Ensemble methods

COMP 135 Intro to Machine Learning
Liping Liu

slides modified from Roni Khardon’s with permission

A motivating example

- Majority vote: Suppose we have 5 completely independent classifiers,
  - if accuracy is 70% for each, then
    majority vote accuracy is
    \[
    0.7^5 + 5 \times (0.7)^4 (0.3) + 10 
      \times (0.7)^3 (0.3)^2 = 83.7\%
    \]
  - With 101 such classifiers - 99.9% majority vote accuracy

Forcing Classifier Diversity

- Can we force the hypotheses produced by different runs to be different (even when base classifiers is not sensitive)?
  - Yes, several methods

Some General and Specialized Alg

- Bagging:
  - use bootstrap sample
  - Bagging of Decision Trees
- Random Forests
  - Bagging
  - Random subset of features at each node

Forcing Classifier Diversity

- Method 1: Bootstrapping: create different training sets by randomly selection with replacement
  - BAgging

<table>
<thead>
<tr>
<th>Algorithm:</th>
</tr>
</thead>
</table>
| For \( t = 1 \) to \( T \):
| Draw \( n \) items from \( X \) with replacement.
| Train a base learner \( f_t(x) \)
| The final classifier is:
| \[ f(x) = \left\lfloor \frac{1}{T} \sum_{t=1}^{T} f_t(x) \right\rfloor \] |
An example of regression

• A regression problem

The solid line is the ground-truth, Circles are noisy observations

• Prediction of a single tree

The solid black line is the ground-truth, The red line is the prediction of a single regression tree

• Prediction of 10 trees

The solid black line is the ground-truth, Red lines are predictions of a single regression tree

• Prediction of tree average

The solid black line is the ground-truth, The blue line is the prediction of the average of 10 regression trees

A classification example

• A classification problem

• Decision boundary of a single tree
• Average of predictions from 25 trees

Images are taken from Adele Cutler’s slides

• Decision boundary of bagging 25 trees

Images are taken from Adele Cutler’s slides

### Why it works

• Decrease error by decreasing the variance in the results of unstable classifiers
  - Unstable algorithm: when small change in the training set causes a large difference in the base learners.
  - Can be applied to arbitrary base classifiers

### Stability of Base Classifiers

• Which of these classifiers are stable/sensitive?
  - kNN
  - Decision Trees
  - Linear classifiers (SVM)

### Forcing Classifier Diversity

• Method 2: Random trees
  - Random Forest algorithm

### Random forest

• Train random trees
  - Fix the structure and depth
  - Randomly choose features for splits

Algorithm:

For \( t = 1 \) to \( T \) (# trees):
  Create an independent bootstrap sample from the training set.
  Train a random tree,
    for each node within a maximum depth:
      Randomly select \( m \) features from \( d \) features
      Find the best split on the selected \( m \) variables, create a tree node
  Average the trees to get predictions for new data.
Weak and Strong Learning

• Suppose we have a learning algorithm that often gives reasonable but not necessarily great performance (e.g., accuracy \( \geq 0.6 \)).

• Can we somehow use this algorithm to do better? How?

Tufts COMP 135
2018 Spring

Weak learners

• How weak can a weak learner be?
  - As long as the base learner has error rate less than 0.5.

Tufts COMP 135
2018 Spring

Example of AdaBoost

• A classification problem
  - 10 classes, not linearly separable
  - Uniform weight at initialization

Example of AdaBoost

• First iteration
  - First classifier: split on a single feature
  - 3 misclassified \( \Rightarrow \) error=0.3, beta=0.42

Example of AdaBoost

• Second iteration
  - 3 misclassified, error=0.21, beta=0.65

Tufts COMP 135
2018 Spring
Example of AdaBoost

• Second iteration
  - 3 misclassified, error=0.14, beta=0.92

Algorithm of AdaBoost

For \( t = 1 \) to \( T \)
Train a weak classifier \( h_t(x) \) using current weights \( w_i(t) \) for all \( i \), by minimizing the weighted classification error.

\[
\epsilon_t = \sum_i w_i(t) \times |y_i - h_t(x_i)|
\]

Compute contribution for this classifier
\[
\beta_t = \frac{1}{2} \log \frac{1 - \epsilon_t}{\epsilon_t}
\]

Update weights on training points,
\[
w_{i(t+1)} = w_{i(t)} \exp(-\beta_t y_i h_t(x_i))
\]
and normalize them such that \( \sum w_{i(t+1)} = 1 \)

Output the final classifier
\[
A(x) = \sum_t \beta_t \cdot h_t(x) > 0
\]

AdaBoost on a more realistic example

From p.340 of the ESL book. Find more experiment details there.

AdaBoost

• Base classifiers must be simple so they do not overfit.
• Can combine an arbitrary number of base learners
• Minimizes exponential loss

Gradient Tree Boosting

• The minimization problem at iteration \( t \):
  \[
  \min_{f_t} \text{Loss}(y, f_{t-1}(X) + f_t(X))
  \]
• \( f_t \) is a tree predicting real values
  - Decide the structure of the tree: gradient regression with a tree
  - Decide the node value for each tree node: minimize a scalar (easy for most losses)
  \[
  \min_{\gamma} \text{Loss}^\text{node}(y_{\text{node}}, f_{t-1}(X_{\text{node}}) + \gamma)
  \]
Gradient regression with a tree

- Let $\eta_t = f_t(X)$, then the negative gradient $-g_t$ represents the steepest direction $f_t(X)$ should go:
  
  $$g_t = \frac{d}{d \eta_t} \text{Loss}(y, f_{t-1}(X) + \eta_t)$$

- Fit the $t$-th regression tree on $g_t$

Gradient Tree Boosting

- Calculate gradient of scores for each data point
- Fit gradient values to get tree structure
- Assign values to tree nodes to minimize the loss
- Add up trees to minimize training loss

Gradient Tree Boosting: Shrinkage

- Controlling model complexity
  - Tree size $J$, typically $4 \leq J \leq 8$
  - Shrinkage $\nu$, typically $\nu \leq 0.1$
  - Subsampling, ratio $\eta$, typically $\eta \approx 0.5$
  - Number of iterations $M$

Algorithm 10.3: Gradient Tree Boosting Algorithm

1. Initialize $S_0 = \arg\min \sum_x L(y, g)$
2. For $m = 1$ to $M$
   a. For $i = 1, 2, \ldots, N$ compute:
      $$r_{im} = -\left(\frac{\partial L(y, f_{im-1})}{\partial f_{im-1}}\right)_{f_{im-1}}$$
   b. Fit a regression tree to the target $r_{im}$ giving terminal region $R_j$, $j = 1, 2, \ldots, J_m$
   c. For $j = 1, 2, \ldots, J_m$ compute:
      $$y_{jm} = \arg\min \sum_{x \in R_j} L(y, f_{xjm} + \gamma_j)$$
   d. Update $f_m(x) = f_{m-1}(x) + \sum_{m \in R_j} y_{jm} \gamma_j$
3. Output $f_M(x) = f_M(x)$

Check section 10.10 of ESML book for details
Other ensemble methods

- Stacking
  - Use classifier outputs as input of a super classifier
- Cascading
  - Postpone unsure instances to next classifier

Ensemble Methods

- Voting among diverse set of hypotheses can help reduce errors
- Loss minimization using a function form of trees
- Bagging, Random Forests, Ada-Boost, Gradient Tree Boosting
- Many variants exist
- Other ways of combining classifiers are also possible