COMP4901K/Math4824B
Machine Learning for Natural Language Processing

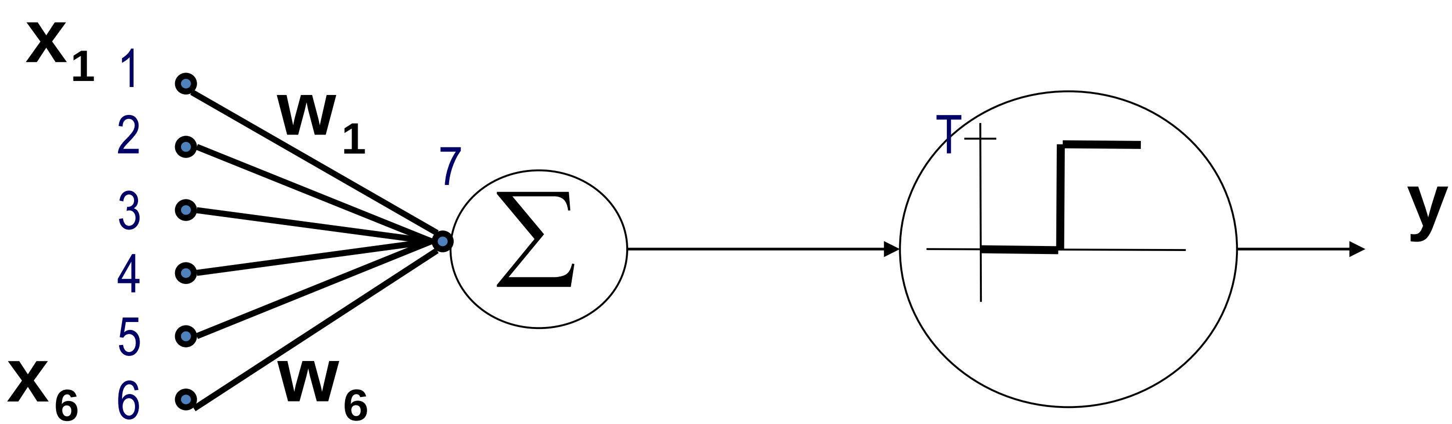
Lecture 10: Perceptron, Error-Driven Classification

Instructor: Yangqiu Song

Slides credits: Dan Roth, Barbara Rosario

Perceptron learning rule

- On-line, mistake driven algorithm.
- Rosenblatt (1959) suggested that when a target output value is provided for a single neuron with fixed input, it can incrementally change weights and learn to produce the output using the Perceptron learning rule
- (Perceptron == Linear Threshold Unit)



Linear binary classification

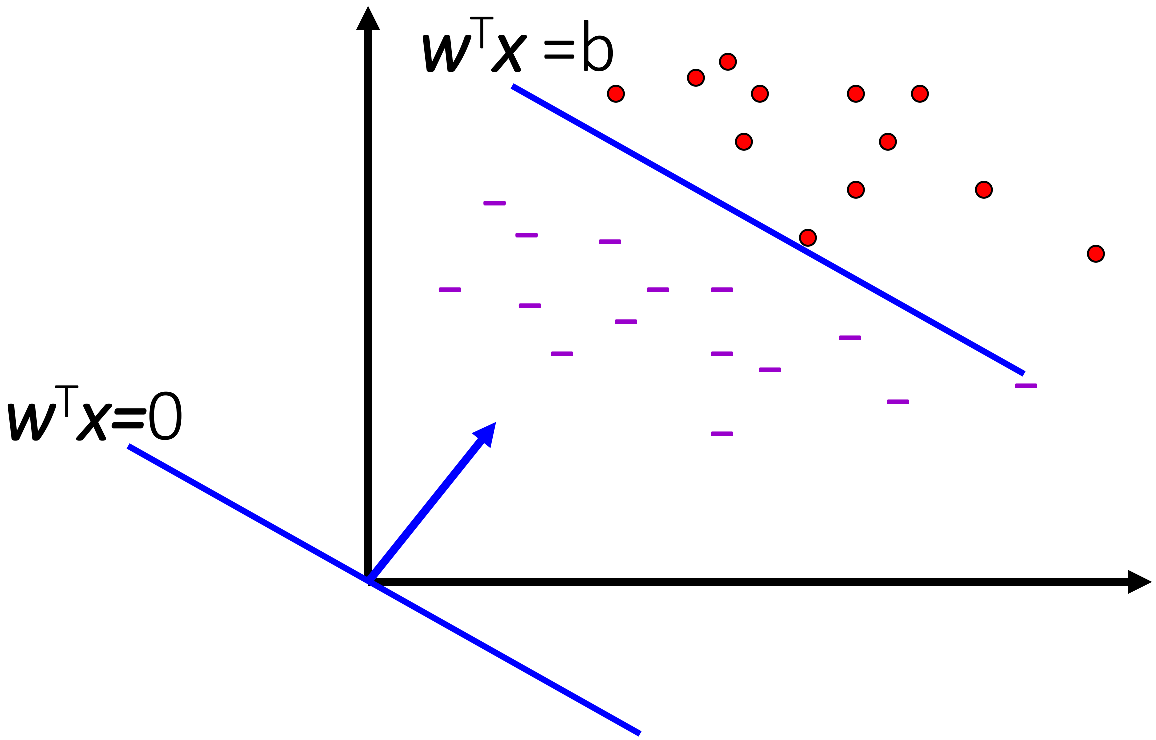
- Data: $\{(\boldsymbol{x}_i, y_i)\}_{i=1\ldots n}$
  - $\boldsymbol{x}$ in R^d (x is a vector in d-dimensional space)

    → feature vector
  - y in {-1,+1}

    → label (class, category)
- Question:
  - Design a linear decision boundary: $\boldsymbol{w}^T\boldsymbol{x} + b$ (equation of hyperplane) such that the classification rule associated with it has minimal probability of error
  - classification rule
    - $y = \text{sign}(\ \boldsymbol{w}^T\boldsymbol{x} + b\ )$ which means:
    - if $\boldsymbol{w}^T\boldsymbol{x} + b > 0$ then $y = +1$
    - if $\boldsymbol{w}^T\boldsymbol{x} + b < 0$ then $y = -1$

Perceptron learning rule

- We learn f:**X**$\rightarrow${-1,+1} represented as f =sgn{$\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}$}
- Where **X**= $\{0,1\}^n$ or **X**= R^n and $\boldsymbol{w} \in R^n$
- Given Labeled examples: $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots (\boldsymbol{x}_N, y_N)\}$

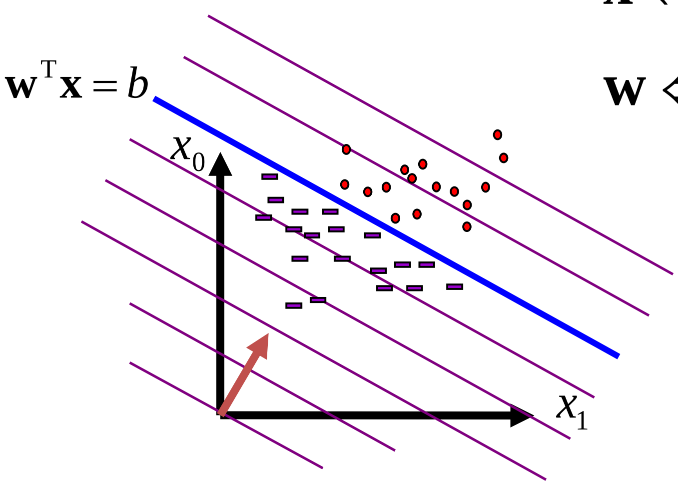


Footnote About the Threshold

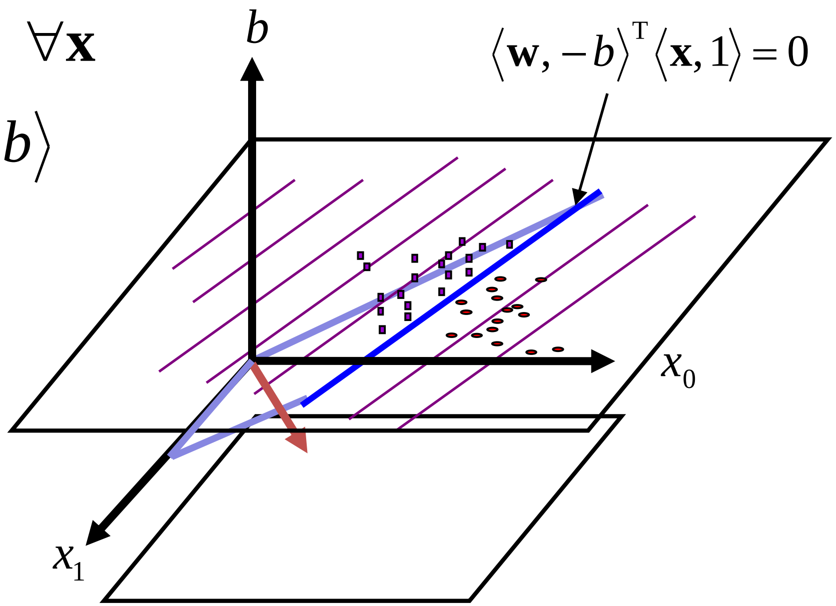
- On previous slide, Perceptron has no threshold
- But we don't lose generality:

$$\mathbf{x} \Leftrightarrow \langle \mathbf{x}, 1 \rangle \quad \forall \mathbf{x}$$

$$\mathbf{w} \Leftrightarrow \langle \mathbf{w}, -b \rangle$$



$$\Leftrightarrow$$



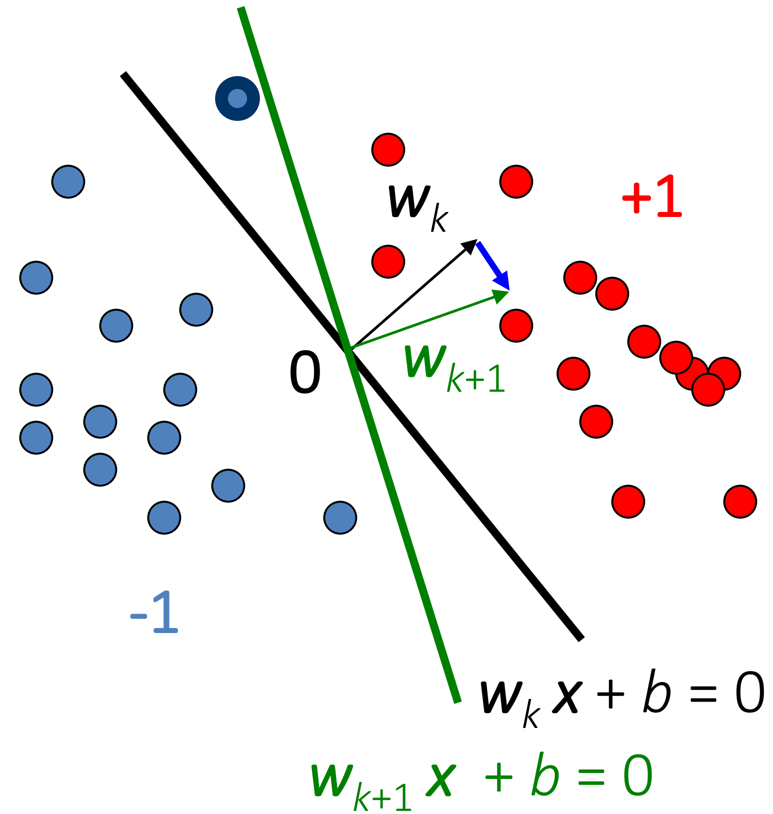
Perceptron algorithm

- Initialize: $\boldsymbol{w}_1 = \mathbf{0}$ in R^n
- Updating rule
- For each data point $\boldsymbol{x}$
  - Predict the label $y' = \text{sgn}\{\boldsymbol{w}^T\boldsymbol{x}\}$
  - if $y' \neq y$, update the weight vector

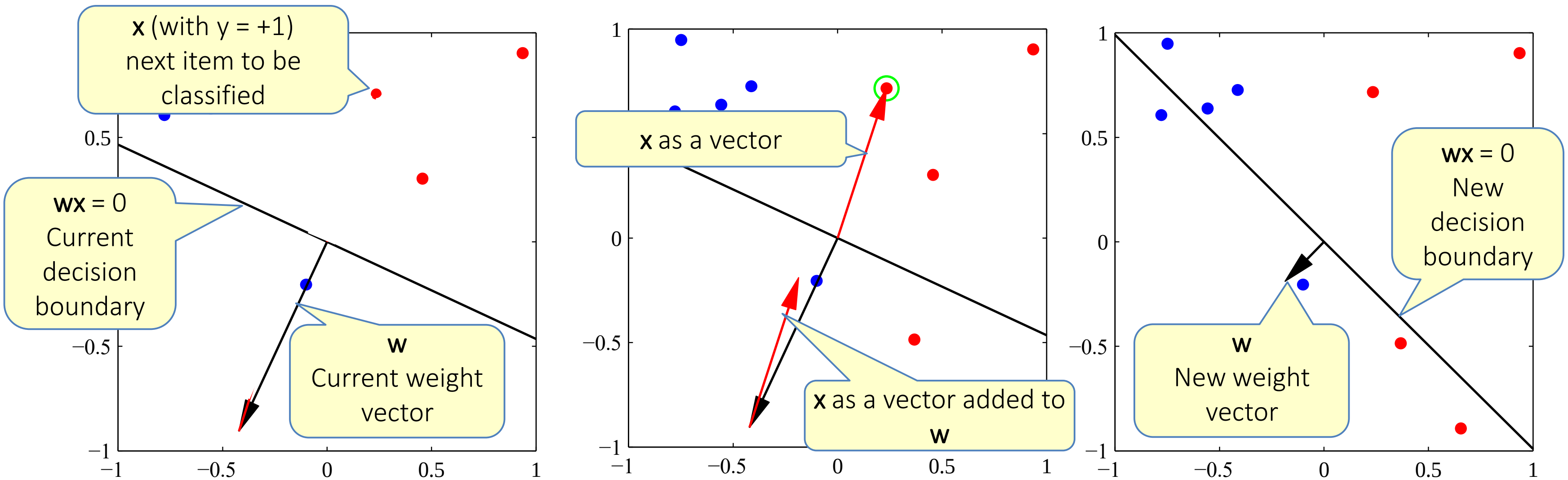
    $\boldsymbol{w}_{k+1} \leftarrow \boldsymbol{w}_k + a\, y_i \boldsymbol{x}_i$

    (a - a constant, learning rate)
  - else

    $\boldsymbol{w}_{k+1} \leftarrow \boldsymbol{w}_k$
- Function: $y' = \text{sgn}\{\boldsymbol{w}^T\boldsymbol{x}\}$
  - if $\boldsymbol{w}^T\boldsymbol{x} > 0$ return +1
  - else return -1



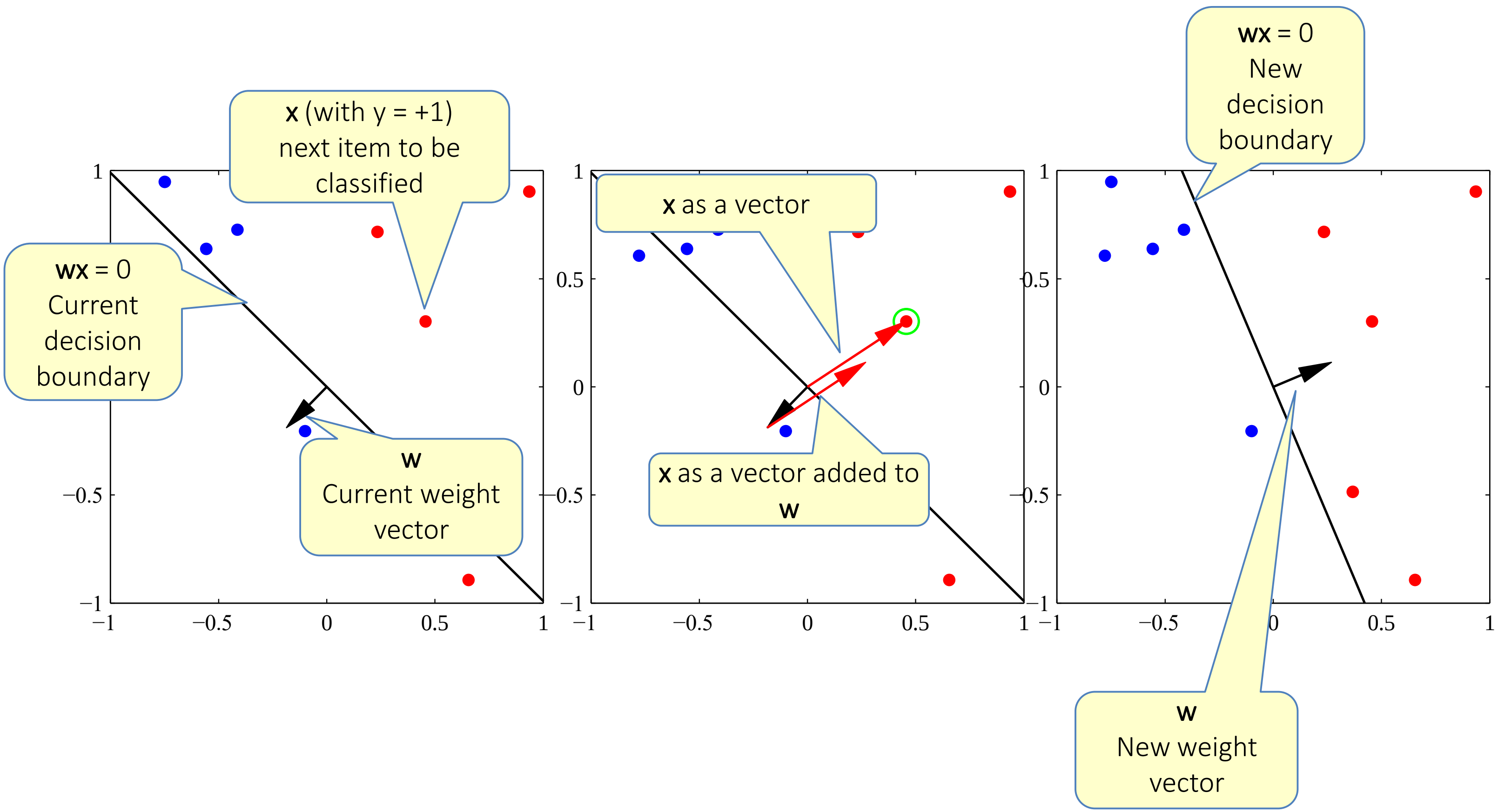
Perceptron in action



(Figures from Bishop 2006)

Positive
Negative

Perceptron in action



(Figures from Bishop 2006)

Perceptron Convergence

- Perceptron Convergence Theorem:
  – If there exist a set of weights that are consistent with the data (i.e., the data is linearly separable), the perceptron learning algorithm will converge
    - How long would it take to converge ?

- Perceptron Cycling Theorem:
  – If the training data is not linearly separable the perceptron learning algorithm will eventually repeat the same set of weights and therefore enter an infinite loop.
    - How to provide robustness, more expressivity ?

Perceptron-Mistake Bound

- Assume that a weight vector $\boldsymbol{w} \in \mathrm{R}^d$. Upon receiving an example $\boldsymbol{x} \in \mathrm{R}^N$, we predict according to a linear threshold function.
- **Theorem [Novikoff,1963]**
  - Let $(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_N, y_N)$ be a sequence of labeled examples with $\boldsymbol{x}_i \in R^d$, $||\boldsymbol{x}|| \leq r$, and $y_i \in \{-1, +1\}$ for all i.
  - Let $\boldsymbol{u} \in \mathrm{R}^N, \gamma > 0$ be such that $||\boldsymbol{u}|| = 1$ and $y_i \cdot \boldsymbol{u}^{\mathrm{T}} \boldsymbol{x}_{\boldsymbol{i}} \geq \gamma$ for all i.
  - Perceptron makes at most $\frac{r^2}{\gamma^2}$ mistakes on this example sequence.
- Assumptions:
  - This theorem assumes that all examples are bounded by some r; for all $\boldsymbol{x}_{\boldsymbol{i}}$, find the largest one, and r is at least this size.
  - The theorem further assumes that there exists some $\boldsymbol{u}$ that separates the data.
  - Requiring that $||\boldsymbol{u}|| = 1$ is simply a constant that could be arbitrarily scaled.
  - Finally, the theorem assumes that there exists some γ such that the inequality is satisfied.
    - We refer to as the complexity parameter: γ is very large, finding a hyperplane is much easier

Proof

- Let $\boldsymbol{w_k}$ be the hypothesis before the kth mistake.
- Assume that the kth mistake occurs on the input example $(\boldsymbol{x}_i, y_i)$

$$\therefore y_i\left(\boldsymbol{w}_k^{\mathrm{T}}\boldsymbol{x}_i\right) \leq 0$$

$$\boldsymbol{w_{k+1}} = \boldsymbol{w_k} + y_i\boldsymbol{x}_i$$

$$\begin{aligned}\boldsymbol{w}_{k+1}^{\mathrm{T}}\boldsymbol{u} = \boldsymbol{w}_k^{\mathrm{T}}\boldsymbol{u} + y_i\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{u} &\geq \boldsymbol{w}_k^{\mathrm{T}}\boldsymbol{u} + \gamma \quad (\because y_i\boldsymbol{u}^{\mathrm{T}}\boldsymbol{x_i} \geq \gamma) \\ &\geq \boldsymbol{w}_{k-1}^{\mathrm{T}}\boldsymbol{u} + 2\gamma \\ &\cdots \\ &\geq k\gamma\end{aligned}$$

$$\begin{aligned}\left|\left|\boldsymbol{w}_{k+1}\right|\right|^2 &= \left|\left|\boldsymbol{w}_k\right|\right|^2 + 2y_i\boldsymbol{w}_k^T\boldsymbol{x}_i + \left|\left|\boldsymbol{x}_2\right|\right|^2 \\ &\leq \left|\left|\boldsymbol{w}_k\right|\right|^2 + r^2 \quad (\because y_i\left(\boldsymbol{w}_k^{\mathrm{T}}\boldsymbol{x}_i\right) \leq 0) \\ &\cdots \\ &\leq kr^2\end{aligned}$$

Therefore, $\sqrt{k}r \geq \left|\left|\boldsymbol{w}_{k+1}\right|\right| \geq \boldsymbol{w}_{k+1}^{\mathrm{T}}\boldsymbol{u} \geq k\gamma$

(Second inequality: $\boldsymbol{u}^{\mathrm{T}}\boldsymbol{v} = ||\boldsymbol{u}|| \cdot ||\boldsymbol{v}|| \cdot cos(\boldsymbol{u}, \boldsymbol{v}) \leq ||\boldsymbol{u}|| \cdot ||\boldsymbol{v}|| \leq ||\boldsymbol{v}||$ and $||\boldsymbol{u}|| = 1$)

$$\therefore k \leq \frac{r^2}{\gamma^2}$$

- If you want to further read
  - https://www.cse.iitb.ac.in/~shivaram/teaching/old/cs344+386-s2017/resources/classnote-1.pdf

Practical Issues and Extensions

- There are many extensions that can be made to these basic algorithms.
- Some are necessary for them to perform well
  - Regularization (later soon)
- Some are for ease of use and tuning
  - Converting the output of a Perceptron to a conditional probability

$$P(y = +1 \mid x) = [1 + \exp(-a\, wx)]^{-1}$$

  - Can tune the hyper-parameter a
- Multiclass classification

Regularization Via Averaged Perceptron

- An Averaged Perceptron Algorithm is motivated by the following considerations:
  - In the mistake bound model: We don't know when we will make the mistakes.

- Averaged Perceptron returns a weighted average of a number of earlier hypotheses; the weights are a function of the length of no-mistakes stretch.

Regularization Via Averaged Perceptron

- **Training:**

[m: #(examples); k: #(mistakes) = #(hypotheses); c_k: consistency count for $\boldsymbol{w}_k$]

- Input: a labeled training set $\{(\boldsymbol{x}_1, y_1),\ldots(\boldsymbol{x}_m, y_m)\}$
- Number of epochs T
- Output: a list of weighted perceptrons $\{(\boldsymbol{w}_1, c_1),\ldots,(\boldsymbol{w}_k, c_k)\}$
- Initialize: k=0; $\boldsymbol{w}_1 = 0$, $\boldsymbol{c}_1 = 0$
- Repeat T times:
  - For i =1,…m:
  - Compute prediction $y' = \mathrm{sign}(\boldsymbol{w}_k^T x_i)$
  - If $y' = y$, then $\boldsymbol{c}_k = \boldsymbol{c}_k + 1$

    else: $\boldsymbol{w}_{k+1} = \boldsymbol{w}_k + \boldsymbol{w}_i\, \boldsymbol{x}$; $\boldsymbol{c}_{k+1} = 1$; $k = k+1$
- **Prediction:**
- Given: a list of weighted perceptrons $\{(\boldsymbol{w}_1, c_1),\ldots(\boldsymbol{w}_k, c_k)\}$; a new example $\boldsymbol{x}$

  Predict the label($\boldsymbol{x}$) as follows:

  $$y(\boldsymbol{x}) = \mathrm{sign}\left[\sum_{1,k} c_k\, \mathrm{sign}(\boldsymbol{w}_k^{T} \boldsymbol{x}) \right]$$

Perceptron algorithm

- Online: can adjust to changing target, over time
- Advantages
  - Simple and computationally efficient
  - Guaranteed to learn a linearly separable problem (convergence, global optimum)
- Limitations
  - Only linear separations
  - Only converges for linearly separable data
  - Not really “efficient with many features”