Linear Classifiers

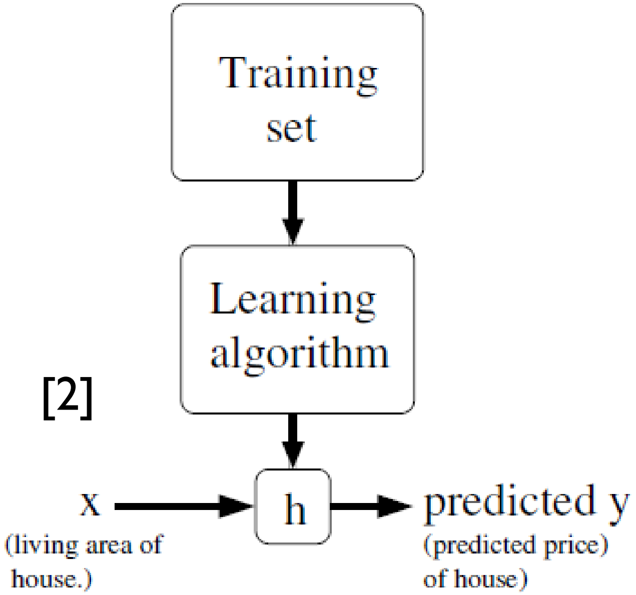
CE-477: Machine Learning - CS-828: Theory of Machine Learning
Sharif University of Technology
Fall 2024

Fatemeh Seyyedsalehi

A supervised problem

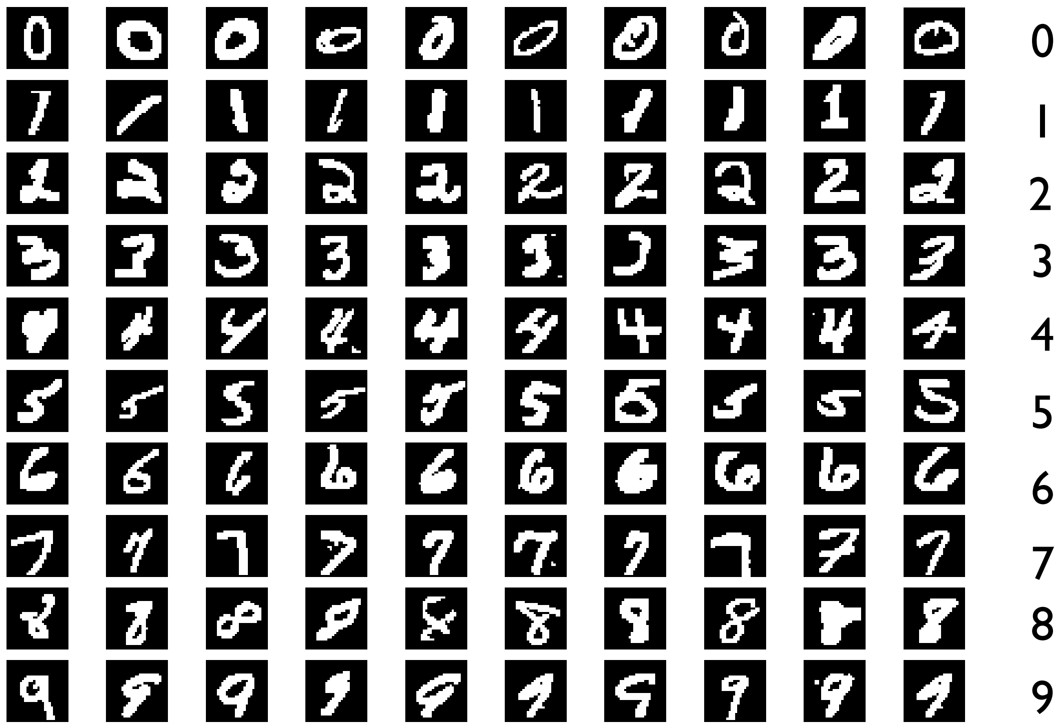
- Our goal is to learn a function $h : \mathcal{X} \rightarrow \mathcal{Y}$
- $h(x)$ should be a good predictor for the corresponding y
- h is called a **hypothesis**

[2]



Classification problem

- The values y takes on only a small number of discrete values.
  - A spam classifier for emails (0, 1)
  - Handwritten digit recognition



[2]

Classification problem

- Given: Training set
  - labeled set of N input-output pairs $D = \left\{\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)\right\}_{i=1}^{N}$
  - $y \in \{1, \dots, K\}$
    - Two classes (binary): $y \in \{0, 1\}$
    - Multi classes: representing outputs by one-hot vectors,

$$y = [0, 1, 0, 0, 0]$$

- Goal: Given an input $\boldsymbol{x}$, assign it to one of K classes

- Discriminant function: takes an input vector $\boldsymbol{x}$ and directly assigns it to one of K classes, denoted C_k.

Linear classifiers

- The hypothesis space:
  - The input space is divided into *decision regions* whose boundaries are called *decision boundaries* or *decision surfaces*.
  - Decision surfaces are linear functions of the input vector $\boldsymbol{x}$
    - Defined by (d-1)-dimensional hyperplanes within the *d*-dimensional input space.

- Linearly separable data: data points that can be exactly classified by a linear decision surface.

- Even when they are not optimal, we can use the simplicity of linear classifiers
  - Easy to compute
  - In the absence of information suggesting otherwise, linear classifiers are an attractive candidates for initial, trial classifiers.

Binary classification

- $h(\boldsymbol{x}; \boldsymbol{w}) = \boldsymbol{w}^T \boldsymbol{x} + w_0 = w_0 + w_1 x_1 + \ldots + w_d x_d$
  - $\boldsymbol{x} = [x_1\ x_2\ \ldots x_d]$
  - $\boldsymbol{w} = [w_1\ w_2\ \ldots w_d]$
  - w_0: bias

- The linear discriminant function:
  - if $\boldsymbol{w}^T \boldsymbol{x} + w_0 \geq 0$ then $\mathcal{C}_1$ else $\mathcal{C}_2$

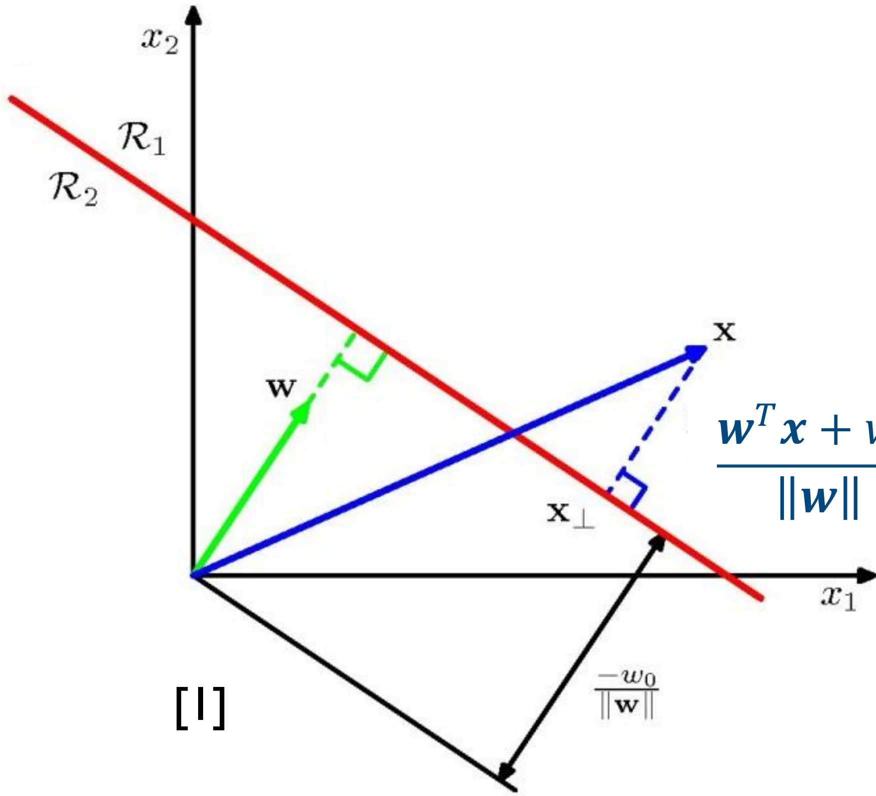
- Decision surface (boundary): $\boldsymbol{w}^T \boldsymbol{x} + w_0 = 0$

Linear boundary: geometry

$$\boldsymbol{w}^T\boldsymbol{x} + w_0 > 0$$

$$\boldsymbol{w}^T\boldsymbol{x} + w_0 = 0$$

$$\boldsymbol{w}^T\boldsymbol{x} + w_0 < 0$$



$$\frac{\boldsymbol{w}^T\boldsymbol{x} + w_0}{\|\boldsymbol{w}\|}$$

[I]

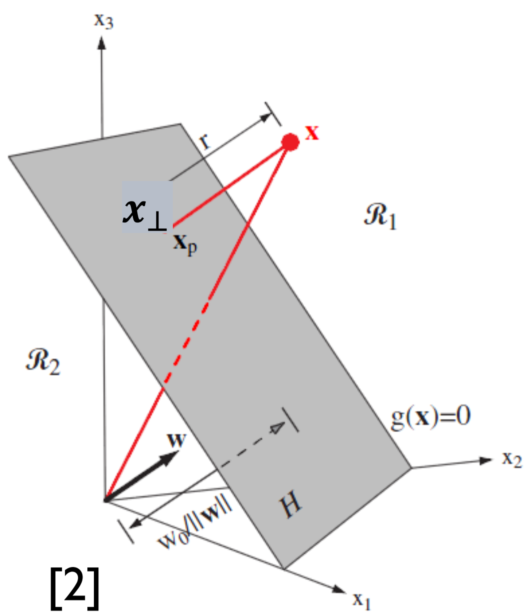
Linear classifier: Two classes

- Decision boundary is a $(d-1)$-dimensional hyperplane H in the d-dimensional feature space
  - The orientation of H is determined by the normal vector $[w_1, \dots, w_d]$
  - w_0 determines the location of the surface.
    - The normal distance from the origin to the decision surface is $\frac{w_0}{\|\boldsymbol{w}\|}$

$$\boldsymbol{x} = \boldsymbol{x}_\perp + r\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}$$

$$\boldsymbol{w}^T\boldsymbol{x} + w_0 = r\|\boldsymbol{w}\| \Rightarrow r = \frac{\boldsymbol{w}^T\boldsymbol{x} + w_0}{\|\boldsymbol{w}\|}$$

gives a signed measure of the perpendicular distance r of the point $\boldsymbol{x}$ from the decision surface



[2]

Cost Function for linear classifiers

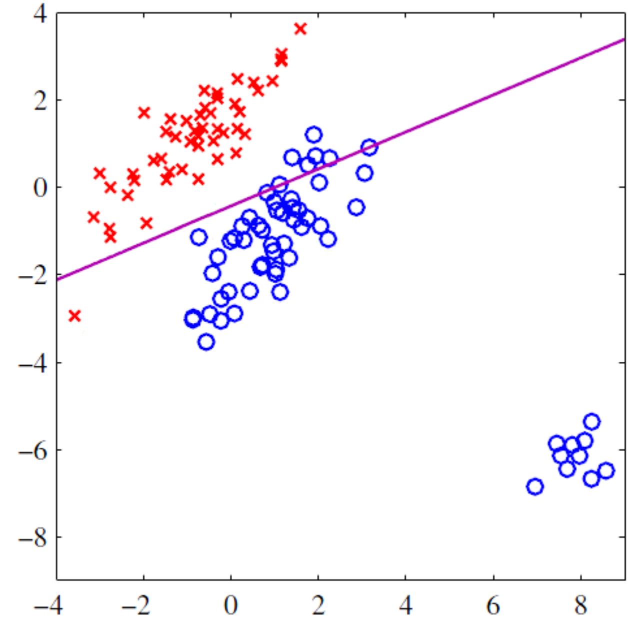
- Finding linear classifiers can be formulated as an optimization problem:
  - Select how to measure the prediction loss
    - Based on the training set $D = \left\{\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)\right\}_{i=1}^{n}$, a cost function $J(\boldsymbol{w})$ is defined
  - Solve the resulting optimization problem to find best parameters:
    - Find optimal $\widehat{\boldsymbol{w}} = \underset{\boldsymbol{w}}{\operatorname{argmin}}\, J(\boldsymbol{w})$
- Criterion or cost functions for classification:
  - We will investigate several cost functions for the classification problem

Cost Function for linear classifiers

- SSE as the linear regression
  - Is not suitable for classification
  - Penalizes 'too correct' predictions (that they lie a long way on the correct side of the decision)

$$J(\boldsymbol{w}) = \sum_{i=1}^{N} \left(\boldsymbol{w}^T \boldsymbol{x}^{(i)} - y^{(i)} \right)^2$$

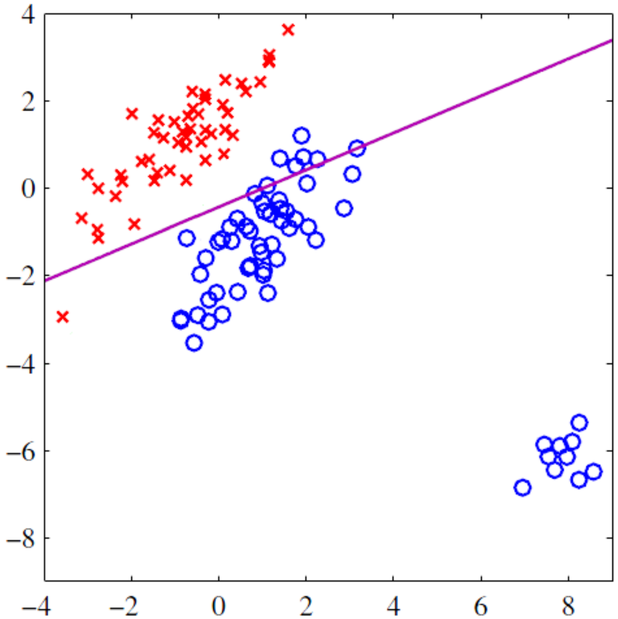
$y \in \{-1, +1\}$



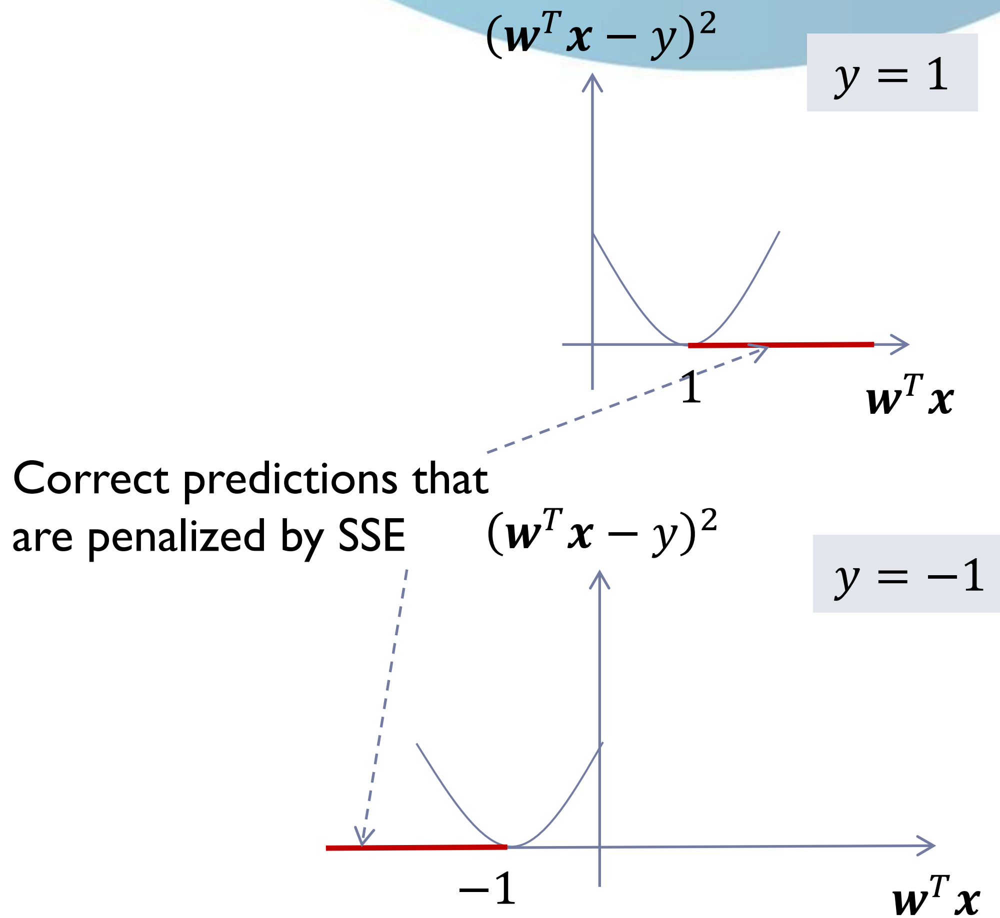
[1]

SSE cost function for classification

$K = 2$



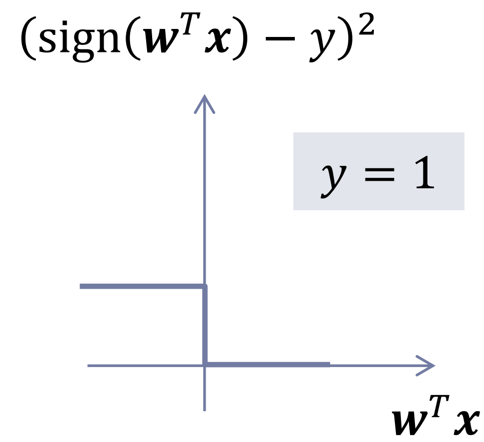
[1]



Cost Function for linear classifiers

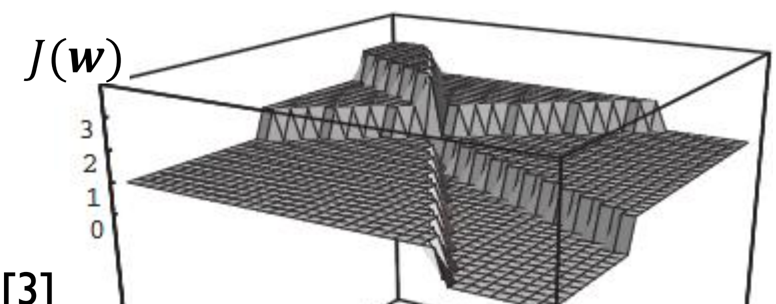
- Is it more suitable if we set $\mathrm{h}(\boldsymbol{x}) = g(\boldsymbol{w}^T\boldsymbol{x})$?

$$J(\boldsymbol{w}) = \sum_{i=1}^{N}\left(\textcolor{red}{\mathrm{sign}}\left(\boldsymbol{w}^T\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2$$



$$\mathrm{sign}(z) = \begin{cases} -1, & z < 0 \\ 1, & z \geq 0 \end{cases}$$

- $J(\boldsymbol{w})$ is a piecewise constant function shows the number of misclassifications

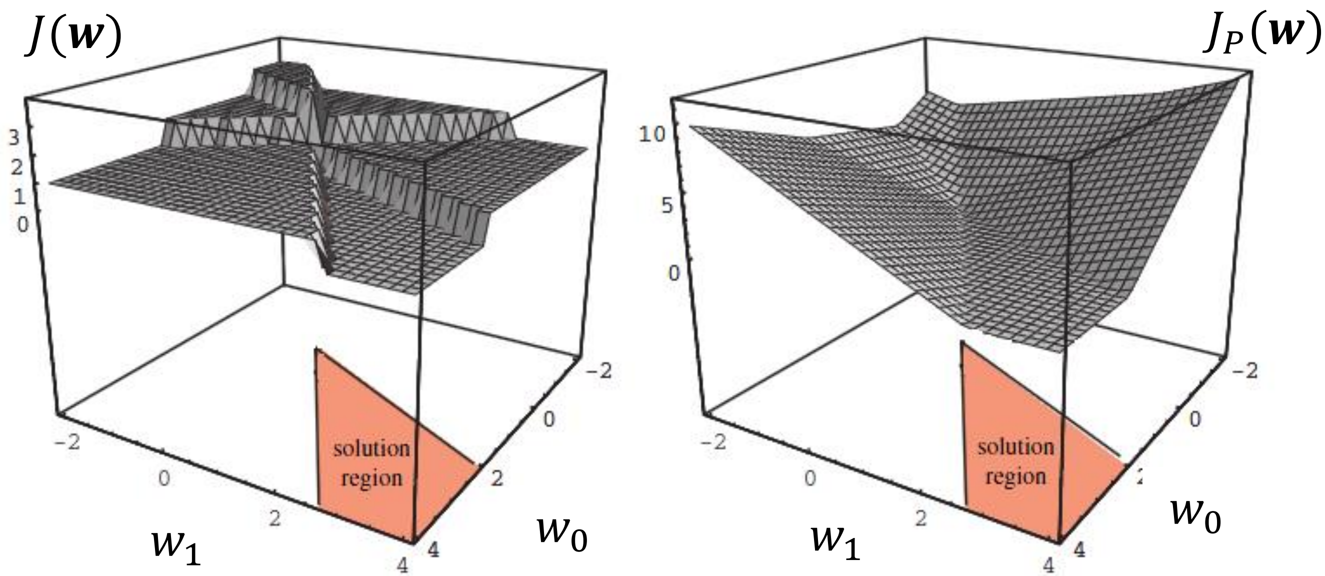


[3]

Perceptron criterion

$$J_P(\boldsymbol{w}) = -\sum_{i \in \mathcal{M}} \boldsymbol{w}^T \boldsymbol{x}^{(i)} y^{(i)}$$

- $\mathcal{M}$: subset of training data that are misclassified



[3]

# of misclassifications as a cost function

Perceptron's cost function

Batch Perceptron

- "Gradient Descent" to solve the optimization problem:

$$\boldsymbol{w}^{t+1} = \boldsymbol{w}^{t} - \eta \nabla_{\boldsymbol{w}} J_P(\boldsymbol{w}^{t})$$

$$\nabla_{\boldsymbol{w}} J_P(\boldsymbol{w}) = -\sum_{i \in \mathcal{M}} \boldsymbol{x}^{(i)} y^{(i)}$$

- Batch Perceptron converges in finite number of steps for linearly separable data:

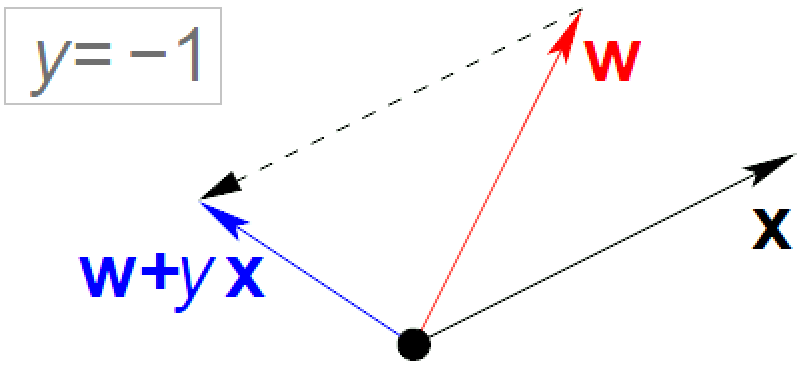
Stochastic gradient descent for Perceptron

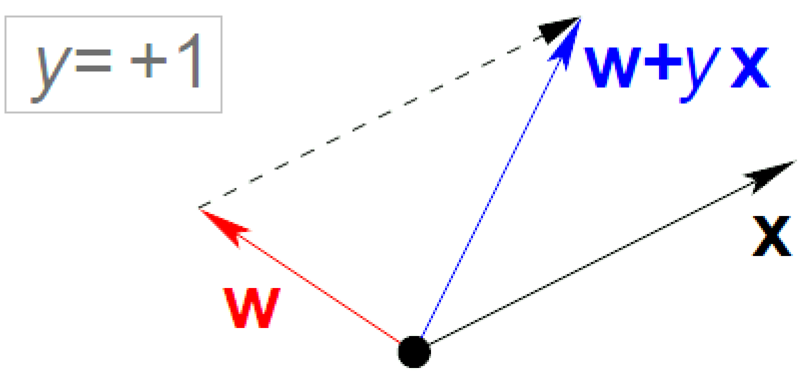
- Single-sample perceptron:
  - If $\boldsymbol{x}^{(i)}$ is misclassified:

$$\boldsymbol{w}^{t+1} = \boldsymbol{w}^{t} + \eta \boldsymbol{x}^{(i)} y^{(i)}$$

- Perceptron convergence theorem: for linearly separable data
  - If training data are linearly separable, the single-sample perceptron is also guaranteed to find a solution in a finite number of steps

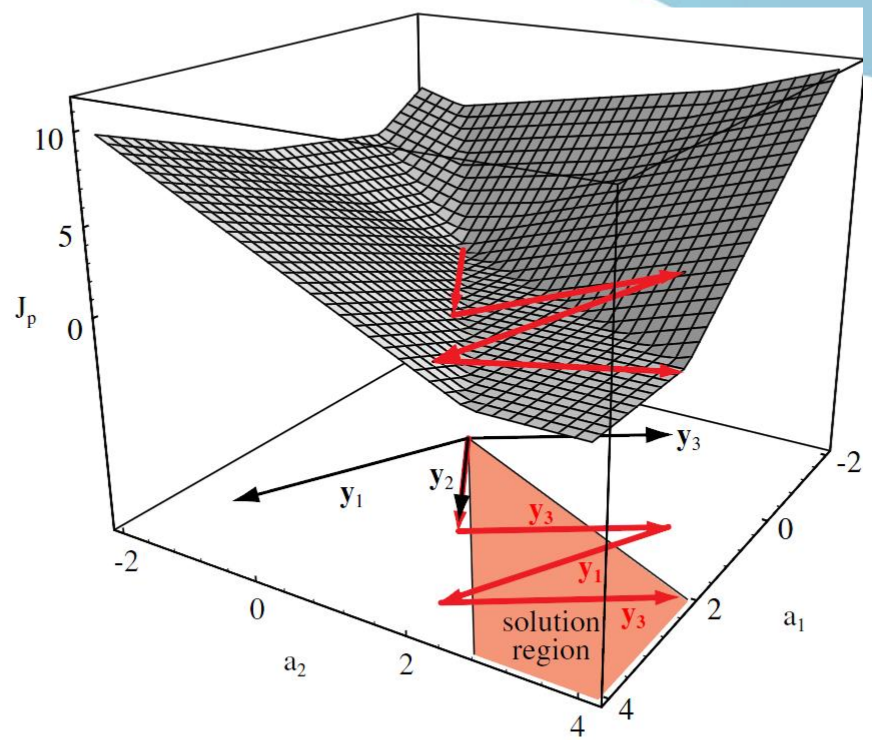
Stochastic gradient descent for Perceptron





[2]

Convergence of Perceptron



[3]

- For data sets that are not linearly separable, the single-sample perceptron learning algorithm will never converge

Pocket algorithm

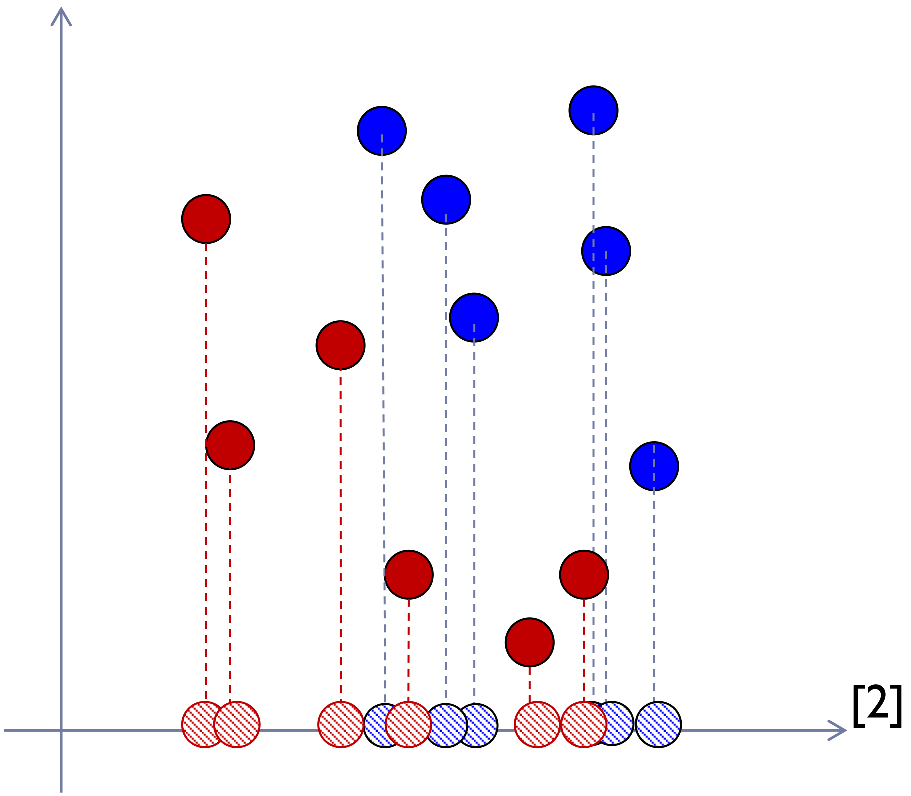
- For the data that are not linearly separable due to noise:
  - Keeps in its pocket the best $\boldsymbol{w}$ encountered up to now.

Initialize $\boldsymbol{w}$
for $t = 1, \dots, T$
    $i \leftarrow t \bmod N$
    if $\boldsymbol{x}^{(i)}$ is misclassified then
        $\boldsymbol{w}^{new} = \boldsymbol{w} + \boldsymbol{x}^{(i)} y^{(i)}$
        if $E_{train}(\boldsymbol{w}^{new}) < E_{train}(\boldsymbol{w})$ then
            $\boldsymbol{w} = \boldsymbol{w}^{new}$
end

$$E_{train}(\boldsymbol{w}) = \frac{1}{N} \sum_{n=1}^{N} \left[sign(\boldsymbol{w}^T \boldsymbol{x}^{(n)}) \neq y^{(n)} \right]$$

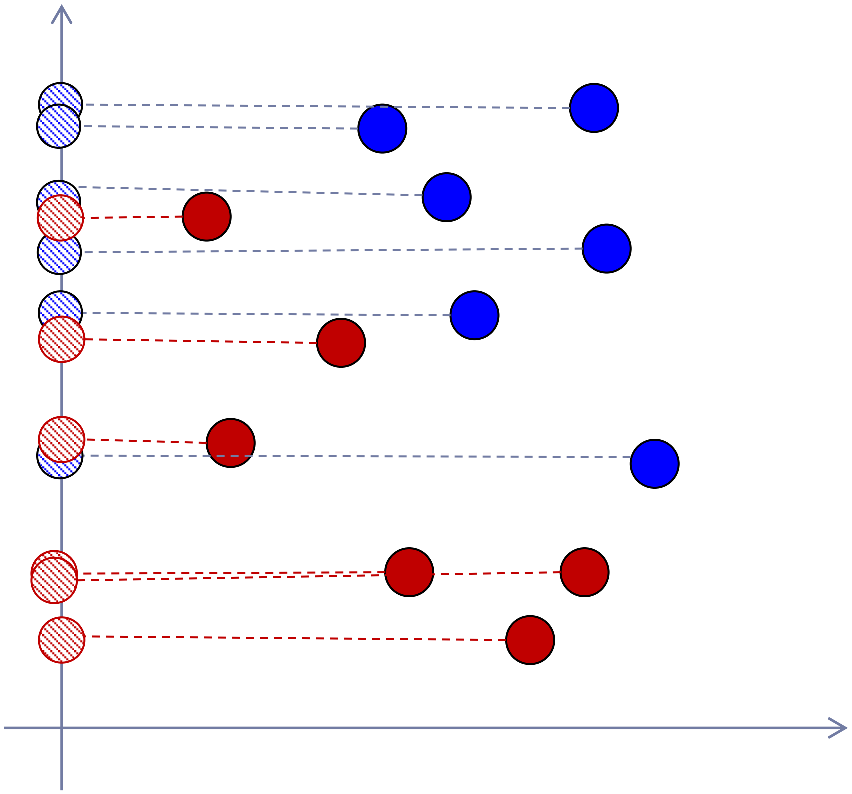
Good Projection for Classification

- What is a good criterion?
  - Separating different classes in the projected space



Good Projection for Classification

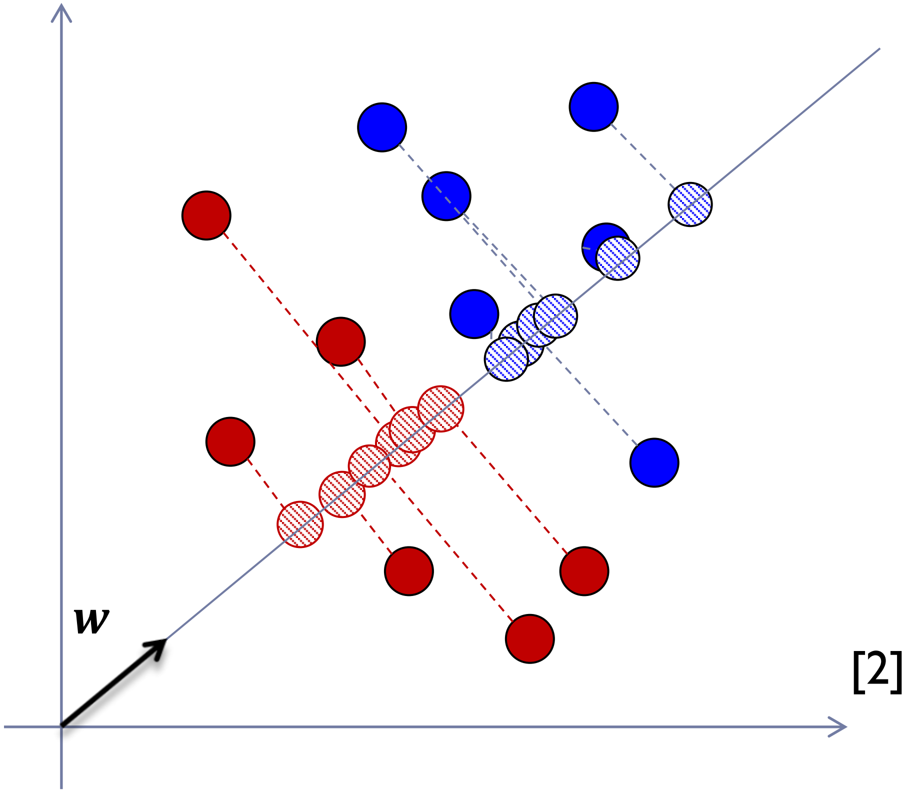
- What is a good criterion?
  - Separating different classes in the projected space



[2]

Good Projection for Classification

- What is a good criterion?
  - Separating different classes in the projected space



LDA Problem

- Fisher's Linear Discriminant Analysis
- Problem definition:
  - $K = 2$ classes
  - $\{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$ training samples with N_1 samples from the first class ($\mathcal{C}_1$) and N_2 samples from the second class ($\mathcal{C}_2$)
  - Goal: finding the best direction $\boldsymbol{w}$ that we hope to enable accurate classification

- The projection of sample $\boldsymbol{x}$ onto a line in direction $\boldsymbol{w}$ is $\boldsymbol{w}^T\boldsymbol{x}$

- What is the measure of the separation between the projected points of different classes?

Measure of Separation in the Projected Direction

- The direction of the line jointing the class means is the solution of the following problem:
  - Maximizes the separation of the projected class means

$$\max_{\boldsymbol{w}} J(\boldsymbol{w}) = (\mu_1' - \mu_2')^2$$
$$\text{s.t. } \|\boldsymbol{w}\| = 1$$

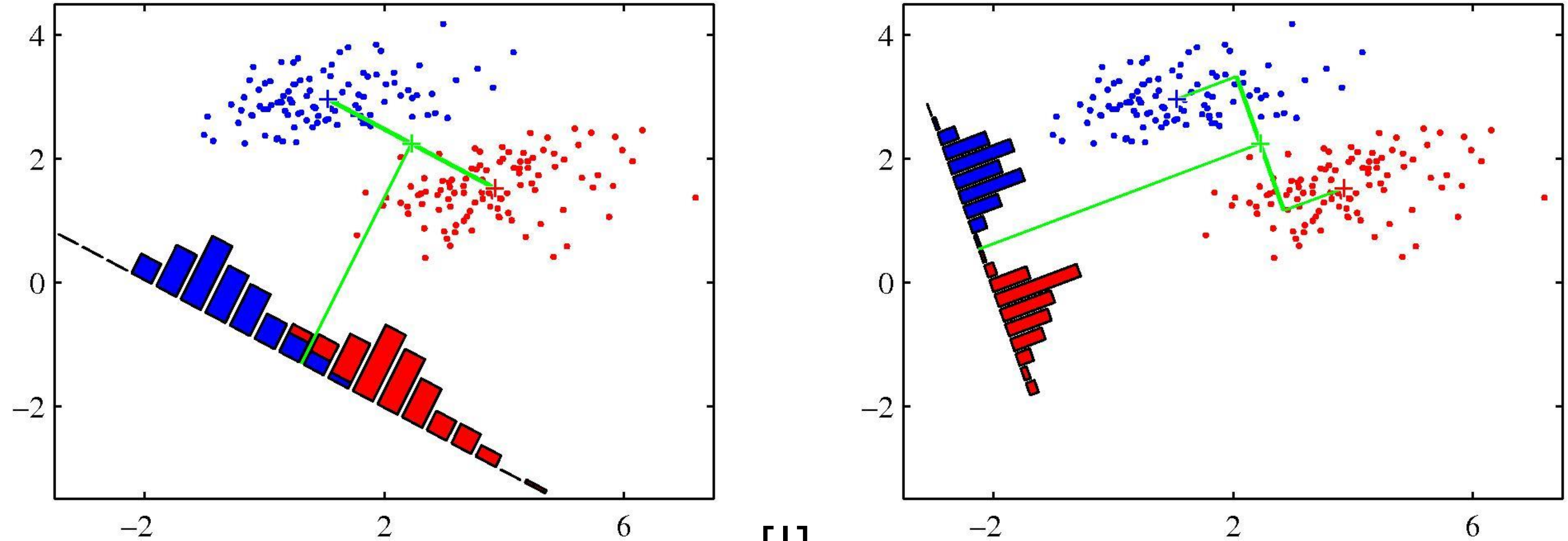
$$\mu_1' = \boldsymbol{w}^T \boldsymbol{\mu}_1 \qquad \boldsymbol{\mu}_1 = \frac{\sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_1} \boldsymbol{x}^{(i)}}{N_1}$$

$$\mu_2' = \boldsymbol{w}^T \boldsymbol{\mu}_2 \qquad \boldsymbol{\mu}_2 = \frac{\sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_2} \boldsymbol{x}^{(i)}}{N_2}$$

- What is the problem with the criteria considering only $|\mu_1' - \mu_2'|$?
  - It does not consider the variances of the classes in the projected direction

Measure of Separation in the Projected Direction

- Is the direction of the line jointing the class means a good candidate for $\boldsymbol{w}$?



[1]

LDA Criteria

- Fisher idea: maximize a function that will give
  - large separation between the projected class means
  - while also achieving a small variance within each class, thereby minimizing the class overlap.

$$J(\boldsymbol{w}) = \frac{|\mu_1' - \mu_2'|^2}{s_1'^2 + s_2'^2}$$

LDA Criteria

- The scatters of projected data are:

$$s_1'^2 = \sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_1} \left(\boldsymbol{w}^T \boldsymbol{x}^{(i)} - \boldsymbol{w}^T \boldsymbol{\mu}_1\right)^2$$

$$s_2'^2 = \sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_2} \left(\boldsymbol{w}^T \boldsymbol{x}^{(i)} - \boldsymbol{w}^T \boldsymbol{\mu}_1\right)^2$$

LDA Criteria

$$J(\boldsymbol{w}) = \frac{|\mu_1' - \mu_2'|^2}{s_1'^2 + s_2'^2}$$

$$|\mu_1' - \mu_2'|^2 = |\boldsymbol{w}^T \boldsymbol{\mu}_1 - \boldsymbol{w}^T \boldsymbol{\mu}_2|^2 = \boldsymbol{w}^T (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T \boldsymbol{w}$$

$$s_1'^2 = \sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_1} \left(\boldsymbol{w}^T \boldsymbol{x}^{(i)} - \boldsymbol{w}^T \boldsymbol{\mu}_1\right)^2 = \boldsymbol{w}^T \left(\sum_{\boldsymbol{x}^{(i)} \in \mathcal{C}_1} \left(\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_1\right)\left(\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_1\right)^T \right) \boldsymbol{w}$$

LDA Criteria

$$J(\boldsymbol{w}) = \frac{\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w}}{\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w}}$$

Between-class scatter matrix ← $\boldsymbol{S}_B = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T$

Within-class scatter matrix ← $\boldsymbol{S}_W = \boldsymbol{S}_1 + \boldsymbol{S}_2$

$$\boldsymbol{S}_1 = \sum_{\boldsymbol{x}^{(i)} \in C_1} \left(\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_1\right)\left(\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_1\right)^T$$

$$\boldsymbol{S}_2 = \sum_{\boldsymbol{x}^{(i)} \in C_2} \left(\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_2\right)\left(\boldsymbol{x}^{(i)} - \boldsymbol{\mu}_2\right)^T$$

scatter matrix=N×covariance matrix

LDA Criteria

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}}$$

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = \frac{\frac{\partial \mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\partial \mathbf{w}} \times \mathbf{w}^T \mathbf{S}_W \mathbf{w} - \frac{\partial \mathbf{w}^T \mathbf{S}_W \mathbf{w}}{\partial \mathbf{w}} \times \mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\left(\mathbf{w}^T \mathbf{S}_W \mathbf{w}\right)^2} = \frac{\left(2\mathbf{S}_B \mathbf{w}\right)\mathbf{w}^T \mathbf{S}_W \mathbf{w} - \left(2\mathbf{S}_W \mathbf{w}\right)\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\left(\mathbf{w}^T \mathbf{S}_W \mathbf{w}\right)^2}$$

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{S}_B \mathbf{w} = \lambda \mathbf{S}_W \mathbf{w}$$

LDA Derivation

If $\boldsymbol{S}_W$ is full-rank

$$\boldsymbol{S}_B\boldsymbol{w} = \lambda\boldsymbol{S}_W\boldsymbol{w} \Longrightarrow \boldsymbol{S}_W^{-1}\boldsymbol{S}_B\boldsymbol{w} = \lambda\boldsymbol{w}$$

- $\boldsymbol{S}_B\boldsymbol{w}$ (for any vector $\boldsymbol{w}$) points in the same direction as $\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2$:

$$\boldsymbol{S}_B\boldsymbol{w} = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T\boldsymbol{w} \propto (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$$

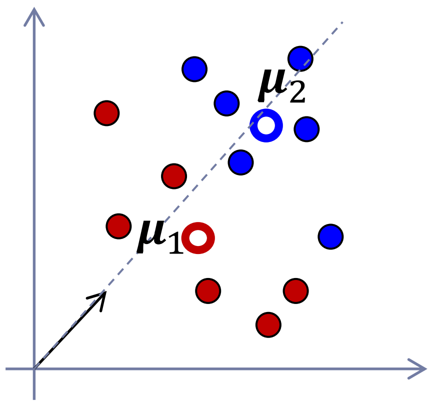
$$\boldsymbol{w} \propto \boldsymbol{S}_W^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$$

- Thus, we can solve the eigenvalue problem immediately

LDA Algorithm

- Find $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$ as the mean of class 1 and 2 respectively
- Find $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$ as scatter matrix of class 1 and 2 respectively
- $\boldsymbol{S}_W = \boldsymbol{S}_1 + \boldsymbol{S}_2$

- Classification
  - $\boldsymbol{w} = \boldsymbol{S}_W^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$
  - Using a threshold on $\boldsymbol{w}^T\boldsymbol{x}$, we can classify $\boldsymbol{x}$



[2]

References

- [1]: C. Bishop, “Pattern Recognition and Machine Learning”, Chapter 4.1.
- [2]: Mahdieh Soleymani, Machine learning, Sharif university of technology
- [3]: Pattern classification, Duda, Hart & Stork, 2002