Lecture 26 - Discriminant Functions

Linear Discriminant Functions

So far we designed classifiers based on probability density or probability functions. In some cases, we saw that the resulting classifiers were equivalent to a set of linear discriminant functions.

We will now focus on the design of linear classifiers, irrespective of the underlying distributions describing the training data.

- The major advantage of linear classifiers is their simplicity and computational attractiveness.
- We will develop techniques for the computation of the corresponding linear functions. In the sequel we will focus on a more general problem, in which a linear classifier cannot classify correctly all feature vectors, yet we will seek ways to design an optimal linear classifier by adopting an appropriate optimality criterion.

Linear discriminant functions are typically presented for a 2-class problem due to its geometry interpretation.

- Linear discriminant functions are generalizable for $K > 2$ classes

We will learn 3 methods to optimize the parameters of a linear discriminant function (classifier):

1. Least Squares Classification
2. Fisher's Linear Discriminant
3. The Perceptron Algorithm

Linear Decision Boundary

Suppose we have a 2-class problem and we want to find a *linear boundary* that separates the two classes, such that, above the decision boundary all points belong to one class and below the decision boundary all points belong to the other class.

- To do this, we would want the mean of the two classes to be as far a part as possible, and the variance of each class to be as small as possible.

We can design a cost function as the ratio between the squared difference of the means over the sum of the variances. In order to find a linear decision boundary, we want to maximize this cost function.

$$\frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}$$

This cost function is a function of the parameters $\mathbf{w}$ that characterize the line or hyperplane.

- The line of hyperplane perpendicular to the decision boundary will point in the **direction of projection** that preserves class separability.
- We will end up with far and compact clusters which are easy to linearly separate.

We will begin with linear discriminant functions:

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

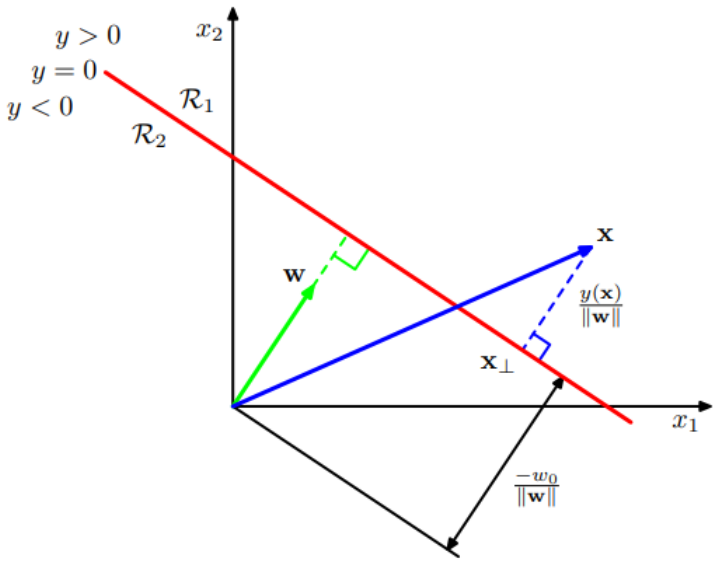
Looks pretty familiar, right? If you are on one side of the line, then you are in class 1. If you are on the other side of the line, then you are in class 2. So, the decision boundary is $y(\vec{\mathbf{x}}) = 0$

- The distance of a point to the decision boundary is: $\frac{y(\vec{\mathbf{x}})}{\|\vec{\mathbf{w}}\|}$
  - See Figure 4.1 from Bishop textbook:

In [1]:
```
from IPython.display import Image
Image('figures/Figure4.1.png', width=800)
```

Out[1]:

Figure 4.1 Illustration of the geometry of a linear discriminant function in two dimensions. The decision surface, shown in red, is perpendicular to $\mathbf{w}$, and its displacement from the origin is controlled by the bias parameter w_0. Also, the signed orthogonal distance of a general point $\mathbf{x}$ from the decision surface is given by $y(\mathbf{x})/\|\mathbf{w}\|$.



We could use a **least squares** error function to solve for $\vec{\mathbf{w}}$ and w_0 as we did in regression. But, there are some issues. *Can you think of any?*

- In regression, the prediction label will be a continuous number between $[-1, 1]$. So the predicted class label will be for example: -0.8, 0.4 or 0.01. To simplify, we can say, if the predicted class $y \geq 0$ than is class 1 otherwise is class 0.
- The problem that comes about is that, if we look at the distribution of our errors, in our estimation $\epsilon = t - y$ is not Gaussian.
- The errors samples are assumed independent, with a mean and a variance independent from each other.
- If we use regression, what we going end up with is an error distribution where the variance is dependent on the mean. This becomes a signal-dependent problem therefore regression is not a good approach to classification.

Fisher's Linear Discriminant

A very popular type of a linear discriminant is the **Fisher's Linear Discriminant**.

- Given two classes, we can compute the mean of each class:

$$\vec{\mathbf{m}}_1 = \frac{1}{N_1} \sum_{n \in C_1} \vec{\mathbf{x}}_\mathbf{n}$$

$$\vec{\mathbf{m}}_\mathbf{2} = \frac{1}{N_2} \sum_{n \in C_2} \vec{\mathbf{x}}_\mathbf{n}$$

We can maximize the separation of the means:

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

- $\vec{\mathbf{w}}^T \vec{\mathbf{x}}$ takes a D dimensional data point and projects it down to 1-D with a weight sum of the original features. We want to find a weighting that maximizes the separation of the class means.

to be continued...