Statistical Modelling for League of Legends Match Data

Abstract

In recent years, electronic sports (e-sports) has gradually become popular in regions covering South Korea, Europe, North America and China, and kept a fast expansion to other countries. To investigate the information, this report is about to show the modeling and analysis on game data. In details, it interprets the relationship between result (win or lose) and some explanatory variables regarding the performance of a team within each match, thereby it evaluates and compares the influence of explanatory variables, furthermore, helps to predict the result conditional on some situations for each team. At last, the best model was given by generalized additive model.

1. Motivation

In recent years, electronic sports (e-sports) has gradually become popular in regions covering South Korea, Europe, North America and China, and kept a fast expansion to other countries. League of Legends (LOL), which was a game developed by Riot Games, has been played a dominant role in the domain of electronic sports within the past decade. Some of its relevant matches like regional professional leagues, Mid-Season Invitational and World Championship consistently attract much focus from audience, public media and capital inflows. According to the statistic of Esports Charts, the number of peak global viewers was over 100 million in LOL World Championship 2017.

Accompanied by the boom of digitalization and big data, developing the application of game data turns a trend for event forecasting, strategy decision and performance analyzing. To make matches more interesting, in 2018 the official and IFLYTEK jointly produced a AI tool to make real-time prediction for win rate. Figure 1 shows an example of it in a match, as the game duration grows, KT is accordingly more possible to defeat TL. On the other hand, data analysts become more and more necessary for teams to get a deeper know of the behaviors of their players and opponents.

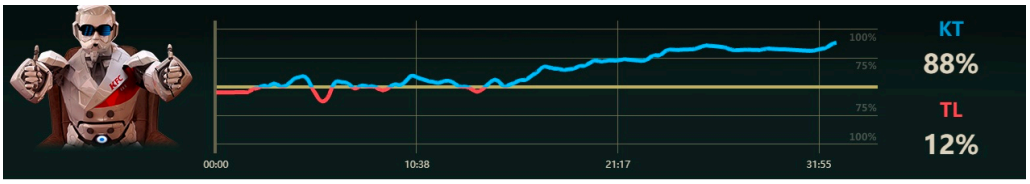

Figure 1. Real-time Win Rate Curve in the Match of KT vs. TL

Based on this background and personal interest, this study aims to process data modelling and analysis for LOL match data. In details, it interprets the relationship between result (win or lose) and some explanatory variables regarding the performance of a team within each match, thereby it evaluates and compares the influence of explanatory variables, furthermore, helps to predict the result conditional on some situations for each team.

2. Data Description

The source of the whole data set is Kaggle community (https://www.kaggle.com/kamalchouhbi/league-of-legends-match-data-2017#2017clean.csv), including League of Legends match data from the all regional professional leagues around the world in 2017. In this study, the target data set focuses on the match data in LCK, the strongest league at that time, and all matches held in the period of spring split and summer split.

The original variables can be divided into two parts in terms of game stages. One represents the information that can be obtained at the beginning or middle of match, while the other one represents that information at the end of match. Due to the purpose of this paper, variables except '*result*' in the second part are dropped because they are equivalent to '*result*' in some degree instead of cause. In other worlds, the remaining variables are meaningful for predicting '*result*' since they represents causal influence to it. Table 1 contains the basic information of all

explanatory variables and the response variable after filtering. In sum, there are totally 14 variables and 880 observations.

Variable Name	Type	Description
split	Binary	The split that a game belongs to (spring or summer).
game	Multi-class	The round in which two teams fight against each other (1,2 or 3). The rule is Best of 3 Games.
side	Binary	The side of the game (red or blue). The situations for teams in opposite sides are not perfectly equal in fact, though Riot tries to balance them.
team	Multi-class	The team name.
fb	Binary	Whether a team gets the first kill (First Blood) or not (1 - yes, 0 - no). It must happens to either of two team of the game.
fbtime	Float	The time (in minute) that First Blood happens.
fd	Binary	Whether a team slays the first dragon or not (1 - yes, 0 - no). It must happens to either of two team of the game.
fdtime	Float	The time (in minute) that First Dragon happens.
ft	Binary	Whether a team destroys the first turret or not (1 - yes, 0 - no). It must happens to either of two team of the game.
fttime	Float	The time (in minute) that First Turret happens.
cspm	Float	Creeps killed (farming) by team per minute.
goldat10	Float	Gold (in thousand) owned by a team at 10 minutes of the game.
Goldat15	Float	Gold (in thousand) owned by a team at 15 minutes of the game.
result	Binary	The response variable. Whether a team wins the game or not (1- yes, 0 - no).

Table 1. Basic Introduction to All Variables

	game	fb	fbtime	fd	fdtime	ft	fttime	cspm	goldat10	goldat15	result
mean	1.7727	0.5	6.2608	0.5	12.6905	0.5	13.3564	32.5808	15.0559	23.5849	0.5
std	0.7348	0.5003	3.6205	0.5003	3.2220	0.5003	3.3592	2.2114	0.8145	1.4289	0.5003
min	1	0	0.6478	0	6.0888	0	6.0025	25.0576	12.979	19.3360	0
25%	1	0	3.5658	0	10.2889	0	10.9750	30.9846	14.506	22.5265	0
50%	2	0.5	5.4402	0.5	12.1841	0.5	13.2266	32.5640	14.942	23.4225	0.5
75%	2	1	8.5114	1	14.6841	1	15.6333	34.0900	15.4553	24.4563	1
max	3	1	29.2994	1	25.2673	1	22.7713	39.0867	19.716	29.8380	1

Table 2. Summary of Numeric variables.

For the overview of data, Table 2 summarizes the mean, standard and percentile of variables except '*split*', '*side*' and '*result*'. Figure 2 displays the scatter plots for each pairs of the

explanatory variables in the lower panel and the histograms for each of them in the diagonal panel. As those continuous variables show, it seems the values vary so much of variable '*fbtime*', as the standard deviation is not slight, however, the points of other variables especially '*cspm*', '*goldat10*' and '*goldat15*' tend to be closer to each other as the standard variables are much smaller compared to the mean. Additionally, the diagonal panel shown in Table 2 tells only '*scpm*' is likely to be normally distributed, whereas the distribution of others seem to be more right-skewed. Also, Figure 2 shows the data is balanced in terms of the frequency of different categories. As we know, the existence of an extremely rare category usually leads to lack of representativeness for models.

For the main problem of binary classification, Figure 2 colors the data points based on the response variable '*result*'. It potentially implies that the data can be separated in some feature space.

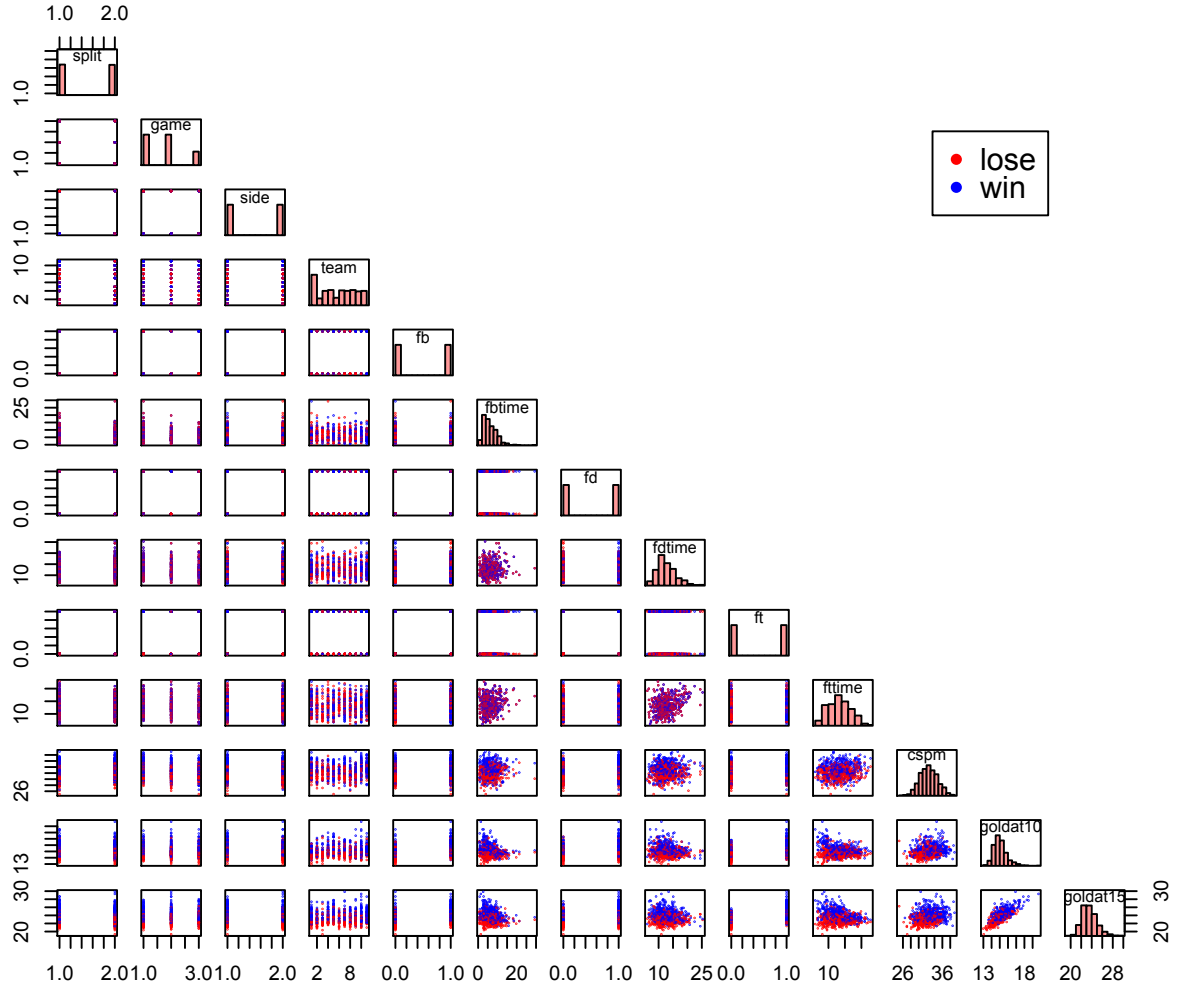

Figure 2. Scatter Plot Matrix of All Explanatory variables

3. Modelling & Methodology

Prior to modelling, the data set was split into training set and testing set with the proportion of 0.8 and 0.2. The fundamental idea was to build various models with the training set, thereafter to

make a comparison among all models with regard to the performance of fitting and predicting. During the model fitting, refinements were implemented to demonstrate some assumptions as so to make the model more capable of fitting.

3.1. Logistic Regression Model

As a simpler approach, logistic regression was initially applied to solve binary response problem as well as reveal the relationship between each two of variables.

Fit a full logistic regression model with link function $logit(\pi) = log(\frac{\pi}{1-\pi})$:

$result \,|\, x_{i,\ldots,p} \sim b(1,\pi)$, the model is in form of $logit(\hat{result}) = \boldsymbol{x}_i'\beta$,

where $x_{i,\ldots,p}$ denotes to all explanatory variables and $\boldsymbol{x}_i = (1, x_{i1}, \ldots, x_{ip})'$.

Term	Coefficient	Pr(>|z|)	Term	Coefficient	Pr(>|z|)
$Intercept$	-20.31763	2.85E-10 ***	$I_{team=ROX\ Tigers}$	1.37287	0.011356 *
$I_{split=summer}$	-0.16699	0.415036	$I_{team=SK\ Telecom\ T1}$	2.3499	4.06E-05 ***
$I_{game=1}$	-0.20702	0.452958	$I_{team=Samsung\ Galaxy}$	1.76132	0.001258 **
$I_{game=2}$	-0.18431	0.501529	$I_{fb=1}$	0.32842	0.123174
$I_{side=Red}$	-0.30225	0.121569	$fbtime$	0.01626	0.554001
$I_{team=Afreeca\ Freecs}$	1.55796	0.003719 **	$I_{fd=1}$	0.73544	0.000226 ***
$I_{team=BBQ\ Olivers}$	0.78372	0.153452	$fdtime$	-0.01102	0.717402
$I_{team=Ever8\ Winners}$	0.49402	0.496853	$I_{ft=1}$	0.97899	1.45E-05 ***
$I_{team=Jin\ Air\ Green\ Wings}$	0.86057	0.112911	$fttime$	0.03592	0.266977
$I_{team=KT\ Rolster}$	1.58471	0.00339 **	$cspm$	0.18504	0.000243 ***
$I_{team=Longzhu\ Gaming}$	1.55164	0.004377 **	$goldat10$	-0.31006	0.114715
$I_{team=MVP}$	1.12535	0.040942 *	$goldat15$	0.70306	7.49E-08 ***
Null deviance: 975.75 on 703 degrees of freedom			Residual deviance: 672.35 on 680 degrees of freedom		
AIC: 720.35 BIC: 829.7135 logLik: -336.1754 Residual deviance/df: 0.9887512					

Table 3. Fitting of Full Logistic Model (Model 1.1)

Due to so many insignificant terms, the model should be reduced to prevent from over-fitting. Besides, there were too many variables to did step-by-step LR test well enough. Here, the approach being used was to automatically find the best model among all possible candidate models by ranking with their corresponding specific information criteria, AIC, by default. This result is contained in Table 4.

For better comparison, the lines at the button of the tables gives some measurement of goodness-of-fit. Overall, the values of D / df were both approximately one, so these two models were likely to fit the data very well. For log-likelihood, exactly the model with more parameters will leads to a larger value of it, but if additionally considering the efficiency, the reduced model would be preferred due to a smaller AIC and BIC and less use of variables. Therefore, Model 1.2 shown in Table 4 tends to be the best because the number of the explanatory variables being used was decreased so much, meanwhile the goodness-of-fit was almost preserved.

Term	Coefficient	Pr(>|z|)	Term	Coefficient	Pr(>|z|)
$Intercept$	-18.96467	1.05e-10 ***	$I_{team=ROX\ Tigers}$	1.21106	0.021072 *
$I_{side=Red}$	-0.30314	0.119407	$I_{team=Samsung\ Galaxy}$	1.61478	0.002377 **
$I_{team=Afreeca\ Freecs}$	1.43222	0.005988 **	$I_{team=SK\ Telecom\ T1}$	2.15881	9.30e-05 ***
$I_{team=BBQ\ Olivers}$	0.66191	0.21777	$I_{fb=1}$	0.37249	0.074885 .
$I_{team=Ever8\ Winners}$	0.15288	0.824671	$I_{fd=1}$	0.72676	0.000245 ***
$I_{team=Jin\ Air\ Green\ Wings}$	0.72292	0.170844	$I_{ft=1}$	1.02194	4.13e-06 ***
$I_{team=KT\ Rolster}$	1.42261	0.006470 **	$cspm$	0.19201	7.95e-05 ***
$I_{team=Longzhu\ Gaming}$	1.42198	0.007664 **	$goldat10$	-0.33086	0.078415
$I_{team=MVP}$	1.01647	0.057743	$goldat15$	0.66232	1.09e-07 ***
Null deviance: 975.75 on 703 degrees of freedom			Residual deviance: 672.35 on 680 degrees of freedom		
AIC: 711.72 BIC: 793.7458 logLik: -337.8619 Residual deviance/df: 0.9850201					

Table 4. Fitting of Reduced Model by Automated Model Selection (Model 1.2)

Based on the coefficients of the reduced logistic regression model, it could be easy to interpret the importances of all predictor variables. Those of large P-value amounts to an insignificant influence on '*result*', while the smaller P-value than 0.05 contributed to more significant effect on '*result*'.

First, teams differed so much in terms of win. Teams like '*SK Telecom T1*', '*Samsung Galaxy*' '*Afreeca Freecs*' performed best among all teams, while '*Ever8 Winners*' was the weakest one. More specifically, without considering about other factors, '*SK Telecom T1*' was the best team of win games due to its highly positive coefficient of 2.1588, which means it was associated with over 700% increment in the odds of win than team '*Kongdoo Monster*'. Even compared with the second best team '*Samsung Galaxy*', there was a 72% increment in the odds of win.

Second, for the performance within a game, First Turret and First Dragon were more important to win the game than First Blood. The appearance of '*ft*' was associated with 177% of increment in the odds and the appearance of '*fd*' was associated with 100% of increment. Also, hire '*cspm*' and '*goldat15*' were beneficial with around 20% and 90% increment in the odds of win respectively.

Apart from the partially single influence, the interaction of '*team*' and other variable was capable of showing how it affected '*result*' differently for the changed values of other variable. For example, the interaction term $I_{team} : I_{side}$ could tells whether a team in various sides affected the result of the game differently or not. To answer this question, this interactive term was added to the previous reduced model, then the partial result of the new model with the interaction terms is listed in Table 5.

The new model still fitted well, but it indicated that different '*side*' did not significantly impact on the results of games for most teams. In other words, the results of games were so similar for most teams regardless of they were in '*red side*' or '*blue side*'. But what could be noticed was that '*SK Telecom T1*' increased 86% in the odds of win if it was in 'red side' rather than in '*blue side*', as $exp(I_{side=red} + I_{side=red,\ team=SK\ Telcom\ T1}) = exp(-1.63018 + 2.25095) = 1.86036.$

Term	Coefficient	Pr(>|z|)	Term	Coefficient	Pr(>|z|)
sideRed	-1.63018	0.074571 .	sideRed:teamAfreeca Freecs	1.11257	0.303635
teamAfreeca Freecs	0.88111	0.227192	sideRed:teamBBQ Olivers	1.14824	0.303456
teamBBQ Olivers	0.0912	0.903216	sideRed:teamEver8 Winners	1.36671	0.335681
teamEver8 Winners	-0.52415	0.567159	sideRed:teamJin Air Green Wings	1.10753	0.312869
teamJin Air Green Wings	0.15912	0.829081	sideRed:teamKT Rolster	1.22071	0.260734
teamKT Rolster	0.81237	0.274993	sideRed:teamLongzhu Gaming	1.5333	0.165803
teamLongzhu Gaming	0.64777	0.395354	sideRed:teamMVP	0.93483	0.394269
teamMVP	0.54965	0.459589	sideRed:teamROX Tigers	1.3986	0.197071
teamROX Tigers	0.51874	0.466366	sideRed:teamSamsung Galaxy	2.01805	0.065544 .
teamSamsung Galaxy	0.60513	0.414677	sideRed:teamSK Telecom T1	2.25095	0.045475 *
teamSK Telecom T1	0.99495	0.200073	(Other terms)	…	…
Null deviance: 975.75 on 703 degrees of freedom			Residual deviance: 669.27 on 676 degrees of freedom		
AIC: 725.27					

Table 5. Fitting of Model with the Interaction Term *team*: *side* (Partial) (Model 1.3)

3.2. Generalized Linear Mixed Model

It is reasonable to think that the same behavior affects differently to '*result*' for various teams. According to Figure 3, there exists some substantial variability among all teams for slopes and intercepts so it means that teams may have different baseline '*cspm*' as well as each unit of increment or decrement of it influences on '*result*' in various degrees. Although the logistic regression models were capable of fitting model well, in this section, mixed models with random effects were implemented to detect these extra-randomness.

To the further decline of the explanatory variables, the part of fixed effects would not comprise those insignificant variables, including '*split*', '*game*', '*fbtime*', '*fdtime*' and 'fttime', so as to avoid models failing to converge due to high dimensionality of data.

The formula of the mixed models can be written in the following form,

$logit(\hat{result}_{ij}) = \boldsymbol{X\beta} + a_i + b_i t_{ij}$, where $\boldsymbol{\beta} = (\beta_0, \ldots, \beta_p)'$ are the fixed effects; $\boldsymbol{X}$ is a $n \times (p+1)$ matrix with columns 1s; a_i is the random intercept for team i and b_i is the random slope for team i; n is the number of observations and p is the the number of variables.

To prove the guess at the beginning of this section, a mixed logistic regression model, in the form of *result ~ side + fb + fd + ft + cspm + goldat10 + goldat15 + (1 | team)*, was fitted with fixed

effects: '*side*', 'fb', '*fd*', '*ft*', '*cspm*', '*goldat10*' and '*goldat15*', and random effects: random intercepts within '*team*'. The summary of it was shown in Figure 4.

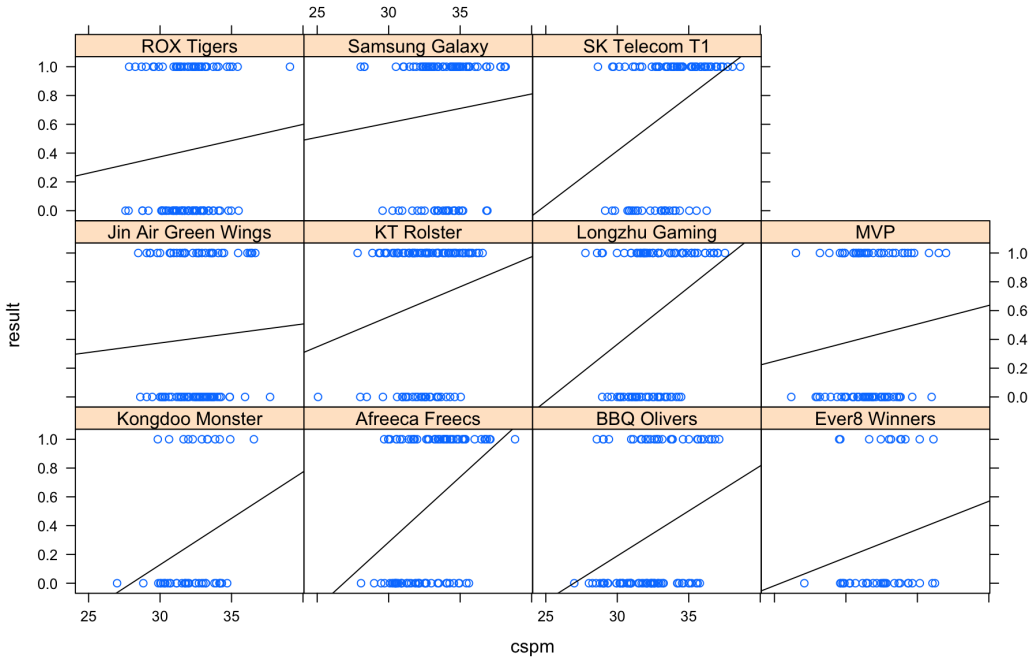

Figure 3. Bivariate Scatterplots for *cpm* among all teams

The mixed model fitted better than the logistic regression models by the value of goodness-of-fit. With likelihood ratio test between the model in the form of *result ~ side + fb + fd + ft + cspm + goldat10 + goldat15* and it, P-value equaled to 0.007793 which proved that the mixed model was significantly better. According to the coefficients of random intercepts in Table 6, the baseline of '*SK Telecom T1*' was the highest while that of '*Kongdoo Monster*' was the lowest. Nevertheless, the gap of the baselines among all teams was not rather explicit. Which implied their strength was similar.

```
Generalized linear mixed model fit by maximum likelihood (Laplace Approximation) ['glmerMod']
 Family: binomial  ( logit )
Formula: result ~ side + fb + fd + ft + cspm + goldat10 + goldat15 + (1 |      team)
   Data: trainData

     AIC      BIC   logLik deviance df.resid
   717.8    758.8   -349.9    699.8      695

Scaled residuals:
    Min      1Q  Median      3Q     Max
-4.6231 -0.5820 -0.1848  0.5852  3.6114

Random effects:
 Groups Name        Variance Std.Dev.
 team   (Intercept) 0.1989   0.4459
Number of obs: 704, groups:  team, 11

Fixed effects:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -17.99929    2.84954  -6.317 2.67e-10 ***
sideRed      -0.30125    0.19234  -1.566   0.1173
fb1           0.37082    0.20683   1.793   0.0730 .
fd1           0.76030    0.19510   3.897 9.74e-05 ***
ft1           1.00326    0.21877   4.586 4.52e-06 ***
cspm          0.20246    0.04814   4.206 2.60e-05 ***
goldat10     -0.35489    0.18600  -1.908   0.0564 .
goldat15      0.66977    0.12306   5.442 5.26e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Figure 4. Summary of Mixed Model with Random Intercepts within *team*

Coefficients	$Intercept$	$I_{side=Red}$	$I_{fb=1}$	$I_{fd=1}$	$I_{ft=1}$	$cspm$	$goldat10$	$goldat15$
Afreeca Freecs	-17.78435	-0.301245	0.3708179	0.7602985	1.003256	0.2024551	-0.354892	0.6697669
BBQ Olivers	-18.31326	-0.301245	0.3708179	0.7602985	1.003256	0.2024551	-0.354892	0.6697669
Ever8 Winners	-18.42402	-0.301245	0.3708179	0.7602985	1.003256	0.2024551	-0.354892	0.6697669
Jin Air Green Wings	-18.27701	-0.301245	0.3708179	0.7602985	1.003256	0.2024551	-0.354892	0.6697669
Kongdoo Monster	-18.5972	-0.301245	0.3708179	0.7602985	1.003256	0.2024551	-0.354892	0.6697669
KT Rolster	-17.77948	-0.301245	0.3708179	0.7602985	1.003256	0.2024551	-0.354892	0.6697669
Longzhu Gaming	-17.80009	-0.301245	0.3708179	0.7602985	1.003256	0.2024551	-0.354892	0.6697669
MVP	-18.07528	-0.301245	0.3708179	0.7602985	1.003256	0.2024551	-0.354892	0.6697669
ROX Tigers	-17.9333	-0.301245	0.3708179	0.7602985	1.003256	0.2024551	-0.354892	0.6697669
Samsung Galaxy	-17.66499	-0.301245	0.3708179	0.7602985	1.003256	0.2024551	-0.354892	0.6697669
SK Telecom T1	-17.32896	-0.301245	0.3708179	0.7602985	1.003256	0.2024551	-0.354892	0.6697669

Table 6. Coefficients of Mixed Model with Random Intercepts within Teams

Furthermore, to detect whether teams have different effects on '*result*' of '*cspm*', random slopes were added but unfortunately, this time the new models failed to converge. As we can see, the correlation of random slopes was -1 in Figured 5 that indicated the model may be overfitting. The problems remained when random intercepts were dropped. In other words, more random effects term than intercepts only were not necessary in this case.

```
Generalized linear mixed model fit by maximum likelihood (Laplace Approximation) ['glmerMod']
 Family: binomial  ( logit )
Formula: result ~ side + fb + fd + ft + cspm + goldat10 + goldat15 + (1 +      cspm | team)
   Data: trainData

     AIC      BIC   logLik deviance df.resid
   720.6    770.7   -349.3    698.6      693

Scaled residuals:
    Min      1Q  Median      3Q     Max
-4.3820 -0.5752 -0.2036  0.5804  3.7196

Random effects:
 Groups Name        Variance Std.Dev. Corr
 team   (Intercept) 3.402125 1.84449
        cspm        0.004832 0.06951  -1.00
Number of obs: 704, groups:  team, 11
```

Figure 5. Summary of Mixed Model with Random Intercepts and Slopes within Teams

3.3. Generalized Additive Model

In the previous sections, logistic regression models and mixed models both applied a link function (logit) that non-linearly transformed the linear relationship between the response variable and the explanatory variables, and then provided directed and comprehensible interpretation. But in fact,

some explanatory variables, probably, are not linearly related to the response variable. Therefore, in generalized additive models, smoothing terms are used in the absence of linear relationship between the responser and the predictors.

This time all continuous variables would be retained for possible significance as smoothing components, but the insignificant factor variables which were proved before would be excluded because additive models did not revise any of them. The initial additive model with smoothing terms for all continuous variables was shown in the following form, and its summary was shown in Figure 6.

$$logit(\hat{result}) = \boldsymbol{X\beta} + \sum_{i}^{j} \hat{g}_i(X_i),$$

where $\boldsymbol{\beta}$ is the coefficients of factor variables, $\boldsymbol{X}$ is a $n \times (p - j + 1)$ matrix with columns 1s and $\hat{g}_i(X_i)$ is the nonparametric components (include linear trend) of continuous variable i.

```
Family: binomial
Link function: logit

Formula:
result ~ side + team + fb + fd + ft + s(fbtime) + s(fttime) +
    s(fdtime) + s(cspm) + s(goldat10) + s(goldat15)

Parametric coefficients:
                       Estimate Std. Error z value Pr(>|z|)
(Intercept)             -2.0103     0.5096  -3.945 7.99e-05 ***
sideRed                 -0.3012     0.1967  -1.532 0.125639
teamAfreeca Freecs       1.4275     0.5321   2.683 0.007302 **
teamBBQ Olivers          0.6196     0.5506   1.125 0.260472
teamEver8 Winners        0.2287     0.7161   0.319 0.749462
teamJin Air Green Wings  0.7301     0.5392   1.354 0.175761
teamKT Rolster           1.4676     0.5336   2.750 0.005954 **
teamLongzhu Gaming       1.4035     0.5429   2.585 0.009739 **
teamMVP                  0.9698     0.5438   1.783 0.074525 .
teamROX Tigers           1.2768     0.5362   2.381 0.017261 *
teamSamsung Galaxy       1.6355     0.5425   3.015 0.002573 **
teamSK Telecom T1        2.0620     0.5693   3.622 0.000292 ***
fb1                      0.3406     0.2148   1.586 0.112774
fd1                      0.7685     0.2012   3.820 0.000134 ***
ft1                      0.9624     0.2267   4.244 2.19e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Approximate significance of smooth terms:
              edf Ref.df Chi.sq  p-value
s(fbtime)   1.000  1.000  0.389 0.533046
s(fttime)   1.000  1.000  1.384 0.239499
s(fdtime)   1.000  1.000  0.055 0.814707
s(cspm)     3.871  4.904 20.934 0.000724 ***
s(goldat10) 1.000  1.000  1.861 0.172518
s(goldat15) 1.000  1.000 28.858  7.8e-08 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

R-sq.(adj) =  0.363   Deviance explained = 32.5%
UBRE = 0.0039162  Scale est. = 1         n = 704
```

Figure 6. Summary of Additive Model with Smoothing Components for All Continuous Variables

In the presence of P-value over 0.05, the nonparametric components of '*fbtime*', '*fttime*', '*fdtime*' and '*goldat10*' were insignificant again, contrarily the smooth terms of '*scpm*' and '*goldat15*' were rather significant. According to the partial prediction plots in Figure 7, only the curve of '*cspm*' was explicitly non-linear, while the other significant term '*goldat15*' was in the full of linearity.

To refine the model, all significant variables were removed and linear component '*goldat15*' was merely retrained. The comparison between the refined model and the original one was given in Figure8. As the result shows, the refined model fitted as well as the original one.

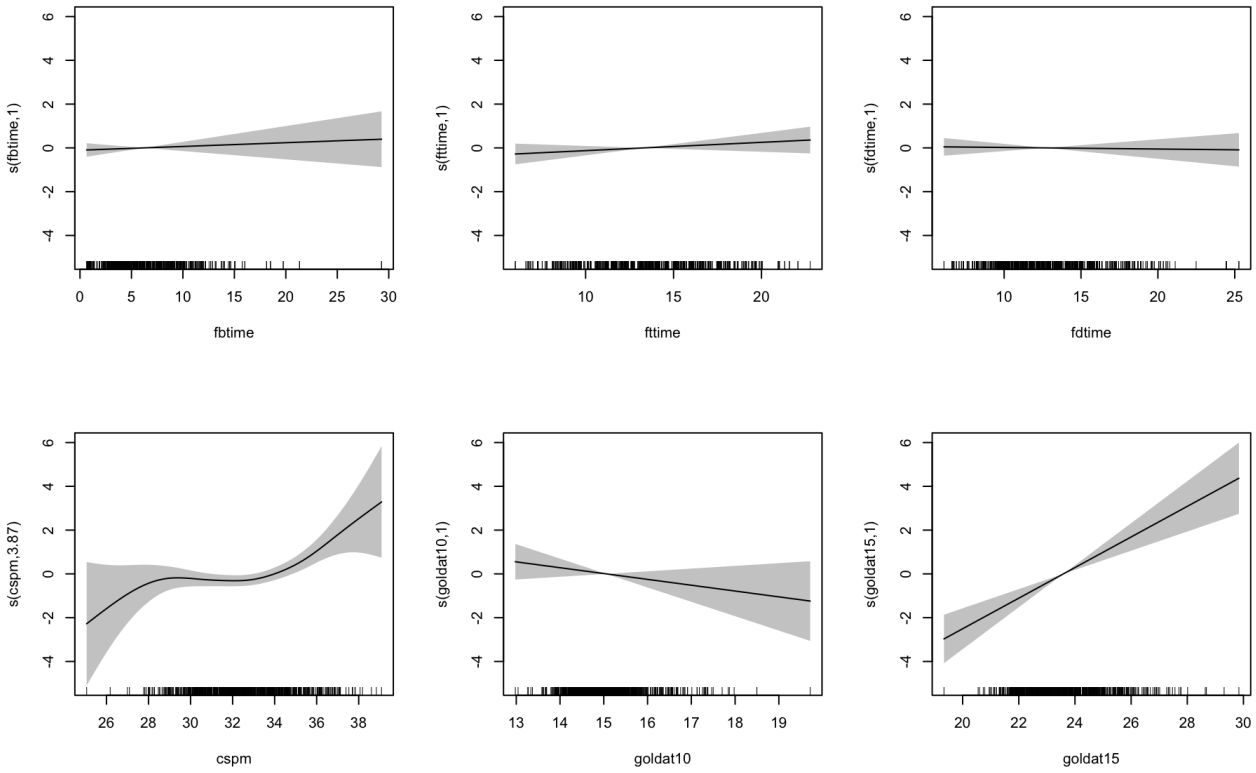

Figure 7. Partial Prediction Plots

```
Analysis of Deviance Table

Model 1: result ~ side + team + fb + fd + ft + s(fbtime) + s(fttime) +
    s(fdtime) + s(cspm) + s(goldat10) + s(goldat15)
Model 2: result ~ side + team + fb + fd + ft + s(cspm) + goldat15
  Resid. Df Resid. Dev      Df Deviance Pr(>Chi)
1    679.10     659.01
2    683.03     663.42 -3.9355  -4.4048   0.3451
```

Figure 8. Comparison of Two Models Using Deviance Test

3.4. Bayesian Network

Bayesian network is a graphic model for making prediction baed on conditional independencies among a set of random variables. In this study, due to some continuous variables, there were two approaches being used to implement Bayesian networks:

- Hybrid BN (assume those continuous variables follow normal distribution)
- Discrete BN (discretize those continuous variables)

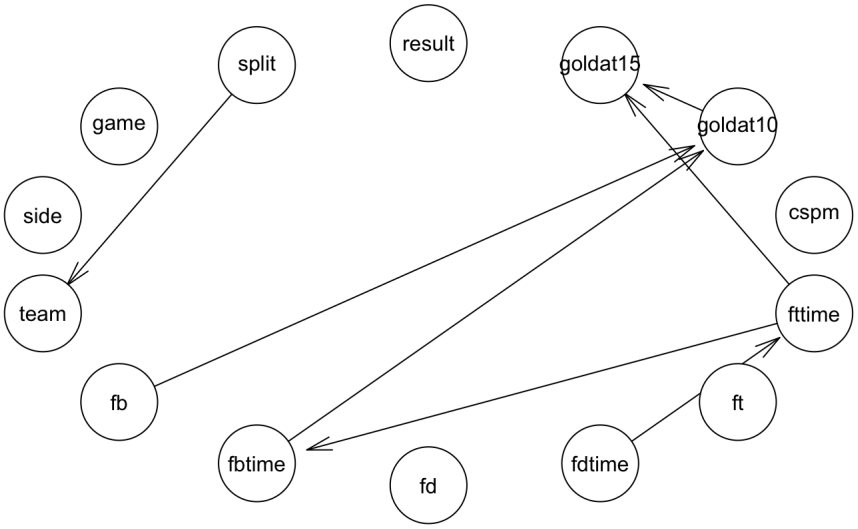

Figure 9. Graph of Hybrid Network

In the first approach, the DAG was learnt from data using Max-Min Hill-Climbing algorithm. The model string from it was [split][game][side][fb][fd][fdtime][ft][cspm][result][team|split][fttime|fdtime] [fbtime|fttime][goldat10|fb:fbtime][goldat15|fttime:goldat10] and the corresponding network graph was shown in Figure 9. The joint probability could be written in the following formula.

$$\begin{aligned} & P(split, game, side, team, fb, fbtime, fd, fdtime, \\ & \quad ft, fttime, cspm, goldat10, goldat15, result) \\ = & P(split)P(game)P(side)P(fb)P(fd)P(fdtime)P(ft)P(cspm) \\ & P(result)P(team \mid split)P(fttime \mid fdtime)P(fbtime \mid fttime) \\ & P(goldat10 \mid fb : fbtime)P(goldat15 \mid fttime : goldat10) \end{aligned}$$

After estimating the parameters based on the DAG learnt by MLE, In Figure 10, the inference of node '*goldat10*' and ''*goldat15*' were displayed as instances. For node '*goldat10*', given '*fb*' and '*fbtime*', the value of '*goldat10*' followed:

$$goldat10 \mid_{fb=0, fbtime} \sim N(14.87037936 - 0.02670403 fbtime,\ 0.6675338^2)$$

$$goldat10 \mid_{fb=1, fbtime} \sim N(15.92632681 - 0.08226725 fbtime,\ 0.7708188^2)$$

For node of '*goldat15*', given '*fttime*' and '*goldat10*', the value of '*goldat15*' followed:

$$goldat15 \mid_{fttime, goldat10} \sim N(4.75729976 - 0.03210497 fftime + 1.27880273 goldat10,\ 0.9537047^2)$$

In terms of the proof of conditional Guassian distribution for continuous nodes, the relevant QQ-plot was display in Figure 11. Some of nodes did not exactly follow this assumption.

```
  Parameters of node goldat10 (conditional Gaussian distribution)

Conditional density: goldat10 | fb + fbtime
Coefficients:
                        0           1
(Intercept)  14.87037936  15.92632681
fbtime       -0.02670403  -0.08226725
Standard deviation of the residuals:
        0         1
0.6675338  0.7708188
Discrete parents' configurations:
   fb
0   0
1   1

  Parameters of node goldat15 (Gaussian distribution)

Conditional density: goldat15 | fttime + goldat10
Coefficients:
(Intercept)       fttime     goldat10
 4.75729976  -0.03210497   1.27880273
Standard deviation of the residuals: 0.9537047
```

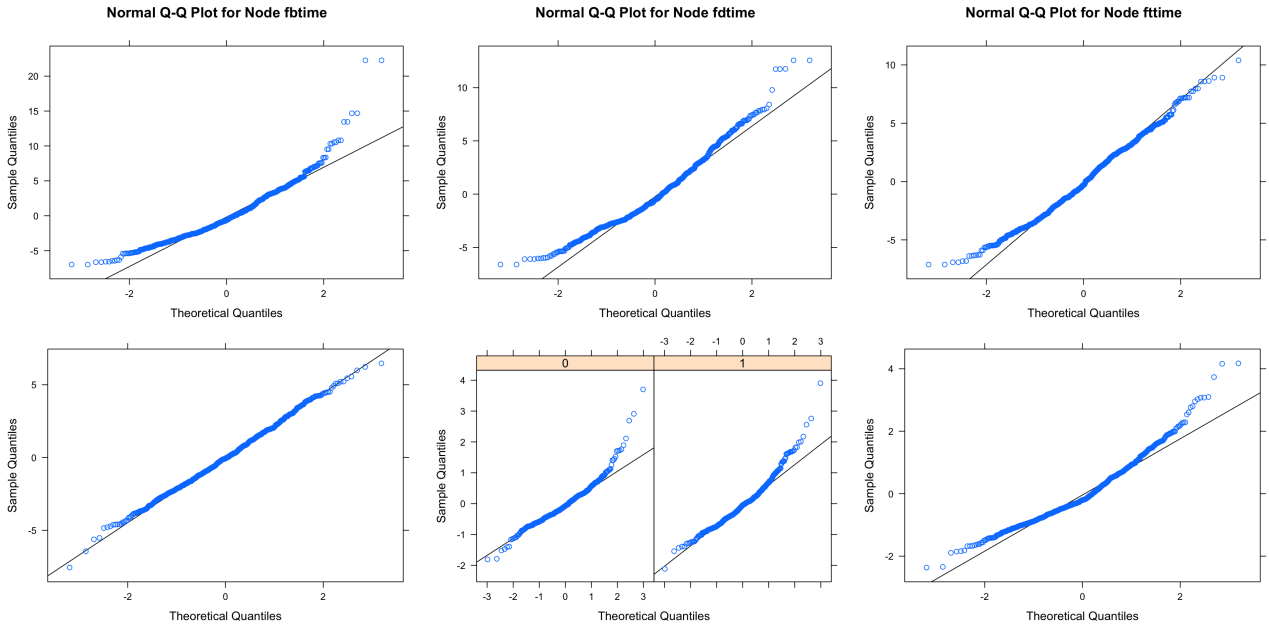

Figure 10. Parameters of Hybrid Network (*goldat10* and *gold15*)

Figure 11. QQ-Plots of Continuous Nodes of Hybrid Network

In the second approach, the data was discretized prior to learning a DAG. The method for discretization in this case was interval discretization instead of quantile discretization in this case, thereby the labels produced were capable of indicating the different strength of teams. Exactly, the most outstanding or poorest performance should be rarer. Besides, another factor should be concerned was how many labels of each variables to be discretized. Here, the basic idea was iteratively using a numbers of breaks for all continuous variables, thereafter evaluated all models according to their scores. All networks would learn the DAG from data using the Hill-Climbing algorithm, and then estimate the parameters by MLE.

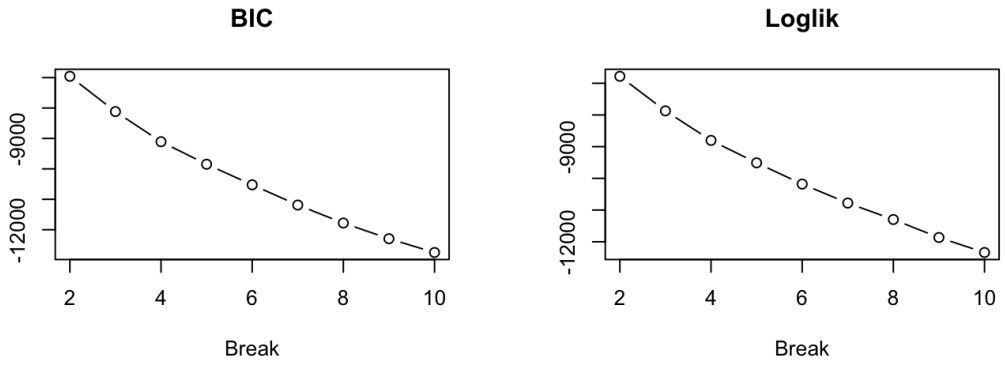

Figure 12. Summary of all Discretized Networks with Various Breaks

According to Figure 12, it was easy to find out when the number of breaks was equal to 2, the model was best at fitting the data. Figure 13 showed the graph of the Bayesian network and The model string from it was [side][fdtime][ft|side][fttime|fdtime][goldat15|ft:fttime][result|ft:goldat15] [fbtime|goldat15][goldat10|goldat15][fb|result:goldat15][fd|result:goldat15][cspm|result][split|cspm] [game|cspm][team|split]. The joint probability could be written in the following formula.

$$\begin{aligned} &P(split, game, side, team, fb, fbtime, fd, fdtime, \\ &ft, fttime, cspm, goldat10, goldat15, result) \\ &= P(side)P(fdtime)P(ft|side)P(fttime|fdtime)P(goldat15|ft:fttime) \\ &P(result|ft:goldat15)P(fbtime|goldat15)P(goldat10|goldat15)P(fb|result:goldat15) \\ &P(fd|result:goldat15)P(cspm|result)P(split|cspm)P(game|cspm)P(team|split) \end{aligned}$$

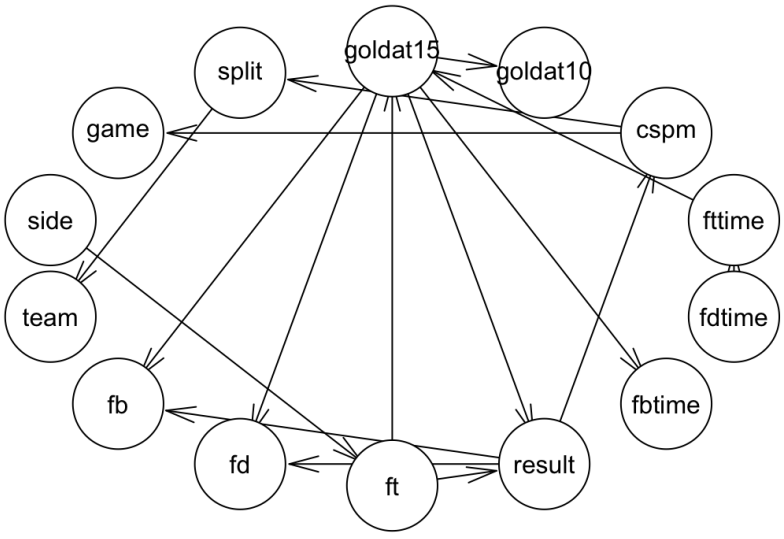

Figure 13. Graph of Discretized Network

Compared to the previous network model, this time the parents of '*result*' could be pointed out, it was conditionally dependent on '*ft*' and '*goldat15*' under the assumption of Bayesian networks model. For easy observation, Figure 14 shows the probability of win or lose in 'result' conditional on various '*ft*' and '*goldat15*' by bar chart. Within a game, if the '*cspm*' of the team was more than

24.6 and it got First Tower, the win rate was over 0.85; if the '*cspm*' of the team was more than 24.6 but it failed to get First Tower, the win rate was decreased to 0.68; if the '*cspm*' of the team was less than 24.6 but it got First Tower, the win rate was decreased to 0.61; if the '*cspm*' of the team was less than 24.6 and it failed to got First Tower, the win rate was only 0.25.

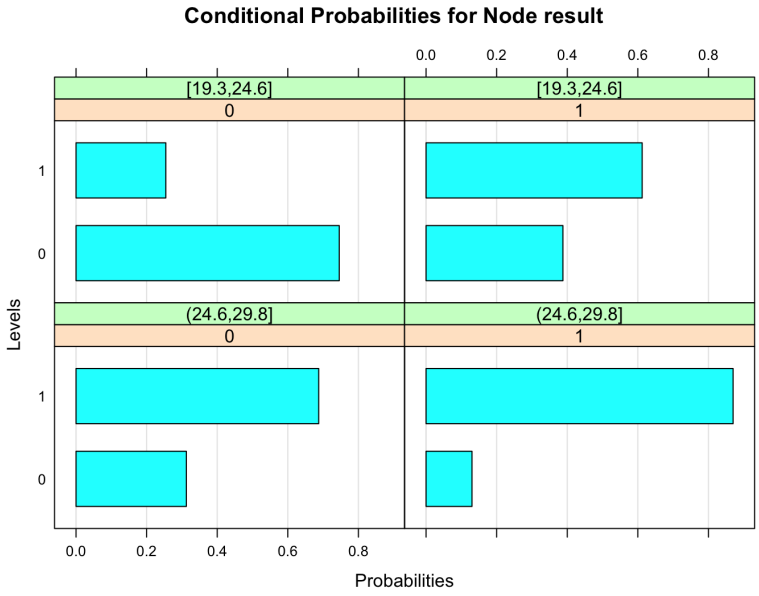

Figure 14. Bar Chart of node '*result*' of Discretized Network

4. Conclusion

In the part of first three models, logistic regression model seemed to fit the data very well but both mixed mode and additive model were successful to make a further improvement.

For the comparison among all teams, 'SK Telecom T1' was the strongest one while '*Kongdoo Monster*' was the weakest one. For the performance within a game, the most significant variables to affect the result were commonly indicated as '*ft*', '*cspm*' and '*goldat15*' which all made positive influence on the result, but only '*cspm*' was proved to nonlinearly relate to the result. In addition, those insignificant terms like 'game' and 'side' demonstrated a slight impact on the result.

	Loglik (score)	AIC	BIC (score)	Test Accuracy	AUC
Full Logistic Regression Model	-336.1754	720.35	829.7135	0.7613636	0.8385
Reduced Logistic Regression Model	-337.8619	711.72	793.7458	0.7556818	0.8368
Mixed Model with Random Intercepts	-349.9	717.8	758.8	0.7670455	0.8422
Additive Model with Smoothing Components for '*cspm*'	-331.7097	703.2729	794.0747	0.7840909	0.853
Hybrid Bayesian Network	For discrete data only		-14273.02	0.4715909	0.7244
Discretized Bayesian Network	-6780.665		-6957.698	0.7386364	0.7710171

Table 7. Summary of All Models

Table 7 summarizes the performance of all appeared models in terms of fitting and forecasting. As we can see, the additive model fitted and predicted the best among all models. For the Bayesian networks, the discretized network seemed to be better than hybrid network. To improve the first one, it required more concern about how to discretize the data more appropriately.

5. Appendix: R code

```
setwd('/Users/kariya/Documents/STAT6014/Ass2')

library(pROC)
library(glm2)
library(glmnet)

DataForWR = read.csv('LOL-LCK-2017.csv')

# Data description
str(DataForWR)
summary(DataForWR)

# pairs plot
# Customize upper panel
lowerPanel = function(x, y){
  points(x,y, pch = 20, cex = 0.1, col = c("red", "blue")[as.factor(DataForWR$result)])
}

# Customize diagonal panel
panelHist = function(x) {
  usr = par("usr"); on.exit(par(usr))
  par(usr = c(usr[1:2], 0, 1.5))
  h = hist(x, plot = FALSE)
  breaks = h$breaks; nB = length(breaks)
  y = h$counts; y = y/max(y)
  rect(breaks[-nB], 0, breaks[-1], y, col = '#FF9999')
}

# Create the plots
pairs(DataForWR[,-14],
      diag.panel = panelHist,
      lower.panel = lowerPanel,
      upper.panel = NULL
      )

par(xpd=TRUE)
legend('topright', col = c("red", "blue"), pch = 20, legend = c("lose", "win"))

# Split the data into training and testing sets
unique(DataForWR$team)
DataForWR$team = relevel(DataForWR$team, ref = 'Kongdoo Monster')
DataForWR$game = as.factor(DataForWR$game)
DataForWR$game = relevel(DataForWR$game, ref = '3')
DataForWR$fb = as.factor(DataForWR$fb)
DataForWR$fd = as.factor(DataForWR$fd)
DataForWR$ft = as.factor(DataForWR$ft)
str(DataForWR)

n = nrow(DataForWR)
set.seed(6014)
testRows = sample(c(1:n), floor(n*0.2), replace = F)
trainData = DataForWR[-testRows,]
testData = DataForWR[testRows,]

# logistic regression model
lr.full = glm2(result~., family = binomial(link = 'logit'), data = trainData)
```

```
summary(lr.full)

# refine the model

# find the best model in another way
fitall = glmulti(result~., family = binomial(link = 'logit'), data = trainData, level = 1, method =
'h', crit = 'aicc')
weightable(fitall) # formula of best model: result ~ 1 + side + team + fb + fd + ft + cspm +
goldat10 + goldat15, which is the same as lr.reduce8
lr.reduce = glm2(result ~ 1 + side + team + fb + fd + ft + cspm + goldat10 + goldat15, family =
binomial, data = trainData)

# compare with full model
summary(lr.full)
summary(lr.reduce)

# logLik
logLik(lr.full) # -336.1754
logLik(lr.reduce) # -337.8619

# GOF
summary(lr.full)$deviance / summary(lr.full)$df[2];
summary(lr.reduce)$deviance / summary(lr.reduce)$df[2]
# 0.9887512; 0.9850201

# AIC
AIC(lr.full); AIC(lr.reduce)
# 720.35; 711.72

# BIC
BIC(lr.full); BIC(lr.reduce)
# 829.7135;  793.7458

# with interaction term
lr.int1 = glm2(result ~ 1 + side + team + fb + fd + ft + cspm + goldat10 + goldat15 + side:team,
family = binomial, data = trainData)
summary(lr.int1)

exp(-1.63018+2.25095)

# test accuracy
pred.full.prob = predict(lr.full, newdata = testData[,-14], type = 'response')
pred.full.label = ifelse(pred.full.prob > 0.5, 1, 0)
pred.reduce.prob = predict(lr.reduce, newdata = testData[,-14], type = 'response')
pred.reduce.label = ifelse(pred.reduce.prob > 0.5, 1, 0)
sum(pred.full.label == testData[,14]) / nrow(testData);
sum(pred.reduce.label == testData[,14]) / nrow(testData);
# 0.7613636; 0.7556818


roccurve.lr.full = roc(testData$result, pred.full.prob) # 0.8385
roccurve.lr.reduce = roc(testData$result, pred.reduce.prob) # 0.8368

plot.roc(roccurve.lr.full, print.thres = "best", col = 'red')
plot.roc(roccurve.lr.reduce,  print.thres = "best", add = TRUE, col = 'blue')
lfull = paste0("Full (",round(auc(roccurve.lr.full),4),")")
lreduce = paste0("Reduced (",round(auc(roccurve.lr.reduce),4),")")
legend("bottomright", legend=c(lfull, lreduce),
       col=c("red", "blue"), lwd=2)
title('Comparison of ROC Curves of Logistic Regression Models', line = 3)

setwd('/Users/kariya/Documents/STAT6014/Ass2')

library(HH)
library(pROC)
library(glm2)
```

```
library(lme4)
library(lmtest)

DataForWR = read.csv('LOL-LCK-2017.csv')

# Split the data into training and testing sets
DataForWR$game = as.factor(DataForWR$game)
DataForWR$fb = as.factor(DataForWR$fb)
DataForWR$fd = as.factor(DataForWR$fd)
DataForWR$ft = as.factor(DataForWR$ft)
str(DataForWR)

xyplot(result~cspm|team,
       panel = function(x, y) {
         panel.xyplot(x, y)
         panel.lmline(x, y)
       },
       data = DataForWR)

xyplot(result~goldat15|team,
       panel = function(x, y) {
         panel.xyplot(x, y)
         panel.lmline(x, y)
       },
       data = DataForWR)

n = nrow(DataForWR)
set.seed(6014)
testRows = sample(c(1:n), floor(n*0.2), replace = F)

trainData = DataForWR[-testRows,]
testData = DataForWR[testRows,]

# recall the best-fitting model in the part of logistic regression
lr = glm2(result ~ side + fb + fd + ft + cspm + goldat10 + goldat15, family = binomial, data =
trainData)
# AIC = 711.72; BIC = 793.7458; logLik = -337.8619; test_acc = 0.7556818; auc = 0.8368

# random intercept
glm1.1 = glmer(result~side + fb + fd + ft + cspm + goldat10 + goldat15+(1|team), family = binomial,
data = trainData)
lrtest(lr, glm4.1) # 0.457

# random slope for cspm
glm1.2 = glmer(result~side + fb + fd + ft + cspm + goldat10 + goldat15+(1+cspm|team), family =
binomial, data = trainData)
# Model failed to converge

# remove random intercepts
glm1.3 = glmer(result~side + fb + fd + ft + cspm + goldat10 + goldat15+(0+cspm|team), family =
binomial, data = trainData)

# random slope for goldat15
glm1.3 = glmer(result~side+fd+ft+cspm+goldat15+(1+goldat15|team), family = binomial, data =
trainData)

# predict
pred.1.1.prob = predict(glm1.1, testData, type = 'response')
pred.1.1.label = ifelse(pred.1.1.prob>0.5, 1, 0)
sum(pred.1.1.label == testData$result) / nrow(testData) # 0.7670455
roccurve.1.1 = roc(testData$result, pred.1.1.prob) # 0.8422
plot.roc(roccurve.1.1, print.thres = "best", print.auc = T, col = 'red')
title('ROC Curves of Generalized Linear Mixed Models with Random Intercepts', line = 3)


setwd('/Users/kariya/Documents/STAT6014/Ass2')
```

```
library(mgcv)
library(pROC)

DataForWR = read.csv('LOL-LCK-2017.csv')

# Split the data into training and testing sets
DataForWR$game = as.factor(DataForWR$game)
DataForWR$game = relevel(DataForWR$game, ref = '3')
DataForWR$team = relevel(DataForWR$team, ref = 'Kongdoo Monster')
DataForWR$fb = as.factor(DataForWR$fb)
DataForWR$fd = as.factor(DataForWR$fd)
DataForWR$ft = as.factor(DataForWR$ft)

n = nrow(DataForWR)
set.seed(6014)
testRows = sample(c(1:n), floor(n*0.2), replace = F)

trainData = DataForWR[-testRows,]
testData = DataForWR[testRows,]

gam0 = gam(result~side + team + fb + fd + ft + cspm + goldat10 + goldat15, family = binomial, data =
trainData)

gam1 = gam(result~side+team+fb+fd+ft+s(fbtime)+s(fttime)+s(fdtime)+s(cspm)+s(goldat10)+s(goldat15),
family = binomial, data = trainData)
summary(gam1)

plot(gam1, pages = 1, se = TRUE, shade = TRUE)

gam2 = gam(result~side+team+fb+fd+ft+s(cspm)+goldat15, family = binomial, data = trainData)
anova(gam1, gam2, test = "Chisq") # no significantly difference

summary(gam2)
AIC(gam2); BIC(gam2); logLik(gam2)
# 703.2729; 794.0747; -331.7097

# predict
pred.prob = predict(gam2, testData, type = 'response')
pred.label = ifelse(pred.prob>0.5, 1, 0)
sum(pred.label == testData$result) / nrow(testData) # 0.7840909
roccurve = roc(testData$result, pred.prob)
auc(roccurve) # 0.853
plot.roc(roccurve, print.auc = TRUE, print.thres = "best", print.thres.best.method =
"closest.topleft")


setwd('/Users/kariya/Documents/STAT6014/Ass2')

library(pROC)
library(bnlearn)

DataForWR = read.csv('LOL-LCK-2017.csv')

# Split the data into training and testing sets
DataForWR$game = as.factor(DataForWR$game)
DataForWR$fb = as.factor(DataForWR$fb)
DataForWR$fd = as.factor(DataForWR$fd)
DataForWR$ft = as.factor(DataForWR$ft)
DataForWR$result = as.factor(DataForWR$result)

n = nrow(DataForWR)
set.seed(6014)
testRows = sample(c(1:n), floor(n*0.2), replace = F)

# Hybrid Bayesian Newtwork
```

```
trainData = DataForWR[-testRows,]
testData = DataForWR[testRows,]

mmhcdag = mmhc(trainData, whitelist = NULL, blacklist = NULL)
plot(mmhcdag)
modelstring(mmhcdag)

mmmhcfit = bn.fit(mmhcdag, trainData, method = 'mle')
for (i in c(6,8,10:13)){
  bn.fit.qqplot(mmmhcfit[[i]])
}

pred = predict(mmmhcfit, data = testData, node = 'result', method = 'bayes-lw')
sum(pred == testData$result) / nrow(testData) # 0.4715909

# Bayesian network

roccurve.hc = roc(testData_d$result, pred.hc.prob) #  0.7244
plot.roc(roccurve.hc, print.thres = "best", col = 'red')
# plot.roc(roccurve.lr.reduce,  print.thres = "best", add = TRUE, col = 'blue')
# lfull = paste0("Full (",round(auc(roccurve.lr.full),4),")")
# lreduce = paste0("Reduced (",round(auc(roccurve.lr.reduce),4),")")
# legend("bottomright", legend=c(lfull, lreduce), col=c("red", "blue"), lwd=2)
#title('Comparison of ROC Curves of Logistic Regression Models', line = 3)


optimal_break = function(num_break,
                         discretize_method = 'interval',
                         test_row,
                         plot_ROC = FALSE, output_model = FALSE){
  # Discretize the continuous variables by interval discretization
  d = discretize(DataForWR[,c(6,8,10:13)], discretize_method, num_break)

  # Split the disretized data into training set and tesing set with the same seed as before
  tempTrainData = cbind(DataForWR[-test_row,-c(6,8,10:13)], d[-test_row,])
  tempTestData = cbind(DataForWR[test_row,-c(6,8,10:13)], d[test_row,])
  tempTrainData$result = as.factor(tempTrainData$result)
  tempTestData$result = as.factor(tempTestData$result)

  # fit a Bayesian Network
  temphcdag = hc(tempTrainData)
  tempBIC = score(temphcdag, data = tempTrainData)
  tempLoglik = score(temphcdag, data = tempTrainData, type = 'loglik')
  temphcfit = bn.fit(temphcdag, tempTrainData, method = "mle")

  # predict
  temppred.label = predict(temphcfit, tempTestData, node = 'result', prob = TRUE)
  temppred.prob = attr(temppred.label, which = 'prob')[2,]
  tempacc = sum(temppred.label == tempTestData$result) / nrow(tempTestData)

  # ROC curve
  temproccurve = roc(tempTestData$result, temppred.prob)
  tempauc = auc(temproccurve)
  if (plot_ROC){
    plot.roc(temproccurve, print.auc = TRUE, print.thres = "best", print.thres.best.method =
"closest.topleft")
    }
  if (output_model){
    return(list(DAG = temphcdag, fit = temphcfit))
  }
  else {
    return(list(BIC = tempBIC, loglik = tempLoglik, test_acc = tempacc, auc = tempauc))
  }
}

bicList = c(); logitList = c(); accList = c(); aucList =c()
```

```
for (b in c(2:10)){
  tempResult = optimal_break(num_break = b, discretize_method = 'interval', test_row = testRows)
  bicList = c(bicList, tempResult$BIC)
  logitList = c(logitList, tempResult$loglik)
  accList = c(accList, tempResult$test_acc)
  aucList = c(aucList, tempResult$auc)
}

par(mfrow = c(2,2))
plot(x = c(2:10), y = bicList, type = 'b', ylab = '', xlab = 'Break')
title('BIC')
plot(x = c(2:10), y = logitList, type = 'b', ylab = '', xlab = 'Break')
title('Loglik')
plot(x = c(2:10), y = accList, type = 'b', ylab = '', xlab = 'Break')
title('Test Accuracy')
plot(x = c(2:10), y = aucList, type = 'b', ylab = '', xlab = 'Break')
title('AUC')
par(mfrow = c(1,1))

i = 1 # break = 2
bicList[i]; logitList[i]; accList[i]; aucList[i]
'''
[1] -6957.698
[1] -6780.665
[1] 0.7386364
[1] 0.7710171
'''

# Recall the best BN
list1 = optimal_break(num_break = 2, test_row = testRows, plot_ROC = T, output_model = T)
plot(list1$DAG)
modelstring(list1$DAG)

for(var in colnames(trainData)){
  print(list1$fit[[var]])
  bn.fit.barchart(list1$fit[[var]])
}

# plot discretized variables
d = discretize(DataForWR[,c(6,8,10:14)], 'interval', 3)
par(mfrow=c(3,3))
for(col in colnames(d)){
  icol = which(colnames(DataForWR) == col)
  icold = which(colnames(d) == col)
  plot(DataForWR[,icol],col=d[,icold], xlab = '', ylab = '')
  legend("topright",legend = unique(d[,icold]), col = c(1:3), pch = 1, cex = 0.8)
  title(main = paste0(col))
}
par(mfrow=c(1,1))

cpquery(list1$fit, event = (result == '0'), evidence = (team == 'KT Rolster' & ft == '1' & fd ==
'1'), n = 10000)
```