Logistic Regression.

We are trying to directly approximate the posterior using linear discriminant functions.

We have,

$$Q_0(x) = \frac{f_0(x) P_0}{f_0(x) P_0 + f_1(x) P_1}$$

$$= \frac{1}{1 + \exp(-\xi)}$$

$$\xi = -\ln\left(\frac{f_1(x) P_1}{f_0(x) P_0}\right) = \log\left(\frac{f_0(x) P_0}{f_1(x) P_1}\right)$$

Now if $\xi = W^T x + w_0$ [a linear function of x]

then $Q_0(\,)$ can be approximated by

$$\hat{y} = h(W^T x + w_0), \quad h(a) = \frac{1}{1 + e^{-a}}$$

The assumption that $\xi_1 = w^T x + w_0$ is valid if f_0, f_1 are Gaussians.

[also called Gaussian discriminant analysis]

Learning of w is done via LMS algorithm

$$J(w) = \sum_{i=1}^{n} \left[h\left(w^T x_i + w_0\right) - y_i \right]^2$$

Another motivation for LR is

Among $p_0(x), p_1(x)$, Baye's Classifier chooses the maximum.

This is equivalent to looking at the sign of $\log\left[\frac{p_0 f_0(x)}{p_1 f_1(x)}\right]$.

A linear dis fn chooses the classes looking at the sign of $w^T x$. Thus if ξ is approximated (154) using linear fn of x, it is close to logit.

One can also formulate Logistic regression from an ML perspective.

Suppose we model the Posterior using a Logit. ~~Model~~

i.e, $$f(y_i|x_i) = \left(\frac{e^{w^T x_i}}{1+e^{w^T x_i}}\right)^{y_i} \left(\frac{1}{1+e^{w^T x_i}}\right)^{1-y_i}$$

$$\text{Log likelihood} = \sum_{i=1}^{n} \log f(y_i|x_i)$$

$$= \sum_{i=1}^{n} y_i w^T x_i - y_i \log\left(1+e^{w^T x_i}\right)$$

$$- \log\left(1+e^{w^T x_i}\right) + y_i \log\left(1+e^{w^T x_i}\right)$$

$$= \sum_{i=1}^{n} y_i w^T x_i - \log\left(1+e^{w^T x_i}\right)$$

$$\frac{\partial L}{\partial w} = \sum_{i=1}^{n} y_i x_i^T - \left(\frac{e^{w^T x_i}}{1+e^{w^T x_i}}\right) x_i^T$$

To find out the optimal W^*, one can use numerical techniques such as gradient descent.

Multi-class discriminat functions.

We saw LDA for binary problems.

Now suppose we have k classes:

$C_1, C_2 \ldots C_k$.

Question: How to solve this problem?

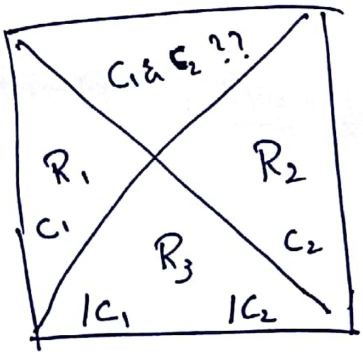
One Possible soln: Learn multiple 2-class classifiers & use one-vs-rest approach.

Eg.

Learn k - binary classifiers & perform a C_i vs not- C_i classifications.

Or learn $\frac{K(K-1)}{2}$ binary classifiers to perform C_i vs C_j [Computationally expensive]

Both of these approaches however leaves ambiguous 'gaps' in the feature space.



similarly for C_i vs C_j

A better approach is to learn K disc fns as follows.

Let us define K fns. $g_s, s=1,\dots,K$

$$g_s(x) = W_s^T x + b_s.$$

Classifier: Assign class C_j to X if

$$g_j(x) \geq g_s(x) \ \forall s.$$

To learn these, one can convert the class-labels into vectors of K-components as follows.

If $X_i \in C_j$, then y_i would be a k-dim vector with j^{th} component 1 & all other zero.

Learn K fn with vector valued targets.

How to generalize this idea with logistic regression?

Recall that we are trying to approximate the posterior with a composite fn of logit & linear fns

In other words, we were seeking which of $f_1(x)P_1$ & $f_0(x)P_0$ were grater.

For a multi-class Baye's classifier,

the class was choosen as following.

if $Q_i(x_i) > Q_j(x_i)$ $\forall j$, then class of x_i was taken to be i.

or we need to find the max of

$Q_i(x) = f_i(x) P_i \; , \; i = 1, \ldots K$

From Bayes rule,

$$Q_j(x) = \frac{f_j(x) P_j}{\sum_s f_s(x) P_s} = \frac{\exp(a_j)}{\sum_s \exp(a_s)}$$

$$a_s = \ln\left[f_s(x) P_s\right]$$

Now, the idea is to approximate

$$a_s = W_s^T x + W_{s0}.$$

[As before, the above will be true if all f_i are Gaussian with same Co-variance]

Essentially, we are seeking the $\underset{a}{\mathrm{argmax}}(a_s)$.

Now, Define a fn $g: R^K \to R^k$ with

$$g(a) = \left[g_1(a) \cdots g_K(a)\right]^T \quad \forall\ j = 1, \ldots, k.$$

$$g_j(a) = \frac{\exp(a_j)}{\sum_s \exp(a_s)}, \quad a = (a_1, \ldots, a_K)^T \in R^K.$$

$g(a)$ is called the soft-max fn.

[The softened version of the argmax fn]

Now, $a_s = W_s^T x \ \forall\ s.$

let W be a matrix with columns as W_s.

After learning, the final classifier will be

$x \in C_j$ if j^{th} component of $g(W^T x)$ is the highest.

(161).

Ideally, when $x \in C_j$, we need

$$g_j(W^T x) = 1 \quad \& \quad g_i(W^T x) = 0 \quad \forall\, i \neq j.$$

Question: How to compute W.

use gradient descent.

$$L(W) = \sum_{i=1}^{n} \| g(W^T x_i) - y_i \|^2.$$

$$x_i \in \mathbb{R}^d, \quad y_i = (y_i^1, \dots y_i^k), \quad y_i^s \in \{0,1\} \; \forall s$$

$$\sum_s y_i^s = 1 \quad \forall i.$$

$$L(W) = \sum_{i=1}^{n} \left[\sum_{s=1}^{k} \left(\frac{\exp(W_s^T x_i)}{\sum_j \exp(w_j^T x_i)} - y_i^s \right)^2 \right]$$

$\downarrow$

different W_s are not decoupled.

use an approximate version of L :

$$L(W) = \sum_{i=1}^{n} \left[\sum_{s=1}^{K} \left(\frac{\exp(W_s^T x_i)}{Z_i} - y_i^s \right)^2 \right]$$

$Z_i = \sum_j \exp\left(W_j^T x_i\right)$ computed using the previous values of W_s.

End of linear familes.