

ECE595 / STAT598: Machine Learning I

Lecture 16.2: Perceptron 1 - Properties of Perceptron Loss

Spring 2020

Stanley Chan

School of Electrical and Computer Engineering
Purdue University



Outline

Discriminative Approaches

- Lecture 16 Perceptron 1: Definition and Basic Concepts
- Lecture 17 Perceptron 2: Algorithm and Property

This lecture: Perceptron 1

- From Logistic to Perceptron
 - What is Perceptron? Why study it?
 - Perceptron Loss
 - Connection with other losses
- Properties of Perceptron Loss
 - Convexity
 - Comparing with Bayesian Oracle
 - Preview of Perceptron Algorithm

Convexity of Perceptron (Soft) Loss

Let us consider the Perceptron (Soft) Loss

$$J_{\text{soft}}(\boldsymbol{\theta}) = \sum_{n=1}^N \max \left\{ -y_n g_{\boldsymbol{\theta}}(\mathbf{x}_n), 0 \right\}$$

- Is this convex?
- Pick any $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$. Pick $\lambda \in [0, 1]$.
- We want to show that

$$J(\lambda\boldsymbol{\theta}_1 + (1 - \lambda)\boldsymbol{\theta}_2) \leq \lambda J(\boldsymbol{\theta}_1) + (1 - \lambda)J(\boldsymbol{\theta}_2)$$

- But notice that

$$\begin{aligned} y_n g_{\boldsymbol{\theta}}(\mathbf{x}_n) &= y_n (\mathbf{w}^T \mathbf{x}_n + w_0) = (y_n \mathbf{x}_n)^T \mathbf{w} + y_n w_0 \\ &= \begin{bmatrix} y_n \mathbf{x}_n^T & y_n \end{bmatrix} \begin{bmatrix} \mathbf{w} \\ w_0 \end{bmatrix} = \mathbf{a}^T \boldsymbol{\theta}. \end{aligned}$$

Convexity of Perceptron (Soft) Loss

- Basic fact: If $f(\cdot)$ is convex, then $f(\mathbf{A}(\cdot) + \mathbf{b})$ is also convex.
- Recognize

$$f(s) = \max \{ -s, 0 \}$$

- So if we can show that $f(s) = \max\{-s, 0\}$ is convex (in s), then

$$f(\mathbf{a}^T \boldsymbol{\theta})$$

is also convex. Put $s = \mathbf{a}^T \boldsymbol{\theta}$.

- Let $\lambda \in [0, 1]$, and consider two points $s_1, s_2 \in \text{dom} f$
- Want to show that

$$f(\lambda s_1 + (1 - \lambda)s_2) \leq \lambda f(s_1) + (1 - \lambda)f(s_2).$$

Convexity of Perceptron (Soft) Loss

- Want to show that

$$f(\lambda s_1 + (1 - \lambda)s_2) \leq \lambda f(s_1) + (1 - \lambda)f(s_2).$$

- Use the fact that $\max(a + b, 0) \leq \max(a, 0) + \max(b, 0)$
- Equality when $(a > 0 \text{ and } b > 0)$ or $(a < 0 \text{ and } b < 0)$
- Then we can show that

$$\begin{aligned} f(\lambda s_1 + (1 - \lambda)s_2) &= \max(-(\lambda s_1 + (1 - \lambda)s_2), 0) \\ &\leq \max\{-\lambda s_1, 0\} + \max\{-(1 - \lambda)s_2, 0\} \\ &= \lambda \max(-s_1, 0) + (1 - \lambda) \max(-s_2, 0) \\ &= \lambda f(s_1) + (1 - \lambda)f(s_2). \end{aligned}$$

- So the perceptron (soft) loss is convex.
- Therefore, $J_{\text{soft}}(\theta)$ is convex in θ .

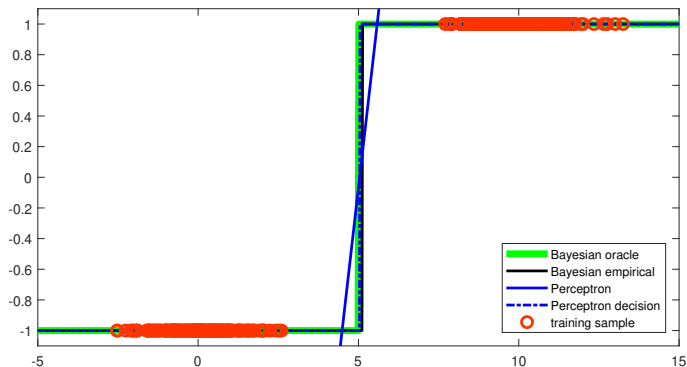
Implication of Convexity

- You can use CVX to solve the (soft) problem!
- **Existence:** There must exist $\theta^* \in \text{dom}J$ such that $J(\theta^*) \leq J(\theta)$ for any $\theta \in \text{dom}J$
- **Uniqueness:** Any local minimizer is also a global minimizer with unique global optimal value.
- **Optimal Value:** If the two classes are linearly separable, then the global minimum is achieved when $J(\theta^*) = 0$
- That means all training samples are classified correctly
- If the two classes are not linearly separable, then you can still get a solution. But $J(\theta^*) > 0$.

Comparing Perceptron and Bayesian Oracle

- **Scenario 1:**

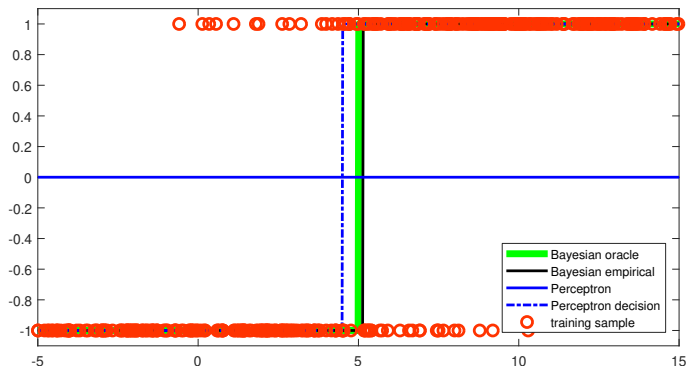
- $\mathcal{N}(0, 2)$ with 50 samples and $\mathcal{N}(10, 2)$ with 50 samples.



- When everything is “ideal”, perceptron is pretty good.

Comparing Perceptron and Bayesian Oracle

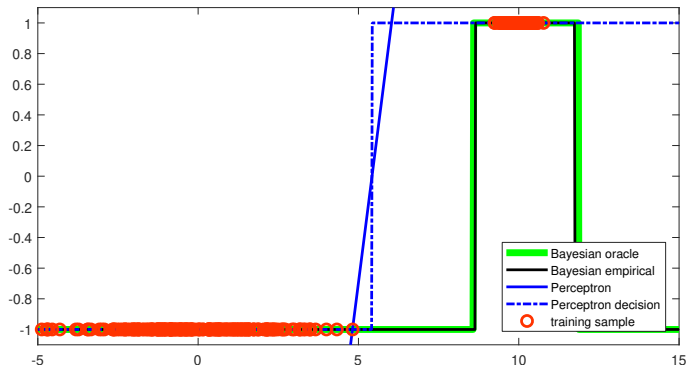
- **Scenario 2:**
- $\mathcal{N}(0, 4)$ with 200 samples and $\mathcal{N}(10, 4)$ with 200 samples.



- Even when datasets are intrinsically overlapping, perceptron is still okay.

Comparing Perceptron and Bayesian Oracle

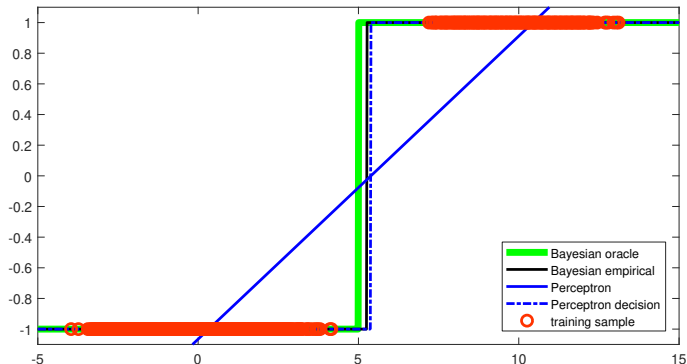
- **Scenario 3:**
- $\mathcal{N}(0, 2)$ with 200 samples and $\mathcal{N}(10, 0.3)$ with 200 samples.



- When Gaussians have different covariances, the perceptron (as a linear classifier) does not work.

Comparing Perceptron and Bayesian Oracle

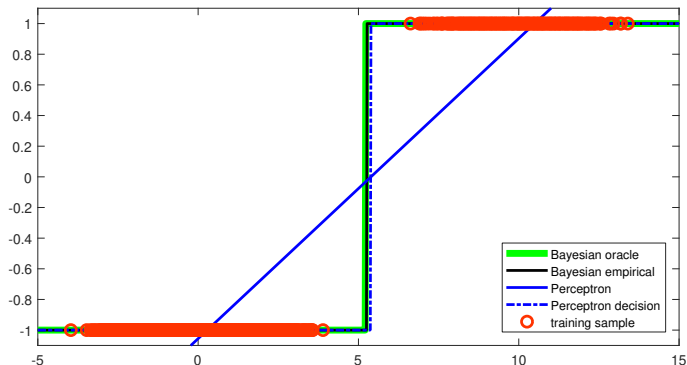
- **Scenario 4:**
- $\mathcal{N}(0, 1)$ with 1800 samples and $\mathcal{N}(10, 1)$ with 200 samples.



- Number of training samples, in this example, does not seem to affect the algorithm.

Comparing Perceptron and Bayesian Oracle

- **Scenario 5:** 1800 samples and 200 samples.
- $\mathcal{N}(0, 1)$ with $\pi_0 = 0.9$ and $\mathcal{N}(10, 1)$ with $\pi_1 = 0.1$.



- Intrinsic imbalance between the two distributions does not seem to affect the algorithm.

Perceptron with Hard Loss

- Historically, we have perceptron algorithm way earlier than CVX.
- Before the age of CVX, people solve perceptron using gradient descent.
- Let us be explicit about which loss:

$$J_{\text{hard}}(\boldsymbol{\theta}) = \sum_{j=1}^N \max \left\{ -y_j h_{\boldsymbol{\theta}}(\mathbf{x}_j), 0 \right\}$$

$$J_{\text{soft}}(\boldsymbol{\theta}) = \sum_{j=1}^N \max \left\{ -y_j g_{\boldsymbol{\theta}}(\mathbf{x}_j), 0 \right\}$$

- Goal:** To get a solution for $J_{\text{hard}}(\boldsymbol{\theta})$
- Approach:** Gradient descent on $J_{\text{soft}}(\boldsymbol{\theta})$

Re-defining the Loss

- **Main idea:** Use the fact that

$$J_{\text{soft}}(\boldsymbol{\theta}) = \sum_{j=1}^N \max \left\{ -y_j g_{\boldsymbol{\theta}}(\mathbf{x}_j), 0 \right\}$$

is the same as this loss function

$$J(\boldsymbol{\theta}) = - \sum_{j \in \mathcal{M}(\boldsymbol{\theta})} y_j g_{\boldsymbol{\theta}}(\mathbf{x}_j).$$

- $\mathcal{M}(\boldsymbol{\theta}) \subseteq \{1, \dots, N\}$ is the set of misclassified samples.
- Run gradient descent on $J(\boldsymbol{\theta})$, but fixing $\mathcal{M}(\boldsymbol{\theta}) \leftarrow \mathcal{M}(\boldsymbol{\theta}^k)$ for iteration k .

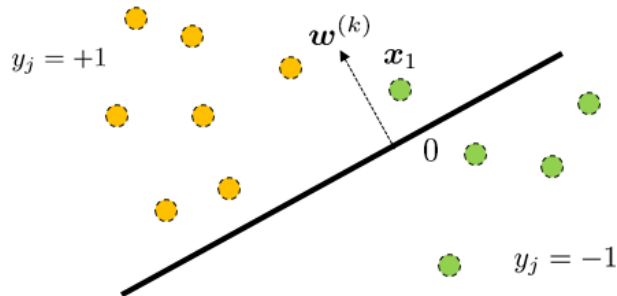
Perceptron Algorithm

- Therefore, the algorithm is
- For $k = 1, 2, \dots$,
- Update $\mathcal{M}_k = \{j \mid y_j g_{\theta}(\mathbf{x}_j) < 0\}$ for $\theta = \theta^{(k)}$.
- Gradient descent

$$\begin{bmatrix} \mathbf{w}^{(k+1)} \\ w_0^{(k+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{w}^{(k)} \\ w_0^{(k)} \end{bmatrix} + \alpha_k \sum_{j \in \mathcal{M}_k} \begin{bmatrix} y_j \mathbf{x}_j \\ y_j \end{bmatrix}.$$

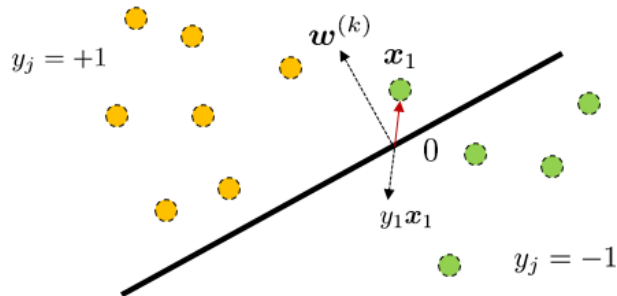
- End For
- The set \mathcal{M}_k can grow or can shrink from \mathcal{M}_{k-1} .
- If training samples are linearly separable, then converge. Zero training loss.
- If training samples are not linearly separable, then oscillates.

Updating One Sample



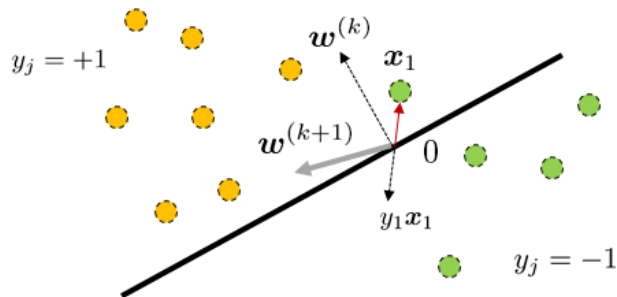
- Initially there is a $\mathbf{w}^{(k)}$.
- There is a mis-classifier training sample \mathbf{x}_1

Updating One Sample



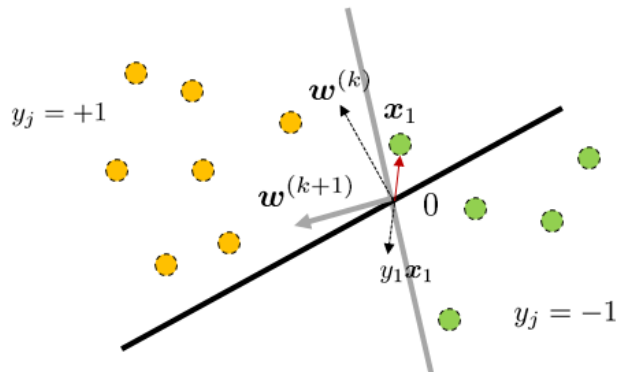
- Perceptron algorithm finds $y_1 \mathbf{x}_1$
- $y_1 \mathbf{x}_1$ is in the opposite direction as \mathbf{x}_1

Updating One Sample



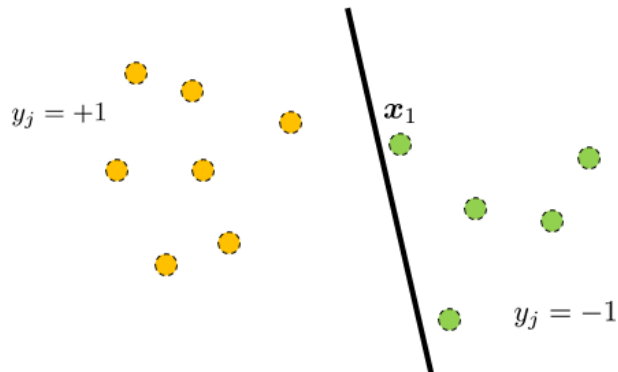
- $w^{(k+1)}$ is a linear combination of $w^{(k)}$ and $y_1 x_1$.

Updating One Sample



- $w^{(k+1)}$ gives a new separating hyperplane.

Updating One Sample



- Now you are happy!

Reading List

Perceptron Algorithm

- Abu-Mostafa, Learning from Data, Chapter 1.2
- Duda, Hart, Stork, Pattern Classification, Chapter 5.5
- Cornell CS 4780 Lecture <https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote03.html>
- UCSD ECE 271B Lecture <http://www.svcl.ucsd.edu/courses/ece271B-F09/handouts/perceptron.pdf>