Lecture: Classification Thresholds and the Confusion Matrix

1. Introduction: From Probabilities to Decisions

Many classification models—logistic regression, neural networks with sigmoid outputs, gradient boosting classifiers—do not directly output class labels. Instead, they output **scores or probabilities** between 0 and 1.

For example, in a spam email detection system:

- A score of **0.75** means the model estimates a **75% probability** that the email is spam.
- A score of **0.10** means the model estimates a **10% probability** that the email is spam.

However, real-world systems must make **binary decisions**:

- Spam or not spam
- Fraud or legitimate
- Malicious or safe

To convert probabilities into class labels, we introduce a **classification threshold**.

---

2. Classification Thresholds

Definition

A **classification threshold** is a value $t \in [0,1]$t \in [0,1]t∈[0,1] such that:

- If `predicted probability ≥ t` → predict **positive class**
- If `predicted probability < t` → predict **negative class**

In the spam example:

- Positive class = spam
- Negative class = not spam

Example

Assume two emails:

- Email A: probability = 0.99
- Email B: probability = 0.51

Threshold	Email A	Email B
0.50	Spam	Spam
0.95	Spam	Not spam

As the threshold increases, the classifier becomes **more conservative** about predicting the positive class.

Edge Case: Probability Equals Threshold

If the predicted probability is exactly equal to the threshold:

- Behavior depends on the implementation.
- For example, **Keras predicts the negative class** when `probability == threshold`.

---

3. Why 0.5 Is Often Not the Right Threshold

Although 0.5 is intuitive, it is often **suboptimal** due to:

3.1 Unequal Error Costs

- False positives and false negatives may not have equal consequences.
- In email filtering:
    - **False positive**: legitimate email sent to spam (high cost)

- **False negative**: spam reaches inbox (lower cost)

3.2 Class Imbalance

If only **0.01% of emails are spam**, predicting spam whenever probability ≥ 0.5 may produce too many errors.

In such cases, thresholds should be chosen based on:

- Business impact
- Risk tolerance
- Regulatory constraints
- User experience

---

4. The Confusion Matrix

To evaluate classification performance, we compare **model predictions** to **ground truth** using a **confusion matrix**.

Structure of the Confusion Matrix

	Actual Positive	**Actual Negative**
Predicted Positive	True Positive (TP)	False Positive (FP)
Predicted Negative	False Negative (FN)	True Negative (TN)

Definitions (Spam Example)

- **True Positive (TP)**
  Spam email correctly classified as spam.
- **False Positive (FP)**
  Legitimate email incorrectly classified as spam.
- **False Negative (FN)**
  Spam email incorrectly classified as legitimate.

- **True Negative (TN)**
  Legitimate email correctly classified as legitimate.

Interpretation

- Rows summarize **model predictions**
- Columns summarize **actual outcomes**
- Class imbalance occurs when:
  $TP+FN \ll FP+TN$or vice versaTP + FN \ll FP + TN \quad \text{or vice versa}$TP+FN \ll FP+TN$or vice versa

---

5. Effect of Threshold on Classification Outcomes

Changing the threshold directly affects the entries of the confusion matrix.

5.1 Increasing the Threshold

When the threshold increases:

- The model predicts **fewer positives**
- It becomes harder to label an instance as positive

Effects:

Metric	Effect
True Positives (TP)	Decrease
False Positives (FP)	Decrease
False Negatives (FN)	Increase
True Negatives (TN)	Increase

At an extremely high threshold (e.g., 0.9999):

- Almost everything is classified as negative

- Spam is rarely caught, but legitimate emails are rarely misclassified

---

5.2 Decreasing the Threshold

When the threshold decreases:

- The model predicts **more positives**
- It becomes easier to label an instance as positive

Effects:

Metric	Effect
True Positives (TP)	Increase
False Positives (FP)	Increase
False Negatives (FN)	Decrease
True Negatives (TN)	Decrease

At an extremely low threshold (e.g., 0.01):

- Almost everything is classified as spam
- Spam is caught, but many legitimate emails are misclassified

---

6. Understanding Errors Through Examples

Example 1: Malware Detection

A model classifies websites as:

- **1** = malware/phishing
- **0** = legitimate

If a legitimate website is flagged as malware:

- Actual class = 0
- Predicted class = 1

This is a **false positive**.

Example 2: Threshold Experiment

Adjusting the threshold from 0.1 to 0.9:

- **True positives decrease**
- **False positives decrease**
- **False negatives increase**
- **True negatives increase**

This reflects the trade-off between **sensitivity** and **conservativeness**.

---

7. Threshold Selection as a Design Decision

Choosing a threshold is not a purely technical step—it is a **policy decision**.

Thresholds should be selected using:

- Confusion matrix analysis
- Cost-benefit analysis
- Precision–recall trade-offs
- Domain expertise

Common strategies include:

- Maximizing F1 score
- Fixing an acceptable false positive rate
- Optimizing expected cost

---

8. Summary

Key takeaways:

1. Many classifiers output probabilities, not labels.
2. A classification threshold converts probabilities into decisions.
3. The confusion matrix provides a structured view of prediction outcomes.
4. Increasing the threshold reduces both true and false positives.
5. Decreasing the threshold reduces false negatives but increases false positives.
6. Threshold choice must reflect real-world costs and class imbalance.

In practice, **there is no universally optimal threshold**—only a threshold that is optimal for a given problem, dataset, and business objective.

Lecture: Classification Metrics — Accuracy, Recall, Precision, and Related Measures

1. Introduction

Binary classification models are widely used in real-world systems such as spam detection, fraud detection, medical diagnosis, and cybersecurity. Evaluating these models requires more than simply checking whether predictions are correct or incorrect. Different types of errors have different consequences, and datasets are often imbalanced.

To evaluate classifiers properly, we rely on **metrics derived from the confusion matrix**, computed at a **fixed classification threshold**. Because these metrics depend on the threshold, practitioners often tune the threshold to optimize a metric aligned with business or scientific objectives.

---

2. The Confusion Matrix as the Foundation

All classification metrics in this lecture are derived from four quantities:

- **True Positives (TP)**: correctly predicted positives
- **False Positives (FP)**: negatives incorrectly predicted as positives
- **False Negatives (FN)**: positives incorrectly predicted as negatives
- **True Negatives (TN)**: correctly predicted negatives

These values summarize model behavior at a particular threshold.

---

3. Accuracy

Definition

Accuracy is the proportion of all predictions that are correct:

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

Interpretation

Accuracy measures how often the model is correct overall, regardless of class.

In spam detection, accuracy is the fraction of all emails—spam and non-spam—that are correctly classified.

Strengths

- Easy to understand
- Useful as a **coarse indicator** of training progress
- Works reasonably well when:
    - Classes are balanced
    - False positives and false negatives have similar costs

Limitations

Accuracy is **misleading for imbalanced datasets**.

Example:

- 99% legitimate emails, 1% spam
- A model that predicts “not spam” every time achieves **99% accuracy**
- Yet it detects **zero spam emails**

For this reason, accuracy should not be used alone in most real-world applications.

---

4. Recall (True Positive Rate)

Definition

Recall, also known as **true positive rate (TPR)**, measures the proportion of actual positives correctly identified:

Recall=TPTP+FN\text{Recall} = \frac{TP}{TP + FN}Recall=TP+FNTP

Interpretation

Recall answers the question:

> “Of all the actual positive cases, how many did the model detect?”

In spam detection, recall is the fraction of spam emails that were correctly classified as spam.

This is why recall is sometimes called **probability of detection**.

Strengths

- Focuses on minimizing **false negatives**
- Especially important when missing a positive case is costly

When Recall Matters Most

- Medical diagnosis (missing a disease)
- Fraud detection
- Malware or intrusion detection
- Early-warning systems

In imbalanced datasets with very few positives, recall is often far more informative than accuracy.

---

5. False Positive Rate (FPR)

Definition

The **false positive rate** measures the proportion of actual negatives incorrectly classified as positives:

FPR=FPFP+TN\text{FPR} = \frac{FP}{FP + TN}FPR=FP+TNFP

Interpretation

FPR answers the question:

> “Of all the negative cases, how many did the model incorrectly flag?”

In spam filtering, FPR is the fraction of legitimate emails sent to the spam folder.

Strengths

- Useful when false alarms are costly or disruptive
- Important in systems where unnecessary alerts create operational burden

Limitations

When the number of actual negatives is very small, FPR can be volatile. A single misclassification can cause a large jump in the metric.

---

6. Precision

Definition

Precision measures how many predicted positives are actually positive:

Precision=TPTP+FP\text{Precision} = \frac{TP}{TP + FP}Precision=TP+FPTP

Interpretation

Precision answers the question:

> “When the model predicts positive, how often is it correct?”

In spam detection, precision measures how many emails labeled as spam truly are spam.

Strengths

- Focuses on minimizing **false positives**
- Useful when positive predictions trigger costly actions

Limitations

In extremely imbalanced datasets with very few actual positives, precision can become unstable or uninformative.

---

7. Precision–Recall Trade-off

Precision and recall often have an **inverse relationship** driven by the classification threshold:

- Increasing the threshold:
  - Precision increases
  - Recall decreases
- Decreasing the threshold:
  - Recall increases
  - Precision decreases

This trade-off reflects the balance between **missing positives** and **raising false alarms**.

---

8. NaN Values in Metrics

Metrics can produce **NaN (Not a Number)** values when division by zero occurs.

Example:

- If a model never predicts positive:

- TP = 0, FP = 0
- Precision = 0 / 0 → NaN

NaN does not necessarily indicate good or bad performance. It must be interpreted in context.

---

9. Choosing the Right Metric

Metric selection depends on **error costs and risk tolerance**.

Metric	Use When
Accuracy	Balanced datasets; rough progress tracking
Recall (TPR)	False negatives are very costly
False Positive Rate	False positives are very costly
Precision	Positive predictions must be highly reliable

In practice, models are often evaluated using **multiple metrics simultaneously**.

---

10. F1 Score (Advanced)

Definition

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

F1=2⋅Precision⋅RecallPrecision+RecallF1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}F1=2⋅Precision+RecallPrecision⋅Recall

Interpretation

- Penalizes extreme imbalance between precision and recall
- Dominated by the weaker of the two metrics
- Preferred over accuracy for imbalanced datasets

A perfect model has:

- Precision = 1.0
- Recall = 1.0
- F1 = 1.0

---

11. Applied Example: Invasive Species Detection

A model monitors insect traps for a dangerous invasive species.

- **False positives**: easy to review and dismiss
- **False negatives**: risk of infestation and ecological damage

In this scenario:

- False negatives are far more costly
- The model should be optimized for **recall**

---

12. Summary

Key points:

1. All classification metrics derive from the confusion matrix.
2. Accuracy alone is often misleading.
3. Recall measures detection ability.
4. Precision measures prediction reliability.
5. FPR measures false alarm rates.
6. Threshold tuning controls trade-offs.
7. Metric choice must align with real-world costs.

8. F1 balances precision and recall in imbalanced datasets.

Effective evaluation is not about maximizing a single number, but about **choosing the right metric for the right problem**.

Lecture: Classification Evaluation — Metrics, ROC/AUC, and Prediction Bias

1. Introduction: Evaluating Classification Models Holistically

Evaluating a classification model involves more than computing a single metric. A robust evaluation framework must address:

- Performance at a specific threshold (accuracy, recall, precision)
- Performance across all thresholds (ROC and AUC)
- Whether the model's **overall predictions are calibrated and reasonable** (prediction bias)

A model can score well on standard metrics and still be fundamentally flawed. Prediction bias serves as a **quick, global sanity check** that complements other evaluation tools.

---

2. Foundations: The Confusion Matrix

All threshold-based metrics originate from the **confusion matrix**, which compares predictions to ground truth.

	Actual Positive	**Actual Negative**
Predicted Positive	True Positive (TP)	False Positive (FP)
Predicted Negative	False Negative (FN)	True Negative (TN)

This matrix changes as the classification threshold changes and forms the basis for most evaluation metrics.

---

3. Threshold-Based Metrics

3.1 Accuracy

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

Accuracy measures the proportion of correct predictions overall.

- Useful for balanced datasets
- Misleading for imbalanced datasets
- Should never be used alone in real-world applications

3.2 Recall (True Positive Rate)

Recall=TPTP+FN\text{Recall} = \frac{TP}{TP + FN}Recall=TP+FNTP

Recall measures the model's ability to detect actual positives.

- Prioritize recall when false negatives are costly
- Especially important in medical, security, and early-warning systems

---

3.3 Precision

Precision=TPTP+FP\text{Precision} = \frac{TP}{TP + FP}Precision=TP+FPTP

Precision measures the reliability of positive predictions.

- Prioritize precision when false positives are costly
- Common in spam filtering and automated decision systems

3.4 False Positive Rate (FPR)

FPR=FPFP+TN\text{FPR} = \frac{FP}{FP + TN}FPR=FP+TNFP

FPR measures how often negatives are incorrectly flagged.

- Useful when false alarms are disruptive or expensive

3.5 Trade-offs and Threshold Selection

- Increasing the threshold:
    - Precision increases
    - Recall decreases
- Decreasing the threshold:
    - Recall increases
    - Precision decreases

Threshold selection is a **design decision**, not a purely mathematical one.

---

4. ROC Curves and AUC: Threshold-Independent Evaluation

4.1 ROC Curve

The **Receiver Operating Characteristic (ROC) curve** plots:

- **True Positive Rate (TPR)** on the y-axis
- **False Positive Rate (FPR)** on the x-axis

Each point corresponds to a different classification threshold.

Key reference points:

- (0,1): perfect classification
- Diagonal from (0,0) to (1,1): random guessing

4.2 Area Under the Curve (AUC)

AUC summarizes the ROC curve as a single number.

Interpretation:

- AUC = 1.0 → perfect ranking
- AUC = 0.5 → random guessing
- AUC < 0.5 → worse than chance

Probabilistic meaning:

> AUC is the probability that the model ranks a randomly chosen positive example higher than a randomly chosen negative example.

4.3 Using ROC/AUC in Practice

- Use AUC to compare models
- Use ROC curves to select thresholds
- Best thresholds lie closest to the point (0,1)
- ROC/AUC are most reliable when datasets are roughly balanced

---

5. Class Imbalance and Metric Choice

In imbalanced datasets:

- Accuracy becomes unreliable
- ROC curves can appear overly optimistic
- Recall, precision, F1 score, and prediction bias become more important

---

6. Prediction Bias: A Global Sanity Check

6.1 Definition

Prediction bias is defined as:

Prediction Bias=Mean(predictions)−Mean(ground truth labels)\text{Prediction Bias} = \text{Mean(predictions)} - \text{Mean(ground truth labels)}Prediction Bias=Mean(predictions)−Mean(ground truth labels)

It measures whether the model's **average prediction matches the observed average outcome**.

6.2 Intuition

If 5% of emails in the dataset are spam:

- The average ground-truth label = 0.05
- A well-trained model should, on average, predict spam with probability ≈ 0.05

If instead the model predicts spam 50% of the time on average, something is wrong—even if traditional metrics appear acceptable.

6.3 Example

Spam dataset:

- 1,000 emails
- 50 spam emails → mean label = 0.05

Model outputs probabilities:

- Mean predicted probability = 0.50

Prediction bias:

0.50−0.05=0.450.50 - 0.05 = 0.450.50−0.05=0.45

This is a **large prediction bias**, indicating a serious issue.

6.4 What Zero Prediction Bias Means

A model with **zero prediction bias**:

- Predicts the correct overall rate of positives
- Is properly aligned with the base rate of the data

Important note:

- Zero prediction bias does **not** guarantee good classification performance
- It is a necessary but not sufficient condition for a healthy model

---

7. Common Causes of Prediction Bias

Significant prediction bias often indicates systemic problems, such as:

7.1 Biased or Noisy Data

- Non-representative training samples
- Labeling errors
- Distribution shift between training and inference data

7.2 Excessive Regularization

- Model is oversimplified
- Important signals are suppressed
- Predictions collapse toward the mean

7.3 Bugs in the Training Pipeline

- Incorrect label encoding
- Data leakage
- Feature misalignment between training and serving

7.4 Insufficient or Weak Features

- Features lack predictive signal
- Model compensates by predicting near the global mean

---

8. When to Use Prediction Bias

Prediction bias is especially useful:

- As an **early diagnostic check**
- Before deep metric analysis
- When deploying a model to new data
- When monitoring models in production

It is fast to compute and can immediately flag major issues.

---

9. How Prediction Bias Complements Other Metrics

Tool	Purpose
Accuracy	Overall correctness (limited)
Precision / Recall	Error-type control
ROC / AUC	Ranking quality across thresholds
Prediction Bias	Global alignment with reality

A robust evaluation uses **all of these together**, not in isolation.

---

10. Summary

Key takeaways:

1. Classification metrics depend on the confusion matrix.

2. Threshold selection determines precision–recall trade-offs.
3. ROC curves and AUC evaluate performance across all thresholds.
4. AUC measures ranking quality, not calibration.
5. Prediction bias checks whether average predictions match reality.
6. Large prediction bias signals data, model, or pipeline problems.
7. Zero prediction bias does not guarantee good performance—but large bias guarantees a problem.
8. Effective model evaluation is multidimensional.

Lecture: Multi-Class Classification

1. Introduction

So far, classification has been discussed primarily in the **binary** setting, where each example belongs to exactly one of two classes. Many real-world problems, however, require choosing among **more than two possible classes**.

Multi-class classification extends binary classification to handle problems where each example belongs to **one and only one** class out of several possible classes.

Examples include:

- Handwritten digit recognition (digits 0–9)
- Document topic classification
- Image recognition with multiple object categories
- Language identification

---

2. Formal Definition

In **multi-class classification**:

- There are **K > 2** possible classes

- Each example is assigned to **exactly one class**
- Class membership is **mutually exclusive**

Formally, for a label set:

Y={1,2,…,K}\mathcal{Y} = \{1, 2, \dots, K\}Y={1,2,…,K}

each example xxx has exactly one true label y∈Yy \in \mathcal{Y}y∈Y.

---

3. Multi-Class vs. Multi-Label Classification

It is important to distinguish multi-class classification from a related but different problem.

Multi-Class Classification

- One and only one class per example
- Classes are mutually exclusive
- Example: handwritten digit = exactly one digit

Multi-Label Classification

- An example can belong to **multiple classes simultaneously**
- Example: an image tagged with “dog,” “outdoor,” and “person”

This lecture focuses on **multi-class classification**.

---

4. Extending Binary Classification to Multi-Class Problems

Most machine learning algorithms are fundamentally designed for **binary classification**. Multi-class classification is often implemented by **reducing** the problem into multiple binary classification tasks.

---

5. One-vs-Rest (OvR) / One-vs-All Strategy

Concept

For each class:

- Treat that class as the **positive class**
- Treat all other classes combined as the **negative class**
- Train one binary classifier per class

If there are KKK classes, this results in KKK binary classifiers.

Example: Three Classes (A, B, C)

We build three classifiers:

1. Classifier 1: A vs (B + C)
2. Classifier 2: B vs (A + C)
3. Classifier 3: C vs (A + B)

At prediction time:

- Each classifier outputs a score or probability
- The class with the **highest score** is selected

Advantages

- Simple and widely supported
- Scales linearly with the number of classes
- Works with most binary classifiers

Disadvantages

- Class imbalance is common (one class vs many)
- Probabilities may not be well calibrated across classes

---

6. Hierarchical Binary Classification

Multi-class classification can also be performed by arranging binary classifiers in a **decision hierarchy**.

Example: Three Classes (A, B, C)

1. First classifier: (A + B) vs C
2. Second classifier: A vs B (applied only if step 1 predicts A+B)

This approach reduces the number of comparisons but introduces:

- Error propagation
- Sensitivity to early mistakes

---

7. Native Multi-Class Models

Some models support multi-class classification **natively**, without reduction to binary tasks.

Examples:

- Softmax regression (multinomial logistic regression)
- Neural networks with softmax output layers
- Decision trees and random forests
- Gradient boosting with multi-class objectives

These models typically:

- Output a probability distribution over classes

- Predict the class with the highest probability

---

8. Output Interpretation in Multi-Class Models

For a KKK-class problem, the model outputs:

$P(y=k \mid x), k=1,\dots,K$

Constraints:

- All probabilities are between 0 and 1
- The probabilities sum to 1

The predicted class is:

$\arg\max_k P(y=k \mid x)$

---

9. Evaluation Metrics for Multi-Class Classification

Many binary classification metrics extend naturally to the multi-class setting.

Common Approaches

- **Overall accuracy**
  Fraction of correctly classified examples
- **Per-class precision, recall, F1**
  Treat each class as positive in turn
- **Macro-averaging**
  Average metric equally across classes
- **Micro-averaging**
  Aggregate all predictions and compute a single metric

Choice of averaging depends on:

- Class imbalance
- Importance of minority classes

---

10. Confusion Matrix in Multi-Class Classification

The confusion matrix generalizes to a **K × K matrix**:

- Rows: predicted classes
- Columns: true classes
- Diagonal: correct predictions
- Off-diagonal entries: misclassifications between specific classes

This matrix provides insight into **which classes are being confused** with each other.

---

11. Practical Example: Handwritten Digit Recognition

Problem:

- Input: image of a handwritten digit
- Output: one of {0, 1, 2, ..., 9}

Characteristics:

- Exactly one correct label per image
- Naturally suited to multi-class classification
- Often implemented using softmax-based models or OvR strategies

---

12. Key Challenges in Multi-Class Classification

- Increased model complexity
- Class imbalance across multiple classes
- Error propagation in hierarchical approaches

- Evaluation becomes more nuanced

Despite these challenges, most concepts from binary classification—thresholds, metrics, calibration, and bias—still apply, either directly or through extensions.

13. Summary

Key takeaways:

1. Multi-class classification handles problems with more than two mutually exclusive classes.
2. Each example belongs to exactly one class.
3. Multi-class problems can be reduced to multiple binary classification tasks.
4. One-vs-Rest is the most common reduction strategy.
5. Some models natively support multi-class outputs.
6. Multi-class classification is distinct from multi-label classification.
7. Evaluation extends binary metrics using per-class and averaged measures.
8. Confusion matrices generalize naturally to multiple classes.