

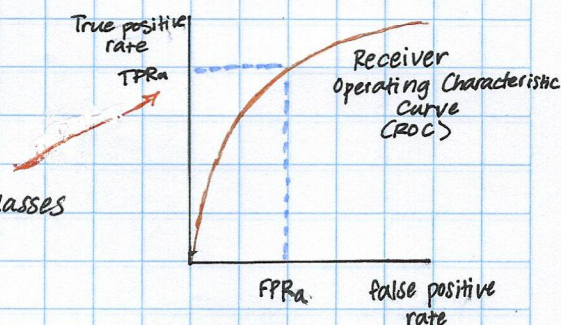
comparing trial programs

OPTIMIZATION, EVALUATION, & CLASSIFICATION



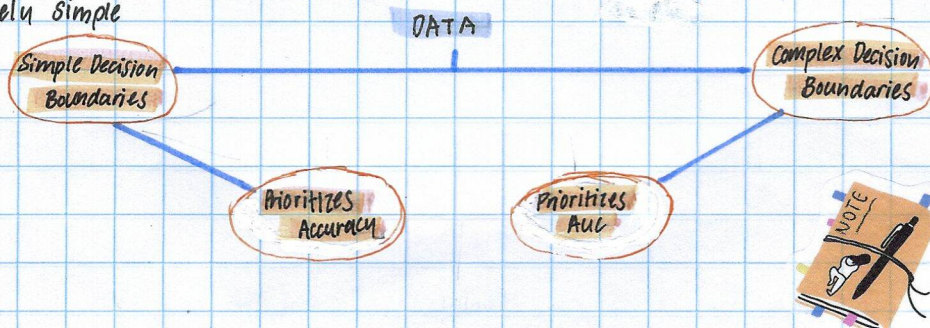
Main Question:

- ↳ How do different learning programs optimize performance?
- ↳ Metrics commonly used:
 - ↳ Accuracy → how many predictions are correct overall
 - ↳ AUC (Area under ROC curve) → how well the model separates classes
 - ↳ Logit Boost score → measures improvement through boosting
 - ↳ BAGging performance → measures stability & variance reduction



ACA VS. MLS: main comparison

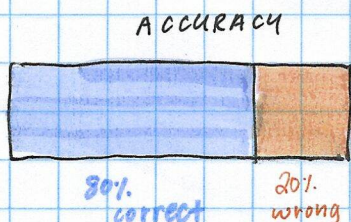
- ↳ Anomalous Class-Adapted Method (ACA) VS. Multilevel Sequencing (MLS)
- ↳ focuses on maximizing classification
- ↳ often emphasizes interpretability
- ↳ performs better when relationships are relatively simple
- ↳ uses more complex statistical relationships
- ↳ can capture nonlinear patterns
- ↳ often optimized using ensemble methods



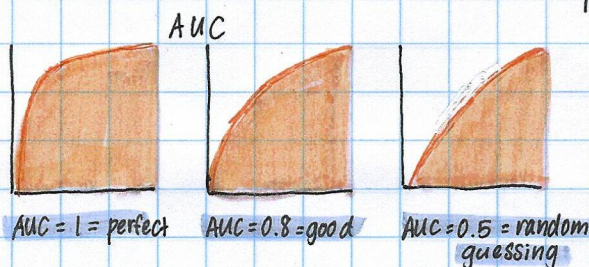
Accuracy & AUC:

- ↳ Accuracy measures the fraction of correct predictions = $\frac{TP + TN}{TP + TN + FP + FN}$
 - ↳ TP = true positives
 - ↳ TN = true negatives
 - ↳ FP = false positives
 - ↳ FN = false negatives
- ↳ AUC measures how well the model ranks positives over negatives
 - ↳ higher AUC means better separation
 - ↳ especially useful when classes are imbalanced & false positives and false negatives have different costs

Typically use BOTH



VS.



anything below 0.5 is considered bad

LogitBoost:

↳ common boosting algorithm used for classification

↳ key idea: instead of building one powerful classifier, build many weak classifiers & repeatedly focus on mistakes

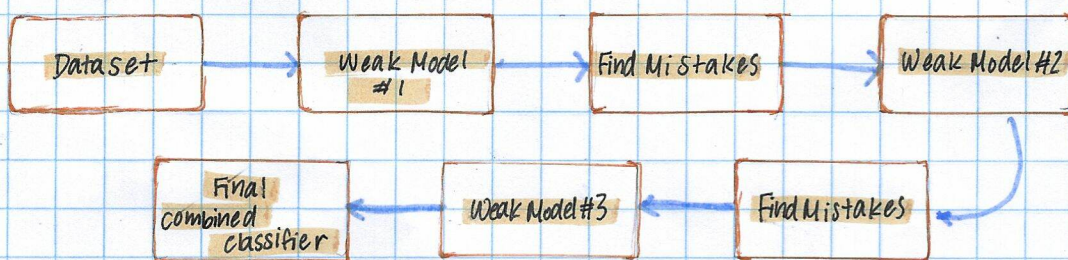
↳ process:

1. Train a weak learner
2. Identify errors.
3. Increase emphasis on misclassified points.
4. Train again. (might need to repeat over & over)
5. combine all models

↳ Key properties:

- ↳ uses logistic regression ideas
- ↳ Improves classification accuracy
- ↳ Sequential learning
- ↳ Reduces bias

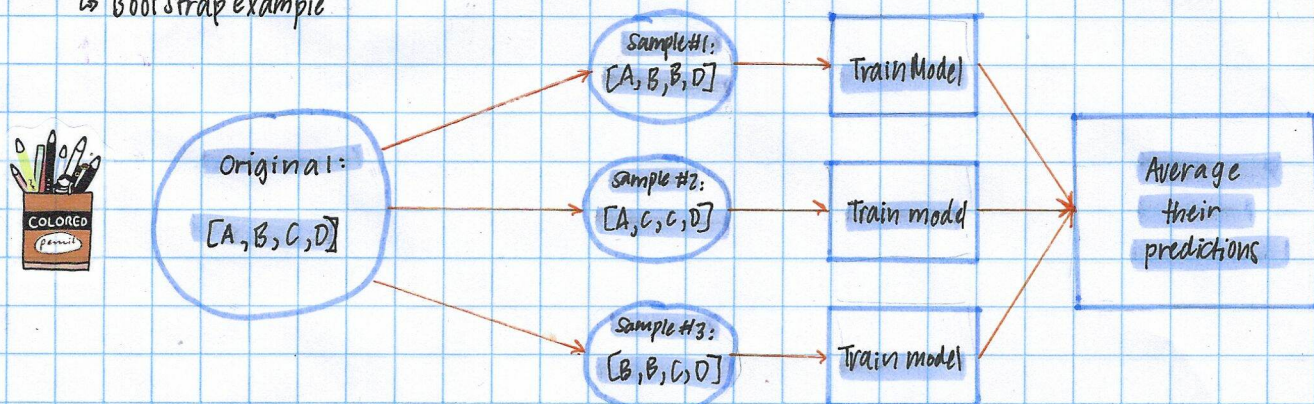
↳ Logit Boost flow example



Bagging / Bootstrap Aggregating:

↳ Bagging reduces variance by training multiple models on different samples of the data.

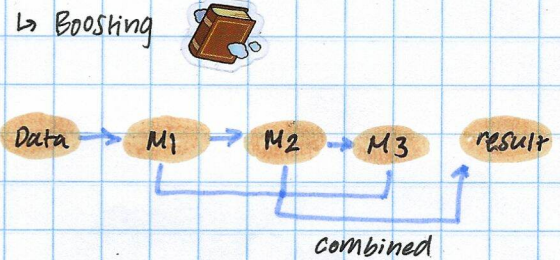
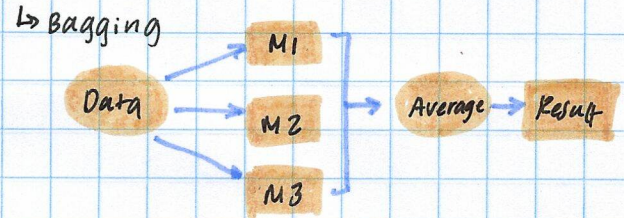
↳ Bootstrap example



↳ Why bagging works: reduces over fitting, makes predictions more stable, works especially well for decision trees, lowers variance

Boosting vs. Bagging:

BAGGING	BOOSTING
Models trained <i>independently</i>	Models trained <i>sequentially</i>
Reduces <i>variance</i>	Reduces <i>bias</i>
<i>Parallel</i> training	<i>Step-by-step</i> training
<i>Voting / Averaging</i>	<i>Weighted combination</i>
Example: Random Forest	Example: LogitBoost



The Bias-Variance Trade-off:

↳ Machine learning models generally fall in 2 ways:

↳ high bias (underfitting)

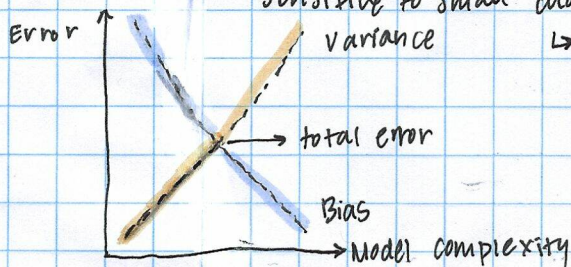
- model is too simple
- misses important patterns
- performs poorly on both training & testing data
- often caused by overly restrictive assumptions

ex: linear models on non-linear data, small decision trees

↳ high variance (overfitting)

- model memorizes training data
- performs extremely well on training examples
- generalizes poorly to unseen data
- sensitive to small changes in data set

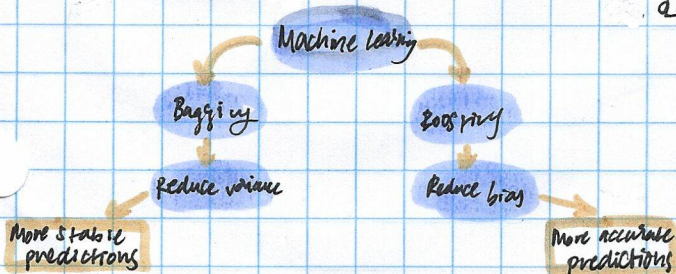
ex: very deep trees, complex neural networks with regularization



↳ connection to Bagging & Boosting

↳ Goal: Reduce variance for Bagging → trains multiple models independently & averages → stabilizes predictions, prevents overfitting, works especially well for high-variance models, risk of underfitting

↳ Goal: reduce bias for Boosting → Boosting forces later models to focus on mistakes → learns increasingly complex patterns, can improve weak learners, risk of overfitting if pushed too far



Confusion Matrix:

↳ a confusion matrix summarizes prediction performance

	Predicted +	Predicted -
Actual +	TP = correct + prediction	FN: missed detection
Actual -	FP = false alarm	TN = correct - prediction



↳ $\text{precision} = \frac{TP}{TP + FP}$ → how many predicted positives were actually positives?

↳ $\text{recall} = \frac{TP}{TP + FN}$ → how many actual positives were identified?

↳ $F_1 \text{ score} = F_1 = 2 \left(\frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \right)$ → balances precision & recall

↳ why accuracy can be misleading: suppose data set contains 990 healthy & 10 sick patients, a model predicts every one is healthy

↳ $\text{Accuracy} = 990/1000 = 99\%$ → looks good

↳ $\text{Recall} = 0/10 = 0$ (TP = sick in this case) → fails to identify ANY sick patients

Ensemble Learning Summary:

↳ An ensemble model combines multiple learners to improve performance

↳ Higher accuracy

↳ More robust predictions

↳ Better generalization

↳ Reduced sensitivity to noise

} advantages

↳ More computationally expensive

↳ Harder to interpret

↳ longer training time

} disadvantages

