SL : LINEAR CLASSIFIERS

The classification task consists in predicting the unknown label $Y \in \{0,1\}$ (sometimes convenient to encode labels in $\{-1,1\}$) of an observation $X \in \mathcal{X}$ ($= \mathbb{R}^d$; d = dimension of the input space / number of predictors). The prediction task is carried out by constructing a mapping $f_n : \mathcal{X} \to \{0,1\}$ based on a learning sample $\mathcal{L}_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$, where each (x_i, y_i) is an observation of a generic (X, Y), distributed $\sim P = P_{X,Y}$. The performance of f_n is evaluated in terms of the conditional expectation

↳ Risk of f_n

$$R(f_n) = E\{\ell(Y, f_n(X)) \mid \mathcal{L}_n\},$$

where $\ell : \{0,1\} \times \{0,1\} \to \mathbb{R}_+$ is a loss function, which accounts for the error of predicting Y by $f_n(X)$.

In binary classification, the risk of f_n is measured using the 0/1 loss

$$\ell_{0/1}(y, f(x)) = \mathbb{1}(y \neq f(x)) = \begin{cases} 1 & \text{if } y \neq f(x) \\ 0 & \text{otherwise,} \end{cases}$$

so that $R(f_n) = P(Y \neq f_n(X) \mid \mathcal{L}_n)$.

↳ For a fixed learning rule f, $R(f) = P(Y \neq f(X))$.

$r(x) = P(Y=1 \mid X=x)$ → Bayes Classifier, $f^*(x) = \begin{cases} 1 & \text{if } r(x) \geq 1/2 \\ 0 & \text{otherwise,} \end{cases}$ known as

is such that $R(f^*) = R^* = \inf_f R(f)$

↘ The expression of Bayes Classifier suggests that we may estimate the conditional probability $P(Y=1 \mid X=x)$, and use this estimate to construct a plug-in estimator

$$f_n(x) = \begin{cases} 1 & \text{if } \hat{r}(x) \geq 1/2 \\ 0 & \text{otherwise} \end{cases}$$

↑ our estimate of $P(Y=1 \mid X=x) = E(Y \mid X)$.

↳ We have theoretical support for doing this: a 'good' estimate of $r(x)$ yields a 'good' plug-in classifier with small excess risk

$$R(f_n) - R^* \leq 2 \int_{\mathcal{X}} |\hat{r}(x) - r(x)| \, P_X(dx).$$

"discriminative approach" → Eg: Logistic Regression

↘ Alternatively, one may want to estimate the joint probability:

$$P(Y=y, X \in dx) = P(Y=y \mid X=x) \, P(X \in dx)$$

↑ Attach a parametric model here for example, estimate the parameters using Maximum Likelihood, and then use $P(Y=y \mid X=x)$ for predicting the label of X → Eg: LDA.

"generative approach"

↘ We mostly focus on the binary classification task in this chapter, but we indicate how the methods can be extended to the K-class classification problem, $K \geq 3$.

I- GENERALITIES.

I.1. Definition of a linear classifier.

In the context of binary classification, linear classifiers are classifiers which assign a label to a feature point depending on its relative position to a hyperplane.

↘ Definition & Properties of Hyperplanes.

> In a d-dimensional space, a HYPERPLANE $\mathcal{H}$ is a flat affine subspace of dimension (d-1).

Ex:
- In 2 dim, a hyperplane is a line $\beta_0 + \beta_1 x + \beta_2 y = 0$
- In 3 dim, a hyperplane is a plane
- In d dim, a hyperplane is defined by the equation:

$$\beta_0 + \beta_1 x_1 + \cdots + \beta_d x_d = 0.$$

↑ Equality sign is replaced by $\geq$ or $\leq$ for points not in $\mathcal{H}$.

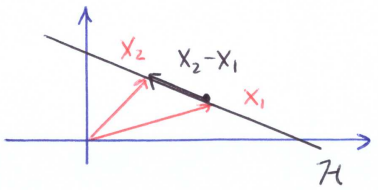
Put $y(x) := \beta_0 + \beta^t x$.
We list two properties of hyperplanes:

> (i) For any two points x_1 and $x_2 \in \mathcal{H}$, $\beta^t(x_1 - x_2) = 0$.
> Hence, $\beta^* = \beta / \|\beta\|$ is a unit vector normal to $\mathcal{H}$.
>
> (ii) The signed distance of any point x to $\mathcal{H}$ is
> $$\frac{1}{\|\beta\|}(\beta_0 + \beta^t x) = \frac{y(x)}{\|\beta\|}$$

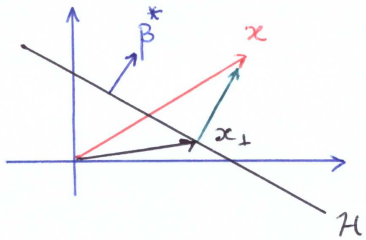
proof = (i) Obviously, $\beta^t(x_1 - x_2) = 0$

$$\langle \beta, x_1 - x_2 \rangle = 0$$

$\Rightarrow \beta$ is perpendicular to $x_1 - x_2$.
Since $x_1, x_2 \in \mathcal{H}$, $x_1 - x_2$ is parallel to $\mathcal{H}$, and we conclude that $\beta \perp \mathcal{H}$.



(ii) Let $x_\perp$ be the orthogonal projection onto $\mathcal{H}$.



$\Rightarrow x = x_\perp + r\beta^*$
↑ Some positive number.
$\beta^* =$ unit vector

↘ $(\times \beta^t)$ and add β_0:

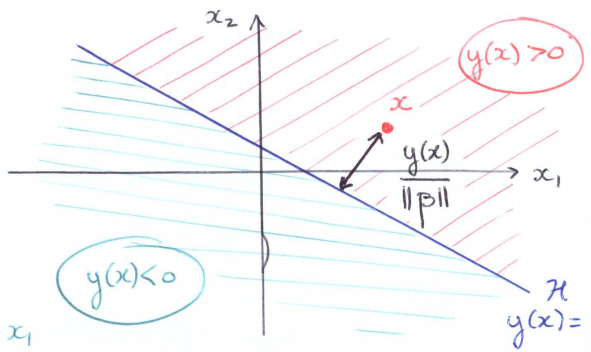
$$\beta_0 + \beta^t x = \beta_0 + \beta^t x_\perp + r\beta^t\beta^*$$
$$y(x) = \underbrace{y(x_\perp)}_{=0 \text{ since } x_\perp \in \mathcal{H}} + r\beta^t\beta^*$$

$$\Rightarrow y(x) = r\beta^t\beta^* = r\frac{\beta^t\beta}{\|\beta\|} = r\|\beta\|,$$

so that $r = \frac{y(x)}{\|\beta\|}$, as required. ∎

↘ Illustration with $d = 2$.

$y(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$, with $\beta_1, \beta_2 > 0$. $\beta = (\beta_1, \beta_2)$



As you decrease x_1 or x_2, $y(x)$ becomes negative, since $\beta_1, \beta_2 > 0$.

⇒ Hyperplane can be used as a decision surface : points such that $y(x) > 0$ are classified as $+1$, and points such that $y(x) < 0$ as 0 (or -1, depending on your notation).

Given $\beta_0 \in \mathbb{R}$, $\beta \in \mathbb{R}^d$, we define the linear classifier $c_{\beta_0,\beta}$ by

$$c_{\beta_0,\beta}(x) = \begin{cases} 1 & \text{if } \beta_0 + \beta^t x \geq 0 \\ 0 & \text{if } \beta_0 + \beta^t x < 0 \end{cases}$$

I.2. Optimal linear risk

- The risk of $c_{\beta_0,\beta}$ is

$$R(c_{\beta_0,\beta}) = \mathbb{E}\, \ell(Y, c_{\beta_0,\beta}(X)) = \mathbb{P}(Y \neq c_{\beta_0,\beta}(X))$$

↑ a fixed classifier ; ↑ under a $\ell_{0/1}$ loss

- The risk of a classifier $\hat{c}_{\beta_0,\beta}$ constructed from $\mathcal{L}_n$ is $R(\hat{c}_{\beta_0,\beta}) = \mathbb{P}(Y \neq \hat{c}_{\beta_0,\beta}(X) \mid \mathcal{L}_n)$.

The optimal linear risk $\bar{R}$ is

$$\bar{R} = \inf_{\beta_0, \beta} R(c_{\beta_0,\beta}) \quad \left(= \inf_{f \in \mathcal{F}} R(f) \right).$$

$\mathcal{F}$ = class of linear classifiers

- The excess risk of $\hat{c}_{\beta_0,\beta}$ is

$$R(\hat{c}_{\beta_0,\beta}) - R^* = \underbrace{\left(R(\hat{c}_{\beta_0,\beta}) - \bar{R} \right)}_{\text{estimation error}} + \underbrace{\left(\bar{R} - R^* \right)}_{\text{approximation error}}$$

R^* : Bayes Risk

- We turn our attention to the approximation error.

For $j \in \{0, 1\}$, put $m_j = \mathbb{E}(X \mid Y = j)$

$$\Sigma_j = \mathbb{E}\left\{ (X - m_j)(X - m_j)^t \mid Y = j \right\}$$

conditional mean & covariance matrix, given $Y = j$.

Proposition: We have

$$R^* \leq \bar{R} \leq \inf_{\beta \in \mathbb{R}^d} \left(1 + \frac{\left[\beta^t (m_1 - m_0) \right]^2}{\left[\sqrt{\beta^t \Sigma_0 \beta} + \sqrt{\beta^t \Sigma_1 \beta} \right]^2} \right)^{-1}$$

In particular, in the case $d = 1$,

$$\bar{R} \leq \left(1 + \left(\frac{m_1 - m_0}{\sigma_1 + \sigma_0} \right)^2 \right)^{-1},$$

where $\sigma_j^2 = \mathbb{E}\left\{ (X - m_j)^2 \mid Y = j \right\}$ is the conditional variance of X given $Y = j$.

The take away message is that whenever the conditional means m_0 and m_1 are far apart (or whenever the covariance matrices are small), the optimal linear risk is relatively close to the optimal risk.

$\Rightarrow$ linear classification is OK.

II - LOGISTIC REGRESSION

II.1. The model & its estimation

- An adaptation of the linear model for regression leads naturally to model $Y \in \{0,1\}$ as

$$Y = \mathbb{1}(\beta_0 + \beta^t X + \varepsilon > 0) \qquad (*)$$

$X \in \mathbb{R}^d$

Noise, independent of X, with distribution $F_\varepsilon(u) = \mathbb{P}(\varepsilon \leq u)$

Theorem. Consider model $(*)$. Suppose that F_ε is continuous and strictly increasing. Then the optimal linear risk $\bar{R}$ is equal to Bayes risk R^*. Moreover, the linear classifier

$$f^*(x) = \begin{cases} 1 & \text{if } \beta_0 + \beta^t x - \sigma^{-1}(\tfrac{1}{2}) \geq 0 \\ 0 & \text{if } \text{---}''\text{---} \quad < 0, \end{cases}$$

where $\sigma(u) := 1 - F_\varepsilon(-u)$ is optimal: $R(f^*) = R^* = \bar{R}$

proof. First, we compute

$$r(x) = P(Y=1 \mid X=x) = \mathbb{P}(\beta_0 + \beta^t x + \varepsilon > 0 \mid X = x)$$

$$\begin{aligned} r(x) &= \mathbb{P}(\beta_0 + \beta^t x + \varepsilon > 0) \qquad \text{since } \varepsilon \text{ is assumed independent of } X \\ &= 1 - P(\beta_0 + \beta^t x + \varepsilon \leq 0) \\ &= 1 - \mathbb{P}(\varepsilon \leq -(\beta_0 + \beta^t x)) \\ &= 1 - F_\varepsilon(-(\beta_0 + \beta^t x)) \\ &= \sigma(\beta_0 + \beta^t x). \end{aligned}$$

Next, note that by assumption both F_ε and σ are invertible.

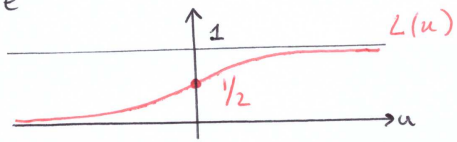
Bayes classifier is $f^*(x) = \begin{cases} 1 & \text{if } r(x) \geq 1/2 \\ 0 & \text{otherwise.} \end{cases}$, and such that $R(f^*) = R^*$.

It remains to see that $r(x) \geq \frac{1}{2} \iff \beta_0 + \beta^t x \geq \sigma^{-1}(1/2)$.

Since $f^*(x) = \begin{cases} 1 & \text{if } \beta_0 + \beta^t x \geq \sigma^{-1}(1/2) \\ 0 & \text{otherwise} \end{cases}$ is linear,

we conclude that $R(f^*) = R^* = \bar{R}$. ∎

- Logistic Regression (LR) assumes that the noise variable ε has distribution $F_\varepsilon = L$, where L is the logistic function defined by $L(u) = \dfrac{e^u}{1+e^u} = (1+e^{-u})^{-1}$.



Facts: (i) $1 - L(-u) = L(u)$ → Because of this property, we denote in the remainder L by σ, since σ was defined on page 7 as $\sigma(u) = 1 - F_\varepsilon(-u)$.

(ii) $L^{-1}(u) = \log\left(\dfrac{u}{1-u}\right)$

(iii) $L^{-1}(1/2) = 0$

With this choice of F_ε, Bayes Classifier reduces to

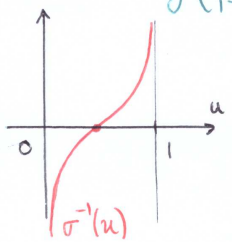
$$f^*(x) = \begin{cases} 1 & \text{if } \beta_0 + \beta^t x \geq 0 \\ 0 & \text{otherwise.} \end{cases}$$

$$\Updownarrow$$

$$P(Y=1 \mid X=x) = \sigma(\beta_0 + \beta^t x)$$

inverse = logit function

$$\sigma^{-1}(u) = \log\left(\frac{u}{1-u}\right)$$



$$\log\left(\frac{P(Y=1 \mid X=x)}{1-P(Y=1 \mid X=x)}\right) = \beta_0 + \beta^t x$$

Starting point of most textbooks: the logit transform of $P(Y=1 \mid X=x)$ behaves linearly with x.

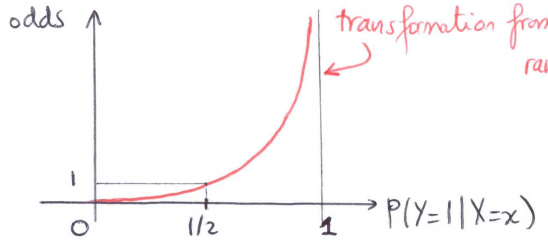
Remarks: (i) Modeling directly $P(Y=1 \mid X=x) = \beta_0 + \beta^t x$ is not a good move: you will end up with estimates of the conditional probability outside of the interval $[0,1]$. Taking the logit transform solves this problem.

(ii) Interpretation of the model.

The ratio $\dfrac{P(Y=1 \mid X=x)}{1-P(Y=1 \mid X=x)} = \dfrac{P(Y=1 \mid X=x)}{P(Y=0 \mid X=x)}$

is usually referred to as the ODDS RATIO. Odds ranges from 0 to ∞, while $P(Y=1 \mid X=x)$ ranges from 0 to 1.

$P(Y=1 \mid X=x)$	0.2	0.4	0.5	0.75	0.9
odds	0.25	0.67	1	3	9



transformation from proba to odds range is $[0, \infty)$.

$\Rightarrow$ Taking the log of the odds show that the range of the log odds is $\mathbb{R}$.

$\hookrightarrow$ Increasing x_j (the j-th predictor) by one unit, keeping all other predictors fixed, multiplies the odds by e^{β_j}.

(iii) Logistic Regression for K-class classification.

$$\log\left(\frac{P(Y=1 \mid X=x)}{P(Y=K \mid X=x)}\right) = \beta_{10} + \beta_{11} x_1 + \cdots + \beta_{1d} x_d$$

$x = (x_1, \cdots, x_d) \in \mathbb{R}^d$

K = our reference class. In binary classification, we have no choice and took 0 as our reference class. In K-class classification, the choice of the reference class is arbitrary.

$$\vdots$$

$$\log\left(\frac{P(Y=K-1 \mid X=x)}{P(Y=K \mid X=x)}\right) = \beta_{(K-1),0} + \beta_{K-1,1} x_1 + \cdots + \beta_{K-1,d} x_d$$

Inverting gives

$$P(Y=j \mid X=x) = \frac{e^{\beta_{j,0} + \cdots + \beta_{j,d} x_d}}{1 + \sum_{\ell=1}^{K-1} e^{\beta_{\ell,0} + \cdots + \beta_{\ell,d} x_d}}$$

$j = 1, \cdots, K-1$

and

$$P(Y=K \mid X=x) = \left(1 + \sum_{l=1}^{K-1} e^{\beta_{l,0} + \cdots + \beta_{l,d} x_d}\right)^{-1}$$

↑ Probabilities sum to one.

In the remainder, we derive an algorithm for parameter estimation in the context of binary logistic regression. The case of K-class classification is left as an exercise.

- **Parameter estimation.**

→ Use Maximum likelihood Estimation:

$$(\hat{\beta}_0, \hat{\beta}) = \underset{\beta_0, \beta}{\operatorname{argmax}}\ \ell(\beta_0, \beta)$$

where

$$\ell(\beta_0, \beta) = \log \prod_{i=1}^{n} P(Y=y_i \mid X=x_i)$$

$$= \sum_{i=1}^{n} \log \left[\sigma_{\beta_0,\beta}(x_i)\right]^{y_i} \left[1 - \sigma_{\beta_0,\beta}(x_i)\right]^{1-y_i},$$

$\sigma_{\beta_0,\beta}(x) := \sigma(\beta_0 + \beta^t x)$

and then construct the plug-in estimator

$$f_n(x) = \begin{cases} 1 & \text{if } \hat{\beta}_0 + \hat{\beta}^t x \geq 0 \\ 0 & \text{otherwise.} \end{cases}$$

→ Maximizing the log-likelihood is equivalent to minimizing $-\ell(\beta_0, \beta)$.

Put $\sigma_i = \sigma(\beta_0 + \beta^t x_i) = P(Y=1 \mid X=x_i)$

So that $P(Y=y_i \mid X=x_i) = \sigma_i^{y_i} (1-\sigma_i)^{1-y_i}$

↑ $y_i \in \{0, 1\}$

$$-\ell(\beta_0, \beta) = -\sum_{i=1}^{n} \left(y_i \log \sigma_i + (1-y_i) \log(1-\sigma_i)\right).$$

We compute the first & second order partial derivatives of $\ell(\beta_0, \beta)$ with respect to β_0 and β. First, we derive the partial derivatives of $\log \sigma_i$ and $\log(1-\sigma_i)$.

- $\dfrac{\partial \log \sigma}{\partial \beta_j} = \dfrac{\partial}{\partial \beta_j} \left\{\log \sigma(\beta_0 + \beta_1 x_1 + \cdots + \beta_d x_d)\right\}$

↑ Work with one observation x: $\sigma := \sigma(\beta_0 + \beta^t x)$

$$= \frac{\partial}{\partial \beta_j} \left\{-\log\left(1 + e^{-(\beta_0 + \beta_1 x_1 + \cdots + \beta_d x_d)}\right)\right\}$$

$$= \frac{x_j e^{-(\beta_0 + \beta^t x)}}{1 + e^{-(\beta_0 + \beta^t x)}}$$

↓ Put $x_0 := 1$

$$\boxed{\frac{\partial \log \sigma}{\partial \beta_j} = x_j (1-\sigma), \quad j = 0, \cdots, d}$$

- Next, $\log(1-\sigma) = -(\beta_0 + \beta^t x) - \log\left(1 + e^{-(\beta_0 + \beta^t x)}\right)$

↑ $1 - \sigma = \dfrac{e^{-u}}{1+e^{-u}}$

⇓

$$\boxed{\frac{\partial \log(1-\sigma)}{\partial \beta_j} = -x_j + x_j(1-\sigma) = -\sigma x_j, \quad j = 0, \cdots, d}$$

Thus

$$\frac{\partial -\ell(\beta_0, \beta)}{\partial \beta_j} = -\sum_{i=1}^{n} \left\{ y_i \frac{\partial \log \sigma_i}{\partial \beta_j} + (1-y_i) \frac{\partial \log(1-\sigma_i)}{\partial \beta_j} \right\}$$

$$\frac{\partial -\ell(\beta_0,\beta)}{\partial \beta_j} = -\sum_{i=1}^{n}\left\{ y_i x_{ij}(1-\sigma_i) + (1-y_i)(-\sigma_i x_{ij})\right\}$$

where $x_i = (x_{i1},\dots,x_{id})^t$, and $x_{i0} = 1$

$$\Rightarrow \quad \frac{\partial -\ell(\beta_0,\beta)}{\partial \beta_j} = \sum_{i=1}^{n}(\sigma_i - y_i)\,x_{ij} = 0$$

In matrix form, $\underset{n\times(d+1)}{X} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1d} \\ 1 & x_{21} & \cdots & x_{2d} \\ \vdots & \vdots & & \vdots \\ 1 & x_{n1} & \cdots & x_{nd} \end{pmatrix} = \begin{pmatrix} 1 & x_1^t \\ 1 & x_2^t \\ \vdots & \vdots \\ 1 & x_n^t \end{pmatrix}$

- $y = (y_1,\dots,y_n)^t$
- $\sigma = (\sigma_1,\dots,\sigma_n)^t$

We get

$$\nabla_{\beta_0,\beta}\{-\ell(\beta_0,\beta)\} = \boxed{X^t(\hat{\sigma} - y) = 0}$$

where $\hat{\sigma} = (\hat{\sigma}_1,\dots,\hat{\sigma}_n)$, $\hat{\sigma}_j = \sigma(\hat{\beta}_0 + \hat{\beta}^t x_j)$

$(d+1)$ non-linear equations $\rightarrow$ solve numerically.

- Compare with Linear Regression + Least Squares:
  $(X^tX)\hat{\beta} = X^ty$
  $\Rightarrow (d+1)$ equations
- In high-dimension, when $d > n$, we need to regularize by adding a penalty term to the log-likelihood.

Before deriving an algorithm to solve $X^t\hat{\sigma} = X^ty$, we turn our attention to the matrix of second order partial derivatives.

$$\frac{\partial^2 -\ell(\beta_0,\beta)}{\partial\beta_j\,\partial\beta_k} = \sum_{i=1}^{n} x_{ij}\,\frac{\partial\sigma_i}{\partial\beta_k}$$

Make use of $\sigma'(u) = \dfrac{e^u}{(1+e^u)^2} = \sigma(u)(1-\sigma(u))$,

so that

$$\frac{\partial\sigma_i}{\partial\beta_k} = \frac{\partial}{\partial\beta_k}\sigma(\beta_0+\beta^t x_i) = x_{ik}\,\sigma_i(1-\sigma_i).$$

$$\Rightarrow \quad \frac{\partial^2 -\ell(\beta_0,\beta)}{\partial\beta_j\,\partial\beta_k} = \sum_{i=1}^{n} x_{ij}\,x_{ik}\,\sigma_i(1-\sigma_i).$$

$$= z_j^t W z_k, \quad \text{where } W = \begin{pmatrix} \sigma_1(1-\sigma_1) & & 0 \\ & \ddots & \\ 0 & & \sigma_n(1-\sigma_n) \end{pmatrix}$$

- $z_j = \begin{pmatrix} x_{1j} \\ \vdots \\ x_{nj} \end{pmatrix}$ $j = 0,\dots,d$

$$\boxed{\nabla^2_{\beta_0,\beta}\{-\ell(\beta_0,\beta)\} = X^tWX}$$

HESSIAN

The Hessian is positive semi-definite: all entries on the diagonal of B are strictly positive.

Put $W^{1/2} = \operatorname{diag}\left(\sqrt{\sigma_i(1-\sigma_i)}\right)$, so that

$$X^tWX = X^tW^{1/2}W^{1/2}X = \|W^{1/2}X\|^2 \geq 0.$$

$\Rightarrow -\ell(\beta_0,\beta)$ is convex, and the solution to $X^t(\hat{\sigma} - y) = 0$ indeed corresponds to a minimum.

$\Rightarrow$ Moreover, efficient algorithms exist to compute $\hat{\sigma}$.

Asymptotic ML theory ensures that the distribution of $n^{1/2}(\hat{\beta} - \beta)$ converges to $\mathcal{N}(0, (X^tWX)^{-1})$

• Computation of the MLE

We solve $X^t \hat{\sigma} = X^t y$ numerically using Newton's method.
Newton's method belongs to the family of descent methods, used for solving the unconstrained minimization problem

$$\text{minimize } f(x), \quad x = (x_1, \dots, x_d) \in \mathbb{R}^d$$

where $f: \mathbb{R}^d \to \mathbb{R}$ is convex & twice continuously differentiable.
A necessary and sufficient condition for a point x^* to be optimal is $\nabla f(x^*) = 0$

↳ A set of d equations in the d variables $x_1, \dots, x_d$ ⇒ not always analytically possible.

Solving the minimization pb is usually performed iteratively, by means of an algorithm computing a sequence of points $x^{(0)}, x^{(1)}, \dots$, such that $f(x^{(k)}) \to f(x^*)$ as $k \to \infty$. Descent methods produce a sequence $\{x^{(k)}\}$ such that

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} \Delta x^{(k)}$$

$\alpha^{(k)}$: step size (>0); $\Delta x^{(k)}$: search direction.

$$f(x^{(k+1)}) < f(x^{(k)})$$

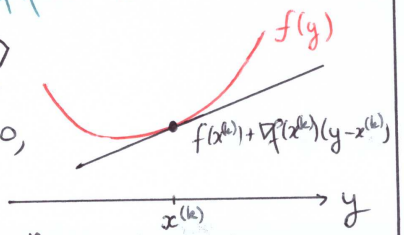
For a convex & differentiable function f,

$$f(y) \geq f(x^{(k)}) + \nabla f(x^{(k)})^t (y - x^{(k)})$$

↖ gradient of f at x_k.

$$f(y) - f(x^{(k)}) \geq \langle \nabla f(x^{(k)}), y - x^{(k)} \rangle$$

If this inner product is ≥ 0, then $f(y) \geq f(x^{(k)})$.



⇒ To get $f(y) < f(x^{(k)})$, necessarily the direction of search $y - x^{(k)}$ must be towards the negative gradient.

• The gradient descent method uses the negative gradient as the search direction:

$$x^{(k+1)} = x^{(k)} - \alpha^{(k)} \nabla f(x^{(k)}).$$

($\alpha^{(k)} > 0$)

↳ A very simple method, but often very slow, and rarely used in practice.

• Alternatively, relying on a first-order Taylor approximation of $f(x+v)$ around x, $f(x+v) \approx f(x) + \nabla f(x)^t v$, we may select the direction v to make the directional derivative $\nabla f^t v$ as negative as possible. Such methods are known as steepest descent methods.

Let $\|.\|$ be a norm on $\mathbb{R}^d$. We define

$$\Delta x_{nsd} = \operatorname{argmin} \{ \nabla f(x)^t v \mid \|v\| \leq 1 \}$$

↑ normalized steepest descent direction

↑ otherwise taking v as large as possible will further decrease the directional derivative

Δx_{nsd} = direction in the unit ball of $\|.\|$ that extends farthest in the direction $-\nabla f(x)$.

Put $\Delta x_{sd} = \|\nabla f(x)\|_* \cdot \Delta x_{nsd}$

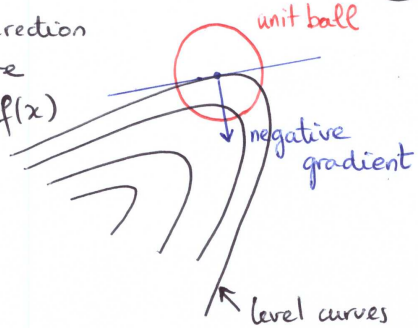
↑ $\|.\|_*$ is the dual norm of $\|.\|$, defined as $\|z\|_* = \sup \{ z^t x \mid \|x\| \leq 1 \}$

The steepest descent algorithm uses:

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} \Delta x_{sd}^{(k)}.$$

Ex: → Steepest descent for Euclidean norm.

The steepest descent direction is simply the negative gradient $\Delta x_{sd} = -\nabla f(x)$

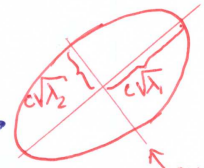


→ Steepest descent for quadratic norm:

$\|z\|_P = (z^T P z)^{1/2} = \|P^{1/2} z\|_2$

P = positive definite matrix.

The quadratic norm can be interpreted in "statistical terms": a unit $\|\cdot\|_P$ - ball is an ellipsoid:



axes are given by the eigenvalues & eigenvectors of P.

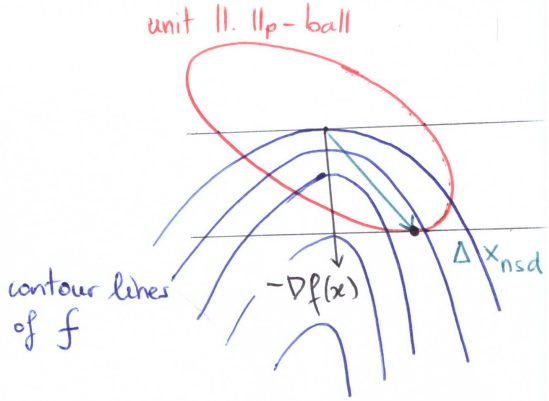
The matrix P can be interpreted as a statistical distance: contour of constant probability density for a multivariate normal distribution:

$$f(x) = \frac{1}{|\det P|(2\pi)^{d/2}} \exp\left\{-\frac{1}{2}\underbrace{(x-\mu)^T P^{-1}(x-\mu)}\right\}$$

$$\{x \in \mathbb{R}^d \mid (x-\mu)^T P^{-1}(x-\mu) = c^2 = \text{constant}\}$$

= ellipsoid centered at μ, with axes $\pm c\sqrt{\lambda_i} e_i$, where $e_i = \lambda_i e_i$, $i = 1, \dots, d$.

The steepest descent in the $\|\cdot\|_P$ norm is given by $\Delta x_{sd} = -P^{-1}\nabla f(x)$.



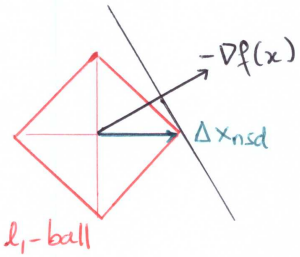
Δx_{nsd} = how far you can go in the $\|\cdot\|_p$-unit ball, in the $-\nabla f(x)$ direction.

→ Steepest descent in the ℓ_1-norm $\|\cdot\|_1$ is given by

$$\Delta x_{sd} = -\left(\frac{\partial f(x)}{\partial x_i}\right) e_i,$$

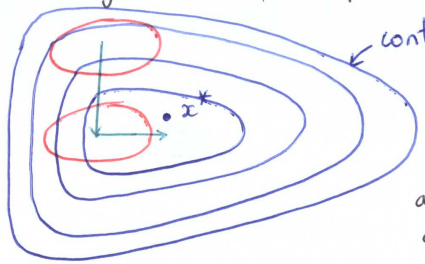
where

$$\left|\frac{\partial f(x)}{\partial x_i}\right| = \|\nabla f(x)\|_\infty$$



⇒ Always along a unit vector: you are updating only one component of the variable at the time: steepest descent in ℓ_1 is also known as a coordinate descent algorithm.

⇒ Choice of a norm for steepest descent ?



Choosing a norm that is aligned with the overall geometry of the sublevel sets, convergence is very fast. (you are going in the right direction)

Near the minimum, the sublevel sets look like ellipsoids:

$$f(x) \approx f(x^*) + \frac{1}{2}(x-x^*)^t \nabla^2 f(x^*)(x-x^*)$$

($\nabla f(x^*)$: gradient vanishes at x^*.)

⇒ The Hessian is telling us how the level curves behave near the optimal point: what we want!

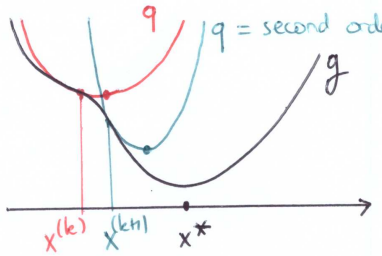
Steepest descent in the norm induced by the Hessian gives Newton's method. $\Delta x_{sd} = -(\nabla^2 f(x))^{-1} \nabla f(x)$.

⇒ Works extremely well.

Alternatively, $x + \Delta x_{sd}$ minimizes the second order approximation of f:

$$f(x^{(k)}+v) \approx \underbrace{f(x^{(k)}) + \nabla f(x^{(k)})v + \frac{1}{2} v^t \nabla^2 f(x^{(k)})v}_{=: q(v)}$$

$$\nabla q(v) = \nabla f(x^{(k)}) + \nabla^2 f(x^{(k)})v$$



q = second order approx of g.

Remark: With quadratic f, Newton works perfectly well, since it converges in one step.

⇒ Expect Newton method to work well when the Hessian of f is slowly varying. Traditional convergence analysis provides bounds on the convergence of Newton's method in terms of lipschitz continuous Hessian of f: $\|\nabla^2 f(x) - \nabla^2 f(y)\|_2 \leq L\|x-y\|_2$.

In particular, it is possible to show that $\exists \eta$ and γ with $0 < \eta \leq \frac{m^2}{L}$, $\gamma > 0$, $\nabla^2 f(x) \geq mI$ (i.e. f is strictly convex) such that:

↘ If $\|\nabla f(x^{(k)})\|_2 \geq \eta$, then (far from optimal point)

$$f(x^{(k+1)}) - f(x^{(k)}) \leq -\gamma$$

↘ If $\|\nabla f(x^{(k)})\|_2 < \eta$, then (close to optimal point)

$$\frac{L}{2m^2}\|\nabla f(x^{(k+1)})\|_2 \leq \left(\frac{L}{2m^2}\|\nabla f(x^{(k)})\|_2\right)^2$$

This result shows that while we are away from the optimal point, we are guaranteed to decrease f by a constant. Since we can have $\|\nabla f(x^{(k)})\|_2 \geq \eta$ only a finite number of times, we eventually enter the second phase of the algorithm, for which the error is squared at each step ⇒ geometric convergence.

More on convex optimization techniques can be found in Boyd & Vandenberghe (2004).

- Back to logistic regression.

Put $\tilde{\beta}^{(t)} = (\beta_0^{(t)}, \beta_1^{(t)}, \dots, \beta_d^{(t)})$

↳ current value of the parameters, after t iterations.

Then

$$\tilde{\beta}^{(t+1)} = \tilde{\beta}^{(t)} - \left(\nabla^2_{\beta_0,\beta} \ell(\tilde{\beta}^{(t)})\right)^{-1} \nabla_{\beta_0,\beta} \ell(\tilde{\beta}^{(t)}).$$

$$\tilde\beta^{(t+1)} = \tilde\beta^{(t)} - (X^t W X)^{-1} X^t (\sigma^{(t)} - y)$$

$$= (X^t W X)^{-1} X^t B \left(X\tilde\beta^{(t)} - W^{-1}(\sigma^{(t)} - y) \right)$$

$$= z^{(t)} \text{ (adjusted response)}$$

$$\boxed{\tilde\beta^{(t+1)} = (X^t W X)^{-1} X^t W z^{(t)}}$$

← W is computed from the current parameter estimates $\tilde\beta^{(t)}$.

Interpretation: $\tilde\beta^{(t+1)}$ = solution to a reweighted least squares problem:

$$= \underset{\tilde\beta \in \mathbb{R}^{d+1}}{\operatorname{argmin}} \left\{ (z^{(t)} - X\tilde\beta)^t W^{(t)} (z^{(t)} - X\tilde\beta) \right\}$$

$$= \underset{\beta_0, \beta}{\operatorname{argmin}} \sum_{i=1}^n w_i^{(t)} \left(z_i^{(t)} - \beta_0 - \beta^t x_i \right)^2$$

where $\begin{cases} w_i^{(t)} = \sigma_i^{(t)}(1 - \sigma_i^{(t)}) \\ W^{(t)} = \operatorname{diag}(w_i^{(t)}) \\ z_i^{(t)} = i\text{-th component of } z^{(t)}. \end{cases}$

Note that weights w_i are computed for the current parameter estimates $(\beta_0^{(t)}, \dots, \beta_d^{(t)})$ ⇒ weights are recomputed at each iteration; the adjusted response as well.

Newton method for Logistic Regression = "Reweighted Least Square"

Remark: Newton algorithm can be derived similarly for K-class logistic regression, and leads to a non-diagonal reweighted least-squares problem.

II.2. Penalized Logistic Regression.

Fast algorithms can be developed as well in the context of penalized log-likelihood:

$$-\ell(\beta_0, \beta) = -\sum_{i=1}^n \log\left[\sigma_{\beta_0,\beta}(x_i)\right]^{y_i} \log\left[1 - \sigma_{\beta_0,\beta}(x_i)\right]^{1-y_i} + \lambda P_\alpha(\beta)$$

where . $\lambda > 0$

. $P_\alpha(\beta) = \frac{1}{2}(1-\alpha)\|\beta\|_2^2 + \alpha\|\beta\|_1$

= elastic-net penalty; a compromise between the ridge penalty ($\alpha = 0$) and the lasso penalty ($\alpha = 1$).

Friedman et al (2010) suggest a coordinate descent algorithm to maximize the penalized log-likelihood (glmnet in R).

The Newton algorithm for maximizing the unpenalized log-likelihood amounts to iteratively reweighted least squares (see page 21).

⇒ If the current estimates are $\tilde\beta^{(t)} = (\beta_0^{(t)}, \beta^{(t)})$, consider a quadratic approximation to the log-likelihood,

$$\boxed{-\ell_Q(\beta_0, \beta) = \sum_{i=1}^n w_i^{(t)} \left(z_i^{(t)} - \beta_0 - \beta^t x_i \right)^2, \text{ where}}$$

- $z_i^{(t)} = i$-th component of $z^{(t)} = X\tilde\beta^{(t)} - B^{-1}(\sigma^{(t)} - y)$
- $w_i^{(t)} = \sigma_{\beta_0^{(t)}, \beta^{(t)}}(x_i)\left(1 - \sigma_{\beta_0^{(t)}, \beta^{(t)}}(x_i)\right)$.
- $B = \operatorname{diag}(w_i^{(t)})$

Newton update is obtained by minimizing $\ell_Q(\beta_0, \beta)$.

The approach of Friedman et al (2010) goes as follow:

For each value of λ, compute the quadratic approximation l_Q about the current parameter estimates $(\beta_0^{(t)}, \beta^{(t)})$, and then use coordinate descent to solve the penalized weighted least-squares problem:

$$\min_{\beta_0, \beta} \left\{ -l_Q(\beta_0, \beta) + \lambda P_\alpha(\beta) \right\}$$

As usual with penalized criterions, one should work with standardized entries: columns of X have zero mean and unit l_1-norm.

Suppose we have estimates $\tilde{\beta}_0$ and $\tilde{\beta}_k$ for $k \neq j$, and we wish to optimize with respect to β_j:

$$f(\beta_j) = \sum_{i=1}^{n} w_i \Big(\underbrace{z_i - \tilde{\beta}_0 - \sum_{k \neq j} \tilde{\beta}_k x_{ik}}_{\tilde{z}_i} - \beta_j x_{ij} \Big)^2 + \alpha\lambda|\beta_j| + \tfrac{1}{2}(1-\alpha)\lambda\beta_j^2 + \text{something indpt of } \beta_j$$

(superscript t omitted)

$\Leftrightarrow$

Minimize

$$f(\beta_j) = \sum_{i=1}^{n} \left(-2 w_i \tilde{z}_i x_{ij} \beta_j + w_i \beta_j^2 x_{ij} \right)^2 + \alpha\lambda|\beta_j| + \tfrac{1}{2}(1-\alpha)\lambda\beta_j^2$$

If $\tilde{z}_i > 0$, then necessarily $\beta_j > 0$; and we need to minimize f on the positive half line:

$$f'(\beta_j) = \sum_{i=1}^{n} -2 w_i \tilde{z}_i x_{ij} + 2 w_i x_{ij}^2 \tilde{\beta}_j + \alpha\lambda + (1-\alpha)\lambda\tilde{\beta}_j = 0$$

$$\Rightarrow \tilde{\beta}_j = \frac{\sum w_i \tilde{z}_i x_{ij} - \alpha\lambda/2}{\sum_{i=1}^{n} w_i x_{ij}^2 + (1-\alpha)\lambda}$$

← provided this quantity is positive.

On $\hat{z}_i > 0$, the solution is

$$\tilde{\beta}_j = \left(\frac{\sum w_i \tilde{z}_i x_{ij} - \alpha\lambda/2}{\sum w_i x_{ij}^2 + (1-\alpha)\lambda} \right)_+$$

Similarly, the solution can be derived when $\tilde{z}_i < 0$.

Summarizing: introducing the soft-thresholding operator $S(z, \gamma) := \text{sign}(z)(|z| - \gamma)_+$, the update on the j-th coordinate is:

$$\tilde{\beta}_j \leftarrow \frac{S\left(\sum_i w_i x_{ij} \left(z_i - \tilde{\beta}_0 - \sum_{k \neq j} \tilde{\beta}_k x_{ik} \right), \lambda\alpha/2 \right)}{\sum_i w_i x_{ij}^2 + \frac{1}{2}(1-\alpha)\lambda}$$

$j = 1, \dots, d$

Once $\tilde{\beta}_1, \dots, \tilde{\beta}_d$ are updated, the intercept can be updated.

For each iteration of the Newton algorithm, we need to cycle through the coordinate descent algorithm. The regularized multinomial regression is treated similarly.

II. 3. Nonparametric Logistic Regression

Suppose for simplicity that $x \in \mathbb{R}$ (one predictor: $d = 1$).

Non-parametric logistic regression makes little assumption on the behaviour of the log odds:

$$\log\left(\frac{P(Y=1 \mid X=x)}{1 - P(Y=1 \mid X=x)} \right) = f(x)$$

A smooth, twice differentiable function.

Consider a penalized likelihood approach to fit the model (and thus preventing overfitting):

$$l(f,\lambda) = \sum_{i=1}^{n} \log\left\{\sigma_i^{y_i}(1-\sigma_i)^{1-y_i}\right\} - \lambda \int |f''(u)|^2 \, du$$

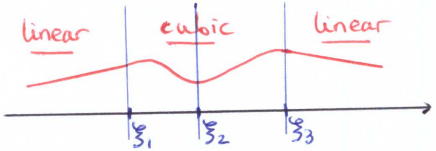
↑ Heavier penalty attached to functions f varying too much ⇒ smoothness is preferred.

Put

$$f^* \in \underset{f}{\operatorname{argmin}}\ l(f,\lambda).$$

↑ Although the problem is ∞-dimensional, it can be shown that the solution is necessarily a natural cubic spline; with n knots, placed at observation points $x_1, \dots, x_n$.

Ex: A natural cubic spline with 3 knots ξ_1, ξ_2, ξ_3:



A NCS is such that smoothness of the first and second derivatives at the knots holds (but not of the third derivative; otherwise the NCS would reduce to a polynomial of degree 3 on the whole region $[\xi_1, \xi_3]$.

⇒ Solution f^* can be expressed in terms of basis functions $\{g_j(x)\}$: $f^*(x) = \sum_{j=1}^{n} \beta_j g_j(x)$, and a Newton procedure can be used here as well to estimate the coefficients β_j. (more on this in the chapter: SL: SPLINES)

II.4. Link with generalized linear models

The Gaussian linear model & logistic regression can be further generalized.

- In linear regression, we assume that

$$Y = \beta_0 + \beta^t X + \varepsilon \quad ; \quad \varepsilon \sim \mathcal{N}(0, \sigma^2).$$

In other words, $Y \sim \mathcal{N}(\mu, \sigma^2)$, where the mean response μ depends linearly on the covariates; $\mu = \beta_0 + \beta^t X$.

- If the response variable is discrete (e.g. count data), a general approach is to model (a transform of) the mean response as a linear combination of the predictors.

Ex:
- Poisson $\sim \mathcal{P}(\lambda)$ $\quad g(\lambda) = \beta_0 + \beta^t X$
- Binomial $\sim \mathcal{B}i(n,p)$ $\quad g(p) = \beta_0 + \beta^t X$.

↑ Called a link function. (NOT UNIQUE)

↑ Modelling directly λ or p as a linear combination $\beta_0 + \beta^t X$ yiels several difficulties, since e.g. we have constraints $\lambda > 0$; $p \in [0,1]$.

Ex of link functions
- Gaussian linear model: identity function $g(x) = x$
- Logistic regression: logistic link $g(x) = \log\left(\frac{x}{1-x}\right)$

- Generalized Linear Models (GLMs) generalizes the approach for response variables belonging to the exponential family.

The exponential family can be represented in several ways. To introduce GLMs, we consider the representation

$$f(y) = \exp\left\{ \frac{y\theta - b(\theta)}{\phi} + c(y;\phi) \right\},$$

Can be a density of a probability mass function

where
- θ, ϕ = parameters
  - ϕ: usually associated with a measure of dispersion ...
  - ... and θ a measure of scale.
- b, c = known functions, such that f integrates to one.

For $Y \sim f$, we can show that $EY = \mu = b'(\theta)$
$\mathrm{Var}\, Y = \phi\, b''(\theta)$

→ Examples:

(i) Gaussian family $\mathcal{N}(\mu, \sigma^2)$.

$$f(y) = \exp\left\{ \frac{y\mu - \mu^2/2}{\sigma^2} - \frac{y^2}{2\sigma^2} - \frac{1}{2}\log(2\pi\sigma^2) \right\}$$

$\Rightarrow$ $\theta = \mu$, $b(\theta) = \frac{1}{2}\theta^2$
$\phi = \sigma^2$, $c(y;\phi) = -\frac{y^2}{2\phi} - \frac{1}{2}(2\pi\phi)$

(ii) Poisson family $\mathcal{P}(\lambda)$

$$f(y) = P(Y=y) = \exp\{ \log\lambda - \lambda - \log y! \}$$

$\Rightarrow$ $\theta = \log\lambda$, $b(\theta) = e^\theta$
$\phi = 1$, $c(y;\phi) = -\log(y!)$

↑ No dispersion parameter since for a Poisson distribution, the mean and variance are both equal to λ.

(iii) Binomial family $Bi(n, p)$

$$f(y) = P(Y=y) = \exp\left\{ y\log\frac{p}{1-p} + n\log(1-p) + \log\binom{n}{y} \right\}$$

$\Rightarrow$ $\theta = \log\left(\frac{p}{1-p}\right)$, $b(\theta) = n\log(1+e^\theta)$
$\phi = 1$, $c(y;\phi) = \log\binom{n}{y}$

↖ Same remark as before.

The parameter θ is usually (a transformed version of) the parameter of interest, to be estimated. GLMs assume that θ varies linearly with the predictors:

$$\theta = \beta_0 + \beta^t X$$
$$= (b')^{-1}(\mu)$$

since we saw page 27 that $\mu = EY = b'(\theta)$

The function $g(u) = (b')^{-1}(u)$ is called the CANONICAL LINK. It leads to desirable statistical properties and tends to be used by default.

Ex: • Gaussian family: $b(\theta) = \frac{1}{2}\theta^2 \Rightarrow b'(\theta) = \theta$ and $g(u) = u$ = Identity link

• Binomial family: $b(\theta) = n\log(1+e^\theta)$
$b'(\theta) = n\frac{e^\theta}{1+e^\theta}$
$g(u) = \log\left(\frac{u}{n-u}\right)$
$\Rightarrow g(np) = \log\frac{np}{n-np} = \log\left(\frac{p}{1-p}\right)$

↑ Mean of a $Bi(n,p)$

↑ LOGISTIC REGRESSION

- Poisson family: $b(\theta) = b'(\theta) = e^{\theta}$

$$g(u) = \log u$$

$\Rightarrow$ GLM for Poisson count is

$$g(\lambda) = \log \lambda = \beta_0 + \beta^t X$$

↑ Modelling λ this way ensures that $\lambda > 0$.

- Parameter estimation: use maximum likelihood.

log-likelihood associate with a sample $(x_1, y_1) \cdots (x_n, y_n)$ of size n, where $y_i \sim$ exponential distribution, modeled using a GLM is:

$$\ell = \sum_{i=1}^{n} \left\{ \frac{y_i \theta_i - b(\theta_i)}{\phi_i} + c(y_i, \phi_i) \right\}$$

where $\theta_i = \beta_0 + \beta^t x_i$. $\quad x_i = (x_{i1}, \cdots, x_{id}) \in \mathbb{R}^d$

$$\rightarrow \frac{\partial \ell(\beta_0, \beta)}{\partial \beta_j} = \sum_{i=1}^{n} \frac{\partial}{\partial \beta_j} \left(\frac{y_i(\beta_0 + \beta^t x_i) - b(\beta_0 + \beta^t x_i)}{\phi_i} \right)$$

$(j = 0, 1, \cdots, d)$

$$= \sum_{i=1}^{n} \frac{x_{ij}}{\phi_i} \left(y_i - b'(\theta_i) \right)$$

(we defined $x_{i0} = 1$ for all i)

$$\rightarrow \frac{\partial^2 \ell(\beta_0, \beta)}{\partial \beta_k \, \partial \beta_j} = - \sum_{i=1}^{n} \frac{x_{ij}}{\phi_i} \frac{\partial}{\partial \beta_k} b'(\theta_i)$$

$(j, k = 0, \cdots, d)$

$$= - \sum_{i=1}^{n} \frac{x_{ij} x_{ik}}{\phi_i} b''(\theta_i)$$

In matrix notation,

$$\nabla_{\beta_0, \beta} \ell(\beta_0, \beta) = X^t (y_\phi - \mu_\phi)$$

$$\nabla^2_{\beta_0, \beta} \ell(\beta_0, \beta) = - X^t B X \quad ,$$

where

- $X = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1d} \\ \vdots & \vdots & & \vdots \\ 1 & x_{n1} & \cdots & x_{nd} \end{pmatrix}$
- $y_\phi = \begin{pmatrix} y_1/\phi_1 \\ \vdots \\ y_n/\phi_n \end{pmatrix}$
- $\mu_\phi = \begin{pmatrix} b'(\theta_1)/\phi_1 \\ \vdots \\ b'(\theta_n)/\phi_n \end{pmatrix}$
- $B = \begin{pmatrix} b''(\theta_1)/\phi_1 & & 0 \\ & \ddots & \\ 0 & & b''(\theta_n)/\phi_n \end{pmatrix}$ ← positive semi-definite.

To maximize the log-likelihood numerically, use Newton method:

$\tilde{\beta}^{(t)} = (\beta_0^{(t)}, \cdots, \beta_d^{(t)})$ = current parameter estimate, after t iterations.

$$\tilde{\beta}^{(t+1)} = \tilde{\beta}^{(t)} - \left[\nabla^2_{\beta_0, \beta} \ell(\tilde{\beta}^{(t)}) \right]^{-1} \left[\nabla_{\beta_0, \beta} \ell(\tilde{\beta}^{(t)}) \right]$$

$$= \tilde{\beta}^{(t)} - (- X^t B X)^{-1} (X^t (y_\phi - \mu_\phi))$$

↑ These use the current parameter estimate to be computed.

$$= (X^t B X)^{-1} X^t B \left(X \tilde{\beta}^{(t)} + B^{-1} (y_\phi - \mu_\phi) \right)$$

$z^{(t)}$ = adjusted response

$$= (X^t B X)^{-1} X^t B \, \tilde{z}^{(t)}.$$

Remark: The adjusted response $z^{(t)}$ is given by $z^{(t)} = X\tilde{\beta}^{(t)} + B^{-1}(y_\phi - \mu_\phi)$, where

$$B^{-1}(y_\phi - \mu_\phi) = \begin{pmatrix} \phi_1/b''(\theta_1) & & 0 \\ & \ddots & \\ 0 & & \phi_n/b''(\theta_n) \end{pmatrix} \begin{pmatrix} (y_1 - b'(\theta_1))/\phi_1 \\ \vdots \\ (y_n - b'(\theta_n))/\phi_n \end{pmatrix}$$

$$= \begin{pmatrix} 1/b''(\theta_1) & & 0 \\ & \ddots & \\ 0 & & 1/b''(\theta_n) \end{pmatrix} \begin{pmatrix} y_1 - b'(\theta_1) \\ \vdots \\ y_n - b'(\theta_n) \end{pmatrix}$$

Note that $g'(\mu) = \dfrac{1}{b''(\theta)}$,

↑ Indeed, $g(u) = (b')^{-1}(u)$

$g'(u) = \left[(b')^{-1}\right]'(u) = \dfrac{1}{b'' \circ (b')^{-1}(u)} = \dfrac{1}{b'' \circ g(u)}$

Since $g(\mu) = \theta$, we get $g'(\mu) = \dfrac{1}{b''(\theta)}$.

so that

$$B^{-1}(y_\phi - \mu_\phi) = \begin{pmatrix} g'(\mu_1) & & 0 \\ & \ddots & \\ 0 & & g'(\mu_n) \end{pmatrix} \begin{pmatrix} y_1 - \mu_1 \\ \vdots \\ y_n - \mu_n \end{pmatrix}$$

↑ where $\mu_i = \beta_0^{(t)} + (\beta^{(t)})^t x_i$

Put $\cdot\ \Gamma = \begin{pmatrix} g'(\mu_1) & & 0 \\ & \ddots & \\ 0 & & g'(\mu_n) \end{pmatrix}$ $\cdot\ y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$ $\cdot\ \mu = \begin{pmatrix} \mu_1 \\ \vdots \\ \mu_n \end{pmatrix}$

ALGORITHM:

Repeat Until Convergence

$$z^{(t)} = X\tilde{\beta}^{(t)} + \Gamma(y - \mu)$$
$$\tilde{\beta}^{(t+1)} = (X^t B X)^{-1} X^t B\, z^{(t)}$$

Examples: × **Bernoulli family.** $Bi(n=1, p)$

Canonical link is $\cdot\ g(u) = \log\left(\dfrac{u}{1-u}\right)$

$g'(u) = \dfrac{1}{u(1-u)}$.

$\cdot\ b(\theta) = \log(1 + e^\theta)$

$b'(\theta) = \dfrac{e^\theta}{1+e^\theta}$

$b''(\theta) = \dfrac{e^\theta}{1+e^\theta}\left(1 - \dfrac{e^\theta}{1+e^\theta}\right)$

$\cdot$ mean is $\mu = p$, so that $g(p) = \beta_0 + \beta^t x$

$\log\left(\dfrac{p}{1-p}\right)$

$\Rightarrow p = \dfrac{e^{\beta_0 + \beta^t x}}{1 + e^{\beta_0 + \beta^t x}} = \sigma(\beta_0 + \beta^t x)$ (in the notation page 8)

$\cdot\ \phi = 1$

Algorithm page 31 → $B = \begin{pmatrix} \sigma_1(1-\sigma_1) & & 0 \\ & \ddots & \\ 0 & & \sigma_n(1-\sigma_n) \end{pmatrix}$

$\sigma_i = \sigma(\beta_0 + \beta^t x_i)$

$\Gamma = \begin{pmatrix} 1/\sigma_1(1-\sigma_1) & & 0 \\ & \ddots & \\ 0 & & 1/\sigma_n(1-\sigma_n) \end{pmatrix} = B^{-1}$

= Algorithm page 21

× **Poisson family** $\mathcal{P}(\lambda)$

Canonical link $\cdot\ g(u) = \log u$

$g'(u) = \dfrac{1}{u}$

- $b(\theta) = b'(\theta) = b''(\theta) = e^{\theta}$
- Mean is $\lambda = b'(\theta) = e^{\theta}$
- $\phi = 1$

$\Rightarrow$ Algorithm page 31 →

$$B = \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix}$$

$$\lambda_i = \exp(\beta_0 + \beta^t x_i)$$

$$\Gamma = \begin{pmatrix} 1/\lambda_1 & & 0 \\ & \ddots & \\ 0 & & 1/\lambda_n \end{pmatrix} = B^{-1}$$

ALGORITHM – POISSON COUNT DATA

Repeat Until Convergence

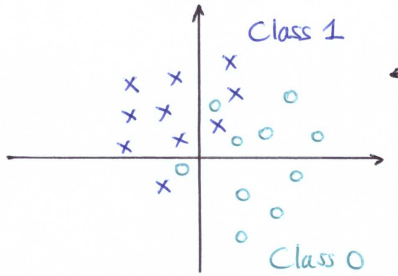
- $\lambda_i^{(t)} = \exp\{\beta_0^{(t)} + (\beta^{(t)})^t x_i\}$
- $B = \begin{pmatrix} \lambda_1^{(t)} & & 0 \\ & \ddots & \\ 0 & & \lambda_n^{(t)} \end{pmatrix}$; $\lambda^{(t)} = \begin{pmatrix} \lambda_1^{(t)} \\ \vdots \\ \lambda_n^{(t)} \end{pmatrix}$
- $z^{(t)} = X\tilde{\beta}^{(t)} + B^{-1}(y - \lambda^{(t)})$
- $\tilde{\beta}^{(t+1)} = (X^t B X)^{-1} X^t B z^{(t)}$

Remark: The exponential family is not limited to the Gaussian, Poisson & Binomial distributions, but include as well
- Gamma / Exponential
- Multinomial
- Inverse Gaussian.

$\Rightarrow$ Algorithm page 31 can be used for parameter estimation for response variable of this type.

II – LINEAR DISCRIMINANT ANALYSIS (LDA)

LDA is a generative approach: classification is based on the modeling of the class conditional distribution as well as feature distribution.



Assume that observations X in class j ($j = 0$ or 1) have density f_j:

$$\mathbb{P}(X \in dx \mid Y = j) = f_j(x)\,dx$$

- In the context of binary classification, we have

$$\mathbb{P}(X \in dx) = \mathbb{P}(X \in dx, Y=0) + \mathbb{P}(X \in dx, Y=1)$$

Law of Total Probability ↑

$$= \mathbb{P}(X \in dx \mid Y=0)\,\mathbb{P}(Y=0) + \mathbb{P}(X \in dx \mid Y=1)\,\mathbb{P}(Y=1)$$

$$= (1-p)\,f_0(x)\,dx + p\,f_1(x)\,dx$$

where we put $\mathbb{P}(Y=1) = p = 1 - \mathbb{P}(Y=0)$ ↑
" ≡ prior probability."

- Goal: classify observations based on the posterior probability $r(x) = \mathbb{P}(Y=1 \mid X=x)$. We have:

$$\mathbb{P}(Y=1 \mid X=x) = \frac{\mathbb{P}(Y=1,\ X \in dx)}{\mathbb{P}(X \in dx)}$$

$$= \frac{\mathbb{P}(X \in dx \mid Y=1)\,\mathbb{P}(Y=1)}{\mathbb{P}(X \in dx)}$$

$$= \frac{p\,f_1(x)}{p\,f_1(x) + (1-p)\,f_0(x)}.$$

The choice of $f_j(x)$ leads to different classification rules:

(i) Gaussian: Linear & Quadratic Discriminant Analysis.

(ii) Mixture of Gaussian: MDA

(iii) Components of $X \in \mathbb{R}^d$ are conditionally independent in each class: $f_j(\underline{x}) = f_{j1}(x_1) \times \dots \times f_{jd}(x_d)$; where $\underline{x} = (x_1, \dots, x_d)$: Naive Bayes.

$\Rightarrow$ We focus on (i); that is when

$$f_j(x) = \frac{1}{(2\pi)^{d/2} (\det \Sigma_j)^{1/2}} \exp\left\{ -\frac{1}{2} (x - m_j)^t \Sigma_j^{-1} (x - m_j) \right\},$$

in the special case where each class shares a common covariance matrix $\Sigma := \Sigma_0 = \Sigma_1$.

- Expression of Bayes Classifier.

Bayes Classifier $f^*(x)$ assigns label 1 to observations with $P(Y=1 \mid X=x) \geq 1/2$. Equivalently,

$$P(Y=1 \mid X=x) \geq P(Y=0 \mid X=x)$$

$$\Leftrightarrow \quad p f_1(x) \geq (1-p) f_0(x)$$

$$\Leftrightarrow \quad \log p + \log f_1(x) \geq \log(1-p) + \log f_0(x)$$

LDA $\Leftrightarrow$

$$\log p - \frac{d}{2} \log 2\pi - \frac{1}{2} \log \det \Sigma - \frac{1}{2} (x - m_1)^t \Sigma^{-1} (x - m_1)$$

$$\geq \log(1-p) - \frac{d}{2} \log 2\pi - \frac{1}{2} \log \det \Sigma - \frac{1}{2} (x - m_0)^t \Sigma^{-1} (x - m_0)$$

$\Leftrightarrow$

$$\log\left(\frac{p}{1-p}\right) - \frac{1}{2}(x - m_1)^t \Sigma^{-1} (x - m_1) + \frac{1}{2}(x - m_0)^t \Sigma^{-1} (x - m_0) \geq 0$$

$\Leftrightarrow$

$$2 \log\left(\frac{p}{1-p}\right) - (x - m_1)^t \Sigma^{-1} (x - m_1) + (x - m_0)^t \Sigma^{-1} (x - m_0) \geq 0$$

$\Leftrightarrow$

$$2 \log\left(\frac{p}{1-p}\right) - \left(x^t \Sigma^{-1} x - 2 m_1^t \Sigma^{-1} x + m_1^t \Sigma^{-1} m_1 \right) + \left(x^t \Sigma^{-1} x - 2 m_0^t \Sigma^{-1} x + m_0^t \Sigma^{-1} m_0 \right) \geq 0$$

↑ quadratic terms cancel out

$\Leftrightarrow$

$$2 \log\left(\frac{p}{1-p}\right) + m_0^t \Sigma^{-1} m_0 - m_1^t \Sigma^{-1} m_1 + 2 (m_1 - m_0)^t \Sigma^{-1} x \geq 0$$

$\Leftrightarrow$

$$\underbrace{2 (m_1 - m_0)^t \Sigma^{-1}}_{=: \beta^t} x + \underbrace{2 \log \frac{p}{1-p} + m_0^t \Sigma^{-1} m_0 - m_1^t \Sigma^{-1} m_1}_{=: \beta_0} \geq 0$$

Conclusion: Under Gaussian assumption, the optimal classifier has linear decision boundaries. This result is summarized in the next proposition.

Proposition: Under the assumption that $P(X \in dx \mid Y = j) = f_j(x)dx$, where f_j is the multivariate normal density with mean m_j and covariance matrix Σ, and that $P(Y=1) = p = 1 - P(Y=0)$, the optimal classifier under a 0/1 loss is linear, and given by

$$f^*(x) = \begin{cases} 1 & \text{if } \beta_0 + \beta^t x \geq 0 \\ 0 & \text{otherwise,} \end{cases}$$

where
- $\beta = 2 \Sigma^{-1} (m_1 - m_0)$
- $\beta_0 = m_0^t \Sigma^{-1} m_0 - m_1^t \Sigma^{-1} m_1 + 2 \log\left(\frac{p}{1-p}\right)$.

• Parameter estimation.

Use maximum likelihood.
We derive the expression of the maximum likelihood estimator in the context of K-class LDA:

→ Prior probabilities $P(Y=j) = p_j$; $j=1,\dots,K$; such that $\sum_{j=1}^{K} p_j = 1$

→ $P(X \in dx \mid Y=j) = f_j(x)\,dx$, where $f_j(x) \sim \mathcal{N}(x \mid m_j, \Sigma)$

• The likelihood associated with a sample $\mathcal{L}_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ of size n is given by

$$L(p, m, \Sigma) = \prod_{i=1}^{n} \prod_{k=1}^{K} p_k^{y_{ik}} \left[f_k(x_i) \right]^{y_{ik}}$$

where

$y_i = (y_{i1}, \dots, y_{iK}) \in \mathbb{R}^K$; $i=1,\dots,n$,

$y_{im} = \begin{cases} 1 & \text{if the } i\text{-th observation belongs to class } m \\ 0 & \text{otherwise.} \end{cases}$

• The log-likelihood is:

$$\ell(p, m, \Sigma) = \sum_{i=1}^{n} y_{i1} \log\left(1 - \sum_{k=2}^{K} p_k\right) + \sum_{i=1}^{n} \sum_{k=2}^{K} y_{ik} \log(p_k) + \sum_{i=1}^{n} \sum_{k=1}^{K} y_{ik} \log \mathcal{N}(x \mid m_k, \Sigma)$$

↑ Maximization of the log-likelihood can be done separately for p, m, and Σ.

↘ Maximization with respect to p.

$$\frac{\partial \ell}{\partial p_m} = \sum_{i=1}^{n} \frac{y_{im}}{p_m} - \sum_{i=1}^{n} \frac{y_{i1}}{1 - \sum_{k=2}^{K} p_k}$$

$m = 2, \dots, K$

It follows that $\dfrac{\sum y_{im}}{p_m} = \dfrac{\sum y_{i1}}{1 - \sum_k p_k}$

$$\Rightarrow \frac{p_m}{1 - \sum_k p_k} = \frac{\sum y_{im}}{\sum y_{i1}} = \frac{n_m}{n_1} \quad ; \quad m = 2, \dots, K,$$

where $n_\ell = \#$ of observations in class $\ell = \sum_{i=1}^{n} y_{i\ell}$.

⇒ p_m is proportional to n_m for all $m = 2, \dots, K$.
+ constraint that the p_m sum to 1, we get

$$\hat{p}_m = \frac{n_m}{n}$$

Remark: Alternatively, introduce the lagrangian function $L(p) = \sum_{i=1}^{n} \sum_{k=1}^{K} y_{ik} \log p_k + \nu \left(\sum_{k=1}^{K} p_k - 1 \right)$.

KKT conditions ⇒ gradient of the lagrangian must vanish at the solution so that p_m must be proportional to $\sum_i y_{im}$ for all m. Primal constraints ensures that $\hat{p}_k = \sum y_{im} / n$.

↘ Maximization with respect to m_j.

We need to consider

$$\sum_{i=1}^{n} \sum_{k=1}^{K} -\frac{1}{2} y_{ik} (x_i - m_k)^t \Sigma^{-1} (x_i - m_k)$$

Toolbox: $\dfrac{\partial m_j^t \Sigma^{-1} m_j}{\partial m_j} = 2 m_j^t \Sigma^{-1}$ $\quad$ $\dfrac{\partial x^t \Sigma^{-1} m_j}{\partial m_j} = x^t \Sigma^{-1}$

$$\Rightarrow \frac{\partial \ell}{\partial m_j} = -\frac{1}{2}\sum_{i=1}^{n} 2\, y_{ij}\,(\hat{m}_j - x_i)^t \Sigma^{-1} = 0$$

$j = 1, \dots, K$

Assuming Σ^{-1} positive definite

$$\Rightarrow \sum_{i=1}^{n} y_{ij}\, x_i = \sum_{i=1}^{n} y_{ij}\, \hat{m}_j$$

$$\boxed{\hat{m}_j = \frac{1}{n_j}\sum_{i=1}^{n} x_i\, 1(y_{ij} = 1)}$$

= average value of the feature points belonging to class j.

↓ Maximization with respect to Σ

We need to maximize

$$\sum_{i=1}^{n}\sum_{k=1}^{K} -\frac{1}{2} y_{ik} \underbrace{(x_i - m_k)^t \Sigma^{-1} (x_i - m_k)}_{\in \mathbb{R} \Rightarrow \text{equal to its trace}} - \frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{K} y_{ik} \log\det\Sigma$$

Moreover, making use of $\mathrm{Tr}(AB) = \mathrm{Tr}(BA)$,

$$= -\frac{1}{2}\left\{ \sum_{i,k} y_{ik}\, \mathrm{Tr}\left\{ \Sigma^{-1}(x_i - m_k)(x_i - m_k)^t \right\} + \log\det\Sigma \underbrace{\sum_{i,k} y_{ik}}_{=n} \right\}$$

Putting $S_k := \frac{1}{n_k} \sum_{i \,|\, y_{ik}=1} (x_i - m_k)(x_i - m_k)^t$

$$= -\frac{1}{2}\left\{ \sum_{k=1}^{K} n_k\, \mathrm{Tr}(\Sigma^{-1} S_k) + n \log\det\Sigma \right\}$$

Toolbox:

- $\frac{\partial}{\partial A}\mathrm{Tr}(AB) = \frac{\partial}{\partial A}\mathrm{Tr}(BA) = B^t$
- $\frac{\partial}{\partial A}\log(\det A) = (A^{-1})^t$
- $\det \Sigma^{-1} = \frac{1}{\det \Sigma}$

Differentiating the expression with respect to Σ^{-1} yields

$$\sum_{k=1}^{K} n_k S_k - n\hat{\Sigma}^t = 0$$

(S_k: symmetric; $\hat{\Sigma}^t = \hat{\Sigma}$)

$$\Rightarrow \boxed{\hat{\Sigma} = \frac{1}{n}\sum_{k=1}^{K} n_k S_k \quad \text{where} \quad S_k = \frac{1}{n_k}\sum_{i \,|\, y_{ik}=1} (x_i - \hat{m}_k)(x_i - \hat{m}_k)^t}$$

$\hat{\Sigma}$ = weighted average of covariance matrices computed separately within the K classes.

Remarks =

(i) Once $\hat{p}_1, \dots, \hat{p}_K$, $\hat{m}_1, \dots, \hat{m}_K$, $\hat{\Sigma}$ are computed, we can calculate the posterior probabilities:

$$\hat{\mathbb{P}}(Y = j \mid X = x) \leftarrow \frac{\hat{p}_j\, \hat{f}_j(x)}{\sum_{k=1}^{K} \hat{p}_k\, \hat{f}_k(x)}, \text{ where}$$

$\hat{f}_j(x) = \mathcal{N}(x \mid \hat{m}_j, \hat{\Sigma})$, and classify a new observation x according to $\underset{1 \le j \le K}{\mathrm{argmax}} \left\{ \hat{\mathbb{P}}(Y = j \mid X = x) \right\}$

The denominator in the expression of $\hat{\mathbb{P}}(Y = j \mid X = x)$ is common to all classes:

$$\log \hat{\mathbb{P}}(Y = j \mid X = x) = C + \log \hat{p}_j + \log \hat{f}_j(x)$$

$$= C' + \log \hat{p}_j - \frac{1}{2}\hat{m}_j^t \hat{\Sigma}^{-1} \hat{m}_j + \hat{m}_j^t \hat{\Sigma}^{-1} x$$

$\vdots$ $\delta_j(x)$

Incorporate into the constant the common term to all classes: $\frac{1}{2} x^t \hat{\Sigma}^{-1} x$

$$\Rightarrow \hat{\mathbb{P}}(Y=j \mid X=x) = C'' e^{\delta_j(x)} = \frac{e^{\delta_j(x)}}{\sum_{\ell=1}^{K} e^{\delta_\ell(x)}}$$

sum up to one ↑ — ↑ The SOFTMAX function.

$$\delta_j(x) = \left(\hat{m}_j^t \hat{\Sigma}^{-1}\right) x + \left(\log \hat{p}_j - \tfrac{1}{2} \hat{m}_j^t \hat{\Sigma}^{-1} \hat{m}_j\right)$$

= linear function

= linear DISCRIMINANT

Summarizing: The final LDA classifier is

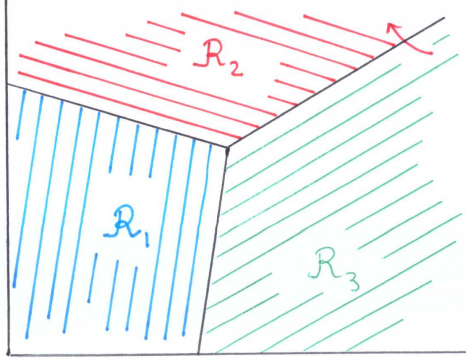
$$\underset{1 \le j \le K}{\operatorname{argmax}}\ \delta_j(x) \iff \underset{1 \le j \le K}{\operatorname{argmax}}\ \frac{\exp(\delta_j(x))}{\sum_{\ell=1}^{K} \exp(\delta_\ell(x))},$$

where $\delta_j(x)$ are linear discriminant functions, given by $\delta_j(x) = \hat{m}_j^t \hat{\Sigma}^{-1} x + \log \hat{p}_j - \frac{1}{2} \hat{m}_j^t \hat{\Sigma}^{-1} \hat{m}_j$.

↑ Later, we revisit the problem of K-class classification, expressing the posterior probabilities as $\frac{e^{f_j(x)}}{\sum_{\ell=1}^{K} e^{f_\ell(x)}}$, but for non-linear discriminant functions $f_j(x)$, see e.g. SL = BOOSTING

Rewriting $\delta_j(x) = \beta_{j0} + \beta_j^t x$, $x, \beta_j \in \mathbb{R}^d$, $\beta_0 \in \mathbb{R}$, the decision boundary between classes j and k is $\{x \mid \delta_j(x) = \delta_k(x)\}$.

The decision regions with linear discriminant boundaries look like:



Points in R_2 are such that $\delta_2(x) > \delta_1(x)$ & $\delta_2(x) > \delta_3(x)$.

Indeed, let $x_1, x_2 \in R_k$, and put $x := \lambda x_1 + (1-\lambda) x_2$, $\lambda \in [0,1]$.

Then adding β_{k0} and multiplying by β_k^t yields

$$\begin{aligned}\beta_{k0} + \beta_k^t x &= \beta_{k0} + \lambda \beta_k^t x_1 + (1-\lambda)\beta_k^t x_2 \\ &= (\lambda + 1 - \lambda)\beta_{k0} + \lambda \beta_k^t x_1 + (1-\lambda)\beta_k^t x_2 \\ &= \lambda\, \delta_k(x_1) + (1-\lambda)\, \delta_k(x_2).\end{aligned}$$

Since $x_1, x_2 \in R_2$, we have that $\delta_k(x_1) > \delta_j(x_1)$, $\delta_k(x_2) > \delta_j(x_2)$, $\forall j \neq k$

It follows that $\delta_k(x) > \delta_j(x)$ $\forall j \neq k \Rightarrow x \in R_k$.

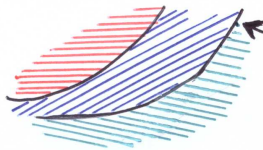
$\Rightarrow$ Regions R_k are convex.

(ii) LDA assumes a common covariance matrix across classes: $f_j(x) = \mathcal{N}(x \mid m_j, \Sigma)$.

Relaxing this assumption and writing $f_j(x) = \mathcal{N}(x \mid m_j, \Sigma_j)$ yields quadratic decision boundaries (terms $x^t \Sigma_j x$ on page 36 do not cancel anymore).

The QUADRATIC DISCRIMINANT FUNCTION is

$$\delta_k(x) = -\frac{1}{2}(x-m_k)^t \Sigma_k^{-1}(x-m_k) + \log p_k - \frac{1}{2}\log(\det \Sigma_k).$$



← decision boundary is $\{x \mid \delta_k(x) = \delta_j(x)\}$.

To obtain non linear decision boundaries using LDA, enlarge the feature space, by including for example variables x_1^2, x_1^3, ...

↳ Quadratic Discriminant Analysis (QDA) works well if the number of variables is small, otherwise you need to estimate large covariance matrices. Alternatively, you may use a Naïve Bayes approach.

(iii) Logistic Regression vs LDA.

Both LR and LDA are such that

$$\log\left(\frac{P(Y=k|X=x)}{P(Y=j|X=x)}\right) = \text{something linear in } x.$$

(compare expressions on pages 10 and 40)

The difference in theses two approaches lies in the way the parameters are estimated: expression of the coefficients β_0, β in LDA are constrained by the model assumptions.

(iv) LDA in high dimension.

Computation of the linear discriminant function involves the inversion of a $d \times d$ matrix.

When $d > n$, the rank of this matrix is at most n, and hence is singular.

⇒ Regualized Discriminant Analysis (RDA) overcomes this issue by requalizing the estimate $\hat{\Sigma}$, and shrinking it towards its diagonal:

$$\hat{\Sigma}(\gamma) = \gamma\,\hat{\Sigma} + (1-\gamma)\,\text{diag}\,\hat{\Sigma},$$

for $\gamma \in [0,1]$.

The choice of γ can be guided by cross-validation.

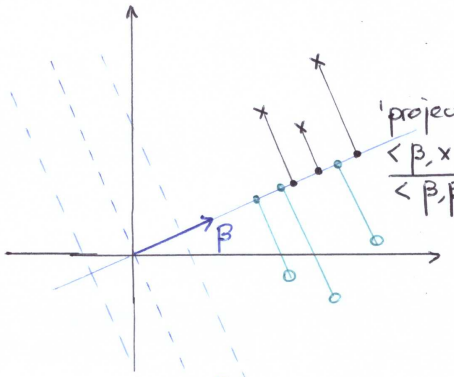
⊕ LDA can achieve very good results, despite its simplicity

⊖ With many dos & predictors, LDA can underfit
Dimension reduction is limited by the number of classes.
With correlated predictor, LDA returns noisy coefficients.

III- FISHER'S LINEAR DISCRIMINANT

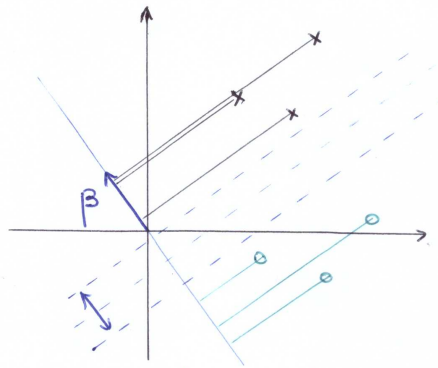
We consider the case of binary classification, with $x \in \mathbb{R}^d$.
Fisher's approach is to project x down to one dimension to achieve class separation.

Compute $\langle \beta, x \rangle = \beta^t x =: z$; where $\beta \in \mathbb{R}^d$.
Then put a threshold β_0 on z. If $z \geq -\beta_0$, classify as '1', if $z < -\beta_0$, classify at '0'.



separating hyperplane defined by $\{x \mid \beta_0 + \beta^t x = 0\}$

⇒ We see that no matter what β_0 is, we achieve bad classification results if we project the data in this direction.



Good separation of classes x and o can be achieved here.

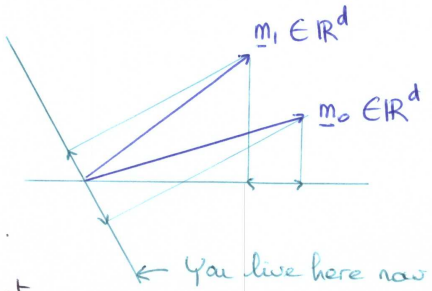
⇒ What makes a 'good' value of β ?

- **First attempt:** Maximize the separation of the projected means of the two classes.

$$\underline{m}_1 = \frac{1}{n_1} \sum_{i|y_i=1} x_i \in \mathbb{R}^d$$

$$\underline{m}_0 = \frac{1}{n_0} \sum_{i|y_i=0} x_i \in \mathbb{R}^d$$

n_i = # elements in class i.



Projected means are $m_1 = \beta^t \underline{m}_1$ and $m_0 = \beta^t \underline{m}_0 \in \mathbb{R}$.

⇒ Maximize $m_1 - m_0 = \beta^t (\underline{m}_1 - \underline{m}_0)$ [Subject to $\|\beta\| = 1$, otherwise $m_1 - m_0$ can be made arbitrarily large].

↳ Use Lagrange multipliers and define the Lagrangian

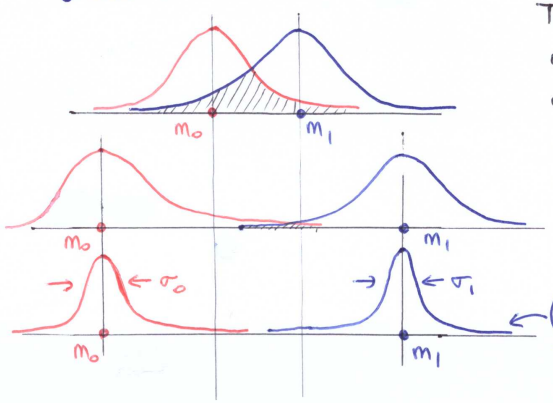
$$L(v) = \beta^t (\underline{m}_1 - \underline{m}_0) + v(\beta^t \beta - 1)$$

$$\nabla_\beta L = \underline{m}_1 - \underline{m}_0 + 2v\beta \Rightarrow \beta = -\frac{1}{2v}(\underline{m}_1 - \underline{m}_0)$$

Optimal direction is proportional to $m_1 - m_0$

$$\boxed{\beta \propto (\underline{m}_1 - \underline{m}_0)}$$

- We can do better.



To further reduce the overlap between the two classes, not only separate the means as much as possible, but also reduce the variance.

← (smoothed) histogram of the projected data

The within class variance of the transformed data $z_i = \beta^t x_i$ is

$$s_j^2 = \sum_{i|y_i=j} (z_i - m_j)^2 \quad , \quad j = 0, 1$$

Goal: Minimize the total within-class variance $s_0^2 + s_1^2$ and at the same time maximize the distance between the projected means. An answer to this, maximize

$$\boxed{\text{FISHER CRITERION} \qquad J(\beta) = \frac{(m_1 - m_0)^2}{s_1^2 + s_0^2}}$$

↳ Remember, R^2 close to R^* when means are far apart, and variances are small (page 6/7)

Alternative expression:

- $$(m_1 - m_0)^2 = [\beta^t (\underline{m}_1 - \underline{m}_0)]^2 = \beta^t (\underline{m}_1 - \underline{m}_0)(\underline{m}_1 - \underline{m}_0)^t \beta =: \beta^t S_B \beta$$

  where $S_B := (\underline{m}_1 - \underline{m}_0)(\underline{m}_1 - \underline{m}_0)^t$ = between class covariance matrix

- $$s_0^2 + s_1^2 = \sum_{y_i=0} [\beta^t (x_i - \underline{m}_0)]^2 + \sum_{y_i=1} [\beta^t (x_i - \underline{m}_1)]^2$$

$$= \sum_{y_i=0} \beta^t (x_i - \underline{m}_0)(x_i - \underline{m}_0)^t \beta + \sum_{y_i=1} (\dots)$$

$$= \beta^t \left\{ \sum_{y_i=0} (x_i - m_0)(x_i - m_0)^t + \sum_{y_i=1} (x_i - m_1)(x_i - m_1)^t \right\} \beta$$

$$=: \beta^t S_W \beta$$

where S_W = within class covariance matrix.

$$\Rightarrow \boxed{J(\beta) = \frac{\beta^t S_B \beta}{\beta^t S_W \beta}}$$

a.k.a. RAYLEIGH QUOTIENT

Optimization task is $\beta^* = \underset{\beta}{\operatorname{argmax}}\, J(\beta)$.

→ **Remark**: Let $\tilde{\beta} = c\beta$ for some $c \neq 0$. Then

$$J(\tilde{\beta}) = \frac{\tilde{\beta}^t S_B \tilde{\beta}}{\tilde{\beta}^t S_W \tilde{\beta}} = \frac{(c\beta)^t S_B (c\beta)}{(c\beta)^t S_W (c\beta)} = \frac{\beta^t S_B \beta}{\beta^t S_W \beta} = J(\beta)$$

⇒ Restrict the maximization task to coefficients β such that $\beta^t S_W \beta = 1$.

Maximizing the Rayleigh quotient is equivalent to:

$$\begin{aligned} &\text{maximize } \beta^t S_B \beta \\ &\text{subject to } \beta^t S_W \beta = 1 \end{aligned} \equiv \max \frac{\beta^t S_B \beta}{\beta^t S_W \beta} \quad \text{s.t. } \beta^t S_W \beta = 1$$

A standard optimization problem.

Lagrangian: $L(\nu) = \beta^t S_B \beta + \nu(\beta^t S_W \beta - 1)$

$$\nabla_\beta L = 2(S_B - \nu S_W)\beta$$

The solution satisfies $S_B \hat{\beta} = \nu S_W \hat{\beta}$

An eigenvalue - eigenvector problem.
⇒ Which eigenvalue ν to choose?

Recall that our goal is to maximize $\beta^t S_B \beta$

$$\Rightarrow (\hat{\beta})^t S_B \hat{\beta} = \nu \underbrace{(\hat{\beta})^t S_W \hat{\beta}}_{=1} = \nu$$

⇒ Select the eigenvector corresponding to the largest eigenvalue. With S_W invertible,

$$\boxed{S_W^{-1} S_B \hat{\beta} = \nu \hat{\beta}}$$

↳ $\hat{\beta}$ = eigenvector associated with the largest eigenvalue of $S_W^{-1} S_B$.

Note that $S_B \hat{\beta} = \nu S_W \hat{\beta}$

$$\| \quad (m_1 - m_0)\underbrace{(m_1 - m_0)^t \hat{\beta}}_{\in \mathbb{R}} \quad \Rightarrow S_W \hat{\beta} \text{ is in the direction of } (m_1 - m_0).$$

⇒ $\hat{\beta}$ is in the direction of $S_W^{-1}(m_1 - m_0)$

$$\boxed{\hat{\beta} \propto S_W^{-1}(m_1 - m_0)}$$

↑ If S_W is proportional to I, $\hat{\beta}$ is in the direction of $m_1 - m_0$.
⇒ S_W adjusts the optimal direction, by taking into account the within-class variances.

Remarks: (i) The projected data is then classified by selecting a threshold β_0.

Ex: Model $\beta^t x_i$ within each class as a Gaussian (CLT in action) as guidance for choosing β_0.

(ii) **Relation to LDA**.

Bayes Classifier under LDA assumptions classifies as 1 if $\beta_0 + \beta^t x \geq 0$, and 0 otherwise, where $\beta \propto \Sigma^{-1}(m_1 - m_0)$ (see expression on page 36). The MLE returns $\hat{\beta} \propto \hat{\Sigma}^{-1}(\hat{m}_1 - \hat{m}_0)$, where $\hat{m}_j := \frac{1}{n_j} \sum_{x_i = j} x_i$, (p. 39)

and $\hat{\Sigma} = \frac{1}{n} \sum_k n_k S_k$ (p. 40)

$$= \frac{1}{n} \sum_k \left\{ \sum_{y_i = k} (x_i - \hat{m}_k)(x_i - m_k)^t \right\}$$

⇒ Fisher & LDA return the same direction!
↳ But LDA provides an expression for β_0.

IV - LEAST SQUARES APPROACH

We adopt a discriminant approach described on page 41. Each class k is described by its discriminant

$$\delta_k(x) = \beta_{k0} + \beta_k^t x \quad , \quad \begin{matrix} k = 1, \dots, K \\ x \in \mathbb{R}^d \end{matrix}$$

These can be grouped together in a matrix form

$$\underset{1\times(d+1)}{\begin{pmatrix} 1 & x^t \end{pmatrix}} \underset{(d+1)\times K}{\begin{pmatrix} \beta_{10} & \cdots & \beta_{K0} \\ | & & | \\ \beta_1 & \cdots & \beta_K \\ | & & | \end{pmatrix}} = \underset{1\times K}{\begin{pmatrix} \delta_1(x) & \cdots & \delta_K(x) \end{pmatrix}}$$

For n observations $\{(x_1, y_1), \dots, (x_n, y_n)\}$, we have:

$y_i = (y_{i1}, \dots, y_{iK})^t \in \mathbb{R}^K$, $i = 1, \dots, n$

$$\underset{\underline{\underline{X}}}{\begin{pmatrix} 1 & - & x_1^t & - \\ & & \vdots & \\ 1 & - & x_n^t & - \end{pmatrix}} \underset{n\times(d+1)}{} \underset{\underline{\underline{B}}}{\begin{pmatrix} \beta_{10} & \cdots & \beta_{K0} \\ | & & | \\ \beta_1 & \cdots & \beta_K \\ | & & | \end{pmatrix}} \underset{(d+1)\times K}{} = \underset{\underline{\underline{\delta}}}{\begin{pmatrix} \delta_1(x_1) & \cdots & \delta_K(x_1) \\ \vdots & & \vdots \\ \delta_1(x_n) & \cdots & \delta_K(x_n) \end{pmatrix}} \underset{(n\times K)}{}$$

In addition, put

- $\underset{\underline{\underline{Y}}}{} = \underset{(n\times K)}{\begin{pmatrix} - & y_1^t & - \\ & \vdots & \\ - & y_n^t & - \end{pmatrix}}$
- $\underset{\underline{\underline{\delta}}}{} = \begin{pmatrix} - & \delta_1^t & - \\ & \vdots & \\ - & \delta_n^t & - \end{pmatrix}$

→ Select coefficients β_{ki} to make $\underline{\underline{\delta}}$ as "close" as possible to $\underline{\underline{Y}}$. Indeed, $\underline{\underline{Y}}$ typically looks like:

$k=5$

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ & \vdots & & \vdots & \end{pmatrix}$$

A high value of the discriminant $\delta_k(x)$ indicates that we are likely to classify x in class k.

⇒ Make $\delta_k(x)$ as large as possible whenever the associated response variable y is $y = (0, \dots, 0, \underset{k}{1}, 0, \dots, 0)$, and $\delta_j(x)$, $j \neq k$ as small as possible.

→ One possible approach = "Least Squares".

Select β_{ki} minimizing

$$\sum_{i=1}^{n} \sum_{k=1}^{K} (\delta_k(x_i) - y_{ik})^2 = \sum_{i=1}^{n} (\delta_i - y_i)^t (\delta_i - y_i)$$

$$\boxed{RSS = Tr\{(XB - Y)(XB - Y)^t\}}$$

→ The solution to this LS problem is

$$\boxed{\hat{B} = (X^t X)^{-1} X^t Y}$$

$$\boxed{\delta := X\hat{B} = X(X^t X)^{-1} X^t Y}$$

Use $\frac{\partial}{\partial A} Tr\,AB = B^t$

$\frac{\partial}{\partial A} Tr(A^t B A) = BA + B^t A$

→ For a new observation x, $\underline{\delta}(x)^t := (\delta_1(x), \dots, \delta_K(x)) = (1 \; x^t)\,\hat{B}$

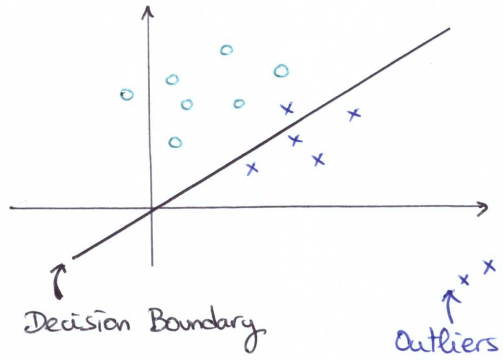
Remarks: (i) If the output variable $y_i \in \mathbb{R}^K$ is such that $a^t y_i + b = 0 \quad \forall i = 1, \dots, n$, then it is possible to show that for any $x \in \mathbb{R}^d$,

$$a^t \delta(x) + b = 0$$

↳ Thus, if we use a 1-of-K coding scheme, then the discriminants returned by the model have the property that $\sum_{k=1}^{K} \delta_k(x) = 1$.

OCTOpoXHLO = the δ_k may be outside the interval $[0, 1]$ and thus cannot be interpreted as probabilities.

(ii) LS solution is not robust to outliers.



Typically, the LS solution can return a non-zero training error, even on linearly separable points; in the presence of outliers.

• **Relation to Fisher's approach**.

We show that for binary classification, under an appropriate coding scheme of the output variable, the solution minimizing Fisher criterion coincides with the least squares solution.

→ Training data $\mathcal{L}_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$, where each observation belongs either to class 0 or 1. Encode the output variable y_i as:

$$y_i = \begin{cases} -n/n_0 & \text{if } x_i \text{ belongs to class } 0 \\ n/n_1 & \text{if } x_i \text{ belongs to class } 1, \end{cases}$$

where n_j = # observations belonging to class j $(j = 0, 1)$.

[Values of y_i correspond to the reciprocal of prior probabilities]

→ Consider first the LS solution, minimizing

$$RSS = \sum_{i=1}^{n} (y_i - \beta_0 - \beta^t x_i)^2$$

Hyperplane $\{x \mid \beta_0 + \beta^t x = 0\}$ is the decision boundary, and is such that $\delta_1(x) = \delta_0(x)$

$$\beta_{10} + \beta_1^t x = \beta_{00} + \beta_0^t x$$

$$\Leftrightarrow \underbrace{(\beta_{10} - \beta_{00})}_{=: \beta_0} + \underbrace{(\beta_1 - \beta_0)^t}_{=: \beta^t} x = 0$$

⇒ In the binary classification problem, we only need to estimate one set of parameters (β_0, β).

$$\begin{cases} \dfrac{\partial RSS}{\partial \beta_0} = -2\sum_{i=1}^{n}\left(y_i - \hat{\beta}_0 - \hat{\beta}^t x_i\right) = 0 \\ \dfrac{\partial RSS}{\partial \beta} = -2\sum_{i=1}^{n}\left(y_i - \hat{\beta}_0 - \hat{\beta}^t x_i\right) x_i = 0 \end{cases}$$

↳ We get from the first relation that

$$n\hat{\beta}_0 = \underbrace{\sum_{i=1}^{n} y_i}_{= -n_0\frac{n}{n_0} + n_1\frac{n}{n_1} = 0} - \sum_{i=1}^{n}\hat{\beta}^t x_i = -\sum_{i=1}^{n}\hat{\beta}^t x_i .$$

$$\Rightarrow \boxed{\begin{aligned} \hat{\beta}_0 &= -\frac{1}{n}\hat{\beta}^t\sum_{i=1}^{n} x_i \\ &= -\hat{\beta}^t m \quad , \text{ where } m := \frac{1}{n}\sum_{i=1}^{n} x_i . \end{aligned}}$$

↳ We get from the second relation that

$$\sum_{i=1}^{n}\left(\hat{\beta}_0 + \hat{\beta}^t x_i\right) x_i = \underbrace{\sum_{i=1}^{n} y_i x_i}_{= \sum_{\text{obs in class 0}} -\frac{n}{n_0} x_i + \sum_{\text{obs in class 1}} \frac{n}{n_1} x_i}$$

$\hat{\beta}_0 = -\hat{\beta}^t m$

$$= n(m_1 - m_0) ,$$

where $m_j := \frac{1}{n_j}\sum_{\text{obs in class } j} x_i$

$$\Downarrow$$

$$\sum_{i=1}^{n}\left(\hat{\beta}^t x_i - \hat{\beta}^t m\right) x_i = n(m_1 - m_0)$$

Note that $m = \frac{1}{n}\sum x_i = \frac{n_0}{n} m_0 + \frac{n_1}{n} m_1$

Thus

$$\begin{aligned} \sum_{i=1}^{n}\hat{\beta}^t(x_i - m)x_i &= \sum_{i=1}^{n}\hat{\beta}^t\left(x_i - \frac{n_0}{n}m_0 - \frac{n_1}{n}m_1\right)x_i \\ &= \sum_i \underbrace{\left(\hat{\beta}^t x_i\right)}_{\in \mathbb{R}} x_i - \frac{n_0}{n}\sum_i \hat{\beta}^t m_0 x_i \\ &\quad - \frac{n_1}{n}\sum_i \hat{\beta}^t m_1 x_i \\ &= \sum_i x_i^t\hat{\beta}\, x_i - \frac{n_0}{n}\hat{\beta}^t m_0\left(\sum_i x_i\right) \\ &\quad - \frac{n_1}{n}\hat{\beta}^t m_1\underbrace{\left(\sum_i x_i\right)}_{= nm = n_0 m_0 + n_1 m_1} \end{aligned}$$

Expanding ↓

$$\begin{aligned} &= \sum_i x_i^t\hat{\beta}\, x_i - \frac{n_0}{n}\underbrace{\hat{\beta}^t m_0}_{\in\mathbb{R}}(n_0 m_0 + n_1 m_1) \\ &\quad - \frac{n_1}{n}\underbrace{\hat{\beta}^t m_1}_{\in\mathbb{R}}(n_0 m_0 + n_1 m_1) \\ &= \sum_i x_i^t\hat{\beta}\, x_i - \frac{n_0^2}{n} m_0^t\hat{\beta} m_0 - \frac{n_0 n_1}{n} m_0^t\hat{\beta} m_1 \\ &\quad - \frac{n_0 n_1}{n} m_1^t\hat{\beta} m_0 - \frac{n_1^2}{n} m_1^t\hat{\beta} m_1 . \end{aligned}$$

↳ Cross-product terms :

Introduce $S_B := (m_1 - m_0)(m_1 - m_0)^t$

Then

$$\begin{aligned} S_B\hat{\beta} &= (m_1 - m_0)(m_1 - m_0)^t\hat{\beta} \\ &= m_1^t\hat{\beta} m_1 + m_0^t\hat{\beta} m_0 \\ &\quad - m_0^t\hat{\beta} m_1 - m_1^t\hat{\beta} m_0 \end{aligned}$$

$$\Downarrow$$

Cross-product terms are equal to

$$\frac{n_0 n_1}{n} S_B\hat{\beta} - \frac{n_0 n_1}{n}\left(m_0^t\hat{\beta} m_0 + m_1^t\hat{\beta} m_1\right)$$

Thus

$$\sum_{i=1}^{n} \hat{\beta}^t (x_i - m)\, x_i = \sum_i x_i^t \hat{\beta}\, x_i - \left(\frac{n_0^2}{n} + \frac{n_0 n_1}{n}\right) m_0^t \hat{\beta}\, m_0 - \left(\frac{n_1^2}{n} + \frac{n_0 n_1}{n}\right) m_1^t \hat{\beta}\, m_1 + \frac{n_0 n_1}{n} S_B \hat{\beta}$$

(n_0 and n_1 are the number of observations in class 0 and class 1)

$$= \sum_i x_i^t \hat{\beta}\, x_i - n_0 m_0^t \hat{\beta} m_0 - n_1 m_1^t \hat{\beta} m_1 + \frac{n_0 n_1}{n} S_B \hat{\beta}$$

$$= \sum_{\text{obs in class 0}} x_i^t \hat{\beta}\, x_i - n_0 m_0^t \hat{\beta} m_0 + \sum_{\text{obs in class 1}} x_i^t \hat{\beta}\, x_i - n_1 m_1^t \hat{\beta} m_1 + \frac{n_0 n_1}{n} S_B \hat{\beta}$$

($x_i^t \hat{\beta} \in \mathbb{R}$, $m_1^t \hat{\beta} \in \mathbb{R}$)

$$= \sum_{\text{obs in class 1}} x_i (x_i^t \hat{\beta}) - n_1 m_1 (m_1^t \hat{\beta})$$

$$= \sum_{\text{obs in class 1}} (x_i x_i^t - m_1 m_1^t)\, \hat{\beta}$$

$$= \left\{ \sum_{\text{obs in class 0}} (x_i x_i^t - m_0 m_0^t) + \sum_{\text{obs in class 1}} (x_i x_i^t - m_1 m_1^t) \right\} \hat{\beta} + \frac{n_0 n_1}{n} S_B \hat{\beta}$$

Observe that

$$\sum_{\text{obs in class 1}} (x_i - m_1)(x_i - m_1)^t = \sum x_i x_i^t - \sum x_i m_1^t - \sum m_1 x_i^t + \sum m_1 m_1^t$$
$$= \sum x_i x_i^t - 2 n_1 m_1 m_1^t + n_1 m_1 m_1^t$$
$$= \sum x_i x_i^t - n_1 m_1 m_1^t$$
$$= \sum (x_i x_i^t - m_1 m_1^t)$$

$\Rightarrow$ Defining $S_W := \sum_{\text{obs in class 0}} (x_i - m_0)(x_i - m_0)^t + \sum_{\text{obs in class 1}} (x_i - m_1)(x_i - m_1)^t$, we obtain:

$$\sum_{i=1}^{n} \hat{\beta}^t (x_i - m)\, x_i = \left(S_W + \frac{n_0 n_1}{n} S_B \right) \hat{\beta} .$$

Together with the relation established at the bottom of page 53, we arrive at

$$\left(S_W + \frac{n_0 n_1}{n} S_B \right) \hat{\beta} = n (m_1 - m_0) . \quad (*)$$

Our LS solution.

→ Comparison with Fisher solution.

Since $S_B \hat{\beta} = (m_1 - m_0)\underbrace{(m_1 - m_0)^t \hat{\beta}}_{\in \mathbb{R}}$ is always in the direction of $(m_1 - m_0)$, we conclude that the least squares solution satisfying $(*)$ is always in the direction of $S_W^{-1}(m_1 - m_0)$.

Compare with the solution to Fisher's approach, established on page 48 $\Rightarrow$ these are exactly the same.

→ Advantage of LS here is that an expression for the bias term $\hat{\beta}_0 = -\hat{\beta}^t m$ is also available.

References.

Friedman J, Hastie T, and Tibshirani R. (2010) Regularization Paths for Generalized Linear Models via Coordinate Descent. Journal of Statistical Software. Vol 33, Issue 1