DS 4400

Machine Learning and Data Mining I
Spring 2024

David Liu

Khoury College of Computer Science

Northeastern University

February 9 2024

Outline

- Review of Cross Validation
- Logistic regression
  - Objective for logistic regression
  - Gradient descent training
  - Regularization
- Logistic regression lab
- Evaluation of classifiers
  - Accuracy, error, precision, recall
  - ROC curves and the AUC metric
  - Why multiple metrics

Announcements

Based on class feedback there will be frequent code demos / examples to complement lecture slides.

Many of these will be the textbook's labs. However, in coming weeks I will likely create my own code demos as well.

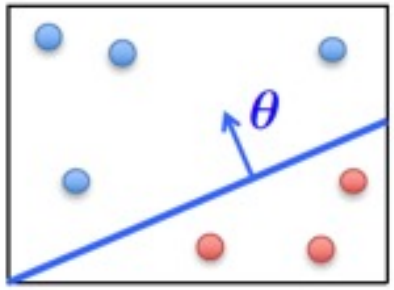
Lab Demo for Cross Validation and Regularization

LOGISTIC REGRESSION

Linear Classifiers

- **Linear classifiers**: represent decision boundary by hyperplane

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix} \quad \boldsymbol{x}^\intercal = \begin{bmatrix} 1 & x_1 & \dots & x_d \end{bmatrix}$$



$h_\theta(x) = f(\theta^T x)$ linear classifier

- If $\theta^T x > 0$ classify "Class 1"
- If $\theta^T x < 0$ classify "Class 0"

All the points x on the hyperplane satisfy: $\theta^T x = 0$

Logistic Regression

- Setup
  - Training data: $\{x_i, y_i\}$, for $i = 1, \ldots, N$
  - Labels: $y_i \in \{0,1\}$
- Goals
  - Learn $h_\theta(x) = P(Y = 1 | X = x)$
- Highlights
  - Probabilistic output
  - At the basis of more complex models (e.g., neural networks)
  - Supports regularization (Ridge, Lasso)
  - Can be trained with Gradient Descent

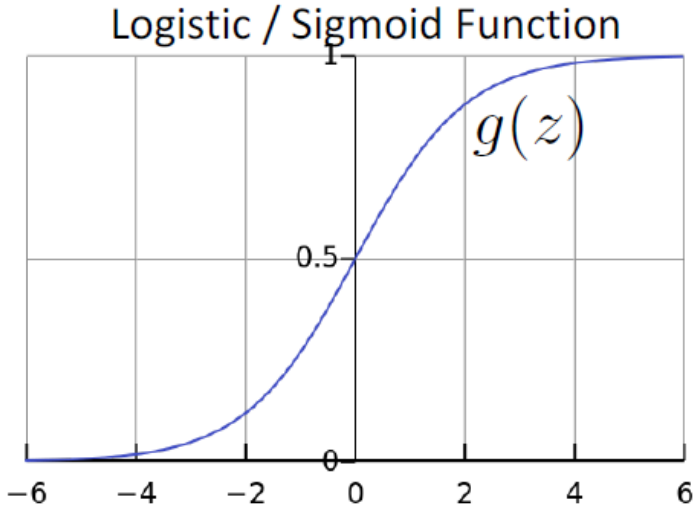
Logistic Regression

- Takes a probabilistic approach to learning discriminative functions (i.e., a classifier)
- $h_{\boldsymbol{\theta}}(\boldsymbol{x})$ should give $P(Y = 1|X; \theta)$
  - Want $0 \leq h_{\boldsymbol{\theta}}(\boldsymbol{x}) \leq 1$
- Logistic regression model:

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = g\left(\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{x}\right)$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

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



Maximum Likelihood Estimation (MLE)

Training data $X = \{x_1, \ldots, x_N\}$, labels $\mathrm{Y} = \{y_1, \ldots, y_N\}$

What is the likelihood of training data for parameter θ?

Define likelihood function $Max_\theta\, L(\theta) = P[Y|X;\theta]$

Assumption: *training labels are conditionally independent*

$$L(\theta) = \prod_{i=1}^{N} P[Y = y_i | X = x_i; \theta]$$

Log likelihood has the same maximum

$$\log L(\theta) = \sum_{i=1}^{N} \log P[Y = y_i | X = x_i; \theta]$$

MLE for Logistic Regression

$$\mathrm{P}(Y = y_i | X = x_i; \theta) = h_\theta(x_i)^{y_i}\left(1 - h_\theta(x_i)\right)^{1-y_i}$$

$$\theta_{MLE} = \mathrm{argmax}_\theta \sum_{i=1}^{N} \log P[Y = y_i | X = x_i; \theta]$$

$$= \mathrm{argmax}_\theta \sum_{i=1}^{N} y_i \log h_\theta(x_i) + (1 - y_i) \log\left(1 - h_\theta(x_i)\right)$$

Logistic Regression Cross-Entropy Loss Objective

$$\min_\theta J(\theta)$$

$$J(\theta) = -\sum_{i=1}^{N} [y_i \log h_\theta(x_i) + (1 - y_i) \log\left(1 - h_\theta(x_i)\right)]$$

Cross-Entropy Objective

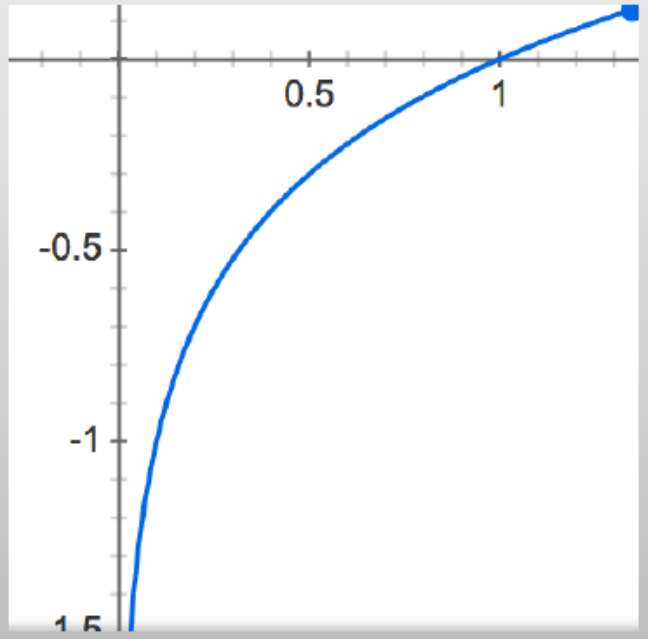
$$J(\theta) = -\sum_{i=1}^{N} [y_i \log h_\theta(x_i) + (1 - y_i) \log \left(1 - h_\theta(x_i)\right)]$$

- Loss of a single instance:

Intuition

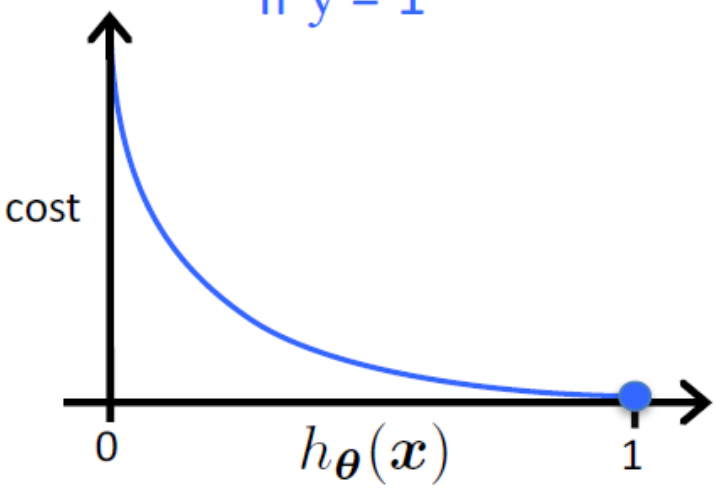
$$\text{loss } (h_{\boldsymbol{\theta}}(\boldsymbol{x}), y) = \begin{cases} -\log(h_{\boldsymbol{\theta}}(\boldsymbol{x})) & \text{if } y = 1 \\ -\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x})) & \text{if } y = 0 \end{cases}$$

Aside: Recall the plot of log(z)



Intuition

$$\text{loss} \quad (h_{\boldsymbol{\theta}}(\boldsymbol{x}), y) = \begin{cases} -\log(h_{\boldsymbol{\theta}}(\boldsymbol{x})) & \text{if } y = 1 \\ -\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x})) & \text{if } y = 0 \end{cases}$$

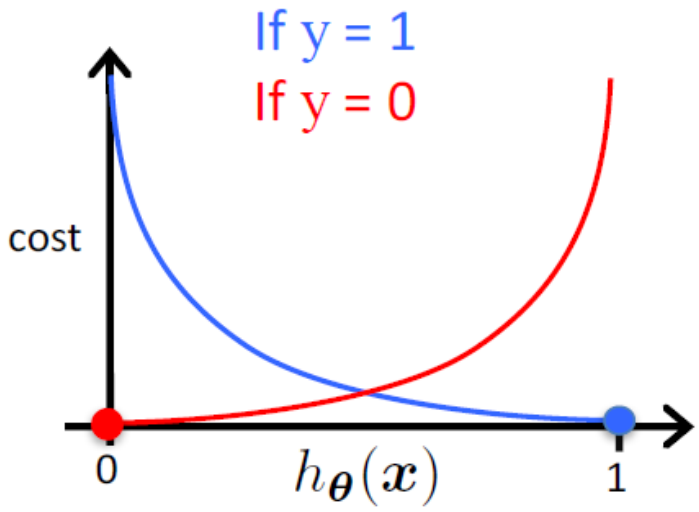


If y = 1

- loss = 0 if prediction is correct
- As $h_{\boldsymbol{\theta}}(\boldsymbol{x}) \to 0$, loss $\to \infty$
- Captures intuition that larger mistakes should get larger penalties
  - e.g., predict $h_{\boldsymbol{\theta}}(\boldsymbol{x}) = 0$, but y = 1

Intuition

$$\text{loss } (h_{\boldsymbol{\theta}}(\boldsymbol{x}), y) = \begin{cases} -\log(h_{\boldsymbol{\theta}}(\boldsymbol{x})) & \text{if } y = 1 \\ -\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x})) & \text{if } y = 0 \end{cases}$$



If y = 0

- loss = 0 if prediction is correct
- As $(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x})) \to 0$, loss $\to \infty$
- Captures intuition that larger mistakes should get larger penalties

Gradient Descent for Logistic Regression

$$J(\theta) = -\sum_{i=1}^{N} [y_i \log h_\theta(x_i) + (1 - y_i) \log \left(1 - h_\theta(x_i)\right)]$$

Want $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

- Initialize $\boldsymbol{\theta}$
- Repeat until convergence

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta})$$

simultaneous update
for j = 0 ... d

Gradient Computation

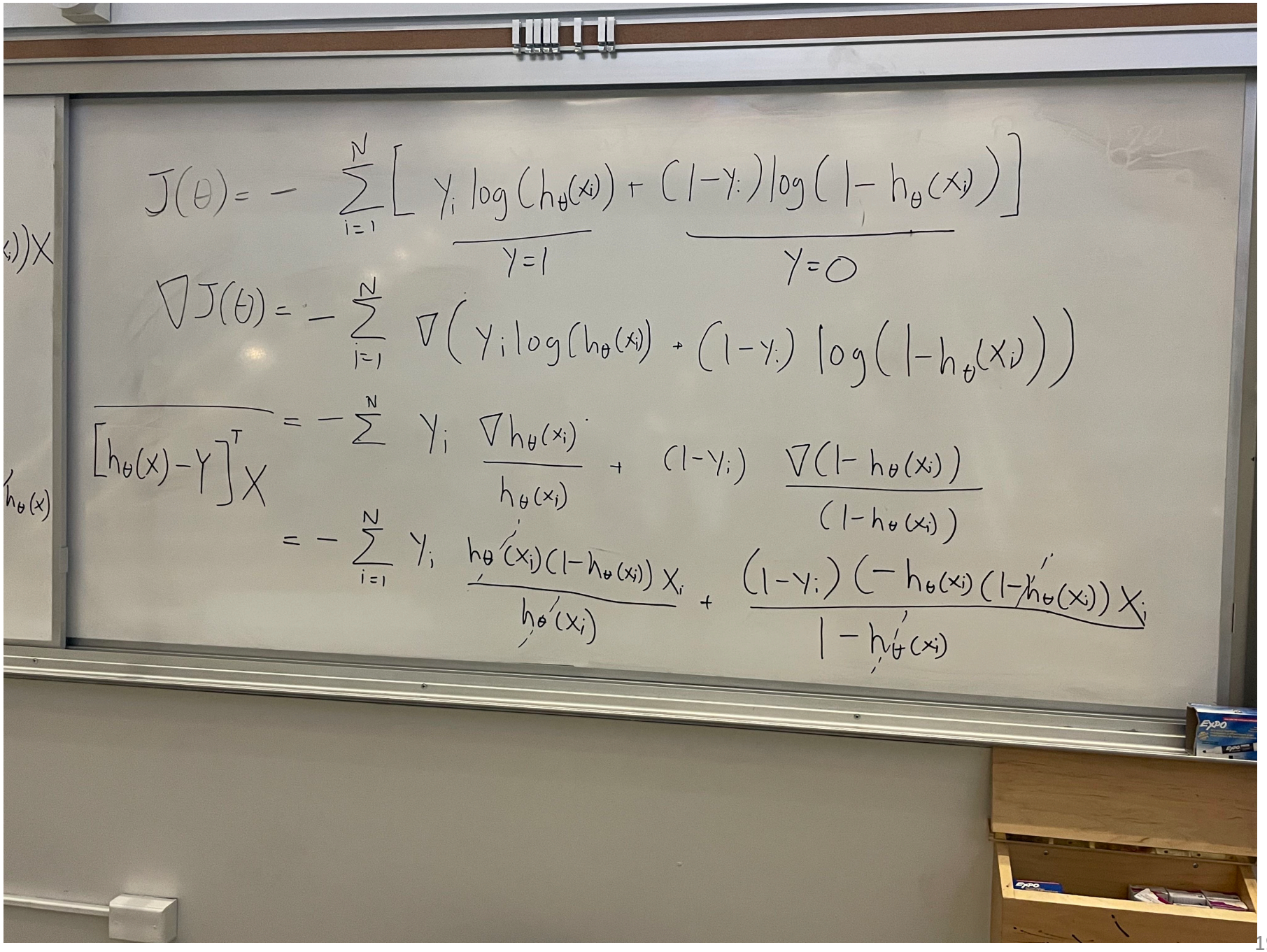
$$J(\theta) = -\sum_{i=1}^{N}[y_i \log h_\theta(x_i) + (1-y_i)\log\left(1-h_\theta(x_i)\right)]$$

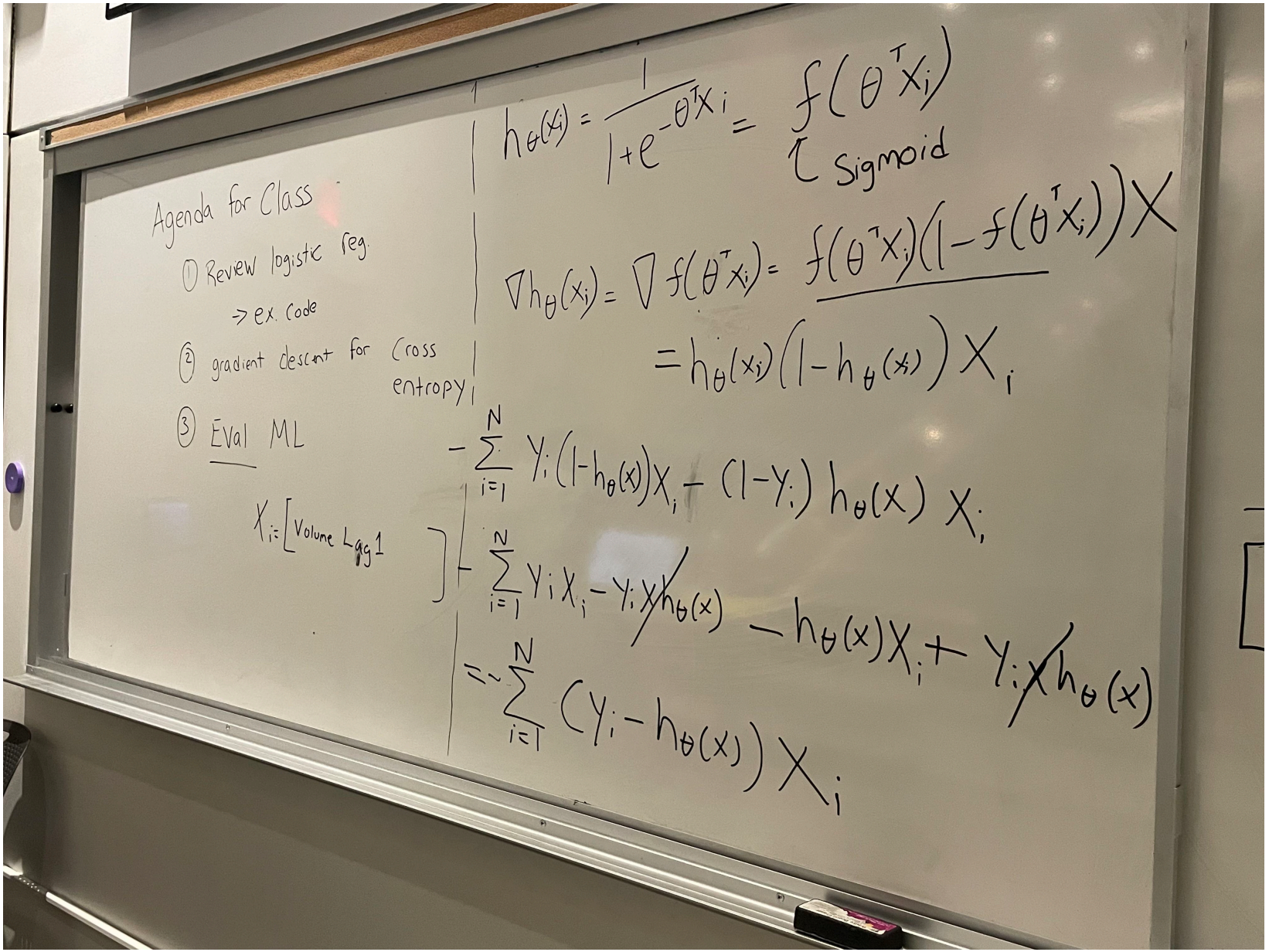
Derivative Facts to Know:

$$\frac{d}{dx}\log x = \frac{1}{x}$$

If $f(\theta^T x) = \frac{1}{1+e\text{^}-\theta^T x}$, then

$$\frac{df}{dx} = f(\theta^T X)\left(1-f(\theta^T X)\right)X$$


$J(\theta) = -\sum_{i=1}^{N} [\underbrace{y_i \log(h_\theta(x_i))}_{y=1} + \underbrace{(1-y_i)\log(1-h_\theta(x_i))}_{y=0}]$
$\nabla J(\theta) = -\sum_{i=1}^{N} \nabla(y_i \log(h_\theta(x_i)) + (1-y_i)\log(1-h_\theta(x_i)))$
$[h_\theta(x) - y]^T X = -\sum^{N} y_i \frac{\nabla h_\theta(x_i)}{h_\theta(x_i)} + (1-y_i)\frac{\nabla(1-h_\theta(x_i))}{(1-h_\theta(x_i))}$
$= -\sum_{i=1}^{N} y_i \frac{h_\theta'(x_i)(1-h_\theta(x_i))x_i}{h_\theta(x_i)} + \frac{(1-y_i)(-h_\theta(x_i)(1-h_\theta(x_i))x_i}{1-h_\theta(x_i)}$


Agenda for Class
① Review logistic reg.
→ ex. code
② gradient descent for Cross entropy
③ Eval ML
$X_i = [\text{Volume Lag1}]$
$h_\theta(x_i) = \frac{1}{1+e^{-\theta^T x_i}} = f(\theta^T x_i)$
↑ Sigmoid
$\nabla h_\theta(x_i) = \nabla f(\theta^T x_i) = f(\theta^T x_i)(1 - f(\theta^T x_i))X$
$= h_\theta(x_i)(1 - h_\theta(x_i))X_i$
$-\sum_{i=1}^{N} y_i(1-h_\theta(x))X_i - (1-y_i)h_\theta(x)X_i$
$-\sum_{i=1}^{N} y_i X_i - y_i X h_\theta(x) - h_\theta(x)X_i + y_i X h_\theta(x)$
$= -\sum_{i=1}^{N} (y_i - h_\theta(x))X_i$

Computing Gradients

- Derivative of sigmoid
  - $g(z) = \frac{1}{1+e^{-z}}$; $g'(z) = \frac{e^{-z}}{(1+e^{-z})^2} = g(z)(1 - g(z))$
- Derivative of hypothesis
  - $h_\theta(x) = g(\theta^T x) = g(\theta_j x_j + \sum_{k \neq j} \theta_k x_k)$
  - $\frac{\partial h_\theta(x)}{\partial \theta_j} = \frac{\partial g(\theta^T x)}{\partial \theta_j} x_j = g(\theta^T x)\big(1 - g(\theta^T x)\big) x_j$
- Derivation of C_i
  - $\frac{\partial C_i}{\partial \theta_j} = y_i \frac{1}{h_\theta(x_i)} \mathrm{g}(\theta^T x_i)\big(1 - \mathrm{g}(\theta^T x_i)\big) x_{ij}$ -

$$(1 - y_i) \frac{1}{1 - h_\theta(x_i)} \mathrm{g}(\theta^T x_i)\big(1 - \mathrm{g}(\theta^T x_i)\big) x_{ij}$$

$$= \big(y_i - h_\theta(x_i)\big) x_{ij}$$

Gradient Descent for Logistic Regression

$$J(\theta) = -\sum_{i=1}^{N}[y_i \log h_\theta(x_i) + (1-y_i)\log\left(1-h_\theta(x_i)\right)]$$

Want $\min_{\theta} J(\theta)$

- Initialize θ
- Repeat until convergence (simultaneous update for j = 0 ... d)

$$\theta_0 \leftarrow \theta_0 - \alpha \sum_{i=1}^{N}(h_\theta(x_i) - y_i)$$

$$\theta_j \leftarrow \theta_j - \alpha \sum_{i=1}^{N}(h_\theta(x_i) - y_i)x_{ij}$$

Gradient Descent for Logistic Regression

Want $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

- Initialize $\boldsymbol{\theta}$
- Repeat until convergence (simultaneous update for j = 0 ... d)

$$\theta_0 \leftarrow \theta_0 - \alpha \sum_{i=1}^{N} (h_\theta(x_i) - y_i)$$

$$\theta_j \leftarrow \theta_j - \alpha \sum_{i=1}^{N} (h_\theta(x_i) - y_i) x_{ij}$$

This looks IDENTICAL to Linear Regression!

- However, the form of the model is very different:

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

Regularized Logistic Regression

$$J(\theta) = -\sum_{i=1}^{N} [y_i \log h_\theta(x_i) + (1 - y_i) \log \left(1 - h_\theta(x_i)\right)]$$

- We can regularize logistic regression exactly as before:

$$J_{\text{regularized}}(\boldsymbol{\theta}) = J(\boldsymbol{\theta}) + \lambda \sum_{j=1}^{d} \theta_j^2$$
$$= J(\boldsymbol{\theta}) + \lambda \|\boldsymbol{\theta}_{[1:d]}\|_2^2$$

L2 regularization

Logistic Regression
Lab Example

Classifier Evaluation

- Classification is a supervised learning problem
  - Prediction is binary or multi-class
- Classification techniques
  - Linear classifiers
    - 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

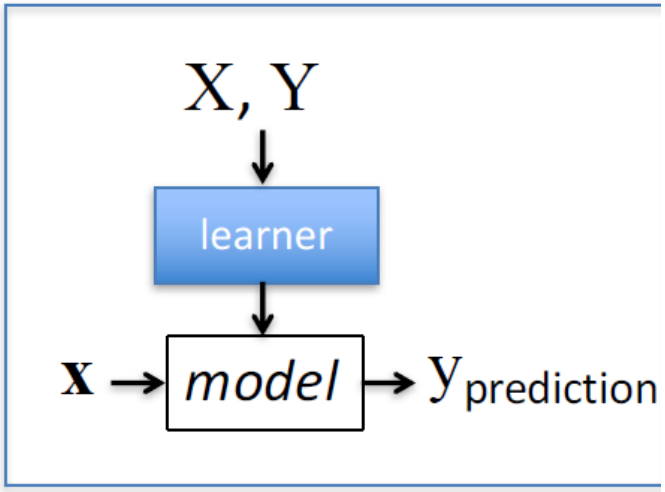
Evaluation of classifiers

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 $\boldsymbol{x} \sim \mathcal{D}(\mathcal{X})$

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

Classification Metrics

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

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

- Can evaluate on both training or testing
- Training set accuracy and error
- Testing set accuracy and error

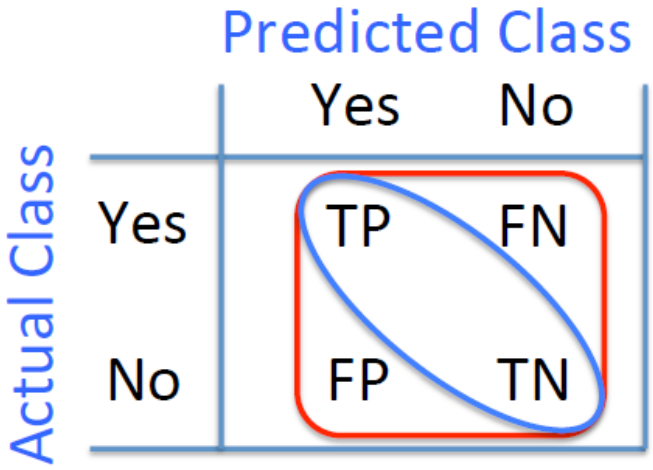
Confusion Matrix

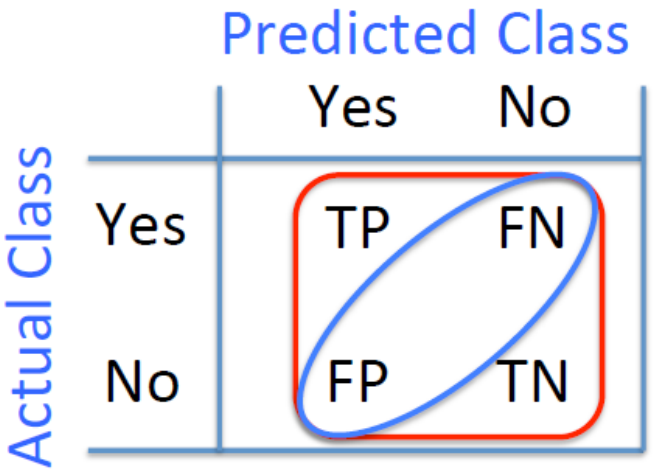
Given a dataset of P positive instances and N negative instances:

		Predicted Class: Yes	Predicted Class: No
Actual Class	Yes	TP	FN
	No	FP	TN

Accuracy and Error

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



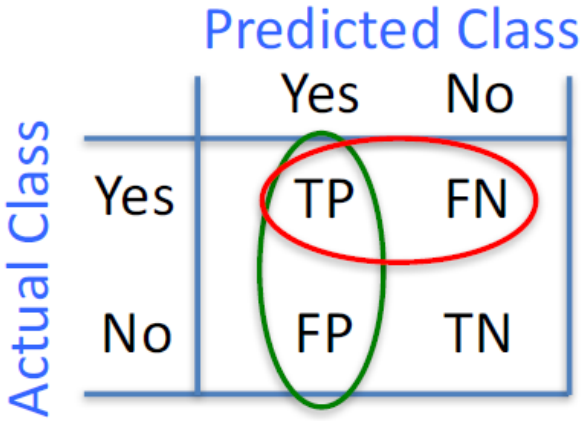


$$\text{accuracy} = \frac{TP + TN}{P + N}$$

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

Confusion Matrix

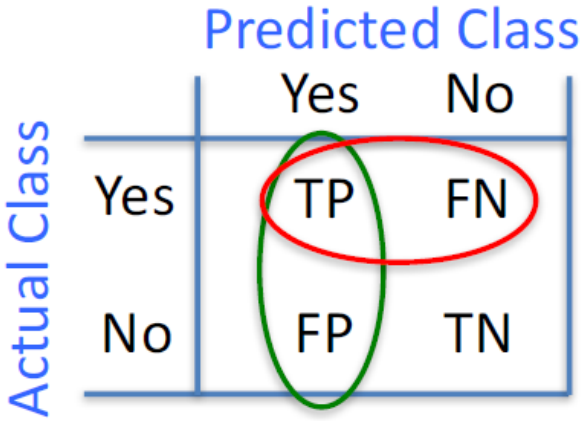
- Given a dataset of P positive instances and N negative instances:



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

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?
- Scenario 2
  - Predict one SPAM email as SPAM, all other emails as legitimate
  - What is the precision?
- Scenario 3
  - Output always SPAM!
  - What is the recall?

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

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
Predictions 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)

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

Classifiers can be tuned

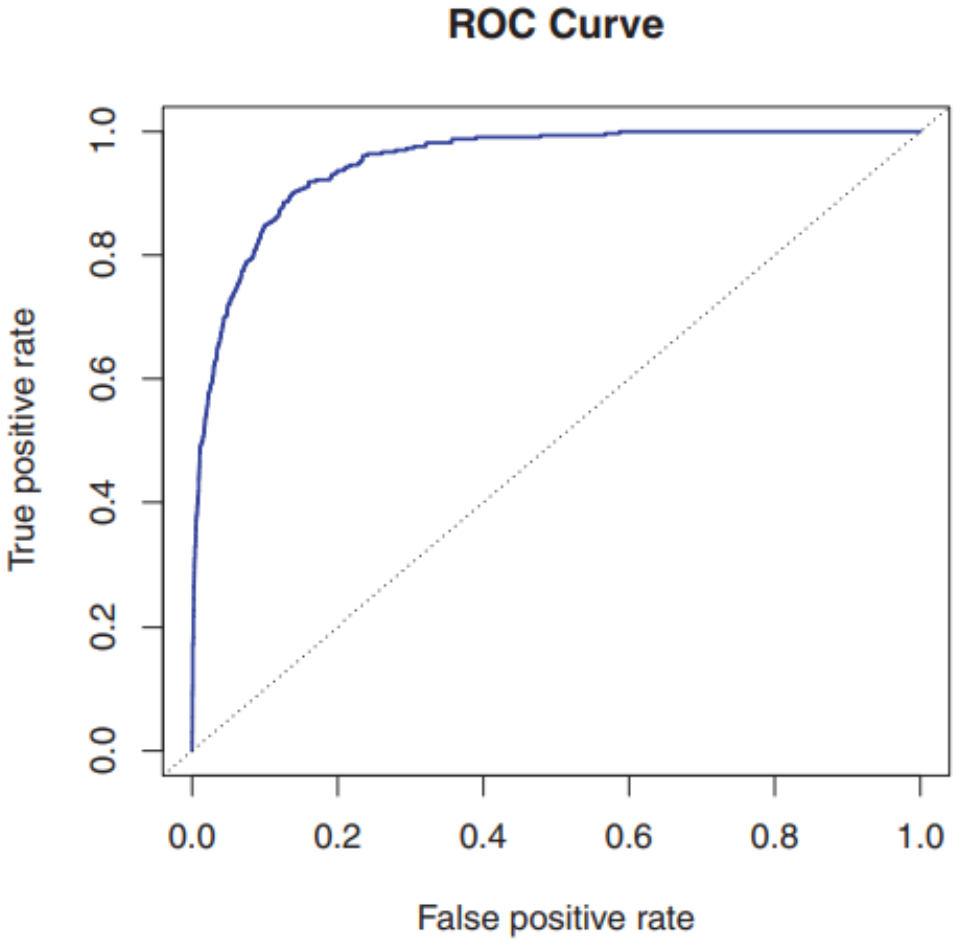
- Logistic regression sets by default the threshold at 0.5 for classifying positive and negative instances
- Some applications have strict constraints on false positives (or other metrics)
  - Example: very low false positives in security (spam)
- Solution: choose different threshold

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

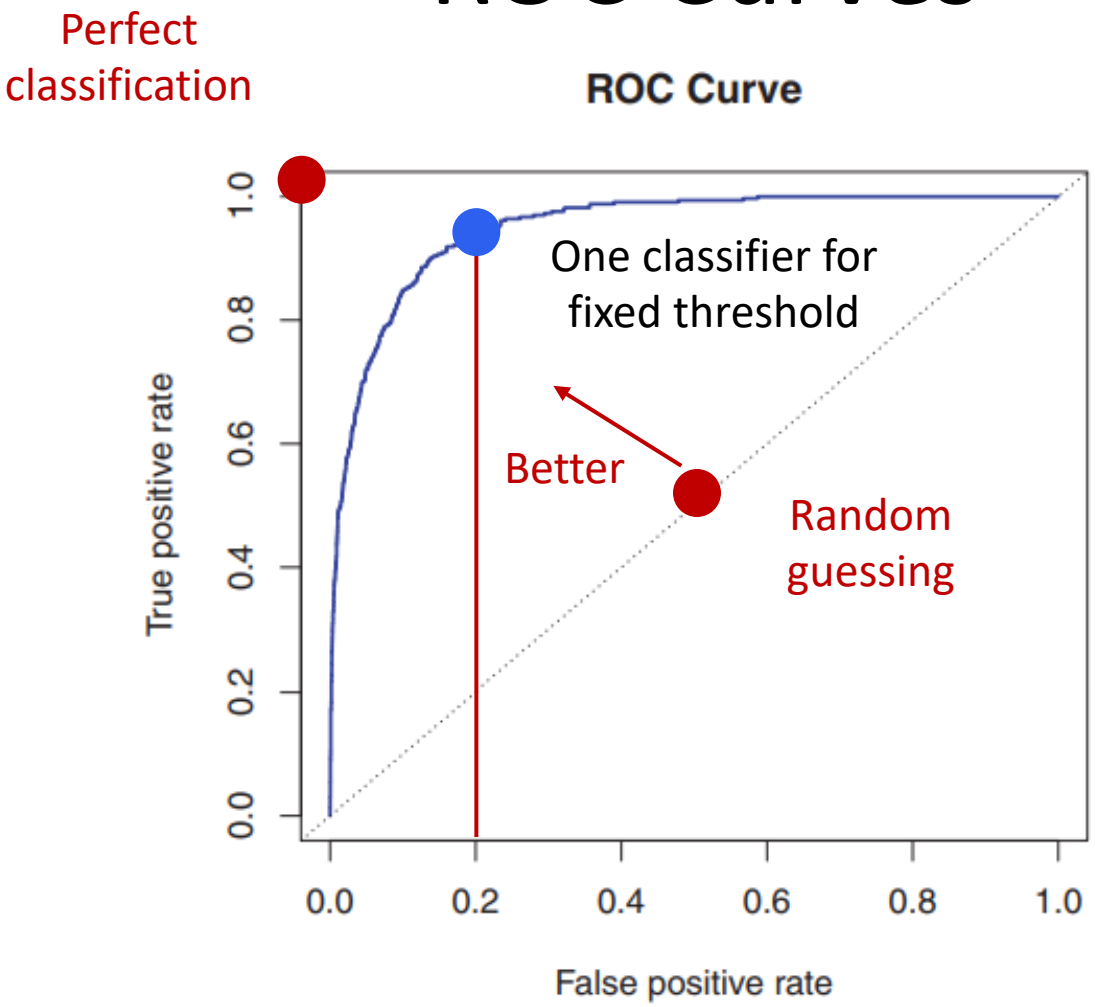
Higher T, lower FP
Lower T, lower FN

ROC Curves



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

ROC Curves

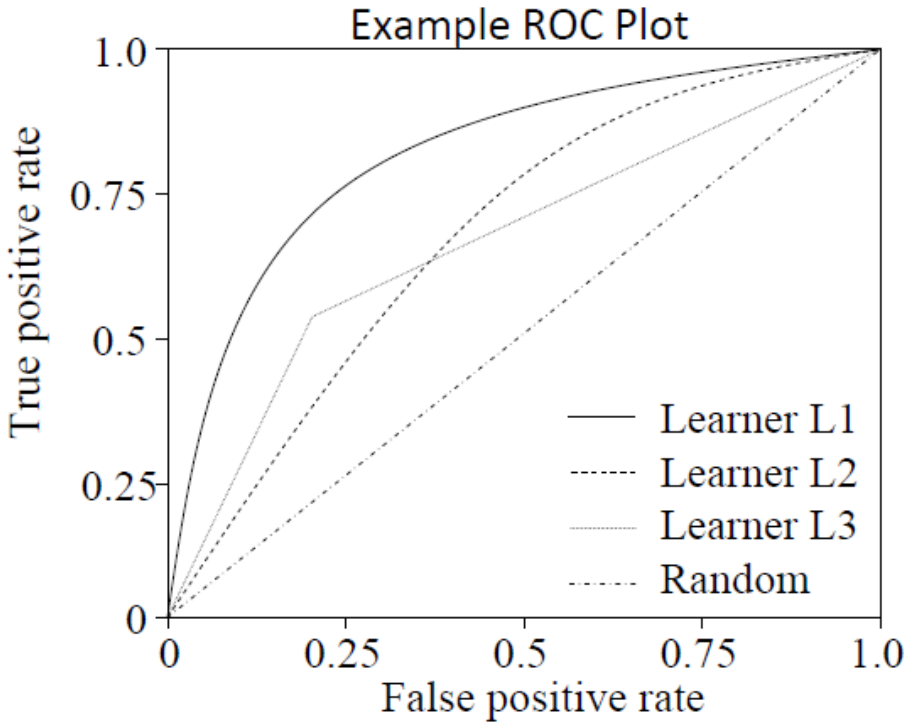


- Receiver Operating Characteristic (ROC)
- Determine operating point (e.g., by fixing false positive rate)

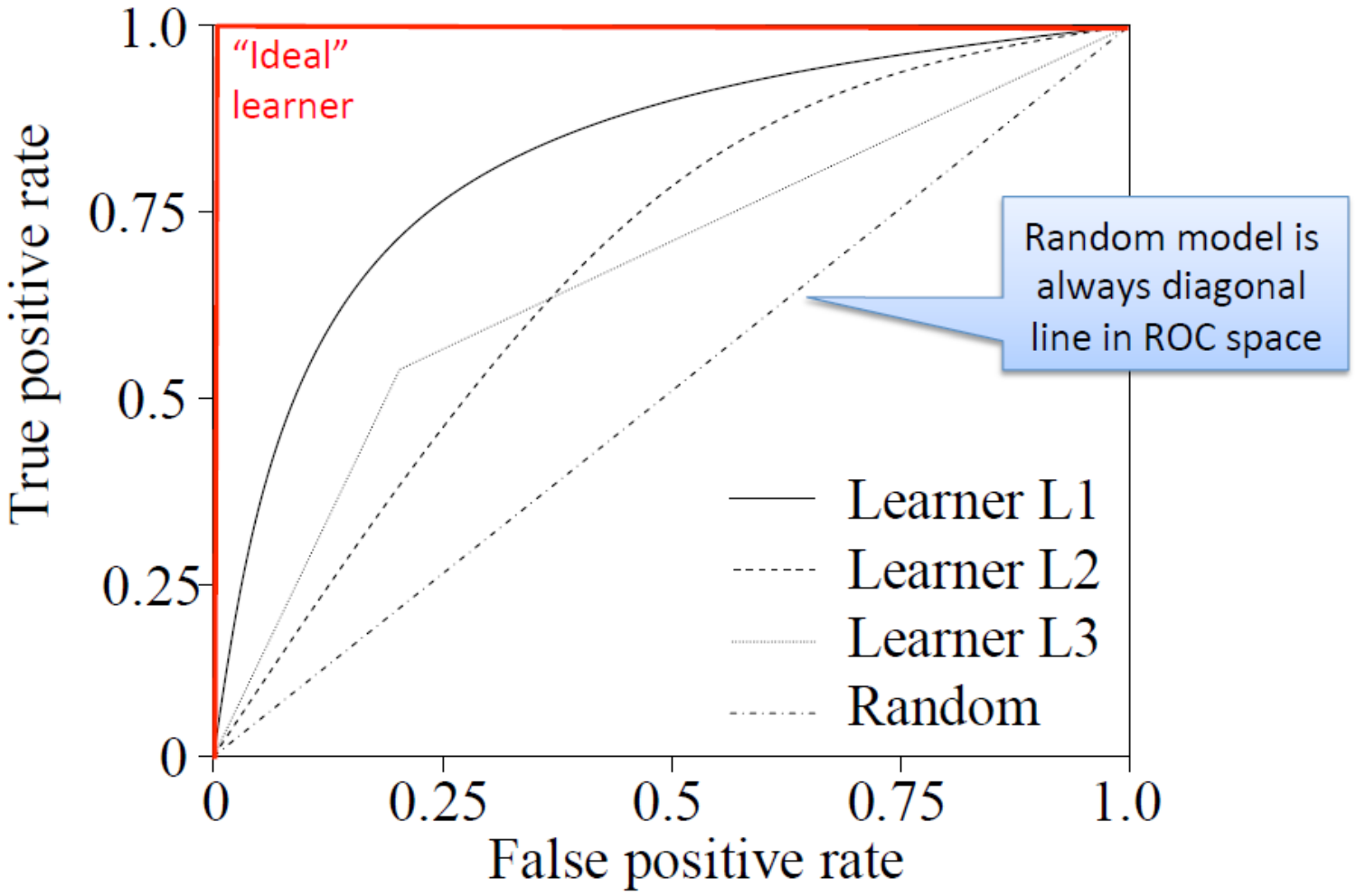
Performance Depends on Threshold

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

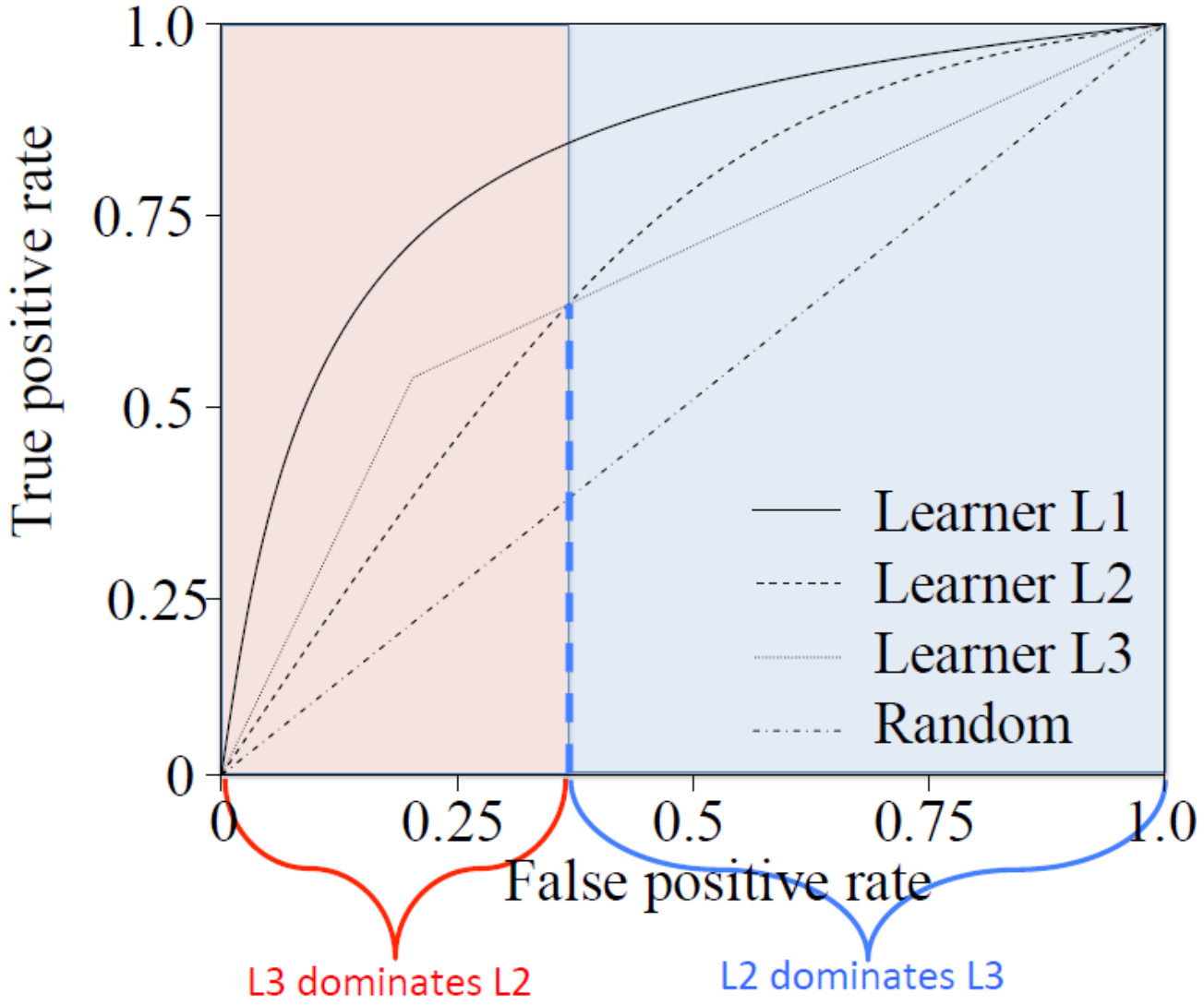
- Number of TPs and FPs depend on threshold τ
- As we vary τ we get different (TPR, FPR) points



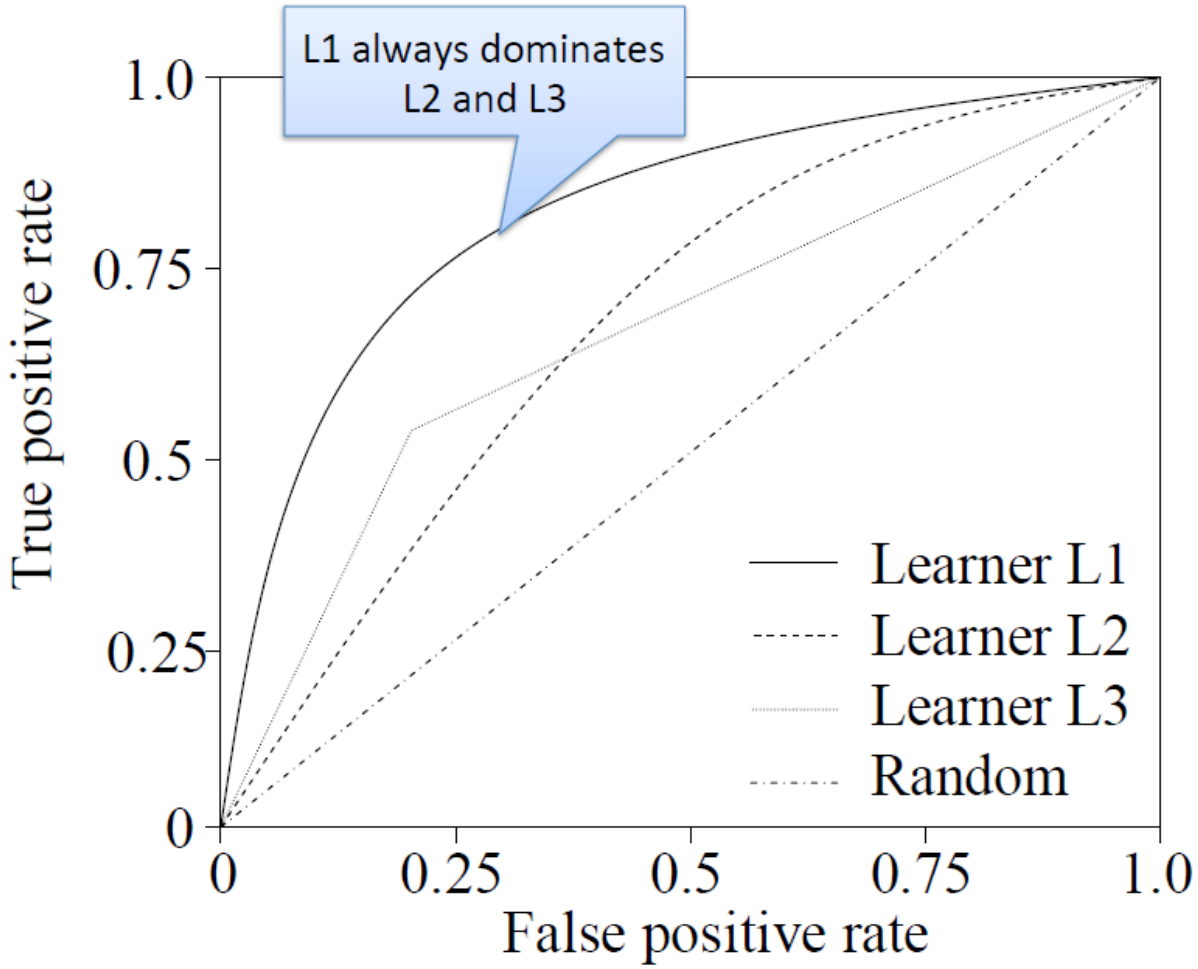
ROC Curve


"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

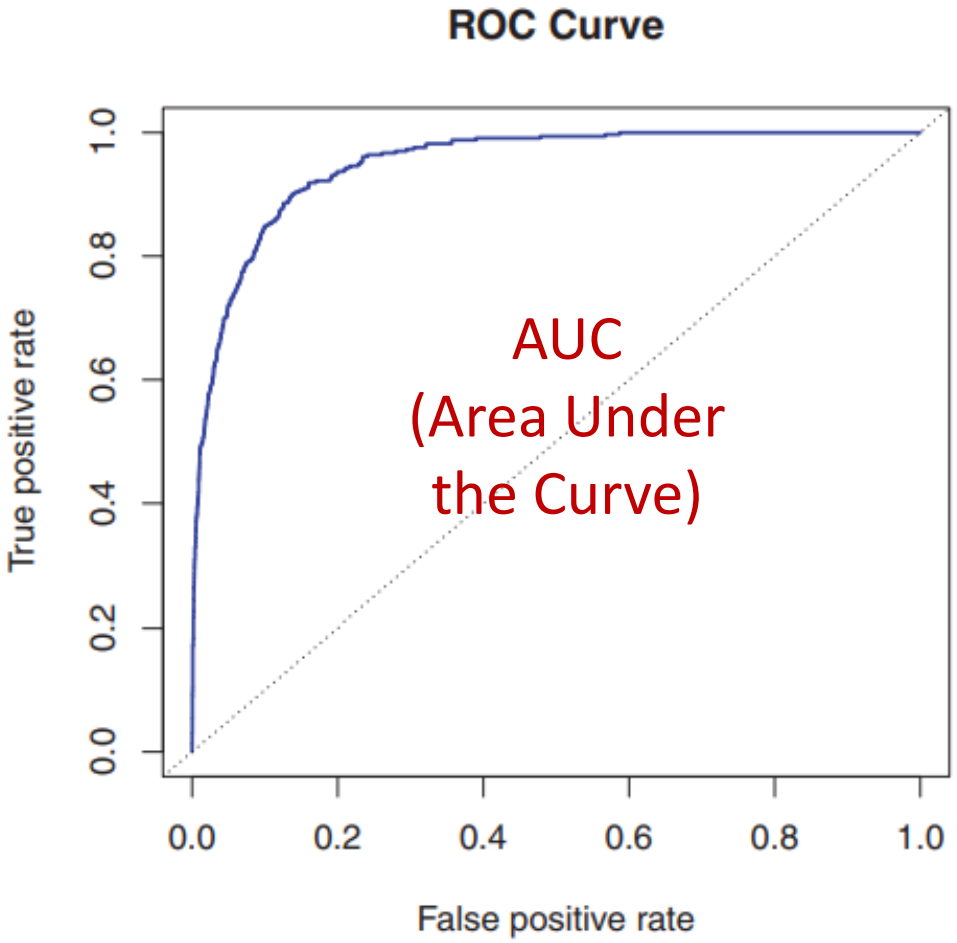
ROC Curve


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

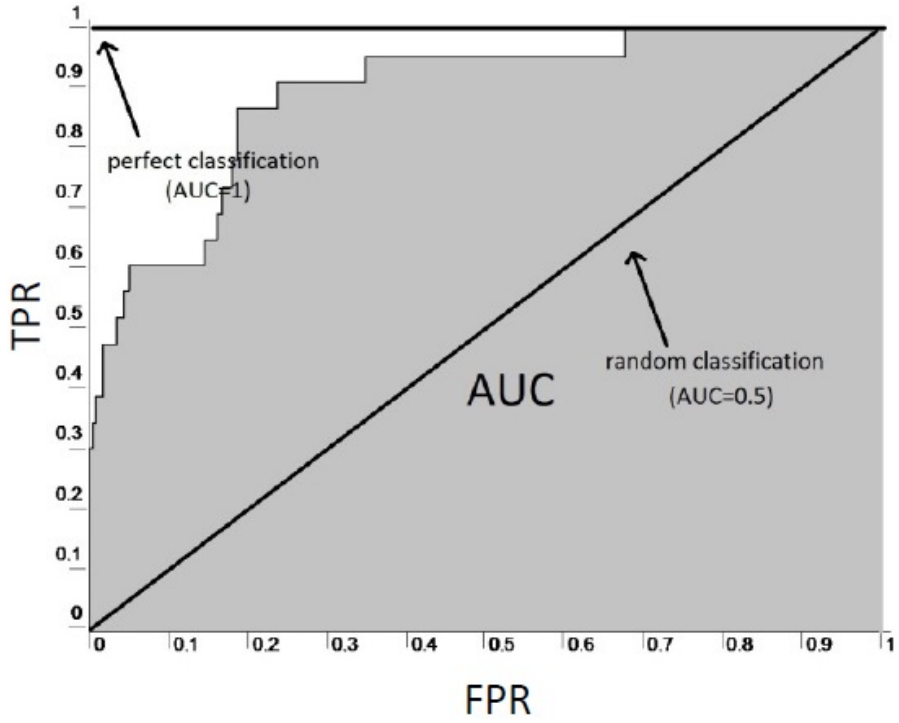
ROC Curves

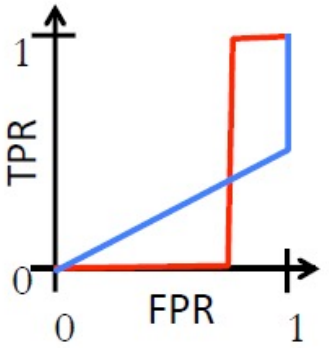


- Another useful metric: Area Under the Curve (AUC)
- The closest to 1, the better!

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





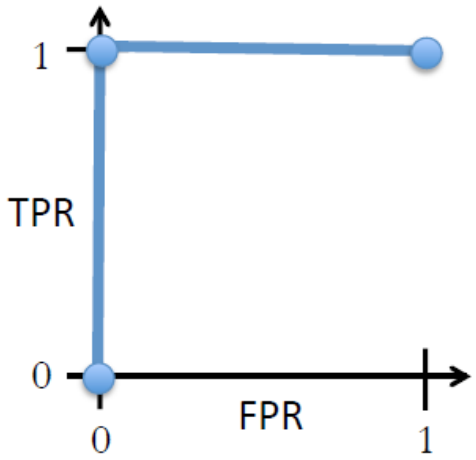
Same AUC, very different performance

ROC Example

i	y_i	$p(y_i = 1 \mid \mathbf{x}_i)$	$h(\mathbf{x_i} \mid T = 0)$	$h(\mathbf{x_i} \mid T = 0.5)$	$h(\mathbf{x_i} \mid T = 1)$
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
			$TPR =$ $FPR =$	$TPR =$ $FPR =$	$TPR =$ $FPR =$

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

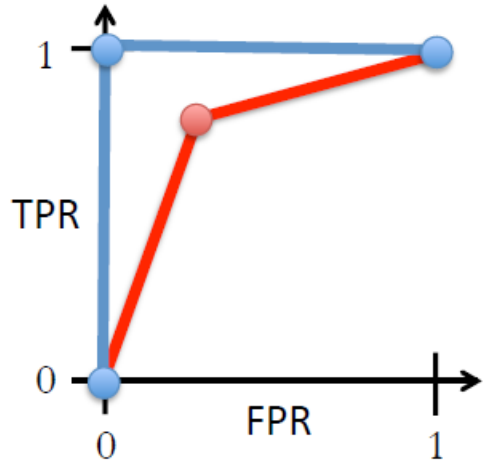


ROC Example

i	y_i	$p(y_i = 1 \mid \mathbf{x}_i)$	$h(\mathbf{x_i} \mid T = 0)$	$h(\mathbf{x_i} \mid T = 0.5)$	$h(\mathbf{x_i} \mid T = 1)$
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
			$TPR =$	$TPR =$	$TPR =$
			$FPR =$	$FPR =$	$FPR =$

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



$TPR = 5/5 = 1$ $\quad FPR = 4/4 = 1$

$TPR = 4/5 = 0.8$ $\quad FPR = 1/4 = 0.25$

$TPR = 0/5 = 0$ $\quad FPR = 0/4 = 0$

Review

- Maximum Likelihood Estimation (MLE) is a general statistical method for parameter estimation
- Logistic regression is a linear classifier that predicts class probability
  - Cross-entropy objective derived with MLE
- Can be trained with Gradient Descent
- Multiple metrics for classifier evaluation
  - Accuracy, error, precision, recall, F1 score
  - RIC curves and AUC