lab_08

2023-11-17

Preliminaries

```
## [1] -0.251  0.281
```

```
## [1] -0.625  0.626
```

Logs Regression

We perform a logistic regression here, with the intercept term and the `FT_PCT_DIFF` variable:

```
fit.glm <- glm(WON ~ FG_PCT_DIFF + FT_PCT_DIFF, data = nba2,
               family = binomial())

summary(fit.glm)
```

```
##
## Call:
## glm(formula = WON ~ FG_PCT_DIFF + FT_PCT_DIFF, family = binomial(),
##     data = nba2)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.001455   0.078197   0.019    0.985
## FG_PCT_DIFF 32.336946   1.869744  17.295  < 2e-16 ***
## FT_PCT_DIFF  3.685480   0.574464   6.416  1.4e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1704.8  on 1229  degrees of freedom
## Residual deviance: 1011.5  on 1227  degrees of freedom
## AIC: 1017.5
##
## Number of Fisher Scoring iterations: 5
```

Looking at the p-values, we see that the intercept term has a high p-value of 0.985, which is much greater than a 5% significance level. So, there is not enough evidence to reject the null hypothesis that the intercept is 0 when `FG_PCT_DIFF` and `FT_PCT_DIFF` have values of 0.

For the `FT_PCT_DIFF`, there is a very low p-value, of much less than 0.001, so there is enough evidence to reject the null hypothesis that the coefficient associated with `FT_PCT_DIFF` is 0.

So, we do not want to include the intercept, but we want to include `FT_PCT_DIFF`.

The appropriate equation is: chance of winning = 32.34(FG_PCT_DIFF) + 3.69(FT_PCT_DIFF), where the coefficients are rounded up to 2 d.p.

Classifying with Probability

```
p2class <- function(p, threshold) {
  ifelse(p >= threshold, 1, 0)
}
```

Here, we set up the grid and calculate the predicted classes using our grid:

```
range(nba2$FG_PCT_DIFF)
```

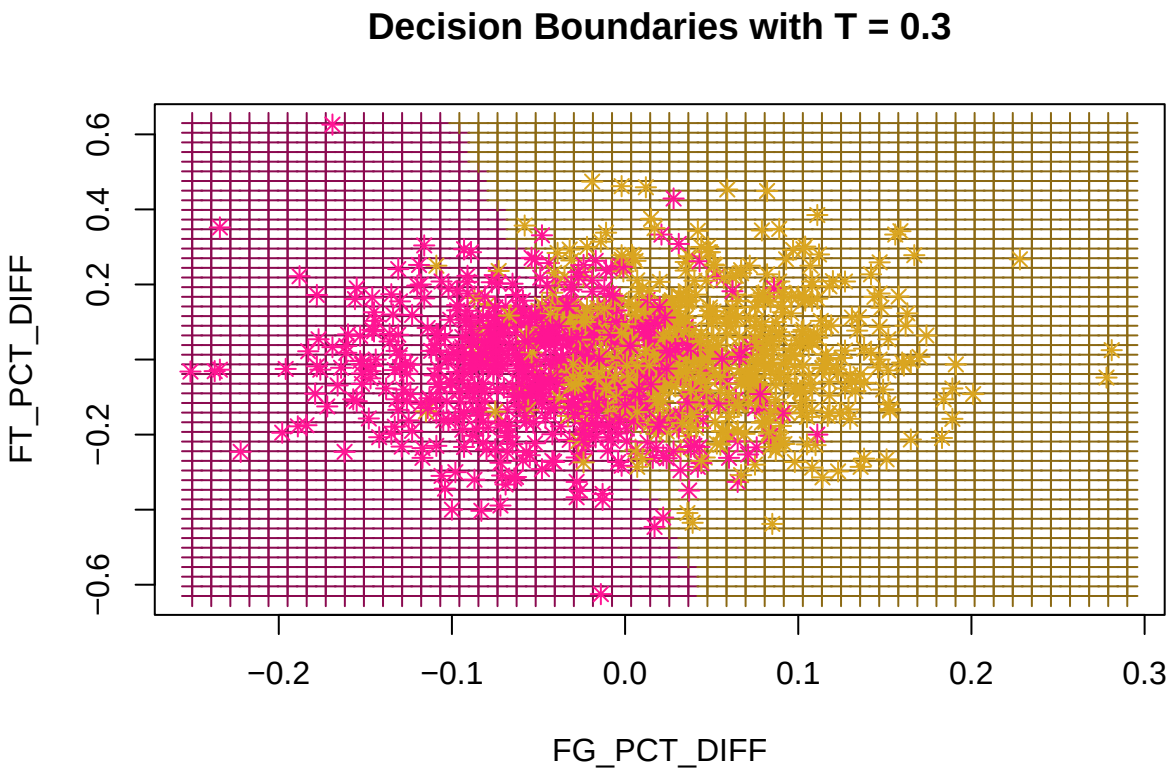
```
## [1] -0.251  0.281
```

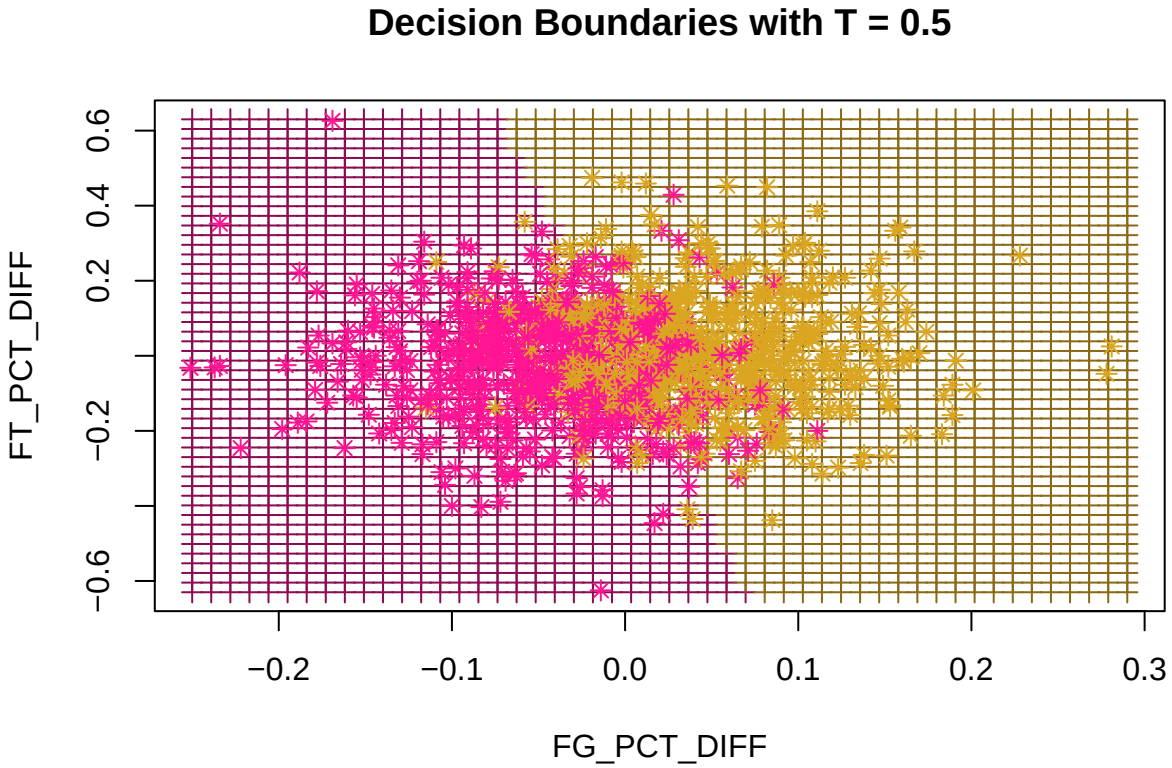
```
range(nba2$FT_PCT_DIFF)
```

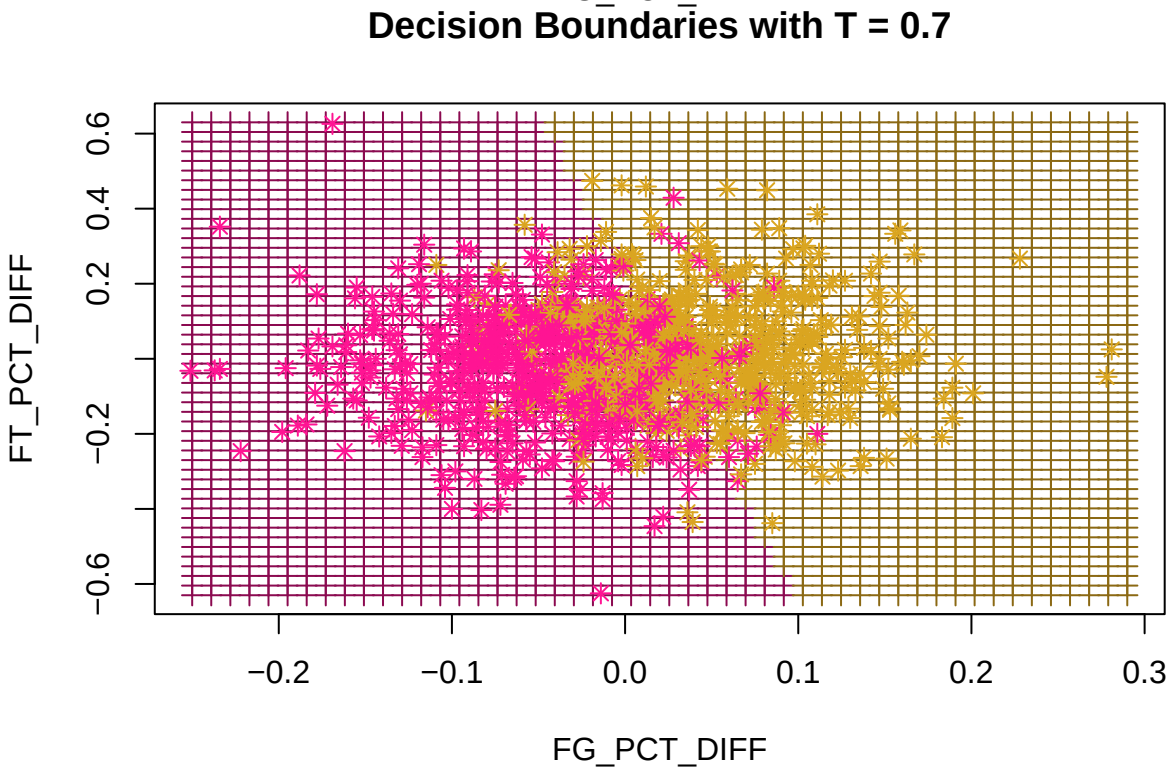
```
## [1] -0.625  0.626
```

```
#setting up the axes
fg_grid <- seq(-0.25, 0.29, length=50)
ft_grid <- seq(-0.63, 0.63, length=50)

#setting up the grid
two_variable_grid <- expand.grid(FG_PCT_DIFF = fg_grid, FT_PCT_DIFF = ft_grid)

predicted_probabilities <- predict(fit.glm, newdata = two_variable_grid, type = "response")
classes_0.3 <- p2class(predicted_probabilities, 0.3)
classes_0.5 <- p2class(predicted_probabilities, 0.5)
classes_0.7 <- p2class(predicted_probabilities, 0.7)
```

Below, we see the three required plots:







Looking at the three plots above, it seems like the decision boundary looks pretty linear on all three plots, and by eyeballing, they seem to have similar slopes. However, as the threshold increases, for each value of `FG_PCT_DIFF`, a higher value of `FT_PCT_DIFF` is necessary for the model to class a point as `1`, so the boundary "moves to the right" as the threshold increases.

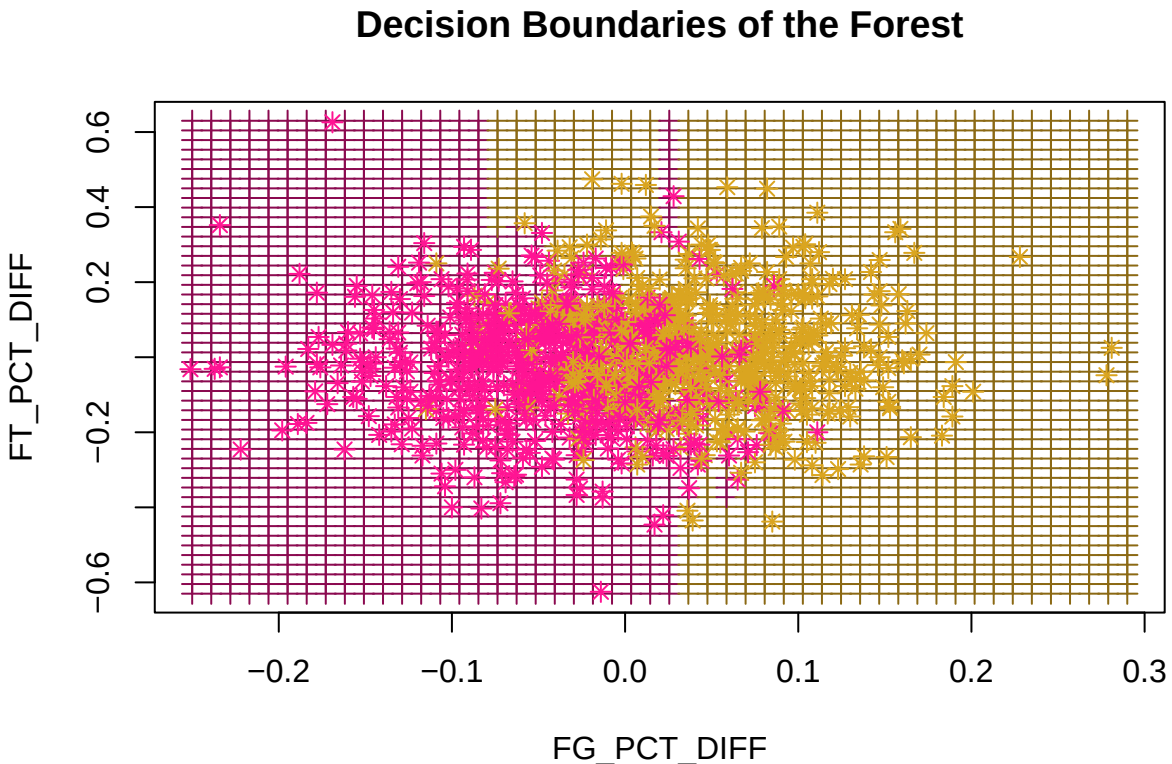
Helping Stop Deforestation

Now, we grow a forest:

```
set.seed(2000)
T <- 1000
m <-2
nba2$WON <- as.factor(nba2$WON)
forest <- randomForest(WON ~ FG_PCT_DIFF + FT_PCT_DIFF, data = nba2, ntree = T, mtry = m)
forest
```

```
##
## Call:
##  randomForest(formula = WON ~ FG_PCT_DIFF + FT_PCT_DIFF, data = nba2,      ntree = T, mtry = m)
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 22.6%
## Confusion matrix:
##     0   1 class.error
## 0 488 137   0.2192000
## 1 141 464   0.2330579
```

The estimated accuracy of the forest is 77.8%.



As we can see, the decision boundary no longer looks linear, compared to the plots from earlier. The boundary also looks less clean, as we can see two sections of pink in the yellow part of the grid near the top of the plot.

```
m <-1
forest_1 <- randomForest(WON ~ FG_PCT_DIFF + FT_PCT_DIFF, data = nba2, ntree = T, mtry = m)
forest_1
```

```
##
## Call:
##  randomForest(formula = WON ~ FG_PCT_DIFF + FT_PCT_DIFF, data = nba2,      ntree = T, mtry = m)
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 1
##
##         OOB estimate of  error rate: 22.2%
## Confusion matrix:
##     0   1 class.error
## 0 491 134   0.2144000
## 1 139 466   0.2297521
```

Repeating with only one variable tried at every split, we have a higher accuracy rate of 78.13%. We should do a hypothesis test to see if this increase in accuracy is significantly higher.