ModelDefinition

Brian Hallberg

June 27, 2018

Load Data

```
eligible = readRDS("eligible.rds")
```

Define functions

```
myconfusion <- function(x, y) {
  tmp <- table(x, y)
  Accuracy <- (tmp[1,1]+tmp[2,2])/sum(tmp)
  sensitivity <- tmp[2,2]/(tmp[2,2]+tmp[2,1])
  specificity <- tmp[1,1]/(tmp[1,1]+tmp[1,2])
  result <- list(Accuracy=Accuracy, sensitivity=sensitivity, specificity=specificity)
  return(result)
}
```

Add any needed columns to the table

The column *inducted* either has the value "Y" or "N". To process the regression we need to create another column that has the value 1 if *inducted* is "Y" or 0 if it is "N".

```
eligible$Indicator <- ifelse(eligible$inducted == "Y", 1, 0)
```

Start with simply Logistic Regression

Looking at all players and their All-Star game appearances only

Using the full data set of eligible players. Let's look at all-star appearances only in regression and see what we get.

Make the trial and test sets of data

```
split <- sample.split(eligible$Indicator, SplitRatio = 0.7)
Train <- subset(eligible, split == TRUE)
Test <- subset(eligible, split == FALSE)
fit.allstar <- glm(Indicator ~ allstar, data = Train, family = "binomial")
summary(fit.allstar)
```

```
##
## Call:
## glm(formula = Indicator ~ allstar, family = "binomial", data = Train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3493  -0.3323  -0.2804  -0.2804   2.5518
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.21658    0.13081  -24.59   <2e-16 ***
## allstar      0.34770    0.02699   12.88   <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: 999.79  on 1669  degrees of freedom
## Residual deviance: 791.75  on 1668  degrees of freedom
## AIC: 795.75
##
## Number of Fisher Scoring iterations: 5
```

Now we'll fit the test data with the model and compute accuracy

```
predTest <- predict(fit.allstar, newdata = Test)
table(Test$Indicator, predTest > 0.5)
```

```
##
##     FALSE TRUE
##   0   648    4
##   1    50   13
```

```
myconfusion(Test$Indicator, predTest > 0.5)
```

```
## $Accuracy
## [1] 0.9244755
##
## $sensitivity
## [1] 0.2063492
##
## $specificity
## [1] 0.993865
```

The accuracy is .92. However it does not do a good job of predicting the true hall of fame players from the test set. Only correct on .14 of the cases.

If we change the threshold to 0.01, the true hall of famers from all star appearances only gets to .23. The accuracy is .93.

Add all the other predictors and see what happens

```
fit.all <- glm(Indicator ~ allstar + hit + hr + rbi + run + ba + w + gpitch + sopitch, data = Train, fa
summary(fit.all)
```

```
##
## Call:
## glm(formula = Indicator ~ allstar + hit + hr + rbi + run + ba +
##     w + gpitch + sopitch, family = "binomial", data = Train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8380  -0.1949  -0.0927  -0.0446   3.7714
```

```
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -8.9654202  1.2951842  -6.922 4.45e-12 ***
## allstar      0.3559446  0.0946055   3.762 0.000168 ***
## hit         -0.0002931  0.0015436  -0.190 0.849418
## hr          -0.0221688  0.0074899  -2.960 0.003078 **
## rbi          0.0068956  0.0026820   2.571 0.010137 *
## run          0.0024912  0.0020764   1.200 0.230226
## ba           0.8482256  4.9984541   0.170 0.865248
## w            0.0259435  0.0049304   5.262 1.43e-07 ***
## gpitch       0.0020219  0.0014507   1.394 0.163401
## sopitch     -0.0001969  0.0004949  -0.398 0.690766
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 447.50  on 728  degrees of freedom
## Residual deviance: 183.23  on 719  degrees of freedom
##   (941 observations deleted due to missingness)
## AIC: 203.23
##
## Number of Fisher Scoring iterations: 7
```

Lots of extras, let's remove some and see how it changes

```
fit.all <- glm(Indicator ~ allstar + hit + hr + rbi + run + w + gpitch + sopitch, data = Train, family =
summary(fit.all)
```

```
##
## Call:
## glm(formula = Indicator ~ allstar + hit + hr + rbi + run + w +
##     gpitch + sopitch, family = "binomial", data = Train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8448  -0.1901  -0.0909  -0.0435   3.7663
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -8.8665768  0.9439630  -9.393  < 2e-16 ***
## allstar      0.3557390  0.0943925   3.769 0.000164 ***
## hit         -0.0002446  0.0015309  -0.160 0.873042
## hr          -0.0222751  0.0074777  -2.979 0.002893 **
## rbi          0.0069665  0.0026839   2.596 0.009441 **
## run          0.0024852  0.0020825   1.193 0.232737
## w            0.0263051  0.0047161   5.578 2.44e-08 ***
## gpitch       0.0020211  0.0014342   1.409 0.158765
## sopitch     -0.0002250  0.0004733  -0.475 0.634483
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
##     Null deviance: 450.36  on 743  degrees of freedom
## Residual deviance: 183.61  on 735  degrees of freedom
##   (926 observations deleted due to missingness)
## AIC: 201.61
##
## Number of Fisher Scoring iterations: 7
```

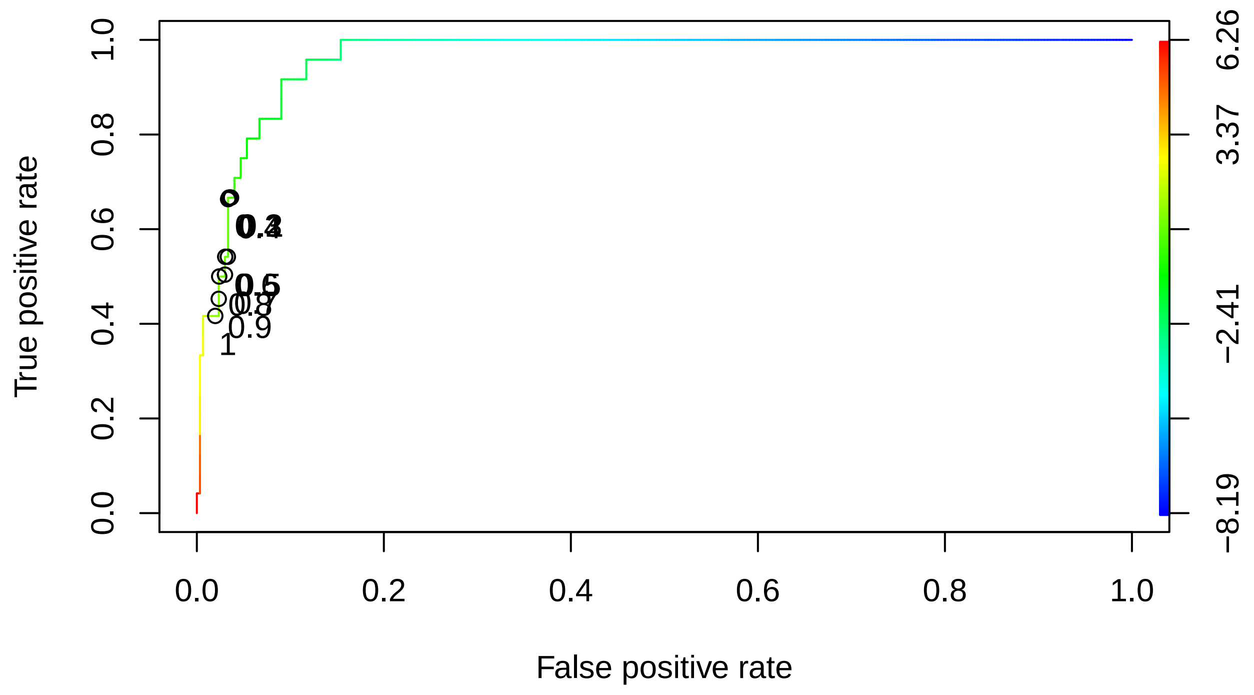
Drop another

```
fit.all <- glm(Indicator ~ allstar + hr + rbi + w, data = Train, family = "binomial")
summary(fit.all)
```

```
##
## Call:
## glm(formula = Indicator ~ allstar + hr + rbi + w, family = "binomial",
##     data = Train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7357  -0.2044  -0.0898  -0.0441   3.6635
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -8.186093   0.776280 -10.545  < 2e-16 ***
## allstar      0.346588   0.076767   4.515 6.34e-06 ***
## hr          -0.023659   0.006242  -3.790 0.000151 ***
## rbi          0.008880   0.001109   8.003 1.21e-15 ***
## w            0.026716   0.003444   7.757 8.68e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 450.36  on 743  degrees of freedom
## Residual deviance: 187.70  on 739  degrees of freedom
##   (926 observations deleted due to missingness)
## AIC: 197.7
##
## Number of Fisher Scoring iterations: 7
```

Let's use some ploting to see what the threshold should be

```
predTest <- predict(fit.all, newdata = Test)
ROCRpred <- prediction(predTest, Test$Indicator)
ROCRperf <- performance(ROCRpred, "tpr", "fpr")
plot(ROCRperf, colorize=TRUE, print.cutoffs.at=seq(0, 1, 0.1), text.adj=c(-0.2, 1.7))
```



```r
table(Test$Indicator, predTest > 0.5)
```

```
##
##      FALSE TRUE
##   0    290    9
##   1     11   13
```

```r
myconfusion(Test$Indicator, predTest > 0.5)
```

```
## $Accuracy
## [1] 0.9380805
##
## $sensitivity
## [1] 0.5416667
##
## $specificity
## [1] 0.9698997
```

Not this does very good with those not inducted, but we miss most of those that were inducted

So let's split the data between pitchers and nonpitchers and see what we can learn.

Break the eligible into pitchers and nonpitchers

```r
pitchelig <- eligible %>% filter(!is.na(gpitch)) %>% filter(gpitch>163)
nonpitchelig <- eligible %>% filter(gpitch<164 | is.na(gpitch))
```

fit again, make train and test sets for both

```r
split = sample.split(pitchelig$Indicator, SplitRatio = 0.7)
pTrain = subset(pitchelig, split == TRUE)
pTest = subset(pitchelig, split == FALSE)

split = sample.split(nonpitchelig$Indicator, SplitRatio = 0.7)
npTrain = subset(nonpitchelig, split == TRUE)
npTest = subset(nonpitchelig, split == FALSE)

pfit.all <- glm(Indicator ~ allstar + w, data = pTrain, family = "binomial")
summary(pfit.all)
```

```
##
## Call:
## glm(formula = Indicator ~ allstar + w, family = "binomial", data = pTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8578  -0.1898  -0.0940  -0.0494   3.6433
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -8.235841   0.862583  -9.548  < 2e-16 ***
## allstar      0.403959   0.086270   4.682 2.83e-06 ***
## w            0.028297   0.003812   7.423 1.14e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 326.80  on 582  degrees of freedom
## Residual deviance: 133.53  on 580  degrees of freedom
## AIC: 139.53
##
## Number of Fisher Scoring iterations: 7
```

```r
npfit.all <- glm(Indicator ~ allstar + hr + rbi + run + ba, data = npTrain, family = "binomial")
summary(npfit.all)
```

```
##
## Call:
## glm(formula = Indicator ~ allstar + hr + rbi + run + ba, family = "binomial",
##     data = npTrain)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.30589  -0.17046  -0.06065  -0.02134   3.03093
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.865e+01  2.667e+00  -6.995 2.66e-12 ***
## allstar      3.239e-01  5.349e-02   6.055 1.40e-09 ***
## hr          -1.216e-02  2.360e-03  -5.152 2.58e-07 ***
## rbi          4.171e-03  1.048e-03   3.979 6.91e-05 ***
## run          3.106e-03  7.485e-04   4.149 3.34e-05 ***
## ba           3.600e+01  9.469e+00   3.802 0.000144 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 672.08  on 1085  degrees of freedom
## Residual deviance: 252.10  on 1080  degrees of freedom
##   (1 observation deleted due to missingness)
## AIC: 264.1
##
## Number of Fisher Scoring iterations: 8
```

How is the prediction on the new sets

```
ppredTest <- predict(pfit.all, newdata=pTest)
table(pTest$Indicator, ppredTest > 0.5)
```

```
##
##     FALSE TRUE
##   0   225    4
##   1    10   10
```

```
myconfusion(pTest$Indicator, ppredTest > 0.5)
```

```
## $Accuracy
## [1] 0.9437751
##
## $sensitivity
## [1] 0.5
##
## $specificity
## [1] 0.9825328
```

```
nppredTest <- predict(npfit.all, newdata=npTest)
table(npTest$Indicator, nppredTest > 0.5)
```

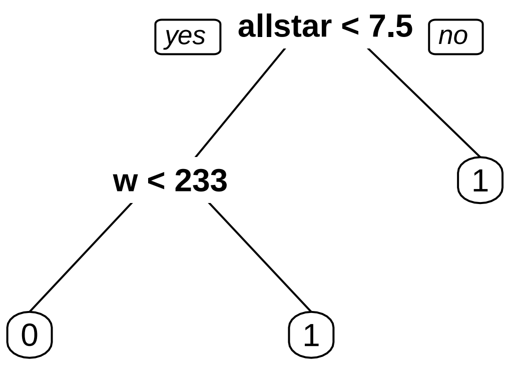
```
##
##     FALSE TRUE
##   0   419    4
##   1    23   20
```

```
myconfusion(npTest$Indicator, nppredTest > 0.5)
```

```
## $Accuracy
## [1] 0.9420601
##
## $sensitivity
## [1] 0.4651163
##
## $specificity
## [1] 0.9905437
```

Look at CART instead

```
hofTree <- rpart(Indicator ~ allstar + hr + rbi + w, data = Train, method="class", control=rpart.control
prp(hofTree)
```



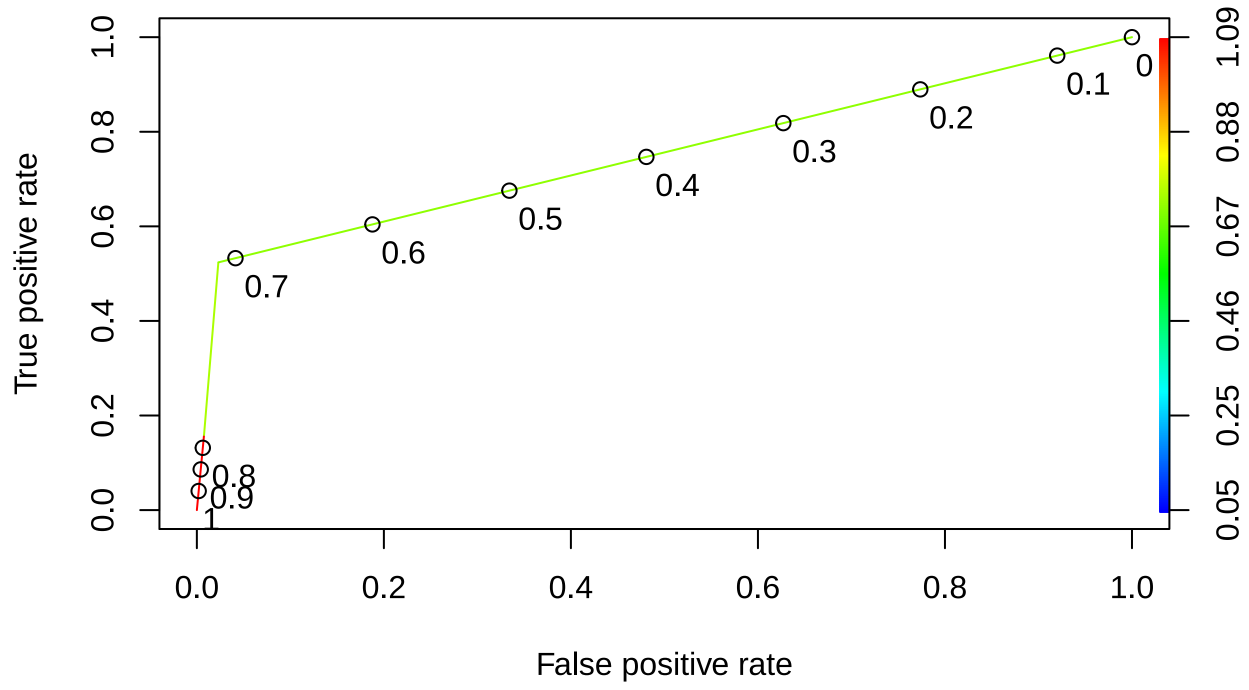
```
PredCART <- predict(hofTree, newdata=Test, type="class")
table(Test$Indicator, PredCART)
```

```
##    PredCART
##       0   1
##   0 637  15
##   1  30  33
```

```
myconfusion(Test$Indicator, PredCART)
```

```
## $Accuracy
## [1] 0.9370629
##
## $sensitivity
## [1] 0.5238095
##
## $specificity
## [1] 0.9769939
```

```
PredROC <- predict(hofTree, newdata=Test)
pred <- prediction(PredROC[,2], Test$Indicator)
perf <- performance(pred, "tpr", "fpr")
plot(perf, colorize=TRUE, print.cutoffs.at=seq(0, 1, 0.1), text.adj=c(-0.2, 1.7))
```



Look at Random Forest instead

```
Train$Indicator <- as.factor(Train$Indicator)
Test$Indicator <- as.factor(Test$Indicator)
allForest <- randomForest(Indicator ~ allstar + hr + rbi + w, data = Train, nodesize=25, ntree=500, na.a
predForest <- predict(allForest, newdata = Test)
table(Test$Indicator, predForest)
```

```
##    predForest
##       0   1
##   0 288  11
##   1   7  17
```

```
myconfusion(Test$Indicator, predForest)
```

```
## $Accuracy
## [1] 0.9442724
##
## $sensitivity
## [1] 0.7083333
##
## $specificity
## [1] 0.9632107
```