Machine Learning for Intelligent Systems

Lecture 7: Convergence of Perceptron

Reading: UML 9.1

Instructors: Nika Haghtalab (this time) and Thorsten Joachims
Linear Classifiers

For a vector $\vec{w} \in \mathbb{R}^d$ and $b \in \mathbb{R}$, the hypothesis $h_{\vec{w},b} : \mathbb{R}^d \rightarrow \mathbb{R}$ defined below is called a linear classifier/linear predictor/halfspace

$$h_{\vec{w},b}(\vec{x}) = \text{sign}(\vec{w} \cdot \vec{x} + b) = \begin{cases} +1 & \vec{w} \cdot \vec{x} + b > 0 \\ -1 & \vec{w} \cdot \vec{x} + b \leq 0 \end{cases}$$

Homogenous linear classifier: $b = 0$
Homogenous vs. Non-homogenous

Any $d$-dimensional learning problem for **non-homogenous linear classifiers** has a **homogenous** form in $(d+1)$ dimension.

<table>
<thead>
<tr>
<th>Non-Homogenous</th>
<th>Homogenous</th>
</tr>
</thead>
<tbody>
<tr>
<td>$HS^d = {h_{\vec{w},b}</td>
<td>\vec{w} \in \mathbb{R}^d, b \in \mathbb{R}}$</td>
</tr>
<tr>
<td>$\vec{x}$</td>
<td>$\vec{x}' = (\vec{x}, +1)$</td>
</tr>
<tr>
<td>$\vec{w}, b$</td>
<td>$\vec{w}' = (\vec{w}, b)$</td>
</tr>
<tr>
<td>$\vec{w} \cdot \vec{x} + b$</td>
<td>$\vec{w}' \cdot \vec{x}' = \vec{w} \cdot \vec{x} + b$</td>
</tr>
</tbody>
</table>

Without loss of generality, focus on **homogenous linear classifiers**.
Improving a linear classifier

If there is a homogeneous linear classifier that is consistent with \{((\vec{x}_1, y_1), (\vec{x}_2, y_2), ..., (\vec{x}_m, y_m))\}, how can we find it?

Last time: Do it with a linear program

This time: Start with a guess and improve it.

Move away from misclassified negative points: \( \vec{w} - \vec{x} \)
Improving a linear classifier

If there is a homogeneous linear classifier that is consistent with \{\(\vec{x}_1, y_1\), \(\vec{x}_2, y_2\), \ldots, \(\vec{x}_m, y_m\)\}, how can we find it?

Last time: Do it with a linear program

This time: Start with a guess and improve it.

Move away from misclassified negative points: \(\vec{w} - \vec{x}\)
Improving a linear classifier

If there is a homogeneous linear classifier that is consistent with \{(\vec{x}_1, y_1), (\vec{x}_2, y_2), ..., (\vec{x}_m, y_m)\}, how can we find it?

**Last time:** Do it with a linear program

**This time:** Start with a guess and improve it.

---

**Move away from misclassified negative points:** $\vec{w} - \vec{x}$

**Move towards misclassified positive points:** $\vec{w} + \vec{x}$
Improving a linear classifier

If there is a homogeneous linear classifier that is consistent with\n\{(\vec{x}_1, y_1), (\vec{x}_2, y_2), ..., (\vec{x}_m, y_m)\}, how can we find it?

Last time: Do it with a linear program
This time: Start with a guess and improve it.

Move away from misclassified negative points: \(\vec{w} - \vec{x}\)

Move towards misclassified positive points: \(\vec{w} + \vec{x}\)
Perceptron (homogeneous & batch)

**Input:** Training data set \{ (\vec{x}_1, y_1), (\vec{x}_2, y_2), ..., (\vec{x}_m, y_m) \}

**Initialize** \( \vec{w}^{(0)} = (0, ..., 0) \), \( t = 0 \)

**While** there is \( i \in [m] \), such that \( y_i (\vec{w}^{(t)} \cdot \vec{x}_i) \leq 0 \) then,

- \( \vec{w}^{(t+1)} = \vec{w}^{(t)} + y_i \vec{x}_i \)

- \( t \leftarrow t + 1 \)

**End While**

**Output** \( \vec{w}^{(t)} \)
Example

\[
\begin{align*}
\vec{w}(s) &= (0, 0) \\
\Rightarrow y_1(\vec{w}(0) \cdot \vec{x}_1) &= 0 \leq 0 \\
\vec{w}(1) &= \vec{w}(0) + x_1 = (1,2) \\
\Rightarrow y_1(\vec{w}(1) \cdot \vec{x}_1) &= 5 > 0 \\
\Rightarrow y_2(\vec{w}(1) \cdot \vec{x}_2) &= 4 > 0 \\
\Rightarrow y_3(\vec{w}(1) \cdot \vec{x}_3) &= 3 > 0 \\
\Rightarrow y_4(\vec{w}(1) \cdot \vec{x}_4) &= -1 \leq 0 \\
\vec{w}(2) &= \vec{w}(1) - x_4 = (2,1)
\end{align*}
\]
Given a data set \( S = \{ (\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_m, y_m) \} \) and a linear classifier \( h_{\vec{w}} \) that is consistent with \( S \), that is, \( y_i(\vec{w} \cdot \vec{x}_i) > 0 \), the geometric margin of \( h_{\vec{w}} \) is defined as:

\[
\gamma := \min_{i \in S} \frac{y_i(\vec{w} \cdot \vec{x}_i)}{\| \vec{w} \|}
\]

Margin \( \gamma \) is the distance of the closest instance to hyperplane \( \vec{w} \cdot \vec{x} = 0 \).
Convergence of Perceptron

Theorem: Convergence of Perceptron

Given a data set $S = \{ (\vec{x}_1, y_1), (\vec{x}_2, y_2), ..., (\vec{x}_m, y_m) \}$ and radius $R$ such that $\| \vec{x}_i \| \leq R$ for all $i \in [m]$.

If $h_{\vec{w}^*}$ is consistent with $S$ with margin $\gamma := \min_{i \in S} \frac{y_i (\vec{w}^* \cdot \vec{x}_i)}{\| \vec{w}^* \|}$ then Perceptron makes at most $R^2/\gamma^2$ updates before predicting every label perfectly.

Idea: $h_{\vec{w}^*}$ has $\gamma$ margin $\rightarrow$ there is wiggle room.
$\rightarrow$ Show that within $t = R^2/\gamma^2$, 
$\vec{w}^t$ is close to $\vec{w}^*$ in angle.
Proof Ideas

**Theorem: Convergence of Perceptron**

Perceptron makes at most \( R^2 / \gamma^2 \) updates before predicting every label perfectly. Recall margin \( \gamma := \min_{i \in S} \frac{y_i (\vec{w}^* \cdot \vec{x}_i)}{\|\vec{w}^*\|} \).

**Idea: Proof by Contradiction.** Show that if within \( t > R^2 / \gamma^2 \),

\[
\cos(\theta(\vec{w}^*, \vec{w}^{(t+1)})) = \frac{\vec{w}^* \cdot \vec{w}^{(t+1)}}{\|\vec{w}^*\| \|\vec{w}^{(t+1)}\|} > 1 \quad \text{Impossible}
\]

**Plan:**

- Assume \( \vec{w}^* \) is normalized to be unit vector \( \rightarrow \) margin doesn’t change.
- 1. Show that \( \vec{w}^* \cdot \vec{w}^{(t+1)} \) is large (find a lower bound).
- 2. Show that \( \|\vec{w}^{(t+1)}\| \) is not too large (find an upper bound).
- \( \rightarrow \) So, if \( t > R^2 / \gamma^2 \), the cosine will be larger than 1 \( \rightarrow \) Contradiction.
Recall: Example

- Update on \((x_2, y_2)\)
  \(\vec{w}^{(1)} = (2, 1)\) converges in 1 step.

- Update on \((x_3, y_3)\)
  \(\vec{w}^{(1)} = (1, 1)\)
  Update on \((x_4, y_4)\)
  \(\vec{w}^{(2)} = (2, 0)\)

- \(\vec{w}^{(0)} = (0, 0)\)
  \(\Rightarrow y_1(\vec{w}^{(0)} \cdot \vec{x}_1) = 0\)

- \(\vec{w}^{(1)} = \vec{w}^{(0)} + x_1 = (1,2)\)
  \(\Rightarrow y_1(\vec{w}^{(1)} \cdot \vec{x}_1) = 5 > 0\)
  \(\Rightarrow y_2(\vec{w}^{(1)} \cdot \vec{x}_2) = 4 > 0\)
  \(\Rightarrow y_3(\vec{w}^{(1)} \cdot \vec{x}_3) = 3 > 0\)
  \(\Rightarrow y_4(\vec{w}^{(1)} \cdot \vec{x}_4) = -1 \leq 0\)

- \(\vec{w}^{(2)} = \vec{w}^{(1)} - x_4 = (2, 1)\)
Online Perceptron

Theorem: Mistake Bound of Online Perceptron

Given a sequence of data \((\vec{x}_1, y_1), (\vec{x}_2, y_2), \ldots, (\vec{x}_m, y_m)\) one by one, with radius \(R\) and margin \(\gamma \coloneqq \min_{i \in S} \frac{y_i(\vec{w}^* \cdot \vec{x}_i)}{\|\vec{w}^*\|}\) for some \(\vec{w}^*\).

**Online prediction:** At each time use the current \(\vec{w}\) to predict the label of incoming \((\vec{x}_i, y_i)\), update if needed.

**Mistake Bound:** The number of mistake that perceptron makes is at most \(R^2 / \gamma^2\).
Example: Reuters Text Classification

"optimal hyperplane"