on), oversights of this type should not be this common when dealing with applications with such a huge scale.

The preliminary work is especially relevant for three different aspects of this project: the machine learning techniques used, the insight gained into game adaptivity (and how it affects players) and how to effectively design a simulation of player behavior and feedback. This last aspect should be highlighted further, as it might be necessary to implement such a system for the crucial data collection segment of the proposed project. Given the lack of open user data in commercial pervasive games, obtaining a large amount of player data will be difficult, and simulating that data (instead of developing the shallow integration app) is one of the available options.

Regarding the project proposal itself, some of its aspects are still not final and were added to this model for a comprehensive overview of all the possible technologies. For example, the front-end of program developers will interface with is still not completely decided, as experimentation might lead us to prefer certain solutions over others (command-line, website front-end and phone app being some of the alternatives). The shallow integration app is another aspect that, as detailed in the last paragraph, is not guaranteed to be a part of the final product.

In sum, this chapter provides an overview on the specific problems this thesis takes on and the proposed solutions. It is important to note that the three expansions of the magic circle will be accounted for when it comes to balance, and not just the spatial expansion, despite it being the major focus of most popular pervasive games.

Chapter 4

Data Analysis Platform Implementation

Contents

This chapter tackles the development of the data analysis platform that aims to solve the problems stated in Chapter 3. The present chapter covers the platform itself and many of the other tools developed in the scope of this project that was invaluable to obtaining the desired results. Due to various factors, the structure of the solution has been radically altered from what was described in Chapter 3. These changes will be addressed in the second section of this chapter, along with their reasons.

Section 4.1 provides an overview of the implementation, covering the overall structure, technologies and strategies adopted. Section 4.3 is dedicated to describing how the data for the analysis was generated, including deep explorations of how the player data was created to emulate realistic humans and how the simulation of their actions within the game functions. Section 4.4 is about the core of the platform, the tools used for analysis, which are expanded upon in their dedicated subsections of basic analysis of the database, interactive maps and a more in-depth analysis aided by machine learning.

Once the implementation has been detailed in full, section 4.5 includes a discussion about the challenges and noteworthy details of the development process, while the last section, section 4.6 provides a synthesis of the chapter as a whole.

4.1 Overview

This chapter tackles the development of the data analysis platform that aims to solve the problems stated in Chapter 3. The present chapter covers the platform itself and many of the other tools developed in the scope of this project that was invaluable to obtaining the desired results. Due to various factors, the structure of the solution has been radically altered from what was described in Chapter 3. These changes will be addressed in the second section of this chapter, along with their reasons.

The pervasive game data analysis platform is software aimed at developers of pervasive games with the intent of supporting their game design decisions.

The platform provides knowledge of how the player demographics are interacting and being affected by the game's systems. It displays the distribution of activity and game events over dynamic maps and provides predictions of how different circumstances - be them real-world occurrences such as holidays or gameplay challenges by the developers - might affect the results obtained.

To do this, two models had to have been developed for the platform to be complete and functional: a model for creating fictional players that very closely resemble their real-life counterparts, and a simulation tool that, using these fictional players, simulates player actions for a developer-determined amount of time. The results from these simulations would then be analyzed by the various tools in the platform and allow for the developers to test any change they might want to implement within their game.

The models in question were implemented on top of a database and project implementation that closely follows the structure depicted in 3.4. While each of them will be expanded upon in their own following sub-section, it is vital to give a short description of how they were implemented. The simulation system takes player data and, given their personality, past game experiences, the game activity, and other player activity near their location predicts each player's likeliness to interact with the game's systems. This simulation goes on for a user-determined amount of days, simulating the impact of weather, holidays, the current time, and day of the week, among external factors.

Regarding the player generation system, the realization that it would be required (rather than what was planned) happened midway through the project's development. Its relation to the rest of the system must be outlined. The reasons for these changes to the planning are summarized in section 4.2.

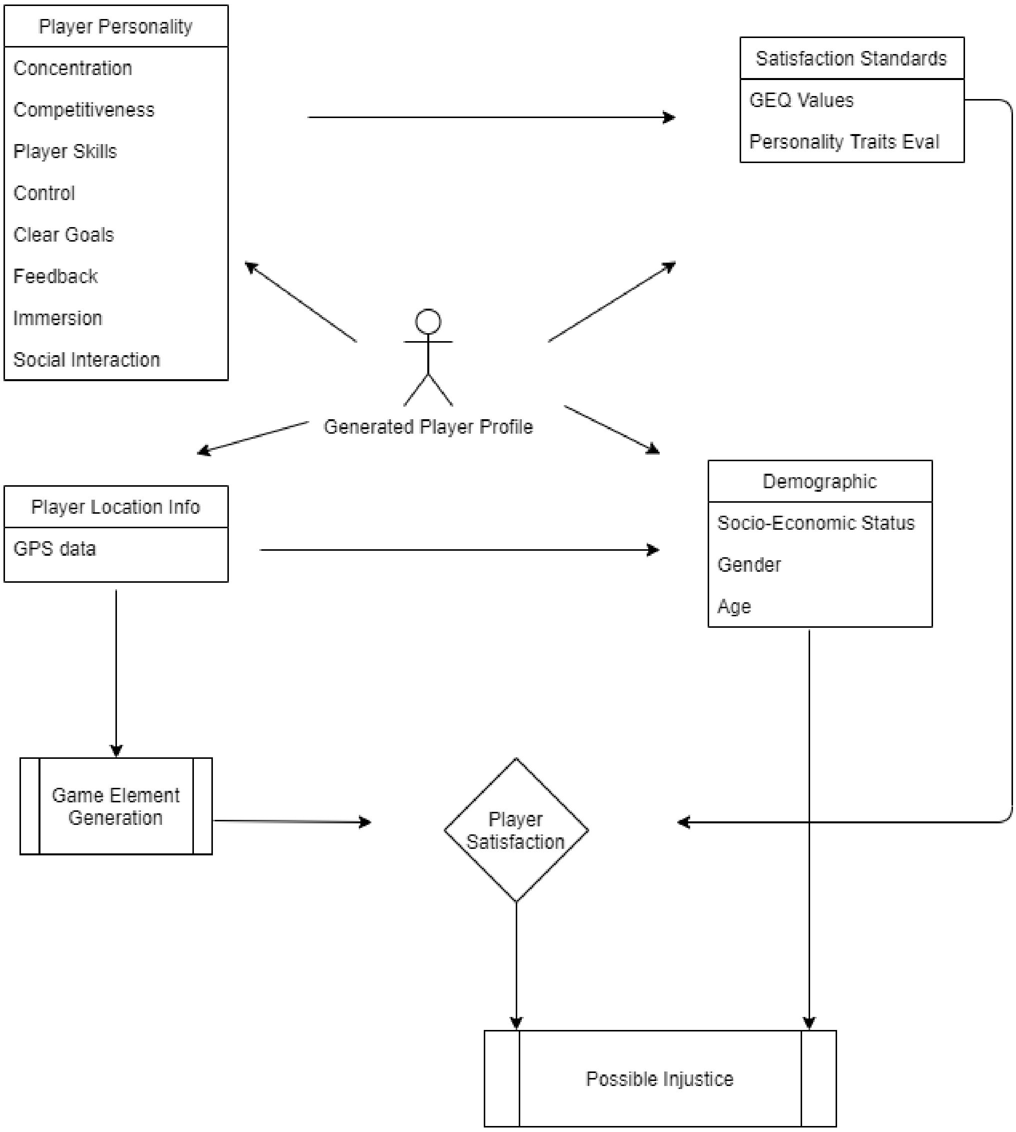

Figure 4.1: UML diagram for the player relations

In this diagram, it is possible to see in detail how a player is modeled in this platform. A player profile is comprised of their personality (using the characteristics outlined in subsection 2.2.4), internal satisfaction standards determined by the clash of their personality and game experience, their demographic data, and their physical location. Finally, player satisfaction is calculated, which translates directly into how likely the player is to further engage with the game - the type and intensity of that engagement being mediated by their personality values, leading to more playtime, challenges completed, or purchases. How the player data itself is generated and determined will be expanded upon in subsection 4.3.2.

An issue that revealed itself within the implementation of the model depicted in 3.4 is its simplicity, as it was designed for the developers to retroactively fit their player data to its molds for subsequent analysis. For example, if the developers want to test the introduction of a Challenge that would change the spawn rate for other Challenges once completed, this would be very hard to represent and simulate (it would, of course, be trivial to analyze, which is a scenario where the data could only be retrofitted). The goal of the simulation is to predict issues manifesting from simple challenges as a balancing tool and not as a sandbox for wild new ideas. The simulation aspect is, of course, also very important for the thesis as obtaining genuine player data in the scale required for a meaningful analysis was not viable.

To ensure the ease of use of the platform, a graphical user interface was developed that facilitates tool navigation and information display. The main menu displays the main options described earlier:

- **Create** - Completely wipes the database.

- **Simulate** - Source for the maps used in the dynamic map displays, coordinate references, and source for relevant Overpass API tags.

- **Analysis** - This option leads to further sub-menus that display superficial analysis of the data. Some of the tools available include displaying average player KPIs, data filtering, and various types of graphs.

- **Interactive Maps** - Plots player activity such as completed challenges and player activity heatmaps in an interactive map.

- **Comprehensive ML Analysis** - Complete analysis of the database using ML to predict problematic trends and including a trigger system for fairness and ethicality.

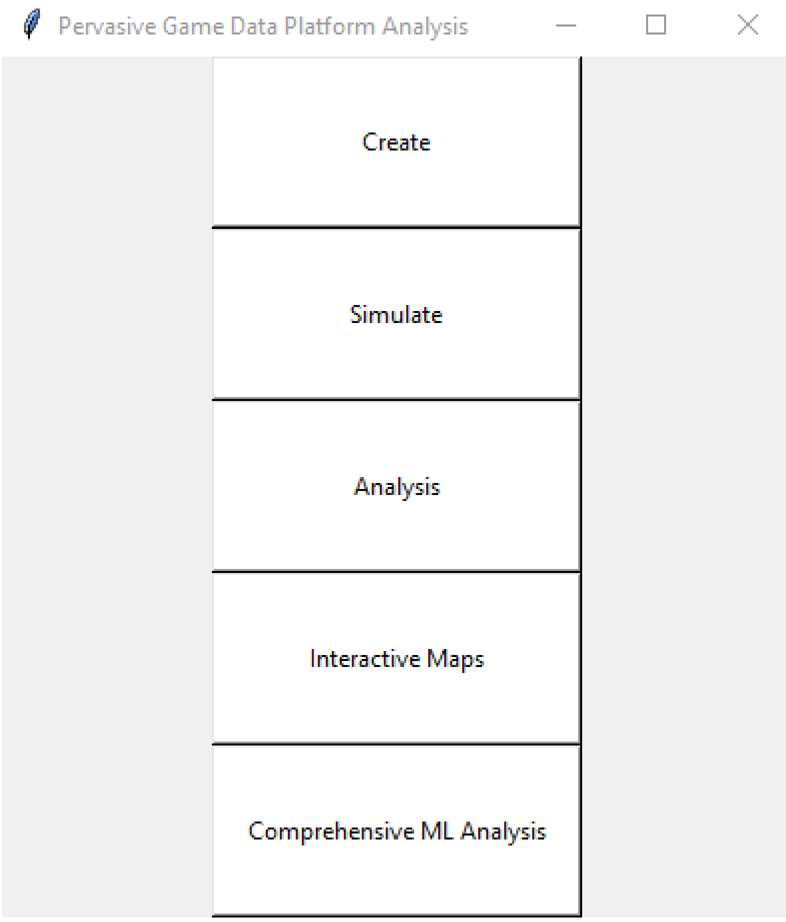

Figure 4.2: The main menu of the GUI

The graphical user interface (GUI for short) aims to be simple and concise, with much of the information regarding the status of processes being conveyed via the command line, and only the actual analysis being written out in the GUI windows. The first two options only trigger their process, pausing user inputs until the process is complete. If the database is empty, "Create" needs to be the option chosen by the user, and the same applies to "Simulate" if the database does not have any user activity logged. Without fulfilling these conditions, the analysis options will not work as intended, given that there is no data for them to analyze.

The analysis menu features a superficial analysis option and is also a gateway to other, more specific analysis pages - graphs, filtering, and machine learning with filtering. These options and how they function will be expanded upon in Section 4.4. When pressing the "Analysis" button, the KPIs are calculated for every player in the game and displayed alongside the game's averages. This process provides developers with a quick overview of their game's success or lack thereof, displaying the most important indicators for player activity. The KPIs for every player are shown to give a more granular perspective, allowing developers to analyze individual experiences.

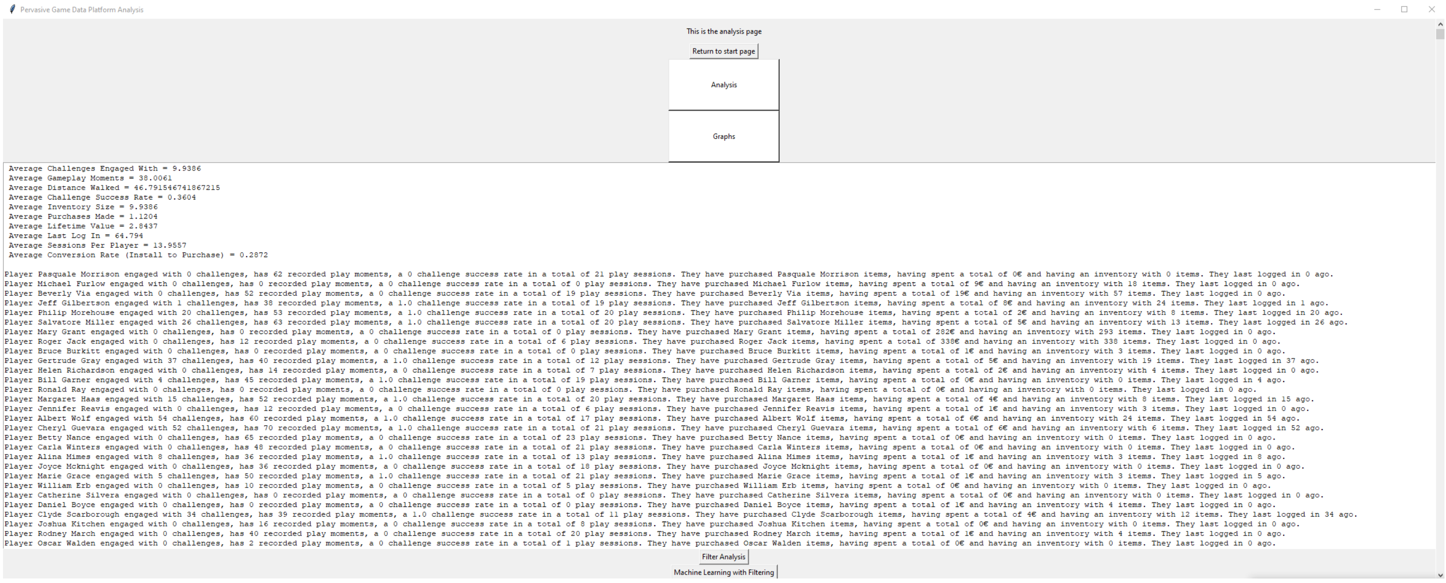

Figure 4.3: The analysis menu

The Interactive Maps option opens a browser window for the developer with an OpenStreetMaps map that has been populated with markers or heatmaps displaying information of their choosing. The default option is for the player activity to be shown in heatmap form, while challenges and their completion are presented as markers. The following is a simple example of the interactive map that catalogs the spawn of one Challenge in various locations.

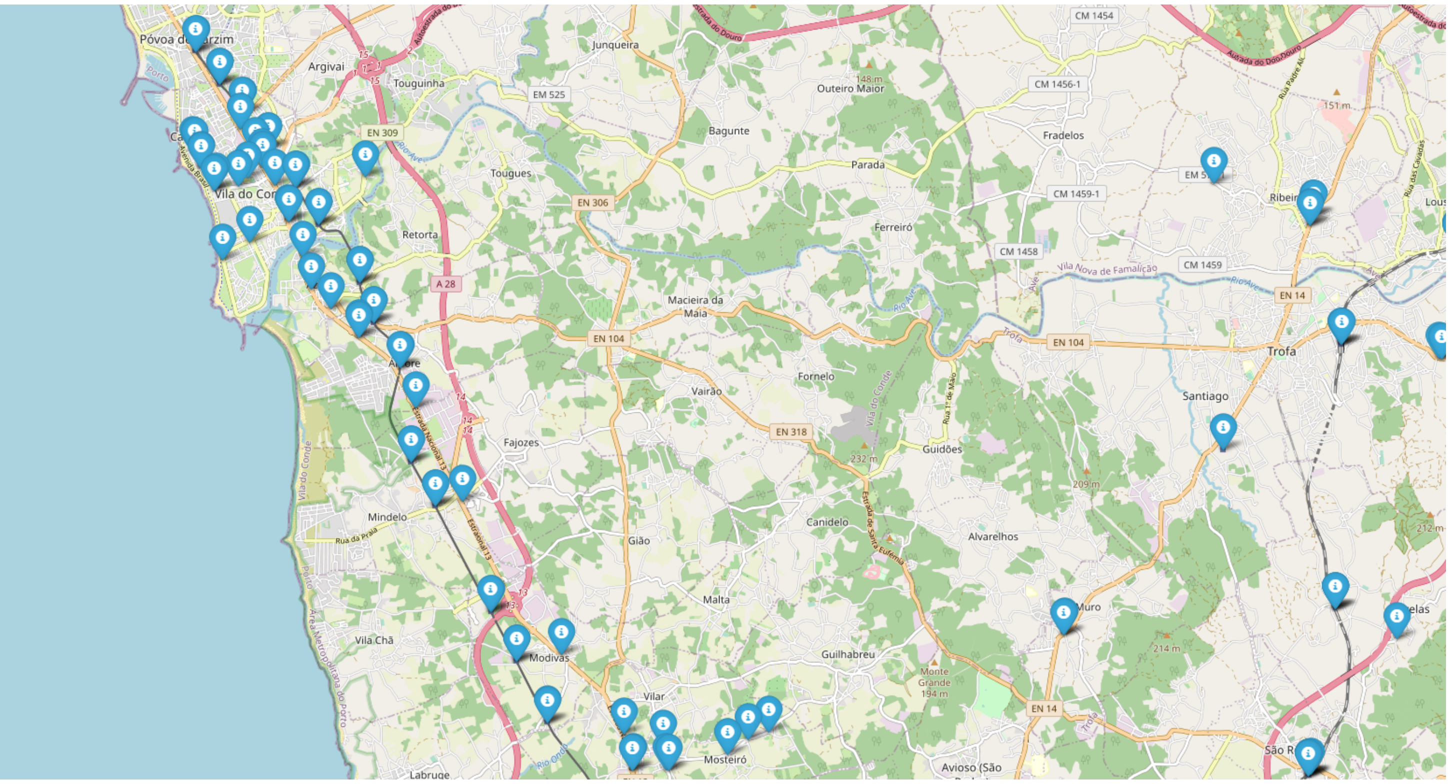

Figure 4.4: An interactive map example

Finally, the "Comprehensive ML Analysis" option opens a menu with a textbox similar to the Analysis menu, but the only available option is to run the comprehensive analysis. Once run, the predictions and warnings of the machine learning model are displayed in that same textbox.

The primary programming language used in the platform is Python, being chosen primarily due to its flexibility, ease of use, extensive machine learning library inclusion, and having straightforward interfacing with external APIs. The database language is SQLite, a fast and efficient language for high volume projects that are easily integrated within the Python environment.

The platform uses the MVC (model-view-controller) software design pattern to maintain the code as organized as possible and enforce modularity, which is essential given the scale and variety of the project's requirements. Following these guidelines, the model component is comprised of the concepts outlined in the 3.4, such as Challenge, Player and Object, the view contains the GUI elements and all of the plotting and mapping features, and the controller includes all of the analysis functions, data generation features, machine learning technology implementations and the remaining necessary platform utilities.

At the center of this structure is the class "GameManagement", which, as the name indicates, is responsible for managing all the game and player data. This includes saving and loading data to the database and instantiating other management classes, such as "PlayerGeneration" and "Simulation". The "GameManagement" class is then subject to analysis directly, whenever it is requested.

The following is a list of the external APIs that are essential to the platform and the purpose of their use:

- **WorldPop [15]** - Used to obtain general information on countries and cities, such as population density, socio-economic status, education level, among other indicators and indexes that are relevant for demographic analysis.
- **OpenStreetMaps [56]** - Source for coordinate references and for the Overpass API tags. Folium and Overpass both rely on OpenStreetMaps for their core functions.
- **Overpass [23]** - API that provides information on landmarks and infrastructure.
- **Folium [1]** - Allows for the creation and display of interactive maps.
- **Open Weather Map [14]** - Source for weather conditions in specific days. Mostly used in experimental phases as fixed weather conditions were implemented for the simulation.
- **Nager.Date [13]** - Provides cross-referencing for holidays and other special dates that help make the user simulation more realistic
- **Humanitarian Data Exchange [3]** - The HDE was the main source for coordinate-specific population data, providing demographic quantities and ratios.

It is also important to mention the relevant libraries featured in the project, namely the Python data handling library Pandas [6], Python UI utility TKinter [12], graphical tool Matplotlib [4], and Machine Learning library SciKitLearn [7].

The full project, including the original database files generated and used to obtain the results, can be found in the author's GitHub repository, JoaoAlvaroFerreira/Pervasive-Game-Balancing [1].

4.2 Changes

It is quickly apparent, from a simple contrasting of the project description in Chapter 3 with the Overview presented in the previous section, that the platform components suffered massive changes, with some of these components being replaced altogether. Although it might seem that the project is radically different from what it got described during the planning stages, the goals and concepts that drive it to remain very similar, as does the core of its product - a software platform that game developers can use to test the fairness and balance of their games.

A shift that occurred during development was wanting the work to become more of a theory-bound project rather than a fully polished, usable final product. The initial scope of the project - split through multiple components that developers would have to combine together - could possibly be a very interesting commercial product but would not lead to very useful results from the standpoint of a dissertation due to the constraints faced. Although the initial draft depicted in Section 3.3 is not the solution that was implemented, we maintain that if it could be achieved in full as it was planned, it would bring the expected results. Unfortunately, many of the issues we predicted would get in the way of the development came to pass, and adjustments had to be made.

One of the main issues we faced and that mandated a change was our plans regarding the collection of player data. Ideally, player data would be collected in a massively popular pervasive game (such as Niantic's games), which would mean that the bulk of the work would be dedicated to the analysis components of the project. We were aware of that impossibility from the planning stage - not only because there are no viable sources for that sort of information, but also because it would be illegal (on our part, as a third party unrelated to the developers) to collect player's data without their consent.

Our solution to that problem was simple - develop our own mobile application that would passively collect data from its users (running passively in the background), with those users having consented to the process. After that, this data would be analyzed as if it was the game's own. A few issues arose from this idea.

Firstly, it would be almost impossible to gather a large enough install-base for this shallow integration service for the data gathering to be meaningful. Constructing a machine learning model that can predict such complex scenarios accurately requires very high volumes of data - our current database employs a simulation of over 10000 players, and this still means there are quite a few blind spots. Aside from the friends and family of the authors of this dissertation, not many people would consciously consent to install a third-party application from an unknown source whose only purpose would be to collect their personal data. A small population would then lead to biases and bad conclusions.

[1] https://github.com/JoaoAlvaroFerreira/Pervasive-Game-Balancing

Secondly, for this application to be useful, it would have to be hosted on the App Store and receive regular updates to ensure its functions were kept working as intended with the current state of the WebAPI. The implications from that are much more development time and work than expected and a relatively high cost for the hosting of the application on the store.

Lastly, the player data collection via the application would not have the level of complexity that a real, deep interactive experience would entail. It would run concurrently with a pervasive game, but the extent to which the game data could be logged on the shallow integration service is questionable. In the worst-case scenario, it would only amount to a collection of location patterns; and while that is still very useful, it would not grant the project the depth it deserves. Given all of these downsides, finding an alternative was a necessity.

The alternative that was decided upon was simulating the players and their interactions with the game altogether. This process is explained in detail in the following section, and it is composed of two main parts - using real-world data collected from reliable APIs to create fictional humans that accurately represent their population and a simulation of the actions those players would take when confronted with the game's systems and respective context.

The second major alteration was the changing of the WebAPI service to a local, Python-based platform. The connections to the third-party sources and database remain as they were depicted in the earlier system architecture schematic (Figure 3.5), but the "WebAPI" and Analytics component were merged into one under the new local platform. The reasons for this change are closely tied to the removal of the shallow integration service.

One of the motivations behind maintaining an elaborate system with all of the separate components was for the project to display a prototype of the pipeline a developer could expect to interact with in a commercial version of this platform. Given that that goal was discarded with the removal of the mobile components, the main reason behind the development of a WebAPI also disappeared.

Alongside the change in direction regarding the project, there were also concerns regarding the amount of work, time constraints, and unnecessary technological upkeep. Maintaining the communication between a web service and all of the other components while offering an appealing and flexible front-end experience was deemed to be superfluous and a waste of resources that could instead be invested into bettering the other aspects of the project. Furthermore, the integration of the developer interactions, the analysis platform itself, the database, and third-party API calls into one self-contained piece of software allows for easier bug-fixing and makes it easier to ensure the quality of the flow of information at all times. The fewer modules with different technologies are present in a project, the easier it is to guarantee that it is working as intended.

To offer a more detailed and analytical breakdown of the changes that took place in comparison to the original draft, the tables 4.1 and 4.2 list the proposed features and components, along with the changes they were subject to.

System Architecture Components	Implemented?	Changes
Phone component	No	Replaced with player generation system
WebAPI	No	Features integrated into local platform
Database	Yes	Implemented as intended, with minor attribute changes
Analytics component	Yes	Implemented in a Python local platform rather than on a server
Third party sources	Yes	Implemented as intended

Table 4.1: System architecture changes

Proposed Features	Implemented?	Changes
Shallow Information Tracking	No	Replaced with player generation system
Data Communication	Yes	Done internally rather than with calls between different systems
Data Storage	Yes	Implemented as intended
Developer Access	Yes	Available via local requests and the platform GUI
Cross-referencing with third party data	Yes	Implemented as intended
Statistics Calculation	Yes	Implemented as intended
Trend finding	Yes	Implemented as intended
Player profiling	Yes	Implemented as intended
Suggestions	Yes	Implemented as intended

Table 4.2: System architecture changes

As is visible in these tables, nearly all of the intended features and components were kept, with the exceptions being the mobile and web components that were deemed unfit to achieve the goals of the project.

Finally, when it comes to technologies, Python and SQLite were implemented as initially intended. However, due to the changes mentioned in tables 4.1 and 4.2, all of the web technologies

(REST API, Javascript, HTML, CSS) and mobile technologies (Flutter) that were intended for use in the initial planning ended up not being used.

4.3 Data Generation

As was described previously, the data generation components are a core part of this project. Despite not being essential to the analysis platform itself, it is very important for the project, as it is the only way to display the analytic capabilities.

Due to the fact that the analysis systems could not even be tested before the data generation systems were working (due to otherwise not having any data to analyze), the data generation components were built first. After building a database following the schematic of Figure 3.4, the API calls were first tested. Given that the gameplay elements of pervasive games are very reliant on the location data and time-dependent characteristics, establishing a smooth interface that allowed us to go from API calls to relevant information was imperative.

In order to create a flexible system for API calls, the "API module" was developed that included different functions for each one of the APIs chosen (that were detailed in Section 4.1). Each function takes different arguments, depending on what is being asked, and then filters the responses.

After the database was created and the APIs were ready, a systematic way of managing the data that would then be generated was needed. To do this, a GameManagement class was developed that included a pipeline for saving and loading data to and from the database. If the selection in the GUI is "create", the database gets its data wiped, and the data generation functions are run on an empty database, with objects getting inserted into the database in bulk, in the end, to ensure efficiency with large amounts of data. In every other user option, the first action by the GameManagement class - after receiving the database connection - is to load the data from the database, class by class, and inserting it all into class member lists.

The loading of objects was designed to depend on their relations, ensuring that any "error objects" - such as a Challenge with an invalid ChallengeType - were not loaded. This was done by loading hierarchically, for example, first loading the ChallengeTypes and then selecting the database for Challenges related to each type, in order. For objects that might already be loaded into the class lists - such as an item that is a reward for a challenge, but that due to the hierarchy has already been loaded - these are searched internally and not on the database.

Initially, data was intended to be generated over all of Europe, which was then reduced to the Iberian Peninsula to narrow down the scope. Issues that occurred during the data generation that will be described in the following subsections led to the necessity of making the space covered by this data generation much smaller. The authors of this project then chose to cover only an area in the Porto district in Portugal, from latitude 41.1042 to 41.2788 and longitude -8.6627 to -8.4361. This decision was made due to the authors living in the region and thus knowing details that might be relevant for the development of a pervasive game (such as simulating the impact of a local holiday or what areas have more landmarks and thus require more focus).

4.3.1 Game Generation

Generating the game's components was a complicated process. As this segment is where we would be generating the data for a fake game to then analyze, some creative input was required to ensure that the game had a direction for its content and gameplay and to ensure that the generated content was consistent with this direction.

The goal is to simulate a realistic pervasive game, after all, and a game composed of completely antagonistic gameplay elements that do not coordinate would not be realistic (for example, requiring a specific item to do a common Challenge but then that item being extremely rare). We also wanted the analysis of this game to be a display of what the platform can do for a majority of location-based pervasive games, and thus it was in its best interest to be relatively simple. Ensuring that its challenges function similarly to the challenges in a majority of popular pervasive games leads to the results obtained to be more useful and relevant.

Due to this, a few "template challenges" were decided upon:

- **Item Spots** - Challenges that require no item investment from the player and that offer basic items in return but that are only active once per hour. These require the player's location to be the same as the challenge.
- **Mission** - Challenges that require the player to complete item, location, or time requirements, usually in combination. Success is obtained once the conditions are met, with "failed attempts" not consuming items.
- **Conflicts** - Challenges in a specific location that the player does not have a guarantee of succeeding and that require item investment to attempt (which are lost after use).
- **Raids** - Multiplayer challenges that require collaboration between multiple players in the same location at the same time. These provide significant rewards.

Given the influence of Pokémon Go! on the genre, all of these challenges can be recognized as local equivalents of Pokémon Go! challenges. Following its footsteps, the decision was made to generate the challenges in a similar way, relying on geographical landmarks and activity hotspots for gameplay element creation.

Item spots are generated based on data collected from the APIs. Porto is a town with a fair amount of historical landmarks and extensive public transport systems (including metros and buses). Knowing this, the Overpass API was queried for all nodes of "public transport" and "tourism" within the latitude and longitude limits mentioned above. These nodes were then used as a reference for Item Spots that provided players with two distinct item types. This allows for some variety while still controlling closely which items the players can easily replenish based on their location.

Issues came up during the game generation regarding the number of API calls made being excessive and not only hitting hard API limits but also massively slowing down the data generation. This was one of the reasons for reducing the size of the land covered by this simulation. This reason is also why only two geographic queries/references were used for the item spots, as we did not want to generate an overwhelming amount of challenges for players. Furthermore, visualizing these challenges became very hard on the interactive maps, as too many map markers led to an incredibly cluttered map that was very slow to interact with.

The missions are created repeatedly, with players getting a new mission every two days. The content of these missions is randomly generated based on the items listed in the database, guaranteeing that the items invested and the rewards are not the same.

Conflicts are intended as one of the main focuses of the game from a player's perspective, with much of the game's ecosystem being centered on providing resources for players to challenge Conflicts. These conflicts then reward players with unique items that they would ideally be able to use in raids.

Due to time constraints and the complexity of the task at hand, raids ended up not being implemented completely and are not considered in the final analysis. The way raids were implemented entered in conflict with the player-by-player simulation system, which will be described in the following sub-section and, due to the mechanics involved, solutions ended up providing unsatisfactory results that would only make the analysis of the other gameplay elements less clear.

Other planned gameplay elements were not implemented in any way, as their inclusion would be too complex, and they would require different perspectives to be relevant for analysis. These gameplay elements include the entirety of the "constructive activity" object, which would use the items obtained from Raids and Conflicts for further goals (including, possibly, player interaction or permanent outcomes). In-game factions were also not fully implemented, given that once Raids and Constructive Activities failed to be finished, this social aspect of the game would be pointless and not lead to any meaningful conclusion. Challenge Instances were simplified, not including the segmented "targets" that were planned for, with complex multi-stage missions being instead implemented as multiple missions. Targets are still built into the models and usable if the data is inserted manually, like by converting another pervasive game's data to work on the platform, but they are not considered by the data generation system. Lastly, in-game currency was not used in a meaningful way, as its inclusion led to inconclusive results, and it was replaced by direct purchases by the player. However, as with Targets, it is still built into the Models even if not generated by the data generation systems.

Along with challenges, items are at the center of the gameplay experience. Three main item types were implemented: cosmetic items, conflict items (such as Pokéballs in Pokémon Go!), and challenge boosting items, the last of which increase the rate at which conflicts spawn for each player. Each item type has multiple items of varied quality and price (in the in-game shop, available for purchase by every player), allowing for more variety in outcomes.

4.3.2 Player Generation

Generating a model that The player generation component of this project tries to be mostly standalone, working

Once the Create option has been selected, the Game Manager class calls the Player Generation module and calls its main function repeatedly. This function returns batches of ten players at a time to minimize API calls and ensure there are enough players in similar areas to get a good area representation, reducing the chance of outliers giving misleading results. Inside each batch, a very small random number is added to both the latitude and longitude value of each player's "origin" location to make their spread in the map more coherent.

The first step in the player generation is selecting two random numbers in the latitude ranges (41.1042 to 41.2788) and longitude range (-8.6627 to -8.4361), with a small modifier being applied (up to 0.02 in absolute). After that, a query is made to the OpenStreetMap to obtain information on their address and check if it is valid (it is not placed in a body of water and is located in Portugal). Once that is done, a fake name is generated, and the player generation is split into multiple sub-functions that represent each component of a player's profile.

As has been established earlier, there are three components to a player's virtual profile that attempt to emulate as closely as feasible the behaviors of a real person within the game's systems. To do this, it is not only imperative that the components are valid and are represented by well-developed models but also that these models are congruent with each other. This means that, for example, an Elderly player is much less likely to have a Competitive personality compared to a Young player, and that a Young player would probably have less of a chance of spending money on a free video game, so they are more likely to spend time to grind out items rather than purchase them at the shop. These ideas - modeled in Figure 4.1 - are represented in this player generation system. It is not necessary to independently generate the Satisfaction Standards during the PlayerGeneration, as they can be determined based on the Player Personality during the Simulation.

Firstly, a PlayerLocationInfo is created based on the API information and coordinates that have been chosen. After that, a Demographic is determined based on this location. CSV files containing incredibly detailed coordinate-by-coordinate demographic information from the Humanitarian Data Exchange are used as a source for demographic info. The closest latitude and longitude values in the CSV to the player's origin coordinates are found, and from that are calculated probabilities regarding the player's gender and age. The source for age demographics divides people into four groups:

- **Kid** - Ages 5 to 15
- **Youth** - Ages 15 to 24
- **Adult** - Ages 24 to 60
- **Elderly** - Above 60 years old

Wealth is calculated based on Portugal's Purchasing Power index and by its Cost of Life index, obtained in the WorldPop API, and a small random number to introduce some variance.

Adding ethnic elements to the simulation could be very interesting as it would lead to tackling the issues of racial discrimination that were already found in Pervasive Games, as was described in

Chapter 2. However, Portugal does not collect ethnic or racial data in its census for ethical reasons, so this was impossible. Furthermore, Porto is a very ethnically homogeneous region, meaning that analysis without very accurate data in a large volume (so that minorities would have statistically significant representation) would likely be inconclusive.

The last step in the player generation is creating a simulated "Personality" profile. The elements in this profile are based on the model Kalle Jeggers [41] proposed for Pervasive Game principles, with each attribute being representative of the worth of a specific value for that player. The attributes are, as discussed previously, Concentration, Competitiveness, Player Skills, User Control, Clear Goals, Feedback, Immersion, and Social Interaction. An extra attribute, "Free2Play", was added to emulate the importance of a game's mechanics, not requiring the player to spend money. We believe this attribute to be relevant to simulate the impact of Age and Socioeconomic Status on a player's disposition towards game components.

It is important that player personalities are internally consistent - a player can't value all of these attributes at a maximum, after all. So, to balance personalities, overarching personalities were designed that players can fit into (being more or less likely to classify in some of them due to their Age or Socioeconomic Status). After the "Personality Type" has been selected, each individual value is calculated based on random values, the range of which is determined based on the Personality Type. Table 4.3 provides the ranges for each personality:

	Balanced	Competitive	Relaxed
Concentration	2-4	3-5	1-3
Competitiveness	2-4	4-5	1-3
Player Skills	1-5	3-5	1-4
User Control	2-4	3-5	2-4
Clear Goals	1-5	1-4	3-5
Feedback	2-4	1-5	4-5
Immersion	1-5	1-3	4-5
Social Interaction	2-4	1-3	1-3
Free 2 Play	2-4	3-5	1-3

Table 4.3: Personality generation values

The balanced personality was kept with an average value of 3 in all camps, as they are intended to be a "baseline" personality. Competitive and relaxed personalities then counter-act each other, with average values being higher in one where the other is lower.

4.3.3 Simulation

The simulation is started in the Game Management once the user has pressed the corresponding option in the GUI. For the Simulation to start, Players and Game Elements must have been generated already.

Once an iteration of the simulation is started, previous simulation data is removed from the database. The simulation includes quite a few default values, such as its length and if it does or does not over-write previous simulations. To change these parameters, developers must edit the platform manually, as a suitable GUI for this function was not developed due to time constraints. These changes are, nonetheless, very simple, amounting to changing fixed values in the platform's code.

An iteration of the simulation results in multiple instances of the following objects being generated:

- **Gameplay Moment** - A periodically logged player location. Location-based games tend to keep track of an equivalent for player movement or periodic location, as they are crucial for analysis.
- **Challenge Instance** - An instance of a player attempting, successfully or unsuccessfully, an available challenge.
- **Purchase** - A register of a purchase made by a player
- **Inventory** - Items that a player has attained during their gameplay loop, either by purchase or as a challenge reward

All of these objects contain not only the information that has been described but the timestamp and location coordinates associated with the event. As has been mentioned earlier, players should be able to at least complete the Starter Mission, which should be one of the first "Challenge Instances" they generate. Regarding gameplay moments, playing the game without finding any challenges to interact with still generates gameplay moments. Thus, having a high ratio of gameplay moments to challenge instances indicates a low availability of challenges, or at least of challenges that the player is able to engage with.

The simulation occurs by iterating through each player individually and predicting how they will behave for a pre-set number of days. This structure is part of why multiplayer aspects of the simulation would've had a troublesome implementation, as multiple players would've had to have been simulated at the same time.

The pre-set amount of days is 21, starting on January 1st. This set-up is what was used to generate the dataset used for the analysis that will be displayed in the results, although it can easily be changed by the developers (and was experimented with at different values to perform a variety of tests). The reason for such a short timeframe is the large number of players, which led to overly long simulation stages - three weeks is what was decided upon as a reasonable compromise. Simulation time varies based on the number of players in the database and the number of days that it runs for. With our largest database of 10000 players and 21 days, a full simulation can take up to 20h to run.

Each player's simulation is run by keeping track of some over-arching values while simulating each day. The most important of these values is the player's "base motivation", which is used to determine if the player will play the game or not that day. Base motivation is calculated off of each player's Personality values in a sum of their ClearGoals and SocialInteraction attributes. This value is used for other purposes as well, for example, to decide if a player will invest in an in-game purchase in that day - if they're invested enough in the game, and their other data facilitates it (not already having the desired items, appropriate socio-economic status, etc.), a purchase will be made.

For each day, a local motivation value (referred to in the code as playval) is calculated based on motivation and a mix of Personality stats. Then, that value gets influenced by the day's context, such as weather. Weather can heighten or lower the playval. Sunny weather makes it higher; Cloudy weather lowers it. Each weather's most intense version (very sunny or rainy) augments their respective impact. Stormy weather nullifies it entirely. Another attribute that can alter playval is the day of the week or month - if the specific day is a weekend or a holiday, playval is incremented.

Playval is then used to, on an hourly basis, decide if the player will have their first gameplay session of the day. As kids, teens/young adults, adults, and the elderly often follow very different schedules; this was represented in this simulation. For example, kids and the elderly are least likely to play at night, young players have a later start and end time than others, and the elderly get up really early. Adults are also not likely to play during work hours but then compensate for it during their free time.

Depending on the time of the day, specific demographic, base motivation, and playval values, a probability is calculated of the player engaging with the game at that time. If the calculation returns a positive, a gameplay session occurs, and the player gets branded as "having already played today". If a player has already played today, they are less likely to play again. Thus, for every hour on this day after the one where they've had their first session, another probability is tested with the player's motivation values to check if they are willing to play again. If it returns true, the process described in the last two paragraphs repeats.

A gameplay session is a self-contained continuous segment of interaction between the player and the game. Each event gets associated with the gameplay session it occurs during, and it is a valuable metric for analysis. In order to measure the player's mindset during it, to simulate decisions during the game session, an additional motivation value is generated based on the base motivation.

A game session begins with a player searching for visible challenges nearby. If a challenge is visible to the player but not doable from the player's current position, the player moves in the challenge's direction, but the currently doable challenges are prioritized. These include every challenge in the active range (that the player can do without moving), that the player hasn't yet completed (except for those that are repeatable, such as Item Stops) and that the player has the capability to attempt given their items. If any are available, the closest is done. If not, the player moves in a random direction, and a gameplay moment is logged. Gameplay moments are logged every time the player moves, ensuring that every session contains a path between gameplay moments that can be traced.

Once a challenge is attempted, checks are made to verify (again) if the event itself is valid -

if the location hasn't changed for the player, if the challenge isn't already completed, and if the player has the ability to complete the challenge. If the challenge is a Conflict, a percentage change associated with it is run. If the player wins, the reward items are added to their inventory. If the player still requires some items to finish the mission, and both their motivation and SocioEconomic status are sufficient to justify the purchase of those items, then the player can purchase a missing item that will allow them to complete the challenge. Regardless of the result, the player's required items are spent, the challenge instance is registered, and both the base motivation and the current session motivation are affected (positively if the challenge is completed successfully, negatively if not).

A session ends when a player's session motivation reaches 0 (essentially, when they decide to stop playing). Each gameplay moment that passes that motivation decreases, and the rate at which it decreases gets higher the longer the gameplay session goes on, requiring an increasingly higher number of positive interactions from the game (successfully completed challenges) for the player to want to keep playing. This is also dependent on Personality values, of course, as different Personalities "get tired" of the game at different rates.

Once a session is done, the base motivation value for that player gets updated depending on its results, and the hours of that day keep being simulated as described above. Eventually, all of the days that were planned for are done, and the simulation moves on to the next player. It is not rare that, like in real life, some players will have bad early experiences and see their motivation to play the game drop to 0, meaning they end up giving up on the game entirely. To further expand upon this phenomenon but allow for players that want to give the game a second chance or that regain their curiosity after some time, a small random value (again dependent on the player's Personality attributes) can slowly raise the player's base motivation back to levels where they will attempt to engage in play sessions. This is, however, a lengthy process.

Once the simulation is done, the gameplay moments, challenge instances, updated inventories, and purchase records get saved in the Game Management class and in the Database.

4.4 Analysis

Although it is one of three main focuses of the project, the data analysis components are the critical motivation behind the platform's development and the execution of the main concepts explored in this dissertation.

Ideally, in order for these ideas to be expanded to their fullest, analytics functions would be the only module, allowing for more time and effort to be dedicated to them. Instead, and due to the constraints already mentioned in past sections regarding obtaining reliable data, the analysis sections of this work were developed over a two-month timespan, with much of that first month overlapping with the development of the Simulation system.

The machine learning analysis's implementation occurred in the last month of development, with its creation and testing happening concurrently with final touches on all the other systems.

Due to all of this, the work developed in this section is not as complete, complex, or detailed as it was idealized, though all of the main features have been implemented.

The struggles and questions left unanswered of the development process are expanded upon in Section 4.5.

4.4.1 Surface Analysis

This subsection aims to cover the components of the project that mimick a traditional data analysis platform. Each of these components is quite simple but serves an important role in providing a full view to a developer of the data and the conclusions that can be drawn from it.

The term "surface analysis" is meant to describe the methods used for the analysis rather than the depth or value of the conclusions obtained. As this subsection will describe an aggregate of techniques that were used, this name was chosen to differentiate it from the other types of analysis, which are more complex or innovative in their approach. As they were implemented, all of the methods labeled under "surface analysis" are simple calculations that would be present in any analysis platform. The exception to this is "machine learning with filtering", which is not technically a "surface analysis component" and thus will be described in full in Subsection 4.4.3.

This module can be interacted with by choosing "Analysis" on the main menu after having Created a database and run a Simulation. The options within it are:

- **Analysis** - Calculation of averages, KPIs, and detection of outliers when it comes to Challenges and their activity. This analysis is written on the menu itself.
- **Graphs** - Tools that allow the user to generate graphs (bar, pie charts, among others)
- **Filter Analysis** - Running the same analysis as before, but with filters chosen by the user.
- **Machine Learning with Filtering** - An early version of the Machine Learning analysis. By applying filters to the database, predictions of what best conditions would suit players are generated. Then, a small analysis of those conditions is made.

The first item on the list, "Analysis", is the most developed element of the "surface analysis" and its biggest contribution to the platform as a whole. Once activated, the Analysis module receives the datasets from the GameManagement and runs various processes on them. Firstly, it calculates the most and least successfully completed challenges by measuring completed attempts and subtracting failed attempts from them. Using this metric, the three best and worst challenges are found, and an average is calculated and displayed. After that, relevant key performance indicators (KPI) are calculated for every player in the game. The KPIs chosen are:

- **Challenges** - Total number of challenge instances associated with that player, representative of every challenge they interacted with.
- **Gameplay Moments** - Total number of gameplay moments associated with that player, which is the most basic unit of player interaction with the game, representing the duration of their sessions.

- **Challenge Success Rate** - The success rate for challenge instances attempted by a specific player

- **Inventory Size** - Number of items in the player's ownership at the end of the simulation (of note that it is not necessarily a uniquely positive statistic, as interacting with some challenges spends items and endless hoarding is not a desirable player behavior)

- **Purchases Made** - Number of purchases made by the player

- **Lifetime Value** - The sum of the total value of every purchased item by a specific player.

- **Last Log In** - How many days before the end of the simulation the player last logged in.

- **Sessions** - Number of distinct sessions the player played.

These KPIs are averaged so that an idea of the game's overall success and balance status is conveyed.

To help demographic analysis, percentages of the revenue distribution by demographic are calculated. The lifetime value of each player gets added to their own respective demographic indicators, and then percentages are shown to the user.

After that, the player list is shown with each player's KPI, allowing for the developers to parse individual cases and manually diagnose if any undesirable behavior is occurring. Averages alone don't provide a complete picture, after all. They can be the result of unhealthy extremes on opposite sides, and one player's experience might be telling of over-arching problems with the game that go undetected in other methods.

Finally, all of this information gets displayed ot the user in the menu's textbox, following this format:

Average Challenges Engaged With = 8.955
Average Gameplay Moments = 36.26
Average Distance Walked = 44.39256608988927
Average Challenge Success Rate = 0.348
Average Inventory Size = 8.955
Average Purchases Made = 1.495
Average Lifetime Value = 2.903
Average Last Log In = 1.464
Average Sessions Per Player = 13.595
Average Conversion Rate (Install to Purchase) = 0.267
Three Day Retention = 0.791
The following is the distribution of income by demographic:
Male: 1562 0.5380640716500172
Female: 1341 0.46193592834998276
Kid: 122 0.04202549087151223
Youth: 361 0.12435411643127799
Adult: 1894 0.6524285222183948
Elderly: 526 0.18119187047881502
Total: 2903
The average of challenge completions per challenge is 9.58779443254818.
The three best challenges are:
PokeStop Metro at 41.1195255,-8.6061806 with 52 successful completions
PokeStop Metro at 41.1484882,-8.6475818 with 52 successful completions.
PokeStop Metro at 41.1243381,-8.6341354 with 49 successful completions
The three worst challenges are:
PokeStop Metro at 41.1626367,-8.5360712 with 1 successful completions
PokeStop Metro at 41.1767422,-8.5439788 with 1 successful completions.
PokeStop Metro at 41.1600642,-8.529575 with 1 successful completions.
NOTE: All attempts are summed successful completions count as +1 while failures count as -1.
Player Pasquale Morrison engaged with 0 challenges, has 62 recorded play moments, a 0 challenge success rate in a total of 21 play sessions. They have purchased 0 items, having spent a total of 0€ and having an inventory with 0 items. They last logged in 0.0 days ago.
Player Michael Furlow engaged with 0 challenges, has 0 recorded play moments, a 0 challenge success rate in a total of 0 play sessions. They have purchased 9 items, having spent a total of 18€ and having an inventory with 0 items. They last logged in never ago.
Player Beverly Via engaged with 0 challenges, has 52 recorded play moments, a 0 challenge success rate in a total of 19 play sessions. They have purchased 19 items, having spent a total of 57€ and having an inventory with 0 items. They last logged in 0.0 days ago.
(list continues)

The graphs functionality covers the use of plotting capabilities to draw a variety of visual representations for data in the form of pie charts, bar graphs, and line charts. These diagrams include filtering of demographic categories and of which indicators to measure, allowing for users to get a full and incredibly detailed statistical perspective on the datasets. They are drawn using Python's Matplotlib.

During the implementation process, all three chart forms were coded and tested, along with the way these would handle some of the demographic characteristics. However, the data pre-processing required to handle all possible combinations of user choice, as, given the different meanings and readings in the KPIs and demographic divisions (age, gender, location, socio-economic status), it would not be feasible to abstract them with a dynamic system (or such a thing would require much more development time).

Figure 4.5 displays an example of a bar graph plotted on one of the tested datasets. In it, we can see challenges attempted by age demographic, with an obvious disparity against young people (ages 15-24).

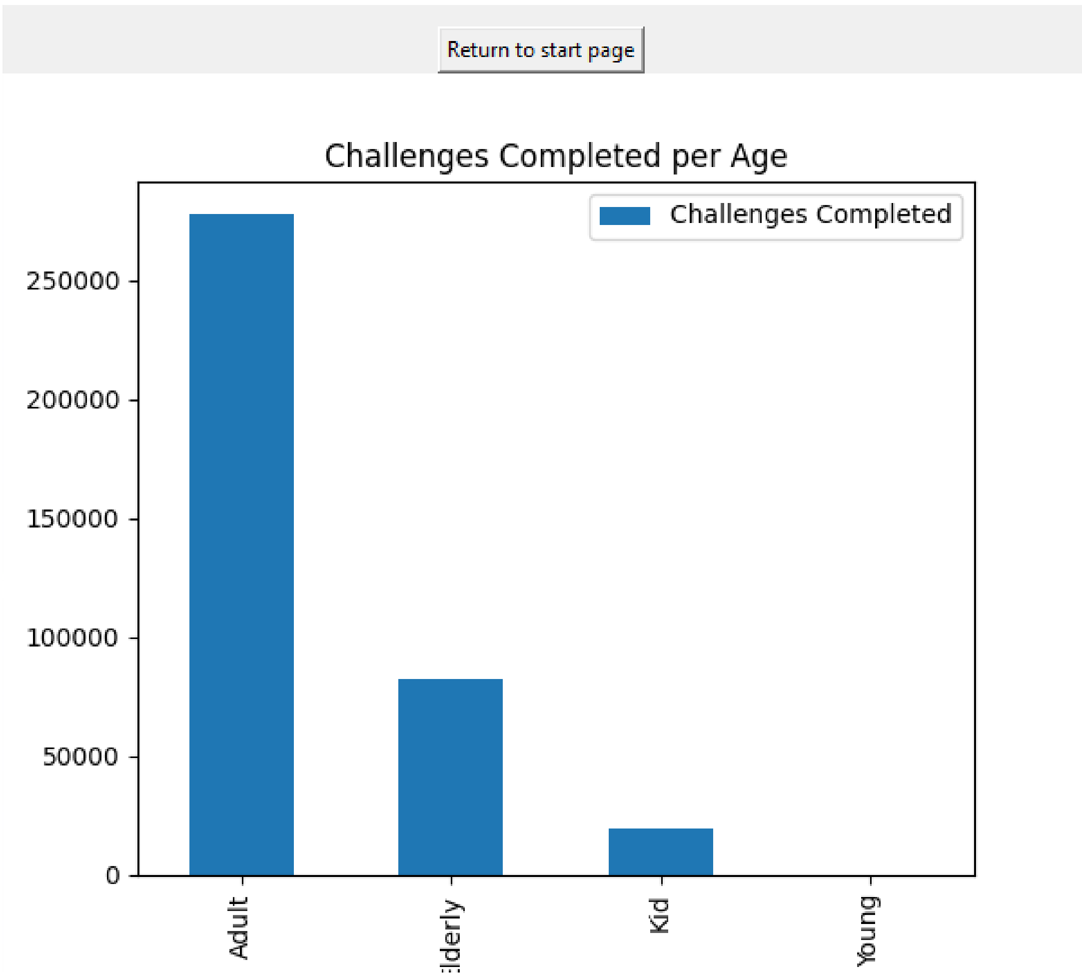

Figure 4.5: A bar graph example

Due to the development time required to fully implement these features to take in player choice, which would cut into the already diminished time allocated for the implementation of the analysis components, the decision was made to only implement a simple example to display how this feature would function in a finished platform.

A similar issue occurred with "filtered analysis". Although analysis with the demographics and/or KPIs being filtered was tested, it had some implementation issues and ended up either being extremely similar to the normal analysis or requiring a very substantial amount of work to make a worthwhile feature. Thus, even though it was tested and functional for specific cases, a fully functional version is not available in the finalized prototype.

Another issue that is important to mention is how both the graphs and the filter analysis components, while important for a comprehensive data analysis platform, are not particularly innovative or unique. Despite them being expected of a commercial product aiming to fulfill the needs that this dissertation aims to answer, a complete implementation of these features would be too time-consuming. Instead, this time was dedicated to the more innovative features that are the bigger focus of this thesis (and which are presented in the following subsections), while these components are left as examples of what the finished product would be like, considering our platform as only a prototype.

As mentioned previously, the "machine learning with filtering" option is not a "surface method". A prototype version of the comprehensive machine learning analysis, it is not considered a "surface component" and is placed in this menu for usability reasons, as it was initially created for machine learning tests but is still on its own a useful tool. Its inner working and implementation will be expanded upon in subsection 4.4.3.

4.4.2 Interactive Maps

The interactive maps feature is used by choosing that option on the main menu, after which (once a lengthy loading time is passed) a browser window opens with an OpenStreetMaps map with the relevant information drawn on top of it.

The implementation of the interactive maps features relies on the Folium API [1]. Initially, map images were taken, and the data was plotted in relation to the coordinates of the images' edges. Later a more dynamic approach was found with Folium. The Folium API allows programmers to generate custom interactive real-world maps populated with various unique objects or schemes. This API relies on OpenStreetMaps for the maps themselves, contributing itself with other graphical features that can then be applied on top of the maps.

The two main "graphical features" that are used for our platform's interactive maps are heatmaps and map markers. Heatmaps take a group of nodes and then determine the map color depending on the volume of nodes in each area, while the map markers are objects the user can click on with names and characteristics that get placed on the dynamic maps.

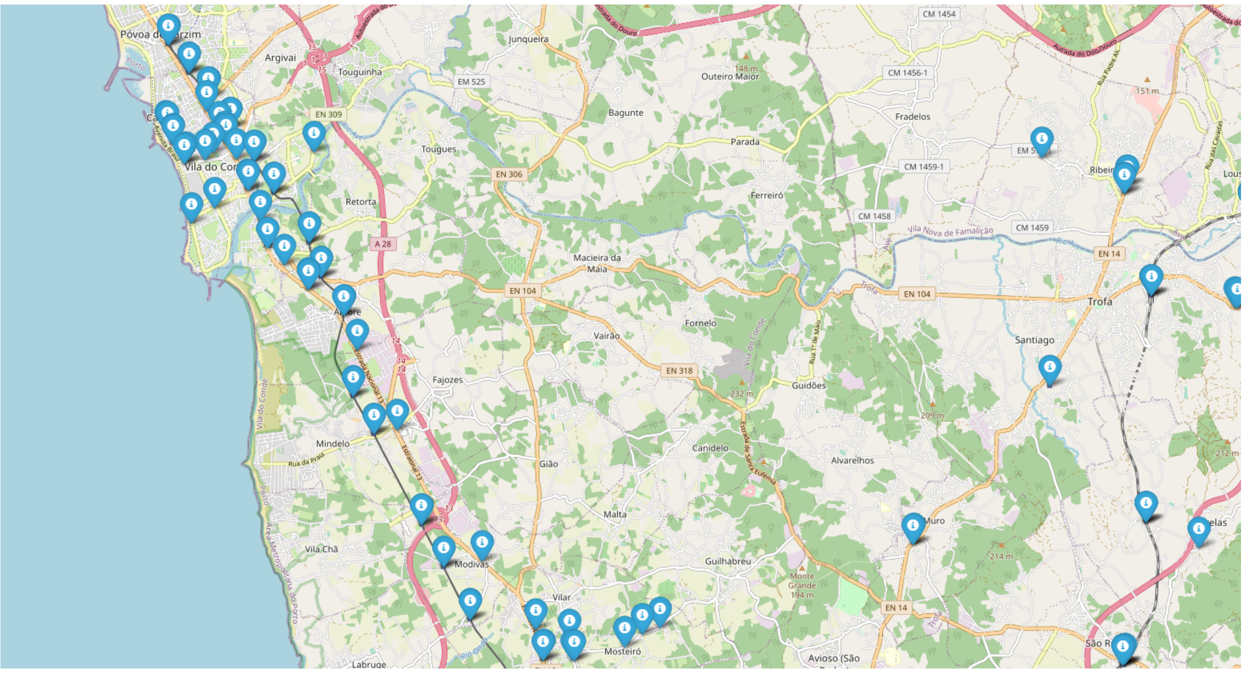

Figure 4.6: An instance of an interactive map with markers representing challenges

Along the development of this project, many iterations of the interactive map existing, cataloging different database attributes or characteristics by location, and using a mixture of map markers and a heatmap. Many of these versions of the interactive map plotting are still available in the platform's code and can easily be activated by a developer, changing which line is active in the function calls. This was intended to be implemented as a GUI menu that would allow users to select which function they wanted to be used for each characteristic and for different KPIs to be attributed to either heatmaps or map markers, depending on the user's goals. However, like many other analysis functions, it had to be streamlined due to a lack of time, and a solution was chosen instead that displays the interactive map capabilities. This solution associates map markers with challenge instances, signaling which ones have been completed successfully and which ones haven't, and the heatmap is associated with gameplay moments.

The main reason behind the interest in developing an interactive map is that it is a very fitting way for developers of the pervasive game genre to analyze their own game's data. As a location-based pervasive game heavily relies on its geographic components, an analysis detached from them will always be somewhat lacking, and this is what we aim to fix.

This gives users a more graphic medium to interact with the data, allowing for easy detection of volume imbalances in the data. If there is a large discrepancy in player activity, be that positive or negative, this issue would be easy to parse after a look at an activity heatmap or a cluster of challenge markers.

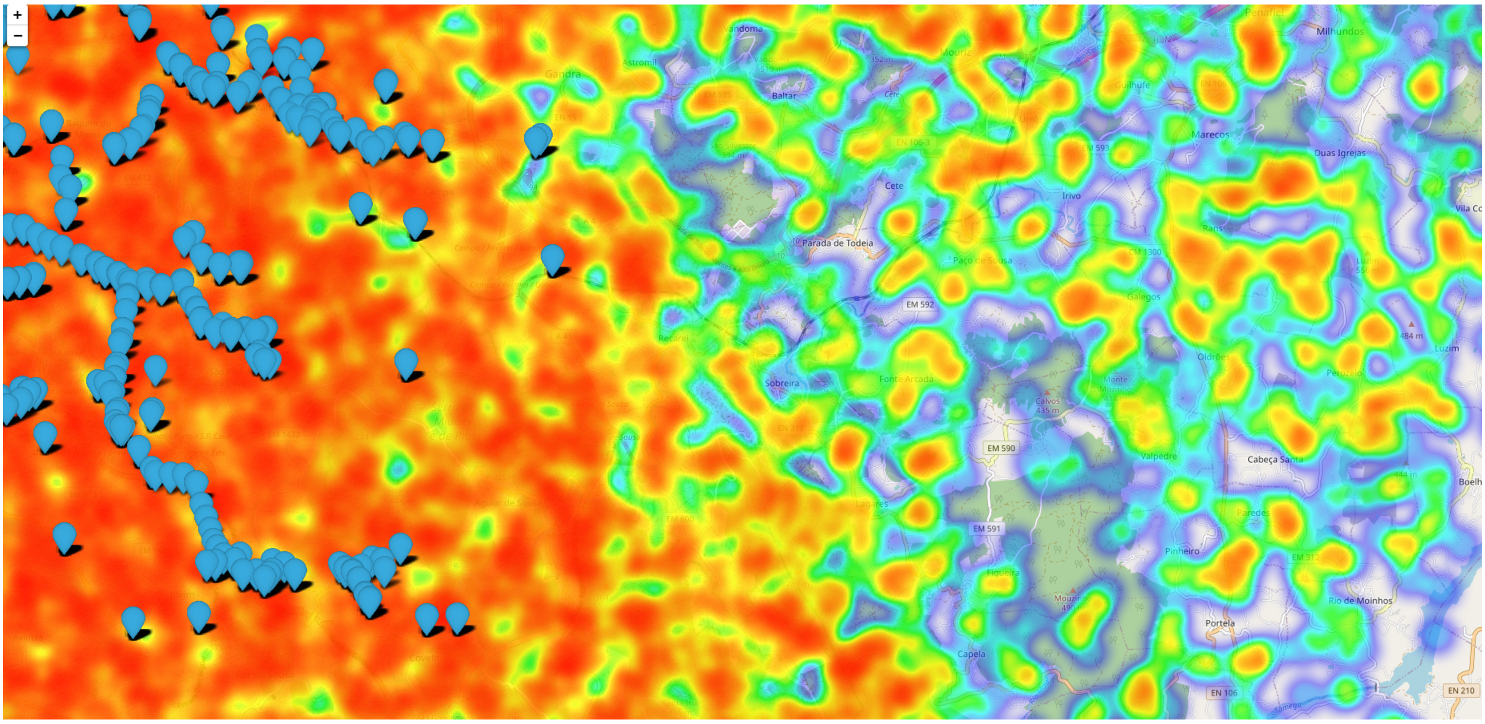

Figure 4.7: A map with heatmap and markers

A major problem present with this feature that we did not manage a workaround for is it has many issues handling large volumes of data. For the heatmap generation, it doesn't cause major issues except for some slight slowdown, but when it comes to map markers, the map will not work with a number that has a large order of magnitude. Due to this, even though it is not the ideal solution, the default behavior of the map in the platform is to generate a heatmap of gameplay moments only, as including map markers would lead to a very poor or even impossible experience. As we can see in figure 4.7, visually, this can also very easily lead to a slightly cluttered display.

If using map markers, it is important to keep the volumes of data low or filter the data used to ensure the few markers that stay on the map are relevant. For reference, image 4.8 is an example of a heatmap display on its own, which is what will be available in the prototype platform by default:

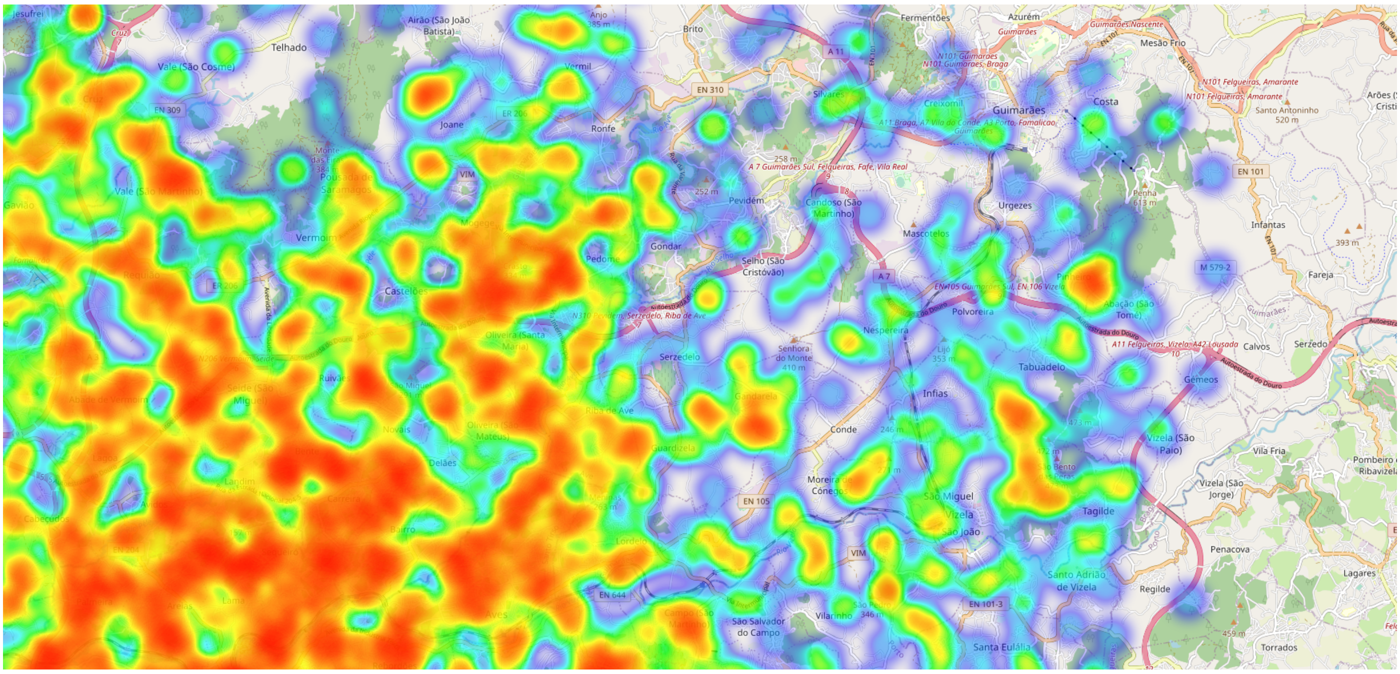

Figure 4.8: A heatmap of player activity

4.4.3 Comprehensive Machine Learning Analysis

To elaborate on the work developed under the "comprehensive machine learning analysis" label, we must first expand upon the "Machine Learning with Filtering" feature that was present in the Surface Analysis menu. This feature, initially an experimental version of the machine learning capabilities, ended up serving as a building block to what we'd then call the comprehensive machine learning analysis, which is depicted in figure 4.10.

This machine learning tool, created using a random forest multi-output model, takes player data filtered by the user and uses it to predict the demographics of a positive or negative scenario, depending on what is chosen. The random forest multi-output is the technique used in the preliminary work, in which the authors used it successfully to predict player data in a similar way to what is done in this project. Random Forest is an algorithm that combines many decision trees, limiting overfitting and errors caused by bias. It is an extremely robust technique that, among the other machine learning techniques tested in experiments made during the development (support vector machines, k-nearest neighbor models, XGBoost), led to the best results. The metrics used to measure the results of these experiments were Mean Absolute Error, Mean Squared Error, and Root Mean Squared Error, which are standard and reliable metrics to assess the precision of machine learning models.

In this implementation, there are two models used, a classification model and a regression model, due to the nature of the data being analyzed. It would be less accurate to make predictions based on age and gender categories - as they were implemented in the rest of this project - with a regression model, as for gender there are only two options (male and female), and for age there are four options (child, youth, adult and elderly). For them, a random forest classification model was chosen instead. This strategy was chosen due to the way data was collected and used to generate the player profiles, as other approaches could evaluate both of these variables as intervals of values. The socioeconomic status and location coordinates, which are here implemented as value intervals, are left up to a random forest regression model.

The models are trained on the data chosen, with 1/4 of it being instead used for testing, with the testing data then being used for predictions regarding demographic inequalities. The idea behind this method is that if you only filter by extremely positive or negative KPIs and the machine learning model detects some underlying demographic trends that would otherwise be unnoticed by a regular statistic analysis, this would detectable by an analysis of the predictions. The variables predicted (labels) by this tool are location coordinates (latitude and longitude) and the player's demographics (age group, gender, and socioeconomic status), while the features are the player's key performance indicators.

The responses by this tool will be based only on demographic statistics, detecting scenarios where trends or distributions become worrying. Once the predictions by the representation of the dominant demographic group for the three available demographic characteristics are calculated, and then the ratio at which they are superior to other groups is compared to the accurate ratio calculated directly in the player database. The goal is not to aim for equity in demographic data but for accurate representation.

The "Machine Learning with filtering" tools in the analysis menu allows for selecting one of three major KPIs that we concluded were the most representative of the player experience or that at least are of incredible significance from an analysis perspective. The indicators in question are "Session amount", "Lifetime value," and "Challenges done". We also looked to have the least overlap possible between these indicators, which is why the number of gameplay moments or purchases isn't used - the other indicators already cover similar topics in more conclusive ways.

The user then selects a value and Max or Min - this represents the cut-off for the analysis. If the user selects Min, it will filter the dataset by only including users whose value for that specific KPI is below the number inputted in the field, while the opposite will happen if the user selects Max. The value must be picked carefully, as choosing a value that isn't fitting of the KPI and cut-off in question might lead to either a useless analysis as it includes a large majority of the player-base or to a misleading one due to having too small of a dataset to create a reliable machine learning model off of.

Figure 4.9 displays what the GUI looks like, along with the results from a basic execution of the tool:

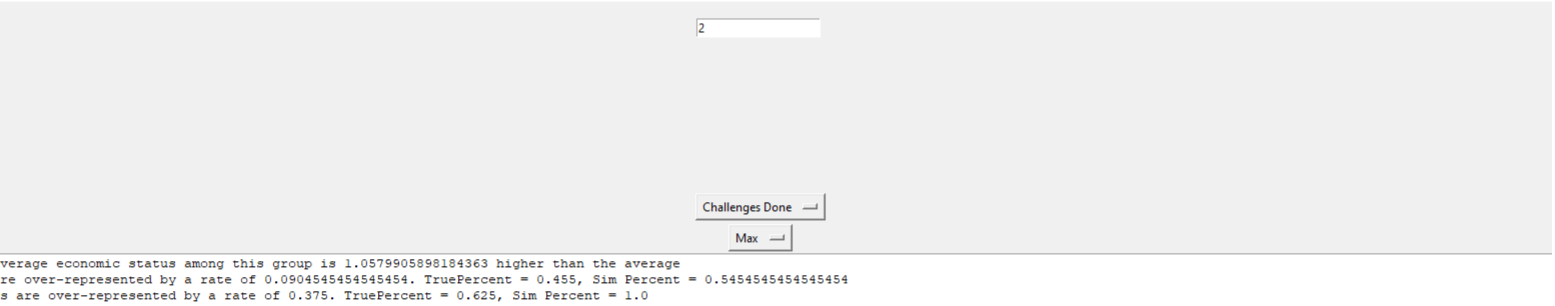

Figure 4.9: The machine learning with filtering tool

The results are obtained by using the filtered data as a dataset for Random Forest models, using both Random Forest multi-output regression and Random Forest classification for accurate results with classification variables (which age and gender are, in this case). The inclusion of the Random Forest classification happened during development as originally, only Random Forest regression was planned. However, the results for binary characteristics were never read as such by the model, who returned decimal values in all fields. To obtain more accurate results, those features were split for a classification model while the others were kept in the regressor.

This method is used as a building block for the comprehensive analysis. Figure 4.10 depicts the entire comprehensive machine learning pipeline, which will be described in the rest of this section.

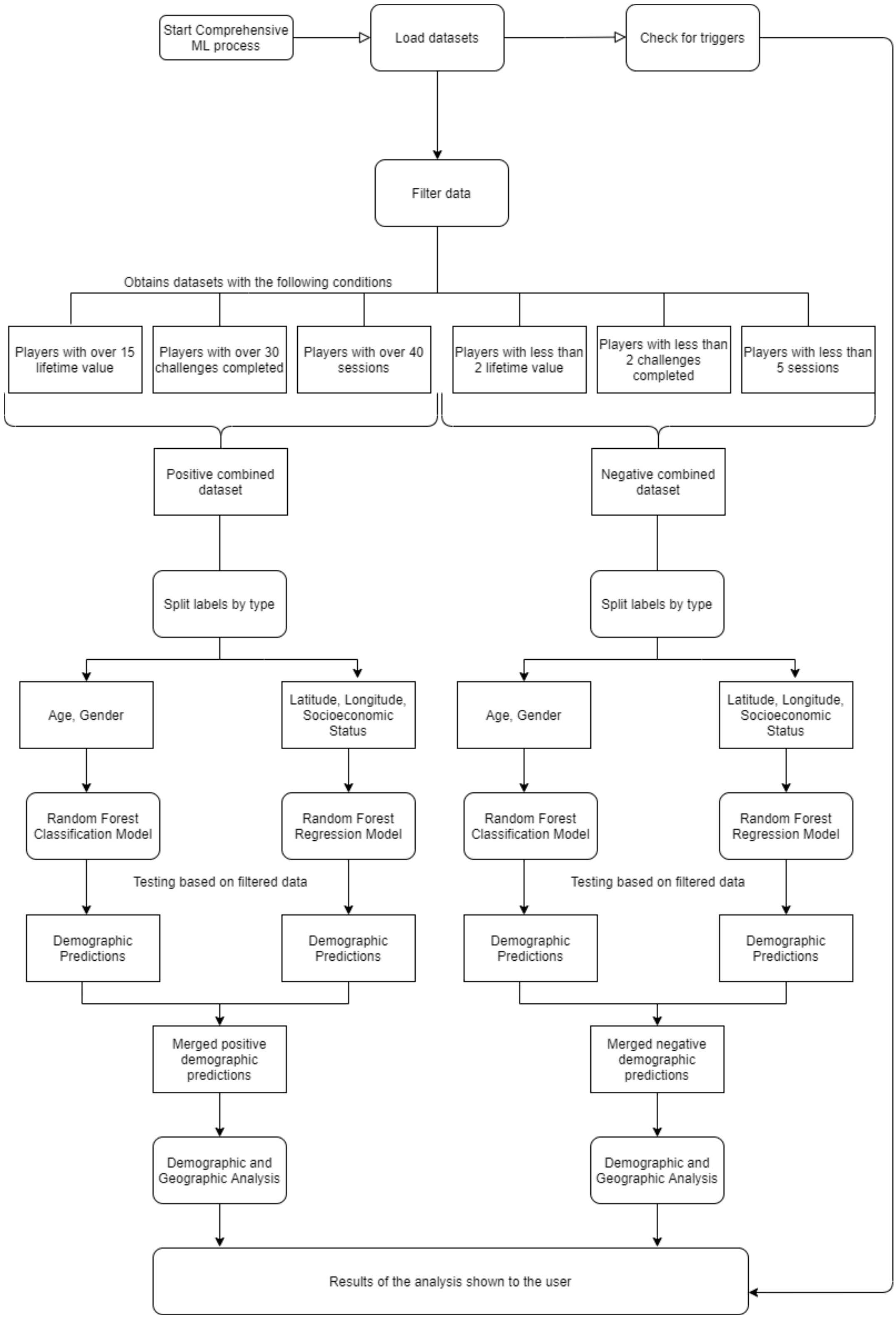

Figure 4.10: The comprehensive machine learning analysis pipeline

Instead of filtering only one KPI value and cutting off either above or below a certain value, all three indicators are used with both negative and positive analysis, combining six different datasets. The cut-off values for all of them filter around 80% of players off of the dataset, maintaining either the top or bottom 20% (depending on if it's a negative or positive filter). The positive datasets are merged, creating a dataset of "ideal" experiences, and the same happens to the negative datasets that represent the opposite reality. The demographics of the predictions made by the models trained on these datasets will then be analyzed to determine which demographics are mostly advantaged and disadvantaged.

Two new forms of analysis are also integrated with this comprehensive analysis. The first one is a trigger system, which aims to detect any other ethical or unjust statistical occurrence present in the game's data, but that is not being considered by mere statistical analysis. The trigger system functions by adding manual conditions that get triggered whenever a possible unethical situation is at hand but that otherwise don't output information to minimize visual noise. An example of a trigger that is in place is assessing in-game spending by the children, the youngest demographic, and verifying if a statistically significant percentage of them has a lifetime value that is higher than the average. If so, the system posts a message warning the user.

The second system in place to verify injustices is one aimed at pinpointing geographic advantages. Although the interactive maps are very useful, they might not be clear enough for the game developers to detect when a specific area is advantaged compared to its neighbors. To address this, the area considered by the game is split into 25 equal quadrants, and the predicted players get fit into them. Then, an analysis is made to calculate the average player per quadrant, and the fairness of the distribution is calculated, with any quadrants that have an above-average player amount (which indicates higher player concentration, given the equal size) being listed. The ratio they have to the average and the other quadrants are also given, displaying if the higher concentration of players in that sector is problematic or if it's nothing to cause concern.

The full analysis returns results following this format:

EARLY ANALYSIS SHOWS...:
Warning: 0.37209302325581395 of children seem to have a lifetime spending over the average.
POSITIVE RESULTS:
The Elderly are over-represented by a rate of 0.08971428571428569. TruePercent = 0.196, Sim Percent = 0.2857142857142857
The average player count per quadrant is 0.12
The following quadrants have an above average concentration of players:
The quadrant from [41.13912,-8.617379999999999] to [41.17404,-8.572059999999999] has a population 16.666666666666668 above the average.
The quadrant from [41.17404,-8.572059999999999] to [41.20896,-8.52674] has a population 8.333333333333334 above the average.
NEGATIVE RESULTS:
The average economic status among this group is 0.5208381784634794 higher than the average
WARNING!! Women are over-represented by a rate of 0.2891346153846154. TruePercent = 0.545, Sim Percent = 0.8341346153846154
Adults are over-represented by a rate of 0.009615384615384581. TruePercent = 0.625, Sim Percent = 0.6153846153846154
The average player count per quadrant is 0.68
The following quadrants have an above average concentration of players:
The quadrant from [41.17404,-8.572059999999999] to [41.20896,-8.52674] has a population 7.352941176470588 above the average.
The quadrant from [41.24388,-8.48142] to [41.2788,-8.4361] has a population 7.352941176470588 above the average.
The quadrant from [41.13912,-8.617379999999999] to [41.17404,-8.572059999999999] has a population 5.88235294117647 above the average.
The quadrant from [41.1042,-8.6627] to [41.13912,-8.617379999999999] has a population 2.941176470588235 above the average.
The quadrant from [41.20896,-8.52674] to [41.24388,-8.48142] has a population 1.4705882352941175 above the average.

As is visible in this text box, it is very frequent for demographics not to fit exactly their "real" distribution, but only once it reaches a concerning percentage (which depends on the variable and demographic being analyzed) is it considered a cause for alarm by the system (which writes Warning!) behind it. Furthermore, the trigger system worked as expected, and an unusually high percentage of children seem to have spent more money on the game than the average player. This analysis lends itself very easily for pinpointing issues and suggesting to developers what issues to look into and fix.

4.5 Discussion

The goal of this section is to disclose the author's insights regarding the project's development, going over what went right, wrong, what was planned bet left incomplete, what could've been done better, the main constraints and issues that came up, ethical questions about the techniques used, and any other reflection that is relevant to the implementation.

The development of this dissertation lasted a total of six months, beginning in January 2021 and ending in June of the same year. This, of course, delineates only the focus on the project itself, as there was preliminary work done in the months leading up to the launch of development. The project started with an initial research and planning stage that took up the months of January and early February, leading to the writing of the first three chapters of this dissertation.

Chapter three contained an outline for the development of the platform, but as was described in Section 4.2, a lot of changes had to be made due to implementation constraints. The last two weeks of February were dedicated to planning out these changes, drawing new schematics, fully outlining which technologies would be used in practice and which ones would be removed from the current plan. This was when the overall strategy for the project was decided, including foregoing the data collecting components and replacing them with data generation elements. Beginning in March, the start of the coding began.

Figure 4.11 is a rough timeline of the project's development.

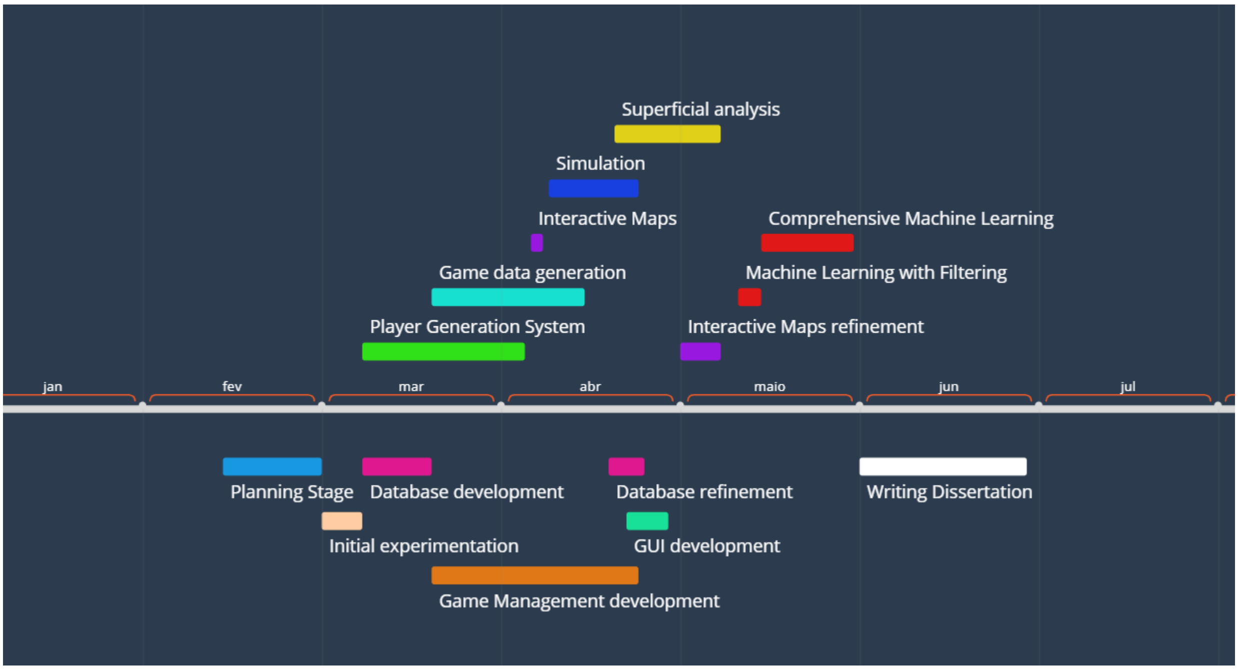

Figure 4.11: Timeline of the project development

As is to be expected, a lot of components were built in parallel to each other. The reason for this is that these systems can only be tested and be deemed to be working as intended when interacting with others - for example, it's hard to detect if the game data generation system is being well implemented if the is no simulation to ensure that the challenges are working as intended.

The process of implementing and developing the platform itself started by testing out various APIs collected during the research stage, assessing their usefulness and how they would be integrated. After that, the APIs started being tested in generating players and challenges in rudimentary ways, and the development of those components continued from there.

The development of the full player and gameplay component generation system took around a month. The reason for this is that along with developing those systems specifically, which have a reasonable degree of complexity, the rest of the platform object organization components also had to have been developed. The main class, GameManagement, and all of the models, database interactions, and data manipulation functions were developed during this time.

The development of the simulation and GUI started in April and took up the first weeks of that month, along with the start of the analysis methods. This meant that the month of May was dedicated to the development of the Interactive Maps, the Machine Learning methods, the trigger system, and refining the earlier analysis methods.

As the month of June was intended to be solely used for the writing of this dissertation (chapters four through six along with updating and adding new content to the first three chapters), it means that the focus of this project was developed on a very limited span of time. Due to all of the preparatory work and extra components that ended up being required for the platform to function as intended, the main contributions this project had in mind suffered from heavy time constraints and ended up not being developed to the extent that would be desired.

Some ethical issues came up during development regarding the use of APIs and the simulation of player data. One of the main reasons for the player data to have had to been simulated in the first place was due to not having genuine player data from pervasive games to use, in part because of the ethical issues with using player's personal data without their consent.

Alongside this issue, which would have to be addressed in any future iterations of this work that use genuine player data, there is also the concern of making unethical prescriptive claims regarding the behaviors of certain demographics. For example, differentiating player habits by gender in the simulation or in the player profiling (for example, giving different odds for personality stats or using different values for simulation thresholds) would not only be unethical and against the goals of this dissertation (which aims to combat injustice) but also lead to bad data.

Building a model that takes into account the player habits that different genders have on average, if those contrasts exist to a significant extent, would require a lot more research and is outside of the scope of this project. Due to this, the gender attribute of the player profiling ends up only being representative of gender distribution in the local population and how they might be affected by the game, rather than predicting their behavior based on that demographic.

Another issue that must be mentioned about the development of this project is the lack of time for concepts to be fully realized. Many of the intended parts of the project, in all of its stages, had to be cut due to not having enough time to implement them in a functional fashion. Some of these are concepts that are part of the diagrams and critical in the planning stages, such as the in-game currency, factions, and multiplayer elements. Others were implemented into the project but not fully, and thus they did not factor into the following stages and did not get to be a part of either the implementation or results chapter (as they do not reflect in any function of the final work). These include implementing a commute system for the simulation, taking note of player connections to understand which types of players more often interact, day-by-day analysis to understand what weather conditions, holidays, and days of the week are more popular, and introducing simulated in-game ads.

One of the main motivators for the development of this project was the issues of ethnic injustice present in early Pokémon Go. It is unfortunate then that, due to Portugal not collecting data regarding ethnicity or race in its census, we were unable to represent the ethnic distribution accurately in the geographic areas covered - which is why it was not done at all, focusing on age, location, socioeconomic status, and gender instead.

Finally, it is important to mention that much of the work developed here in the Machine Learning analysis components was heavily inspired by the preliminary work described in section 3.2. Due to the heavy constraints that this platform was being developed under and for an assurance that the results would be reliable, systems were chosen that have been tried and tested by the authors previously. This project improves and iterates on the systems built in the preliminary work, using its models in very different ways (such as combining two different models for predictions).

4.6 Summary

In this chapter, the details of the implementation of the data analytics platform were discussed, discussing the features implemented, the programming strategies used, and expanding upon the reasons for the choices that were made and the behind-the-scenes of the development process.

In the Overview, the platform as a whole is shown from a superficial perspective, presenting the main features that were developed along with the overall structure of the implementation. These features include the graphical user interface, the player profiling/generation, the creation of fake game data, the gameplay simulation, and the multiple analysis types - the "surface analysis" via various stats, indexes, and indicators, the interactive maps, and finally the machine learning analysis.

As there are significant changes in comparison with the planned project described in Chapter 3, some new concepts (such as the player profiling) had to be presented in detail in the Overview, and justifications were then given for the decisions made regarding the solutions to the constraints faced during the development. The question of whether to use the region's demographic data or one more resembling the player-base of a pervasive game for the player generation system is somewhat up in the air, with possibly the latter providing results that more closely resemble other popular pervasive games, but the accuracy and reliability of the demographic data used were valued over it.

The data generation section covers two of the project's main features: the creation of reliable and lifelike player profiles and the simulation of the game data (both the game's pre-set game objects and the interactions of the players with them). The resources relied upon for player profiling and the methods to calculate the attributes for the model described earlier are shown. When it comes to game data generation, a lot of "executive decisions" had to have been made in order to obtain a useful and realistic dataset. These decisions are explained, and the entire simulation process is shown in detail.

As the most ambitious and complex module, the analysis section also suffered from the most setbacks due to development issues, which are detailed in this chapter. This module can be split into three subsections - "surface analysis", interactive maps, and machine learning analysis. Surface analysis is a basic collection of stats on the database that provides the user with an overview that they'd be able to obtain in other analytics platforms. Interactive maps display the game data dynamically over an OpenStreetMaps map, allowing the developers to freely interact with it and giving a visual medium for the unique location-based dynamics. Machine learning analysis is the culmination of the project's work and provides the user with conclusions based on predictions made by a Random Forest model that was trained on the game's data, informing the user on both balance and ethics problems that might appear.

Finally, the last section before this one provides a small discussion of the development process as a whole, showcasing some of the trials and tribulations that the evolution of the project underwent. Along with a reflection by the author on the implementation of the overall platform, this section showcases the schedule that features and components were created in, presenting a grander picture of the creative process.

In sum, the implementation of this platform is mostly complete, with a majority of its intended features being implemented and goals reached. However, it is not possible to gauge the success of the methods themselves without accessing the results with care. The following chapter has that goal in mind - displaying what was actually done with the work done here and what concrete judgments can be drawn regarding the methods and data obtained.

Chapter 5

Results Evaluation

Contents

This chapter aims to inspect the results obtained by experiments done with the platform that was developed and expand on those results with a discussion of their meanings and implications towards balance and fairness. Section 5.1 provides a short introduction on the topic, while sections 5.2 and 5.3 detail the results obtained in the Data Generation and Analysis modules, respectively. Finally, section 5.4 contains a discussion of the information provided in the previous sections.

5.1 Introduction

In this chapter, we will explore the results obtained from the work developed in the previous chapter, describing the experiences made and discussing how to interpret those results and what approaches to take. The goals and questions being dealt with in this project can be taken from many angles, and it is our job in this chapter to also define which valid approaches exist and decide on the best one for each scenario.

As it pertains to the data generation module and its respective components, the most important thing to validate their success is how closely the data generated is to data that could be obtained by using real people. However, this can mean different things - do we compare the simulated data to real-world demographics, or only to demographics of the users of the average pervasive game?

How important is achieving a balance between these demographics, sacrificing some accuracy to the real world to obtain more useful and representative data from the simulation?

When it comes to the analysis, the results we aim to analyze are not only of the efficiency of the analysis methods but also of how closely these concluded that the simulation came to represent real human behavior within the game. Furthermore, as the project is meant to be a platform that game developers will use regularly, some of its components also need to be evaluated for readability and usability.

The experiments made on the platform were produced based on two different databases, both generated using the data generation systems, but that differ from each other in volume. Both databases used the same APIs as sources and had the same parameters, engaging in three week-long simulations - however, one was generated with ten times as many players as the other, leading to much more data. Database 1 contains 1000 players, around 9000 challenge instances, and 36000 play moments. Database 2, on the other hand, is significantly larger, having 10000 players, approximately 10000 challenge instances, and 380000 gameplay moments.

The goal of using two different datasets, despite the lack of differentiation in their creation process outside of the volume, is to test how well the platform behaves at different volumes. We're not only worried about demographic imbalances that can occur but also about statistics and conclusions being radically different, group imbalances appearing significantly more often, and for the platform's usability to suffer (for example, with long processing time).

5.2 Data Generation Results

The results obtained in the data generation were either calculated by the platform itself or by extra functions built for that purpose that provide information on the database. As mentioned previously, the strategies used to validate data will be mostly comparing the data to outside sources for other pervasive games or evaluating it on its own (verifying if a value makes sense in its context). Due to emulating many of its design trends and philosophy in the simulation and to being the most popular pervasive game with, naturally, the largest amount of public stats and APIs to compare our game's stats to, Pokémon Go's publicly available user statistics were used for comparison.

5.2.1 Player Generation

Tables 5.1 and 5.2 provide the player distribution across demographics for both databases.

Demographic	Absolute Number	Percentage
Male	455	45.5%
Female	545	54.5%
Child (5-15)	43	4.3%
Young (15-24)	136	13.6%
Adult (25-65)	625	62.5%
Elderly (65+)	196	19.6%
Total	1000	100%

Table 5.1: Demographic statistics for database 1

Demographic	Absolute Number	Percentage
Male	4539	45.4%
Female	5461	54.6%
Child (5-15)	462	4.6%
Young (15-24)	1126	11.3%
Adult (25-65)	6549	65.5%
Elderly (65+)	1863	18.6%
Total	10000	100%

Table 5.2: Demographic statistics for database 2

Table 5.3 describes the real world demographics for Portugal, covering age and gender.

Demographic	Absolute Number	Percentage
Male	4.859.400	48.6%
Female	5.437.700	54.4%
Child (5-15)	953.689	9.5%
Young (15-24)	1.098.001	11.0%
Adult (25-65)	5.514.241	55.1%
Elderly (65+)	2.295.038	23.0%
Total	10.297,1	100%

Table 5.3: Portugal Demographic Statistics

The sources for this data are Pordata [61] [62], the contemporary Portugal database.

On the other hand, there is also this information regarding the demographics of the Pokémon Go player base, with SurveyMonkey Intelligence [70] being the source:

Demographic	Percentage
Male	53%
Female	47%
13-17	20%
18-29	43%
30-50	30%
50+	8%

Table 5.4: Pokémon Go! Demographic Statistics

This data is not directly comparable to the other tables due to the age ranges being different, but even so, some valuable observations can be obtained from it. It is also relevant to ask the question of if it is more accurate for the simulation to use this data or more general location-centric demographic information. That said, the comparison across age ranges is favorable.

The socioeconomic data is purely a WorldPop comparative value with random variation, so there is not much to verify, and the locations were distributed randomly and uniformly in the area chosen.

Other than demographic data, it is hard to find reliable metrics for the player profiling system. The personality system is based on an established model, but it is somewhat arbitrary, and its results can only be verified by the success of the simulation.

5.2.2 Game Data Generation

The game's data that gets generated before the simulation is composed of direct API requests for reference points. The way popular pervasive games generate gameplay elements is fundamentally different than this simulation, as this relied on two very small reference labels and its own player activity, and pervasive games each use their own method - often relying on cell phone activity data or other procedure unique to the game.

That said, it is at least possible to compare the game data simulated to anecdotal examples of pervasive game data regarding distribution and density, ensuring its validity.

Using images from PogoMap [60], we can compare specific areas in the map to the challenge distribution in the simulation (using the interactive map feature).

We have analyzed two different areas chosen arbitrarily, with one being a city center and the second being a rural area with less activity. Figures 5.1, 5.2, 5.3, 5.4 are how they are represented in the interactive map feature and in PogoMap. They are presented at similar levels of zoom and with the framing in the same coordinates, ensuring a fair comparison.

City Center:

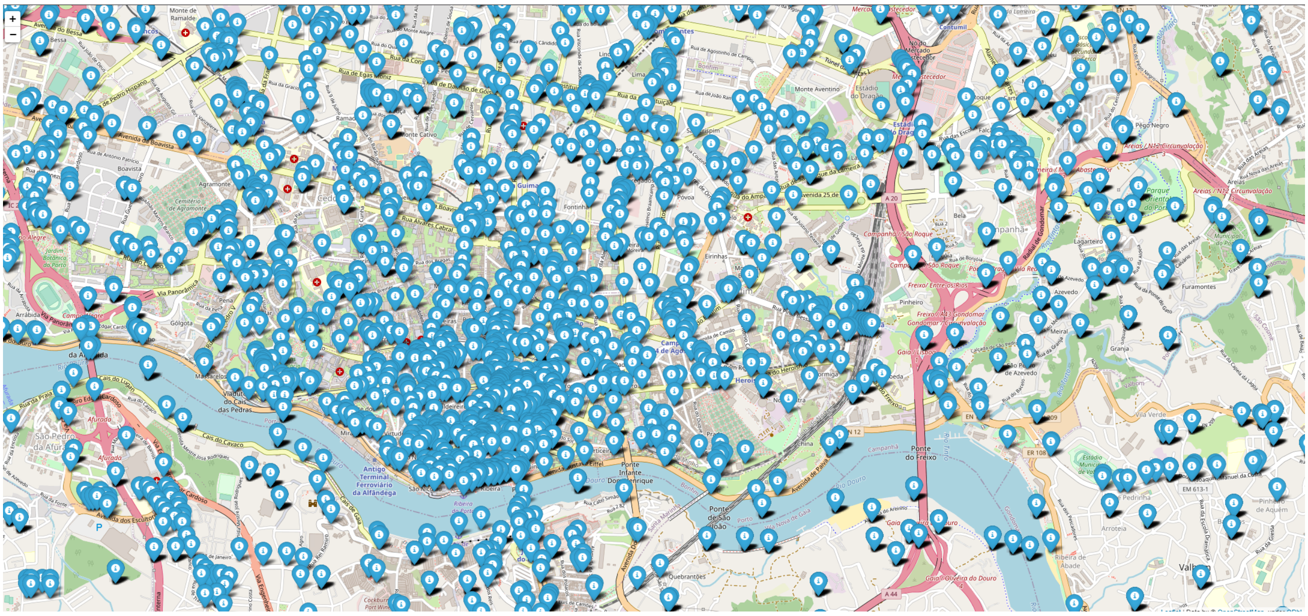

Figure 5.1: A city center in the simulation

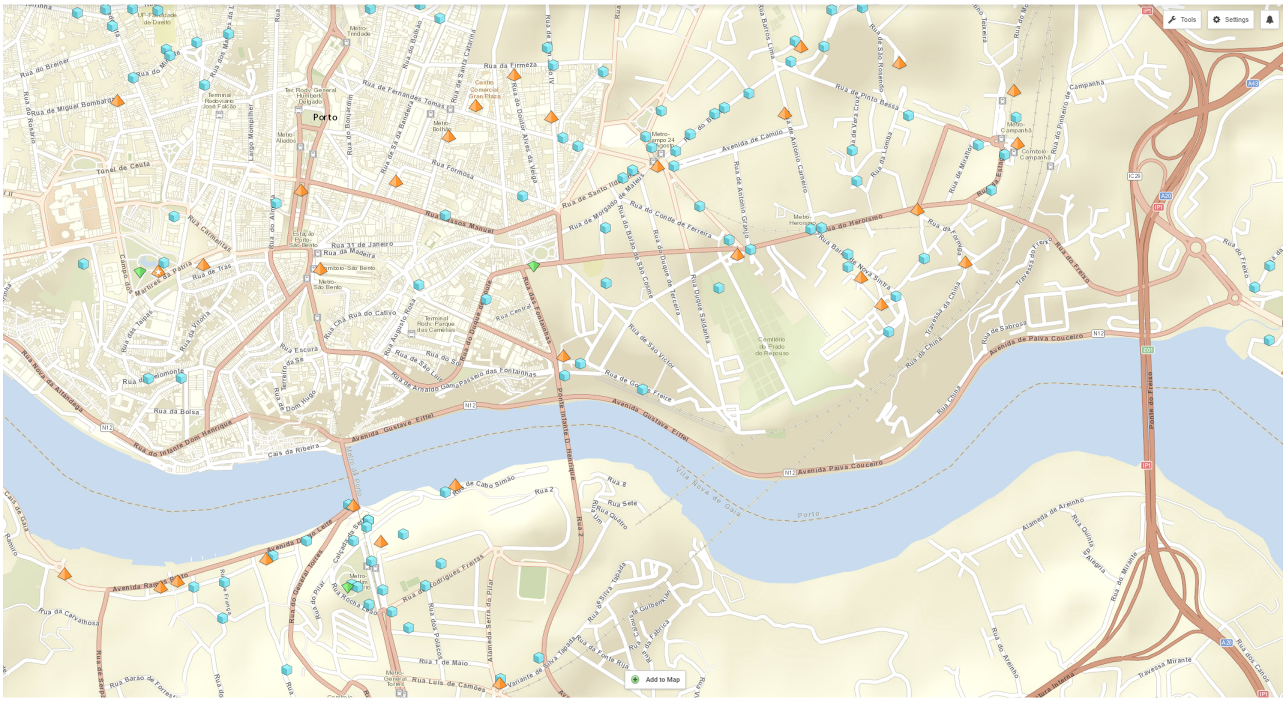

Figure 5.2: The same city center as before but in Pokémon Go

Rural Area:

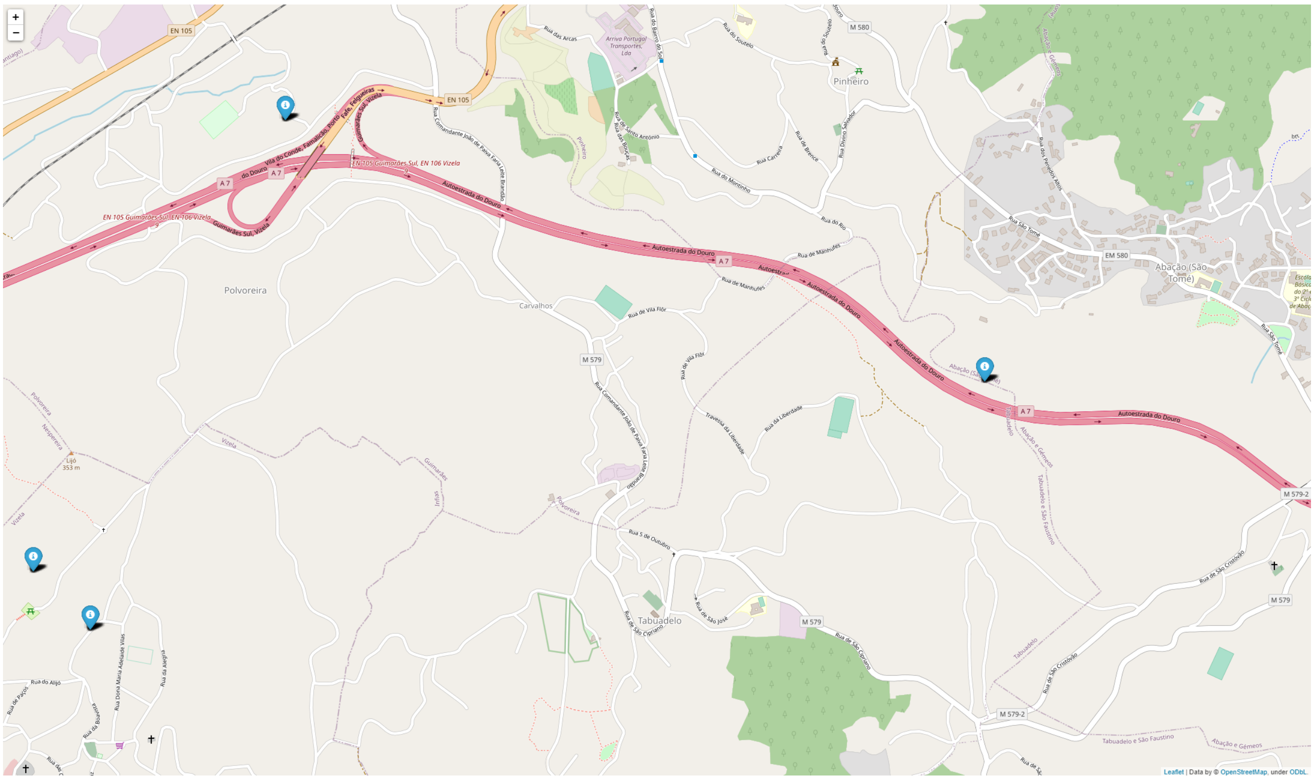

Figure 5.3: A rural area in the simulation

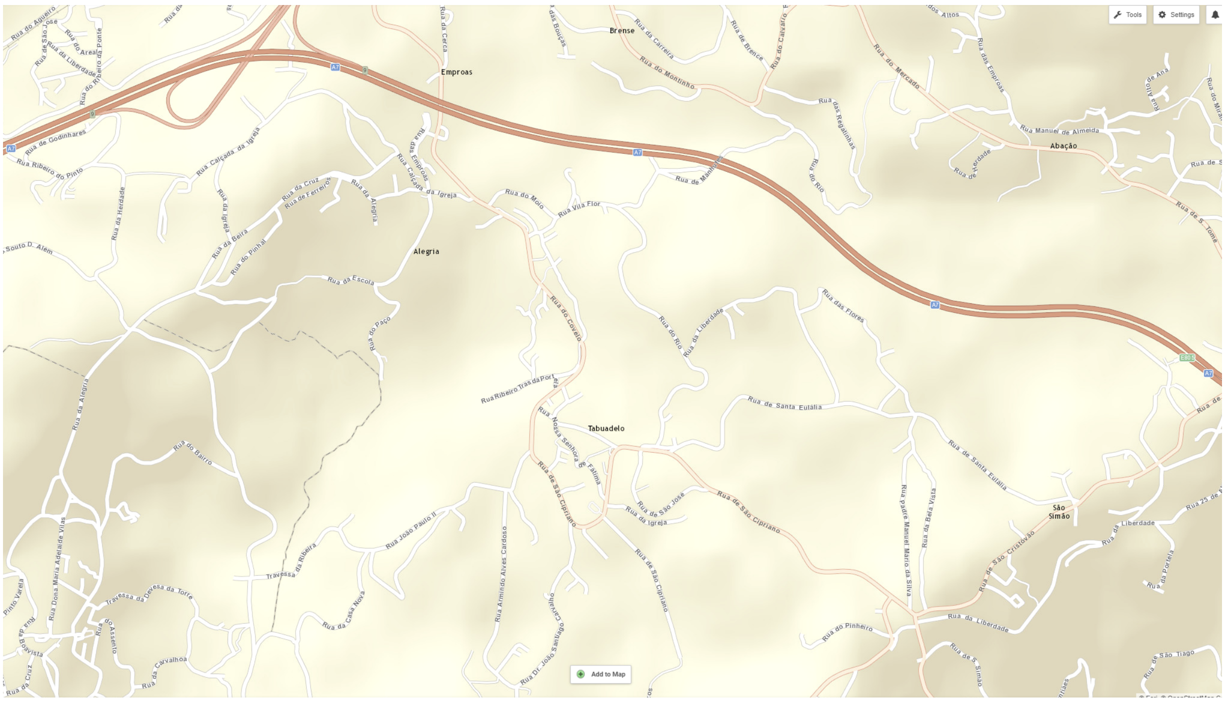

Figure 5.4: The same rural area as before but in Pokémon Go (no challenges visible)

It is clear from these images that challenge density and volume are much higher overall in the simulation. However, the same trend is visible in both: the rural area has significantly fewer challenges than the city center, which is crowded.

As a side note, it is important to mention that the representations in tools such as PogoMap use fan-submitted resources. Although there is an approval process, all of the data in it can't be reliably and independently verified.

5.2.3 Simulation

To analyze the success of the simulation, the best course of action is to do a full analysis using the platform and - taking into account what we know from the previous sections - detect if there is any major unexpected issue in the simulation.

As an important note, all of the numbers displayed in this subsection will be from the simulation ran using the larger database, which contains 10000 players. The reason for this is that it was concluded after thorough experimentation that the larger amount of data, following the same generation process, only lead to more accurate results.

As has been established, the goal of the simulation is only to present a feasible representation of what a pervasive game could look like. So while the validation of our results will include comparisons with data from other pervasive games, the main goal of this section is to assess if the data is "reasonable" - that is, if the values obtained in the analysis are not extremely different from what would be expected or from other results.

The search for "unexpected values" doesn't include explicitly wrong calculations - for example, the analysis claiming that the average challenge success rate is 2.3 or 230% - as these would be considered analysis errors and not simulation ones. A simulation error would be if the challenge success rate was 0, for example, implying that due to some fault in the gameplay interactions, the players were unable to engage with any of the generated challenges.

Table 5.5 is a table of reference data we could find from online sources. Information on pervasive games is, as we've mentioned before, very scarce due to the small number of popular pervasive games and their lack of openness with data. As such, the data from this table is once again about Pokémon Go, obtained from ARInsider [17].

KPI	Value
Average revenue per daily active user	0.25$
Percentage of users that do in-app purchases (iOS)	80%
Three day retention	>60%

Table 5.5: Pokémon Go KPIs

And here are similar KPIs for this simulation:

KPI	Value
Average revenue per daily active user	0.13
Percentage of users that do in-app purchases	26%
Three day retention	79.1%

Table 5.6: Partial Simulation KPIs

The most notable difference in these tables comes from the percentage of users that make in-app purchases. It is possible that this low percentage occurs due to the small duration of the simulation - as the purchasing system intends to represent the player's growing motivation towards making a purchase, it is feasible that many of the players that did not make any would eventually. Compared to the short 21 days that the simulation runs for, the data collected from Pokémon Go considers lifetime engagement with the game, so it is feasible that these numbers don't indicate any issues with the simulation system.

To continue our assessment, let's verify the rest of the simulation's KPIs independently.

KPI	Value
Average Number of Challenges Engaged With	36.26
Average Distance Walked	44.39 km
Average Challenge Success Rate	34.8%
Average Inventory Size	8.955
Average Lifetime Value	2.9€
Average Sessions Per Player	13,595

Table 5.7: Full simulation KPIs

According to a formal study on the physical activity of Pokémon Go players [36], they walk on average 2.4km per day. Considering the simulation ran for 21 days, the possibility of multiple sessions per day by a single player, and the fact that nearly 20% of players did not play more than three days, the distance walked value seems reasonable.

The remaining values seem reasonable as well, with lifetime spending being low again due to the short duration of the experiment and challenge success rate, possibly denoting some of the problems that led to other players giving up on the game.

A short demographic analysis displays that for some reason, players in the 15 to 24 range were extremely disadvantaged when it comes to challenges.

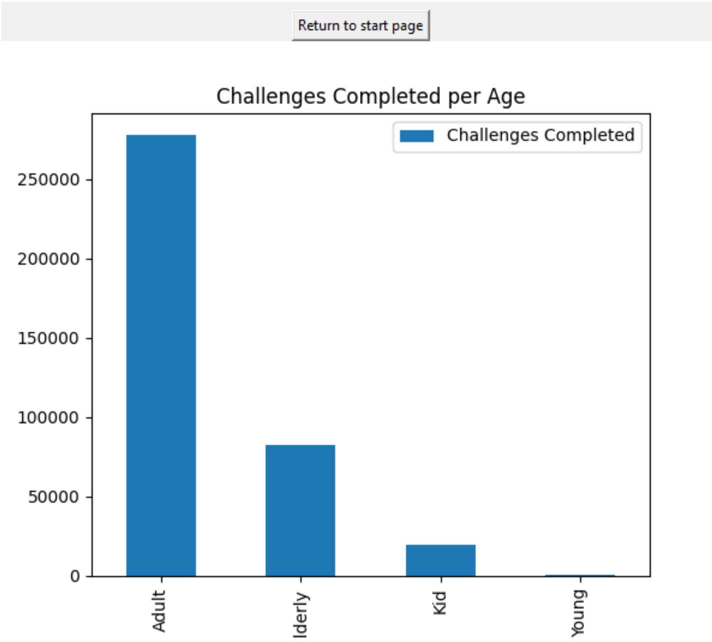

Figure 5.5: Bar chart of challenges completed by age demographic.

As they still manage to bring in a significant amount of revenue, somewhat proportional to their demographic, it displays some issues in the implementation of both the challenge system (as there is no reason for young people to be disadvantaged) and in the motivation system, as they should not have as many in-app purchases as they do if they're not completing challenges at as much of a consistent rate as other players.

It was verified independently that there are a total of 614 challenge instances where the player is young, so the graph is accurately representing the values. This also tells us that despite disadvantaged, young people were still able to engage with challenges; they were just extremely unlikely to do so in comparison to other demographics.

Lastly, we should analyze a sample of some of the individual player experiences:

Player Michael Rivera engaged with 0 challenges, has 44 recorded play moments, a 0 challenge success rate in a total of 22 play sessions. They have purchased 0 items, having spent a total of 0€ and having an inventory with 0 items. They last logged in 0.0 ago.
Player Irene Alkire engaged with 74 challenges, has 74 recorded play moments, a 1.0 challenge success rate in a total of 21 play sessions. They have purchased 9 items, having spent a total of 18€ and having an inventory with 74 items. They last logged in 0.0 ago.
Player Leon Servoss engaged with 20 challenges, has 56 recorded play moments, a 1.0 challenge success rate in a total of 21 play sessions. They have purchased 19 items, having spent a total of 57€ and having an inventory with 20 items. They last logged in 0.0 ago.
Player Daniel Goodwin engaged with 66 challenges, has 75 recorded play moments, a 1.0 challenge success rate in a total of 21 play sessions. They have purchased 8 items, having spent a total of 24€ and having an inventory with 66 items. They last logged in 0.0 ago.
Player Linda Cunningham engaged with 0 challenges, has 0 recorded play moments, a 0 challenge success rate in a total of 0 play sessions. They have purchased 0 items, having spent a total of 0€ and having an inventory with 0 items. They last logged in never ago.
Player Tracie Martinez engaged with 0 challenges, has 62 recorded play moments, a 0 challenge success rate in a total of 22 play sessions. They have purchased 1 item, having spent a total of 1€ and having an inventory with 0 items. They last logged in 0.0 ago.
Player Annette Vigil engaged with 18 challenges, has 34 recorded play moments, a 1.0 challenge success rate in a total of 13 play sessions. They have purchased 282 items, having spent a total of 293€ and having an inventory with 18 items. They last logged in 0.0 ago.
Player Edna Hashimoto engaged with 0 challenges, has 0 recorded play moments, a 0 challenge success rate in a total of 0 play sessions. They have purchased 338 items, having spent a total of 338€ and having an inventory with 0 items. They last logged in never ago.
Player James Schlichter engaged with 0 challenges, has 31 recorded play moments, a 0 challenge success rate in a total of 14 play sessions. They have purchased 1 item, having spent a total of 3€ and having an inventory with 0 items. They last logged in 0.0 ago.
Player Katherine Corella engaged with 0 challenges, has 43 recorded play moments, a 0 challenge success rate in a total of 17 play sessions. They have purchased 0 items, having spent a total of 0€ and having an inventory with 0 items. They last logged in 0.0 ago.

This short sample of player experiences allows us to verify a big issue in the simulation - that the experiences of players are extremely polarized. Players often do not have any engagement with the game as a whole, leading to no days played - but if they do, the engagement becomes an unstoppable feedback loop that leads to them playing every day, often even without having any successful attempt at any challenge.

It is also visible that the challenge success rate is either 1 or 0, showing that not only is the "motivation" system a feedback loop, but so is the inventory one, as players unable to beat nearby challenges won't find the items required for them, and the players that are able to beat them will always have the items required to keep on beating upcoming challenges.

A significant amount of players never engaged with any challenge successfully despite their activity, but this can speak more to the lack of balance in the game's components rather than any flaws in the simulation itself.

5.3 Analysis Results

When it comes to assessing the results of the analysis, our focus shouldn't be only on the numbers produced but on the merit and relevancy of the techniques and on how valuable the observations obtained are. Some result assessment is required, of course, and metrics will be employed to test the quality of the machine learning models, for example, but many of the numbers themselves were already analyzed in the previous section.

The "surface analysis" techniques were used to draw conclusions on the player generation, game data generation, and simulation components of the project, the results of which were double-checked manually with direct database calls. It is thus redundant to make a unique subsection dedicated to the surface analysis results, as the results themselves were verified to be accurate and the techniques used are very common and present in most analysis platforms (which is why they were included in the project.

As such, the following subsections will only cover the interactive map and comprehensive machine learning analysis components.

5.3.1 Interactive Map

The interactive map is a critical component in this platform, allowing for a mode of analysis that is especially important for pervasive games. It is hard to measure its quality and effectiveness with objective metrics as it is a visual medium, and its utility can be extremely context-dependent. Depending on the attributes chosen for display and on the volume of data at hand, different approaches should be taken to ensure good usability and clarity of information.

There are two main elements used to display information in the interactive map, the heatmap and the map marker, which can be shown together or independently. The heatmap is an overlay on the map itself that represents the volume of data with color (with warmer colors representing a higher volume or intensity), while the map marker represents singular instances of any chosen object and can be customized in color, icon, and information displayed on hover.

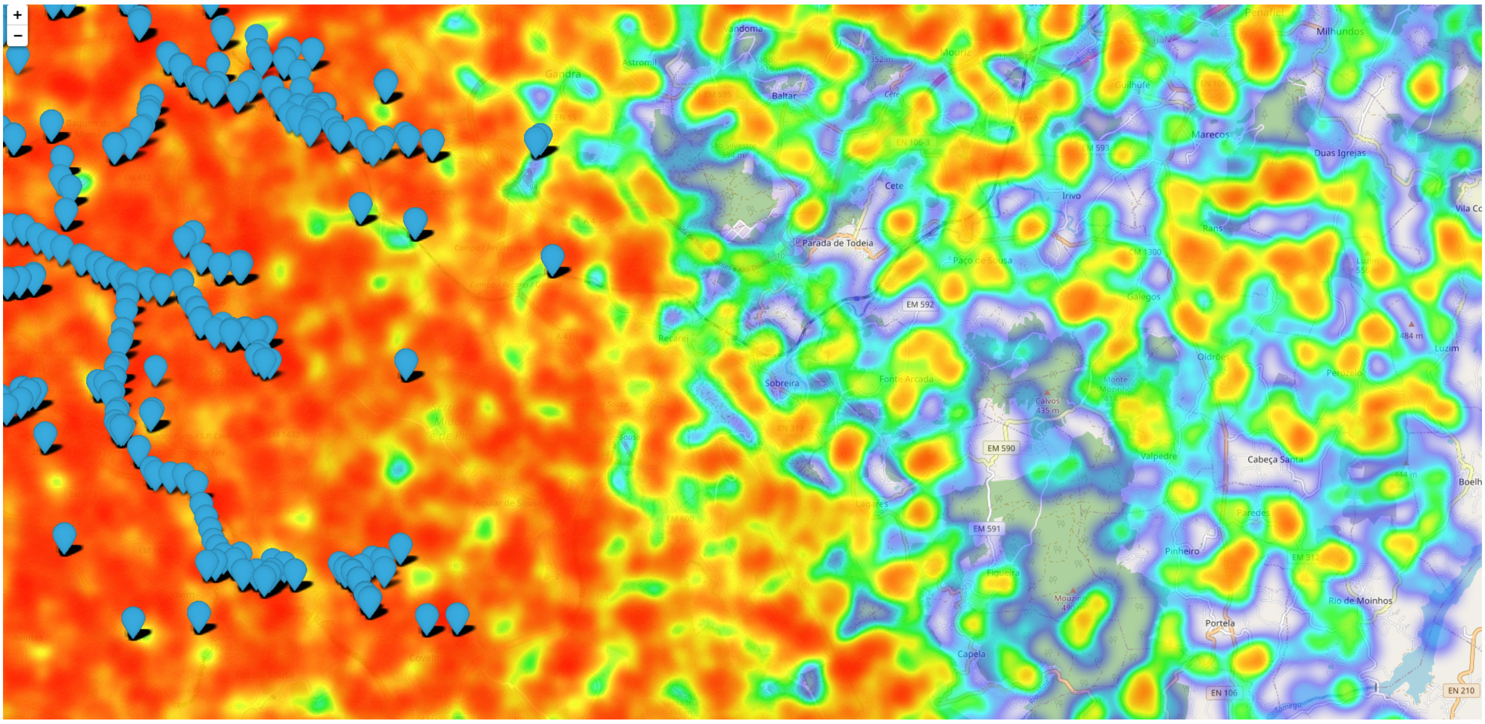

Figure 5.6: Heatmap and map markers being displayed simultaneously.

One of the goals in implementing this interactive map system was seeing how the player activity could coalesce around specific challenges, relying on the heatmap for the former and the map markers for the latter. Although this is possible (as seen in figure 5.6), the user experience is quite poor, and it is nearly impossible to see the real map and geographical references before.

Furthermore, map markers have been shown in various experiments to be extremely resource-intensive. The interactive map is a browser application, which means it has the extra constraint of its resources being naturally much more limited than the rest of the platform. The consequence of this is that map markers must be used extremely carefully, or they - even if displayed on their own - can easily become unusable. If overloaded with too many markers, the browser window can take minutes to open and end up crashing or not displaying the map at all. Even when it does display the map, it is common for it to be extremely slow to operate and for any movement by the user to cause the browser window to freeze.

Figure 5.7 is a display of the kind of freeze that can occur when a map is being rendered unusable by too many map markers.

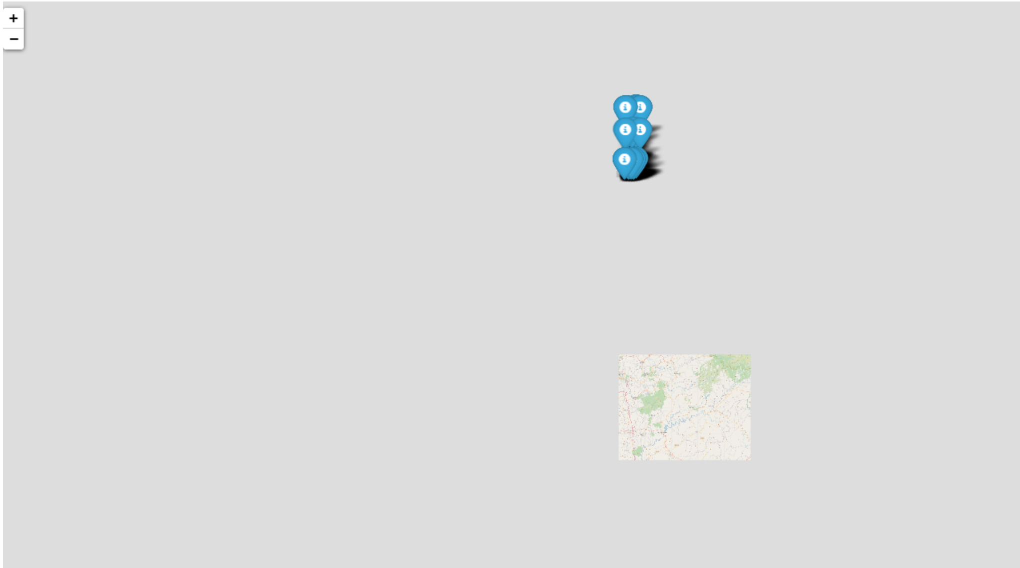

Figure 5.7: An image of a common map freeze due to too many markers

If the map markers are reserved for items with relatively low occurrence, they can be easily integrated successfully into the interactive experience. When it comes to the heatmap, no such problem exists, with performance always being relatively smooth and with no delays.

To assess the utility of the heatmap functions, a simple experiment can be done by comparing the same geographic region but with the element of user interactivity. Using two images that show the same area in different levels of zoom, with only a heatmap in place. The granularity and interactivity of the display allow for careful exploration of the data in the area, easily being able to follow player routines and movement hotspots at various levels of intensity. Figures 5.8 and 5.9 provide a comparison of the same area at a different depth of analysis.

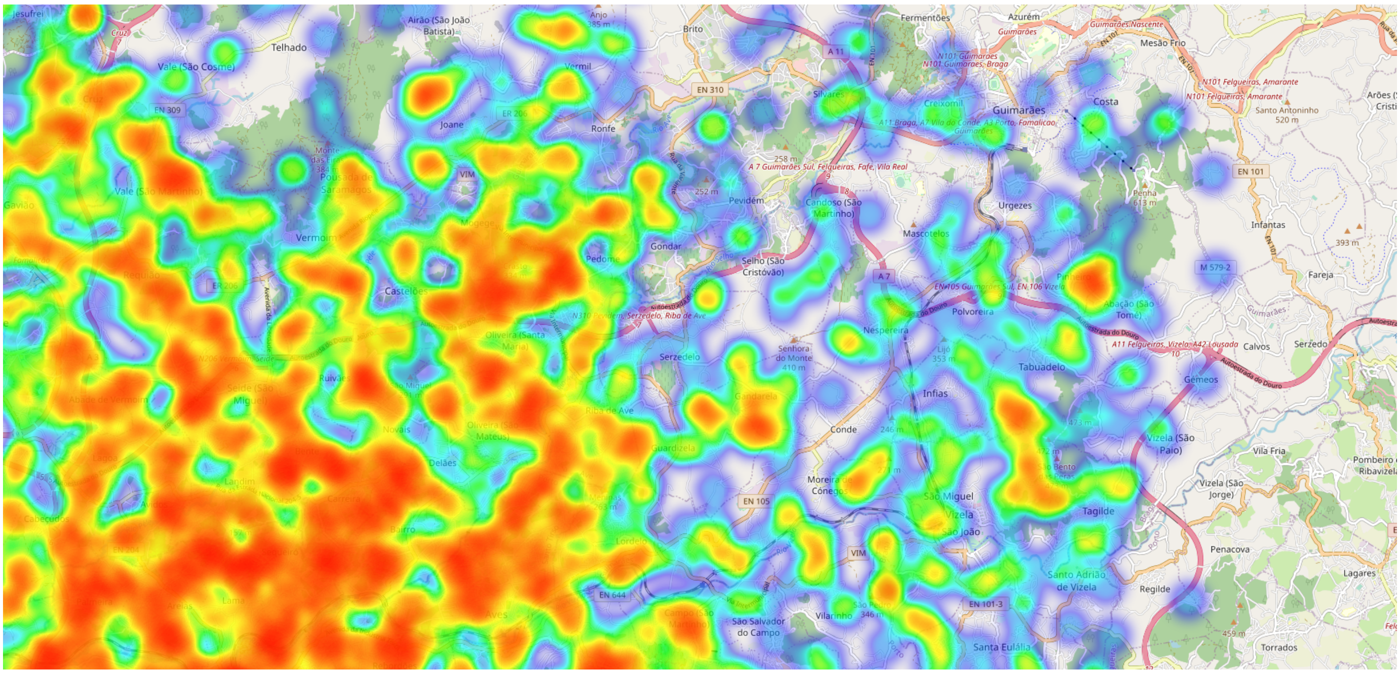

Figure 5.8: A heatmap of player activity

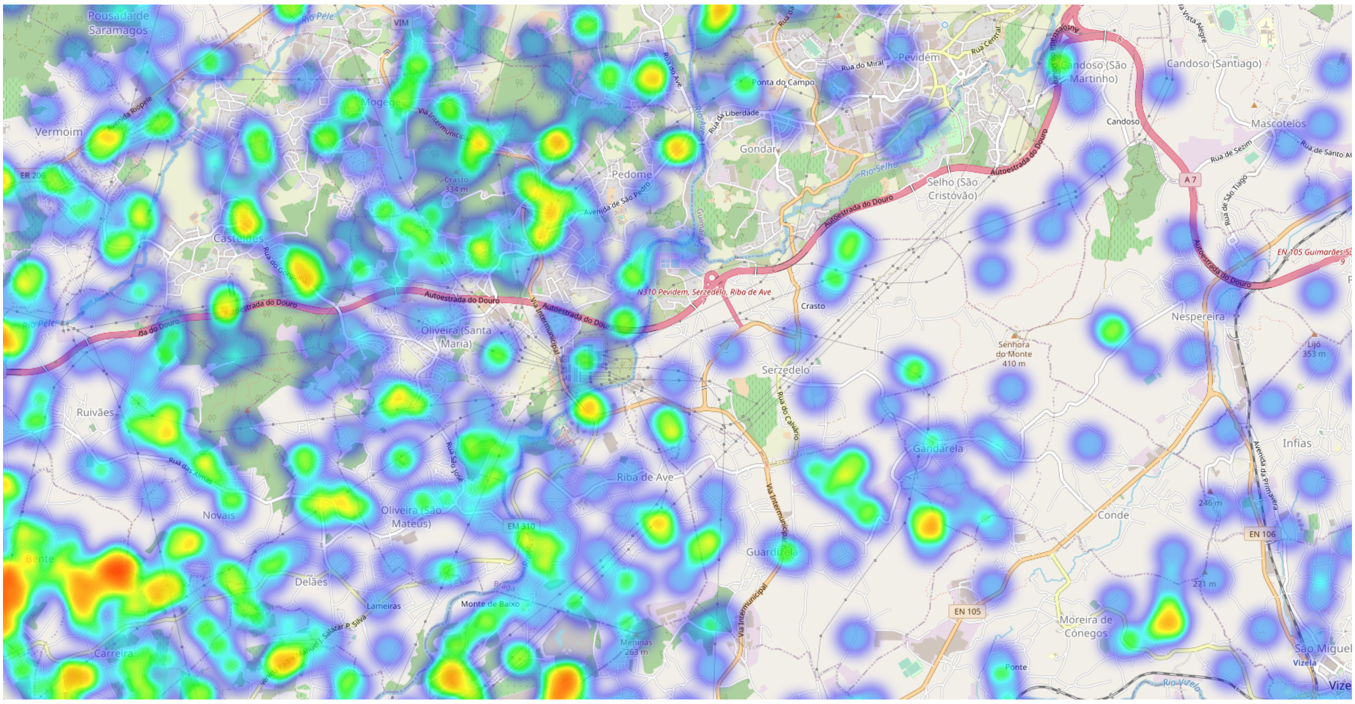

Figure 5.9: The same area as figure 5.8, but slightly more zoomed in.

As we can see in figures 5.8 and 5.9, the geographic features below the heatmap aren't completely obscured as they can always be uncovered by the zoom capabilities. The detail of the heatmap is also on full display, with higher intensity areas that seem more confusing from a distance allowing for easy identification of the relevant hotspots once the user takes a closer look.

5.3.2 Comprehensive Machine Learning Analysis

The machine learning portion of this project can be divided into two main tools - the filtered machine learning and the comprehensive machine learning analysis, the latter of which provides a full analysis using a combination of filtered machine learning components.

As a first step, we should look at the results obtained using these methods. First, doing a small experiment with Filtered Machine Learning, with challenges done and a positive cut-off of over 36 (the average value). The database used for this experiment is the larger one (10000 players).

The average economic status among this group is 1.037274035374491 higher than the average Men are over-represented by a rate of 0.21276666666666666. Database Ratio = 0.4539, Prediction Ratio = 0.6666666666666666 WARNING WARNING Adults are over-represented by a rate of 0.11176666666666668. Database Ratio = 0.6549, Prediction Ratio = 0.7666666666666667

Once the filtering is done, the model is trained on the remaining data, and the predictions are made, resulting in a new dataset. These results are a simple analysis of this dataset, and they seem reasonable and accurate to the numbers described. As expected, a warning pops up when the over-representation of a specific demographic crosses a certain threshold (in this case, manually set to 20%). A similar thing can be seen when the same experiment is run on the smaller database (1000 players):

The average economic status among this group is 1.110246368850752 higher than the average Women are over-represented by a rate of 0.45499999999999996. Database Ratio = 0.545, Prediction Ratio = 1.0 WARNING WARNING Adults are over-represented by a rate of 0.375. Database Ratio = 0.625, Prediction Ratio = 1.0 WARNING WARNING

Although in the tests done on the smaller database, the filtering seems to have reduced the amount of data to such a small number that the predictions become too one-sided and favor one demographic too much. This can be attributed to the volume of data being too small for the filtering process.

The following are the results of the comprehensive machine learning analysis of the larger database:

EARLY ANALYSIS SHOWS...:
Warning: 0.37445887445887444 of children seem to have a lifetime spending over the average.
POSITIVE RESULTS:
The average economic status among this group is 0.6120318067396247 higher than the average
Women are over-represented by a rate of 0.45389999999999997. Database Ratio = 0.5461, Prediction Ratio = 1.0
WARNING WARNING
Adults are over-represented by a rate of 0.1355761904761904. Database Ratio = 0.6549, Prediction Ratio = 0.7904761904761904
The average player count per quadrant is 0.72
The following quadrants have an above average concentration of players:
The quadrant from [41.17404,-8.572059999999999] to [41.20896,-8.52674] has a population 16.666666666666668 above the average.
The quadrant from [41.13912,-8.617379999999999] to [41.17404,-8.572059999999999] has a population 4.166666666666667 above the average.
The quadrant from [41.20896,-8.52674] to [41.24388,-8.48142] has a population 4.166666666666667 above the average.
NEGATIVE RESULTS:
The average economic status among this group is 0.5192511592105202 higher than the average
Women are over-represented by a rate of 0.3057152630279081. Database Ratio = 0.5461, Prediction Ratio = 0.8518152630279081 WARNING WARNING
Adults are over-represented by a rate of 0.012025216102741432. Database Ratio = 0.6549, Prediction Ratio = 0.6428747838972586
The average player count per quadrant is 7.6
The following quadrants have an above-average concentration of players:
The quadrant from [41.17404,-8.572059999999999] to [41.20896,-8.52674] has a population 5.921052631578948 above the average.
The quadrant from [41.24388,-8.48142] to [41.2788,-8.4361] has a population 5.657894736842105 above the average.
The quadrant from [41.20896,-8.52674] to [41.24388,-8.48142] has a population 4.736842105263158 above the average.
The quadrant from [41.1042,-8.6627] to [41.13912,-8.617379999999999] has a population 4.342105263157895 above the average.
The quadrant from [41.13912,-8.617379999999999] to [41.17404,-8.572059999999999] has a population 4.342105263157895 above the average.

Most of the values displayed in this analysis seem to line up with what is expected from the database in question. An addition to the earlier machine learning analysis is the geographic distribution by equal quadrants and the trigger system, both of which seem to be working as intended. It is visible from this analysis also that the negative cut-off allows for a larger volume of data than the positive one, presumably due to there being a significant number of players that did not interact with the game at all after the install. The warning system with the representation is also acting according to expectations.

The following are the results of the same experiment with the comprehensive machine learning analysis, but this time on the smaller database.

EARLY ANALYSIS SHOWS...:
Warning: 0.37209302325581395 of children seem to have a lifetime spending over the average.
POSITIVE RESULTS:
The Elderly are over-represented by a rate of 0.08971428571428569. TruePercent = 0.196, Sim Percent = 0.2857142857142857
The average player count per quadrant is 0.12
The following quadrants have an above average concentration of players:
The quadrant from [41.17404,-8.572059999999999] to [41.20896,-8.52674] has a population 16.666666666666668 above the average.
The quadrant from [41.13912,-8.617379999999999] to [41.17404,-8.572059999999999] has a population 8.333333333333334 above the average.
NEGATIVE RESULTS:
The average economic status among this group is 0.5103925407737663 higher than the average
Women are over-represented by a rate of 0.29394230769230767. TruePercent = 0.545, Sim Percent = 0.8389423076923077 WARNING WARNING
Adults are over-represented by a rate of 0.009615384615384581. TruePercent = 0.625, Sim Percent = 0.6153846153846154
The average player count per quadrant is 0.44
The following quadrants have an above average concentration of players:
The quadrant from [41.17404,-8.572059999999999] to [41.20896,-8.52674] has a population 11.363636363636363 above the average.
The quadrant from [41.24388,-8.48142] to [41.2788,-8.4361] has a population 9.090909090909092 above the average.
The quadrant from [41.1042,-8.6627] to [41.13912,-8.617379999999999] has a population 4.545454545454546 above the average.

These results are very similar to the ones in the larger database, with the same warning occurring due to the high prevalence of women in the negative results (results from the negative cut-off). The most notable difference is visible in the quadrant distribution system, which has a very small amount of data to work with and leads to less useful conclusions (whenever the average is 1 or below, any quadrant listed as "above average" is simply a quadrant that has players). That said, it still displays the prevalence of good experiences in specific areas.

The metrics used to determine the accuracy and quality of the machine learning methods used were the mean absolute error, the mean squared error, and the root mean squared error. Although similar in what they mean regarding the accuracy of the model's predictions, they have advantages over one another in interpretability, which is useful to best convey the results obtained.

When it comes to error analysis, we should first look at the metrics from the Filtered ML tool available in the Surface Analysis menu. Due to the variety of possibilities and experiments that can be done with this versatile component, we have decided that for demonstration purposes, we would display only one experiment, repeated it in both databases. The test in question was a filter of the most representative attribute, challenges done, with the value of 36 (the average for the attribute in question) being used as the cut-off.

As in there are always two models used per execution, two sets of results are produced. One of these models is a Random Forest Classifier - for the classification labels such as age and gender - and the other is a Random Forest Regressor - for the geographic coordinates and socioeconomic indicator.

Table 5.8 contains the metrics obtained when running the experiment described with the smaller, 1000 player database.

	RF Regressor	RF Classifier
Mean Absolute Error	0.421	0.417
Mean Squared Error	0.539	0.417
Root Mean Squared Error	0.734	0.645

Table 5.8: Filtered ML experiment (small database)

Table 5.9 contains the metrics obtained when running the experiment described with the bigger, 10000 player database.

	RF Regressor	RF Classifier
Mean Absolute Error	0.141	0.383
Mean Squared Error	0.055	0.383
Root Mean Squared Error	0.235	0.619

Table 5.9: Filtered ML experiment error values (small database)

It is notable how the error values regression values are much smaller in the larger database, with the classifier metrics being somewhat closer due to the naturally smaller variance in these, lowering the impact of the larger database.

Regarding the comprehensive machine learning analysis, this one produces double the number of tables as four models are run instead of two, providing machine learning models that represent the best results players can obtain and the worst (with two models each).

The error metrics obtained by filtering the best results in the smaller database:

	RF Regressor	RF Classifier
Mean Absolute Error	0.175	0.285
Mean Squared Error	0.086	0.285
Root Mean Squared Error	0.294	0.535

Table 5.10: Comprehensive ML error values (small database, positive cut-off)

The error metrics obtained by filtering the worst results in the smaller database:

	RF Regressor	RF Classifier
Mean Absolute Error	0.108	0.055
Mean Squared Error	0.039	0.055
Root Mean Squared Error	0.197	0.236

Table 5.11: Comprehensive ML error values (small database, negative cut-off)

It is notable from these results already that the negative results have much lower values, implying that there's less variation in the data despite a similar scale, as was described in its implementation.

Finally, here are the same two tables of results but for the larget database. First, the table for the positive cut-off:

	RF Regressor	RF Classifier
Mean Absolute Error	0.129	0.112
Mean Squared Error	0.053	0.111
Root Mean Squared Error	0.23	0.335

Table 5.12: Comprehensive ML error values (larger database, positive cut-off)

Comparing these results to the positive cut-off from the smaller database, the values from the regressor are relatively close to the earlier ones, but the classifier values have changed a bit more. This implies that the larger amount of data allows for more accurate demographic classification (age and gender) but that the coordinates and socioeconomic status aren't that different from earlier values, obtaining only a minimal improvement.

	RF Regressor	RF Classifier
Mean Absolute Error	0.097	0.060
Mean Squared Error	0.032	0.060
Root Mean Squared Error	0.180	0.244

Table 5.13: Comprehensive ML error values (larger database, negative cut-off)

When it comes to the regression model, the changes due to database size are similar to what occurred with the positive cut-off. However, analyzing the results from the classification value tells a different story, as the error values have actually gone up compared to the smaller database, if only slightly. As suspected, this possibly showcases that the values were already incredibly low, and thus instead of helping produce more accurate predictions, the added data only introduced a very small amount of variability.

Overall, the error metrics obtained in all of these tables are relatively low given the scale of the values in the attributes involved (with variables having deltas in the scale of units), so we can surmise that the Comprehensive Machine Learning Analysis is an effective method of providing information regarding the pervasive game.

5.4 Overall Results Discussion

One of the issues with our data generation is that the demographics are not realistic to pokémon go, but accurate to the real world. This was a necessity as the census data used provided a direct link to player coordinates, and any quality player data from other pervasive games was only general demographic divisions that covered data from the entire world or only from the United States of America. Due to this, we felt that data would not be the best to generate a system that required a certain level of accuracy from the association with its location and that other data would be better served only as a form of comparison after the fact. This validation shouldn't be taken at face value because, as has been mentioned, the demographics compared are of two entirely different places. It is, however, the best way to validate those results we can find.

Ideally, a focus group of pervasive game players would be used to detect average demos, but that we do not have access to such resources in the numbers needed (or they would even be used for the initial player generation). Averages of the database allow us to verify that the ratios fit the general population of the area and that issues coming from their representation exist due to the simulation.

Demographic results might not be entirely accurate to the complete country as the in-game data was generated only on demographics from Porto, ensuring a connection between player profile and location. This explains why the overall Portuguese data has a smaller percentage of Adults in comparison to our data, as it is common for adults to move from the interior to Porto (a big city) to work. The inner parts of the country have significantly more elderly people and children.

Unfortunately, with the resources available to us, many of this project's components are impossible to test with the scientific rigor we would desire, as they are attempting to emulate data that we'd use were it available to us, and the success of that emulation can only be measured in absolute terms by comparing it with that data we can't access. It is also hard to find useful metrics for some of the goals of this project, such as measuring the usefulness to a developer of some of the functions available in the platform.

One of the main goals of this project is to present a proof of concept for an innovative pervasive games platform, and as such, many of the features proposed and implemented have been so to explore the limits of innovative techniques and structures. It is then difficult to access with perfect scientific metrics the impact they might have in a real-world context. On this subject, we can only speculate, so the focus is to measure the accuracy of the techniques and leave the question of their potential value to those who might be interested in using such a platform and leading to its creation as a complete commercial product.

Many of the evaluation methods used in this section entail results from one module being checked or dependent on another module of the platform. A flaw in the player profiling system can maybe not appear in it but only show itself in the simulation, and it is hard to determine the true source of the problem.

It is hard to discuss the success of the surface analysis methods as they are not innovative and are mostly simple math - as such, there are no valuable analyses to be made of them other than to say they've been verified to be correct. As the methods used have been checked many times via separate database queries and independent calculations, we can conclude that the surface analysis techniques are a good implementation of the task they were intended to perform.

One point of contention that can be had with the KPIs regarding purchases is that the item pricing for the game items is not 1:1 with real-life examples, and thus, the comparison is flawed. The reason for this is that Portugal has, on average, lower socioeconomic status than other Western nations in which games such as Pokémon Go are popular, and using the same pricing would naturally lead to worse results than in other nations. The simulation and the prices of items within it were created with the disposable income of the generated players in mind, with the goal in mind of finding equivalent results.

An issue that becomes obvious when using the interactive map for analysis is that the game data generation systems ended up creating an unusually high volume of data compared to what was expected. The higher volume of data in the simulation was somewhat intentional, as the goal was to ensure it is possible to collect valuable data from urban players, and this means that a significant portion of them need to be able to engage with challenges. To try to maintain the same dynamics as popular pervasive games, the ratio was kept, and similar methods were used. This strategy was possibly harmful by providing too many challenges to urban players and not being realistic to the content drought that occurs in rural areas in popular pervasive games.

The interactive map system itself can not be said to be a complete success. Despite the valuable conclusions that can be drawn from it and the quality of its use with the heatmap alone, the finished system with only heatmaps leaves a bit to be desired with its simplicity. The high volume of data leading to massive slowdown when using map markers is a problem, but time constraints prevented implementing other visual indicators that could be interesting or fixing the performance issues associated with the map markers. Due to this, we recommend only activating the map marker features on either fixed landscape points - for higher relevance - or for developers to only choose objects to be represented as map markers if they know they'll show up in lower volumes and that they are essential to the analysis. This was a problem when doing an analysis of a single district, so it will almost assuredly be an issue if the same methods are applied in platforms that aim to represent larger areas. Here, the problem lies in the specific implementation with Folium and OSM, and as we've seen in other examples such as PoGoMap [60], a similar goal can be achieved without the performance issues via slightly different implementations.

Comprehensive machine learning data is a success, though it would be useful to have more metrics that allowed us deeper insights into its performance.

Chapter 6

Conclusions

Contents

The purpose of this chapter is to serve as a conclusion to the dissertation. The following sections contain the author's thoughts regarding the main problems, breakthroughs, and takeaways of the project as a whole, along with reflections on how the work developed could be used and expanded upon in the future. In the first section, Section 6.1, the constraints that prevented the project from being implemented as idealized are discussed. In section 6.2, the biggest takeaways from this project are discussed, along with examining what elements were most valuable and innovative. Section 6.3 provides a look at future work, discussing how the contributions talked about in the previous section can be used by others - either by exploring further research into this topic or by using one of the contributions for related projects. Finally, section 6.4 is comprised of a summary of the work developed and provides a conclusion to the dissertation.

6.1 Limitations

In this project, the limitations and constraints found during the early stages ended up having such an impact on the work that could be done that they shaped the development process entirely, adding heavy new requirements to the work that needed to be done and aggravating the constraint of time, shortening the amount that could be spent on the main focus of this thesis. Due to this, one of the main constraints that must be pointed out is time. With only one month of work for a single developer, the data analysis components that aimed to tackle balance and injustice in pervasive games weren't given the time and work required for their potential to be fully realized.

Time constraints led to a lot of corners being cut, and as was mentioned in section 4.5, many of the intended features that would make the simulation and analysis more interesting were not able to be implemented. Overall, all steps of this project ended up being hurt by this, and it is important to mention that there's room to explore their ideas in individual projects, but that all of these areas do not have enough tools and past work for them to be integrated effectively in a platform such as the one that was developed without severe compromises.

The focus of this project was, initially, only the analysis features that could serve as the basis for a pervasive platform. However, the lack of reliable pervasive game data and player data, or even any tools that would help us generate fake player data reliably, led to the necessity of the game data generation modules. Old Pokémon Go APIs that list challenges and in-game data are out-of-order or unreliable, and many of the other available sources that we used to check the validity of our data (such as PogoMap [60]) are comprised entirely of user-submitted data and do not have an open API that users can extract data from.

The games themselves are, of course, also not very open with their data. This is due to ethical concerns regarding the sharing of player data without their consent and - where it does not pertain to user data - due to their own secrecy and privacy. Both of these constraints combined made it so that it is very hard to base a project around a massive amount of data from existing pervasive games without being closely linked to them.

This project was incapable of fully tackling its goals, being unable to address the issues of injustice against protected minority groups. Protected minority groups that weren't included were those based on race, color, national origin, religion, sexual orientation, and physical or mental disability. As there was reliable data for them, the sex and age groups were used for demographic profiling. There are multiple reasons for the project being unable to tackle these issues, which are core to its goals and motivations. The main constraint that prevented this analysis was a lack of reliable distribution data for these groups in the area that was analyzed. As has been mentioned, Portugal does not collect data regarding some of these groups in its census, as there are ethical concerns associated with it. Due to this, there is no trustworthy source for, for example, location-specific ethnic distribution in the district of Porto, which was the region analyzed.

Furthermore, it was already a concern of the authors that relying on demographic data for player profiling could end up producing unbacked and unjustified prescriptive claims in the player data generation and simulation systems. It is for this reason that sex does not cause any difference in the player's personality values or their behavior in the simulation. In fact, the only prescriptive claims done regarding any group in the project are about age, and they were about typical work and school routine schedules that influenced times where players could engage with the game. Due to this ethical concern, the constraints mentioned regarding data availability and the fact that the methods would end up being the same as they were for gender and age (meaning, this demographic would end up being just another variable with no causal links in the simulation), it was decided that it was better not to tackle injustices regarding any other demographic group for they could be poorly realized.

6.2 Contributions

The contributions of this work mainly lie in work done in the following four components: the artificial player generation system for a pervasive game, the simulation of gameplay within a pervasive game, the interactive map system, and, most importantly, the comprehensive ML analysis system.

The player generation system is relatively unique in how, just like the pervasive games it is meant for, it relies extensively on data obtained from the geographic region. As such, its unique implementation brings about a new contribution to the area that might be useful in upcoming work with other scopes. It is also a useful tool for new pervasive game developers that want to begin making their games but lack a player-base to test on.

The gameplay simulation system wasn't as successful as expected, which changes the way in which it should be evaluated as a contribution. The core of the system seems to function as intended, while the flaws that led it to become a feedback loop for many players should be taken as a cautionary tale for future attempts. On its own, it can be an extremely useful tool to test new changes to a pervasive game, and the concept should not be discarded due to faulty implementation.

The interactive map system is mostly an innovation when it comes to it being integrated into an analysis platform, as interactive maps for pervasive games (such as the Harry Potter community map [52] or PoGoMap[60]) are very common tools for fans. This is one of the reasons why it is so important as a way for analyzing pervasive games, ensuring developers do not forget how the game looks like to its players and that the location-based elements are not abstracted from the geography itself. Its integration in a universal, pervasive game analysis platform is essential, and its feature-set can be expanded greatly beyond simple map markers and heatmaps.

When it comes to the comprehensive machine learning analysis system, we consider its implementation a success and the main contribution that can be taken from this work. As was the goal originally, the use of a machine learning model allows for entirely new conclusions and perspectives to be obtained from its predictions. Furthermore, the model used is the best technique we've found for determining where issues of balance and injustice exist or can come up, meaning that it can be an essential tool for pervasive game analysis going forward if implemented in a fully realized version of the platform developed within this work.

6.3 Future Work

The limited-time and suddenly increased scope of the project led to many of the goals aimed for initially not being explored to the desired extent. The added amount of systems that spawned from these constraints, alongside exacerbating them, also introduce an element of unreliability in the results obtained. As all of the components rely on one another, implementation flaws in one of the earliest components can have repercussions in all of the other systems. For these reasons, it would be ideal for these components to be explored in-depth independently in projects that do not have the same constraints. A larger development team and more time available to explore these issues would also be important.

As has been mentioned previously, many of the constraints in the development of this project come from not being associated with any existing large-scale pervasive game or other location-based apps whose users consent to their data being used. The issue with this is that developers such as Niantic games probably already have their in-house tools to make decisions regarding balance and injustice that will not be made openly available, so finding a suitable partner for development that would also accept keeping the application as a public tool would be quite difficult.

Many of the ideas described in the earlier chapters, and in particular in section 4.5, were not fully implemented due to many of the constraints mentioned. Featuring them in future work would be a good way to further the goals of this project and to make any final product that results from them a deeper and more complex tool.

Developments and applications of the player generation system are easier to envision. A tool with such a wide scope can also be useful for pervasive game developers everywhere and possibly developers of other location-based applications, so it would be interesting to see how projects with other needs iterate upon the concept implemented here. A project solely focused on creating a reliable player profiling and simulation system, with a dedicated focus on studying player behaviors (instead of all of the other goals of this project), could easily obtain extremely good results that do explore this subject with a greater degree of complexity.

In fact, one of the main things that should be tackled in future work is the project's handling of user demographics. Due to the specific constraints that have already been mentioned, the only demographics that have been taken into account were age, gender, socioeconomic status, and location. While these are very important, other demographics that also are common targets of systemic discrimination (which is often unintentional, as in pervasive games, but still has massive effects and implications) were not covered by this project. These include race, ethnicity, sexuality, disabilities, among others. As minority groups of these identities are often subject to injustice, it is imperative that future work takes them into account.

Regarding the comprehensive machine learning system and trigger system, future work should be able to test these components in different contexts from the ones in this project. Although they were relatively successful with the data provided, testing them on real data obtained in genuine player interactions is the best way to improve upon it and ensure the system is robust enough to provide reliable results and utility in versions of the platform that will get used by developers.

6.4 Summary

Just as with innovations in any other art form, new game genres are often born from the way designers integrate cutting-edge technology with previously unrealized concepts and ideas. The pervasive game genre is the perfect example of this. Using innovative technology, pioneers in pervasive gaming explored entirely new and unique experiences in gaming, even if the early products were quite unpolished and rudimentary by today's standards. As outlined by its relation to the magic circle concept, the pervasive game genre was a breakthrough in experiences that expand the scope of games temporally, spatially, and even socially in ways never previously thought possible.

Taking a look through its history, it is clear how much the genre has evolved in the past 20 years and how many of its concepts are now present in other types of games such as LARPs and ARGs. Due to its reliance on technology, the viability of quality commercial products has grown with time, reaching a point where its quality and polish would be on par with other popular commercial games in the mid-2010s. It was then that the popularity of the genre exploded with the release of Pokémon Go, a game that leveraged one of the world's most popular intellectual properties and combined it with a new exciting game genre. The genre's popularity exploded, and there are still millions of daily players.

The explosion of the genre was not as smooth as one would expect, however. The lack of balancing done in handling contextual data by these new popular location-based mobile pervasive games leads to many players being disenfranchised. As part of large-scale social phenomena, these can have severe impacts on our society's social fabric. The biases within their systems caused social injustices that must be fixed, and the genre as a whole will not evolve until the issue of balance is addressed.

It is here that a project like this dissertation comes into place. The platform that was developed has the capability of providing a framework and tools to facilitate game balancing in pervasive games, fixing imbalances in existing games, and helping develop new ones. The objective is not just to help pervasive games thrive in their current form but also to help new ideas for the use of pervasive elements to be developed more easily and effectively, ensuring that balance and justice do not get disrupted in the process.

It is impossible to cover the issues we aim to fix in this work without addressing the question of equity against equality. It is not our goal to make player experiences homogeneous and the same. Part of what is so interesting about pervasive games, and many games in general, is how well they lend themselves to creating unique stories for their players that can then share them with one another. Allowing for singular experiences and for the player environment to shape the gameplay sessions is one of the most important traits in pervasive games, and we would not want to balance it out of the genre. However, it is important to guarantee some degree of equity, ensuring that player experiences never dip below a certain threshold of activity and boredom. This is why the analysis techniques used allow for imbalances, sometimes significant ones, but only signal warnings to the developers when these imbalances become very severe or other ethical concerns are raised by the numbers involved (such as the imbalance being kids spending too much money on the game compared to other demographics). The question of equity and equality is one that the platform also aims to strike a balance in.

The project has somewhat achieved its goals, obtaining good results considering the constraints that it was developed under. The strategies and techniques used allow for detecting imbalances and the problems that lead to them but focus on a narrow set of demographics when it comes to enforcing justice and fairness, which needs to be expanded upon in future works.

This platform cannot be said to be a fully finished product. Due to the constraints that have been described, the work developed is unfortunately very self-contained and it would take developers a lot of effort to alter their game data to work with the existing platform. It is our goal that the data analysis platform developed is seen more as a proof of concept, as a blueprint for future software that expands upon it and sees it become fully realized. The analysis methods used are reliable, and the systems developed can be extremely useful, as was seen in the previous chapter. The innovative aspects this platform tested proved to be effective, and their implementation in commercial, full-scale products is perfectly feasible, which is what we hope this dissertation inspires.

It is critical to mention that this work is important in part because it is applicable to not just pervasive games but to any interactive application that uses pervasive elements (location, time, and social aspects) as a way to generate content for its users in the world. Balance and justice in the distribution of content and providing good experiences is important, regardless of the type of interactive application.

In the future, it is our aim that this work can get used to not only facilitate the work of developers interested in creating new pervasive games but also that it helps make the experience of players everywhere better, more balanced, and more just.

References

[1] Folium — Folium 0.12.1 documentation.

[2] Home :: FAT ML.

[3] Humanitarian Data Exchange.

[4] Matplotlib: Python plotting — Matplotlib 3.4.2 documentation.

[5] Niantic Support.

[6] scikit-learn: machine learning in Python — scikit-learn 0.24.2 documentation.

[7] scikit-learn: machine learning in Python — scikit-learn 0.24.2 documentation.

[8] Sponsored Locations for Business – Niantic.

[9] The Disappearing Computer Initiative - Main Page.

[10] The Historical Marker Database.

[11] The Silph Road | Pokemon GO Resources.

[12] tkinter — Python interface to Tcl/Tk — Python 3.9.5 documentation.

[13] Worldwide Public Holidays - Nager.Date.

[14] urrent weather and forecast - OpenWeatherMap.

[15] WorldPop, 2021.

[16] Jeferson Arango-López, Francisco Luis Gutiérrez Vela, Cesar A. Collazos, Jesús Gallardo, and Fernando Moreira. GeoPGD: methodology for the design and development of geolocated pervasive games. *Universal Access in the Information Society*, pages 1–13, oct 2020.

[17] ARInsider. Data Point of the Week: What Makes Pokemon Go Successful? - AR Insider.

[18] Clemens Arth, Raphael Grasset, Lukas Gruber, Tobias Langlotz, Alessandro Mulloni, and Daniel Wagner. The History of Mobile Augmented Reality. may 2015.

[19] Staffan Björk, Jussi Holopainen, Peter Ljungstrand, and Karl Petter Åkesson. Designing ubiquitous computing games - A report from a workshop exploring ubiquitous computing entertainment. In *Personal and Ubiquitous Computing*, volume 6, pages 443–458. Springer London, 2002.

[20] Craig Chapple. Pokémon GO Hits $1 Billion in 2020 as Lifetime Revenue Surpasses $4 Billion, nov 2020.

[21] Darryl Charles and Benjamin Ultan Cowley. Behavlet Analytics for Player Profiling and Churn Prediction. pages 631–643. 2020.

[22] Ashley Colley, Jacob Thebault-Spieker, Allen Yilun Lin, Donald Degraen, Benjamin Fischman, Jonna Häkkilä, Kate Kuehl, Valentina Nisi, Nuno Jardim Nunes, Nina Wenig, Dirk Wenig, Brent Hecht, and Johannes Schöning. The geography of Pokémon GO: Beneficial and problematic effects on places and movement. In *Conference on Human Factors in Computing Systems - Proceedings*, volume 2017-May, pages 1179–1192. Association for Computing Machinery, may 2017.

[23] OpenStreetMap Wiki Contributors. Overpass turbo, 2016.

[24] Eddie Duggan. Squaring the (Magic) Circle: A Brief Definition and History of Pervasive Games. pages 111–135. 2017.

[25] Ben Egliston. Big playerbase, big data: On data analytics methodologies and their applicability to studying multiplayer games and culture by Ben Egliston. *First Monday*, 21(7), 2016.

[26] Katsuya Eguchi, Hisashi Nogami, and Takashi Tezuka. Animal Crossing, 2001.

[27] Jennica Faik, Peter Ljungstrand, Staffan BjÖrk, and Rebecca Hansson. Pirates: Proximity-triggered interaction in a multi-player game. In *Conference on Human Factors in Computing Systems - Proceedings*, pages 119–120, 2001.

[28] Jennica Falk and Glorianna Davenport. Live role-playing games: Implications for pervasive gaming. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3166:127–138, 2004.

[29] Adnan Farooqui. Pokemon Go Daily Active Users Drop By 23 Million | Ubergizmo, apr 2017.

[30] Adam Fletcher. Applying Data Science to Understand Your Players. *Gamasutra*, 2017.

[31] Jens Grubert, Tobias Langlotz, Stefanie Zollmann, and Holger Regenbrecht. Towards pervasive augmented reality: Context-awareness in augmented reality. *IEEE Transactions on Visualization and Computer Graphics*, 23(6):1706–1724, jun 2017.

[32] Juho Hamari, Aqdas Malik, Johannes Koski, and Aditya Johri. Uses and Gratifications of Pokémon Go: Why do People Play Mobile Location-Based Augmented Reality Games? *International Journal of Human-Computer Interaction*, 35(9):804–819, may 2019.

[33] Deborah Hellman. ARTICLES MEASURING ALGORITHMIC FAIRNESS. Technical Report 4, 2020.

[34] Johanness Henriksson. Validating Game Design with Game Analytics in a Location-based Game. Technical report, 2018.

[35] Jussi Holopainen. D13.6 Mythical: The Mobile Awakening Final Report - IPerG. Technical report, 2008.

[36] Katherine B. Howe, Christian Suharlim, Peter Ueda, Daniel Howe, Ichiro Kawachi, and Eric B. Rimm. Gotta catch'em all! Pokémon GO and physical activity among young adults: Difference in differences study. *BMJ (Online)*, 355, 2016.

[37] Christopher Huffaker. Why are there fewer Pokemon locations in black neighborhoods? | Miami Herald, 2016.

[38] W A Ijsselsteijn, De Kort, and Poels. GAME EXPERIENCE QUESTIONNAIRE. Technical report, 2013.

[39] Steve Jackson. *Killer – The Game of Assassination*. 1982.

[40] Joo Tiago Pinheiro Neto Jacob and Antnio Fernando Coelho. Issues in the development of location-based games. *International Journal of Computer Games Technology*, 2011.

[41] Kalle Jegers. Pervasive game flow: Understanding player enjoyment in pervasive gaming. *Computers in Entertainment*, 5(1):9, jan 2007.

[42] Levente Juhász and Hartwig H. Hochmair. Where to catch 'em all?–a geographic analysis of Pokémon Go locations. *Geo-Spatial Information Science*, 20(3):241–251, jul 2017.

[43] Hideo Kojima. Boktai: The Sun is in Your Hand, 2003.

[44] Shiva Kooragayala and Tanaya Srini. Pokémon GO is changing how cities use public space, but could it be more inclusive? | Urban Institute, 2016.

[45] Jeff Larson, Surya Mattu, Lauren Kirchner, and Julia Angwin. How We Analyzed the COMPAS Recidivism Algorithm — ProPublica. *ProPublica*, may 2016.

[46] Riley Little. Report: Pokemon GO Paying Player Total Has Dropped 79% Since Peak, 2016.

[47] Frans Mäyrä and Olli Sotamaa. All The World's A Botfi ghter Stage: Notes on Location-based Multi-User Gaming. Technical report.

[48] G. Melki, A. Cano, V. Kecman, and S. Ventura. Multi-target support vector regression via correlation regressor chains. *Information Sciences*, 415-416:53–69, 2017.

[49] Markus Montola. Exploring the edge of the magic circle: Defining pervasive games. 01 2005.

[50] Markus Montola, Jaakko Stenros, and Annika Waern. *Pervasive Games: Theory and Design - Experiences on the Boundary Between Life and Play*. 2009.

[51] L.E. Nacke, M.N. Grimshaw, and C.A. Lindley. More than a feeling: Measurement of sonic user experience and psychophysiology in a first-person shooter game. *Interacting with Computers*, 22(5):336–343, 2010.

[52] Niantic. Community Map - Harry Potter Wizards Unite.

[53] Eva Nieuwdorp. The Pervasive Interface: Tracing the Magic Circle. Technical report, 2005.

[54] Eva Nieuwdorp. The Pervasive discourse: An analysis. *Computers in Entertainment*, 5(2):13, apr 2007.

[55] Nintendo Dream and Hideo Kojima. Boktai - 2003 Developer Interview. *Nintendo Dream*, apr 2003.

[56] OpenStreetMap Contributors. OpenStreetMap, 2021.

[57] Katriina Penttilä. History of Escape Games examined through real-life-and digital precursors and the production of Spygame Digital Culture Degree Programme in Cultural Production and Landscape Studies School of History, Culture and Arts Studies. Technical report, 2018.

[58] Victor M. Perez-Colado, Dan Cristian Rotaru, Manuel Freire, Ivan Martinez-Ortiz, and Baltasar Fernandez-Manjon. Learning analytics for location-based serious games. In *IEEE Global Engineering Education Conference, EDUCON*, volume 2018-April, pages 1192–1200. IEEE Computer Society, may 2018.

[59] Daniel Pimentel Nunes, João Tiago Pinheiro Neto Jacob Co-supervisor, and Rui Pedro Amaral Rodrigues. FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO Safety in Pervasive Gaming. Technical report, aug 2020.

[60] PogoMap. PogoMap.Info - Pokemon GO World Map, 2021.

[61] Pordata. PORDATA - População residente, média anual: total e por grupo etário, 2021.

[62] Pordata. PORDATA - População residente, média anual: total e por sexo, 2021.

[63] Noah Raffin, Antonin and Hill, Ashley and Ernestus, Maximilian and Gleave, Adam and Kanervisto, Anssi and Dormann. Stable Baselines3, 2019.

[64] Rustici Software. What is xAPI aka the Experience API or Tin Can API.

[65] Katie Salen and Eric Zimmerman. *Rules of Play: Game Design Fundamentals*. 2003.

[66] Jay Schneider, Jay Schneider, and Gerd Kortuem. How to Host a Pervasive Game - Supporting Face-to-Face Interactions in Live-Action Roleplaying. *INTERACTIONS IN LIVE-ACTION ROLEPLAYING. UBICOMP WORKSHOP ON DESIGNING UBIQUITOUS COMPUTING GAMES*, 2001:6, 2001.

[67] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, jul 2017.

[68] Six to Start. Zombies, Run!, 2011.

[69] Norbert A Streitz. AUGMENTED REALITY AND THE DISAPPEARING COMPUTER. Technical report.

[70] SurveyMonkey Intelligence. Pokémon GO demographics: The evolving player mix of a smash-hit game | by SurveyMonkey Intelligence | Medium, 2016.

[71] Paul Tassi. 'Pokémon GO' YouTubers Are Sparring With Fans Over Free Mewtwos And 'Fairness'. 2017.

[72] Dave Thier. No, 'Pokémon GO' Is Not Fair. Yes, It's A Problem. 2017.

[73] Sahil Verma and Julia Rubin. Fairness Definitions Explained. *IEEE/ACM International Workshop on Software Fairness*, 18, 2018.

[74] Willem Waegeman, Krzysztof Dembczyński, and Eyke Hüllermeier. Multi-target prediction: a unifying view on problems and methods. *Data Mining and Knowledge Discovery*, 33(2):293–324, 2019.

[75] Mark Weiser. The computer for the 21 st century . *ACM SIGMOBILE Mobile Computing and Communications Review*, 3(3):3–11, jul 1991.

[76] WizardsUniteWorld. Wizards Unite World - News, Guides, Updates and Wiki.

[77] Nathan Young. Pokémon go in the field: Digital gaming apps and ethnographic methods. *Journal of Folklore Research*, 57(3):63–85, 2020.

[78] Bo Zhao and Shaozeng Zhang. Rethinking Spatial Data Quality: Pokémon Go as a Case Study of Location Spoofing. *Professional Geographer*, 71(1):96–108, jan 2019.