Pattern Classification

EET3053

Lecture 06: Linear Discriminant Functions

Dr. Kundan Kumar
Associate Professor
Department of ECE

Faculty of Engineering (ITER)
S'O'A Deemed to be University, Bhubaneswar, India-751030

Linear Discriminant Functions

Introduction

- In parametric estimation, we assumed that the forms for the underlying probability densities were known, and used the training samples to estimate the values of their parameters.
- Instead, assume that the proper forms for the discriminant functions is known, and use the samples to estimate the values of parameters of the classifier.
- None of the various procedures for determining discriminant functions require knowledge of the forms of underlying probability distributions so called nonparametric approach.
- Linear discriminant functions are relatively easy to compute and estimate the form using training samples.

Linear discriminant functions and decisions surfaces

- A discriminant function is a linear combination of the components of x can be written as

$$g(\mathrm{x}) = \mathrm{w}^T \mathrm{x} + w_0$$

  where w is the weight vector and w_0 the bias or threshold weight.
- The equation $g(\mathrm{x}) = 0$ defines the decision surface that separates points from different classes.
- Linear discriminant functions are going to be studied for
  - two-category case,
  - multi-category case, and
  - general case

  For the general case there will be c such discriminant functions, one for each of c categories.

Two-Category Case

- A two-category classifier with a discriminant function of the form $g(\mathrm{x}) = \mathrm{w}^T\mathrm{x} + w_0$ uses the following rule:

$$\text{Decide} \quad \begin{cases} \omega_1 & \text{if } g(\mathrm{x}) > 0 \\ \omega_2 & \text{otherwise} \end{cases}$$

- Thus, x is assigned to ω_1 if the inner product $\mathrm{w}^T\mathrm{x}$ exceeds the threshold $-w_0$ and to ω_2 otherwise.

- If $g(\mathrm{x}) = 0$, x can ordinarily be assigned to either class, or can be left undefined.

A simple linear classifier

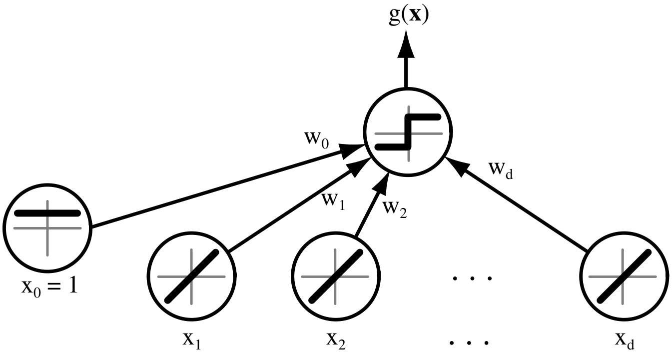
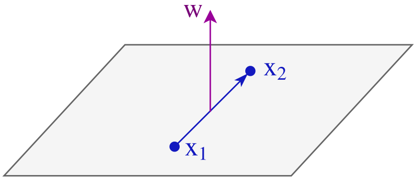


Figure: A simple linear classifier having d input units, each corresponding to the values of the components of an input vector. Each input feature value x_i is multiplied by its corresponding weight w_i; the output unit sums all these products and emits $+1$ if $\mathrm{w}^T\mathrm{x} + w_0 > 0$ or -1 otherwise

Two-Category Case

- The equation $g(\mathrm{x}) = 0$ defines the decision surface that separates points assigned to the category ω_1 from points assigned to the category ω_2
- When $g(\mathrm{x})$ is linear, the decision surface is a hyperplane.
- If x_1 and x_2 are both on the decision surface, then

$$\begin{aligned} & \mathrm{w}^T \mathrm{x}_1 + w_0 = \mathrm{w}^T \mathrm{x}_2 + w_0 \\ \Rightarrow \quad & \mathrm{w}^T (\mathrm{x}_1 - \mathrm{x}_2) = 0 \end{aligned}$$

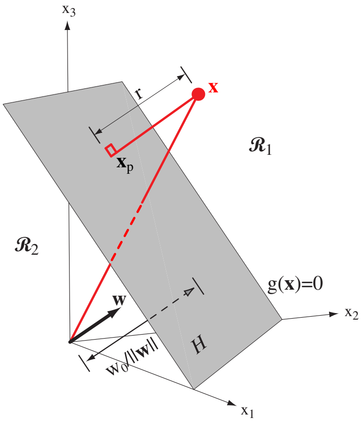


- This shows that w is normal to any vector lying in the hyperplane.

Two-Category Case

- The discriminant function $g(\mathrm{x})$ gives an algebraic measure of the distance from x to the hyperplane. The easiest way to see this is to express x as

$$\mathrm{x} = \mathrm{x}_p + r\frac{\mathrm{w}}{\|\mathrm{w}\|}$$



- where x_p is the normal projection of x onto H, and r is the desired algebraic distance which is positive if x is on the positive side and negative if x is on the negative side.
- Because, $g(\mathrm{x}_p) = 0$

$$r = \frac{g(\mathrm{x})}{\|\mathrm{w}\|}$$

Two-Category Case

- The distance from the origin to H is given by $\frac{w_0}{||\mathrm{w}||}$.
- If $w_0 > 0$, the origin is on the positive side of H, and if $w_0 < 0$, it is on the negative side.
- If $w_0 = 0$, then $g(\mathrm{x})$ has the homogeneous form $\mathrm{w}^T\mathrm{x}$, and the hyperplane passes through the origin.

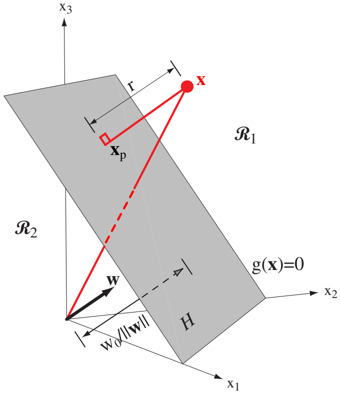


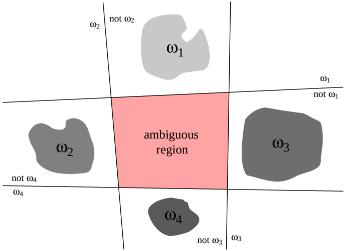
Figure: The linear decision boundary H, where $g(x) = \mathrm{w}^T\mathrm{x} + w_0$, separates the feature space into two half-spaces $\mathcal{R}_1$ (where $g(\mathrm{x}) > 0$) and $\mathcal{R}_2$ (where $g(\mathrm{x}) < 0$))

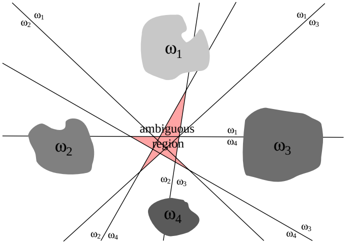
Two-Category Case

- In conclusion, a linear discriminant function divides the feature space by a hyperplane decision surface.
- The orientation of the surface is determined by the normal vector w and the location of the surface is determined by the bias w_0.
- The discriminant function $g(\mathrm{x})$ is proportional to the signed distance from x to the hyperplane, with $g(\mathrm{x}) > 0$ when x is on the positive side, and $g(\mathrm{x}) < 0$ when x is on the negative side.

Multi-category case

- There is more than one way to devise multi-category classifiers employing linear discriminant functions.
  - c two-class problem (one-vs-rest)
  - $c(c-1)/2$ linear discriminants, one for every pair of classes (one-vs-one).





- Pink regions have ambiguous category assignment.

Multi-category case

- More effective way is to define c linear discriminant functions

$$g_i(\mathrm{x}) = \mathrm{w}_i^T \mathrm{x} + w_{i0} \qquad i = 1, 2, \ldots, c$$

and assign x to ω_i if $g_i(\mathrm{x}) > g_j(\mathrm{x})$ for all $j \neq i$; in case of ties, the classification is undefined

- In this case, resulting classifier is a "*linear machine*".
- A linear machine divides the feature space into c decision regions, with $g_i(\mathrm{x})$ being the largest discriminant if x is in the region $\mathcal{R}_i$.
- For a two contiguous regions $\mathcal{R}_i$ and $\mathcal{R}_j$; the boundary that separates them is a portion of hyperplane H_{ij} defined by:

$$g_i(\mathrm{x}) = g_j(\mathrm{x}) \qquad or \qquad (\mathrm{w}_i - \mathrm{w}_j)^T \mathrm{x} + (w_{i0} - w_{j0}) = 0$$

Multi-category case

- It follows at once that $\mathrm{w}_i - \mathrm{w}_j$ is normal to H_{ij}, and the signed distance from x to H_{ij} is given by

$$r = \frac{(g_i(\mathrm{x}) - g_j(\mathrm{x}))}{\|\mathrm{w}_i - \mathrm{w}_j\|}$$

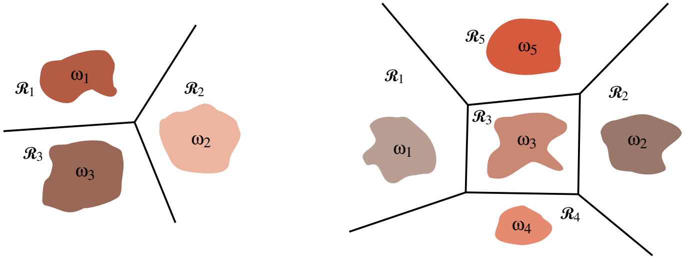


Figure: Decision boundaries produced by a linear machine for a three-class problem and a five-class problem

The Two-Category Case

- For the two-category case, the decision rule can be written as

$$\text{Decide} \quad \begin{cases} \omega_1 & \text{if } g(\mathrm{x}) > 0 \\ \omega_2 & \text{otherwise} \end{cases}$$

- The equation $g(\mathrm{x}) = 0$ defines the decision boundary that separates points assigned to ω_1 from points assigned to ω_2.
- When $g(\mathrm{x})$ is linear, the decision surface is a hyperplane whose orientation is determined by the normal vector w and location is determined by the bias w_0.

Generalized Linear Discriminant Functions

Generalized Linear Discriminant Functions

- The linear discriminant function $g(\mathrm{x})$ is defined as

$$g(\mathrm{x}) = \mathrm{w}^T \mathrm{x} + w_0 \quad (1)$$

$$= w_0 + \sum_{i=1}^{d} w_i x_i \quad (2)$$

where $\mathrm{w} = [w_1, \ldots, w_d]^T$, and $\mathrm{x} = [x_1, x_2, \ldots, x_d]^T$

- We can obtain the *quadratic discriminant function* by adding second-order terms as

$$g(\mathrm{x}) = w_0 + \sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d} \sum_{j=1}^{d} w_{ij} x_i x_j \quad (3)$$

Because $x_i x_j = x_j x_i$, we can assume that $\mathrm{w}_{ij} = \mathrm{w}_{ji}$ with no loss in generality. Which result in more complicated decision boundaries. (hyperquadrics)

Generalized Linear Discriminant Functions

- The quadratic discriminant function has an additional $d(d+1)/2$ coefficients at its disposal with which to produce more complicated separating surfaces.
- The separating surface defined by $g(\mathrm{x}) = 0$ is a second-degree or hyperquadric surface.
- If the symmetric matrix, $\mathrm{W} = [\mathrm{w}_{ij}]$, is nonsingular, the linear term in $g(\mathrm{x})$ can be eliminated by translating the axes.

Generalized Linear Discriminant Functions

- The basic character of the separating surface can be described in terms of scaled matrix

$$\bar{\mathrm{W}} = \frac{\mathrm{W}}{\mathrm{w}^T \mathrm{W}^{-1} \mathrm{w} - 4w_0}$$

where $\mathrm{w} = (w_1, \ldots, w_d)^T$ and $\mathrm{W} = [w_{ij}]$

- The types of quadratic separating surfaces that arise in the general multivariate Gaussian case are as follows
  1. If $\bar{\mathrm{W}}$ is a positive multiple of the identity matrix, the separating surface is a *hypersphere* such that $\bar{\mathrm{W}} = kI$.
  2. If $\bar{\mathrm{W}}$ is positive definite, the separating surfaces is a *hyperellipsoid* whose axes are in the direction of the eigenvectors of $\bar{\mathrm{W}}$.
  3. If none of the above cases holds, that is, some of the eigenvalues of are positive and others are negative, the surface is one of the varieties of types of *hyperhyperboloids*.

Generalized Linear Discriminant Functions

- By continuing to add terms such as $w_{ijk}x_ix_jx_k$, we can obtain the class of *polynomial discriminant functions*. These can be thought of as truncated series expansions of some arbitrary $g(\mathrm{x})$, and this in turn suggest the *generalized linear discriminant function*.

$$g(\mathrm{x}) = \sum_{i=1}^{\hat{d}} a_i \mathrm{y}_i(\mathrm{x}) = \mathrm{a}^T \mathrm{y}$$

  where a is a $\hat{d}-$dimensional weight vector and $\hat{d}$ functions $\mathrm{y}_i(\mathrm{x})$ are arbitrary functions of x.

- The physical interpretation is that the functions $\mathrm{y}_i(\mathrm{x})$ map points x from d-dimensional space to point y in $\hat{d}$-dimensional space.
- The resulting discriminant function is not linear in x, but it is linear in y.

Generalized Linear Discriminant Functions

- Then, the discriminant $g(\mathrm{x}) = \mathrm{a}^T\mathrm{y}$ separates points in the transformed space using a hyperplane passing through the origin.
- The mapping to a higher dimensional space may increase the complexity of the learning algorithms.
- However, certain assumptions can make the problem tractable.
- Let the quadratic discriminant function be

$$g(\mathrm{x}) = a_1 + a_2\mathrm{x} + a_3\mathrm{x}^2$$

- So that the three-dimensional vector y is given by

$$\mathrm{y} = [1 \;\; \mathrm{x} \;\; \mathrm{x}^2]^T$$

Generalized Linear Discriminant Functions

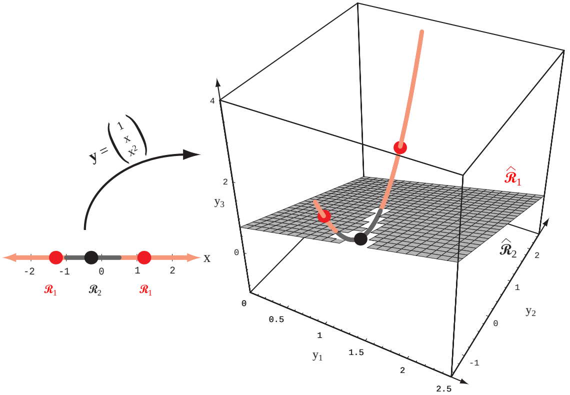


Figure: The mapping $\mathrm{y} = (1 \quad \mathrm{x} \quad \mathrm{x}^2)^T$ takes a line and transforms it to a parabola in three dimensions. A plane splits the resulting y space into regions corresponding to two categories, and this in turn gives a non-simply connected decision region in the one-dimensional x space.

Generalized Linear Discriminant Functions

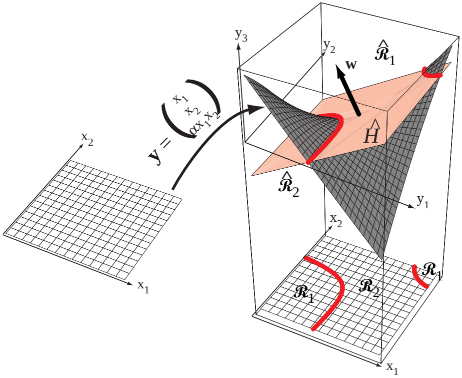


Figure: The two-dimensional input space x is mapped through a polynomial function f to y. Here the mapping is $y_1 = x_1$, $y_2 = x_2$ and $y_3 \propto x_1 x_2$. A linear discriminant in this transformed space is a hyperplane, which cuts the surface. Points to the positive side of the hyperplane $\hat{H}$ correspond to category ω_1 , and those beneath it ω_2. Here, in terms of the x space, $\mathcal{R}_1$ is a not simply connected.

Problem to be solved

Question:

The following three decision functions are given for a three-class problem.

$$g_1(\mathrm{x}) = 10x_1 - x_2 - 10 = 0$$
$$g_2(\mathrm{x}) = x_1 + 2x_2 - 10 = 0$$
$$g_3(\mathrm{x}) = x_1 - 2x_2 - 10 = 0$$

i. Sketch the decision boundary and regions for each pattern class.

ii. Assuming that each pattern class is pairwise linearly separable from every other class by a distinct decision surface and letting

$$g_{12}(\mathrm{x}) = g_1(\mathrm{x})$$
$$g_{13}(\mathrm{x}) = g_2(\mathrm{x})$$
$$g_{23}(\mathrm{x}) = g_3(\mathrm{x})$$

as listed above, sketch the decision boundary and regions for each pattern class.

Two-category linearly separable case

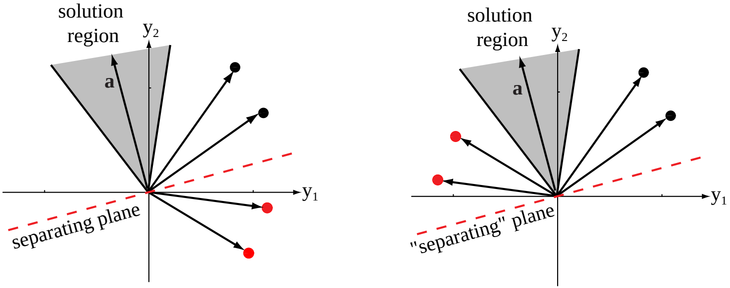
2-category linearly separable case

- Suppose, we have a set of n samples $\mathrm{y}_1, \ldots, \mathrm{y}_n$ some labeled ω_1 and some labeled ω_2.
- Note that all samples are augmented feature vectors.
- We want to use these samples to determine the weights a in a linear discriminant function $g(\mathrm{x}) = \mathrm{a}^T\mathrm{y}$.
- If such a exists that
  - $\mathrm{a}^T\mathrm{y}_i > 0$ for all y_i belonging to ω_1, and
  - $\mathrm{a}^T\mathrm{y}_i < 0$ for all y_i belonging to ω_2

  samples $\mathrm{y}_1, \ldots, \mathrm{y}_n$ are called linearly separable.
- Then, it is reasonable to try to find such a that all the training samples are classified correctly.

2-category linearly separable case

- Normalize the samples $\mathbf{y}_1, \ldots, \mathbf{y}_n$: replace all $\mathbf{y}_i$ labeled ω_2 by their negatives.



- With this normalized set of training samples, we can forget about labels and look for the weight vector $\mathbf{a}$ that satisfies

$$\mathbf{a}^T\mathbf{y}_i > 0 \qquad \text{for all } \mathbf{y}_i.$$

- Such $\mathbf{a}$ is called a solution vector.

Solution regions

- A solution vector - if exists - is not unique. The set of possible solution vectors, that are interpreted as points in $\Re^d$, is called the solution region.
- More formally the solution region is the set

$$\{\mathrm{a} \mid \mathrm{a}^T \mathrm{y}_i > 0; \qquad \text{for all } i = 1, \ldots, n\}$$

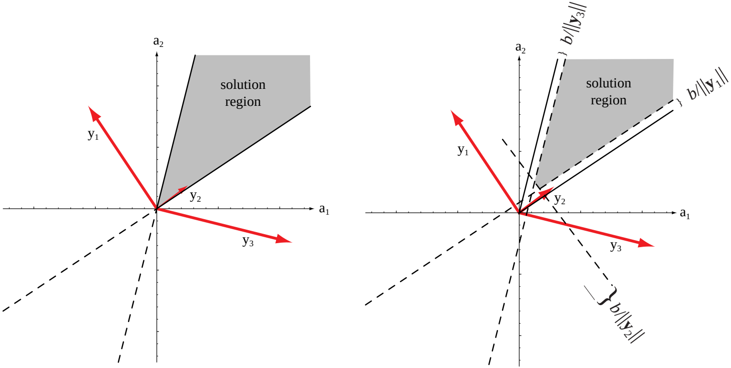
- There are several ways to impose additional requirements to constrain the solution vector.
- One possibility is to seek a unit-length weight vector that maximizes the minimum distance from the samples to the separating plane.

Solution regions

- Another possibility is to seek the minimum-length weight vector satisfying

$$\mathrm{a}^T \mathrm{y}_i \geq b, \quad \forall\, i = 1, \ldots, n$$

where, b is a positive constant, called the margin.



Solving inequalities

- To find a solution to the set of linear inequalities

$$\mathrm{a}^T \mathrm{y}_i > 0$$

  we define a criterion function $J(\mathrm{a})$ that is minimized if a is a solution.
- This kind of problem can be solved by gradient descent.
- The idea is very simple: Start with some vector $\mathrm{a}(1)$. Generate then $\mathrm{a}(2)$ by taking a small step in the direction of $-\nabla J(\mathrm{a}(1))$ and so on.
- Explanation: $-\nabla J(\mathrm{a}(k))$ is the direction of the steepest descent.
- In general, $\mathrm{a}(k+1)$ is obtained from $\mathrm{a}(k)$ by the equation

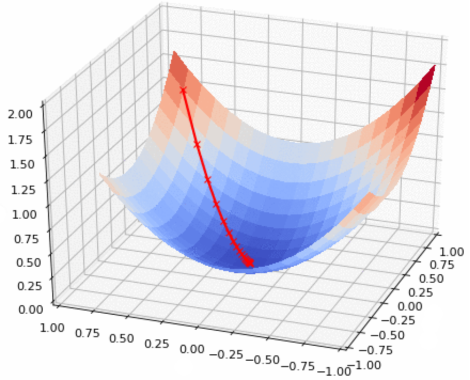
$$\mathrm{a}(k+1) = \mathrm{a}(k) - \eta(k)\nabla J(\mathrm{a}(k)),$$

  where η is a positive scale factor or learning rate that sets the step size.

Basic gradient descent algorithm

Algorithm 1 (Basic gradient descent)

begin initialize $\mathbf{a}$, criterion $\theta, \eta(\cdot), k = 0$
  do $k \leftarrow k + 1$
    $\mathbf{a} \leftarrow \mathbf{a} - \eta(k)\boldsymbol{\nabla} J(\mathbf{a})$
  until $\eta(k)\boldsymbol{\nabla} J(\mathbf{a}) < \theta$
return $\mathbf{a}$
end



- The learning rate can be set

$$\eta(k) = \frac{\|\nabla J(\mathrm{a}(k))\|^2}{\nabla J(\mathrm{a}(k))^T H \nabla J(\mathrm{a}(k))}$$

where H is the Hessian at $\mathrm{a}(k)$.

Newtons algorithm

Algorithm 2 (Newton descent)

begin initialize $\mathbf{a}$, criterion θ
  do
    $\mathbf{a} \leftarrow \mathbf{a} - \mathbf{H}^{-1}\nabla J(\mathbf{a})$
  until $\mathbf{H}^{-1}\nabla J(\mathbf{a}) < \theta$
 return $\mathbf{a}$
end

$$\mathbf{H}(\mathbf{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1 \partial x_1} & \cdots & \frac{\partial f}{\partial x_1 \partial x_d} \\ \vdots & \ddots & \vdots \\ \frac{\partial f}{\partial x_d \partial x_1} & \cdots & \frac{\partial f}{\partial x_d \partial x_d} \end{bmatrix}.$$

- Another possibility is to set the learning rate to be a constant that is small enough. This makes one iteration of the descent algorithm much faster, but the descent takes with a constant learning rate more iterations. There is no general answer how to set the learning rate optimally: The best selection depends on the application.

Minimizing Perceptron Criterion Function

Perceptron Criterion Function

- Consider now the problem of constructing a criterion function for solving the linear inequalities. Assume that the margin $b = 0$.
- The most obvious choice would be the number of samples misclassified by a. However, this criterion is a piece-wise constant function and a poor candidate for a gradient search.
- The perceptron criterion function is defined by

$$J_p(\mathrm{a}) = \sum_{\mathrm{y} \in \mathcal{Y}} -\mathrm{a}^T \mathrm{y},$$

where $\mathcal{Y}$ is the set of samples misclassified by a, i.e. samples for which the inner product with a is negative.

Perceptron Criterion Function

- The gradient

$$\nabla J_p = \sum_{\mathrm{y} \in \mathcal{Y}} -\mathrm{y},$$

- The update rule in gradient descent is

$$\mathrm{a}(k+1) = \mathrm{a}(k) + \eta(k) \sum_{\mathrm{y} \in \mathcal{Y}_k} \mathrm{y}$$

where $\mathcal{Y}_k$ is the set of samples misclassified by $\mathrm{a}(k)$.

Perceptron Algorithm

Algorithm 3 (Batch Perceptron)

begin initialize $\mathbf{a}, \eta(\cdot)$, criterion $\theta, k = 0$
    do $k \leftarrow k + 1$
        $\mathbf{a} \leftarrow \mathbf{a} + \eta(k) \sum_{\mathbf{y} \in \mathcal{Y}_k} \mathbf{y}$
    until $\eta(k) \sum_{\mathbf{y} \in \mathcal{Y}_k} \mathbf{y} < \theta$
  return $\mathbf{a}$
end

- A good feature of the perceptron algorithm is that it will converge to a solution vector if training samples are linearly separable and the learning rate satisfies certain conditions.
- A bad feature of the perceptron algorithm is that it does not (necessarily) converge if the training samples are not linearly separable.

Other criterion functions

- Relaxation Criterion:

$$J_r(\mathrm{a}) = \frac{1}{2} \sum_{y \in \mathcal{Y}} \frac{(\mathrm{a}^T \mathrm{y} - b)^2}{\|\mathrm{y}\|^2}$$

  where b is the margin and $\mathcal{Y}(\mathrm{a})$ is the set of samples for which $\mathrm{a}^T \mathrm{y} \leq b$.

- Sum-of-squared-error criterion:

$$J_s(\mathrm{a}) = ||Y\mathrm{a} - b||^2 = \sum_{i=1}^{n} (\mathrm{a}^T \mathrm{y}_i - b)^2$$

Minimum Squared-Error and the Pseudoinverse

- Let Y be the $n \times \hat{d}$ matrix ($\hat{d} = d + 1$), whose ith row is the vector y_i^T.
- Treat all linear equations simultaneously.

$$\mathrm{a}^T \mathrm{y}_i = \mathrm{b} \qquad \forall i = 1, \ldots, n$$

- Combining all linear equation in a matrix form

$$\begin{bmatrix} y_{10} & y_{11} & \cdots & y_{1d} \\ y_{20} & y_{21} & \cdots & y_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ y_{n0} & y_{n1} & \cdots & y_{nd} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_d \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

$$\boxed{Y\mathrm{a} = b}$$

Minimum Squared-Error and the Pseudoinverse

- We seek for a weight vector a that minimizes some function of the error between $Y\mathrm{a}$ and b.

$$\mathrm{e} = Y\mathrm{a} - \mathrm{b}$$

- Sum-of-squared-error (SSE) criterion function:

$$J_s(\mathrm{a}) = ||Y\mathrm{a} - b||^2 = \sum_{i=1}^{n} \left(\mathrm{a}^T \mathrm{y}_i - b\right)^2$$

- Minimizing the criterion function

$$\nabla J_s = \sum_{i=1}^{n} 2(\mathrm{a}^T \mathrm{y}_i - \mathrm{b}_i)\mathrm{y}_i = 2Y^T(Y\mathrm{a} - \mathrm{b}) = 0$$

$$Y^T Y \mathrm{a} = Y^T \mathrm{b}$$

$$\mathrm{a} = (Y^T Y)^{-1} Y^T \mathrm{b}$$

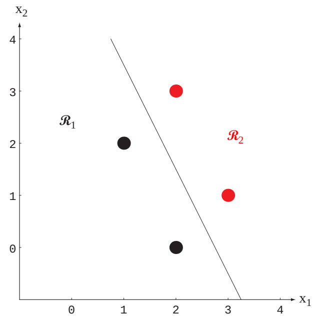
$$\mathrm{a} = Y^\dagger \mathrm{b}$$

- However, $Y^\dagger$ is defined more generally by $\boxed{Y^\dagger \equiv \lim_{\varepsilon \to 0} (Y^T Y + \varepsilon I)^{-1} Y^T}$

Example

Question:

Suppose we have the following two-dimensional point for two categories: ω_1: $(1,2)^T$ and $(2,0)^T$, and ω_2: $(3,1)^T$ and $(2,3)^T$. Construct a Linear Classifier by Matrix Pseudoinverse.



References

[1] Hart, P. E., Stork, D. G., & Duda, R. O. (2000). Pattern classification. Hoboken: Wiley.

[2] Gose, E. (1997). Pattern recognition and image analysis.

Thank you!