National College of Ireland

Valorant Esports Pre-Match Betting Advisory System uses Machine Learning to Predict Winning Probability and Simulate Odds and Earning Projections.

MSc Research Project
Data Analytics

Abhishek Mahendra Pawar
Student ID: x23214112

School of Computing
National College of Ireland

Supervisor: Barry Haycock

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: ABHISHEK MAHENDRA PAWAR

Student ID: X23214112

Programme: DATA ANALYTICS **Year:** 2024 - 25

Module: MSc Research Project

Supervisor: BARRY HAYCOCK

Submission Due Date: 12 December 2024

Project Title: Valorant Esports Pre-Match Betting Advisory System uses Machine Learning to Predict Winning Probability and Simulate Odds and Earning Projections.

Word Count: 7736 **Page Count**: 22

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ABHISHEK PAWAR

Date: 12 December 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Valorant Esports Pre-Match Betting Advisory System uses Machine Learning to Predict Winning Probability and Simulate Odds and Earning Projections.

Abhishek Mahendra Pawar
X23214112

Abstract

Over the last few years, the esports world has seen significant growth and Valorant has become the most popular competitive game. As a result of this growth, the betting market has grown, and however betting platforms that exist today still heavily rely on common statistical methods, which are not suitable in the case of the dynamic game. In this thesis, I present a machine learning-based pre-match betting advisory system to predict match outcomes of Valorant professional tournaments. The system goes over historical data like player and team statistics and matches details and uses that to forecast winning probabilities, simulate betting odds that seem genuine, and potential earnings. We implement a robust methodology on complex datasets involving data preprocessing, synthetic data generation using CTGAN and Gaussian Mixture Models, feature engineering, and model evaluation. We developed and compared multiple machine learning models including XGBoost, CatBoost, and a Hybrid Stacking model. We chose XGBoost as its scalability and efficiency make it good for deploying, and beat out others with 73% accuracy. The system consists of a user friendly Streamlit interface that will enable bettors to input a match and get winning probability and earnings projection. The paper points out possibilities of advanced machine learning in esports betting, The future improvements will focus on expanding to other games and improving accuracy with real time performance metrics.

1 Introduction

Over the last few years, the esports world has seen significant growth and Valorant has become the most popular competitive game. Riot Games First Person Shooter (FPS) game Valorant has become one of the leading games since released in 2020. The dynamics of the team, the competitive esports scene, and tactical gameplay are contributing to the popularity (Riot Games, 2020). With the rapid growth of the betting industry, betting platforms have started to take advantage of this opportunity and are offering odds (offer a bet) on professional matches such as VCT (VALORANT Champions Tour). Not like typical sports, forecasting match outcomes in esports is also very unique because of the dynamic and fastchanging nature

of the game and its meta. In this dissertation, a pre-match betting advisory system for Valorant is developed by combining machine learning and deep learning. The system is constructed based on historical data and other influencing factors to predict winning probabilities for two professional competitive teams. It also intends to provide bettors with simulated odds, along with the expected amount of gained bet based on their bets to give bettors a complete solution for decision-making.

Figure 1: Valorant game

Figure 2: Valorant Champions Tour

1.1 The Growth of Esports and Betting

World esports revenue exceeded $1.5 billion in 2023 according to (Fortune Business, 2024), That is why this growth led to the emergence of a secondary betting market where people bet on their favorite teams or players. While in traditional sports past performance data, and trends are easier to predict since the game is relatively stable, esports match often involves quickly changing strategies, patch updates, and the performance of a particular player, which makes predictions more complex (Hamari et al, 2017). Valorant, in particular has a kind of unique case study because it has structured gameplay and it also depends on individual skill and team coordination. Match results are heavily influenced by parameters such as team formation, map selection, and recent performance. However, this complexity also requires an advanced approach for predicting match results using machine or deep learning algorithms for analyzing and forecasting the results. (Ke et al., 2022).

1.2 Problem Statement

Today, the majority of esports betting platforms depend on simple odds calculation methods such as basic statistical or manual calculation. These approaches fail to consider all the important factors that could influence the match results. instance platforms wouldn't take into account variables like team selection, map, players stats, and recent performance as a team. Without clear, detailed, and trustworthy predictions, bettors cannot make wise decisions. To deal with this issue, a system is needed that can accomplish the following:

- Analyze historical data and specific match data.
- Effectively forecasting team victory.
- Simulating betting odds and calculating the potential returns for the bettors.

Research question: How machine learning and deep learning models can be utilized to predict match results and generate simulated odds for Valorant, providing bettors with accurate probability-based information and earnings projections?

1.3 Objectives of the Thesis

This project aims to design and implement a Valorant PreMatch Betting Advisory System using machine learning or deep learning models to help determine efficient predictions and simulated odds. The specific goals include:

a. Collecting and preprocessing data such as match data, player statistics, map performance, and team data.
b. Building and comparing the performance of multiple machine learning and deep learning models.
c. Deploying the best model for predicting match outcomes and betting odds.
d. Creating a user interface that allows bettors to input team selections and get predictions and earnings projections.

1.4 Research Significance

This paper helps in the domains of betting systems and esports analytics. By applying stateof-the-art machine learning methods, the proposed approach can enhance bettor's decisionmaking and pave the way to other methods in the esports betting domain. This work also comes handy with the rising trend of the application of data in the esports industry which is likely to expand with the growth of the market.

2 Related Work

This section briefly discusses prior studies on data pre-processing, synthetic data generation, esports predictive modeling and data visualization. These studies demonstrate the issues involved and possible strategies with these areas of focus and thus form the basis of this research.

2.1 Data Cleaning and Merging for Predictive Analysis.

Probably the most important step in creating reliable machine learning results is data cleaning. As per (Cote et al, 2023), appropriate data cleaning can increase the efficacy of machine learning models by removing errors, inconsistencies, and redundant information. The importance of scalable and domain specific cleaning techniques for complex data management is highlighted by their study. Esports is especially dependent on this as data sourced from multiple places must be organized and collected carefully. Additionally, (Nguyen et al, 2022) discussed the challenges of merging datasets for analysis. the author points out that careful planning during this phase can help overcome data limitations, leading to better model performance. These observations highlight the need for thoughtful data integration in this thesis, where data from player statistics, match history, and map performance will be combined.

2.2 Synthetic Data Generation for Predictive Analysis

As presented by (Xu et al, 2019), in the article Conditional Generative Adversarial Networks (CTGAN) is a method to synthetically generate tabular data. The authors address the challenge of modeling the probability distribution of rows in tabular data and generating realistic synthetic datasets. The complexity presents in tabular data, such as the coexistence of discrete and continuous columns, multimodal distributions in continuous columns, and imbalances in discrete columns, are frequently difficult for traditional statistical and deep learning models to handle. As solutions to these problems, the Author suggests CTGAN, a model that handles these complexities successfully. The research also provides a comprehensive benchmark, along with several baselines for Bayesian networks, incorporating seven simulated and eight real datasets. CTGAN's mobility with different datasets makes it an essential tool for data driven applications. In esports analytics, for instance, synthetic data can simulate player statistics, allowing for better modeling of game dynamics. More importantly, synthetic and real data can increase the performance and robustness of models. Another relevant study by (Chen et al., 2023) This research explores how small amounts of controlled randomness called dithering to data can improve prediction accuracy when data is reduced to lower precision (quantized) for storage or processing. Introducing a bit of random noise before rounding the data makes the model better at estimating and predicting outcomes in complex settings. Their experiments on synthetic data and image processing show that this approach helps capture more accurate patterns, making predictions closer to real results. (Edwards-Thro, 2021) show the feasibility of Gaussian Mixture Models (GMMs) for clustering NBA player's performance metrics. This method shows distinct player roles and strategies in the rapid game of basketball. These observations are particularly relevant to e-sports, where performance measurements and player roles are equally complex. Both player and team analysis in esports can benefit from the use of GMMs, which enable customized tactics and solve data shortages by creating synthetic data. The paper offers a concise overview of GMMs and demonstrates how this can be used to find trends in performance data from a variety of fields. Integrating a uniform noise component, (Coretto, 2022) improves Gaussian Mixture Models (GMMs) and helps in addressing data flaws like as noise and outliers. By using adaptive separation constraints, the Gaussian components and noise are efficiently balanced, improving the model's ability to manage irregularities. Random noise is added to GMM-based probability models to increase their durability and their ability to replicate the flaws frequently present in real-world data. When working with complex, inaccurate datasets, the model becomes accurate and reliable due to its realistic integration of noise. According to Coretto's research, introducing noise is crucial for improving model performance in difficult data scenarios

2.3 Predictive Analytics in Esports

Random Forest classifiers are used by (Groll et al, 2021) to predict match outcomes by combining tree-based techniques with diverse covariates. Their research demonstrates how machine learning may be used in sports analytics and provides a strong basis for esports adaptation. The paper might, however, expand its scope to include challenges in forecasting results in esports competitions, such as different map selection and team configurations. The study's conclusions might be more applicable to esports professionals looking to create prediction models if these differences are taken into considerations (Yang et al. 2022) The study shows how efficient the machine learning algorithm can be to predict the matches in League of Legends using important player and team performance metrics like kills, and assistances along

with the team performance. It compares such models as Logistic Regression, Random Forest and Gradient Boosting and notes that when it comes to several interactions with the data, ensemble methods serve the purpose best. The study emphasizes the methods of data preprocessing like label encoding, and scaling of the features, since the models require high-quality input data. Nonetheless, the study has some limitations due to a focus on canonical models in place and specific games, stressing the necessity to investigate a potential expansion of the dataset and the use of the neural network models. These results are immediately comparable with the code provided below, which presents additional data on each player and team Performance Rates such as ACS and headshot % and applies various Machine learning models including but not limited to XGBoost, Random Forest, and Naive Bayes. The works of (Bahrololloomi et al, 2022), (Yang et al. 2022) and (Hodge et al, 2022) show the importance of machine learning, feature engineering and integration in modeling esports match outcomes. Yang et al. highlight the use of player-level data and team distributions for constructing pre-game prediction where the complexity of the data requires encoding and scaling features; Gradient Boosting is also established as the best-performing model. Yet, it is still suggested to look into issues such as unique compositions of players, and board strategies. (Bahrololloomi et al). very much emphasize feature engineering explaining how complex metrics such as kills, assists, and statistics per map make the models more interpretable and accurate. I find it good that both prioritize the integration of domain knowledge is crucial, but it should also involve the verification of the results with actual esports tournaments. Hodge et al. strengthen the argument for ensemble methods, the paper shows how Random Forest and Gradient Boosting models trained on ACS (Average Combat Score), and team interactions perform better than just a classifier. Based on them, one may assume that ensemble approaches are a rather stable platform to address the challenges that esports data set implies; however, extended research within the frameworks of deep learning paradigms may bring more benefits. As a whole, these works are used to form the code that uses the metrics including kills, assists, headshot percentages, and ACS, along with methods such as XGBoost and Random Forest.

2.4 Predictive Analytics in Sport Betting

Here, the work of (Walsh et al 2024) highlights the ability of model calibration over model accuracy in sports betting saying that correctly scaled models yield noticeably more profits by identifying “value bets.” As the authors noted there is a tendency to regard calibration as the main key for probabilistic betting decision-making instead of accuracy-oriented methods. Using NBA data, results show that calibration results in an investment return of 35% while accuracy-based models incurred losses. Given this, even though the investigation is confined to basketball, it provides guidelines for using calibration strategies in other fields including Esports betting, where exact probability estimation is crucial for identifying profitable opportunities.

2.5 Data Visualization in Esports Analytics

The research paper (Du et al, 2020) presents a complete guide for creating excellent visualisations for performance analysis and strategy. Its emphasis is on integrating descriptive, diagnostic, and predictive visualisations which are explicitly reflected in the code, which analyses esports data using scatter plots and heatmaps. While the research is thorough, introducing examples particular to esports metrics like kill-death ratios and mapspecific strategies could improve its application.

In conclusion, the related works show the early developments of predictive modeling in esports, data preprocessing. and synthetic data generation. Data challenges could be solved with CTGAN and GMM methods and who machine learning can help to improve prediction accuracy. It is based on these studies of data, for creating a solid system of predicting match results and making esports betting.

3 Research Methodology

The research method for this thesis involves designing a machine learning automatic betting assistant for Valorant Esports pre-match betting. The approach comprises data understanding, data acquisition and preprocessing, synthesizing data, feature selection, modeling, model assessment, system implementation, and usability. The research methodology is a customized version of CRISP-DM, tailored as per project.

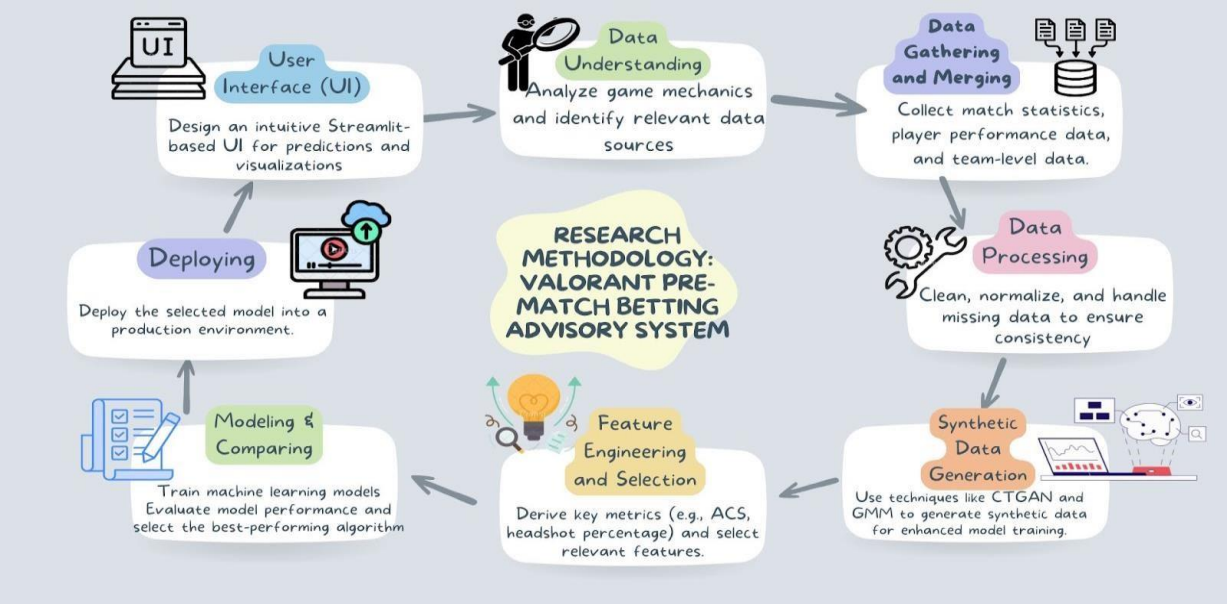

Figure 3: Research Methodology

3.1 Data Understanding

The research highlights the Valorant game, an esports game that offers interesting features for the construction of the pre-match betting advisory system. The first objective of the study was to forecast match results and generate betting odds for professional matches employing machine learning. The data sources included player statistics, team statistics, statistics for particular maps, and statistics for particular matches as recorded through historical data. Again, other parameters like kills, assists, Deaths, and map etc. were considered as performance indicators. These variables helped to assess either individual or team performances regarding match along with providing a strong predictor base.

3.2 Data Gathering and Merging

Information was gathered from different sources including tournament archives, player performance documentation, as well as team statistics files. Four primary datasets were

compiled: This includes game level data (for example, individual player performance), team IDs (the mapping between the team's unique identifiers), map performance data (the mapping between the map and statistics provided on that map), and match records (recorded match details). Unique identifiers, e.g., player ID, team ID, and match ID, were used to merge the datasets. For instance, player statistics were merged with their team information as well as match level data to form a unified dataset. In order to analyze and model these results, the study provided a comprehensive merging process that merged all relevant data.

3.3 Data Processing

The collected data was cleaned and transformed to ensure quality and consistency. Numerical attributes were handled using median imputation so that the data did not lose its originality. Duplicate records were removed to prevent biases in the analysis Numeric values like headshot percentage had a "%" symbol that was removed and the remaining string was converted to float data types. Moreover, maps were among the categorical features of the datasets, and so maps were encoded numerically using label encoding for machine learning. Meaningful scaling was performed to ensure contributions such as kills, deaths, and assists, to the models were all on the same scale.

3.4 Synthetic Data Generation

As a solution to the problem of doing more with limited data, synthetic data was produced with a Conditional Generative Adversarial Network (CTGAN). This process proved to be efficient in producing reasonable tabular data that could mimic the original datasets. The metrics: kills, deaths, assists, ACS, and headshots percentage were merged as it was relative; higher kills contribute to a higher ACS value. Gaussian Mixture Models (GMM) were then used again to estimate match-level probability and odds which included elements of risk for bettors. It was argued that synthetic data greatly improved the training process by addressing data shortage and generalization of the model. Similarly, Random Forest (RF) was used to improve the quality of the synthetic dataset by predicting match winners in the synthetic cases. RF model took account of historical match data to predict the winners including factors like team and player statistics information, map performance, or simply past simulated chances to predict the winners. Thus, by using such RF in combination with synthetic data during the formulation of the approach, the potential match outcomes were considered coherent with historical statistics and typical game patterns. This approach improved the quality of synthetic data for downstream modeling and its utility in bridging the realism gap when data is scarce.

3.5 Feature Engineering and Selection

After preparing the data, additional features were engineered to improve the model's predictive capabilities. For example, player statistics were aggregated to calculate team-level metrics, such as total kills and average ACS. New attributes, such as win rates and side advantages (attacker vs. defender performance), were also added to capture team strategies and map-specific dynamics. Normalization techniques were applied to ensure fairness when comparing teams across different matches and maps. These engineered features played a crucial role in enhancing the model’s accuracy.

3.6 Modeling

Some machine learning models were built and compared to predict match outcomes. These included models like Logistic Regression and tree based models like Random Forest and Gradient Boosting. Ensemble methods, such as Bagging and AdaBoost were also used, along with a hybrid stacking model trained using predictions from several base models. To capture sequential dependencies, deep learning models including a Recurrent Neural Network (RNN) were explored. Trained and validated our models using stratified train test splits that ensured a balanced representation of match outcomes across datasets.

3.7 Comparing and Evaluation

Model accuracy, precision, recall, F1 score, and ROC-AUC were used as measures of performance in modeling. The results revealed models, provided high accuracy in identifying match-winners. Leveraging base model strengths, the hybrid stacking model showed good performance. The evolution of each model was described along with its merits and demerits and the best model was selected based on the comparison based on the aim of the project i.e. accuracy and recall.

3.8 Deploying

The main predictive model was deployed. after serializing the model and the preprocessing pipeline (label encoder and scaler) using joblib and storing them in MongoDB for easy integrations. The real time deployments ensure bettors can generate predictions in real time and allow a bettor to make an informed decision. In addition to making, it easy to update and retrain the system with new data as it became available to make them relevant to changing gameplay dynamics.

3.9 User Interface

It has been implemented with Streamlit, an open-source app framework for building data, visual, and interactive applications, to provide a minimalistic interface. It enabled users to input team IDs, map names, and bet amounts and then predict the chances along with simulated probability and earnings. The victory probabilities of both teams were represented in the frequentative form of a horizontal bar chart to provide a user representation of predictions. Mistakes that occurred on the UI were well managed.

4 Design Specification

This thesis presents a structured approach to developing a prematch Valorant esports betting advisory system. Data collection and preprocessing is the first step in the system, meaning that raw data of players, teams, and matches is pre-processed using Python libraries like pandas and NumPy, etc. After collecting it from numerous datasets, the data is sorted, cleaned, and merged into a single dataset. When there is a lack of data, bias or imbalances, synthetic data is generated by a CTGAN and GMM to address the issue to allow the machine learning models to train on diverse data. The processed data is stored in a Mongodb, giving users secure storage for further

analysis. Feature engineering was used to come up with meaningful metrics like team performance and produce map-based statistics, and feature selection to see what features have a direct impact on the outcome of the match. Several Machine Learning models like Logistic regression, Decision trees, Random Forests, naïve Bayes, and Recurrent Neural Networks (RNNs). next, these models will be evaluated, with metrics such as accuracy recall, and precision, to determine which algorithm performs the best.

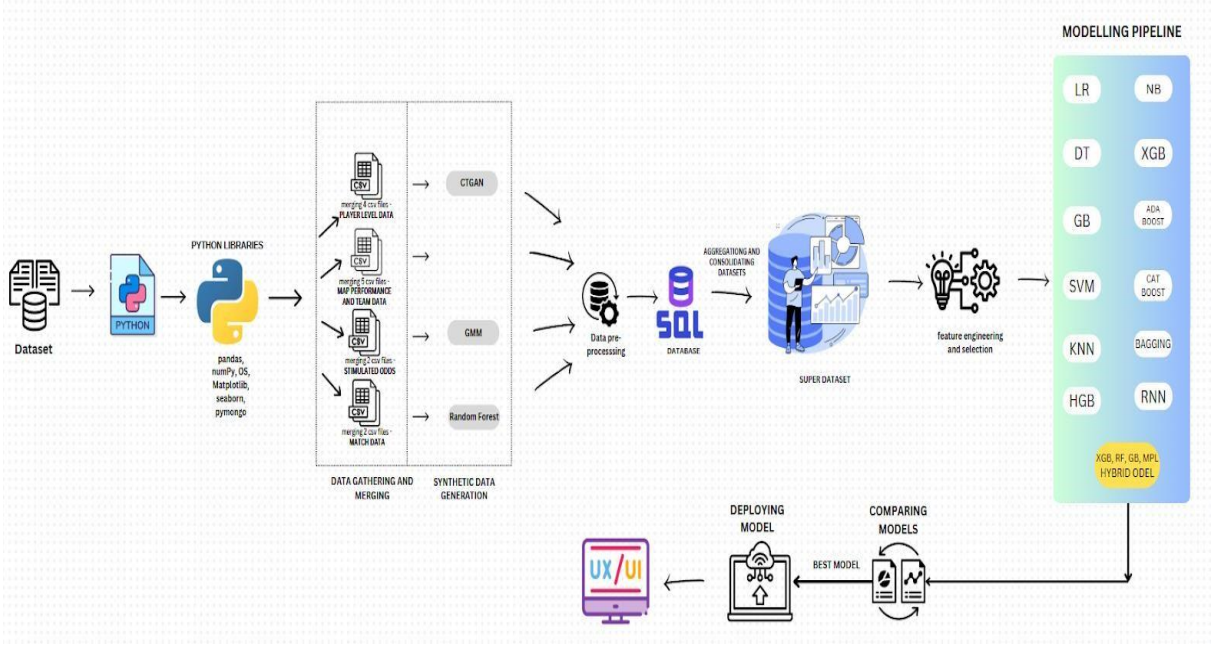

Figure 4. Research Design Specification

Streamlit is used to deploy the chosen model with a user-friendly interface that is accessible for users to view predictions, betting odds, and potential earnings. The modular pipeline incorporates data science methods, machine learning, and user interaction to create an innovative and practical solution to the esports betting problem.

5 Implementation

The implementation phase involves systems building of the Valorant Pre-Match Betting Advisory System by integrating data collection, preprocessing, modeling, and deployment into a pipeline. It does involve merging various esport datasets, cleaning the existing dataset, transforming the dataset, creating synthetic datasets to fill up gaps and then engineering meaningful features. multiple machine learning models are trained and evaluated to predict match results. lastly, application is deployed using the user interface.

5.1 Data Gathering and Merging

collection and merging methods were to combine many datasets into a single structure dataset for analysis. Different data was collected based on player level statistics, team identifiers, match records, as well as map specific performance data. These datasets required detailed information for player performance, team composition, match outcome, and map change. The first part was loading the datasets and making sure the column names were all the same. To

effectively merge datasets, unique identifiers like player, team, and match IDs were used. Individual performance metrics were identified by connecting player IDs to player statistics. These were merged with team IDs in order to link players to their teams.

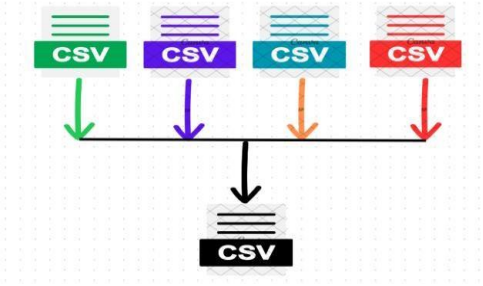

Figure 5: Dataset Merging

To give a complete overview of every game, match-level data like map and tournament details were also integrated. The study utilizes these unique data to produce a strong foundation that reliably depicts the complex aspects of Valorant esports competitions on predictive modeling models. (Nguyen et al, 2022)

5.2 Data Processing

One of the important steps for getting the data ready for the machine learning model is data preprocessing. The main goal is to have clean data, which is consistent and structured for analysis. The preprocessing stage for this study further refined the dataset obtained through player statistics, team performance and match records. The initial dataset had inconsistencies, like missing values, duplicates and formatted wrongly. To tackle them, several things were done. I first checked for outliers in the numerical columns such as Kills, Deaths, and Assists, and then implemented a maximum threshold so that the values don't exceed those of professional gameplay standards. Furthermore, percentage-based columns like Headshot Percentage were converted to float types by removing the "%" symbol to make them uniform for the analysis. Encoded categorical features like map names, which were treated as machine-readable, but still represented categorical values. In order to avoid biases in the data, duplicate rows and records containing missing critical identifiers, for example, Match ID and Player ID, were removed. Numerical features, such as Average Combat Score (ACS), were scaled through normalization to get values on the same range so as to prevent the same model from being biased toward features with larger values of numerical scales. All the processed data was validated and checked for completeness and accuracy and was then stored in MongoDB for further analysis. This study addressed these preprocessing challenges systematically to ensure that the resulting dataset was clean and reliable and had a decent number of meaningful features. Accurate prediction and robust model performance depend upon this painstaking preprocessing. (Cote et al, 2023)

5.3 Synthetic Data Generation

When real data is insufficient or imbalanced, it is an important part of research to generate synthetic data. This study uses the dataset to enhance the synthetic data for training the machine learning models. By running the model on synthetic data generated according to realistic data patterns and filling the gaps in the original dataset, the model gained diversity in different scenarios while generalizing and predicting the data accuracy. For synthetic data generation,

there were two primary techniques used, Conditional Tabular Generative Adversarial Networks (CTGAN) and Gaussian Mixture Models (GMM).

CTGAN algorithm was used to synthesize player-level data from features such as Kills, Deaths, Assists, First Bloods, Average Combat Score (ACS), and Headshot Percentage. The model was trained on 3,000 data points that were sampled from the existing dataset over 150 epochs, so the synthetic data matched the real patterns found in professional gameplay. For instance, synthetic data instance thresholds were applied to keep things consistent (maximum Kills = 52, Deaths = 36, in line with the statistical range of professional matches). Calculating Average Combat Score (ACS) is the representation of a player's performance based on kills and assists. (Xu et al, 2019)

$$ACS = (Kills \times 10) + (Assists \times 5)$$

Probabilities and simulated betting odds for teams were created using Gaussian Mixture Models. Modeled distributions of implied probabilities for Team A and Team B in the range of 0.35–0.65 using the GMM algorithm. This was chosen to approximate the range of realistic odds in competitive matches, implied probabilities are neither too high nor too low for either team. To simulate the uncertainty of real-world scenarios, noise and randomness were put in the generated probabilities np.random.normal (0, 0.02). From these probabilities,
Simulated Odds were calculated for both teams. Implied Probabilities Calculation: (Edwards-Thro, 2021)

$$(Implied\_Prob\_A) = Value\ from\ GMM + Random\ Noise$$
$$(Implied\_Prob\_B) = 1 - Implied\_Prob\_A + Random\ Noise$$ (Kaunitz et al, 2017)

To predict match outcomes, the Random Forest Classifier was trained on merged synthetic data. The features were *Simulated_Odds_A*, *Simulated_Odds_B*, *Implied_Prob_A*, *Implied_Prob_B*, and the target Variable was *Simulate_Winner* as a binary (1 for *Team_A*, 0 for *Team_B*). The model achieved 99.13% accuracy, with a 70% vs. 30% train vs. test split on *Predicted_Team_A_Win*. Synthetic data was effective in forecasting esports match results, while this robust approach validated the utility of synthetic data. The final synthetic dataset was created by combining player level, team level, and match level data such as Kills, assists, ACS, implied probabilities, simulated odds and predicted winners. This complete dataset filled the gaps in real world data that allowed the development of a robust and good betting advisory system. (Groll et al, 2021)

5.4 Uploading datasets in MongoDB

Once data understanding, data gathering and merging, data preprocessing, and synthetic data are generated then data is stored and managed, using MongoDB as the primary database system for this research. *player_level_data, map_performance, simulated_odds, player_ids, tournaments_stages_matches_games_ids, teams_ids*, and *Match_Data* were uploaded to MongoDB collections in Python as key datasets. Datasets are converted into a structured format, inserted into the relevant collections, for efficient retrieval and analysis, and ensured accurate storage as well as seamless integration with the predictive modeling pipeline.

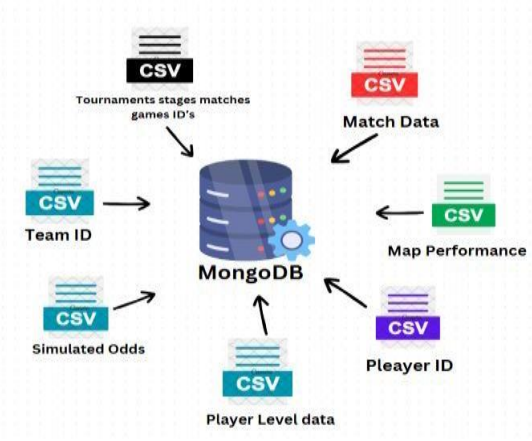

Figure 6: Uploading datasets in MongoDB

5.5 Feature Engineering

The process of feature engineering is to transform the data into a structured form for our machine learning model to improve our predictive abilities. The feature engineering process in this project involved aggregating and combining data from several databases *player_level_data, map_performance, Match_Data* and simulated_odds into a unified database for further analysis. mapping between team names and their respective IDs using the *simulated_odds* dataset, ensuring consistency across datasets. Team names in the *match_data* dataset were mapped to these IDs, and rows, where the mapping failed, were removed to maintain data integrity. Team IDs were converted to integers for uniformity. Aggregated the *player_level_data* by match and team to produce features for team performance, such as total kills, total deaths, total assists, average combat score (ACS), average headshot percentage, and total first blood. These aggregated statistics gave us a full view of what a team does, and what to expect in a match. It was then merged with map performance data to ensure all map specific stats were integrated. The records of each team (Team A and Team B) were then combined in the database in order to serve as features for the two teams. Renaming the columns so that it is obvious when a column indicates Team A or for the column with Team B data. Simulated_odds dataset was merged, which included features such as implied probabilities and the simulated odds for both teams. Columns that were redundant and irrelevant were dropped, and the dataset was cleaned by removing rows with missing values and duplicates. These features such as team-specific aggregated stats, match specific map data, and simulated odds were collected together and were called the superdataset that provided the backbone for model training and prediction. By using this systematic approach to feature engineering, the dataset was complete, consistent, and in a form ready for subsequent modeling, increasing the analysis's quality.

5.6 Modeling

A number of machine learning models were used during the modeling phase to predict match outcomes. Our aim is to find the most accurate model with respect to accuracy, precision, recall, and F1 score while checking data leaks, imbalances, and biases.

5.6.1 Feature Selection and Preprocessing

Firstly, the best features from the superdataset were chosen for modeling. The features included match, map, and team statistics for Team A and Team B. Categorical features like map names

were also encoded using LabelEncoder with text data being transformed into numerical data good for machine learning. Since the model wants all features to have the same scales, StandardScaler was used to standardize the data. StandardScaler make sure that models based on distance metrics like KNN, and SVM are not biased by large scale features.

5.6.2 Data Splitting and Prevent Leaks

train_test_split was used to split the dataset into two sets train and test 80:20 respectively to maintain class balance in both. To prevent data leakage, data was strictly separated between train and test data. Each training fold was independent from the test fold, which was ensured by cross-validation. also, make sure during feature selection that no such feature is introduced that can leak match outcome.

5.6.3 model pipeline

1) **Extreme Gradient Boosting (XGBoost) -** is a gradient boosting algorithm for speed and performance. The hyperparameters are set at 200 estimators, a maximum tree depth of 4, and a learning rate 0.05. It was chosen due to its capacity to manage missing values and tabular data feature importance. Both in terms of balance of complexity and computational efficiency, it creates a strong baseline for ensemble learning.

2) **Logistic Regression** - An algorithm was trained on a maximum of 100 iterations. This model is an improvement of the traditional binary classification model that enables us to predict the probability of a team's win. Because of its simplicity and it is to interpret it can be used as a baseline model.

3) **Naive Bayes** - The probabilistic model was based on normal distribution of continuous variables. Due to potential limitations with feature correlations, it was fast computation and a nice baseline comparison type tool.

4) **Decision Tree Classifier** - This model uses default parameters and splits data into binary decision paths. It overfitted without an ensemble method and gave details into how features influence outcomes.

5) **Gradient Boosting Classifier (GBM)** - GBM was configured = 200 estimators, a maximum depth = 4, and a learning rate of 0.05 that incrementally improved weak learners for increased optimization of predictions.

6) **Support Vector Machine (SVM)** - It arranges data points to the high dimensional space to find the optimal hyperplane using probability estimation for classification. Although this approach was considerably strong in theory but its computational complexity was too heavy for large scale application.

7) **K-Nearest Neighbors (KNN)** - used 5 neighbors to predict class membership by proximity in feature space. The method was computationally simple, but sensitive to scaling. While it performed well for smaller datasets, it was less competitive on this dataset.

8) **AdaBoost** - It started with 200 estimators. We focused on improving weak classifiers by reweighting misclassified samples during training.

9) **CatBoost** - was configured with 200 iterations, a depth of 4, and a learning rate of 0.05, and was realized successfully as a gradient boosting algorithm optimized for categorical data. Without preprocessing, it was able to properly handle categorical features.

10) **Bagging classifier** - This ensemble method combined multiple Decision Tree Classifiers with 100 estimators to reduce variance and improve stability.

11) **Histogram-Based Gradient Boosting** - with 100 iterations, used histogram-based techniques for Strong gradient boosting. It was particularly suitable for large datasets.

12) **Recurrent Neural Network (RNN)** - The architecture used was a RNN layer of size SimpleRNN (128), a dropout layer (0.2), and 2 dense layers (64 units and 1 output unit). By training the model for 50 epochs with a batch size of 50 and binary cross entropy loss as well as Adam optimized with a learning rate of 0.0001, the model was trained as described.

13) **Hybrid Stacking Model** - In this method the predictions by XGBoost, Random Forest, Gradient Boosting, MLPClassifier were combined as base models. It learned to maximize predictions from the base models outputs using logistic regression as the meta-model.

A pipeline was used to implement each model. It guarantees consistent data preparation, training and results. The pipeline tried different models to predict of match outcome. (Yang et al. 2022)

6 Evaluation

Any thesis based on machine learning should be evaluated to show that models are accurate, reliable, and effective. This thesis used accuracy, F1 score, precision, and recall to evaluate model performance. With this, a complete understanding of the model's strengths and weaknesses was made. The process itself led to the selection of the best model, which had strong transparency, reliability, and a solid base from which to extend in the future.

6.1 Top 3 Models

XGBoost shows accuracy of 73%, F1 of 73%, and ROC-AUC of 76% were found in XGBoost which do well with handling true positives and false positives. However, it performed well in all aspects of evaluation, having a Recall and Precision of 0.73 on both. It predicted 21 losses and 21 wins incorrectly, but 54 losses and 57 wins correctly. model's key merits are handles large datasets, reduces overfitting, and handles missing values.

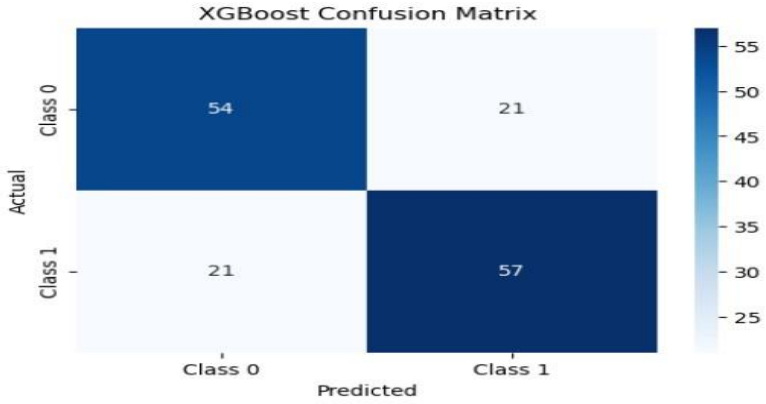

Figure 7: XGBoost confusion matrix

CatBoost model with an accuracy of 72%, an F1 score of 73% and a ROC-AUC score of 76%, catboost also performed reasonably well. Both precision and recall were 72% and 73. In fact, the model correctly predicted 53 losses and 57 wins and wrongly predicted 22 losses and 21 wins. CatBoost model increased the predictive performance when used on structured data and is good at handling feature interactions.

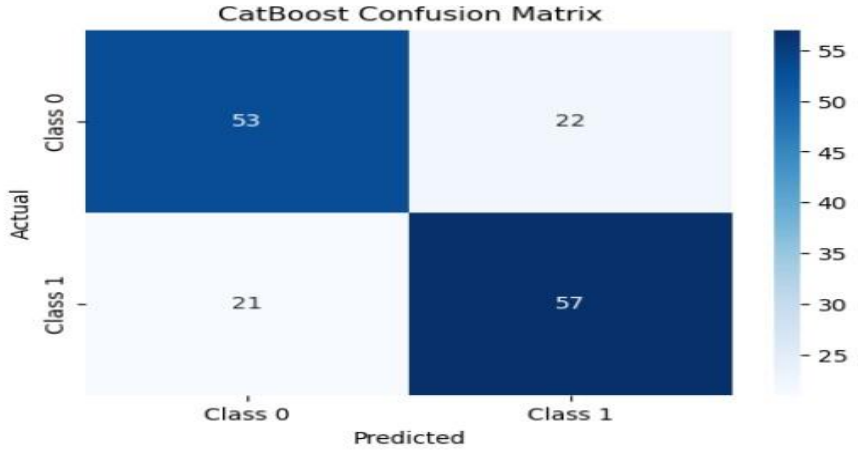

Figure 8: CatBoost confusion matrix

The Hybrid Model achieved 70% accuracy, 72% F1 score and 74% ROC-AUC score. The meta classifier combined the strength of the various base models such as XGBoost, Random Forest and Gradient Boosting. It correctly predicted 38 of the losses and 48 of the wins but flipped wrong on 15 of the losses and 22 of the wins.

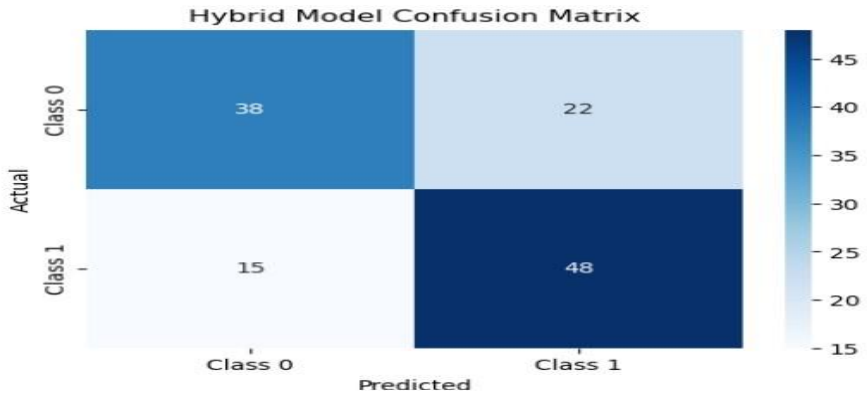

Figure 9: Hybrid Model confusion matrix

6.2 Comparison of All Models

The performance table shows the reasons why some of these models, despite their merits, aren't on the shortlist of the top three. With an accuracy of 66%, Logistic Regression is unable to capture nonlinear relations that would require better complexity. Although simple and efficient, the accuracy of 67% of naive Bayes is limited by conditional independence assumptions enshrined in it. Decision Tree easily overfits at 55% accuracy and generalization is hard. Gradient Boosting and SVM produce reasonable accuracy (71% and 71%, respectively) but miss the mark with precision and recall scores for the best models. The computationally expensive KNN gave out 62 % accuracy with large dataset. Histogram
Gradient Boosting, Bagging, and AdaBoost show fair results but lack consistency regarding the recall and ROC AUC. All of these limitations, taken together, justify not putting these models into the top three.

Table 1: Model Comparison

Model	Accuracy	F1-Score	ROC-AUC	Precision	Recall
XGBoost	0.73	0.73	0.76	0.73	0.73
Logistic Regression	0.66	0.66	0.71	0.68	0.64
Naive Bayes	0.67	0.68	0.71	0.68	0.69
Decision Tree	0.55	0.55	0.55	0.56	0.54
Hybrid Model	0.70	0.72	0.74	0.69	0.76
AdaBoost	0.63	0.64	0.67	0.63	0.64
SVM	0.71	0.72	0.75	0.70	0.73
KNN	0.62	0.63	0.66	0.62	0.64
CatBoost	0.72	0.73	0.76	0.72	0.73
Bagging	0.67	0.69	0.72	0.67	0.71
Histogram-Based Gradient Boosting (HGB)	0.70	0.71	0.74	0.71	0.71
RNN	0.69	0.70	0.73	0.68	0.72

6.3 Best Model For Deployment

XGBoost is a primary choice due to its better performance metrics and computational efficiency compared to other similar algorithms here. The ability to make real time predictions in esports betting with the ability to process big data quickly. The algorithm XGBoost performs either better or as same as the Hybrid Model but it is faster and easy to deploy.

7 Deployment

In the deployment phase, we operationalize the predictive system and guarantee it is available and working for real time predictions. The purpose of this phase is to make sure that the predictive model is integrated seamlessly into a user facing application, where a bettor can make wise decisions. It starts with saving the trained XGBoost model, label encoder, and scaler on MongoDB. For the XGBoost model, we optimized and configured with 300 estimators, a maximum depth of 4, and a learning rate of 0.05. The computational efficiency of this setup is balanced with model complexity. The model was serialized with joblib and converted to BSON so that it can be used in MongoDB. Python's pickle module was used to serialize the model alongside the label encoder and the use of StandardScaler. These were stored in a MongoDB collection model with identifiers such as *model_name*, for later use. We retrieve from MongoDB the stored XGBoost model, label encoder, and scaler. The final model data are deserialized back into a usable format using joblib, and the label encoder and scaler are deserialized using Pickle. In this case, it makes sure that the deployment environment is similar to the training environment. The real time prediction system takes input parameters, like team IDs, map name & bet amount to calculate match outcomes, betting odds & potential returns. It checks inputs against past data and calculates median team performance metrics (such as kills, ACS, etc.) with encoded and scaled data for compatibility with the XGBoost model. The model outputs winning probabilities for both teams, such that summing these gives approximately 1. Probabilities are adjusted with a custom function to include a 5% bookmaker margin, to turn them into real odds. These odds will give us potential earnings based on their bet amount. This deployment ensures that the outputs of the model, probabilities, adjusted odds and potential earnings are accurate. For example, take a *team_a_id=17*, *team_b_id=2*, *map_name="Abyss"*, the model always predicts probabilities first, then it calculates odds with a margin and shows potential earnings for the bet of €22. This pipeline can produce accurate, actionable predictions that are ready for real world usage in esports betting.

```
Winning Probabilities - Team A: 0.59, Team B: 0.41
Realistic Odds (with margin) - Team A: 1.61, Team B: 2.34
Potential Earnings - Bet €22 on Team A: €35.32, on Team B: €51.50
```

Figure 10: Deployment Output

8 User Interface

Using Streamlit, we built a user interface (UI) for the Valorant Pre-Match Betting Advisory System, making complex backend processes easier to use. Users can input team ID, map name and bet amount, and have real time validation to make sure it’s correct. The system produces output for every game detailed - winning probabilities shown in a horizontal bar chart, realistic betting odds with a bookmaker margin, and earning projections generated from the money that is bet. These insights enable users to make realistic judgments about the different possible financial outcomes. In case there were errors, UI handles that gracefully and shows users invalid input or system error. Esport betting would benefit from this interface as it offers real time predictions and insights.

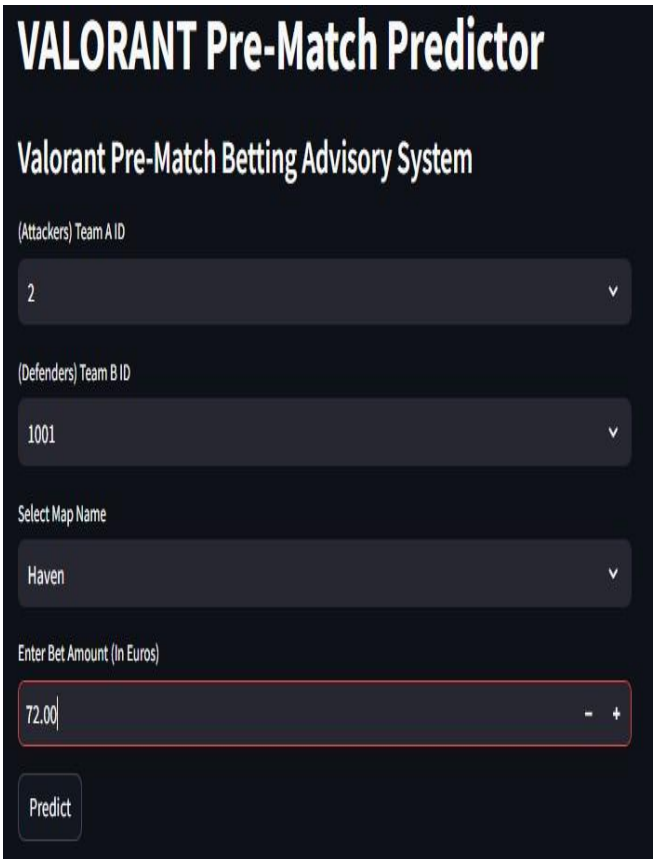

Figure 11: User Interface

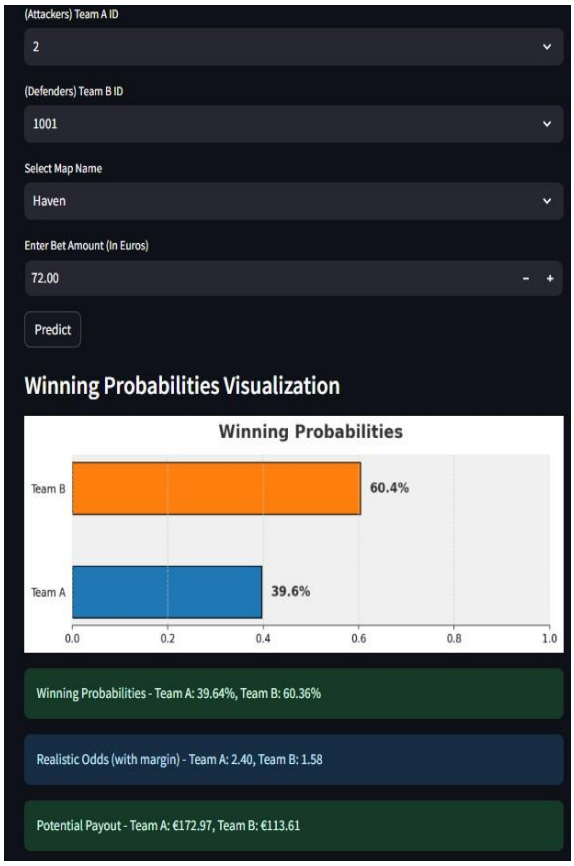

Figure 12: Predictions on UI

9 Conclusion

The aim of this thesis was to design and deploy a Valorant Esports Pre-Match Betting Advisory System equipped with machine learning that offers actionable insights to bettors. It also included steps like data collection, preprocessing, feature engineering, model evaluation and deployment, user interface design, etc. The system predicts match outcomes using historical esports data providing simulated odds and potential earnings, and an improved decision making process for anyone involved in esports betting. Different Machine Learning models, XGBoost, CatBoost, and Hybrid models were deployed and it was found that the XGBoost was very stable and versatile providing an accuracy of 73% and an F1 score of 0.73. Due to XGBoost's great efficiency and scalability for handling several large datasets quickly and its speed, XGBoost was the most suitable for this real-time prediction system. The thesis analyzed other models in the literature, and insights were drawn on the trade-offs between simplicity, accuracy, and computational requirements, and the hybrid model also shows promising results. Streamlit is being used as the user interface which was designed for ease of familiarity (for bettors) so they can enter match details and then access predictions, odds, and potential earnings. In addition, MongoDB was used to store models, data, and configuration files as well as to provide real time functionality on the deployment pipeline. This setup makes sure the system is both scalable and adaptable for the future enhancement and upgrades.

10 Future Work

Future improvements include expanding the scope of this project and improving predictive capabilities. This thesis provides a framework and the techniques used that can be applied to other popular esports games like CSGO and League of Legends to make the system more relevant. By adding more data player level data, such things as recent performance trends, player form and team dynamics, predictions can become more accurate. try to demonstrate more advanced machine learning models.

References

Bahrololloomi, F., et al. (2022). A Machine Learning Based Predictive Analysis Use Case for eSports Games. Dergipark. https://dergipark.org.tr/en/download/article-file/2990904.

Chen, J., Wang, Y., & Ng, M. K. (2023). Quantized Low-Rank Multivariate Regression With Random Dithering. *IEEE Transactions on Signal Processing*, *71*. https://doi.org/10.1109/TSP.2023.3322813

Coretto, P., 2021. Estimation and computations for Gaussian mixtures with uniform noise under separation constraints. *Journal of the Italian Statistical Society*, 31(3). Available at: https://doi.org/10.1007/s10260-021-00578-2

Côté, Pierre-Olivier & Nikanjam, Amin & Ahmed, Nafisa & Humeniuk, Dmytro & Khomh, Foutse. (2024). Data cleaning and machine learning: a systematic literature review. Automated Software Engineering. 31. 10.1007/s10515-024-00453-w. Nguyen et al. (2022). *Significance of dataset merging in machine learning.*

Du, M. and Yuan, X. (2021). A survey of competitive sports data visualization and visual analysis. *Journal of Visualization*, 24(1), pp.47–67. https://doi.org/10.1007/s12650-020-00687-2

Edwards-Thro, N. (2021). *Gaussian Mixture Models for Clustering NBA Player Performance Metrics*. https://nedwardsthro.github.io/files/Thro_Honors_Thesis.pdf

Fortune Business Insights (2024) eSports Market Size, Share, and Industry Analysis, By Streaming Type (Live and On-demand), Multiplayer Online Battle Arena Games, Mass Multiplayer Online Role-Playing Games, and Others), and Regional Forecast, 2024-2032. Report ID: FBI106820 . https://www.fortunebusinessinsights.com/esports-market-106820.

Groll, A., Ley, C., Schauberger, G. & Van Eetvelde, H., 2024. Soccer match outcome prediction with random forest and gradient boosting models. *ResearchGate*. https://www.researchgate.net/publication/378355415_Soccer_match_outcome_prediction_with_random_forest_and_gradient_boosting_models

Hamari, J., & Sjöblom, M. (2017). What is eSports and why do people watch it? *Internet Research*, *27*(2). https://doi.org/10.1108/IntR-04-2016-0085

Hodge, V., et al. (2022). E-Sports Player Performance Metrics for Predicting the Outcome of Professional League of Legends Matches. Springer. https://link.springer.com/article/10.1007/s42979-022-01660-6.

Kaunitz, L.N., Zhong, S., & Kreiner, J. (2017). Beating the bookies with their own numbers - and how the online sports betting market is rigged. *ArXiv, abs/1710.02824*.

Ke, C. H., Deng, H., Xu, C., Li, J., Gu, X., Yadamsuren, B., Klabjan, D., Sifa, R., Drachen, A., & Demediuk, S. (2022). DOTA 2 match prediction through deep learning team fight models. *IEEE Conference on Computatonal Intelligence and Games, CIG*, *2022-August*. https://doi.org/10.1109/CoG51982.2022.9893647

Metulini, R., Manisera, M. & Zuccolotto, P., 2018. Preprocessing techniques in basketball analytics. *arXiv preprint arXiv:1707.01430*. https://arxiv.org/abs/1707.01430

Noroozi Fakhabi, A., Hasan, M. S., Ravan, M., Norouzi, E. & Law, Y.-Y., 2024. An efficient machine learning approach for extracting eSports players' distinguishing features and classifying their skill levels using symbolic transfer entropy and consensus nested cross-validation. *International Journal of Data Science and Analytics*. https://link.springer.com/article/10.1007/s41060-024-00529-6.

Pratama, H.K. & Priyanta, S., 2024. *Prediction Valorant Match Outcome Using Supervised Learning Algorithm*. Thesis. Universitas Gadjah Mada. Available at: https://etd.repository.ugm.ac.id/penelitian/detail/242904

Riot Games, 2020. *Valorant Official Website*. https://playvalorant.com/

Walsh, C., & Joshi, A. (2024). Machine learning for sports betting: Should model selection be based on accuracy or calibration? *Machine Learning with Applications*, *16*. https://doi.org/10.1016/j.mlwa.2024.100539

Xu, L., Skoularidou, M., Cuesta-Infante, A., & Veeramachaneni, K. (2019). Modeling tabular data using conditional GAN. *Advances in Neural Information Processing Systems*, *32*.

Yang, Y., Qin, T., Lei, Y.H. (2022). Machine Learning Methods for Predicting League of Legends Game Outcome. IEEE. https://ieeexplore.ieee.org/document/9720122.