Ensembles

• Definition
• Why do the work?
• How to build them
Ensembles

• Ensembles are groups of classifiers or regressors

• Given an ensemble $E=\{c_1,\ldots,c_n\}$ and a test example $x$, the output of an ensemble is given by:
  • Classification: the class among $[0,\ldots,v-1]$ with the most votes among the members of the ensemble
    • $E(x) = \text{argmax}_{t \in [0,\ldots,v-1]} \sum_{i=1}^{n} \text{int}(c_i(x) == t)$
  • Regression: the average prediction among the members of the ensemble
    • $E(x) = \frac{\sum_{i=1}^{n} c_i(x)}{n}$
Ensembles

Suppose

• We have a set of binary classifiers, each with an accuracy of 2/3
• The errors of any two classifiers are independent (two events are independent if $p(A \& B) = p(A)p(B)$).

Consider an ensemble of the 3 classifiers $E=\{c_1, c_2, c_3\}$, each of which has an accuracy of 2/3. What is the accuracy of the ensemble is calculated by simple voting?
Ensembles

Consider an ensemble of the 3 classifiers \( E=\{c_1, c_2, c_3\} \), each of which has an accuracy of \( \frac{2}{3} \). What is the accuracy of the ensemble is calculated by simple voting?

What is accuracy? The probability that a classifier will classify a randomly-chosen test example correctly.

Let \( x \) be a randomly chosen example

\[
p(c_1(x) == t(x)) = p(c_2(x) == t(x)) = p(c_3(x) == t(x)) = \frac{2}{3}
\]

where \( t(x) \) is the true class of \( x \)
Ensembles

Now we need to compute
\[ p(E(x) == t(x)) \]

For the ensemble to classify an example correctly, most of its members need to classify the example correctly. For our example, the ensemble will classify \( x \) correctly when 2 or 3 members classify \( x \) correctly

\[ p(E(x) == t(x)) = \]
\[ p(c_1(x) == t(x)) \cdot p(c_2(x) == t(x)) \cdot p(c_3(x) == t(x)) \quad (\text{all 3 are correct}) \]
\[ + p(c_1(x) == t(x)) \cdot p(c_2(x) == t(x)) \cdot p(c_3(x) != t(x)) \quad (c_1 \text{ and } c_2 \text{ are correct, } c_3 \text{ is incorrect}) \]
\[ + p(c_1(x) == t(x)) \cdot p(c_2(x) != t(x)) \cdot p(c_3(x) != t(x)) \quad (c_1 \text{ and } c_3 \text{ are correct, } c_2 \text{ is incorrect}) \]
\[ + p(c_1(x) != t(x)) \cdot p(c_2(x) == t(x)) \cdot p(c_3(x) != t(x)) \quad (c_2 \text{ and } c_3 \text{ are correct, } c_1 \text{ is incorrect}) \]
Ensembles

\[ p(E(x) == t(x)) = \]

\[ p(c_1(x) == t(x)) \cdot p(c_2(x) == t(x)) \cdot p(c_3(x) == t(x)) \]

\[ + p(c_1(x) == t(x)) \cdot p(c_2(x) == t(x)) \cdot p(c_3(x) != t(x)) \]

\[ + p(c_1(x) == t(x)) \cdot p(c_2(x) != t(x)) \cdot p(c_3(x) == t(x)) \]

\[ + p(c_1(x) != t(x)) \cdot p(c_2(x) == t(x)) \cdot p(c_3(x) == t(x)) \]

\[ = (2/3)^3 + 3(2/3)^2(1/3) = 8/27 + 4/9 = 20/27 = 0.74 \]

Accuracy increased from 0.67 to 0.74!
Ensembles

When do ensembles work?

- Each member of the ensemble is better than random guessing
- The errors made by individual members are not strongly correlated
Ensembles

Repeat the previous question but now the individual accuracies are $\frac{1}{2}$
Ensembles

Repeat the previous question but now the individual accuracies are $\frac{1}{3}$
Ensembles

Answer the original question, but now the ensemble members have identical outputs for any example.
Thus, for any given example
\[ p(c_1(x) == t(x)) = p(c_2(x) == t(x)) = p(c_3(x) == t(x)) \]
\[ p(c_1(x) != t(x)) \& p(c_2(x) != t(x)) = p(c_3(x) != t(x)) \]
\[ p(c_1(x) == t(x)) \& p(c_2(x) != t(x)) = 0 \]
How to build ensembles?

1. Use heterogeneous ensembles (same training data, different learning algorithms)
2. Manipulate training data (same learning algorithm, different training data)
3. Manipulate input features (use different subsets of the attribute sets)
4. Manipulate output targets (same data, same algorithm, convert multiclass problems into many two-class problems)
5. Inject randomness.
Random forests - Attribute selection

RandomForest(x, y)

1. For i = 1 to n:
   1. let a be a random boolean vector of size x.shape[1]
   2. x* = x[1:n, a]
   3. Ti = ID3(x*, y)

2. Return [T1, ..., Tn]
Random forests - Bagging

RandomForest(x,y)
1. For i = 1 to n:
   1. let x\* be a training set of the same size as x obtained by randomly sampling (with replacement) elements from x
   2. T\_i = ID3(x\*,y)
2. Return [T\_1, ..., T\_n]
Random forests - Boosting

RandomForest(x, y)
1. $p_i = 1/\text{len}(x)$
2. For $i = 1$ to $n$:
   1. Let $x^*$ be a training set of the same size as $x$ obtained by randomly sampling (with replacement) elements from $x$ using probability distribution $p$
   2. $T_i = \text{ID3}(x^*, y)$
   3. Classify $x$ using $T_i$. For every example $x_j$ in $x$, if $T_i$ classifies $x_j$ correctly, decrease $p_j$, otherwise increase it
3. Return $[T_1, ..., T_n]$
Random forests – Random Splits

Modify ID3 so that instead of choosing the best attribute it chooses randomly among the best m attributes or attribute,threshold combinations

RandomForest(x,y)

1. For i = 1 to n:
   1. \( T_i = \text{ID3\_random}(x,y) \)
2. Return \([T_1,\ldots,T_n]\)
Random forests - Gradient Boosting

(only for regression)

RandomForest(x,y)

1. For i = 1 to n:
   1. $T_i = \text{ID3}(x,y)$
   2. $p = T_i.\text{predict}(x)$
   3. $y = y - p$ (the prediction error)

2. Return $[T_1,\ldots,T_n]$

- The prediction of the ensemble is the sum (not the mean) of the predictions of individual members
Random forests

What works best?
In general, boosting, gradient boosting, and random splits work well.
The construction of forests with random splits is easier to parallelize, since it doesn’t require results from previous trees.
Ensembles

Why emphasis on decision/regression trees?

• Fast to build
• Easier to obtain uncorrelated outputs