```
library(tidyverse)
library(rusboost)
library(pROC)
load("C:/Users/cjy2001/OneDrive - Middlebury College/Middlebury Academic/Fall 2021/Project/Final
_Data.RData")
```

This R markdown aims to study why the prediction for year8(2008)’s model behaves so strangely.

At first, I compute all the ROC curves from year 2003 to 2008. It seems like only year8’s curve looks so abnormal comparing to others: the ROC curve looks promising at first. However, there is a sharp turn when x approximately approaches 0.3, and the weird shape continues until x reaches 0.75, thereby reudcing its AUC value largely. Therefore, I’d like to find those threshold values which causes that shape and try to find the reason behind them.

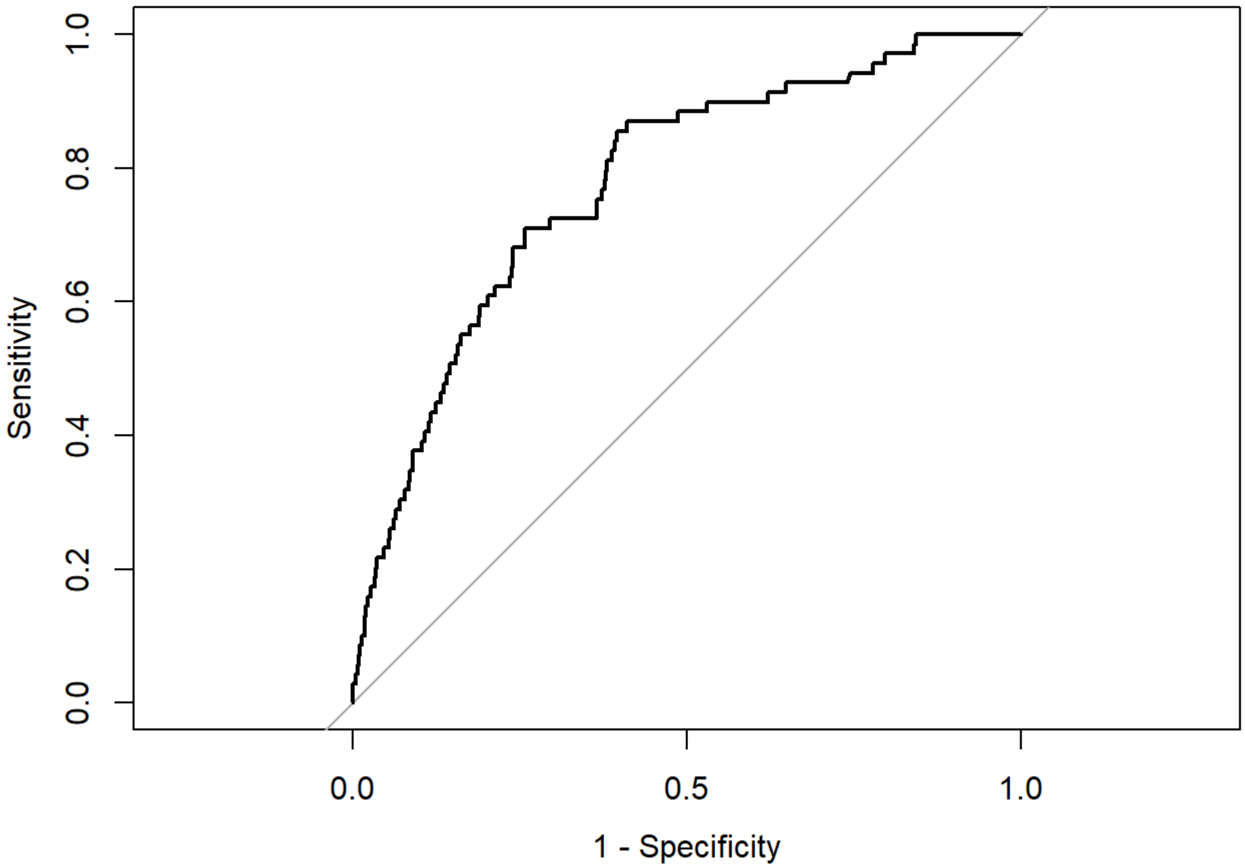
```
new_yearly_auc <- NULL

for (i in 1:6) {
  roc_obj <- roc(predictor = yearly_preds[[i]]$prob[,2],
                 response = yearly_test_set[[i]]$misstate)
  new_yearly_auc[i] = auc(roc_obj)
  plot(roc_obj, legacy.axes = TRUE)
}
```

```
## Setting levels: control = 0, case = 1
```

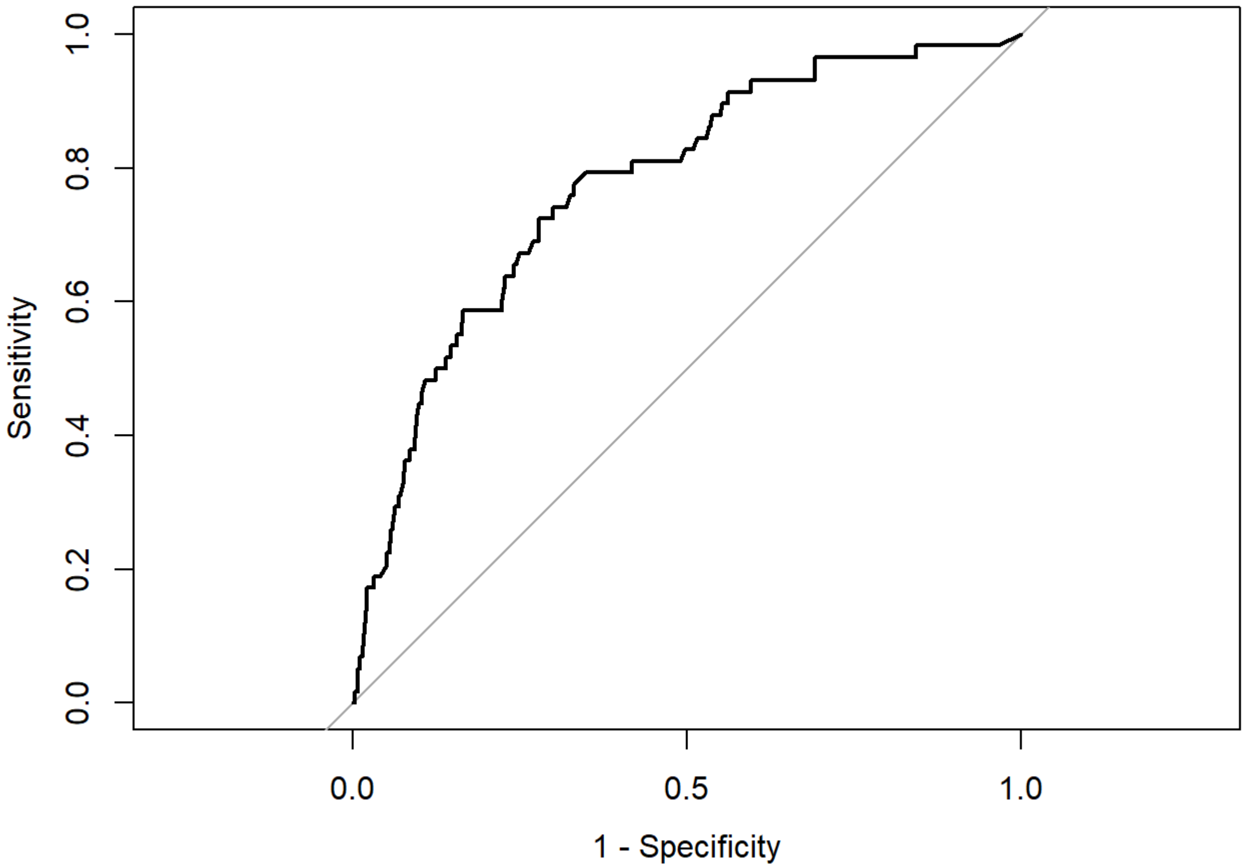
```
## Setting direction: controls < cases
```

```
## Setting levels: control = 0, case = 1
```

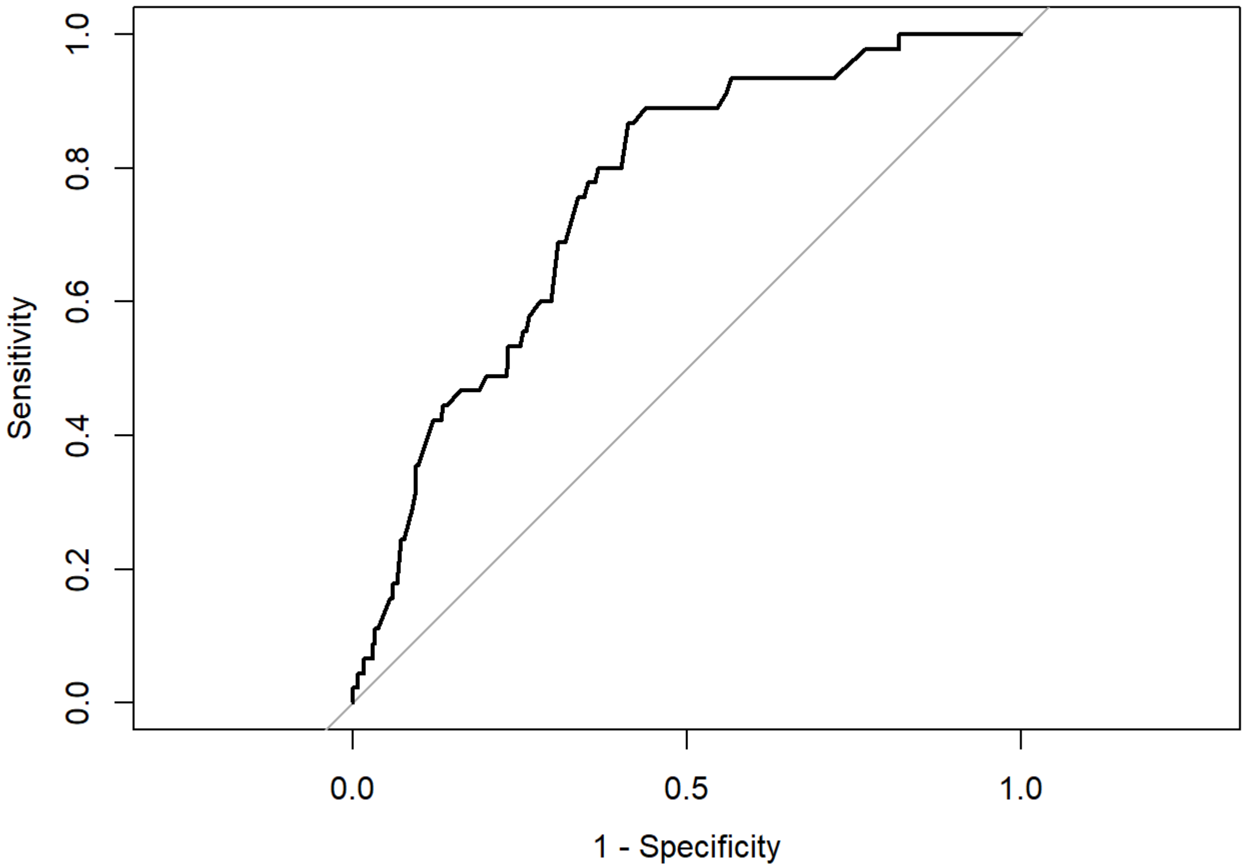
```
## Setting direction: controls < cases
```



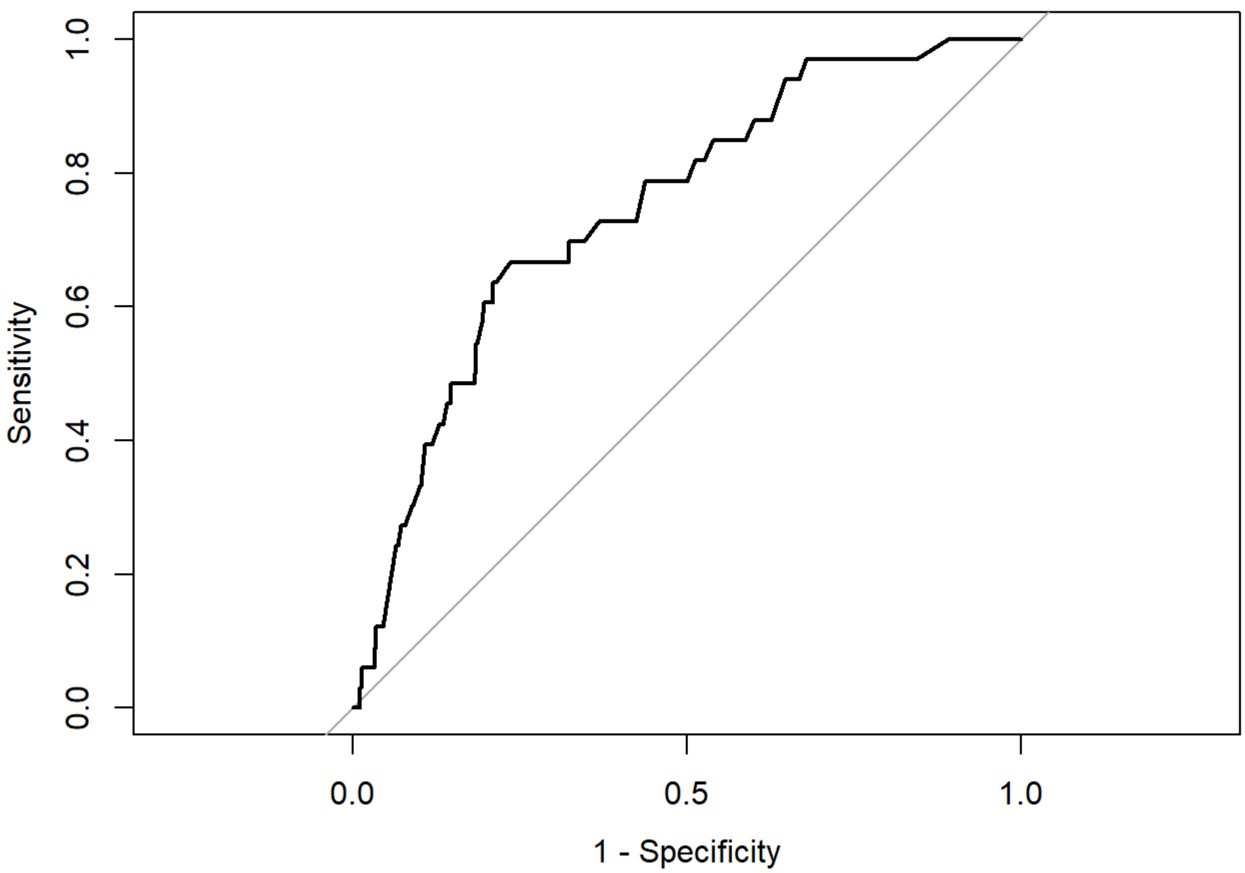
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```



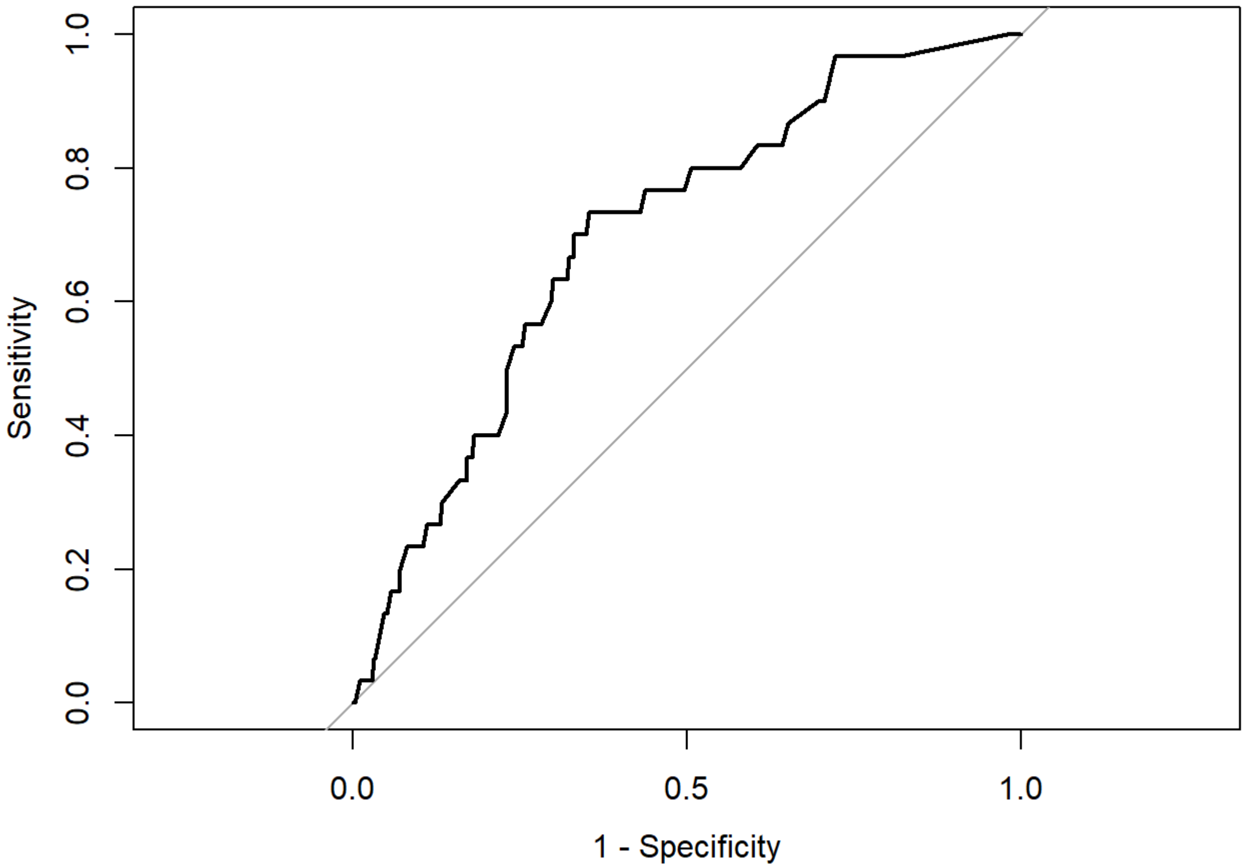
```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

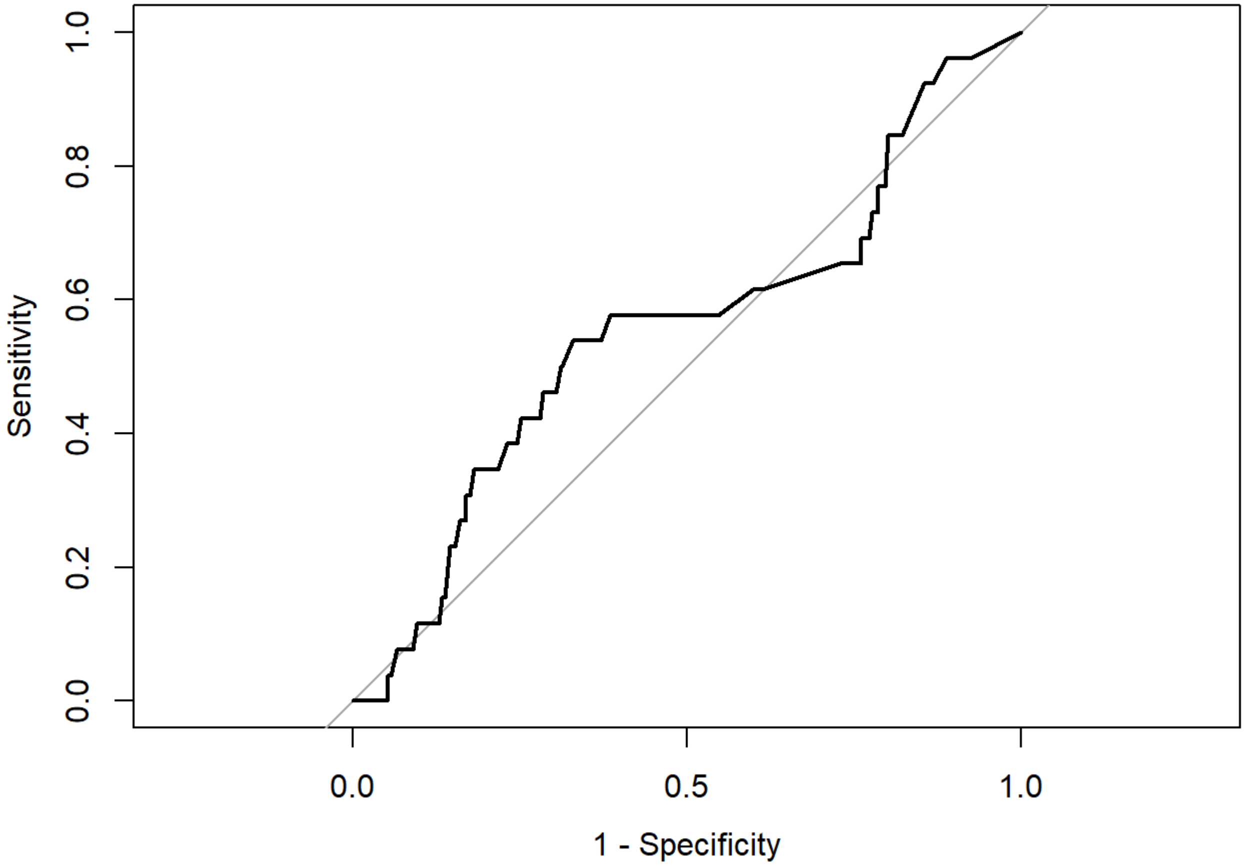


```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```



```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```


Sensitivity
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.5
1.0
1 - Specificity


Sensitivity
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.5
1.0
1 - Specificity

Next I am going to adjust threshold values and record each value’s corresponding coordinates (1-specificity as x-axis and sensitivity as y-axis.) The loop will stop and report an error when it reaches the limit of predicted probability, which means based on that threshold and afterwards, every observation will be predicted as negative.

```
threshold <- seq(0.01, 1, by = 0.01) # Setting threshold vectors
new_sensitivity <- NULL
one_minus_specificity <- NULL

# Append observation's probability
new_test_8 <- cbind(test_8, year8_preds_3000$prob) %>%
  rename("Yes" = "2", "No" = "1")
```

```
for (i in 1:length(threshold)) {
  print(threshold[i])
  new_test_8 <- new_test_8 %>%
    mutate(preds = case_when(new_test_8$Yes >= threshold[i] ~ "Yes",
                           TRUE ~ "No"))

#Confusion Matrix
new_result <- table(new_test_8$misstate, new_test_8$preds)[2:1, 2:1]

# Sensitivity = TP/P
# Specificity = TN/N
new_sensitivity[i] <- new_result[1,1]/sum(new_result[1,])
one_minus_specificity[i] <- 1 - new_result[2,2]/sum(new_result[2,])
}
```

```
## [1] 0.01
## [1] 0.02
## [1] 0.03
## [1] 0.04
## [1] 0.05
## [1] 0.06
## [1] 0.07
## [1] 0.08
## [1] 0.09
## [1] 0.1
## [1] 0.11
## [1] 0.12
## [1] 0.13
## [1] 0.14
## [1] 0.15
## [1] 0.16
## [1] 0.17
## [1] 0.18
## [1] 0.19
## [1] 0.2
## [1] 0.21
## [1] 0.22
## [1] 0.23
## [1] 0.24
## [1] 0.25
## [1] 0.26
## [1] 0.27
## [1] 0.28
## [1] 0.29
## [1] 0.3
## [1] 0.31
## [1] 0.32
## [1] 0.33
## [1] 0.34
## [1] 0.35
## [1] 0.36
## [1] 0.37
## [1] 0.38
## [1] 0.39
## [1] 0.4
## [1] 0.41
## [1] 0.42
## [1] 0.43
## [1] 0.44
## [1] 0.45
## [1] 0.46
## [1] 0.47
## [1] 0.48
## [1] 0.49
## [1] 0.5
## [1] 0.51
## [1] 0.52
## [1] 0.53
## [1] 0.54
## [1] 0.55
```

```
## Error in `[.default`(table(new_test_8$misstate, new_test_8$preds), 2:1, : subscript out of bo
unds
```

It seems like when threshold is set to 0.55, all observations will be predicted as non-fraudulent. Here I just want to test if that's right by creating two confusion matrices here, and it proves my finding.

```
new_test_8 <- new_test_8 %>%
  mutate(preds = case_when(new_test_8$Yes >= 0.54 ~ "Yes",
                           TRUE ~ "No"))
table(new_test_8$misstate, new_test_8$preds)
```

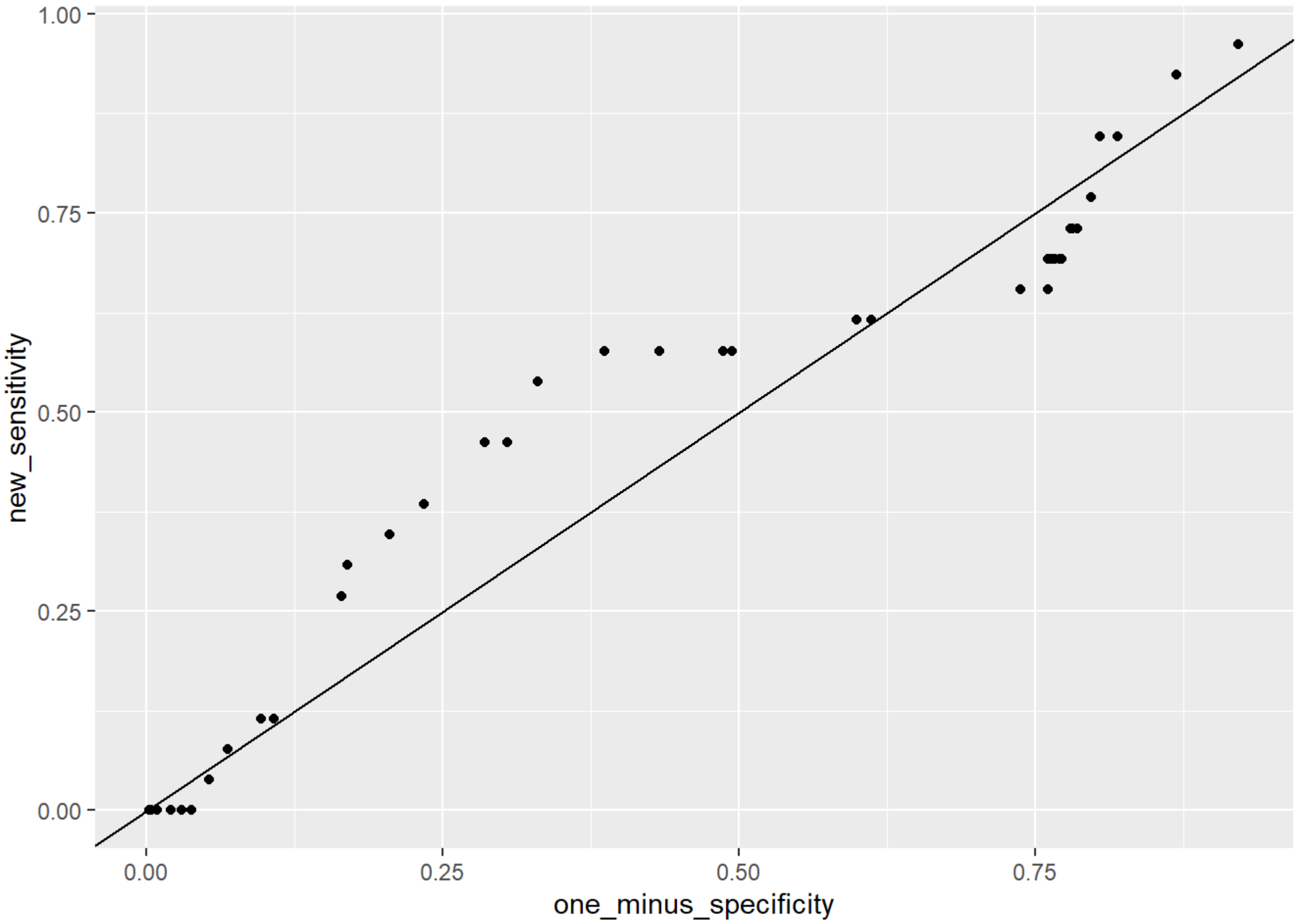
```
##
##       No  Yes
##   0 5574   12
##   1   26    0
```

```
new_test_8 <- new_test_8 %>%
  mutate(preds = case_when(new_test_8$Yes >= 0.55 ~ "Yes",
                           TRUE ~ "No"))

table(new_test_8$misstate, new_test_8$preds)
```

```
##
##       No
##   0 5586
##   1   26
```

Lastly, I plot all threshold values on the same graph based on their corresponding sensitivity and specificity values. The graph is the same as the one we get from the AUC function.

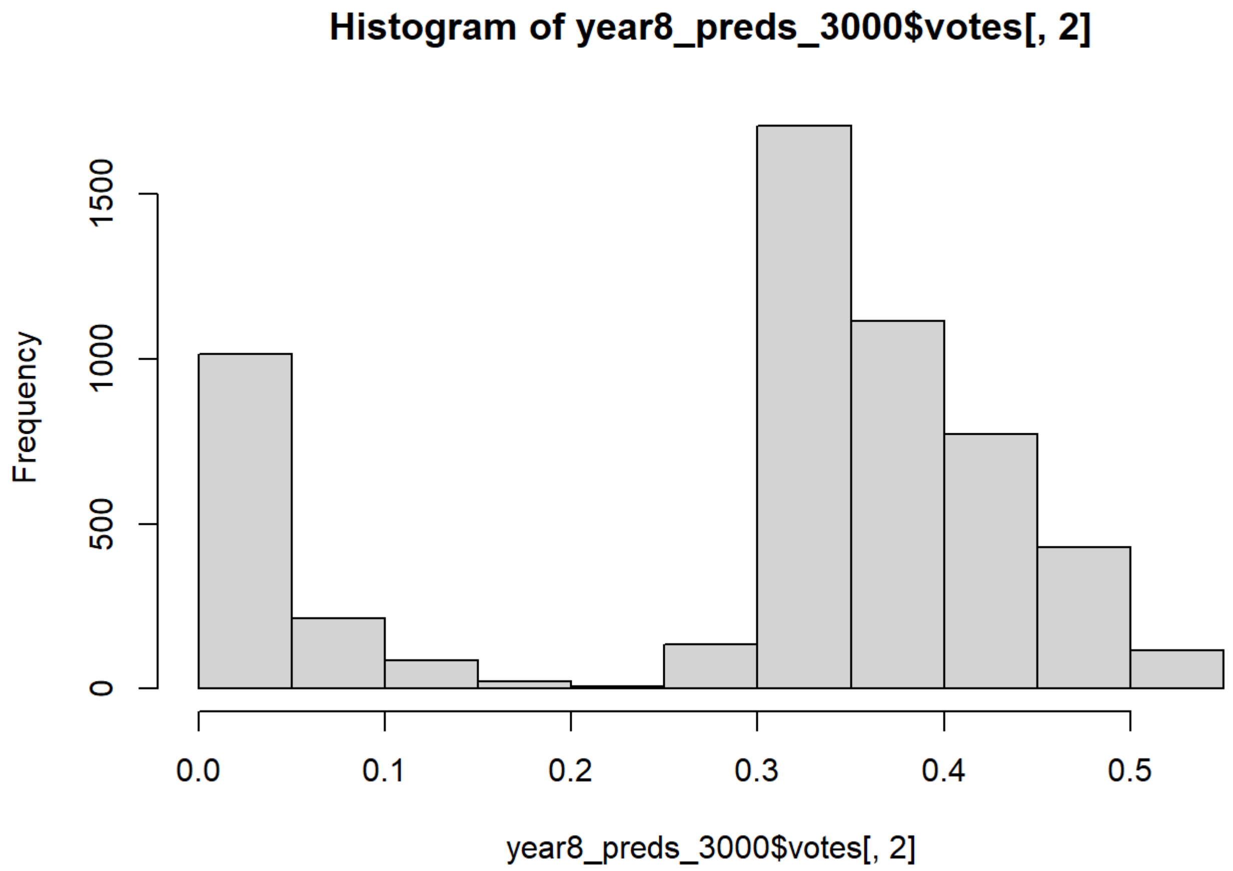
```
ggplot() +
  geom_point(aes(x = one_minus_specificity, y = new_sensitivity)) +
  geom_abline(intercept =0 , slope = 1)
```



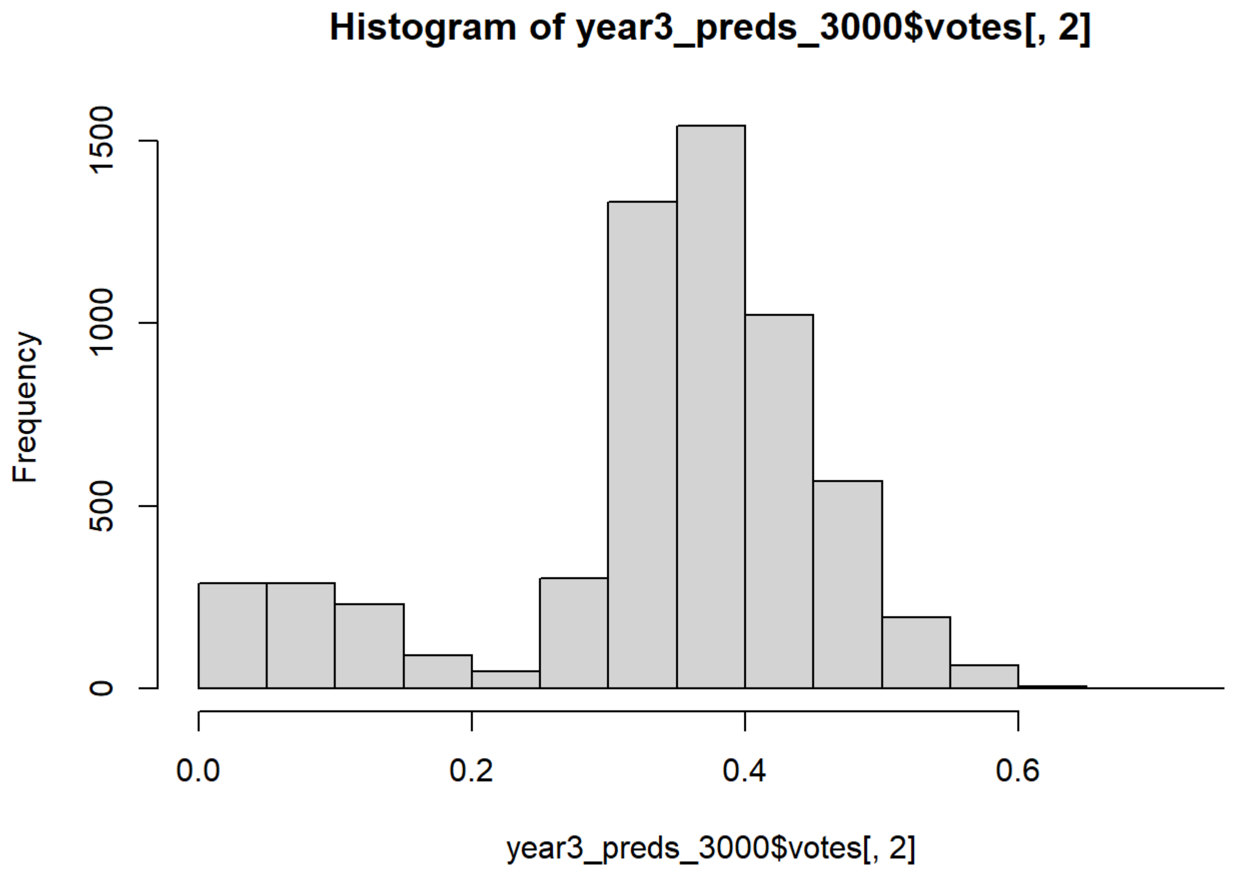
```
data.frame(threshold = threshold[1:54], one_minus_specificity, new_sensitivity) %>%
  filter(one_minus_specificity > 0.38 & one_minus_specificity < 0.875) %>%
  arrange(one_minus_specificity)
```

```
##    threshold one_minus_specificity new_sensitivity
## 1       0.36             0.3865020       0.5769231
## 2       0.35             0.4323308       0.5769231
## 3       0.34             0.4867526       0.5769231
## 4       0.33             0.4942714       0.5769231
## 5       0.32             0.5989975       0.6153846
## 6       0.31             0.6117078       0.6153846
## 7       0.30             0.7373792       0.6538462
## 8       0.29             0.7602936       0.6538462
## 9       0.28             0.7604726       0.6923077
## 10      0.27             0.7606516       0.6923077
## 11      0.26             0.7610097       0.6923077
## 12      0.24             0.7613677       0.6923077
## 13      0.25             0.7613677       0.6923077
## 14      0.18             0.7624418       0.6923077
## 15      0.19             0.7624418       0.6923077
## 16      0.20             0.7624418       0.6923077
## 17      0.21             0.7624418       0.6923077
## 18      0.22             0.7624418       0.6923077
## 19      0.23             0.7624418       0.6923077
## 20      0.16             0.7631579       0.6923077
## 21      0.17             0.7631579       0.6923077
## 22      0.14             0.7662012       0.6923077
## 23      0.15             0.7662012       0.6923077
## 24      0.13             0.7704977       0.6923077
## 25      0.12             0.7721088       0.6923077
## 26      0.11             0.7798067       0.7307692
## 27      0.10             0.7814178       0.7307692
## 28      0.09             0.7851772       0.7307692
## 29      0.08             0.7968135       0.7692308
## 30      0.07             0.8045113       0.8461538
## 31      0.05             0.8191908       0.8461538
## 32      0.06             0.8191908       0.8461538
## 33      0.03             0.8686001       0.9230769
## 34      0.04             0.8686001       0.9230769
```

```
hist(year8_preds_3000$votes[,2])
```



```
hist(year3_preds_3000$votes[,2])
```



So what makes this year’s model not classification-threshold invariant? I think one possible explanation could be the predicted probability’s distribution. According to the histogram, there are too many points distributed between 0.3 and 0.35, and that’s where our ROC curve gets weird. As a comparison, the year3’s (2003) distribution is more even. As the threshold value decreases from 0.36, sensitivity does not vary too much, implying that even though there are many observations distributed on that range, few of them are correctly predicted as fraudulent. In contrast, specificity is computed as TN/N, and it will keep changing because the model tends to predict more negatives. Therefore, as x value increases while y value stays the same, the ROC curve will become more flat, thereby having a lower AUC score.