Ensemble Learning
Ensemble Learning

- So far we have seen learning algorithms that take a training (data) set and output a classifier (or clustering solution)
- What if we want more accuracy than current algorithms afford?
  - Develop new learning algorithm
  - Improve existing algorithms
- Another approach is to leverage the algorithms we have via ensemble methods
  - Instead of calling an algorithm just once
  - Call algorithm multiple times and combine the multiple outputs
- Can be used in both supervised and unsupervised learning
  - We will focus on supervised ensemble learning here
Supervised Ensemble Learning

Traditional:

Training set \( \rightarrow \) Learner \( \rightarrow \) Classifier \( \leftrightarrow (x, ?) \)

\( y^* = h_1(x) \)

Ensemble method:

Training set \( \rightarrow \) Learner \( \rightarrow \) Classifier \( \leftrightarrow (x, ?) \)

\( h^* = F(h_1, h_2, \ldots, h_S) \)

\( y^* = h^*(x) \)

different training sets and/or learning algorithms
Why Ensemble Learning?

- **INTUITION**: Combining predictions of multiple classifiers (an ensemble) is more accurate than a single classifier.
- Justification:
  - easy to find quite good “rules of thumb” however **hard to find single highly accurate prediction rule**.
  - If the training set is small and the hypothesis space is large then there may be many equally accurate classifiers.
  - **Hypothesis space does not contain the true function**, but it has several good approximations.
  - **Exhaustive global search** in the hypothesis space is **expensive** so we can combine the predictions of several locally accurate classifiers.
How to generate ensemble?

• There are a variety of methods developed
• We will look at two of them:
  – Bagging
  – Boosting (Adaboost: adaptive boosting)
• Both of these methods takes a single learning algorithm (we will call it the base learner) and use it multiple times to generate multiple classifiers
Bagging: **Bootstrap Aggregation**
(Breiman, 1996)

Bagging carries out the following steps:

- **S** - Training data set.

1. Create $T$ bootstrap training sets $S_1, \ldots, S_T$ from $S$
   (see next slide for bootstrap procedure)

2. For each $i$ from 1 to $T$ $h_i = \text{Learn}(S_i)$.

3. Hypothesis: $H(x) = \text{majorityVote}(h_1(x), h_2(x), \ldots, h_T(x))$

Final hypothesis is just the majority vote of the ensemble members.
That is, return the class that gets the most votes.
Generate a Bootstrap sample of S

Given S, let $S' = \emptyset$
For $i=1, \ldots, N$ (the total number of points in S)
    draw a random point from S and add it to $S'$
End
Return $S'$

• **This procedure is called** sampling with replacement
  – Each time a point is drawn, it will not be removed
  – This means that we can have multiple copies of the same data point in my sample
  – size of $S'$= size of S
  – On average, 66.7% of the original points will appear in $S'$
Bagging Example

The true decision boundary
Decision Boundary by the CART Decision Tree Algorithm

Note that the decision tree has trouble representing this decision boundary
By averaging 100 trees, we achieve better approximation of the boundary, together with information regarding how confidence we are about our prediction.
Another Example

- Consider bagging with the linear perceptron base learning algorithm
- Each bootstrap training set will give a different linear separator
- Voting is similar to averaging them together (not equivalent)
Empirical Results for Bagging Decision Trees
(Freund & Schapire)

Why can bagging improve the classification accuracy?
The Concept of Bias and Variance

The circles represent the hypothesis space.

- **Target**
- **Bias**
- **Variance**
Bias/Variance for classifiers

• Bias arises when the classifier cannot represent the true function – that is, the classifier underfits the data
  – If the hypothesis space does not contain the target function, then we have bias

• Variance arises when classifiers learned on minor variations of data result in significantly different classifiers – variations cause the classifier to overfit differently
  – If the hypothesis space has only one function then the variance is zero (but the bias is huge)

• Clearly you would like to have a low bias and low variance classifier!
  – Typically, low bias classifiers (overfitting) have high variance
  – high bias classifiers (underfitting) have low variance
  – We have a trade-off
Effect of Algorithm Parameters on Bias and Variance

- $k$-nearest neighbor $k$: increasing $k$ typically increases bias and reduces variance

- Decision trees of depth $D$: increasing $D$ typically increases variance and reduces bias
Why does bagging work?

• Bagging takes the average of multiple models -- reduces the variance
• This suggests that bagging works the best with low bias and high variance classifiers such as ...
• Un-pruned decision trees
• Bagging typically will not hurt the performance