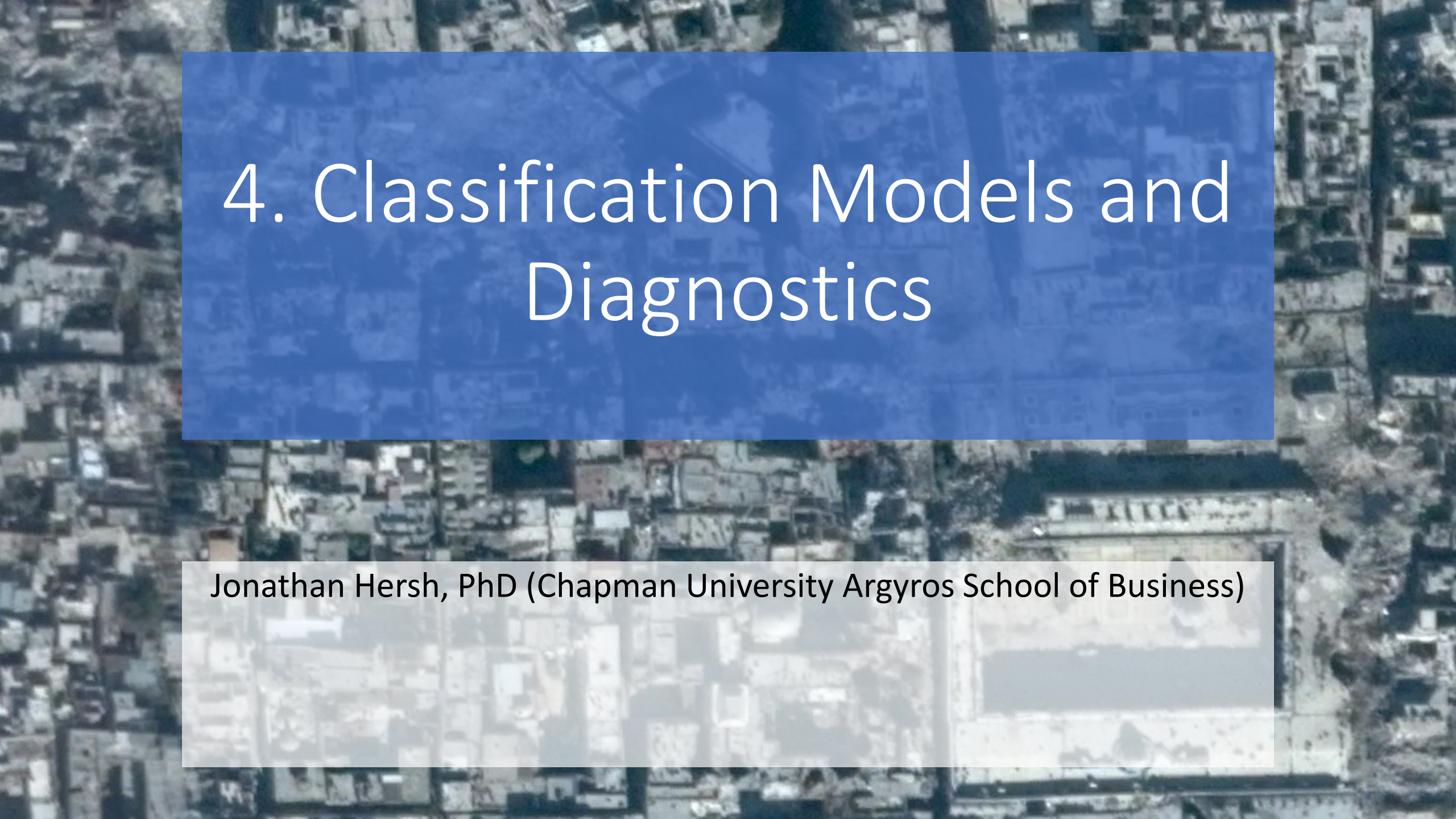


4. Classification Models and Diagnostics

Jonathan Hersh, PhD (Chapman University Argyros School of Business)

Outline

1. Classification with Logistic
2. Diagnostics
   - Confusion Matrices
   - ROC Curves
   - Calibration Plots
3. Class Imbalance

Classification Basics

Classification examples

Credit Card Default Dataset

Default {ISLR} R Documentation

Credit Card Default Data

Description

A simulated data set containing information on ten thousand customers. The aim here is to predict which customers will default on their credit card debt.

Usage

```
Default
```

Format

A data frame with 10000 observations on the following 4 variables.

`default`

A factor with levels `No` and `Yes` indicating whether the customer defaulted on their debt

`student`

A factor with levels `No` and `Yes` indicating whether the customer is a student

`balance`

The average balance that the customer has remaining on their credit card after making their monthly payment

`income`

Income of customer

Source

Simulated data

Using the Logistic/Sigmoid Function to Generate Probabilities

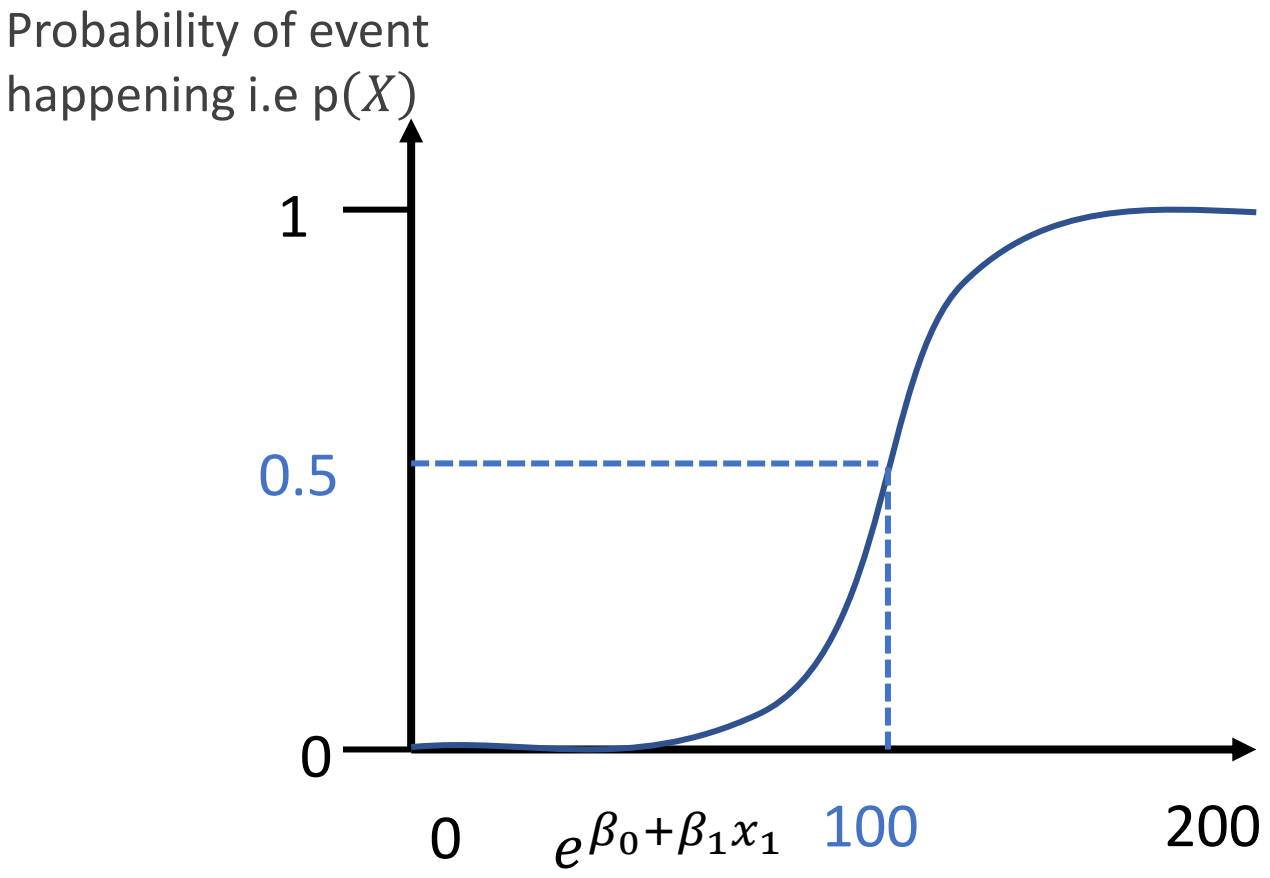
- How do we generate probabilities from the logistic function?

- We let X = $\beta_0 + \beta_1 \cdot x_1 + \cdots + \beta_k \cdot x_k$ and plug this into the logistic function

$$\sigma(X) = \frac{1}{1 + e^{-X}} = \frac{e^X}{e^X + 1}$$

$$Pr(Y = 1|X) = \frac{e^{\beta_0 + \beta_1 \cdot x_1}}{e^{\beta_0 + \beta_1 \cdot X} + 1}$$

$$Pr(Y = 1|X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1)}}$$



This is equivalent mathematically! I promise. Work it out on pen and paper if you don't believe me

Generating Predicted Probabilities from a Logit Model

- To generate predictions, we use the estimated coefficients in the logit equation

$$\hat{p}(X = 1000) = \frac{e^{\hat{\beta}_0 + \hat{\beta}_1 x_1}}{1 + e^{\hat{\beta}_0 + \hat{\beta}_1 x_1}} =$$

- The estimated probability of default with a balance of $1,000 is given by
- The estimated probability of default with a balance of $2,000 is given by
- To generate predicted probability for all observations in a dataset we use the predict function, **but note type = "response"!**
- This is also called "scoring" a dataset

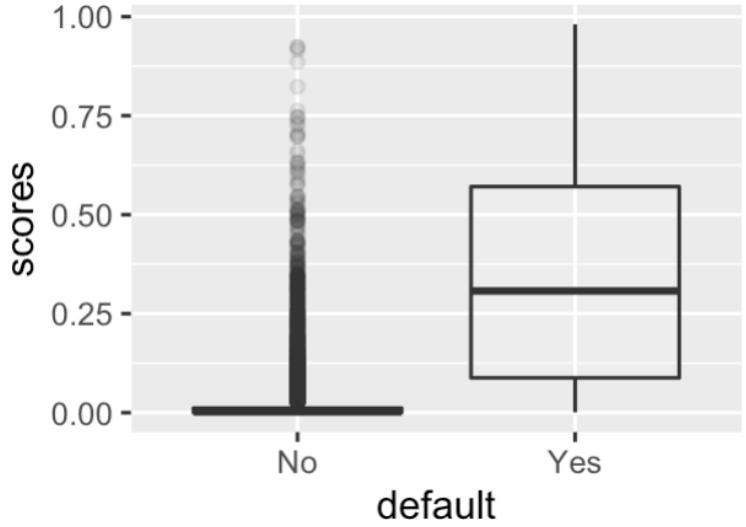
```
glm(formula = default ~ balance, family = binomial, data = Default)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2697  -0.1465  -0.0589  -0.0221   3.7589

Coefficients:
             Estimate  Std. Error z value Pr(>|z|)
(Intercept) -10.6513306   0.3611574  -29.49   <2e-16 ***
balance       0.0054989   0.0002204   24.95   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

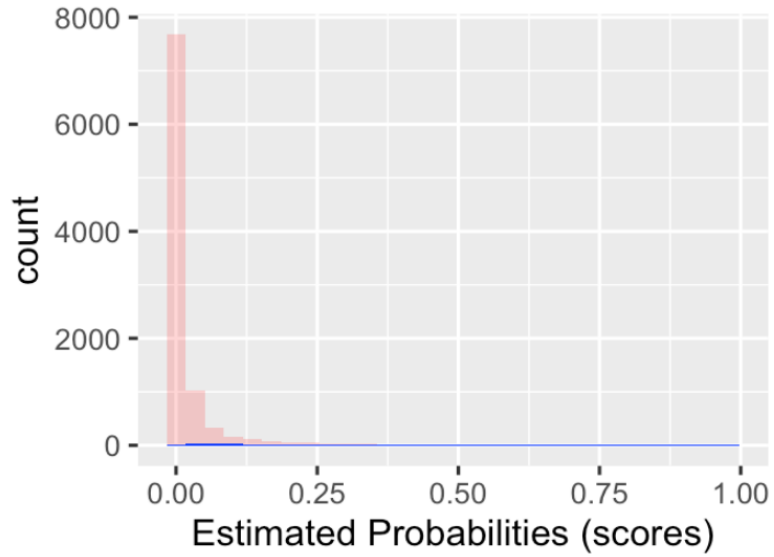
$$\hat{p}(X = 1000) = \frac{e^{-10.6513 + 0.0055 \cdot \mathbf{1000}}}{1 + e^{-10.6513 + 0.0055 \cdot \mathbf{1000}}} = 0.00575$$

$$\hat{p}(X = 2000) = \frac{e^{-10.6513 + 0.0055 \cdot \mathbf{2000}}}{1 + e^{-10.6513 + 0.0055 \cdot \mathbf{2000}}} = 0.55857$$

```
scores <- predict(logit_fit3,
                  type = "response")
```

What Do We Do With Scores or Estimated Probabilities?



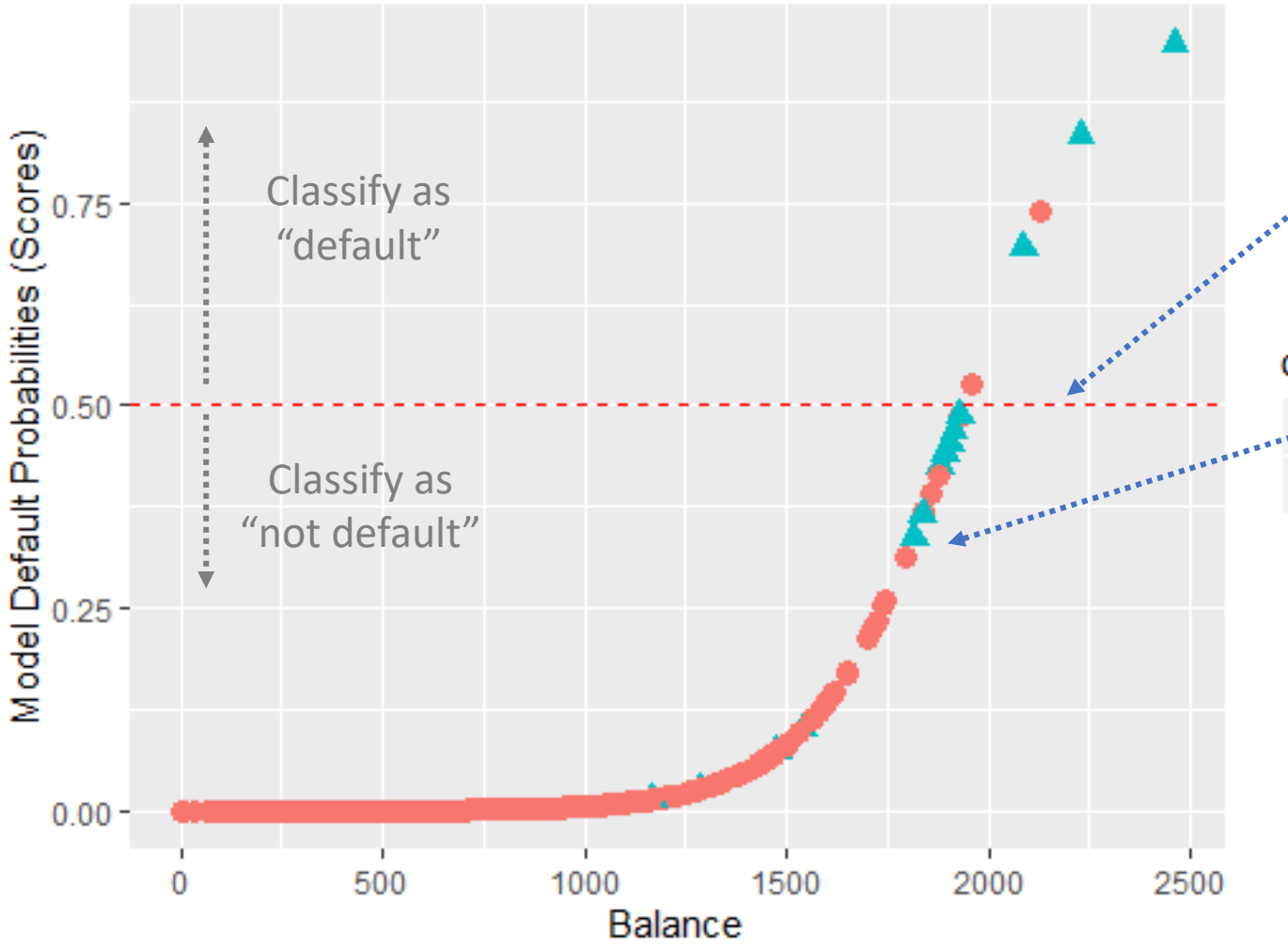


- Okay, so we have probabilities, what then?
- Note there is almost always some overlap between the probabilities of the classes
- We can't choose a probability such that above this all actual defaulters are correctly identified, and below this all non-defaulter are identified
- So we will always have some **false positives** and **false negatives**

Confusion Matrix: Table of False/True Positives and False/True Negatives

		True default status	
		No	**Yes**
Predicted default status	**No**	True negative (TN)	False Negative (FN)
	Yes	False Positive (FP)	True Positive (TP)

Assigning Class=Default to $\hat{p} > 0.5$

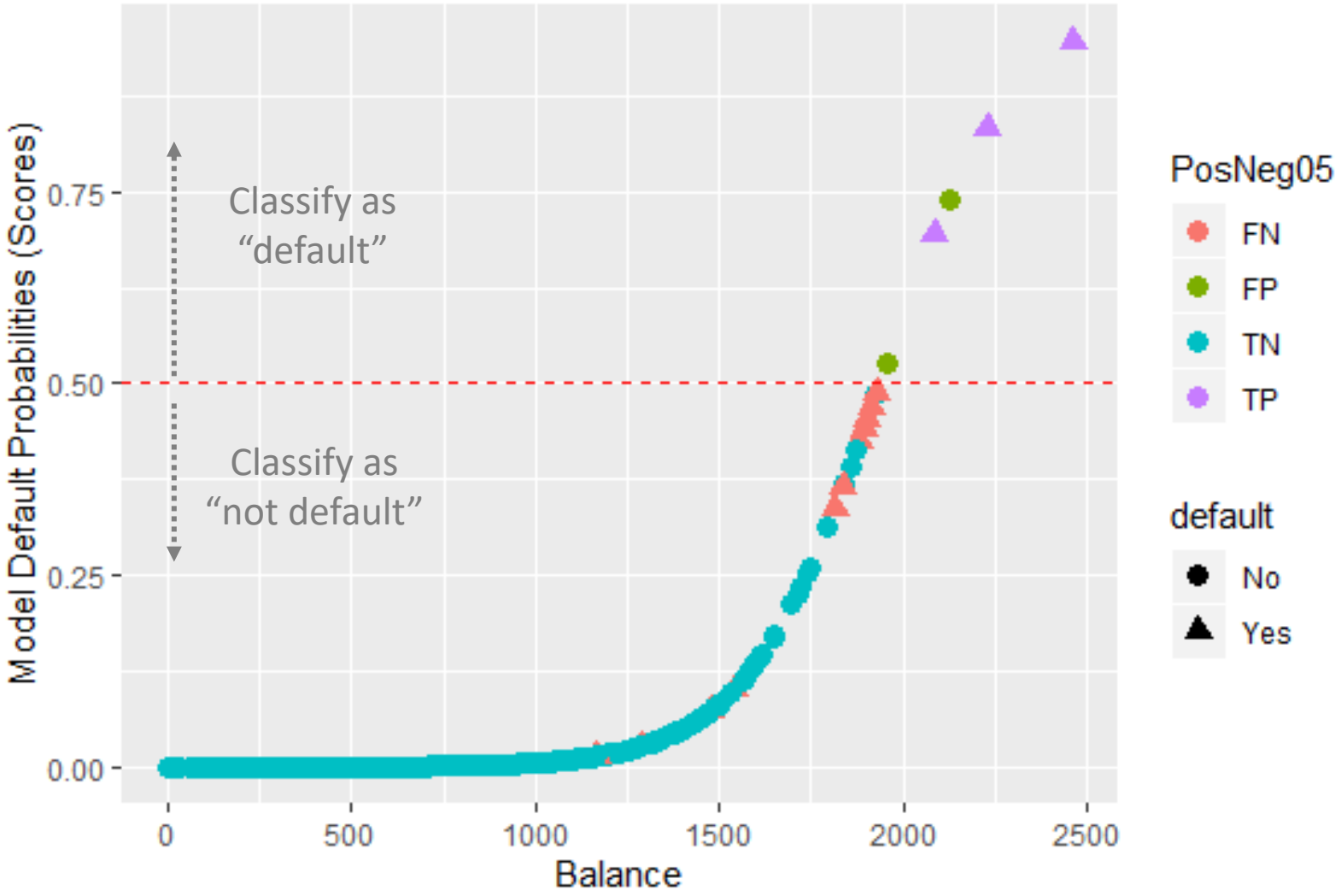


- Above this line, observations are classified as defaulting
- Below this line, observations classified as **not** defaulting
- Actual default = teal triangles
- Actual not default = circle
- If we choose a probability cutoff of 0.5, then we see we have 2 false positives and 11 FN

```
> table(preds_sample$PosNeg05)

 FN  FP  TN  TP
 11   2 484   3
```

Note I'm working with a 5% sample of the dataset to make the numbers easier

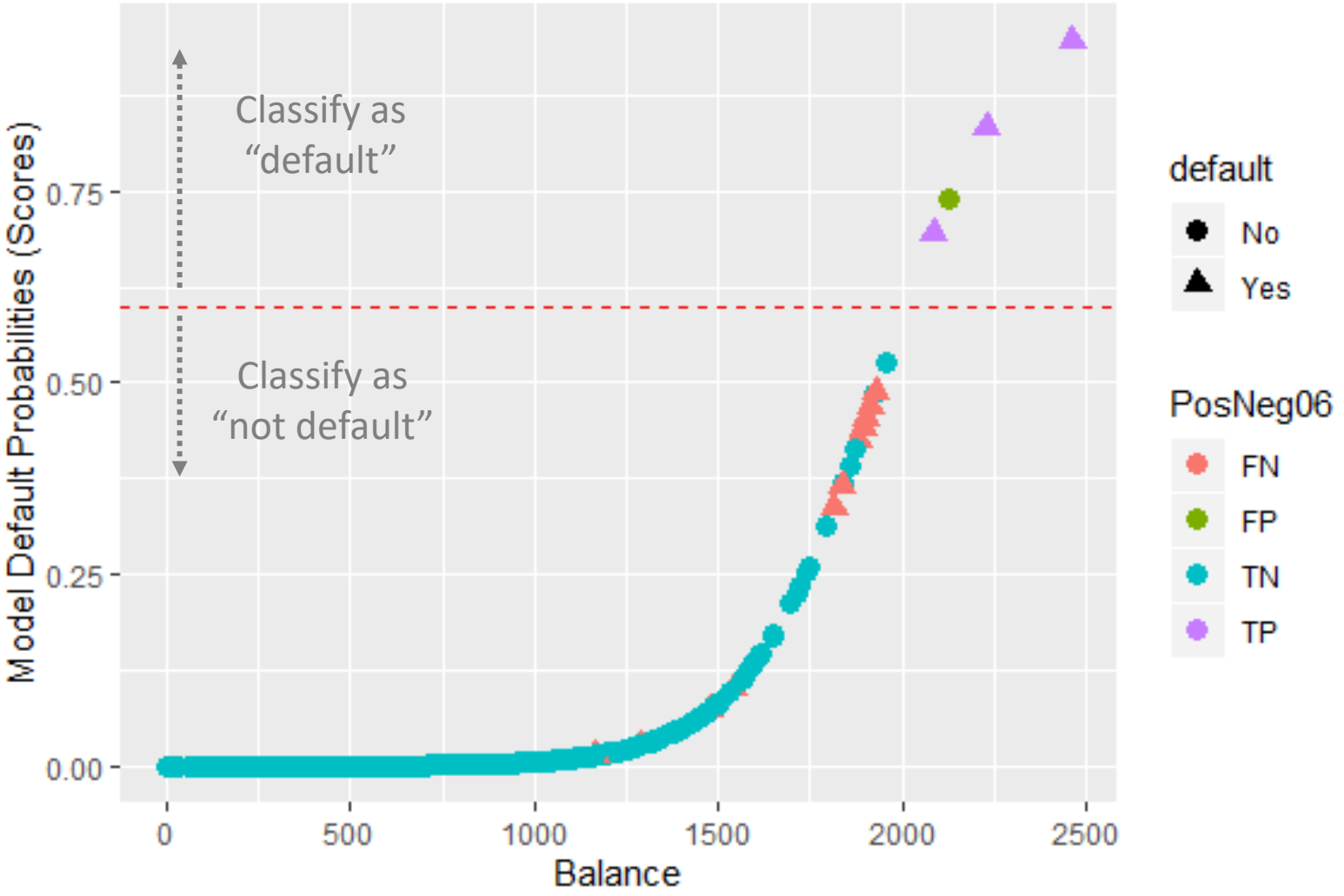
Assigning Class=Default to $\hat{p} > 0.5$



- If we choose a probability cutoff of 0.5, then we see we have 2 false positives and 11 FN

```
> table(preds_sample$PosNeg05)

 FN  FP  TN  TP
 11   2 484   3
```

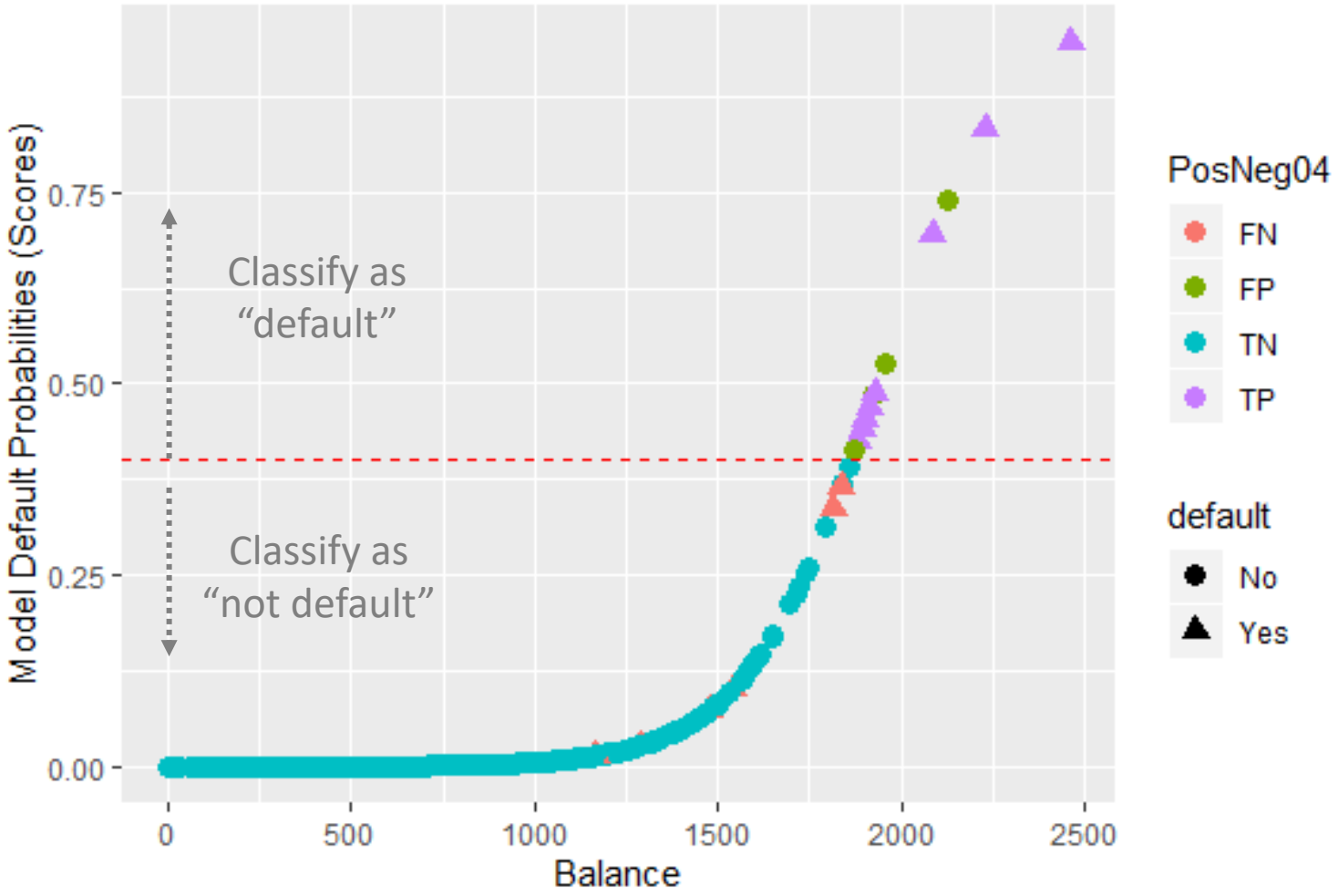
Assigning Class=Default to $\hat{p} > 0.6$



- Raising the cutoff to 0.6, then we see we have 1 false positives and 11 FN

```
> table(preds_sample$PosNeg06)

 FN  FP  TN  TP
 11   1 485   3
```

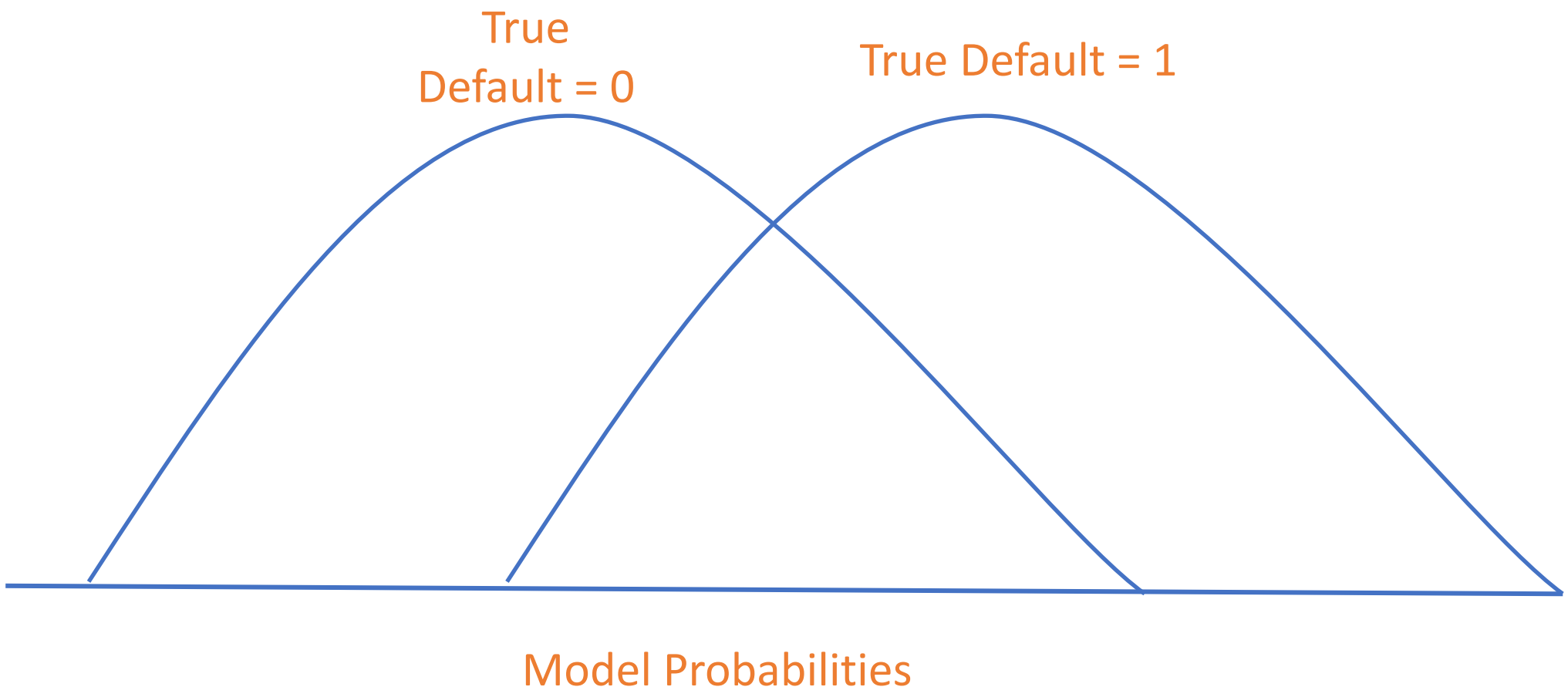
Assigning Class=Default to $\hat{p} > 0.4$



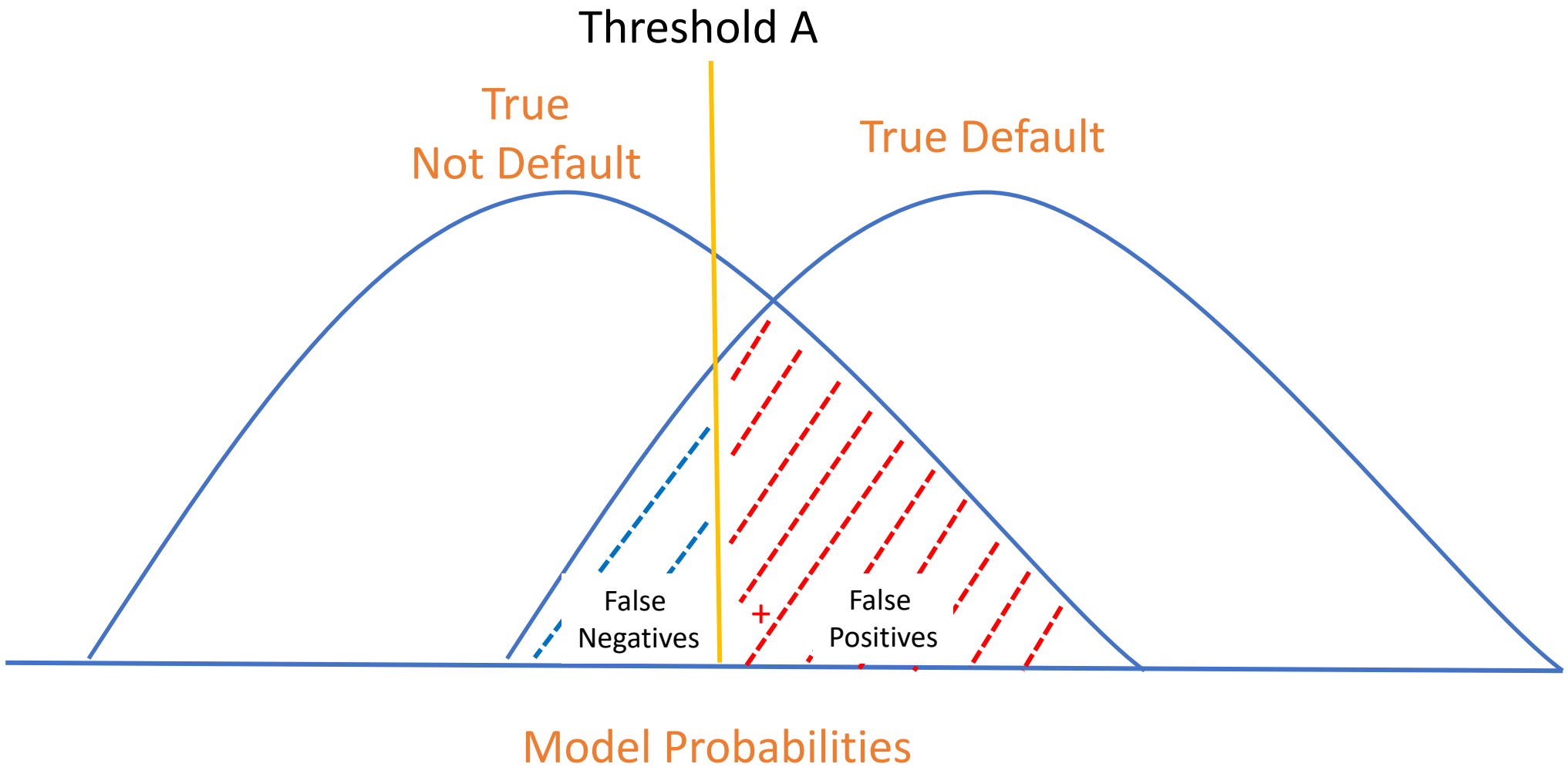
- Lowering the cutoff to 0.4 results in more FPs (4) but fewer FNs (6)

```
> table(preds_sample$PosNeg04)

 FN  FP  TN  TP
  6   4 482   8
>
```

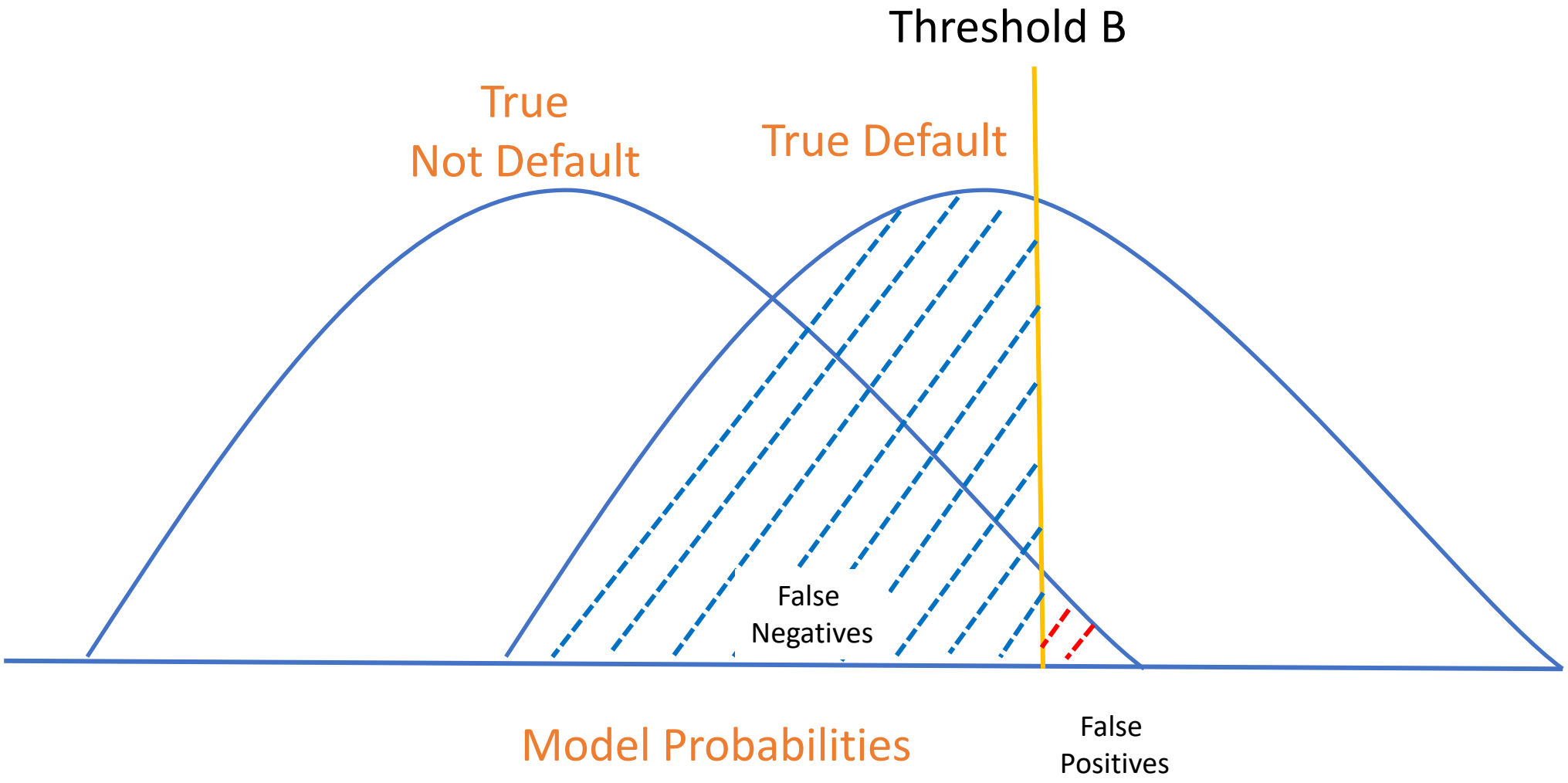
Choosing Probability Cutoff to Assign Class



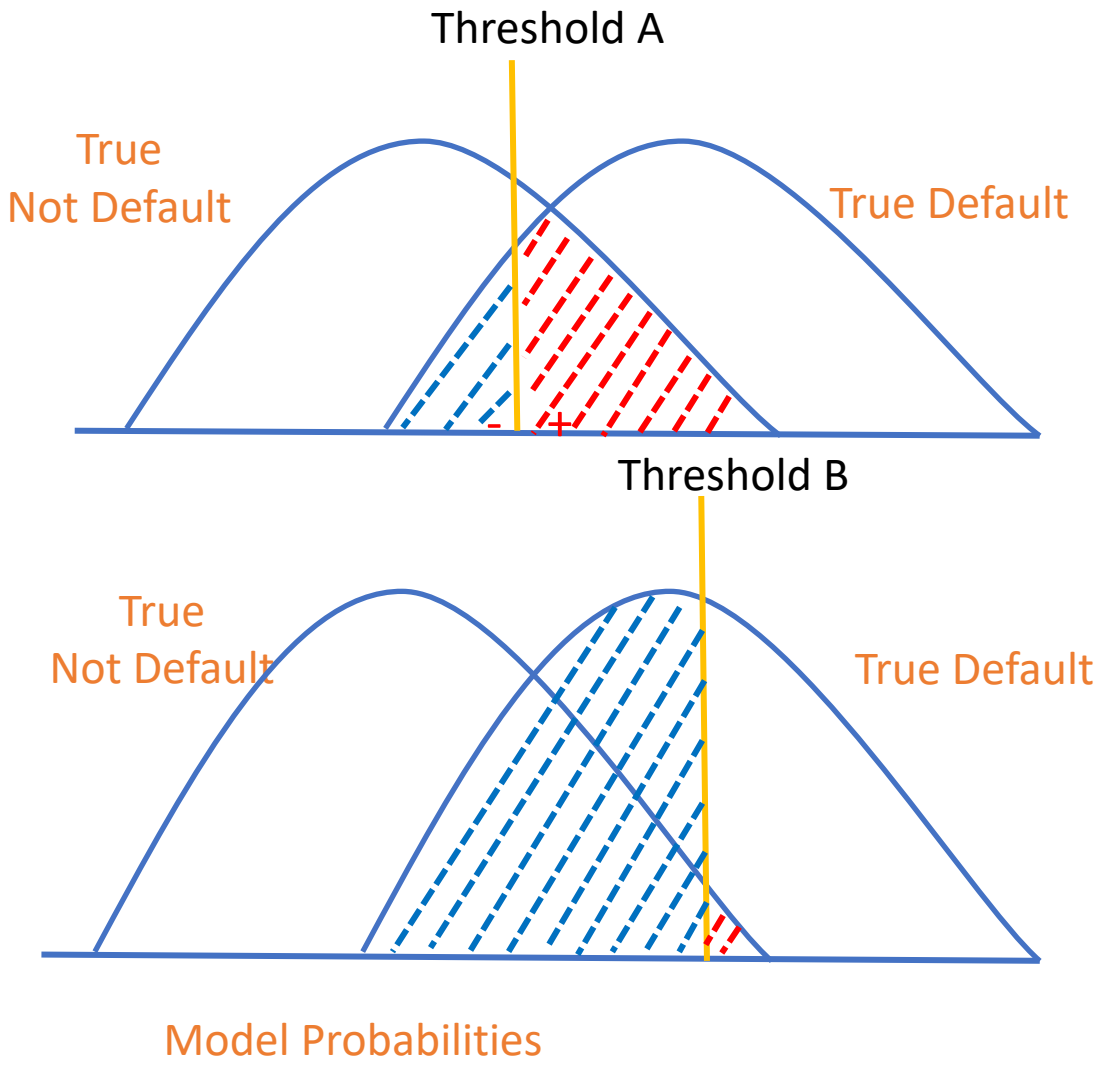
Threshold A: Moderate Threshold



Threshold B: Higher Threshold



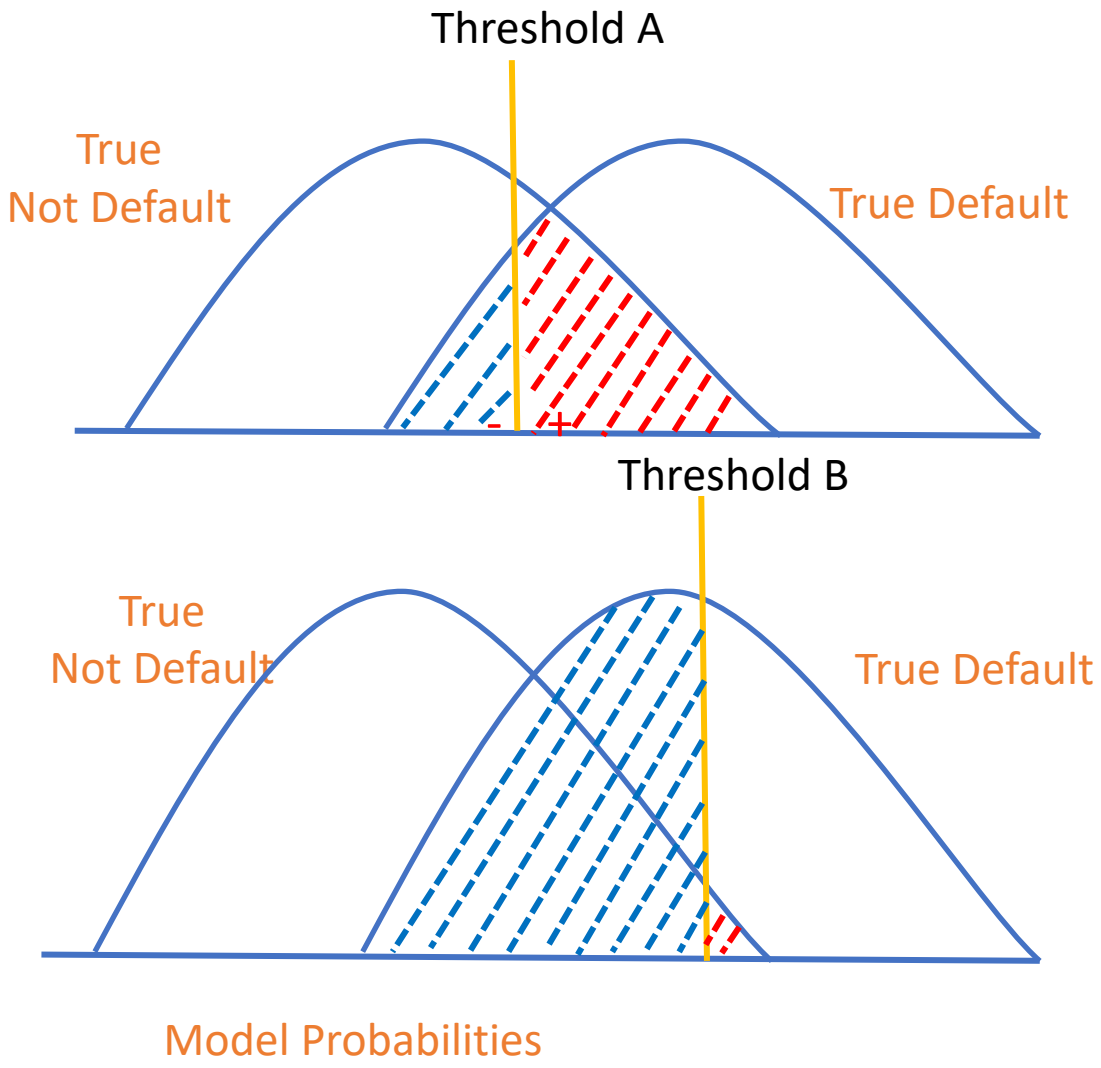
Comparing cutoffs:



Many false positives

Few false positives

Which Probability Cutoff To Use?



- Threshold you choose should depend on relative costs of FPs and FN
  - e.g. screening at airport (cost of false neg high)
  - e.g. direct mail advertisement (cost of false positive low)
- Some common choices
  - **Maximize Accuracy** (equal weighting of FPs and FNs)
  - **Threshold p_hat** > 0.5
  - **Minimize cost**: TC = costFP *FPs + costFN * FNs

Sensitivity and Specificity, Confusion Matrix at P Cutoff > 0.5

		True default status		
		No	**Yes**	
Predicted default status (cutoff p>0.5)	**No**	TN = 484	FN = 11	$N^* = 495$
	Yes	FP = 2	TP = 3	$P^* = 5$
		N = 486	P = 14	

- **Sensitivity:** True positive rate (aka 1 – power or recall)
  - TP/P = 3 / 14 = 21.4%
- **Specificity:** True negative rate
  - TN/N = 484 / 486= 99.5%

- **False positive rate** (aka Type I error, 1 - Specificity)
  - FP/N = 2/486= 0.004%

Generating Confusion Matrices in R

- To produce a confusion matrix in R we will use the yardstick package
- The function conf_mat() produces confusion matrices but we must format our data correctly
- We need to specify a data frame with
- Actual event (Y = 1) values
- Our estimated probabilities (scores)
- This example data frame shows how we need to structure our results data frame

Usage

```
conf_mat(data, ...)

## S3 method for class 'data.frame'
conf_mat(data, truth, estimate, dnn = c("Prediction", "Truth"), ...)

## S3 method for class 'conf_mat'
tidy(x, ...)

autoplot.conf_mat(object, type = "mosaic", ...)
```

Formatting Results Matrix for Confusion Matrix

- Let's store the model results in a data frame
- We must specify the actual poverty status
- We *must* specify a cutoff above which probabilities are classified as "class1" (or having the event) and below which they are not
- Here I've chosen 0.7 but we may need to alter this
- The cutoff probability is determined by the relative cost of false positives and false negatives! Do not use rules of thumb!

```
preds <- tibble(
   `true` = CR_dat$poor_stat,
    `preds_lasso` = predict(lasso_mod,
                            newdata = CR_dat,
                            s = "lambda.min",
                            type = "response")[,1],
   `preds_ridge` = predict(ridge_mod,
                           newdata = CR_dat,
                           s = "lambda.min",
                           type = "response")[,1]
)

preds_df <- preds %>%
   mutate(class_lasso =
                      as.factor(if_else(preds_lasso > 0.7,1,0)),
          class_ridge =
                      as.factor(if_else(preds_lasso > 0.7,1,0)))
```

Why Do So Many Practicing Data Scientists Not Understand Logistic Regression?

https://ryxcommar.com/2020/06/27/why-do-so-many-practicing-data-scientists-not-understand-logistic-regression/

Producing Confusion Matrix Using Formatted Results Data

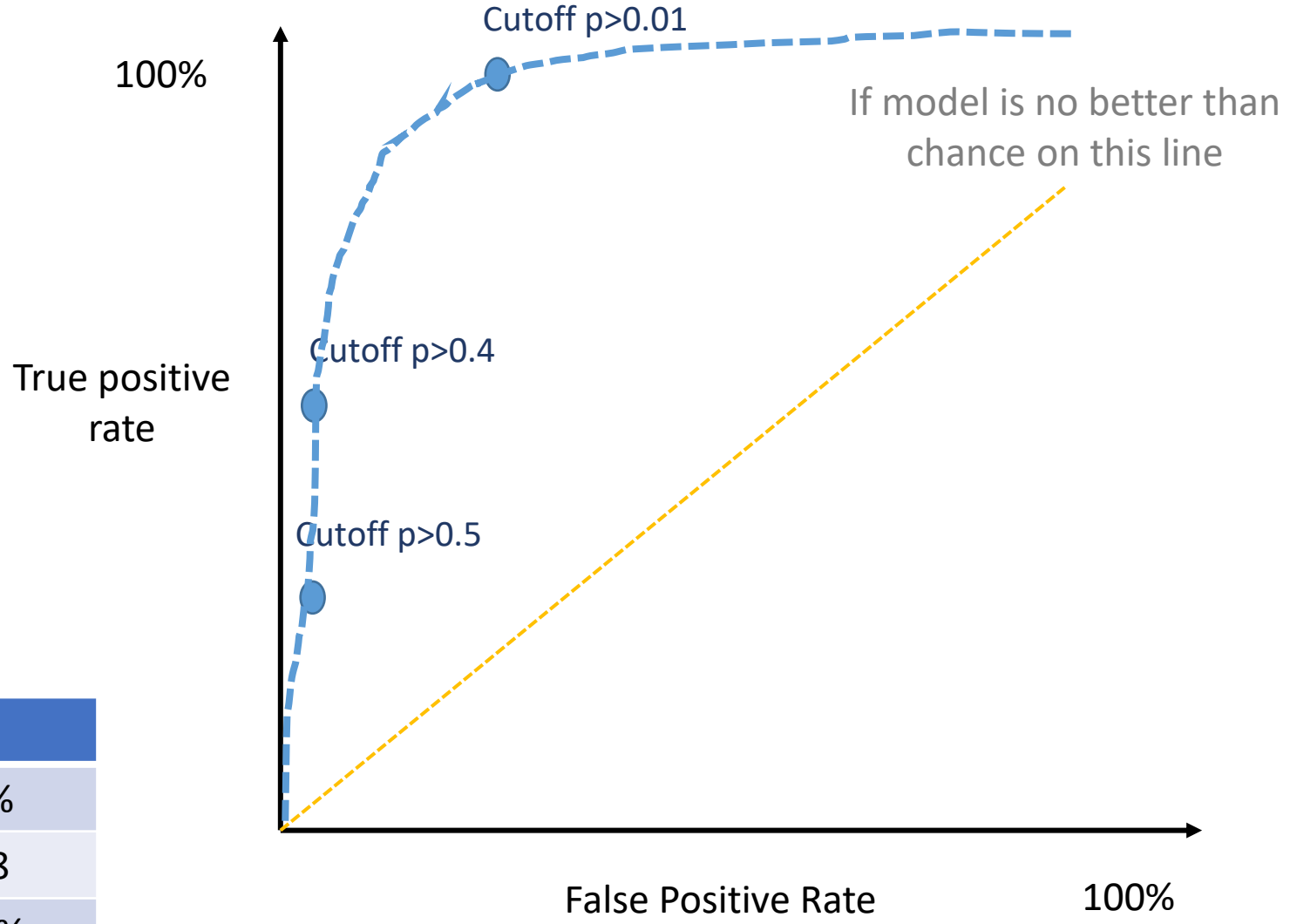
- The conf_mat() function shows the confusion matrix
- If we summarize the conf_mat() object we see more binary metrics of classification (don't need to know all of these)
- **Sensitivity** is the true positive rate (TP/P) and here we identify of the true positives = 131/333 = 39.3%
- We may need to lower our threshold of cutoff probability

```
> conf_mat(preds_df,
+          truth = true,
+          estimate = class_lasso)
          Truth
Prediction    0    1
         0  796  802
         1  532 3810
```

Continuous Cutoff: ROC Curve

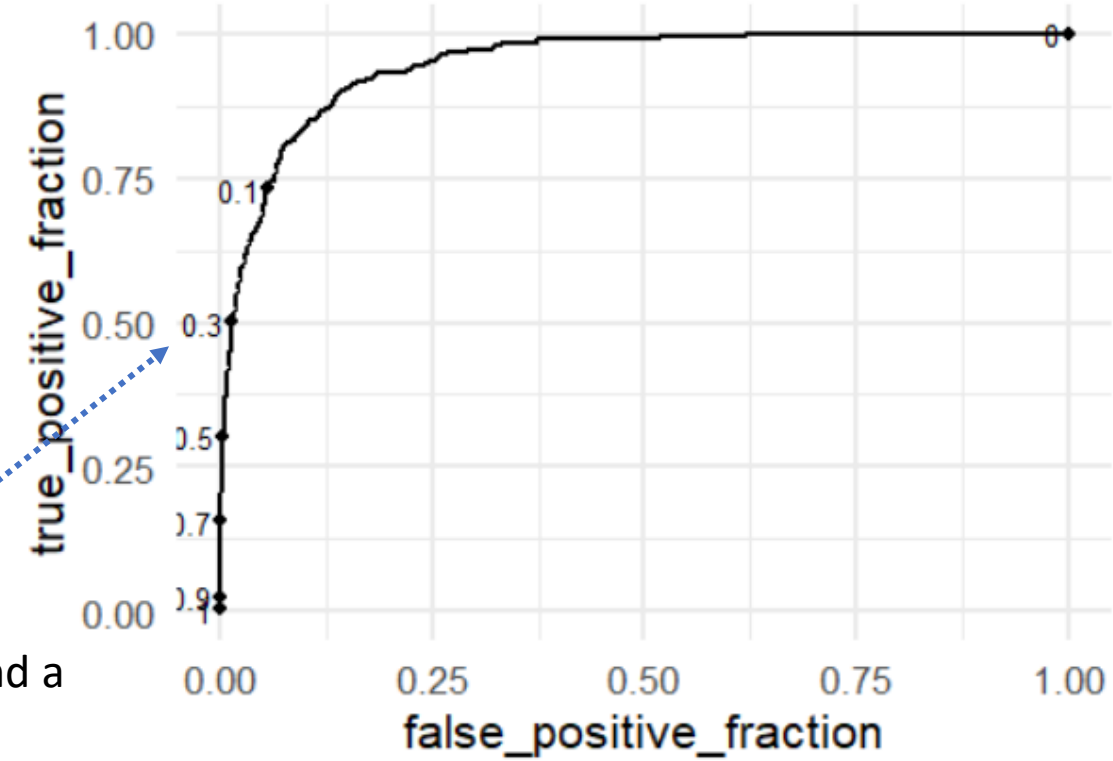
- Can we show consequences of FPs and FNs as we vary the cutoff probability to assign classes?
- Idea of a ROC (Receiver Operator Curve) plot

Cutoff	TPR	FPR
0.01	100%	22.6%
0.4	57%	0.008
0.5	21.4%	0.004%
0.6	21.4%	0.002%



ROC Curves in R

```
#-----------------------------------------
# ROC plots
#-----------------------------------------
library('ggplot2')
library('plotROC')

p <- ggplot(results_logit,
       aes(m = Class1, d = truth)) +
  geom_roc(labelsize = 3.5,
           cutoffs.at =
             c(0.99,0.9,0.7,0.5,0.3,0.1,0)) +
  theme_minimal(base_size = 16)
print(p)
```



- At a cutoff of 0.3, we get a true positive fraction of 0.5 and a false positive fraction of a very low number
- Better models lie up and to the left in the ROC plot
- AUC calculates how much total area is under a particular curve
- AUC of 0.947 is pretty good

```
> calc_auc(p)
  PANEL group       AUC
1     1    -1 0.9479842
```

Outline

1. Classification with Logistic
2. Confusion Matrices
3. ROC Curves
4. **Calibration Plots**

Chuck Klosterman on Probability

"Life is chock full of lies, but the biggest lie is math. That's particularly clear in the discipline of probability, a field od study that's completely and wholly fake. When push comes to shove - when you truly get down to the core essence of existence - there is only one mathematical possibility: Everything is 50-50. Either something will happen or it won't"

- Chuck Kloserman, being wrong about probability

- This is an incorrect understanding of probability. Obviously something can only happen or not happen
- Ex-post (after the fact) we can say odds were 50-50. But ex-ante, odds are clearly not 50-50
- There is not a 50% chance it snows tomorrow in LA. How do we know? History tells us.

Climate Averages

	Los Angeles, California	United States
Rainfall	15.5 in.	38.1 in.
Snowfall	0.0 in.	27.8 in.
Precipitation	33.7 days	106.2 days
Sunny	284 days	205 days

So What Does it Mean to Predict Binary Events Well?

FiveThirtyEight

Politics Sports Science Podcasts Video

APR. 4, 2019, AT 5:16 PM

When We Say 70 Percent, It Really Means 70 Percent

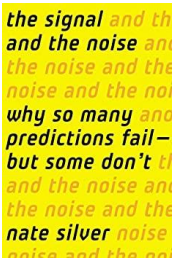
By Nate Silver

Filed under Housekeeping

One of FiveThirtyEight's goals has always been to get people to think more carefully about probability. When we're forecasting an upcoming election or sporting event, we'll go to great lengths to analyze and explain the sources of real-world uncertainty and the extent to which events — say, a Senate race in Texas and another one in Florida — are correlated with one another. We'll spend a lot of time working on how to build robust models that don't suffer from p-hacking or overfitting and which will perform roughly as well when we're making *new* predictions as when we're backtesting them. There's a lot of science in this, as well as a lot of art. We really care about the difference between a 60 percent chance and a 70 percent chance.

That's not always how we're judged, though. Both our fans and our critics sometimes look at our probabilistic forecasts as *binary predictions*. Not only might they not care about the difference between a 60 percent chance and a 70 percent chance, they sometimes treat a 55 percent chance the same way as a 95 percent one.

- One person you should read if you care about prediction is Nate Silver



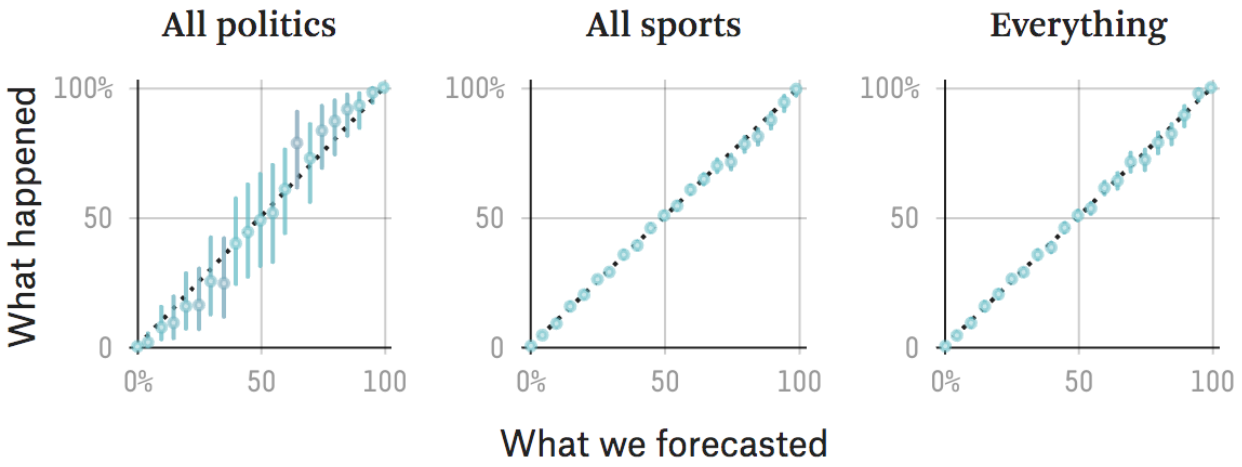
Calibration measures whether, over the long run, events occur about as often as you say they're going to occur. For instance, of all the events that you forecast as having an 80 percent chance of happening, they should indeed occur about 80 out of 100 times; that's good calibration. If these events happen only 60 out of 100 times, you have problems — your forecasts aren't well-calibrated and are *overconfident*. **But it's just as bad if they occur 98 out of 100 times, in which case your forecasts are *underconfident*.**

https://fivethirtyeight.com/features/when-we-say-70-percent-it-really-means-70-percent/

Are Nate Silver's Forecasts Well Calibrated?

Forecast calibration for FiveThirtyEight "polls-only" forecast

WIN PROBABILITY RANGE	FORECASTS	EXPECTED WINNERS	ACTUAL WINNERS
95-100%	31	30.5	30
75-94%	15	12.4	13
50-74%	11	6.9	9
25-49%	12	4.0	2
5-24%	22	2.4	1
0-4%	89	0.9	1



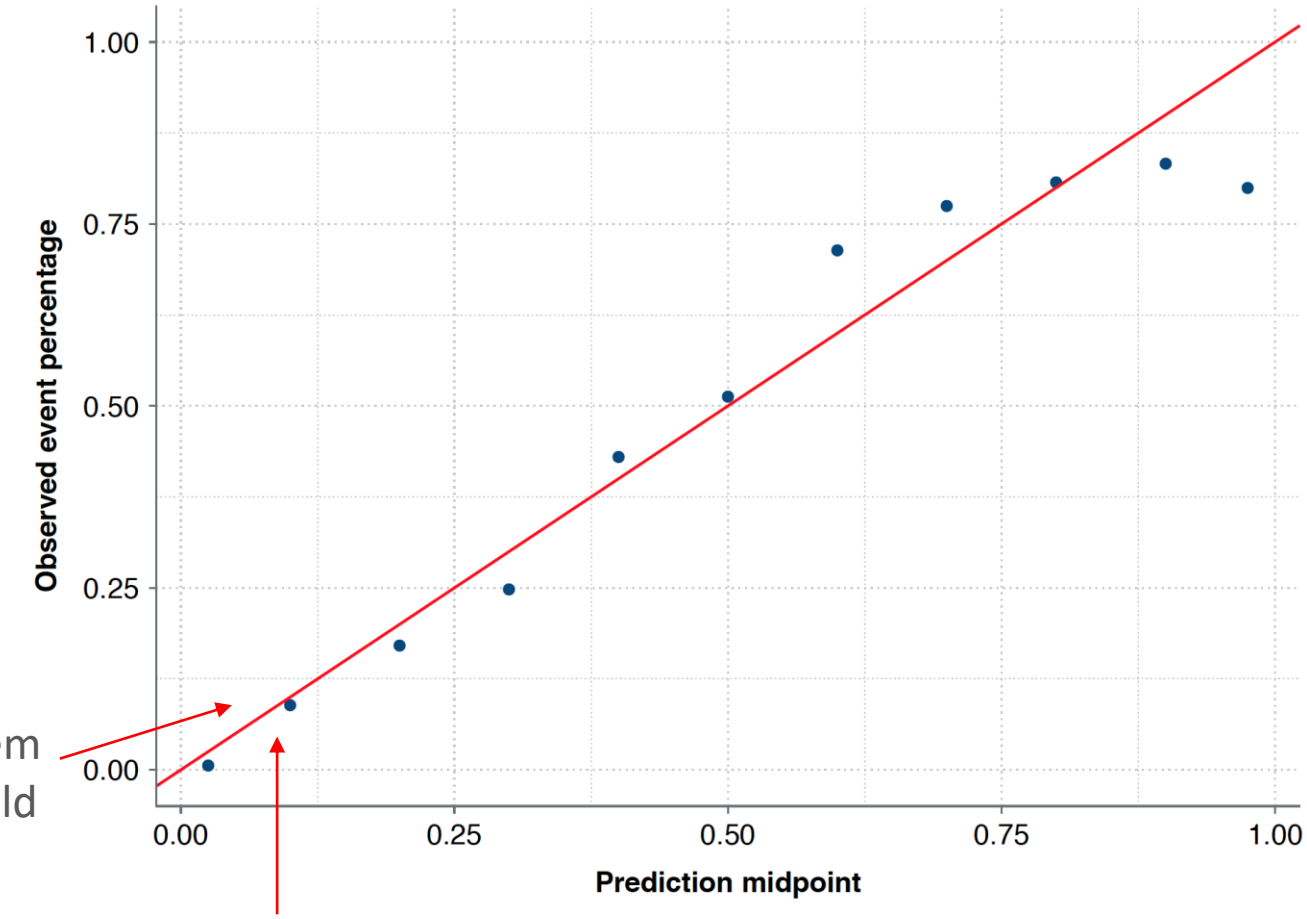
- 95% probability of win means out of 20 you forecast with 95% probability, 19 should win!
- 10% probability of win means out of 20 forecasts with 10% probability 2/20 should win!
- Overall Nate Silver is well calibrated, but politics is more noisy!

https://fivethirtyeight.com/features/when-we-say-70-percent-it-really-means-70-percent/

Calibration plot

- Idea of a calibration plot:

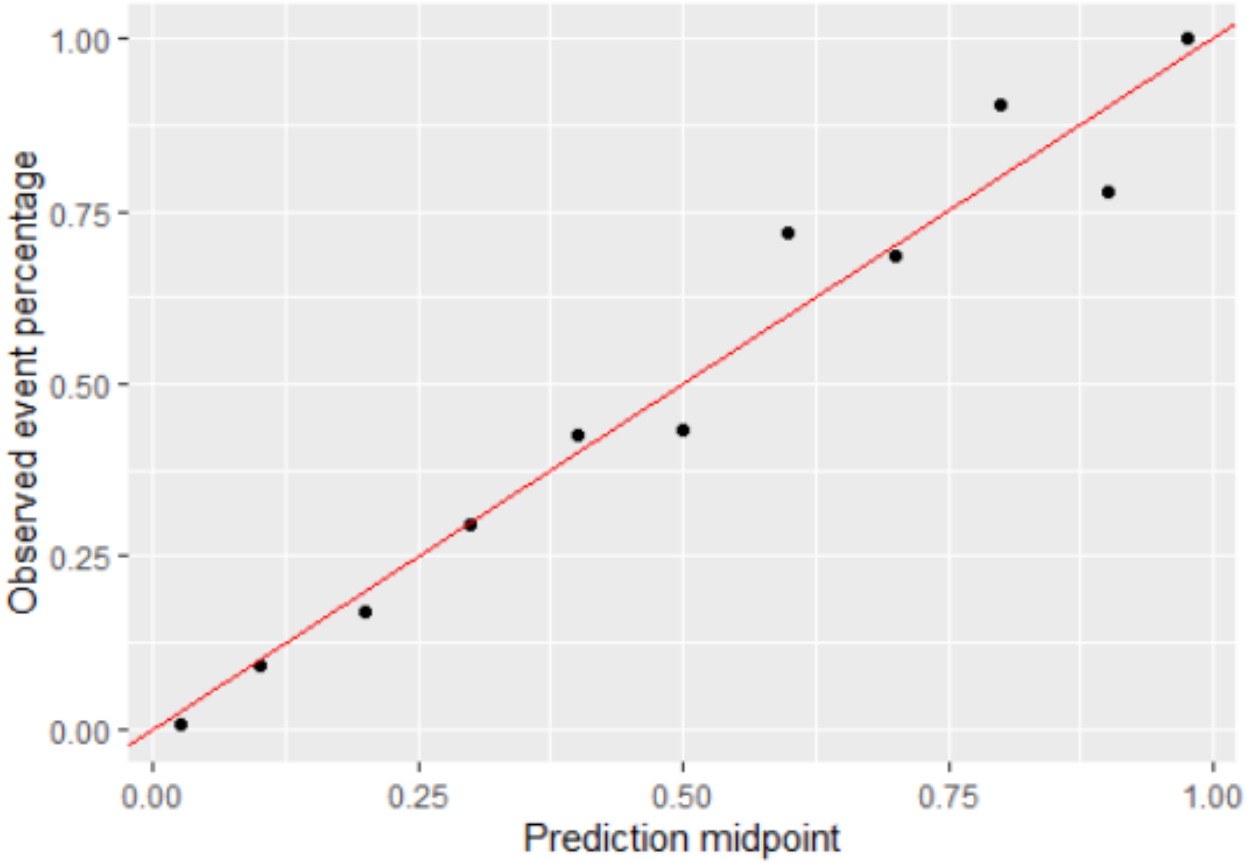
- A well-calibrated prediction is one where if we give an X% probability of an event occurring, X/100 of the time the event happens.



For 1/10 of them the event should actually occur

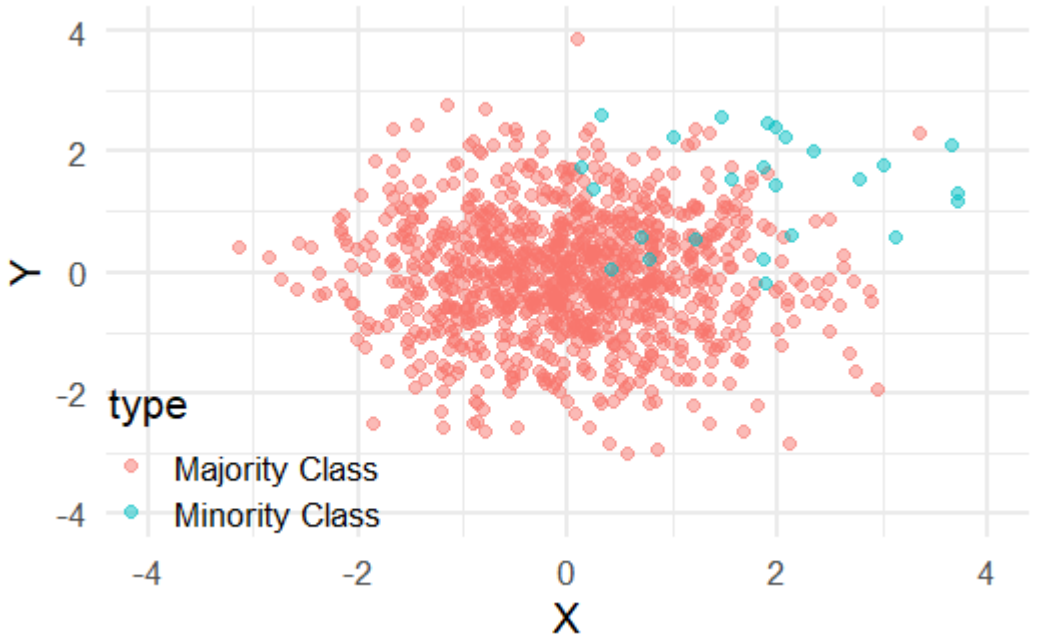
All obs where we predict 10% chance of event occurring

Calibration plot




```r
# calibration chart

scores3DF <- data.frame(default =
                          ifelse(Default$default == "Yes",1,0),
                        scores =
                          predict(logit_fit3, type = "response"))
library(plyr)
calData <- ddply(scores3DF, .(cut(scores3DF$scores,
                                  c(0,0.05,0.15,0.25,0.35,0.45,
                                    0.55,0.65,0.75,0.85,0.95,1))),
                 colwise(mean))
calData$midpoint <- c(0.025,.1,.2,.3,.4,.5,.6,.7,.8,.9,.975)
colnames(calData) <- c("preds", "true", "midpoint")
calPlot <- ggplot(calData, aes(x = midpoint, y = true)) +
  geom_point() + ylim(0,1) +
  geom_abline(intercept = 0, slope = 1, color = "red") +
  xlab("Prediction midpoint") + ylab("Observed event percentage")
plot(calPlot)
```

What Is Class Imbalance?



- Class Imbalance occurs with classification models where one class severely outnumbers the other class(es)
- What is significant imbalance? Anything less than ~10-5%
- This creates problems for **any** classification algorithm. Even AI isn't immune!

Severe Class Imbalance: Why Better Algorithms Aren't the Answer

Chris Drummond[1] and Robert C. Holte[2]

[1] Institute for Information Technology, National Research Council Canada, Ottawa, Ontario, Canada, K1A 0R6 Chris.Drummond@nrc-cnrc.gc.ca
[2] Department of Computing Science, University of Alberta, Edmonton, Alberta, Canada, T6G 2E8 holte@cs.ualberta.ca

Abstract. This paper argues that severe class imbalance is not just an interesting technical challenge that improved learning algorithms will address, it is much more serious. To be useful, a classifier must appreciably outperform a trivial solution, such as choosing the majority class. Any application that is inherently noisy limits the error rate, and cost, that is achievable. When data are normally distributed, even a Bayes optimal classifier has a vanishingly small reduction in the majority classifier's error rate, and cost, as imbalance increases. For fat tailed distributions, and when practical classifiers are used, often no reduction is achieved.

Monitoring War Destruction from Space: A Machine Learning Approach

Hannes Mueller[a,b,1], **Andre Groeger**[b,c,1], **Jonathan Hersh**[d,2], **Andrea Matranga**[d,e,2], **and Joan Serrat**[f,1]

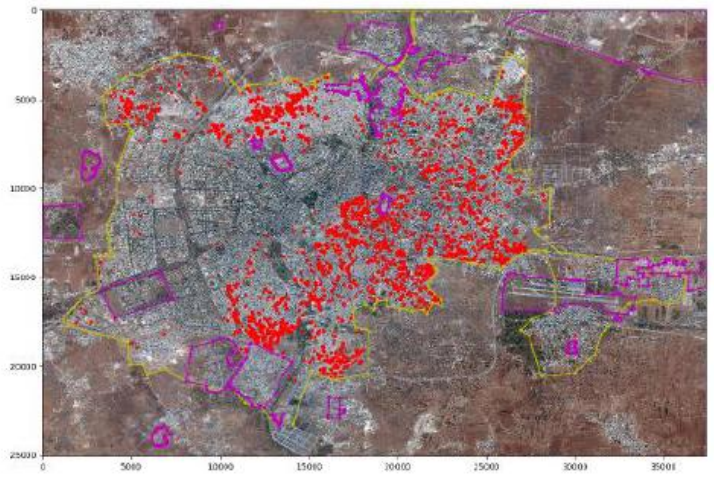
[a]Institute of Economic Analysis (IAE-CSIC), 08193 Bellaterra, Barcelona, Spain; [b]Barcelona Graduate School of Economics (BGSE), 08005 Ba
Economics and Economic History, Universitat Autònoma de Barcelona (UAB), 08193 Bellaterra, Spain; [d]Argyros School of Business, Chapma
USA; [e]Smith Institute for Political Economy and Philosophy, Orange, CA 92868 USA; [f]Computer Science Department and Computer Vision C
Barcelona (UAB), 08193 Bellaterra, Spain

This manuscript was compiled on October 2, 2020

Existing data on building destruction in conflict zones rely on eyewitness reports or manual detection, which makes it generally scarce, incomplete and potentially biased. This lack of reliable data imposes severe limitations for media reporting, humanitarian relief efforts, human rights monitoring, reconstruction initiatives, and academic studies of violent conflict. This article introduces an automated method of measuring destruction in high-resolution satellite images using deep learning techniques combined with data augmentation to expand training samples. We apply this method to the Syrian civil war and reconstruct the evolution of damage in major cities across the country. The approach allows generating destruction data with unprecedented scope, resolution, and frequency - only limited by the available satellite imagery - which can alleviate data limitations decisively.

Destruction | Conflict | Deep Learning | Remote Sensing | Syria

- My own work: finding which buildings have been bombed using machine learning applied to satellite imagery is hard because most buildings are not bombed, even in Syria!

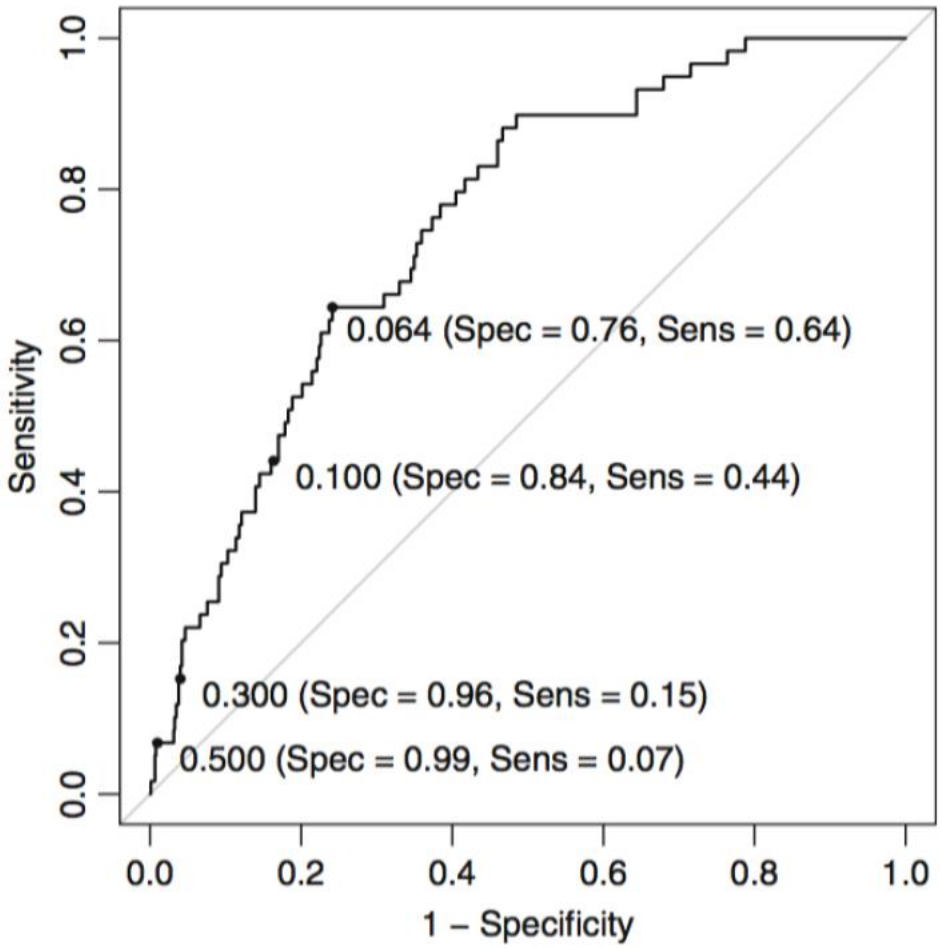


(a) Full Extent of Aleppo, Syria.

(b) Zoom Showing Destruction is Sparse

Fig. 1. Imagery of Aleppo 09/18/2016. Red dots indicate UNOSAT annotations as *destroyed*. The regions enclosed in magenta are *no analysis zones*, excluded from the UNOSAT damage assessment due to being non-civilian. The yellow lines indicates the boundary of populated areas in Aleppo under analysis. Sources: Google Earth/Maxar Technologies and UNITAR/UNOSAT.

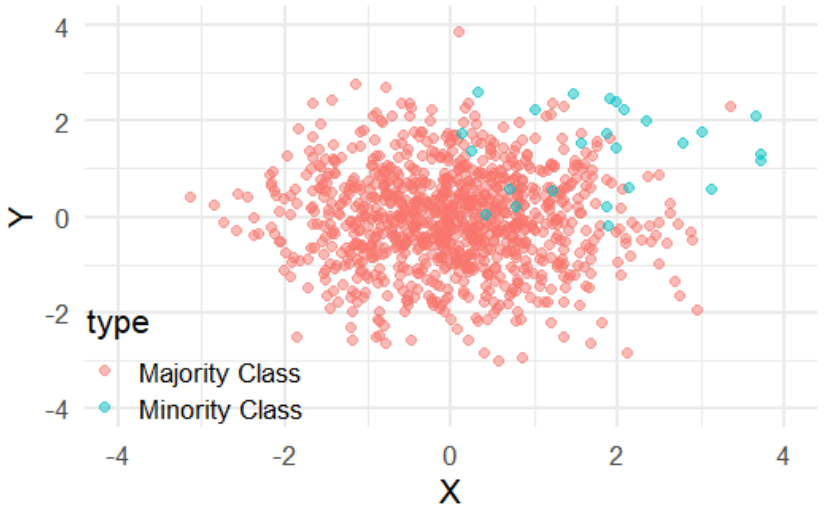
Remedy 1: Alternative Class Threshold



- First solution may be to choose a different probability threshold
- Each threshold from the ROC plot shows a different resulting specificity or sensitivity
- For applications we care more about sensitivity (low FNs), for others we care are specificity (low FPs)

Other Remedies: Up-Sampling and Down-Sampling

- Down-Sampling creates an artificial data set with equal minority and majority cases



Down-Sampling

Up-Sampling

- Up-Sampling creates an artificial data set with more (repeated copies of) the minority class