Project Report : Predicting Win Probability in DoTA2 Using Logistic Regression

Paraskumar, Karthigeyan, Shanmuga Priya, Raju, Niranjan

03/08/2019

Introduction

The World Economic Forum projects eSports (Electronic Sports) industry to reach $1.4 billion by 2020. The growing popularity of eSports has also cought the attention of social media sites like YouTube and Twitch, which has recetly started live gaming streams to capture this market. One of the major eSports tounament covered is annual DoTA2 tournament "The International" with a total prize money worth around $20 million in 2016. With this context we plan to build a logistic regression model to predict the probability of winnig a DoTA2 game given the hero selection by the teams.

DoTA2 is a popular MOBA (Multiplayer online battle arena) game. The gameplay consists of two teams with 5 players each. The games obejective is to select and control appropriate heros from a selection and defeat the other team. The heros chosen by the teams are mutually exclusive. For the present problem we use the data obtained from the UCI Machine learning repository. The data was collected on the last day of the "The International" tournament held on the 13^{th} of August 2016 within a span of two hours using the API provided by the game developers.

Data Cleaning

For our convenience we will use only the data corresponding to the game type as "Tournament" and game mode as "Captains Mode". This data is composed of 41,229 rows. Further we decoupled the hero selections for the two teams into two set of 111 each for each side. The cell value of 1 indicates that the hero is chosen while 0 indicates not choosing. The hero names prefixed with "Choose" indicate the team whose result we are interested in while those hero names prefixed with "find" indicate the opposing team selecctions.

The majority of the data cleaning ws carried out in Microsoft Excel. Also the region column had to be converted from integer to a string to increase readability. See Appendix 1 for the structure of the dataframe.

Exploratory Data Analysis

Win/Loss Count

First, we would like to know the distribution of the games in terms of win/loss.

```
##
##     0     1
## 19567 21662
```

We can see that the win/loss is evenly distributed in our dataset. Next, we would like to know how the games are spread over the different server regions.

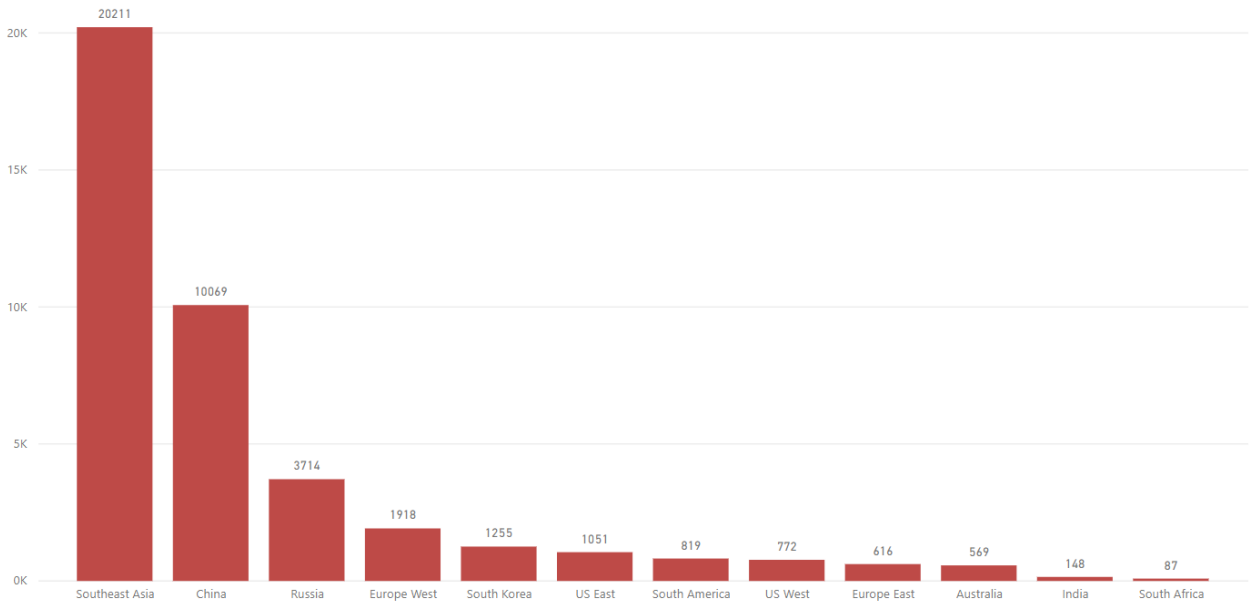


Figure 1: Game count by regions

Geographical Distribution

We can see that a majority of the games are played in Southeast Asia. The second highest number of games are played in China nearly half the number of games in Southeast asia. Both these regions together account for around 70% of the games in our dataset, hence our results might be relevant to this geographical setting.

Heroes by popularity

The top five most selected characters are Phantom Assasin, Pudge, Legion Commander, Mirana and Invoker respectively. Also another point to note is that a few characters are disproportionately used more frequently than the majority which is used relatively less frequently.

Modelling

Data Splitting

In order to test our model we split it into a train and a test group using 80:20 split.

```
set.seed(999)
train.index <- sample(1:nrow(dota2.hero), nrow(dota2.hero)*.8)
train.dota2 <- dota2.hero[train.index,]
test.dota2 <- dota2.hero[-train.index,]
```

Step Model

In order to use the step modeling procedure, we first create a base model with no independent variables and a full mode with all the independent variables. We then use these models to iteratively build models using a

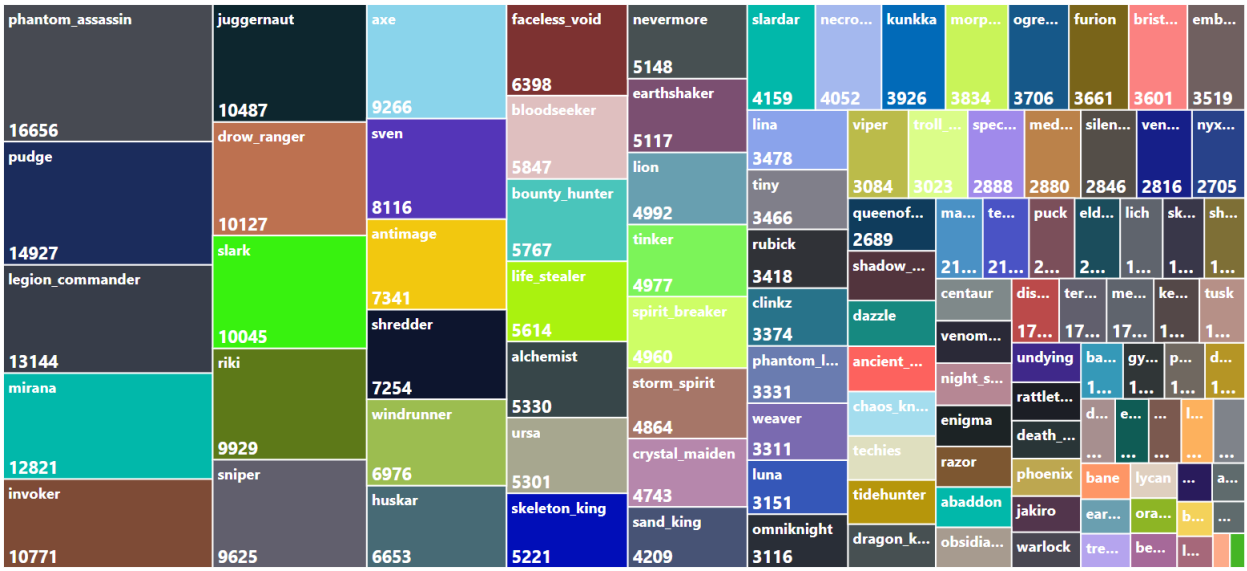


Figure 2: Heroes by popularity

combination of the inputs and minimizing the AIC score of the resultant model.

```
fullmodel <- glm(data=train.dota2, result~., family=binomial)
zeromodel <- glm(data=train.dota2, result~1, family=binomial)
#This step takes a long time to process and commented
#stepmodel <-  step(zeromodel, list(lower=formula(zeromodel),
                                    #upper=formula(fullmodel)),
                                      #direction="both",trace=0)
```

We finally arrive at the parsimonious model as suggested with an AIC score of 45520. The model coefficients are shown below. The coefficients with positive sign contribute towards winning while the coefficients with negative sign contribute towards the loosing.

```
m3<-glm(formula = result ~ find_furion + choose_skeleton_king + choose_furion +
          find_oracle + choose_nevermore + choose_morphling + choose_shadow_demon +
          choose_omniknight + choose_witch_doctor + choose_invoker +
          choose_mirana + find_lion + choose_pudge + find_pudge + choose_earth_spirit +
          choose_leshrac + choose_obsidian_destroyer + find_ogre_magi +
          find_ancient_apparition + choose_wisp + choose_rattletrap +
          find_faceless_void + choose_rubick + find_phantom_assassin +
          choose_visage + find_weaver + choose_spectre + choose_clinkz +
          find_omniknight + find_techies + find_riki + find_chaos_knight +
          choose_lion + choose_disruptor + choose_bane + choose_luna +
          find_spectre, family = binomial, data = train.dota2)
m3$coefficients
```

```
##                (Intercept)               find_furion1
##               0.036500552              -0.310194149
##      choose_skeleton_king1             choose_furion1
##              -0.185677276               0.218381416
##               find_oracle1         choose_nevermore1
##               0.253097530               0.102841402
##          choose_morphling1       choose_shadow_demon1
##               0.130992232               0.153488085
```

```
##         choose_omniknight1       choose_witch_doctor1
##                0.074632593                0.121521391
##            choose_invoker1             choose_mirana1
##                0.059903069                0.026273105
##                find_lion1              choose_pudge1
##               -0.117094460                0.090699493
##                find_pudge1       choose_earth_spirit1
##                0.059480362                0.243365485
##            choose_leshrac1 choose_obsidian_destroyer1
##                0.231515179                0.132245098
##            find_ogre_magi1   find_ancient_apparition1
##                0.160909292                0.094960665
##               choose_wisp1         choose_rattletrap1
##                0.367455372               -0.127754322
##        find_faceless_void1             choose_rubick1
##               -0.059480148                0.131655304
##    find_phantom_assassin1             choose_visage1
##               -0.058605378                0.094778501
##               find_weaver1            choose_spectre1
##               -0.000846098               -0.091354838
##             choose_clinkz1           find_omniknight1
##               -0.087152102                0.098210786
##              find_techies1                find_riki1
##                0.029123342                0.067378142
##         find_chaos_knight1              choose_lion1
##                0.061391420                0.061362569
##          choose_disruptor1              choose_bane1
##                0.129775627                0.109296543
##               choose_luna1             find_spectre1
##                0.064503418                0.001161157
```

Model Variables

We can see individual contribution each of the significant variables make towards the odds of winning. Choosing Wisp has the highest contribution (increases odds by 1.44 times) while finding Furion in the opponent's team reduces your odds of winning by 0.733 times (the lowest contribution) .

```
exp(m3$coefficients)
```

```
##                (Intercept)              find_furion1
##                  1.0371749                 0.7333046
##      choose_skeleton_king1             choose_furion1
##                  0.8305416                 1.2440615
##               find_oracle1          choose_nevermore1
##                  1.2880089                 1.1083156
##          choose_morphling1       choose_shadow_demon1
##                  1.1399589                 1.1658939
##         choose_omniknight1       choose_witch_doctor1
##                  1.0774882                 1.1292135
##            choose_invoker1             choose_mirana1
##                  1.0617336                 1.0266213
##                find_lion1              choose_pudge1
##                  0.8895012                 1.0949399
##                find_pudge1       choose_earth_spirit1
```

```
##                 1.0612849                 1.2755347
##           choose_leshrac1 choose_obsidian_destroyer1
##                 1.2605085                 1.1413880
##           find_ogre_magi1  find_ancient_apparition1
##                 1.1745784                 1.0996156
##              choose_wisp1        choose_rattletrap1
##                 1.4440554                 0.8800696
##       find_faceless_void1            choose_rubick1
##                 0.9422542                 1.1407151
##    find_phantom_assassin1            choose_visage1
##                 0.9430789                 1.0994153
##              find_weaver1           choose_spectre1
##                 0.9991543                 0.9126938
##            choose_clinkz1          find_omniknight1
##                 0.9165377                 1.1031953
##             find_techies1                find_riki1
##                 1.0295516                 1.0696999
##        find_chaos_knight1              choose_lion1
##                 1.0633150                 1.0632844
##         choose_disruptor1              choose_bane1
##                 1.1385729                 1.1154931
##              choose_luna1             find_spectre1
##                 1.0666292                 1.0011618
```

Model Evaluation

Training Data

ROC Chart

Firstly we would like to construct a confusion matrix to have an idea of the type 1 and type 2 errors. For this wewould like to first the ROC chart for the training dataset.

```
## Setting levels: control = 0, case = 1
```

From the tradeoff graph we observe that our model is performing better than the zero model (no independent variables) albeit only marginally. By visual inspection we find that a cutoff value of 0.5 gives us the mazimum sensitivity and specficity.

Confusion Matrix

Below shown is the specificity and sensitivity values for a cutoff value of 0.5 in the training data.

```
CUTOFF <- quantile(train.dota2$prob,0.5)
train.dota2$pred <- ifelse(train.dota2$prob > CUTOFF,1,0)
table(train.dota2$pred, train.dota2$result)
```

```
##
##        0    1
##   0 8266 8231
##   1 7441 9045
```

```
Conf(x = train.dota2$pred,ref=train.dota2$result, pos = 1)
```

```
##
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    1    0
##          1 9045 7441
##          0 8231 8266
##
##                Total n : 3e+04
##               Accuracy : 5.2485e-01
```

```
##                  95% CI : (5.1945e-01, 5.3023e-01)
##     No Information Rate : 5.2378e-01
##     P-Value [Acc > NIR] : 0.3519
##
##                   Kappa : 4.9707e-02
##  Mcnemar's Test P-Value : 2.93e-10
##
##             Sensitivity : 5.2356e-01
##             Specificity : 5.2626e-01
##          Pos Pred Value : 5.4865e-01
##          Neg Pred Value : 5.0106e-01
##              Prevalence : 5.2378e-01
##          Detection Rate : 2.7423e-01
##    Detection Prevalence : 4.9983e-01
##       Balanced Accuracy : 5.2491e-01
##          F-val Accuracy : 5.3581e-01
##
##        'Positive' Class : 1
```

Test data

ROC Chart

There's only a minute improvement in performance. This raises concerns about the variables taken into the model.

```
## Setting levels: control = 0, case = 1
```

Confusion Matrix

The sensitivity and specificity values are almost similar to the training case.

```
CUTOFF <- quantile(test.dota2$prob,0.5)
test.dota2$pred <- ifelse(test.dota2$prob > CUTOFF,1,0)
table(test.dota2$pred, test.dota2$result)
```

```
##
##        0    1
##   0 1988 2135
##   1 1872 2251
```

```
Conf(x = test.dota2$pred,ref=test.dota2$result, pos = 1)
```

```
##
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    1    0
##          1 2251 1872
##          0 2135 1988
##
##                Total n : 8e+03
##               Accuracy : 5.1407e-01
##                 95% CI : (5.0328e-01, 5.2485e-01)
##     No Information Rate : 5.3189e-01
##     P-Value [Acc > NIR] : 0.9994
##
##                  Kappa : 2.8135e-02
##  Mcnemar's Test P-Value : 3.49e-05
##
##             Sensitivity : 5.1322e-01
##             Specificity : 5.1503e-01
##          Pos Pred Value : 5.4596e-01
##          Neg Pred Value : 4.8217e-01
##              Prevalence : 5.3189e-01
##          Detection Rate : 2.7298e-01
##    Detection Prevalence : 5.0000e-01
##       Balanced Accuracy : 5.1412e-01
##          F-val Accuracy : 5.2909e-01
##
##        'Positive' Class : 1
```

Model Accuracy

```
mean(test.dota2$pred == test.dota2$result)
```

```
## [1] 0.5140674
```

The accuracy for the test dataset around 50%. This does not give us confidence in the model predictions.

Model Analysis

```
pR2(m3)
```

```
##           llh       llhNull            G2      McFadden          r2ML
## -2.273360e+04 -2.282474e+04  1.822731e+02  3.992883e-03  5.511031e-03
##          r2CU
##  7.353597e-03
```

```
hoslem.test(train.dota2$result, fitted(m3))
```

```
## Warning in Ops.factor(1, y): '-' not meaningful for factors
```

```
##
##  Hosmer and Lemeshow goodness of fit (GOF) test
##
## data:  train.dota2$result, fitted(m3)
## X-squared = 32983, df = 8, p-value < 2.2e-16
```

```
PseudoR2(m3,which = c("Nagelkerke","McFadden", "Nagel"))
```

```
##  Nagelkerke    McFadden  Nagelkerke
## 0.007353597 0.003992883 0.007353597
```

```
HosmerLemeshowTest(fitted(m3), train.dota2$result)
```

```
## Warning in Ops.factor(1, obs): '-' not meaningful for factors
```

```
## Warning in Ops.factor(1, obs): '-' not meaningful for factors
```

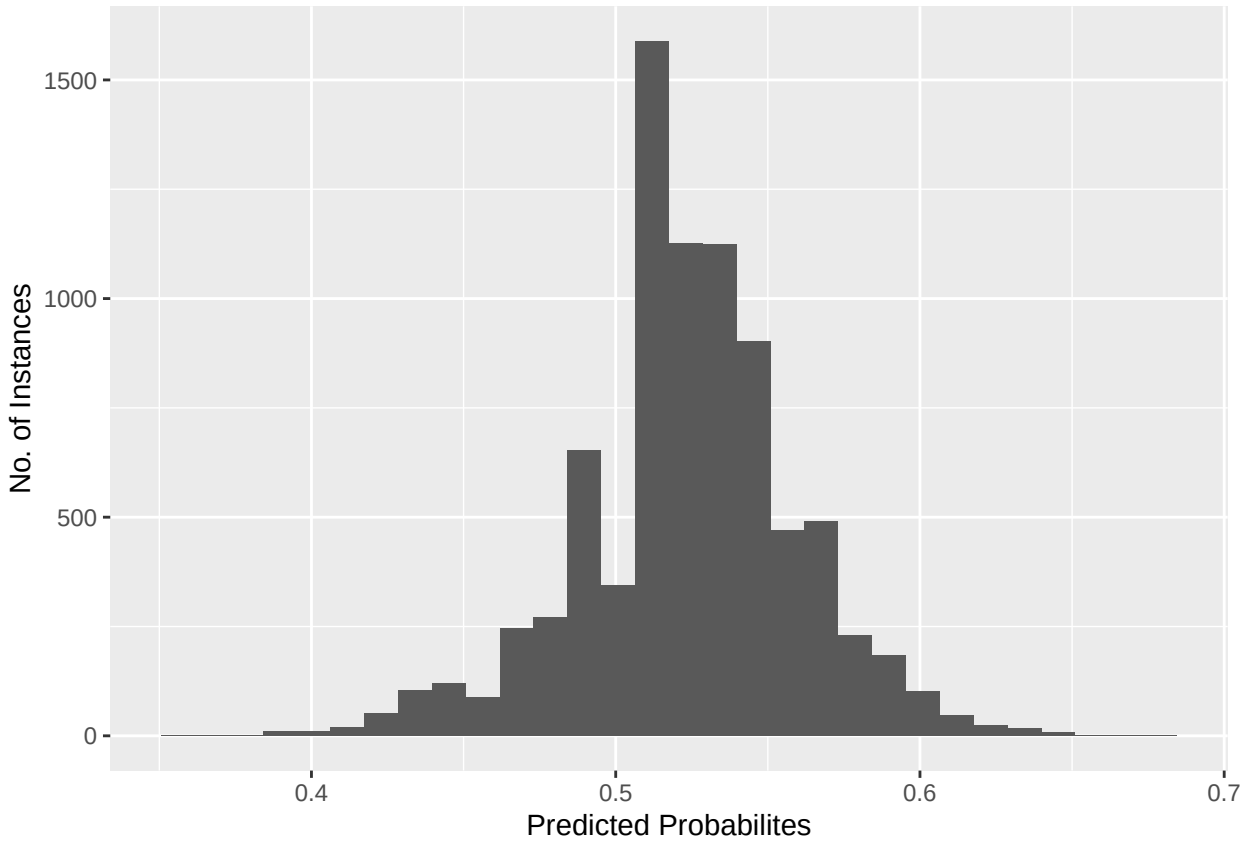
```
## $C
##
##  Hosmer-Lemeshow C statistic
##
## data:  fitted(m3) and train.dota2$result
## X-squared = 32983, df = 8, p-value < 2.2e-16
##
##
## $H
##
##  Hosmer-Lemeshow H statistic
##
## data:  fitted(m3) and train.dota2$result
## X-squared = 32983, df = 8, p-value < 2.2e-16
```

The Hosmer-Lemshow test tells us that there's no statitical difference between the obeserved and the predicted ditribtuions, i.e. there's no difference in the outcome distribution. However the very low values of (non-parametric) R^2 tell us that our model is not able to explain the differences in the variances. The variable of hero selection does not appropriately give us a better estimate in predicting the odds of winning the match.

If we draw the probability distribution of the outcome varibable from our model we see that many of the predicted probabilities are in the middle at the boundary. We don't have an inverted "U" shape. Hence the predictions are very similar to

```
## NULL
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Conclusions

From our logistic model the highest contirbution to winning odds is given by choosing Wisp (x1.44 times) while finding Furion in your opponent's team reduces your winning odds by 0.733 times (the lowest contribution). From the model evaluations we conclude that the the variables for hero selection alone does not give us a good model for predicting the winning odds in DoTA2. This gives us clues to look at interaction effects or the association rules. Also anothe point to note is the absence of the popular characters in our model. The popular characters (except Pudge) don't have a significant effect affecting winning odds.

Appendix 1

Structure of the data frame

Here the first column tells whether radiant team won (=1) or lost the match (=0). The region tell us the geographical server on which the game was played. The columns names with "choose_" before the hero name tells if the radiant team chose (=1) that particular hero or not (=0). The column names with "find_" before the hero name tells if the dire team chost (=1) that particular hero or not (=0). The entire dataset contains

```
str(dota2)
```

```
## 'data.frame':    41229 obs. of  224 variables:
##  $ result                    : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ region                    : Factor w/ 12 levels "Australia","China",..: 2 10 2 6 6 10 9 9 10 10 .
##  $ Choose.antimage           : Factor w/ 2 levels "0","1": 1 1 2 1 1 1 1 1 2 1 ...
##  $ choose_axe                : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_bane               : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_bloodseeker        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_crystal_maiden     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_drow_ranger        : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 1 1 1 1 ...
##  $ choose_earthshaker        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_juggernaut         : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ choose_mirana             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_morphling          : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 1 1 1 ...
##  $ choose_nevermore          : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_phantom_lancer     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 2 1 ...
##  $ choose_puck               : Factor w/ 2 levels "0","1": 1 1 2 1 1 1 1 1 1 1 ...
##  $ choose_pudge              : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 2 1 1 1 ...
##  $ choose_razor              : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_sand_king          : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_storm_spirit       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_sven               : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 1 1 1 ...
##  $ choose_tiny               : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 1 1 1 ...
##  $ choose_vengefulspirit     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_windrunner         : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 1 1 1 1 ...
##  $ choose_zuus               : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 1 1 1 ...
##  $ choose_kunkka             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_lina               : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_lion               : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_shadow_shaman      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_slardar            : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 1 1 1 ...
##  $ choose_tidehunter         : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_witch_doctor       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_lich               : Factor w/ 2 levels "0","1": 1 1 2 1 1 1 1 1 1 1 ...
##  $ choose_riki               : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ choose_enigma             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 2 1 1 1 ...
##  $ choose_tinker             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_sniper             : Factor w/ 2 levels "0","1": 1 1 2 1 1 1 2 1 1 1 ...
##  $ choose_necrolyte          : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_warlock            : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_beastmaster        : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 1 1 2 ...
##  $ choose_queenofpain        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_venomancer         : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_faceless_void      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
```

```
##  $ choose_skeleton_king      : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 1 1 1 ...
##  $ choose_death_prophet      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 2 1 1 ...
##  $ choose_phantom_assassin   : Factor w/ 2 levels "0","1": 1 2 1 2 1 1 2 1 1 2 ...
##  $ choose_pugna              : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_templar_assassin   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 2 1 2 ...
##  $ choose_viper              : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_luna               : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 2 1 ...
##  $ choose_dragon_knight      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_dazzle             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_rattletrap         : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_leshrac            : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_furion             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 2 1 ...
##  $ choose_life_stealer       : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 1 1 1 ...
##  $ choose_dark_seer          : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_clinkz             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_omniknight         : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_enchantress        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_huskar             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 2 1 1 1 ...
##  $ choose_night_stalker      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_broodmother        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_bounty_hunter      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_weaver             : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 1 1 1 ...
##  $ choose_jakiro             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_batrider           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_chen               : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_spectre            : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_ancient_apparition : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_doom_bringer       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_ursa               : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_spirit_breaker     : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 1 1 1 ...
##  $ choose_gyrocopter         : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_alchemist          : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_invoker            : Factor w/ 2 levels "0","1": 2 1 1 1 1 2 1 1 1 1 ...
##  $ choose_silencer           : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 1 1 1 ...
##  $ choose_obsidian_destroyer : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_lycan              : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_brewmaster         : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_shadow_demon       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_lone_druid         : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_chaos_knight       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_meepo              : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_treant             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_ogre_magi          : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ choose_undying            : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_rubick             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_disruptor          : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 1 1 ...
##  $ choose_nyx_assassin       : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 1 1 1 ...
##  $ choose_naga_siren         : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ choose_keeper_of_the_light: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_wisp               : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 1 1 1 1 ...
##  $ choose_visage             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_slark              : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 2 2 1 ...
##  $ choose_medusa             : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_troll_warlord      : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
```

```
##  $ choose_centaur            : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ choose_magnataur          : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 2 1 1 ...
##  $ choose_shredder           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##   [list output truncated]
```