Predicting the Outcome of a Dota 2 Game

Valeriia Chirkova
Central Supelec
Russia
valeriia.chirkova@student-cs.fr

Julien Crabie
Central Supelec
France
julien.crabie@student-cs.fr

Victor Sanchez Chavez
Central Supelec
Peru
victor.sanchezch@student-cs.fr

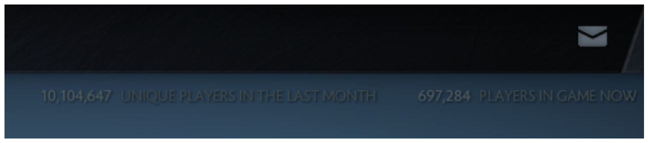
Arnaud Cluzel
Central Supelec
France
arnaud.cluzel@student-cs.fr

ABSTRACT

The purpose of the project is the outcome prediction of Dota 2 matches, a popular video game with 2 teams competing against each other. Prediction is based on the strength/aptitudes of each player based on their previous performance. We 'll consider that our model is successful if it gives better results than luck (50%).

1 INTRODUCTION

Dota 2 is one of the most played game in history. It had between 10 and 11 million unique players in the last month and considers approximately 800 thousand players, per day, connected at the same time all around the world. It 's so famous that championships are organized all around the world. These are later considered as qualifiers for the biggest event of the year (during August): The International. The tournament 's first price lies around 10 million dollars. The best teams around the world participate in this world cup of Dota 2.



10,104,647 UNIQUE PLAYERS IN THE LAST MONTH 697,284 PLAYERS IN GAME NOW

Figure 1: Number of players

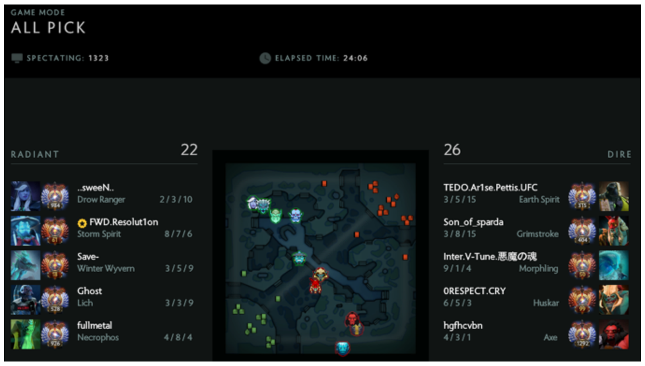
To make this project comprehensible, we provide a brief overview of the gameplay of this e-sport. Each match of Dota 2 takes place on the same virtual map with two sides: the Radiant and the Dire. We note here that both sides are equivalent in winning chances. Each team plays on one side and "attack" the other side. The winning team is the team that destroys the last building, called the ancient, of the opposite team.

Heroes

During a game, each player controls one hero (the in-game character). Today, there are 116 heroes and new heroes are frequently created by game developers. Each hero has specific characteristics with unique abilities and powers. The "Force" heroes are the most resistant, but they don 't make too much damage; the "Intelligence" heroes are not resistant at all but they have a lot of magic that makes a lot of damage. Finally the "agile" heroes are the fastest ones and are also called "hard carry" because they can make critical damage to other heroes. The underlying information here is that some heroes synergize well together and that players have to compose their team with heroes that are aligned with a common strategy. Picking wisely is essential to win a game.

Draft

The hero each player controls is determined during the draft. In the draft, each team 's captain will take turns banning and picking heroes from the pool of 116. Banning a hero removes it from the pool so neither team can have a player control it, and picking it selects a hero for one of their players to control (two players cannot pick the same hero). As each pick or ban is made, team captains will react and adjust their upcoming draft choices to best suit their own strategy while simultaneously trying to counter the strategy their opponents appear to be crafting. In total, the draft process takes about 10 minutes, the result of which is both teams now have their 5 hero lineups with which to execute their strategy of destroying the opposite ancient.



GAME MODE
ALL PICK
SPECTATING: 1323 ELAPSED TIME: 24:06

RADIANT		22	26		DIRE
..sweeN..	Drow Ranger	2 / 3 / 10	TEDO.Ar1se.Pettis.UFC	Earth Spirit	3 / 5 / 15
FWD.Resolut1on	Storm Spirit	8 / 7 / 6	Son_of_sparda	Grimstroke	3 / 8 / 15
Save-	Winter Wyvern	3 / 5 / 9	Inter.V-Tune.悪魔の魂	Morphling	9 / 1 / 4
Ghost	Lich	3 / 3 / 9	0RESPECT.CRY	Huskar	6 / 5 / 3
fullmetal	Necrophos	4 / 8 / 4	hgfhcvbn	Axe	4 / 3 / 1

Figure 2: Drafting phase

The Game

After the draft, players control their heroes over a map. All heroes begin the game in lvl 1 and gain power throughout the 30-60 minutes of the gameplay phase. Typically, the team that increases their heroes ' power faster than the other team overpowers their opponent and is able to destroy their ancient. There are two ways for a hero to increase its power: killing creeps, these are little soldiers that go out every 2 minutes from the ancient to each lane (Top, Middle and Bottom), and killing the heroes of the opposite team. Accumulating gold helps you to later purchase items (that gives you more powers) and accumulating experience to gain access to powerful spells and abilities.

Figure 3: Map of Dota2

2 RELATED WORK

According to the paper done by Kevin Conley, Daniel Perry. *How Does He Saw Me? A Recommendation Engine for Picking Heroes in Dota 2, 2013*, they could predict an outcome that solely looking at the hero composition of a team and whether or not that team was victorious can provide a useful model for match outcome prediction and as a result, hero recommendation. Despite the high accuracy, they didn 't consider the changes in the game patches (during several months, something that can change the outcome) and the players. Later on, this article was taken in consideration by *Song, Kuangyan, Tianyi Zhang, Chao Ma "Predicting the winning side of DotA2"*. CS229 2015and they started working with the players information, but there were some big assumptions:

(1) The players in one game have a similar simulation level
(2) The players could play every hero equally well

These assumptions were clear that they could affect the outcome, but they were necessary to just work with the problem of "Hero Selection". Finally, articles were taken in consideration by *Petra Grutzik Joe Higgins Long Tran "Predicting outcomes of professional Dota 2 matches"* Dec16, 2017 and include the work from the perspective of sports analytics seeing it in parallel: Sabermetrics and rigorous statistical analysis first transformed baseball management during the 1930s-1940s, which then led to a revolution in statistical applications in other sports and opened new fields of research in statistics itself. Progress has been made on predicting the outcomes of two player games. Nevertheless, games with multiple teammates on each team prove more difficult to predict. There were good outcomes from the research but there is still missing the good relation between the information of heroes and the information of players.

3 PROBLEM DEFINITION

The objective of the project is to predict the outcome of a Dota 2 game based on previous experience of the players using the created set of performance metrics for each player. This approach enables us not to use during-game information for prediction, thus, imitating, real situation of game predictions. In order to realize this, numerous steps of data manipulation were needed. Details are described below.

4 DATASET

4.1 Data collection

For this project the dataset was collected from the website www.opendota.com. The website has statistics of Dota 2 game all around the world and of all their players. Unlike approaches used in other papers reviewed, our strategy was to focus only on the best 100 professional teams. The process of getting specific data of the players for the ranking of the 1st 100 professional teams appeared to be quite challenging. Prior to collecting data we created a manual list considering the ID of the team and the ID of the player.
With the list of 500 players created and the Dota 2 API loader Library (https://dota2api.readthedocs.io/en/latest/), we considered some functions to import the Data that we required for the project. The main functions we used were:

- get match history() - to take out the first 100 games of each professional user;
- get match details() - to take out the information required for each of the games.

For the first Function, we took out the fi*account_id*fi that is the ID of the professional User. For the second function, we take the following features to build our dataset:

- match ID - ID of the game;
- radiant win - the winner team - Radiant (1) or Dire (-1);
- cluster - server of the game;
- game mode name - type of game;
- duration - time the game was played;
- hero id - hero used for the game;
- account id - identifier of the user playing the game;
- player slot - the role of the player ;
- kills - number of kills during the game;
- deaths - number of deaths during the game;
- assists - number of assists during the game (when a player helps to kill another hero);
- gold per min - the gold earned by killing heroes,creeps and assists;
- last hits - a key to become a good player, the last hit to kill a creep and get gold.

- xp per min - the experience that the player gets during the game (his level)
- league id - type of game: professional or regular.

Total number of features is 97 considering three important blocks:

- Identification (match id , account id and the outcome of the game);
- Information of the game;
- Information of players performance.

The dataset built considered 438 professional users (based on initial list), 29 185 games and 97 features. The construction of our dataset ensures that we have good data and that covers our requirements.

4.2 Data preparation

Resulting dataset required a lot of data preparation including data transformation. First thing was data cleaning i.e. checking dataset for inconsistent values. We applied the following modifications:

- Get rid of the duplicated games which appeared because selected players in the list can participate in the same game;

- Select only the games that lasted more than 10 min. Histogram below shows that dataset was composed of some games with a duration of 0 min corresponding to inconsistent data as well as other games with duration of less than 10 min which are not representative in terms of game statistics.

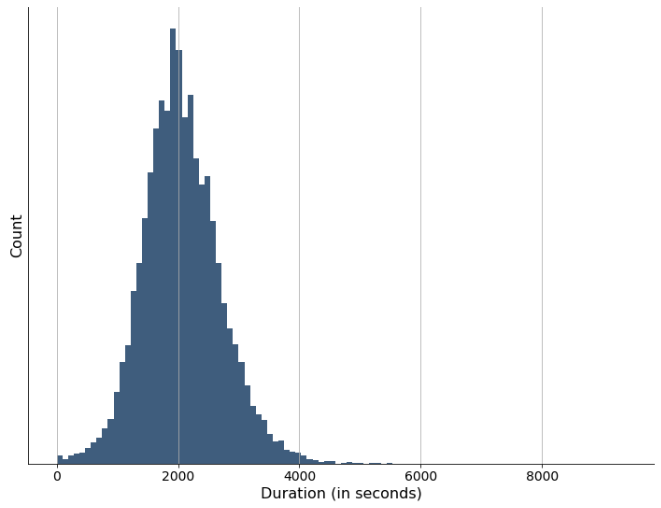


Figure 4: Duration of games

The dataset was then split into one training set and one testing set with a 90/10 ratio. As we wanted to predict the outcome of a game before it begins the split was made with respect to time of the game. So the past games were used to build features associated with the player performance and train the models. Recent games were included in the test set.

Next step was focused on constructing performance measures for the players based on in-game statistic. Initially, each row of the dataset corresponded to one game - linked to a match Id - with information on the 10 players that participated to the match. We created a new transformed dataset so that each row contained only the information of one player in one game (by moving columns in rows).

Now that we had information based on players, our goal was to aggregate the performance statistics of each player. After analyzing the data for each player, some inconsistencies appeared regarding our features. Every player didn't play the same number of games or the same amount of time, so our in-game performance statistics such as number of kills, number of deaths, number of assists, etc. were not comparable. To deal with that, we simply divided these features by the variable duration to get new features such as deaths per min, kills per min, etc. At this point we had six variables for each players: gold per min, xp per min (i.e. experience points per min), kills per min, deaths per min, assists per min, last hits per min. Next, the mean values were calculated for each player.
In order to ensure that created features do not cause multicollinearity of the model the correlation matrix was built. According to it constructed features are not correlated and can be include in modeling.

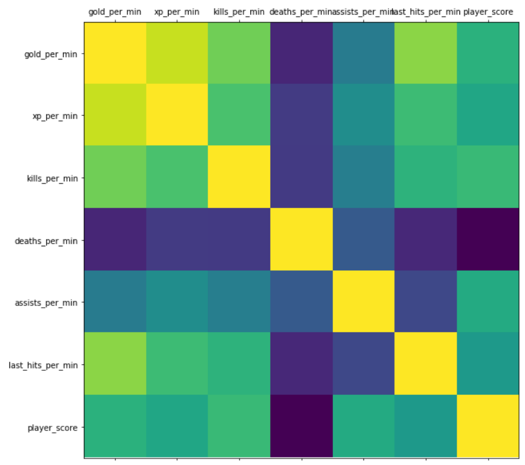


Figure 5: Player metrics correlation plot

Among the players there is one player '4294967295' associated with "unknown" in cases when players didn't want to share their real ID. In the 3D figure it is represented with the black star in the top of the figure. This player ID, linked to more than 40,000 games, actually represents the missing IDs of our dataset.

The training set does not have any missing values due to the way it was collected. However, the test set can have missing values for players that were not recorded in the train set and, thus, don't have performance metrics. In this scenario the "unknown player" ID came to be useful as a representative of the new players.

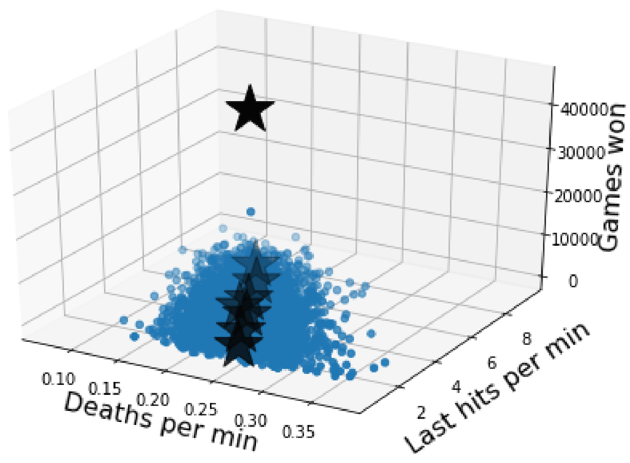


Figure 6: Visualization of unknown player

This was possible because the large number of games linked to that player ID made its statistics representative.

Next we performed modification of ordering players according to their performance based on hypothesis that not having player organized will lead to high variance when trying to create a predictive model.
We have ten columns for players (five in each teams), and then ten columns for each player's features. The players were assigned a column by Steam when we extracted the dataset, but the API documentation did not mention any reason for the specific ordering of the players, and so it seems that they were ordered arbitrarily. Thus, given that for each row in our dataset there are $(5!) * 2 = 240$ different combinations of players that would yield the same results(it could even be $(5!) * 2 * 2 = 480$ if we have teams switch their Radiant/Dire side), we were afraid that our newly created dataset would display a high variance. For most models, like logistic regression, each column would get a specific coefficient. Thus, given that our players can be organized in 240 different ways, our model would have different parameters depending on how the players would be organized in the training set, which would lead to a high variance.

So as to avoid the expected variance, we decided to create a new dataset where we would organize the players in each team in a specific order. Our first idea was to organize the players per role. In Dota, each players has a specific role in the team, thus by ordering the players in their respective role, we would lower variance by having parameters related to each role when training our model. Unfortunately we were not able to implement this idea. There are different teams compositions, and players can change roles from one game to another, so given the constraint we had to drop this idea, but revisit it in future works.
Our second idea for ordering players was to give each player a score given their past performances, and order them from the best player to weakest player in each team. Having such ordering should lower variance as best players and weakest player help their teams in different manners, and thus having our players ordered in such a way should reduce the variance of our parameters.
This method was simpler to establish, but came with some issues. Our dataset dis not include all the games a player played, thus the score we will give a certain player had a chance to be inaccurate, especially for players for whom we only have one or two games of informations on them. However, given that our dataset was created by extracting the last hundred games played by the top players, we still expected to have a good score for them.
To score our players, we created a new "player score" variable. This variable was created through this formula:

$$\frac{\text{Kills per min} + \text{Assists per min}}{\text{Deaths per min}}$$

Some players had a Death per min of zero, which lead to infinity results in our pandas dataframe. We decided to leave these values as no death showed great skills, and we were still able to sort the players with the infinity score. We then used numpy arg.sort command to find the order each players in the Radiant and Dire team should have to follow the strongest to weakest order we were looking for. Once every modifications on the player data were finished we transposed back the dataset so that every row would be related to one match with 10 players and their associated performance metrics, which now are representing the whole past experience of the player. The final dataset contained:

- The Match ID.
- A dummy variable for Radiant win.
- A column for the ID of each player.
- Gold per min for the ten players.
- XP per min for each player.
- Kills per min for each player.
- Deaths per min for each player.
- Assists per min for the ten players.
- Last hits per min for the ten players.
- Player score for each player.

After the creation of the dataset, we proceeded with the creation of our model.

5 METHODOLOGY

The main goal is to predict the outcome of the game. Therefore, the target variable is a binary variable, where

- "1" - stands for Radiant win;
- "-1" - stands for Radiant lose (Dire win).

For for the binary classification problem we choose Logistic regression as our first choice. But we were also interested in applying SVM as a more powerful technique. Then we will compare the models using accuracy score and confusion matrix.

Logistic regression
Logistic regression predicts the winner of the game based on the probability of the each label ("Radiant" or "Dire") for a given match, and then simply choosing the one with larger probability as its

prediction. It is a reliable technique for prediction in case with binary target variables and independent input variables. With the use of logistic regression we can also define significant variables making the most impact on the game result. Thus, the variables death per min and assists per min play the most important role. Moreover, it is relevant for both teams but with the opposite sign. On the contrary, gold per min and xp per min have coefficients close to 0 meaning that they are not very influential.

Support vector machine
The other model we chose to use was the Support vector machine model. SVM is an effective tool for classification in high dimensional features spaces which is the case in our project. By applying SVM we had the possibility of choosing between various kernels. We tried the Gaussian kernel and the linear kernel.

Other trials
In order to improve models we tried different combinations:

- Clustering: using K-means we tried to detect different groups of players based on their experience (prior to ordering features). However, since different features didn't have the same pattern the algorithm resulted in only one cluster of players;
- Dimensionality reduction with SVD and Logistic regression with feature selection. In order to try to reduce the number of input variables (number 60) we tried these two techniques. However thanks to prior work (data collection and feature construction) these measures appeared to be redundant: they didn't improve models performance.

6 EVALUATION

In order to assess model performance we used ROC curve, AUC and accuracy score. Results are outlined in the table below:

	Log.regression	SVM (gaussian)	SVM (linear)
Score (train)	71, 9	99,9	71,6
Score (cv)	-	59,8	-
Score (test)	53,0	58,5	53,0
AUC (test)	51,7	54,7	51,6

As we can see all models perform good in the training set. However, applying the to the test set reduces accuracy significantly. This can be explained both by the fact that only previous games are used for prediction and by presence of new players.
Logistic regression gave the expected result close to 50 per cent of accuracy. Since the data has a complex structure and contains a lot of features there is no clear boundary between winning and loosing teams.That is why many games were misclassified. On the contrary, using SVM with gaussian kernel clearly caused overfitting (99,9 per cent of fit on train set), which, unfortunately, made the model perform significantly lower on the test data.

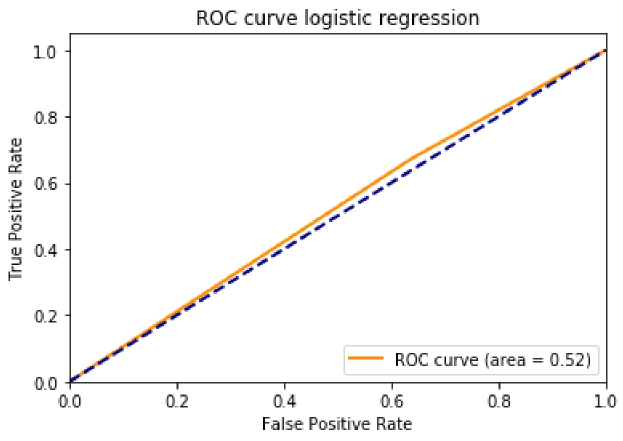


Figure 7: ROC curve for Logistic Regression

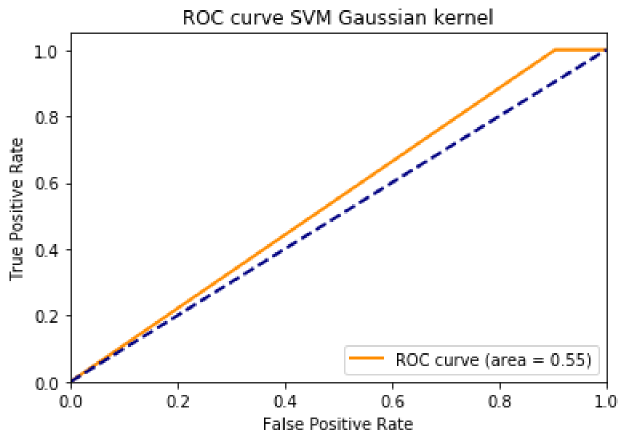


Figure 8: ROC curve for SVM with Gaussian Kernel

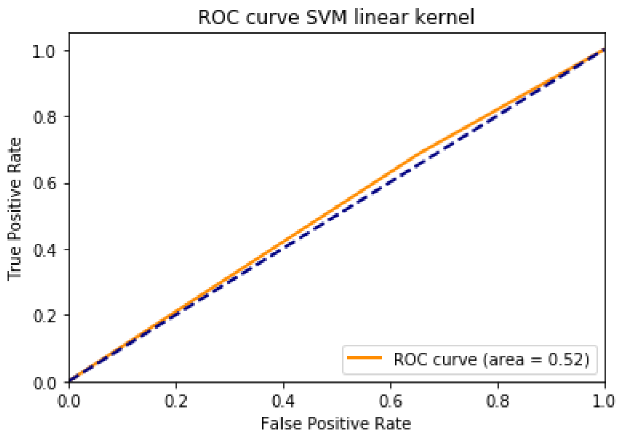
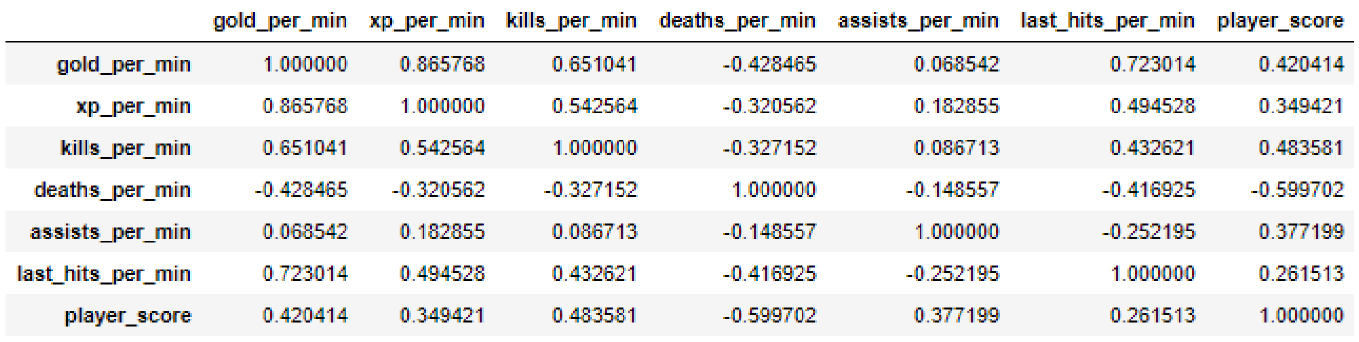


Figure 9: ROC curve for SVM with Linear Kernel

Interesting results are observed in confusion matrices.Both models fail to predict game losses resulting in high number of false



	gold_per_min	xp_per_min	kills_per_min	deaths_per_min	assists_per_min	last_hits_per_min	player_score
gold_per_min	1.000000	0.865768	0.651041	-0.428465	0.068542	0.723014	0.420414
xp_per_min	0.865768	1.000000	0.542564	-0.320562	0.182855	0.494528	0.349421
kills_per_min	0.651041	0.542564	1.000000	-0.327152	0.086713	0.432621	0.483581
deaths_per_min	-0.428465	-0.320562	-0.327152	1.000000	-0.148557	-0.416925	-0.599702
assists_per_min	0.068542	0.182855	0.086713	-0.148557	1.000000	-0.252195	0.377199
last_hits_per_min	0.723014	0.494528	0.432621	-0.416925	-0.252195	1.000000	0.261513
player_score	0.420414	0.349421	0.483581	-0.599702	0.377199	0.261513	1.000000

Figure 10: Correlation table of features

negatives. However, the SVM algorithm managed to predict correctly all the winning games. In general models are more likely to predict Radiant win.

Logistic regression

	-1	+1
-1	409	718
+1	436	893

SVM

	-1	+1
-1	108	1019
+1	0	1329

7 CONCLUSION

This project is in continuity with an important literature on sports and games' predictions. Predictions of sports generally hold somewhat low accuracy results, and it is commonly believed that there is a glass ceiling around 75% of accuracy, therefor, our results are standards for such a problem. But even if we did not arrived at incredible results, this project came with some interesting insight.

Our best model was a Gaussian SVM which only yield 58.5% accuracy, but if we looked at it's AUC score and its confusion matrix, we realize that the model actually has probably an even lower general accuracy. When looking at the confusion matrix we noticed that we have an important number of false positives, and no false negatives, which was related with a really low number of negative predictions. As we mentioned earlier in this paper, we assumed that starting the game on the Radiant or Dire side does not affect the chance of a team to win a game. Our initial dataset was fairly balanced in that way, but not completely: 54% of the games were won by the Radiant team, and 46% were won by the Dire team. Thus, our dataset was slightly unbalanced towards Radiant wins. The lack of false negative predictions showed that our model generally gave the most common value (Radiant wins) as the prediction to most games, but was able to accurately identify some conditions for a sure Dire win. This shows that our model has a heavy bias, as it was only able to capture the condition for a Dire win in a small number of cases, and lacked the necessary information to successfully divide the rest of our data. So our false negative number of zero shows that our model does capture some of the information necessary for a win, but also lacks a lot of information, and thus more data as well as the creation of more variables are needed to reduce our prediction bias.

To reduce the bias of our model, in future work we want to double our dataset by creating two records of each game, one version where we consider one team as the Radiant team, and a second version where the same team is considered the Dire team. This would not only increase our number of observation, but would also negate any bias our dataset would have towards Radiant or Dire win. Another way to reduce our model bias, would be to create variables related to teams, such as how often the teammates of one team played together.

Apart from our model bias, our model also displayed an important variance. Our training score was of 99.99 while our test score was 58.5 . To lower the variance, one approach would be to reduce our number of features. We could, for example, take away some of the variables that are highly correlated with each other, such as gold_per_min and xp_per_min as we can see in the correlation matrix (fig. 10). However we believe that in this case, this would not be the right approach. Because we have ten players which could be ordered in a lot of different ways inside the dataset, and given that the order of our features depends on the order of the players, this project was bounded to display variance in its results. Establishing a way to order the players did improve our models accuracy and lowered our variance by a little, but not enough. The change in variance seems to indicate that an ordering of players in the dataset might be the best road to follow as to lower the variance in future works. We believe that better results could be obtained by obtaining more data on each players, which would lead to a better scoring system and consequently a better ordering of players. Other ordering method such as ordering players given their specific strategic role would also yield interesting results.

REFERENCES

[1] K. Conley and D. Perry, How Does He Saw Me? A Recommendation Engine for Picking Heroes in Dota 2, tech. rep., 2013.
[2] Song, Kuangyan, Tianyi Zhang, Chao Ma Predicting the winning side of DotA2. CS229 2015.
[3] Petra Grutzik, Joe Higgins, Long Tran, Predicting outcomes of professional DotA 2 matches. CSS229 2017
[4] Zhengyao Li, Dingyue Cui, and Chen Li, Dota2 Outcome Prediction.
[5] Aleksandr Semenov, Peter Romov, Kirill Neklyudov, Daniil Yashkov,and Daniil Kireev, Applications of Machine Learning in Dota 2: Literature Review and Practical Knowledge Sharing
[6] Kevin Conley, Daniel Perry. How Does He Saw Me? A Recommendation Engine for Picking Heroes in Dota 2, 2013.