Ensemble Learning

CS 550: Machine Learning
Ensemble Learning

- **Problem:** Given M base learners \( \{L_1, L_2, ..., L_M\} \), find a combined (meta) learner with better performance
  - Very effective in many applications
  - Usually easy to implement

1. How to generate the base learners?
2. How to combine them?
How to Generate Base Learners?

- Ensemble techniques usually work well when base learners are “reasonably” accurate (but not too much) and diverse

- Base learners can be generated using
  - Different learning algorithms
  - Same algorithm with different parameters
  - Different representations of the same input
    - Sensor fusion at the data level, feature level or decision level
  - Different training sets
    - Bagging (samples are randomly drawn)
    - Boosting (samples are drawn to generate complementary learners)
How to Combine Base Learners?

- We combine the base learners after training them in parallel

\[
L_1 \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow Final\ decision
L_2
\ldots
L_M
\]

- We combine them while training them in serial

\[
L_1 \rightarrow \rightarrow \rightarrow \rightarrow \rightarrow Final\ decision
L_2
\ldots
L_M
\]
Voting

- Simplest ensemble method
- Suppose that learner \( L_j \) has prediction \( d_j \) with weight \( W_j \)

\[
\text{Final output } \quad y = \sum_{j=1}^{M} W_j d_j, \quad W_j \geq 0 \quad \text{and} \quad \sum_{j=1}^{M} W_j = 1
\]

- Simple voting (majority voting in classification) \( W_j = \frac{1}{M} \)

- Weighted voting
  - For example, use posteriors as weights and select the class for which \( y_i \) is the maximum 

\[
y_i = \sum_{j=1}^{M} P(C_i \mid x, L_j)
\]

- You can also consider the whole procedure as a Bayesian model

\[
y_i \equiv P(C_i \mid x) = \sum_{j=1}^{M} P(C_i \mid x, L_j) P(L_j)
\]
Bagging (Bootstrap AGgregating)

- Generate L base learners from the same training set \( \mathcal{D} \)
  - For each learner, use a separate training set that is generated by drawing N samples randomly from \( \mathcal{D} \) by replacement (each training set may have duplicate samples)
  - For a given new sample, combine the decisions of all learners (for example by simple voting)
Boosting

- Generate L base learners from the same training set $\mathcal{D}$
  - For the first classifier, generate a training set similar to bagging
  - Then, for the next classifier, generate a training set that more likely contains samples misclassified by the previous classifiers
  - The most famous boosting algorithm is called AdaBoost (by Freund and Schapire, 1996), which has many variants
AdaBoost – Training

There are $M$ learners, $L_j$, each of which will be trained on $D_j$ generated from the training set $D = \{x^t\}_{i=1}^{T}$

$p_j^t$: probability of selecting sample $x^t$ for the training set $D_j$ of the learner $L_j$

initialize $p_1^t = 1/T$

for $j = 1$ to $M$

construct $D_j$ by drawing $N$ samples from $D$ according to $p_j^t$

train $L_j$ on $D_j$

calculate the class of each sample using $L_j$ and compute the error rate $\epsilon_j$

if $\epsilon_j > 0.5$ stop

$\beta_j = \frac{\epsilon_j}{1-\epsilon_j}$ \hspace{1cm} ($\beta_j < 1$ when $\epsilon_j < 0.5$)

for each sample $x^t$

if $x^t$ is correctly classified

$p_{j+1}^t = \beta_j \cdot p_j^t$ \hspace{1cm} (decrease its probability)

else

$p_{j+1}^t = p_j^t$

normalize probabilities by $p_{j+1}^t = \frac{p_{j+1}^t}{\sum_u p_{j+1}^u}$
for a given $x$, calculate $P(C_i|x, L_j)$ using each classifier $L_j$
for each class $i$

$$y_i = \sum_{j=1}^{M} \log\left(\frac{1}{\beta_j}\right) \cdot P(C_i|x, L_j)$$

select the class with the maximum $y_i$
Random Forests

- Construct many classification trees (diversity is important) and combine their decisions (for example by voting)

- Each tree may be grown
  - Using a different training set (e.g., draw N samples from the original set with replacement)
  - Randomly selecting $k$ features out of $d$ features and considering only the splits on the selected features
  - Using a different training set without any pruning
Mixture of Experts

- Each base learner is considered as an expert
- There is a gating network that outputs the weight of each expert for a given sample $x$

Voting

$$y = \sum_{j=1}^{M} W_j d_j(x)$$

same for all instances

Mixture of experts

$$y = \sum_{j=1}^{M} W_j(x) d_j(x)$$
determined for each sample separately by the gating network

How do you learn the gating network?
Stacking

- There is a meta learner that learns the output of a sample from the outputs of the base learners (not directly from the inputs of the sample)

How do you learn the meta learner?
Arbiter Trees

- Base learners are trained on disjoint subsets of training data
- $D_{ij}$ can be formed
  1. Considering samples on which base classifiers disagree
  2. Item 1 + incorrectly classified samples
  3. Item 2 + some (or all) correctly classified samples

- To classify an unseen sample, one may
  - Use the arbiter if there exists disagreement
  - Combine its decision with those of the base learners
  - Use your own technique

Arbiter provides an alternative decision if base classifiers do not agree
Error-Correcting Output Codes

- Create many binary classifiers that distinguish one class from the others and then combine their decisions.

- After training binary classifiers, classify a sample with each of them and select the class whose coding is the most similar to the coding of the sample.
  - Sum of squared errors
  - Hamming distance

\[
W = \begin{bmatrix}
1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
\end{bmatrix} \quad K \text{ classes}
\]

\[M \text{ binary base learners}\]

Binary classification by the 5th learner gives label 0 for Class 1 and 5, gives label 1 for Class 2, 3, 4, and 6.
Error-Correcting Output Codes

- How to construct a codebook? ← IMPORTANT CHALLENGE
  - Could be set a priori
  - Could be formed in a random manner
  - Could be designed to optimize accuracy

\[ W = \begin{bmatrix}
1 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 \\
\end{bmatrix} \]

- \( K \) classes
- \( M \) binary base learners

Binary classification by the 5\(^{th}\) learner gives label 0 for Class 1 and 5.
gives label 1 for Class 2, 3, 4, and 6