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 $[0,..\mathrm{v}-1]$ with the most votes among the members of the ensemble
    - $\mathrm{E(x)} = \mathrm{argmax}\ (\mathrm{t\ in}[0,..\mathrm{v}-1]) \sum_{i=1}^{n} int(c_i(x) == t)$
  - Regression: the average prediction among the members of the ensemble
    - $\mathrm{E(x)} = \sum_{i=1}^{n} c_i(x)/n$

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 if the ensemble prediction 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 2/3. What is the accuracy of the ensemble if the ensemble prediction 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

$p(c_1(x) == t(x)) = p(c_2(x) == t(x)) = p(c_3(x) == t(x)) = 2/3$

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)) =

$p(c_1(x) == t(x))\ p(c_2(x) == t(x))\ p(c_3(x) == t(x))$ (all 3 are correct)

+ $p(c_1(x) == t(x))\ p(c_2(x) == t(x))\ p(c_3(x) != t(x))$ (c_1 and c_2 are correct, c_3 is incorrect)

+ $p(c_1(x) == t(x))\ p(c_2(x) != t(x))\ p(c_3(x) == t(x))$ (c_1 and c_3 are correct, c_2 is incorrect)

+ $p(c_1(x) != t(x))\ p(c_2(x) == t(x))\ p(c_3(x) == t(x))$ (c_2 and c_3 are correct, c_1 is incorrect)

Ensembles

$p(E(x) == t(x)) =$

$p(c_1(x) == t(x))\ p(c_2(x) == t(x))\ p(c_3(x) == t(x))$

$+\ p(c_1(x) == t(x))\ p(c_2(x) == t(x))\ p(c_3(x) != t(x))$

$+\ p(c_1(x) == t(x))\ p(c_2(x) != t(x))\ p(c_3(x) == t(x))$

$+\ p(c_1(x) != t(x))\ p(c_2(x) == t(x))\ p(c_3(x) == t(x))$

$=(2/3)^3 + 3(2/3)^2(1/3) = 8/27 + 4/9 = 20/27 = 0.741$

Accuracy increased from 0.667 to 0.741!

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 ½

$p(E(x) == t(x)) =$

$p(c_1(x) == t(x))\ p(c_2(x) == t(x))\ p(c_3(x) == t(x))$

$+\ p(c_1(x) == t(x))\ p(c_2(x) == t(x))\ p(c_3(x) != t(x))$

$+\ p(c_1(x) == t(x))\ p(c_2(x) != t(x))\ p(c_3(x) == t(x))$

$+\ p(c_1(x) != t(x))\ p(c_2(x) == t(x))\ p(c_3(x) == t(x))$

$=(1/2)^3 + 3(1/2)^2(1/2) = 1/8 + 3/8 = 4/8 = ½$

No gain!

Ensembles

Repeat the previous question but now the individual accuracies are 1/3

$p(E(x) == t(x)) =$

$p(c_1(x) == t(x))\ p(c_2(x) == t(x))\ p(c_3(x) == t(x))$

$+\ p(c_1(x) == t(x))\ p(c_2(x) == t(x))\ p(c_3(x) != t(x))$

$+\ p(c_1(x) == t(x))\ p(c_2(x) != t(x))\ p(c_3(x) == t(x))$

$+\ p(c_1(x) != t(x))\ p(c_2(x) == t(x))\ p(c_3(x) == t(x))$

$=(1/3)^3 + 3(1/3)^2(2/3) = 1/27 + 6/27 = 7/27 = 0.259$

Accuracy went down from 0.333 to 0.259

Ensembles

Answer the original question, but now the ensemble members have identical outputs for any example

Thus, for any given example

$p(c_1(x) == t(x) \;\&\; c_2(x) == t(x) \;\&\; c_3(x) == t(x)) = p(c_1(x) == t(x))$

$p(c_1(x) != t(x) \;\&\; c_2(x) != t(x)) = 0$ for i!=j

Therefore $p(E(x) == t(x)) = p(c_1(x) == t(x))$ (i.e. the ensemble's output is identical to the output of an individual classifier)

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.

Ensemble –Random Attribute Sets

Ensemble(X,y)

1. E =[]
2. For i = 1 to n:
   i. m = model() #Model can be built using any algorithm
   ii. let a be a random boolean vector of size X.shape[1]
   iii. m.fit(X[:, a]
   iv. E.append(m)
3. Return E

Ensemble –Random Example multisets (or bags)

Ensemble(X,y)

1. E =[]
2. For i = 1 to n:
   i. m = model() #Model can be built using any algorithm
   ii. let a be a random vector of integers of length X.shape[0] in the 0 to X.shape[1] range

       (*a = np.random.randint(low=0, high=X.shape[0], size=X.shape[0])*
   i. m.fit(X[a], y[a])
   ii. E.append(m)
3. Return E

Ensemble – Random example multisets with probabilistic selection (Boosting)

Ensemble(X,y)

1. E =[]
2. p = np.ones(X.shape[0]) – p[i] is the probability of choosing example X[i]
3. For i = 1 to n:
   i. m = model() #Model can be built using any algorithm
   ii. let a be a random vector of integers obtained probabilistically from p
   iii. m.fit(X[a], y[a])
   iv. E.append(m)
   v. pred = E.predict(X)
   vi. for i in range(len(X)):
      i. if pred[i] == y[i]: p[i] = p[i]/α - α>1 - Decrease the probability of choosing example X[i]
      ii. else: p[i] = p[i]*α - - Increase the probability of choosing example X[i]
   vii. p = p/np.sum(p) - Re-normalize p, so it is a probability
4. Return E

Ensemble – Algorithm Randomization

Normally used with decision and regression trees

Idea: Eliminate exhaustive search for best split at every node. Instead evaluate some random splits and pick the best

Implemented in sklearn as

- RandomForestClassifier
- RandomForestRegressor

Random forests - Gradient Boosting

(only for regression)

RandomForest(x,y)

1. For i = 1 to n:
    1. T_i = ID3(x,y)
    2. p = T_i.predict(x) .
    3. y = y – p (the prediction error)
2. Return [T_1,...,T_n]

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

Ensembles

What works best?

In general, boosting, gradient boosting, and random splits all 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