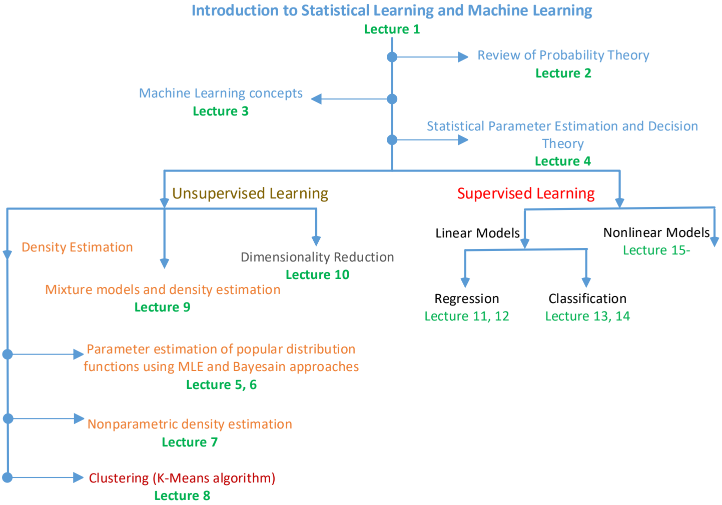
Statistical Learning and Machine Learning

Lecture 13 - Linear Models for Classification 1

October 10, 2021

Course overview and where do we stand



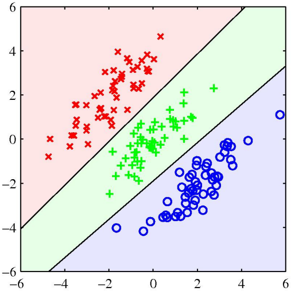
Objectives of the lecture

- Linear models for classification
  - Discriminant functions
  - Least squares for classification
  - The perceptron algorithm

Linear classification

The goal of classification is to assign an input vector $\boldsymbol{x}$ to one of the K classes $\mathcal{C}_k$, $k = 1, \ldots, K$:

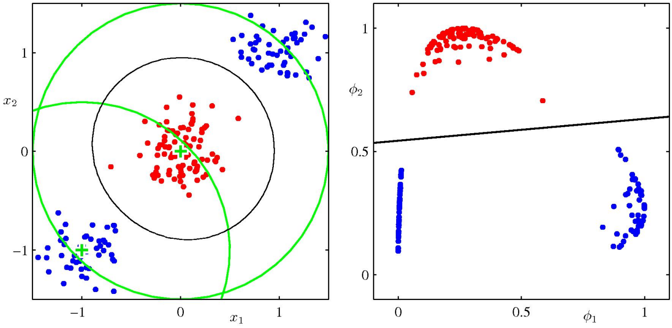
- The input space is divided into K decision regions, separated by decision boundaries or decision surfaces
- **Linear models**: are those for which decision boundaries are linear functions of $\boldsymbol{x}$ i.e., $\boldsymbol{w}^T\boldsymbol{x} + w_0$



Decision boundaries for a linearly separable data set

Generalized linear classification

- Generalized linear classification corresponds to using linear combination of (nonlinearly) transformed vectors $\phi(\boldsymbol{x})$ i.e., $y(\boldsymbol{x}) = f(\boldsymbol{w}^T\phi(\boldsymbol{x}))$.
- Appropriately chosen linear models of basis functions $\phi(\boldsymbol{x})$ can be equivalent to nonlinear models in $\boldsymbol{x}$



Three approaches to classification

1. Finding a function $y(\boldsymbol{x})$ called discriminant function which maps a new input $\boldsymbol{x}_*$ onto a class label.
2. **Generative models:** Determining class-conditional densities $p(\boldsymbol{x}|\mathcal{C}_k)$ or joint distributions $p(\boldsymbol{x}, \mathcal{C}_k)$, followed by the estimation of posterior densities:

$$p(\mathcal{C}_k|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\boldsymbol{x})} = \frac{p(\boldsymbol{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{\sum_j p(\boldsymbol{x}|\mathcal{C}_j)p(\mathcal{C}_j)}$$

   Finally, decision theory is used to determine class membership for new $\boldsymbol{x}$.
3. **Discriminative models:** Obtaining posterior class probabilities $p(\mathcal{C}_k|\boldsymbol{x})$ directly, followed by discrimination.

Discriminant Functions: Two classes

Linear discriminant function:

$$y(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + w_0 \tag{1}$$

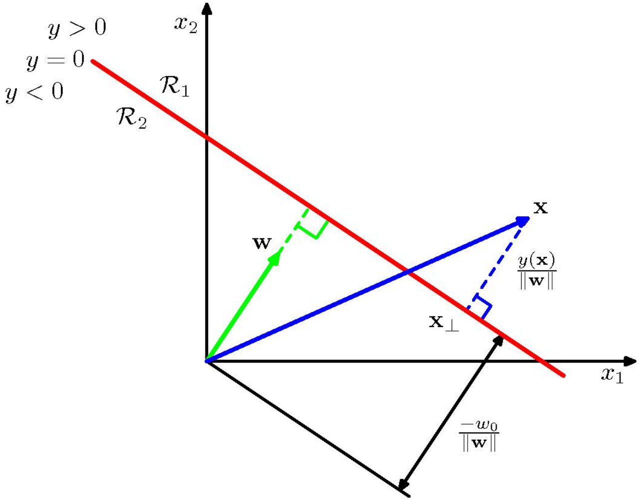
$\boldsymbol{w}$ is called *weight vector* and w_0 is called *bias* ($-w_0$ is called *threshold*).

Classification rule:

- Assign $\boldsymbol{x}$ to class $\mathcal{C}_1$ if $y(\boldsymbol{x}) \geq 0$
- Assign $\boldsymbol{x}$ to class $\mathcal{C}_2$ if $y(\boldsymbol{x}) < 0$

The decision boundary at $y(\boldsymbol{x}) = 0$, which corresponds to a $(D-1)$-dimensional hyperplane in $\mathbb{R}^D$.

Discriminant Functions: Two classes



Discriminant Functions: Two classes

- $\boldsymbol{w}$ is orthogonal to any vector lying on the decision surface. How? Consider two arbitrary points $\boldsymbol{x}_A$ and $\boldsymbol{x}_B$ on the decision boundary or hyperplane, then

$$y(\boldsymbol{x}_A) = y(\boldsymbol{x}_B) = 0 \Rightarrow \boldsymbol{w}^T(\boldsymbol{x}_A - \boldsymbol{x}_B) = 0. \tag{2}$$

  Thus, $\boldsymbol{w}$ is orthogonal to the vector $(\boldsymbol{x}_A - \boldsymbol{x}_B)$.

- The normal distance from the origin to the decision hyperplane is:

$$\frac{\boldsymbol{w}^T\boldsymbol{x}}{\|\boldsymbol{w}\|} = -\frac{w_0}{\|\boldsymbol{w}\|}. \tag{3}$$

  Thus, w_0 determines the location of the decision hyperplane.

Discriminant Functions: Two classes

- The value of $y(\boldsymbol{x})$ gives a signed measure of the perpendicular distance r of $\boldsymbol{x}$ to the decision hyperplane:
  - Let $\boldsymbol{x}$ be an arbitrary vector and $\boldsymbol{x}_\perp$ be its orthogonal projection to the decision hyperplane:

$$\boldsymbol{x} = \boldsymbol{x}_\perp + r\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}. \tag{4}$$

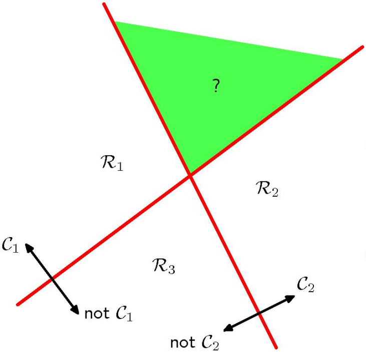
  - We multiply with $\boldsymbol{w}^T$ and add w_0 both sides
  - We use:
    - $y(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + w_0$
    - $y(\boldsymbol{x}_\perp) = \boldsymbol{w}^T\boldsymbol{x}_\perp + w_0 = 0$
  - Then:

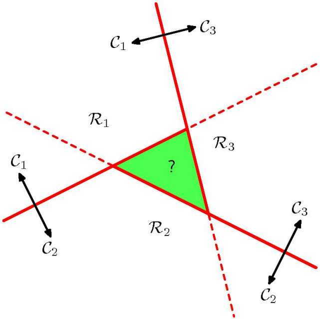
$$r = \frac{y(\boldsymbol{x})}{\|\boldsymbol{w}\|}. \tag{5}$$

Discriminant Functions: $K > 2$ classes

We can extend binary discriminant functions to K-class discriminant functions using two schemes:

- One-versus-rest: Use $K - 1$ binary discriminant functions, each of which separating points in $\mathcal{C}_k$ and not in that class
- One-versus-one: Use $K(K-1)/2$ binary discriminant functions, one for every possible pair of classes $\mathcal{C}_k$, $\mathcal{C}_{j \neq k}$.





Discriminant Functions: $K > 2$ classes

Single K class Discriminant: comprises K linear functions of the form:

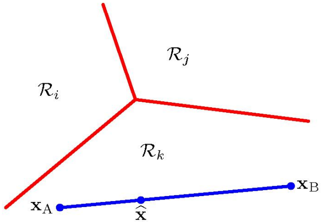
$$y_k(\boldsymbol{x}) = \boldsymbol{w}_k^T \boldsymbol{x} + w_{k0}. \tag{6}$$

The classification rule:

assigning $\boldsymbol{x}$ to class $\mathcal{C}_k$ if $y_k(\boldsymbol{x}) > y_j(\boldsymbol{x})$ for all $j \neq k$.

The decision boundary between $\mathcal{C}_k$ and $\mathcal{C}_j$ is given by $y_k(\boldsymbol{x}) = y_j(\boldsymbol{x})$, corresponding to

$$(\boldsymbol{w}_k - \boldsymbol{w}_j)^T \boldsymbol{x} + (w_{k0} - w_{j0}) = 0. \tag{7}$$



Least squares for classification

Each class $\mathcal{C}_k,\ k = 1, \dots, K$ is described by:

$$y_k(\boldsymbol{x}) = \boldsymbol{w}_k^T \boldsymbol{x} + w_{k0} = \tilde{\boldsymbol{w}}_k^T \tilde{\boldsymbol{x}} \tag{8}$$

where $\tilde{\boldsymbol{w}}_k = [w_{0k}, \boldsymbol{w}_k^T]^T$, and $\tilde{\boldsymbol{x}} = [1, \boldsymbol{x}^T]^T$.

We can group all K outputs together:

$$\boldsymbol{y}(\boldsymbol{x}) = \tilde{\boldsymbol{W}}^T \tilde{\boldsymbol{x}} \tag{9}$$

where $\tilde{\boldsymbol{W}} = [\tilde{\boldsymbol{w}}_1, \dots, \tilde{\boldsymbol{w}}_K]$.

The classification rule is:

assign $\boldsymbol{x}$ to class $\mathcal{C}_k$ if $y_k(\boldsymbol{x}) > y_j(\boldsymbol{x})$ for all $j \neq k$.

Least squares for classification

Problem: Given a training set $\{\boldsymbol{x}_n, \boldsymbol{t}_n\}$ for $n = 1, \ldots, N$, we want to estimate the parameters $\tilde{\boldsymbol{W}}$ of the regression model.

We use the 1-of-K binary coding scheme for $\boldsymbol{t}$. Denoting

$$\tilde{\boldsymbol{X}} = \begin{bmatrix} \tilde{\boldsymbol{x}}_1^T \\ \tilde{\boldsymbol{x}}_2^T \\ \vdots \\ \tilde{\boldsymbol{x}}_N^T \end{bmatrix} \text{ and } \tilde{\boldsymbol{T}} = \begin{bmatrix} \tilde{\boldsymbol{t}}_1^T \\ \tilde{\boldsymbol{t}}_2^T \\ \vdots \\ \tilde{\boldsymbol{t}}_N^T \end{bmatrix} \tag{10}$$

Cost function: The sum-of-squares error for all training data points is

$$E_D(\tilde{\boldsymbol{W}}) = \frac{1}{2} Tr\{(\tilde{\boldsymbol{X}}\tilde{\boldsymbol{W}} - \boldsymbol{T})^T(\tilde{\boldsymbol{X}}\tilde{\boldsymbol{W}} - \boldsymbol{T})\}$$

Least squares for classification

By minimizing the above cost function w.r.t $\tilde{\boldsymbol{W}}$, we get

$$\tilde{\boldsymbol{W}} = \tilde{\boldsymbol{X}}^{\dagger}\boldsymbol{T} \tag{11}$$

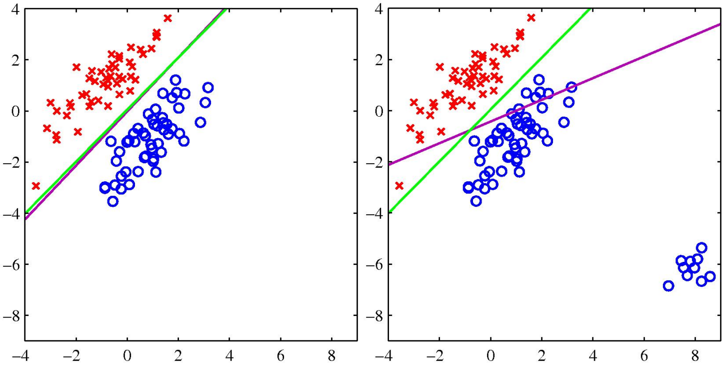
where $\tilde{\boldsymbol{X}}^{\dagger}$ is the pseudo-inverse of the matrix and is given by

$$\tilde{\boldsymbol{X}}^{\dagger} = (\tilde{\boldsymbol{X}}^{T}\tilde{\boldsymbol{X}})^{-1}\tilde{\boldsymbol{X}}^{T} \tag{12}$$

How is this different from the least squares solution for the regression problem?

Least squares for classification

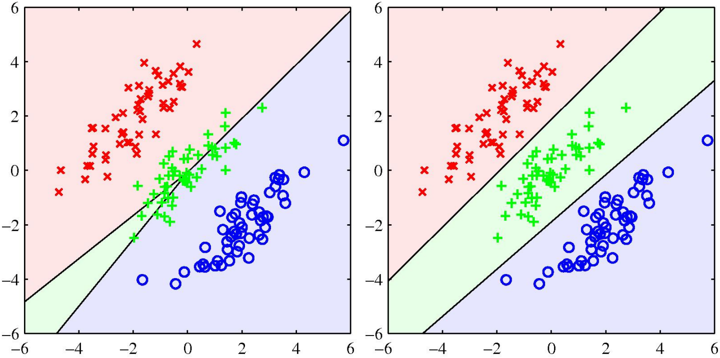
Least squares-based classification is not robust to outliers.



Magenta line corresponds to least squares-based regression and green line corresponds to *logistic regression*.

Least squares for classification

Not optimal even for linearly separable data sets



least squares

logistic regression.

The Perceptron algorithm

Given an input vector $\boldsymbol{x}$, the Perceptron algorithm:

- uses a fixed nonlinear transformation $\phi(\boldsymbol{x})$ (also including $\phi_0(\boldsymbol{x}) = 1$)
- uses a generalized linear model:

$$y(\boldsymbol{x}) = f(\boldsymbol{w}^T \phi(\boldsymbol{x})) \tag{13}$$

  where:

$$f(a) = \begin{cases} +1, & a \geq 0 \\ -1, & a < 0. \end{cases} \tag{14}$$

  is the nonlinear activation function.

- Target values: $t = +1$ for $\mathcal{C}_1$ and $t = -1$ for $\mathcal{C}_2$

Goal: Finding $\boldsymbol{w}$ that is optimal for classification (in some sense).

The Perceptron algorithm

We want a $\boldsymbol{w}$ such that:

- for all $\boldsymbol{x}_n \in \mathcal{C}_1$ we have $\boldsymbol{w}^T \phi(\boldsymbol{x}_n) > 0$
- for all $\boldsymbol{x}_n \in \mathcal{C}_2$ we have $\boldsymbol{w}^T \phi(\boldsymbol{x}_n) < 0$

Using $t_n \in \{-1, +1\}$ we can unify the two cases to: $\boldsymbol{w}^T \phi(\boldsymbol{x}_n) t_n > 0$.

The Perceptron criterion:

- assigns zero error for any correclty classified data point
- assigns an error equal to $-\boldsymbol{w}^T \phi(\boldsymbol{x}_n) t_n$ to $\boldsymbol{x}_n$ if it is misclassified.

$$E_P(\boldsymbol{w}) = -\sum_{n \in \mathcal{M}} \boldsymbol{w}^T \phi(\boldsymbol{x}_n) t_n, \tag{15}$$

where $\mathcal{M}$ is the set of misclassified data points.

The Perceptron algorithm

1. Randomly initialize $\boldsymbol{w}^0$
2. Iterate (until convergence)
    1. shuffle the training vectors $\boldsymbol{x}_n,\ n = 1, \ldots, N$
    2. set $E_P(\boldsymbol{w}) = 0$
    3. iterate through the training vectors $\boldsymbol{x}_\tau$
    4. if $\boldsymbol{x}_\tau$ is misclassified:
        - Compute the gradient $\nabla E_P(\boldsymbol{w}) = -\phi(\boldsymbol{x}_\tau)t_n$
        - Update the weight vector

$$\boldsymbol{w}^{(\tau)} = \boldsymbol{w}^{(\tau-1)} - \eta \nabla E_P(\boldsymbol{w}) = \boldsymbol{w}^{(\tau-1)} + \eta \phi(\boldsymbol{x}_\tau)t_n \tag{16}$$

where $\eta > 0$ is a *learning rate* parameter.

Limitations: Applicable only for linearly separable data and for $K = 2$ classes

The Perceptron algorithm