Machine Learning
Academic Year 2024/25

Winning in League of Legends

Pol Ballart Álvarez - 1638390
Miguel Ibáñez Molina - 1571691
Juan José Rodríguez Potosí - 1606379

January 23

Universitat Autònoma de Barcelona

1 Introduction

In this project we decided to focus our study on match data from the videogame and e-sport (electronic sport) League of Legends.
League of Legends is a MOBA (multiplayer online battle arena), that is, a videogame in which two teams of 5 players each face each other with the goal of destroying the enemy nexus, which is the main and most protected structure of each team.
From each nexus, the so-called minions are periodically spawned, units not controlled by any player and which march toward their assigned lane on the map. Their purpose is to protect the players on their own team and to grant experience and gold to players on the opposing team.
Each base also has 3 inhibitors, structures that, when destroyed, allow the enemy team to spawn more powerful minions for 5 minutes in that specific lane.
There are 3 lanes on the map, known as Top lane, Mid(dle) lane and Bot(tom) lane, which would translate as upper, middle and lower lane. In these lanes there are usually, respectively, one player (called the toplaner), another player (called the midlaner) and two players (called ADC and support), where ADC stands for attack damage carry (the main source of physical damage on the team).
The area between the lanes is called the jungle, where the well-known jungle camps are located. These contain other units not controlled by any player; they do not leave their camps and they grant gold and experience to the players (one per team) who have the jungle role.
In each lane there are 3 towers that serve as defensive structures that create a protected zone for the players of each team.
The map is divided diagonally by the river, which lies in a region equidistant from both teams' bases. In the river there are two pits, areas where very powerful neutral monsters spawn and provide benefits to the team that defeats them.

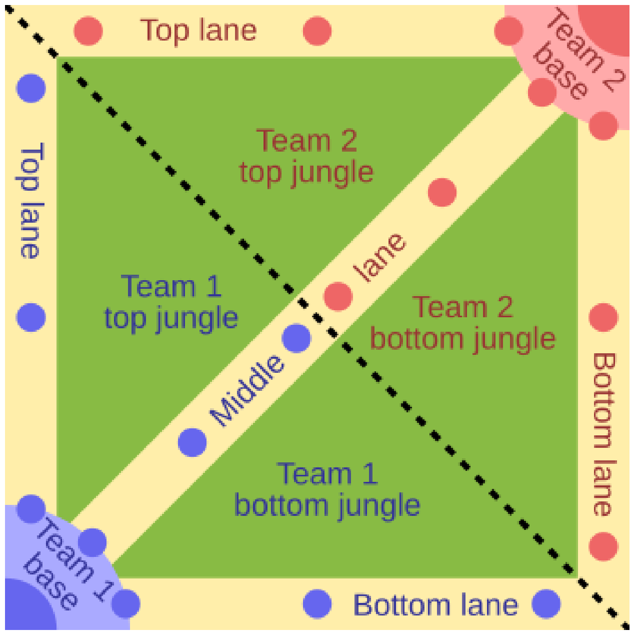


Figure 1: Generic MOBA map.

We chose as our dataset the official match record from Riot Games, the company that created the videogame. Specifically, we selected a dataset of ranked matches from Season 7, i.e., matches played during the year 2017.
Ranked matches are those in which participants give their best to win; that is, they are not casual matches just to pass the time. This is because the game has a tiered ranking system measured by different competitive leagues, and each player's objective is to reach the highest possible league each season. This is achieved by gaining points (League Points) with wins and losing them with defeats. This would be comparable to a regional football league, where even if the level is not the highest, players try their hardest and compete seriously. A casual match would be equivalent to playing football with friends.
For this reason, we believe this record provides the most necessary and suitable data for a study, since it includes several levels of play (low, intermediate and high) in which players still compete seriously.

Our study objectives are:

- Predict, from the individual and team statistics of a match, whether a player belongs to the winning team or not.
- Study whether there are champions that influence a team's victory more (whether picked or banned), i.e., assess the impact of playing with stronger champions.
- Study the relevance of the different neutral objectives in the match and compare them.

2 Dataset description

The dataset comes from Riot Games' official match record for the year 2017, as explained above. The source of the data is the Kaggle website. Since it is an official match record, we assume there are no sampling errors and that all the information provided is true.

The variables we have are:

Variable	**Description**	**Type**
`matchid`	Unique match identifier.	Categorical, independent
`platformid`	Region where the match was played.	Categorical, independent
`seasonid`	Game season to which the match belongs.	Categorical, independent
`duration`	Match duration (in seconds).	Numeric, dependent
`version`	Game version used for the match.	Categorical, independent
`id`	Unique identifier of the player within the match.	Categorical, independent
`player`	Players 1–5 belong to team 1, 6–10 to team 2.	Categorical, independent
`championid`	Identifier of the champion used by the player.	Categorical, independent
`ss1, ss2`	First and second summoner spell used by the player.	Categorical, independent
`role, position`	Role within the team (jungle, support, etc.).	Categorical, independent
`win`	Whether the player won the match or not.	Categorical, dependent
`item1-item6, trinket`	Items purchased by the player during the match.	Categorical, independent
`kills, deaths, assists`	Basic player statistics.	Numeric, dependent
`largestkillingspree`	Longest killing spree without dying.	Numeric, dependent
`largestmultikill`	Highest number of consecutive kills in a short time.	Numeric, dependent
`killingsprees`	Total number of killing sprees.	Numeric, dependent
`longesttimespentliving`	Longest time span without dying.	Numeric, dependent

Variable	Description	Type
`doublekills, triplekills, quadrakills, pentakills`	Number of double, triple, quadra and penta kills achieved.	Numeric, dependent
`legendarykills`	Number of legendary kills (streak of more than 7 kills without dying).	Numeric, dependent
`totdmgdealt, magicdmgdealt, physicaldmgdealt, truedmgdealt`	Total, magic, physical and true damage dealt.	Numeric, dependent
`largestcrit`	Largest critical hit dealt by the player.	Numeric, dependent
`totdmgtochamp, magicdmgtochamp, physdmgtochamp, truedmgtochamp`	Damage dealt to champions (total, magic, physical and true).	Numeric, dependent
`totheal`	Total amount of healing done.	Numeric, dependent
`totunitshealed`	Total number of units healed.	Numeric, dependent
`dmgselfmit`	Damage mitigated by the player.	Numeric, dependent
`dmgtoobj, dmgtoturrets`	Damage dealt to objectives and enemy turrets.	Numeric, dependent
`visionscore`	Vision score (use of wards and enemy detection).	Numeric, dependent
`totdmgtaken, magicdmgtaken, physdmgtaken, truedmgdealt`	Damage taken by the player (total, magic, physical and true).	Numeric, dependent
`goldearned, goldspent`	Gold earned and spent by the player.	Numeric, dependent
`turretkills, inhibkills`	Turrets and inhibitors destroyed by the player.	Numeric, dependent
`totminionskilled`	Total number of minions killed.	Numeric, dependent
`neutralminionskilled, ownjunglekills, enemyjunglekills`	Neutral minions killed, including those in own and enemy jungle.	Numeric, dependent
`totcctimedealt`	Total time of crowd control applied to enemies.	Numeric, dependent
`champlvl`	Champion level at the end of the match.	Numeric, dependent

Variable	**Description**	**Type**
`pinksbought, wardsbought, wardsplaced, wardskilled`	Control wards bought, placed and destroyed by the player.	Numeric, dependent
`firstblood`	Whether the player got first blood.	Categorical, dependent
`teamid`	Identifier of the player's team.	Categorical, independent
`firsttower, firstinhib, firstbaron, firstdragon, firstharry`	Whether the team destroyed the first turret, inhibitor, Baron, dragon or Rift Herald of the match.	Categorical, dependent
`towerkills, inhibkills, baronkills, dragonkills, harrykills`	Turrets, inhibitors, Barons, dragons and Heralds destroyed by the team.	Numeric, dependent

Our dataset is relevant because it records several seasons and includes a reliable amount of information. In our case, we restrict to Season 7 specifically because it is the one for which we have the most data, and by focusing on a single season the resulting dataset is more consistent.
Matches across different seasons vary a lot because very important game features are changed and then remain unchanged for the rest of that season.
Having so much information about each match and player allows us to make predictions or studies on many different aspects, such as which team will win, whether they surrendered, etc.
This dataset is ideal since it contains a large variety of information in large quantity, and it more than provides the variables we need.
We have the intuition that within the large dataset there will be a subset of variables that are not very relevant to our objectives and that we will be able to discard.

3 Data exploration

Table 1: Time

Statistic	Mean	Std. Dev.	Minimum	Q1	Median	Maximum
Match duration (s)	1824.08	509.89	190	1536	1830	4991
Longest time alive (s)	626.71	309.60	0	431	587	2982

Table 2: Kill and death statistics

Statistic	Mean	Std. Dev.	Minimum	Q1	Median	Maximum
Kills	4.94	3.94	0	2	4	67
Deaths	5.79	3.87	0	3	5	44
Assists	7.96	5.12	0	4	7	53
Firstblood	0.099	0.299	0	0	0	1
DoubleKills	0.43	0.69	0	0	0	10
TripleKills	0.10	0.32	0	0	0	5
QuadraKills	0.02	0.15	0	0	0	3
PentaKills	0.004	0.06	0	0	0	2
Killingsprees	1.34	1.26	0	0	1	13
Legendarykills	0.000001	0.001	0	0	0	1
Largestkillingspree	2.66	2.53	0	0	2	35

Table 3: Damage statistics

Statistic	Mean	Std. Dev.	Minimum	Q1	Median	Maximum
TotDmgDealt	113353	1632762	-2147484000	59350	108976	1064761
MagicDmgDealt	38636	48798	0	4421	17459	789034
PhysicalDmgDealt	69574	70604	0	11428	42822	917986
TrueDmgDealt	5142	1631145	-2147484000	462	2026	713886
LargestCrit	242	420	0	0	0	4876
TotDmgToChamp	17654	11473	0	9378	1,508	152607
MagicDmgToChamp	7697	9242	0	1307	4106	142469
PhysDmgToChamp	8948	9946	0	1448	4600	123871
TrueDmgToChamp	1008	1642	0	26	460	36143

Table 4: Healing and mitigation statistics

Statistic	Mean	Std. Dev.	Minimum	Q1	Median	Maximum
TotHeal	5439	5525	0	1765	3850	99164
TotUnitsHealed	2.25	2.33	0	1	1	49
DmgSelfMit	11456	14842	0	177	7184	311225
Totdmgtaken	23138.46	1186.306	0	15186	21473	165152
Magicdmgtaken	8084.95	5136.359	0	4495	7205	8458
Physdmgtaken	13994.38	7735.014	0	8592	12769	90,430
Truedmgtaken	1058.920	1285.725	0	275	660	31648

Table 5: Objective and gold statistics

Statistic	**Mean**	**Std. Dev.**	**Minimum**	**Q1**	**Median**	**Maximum**
DmgToObj	4365	6771	0	0	1091	84057
DmgToTurrets	2271	2991	0	0	1170	62488
Turretkills	0.91	1.21	0	0	0	10
Inhibkills	0.18	0.46	0	0	0	7
VisionScore	14.64	17.56	0	0	11	179
Pinksbought	0.9889597	1.407279	0	0	1	54
Wardsplaced	11.48005	7.476475	0	7	1	69
Wardskilled	1.769911	17.56	0	1	3	86
GoldEarned	11395	4019	643	8852	11336	40982
GoldSpent	10348	3879	0	7925	10275	70255

Table 6: Minion and control statistics

Statistic	**Mean**	**Std. Dev.**	**Minimum**	**Q1**	**Median**	**Maximum**
TotMinionsKilled	120.66	83.91	0	39	125	673
NeutralMinionsKilled	19.91	31.74	0	0	4	299
OwnJungleKills	11.95	19.91	0	0	2	174
EnemyJungleKills	7.96	13.25	0	0	1	181
ChampLvl	14.18	3.14	1	13	15	18
TotCcTimeDealt	426.98	659.15	0	112	251	25204

Table 7: Team statistics

Statistic	**Mean**	**Std. Dev.**	**Minimum**	**Q1**	**Median**	**Maximum**
Towerkills	5.7504	3.879	0	2	6	11
Inhibkills	1.0373	1.2597	0	0	1	13
Baronkills	0.4135	0.6069	0	0	0	5
Dragonkills	1.4274	1.2317	0	0	1	7

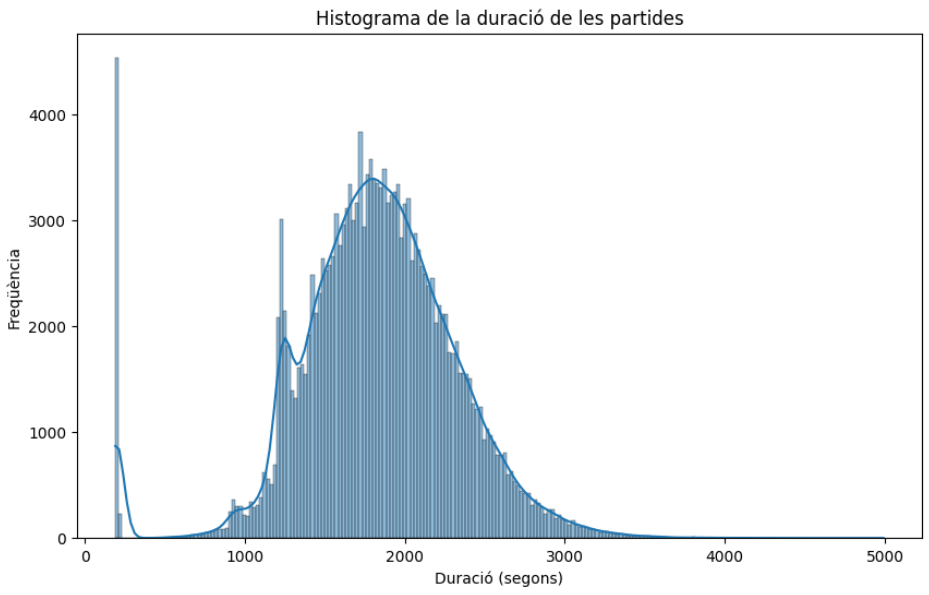


Figure 2: Histogram of match duration

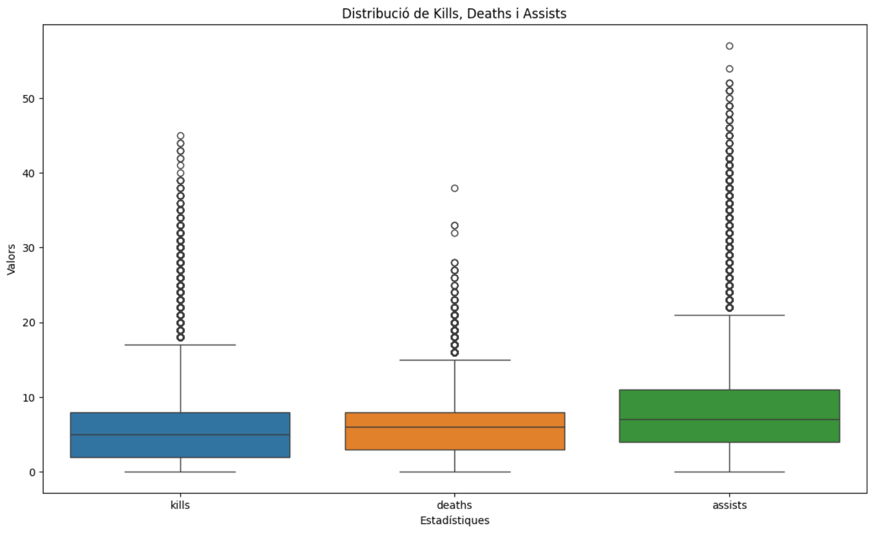


Figure 3: K/D/A boxplot

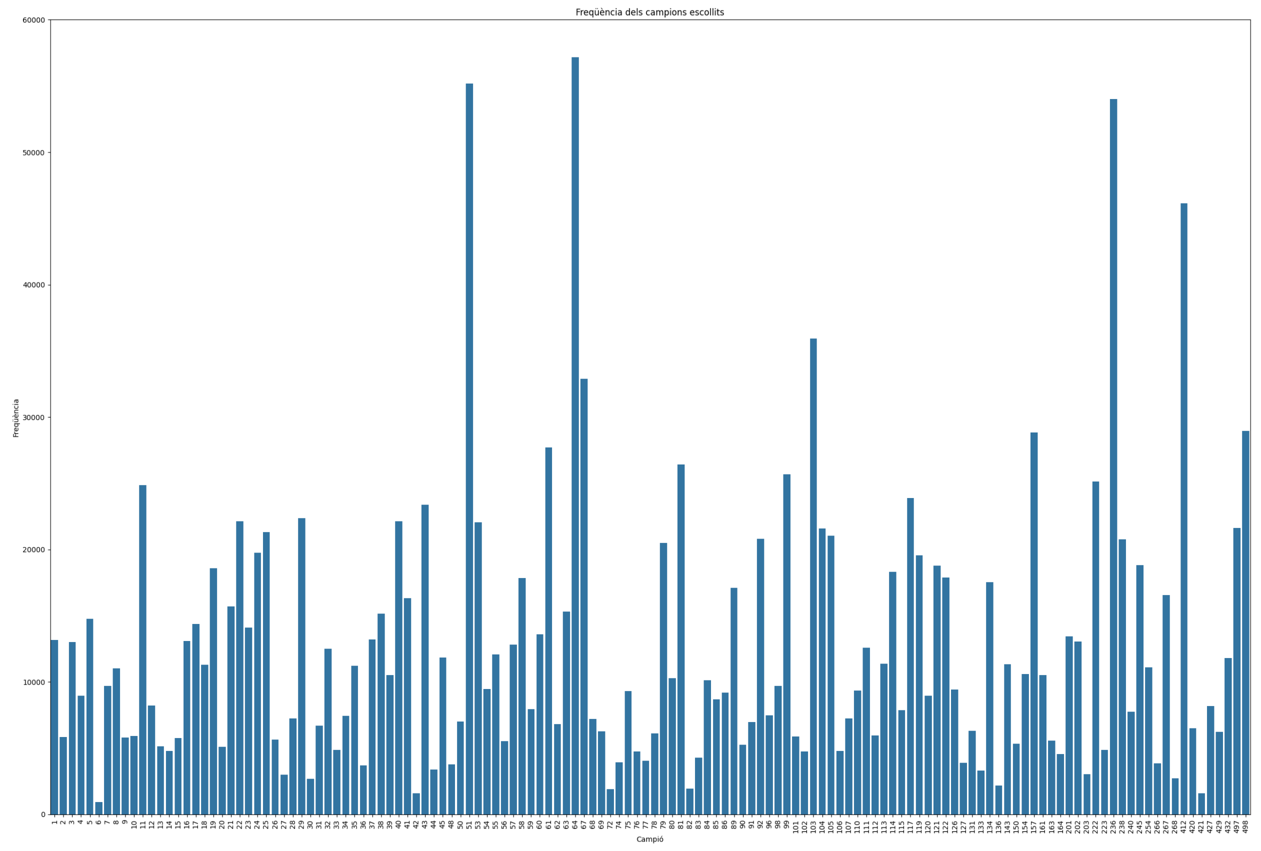


Figure 4: Champion frequency

There are different factors that can indicate that an entry in the dataset is an outlier. Among them we find:

- **Teams without the appropriate SS (Summoner Spell)**
  - SMITE: It is “mandatory” to have exactly one per team (the person playing the JUNGLE role).
  - An unusual SS combination is often an indicator that the player is intentionally trying to lose.
  - FLASH: the vast majority of champions take FLASH as one of their summoner spells, so it can be used as an indicator, although there are several cases where it is not strictly necessary.
- **Matches that end around the 20:00 minute mark**
  The typical reasons for this are:
  - One of the two teams gains such a large advantage in the first minutes of the game that the other team unanimously votes to surrender.

  - One of the players disconnects mid-game.
  - A team's mental factor. That is, even without a large disadvantage and still being able to win, a team may become demoralized.

- **Teams with disconnected players:**
  This can be identified because they did not buy any items during the entire match.

- **Matches in which a "remake" happens:**
  Since one of the players is disconnected from the start, the rest of the team can vote to end the game at minute 3:00 without it counting as a win or a loss.

- **Winning teams with the minimum number of turrets destroyed (5):**
  Normally, the number of enemy turrets destroyed is a clear indicator of a team's probability of winning. If a winning team took down the minimum possible number of turrets, this indicates that the deciding factor was a late-game teamfight and they ended up winning against all odds.

4 Application of machine learning algorithms

We chose the Random Forest Classifier and Gradient Boosting models.

Random Forest Classifier

A **random forest** is a supervised learning model that combines multiple decision trees to produce more accurate and robust predictions. It uses an ensemble approach where each tree is trained on a random subset of the data and features, and the final predictions are obtained by averaging (in regression) or voting (in classification) the results from the individual trees, which reduces overfitting and improves generalization. We found this model suitable because it yields good results when classifying data and it takes into account multiple models trained on random subsets of the dataset, which provides higher reliability. In addition, recall that our objectives were:

- Predict a player's victory given information about them and their team's statistics.
- Study whether some champions influence victory more than others.
- Study the relevance of the different neutral objectives.

Therefore, in the end this is a classification model between win and loss, and we want to see the importance of each variable to draw conclusions about what we are studying. For this reason, and given our data type, we believed a Random Forest model was appropriate. Since it is based on combining certain decision trees, we observe that these models in particular follow an intuitively understandable structure when making predictions. That is, with this type of data (a clear target variable—win or loss—and a large number of predictors) it is reasonable to follow a decision-tree-like structure for prediction, because we can anticipate that certain variables will stand out far more than others.

When building the model, we found that the vast majority of variables had zero relevance, so we filtered to keep only the most important ones, based on the relevance histogram and our knowledge of the data.

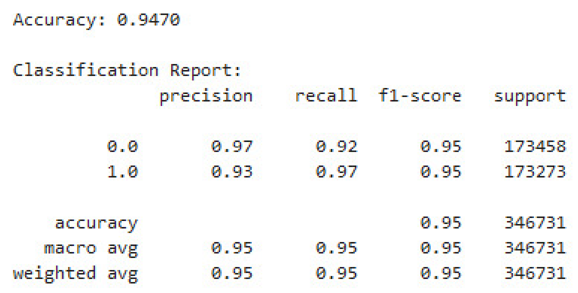
We then trained another model using this filtered dataset.

The variables used to train the model were: `championid`, `role`, `win`, `totdmgdealt`, `totheal`, `visionscore`, `totdmgtaken`, `goldearned`, `towerkills`, `inhibkills_y`, `firsttower`, `firstinhib`, `firstbaron`, `firstdragon`, `baronkills`, `dragonkills`, `harrykills`, `firstblood_y`, `totcctimedealt`, `kda`

We note that we added a new variable called `kda`, which combines `kills`, `deaths`, `assists` as $KDA = \frac{kills+assists}{deaths}$.

From this model and the previous one, and as we expected, we conclude that the most important variable is undoubtedly the number of turrets destroyed by the team. "If we remove all variables related to structure destruction, how good would the new model be?"
We thought the answer to this question would be negative, but it turns out that KDA becomes the most relevant variable to determine the outcome and it still achieves decent results.

Below are the results obtained from the three models:



```
Accuracy: 0.9470

Classification Report:
              precision    recall  f1-score   support

         0.0       0.97      0.92      0.95    173458
         1.0       0.93      0.97      0.95    173273

    accuracy                           0.95    346731
   macro avg       0.95      0.95      0.95    346731
weighted avg       0.95      0.95      0.95    346731
```

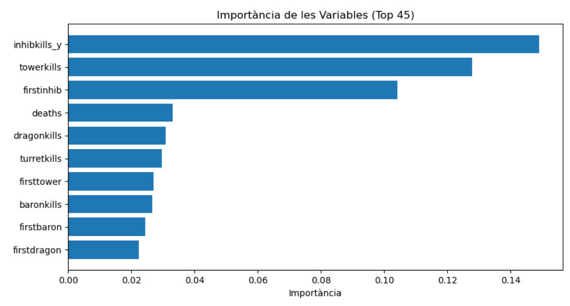
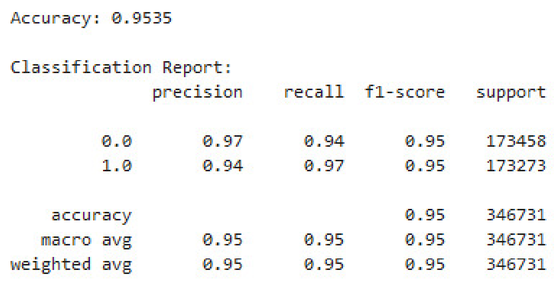


Figure 5: First model



```
Accuracy: 0.9535

Classification Report:
              precision    recall  f1-score   support

         0.0       0.97      0.94      0.95    173458
         1.0       0.94      0.97      0.95    173273

    accuracy                           0.95    346731
   macro avg       0.95      0.95      0.95    346731
weighted avg       0.95      0.95      0.95    346731
```

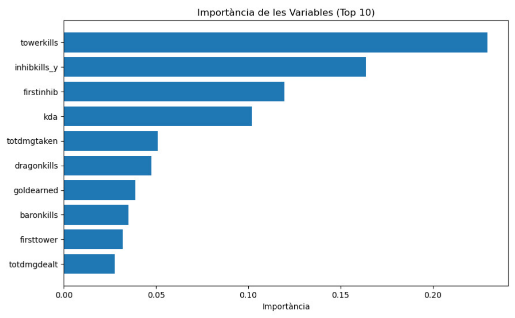
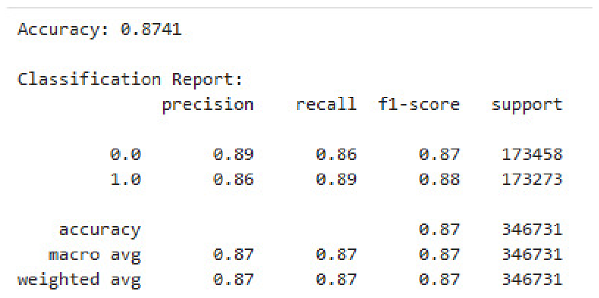


Figure 6: Second model



```
Accuracy: 0.8741

Classification Report:
              precision    recall  f1-score   support

         0.0       0.89      0.86      0.87    173458
         1.0       0.86      0.89      0.88    173273

    accuracy                           0.87    346731
   macro avg       0.87      0.87      0.87    346731
weighted avg       0.87      0.87      0.87    346731
```

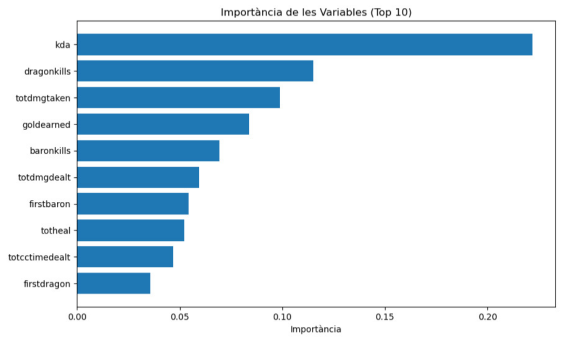


Figure 7: Third model

Left: Performance metrics. **Right:** Relative importance.

We see that the accuracy in the first model and in the second one is almost the same, with a difference of 0.8%, but when we reach the third model the accuracy clearly drops from 95% to 87%, which reflects once again that the most important variables are the turrets, as was to be expected.

Gradient Boosting

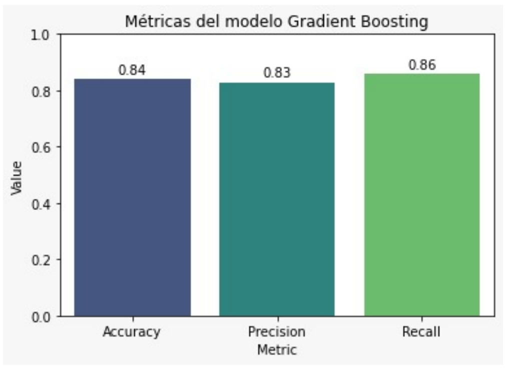
Gradient boosting models are robust and have a strong ability to capture complex relationships between variables. They are among the most widely used models today thanks to their strong predictive power, and for this reason we selected them.
The goal of the model is to predict, based on data from a match that has already been played,

whether a given player was part of the winning team or not. The large amount of available data and the strong influence of some variables on the final outcome allow us to make predictions with high confidence.
Structural measures such as: `towerkills`, `inhibkills_y`, `firsttower` and `firstinhib` are highly decisive in a match. In fact, almost "at a glance", it is possible, following simple and consistent criteria, to determine whether the player won with their team or not. When the model includes these variables, it achieves precision (*precision*), accuracy (*accuracy*) and recall (*recall*) metrics above 95%.
However, this result is not particularly interesting for our study, since structural data almost guarantees the outcome. For this reason, we trained two additional models where we excluded these structural variables.

- **First model**: We include data related to KDA (also highly decisive).
- **Second model**: We also exclude KDA data to further stress-test the model's performance.

The metrics and feature importances of the models are as follows:

Left: Performance metrics. **Right:** Relative importance.



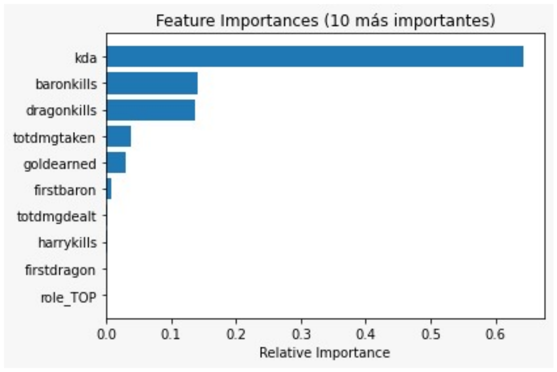
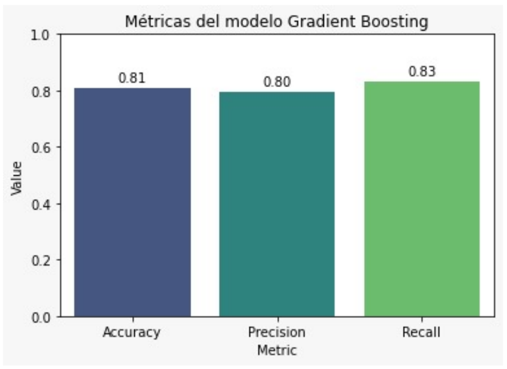


Figure 8: First model



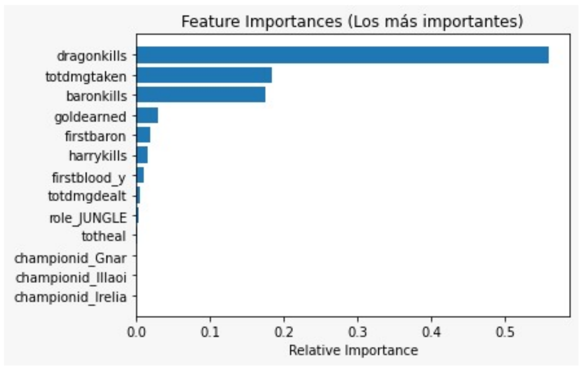


Figure 9: Second model

5 Results and interpretation

Regarding the comparison between the **Random Forest Classifier** and **Gradient Boosting** models, we should limit ourselves to comparing the results presented by the last Random Forest model and the first Gradient Boosting model, since both correspond to the most interesting set of variables. Looking at the metrics, we observe that Random Forest achieves better results, but this has an explanation that goes beyond the difference between models: it is mainly the difference in the number of estimators used. For Random Forest we used `n_estimators=100, random_state=42`, i.e., 100 trees with unlimited depth, at the cost of higher computational load time. For Gradient Boosting, instead, we used `(n_estimators=20, learning_rate=0.1, max_depth=3, random_state=42)`, which implies shorter computational time but worse results, although with a small difference.

The results we obtained do not surprise us in general; they are expected results. This happens because we have knowledge of the subject and we already had a prior intuition that ended up being confirmed.

It is easy to understand that in a videogame whose objective is to destroy the enemy base, the most important data to predict which team won are the structures destroyed during the match.
Once we ignore the structural data, it is also reasonable that neutral objectives become important, with Baron standing out (a very important objective that can decide the outcome in many matches) and dragons (which grant additional stats to the team that secures them throughout the match).
In this respect, we are somewhat surprised by the relatively low importance of the neutral objective called "Herald" (harrykills), since this objective can heavily damage an enemy structure and weaken it, and we previously saw that structures are the most important part of the game.
However, knowing the context, this can be understood because this objective was introduced later into the game. That is, since the game's release it has had dragons, Baron and structures, but the Herald appeared later (5 years after the game's launch, and our study uses matches 7 years after launch). This helps explain why players may not have been as familiar with this neutral objective and did not exploit it as effectively as the others, despite its potential.

Another important topic is KDA, and within it specifically deaths. Given how the game works, it is natural that KDA is important, since getting kills and assists grants gold that can be used to buy better items.
It is also understandable that deaths, among the components of KDA, are the most relevant, because when a champion dies, they are removed from play and cannot act for a few seconds. This downtime increases proportionally with match time; that is, if a champion dies early they may be unable to play for 5 seconds, but if they die at minute 40 they may be unable to play for 50 seconds. This gives a major advantage to the team that killed the champion, since during that interval they have a numerical advantage.

Even so, there are some results that surprised us, such as the fact that the chosen champions are, in general, a factor that has little influence on match outcome, especially compared to other match factors.
If we consider only champion relevance, the results again are not surprising: the champions with higher importance are the most played, which in turn are generally the most played because they are stronger. There are other reasons why a champion is played more, such as champions that were added recently to the game or champions that recently received new "skins" (alternate appearances).

Within this, it is logical that roles have more importance than champions themselves, since a role includes many champions. The ordering within roles is also natural, since the midlaner is located in the middle of the map and can relatively easily influence the other two lanes or the jungler. The same can be said of the jungler, who plays across the map and can help all teammates or become stronger by taking advantage of possible kills in different lanes. The jungler is also responsible for securing neutral objectives, since they have a summoner spell and an item dedicated to that within the game.
Going deeper, we find the items each champion bought. It is also natural that different items have their own relevance, and this is in fact already partially captured in champion relevance because a champion can be better for different reasons: their base stats are very good (high damage, high durability) or they have complex abilities that, even if they do not deal damage, provide strong utility to the team or hinder the enemy team. On the other hand, if the items typically bought on a champion have very strong stats or effects, the champion benefits from them.

Finally, we must keep in mind that the data comes from matches at all skill levels, even though they are ranked matches. In professional matches these parameters would have different relevance. One aspect we see very clearly is vision score, which reflects how much map information each player obtained through vision tools in the game (invisible wards that grant vision, tools that reveal them, etc.). Vision is much more important in professional matches, since players communicate verbally, which generally does not happen in non-professional matches (at most, two people might be playing together).

6 Conclusions and future work

In this study we used Gradient Boosting and Random Forest models to predict whether a player was part of the winning team of a match, based on both individual and global match data. The inherent complexity of these data requires models capable of capturing subtle relationships between variables and the outcome (target), which are not easily observable through simple statistical studies or our own empirical experience of the game.

The selected models not only demonstrated high predictive power, but they also allowed us to delve deeper into how deterministic certain variables are. This helped us better understand which features are most influential when deciding the outcome of a match. This knowledge is especially valuable, since it provides a clearer view of how some game elements directly impact victory or defeat.

This approach may have future applications both in the design of gameplay strategies and in the performance analysis of teams and players, and it is an example of how a model can help not only to predict but also to provide understanding of the deep behavior of a complex process. When carrying out this project we faced clear limitations. Some of these limitations are:

- **Data volume:** Since 10 players participate in each match and each player has many associated variables as we have seen, processing becomes heavy even after discarding some variables. The large data volume was beneficial for model training, but it made management harder, as well as cleaning and selecting the most relevant variables.
- **Accounting for different match states:** What would have been most interesting in this type of project would be a real-time computation of each team's probability of winning. In fact, in professional League of Legends World Championship matches this feature is already implemented and they occasionally show the probability of victory for each team at that moment. At the very least, making a win prediction using data from a specific intermediate minute, such as minute 20 or 15 (when surrender is not yet possible), would enable an interesting future-oriented prediction approach for an ongoing match.

We also have proposals for what could be done as a continuation of this project with more resources:

- **Applying what we learn to in-game bots:** This is a very good opportunity to improve the bots in the game. These are very limited in terms of how they play. They play poorly and are more a tool to get introduced to the game than to improve performance once you already know how to play.
  A project could aim to add different bot skill levels, that is, improve bot quality up to the point of creating a prototype of bots that win 100% of matches (as happens in chess). This would be very useful, for example for professional players, to study the game and see how a "perfect" machine understands it. This also raises the debate of whether there exists a perfect way to play League of Legends, since there are different ways to interpret the game.

- **Real-life example:** Recently, we found on social media a machine learning project similar to ours carried out by a Spanish professional League of Legends team (Giants Gaming).
  We found that they have been working on a tool that computes, given the champions already selected in a match (allies and enemies), which champion has the highest probability of winning. It takes into account synergies with allied champions and disadvantages against enemy champions.
  It is a project more focused on champions; our work is more general and considers variables that their work does not emphasize as much. On the other hand, their work has a more commercial focus.

7 Appendices

The code used for the project is provided as separate files, apart from the PDF.
The data were obtained from the website:
`https://www.kaggle.com/datasets/paololol/league-of-legends-ranked-matches`
where the person who uploaded the dataframes states that they were obtained from Riot Games' database (the owners of the videogame). This database can be consulted at the following website:
`https://developer.riotgames.com/docs/lol#data-dragon`