Classification

From Binary to Multioutput Systems

Gauranga Kumar Baishya

August 29, 2025

Outline

1. Introduction to Classification & MNIST
2. Training a Binary Classifier
3. Performance Measures
4. Multiclass Classification
5. Error Analysis
6. Advanced Classification Tasks

The MNIST Dataset: "Hello World" of ML

What is MNIST?

A dataset of 70,000 small, grayscale images of handwritten digits (0-9). It's a benchmark for testing new classification algorithms.

Dataset Structure

- 70,000 instances (images).
- 784 features per instance.
- Each image is 28x28 pixels.
- Each feature represents one pixel's intensity (0-255).

Figure: A set of few digits from the MNIST dataset.

Important First Step

Creating a "5-Detector"

Simplifying the Problem

To start, we'll build a **binary classifier** that can only distinguish between two classes: "5" and "not-5".

Target Vector Creation

We create new target labels that are boolean: True for all 5s, False for all other digits:

$$y_{train5} = (y_{train} == 5)$$
$$y_{test5} = (y_{test} == 5)$$

Training an SGD Classifier

A good starting point is the **Stochastic Gradient Descent (SGD)** classifier. It's efficient and handles large datasets well.

The Problem with Accuracy

Initial Accuracy Score

Using 3-fold cross-validation, the `SGDClassifier` achieves over 93% accuracy.

```
array([0.96355, 0.93795, 0.95615])
```

This seems great, but is it?

The Pitfall of Skewed Datasets

Let's consider a classifier that *always* predicts "not-5".

- Only about 10% of the images are 5s.
- So, this "dumb" classifier will be correct about 90% of the time!
- This shows that **accuracy is not a good performance measure for classifiers, especially on skewed datasets**.

The Confusion Matrix

A Better Way to Evaluate

The confusion matrix provides a much better view of a classifier's performance by showing the number of times instances of class A are classified as class B.

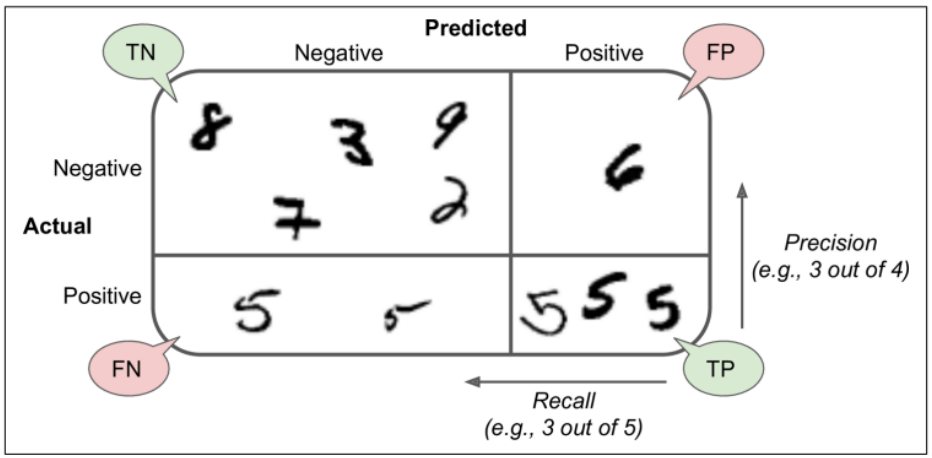
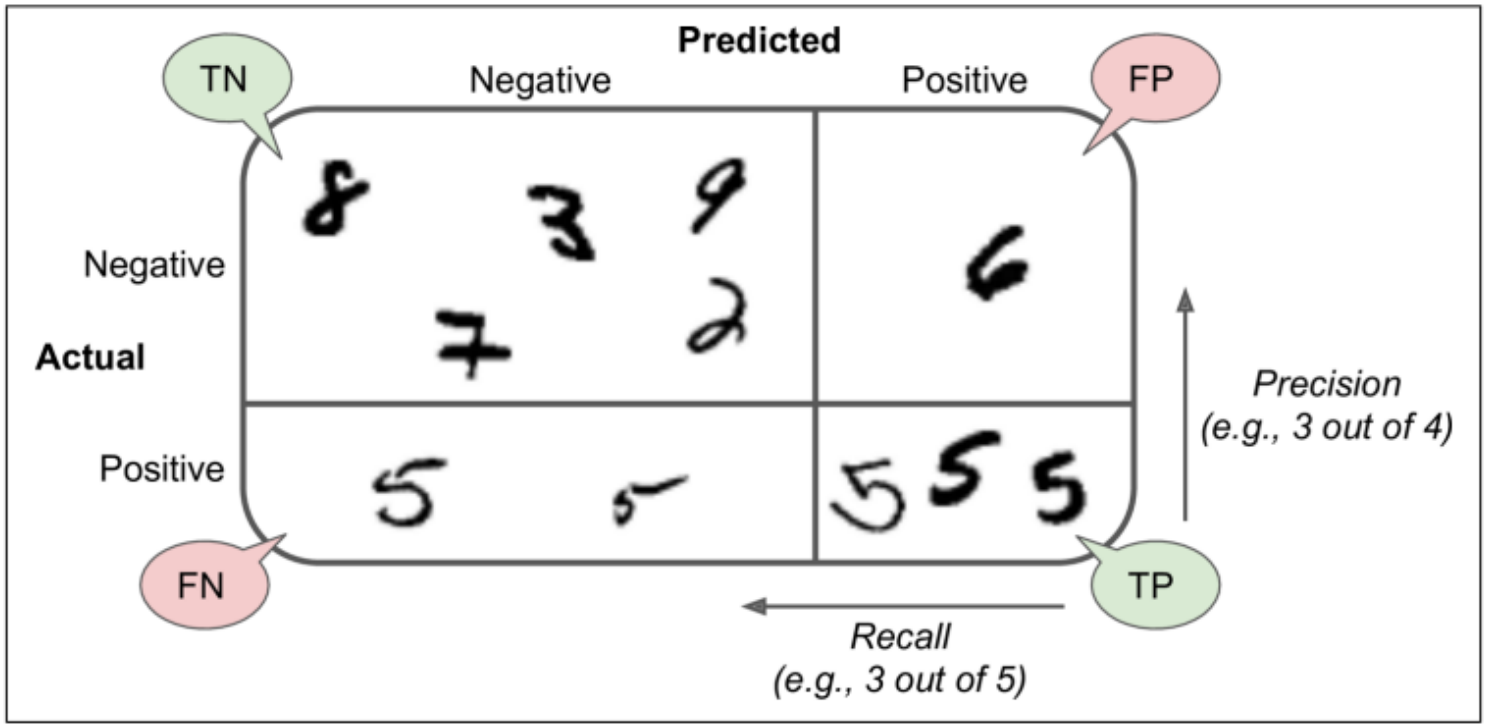


Figure: Structure of a confusion matrix.

Terminology:

- **True Negatives (TN)**: Correctly classified as not-5.
- **False Positives (FP)**: Incorrectly classified as 5.
- **False Negatives (FN)**: Incorrectly classified as not-5.
- **True Positives (TP)**: Correctly classified as 5.

Our 5-Detector's Matrix



$$\begin{bmatrix} 53057 & 1522 \\ 1325 & 4096 \end{bmatrix}$$

- **1522** non-5s were wrongly classified as 5s (FP).
- **1325** 5s were wrongly classified as not-5s (FN).

Precision, Recall, and F1 Score

Precision: Accuracy of Positive Predictions

What proportion of positive identifications was actually correct?

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

For our model, precision is $4096/(4096 + 1522) \approx 72.9\%$.

Recall (Sensitivity): True Positive Rate

What proportion of actual positives was identified correctly?

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

For our model, recall is $4096/(4096 + 1325) \approx 75.6\%$.

Precision, Recall and F1 Score

F1 Score: The Harmonic Mean

A single metric that combines precision and recall. It gives more weight to low values, so a high F1 score requires both high precision and high recall.

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

For our model, F1 is 74.22.

The Precision-Recall Tradeoff

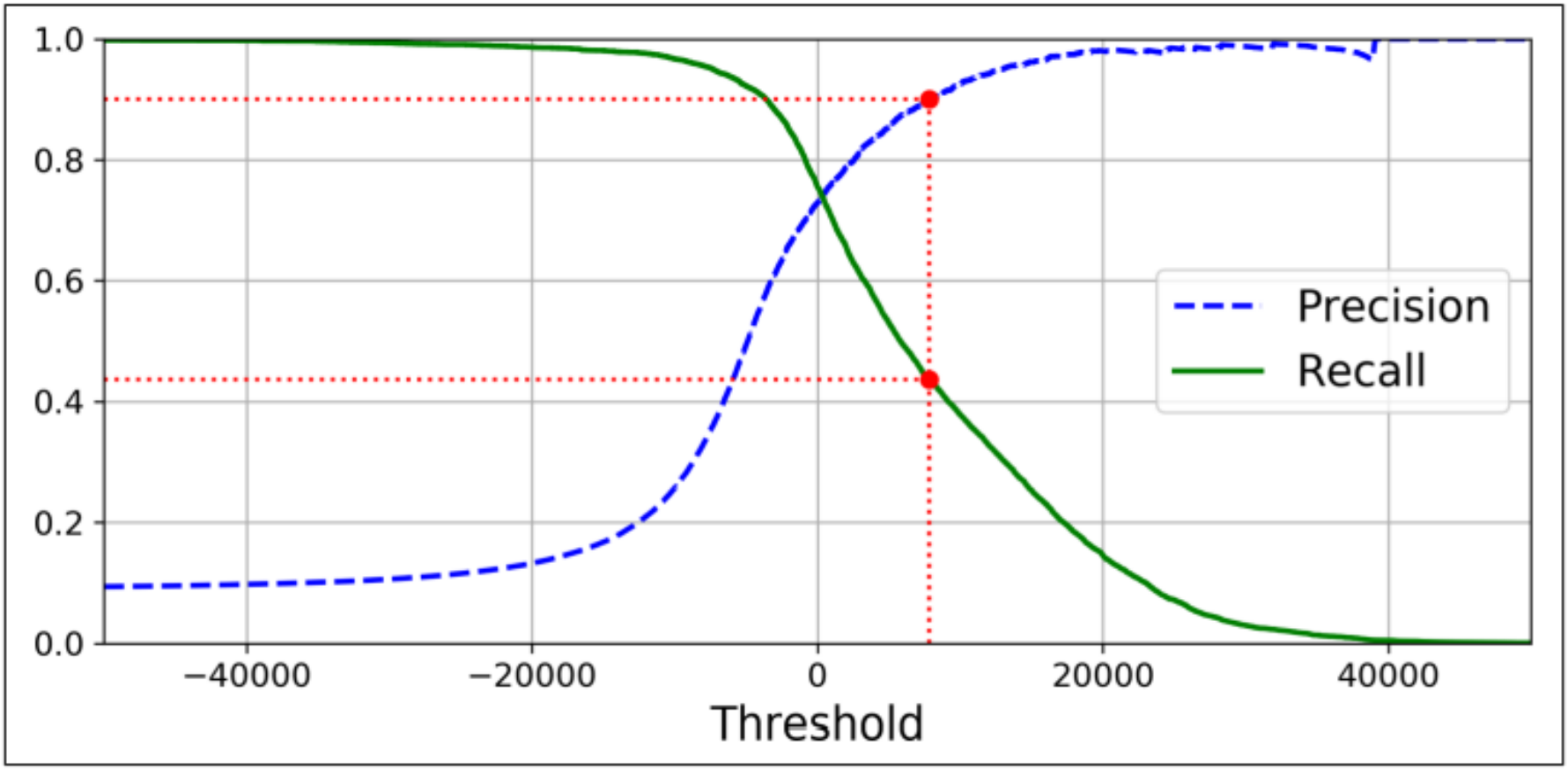


Figure: Plotting precision and recall against the decision threshold.

The Precision-Recall Trade-off

The Inherent Conflict

Unfortunately increasing precision reduces recall, and vice versa; the precision-recall trade-off.

How it Works: The Decision Threshold

- Classifiers compute a score for each instance. If the score is above a threshold, it's classified as positive.
- **Raising the threshold**: Increases precision (fewer false positives) but decreases recall (more false negatives).
- **Lowering the threshold**: Increases recall (fewer false negatives) but decreases precision (more false positives).

The ROC Curve

Receiver Operating Characteristic (ROC)

Another common tool for binary classifiers. It plots the **True Positive Rate (Recall)** against the **False Positive Rate (FPR)**.

- **FPR**: The ratio of negative instances that are incorrectly classified as positive.
- A good classifier stays as far away from the diagonal line as possible (toward the top-left corner).

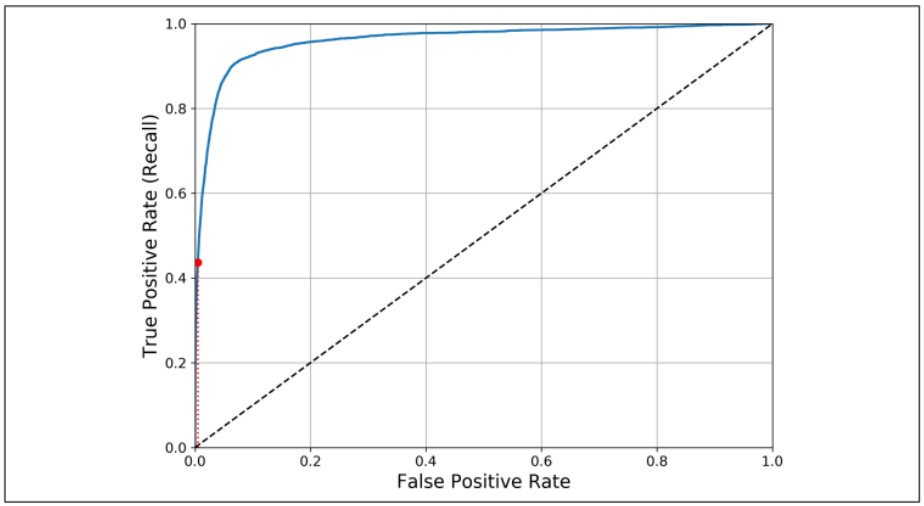


Figure 3-6. This ROC curve plots the false positive rate against the true positive rate for all possible thresholds; the red circle highlights the chosen ratio (at 43.68% recall)

Facts!

- A perfect classifier has an AUC of 1.
- A purely random classifier has an AUC of 0.5.

Receiver Operating Characteristic (ROC) – Rule of Thumb

When to use ROC vs. Precision-Recall?

Since the ROC curve is so similar to the precision/recall (PR) curve, one may wonder how to decide which one to use. As a rule of thumb, it is preferable to use the PR curve whenever the positive class is rare or when you care more about the false positives than the false negatives; otherwise, the ROC curve is suitable. For example, when looking at the ROC curve and the ROC AUC score for the digit classifier, one might think that the classifier is very good. However, this is mostly because there are few positives (5s) compared to the negatives (non-5s). In contrast, the PR curve makes it clear that the classifier has room for improvement, as the curve could be closer to the top-left corner.

Handling More Than Two Classes

Multiclass (or Multinomial) Classifiers

These classifiers can distinguish between more than two classes. Some algorithms (like SGD, Random Forests) support this natively. Others (like SVMs) are strictly binary.

One-vs-the-Rest (OvR)

Train 1 binary classifier for each class (e.g., a 0-detector, a 1-detector, etc.). To classify a new image, get the decision score from each classifier and pick the class with the highest score.

One-vs-One (OvO)

Train 1 binary classifier for every pair of classes (0 vs 1, 0 vs 2, 1 vs 2, etc.). For N classes, this requires N*(N-1)/2 classifiers. The class that wins the most "duels" is chosen.

Scikit-Learn automatically applies OvR or OvO based on the algorithm.

Improving Models by Analyzing Errors

The Multiclass Confusion Matrix

Just like with binary classification, we can create a confusion matrix to see where the model is making mistakes.

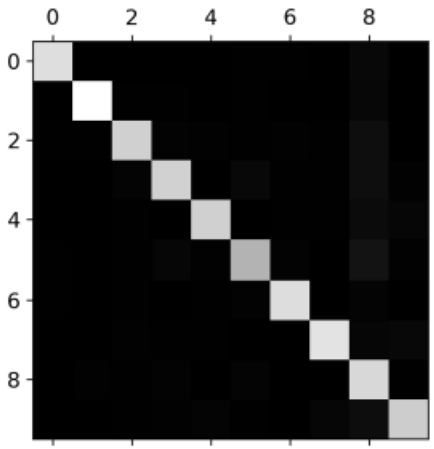


Figure: Confusion matrix

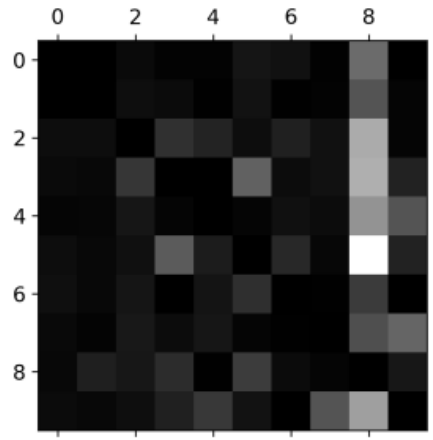


Figure: Rows are actual classes, columns are predicted.

- The column for class 8 is bright, meaning many other digits are misclassified as 8s.
- 3s - 5s & 7s/4s - 9s, are often confused.

Multilabel and Multioutput Classification

Multilabel Classification

A system that can output multiple binary classes for each instance.

- **Example**: A face-recognition system that identifies multiple people in one photo. If it sees Alice and Charlie, the output would be `[1, 0, 1]`.
- Evaluation can be done by calculating the F1 score for each label and averaging the result.

Multioutput Classification

A generalization of multilabel classification where each label can be multiclass (i.e., have more than two possible values).

- **Example**: A system that removes noise from an image. The input is a noisy image, and the output is a clean image.
- Here, the output is multilabel (one label per pixel) and each label is multiclass (pixel intensity from 0 to 255).

Questions?

Thank You!