Classifier Evaluation and Selection

Review and Overview of Methods

Considerations for Evaluating Classification Models

- Interpretation vs. Prediction

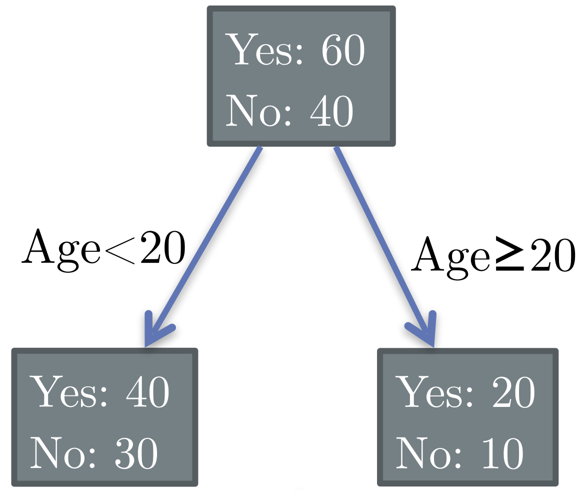
  (Model Parsimony vs. Model Error)

- Type of prediction goal:
  - **Decisions** – Interested only in **resulting classification (ex: 'Yes'/'No')** *pick out all the winning proposals*
  - **Rankings** – Interested in **ranking individuals by their 'true likelihood'** of an outcome - *who are the best 10% to market to*
  - **Estimates** – Interested in **predicting probabilities** or a continuous outcome accurately - *compute expected annual cost of each machine using failure probabilities*

Model Fit Statistics Summary

Prediction Type	Model Fit Statistics
Decisions	Lift/Gain/Profit/Loss Accuracy/ Misclassification KS-Statistic
Rankings	ROC Index concordance statistic Gini Coefficient
Estimates	Average Squared Error SBC/Likelihood MAPE R^2

Practical Difference



I(t) is **<u>misclassification rate</u>**

Parent misclassification rate: 40%

Misclassification rate after split: 40%

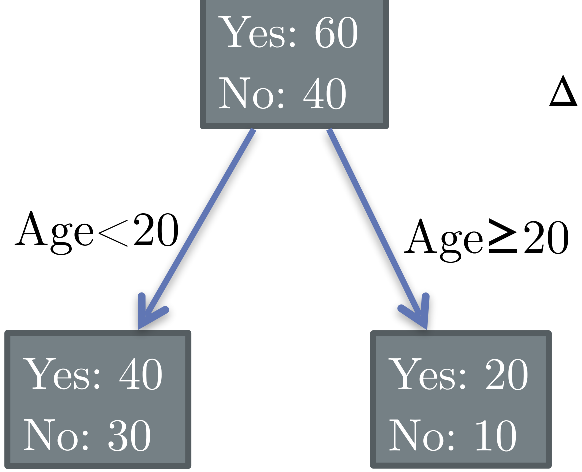
Gain: **0** => Don't make this split.

I(t) is **<u>Average Squared Error</u>**

Parent averaged squared error: 0.24

Average squared error after split: 0.23

Gain: **0.01** => Consider this split.



$$\Delta = I(t) - \left(\frac{n_L}{n} I(t_L) + \frac{n_R}{n} I(t_R)\right)$$

I(t) is **misclassification rate**

Parent misclassification rate: 40%

Misclassification rate after split: 40%

Gain: **0** => Don't make this split.

Details:

Left child misclass. rate: 42%

Right child misclass. rate: 33.3%

$$\text{Gain} = 0.40 - \left(\frac{70}{100}0.42 + \frac{30}{100}0.33\right) = 0$$

I(t) is **Average Squared Error**

Parent averaged squared error: 0.24

Average squared error after split: 0.23

Gain: **0.01** => Consider this split.

Details:

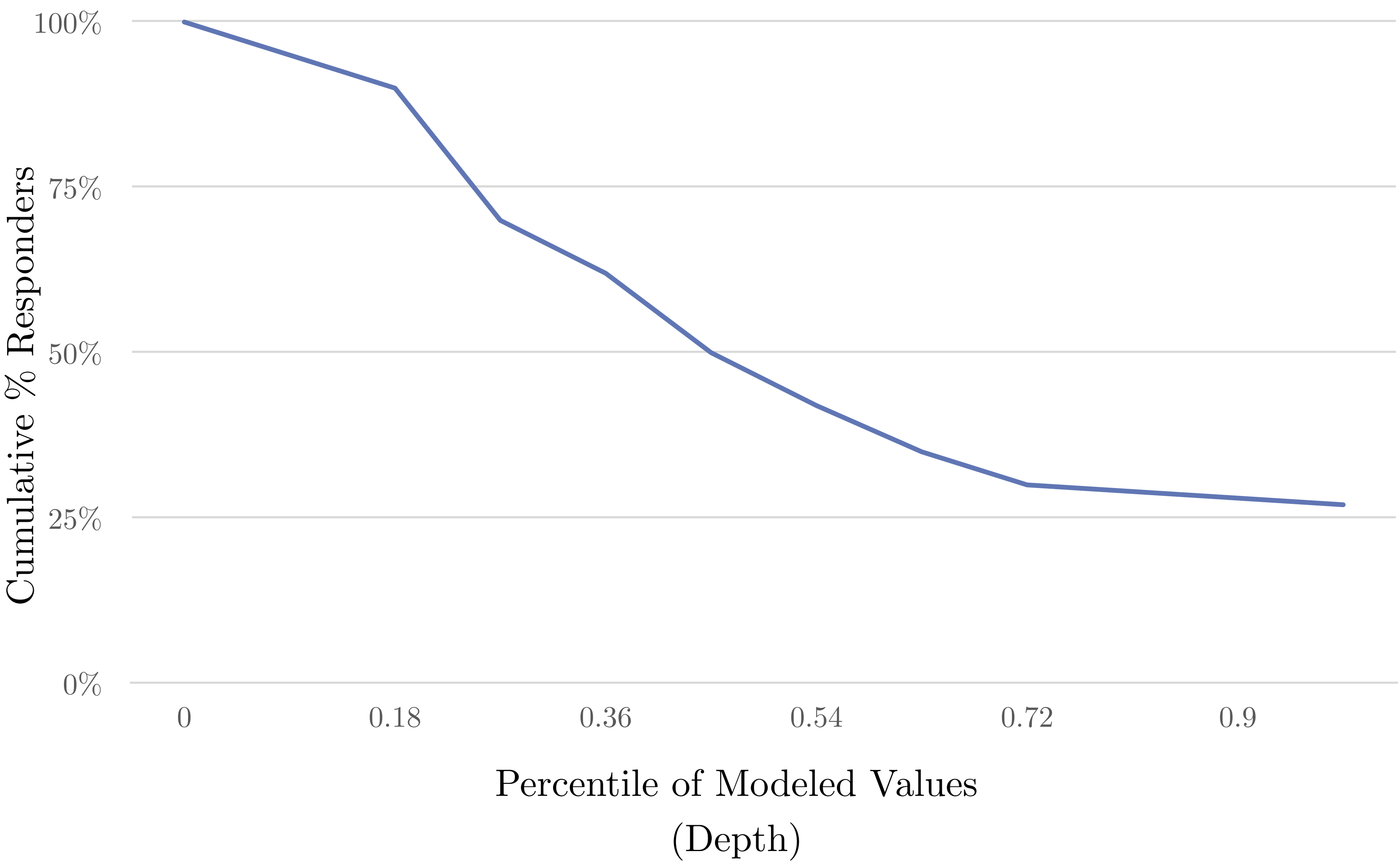
Parent ASE: $\frac{1}{100}\left(60\,(1-0.6)^2 + 40\,(0-0.6)^2\right)$

Left child ASE: $\frac{1}{70}\left(40\left(1-\frac{4}{7}\right)^2 + 30\left(0-\frac{4}{7}\right)^2\right)$

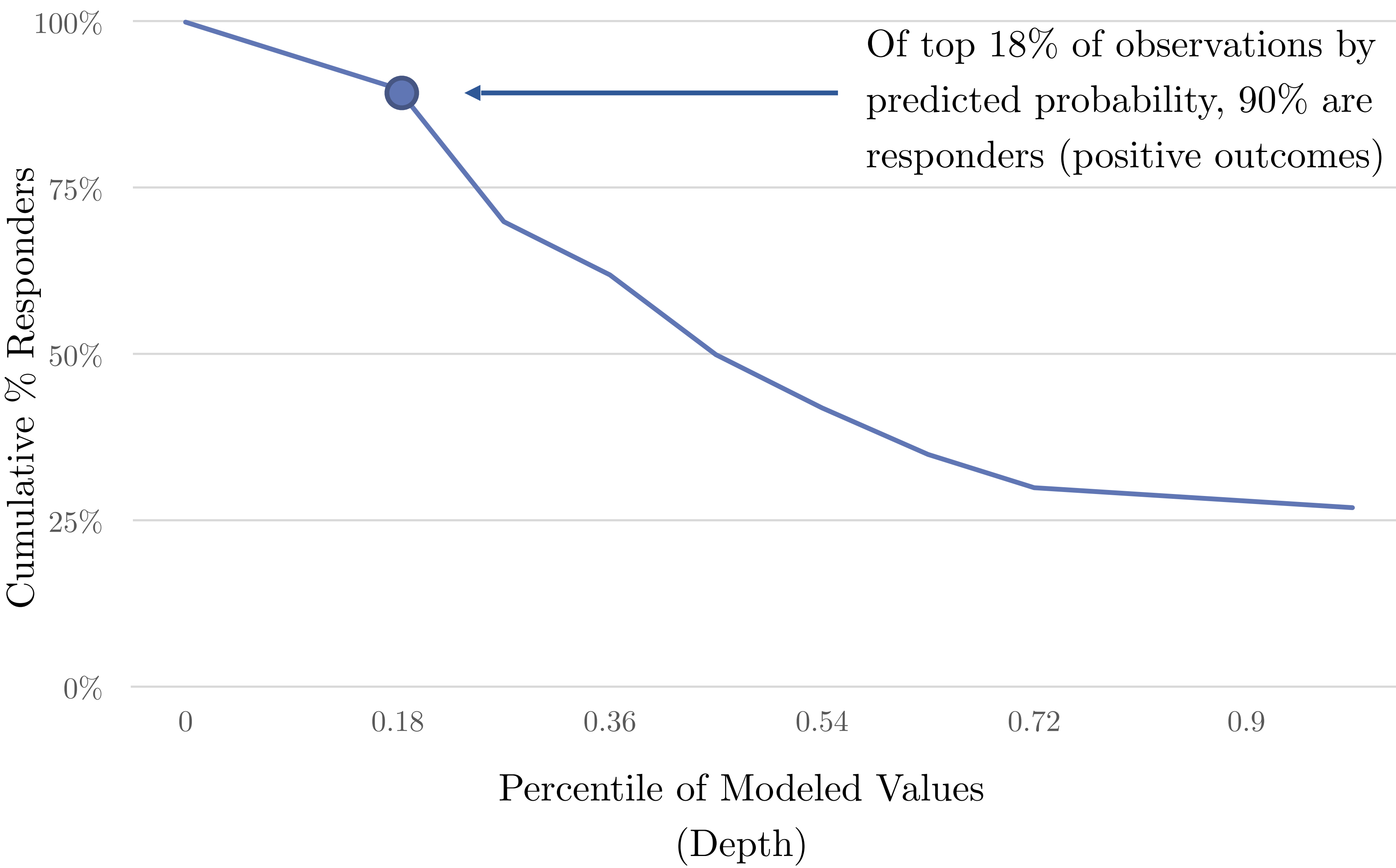
Right child ASE: $\frac{1}{30}\left(20\left(1-\frac{2}{3}\right)^2 + 10\left(0-\frac{2}{3}\right)^2\right)$

$$\text{Gain} = 0.24 - \left(\frac{70}{100}0.245 + \frac{30}{100}0.222\right) = 0.01$$

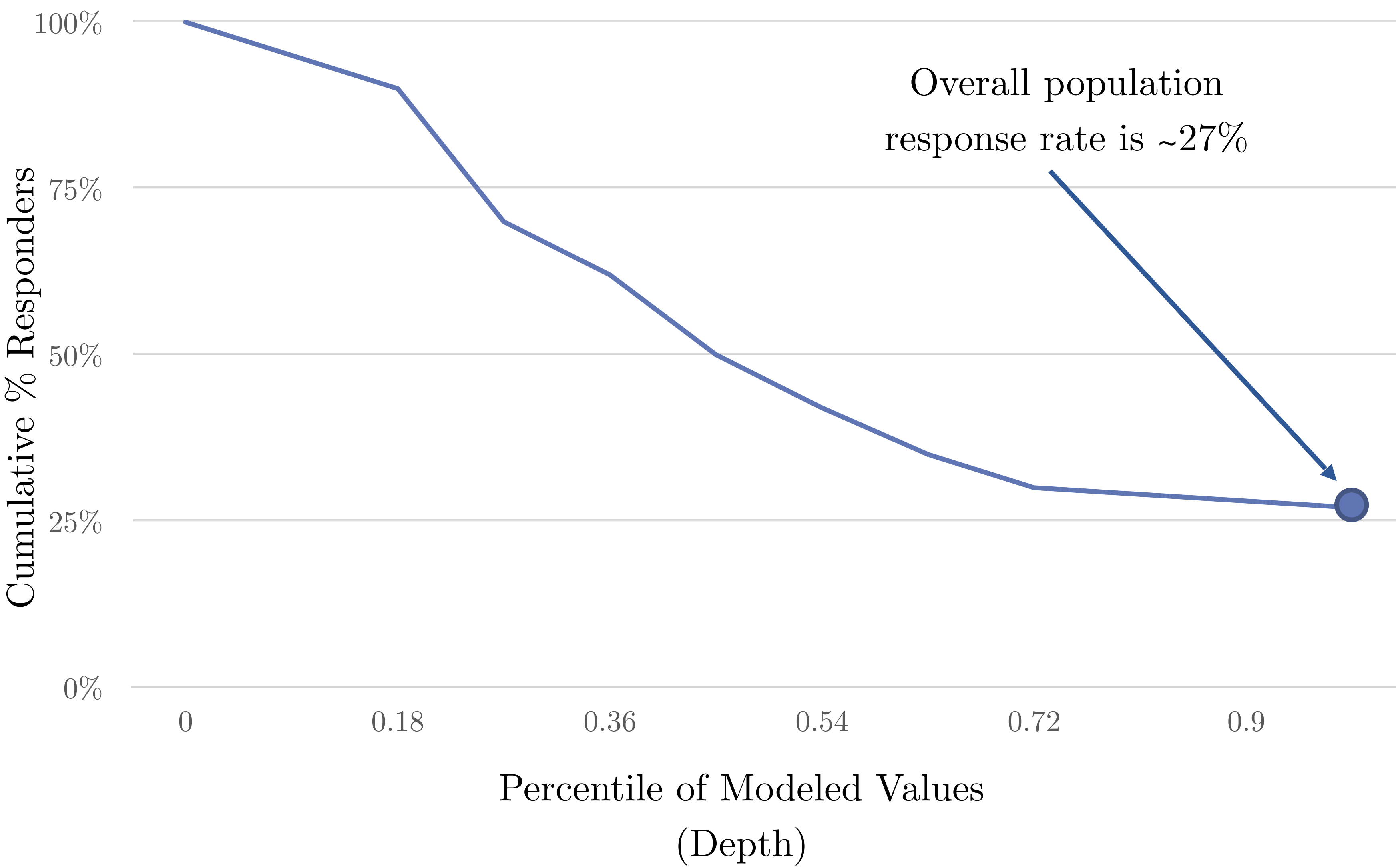
Response/Gain Charts



Response/Gain Charts



Response/Gain Charts

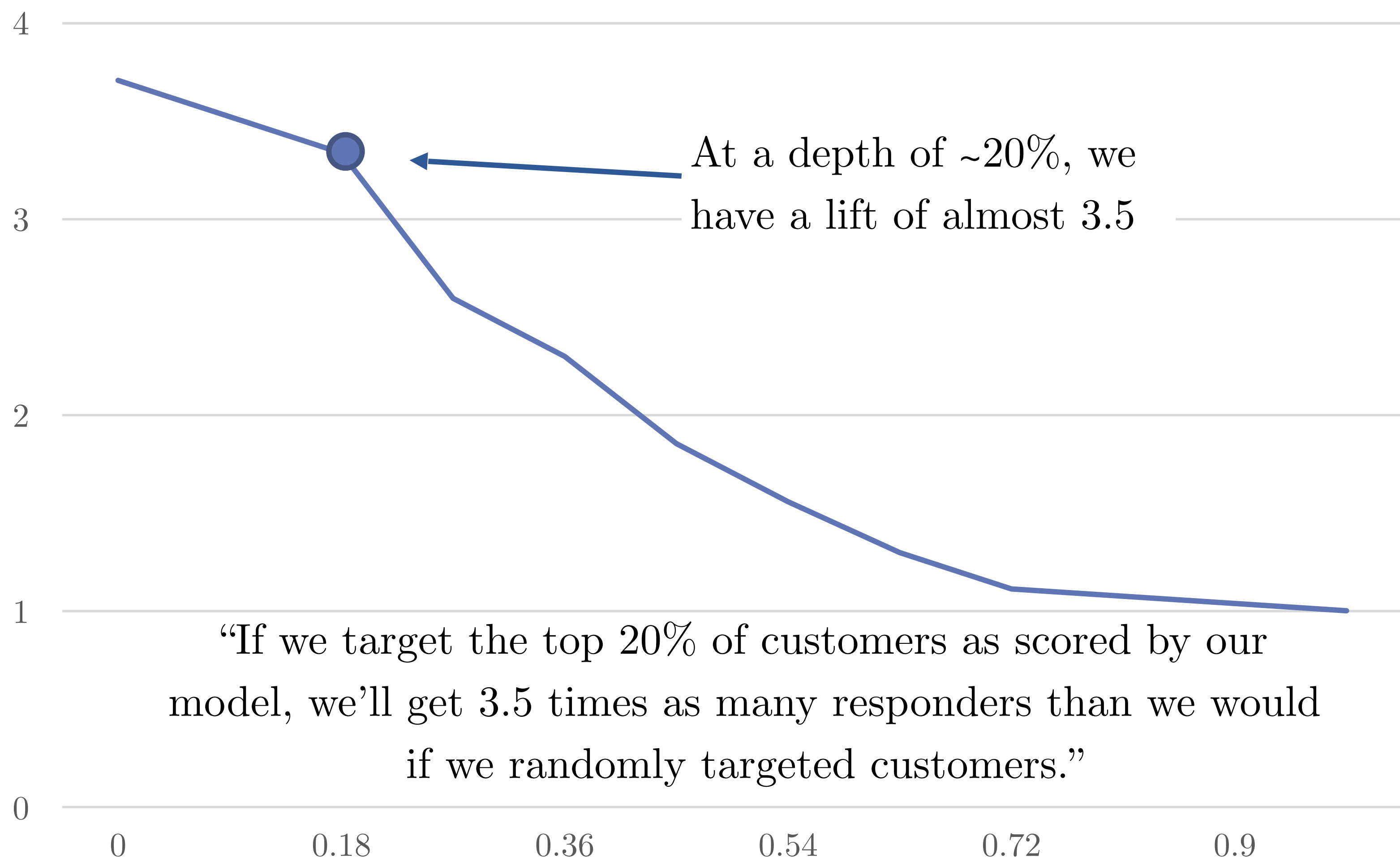


Lift Chart

While it's great to know how many responders you got in the top p% of observations scored by the model, it's *even better* to **know how your model compares to random selection.**

$$\text{Lift} = \frac{\%\ Responders\ from\ Model}{\%\ Responders\ from\ Random\ Selection}$$

Cumulative Lift



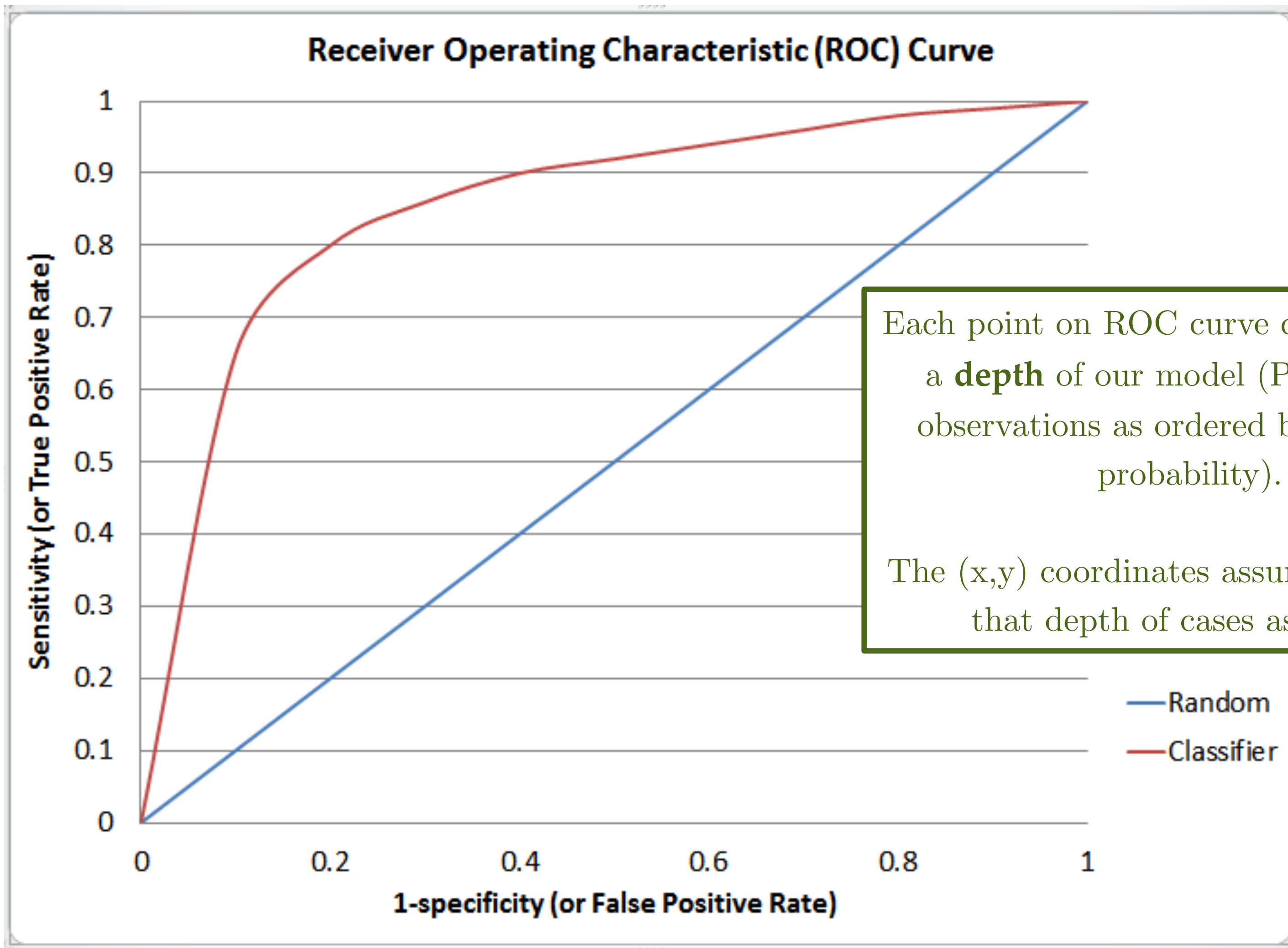
Confusion Matrix

Confusion Matrix		Target			
		Positive	Negative		
Model	Positive	a	b	*Positive Predictive Value*	a/(a+b)
	Negative	c	d	*Negative Predictive Value*	d/(c+d)
		Sensitivity	*Specificity*	**Accuracy** = (a+d)/(a+b+c+d)	
		a/(a+c)	d/(b+d)		

Metrics from Confusion Matrix:

1. Accuracy: Proportion of total predictions that were correct
2. Precision/ Positive Predictive Value: Proportion of predicted positive that were actually positive
3. Negative Predictive Value: Proportion of predicted negative that were actually negative
4. Sensitivity/Recall: Proportion of actual positive cases correctly identified
5. Specificity: Proportion of actual negative cases which are correctly identified

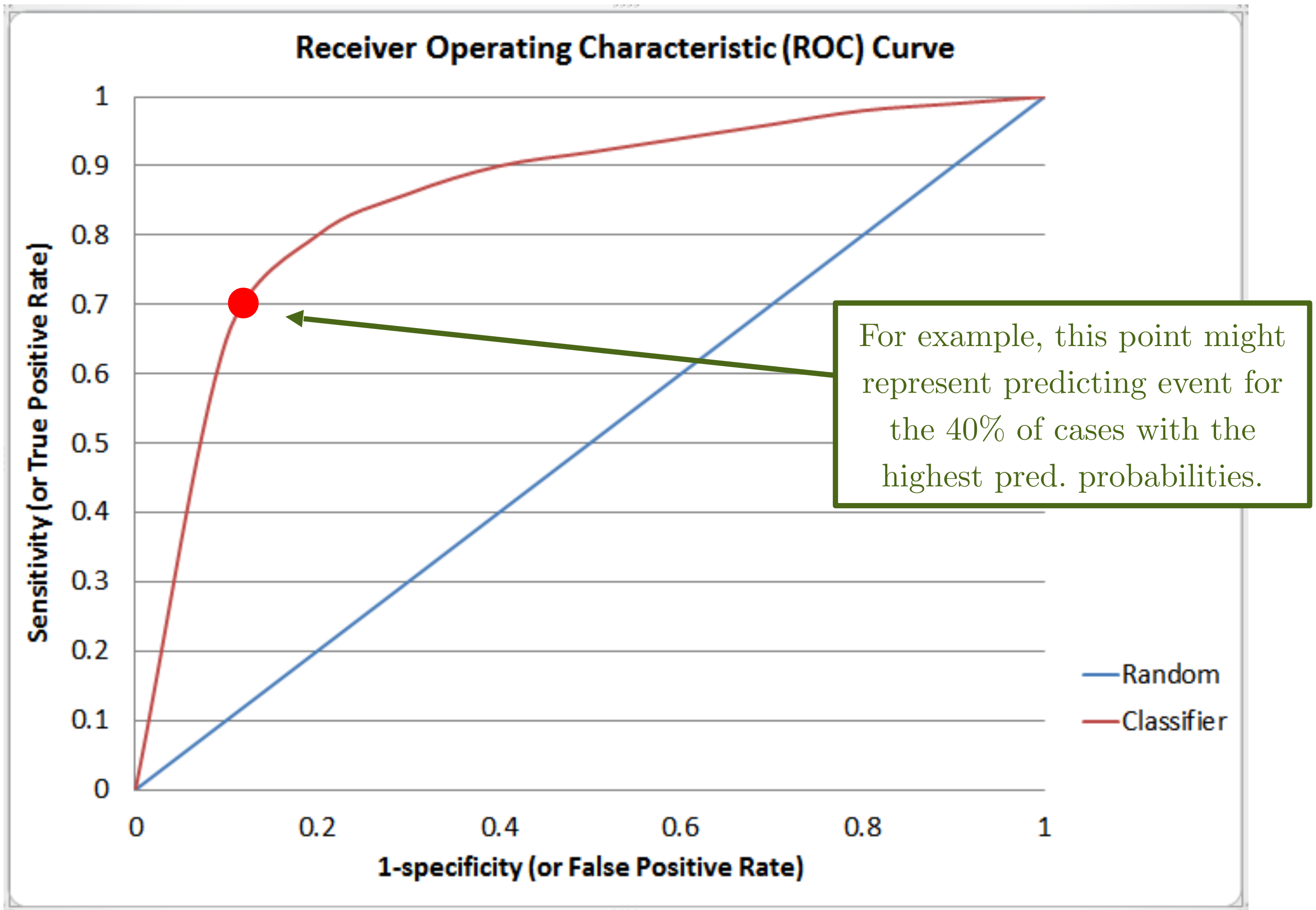
ROC Charts



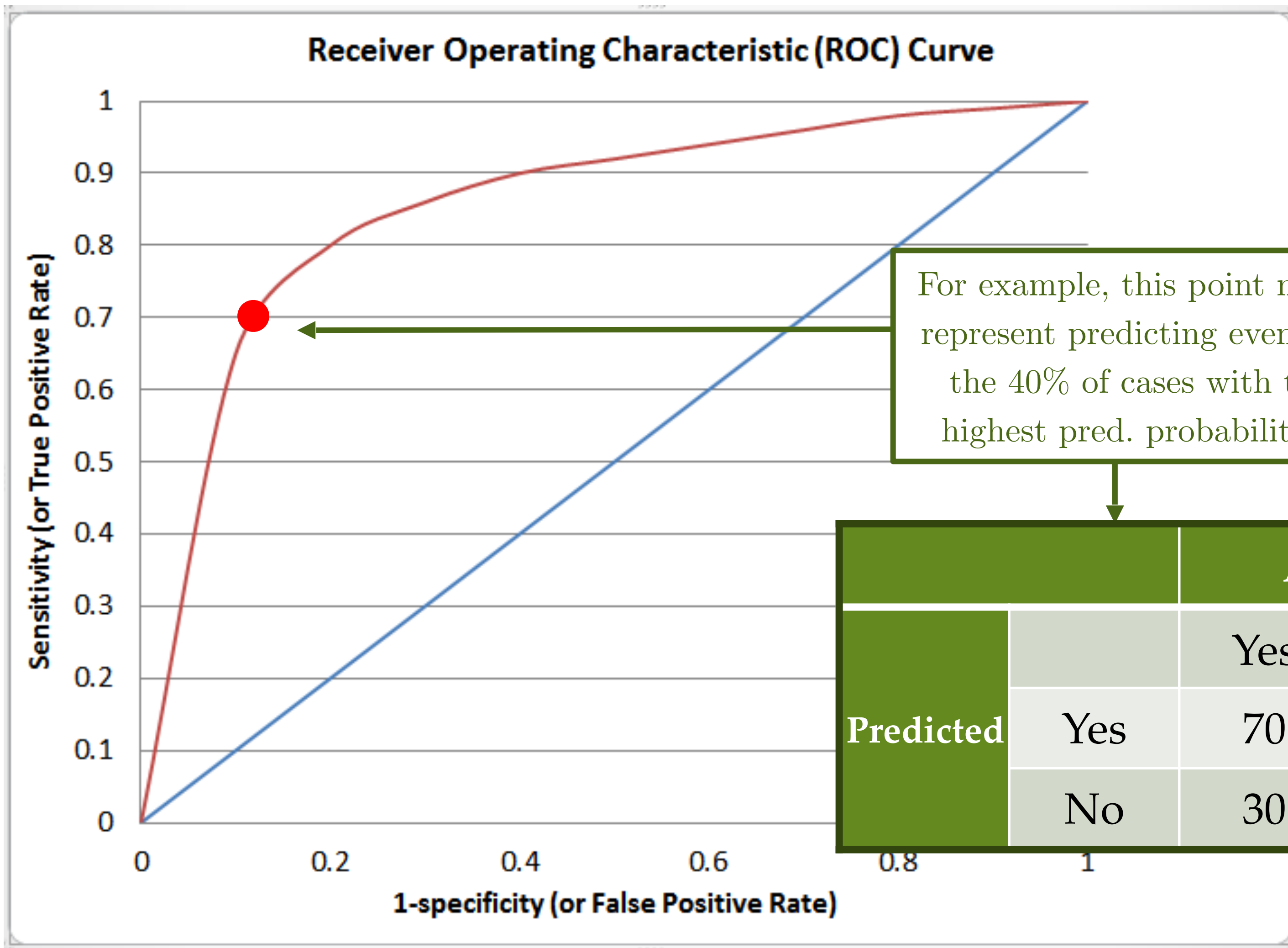
Each point on ROC curve corresponds to a **depth** of our model (Percentile of observations as ordered by predicted probability).

The (x,y) coordinates assume we predict that depth of cases as events.

ROC Charts


Receiver Operating Characteristic (ROC) Curve
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Sensitivity (or True Positive Rate)
0
0.2
0.4
0.6
0.8
1
1-specificity (or False Positive Rate)
Random
Classifier
For example, this point might represent predicting event for the 40% of cases with the highest pred. probabilities.

ROC Charts



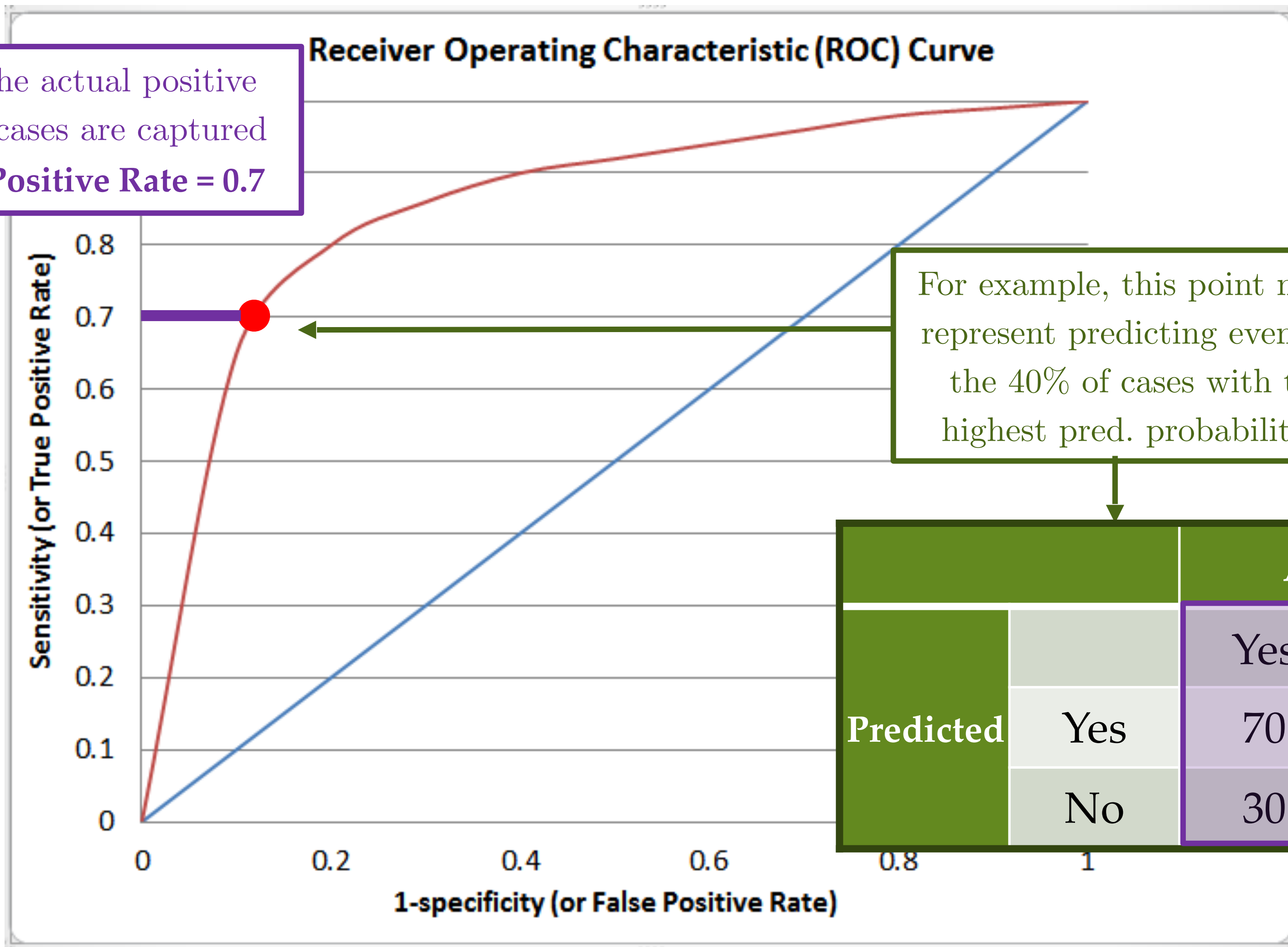
For example, this point might represent predicting event for the 40% of cases with the highest pred. probabilities.

		Actual	
		Yes	No
Predicted	Yes	70	10
	No	30	90

ROC Charts

70% of the actual positive outcome cases are captured

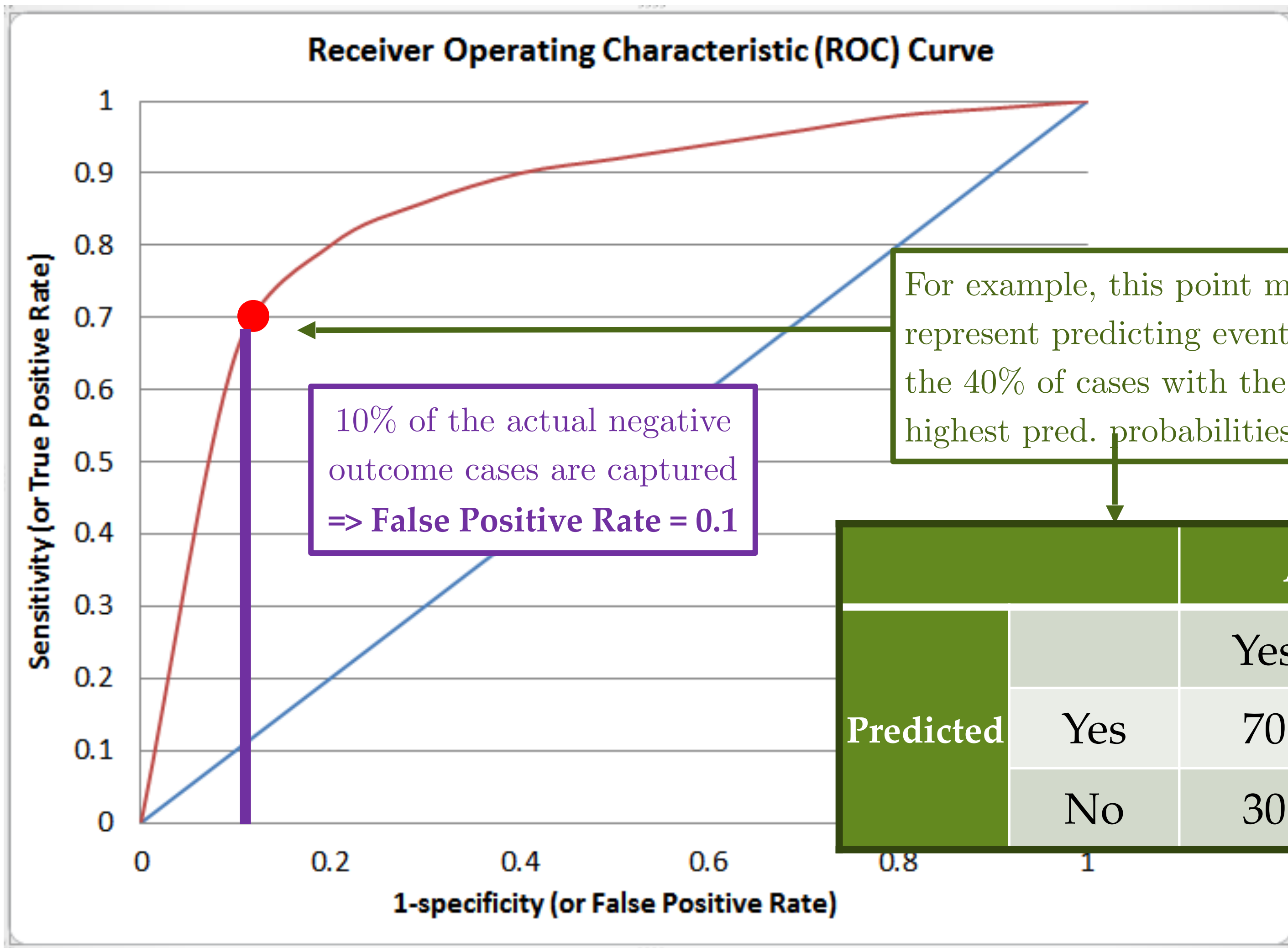
=> True Positive Rate = 0.7



For example, this point might represent predicting event for the 40% of cases with the highest pred. probabilities.

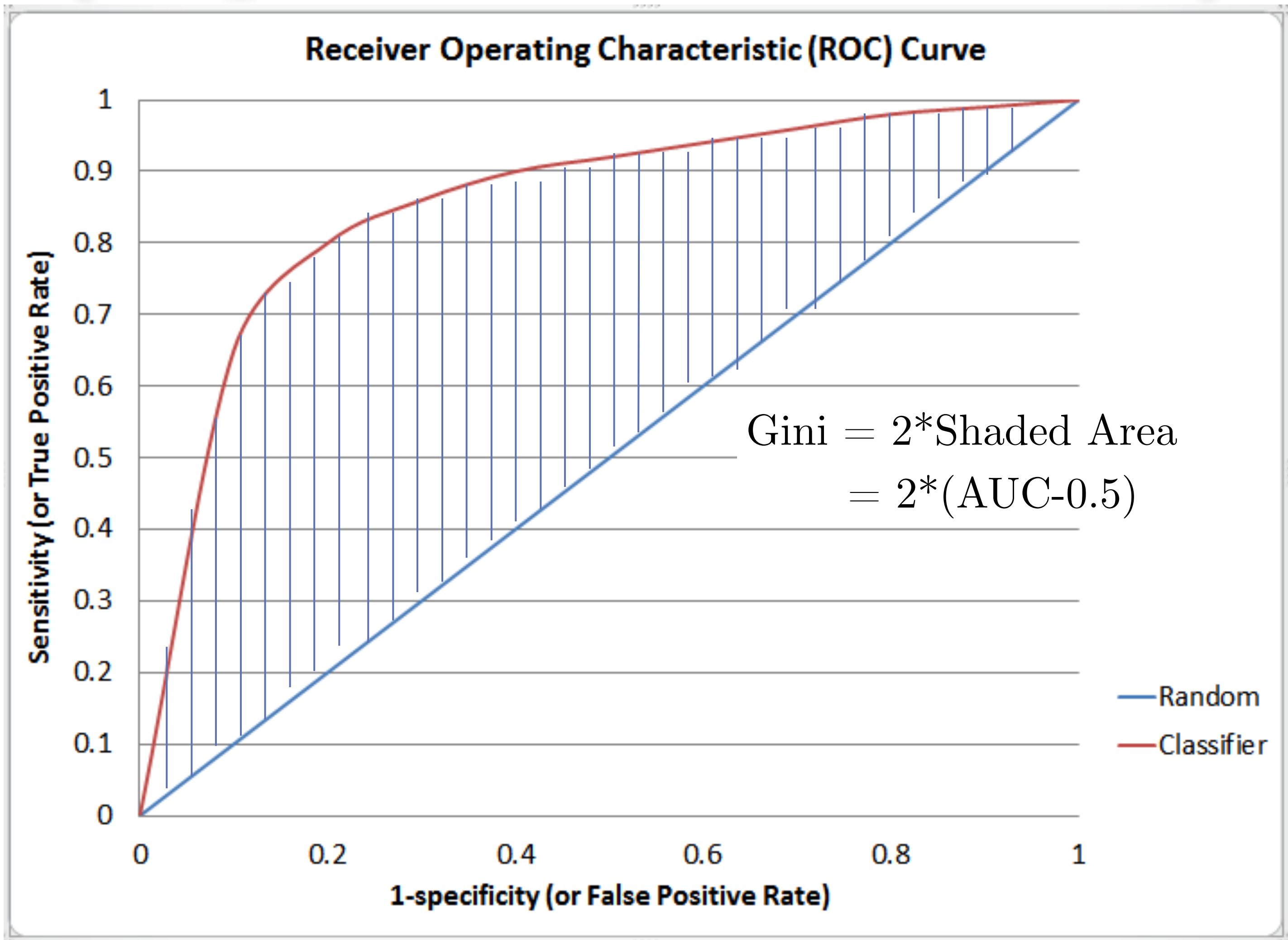
		Actual	
		Yes	No
Predicted	Yes	70	10
	No	30	90

ROC Charts

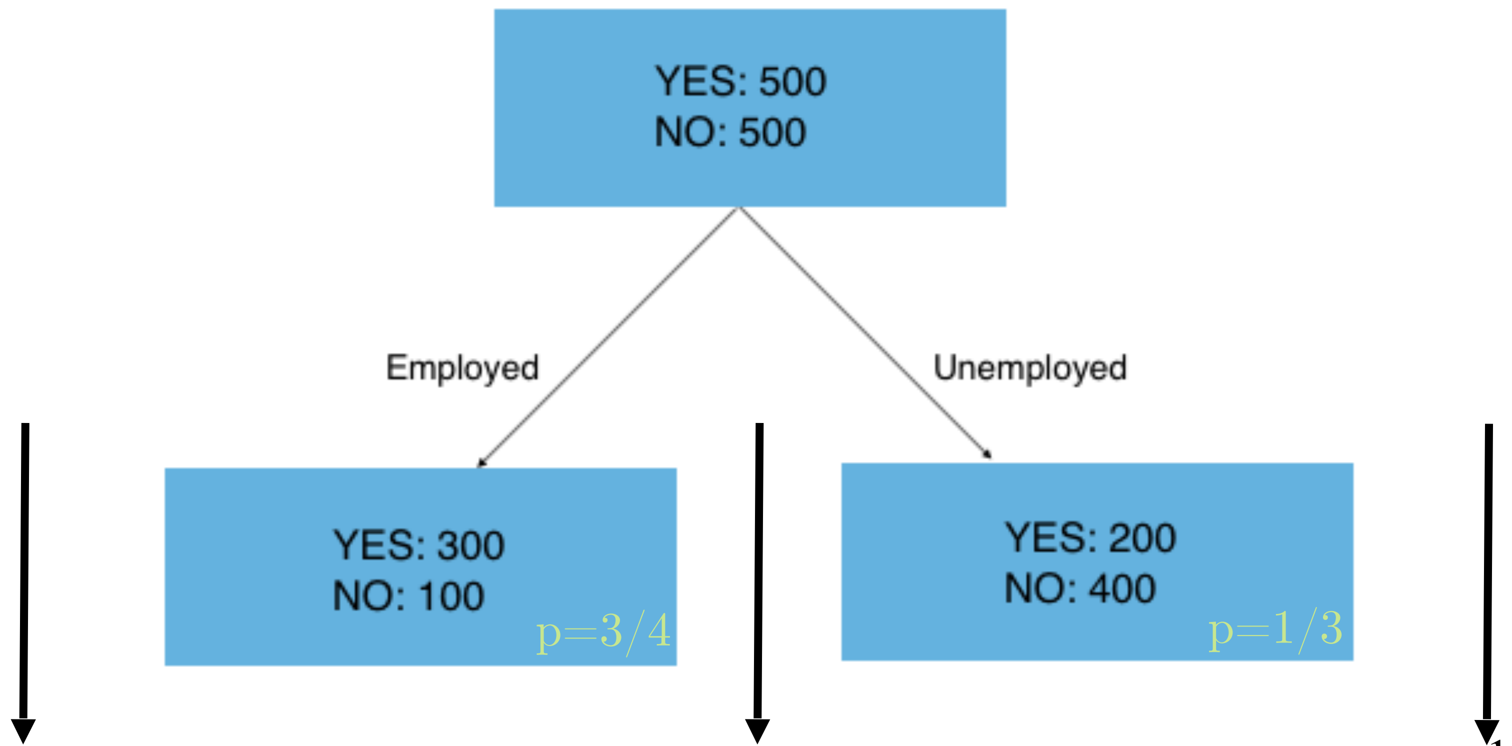


		Actual	
		Yes	No
Predicted	Yes	70	10
	No	30	90

Gini Coefficient (Equivalent to AUC)



ROC Charts for Decision Trees



$p_{cutoff} = 1$

TPR $= 0$

FPR $= 0$

$\frac{1}{3} < p_{cutoff} < \frac{3}{4}$

TPR $= 0.6$

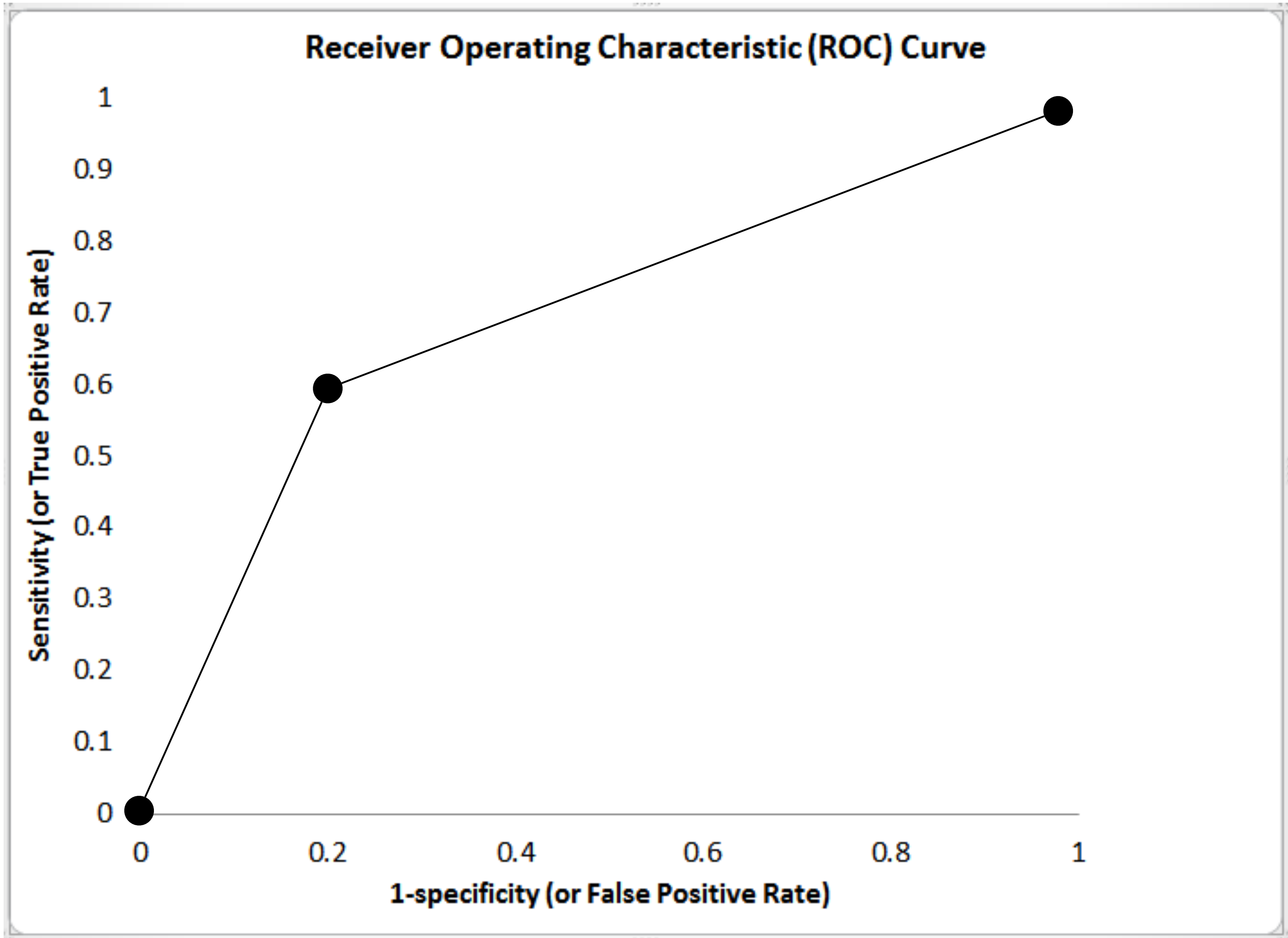
FPR $= 0.2$

$p_{cutoff} < \frac{1}{3}$

TPR $= 1$

FPR $= 1$

ROC Charts for Decision Trees



Average Squared Error (ASE)

$$\frac{1}{nL}\sum_{i=1}^{n}\sum_{j=1}^{L}(y_{ij}-\hat{y}_{ij})^2$$

- **Computes sum of squared error between probabilities and binary (0/1) target.**
- For class targets, let L be the number of levels in the target.
- This objective function sets $y_{ij} = 1$ if observation i takes level j of the target and 0 otherwise.

Average Squared Error (ASE)

$$\frac{1}{nL}\sum_{i=1}^{n}\sum_{j=1}^{L}(y_{ij}-\hat{y}_{ij})^2$$

Example:

Name	P(red)	P(blue)	P(none)		Actual
JimBob	0.3	0.4	0.3		BLUE
BillyBob	0.1	0.5	0.4		NONE

Average Squared Error (ASE)

$$\frac{1}{nL}\sum_{i=1}^{n}\sum_{j=1}^{L}(y_{ij}-\hat{y}_{ij})^2$$

Example:

Name	P(red)	P(blue)	P(none)		Actual
JimBob	0.3	0.4	0.3		BLUE
BillyBob	0.1	0.5	0.4		NONE

P(red)	P(blue)	P(none)
0	1	0

Average Squared Error (ASE)

$$\frac{1}{nL}\sum_{i=1}^{n}\sum_{j=1}^{L}(y_{ij}-\hat{y}_{ij})^2$$

Example:

Name	P(red)	P(blue)	P(none)		Actual
JimBob	0.3	0.4	0.3		BLUE
BillyBob	0.1	0.5	0.4		NONE

$$\frac{(0-0.3)^2+(1-0.4)^2+(0-0.3)^2 \quad + \quad (0-0.1)^2+\left(0-0.5\right)^2+(1-0.4)^2}{2*3}$$

Average Squared Error (ASE)

$$\frac{1}{nL}\sum_{i=1}^{n}\sum_{j=1}^{L}(y_{ij}-\hat{y}_{ij})^2$$

Example:

Name	P(red)	P(blue)	P(none)		Actual
JimBob	0.3	0.4	0.3		BLUE
BillyBob	0.1	0.5	0.4		NONE

$$\frac{(0-0.3)^2+(1-0.4)^2+(0-0.3)^2 \quad + \quad (0-0.1)^2+\left(0-0.5\right)^2+(1-0.4)^2}{2*3}$$

Things Customers Say

- "We need a model that is accurate when it signals an event is coming - false positives can cause unpredictable losses."

  Lift at Depth

  Positive Predicted Value

- "We need a model that sorts out group A from group B as best as possible."

  Misclassification Rate

  K-S Statistic

- "We need to develop a risk score to measure a client's likelihood of default."

  Log Likelihood

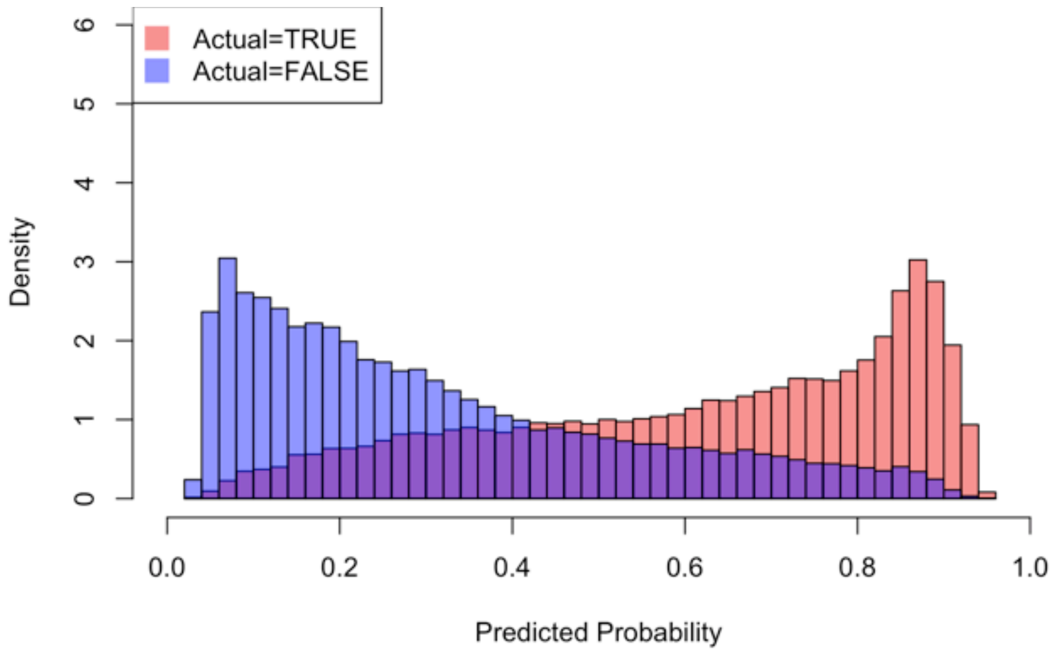
  Average Squared Error

- "We want to rank our machines in terms of failure likelihood so we can rotate through daily maintenance in a logical ordering."

  AUC

  c-statistic

Other Visual Exploration



Plot the distribution of predicted probabilities for each level of the target value.

We'd want these distributions to look as distinct as possible.

Here I used overlaid histogram with transparent colors so you can *see* both distributions.

In case you want to steal my picture:

```
hist(test$pred.probs[test[,"target"]==1], breaks=50, freq=F,   xlim=c(0,1),ylim=c(0,9), col=rgb(1,0,0,0.5),
xlab="Predicted Probability", ylab="Density", main="Test Data Distribution of Predicted Prob. by Actual Outcome" )

hist(test$pred.probs[test[,"target"]==0], breaks=50,freq=F, xlim=c(0,1),ylim=c(0,9), col=rgb(0,0,1,0.5), xlab="",
ylab="Density", add=T)

legend("topleft", legend=c("Actual=TRUE","Actual=FALSE"), col=c(rgb(1,0,0,0.5),rgb(0,0,1,0.5)), pt.cex=2, pch=15 )
```

Undersampling, Oversampling and Prior Probabilities

• • •

How to adjust your model to account for under/oversampling

Undersampling and Prior Probabilities

- Say you have a rare event as target (<10% of data)
  - Fraud
  - Catastrophic failure
  - 10%± single day change in value of stock market index
- May have trouble modeling because a model is accurate for classifying everything as nonevent!
- *Potential* Solution: Create a biased sample

Undersampling and Prior Probabilities

Undersample:

- Under-represent common events in training data.
- Keep all rare events and only a fraction of common events
- Ratio of Common:Rare events is up for debate.
    - 70:30 ought to be fine.
    - 50:50 is sometimes encouraged.

Oversample:

- Replicate the rare events in training.
- Do this *after* the training/validation split so don't have the same observation in both training and validation set!
- OR, use a hybrid technique like **SMOTE** (Chawla, 2002) that creates new data points *like* the rare events (not exact replicates)

Undersampling and Prior Probabilities

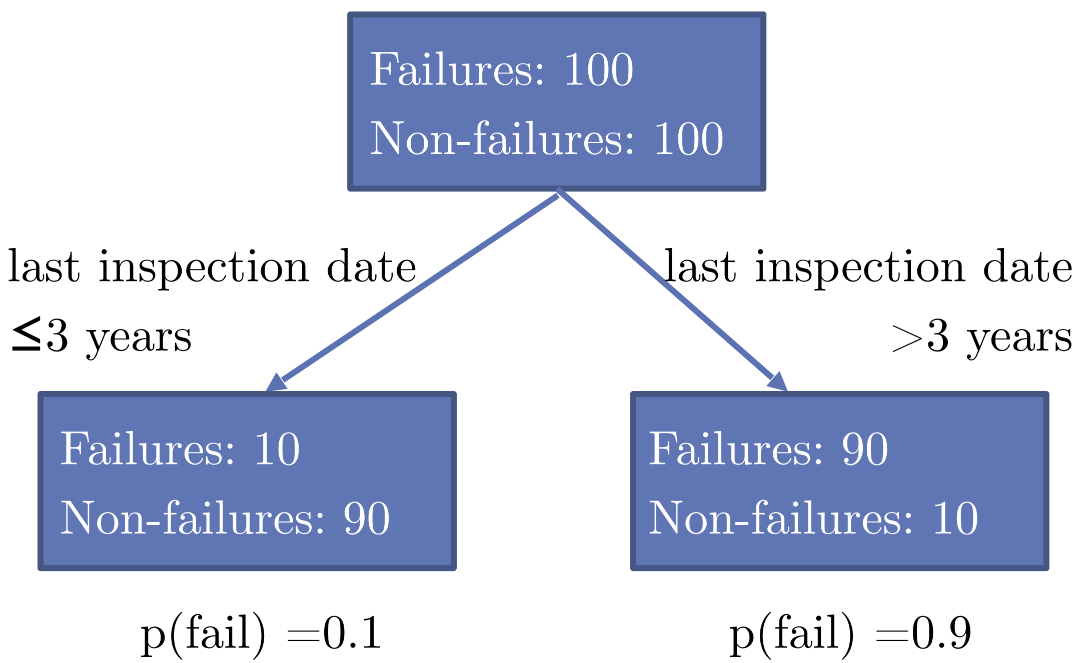
- Models provide **posterior probabilities** for events.
- The accuracy of the posterior probabilities rely on a representative sample.
- If we bias our sample, must adjust the posterior probabilities to account for this.

Why Adjustment is Necessary

Goal: Predict voting machine failure. Only 100 voting machines failed out of 10,000.

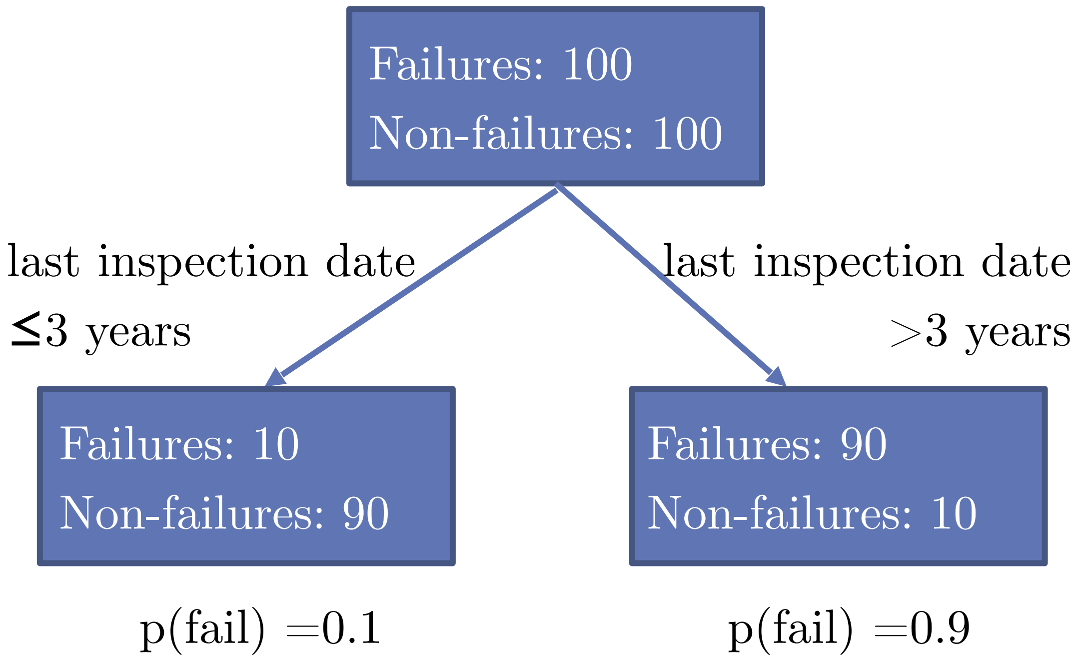
Undersample: Dataset has 100 failures and 100 non-failures.

Create Model:



Why Adjustment is Necessary

Does a new machine with last inspection date >3 years really have a 90% probability of failing?



Why Adjustment is Necessary

- We'd have to go back to the data to answer this question.
- Assuming the 100 non-failures chosen were random, representative sample, we expect inspection date to be ≤ 3 years 90% of the time.
- That is 8,910 non-failing machines with inspection date ≤ 3 years. (8,910 = 90% of 9,900)
- Similarly, 10% of non-failures have expect inspection date >3 years ago. This is 990 machines.

	≤ 3 years	>3 years
Failures	10	90
Nonfailures	8910	990

P(Failure | last inspection date >3 years)
90/(90+990) = 8%
(Still failing at 8 times the rate of recently inspected machines)

Summary: Adjusting for Undersampling

- Let $l = l_1,\ l_2, \ldots,\ l_L$ be the levels of the target variable
- Let $i = 1{,}2, \ldots,\ n$ index the observations in the data
- Let $OldPost(i, l)$ be the posterior probability from the model on oversampled data
- Let $OldPrior(l)$ be the proportion of target level in the oversampled data
- Let $Prior(l)$ be the correct proportion of target level in true population

$$NewPost(i, l) = \frac{OldPost(i, l)\frac{Prior(l)}{OldPrior(l)}}{\sum_{j=1}^{L} OldPost(i, l_j)\frac{Prior(l_j)}{OldPrior(l_j)}}$$