Classification

Mason Cushing

2023-02-17

Write a paragraph explaining in general terms how linear models for classification work, and what are the strengths and weaknesses of these linear models.

Logistic regression calculates the log odds of the result given each input and finds the line of best fit for the resulting data points. The result is a set of values for each predictor that represents the slope of this line (which corresponds to the change in the log odds for each unit of the predictor) and a set of p-values that correspond to the relevance of each indicator.

Logistic regression works well with large data sets, but should no line be present in the graph that it adds the line of best fit to, the model will be underfit and thus somewhat inaccurate. The model has a high bias and low variance.

We begin by taking our Kaggle dataset (found at https://www.kaggle.com/datasets/datasnaek/chess (https://www.kaggle.com/datasets/datasnaek/chess)) and cleaning it for our purposes. I'll be using opening name and ratings for this assignment. Only the first word of each opening will be used, and I'll be removing the opening types that were only played a few times to avoid errors with data partitioning.

```
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
library(e1071)
games <- read.csv("~/Downloads/games.csv")
games$opening_name <- vapply(strsplit(games$opening_name," "),
                             `[`, 1, FUN.VALUE=character(1))
t <- table(games$opening_name)
uu <- unique(games$opening_name)
sort(uu)
```

```
##   [1] "Alekhine"            "Amar"                "Amazon"
##   [4] "Anderssen"           "Australian"          "Barnes"
##   [7] "Benko"               "Benoni"              "Bird"
##  [10] "Bishop's"            "Blackmar-Diemer"     "Blumenfeld"
##  [13] "Bogo-Indian"         "Borg"                "Budapest"
##  [16] "Canard"              "Caro-Kann"           "Carr"
##  [19] "Catalan"             "Center"              "Clemenz"
##  [22] "Colle"               "Crab"                "Creepy"
##  [25] "Czech"               "Danish"              "Doery"
##  [28] "Duras"               "Dutch"               "East"
##  [31] "Elephant"            "English"             "Englund"
##  [34] "Four"                "Franco-Benoni"       "French"
##  [37] "Gedult's"            "Giuoco"              "Global"
##  [40] "Goldsmith"           "Grob"                "Gruenfeld"
##  [43] "Guatemala"           "Gunderam"            "Hippopotamus"
##  [46] "Horwitz"             "Hungarian"           "Indian"
##  [49] "Irish"               "Italian"             "Kadas"
##  [52] "Kangaroo"            "King's"              "Latvian"
##  [55] "Lemming"             "Lion"                "London"
##  [58] "Mexican"             "Mieses"              "Mikenas"
##  [61] "Modern"              "Neo-Gruenfeld"       "Nimzo-Indian"
##  [64] "Nimzo-Larsen"        "Nimzowitsch"         "Nimzowitsch-Larsen"
##  [67] "Old"                 "Owen"                "Paleface"
##  [70] "Petrov:"             "Petrov's"            "Philidor"
##  [73] "Pirc"                "Polish"              "Ponziani"
##  [76] "Portuguese"          "Pterodactyl"         "Queen's"
##  [79] "Rat"                 "Reti"                "Richter-Veresov"
##  [82] "Robatsch"            "Rubinstein"          "Russian"
##  [85] "Ruy"                 "Saragossa"           "Scandinavian"
##  [88] "Scotch"              "Semi-Bononi"         "Semi-Slav"
##  [91] "Sicilian"            "Slav"                "Sodium"
##  [94] "St."                 "System:"             "Tarrasch"
##  [97] "Three"               "Torre"               "Trompowsky"
## [100] "Valencia"            "Van"                 "Van't"
## [103] "Vienna"              "Wade"                "Ware"
## [106] "Yusupov-Rubinstein" "Zukertort"
```

```
cc <- data.frame(uu,t)
cc <- cc[cc$Freq >= 6,]
games <- games[games$opening_name %in% cc$Var1,]
games <- games[games$winner != "draw",]
games$winner[games$winner == "white"]<-1
games$winner[games$winner == "black"]<-0
games$winner <- as.numeric(games$winner)
p <- sample(1:nrow(games), 0.8*nrow(games), replace=FALSE)
dtrain <- games[p,]
dtest <- games[-p,]
```

We next will be using a few functions to explore the data.

```r
table(games$opening_name)
```

```
##
##          Alekhine              Amar            Amazon         Anderssen
##               182                14                 5                25
##            Barnes             Benko            Benoni              Bird
##                16                18                57               142
##          Bishop's  Blackmar-Diemer        Blumenfeld       Bogo-Indian
##               306                61                14                 9
##              Borg          Budapest         Caro-Kann           Catalan
##                13                25               563                 6
##            Center           Clemenz             Colle              Crab
##               172                 9                26                 8
##            Danish             Duras             Dutch              East
##                68                11               119                21
##          Elephant           English           Englund              Four
##                71               758               108               355
##     Franco-Benoni            French          Gedult's            Giuoco
##                 8              1342                19               102
##         Goldsmith              Grob         Gruenfeld          Gunderam
##                17                35                59                11
##       Hippopotamus           Horwitz         Hungarian            Indian
##                 6               204               173               299
##           Italian             Kadas          Kangaroo            King's
##               934                27                 6              1580
##           Latvian            London           Mexican            Mieses
##                23                17                 6               102
##           Mikenas            Modern      Neo-Gruenfeld      Nimzo-Indian
##                37               216                 8               146
##      Nimzo-Larsen      Nimzowitsch Nimzowitsch-Larsen               Old
##               156               216                23                77
##              Owen          Paleface           Petrov:          Petrov's
##               162                10                 6                80
##          Philidor              Pirc            Polish          Ponziani
##               663               270                93                65
##        Portuguese           Queen's               Rat              Reti
##                22              2237                87                62
##   Richter-Veresov          Robatsch           Russian               Ruy
##                11                40               243               809
##         Saragossa      Scandinavian            Scotch         Semi-Slav
##                50               690               457                96
##          Sicilian              Slav               St.          Tarrasch
##              2502               228                44                26
##             Three             Torre        Trompowsky               Van
##               111                30                27                55
##             Van't            Vienna              Ware Yusupov-Rubinstein
##               352               134                35                20
##         Zukertort
##               308
```

```
mean(games$white_rating)
```

```
## [1] 1593.618
```

```
mean(games$black_rating)
```

```
## [1] 1586.366
```

```
median(games$white_rating)
```

```
## [1] 1564
```

```
median(games$black_rating)
```

```
## [1] 1560
```

```
mean(games$winner == 1)
```

```
## [1] 0.5231948
```

```
mean(games$winner)
```

```
## [1] 0.5231948
```

```
diff<-games$white_rating-games$black_rating
mean(diff)
```

```
## [1] 7.252361
```

We next will add two box plots to shed some more light on the data.

```
t <- table(games$opening_name)
uu <- unique(games$opening_name)
sort(uu)
```

```
##  [1] "Alekhine"             "Amar"                 "Amazon"              
##  [4] "Anderssen"            "Barnes"               "Benko"               
##  [7] "Benoni"               "Bird"                 "Bishop's"            
## [10] "Blackmar-Diemer"      "Blumenfeld"           "Bogo-Indian"         
## [13] "Borg"                 "Budapest"             "Caro-Kann"           
## [16] "Catalan"              "Center"               "Clemenz"             
## [19] "Colle"                "Crab"                 "Danish"              
## [22] "Duras"                "Dutch"                "East"                
## [25] "Elephant"             "English"              "Englund"             
## [28] "Four"                 "Franco-Benoni"        "French"              
## [31] "Gedult's"             "Giuoco"               "Goldsmith"           
## [34] "Grob"                 "Gruenfeld"            "Gunderam"            
## [37] "Hippopotamus"         "Horwitz"              "Hungarian"           
## [40] "Indian"               "Italian"              "Kadas"               
## [43] "Kangaroo"             "King's"               "Latvian"             
## [46] "London"               "Mexican"              "Mieses"              
## [49] "Mikenas"              "Modern"               "Neo-Gruenfeld"       
## [52] "Nimzo-Indian"         "Nimzo-Larsen"         "Nimzowitsch"         
## [55] "Nimzowitsch-Larsen" "Old"                  "Owen"                
## [58] "Paleface"             "Petrov:"              "Petrov's"            
## [61] "Philidor"             "Pirc"                 "Polish"              
## [64] "Ponziani"             "Portuguese"           "Queen's"             
## [67] "Rat"                  "Reti"                 "Richter-Veresov"     
## [70] "Robatsch"             "Russian"              "Ruy"                 
## [73] "Saragossa"            "Scandinavian"         "Scotch"              
## [76] "Semi-Slav"            "Sicilian"             "Slav"                
## [79] "St."                  "Tarrasch"             "Three"               
## [82] "Torre"                "Trompowsky"           "Van"                 
## [85] "Van't"                "Vienna"               "Ware"                
## [88] "Yusupov-Rubinstein" "Zukertort"
```

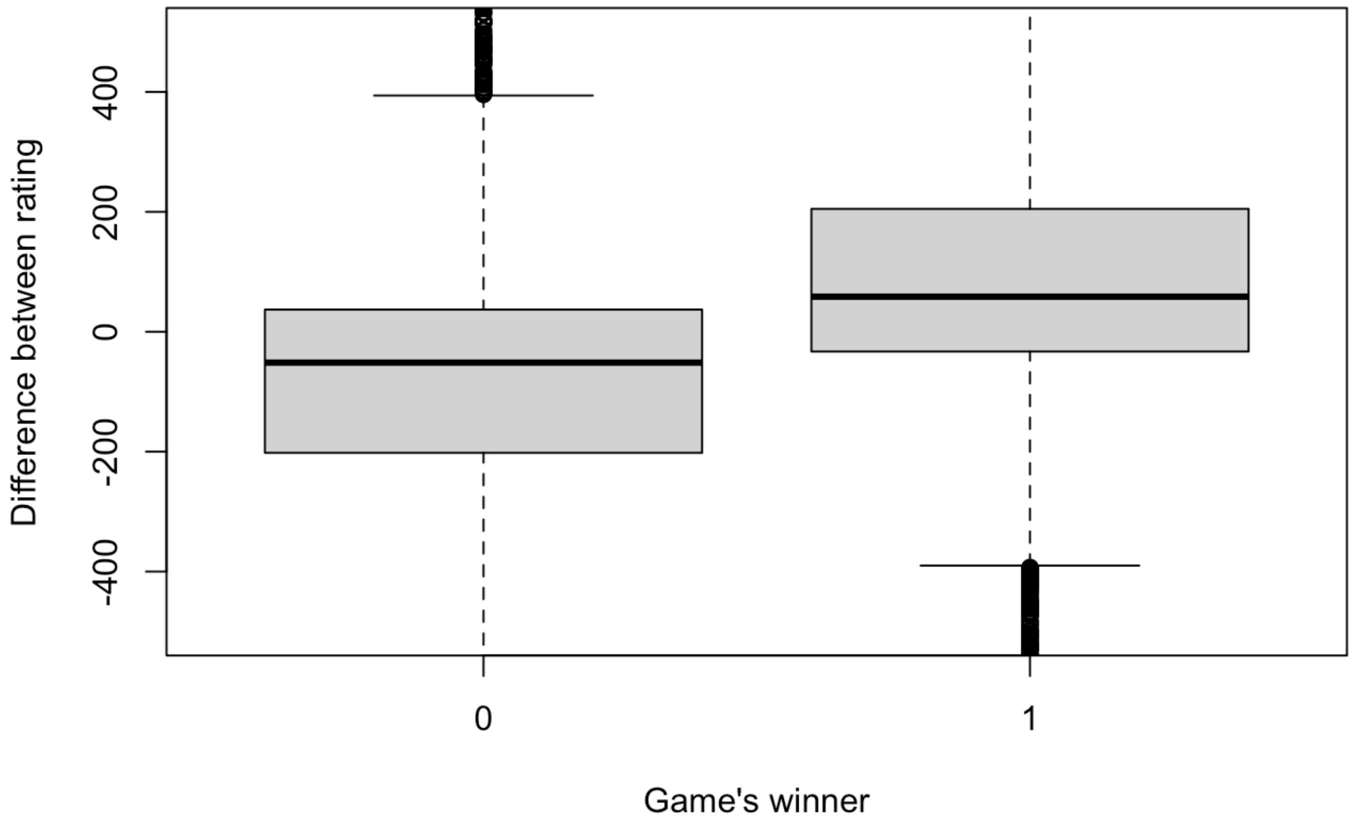
```
cc <- data.frame(uu,t)
cc <- cc[cc$Freq >= 700,]
g2 <- games[games$opening_name %in% cc$Var1,]
mean(games[games$winner == 0,]$white_rating-games[games$winner == 0,]$black_rating)
```

```
## [1] -88.99087
```

```
mean(games[games$winner == 1,]$white_rating-games[games$winner == 1,]$black_rating)
```

```
## [1] 94.96209
```

```
boxplot(diff~winner,data=games,
        xlab="Game's winner",ylab="Difference between rating",ylim=c(-500,500))
```




```
mean(g2[g2$opening_name=="English",]$winner)
```

```
## [1] 0.5659631
```

```
mean(g2[g2$opening_name=="French",]$winner)
```

```
## [1] 0.5134128
```

```
mean(g2[g2$opening_name=="Italian",]$winner)
```

```
## [1] 0.5171306
```

```
mean(g2[g2$opening_name=="King's",]$winner)
```

```
## [1] 0.5341772
```

```
mean(g2[g2$opening_name=="Queen's",]$winner)
```

```
## [1] 0.5386679
```

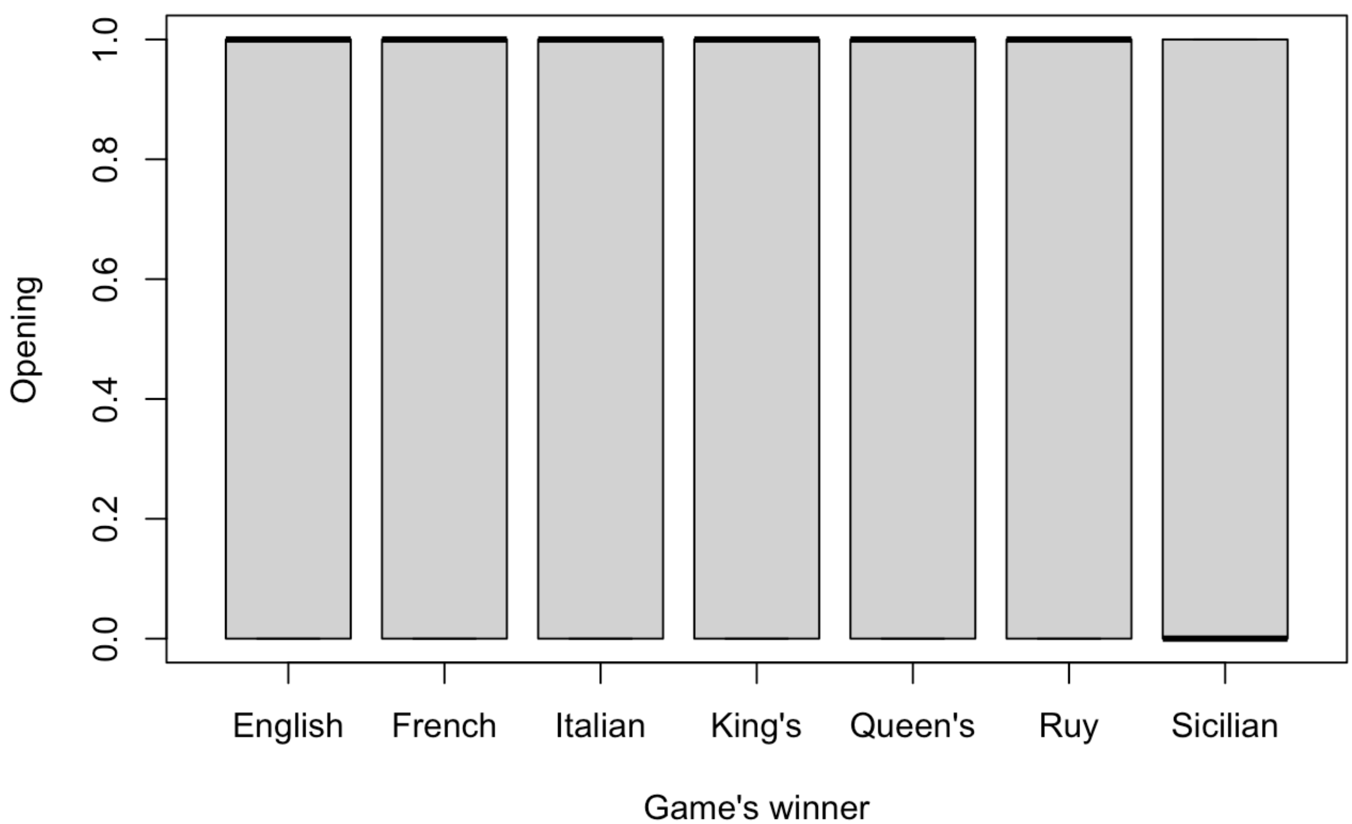
```
mean(g2[g2$opening_name=="Ruy",]$winner)
```

```
## [1] 0.5574784
```

```
mean(g2[g2$opening_name=="Sicilian",]$winner)
```

```
## [1] 0.4808153
```

```
boxplot(winner~opening_name,data=g2,xlab="Game's winner",ylab="Opening")
```



We notice above that the difference in rating of nearly 100 points is the average rating difference for the winner (i.e. on average, the winner is 100 more rating points stronger than their opponent).

We notice from our second graph (which contains data on the seven most common openings found in the dataset) that all slightly favor white except the Sicilian (as shown both by the statistics included above and the placement of the medians in each box plot shown).

Next, we use our data to create a model to classify the data. Given the predictors in the data about the game, we are trying to predict the winner. We will start by using logistic regression.

```
model <- glm(winner ~ opening_name + white_rating + black_rating,
             family=binomial, data=dtrain)
summary(model)
```

```
##
## Call:
## glm(formula = winner ~ opening_name + white_rating + black_rating,
##     family = binomial, data = dtrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6464  -1.0936   0.4513   1.0485   3.0873
##
## Coefficients:
##                                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)                    3.865e-01  2.111e-01   1.831  0.06707 .
## opening_nameAmar               5.990e-01  7.682e-01   0.780  0.43551
## opening_nameAmazon            -4.599e-01  9.339e-01  -0.492  0.62244
## opening_nameAnderssen          4.984e-01  5.731e-01   0.870  0.38453
## opening_nameBarnes             4.688e-01  7.215e-01   0.650  0.51589
## opening_nameBenko              1.193e-01  6.189e-01   0.193  0.84713
## opening_nameBenoni            -2.103e-01  3.808e-01  -0.552  0.58070
## opening_nameBird              -5.091e-01  2.689e-01  -1.894  0.05828 .
## opening_nameBishop's           8.141e-02  2.306e-01   0.353  0.72400
## opening_nameBlackmar-Diemer   -1.487e-01  3.510e-01  -0.424  0.67183
## opening_nameBlumenfeld         9.592e-01  7.289e-01   1.316  0.18821
## opening_nameBogo-Indian       -4.707e-01  8.160e-01  -0.577  0.56403
## opening_nameBorg              -6.305e-01  8.167e-01  -0.772  0.44015
## opening_nameBudapest          -1.647e+00  6.147e-01  -2.680  0.00737 **
## opening_nameCaro-Kann         -2.528e-02  2.089e-01  -0.121  0.90364
## opening_nameCatalan           -1.038e-01  1.440e+00  -0.072  0.94254
## opening_nameCenter            -4.366e-01  2.620e-01  -1.667  0.09554 .
## opening_nameClemenz            9.754e-02  8.376e-01   0.116  0.90729
## opening_nameColle             -8.633e-01  5.435e-01  -1.589  0.11217
## opening_nameCrab              -1.338e+01  1.999e+02  -0.067  0.94664
## opening_nameDanish             3.401e-01  3.506e-01   0.970  0.33191
## opening_nameDuras             -2.789e-01  8.159e-01  -0.342  0.73242
## opening_nameDutch             -3.455e-01  2.863e-01  -1.207  0.22759
## opening_nameEast               8.951e-03  5.497e-01   0.016  0.98701
## opening_nameElephant           3.590e-01  3.451e-01   1.040  0.29827
## opening_nameEnglish            4.526e-02  2.009e-01   0.225  0.82180
## opening_nameEnglund            3.376e-02  2.953e-01   0.114  0.90899
```

```
## opening_nameFour                 1.638e-01  2.242e-01   0.731  0.46502
## opening_nameFranco-Benoni        1.307e+01  2.223e+02   0.059  0.95311
## opening_nameFrench              -1.303e-01  1.930e-01  -0.675  0.49942
## opening_nameGedult's            -4.375e-01  6.523e-01  -0.671  0.50242
## opening_nameGiuoco              -4.327e-01  2.982e-01  -1.451  0.14674
## opening_nameGoldsmith            1.444e-01  7.352e-01   0.196  0.84434
## opening_nameGrob                -8.755e-01  5.119e-01  -1.710  0.08723 .
## opening_nameGruenfeld           -2.580e-02  3.543e-01  -0.073  0.94194
## opening_nameGunderam             5.986e-01  7.418e-01   0.807  0.41973
## opening_nameHippopotamus         1.263e+01  2.544e+02   0.050  0.96040
## opening_nameHorwitz             -7.197e-02  2.482e-01  -0.290  0.77185
## opening_nameHungarian           -3.753e-01  2.604e-01  -1.441  0.14953
## opening_nameIndian              -3.254e-01  2.310e-01  -1.409  0.15887
## opening_nameItalian             -1.367e-01  1.982e-01  -0.689  0.49054
## opening_nameKadas               -1.795e+00  6.483e-01  -2.769  0.00562 **
## opening_nameKangaroo            -1.359e+00  9.911e-01  -1.371  0.17028
## opening_nameKing's              -9.644e-02  1.915e-01  -0.504  0.61447
## opening_nameLatvian             -3.900e-01  5.219e-01  -0.747  0.45493
## opening_nameLondon              -1.006e+00  7.080e-01  -1.421  0.15539
## opening_nameMexican             -1.354e+00  1.157e+00  -1.171  0.24178
## opening_nameMieses              -1.372e-02  3.014e-01  -0.046  0.96369
## opening_nameMikenas             -8.684e-01  4.447e-01  -1.953  0.05083 .
## opening_nameModern              -2.984e-01  2.433e-01  -1.227  0.21986
## opening_nameNeo-Gruenfeld        1.532e+00  1.151e+00   1.331  0.18302
## opening_nameNimzo-Indian         1.158e-01  2.707e-01   0.428  0.66893
## opening_nameNimzo-Larsen        -1.526e-01  2.643e-01  -0.577  0.56365
## opening_nameNimzowitsch          4.759e-01  2.564e-01   1.856  0.06340 .
## opening_nameNimzowitsch-Larsen  2.111e-01  5.189e-01   0.407  0.68411
## opening_nameOld                  7.776e-02  3.253e-01   0.239  0.81107
## opening_nameOwen                -4.266e-01  2.634e-01  -1.619  0.10540
## opening_namePaleface             2.846e-02  7.121e-01   0.040  0.96812
## opening_namePetrov:             -4.064e-01  8.964e-01  -0.453  0.65033
## opening_namePetrov's            -4.836e-01  3.271e-01  -1.479  0.13926
## opening_namePhilidor             7.407e-02  2.048e-01   0.362  0.71765
## opening_namePirc                 1.233e-01  2.382e-01   0.518  0.60466
## opening_namePolish               5.404e-01  3.142e-01   1.720  0.08545 .
## opening_namePonziani            -6.776e-01  3.612e-01  -1.876  0.06064 .
## opening_namePortuguese          -2.351e-01  5.388e-01  -0.436  0.66264
## opening_nameQueen's             -4.847e-02  1.889e-01  -0.257  0.79751
## opening_nameRat                  3.079e-01  3.244e-01   0.949  0.34254
## opening_nameReti                -1.468e-01  3.695e-01  -0.397  0.69110
## opening_nameRichter-Veresov     -1.465e+00  8.921e-01  -1.642  0.10062
## opening_nameRobatsch             4.438e-02  4.196e-01   0.106  0.91576
## opening_nameRussian              2.365e-01  2.409e-01   0.981  0.32639
## opening_nameRuy                  8.690e-03  2.010e-01   0.043  0.96552
## opening_nameSaragossa            1.858e-01  3.857e-01   0.482  0.62997
## opening_nameScandinavian        -8.356e-02  2.039e-01  -0.410  0.68191
## opening_nameScotch              -5.475e-02  2.154e-01  -0.254  0.79934
## opening_nameSemi-Slav           -5.158e-02  2.992e-01  -0.172  0.86313
## opening_nameSicilian            -1.349e-01  1.884e-01  -0.716  0.47381
```

```
## opening_nameSlav                -2.820e-01  2.443e-01  -1.154  0.24836
## opening_nameSt.                  2.725e-01  4.244e-01   0.642  0.52077
## opening_nameTarrasch            -4.489e-01  5.010e-01  -0.896  0.37018
## opening_nameThree               -1.719e-02  2.919e-01  -0.059  0.95303
## opening_nameTorre               -7.635e-01  4.993e-01  -1.529  0.12621
## opening_nameTrompowsky           8.476e-02  4.939e-01   0.172  0.86374
## opening_nameVan                 -2.194e-01  3.923e-01  -0.559  0.57600
## opening_nameVan't               -5.769e-01  2.259e-01  -2.554  0.01064 *
## opening_nameVienna               1.432e-01  2.813e-01   0.509  0.61077
## opening_nameWare                -2.914e-01  4.844e-01  -0.602  0.54750
## opening_nameYusupov-Rubinstein -2.439e-01  6.773e-01  -0.360  0.71874
## opening_nameZukertort            1.967e-01  2.320e-01   0.848  0.39651
## white_rating                     3.818e-03  1.017e-04  37.555  < 2e-16 ***
## black_rating                    -3.957e-03  1.022e-04 -38.701  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 21099  on 15243  degrees of freedom
## Residual deviance: 18544  on 15153  degrees of freedom
## AIC: 18726
##
## Number of Fisher Scoring iterations: 12
```

We see above that the various predictors are mostly not very relevant. The various openings don't seem to affect the outcome spectacularly much. The p value will have a number of * next to it (or a dot) and the significance of these markings is denoted, and the more * there are, the better a predictor is The value to the left is the difference in the log odds per unit of difference (in the case of the openings it either applies or doesn't) and the other values are the errors and another fitting statistic respectively. Neither of the middle two values are terribly significant to a human analysis.

Next, we use this model to predict the winner. We then compare its results to the actual results and use this to compute the accuracy.

```
p<-predict(model,newdata=dtest,type="response")
predictions<-round(p)
mean(predictions == dtest$winner)
```

```
## [1] 0.6487408
```

```
predictions<-factor(predictions,levels=c("0","1"))
actuals<-factor(dtest$winner,levels=c('0','1'))
confusionMatrix(predictions,actuals)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1049  567
##          1  772 1424
##
##                Accuracy : 0.6487
##                  95% CI : (0.6333, 0.6639)
##     No Information Rate : 0.5223
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.2927
##
##  Mcnemar's Test P-Value : 2.476e-08
##
##             Sensitivity : 0.5761
##             Specificity : 0.7152
##          Pos Pred Value : 0.6491
##          Neg Pred Value : 0.6485
##              Prevalence : 0.4777
##          Detection Rate : 0.2752
##    Detection Prevalence : 0.4239
##       Balanced Accuracy : 0.6456
##
##        'Positive' Class : 0
##
```

We will next use naive Bayes to predict the same data. Output is the accuracy of this new model.

```
model2 <- naiveBayes(winner ~ opening_name + white_rating + black_rating,
                     data=dtrain)
model2$tables$opening_name
```

```
##    opening_name
## Y       Alekhine         Amar       Amazon    Anderssen       Barnes
##   0 0.0089470062 0.0005505850 0.0004129387 0.0008258775 0.0004129387
##   1 0.0102769771 0.0006266449 0.0002506580 0.0016292769 0.0011279609
##    opening_name
## Y          Benko       Benoni         Bird     Bishop's Blackmar-Diemer
##   0 0.0008258775 0.0027529250 0.0101858224 0.0134893324    0.0034411562
##   1 0.0008773029 0.0023812508 0.0060157915 0.0171700714    0.0032585537
##    opening_name
## Y     Blumenfeld  Bogo-Indian         Borg     Budapest    Caro-Kann
##   0 0.0004129387 0.0005505850 0.0006882312 0.0023399862 0.0289057123
##   1 0.0010026319 0.0005013160 0.0005013160 0.0005013160 0.0298282993
##    opening_name
## Y        Catalan       Center      Clemenz        Colle         Crab
##   0 0.0001376462 0.0103234687 0.0005505850 0.0016517550 0.0009635237
```

```
##   1 0.0001253290 0.0075197393 0.0003759870 0.0008773029 0.0000000000
##    opening_name
## Y         Danish        Duras        Dutch         East     Elephant
##   0 0.0028905712 0.0006882312 0.0067446662 0.0011011700 0.0028905712
##   1 0.0038851986 0.0005013160 0.0058904625 0.0010026319 0.0045118436
##    opening_name
## Y        English      Englund         Four Franco-Benoni       French
##   0 0.0386785960 0.0052305575 0.0169304886  0.0000000000 0.0740536820
##   1 0.0434891590 0.0063917784 0.0193006642  0.0006266449 0.0706855496
##    opening_name
## Y       Gedult's       Giuoco    Goldsmith         Grob    Gruenfeld
##   0 0.0013764625 0.0064693737 0.0004129387 0.0026152787 0.0034411562
##   1 0.0007519739 0.0045118436 0.0010026319 0.0010026319 0.0031332247
##    opening_name
## Y       Gunderam Hippopotamus      Horwitz    Hungarian       Indian
##   0 0.0004129387 0.0000000000 0.0103234687 0.0103234687 0.0183069511
##   1 0.0008773029 0.0005013160 0.0112796090 0.0078957263 0.0127835568
##    opening_name
## Y        Italian        Kadas     Kangaroo       King's      Latvian
##   0 0.0483138334 0.0022023400 0.0004129387 0.0824501032 0.0012388162
##   1 0.0487529766 0.0005013160 0.0002506580 0.0854743702 0.0013786189
##    opening_name
## Y         London      Mexican       Mieses      Mikenas       Modern
##   0 0.0012388162 0.0004129387 0.0066070200 0.0020646937 0.0126634549
##   1 0.0003759870 0.0002506580 0.0046371726 0.0017546058 0.0107782930
##    opening_name
## Y   Neo-Gruenfeld Nimzo-Indian Nimzo-Larsen  Nimzowitsch Nimzowitsch-Larsen
##   0  0.0001376462 0.0075705437 0.0092222987 0.0070199587       0.0011011700
##   1  0.0006266449 0.0080210553 0.0073944103 0.0141621757       0.0013786189
##    opening_name
## Y            Old         Owen     Paleface      Petrov:     Petrov's
##   0 0.0039917412 0.0088093599 0.0006882312 0.0004129387 0.0046799725
##   1 0.0045118436 0.0076450683 0.0005013160 0.0003759870 0.0035092117
##    opening_name
## Y       Philidor         Pirc       Polish     Ponziani   Portuguese
##   0 0.0297315898 0.0119752237 0.0037164487 0.0038540950 0.0012388162
##   1 0.0382253415 0.0146634917 0.0065171074 0.0025065798 0.0010026319
##    opening_name
## Y        Queen's          Rat         Reti Richter-Veresov     Robatsch
##   0 0.1130075705 0.0033035100 0.0031658637    0.0009635237 0.0020646937
##   1 0.1213184610 0.0061411204 0.0032585537    0.0002506580 0.0021305928
##    opening_name
## Y        Russian          Ruy    Saragossa Scandinavian       Scotch
##   0 0.0100481762 0.0388162423 0.0023399862 0.0356503785 0.0213351686
##   1 0.0149141496 0.0453690939 0.0028825667 0.0367213937 0.0244391528
##    opening_name
## Y      Semi-Slav     Sicilian         Slav          St.     Tarrasch
##   0 0.0052305575 0.1441156228 0.0122505162 0.0016517550 0.0015141087
##   1 0.0055144755 0.1205664870 0.0116555959 0.0030078957 0.0012532899
##    opening_name
```

```
## Y           Three        Torre   Trompowsky          Van        Van't
##   0 0.0055058500 0.0022023400 0.0013764625 0.0034411562 0.0264280798
##   1 0.0057651335 0.0010026319 0.0015039479 0.0022559218 0.0127835568
##    opening_name
## Y          Vienna         Ware Yusupov-Rubinstein    Zukertort
##   0 0.0060564350 0.0019270475       0.0009635237 0.0136269787
##   1 0.0076450683 0.0013786189       0.0006266449 0.0175460584
```

```
model2$tables$white_rating
```

```
##    white_rating
## Y       [,1]     [,2]
##   0 1551.295 281.9329
##   1 1634.721 288.7209
```

```
model2$tables$black_rating
```

```
##    black_rating
## Y       [,1]     [,2]
##   0 1641.184 289.5239
##   1 1539.231 282.3095
```

```
p2<-predict(model2,newdata=dtest)#,type="response")
predictions2<-as.numeric(p2)
predictions2<-predictions2-1
mean(predictions2 == dtest$winner)
```

```
## [1] 0.6151626
```

```
predictions2<-factor(predictions2,levels=c("0","1"))
actuals<-factor(dtest$winner,levels=c('0','1'))
confusionMatrix(predictions2,actuals)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0  840  486
##          1  981 1505
##
##                Accuracy : 0.6152
##                  95% CI : (0.5995, 0.6306)
##     No Information Rate : 0.5223
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.2197
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4613
##             Specificity : 0.7559
##          Pos Pred Value : 0.6335
##          Neg Pred Value : 0.6054
##              Prevalence : 0.4777
##          Detection Rate : 0.2204
##    Detection Prevalence : 0.3478
##       Balanced Accuracy : 0.6086
##
##        'Positive' Class : 0
##
```

Overall, Naive Bayes had a slightly lower accuracy in all of the runs I tested this code in. While Naive Bayes works better for smaller datasets, logistic regression works better for larger ones like this dataset. Both algorithms are high bias and low variance, but Naive Bayes is more extreme in both of those metrics.

The benefits of using accuracy as a main measurement is that it directly and intuitively measures how well the model does on a set of data. For scenarios where profit might increase proportionately with the accuracy of predictions, knowing the accuracy is vital. In some scenarios, the accuracy might not tell the whole story, but given enough tests of the method and enough data, eventually it will tell most of it.

The other metrics, namely specificity, sensitivity, and kappa, all corroborate with the accuracy for the most part. Interestingly, the specificity is consistently much higher than the sensitivity in this project, meaning that it is much more likely to say that black won when white did than the other way around. Also, the kappa value isn’t perfect, but it is enough to indicate that some real predictors are being found.