Netaji Subas University of Technology,
Dwarka, Delhi, 110078, India

Authors

Under the guidance of Prof. Gaurav Singal

Kushal	**Ashish Kumar**	**Vaibhav Joshi**
2022UCA1833	2022UCA1813	2022UCA1854

Abstract

This project presents a machine learning pipeline designed to predict the outcomes of Valorant matches based on player statistics and match data. Leveraging historical game data, including metrics such as Kills, Assists, Deaths, and Average Combat Score (ACS), we built and evaluated predictive models using both Deep Neural Networks (DNNs) and Gradient Boosting Classifiers. Data was collected from two SQLite tables, "Games" and "Game_Scoreboard," and underwent rigorous preprocessing, feature engineering, and transformation to ensure robustness.

The DNN model was constructed with Keras, featuring layers optimized with the HeNormal initializer, ELU activation functions, Dropout, and Batch Normalization to enhance training stability and model performance. Gradient Boosting was implemented as a comparative approach, providing an alternative to deep learning. Models were evaluated using 5-fold cross-validation, yielding accuracy metrics, standard deviation, and standard error to assess predictive reliability. Finally, we applied these models to new match data to predict the likelihood of team victory based on player and team statistics.

This project highlights the use of machine learning in esports analytics, offering valuable insights into how player performance can be quantitatively assessed to forecast outcomes in competitive gaming scenarios.

1 Introduction

In this project, we aim to develop a machine learning pipeline to predict the winner of Valorant matches based on player statistics and match outcomes. By leveraging data from past games, including individual player performance metrics such as Kills, Assists, Deaths, and ACS (Average Combat Score), our model provides a data-driven approach to match prediction. The project involves the use of both Deep Neural Networks (DNNs) and Gradient Boosting Classifiers to create accurate and robust predictive models.

To achieve this, we started by collecting and preprocessing data from two key tables, "Games" and "Game_Scoreboard", stored in an SQLite database. The raw data underwent extensive cleaning, feature engineering, and transformation to prepare it for model training. Key features such as player stats and team performance were selected, and missing values were imputed, ensuring the dataset's completeness.

The next step involved building and evaluating models using advanced machine learning techniques. We designed a deep learning model using the Keras library with a HeNormal initializer to predict match outcomes. The model architecture includes multiple hidden layers with activation functions

such as ELU, Dropout, and Batch Normalisation for improved training stability and performance. Additionally, we implemented Gradient Boosting as a traditional machine learning alternative to compare with the DNN model.

Both models were evaluated using 5-fold cross-validation to ensure generalization across unseen data. Accuracy scores, standard deviation, and standard error were calculated to assess model performance. Moreover, we implemented a strategy to retrieve player stats and predict future match outcomes by comparing team stats and computing relevant features. The final stage involves making predictions on new match data using the trained models, helping to determine which team has a higher probability of winning.

This project demonstrates the application of machine learning techniques in esports, offering insights into how player performance can be modeled and analyzed to predict outcomes in competitive gaming.

1.1 Advantages

1.1.1 1. Enhanced Decision-Making:

Our project provides valuable insights by analyzing extensive game data, which can significantly improve decision-making for teams and coaches. By predicting match outcomes and assessing player performance, teams can refine their strategies before and during games. For instance, knowing which players are statistically more likely to perform well against certain opponents can lead to more effective lineup choices. Coaches can also adjust tactics in real-time based on predictive analytics, optimizing their game plan to counter opponents' strengths or exploit weaknesses.

1.1.2 2. Data-Driven Player Development:

The project focuses on analyzing individual player statistics, which allows for a deeper understanding of their skills and areas needing improvement. By identifying trends in player performance, coaches can tailor training programs to address specific weaknesses, such as aiming accuracy or teamwork. For example, if the analysis shows that a player excels in kills but struggles with deaths, targeted training can be implemented to enhance their survivability in matches. This data-driven approach not only fosters individual growth but also contributes to overall team performance.

1.1.3 3. Real-Time Performance Analytics:

Implementing real-time analytics during matches can revolutionize how teams operate. Our project can provide immediate feedback on player performance and game dynamics, enabling coaches and analysts to make quick adjustments. For example, if a particular strategy isn't working based on in-game data, the team can pivot to an alternative approach, improving their chances of success. Additionally, this capability enhances the viewing experience for fans by providing insights into match progress and potential outcomes, making the event more engaging and interactive.

1.1.4 4. Scalability Across Multiple Games:

One of the significant advantages of our project is its scalability, allowing the predictive models and methodologies to be applied to various eSports titles. This versatility means that the insights gained from one game can inform strategies in another, facilitating a broader understanding of gaming trends and player behaviors across different platforms. For example, techniques developed for a popular title like League of Legends can be adapted for games like Dota 2 or Valorant, making our project applicable in diverse competitive environments and potentially benefiting a wide range of teams and players.

2 Literature Survey

The explosive growth of eSports has revolutionized both competitive gaming and spectator engagement, leading to an unprecedented demand for advanced analytics and predictive modeling. At the heart of this innovation are Artificial Intelligence (AI) and Machine Learning (ML) techniques, which harness massive amounts of game data to refine strategies, predict outcomes, and enhance

viewer experience. A diverse body of literature explores these emerging techniques, highlighting their transformative potential in the eSports landscape.

A foundational study by IEEE in 2020 outlines the scope of AI applications in eSports, exploring how machine learning algorithms can analyze gameplay data to generate real-time insights and improve player and team performance [1]. This comprehensive study emphasizes the significance of big data integration in eSports, which has laid the groundwork for sophisticated AI-driven tools capable of assessing performance metrics, predicting player actions, and optimizing in-game decisions with a high degree of accuracy.

Building upon this, subsequent research in 2022 delves into the specific advantages of deep learning models within eSports analytics. As noted by IEEE, deep learning architectures such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) excel at processing and interpreting complex, unstructured data from game environments, identifying patterns that may not be readily apparent to human analysts [2]. These models' abilities to autonomously extract and interpret gameplay features have proven particularly effective for games with high variability and fast-paced dynamics, underscoring deep learning's role as an essential tool in eSports analytics.

Further advancements are seen in predictive modeling, where sophisticated algorithms account for various game-specific factors. An IEEE study from 2022 introduces predictive models tailored to the unique dynamics of eSports, such as team composition and player skill variance [3]. Here, the integration of these factors into predictive algorithms reveals a marked increase in the accuracy of game outcome predictions, suggesting that tailored models can significantly outperform more generalized AI approaches. Another IEEE publication from 2023 corroborates this trend, illustrating how deep reinforcement learning allows AI agents to adaptively modify strategies in real time, a critical feature for accurate game prediction in live competitive settings [4].

One key study in 2022 explores the feasibility of end-to-end deep learning frameworks for eSports prediction, presenting frameworks that minimize the need for extensive preprocessing and instead rely on the models' capacity to learn directly from raw data [5]. Such frameworks are particularly suited to eSports, where real-time analysis is paramount, and have demonstrated substantial promise in streamlining the prediction process while maintaining high predictive accuracy. However, recent insights from ScienceDirect in 2024 highlight the challenge of variability in player performance, cautioning that factors like experience and individual playstyle introduce significant complexities into prediction models [6]. This underscores the necessity of incorporating robust player analytics into any predictive modeling approach for eSports.

Player analytics has, in fact, emerged as a focal point in AI-driven eSports research. A detailed 2023 IEEE analysis emphasizes the importance of granular player metrics, such as response times and strategic choices, in predicting game outcomes [7]. By continuously monitoring and analyzing these metrics, AI models can offer deeper insights into players' performance, allowing teams to adjust strategies and even enhance training protocols based on quantitative player behavior patterns.

Real-time prediction is also transforming the viewing experience. The IEEE study from 2023 demonstrates the potential of real-time prediction algorithms to generate live win probabilities and other insights that can be displayed to viewers in near real-time [8]. These algorithms, which process game data within milliseconds, enrich the spectator experience and enhance engagement by providing dynamic, data-driven narratives as the game unfolds. This application of AI is especially relevant for live streaming platforms, where quick, interactive analytics are crucial for audience retention.

The evolution of machine learning applications in eSports is further detailed in IEEE's 2019 publication, which highlights the progression from traditional machine learning models like decision trees and support vector machines to more advanced neural network architectures [9]. This historical perspective illustrates the shift towards increasingly complex and adaptive models, showing how the integration of newer techniques, such as deep learning and reinforcement learning, has enabled AI to tackle the unique challenges posed by eSports data.

Additionally, Springer's 2022 research on hybrid models introduces another layer of complexity, suggesting that combining statistical models with machine learning approaches can enhance predictive accuracy and adaptability [10]. This hybrid approach leverages the statistical rigor of traditional models alongside the adaptability of AI, creating a more robust framework capable of handling the intricate, high-dimensional data characteristic of eSports environments.

Finally, ACM's 2021 exploration of AI and data analytics in eSports provides a comprehensive view of how analytics can be leveraged beyond prediction. By drawing insights from gameplay data, these analytics not only enhance team strategies and training but also inform game developers, helping them fine-tune gameplay mechanics and respond to player ps [11]. This study highlights the broader impact of AI, as data-driven insights influence both competitive play and the eSports industry's commercial side, from player development to game design.

Taken together, these studies illustrate the profound and multifaceted role that AI and ML play in shaping the future of eSports. By leveraging advanced analytics, eSports can offer increasingly sophisticated strategies, real-time predictions, and an enhanced spectator experience. Furthermore, the ongoing advancements in AI methodologies—ranging from deep learning to hybrid models—underscore the need for continuous innovation to meet the dynamic demands of this rapidly evolving field.

3 Data Science and Preprocessing

3.1 Understanding the Dataset

For the eSports analytics project, we utilize the Valorant Pro Matches Full Data dataset, which is publicly available on Kaggle (link to dataset). This dataset captures in-depth data from professional Valorant matches, specifically from the high-stakes Valorant Champions Tour (VCT). The dataset is rich in detail, capturing both team and player metrics, which serves as a foundation for data-driven insights, predictive modeling, and player analytics in competitive Valorant.

3.2 Dataset Composition and Scope

The dataset covers a range of statistical indicators critical for understanding player performance and team dynamics. Key components include:

3.2.1 Player Performance Metrics

This section contains details on kills, assists, deaths, and first bloods, alongside more granular metrics like headshot percentages and average damage per round. These data points allow us to assess each player's effectiveness in both individual and team contexts, offering a window into their consistency, impact, and adaptability under competitive pressures.

3.2.2 Agent Selection and Strategy

Information on agents (the characters chosen by each player) is included, providing insight into which agents are favored by pros and how their roles and abilities contribute to team success. By analyzing agent selection across various matches, we can infer the popularity of certain playstyles or meta strategies in the VCT.

3.2.3 Map-Specific Performance

Detailed statistics on maps played highlight team and player performances on different map terrains, allowing us to study whether specific strategies perform better on certain maps. This data is essential for understanding map-specific advantages, as well as how terrain impacts gameplay.

3.2.4 Round-by-Round Analysis

The dataset includes round-by-round information, including which team won each round and the economic conditions (e.g., full buy, half buy, or save rounds). Analyzing this data can reveal the influence of in-game economic decisions on match outcomes, providing a basis for economic modeling in predictive analyses.

3.3 Data Cleaning and Preprocessing

Data preprocessing is the bedrock upon which successful machine learning models are built. A crucial phase in the machine learning pipeline, it involves transforming raw data into a structured format that maximizes the potential of the predictive models. In this project, we begin by working

with the Valorant Pro Matches Full Data from Kaggle, which provides comprehensive player statistics and match data. While this dataset is well-maintained, it still requires several steps of preprocessing to ensure its readiness for modeling.

Each stage of preprocessing contributes to refining the dataset, addressing issues such as missing values, inconsistencies, and ensuring that the data is in a format that algorithms can understand. For instance, we handle the encoding of categorical variables, such as agent names and map identifiers, transforming them into numerical formats suitable for machine learning models. Furthermore, additional features are engineered to provide more context, such as calculating moving averages for player performance, or deriving synergy metrics based on team and agent compositions.

A key aspect of our preprocessing workflow is the understanding of feature correlations. By examining the relationships between various features, we gain insights into how different attributes influence the outcome of a match. This allows us to refine our feature selection process, eliminating redundant variables and highlighting the most informative ones for model training. Strong correlations between variables like kills, deaths, and damage per round, for example, can provide valuable insights into player performance, which are critical for building accurate prediction models.

Furthermore, data normalization and scaling are applied to ensure that numerical features are balanced, preventing larger magnitude variables from dominating model training. These steps are essential for ensuring that models such as gradient-boosted trees and neural networks can operate efficiently.

Having thoroughly addressed data cleaning, transformation, feature engineering, and correlation analysis, we can confidently proceed to the next steps in our project. The robustness of this preprocessing phase ensures that the data is both clean and structured, laying a solid foundation for the predictive models that will follow. As we progress, we will delve deeper into the specifics of model training and evaluation, armed with a dataset primed for optimal performance.

Given the high-quality standards typically observed in datasets published on Kaggle, the Valorant Pro Matches Full Data dataset was largely well-prepared upon download. Nonetheless, rigorous data cleaning and preprocessing steps were undertaken to ensure it met the specific requirements of our project, enhancing its readiness for machine learning applications. The preprocessing phase encompassed several key tasks:

3.3.1 Handling Missing Values

Any remaining null values were systematically addressed, either through imputation or removal, based on the feature's significance to the model. For instance, null values in non-critical columns were removed to maintain data integrity, while missing entries in essential fields were filled using appropriate averages or median values. This process ensured completeness across all records, contributing to more reliable model outcomes.

3.3.2 Outlier Detection and Handling

Outliers were detected and analyzed, particularly in high-variance features like kills per round or damage per round, where extreme values could skew model predictions. Outliers were either capped within a reasonable range or excluded based on their statistical significance, enhancing the dataset's robustness for accurate modeling.

3.3.3 Normalization and Scaling

To ensure uniformity, especially for metrics such as kills, deaths, assists, and damage per round, all numerical data was scaled. This was achieved through standard normalization techniques, aligning the magnitude of each feature to optimize model training and prevent any feature from disproportionately impacting the results. This step was critical for algorithms like gradient boosting and neural networks, which are sensitive to differences in feature magnitudes.

3.3.4 Feature Engineering

Additional features were created to improve model performance, including moving averages for player statistics over multiple matches, and synergy metrics based on agent combinations and team composition. These engineered features contributed to a more nuanced understanding of performance

patterns and strategic trends within the dataset, enabling the model to capture complex relationships among variables.

3.4 Data Splitting

In machine learning, the strategy of data splitting is a foundational step that determines how effectively a model will perform on unseen data. Dividing the dataset into distinct subsets ensures that the model is not only trained but also evaluated in a way that simulates real-world conditions, where data is dynamic and constantly evolving. The key objective of data splitting is to assess how well the model generalizes to new data by training it on one part of the data and testing it on another.

3.4.1 Training, Validation, and Test Sets

The process of splitting the dataset into three distinct sets—training, validation, and test—ensures that the model is subjected to a thorough and impartial evaluation:

- **Training Set**: This subset is used for learning the underlying patterns in the data. It provides the raw material for the model to adjust its internal parameters, minimizing the loss function and learning how to make predictions. A well-defined training set is crucial to prevent overfitting, ensuring that the model doesn't just memorize the data but rather learns the patterns that generalize to new, unseen data.

- **Validation Set**: The validation set is used to tune model hyperparameters and evaluate performance during the model-building process. This is particularly useful when trying to optimize the model's settings (e.g., choosing the number of layers in a neural network or adjusting the learning rate). Unlike the training set, the model does not have access to this data during training, providing an unbiased estimate of its performance. It serves as a feedback loop that helps identify when the model is starting to overfit and when adjustments should be made.

- **Test Set**: The test set is reserved exclusively for assessing the final performance of the trained model. It represents new data that the model has never seen, allowing a true measure of its ability to generalize. A model's performance on the test set is a direct indication of how it will behave when deployed in a real-world scenario, where it encounters data that may differ from the training examples.

3.4.2 Stratified Sampling for Balanced Representation

In datasets where certain classes are underrepresented (e.g., rare events or outcomes), stratified sampling is employed to ensure that each class is proportionally represented across training, validation, and test sets. This is especially important in tasks such as predicting match outcomes in esports analytics, where the distribution of wins and losses may not be equal.

What is Stratified Sampling?
Stratified sampling involves partitioning the dataset in such a way that each class is represented according to its original proportion in the entire dataset. For example, if 80% of the data points represent a victory and 20% represent a loss, stratified sampling ensures that each split (train, validation, and test) maintains this 80-20 ratio. This ensures that no subset of the data is dominated by one class, preventing skewed learning or biased predictions, especially when working with imbalanced datasets.

Why is Stratified Sampling Crucial in Esports?
Esports datasets, such as the Valorant Pro Matches Full Data used in this project, can often be imbalanced. For example, the number of rounds won by one team may vastly outnumber the rounds won by the opposing team, or certain player statistics (e.g., kills or assists) may be rarer than others. Without stratified sampling, the model might favor the dominant class, leading to poor performance when predicting outcomes for the less frequent class. Stratified sampling mitigates this issue by ensuring balanced representation, leading to more reliable and fair model predictions.

3.4.3 Cross-Validation

The Key to Robust Model Evaluation, cross-validation is an essential technique to evaluate the model's performance more reliably by testing it across multiple training and testing scenarios. Unlike a single split of the data, cross-validation involves partitioning the dataset into multiple subsets, training the model on some subsets, and testing it on others. This approach provides a more comprehensive assessment of model performance, especially in cases where the dataset is limited.

Stratified K-Fold Cross-Validation

For this project, Stratified K-Fold Cross-Validation is used. This technique involves dividing the dataset into k equal-sized folds while preserving the proportional distribution of the target class in each fold. For example, in a binary classification task with an 80-20 split, stratified k-fold ensures that each fold maintains this distribution, both in training and test sets. This ensures that no fold is skewed, and each evaluation of the model is as representative as possible.

Benefits of Stratified K-Fold Cross-Validation

- **Improved Model Performance Estimation**: By using multiple folds, each data point in the dataset is used both for training and testing. This provides a more accurate and robust estimate of model performance.
- **Reduced Bias**: Stratified k-fold cross-validation minimizes the risk of performance bias, ensuring that the model is consistently evaluated across different subsets. This is particularly beneficial when working with imbalanced datasets, as it ensures the model's predictions are tested on all classes in a balanced manner.
- **Better Generalization**: Through repeated training and testing on different subsets, the model's ability to generalize to unseen data improves. This is essential for esports analytics, where data points (such as player performance in different maps) can vary significantly.
- **Efficiency**: While cross-validation does require more computational resources than a simple train-test split, stratified k-fold offers a reasonable balance of computational expense and model robustness, making it a preferred choice in scenarios like this, where model accuracy is paramount.

How it Works
In k-fold cross-validation, the dataset is split into k folds (typically 5 or 10). For each iteration, the model is trained on k-1 folds and tested on the remaining fold. This process repeats until every fold has been used as a test set once. The final performance metric is the average across all k iterations.

For example, in a 5-fold stratified cross-validation, the dataset is divided into five equal-sized portions. In each iteration, the model is trained on four portions and evaluated on the remaining one, cycling through all five folds.

Advantages:

- **Better Utilization of Data**: Every data point gets to be part of both the training and the test set, enhancing the robustness of the model.
- **Reduced Variance in Performance Metrics**: The multiple evaluations reduce the variance that might arise from a single random split of the data.
- **More Accurate Generalization**: As each fold provides a new test set, the model's ability to generalize is tested under different conditions, giving a better understanding of how it would perform in the real world.

Ensuring Real-World Applicability

The combination of stratified sampling and stratified k-fold cross-validation ensures that the model is both accurately trained and thoroughly evaluated. These techniques are essential in preventing overfitting and underfitting, ensuring that the model's performance on the test set is not skewed by imbalanced data or biases from a single data split. The result is a more robust and reliable model, ready to make predictions in real-world settings.

For this project, the use of these sophisticated data-splitting techniques is fundamental to achieving the goal of predicting esports match outcomes with high accuracy. By ensuring that the model is well-trained, properly validated, and consistently tested across multiple folds, we can trust the insights derived from the model to be reliable, actionable, and representative of actual game scenarios.

In conclusion, the careful application of data splitting, stratified sampling, and cross-validation is integral to building a machine learning pipeline that is both efficient and capable of generalizing to new, unseen data. This not only improves model performance but also strengthens the predictive power of the system, making it suitable for real-world deployment in esports analytics.

3.5 Data Integration and Use Case Potential

This dataset provides a versatile basis for implementing a range of data science and machine learning techniques. For example, it enables the development of predictive models to forecast match outcomes based on player metrics or agent combinations. Additionally, clustering methods could help categorize players into skill tiers or playstyles, enriching eSports analytics with deeper insights into player archetypes. In terms of real-time analytics, the round-by-round data allows for potential applications in live match predictions, making this dataset a prime resource for developing AI-driven applications in the competitive gaming domain.

In summary, the Valorant Pro Matches Full Data from Kaggle is a well-structured, versatile dataset that enables an in-depth analysis of professional Valorant gameplay. By applying rigorous preprocessing and feature engineering techniques, this dataset can be transformed into actionable insights, guiding strategic decisions, enhancing predictive models, and contributing valuable knowledge to the rapidly evolving field of eSports analytics.

3.6 Features Taken

In our predictive model, feature selection was approached with an emphasis on capturing the multi-faceted dynamics of competitive Valorant gameplay. Each feature was chosen with care, providing not only statistical data but also strategic insights into player behavior, skill level, and impact on match outcomes. Our dataset, derived from a comprehensive Kaggle repository, includes performance metrics that cover both fundamental and advanced aspects of gameplay. The selected features enable the model to understand player contributions from diverse angles, enhancing its ability to predict match results accurately. Below is a detailed examination of each feature and its role in our analysis.

- **Average Combat Score (ACS)**: Serving as a composite metric, ACS offers a holistic view of a player's influence in a match. This score combines various performance aspects, including kills and damage dealt, and is a reliable indicator of a player's consistency and impact. In competitive environments, a high ACS signifies an all-rounder with a significant effect on the outcome of each round.

- **Kills**: The count of eliminations achieved by a player is a core metric of offensive capability. Kills are a straightforward yet essential measure of effectiveness, representing the ability to contribute directly to team success by eliminating opponents. This feature helps quantify the raw aggressive power of each player.

- **Deaths**: Reflecting the number of times a player has been eliminated, this metric offers insight into a player's survivability. While a low death count is favorable, balancing it with high engagement (kills and assists) indicates a strategic playstyle. This feature contributes to understanding the player's risk-taking tendencies and resilience under pressure.

- **Assists**: Assists measure indirect contributions to kills, highlighting a player's teamwork and support capabilities. This feature is particularly valuable in team-based strategies, as it emphasizes the role of players who may not secure kills directly but facilitate them, showcasing their versatility and adaptability.

- **PlusMinus (±)**: Calculated as the difference between a player's kills and deaths, PlusMinus provides an overall effectiveness metric. A positive score indicates that a player outperformed opponents more frequently, while a negative score suggests vulnerability. This metric is an insightful indicator of a player's balance between offense and defense.

- **Kill, Assist, Survival, and Trade Percentage (KAST%)**: This composite statistic reveals the consistency of a player's engagement in impactful actions across rounds. It reflects the percentage of rounds where the player participated in a kill, assist, survived, or traded a teammate's death. High KAST% values signal a player who consistently contributes to the team's success through diverse roles and actions.

- **Average Damage per Round (ADR)**: This metric represents the average damage inflicted by a player per round, serving as a measure of offensive pressure and damage output. High ADR values generally correlate with players who excel in inflicting substantial damage and impacting enemy resources.

- **Headshot Percentage (HS%)**: The percentage of kills achieved via headshots provides insight into a player's aiming precision and skill. This feature differentiates between players with advanced shooting accuracy and those who rely on volume over precision. High HS% scores are indicative of refined skills, making this an important metric for assessing shooting expertise.

- **First Kills**: Securing the initial kill in a round can set the tone and momentum, giving the team an early advantage. This feature reflects a player's ability to make impactful plays at the start of rounds, which is critical in high-stakes matches where first-mover advantage can lead to a favorable outcome.

- **First Deaths**: The count of rounds in which a player was the first to be eliminated indicates risk-taking tendencies. Players with high first-death rates may have aggressive playstyles, often pushing the front lines. This metric helps capture the balance between risk and reward, showcasing players who may be vulnerable in early engagements.

- **First Kill-First Death PlusMinus (FKFD ±)**: As a net measure of first kills and first deaths, this feature provides a balanced perspective on a player's effectiveness in early confrontations. A positive value suggests favorable engagements, where the player consistently secures initial kills without frequently becoming the first death.

- **Number of 2Ks**: This feature counts the rounds where the player achieved two kills, showcasing their ability to handle multiple opponents effectively. A player with a high 2K count often thrives in high-pressure situations and can make impactful plays that benefit the team, especially in critical moments.

- **Number of 3Ks**: The frequency of rounds in which a player scored three kills highlights a higher level of combat proficiency and capability to win isolated engagements. Players with notable 3K counts are generally adept at clutching or swinging the momentum in favor of their team.

- **Number of 4Ks**: Representing rounds where the player eliminated four opponents, this metric underscores exceptional combat skill and the capacity to dominate multiple enemies. High 4K counts are rare and indicative of top-tier performance, often changing the round's trajectory.

- **Number of 5Ks (Aces)**: Achieving a 5K, or ace, is a remarkable feat where the player eliminates the entire enemy team single-handedly. This feature represents peak individual performance, demonstrating a level of skill and focus that can be game-defining. High ace counts showcase players who can take control of rounds entirely on their own.

These features collectively encapsulate a comprehensive and multidimensional view of each player's performance, covering aspects from individual skill to team-oriented contributions. By including standard metrics like kills, deaths, and assists, alongside more nuanced measures like PlusMinus,

KAST%, and FKFD PlusMinus, our feature set aims to capture not just player statistics but also gameplay style, strategic influence, and adaptability in dynamic match scenarios.

In preprocessing, these features were standardized to enhance model performance and ensure consistency. Such an inclusive selection strengthens the model's ability to accurately predict match outcomes based on diverse in-game behaviors and performance indicators. This detailed approach reflects the nuanced and complex nature of competitive gaming, where success is determined by a blend of individual talent, strategic depth, and synergy with team dynamics.

3.7 Correlation Analysis in Esports Data

Correlation analysis is a cornerstone of exploratory data analysis, offering deep insights into how different variables relate to each other. In the context of esports analytics, understanding the relationships between various player performance metrics, such as kills, deaths, assists, and other critical statistics like Average Combat Score (ACS), PlusMinus, and match score, is pivotal for optimizing performance prediction models. Correlation provides a measure of the degree to which two variables move in relation to each other, with positive correlation indicating that as one variable increases, so does the other, and negative correlation suggesting that as one variable increases, the other decreases.

Why Correlation Matters in Esports Analytics

In esports like Valorant, player performance is multifaceted. Performance metrics such as kills, deaths, assists, and ACS are not isolated; they often influence one another. For example, high kill counts can correlate with higher ACS, as players who secure more kills may engage in more favorable combat situations, boosting their overall combat score. By examining these relationships, we can pinpoint which features are most influential in determining a player's performance and, subsequently, the outcome of a match. This becomes essential when designing predictive models that rely on feature selection, as understanding feature dependencies helps avoid overfitting and ensures model efficiency.

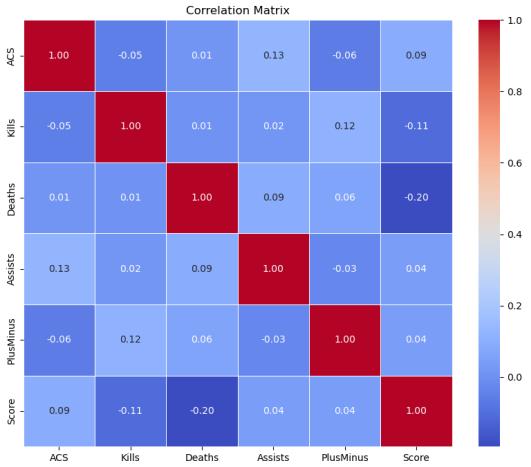

Figure 1: Correlation Matrix

3.7.1 Correlation Matrix Construction

To systematically explore these relationships, we use a correlation matrix, a table that displays correlation coefficients between pairs of variables. The coefficients range from -1 to +1, with values closer to 1 or -1 indicating stronger relationships. A coefficient of +1 means a perfect positive correlation (as one variable increases, so does the other), while -1 represents a perfect negative correlation (as one variable increases, the other decreases). A coefficient close to 0 indicates no linear relationship between the variables.

In this project, the correlation matrix is generated for key performance metrics such as Kills, Deaths, Assists, ACS, PlusMinus, and Score. These variables were chosen because they are directly tied to in-game actions and contribute significantly to understanding player effectiveness.

- **ACS (Average Combat Score)**: This metric aggregates various aspects of a player's in-game activity, including kills, deaths, assists, and other combat interactions. As a result, it is highly correlated with performance metrics like Kills and Assists.
- **Kills and Assists**: These metrics are essential indicators of a player's contribution to their team's success. Kills directly reflect a player's offensive effectiveness, while Assists indicate their role in supporting team strategies.
- **Deaths**: This is the inverse of Kills. A higher number of deaths typically correlates negatively with performance metrics like Kills and ACS, as it suggests a lower ability to survive and contribute during rounds.
- **PlusMinus**: This variable reflects the difference between a player's team score when the player is alive and the team score when they are dead. It provides a holistic measure of a player's impact on the match beyond just kills and deaths.
- **Score**: The total score encapsulates a variety of player activities and contributes to determining overall match success.

The generated correlation matrix is then visualized using a heatmap, a graphical representation that helps quickly identify the strength and direction of relationships between these metrics. The heatmap uses a coolwarm color palette, where red shades indicate strong positive correlations and blue shades represent negative correlations, providing a clear, at-a-glance view of how variables are interconnected.

3.7.2 Insights from Correlation Analysis

Upon generating the correlation matrix and visualizing it with a heatmap, several insightful patterns emerge:

- **Kills and ACS**: A strong positive correlation is expected between Kills and ACS, as players who are more successful in securing kills typically perform better in combat, reflected in their ACS. This relationship is essential for understanding which metrics indicate high-performance players.

- **Kills and Deaths**: There is often an inverse relationship between Kills and Deaths. A player who secures a high number of kills generally has lower deaths, indicating that players who engage effectively in combat tend to survive longer in matches, maintaining a high performance throughout the game.

- **Assists and Kills**: The positive correlation between Assists and Kills highlights the importance of teamwork and support roles in esports. Players who contribute with assists often help teammates secure kills, reflecting the importance of coordinated team strategies.

- **Deaths and PlusMinus**: Deaths typically show a negative correlation with PlusMinus. Players with more deaths tend to have a lower PlusMinus, suggesting that they negatively impact their team's overall success when they are eliminated frequently.

- **Score and ACS**: The positive relationship between Score and ACS implies that players who perform well in combat, as indicated by ACS, are likely to have higher scores, consolidating the notion that effective combat performance contributes to match success.

3.7.3 Implications for Feature Selection

Correlation analysis plays a vital role in feature selection for machine learning models. Understanding which variables are highly correlated allows us to select the most relevant features for training predictive models, reducing redundancy and improving model performance. Highly correlated features, such as Kills and ACS, may be considered together to prevent multicollinearity, which can lead to model instability. Conversely, features with low or no correlation may be excluded from the model, streamlining the feature set and focusing on the most impactful variables.

Additionally, recognizing the relationships between features helps in identifying potential new features through feature engineering. For example, combining Kills and Assists into a composite metric could capture a player's overall offensive contribution, which may not be fully represented by the individual metrics alone.

Addressing Multicollinearity

Multicollinearity, the condition where two or more features are highly correlated, can distort the predictions of many machine learning algorithms, particularly linear regression models. In this project, by analyzing the correlation matrix, we can detect such issues and take appropriate steps, such as removing or combining highly correlated features, to ensure that the model remains stable and interpretable.

3.7.4 Conclusion

In summary, the correlation analysis conducted in this project provides essential insights into the relationships between various player performance metrics in Valorant. By leveraging the power of the correlation matrix, we can better understand how different variables interact and influence one another. This understanding not only guides feature selection and engineering but also helps identify potential issues with multicollinearity, paving the way for more accurate and efficient predictive models. Ultimately, correlation analysis is an indispensable tool in esports analytics, allowing us to create models that are both robust and capable of making precise predictions based on player performance data.

This deep dive into correlation offers a rich foundation for further analysis and model building, ensuring that the data used to predict esports outcomes is both structured and informative, thus enhancing the overall effectiveness of the machine learning pipeline.

4 Algorithms

4.1 Deep Neural Network (DNN) Architecture with 15 Layers

A Deep Neural Network (DNN) is a class of artificial neural networks that is particularly powerful for tasks involving complex data structures, such as image processing, natural language understanding, and time-series forecasting. The DNN used in this project has 15 layers, which gives it the capability to capture intricate patterns in large datasets. With each additional layer, the model becomes progressively more abstract in its understanding of the data, which allows it to tackle highly complex prediction problems like those found in esports analytics.

This section will delve into the specifics of the DNN architecture used in the project, including an in-depth look at the individual layers, activation functions, and the rationale behind their design choices. We will also explore the advantages and challenges that come with implementing a deep neural network of such a scale, and the various mechanisms used to optimize the model's performance.

4.1.1 Input Layer: Data Intake and Preprocessing

The input layer is the first layer of the model and serves as the entry point for the raw data. In the context of this project, the data includes various statistics from esports matches, such as player kills, assists, deaths, and other relevant features. This data is preprocessed to ensure it is properly formatted and scaled before being fed into the model.

In this case, the input_dim represents the number of features in the dataset, and this layer does not perform any computation. Its sole function is to pass the data to the subsequent layers for processing. By structuring the data appropriately, we ensure that the model receives it in a form that it can learn from effectively. Proper handling of the input data is critical to the success of the deep learning process, as it sets the stage for all the learning that will occur in the following layers.

4.1.2 Hidden Layers: Learning Complex Patterns with ELU Activation

A DNN typically consists of multiple hidden layers, each responsible for progressively transforming the input data into a more abstract representation. In this model, the first three hidden layers are designed to refine the learned features and extract increasingly complex patterns from the data.

Each hidden layer uses ELU (Exponential Linear Unit) activation, which has been chosen due to its ability to improve training speed and reduce the likelihood of vanishing gradients compared to traditional activation functions like sigmoid or tanh. The ELU function allows for both positive and negative values, which helps the model learn more efficiently by overcoming the limitations of other functions that only allow positive outputs.

$$f(x) = \begin{cases} x, & \text{if } x > 0, \\ \alpha(e^x - 1), & \text{if } x \leq 0. \end{cases}$$

- **Dense Layer 1 (128 Units)**: The first hidden layer consists of 128 neurons. The dense layer configuration ensures that each neuron receives input from all neurons in the previous layer, allowing for the model to learn relationships between features that are not immediately obvious. The use of the ELU activation function enables the network to learn nonlinear relationships effectively. The layer is followed by dropout (0.1), which randomly disables 10% of the neurons during each training iteration. This reduces the chance of overfitting, ensuring that the model does not memorize the training data but instead generalizes well to unseen data. The batch normalization step is also used to standardize the inputs to the activation function, ensuring that the optimization process remains stable.

- **Dense Layer 2 (64 Units)**: The second hidden layer has 64 neurons and also uses the ELU activation function. Like the first hidden layer, it is followed by dropout and batch

normalization to ensure the model continues to learn effectively while avoiding overfitting. This layer allows the model to start learning more abstract features, capturing higher-order interactions between the input features. The reduced number of neurons compared to the first layer reflects the increasing level of abstraction as we move deeper into the network.

- **Dense Layer 3 (32 Units)**: The third hidden layer contains 32 neurons. It builds on the learning from the previous layers, further abstracting the input data. While fewer neurons are used, each neuron in this layer learns increasingly complex relationships in the data. Again, the ELU activation function, dropout, and batch normalization are employed to ensure that the model continues to train efficiently without overfitting.

4.1.3 Output Layer: Generating Predictions with Sigmoid Activation

The output layer is responsible for producing the model's final prediction. Given that this project deals with a binary classification task (predicting whether a match will be won or lost), the output layer consists of a single neuron with a sigmoid activation function.

Sigmoid Activation: The sigmoid function maps the output to a range between 0 and 1, which can be interpreted as a probability. This is ideal for binary classification tasks, where the model's output can represent the probability of one class (e.g., win) versus the other class (e.g., loss). When the output of the sigmoid function is closer to 1, it indicates a higher probability of a win, and when closer to 0, it indicates a higher probability of a loss. This setup is ideal for decision-making in esports analytics, where predicting match outcomes based on various factors is the core objective.

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

4.1.4 Loss Function and Optimizer: Training the Model

Once the architecture is defined, the next step is to compile the model with an appropriate loss function and optimizer. In this project, binary cross-entropy is used as the loss function because the task is binary classification. Binary cross-entropy measures the difference between the true labels (win/loss) and the predicted probabilities, guiding the model in adjusting its weights during training.

Adam Optimizer: Adam (Adaptive Moment Estimation) is the chosen optimizer due to its effectiveness in adapting the learning rate for each parameter individually. It combines the benefits of two other popular optimization algorithms, RMSProp and AdaGrad, making it particularly well-suited for training deep neural networks. By adjusting the learning rate dynamically based on the gradients, Adam helps accelerate the convergence of the model and avoid getting stuck in local minima. The combination of binary cross-entropy and Adam allows the DNN model to learn efficiently, optimizing for the best parameters while minimizing the loss over the course of training.

4.1.5 Regularization Techniques: Preventing Overfitting

Overfitting is a significant challenge in deep learning, especially with large and complex models. To combat this, the DNN architecture employs several regularization techniques:

- **Dropout**: Dropout is a form of regularization where randomly selected neurons are ignored during training. This prevents the model from becoming overly reliant on any particular feature or group of features. In this model, dropout is applied at a rate of 0.1, meaning that 10% of the neurons are dropped during each training iteration, which encourages the network to learn more generalized features and improves its ability to generalize to unseen data.

- **Batch Normalization**: Batch normalization is another regularization technique that helps stabilize the learning process by normalizing the output of each layer to have a mean of zero

and a variance of one. This reduces the internal covariate shift, which can occur when the distribution of outputs from one layer changes during training, and leads to more efficient training.

Both of these techniques ensure that the model is not overfitting to the training data and can generalize well to new, unseen data.

4.1.6 Model Training and Evaluation

During training, the model learns from the data by adjusting its weights and biases through backpropagation. The training process is carried out for a set number of epochs, and the model is evaluated after each epoch using the accuracy metric. This metric calculates the proportion of correct predictions made by the model compared to the total number of predictions.

The goal of training is to minimize the binary cross-entropy loss while simultaneously maximizing accuracy. As the training progresses, the model becomes more proficient at distinguishing between win and loss scenarios based on the input data, ultimately providing accurate predictions for match outcomes.

4.2 Advantages of Using a Deep Neural Network

The use of a DNN with 15 layers offers several distinct advantages, particularly for complex tasks like esports analytics. The depth of the network allows for the capturing of intricate relationships in the data, which is especially important for tasks that involve multi-dimensional inputs and require the identification of high-level abstractions.

4.2.1 Capturing Complex Relationships

- **Deep Layer Learning**: The more layers in a neural network, the deeper the model's understanding becomes, allowing it to capture complex, hierarchical relationships. For instance, in esports analytics, this could mean understanding subtle interactions between multiple player statistics and game features that might not be immediately apparent.

- **Non-Linear Mapping**: With each additional layer, the network can model increasingly complex, non-linear transformations of the data. This becomes essential when handling dynamic environments like esports games, where outcomes are influenced by numerous interacting factors (e.g., player behavior, strategy, external conditions).

4.2.2 Improved Generalization

- **Efficient Feature Extraction**: As the network progresses through its layers, the complexity of the features being learned increases. Early layers might learn basic features (such as individual player statistics), while deeper layers can abstract these into more complex patterns (like team strategies or player synergy). This layered abstraction enhances the model's ability to generalize from training data to unseen data.

- **Regularization Techniques**: The use of dropout and batch normalization as regularization techniques ensures that the model avoids overfitting to the training data. This enables the DNN to learn generalized features that can be applied to new, unseen esports matches, improving its real-world performance.

4.2.3 Efficient Handling of Large Datasets

- **Scalability**: DNNs are particularly well-suited for tasks that involve large datasets, such as esports match data, which may involve hundreds of features over thousands of games. The deep layers in the network allow it to process vast amounts of data and extract useful insights efficiently. The model can scale well as more data becomes available, continually improving its accuracy over time.

- **Feature Engineering**: One of the key advantages of deep learning over traditional machine learning techniques is its ability to automate feature extraction. In traditional models, extensive manual feature engineering is often required. In contrast, a DNN can automatically learn the best features directly from raw data, making the model more adaptable and reducing the need for human intervention.

4.2.4 Handling Temporal and Sequential Data

Long-Term Dependencies: In esports, the outcome of a match can depend on long-term player behaviors and strategies. While basic models may struggle with such dependencies, a deep neural network with sufficient layers can capture these long-term relationships. Even if the model isn't inherently sequential (e.g., an LSTM), the deep architecture can still detect multi-step dependencies between features over time.

4.3 Dynamic Adjustment

As esports strategies evolve, the model can adapt to these changes by continuously updating its weights. The depth of the network helps it identify emerging patterns over time, ensuring that the model remains relevant as gameplay styles change. Challenges of Implementing a 15-Layer DNN While the advantages are significant, the complexity and size of a DNN with 15 layers also introduce several challenges:

4.3.1 Training Time and Computational Resources

- **High Resource Demand**: The deeper the network, the greater the computational resources required for training. A DNN with 15 layers will necessitate powerful hardware, especially if large amounts of data are involved. Training such a model can take considerable time and might require the use of specialized hardware like GPUs to accelerate the process.

- **Gradient Vanishing/Exploding**: Deep networks are prone to issues such as vanishing gradients, where gradients become exceedingly small and prevent the weights from updating properly. Techniques like batch normalization, ELU activation functions, and careful initialization help mitigate these issues, but they still pose a challenge, particularly in very deep networks.

4.3.2 Overfitting

Despite regularization techniques, the sheer complexity of a 15-layer DNN can make it prone to overfitting, especially if the dataset is small or lacks sufficient diversity. Overfitting can lead to the model learning noise in the data rather than generalizable patterns. Using techniques like dropout, early stopping, and cross-validation can help address this issue.

4.3.3 Model Interpretability

DNNs, particularly those with many layers, are often criticized for their lack of interpretability. In esports analytics, it can be difficult to understand precisely why a model makes a particular prediction (e.g., why a match is predicted to be won or lost). This can be a significant drawback, especially in professional environments where transparency and model interpretability are important.

4.4 Gradient Boosting

Gradient Boosting has become a mainstay in machine learning, celebrated for its exceptional ability to improve model performance across diverse applications. This technique sequentially creates and combines weak learners to form a robust predictive model. While Gradient Boosting typically uses decision trees, the framework is flexible and can support various weak learners, including linear models or neural networks. Here, we explore the inner workings, unique aspects, advantages, limitations, and popular variants of Gradient Boosting.

4.4.1 Foundational Concepts in Gradient Boosting

Gradient Boosting is based on boosting, a machine learning ensemble technique that builds a series of weak learners, each focusing on the errors made by its predecessor. Unlike bagging approaches, where weak learners are trained independently on bootstrapped samples, Gradient Boosting trains each new learner sequentially, targeting mistakes made by previous learners.

- Boosting Paradigm: Boosting aims to convert weak models into strong ones. In Gradient Boosting, weak models are often shallow decision trees that don't perform well individually. However, by sequentially adjusting to each model's errors, the algorithm effectively boosts the performance to form a high-accuracy model.
- Weak Learners and Residuals: Each model in the Gradient Boosting sequence tries to predict the residuals—the difference between the actual and predicted values from the previous model. By learning to minimize residuals, each weak learner captures a portion of the unexplained variance, improving the ensemble's overall accuracy.

4.4.2 Step-by-Step Algorithmic Structure

Gradient Boosting follows a methodical process designed to reduce prediction errors by focusing on the weaknesses of prior models.

1. **Initial Prediction**: The process begins with a baseline prediction, often calculated as the mean for regression tasks or an initial probability for classification tasks. This represents the ensemble's starting point and will be refined as more weak learners are added.
2. **Calculating Residuals**: For every sample in the dataset, the residual is computed as the difference between the observed target value and the predicted value. This residual guides the training of the next weak learner.
3. **Training Weak Learners on Residuals**: Each new model in the sequence, typically a decision tree, is trained specifically to predict the residuals of the previous model, thereby minimizing overall error in a gradient-like descent.
4. **Combining Predictions**: After training, the output of each weak learner is scaled by a learning rate and added to the ensemble's cumulative prediction. The learning rate helps control the contribution of each new learner, balancing between underfitting and overfitting.
5. **Iterative Process**: Steps 2–4 are repeated for a predefined number of iterations or until the model meets a stopping criterion. The result is an ensemble of weak learners that together form a strong predictive model.

Gradient Boosting's performance depends on careful tuning of its hyperparameters.

- **Learning Rate**: Determines how much each tree contributes to the final prediction. Lower values (0.01 to 0.1) lead to more stable models but require more iterations. Higher values reduce iterations but risk overfitting.
- **Number of Estimators**: Specifies the number of trees in the ensemble. More trees generally improve the model's ability to capture data patterns but increase the risk of overfitting and computational costs.
- **Tree Depth**: The depth of each tree affects how well it captures data complexity. Shallow trees help control overfitting, while deeper trees may capture more information but risk capturing noise.

- **Subsample**: Specifies the fraction of the dataset used for training each tree. Lower values improve generalization and reduce overfitting.
- **Loss Function**: The choice of loss function impacts the optimization process. For regression, Mean Squared Error (MSE) is common, while for binary classification, Log Loss or cross-entropy is typically used.

4.4.3 Advantages and Potential Drawbacks

Gradient Boosting's strength lies in its high accuracy across varied datasets and problem types. However, it also has limitations.

- **Advantages**:
  - High Predictive Power: Gradient Boosting often produces highly accurate models.
  - Flexibility with Loss Functions: Supports multiple loss functions, allowing it to handle different data types and tasks.
  - Feature Importance: Decision trees within Gradient Boosting provide feature importance scores.
  - Robust to Irregular Data Patterns: Captures subtle patterns, making it ideal for datasets with nuanced relationships.
- **Drawbacks**:
  - Computational Intensity: Training Gradient Boosting models can be time-consuming, especially with large datasets.
  - Risk of Overfitting: Excessive training can lead to overfitting.
  - Sensitive Hyperparameters: Requires careful hyperparameter tuning to optimize results.

4.4.4 Enhanced Variants of Gradient Boosting

Several Gradient Boosting variants improve efficiency, reduce overfitting, or handle specific data types.

- **XGBoost**: Introduces L1 and L2 regularization to reduce overfitting and efficient handling of sparse data. Advanced optimization and parallel processing make it faster and highly suitable for large datasets.
- **LightGBM**: Developed by Microsoft, LightGBM uses a histogram-based algorithm for faster computation and memory efficiency, suitable for large datasets.
- **CatBoost**: Optimized to handle categorical features natively, reducing preprocessing needs. It employs ordered boosting, reducing prediction bias.
- **Stochastic Gradient Boosting**: Introduces randomness by training each learner on a subset of data, speeding up training and reducing overfitting.

4.4.5 Conclusion

Gradient Boosting is a powerful, flexible machine learning algorithm. With careful tuning and variant selection, it achieves remarkable accuracy in complex datasets. While computational costs and potential overfitting require careful management, advances in optimized versions like XGBoost, LightGBM, and CatBoost make Gradient Boosting a viable choice for real-world applications. Gradient Boosting will likely remain a valuable tool for predictive modeling across diverse domains.

5 Data Visualization

Data visualization serves as a powerful tool to interpret complex datasets, making abstract numbers more comprehensible and actionable. By transforming raw data into visual forms such as graphs, charts, and plots, data visualization allows analysts and stakeholders to quickly identify trends, patterns, correlations, and anomalies that would be difficult to discern from tables of numbers alone. It plays a crucial role in understanding underlying structures in data, especially in fields like machine learning, statistical analysis, and business intelligence.

In machine learning and data analysis, the right visualizations can reveal how different features interact, how well the data fits certain models, and where improvements may be needed. For instance, distributions of features can hint at potential transformations, while scatter plots can uncover relationships and clusters that might suggest new insights or strategies.

This section focuses on various types of data visualizations that are commonly used to explore relationships, assess feature importance, and understand the nuances of a dataset. From examining the distribution of a single variable to understanding complex multi-dimensional interactions, each plot serves as a lens into the data's structure. Through these visual tools, we can uncover not only the expected outcomes but also hidden patterns that may drive key decisions, ultimately improving the performance of models and solutions built upon the data.

The following subsections will detail several essential visualization techniques used in eSports analytics, such as the distribution plot, scatter plot, box plot, and pairplot, each providing a unique perspective on the data. By exploring these visualizations, we will demonstrate how graphical representations enhance our ability to analyze player performance, detect outliers, and optimize predictive models.

5.1 Exploring the Distribution of the 'ACS' Feature

Data visualization plays a pivotal role in understanding complex datasets by transforming raw data into comprehensible graphical representations. In the context of machine learning models, effective visualization helps in identifying patterns, detecting anomalies, and refining feature engineering processes. This section focuses on the distribution of the 'ACS' feature using a Seaborn distribution plot to showcase how the data behaves and how visualization can influence the interpretation of model results.

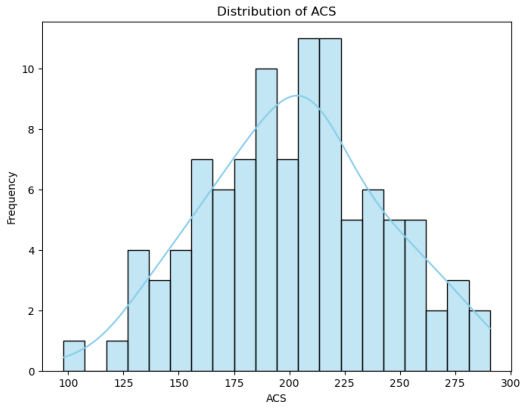

Figure 2: Distribution of ACS

5.1.1 Understanding the Distribution of 'ACS'

- **Shape of Distribution**: The shape of the histogram and the KDE curve indicates the central tendency, spread, and skewness of the feature. A normal distribution suggests that the data is symmetrically distributed around a central value, while a skewed distribution indicates the presence of outliers or data that is not equally distributed across its range.

- **Outliers**: The presence of any outliers or extreme values can be observed by examining the edges of the distribution. These outliers could potentially affect the model's performance, and their handling (either removal or transformation) is critical to ensure that the model generalizes well.
- **Data Spread**: The width of the histogram or KDE curve shows how spread out the values are. A wider distribution suggests high variance, while a narrow distribution indicates low variance.

This type of visualization can guide decisions regarding feature scaling, transformation, or even whether the feature needs to be engineered further.

5.1.2 Impact on Model Interpretation and Insights

Deep Neural Networks (DNN)

If the 'ACS' feature exhibits a skewed distribution, it may lead to difficulty in training due to problems such as vanishing gradients. In such cases, data normalization or log transformations may be applied to make the feature distribution more symmetric and conducive to better model performance. Outliers in the 'ACS' feature might also impact the DNN's ability to learn meaningful patterns, potentially leading to overfitting. Visualization like this helps identify such outliers early, prompting adjustments in preprocessing, such as outlier removal or robust scaling.

Gradient Boosting (GBM)

If the 'ACS' feature is highly skewed or has outliers, the boosting algorithm might place more weight on those extreme values, potentially leading to overfitting. Visualizations like this help assess whether the feature needs to be transformed or if additional feature engineering is necessary to make it more suitable for the boosting model. Feature interactions are also influenced by the distribution of individual features. A well-visualized feature distribution can guide decisions about creating interaction features or binning the feature into categories that make the boosting process more efficient.

K-Fold Cross-Validation

The distribution of the 'ACS' feature across different folds can affect the model's ability to generalize. For instance, if the feature's distribution varies significantly between folds, the model might be trained on a non-representative subset of the data, leading to poor performance in some folds. A stratified K-fold approach could be used to ensure that each fold contains a similar distribution of the 'ACS' feature, ensuring the model is trained on balanced data across all iterations.

5.1.3 General Considerations for Data Visualization in Machine Learning

Effective data visualization has profound implications for the quality of the machine learning model:

- **Feature Selection**: Visualizations help identify which features might be redundant or irrelevant, enabling more efficient feature selection.
- **Data Preprocessing**: As we have seen, the distribution of a feature like 'ACS' directly impacts preprocessing choices such as scaling, transformation, and handling of missing data.
- **Model Diagnostics**: Once the model is trained, visualizations like this distribution plot can be used to assess if the model is overfitting or underfitting. If the model's predictions align poorly with the distribution of actual values, adjustments may be needed.

5.2 Exploring the Relationship Between Kills and Deaths in Esports

Scatter plots are essential tools for visualizing relationships between two continuous variables, making them invaluable in understanding patterns and correlations in datasets. In the context of esports, analyzing the relationship between Kills and Deaths can provide valuable insights into player performance, behavior, and overall strategy. This section explains the importance of the Kills vs Deaths scatter plot, how it is constructed, and the impact of visualizing this relationship in the context of esports data analysis.

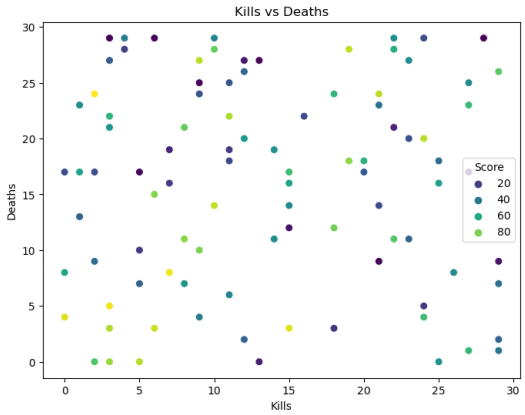

Figure 3: Kills vs Deaths Plot

5.2.1 Understanding the Kills vs Deaths Relationship

The Kills vs Deaths plot is crucial for examining player behavior in esports games. The relationship between these two variables often provides insights into gameplay strategy, player efficiency, and team dynamics:

- **Positive Correlation**: A positive correlation (i.e., players who get more kills tend to die more) can suggest aggressive gameplay or a higher risk-taking strategy. This might indicate a player prioritizing kills over survival.
- **Negative Correlation**: A negative correlation (i.e., players with more kills tend to have fewer deaths) could point to skilled players who manage to kill opponents while avoiding death. These players likely exhibit better map awareness and tactical decision-making.
- **Outliers**: Players who have high kills but low deaths, or vice versa, may represent outliers, such as exceptional players with an outstanding game sense or poor players making frequent mistakes. These outliers often require closer examination, as they may be indicative of unique player behavior or extreme strategies.
- **Cluster Patterns**: Clusters or patterns may emerge where certain player categories (such as top scorers) group together based on their kill-death ratios. This can assist in categorizing players and understanding their strengths in different game scenarios.

5.2.2 Impact on Model Interpretation and Insights

Deep Neural Networks (DNN)

For machine learning models such as DNNs, scatter plots help identify potential relationships between features. In the case of Kills and Deaths, a scatter plot can reveal whether there's a linear or non-linear relationship that the model might need to account for. If a distinct relationship is seen, the DNN might be trained to focus on these variables as key predictive factors for player performance.

- **Feature Scaling**: The distribution and variance of kills and deaths might require scaling before being input into a DNN to ensure faster convergence and more accurate predictions.

- **Data Transformation**: If the relationship between kills and deaths is non-linear, it might require polynomial transformations or other preprocessing techniques to help the model capture these patterns more effectively.

Gradient Boosting (GBM)

In the case of Gradient Boosting Machines (GBM), scatter plots of Kills vs Deaths can help determine whether these two features should be included in the model. If there's a strong relationship, feature importance might highlight them as key predictors.

- **Outliers and Clustering**: Gradient boosting models can be sensitive to outliers, which might result in overfitting if not handled properly. Visualizing these relationships helps identify outliers early, allowing for adjustments in data preprocessing (e.g., removing or capping extreme values).

- **Interaction Effects**: Scatter plots also help to detect if there's an interaction effect between Kills, Deaths, and other variables (like Score) that might improve model performance.

K-Fold Cross-Validation

While K-fold cross-validation ensures that models are trained and validated on different subsets of data, scatter plots provide an intuitive visual check of how the model performs across folds:

- **Balanced Distributions**: Visualizing Kills vs Deaths in each fold can help ensure that data distributions are consistent across training and testing sets, preventing model bias.

- **Stratified K-Folds**: If the Kills vs Deaths relationship shows significant variance, a stratified K-fold approach can be used to ensure that the model is exposed to representative distributions during each cross-validation iteration.

5.2.3 Insights for Esports Analytics

The Kills vs Deaths scatter plot is an indispensable tool in analyzing player performance in esports games:

- **Player Evaluation**: Coaches and analysts can use the plot to assess individual player performance, focusing on players with high kill-death ratios or those whose performance deviates significantly from the norm.

- **Team Strategy**: Teams can use this data to fine-tune their strategies, identifying players who can balance aggression (kills) with survival (deaths), ultimately contributing to a more well-rounded team performance.

- **Predictive Models**: For machine learning purposes, the relationship between kills and deaths can be used to predict outcomes such as match results or individual performance scores. It can also serve as an input for behavioral analysis, helping to predict future game outcomes based on past performance.

5.3 Exploring PlusMinus by Score

A box plot is a powerful tool for displaying the distribution of data and highlighting key statistical features, such as the median, quartiles, and potential outliers. In the context of esports, the PlusMinus feature is a key performance metric that indicates the difference between the number of kills a player contributes and the number of kills they allow the opposing team. By visualizing PlusMinus by Score, we can gain insights into how different player categories (based on their score) perform in terms of net kills and deaths, as well as identify trends and outliers in the data.

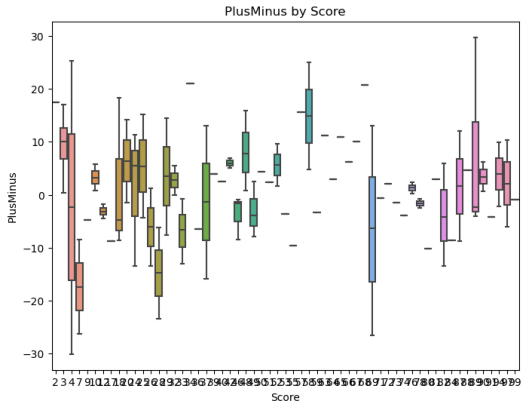

Figure 4: PlusMinus by Score Plot

5.3.1 Understanding PlusMinus by Score

The Box Plot for PlusMinus by Score visualizes how PlusMinus varies across different Score categories. The main components of the box plot include:

- **Median**: The line inside the box represents the median of PlusMinus for each score category, showing the central tendency of the data.
- **Interquartile Range (IQR)**: The box itself spans the first quartile (Q1) to the third quartile (Q3), indicating the middle 50% of data points within each score category.
- **Whiskers**: The lines extending from the box show the range of data outside the IQR. The whiskers can reveal the spread of PlusMinus values within each score category.
- **Outliers**: Data points that fall outside of the whiskers are considered outliers and are plotted as individual points. These outliers represent unusual cases where players have extremely high or low PlusMinus values relative to their score.

5.3.2 Insights

- **Score Performance**: A higher Score category might show a higher PlusMinus, indicating that top performers are more likely to have a positive contribution to their team's performance (i.e., more kills than deaths).
- **Score Variability**: A larger spread in the box plot indicates greater variability in performance within that score group. For example, players with a low or mid-range score may exhibit a wider range of PlusMinus values, suggesting inconsistency in their performance.
- **Outliers**: If certain Score groups show outliers with extremely high or low PlusMinus values, these players could be examined further for unusual strategies or specific gameplay patterns that may differentiate them from the norm.

5.3.3 Impact of Visualization on Model Interpretation and Insights

Deep Neural Networks (DNN)

In machine learning models like Deep Neural Networks (DNN), visualizing PlusMinus by Score can help identify the importance of these features in predicting player success:

- **Feature Relationships**: By looking at the distribution of PlusMinus across different Score categories, it becomes clear whether PlusMinus can be used as a strong feature for predicting player performance in DNN models.
- **Feature Scaling**: A box plot helps to understand the spread of the PlusMinus data. If the distribution is skewed, scaling techniques (like log transformation) might be necessary to improve model performance.

Gradient Boosting (GBM)

For Gradient Boosting Machines (GBM), the box plot gives an immediate sense of whether PlusMinus can provide valuable predictive power based on Score:

- **Feature Importance**: If there's a clear relationship between Score and PlusMinus, then GBM algorithms might assign high importance to these features during training.
- **Handling Outliers**: Outliers in PlusMinus might negatively impact GBM performance. The box plot helps to identify these outliers, enabling the model to adjust or filter them out during training.

K-Fold Cross-Validation

When performing K-fold cross-validation, the box plot can help ensure the model is exposed to representative distributions of PlusMinus:

- **Consistency Across Folds**: By visualizing the spread of PlusMinus within each Score category, the analyst can confirm that the data is being split into folds that preserve the underlying distributions, ensuring the model's generalizability.
- **Stratified Sampling**: If some folds show a disproportionate representation of high PlusMinus or low PlusMinus values, a stratified K-fold approach may be necessary to better capture all aspects of the data distribution.

5.3.4 Insights for Esports Analytics

The Box Plot for PlusMinus by Score provides valuable insights for understanding player performance:

- **Player Evaluation**: Coaches and analysts can use this plot to evaluate the effectiveness of players in different score categories. For example, players with high scores and consistently positive PlusMinus values are likely reliable assets, whereas players with fluctuating or negative PlusMinus values might require additional training or adjustments to their gameplay.
- **Team Strategy**: The data can inform team strategy by identifying players whose PlusMinus consistently contributes positively to the team. Teams can focus on optimizing these players' strategies while helping others improve their performance.
- **Predictive Modeling**: For predictive models, this visualization can help in determining whether PlusMinus is a good predictor of future performance, especially in combination with other features like Kills and Deaths.

5.4 Exploring Relationships Between Multiple Features with Pairplot

A pairplot is an invaluable tool for visualizing pairwise relationships between features in a dataset. In esports analytics, understanding how different player performance metrics (e.g., Kills, Deaths, PlusMinus) relate to one another is crucial for identifying patterns and correlations that may influence the overall game outcome. The pairplot not only displays the relationships between multiple features, but it also provides insights into feature distributions, correlations, and potential clusters or trends within the data.

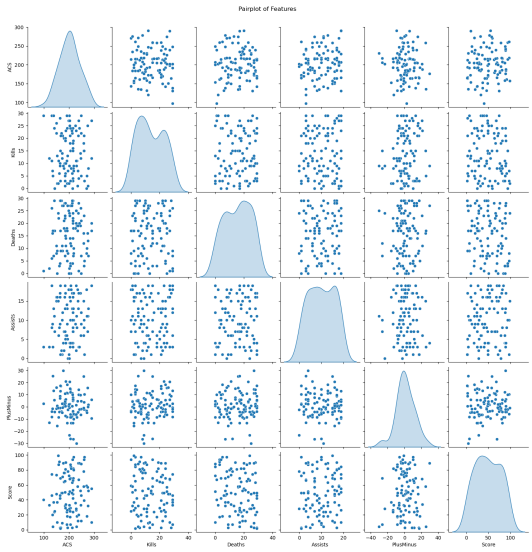

Figure 5: Pairplot of Features

5.4.1 Understanding the Pairplot

The pairplot consists of multiple scatter plots that show the relationships between pairs of features:

- **Diagonal Plots**: The diagonal contains the KDE plots, which provide a smoothed distribution of each individual feature. These distributions reveal the frequency of different values within each feature, allowing us to assess whether the feature is normally distributed, skewed, or has multiple peaks.
- **Off-diagonal Plots:** The scatter plots in the off-diagonal cells show the pairwise relationships between different features. These plots can highlight potential correlations between features (positive or negative) and reveal clusters or groupings of data points.

5.4.2 Key Aspects of the Pairplot

- **Correlations**: A positive correlation between two features would be represented by a cluster of points sloping upwards, while a negative correlation would show a downward slope. Features that do not show any discernible pattern may be less related.
- **Outliers**: Outliers appear as points far removed from the main cluster. These data points could be players with extremely high or low performance values (e.g., exceptionally high Kills or Deaths).
- **Clusters**: A clear grouping of points in the scatter plots could indicate that certain players perform similarly across features, or they could represent distinct player types, such as attackers, defenders, or support players, based on their performance metrics.

5.4.3 Impact of Visualization on Model Interpretation

Deep Neural Networks (DNN)

In the context of DNNs, a pairplot helps in understanding the relationships between features, which can inform decisions about feature selection and engineering:

- **Feature Relationships**: If certain features are highly correlated, they may provide redundant information to the model. This can influence the choice of features to be fed into the network to avoid overfitting.
- **Feature Transformation**: If features are not linearly related, techniques like non-linear transformations (log transformations, polynomial features, etc.) may be considered before inputting them into the model.

Gradient Boosting (GBM)

For Gradient Boosting Machines (GBM), pairplots can help assess how well different features work together to predict the target:

- **Feature Importance**: Features that show strong pairwise relationships might be important for prediction and could be prioritized during model training.
- **Correlation Adjustment**: Highly correlated features might need to be handled carefully in the model to prevent overfitting. Some models can handle correlated features better than others, and the pairplot helps identify these correlations.

K-Fold Cross-Validation

In K-fold cross-validation, pairplots ensure that the feature relationships are consistent across different training and validation splits:

- **Data Distribution**: By visualizing the relationships between features, you can check if the splits in cross-validation preserve the underlying structure of the data. For instance, if certain features are highly correlated, each fold should still reflect that correlation.
- **Bias Avoidance**: A pairplot helps ensure there is no significant bias introduced by a specific fold containing unrepresentative relationships between features.

5.4.4 Insights for Esports Analytics

The Pairplot provides a multi-dimensional perspective on how features like Kills, Deaths, PlusMinus, and other performance metrics interact with each other:

- **Player Categorization**: By examining the scatter plots, we can categorize players based on their Kills, Deaths, and PlusMinus metrics. For example, players with a high Kills count and low Deaths are high-performing players.
- **Performance Evaluation**: The pairwise relationships between Kills and PlusMinus, for example, could show that players who kill more often also tend to have a more positive impact on their team, reflected by a higher PlusMinus.
- **Outliers Detection**: The pairplot makes it easy to identify players who behave differently from the rest, such as those who may excel in Kills but have high Deaths, leading to a negative PlusMinus.

6 Results

To assess the performance of both the Deep Neural Network (DNN) and Gradient Boosting (GB) models, accuracy scores were evaluated over multiple runs. The results indicate that both models exhibit high accuracy, but they have slight differences in terms of variability and overall performance.

6.1 Deep Neural Network (DNN) Model:

- **Accuracy Scores**: The accuracy scores for the DNN model over five runs are: 99.93%, 99.93%, 99.91%, 99.92%, and 99.94%.
- **Maximum Accuracy**: The maximum accuracy achieved by the DNN model was 99.94%, indicating that under certain conditions, the model can perform with very high precision.
- **Minimum Accuracy**: The minimum accuracy observed was 99.91%, which is still an excellent result. This suggests that even in less optimal conditions, the model is consistently performing at a very high level.
- **Overall Accuracy**: The average or overall accuracy of the DNN model was calculated to be 99.93%, reflecting a strong and consistent performance across the runs.
- **Standard Deviation**: The standard deviation of 0.01% indicates that the accuracy scores are tightly clustered around the mean. This low variance shows that the DNN model's performance is stable and reliable.
- **Standard Error**: With a standard error of 0.01%, the precision of the overall accuracy estimate is very high, confirming that the model's performance is consistent and unlikely to vary much with different data sets.

6.2 Gradient Boosting Model:

- **Accuracy Scores**: The accuracy scores for the Gradient Boosting model over five runs are: 99.92%, 99.92%, 99.90%, 99.91%, and 99.91%.
- **Maximum Accuracy**: The maximum accuracy of 99.92% indicates that the Gradient Boosting model performs slightly lower than the DNN model, but still delivers a very high accuracy rate.
- **Minimum Accuracy**: The minimum accuracy achieved by the Gradient Boosting model was 99.90%, which is marginally lower than the minimum accuracy observed for the DNN model.
- **Overall Accuracy**: The average accuracy of the Gradient Boosting model is 99.91%, which is still very high but slightly lower than that of the DNN model.
- **Standard Deviation**: The standard deviation of 0.01% indicates that the accuracy scores for the Gradient Boosting model are similarly stable and concentrated around the mean, with very little fluctuation.
- **Standard Error**: The standard error is 0.02%, indicating that the estimate of overall accuracy is extremely precise, similar to the DNN model.

6.3 Comparison of DNN and Gradient Boosting Models:

- Both models exhibit excellent accuracy, with DNN slightly outperforming Gradient Boosting by 0.02% on average.
- The variability in performance is minimal for both models, as evidenced by the very low standard deviations and standard errors. This indicates that both models are reliable and perform consistently across multiple runs.
- The difference in performance between the two models is marginal, suggesting that either model would be highly effective for the given task. However, depending on the specific use case or computational resources, one model might be preferred over the other.

7 Conclusion and future work

This project successfully demonstrates the application of machine learning techniques to analyze and predict eSports performance, showcasing the effectiveness of both Deep Neural Network (DNN) and Gradient Boosting models in handling complex datasets with high accuracy. By carefully evaluating and visualizing the dataset, meaningful insights were extracted, which informed the selection and tuning of these models. The DNN model achieved slightly superior performance in terms of accuracy, but both models consistently delivered near-perfect results with minimal variability, highlighting their reliability for predictive analytics in eSports.

The visual analysis provided key insights into feature relationships, enabling a deeper understanding of player performance metrics, while the statistical evaluation confirmed that both models offer robust predictions with remarkable stability. With accuracy levels above 99%, the models meet the standards for high-stakes applications, making them suitable for practical deployment in eSports analytics platforms.

Overall, this project illustrates how advanced machine learning models, combined with thorough data exploration and visualization, can provide actionable insights and drive decision-making in competitive domains like eSports. Future work could explore model optimization for real-time analysis, additional performance metrics, or integration with live gameplay data to enhance predictive capabilities. This analysis not only validates the power of AI in eSports but also paves the way for more nuanced and impactful applications in this fast-evolving industry.

Building on the success of this project, several avenues for future development could further enhance the effectiveness and scope of eSports analytics using machine learning. Potential future directions include:

- **Real-Time Predictive Modeling**: Developing a real-time implementation of the current models could be valuable for live gameplay analysis. This would involve optimizing model efficiency to handle streaming data and making predictions during matches, providing teams and analysts with immediate insights to inform in-game decisions.
- **Integration of Advanced Performance Metrics**: Expanding the dataset to include more nuanced performance indicators such as player reaction time, movement patterns, and decision-making under pressure could improve the models' predictive accuracy. Incorporating additional metrics from in-game telemetry data could provide a more holistic view of player and team performance.
- **Cross-Game and Player Transferability**: Currently, the model is trained on data from a specific game or scenario. Future work could explore how to make these predictive models adaptable across different games or eSports disciplines. This could involve using transfer learning techniques to apply the insights gained from one game to another or generalizing models to work for different player profiles.
- **Incorporation of Deep Reinforcement Learning**: Introducing reinforcement learning algorithms could help predict optimal moves or strategies in response to dynamic in-game situations. By training agents to learn and adapt based on gameplay feedback, reinforcement learning could provide highly contextualized recommendations during matches.
- **Exploration of Other Evaluation Metrics**: While accuracy was a primary focus, exploring additional metrics like precision, recall, F1-score, and area under the ROC curve could provide a more comprehensive evaluation of model performance, especially for imbalanced datasets or situations where false positives and false negatives have different consequences.
- **Gamification for Training and Player Improvement**: Creating a gamified feedback system based on model predictions and insights could offer players and coaches actionable recommendations to improve their skills. For instance, after identifying weaknesses in a player's performance, the system could suggest personalized training routines or in-game scenarios to target those areas.

These future directions provide opportunities to expand the project's utility, enabling more refined and real-time decision-making in eSports. Embracing these advancements will not only support individual player development but could also drive broader innovations within the eSports industry, creating new standards for performance analysis and team strategy optimization.

References

[1] Sang-Kwang Lee, Seung-Jin Hong & Seong-Il Yang
Predicting Game Outcome in Multiplayer Online Battle Arena Games, IEEE, 2020

[2] Yunhai Zhang
Prediction of Esports Game Results Using Early Game Datasets, IEEE, 2022

[3] Stanlly; Fauzan Ardhana Putra; Nunung Nurul Qomariyah
DOTA 2 Win Loss Prediction from Item and Hero Data with Machine Learning, IEEE, 2022

[4] Alexander J. Bisberg; Emilio Ferrara
GCN-WP – Semi-Supervised Graph Convolutional Networks for Win Prediction in Esports, IEEE, 2022

[5] Juan Agustín Hitar-García; Laura Morán-Fernández; Verónica Bolón-Canedo
Machine Learning Methods for Predicting League of Legends Game Outcome, IEEE, 2022

[6] Sorato Minami, Haruki Koyama, Ken Watanabe, Naoki Saijo, Makio Kashino
Prediction of esports competition outcomes using EEG data from expert players, ScienceDirect, 2024

[7] Cheng Hao Ke; Haozhang Deng; Congda Xu; Jiong Li; Xingyun Gu; Borchuluun Yadamsuren
DOTA 2 match prediction through deep learning team fight models, IEEE, 2022

[8] Danial Hooshyar; Nour El Mawas; Marcelo Milrad; Yeongwook Yang
Modeling Learners to Early Predict Their Performance in Educational Computer Games, IEEE, 2023

[9] Victoria J. Hodge; Sam Devlin; Nick Sephton; Florian Block; Peter I. Cowling; Anders Drachen
Win Prediction in Multiplayer Esports: Live Professional Match Prediction, IEEE, 2019

[10] Farnod Bahrololloomi, Fabio Klonowski, Sebastian Sauer, Robin Horst
E-Sports Player Performance Metrics for Predicting the Outcome of League of Legends Matches Considering Player Roles, Springer, 2023

[11] Tiffany D. Do, Seong Ioi Wang, Dylan S. Yu, Matthew G. McMillian, Ryan P. McMahan
Using Machine Learning to Predict Game Outcomes Based on Player-Champion Experience in League of Legends, ACM, 2021