Confusion Matrix & Core Metrics

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

Accuracy
Accuracy = (a + d) / (a +b + c + d)
Accuracy is misleading when classes are imbalanced
(e.g., 10 positives out of 10,000 → a model predicting all “No” gets 99.9% accuracy).

Cost-Based Evaluation (Cost Matrix)
Sometimes accuracy isn't the goal — cost minimization is. Example cost matrix:

Cost:

Actual	**Predicted Yes**	**Predicted No**
Yes	-1	100
No	1	0

Model 1: Accuracy = 80%, Cost = 3910

Actual	**Predicted Yes**	**Predicted No**
Yes	150	40
No	60	250

Model 2: Accuracy = 90%, Cost = 4255

Actual	**Predicted Yes**	**Predicted No**
Yes	250	45
No	5	200

Different errors have different costs.
A model with lower accuracy may have lower total cost → more desirable.

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

Methods of Estimation

A. Holdout Method
- Train/test split (e.g., 75% train, 25% test).
- Simple but less reliable with small datasets.

B. Cross-Validation (K-fold)
- Partition into k equal folds.
- Train on k−1 folds, test on the remaining one.
- Average performance across folds.
- Leave-one-out (LOO) = k = n (max reliability, slowest compute).

C. Validation Set
- 3-way split: Training → Validation → Testing
    - Use validation for tuning parameters (hyperparameters).
    - Use test only once at the end.

Ensemble Methods
Goal: Combine multiple weak models → strong model

Underlying idea:
If each model has error rate ϵ, and errors are independent, majority vote reduces probability of incorrect output dramatically.

Example:
17 classifiers, each with error 0.20 →
ensemble error = P(at least 9 mistakes out of 17)

How to Create Independent Classifiers
Two main strategies:

A. Bagging (Bootstrap Aggregating)
How it works:
1. Create many bootstrap samples (sampling with replacement).
2. Train a classifier on each sample.
3. Final prediction = majority vote (classification) or average (regression).

Properties:
- Reduces variance.
- Helps unstable models (e.g., decision trees).

- Each classifier sees a slightly different dataset.

B. Boosting

An adaptive process where each new classifier focuses on mistakes of the previous ones.

Key ideas:

- Start with all samples weighted equally.
- Increase weights of misclassified points.
- Decrease weights of correctly classified points.
- Combine classifiers via weighted vote (models with lower error get higher weight).

Boosting characteristics:

- Often much more accurate than bagging.
- More prone to overfitting.
- Learns complex boundaries.

Boosting Example (from slides)

Five classifiers produce predictions:

- C1: predicts 1 (weight .25)
- C2: 0 (.1)
- C3: 0 (.5)
- C4: 1 (.05)
- C5: 0 (.1)

Final decision = weighted vote → class with larger total weight wins.

Bagging vs Boosting Summary

Aspect	Bagging	Boosting
Sampling	Random bootstrap	Reweighted sampling
Goal	Reduce variance	Reduce bias + variance
Focus	Independent models	Sequentially dependent models
Overfitting	Less likely	More likely
Speed	Parallelizable	Sequential → slower

Why Ensembles Work

They:

- Reduce noise.
- Reduce variance.
- Reduce overfitting (especially bagging).
- Capture complex decision boundaries (boosting).

Core principle: different perspectives give a more stable final answer.