# Classification Performance Metrics

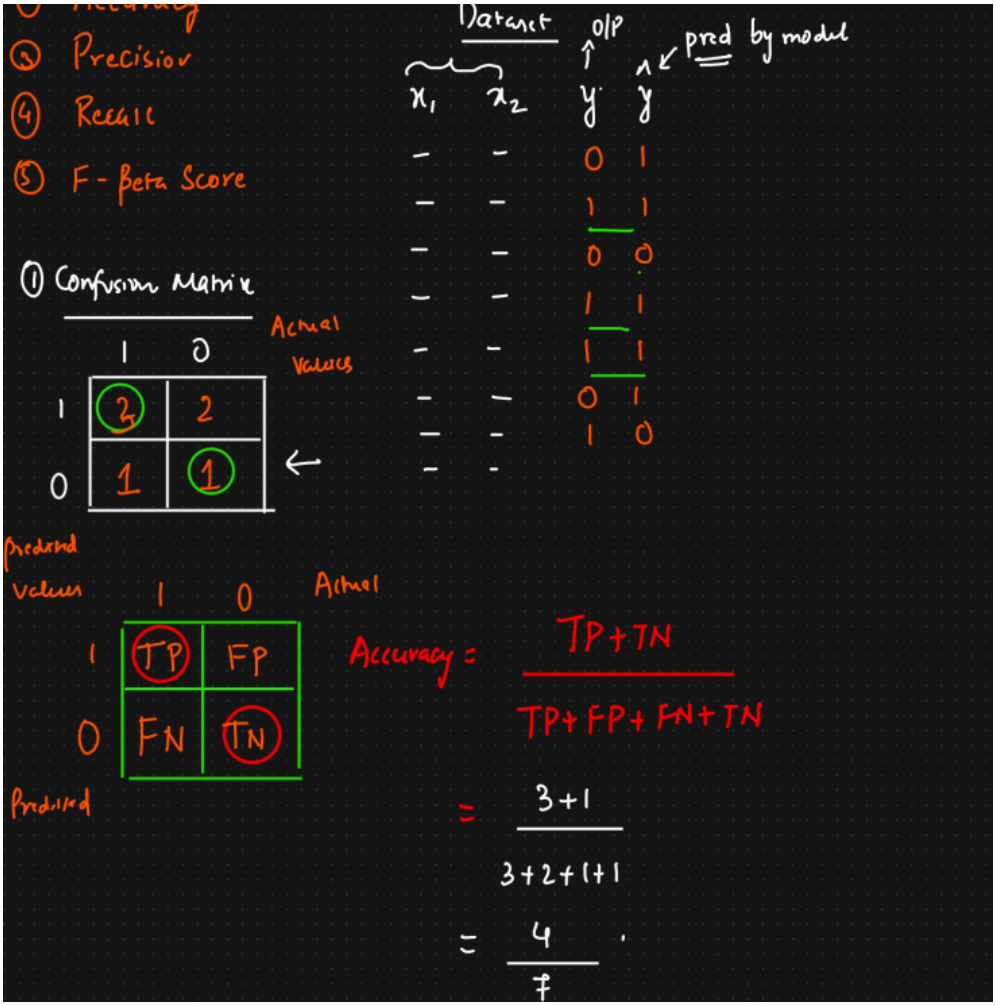
Evaluating classification models requires looking beyond simple accuracy, especially when dealing with imbalanced datasets or situations where different types of errors have different consequences. These metrics are typically derived from the **Confusion Matrix**.

## 1. The Confusion Matrix

A confusion matrix is a table that summarizes the performance of a classification algorithm. For a binary classification problem (Positive/Negative classes), it looks like this:

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

- **True Positive (TP):** The model correctly predicted Positive (e.g., correctly identified spam).
- **True Negative (TN):** The model correctly predicted Negative (e.g., correctly identified a non-spam email).
- **False Positive (FP):** The model incorrectly predicted Positive when it was actually Negative (Type I Error) (e.g., flagged a normal email as spam).
- **False Negative (FN):** The model incorrectly predicted Negative when it was actually Positive (Type II Error) (e.g., failed to detect a spam email).




## 2. Accuracy

- **What it is:** The most intuitive metric; represents the overall proportion of correct predictions (both TP and TN) among the total number of instances.
- **Formula:** `Accuracy = (TP + TN) / (TP + TN + FP + FN)`
- **Interpretation:** "What percentage of all predictions did the model get right?"
- **Pros:** Simple to understand and calculate.
- **Cons:** Can be highly misleading, especially on **imbalanced datasets**. If 99% of emails are not spam, a model that *always* predicts "not spam" will have 99% accuracy but is useless for detecting spam.
- **When to use:** Suitable when class distributions are balanced, and the cost of False Positives and False Negatives is roughly equal.

## 3. Precision (Positive Predictive Value)

- **What it is:** Measures the accuracy of positive predictions. Answers the question: "Of all instances the model predicted as Positive, how many were actually Positive?"

- **Formula:** `Precision = TP / (TP + FP)`
- **Interpretation:** High precision indicates that when the model predicts the positive class, it is likely correct. It measures the "quality" or "exactness" of positive predictions.
- **Focus:** Minimizing **False Positives (FP)**.
- **When to use:** When the cost of a False Positive is high.
    - *Example:* Email spam detection. You want high precision to avoid classifying important emails (non-spam) as spam (FP).
    - *Example:* Search engine results. You want high precision so that the top results returned (predicted positive/relevant) are actually relevant (TP), avoiding irrelevant results (FP).

## 4. Recall (Sensitivity, True Positive Rate - TPR)

- **What it is:** Measures the model's ability to find all the actual positive instances. Answers the question: "Of all the actual Positive instances, how many did the model correctly identify?"
- **Formula:** `Recall = TP / (TP + FN)`
- **Interpretation:** High recall indicates that the model identifies most of the actual positive cases. It measures the "completeness" or "quantity" of positive predictions found.
- **Focus:** Minimizing **False Negatives (FN)**.
- **When to use:** When the cost of a False Negative is high.
    - *Example:* Medical diagnosis (e.g., cancer detection). You want high recall to ensure you identify as many actual patients (Positives) as possible, minimizing missed diagnoses (FN).
    - *Example:* Fraud detection. You want high recall to catch as many fraudulent transactions (Positives) as possible, minimizing missed frauds (FN).

## 5. Precision-Recall Trade-off

- Often, increasing Precision leads to a decrease in Recall, and vice-versa. This is because adjusting the classification threshold (the probability cutoff used to decide between classes) affects TP, FP, and FN counts differently.
- For instance, requiring a very high probability threshold to classify as Positive will likely reduce FPs (increasing Precision) but may increase FNs (decreasing Recall).
- This trade-off can be visualized using a Precision-Recall curve.

## 6. F-beta Score (and F1-Score)

- **What it is:** A single metric that combines Precision and Recall using their harmonic mean. It provides a way to balance the two metrics.
- **Formula (General F-beta):** `Fβ = (1 + β²) * (Precision * Recall) / (β² * Precision + Recall)`

The general formula for non-negative real $\beta$ is:

$$F_\beta = \frac{(1 + \beta^2) \cdot (\text{precision} \cdot \text{recall})}{(\beta^2 \cdot \text{precision} + \text{recall})}$$

- **The Beta (β) parameter:** Controls the relative importance of Recall over Precision.
  - **`β = 1` -> F1-Score:** This is the most common form, giving equal weight to Precision and Recall. It's the harmonic mean of the two. `F1 = 2 * (Precision * Recall) / (Precision + Recall)` A high F1 score requires both Precision and Recall to be high.
  - **`β > 1` (e.g., F2-Score):** Gives more weight to Recall. Use when minimizing False Negatives is more important than minimizing False Positives.
  - **`0 < β < 1` (e.g., F0.5-Score):** Gives more weight to Precision. Use when minimizing False Positives is more important than minimizing False Negatives.
- **Why Harmonic Mean?** The harmonic mean penalizes low values more severely than the arithmetic mean. To get a high F-score, *both* Precision and Recall must be reasonably high. A model with very high Precision but near-zero Recall (or vice-versa) will have a low F-score.
- **When to use:** Useful when you need a single number to summarize performance, especially on imbalanced datasets where Accuracy is misleading. Use F1 for a balanced view; use F-beta (with `β ≠ 1`) when one metric (Precision or Recall) is explicitly more important based on the problem context.