Blog 5: Logistic Regresssion

Michael O'Donnell

November 1, 2020

In this Blog I will setup a simple Logistic Regression model on NBA data.

Import Libraries

```r
# load required packages
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```r
library(MASS)
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

```r
library(caret)
```

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

```r
library(RCurl)
```

```
## Loading required package: bitops
```

```r
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
library(RCurl)
library(haven)
```

Now, start by loading a dataset This dataset contains data for all NBA teams from 2014-2018

```r
nbaData <- read.csv("data/nba_data.csv")
colnames(nbaData)[1] <- "Team"
nbaData$WinningTeam <- nbaData$WinPercentage
nbaData$WinningTeam[nbaData$WinningTeam > .5] <- 1
nbaData$WinningTeam[nbaData$WinningTeam <= .5] <- 0

#head(nbaData, 1)
str(nbaData)
```

```
## 'data.frame':    214 obs. of  40 variables:
##  $ Team                      : Factor w/ 30 levels "Atlanta Hawks",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ Season                    : int  2018 2018 2018 2018 2018 2018 2018 2018 2018 2018 ...
##  $ SeasonType                : Factor w/ 2 levels "POFF","REG": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Win                       : int  28 49 42 39 22 19 33 53 41 57 ...
##  $ Loss                      : int  53 33 40 43 60 63 48 28 40 25 ...
##  $ MatchCount                : int  81 82 82 82 82 82 81 81 81 82 ...
##  $ WinPercentage             : num  0.346 0.598 0.512 0.476 0.268 ...
##  $ Pts                       : num  113 112 112 111 105 ...
##  $ OppPts                    : num  119 108 112 112 113 ...
##  $ Pace                      : num  103.5 99 100.3 97.8 98.2 ...
##  $ OffEff                    : num  108 113 110 112 106 ...
##  $ DefEff                    : num  115 108 110 114 114 ...
##  $ EFgPercentage             : num  0.521 0.534 0.52 0.514 0.505 0.503 0.517 0.528 0.51 0.564 ...
##  $ OppEFgPercentage          : num  0.541 0.514 0.512 0.538 0.541 0.564 0.521 0.522 0.527 0.508 ..
##  $ TsPercentage              : num  0.555 0.567 0.556 0.554 0.541 0.54 0.554 0.558 0.545 0.596 ...
##  $ OppTsPercentage           : num  0.58 0.55 0.548 0.57 0.573 0.593 0.556 0.557 0.563 0.546 ...
##  $ RebRate                   : num  50.1 49.2 50.2 48.9 48 ...
##  $ EffPts                    : num  125 132 123 124 115 ...
##  $ OppEffPts                 : num  138 120 127 132 133 ...
##  $ FastBreakPts              : num  15.3 16.2 11.6 11.7 12.1 ...
##  $ OppFBPts                  : num  16.5 13.2 11.8 13.3 13 ...
##  $ PointsInPaint             : num  51.2 44.8 48.8 46.8 50.8 ...
##  $ OppPointsInPaint          : num  49.4 45.9 51.2 49 49.1 ...
##  $ PointsOffTO               : num  21.1 14.8 17.4 13.6 16.6 ...
##  $ OppPointsOffTO            : num  16.9 18.1 15.4 16.1 15.2 ...
##  $ SecondChancePTS           : num  14.1 12.5 13.8 13 10.9 ...
##  $ OppSecondChancePTS        : num  14.5 13.5 14.4 13.4 13.4 ...
##  $ PersonalFoulsPG           : num  23.5 21.5 20.4 18.9 20.3 ...
```

```
##  $ OppPersonalFoulsPG          : num  22.1 22 19.5 20.6 18.7 ...
##  $ ShootingFoulsPG             : num  14.9 12.3 12.1 10.9 12 ...
##  $ ShootingFoulsDrawnPG        : num  12.6 13.4 10.5 12.3 11.2 ...
##  $ LessThnEightFeedUsage       : num  43.5 43.5 36.2 41.9 46.2 ...
##  $ EightToSixteenFeedUsage     : num  11.5 11.5 14.8 12 14.3 ...
##  $ SixteenToTwentyFourFeetUsage: num  4.8 4.89 10.9 8.39 10.05 ...
##  $ TwentyFourPlusFeetUsage     : num  39.9 40 38 37.2 29.2 ...
##  $ AvgShotDistance             : num  13.1 13.2 14 13.4 11.9 ...
##  $ OppAvgShotDistance          : num  13.3 12.9 13.5 13.2 13.2 ...
##  $ AvgMadeShotDistance         : num  10.34 10.7 11.64 10.96 9.58 ...
##  $ OppMadeAvgShotDis           : num  10.8 10.4 10.8 10.6 10.6 ...
##  $ WinningTeam                 : num  0 1 1 0 0 0 0 1 1 1 ...
```

Binary Logistic Regression

```
all_preds = glm(WinningTeam ~ Pts+OppPts+Pace+OffEff+DefEff+FastBreakPts, family = binomial(), data = nl
summary(all_preds)
```

```
##
## Call:
## glm(formula = WinningTeam ~ Pts + OppPts + Pace + OffEff + DefEff +
##     FastBreakPts, family = binomial(), data = nbaData)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8437  -0.3562  -0.0194   0.3500   2.2807
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -28.69854   35.51435  -0.808    0.419
## Pts           -0.02397    0.83247  -0.029    0.977
## OppPts        -0.05538    0.78178  -0.071    0.944
## Pace           0.24356    0.41943   0.581    0.561
## OffEff         0.84244    0.79723   1.057    0.291
## DefEff        -0.71051    0.74869  -0.949    0.343
## FastBreakPts  -0.06693    0.08760  -0.764    0.445
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 294.40  on 213  degrees of freedom
## Residual deviance: 124.33  on 207  degrees of freedom
## AIC: 138.33
##
## Number of Fisher Scoring iterations: 7
```

```
nbaData$preds = ifelse(all_preds$fitted.values > 0.5, 1, 0)

# look at confusion matrix
cm = confusionMatrix(as_factor(nbaData$preds), as_factor(nbaData$WinningTeam), positive = "1")
cm
```

```
## Confusion Matrix and Statistics
```

```
##
##           Reference
## Prediction   0   1
##          0 105  16
##          1  13  80
##
##                Accuracy : 0.8645
##                  95% CI : (0.8112, 0.9073)
##     No Information Rate : 0.5514
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.7253
##
##  Mcnemar's Test P-Value : 0.7103
##
##             Sensitivity : 0.8333
##             Specificity : 0.8898
##          Pos Pred Value : 0.8602
##          Neg Pred Value : 0.8678
##              Prevalence : 0.4486
##          Detection Rate : 0.3738
##    Detection Prevalence : 0.4346
##       Balanced Accuracy : 0.8616
##
##        'Positive' Class : 1
##
```

Now, step through variables to find optimal model

```
step_all_preds = stepAIC(all_preds)
```

```
## Start:  AIC=138.33
## WinningTeam ~ Pts + OppPts + Pace + OffEff + DefEff + FastBreakPts
##
##                Df Deviance    AIC
## - Pts           1   124.33 136.33
## - OppPts        1   124.34 136.34
## - Pace          1   124.67 136.67
## - FastBreakPts  1   124.92 136.92
## - DefEff        1   125.31 137.31
## - OffEff        1   125.57 137.57
## <none>              124.33 138.33
##
## Step:  AIC=136.33
## WinningTeam ~ OppPts + Pace + OffEff + DefEff + FastBreakPts
##
##                Df Deviance    AIC
## - OppPts        1   124.39 134.39
## - Pace          1   124.70 134.70
## - FastBreakPts  1   124.93 134.93
## <none>              124.33 136.33
## - DefEff        1   130.80 140.80
## - OffEff        1   257.18 267.18
```

```
##
## Step:  AIC=134.39
## WinningTeam ~ Pace + OffEff + DefEff + FastBreakPts
##
##                 Df Deviance    AIC
## - FastBreakPts  1   124.94 132.94
## <none>              124.39 134.39
## - Pace          1   129.62 137.62
## - DefEff        1   239.30 247.30
## - OffEff        1   264.84 272.84
##
## Step:  AIC=132.93
## WinningTeam ~ Pace + OffEff + DefEff
##
##           Df Deviance    AIC
## <none>        124.94 132.94
## - Pace    1   129.65 135.65
## - DefEff  1   239.35 245.35
## - OffEff  1   271.29 277.29
```

```r
summary(step_all_preds)
```

```
##
## Call:
## glm(formula = WinningTeam ~ Pace + OffEff + DefEff, family = binomial(),
##     data = nbaData)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.97363  -0.31904  -0.01972   0.35267   2.24921
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -19.02453    9.54463  -1.993   0.0462 *
## Pace          0.12822    0.06207   2.066   0.0389 *
## OffEff        0.81706    0.12708   6.429 1.28e-10 ***
## DefEff       -0.75702    0.12008  -6.304 2.89e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 294.40  on 213  degrees of freedom
## Residual deviance: 124.93  on 210  degrees of freedom
## AIC: 132.93
##
## Number of Fisher Scoring iterations: 7
```

```r
nbaData$preds = ifelse(step_all_preds$fitted.values > 0.5, 1, 0)

# look at confusion matrix
cm = confusionMatrix(as_factor(nbaData$preds), as_factor(nbaData$WinningTeam), positive = "1")
cm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 104  16
##          1  14  80
##
##                Accuracy : 0.8598
##                  95% CI : (0.806, 0.9034)
##     No Information Rate : 0.5514
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.7161
##
##  Mcnemar's Test P-Value : 0.8551
##
##             Sensitivity : 0.8333
##             Specificity : 0.8814
##          Pos Pred Value : 0.8511
##          Neg Pred Value : 0.8667
##              Prevalence : 0.4486
##          Detection Rate : 0.3738
##    Detection Prevalence : 0.4393
##       Balanced Accuracy : 0.8573
##
##        'Positive' Class : 1
##
```