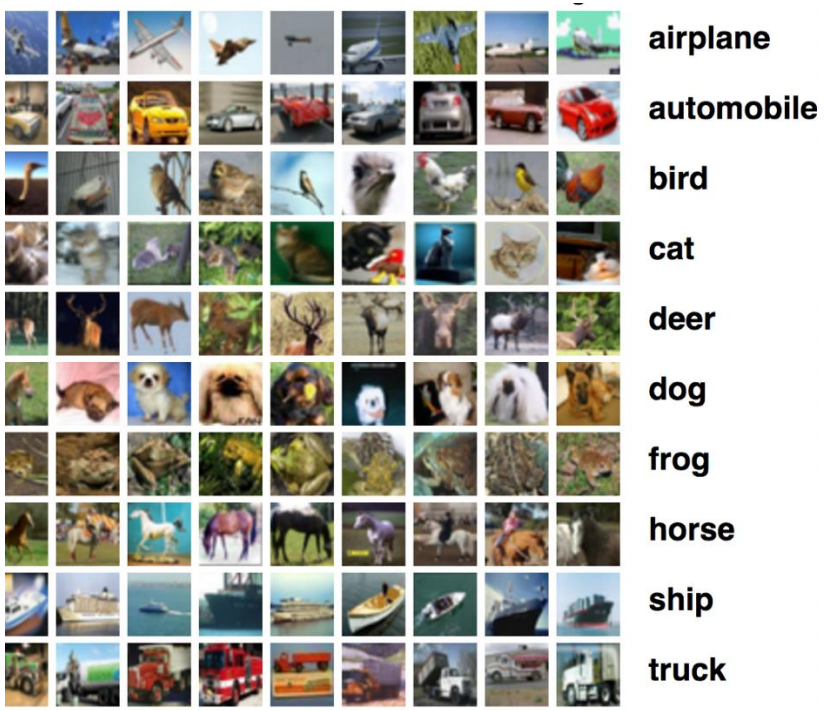
ECE 4252/8803: Fundamentals of Machine Learning (FunML)
Fall 2024

Lecture 6: Classifier Performance Evaluation



OLIVES @GeorgiaTech

Georgia Tech

Reminders

- Reminders
  - HW2 is due on Friday, 13-Sept

Classifier Comparison

Methods	Assumptions on Feat. Dist.	Feat. Normalization	Cost Function	Regularization	Linear Classifier	Prob. View of Prediction	Generative/Discriminative	Parametric/ Non-parametric	Overfitting
Logistic Regression	No	Required	BCE (convex)	Additional term	Linear	Yes	Discriminative	Parametric	Not often
K Nearest Neighbors	No	Required	N/A	N/A	Non-linear	N/A	Discriminative	Non-parametric	when k is too small
Decision Trees	No	Not Required	N/A	N/A	Non-linear	N/A	Discriminative	Non-parametric	with large depth
Support Vector Machines	No	Required	Hinge (convex)	C (control robustness)	Linear/ Non-linear(kernel)	N/A	Discriminative	Parametric	Not often
Naïve Bayes	Conditional independent	Not Required	N/A	N/A	Non-linear /Linear (Gaussian)	Yes	Generative	Parametric	Not often
Artificial Neural Networks	No	Required	Non-convex	Additional term	Non-linear	Yes	Discriminative/ Generative	Parametric	with many layers

Overview

In this Lecture..

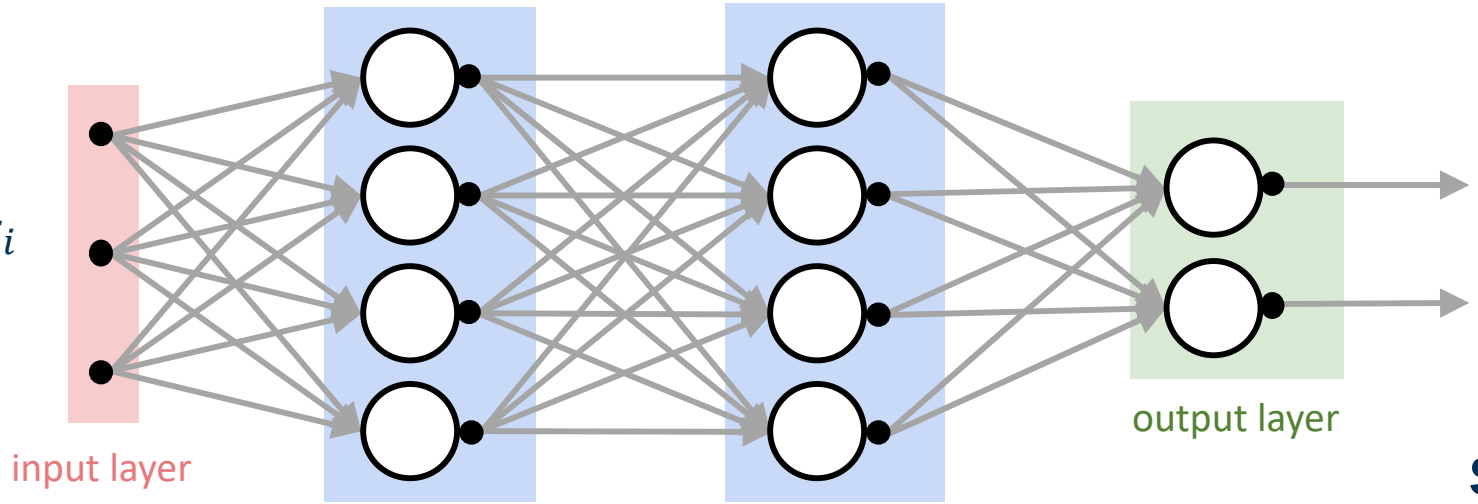
- Nearest Neighbor
- Naïve Bayes
- Logistic Regression
- Decision Trees
- Support Vector Machines
- Artificial Neural Networks
  - Overview
  - Activation Function
  - Perceptron Network
  - Multi-layer ANN
  - Feedforward and Backward Error Propagation
  - Learning Algorithm
  - Image Classification using ANNs

Artificial Neural Networks

Backpropagation Summary

During the inference, predict the class with the highest probability with SoftMax activation: $\mathbf{argmax}_j\, f(y_{ij})$

Sample x_i



Assume the raw outputs $y_i = [y_{i1}, y_{i2}, y_{i3}] = [3.2, 5.1, -1.7]$

Raw outputs

y_{i1}= 3.2
y_{i2}= 5.1
y_{i3}= -1.7

exp ⇨

unnormalized probabilities

24.5
164.0
0.18

normalize ⇨

probabilities

$f(y_{i1}) = 0.13$
$f(y_{i2}) = 0.87$
$f(y_{i3}) = 0.00$

SoftMax activation:

$$f(y_{ij}) = \frac{e^{y_{ij}}}{\sum_k e^{y_{ik}}}$$

predict class 2 with highest probability

Artificial Neural Networks

Backpropagation Summary

During training, for every sample, we set the ground-truth label as a one-hot vector $[1, 0, \dots, 0]^{\mathrm{T}}$ with 1 for the correct class and 0 for every other class. Backpropagate the error and repeat for every sample.

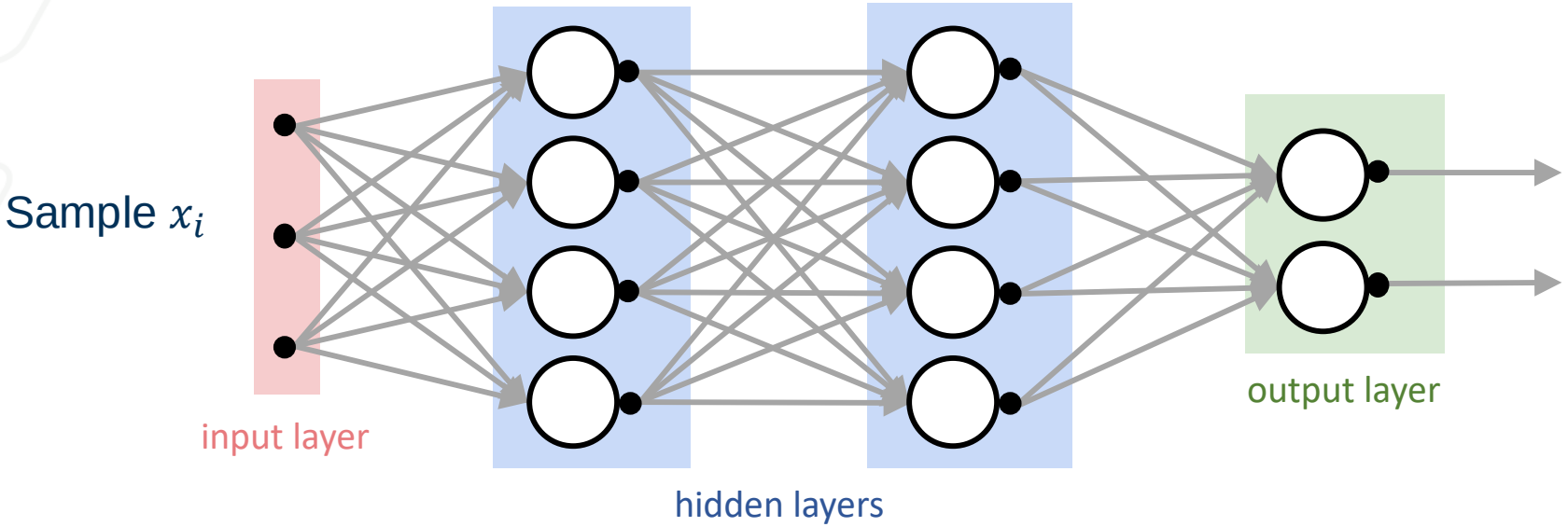


Image Classification: mapping the image pixels to probabilities for each category

For simplicity, we take a linear function:

$$\widehat{\boldsymbol{Y}} = \boldsymbol{\phi}(\boldsymbol{X}\boldsymbol{W}^T + \boldsymbol{b}^T)$$

where

- $\boldsymbol{X} \in \mathbb{R}^{N \times P}$: dataset containing N vectorized images
- $P \in \mathbb{R}^{H \times W \times C}$: the number of pixels (*features*) of each image
- $\widehat{\boldsymbol{Y}} \in \mathbb{R}^{N \times P^{(k)}}$: the associated probabilities of each category $1,2, \dots, P^{(k)}$
- $\boldsymbol{\phi}(\boldsymbol{X}, \boldsymbol{W}, \boldsymbol{b})$: the activation function
- $\boldsymbol{W} \in \mathbb{R}^{P \times P^{(k)}}$: the weight matrix
- $\boldsymbol{b} \in \mathbb{R}^{P^{(k)} \times 1}$: the bias vector

Single image binary classification (predicting dog/cat)

stretch pixels into single column vector

Input 32x32 RGB image

$$\phi\left(\begin{bmatrix} 0.2 & 2.1 \\ -0.5 & 0.0 \\ 0.1 & 0.25 \\ 2.0 & 0.2 \\ 1.5 & -0.3 \\ \vdots & \vdots \\ 1.3 & 1.2 \end{bmatrix}^{T} \begin{bmatrix} 56 \\ 231 \\ 24 \\ 188 \\ 75 \\ \vdots \\ 32 \end{bmatrix} + \begin{bmatrix} 0.2 \\ 2.4 \end{bmatrix}\right) = \begin{bmatrix} 0.8 \\ 0.2 \end{bmatrix} \begin{matrix} \text{Cat score} \\ \text{Dog score} \end{matrix}$$

$b \in \mathbb{R}^{2\times 1}$ $y_i \in \mathbb{R}^{2\times 1}$

$w^T \in \mathbb{R}^{2\times(32)(32)(3)}$ $x_i \in \mathbb{R}^{(32)(32)(3)\times 1}$

Artificial Neural Networks

Training MLP for Image Classification

Example Datasets:

- MNIST (hand-written digits)
  - # total: 70,000 grayscale images
  - # classes: 10
  - # size: 28x28
  - # training samples: 60,000 images
  - # test samples: 10,000 images

https://www.researchgate.net/figure/Sample-images-of-MNIST-data_fig3_222834590

- CIFAR-10 (subsets of the 80 million tiny images)
  - # total: 60,000 color images
  - # classes: 10
  - # size: 32x32
  - # training samples: 50,000 images
  - # test samples: 10,000 images

https://www.kaggle.com/c/cifar-10

Overview

In this Lecture..

- Cross-validation
- Precision and Recall
- Confusion Matrix
- Precision/Recall Tradeoff
- The ROC Curve
- Error Analysis

Terminologies

- Batch size: split the data into several batches
- Epoch: the model going through the entire data
- Round: the model goes through a feed forward and a backpropagation
- Validation set versus training set versus testing set

Terminologies

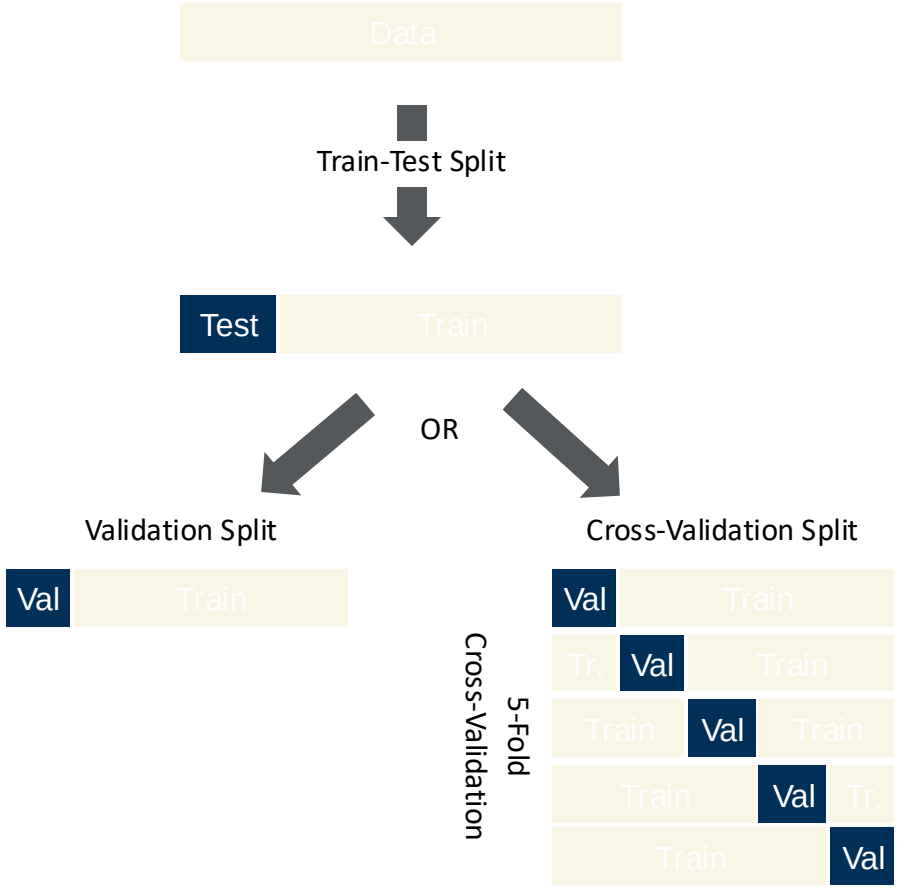
Model Validation

- Training/Validation/Test Split
  - Testing subset kept away and never used during model training
  - Training subset used for training and typically split further into Training/Validation
  - Validation subset for model validation
- Learning Curves
  - Show how *training* and *validation* scores compare as a function of increasing training set size
  - Gives insight on a model's generalization ability
- Cross-Validation
  - Used for assessing how the training results of a model will generalize to unseen data

Terminologies

Train/Test/Validation Split

- Test Dataset
  - Only used once a model is completely trained (using the train and validation sets)
  - Generally used to evaluate competing models
  - Contains data that spans the various classes in the real world.
- Training Dataset
  - The actual dataset used to train the model.
  - The model sees and learns from this data
- Validation Dataset
  - Used to evaluate a given model and fine-tune its *hyperparameters*
  - The model sees this data during training but never learns directly from it
  - Typically implemented as cross-validation
- Split ratio
  - Typically: 20%-30% test, and the remaining training/validation

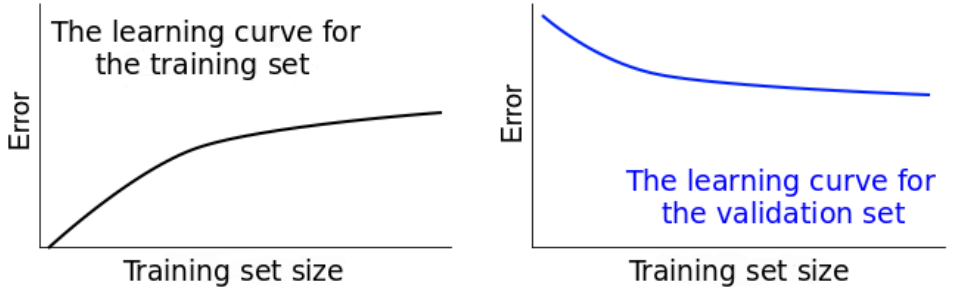


Terminologies

Learning Curves

- Training error starts very low when training set is small, and increases as more training is data added
- Validation error starts high and decreases as more training data added.
- The general procedure for generating learning curves is as follows (assuming a dataset of size $n = 100$ samples):

1. Set aside validation set (e.g., $v = 20$ samples)
2. For $k = 1$ to $n - v$
   1. Take the first k samples as one training dataset
   2. Fit the model on the training set and evaluate it on the validation set
   3. Retain the training score and the evaluation score and discard the model
3. Plot the training and evaluation scores recorded in the iterations above against training set sizes



Terminologies

Visualization

- Check this tool

Performance Evaluation

Hyperparameters

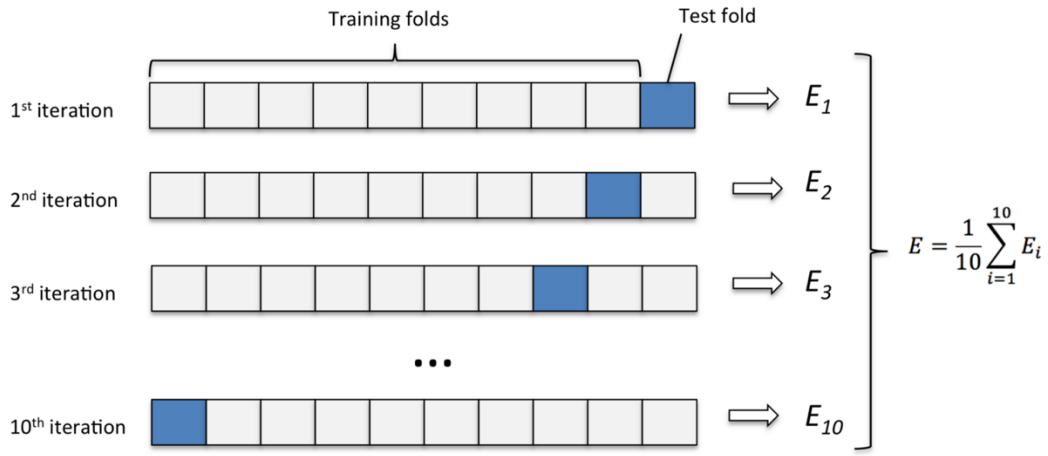
- A hyperparameter is a parameter used **to control the learning process** and have an **impact on the model performance**.
- Hyperparameters need to be **initialized before training.**
- Common hyperparameters:
  - k Nearest Neighbor:
    - Number of neighbors (k): a too small value will cause overfitting while a too large value will increase the bias towards the major classes.
  - Support Vector Machines:
    - Regularization parameter (C): a smaller value allows larger margin and enhances robustness to outliers
  - Artificial Neural Networks:
    - Number of hidden layers (K): a larger value increases model capacity to learn from massive data on complex tasks
    - Learning rate (α): a too high value will cause unstable learning while a too low value will either take too long to converge or stay in an undesirable local minimum.

Performance Evaluation

Cross-validation

- Helps determine hyperparameters?
- More rigorous and randomized than single validation split
- A cross-validation procedure is performed for each combination of hyperparameters
- The general procedure is as follow:

1. Shuffle the dataset randomly
2. Split the dataset into k groups
3. For each group
   1) Take that group as a hold-out or validation dataset
   2) Combine the remaining k-1 groups as one training dataset
   3) Fit a model on the training set and evaluate it on the validation (hold-out) set
   4) Retain the evaluation scores and discard the model
4. Average the scores of the model to get a single k-fold validation score



E_i is validation error

Source: Medium - Python Model Tuning Methods Using Cross Validation and Grid Search

Overview

In this Lecture..

- Cross-validation
- Precision and Recall
- Confusion Matrix
- Precision/Recall Tradeoff
- The ROC Curve
- Error Analysis

Performance Evaluation

Accuracy

- Accuracy: The percentage of correctly labeled samples

$$Accuracy = \frac{\#\ of\ correctly\ labled\ samples}{Total\ \#\ of\ samples}$$

$$= \frac{7}{10} = 0.7$$

Question: Let's say your dataset has 1,000,000 samples. Also assume that 90% of them have label (T).

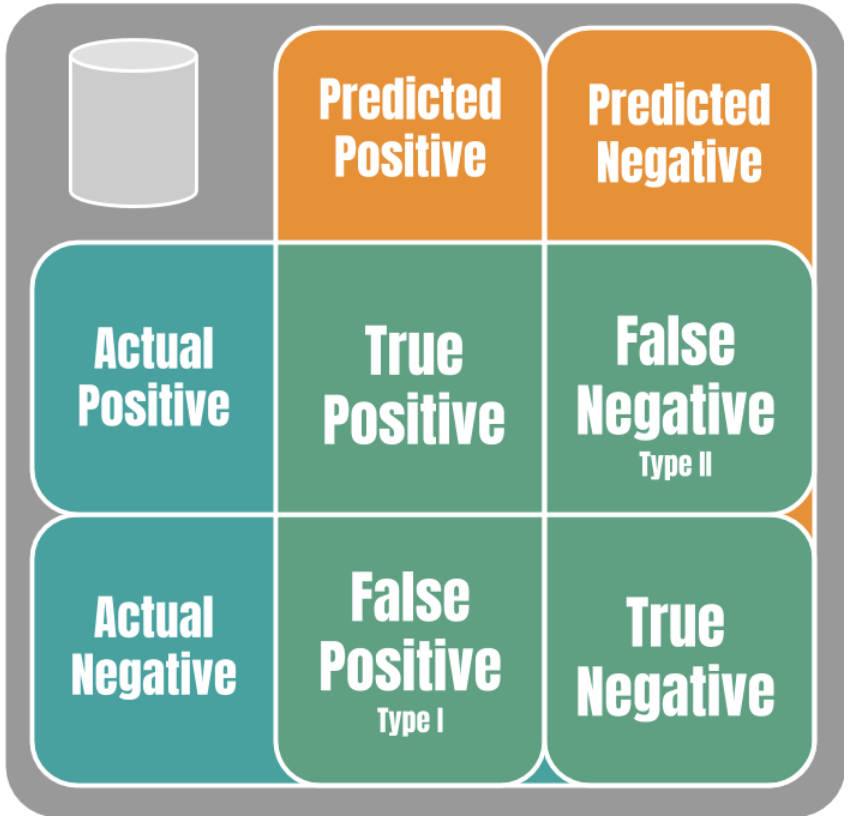
A lazy friend of yours designs a very simple system that guesses True all the time. What's the accuracy?

Q#	Answer Key	Your Answer	Grading
1	F	T	✗
2	F	F	✓
3	T	T	✓
4	F	F	✓
5	T	T	✓
6	T	F	✗
7	T	T	✓
8	F	F	✓
9	F	T	✗
10	F	F	✓

Performance Evaluation

Beyond accuracy: TP, FP, TN, and FN

- TP – True Positive:
  Positive AND successfully accepted
- FP – False Positive:
  Negative, BUT mistakenly accepted
- TN – True Negative:
  Negative, AND successfully rejected
- FN – False Negative:
  Positive, BUT mistakenly rejected



Performance Evaluation

Beyond accuracy: TP, FP, TN, and FN

- **TP (3)** – True Positive:
  Positive AND successfully accepted
- **FP (2)** – False Positive:
  Negative, BUT mistakenly accepted
- **TN (4)** – True Negative:
  Negative AND successfully rejected
- **FN (1)** – False Negative:
  Positive, BUT mistakenly rejected

Q#	Answer Key	Your Answer	Grading	?
1	F	T	✗	FP
2	F	F	✓	TN
3	T	T	✓	TP
4	F	F	✓	TN
5	T	T	✓	TP
6	T	F	✗	FN
7	T	T	✓	TP
8	F	F	✓	TN
9	F	T	✗	FP
10	F	F	✓	TN

Performance Evaluation

Precision, Recall, and F1 Scores

		Predicted Class P	Predicted Class N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

Precision is a good measure, when the cost of FP is high.

- **Precision**: how many selected items are relevant? How many of the selected +ve's are truly +ve?

$$precesion = \frac{TP}{TP + FP} = \frac{3}{5} = 0.6$$

Recall is a good measure, when the cost of FN is high.

- **Recall**: how many relevant items are selected? How many of the Actual +ve's has our model labeled as +ve (TP)?

$$recall = \frac{TP}{TP + FN} = \frac{3}{4} = 0.75$$

F1 is a balance between the two measures and when the classes are uneven.

- **F1 score**: Combines precision and recall:

$$F = 2 \times \frac{precision \times recall}{precision + recall} = 2 \times \frac{0.6 \times 0.75}{0.6 + 0.75} = 0.667$$

Q#	Answer Key	Your Answer	Grading	?
1	F	T	✗	FP
2	F	F	✓	TN
3	T	T	✓	TP
4	F	F	✓	TN
5	T	T	✓	TP
6	T	F	✗	FN
7	T	T	✓	TP
8	F	F	✓	TN
9	F	T	✗	FP
10	F	F	✓	TN

Performance Evaluation

Confusion Matrix and Multi-class Classification

		Predicted Class: P	Predicted Class: N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

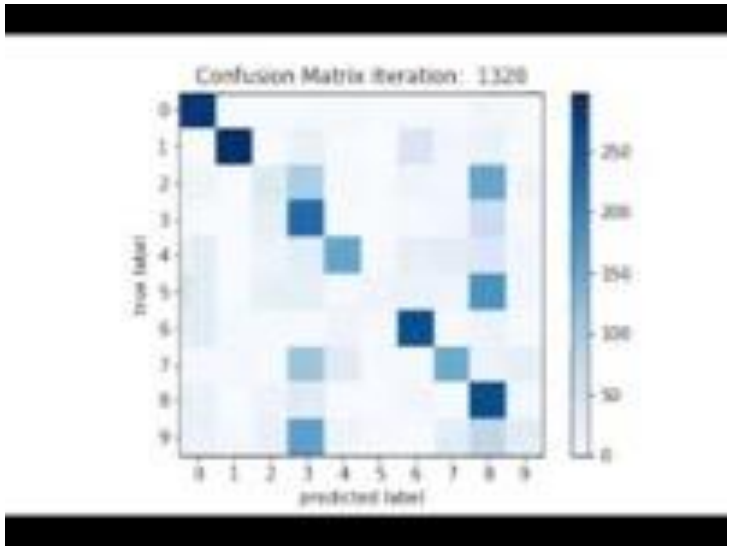
- If we take class Apple:
  - TP = 7
  - TN = (2+3+2+1) = 8
  - FP = (8+9) = 17
  - FN = (1+3) = 4
  - Precision = 7/(7+17) = 0.29
  - Recall = 7/(7+4) = 0.64
  - F1-score = 0.40

Predicted Class \ True Class	Apple	Orange	Mango
Apple	7	8	9
Orange	1	2	3
Mango	3	2	1

Performance Evaluation

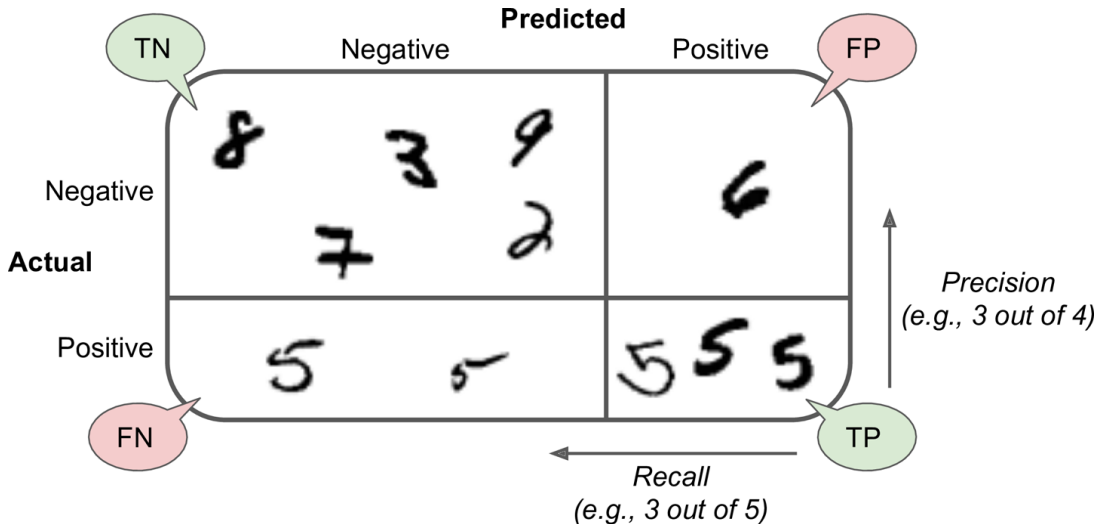
Confusion Matrix and Multi-class Classification on MNIST Dataset

Confusion Matrix for a Multi-Class MNIST Classifier



Video:
https://www.youtube.com/watch?v=H1viGe41YnA

Confusion Matrix for a Binary MNIST Classifier (5 vs rest)

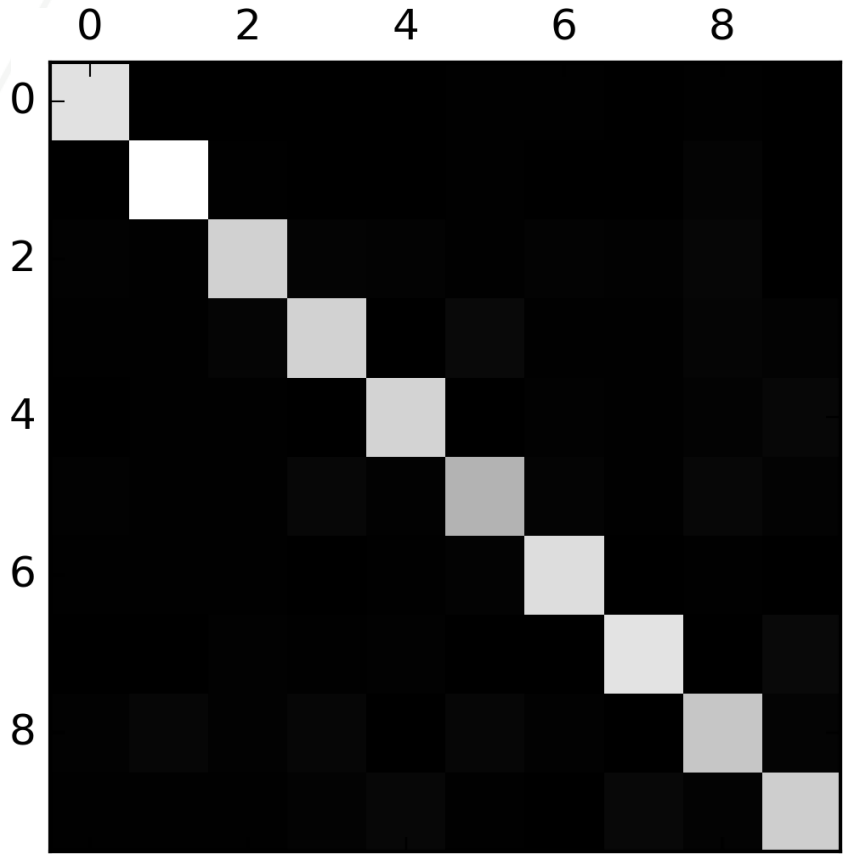


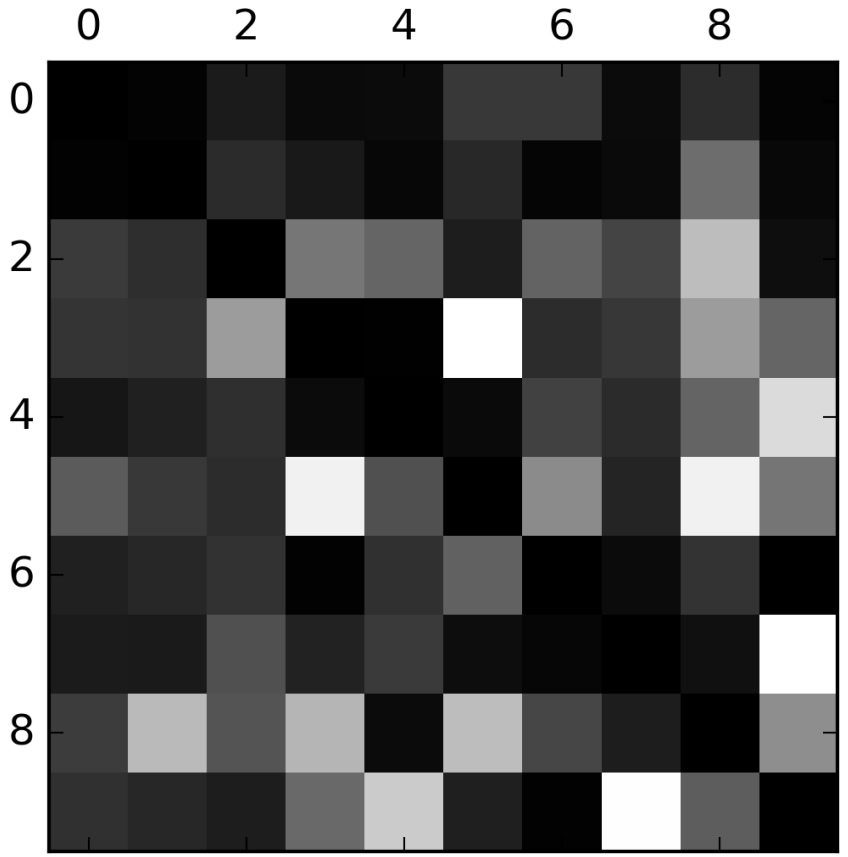
- Input image is for digit 5 ?

Performance Evaluation in Action

Error Analysis

Using Confusion Matrix. Which matrix reflects higher error?



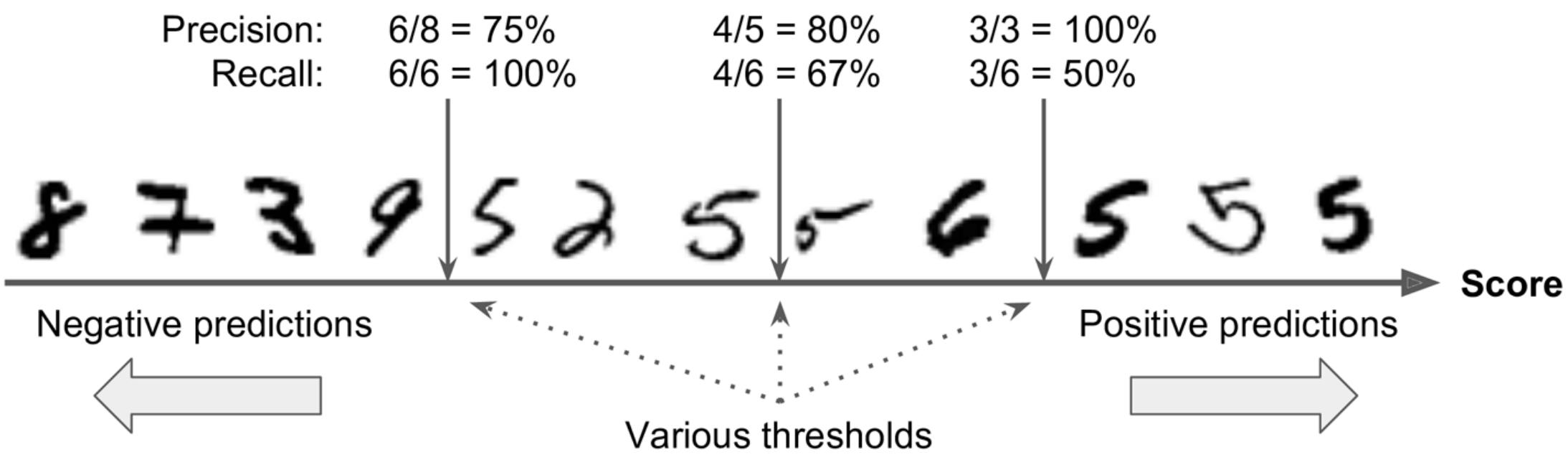


Rows represent actual classes, while columns represent predicted classes

Performance Evaluation in Action

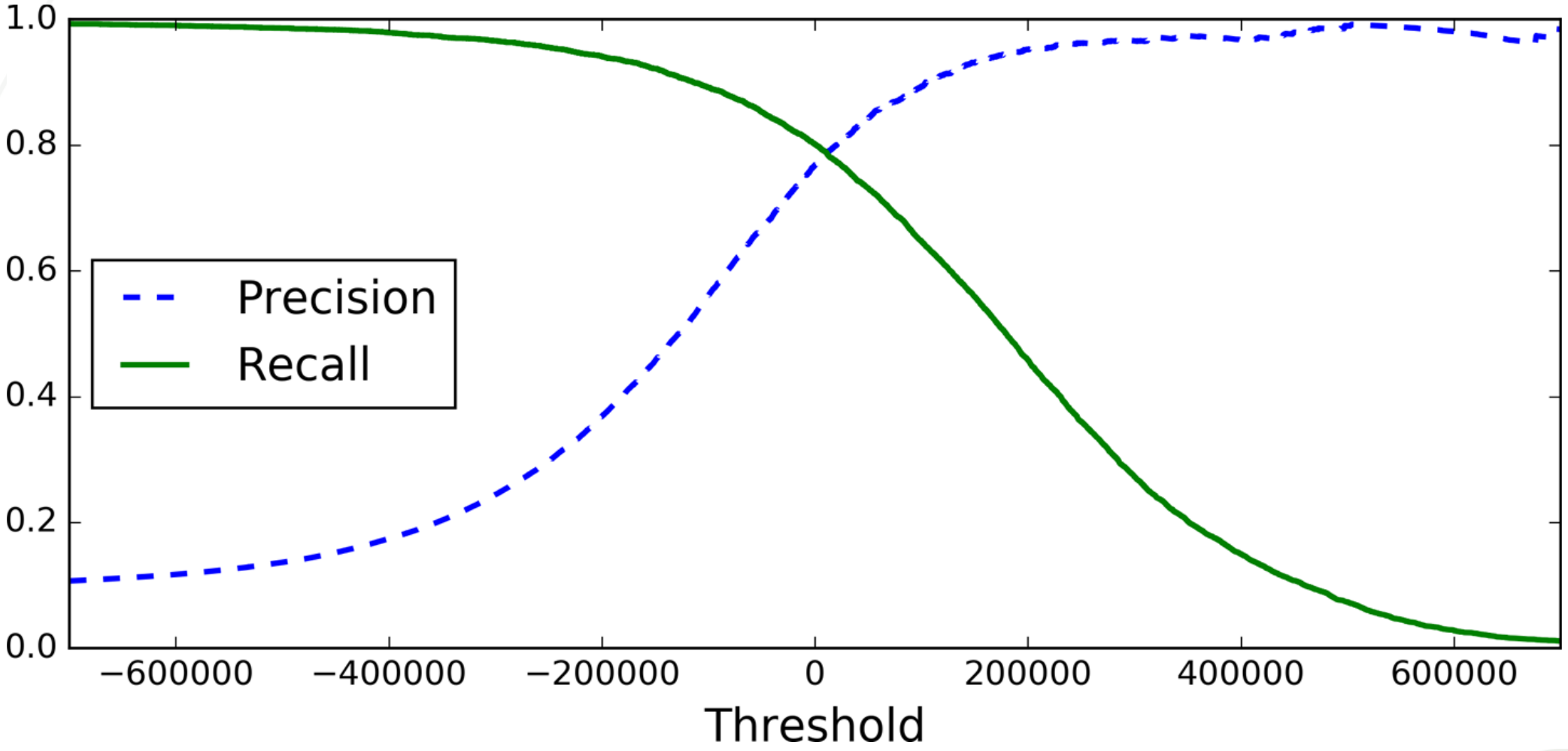
Precision-Recall Tradeoff

Goal: Evaluate the performance of a ‘5’ detector



Performance Evaluation in Action

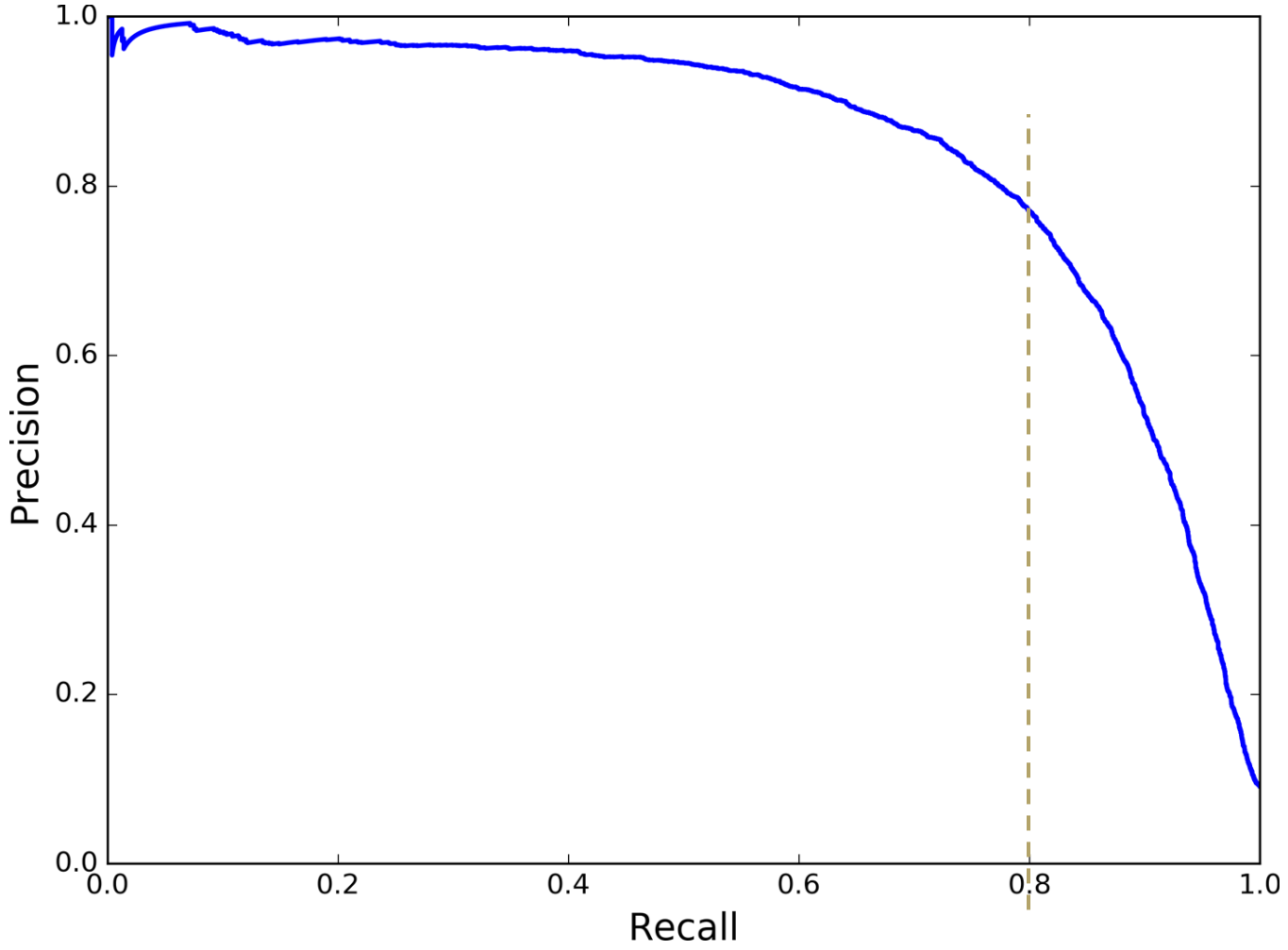
Precision-Recall Tradeoff



Performance Evaluation in Action

Precision-Recall Curve

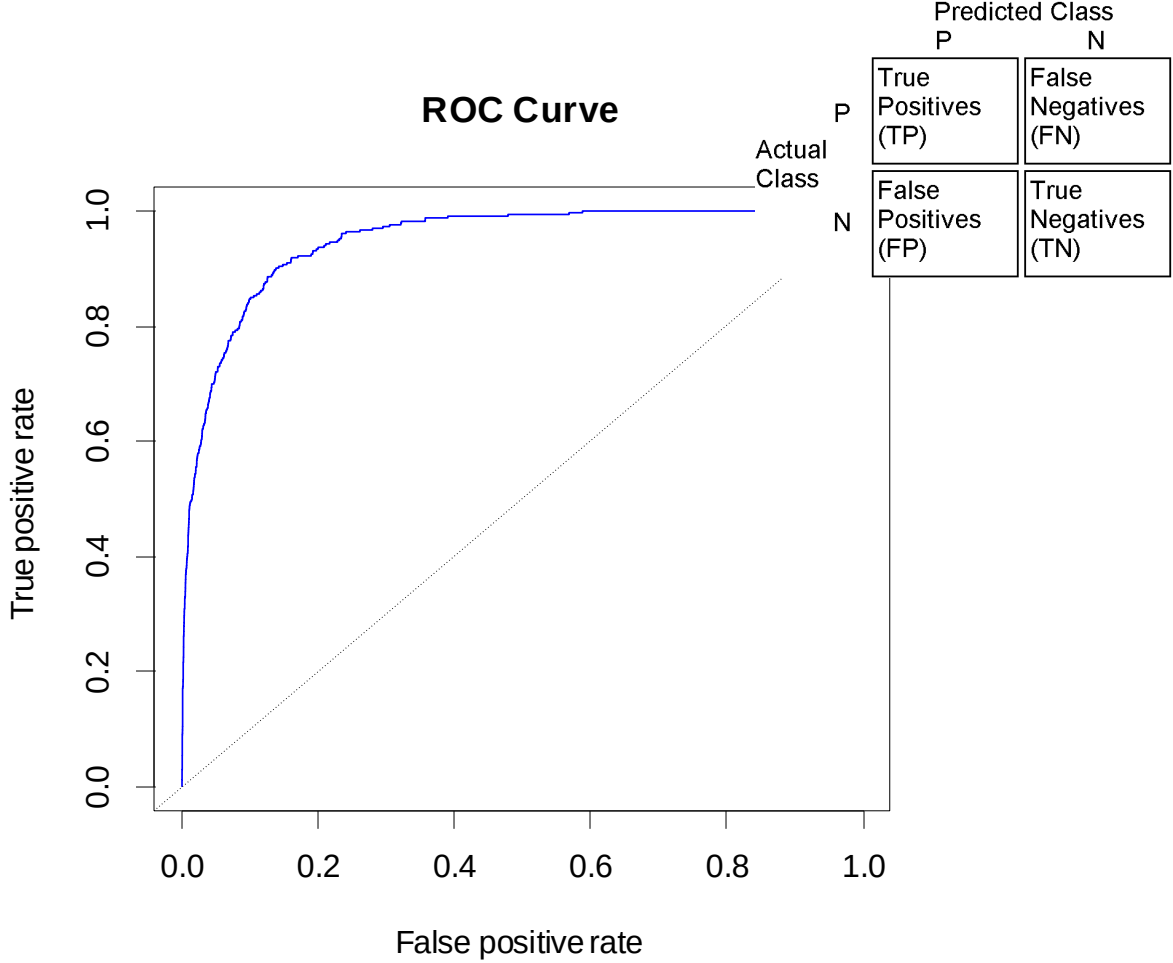
precision starts to fall sharply around 80% recall



Performance Evaluation in Action

ROC Curves

- Receiver Operating Characteristic (ROC), or ROC curve, is a graphical plot that illustrates the performance of classifiers as with varying thresholds.
- The ROC curve is generated by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings.
- TPR (sensitivity/recall): $\mathrm{TPR} = \frac{TP}{TP+FN}$
- FPR (1-specificity): $\mathrm{FPR} = \frac{FP}{FP+TN}$
- Specificity: $\frac{TN}{FP+TN}$
- Sensitivity and specificity are measures of the performance of a binary classification test that are widely used in medicine

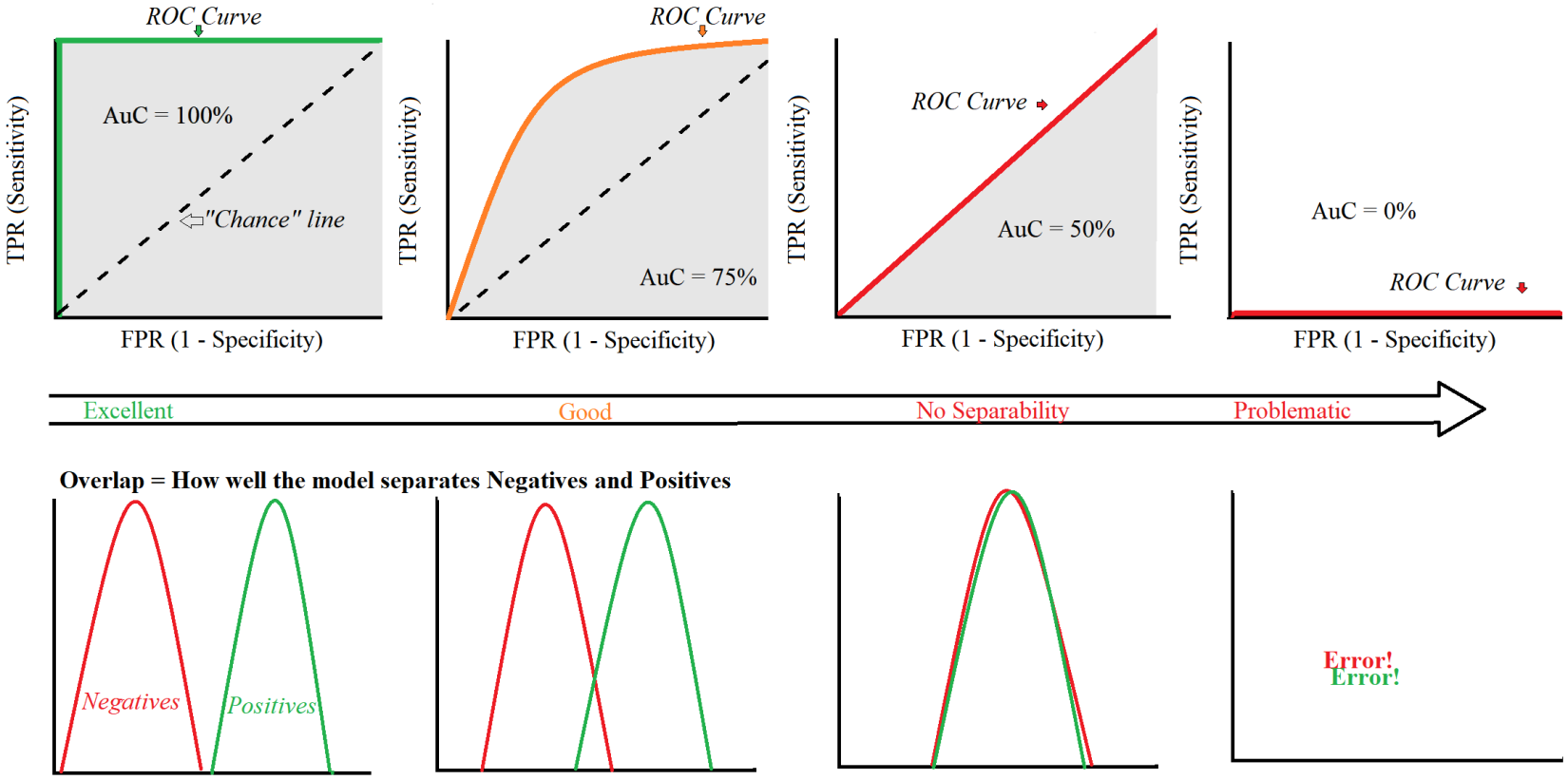


Sensitivity: the ability of a test to correctly identify patients with a disease.
Specificity: the ability of a test to correctly identify people without the disease.

Performance Evaluation in Action

Area Under the Curve

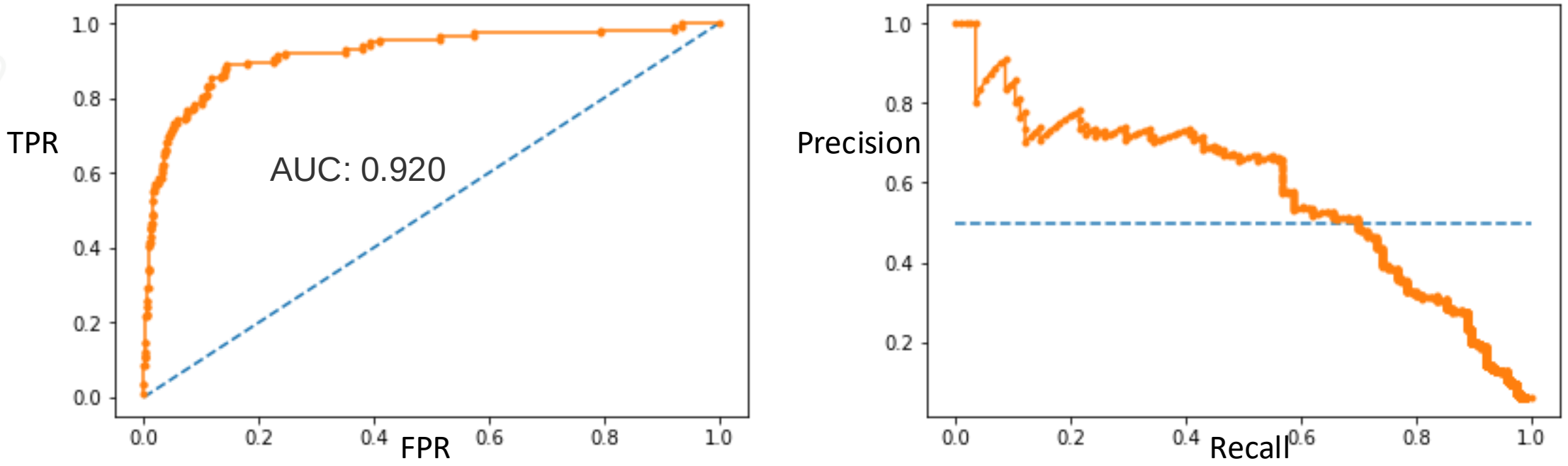
- The area under the curve (AUC) measures the quality of the classifier
- $AUC \in [0, 1]$. The higher the AUC, the better the classifier



Performance Evaluation in Action

Evaluation under Imbalanced Data

- The ROC curve is widely used to evaluate the performance of classifiers. However, it does not measure well for imbalanced data. And a common alternative is the precision-recall curve.



ROC Curve (left) vs. Precision-Recall Curve (right) with Imbalanced Data (negative class : positive class = 0.95 : 0.05)

Appendix A: Notations

- x_i: a single feature
- $\boldsymbol{x}_i$: feature vector (data sample)
- $\boldsymbol{X}$: matrix of feature vectors (dataset)
- N: number of data samples
- m: degree of polynomial
- P: number of features in a feature vector
- θ_i: a single model coefficient (parameter)
- $\boldsymbol{\theta}$: coefficient vector
- ε: error margin
- α: learning rate
- γ: bias factor
- Bold letter/symbol: vector
- Bold capital letters/symbol: matrix