Blue Team Wins in League of Legends

Claudia Lyu, Dhruv Chokshi, Leona Du, Summer Wang

Introduction

The problem we are tackling concerns the game League of Legends. This is a MOBA where the blue team and red team, each with 5 players compete with each other. Players use gold and experience to get stronger with gold being obtained through killing minions, jungle monsters, and opponents. The problem being tackled is the win probability of the blue team in a ranked high ELO game, Diamond 1 to Master rank, given statistics of both teams at 10 minutes. This question was motivated by seeing the win probability graphs during pro games and wondering how they got these percentages based on the game state. This problem is interesting since a League game has so many things going on at a time, and it is almost always the case that one team will have the advantage in one area while the other is winning in another category. It is important that the data looks at ranked games in a higher skill bracket since people play more seriously and games are more predictable, as people care about winning and know how to use any advantages. When tackling this question, we can also determine which factors in the game are most important and influence win probability most.

Data

We accessed the dataset we used from Kaggle. The dataset contains around 10k unique games with each game correlating to a gameID, 19 features per team, giving 38 in total, and a column blueWins that contains the target value of 1 meaning the blue team wins and 0 meaning blue team loses, or red wins. This dataset does not contain any NULL values and is already in csv format, so no pre-processing needs to be done in these regards. These games are all ranked soloqueue games in the Diamond 1 to Master ranks, thus all being around the same skill level and game type which ensures some consistency. Since there is a winner and loser each game and blue and red side are randomly chosen and the game is designed to be equal for both sides, we will only be looking features concerning the blue team. Additionally, not all 19 features are very important to the game and many are correlated, so we will decide in the following steps which features to retain.

Pre-processing

We first look at the all the blue team variables and importance of each of them given by random forest. We immediately see that blueGoldDiff and blueExperienceDiff are far ahead the other ones. This makes sense as these two are the only variables that compare the two teams. Since these two variables contain information on total gold and total experience for both teams, it would be redundant to include variables that are purely based on gold or experience. Thus, we should not use blueGoldPerMin, blueTotalGold, blueTotalExperience, blueTotalJungleMinionsKilled, blueCSPerMin, blueTotalMinionsKilled, and blueAvgLevel. Additionally, in the first ten minutes of the game, warding, elite monsters, dragons, and towers destroyed are not critical due to the time frame being simply too small and it being hard for either team to do much of these things. This is also seen through blue EliteMonsters, blueDragons, blueHeralds, blueWardsPlaced, and blueTowersDestroyed to be considered to have little importance. Thus, we won't use these either. This leaves blueDeaths, blueKills, blueFirstBlood, and blueAssists. First blood is generally not considered critical because it is one small instance and does not have too much of an impact. Getting a kill or dying can lead

to many other things in the game, thus they are not just purely gold or experience and should be included. However, assists are inherently included in death and kill numbers, so we should not include this. The 4 predictor variables we will be looking at are blueKills, blueDeaths, blueGoldDiff, and blueExperienceDiff. Our response variable will be blueWins, the binary target variable indicating whether blue won or lose.

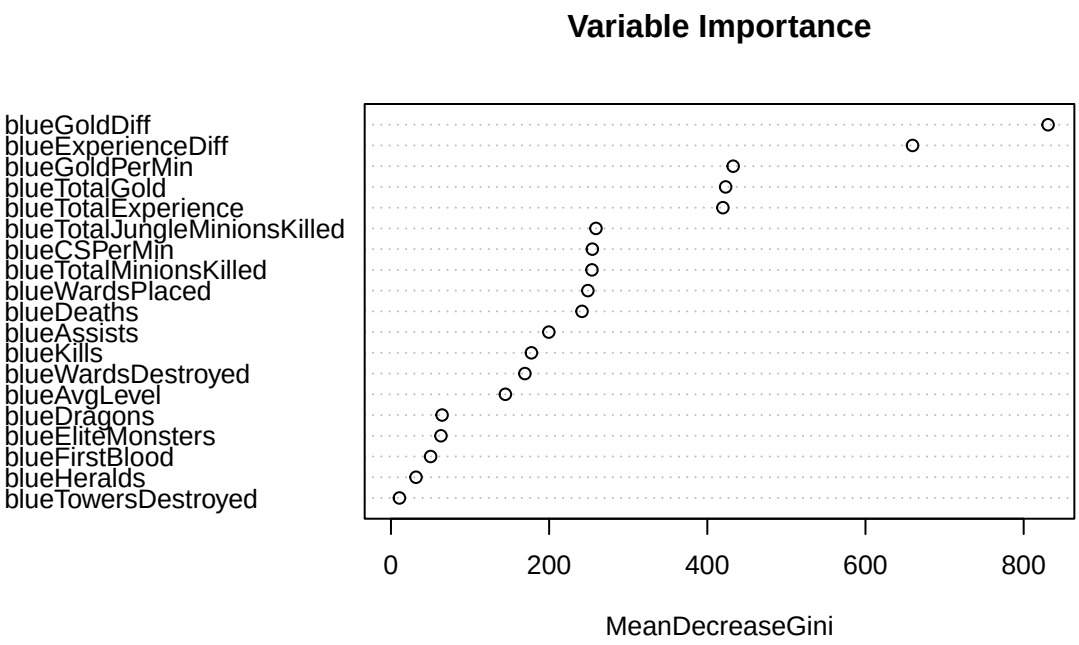

Figure 1: Variable importance for blue team

EDA

We now perform some exploratory data analysis on the 4 predictor variables chosen. We first look at the histograms of each of them. They all appear to be bell curved and around normally distributed, which is good. We next look at the box plots of the predictor variables with the response variable of whether blue team wins. We see from the boxplots that these variables do seem to impact whether the blue team wins or not with more kills, higher gold difference, and higher experience difference yielding more wins and more deaths giving more losses, which makes sense. Lastly, we check to make sure that the 4 predictor variables are not too correlated with each other. The correlation plot show some correlations between them, but this is not too alarming because if a team is super ahead in one area it makes sense for them to be ahead in a different one as well. We should still use all 4 of these variables as they show different things and have different impacts on the game.

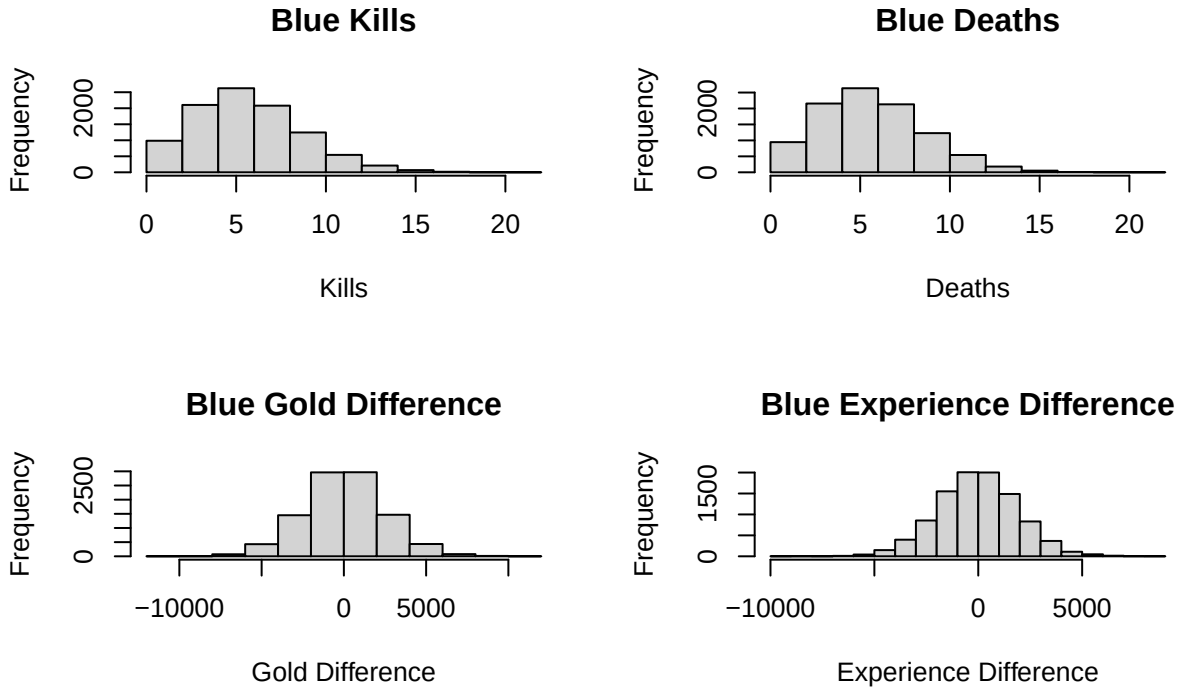

Figure 2: Hisograms of Blue kills, deaths, gold difference, and experience difference.

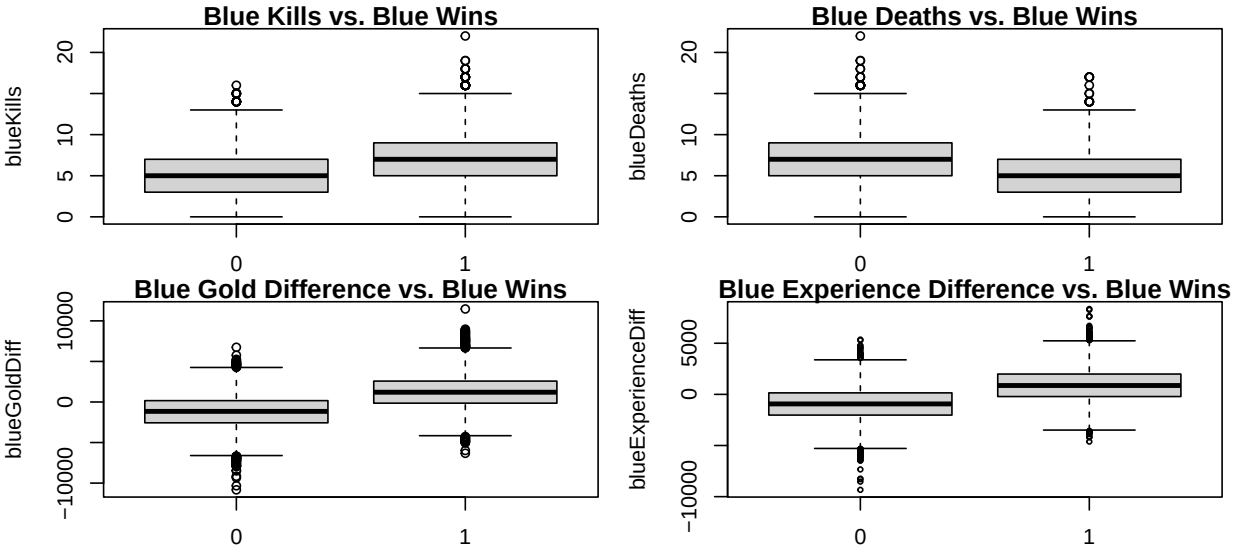

Figure 3: Boxplots of Blue Wins v.s. Blue kills, deaths, gold difference, and experience difference.

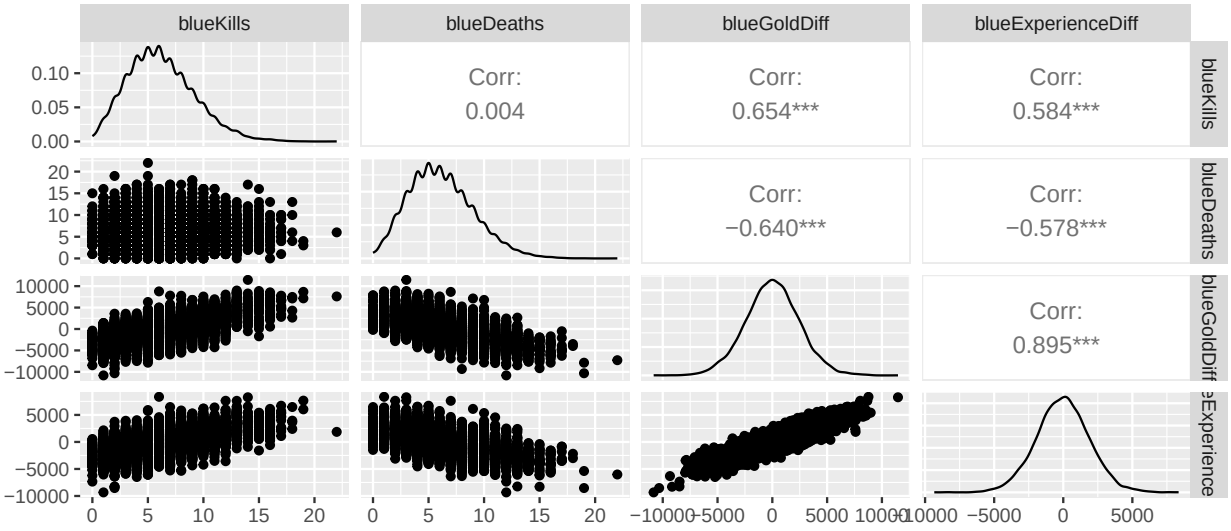

Figure 4: Correlation plots of interested variables, including blueKills, blueDeaths, blueGoldDiff, and BlueExperienceDiff

Methods

Statistical Modeling Technique

Logistic regression is a statistical modeling technique used to predict the probability of a binary outcome based on one or more predictor variables. With the interest in predicting whether the blue team wins (`blueWins` = 1) or not (`blueWins` = 0) based on early-game (first 10 minutes) performance metrics, including `blueKills`, `blueDeaths`, `blueGoldDiff`, and `blueExperienceDiff`, this project fits a multiple logistic regression model to estimate the win probability of a game for blue team as a function of number of kills of blue team, number of deaths of blue team, gold difference between blue and red team, and experience difference between blue and red team. More specifically, we model the log-odds of a win for blue team with a linear model:

$$\log\left[\frac{Pr(\ blueWins = 1|\ blueKills,\ blueDeaths,\ blueGoldDiff,\ blueExperienceDiff)}{Pr(\ blueWins = 0|\ blueKills,\ blueDeaths,\ blueGoldDiff,\ blueExperienceDiff)}\right]$$
$$= \beta_0 + \beta_1 \cdot blueKills + \beta_2 \cdot blueDeaths + \beta_3 \cdot blueGoldDiff + \beta_4 \cdot blueExperienceDiff$$

The assumption of our model is that the log(odds) is a linear function of `blueKills`, `blueDeaths`, `blueGoldDiff`, and `blueExperienceDiff`. We're going to create an empirical logit plot to assess the linearity assumption of our logistic regression model. Specifically, for each predictor, we make a plot with the empirical logit values along the y-axis against each predictor (via the midpoints of the bins) on the x-axis. For each plot, we also add a smooth line to fit a flexible trend to help us determine if the relationship is linear or not.

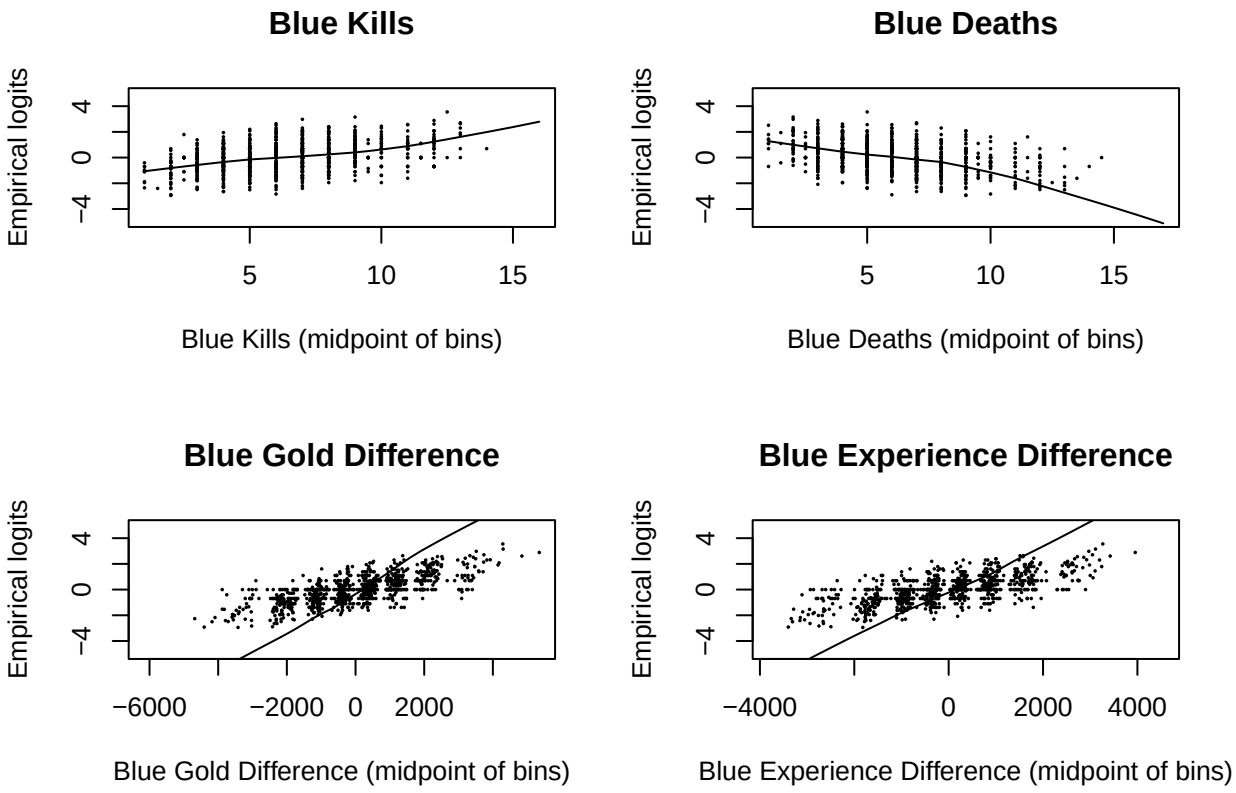

Figure 5: Empirical logit plots, where the empirical logit values are along the y-axis against each predictor (via the midpoints of the bins) on the x-axis.

The empirical logit plots look generally reasonable in terms of indicating the linear relationships between the log(odds) of a win for blue team and the predictors, including `blueKills`, `blueDeaths`, `blueGoldDiff`, and `blueExperienceDiff`. We see that the log(odds) decreases as `blueKills`, `blueGoldDiff`, and `blueExperienceDiff` decreases, i.e., the probability of a win for blue team decreases as the number of blue team kills, the gold difference between blue and red team, and the experience difference between blue and red team decreases; in contrast, we notice that the log(odds) decreases as `blueDeaths` increases, i.e., the probability of a win for blue team decreases as the number of blue team deaths increases.

With the assumption of a linear relationship satisfied, we finally fit the logistic regression model using the `glm()` function in `R`. In this project, although we are technically modeling a variable (`blueWins`) that follows the Bernoulli distribution, we set `family = "binomial"` to fit the logistic regression model (since it's a Binomial distribution with $n = 1$).

Model Evaluation and Comparision

After building the model, we also evaluate the model performance by creating the confusion matrix along with performance metrics, particularly accuracy score. We will randomly assign 60% of data to training data and 40% of data to testing data. The confusion matrix along with performance statistics is useful here given that we are doing a classification problem, which provides a tabular view of the model's predictions compared to the actual values.

First, we build the confusion matrix of our original logistic regression model which predicts the probability of blue team wins by all four features, along with the model accuracy score:

Similarly, we use accuracy score to compare our original logistic model to an alternative model with different predictor variables and evaluate which model is better based on the accuracy score. As shown in Data Section, `blueGoldDiff` appears to be the most important variable associated with if blue team wins the game, so one alternative model can be a logistic regression model of `blueWins` on `blueGoldDiff` only. Again, we build the confusion matrix of the alternative logistic regression model which predicts the probability of blue team wins by only `blueGoldDiff`, along with the model accuracy score:

Table 1: Combined Confusion Matrices and Accuracy Values

Method	True_Positive	False_Positive	Accuracy
Full	1487	540	0.731
Partial	1483	561	0.725

Comparing the accuracy of two models, we notice that the accuracy score of our original regression model (0.731) is higher than the accuracy score of the alternative model (0.724). It suggests that we may still keep our original logistic regression model of `blueWins` on `blueKills`, `blueDeaths`, `blueGoldDiff`, and `blueExperienceDiff`, which has a slightly better performance and is straightforward to interpret how these related and interesting features affect the wins of blue team.

Quantifying Uncertainty

The data in this project contains game data in the first ten minutes of League of Legends, so it's necessary to quantify the uncertainty in our estimates, which helps us better understand the variability in how these predictors relate to if the blue team wins. Considering the data structure, we predict the response `blueWins` (binary) by four numeric predictors under a logistic regression model with the assumption of linearity. Again, although the linearity assumption appears to be met, it's still helpful to quantify the uncertainty of coefficient estimates to measure if our model is reliable. To quantify the uncertainty, we use case bootstrapping (i.e., resampling of games) to find 90% confidence intervals for the coefficient estimates of the logistic model.

Table 2: Case-bootstrapped 90% confidence intervals for the coefficient estimates and average predicted probability of blueTeam win for the logistic model.

	lower	main	upper
(Intercept)	-0.078	0.037	0.17
blueKills	-0.0011	0.025	0.05
blueDeaths	-0.061	-0.031	-0.003
blueGoldDiff	0.00033	0.00038	0.00043
blueExperienceDiff	0.00019	0.00024	0.00029
Mean predicted probability of blueTeam win	0.49	0.5	0.51

Providing a measure of uncertainty around the estimates from our model, all 90% confidence intervals (CIs) indicate that we are 90% confident that the true value of the given estimate falls within this interval. It appears that with higher number of blue team kills, larger gold difference between blue and red team, larger experience difference between blue and red team, and lower number of blue team deaths, it's more likely for the blue team to win. Additionally, the CIs of `blueDeaths` ([-0.057, -0.0043]), `blueGoldDiff` ([0.00032, 0.00043]), and `blueExperienceDiff` ([0.00019, 0.00029]) do not contain zero, meaning that these predictors are statistically significant in predicting if blue team wins. However, the CI of `blueKills` ([-0.0015, 0.05]) contains zero, meaning that it may not be statistically significant in predicting the outcome (future work may consider replacing `blueKills` with something more significant). Moreover, the CI of the mean predicted probability of blue team win from our model is [0.49, 0.51], suggesting a nearly 50% of chance to win for blue team. Overall, after quantifying the uncertainty, we find that our logistic regression model is robust to estimate the probability of blue team wins, offering valuable insights for strategic decision-making in League of Legends.

Results

Our logistic regression model offers crucial insights into predicting the blue team's probability of winning in League of Legends based on early-game performance metrics. By examining blueKills, blueDeaths, blueGoldDiff, and blueExperienceDiff, we can anticipate match outcomes, providing valuable guidance for esports strategies.

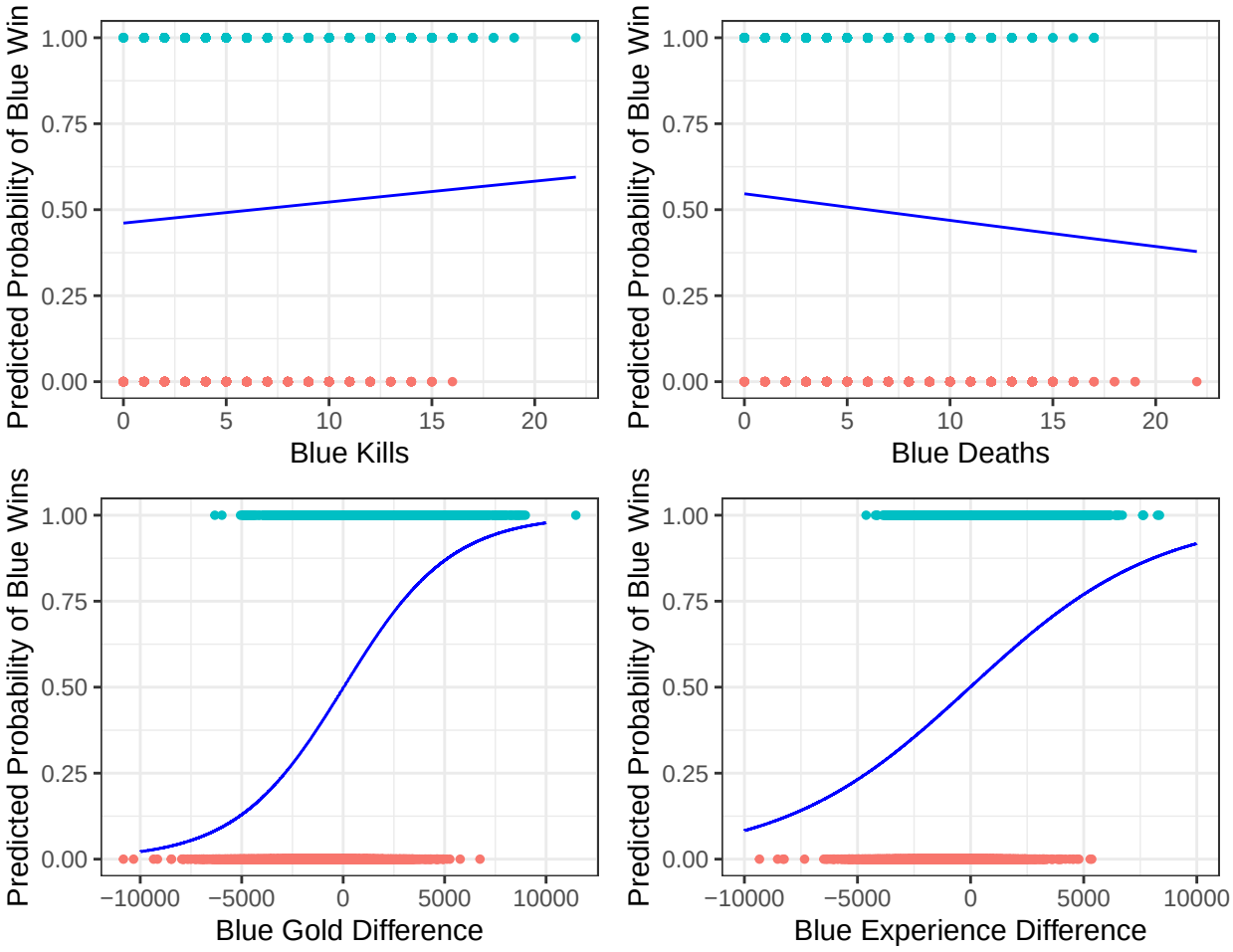

Figure 6: Logistic Regression Predictions of Blue Team Victories.

Figure 6 illustrates the predicted probabilities of blue team wins against each predictor variable: blueKills, blueDeaths, blueGoldDiff, and blueExperienceDiff. These subplots visually depict the relationship between predictor variables and the likelihood of the blue team winning, offering valuable interpretive insights.

Our analysis highlights that higher blue team kills, fewer blue team deaths, larger gold and experience differences, favor the blue team's chances of winning. These findings align with gaming intuition, where securing more kills, minimizing deaths, and gaining significant resource and experience advantages often lead to victory.

In evaluating our logistic regression model, we utilized 5-fold cross-validation to assess its out-of-sample performance. The resulting root mean squared error (RMSE) of 1.0864, with a standard error interval of [1.0804, 1.0923], reflects the model's average prediction error and variability across validation folds.

Comparing our model to an alternative focusing solely on blueGoldDiff, yielded an RMSE of 1.087, with a standard error interval of [1.0813, 1.0927]. Although our original model exhibited a slightly lower RMSE, the differences were minimal, suggesting comparable predictive performance. Hence, we retained our original logistic regression model, given its comprehensive consideration of multiple performance metrics.

To quantify uncertainty, we employed case bootstrapping to derive 90% confidence intervals for coefficient estimates in our logistic regression model. These intervals provide a range of plausible values for each predictor's effect on blue team wins. Notably, blueDeaths, blueGoldDiff, and blueExperienceDiff are statistically significant predictors, as their confidence intervals do not include zero. However, the confidence interval for blueKills contains zero, suggesting it may not be statistically significant in predicting outcomes. Additionally, the confidence interval for the mean predicted probability of blue team win spans [0.49, 0.51], indicating a nearly 50% chance of victory for the blue team.

Discussion

Our research determined, holding each variable constant for the blue team, having a large positive gold differential, a large positive experience differential, more kills, and less deaths improves the probability of winning. This data only contains information from Diamond ranked games at the 10 minute mark. The results we found directly connect to the development of analytics in the e-sports space. Information like this can help team understand their best options in given match. If the game appears lost based on the factors we have examined, a team can use this model and forfeit the match as to not waste time and continue ranking up in a new match. However, there is no way for a team to easy or immediately calculate the exact gold difference or experience difference for each team. Further research could examine how the different observable factors can act as metrics for a team's performance and chance of winning.

References

Ma, Y. (2020, May). League of Legends Diamond Ranked Games (10 min). Kaggle. https://www.kaggle.com/datasets/bobbyscience/league-of-legends-diamond-ranked-games-10-min

Shalizi, C. (2024, Feb). Chapter 6: The Bootstrap. http://www.stat.cmu.edu/~cshalizi/uADA/24/lectures/ch06.R

Yurko, R. (2024, Jan). Lecture 2 demo: Building an Expected Goals Model.

Yurko, R. (2024, Jan). Lecture 3 demo: Calibration and Cross-Validation.