MA289: Mathematics of AI

Lesson 8 Outline — 03 March 2026

United States Military Academy, West Point
Instructor: MAJ Patrick Kuiper

---

1 Shrinkage Lesson Objectives

- Understand common performance measures used in classification to include accuracy, precision, recall, and F_1 score.
- Use a receiver operating characteristic (ROC) curve to assess classifiers.
- Understand the difference between multiclass and multioutput classification.

2 Student Review

3 Shrinkage Methods Lecture

Shrinkage Methods (Ridge Regression and the Lasso)

4 Motivating Classification with Logistic Regression

Opening Discussion Questions

1. Why might accuracy be a misleading measure of performance in a classification problem?
2. What does it mean to say a model outputs a probability rather than a label, and why might that be useful?
3. How should we decide what "good" predictions mean when different types of errors have different consequences?
4. Give an example of a scenario where high precision is appropriate.
5. Give an example of a scenario where high recall is appropriate.
6. Why do we need an objective function to train a classifier, and how should it relate to probability?
7. How are the quantities we optimize during training connected to the metrics we use to evaluate a model afterward?

Basic Notation for Classification

Consider a supervised classification problem with input–output pairs

$$\{(x_i, y_i)\}_{i=1}^{n},$$

where $x_i \in \mathbb{R}^p$ denotes a vector of p features and $y_i \in \{0, 1\}$ denotes the class label for the ith observation.

A classifier is a function

$$f : \mathbb{R}^p \to \{0, 1\}$$

that assigns a class label to each input. In probabilistic classification, the model estimates the conditional probability

$$P(Y = 1 \mid X = x).$$

In logistic regression, this probability is modeled as

$$P(Y = 1 \mid X = x) = \sigma(z), \quad \text{where} \quad z = w^\top x + b,$$

with coefficient vector $w \in \mathbb{R}^p$, intercept $b \in \mathbb{R}$, and sigmoid function

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

Predicted class labels are obtained by applying a decision threshold $\tau \in (0,1)$:

$$\hat{y}_i = \begin{cases} 1 & \text{if } P(Y=1 \mid x_i) \geq \tau, \\ 0 & \text{otherwise.} \end{cases}$$

The collection of predicted labels $\{\hat{y}_i\}_{i=1}^n$ is compared to the true labels $\{y_i\}_{i=1}^n$ to compute evaluation quantities such as true positives, false positives, true negatives, and false negatives.

4.1 Why Accuracy Is Often Insufficient

Confusion Matrix

For a binary classification problem, the confusion matrix summarizes the relationship between true class labels and predicted class labels. It is given by

	Predicted 0	Predicted 1
Actual 0	TN	FP
Actual 1	FN	TP

where TP denotes true positives, TN denotes true negatives, FP denotes false positives, and FN denotes false negatives. The confusion matrix forms the basis for commonly used classification metrics such as accuracy, precision, recall, and the F1 score.

In binary classification, accuracy is defined as

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}.$$

Accuracy measures overall correctness, but it treats all errors equally. In practice, the two types of classification errors often have very different consequences.

- **False positives (FP):** predicting a positive outcome when none exists.
  - Medical screening: unnecessary follow-up tests.
  - Fraud detection: legitimate transactions incorrectly flagged.
  - Security systems: false alarms.
- **False negatives (FN):** failing to detect a true positive.
  - Medical diagnosis: missed disease.
  - Fraud detection: undetected fraud.
  - Safety or surveillance systems: missed threats.

Because accuracy cannot reflect these asymmetric costs, alternative metrics that focus on specific error types are often required.

4.2 Logistic Regression as a Probabilistic Classifier

Logistic regression models the conditional probability of the positive class as

$$P(Y=1 \mid X=x) = \sigma(z), \quad \text{where} \quad z = w^\top x + b.$$

Here,

- $w \in \mathbb{R}^p$ is the vector of coefficients,
- $x \in \mathbb{R}^p$ is the feature vector,
- $b \in \mathbb{R}$ is the intercept,
- z is the linear predictor (or logit).

The sigmoid function is defined as

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

Predicted class labels are obtained by thresholding the probability:

$$\hat{y} = \begin{cases} 1 & \text{if } \sigma(z) \geq \tau, \\ 0 & \text{otherwise.} \end{cases}$$

Logistic regression assumes a linear relationship between the predictors and the log-odds:

$$\log\left(\frac{P(Y = 1 \mid x)}{1 - P(Y = 1 \mid x)}\right) = z = w^\top x + b,$$

which implies a linear decision boundary defined by $z = 0$.

4.3 Precision, Recall, and Error Interpretation

From the confusion matrix, we define the following metrics.

Precision

$$\text{Precision} = \frac{TP}{TP + FP}.$$

Precision measures the proportion of predicted positives that are correct and is sensitive to false positives.

Recall

$$\text{Recall} = \frac{TP}{TP + FN}.$$

Recall measures the proportion of actual positives that are correctly identified and is sensitive to false negatives.

By adjusting the decision threshold τ, one can trade precision against recall, reflecting different application priorities.

Receiver Operating Characteristic (ROC) Curve

The Receiver Operating Characteristic (ROC) curve evaluates the performance of a binary classifier across all possible decision thresholds. It is a plot of the true positive rate (TPR) against the false positive rate (FPR), where

$$\text{TPR} = \frac{TP}{TP + FN}, \qquad \text{FPR} = \frac{FP}{FP + TN}.$$

Each point on the ROC curve corresponds to a specific classification threshold applied to the model's predicted probabilities. The area under the ROC curve (AUC) provides a threshold-independent measure of a model's ability to distinguish between the two classes.

4.4 The F1 Score

The F1 score is the harmonic mean of precision and recall:

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}.$$

The harmonic mean penalizes extreme imbalance between precision and recall, making the F1 score useful when both types of errors matter.

4.5 Maximum Likelihood Estimation

Logistic regression parameters are estimated using maximum likelihood. Assume

$$Y_i \mid X_i \sim \text{Bernoulli}(p_i), \quad p_i = \sigma(z_i), \quad z_i = w^\top x_i + b.$$

The likelihood function is

$$L(w, b) = \prod_{i=1}^{n} \sigma(z_i)^{y_i} \left(1 - \sigma(z_i)\right)^{1-y_i}.$$

Taking logs yields the log-likelihood

$$\ell(w, b) = \sum_{i=1}^{n} \left[y_i \log \sigma(z_i) + (1 - y_i) \log\left(1 - \sigma(z_i)\right)\right].$$

Maximizing this expression is equivalent to minimizing the negative log-likelihood, also known as the binary cross-entropy loss.

The gradient with respect to w is

$$\nabla_w \ell = \sum_{i=1}^{n} (y_i - \sigma(z_i)) x_i,$$

showing that parameter updates are driven by differences between observed labels and predicted probabilities.

4.6 Connecting Training and Evaluation

Maximum likelihood estimation focuses on learning well-calibrated probabilities through the linear predictor $z = w^\top x + b$. Precision, recall, and the F1 score evaluate how these probabilities are converted into decisions via thresholding.

Thus, the training objective and evaluation metrics serve distinct but complementary roles in classification.

5 Gradient Descent

Gradient Descent is an iterative optimization method used to minimize a cost function $J(\boldsymbol{\theta})$ with respect to model parameters $\boldsymbol{\theta}$.

5.1 General Optimization Framework

Let

$$J(\boldsymbol{\theta}) = \frac{1}{m} \sum_{i=1}^{m} L(f_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i)$$

where

- m = number of training samples
- $\mathbf{x}_i$ = feature vector
- y_i = target value
- $L(\cdot)$ = loss function

Gradient Descent updates parameters in the direction of steepest decrease of the cost function:

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}^{(t)})$$

where

- η = learning rate
- $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ = gradient vector

Convergence occurs when

$$\|\nabla J(\boldsymbol{\theta})\| < \epsilon$$

for tolerance ϵ.

5.2 Example: Linear Regression with MSE

For linear regression

$$\hat{y}_i = \boldsymbol{\theta}^T \mathbf{x}_i$$

the Mean Squared Error cost function is

$$J(\boldsymbol{\theta}) = \frac{1}{m} \sum_{i=1}^{m} (\boldsymbol{\theta}^T \mathbf{x}_i - y_i)^2$$

The gradient is

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{2}{m} X^T (X\boldsymbol{\theta} - \mathbf{y})$$

Thus the update rule becomes

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta \frac{2}{m} X^T (X\boldsymbol{\theta}^{(t)} - \mathbf{y})$$

5.3 Algorithm (Batch Gradient Descent)

1. Initialize $\boldsymbol{\theta}^{(0)}$ randomly.
2. Repeat until convergence:
   (a) Compute gradient
   $$g = \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$$
   (b) Update parameters
   $$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta g$$

Each iteration requires computation over the full dataset.

5.4 Learning Rate Effects

$$\eta \text{ too small} \Rightarrow \text{slow convergence}$$

$$\eta \text{ too large} \Rightarrow \text{divergence or oscillation}$$

Convergence rate for convex smooth functions is approximately

$$O\left(\frac{1}{\epsilon}\right)$$

iterations to reach accuracy ϵ.

5.5 Feature Scaling

If features have different scales, the Hessian of $J(\boldsymbol{\theta})$ becomes ill-conditioned, producing elongated level sets and slow convergence. Standardization improves conditioning:

$$x_j^{(scaled)} = \frac{x_j - \mu_j}{\sigma_j}$$

5.6 Stochastic Gradient Descent (SGD)

Instead of computing gradients using all m observations, SGD updates parameters using a single randomly selected sample:

$$\nabla J_i(\boldsymbol{\theta}) = 2\,\mathbf{x}_i(\mathbf{x}_i^T\boldsymbol{\theta} - y_i)$$

Update rule:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta_t \nabla J_i(\boldsymbol{\theta})$$

Learning rate typically follows a schedule

$$\eta_t = \frac{t_0}{t + t_1}$$

SGD properties:

- Fast for large datasets
- Supports streaming / out-of-core training
- Converges to neighborhood of optimum

5.7 Mini-batch Gradient Descent

Mini-batch GD uses a subset B of size $|B| = b$:

$$\nabla J_B(\boldsymbol{\theta}) = \frac{2}{b}\sum_{i\in B}\mathbf{x}_i(\mathbf{x}_i^T\boldsymbol{\theta} - y_i)$$

Update rule

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla J_B(\boldsymbol{\theta})$$

Properties:

- Computationally efficient with matrix operations
- Reduced gradient variance vs. SGD
- Widely used in modern deep learning

5.8 Algorithm Comparison

Let m be samples and n features.

Method	Data per Update	Speed (Large m)	Noise
Batch GD	m	Slow	None
SGD	1	Fast	High
Mini-batch GD	b	Fast	Moderate

All methods follow the same optimization principle:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \nabla J(\boldsymbol{\theta}_t)$$

but differ in how the gradient estimate is computed.