Math 536 Final

Jonathan Brinkerhoff

2023-05-18

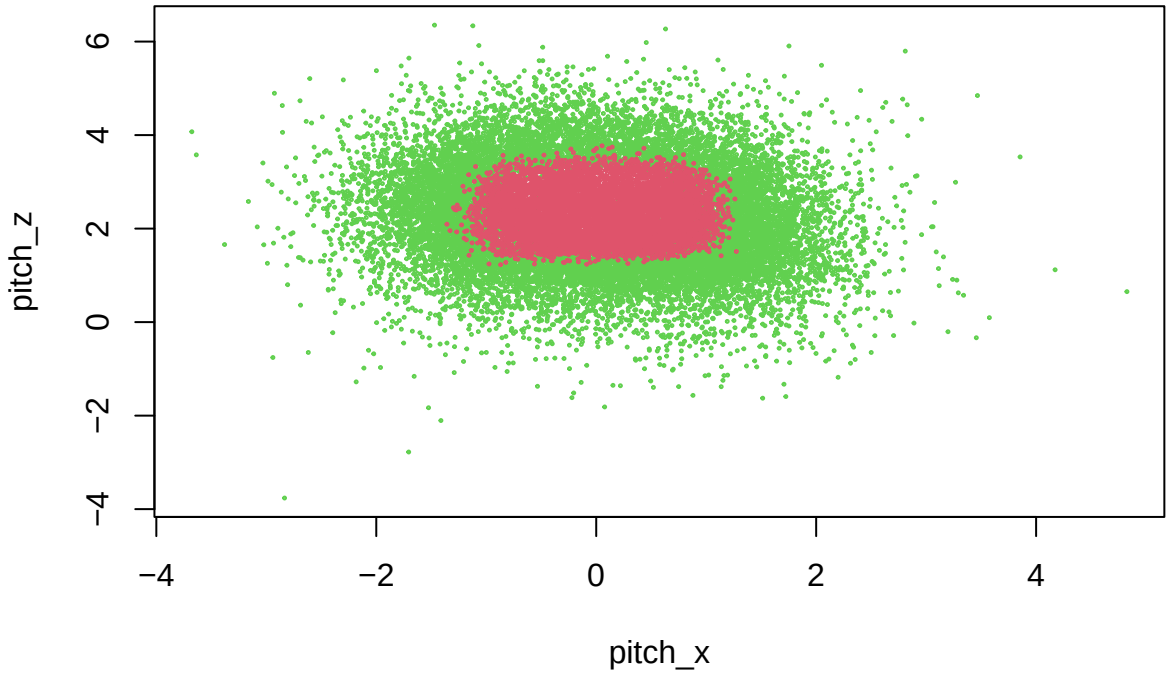
Overview

I have some pitching data from the 2018-2019 MLB season, specifically pitches that were not swung at. The data includes several predictors, including pitch type, pitch position, release speed, balls, strikes, etc., and a single response variable, whether the pitch is called a strike or a ball. I am interested in exploring whether high-heat fast balls, that is, fast balls pitched above the strike zone, are more likely to be called strikes than other pitches similarly outside the strike zone. Pitch type and pitch position are the most important predictors in this scenario.

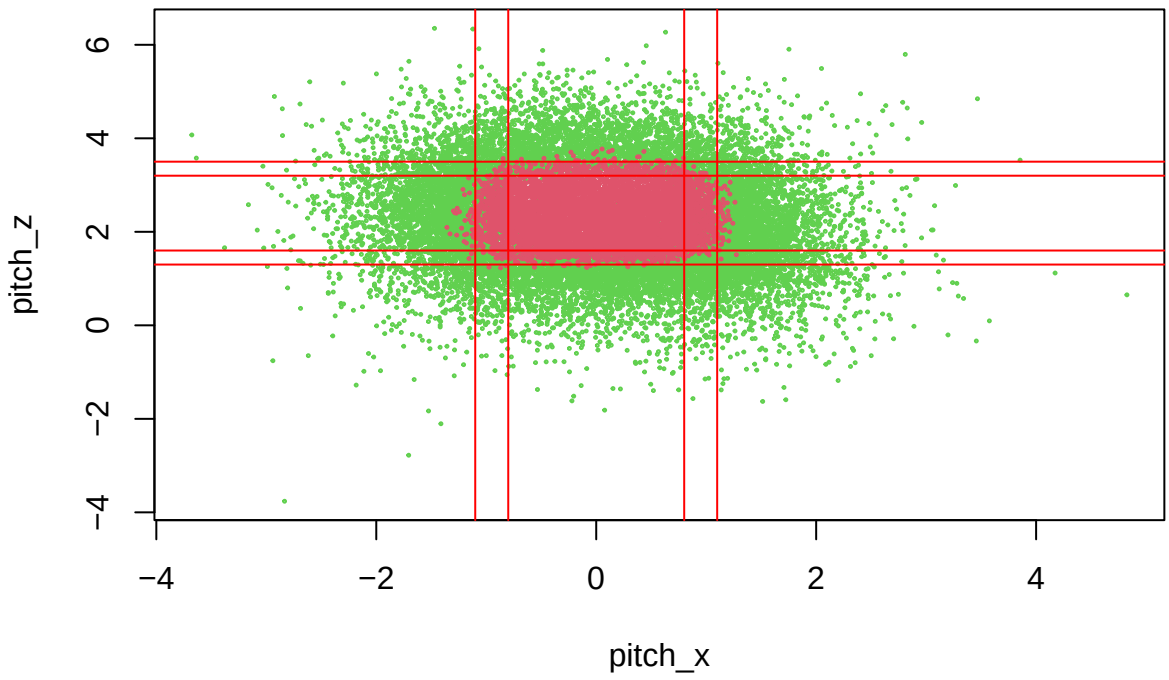
Fitting the data to a random forest model yields an interesting result: high-heat pitches are significantly less-likely to be called strikes than other pitches in the fringe of the strike zone. The average probability that a high-heat pitch will be called a strike is about 29.93%, and by bootstrapping I can say with 95% confidence that the probability of a high-heat pitch being called a strike is between about 29.37% and 30.61%. The average probability that a fringe pitch that is not high-heat is called a strike is about 42.42%, and I can say with 95% confidence that the probability of such a pitch being called a strike is between about 41.37% and 43.57%. There are many possible explanations for this outcome, I propose that umpires may be more vigilant with respect to high fast balls due to the prevailing notion that such pitches are more difficult to call accurately.

Analysis

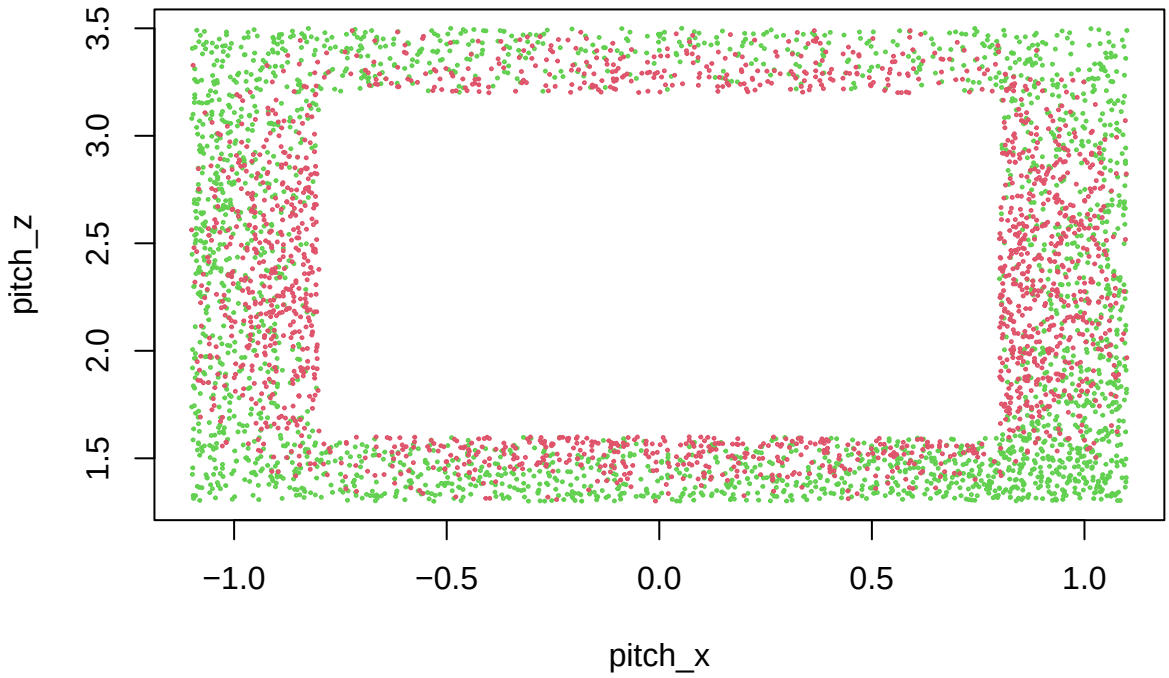
Following is a plot of balls (in green) and strikes (in red).



I am only interested in pitches around the fringe of the strike zone, where green and red mix. Obvious strikes well within the strike zone, and obvious balls well without, do not help to answer my question. I draw some lines to frame the area of interest.



Now truncating the data to only include fringe pitches leaves me with a smaller data set that I will use to answer my question. The truncated data looks as follows.



There are 4371 pitches in this fringe region, 2575 of which are balls, and 1796 of which are strikes.

I will investigate two models to determine which better fits my data, logistic regression and random forest. Following is the output from fitting my truncated data to a logistic regression model.

```
## Warning: package 'randomForest' was built under R version 4.2.3
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Call:
## glm(formula = description ~ pitch_type + release_speed + balls +
##     strikes + plate_x + plate_z + home_score + away_score, family = "binomial",
##     data = trunc_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4389  -1.0442  -0.8417   1.2464   1.9343
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)     2.149060   0.893044   2.406  0.01611 *
## pitch_typeCU   -0.232098   0.177538  -1.307  0.19111
## pitch_typeEP   12.324831 226.677994   0.054  0.95664
```

```
## pitch_typeFC    0.009858   0.196922   0.050  0.96008
## pitch_typeFF    0.479824   0.164033   2.925  0.00344 **
## pitch_typeFS    0.002455   0.296683   0.008  0.99340
## pitch_typeFT    0.537647   0.172112   3.124  0.00179 **
## pitch_typeKC   -0.103876   0.228133  -0.455  0.64887
## pitch_typeSI    0.238741   0.182972   1.305  0.19196
## pitch_typeSL    0.280941   0.145485   1.931  0.05348 .
## release_speed  -0.034148   0.010603  -3.221  0.00128 **
## balls           0.064143   0.035952   1.784  0.07440 .
## strikes        -0.384760   0.046898  -8.204 2.32e-16 ***
## plate_x         0.051142   0.039789   1.285  0.19867
## plate_z         0.196329   0.042418   4.628 3.68e-06 ***
## home_score     -0.030720   0.013530  -2.271  0.02318 *
## away_score     -0.001992   0.013123  -0.152  0.87934
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5919.9  on 4370  degrees of freedom
## Residual deviance: 5779.1  on 4354  degrees of freedom
## AIC: 5813.1
##
## Number of Fisher Scoring iterations: 11
```

There are some odd results from this model. For example, I know that whether a pitch is considered a strike is strongly correlated with the position of the pitch, but that fact is not really reflected by the logistic regression model.

I will also fit the data to a random forest model, and compare the two models objectively. I can do this by evaluating the log-likelihood $\sum_{i=1}^{n} y_i ln(\hat{p}_i) + (1 - y_i) ln(1 - \hat{p}_i)$ for predictions from each model. A larger value indicates better fit. The value of the log-likelihood for the logistic regression model is:

```
## [1] -2889.547
```

And the value of the log-likelihood for the random forest model is:

```
## [1] -679.3283
```

Clearly the random forest fits the data better, So that is the model that I will use to answer my research question, whether high-heat pitches are more likely to be called strikes than other pitches similarly outside the strike zone.

To answer this question, I'll create a new model that replaces the predictors pitch_type and release_speed with a single predictor called high_heat. High_heat is an indicator that a pitch is a fast ball in the upper range of the fringe zone. Next I'll create two data sets, one that treats all pitches as high-heat pitches, and another that treats no pitches as high-heat. I'll fit these data sets to the new model, and see if there is a difference in the proportion of strikes called.

The average proportion of strikes called on high heat pitches is:

```
## [1] 0.3209792
```

And the avereage proportion of strikes called on fringe pitches that are not high heat is:

```
## [1] 0.4242462
```

Which is an interesting result, as it contradicts conventional wisdom. Apparently, high-heat pitches are less likely to be called strikes than other pitches similarly outside the strike zone.

I can produce some 95% confidence intervals via boostrapping. The 95% confidence interval for the proportion of high-heat pitches called strikes is:

```
## ( 0.3141295 , 0.3252757 )
```

And the 95% confidence interval for the proportion of pitches other than high-heat pitches called strikes is:

```
## ( 0.4161107 , 0.4318302 )
```

Please note that these values will not match what I report on the overview page, as new values are produced when I knit a new document.

I should mention a concern I have about the random forest model. Using ten-fold cross validation and evaluating the log likelihood as before, The random forest model performs significantly worse than it does with the initial data. This is an indication that the random forest model was overfitting the data. Following is the cross validated log-likelihood for the logistic regression model, which performs about as poorly as it did initially.

```
## [1] -2903.909
```

And the cross validated log-likelihood for the random forest model, which performs significantly worse than before.

```
## [1] -2463.027
```

Appendix

Follwing is the code I used for this project

```
#plot balls and strikes
plot(data$plate_x[data$description=="ball"],data$plate_z[data$description=="ball"],col=3,cex=.2)
points(data$plate_x[data$description!="ball"],data$plate_z[data$description!="ball"],col=2,cex=.2)

#frame the fringe zone
#upper range
abline(h = 3.2, col="red")
abline(h = 3.5, col="red")
#lower range
abline(h = 1.6,col="red")
abline(h = 1.3,col="red")
#right range
abline(v=.8,col="red")
abline(v=1.1,col="red")
#left range
abline(v=-0.8,col="red")
abline(v=-1.1,col="red")

#truncate data to only include data in the fringe zone
trunc_data = subset(data, (data$plate_z > 3.2 & data$plate_z < 3.5 & data$plate_x > -1.1  & data$plate_
                        | (data$plate_z > 1.3 & data$plate_z < 1.6 & data$plate_x > -1.1  & data$plate_
                        | (data$plate_x > -1.1 & data$plate_x < -.8 & data$plate_z > 1.6  & data$plate_
                        | (data$plate_x > .8 & data$plate_x < 1.1 & data$plate_z > 1.6  & data$plate_z

#plot truncated balls and strikes
plot(trunc_data$plate_x[trunc_data$description=="ball"],trunc_data$plate_z[trunc_data$description=="ball
points(trunc_data$plate_x[trunc_data$description!="ball"],trunc_data$plate_z[trunc_data$description!="ba

#try a couple models. glm and random forest
attach(trunc_data)
library(randomForest)

trunc_data$description = as.factor(trunc_data$description)#Why is this necessary?

model.a = glm(description~pitch_type+
                release_speed+
                balls+
                strikes+
                plate_x+
                plate_z+
                home_score+
                away_score,
                data=trunc_data,family='binomial')

model.b = randomForest(description~pitch_type+
                release_speed+
                balls+
                strikes+
                plate_x+
                plate_z+
```

```
                home_score+
                away_score,
                data=trunc_data,ntree=100,mtry=3,
                control=rpart.control(minsplit=5,cp=.01))

#compute a log likelihood for each model and then compare them.
y = as.numeric(trunc_data$description) - 1
p.a = predict.glm(model.a,newdata=trunc_data,type="response")
p.b = predict(model.b,newdata=trunc_data,type="prob")[,2] #why column 2?
#Probably don't want rounded probabilities to exactly 0 or 1
p.b[p.b==0] = 0.00001
p.b[p.b==1] = .9999

LL.a = sum(y*log(p.a) + (1-y)*log(1-p.a))
LL.b = sum(y*log(p.b) + (1-y)*log(1-p.b))
LL.a
LL.b

#add high_heat indicator to truncated data
trunc_data$high_heat = rep(0,length(trunc_data$description))
trunc_data$high_heat[trunc_data$pitch_type %in% c("FF","FT","FC","FS") & trunc_data$plate_z > 3.3] = 1

#new model removes pitch_type and release speed predictors
#replaces with high_heat predictor
model.c = randomForest(description~balls+
                         strikes+
                         plate_x+
                         plate_z+
                         home_score+
                         away_score+high_heat,
                         data=trunc_data,ntree=100,mtry=3,
                         control=rpart.control(minsplit=5,cp=.01))


trunc_data_a = trunc_data
trunc_data_b = trunc_data
trunc_data_a$high_heat = 1 #all pitches in a are high heat
trunc_data_b$high_heat = 0 #no pitches in b are high heat

avg.prob.hh = mean(predict(model.c,newdata=trunc_data_a,type="prob")[,2])
avg.prob.nhh = mean(predict(model.c,newdata=trunc_data_b,type="prob")[,2])

#bootstrapping
#this code takes time to process, so only run 100 times
bs.avg.prop.hh = rep(0,100)
bs.avg.prop.nhh = rep(0,100)
for(i in 1:100){

  row.index= sample(1:dim(trunc_data)[1],dim(trunc_data)[1],replace=T) #sample row indices with replace

  bs.trunc.data = data.frame(trunc_data[row.index,]) #bootstrapped data from sampled indices

  bs.trunc_data_a = bs.trunc.data
```

```
  bs.trunc_data_b = bs.trunc.data
  bs.trunc_data_a$high_heat = 1 #all pitches in a are high heat
  bs.trunc_data_b$high_heat = 0 #no pitches in b are high heat

  bs.avg.prop.hh[i] = mean(predict(model.c,newdata=bs.trunc_data_a,type="prob")[,2])
  bs.avg.prop.nhh[i] = mean(predict(model.c,newdata=bs.trunc_data_b,type="prob")[,2])
}

hh_lower_bound = sort(bs.avg.prop.hh)[2.5]
hh_upper_bound = sort(bs.avg.prop.hh)[97.5]

nhh_upper_bound = sort(bs.avg.prop.nhh)[2.5]
nhh_lower_bound = sort(bs.avg.prop.nhh)[97.5]

########################attempt cross-validation
#Adapted from "model comparison ..." code

n = dim(trunc_data)[1]
n

###with 4370 data points, going to have 437 data points in each testing set
###Obviously that means there is 1 data point that never gets partitioned
###as testing data.  But we can pick those up as their own testing group at the end if we want to.

CV.ind.mat = matrix(sample(1:n,4370,replace=F),nrow=10)

#Run same code as before, but each time, we're going to run our models using only train data
#and testing them using only testing data.

CVLL.a = 0
CVLL.b = 0

for(k in 1:10){
  train.data = trunc_data[(1:n)[-CV.ind.mat[k,]],]
  test.data = trunc_data[CV.ind.mat[k,],]

  model.a = glm(description~pitch_type+
                  release_speed+
                  balls+
                  strikes+
                  plate_x+
                  plate_z+
                  home_score+
                  away_score,
                  data=train.data,family='binomial')

  model.b = randomForest(description~pitch_type+
                  release_speed+
                  balls+
                  strikes+
                  plate_x+
                  plate_z+
                  home_score+
```

```
                  away_score,
                  data=train.data,ntree=100,mtry=3,
                  control=rpart.control(minsplit=5,cp=.01))



  y = (as.numeric(trunc_data$description) - 1)[CV.ind.mat[k,]]
  p.a = predict.glm(model.a,newdata=test.data,type="response")
  p.b = predict(model.b,newdata=test.data,type="prob")[,2]
  #Probably don't want rounded probabilities to exactly 0 or 1
  p.b[p.b==0] = 0.00001
  p.b[p.b==1] = .9999

  LL.a1 = sum(y*log(p.a) + (1-y)*log(1-p.a))
  LL.b1 = sum(y*log(p.b) + (1-y)*log(1-p.b))
  CVLL.a = CVLL.a + LL.a1
  CVLL.b = CVLL.b + LL.b1
}
```