MTECH KE-5107 (DMM) PROJECT REPORT

LEAGUE OF LEGENDS DATA ANALYSIS

TEAM FT01

ANURAG CHATTERJEE (A0178373U)
BHUJBAL VAIBHAV SHIVAJI (A0178321H)
CHARLES THOMAS DE LEAU (A0178316Y)
LIM PIER (A0178254X)
LIU THEODORUS DAVID LEONARDI (A0178263X)
TSAN YEE SOON (A0178316Y)

MASTER OF TECHNOLOGY IN
KNOWLEDGE ENGINEERING
BATCH KE-30(2018)

Contents

Executive Summary

The eSports Industry has experienced tremendous growth over the past few years having a market value of over US$ 1.5 billion in 2017. With the increase in popularity and wider mainstream coverage of games as a sport, the eSports wagering industry grew. More established Licenced and Legal betting operators worldwide are exploring this new segment. To be able provide the correct betting odds requires new understanding and analysis.

In this project, we assume the role of a team of analysts working for a Licenced and Legal Betting operator, who is looking to explore the possibility of providing game-in-progress bets for the most popular eSports game in the world, the League of Legends. To be able to provide odds independently, the betting operator will require a model that is able to generate internal base reference betting odds, and this model is our business success objective. To achieve our business objective, we determine that our datamining goal is to create predictive models that are to predict the winner of competitive League of Legends matches.

We followed the CRISP-DM methodology for this project. The data used in this predictive task is sourced from Kaggle. We only used professional team competition matches data as these are the type of matches that the business is interested in. The data contains 7058 observations and 57 variables. After further data understanding and preparation with increased domain knowledge including conducting association analysis, we discarded unnecessary columns and derived new variables to construct the sample dataset we used for building our model.

Logistics regression, decision trees, random forest, xgboost, neural networks and support vector machines were included as our predictive models. Models mentioned were implemented and their performance parameters were evaluated. Grid search was used to obtain the optimum hyper-parameters modeling techniques. Cross-validation was also used to further validate model performance. We observed that logistic regression is one of the better models for our dataset.

The evaluation results show that it is possible to predict the winner of the matches with 71.3% accuracy at the beginning stages of the game and the accuracy of the models increases over time as the game progress, up to 75% and 83% accuracy at the 10 min. and 20 min. mark into the game respectively. Other model's measurements such as F1 score, recall, precision, true positive rate support the above findings. The change in accuracy of the prediction models will enable the betting operator to compute new internal reference betting odds which they can use to skew or shift actual betting odds it provided to its bettors. Additional findings from the association analysis, we also discovered that "champions bans" usually happened together, this insight may allow the betting operator to offer new kinds of sub-bets offering such as betting on the next champion selected to be banned with adjusted odds from the analysis.

In conclusion, we successfully completed the data analytics and data mining project. We selected suitable and relevant data set for the objectives; performed successful exploratory data analysis; prepared the data and built successful predictive models with appropriate validation.

1.0 Business Understanding

1.1 Introduction

The eSports industry brought in over $1.5 billion total revenue in 2017 and is on course to hit $2.3 billion by 2022 (SuperData: Games & Interactive Media Intelligence, 2018). Mainstream coverage of eSports has also increased tremendously, traditional media groups such as ESPN have also started dedicated eSports coverage. As with any popular sport, the increasing mainstream exposure of eSports has also created more opportunities for wagering. More established Licenced and Legal Betting Operators are now setting up shop to try to reap the rewards of this new "sports" segment where the total amount wagered has reached US$550 million in 2016 (Statista, 2018). In an announcement last year, established betting operator Pinnacle reported that it had already taken its five millionth eSports bet and expects to double its takings in 2018 (Pinnacle, 2017). The eSports segment is now their 7th largest market for them in terms of volume, exceeding regular sports like golf and rugby.

1.2 Business Objectives

We assume the role of a team of analysts working for a Licensed and Legal Betting Operator (e.g. Singapore Pools in Singapore; Ladbrokes in UK/EU), who is looking to expand its bet offerings to the new eSports segment. To stand out from the competition, the company is exploring live betting opportunities at different points of the game-in-progress for the most popular global eSports team game, the League of Legends (LoL) (Dallas Observer, 2017). For this purpose, there is a need to create a model that can generate correct reference betting odds, without the influence of bettors' bets, to moderate the actual shown bet odds as the game progresses. The described model that is able to be used as a basis to generate an internal base reference betting odds is our business success outcome.

1.3 Data Mining Goals and Project Plan

Our data mining goal is to create a model that is able to predict the winner of a competitive League of Legends game. The success criteria is to be able to predict a winner with minimum 60% accuracy. With the performance metrics of the model across different stages of the game in progress, we then proceed to create a simple algorithm that is able to use the performance metric to generate a base reference betting odds. The CRISP-DM methodology is used for this project.

2.0 Data Understanding

2.1 Initial Data

For our project, we will need all the in-game match information, such as in-game-champions selected and the gold earned by each team. The data must be of games that are played by professional teams in the same competitive matches that eSports betting operators are interested in providing betting odds on. Due to the granularity of the information required, the dataset must also have the time series information of the matches.

We identified the dataset that fit requirements on the online data science portal, Kaggle (Chuck Ephron-Kaggle, 2018) and proceeded to acquire the League of Legends – Competitive matches dataset which contains the game's competitive matches information between 2015 to 2017.

2.2 Data Description

The eSports League of Legends game competitive matches data from Kaggle contains 7058 observations and 57 columns. The data included key parameters performance of each team over time in addition to pre-start champions selection data. The key outcome variable, the winner team of the match, is also in the same dataset.

2.2.1 Background information about League of Legends key game mechanism

Players are formed into two even teams, blue and red, comprising of 5 members each. The game has 2 stages: Before the game starts “Champions Selection” phrase and the “Main Game” phrase. In the Champions Selection phrase, each team member selects from a pool of over 140 in-game “Champions” that will be their avatar representing them in the game. The teams will also pick up a few champions to “Ban” from usage in the game. As no champions can be selected more than once, there is a huge strategic value in this stage trying to select the champions favourable to their team and depriving the other team of their favourable champions.

In the Main Game stage, each team starts at opposing sides of a map, near what is called a "Nexus." To win a match, a team must destroy the opposing team's Nexus. To do so, each team must work through a series of automatic defensive towers called 'turrets' that are placed along three “lanes” to each base.

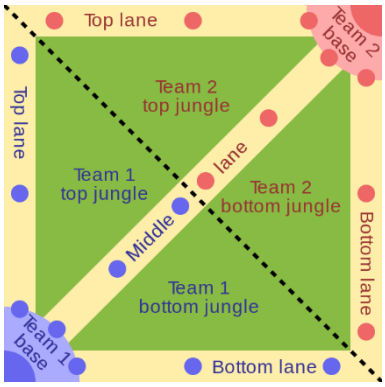


Figure 1: Map Layout of League of Legends Main Game

Along the way, each player gains power by completing secondary game objectives, earning them experience points and gold that are used to increase the player's level and to purchase powerful items, potentially giving players an advantage over their opponents. Examples of these objectives include destroying the opposing team's defensive towers, players, and 'minions' (Non-Player-Characters generated to assist each team), and neutral Non-Player Characters such as Dragons and Barons.

2.2.2 Key Data Columns Description

Col Name	Description
bResult*	Our target variable, match result. “1” for blue team win, “0” for loss
blueTeamTag*	Blue Team’s tag name
blue# blue#Champ*	Name of Blue Team player and champion in different position
blueBans	List of champions Blue Team banned (in order)
goldblue*	Blue Team’s total gold by minute

Col Name	Description
bkills*	List of Blue Team's kills [Time in minutes, Victim, Assist, Location]
bDragon*	List of minutes that Blue Team killed a dragon (Killing a Dragon gives a large gold bonus to all teammates)
bBaron*	List of minutes that Blue Team killed a baron (Killing a Baron give a huge statistic bonus to all team characters)
bTowers*	List of minutes that Blue Team destroyed an enemy defensive tower and location
bInhibs*	List of minutes that Blue Team destroyed an inhibitor and location (Destroying an enemy inhibitor will allow upgraded friendly minions to be created)

Table 1: Key Data Columns Description

**There are corresponding columns for the Red Team*
 # There are 5 standard positions of each team, Top, Mid, ADC, Support, Jungle

2.3 Data Exploration

There are 25 columns which have time-series data providing a per minute value for the metric. Also, the columns: redBans and blueBans have data in a list like format, since more than one champion can be banned by a team. The remaining columns are categorical or continuous.

We get the below insights from the exploration which are leveraged for later analysis:

1. Distribution of number of games with their length in minutes

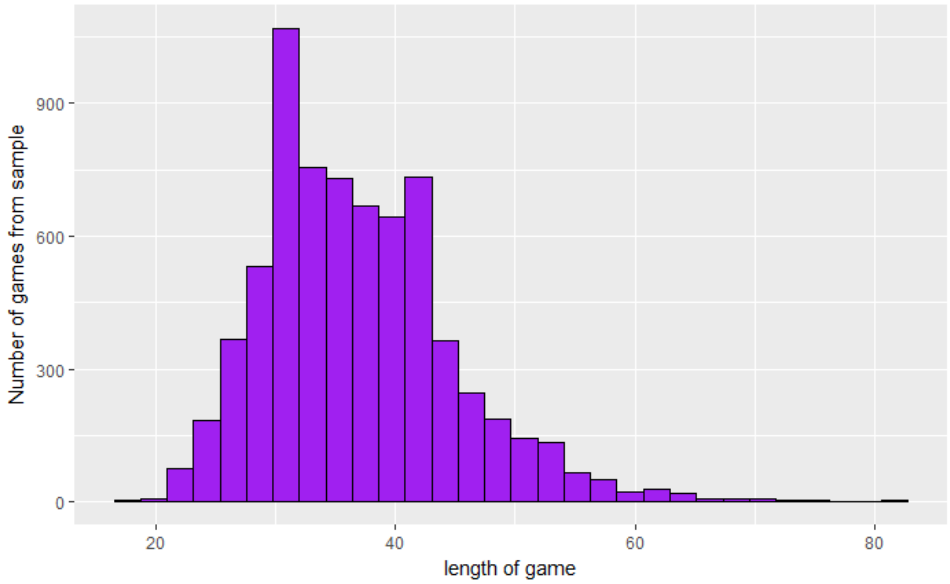


Figure 2: Distribution of number of games with their length in minutes

From the above figure, we see that the number of games are normally distributed by their length. We also observe that most games are longer than 20 minutes. Since these are professional games, it would be difficult to get a result before 20 minutes.

2. Top teams

The below figures show the top teams by the blue and red colours

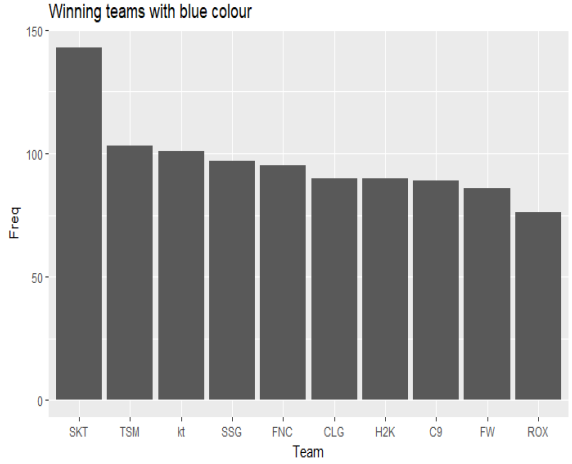


Figure 3: Freq of Teams winning when on blue side

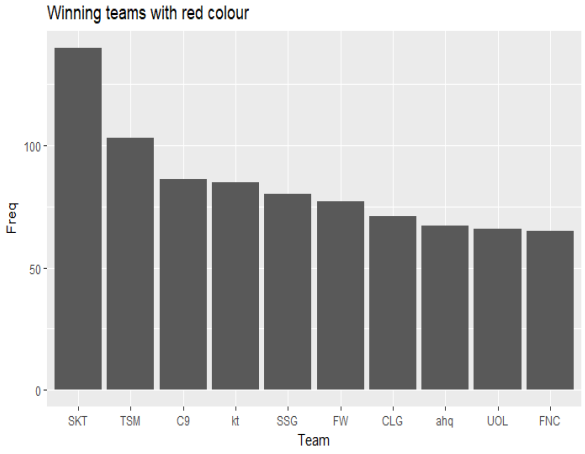


Figure 4: Freq of Teams winning when on red side

We see that SKT and TSM teams are the top 2 teams irrespective of their team colour.

3. Top champions to ban by team colour when they won

The below shows the popular choices of champions that were banned and the colour of the team that banned them when the team won.

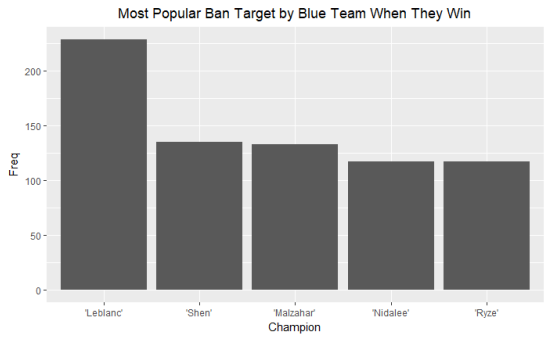


Figure 5: Most popular ban target by blue team as winners

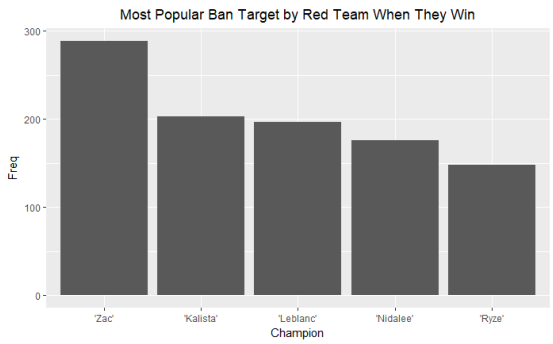


Figure 6: Most popular ban target by red team as winners

4. Champions used by position

The below table shows the top 10 picks for the champions in the different map positions. Since there is a substantial difference between the number of times the first champion is picked compared to the number of times the 10th preferred champion is chosen, it shows that there is a strong preference for certain champions in specific positions.

```
[1] "Location:  blueTopChamp"
champNames
  Maokai     Shen   Rumble     Gnar Nautilus    Poppy  Trundle Renekton    Fiora JarvanIV
     769      628      544      540      387      367      320      296      242      229
[1] "Location:  blueJungleChamp"
champNames
  RekSai    Elise   Gragas   LeeSin   Graves   Khazix  Nidalee   Rengar JarvanIV  Sejuani
    1089     1030      917      613      562      428      403      329      248      228
[1] "Location:  blueMiddleChamp"
champNames
    Viktor    Orianna     Syndra       Azir      Corki    Leblanc Cassiopeia    Taliyah
       609        597        488        424        416        394        373        363
      Ryze       Ahri
       326        303
[1] "Location:  blueADCChamp"
champNames
    Ashe    Sivir   Lucian  Kalista     Jhin    Varus   Ezreal  Caitlyn Tristana    Corki
     731      730      729      666      659      595      589      557      380      338
```

Figure 7: Freq analysis of champions picked

This information will be leveraged to create a column that identifies whether a team gets to use their preferred champions in their preferred positions.

5. Number of matches in the last 4 years

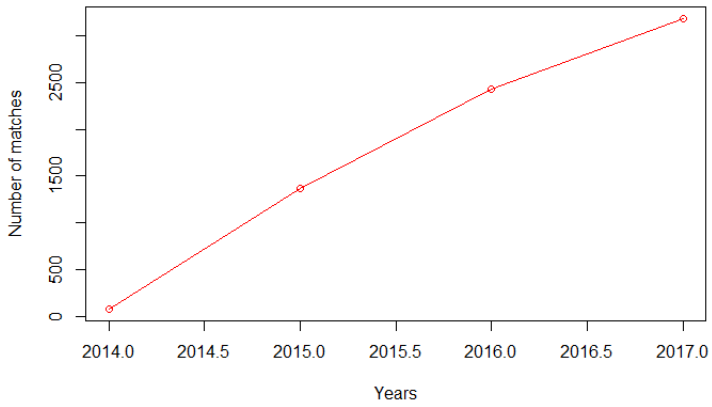


Figure 8: Total Number of Competitive Matches played per year

As seen from the above figure, the number of matches played per year has been increasing over the last 4 years, which can be attributed to the general increase in the popularity of eSports.

2.4 Association analysis

It is useful for an observer to identify patterns based on the “Champions” characters that have been banned by the different teams for each match.

2.4.1 Data processing for association analysis

The raw data for the bans was processed and transformed into “Basket” type of data as shown below using a Python script

Match number	blueBans	redBans
1	['Rumble', 'Kassadin', 'Lissandra']	['Tristana', 'Leblanc', 'Nidalee']
2	['Kassadin', 'Sivir', 'Lissandra']	['RekSai', 'Janna', 'Leblanc']
3	['JarvanIV', 'Lissandra', 'Kassadin']	['Leblanc', 'Zed', 'RekSai']
4	['Annie', 'Lissandra', 'Kassadin']	['RekSai', 'Rumble', 'LeeSin']
5	['Irelia', 'Pantheon', 'Kassadin']	['Rumble', 'Sivir', 'Rengar']

Table 3: Original Data

Id	Banned hero
1	Rumble
1	Kassadin
1	Lissandra
1	Tristana
1	Leblanc
1	Nidalee
2	Kassadin
2	Sivir
2	Lissandra
2	RekSai
2	Janna
2	Leblanc
3	JarvanIV
3	Lissandra
3	Kassadin
3	Leblanc
3	Zed
3	RekSai

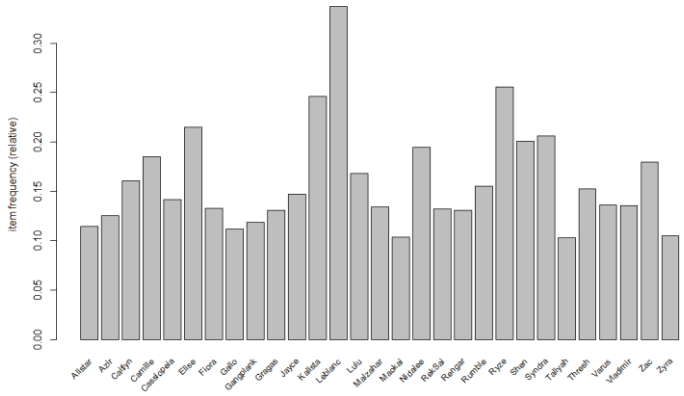


Figure 9: Basket Data (Left) and Frequency plot (right) for the banned “champions” character with a support of at least 0.1

2.4.2 Associations

The below image shows the associations between the characters that have a support of at least 0.1

```
> inspect(multiItems)
    items                support    count
[1] {Elise,Zac}          0.1034287 730
[2] {Camille,Leblanc}    0.1205724 851
[3] {Leblanc,Syndra}     0.1093794 772
```

The figure on the left shows that these characters were likely to be banned together before the match. Most likely these characters because they were inherently too “strong” when used in a certain way in the game.

2.4.3 Apriori Algorithm

The table below shows the sequence of bans that were the result of running the Apriori association analysis algorithm on the basket data in R with a support of at least 0.05

```
> inspect(sort(grules, by = "lift"))
     lhs                      rhs       support    confidence lift     count
[1]  {Camille,Leblanc}     => {Rengar}  0.07438368 0.6169213  4.722593 525
[2]  {Leblanc,Rengar}      => {Camille} 0.07438368 0.7686676  4.150923 525
[3]  {Caitlyn,Elise}       => {Zac}     0.05568150 0.7332090  4.084443 393
[4]  {Galio}               => {Zac}     0.08146784 0.7269279  4.049453 575
[5]  {Leblanc,Malzahar}    => {Camille} 0.05199773 0.7354709  3.971656 367
[6]  {Rengar}              => {Camille} 0.09308586 0.7125813  3.848048 657
[7]  {Camille}             => {Rengar}  0.09308586 0.5026779  3.848048 657
[8]  {Caitlyn}             => {Zac}     0.09889487 0.6166078  3.434900 698
[9]  {Zac}                 => {Caitlyn} 0.09889487 0.5509077  3.434900 698
[10] {Jayce,Leblanc}       => {Camille} 0.05582318 0.6314103  3.409712 394
[11] {Elise,Zac}           => {Caitlyn} 0.05568150 0.5383562  3.356641 393
[12] {Malzahar}            => {Camille} 0.07934259 0.5919662  3.196708 560
[13] {Zyra}                => {Camille} 0.05837348 0.5552561  2.998468 412
[14] {Kennen}              => {Zac}     0.05114763 0.5348148  2.979260 361
[15] {Zac}                 => {Elise}   0.10342873 0.5761642  2.677134 730
[16] {Caitlyn,Zac}         => {Elise}   0.05568150 0.5630372  2.616140 393
[17] {Camille,Rengar}      => {Leblanc} 0.07438368 0.7990868  2.371722 525
[18] {Camille,Jayce}       => {Leblanc} 0.05582318 0.7864271  2.334148 394
[19] {Rengar}              => {Leblanc} 0.09676962 0.7407809  2.198668 683
[20] {Camille,Malzahar}    => {Leblanc} 0.05199773 0.6553571  1.945126 367
[21] {Camille}             => {Leblanc} 0.12057240 0.6511094  1.932519 851
[22] {Zyra}                => {Leblanc} 0.06474922 0.6159030  1.828025 457
[23] {Jayce}               => {Leblanc} 0.08841031 0.6017358  1.785976 624
[24] {Varus}               => {Leblanc} 0.07693398 0.5638629  1.673568 543
[25] {Syndra}              => {Leblanc} 0.10937943 0.5313145  1.576963 772
[26] {Malzahar}            => {Leblanc} 0.07069991 0.5274841  1.565594 499
```

Table 4: Apriori association analysis output on the basket data in R with a support of at least 0.05

2.4.4 Association Findings

As can be seen from the above association analysis, due to a lift > 1, it has been a consistent strategy across matches to ban the character Renger whenever the characters Camille and Leblanc have been banned. This can be explained by the fact that these characters are among the strongest in the game. These association findings may allow for new betting offerings for bettors to bet on what is the next champion ban.

2.5 Data Quality

The snapshot below shows the summary of the raw data along with the number of missing values.

	Levels	Storage	NAs
League	15	integer	0
Season	2	integer	0
Type	4	integer	0
blueTeamTag	185	integer	30
bResult		integer	0
rResult		integer	0
redTeamTag	184	integer	29
gamelength		integer	0
bKills	7021	integer	0
bTowers	6823	integer	0

Since the data was collected from in-game statistics, there are not many missing data or data with inconsistent data-types for a specific variable.

Figure 10: Summary of missing values

For the later analysis rows with missing values were ignored as they constitute only 0.5 % of the records. It was seen that four matches were less than 20 minutes, which were too short for any valuable prediction analysis in the 20th minute. These matches data rows were ignored for the prediction analysis for the 20th minute.

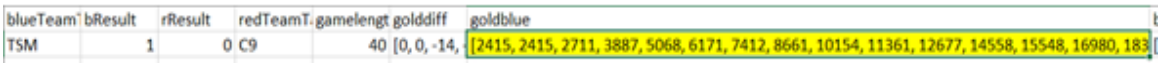
League	Year	Season	Type	blueTeamTag	bResult	rResult	redTeamTag	gamelength
NALCS	2016	Spring	Season	IMT	1	0	TIP	19
TCL	2017	Summer	Season	FB	1	0	CLK	17
OPL	2016	Summer	Season	HLN	0	1	AE	18
OPL	2017	Spring	Season	CHF	1	0	TM	18

Figure 11: Data of 4 matches that has game length less than 20 mins

3.0 Data Preparation

3.1 Data Selection and Construction

The raw data file "matches.csv" has **57 columns** and **7058 rows**. The time series data for some of the columns is shown below.



blueTeam	bResult	rResult	redTeamT	gamelengt	golddiff	goldblue
TSM	1	0	C9	40	[0, 0, -14,	[2415, 2415, 2711, 3887, 5068, 6171, 7412, 8661, 10154, 11361, 12677, 14558, 15548, 16980, 183

Figure 12: Data with time series information in columns

As can be seen from above fig., the goldblue column has the cumulative gold per minute for the match, but as a part of data preparation, only relevant values from this time series data has been considered. The column names mentioned below are the replacements for the placeholders from which 89 new columns have been derived as described in the table below:

- {**color**} is one from b, r
- {**game_element**} is one from Towers, Inhibs, Dragons, Barons, Kills
- {**time_interval**} is one from 5Mins, 10Mins, 20Mins, QuarterGame (0.25 * game length), HalfGame (0.5 * game length)
- {**team**} is one from blue, red

ID	Derived Column pattern	Example column name	# new columns with pattern	Description of the newly created column
1	**{color}{game_element}** Num	bTowersNum	8	Number of **{game_elements}** destroyed/killed by the **{color}** team in the game
2	**{color}{game_element}** FirstDestroyedTime	rTowersFirstDestroyedTime	10	Time when the **{game_element}** was first destroyed by the **{color}** team. Note: A value of 10000 means that the **{color}** team did not destroy/kill the **{game_element}** item during the duration of the game
3	**{game_element}** FirstKillBy	InhibsFirstKillBy	5	The team who made the first kill for the **{game_element}** in the game. A value of 1 means the blue team made the first kill/ destroy and 0 means the red team did so.
4	**{color}{game_element} {time_interval}**	rTowers20Mins, rInhibs20Mins	50	The number of **{game_elements}** that were destroyed/killed by the **{color}** team by the end of the **{time_interval}** time
5	gold**{team}{time_interval}**	goldblue20Mins, goldred5Mins	10	Gold earned by **{team}** per {time_interval}
6	**{color}**WinEfficiency	rWinEfficiency	2	How efficient is the team playing as **{color}** at winning matches? (Matches Won / Matches Played)
7	Is**{color}**PreferredLineup	isRedPreferredLineup	2	Is the current config of players in team **{color}**'s preferred winning formation? Formation refers to the position of each player in each specific position provided by the data.
8	Is**{color}**PreferredChamps	isRedPreferredChamps	2	Is the current config of champions in team **{color}**'s preferred winning champions formation. Formation refers to the position of each champion in each specific position provided by the data.

Table 5: Columns derived from the original columns

After the new columns were created as mentioned in the above table, a match record would look like the below:

rWinEffici	bWinEffic	Year	bTowersN	bInhibsNu	bDragonsI	bBaronsN	rTowersN	rInhibsNu	rDragonsN	rBaronsNu	bTowersF	rTowersFi	TowersFir	bTowers5	rTowers5I	bTowers1	rTowers1(	bTowers2	rTowers2(
0.3125	0.658824	2016	7	2	3	1	10	3	2	1	13.188	15.673	1	0	0	0	0	3	1
0.3125	0.387755	2016	3	0	2	0	10	2	3	2	19.616	14.983	0	0	0	0	0	1	2

Table 6: Data with time series information removed

3.2 Data Cleaning

Data cleaning included the following steps:

- Time series columns are removed from the processed file as the values from them were transformed to new columns after aggregating interval values.
- rResult column is removed as it is redundant. The column bResult alone, gives us the outcome of the game. If blue loses, it means that red wins. There is no draw in this game.

4.0 Modeling

For the modeling the predictive task, we consider Support Vector Machines, Decision Trees, Random Forests, Neural Networks, Logistic Regression and Extreme Gradient Boosting techniques. We have excluded Naïve Bayes and other clustering techniques.

4.1 Partitioning data for training and evaluation

We partitioned the dataset into three parts with **70% training data, 15% validation data and 15% test data, using a seed value of 42**. For all purposes of assessing the model during training, the validation data was used. To further validate our models, we also used **10-fold cross validation** using **85%** of the data which is the combination of 70% training data and 15% validation data. The test data was used at the end for doing a final unbiased evaluation of the model.

4.2 Hyper parameters tuning and feature selection

The tuning of parameters and selection of features is an iterative process that will be described here in detail.

4.2.1 Hyper parameters selection & tuning

For each of the modeling techniques considered, we used grid-search to tune the hyper-parameters. The below sections discuss the hyper-parameters identified for the different modeling techniques and the process used to get the optimal values for the hyper parameters. Note that there are four scenarios (pre-game, 5 min, 10 min and 20 min into game) considered, and the optimal hyper-parameters are different for each of them due to the different predictors that they use. Here we illustrate the pre-game model hyper-parameter tuning process.

Support Vector Machine

For Support Vector Machine, the following parameters were identified.

- C : regularisation term in the Lagrange formulation
- sigma: inverse kernel width for the Radial Basis kernel function

Running the grid search in R with 10-fold cross validation, we got the following values:

C (0.01 – 0.5)	sigma (0.005 – 0.5)	Mean Accuracy
0.01	0.005	0.543
0.05	0.005	0.67

...	...	...
0.5	**0.005**	**0.696**

Table 7: -SVM-Grid Search Parameters and resultant mean accuracy (not all values are shown for brevity)

As the accuracy was the highest, C = 0.5 and sigma = 0.005 were used for this particular model. This gave us 0.6% better accuracy for the SVM model compared to the default values.

Decision Tree

For the model, we used both Grid Search to avoid overfitting and afterwards pruning to restrict the growth of the tree. We focused mostly on the cross-validated error, but also used the bin size, min split and max depth to prevent the growth. The pruning afterwards showed that the gold collected was a very important factor; if the cross-validated error was set to a maximum of 0.02, only the gold earned remained as the distinctive factor.

Random Forest

For random forest, there were two hyper-parameters identified:

- ntree: Number of trees to grow.
- mtry: Number of variables randomly sampled as candidates at each split.

Running the grid search in R with 10-fold cross validation, we got the following values (not all values are shown for brevity):

ntree (100, 200, 500, 1000, 2000)	mtry (2, 3, 4)	Mean Accuracy
100	2	0.69
500	3	0.678
...	...	...
250	**2**	**0.691**

Table 8: RF – Grid search parameters and resultant mean accuracy

As the accuracy was the highest, ntree = 250 and mtry = 2 were used for this particular model. An extremely small accuracy boost of 0.07% was observed compared to the default values.

Neural Network

For neural network, there was one hyper-parameter identified:

- size: Number of hidden layer nodes.

Running the grid search in R with 10-fold cross-validation, we got the following values (not all values are shown for brevity):

size	Mean Accuracy
2	**0.6957232**
5	0.6932924
6	0.6924905
8	0.6922791
10	0.6916759
20	0.6922873

Table 9: NN-Grid-search parameters and resultant Accuracy (decay is set to 0.01)

As the accuracy was the highest for size = 2, it was used for this particular model. An extremely small accuracy boost of 0.4% was observed compared to the default values.

4.2.2 Feature selection

It is very unlikely that some variables provide additional power to our different scenarios because they typically only play a role after 20 minutes, which is why it is necessary to do feature selection. After running logistic regression, we found that some features (variables) were not contributing significantly to the prediction of a win/loss. From the below table we can see that the blue towers, blue inhibitors and blue barons are not significantly related to the outcome, even by the twentieth minute (the 20-minute scenario). These results hold for the scenarios where we predict the outcome 5 and 10 minutes into the game.

As for the statistically significant variables, rWinefficiency, bWinefficiency and the variables related to the teams' preferred champs have the highest z-value, suggesting that these are very important for predicting the win/lose. The negative coefficient for rWinefficiency suggests that if the red team has proved to get bad results in the previous games, this will positively influence the win odds of blue.

After removing the non-significant variables, we see that the remaining variables are all significant as shown in the new Pr(>|z|) column. Also, the accuracy of the model was not impacted upon removing the non-significant variables.

20 minutes variables	old z val.	old Pr(>|z|)	new z val.	new Pr(>|z|)
rWinEfficiency	-14.997	< 2e-16 ***	-14.582	< 2e-16 ***
bWinEfficiency	13.899	< 2e-16 ***	14.043	< 2e-16 ***
bTowers20Mins	1.489	0.13646		
rTowers20Mins	-3.08	0.00207 **	-2.279	0.0227 *
bInhibs20Mins	-0.303	0.76201		
rInhibs20Mins	0.072	0.94287		
bDragons20Mins	4.803	1.57e-06 ***	4.705	2.54e-06 ***
rDragons20Mins	-4.103	4.08e-05 ***	-4.34	1.43e-05 ***
bBarons20Mins	NA	NA		
rBarons20Mins	NA	NA		
bKills20Mins	7.243	4.40e-13 ***	7.575	3.60e-14 ***
rKills20Mins	-7.384	1.53e-13 ***	-7.446	9.59e-14 ***
goldblue20Mins	7.244	4.36e-13 ***	11.201	< 2e-16 ***
goldred20Mins	-7.397	1.39e-13 ***	-9.796	< 2e-16 ***
isBluePreferredLineup	7.686	1.52e-14 ***	8.618	< 2e-16 ***
isRedPreferredLineup	-8.945	< 2e-16 ***	-8.714	< 2e-16 ***
isBluePreferredChamps	4.635	3.56e-06 ***	4.596	4.30e-06 ***
isRedPreferredChamps	-6.163	7.15e-10 ***	-5.78	7.49e-09 ***

Table 10: Significance of variables with Significance. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

4.3 Model Building

It would be of interest for a betting house to predict the winner of a League of Legends match at various stages into the game. Based on the exploratory analysis which reveals that most games carry on for more than 20 minutes and some understanding of the game itself, models have been built for 4 different scenarios into a game: before the game begins, 5 mins into the game, 10 mins into the game and finally 20 mins into the game, which is just about into the final phrase of the game for most of the professional matches based on our domain understanding.

4.3.1 Scenario 1: Before the Game Starts

Before the game starts, most of the in-game metrics like amount of gold collected, number of game elements killed are not known. The only variables known to us at this point are the efficiency of the red and blue teams based on their overall historical performance (e.g. rWinEfficiency), whether the red and blue teams have their preferred lineups (e.g. isBluePreferredLineup) and whether the teams have their preferred champions in their preferred positions (isRedPreferredChamps).

We created the models using several techniques in R, and the following are the results on the validation set.

Model	Initial parameters for 70 (train) and 15 (validation) data	Overall Error	*Grid-search optimized parameters for 10-fold cross validation*	*Error for 10-Fold cross-validation*
Decision Tree	Min. Split = 20, Min. Bucket = 7, Max Depth = 5, Complexity = 0.0100	35%	*Min. Split = 20, Min. Bucket = 7, Max Depth = 5, Complexity = 0.0100*	*37.58%*
Extreme Boost	Max Depth = 6, Learning Rate = 0.3, Iterations = 50	31.4%	*Max Depth = 5, Learning Rate = 0.2, Iterations = 25*	*32.29%*
Random Forest	Trees = 500, Variables = 2	31.7%	*Trees = 1000, Variables = 2*	*30.59%*
Support Vector Machine	C = 0.5, sigma = 0.5, kernel = Gaussian Radial Basis	29.2%	*C = 0.5, sigma = 0.01, kernel = Gaussian Radial Basis*	*30.69%*
Logistic Regression		28.7%		*30.41%*
Neural Nets	Hidden layer nodes = 10	30%	*Hidden layer nodes = 5*	*30.22%*

Table 11: Errors for different models before match starts

We see that **Support Vector Machine** and **Logistic Regression** have the two lowest overall error. We select **Logistic Regression** as the modeling technique for before the game starts as it produces the lowest overall error and is fast to train and validate compared to other models.

4.3.2 Scenario 2: 5 minutes into the game

The variables selected for predicting the winner five mins into the game included the ones that were available before the game and the ones that were available for 5 minutes into the game, e.g. bTowers5Mins, rDragons5Mins. Any variable that gave information on the game after 5 minutes were discarded for this modeling.

We created the models using several techniques in R, and the following are the results on the validation set:

Model	Initial parameters for 70 (train) and 15 (validation) data	Overall Error	*Grid-search optimized parameters for 10-fold cross validation*	*Error for 10-Fold cross-validation*
Decision Tree	Min. Split = 20, Min. Bucket = 7, Max Depth = 5, Complexity = 0.0100	33.80%	*Min. Split = 20, Min. Bucket = 7, Max Depth = 5, Complexity = 0.0100*	*34.66%*
Extreme Boost	Max Depth = 6, Learning Rate = 0.3, Iterations = 50	32.20%	*Max Depth = 5, Learning Rate = 0.2, Iterations = 25*	*30.31%*
Random Forest	Trees =2000, Variables = 4	31.4%	*Trees = 250, Variables = 2*	*30.22%*
Support Vector Machine	C = 1, sigma = 0.4909, kernel = Gaussian Radial Basis	40.7%	*C = 0.5, sigma = 0.005, kernel = Gaussian Radial Basis*	*37.96%*
Logistic Regression		28.60%		*28.52%*

Neural Nets	Hidden layer nodes = 10	29.2%	*Hidden layer nodes = 8*	*29.84%*

Table 12: Errors for different models after 5 mins in the game

From the above table, we can see **logistic regression** is still the most accurate with a cross-validation error rate of **28.52%.**

4.3.3 Scenario 3: 10 minutes into the game

The variables selected for predicting the winner 10 mins into the game included the ones that were available before the game and the ones that were available for 10 minutes into the game, e.g. bTowers10Mins, rDragons10Mins. Any variable that gave information on the game after 10 minutes or for 5 minutes were discarded for this modeling.

We created the models using several techniques in R, and the following are the results on the validation set:

Model	Initial parameters for 70 (train) and 15 (validation) data	Overall Error	*Grid-search optimized parameters for 10-fold cross-validation*	*Error for 10-Fold cross-validation*
Decision Tree	Min. Split = 20, Min. Bucket = 7, Max Depth = 5, Complexity = 0.0100	32.60%	*Min. Split = 20, Min. Bucket = 7, Max Depth = 5, Complexity = 0.0100*	*37.58%*
Extreme Boost	Max Depth = 6, Learning Rate = 0.3, Iterations = 50	27.50%	*Max Depth = 4, Learning Rate = 0.3, Iterations = 25*	*28.61%*
Random Forest	Trees =500, Variables = 4	28.60%	*Trees = 250, Variables = 2*	*27.38%*
Support Vector Machine	C = 1, sigma = 0.4909, kernel = Gaussian Radial Basis	27.60%	*C = 0.5, sigma = 0.01, kernel = Gaussian Radial Basis*	*25.87%*
Logistic Regression		25.60%		*25.12%*
Neural Nets	Hidden layer nodes = 10	27.60%	*Hidden layer nodes = 8*	*26.35%*

Table 13: Errors for different models after 10 mins in the game

We again see that **Logistic Regression** has the least overall error.

4.3.4 Scenario 4: 20 minutes into the game

Like the earlier scenarios, the variables selected for predicting the winner 20 mins into the game included the ones that were available before the game and the ones that were available for 20 minutes into the game, e.g. bTowers20Mins, rDragons20Mins. Any variable that gave information on the game after 20 minutes or earlier intervals were discarded for this modeling.

Model	Initial parameters for 70 (train) and 15 (validation) data	Overall **Error**	*Grid-search optimized parameters for 10-fold cross validation*	*Error for 10-Fold cross-validation*
Decision Tree	Min. Split = 20, Min. Bucket = 7, Max Depth = 5, Complexity = 0.0100	26.9%	Min. Split = 20, Min. Bucket = 7, Max Depth = 5, Complexity = 0.0050	23.32%
Extreme Boost	Max Depth = 6, Learning Rate = 0.3, Iterations = 50	23.7%	Max Depth = 4, Learning Rate = 0.2, Iterations = 25	18.60%
Random Forest	Trees =500, Variables = 4	23%	Trees = 250, Variables = 2	21.06%

Support Vector Machine	C = 1, sigma = 0.4909, kernel = Gaussian Radial Basis	25.1%	C = 0.5, sigma = 0.005, kernel = Gaussian Radial Basis	19.65%
Logistic Regression		21%		16.81%
Neural Nets	Hidden layer nodes = 10	24.9%	Hidden layer nodes = 10	17.28%

Table 14: Errors for different models after 20 mins in the game

4.5 Model Assessment

Based on the analysis from the above scenarios, we have found that **Logistic regression** consistently gives the least overall error for the validation data. This will be the preferred modeling technique that will be considered for the evaluation phase.

5.0 Evaluation

5.1 Model Evaluation

The Logistic regression model that was found to perform the best on the validation data is chosen for evaluation on the test partition (**15 %**) of the data set.

Confusion Matrix

For all cases below, we adopted the convention that '1' means bResult was 1, meaning that Blue won the match. If bResult is 0, it means that Red won the match.

Scenario 1: Before game starts		**Predicted**	
		0	1
Actual	0	312 (TN)	171 (FP)
	1	133 (FN)	444 (TP)

Scenario 2: 5 mins into game		**Predicted**	
		0	1
Actual	0	312 (TN)	166 (FP)
	1	137 (FN)	449 (TP)

Scenario 3: 10 mins into game		**Predicted**	
		0	1
Actual	0	327 (TN)	147 (FP)
	1	117 (FN)	469 (TP)

Scenario 4 : 20 mins into game		**Predicted**	
		0	1
Actual	0	383 (TN)	91 (FP)
	1	88 (FN)	497 (TP)

Table 15: Confusion matrix for Logistic regression for 4 different scenarios

	Accuracy	**Precision**	**Recall (TPR)**	**False Positive Rate**	**F1 Score**
Before Game Starts	**0.713**	**0.722**	**0.769**	**0.354**	**0.745**
5 Mins into Game	**0.715**	**0.730**	**0.766**	**0.347**	**0.747**
10 Mins into Game	**0.750**	**0.761**	**0.800**	**0.310**	**0.780**
20 Mins into Game	**0.830**	**0.845**	**0.890**	**0.192**	**0.847**

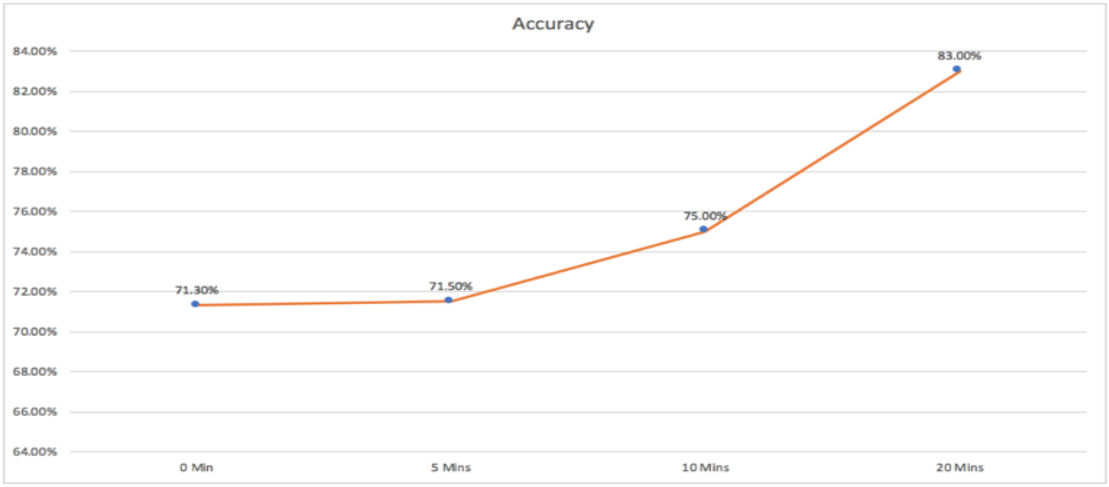


Figure 13: Comparison of model evaluation metrics for different scenarios and accuracy curve for different scenarios

Accuracy Analysis

Before the game starts, the prediction of whether blue or red wins the match is correct **71.3%** of the time. This is quite good considering that we only feed the model the predictors that happen before the game. 5 minutes into the game, the accuracy does not differ much as nothing significant has happened yet. 10 minutes into the game, the "live" predictors like amount of gold red team obtains vs amount of gold blue team obtains start to influence the match outcome. The accuracy of prediction goes up to **75%**. 20 minutes into the game, the model is fed with predictors that have a direct impact on the outcome, and the accuracy shoots up to **83%**.

Precision/Recall Analysis

Before the game starts, the precision value is **72.2%.** This means that when the model predicts blue will win, it is correct **72.2%** of the time. The recall value states that the model detects correctly **76.9%** of the cases where blue wins. At 5 minutes, the precision and recall values are like that before the game, as nothing significant has happened. 10 minutes into the game, the precision becomes **76.1%**, and the recall value becomes **80%**. At 20 minutes, we see the precision and recall increase significantly from before. Now when the model predicts blue will win, it is correct **84.5%** of the time. The recall value states that the model detects correctly **89%** of the cases where blue wins.

False Positive Rate Analysis

False positive rate measures the misclassifications of blue losses as wins. Before the game starts, this happens 35.4% of the time. The FPR at the first 5 minutes mark is 34.7%, comparable to that before the game starts. 10 minutes into the game, with more predictors to work on, the model's FPR goes down to 31%. This is in line with the model becoming more accurate. At 20 minutes into the game, the false positive rate goes down to 19.2%. This is a significant reduction of misclassifying blue losses as blue wins.

F1 Score Analysis

As we are predicting the outcome of the match, and not solely focused on whether one side wins or not, a high F1 score is desirable. This means the model has high precision and high recall, and a good trade-off is seen. 5 minutes into the game, not much has changed. 10 minutes into the game, F1 score has increased in tandem with the model becoming more accurate. As precision and recall are still relatively similar, the F1 score is still high. At the 20-minute mark, the F1 score is now 0.847, a significant jump from the previous value of 0.780 at the 10-minute mark. This high value means that precision and recall are still relatively similar, while the model is getting more accurate.

ROC Curve Analysis

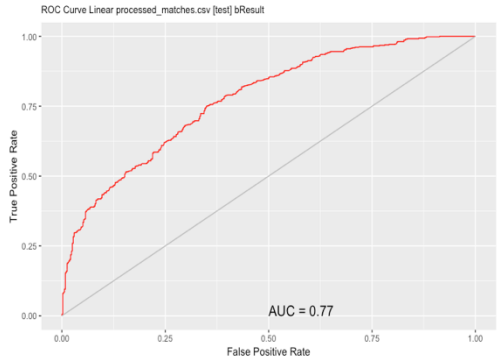


Figure 14 : ROC Curve for Before Game Starts

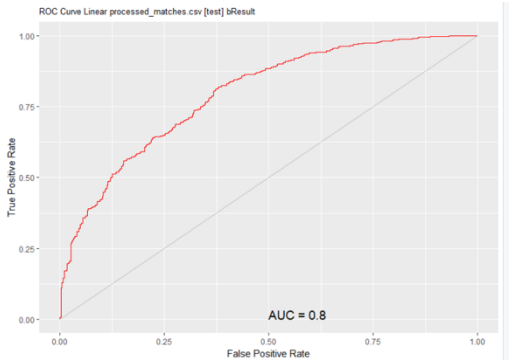


Figure 15: ROC Curve 5 Min Into Game

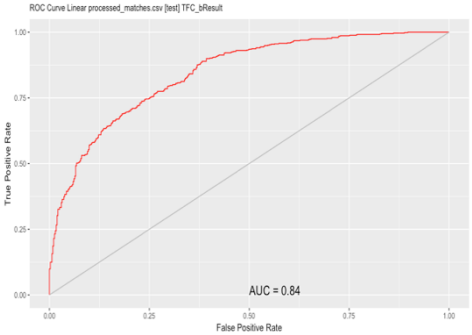


Figure 16: ROC Curve for 10 Min Into Game

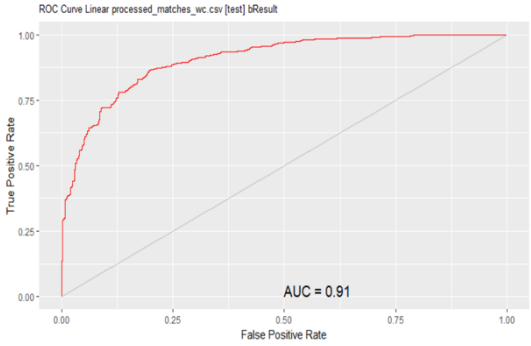


Figure 17: ROC Curve for 20 Min Into Game

A true random model would have the ROC curve as that of the grey line that passes through the diagonal of the charts in the ROC graph. The red line shows the ROC curve of the logistic regression results. As shown, it tends towards the top left of the graph. This shows that the logistic regression can get a high true positive rate with a relatively lower false positive rate. In other words, it means that the model can have a high percentage of detecting whether blue will win while having relatively minimal misclassifications of blue losses as wins. At 5 minutes, the AUC of the ROC curve is slightly more than that for before the game starts, at **0.8**, showing that we can get a higher detection of whether blue will win vs misclassification of blue losses as wins. The AUC of the ROC curve for 10 min into the game now increases to 0.84. It shows basically an increment in accuracy for the model as the model now considers predictors specific to the 10-min mark. The AUC is now 0.91, a big jump from the previous value. We can see now that the curve tends towards the top left, showing signs of an accurate model.

Sensitivity / Specificity Chart

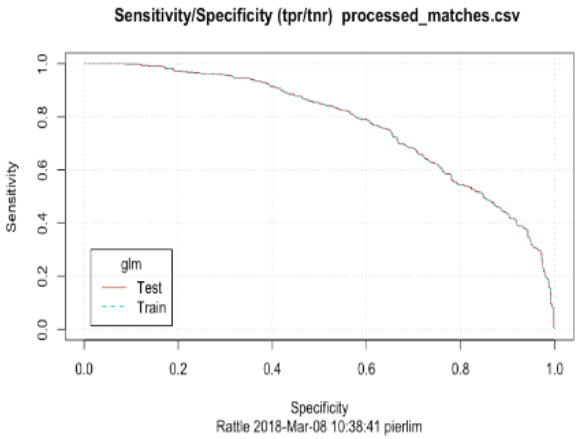


Figure 18: Before Game Starts

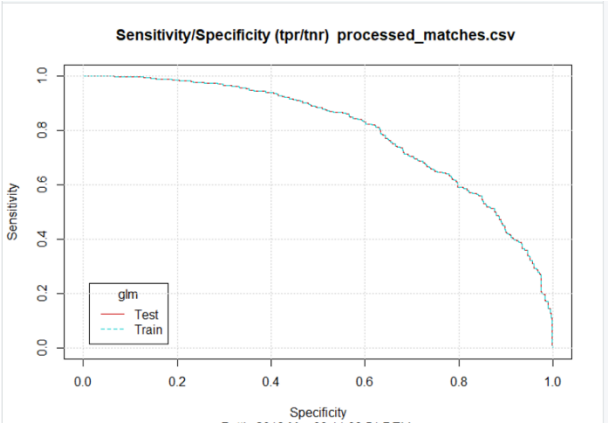


Figure 19: 5 Mins into Game

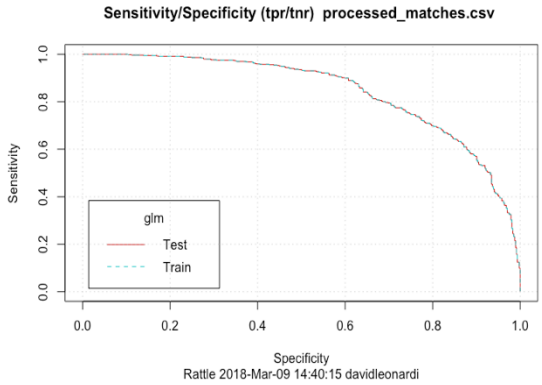


Figure 20: 10 Mins into Game

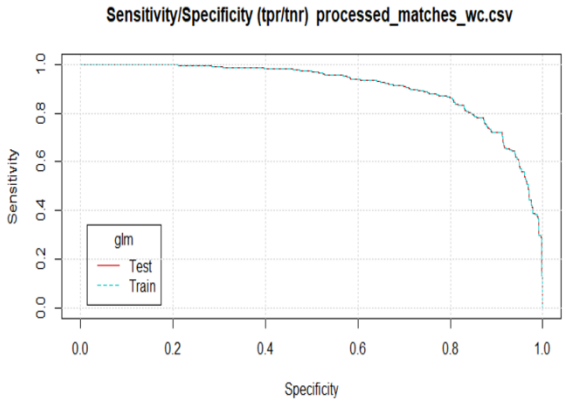


Figure 21: 20 Mins into Game

Before the game starts, we can see when the sensitivity is high, the specificity is also high. A perfect model will have the shoulder of the curve at the top right of the chart i.e. It predicts all actual wins as wins and all actual losses as losses. For the 5 minutes graph, we can see that there is not much improvement from the graph before the game starts. This is likely because in the first 5 minutes of the game, nothing significant has happened that might tilt the game in either side's favour. At the 10-minute mark, we are starting to see that the shoulder of the graph is tending towards the right. This shows that the accuracy is increasing. At 20 minutes, the sensitivity is now high, along with the specificity. This means that the model has a high true positive rate coupled with a high true negative rate – it is predicting matches very accurately as positive or negative.

5.2 Evaluation Findings

In the pre-game situation, considering that we have minimal information about the match except the win history of the team, whether they have their preferred player line-up and whether they got to select their favourite champions in the game, the accuracy of **71.3%** on the test data before the start of the game is pretty good. One may also conclude from this that the skills of the players, and the "set-plays" that they developed in the team because of playing together for extended periods of time contribute favourably to whether they win or lose. From an e-sports perspective, **71.3%** is considered good odds for the operators to establish their wagers on.

In the first 5 minutes, we see that nothing much has changed. The model gets slightly better at predicting wins/losses. This can be explained by the fact that in the first 5 minutes, strategies to tilt the game in either side's favour have not been fully executed i.e. getting as much gold as possible.

In the first 10 minutes, we start to see the accuracy of the model getting better. This can be explained by the fact that blue and red have started to unfold their strategies. For example, blue may be going for getting as much gold as possible, and has successfully gathered much more gold than red. From our model evaluation, we found that the gold acquisition is an important factor for winning, so this tilts the match outcome in their favour.

At the 20 minutes mark, the accuracy of the model is already quite good. Most of the strategies have been deployed and it is clear which side is starting to obtain an upper hand. At this point, we can say with 83% accuracy which side will win the match. From the e-sports angle, the pay-out on the predicted winner at this stage for the customer should be tuned much lower.

Also, we evaluated the year on year accuracy of the models to account for any changes that might be made to the game due to updates to the game. However, as can be seen below, an increasing trend in the accuracy of prediction for the different scenarios can be consistently observed. The year wise increase in overall accuracy can be attributed to greater availability of training data due to the increasing number of matches played as was seen during the data exploration.

Scenario	Year	Overall error	Overall Accuracy
After 5 mins	2014	33.40%	66.60%
After 10 mins	2014	30%	70%
After 20 mins	2014	20%	80%
After 5 mins	2015	27.70%	72.30%
After 10 mins	2015	23.30%	76.70%
After 20 mins	2015	19.90%	80.10%
After 5 mins	2016	30.40%	69.60%
After 10 mins	2016	24.40%	75.60%
After 20 mins	2016	20%	80%
After 5 mins	2017	27%	73%
After 10 mins	2017	26.80%	73.20%
After 20 mins	2017	18.40%	81.60%

Table 16: Comparisons of the accuracy of Logistic Regression between 2014 and 2017

6.0 Conclusion

We applied the CRISP-DM methodology to gain insights into a popular eSports "League of Legends" and derive business value from the point of view of a professional betting house.

We started by doing an exploration of the initial data set looking for general trends and popular strategies employed by the top teams of the game. We also performed association analysis to understand patterns employed by teams when they ban certain champions from the match.

Since most of the in-game statistics were captured as time series data providing minute by minute values, we aggregated the values from these columns based on domain understanding and business goals and generated new columns.

We concentrated on 4 scenarios: before the start of the game, after 5 minutes into the game, after 10 minutes into the game and after 20 minutes into the game to generate models for predicting the outcome using the derived columns. We validated our models using 10-fold cross-validation to iron out inconsistencies in the error rate due to the choice of the validation data.

The models generated using **Logistic Regression** were found to be the most accurate in terms of overall accuracy on the validation data and were used during the evaluation phase. It was observed that the accuracy of the model improved as more details of the game was available. An accuracy of **83%** was obtained using Logistic regression on the test data set when used to predict the winner of the game 20 minutes into the match. Important events or features are also

6.1 Plan Deployment/Monitoring

After achieving our data mining goals of verifying that prediction can be done for the game and developing a robust prediction model to predict the winner of the game, we proceed to complete our business objective to setup the prediction system for our company. The plan is to use the final comprehensive model developed which consider all the key features used for prediction of the constructed dataset to predict the match winner. The overall error rate can be used as a probability value to generate the base reference odds.

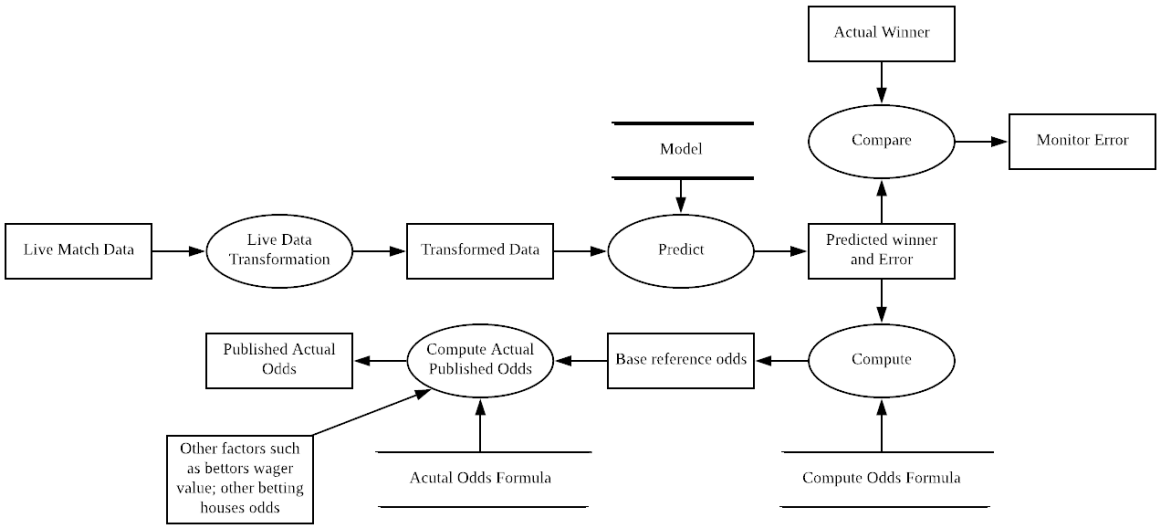


Figure 22: System Deployment and Monitoring

The figure above gives the high-level system diagram or inference diagram of the deployed system. After our model predicted the winner, the result is also compared with the actual winner of the game later as our monitoring subsystem.

6.2 Process Review

We tried various modeling techniques on validation data. Although, for some instances we were getting better accuracy by using **SVM** after using tuned hyper parameters, we found that it is **inefficient** in terms of time and resources consumption for training. So, we decided to go with **Logistic Regression** as it achieved a better **balance** between **accuracy** and **time required for training.**

6.3 Next Steps

Currently, the models are generated only for pre-game, 5, 10 and 20 minutes into the game. However, to make the system more accurate and feasible as a live production system, minute-by-minute modeling needs to be introduced. This is because the game is continuously “evolving” with new tactics developed

by players, new patches made to the game. Therefore the system needs to be able to do online learning, learning incrementally from a mini-batch of instances and combine this new knowledge with knowledge gained in past training instances. To be able to do this, speed of training and the size of the data becomes an important factor.

This can also be done with incremental learning algorithms where new data from live game matches are used to continuously extend the existing knowledge, training it to be more accurate for the live production system. With more accurate models trained, the improved models will be useful for the legal betting operator to provide live minute by minute betting odds. With this kind of requirements, the time spent on training becomes an important factor. In our tests, the model we chose, logistic regression, was one of the quickest on our dataset.

Also, as the match data size becomes an issue eventually as we will be unable to fit all training data into memory. One way to get around this is to use out-of-core / incremental learning algorithms, where it is possible to train the system with only the new training data in memory, and appending this new knowledge to existing knowledge after training. In our case of logistic regression, we may be able to leverage upon stochastic gradient descent or mini-batch gradient descent to accomplish this. Parallel computing can also be explored for the model algorithms to further accelerate the training time.

Appendix

A. List of references

Chuck Ephron-Kaggle. (2018). *League of Legends - Competitive matches, 2015 to 2017*. Retrieved from https://www.kaggle.com/chuckephron/leagueoflegends/version/6

Dallas Observer. (16 Nov, 2017). *You Think It's Just a Game? 15 of eSports' Most Popular Video Games.* Retrieved from http://www.dallasobserver.com/arts/15-of-esports-most-popular-video-games-10075907

Fandom: Powered By Wikia. (n.d.). League of Legnds Wiki. Retrieved from Fandom Site dedicated to LoL: http://leagueoflegends.wikia.com/wiki/League_of_Legends_Wiki

Pinnacle. (2017). *The road to five million eSports bets*. Retrieved from https://www.pinnacle.com/en/esports/betting-articles/home/esports-betting-growth-at-pinnacle/ay22gtmplb93agpa

Riot Games - League Of Legends. (n.d.). Welcome to League of Legends. Retrieved from https://na.leagueoflegends.com/en/featured/new-player-guide#/?_k=w1e4rs

Statista. (2018). *Estimated total amount wagered on eSports on cash gambling sites worldwide in 2016, by product type*. Retrieved from https://www.statista.com/statistics/618896/esports-gambling-market-worldwide/

SuperData: Games & Interactive Media Intelligence. (2018). *Esports Market Report: Courtside—Playmakers of 2017.* Retrieved from https://www.superdataresearch.com/market-data/esports-market-report/