**PREDICTIVE E-SPORTS GAME ANALYSIS USING MACHINE LEARNING APPROACHES**

# TITLE PAGE

**Atakan TUZCU**

**MASTER OF SCIENCE THESIS**

**Department of Computer Engineering**

**Supervisor: Dr. Okan BURSA**

**İzmir**

**İzmir Bakırçay University**

**Graduate Education Institute**

**June 2023**

# FINAL APPROVAL FOR THESIS

This thesis titled “Predictive E-Sports Game Analysis Using Machine Learning Approaches” has been prepared and submitted by Atakan TUZCU in partial fulfilment of the requirements in “İzmir Bakırçay University Directive on Graduate Education and Examination” for the Degree of Master of Science in Computer Engineering Department has been examined and approved on …/…/…

|  |  |  |
| --- | --- | --- |
| **Committee Members** | **Title, Name, and Surname** | **Signature** |
| Member (Supervisor) | Dr. Okan BURSA |  |
| Member |  |  |
| Member |  |  |
| Member |  |  |
| Member |  |  |

……………………………

Prof. Dr. Özge TÜZÜN ÖZMEN

Director of Graduate Education Institute

# ÖZET

MAKİNE ÖĞRENMESİ YAKLAŞIMIYLA KESTİRİMCİ E-SPOR OYUN ANALİZİ

Atakan Tuzcu

Bilgisayar Mühendisliği Anabilim Dalı

İzmir Bakırçay Üniversitesi, Lisansüstü Eğitim Enstitüsü, Haziran 2023

Danışman: Dr. Okan BURSA

Oyun analizi günümüzde oldukça rağbet gören bir alandır ve League of Legends e-Spor alanında popüler oyunlardan biridir. Bir MOBA oyunu olan League of Legends aslında ikili sınıflandırma problemi olarak değerlendirilebilir. Bu çalışmada oyunun geliştiricisi tarafından yayınlanan RIOT API kullanarak veri çekilmiş ve çekilen veriler modele beslenmeden önce önişleme tabi tutulmuştur. Veri iki farklı yaklaşımla düzenlenip iki farklı veri kümesi olarak sunulmuştur: oyuncu-tabanlı veri kümesi ve takım-tabanlı veri kümesi. Modeller bu veri kümeleri ile ayrı ayrı eğitilmiş ve sonuçları değerlendirilmiştir. Bu değerlendirme sonucunda performansı en yüksek olan modeller 0,950 ve 0,969 F1-score ile sırasıyla LightGBM ve AdaBoost modelleri olmuştur.

Deneysel çalışma olarak boyut azaltma yöntemlerinden öznitelik seçimi uygulanmış ve performansı en yüksek olan AdaBoost modelinin sunduğu en önemli 10 öznitelik seçilmiştir. Seçilen öznitelikler arasında korelasyon analizi yapılarak korelasyonu en az olan 7 öznitelik ile veri kümeleri filtrelenerek, modeller bu deneysel veri kümeleri ile tekrar eğitilip test edilmiştir. Sonuç olarak performanslarda belirli bir düşüş gözlendiğinden elenen özniteliklerin oyun açısından stratejik bir öneme sahip olduğu ortaya konulmuştur.

**Anahtar Sözcükler:** League of Legends; Oyun Analizi; Makine Öğrenmesi; Sınıflandırma Problemi

# ABSTRACT

Predictive E-Sports Game Analysis Using Machine Learning Approaches

Atakan TUZCU

Computer Engineering Department

İzmir University of Bakırçay, Graduate Education Institute, June 2023

Supervisor: Dr. Okan BURSA

Game analytics are highly demanding topic nowadays and League of Legends is one of the popular games in professional e-sports area. League of Legends is a MOBA game which can be simplified as binary classification problem. This research will retrieve data using RIOT API which is distributed by the developer of the game League of Legends, preprocess the extracted data before feeding the models with it and build several machine learning models using these data. In this research, two different approaches used while creating the dataset: player-based dataset and team-based dataset. Models have been trained and tested separately using these datasets. AdaBoost and LightGBM has the best performance metrics with 0,950 and 0,969 F1-scores respectively.

For experimental study, dimensionality reduction has been operated using feature selection. We used most 10 important features of the most successful model, AdaBoost. Correlation analysis have been done and 7 attributes have been selected to filter and create new experimental datasets. These attributes are key to build gaming strategy. Models have been trained and tested again with the new datasets. Dimension reduction degrade the performances, and this puts out that the dropped attributes are strategic in the game.

**Keywords:** League of Legends; Game Analytics; Machine Learning; Classification Problem

# ACKNOWLEDGMENTS

During my graduate education at İzmir University of Bakırçay, I encountered numerous challenges, such as the Covid-19 pandemic and the resulting isolation. Lectures shifted to a remote format, and I had to juggle my studies with a demanding full-time job, leaving me with very little time. Throughout this challenging journey, I also faced the heavy burden of major depressive disorder. However, I am incredibly grateful to the many individuals who provided unwavering support and contributed to the completion of my degree.

First and foremost, I want to express my heartfelt gratitude to my supervisor, Prof. Dr. Deniz Kılınç, who has been by my side every step of the way. Graduate degrees can lead one into a deep abyss, but Prof. Dr. Deniz Kılınç has not only been a mentor but has shown me the kindness and sincerity of a father figure. He went above and beyond by connecting me with his wife, a psychiatrist, who played a crucial role in helping me overcome my depression. Prof. Dr. Deniz Kılınç's presence in my life extends far beyond his role as a supervisor.

I would also like to extend my deepest appreciation to my family for their unwavering support throughout my research. My mother, Sevgi Tuzcu, and my father, Nadir Tuzcu, stood by my side. Additionally, my brother, Canberk Tuzcu, an electrical and electronics engineer, provided invaluable assistance during my research, for which I am truly grateful.

My dear friend, Merve Rumelli, who was also a graduate student at IZTECH, deserves my immense gratitude. She guided me through challenging coding and analysis tasks and offered unwavering support in both my academic and personal life. I would also like to express my appreciation to my beloved friends İrem İlkehan, İdil Umut, and Gürkan Yıldızcan, who were always there for me when I needed them the most.

Furthermore, I want to acknowledge the invaluable contributions of my friends Emel Gizem Ay and Ayşegül Umay Uçar from İzmir University of Bakırçay, who took on a significant portion of my workload for this research especially when I was at my compulsory military duty. Their support and assistance were vital to the completion of my degree, and I am forever grateful to them.

# STATEMENT OF COMPLIANCE WITH ETHICAL PRINCIPLES AND RULES

I hereby truthfully declare that this thesis is an original work prepared by me; that I have behaved in accordance with the scientific ethical principles and rules throughout the stages of preparation, data collection, analysis, and presentation of my work; that I have cited the sources of all the data and information that could be obtained within the scope of this study, and included these sources in the references section; and that this study has been scanned for plagiarism with Turnitin scientific plagiarism detection program used by İzmir Bakırçay University, and that “it does not have any plagiarism” whatsoever. I also declare that, if a case contrary to my declaration is detected in my work at any time, I hereby express my consent to all the ethical and legal consequences that are involved.

……………..…

Atakan TUZCU

# CONTENTS

**page**

[TITLE PAGE i](#_Toc136527639)

[FINAL APPROVAL FOR THESIS ii](#_Toc136527640)

[ÖZET iii](#_Toc136527641)

[ABSTRACT iv](#_Toc136527642)

[ACKNOWLEDGMENTS v](#_Toc136527643)

[STATEMENT OF COMPLIANCE WITH ETHICAL PRINCIPLES AND RULES vi](#_Toc136527644)

[CONTENTS vii](#_Toc136527645)

[FIGURES ix](#_Toc136527646)

[TABLES x](#_Toc136527647)

[1. INTRODUCTION 1](#_Toc136527648)

[1.1. Problem Definition 2](#_Toc136527649)

[1.2. Purpose of Thesis 2](#_Toc136527650)

[1.3. Methodology 3](#_Toc136527651)

[1.4. Contribution of Thesis 4](#_Toc136527652)

[1.5. Organization of Thesis 4](#_Toc136527653)

[2. BACKGROUND INFORMATION 5](#_Toc136527654)

[2.1. League of Legends 5](#_Toc136527655)

[2.2. Related Works 6](#_Toc136527656)

[3. METHODOLOGY 9](#_Toc136527657)

[3.1. Collecting Data 9](#_Toc136527658)

[3.1.1. Introduction 9](#_Toc136527659)

[3.1.2. RIOT API 10](#_Toc136527660)

[3.1.3. Implementation 10](#_Toc136527661)

[3.1.4. Preprocessing 11](#_Toc136527662)

[3.1.5. Data Schema 12](#_Toc136527663)

[3.2. Machine Learning Approaches 14](#_Toc136527664)

[3.2.1. Logistic Regression 14](#_Toc136527665)

[3.2.2. Naïve Bayes Classifier 15](#_Toc136527666)

[3.2.3. Decision Tree 15](#_Toc136527667)

[3.2.4. Random Forest 16](#_Toc136527668)

[3.2.5. Gradient Boosting 16](#_Toc136527669)

[3.2.6. AdaBoost 16](#_Toc136527670)

[3.2.7. LightGBM 17](#_Toc136527671)

[3.3. Model Evaluation Metrics 17](#_Toc136527672)

[3.3.1. Confusion Matrix 17](#_Toc136527673)

[3.3.2. Precision 18](#_Toc136527674)

[3.3.3. Recall 18](#_Toc136527675)

[3.3.4. Accuracy 18](#_Toc136527676)

[3.3.5. F1-Score 18](#_Toc136527677)

[4. RESULTS 20](#_Toc136527678)

[4.1. Evaluation of Models 20](#_Toc136527679)

[4.2. Evaluation of Datasets 25](#_Toc136527680)

[4.3. Results 26](#_Toc136527681)

[5. EXPERIMENTS 27](#_Toc136527682)

[5.1. Feature Selection 27](#_Toc136527683)

[5.2. Implementation 27](#_Toc136527684)

[5.3. Correlation Between Most Important Features 28](#_Toc136527685)

[5.4. Evaluation of Models with Feature Selection 29](#_Toc136527686)

[5.5. Evaluation of Datasets with Feature Selection 35](#_Toc136527687)

[6. CONCLUSION 37](#_Toc136527688)

[6.1. Summary 37](#_Toc136527689)

[6.2. Discussions 38](#_Toc136527690)

[6.3. Future Works 38](#_Toc136527691)

[REFERENCES 39](#_Toc136527692)

# FIGURES

[Figure 1.1 Diagram of the methodology 3](#_Toc136375604)

[Figure 2.1 Simplified Representation of Summoner’s Rift map. [2] 6](#_Toc136375605)

[Figure 3.1 Flow chart of the implementation of Data Collection phase 11](#_Toc136375606)

[Figure 4.1 Confusion matrices for logistic regression model 20](#_Toc136375607)

[Figure 4.2 Confusion matrices for naïve bayes classifier model 21](#_Toc136375608)

[Figure 4.3 Confusion matrices for decision tree model 22](#_Toc136375609)

[Figure 4.4 Confusion matrices for random forest model 22](#_Toc136375610)

[Figure 4.5 Confusion matrices for gradient boosting model 23](#_Toc136375611)

[Figure 4.6 Confusion matrices for AdaBoost model 24](#_Toc136375612)

[Figure 4.7 Confusion matrices for LightGBM model 24](#_Toc136375613)

[Figure 5.1 10 most important features of AdaBoost model 28](#_Toc136375614)

[Figure 5.2 Correlation matrix of 10 most important features. 29](#_Toc136375615)

[Figure 5.3 Confusion matrices for logistic regression with selected features 30](#_Toc136375616)

[Figure 5.4 Confusion matrices for naïve bayes classifier with selected features 31](#_Toc136375617)

[Figure 5.5 Confusion matrices for decision tree with selected features 31](#_Toc136375618)

[Figure 5.6 Confusion matrices for random forest with selected features 32](#_Toc136375619)

[Figure 5.7 Confusion matrices for gradient boosting with selected features 33](#_Toc136375620)

[Figure 5.8 Confusion matrices for AdaBoost with selected features 34](#_Toc136375621)

[Figure 5.9 Confusion matrices for LightGBM with selected features 34](#_Toc136375622)

# TABLES

[**Table 3.1** Data schema of the dataset 12](#_Toc136527770)

[**Table 3.2** Confusion Matrix 18](#_Toc136527771)

[**Table 4.1** Metrics for logistic regression model. 21](#_Toc136527772)

[**Table 4.2** Metrics for naïve bayes classifier model. 21](#_Toc136527773)

[**Table 4.3** Metrics for decision tree model. 22](#_Toc136527774)

[**Table 4.4** Metrics for random forest model. 23](#_Toc136527775)

[**Table 4.5** Metrics for gradient boosting model. 23](#_Toc136527776)

[**Table 4.6** Metrics for AdaBoost model. 24](#_Toc136527777)

[**Table 4.7** Metrics for LightGBM model. 25](#_Toc136527778)

[**Table 4.8** Metrics for models that are trained with player-based dataset. 25](#_Toc136527779)

[**Table 4.9** Metrics for models that are trained with team-based dataset. 25](#_Toc136527780)

[**Table 5.1** 10 most important features selected from the dataset. 28](#_Toc136527781)

[**Table 5.2** Metrics for logistic regression model with selected features. 30](#_Toc136527782)

[**Table 5.3** Metrics for naïve bayes classifier model with selected features. 31](#_Toc136527783)

[**Table 5.4** Metrics for decision tree model with selected features. 32](#_Toc136527784)

[**Table 5.5** Metrics for random forest model with selected features. 32](#_Toc136527785)

[**Table 5.6** Metrics for gradient boosting model with selected features. 33](#_Toc136527786)

[**Table 5.7** Metrics for AdaBoost model with selected features. 34](#_Toc136527787)

[**Table 5.8** Metrics for LightGBM model with selected features. 35](#_Toc136527788)

[**Table 5.9** Metrics for models trained with selected features of player-based dataset. 35](#_Toc136527789)

[**Table 5.10** Metrics for models trained with selected features of team-based dataset. 36](#_Toc136527790)

# INTRODUCTION

Sports analytics is a fascinating and dynamic field of data analysis that involves extracting valuable insights from match data. Whether it's historical data or real-time information during a match, the outcome information can be extremely useful for a variety of stakeholders, including coaches, players, fans, and bettors. By leveraging this information, game tactics can be tailored to predicted outcomes, and strategies can be honed and refined over time.

With the rise of electronic sports, or e-Sports, we are seeing a new generation of competitive gaming emerge. Counter Strike: Global Offense, League of Legends, DotA2, Warcraft, Heartstone are the examples of these e-Sport games. These games are typically played online, which enables organizers to host tournaments and world championships that attract millions of viewers from around the globe. The most popular types of e-Sports include first-person shooter games, fighting games, multiplayer online battle arena games, and role-playing games. Like traditional sports, e-Sports can be played both individually and as part of a team.

As e-Sports streaming continues to gain popularity, they are also becoming more widely accessible through streaming platforms like YouTube and Twitch, as well as traditional television sports networks like Entertainment and Sports Programming Network (ESPN). Among the many e-Sports categories, Multiplayer Online Battle Arena (MOBA) games are leading the industry. Games like League of Legends (LoL), Defense of the Ancients 2 (DotA 2), Mobile Legends: Bang Bang (MLBB), and League of Legends: Wild Rift (Wild Rift) have captured the attention of millions of players and fans worldwide (A reference needed for the numbers).

Given the explosive growth of e-Sports, research studies are now focusing on this area to better understand its impact and potential. Tournaments generate vast amounts of data over time, and researchers can use this data to develop models and insights that help players and teams improve their performance. Game developers and publishers also provide data through APIs, such as RIOT API[[1]](#footnote-1), which is a free-to-use API with a Representational State Transfer (REST) architecture. In this research study, we have selected League of Legends as our game of focus because it has a larger player base than DotA 2. Using the data provided by RIOT API, we can retrieve information about players and their recent game data, which can be analyzed to provide valuable updates on the game. This is particularly important because the game meta changes frequently in response to new patches and updates.

Overall, e-Sports analytics is an exciting and rapidly evolving field that is poised to have a significant impact on the future of competitive gaming.

## Problem Definition

Binary classification is the task that classifying the instance into two different groups. By the nature of the gameplay, as in most MOBA games, League of Legends ends with one team’s victory and the other team’s defeat. Each action in the game affects the outcome, so the prediction will be based on the end-game dataset. There is no draw status for the end of a League of Legends match final. Thus, this nature comes up with a binary classification problem in which we can predict if a match ends as a win or lose, in other words, victory or defeat.

As each action affects the result of the game, each player is dependent on their teammates. The official Riot API provides data as player-based but if we classify a match according to the result of the game, we can get the team-based data. So, we can apply this classification, we can create two different datasets one of them is player-based, and the other one is team-based. By doing this, we can have two different approaches: team perspective and player perspective. But the problem will remain as if the team or player wins, or if the player or team loses.

## Purpose of Thesis

Every action taken during the game affects the outcome of the game result and this requires a deep analysis of end-game data. This study aims to provide a dataset using the official Riot API, briefly explaining the preprocessing steps over the used dataset, building, and implementing machine learning models to predict the outcome of the match result, and do experimental studies to identify the weights of features to put it out the key factors for developing a better gameplay strategy.

## Methodology

There will be three phases in this study as can be shown in Figure 1.1, which are collecting data, modeling, and evaluation. This research provides a dataset and guide to preprocess the dataset before evaluating models, then implement the models over the clean and preprocessed data to get a deep analysis of a game to extract the key impacts using different machine learning approaches.

Figure 1.1 Diagram of the methodology

In the initial phase of this study, data acquisition and preprocessing of the retrieved data will be the primary focus. To accomplish this task, RIOT API which is the official tool that has been provided by the developer of League of Legends will be used. This tool is built using REST architecture thus basic Hyper-Text Transfer Protocol (HTTP) requests will operate to retrieve data. This API can be found via Riot Game Developer Portal and its main purpose is to give freedom to developers to build their applications using this statistical information.

The second stage is implementing the machine learning models which are logistic regression, Naïve Bayes, Decision Tree, Random Forest, Gradient Boosting, LightGBM, and AdaBoost. Given the nature of the data and the definition of the problem, this is a binary classification task. Python programming language will be used with basic machine learning libraries that are being used by the community. The models will be trained on the data collected in the first phase of this study.

Third and the final phase of this study will focus evaluate and analyze the output of the trained models. Comparison will be done over defined criteria such as F1-score, precision, recall, accuracy. Improvements will be applied according to the results. Gini score-based feature selection method will be applied to the data and observation will be done over the enhanced results. Data visualization will be over the result to make it clear for better understanding.

## Contribution of Thesis

At the output of this research, a brand-new dataset will be presented to any other research that can be done over this dataset, besides finding which approach works better and give better accuracy on this dataset. Also, the preparation of the dataset will be stated in the thesis, thus future research can use this methodology to create their datasets. And a data retrieving tool will be provided to retrieve data easily by giving just a summoner names list.

There are several applications, research, and even a thesis on game result prediction or forecasting using machine learning approach and deep learning approach, whereas no research has compared these approaches at the same time over the game data. Through this research, we put forth which approach is much more available for the structure of the dataset. This thesis, compelling than others by applying the methods that have been mentioned in the methods chapter.

## Organization of Thesis

In Chapter 1, the purpose and some main definitions have been stated. We pointed out the purpose of this research and emphasize the contribution of this research. The rest of this research paper is organized as follows. Chapter 2 will be a literature survey and background information on game analytics and approaches to this topic. Chapter 3 will introduce the methodology and describe the implementation of the models. Chapter 4 will discuss the results of the models and try to do an experimental study of the results to get better outputs. Chapter 5 will summarize the study, discuss the results and give suggestions for future works.

# BACKGROUND INFORMATION

Multiplayer Online Games can be categorized as Role Playing Games, First-Person Shooters, Real-Time Strategies, Battle Arenas. MOBA category is one of the popular in the multiplayer games, and by 2018, the most popular MOBA game is League of Legends (LoL) [1]. Beside LoL, there are other options for MOBA games like DotA2 which is the second popular game in MOBA category for PC gaming, and Mobile Legends: Bang Bang, Vainglory, and League of Legends: Wild Rift, Pokémon UNITE for mobile gaming.

MOBA games structured on a battle arena and two opponent team. Each team made of generally 5 players and tries to destroy opponent base while trying to protect their own base. Each player can select a character from the character list and play as that character until the game ends. Players can select different characters at different games; thus, character proficiency is a metric to play better in MOBAs.

## League of Legends

League of Legends is one of the MOBA games which developed by RIOT Games and released on 27 October 2009. The game is free-to-play, means that skins and other purchasable items in the game does not affect directly on the gameplay but the visual comfort and this can put forward the players’ gaming skills.

Like other MOBA games League of Legends are being played using isometric perspective and players control their character’s movement using mouse clicks and control their character’s abilities using keyboard buttons. As the date of this research study, there are 163 unique champion characters playable, and the game result can be affected by player’s experience on the specific character.

League of Legends World Championships are being done using Summoner’s Rift which is one of the maps are being used in the game. Summoner’s Rift made of basically 3 lanes which are Top, Middle, Bottom and a Jungle between these lanes. Simplified map representation is given in the Figure 2.1 There are 3 turrets in each lane and 2 turrets on the base, at total there are 11 turrets for each team. Players’ main objective is destroying the opponent team’s base “Nexus” which is guarded by players and turrets. Nexus produces minions at the 1:30 minute in the game time and lasts till the end of the game. Each team’s base has 3 inhibitors behind the last turret of the lane and before destroying Nexus, at least one of the inhibitors must be destroyed. Also destroying the opponent’s inhibitor makes player’s Nexus produce stronger minions.

A picture containing text, screenshot, font, colorfulness

Description automatically generated

Figure 2.1 Simplified Representation of Summoner’s Rift map. [2]

In order to win in a team-oriented sports games, the players have to be created in a coordination as a team. However, being a team is difficult in traditional sport games, but it is not that different in e-Sport area, too. There is main objective, to destroying the Nexus, and there are some sub-objectives, such as killing Baron Nashor and killing Dragon to accomplish this task. Being a team is one of the important keys to achieve the goal and win the game, thus the team must be coordinated properly.

This challenging environment of LoL make it a big attraction for the game followers. These competitive gameplays increased the number from 210.000 viewer of 2011 League of Legends World Championships to 200 million in the year 2018 [3].

In this research study, we will try to predict the end-game results of League of Legends matches using several machine learning approaches. In the following title, we will scan the latest and important research and summarize them to have some knowledge about what the other researchers used to accomplish this task.

## Related Works

Wang K. and Shang W. preferred DotA2 as the game subject to predict the game result. They tried to find out the winning probability of each team. They implemented a model using Naïve Bayes classifier and they used it on the DotA2 dataset from the UCI Machine Learning Repository[[2]](#footnote-2). DotA2 is 10 player game like League of Legends, so they improved their application by reducing the dataset according to chosen champions. There were 113 champions that the time they have applied. There were 117 attributes on their dataset and thus, there were 92.649 data in total. For the target attribute, 1 shows the radiant team and -1 shows the dire team as the winner. They have got 85,33% accuracy on their training set and 58,99% accuracy on their test dataset. [4]

Ani R. Et. Al. preferred League of Legends as the game to predict the match result and used ensemble methods to achieve the prediction task. They used a dataset of 1,500 matches and each match was played with 10 as nature of the game thus the dataset includes 15.000 instances and 97 attributes about both pre-match and within-match data. They used four different types of ensemble methods which are Random Forest, AdaBoost, Gradient Boosting and Extreme Gradient Boosting with the accuracies of 99,75%, 96,25%, 97,01% and 97,21% respectively using both pre-match and within-match data combined. [5]

Hodge V. Et. Al. stated that pre-match data are more easily available but commercial value on real-time data is much more important. They had presented that their paper presents the first professional-level prediction for the DotA2 matches. They used two different datasets, one of them contains 186 professional matches and the other one contains 5.744 mixed matches. They had evaluated three different ensemble models which are Linear Regression, Random Forest, and Microsoft’s LightGBM framework. They have got 77.35%, 77.51%, and 77.46 accuracy respectively on the mixed dataset, and got 72.97%, 74.59%, and 73.51% accuracy respectively on the professional dataset. [6]

Chan A.S Et. Al. researched on Mobile Legends Bang Bang as the game to predict and used Naïve Bayes classifier as the prediction model to operate this task. They stated out that they have choose this classifier because it works well on the input with high dimensionality and their results shows that their statement is correct. [7]

Akhmedov K. and Phan A.H. observed the prediction analysis over DotA2 using Linear Regression, Neural Networks and LSTM. They collected their data using Game Tate Integration python server. They were collecting data over 100 matches in real-time, thus the data row for one match corresponds to duration of game. Their data dataset contains 3833 rows that each row is one minuet of the game and 71 attributes which one of them is target attribute. They had split their dataset as 80% for training and %20 for test. At the conclusion of their research, they stated out that linear regression hit the accuracy 82% average, artificial neural network with the accuracy of 88% and LSTM model with the accuracy of 93%. [8]

# METHODOLOGY

This chapter provides an overview of the data collecting phase, preprocessing the collected raw data to feed the models with, analyze the data and introduce the dataset with a data schema to better understanding of data. And the models that can applied over the data to predict the match outcome.

## Collecting Data

### Introduction

Collecting data is the main problem of this thesis. The crucial part was that it is about evaluating the data and knowing the game. Also, there was such a problem what data we should be collected and what time of the game we are going to collect it. Because of the Game patches, data extraction time differs a lot. Also, game patches change the gameplay drastically, and it effects the data directly.

We will use the RIOT API, which is the official tool that RIOT games provided to researchers, as the main tool and will use Python programming language to extract the data from this API. As RIOT API made with the REST architecture, it is familiar to use HTTP requests to get responses and these responses were established as our raw data.

After getting the data, we preprocessed it before giving it to the models to reduce the training time. Some attributes were nested on the data because raw data were made of JSON responses, and we had need to flatten it first. We have used prepared methods and Python libraries to achieve this task. Some attributes were redundant and some of them were important. We had detected that those are important in order to create a new structured dataset. Over this dataset, we had operated some preprocess tasks such as removing the missing values, changing some values to categorical etc.

At last, after we did got the clean data, and we discussed some machine learning methods over the data to guess what we will expect from the results. We had operated experimental studies afterwards about the outputs in the Chapter 4. In this chapter, implementation of the data extraction, data cleaning and implementing machine learning methods will be discussed briefly and we will provide the codes of the implementation over GitHub.

### RIOT API

RIOT is the developer of the game League of Legends and RIOT provides developer portal to extract the game data in a secure and reliable way. RIOT API is built with this perspective and as the time of this research, it provides 27 different APIs to extract data by different ways. In this research we have used the APIs given below:

Summoner-v4: This API provides 6 different GET methods to reach the summoner with different ways. In this research, we used just the one to get a summoner data using just “Summoner Name” and “Region”. Summoner name is the username that is being used in the game and it is not the player’s personal and private data. Region is the server abbreviation that player reaches to play the game. By this method, we will reach summoner data and the data includes the PUUID which is the unique key value to identify the summoner.

Match-v4: This API provides 3 different GET methods to reach the match data. In this research, we used two of them which first of them provides the unique matchIDs using PUUID and the second one provides match data using matchIDs. Second Match-v4 method had provided us 105 attributes as the time of this research.

### Implementation

To get the data of the players who have played on the League of Legends 2022 World Champion ship, we needed the list of the players and their server codes, because professional players have much more accuracy in the game than the regular players. We noted the player names and regions of those players and create an input CSV file which is made of this pre-data[[3]](#footnote-3).

This pre-data was the input of our implementation. Using this input, we did get the PUUIDs, so that later we can reach the matchIDs and, at last, we can use the matchIDs to reach the raw match data. This data was reached using this referencing. To implement this, we used a python library named “riotwatcher” which is a wrapper to use RIOT APIs [[4]](#footnote-4). The RIOT APIs returns data as JSON format and we used “json” library to parse the result. JSON data are being flattened to create and visualize the raw data. For these tasks, we used “pandas” library[[5]](#footnote-5).

Flowchart of the implementation of data collection phase is given in the Figure 3.1 below. This figures clearly shows the inputs and outputs of the methods and the way we used them to extract the data.

A diagram of a api

Description automatically generated with medium confidence

Figure 3.1 Flow chart of the implementation of Data Collection phase

By this implementation we get match info but in JSON file. We did need it to be normalized to flatten as to get reach in the form of data row. Flattening had been done according to participant column to get each player data as different rows. And at the end of data collection phase, this flattened data was saved as CSV file.

### Preprocessing

The final data came from the data collection phase was not clean and needed to be preprocessed before feeding the models with this data. CSV file has been loaded to data-frame using “pandas” library in Python and preprocessing phase have been started.

Missing values has been detected and dropped from the data-frame using “pandas” data manipulation functions over data frames. Sometimes users can create custom game and these kinds of games create “1-player-game” data and these data detected and removed from the data-frame. “platformID” and “gameID” columns have been concatenated and transformed into just on feature as “id”. “win” attribute which implies if the game has won or lost for that player was boolean as True and False, this attribute has been reassigned as True = 1, and False = 0 to make the classification easier. In the end, the clean data was collected with 47 attributes, 1 of them was the target, and 5232 of them were instances.

### Data Schema

After using preprocessing operations, a dataset with 47 attributes and 5232 instances had been collected as output were accepted as clean data. This data is in the form that can be feed to machine learning models. Data schema of this dataset can be seen in the Table 3.1. Grouping by gameId and win attributes was collected as team-based data. At last, we have got two different dataset which one of them is player-based and the other of them is team-based.

**Table 3.1** Data schema of the dataset

| Attribute Name | Explanation | Sample Data |
| --- | --- | --- |
| id | Unique identification number for each game | TR11339009266.0 |
| assists | Score point when player helped another teammate to get kill point | 3.0 |
| basicPings | Signals use counts to communicate with teammates | 21.0 |
| consumablePurchased | Count of the player’ bought consumable items like potions or control ward | 8.0 |
| damageDealtToBuildings | Damage point to buildings like turrets or base. | 784.0 |
| damageDealtToObjectives | Damage point to objectives like drake or Baron Nashor | 5298 |
| deaths | Death score of the player | 6.0 |
| detectorWardsPlaced | Placed control ward count | 1.0 |
| goldEarned | Earned gold amount in the overall gameplay of player | 5638 |
| goldSpent | Spent gold amount in the overall gameplay of player | 5435 |
| individualPosition | Position of the player in the game, like corridor or role. | TOP |
| inhibitorKills | Inhibitor takedowns count of the team | 4.0 |
| inhibitorLost | Inhibitor lost count | 1.0 |
| inhibitorTakedowns | Inhibitor takedowns count of the player | 2.0 |
| itemsPurchased | Purchased items count in the overall gameplay | 26.0 |
| killingSprees | Kill point sequentially in specific time range | 5.0 |
| kills | Kill points individually taken by player | 5.0 |
| largestKillingSpree | Largest killing point in a killing spree | 3.0 |
| longestTimeSpentLiving | Longest time spent as living in ms. | 7083.0 |
| magicDamageDealt | Magic damage to enemy mobs at total | 1378.0 |
| magicDamageDealtToChampions | Magic damage to enemy champions | 12373.0 |
| magicDemageTaken | Magic damage taken from enemy mobs at total | 1412.0 |
| neutralMinionsKilled | Neutral mobs killed like jungle monsters. | 45.0 |
| physicalDamageDealt | Physical damage to enemy mobs at total | 4123.0 |
| physicalDamageDealtToChampions | Physical damage to enemy champions | 3879.0 |
| physicalDamageTaken | Physical damage taken from enemy mobs at total | 23819.0 |
| summonerLevel | Player’s account level | 135.0 |
| timeCCingOthers | Time spent in ms when player stunned an enemy | 87.0 |
| totalDamageDealt | Magical, physical and true damage at total to enemy mobs at total | 62342.0 |
| totalDamageDealtToChampions | Magical, physical and true damage at total to enemy champions at total | 37653.0 |
| totalDamageShieldedOnTeammates | Magical, physical and true damage at total buffered with player’s shield on another teammate | 7231.0 |
| totalDamageTaken | Magical, physical and true damage taken at total | 124128.0 |
| totalHeal | Heals used on self and teammates | 24134.0 |
| totalHealsOnTeammates | Heals used on just teammates | 18041.0 |
| totalMinionsKilled | Minion score | 175.0 |
| totalTimeCCDealt | Time spent on stunned in ms | 85.0 |
| totalTimeSpentDead | Time spent as dead in ms | 246.0 |
| trueDamageDealt | True damage to enemy mobs at total | 1242.0 |
| trueDamageDealtToChampions | True damage to enemy champions | 5427.0 |
| trueDamageTaken | True damage taken from enemy | 3134.0 |
| turretKills | Turrets count that the player’s team takedown | 11.0 |
| turretsLost | Turret count that enemy team takedown | 3.0 |
| visionScore | Score that has been calculated with vision wards and control wards placements and takedowns | 48.0 |
| visionWardsBoughtInGame | Control wards count have been bought in gameplay | 3.0 |
| wardsKilled | Control and vision wards takendown | 13.0 |
| wardsPlaced | Control and vision wards placed | 27.0 |
| win | Target attribute as victory = 1 and defeat = 0 | 1 |

## Machine Learning Approaches

In the collected clean data, the target attribute shows the team or player wins or loses and this nature brings us a binary classification problem. And in this thesis, we decided to use several machine learning approaches to solve this classification which methods are Logistic Regression, Naïve Bayes, Decision Tree, Random Forest, Gradient Boosting, LightGBM, AdaBoost. All these methods are classification methods, and they try to solve the classification problems with different approaches. In this title, these methods will be introduced and define the implementations properly for being a guide to further research.

### Logistic Regression

Regression is a form of machine learning models which can be used for predictions and classifications. Linear regression is mainly used for predicting the continuous targets and logistic regression is mainly used for predicting categorical targets. Linear regression can be simplified as the formula in the Equation 3.1

In this study, we need to predict categorical target classes as win = 1 and lose = 0. Logistic function can be written in the Equation 3.2, as below.

Thus, the logistic regression can be written using linear regression as in the Equation 3.3 where the W is the weights of all descriptive features and D is the array of all the descriptive features.

By this, logistic regression can calculate the probability of a particular objective rather than to calculate the prediction outcome themselves. This feature of logistic regression makes it to produces predictions for categorical target attributes. This makes it to perfect tool to use on binary classification problem, like the problem of this research.

### Naïve Bayes Classifier

Naïve Bayes Classifier is a method uses Bayes’ Theorem which is a probability-based method that can be used for classification. Bayes’ Theorem formula is given in the Equation 3.4 where is the probability, is the hypothesis and is the evidence.

In this study, we are having a binary classification problem with has 2 different classes as target to predict as win = 1 and lose = 0. Which we can write the Bayes’ Theorem of the form in the Equation 3.5 where win is the game result and d1, d2, …dn are the descriptive features.

Descriptive features are kind of an array and can be written as D so the formula for this study will be as in the Equation 3.6.

### Decision Tree

Decision Tree is a supervised machine learning model which can be used for classification and regression. Decision tree is based on the binary tree data structure, and recursively splits the data to determine the effective features from the data and create a flow-chart like structure to make decisions. Effectiveness of the features can be measured with entropy formula. [9]

### Random Forest

Random Forest is an ensemble machine learning methods that combines multiple decision trees and can be used for both regression and classification. Classification task is being done by voting of outputs of the multiple decision trees. Regression task is being done by averaging the outputs of the decision trees. Random sampling and the random feature selection is the key aspects of this method. [10]

### Gradient Boosting

Gradient Boosting is an ensemble machine learning method that uses and combines decision trees and can be used for both classification and regression tasks. The differences of the gradient boosting method from the random tree methods are that the decision trees are independent from each other, and the structure of the models include trees are sequential. Loss function minimized by adding the trees to the model which each tree uses residual error of the previous tree and optimize the error using that. Gradient Boosting equation is given in the Equation 3.7 where is the predicted value of the gradient boosting model, and is the number of trees that the model built with, is the prediction of n-th tree of the model, and is the i-th instance from the dataset. [11]

### AdaBoost

AdaBoost is an ensemble machine learning method that uses decision trees and can be used for both classification and regression tasks. The difference of the AdaBoost is iteratively focus and adjusts the weights of the training instances from dataset. Model assigns higher weights to total of the outputs that are false positives and false negatives, in other words misclassified instances, and assigns lower weights of outputs that are true negative and true positives, in other words correctly classified. At the result model can focus on the instances that are hard to classify and the final prediction is obtained by combining all the decision trees that the model built with. AdaBoost equation is the given in the Equation 3.8 where is the predicted value of the AdaBoost model, and the N is the number of trees that the model built with, is the weight of the n-th tree in the model, is the prediction of n-th tree of the model, and is the i-th instance from the dataset, and classify is the function that convert the sum to the classes of the model. [12]

### LightGBM

LightGBM is a gradient boosting framework that has been presented by Microsoft researchers to designed to be efficient and. It uses Gradient-Based One-Side Sampling algorithm to handle large-scale datasets. It is aimed to present and give a faster way to Gradient Boosting based algorithms. [13]

## Model Evaluation Metrics

### Confusion Matrix

Confusion matrix is a comparison metric calculation method and can be used for to state out the performance of the classification algorithms. [14] In this study, binary classification problem has been put forwarded and several machine learning approaches have been implemented. To compare these machine learning approaches, confusion matrix and its output metrics can be helpful.

Confusion matrix is a two-dimensional matrix which one dimension stands for actual class of the classifier and the other dimension stands for the class that classifier method assigned. For our case, the classes are stands for the victory status for the match end and can be shown as victory = 1 (positive) and defeat = 0 (negative). As evaluation metrics, precision, recall, accuracy, and F1-scores can be calculated using confusion matrix to compare the models’ performance of classifiers.

**Table 3.2** Confusion Matrix

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Predicted Classes | |
| Positive | Negative |
| Actual Classes | Positive | TP | FN |
| Negative | FP | TN |

### Precision

Precision can be defined as the ratio of true positives and the total numbers of positives predicted by the model, also called as *positive predictive value*. The calculation formula of precision is given in the Equation 3.9.

### Recall

Recall can be defined as the ratio of true positives and the total number of relevant samples, also called as sensitivity. The calculation formula of recall is given in the Equation 3.10.

### Accuracy

Accuracy can be defined as the ratio of true classified samples by the predictive model and the total instances. The calculation formula of accuracy is given in the Equation 3.11.

### F1-Score

F1- score is the harmonic mean of precision and recall, as such it lies between precision and recall, but it is closer to the smaller one of these two value. While accuracy takes the problem in just one perspective, F1-score uses both precision and recall calculating and creating a new different perspective to the success of model. Thus, we will select F1-score to evaluation the models and the formula of F1-Score is given in the Equation 3.12.

# RESULTS

In this research, we have focused to make prediction of match results of the game League of Legends. Several machine learning approaches have been implemented to accomplish this task which are Logistic Regression, Naïve Bayes, Decision Tree, Random Forest, Gradient Boosting, AdaBoost, and LightGBM. All these approaches can be used for classification and our problem was the binary classification problem, as predicting if a player or team wins or loses.

## Evaluation of Models

In this study, League of Legends match data have been extracted and cleaned properly. The models have been implemented over the two datasets: team-based and player-based. For the results of the models, confusion matrices have been created. Confusion matrices also produces several metrics as precision, recall, accuracy, and F1-score. In the previous chapter, computation of these metrics has been introduced. In this chapter, confusion matrices is shown, and the results are compared using these metrics.

Using the confusion matrices of logistic regression model trained with player-based data and team-based data, given in the Figure 4.1, comparison metrics can be calculated.

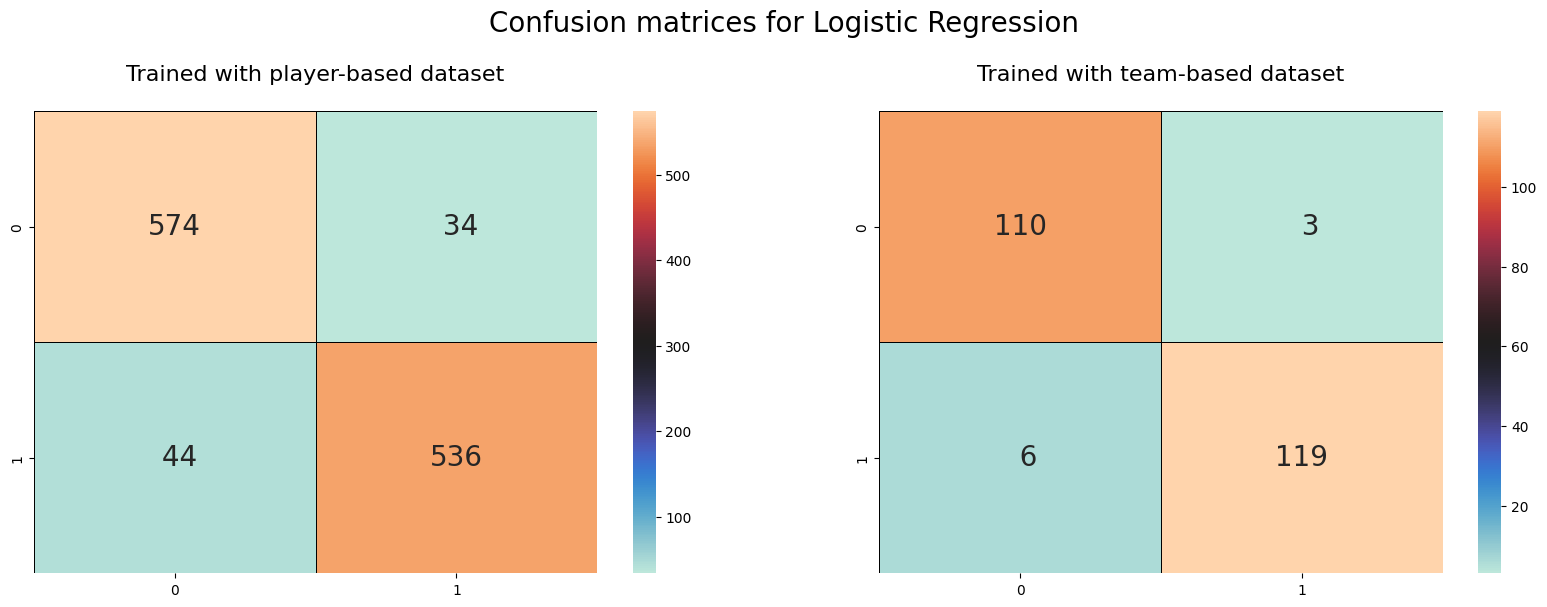


Figure 4.1 Confusion matrices for logistic regression model

Calculated metrics are given in the Table 4.2 to make ease the comparison of datasets over the logistic regression model. Table 4.2 shows that logistic regression model worked better on the team-based dataset.

**Table 4.1** Metrics for logistic regression model.

| **Dataset** | **Precision** | **Recall** | **Accuracy** | **F1-Score** |
| --- | --- | --- | --- | --- |
| Player Based Data | 0.929 | 0.944 | 0.934 | 0.936 |
| Team Based Data | 0.948 | 0.973 | 0.962 | 0.960 |

Using the confusion matrices of naïve bayes classifier model trained with player-based data and team-based data, given in the Figure 4.2, comparison metrics can be calculated.

A picture containing text, screenshot, diagram, rectangle

Description automatically generated

Figure 4.2 Confusion matrices for naïve bayes classifier model

Calculated metrics are given in the Table 4.3 to make ease the comparison of datasets over the naïve bayes classifier model. Table 4.3 shows that naïve bayes classifier model worked better on team-based dataset.

**Table 4.2** Metrics for naïve bayes classifier model.

| **Dataset** | **Precision** | **Recall** | **Accuracy** | **F1-Score** |
| --- | --- | --- | --- | --- |
| Player Based Data | 0.758 | 0.896 | 0.801 | 0.821 |
| Team Based Data | 0.846 | 0.876 | 0.866 | 0.861 |

Using the confusion matrices of decision tree model trained with player-based data and team-based data, given in the Figure 4.3, comparison metrics can be calculated.

A picture containing screenshot, text, diagram, line

Description automatically generated

Figure 4.3 Confusion matrices for decision tree model

Calculated metrics are given in the Table 4.4 to make ease the comparison of datasets over the decision tree model. Table 4.4 shows that decision tree classifier model worked better on team-based dataset.

**Table 4.3** Metrics for decision tree model.

| **Dataset** | **Precision** | **Recall** | **Accuracy** | **F1-Score** |
| --- | --- | --- | --- | --- |
| Player Based Data | 0.931 | 0.933 | 0.93 | 0.932 |
| Team Based Data | 0.963 | 0.929 | 0.95 | 0.946 |

Using the confusion matrices of random forest model trained with player-based data and team-based data, given in the Figure 4.4, comparison metrics can be calculated.

A picture containing screenshot, text, diagram, line

Description automatically generated

Figure 4.4 Confusion matrices for random forest model

Calculated metrics are given in the Table 4.5 to make ease the comparison of datasets over the random forest model. Table 4.5 shows that random forest model worked better on team-based dataset.

**Table 4.4** Metrics for random forest model.

| **Dataset** | **Precision** | **Recall** | **Accuracy** | **F1-Score** |
| --- | --- | --- | --- | --- |
| Player Based Data | 0.929 | 0.964 | 0.944 | 0.946 |
| Team Based Data | 0.948 | 0.965 | 0.958 | 0.956 |

Using the confusion matrices of gradient boosting model trained with player-based data and team-based data, given in the Figure 4.5, comparison metrics can be calculated.

A picture containing screenshot, text, diagram, rectangle

Description automatically generated

Figure 4.5 Confusion matrices for gradient boosting model

Calculated metrics are given in the Table 4.6 to make ease the comparison of datasets over the gradient boosting model. Table 4.6 shows that gradient boosting model worked better on team-based dataset.

**Table 4.5** Metrics for gradient boosting model.

| **Dataset** | **Precision** | **Recall** | **Accuracy** | **F1-Score** |
| --- | --- | --- | --- | --- |
| Player Based Data | 0.919 | 0.951 | 0.932 | 0.935 |
| Team Based Data | 0.964 | 0.947 | 0.958 | 0.955 |

Using the confusion matrices of AdaBoost model trained with player-based data and team-based data, given in the Figure 4.6, comparison metrics can be calculated.

A picture containing screenshot, text, diagram, line

Description automatically generated

Figure 4.6 Confusion matrices for AdaBoost model

Calculated metrics are given in the Table 4.7 to make ease the comparison of datasets over the AdaBoost model. Table 4.7 shows that AdaBoost model worked better on team-based dataset.

**Table 4.6** Metrics for AdaBoost model.

| **Dataset** | **Precision** | **Recall** | **Accuracy** | **F1-Score** |
| --- | --- | --- | --- | --- |
| Player Based Data | 0.934 | 0.961 | 0.945 | 0.947 |
| Team Based Data | 0.973 | 0.965 | 0.971 | 0.969 |

Using the confusion matrices of LightGBM model trained with player-based data and team-based data, given in the Figure 4.7, comparison metrics can be calculated.

A picture containing screenshot, text, diagram, rectangle

Description automatically generated

Figure 4.7 Confusion matrices for LightGBM model

Calculated metrics are given in the Table 4.7 to make ease the comparison of datasets over the LightGBM model. Table 4.7 shows that LightGBM model worked better on team-based dataset.

**Table 4.7** Metrics for LightGBM model.

| Dataset | Precision | Recall | Accuracy | F1-Score |
| --- | --- | --- | --- | --- |
| Player Based Data | 0.935 | 0.965 | 0.948 | 0.950 |
| Team Based Data | 0.965 | 0.965 | 0.966 | 0.965 |

## Evaluation of Datasets

In this study, game-end match data have been retrieved and preprocessed. When the data retrieved, it was processed as to create two different approached datasets. One of them was player-based dataset and the other of them was team-based dataset. Team based dataset has been created by grouping and summing the team members attributes to reach the total team attributes.

Machine learning models have been implemented using these two datasets. We used precision, recall, accuracy, and F1-score for the evaluation metrics for the models. To compare the models by dataset-based we will separate the output tables by dataset.

In the Table 4.9, the results of the models those were trained with player-based dataset are displayed. LightGBM model get the highest F1-score model that were trained with player-based dataset.

**Table 4.8** Metrics for models that are trained with player-based dataset.

| **Model Name** | **Precision** | **Recall** | **Accuracy** | **F1-Score** |
| --- | --- | --- | --- | --- |
| AdaBoost | 0.934 | 0.961 | 0.945 | 0.947 |
| Decision Tree | 0.931 | 0.933 | 0.930 | 0.932 |
| Gradient Boosting | 0.919 | 0.951 | 0.932 | 0.935 |
| LightGBM | 0.935 | 0.965 | 0.948 | **0.950** |
| Logistic Regression | 0.929 | 0.944 | 0.934 | 0.936 |
| Naive Bayes Classifier | 0.758 | 0.896 | 0.801 | 0.821 |
| Random Forest | 0.929 | 0.964 | 0.944 | 0.946 |

In the Table 4.10, results of the models those were trained with player-based dataset are displayed. AdaBoost model get the highest F1-score model that were trained with player-based dataset.

**Table 4.9** Metrics for models that are trained with team-based dataset.

| Model Name | Precision | Recall | Accuracy | F1-Score |
| --- | --- | --- | --- | --- |
| AdaBoost | 0.973 | 0.965 | 0.971 | **0.969** |
| Decision Tree | 0.963 | 0.929 | 0.950 | 0.946 |
| Gradient Boosting | 0.964 | 0.947 | 0.958 | 0.955 |
| LightGBM | 0.965 | 0.965 | 0.966 | 0.965 |
| Logistic Regression | 0.948 | 0.973 | 0.962 | 0.960 |
| Naive Bayes Classifier | 0.846 | 0.876 | 0.866 | 0.861 |
| Random Forest | 0.948 | 0.965 | 0.958 | 0.956 |

## Results

In this study, several machine learning models which are logistic regression, naïve bayes classifier, decision tree, random forest, gradient boosting, AdaBoost, LightGBM models have been implemented and trained with two different approached datasets.

Results show that player-based dataset gave the better performance output when we compare the models using F1- score. Also, we separated the dataset and compare the models. For the player-based dataset, LightGBM model has the better performance with 0,950 F1-score and the naïve bayes classifier model has the worst performance with 0.821 F1-score. For the team-based dataset, AdaBoost model has the better performance with 0,969 F1-score and again, the naïve bayes classifier model has the worst performance with 0,861 F1-score.

# EXPERIMENTS

In this research, we focused to predict the match result using the game-end data and implement several machine learning models over these data. Data has been retrieved from RIOT API which is the developer the game League of Legends and this ensure raw data which is not processed. While retrieving the raw data, some filters have been applied over, then some preprocessing operations have been done. At the result, we had the clean data as the form of in the Table 3.1 which is displaying the data schema.

Clean data consist of 47 attributes. In the clean dataset there were 5940 instances, which is the player-based dataset. When grouping the players on the same team, second dataset have been retrieved with the same attribute count, 47 attributes and 1188 instances. Machine learning models have been implemented and results shows that the LightGBM model works best with 0,950 F1-score for the player-based dataset and AdaBoost models works best with 0,969 F1- score for the team-based dataset.

In this chapter, we will focus and implement some experiments on the datasets and models to get better results (REWRITE). Because of the both datasets share the same attributes, the highest F1-score belongs to the AdaBoost model. We used AdaBoost model and its attributes for experiments.

## Feature Selection

Complexity of a classification or regression model is related to the attributes of input data. Feature selection is a method that is being used to create a subset from a dataset to reduce the dimensions. Using feature selection, the aim is to reduce the running time by creating a simpler dataset. [15]

## Implementation

AdaBoost had the best performance with the highest F1-score on the model training phase, that was the reason we did selected this model’s feature importance. 10 most important attributes which is displaying in the Figure 5.1, have been selected and the dataset was filtered with these attributes.

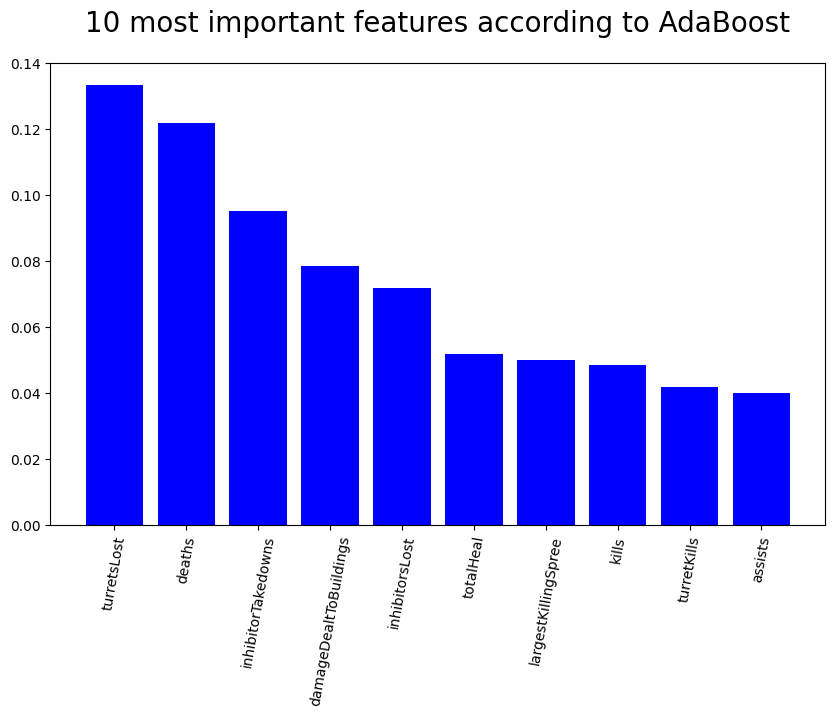


Figure 5.1 10 most important features of AdaBoost model

As the result of the feature selection, selected attributes are given in the Table 5.1 as a subset of Table 3.1.

**Table 5.1** 10 most important features selected from the dataset.

| Attribute Name | Explanation | Sample Data |
| --- | --- | --- |
| id | Unique identification number for each game | TR11339009266.0 |
| assists | Score point when player helped another teammate to get kill point | 3.0 |
| damageDealtToBuildings | Damage point to buildings like turrets or base. | 784.0 |
| deaths | Death score of the player | 6.0 |
| inhibitorLost | Inhibitor lost count | 1.0 |
| inhibitorTakedowns | Inhibitor takedowns count of the player | 2.0 |
| kills | Kill points individually taken by player | 5.0 |
| largestKillingSpree | Largest killing point in a killing spree | 3.0 |
| totalHeal | Heals used on self and teammates | 24134.0 |
| turretKills | Turrets count that the player’s team takedown | 11.0 |

## Correlation Between Most Important Features

Correlation between important features have been analyzed and the correlation matrix is given in the Figure 5.2. Low correlation between attributes displayed with the colder color and the high correlation between attributes displayed with the warmer color.

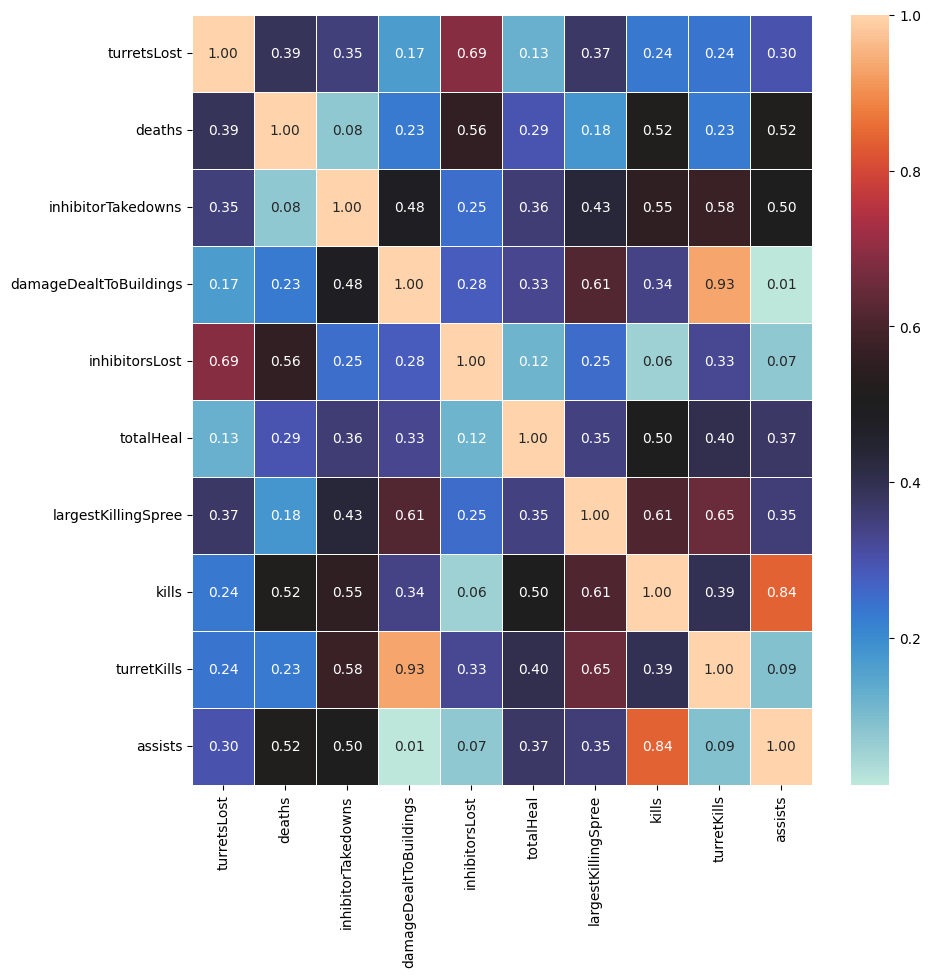


Figure 5.2 Correlation matrix of 10 most important features.

As a threshold for the correlation, we have selected as 0,65 correlation. Highly correlated attributes which are “inhibitorsLost, turretKills, assists” eliminated, and feature selection has been done with the experimental data schema with 7 attributes. This data schema has been applied to player-based and team-based datasets to preprocess the experimental study.

## Evaluation of Models with Feature Selection

Feature selection was done to create a subset of the dataset, models have been implemented with the selected features and the outputs are analyzed using the evaluation criteria as it was being done on the Chapter 4.

Using the confusion matrices of logistic regression model trained with selected features of player-based data and team-based data, given in the Figure 5.3, comparison metrics can be calculated.

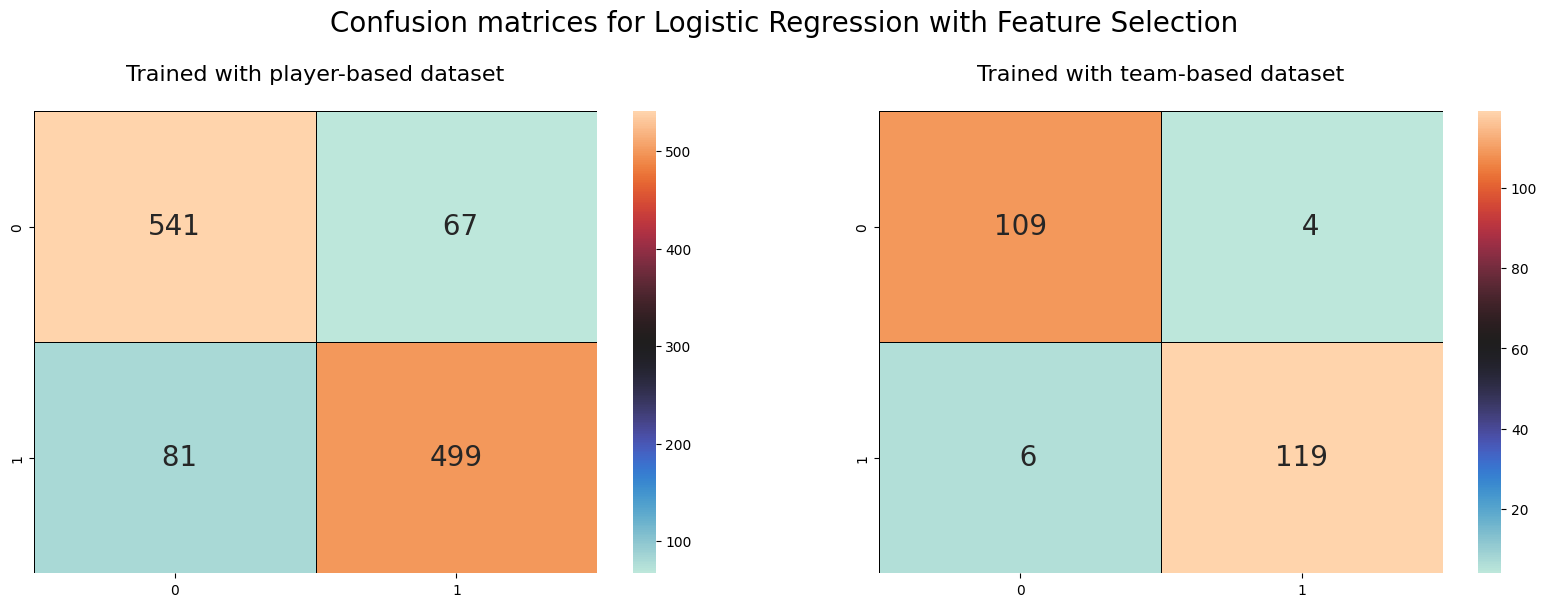


Figure 5.3 Confusion matrices for logistic regression with selected features

Calculated metrics are given in the Table 5.2 to make ease the comparison of datasets over the logistic regression model. Table 5.2 shows that feature selection was not effective on the logistic regression model.

**Table 5.2** Metrics for logistic regression model with selected features.

| **Dataset** | **Precision** | **Recall** | **Accuracy** | **F1-Score** |
| --- | --- | --- | --- | --- |
| Player Based Data | 0.929 | 0.944 | 0.934 | 0.936 |
| Team Based Data | 0.948 | 0.973 | 0.962 | 0.960 |
| Player Based Data with Feature Selection | 0.870 | 0.890 | 0.875 | 0.880 |
| Team Based Data with Feature Selection | 0.948 | 0.965 | 0.958 | 0.956 |

Using the confusion matrices of naïve bayes classifier model trained with selected features of player-based data and team-based data, given in the Figure 5.4, comparison metrics was calculated.

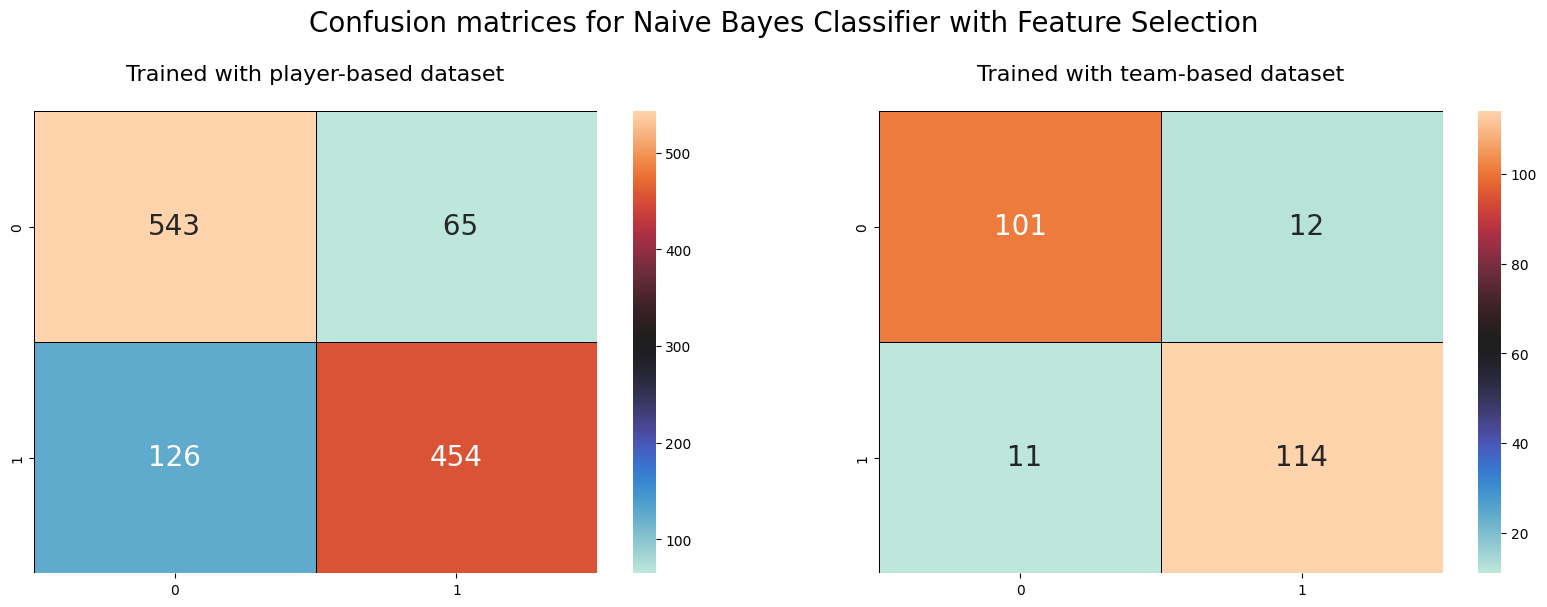


Figure 5.4 Confusion matrices for naïve bayes classifier with selected features

Calculated metrics are given in the Table 5.3 to make ease the comparison of datasets over the naïve bayes classifier model. Table 5.3 shows that feature selection was a little effective on the naïve bayes classifier model.

**Table 5.3** Metrics for naïve bayes classifier model with selected features.

| **Dataset** | **Precision** | **Recall** | **Accuracy** | **F1-Score** |
| --- | --- | --- | --- | --- |
| Player Based Data | 0.758 | 0.896 | 0.801 | 0.821 |
| Team Based Data | 0.846 | 0.876 | 0.866 | 0.861 |
| Player Based Data with Feature Selection | 0.812 | 0.893 | 0.839 | 0.851 |
| Team Based Data with Feature Selection | 0.902 | 0.894 | 0.903 | 0.898 |

Using the confusion matrices of decision tree model trained with selected features of player-based data and team-based data, given in the Figure 5.5, comparison metrics was calculated.

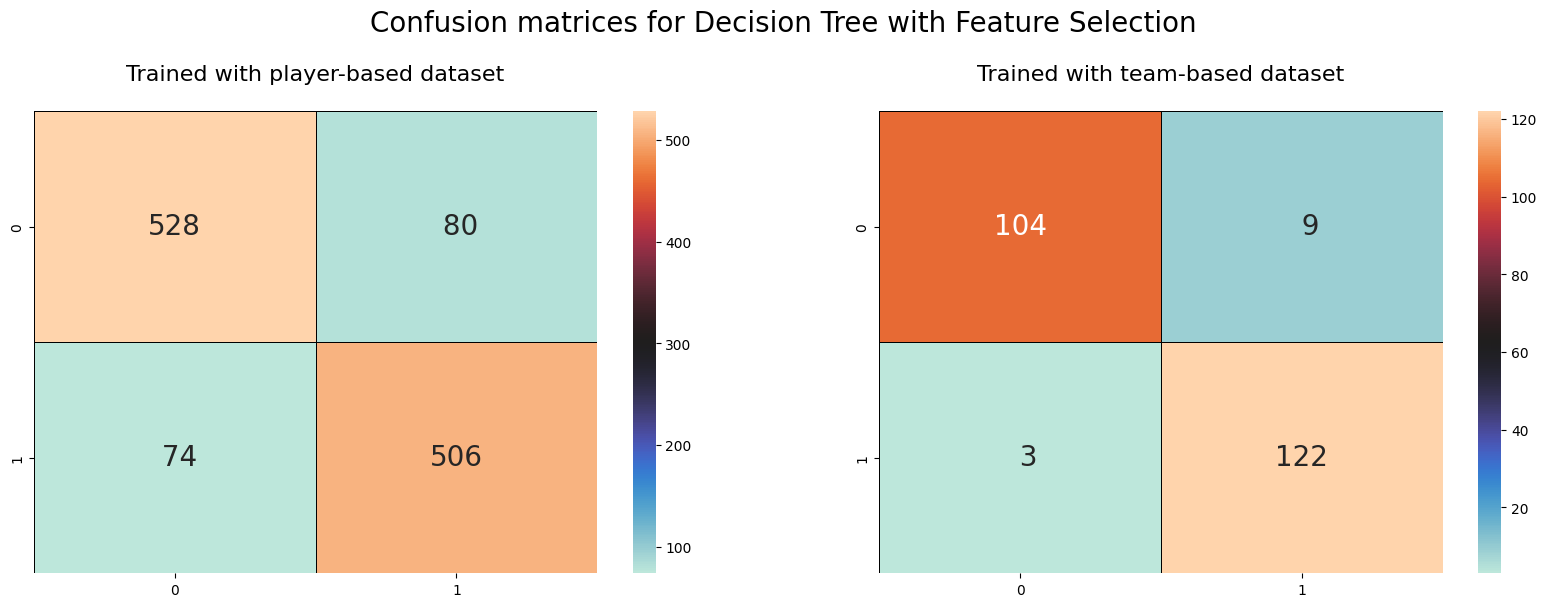


Figure 5.5 Confusion matrices for decision tree with selected features

Calculated metrics are given in the Table 5.4 to make ease the comparison of datasets over the decision tree model. Table 5.4 shows that feature selection was not effective on the decision tree model.

**Table 5.4** Metrics for decision tree model with selected features.

| **Dataset** | **Precision** | **Recall** | **Accuracy** | **F1-Score** |
| --- | --- | --- | --- | --- |
| Player Based Data | 0.931 | 0.933 | 0.93 | 0.932 |
| Team Based Data | 0.963 | 0.929 | 0.95 | 0.946 |
| Player Based Data with Feature Selection | 0.877 | 0.868 | 0.87 | 0.872 |
| Team Based Data with Feature Selection | 0.972 | 0.920 | 0.95 | 0.945 |

Using the confusion matrices of random forest model trained with selected features of player-based data and team-based data, given in the Figure 5.6, comparison metrics was calculated.

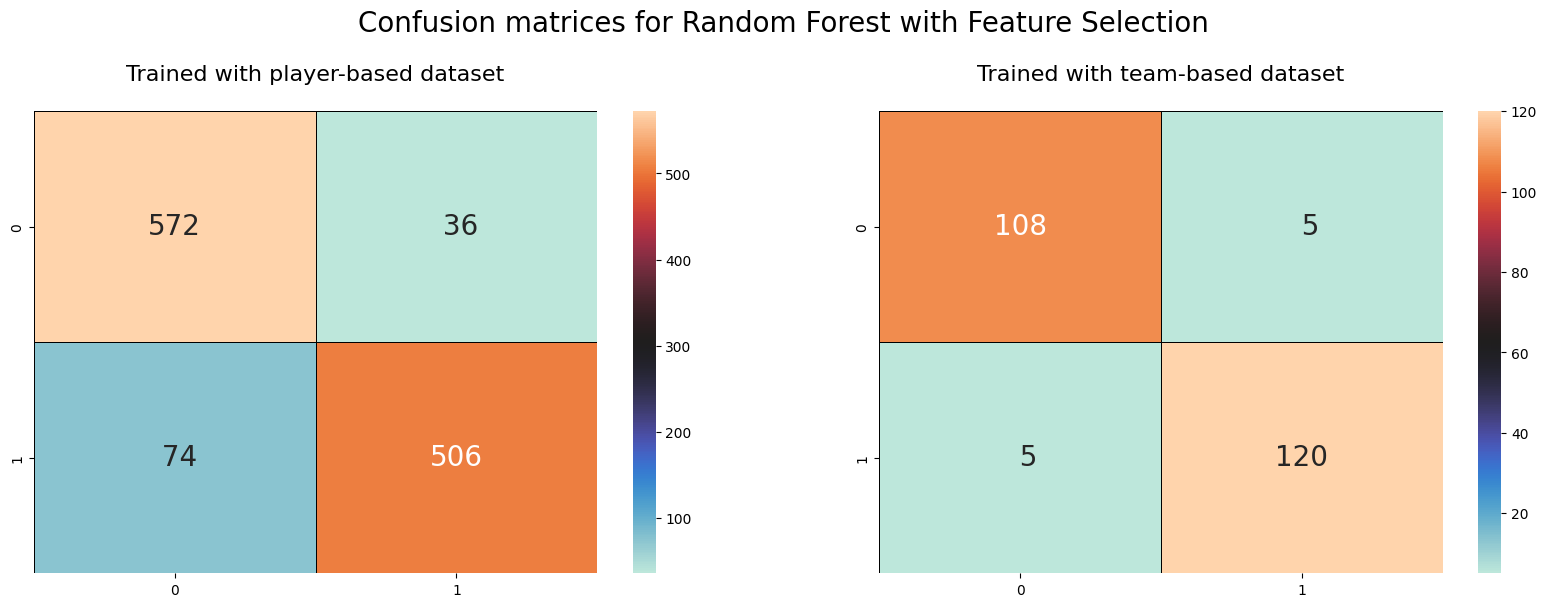


Figure 5.6 Confusion matrices for random forest with selected features

Calculated metrics are given in the Table 5.5 to make ease the comparison of datasets over the random forest model. Table 5.5 shows that feature selection was not effective on the random forest model.

**Table 5.5** Metrics for random forest model with selected features.

| **Dataset** | **Precision** | **Recall** | **Accuracy** | **F1-Score** |
| --- | --- | --- | --- | --- |
| Player Based Data | 0.929 | 0.964 | 0.944 | 0.946 |
| Team Based Data | 0.948 | 0.965 | 0.958 | 0.956 |
| Player Based Data with Feature Selection | 0.885 | 0.941 | 0.907 | 0.912 |
| Team Based Data with Feature Selection | 0.956 | 0.956 | 0.958 | 0.956 |

Using the confusion matrices of gradient boosting model trained with selected features of player-based data and team-based data, given in the Figure 5.7, comparison metrics was calculated.

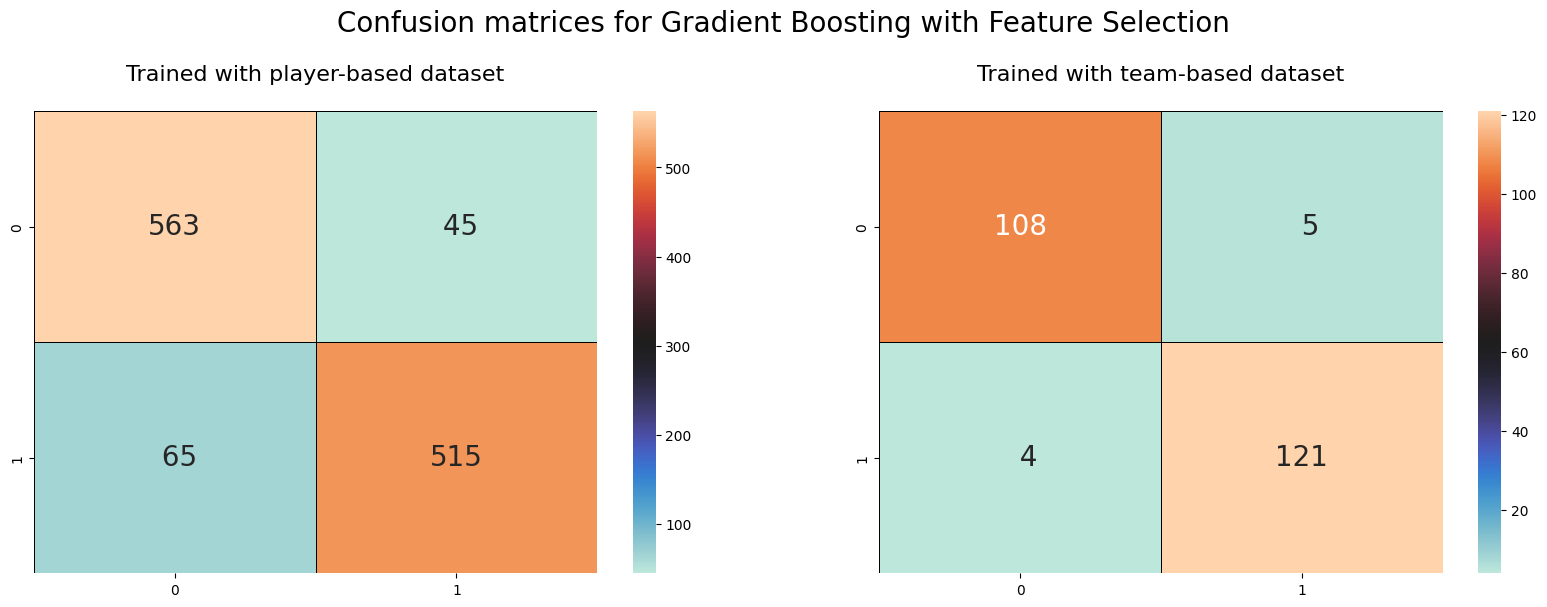


Figure 5.7 Confusion matrices for gradient boosting with selected features

Calculated metrics are given in the Table 5.6 to make ease the comparison of datasets over the gradient boosting model. Table 5.6 shows that feature selection was not effective on the gradient boosting model.

**Table 5.6** Metrics for gradient boosting model with selected features.

| **Dataset** | **Precision** | **Recall** | **Accuracy** | **F1-Score** |
| --- | --- | --- | --- | --- |
| Player Based Data | 0.919 | 0.951 | 0.932 | 0.935 |
| Team Based Data | 0.964 | 0.947 | 0.958 | 0.955 |
| Player Based Data with Feature Selection | 0.896 | 0.926 | 0.907 | 0.911 |
| Team Based Data with Feature Selection | 0.964 | 0.956 | 0.962 | 0.960 |

Using the confusion matrices of AdaBoost model trained with selected features of player-based data and team-based data, given in the Figure 5.8, comparison metrics was calculated.

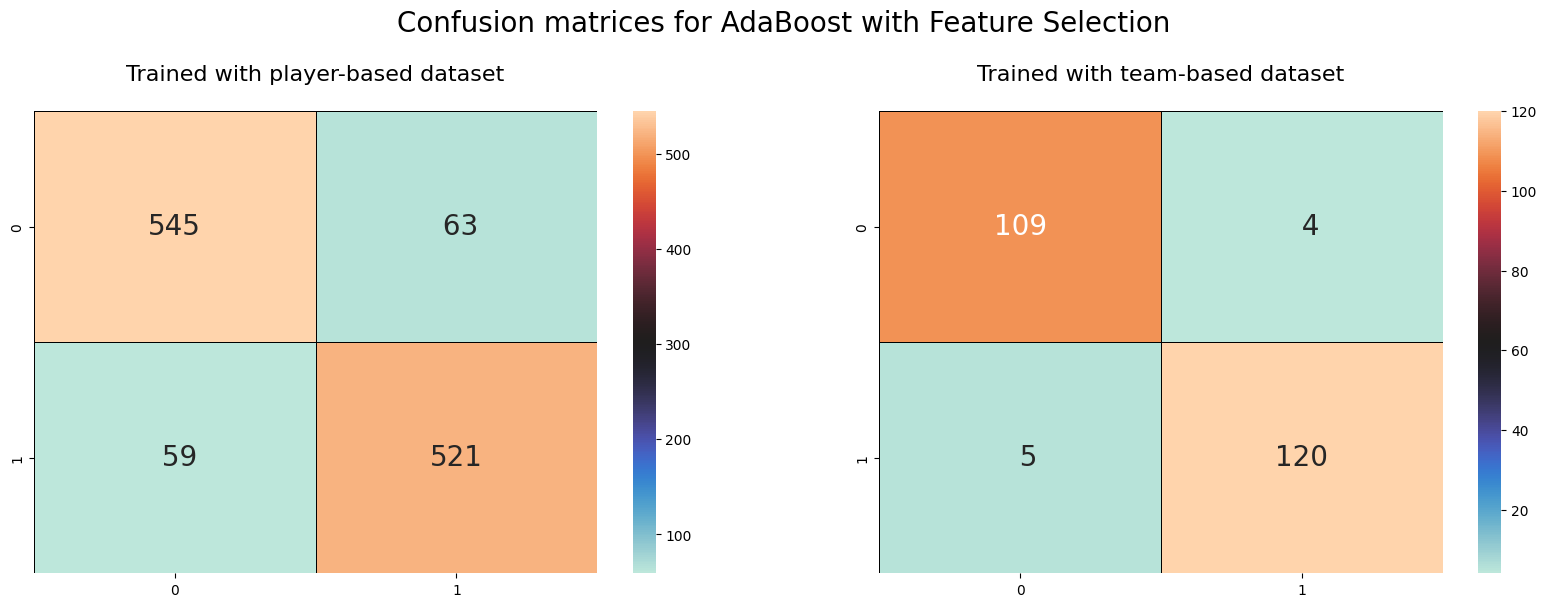


Figure 5.8 Confusion matrices for AdaBoost with selected features

Calculated metrics are given in the Table 5.7 to make ease the comparison of datasets over the random forest model. Table 5.7 shows that feature selection was not effective on the random forest model and made negative effect on player-based dataset.

**Table 5.7** Metrics for AdaBoost model with selected features.

| **Dataset** | **Precision** | **Recall** | **Accuracy** | **F1-Score** |
| --- | --- | --- | --- | --- |
| Player Based Data | 0.934 | 0.961 | 0.945 | 0.947 |
| Team Based Data | 0.973 | 0.965 | 0.971 | 0.969 |
| Player Based Data with Feature Selection | 0.902 | 0.896 | 0.897 | 0.899 |
| Team Based Data with Feature Selection | 0.956 | 0.965 | 0.962 | 0.960 |

Using the confusion matrices of AdaBoost model trained with selected features of player-based data and team-based data, given in the Figure 5.9, comparison metrics was calculated.

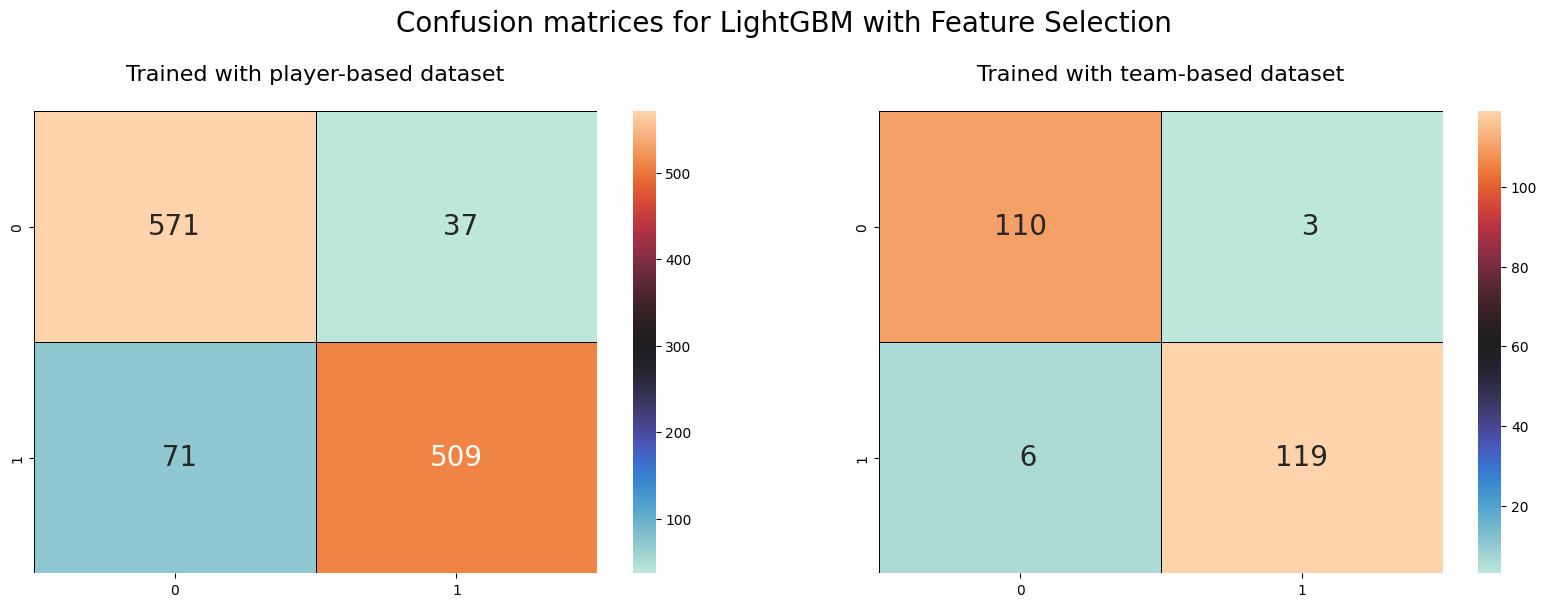


Figure 5.9 Confusion matrices for LightGBM with selected features

Calculated metrics are given in the Table 5.8 to make ease the comparison of datasets over the LightGBM model. Table 5.8 shows that feature selection was not effective on the LightGBM model.

**Table 5.8** Metrics for LightGBM model with selected features.

| **Dataset** | **Precision** | **Recall** | **Accuracy** | **F1-Score** |
| --- | --- | --- | --- | --- |
| Player Based Data | 0.935 | 0.965 | 0.948 | 0.950 |
| Team Based Data | 0.965 | 0.965 | 0.966 | 0.965 |
| Player Based Data with Feature Selection | 0.935 | 0.965 | 0.948 | 0.950 |
| Team Based Data with Feature Selection | 0.965 | 0.965 | 0.966 | 0.965 |

## Evaluation of Datasets with Feature Selection

In this study, when we retrieved and reprocessed the data, we had two different approached datasets: player-based dataset and team-based dataset. AdaBoost model has the better performance with the 0,969 F1-score when trained with team-based dataset. For this model, dimension reduction has been performed as the feature selection and attributes were diminished from 47 attributes to 7 attributes. This new schema was applied to both datasets.

In the Table 5.9, models that were trained with selected features of player-based dataset and the results of the models are displaying with the models trained with the dataset before experiments. Just naïve bayes classifier results are increased compared to the F1-scores. All other models stayed same or decreased by the F1-score.

**Table 5.9** Metrics for models trained with selected features of player-based dataset.

| **Model** | **Precision** | **Recall** | **Accuracy** | **F1-Score** |
| --- | --- | --- | --- | --- |
| AdaBoost | 0.934 | 0.961 | 0.945 | **0.947** |
| AdaBoost with Feature Selection | 0.902 | 0.896 | 0.897 | 0.899 |
| Decision Tree | 0.931 | 0.933 | 0.930 | **0.932** |
| Decision Tree with Feature Selection | 0.877 | 0.868 | 0.870 | 0.872 |
| Gradient Boosting | 0.919 | 0.951 | 0.932 | **0.935** |
| Gradient Boosting with Feature Selection | 0.896 | 0.926 | 0.907 | 0.911 |
| LightGBM | 0.935 | 0.965 | 0.948 | **0.950** |
| LightGBM with Feature Selection | 0.935 | 0.965 | 0.948 | 0.950 |
| Logistic Regression | 0.929 | 0.944 | 0.934 | **0.936** |
| Logistic Regression with Feature Selection | 0.870 | 0.890 | 0.875 | 0.880 |
| Naive Bayes Classifier | 0.758 | 0.896 | 0.801 | 0.821 |
| Naive Bayes Classifier with Feature Selection | 0.812 | 0.893 | 0.839 | **0.851** |
| Random Forest | 0.929 | 0.964 | 0.944 | **0.946** |
| Random Forest with Feature Selection | 0.885 | 0.941 | 0.907 | 0.912 |

In the Table 5.10, models that are trained with selected features of team-based dataset and the results of the models are displaying with the models trained with the dataset before experiments.

**Table 5.10** Metrics for models trained with selected features of team-based dataset.

| **Model** | **Precision** | **Recall** | **Accuracy** | **F1-Score** |
| --- | --- | --- | --- | --- |
| AdaBoost | 0.934 | 0.961 | 0.945 | **0.947** |
| AdaBoost with Feature Selection | 0.902 | 0.896 | 0.897 | 0.899 |
| Decision Tree | 0.931 | 0.933 | 0.930 | **0.932** |
| Decision Tree with Feature Selection | 0.877 | 0.868 | 0.870 | 0.872 |
| Gradient Boosting | 0.919 | 0.951 | 0.932 | **0.935** |
| Gradient Boosting with Feature Selection | 0.896 | 0.926 | 0.907 | 0.911 |
| LightGBM | 0.935 | 0.965 | 0.948 | **0.950** |
| LightGBM with Feature Selection | 0.935 | 0.965 | 0.948 | 0.950 |
| Logistic Regression | 0.929 | 0.944 | 0.934 | **0.936** |
| Logistic Regression with Feature Selection | 0.870 | 0.890 | 0.875 | 0.880 |
| Naive Bayes Classifier | 0.758 | 0.896 | 0.801 | 0.821 |
| Naive Bayes Classifier with Feature Selection | 0.812 | 0.893 | 0.839 | **0.851** |
| Random Forest | 0.929 | 0.964 | 0.944 | **0.946** |
| Random Forest with Feature Selection | 0.885 | 0.941 | 0.907 | 0.912 |

# CONCLUSION

## Summary

In this research, data have been retrieved from the RIOT API for the game “League of Legends”. There was one raw dataset, after preprocessing operation two different approached dataset have been extracted: player-based dataset and team-based dataset. Also, the data extraction tool is published on the GitHub[[6]](#footnote-6).

These two datasets have been used to build several machine learning models which are logistic regression, naïve bayes classifier, decision tree, random forest, gradient boosting, AdaBoost, LightGBM. Model metrics have been calculated using confusion matrices for datasets and used to compare of their successes. Before experimental studies, for the player-based dataset, LightGBM was the best performance model with 0,950 F1-score and naïve bayes classifier was the worst performance model with 0,821 F1-score; for the team-based dataset, AdaBoost was the best performance model with 0,969 F1-score and naïve bayes classifier model was the worst performance model with 0,861 F1-score.

Furthermore, some experimental studies have been done over the datasets. Dimension reduction was applied as feature selection. Since the AdaBoost model that trained with the team-based dataset gave the best performance, important features have been selected using this model as N=10, so 10 important features have been retrieved. Then correlation analysis has been done and more than 0,65 correlated attributes were eliminated. There were 7 attributes left and this filter applied both datasets to create experimental datasets. Also, we can have an idea about what strategies that players should consider while aiming to win a match.

Machine learning models have been trained with new experimental datasets and results have been pointed out that feature selection was not very effective on this data except for the naïve bayes classifier model. Naïve bayes classifier had 0,821 F1-score with player-based dataset and 0,861 F1-score with team-based dataset. After feature selection, naïve bayes classifier model gets 0,851 F1-score with the player-based dataset and 0,851 F1-score with team-based dataset.

## Discussions

In this research, we have retrieved and preprocess data of League of Legends, build machine learning models using these data and evaluate performances comparatively using metrics. Experiments have been operated on these datasets which is dimensionality reduction using feature selection and AdaBoost model used to perform this task. Most Important 10 attributes have been selected and correlation have been analyzed among these attributes. Result of correlation analysis 7 attributes has been selected and models have been trained and tested again with experiment dataset.

Output of the AdaBoost model, 10 important attributes have been put forward which are “turretsLost, deaths, inhibitorTakedowns, damageDealtToBuildings, inhibitorsLost, totalHeal, largestKillingSpree, kills, turretKills, assists” are the key factors for building a strategy in gameplay. As in the experimental study, elimination the attributes “inhibitorsLost, turretKills, assists” degraded the performance of the models so it can be said that these attributes can be used strategically, too.

## Future Works

In this study, several machine learning algorithms have been used which are logistic regression, naïve bayes classification, decision tree, random forest, gradient boosting, AdaBoost, LightGBM models. Some other machine learning approaches like SVM, or some deep learning models can be applied for the future works. Also, data span can be enlarged as we eliminated many attributes while retrieving the data.

RIOT API provides game data by timeline. As a future work, the problem can be evaluated as time-series analysis and build some LSTM models and compare the performances.

# REFERENCES

|  |  |
| --- | --- |
| [1] | M. Mora-Cantallops and M.-Á. Sicilia, "MOBA games: A literature review," *Entertainment Computing,* pp. 128-138, 2018. |
| [2] | Raizin, "Wikipedia," 18 05 2023. [Online]. Available: https://commons.wikimedia.org/wiki/File:Map\_of\_MOBA.svg. |
| [3] | P. Matuszewski, P. Dobrowolski and B. Zawadzki, "The Association Between Personality Traits and eSports Performance," *Frontiers in Psychology,* vol. 11, no. 1490, July 2020. |
| [4] | Kaixiang Wang and S. Wenqian, "Outcome prediction of DOTA2 based on Naïve Bayes classifier," in *2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS)*, Wuhan, China, 2017. |
| [5] | R. Ani, V. Harikumar, A. K. Devan and O. Deepa, "Victory prediction in League of Legends using Feature Selection and Ensemble methods," in *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, Madurai, India, 2019. |
| [6] | V. J. Hodge, S. Devlin, N. Sephton, F. Block, P. I. Cowling and A. Drachen, "Win prediction in multiplayer esports: Live professional match prediction," *IEEE Transactions on Games,* p. 368379, 2019. |
| [7] | A. S. Chan, F. Fachrizal and A. R. Lubis, "Outcome Prediction Using Naïve Bayes Algorithm in the Selection of Role Hero Mobile Legend," *Journal of Physics: Conference Series,* vol. 1566, no. 1, p. 012041, June 2020. |
| [8] | K. Akhmedov and A. H. Phan, "Machine learning models for DOTA 2 outcomesprediction," 3 June 2021. [Online]. Available: https://arxiv.org/abs/2106.01782v1. |
| [9] | L. Breiman, Classification and Regression Trees, Routledge, 2017. |
| [10] | L. Breiman, "Random Forests," *Machine Learning,* vol. 45, no. 1, pp. 5-32, 2001. |
| [11] | J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics,* vol. 29, no. 5, pp. 1189-1232, 2001. |
| [12] | Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences,* vol. 55, no. 1, pp. 119-139, 1997. |
| [13] | G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye and L. Tie-Yan, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," *Advances in neural information processing systems,* vol. 30, 2017. |
| [14] | K. M. Ting, "Confusion Matrix," in *Encyclopedia of Machine Learning and Data Mining (Second Edition)*, Springer Publishing Company, 2017. |
| [15] | E. Alpaydın, Introduction to Machine Learning, Second Edition, London, England: The MIT Press, 2010. |

1. https://developer.riotgames.com/apis [↑](#footnote-ref-1)
2. https://archive.ics.uci.edu [↑](#footnote-ref-2)
3. https://lol.fandom.com/wiki/2022\_Season\_World\_Championship [↑](#footnote-ref-3)
4. https://riot-watcher.readthedocs.io/en/latest/riotwatcher/Riot/index.html [↑](#footnote-ref-4)
5. https://pandas.pydata.org [↑](#footnote-ref-5)
6. https://github.com/atakantuzcu/MScInCSEThesis [↑](#footnote-ref-6)