Analyzing League of Legends Game Mechanics to Predict Match Performance: **Predicting the total damage players with TopLaner or Jungler roles deal in a match**

Project members: Victoria Halim, Nicholas Setyawan, Jeff Mario Kartanegara, Stephanie Daniella Hartono

1. Introduction

League of Legends is a competitive Multiplayer Online Battle Arena game where ten players are split into two teams of five to battle one another. Each team gathers useful resources such as gold and experience, engages opposing players in combat, and weakens the opposing team's buildings to accomplish the main objective of each match—destroying the opposing team's *Nexus* (the enemy base). Multiple factors decide the outcome of a match: strategy, playstyle, and more.

2. Research Question

In every match, players in each team allocate themselves to 5 distinct roles. *TopLaner* and *Jungler* roles in particular are responsible for the majority of the damage dealt in a match, which is a crucial factor since higher damage outputs would mean more dominance over the opposing team.

Different players focus on varying aspects of the game. To maximize total damage dealt in a match, players must balance four different game mechanics:

1. "*Farming*": Leveling up and earning gold to purchase useful game items.
2. *"Combat"*: Battling enemy champions to weaken their offensive power.
3. "*Objective*": Damaging the opposing team's offensive structures and the *Nexus*.
4. "*Support*": Assisting teammates and keeping them alive.

Thus, this research will aim to **predict the total damage a player with a TopLaner or Jungler role would inflict in a match, given the game mechanics they decide to focus on**.

3. Target Audience

League of Legends has a large competitive scene. All players are able to compete on an online ladder ranking, and highly-skilled players get to compete with their teams in official tournaments. The largest tournament is the annual League of Legends World Championship, one of the most-streamed competitions worldwide. The outcome of this report may help determine the game mechanics the *TopLaners* and *Junglers* of competitive teams should focus on to maximize their damage output, and therefore win. It can also aid non-professional players in emphasizing a playstyle, assisting them in climbing the online competitive ranked ladder.

4. Dataset and Features

The dataset chosen for this experiment was collected by Andrew Suter[1] and contains the end-game data of every *Challenger* ranked (Top 50 in their server) player's matches throughout January 2022. The dataset spans across three regions: North America (NA), Europe (EU), and Korea (KR). All the data points collected are distinct and were randomly selected per match to ensure independence. Selected features of the dataset are described in *Table 1* below.

Feature	Description
d_spell	Amount of times the summoner spell[2] on the *d* key is used
f_spell	Amount of times the summoner spell on the *f* key is used
champion	The character the player is playing
side	The team the player is on
role	The role the player is playing as
assists	Number of times the player helped a teammate kill an enemy
damage_turrets	Amount of damage dealt to turrets [3]
damage_objectives	Amount of damage dealt to objectives[4]
damage_building	Amount of damage dealt to buildings[5]
deaths	Number of deaths
gold_earned	Amount of gold earned
kda	The player's **k**ill **d**eath **a**ssist ratio[6]
kills	Number of kills
level	Player's champion level
time_cc	Time spent crowd controlling[7]
damage_total	Total damage dealt by a champion
damage_taken	Total damage a champion took

[1] Original dataset: LoL Challenger Soloq Data (Jan, Kr-Na-Euw) | Kaggle
[2] Helpful abilities players can use on their champions
[3] Enemy structures that deal damage to the player
[4] Includes enemy's offensive structures, the Nexus, and enemy 'Epic Monsters'
[5] Includes enemy's offensive structures and the Nexus
[6] Ratio is calculated using the formula: *(kills + assists) / deaths*
[7] Abilities that limit the mobility/combat ability of enemies, such as *stun*, *slow*, *charm*, etc.

minions_killed	Number of minions[8] killed
vision_score	Player's vision score[9]

Table 1: Selected Features from the Dataset

Support	Objective	Combat	Farming
assists	damage_objectives	kda	gold
vision_score	damage_turrets	kills	level
time_cc	damage_building	deaths	gold_earned
	turret_kills	damage_taken	minions_killed

Table 2: Categorization of Features into Predefined Game Mechanics

It is important to note that the *role* feature of the original dataset had been modified by the project assessors to only include two roles: *TopLane_Jungle* or *Other*. *TopLane_Jungle* means that in the match instance, the player played as *either* a *TopLaner* or *Jungler*. *Other* means that the player played as one of the remaining roles outside *TopLaner* and *Jungler*.

Several columns were removed from the original dataset due to the rationales below:

- *d_spell* and *f_spell* were removed since they are too vague; they do not specify which summoner spell was used.
- *minions_killed* was removed since it can only take two, overly broad values; either "Few" or "Many". It is likely to be unhelpful to the research.
- *side* was removed since it is irrelevant to the research question.

[8] Offensive enemy units that are continuously spawned by the Nexus
[9] How much vision a player has influenced in the game

5. Methodology

5.1 Merging datasets

The original dataset stored all the match data in three separate CSV files, organized by region (NA, EU, KR). Region is irrelevant to this research, so all the match data was merged into a single dataframe. The merged dataframe contains 17228 data points.

5.2 Filtering relevant data

The research question is only relevant to matches where the player played as a *TopLane_Jungle* role. Thus the dataset was filtered to reflect this.

In addition, each champion tends to be played in certain roles to maximize their unique abilities. For instance, champion *'Lulu'* is almost always played as a supportive role, but can be chosen to be played as a *TopLaner*, though this happens very rarely.

Champions unsuited as a *TopLaner* or *Jungler*, but were played as either of those roles, are likely to introduce noise into the research data. Thus we must only focus on champions *typically* played as *TopLaner* or *Jungler* roles, since that is what is relevant to the research. To achieve this, the dataset was filtered again to only include champions played as *TopLane_Jungle* role more than 50% of the time. Overall, the number of data points used for the research was reduced to 6123 (~35% of the original dataset)

Now since the champions used are out of the scope of the research question (we are only focusing on game *mechanics*) and all instances have the role *TopLane_Jungle*, the *role* and *champion* columns can be dropped from the dataset.

5.3 Train-test split

The filtered dataset was shuffled and split into training and testing sets (75/25 split). The models will be built using the train set, and then evaluated using the test set. This split was done to more accurately test the performance of the models created against unseen data. *All subsequent steps performed on the data are done only on the **training** set*, until stated otherwise.

5.4 Numerical Imputation

The current dataset has missing values, represented by *NaN* values. It is safe to assume that these values are Missing Completely at Random. Numerical imputation was chosen to deal with this issue to preserve as much data as possible for data processing, which, in turn, will help the model become more reliable. This was done by replacing missing data points with the mean of the entire column. The imputation was done in such a way that integer values remained as integers, and decimal values as decimals.

5.5 Ordinalization

5.5.1 Rationale

Though the dataset consists of continuous values, we have chosen to treat this research as a classification problem, ordinalizing all the features into three categories. Although this leads to loss of a certain degree of predictive power in the resulting predictive models, displaying and analyzing the data in a simpler manner allows more interpretability for members of the target audience who may not be familiar with data science.

5.5.2 Binning

All the columns were ordinalized into three different classes depending on their numeric value: "*low*" (0), "*mid*" (1), and "*high*" (2). Each feature was binned into the three classes with the assumption that most values would be classified into the *"mid"* category, with less occurrences of the two extremes *"low"* and *"high"*. As such the values of each feature were classified such that the frequencies of *"low"*, *"mid"* and *"high"* would mimic the shape of a standard normal distribution curve as much as possible, as shown on *Figure 2*.

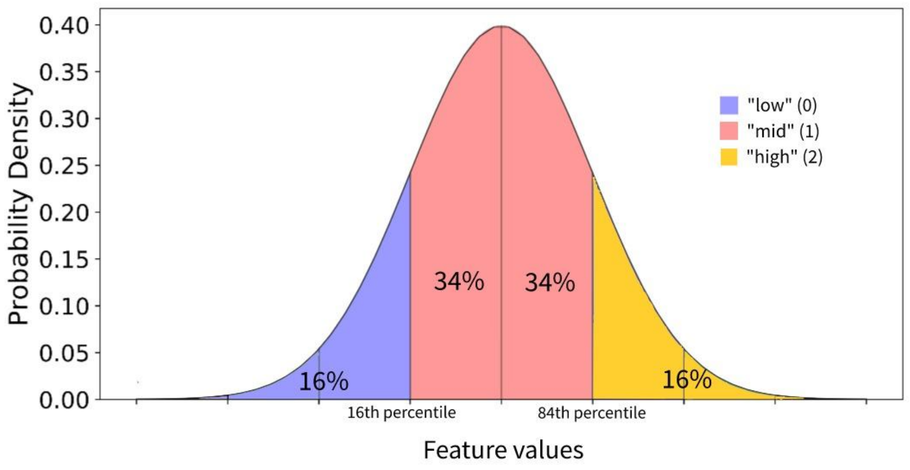

Figure 1: Normal distribution curve, modified to visualize how features will be binned (Galarnyk, 2018)

This was achieved by the following steps:

For each feature,

1. Sort the dataset in ascending order of the feature.
2. Find the 16th percentile of the dataset. We will call this value *α*.
3. Find the 84th percentile of the dataset. We will call this value .
4. Form the bins for each ordinal category of the feature, where:
    a. "low" (0): [0, *α*]
    b. "mid" (1): (*α*,)
    c. "high" (1): [, infinity]
5. Use the bins to ordinalize the training set.
6. Store the bins used to ordinalize the training set in a dictionary so the *same* bins can be used to ordinalize the test set later on.

5.6 Feature selection

Considering too many features for the research would just contribute to noise. Thus, we must only select features sufficiently correlated with *damage_total* for further data processing. Normalized Mutual Information (NMI) was the chosen measure of correlation since its limited range of [0, 1] provides more interpretability.

A feature was deemed to share a significant relationship with *damage_total* if its NMI with the variable exceeds 0.125; this threshold was chosen since it filters out an appropriate amount of features. Below are the features which were selected for further processing on the basis of NMI.

Feature	**NMI with *damage_total***
gold_earned	0.3305
level	0.2767
damage_building	0.1715
damage_taken	0.1311

Table 3: Selected features and their mutual information with damage_total

5.7 Predictive Models

5.7.1 Decision Tree

The training set was used to model a decision tree which predicts the class of *damage_total*, given input data containing the selected features (*gold_earned*, *level*, *damage_taken*, *damage_building*). The choice of a decision tree is because they effectively visualize classification.

10-fold cross validation was used to determine the decision tree depth between the range 3 to 7 which gives the decision tree model the highest accuracy. The limited range is necessary to retain the decision tree's interpretability. The best depth was found to be three, and the decision tree was modeled with this value. The model visualization can be found in *section 6.1*.

5.7.2 k-Nearest Neighbors

A *k*-Nearest Neighbors (k-NN) algorithm was also used to create another predictive model. The algorithm places each unclassified data point into a class based on the classes of the top *k classified* data points with the shortest distance (most similar) to it. It is crucial to pick a *k*-value of the right size, as *k*-values that are too small are more sensitive to noise in data, and a *k*-value that is too big may include points that belong to other classes.

10-fold cross validation was used to determine the k-value between 3 and 11 which gives the k-NN model the highest accuracy. The limited range was to avoid computations from being too expensive and unnecessary, given the small scope of this research. The best k-value was found to be seven, and thus the final k-NN model used this k-value.

5.8 Model Evaluation

The test set was modified such that it only included selected features from *section 5.6*, and was ordinalized with the bins from *section 5.5*. The decision tree and k-NN models were then evaluated using the modified test set. Results of the evaluation are as shown below.

5.8.1 Confusion Matrices

Confusion matrices were used to visualize the performance of the predictive models. The matrices compare the classes given by the predictive model to their true class. The diagonal boxes (spanning from the top-left to the bottom-right boxes) represent the correct classifications from the model; in other words, the model was able to predict the same class as the actual data.

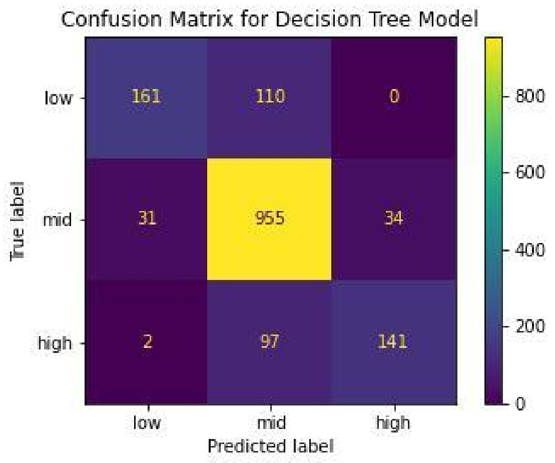

Figure 2: Confusion Matrix for Decision Tree Model

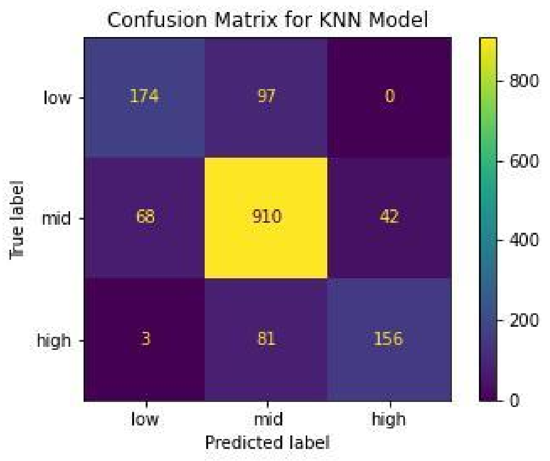

Figure 3: Confusion Matrix for k-Nearest Neighbors Model

Both models make most accurate predictions for *damage_total* values that fall into the "*mid*" category. However, both lack accuracy for predicting the minority classes "*high*" and "*low*". The decision tree was able to more accurately classify *"mid"* values, however the k-NN model was better at classifying the minority classes *"low"* and *"high"*.

5.8.2 Performance Metrics

The models' accuracy score was used as a performance metric since it indicates how often they predict the correct class for the test set. However, accuracy can be misleadingly high in imbalanced problems such as in this research, where *"low"* and *"high"* are minority classes. *f1* score, which is the harmonic mean between the calculated precision[10] and recall[11], takes into account this class imbalance. Thus it was also used as a performance metric.

[10] Agreement of the true class labels with those of the classifier's
[11]How many positive predictions the model made compared the the actual number of positives from the data

Both metrics range from 0 to 1; as the value approaches 1, it can be said that the model is more accurate/classifies the data well.

Statistic	Decision Tree Model	KNN Model
Accuracy	0.8210	0.8099
f1	0.7491	0.7500

Table 4: Comparing the Performance Metrics of Both Models

Both models have very similar and relatively high accuracy and *f1* scores. All in all this indicates that they can predict correct classes well (though not excellent), despite the class imbalances.

6. Results and discussion

6.1 Interpretation of decision tree

Color-wise, the decision tree is interpreted as such:

- Boxes with darker shades of **green** have higher proportion of **low** *damage_total*,
- Boxes with darker shades of **purple** have higher proportion of **medium** *damage_total*,
- Boxes with darker shades of **orange** have higher proportion of **high** *damage_total*.

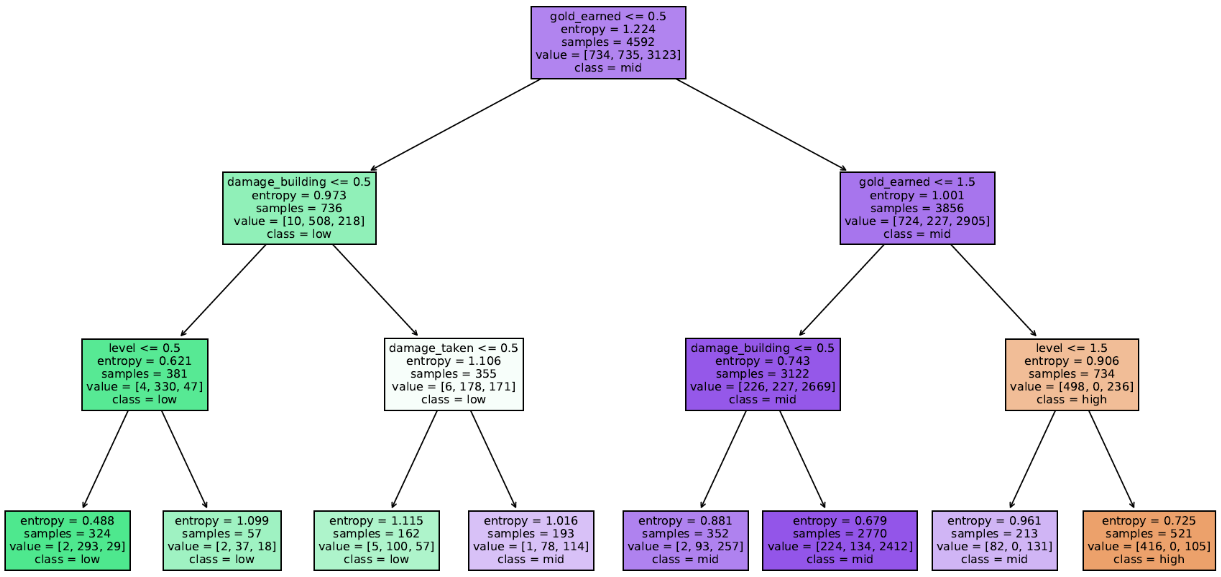

Figure 4: Decision Tree Model

The decision tree reveals that *gold_earned* and *level* seem to be the deciding factors of having a high *damage_total* by far. Low *gold_earned* (<=0.5) only allows a player to deal mid damage at most. High *gold_earned* guarantees that a player will at least deal mid damage, and increases the chance that a player will deal high damage by about 70%. High *gold_earned* coupled with high *level*, greatly increases the probability of dealing high damage.

damage_building is far less pivotal in determining a player's *damage_total*, but still supplement it nonetheless; players with higher *damage_building* are more likely to deal slightly higher damage. *damage_taken* is the least influential variable of the four, and does not show a clear relationship to *damage_total*.

6.2 Application to research

In descending order, the four most influential variables that affect the total damage at the end of the game are: *gold_earned*, *level*, *damage_building*, and *damage_taken*. As was described on *Section 1* and *Table 2*, these variables can be classified into broader game mechanics.

Objective	Combat	Farming
damage_building	damage_taken	gold
		level

Table 5: Game mechanics the four most influential variables fall into

Focusing on *farming* generates a significantly larger net total damage than the other game mechanics. This is since when players focus more on *farming*, the following occur:

- They can purchase more performance-boosting items to aid their champions since they have more *gold*
- Their champions have stronger abilities and higher damage output since they are of a higher *level*

This allows the player to deal significantly more damage in the later parts of the game—also called the "late-game phase".

The late-game phase is considered to be high stakes as a single action can change the results of the game. At this phase, it is expected that some of the champions (also known as the *carry*[12]) would have reached their maximum potential damage, as sufficient time has passed to allow them to purchase the performance-boosting items, and to level up. As a result of having the ability to deal more damage, the champion will be able to win fights with enemy champions, destroy enemy buildings, and eventually capture the *Nexus*, in a much swifter manner.

Thus to maximize the chances of winning the match at the late-game phase, teams should have a carry champion that can "*farm*" efficiently.

[12] The term *carry* comes from the phrase "easily carrying the team to victory"

7. Limitations and future research

One of the main limitations of this investigation was the assumptions made during ordinalization. It was assumed that the frequency of *“low”*, *“mid”* and *“high”* values followed a normal distribution, and were binned according to this assumption. This could have caused some of the data to be ordinalized into inappropriate categories. Future research should seek cooperation with a domain expert to determine the precise values considered as “*low*”, “*mid*”, and “*high*” values for each feature.

In addition, this investigation did not determine the exact number of features which result in optimal model performance. Future research could experiment with different normalized mutual information thresholds and determine which one results in the best feature selection process.

Since many features in the dataset are continuous, the research question could perhaps be explored further by utilizing regression analysis in future iterations.

8. References

Galarnyk, M. (2018, June 4). *Explaining the 68-95-99.7 Rule for a Normal Distribution*. Medium; Towards Data Science. https://towardsdatascience.com/understanding-the-68-95-99-7-rule-for-a-normal-distribution-b7b7cbf760c2

Suter, A. (2022). *LoL challenger soloq data (jan, kr-na-euw)*. Www.kaggle.com. https://www.kaggle.com/datasets/andrewasuter/lol-challenger-soloq-data-jan-krnaeuw