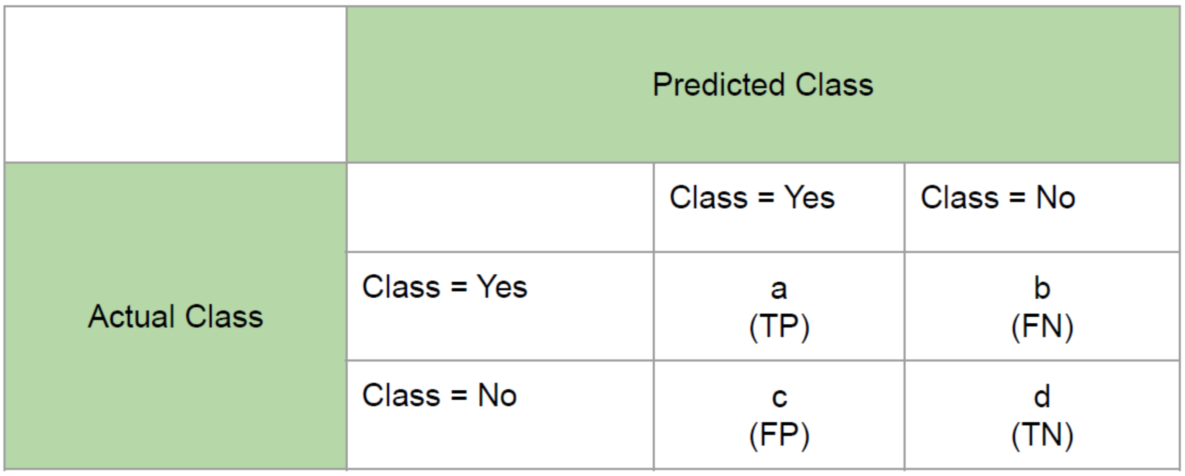
Lecture 13 A

Model Evaluation

Confusion Matrix



		Predicted Class	
		Class = Yes	Class = No
Actual Class	Class = Yes	a (TP)	b (FN)
	Class = No	c (FP)	d (TN)

Accuracy = (a + d) / (a + b + c + d)

- True Positive (TP): Model correctly predicts positive class.
- True Negative (TN): Model correctly predicts negative class.
- False Positive (FP): Model incorrectly predicts positive (Type I error).
- False Negative (FN): Model incorrectly predicts negative (Type II error).

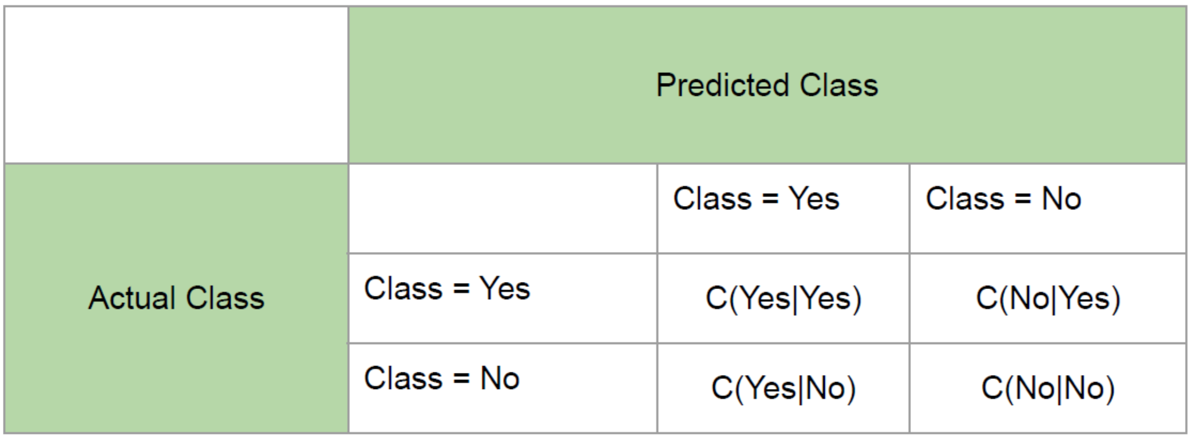
Situation	Possible Impact
High TP & High TN	Model is performing well overall.
High FP	Many false alarms, reduced precision.
High FN	Many misses, reduced recall; can be dangerous.
Low TP	Poor detection of positive cases; ineffective classifier.
Low TN	Poor identification of negative cases; many false alarms.

Trade-offs

- Reducing false positives often increases false negatives, and vice versa.
- The balance depends on the problem context (e.g., medical diagnosis usually prioritizes minimizing FN).

Cost Matrix

A Cost Matrix quantifies the costs (or penalties) associated with each possible outcome of a classification decision.



		Predicted Class	
		Class = Yes	Class = No
Actual Class	Class = Yes	C(Yes|Yes)	C(No|Yes)
	Class = No	C(Yes|No)	C(No|No)

Structure of a Cost Matrix (Binary Classification)

Actual \ Predicted	**Positive Prediction**	**Negative Prediction**
Positive Class	Cost of TP (often 0)	Cost of FN (high; missing positive)
Negative Class	Cost of FP (false alarm)	Cost of TN (often 0)

Why?

- In many situations, false positives and false negatives have very different consequences.
- Example in medical diagnosis:
  - False negative (missing disease) may be life-threatening → high cost.
  - False positive (false alarm) may cause anxiety or extra tests → lower cost.
- Optimizing purely for accuracy ignores these cost differences.
- Cost matrices help in:
  - Designing models sensitive to costs.
  - Selecting thresholds for classification.
  - Making decisions in risk-sensitive applications.

Accuracy alone does not always reflect the best model choice.

In cost-sensitive settings, the model with the lowest expected cost (from the cost matrix) is preferred—even if it does not have the highest accuracy.

Cost of Classification

COST		Predicted Class	
		Yes	No
Actual Class	Yes	-1	100
	No	1	0

Model 1		Predicted Class	
		Yes	No
Actual Class	Yes	150	40
	No	60	250

Accuracy = 80%
Cost = 3910

Model 2		Predicted Class	
		Yes	No
Actual Class	Yes	250	45
	No	5	200

Accuracy = 90%
Cost = 4255

Model 1:

Accuracy:

$$\frac{150+250}{150+40+60+250}=\frac{400}{500}=\frac{4}{5}=80\%$$

Cost Calculations:

$$Total\ Cost=\big(TP+cost(TP)\big)+\ \big(FN+cost(FN)\big)+\big(FP+cost(FP)\big)+\ \big(TN+cost(TN)\big)$$

$$=150(-1)+40(100)+60(1)+250(0)$$

$$-150+4000+60=3910$$

Model 2:

Accuracy:

$$\frac{250+200}{250+45+5+200}=\frac{450}{500}=\frac{9}{10}=90\%$$

Cost Calculations:

$$Total\ Cost=\big(TP+cost(TP)\big)+\ \big(FN+cost(FN)\big)+\big(FP+cost(FP)\big)+\ \big(TN+cost(TN)\big)$$

$$=250(-1)+45(100)+5(1)+200(0)$$

$$-250+4500+5=4225$$

- Model 1 has lower accuracy (80%) but lower total cost (3,910).
- Model 2 has higher accuracy (90%) but higher total cost (4,255).
- We will prefer Model 1

Some Metrics:

$$Accuracy \ = \frac{(TP + TN)}{TP + TN + FP + FN}$$

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

$$Recall\ or\ Sensi\ or\ TPR = \frac{TP}{TP + FN}$$

$$F1 = \frac{2(Precision * Recall)}{Precision + \ Recall} = \frac{2TP}{2TP + FP\ FN}$$

1. <u>Accuracy</u>
   - *Measures overall correctness:* proportion of total predictions the model got right.
   - Simple and intuitive metric.
   - *Limitation:* Can be misleading if classes are imbalanced (e.g., very high accuracy if majority class dominates).
2. <u>Precision</u>
   - Focuses on the quality of positive predictions.
   - High precision means few false positives—when the model says "positive," it's usually correct.
   - Important in scenarios where false alarms are costly (e.g., spam filtering).
3. <u>Recall (Sensitivity)</u>
   - Focuses on completeness in capturing positive cases.
   - High recall means few false negatives—the model finds most actual positives.
   - Vital when missing positives is critical (e.g., cancer detection).
4. <u>F1 Score</u>
   - Balances precision and recall into a single number.
   - Useful when you want a trade-off between precision and recall.
   - F1 is low if either precision or recall is low, highlighting poor performance in either dimension.

Methods of Estimation:

1. *<u>Holdout Method</u>*
    a. Working:
        i. Split the available dataset into two (or three) parts:
            1. Training set: Used to train the model.
            2. Testing set: Used to evaluate model performance on unseen data.
        ii. Sometimes a third validation set is created for hyperparameter tuning.
    b. Typical splits: 70%-30%, 80%-20%, or 60%-20%-20% (train/validation/test).
    c. Advantages:
        i. Simple and fast.
        ii. Easy to implement.
    d. Disadvantages:
        i. Performance estimate can be highly variable depending on how data is split.
        ii. Not suitable for small datasets because the model might not see enough data during training.
2. *<u>Cross-Validation (CV)</u>*
    a. Working:
        i. The data is split into k equal subsets (folds).
        ii. The model is trained and tested k times, each time using a different fold as the test set and the remaining k-1 folds as training data.
        iii. The performance measure is averaged over the k trials for a stable estimate.
    b. Common variant:
        i. <u>k-Fold Cross-Validation</u>, often with k=5 or k=10.
        ii. <u>Leave-One-Out Cross-Validation (LOOCV):</u> k = n, where n is the number of data points. Very computationally expensive but useful for small datasets.
    c. Advantages:
        i. Provides a more reliable and less biased estimate of model performance.
        ii. Better utilization of data: each sample is used for training and testing.
    d. Disadvantages:
        i. More computationally intensive than holdout.
        ii. Results depend on choice of k.

Ensemble Methods

Ensemble methods combine multiple models (often called “base learners” or “weak learners”) to create a stronger overall model that improves predictive performance and robustness compared to any individual model.

1. Bagging (Bootstrap Aggregating)
    a. Working:
        i. Multiple base models (often decision trees) are trained independently on different random bootstrap samples of the training data (sampling with replacement).
        ii. Each model makes predictions, and results are combined by majority voting (classification) or averaging (regression).
    b. Key ideas:
        i. Reduces variance and helps prevent overfitting.
        ii. Builds diverse models by training on varied subsets of data.
    c. Example: Random Forest, which combines many decision trees trained with bagging plus random feature selection.
2. Boosting
    a. Working:
        i. Models are trained sequentially. Each new model focuses on correcting the errors made by previous models.
        ii. Training assigns higher weights to misclassified examples so the next model pays more attention to these harder cases.
        iii. Final prediction is a weighted sum (or vote) of all models.
    b. Key ideas:
        i. Reduces bias and can greatly improve accuracy.
        ii. Models are dependent since each model learns from mistakes of the prior ones.
    c. Popular algorithms: AdaBoost, Gradient Boosting Machines (GBM), XGBoost, LightGBM, CatBoost.

- Bagging creates diverse models trained on different data samples to reduce variance.
- Boosting builds models sequentially, correcting errors iteratively to reduce bias and variance.
- Both improve robustness and predictive performance over single models.

When to Use Bagging:

- When base learners are high variance (complex, deep trees).
- When you want to improve stability and reduce overfitting.
- When you have enough computational resources to train many models independently.

When to Use Boosting:

- When base learners are weak learners (usually shallow trees).
- When you want to improve accuracy and reduce bias.
- When some overfitting can be controlled via regularization.