Machine Learning

Discriminant Functions

Arun Chauhan

Linear Models for Classification

- So far we study regression models.
- This lecture discuss an analogous class of models for solving classification problems.
- Goal in classification is to take an input vector $\mathbf{x}$ and to assign it to one of K discrete classes C_k
  - ✓ Where $k = 1,2,\ldots., K$
- In the most common scenario
  - ✓ the classes are taken to be disjoint
  - ✓ so that each input is assigned to one and only one class
- The input space is thereby divided into *decision regions* whose boundaries are called *decision boundaries* or *decision surfaces*
- We consider linear models for classification
  - ✓ by which we mean that the decision surfaces are linear functions of the input vector $\mathbf{x}$
  - ✓ Hence are defined by $(D - 1)$-dimensional hyperplanes within the D-dimensional input space

Linear Models for Classification

- In classification, there are various ways of using target values to represent class labels.

For probabilistic models,

- The most convenient, in the case of **two-class problems**,
  - ✓ is the binary representation in which there is a single target variable, $\mathbf{t \in \{0,1\}}$
  - ✓ $t = 1$ represents the class C_1
  - ✓ $t = 0$ represents the class C_2
  - ✓ We can interpret the value of t as the probability that the class is C_1
- For $K > 2$ classes, it is convenient to use a 1-of-K coding scheme in which $\mathbf{t}$ is a vector of length K
  - ✓ such that if the class is C_j , then all elements t_k of $\mathbf{t}$ are zero except element t_j, which takes the value 1
  - ✓ For example, if we have $K = 5$ classes, then a pattern from class 2 would be given the target vector

$$t = (0, 1, 0, 0, 0)^T$$

  - ✓ Value of t_k as the probability that the class is C_k

Solving decision problems

In general, decision problems can be solved using three distinct approach as follows given in decreasing order of complexity.

(a) First solve the inference problem of determining the class-conditional densities $p(\mathbf{x}|\mathcal{C}_k)$ for each class $\mathcal{C}_k$ individually. Also separately infer the prior class probabilities $p(\mathcal{C}_k)$. Then use Bayes' theorem in the form

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})}$$

Approaches that explicitly or implicitly model the distribution of inputs as well as outputs are known as *generative models*, because by sampling from them it is possible to generate synthetic data points in the input space.

(b) First solve the inference problem of determining the posterior class probabilities $p(\mathcal{C}_k|\mathbf{x})$, and then subsequently use decision theory to assign each new $\mathbf{x}$ to one of the classes. Approaches that model the posterior probabilities directly are called *discriminative models*.

(c) Find a function $f(\mathbf{x})$, called a discriminant function, which maps each input $\mathbf{x}$ directly onto a class label. For instance, in the case of two-class problems, $f(\cdot)$ might be binary valued and such that $f = 0$ represents class $\mathcal{C}_1$ and $f = 1$ represents class $\mathcal{C}_2$. In this case, probabilities play no role.

Generalized Linear Models

- For classification problems, however, we wish to predict discrete class labels, or more generally posterior probabilities that lie in the range (0, 1).
- To achieve this, we consider a **generalization of linear model** in which we transform the linear function of **w** using a nonlinear function *f*(·) so that $y(\mathbf{x}) = f\left(\mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0\right)$
- It is also know as **activation function**.
- The decision surfaces correspond to $y(\mathbf{x})$ = constant
  - ✓ So that $\mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0 = \mathrm{constant}$
  - ✓ Hence the decision surfaces are linear functions of **x**
  - ✓ Even if the function *f*(·) is nonlinear
  - ✓ Therefore this class of models are called *generalized linear models*

Discriminant Functions

- A **discriminant** is a function that takes an input vector **x** and assigns it to one of *K* classes, denoted C_k.

- We restrict our attention to *linear discriminants* where the decision surfaces are hyperplanes.

- The simplest representation of a linear discriminant function is obtained by taking a linear function of the input vector so that

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

Hyperplane

- Projection of vector **x** on unit vector $\hat{w}$

- Basics of hyperplane: Assume two dimension plane with axis x_1 and x_2 , let $\hat{w}$ be a unit vector orthogonal to hyperplane (line)

Two classes classification

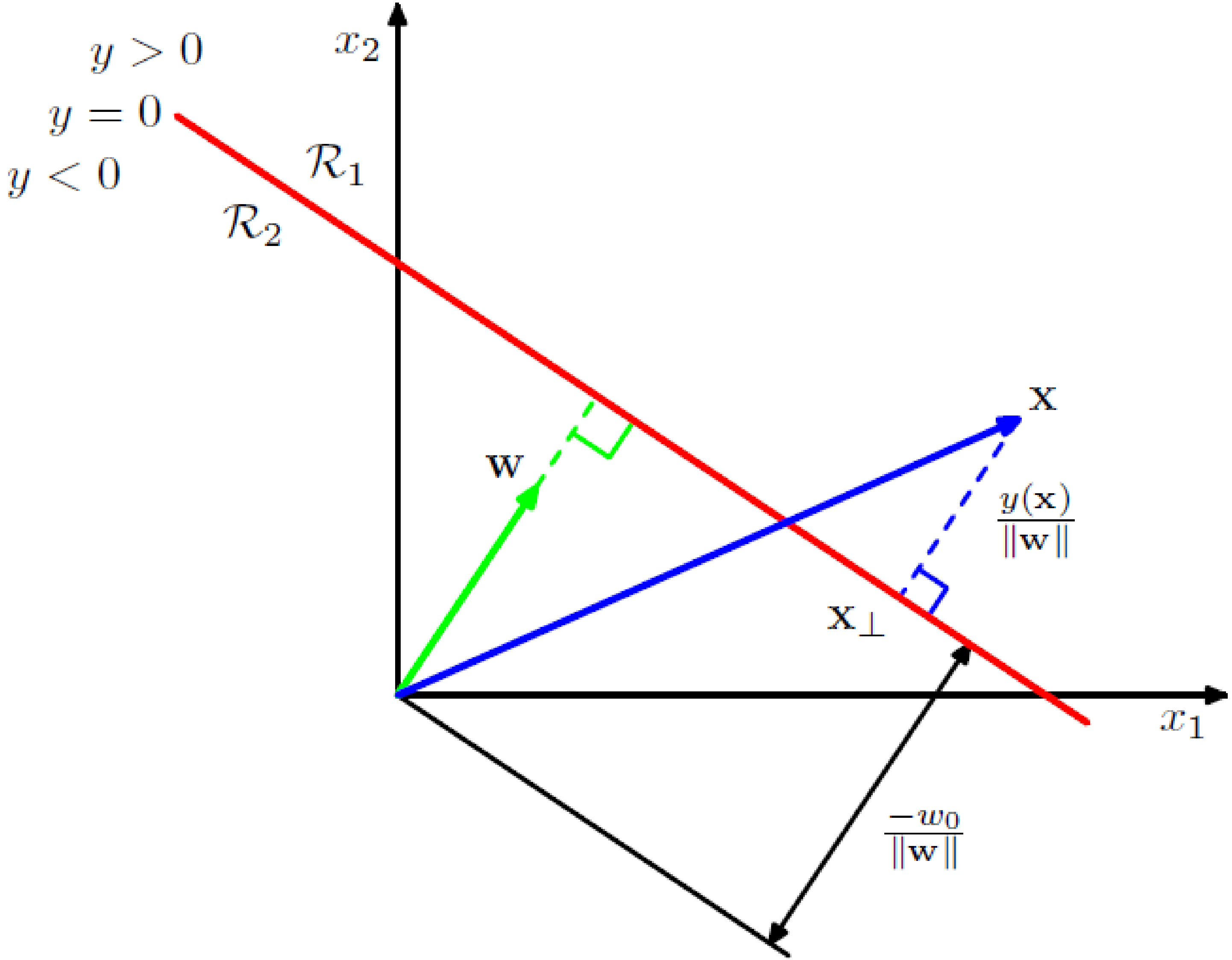
- In two class classification, an input vector **x** is assigned to class C_1 if y(**x**) >= 0 and to class C_2 otherwise.
- The corresponding decision boundary is therefore defined by the relation y(**x**) = 0
  - ✓ Which corresponds to a $(D - 1)$-dimensional hyperplane within the D-dimensional input space.
- Consider two points $\mathbf{x}_A$ and $\mathbf{x}_B$ both of which lie on the decision surface
  - ✓ Because $y(\mathbf{x}_A) = y(\mathbf{x}_B) = 0$, we have $\mathbf{w}^T(\mathbf{x}_A - \mathbf{x}_B) = 0$
  - ✓ Hence the vector **w** is orthogonal to every vector lying within the decision surface.
  - ✓ So **w** determines the orientation of the decision surface.

Two classes classification

- Therefore, the normal distance from the origin to the decision surface is given by

$$\frac{\mathbf{w}^{\mathrm{T}}\mathbf{x}}{\|\mathbf{w}\|} = -\frac{w_0}{\|\mathbf{w}\|}$$

- w_0 determines the location of decision surface.
- $y(\mathbf{x})$ gives a signed measure of the perpendicular distance r of the point $\mathbf{x}$ from the decision surface
- Consider an arbitrary point $\mathbf{x}$ and $\mathbf{x}_\perp$ its orthogonal projection onto the decision surface.
- Therefore: $\mathbf{x} = \mathbf{x}_\perp + r\frac{\mathbf{w}}{\|\mathbf{w}\|}$
- Multiplying both sides of this result by $\mathbf{w}^{\mathrm{T}}$ and adding w_0 to $\mathbf{x} = \mathbf{x}_\perp + r\frac{\mathbf{w}}{\|\mathbf{w}\|}$ we get $r = \frac{y(\mathbf{x})}{\|\mathbf{w}\|}$



x_2
$y > 0$
$y = 0$
$y < 0$
$\mathcal{R}_1$
$\mathcal{R}_2$
$\mathbf{x}$
$\mathbf{w}$
$\frac{y(\mathbf{x})}{\|\mathbf{w}\|}$
$\mathbf{x}_\perp$
x_1
$\frac{-w_0}{\|\mathbf{w}\|}$

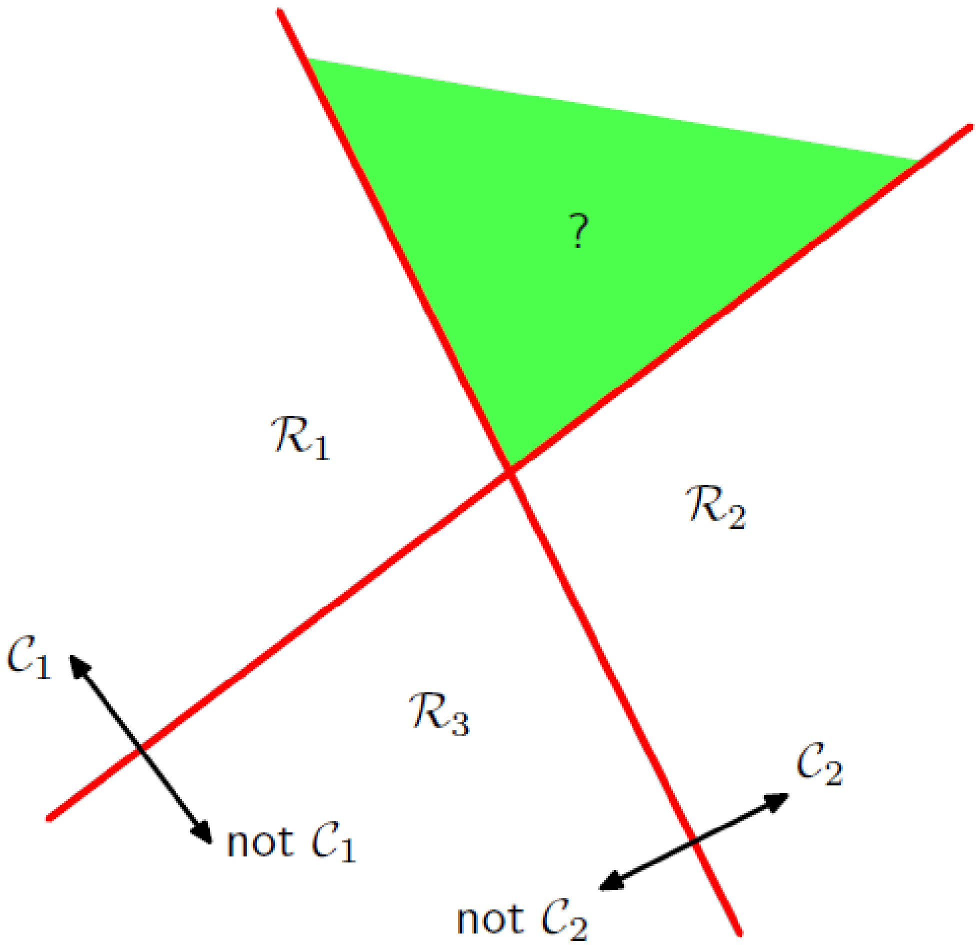
Multiple classes classification

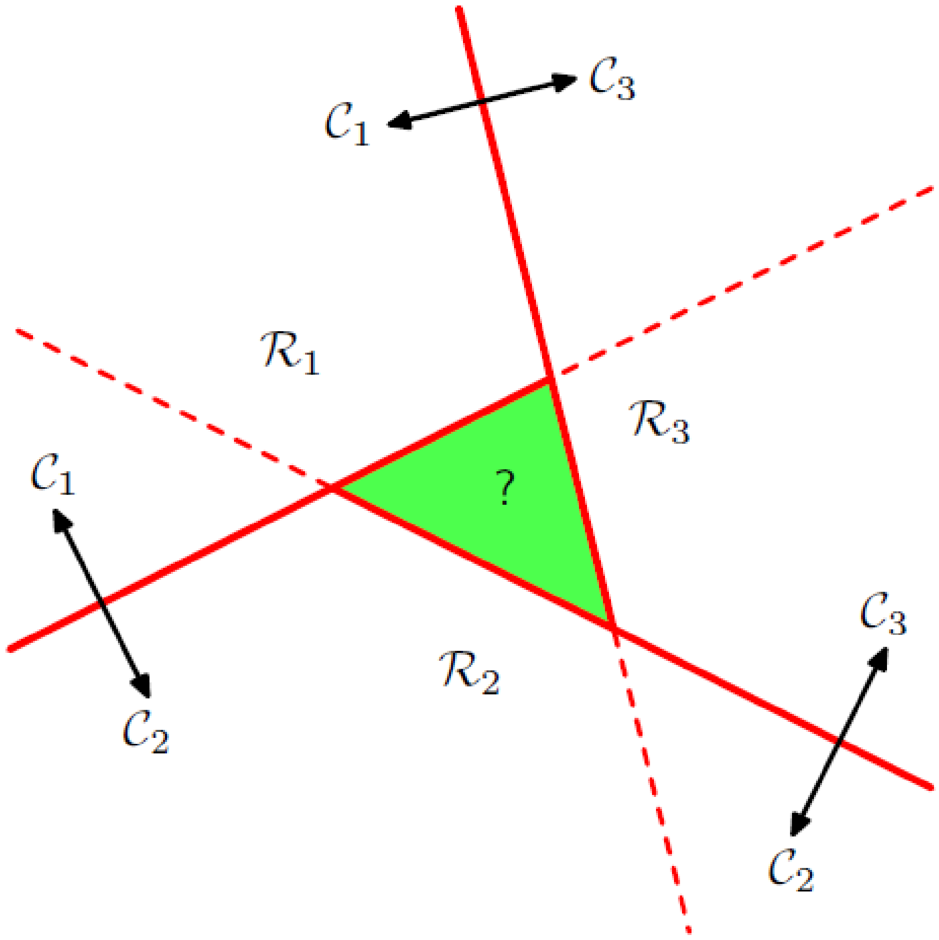
- Now consider the extension of linear discriminants to $K > 2$ classes

- How to use this for multiple classes?
  - ✓ **One-versus-the-rest method**: build $K-1$ classifiers, between C_k and all others
  - ✓ **One-versus-one method**: build $K(K-1)/2$ classifiers, between all pairs

- There are limitation to this kind of classifiers

Multiple classes classification





Multiple classes classification

- We can avoid these difficulties by considering a single *K*-class discriminant comprising *K* linear functions of the form

$$y_k(\mathbf{x}) = \mathbf{w}_k^{\mathrm{T}}\mathbf{x} + w_{k0}$$

- and then assigning a point $\mathbf{x}$ to class $\mathcal{C}_k$ if $y_k(\mathbf{x}) > y_j(\mathbf{x})$ for all $j \neq k$.
- The decision boundary between class C_k and class C_j is therefore given by y_k ($\mathbf{x}$) = y_j ($\mathbf{x}$)
- Hence decision boundary corresponds to a (D − 1)-dimensional hyperplane defined by

$$(\mathbf{w}_k - \mathbf{w}_j)^{\mathrm{T}}\mathbf{x} + (w_{k0} - w_{j0}) = 0$$

- The decision regions of such a discriminant are always singly connected and convex.

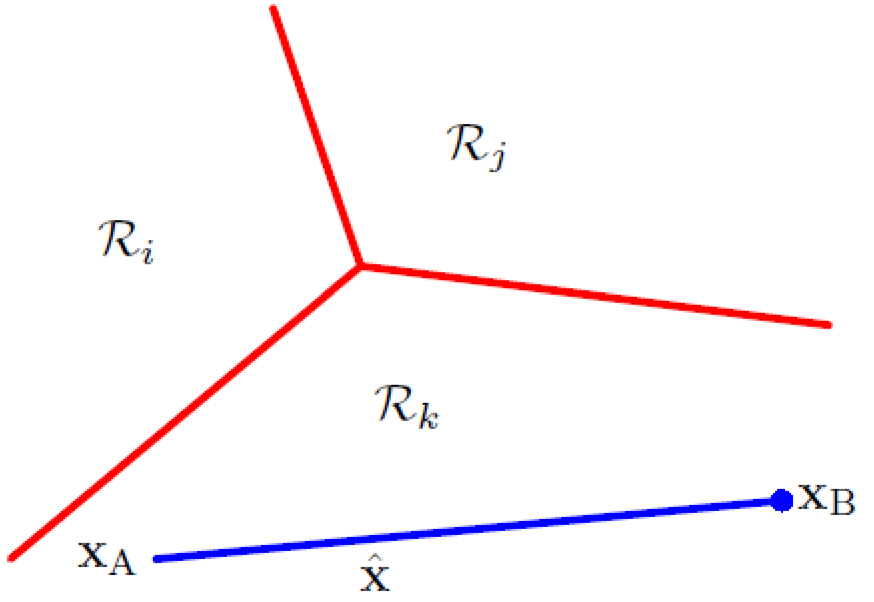
Multiple classes classification

- To see this, consider two points both of which lie inside decision region R_k
- Any point $\widehat{\mathbf{x}}$ that lies on the line connecting $\mathbf{x}_A$ and $\mathbf{x}_B$ can be expressed in the form

$$\widehat{\mathbf{x}} = \lambda \mathbf{x}_A + (1-\lambda)\mathbf{x}_B \qquad \text{where } 0 \leqslant \lambda \leqslant 1.$$

- From the linearity of the discriminant functions, it follows that

$$y_k(\widehat{\mathbf{x}}) = \lambda y_k(\mathbf{x}_A) + (1-\lambda) y_k(\mathbf{x}_B)$$



Because both $\mathbf{x}_A$ and $\mathbf{x}_B$ lie inside $\mathcal{R}_k$, it follows that $y_k(\mathbf{x}_A) > y_j(\mathbf{x}_A)$, and $y_k(\mathbf{x}_B) > y_j(\mathbf{x}_B)$, for all $j \neq k$, and hence $y_k(\widehat{\mathbf{x}}) > y_j(\widehat{\mathbf{x}})$, and so $\widehat{\mathbf{x}}$ also lies inside $\mathcal{R}_k$. Thus $\mathcal{R}_k$ is singly connected and convex.

Least squares for classification

Each class $\mathcal{C}_k$ is described by its own linear model so that $y_k(\mathbf{x}) = \mathbf{w}_k^{\mathrm{T}}\mathbf{x} + w_{k0}$

where $k = 1, \ldots, K$. We can conveniently group these together using vector notation so that

$$\mathrm{y}(\mathrm{x}) = \widetilde{\mathrm{W}}^{\mathrm{T}}\widetilde{\mathrm{x}}$$

where $\tilde{\mathbf{W}} = [\tilde{\mathbf{w}}_1, \ldots, \tilde{\mathbf{w}}_K] = \begin{bmatrix} w_{10} & \cdots & w_{K0} \\ w_{11} & \cdots & w_{K1} \\ \vdots & \ddots & \vdots \\ w_{1D} & \cdots & w_{KD} \end{bmatrix}, \tilde{\mathbf{x}} = \begin{bmatrix} 1 \\ \mathbf{x} \end{bmatrix}$

new input $\mathbf{x}$ is then assigned to the class for which the output $y_k = \widetilde{\mathbf{w}}_k^{\mathrm{T}}\widetilde{\mathbf{x}}$ is largest.

Least squares for classification

- By minimizing a sum-of-square error function parameter matrix $\widetilde{\mathbf{W}}$ can be determined
- Consider a training data set $\{\mathbf{x}_n, \mathbf{t}_n\}$ where $n = 1, \ldots, N$, and define a matrix $\mathbf{T}$ whose n^{th} row is the vector $\mathbf{t}_n^{\mathrm{T}}$ where $\mathbf{t} = [t_1, \ldots, t_K]^T$ (*e.g.*, $[0, 0, 1, 0]^T$)
- together with a matrix $\widetilde{\mathbf{X}}$ whose n^{th} row is $\widetilde{\mathbf{x}}_n^{\mathrm{T}}$

Least squares for classification

- The sum-of-squares error function can then be written as

$$E_D(\widetilde{\mathbf{W}}) = \frac{1}{2}\mathrm{Tr}\left\{(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})^{\mathrm{T}}(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})\right\}$$

Least squares for classification

- Setting the derivative with respect to $\widetilde{\mathbf{W}}$ to zero, and rearranging, we then obtain the solution for $\widetilde{\mathbf{W}}$ in the form

$$\widetilde{\mathbf{W}} = (\widetilde{\mathbf{X}}^{\mathrm{T}}\widetilde{\mathbf{X}})^{-1}\widetilde{\mathbf{X}}^{\mathrm{T}}\mathbf{T}$$

- We then obtain the discriminant function in the form

$$\mathbf{y}(\mathbf{x}) = \widetilde{\mathbf{W}}^{\mathrm{T}}\widetilde{\mathbf{x}}$$

Least squares for classification

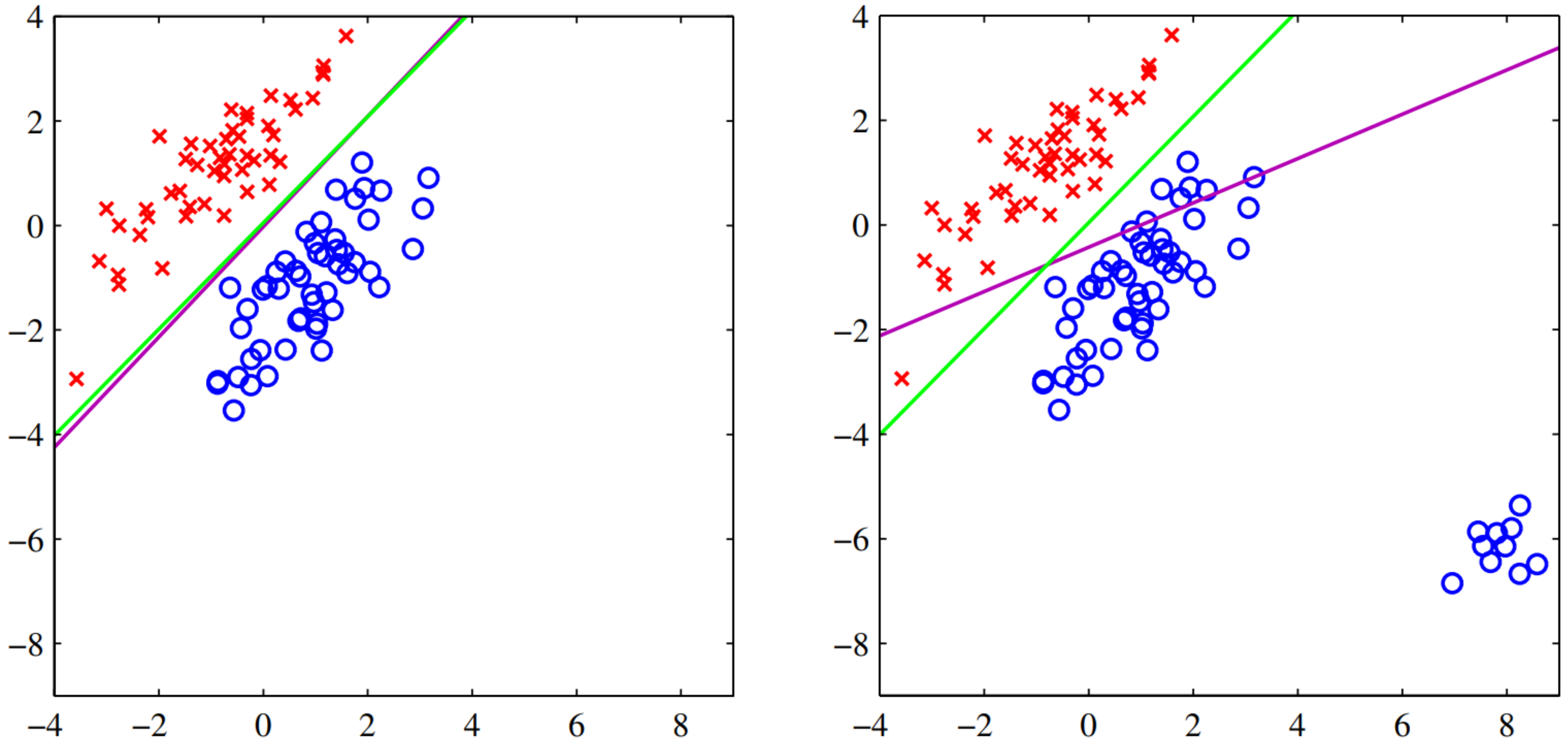
An interesting property of least-squares solutions with multiple target variables is that if every target vector in the training set satisfies some linear constraint

$$\mathbf{a}^{\mathrm{T}}\mathbf{t}_n + b = 0$$

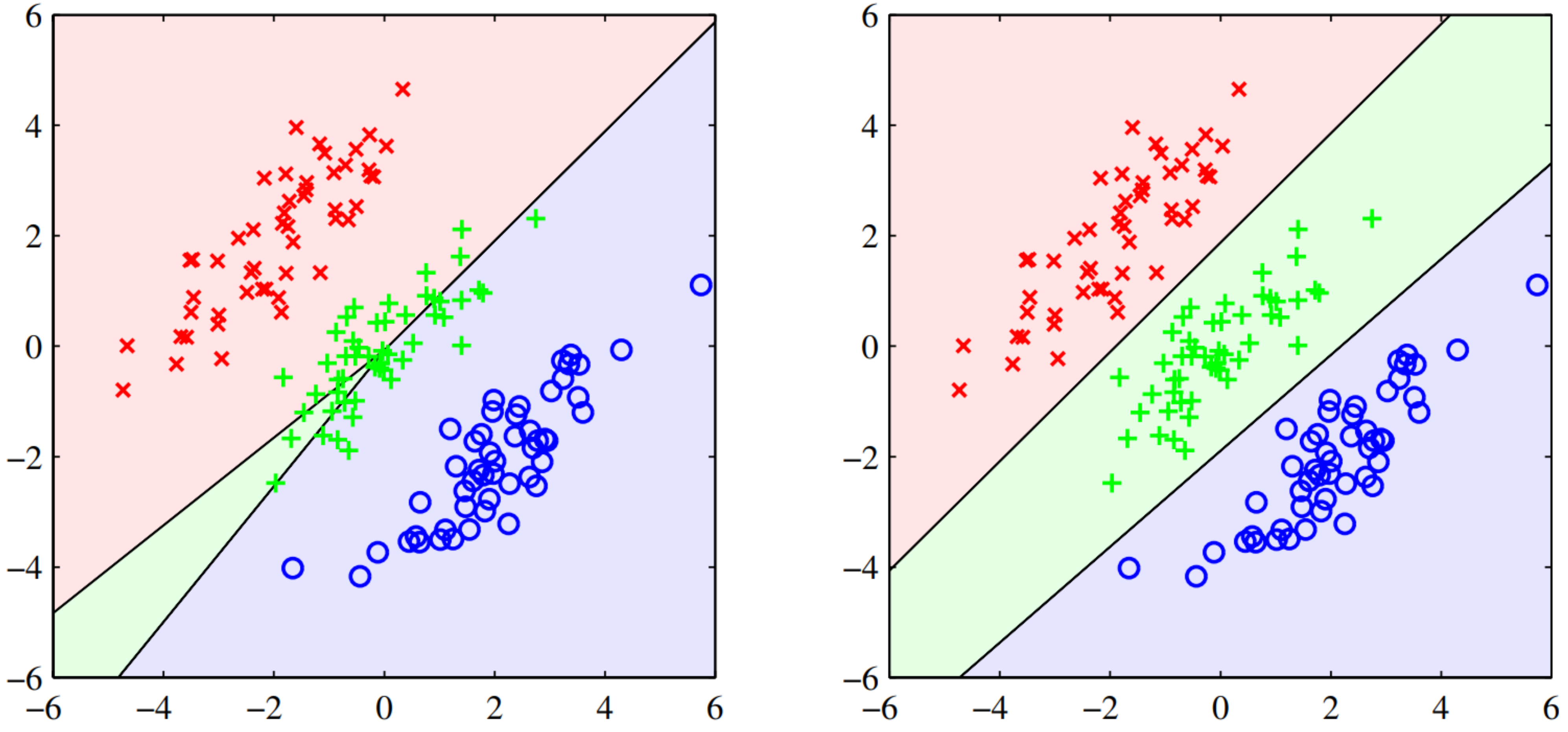
Then: $$\mathbf{a}^{\mathrm{T}}\mathbf{y}(\mathbf{x}) + b = 0$$

interpreted as probabilities ????

Sensitive to outliers



Least square Vs Logistic regression



Fisher's linear discriminant

- Two classes C_1 and C_2 with N_1 and N_2 points respectively

$$\mathbf{m}_1 = \frac{1}{N_1}\sum_{n \in \mathcal{C}_1} \mathbf{x}_n, \qquad \mathbf{m}_2 = \frac{1}{N_2}\sum_{n \in \mathcal{C}_2} \mathbf{x}_n$$

we might choose $\mathbf{w}$ so as to maximize

$$m_2 - m_1 = \mathbf{w}^{\mathrm{T}}(\mathbf{m}_2 - \mathbf{m}_1)$$

$$m_k = \mathbf{w}^{\mathrm{T}}\mathbf{m}_k$$

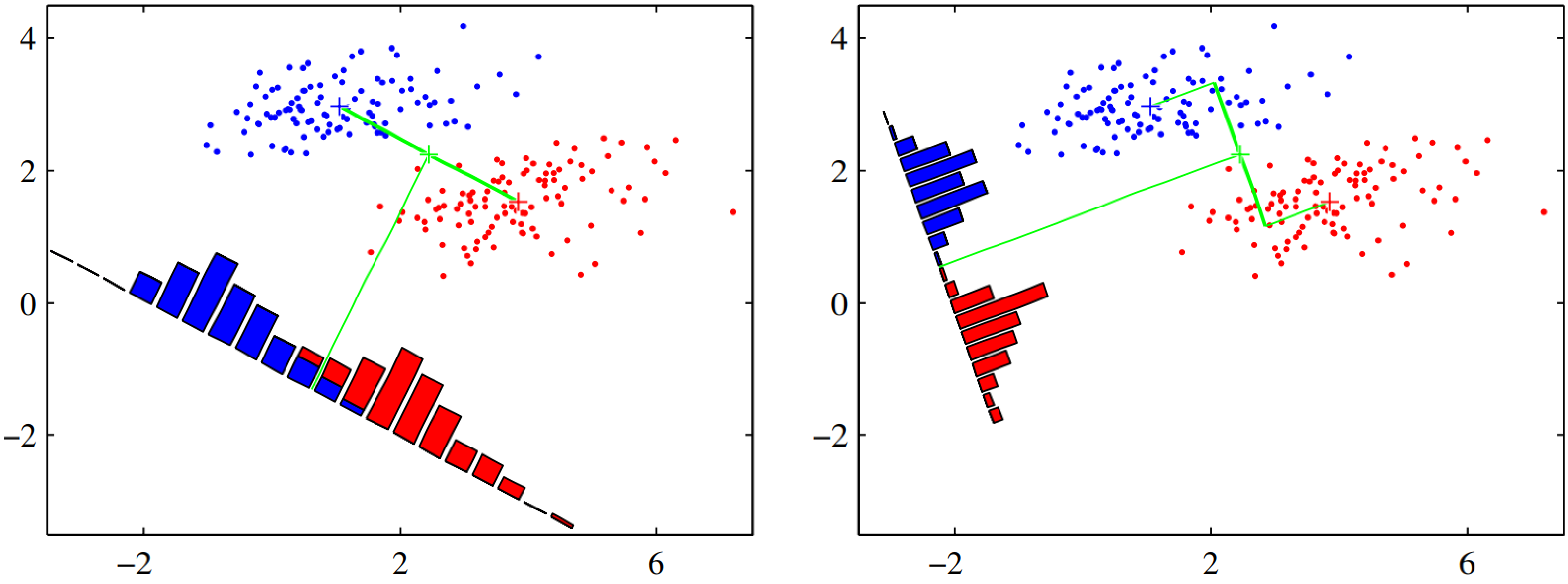
Fisher's linear discriminant

- Using constrain $\mathbf{w}$ to have unit length, so that $\sum_i w_i^2 = 1$

$$\mathbf{w} \propto (\mathbf{m}_2 - \mathbf{m}_1)$$

There is still a problem with this approach,

Fisher's linear discriminant



Fisher's linear discriminant

- The idea proposed by Fisher is to maximize:

  a function that will give a large separation between the projected class means while also giving a small variance within each class, thereby minimizing the class overlap.

- The within-class variance of the transformed data from class $\mathcal{C}_k$ is therefore given by

$$s_k^2 = \sum_{n \in \mathcal{C}_k} (y_n - m_k)^2$$

where $y_n = \mathbf{w}^{\mathrm{T}}\mathbf{x}_n$.

The Fisher criterion

- The Fisher criterion is defined to be the ratio of the :

  between-class variance to the within-class variance and is given by

$$J(\mathbf{w}) = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w}}{\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w}}$$

where $\mathbf{S}_{\mathrm{B}}$ is the *between-class* covariance matrix and is given by

$$\mathbf{S}_{\mathrm{B}} = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^{\mathrm{T}}$$

and $\mathbf{S}_{\mathrm{W}}$ is the total *within-class* covariance matrix, given by

$$\mathbf{S}_{\mathrm{W}} = \sum_{n \in \mathcal{C}_1} (\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^{\mathrm{T}} + \sum_{n \in \mathcal{C}_2} (\mathbf{x}_n - \mathbf{m}_2)(\mathbf{x}_n - \mathbf{m}_2)^{\mathrm{T}}$$

The Fisher criterion

$$\begin{aligned}(m_2 - m_1)^2 &= \left(\mathbf{w}^{\mathrm{T}}(\mathbf{m}_2 - \mathbf{m}_1)\right)^2 \\ &= \mathbf{w}^{\mathrm{T}}(\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^{\mathrm{T}}\mathbf{w} \\ &= \mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w}.\end{aligned}$$

$$\begin{aligned}s_1^2 + s_2^2 &= \sum_{n \in \mathcal{C}_1}(y_n - m_1)^2 + \sum_{k \in \mathcal{C}_2}(y_k - m_2)^2 \\ &= \sum_{n \in \mathcal{C}_1}\left(\mathbf{w}^{\mathrm{T}}(\mathbf{x}_n - \mathbf{m}_1)\right)^2 + \sum_{k \in \mathcal{C}_2}\left(\mathbf{w}^{\mathrm{T}}(\mathbf{x}_k - \mathbf{m}_2)\right)^2 \\ &= \sum_{n \in \mathcal{C}_1}\mathbf{w}^{\mathrm{T}}(\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^{\mathrm{T}}\mathbf{w} \\ &\qquad + \sum_{k \in \mathcal{C}_2}\mathbf{w}^{\mathrm{T}}(\mathbf{x}_k - \mathbf{m}_2)(\mathbf{x}_k - \mathbf{m}_2)^{\mathrm{T}}\mathbf{w} \\ &= \mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w}.\end{aligned}$$

The Fisher criterion

Differentiating with respect to $\mathbf{w}$, we find that $J(\mathbf{w})$ is maximized when

$$(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w})\mathbf{S}_{\mathrm{W}}\mathbf{w} = (\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w})\mathbf{S}_{\mathrm{B}}\mathbf{w}.$$

$\mathbf{S}_{\mathrm{B}}\mathbf{w}$ is always in the direction of $(\mathbf{m}_2 - \mathbf{m}_1)$.

we do not care about the magnitude of $\mathbf{w}$, only its direction, and so we can drop the scalar factors $(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w})$ and $(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w})$. Multiplying both sides of by $\mathbf{S}_{\mathrm{W}}^{-1}$ we then obtain

$$\mathbf{w} \propto \mathbf{S}_{\mathrm{W}}^{-1}(\mathbf{m}_2 - \mathbf{m}_1).$$

References

- Chapter 4, Pattern Recognition and Machine Learning, C. Bishop