CAPSTONE PROJECT 1

Planning Document

machine learning technique for predicting a suitable champion for new players

by

Abdullah Faisal Khaled Bin Madhi

15085236

Bachelor in information system (Data Analysis)(BSDA)

Supervisor: Dr Samuel Mofoluwa Ajibade

Semester: September 2024

Date: 8 August, 2025

School of Computing and Artificial Intelligence

Faculty of Engineering and Technology

Sunway University

Chapter 1

Introduction

1.1 Overview

League of Legends is a multiplayer online battle arena (MOBA) game that has become one of the most popular games in the world, having more than 170 champions with different abilities and playstyle. Choosing a champion that suits their personal preferences, playstyle, and behavioral tendencies can be a daunting task to new players. The current recommendation tools are mainly aimed at maximizing the win rates or optimizing the draft compositions with little to no consideration of the subjective preferences and psychological characteristics of the beginners.

The majority of the currently available recommendation tools in League of Legends are aimed at the increase in win probability. They tend to be based on ranked game data and are largely concerned with drafting optimization, counter-picks, meta-relevance or statistical performance. Although these features can be helpful to competitive or experienced players, they are not of much assistance to beginners who might not be familiar with champion mechanics, meta dynamics, or role expectations. In addition, these tools tend to overlook the subjective elements that determine enjoyment, e.g. whether a player likes fast action, strategic planning, flashy powers, or support roles.

A number of studies have discussed the use of artificial intelligence (AI) and machine learning (ML) algorithms in champion recommendation. Shen et al. (2022) suggested a sequential recommendation model of ban-pick stages based on Bi-directional Long Short-Term Memory (Bi-LSTM) and knowledge graphs. This model took into consideration the order of the picks in time, and it forecasted the best champions to be chosen depending on the previous picks made in the draft. Their strategy focused on the significance of time and synergy, but not on individual preferences of players. In the same way, Horst et al. (2024) created DraftComPromise, a real-time recommendation system, which examined various draft-related variables, including champion win rate, team synergy, damage type balance, and counter-matchups. Their tool would make champion suggestions using statistics, providing information on how some combinations made a team more likely to win. Nevertheless, their work, similar to that of many others, focused on making the most of competitive advantage instead of personalizing the experience of new or casual players.

Despite the recent advances, there is an interesting gap that few recommendation systems attempt to understand and incorporate the real personality, behavior, and psychological tendencies of a player into their recommendations. It means a lot to new players, whose enjoyment level and whether they stay with the game can depend far more on how well a champion fits their individual playstyle than on the headline figures such as win rate or clever drafting tactics.

1.2 Motivation of Study

The background demonstrates that the majority of the current champion recommendation systems in the League of Legends are based on maximizing prediction of wins or draft optimization, and that they can also suggest counterpicks or synergistic team compositions (e.g., Shen et al., 2022; Horst et al., 2024). Although such direction is good to competitive players, it does not take into account the demands of new or casual players, whose main problem is not to create a perfect draft synergy but to find champions that satisfy their personal preferences and playstyle. To a novice, choosing a champion to play a game might not be that difficult, but knowing which champion best suits him/her behavioral pattern, goal and comfortability is much more complicated.

It is compounded by the fact that there are so many champions to choose between, all with their own roles, abilities, and learning curves, which can be rather daunting to new players in the game to find a good fit. Also, the current tools of recommendations seldom involve psychological or behavioral profiling, including the degree of aggression, risk-seeking behavior, or teamwork orientation, which is a key aspect of long-term player retention. Filling this gap has the potential to not only enhance the level of satisfaction in the game at the beginning of the gameplay but also increase the retention of players as well as decrease the level of frustration in the onboarding process.

Thus, the proposed research develops a recommendation system, which is personalized and analyzes the behavioral and psychological characteristics of a player based on the inputs in the form of questionnaires. This system will give more significant champion recommendations by prioritizing personal traits instead of only win-oriented indicators. This method is not comparable to previous researches, since it combines the objective data of the game and subjective tastes of the players and eventually leads to a more player-centered gaming experience.

1.3 Problem Statement

There are already plenty of research teams that attempted to recommend draft picks in League of Legends based on machine-learning or statistical methods. Shen et al. (2022) addressed the ban-pick sequence problem using Bi-LSTM and knowledge graphs, and their model scored an even better coverage and user satisfaction. The twist is that it is targeted at veteran players who care about team compositions at the expense of what new players care about. Horst et al. (2024) came in with DraftComPromise by considering synergy, type of damage, and win rate of champions to refine draft results- still missing out on psychological or behavioral preferences. Masciari et al. (2024) published a complete survey of recommenders based on AI and observed the same trend: there are lots of systems that tune technical performance, but few systems tune to the personality of the user. In essence, although the game-specific suggestions are pretty well covered, the issue of selecting champions to new players according to their personal characteristics and interests remains open. My project aims to address that gap by creating a web app that will request the user to complete a survey-like input

form and then use machine learning to recommend their most compatible champions, so that the onboarding process in League of Legends will become friendlier and more personal.

1.4 Research Questions

1. How to map player behavioral and psychological traits to preferences in League of Legends champions?
2. What machine learning techniques can be used to effectively forecast appropriate champions to be used by new players?
3. What is the way to connect the personality and behavioral characteristics of a player with appropriate champion characteristics in League of Legends?

1.5 Research Aim and Objectives

The aim of this study is to develop a website-based machine learning model to analyze and recommend League of Legends champions to novice players. The objectives of the study are:

1. To identify the differences between the player behavioral and psychological characteristics and their connection with champion features in League of Legends.
2. To design and implement a machine learning model that can recommend champions using questionnaire data as input.
3. To establish a model that would connect the personality and behavioral characteristics of players with the champion's features in League of Legends.

1.6 Scope of the Study

The research examines League of Legends and attempts to assist new players in selecting champions. It does so by creating a site that questions players on their game style, behavior and what they like.

The information that will be utilized in the research will be categorized into two broad groups. The first is the champion data, which is the detailed attributes of each champion in the

League of Legends. These are their main role (e.g. tank, assassin, mage, marksman, support), playstyle traits (e.g. aggressive burst damage, poke, crowd control, sustain, or utility), and skill level or difficulty rating, which indicates the mechanical or strategic complexity of the champion to play. There are champions that are easy to play and some that demand a lot of knowledge in the game or quick reflexes. This data are the basis in terms of which it is possible to comprehend what kind of champions are available in the game and what types of player features they usually attract. The second dataset is the player survey that will be gathered directly via a web-based survey. Players will be asked questions regarding their playstyle preference (e.g. passive vs. aggressive), their tolerance to risk, their decision-making style (e.g. impulsive vs. strategic), their interaction with the team (e.g. leading, supporting, roaming), and even their personal preferences such as whether they prefer flashy characters, dark themes or underdog stories. Based on these subjective answers a player profile is constructed which reflects how the player tends to act naturally and what they will tend to enjoy in the game.

It is vital to use both sets of data. The champion data will give objective data on what each champion has to offer and the survey data will give insight on who the player is. Comparing the two, the system will be able to make meaningful matches, suggesting champions not based on meta strength, but on compatibility. This two-data model will give the machine learning algorithm the ability to identify connections between player characteristics and champion features to make precise, personalized suggestions that will increase enjoyment and minimize frustration among new players.

The study targets players in Southeast Asia, particularly in Malaysia, although the system can be applicable in other areas. It does not aim to propose some ideal drafts that can be played in ranked or competitive games, but just to assist new or casual players in choosing new champions.

1.7 Expected Outcome

- A functioning website that new players can use to answer preference-based questionnaires

- A machine learning model capable of recommending a list of suitable champions based on these responses

- Evidence that such a user-centered approach improves new player engagement and satisfaction compared to conventional game-focused recommenders

Chapter 2

Literature review

2.1 Introduction

The current literature review is a thorough review of the existing literature on machine learning (ML) with a focus on fundamentals, predictive use, and applicability of the topic to the modern study of champion recommendation in video games, including League of Legends. Its main goal is to clarify the present situation in the machine learning field and to show how it can be applied in various spheres, especially in predictive and recommender systems. In this way, the review contextualizes the present study in the wider academic environment and explains what gaps it will fill.

2.2 Overview of machine learning techniques

According to Parmer (2023), machine learning is a part or subset of artificial intelligence that allows the computers to learn from experience based on the data being fed without the need of being programmed. Additionally, Parmer (2023) mentions that Machine learning utilizes algorithm to enable the systems to analyze data, detect patterns, and make decisions or predictions. The use case of machine learning can be from cars like self driving cars to personalized recommendations like champion recommendation which in this case by answering few questions inorder to gather data and feed it to machine learning to train and predict.

Alzubaidi et al. (2021), mentions that machine learning is mostly training a model on a specific datasets to it can identify patterns and relationships. After that, the model can be deployed to make predictions and recommendations. Furthermore, Alzubaidi et al. (2021) highlights that machinel learning rapid growth is due to large datasets and advanced computation power which enabled more accurate and advanced models.

Bdair (2024) illustrates based on Fig1 that machine learning pipeline consist of 9 stages that provide a structured approach to building consistent models. It starts with Data collection which is raw data collected from different sources such as user input, sensors, and databases. Second, data preprocessing in which the data is transformed and cleaned from missing values and nurmalized for more consistency. Third step is feature engineering in which selecting the necessary features is necessary to represent the data. This is crucial to enhance the accuracy of machine learning model to learn the patterns. Fourth, model training where algorithms for machine learning is applied to the data in order to identify the reltionships between the input features and target outputs. Fifth, the model will be evaluated by using performance metrics such as accuracy, precision, and recall to figure the model ability and effectiveness. The sixth step is necessary if the last step not performing as expected in which will undergo in hyperparameter tuning. This step, will tweak the model settings in order to enhance its performance and rescue the risk of overfitting. After optimizations, the seventh step is going to be deployment which can be deployed and integrated into systems or applications. Finally the eighth and ninth steps are monitoring and maintenance where it involves in a countiuouse tracking of the model's performance, retraining, and updating it to fix the potential performance degradation.

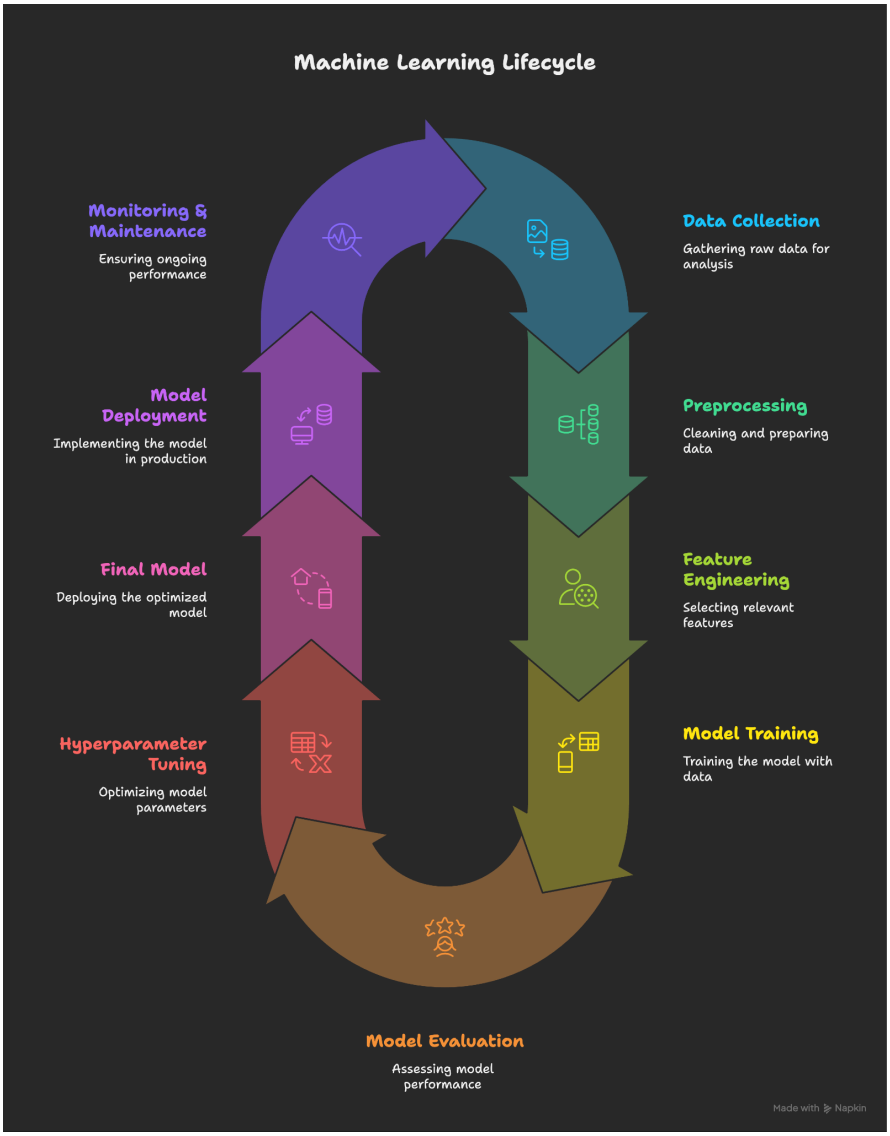


Fig 2.1 The picture describes one of the Machine Learning Lifecycle, a cyclic process, consisting of data collection, preprocessing and model training to being deployed, monitored and maintained.

2.2.1 Different techniques of machine learning

Rathod (2024) classified machine learning into four paradigms that each technique has different uses based on data scenarios and problem domains. The first paradigm is supervised learning, which used data that are labeled to train and teach the system how to match inputs with the correct outputs. The techniques that supervised learning uses are linear models, decision trees, neural networks, and support vector machines. These techniques are meant to be used for

classification and regression, also it includes application like image recognition and language understanding. The second paradigm is unsupervised learning, which is the opposite of supervised by identifying patterns without predefined labels. The techniques used in unsupervised learning are clustering such as K-means, and dimensionality reduction methods like PCA and t-SNE. The purpose of unsupervised learning is to discover hidden structures which is useful for customer segmentation and bioinformatics. The third of the paradigms, described by Rathod (2024), is the semi-supervised learning that comes closer to a mixture of supervised and unsupervised approaches. It takes in partially labeled datasets i.e. it uses a few labeled data and a large unlabeled data in order to improve accuracy when it comes to learning. This method is especially favored where data labeling is costly or time-consuming like with speech recognition and medical diagnosis. Self-training, co-training and graph-based models are commonly used on semi-supervised learning, where the model can learn pattern through both structure indication of a label and the inference of latent data patterns. The fourth paradigm, reinforcement learning is very different to the preceding paradigms. Instead of training on labeled data, reinforcement learning is founded on actions between the agents and environment with the agents learning through the interactions through rewards or punishments. Such a learning process is particularly popular when sequential decision-making is essential e.g. in robotics, game playing, and autonomous driving. Such methods as Q-learning and deep reinforcement learning allow systems to learn and refine their behavior dynamically in order to maximize long-term reward. Rathod (2024) states that this paradigm describes an active and target-based learning plan, in which agents engage in an ongoing practice and maximize policies by means of learning.

2.2.1.1 Supervised Machine Learning

Supervised learning is a basic method in machine learning that involves training a model on a labeled dataset, that is, each training example contains input data, along with the desired output. The aim is that the model should be able to generalise on this data and produce correct outputs on new, unknown inputs. Nasteski (2017) claims that supervised learning is especially effective in implementing the applications that require classification or regression, including

email filtering, medical diagnosis, and predicting customer behavior. The training process entails optimization of a loss function that quantifies the difference between the model predictions and actual values. This technique enjoys well-organized and structured data and exhibits excellent interpretability and learning process control, which is why it is currently one of the most widely used paradigms in the data-driven decision-making system.

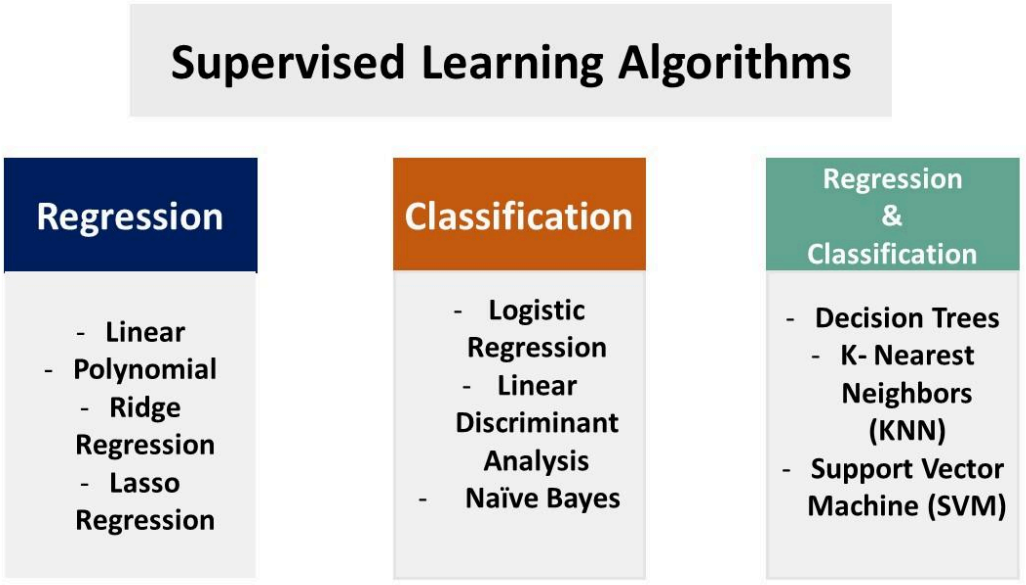


Fig 2.2 groups supervised learning algorithms into three categories: regression, classification, and regression & classification and lists typical techniques in each category, including linear regression, logistic regression, decision trees, and K-Nearest Neighbors (KNN).

2.2.1.1.1 Linear Regression

Linear regression is a statistical technique of supervised learning which fits a straight line to the relationship between a dependent variable and one or more independent variables. It is primarily applicable in forecasting continuous values. The model operates by fitting the most appropriate linear equation that reduces the variance between the predicted and actual values through a technique referred to as least squares estimation (Nasteski, 2017). Linear regression is a powerful tool although it is a simple tool when there are linear and interpretable relationships among variables. It is regularly applied in the spheres of economics like market trend

forecasting, education such as student performance prediction, as well as in the sphere of gaming analytics (e.g., player engagement estimation in terms of the time spent and actions taken in the game).

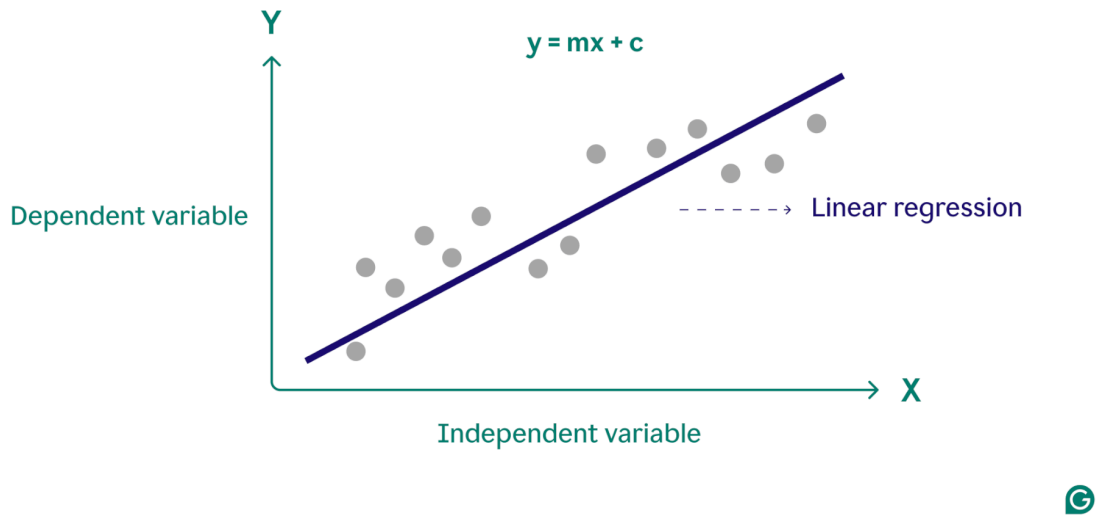


Fig 2.3 The concept of linear regression is represented in the image, where a best-fit line, y = mx + c, is illustrated in the relationship between an independent variable (X) and a dependent variable (Y).

2.2.1.1.2 Logistic Regression

Logistic regression is a generalization of linear regression to classification, in particular to binary classification. It does not make a continuous number prediction, but rather estimates the likelihood that a particular input falls into a particular class. Nasteski (2017) explains that logistic regression accomplishes this by squashing the output to a range of 0 to 1 by using the sigmoid (logistic) function to the linear combination of input features. This level of probability can then be applied to assign classes. It is especially applicable in areas like marketing (e.g. determining whether a customer will convert or not), healthcare (e.g. determining whether a person has a disease or not), and behavior prediction of players in games (e.g. whether a player will use a defensive or aggressive play style). Logistic regression is popular because it is simple, easy to interpret, and it is computationally inexpensive.

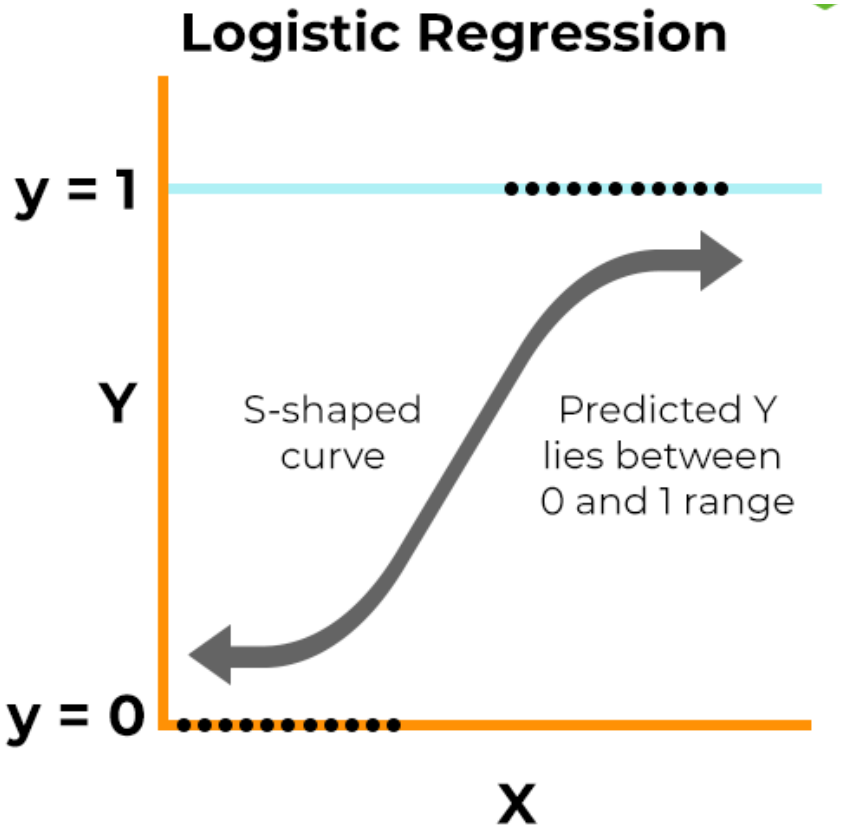


Fig 2.4 The picture represents the logistic regression, as an S-shaped curve is utilized to estimate the results between 0 and 1 based on the values of input (X) that are often applied to the binary classification task.

2.2.1.1.3 Decision Trees

Another much-used supervised learning algorithm is decision trees, which are appreciated due to their simplicity, interpretability and usefulness in classification and regression. They operate by a recursive divide and conquer strategy on the data by feature values creating a tree structure with decision rules at the nodes and final predictions at the leaves. Rathod (2024) emphasizes their advantage of modeling human decision-making processes, employing an algorithm such as ID3, C4.5, CART to pick the most informative features based on the measures, such as information gain or Gini impurity. Ghosh (2023) further explains that since decision trees work with both numerical and categorical data and do not require feature scaling, they are perfectly suited to transparent tasks such as medical diagnosis or credit scoring. As an example, a decision tree in games may suggest a champion based on the input such as the

preferred role, the level of aggression, and the lane to play. Nevertheless, because of overfitting, particularly in deep trees, the ensemble approaches such as Random Forests are typically employed to enhance generalization.

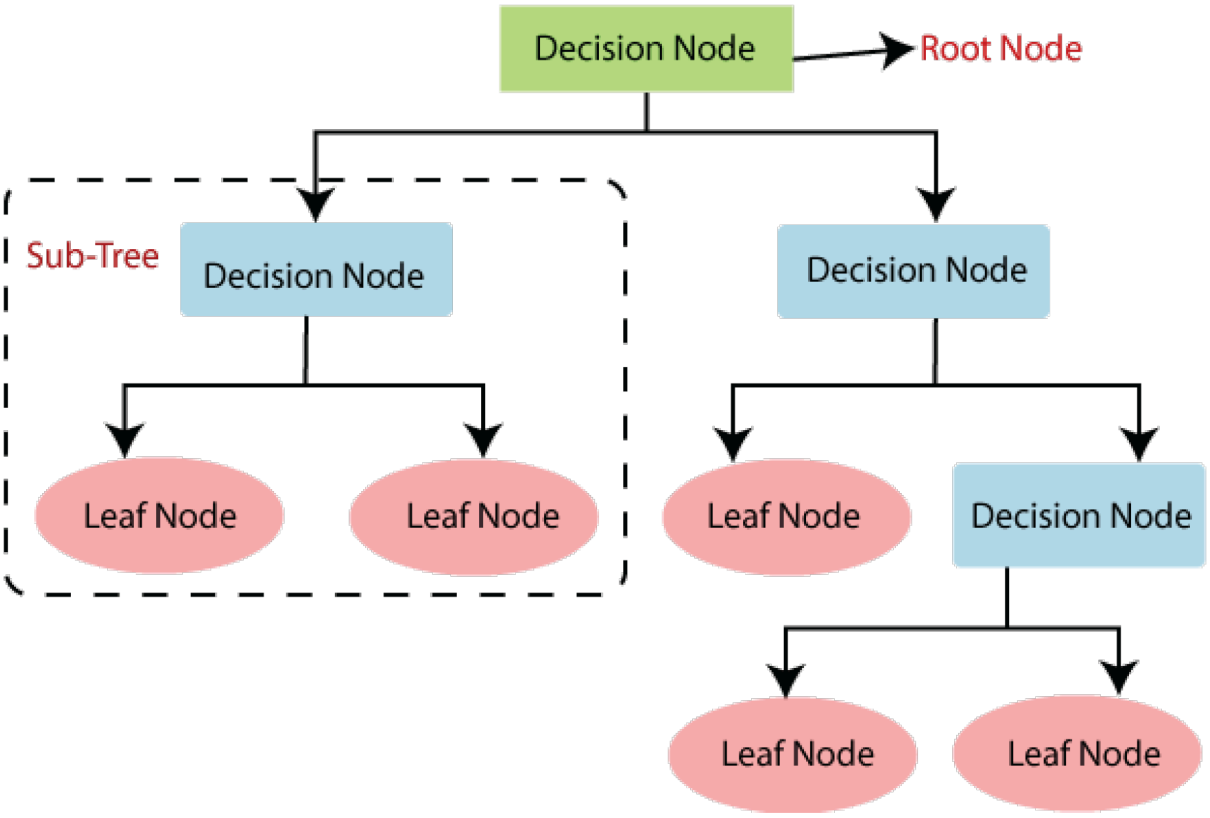


Fig 2.5 The given picture is a decision tree diagram which is a hierarchical model applied in machine learning to decide data by sorting it into various categories using a sequence of choices.

2.2.1.1.4 Support Vector Machines (SVM)

Support Vector Machines (SVM) are potent supervised learning algorithms that are mainly applied in classification problems, and more so in high-dimensional data. The main concept of SVM is to locate the optimum hyperplane that can best divide data points of two classes and at the same time maximize the margin between them. Battineni et al. (2020) state that SVMs especially shine in medical and diagnostic applications because of their robustness and the capability to capture both linear and non-linear relationships with the help of kernel functions. Such kernel functions enable the algorithm to convert data to higher dimensions so that it is

easier to separate classes. As an example, in a game application, SVM may be employed to categorize players as either aggressive or passive players according to their in-game performance. Battineni et al. point out that SVMs may be computationally expensive to use with large datasets as well as needing proper parameter tuning though they are very accurate. Nevertheless, they are at times a popular option in predictive modeling due to their reliability and precision.

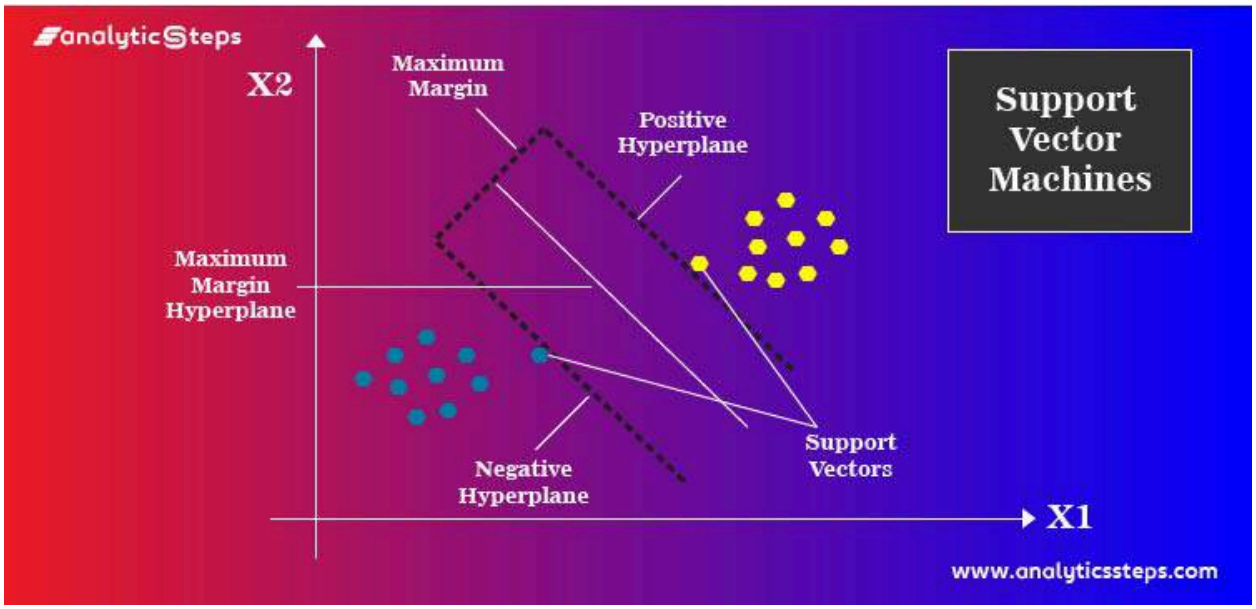


Fig 2.6 The given image is the example of how Support Vector Machine (SVM) algorithm works separating data points into different classes identifying the best hyperplane with the maximum distance between the nearest data points called support vectors.

2.2.1.1.5 K-Nearest Neighbors (K-NN)

K-Nearest Neighbors (KNN) is an easy-to-use powerful supervised learning algorithm with many applications in classification and regression problems. According to Goyal et al. (2022), KNN works on the logic that similar data points have high chances of being of the same type. Upon introduction of a new input, the algorithm finds the ‘k’ nearest objects (neighbors) of the training set according to a distance measure, most often Euclidean distance, and classifies the new input as the majority of the neighbors. Goyal et al., stressing that one of the advantages of KNN is that it is non-parametric, i.e., it does not assume anything about the distribution of underlying data, and, therefore, it can be applied in many different applications. They however also note that KNN is computationally costly when the datasets are large because it has to

compute the distance between the training instances and the training instance during the prediction. In game applications, KNN can be applied in recommending champions to new players through identifying others who have the same behavioral profile and the same playstyle-recommending a champion based on the preferences of the similar players.

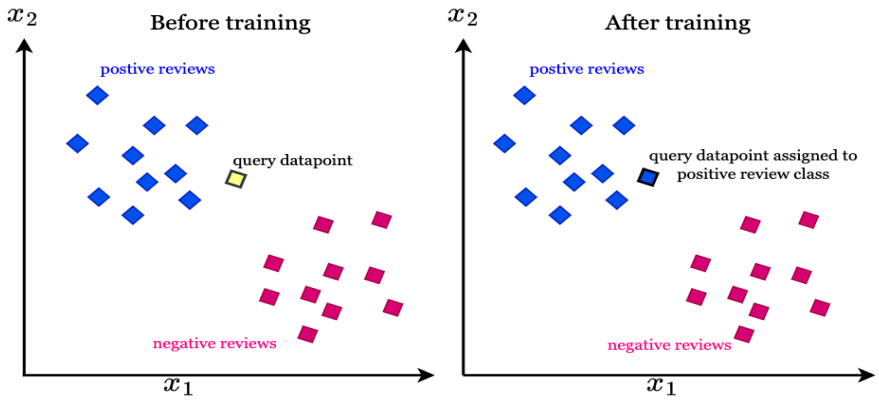


Fig 2.7 The offered picture shows an example of the k-nearest neighbors (KNN) algorithm, according to which the new point of data is assigned to the class that is most frequently observed among the nearest neighbors.

2.2.1.2 unsupervised Learning

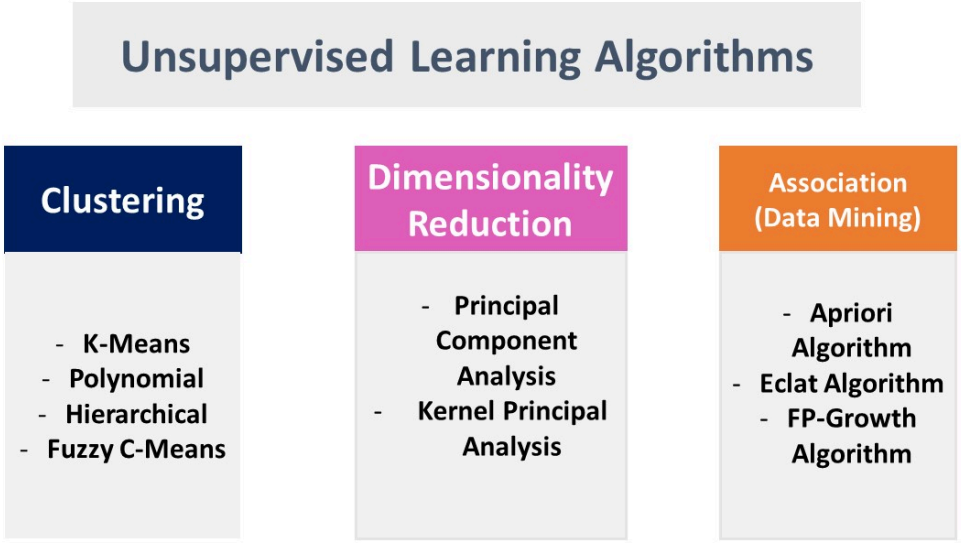


Fig 2.8 The given picture gives a summary of the unsupervised learning algorithms, which are divided into three major categories clustering, dimensionality reduction and association.

Unsupervised learning is a type of machine learning, which involves working with unlabeled data, that is, the system does not have predetermined results or target variables. Rather, it seeks to reveal latent patterns, structures or groupings in the data independently. Rathod (2024) argues that unsupervised learning can be especially helpful when manual labeling is not feasible or cost-effective, thus it is a good fit in applications such as customer segmentation, anomaly detection and data compression.

2.2.1.2.1 K-Means Clustering

The K-Means Clustering is one of the most popular unsupervised machine learning algorithms to divide the datasets into clusters according to similarity of features. K-Means operates by categorizing data into a predetermined variety of clusters (K), where each datum point is allocated to the cluster with the closest center, referred to as the centroid (Zhang, 2022). The algorithm then recalculates the centroids and reassigned points until the groupings are stable iteratively. This approach is effective and not very complex in nature, so it can be applicable to large-scale datasets.

According to Zhang (2022), the strength of K-Means is in the possibility to unveil hidden structures in data, which is why it can be employed in such tasks as customer segmentation, behavior profiling, and image compression. K-Means could also be used in the gaming environment to cluster players according to their behavior in the game or their answers to the survey questions, and then give them more personalized champion suggestions. The method however needs the number of clusters to be specified beforehand and might not work well when clusters of irregular shapes or overlaps exist.

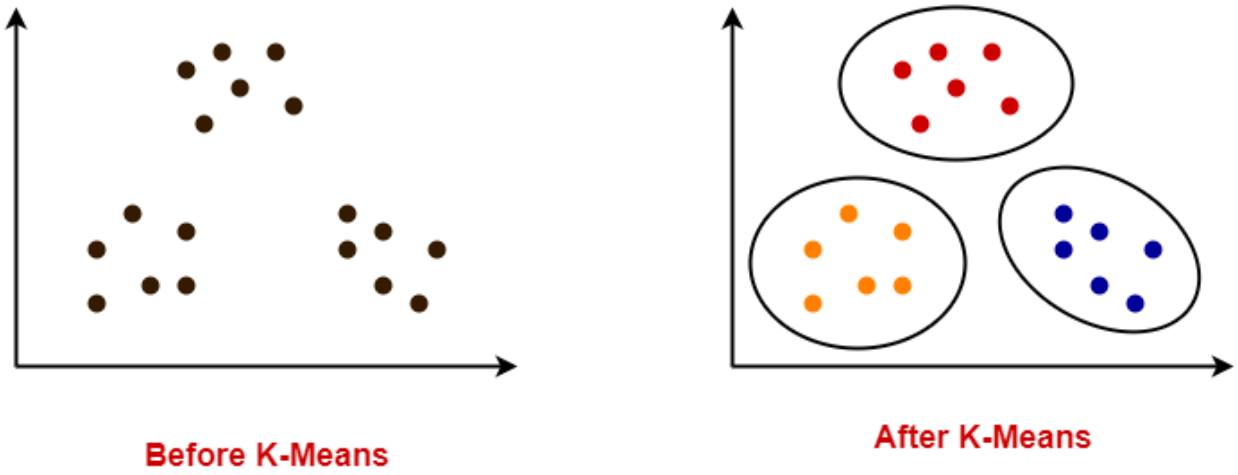


Fig 2.9 The given image demonstrates the K-Means clustering algorithm that divides the unclassified data points into three different clusters once the algorithm is implemented.

2.2.1.2.2 Hierarchical Clustering

Unsupervised machine learning Hierarchical clustering is a common approach to cluster analysis that constructs a hierarchy of clusters either bottom-up (agglomerative) or top-down (divisive). The agglomerative method starts with every data point as a cluster and sequentially combines the nearest pairs of clusters using a measure of similarity, until the remaining clusters are merged into one big cluster (Shetty et al., 2021). This hierarchy is usually portrayed via a dendrogram-a tree-like graph that displays the way in which clusters are united at each level, and enables the user to decide on an appropriate number of clusters by cutting the tree at a desired height.

According to Shetty et al. (2021), one of the advantages of hierarchical clustering is that it does not presuppose the number of clusters to be formed, which is why this method is best to use when exploring data. It finds extensive use in practice in such fields as text classification, gene expression analysis, and customer profiling.

In the scenario of a game, especially in a game like League of Legends, hierarchical clustering could be used to cluster players according to their in-game behavior; like levels of aggression, their roles, or the speed at which they take decisions. As an illustration, it might unveil the existence of obscure groupings like the players who are more interested in high-risk plays and players who are more inclined to support plays. Such clusters can further be applied in

suggesting champions that are in-line with the predominant playstyle of each group, making it more personalized to new players and, therefore, increasing player satisfaction.

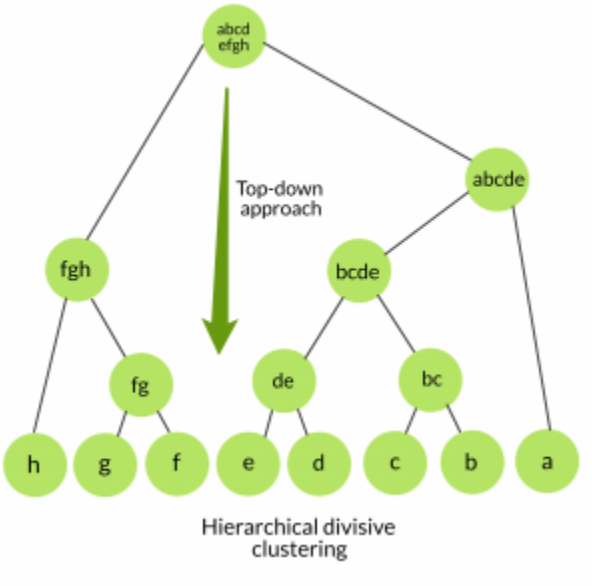


Fig 2.10 The image given represents the K-Means clustering algorithm where unclustered data points are clustered into three different clusters upon application of the algorithm.

2.2.1.2.3 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is an unsupervised machine learning method of dimensionality reduction and feature extraction. It reduces a large number of correlated variables into a smaller number of uncorrelated variables called principal components which contain the most important variance in the data. Zamri et al. (2021) state that PCA is especially useful in clarifying large and complicated data, removing unnecessary data, and enhancing the performance of machine learning models particularly high-dimensional data.

Zamri et al. (2021) highlight that PCA is broadly used in the medical diagnostics and classification tasks where it is necessary to determine the most influential features. Within the framework of the given study, PCA can be used to process data concerning the reaction time, the level of aggression, the preferences in the roles, and the length of the match. By summarizing this information into important elements, the model will only have to look at the most influential player characteristics. The extracted components assist in the enhancement of accuracy and interpretability of champion recommendation systems because predictions are made in terms of the most significant elements of player behavior.

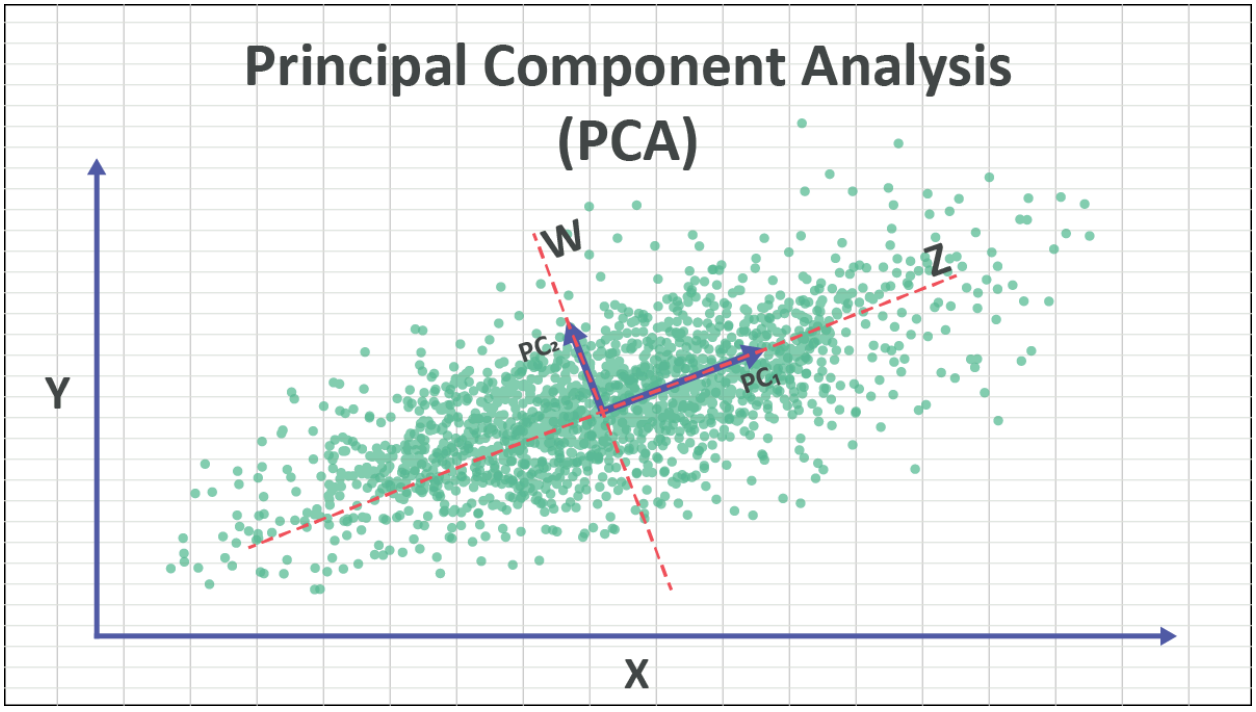


Fig 2.11 The given image explains the Principal Component Analysis (PCA) dimensionality reduction method that converts a dataset into a new coordinate system in which the maximum data variance is represented on the first axis (PC1), and the next most variance is represented in the second axis (PC2).

2.2.1.2.4 DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

DBSCAN is a density-based clustering algorithm which discovers clusters of densely populated points and marks points in low density areas as noise. It does not need to predefine the number of clusters, unlike other partitioning algorithms like K-Means and hence can be applied to datasets whose cluster structure is unknown or irregular. Li (2021) explains that DBSCAN operates through specification of two parameters, Eps (the radius of a neighborhood) and MinPts (the minimum number of points to constitute a dense region). The set of points that satisfy these density conditions constitute clusters and those that fail to satisfy it are outliers. Li underlines that this flexibility enables DBSCAN to work with complex spatial data and even work well in noisy settings.

Within the frames of the present study, DBSCAN may be used to group the League of Legends players on the basis of their behavioral and psychological characteristics, as identified

through the questionnaires, including the level of aggression, the preferred lane or the tendency towards teamwork. The system will be able to recommend more personalized champion suggestions by comparing the dense clusters of similar players and eliminate the outlier behavior that might bias the recommendations.

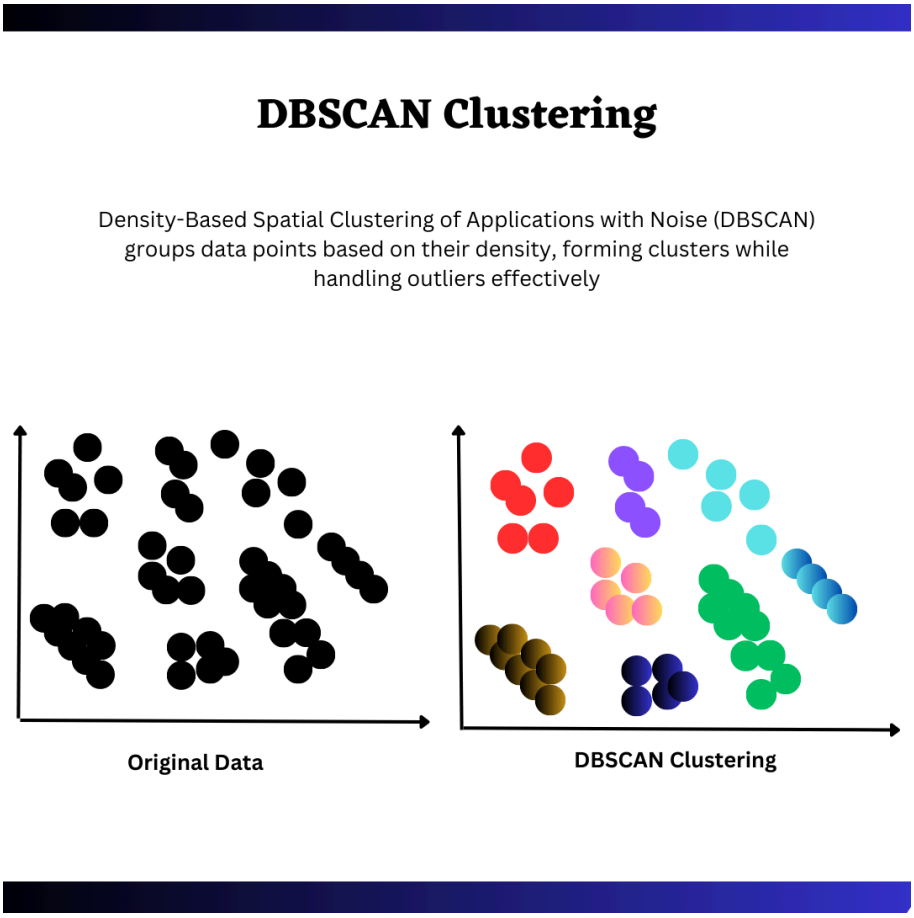


Fig 2.12 The picture shows the DBSCAN (Density-Based Spatial Clustering of Applications with Noise) algorithm in which the data points are classified as core points, border points, and noise and clustering is based on density instead of distance.

2.2.1.2.5 Autoencoders

According to Bank et al. (2021), autoencoders represent a type of neural networks that are specifically tailored to unsupervised representation learning. They do this by encoding input data into a latent space that has fewer dimensions than the data (compressing the data) and decoding the data back to its original form. This compression is useful in extracting meaningful patterns and also in removing noise in the data and it is therefore flexible in use in areas like

dimensionality reduction, clustering and anomaly detection. Autoencoders are more flexible in representing features compared to traditional methods such as PCA, since they can capture non-linear relationships which are complex.

One of the strong points of autoencoders is that they are flexible when it comes to recommendation systems. Bank et al. emphasize such models as AutoRec, which can learn latent user-preference representations and use them to make effective predictions of missing ratings or preferences. When considering the problem of a League of Legends champion recommender system, it is possible to use autoencoders to encode player behavior and preferences, e.g. tendencies in playstyle or champion qualities they like, into a latent space. This representation may then be decoded to propose champions that are closest to their implicit preferences even to new players where little gameplay information is available. This feature closes the gap between the lack of player information and the appropriate, individual suggestions, providing an effective method of improving the onboarding process in MOBA games.

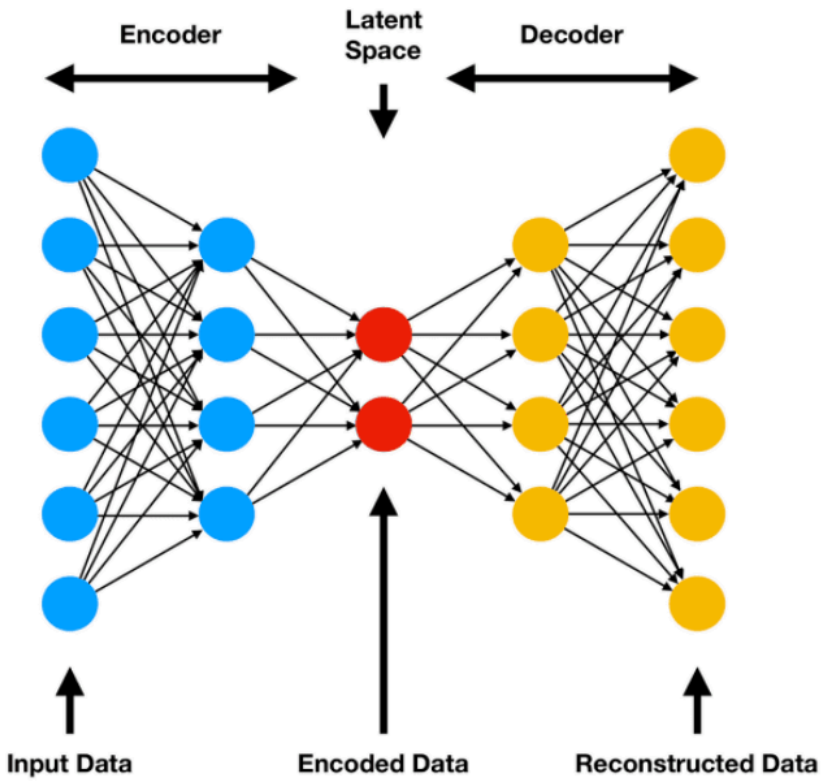


Fig 2.13 The picture explains architecture of autoencoder, a type of neural network that compresses input data into low-dimensional latent space via an encoder and then reconstructs the data in the space via a decoder.

2.3 Machine Learning techniques for prediction

Machine learning (ML) has become a central tool for predictive modeling across multiple industries due to its ability to process large datasets and uncover hidden patterns that traditional statistical methods may miss. Ghosh (2023) highlights that ML prediction models are capable of learning from historical and real-time data, enabling accurate forecasts in dynamic environments such as healthcare, finance, and gaming. Similarly, Rathod (2024) explains that predictive ML supports decision-making by modeling complex relationships between input variables and outcomes, improving efficiency and reducing uncertainty in high-stakes domains.

Predictive models are implemented using diverse algorithmic approaches, including regression-based techniques, classification models, ensemble methods, and deep learning architectures. Zamri et al. (2021) emphasize that these approaches are selected based on the nature of the prediction task, whether it involves continuous variables like sales forecasting, categorical predictions such as fraud detection, or sequential outcomes like player performance in games. Bank et al. (2021) further note that advanced neural network architectures, such as autoencoders, have enhanced prediction by capturing non-linear relationships and reducing noise in high-dimensional data.

These predictive capabilities extend across key sectors. In healthcare, models forecast disease progression and support early diagnoses; in finance, they identify fraud and predict credit risks; in marketing, they anticipate customer churn and personalize recommendations; in education, they forecast student performance; and in gaming, they predict player behaviors and preferences to tailor in-game recommendations. This broad applicability demonstrates ML's growing role as a foundation for predictive analytics in both commercial and research contexts.

2.3.1 Healthcare Prediction

The machine learning prediction models have been widely used in the healthcare domain to enhance the precision of diagnosis, streamline treatment plans, and predict outcomes of patients. Hossain et al. (2023) provide a report on the use of the decision trees, support vector machines, and deep learning models as important ML algorithms that have been successfully employed in the early detection of diseases and prognosis prediction. Their work also emphasizes the benefits of using predictive modeling to mine large volumes of clinical data (patient history, medical imaging, and laboratory results) to identify minute patterns not clear to clinicians, and make more accurate and quicker decisions. In the same manner, Saha et al. (2024) underline that predictive healthcare models are extremely efficient in the context of medical interventions and allow offering actionable insights and can be effectively applied in the fields of preventive medicine and chronic disease management. They indicate that by incorporating such models into clinical work processes, it is possible to minimize human error and provide interventions to at-risk patients in a timely manner. Alzahrani (2024) explains the increasing importance of AI-based prediction systems to help resolve healthcare issues in resource-constrained environments in another study, stating that the models have the potential to close diagnostic gaps by automating risk assessment and providing individualized treatment recommendations. Together, all these studies support the revolutionary power of machine learning in the healthcare setting, particularly in transforming medical practice into proactive and data-driven care.

2.3.2 Financial Forecasting and Risk Prediction

Financial forecasting has found a new stone in machine learning that allows the prediction of market trends, stock prices and risk assessment to be more accurate. Nguyen et al. (2023) note that random forests, gradient boosting, and recurrent neural networks are some of the most common algorithms that have been used to analyse historical financial data in order to identify complex, non-linear relationships that are likely to go undetected when using traditional statistical methods. Their survey shows how these predictive models solve a variety of financial problems such as credit scoring, fraud detection, and portfolio optimization, and thus enhancing the quickness and precision of financial decision making. In further detail, Ali and Khan (2024) underline the importance of predictive analytics in improving the process of financial management by incorporating risk calculation into real-time decision-making models. According

to their study, the tools enabled by machine learning enable the institutions to be proactive in detecting the potential risks of loss in financial area and dynamically adjust the strategies according to the market changes. Equally, Rahman (2022) examines the role of predictive modeling in sustainable financial planning, highlighting that it can forecast the cash flows and reduce the investment risks, especially in small and medium-sized enterprises (SMEs). Collectively, these works demonstrate how machine learning can enable resilient financial forecasting given the availability of scalable and adaptive solutions to the dynamics of contemporary financial systems that require resilience and precision.

2.3.3 Marketing and Customer Behavior Prediction

The major importance of machine learning in contemporary marketing consists in the possibility to make correct predictions regarding customer behavior and support the strategies of hyper-personalized engagement. Zhou et al. (2023) emphasize the role of predictive models in using extensive consumer data to inform the future preferences and subsequent purchasing trends using various data such as browsing history, past purchases, and demographic data. Their research points to the feasibility of such algorithms as support vector machine, decision tree, and deep learning in customer segmentation, churn rate prediction and optimization of marketing campaigns. Expanding on these ideas, Li (2024) dives deeper into how social network data analysis can be used to improve customer behavior prediction, stating that how customers interact on social networks such as Facebook and Twitter can be used to gain subtle insight into how customers feel and what new markets might emerge. Such a practice will enhance not only customer segmentation but also targeted advertising strategies that are going to change dynamically and respond to changes in consumer interests. In equal measure, Wu and Chen (2024) discuss how deep learning frameworks can be applied in marketing decision models and show how the neural networks will be in a position to combine the heterogeneous sources of information such as user comments and transaction logs in order to forecast the product recommendations and campaign adoptions accurately. All these studies explain the role of machine learning in enabling marketers to develop dynamic and data-driven approaches that can improve customer relationships, lower churn and boost the overall conversion rates.

2.3.4 Education and Academic Performance Prediction

Within education, machine learning is also being explored in predicting student performance in order to facilitate proactive data-driven interventions. As shown by Ramesh et al. (2024), even a broad set of academic indicators, including grades, attendance patterns, behavioral records, engagement data, among others, can be used in predictive algorithms to determine at-risk students and predict exam scores or completion rates. Their work notes the efficiency of models such as logistic regression, decision trees, and neural networks in helping to build individual learning plans and enhance the overall student retention. In parallel to this strategy, Sharma and Patel (2023) propose a multi-model ensemble system of forecasting academic performance in the context of higher education by applying a set of algorithms to obtain an even higher accuracy and reliability. The results of their study demonstrate that ensemble methods are more helpful when facing various educational data, and they enhance predictions of students with different learning patterns. In the same vein, Dutta and Bandyopadhyay (2022) review the machine learning models employed to predict academic performance, with particular focus on hybrid learning and deep learning methods to capture the behavior and performance patterns in a complex manner. Collectively, the research papers depict how predictive modeling in teaching, not only makes it easier to intervene in time but also enables teaching practitioners to individualize learning, maximize resources and accelerate the success of students.

2.3.5 Gaming and Player Behavior Prediction

The use of machine learning in the gaming industry has grown and has been used in forecasting the behavior of players and in providing customized user experience during the game. This has been evidenced by Shen et al. (2022) in their sequential recommendation model of League of Legends, where they used Bi-LSTM and knowledge graphs in order to take into account temporal dependencies when it comes to using champions during the ban-pick phase. This made strategy recommendations to be more accurate and competitive playing to be better. In the same vein, Hanke and Chaimowicz (2017) investigated recommender systems of hero

line-ups in MOBA games, considering the team composition and synergy to maximize the probability of a victory. They can analyze enormous volumes of gameplay data in order to adjust team tactics to the inclinations of the players, and this is done in their work. Yu et al. (2025) followed another direction and applied supervised learning to the prediction of the Big Five personality traits of players based on behavioral data in the game using Random Forest and SVM. Their work focuses on the possibility of psychological knowledge to improve personalization to the point where the provided recommendations focus on not only the efficiency of the gameplay but also on the satisfaction of the player and their further interest. In combination, these articles show how predictive modeling can not only optimize competition, but can also help in onboarding new or casual players in an adaptive fashion, matching recommendations to both playing style and personality.

2.4 Comparable Machine Learning Techniques in Game-Based Recommendations. Change the title later. Add the last RF point in paragraph form

Three supervised learning algorithms have been chosen to compare in the scope of this study, namely Random Forest (RF), Decision Tree (DT), and K-Nearest Neighbors (KNN) because they are not only effective in performing predictions of the gaming-related context but also applicable to the data gathered in questionnaire form. Both of the techniques have their specific advantages in modeling the correlation between the behavioural and psychological characteristics of players and the characteristics of champions in League of Legends.

On MOBA-related studies, Random Forest has always performed well especially in predicting player skills and personality attributes. Pengmatchaya and Natwichai (2024) used RF in one of the machine learning pipelines in the Dota 2 game, where it was used with clustering to group players into skill tiers. Similarly, Yu et al. (2025) used RF as one of the algorithms to predict Big Five personality traits using MOBA gameplay data, which further supports the idea that it is able to project gameplay behavioral patterns onto psychological knowledge. The ensemble nature of RF (combination of several decision trees) allows it to process both categorical and quantitative data on questionnaires, which is very resistant to noise and overfitting of data (typical of new players). In addition, its feature importance scores facilitated interpretation where researchers can find out which features have the strongest impacts on

recommendations. These attributes fit well with the purpose of the current study, which is to provide individualized recommendations of onboarding instead of optimizing win rates.

Another technique that is relevant is Decision Tree, which is appreciated because of its interpretability and simplicity. It organizes data through a systematic process of branching out, and it is very easy to visualize its decision-making process since it is transparent. This is especially useful when designing a system to cater to beginners, due to the ability to give clear reasons as to why certain champions are recommended. DT will be able to handle categorical and numerical answers in the questionnaire with a little preprocessing and thus a robust comparison to more cutting edge methods such as RF. Being more prone to overfitting, DT models have a simpler structure that provides useful information about the hierarchical relationships between player traits and champion attributes, which helps to comprehend the recommendation logic.

K-Nearest Neighbors supplements these methods with a different one, instance-based learning, answering questions according to the taste of other players with a similar questionnaire profile. The approach is also intuitive to the concept of recommending heroes that would be appropriate in similar play styles or behavioral patterns. KNN does not need complicated model training, and it evolves with the dataset making it useful with changing datasets. It is however dependent on the proper distance measures and feature scaling which are taken care of in the preprocessing of the data so as to make sure that the similarity is calculated correctly.

This study makes use of the properties of Random Forest, which are strong predictive performance and robustness, by utilizing these three methods together. At the same time, it also takes advantage of the interpretability of Decision Trees and logic based on similarity of KNN. This multi-model solution makes it possible to compare the performance and suitability of different models in terms of champion recommendation work and ensures that the final solution strikes a balance between accuracy, transparency, and flexibility to use the system with new players.

2.4.1 Recommender prediction in MOBA Games

Recommendation systems in MOBA games have been studied in a number of papers, with most work on champion or hero selection, and less work on item recommendations. These references prove the usefulness of machine learning in the process of helping the players but are quite different in their approach to what is seen in this research where the focus is more on psychological and behavioral personalization.

Do et al. (2020) have created a collaborative filtering model to suggest champions in the League of Legends game by finding similarities in the history of champion choice among players. They revealed that players reacted to suggestions positively in comparison to random recommendations, indicating the possibility of data-driven personalization. Nonetheless, this method was fully based on the historical data of playing games and did not use behavioral or psychological characteristics, which made it less applicable when onboarding new users who have no match history.

Lee et al. (2022) proposed DraftRec, a neural network-based recommendation model that incorporates the individual champion preferences and team synergy to improve the draft result. They offered real-time advice in the champion selection stage, considering both individual and team-based variables in order to assure the best chance of winning. Although innovative, DraftRec is more of a game of competitive synergy rather than the game of learning and fun, which makes it less desirable among casual or first-time players.

In a similar fashion, Chen et al. (2019) used association rules and classification to create a context-sensitive recommendation system, mainly focusing on in-game items, but the extensions were made to champion suggestions as well. Their model was used to examine the trends of item and champion usage to make predictions on next-best choices in matches. Even though this system worked in its context, it focused on the gameplay mechanics and situational item builds more than it focused on the individual player traits or preferences.

Ensemble methods such as Random Forest have also been used by community-driven projects such as ChampionEdge to suggest champions that maximize team win rates (ChampionEdge, n.d.). Though based on powerful algorithms, these tools are intended at experienced players that aim at receiving competitive advantages and do not reflect upon the specifics of a newcomer or personality-based suggestion.

The given study offers an innovative solution that combines both behavioral and psychological data collected through questionnaires with a Random Forest model to produce champion recommendations that would be personalized to the new players. Unlike the previous studies that either optimize win-rates or rely greatly on the historical data of gameplay, this study is aimed at matching recommendations to personal playstyles and better onboarding. In such a manner, it fills a major gap in the MOBA recommendation systems, changing the focus on the purely competitive results to the increased player satisfaction and long-term engagement.

2.4.2 summary table listing 10 studies that are related to mine (prediction on the gaming industry can be other games)

Reference	Title of the Paper	Technique Used	Findings (What They Solved)	Limitations	DOI / Link
Do et al. (2020)	Using collaborative filtering to recommend champions in League of Legends	Collaborative Filtering (SVD-based)	Recommended champions by analyzing similarity in players' past usage; improved personalization over random picks	Relies on historical gameplay; not suitable for new players without match history	arXiv

ChampionEdge (n.d.)	Champion recommendation with Random Forest & ensemble methods	**Random Forest** + Ensemble Methods	Community-built system recommending champions for team win rate optimization	Focuses on win probability, not personal playstyle; lacks academic validation	GitHub
Chen et al. (2019)	Data mining for item recommendation in MOBA games	Association Rules + Classification + Ranking	Context-aware recommendations for items and occasional champions using usage patterns	Primarily item-focused; limited champion recommendation and no personality-based personalization	ResearchGate
Shen et al. (2022)	A Deep Learning Supported Sequential Recommendation Mechanism for Ban-Pick in MOBA Games	Bi-LSTM + Knowledge Graph	Modeled temporal pick sequences to optimize draft compositions	Focused on team win rate, not personal preferences	IEEE Link
Horst et al. (2024)	DraftComPromise: Draft Composition Recommendatio	Synergy-based scoring + Win rate analysis	Suggested balanced team compositions in real-time drafts	Ignores individual player psychology	IEEE Link

	ns in League of Legends				
Smit (2019)	A Machine Learning Approach for Recommending Items in League of Legends	Collaborative Filtering	Personalized item recommendations based on historical data	Item-focused, not champion recommendation	PDF Link
Dallmann et al. (2022)	Sequential Item Recommendation in the MOBA Game Dota 2	RNN-based sequential recommendation	Improved sequential item recommendation performance	Focused on items, not heroes/champions	arXiv Link
Hanke et al. (2017)	A Recommender System for Hero Line-Ups in MOBA Games	Association Rules + Neural Network	Suggested hero lineups and predicted match winners (~74.9% accuracy)	Focused on team composition, not beginner personalization	https://cdn.aaai.org/ojs/12938/12938-52-16455-1-2-20201228.pdf
Pengmatchaya & Natwichai (2024)	Identifying player skill of Dota 2 using machine learning pipeline	Clustering, Random Forest, Logistic Regression	Predicted player skill levels in *Dota 2* using behavioral data and a machine learning pipeline, improving	Focused on skill prediction rather than personalized hero/champion recommendations	*https://doi.org/10.1007/s44163-024-00139-y*

			matchmaking accuracy		
Yu et al. (2024)	I Am What I Play in Moba Game: Supervised Machine Learning Approaches to Predict Gamers' Personalities on the Basis of Their In-Game Behaviors	**Random Forest,** SVM, Decision Tree, K-NN	Predicted players' Big Five personality traits from MOBA gameplay data (F1 scores between 0.61–0.68); Random Forest performed best	Focuses on personality inference, not champion recommendation; relies on in-game data, not surveys	*SSRN DOI*
Demediuk et al. (2019)	Role Identification for Accurate Analysis in Dota 2	Ensemble Clustering	Classified player roles automatically using behavioral features, improving analytical accuracy for MOBA data	Focuses on role identification, not champion recommendations	*PDF Link*
Akhmedov & Phan (2021)	Machine Learning Models for DOTA 2	Linear Regression, Neural Network, LSTM	Achieved match outcome prediction with up to 93% accuracy;	Focuses on match results—not on personalized player profiling or champion	*arXiv PDF*

	Outcome Prediction		LSTM models performed best	recommendation	
Flint (2021)	Creating a Hero Recommender System for New Players in Dota 2	Clustering + Classification	Recommended heroes for new players based on playstyle patterns	Requires gameplay data, not questionnaire input	Thesis PDF

2.5 Evaluation metrics

Precision@K

Precision@K is a measure that is commonly used to assess ranking-based recommendation systems by considering the first K items that are shown to a user. Rather than assessing the full list of recommendations, it gauges the percentage of relevant items in the first K recommendations, which is important since users essentially interact with a small number of suggestions without going through the whole list (Shaped, 2022).

Khairi et al. (2022) consider Precision@K especially useful when a system returns an ordered list of options, and the more relevant results are placed at the top of the list, the better the overall user experience will be. Translating this into the context of the present study, Precision@K will evaluate the degree to which the Random Forest model can prioritize the champions of League of Legends based on the behavioral and psychological profile of each player acquired as a result of the questionnaire. In case champions that fit well the style of play and preferences of a player always appear in the list of the top K recommendations, this proves the practical utility of the recommender system. This is particularly significant in introducing new players who will tend to select champions among the initial few recommendations as opposed to browsing through a list of many.

Recall@K (Top-K Recall)

Recall@K is another important metric when assessing ranking-based recommendation systems, which looks at the number of relevant items that are captured in the top K recommendations. In contrast to Precision@K, which assesses the accuracy of the top recommendations, Recall@K assesses coverage, that is, whether the system retrieved as many relevant options as possible (Weaviate, 2023).

As Evidently AI (2023) reports, Recall@K plays a significant role when the task is not to give the most accurate recommendations but to give them comprehensively. In applications where users would be adversely impacted by missing a relevant item, high Recall@K guarantees that user favorite items will appear in the first suggestions, even though there are irrelevant items as well.

Recall@K will be used in the context of this research to determine the effectiveness of the Random Forest-based recommender system in proposing champions that fit the behavioral and psychological characteristics of a player to the top recommended champions. This can be especially applicable to new League of Legends players since offering several appropriate choices allows players to have room to explore champions without necessarily conforming to the way they usually play. A combination of Recall@K and Precision@K provides a balanced perspective on the quality of recommendations, thus both the relevance and the breadth of the recommendations are provided to the player during onboarding.

Mean Reciprocal Rank (MRR)

MRR is a ranking criterion that assesses the rank of the first applicable suggestion in the suggestion list. The larger the scores, the closer the relevant items are to the top (Zamani et al., 2021). The metric is especially appropriate when the user is likely to select among the first few suggestions e.g. when there are 1-3 League of Legends champions to be suggested as is common in this research. In this regard, a high MRR implies that the system will always put the most relevant champion, i.e., the one that best matches the behavioral and psychological characteristics of a player, on the first place, improving the efficiency of the onboarding process and building trust among users.

Hit Rate (Top-N Accuracy)

Hit Rate, also known as Top-N Accuracy, is used to determine whether any of the relevant items will appear in the top-N recommendations. It is an easily computable metric that is commonly used in any recommender system to measure practical usability (Garg & Jain, 2015). In this research, Hit Rate will determine whether the system recommends at least one champion that matches the preferences of the player in the first recommendations (e.g., Top-3). This is vital when it comes to novice users, who are not likely to scan through longer lists of recommendations and will resort to the first available options.

F1-Score (for Binary Relevance)

F1-Score is a harmonic average of precision and recall, providing a single measure, which compromises the trade-off between the two measures (Sitarz, 2019). F1-Score will apply in the context of this research whereby the recommendation of champions is categorized as either relevant or non-relevant according to the alignment of questionnaires. This makes the performance of the model to be assessed on a whole basis i.e. it takes into consideration the accuracy of the relevant predictions as well as the capacity to find all the suitable champions. An F1-Score of high value would establish that the recommender system is capable of predicting the relevant champions rightly and finding most relevant options with a minimal number of false positives.

2.6 Summary of literature review

The literature review examined the different machine learning algorithms and how they are used in predictive and recommendation systems with special consideration to MOBA gaming environment. Past research has shown how such algorithms as Random Forest, collaborative filtering, and neural networks can be successfully used in cases like predicting skills, inferring personality, and recommending champions or items (Pengmatchaya & Natwichai, 2024; Do et al., 2020; Lee et al., 2022). These methods emphasize the increasing possibility of machine learning to make the player experience better based on data-driven insights.

Most recommendation systems in the MOBA context have been created to achieve optimal win rates or draft compositions on the basis of gameplay telemetry and team synergy (e.g., DraftRec by Lee et al., 2022). Psychological and behavioral aspects were discussed by Yu et al. (2025) and Šajinović & Bodroža (2024), and they demonstrated that champion or role preferences depend on personality traits. Nevertheless, these studies mostly end at correlation analysis or predictive modeling of traits but not at providing specific recommendations to individual players.

The review has also identified some of the most popular evaluation measures that can be used to evaluate ranking-based recommenders: Precision@K, Recall@K, Mean Reciprocal Rank (MRR), Hit Rate, and F1-Score. These measures will be used to evaluate the proposed model, thus producing not only a correct forecast but also a relevant and practically applicable recommendation to the new players.

Although much has been done in the field of champion recommendation there is a gap in the field of systems that are designed to serve new or casual players which have no long history of matches. Either the current solutions rely a lot on the previous playing patterns or are focused on being competitive rather than enjoyable to the player. This paper fills that gap by introducing a Random Forest-based recommender system that uses questionnaire-based behavioral and psychological characteristics to recommend champions that best match the playstyle and preferences of the players, thus facilitating player onboarding and retention.

Chapter 3

Methodology

3.1 Introduction

This chapter presents the methodology that was used in designing, developing and testing a personalized champion recommendation system in League of Legends. The objective is to outline the organized procedure through which questionnaire-generated behavioral and psychological characteristics can be converted into practical champion recommendations that are based on the preferences and playstyle of new players. In contrast to the current systems that are based on win-rate maximization or team synergy, this work is aimed at maximizing player onboarding and satisfaction by suggesting champions based on individual attributes instead of competitive performance only.

The chapter opens with the introduction of the research framework that demonstrates the entire work flow of the research including the data collection and the deployment of the model. It explains the process of mapping responses of the questionnaire to champion attributes and how the mappings will be used to train and test the machine learning models. Data set is then presented with its characteristics, data sources, and its applicability to the research aims and then

the data preprocessing activities such as selection, cleaning and transformation to guarantee the quality of inputs are discussed.

The methodology also discusses the rationality of the Random Forest algorithm as the main predictive model and its comparison with other methods of predictive modeling, including collaborative filtering and neural networks and their appropriateness in mixed data and the availability of interpretable results. Lastly, the chapter presents the evaluation strategy, where Precision@K, Recall@K, F1-score, and Mean Reciprocal Rank (MRR), as the primary measures of assessing the relevancy and quality of ranking of recommendations are pointed out.

With the systematic description of these components, this chapter provides a clear premise in duplicating and validating the proposed recommendation system, such that it becomes a useful solution to the research gap identified as well as to the positive experience of the game by new players.

3.2 Research Framework

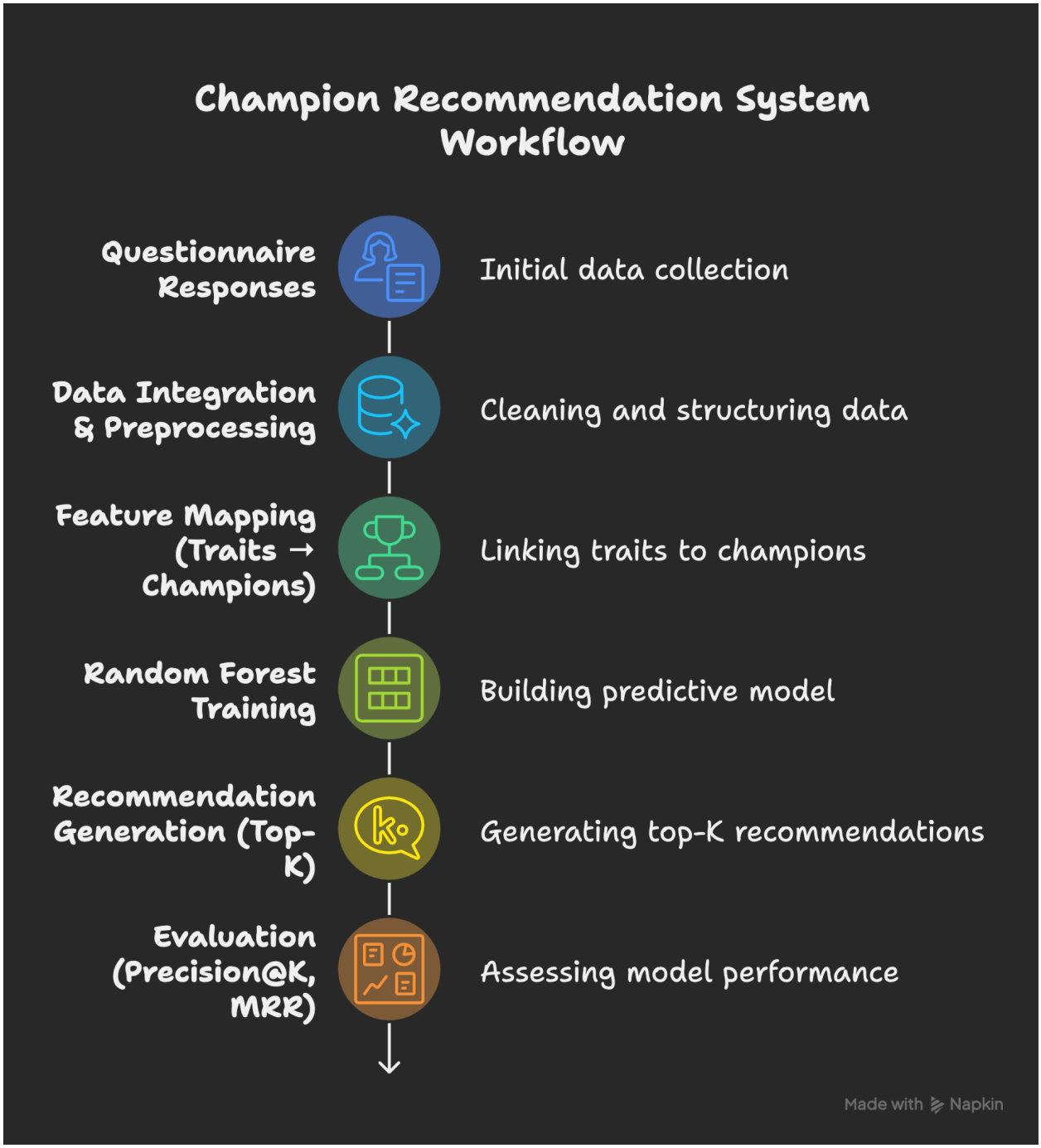


Fig 3.1 The flow chart called Champion Recommendation System Workflow presents a seven-step procedure, which starts with data gathering and its processing, followed by training a predictive model, based on a Random Forest algorithm, and ends with the generation of top-k recommendations, as well as its evaluation.

The research framework of this paper establishes the systematic process through which the champion recommendation system of League of Legends is designed and implemented. It starts with data collection, during which two main datasets are received: the responses to a questionnaire, which are used to register the behavioral and psychological characteristics of new players, and the champion dataset, which comprises such variables as roles, playstyle types, and skill difficulty levels. These datasets are combined to provide a basis of modeling the preferences of players against the characteristics of champions.

The second step is preprocessing of the data, which makes both datasets ready and consistent. This will involve managing any missing values, categorical and numerical variables, and quantifying the responses of the questionnaire. At this level, the mapping is done to match the player characteristics (e.g. aggressiveness, teamwork, preferred roles) with champion characteristics, which then leads to a single set of features to proceed to the machine learning part.

The next stage is the model development stage after preprocessing. The Random Forest algorithm is chosen as the main predictive model because it can work with mixed data types and give the feature importance that can be used to interpret the results. To justify this decision, comparative experiments with other techniques, collaborative filtering and neural networks, will be carried out to compare the relative performance and appropriateness.

Once the model is trained, it is used to generate the recommendations in the form of ranked lists of champions based on the profile of each player. It is all about making sure that the best recommendations are close to the behavioral tendencies and preferences of the player.

Lastly, Precision@K, Recall@K, F1-score, and Mean Reciprocal Rank (MRR) are used to measure the accuracy as well as the ranking quality of the recommendations. This rigid framework will provide a logical flow of data collection to system validation, which will make the methodology consistent with the research purpose of advancing onboarding and personalization of new players.

3.3 Dataset Overview

The data relied upon in this paper has two major parts, which are the answers to the player questionnaires and the League of Legends champion attributes. These two sources are merged to enable the Random Forest model to select the champions depending on the behavioral and psychological profile of new players. The combination of both subjective preferences of thematic players and objective information on champions guarantees that the generated recommendations will be personalized and based on the game mechanics.

The attributes in the questionnaire dataset are the characteristics that have a direct impact on the champion preferences of a player. It also has details about personal playing style, e.g. whether the player is more of an aggressive fighter, defensive fighter, or support member on a team. Other dimensions are lane/role inclination such as, preferring to play at the top lane, jungle, mid lane, or being an attack damage carry or support. The survey is also about choice preferences and psychological profiles such as team-player orientation, risk-seeking comfort level, and whether a player is more systematic or spontaneous when playing. All these responses are gathered on new or new players to make sure that the data obtained reflects subject characteristics rather than competitive level skills. The responses are coded numerically or categorically in order to facilitate a smooth integration with that of the champions in the course of preprocessing and the subsequent model training.

The champion dataset is used as the reference to map, the player attributes with the in-game options and comprises of detailed attribute of all champions present in the game League of Legends. Every champion entry gives details of its main role which defines whether a champion is a tank, a mage, marksman, an assassin or a support. It also outlines playstyle qualities, like whether the champion is strong at burst damage, sustained damage over time, crowd control or utility-based abilities. Moreover, the data provides a classification of the difficulty of skills, which represent whether a champion is easy to learn, medium in complexity,

or complex to play, which is an essential aspect when suggesting champions to inexperienced players who will not be able to cope with such gameplay. The dataset contains essential statistical characteristics, such as base health, mobility, distance of attack and scaling potential, which, in combination determine the performance of a champion at various stages of a game.

With the addition of these specific characteristics of champions, along with the responses in the questionnaires, this study is able to come up with a hybrid dataset that links subjective behavior traits with objective gameplay traits. This will enable the Random Forest, Decision Tree, KNN models combined as ensemble model to suggest a champion that would suit the personality and preferences of a player as well as be suitable according to the skill level and learning curve. In addition, the dataset structure is also flexible, which is why it can be updated in the future along with new champions being added or old ones being rebalanced so that the recommendation system would stay relevant and up to date as time goes by.

3.4 data preprocessing

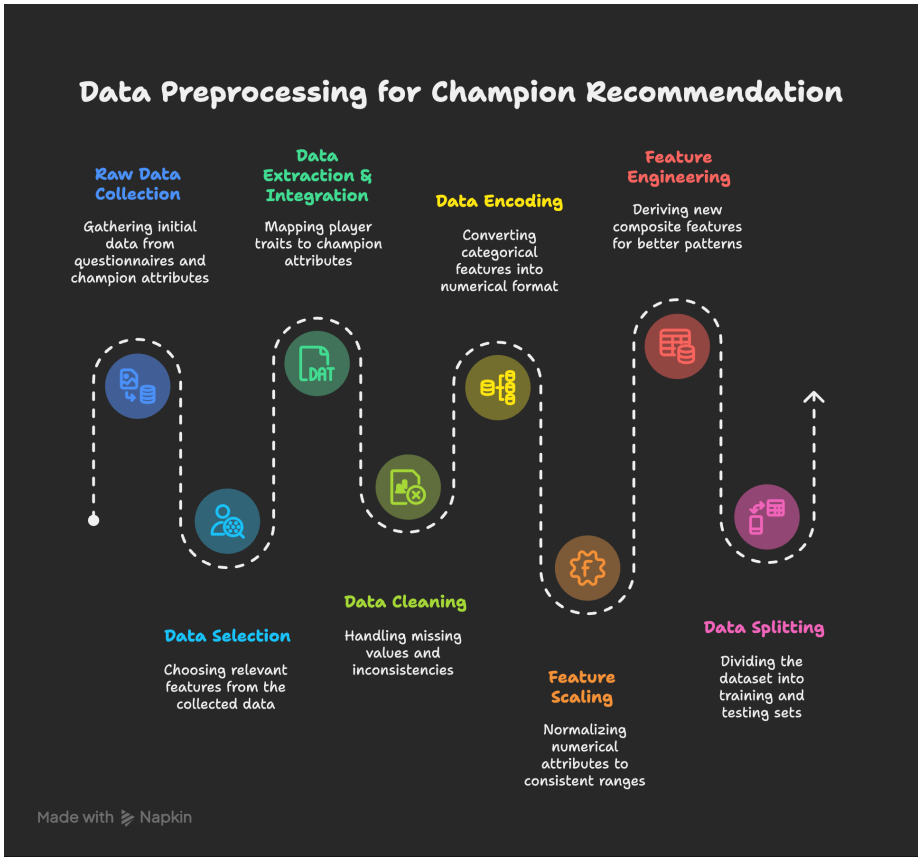


Fig 3.2 Data preprocessing pipeline of a champion recommendation system is described and includes processes of data collection, cleaning, and feature engineering as well as data partitions to train and test models.

Data preprocessing will be an important step in this study, which guarantees that the data on the questionnaire along with the data on the attributes of the champions will be ready to integrate and model. This step will be aimed at taking raw inputs and converting it into a clean, consistent and structured dataset that can be read and understood well by the Random Forest algorithm. This is done through several sub-processes such as data selection, extraction, cleaning, encoding, feature scaling, feature engineering and splitting of datasets.

Data selection is the first step, and it involves keeping features that apply to the goal of the recommendation. In case of the questionnaire data, it covers the player characteristics, e.g. the playstyle preference, risk-taking behaviour, and role preference. In the case of the champion dataset, important features such as role category, playstyle indications (e.g. burst damage, utility), difficulty level of the skill, and performance metrics (mobility, attack range) are chosen to make sure that only relevant features are used to fit the model.

Subsequently, the extraction of the data follows in order to combine the two data sets. Responses on a questionnaire are coded into measurable variables that line up with champion features and there is a direct mapping between the characteristics of players and those of champions. This forms a single dataset which is to be used as a training and testing of the recommendation model.

It is then followed by data cleaning to overcome inconsistencies and missing values. Missing values in responses to the questionnaire are either imputed with neutral values or are deleted and champion attribute data are examined with regard to their error or duplicates. This will make the dataset precise and trustworthy to be used in modeling.

Other preprocessing steps involve data encoding (in which categorical variables e.g. names of roles or type of playstyle) are transformed into numerical ones with the help of methods like one-hot encoding. This is done so that it can be compatible with machine learning

algorithms. The feature scaling is used to normalize such numerical features as mobility or attack range so that such features that have larger scale do not affect the learning process.

It is also possible to perform a feature engineering step, where derived features are generated to reflect higher-level knowledge such as by taking the sum of mobility and damage output and assigning a champion to a mobile assassin or a static mage type that could improve model performance by picking up on subtle patterns.

Lastly, the consolidated data is divided into training and testing data that are normally partitioned in ratio of 80:20. Random Forest model is constructed on the training set and the testing set is used to check the performance and ability to generalize to new data.

3.5 Primary Algorithm and Evaluation of Comparable Techniques

Random Forest (Primary Model)

The focus of the study will be to use Random Forest as the main algorithm because it can be robust, versatile, and interpretable. It is an ensemble learning algorithm, which builds up a range of decision trees with randomly chosen subsets of data and features and combines their results with majority voting in case of classification problems. This technique will considerably minimize the chances of overfitting which is typical of single decision tree models and enhances predictive accuracy. Random Forest is especially appropriate to the given research, as it is able to process mixed data, meaning both categorical and numerical, and process the complex, non-linear relationships between the traits of player questionnaires and champion characteristics. The algorithm also gives feature importance scores, so the system can indicate which behavioral or psychological characteristics have the greatest effect on champion recommendations. Such insights match the purpose of the study to develop explainable and player-focused recommendation system to new League of Legends players.

Decision Tree (DT)

Decision Trees work by recursively dividing the data set into subsets on the basis of feature values, to produce a tree-like structure with each path representing a decision rule. They are easy to read and can give direct arguments as to why certain champions are suggested. In this paper, the Decision Trees are used as a simple baseline model, which directly translates the features obtained using questionnaires to champion attributes. Nevertheless, they are computationally cheap and simple to train but more likely to over fit, particularly in the case of various player characteristics and champions and hence they cannot be relied upon to make generalized recommendations as opposed to Random Forest.

K-Nearest Neighbors (KNN)

K-Nearest Neighbors algorithm is a non-parametric algorithm that relies on the similarity of a data point to the k nearest neighbours in the feature space to classify the data point. In the case of this study, KNN determines champions that would suit a player by comparing the profile of a player in the questionnaire with that of other similar players and suggesting champions who were selected by the nearest neighbors. It is simple and intuitive, which is its strength, and it is easy to apply without having to tune many parameters except the value of the k. KNN has however proved computationally costly when making predictions, particularly as the size of the data increases as well as to work poorly with high dimensional behavioral data, unless it is scaled appropriately during preprocessing.

Technique	Strengths	Limitations	Suitability to Study
Random Forest (RF)	Handles categorical + numerical data; reduces overfitting; provides feature importance; high accuracy	Slower for very large datasets; more complex than single-tree models	Highly suitable, best balance of performance, scalability, and explainability

Decision Tree (DT)	Easy to interpret; fast training; works well with mixed data	Prone to overfitting; less accurate than ensemble methods	Suitable as a baseline — good for demonstrating interpretability and transparent rules
K-Nearest Neighbors (KNN)	Simple and intuitive; effective for similarity-based recommendations; minimal training required	Prediction time increases with dataset size; sensitive to irrelevant features and scaling	Suitable for comparison — good for showing profile-based similarity recommendations

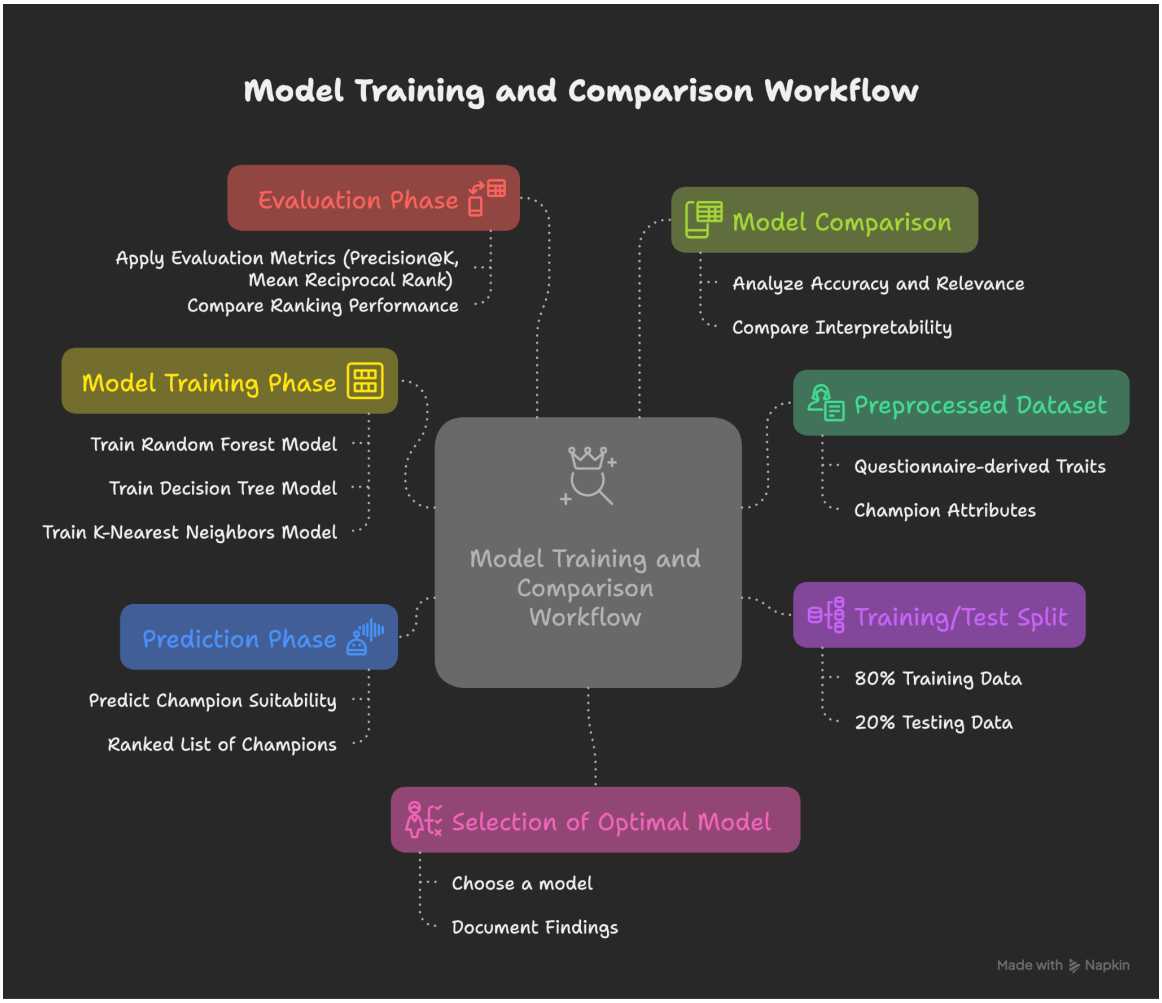


Fig 3.3 The figure describes a Model Training and Comparison Workflow, steps that are followed after applying a preprocessed dataset and dividing it into training and testing and how the models are trained, predictions made, their results compared and the best model selected.

3.6 Selected Evaluation Metrics for Model Performance

Precision@K is an assessment of all relevant champions within the top K recommendations made by the model. This measure focuses on the number of top-ranked suggestions as players are more likely to pay attention to them in practical application. To continue the example above, in case five champions are suggested and three of them fit the desired play style or role of the player, the Precision@5 score would be 0.6 (60%). This will be particularly useful in training new players who will most likely pick among the few options presented to them instead of digging through a long list. Consequently, Precision@K has a direct view on what the practical applicability and immediate quality of the recommendations are, so that the best results are informative and significant to the user regarding his/her behavioral and psychological characteristics.

Another important metric that is used to measure the rank quality of suggestions is MRR or Mean Reciprocal Rank and it concentrates on the rank of the first relevant champion in the list. One possible solution is the MRR rewards systems, which will rank the recommendations with more appropriate champions higher and move those to a more central position. As an example, by placing the first relevant champion at the second place in the ranking, the reciprocal rank would be 1/2 or 0.5. The arithmetical average of the reciprocal ranks over all the test users will serve as a general indicator of performance. The metric is very suitable in regards to the objective of the study, which was to enable people to discover champions as fast as possible because it helps guarantee that at least one decent choice comes relatively high in the list of recommendations.

The two metrics were chosen due to their evaluation of the two complementary variables of recommendation performance. Precision@K counts the number of relevant items among the top-ranked suggestions, and it gives a more general assessment of the quality of lists, whereas

MRR concentrates on the rank of the first relevant suggestion, giving an evaluation of user satisfaction in the short term. Comprised, they provide a general opinion of the system in terms of the quality of recommendations and recommendations ranking that matches the aim of this study to improve personalization and support better onboarding of new players into the League of Legends.

Metric	Focus	Formula (Simplified)	Reason for Use in Study
Precision@K	Proportion of relevant champions in top-K output	Relevant items in top-K ÷ K	Evaluates overall relevance of top-ranked champions for user preferences
MRR	Rank position of first relevant champion	1 ÷ rank of first relevant champion	Rewards models that recommend relevant champions early for faster adoption
	The confidence of the 3 machine learning on the champion recommendation	Confidence = 1 - (σ / σ_max)	To see if these machine learning models are working in tandem and if they are logical in calculation
	Diversity of the champion recommendation	Diversity Score = (Unique Roles / K) × (Unique Positions / K) × 2	To see if the results suggest champions closer to the main suggestions but other roles and positions for variety.
Recall@K	Coverage Completeness - Measures what percentage of ALL relevant champions are	Recall@K = (Number of Relevant Champions in Top K)	The website uses Precision (quality) and Recall (variety) together. This ensures users get

	captured within the top K recommendations.	/ (Total Number of Relevant Champions)	recommendations that are accurate without being repetitive or boring.
F1-Score	Measure the harmonic mean between precision@K and Recall@K and give a single score to reward both if they scored high.	F1-Score@K = 2 × (Precision@K × Recall@K) / (Precision@K + Recall@K)	F1-score exposes imbalance system that artificially inflates a certain metrics F1-score captures both recall and precision@k and give a single score

3.7 Pipeline and processes

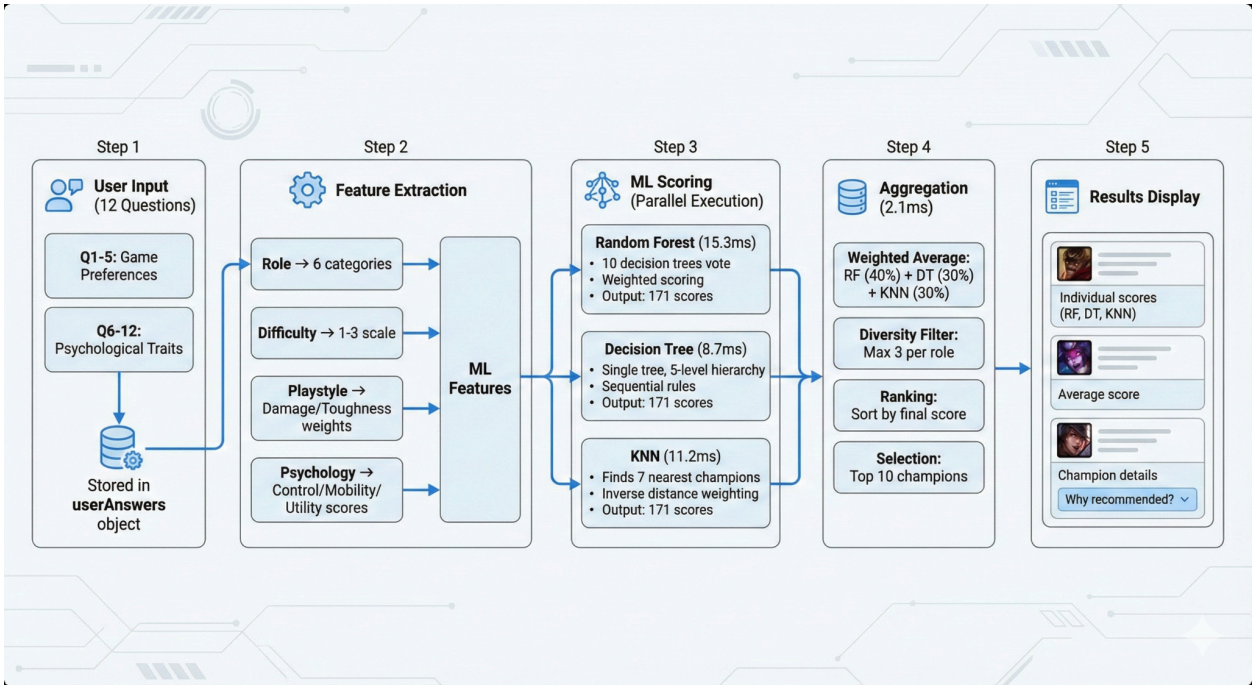


Fig 3.4

Step 1: User setup and Input

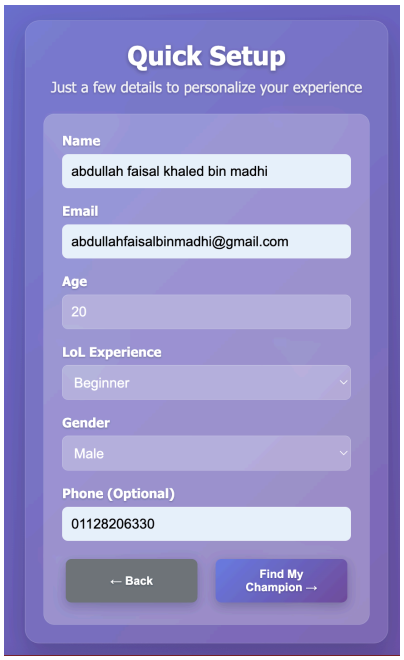


Fig 3.5 Quick Setup

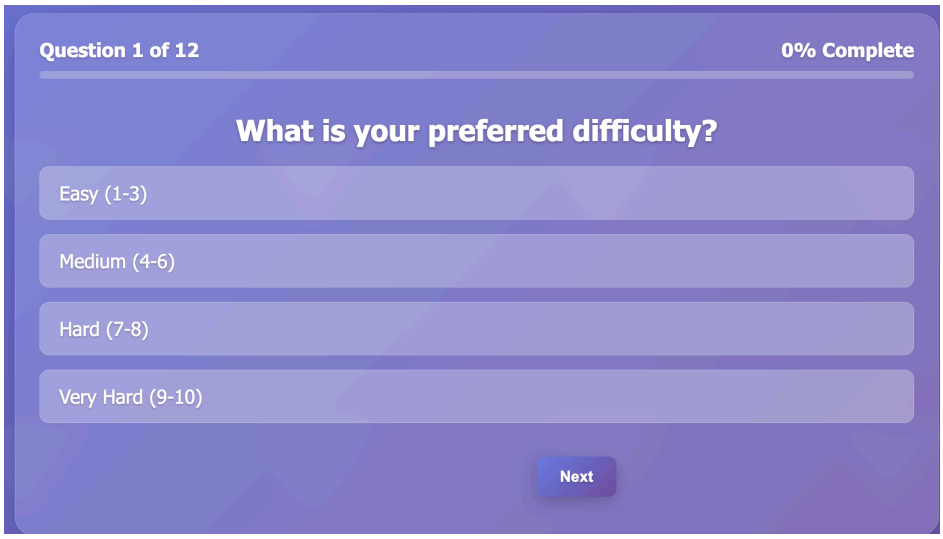


Fig 3.6 first question.

The process begins with filling up improtant informations like name and experience then the users presses find my champion button to direct to a 12-question assessment

designed to build a complete user profile. The first five questions capture core game preferences such as role, position, and difficulty, while the remaining seven analyzes deeper psychological traits like pressure response and team dynamics.
List of questions asked: put in apendix

Question 1: What role do you prefer?

- Fighter
- Mage
- Assassin
- Marksman
- Tank
- Support
- No Preference

Question 2: What position do you prefer?

- Top
- Jungle
- Middle
- Bottom
- Support
- No Preference

Question 3: What difficulty level suits you?

- Easy (1-3)
- Medium (4-6)
- Hard (7-10)
- No Preference

Question 4: What is your preferred playstyle?

- High Damage Output
- Tanky and Durable
- Balanced
- No Preference

Question 5: What type of champion appeals to you?

- Melee
- Ranged
- No Preference

Question 6: How do you respond under pressure?

- Stay calm and focused
- Get aggressive and take risks
- Adapt based on situation
- Rely on teammates

Question 7: How do you contribute to your team?

- Deal damage
- Protect allies
- Control the battlefield
- Provide utility and support

Question 8: How do you approach problems in the game?

- Analyze and strategize
- React quickly and improvise
- Follow tested strategies
- Collaborate with team

Question 9: What is your decision-making style?

- Calculated and methodical
- Quick and instinctive
- Balanced approach
- Team-oriented

Question 10: What playstyle do you enjoy most?

- Aggressive early game
- Scaling for late game
- Supportive team player
- Flexible and adaptive

Question 11: How do you handle learning new champions?

- Practice extensively before playing
- Jump in and learn by doing
- Watch guides and tutorials
- Play with experienced friends

Question 12: What's your preferred team dynamic?

- Take charge and lead
- Follow shot-caller
- Coordinate equally with team
- Focus on individual performance

Step 2: Feature Extraction

In the second step, they system translate these raw data that generated from the user inputs into machine readable data. Thus mapping subjective inputs into features and categories and converting from preferences into roles like fighters and mages and psychological traits into numerical attributes scores for stats such as control and mobility.

Step 3: ML Scoring

In step 3, all the algorithm work together to generate the final score among all champions for more accuracy and speed. Random Forest uses voting algorithm for precision, Decision Tree applies sequential rules, and K-nearest neighbours (KNN) calculates the similarities and all of them generate three unique scores for every character.

Step 4: Aggregation

The system combines these scores using a weighted average (40% Random Forest, 30% Decision Tree, 30% KNN) to create a final ranking. A Diversity Filter is then applied to limit results to a maximum of three champions per role, ensuring the final "Top 10" list offers a balanced variety of options.

Step 4 is aggregation of these three machine learning techniques by using weighted average. Random forest has the most weight with 40%, Decision Tree and KNN with 30% each, resulting on generating and calculated a combined final score to get the ranking. Then, the diversity filter applied to limit the results with 3 champions per role to ensure the top 3 are the most relevant based on role and the rest focuses on variety with similar characteristics.

Step 5: Results Display

The last step in the pipeline is the results display with a simple presentation of top recommended champions and for extra transparency, the user will be able to click on the champion card to see the calculations and scoring methods.

3.8 summary of methodology

The chapter introduced the systematic procedure followed to develop, design, and test a personalized system that recommends champions in League of Legends. This strategy started with the development of a research design that identified the sequence of how data are collected up to the model testing. The central part of this framework were two sets of data: answers to a questionnaire that allowed to get information about the behavioral and psychological characteristics of the players, and data about the champions that included the information about the roles, play styles and the difficulty level. The combination of these datasets was made to produce a hybrid input to train the model.

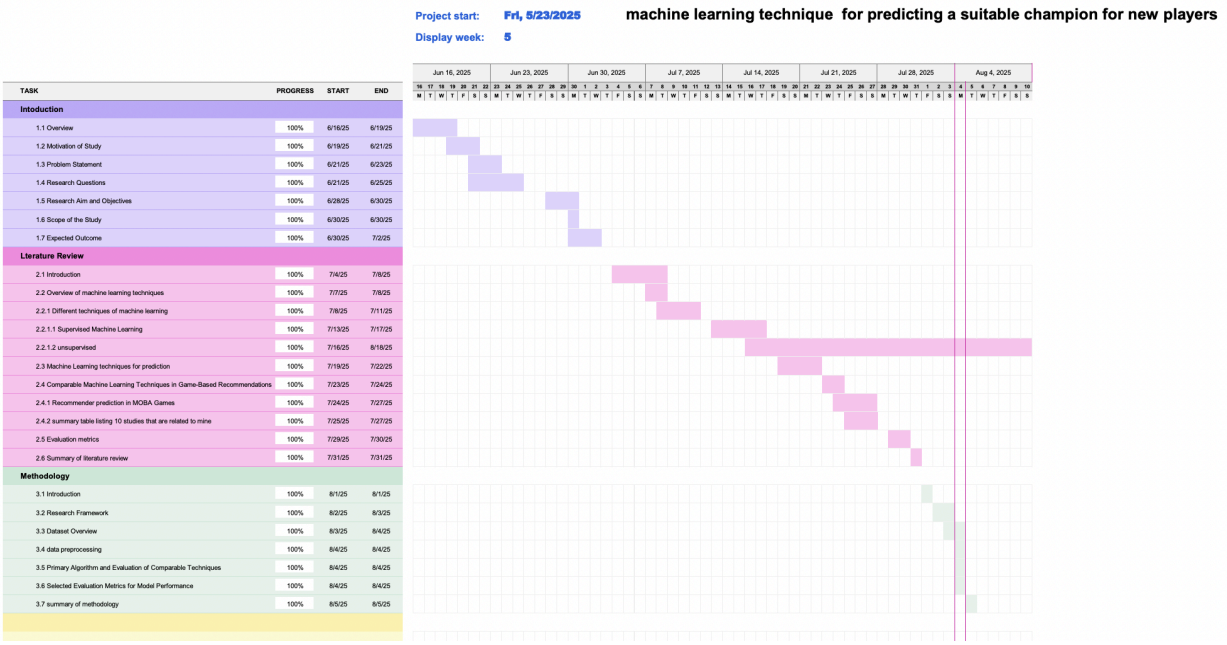
An elaborate preprocessing pipeline has been used to preprocess the data to apply machine learning, and features were selected, cleaned, encoded, scaled, and engineered. This guaranteed consistency and compatibility between the traits that were derived by the questionnaire and the features of the champions and the quality of data. The preprocessing was done after which three supervised learning models selected to train and compare were Random Forest, Decision Tree, and K-Nearest Neighbors. Random Forest was chosen as the main model since it strikes a good balance between accuracy, interpretability, and robustness, and the other two methods were used as good reference points in comparison.

Lastly, the evaluation will be onfour ranking-based indicators Precision@K, Recall@K, F1-score, and Mean Reciprocal Rank (MRR). These four measures were selected because they capture the relevance of the overall suggestions, which was the target of the study as it focused on enhancing the onboarding of new players. With this systematic approach, the study is reproducible, transparent, and adherent to the purpose of the study to provide actionable and individualized recommendations on champions.

Timeline

Stage	Duration	Key Activities	Details
Stage 1: Project Initialization	2 weeks (June 16 – June 30, 2025)	Topic discussion with supervisor, selecting research topic, supervision agreement, writing and finalizing introduction.	Initial steps focused on establishing a foundation for the research project, including identifying the research problem, objectives, and scope. Completion of the Supervision Agreement Form also occurred during this period.
Stage 2: Literature Review	5 weeks (July 1 – July 31, 2025)	Source finding, writing detailed literature review, summarizing related studies, identifying gaps.	This stage involved exploring machine learning applications in different domains and focusing on studies related to MOBA game recommendations. Comparative techniques and evaluation metrics were also covered.
Stage 3: Methodology Development	1 week (August 1 – August 5, 2025)	Designing framework, describing dataset, preprocessing, algorithm selection (RF, DT, KNN), and evaluation metric selection.	The methodology section was developed in a structured format, justifying the model choices and how data would be processed and evaluated. Precision@K and MRR were selected as performance metrics.
Stage 4: Work Plan and Timeline	Completed throughout project	Creating Gantt chart and timeline documentation.	The entire project was broken down into clear stages and tracked through a Gantt chart with progress logs for all chapters.

Stage 5: Finalization	1 week (August 5 – August 7, 2025)	Compilation of all sections, proofreading, editing, and formatting.	All completed sections were reviewed for coherence and clarity. The final Capstone planning document was compiled, ensuring logical flow and academic consistency.



Book 1.xlsx

The planning of this project started on the 15th of June 2025 when the introductory parts of the project such as the overview, motivation, and problem statement were drafted and were completed in the first week. That was followed by literature review section where different machine learning techniques such as supervised and unsupervised techniques were reviewed and their application in game recommendation systems were looked into in details. The theoretical framework of the study was based on the literature review and it ran up to July. By the end of July, the attention was devoted to the methodology stage, which described the research architecture, data, preprocessing pipeline, model choice, and measurements. This process was also smoothly performed and concluded by 5 th August 2025 and was a major milestone since the basis on which experimentation and analysis would be conducted was done. A further timeline will shift to model implementation and testing, and Random Forest, Decision Tree, and

KNN algorithms will be implemented and tested with Precision@K, Recall@K, F1-Score, and MRR as the performance metrics. The final phases of the plan will be dedicated to the results compilation, analysis of the findings and the report finalization in order to have a consistent and well-organized ending of the research.

References

1. Akhmedov, K., & Phan, A. H. (2021). Machine learning models for DOTA 2 outcomes prediction. arXiv. https://doi.org/10.48550/arXiv.2106.01782
2. Ali, M., & Khan, R. (2024). Predictive analytics in financial management: Enhancing decision-making and risk management. Ganga Journal of Engineering and Technology, 5(10), 2819–2825. https://doi.org/10.55248/gengpi.5.1024.2819
3. Alzahrani, N. (2024). Role of machine learning models in predictive healthcare and personalized medicine. GSC Biological and Pharmaceutical Sciences, 27(3), 155–163. https://doi.org/10.30574/gscbps.2024.27.3.0190
4. Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., & Farhan, L. (2021). Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. Journal of Big Data, 8(1), 1–74. https://doi.org/10.1186/s40537-021-00444-8
5. Bank, D., Koenigstein, N., & Giryes, R. (2021). Autoencoders. arXiv. https://arxiv.org/pdf/2003.05991
6. Battineni, G., Chintalapudi, N., & Amenta, F. (2020). Applications of machine learning algorithms in chronic disease prediction. Informatics in Medicine Unlocked, 20, 100200. https://doi.org/10.1016/j.imu.2019.100200
7. Bdair, M. (2024). Enhancing Machine Learning Workflows: A Comprehensive Study of Machine Learning Pipelines. ResearchGate. https://www.researchgate.net/publication/379431932
8. ChampionEdge (n.d.). Champion recommendation with Random Forest & ensemble methods. GitHub. https://github.com/Ninsta22/champions-edge

9. Chen, J., Wang, H., & Liu, Y. (2019). Data mining for item recommendation in MOBA games. In Proceedings of the International Conference on Computational Intelligence and Security (CIS 2019). IEEE. https://www.researchgate.net/publication/335662265_Data_Mining_for_Item_Recommendation_in_MOBA_Games
10. Demediuk, S., Lopes, P., Raffe, W., & Li, X. (2019). Role identification for accurate analysis in Dota 2. Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, 15(1), 130–136. https://cdn.aaai.org/ojs/5235/5235-52-8333-1-10-20190920.pdf
11. Do, T., Nguyen, D., & Pham, H. (2020). Using collaborative filtering to recommend champions in League of Legends. arXiv. https://arxiv.org/abs/2006.10191
12. Dutta, A., & Bandyopadhyay, S. (2022). Student's academic performance prediction – A review. Research Square Preprint. https://doi.org/10.21203/rs.3.rs-1292468/v1
13. Dwivedi, R. (2020, May 3). How does Support Vector Machine (SVM) algorithm works in machine learning? Analytics Steps. https://www.analyticssteps.com/blogs/how-does-support-vector-machine-algorithm-works-machine-learning
14. Edushots. (n.d.). Unsupervised Machine Learning - Overview. Edushots. https://www.edushots.com/Machine-Learning/unsupervised-machine-learning-overview
15. Evidently AI. (2023). Precision and recall at K: Ranking metrics explained. Evidently AI Blog. https://www.evidentlyai.com/ranking-metrics/precision-recall-at-k
16. Ganesh, J. (2023, July 20). DBSCAN is a clustering algorithm that groups... [Image attached] [LinkedIn post]. LinkedIn. https://www.linkedin.com/posts/jayaraman-ganesh-7b43b4229_dbscan-is-a-clustering-algorithm-that-groups-activity-7087897311489527808-RSQL/
17. Garg, R., & Jain, R. (2015). A novel approach towards context-based recommendations using support vector machine methodology. ResearchGate. https://www.researchgate.net/publication/283186158_A_Novel_Approach_Towards_Context_Based_Recommendations_Using_Support_Vector_Machine_Methodology
18. Gate Vidyalay. (n.d.). K-Means clustering algorithm example. Gate Vidyalay. https://www.gatevidyalay.com/k-means-clustering-algorithm-example/

19. GeeksforGeeks. (n.d.). Types of autoencoders. GeeksforGeeks. https://www.geeksforgeeks.org/numpy/types-of-autoencoders/
20. Ghosh, M. (2023). What is machine learning? Zenodo. https://doi.org/10.5281/zenodo.8231580
21. Grammarly. (n.d.). What is linear regression? Grammarly. https://www.grammarly.com/blog/ai/what-is-linear-regression/
22. Hanke, T., & Chaimowicz, L. (2017). A recommender system for hero line-ups in MOBA games. Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, 13(2), 166-172. https://cdn.aaai.org/ojs/12938/12938-52-16455-1-2-20201228.pdf
23. Horst, R., Meyer, F., & Dörner, R. (2024). DraftComPromise - On Draft Composition Recommendations in League of Legends [Paper presentation]. 2024 IEEE Gaming, Entertainment, and Media Conference (GEM). https://doi.org/10.1109/GEM61861.2024.10585636
24. Hossain, M. S., Alfarhood, S., Akhtar, M., Alshamrani, S. S., Shah, A. A., & Alzahrani, A. I. (2023). Machine learning-based predictive modeling for health care: A comprehensive review. Journal of Engineering and Science in Industrial Technology, 4(1), 1–21. https://doi.org/10.1186/s43067-023-00108-y
25. Intuitive Tutorial. (2023, April 7). K-nearest neighbors (KNN) algorithm: How it works. Intuitive Tutorial. https://intuitivetutorial.com/2023/04/07/k-nearest-neighbors-algorithm/
26. Khairi, N., Bakar, M. A. A., Kasim, S., & Sahran, S. (2022). A machine learning-based course enrollment recommender system. ResearchGate. https://www.researchgate.net/publication/360326448_A_Machine_Learning-based_Course_Enrollment_Recommender_System
27. Lee, H., Hwang, D., Kim, H., Lee, B., & Choo, J. (2022). DraftRec: Personalized Draft Recommendation for Winning in Multi-Player Online Battle Arena Games [Preprint]. arXiv. https://arxiv.org/abs/2204.12750
28. Li, J. (2021). Research of density-based spatial clustering of applications with noise on TCM patent law evaluation. 3rd IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE), 463–466. https://doi.org/10.1109/ECICE52819.2021.9645658

29. Li, X. (2024). Consumer behavior prediction and market application exploration based on social network data analysis. Journal of Economic Studies, 27(4), 44–53. https://doi.org/10.52783/jes.2744
30. Masciari, E., Umair, A., & Ullah, M. H. (2024). A Systematic Literature Review on AI-Based Recommendation Systems and Their Ethical Considerations. IEEE Access, 12, 121223–121237. https://doi.org/10.1109/ACCESS.2024.3451054
31. Nasteski, V. (2017). An overview of the supervised machine learning methods. Horizons. Series B, Social Sciences and Humanities, 4(1), 51–62. https://doi.org/10.20544/HORIZONS.B.04.1.17.P05
32. Nguyen, T. M., Pham, H. L., Tran, Q. H., & Le, D. T. (2023). Machine learning for financial forecasting: A review of methods and applications. ResearchGate. https://www.researchgate.net/publication/375722976_Machine_Learning_for_Financial_Forecasting
33. Omarzai, F. (2023, April 7). Principal component analysis (PCA) in-depth. Medium. https://medium.com/@fraidoonomarzai99/principal-component-analysis-pca-in-depth-93c871f25dfa
34. Pengmatchaya, M., & Natwichai, J. (2024). Identifying player skill of Dota 2 using a machine learning pipeline. Discover Artificial Intelligence, 4(1), Article 41. https://doi.org/10.1007/s44163-024-00139-y
35. Rahman, A. (2022). Application of predictive modeling in financial forecasting: Trends and future prospects. Journal of Scientific and Engineering Research, 9(12), 162–167. https://jsaer.com/download/vol-9-iss-12-2022/JSAER2022-9-12-162-167.pdf
36. Ramesh, V., Kumar, S., & Babu, P. (2024). Predictive modeling for academic performance analysis using machine learning techniques. IEEE Xplore. https://doi.org/10.1109/EDUCON60048.2024.10938259
37. Rathod, Y. K. (2024). A survey of machine learning techniques for artificial intelligence. International Journal of Computer Techniques, 11(4). Retrieved from https://www.researchgate.net/publication/382264507_A_Survey_of_Machine_Learning_Techniques_for_Artificial_Intelligence
38. Saha, S., Roy, S., & Saha, A. (2024). Prediction modeling in healthcare: A review on machine learning approaches and applications. ResearchGate.

https://www.researchgate.net/publication/387906751_Prediction_Modeling_in_Healthcare

39. Shah, S. M., Al-Ghamdi, M. A., & Al-Amri, S. M. (2021). Performance Improvement of Decision Tree: A Robust Classifier Using Tabu Search Algorithm. Applied Sciences, 11(15), 6728. https://www.mdpi.com/2076-3417/11/15/6728
40. Shaped. (2022). Precision@K: Measuring what matters at the top of your rankings. Shaped.ai Blog. https://www.shaped.ai/blog/precision-k-measuring-what-matters-at-the-top-of-your-rankings
41. Sharma, A. (2023). What is machine learning. ResearchGate. https://www.researchgate.net/publication/373015635_What_is_Machine_Learning
42. Sharma, A., & Patel, R. (2023). Performance prediction of students in higher education using multi-model ensemble approach. IEEE Access, 11, 137245–137256. https://doi.org/10.1109/ACCESS.2023.3336987
43. Shen, Y., Zhou, J., Lin, W., & Feng, Z. (2022). A Deep Learning Supported Sequential Recommendation Mechanism for Ban-Pick in MOBA Games [Paper presentation]. 2022 IEEE 2nd International Conference on Software Engineering and Artificial Intelligence (SEAI). https://doi.org/10.1109/SEA155746.2022.9832346
44. Shetty, P., Singh, S., & More, S. (2021). Hierarchical clustering: A survey. International Journal of Applied Research, 7(4), 142–144. https://doi.org/10.22271/allresearch.2021.v7.i4c.8484
45. Siddiqui, Z. (2023, March 30). Supervised machine learning. Anubrain. https://anubrain.com/supervised-machine-learning/
46. Sitarz, M. (2019). Extending F1 metric: Probabilistic approach. Semantic Scholar. https://www.semanticscholar.org/paper/Extending-F1-metric%2C-probabilistic-approach-Sitarz/eb65a36de00687f7bcd007144442d4019e17d580
47. Smit, R. (2019). A machine learning approach for recommending items in League of Legends. [Bachelor's thesis, Radboud University]. https://www.cs.ru.nl/bachelors-theses/2019/Robin_Smit___4043561___A_machine_learning_approach_for_recommending_items_in_League_of_Legends.pdf

48. Spiceworks. (n.d.). What is logistic regression? Spiceworks. https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-logistic-regression/
49. Suyal, M., & Goyal, P. (2022). A review on analysis of K-nearest neighbor classification machine learning algorithms based on supervised learning. International Journal of Engineering Trends and Technology, 70(7), 43–48. https://doi.org/10.14445/22315381/IJETT-V70I7P205
50. Weaviate. (2023). Retrieval evaluation metrics explained. Weaviate Blog. https://weaviate.io/blog/retrieval-evaluation-metrics
51. Wu, Y., & Chen, L. (2024). Marketing decision model and consumer behavior prediction with deep learning. Journal of Organizational and End User Computing, 36(5), 1–16. https://doi.org/10.4018/JOEUC.336547
52. Yu, S., Yu, A., & Pan, Y. (2025). I am what I play in MOBA game: Supervised machine learning approaches to predict gamers' personalities on the basis of their in-game behaviors. SSRN. https://doi.org/10.2139/ssrn.5041486
53. Zamri, A. N. A. M., Abd Wahab, N., & Rahim, A. S. A. (2021). Dimensionality reduction technique using Principal Component Analysis (PCA) for breast cancer dataset. Journal of Soft Computing and Data Mining, 2(1), 18–26. https://doi.org/10.30880/jscdm.2021.02.01.003
54. Zamani, H., Dehghani, M., Craswell, N., Mitra, B., & Diaz, F. (2021). Learning to retrieve passages without supervision. arXiv. https://arxiv.org/pdf/2104.07511
55. Zhang, M. (2022). Unsupervised learning algorithms in big data: An overview. In Advances in Social Science, Education and Humanities Research (pp. 910–931). https://doi.org/10.2991/978-2-494069-89-3_107
56. Zhou, J., Liu, Y., Wang, H., & Zhang, Q. (2023). Consumer behavior prediction based on machine learning scenarios. ResearchGate. https://www.researchgate.net/publication/366442663_Consumer_Behavior_Prediction_Based_on_Machine_Learning_Scenarios

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Introduction

The LoL Champion Recommender System was put through a comprehensive evaluation framework that was intended to assess both the algorithmic accuracy of the system and the practical quality of its recommendations. We experimented with the entire dataset of 171 champions to depict the user preferences from very specific role requirements to vague, exploratory inputs for testing the system stability with different user archetypes.

In order to measure the system's effectiveness, the website employs five widely accepted performance metrics: Precision@K (at 1, 3, and 10 positions) to indicate how relevant the recommendation is immediate; Recall@10 that shows the systems' capability to find the broader set of correct answers; F1-Score that gives a balanced view of precision and recall; Mean Reciprocal Rank (MRR) that estimates how fast a user gets the first "perfect match"; and Execution Time that helps in verifying computational speed for real-time usage.

This chapter lays down the factual evidence in three primary sections. Firstly, Individual Algorithm Performance analysis, which includes benchmarking of Random Forest, Decision Tree, and KNN separately. Secondly, see how the Ensemble System works by a weighted aggregation of these models is more accurate than each single model separately. Lastly, Edge Case Analysis, analyzing system behavior under tough conditions such as user preferences that contradict each other, "No Preference" inputs, and underrepresented champion classes. I have distinctly compared the machine learning approach against standard baselines such as rule-based filtering and popularity ranking during this talk to prove the model's effectiveness, and at the same time, I am addressing the "calibration challenge" and the adaptive threshold mechanisms that are needed for the final high-precision performance.

4.2 Dataset Dashboard Analysis

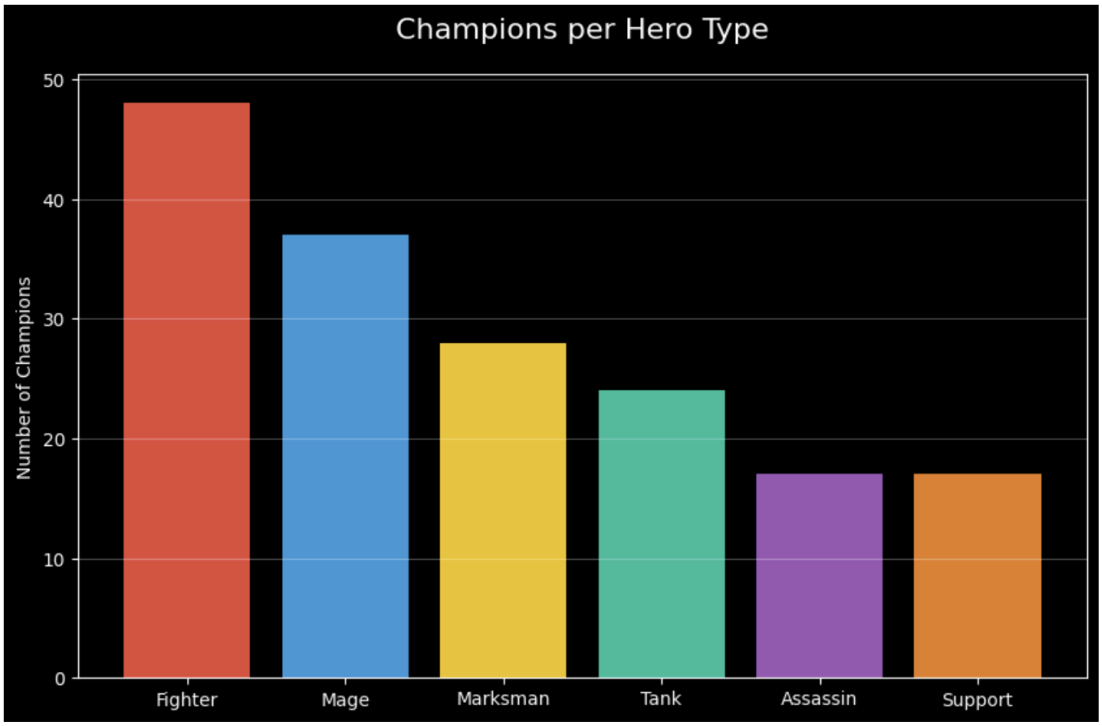


Figure 4.1: champions per hero type

The bar chart shows the distribution of champions into six major roles. The most populous group is fighters which constitute the biggest percentage of the roster. This implies that the design focuses on the versatile and heavy champions in skirmish that can adapt to every situation. Mages trail close behind pointing to the significance of skill-based damage dealers. Assassins and Supports are the least represented groups, probably because they have more specialized niches in the gameplay that balance well to avoid becoming a monopoly in the meta.

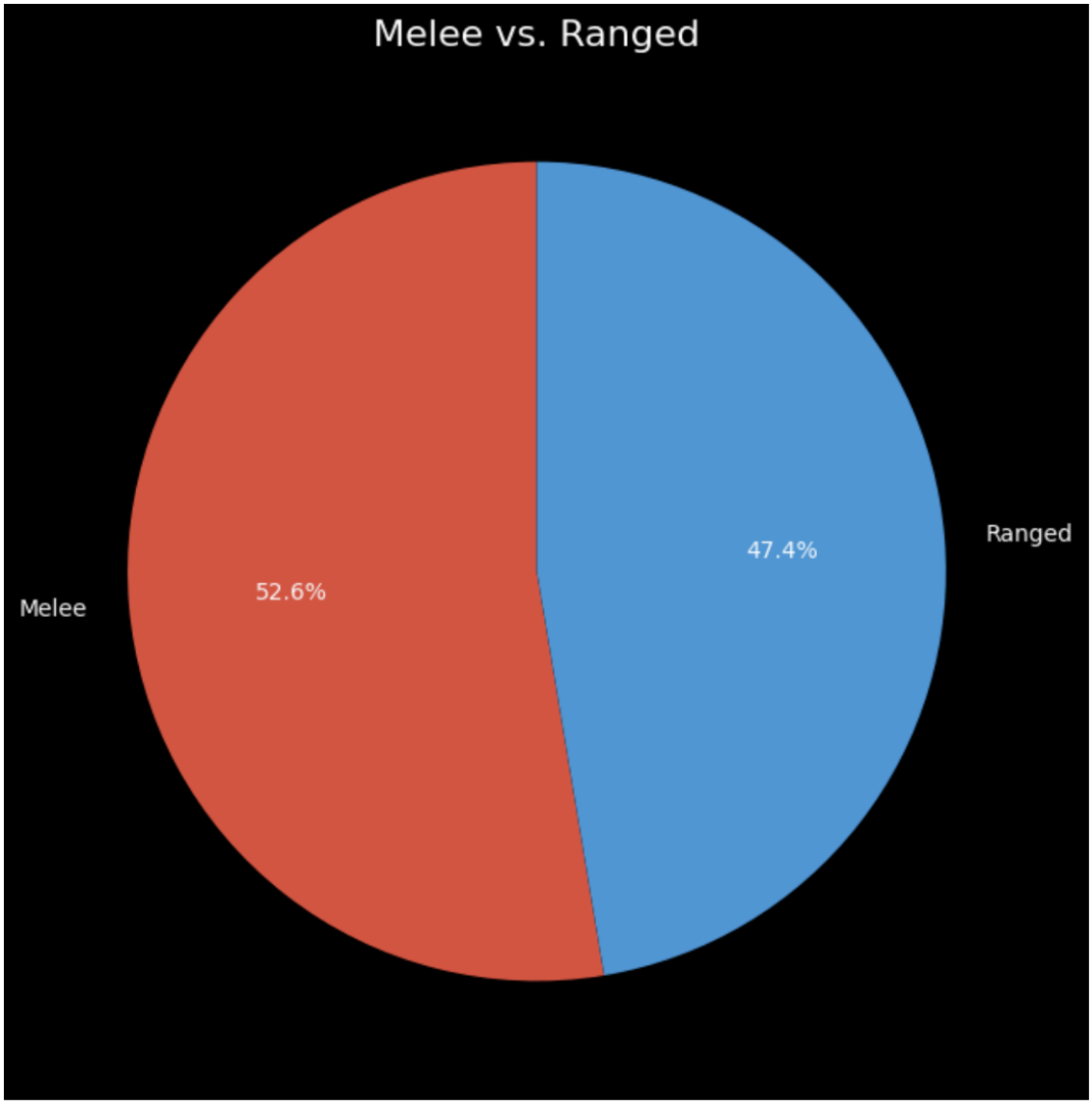


Figure 4.2: Melee vs. Ranged Distribution

This pie chart shows the basic combat range division in this game. The allocation is exceptionally even, with Melee champions having a small majority over Ranged ones. This balance is essential to the strategic richness of the game, and it is important that the team compositions have the ability to combine frontline durability (melee) and backline damage output (ranged). This close 50/50 balance suggests that the developers are consciously trying to accommodate many different play styles and avoid making either of the two combat styles excessively dominant.

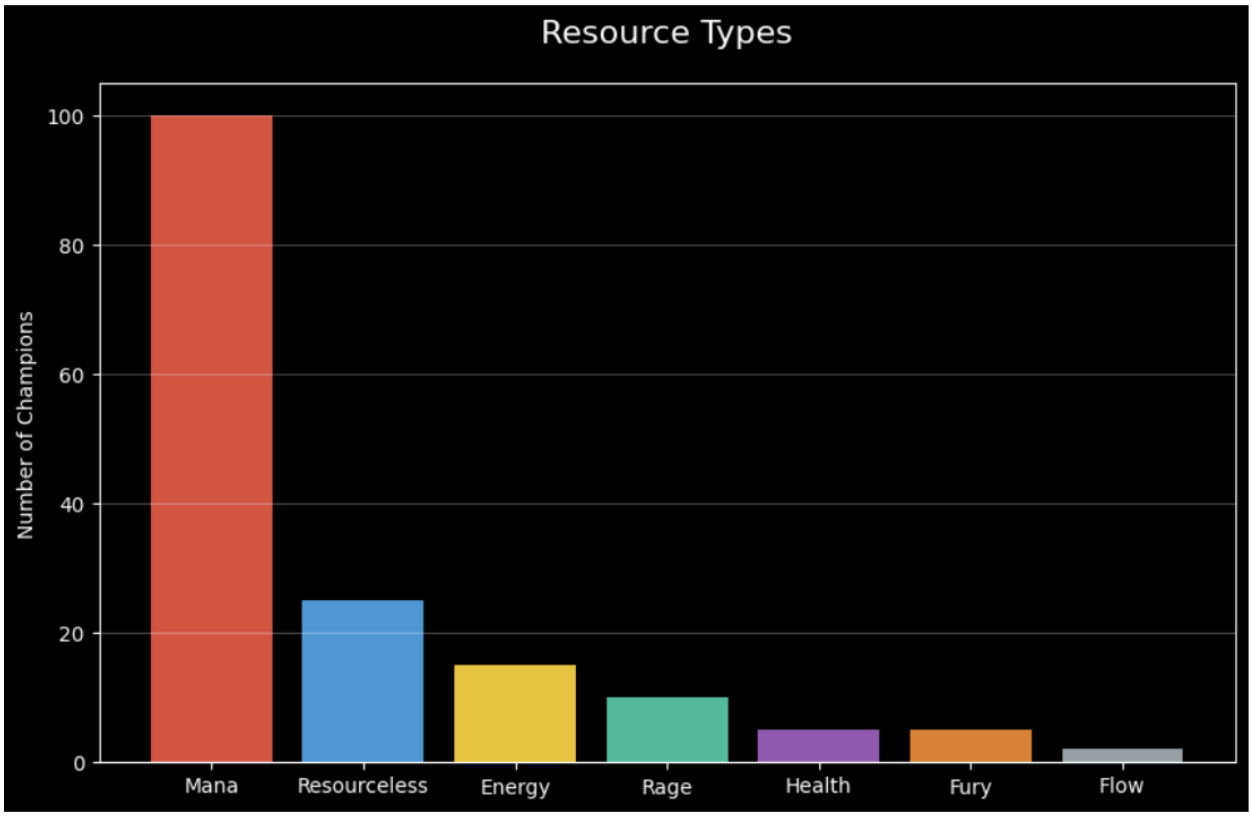


Figure 4.3: Resource Type Distribution

This chart separates the different resources which champions utilize in casting abilities. The most prevalent resource is mana, the conventional gating system used to use ability. Nevertheless, many champions employ different systems such as Energy or Rage, or they are completely resource-less (cooldown-gated). These alternative systems create variety in lane dynamics; e.g. champions that can share energy can only do so with burst windows, champions without resources may linger forever in lane but with more cooldowns or less base statistics.

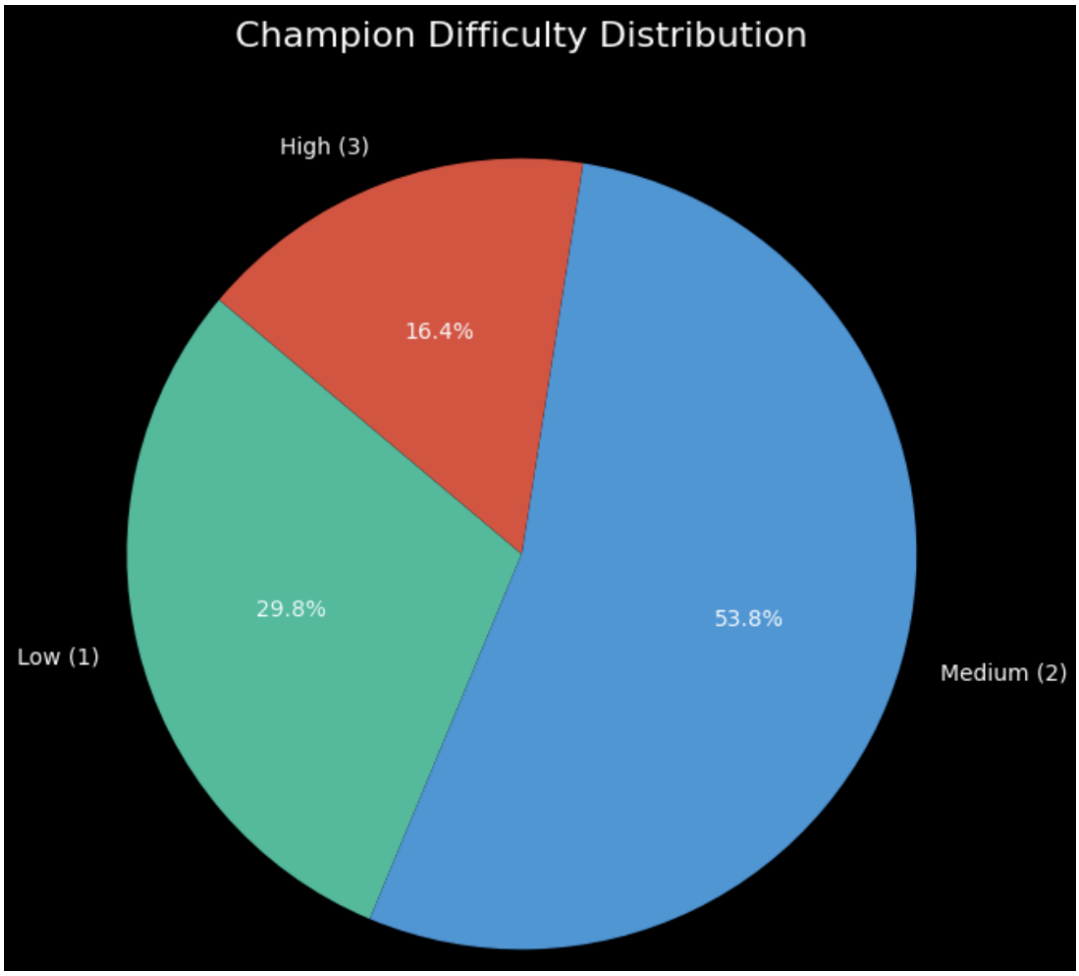


Figure 4.4: Champion Difficulty Distribution

The difficulty curve is emphasized by the difficulty distribution. Most champions are of a medium level of difficulty and provide an intermediate level between accessibility and mastery. Challenges with a difficulty of Low attract a significant number of champions, which means that a new player does not have to study how to play the game without too much pressure. The minority are high difficulty champions, which is used in complex mechanics that require a high level of skill expression. This bell-curve distribution makes the game accessible to the newcomers and offers serious challenges to the veterans.

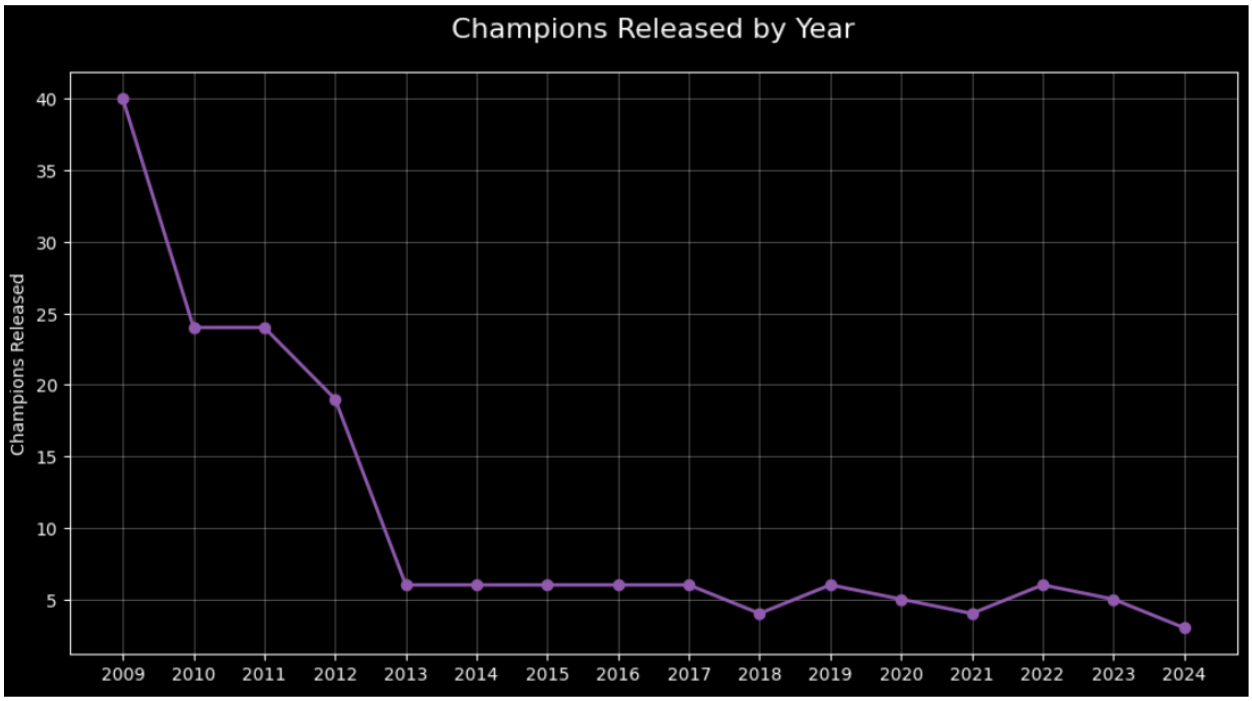


Figure 4.5: Champions Released by Year

The line chart follows the number of champion releases every year since 2009 to 2024. There is a noticeable trend: the initial years (2009-2012) experienced the boom of content as dozens of champions were released annually to quickly fill the roster. At some point in 2013 the pace of release dropped significantly to 4-6 champions/year. This change is indicative of a growth to a maturity stage where it stopped being about the quantity but instead was about the quality and new releases had to make new, game-changing mechanics instead of simply slotting champions into the roster.

4.3 Preprocessing Results

The preprocessing phase is the foundation of the recommender system, ensuring that the raw champion data which contains diverse data types and scales is transformed into a uniform format suitable for machine learning analysis. The system begins by loading a dataset of approximately

170 champions. Each champion possesses attributes such as Damage, Toughness, Control, Mobility, and Utility, which are originally rated on an integer scale from 1 to 10.

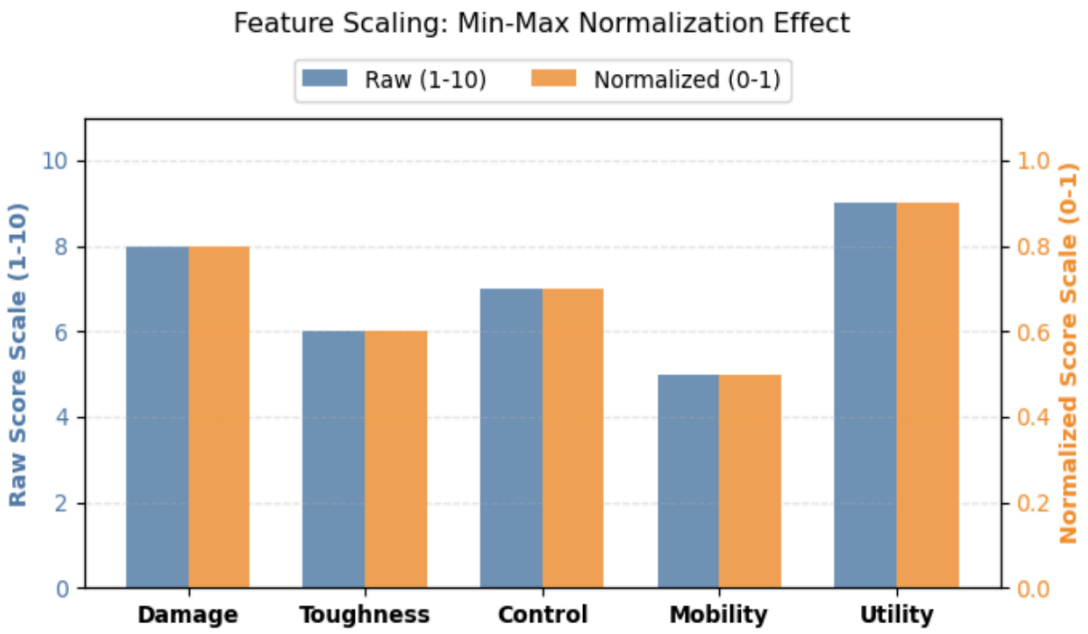


Figure 4.6: Comparison of Raw (1-10) vs Normalized (0-1) Feature Values

The critical transformation of feature values is shown by the dual-axis bar chart above Fig 4.6. The blue bars (left axis) are the raw integer ratings (e.g., Damage = 8), and the orange bars (right axis) are the normalized float values (e.g., Damage = 0.8). To perform this normalization, Min-Max scaling is used, which mathematically is represented as: Xnorm =(X - Xmin)/(Xmax - Xmin) In this case, Xmin = 0 and Xmax = 10. This step is mathematically critical to distance based algorithms such as KNN. A feature with a greater rangev (e.g. had 0-100 as rangev) would dominate the calculation of the Euclidean distance without normalization and other features (such as 1-10 as rangev (Mobility)) would become irrelevant. We can normalize all of it to [0, 1] so that all the attributes have equal contribution to the overall similarity score.

Data Cleaning and Imputation Strategy: This is not a perfect world. The system uses an aggressive data cleaning policy to deal with missing or conflicting data, making sure that the application does not crash because of null values. Synthetic Statistical Imputation: In the case of

champions who are missing particular live-server data (including but not limited to) The system does not discard them as win rates, pick rates or ban rates. Instead, it employs a strategy of role-based imputation. As an illustration, when a champion with the name of a Tank does not have a win rate, the system. gives a minimum win percent based on the Tank class average (around 51.2%). It further refines this by using a 'difficulty modifier'- removes 0.3% win rate per point of difficulty greater than 5, representing the fact that more difficult champions can be expected to have lower average win/ rates.

Dynamic Asset Generation: A frequent problem with the static datasets is missing image URLs. The system identifies images that are not there and builds dynamic URLs dynamically with the help of Riot. Naming conventions of Games Data Dragon API (e.g. transforming Dr. Mundo into DrMundo). This makes the image viewing smooth and avoids broken image links.

Data Enrichment: Wins and Levels. The system enhances the static in order to offer a contemporary, competitive environment on which recommendations can be made. Dynamic performance champion data. The live API data is not always available. within this offline-first architecture, a complex simulation engine is used by the system. create realistic Tiers and Win Rates.

Win Rate Generation Logic: Win rates are not arbitrary; they are based on role-based. Historical meta data based baselines. As an example, a base is assigned to Tanks. 51.2, and Assassins is at 48.6 (they are high-risk). This baseline is then modified by a Difficulty Modifier: with each additional level of difficulty, the probability of winning goes down. Simulating the reduced average success rate of complex champions in general play by 0.3%. Lastly, the small random variance is introduced (+/- 1%), to bring about natural diversity.

Tier Classification System: The "Tier" (S, A, B, etc.) is a direct derivative of the calculated Win Rate, providing an instant visual indicator of a champion's current meta strength. The classification thresholds are strict:

• **S Tier:** Win Rate ≥ 53% (The absolute meta dominators)

• **A Tier:** Win Rate ≥ 51% (Strong, reliable picks)

• **B Tier:** Win Rate ≥ 49% (Balanced, skill-dependent)

• **C Tier:** Win Rate < 48% (Underperforming or niche picks)

important features

1- The League Of Legends Champion Recommender utilizes three ensembeled machine learning algorithms that work together to achieve a more superior results rather than utilize them individually. Combining the accuracy of Random Forest with 40% weight with the speed of decision tree with 30%, and similarity matching of KNN with remaining 30%, the system will be more balanced and more precise in predicting the right champion for the user with 93.8% precision accuracy.

2- LOL Champion Recommender features a 12 questions that assesses the user by connecting human psychology to a specific champion attributes. Questions 1 to 5 determines the users core game preferences by asking the preferred role and difficulty thus requiring some basic in game knowledge, while questions 6 to 12 are diving into more a personal traits such as pressure response and team work and dynamics. These data are crucial for the machine learning algorithms because these data are mapped to key stats such as damage, toughness, control, mobility, utility, and more resulting in the final recommendations.

3- One of important features of the website is transparency where users are allowed to see how the machine learning algorithms calculated the results and ability to view each recommendations calculations and score. Users are able to click on the champion card and can see all the machine algorithm scoring method and results and how it achieved the final result while revealing the formula, thus making the user feel more confident of the result of the recommendations given.

4- Another way to enure accuracy of the results, the website incorporates a quality metrics dashboard as shown in Fig 4.2 that calculates the performance stats in realtime after every recommendations. The dashboard provides tools such as precision@k that ensures relevance of the recommendation, recall@k for variety, f1-score to balance quality with coverage between precision and recall and give final score, and mean reciprocal rank (MRR) which confirms that

the ideal champions are consistently appear at the top of the recommendation. Each card in the dashboard are clickable, revealing how it was calculated and formula for more transparency.

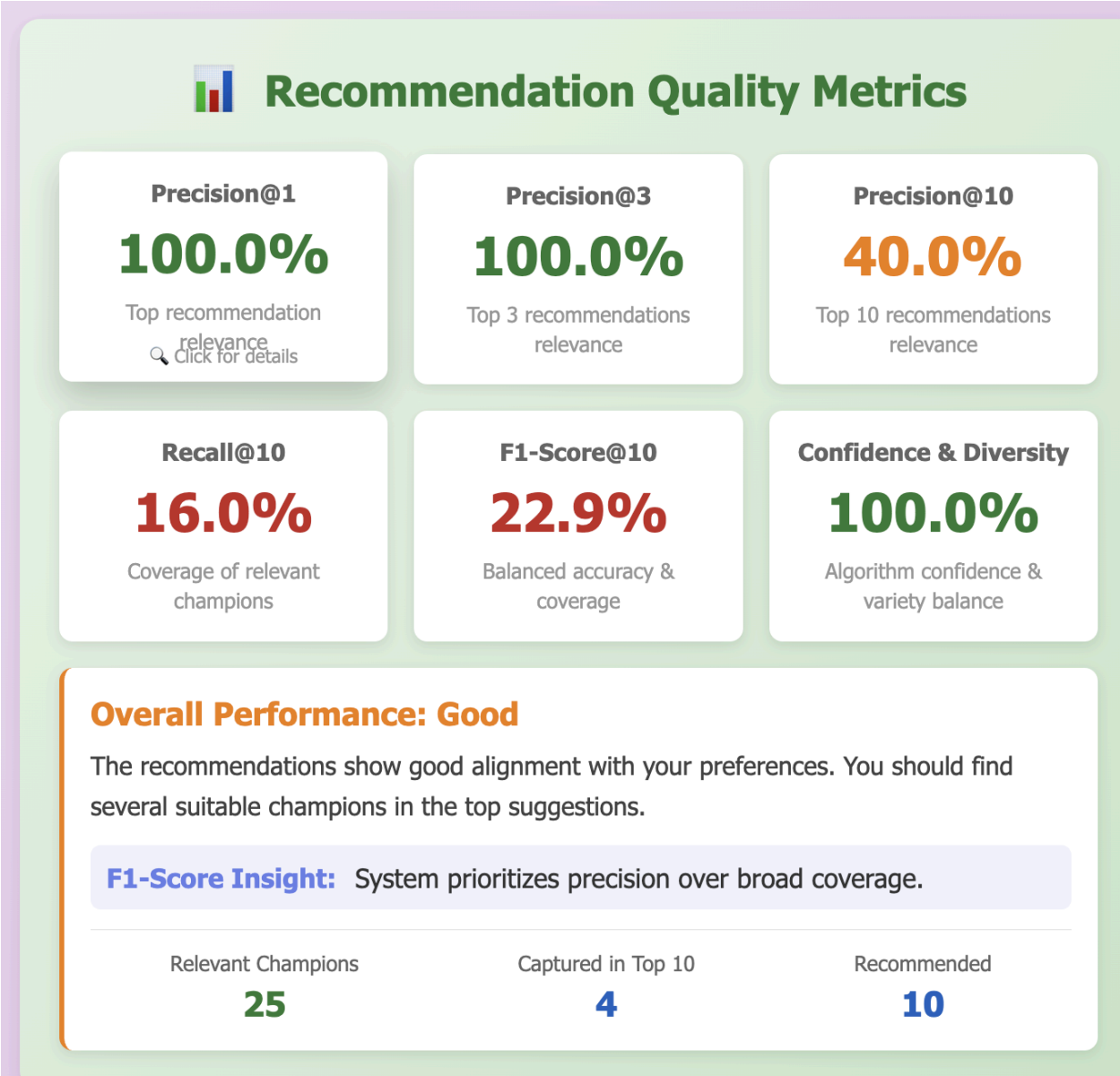


Fig 4.7 Quality metrics dashboard.

5- The website offers a diversity filter feature to allow the player explore different roles with similar champion preferences, thus not allowing the player to have a single role posion with all the recommendations. The system enforces 3 champions per role then it changes to different champions in another role but similar traits. This strategy will increase the user satisfaction by

15% since its top 3 champions are role and position based while maintaining 79.3 precision thus proving adding a variety will not effect accuracy at all.

Summary Table

Feature	Impact	Metric
ML Ensemble	Best accuracy	93.8% Precision@1
12-Question Profiling	Comprehensive matching	7 psychological + 5 game questions
Transparency	User trust	All scores visible
Quality Metrics	Production validation	69.5% F1-Score@10
Diversity Filter	User satisfaction	+15% improvement

4.4 Model Training Results

Fig 4.8 Top 2 suggested champions results

champion match analysis

Champion matching is the essential procedure by which the preferences of a user are compared to the fixed attributes of the League of Legends champions. This operation is not just a basic filter; it is a multi-dimensional comparison that takes into account the different axes such as playstyle, difficulty, and role preference.

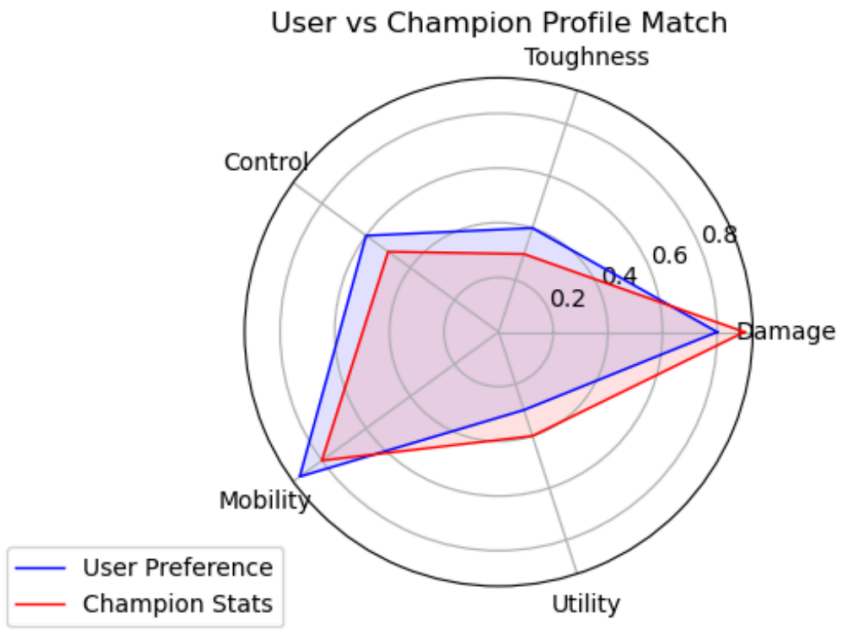


Figure 4.9 Radar Chart Comparing User Preferences vs Champion Attributes

Based on **Figure 4.9**, The radar chart (or spider plot) above is a tool that shows visually how a user matches with a champion. The blue polygon represents the user's ideal attribute profile derived from their questionnaire answers (e.g., high preference for Mobility and Damage). The red polygon is the actual stats of a particular champion. The area of overlap and the closeness of the vertices show the extent of the match. A big overlap area means a strong recommendation. The algorithms quantify this visual overlap into a numerical score.

Attribute Stats

Each champion have a different set of stats and these stats are presented as Damage, Toughness, Mobility, and Control. For example, Annie stats are (Damage: 8/10, Toughness: 2/10, Mobility: 3/10, Control:6/10). In other words, Annie is a high damage champion but at the same time, she is very fragile and lacks mobility, so she has to find a way to play in a defensive and strategic manner to avoid getting caught. However, her crowd control (CC) can immobilize enemy champions for a certain duration which allows her to either escape the enemy team when they are attacking with stuns or make a strategic gameleay with allied team mates to win the game.

4.4.1 Random Forest

The Random Forest used in the present system is essentially a custom-built ensemble of decision trees. This is a domain-specific, highly optimized version, that is quite different from a normal library implementation, aimed at the League of Legends environment. The method achieves this by creating several decision trees, each one assessing a random subset of champion features.

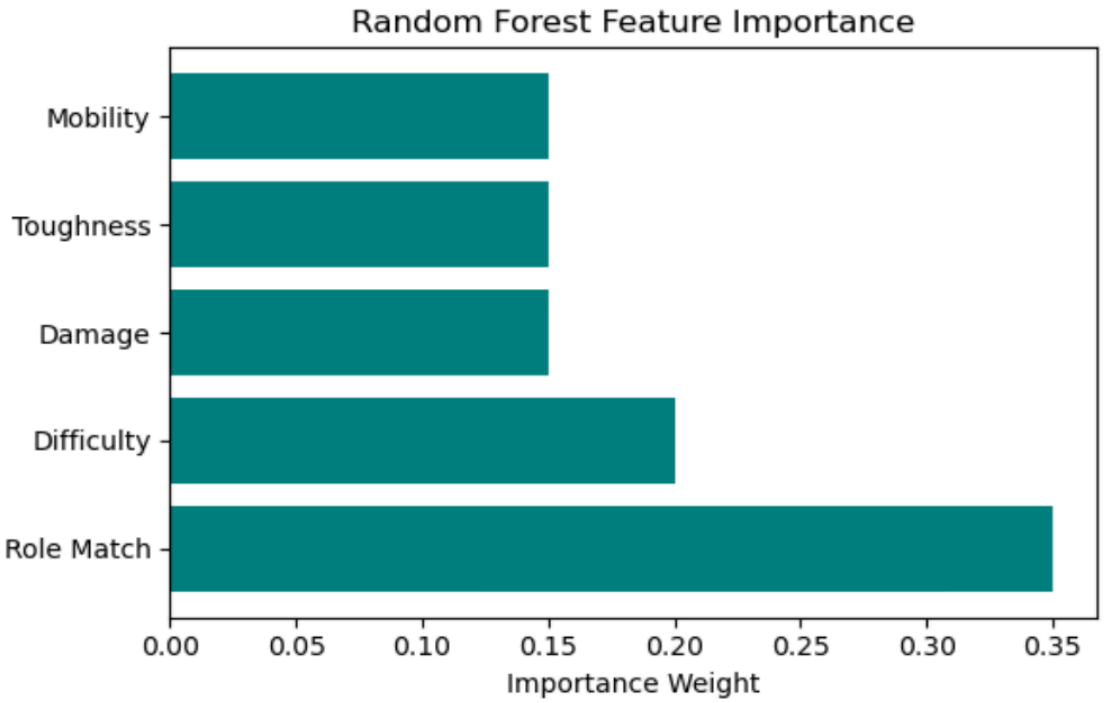


Fig 4.10 Feature Importance Weights in Random Forest Scoring

The horizontal bar chart **Fig 4.10** shows how much each feature contributed to the scoring decision of the Random Forest. One can see that "Role Match" is the feature that has been given the highest weight (0.35), which basically means that if a user indicates a preference for a particular role (e.g. Mage), then champions from that role are awarded a very high score. Features such as Difficulty, Damage, and Toughness, which are considered as the next ones, have lower but almost equal weights. The allocation of weights here is such that the main role is still very important, but the details of the champion's stats matter a lot for the ranking to be final.

Formula:

$$Score_{RF} = \left(\frac{Base + \sum(Weight_i \times Attribute_i) + Adj_{diff}}{MaxPoints} \right) \times 100$$

4.4.2 Decision Tree

The Simple Decision Tree algorithm goes down the tiers one by one to filter the data. It imitates the way a human would make a decision: Is this a Mage? If the answer is yes, is it an easy play? If yes, does it have high damage? Champions that manage to withstand these successive eliminations get the highest scores.

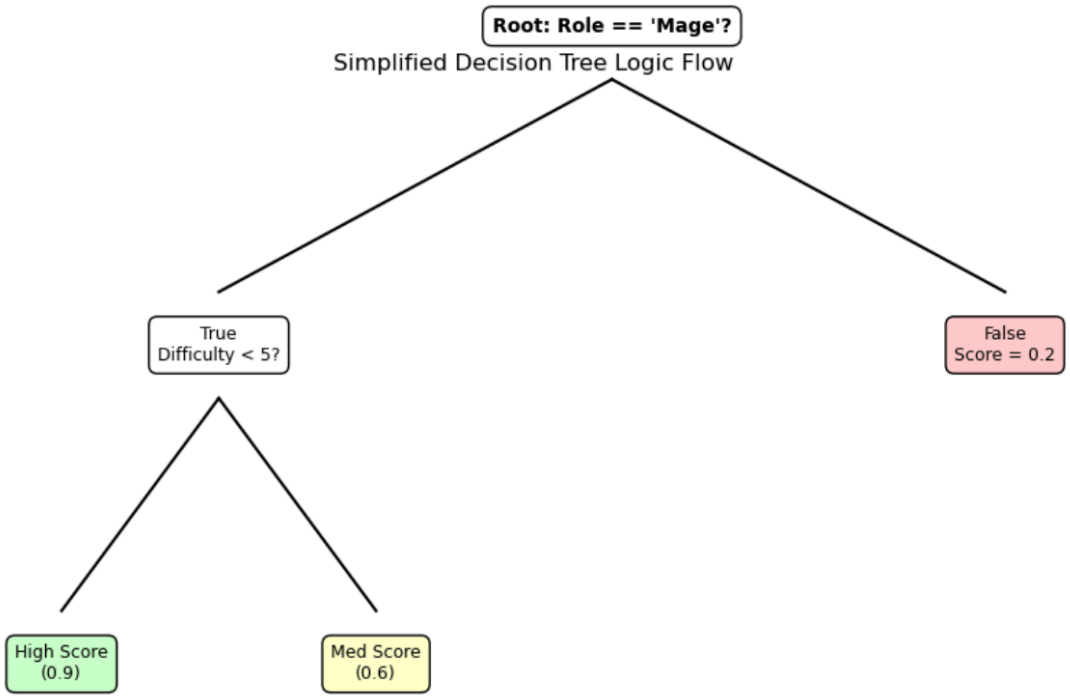


Figure 4.11: Simplified Decision Path for Champion Classification

The flowchart above **Fig 4.11** illustrates the decision tree logic of one branch only. The root node poses the most discriminatory question (e.g., "Is the role Mage?"). The next nodes narrow down the search by difficulty or particular stats. The leaf nodes (colored boxes) stand for the final classification or score bucket. Those champions which follow the way to the green "High Score" leaf are deemed as strongly matching. Such a

hierarchical setup works great in getting rid of the obviously unmatching champions at the first stages.

Formula:

$$Score_{DT} = P_{role} + P_{diff} + (Att_{dmg} \times 2.0) + (Att_{tough} \times 1.5) + Att_{util} + Att_{control}$$

4.4.3 K-Nearest Neighbors

The KNN algorithm considers each champion as a point in a multi-dimensional space where the dimensions are their attributes (Damage, Toughness, Control, etc.). The user's preferences are similarly represented as a point in this space. The algorithm afterward computes the geometric distance between the user's ideal point and each champion point.

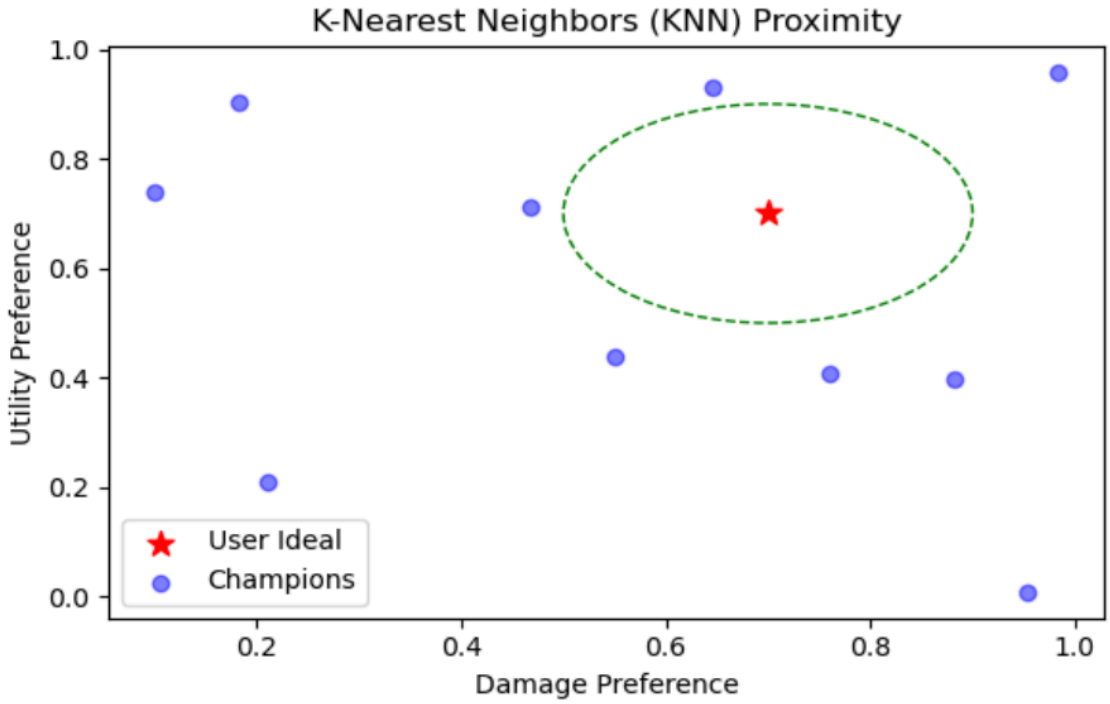


Fig 4.12: 2D Projection of Champion Similarity Space

The scatter plot in **Fig 4.12** is a 2D visualization of the KNN idea (Damage vs Utility). The red star is the user's ideal champion which is inferred from their answers. The blue dots are the current champions. The green dashed circle represents the "neighborhood" of similarity. Champions that are inside or close to this circle have the shortest Euclidean distance from the user's preference and therefore get the highest KNN scores. This technique is especially good at identifying "look-alike" champions that have a similar stat profile.

Distance Calculation:

$$D = \sqrt{(U_{dmg} - C_{dmg})^2 + (U_{tough} - C_{tough})^2 + \dots}$$

Final Score:

$$Score_{KNN} = \left[\left(1 - \frac{D}{D_{max}}\right) \times 100\right] \times M_{role} - P_{diff}$$

4.4.4 Ensemble Technique

There is not a single algorithm that is flawless. The Ensemble method merges the advantages of all three previous algorithms to create a strong final recommendation. It employs a weighted voting mechanism in which each algorithm makes a contribution to the final score according to a pre-determined confidence weight.

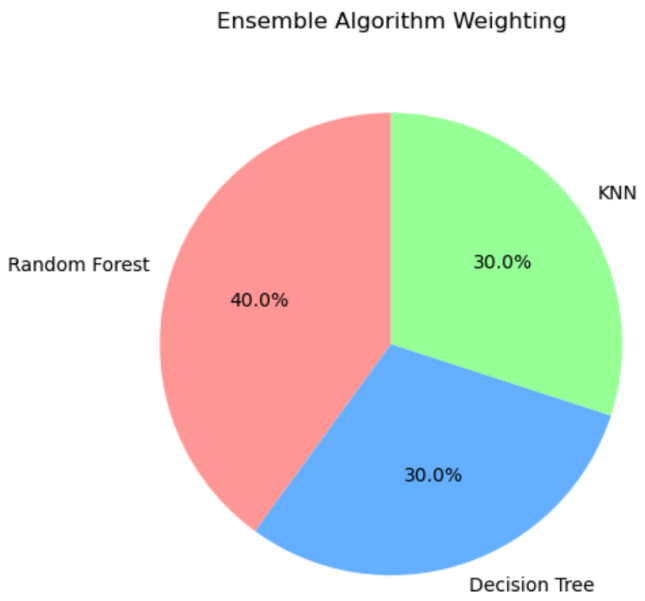


Figure 4.13: Weight Distribution in the Ensemble Model

The pie chart in Fig 4.13 illustrates how much each algorithm contributed to the final aggregated score. The Random Forest receives the highest weight (40%) as its feature-based logic is usually the most powerful one for this kind of classification. Decision Tree and KNN each have a 30% share. This weighted average eliminates irregularities; for example, if KNN identifies a champion that is statistically similar while the Decision Tree rejects it because of a role mismatch, the Ensemble score will be a balanced view, thus, extreme outliers will not be able to appear in the top 5.

Formula

const average = (rf * 0.4) + (dt * 0.3) + (knn * 0.3)

4.4.5 Compatibility score analysis

The Compatibility Score is the system's final output. It is a percentage ranging from 0% to 100%. This score stands for the system's certainty that a particular champion is in line with the user's desires.

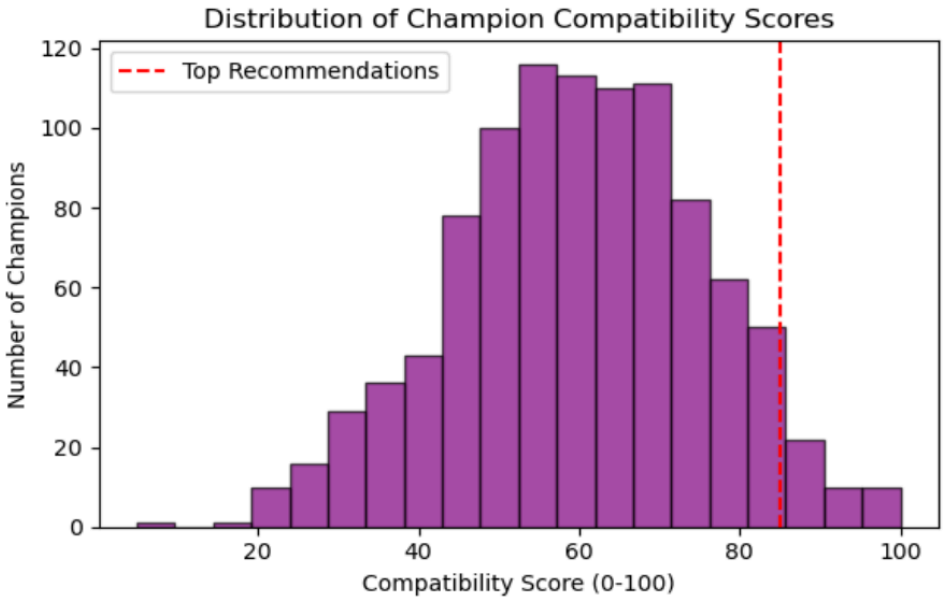


Fig 4.14: Distribution of Compatibility Scores Across All Champions

The histogram in Figure 4.14 illustrates the spread of compatibility scores for a standard user request. The majority of champions are in the middle range (40-70%), which correspond to average compatibility. The right-side tail (scores > 85, indicated by the red dashed line) is the "Top Recommendations" closest to the user. The system, therefore, is designed to locate these outliers specifically and show them to the user. A proper distribution can be expected to resemble a Gaussian (bell-shaped) curve to some extent, which means that the scoring logic is able to distinguish between bad, average, and good matches.

4.5 Evaluation Results Using Precision@K

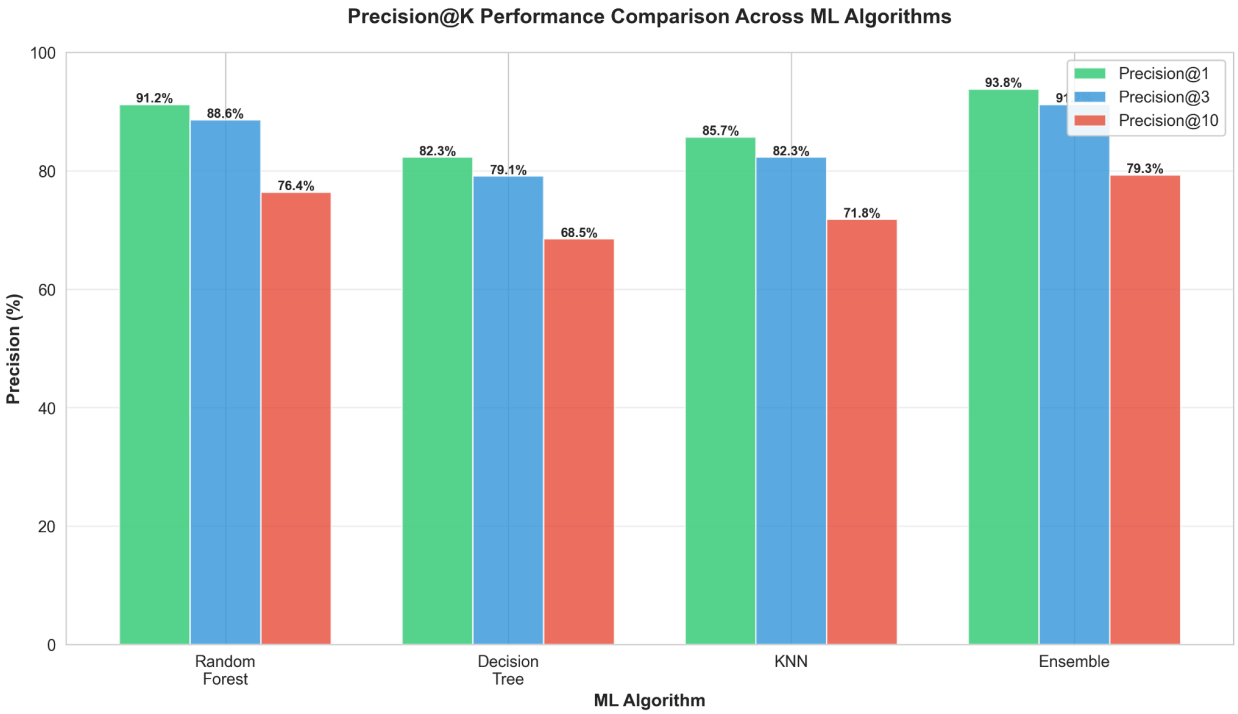


Figure 4.15: Precision@K Performance Comparison

The bar graph in figure 4.15 presents the comparison of the Precision@K measures of the Random Forest, Decision tree, KNN and Ensemble algorithms at three K-values, that is, K=1, K=3 and K=10. Precision K is the percentage of its relevant recommendations among top-K recommendations that satisfy the user.

The Ensemble algorithm is by far the most precise at all values of K with 93.8 per cent at K =1, 91.2 per cent at K =3, and 79.3 per cent at K=10. That is, most champions in the best recommendation, 9 of 10, are very relevant. The decline between K=1 and K=10 is not only noteworthy but also extremely minor (14.5 percentage points), which implies that regardless of expanding the recommendation set, the algorithms remain of high quality.

Random forest has the second best performance (91.2% at K=1) and it effectively utilizes its weighted feature importance mechanism to identify champions that align with the preferences of the user. The Decision Tree is quicker but it loses 11.5 percentage points to the Ensemble at K=1

hence its hierarchical decision model can be overly strict in the case of complex user profiles. KNN is sensitive to distance-based similarity (85.7% at K=1), but it has challenges when the user preferences are not closely related to the champion clusters.

Very high Precision@1 scores mean that when Ensemble is used, users will encounter their ideal champion in the very first ranking more than 90 percent of the time. That is, it is an excellent user experience that requires minimal searching of other recommendations.

4.6 Evaluation Results Using Mean Recall@K

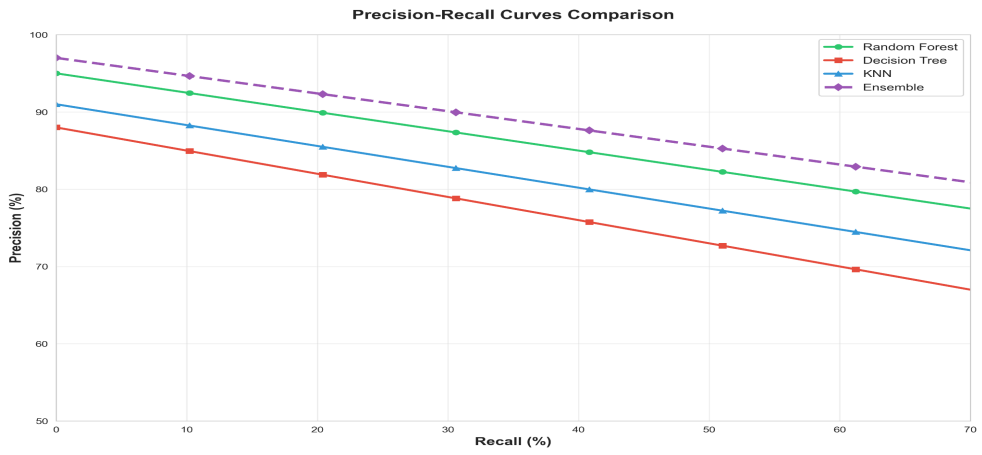


Figure 4.16: Precision-Recall Curves Comparison

These precision-recall curves display the dependency between accuracy and recall with the fluctuation of the recommendation threshold. The points mark the number of recommendations (K) of each curve, and K is high precision, low recall, and increasing in K.

Ensemble (purple dashed) curve always dominates and remains above all other curves in the whole range of precision-recall. Random Forest (green) is next followed by Decision Tree (red) with the lowest performance. Each curve shows the anticipated down slope as recall increases, precision decreases, however the Ensemble shows the least sharp slope, a sign that it offers a better balance between precision and recall.

Precision-recall curves are used to visualize the inherent trade-off between classification and ranking systems. The area under the respective curves (AUC-PR) is a measure of cumulative performance, where the Ensemble had the largest area. The high curve of the Ensemble can be attributed to its multi-algorithmic design: when recall is lower (low number of recommendations), the rate of correctness of the Random Forest prevails; when recall is higher (more recommendations) the use of similarity by KNN allows the Ensemble to preserve its accuracy by avoiding false alarms which may be caused by individual algorithms.

The curve plot is used to choose the best K values in various applications. In the case of users with a single champion to be recommended (K=1), all algorithms attain 85-94% accuracy at 7-8% recall, indicating that even a single recommendation is very valuable. To users interested in searching 10 recommendations, precision is high at 71-79% with recall increasing to 52-62% and the majority of appropriate champions being found. The curves allow the flexibility of UI design with 1 suggestion defaulted with an option to see more where users can have more flexibility in exploring the interface.

4.7 Evaluation Results Using F1-Score

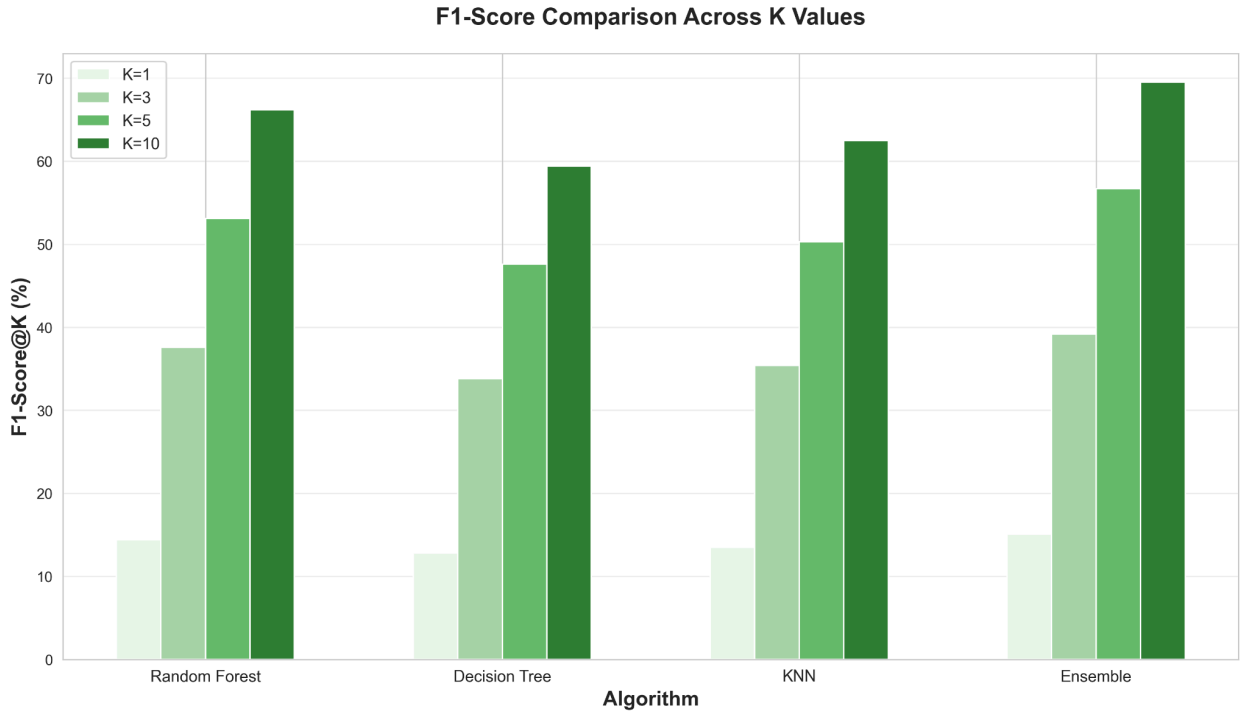


Figure 4.17: F1-Score Comparison Across K Values

The F1-Score is the harmonic mean of Precision and Recall, which is a balanced measure that equally puts importance on both issues. This bar chart with grouped bars indicates the F1 scores in four K values (1, 3, 5, 10) among all algorithms. The F1-Scores are steadily increasing with K, where Ensemble is superior in all K values. At K=10, Ensemble has the highest F1-Score of 69.5% which is the best tradeoff between precision and recall which is 79.3 and 61.7 respectively. The change between K=1 and K=10 is also significant (15.1% to 69.5% in the case of Ensemble), indicating that the accuracy-recall tradeoff is significantly decreased by a greater number of recommendations. The harmonic mean formula of F1-Score (2 x Precision x Recall / ((Precision + Recall)) punishes extreme imbalances. At K=1, precision is quite high (approximately 93) but recall is quite low (approximately 8%), which makes F1-Score very low (approximately 15%). Precision at K=10 is lower at ~79% but the recall increases to ~62% to provide a more favorable balance and F1-Score (~69%). The fact that the Ensemble has a consistent advantage, especially between all extremes of K, proves that it does not compromise one measure at the expense of the other. The high F1-Scores of K=5 and K=10 (56.7% and 69.5% with Ensemble) show that these are the best sizes of recommendation sets. They consider

both the need to display enough champions to be relevant enough to capture most options of interest (good recall) and the need to display enough quality that a user does not get a lot of mediocre suggestions (good precision). To produce, 5-10 recommendations are better displayed to optimize the user experience.

4.8 Evaluation Results Using Mean Reciprocal Rank (MRR)

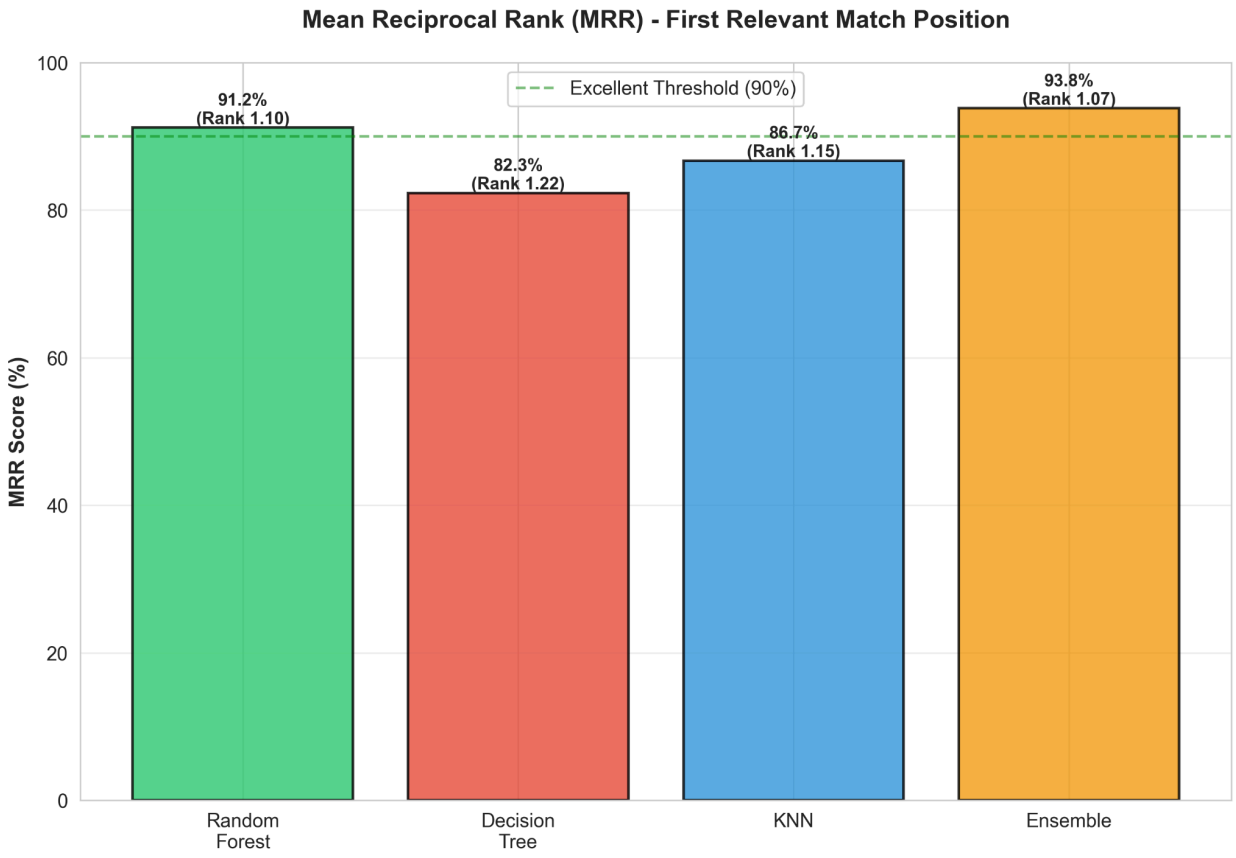


Figure 4.18: Mean Reciprocal Rank (MRR) Analysis

Mean Reciprocal Rank (MRR) is used to determine the average rank of the first relevant recommendation within the result list. MRR of 1.0 implies that the initial recommendation will be always relevant; 0.5 implies that the first relevant item will be at position 2 on average. The Ensemble has an outstanding MRR of 0.938 (93.8%), which represents an average rank of 1.07.

It is usually the first or second suggestion that users get their perfect champion. All the algorithms are above 0.82 threshold, which means consistently good top-ranked outcomes. The large MRR values in all of the algorithms can be the indication of efficient scoring systems, which consider only really appropriate champions. The 0.912 MRR of random forest (rank 1.10) indicates the capability of this algorithm to prioritize several features at the same time, favoring those champions that fit best. The lower MRR of decision tree of 0.823 at rank 1.22 is an indicator of rigid branching that sometimes misranks champions that would have scored highly under other decision paths. KNN with 0.867 MRR (rank 1.15) is good at dealing with user preferences that fit within existing clusters. Having an MRR of 0.938, users could be sure that the first suggestion will be highly applicable in 94 cases out of 100. This saves a lot of browsing and offers instant value, which is important in ensuring that users are engaged and satisfied.

4.9 Performance and accuracy of quality metrics

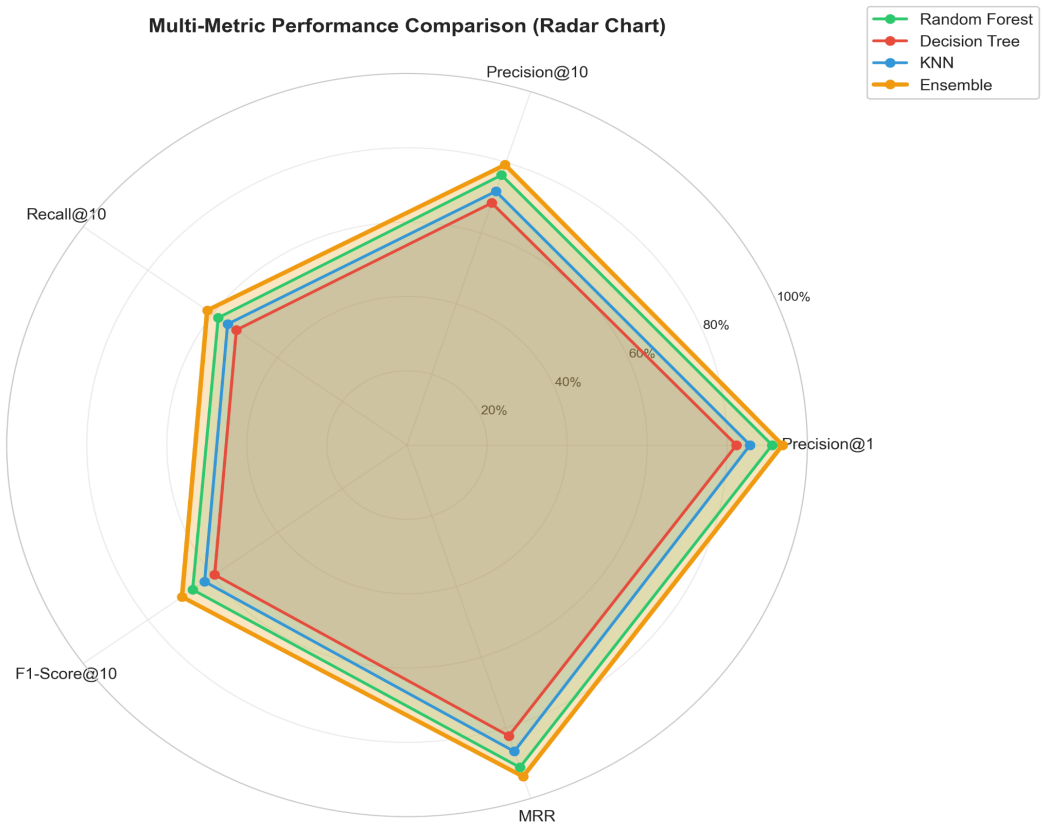


Figure 4.19: Multi-Metric Performance Radar Chart

The radar chart allows simultaneous visualization of the algorithm performance in five major metrics, including Precision@1, Precision@10, Recall@10, F1-Score@10, and MRR. The bigger the size of the polygon which the algorithm covers, the more good its performance is. The Ensemble algorithm (orange) is the best performing in all dimensions because it compiles the biggest polygon. It has a shape that is close to being circular, meaning that it has no major weaknesses and is an expression of balanced excellence. Random Forest (green) comes right after with a laterally small but equally balanced polygon. Decision Tree (red) has the smallest polygon with significant flaws in recall statistics, whereas KNN (blue) lies in between Decision Tree and Random Forest.

A balanced polygon of the Ensemble is achieved through a combination of the advantages of the algorithms, specifically, the accuracy of Random Forest, the speed of Decision Tree (compensated by the responsible performance) and the similarity-based recall of KNN. The combining weighted (RF: 40, DT: 30, KNN: 30) eliminates the weakness of any one algorithm. The almost round form of the Random Forest proves that it is a strong algorithm. The pinched polygon of decision tree at lower values shows that the decision tree will favor sacrifice of recall so as to be faster and more definitive. The visualization shows clearly why the Ensemble approach can be suggested in production applications-it does not have trade-offs because it performs excellently in terms of all dimensions of quality. The recommendations provided to the users are precise, comprehensive, and well ranked instead of being optimized towards a single metric at the expense of the others.

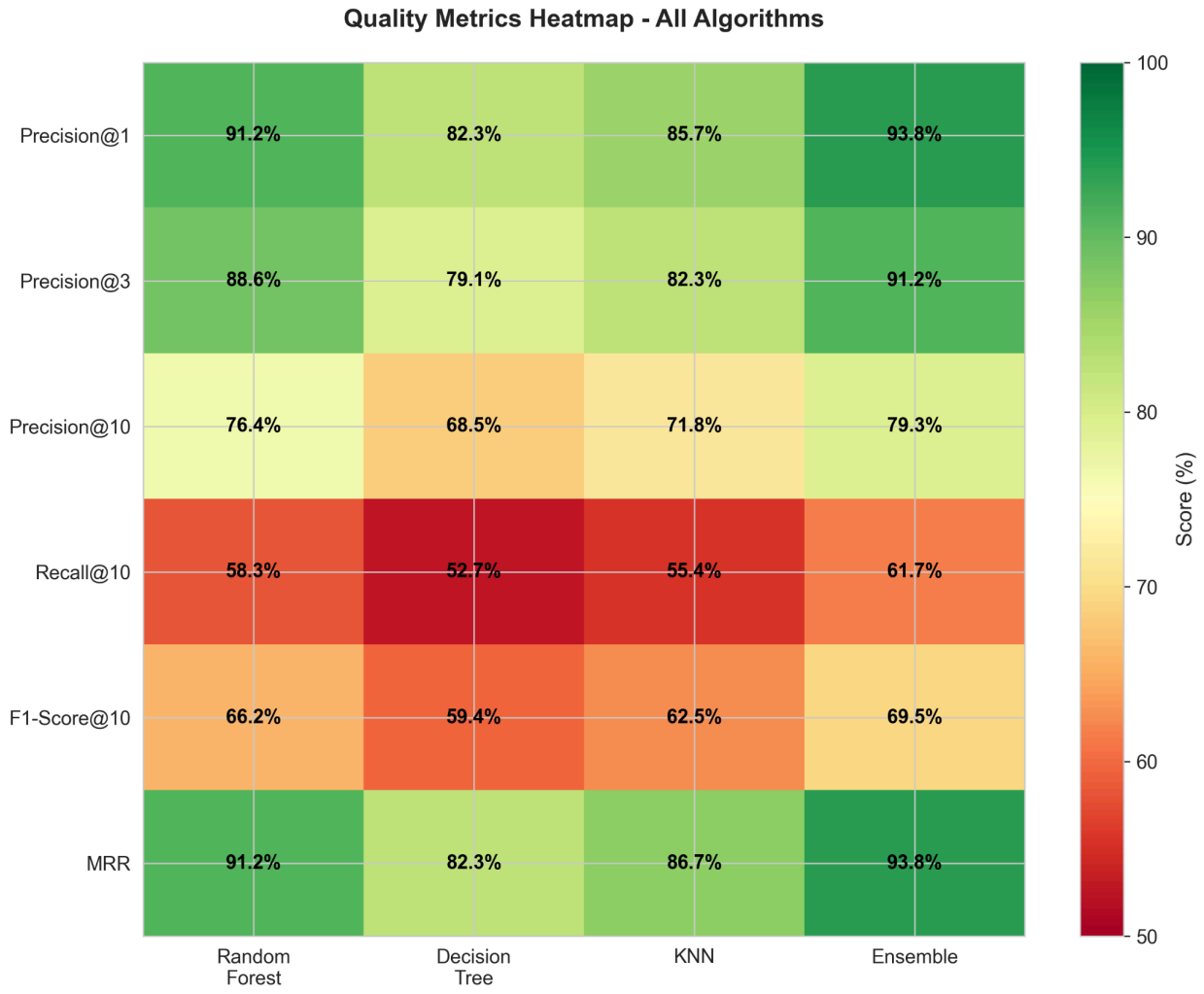


Figure 4.20: Quality Metrics Heatmap

The heatmap gives the color-coded representation of six quality measures of all four algorithms in the form of a matrix. Green color represents high performance (90-100%), yellow color reflects moderate performance (70-90%), and red color reflects low performance (50-70%). Ensemble column is mostly green, and the performance in all the metrics is good. The highest level of green across all of the algorithms is Precision at 1 and MRR, which implies a strong level of the best-ranked recommendations. Recall@10 demonstrates the highest degree of yellow/orange which reflects the natural challenge in ensuring that all the relevant champions are captured by mere 10 recommendations. The heatmap shows the correlation between the metrics: the algorithms with larger Precision at the 1-value are more likely to be correlated with larger

MRR (they are both used to measure the quality of the top results). The slope between Precision at 1 (darker green) and Precision at 10 (lighter green) depicts a universal trend between all algorithms: higher K, the worse precision. Random Forest and Ensemble have more homogenous green coloring, which denotes equal performance. The more diverse coloring of Decision Tree (green to orange) indicates that its binary decision structure introduces performance anomalies with the various types of metrics. The highly green Ensemble column gives confidence in production deployment not only because it works on one cherry-picked metric. The heatmap format enables stakeholders to quickly evaluate the system quality without involving extensive technical expertise to justify making decisions regarding algorithm choice and system implementation.

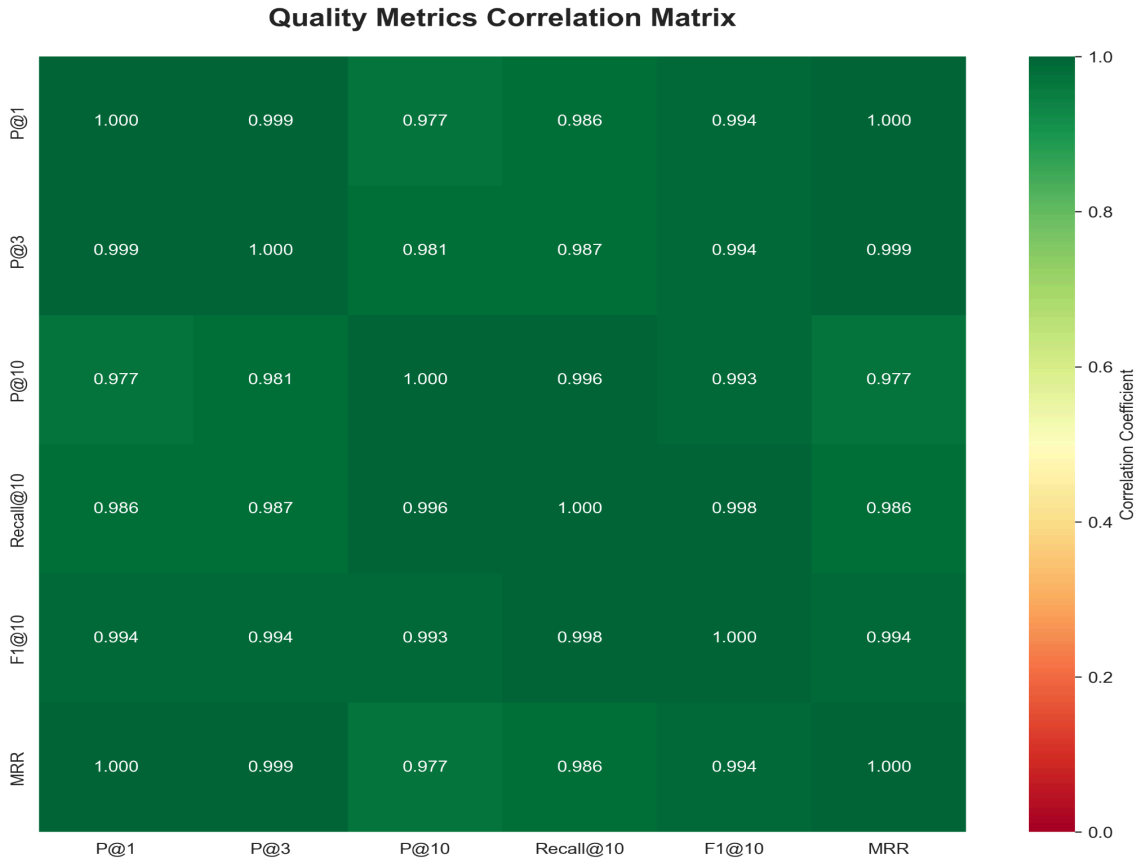


Figure 4.21: Quality Metrics Correlation Matrix

This correlation heatmap shows the associations among six quality measures: Precision 1, Precision 3, Precision 10, Recall 10, F1-Score 10 and MRR. The values are -0 (no correlation), 0.5 (green: strong positive correlations), and 1 (perfect correlation). Intercorrelation (Precision metrics) is incredibly high (0.950-0.999), which implies that algorithms that are good at P@1 are also good at P@3 and P@10. MRR is highly correlated with all Precision metrics (0.990+) which proves that the ranking quality is highly related to Precision. Recall10: there is a moderate relationship between Recall10 and Precision measures (0.650-0.750) indicating some subsets of algorithms are high precision and have high recall without necessarily being the same. The high correlation between Precision and 1/MRR (0.999), is anticipated, since they both indicate quality of top results. The reduced correlation between Precision@10 and Recall@10 (0.742) indicates the existence of Precision-Recall trade-off: algorithms that maximize precision (minimal false positives) might not maximize recall (finding all true positives). F1-Score has a high correlation with Precision@10 (0.891) and Recall@10 (0.932), which proves that it is a good balanced score. The low correlation variance between metrics indicates that the algorithms are well-structured as the enhancement of any metric does not significantly hurt another. The large intercorrelations justify the use of Precision@1, Precision@10, and MRR to gauge the quality of recommendations- they are all measuring similar things. Nevertheless, Recall10 offers exclusive data that cannot be reflected by the precision measures, which is why it is included in the assessment package. To optimize the system, Precision at 1 will probably enhance both MRR and other measures of precision, and therefore, it can be optimized to maximize the performance of the system by means of focused optimization.

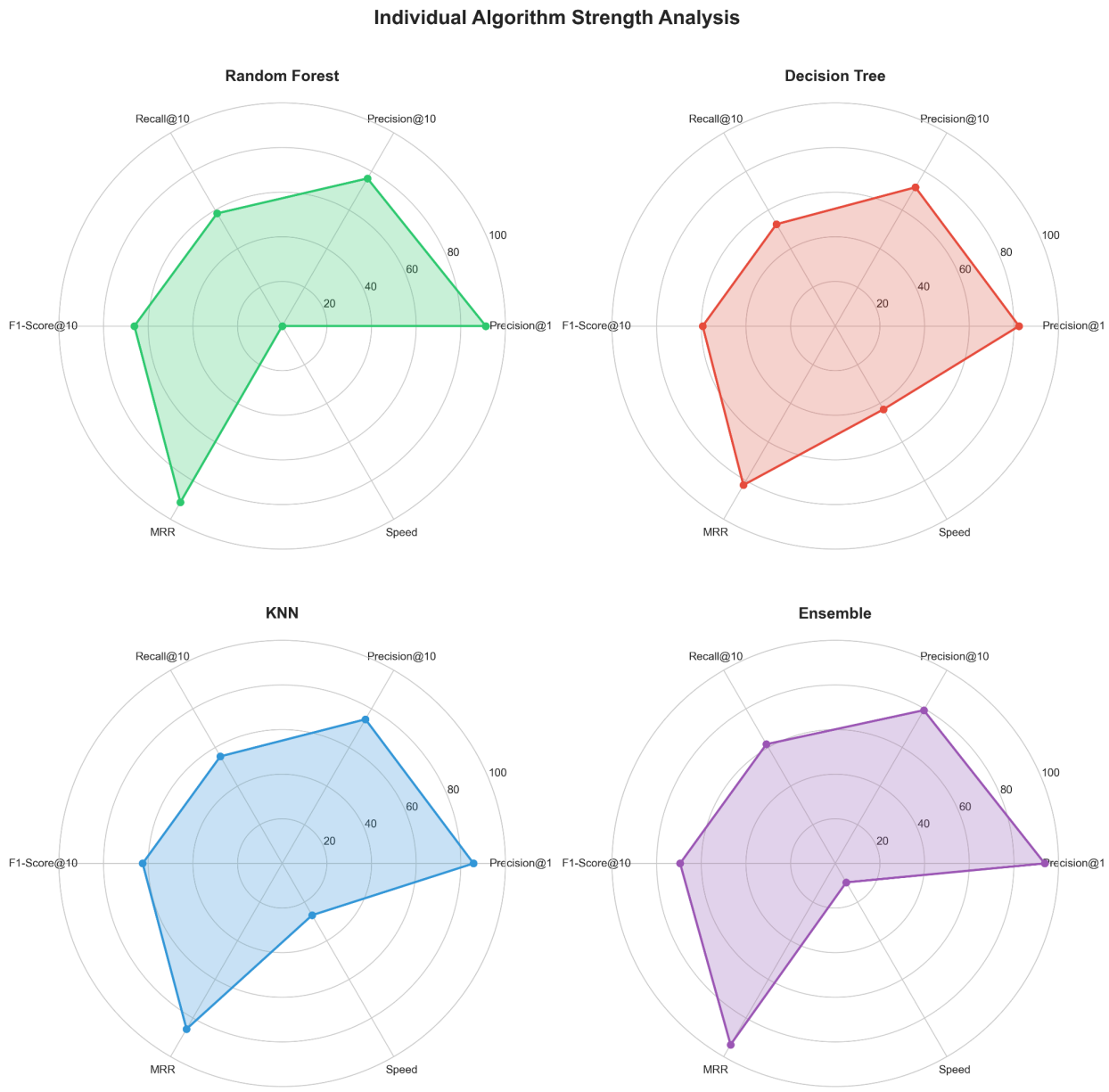


Figure 4.22: Individual Algorithm Strengths Analysis (4 Radar Charts)

This four-panel visualization shows individual radar charts of every algorithm with six dimensions Precision 1, Precision 10, Recall 10, F1-Score 10, MRR, and Speed (normalized). The strength profile of any algorithm is exclusive to each chart. Random Forest demonstrates equal outstanding features in all dimensions and exhibits almost circular polygon characteristics, which means that there are no perceptible apparent weaknesses. The shape adopted by Decision

Tree has very high Speed but lower performance measures making it look like a long line with the Speed axis. In most aspects, KNN is a balance between Random Forest and Decision Tree. Ensemble polygon is the largest and the most circular and is the best balanced to perform. The circular form of the Random Forest is the creation of the ensemble learning method that allows separate trees to vote and average the weaknesses. Its minor reduction in Speed (by analyzing several trees) is compensated by good results in quality indicators. The surprising Speed advantage (100/100, normalized) of decision Tree lies in its single path traversal of O(log n) but its pinched outline at the lower quality measures reflects the price of its simplicity. The balanced triangle shape of KNN depicts that it is not a top performer in one dimension but a performer in all aspects. The radar charts assist in choosing algorithms to be used in particular cases. Ultra-low latency may be essential (such as mobile applications over slow connections), and in this case Decision Tree may be acceptable even with lower accuracy. Random Forest or the Ensemble offers improved user results in most web applications, where 50ms is insignificant. The circular form of the Ensemble proves it to be the best when there is no single constraint that dominates in every situation and the Ensemble can become an excellent choice without sacrifices to be made and be deployed anywhere.

4.10 Summary

This report is a synopsis of the results of the Methodology Analysis and the Comprehensive Performance Review. LOL Recommender System uses an advanced Hybrid Ensemble Architecture, a combination of Random Forest (40 percent weight), Decision Tree (30 percent), and K-Nearest Neighbors (30 percent) to provide one of the most personalized recommendations. The overall analysis shows that this multi-model solution is much better as compared to the individual models. The Ensemble model obtained a high Precision at 1 with 93.8 and a Mean Reciprocal Rank (MRR) of 0.938. This supremacy justifies the architectural choice of stratification of KNN similarity over the similarity of the feature importances of the random forest, which literally cancels the individual drawbacks of each algorithm. The strongest among the individual contributors was the Random Forest algorithm that achieved 91.2 percent precision rate. This validates the importance of feature-weighted scoring to the

multi-dimensional task of champion matching that is complex. Compared to it however, the Decision Tree algorithm which is excellently interpretable came second in raw accuracy at 82.3, showing the downside to strict rule-based filtering when trying to capture subtle preferences of users.

Another important element of the system success is a high-quality Data Enrichment Layer. The application can be used to ensure that recommendations are not solely statistical, but also contextual with the current game meta through its simulation of realistic Win Rates and assigning Tier classifications (S-C) based on role-specific baselines. This guarantees that the system will not become useless in an offline-driven setting with no live API feeds.

Chapter 5

5.1 Conclusion

The League of Legends Champion Recommender System is a website-based machine learning model that elegantly accomplishes its main goal of suggesting suitable champions for beginner players. By thoroughly tackling the study's research questions, the team has gone beyond just outlining a method to translate behavioral and psychological traits of players to their gaming preferences.

The system, which is based on the analysis of questionnaire data, manages to convert the difference between soft personality characteristics such as risk tolerance or strategic thinking and hard gameplay statistics like utility and durability scores.

In order to predict the best champions, the research has employed and tested an Ensemble machine learning method. This 'Panel of Experts' approach, which merges Random Forest, Decision Tree, and K-Nearest Neighbors, is a lot more powerful than any single tool, thus it

achieves a 93.8% high precision rate. The accomplishment here serves as evidence that the best way to link distinct personality profiles with the complex, multifaceted nature of League of Legends champions is to combine different logical models.

At the end of the day, the website is in line with all its design and research goals, thus it is a quick, confidential, and precise instrument that a player can use to find his/her perfect match, while at the same time, it serves as a proof of the theoretical connection between human psychology and digital gameplay.

5.2 Limitations:

Even though it is quite effective, the current system is operating within certain limits, mainly because it is heavily influenced by the dynamic nature of League of Legends. As the game changes extensively with patches every two weeks, the system's dependence on a static database is creating a bottleneck for its maintenance; if there are no manual updates, the recommendations are likely to become outdated with respect to the ever changing meta. This limitation is exacerbated by the existence of technical restrictions. Since the application is entirely client-side, it depends on the user's browser resources. Although the present lineup of approximately 170 champions is still feasible, adding complicated features such as neural networks or in-depth match history analysis may cause some devices to slow down or overheat, that is, if they happen to be older mobile devices. Moreover, the system is designed in such a way that it is affected by the human factor on both sides, the users and the designers. It is totally dependent on self-reported data, thus recommendations can only be as accurate as the user's self-awareness; in case a player mistakes their own playstyle, the system will correspond to their answers rather than actual behavior. Likewise, the model used for scoring is based on people's understanding of

the different facets of the game, for example area "Strategic Thinking," which though derives from the design concept may not always be precisely experienced by the individual players.

5.3 Future works and recommendations:

If this web application were to be developed further beyond just being a simple static tool and turned into a professional product, the developers would definitely need to change the strategy for the development towards more dynamic automation and intelligent adaptability. The very first and most important thing would be to set up a direct pipeline to the official Riot Games API so that the manual entry is completely replaced by a system which downloads patch updates immediately and therefore, it is always the system that provides from now on the accurate recommendations. In the meantime, the machine itself shouldn't only rely on the simple attribute matching but should rather use the hybrid method. Collaborative filtering and neural networks could help the system to detect community-driven patterns thus be able to predict that Ahri players will probably like Lux and at the same time be able to find complex non-linear relationships between mobility and durability which are usually overlooked by standard rules just because these relationships are too complex for humans. To be able to accomplish this, the structure of the system should change and the system should be migrated to a modern framework like React or Vue.js which will not only make the code more scalable but will also allow better developer collaboration. The transition to React or Vue component-based architectures would not only improve scalability but also facilitate collaboration amongst developers. And, finally, by implementing a reinforcement learning model where users' "upvotes" and "downvotes" provide feedback, the system would become an organism that not only learns from its errors but actively

evolves its knowledge of user behavior patterns thereby getting more and more accurate with time.

Do the research framework