Alkım Karakurt

Summary

Try to predict that blue team wins or lose the game with based on first ten minutes gameplay action metrics. Secondly, try to understand that how much minions killed by blue team until first ten minutes.

Statıstıcal learnıng wıth league of legends

Term Project | ADS542

çizim içeren bir resim

Açıklama otomatik olarak oluşturuldu

# Introduction

In last years, game sector developing too much and attract lots of people with different age ranges, especially younger people. In this period, e-sports start to be more important and sector grow up so much last 2-3 years. In all over world, there are most of e-sport player related with different games and successful ones earn so much higher than expected. Like the other sports, in e-sports people should work too much and improve themselves to protect their place at the stage. There is huge competition and it led to people focusses on different areas for improvement. Data science become very important for these competitive sports to create some difference than others. This journey starts in Baseball. This story described in Moneyball movie which is based on the book “Moneyball: The Art of Winning an Unfair Game”. After that data science applications used different sports the analyze performance of player or all team. In e-sports case, lots of team have their data scientist and they try to improve their games in data-driven solutions, especially in League of Legends.

League of Legends (LOL) is a MOBA game which is explained like “League of Legends is a team-based strategy game where two teams of five powerful champions face off to destroy the other’s base. Choose from over 140 champions to make epic plays, secure kills, and take down towers as you battle your way to victory.” There are 5 different roles in game which are Top, Jungler Mid, ADC and Support. Most of the champions usually playing only specific lanes. As their explanation, there is 140 champions with different features. Because of that, teams should be aware that which composition is better against their opponents. In the selection tame, data have huge importance for teams. They choose their teams based on meta[[1]](#footnote-1). Creating and understanding meta or solutions against meta champions can be found with data analysis for teams. They can find different compositions or gameplay actions with analyze data and create a specific difference in their gameplay metrics. This metrics can be described lots of different features which will be explained in data section of this paper.

In this project, I try to understand the first 10 minutes gameplay actions and their effects on the winning condition of teams. First 10 minutes is very important for League of Legends, because this is continuous game and it generally takes 35 minutes. With time passes, champions became more stronger and can increase their abilities. Because of that, most of the team try to don’t fail early times of game, even some team composition has a higher advantage of late game. This late or early game advantages changes with champions and team composition. Teams should choose their gameplay style based on chosen champions for every game. However, even they can stronger at late game, they should not fall in the early game. They should be equal in this period with their opponents. Because of that early game is very important for this teams and lots of teams have some plans to get an advantage on early game. This paper maybe helps them to win game with improve their early game plans. With this purpose, it is tried to predict that is blue team win or lose game with considering their team-based gameplay metrics with understanding the which metrics are important and how they are affecting teams’ win conditions. Secondly, I try to predict the total minions killed by blue team which has a very important in game metrics to understand that teams how much gold gain and experience. Minions are very important resources for teams. At the 10:00, most of the team expect their professional players at least kill 80 minions. This numbers can change with respect the performance of players. Also, unprofessional areas, these numbers can change by the rank of the player. The rank of our players is very good for unprofessional level because of that the killing minions is very important for analyze their success.

# Data

For this project, League of Legends Diamond Ranked Games (10 min) data is taken from Kaggle. This data includes high level lol matches between players in ranked games. This is important because the gameplay changes too much between ranks. In addition, higher level players can understand the game better than others and these matches can be better for analyzing because they always focus to win and play more carefully than lower level players.

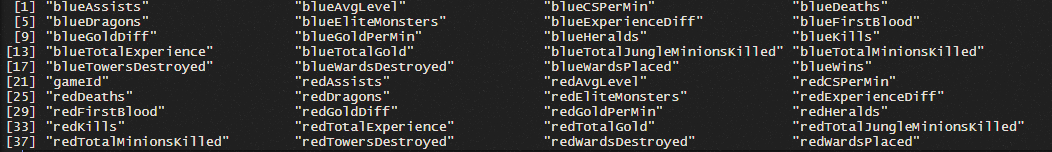
Data includes 19 features per team and 9879 different matches which are making unique with gameID. However, in data same features represent same values because of the type of game. For example, blueKills and redDeaths represent same thing and vice versa. Because red team only can be killed by blue team. In addition, first blood only taken by one team, so we do not need this variable for both teams. blueFirstBlood = 0 represent that red team take first blood. These are directly represented same thing for matches. Secondly, there are some other things to be fixed. Players can only increase their levels with getting experiences. Because of they, if we know the experience level of players, we can directly know that their level. Average level and total experience also represent same thing. Until the 10 minutes, only two elite monster occurs in a map which are heralds and dragon. Also, these elite monsters can be taken only once in first ten minutes. Because of that we do not need to blueEliteMonster and redEliteMonster variables in our dataset. In addition, blueCSPerMin, blueGoldPerMin, redCSPerMin and redGoldPerMin were dropped to data because they represent same things with total number. If we have different these variables for different periods until the 10 minutes, it can be usable. However, for this dataset, they only divided 10 from other variables which are represent their total numbers. In addition, gold difference includes same things again because one of them is positive and other one is negative because of that I drop redGoldDiff.

Figure 1: Names of Variables

The description of selected variables is like that:

* BlueWins: 0 represent loose, 1 represent win for blue team.
* bluewardPlaced: How many wards placed until the firs 10 minute. It is very important the protect themselves from jungle ganks. Wards can increase the vision at the map and shows unseen places to players.
* blueward Destroyed: Number of enemy warding totems the blue team has destroyed.
* blueFirstBlood: It is factor variable and very important for the teams to get an early lead. For some champion it can be huge effects on winning conditions. Most of the teams have plan to get first blood.
* blueKills: Represent the number of kills. Kill can increase the gold and experience of players.
* blueDeath: Number of deaths. Death can cause the loose the experience and golds with loosing minions. Also, it can increase the power of opponent.
* blueAssists: Assists can increase gold and experience, but lower than kills.
* blueDragons: Dragons are elite monsters in map. When one team kill the dragon, they can have some specific increase in players' features like higher movements speed or increase in armors. Occurs at 05:00
* blueHeralds: Rift Herald can give the huge damage the enemies towers. It is occurring at 8:00
* blueTowersDestroyed: The number of towers destroyed by blue team. Towers can increase gold and experience. Increase vision at the map
* blueGoldDifference: The total difference between two team at 10 min.
* blueExperienceDifference: Experience is important to increase skills levels. In early game, even small experience differences can have huge effects.
* blueTotalMinionsKilled: When player killed an enemy minion, player gain a gold and experience. The basic resource in the map to improve champions.
* TotalJungleMinionsDiff: Differences between two team junglers. Created by blueTotalJungleMinionsDiff - redTotalJungleMinionsDiff
* redWardsPlaced: How many wards placed by opponent team.
* redWardsDestroyed: Number of enemy destroying wards by opponent. It is important because timing and placement of wards are very important.
* redDragons: They get dragon or not.
* redHerald: They get herald or not.
* redTowersDestroyed: number of towers destroyed by red team at 10:00
* redTotalMinionsKilled: When player killed a minion, player gain a gold and experience. Higher minions should have negative effect to blue team.

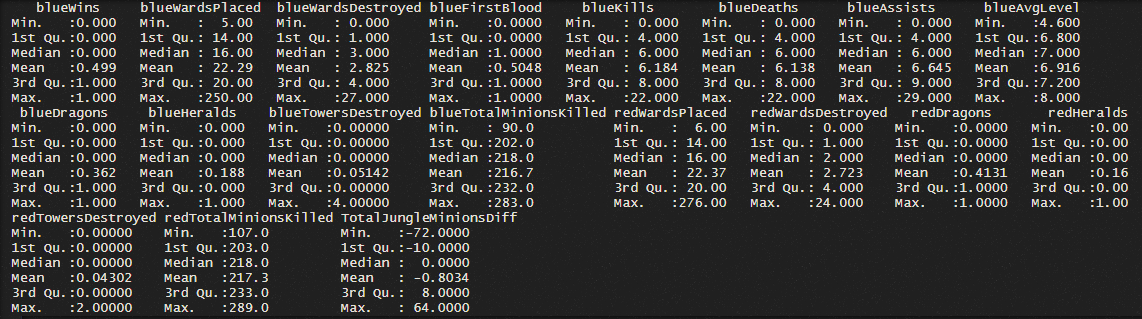
For this project we have two different part, classification and prediction. Then I should select a one categoric and one numeric variable for analysis. For classification part, blueWins variable is chosen because I think that it has most important variable for teams. To predict which team wins under certain conditions can help significantly to teams and players. For prediction, I choose blueTotalMinionsKilled because, minions are most important and basic resources for teams and players to developed. Minions waves coming continuously in game and even teams cannot take kills or other features, if can killed minions successfully, they can be better than opponent team.

Figure2 tell us the basic descriptive statistics for our dataset. When I check the summary of variable, I find out that, there are some misinformation in blueWardsPlaced and redWardsPlaced. It is impossible to places 250 or 276 wards until the 10 minutes in a game. I control lots of professional game and mathematics about that I understand that the higher number of 75 is not possible for warding at this game. It can only occur with some people who do not play game and just warding in the map for bad purposes. Even they do that, the 250 number is not possible. Because of that, I drop the matches which includes higher number than 75 wards placement. The other features have acceptable and reasonable means and other values. For example, in the first 10 minutes, it is hard to destroy towers and we see that the mean of this variable is 0,05. However, structure of some variables is wrong, and I need to fix this before the analysis. The below picture shows the type of variables and their levels. Even teams can take more than one dragon, it is not possible to take more dragon in first ten minutes and similar for heralds. Because of that, I change herald and dragon variables into factor.

Figure 2: Summary of Dataset

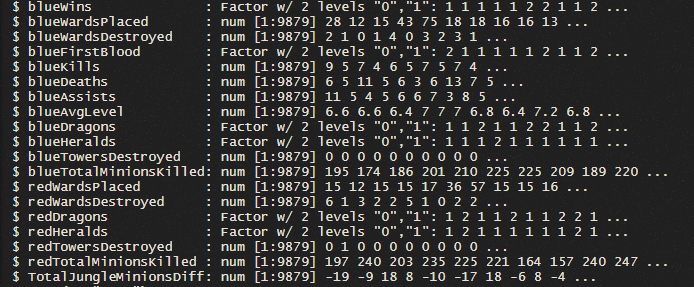


Figure 3: Type of Variables

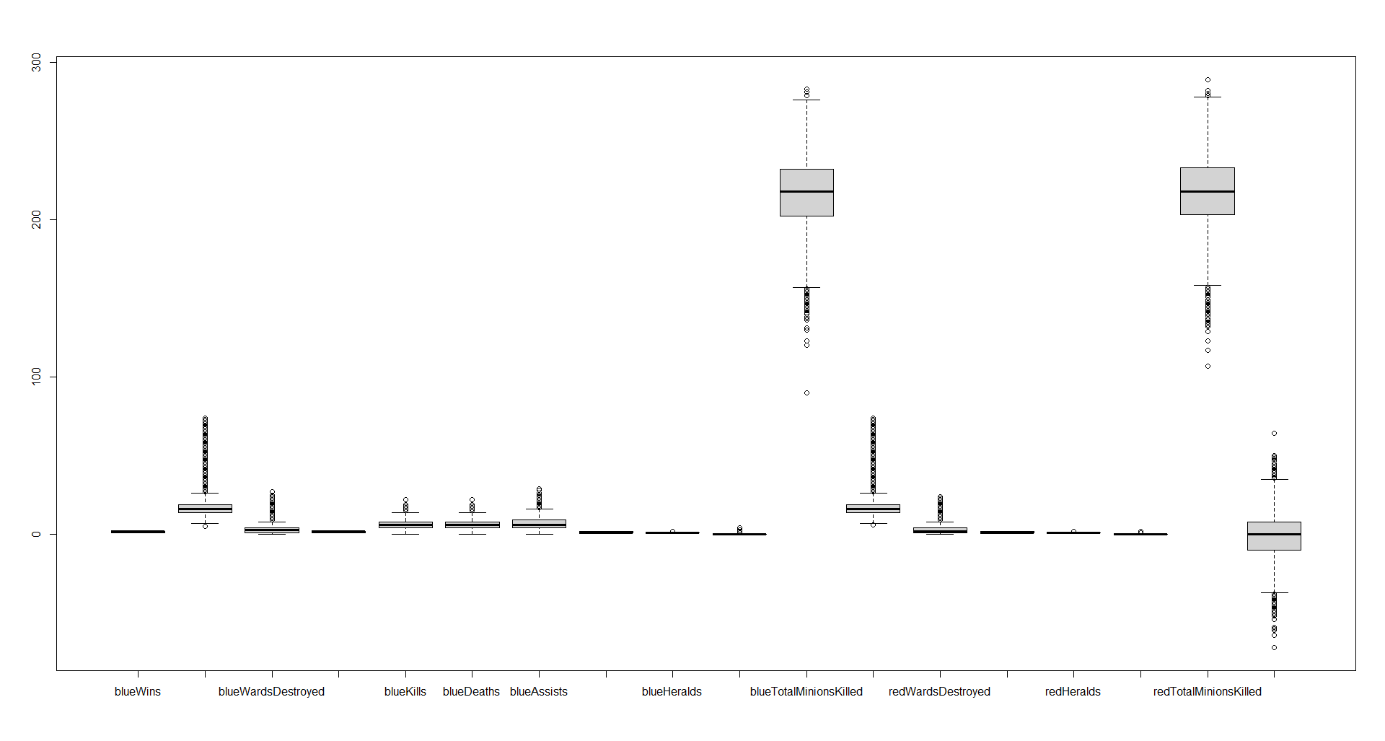


Figure 4: Boxplot of variables

After that, I check the distribution of variables. Even some of them have similar ranges, there is also very different ranges for our variables. In addition, even there are outlier in my dataset, these outliers can be important for our analysis because, some of the player can play better than their usual and worse. In addition, in some cases, they can get a very good early game or bad. Because of that I do not want to eliminate these outliers. However, before the analysis we also need to do some steps with respect this boxplot graph.

Before the analysis, data is divided two sample which are train and test to analysis. In this process data should divided randomly and factor variable can have same distribution in both datasets. In this sample proportion of blue team win or not is very close to 50%. BlueWins:0 = 0.501, blueWins:1 = 0.499. After splitting data, I need to use standardization this data. Standardization process is very helpful when we are comparing different things. For example, when we check the means of our variables from Figure2, it can be noticeable that there is some huge difference between means and lots of the variables explained differently. Because of that standardization must be applied this dataset to make sure that data is consistent and has same format. In this process, both data should standardize differently. First, train dataset standardizes with using their own means and standard deviation values. Secondly test dataset standardizes with using mean of train data and standard deviation. Formula can be represented like that: .

# Analysis

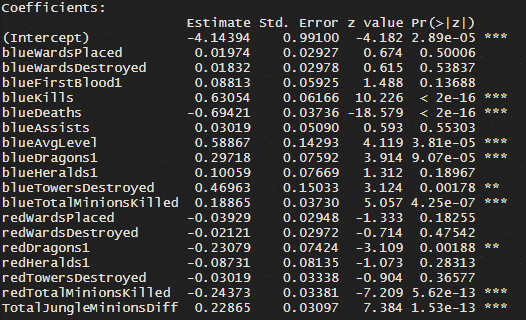
In this part, I try to explain my analysis about our dataset. Firstly, it will be explained to classification part with focusing on different algorithms to predict blueWins variable. Secondly, prediction part will be explained by focusing on blueTotalMinionsKilled with using different models.

## Classification

In this section, you can find the different machine learning algorithms and their performance measures. I mainly focused on Logistic Multiple Regression, Support Vector Machine, Decision – Tree, Bagging, Boosting and Random Forest. These methods are applied to find the predict that blue team wins or lose the game by using gameplay metrics. Finally, I try to decide that which model should have best performance measure and useful.

### Multiple Logistic Regression

In statistics, the logistic model (or logit model) is used to explain the probability of a class or event, such as pass or fail, win or lose, live or dead or healthy or patient. Logistic functions used for the understand the effects of independent variables on binary dependent variable. In this case, because of blueWins a categoric variable, I should use multiple logistic regression.

 In my first model, I check the relation between blueWins and all other variables. Figure5 represent the estimated standard errors and significant levels of our variables. However, this is not a good explanation for my dependent variables. This model includes lots of insignificant variables and the AIC value of this model is 7600. AIC is a good measurement for the understand that our model is good or bad. Akaike information criterion (AIC) (Akaike, 1974) is an in-sample harmonization-based fine technique to predict the probability of a model predicting / predicting future values. Then I should find the best model for my logistic regression. I need to decide which feature selection I used. However, besides that, I want to use stepAIC function from R which used forward and backward selection methods to find a model which have lowest AIC values.

After that, stepAIC functions gives us the best model for our analysis with lower AIC values. My best model for regression is

Figure 5:Model1

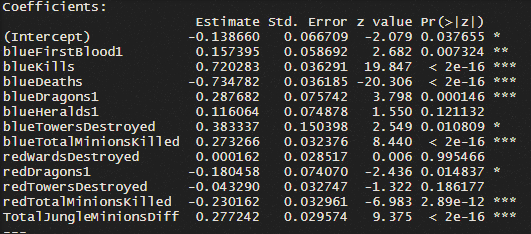


Figure 6: Best Model

The below table you can see that 8 variables have significantly important to blueWins. Which are blueFirstBlood1, blueKills, blueDeaths, blueDragons1, blueTowersDestroyed, blueTotalMinionsKilled, redTotalMinionsKilled, TotalJungleMinionsDiff. This is very fair to understand that these variables have significantly effective for winning game at early game. Unexpectedly heralds are not significant, but it can be related that even teams can take herald before 10 first ten minute, they do not have to use it at 10 minutes. Because of that we cannot see the effect of first herald in our model. In addition, all variable signs are representing the expectation from me. Obviously, kills, dragons, destroyed towers and higher minion kills affect our blue team winning condition positively. Also, death of blue team and variables which cause a improvement at red team have negative sign. However, to see that how these variables effect our dependent variable we should check their odds ratio in logistic regression

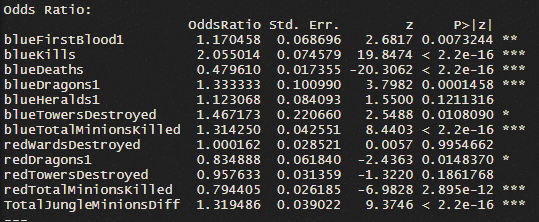


Figure 7:Odds Ratio of Best Model

However, we should check the multicollinearity problem in our model. To check multicollinearity, we use VIF values. The lower VIF values are represent the lower multicollinearity problem. The VIF values which are lower than 10 acceptable for analysis. In Figure 8, you can see that there is no multicollinearity problem.

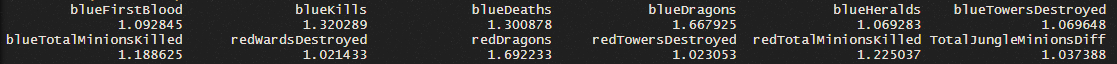


Figure 8:Multicollinearity

blueKills variable have a higher odds ratio with 2.05 value. This means that blue kills can increase the winning condition of blue team 2.05 times. Secondly, blueTowersDestroyed comes with 1.45 odds ratio value. Team who take early tower get a huge advantage because first tower gives them extra +600 gold. Also, when taking opponents tower, you can get a higher vision on map and narrow your opponent vision. The first dragon has a higher positive effect on dependent variable. It is very sensible because with taking dragons teams get permanent buffs for their teams. It is very effective to get an advantage against opponents. Deaths also have very low odds ratio. We can understand the importance of blue deaths to checking other negative sign variables. Even their odds ratio close to 1, the odds ratio of blueDeath is 0.48. It is also obvious that because when a player death, s/he loose experience and opponents gets gold and extra experience. In addition, redTotalMinionsKilled have a 0.79 odds ratio, when your opponent has successfully killed minions than your team, they get higher advantage.

After that, we should check the performance of this model both train and test data. It is expected that the accuracy of train data is higher than test because of we create model with using train data. To comparison I used confusion matrix for both datasets. First table of confusion matrix shows that how much our model predicts 0 and 1 values truly.

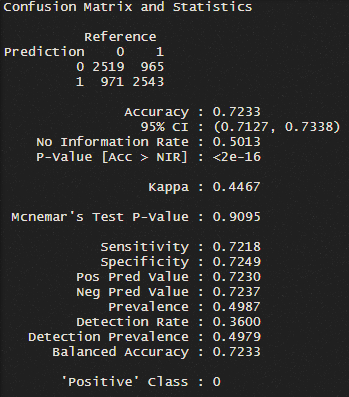
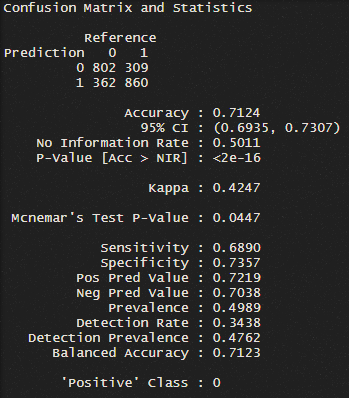


Figure 10:Confusion Matrix of Train

Figure 9: Confusion Matrix of Test

Figure 9 and Figure 10 represent confusion matrix of test and train respectively. In Figure10 you can see that accuracy of our logistic regression model is 0.7233. The accuracy level is not good as much as I expected but accuracy of train is very close the train even there it is decreased.

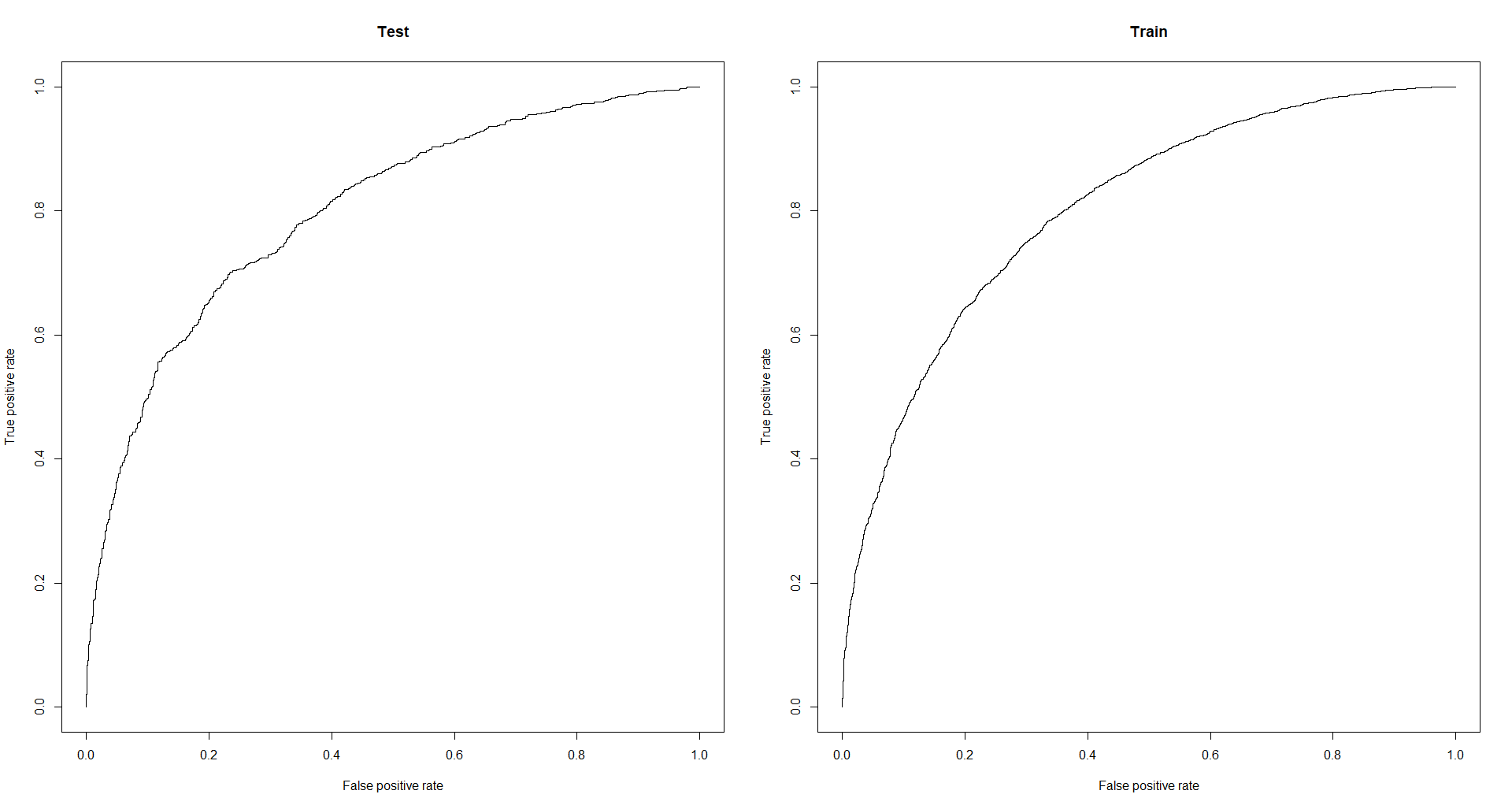


Figure 11:ROC

Figure11 represent the ROC curve of both datasets. Area under the roc curve which is represent as AUC (Area Under Curve) is positively related with performance of our model. Train AUC value is 0.8019263, and test AUC value is 0.799582. The lower decrease is very good for our test prediction.

### Support Vector Machine

Our second machine learning method is support vector machine. Support Vector is models with associated learning algorithms that analyze data used for classification and regression analysis. The purpose of the support vector machine algorithm is to find a hyper plane that classifies the data points separately in an N-dimensional area (N - number of properties).

Firstly, I should decide that which kernel should I use for support vector machine. Linear and radial kernels are mostly used kernels in these processes. When I check the all error values for these different kernels for train data, I found that lower one is polynomial.

|  |  |
| --- | --- |
| **Kernels** | **Error** |
| Linear | 0.2809374 |
| Radial | 0.2623607 |
| **Polynomial** | **0.2622178** |
| Sigmoid | 0.3535296 |

Then with using polynomial kernel, I tune out my first support vector machine model. First model has 4932 support vectors and one cost. The confusion matrix of Support Vector Machine, ROC curve and AUC values are shown below. Accuracy of train is 0.73 and it is decreased to 0.69 in train dataset. It is not too much decrease. However, the accuracy rates are generally lower than I expected for my model. Tune process gives same cost values to my model. Then I do not need to create a second model.

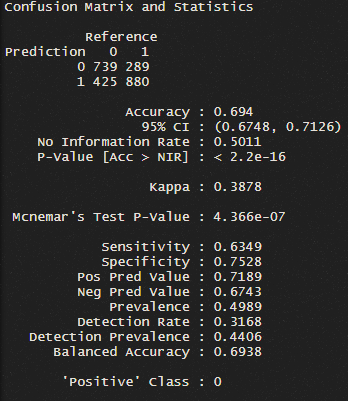
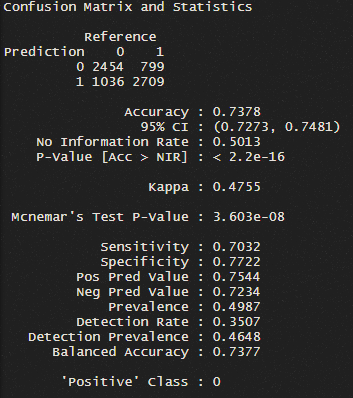
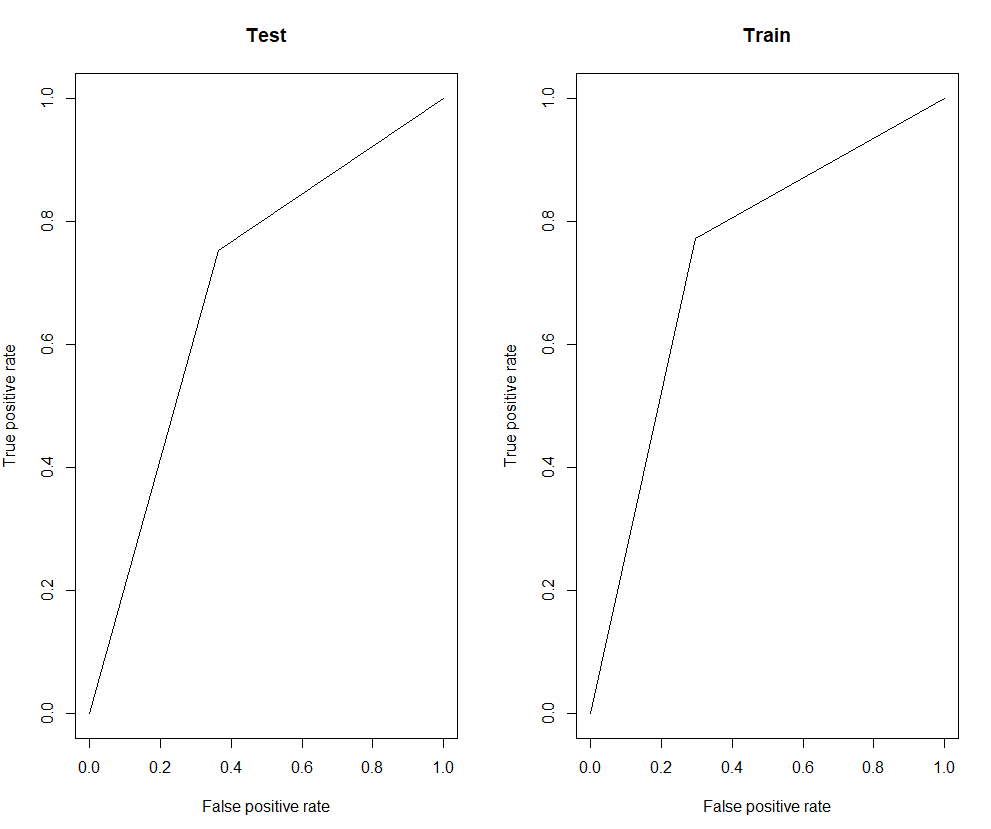


Figure 12:Confusion Matrix of SVM - Train and Test

|  |
| --- |
| AUC Train: 0.7376934  AUC Test: 0.6938299 |



### Decision – Tree

The decision tree is a tree-like decision model and a decision support tool that uses possible results, including chance results, resource costs and benefits. It is a way to display an algorithm that contains only conditional control expressions. Decision tree also shows that how our winning conditions changes by some certain point from our variables. My first decision tree has included 4 nodes and used only two variables for construction. These variables are blueDeaths and blueKills, which are the most effective variables in our logistic regression model. However, it is not enough to decide on this model because we should be sure about our nodes number using the cross validation. After I check cross validation, I see that, the node number is 4 and do not need to prune my decision tree model.

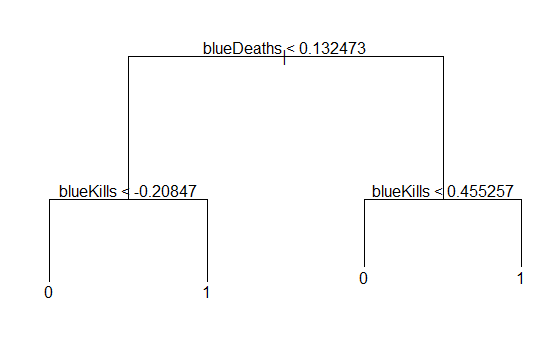
Firstly, these values are standardized values. This tree can be explained that if blueDeaths is lower than 0.13, blueKills should be higher than 0.45 to win game. If it is not, you lose the game. Secondly, if blueDeath is lower the 0.13, blueKills should be higher than -0.208 to win game. It explains that, with lower death rates, you do not need to higher kills to win game. If you do not die in the early game, you can win game easily in the late.

Figure 13: Decision Tree

Confusion matrix tables shows the accuracy and performance of our decision tree. Unexpectedly, my train accuracy is lower than test accuracy. However, the difference is so low. Positive predicted value increase in train dataset, however there is no significant changes in negative predicted value between them.

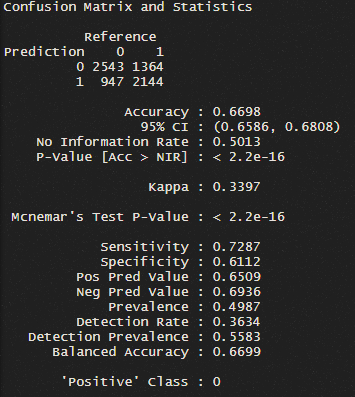
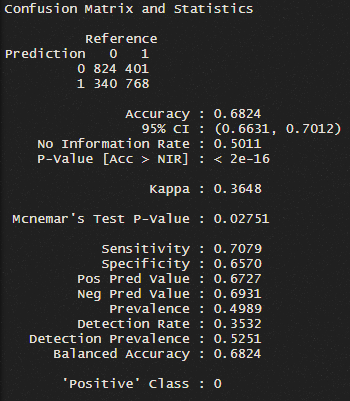


Figure 14:Confusion Matrix of Decision Tree - Test

Figure 15:Confusion Matrix of Decision Tree - Train

Secondly, I should check their ROC curve and AUC values to analyze performance of this model. Below graph and table shows the ROC curve and AUC test of Decision – Tree model.

Figure 16:ROC of Decision Tree

|  |
| --- |
| AUC(Test): 0.6824378  AUC(Train): 0.6699139 |

AUC value of test is 0.02 point is higher than train. We can expect it based on our accuracy values. Because of test data has higher accuracy levels and AUC values, I have to say that our decision tree model is not suitable for analysis. However, it can be related with the distribution of sample. I mean because of that we randomly selected our samples for train and test, if we change another sample by randomly, these results can change because the difference is not significantly important for these two datasets.

### Bagging

Bagging is short for “Bootstrap aggregating”. A group of machine learning algorithm subclasses where we use multiple weak models and collect the predictions, we get from each. It is designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression. It also reduces variance and helps to avoid overfitting.

My bagging classification model created by trees with 100 bootstrap replications and my out – of bag estimate of misclassification error is 0.3099. This is quite acceptable error rate for me because of the earlier tests. However, the error is not enough for understand the successful of bagging classification. We should check the test and train matrix.

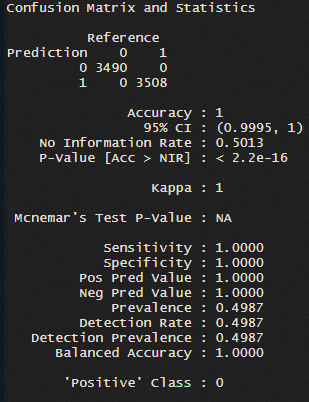
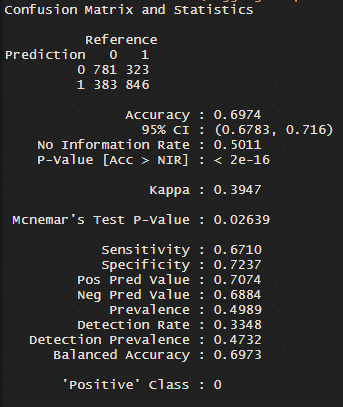


Figure 17: Confusion Matrix of Bagging Classification – Train and Test

Figure16 represent confusion matrix for both datasets. Our train accuracy is 1 which normally cannot acceptable because of overfitting problem. However, in bagging process we can avoid overfitting. Because of that we can say that our model is very effectively work with our train data. However, the accuracy is decrease to 0.69 at train data which is not good. If we can get higher accuracy level, we should feel to positive for bagging model as our best classification method.

Secondly, it is obvious that we have perfectly plotted ROC curve for train with 1 AUC value. AUC value for our test graph is 0.6925964. Again, there is huge decrease in AUC value in test data. Also, in Figure17 you can see the how our roc curve changes significantly. Even I thought that it is best method for my data after the train data, the model does not work effectively as I expected.

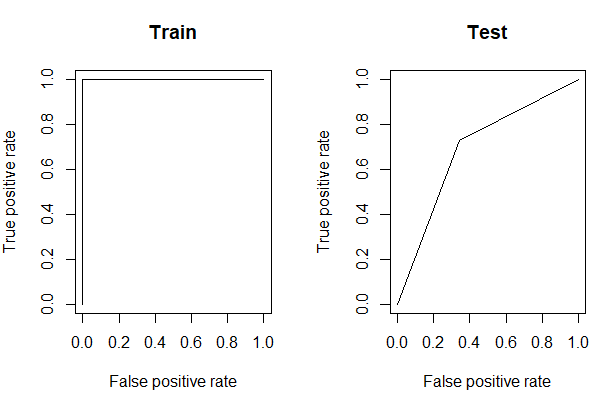


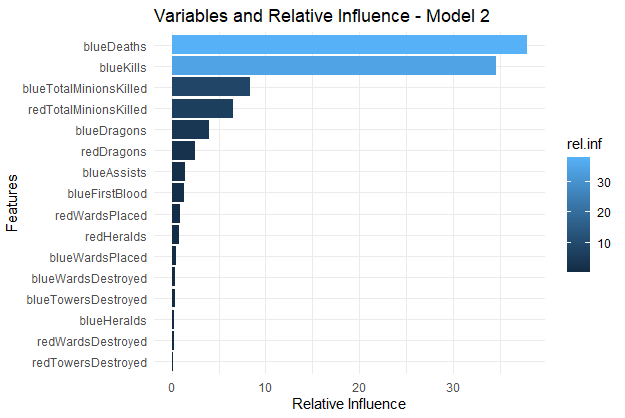
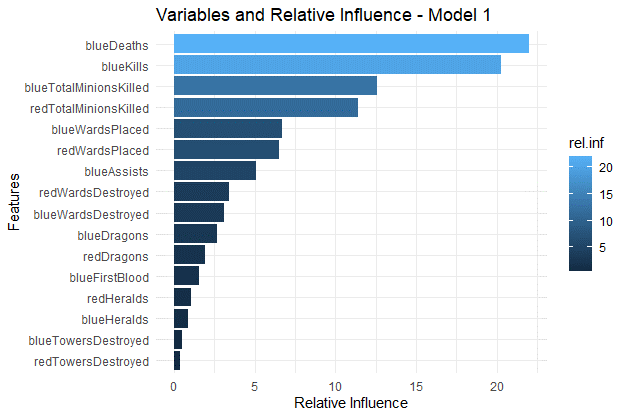
Figure 18:ROC of Train and Test

### Boosting

The term 'Boosting' refers to a family of algorithms which converts weak learner to strong learners. Boosting is an ensemble method for improving the model predictions of any given learning algorithm. The idea of boosting is to train weak learners sequentially, each trying to correct its predecessor.

Boosting process is very effective to understand that relative influence of dependent variables to our response variable. In first model, I used 5000 ntress, 4 interaction depth and 0.01 shrinkage. Most important variables are blueDeaths and blueKills. Secondly, total minions killed by teams are relatively affected in model 1. However, we should check the cross-validation values and should be control that how much ntrees I should use. My cross-validation values give that I should 478 ntrees for my model. Then I change ntrees and create another model for boosting classification.

Figure 19: Features and Relative Influence for Model1 and Model 2



In model 2, there is huge increase in blueDeath and blueKills relative influence. It is also expected because, other model also says that blueDeaths and blueKills variables have the huge impact on winning condition of blue team. Then performance measurement of model 2 is checked.

The accuracy of train dataset is 0.68 in boosting methods. Also, there is increase in train accuracy to 0.70 which is not expected. Because of that we can say that boosting method cannot be correct chose for our model. However, the randomly selected sample issues are same with decision – tree. When I change train and test sample for this method, the accuracy is changed. However, this is not appropriate method to eliminate that because it effects all other models were created before.

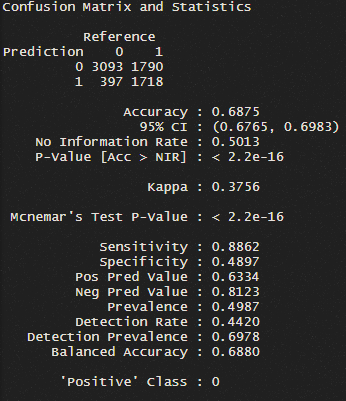
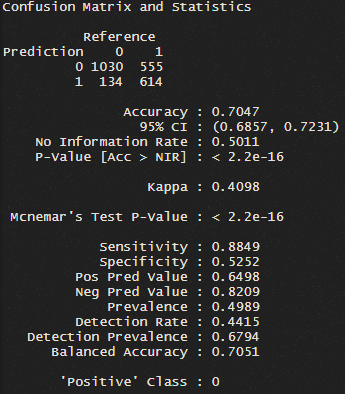
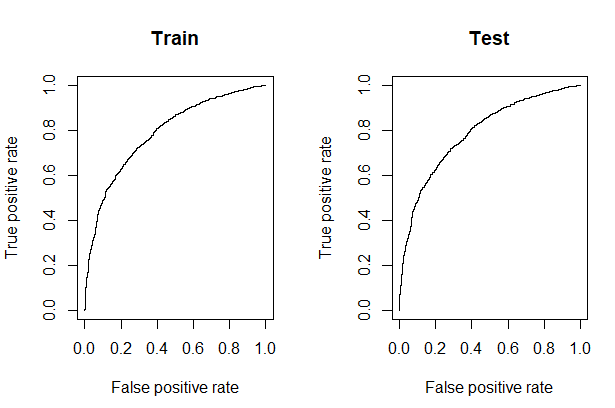


Figure 20:Confusion Matrix of Boosting

|  |
| --- |
| AUC Train0.811  AUC Test: 0.795 |



We have same AUC values and similar ROC curves for train and test data. These are one of closest AUC values to logistic regression model.

### Random Forest

Random Forest is a classification algorithm consisting of many decisions’ trees. Random Forest uses bagging and feature randomness for building each individual tree. It is try to create an uncorrelated forest of trees whose prediction by committee is more accurate than that of any individual tree.

My first random forest model created by using 1000 ntree and 4 mtry value. The mtry value is coming from square root of 16 which is the number of our variables. Then I use tune process the random forest getting lower out of bag errors. After tune process I understand that, my mtry value should be equal to 2. It decreases my OOB error 0.5%. Figure 20 represent that which variables are important based on our random forest model.

The model has a similar result like others. However, in this model, blueDeaths has higher mean decrease accuracy than blueKills. Also, blueAssists variable has this level importance at first time.

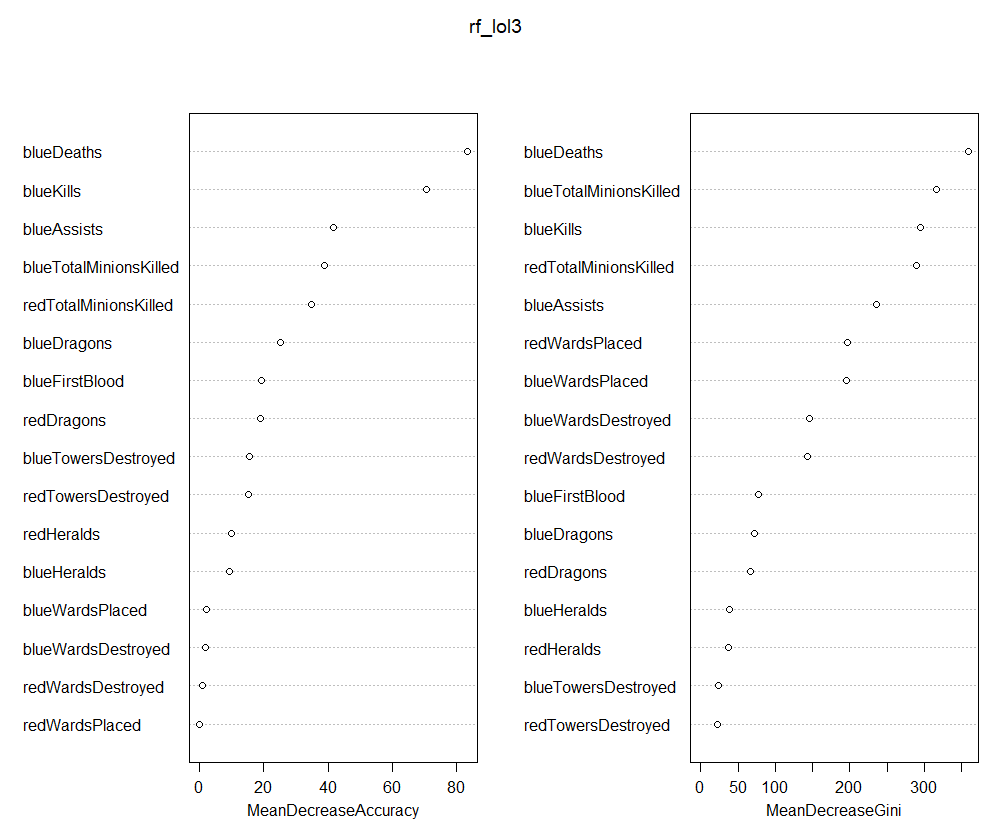


Figure 21:Important Variable in Random Forest

With random forest, I find best accuracy levels in my confusion matrixes. Figure 21 shows the confusion matrix for both train and test data. The train accuracy is 0.95 and it is decreased to 0.70 accuracy at test data. Even it is a huge decrease, it can acceptable if we compare this situation with another machine learning algorithm. We cannot suspect from overfitting because, random forest eliminates overfitting problem.

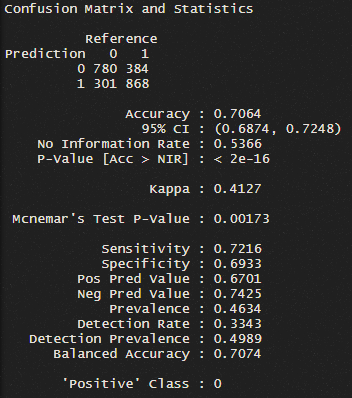
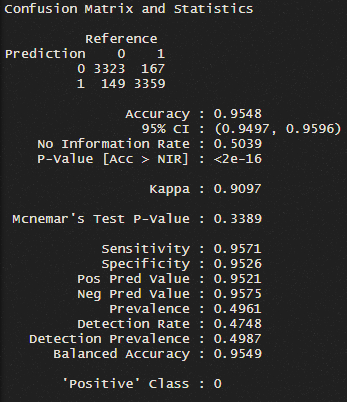
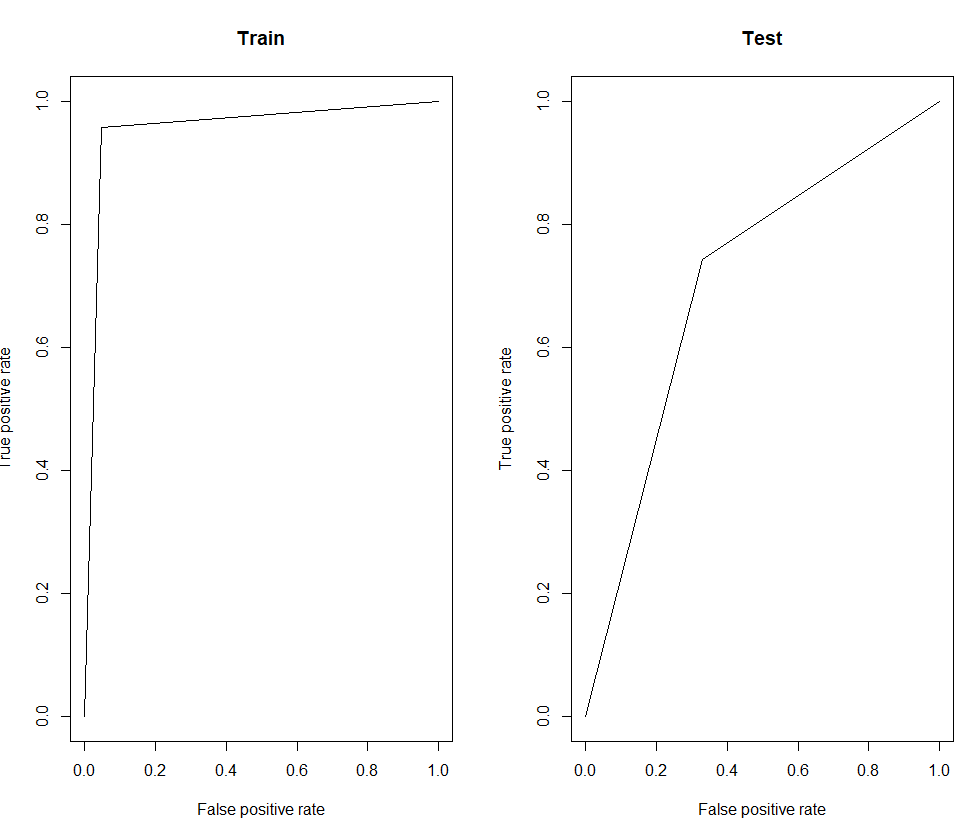


Figure 22:Confusion Matrix of Random Forest - Train and Test

Secondly, after a good accuracy levels, we should check the ROC curve AUC value of this model. The below figure show ROC curve and AUC levels for Random Forest.

|  |
| --- |
| AUC Train: 0.954  AUC Test: 0.706 |



We have higher AUC train value which is 0.954 and our AUC Test value is 0.706.

### Comparison

Below table represent the all accuracy and AUC values of all methods. Before the choose a model, we should understand these values. However, blueDeaths and blueKills variables are main important features about lol teams, in gameplay actions. In all models, these two variables have direct effect on our dependent variable.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm** | **Accuracy Train** | **Accuracy Test** | **AUC Train** | **AUC Test** |
| **Logistic Regression** | 0.72 | 0.71 | 0.80 | 0.79 |
| **SVM** | 0.73 | 0.69 | 0.73 | 0.69 |
| **Decision – Tree** | 0.66 | 0.68 | 0.66 | 0.68 |
| **Bagging** | 1 | 0.69 | 1 | 0.69 |
| **Boosting** | 0.68 | 0.70 | 0.79 | 0.79 |
| **Random Forest** | 0.95 | 0.70 | 0.95 | 0.7 |

For classification part of my project, I need to decide between Bagging and Random Forest algorithm. Both methods have a higher train accuracy level and AUC values, but their test accuracy and AUC values are lower. They have very close values. Also, AUC values of logistic regression are quite better than others. 0,80 for train and 0,79 for test. Because of Random forest has higher performance measurement in train metrics and test metrics are very close to other methods, I prefer to study with Random Forest for my dataset.

## Prediction

In this section, I try to develop a machine learnings models to predict continuous variable which is blueTotalMinionsKilled. I mainly focused on Multiple Linear Regression, Decision-Tree, Bagging, Boosting and Random Forest. Finally, I try to decide which model I should use to predict our dependent variable better.

### Multiple Linear Regression

Linear regression in statistics is a linear approach to model the relationship between scalar response and one or more explanatory variables. The state of an explanatory variable is called simple linear regression. The process for multiple explanatory variables is called multiple linear regression. Multiple Linear Regression one of the most used model in statistics. There are important assumptions for multiple linear regression models. To create a suitable model with multiple linear regression, your model should provide these assumptions.

My first model includes all variables in our dataset. As I explained in classification part, I try to find a best model with using stepAIC method from R. This method gives me to model which is .

The below Figure23 represent the significant level of variables and coefficients.

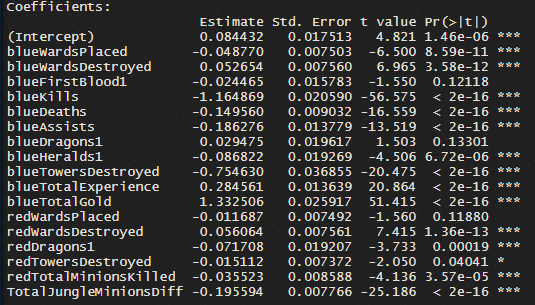


Figure 23:Model 1 – Prediction

When I check the VIF value of this model to understand that is there any multicollinearity problem, I see that blueTotalGold cause to multicollinearity. It can be related with player can earn gold with killing minions and most of the gold earning by players coming from minions because of that I decide to drop blueTotalGold out of my model. After dropping blueTotalGold, there is no multicollinearity problem in my model.



After that, blueHeralds1, blueDragons1 redTowersDestroyed and redWardsPlaced variables become an insignificant variable. Because of that I drop out in my model. After this period, I reach my model for using the linear regression part. I have 0.502 and all variables are significant 0 level expect redDragons1. This variable is significant at 0.001 level. Then I decide to check assumptions for my model.

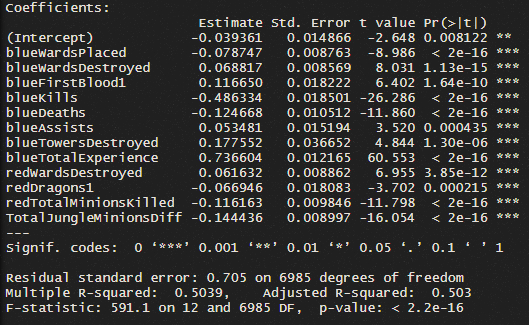


Figure 24:Final Model

Firstly, I control again the VIF values to check multicollinearity. All values are very low and there is no multicollinearity problem for my model.

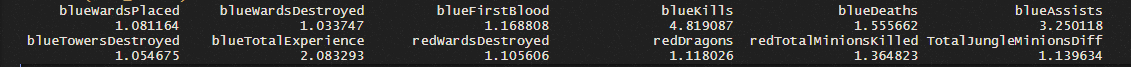


Figure 25:VIF values

Secondly, I should check that my residuals distributed normally or not. I used Anderson Darling test to check normality assumption because of my observation number is not suitable for Shapiro Wilk test. Anderson Darling test has a significant result with 3.126e-06 p value and because of that I should reject the null hypothesis and normality problem occurs in my model. There are different ways to solve normality problem. Like using logarithmic transmission and Boxcox. I used Boxcox transformation, because I already standardized my variables. I faced with problem after using Boxcox. The lambda values of minimum “lik” value is 1.434343. After that I get a p value 0.9346 from Anderson – Darling test and my normality problem was solved.

For autocorrelation problem, I used Durbin Watson test and it gives me 0.02 p value. It means that, null hypothesis is rejected, and I have autocorrelation problem. To solve autocorrelation, I add different variables and create different models. However, other models have normality problem and adding different variables cannot solve my autocorrelation problem. Other techniques for using to solve autocorrelation is higher than my skills. Because of that I moved on without solving this problem. For heteroskedasticity, I used Breusch – Pagan test and it gives me lower p value than 0.05. It means that my model includes heteroskedasticity problem. Even I tried different methods to eliminate heteroskedasticity problem. I mainly try boosting models and robusted linear regression. However, I cannot find any positive results to prevent this problem.

Then, I decided to check my regression model performance by control the Rsquared, RMSE and MAE.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **RMSE** | **Rsquared** | **MAE** |
| **Train** | 561.8239 | 0.5 | 559 |
| **Test** | 565.98 | 0.52 | 563 |

Figure 26:Linear Regression

In the test data even Rsquared increase 0.02-point error rates also increases at the same time. However, for both datasets, the effectiveness of this model is not good. Rsquared values are very low. They should be higher.

### Decision-Tree

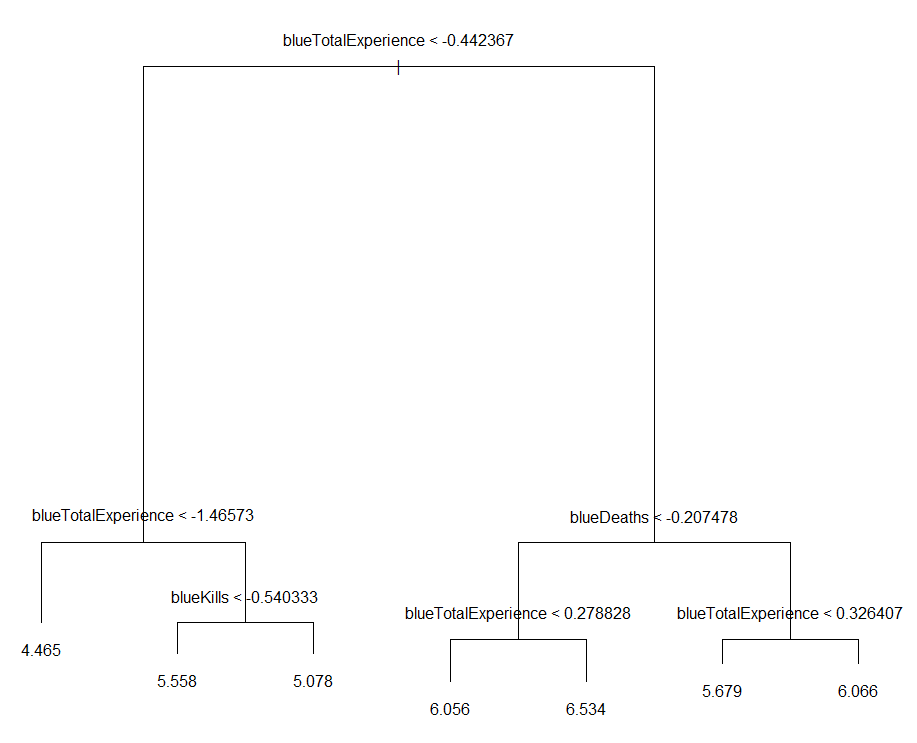
My first decision – tree model includes 7 nodes and 3 variables used for its construction which are blueTotalExperience, blueKills and blueDeaths. However, to be sure that it is the best model for decision tree algorithms, I check the cross-validation value of this tree. Then my lowest deviance value is 7. Then I do not need to prune process. The values on the tree is standard values of our model. In generally, higher experience can increase the total killed number of teams. Also, lower death rates can increase. However, at the left side of tree you can see that lower blueKills increase the number of killed minions. It is not expected but I realized that, most of the player who focus on killing his opponent can loose minions because they mostly choose to fight with their opponents in lane phase. It can cause losing minions. After I understand this logic, it can be very sensible for me. I check my LoL matches and I found the similar thing for myself.

Figure 27:Decision Tree

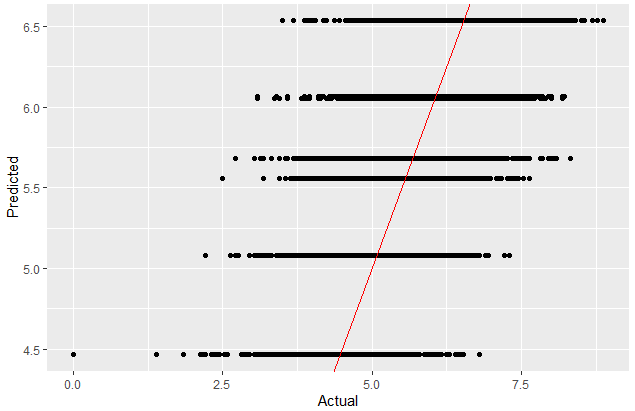
You can see from Figure27, the model is not successful to predict actual values. When we check the Rsquared and error values below table, you can see that the values are very low, and explanation of these model is not good.

Figure 28:Actual vs Predicted Values

|  |  |  |  |
| --- | --- | --- | --- |
|  | **RMSE** | **Rsquared** | **MAE** |
| **Train** | 0.805 | 0.350 | 0.640 |
| **Test** | 5.857 | 0.347 | 5.799 |

Figure 29:Decision Tree

### Random Forest

First Random Forest model created by mtry = 18 and ntreeTry = 1000. I choose 18 because of 18 is the number of independent variables. Mean of squared residuals is 0.42 and %Var explained: 57.88. However, I should use tune process to find a best mtry value for my Random Forest model. The best mtry value is 12 with OOB error = 0.42 which is lower.

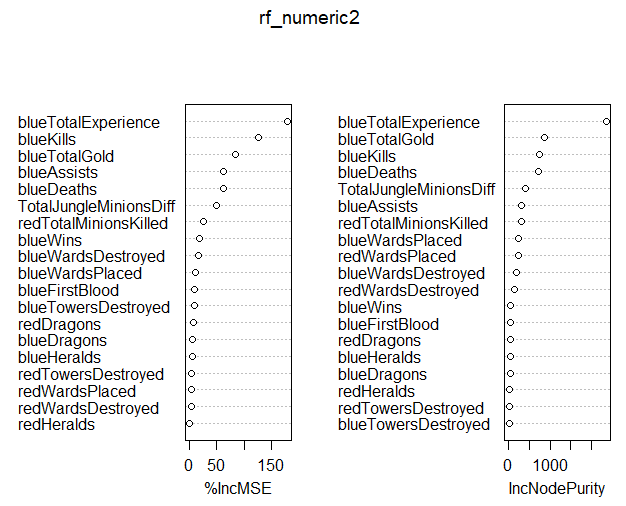


Figure 30:Variable Importance Plot

Figure 30 represent the importance of variables level. blueTotalExperience has a huge importance level in random forest model and as a similarly blueKills placed at second place with other methods.

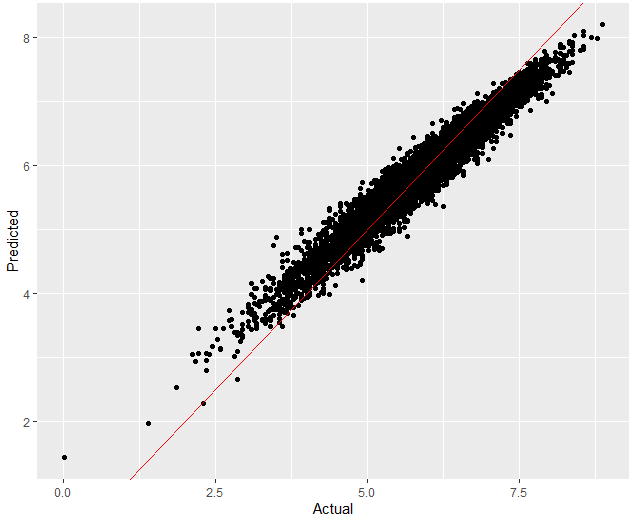
These graph and table show that our model can successfully predict the values of train data. However, to understand the real successful of this method we should also control how does model predicts our train dataset values. As a performance measure, I check the Rsquared and error values of my random forest model.

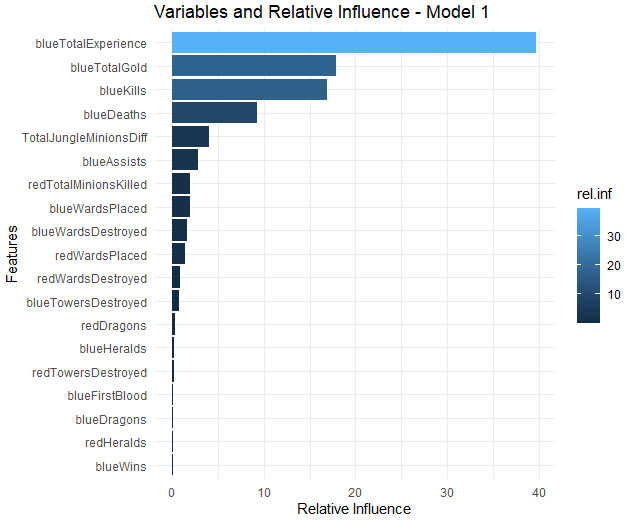
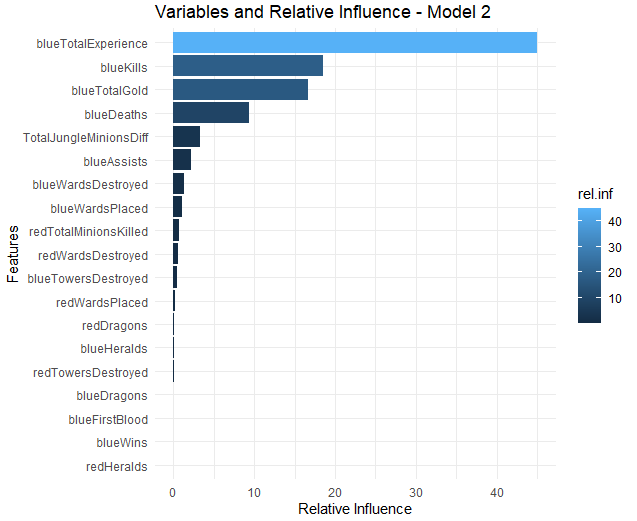
Figure 31: Predicted vs Actual Values

|  |  |  |  |
| --- | --- | --- | --- |
|  | **RMSE** | **Rsquared** | **MAE** |
| **Train** | 0.2710 | 0.94 | 0.211 |
| **Test** | 5.83 | 0.58 | 5.799 |

The table shows that our train Rsquared is very high, but the decrease is too much in train data. In addition, the errors of train data increase very significantly. If it is not Random Forest, we can suspect the overfitting problems, but because of the structure of random forest, it is preventing the overfitting problems.

### Boosting

My boosting model is created with based on 5000 ntree, 4 interaction depth and 0.01 shrinkage value. This model performed 5000 iterations and there were 19 predictors which 19 had non-zero influence. However, I should check cross validation performance of my model and I understand that, I should use 1741 ntree. Then I created second model.



These graphs represent the variables and their relative influences for model 1 and model 2. There are no significant changes after the cross-validation. You can see the predicted and actual value graphs for boosting model. This model does not see successfully like random forest.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **RMSE** | **Rsquared** | **MAE** |
| **Train** | 0.589 | 0.655 | 0.469 |
| **Test** | 5.829 | 0.629 | 5.796 |

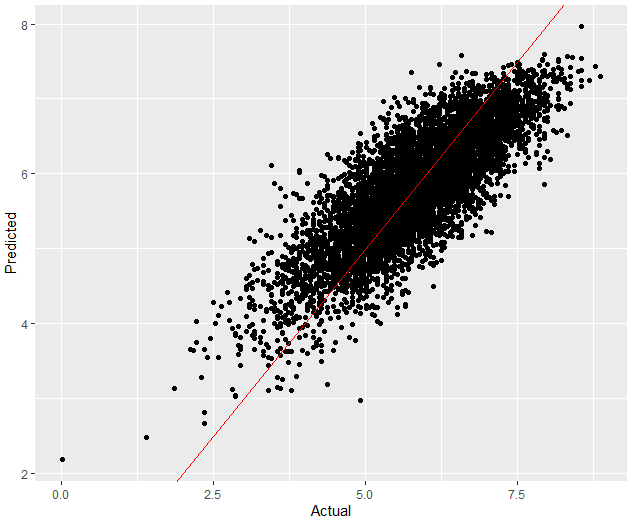
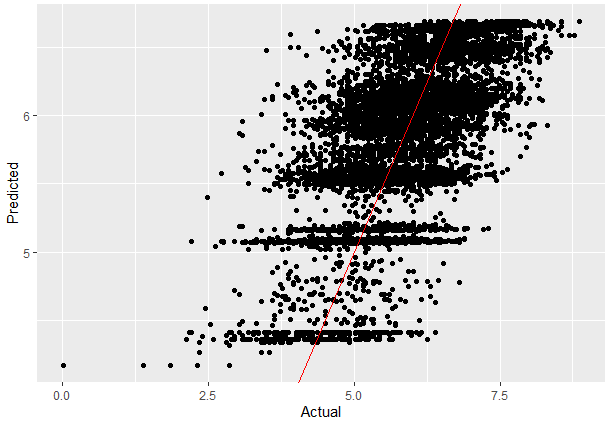
The train R square is 0.655 and the error rate are small. However, 0.655 is not enough for successful models. But it is best Rsquared train value. Also, in train dataset, error rate increases sharply and our Rsquared decrease little bit.

Figure 32: Actual vs Predicted Values

### Bagging

The bagging model gives us a 0.78 with root mean squared error which is very high to consider as a successful method. Also, we can say that it is a very unsuccessful model. The below table and graph are also showing that bagging model is failed to predict our dependent variable.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **RMSE** | **Rsquared** | **MAE** |
| **Train** | 0.775 | 0.402 | 0.617 |
| **Test** | 5.584 | 0.398 | 5.795 |

The train and test Rsquared are very close to each other and they are under the 0.5. The explanatorily of this model is so low.

### Comparison

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **RMSE Train** | **RMSE Test** | **Rsquared Train** | **Rsquared Test** | **MAE Train** | **MAE Test** |
| **Linear Regression** | 561.8239 | 565.98 | 0.5 | 0.52 | 559 | 563 |
| **Decision Tree** | 0.805 | 5.857 | 0.350 | 0.347 | 0.640 | 5.799 |
| **Random Forest** | 0.2710 | 5.83 | 0.94 | 0.58 | 0.211 | 5.799 |
| **Boosting** | 0.589 | 5.83 | 0.655 | 0.629 | 0.469 | 5.796 |
| **Bagging** | 0.775 | 5.584 | 0.402 | 0.398 | 0.617 | 5.795 |

Figure 33:Table of Models Comparison

Figure 33 represent the performance measurement of all models. As you see that Random Forest has the best option for our prediction. It has very good Rsquared value for train dataset. Even it has lower Rsquared test value, the difference between Boosting and Random Forest 4%. When we considering the other factors, we can say that Random Forest is more successful algorithm than as well as others.

1. It means that best champions of game for this patch. Meta changes always after a patch. Most of the players try to adapt meta champions and play with them in this period. [↑](#footnote-ref-1)