Chapter 3: Combining Classifiers

From “Web Data Mining”, by Bing Liu (UIC), Springer Verlag, 2007
Outline

- Ensemble methods: Bagging and Boosting
- Fully supervised learning (traditional classification)
- Partially (semi-) supervised learning (or classification)
  - Learning with a small set of labeled examples and a large set of unlabeled examples (LU learning)
Combining classifiers

- So far, we have only discussed individual classifiers, i.e., how to build them and use them.
- Can we combine multiple classifiers to produce a better classifier?
- Yes, sometimes
- We discuss two main algorithms:
  - Bagging
  - Boosting
Bagging

- Breiman, 1996

- **Bootstrap Aggregating** = Bagging
  - Application of bootstrap sampling
    - Given: set $D$ containing $m$ training examples
    - Create a sample $S[i]$ of $D$ by drawing $m$ examples at random with replacement from $D$
    - $S[i]$ of size $m$: expected to leave out 0.37 of examples from $D$
Bagging (cont…)

- **Training**
  - Build a distinct classifier on each $S[i]$ to produce $k$ classifiers, using the same learning algorithm.

- **Testing**
  - Classify each new instance by voting of the $k$ classifiers (equal weights)
Bagging Example

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Training set 1</td>
<td>2</td>
<td>7</td>
<td>8</td>
<td>3</td>
<td>7</td>
<td>6</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Training set 2</td>
<td>7</td>
<td>8</td>
<td>5</td>
<td>6</td>
<td>4</td>
<td>2</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>Training set 3</td>
<td>3</td>
<td>6</td>
<td>2</td>
<td>7</td>
<td>5</td>
<td>6</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Training set 4</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>3</td>
<td>8</td>
</tr>
</tbody>
</table>
Bagging (cont …)

- **When does it help?**
  - **When learner is unstable**
    - Small change to training set causes large change in the output classifier
    - True for decision trees, neural networks; not true for $k$-nearest neighbor, naïve Bayesian, class association rules
  - Experimentally, bagging can help substantially for unstable learners, may somewhat degrade results for stable learners
Boosting

- A family of methods:
  - We only study AdaBoost (Freund & Schapire, 1996)

- Training
  - Produce a sequence of classifiers (the same base learner)
  - Each classifier is dependent on the previous one, and focuses on the previous one’s errors
  - Examples that are incorrectly predicted in previous classifiers are given higher weights

- Testing
  - For a test case, the results of the series of classifiers are combined to determine the final class of the test case.
AdaBoost

Weighted training set

\[(x_1, y_1, w_1)\]
\[(x_2, y_2, w_2)\]
\[\ldots\]
\[(x_n, y_n, w_n)\]

Non-negative weights sum to 1

Build a classifier \( h_t \) whose accuracy on training set > \( \frac{1}{2} \) (better than random)

called a weaker classifier

Change weights
AdaBoost algorithm

Algorithm AdaBoost.M1

Input: sequence of $m$ examples $\{(x_1, y_1), \ldots, (x_m, y_m)\}$
with labels $y_i \in Y = \{1, \ldots, k\}$
weak learning algorithm WeakLearn
integer $T$ specifying number of iterations

Initialize $D_1(x_i) = 1/m$ for all $i$.

Do for $t = 1, 2, \ldots, T$:

1. Call WeakLearn, providing it with the distribution $D_t$.
2. Get back a hypothesis $h_t : X \rightarrow Y$.
3. Calculate the error of $h_t$: $\epsilon_t = \sum_{x_i \in \{x_i \mid \epsilon_t(x_i) \neq y_i\}} D_t(x_i)$.

If $\epsilon_t > 1/2$, then set $T = t - 1$ and abort loop.
4. Set $\beta_t = \epsilon_t / (1 - \epsilon_t)$.
5. Update distribution $D_t$:

$$D_{t+1}(x_i) = \frac{D_t(x_i)}{Z_t} \times \begin{cases} \beta_t & \text{if } h_t(x_i) = y_i \\ 1 & \text{otherwise} \end{cases}$$

where $Z_t$ is a normalization constant (chosen so that $D_{t+1}$ will be a distribution).

Output the final hypothesis:

$$h_{\text{final}}(x) = \arg \max_{y \in Y} \sum_{x_i \in \{x_i \mid h_t(x_i) = y \}} \log \frac{1}{\beta_t}.$$
### Bagging, Boosting and C4.5

C4.5's mean error rate over the 10 cross-validation.

#### Bagged C4.5 vs. C4.5.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>C4.5 Err (%)</th>
<th>Bagged C4.5 Err (%)</th>
<th>Ratio</th>
<th>Boosted C4.5 Err (%)</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>anneal</td>
<td>7.67</td>
<td>6.25</td>
<td>.914</td>
<td>4.73</td>
<td>.617</td>
</tr>
<tr>
<td>audiology</td>
<td>22.12</td>
<td>19.29</td>
<td>.872</td>
<td>15.71</td>
<td>.710</td>
</tr>
<tr>
<td>auto</td>
<td>17.66</td>
<td>19.66</td>
<td>1.113</td>
<td>15.22</td>
<td>.862</td>
</tr>
<tr>
<td>breast-w</td>
<td>5.28</td>
<td>4.23</td>
<td>.802</td>
<td>4.09</td>
<td>.775</td>
</tr>
<tr>
<td>chess</td>
<td>8.55</td>
<td>8.33</td>
<td>.975</td>
<td>4.59</td>
<td>.537</td>
</tr>
<tr>
<td>colic</td>
<td>14.92</td>
<td>15.19</td>
<td>1.018</td>
<td>18.83</td>
<td>1.262</td>
</tr>
<tr>
<td>credit-a</td>
<td>14.70</td>
<td>14.13</td>
<td>.962</td>
<td>15.64</td>
<td>1.064</td>
</tr>
<tr>
<td>credit-g</td>
<td>28.44</td>
<td>25.81</td>
<td>.908</td>
<td>29.14</td>
<td>1.025</td>
</tr>
<tr>
<td>diabetes</td>
<td>25.39</td>
<td>23.63</td>
<td>.931</td>
<td>28.18</td>
<td>1.110</td>
</tr>
<tr>
<td>glass</td>
<td>32.48</td>
<td>27.01</td>
<td>.832</td>
<td>23.55</td>
<td>.725</td>
</tr>
<tr>
<td>heart-c</td>
<td>22.94</td>
<td>21.52</td>
<td>.938</td>
<td>21.39</td>
<td>.932</td>
</tr>
<tr>
<td>heart-h</td>
<td>21.53</td>
<td>20.31</td>
<td>.943</td>
<td>21.05</td>
<td>.978</td>
</tr>
<tr>
<td>hepatitis</td>
<td>20.39</td>
<td>18.52</td>
<td>.908</td>
<td>17.68</td>
<td>.867</td>
</tr>
<tr>
<td>hypo</td>
<td>.48</td>
<td>.45</td>
<td>.928</td>
<td>.36</td>
<td>.746</td>
</tr>
<tr>
<td>iris</td>
<td>4.80</td>
<td>5.13</td>
<td>1.069</td>
<td>6.53</td>
<td>1.361</td>
</tr>
<tr>
<td>labor</td>
<td>19.12</td>
<td>14.39</td>
<td>.752</td>
<td>13.86</td>
<td>.725</td>
</tr>
<tr>
<td>letter</td>
<td>11.99</td>
<td>7.51</td>
<td>.626</td>
<td>4.66</td>
<td>.389</td>
</tr>
<tr>
<td>lymphography</td>
<td>21.69</td>
<td>20.41</td>
<td>.941</td>
<td>17.43</td>
<td>.804</td>
</tr>
<tr>
<td>phoneme</td>
<td>19.44</td>
<td>18.73</td>
<td>.964</td>
<td>16.36</td>
<td>.842</td>
</tr>
<tr>
<td>segment</td>
<td>3.21</td>
<td>2.74</td>
<td>.853</td>
<td>1.87</td>
<td>.583</td>
</tr>
<tr>
<td>sick</td>
<td>1.34</td>
<td>1.22</td>
<td>.907</td>
<td>1.05</td>
<td>.781</td>
</tr>
<tr>
<td>sonar</td>
<td>25.62</td>
<td>23.80</td>
<td>.929</td>
<td>19.62</td>
<td>.766</td>
</tr>
<tr>
<td>soybean</td>
<td>7.73</td>
<td>7.58</td>
<td>.981</td>
<td>7.16</td>
<td>.926</td>
</tr>
<tr>
<td>splice</td>
<td>5.91</td>
<td>5.58</td>
<td>.943</td>
<td>5.43</td>
<td>.919</td>
</tr>
<tr>
<td>vehicle</td>
<td>27.09</td>
<td>25.54</td>
<td>.943</td>
<td>22.72</td>
<td>.839</td>
</tr>
<tr>
<td>vote</td>
<td>5.06</td>
<td>4.37</td>
<td>.864</td>
<td>5.29</td>
<td>1.046</td>
</tr>
<tr>
<td>waveform</td>
<td>27.33</td>
<td>19.77</td>
<td>.723</td>
<td>18.53</td>
<td>.678</td>
</tr>
</tbody>
</table>

**(Average)**

|                  | 15.66 | 14.11 | .905 | 13.35 | .847 | .939 |

#### Boosted C4.5 vs. C4.5.`

#### Boosting vs. Bagging
Does AdaBoost always work?

- The actual performance of boosting depends on the data and the base learner.
  - It requires the base learner to be unstable as bagging.
- Boosting seems to be susceptible to noise.
  - When the number of outliers is very large, the emphasis placed on the hard examples can hurt the performance.
C4.5 and Boosting
Boosting over Reuters

Source: A Short Introduction to Boosting, (Freund & Schapire, 99)
Chapter 5: Partially-Supervised Learning
Learning from a small labeled set and a large unlabeled set

LU learning
Unlabeled Data

- One of the bottlenecks of classification is the labeling of a large set of examples (data records or text documents).
  - Often done manually
  - Time consuming

- Can we label only a small number of examples and make use of a large number of unlabeled examples to learn?
- Possible in many cases.
Why unlabeled data are useful?

- Unlabeled data are usually plentiful, labeled data are expensive.
- Unlabeled data provide information about the joint probability distribution over words and collocations (in texts).
- We will use text classification to study this problem.
Documents containing “homework” tend to belong to the positive class

Labeled Data

DocNo: k ClassLabel: Positive
......
......homework....
...

DocNo: m ClassLabel: Positive
......
......homework....
...

DocNo: n ClassLabel: Positive
......
......homework....
...

Unlabeled Data

DocNo: x (ClassLabel: Positive)
......
......homework....
...lecture....

DocNo: y (ClassLabel: Positive)
......lecture.....
......homework....
...

DocNo: z ClassLabel: Positive
......
......homework....
......lecture....
How to use unlabeled data

- One way is to use the EM algorithm
  - EM: Expectation Maximization

- The EM algorithm is a popular iterative algorithm for maximum likelihood estimation in problems with missing data.

- The EM algorithm consists of two steps,
  - *Expectation step*, i.e., filling in the missing data
  - *Maximization step* – calculate a new maximum *a posteriori* estimate for the parameters.
Incorporating unlabeled Data with EM
(Nigam et al, 2000)

- Basic EM
- Augmented EM with weighted unlabeled data
- Augmented EM with multiple mixture components per class
Algorithm Outline

1. Train a classifier with only the labeled documents.
2. Use it to probabilistically classify the unlabeled documents.
3. Use ALL the documents to train a new classifier.
4. Iterate steps 2 and 3 to convergence.
Basic Algorithm

**Algorithm** EM($L$, $U$)

1. Learn an initial naïve Bayesian classifier $f$ from only the labeled set $L$ (using Equations (27) and (28) in Chap. 3);

2. repeat
   // E-Step
   3. for each example $d_i$ in $U$ do
   4. Using the current classifier $f$ to compute $\Pr(c_j|d_i)$ (using Equation (29) in Chap. 3).

5. end
   // M-Step

6. learn a new naïve Bayesian classifier $f$ from $L \cup U$ by computing $\Pr(c_j)$ and $\Pr(w_i|c_j)$ (using Equations (27) and (28) in Chap. 3).

7. until the classifier parameters stabilize

Return the classifier $f$ from the last iteration.

**Fig. 5.1.** The EM algorithm with naïve Bayesian classification
Basic EM: E Step & M Step

E Step:

$$
\Pr(c_j \mid d_i; \Theta) = \frac{\Pr(c_j \mid \Theta) \Pr(d_i \mid c_j; \Theta)}{\Pr(d_i \mid \Theta)}
\frac{\Pr(c_j \mid \Theta) \prod_{k=1}^{d_i} \Pr(w_{d_i,k} \mid c_j; \Theta)}{\sum_{r=1}^{\vert C \vert} \Pr(c_r \mid \Theta) \prod_{k=1}^{d_i} \Pr(w_{d_i,k} \mid c_r; \Theta)},
$$

M Step:

$$
\Pr(w_t \mid c_j; \Theta) = \frac{\lambda + \sum_{i=1}^{\vert D \vert} N_{ti} \Pr(c_j \mid d_i)}{\lambda \vert V \vert + \sum_{s=1}^{\vert V \vert} \sum_{i=1}^{\vert D \vert} N_{si} \Pr(c_j \mid d_i)}.
$$

$$
\Pr(c_j \mid \Theta) = \frac{\sum_{i=1}^{\vert D \vert} \Pr(c_j \mid d_i)}{\vert D \vert}.
$$
The problem

- It has been shown that the EM algorithm in Fig. 5.1 works well if the
  - The two mixture model assumptions for a particular data set are true.
- The two mixture model assumptions, however, can cause major problems when they do not hold. In many real-life situations, they may be violated.
- It is often the case that a class (or topic) contains a number of sub-classes (or sub-topics).
  - For example, the class Sports may contain documents about different sub-classes of sports, Baseball, Basketball, Tennis, and Softball.
- Some methods to deal with the problem.
Weighting the influence of unlabeled examples by factor $\mu$

New M step:

\[
\Pr(w_i \mid c_j) = \frac{\lambda + \sum_{i=1}^{\lvert D \rvert} \Lambda(i) N_{ti} \Pr(c_j \mid d_i)}{\lambda \cdot V + \sum_{s=1}^{\lvert V \rvert} \sum_{i=1}^{\lvert D \rvert} \Lambda(i) N_{ti} \Pr(c_j \mid d_i)},
\]

where

\[
\Lambda(i) = \begin{cases} 
\mu & \text{if } d_i \in U \\
1 & \text{if } d_i \in L.
\end{cases}
\]

The prior probability also needs to be weighted.
Experimental Evaluation

- Newsgroup postings
  - 20 newsgroups, 1000/group

- Web page classification
  - student, faculty, course, project
  - 4199 web pages

- Reuters newswire articles
  - 12,902 articles
  - 10 main topic categories
20 Newsgroups
20 Newsgroups

The diagram shows the accuracy of classification systems as a function of the number of unlabeled documents. The x-axis represents the number of unlabeled documents, while the y-axis represents the accuracy. Different lines correspond to different numbers of labeled documents:

- 3000 labeled documents
- 600 labeled documents
- 300 labeled documents
- 140 labeled documents
- 40 labeled documents

As the number of unlabeled documents increases, the accuracy generally increases for all scenarios, indicating the potential benefit of incorporating unlabeled data into the training process.
Another approach: Co-training

- Again, learning with a small labeled set and a large unlabeled set.
- The attributes describing each example or instance can be partitioned into two subsets. Each of them is sufficient for learning the target function.
  - E.g., hyperlinks and page contents in Web page classification.
- Two classifiers can be learned from the same data.
Co-training Algorithm
[Blum and Mitchell, 1998]

Given: labeled data $L$, unlabeled data $U$

Loop:

Train $h_1$ (e.g., hyperlink classifier) using $L$
Train $h_2$ (e.g., page classifier) using $L$
Allow $h_1$ to label $p$ positive, $n$ negative examples from $U$
Allow $h_2$ to label $p$ positive, $n$ negative examples from $U$
Add these most confident self-labeled examples to $L$
Co-training: Experimental Results

- begin with 12 labeled web pages (academic course)
- provide 1,000 additional unlabeled web pages
- average error: learning from labeled data 11.1%
- average error: co-training 5.0%

<table>
<thead>
<tr>
<th></th>
<th>Page-base classifier</th>
<th>Link-based classifier</th>
<th>Combined classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised training</td>
<td>12.9</td>
<td>12.4</td>
<td>11.1</td>
</tr>
<tr>
<td>Co-training</td>
<td>6.2</td>
<td>11.6</td>
<td>5.0</td>
</tr>
</tbody>
</table>
Co-training: Experimental Results

- begin with 12 labeled web pages (academic course)
- provide 1,000 additional unlabeled web pages
- average error: learning from labeled data 11.1%;
- average error: co-training 5.0%

<table>
<thead>
<tr>
<th></th>
<th>Page-base classifier</th>
<th>Link-based classifier</th>
<th>Combined classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised training</td>
<td>12.9</td>
<td>12.4</td>
<td>11.1</td>
</tr>
<tr>
<td>Co-training</td>
<td>6.2</td>
<td>11.6</td>
<td>5.0</td>
</tr>
</tbody>
</table>
When the generative model is not suitable

- **Multiple Mixture Components per Class** (M-EM). E.g., a class --- a number of sub-topics or clusters.
- Results of an example using 20 newsgroup data
  - 40 labeled; 2360 unlabeled; 1600 test
  - **Accuracy**
    - NB 68%
    - EM 59.6%
- **Solutions**
  - **M-EM** (Nigam et al, 2000): Cross-validation on the training data to determine the number of components.
  - **Partitioned-EM** (Cong, et al, 2004): using hierarchical clustering. It does significantly better than M-EM.
Summary

- Using unlabeled data can improve the accuracy of classifier when the data fits the generative model.
- Partitioned EM and the EM classifier based on multiple mixture components model (M-EM) are more suitable for real data when multiple mixture components are in one class.
- Co-training is another effective technique when redundantly sufficient features are available.
Further Topics

- Learning from Positive and Unlabeled Example (PU).

- Graph-based methods for Semi-supervised learning
  - Labeled and unlabeled examples are nodes in a graph
  - mincut: See the labeling of Us as a graph partition process (polynomial time)
  - **Spectral Graph transducer**: map the graph partition into a minimization problem and apply eigenvector analysis to find the best solutions. Parameters: balancing factors between P and U instances

- ICML ‘07 Tutorial (by Jerry Zhu) at: [http://pages.cs.wisc.edu/~jerryzhu/icml07tutorial.html](http://pages.cs.wisc.edu/~jerryzhu/icml07tutorial.html)