• Hard to design accurate classifier which generalizes well
• Easy to find many rule of thumb or weak classifiers
  • a classifier is weak if it is slightly better than random guessing
  • example: if an email has word “money” classify it as spam, otherwise classify it as not spam
    • likely to be better than random guessing
• Can we combine several weak classifiers to produce an accurate classifier?
  • Question people have been working on since 1980’s
  • Ada-Boost (1996) was the first practical boosting algorithm
Ada Boost: General form

- Assume 2-class problem, with labels +1 and -1
  - $y^i \in \{-1, 1\}$
- Ada boost produces a discriminant function:
  $$g(x) = \sum_{t=1}^{T} \alpha_t h_t(x) = \alpha_1 h_1(x) + \alpha_2 h_2(x) + ... + \alpha_T h_T(x)$$
- Where $h_t(x)$ is a weak classifier, for example:
  $$h_t(x) = \begin{cases} 
  -1 & \text{if email has word “money”} \\
  1 & \text{if email does not have word “money”}
  \end{cases}$$
- The final classifier is the sign of the discriminant function
  $$f_{final}(x) = \text{sign}[g(x)]$$
Ada Boost: Weak Classifiers

- Degenerate decision trees (decision stumps) are frequently used as weak classifiers

```
\begin{align*}
  h_t(x) = \begin{cases} 
  -1 & \text{if } x_3 > 10 \\
  1 & \text{if } x_3 \leq 10 
  \end{cases} \\
  h_t(x) = \begin{cases} 
  -1 & \text{if } x_7 > 60 \\
  1 & \text{if } x_7 \leq 60 
  \end{cases}
\end{align*}
```

- Based on thresholding just one feature

```
\begin{align*}
  X_1 > 5 \\
  X_3 > 6 \\
  \text{yes} & \text{no} \\
  \text{class 1} & \text{class 2} \\
  \text{yes} & \text{no} & \text{yes} & \text{no} \\
  \text{class 1} & \text{class 2} & \text{class 2} & \text{class 1}
\end{align*}
```
Ada Boost: Weak Classifiers

• Based on thresholding one feature

\[ h_t(x) = \begin{cases} -1 & \text{if } x_3 > 10 \\ 1 & \text{if } x_3 \leq 10 \end{cases} \]

\[ h_t(x) = \begin{cases} -1 & \text{if } x_7 > 60 \\ 1 & \text{if } x_7 \leq 60 \end{cases} \]

• Reverse polarity:

\[ h_t(x) = \begin{cases} -1 & \text{if } x_3 \leq 10 \\ 1 & \text{if } x_3 > 10 \end{cases} \]

\[ h_t(x) = \begin{cases} -1 & \text{if } x_7 \leq 60 \\ 1 & \text{if } x_7 > 60 \end{cases} \]

• There are approximately \( 2^d n \) distinct decision stump classifiers, where
  • \( n \) is number of training samples, \( d \) is dimension of samples
  • We will see why later

• Small decision trees are also popular weak classifiers
Idea Behind Ada Boost

- Algorithm is iterative
- Maintains distribution of weights over the training examples
- Initially weights are equal
- Main Idea: at successive iterations, the weight of misclassified examples is increased
- This forces the algorithm to concentrate on examples that have not been classified correctly so far
Weighted Examples

- Training examples are weighted with distribution $D(x)$
- Many classifiers can handle weighted examples
- But if classifier does not handle weighted examples we can sample $k > n$ examples according to distribution $D(x)$:

original data:

```
D(x):  1/16  1/4  1/16  1/16  1/4  1/16  1/4
```

data resampled according to $D(x)$:

- Apply classifier to the resampled data
Idea Behind Ada Boost

- misclassified examples get more weight
- more attention to examples of high weight
- Face/nonface classification problem:

### Round 1

<table>
<thead>
<tr>
<th></th>
<th>1/7</th>
<th>1/7</th>
<th>1/7</th>
<th>1/7</th>
<th>1/7</th>
<th>1/7</th>
<th>1/7</th>
</tr>
</thead>
<tbody>
<tr>
<td>best weak classifier:</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>change weights:</td>
<td>1/16</td>
<td>1/4</td>
<td>1/16</td>
<td>1/16</td>
<td>1/4</td>
<td>1/16</td>
<td>1/4</td>
</tr>
</tbody>
</table>

### Round 2

<table>
<thead>
<tr>
<th></th>
<th>1/8</th>
<th>1/32</th>
<th>11/32</th>
<th>1/2</th>
<th>1/8</th>
<th>1/32</th>
<th>1/32</th>
</tr>
</thead>
<tbody>
<tr>
<td>best weak classifier:</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>change weights:</td>
<td>1/8</td>
<td>1/32</td>
<td>11/32</td>
<td>1/2</td>
<td>1/8</td>
<td>1/32</td>
<td>1/32</td>
</tr>
</tbody>
</table>
Idea Behind Ada Boost

Round 3

• out of all available weak classifiers, we choose the one that works best on the data we have at round 3
• we assume there is always a weak classifier better than random (better than 50% error)

• image is 50% of our data
• chosen weak classifier has to classify this image correctly
More Comments on Ada Boost

• Ada boost is very simple to implement, provided you have an implementation of a “weak learner”

• Will work as long as the “basic” classifier $h_t(x)$ is at least slightly better than random
  • will work if the error rate of $h_t(x)$ is less than 0.5
  • 0.5 is the error rate of a random guessing for a 2-class problem

• Can be applied to boost any classifier, not necessarily weak
  • but there may be no benefits in boosting a “strong” classifier
Ada Boost for 2 Classes

**Initialization step:** for each example \( x \), set

\[
D(x) = \frac{1}{N}, \quad \text{where} \quad N \text{ is the number of examples}
\]

**Iteration step** (for \( t = 1 \ldots T \)):

1. Find best weak classifier \( h_t(x) \) using weights \( D(x) \)
2. Compute the error rate \( \varepsilon_t \) as

\[
\varepsilon_t = \sum_{i=1}^{N} D(x^i) \cdot I[y^i \neq h_t(x^i)]
\]
3. Compute weight \( \alpha_t \) of classifier \( h_t \)

\[
\alpha_t = \frac{1}{2} \log \left( \frac{(1 - \varepsilon_t)}{\varepsilon_t} \right)
\]
4. For each \( x^i \), \( D(x^i) = D(x^i) \cdot \exp \left( -\alpha_t \cdot y^i \cdot h_t(x^i) \right) \)
5. Normalize \( D(x^i) \) so that

\[
\sum_{i=1}^{N} D(x^i) = 1
\]

\[
f_{\text{final}}(x) = \text{sign} \left[ \sum \alpha_t h_t(x) \right]
\]
Ada Boost: Step 1

1. **Find best weak classifier** $h_t(x)$ **using weights** $D(x)$
   - use resampled data if classifier does not handle weights
   - decision stump weak classifier handles weights

   \[
   D(x) : \begin{align*}
   &1/16 &1/4 &1/16 &1/16 &1/4 &1/16 &1/4 \\
   X_3 : &1 &8 &7 &6 &4 &9 &9
   \end{align*}
   \]

   
   \[
   \times \quad \checkmark \quad \checkmark \quad \checkmark \quad \checkmark \quad \times \quad \times
   \]

   - weak classifier:
     \[
     h_t(x) = \begin{cases} 
     1 \text{ (face) } & \text{if } x_3 > 5 \\
     -1 \text{ (not face) } & \text{if } x_3 \leq 5
     \end{cases}
     \]

   - error rate: $1/16 + 1/16 + 1/4 = 3/8$
1. Find best weak classifier $h_t(x)$ using weights $D(x)$

- Give to the classifier the re-sampled examples:

- To find the best weak classifier, go through all weak classifiers, and find the one that gives the smallest error on the re-sampled examples:

<table>
<thead>
<tr>
<th>weak classifiers</th>
<th>$h_1(x)$</th>
<th>$h_2(x)$</th>
<th>$h_3(x)$</th>
<th>$h_m(x)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>errors</td>
<td>0.46</td>
<td>0.36</td>
<td>0.16</td>
<td>0.43</td>
</tr>
</tbody>
</table>

the best classifier $h_t(x)$ to choose at iteration $t$
2. Compute $\varepsilon_t$ the error rate as

$$\varepsilon_t = \sum_{i=1}^{N} D(x^i) \cdot I[y^i \neq h_t(x^i)] = \begin{cases} 1 & \text{if } y^i \neq h_t(x^i) \\ 0 & \text{otherwise} \end{cases}$$

- $\varepsilon_t$ is the weight of all misclassified examples added
- the error rate is computed over original examples, not the re-sampled examples
- If a weak classifier is better than random, then $\varepsilon_t < \frac{1}{2}$
3. compute weight $\alpha_t$ of classifier $h_t$

$$\alpha_t = \frac{1}{2} \log \left( \frac{1 - \varepsilon_t}{\varepsilon_t} \right)$$

In example from previous slide:

$$\varepsilon_t = \frac{5}{16} \Rightarrow \alpha_t = \frac{1}{2} \log \left( \frac{1 - \frac{5}{16}}{\frac{5}{16}} \right) = \frac{1}{2} \log \frac{11}{5} \approx 0.4$$

- Recall that $\varepsilon_t < \frac{1}{2}$
- Thus $(1 - \varepsilon_t)/\varepsilon_t > 1 \Rightarrow \alpha_t > 0$
- The smaller is $\varepsilon_t$, the larger is $\alpha_t$, and thus the more importance (weight) classifier $h_t(x)$

$$\text{final}(x) = \text{sign} \left[ \sum \alpha_t h_t(x) \right]$$
Ada Boost: Step 4

4. For each $x^i$, $D(x^i) = D(x^i) \cdot \exp(-\alpha_t \cdot y^i \cdot h_t(x^i))$

from previous slide $\alpha_t = 0.4$

- Weight of misclassified examples is increased
- Weight of correctly classified examples is decreased
5. \textbf{Normalize $D(x^i)$ so that $\sum D(x^i) = 1$}

from previous slide:

\begin{array}{cccccccc}
0.04 & 0.17 & 0.04 & 0.14 & 0.56 & 0.04 & 0.17 \\
\end{array}

• after normalization

\begin{array}{cccccccc}
0.03 & 0.15 & 0.03 & 0.12 & 0.48 & 0.03 & 0.15 \\
\end{array}

• In Matlab, if $D$ is weights vector, normalize with

$$D = D./\text{sum}(D)$$
AdaBoost Example

\[
C_1 = \begin{bmatrix}
2 & 3 \\
1 & 6 \\
5 & 9 \\
6 & 10 \\
8 & 8 \\
\end{bmatrix}
\]

\[
C_2 = \begin{bmatrix}
3 & 2 \\
4 & 7 \\
6 & 6 \\
9 & 3 \\
9 & 9 \\
\end{bmatrix}
\]
AdaBoost Example

- Decision stump weak classifiers

\[ h_t(x) = \begin{cases} 
1 & \text{if } x_1 > 7 \\
-1 & \text{if } x_1 \leq 7 
\end{cases} \]

- 6 samples misclassified, error 0.6

C1 = \[
\begin{bmatrix}
2 & 3 \\
1 & 6 \\
5 & 9 \\
6 & 10 \\
8 & 8 
\end{bmatrix}
\]

C2 = \[
\begin{bmatrix}
3 & 2 \\
4 & 7 \\
6 & 6 \\
9 & 3 \\
9 & 9 
\end{bmatrix}
\]
AdaBoost Example

• How many distinct classifiers based on thresholding feature 1 are there?

\[ h_t(x) = \begin{cases} 
1 & \text{if } x_1 > 7 \\
-1 & \text{if } x_1 \leq 7 
\end{cases} \]

• 6 samples misclassified
AdaBoost Example

- How many distinct classifiers based on thresholding feature 1 are there?

\[ h_t(x) = \begin{cases} 
1 & \text{if } x_1 > 7.5 \\
-1 & \text{if } x_1 \leq 7.5 
\end{cases} \]

- 6 samples misclassified, same classifier as with threshold 7.
AdaBoost Example

- Values of feature 1 in C1 and C2:
  1, 2, 3, 4, 5, 6, 8, 9

- Thresholds between any two consecutive values give same classifier
  - take two thresholds between 4 and 5, for example:

\[
 h_t(x) = \begin{cases} 
 1 & \text{if } x_1 > 4.2 \\
 -1 & \text{if } x_1 \leq 4.2
\end{cases}
\]

\[
 h_t(x) = \begin{cases} 
 1 & \text{if } x_1 > 4.8 \\
 -1 & \text{if } x_1 \leq 4.8
\end{cases}
\]

- get the same classifier with error 0.3
AdaBoost Example

- Values of feature 1 in C1 and C2:
  1, 2, 3, 4, 5, 6, 8, 9

- Take one threshold between each pair of feature values:
  \( a \in \{1.5, 2.5, 3.5, 4.5, 5.5, 7, 8.5\} \)

\[
\begin{align*}
    h_t(x) &= \begin{cases} 
        1 & \text{if } x_1 > a \\
        -1 & \text{if } x_1 \leq a 
    \end{cases}
\end{align*}
\]
AdaBoost Example

- Values of feature 1 in C1 and C2: 1, 2, 3, 4, 5, 6, 8, 9
- Two more distinct classifiers using a value smaller and larger than any value for feature 1, but these classifiers are largely useless: $a \in \{0, 10\}$

$$h_t(x) = \begin{cases} 1 & \text{if } x_1 > a \\ -1 & \text{if } x_1 \leq a \end{cases}$$

- Thresholds leading to distinct classifiers $a \in \{0, 1.5, 2.5, 3.5, 4.5, 5.5, 7, 8.5, 10\}$
AdaBoost Example

- Thresholds leading to distinct classifiers
  \( a \in \{0, 1.5, 2.5, 3.5, 4.5, 5.5, 7, 8.5, 10\} \)

- Reverse polarity to double number of classifiers:
  \[
  h_t(x) = \begin{cases} 
  1 & \text{if } x_1 \leq a \\
  -1 & \text{if } x_1 > a 
  \end{cases}
  \]

- Note error rates are reversed, compared to the same threshold but different polarity
AdaBoost Example

- Similar for feature 2

\[
\begin{bmatrix}
2 & 3 \\
1 & 6 \\
6 & 10 \\
8 & 8
\end{bmatrix}
\quad \begin{bmatrix}
3 & 2 \\
4 & 7 \\
9 & 3 \\
9 & 9
\end{bmatrix}
\]

\[C1 = \begin{bmatrix}
5 & 9 \\
6 & 10 \\
8 & 8
\end{bmatrix} \quad \begin{bmatrix}
6 & 6 \\
9 & 3 \\
9 & 9
\end{bmatrix} \quad C2 = \]

- Distinct values of feature 2:
\{2,3,6,7,8,9,10\}

- Thresholds leading to distinct classifiers
\[a \in \{1, 2.5, 4.5, 6.5, 7.5, 8.5, 9.5, 11\}\]

\[h_t(x) = \begin{cases} 
1 & \text{if } x_2 \leq a \\
-1 & \text{if } x_2 > a 
\end{cases} \]

\[
\begin{array}{c}
\text{err } = 0.5 \\
\text{err } = 0.6 \\
\text{err } = 0.5 \\
\text{err } = 0.7 \\
\text{err } = 0.6 \\
\text{err } = 0.6 \\
\text{err } = 0.6 \\
\text{err } = 0.5
\end{array}
\]
AdaBoost Example

- Reverse polarity
- Thresholds leading to distinct classifiers

\[
h_t(x) = \begin{cases} 
1 & \text{if } x_2 > a \\
-1 & \text{if } x_2 \leq a
\end{cases}
\]

\[a \in \{1, 2.5, 4.5, 6.5, 7.5, 8.5, 9.5, 11\}\]
AdaBoost Example

- Thus total number of decision-stump weak classifiers is, approximately, $2 \cdot n \cdot d$
  - $d$ is number of features
  - $n$ is times number of samples
  - $2$ comes from polarity
- Small (shallow) decision trees are also popular as weak classifiers
  - gives more weak classifiers
• Initialization: all examples have equal weights

\[ D_1 = [0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]^T \]
AdaBoost Example: Round 1

- Classifier chosen:
  
  \[ h_1(x) = \begin{cases} 
  1 & \text{if } x_1 \leq 2.5 \\
  -1 & \text{if } x_1 > 2.5 
  \end{cases} \]

- \( \varepsilon_1 = 0.3, \alpha_1 = 0.42 \)

\[
C_1 = \begin{bmatrix}
2 & 3 \\
1 & 6 \\
5 & 9 \\
6 & 10 \\
8 & 8 
\end{bmatrix}
\]

\[
C_2 = \begin{bmatrix}
3 & 2 \\
4 & 7 \\
6 & 6 \\
9 & 3 \\
9 & 9 
\end{bmatrix}
\]

\[
D_2 = \begin{bmatrix}
0.07 & 0.07 \\
0.07 & 0.07 \\
0.17 & 0.07 \\
0.17 & 0.07 \\
0.17 & 0.07 \\
0.07 & 0.07 \\
0.07 & 0.07 \\
0.07 & 0.07 \\
0.07 & 0.07 \\
0.07 & 0.07 
\end{bmatrix}
\]

\[ x_2 \]

\[ x_1 \]
AdaBoost Example: Round 2

- Classifier chosen:

\[ h_2(x) = \begin{cases} 
1 & \text{if } x_1 \leq 8.5 \\
-1 & \text{if } x_1 > 8.5 
\end{cases} \]

- \( \varepsilon_2 = 0.21, \alpha_2 = 0.66 \)

\[
\begin{bmatrix}
2 & 3 \\
1 & 6 \\
5 & 9 \\
6 & 10 \\
8 & 8
\end{bmatrix}
\quad \begin{bmatrix}
0.04 \\
0.04 \\
0.10 \\
0.10 \\
0.10
\end{bmatrix}
\]

\[
\begin{bmatrix}
3 & 2 \\
4 & 7 \\
6 & 6 \\
9 & 3 \\
9 & 9
\end{bmatrix}
\quad \begin{bmatrix}
0.17 \\
0.17 \\
0.17 \\
0.04 \\
0.04
\end{bmatrix}
\]

\[
C1 = \begin{bmatrix}
2 & 3 \\
1 & 6 \\
5 & 9 \\
6 & 10 \\
8 & 8
\end{bmatrix}
\quad D_3 = \begin{bmatrix}
3 & 2 \\
4 & 7 \\
6 & 6 \\
9 & 3 \\
9 & 9
\end{bmatrix}
\]

\[
\begin{bmatrix}
2 & 3 \\
1 & 6 \\
5 & 9 \\
6 & 10 \\
8 & 8
\end{bmatrix}
\quad \begin{bmatrix}
0.04 \\
0.04 \\
0.10 \\
0.10 \\
0.10
\end{bmatrix}
\]

\[
\begin{bmatrix}
3 & 2 \\
4 & 7 \\
6 & 6 \\
9 & 3 \\
9 & 9
\end{bmatrix}
\quad \begin{bmatrix}
0.17 \\
0.17 \\
0.17 \\
0.04 \\
0.04
\end{bmatrix}
\]
AdaBoost Example: Round 3

- Classifier chosen:

$$h_3(x) = \begin{cases} 1 & \text{if } x_2 > 7.5 \\ -1 & \text{if } x_2 \leq 7.5 \end{cases}$$

- $\varepsilon_3 = 0.12$, $\alpha_3 = 1.0$
AdaBoost Final Classifier

\[ f_{\text{final}}(x) = \begin{bmatrix} \text{sign} \left[ 0.42 \right] & +0.66 & +1.0 \end{bmatrix} \]
AdaBoost Comments

• Can show that training error drops exponentially fast

\[ \text{Err}_{\text{train}} \leq \exp \left( -2 \sum_t \gamma_t^2 \right) \]

• Here \( \gamma_t = \varepsilon_t - 1/2 \), where \( \varepsilon_t \) is classification error at round \( t \)

• Example: let errors for the first four rounds be, 0.3, 0.14, 0.06, 0.03, 0.01 respectively. Then

\[ \text{Err}_{\text{train}} \leq \exp \left[ -2 \left( 0.2^2 + 0.36^2 + 0.44^2 + 0.47^2 + 0.49^2 \right) \right] \]

\[ \approx 0.19 \]

• Thus \( \log(n) \) rounds of boosting are sufficient to get zero training error
  • provided weak learners are better than random
AdaBoost Comments

• We are really interested in the generalization properties of \( f_{\text{FINAL}}(x) \), not the training error

• AdaBoost was shown to have excellent generalization properties in practice
  • the more rounds, the more complex is the final classifier, so overfitting is expected as the training proceeds
  • but in the beginning researchers observed no overfitting of the data
  • It turns out it does overfit data eventually, if you run it really long

• It can be shown that boosting increases the margins of training examples, as iterations proceed
  • larger margins help better generalization
  • margins continue to increase even when training error reaches zero
  • helps to explain empirically observed phenomena: test error continues to drop even after training error reaches zero
AdaBoost Example

- zero training error
- larger margins helps better generalization
## Margin Distribution

<table>
<thead>
<tr>
<th></th>
<th>5</th>
<th>100</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Iteration number</strong></td>
<td>5</td>
<td>100</td>
<td>1000</td>
</tr>
<tr>
<td><strong>training error</strong></td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td><strong>test error</strong></td>
<td>8.4</td>
<td>3.3</td>
<td>3.1</td>
</tr>
<tr>
<td><strong>%margins ≤ 0.5</strong></td>
<td>7.7</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td><strong>Minimum margin</strong></td>
<td>0.14</td>
<td>0.52</td>
<td>0.55</td>
</tr>
</tbody>
</table>

### Graphs

- **Left Graph:**
  - X-axis: # of rounds ($T$)
  - Y-axis: Error
  - Graphs labeled as 'test' and 'train'

- **Right Graph:**
  - X-axis: Margin
  - Y-axis: Cumulative distribution
  - Markers at 5, 100, 1000
Practical Advantages of AdaBoost

- Can construct arbitrarily complex decision regions
- Fast
- Simple
- Has only one parameter to tune, $T$
- Flexible: can be combined with any classifier
- Provably effective (assuming weak learner)
  - Shift in mind set: goal now is merely to find hypotheses that are better than random guessing
• AdaBoost can fail if
  • weak hypothesis too complex (overfitting)
  • weak hypothesis too weak \((\gamma_t \rightarrow 0)\) too quickly,
    • underfitting
• empirically, AdaBoost seems especially susceptible to noise
  • noise is the data with wrong labels
Applications

• Face Detection

• Object Detection

http://www.youtube.com/watch?v=2_0SmxvDbKs