Perceptron

Linear Threshold Algorithms

LTU trick:
• Hypothesis: \((w, b)\) pair,
  predict + if \(w \cdot x > b\), and - otherwise
  (recall \(w \cdot x = \sum_{i=1}^{n} w_ix_i\))
• Set \(x' = x\) with new component \(x_0=1\)
• Set \(w'=w\) with new component \(w_0=-b\)
• Now \(w \cdot x > b\) exactly when \(w' \cdot x' > 0\)

Perceptron Algorithm
• Keeps weights \(w_j\) one per feature
• Online algorithm, initially \(w = (0, \ldots, 0)\)
• Repeat (repeating data if needed):
  get next training example \((x_i, y_i)\)
  if \((w \cdot x_i) y_i \leq 0\) then mistake:
  \(w\) gets \(w + \eta_i y_i x_i\)

\(\eta_i\) values are learning rates (step sizes)

Perceptron Convergence
• If data linearly separable with "gap" then
  converges within \((1/gap)^2\) mistakes when \(\eta_i = 1/|x_i|\) (normalize instances to length 1)
• For data not linearly separable it converges if
  (Robbins-Munro alg):
  \(\eta_i\) values go to 0 (as \(i\) goes to \(\infty\))
  sum of \(\eta_i\) values goes to \(\infty\)
  sum of \((\eta_i)^2\) values finite

Perceptron Class Exercise:
• Assume \(\eta_i\) always 1

<table>
<thead>
<tr>
<th>(x_1)</th>
<th>(x_2)</th>
<th>(y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>+1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>-1</td>
</tr>
<tr>
<td>-3</td>
<td>1</td>
<td>+1</td>
</tr>
<tr>
<td>1</td>
<td>-1</td>
<td>-1</td>
</tr>
</tbody>
</table>

(gap = 2/15)

Perceptron as gradient
descent
• Perceptron criteria: minimize badness of mistake on example \(i\):
  \(-y_i (w \cdot x_j)\)
• Differentiate wrt \(w_j\) gives gradient component:
  \(-y_i x_{ij}\)
• Negative gradient, \(y_j x_i\) is direction of steepest descent, add \(y_j x_i\) to \(w\) (for each \(j\)) or equivalently add \(y_j x_i\) to \(w\)
Perceptron notes

• Can run in batch mode - save updates until complete run through
• Voted perceptron idea
• Multiclass- learn a \( w_y \) for each class,
  predict with \( y \) maximizing \( w_y \mathbf{x} \)
• Learns classifier directly (no probability)