Predictive Modelling

Classification - Classifier Evaluation

Jonathan Mwaura

Khoury College of Computer Sciences

July 23, 2024

The Roux Institute
at Northeastern University

Introduction

Textbook

Reading: Chapter 4 of: Gareth James et al (2021) . An Introduction to Statistical Learning (2nd Edition) .

`https://www.statlearning.com/`

Acknowledgements

These slides have been adapted from the following Professors:

1) Andrew Ng - Stanford
2) Eric Eaton - UPenn
3) David Sontag - MIT
4) Alina Oprea - Northeastern

Classifier Evaluation

- Classification is a supervised learning problem
  - Prediction is binary or multi-class
- Classification techniques
  - Linear classifiers
    - Perceptron (online or batch mode)
    - Logistic regression (probabilistic interpretation)
  - Instance learners
    - kNN: need to store entire training data
- Cross-validation should be used for parameter selection and estimation of model error

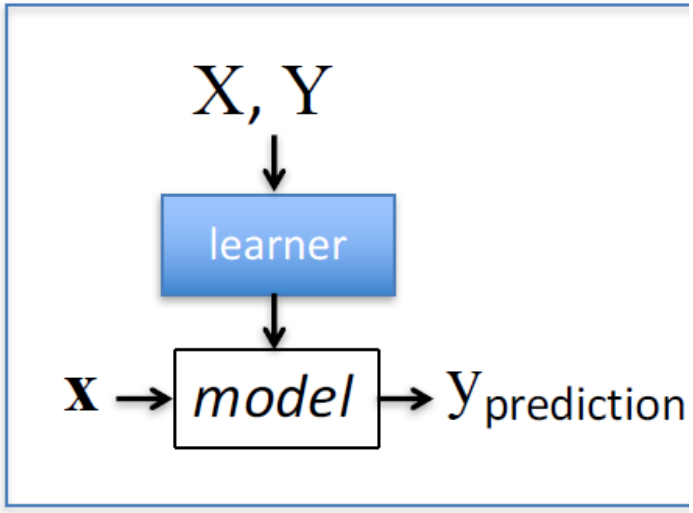
Evaluation of classifiers

Given: labeled training data $X, Y = \{\langle \boldsymbol{x}_i, y_i \rangle\}_{i=1}^{n}$

- Assumes each $\boldsymbol{x}_i \sim \mathcal{D}(\mathcal{X})$

Train the model:

model ← *classifier*.train(X, Y)



Apply the model to new data:

- Given: new unlabeled instance $\boldsymbol{x} \sim \mathcal{D}(\mathcal{X})$

$y_{\text{prediction}}$ ← *model*.predict(**x**)

Classification Metrics

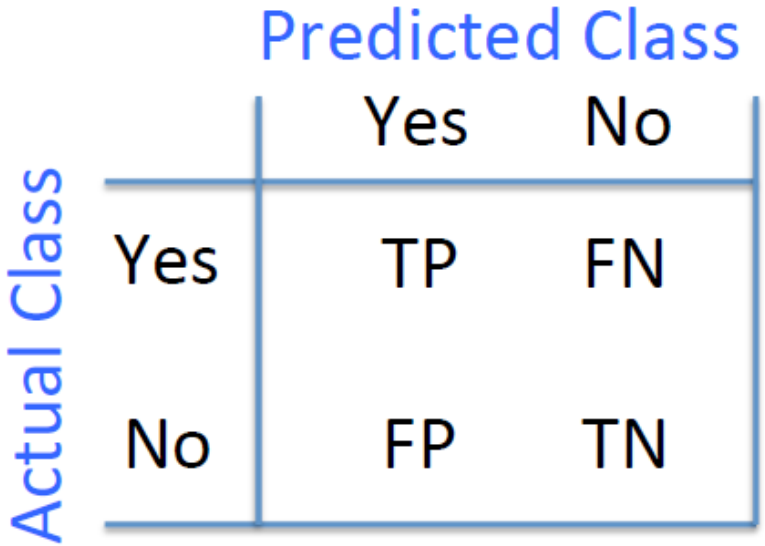
$$\text{accuracy} = \frac{\#\text{ correct predictions}}{\#\text{ test instances}}$$

$$\text{error} = 1 - \text{accuracy} = \frac{\#\text{ incorrect predictions}}{\#\text{ test instances}}$$

- Training set accuracy and error
- Testing set accuracy and error

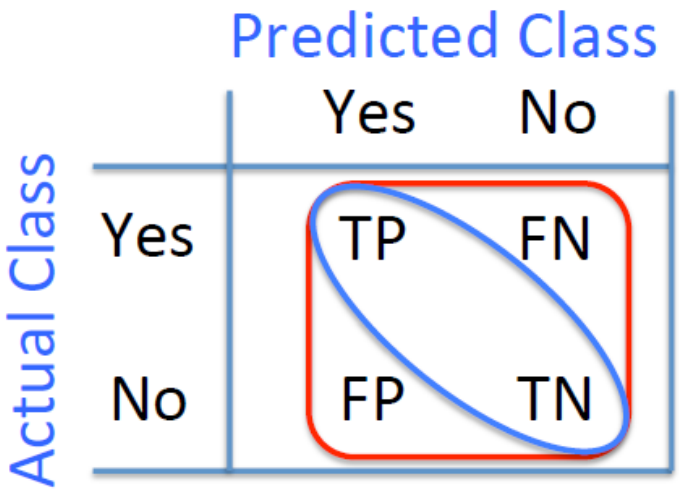
Confusion Matrix

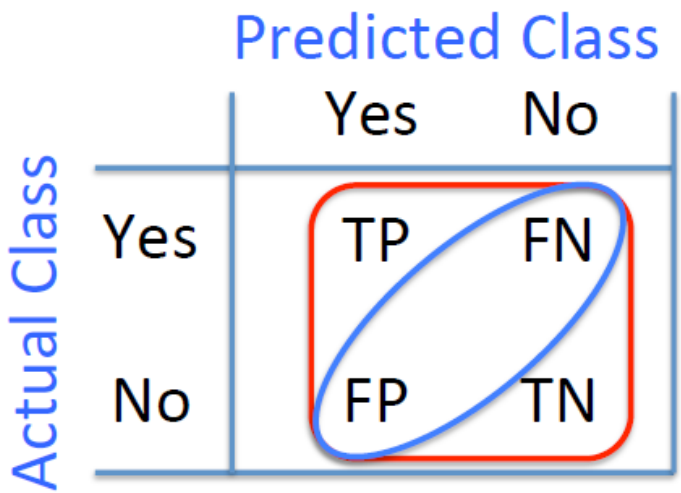
Given a dataset of P positive instances and N negative instances:



Accuracy and Error

Given a dataset of P positive instances and N negative instances:



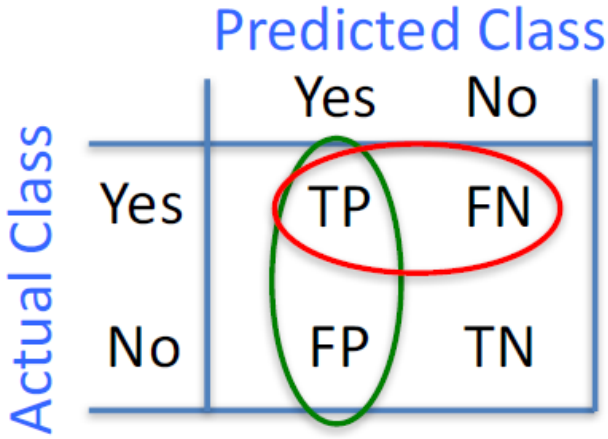


$$\text{accuracy} = \frac{TP + TN}{P + N}$$

$$\text{error} = 1 - \frac{TP + TN}{P + N} = \frac{FP + FN}{P + N}$$

Confusion Matrix

- Given a dataset of P positive instances and N negative instances:



$$\text{accuracy} = \frac{TP + TN}{P + N}$$

- Imagine using classifier to identify positive cases (i.e., for information retrieval)

$$\text{precision} = \frac{TP}{TP + FP}$$

Probability that classifier predicts positive correctly

$$\text{recall} = \frac{TP}{TP + FN}$$

Probability that actual class is predicted correctly

Why One Metric is Not Enough

Assume that in your training data, Spam email is 1% of data, and Ham email is 99% of data

- Scenario 1
  - Have classifier always output HAM!
  - What is the accuracy? 99%
- Scenario 2
  - Predict one SPAM email as SPAM, all other emails as legitimate
  - What is the precision? 100%
- Scenario 3
  - Output always SPAM!
  - What is the recall? 100%

Precision & Recall

Precision

- the fraction of positive predictions that are correct
- P(is pos|predicted pos)

$$\text{precision} = \frac{TP}{TP + FP}$$

Recall

- fraction of positive instances that are identified
- P(predicted pos|is pos)

$$\text{recall} = \frac{TP}{TP + FN}$$

- You can get high recall (but low precision) by only predicting positive
- Recall is a non-decreasing function of the # positive predictions
- Typically, precision decreases as either the number of positive predictions or recall increases
- Precision & recall are widely used in information retrieval

F-Score

- Combined measure of precision/recall tradeoff

$$F_1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

  - This is the harmonic mean of precision and recall
  - In the F_1 measure, precision and recall are weighted evenly
  - Can also have biased weightings that emphasize either precision or recall more (F_2 = 2 × recall; $F_{0.5}$ = 2 × precision)

- Limitations:
  - F-measure can exaggerate performance if balance between precision and recall is incorrect for application
    - Don't typically know balance ahead of time

A Word of Caution

- Consider binary classifiers A, B, C:

		A	.	B	.	C	.
		1	0	1	0	1	0
Predictions	1	0.9	0.1	0.8	0	0.78	0
	0	0	0	0.1	0.1	0.12	0.1

- Clearly A is useless, since it always predicts 1
- B is slightly better than C
  - less probability mass wasted on the off-diagonals
- But, here are the performance metrics:

Metric	A	B	C
Accuracy	0.9	0.9	0.88
Precision	0.9	1.0	1.0
Recall	1.0	0.888	0.8667
F-score	0.947	0.941	0.9286

Classifiers can be tuned

- Logistic regression sets by default the threshold at 0.5 for classifying positive and negative instances
- Some applications have strict constraints on false positives (or other metrics)
  - Example: very low false positives in security (spam)
- Solution: choose different threshold

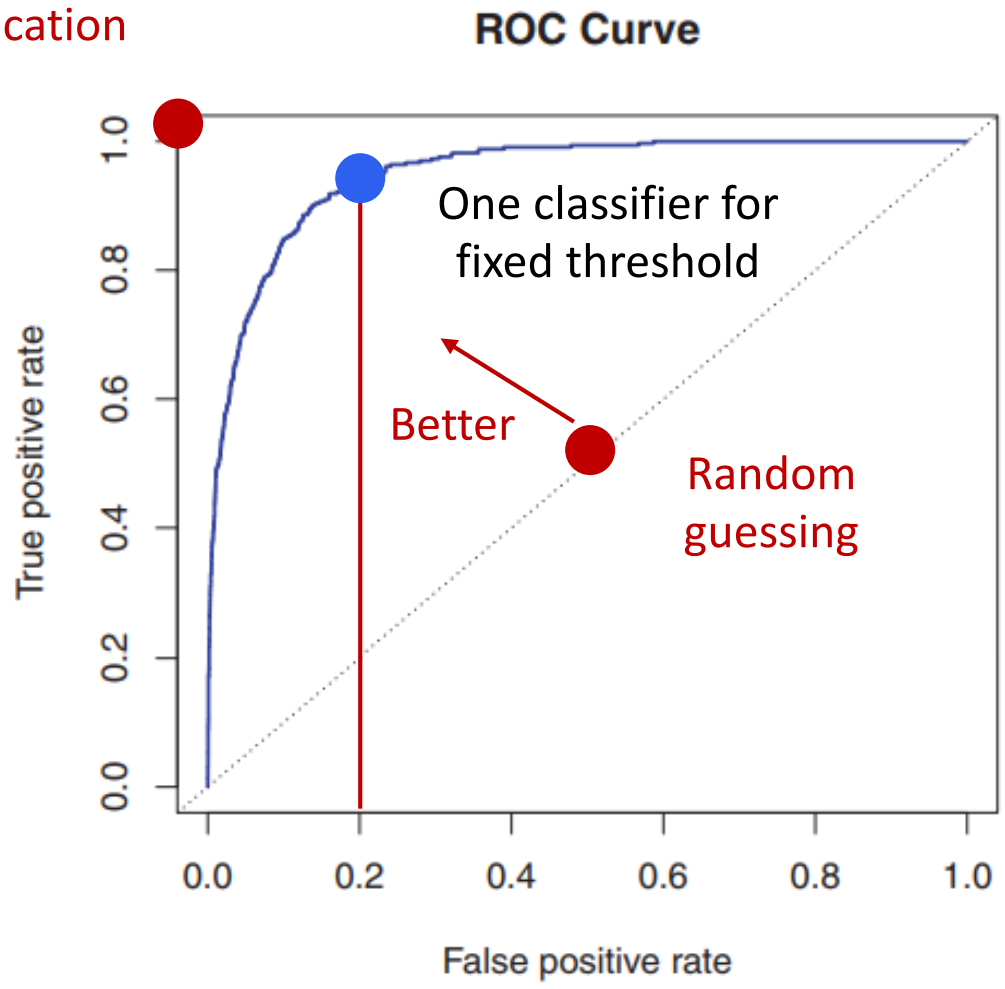
Probabilistic model $h_{\theta(x)} = \mathrm{P}[y = 1|x; \theta]$

- Predict y = 1 if $h_{\boldsymbol{\theta}}(\boldsymbol{x}) \geq$ T
- Predict y = 0 if $h_{\boldsymbol{\theta}}(\boldsymbol{x}) <$ T

Higher T, lower FP
Lower T, lower FN

ROC Curves

Perfect classification

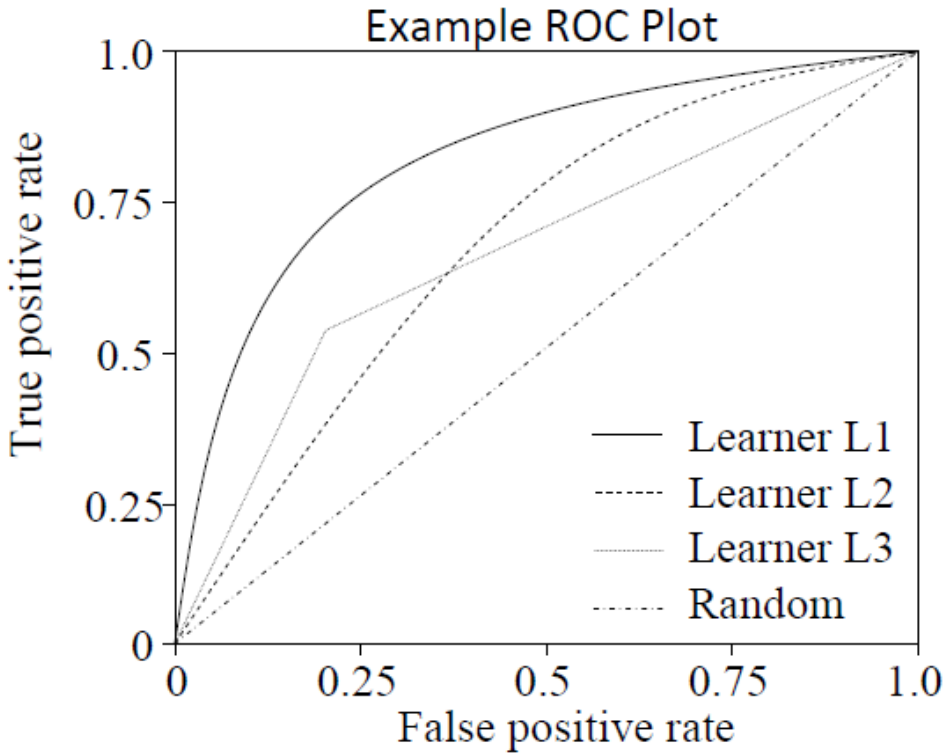


- Receiver Operating Characteristic (ROC)
- Determine operating point (e.g., by fixing false positive rate)

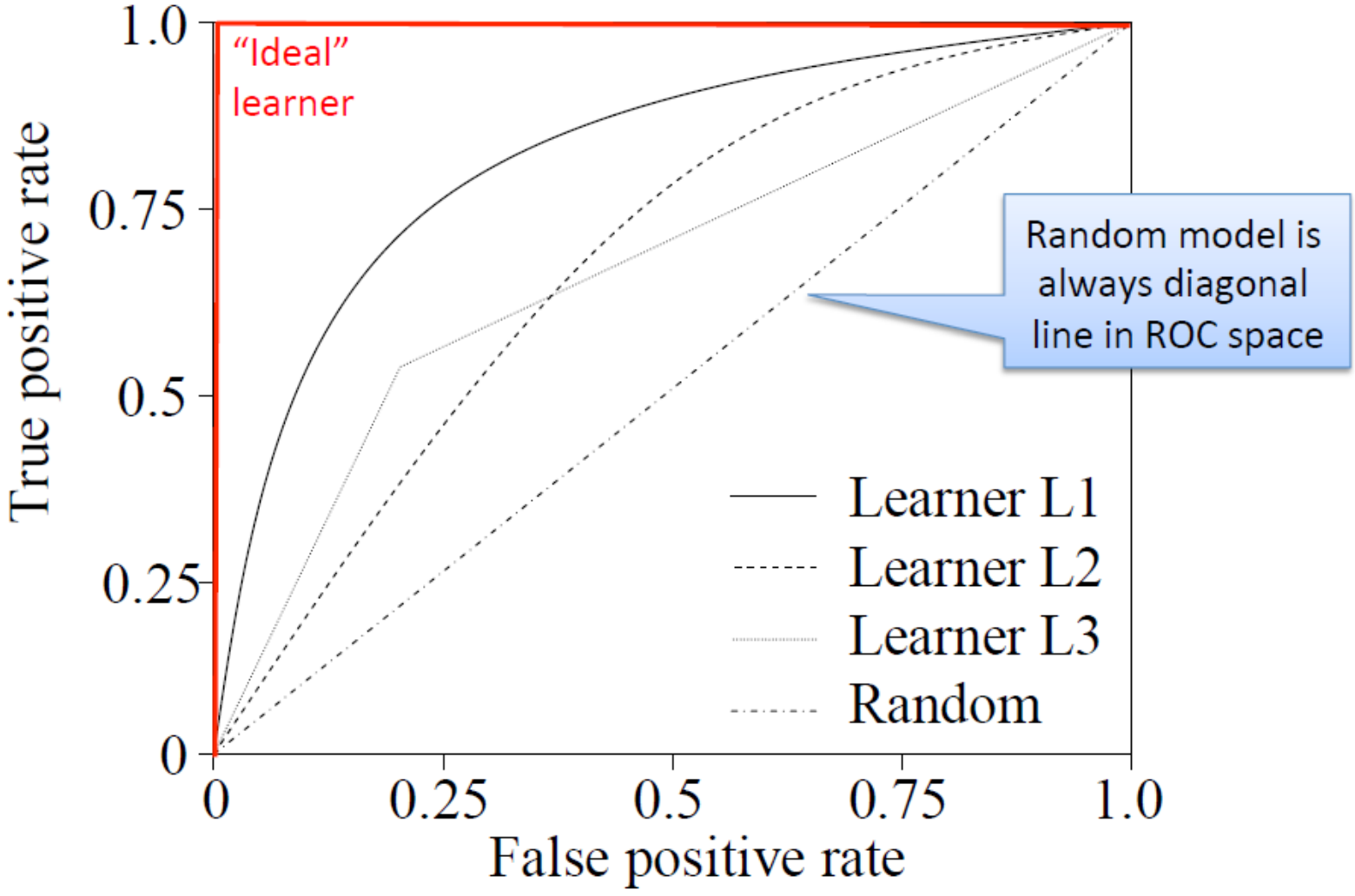
Performance Depends on Threshold

Predict positive if $\mathrm{P}(y = 1 \mid \mathbf{x}) > \theta$, otherwise negative

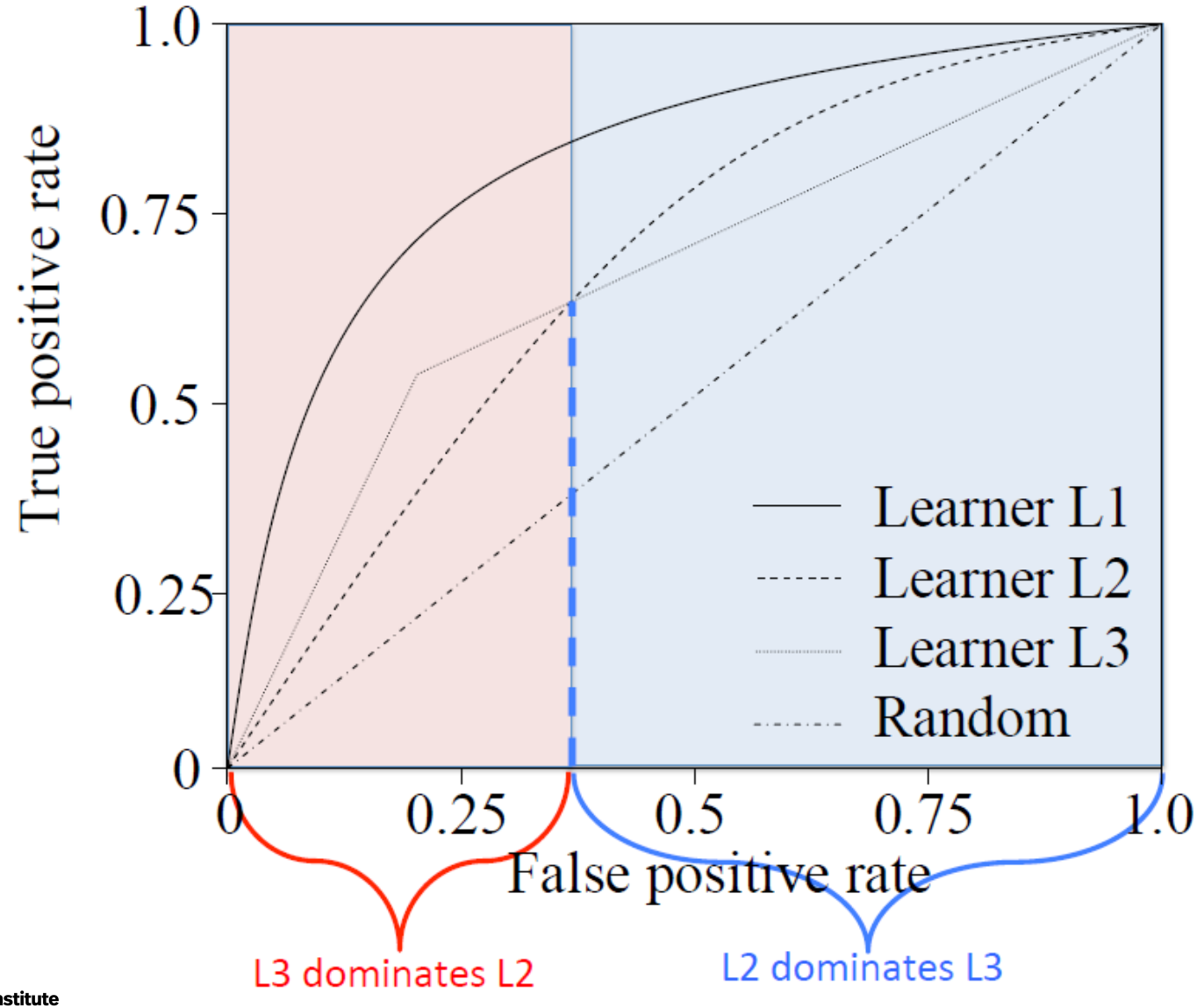
- Number of TPs and FPs depend on threshold θ
- As we vary θ, we get different (TPR, FPR) points



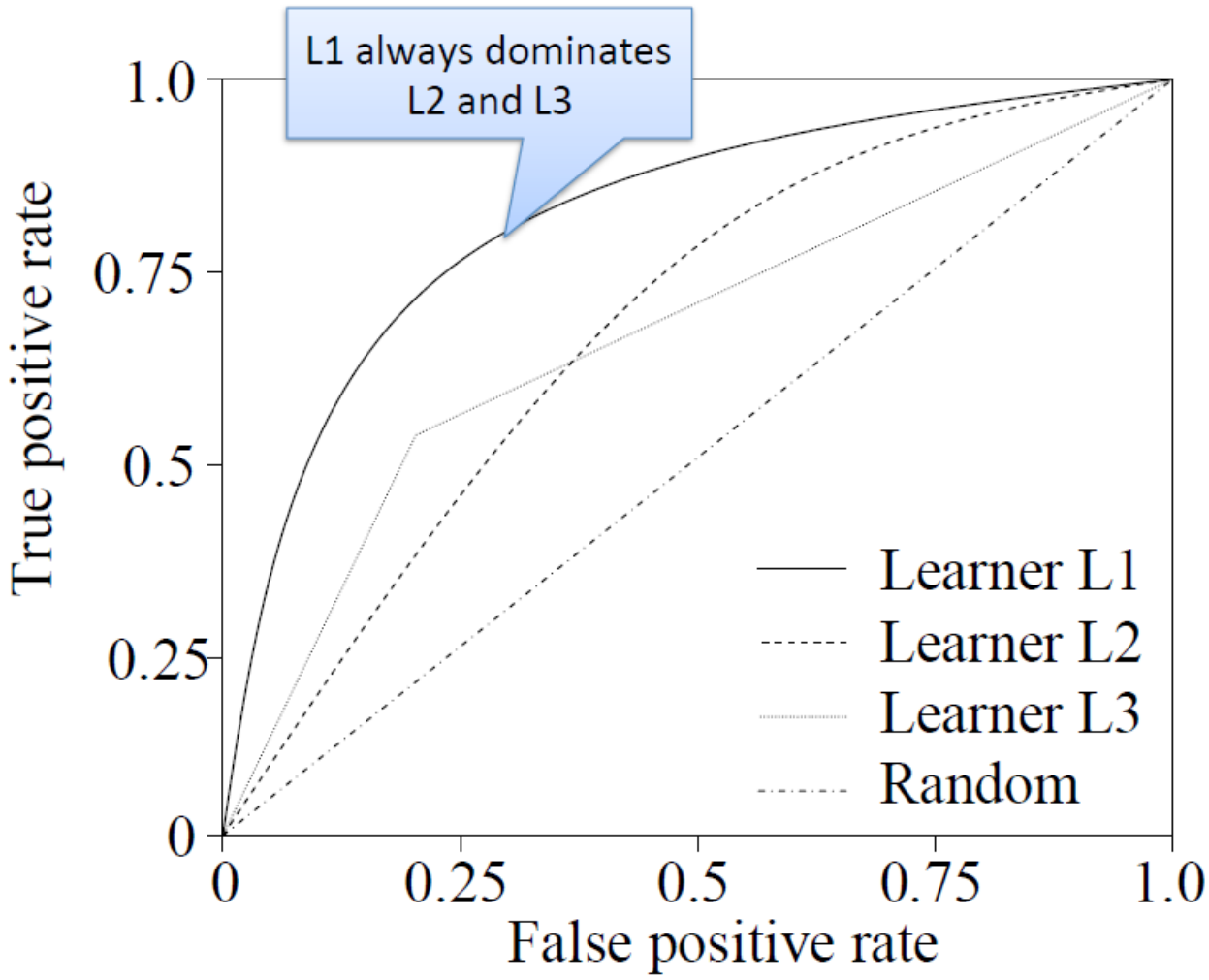
ROC Curve


"Ideal" learner
Random model is always diagonal line in ROC space
Learner L1
Learner L2
Learner L3
Random
True positive rate
False positive rate
1.0
0.75
0.5
0.25
0
0
0.25
0.5
0.75
1.0

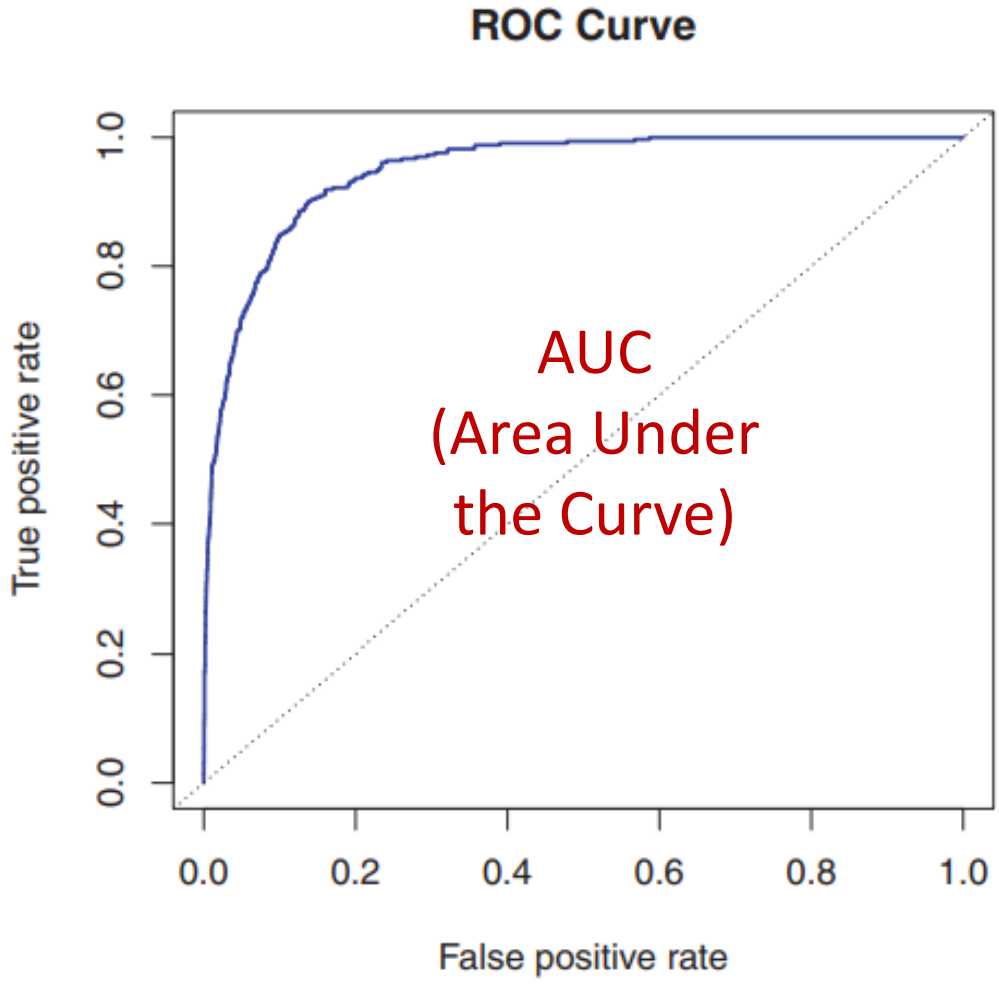
ROC Curve


1.0
0.75
0.5
0.25
0
True positive rate
0
0.25
0.5
0.75
1.0
False positive rate
Learner L1
Learner L2
Learner L3
Random
L3 dominates L2
L2 dominates L3

ROC Curve


L1 always dominates
L2 and L3
True positive rate
False positive rate
1.0
0.75
0.5
0.25
0
0
0.25
0.5
0.75
1.0
Learner L1
Learner L2
Learner L3
Random

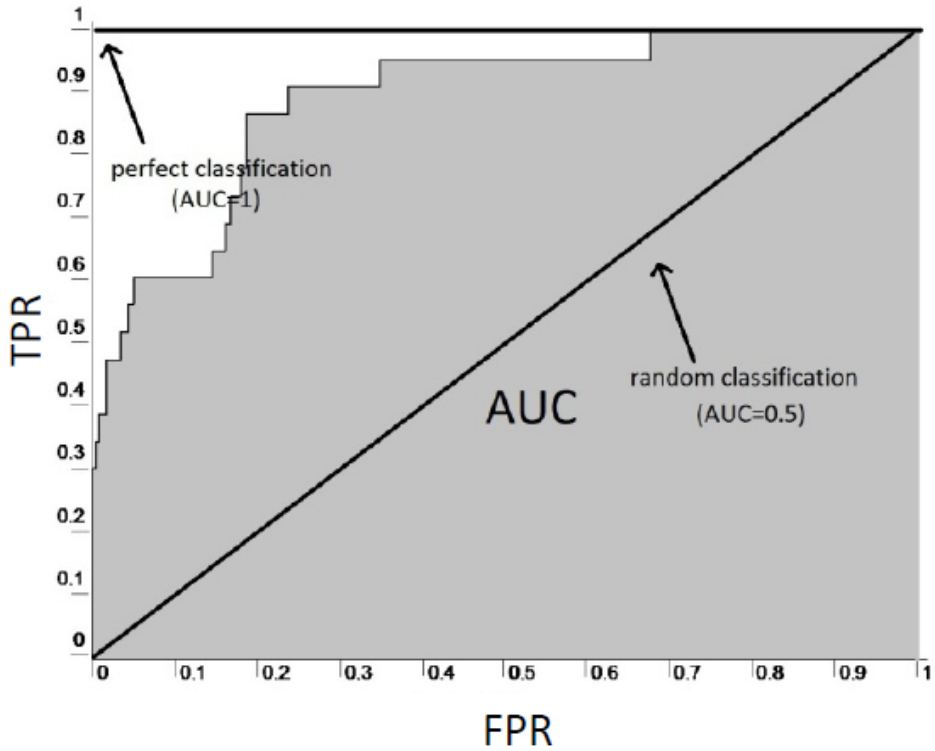
ROC Curves

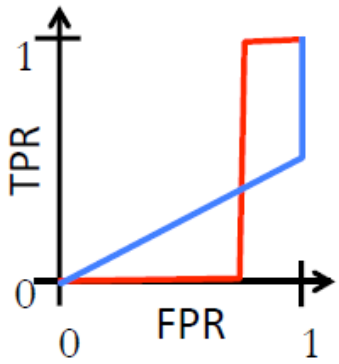


- Another useful metric: Area Under the Curve (AUC)
- The closest to 1, the better!

Area Under the ROC Curve

- Can take area under the ROC curve to summarize performance as a single number
  - Be cautious when you see only AUC reported without a ROC curve; AUC can hide performance issues

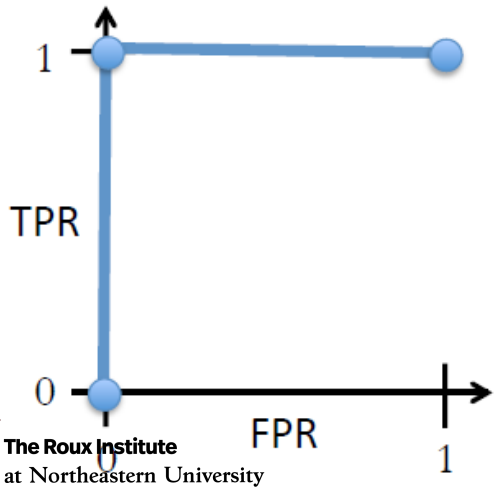




Same AUC, very different performance

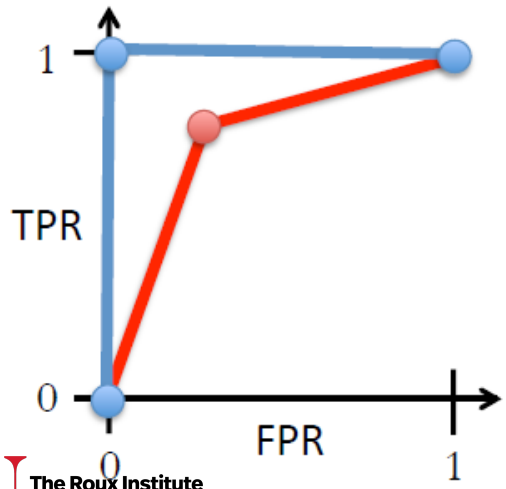
ROC Example

i	y_i	$p(y_i = 1 \mid \mathbf{x}_i)$	$h(\mathbf{x_i} \mid \theta = 0)$	$h(\mathbf{x_i} \mid \theta = 0.5)$	$h(\mathbf{x_i} \mid \theta = 1)$
1	1	0.9	1	1	0
2	1	0.8	1	1	0
3	1	0.7	1	1	0
4	1	0.6	1	1	0
5	1	0.5	1	1	0
6	0	0.4	1	0	0
7	0	0.3	1	0	0
8	0	0.2	1	0	0
9	0	0.1	1	0	0
			$TPR = 5/5 = 1$	$TPR = 5/5 = 1$	$TPR = 0/5 = 0$
			$FPR = 4/4 = 1$	$FPR = 0/4 = 0$	$FPR = 0/4 = 0$



ROC Example

i	y_i	$p(y_i = 1 \mid \mathbf{x}_i)$	$h(\mathbf{x_i} \mid \theta = 0)$	$h(\mathbf{x_i} \mid \theta = 0.5)$	$h(\mathbf{x_i} \mid \theta = 1)$
1	1	0.9	1	1	0
2	1	0.8	1	1	0
3	1	0.7	1	1	0
4	1	0.6	1	1	0
5	1	**0.2**	1	**0**	0
6	0	**0.6**	1	**1**	0
7	0	0.3	1	0	0
8	0	0.2	1	0	0
9	0	0.1	1	0	0



$TPR = 5/5 = 1$ $TPR = 4/5 = 0.8$ $TPR = 0/5 = 0$

$FPR = 4/4 = 1$ $FPR = 1/4 = 0.25$ $FPR = 0/4 = 0$

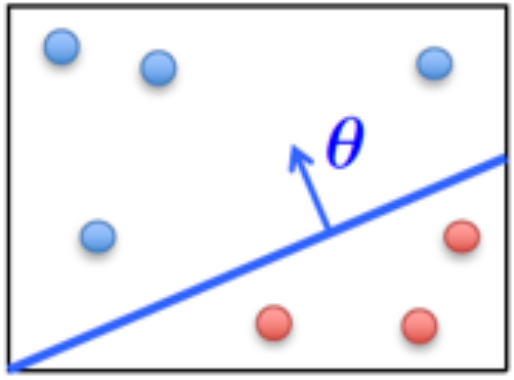
Linear models

- Perceptron

$$h(\boldsymbol{x}) = \text{sign}(\boldsymbol{\theta}^\intercal \boldsymbol{x})$$

- Logistic regression

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^\intercal \boldsymbol{x}}}$$



- LDA

$$Max_k \;\; \delta_k(x) = x \cdot \frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \log(\pi_k)$$

LDA vs Logistic Regression

- Logistic regression computes directly $\Pr[Y = 1|X = x]$ by assuming sigmoid function
  - Uses Maximum Likelihood Estimation
  - Discriminative Model
- LDA uses Bayes Theorem to estimate it
  - Estimates mean, co-variance, and prior from training data
  - Generative model
  - Assumes Gaussian distribution for $f_k(x) = \Pr[X = x|Y = k]$
- Which one is better?
  - LDA can be sensitive to outliers
  - LDA works well for Gaussian distribution
  - Logistic regression is more complex to solve, but more expressive