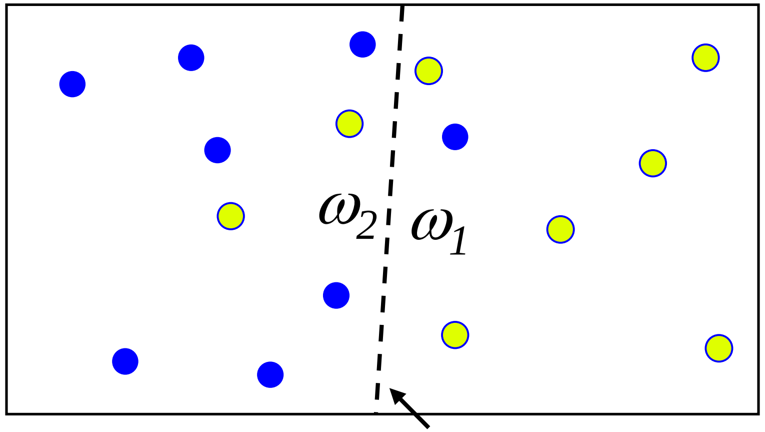
Classifier Evaluation

Classifier Evaluation

The most simple way to evaluate a classifier is to see the number of correct and incorrect classifications.

For a M-class problem with N samples to be classified, the **confusion matrix** is a MxM matrix, whose (i,j) element is the number of vectors that are actually in class ω_i and classified to class ω_j.

Example (2-class): ω_1 ○ ω_2 ●



decision boundary

Confusion Matrix: $\begin{pmatrix} 6 & 2 \\ 1 & 7 \end{pmatrix}$

Correct classification rate

= trace(Confusion Matrix)$/N$

Confusion Matrix Examples

A typical confusion matrix for the Iris dataset:

$$\begin{bmatrix} 50 & 0 & 0 \\ 0 & 46 & 4 \\ 0 & 0 & 50 \end{bmatrix}$$

Clothing color classification:

Pred / Real	Red	Orange	Yellow	Green	Blue	Pink	Purple	Brown	Gray	Black	White
Red	167	17	1	0	4	23	8	4	3	9	2
Orange	4	37	13	0	2	0	0	2	0	1	0
Yellow	3	1	87	5	0	3	0	5	3	1	3
Green	0	0	9	100	7	2	0	3	8	8	3
Blue	0	0	0	13	450	10	6	0	42	114	21
Pink	16	2	2	0	2	124	6	3	5	2	9
Purple	9	0	1	1	23	21	70	1	7	15	2
Brown	3	2	8	12	0	7	0	66	14	22	7
Gray	4	0	1	23	21	15	1	14	289	38	38
Black	10	1	0	15	44	15	15	5	49	903	9
White	1	0	2	7	29	26	2	4	52	9	322

Two-Class Confusion Matrix

Many two-class problems can be considered as "detection" problems where the classifier is expected to answer a "Yes/No" question for each sample, such as in a medical screening test.

Let class#1 be "No", class#2 be "Yes", confusion matrix be $\begin{pmatrix} TN & FP \\ FN & TP \end{pmatrix}$

Common metrics (in pairs) derived from the confusion matrix:

PD (probability of correct detection) = $TP / (TP + FN)$
FA (probability of false positive/alarm) = $FP / (TN + FP)$
Recall = **PD**
Precision = $TP / (TP + FP)$
Sensitivity = **PD**
Specificity = $TN / (TN + FP) = 1 -$ **FA**
PPV (positive predictive value) = **Precision**
NPV (negative predictive value) = $TN / (TN + FN)$

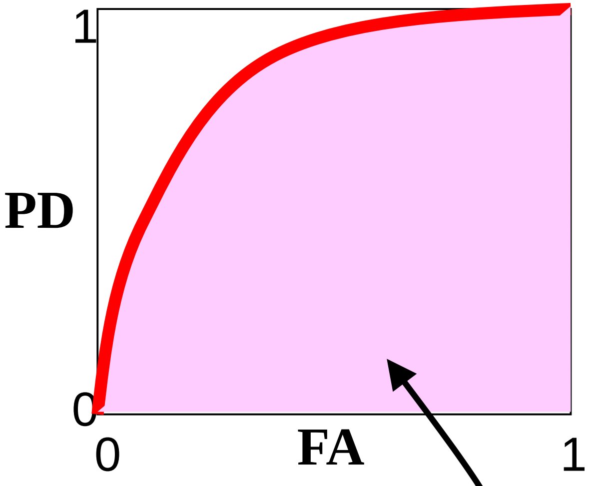
Two-Class Confusion Matrix

- A threshold / bias can be used to adjust how sensitive the classifier is to identify positive cases.
- This adjustment, while increasing one metric in a pair (e.g., **Recall**), is likely to decrease the other (e.g., **Precision**).
- **F1 measure** is one combined metric to allow for easier comparison between such paired classification results. It can also be used to select a “proper” threshold / bias.

$$F1 = \frac{2 * precision * recall}{precision + recall}$$

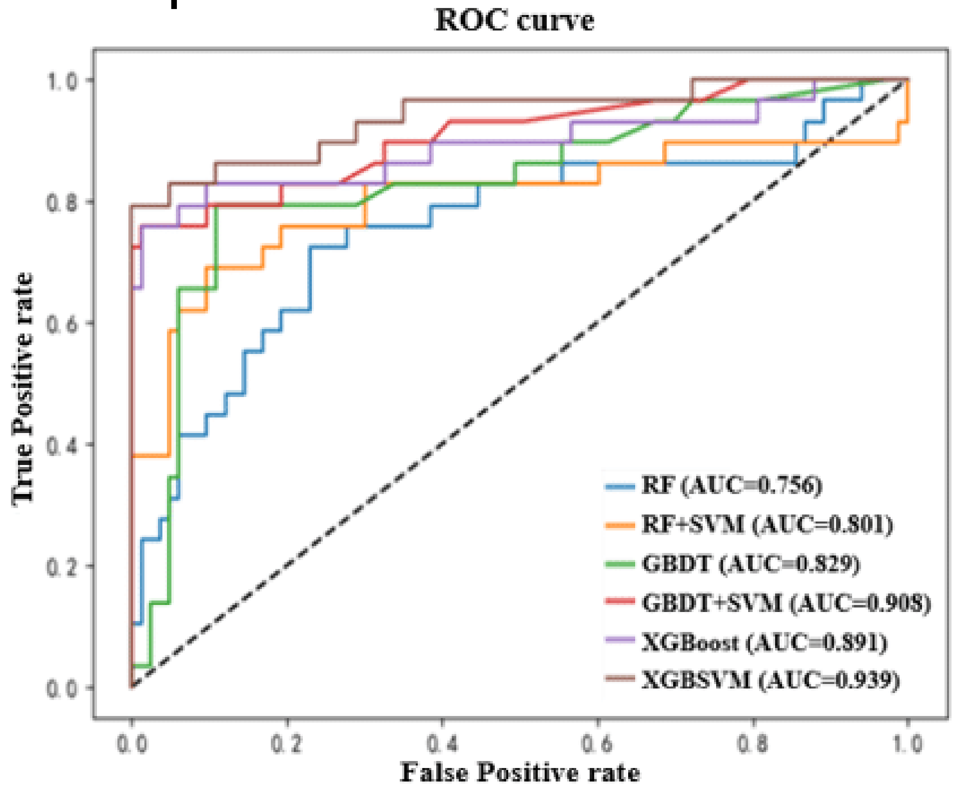
ROC Curves

Receiver Operating Characteristics (ROC) Curve is the plot of **PD** vs. **FA** at different threshold (bias) values. It allows the separation of the evaluation of different classification methods, settings, and/or features from the choice of the threshold.



AUC or **AUROC** (area under the ROC curve): The larger, the better.

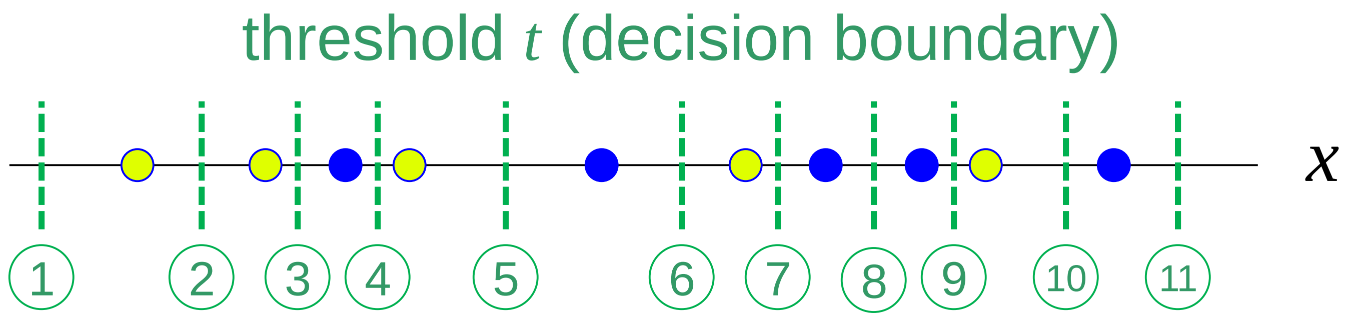
Example:



ROC Curves

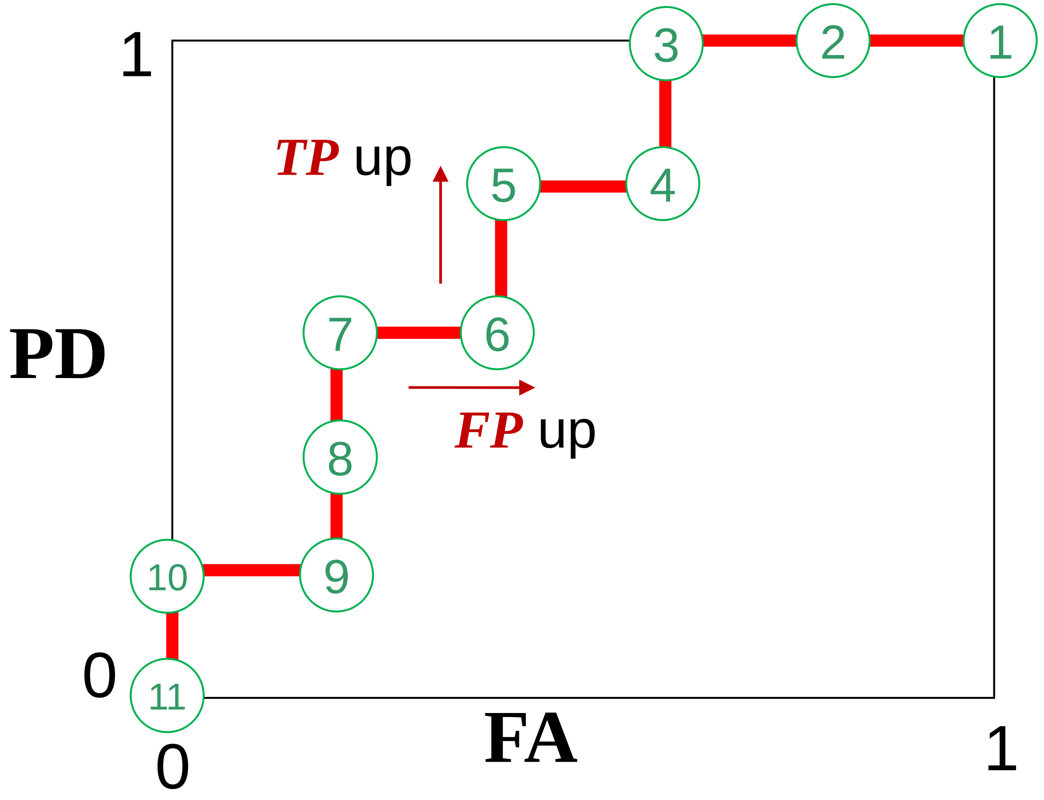
Example (1-D):

- negative • positive



PD = $TP \,/\, (TP + FN)$

FA = $FP \,/\, (FP + TN)$

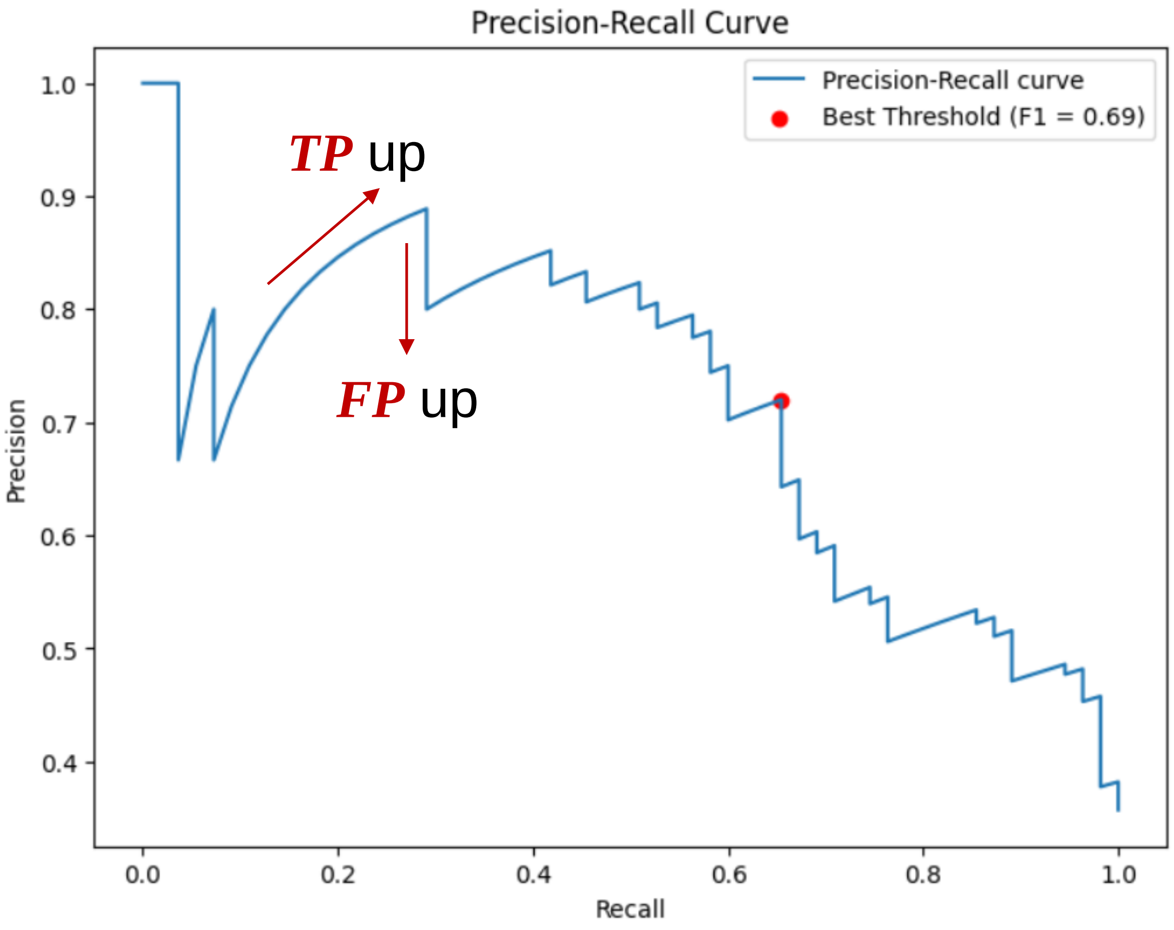


Precision-Recall Curves

Recall = $TP / (TP + FN)$

Precision = $TP / (TP + FP)$

Neither depends on TN, so PR curves are suitable for cases when TN is undefined, such as detection problems. The area under the curve (0~1) is usually called **average precision** (**AP**).



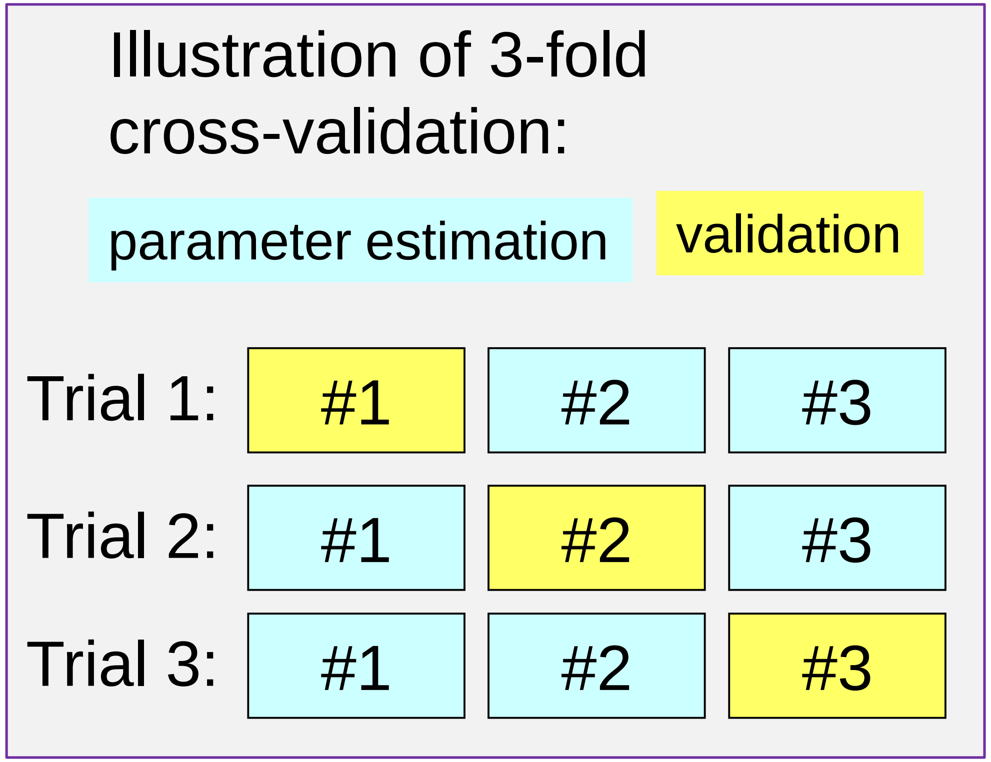
https://www.blog.trainindata.com/precision-recall-curves/

Dataset Division

- It is important that the labeled data for evaluation is separated from the labeled data used for training / model parameter estimation.
- The concept of validation: To estimate the performance of the model on previously unseen data.
- It's often that the majority of labeled data is used for training to get reliable models.
- The distributions of the training and validation/testing subsets are as similar as possible.
- The samples in the training and validation/testing subsets should be unrelated / uncorrelated when possible.
- With limited amount of labeled samples, cross-validation is commonly used to obtain reliable performance estimation.

Cross-Validation

- The N labeled samples are divided into K subsets of approximately equal sizes ($K>1$). They should have similar distributions.
- The training process (model parameter estimation) is run K times (K trials).
- In the K^{th} trial, the K^{th} subset is used for validation, and the other subsets are used for training.
- The overall performance is the combination of the performance on all the validation subsets.
- Extreme case: The leave-one-out method ($K=N$).



Classifier Evaluation w/ Cross-Validation

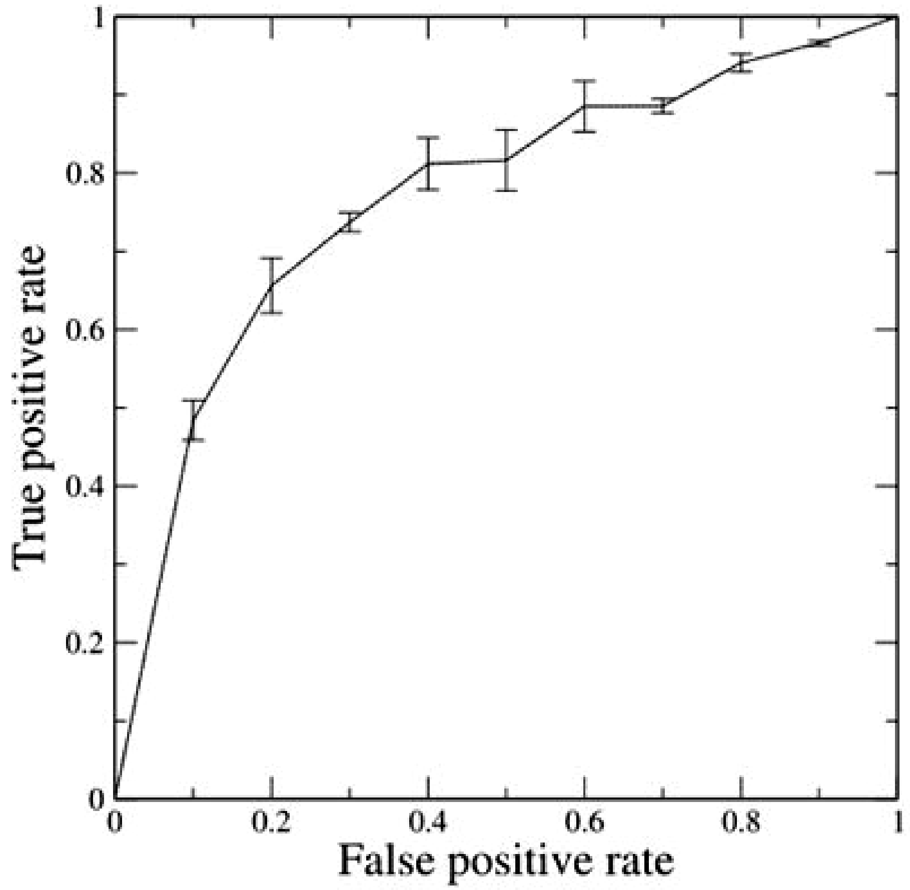
- Confusion matrix: Add the validation-subset confusion matrices from all the trials. Each sample is included exactly once. Metrics based on the confusion matrix, such as **PD** and **FA**, can then be computed.

- ROC curve (two-class problems):
  - Method 1: Just collect the outputs of the validation subsets from all the trials and draw a single ROC curve.
  - Method 2: Compute a ROC curve for each trial. We can average the curves at any given **FA** rate. The standard deviation gives us an estimation of the uncertainty of **PD** at any **FA** rate.
  - Method 3: Use a common set of thresholds to compute separate **PD** and **FA** values for all the validation subsets. This produces a single ROC curve with uncertainties for both **PD** and **FA**.

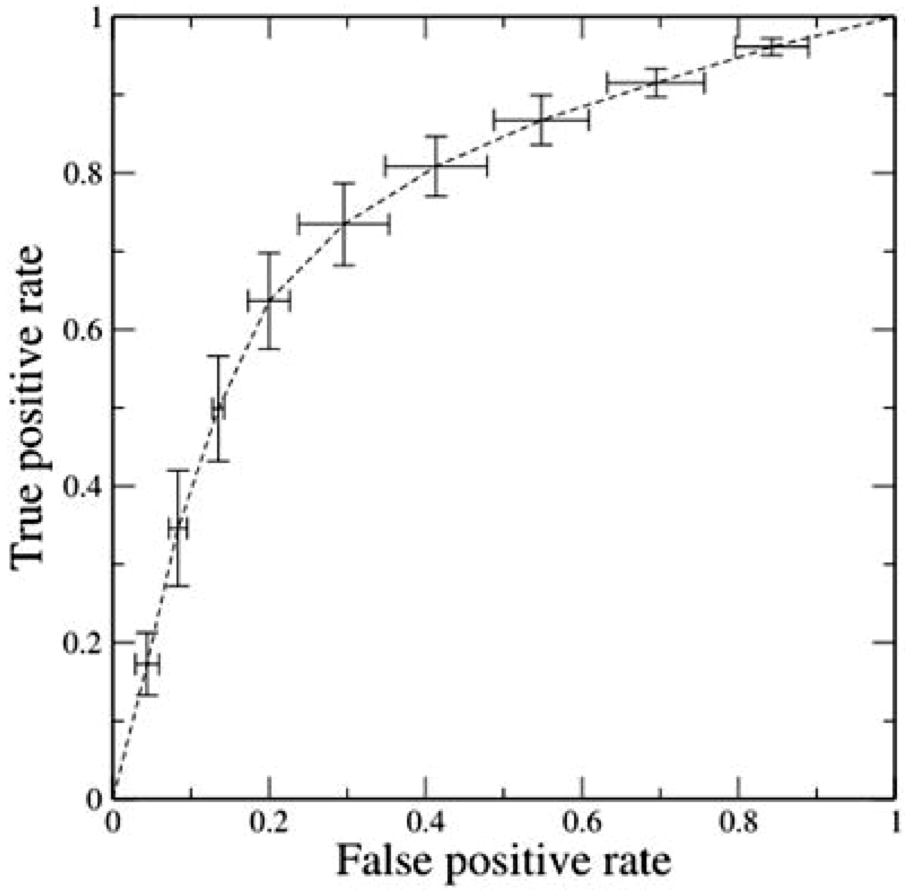
Classifier Evaluation w/ Cross-Validation

Examples of generating ROC curves with cross-validation:

(method 2)



(method 3)



https://slideplayer.com/slide/7028153/

Classification for Imbalanced Datasets

- Imbalanced datasets: Datasets where different classes have different proportions of samples.
- Without extra care, classifiers trained with imbalanced datasets tend to generate outputs that favor the majority class(es).
- Techniques for dealing with imbalanced datasets:
  - Oversampling the minority class and/or under-sampling the majority class
  - Class weighting
  - Generating extra samples for the minority class through synthesis, data augmentation, etc. (related to oversampling of the minority class)
  - Some classifier architectures are less sensitive to imbalanced data.

Classification for Imbalanced Datasets

Examples of imbalanced datasets:

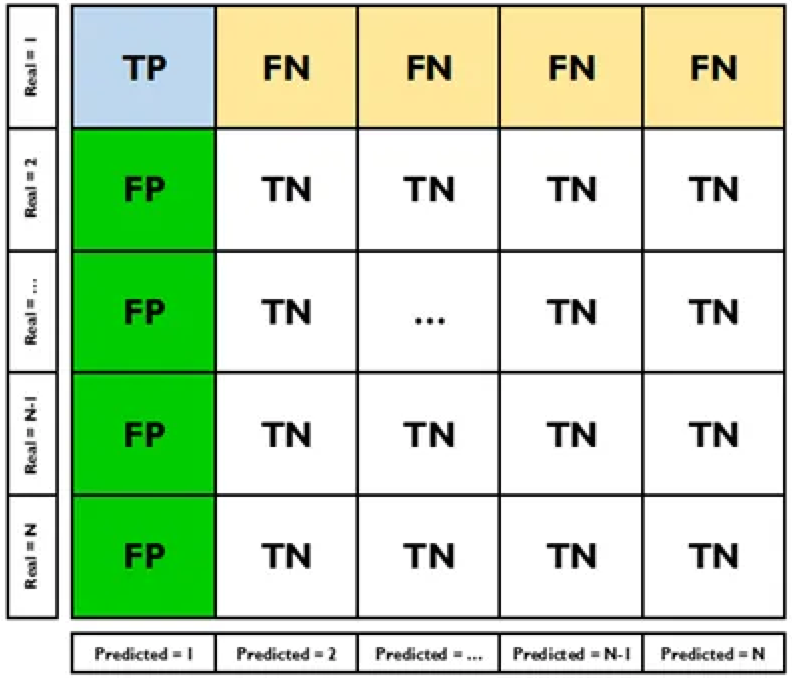
- ➔ ➔
- Anomaly detection:
  - Medical diagnosis
  - Fraud/spam detection
  - Accident/disaster detection
  - Defect detection
  - …
- How/where/when you do the dataset collection…

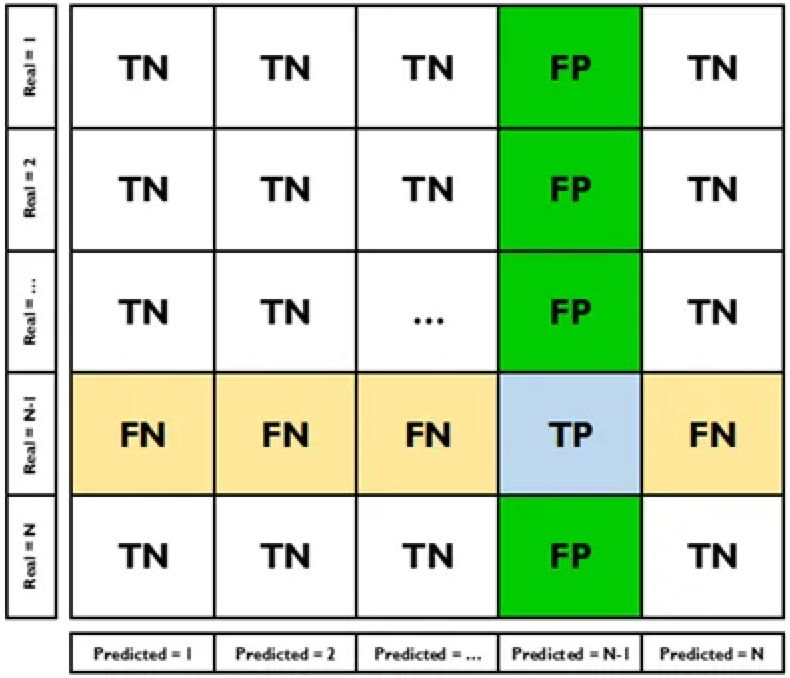
Clothing color classification:

Pred / Real	Red	Orange	Yellow	Green	Blue	Pink	Purple	Brown	Gray	Black	White
Red	167	17	1	0	4	23	8	4	3	9	2
Orange	4	37	13	0	2	0	0	2	0	1	0
Yellow	3	1	87	5	0	3	0	5	3	1	3
Green	0	0	9	100	7	2	0	3	8	8	3
Blue	0	0	0	13	450	10	6	0	42	114	21
Pink	16	2	2	0	2	124	6	3	5	2	9
Purple	9	0	1	1	23	21	70	1	7	15	2
Brown	3	2	8	12	0	7	0	66	14	22	7
Gray	4	0	1	23	21	15	1	14	289	38	38
Black	10	1	0	15	44	15	15	5	49	903	9
White	1	0	2	7	29	26	2	4	52	9	322

Evaluation for Multi-Class Classification

- Q: Other than accuracy, how are the other metrics extended to multi-class problems?
- The metrics for binary classifiers can be computed in a per-class **one-vs-all** basis:





https://medium.com/mcd-unison/multiclass-confusion-matrix-clarity-without-confusion-88af1494c1d1

Evaluation for Multi-Class Classification

- The evaluation metrics of all the classes can be combined to obtain the overall evaluation metrics.
- Example: mAP (mean AP) is the average AP of the different classes in multi-class object detection problems.
- **Micro-average**:
  - The per-class binary confusion matrix components (TP, FP, etc.) are summed over the classes before computing metrics.
  - Each item in the data contributes equally regardless of its class; micro-precision and micro-recall becomes the same as overall accuracy.
- **Macro-average**:
  - Metrics (PD, FA, etc.) are first computed separately for each class then averaged.
  - Macro-F1 is computed from macro-averaged precision and recall.
  - Each class contributes equally.

Evaluation for Multi-Class Classification

An example of multi-class ROC curves from scikit-learn: