# Machine Learning Research Document

League of Legends Match Outcome Prediction

## 1. Introduction

The goal of this project is to develop a machine learning model that predicts the probability of a team winning a League of Legends ranked match. The model will use both pre-game data (champion selections, roles) and early-game data (lane performance and objectives up to 10 minutes). The data will be collected directly from the Riot Games API, providing real, up-to-date match information rather than pre-cleaned datasets. The outcome will be a regression-based AI that outputs a continuous win probability between 0 and 1, which corresponds to 0%–100% chance of winning.

### 2. Algorithm Selection and Alternatives

### Chosen Approach: Regression-Based Modeling

### The main goal of this project is to predict the probability of a team winning, not just whether they win or lose. Therefore, the problem is best approached as a regression task, where the output is a continuous value between 0 and 1 (representing win probability).

### The primary algorithm chosen is the Random Forest Regressor, with Logistic Regression used as a baseline.

### Logistic Regression (Baseline Model)

### How it works: Logistic Regression predicts the probability that an event happens (e.g., the blue team wins) by fitting a curve to the data. It is similar to Linear Regression but uses a sigmoid function to keep predictions between 0 and 1.

### Why it fits: It’s simple, interpretable, and gives direct probability outputs. It’s ideal for testing whether the data structure supports accurate prediction before moving to more complex models.

### Limitations:

### Can only capture linear relationships between features and outcomes.

### Might underperform on complex, non-linear game interactions.

### Random Forest Regressor (Main Model)

### How it works: A Random Forest combines many individual decision trees, where each tree learns different parts of the data. The final prediction is the average of all trees.

### Why it fits:

### Learns non-linear and complex relationships between game statistics and outcomes.

### Handles both numeric and categorical data types well.

### Naturally provides feature importance scores, showing which metrics (gold diff, objectives, etc.) matter most.

### Limitations:

### Harder to interpret than simple models.

### Slower training on very large datasets.

### Gradient Boosting ( XGBoost for Future Iteration)

### How it works: Builds trees sequentially, where each new tree corrects the mistakes of the previous ones. It’s extremely powerful for structured data.

### Why it fits:

### Can capture very complex feature interactions (ex. objective combinations and lane dominance).

### Often achieves top accuracy in predictive tasks.

### Limitations:

### Requires more fine-tuning (hyperparameters).

### Slower to train compared to Random Forest.

### Reason for later use: It’s ideal for a second or third iteration, after simpler models prove the concept.

### Algorithms Not Selected

| Algorithm | Reason Not Selected |
| --- | --- |
| K-Nearest Neighbors (KNN) | Too slow for large datasets, sensitive to scaling, and performs poorly with many numerical + categorical features. |
| Linear Regression | Not suitable for probability outputs (can predict values outside 0–1 range). |
| Neural Networks | Overly complex for this project’s scale, requires large datasets and significant tuning. |
| Support Vector Machines (SVM) | Performs well on small datasets but scales poorly with thousands of samples and many features. |

### Planned Modeling Approach

1. **Iteration 1:** Logistic Regression using the most important features (gold difference, first objectives).  
2. **Iteration 2:** Random Forest Regressor with extended features (lane stats, objectives, roles).  
3. **Iteration 3:** Potential Gradient Boosting (XGBoost) if dataset size permits.

## 3. Dataset Description

The dataset will be created using the **Riot Games Match-v5** and **Timeline-v5** **APIs**. Each record represents one ranked match, with team-level and lane-level features collected at the 10-minute mark.

**Dataset Structure:**  
- **Rows:** One per match (from the blue team perspective).  
- **Columns:** Pre-game champion and role data, early-game lane stats, objective control, and outcome.

## 4. Features and Their Usefulness

### A. Match Metadata

- Match ID: Unique match reference, used for identification. (Not used in modeling)

- Queue ID: Filters for ranked matches (ensures data consistency). (Not used directly in prediction)

- Game Duration: Used to filter out remakes; shorter than 10 minutes are excluded.

- Usefulness: Supports dataset integrity and ensures relevant samples for training.

### B. Champion & Role Data

- Champion IDs per Lane (Blue/Red): Identifies which champions are played in each role (Top, Jungle, Mid, ADC, Support).

- Roles: Confirms lane assignment (important for fair matchup comparison).

- Usefulness: Champions have distinct power levels and synergies. Champion choices define early strengths and weaknesses.

### C. Lane Metrics at 10 Minutes

- Gold Difference per Lane: Indicates economic advantage. One of the strongest predictors of win probability.

- CS (Creep Score) Difference per Lane: Reflects farming and lane control.

- XP Difference per Lane: Shows who reaches power spikes earlier.

- K/D/A per Lane: Measures lane dominance through kills and assists vs. deaths.

- Usefulness: These metrics represent early-game momentum. Teams with consistent lane advantages at 10 minutes typically have higher win probabilities.

### D. Team Objectives at 10 Minutes

- First Blood: Indicates early aggression and tempo control.

- First Tower: Provides gold, map control, and vision — often the most impactful early objective.

- First Dragon (with type): Grants permanent buffs and long-term scaling advantages.

- First Rift Herald: Enables tower pressure, leading to map dominance.

- Tower/Dragon/Herald Kills (Totals): Reflects team-wide coordination and macro play.

- Usefulness: Objectives reflect team synergy and map control. Empirically, First Tower and First Dragon are highly correlated with overall win rate.

### E. Target Variables

- Match Outcome: Binary label (1 = Blue win, 0 = Blue loss).

- Win Probability: Continuous value predicted by the regression model.

- Usefulness: Enables training for both classification (Win/Loss) and regression (Win Probability).

## 5. Model Evaluation

**Train/Test Split:** 80/20 for performance validation.  
**Metrics:**  
- R² Score and Mean Absolute Error (MAE) for regression.  
- Accuracy when logistic regression baseline is used.  
- Feature importance analysis to identify the most significant predictors.

## 6. Iteration Plan

**Iteration 1**: Use core features (GoldDiff10, FirstTower, FirstDragon, MatchOutcome).  
**Iteration 2**: Add lane-level CS/XP/KDA differences.  
**Iteration 3**: Add champion/role features and team totals.  
  
This iterative approach keeps the first version simple and ensures improvement in later stages based on feedback.

## 7. Expected Outcome

The expected result is a functional machine learning model that can predict the probability of a team winning a match after 10 minutes of gameplay. Beyond that, it will offer insights into the relative importance of features — identifying which in-game factors have the strongest predictive influence on match outcomes.

**8. References**

1. **Scikit-learn documentation** – *Random Forests, Logistic Regression, and Gradient Boosting*  
   → Corresponds to: Section 2 (*Chosen Algorithm*), Section 5 (*Algorithm Selection and Alternatives*).  
   https://scikit-learn.org/stable/user\_guide.html