Evaluating Classifiers

and the Class Imbalance Problem

Our First Metrics

❤️

- $Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$

- $Error\ rate = \frac{Number\ of\ wrong\ predictions}{Total\ number\ of\ predictions} = 1 - Accuracy$

- In a binary classification we use a confusion Matrix:

		True condition	
	Total population	Condition positive	Condition negative
Predicted condition	Predicted condition positive	**True positive**	**False positive**, Type I error
	Predicted condition negative	**False negative**, Type II error	**True negative**

Also, look up the best wiki ever! https://en.wikipedia.org/wiki/Sensitivity_and_specificity

Error and Accuracy

- Error rate = fraction of incorrect predictions on the testing set
  - Probability of misclassification

- Accuracy = fraction of correct predictions on the testing set ($1 - error\ rate$)
  - Probability of correct prediction

- Example: A classifier misclassifies 8 out of 30 test cases:
  - Error rate = 8/30 = 0.267
  - Accuracy = 22/30 = 0.733

Class Imbalance Problem

Test set:

Location	Retailer	Amount	Class
Austin	HEB	$50	legitimate
Austin	UT Co-op	$300	legitimate
San Antonio	Mi Tierra	$25	legitimate
Austin	Freebirds	$7	legitimate
Austin	HEB	$75	legitimate
Austin	Target	$150	legitimate
Moscow	Tsum	$5000	fraudulent
Austin	Taco Cabana	$5	legitimate
San Antonio	Target	$25	legitimate
Austin	Trader Joe's	$55	legitimate
Austin	Alamo Drafthouse	$20	legitimate
…	…	…	…
(1000 total records)			(1 of the 1000 is fraudulent)

Run it through a classifier that Classifies everything as legitimate…

Accuracy = 999/1000 = 99.9%
Error Rate = 1/1000 = 0.1%

Confusion Matrix (Binary Classification)

		Predicted Class	
		+	-
Actual Class	+	F_{++} (TP)	F_{+-} (FN)
	-	F_{-+} (FP)	F_{--} (TN)

Error rate: fraction of mistakes
$Error\ rate = (FP + FN) / n$

Accuracy: fraction of correct predictions
$Accuracy = (TP + TN) / n$

True positive rate (TPR), or **sensitivity**: fraction of positive examples correctly predicted
$TPR = TP / (TP + FN)$

True negative rate (TNR), or **specificity**: fraction of negative examples correctly predicted
$TNR = TN / (FP + TN)$

False positive rate (FPR): fraction of negative examples predicted as positive
$FPR = FP / (FP + TN)$

False negative rate (FNR): fraction of positive examples predicted as negative
$FNR = FN / (TP + FN)$

Example

		Predicted Class	
		+	-
Actual Class	+	*2*	*8*
	-	*1*	*989*

Error rate: fraction of mistakes
Error rate = (1 + 8) / 1000 = 0.009 = 0.9%

Accuracy: fraction of correct predictions
Accuracy = (2 + 989) / 1000 = 0.991 = 99.1%

True positive rate (TPR), or **sensitivity**: fraction of positive examples correctly predicted
TPR = 2 / 10 = 0.2 = 20%

True negative rate (TNR), or **specificity**: fraction of negative examples correctly predicted
TNR = 989 / 990 = 0.999 = 99.9%

False positive rate (FPR): fraction of negative examples predicted as positive
FPR = 1 / 990 = 0.001 = 0.1%

False negative rate (FNR): fraction of positive examples predicted as negative
FNR = 8 / 10 = 0.8 = 80%

Confusion Matrix with Cross Validation

- Use the SUM

Fold 1:
20 train / 10 test

		Predicted Class	
		+	-
Actual Class	+	4	2
	-	1	3

+

Fold 2:
20 train / 10 test

		Predicted Class	
		+	-
Actual Class	+	5	3
	-	0	2

+

Fold 3:
20 train / 10 test

		Predicted Class	
		+	-
Actual Class	+	1	8
	-	0	1

=

Final Confusion Matrix:
All 30 records

		Predicted Class	
		+	-
Actual Class	+	10	13
	-	1	6

Precision and Recall

		Predicted Class	
		+	-
Actual Class	+	*2*	*8*
	-	*1*	*989*

Precision, or Positive Predictive Value (PPV) addresses the question: "Given a positive prediction from the classifier, how likely is it to be correct?"

Recall, or True Positive Rate (TPR) addresses the question: "Given a positive example, will the classifier detect it?"

Class-specific **precision/PPV**: fraction of records that actually are of class C, out of records predicted to be of class C
Prec(+) = TP/(TP+FP) = 2 / 3 = 0.667 = 66.7%
Prec(-) = 989 / 997 = 0.991 = 99.1%

Class-specific **recall/coverage/sensitivity/TPR**: fraction of correct predictions of class C, over all points in class C
Rec(+) = TP/(TP+FN) = 2 / 10 = 0.2 = 20%
Rec(-) = 989 / 990 = 0.999 = 99.9%

Typically, we're only concerned with the precision and recall of the positive (rare) class.

Multi Class Confusion Matrix

		Predicted Class		
		Iris-setosa	Iris-versicolor	Iris-virginica
Actual Class	Iris-setosa	10	0	0
	Iris-versicolor	0	7	5
	Iris-virginica	0	3	6

Error rate: fraction of mistakes

$\mathit{Error\ rate} = 8/31 = 0.258 = 25.8\%$

Accuracy: fraction of correct predictions

$\mathit{Accuracy} = 23/31 = 0.742 = 74.2\%$

Class Confusion

Class-specific **precision/PPV**: fraction of records that actually are of class C, out of records predicted to be of class C

$\mathit{Prec}(\mathit{setosa}) = 10 / 10 = 1 = 100\%$

$\mathit{Prec}(\mathit{versicolor}) = 7 / 10 = 0.7 = 70\%$

$\mathit{Prec}(\mathit{virginica}) = 6 / 11 = 0.545 = 54.5\%$

Class-specific **recall/coverage/TPR**: fraction of correct predictions of class C, over all points in class C

$\mathit{Rec}(\mathit{setosa}) = 10 / 10 = 1 = 100\%$

$\mathit{Rec}(\mathit{versicolor}) = 7 / 12 = 0.583 = 58.83\%$

$\mathit{Rec}(\mathit{virginica}) = 6 / 9 = 0.667 = 66.7\%$

Precision/Recall Tradeoff

		Predicted Class	
		+	-
Actual Class	+	*10*	*0*
	-	*990*	*0*

Class-specific **precision/PPV**: fraction of records that actually are of class C, out of records predicted to be of class C

$Prec(+) = 10 / 1000 = 0.01 = 1\%$

Class-specific **coverage/recall/TPR**: fraction of correct predictions of class C, over all points in class C

$Rec(+) = 10 / 10 = 1 = 100\%$

Precision/Recall Tradeoff

		Predicted Class	
		+	-
Actual Class	+	*1*	*9*
	-	*0*	*990*

Class-specific **precision/PPV**: fraction of records that actually are of class C, out of records predicted to be of class C

$Prec(+) = 1 / 1 = 1 = 100\%$

Class-specific **recall/coverage/TPR**: fraction of correct predictions of class C, over all points in class C

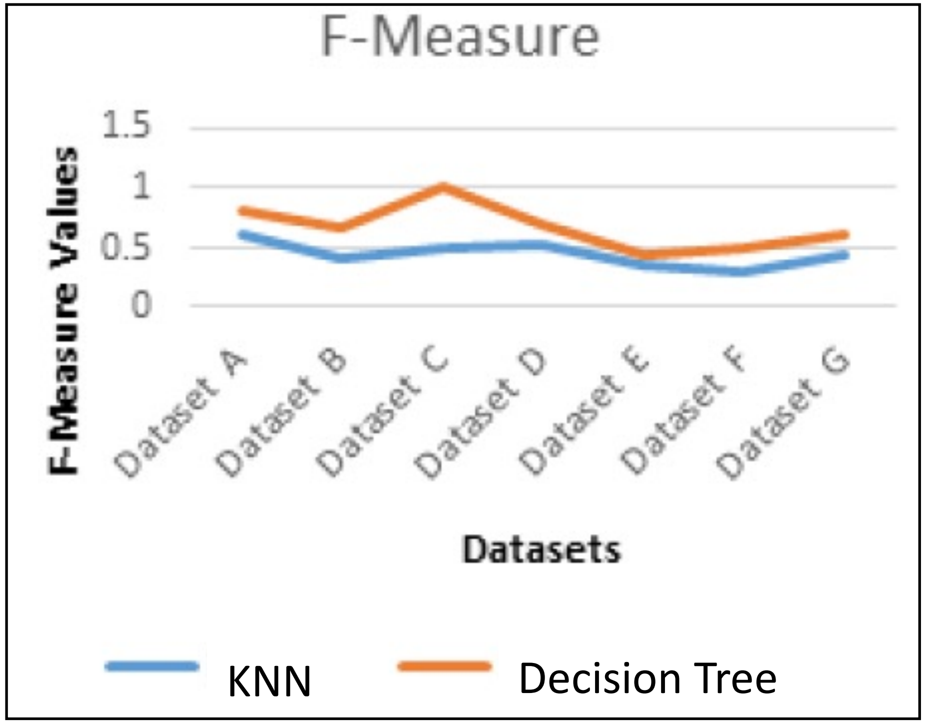
$Rec(+) = 1 / 10 = 0.1 = 10\%$

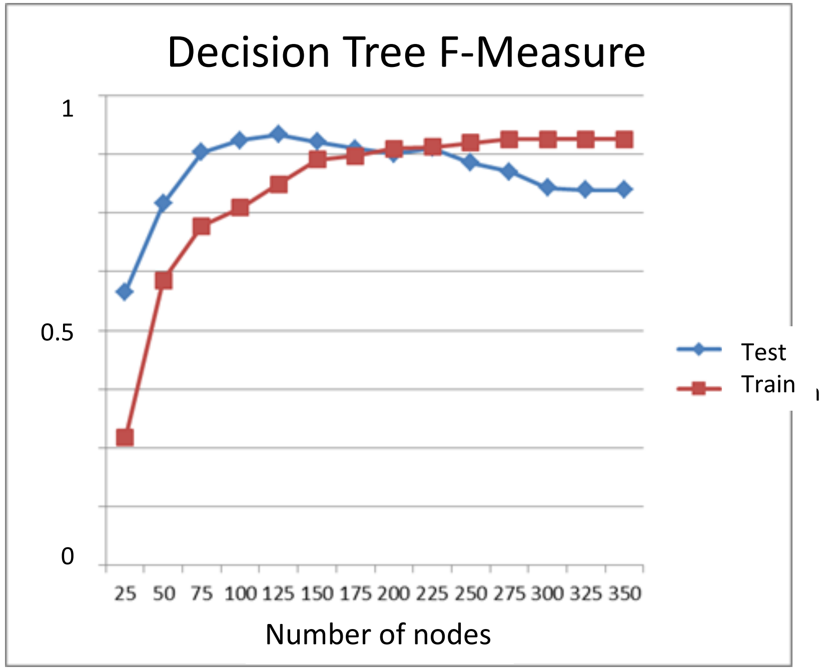
F-measure

- F-measure summarizes both precision and recall into one metric

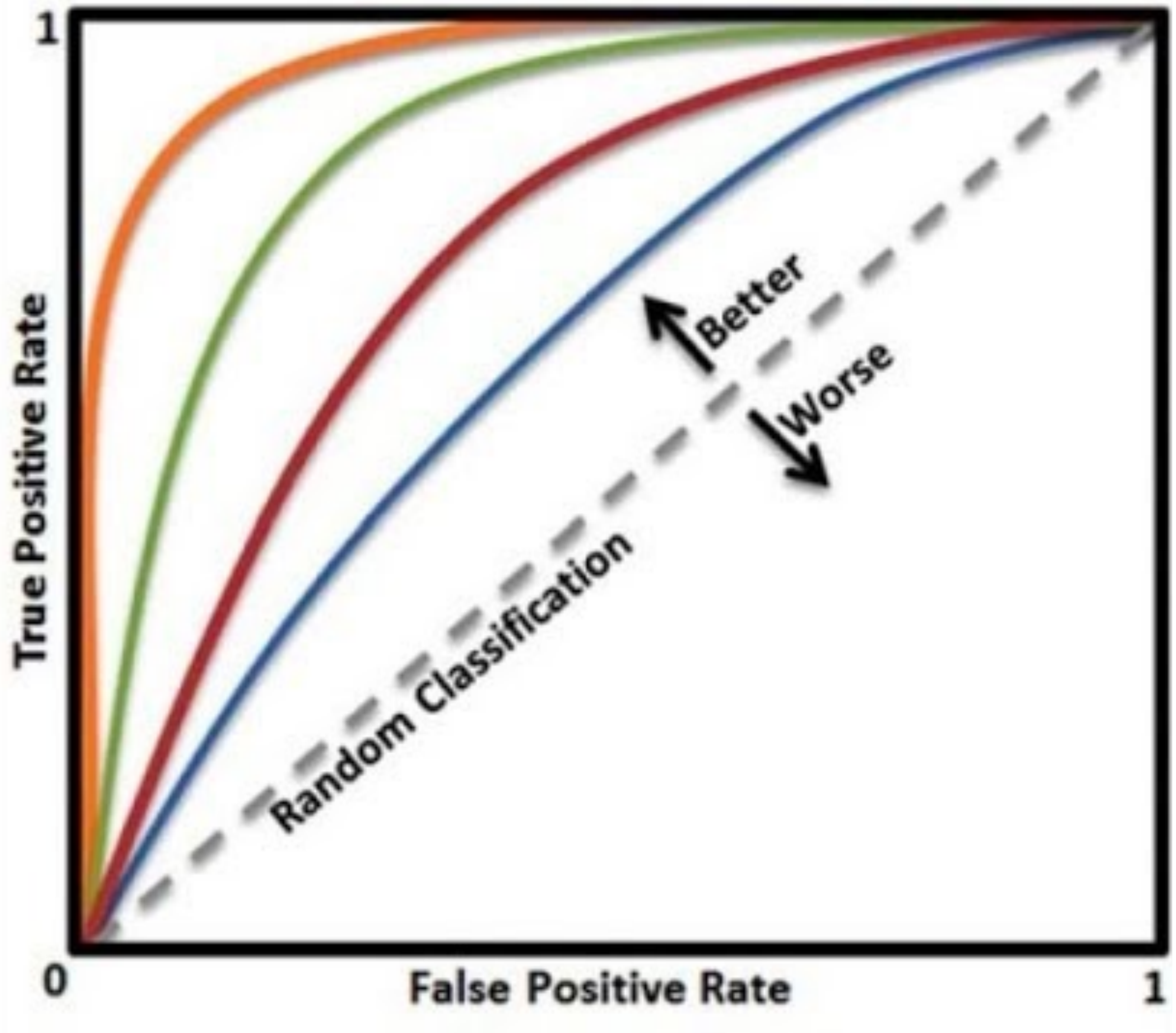
$$F = \frac{2 \times precision \times recall}{precision + recall}$$

- The overall F-measure of the classifier is the mean of the class-specific F-measures
- Or you could consider the F-measure of only the positive class





ROC Curves (Receiver Operating Characteristic)



To draw ROC curve, classifier must produce continuous-valued output

Outputs are used to rank test records, from the most likely positive class record to the least likely positive class record

Example

ID	Actual Class	Probability YES	Probability NO
1	Y	0.35	0.65
2	N	0.23	0.77
3	N	0.55	0.45
4	Y	0.32	0.68
5	Y	0.54	0.46
6	N	0.47	0.53

Example

ID	Actual Class	Probability YES	Probability NO
1	Y	0.35	0.65
2	N	0.23	0.77
3	Y	0.55	0.45
4	N	0.32	0.68
5	Y	0.54	0.46
6	N	0.47	0.53

Sort data by Probability YES

ID	Actual Class	Probability YES	Probability NO
3	Y	0.55	0.45
5	Y	0.54	0.46
6	N	0.47	0.53
1	Y	0.35	0.65
4	N	0.32	0.68
2	N	0.23	0.77

Example

Sort data by Probability YES

ID	Actual Class	Probability YES	Probability NO
3	Y	0.55	0.45
5	Y	0.54	0.46
6	N	0.47	0.53
1	Y	0.35	0.65
4	N	0.32	0.68
2	N	0.23	0.77

Select a cutoff threshold for the YES class = 0.5

Example

Sort data by Probability YES

ID	Actual Class	Probability YES	Probability NO
3	Y	0.55	0.45
5	Y	0.54	0.46
6	N	0.47	0.53
1	Y	0.35	0.65
4	N	0.32	0.68
2	N	0.23	0.77

Select a cutoff threshold for the YES class = 0.5

Calculate TPR (sensitivity) and FPR (1-specificity):

		Predicted Class	
		+	-
Actual Class	+	*2*	*1*
	-	*0*	*3*

TPR = 2/3 = 0.67
FPR = 0/3 = 0

This becomes a point on our ROC curve

Example

Sort data by Probability YES

ID	Actual Class	Probability YES	Probability NO
3	Y	0.55	0.45
5	Y	0.54	0.46
6	N	0.47	0.53
1	Y	0.35	0.65
4	N	0.32	0.68
2	N	0.23	0.77

Now adjust the threshold for the YES class = 0.4

Example

Sort data by Probability YES

ID	Actual Class	Probability YES	Probability NO
3	Y	0.55	0.45
5	Y	0.54	0.46
6	N	0.47	0.53
1	Y	0.35	0.65
4	N	0.32	0.68
2	N	0.23	0.77

Now adjust the threshold for the YES class = 0.4

Calculate TPR (sensitivity) and FPR (1-specificity):

		Predicted Class	
		+	-
Actual Class	+	*2*	*1*
	-	*1*	*2*

TPR = 2/3 = 0.67
FPR = 1/3 = 0.33

This becomes a point on our ROC curve

How to Construct an ROC curve

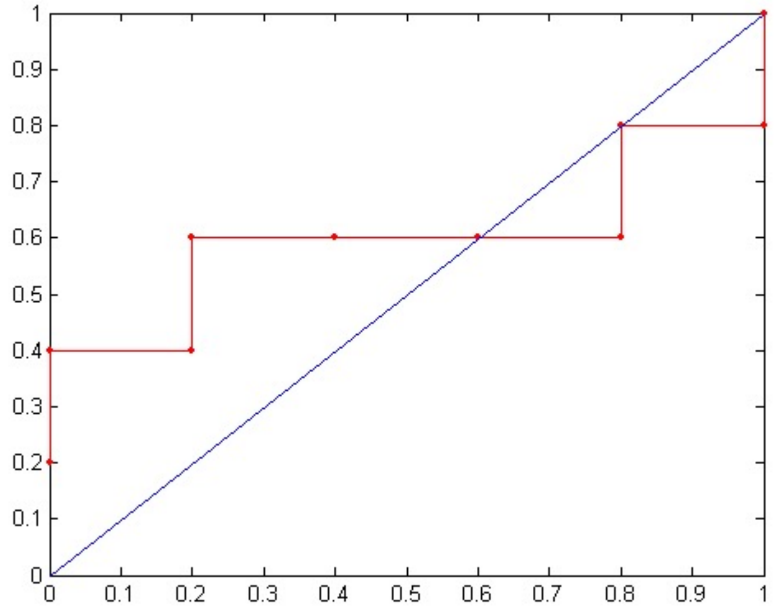
Instance	Score	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

- Use a classifier that produces a continuous-valued score for each instance
  - The more likely it is for the instance to be in the + class, the higher the score
- Sort the instances in decreasing order according to the score
- Apply a threshold at each unique value of the score
- Count the number of TP, FP, TN, FN at each threshold
  - TPR = TP/(TP+FN)
  - FPR = FP/(FP + TN)

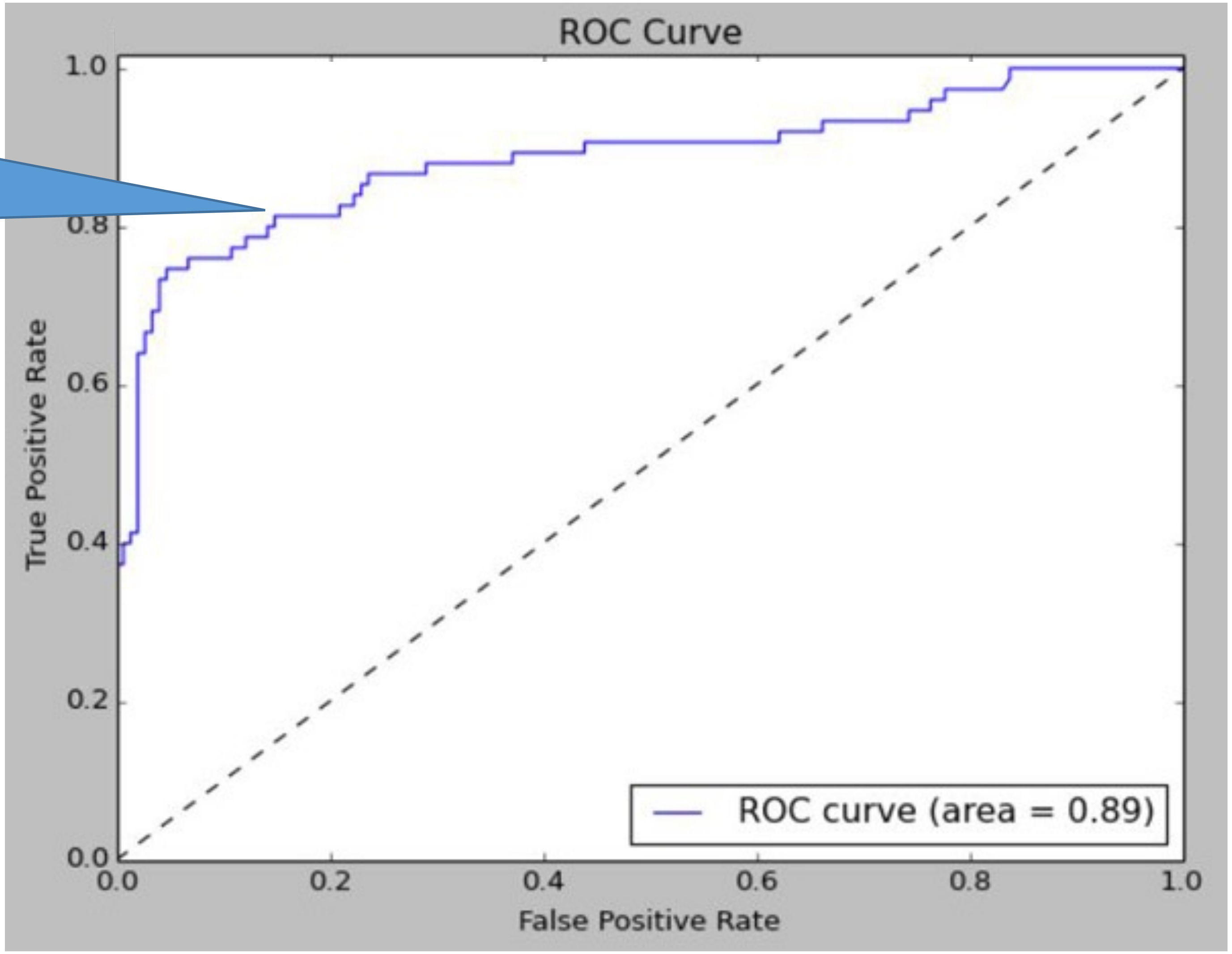
How to construct an ROC curve

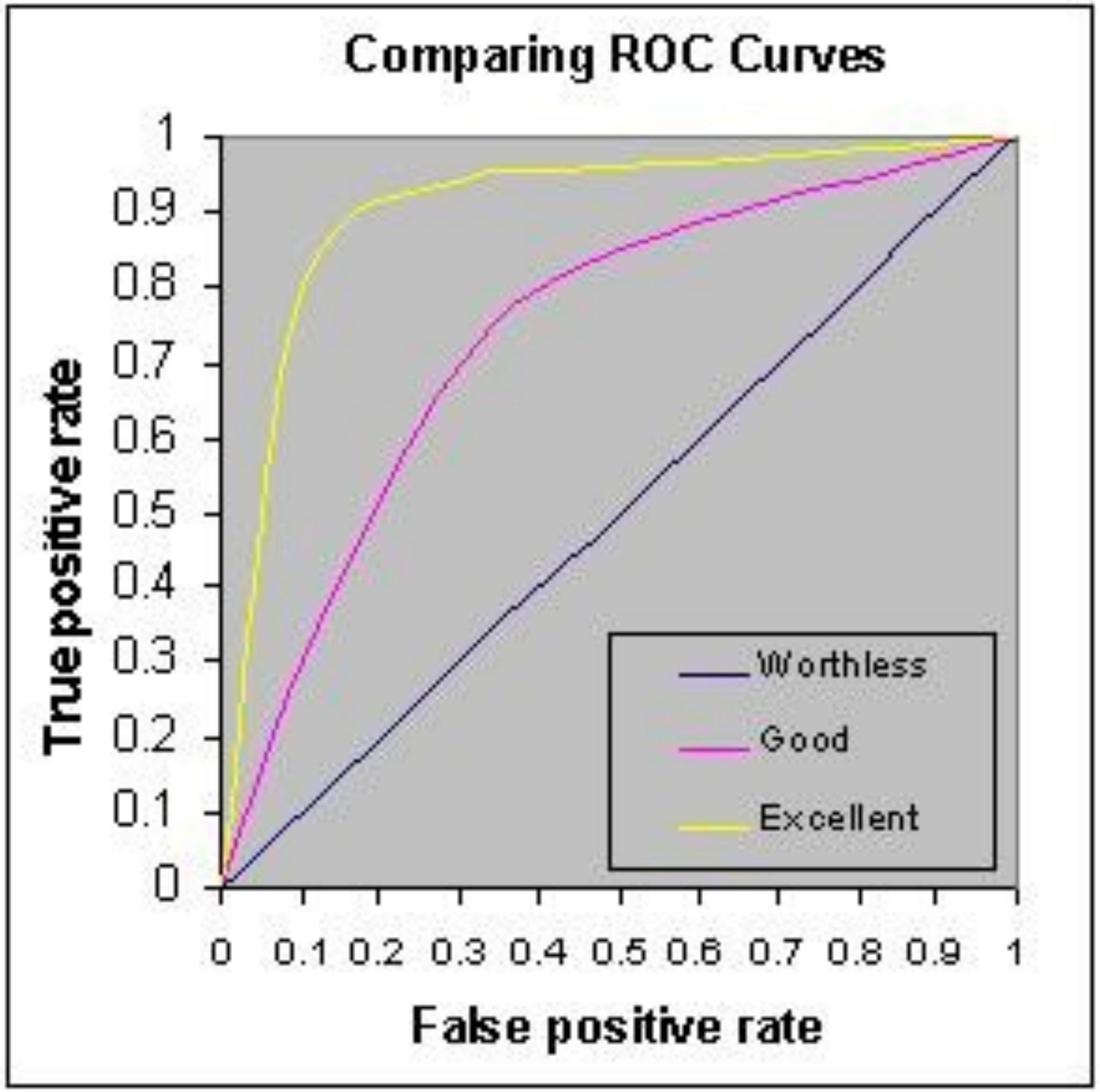
Class	+	-	+	-	-	-	+	-	+	+	
Threshold >=	0.25	0.43	0.53	0.76	0.8	0.82	0.85	0.87	0.93	0.95	1.00
TP	5	4	4	3	3	3	3	2	2	1	0
FP	5	5	4	4	3	2	1	1	0	0	0
TN	0	0	1	1	2	3	4	4	5	5	5
FN	0	1	1	2	2	2	2	3	3	4	5
→ TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
→ FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0

ROC Curve:



Every point on the ROC curve was generated by a single threshold – the threshold selected for run time is called the <u>operating point</u>





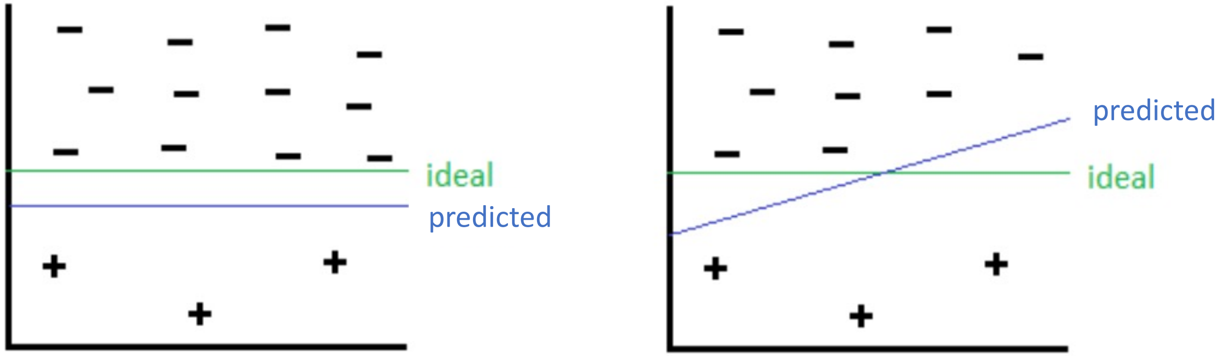
Area Under the Curve (AUC) can be used to compare classifiers

1 >= AUC >= 0.9 : excellent (A)
0.9 > AUC >= 0.8 : good (B)
0.8 > AUC >= 0.7 : fair (C)
0.7 > AUC >= 0.6 : poor (D)
0.6 > AUC >= 0.5 : fail (F)

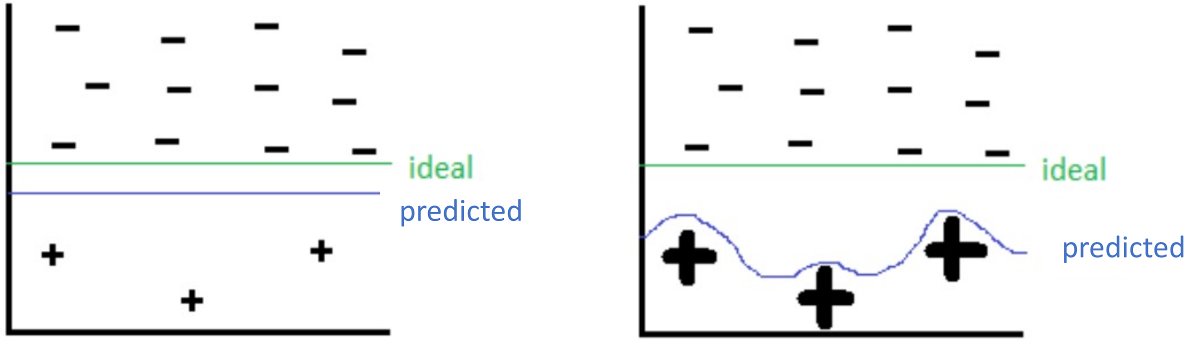
ROC Curves: https://machinelearningmastery.com/assessing-comparing-classifier-performance-roc-curves-2/

Mitigating Class Imbalances

- Sampling based approaches
  - Undersampling: remove some of the majority class

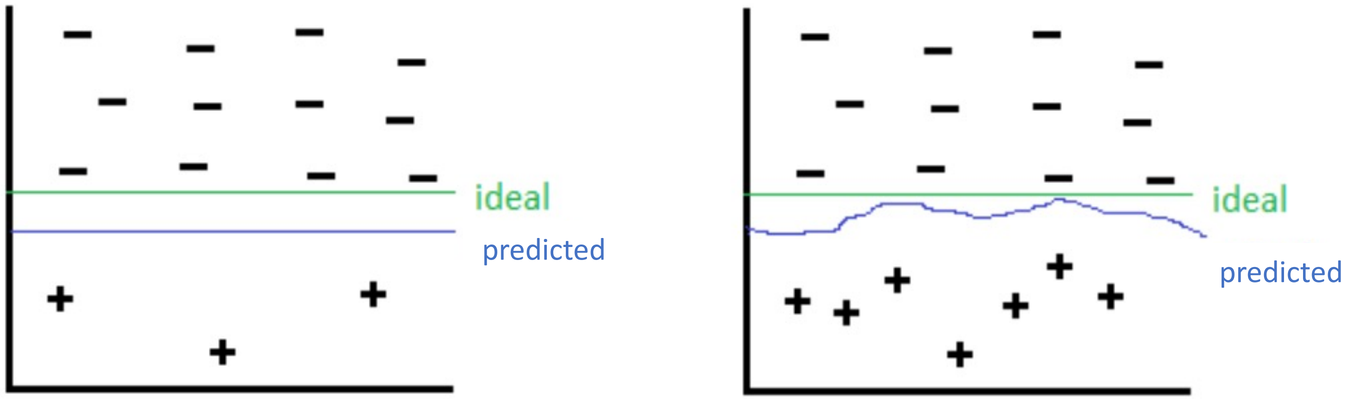


  - Oversampling: duplicate the minority class records



Sampling Algorithms

- SMOTE: Synthetic Minority Over-Sampling Technique
  - For each minority instance C, find it's nearest neighbor N
  - Create a new minority class instance R, using C's features and the difference between C and N's features, multiplied by a random variable
  - R.features = C.features + (C.features – N.features) * rand(0,1)



Cost Matrix

- A cost matrix can encode a penalty for misclassification errors

		Predicted Class	
		+	-
Actual Class	+	*-1*	*100*
	-	*1*	*0*

A negative entry in a cost matrix indicates a reward for making a correct prediction

- Can be used for evaluation

Model 1		Predicted Class	
		+	-
Actual Class	+	*175*	*25*
	-	*50*	*250*

Error = 75 / 500 = 0.375
F-measure = 0.84
Cost = -1(175) + 100(25) + 1(50) + 0(250) = 2375

Model 2		Predicted Class	
		+	-
Actual Class	+	*170*	*30*
	-	*25*	*275*

Error = 55 / 500 = 0.11
F-measure = 0.89
Cost = -1(170) + 100(30) + 1(25) + 0(275) =2855

Lower error,
Better F-score,
But higher cost

Using Cost Matrix to Evaluate Risk

Cost Matrix		Predicted Class	
		+	-
Actual Class	+	-1	10
	-	1	0

For a new record, the probability that it is positive is 20% and the probability that it is negative is 80%:
P(+) = 0.2
P(-) = 0.8

If I classify this record as negative, there is a 20% chance that I'm classifying it wrong and that I'm making a false negative (FN) error. (I would be predicting it as negative, but it is actually positive.) If I'm making that type of error, that is a cost of 10.

There is a 20% chance I'm making an error that costs 10. So the **risk** of classifying this record as negative is:
Risk(-) = (0.2)(10) = 2

Similarly, I can calculate the risk of classifying this record as positive. There would be an 80% chance I'm making an error that costs 1:
Risk(+) = (0.8)(1) = 0.8

Choose to classify as the class that has the **lowest risk**.