Charles Hesketh

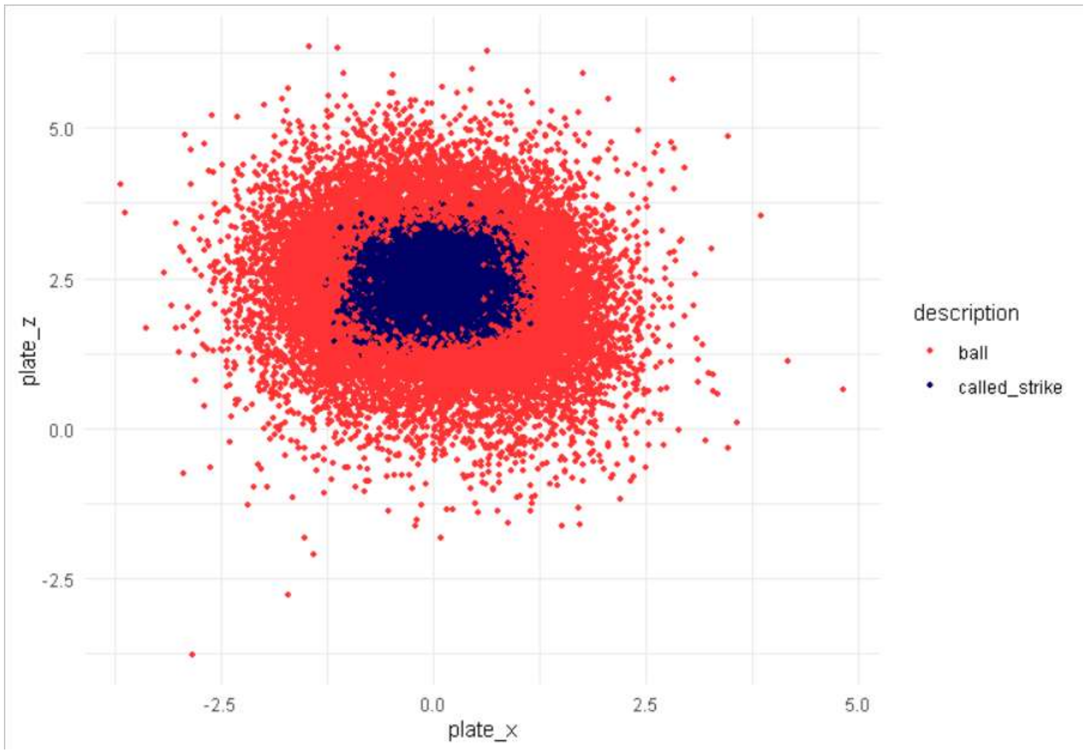
Final Exam

The Effects of Count on Umpire Bias

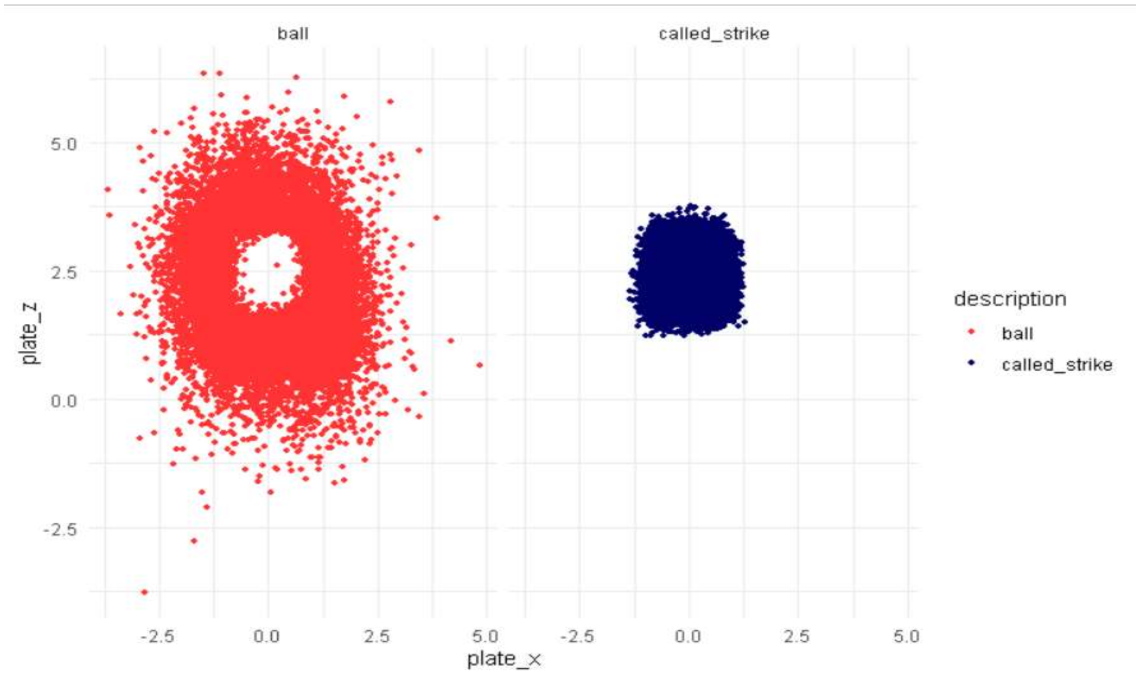
The purpose of this report is to assess if there is umpire bias toward calling strikes when the count is either “full”, “not full” (but 3 balls), “not3” (not having 3 balls). The report will show that the umpires within the Major League Baseball (MLB) do not show bias toward calling strikes based on current count.

Analyzing Raw Data

For this data we are only considering pitches that required the umpire to make a call, thus we are only interested in “called strikes” and “balls.” Since hit by pitch, swinging strikes, foul tips, etc. are not requiring the umpire to make a call we will not be considering these data points. Thus, our starting data looks as follows:



Plot 1



Plot 2

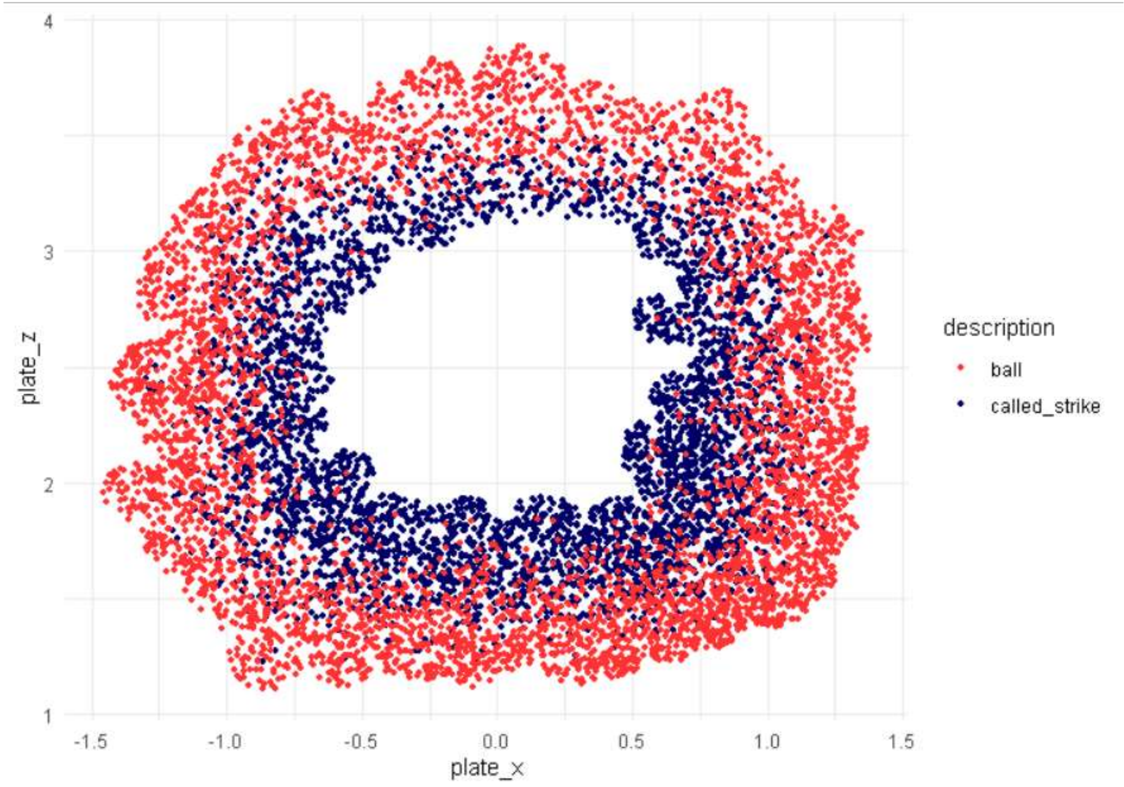
From here there are two very important things to address by looking at the above graph:

1) Problem 1: Pitches “down the middle” are almost certainly going to be a strike while pitches significantly outside the strike zone will almost certainly be called a ball.
2) Problem 2: There seems to be some outliers in the middle that need to be addressed as they were called balls when probably should have been strikes.

Cleaning Raw Data

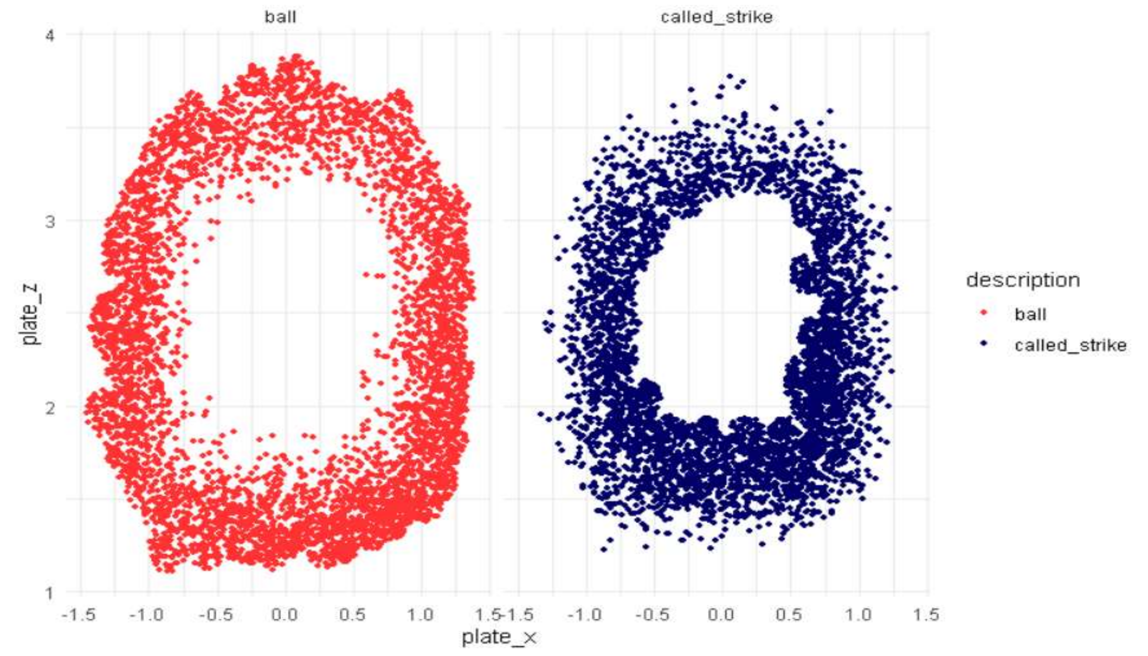
In order to address Problem 1, we need to isolate “fringe pitches”, or pitches that appear to be on or around the border of the strike zone. This will allow us to look at the important pitches to consider. In handling this problem we will handle Problem 2. This will allow us to properly test the bias of umpires using count.

In order to isolate the important pitches, I started by eliminating the outliers that have been labelled as ball by calculating the 2-dimensional distance between any two balls then removing any balls that were not within a reasonable distance from another ball. This cleaned up the center of the red circle in plot 2 so that there would no longer the outlier balls called. From here we calculated the distance of every strike to a ball and every strike too far away was discarded and the same was done for balls to strikes. This yielded the following updated graphs:



4
3
2
1
plate_z
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5
plate_x
description
ball
called_strike

Plot 3



ball
called_strike
4
3
2
1
plate_z
-1.5
-1.0
-0.5
0.0
0.5
1.0
1.5
plate_x
description
ball
called_strike

Plot 4

From here we can clearly see that the egregious ball calls that were toward the center have been reduced, leaving only appropriate called balls and we were able to isolate the fringe pitches. As a result, we can start doing our analysis.

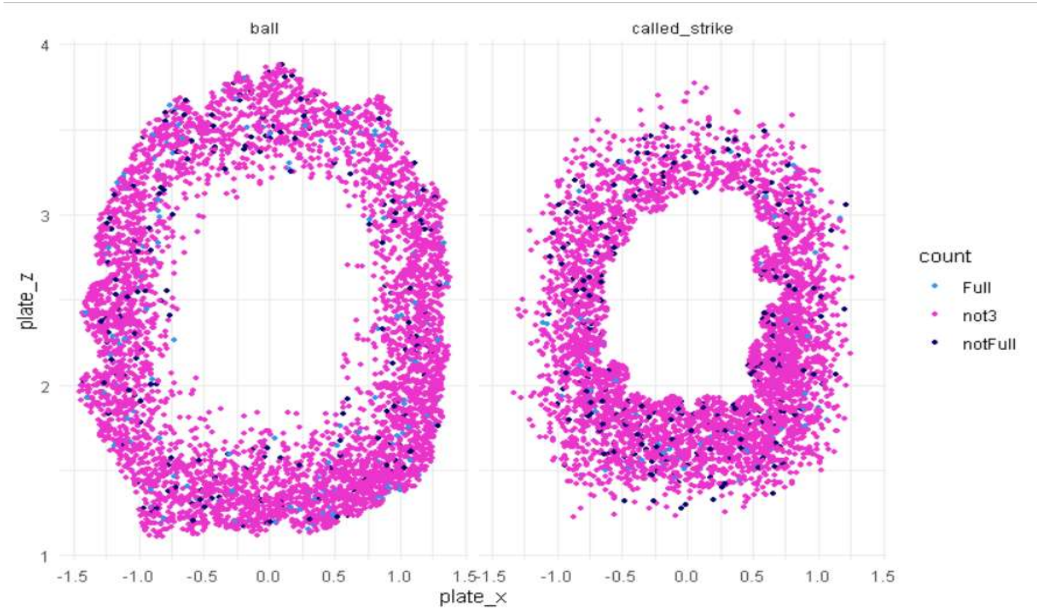
Process and Analysis

Analyzing plot 4 we see that as we progress from the inner circle to the outer circle our called strike plot progresses from denser to less dense while our ball graph shows the opposite. This is expected by the nature of balls and strikes and suggests our cleaning did not alter the fundamentals of the data. Adding a column for count labeled "full" for a full count, "notfull" for a count with balls equal to 3 and strikes less then 2, and "not3" to represent a count with less then 3 balls so that we may isolate which pitch was called based on these grouped counts. From here I implemented a random forest to isolate the importance of each predictor. This method was preferred as the spatial aspect of the data could present issues for general linear models or other potential models. This produced the following table:

Random Forest Variable Importance

Variable	Importance (More important> Less important)
plate_x	1685.84455
plate_z	1576.50340
count	43.11519
release_speed	479.63005
pitch_type	169.78766
home_score	209.53814
away_score	227.12282

From here we can see that position (plate_x and plate_z) show much more importance than any other variable, while our count variable is the least important. This indicates that count is not statistically important to an MLB umpires' decisions to call strikes. We can see this visualized further here:



I followed the random forest analysis up with a boosting model for the random forest, this led to a similar result further verifying the random forest results (the change in number size is not important, but the magnitude difference between variables is:

Boosting Variable Importance

Variable	Importance (More important> Less important)
plate_x	50.18589337
plate_z	49.13265728
count	0.01505900
release_speed	0.17975853
pitch_type	0.45773089
home_score	0.02890093
away_score	0.00000000

Next Steps to Complete Analysis

In order to provide the analysis within the required time frame analysis outside of requested research was not preformed, but the following are some steps that will be explored and reported on when complete.

1) Performing a cross validation on the current model to ensure best results and numbers have been achieved through random forest and boosting analysis. Increased data may increase the accuracy of this model and should be explored
2) In creating the model balls and strikes were removed as they were captured in the count variable; however, when exploring these variables, it suggests they may some impact overall. I have hypothesized that this is more of a reflection on pitch location selection. A count that is 0 balls and 2 strikes would allow a pitcher to throw a curveball in the dirt with hopes the batter will swing. This would need to be investigated throughout the data, not just fringe pitches.

Appendix: Code

```
library(ggplot2)

library(plot3D)

library(dplyr)

library(tidyr)


library(rpart)

library(adabag)

library(randomForest)

library(ROCR)

library(gbm)

#esquisse::esquisser(iPitch)


ballErr=.1

ballStrikeErr=.1

ballDistLim=0.13

strikeDistLim=0.1



setwd("C:\\Users\\User\\Desktop\\School\\Math_536\\Final")


pitches=read.csv("baseball.csv",stringsAsFactors=FALSE)


pitches$X=as.integer(pitches$X)

pitches$plate_x=as.double(pitches$plate_x)

pitches$plate_z=as.double(pitches$plate_z)


pitches=pitches[complete.cases(pitches),]
```

```
strike=pitches[pitches$description=="called_strike",]
ball=pitches[pitches$description=="ball",]

strikeDist=matrix(0,dim(strike)[1],2)
ballDist=matrix(0,dim(ball)[1],2)

ggplot(pitches) +
  aes(x = plate_x, y = plate_z, colour = description) +
  geom_point(size = 1L) +
  scale_color_manual(values=c("#FF3333","#000066")) +
  theme_minimal()



ggplot(pitches) +
  aes(x = plate_x, y = plate_z, colour = description) +
  geom_point(size = 1L) +
  scale_color_manual(values=c("#FF3333","#000066")) +
  theme_minimal() +
  facet_wrap(vars(description))

print(dim(ball))
for (j in 1:dim(ball)[1])
{
  ballTemp=cbind(ball$X,sqrt((ball$plate_x-ball$plate_x[j])^2+(ball$plate_z-ball$plate_z[j])^2))
  ballTemp=ballTemp[!(ballTemp[,2]==0),]
  if(min(ballTemp[,2])>ballErr)
  {
    ball=ball[!(ball$X==ball$X[j]),]
    j=j-1
  }
}

for (j in 1:dim(strike)[1])
{
  ballTemp=cbind(strike$X,sqrt((strike$plate_x-strike$plate_x[j])^2+(strike$plate_z-strike$plate_z[j])^2))
  ballTemp=ballTemp[!(ballTemp[,2]==0),]
  if(min(ballTemp[,2])>ballStrikeErr)
  {
    strike=strike[!(strike$X==strike$X[j]),]
    j=j-1
  }
}

for(i in 1:dim(strike)[1])
{
  ballTemp=cbind(ball$X,sqrt((ball$plate_x-strike$plate_x[i])^2+(ball$plate_z-strike$plate_z[i])^2))
  strikeDist[i,]=c(strike$X[i],min(abs(ballTemp[,2])))
}
strikeDist=strikeDist[!(strikeDist[,1]==0),]



for(i in 1:dim(ball)[1])
{
  ballTemp=cbind(strike$X,sqrt((strike$plate_x-ball$plate_x[i])^2+(strike$plate_z-ball$plate_z[i])^2))
  ballDist[i,]=c(ball$X[i],min(abs(ballTemp[,2])))
}
ballDist=ballDist[!(ballDist[,1]==0),]
```

```
strikeFinal=strikeDist[strikeDist[,2]<strikeDistLim,]

ballFinal=ballDist[ballDist[,2]<ballDistLim,]


iStrike=pitches[pitches$X %in% strikeFinal[,1],]

iBall=pitches[pitches$X %in% ballFinal[,1],]


iPitch=rbind(iStrike,iBall)




ggplot(iPitch) +
  aes(x = plate_x, y = plate_z, colour = description) +
  geom_point(size = 1L) +
  scale_color_manual(values=c("#FF3333","#000066")) +
  theme_minimal()

ggplot(iPitch) +
  aes(x = plate_x, y = plate_z, colour = description) +
  geom_point(size = 1L) +
  scale_color_manual(values=c("#FF3333","#000066")) +
  theme_minimal() +
  facet_wrap(vars(description))

ggplot(iPitch) +
  aes(x = plate_x, y = plate_z, colour = count) +
  geom_point(size = 1L) +
  scale_color_manual(values=c("#3495eb","#eb34cc","#000066")) +
  theme_minimal() +
  facet_wrap(vars(description))

iPitch$count=""

iPitch[iPitch$balls==3 & iPitch$strikes<2,]$count="notFull"
iPitch[iPitch$balls==3 & iPitch$strikes==2,]$count="Full"
iPitch[!(iPitch$balls==3),]$count="not3"

iPitch$description=as.factor(iPitch$description)
iPitch$pitch_type=as.factor(iPitch$pitch_type)
iPitch$count=as.factor(iPitch$count)
as.numeric(iPitch[1,])

model.rf =
randomForest(description~plate_x+plate_z+count+release_speed+pitch_type+home_score+away_score
,
             data=iPitch,ntree=100,mtry=2,control=rpart.control(minsplit=100,cp=.05))

model.rf$importance

prediction.rf = model.rf$predicted
table(prediction.rf,iPitch$description)



model.ab =
boosting(description~plate_x+plate_z+count+release_speed+pitch_type+home_score+away_score,
           data=iPitch,mfinal=100,control=rpart.control(minsplit=100,cp=.02))

model.ab$votes
```

```
prediction.ab = model.ab$class

table(prediction.ab,iPitch$description)


model.ab$importance
```