Week 7

Machine Learning and Big Data - DATA622

CUNY School of Professional Studies

Ensembles (Making *weak learners* better)

Ensemble Methods

- Combination of weak learners to increase accuracy and reduce overfitting.
- Train multiple models with a common objective and fuse their outputs. Multiple ways of fusing them, can you think of some?
- Main causes of error in learning: noise, bias, variance. Ensembles help reduce those factors.
- Improves stability of machine learning models. Combination of multiple learners reduces variance, especially in the case of unstable classifiers.

Ensemble Methods

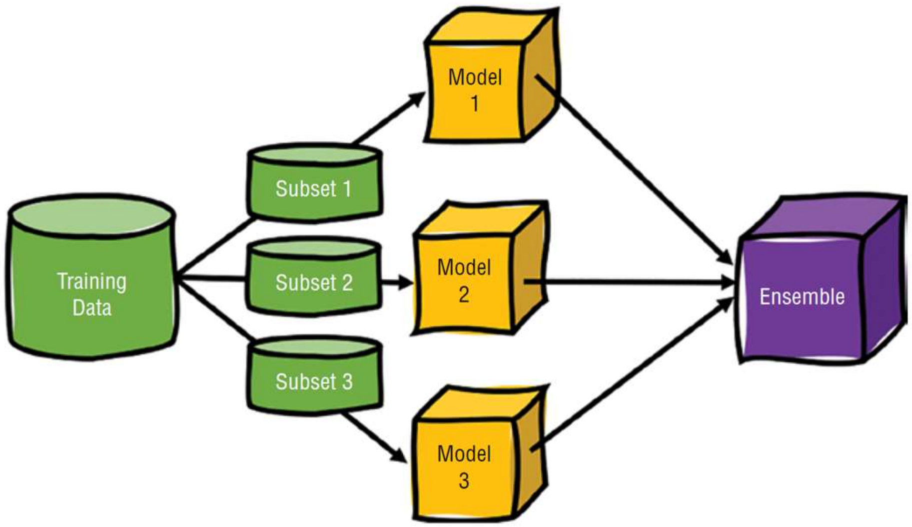
- Typically, decision trees are used as base learners.
- Ensembles usually retrain learners on subsets of the data.
- Multiple ways to get those subsets:
  - Resample original data with replacement: Bagging (decrease variance)
  - Resample original data by choosing troublesome points more often: Boosting (decrease bias)
- The learners can also be retrained on modified versions of the original data (gradient boosting).

Boosting & Bagging

CUNY School of Professional Studies

Bagging

- Boosting works by iteratively creating models and adding them to the ensemble
- Iteration stops when a predefined number of models have been added
- Each new model added to the ensemble is biased to pay more attention to instances that previous models misclassified (weighted dataset).

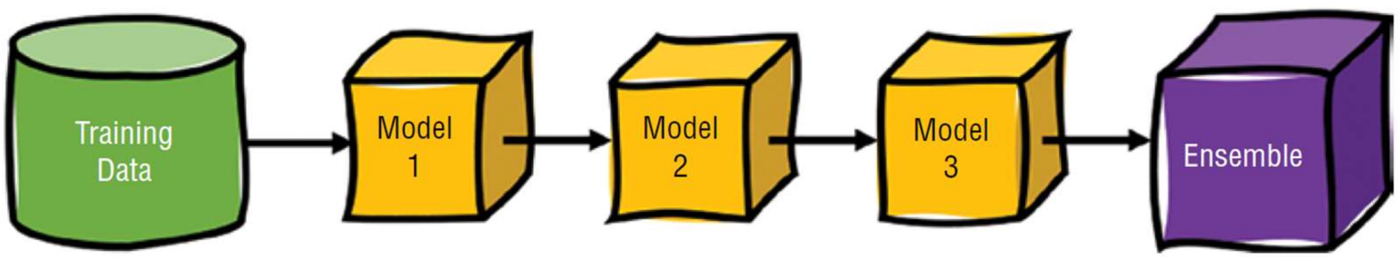


Bagging

- Ensemble method that "manipulates the training set"
- Action: repeatedly sample with replacement according to uniform probability distribution
  - Every instance has equal chance of being picked
  - Some instances may be picked multiple times; others may not be chosen
- Sample Size: same as training set
- Di: each bootstrap sample
- Footnote: also called bootstrap aggregating

Boosting

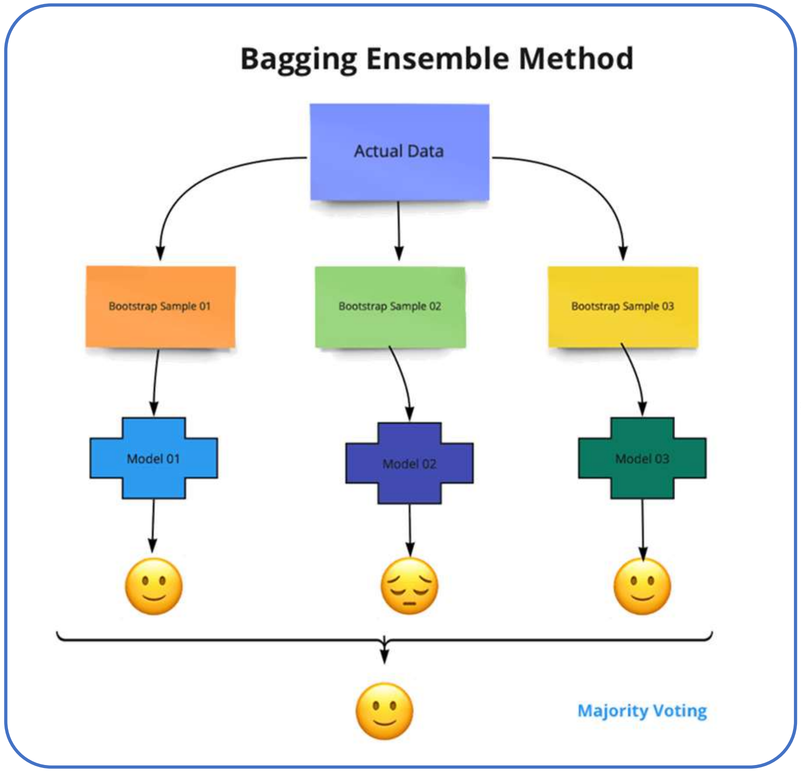
- Boosting works by iteratively creating models and adding them to the ensemble
- Iteration stops when a predefined number of models have been added
- Each new model added to the ensemble is biased to pay more attention to instances that previous models misclassified (weighted dataset).

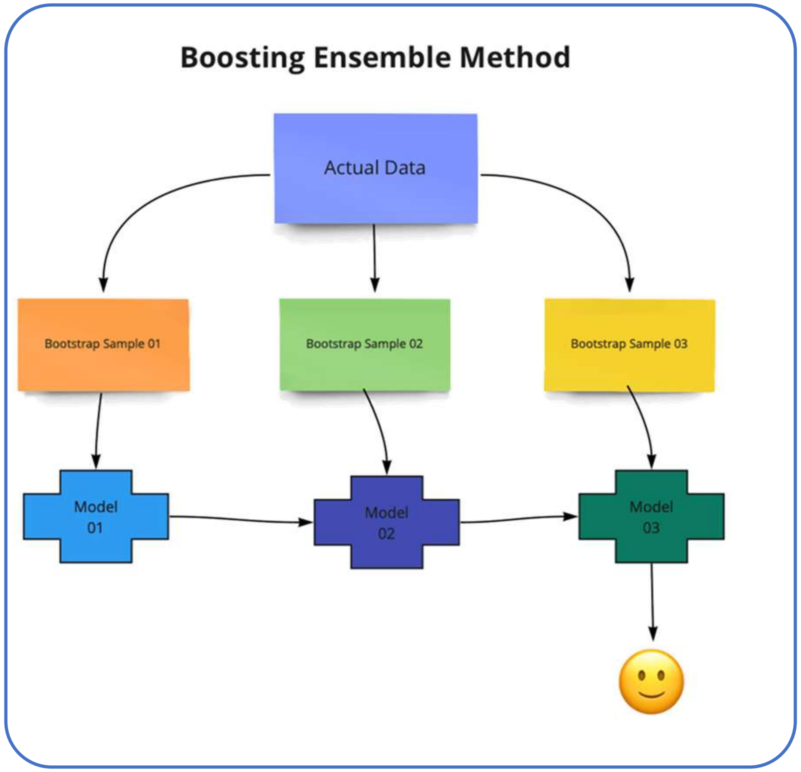


Boosting

- Sequential algorithm where at each step, a weak learner is trained based on the results of the previous learner.
- Two main types:
  - Adaptive Boosting: Reweight datapoints based on performance of last weak learner. Focuses on points where previous learner had trouble. Example: AdaBoost.
  - Gradient Boosting: Train new learner on residuals of overall model. Constitutes gradient boosting because approximating the residual and adding to the previous result is essentially a form of gradient descent. Example: XGBoost.

Bagging & Boosting





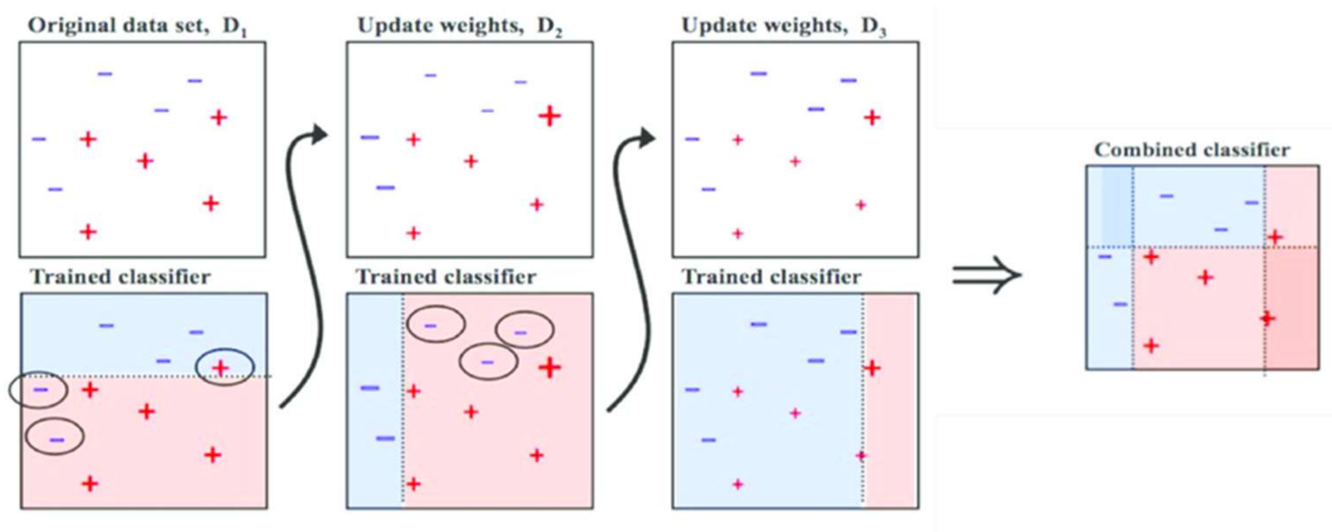
CUNY School of Professional Studies

Adaptive Boosting (AdaBoost)

CUNY School of Professional Studies

AdaBoost

- AdaBoost is essentially a boosting algorithm. It reweights the dataset before each new subsampling based on the performance of the last classifier.
- Main difference with bagging: it is SEQUENTIAL.



AdaBoost

- Input is a set of training examples (x_i, y_i) i = 1 to m.
- We are going to train a sequence of weak classifiers, such as decision trees, neural nets or SVMs. Weak because not as strong as the final classifier.
- The training examples will have weights, initially all equal.
- At each step, we use the current weights, train a new classifier, and use its performance on the training data to produce new weights for the next step.
- But we keep ALL the weak classifiers.
- When it's time for testing on a new feature vector, we will combine the results from all of the weak classifiers.

AdaBoost

- Automatically adjusts its parameters to the data based on the actual performance in the current iteration i.e. both the weights for re-weighting the data and the weights for the final aggregation are re-computed iteratively.
- In practice, this boosting technique is used with simple classification trees or stumps as base-learners, which resulted in improved performance compared to the classification by one tree or other single base-learner.

14

xGBoost (Extreme Gradient Boosting)

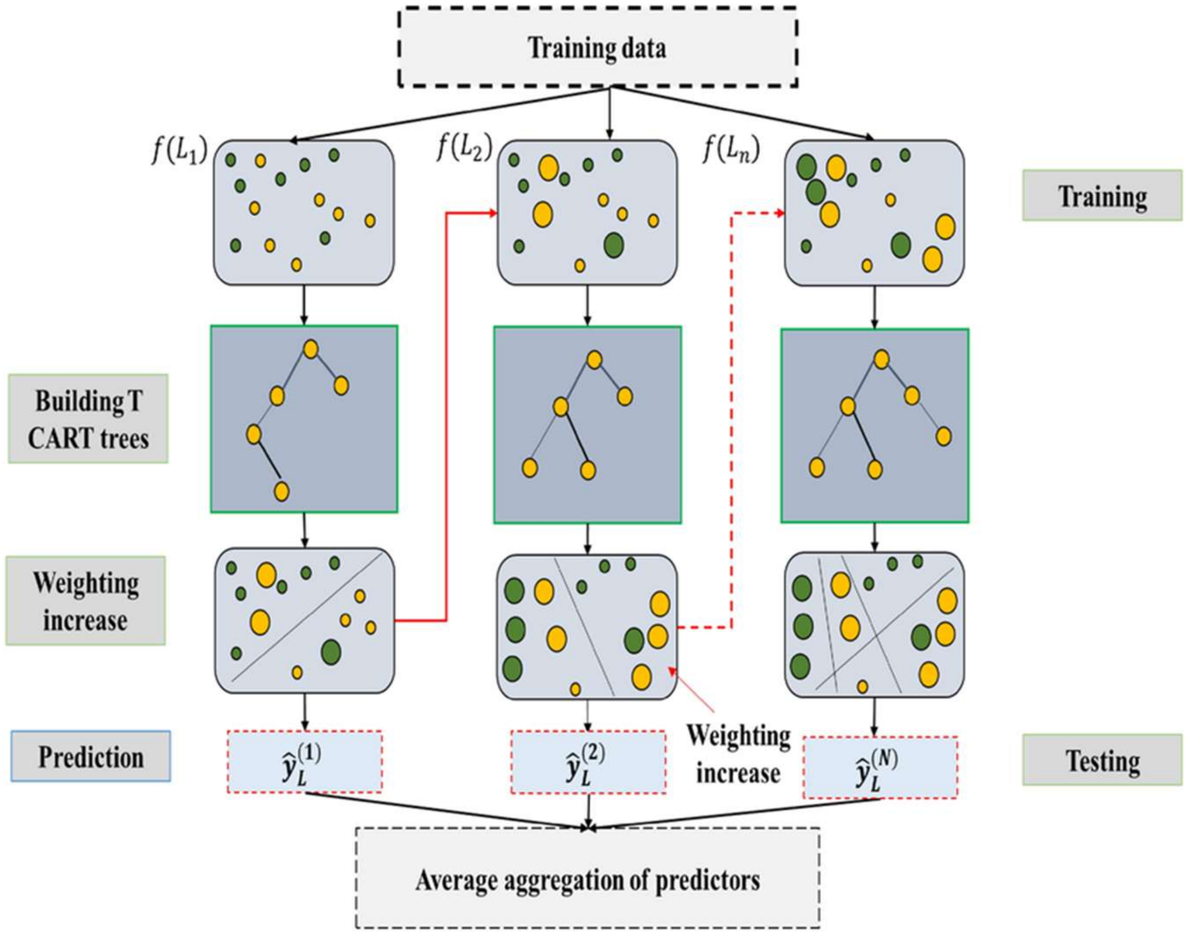
Gradient-based boosting algorithm

CUNY School of Professional Studies

xgBoost

- Gradient Boost is a robust machine learning algorithm made up of Gradient descent and Boosting.
- The word ‘gradient’ implies that you can have two or more derivatives of the same function.
- Gradient Boosting has three main components: additive model, loss function and a weak learner.
- The technique yields a direct interpretation of boosting methods from the perspective of numerical optimization in a function space and generalizes them by allowing optimization of an arbitrary loss function.

xgBoost



Training data
$f(L_1)$
$f(L_2)$
$f(L_n)$
Training
Building T CART trees
Weighting increase
Weighting increase
Prediction
$\hat{y}_L^{(1)}$
$\hat{y}_L^{(2)}$
$\hat{y}_L^{(N)}$
Testing
Average aggregation of predictors

AdaBoost vs Gradient Boost

	AdaBoost	Gradient Boost
Loss Function	Minimizes the exponential loss function that can make the algorithm sensitive to the outliers	any differentiable loss function can be utilized. Gradient Boosting algorithm is more robust to outliers than AdaBoost.
Flexibility	Designed boosting algorithm with a particular loss function	Gradient Boosting is more flexible than AdaBoost.
Benefits	Minimizes loss function related to any classification error and is best used with weak learners	Very good predictive accuracy
Shortcomings	Identified by high-weight data points	Shortcomings of the existing weak learners can be identified by gradients
Use-cases	Binary classification problems	Used for both classification and regression problems

Random Forest

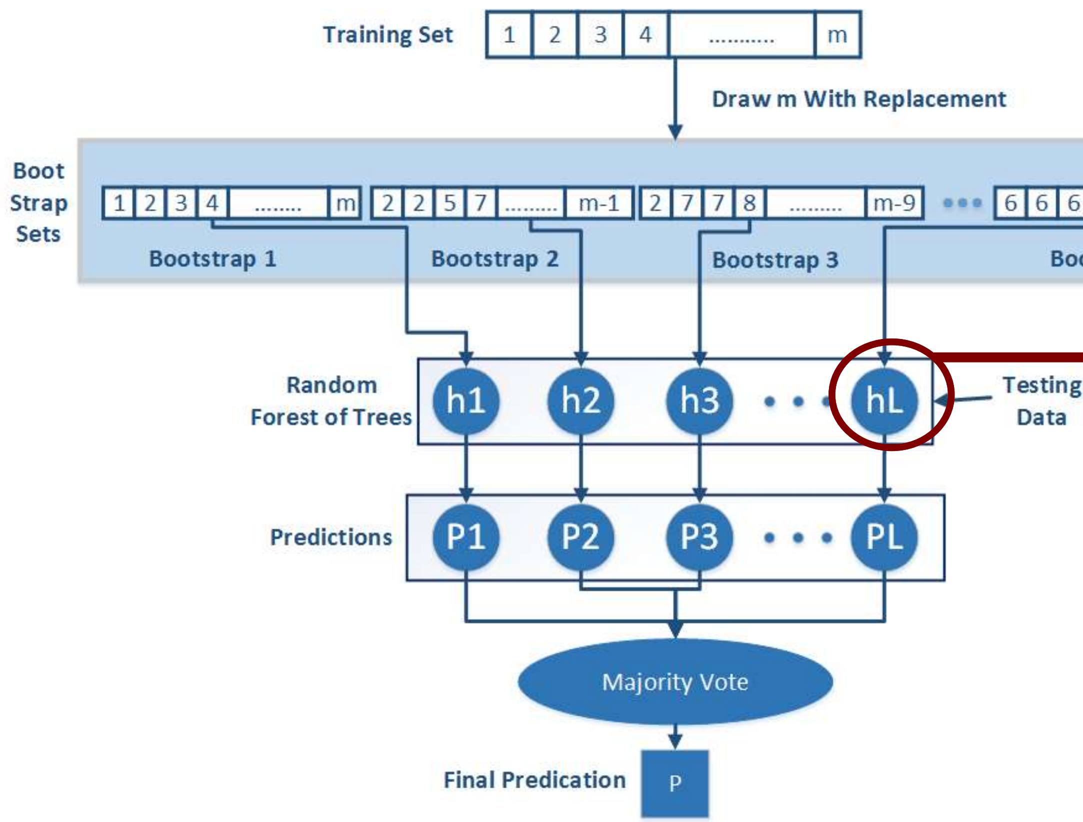
Powerful bagging approach

CUNY School of Professional Studies

Random Forest

- Machine learning ensemble algorithm
  - Combining multiple predictors
- Based on tree models
- Every tree is independent
- Subset of features are selected
- Suitable for both regression and classification
- Automatic variable selection
- Handles missing values
- Robust, improving model stability and accuracy

Random Forest



A Random Tree