Dota 2 Match Data Analysis

Muhan Yuan
yuanmh@umich.edu

Motivation

Dota 2 is a popular Multiplayer Online Battle Arena (MOBA) on Steam. There are more than 550,000 average concurrent users[1] and tens of thousands of games played every hour, which provides a rich set of data for analysis. On the other hand, by analyzing the data of the matches, we can gain a better understanding of the game and come up with inspirations, that can potentially improve players' performance and help the game makers to balance the game.

In the game, two teams (Radiant and Dire) compete to destroy the base of the opposing team while defend their own base. Each player collect experience and items by killing creeps or the enemies. There are 114 heroes for players to choose from, but in each game, one player can only pick one hero.

At the end of last year, Dota 2 went through a major version update. This update introduced "Talent Tree" system to the game, which fundamentally changed the role some heroes play in the game. Also, the map and some items are also remade in the new version, which made many players believe Dota 2 had become an entirely new game. As a long-time player of Dota 2, I am most curious about how this version update actually changed the game.

In this project, I will mainly answer these following questions:

1. **How did this version update change the game features (duration, kill count and heroes win rate)?**
2. **Can heroes be divided into clusters based on their average performance in the game?**
3. **How does the game features (kill count, damage, networth) change**

[1] http://steamcharts.com/app/570

with different game modes?

4. How can we predict the game result based on players' performance?

Data Source

In this project, I used two separated datasets. The first one is found from Kaggle: Dota 2 Matches dataset[2]. This dataset contains 19 csv files and I used 3 of them (match.csv, players.csv and hero_name.csv). Match.csv contains basic information of 50,000 matches, including the match ID (Integer) and the match duration (Integer). Players.csv contains the performance of each player in the match, including the account_id (Integer), match_id (Integer), player_slot (Integer), hero_id (Integer between 1 and 114), kills, deaths, assists, etc (all Integers). Since we cannot tell which hero the player is using simply from the hero_id in the players.csv file, we need to look up their real name from hero_name.csv.

However, this dataset is collected before the version update, so to get the data of new game version, I used Dota 2 API[3] to collected another dataset. This dataset contains another 25,544 matches' data in JSON format, which covers almost the same information as the Kaggle dataset. I found a match on January 3rd from my own match history and used its ID as seed, continuously requested 60,000 matches data. However, since some matches are between human players and bot, which doesn't have any record, only 25,544 matches get successfully cached.

[2] Kaggle: Dota 2 Matches, https://www.kaggle.com/devinanzelmo/dota-2-matches
[3] Dota 2 API: https://dota2api.readthedocs.io/en/latest/

Methods

Data Manipulation

To facilitate following analysis, I used python to preprocess the data on **match level** and **player level**.

On match level, I loaded the data into new csv file contains the following variables:

MatchID: Interge. Directly get from the raw dataset

Radiwin: Factor. The result of the match, 1 if Radiant wins, 0 otherwise

Time: Integer. The duration of the match (in seconds), directly get from the raw dataset

Fbtime: Integer. How many seconds after the game start does the first player get killed

Radi_totalkill: Integer. How many enemies do radiant players kill. To calculate the kill count on each side, we need to use player_slot variable to discriminate players on each side. If player_slot is between 0 and 4, this player is on the radiant side, otherwise, his is on the dire side. The value is the sum of kill count of all radiant players

Dire_totalkill: See above.

Radi_totalassist: Integer. How many assists do radiant players have.

Dire_totalassist: See above.

Radi_denies: Integer. How many denies do radiant players have.

Dire_denies: See above.

Radi_networth: Integer. The total value of all items of radiant players.

Dire_networth: See above.

Radi_damage: Integer. The total damage made by radiant players

Dire_damage: See above.

Radi_level: Integer. The sum of radiant players' level

Dire_level: See above.

Radi_heal: Integer .The total healing made by radiant players

Dire_heal: See above.

Mode: Factor. The number indicates the game mode, which is directly found

from the raw dataset

Version: Charater. 7.00 (new version) or 6.88 (old version)

On player level, I also use python to load data into csv file contains following variables:

Hero: Integer. The name of hero used by the player

Win: Factor. 1 if the player is on the winning side, 0 otherwise. On player level, we don't need to worry about which side the player is on. To see if the player is on the winning side, we need to look at both the game result and the players' slots. For instance, if the radiant side wins the game, and the player's slot number is between 0 and 4, 1 will be assigned to this variable.

Following variables (Integer) are all directly get from the raw dataset:

Kills, death, assists, damage, heal, lasthit, denies, level and **tower_damage**.

For very few matches, there are missing values for hero_id, which indicates there is a player quit the match before starting the game. These records are not appropriate for analysis so I chose to drop them.

The player_rating.csv file is directly used for analyzing the relation between win rates and true skills. To void the influence from extreme values, I dropped those players who played in less than 5 matches.

In most cases, the duration of the game is less than 1 hour (3600 seconds). The median of duration is 2453 and Q3 is 2909. But the max value of it is 16037, which means the match lasted more than four and a half hours. These extreme situations do not make much sense to any players and will impede our analysis. So, I also dropped matches, whose duration is longer than 5000 seconds.

Analysis Method

In this project, I used R Studio to perform the data analysis.

To analyze the difference between two versions, I mainly used match level data. As stated above, I dropped rows with duration longer than 5000. I used ggplot to

draw a density plot for match duration distribution of each version.

```
ggplot(matchtable1, aes(x=time, fill=version)) + geom_density(alpha=.3)
+labs(title = "Match Duration Distribution")
```

Also, I used ggplot to draw an interleaved histogram for total kill count distribution of each version.

```
ggplot(matchtable, aes(x=radi_totalkill + dire_totalkill,y = ..density.., fill=version))+ geom_histogram(binwidth=.5, alpha=.5, position="identity")
+labs(list(title = "Kills Distribution",x = "Kills"))
```

To analyze hero clusters, I used player level data by loading the data into a data table and calculating the average performance of each hero in each version.

```
hero_average <- herodata[ , .(win = mean(win), kills= mean(kills), deaths = mean(deaths), assists = mean(assists), damage = mean(damage), heal = mean(heal), lasthit = mean(lasthit), denies = mean(denies), level = mean(level), tower = mean(tower) ), by = c("Hero","version")]
hero_average6 <- hero_average[version == 6.88,,]
hero_average7 <- hero_average[version == 7.00,,]
```

Then, for data in each version, I calculated the Euclidean Distance between heroes and tried to divide heroes into different clusters.

```
hero6.dist = dist(heroave_ave6,method = "euclidean", diag = TRUE,upper = TRUE)
clusplot(pam(hero6.dist,k=2), main = "K-medoid Clustering of Heroes (6.88)",
labels = 2,lines = 0)
```

To analyze the difference between different mode, I used boxplot to draw the distribution of kills and heal in each mode. Also, I used Tukey's Honest Significant Difference confidence interval to show the pairwise comparison result between different groups.

```
ggplot(matchtable7, aes(x=mode,y = radi_totalkill+ dire_totalkill,color = mode))+
geom_jitter(alpha=0.05)+geom_boxplot()+labs(y = "total kills")
TukeyHSD( aov(I(radi_totalkill+ dire_totalkill) ~ mode, matchtable7) )
```

To predict the match result based on players' performance, I used logistic regression model to predict the winning probability. If the fitted probability of is larger than 0.5, we predict the result to be 1. Otherwise, the result is assumed to be 0. Then I randomly divided the dataset into two part, one part with 75% of the rows as training data while another part with 25% of the rows as testing data. I repeated this procedure for 100 times and draw the distribution of the average accuracy.

```
accuracy <- NULL
set.seed(722)
for(i in 1:100)
{
    samp_size <- floor(0.75 * nrow(matchtable6))
    index <- sample(seq_len(nrow(matchtable6)),size=samp_size)
    train <- matchtable6[index, ]
    test <- matchtable6[-index, ]
    model <- glm(radiwin ~ killsgap + assistsgap + deniesgap + networthgap+
damagegap+ levelgap+ healgap,family=binomial,data=train)
    results_p <- predict(model,subset(test,select=c(21:27)),type='response')
    results <- ifelse(results_p > 0.5,1,0)
    answers <- test$radiwin
    err <- mean(answers != results)
    accuracy[i] <- 1-err
}
ggplot(acc.frame, aes(x=acc)) + geom_histogram(aes(y=..density..),binwidth =
0.002, colour="black", fill="white") + geom_density(alpha=.2, fill="green")+
labs(title = "Accuracy Distribution")
```

Challenges

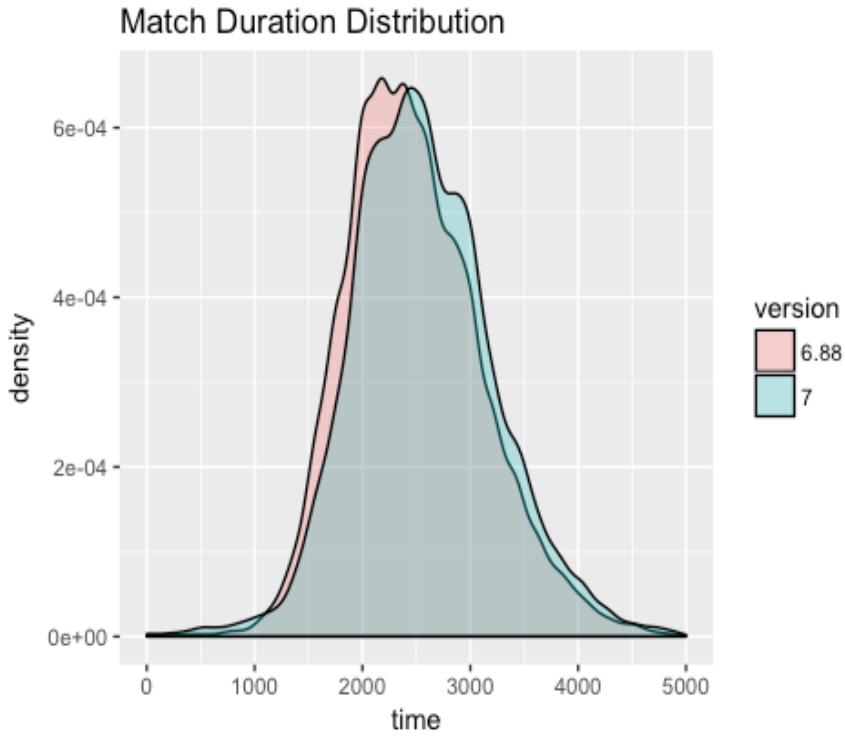
There are two major challenges in data manipulation and analysis.

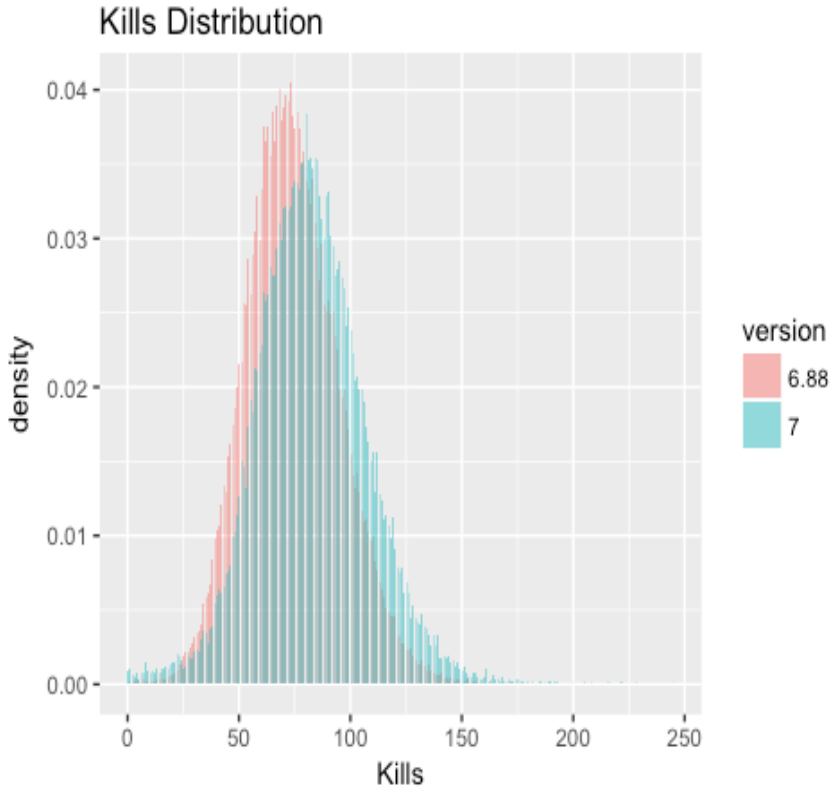
First, the data formats of two datasets are different. For data of new game version, the data is in JSON format, which is hard to used R to do the analysis. And for data of older version, they are stored in different files and I need to extract them carefully and match them together. And some variables are recorded in different types or format in two datasets. For instance, the game result is recorded as 0 or 1 in 7.00 but in True or False in 6.88. So, I have to judge with extra conditions to get the usable data.

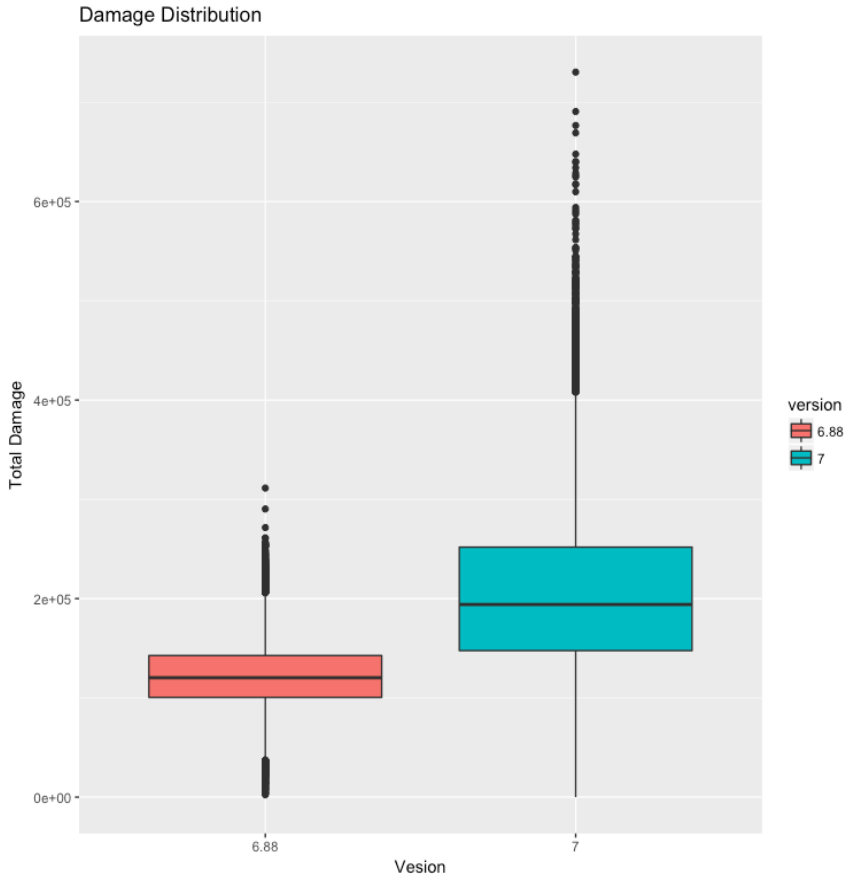
And another challenge is to evaluate the prediction accuracy. I have some experience using python to do the cross validation, but very limited experience with R. To overcome this challenge, I checked R tutorial and learned to sample the data and extract the test result. This experience greatly deepened my understanding about data structure and manipulation in R.

Analysis and Results

Question 1: How did this version update change the game features (duration, kill counts, average net worth)?





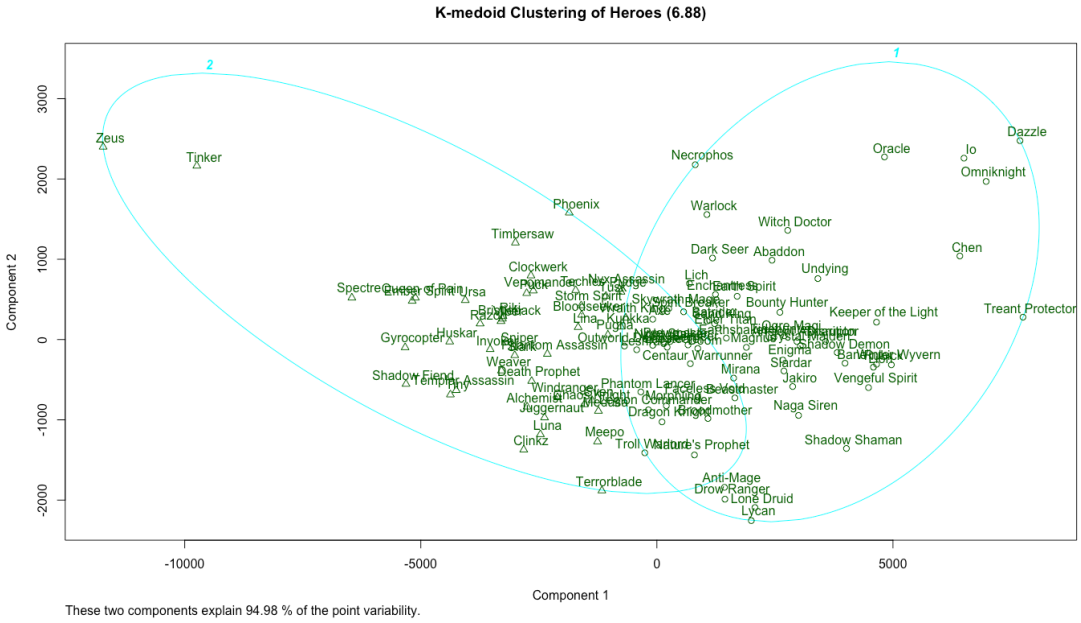


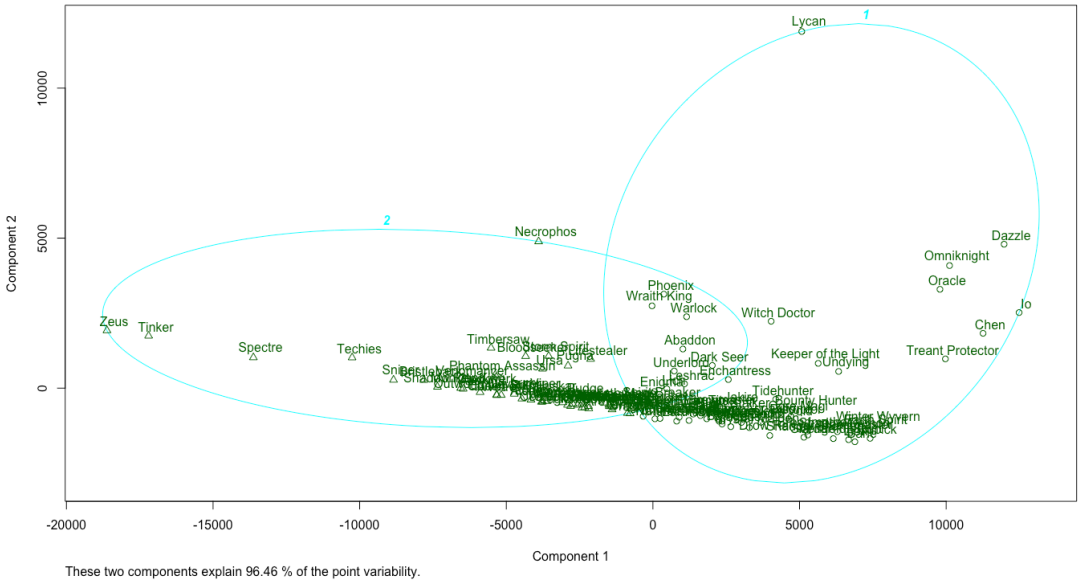
From the above plots, we find the distribution of game duration is slightly different between two versions. It means average game length became longer and the late game power of heroes should be valued more by players.

Also, comparing to earlier version, total kills count is larger in new version. The gap between two versions is similar with the gap of game duration, which means we cannot easily arrive at the conclusion that the fight in this game has become more intense, because more kills may simply come from longer game duration.

However, when we finally look at the total damage distribution of two versions, we notice a remarkable different. The Q1 of damage in new version is almost same as Q3 in earlier version. Since the abilities damage of heroes did not changed very much, it is safe to say damage difference only cause by more fight in the game in the new version.

Question 2: Can heroes be divided into clusters based on their average performance in the game?





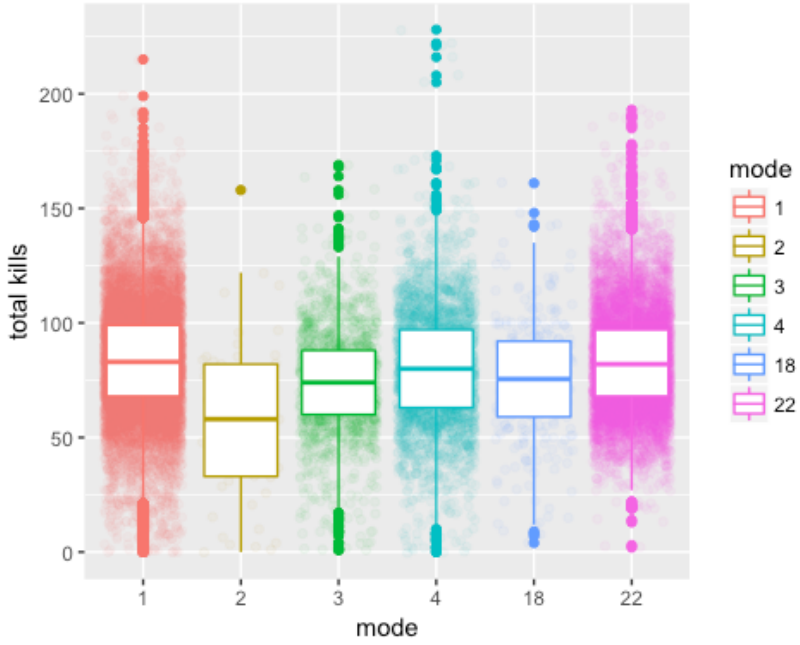
To answer this question, I plotted heroes on two dimensions according to 7 aspects of their average performance. A clear cluster is expected to see from this plot, since player tend to divide hero list into carries and supports, or tanks and mages. However, these plots fail to present two or three distinguishable clusters and I think that is because the role definition of some heroes is relatively flexible.

If we put them into two clusters anyway, we can clearly find that heroes in the left clusters are more considered as carry heroes (kill more enemies, do more damage and have higher net worth) while heroes in the right clusters are more considered as support heroes (do more healing, assist more and have lower net worth). Role of heroes in the middle of two clusters are usually hard to define.

One another interesting finding is, after the version update, heroes on the left side and right side don't change very much. But heroes in the center region are much closer to each other, which means the roles many heroes play in the game become more similar. In my opinion, this change is result from the introducing of "Talent Tree System", which provides more options for players to position themselves in the game.

Question 3: How does the game features (kill count, damage, networth) change with different game modes?

Total kills:



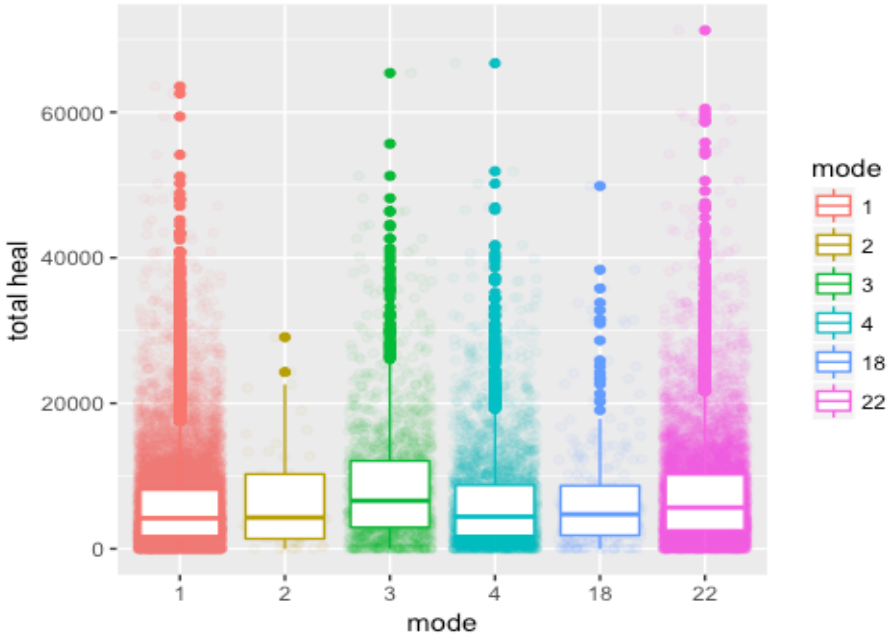
Pairwise Comparison:

```
TukeyHSD( aov(I(radi_totalkill+ dire_totalkill) ~ mode, matchtable7)
 )
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
## Fit: aov(formula = I(radi_totalkill + dire_totalkill) ~ mode, data
 = matchtable7)
## $mode
##             diff        lwr        upr     p adj
## 2-1   -25.1064194 -34.7869101 -15.425929 0.0000000
## 3-1    -9.1349076 -11.0087345  -7.261081 0.0000000
## 4-1    -3.3301781  -4.7145280  -1.945828 0.0000000
## 18-1   -8.9918895 -13.2777429  -4.706036 0.0000000
## 22-1    0.1384369  -0.8543425   1.131216 0.9987285
## 3-2    15.9715118   6.1516507  25.791373 0.0000524
## 4-2    21.7762413  12.0379217  31.514561 0.0000000
## 18-2   16.1145299   5.5652653  26.663795 0.0001947
## 22-2   25.2448562  15.5544458  34.935267 0.0000000
## 4-3     5.8047295   3.6520115   7.957447 0.0000000
## 18-3    0.1430181  -4.4489651   4.735001 0.9999992
## 22-3    9.2733444   7.3489274  11.197761 0.0000000
## 18-4   -5.6617114 -10.0766301  -1.246793 0.0035114
## 22-4    3.4686149   2.0165206   4.920709 0.0000000
## 22-18   9.1303263   4.8221140  13.438539 0.0000000
```

From the boxplot and the pairwise-comparison, we noticed that difference exist between different game modes in term of kills. The kills are significantly less in mode 2 (Captain's mode) comparing to other modes. Captain's mode is the most formal game mode, which is adopted by most of major tournament. So, in this type of matches, players are more cautious and usually implement game strategies strictly, which leaves little opportunities for players to kill enemies or get killed.

Total Healing:




```
TukeyHSD( aov(I(radi_heal+ dire_heal) ~ mode, matchtable7) )

##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = I(radi_heal + dire_heal) ~ mode, data = matchta
ble7)
##
## $mode
##             diff         lwr        upr     p adj
## 2-1     850.3275 -1744.842660  3445.4977 0.9378692
## 3-1    2589.2386  2086.898393  3091.5788 0.0000000
## 4-1     361.2228    -9.897163   732.3428 0.0617228
## 18-1   1034.1819  -114.780371  2183.1443 0.1059778
## 22-1   1429.4454  1163.298649  1695.5922 0.0000000
## 3-2    1738.9111  -893.621852  4371.4440 0.4130193
## 4-2    -489.1047 -3099.777802  2121.5684 0.9948077
## 18-2    183.8544 -2644.218842  3011.9277 0.9999700
## 22-2    579.1179 -2018.711592  3176.9474 0.9883861
## 4-3   -2228.0158 -2805.121809 -1650.9098 0.0000000
```

```
## 18-3  -1555.0567 -2786.087013  -324.0263 0.0043064
## 22-3  -1159.7932 -1675.695714  -643.8907 0.0000000
## 18-4    672.9591  -510.603320  1856.5216 0.5849958
## 22-4   1068.2226   678.941516  1457.5037 0.0000000
## 22-18   395.2635  -759.692855  1550.2198 0.9258880
```

However, when it comes to healing, things are different. First, healing in mode 2 is no longer less than in other modes, which indicates players put a lot more emphasis on assistant work, like healing, in formal matches, given there are much fewer fight in these matches.

Also, we noticed the only in mode 3 and mode 22, the healing is significantly higher than in other modes. In my opinion, the reason is only in this two mode, the game result will directly influence players' ranks, so they play more seriously in these two modes and pay more attention to healing.

Question 4: How can we predict the game result based on players' performance?

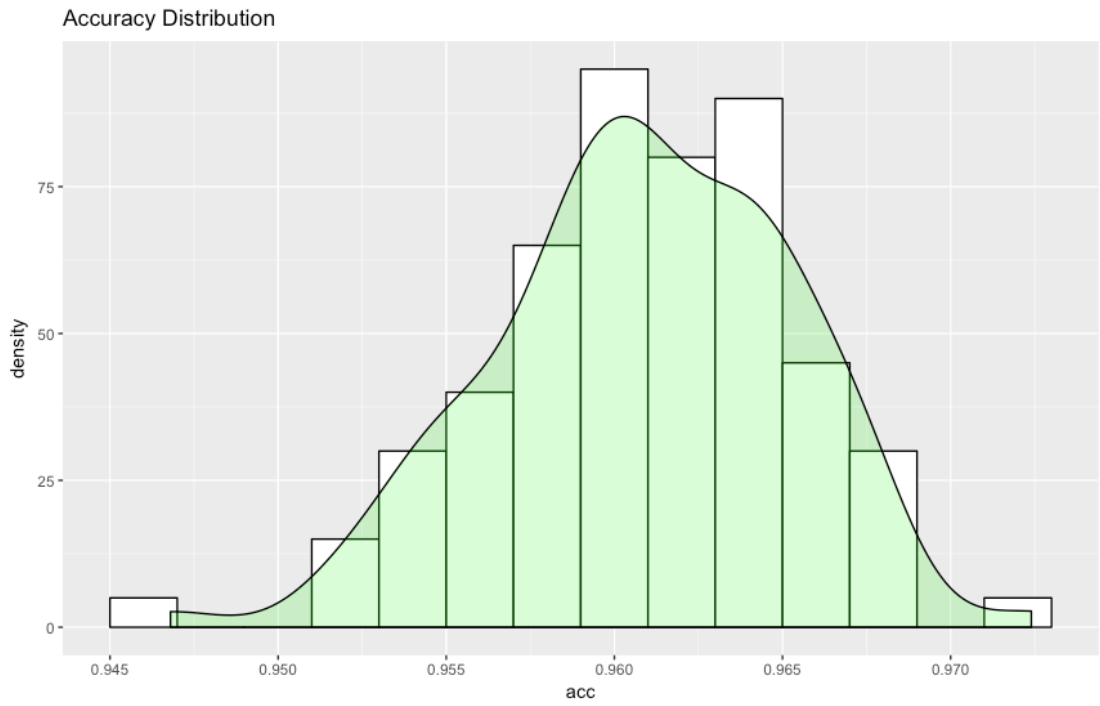
I took the game result as a binomial variable and fit a logistic model to see the influence on the result caused by players' performance difference.

```
## glm(formula = radiwin ~ killsgap + assistsgap + deniesgap + networthgap +
##     damagegap + levelgap + healgap, family = binomial, data = matchtable6)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.9922  -0.0682   0.0051   0.0663   8.4904
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  3.015e-01  2.676e-02  11.269  < 2e-16 ***
```

```
## killsgap    -3.130e-02  4.057e-03  -7.714 1.22e-14 ***
## assistsgap  -1.021e-03  1.382e-03  -0.739    0.460
## deniesgap   -1.377e-03  1.498e-03  -0.919    0.358
## networthgap  1.420e-04  2.756e-06  51.524  < 2e-16 ***
## damagegap    2.876e-05  2.890e-06   9.954  < 2e-16 ***
## levelgap     2.472e-01  6.475e-03  38.179  < 2e-16 ***
## healgap      1.265e-04  7.885e-06  16.047  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 69244  on 49999  degrees of freedom
## Residual deviance: 10006  on 49992  degrees of freedom
## AIC: 10022
##
## Number of Fisher Scoring iterations: 9
```

Net worth, damage and level are three common used for showing the game situation, and it is consistent with my expectation that they have significant positive effect on the game result. Even though number of denies can influence enemies' gold and experience point collection at early stage of each game, it seems have very limited effect on final results of the game. Surprisingly, number of kills has a negative influence on the game result, which presents a challenge to many players understanding of the game. In fact, more experience player would understand more about the nature of the game. The ultimate goal of this game is to destroy the base of the enemies and killing more enemies would bring a great amount of satisfaction but not necessarily ensure you win the game.

Then, I made prediction of game results based on the regression model and used cross-validation to evaluate the prediction accuracy.



I sampled from the match data for 100 times and in each time, I used 75% for training and 25% for testing. In most cases, the accuracy of prediction is higher than 95% percent, which mean we can easily predict the game result from players' performance.