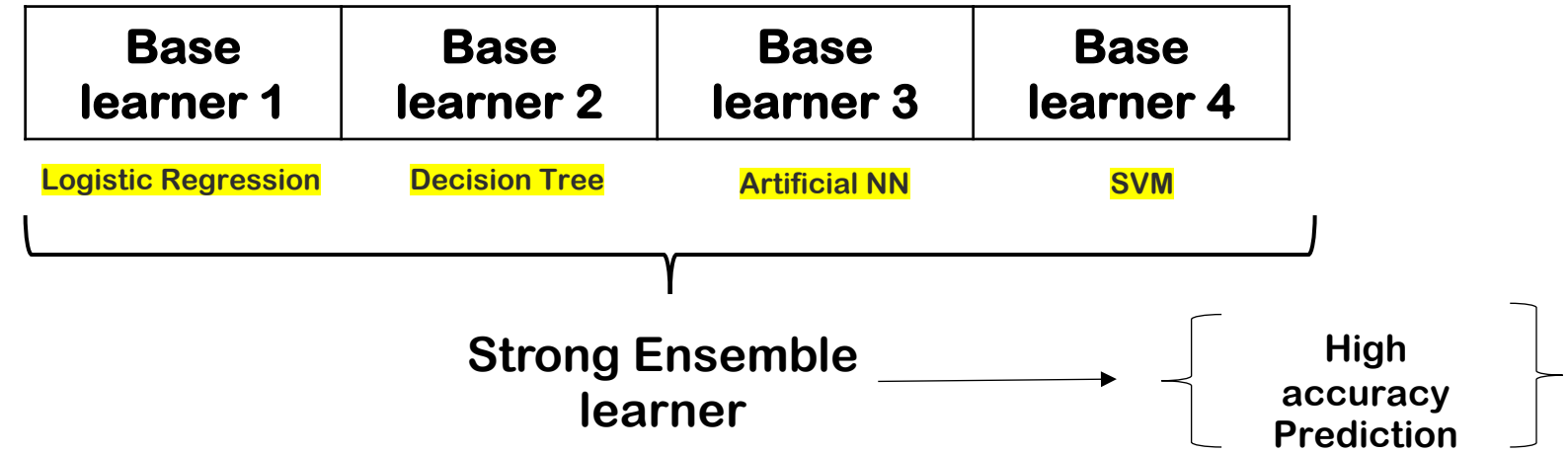
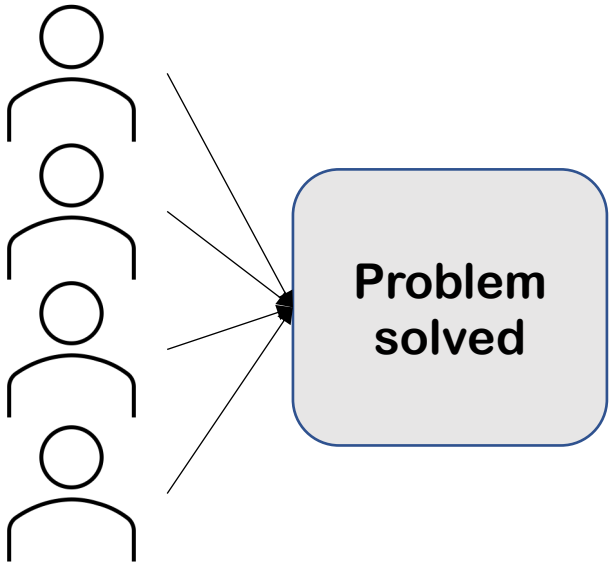
Machine Learning
CSCE 5215

Ensemble Learning

Instructor: Zeenat Tariq

Ensemble (collective) learning





Key idea of Ensemble
Committee of experts better than single expert

https://doi.org/10.1016/j.eswa.2010.06.048

Ensemble learning

- Most supervised learning approaches :
  - Select a single model
  - Find the best hyperparameters based on training data
- Ensemble learning approach:
  - Combine the predictions of many models varying by...
    - hyperparameter
    - structure of the model
    - resampling data features or instances
- Some ensemble methods use the same modeling approach (e.g. bagging) while others can be used to combine the results of wildly different models (e.g. boosting)
- However, both ensemble learning methods are “black box” methods

Ensemble learning

- Trains multiple learners at the same time (BASE LEARNERS)

- Purpose: Improved learning

- Base learners are generated from training data by a base learning algorithm e.g. decision tree, neural network etc.

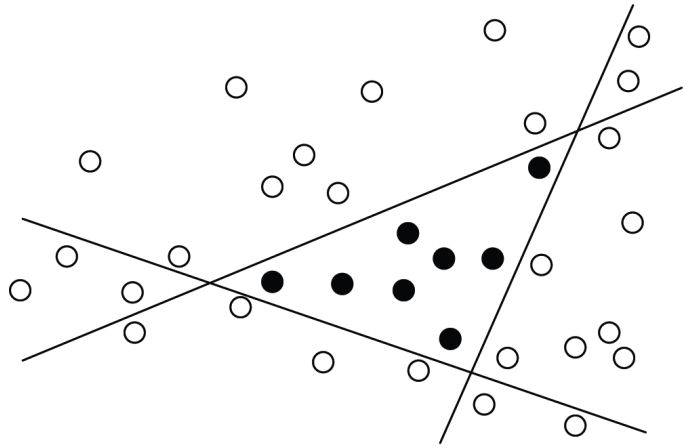
- **Steps** in an ensemble method

  1. Production of base learners:
     - Parallel style
     - Sequential style

  2. Combining the base learners using common combination schemes:
     - Majority Voting
     - Weighted Average

Motivation for ensemble learning

- To obtain smaller classification error
- A group of classifiers can learn a more complex decision boundary separating different classes than a single classifier (but also more prone to overfitting)- how do you avoid that? Don’t Overtrain your ensemble classifiers



Why ensemble improves prediction accuracy?

- Ensemble learners are unlikely to overfit
- Ensemble learners reduce the variance
- Training data might not provide sufficient information for choosing single best learner
- Search processes of individual / base learning algorithms might be imperfect (high variance)
- Hypothesis space being searched might not contain the target function

Applications of Ensemble learning

Optical character recognition (regression + SVM)

Text characterization (NB + SVM)

Face recognition (Nearest Neighbor + Random Subspace)

Credit scoring (Logistic regression + Decision trees)

Computer aided medical diagnosis

Gene expression analysis

Families of ensemble methods

Averaging methods

- -Bagging
- -Forests of randomized trees

Boosting methods

- -AdaBoost
- -Gradient Tree boosting

Common ensemble methods

- Bayes optimal classifier
  - Ensemble of hypotheses
  - Vote of each hypothesis is multiplied by probability of the hypothesis
- **Bootstrap Aggregating (Bagging)**
- **Boosting**
- Bucket of models
  - Model selection algorithm (choose best model for each problem)
  - Evaluate across many problems
- Stacking
  - Train all algorithms using available data
  - Combiner algorithm to make final prediction

How are weak learners generated in Bagging and Boosting?

A weak hypothesis or weak learner is defined as one whose performance is at least slightly better than random chance

BAGGING
Parallel production during training phase

- How is performance of the model improved?
  Train parallelly on multiple weak learners on a bootstrapped dataset (divide main data into bootstrap data)

BOOSTING
Sequential production during training phase

- How is performance of the model improved?
  Higher weight assigned to previously incorrectly classified sample (helps during voting or weighted average)

Bagging

(Bootstrap Aggregating)

Breiman, L. (1996). Bagging predictors. *Machine learning*, *24*(2), 123-140.

Bagging context

- Context: Bootstrap estimators in statistics are found by random resampling of the same data.
  - e.g. estimating a distribution of means from a sample directly by calculating the mean of multiple resamples with replacement
  - Contrast this with the standard parametric statistics approach: standard error = σ / sqrt(n)
- Bagging resamples the data in the training set
  - Either the features (“columns”) or the observations (“rows”) can be subsampled
  - This can be done with or without replacement - usually with
  - e.g. Random Forest picks subsamples of features to generate each tree
- All resampled estimators are weighted equally and the average is reported as the result

Bagging

(Bootstrap Aggregating)

Algorithm 1 Bagging

1: Let n be the number of bootstrap samples
2:
3: **for** i=1 to n **do**
4: Draw bootstrap sample of size m, D_i
5: Train base classifier h_i on D_i
6: $\hat{y} = mode\{h_1(\mathbf{x}), \ldots, h_n(\mathbf{x})\}$

Bootstrap Sampling

Bootstrap Sampling

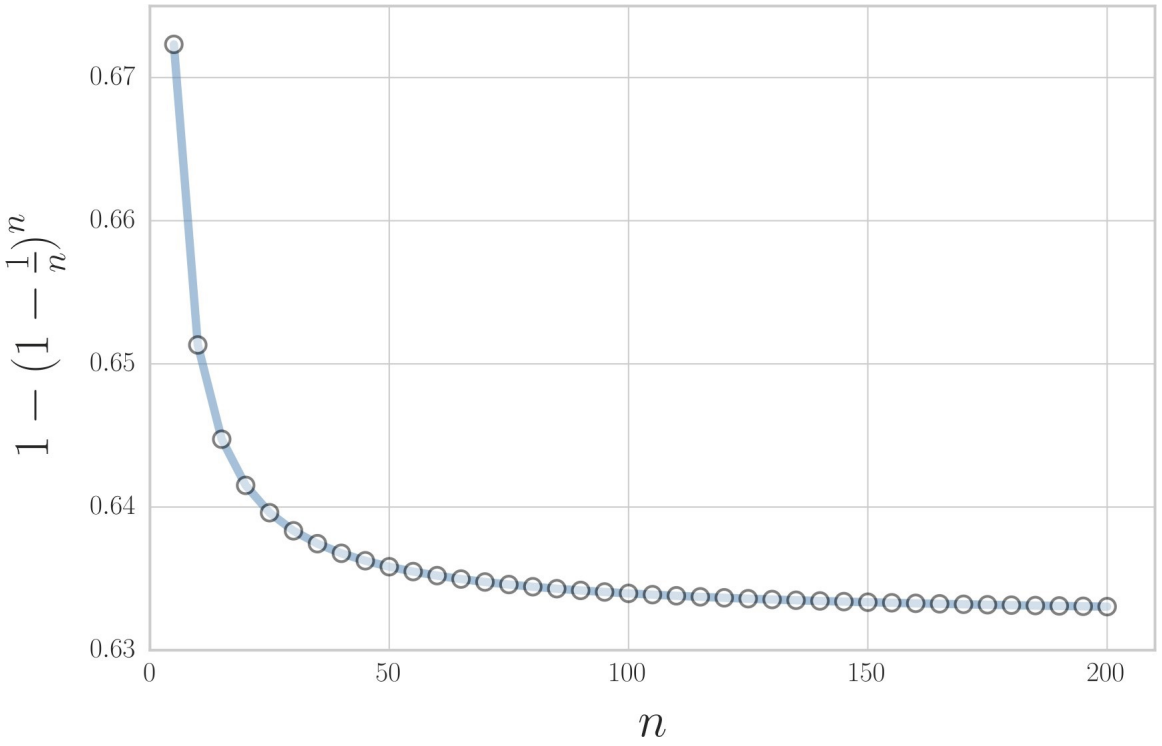
$$P(\text{not chosen}) = \left(1 - \frac{1}{m}\right)^{m},$$

$$\frac{1}{e} \approx 0.368, \quad m \to \infty .$$

$$P(\text{not chosen}) = \left(1 - \frac{1}{m}\right)^m,$$

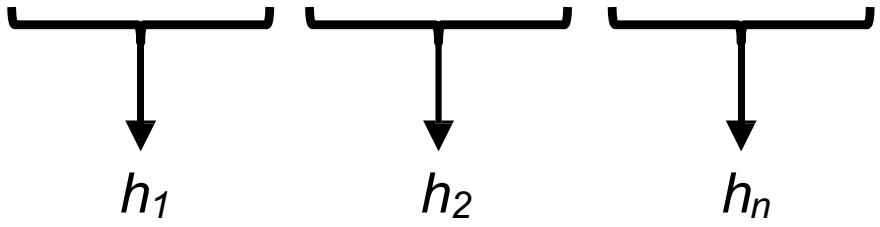
$$\frac{1}{e} \approx 0.368, \quad m \to \infty.$$

$$P(\text{chosen}) = 1 - \left(1 - \frac{1}{m}\right)^m \approx 0.632$$

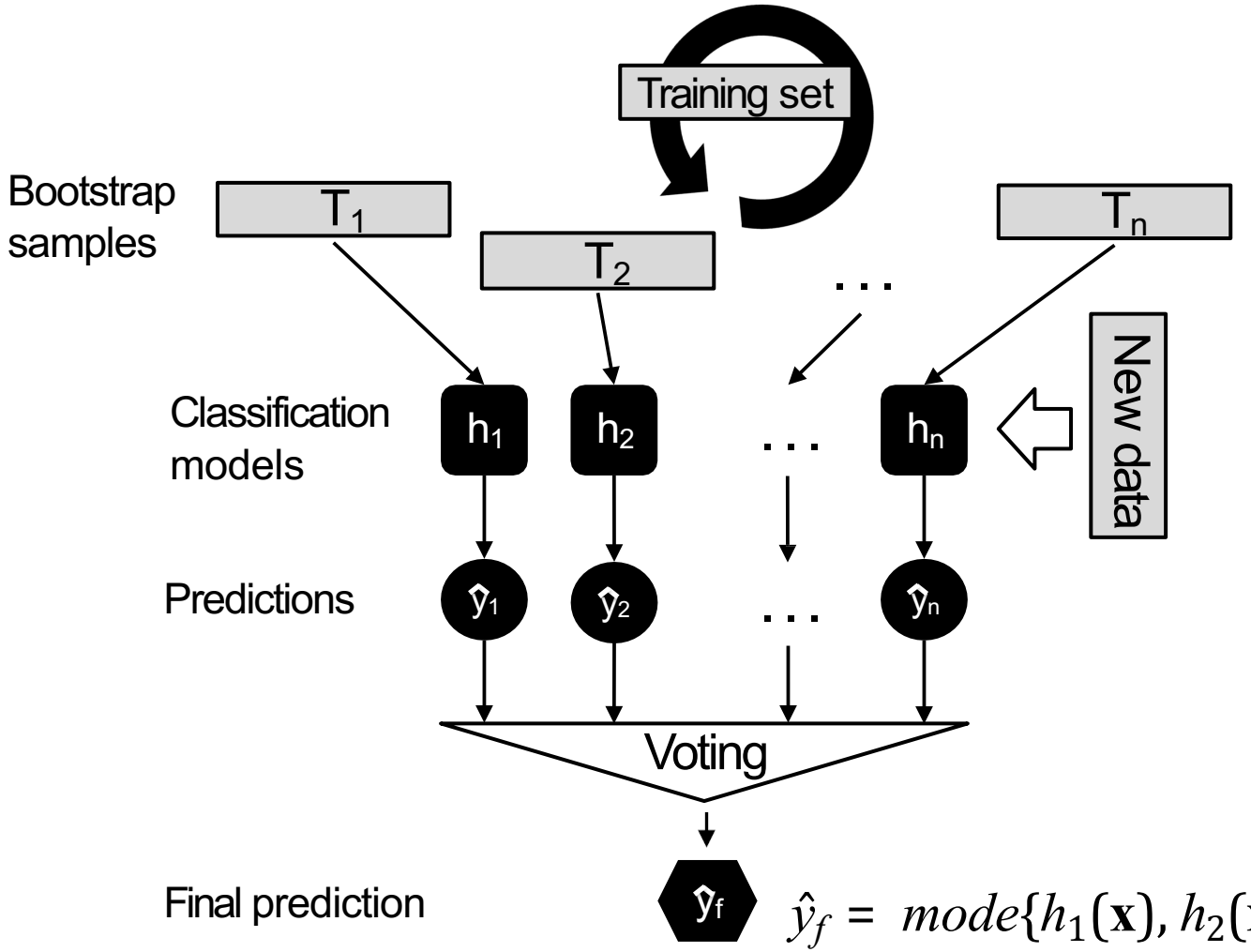


Bootstrap Sampling

Training example indices	Bagging round 1	Bagging round 2	…
1	2	7	…
2	2	3	…
3	1	2	…
4	3	1	…
5	7	1	…
6	2	7	…
7	4	7	…



Bagging Classifier

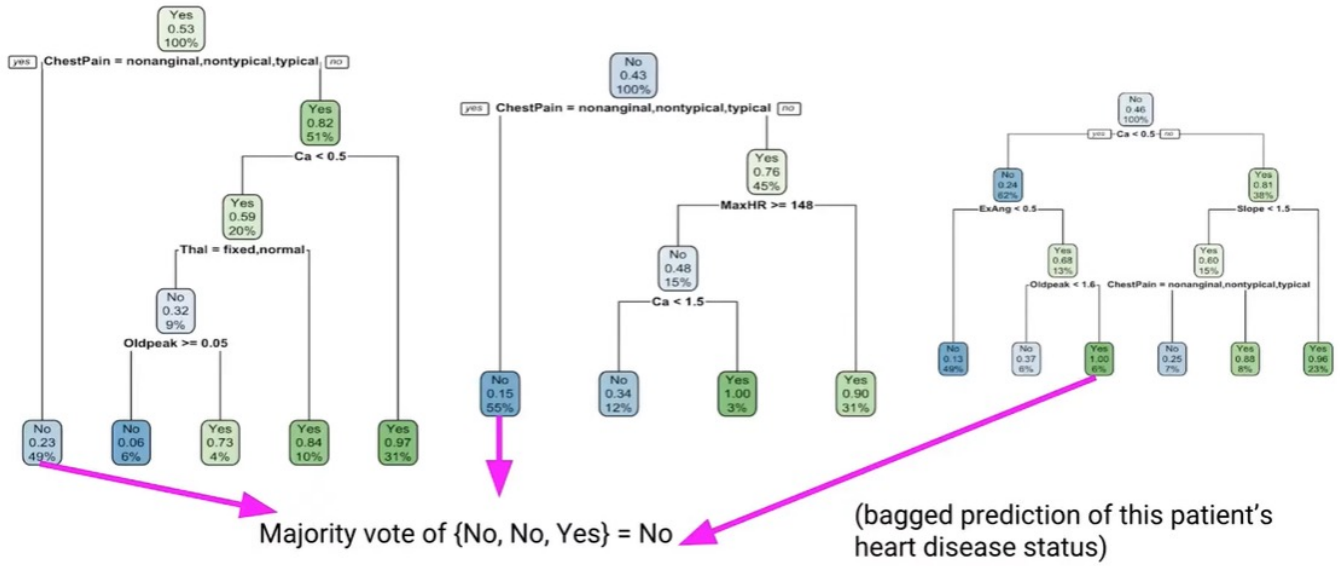


$$\hat{y}_f = mode\{h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots h_n(\mathbf{x})\}$$

where $h_i(\mathbf{x}) = \hat{y}_i$

BAGGING example

- Example dataset: Heart disease data
  Problem: Predict patient's heart disease status

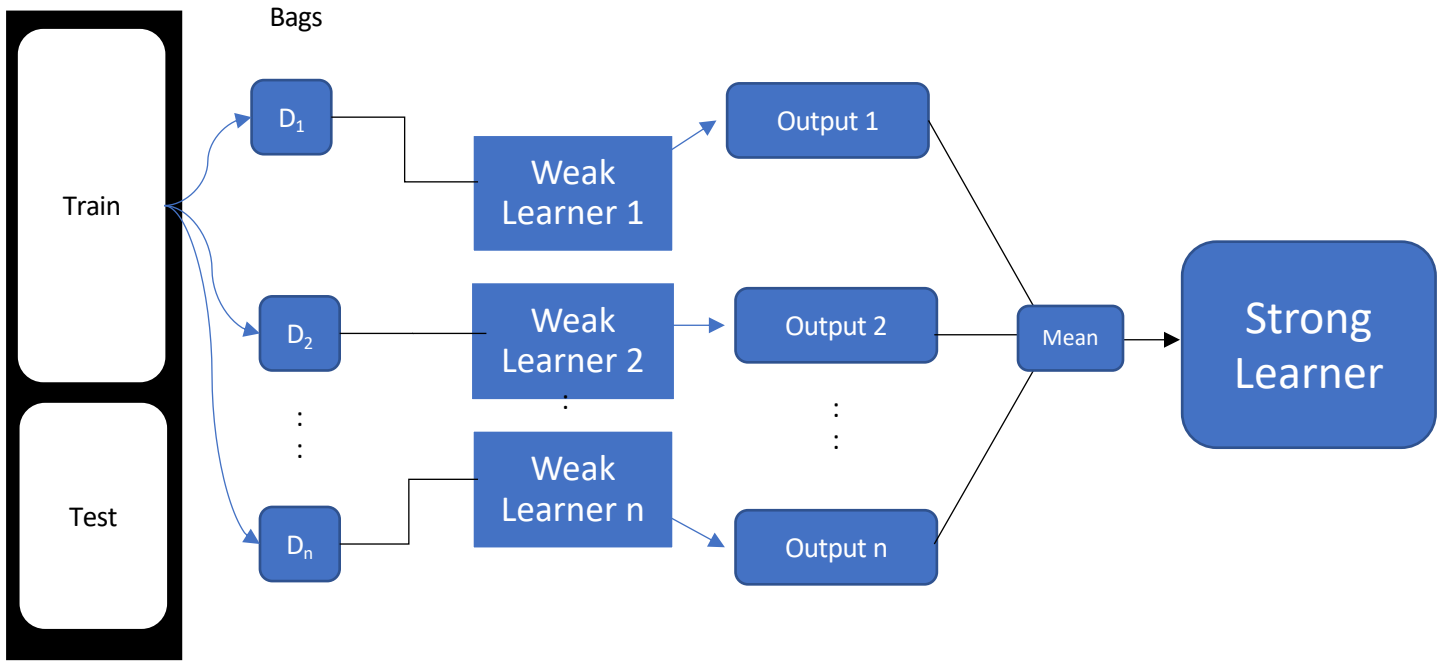


- 3 weak learners (decision trees)
  2 decision trees classify it as a NO, 1 classifies it as a YES

  Majority vote makes the final classification outcome as a NO

Additional Cardiovascular disease dataset:
https://www.sciencedirect.com/science/article/pii/S235291482100143X

Generalizing the BAGGING process



BAGGING summary

- Advantages
  - Improves generalization / helpful to avoid overfitting, as the generated models do not all contain the information of the full data set.
  - Best applied to models in danger of overfitting (e.g. decision trees without limitations)
  - Conceptually straightforward: “bootstrap aggregation” says it all
- Disadvantages
  - Requires learning over multiple resamples of training data
  - potentially costly unless learners are cheap and fast
    - (e.g. trees ---> hence, Random Forest’s popularity)

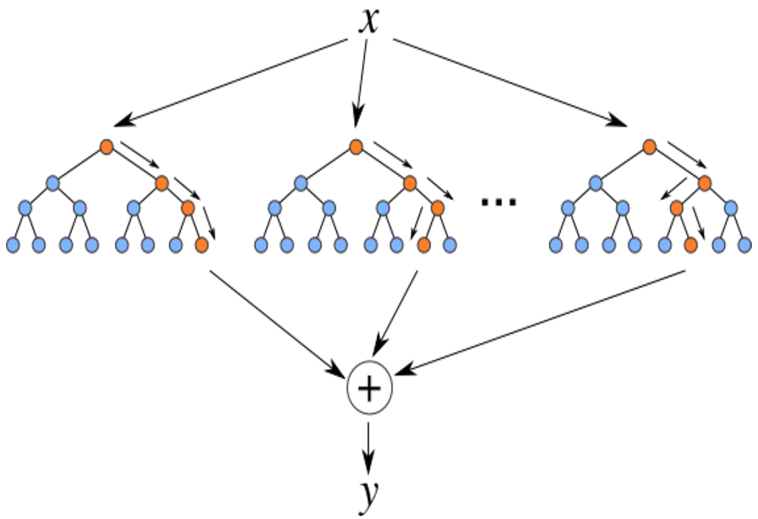
Random Forest

Decision tree learning is fast, but not robust...

Decision trees tend to overfit

Decision trees are created with random subsets of features

Combining these concepts produces a Random Forest classifier



Random Forest, pros and cons

- Robust behavior in many real-world situations
- Relatively fast for an ensemble method (since decision tree learning is fast)
- Works well when there is no strong relation among features
- More hyperparameters to potentially fit than decision trees alone
- Not easily interpretable

Boosting

Boosting

Both bagging and boosting often uses the same model type, but boosting can be used to combine multiple types of learning models.

It’s the way the 2nd and 3rd place groups can combine their efforts to beat #1 in competitions! even though their individual models may be vastly different.

From a theoretical point of view (given infinite samples) boosting is guaranteed to improve accuracy of a model by combining it with any other model which predicts better than chance.

Sounds great. How does it work?...

Boosting

Why is it used?

- Improved efficiency
- Improved accuracy
- Curbs over-fitting

What is Boosting?

Generating weak learners

Working principle of Boosting algorithm

Types of Boosting

- Adaptive Boosting

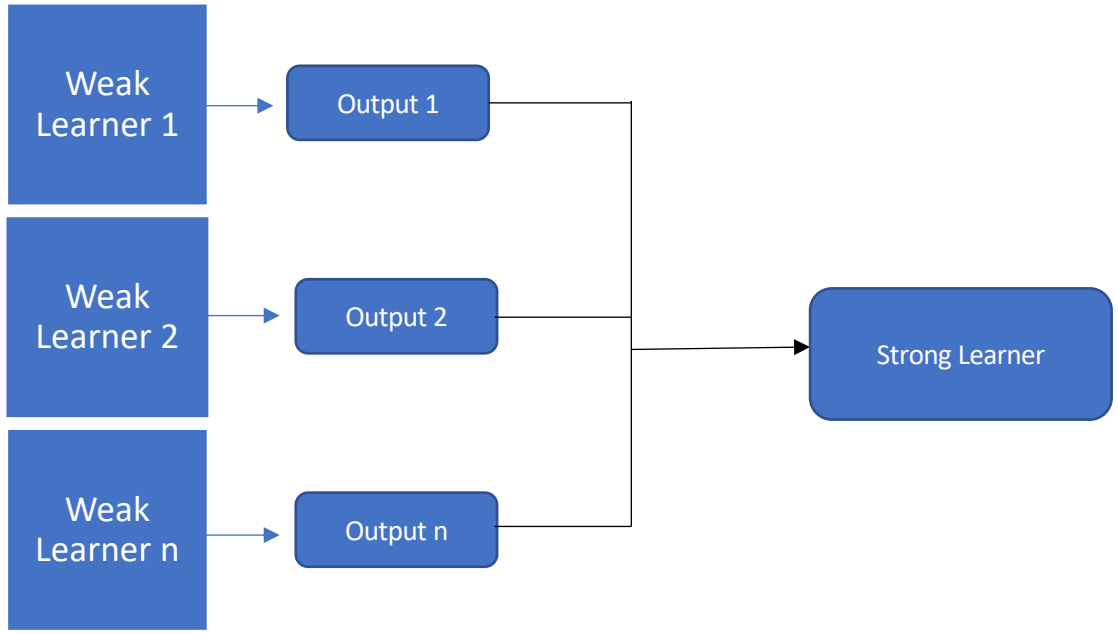
Boosting

- Example dataset: Images of spoons and forks
  Problem: Classify images into spoon class and fork class

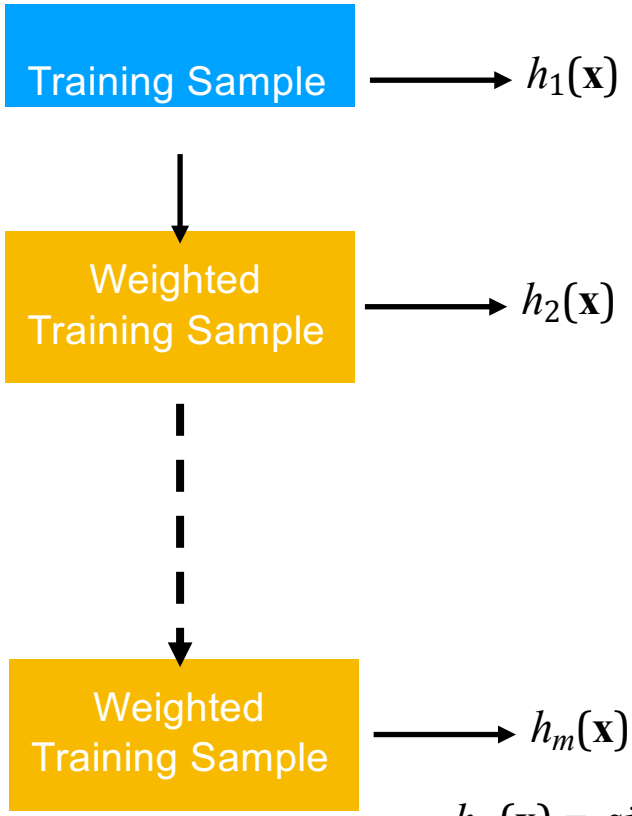
  Create rules for your model :
  Image has sharp edges ---------- > FORK
  Image has a round shape ---------- > SPOON
  Image has multiple branches ------ > FORK
  Image has an oval shape ---------- > SPOON
  Image has narrow tines ---------- > FORK

- 5 weak learners
  3 classify it as a FORK, 2 classify it as a SPOON
  Majority vote makes the final classification outcome as a FORK

Generalizing the Boosting process



General Boosting



$$h_m(\mathbf{x}) = sign\Big(\sum_{j=1}^{m} w_j\, h_j(\mathbf{x})\Big) \quad \text{for} \quad h(\mathbf{x}) \in \{-1,1\}$$

$$\text{or} \quad h_m(\mathbf{x}) = \arg\max_i \Big(\sum_{j=1}^{m} w_j\, \mathbf{1}[h_j(\mathbf{x}) = i]\Big) \quad \text{for} \quad h(\mathbf{x}) = i, \qquad i \in \{1,\ldots,n\}$$

Boosting workflow

- Generate multiple weak learners
- Combine their predictions to form one strong learner

Weak learner : base learning algorithms (Decision trees, neural networks)

Iteration 1:

Build a subset of data
Train the Decision tree (single level)
Check for false predictions (Test)

Iteration 2:

Assign higher weight to misclassified samples

Iteration 3:

Assign higher weight to misclassified samples

Iterate n times

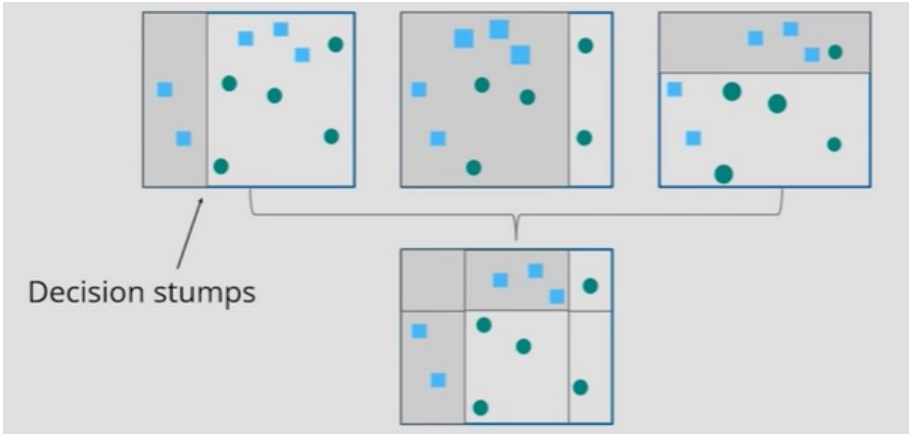
Types of Boosting

- Adaptive boosting (ADABOOST)
- Gradient boosting
- XG (extreme gradient) boosting
- Light GBM
- CatBoost

Adaptive Boosting algorithm

1. Assign equal weight to all samples
   Draw decision stump for single input feature
2. Results from 1st round are analyzed
   Misclassified samples assigned higher weight
3. Create new decision stump
   Repeat the process for all observations / data points

Very short decision tree that only has a single split is called decision stump.



AdaBoost workflow

- The AdaBoost algorithm receives as input **a training set of examples** S = (x1; y1); : : : ; (xm; ym), where for each i, yi = f(xi) for some labeling function f.
- The boosting process proceeds in a sequence of consecutive rounds.
- At round t, the booster first defines a distribution over the examples in S, denoted D(t).
- Then, the booster passes the distribution D(t) and the sample S to the weak learner.
- The weak learner is assumed to return a “weak“ hypothesis, h_t, whose error then, AdaBoost assigns a weight for h_t : i.e., the weight of h_t is inversely proportional to the error of h_t.
- **At the end of the round, AdaBoost updates the distribution so the examples on which h_t errors will get a higher probability mass while examples on which h_t is correct will get a lower probability mass.**
- Intuitively, this will force the weak learner to focus on the problematic examples in the next round
- The output of the AdaBoost algorithm is a “strong" classifier that is based on a **weighted sum of all the weak hypotheses**.

Gradient Boosting

Gradient boosting involves three elements:

1. A loss function to be optimized.
   E.g.: regression may use a **squared error**
   classification may use **logarithmic loss**

2. A weak learner to make predictions.

3. An additive model to add weak learners to minimize the loss function.
   E.g.: A gradient descent procedure is used to minimize the loss when adding trees.

Boosting Summary

Advantages

- Combines weak learners to create a strong learner – avoids overgeneralization
- Can be applied to many distinct supervised learning algorithms

Disadvantages

- Costly in terms of time as multiple learners cycle over the data many times
- Overfitting concern: with observations that are truly noise, their weight becomes substantial at the end of the algorithm

Summary

Ensemble methods allow multiple learners to be used for a classification, rather than just picking the best one

Bagging (Bootstrap aggregation) selects subsets of features and/or samples to create a “bag” of learners. This process is particularly effective at avoiding overfitting compared to the base learners alone

Boosting combines multiple learners by giving more effective learners greater weight. These weights are found by iterating over the data set and increasing the impact of the more “difficult” observations. This process is particularly effective at making weak (overgeneralizing) learning stronger.