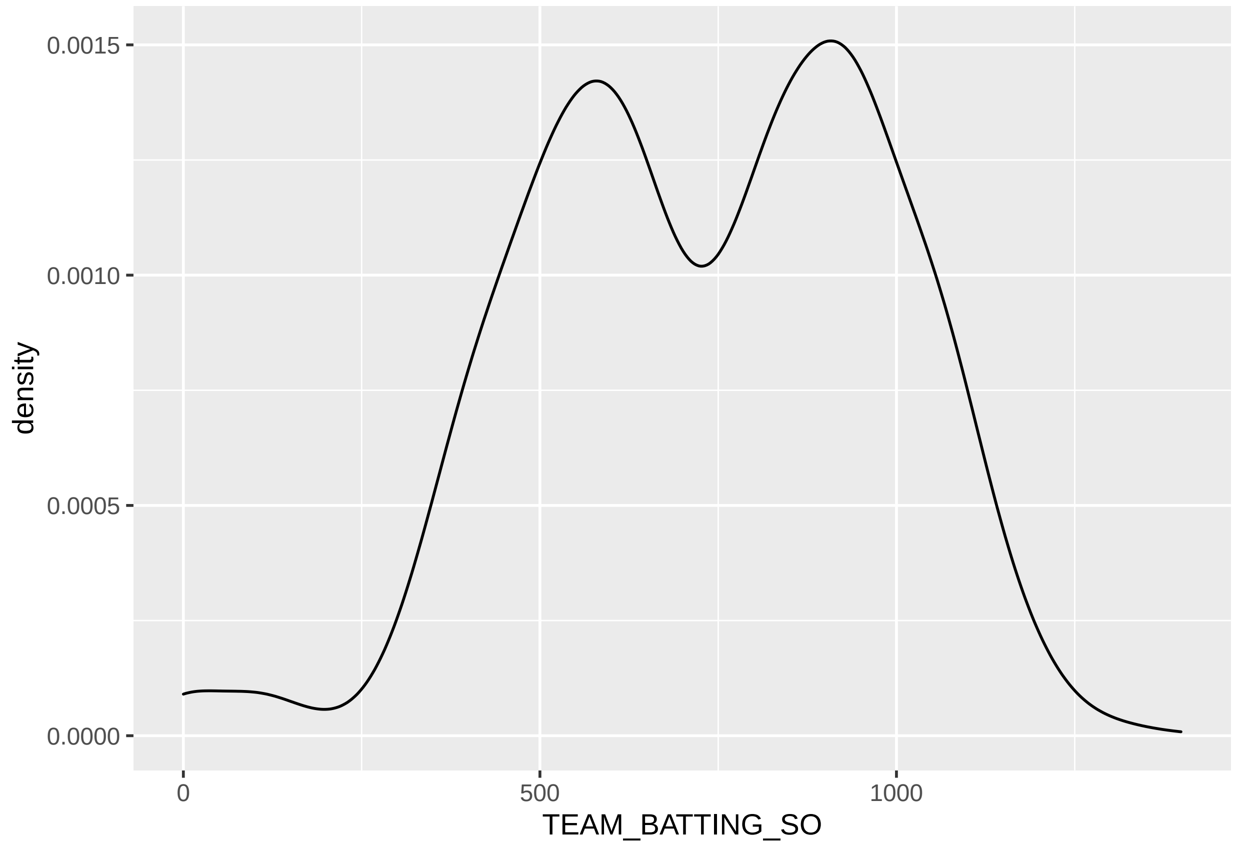
Disentangling the time periods of the Moneyball dataset

Bimodal distribution of strikeouts in `moneyball-training-data.csv`

I was looking at the variable `TEAM_BATTING_SO` which is the number of strikeouts committed by a team in a given season. I noticed it was bimodal

```
ggplot(dat, aes(TEAM_BATTING_SO))+geom_density()
```

```
## Warning: Removed 102 rows containing non-finite values (stat_density).
```



using the `mixtools` library i was able to fit normal distributions to these modes

```
library(mixtools)
```

```
## mixtools package, version 1.2.0, Released 2020-02-05
## This package is based upon work supported by the National Science Foundation under Grant No. SES-0518
```

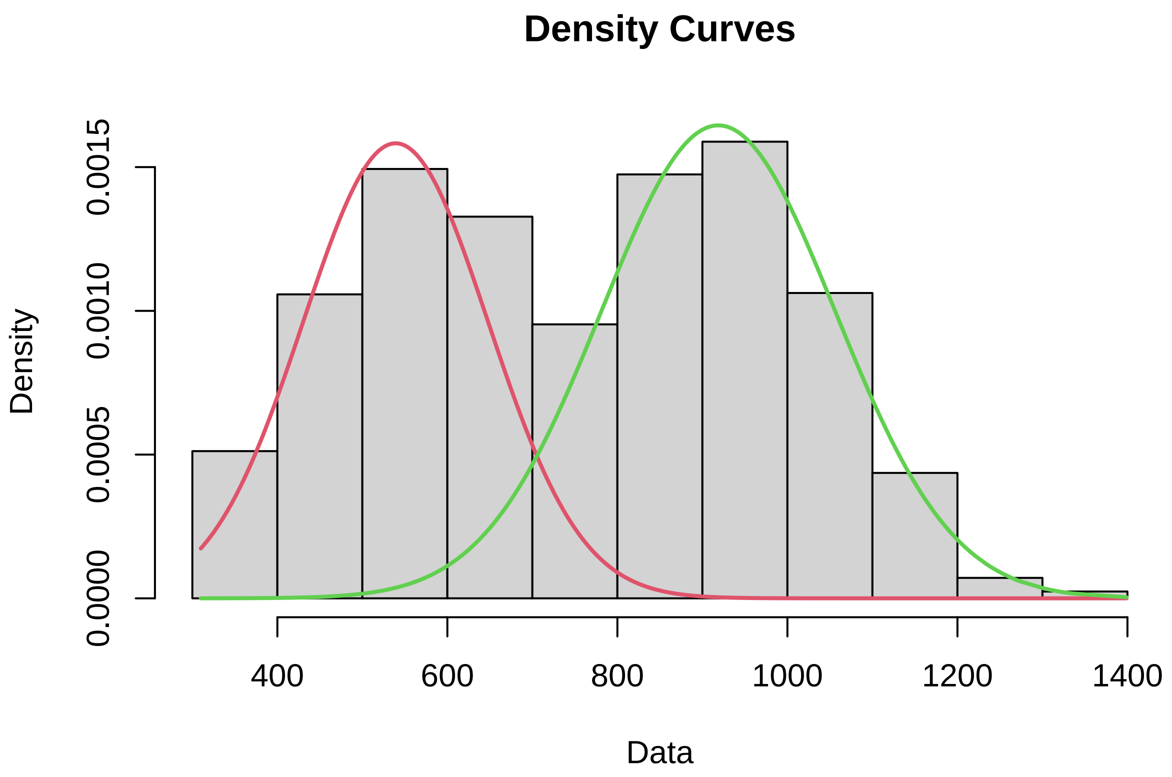
```
df<-dat%>%filter(TEAM_BATTING_SO>308 & !is.na(TEAM_BATTING_SO))
mD<-normalmixEM(df$TEAM_BATTING_SO)
```

```
## number of iterations= 113
```

```
summary(mD)
```

```
## summary of normalmixEM object:
##              comp 1     comp 2
## lambda   0.432332   0.567668
## mu     539.149544 918.666228
## sigma  108.967211 137.655246
## loglik at estimate:  -14279.25
```

```
plot(mD,which=2)
```



## Why this makes sense

Using some domain knowledge, there have been different "eras" in baseball in which teams decided to strike out more, in order to hit more home runs, and based upon the size of the data set, I would estimate that this is close to 100 years of data, so during this time the game has been played very differently.

Why this could be useful

If you look at the correlation between `TEAM_BATTING_SO` and `TARGET_WINS` they don't seem to be particularly useful, but if you segment the data into a `high` and `low` factor based upon which of these two modes they are most likely in, then you see a 10% increase in the correlation between a variable like `TEAM_BATTING_H` and the upper distribution and a 2% decrease in the correlation with the lower distribution.

```
df_factor<-df%>%
  mutate(SO_factor= case_when(
    TEAM_BATTING_SO<726 ~'low',
    TEAM_BATTING_SO>726 ~'high'
  ))

no_segment<-cor(df$TEAM_BATTING_H,df$TARGET_WINS)
df_high<-df_factor%>%filter(SO_factor=='high')
df_low<-df_factor%>%filter(SO_factor=='low')
segment_high<-cor(df_high$TEAM_BATTING_H, df_high$TARGET_WINS)
segment_low<-cor(df_low$TEAM_BATTING_H, df_low$TARGET_WINS)
data.frame('cor_with_no_segment'=no_segment,'cor_on_upper_dist'=segment_high, 'cor_on_lower_dist'=segmen
```

```
##   cor_with_no_segment cor_on_upper_dist cor_on_lower_dist
## 1           0.3886295         0.4298609         0.3798822
```

```
lm1<-lm(TARGET_WINS~TEAM_BATTING_H, data=df)
r1<-summary(lm1)$r.squared
lm2<-lm(TARGET_WINS~TEAM_BATTING_H, data=df_high)
r2<-summary(lm2)$r.squared
lm3<-lm(TARGET_WINS~TEAM_BATTING_H, data=df_low)
r3<-summary(lm3)$r.squared

data.frame('rSquare_with_no_segment'=r1,'rSquare_on_upper_dist'=r2, 'rSquare_on_lower_dist'=r3)
```

```
##   rSquare_with_no_segment rSquare_on_upper_dist rSquare_on_lower_dist
## 1               0.1510329             0.1847804             0.1443105
```

I believe if we could more successfuly disentangle these two modes of the distribution we could potentially feed our test data into two parallel models trained on the segmented data.

I have spoken to Daniel the project lead, and we agreed this would be a nice extra feature to have, but shouldn't be put before the unsegmented analysis. If anyone was interested on focusing on this particular feature with me, let me know.

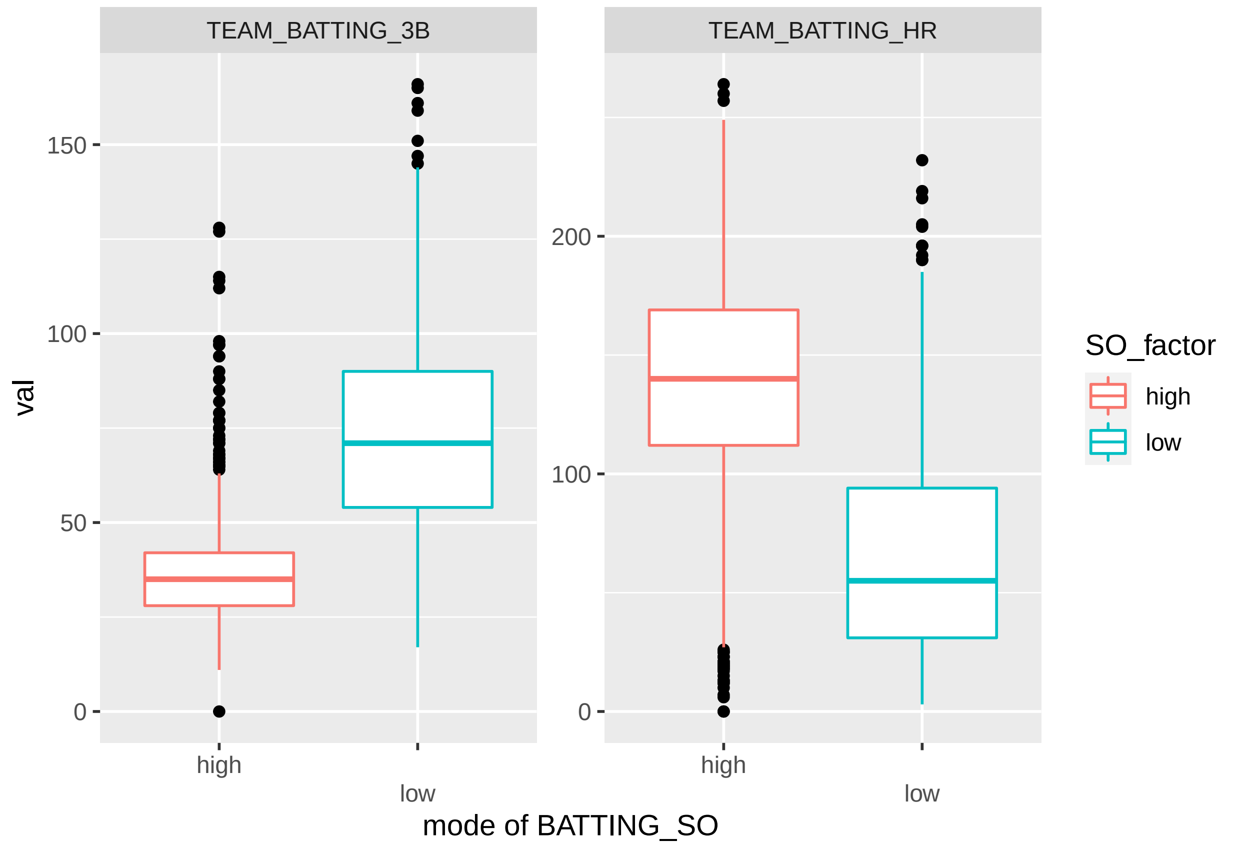
Next Steps

as you can tell from the density distributions of the modes above, there is a fair bit of overlap between the distributions and a single split based on the local minima is probably leaving some meat on the bone.

However if you segment the data, and then look at `TEAM_BATTING_HR` and `TEAM_BATTING_3B` you can see there are clear differences between these distributions

```
#split at 700


#plotting
```

```
dat_SO_2<-df_factor%>%filter(!is.na(SO_factor))
dat_SO_2_hit<-dat_SO_2%>%select(TEAM_BATTING_3B,TEAM_BATTING_HR,SO_factor)

#pivot table for plotting

#piv_hit_2<-dat_SO_2_hit%>% pivot_longer(cols =TARGET_WINS:TEAM_BATTING_HR, names_to = 'source',values_

ggplot(pivot_longer(dat_SO_2_hit, -SO_factor, names_to = 'var', values_to = 'val'), aes(SO_factor, val,
  geom_boxplot(outlier.color = 'black')+xlab('mode of BATTING_SO')+facet_wrap(~var, scales='free')+scal
```



I was wondering if we could create a conditional probability using the p-values to correctly factor the "high" and "low" strikeout modes to create more normal distributions.

My proposed plan in plain language(for my own sake):

1.

segment the data by `TEAM_BATTER_SO` to get initial groups

2.

isolate the data that overlaps between the two distributions

3.

look at the probability that a data point would fall into the IQR of one of the segmented `TEAM_BATTING_HR` groups above

4.

look at the probability that the same data point would fall into the IQR of one of the segmented `TEAM_BATTING_3B` groups above

5.

use this probability to recategorize it as in the 'high' or 'low' distribution.

Hopefully this would create two normally distributed data sets we could then do analysis on.