Chapter 3 Classification

- **ROC Curve**:
  - Plots **True Positive Rate (Recall)** vs **False Positive Rate**.
  - AUC (Area Under Curve) shows classifier performance (1.0 = perfect, 0.5 = random).
  - Use **ROC Curve** when classes are balanced; otherwise, **PR Curve** is better.
- **Classifier Comparison**:
  - SGDClassifier uses decision_function(), RandomForestClassifier uses predict_proba().
  - Random Forest often has higher AUC and better performance than SGD.
- **Multiclass Classification**:
  - Involves predicting one label out of many (e.g., digits 0–9).
  - Strategies:
    - **One-vs-Rest (OvR)**: One classifier per class vs all others.
    - **One-vs-One (OvO)**: One classifier per class pair.
  - Scikit-Learn automatically applies OvR/OvO based on the algorithm.
- **Error Analysis**:
  - Use **confusion matrix** and normalized plots to inspect errors.
  - Helps identify which digits are most often misclassified (e.g., 3s vs 5s).
  - Preprocessing (e.g., centering images) can reduce misclassification.
- **Multilabel Classification**:
  - Each instance can have **multiple labels** (e.g., “is digit large?” and “is it odd?”).
  - Evaluate using **F1 score**, average by macro or weighted methods.
- **Multioutput Classification**:
  - Each label can take **multiple possible values** (e.g., image denoising where each pixel is a label).
  - Often blurs the line between classification and regression.

Training and Evaluating a Binary Classifier ("5-detector")

- **Objective**: Detect whether a digit is **5** or **not 5** using a binary classifier.
- **Model Used**: SGDClassifier from Scikit-Learn — efficient for large datasets and supports online learning.
- **Evaluation**:
  - Used **cross-validation** to check accuracy.
  - Achieved high accuracy (~96%), but this could be misleading due to **class imbalance**.
- **Baseline Check**:
  - A naive classifier predicting only "not 5" still achieves ~91% accuracy.
  - Shows that **accuracy alone is not reliable** with imbalanced datasets.
- **Better Evaluation – Confusion Matrix**:

  - Shows true/false positives and negatives.
  - Helps identify where the model is making errors.
  - For example:
  - [[53057 1522] → TN, FP
  - [1325 4096]] → FN, TP
- **Key Metrics**:
  - **Precision** = TP / (TP + FP): Measures how many predicted 5s are correct.
  - **Recall** = TP / (TP + FN): Measures how many actual 5s were correctly identified.
- **Takeaway**:
  - Use **precision and recall**, not just accuracy, especially with imbalanced classes.
  - Confusion matrices provide valuable insight for binary classification tasks.

- **ROC Curve**: Evaluates binary classifiers using True Positive Rate vs False Positive Rate. Best used with balanced datasets. ROC AUC = 1 (perfect), 0.5 (random).

- **Precision/Recall vs ROC**: Use **Precision/Recall curve** when the **positive class is rare** or **false positives are costly**.
- **Multiclass Classification**: For tasks with more than two classes (e.g., digit recognition).
  - **OvR** (One-vs-Rest): One classifier per class vs all others.
  - **OvO** (One-vs-One): One classifier per class pair.
  - Algorithms like **SGDClassifier** and **SVC** handle this internally.
- **Error Analysis**: Use a **confusion matrix** to understand misclassifications. Normalize it to highlight frequent mistakes (e.g., confusion between 5 and 3).
- **Multilabel Classification**: Each instance can belong to **multiple binary classes** (e.g., is a digit both “odd” and “large”?). Evaluated using macro or weighted **F1 score**.
- **Multioutput Classification**: Each instance has **multiple targets**, each with potentially multiple values (e.g., denoising an image pixel-by-pixel). Output is typically a high-dimensional vector.

SGD Classifier vs Random Forest Performance

Feature / Metric	SGD Classifier	Random Forest
Type	Linear model	Ensemble of decision trees
Speed (Training)	Very fast, especially for large datasets	Slower, especially on large datasets
Speed (Prediction)	Fast	Moderate to fast
Memory Usage	Low	High (stores multiple trees)
Handles Non-linear Data	Poorly (needs feature engineering)	Very well (trees handle non-linearity)
Robust to Outliers	Sensitive	More robust
Overfitting Risk	Lower (with regularization)	Higher (but mitigated with enough trees)
Scalability	High	Moderate
Interpretability	High (coefficients)	Medium (feature importance)
Performance on Noisy Data	Moderate (depends on regularization)	Generally good (ensemble smooths noise)
Use Case Suitability	Text classification, high-dimensional data	Complex structured data, imbalanced datasets

OvR (One-vs-Rest) vs OvO (One-vs-One) Multiclass Strategies

Aspect	One-vs-Rest (OvR)	One-vs-One (OvO)
Number of Classifiers	n classifiers for n classes	n(n-1)/2 classifiers for n classes
Training Time	Faster	Slower (more classifiers to train)
Prediction Time	Faster	Slower
Memory Usage	Lower	Higher
Binary Classification Base	One classifier distinguishes one class vs all	Each classifier distinguishes between two classes
Complexity	Simpler	More complex
Accuracy (in practice)	Good, but can struggle with ambiguous classes	Often better for fine-grained distinctions
Implementation	Easier	More complex
Preferred When	Large datasets, many classes	Small to medium datasets, fewer classes