League of Legends Win Prediction using R

Darwin Khay

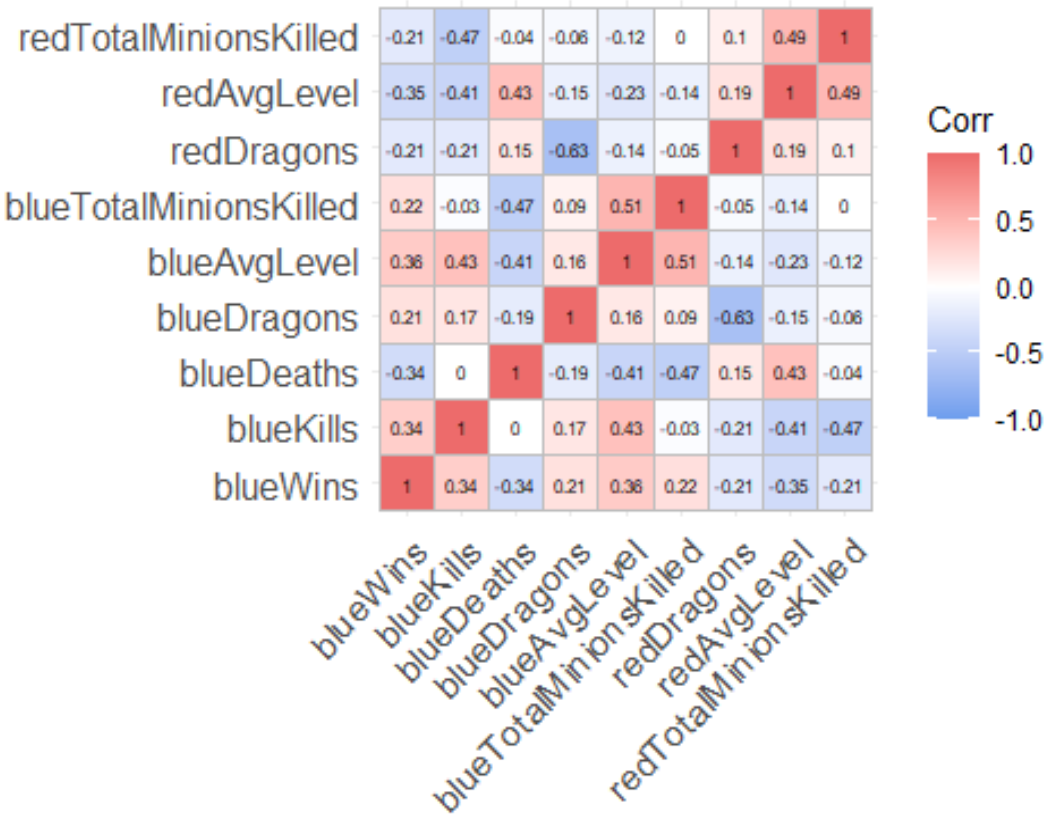
```
# Loading the dataset and appropriate packages

ranked <- read.csv("C:\\Users\\khayd\\Documents\\FALL 2020 Files\\STAT 1601\\
Datasets\\high_diamond_ranked_10min.csv")

library(dplyr)

library(caret)

library(ROCR)

library(ggcorrplot)
```

Previewing the data

```
glimpse(ranked)

## Rows: 9,879
## Columns: 40
## $ gameId                      <dbl> 4519157822, 4523371949, 4521474530, 4
5...
## $ blueWins                    <int> 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1
,...
## $ blueWardsPlaced             <int> 28, 12, 15, 43, 75, 18, 18, 16, 16, 1
3...
## $ blueWardsDestroyed          <int> 2, 1, 0, 1, 4, 0, 3, 2, 3, 1, 3, 2, 1
,...
## $ blueFirstBlood              <int> 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1
,...
## $ blueKills                   <int> 9, 5, 7, 4, 6, 5, 7, 5, 7, 4, 4, 11,
7...
## $ blueDeaths                  <int> 6, 5, 11, 5, 6, 3, 6, 13, 7, 5, 4, 11
,...
## $ blueAssists                 <int> 11, 5, 4, 5, 6, 6, 7, 3, 8, 5, 6, 7,
1...
## $ blueEliteMonsters           <int> 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1
,...
## $ blueDragons                 <int> 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1
,...
## $ blueHeralds                 <int> 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0
,...
## $ blueTowersDestroyed         <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
```

```
,...
## $ blueTotalGold                <int> 17210, 14712, 16113, 15157, 16400, 15
8...
## $ blueAvgLevel                 <dbl> 6.6, 6.6, 6.4, 7.0, 7.0, 7.0, 6.8, 6.
4...
## $ blueTotalExperience          <int> 17039, 16265, 16221, 17954, 18543, 18
1...
## $ blueTotalMinionsKilled       <int> 195, 174, 186, 201, 210, 225, 225, 20
9...
## $ blueTotalJungleMinionsKilled <int> 36, 43, 46, 55, 57, 42, 53, 48, 61, 3
9...
## $ blueGoldDiff                 <int> 643, -2908, -1172, -1321, -1004, 698,
...
## $ blueExperienceDiff           <int> -8, -1173, -1033, -7, 230, 101, 1563,
...
## $ blueCSPerMin                 <dbl> 19.5, 17.4, 18.6, 20.1, 21.0, 22.5, 2
2...
## $ blueGoldPerMin               <dbl> 1721.0, 1471.2, 1611.3, 1515.7, 1640.
0...
## $ redWardsPlaced               <int> 15, 12, 15, 15, 17, 36, 57, 15, 15, 1
6...
## $ redWardsDestroyed            <int> 6, 1, 3, 2, 2, 5, 1, 0, 2, 2, 2, 1, 1
,...
## $ redFirstBlood                <int> 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0
,...
## $ redKills                     <int> 6, 5, 11, 5, 6, 3, 6, 13, 7, 5, 4, 11
,...
## $ redDeaths                    <int> 9, 5, 7, 4, 6, 5, 7, 5, 7, 4, 4, 11,
7...
## $ redAssists                   <int> 8, 2, 14, 10, 7, 2, 9, 11, 5, 4, 5, 9
,...
## $ redEliteMonsters             <int> 0, 2, 0, 0, 1, 0, 0, 1, 2, 0, 1, 0, 0
,...
## $ redDragons                   <int> 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0
,...
## $ redHeralds                   <int> 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0
,...
## $ redTowersDestroyed           <int> 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
,...
## $ redTotalGold                 <int> 16567, 17620, 17285, 16478, 17404, 15
2...
## $ redAvgLevel                  <dbl> 6.8, 6.8, 6.8, 7.0, 7.0, 7.0, 6.4, 6.
6...
## $ redTotalExperience           <int> 17047, 17438, 17254, 17961, 18313, 18
0...
## $ redTotalMinionsKilled        <int> 197, 240, 203, 235, 225, 221, 164, 15
7...
## $ redTotalJungleMinionsKilled  <int> 55, 52, 28, 47, 67, 59, 35, 54, 53, 4
3...
## $ redGoldDiff                  <int> -643, 2908, 1172, 1321, 1004, -698, -
```

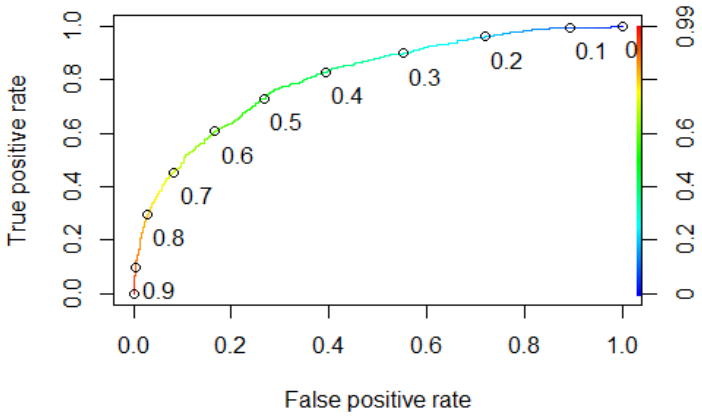
```
2...
## $ redExperienceDiff            <int> 8, 1173, 1033, 7, -230, -101, -1563,
8...
## $ redCSPerMin                  <dbl> 19.7, 24.0, 20.3, 23.5, 22.5, 22.1, 1
6...
## $ redGoldPerMin                <dbl> 1656.7, 1762.0, 1728.5, 1647.8, 1740.
4...

# Selecting the variables that make the most sense and are relevant
ranked_games <- ranked%>%
  select(blueWins, blueKills, blueDeaths, blueDragons, blueAvgLevel, blueTota
lMinionsKilled,redDragons,redAvgLevel,redTotalMinionsKilled)

# Finding correlation between blue team wins and the other variables
ggcorrplot(cor(ranked_games), colors = c("#6b9ded", "white", "#ed6b6b"), lab
= T, lab_size= 2)
```



It seems to be the case that blue team's kills, dragons, average level and total cs positively affect their chances of winning the most. While on the other hand, blue team's deaths and enemy team's dragons, average level and cs negatively affect blue team's chances of winning.

Logistic Regression

```
# Using logistic regression to predict the win or loss of the blue team
trainIndex <- createDataPartition(ranked_games$blueWins, p = 0.75, list = F,
times = 1)
train <- ranked_games[trainIndex,]
test <- ranked_games[-trainIndex,]
win.model <- glm(blueWins ~ blueKills+blueDeaths+blueDragons+blueAvgLevel+blu
eTotalMinionsKilled+redDragons+redAvgLevel+redTotalMinionsKilled, data = trai
n, family = "binomial")
test$model_prob <- predict(win.model, test, type = "response")

# Prediction function
ROCRprediction <- prediction(test$model_prob, test$blueWins)
# Performance function
ROCRperformance <- performance(ROCRprediction, "tpr", "fpr")
# Plot ROC curve
plot(ROCRperformance, colorize = T, print.cutoffs.at=seq(0,1,by=0.1), text.ad
j=c(-0.2,1.7))
```




```
# Checking for accuracy of the model
test <- test%>%
  mutate(model_pred = 1*(model_prob > 0.60))
test <- test%>%
  mutate(accurate = 1*(model_pred == blueWins))
sum(test$accurate)/nrow(test)

## [1] 0.7181045
```

75% of the dataset was used to **train** the logistic regression model while rest of the **25%** of the dataset was used to **test** the model. The **threshold for an appropriate success** is deemed to be around **60%** since it would allow for a decent **true positive rate** (just under **60%**) whle minimizing the **false positive rate** (just under **20%**). Therefore, the **predictive power** of the model is around **70%**.

```
summary(win.model)

##
## Call:
## glm(formula = blueWins ~ blueKills + blueDeaths + blueDragons +
##     blueAvgLevel + blueTotalMinionsKilled + redDragons + redAvgLevel +
##     redTotalMinionsKilled, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5217  -0.9026  -0.2134   0.9003   2.7385
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)            -0.356420   1.022903  -0.348    0.728
## blueKills               0.185007   0.013341  13.867  < 2e-16 ***
## blueDeaths             -0.197184   0.013611 -14.487  < 2e-16 ***
## blueDragons             0.300151   0.073052   4.109 3.98e-05 ***
## blueAvgLevel            0.966310   0.131648   7.340 2.13e-13 ***
## blueTotalMinionsKilled  0.006999   0.001609   4.350 1.36e-05 ***
## redDragons             -0.306185   0.071612  -4.276 1.91e-05 ***
## redAvgLevel            -0.900933   0.130304  -6.914 4.71e-12 ***
## redTotalMinionsKilled  -0.007053   0.001586  -4.446 8.74e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 10272.1  on 7409  degrees of freedom
## Residual deviance:  8038.1  on 7401  degrees of freedom
## AIC: 8056.1
##
## Number of Fisher Scoring iterations: 4
```

The logarithmic coefficients are shown and all variables used are significant since there are 3 stars (***) associated with each of them and the p-values are low.