LECTURE 3: DENSITY & DISCRIMINANTS

In this chapter we consider the important (and obvious) point; the resultant class boundaries are dependent upon the type of analysis model we propose. We will restrict ourselves (for brevity only, as we may regard classification as a special case of regression anyhow) to the problems of classification. As we have seen, classification should be based upon posterior probabilities. How do we get the posteriors? Remember that the posterior probabilities are given in terms of the class-conditional priors and likelihood functions. We have three choices:

1. Estimate the priors and likelihoods in Bayes' theorem.
2. Estimate some *discriminant function* which is monotone in the posteriors i.e. choosing the class with the largest discriminant is equivalent to choosing the maximum posterior class.
3. By-pass the issue of estimating priors and likelihoods altogether and estimate posteriors directly.

Some methods though (e.g. the linear discriminant analyser) can fall into all three - it depends on how we look at it!

Estimating the densities

This section will detail, in principle, how we might make and utilise estimates of the quantities required. Later on we will see in much more detail how parameter estimation may be achieved.

Priors

We may believe in some simple ways to estimate the priors, either by setting them all equal to $1/\#\mathcal{C}$ or, for N examples

$$\hat{P}(C_k) = \frac{n_k}{N}$$

the problem then remains of estimating the likelihood functions, $P(x \mid C_k)$.

Likelihoods

We may regard methods of estimating the likelihood functions as methods of estimating the class-conditional *probability density functions*. We will consider three main approaches, depending on the model:

- Non-parametric

- Parametric
- Semi-parametric

Non-parametric: We estimate the class likelihood functions for example by 'histogram binning' methods – basically counting training set data density over the feature space.

Parametric methods: make a *strong assumption* about the form of the likelihood functions. Often it is assumed that each class is Gaussian distributed. The likelihood for class A, say, is thus the (multi-variate) Gaussian function. So if $\mathbf{x}$ is a p-vector

$$P(\mathbf{x} \mid C_k) = \frac{1}{(2\pi)^{p/2}|\mathbf{F}|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\mathbf{F}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right]$$

where μ is the centre (mean) of class C_k and $\mathbf{F}$ is the $(p \times p)$ *covariance matrix*.

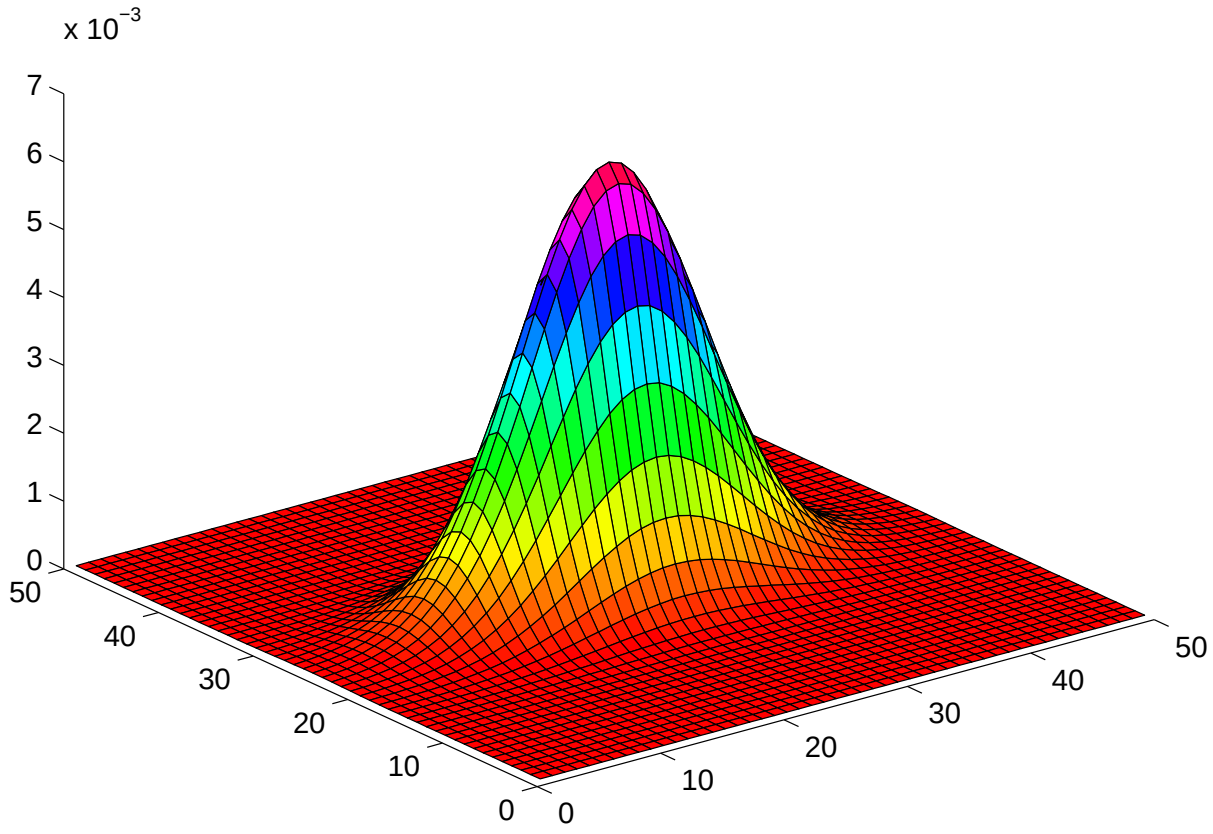

Figure 1: Bivariate normal density.

Semi-parametric: We assume that each class may *not* be modelled as a single Gaussian density function, but that its true density function (likelihood function) may

be approximated using *more than one* Gaussian (or some other type of special function).

Non-parametric methods

Non-parametric methods use all the available data and place a *kernel* function at each point. This is the *Parzen windows* approach. Often this kernel is a Gaussian, whose width may be common to all points and governs the smoothness of the PDF estimate. This may be regarded as a special case of a kernel mixture model where we do not require the locations to be optimised. If we use a variable size (hyper-) sphere around

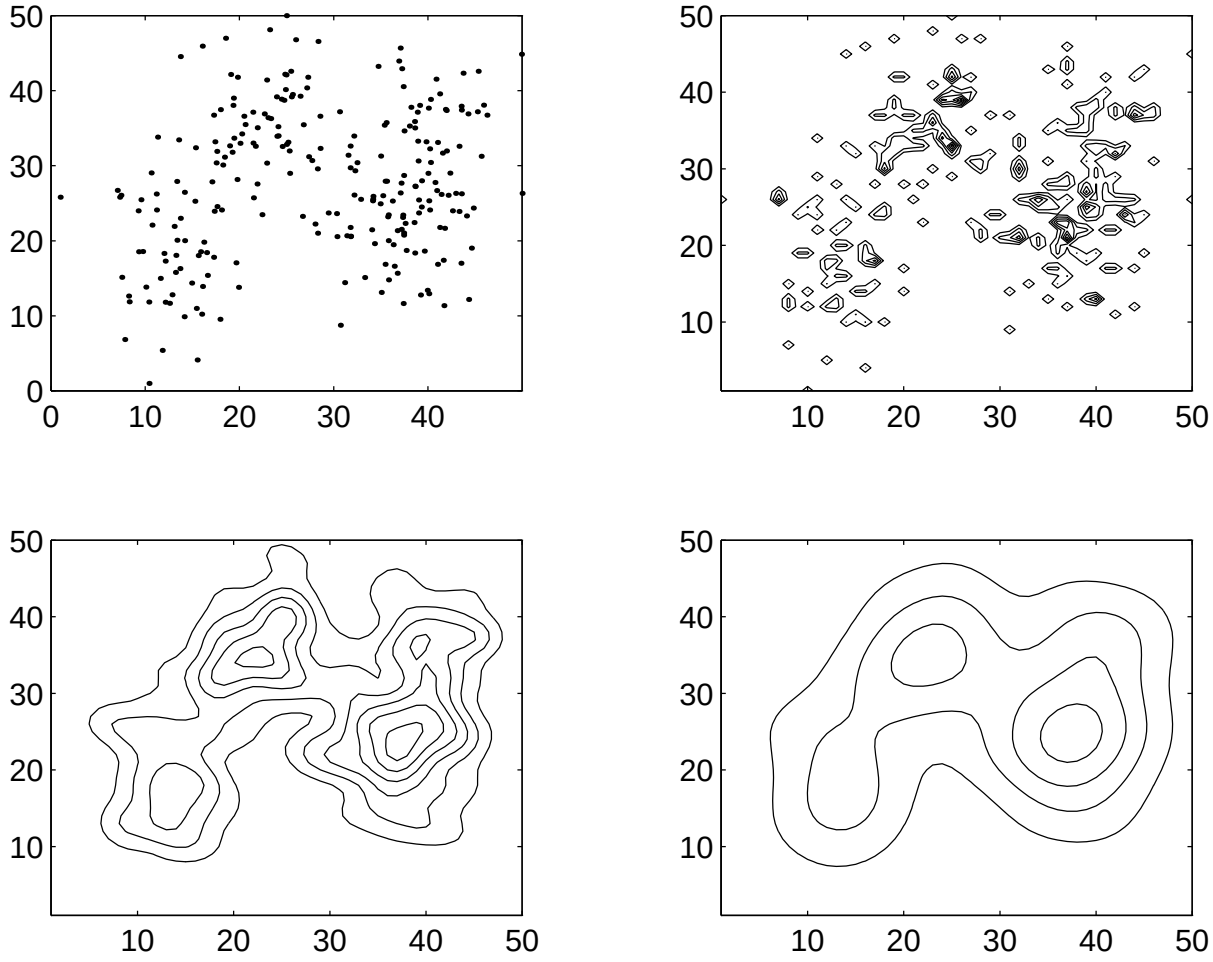

Figure 2: Parzen windows with Gaussians. Data and PDF estimates for widths of 0.5, 2, 4.

each data point, we arrive at.

The K-nearest-neighbour classifier

Basic theory – The probability that a vector, $\mathbf{x}$, drawn from $P(\mathbf{x})$ lies within a region $\mathcal{R}$

$$P(\mathbf{x} \in \mathcal{R}) = \int_{\mathcal{R}} P(\mathbf{x}) d\mathbf{x}$$

if we have a total of N points, then an estimate of this probability is

$$P(\mathbf{x} \in \mathcal{R}) \approx k/N$$

k is number of points in $\mathcal{R}$. Assume that functions are continuous, and variation is small in $\mathcal{R}$

$$P(\mathbf{x} \in \mathcal{R}) \approx P(\mathbf{x})V$$

where V is the (hyper-)volume of $\mathcal{R}$. Combining the above equations gives

$$P(\mathbf{x}) \approx \frac{k}{NV}$$

- Note inherent conflict in assumptions!

Suppose the training set to have n_i vectors for class C_i and n vectors in total. Allow the volume of a small (hyper-)sphere to increase until exactly k points are within it, then estimates may be made...
likelihoods

$$\hat{P}(\mathbf{x} \mid C_i) = \frac{k_i}{n_i V}$$

evidence

$$\hat{P}(\mathbf{x}) = \frac{k}{nV}$$

and priors

$$\hat{P}(C_i) = \frac{n_i}{n}$$

using Bayes' therorem

$$\hat{P}(C_i \mid \mathbf{x}) = \frac{\hat{p}(\mathbf{x} \mid C_i)\hat{P}(C_i)}{\hat{P}(\mathbf{x})} = \frac{k_i}{k}$$

an elegant solution!

If we let $k = 1$ then we get the *nearest-neighbour classifier*. In practise we must search for an 'optimal' value of k, one that gets best results on the *validation set*.

The NN classifier and the Bayes' error

THEOREM
The error rate of the NN classifier is bounded below by the Bayes' error,

$$E_{NN} \geq E_{Bayes}$$

and above by twice the Bayes' error, in the limit of infinite data,

$$E_{NN} \leq 2E_{Bayes}$$

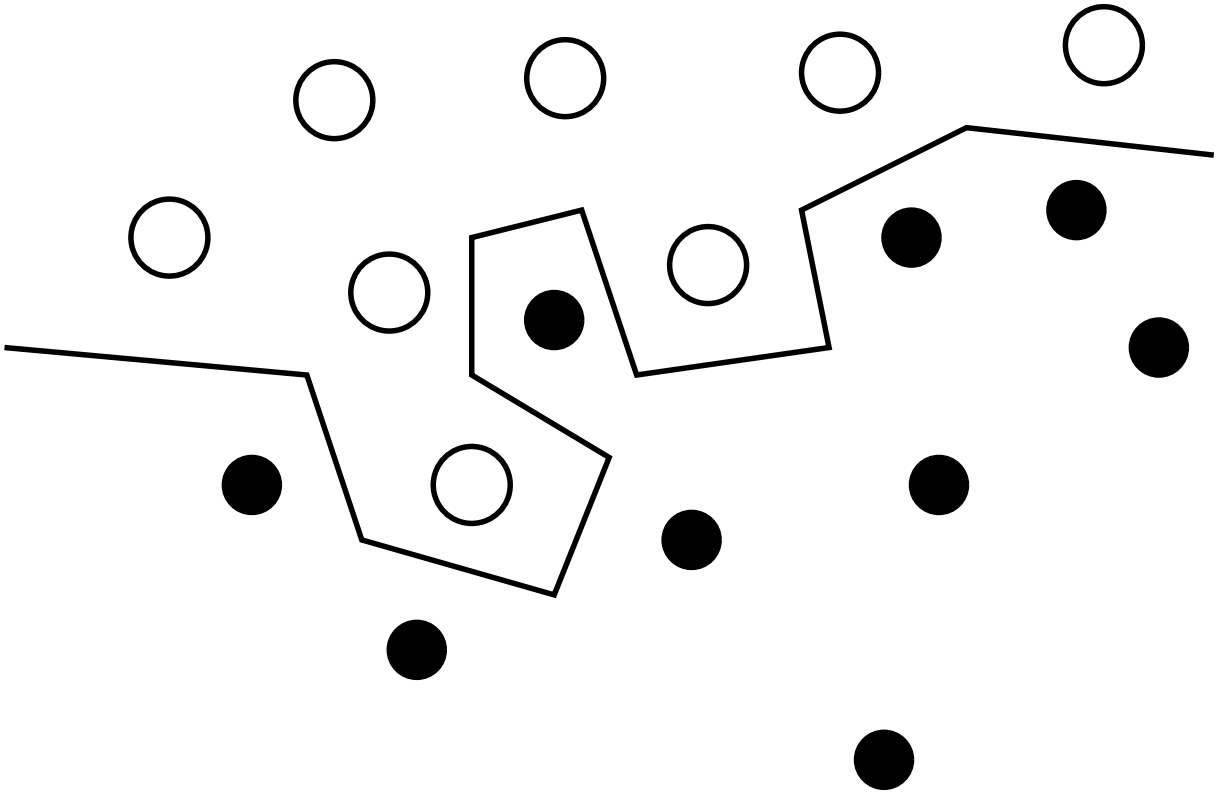

Figure 3: Decision boundary generated by a nearest-neighbour classifier.

PROOF

The proof of the lower bound is trivial. We have already seen that E_{Bayes} represents the lowest error, all classifiers are bounded below by this then.

Let $\mathcal{T}$ be the training set such that $(x', t') \in \mathcal{T}$. The NN rule states that datum x (with true class t) is classified to t' if x' is the NN to x. The error rate on x is hence,

$$E_{NN}(x, x') = P(t \neq t' \mid x, x') = \sum_i P(t = t_i, t' \neq t_i \mid x, x')$$

making independence assumptions this breaks down to:

$$\begin{aligned} E_{NN}(x, x') &= \sum_i P(t = t_i \mid x) P(t' \neq t_i \mid x') \\ &= \sum_i P(t = t_i \mid x)[1 - P(t' = t_i \mid x')] \end{aligned} \tag{1}$$

If we allow the number of points to tend to ∞ then, as x is close to x' (they are NNs) so

$$P(t' = t_i \mid x') \to P(t = t_i \mid x)$$

hence

$$\begin{aligned} E_{NN}(x, x') &\approx \sum_i P(t = t_i \mid x)[1 - P(t = t_i \mid x)] \\ &= 1 - \sum_i P(t = t_i \mid x)^2 \end{aligned} \tag{2}$$

Using the fact that we only classify to the class with the largest posterior, which we assume dominates the summation, and fiddling around a bit more gives:

$$1 - P(t = t^* \mid x)^2 \leq 2[1 - P(t = t^* \mid x)] = 2E_{Bayes}$$

Parametric form

From Bayes' theorem classification goes to class with largest *a posteriori* probability. Taking logs gives

$$\ln P(C_k \mid \mathbf{x}) = \ln P(\mathbf{x} \mid C_k) + \ln P(C_k) - \ln P(\mathbf{x})$$

the decision boundary between classes k and l is when

$$\ln P(C_k \mid \mathbf{x}) = \ln P(C_l \mid \mathbf{x})$$

i.e.

$$\ln P(\mathbf{x} \mid C_k) + \ln P(C_k) = \ln P(\mathbf{x} \mid C_l) + \ln P(C_l)$$

which we will write using a *discriminant function* y as

$$y_k(\mathbf{x}) = y_l(\mathbf{x})$$

for multi-variate Gaussian approximations to each class (parametric case) so,

$$y_k(\mathbf{x}) = -\frac{1}{2}\ln|\mathbf{\Sigma}_k| - \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_k)^T\mathbf{\Sigma}_k^{-1}(\mathbf{x} - \boldsymbol{\mu}_k) + \ln P(C_k)$$

which gives a *quadratic* decision boundary. If the covariance matrices of both classes are equal (to $\mathbf{\Sigma}$ say) then (expanding the terms in brackets)

$$y_k(\mathbf{x}) = -\frac{1}{2}(\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x} - \boldsymbol{\mu}_k^T\mathbf{\Sigma}^{-1}\mathbf{x} - \mathbf{x}^T\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_k + \boldsymbol{\mu}_k^T\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_k) + \ln P(C_k)$$

as $\mathbf{\Sigma}$ is symmetric then so is its inverse and hence

$$\boldsymbol{\mu}_k^T\mathbf{\Sigma}^{-1}\mathbf{x} = \mathbf{x}^T\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_k$$

we thus write

$$y_k(\mathbf{x}) = (\boldsymbol{\mu}_k^T\mathbf{\Sigma}^{-1})\mathbf{x} + (-\frac{1}{2}\boldsymbol{\mu}_k^T\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_k + \ln P(C_k))$$

which is of the form

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

i.e. a *linear discriminant function.*

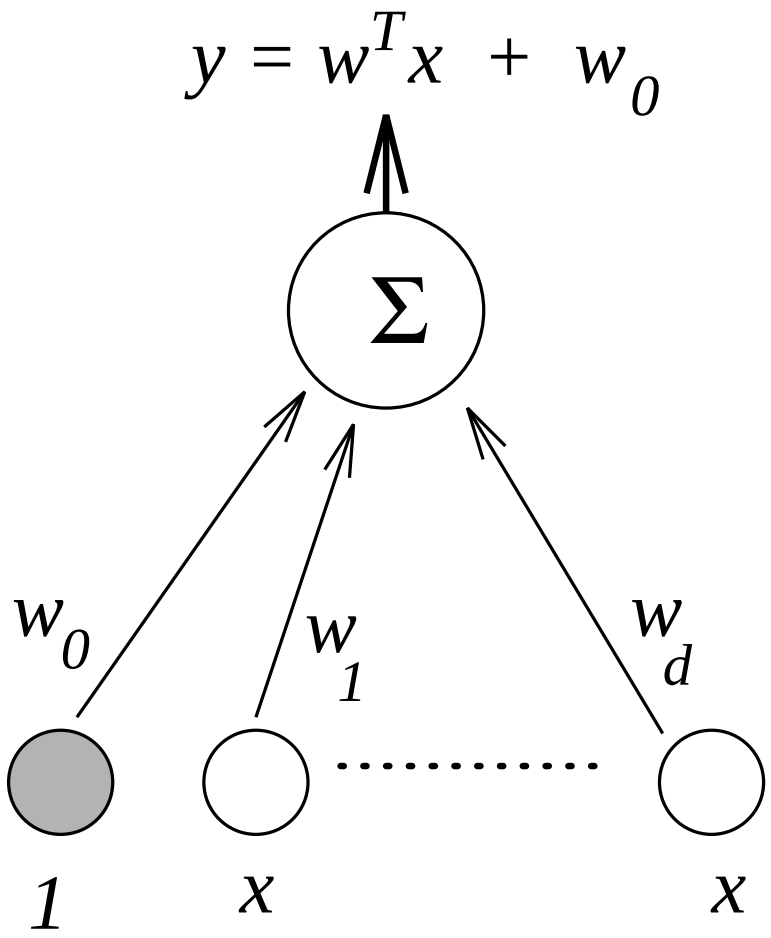

Figure 4: Linear discriminant function.

Linear discriminants - a least-squares approach

We may also consider the setting of weights in for an 'optimal' discriminant via minimisation of an error functional. This error functional is (for LDA) taken to be the sum-of-squares error. You should be familiar with the fact that the LS error function is *convex* with a *single, global* minimum. Good optimisers will get there in one step. We can also pitch the problem as one of matrix psuedo-inversion which is cheap and fast.

For a sum-of-squares error term, the total error may be written as

$$E = \frac{1}{2}\sum_{n=1}^{N}(\mathbf{y}(\mathbf{x}_n) - \mathbf{t}_n)^2$$

$$E = \frac{1}{2}\sum_{n=1}^{N}(\mathbf{w}^\mathsf{T}\mathbf{x}_n - \mathbf{t}_n)^2$$

Differentiation w.r.t. $\mathbf{w}$ gives

$$\sum_{n=1}^{N}(\mathbf{w}^{\mathsf{T}}\mathbf{x}_n - \mathbf{t}_n)\mathbf{x}_n^{\mathsf{T}} = 0$$

rearranging this and putting it into matrix notation gives

$$(\mathbf{X}^{\mathsf{T}}\mathbf{X})\mathbf{W}^{\mathsf{T}} = \mathbf{\Phi}^{\mathsf{T}}\mathbf{T}$$

we wish to solve for $\mathbf{W}^{\mathsf{T}}$ which is obtained from

$$\mathbf{W}^{\mathsf{T}} = \mathbf{X}^{\dagger}\mathbf{T}$$

where $\mathbf{X}^{\dagger}$ is known as the *pseudo-inverse* of $\mathbf{X}$

$$\mathbf{X}^{\dagger} = (\mathbf{X}^{\mathsf{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathsf{T}}$$

Logistic discrimination

Look at a more flexible case where a monotone *non-linear* function g is introduced.

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

Consider parametric likelihoods of Gaussians in a two class problem, with equal covariances. Use a simple discriminant measure

$$a = \ln\left\{\frac{P(\mathbf{x} \mid C_k)P(C_k)}{P(\mathbf{x} \mid C_l)P(C_l)}\right\}$$

From Bayes' theorem

$$P(C_k \mid \mathbf{x}) = \frac{P(\mathbf{x} \mid C_k)P(C_k)}{P(\mathbf{x} \mid C_k)P(C_k) + P(\mathbf{x} \mid C_l)P(C_l)}$$

for the given function a this is equivalent to

$$P(C_k \mid \mathbf{x}) = \frac{1}{1 + \exp(-a)} = g(a)$$

which is a non-linear function known as the *logistic sigmoid function*. Substituting the Gaussian likelihood equations into the above (for a) gives a in the form

$$y_k(\mathbf{x}) = g(a) = g(\mathbf{w}^T\mathbf{x} + w_0)$$

where

$$\mathbf{w} = \mathbf{\Sigma}^{-1}(\boldsymbol{\mu}_k - \boldsymbol{\mu}_l)$$

and

$$w_0 = -\frac{1}{2}\boldsymbol{\mu}_k^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_k + \frac{1}{2}\boldsymbol{\mu}_l^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_l + \ln \frac{P(C_k)}{P(C_l)}$$

this discriminant function allows *probabilities* to be estimated rather than just decision measures – very useful!

It was noticed in the late 1950s that the functional form of the logistic discriminator was *very* similar to that of a biological neuron. The logistic discriminator is also referred to as a *perceptron* or *neuron* for this reason.

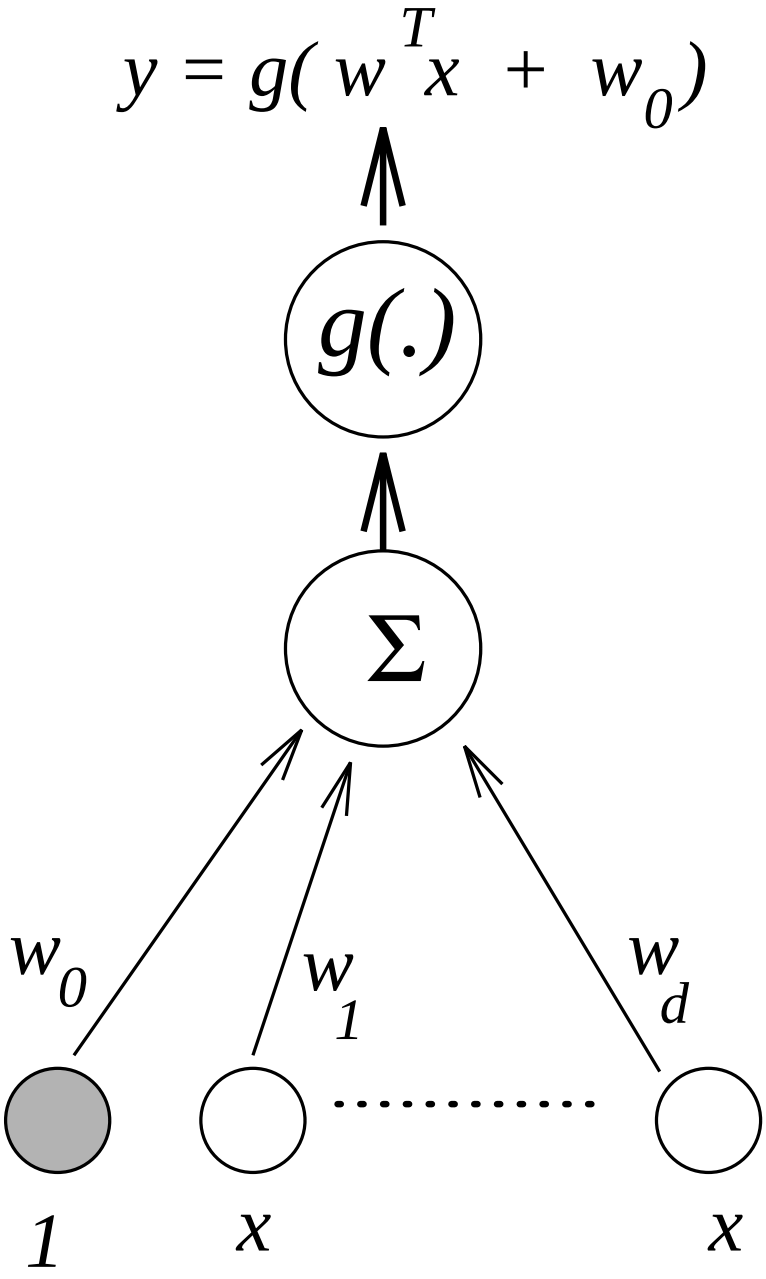

Figure 5: Logistic discrimination.

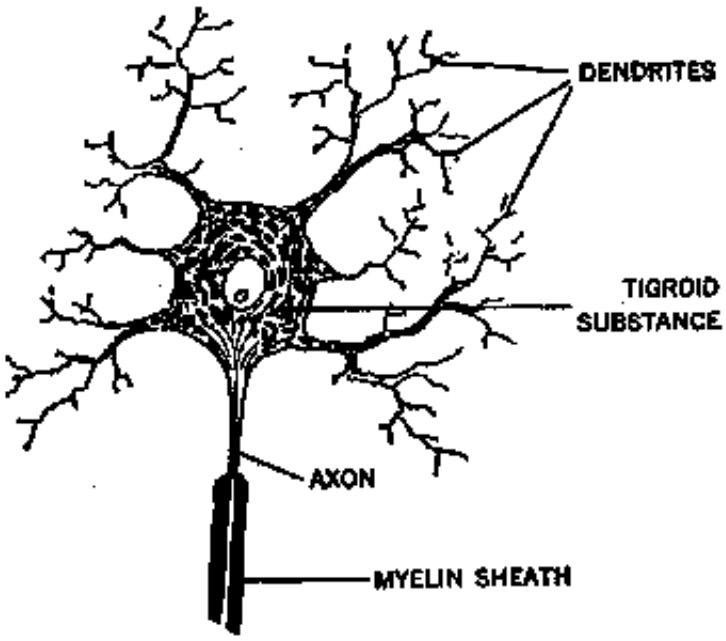

Figure 6: Neuron.

Semi-parametric methods

Later in the course we will see how to model densities using a (small) number of kernel or basis functions. For now we assume that each class-conditional density can be well-modelled using, e.g., a mixture of Gaussians.

Let $\mathcal{G}_k$ be the mixture model for class C_k, hence:

$$P(x \mid C_k) = \sum_{j=1}^{J_k} \pi_j^k P(x \mid g_j^k)$$

from Bayes' mixture theorem. Now consider the posterior,

$$P(C_k \mid x) = P(x \mid C_k)\frac{P(C_k)}{P(x)} = P(x \mid C_k)v^k$$

Combining the above gives,

$$P(C_k \mid x) = v^k \sum_{j=1}^{J_k} \pi_j^k P(x \mid g_j^k)$$

which can be written as:

$$P(C_k \mid x) = \sum_{j=1}^{J_k} w_j^k P(x \mid g_j^k)$$

which can itself be written in vector form as:

$$P(C_k \mid x) = \mathbf{w}_k^\mathsf{T} \boldsymbol{\phi}_k$$

where $\boldsymbol{\phi}_k$ is the vector of responses from the J_k kernels which make up $\mathcal{G}_k$. The form of the above equation is just that of a linear classifier with inputs given by these responses. In this format the classifier is often known as a *mixture classifier*.

We may, of course, make the set of kernels common to all the classes by using a larger model $\mathcal{G} = \cup_k \mathcal{G}_k$. The above formulation is the same, save that the weight vectors $\mathbf{w}_k$ now have a series of zeros. Noting that $\mathcal{G}$ is really just an estimate of the density of $\{x\}$ so we may fit a model to the *class unconditional* data. If $\boldsymbol{\phi}(x)$ is the set of responses of $\mathcal{G}$ then we may consider the classifier:

$$P(C_k \mid x) = \mathbf{w}_k^\mathsf{T} \boldsymbol{\phi} + w_{0,k}$$

where $w_{0,k}$ is a bias (offset) term as in the linear classifier looked at earlier. Giving $\boldsymbol{\phi}$ an extra element, $\phi[0] = 1, \forall x$ lets us write:

$$P(C_k \mid x) = \mathbf{w}_k^\mathsf{T} \boldsymbol{\phi}$$

It turns out that the above forms a general method for function estimation, and that $\mathcal{G}$ does not have to be a density estimate. For example, we could just as easily write for regression:

$$y = \mathbf{w}^\mathsf{T}\boldsymbol{\phi}$$

We started out forming the weights $\mathbf{w}$ from probabilities in this section. This means that the weights are constrained (positive and unity sum). We can relax this constraint (especially if $\mathcal{G}$ is not a density estimator) but the outputs we get may only be discriminants. Following the same argument as for the logistic regressor, we can estimate posteriors by allowing these discriminants to pass through a sigmoidal function.

This system, of forming a *non-linear kernel* representation of the data, followed by a simple transform of the kernel responses is known as the *Radial-Basis Function* or RBF analyser. It is usually referred to as a form of *neural network*. As we have seen, though, there is not much neural about it.

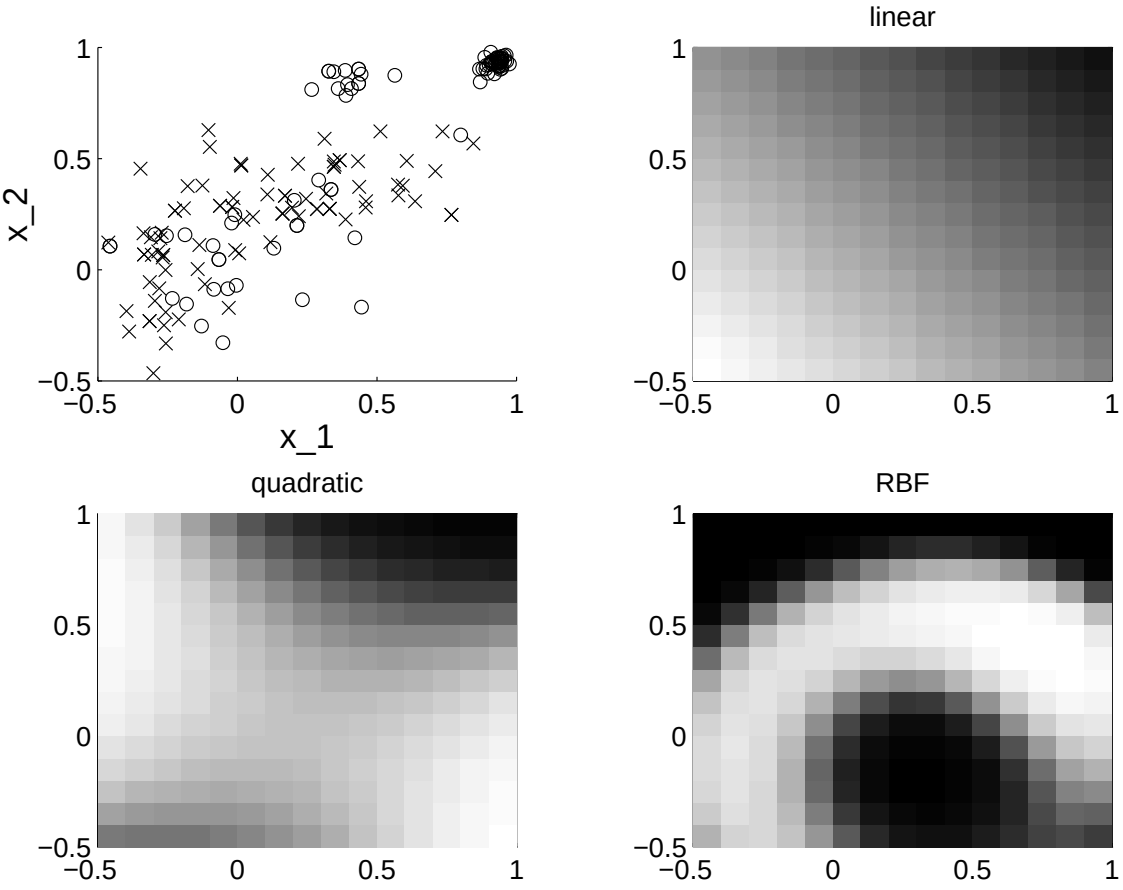

Figure 7: Classification over class ‘non-patient’ of tremor data.