Chapter 4. Linear Methods for Classification

ESL2 review

오영민

Introduction

- In classification, predicted values in a discrete set $\mathcal{G}$.

- Consider linear methods (decision boundaries are linear):

1. Linear Regression of an Indicator Matrix
2. Linear Discriminant Analysis(LDA)
3. Logistic Regression
4. Perceptron Learning
5. Optimal separating hyperplane

Linear Regression of an Indicator Matrix

- **Each of the response categories are coded via an indicator variable.**
- **Suppose** $\mathcal{G} = \{1, ..., K\}$, $Y = (Y_1, ..., Y_K)$ **where** $Y_k = 1 \text{ if } G = K \text{ else } 0$
- **N-training instances of these form** $\mathbf{Y} \in \mathbb{R}^{N \times K}$

- **Then the fit is given by**

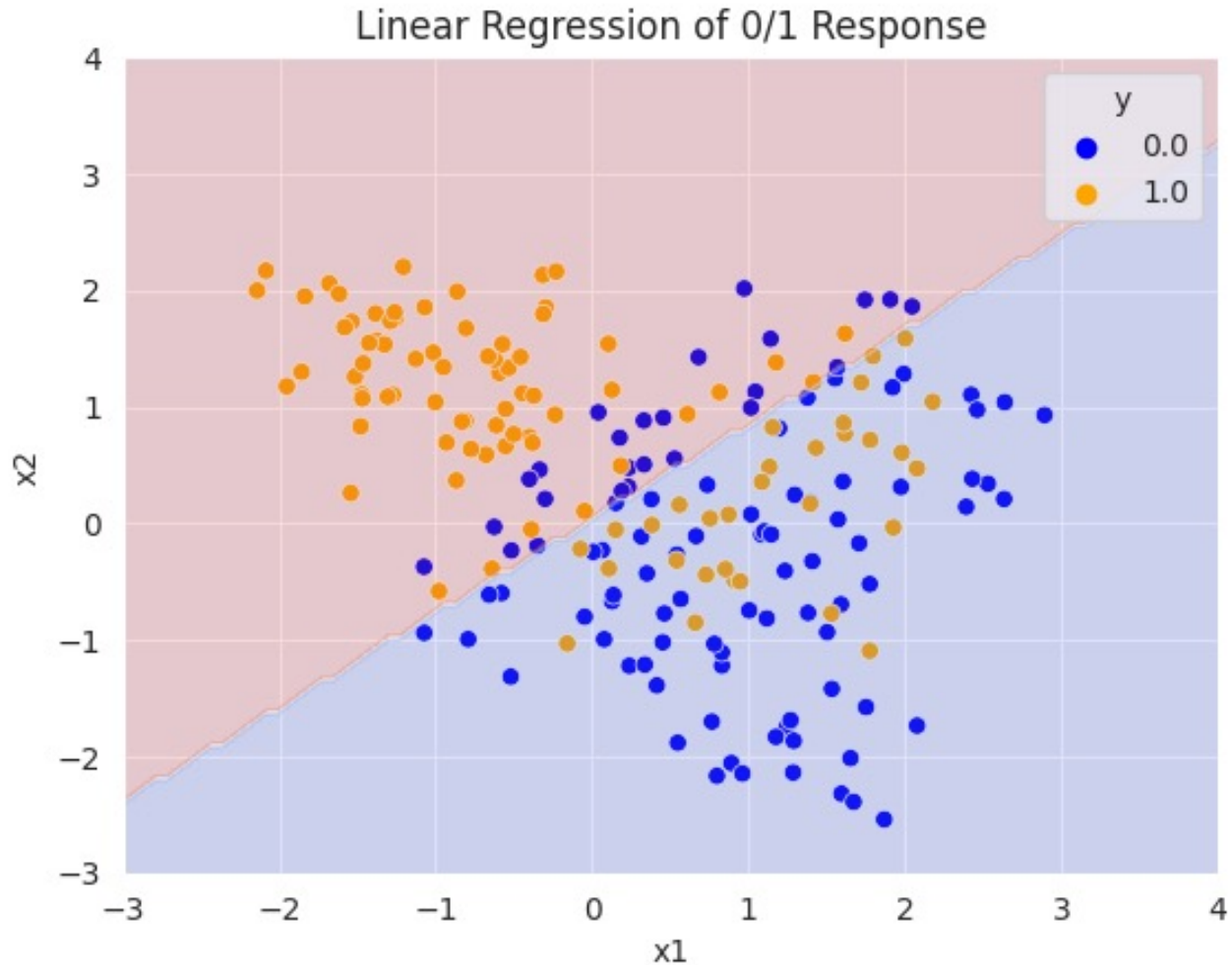
$$\hat{\mathbf{Y}} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y} = \mathbf{X}\hat{\mathbf{B}}$$

- **A new observation with input** x **is classified as follows:**
  1. Compute $\hat{f}(x)^T = (1, x^T)\hat{\mathbf{B}} = (\hat{f}_1(x), ..., \hat{f}_K(x)) \in \mathbb{R}^{1 \times K}$
  2. $\hat{G}(x) = \underset{k \in \mathcal{G}}{\text{argmax}} \hat{f}_k(x)$

In chapter2,

```
desinged_X = np.c_[np.ones(len(y)),X]
beta_hat = np.linalg.inv(desinged_X.T @ desinged_X) @ desinged_X.T @ y

X_new_with_bias = np.c_[np.ones([len(X_new), 1]), X_new]
y_pred = X_new_with_bias @ beta_hat
y_pred[y_pred <= 0.5] = 0
y_pred[y_pred > 0.5] = 1
```



Linear Regression of an Indicator Matrix

- **For** $Y_k,\ \mathbb{E}(Y_k|X=x) = Pr(G=K|X=x)$
- $\hat{f}_k(x)$ **estimates** $Pr(G=k|X=x)$ **rather than conditional expectation.**
- **For designed matrix** $\mathbf{X}$**, let** $\mathbf{X} = (1_N\ \mathbf{x})$ where $\mathbf{x} = (x_1, ..., x_N)^T$
  Then, $\mathbf{X}^T\mathbf{X} = \begin{pmatrix} \mathbf{1}_N^T \\ \mathbf{x}^T \end{pmatrix} (\mathbf{1}_N \quad \mathbf{x}) = \begin{pmatrix} N & \mathbf{1}_N^T\mathbf{x} \\ \mathbf{x}^T\mathbf{1}_N & \mathbf{x}^T\mathbf{x} \end{pmatrix}$

$$\sum_{k\in\mathcal{G}} \hat{f}_k(x) = (1, x^T)\hat{\mathbf{B}}\mathbf{1}_K$$

$$= (1, x^T)(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}\mathbf{1}_K$$

$$= (1, x^T)\begin{pmatrix} N & \mathbf{1}_N^T\mathbf{x} \\ \mathbf{x}^T\mathbf{1}_N & \mathbf{x}^T\mathbf{x} \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{1}_N^T \\ \mathbf{x}^T \end{pmatrix} \mathbf{1}_N$$

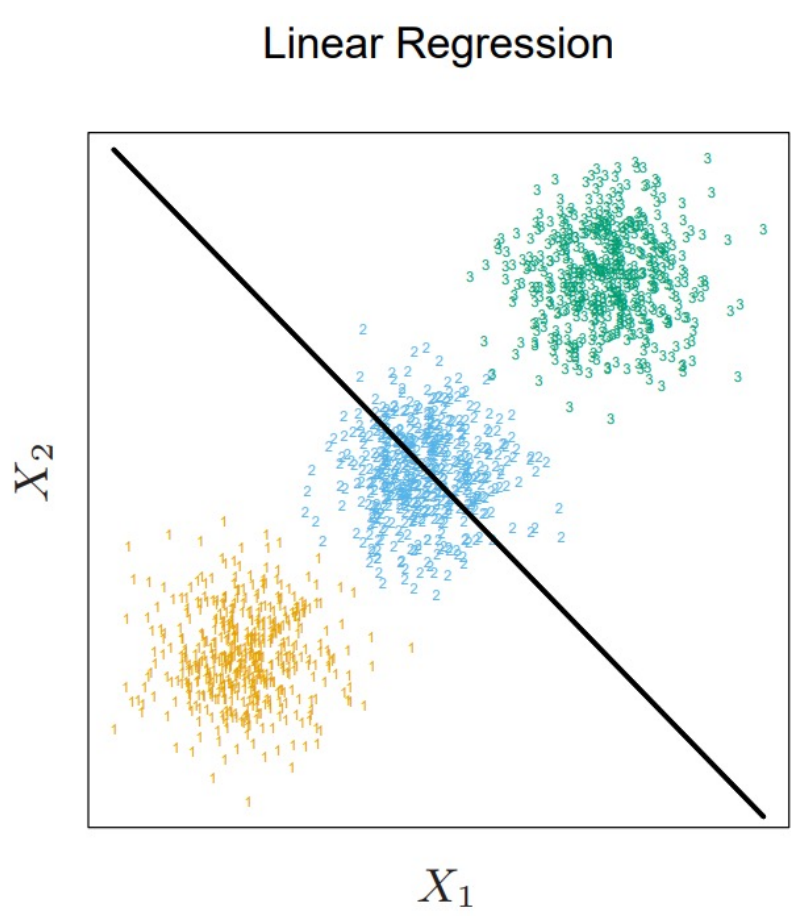
$$= (1, x^T)\begin{pmatrix} N & \mathbf{1}_N^T\mathbf{x} \\ \mathbf{x}^T\mathbf{1}_N & \mathbf{x}^T\mathbf{x} \end{pmatrix}^{-1} \begin{pmatrix} N \\ \mathbf{x}^T\mathbf{1}_N \end{pmatrix} = (1, x^T)\begin{pmatrix} 1 \\ \mathbf{0}_p \end{pmatrix} = 1$$

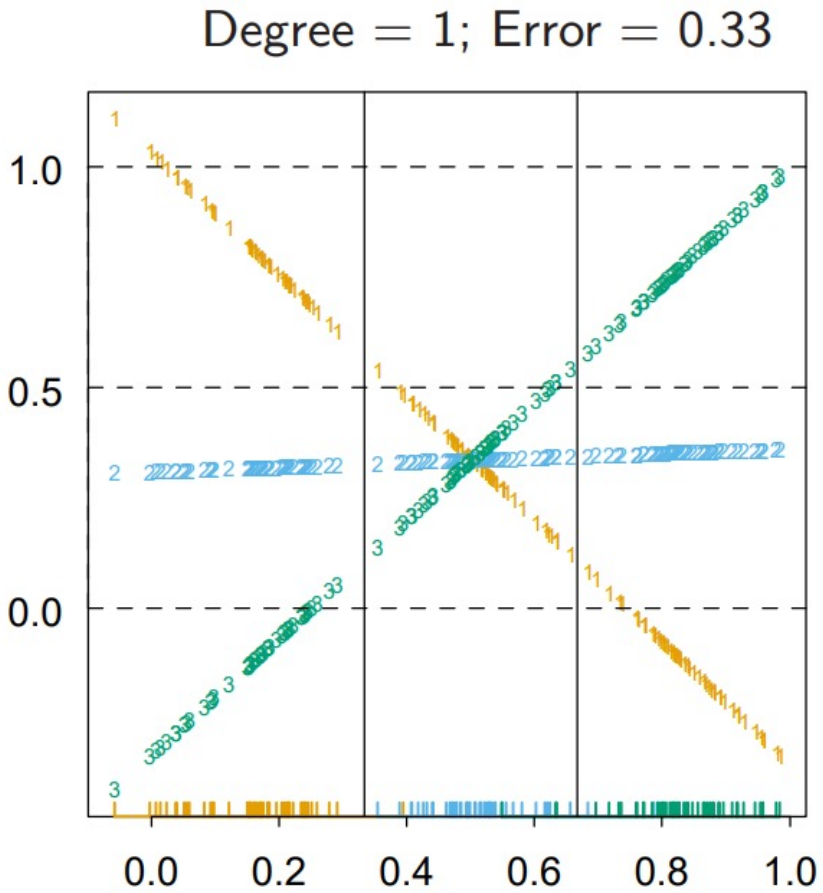
Note. $\hat{f}_k(x) \notin [0, 1]$

Alternative.
Use softmax!

Linear Regression of an Indicator Matrix

- **Masking effect: class 2 is completely masked (never dominates).**





Linear Discriminant Analysis(LDA)

- **In chapter 2, Bayes classifier is best estimator for 0-1 loss**

Statistical Decision Theory: classification

- Thus, it suffices to minimize EPE pointwise:

$$\hat{G}(x) = argmin_{g \in \mathcal{G}} \sum_{k=1}^{K} L(\mathcal{G}_k, g) Pr(G = \mathcal{G}_k | X = x)$$

- Now define loss function by $L(Y, f(X)) = I(Y \neq f(X))$, which is called 0-1 loss function.
- With the 0-1 loss function,

$$\begin{aligned}
\hat{G}(x) &= argmin_{g \in \mathcal{G}} \sum_{g \neq \mathcal{G}} Pr(G = \mathcal{G}_k | X = x) \\
&= argmin_{g \in \mathcal{G}} \sum_{g \neq \mathcal{G}} Pr(G = \mathcal{G}_k | X = x) + Pr(G = g | X = x) - Pr(G = g | X = x) \\
&= argmin_{g \in \mathcal{G}} 1 - Pr(G = g | X = x) = argmax_{g \in \mathcal{G}} Pr(g | X = x)
\end{aligned}$$

this solution is known as **Bayes classifier**

- **Thus, we need to know the class posteriors** $Pr(G|X)$

Linear Discriminant Analysis(LDA)

- **Let** $\pi_k = Pr(G = k),\ f_k(x) = Pr(X = x|G = k)$
- **Since** $\mathcal{G}$ **is discrete,**

$$Pr(X = x) = \sum_{k\in\mathcal{G}} Pr(X = x|G = k)Pr(G = k)$$

- **By Bayes theorem,**

$$Pr(G = k|X = x) = \frac{Pr(X=x|G=k)Pr(G=k)}{Pr(X=x)} = \frac{f_k(x)\pi_k}{\sum_{l=1}^{K} f_l(x)\pi_l}$$

- **In LDA,**

$$f_k(x) = \mathcal{N}(\mu_k, \Sigma_k) = \frac{1}{(2\pi)^{p/2}|\Sigma_k|^{1/2}} \exp(-\frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k))$$

$$\Sigma_k = \Sigma\ \forall k \quad \textbf{i.e.}\ \hat{G}(x) = \underset{k\in\mathcal{G}}{\operatorname{argmax}} f_k(x)\pi_k = \underset{k\in\mathcal{G}}{\operatorname{argmax}} -\frac{1}{2}(x-\mu_k)^T\Sigma^{-1}(x-\mu_k) + \log\pi_k$$

Linear Discriminant Analysis(LDA)

- **For** $k, l \in \mathcal{G}$,

$$\begin{aligned}\log \frac{\Pr(G=k|X=x)}{\Pr(G=\ell|X=x)} &= \log \frac{f_k(x)}{f_\ell(x)} + \log \frac{\pi_k}{\pi_\ell} \\ &= \log \frac{\pi_k}{\pi_\ell} - \frac{1}{2}(\mu_k+\mu_\ell)^T \mathbf{\Sigma}^{-1}(\mu_k-\mu_\ell) \\ &\quad + x^T \mathbf{\Sigma}^{-1}(\mu_k-\mu_\ell),\end{aligned}$$

- **Note**.

$$\{x : Pr(G=k|X=x) = Pr(G=l|X=x)\} = \{x : \log \frac{Pr(G=k|X=x)}{Pr(G=l|X=x)} = 0\}$$

i.e. all the decision boundaries are linear

Linear Discriminant Analysis(LDA)

- **Linear discriminant functions:**

$$\delta_k(x) = x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \log \pi_k$$

- **In practice, estimate μ_k, Σ, π_k using our training data:**

$$\hat{\pi}_k = \frac{N_k}{N} \text{ where } N_k \text{ is the number of class-k observations}$$

$$\hat{\mu}_k = \frac{1}{N_k} \sum_{g_i = k} x_i$$

$$\hat{\Sigma} = \sum_{k=1}^{K} \sum_{g_i = k} \frac{1}{N - k} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T \text{ (pooled covariance, within-class covariance)}$$

derivation of the LDA direction via LS

- **Suppose we have features $x \in \mathbb{R}^p$, a two-class response, coded as $-\frac{N}{N_1}, \frac{N}{N_2}$**
- **LDA rule classifies to class 2 if**

$$x^T\Sigma^{-1}(\hat{\mu}_2 - \hat{\mu}_1) > \tfrac{1}{2}\hat{\mu}_2^T\Sigma^{-1}\hat{\mu}_2 - \tfrac{1}{2}\hat{\mu}_1^T\Sigma^{-1}\hat{\mu}_1 + \log\tfrac{N_1}{N} - \log\tfrac{N_2}{N}$$

and class 1 otherwise

- **Consider least square problem,** $\begin{pmatrix} N & \mathbf{1}_N^T\mathbf{x} \\ \mathbf{x}^T\mathbf{1}_N & \mathbf{x}^T\mathbf{x} \end{pmatrix} \begin{pmatrix} \hat{\beta}_0 \\ \hat{\beta} \end{pmatrix} = \begin{pmatrix} \mathbf{1}_N^T\mathbf{Y} \\ \mathbf{x}^T\mathbf{Y} \end{pmatrix}$

Note that $\sum_{i=1}^N x_i = \sum_{i:g_i=1} x_i + \sum_{i:g_i=2} x_i = N_1\hat{\mu}_1 + N_2\hat{\mu}_2,\ \sum_{i=1}^N Y_i = 0,$

$$\sum_{i=1}^N x_i Y_i = -\tfrac{N}{N_1}N_1\hat{\mu}_1 + \tfrac{N}{N_2}N_2\hat{\mu}_2 = N(\hat{\mu}_2 - \hat{\mu}_1)$$

derivation of the LDA direction via LS

- $\hat{\Sigma} = \frac{1}{N-2}\left(\sum_{i:g_i=1}(x_i-\hat{\mu}_1)(x_i-\hat{\mu}_1)^T + \sum_{i:g_i=2}(x_i-\hat{\mu}_2)(x_i-\hat{\mu}_2)^T\right)$

  $= \frac{1}{N-2}\left(\sum_{i=1}^{N} x_i x_i^T - N_1\hat{\mu}_1\hat{\mu}_1^T - N_2\hat{\mu}_2\hat{\mu}_2^T\right),$

- **And we know that,** $\hat{\beta}_0 = \frac{1}{N}\sum_{i=1}^{N} Y_i - \frac{1}{N}\sum_{i=1}^{N} x_i^T\hat{\beta} = -\frac{1}{N}\sum_{i=1}^{N} x_i^T\hat{\beta}$

$$\left(\sum_{i=1}^{N} x_i x_i^T - \frac{1}{N}\sum_{i=1}^{N} x_i \sum_{i=1}^{N} x_i^T\right)\hat{\beta} = \sum_{i=1}^{N} Y_i x_i - \frac{1}{N}\sum_{i=1}^{N} Y_i \sum_{i=1}^{N} x_i = \sum_{i=1}^{N} Y_i x_i$$

derivation of the LDA direction via LS

- $$(\sum_{i=1}^{N} x_i x_i^T - \frac{1}{N}\sum_{i=1}^{N} x_i \sum_{i=1}^{N} x_i^T)\hat{\beta} = ((N-2)\hat{\Sigma} + N_1\hat{\mu}_1\hat{\mu}_1^T + N_2\hat{\mu}_2\hat{\mu}_2^T - \frac{1}{N}(N_1\hat{\mu}_1 + N_2\hat{\mu}_2)(N_1\hat{\mu}_1^T + N_2\hat{\mu}_2^T))\hat{\beta}$$

$$= ((N-2)\hat{\Sigma} + \frac{N_1 N_2}{N}(\hat{\mu}_2 - \hat{\mu}_1)(\hat{\mu}_2 - \hat{\mu}_1)^T)\hat{\beta} = N(\hat{\mu}_2 - \hat{\mu}_1)$$

- **Since $(\hat{\mu}_2 - \hat{\mu}_1)^T\hat{\beta}$ is a scalar, $\hat{\beta} \propto \hat{\Sigma}^{-1}(\hat{\mu}_2 - \hat{\mu}_1)$ i.e. the LS coefficient is identical to the LDA coefficient, up to a scalar multiple**

- **Define between-class covariance:**

$$\hat{\Sigma}_B = \frac{1}{N}\{N_1(\hat{\mu}_1 - \frac{\hat{\mu}_1 + \hat{\mu}_2}{2})(\hat{\mu}_1 - \frac{\hat{\mu}_1 + \hat{\mu}_2}{2})^T + N_2(\hat{\mu}_2 - \frac{\hat{\mu}_1 + \hat{\mu}_2}{2})(\hat{\mu}_2 - \frac{\hat{\mu}_1 + \hat{\mu}_2}{2})^T\}$$

$$= \frac{1}{4}(\hat{\mu}_2 - \hat{\mu}_1)(\hat{\mu}_2 - \hat{\mu}_1)^T$$

derivation of the LDA direction via LS

- **Classify to class 2 if** $\hat{Y}_i > 0$ **and class 1 otherwise.**
- **Since** $\hat{\beta}_0 = -\frac{1}{N}\sum_{i=1}^{N} x_i^T\hat{\beta} = -\frac{1}{N}(N_1\hat{\mu}_1^T + N_2\hat{\mu}_2^T)\hat{\beta}$ **,**

  classify to class 2 if

$$\hat{f}(x) = \hat{\beta}_0 + x^T\hat{\beta} = [x^T - \frac{1}{N}(N_1\hat{\mu}_1^T + N_2\hat{\mu}_2^T)]\hat{\beta}$$

$$\propto [x^T - \frac{1}{N}(N_1\hat{\mu}_1^T + N_2\hat{\mu}_2^T)]\hat{\Sigma}^{-1}(\hat{\mu}_2 - \hat{\mu}_1) > 0$$

$$x^T\hat{\Sigma}^{-1}(\hat{\mu}_2 - \hat{\mu}_1) > \frac{N_2}{N}\hat{\mu}_2{}^T\hat{\Sigma}^{-1}\hat{\mu}_2 - \frac{N_1}{N}\hat{\mu}_1{}^T\hat{\Sigma}^{-1}\hat{\mu}_1 + \frac{N_2 - N_1}{N}\hat{\mu}_1^T\hat{\Sigma}^{-1}\hat{\mu}_2$$

- **This is not the same as the LDA rule unless the classes have equal numbers of observations**
- **With more than 2 classes, LDA is not the same as Linear regression**

Masking effect

- **Covariance Decomposition: let** $\bar{\mu} = \frac{1}{N}\sum_{i=1}^{N} x_i = \frac{1}{N}\sum_{k=1}^{K} N_k \hat{\mu}_k$.

$$\hat{\Sigma}_T = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{\mu})(x_i - \bar{\mu})^T = \frac{1}{N}\left[\sum_{i=1}^{N} x_i x_i^T - N\bar{\mu}\bar{\mu}^T\right] \text{ (total covariance)}$$

$$\hat{\Sigma}_W = \frac{1}{N}\sum_{k=1}^{K}\sum_{g_i=k}(x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T$$

$$= \frac{1}{N}\sum_{k=1}^{K}\left[\sum_{g_i=k} x_i x_i^T - N_k \hat{\mu}_k \hat{\mu}_k^T\right] = \frac{1}{N}\left[\sum_{i=1}^{N} x_i x_i^T - \sum_{k=1}^{K} N_k \hat{\mu}_k \hat{\mu}_k^T\right] \text{ (within-class covariance)}$$

$$\hat{\Sigma}_B = \frac{1}{N}\sum_{k=1}^{K} N_k(\hat{\mu}_k - \bar{\mu})(\hat{\mu}_k - \bar{\mu})^T = \frac{1}{N}\left[\sum_{k=1}^{K} N_k \hat{\mu}_k \hat{\mu}_k^T - N\bar{\mu}\bar{\mu}^T\right] \text{ (between-class covariance)}$$

$$\therefore \hat{\Sigma}_T = \hat{\Sigma}_W + \hat{\Sigma}_B$$

Masking effect

- **In LDA,** $\delta_k(x) = -\frac{1}{2}(x-\hat{\mu}_k)^T\hat{\Sigma}_W^{-1}(x-\hat{\mu}_k) + \log\frac{N_k}{N}$

- **In linear regression of an indicator matrix,**

$$\hat{G}(x) = \underset{k\in\mathcal{G}}{\operatorname{argmax}}\left(x^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}\right)_k + \frac{N_k}{N} = \underset{k\in\mathcal{G}}{\operatorname{argmax}}\; x^T\hat{\Sigma}_T^{-1}\frac{1}{N}\sum_{g_i=k}x_i + \frac{N_k}{N}$$

$$= \underset{k\in\mathcal{G}}{\operatorname{argmax}}\frac{N_k}{N}x^T\hat{\Sigma}_T^{-1}\hat{\mu}_k + \frac{N_k}{N}$$

This corresponds to minimizing,

$$\delta_k(x) = \frac{N_k}{2N}\{(x-\hat{\mu}_k)^T\hat{\Sigma}_T^{-1}(x-\hat{\mu}_k) - x^T\hat{\Sigma}_T^{-1}x - \hat{\mu}_k^T\Sigma_T^{-1}\hat{\mu}_k - 2\}$$

Masking effect

- When the class priors are equal,

- LDA rules assigns to the class closest in Mahalanobis distance,

$$\delta_k(x) = (x - \hat{\mu}_k)^T \hat{\Sigma}_W^{-1} (x - \hat{\mu}_k)$$

- Linear Regression rule uses the distance,

$$\delta_k(x) = (x - \hat{\mu}_k)^T \hat{\Sigma}_T^{-1} (x - \hat{\mu}_k) - \hat{\mu}_k^T \Sigma_T^{-1} \hat{\mu}_k$$

- They differ in the distance metric, and LR has an additional term. LR uses the total covariance; the configuration of the centroids can distort this metric. Also additional term depends on this.

Masking effect

5.2 Antipenalization

Here we view LDA as a regression method. One form of Fisher's linear discriminant rule is to assign to the class with the largest value of

$$R_{\mathrm{LDA}}(j, \mathbf{x}) = \mathbf{x}^T \mathbf{S}_W^{-1} \mathbf{m}_j - \frac{1}{2} \mathbf{m}_j^T \mathbf{S}_W^{-1} \mathbf{m}_j + \log \pi_j. \quad (24)$$

The corresponding multiple regression rule (*softmax*) is to assign to the class with the largest fitted value

$$\begin{aligned} R_{MR}(j, \mathbf{x}) &= \mathbf{x}^T (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{Y}_j + N_j/N \\ &= N_j/N(\mathbf{x}^T \mathbf{S}_T^{-1} \mathbf{m}_j + 1). \end{aligned} \quad (25)$$

Again, to simplify the comparison, we consider the equal N_j situation and focus on the *class coefficient vectors*. These are up to a constant,

$$b_j^{\mathrm{LDA}} = \mathbf{S}_W^{-1} \mathbf{m}_j \quad (\mathrm{LDA})$$

$$b_j^{\mathrm{Soft}} = \mathbf{S}_T^{-1} m_j \quad (softmax).$$

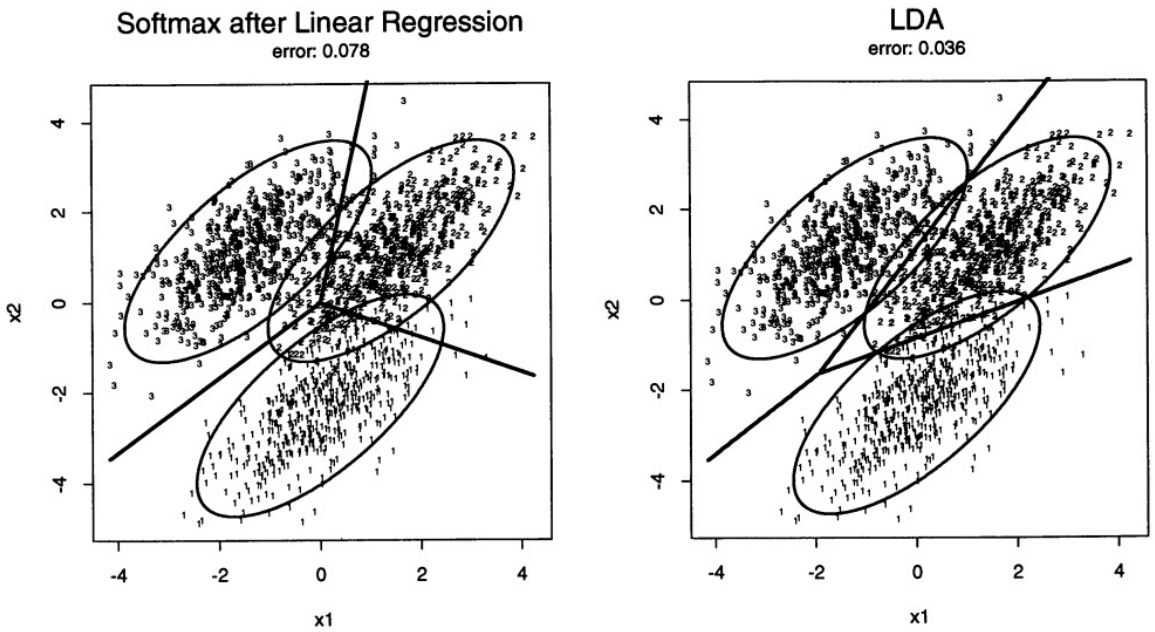
It is not hard to show that these solve the minimization problems:

$$\frac{1}{N} \| \mathbf{y}_j - \mathbf{X}\mathbf{b}_j \|^2 - \mathbf{b}^T \mathbf{S}_{\mathrm{Bet}} \mathbf{b} \quad (\mathrm{LDA}) \quad (26)$$

$$\frac{1}{N} \| \mathbf{y}_j - \mathbf{X}\, b_j \|^2 \quad (softmax). \quad (27)$$

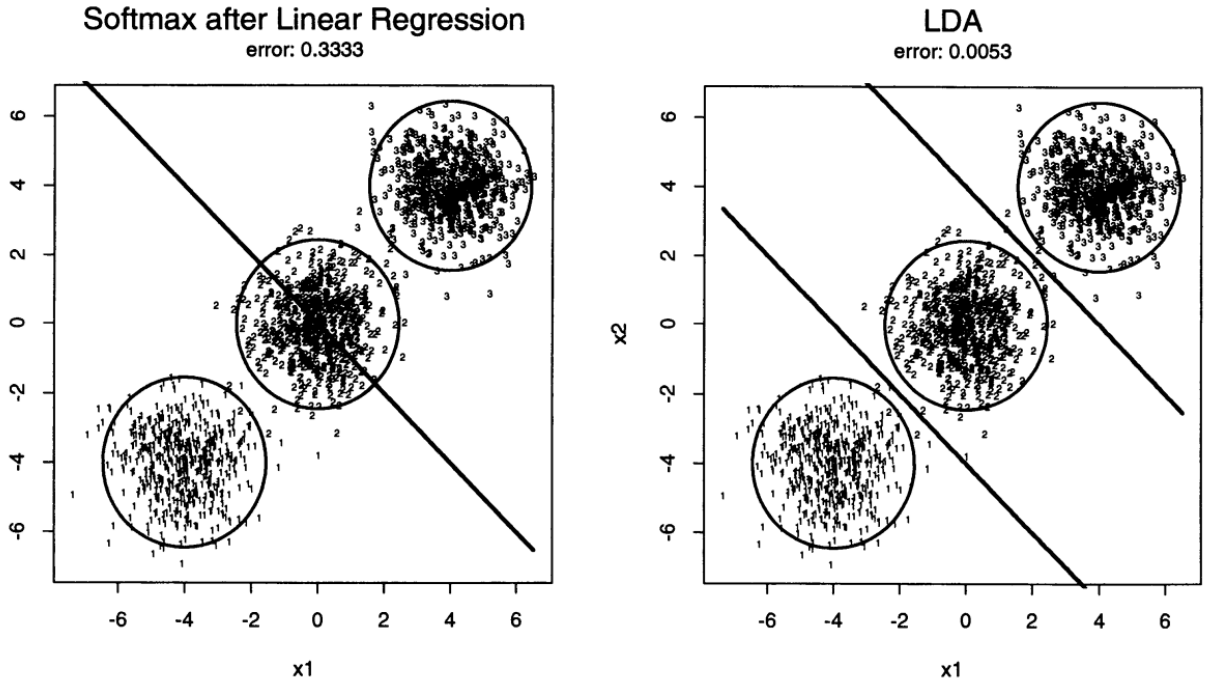
In linear regression, direction b_j is chosen to separate out class k from rest. In contrast, second term in LDA rewards direction b that also separate out all of the K classes

Masking effect



The class centroids are colinear. The center class has centroid zero, while the outer classes do not, and each class benefits from the additional term

The location of their centroid distorts total covariance, and the results in a biased decision boundary for LR but not for LDA

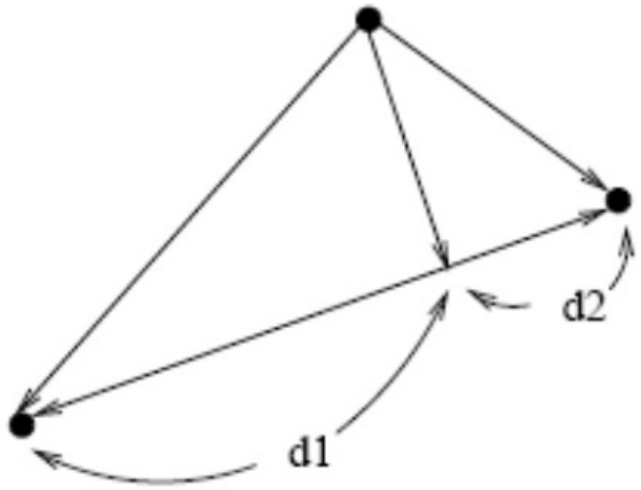


Computations for LDA

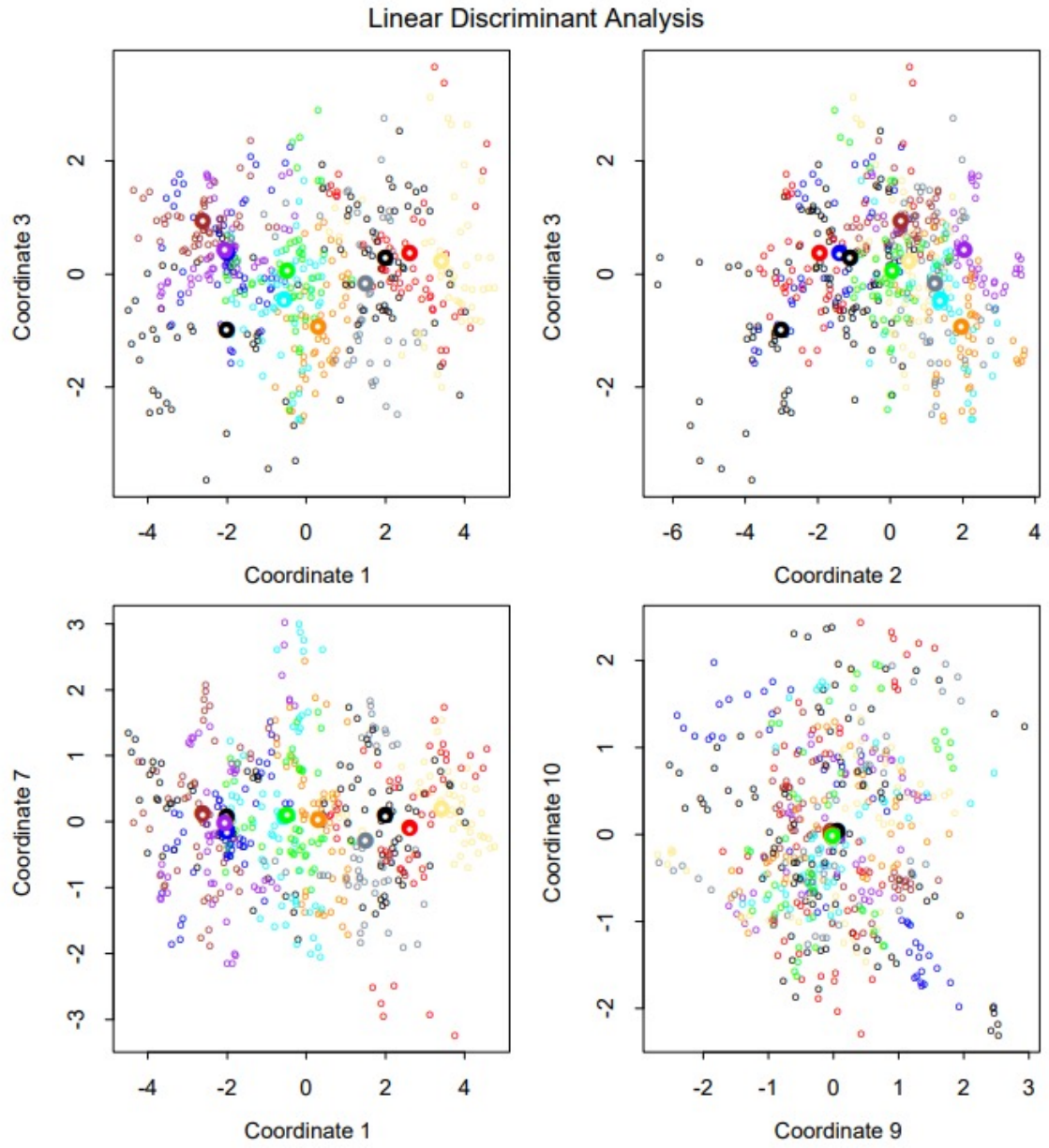
- **Sphere the data with respect to $\hat{\Sigma}$:** $X^* \leftarrow \mathbf{D}^{-\frac{1}{2}}\mathbf{U}^T X,$ where $\hat{\Sigma} = \mathbf{U}\mathbf{D}\mathbf{U}^T$
- **Then** $\delta_k(x^*) = \log[\exp(-\frac{1}{2}(x^* - \mu_k^*)^T(x^* - \mu_k^*))\pi_k] = -\frac{1}{2}\|x^* - \mu_k^*\|^2 + \pi_k$
- **i.e. classify to the closest centroid in the transformed space, modulo the effect of the class prior probabilities π_k**

Reduced-Rank LDA

- **K-centroids in p-dimensional input space lie in an affine subspace with p is much larger than K.**
- **Project X^* onto centroid-spanning subspace H_{K-1}, and make distance comparisons there.**



Reduced-Rank LDA



Reduced-Rank LDA

Finding principal component for H_{K-1}

- **Compute the K by p matrix of class centroids $\mathbf{M}$ and the common covariance matrix $\mathbf{W}$(for within-class covariance).**
- **Compute $\mathbf{M}^* = \mathbf{M}\mathbf{W}^{-\frac{1}{2}}$ using the eigen-decomposition of $\mathbf{W}$.**
- **Compute $\mathbf{B}^*$, the covariance matrix of $\mathbf{M}^*$ ($\mathbf{B}$ for between-class covariance), and its eigen-decomposition $\mathbf{B}^* = \mathbf{V}^*\mathbf{D}_B\mathbf{V}^{*T}$. The columns v_l^* of $\mathbf{V}^*$ in sequence from first to last define the coordinates of the optimal subspaces.**

The l-th discriminant variable is given by $Z_l = v_l^T X$ with $v_l = \mathbf{W}^{-\frac{1}{2}} v_l^*$

Reduced-Rank LDA

- **Fisher's idea:**
  Find the linear combination $Z = a^T X$ s.t. the between-class variance is maximized relative to the within-class variance
- **Note. Between-class variance: the variance of the class means of Z,**
  Within-class variance: the pooled variance about the means.

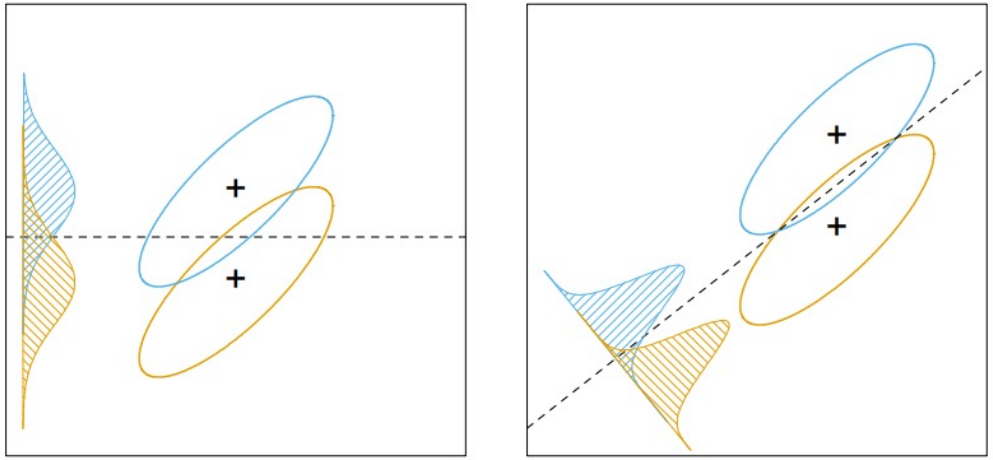


FIGURE 4.9. *Although the line joining the centroids defines the direction of greatest centroid spread, the projected data overlap because of the covariance (left panel). The discriminant direction minimizes this overlap for Gaussian data (right panel).*

Reduced-Rank LDA

- **Fisher's idea:**
  Find the linear combination $Z = a^T X$ s.t. the between-class variance is maximized relative to the within-class variance
- **Note. Between-class variance: the variance of the class means of Z,**
  Within-class variance: the pooled variance about the means.

- **i.e.** $\max_a \frac{a^T \mathbf{B} a}{a^T \mathbf{W} a}$**. For W,** $\mathbf{W} = \mathbf{V}_W \mathbf{D}_W \mathbf{V}_W^T = (\mathbf{D}_W^{\frac{1}{2}} \mathbf{V}_W^T)^T (\mathbf{D}_W^{\frac{1}{2}} \mathbf{V}_W^T) = \mathbf{W}^{\frac{1}{2}^T} \mathbf{W}^{\frac{1}{2}}$**.**

  Let $b = \mathbf{W}^{\frac{1}{2}} a$**, then** $\max_a \frac{a^T \mathbf{B} a}{a^T \mathbf{W} a} = \max_b \frac{b^T \mathbf{W}^{-\frac{1}{2}^T} \mathbf{B} \mathbf{W}^{-\frac{1}{2}} b}{b^T b} = \max_b \frac{b^T \mathbf{B}^* b}{b^T b}$

  $\max_b b^T \mathbf{B}^* b - \lambda(b^T b - 1),\ \mathbf{B}^* b = \lambda b \ \ \therefore b$ is eigen vector for $\mathbf{B}^*, b = v_1^*,\ a = \mathbf{W}^{-\frac{1}{2}} v_1^*$

Quadratic Discriminant Analysis

- Σ_k **are not assumed to be equal.**

$$\delta_k(x) = -\frac{1}{2}\log|\Sigma_k| - \frac{1}{2}(x-\mu_k)^T(x-\mu_k) + \log\pi_k$$

- **For LDA, (K-1) x (p+1) parameters, and QDA will be (k-1) x ((p+1) + (p+1)p/2) parameters.**

- **Bias-Variance tradeoff**

of classification tasks. For example, in the STATLOG project (Michie et al., 1994) LDA was among the top three classifiers for 7 of the 22 datasets, QDA among the top three for four datasets, and one of the pair were in the

Logistic Regression

- **The Logistic regression model arises from the desire to model the posterior probabilities of the K classes via linear functions in x.**

$$\log \frac{Pr(G=k|X=x)}{Pr(G=K|X=x)} = \beta_{k0} + \beta_k^T x \text{ for } k = 1, ..., K-1$$

$$Pr(G=k|X=x) = \frac{\exp(\beta_{k0}+\beta_k^T x)}{1+\sum_{l=1}^{K-1}\exp(\beta_{l0}+\beta_l^T x)} \text{ for } k = 1, ..., K-1$$

$$Pr(G=K|X=x) = \frac{1}{1+\sum_{l=1}^{K-1}\exp(\beta_{l0}+\beta_l^T x)}$$

- **Notation: To emphasize the dependence on the entire parameters,**

$$\theta = \{\beta_{10}, \beta_1^T, ..., \beta_{(K-1)0}, \beta_{K-1}^T, \ Pr(G=k|X=x) = p_k(x;\theta)$$

Logistic Regression

- **Fit by maximum likelihood, using conditional likelihood of $G|X$; the multinomial distribution.**
- **In two-class case, let** $p(x;\theta) = Pr(G = 1|X = x;\theta)$.
  Then $1 - p(x;\theta) = Pr(G = 2|X = x;\theta)$,

$$l(\beta) = \sum_{i=1}^{N} \{y_i \log p(x_i;\beta) + (1 - y_i) \log (1 - p(x_i;\beta))\}$$

$$= \sum_{i=1}^{N} \{y_i \beta^T x_i - \log (1 + \exp(\beta^T x_i))\}$$

Logistic Regression

- **Use Newton-Raphson algorithm,**

$$\frac{\partial l(\beta)}{\partial \beta} = \sum_{i=1}^{N} x_i(y_i - p(x_i;\beta)) = \mathbf{X}^T(\mathbf{y}-\mathbf{p}) \text{ where } \mathbf{p} \text{ the vector of } p(x_i;\beta^{old})$$

$$\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T} = -\sum_{i=1}^{N} x_i x_i^T p(x_i;\beta)(1-p(x_i;\beta)) = -\mathbf{X}^T\mathbf{W}\mathbf{X} \text{ where } \mathbf{W} \text{ a diagonal matrix with}$$

$$\mathbf{W}_{ii} = p(x_i;\beta^{old})(1-p(x_i;\beta^{old})).$$

- **The Newton step is thus**

$$\begin{aligned}\beta^{new} &= \beta^{old} + (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{y}-\mathbf{p}) \\ &= (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}(\mathbf{X}\beta^{old} + \mathbf{W}^{-1}(\mathbf{y}-\mathbf{p})) \\ &= (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}\mathbf{z}, \ \mathbf{z} : \text{adjusted response}\end{aligned}$$

- **Iteratively reweighted least squares, since each iteration solves:**

$$\beta^{new} \leftarrow \underset{\beta}{\operatorname{argmin}}(\mathbf{z}-\mathbf{X}\beta)^T\mathbf{W}(\mathbf{z}-\mathbf{X}\beta)$$

Logistic Regression

- **The algorithm dose converge, since the log-likelihood is concave, but overshooting can occur.**

  → log-likelihood decreases, step size halving

- **For the multiclass case,**

$$l(\beta) = \sum_{i=1}^{N} \sum_{k=1}^{K-1} [I(y_i = k)\beta_k^T x_i - \log(1 + \sum_{l=1}^{K-1} \exp(\beta_l^T x_i))]$$

Logistic Regression

- **Recall that $\mathbf{z} = \mathbf{X}\hat{\beta} + \mathbf{W}^{-1}(\mathbf{y} - \hat{\mathbf{p}})$, and $\hat{\beta}$ are the coefficients of a weight lest squares fit with the weights $\mathbf{W}$.**
- **$\hat{\beta} = (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}\mathbf{z}$ and $z_i = x_i^T\hat{\beta} + \frac{(y_i - \hat{p}_i)}{\hat{p}_i(1-\hat{p}_i)}$, $w_i = \hat{p}_i(1-\hat{p}_i)$**
  - **→ $\hat{\beta}$ satisfy a self-consistency relationship i.e. need iterative method to solve.**
  - **→ require large computational cost; use Rao-score test or Wald test to avoid recomputing the entire weighted least squares fit.**

- **Deviance: the difference of likelihoods between the fitted model and the saturated model; $D = -2l(\beta) + 2l(\text{perfect fitting})$**

Logistic Regression

- **In two-class case, a quadratic approximation to the deviance:**

$$D = -2\sum_{i=1}^{N}\{y_i \log \hat{p}_i + (1-y_i)\log(1-\hat{p}_i)\} + 2\sum_{i=1}^{N}\{y_i \log y_i + (1-y_i)\log(1-y_i)\}$$

$$= 2\sum_{i=1}^{N}[y_i \log \frac{y_i}{\hat{p}_i} + (1-y_i)\log\frac{1-y_i}{1-\hat{p}_i}]$$

$$\approx 2\sum_{i=1}^{N}[(y_i-\hat{p}_i) + \frac{(y_i-\hat{p}_i)^2}{2\hat{p}_i} + \{(1-y_i)-(1-\hat{p}_i)\} + \frac{\{(1-y_i)-(1-\hat{p}_i)\}^2}{2(1-\hat{p}_i)}]$$

$$= \sum_{i=1}^{N}\frac{(y_i-\hat{p}_i)^2}{\hat{p}_i} + \frac{y_i-\hat{p}_i}{1-\hat{p}_i}$$

$$= \sum_{i=1}^{N}\frac{(y_i-\hat{p}_i)^2}{\hat{p}_i(1-\hat{p}_i)}$$

which is Pearson chi-square statistic!

Logistic Regression

- **In two-class case,** $y_i \overset{\text{iid}}{\sim} Bernoulli(p_i)$**;** $\mathbb{E}[\mathbf{y}] = \mathbf{p},\ Var[\mathbf{y}] = \mathbf{W}$**.**

$$\begin{aligned}\mathbb{E}[\hat{\beta}] &= \mathbb{E}[(\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}\mathbf{z}] \\ &= (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}\mathbb{E}[\mathbf{X}\beta + \mathbf{W}^{-1}(\mathbf{y}-\mathbf{p})] \\ &= (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}\mathbf{X}\beta = \beta\end{aligned}$$

$$\begin{aligned}Var[\hat{\beta}] &= Var[(\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}\mathbf{z}] \\ &= (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}Var[\mathbf{X}\beta + \mathbf{W}^{-1}(\mathbf{y}-\mathbf{p})]((\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W})^T \\ &= (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}(\mathbf{W}^{-1}\mathbf{W}\mathbf{W}^{-1^T})((\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W})^T \\ &= (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\end{aligned}$$

- **For large enough N, a CLT then shows that** $\hat{\beta} \sim \mathcal{N}(\beta, (\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1})$

L1 Regularized Logistic Regression

- $\max_{\beta_0,\beta} \left[\sum_{i=1}^{N} \{ y_i(\beta_0 + \beta^T x_i) - \log(1 + \exp(\beta_0 + \beta^T x)) \} - \lambda \sum_{j=1}^{p} |\beta_p| \right]$

- **This criterion is concave but non-linear.**
- **Use Newton algorithm, we can solve this by repeated application of weighted lasso algorithm.**

- **For the variables with non-zero coefficients,**

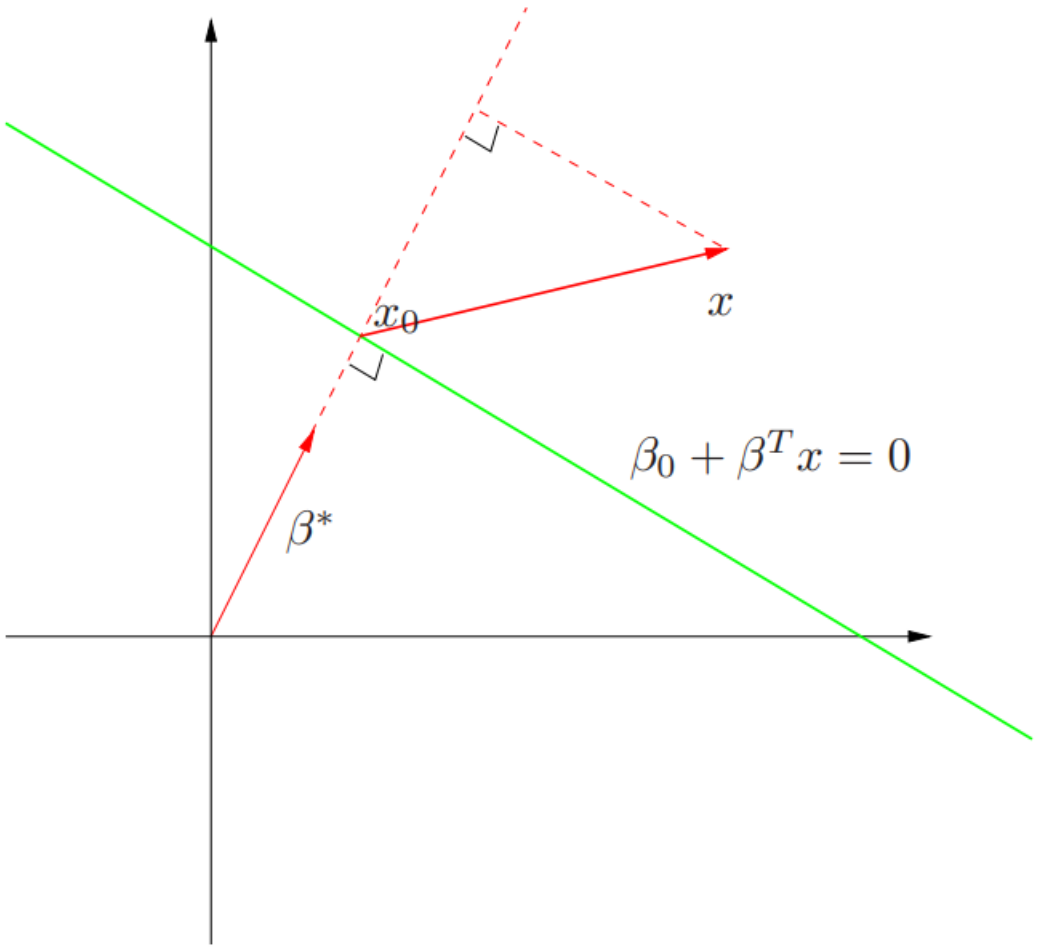
$$\frac{\partial l}{\partial \beta_j} = \mathbf{x}_j^T(\mathbf{y} - \mathbf{p}) - \lambda \cdot \text{sign}(\beta_j) = 0, \;\; \mathbf{x}_j^T(\mathbf{y} - \mathbf{p}) = \lambda \cdot \text{sign}(\beta_j)$$

Logistic Regression or LDA?

- In LDA, we assume $\Pr(X) = \sum_{k=1}^{K} \pi_k \phi(X; \mu_k, \Sigma)$, and Logistic regression leaves the density X.
- By relying on the additional model assumptions, we have more information about the parameters, and hence can estimate them more efficiently (lower variance). Useful for un-classified observations.
- In practice these assumptions are never correct;(qualitive variable case)
- For LDA, observations far from the decision boundary play a role in estimating the common covariance i.e. not robust to outliers.

Perceptron learning

- **Construct decision boundary directly.**



Ex. For x_0 in two-class case:

$$\beta_0 + \beta^T x_0 > 0$$ **, then class 2**

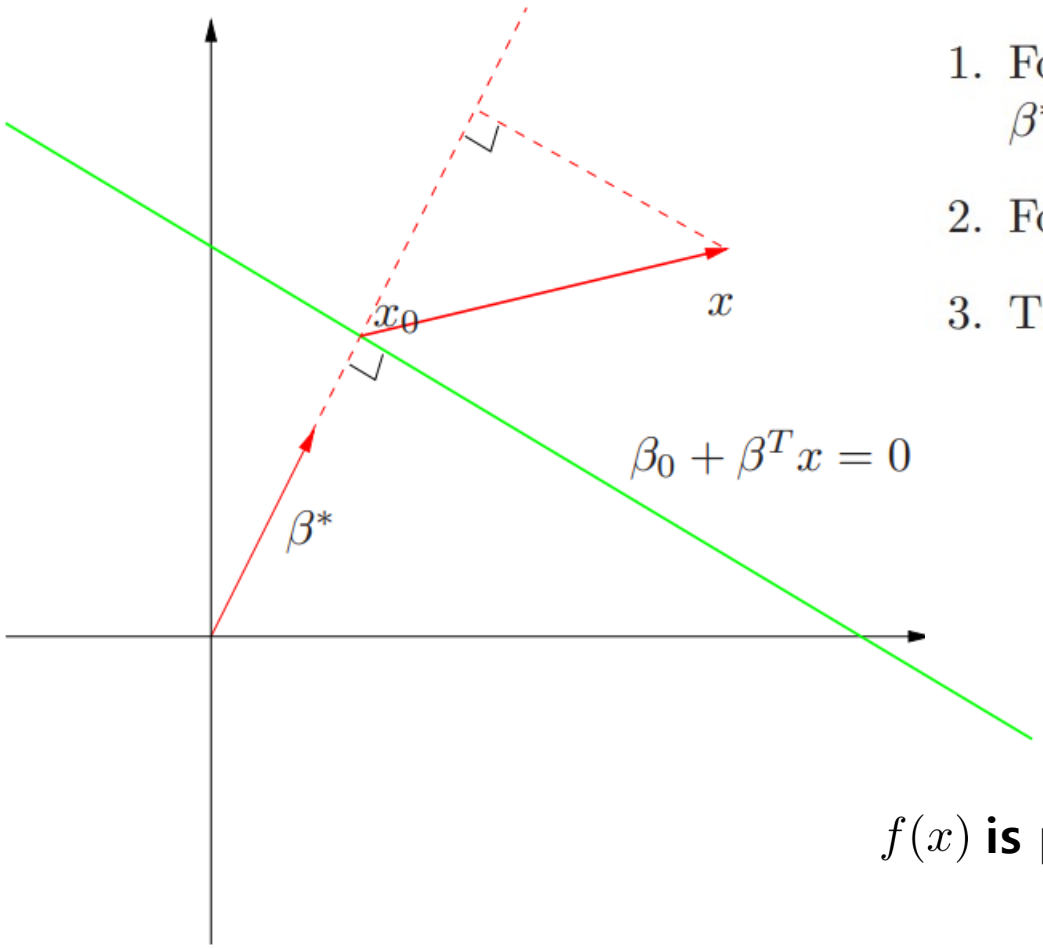
$$\beta_0 + \beta^T x_0 = 0$$ **, on the line**

$$\beta_0 + \beta^T x_0 < 0$$ **, then class 1**

i.e. compute a linear combination of the input features and return the sign, were called perceptron

Perceptron learning

- **Construct decision boundary directly.**



1. For any two points x_1 and x_2 lying in L, $\beta^T(x_1 - x_2) = 0$, and hence $\beta^* = \beta/||\beta||$ is the vector normal to the surface of L.
2. For any point x_0 in L, $\beta^T x_0 = -\beta_0$.
3. The signed distance of any point x to L is given by

$$\begin{aligned} \beta^{*T}(x - x_0) &= \frac{1}{||\beta||}(\beta^T x + \beta_0) \\ &= \frac{1}{||f'(x)||} f(x). \end{aligned} \tag{4.40}$$

$f(x)$ is proportional to the signed distance from x to the hyperplane

Perceptron learning

- **Goal: Find a separating hyperplane by minimizing the distance of misclassified points to the decision boundary.**
- **Let $\mathcal{M}$ be the set of indexes for misclassified points.**

- $\min_{\beta_0,\beta} D(\beta,\beta_0) = \min_{\beta_0,\beta} -\sum_{i\in\mathcal{M}} y_i(x_i^T\beta+\beta_0)$ **; D is non-negative and proportional to the distance of the misclassified points**
- **Optimized by Stochastic Gradient Descent algorithm.**

$$\partial\frac{D(\beta,\beta_0)}{\partial\beta} = -\sum_{i\in\mathcal{M}} y_i x_i,$$
$$\partial\frac{D(\beta,\beta_0)}{\partial\beta_0} = -\sum_{i\in\mathcal{M}} y_i.$$

$$\begin{pmatrix}\beta\\ \beta_0\end{pmatrix} \leftarrow \begin{pmatrix}\beta\\ \beta_0\end{pmatrix} + \rho\begin{pmatrix}y_i x_i\\ y_i\end{pmatrix}$$

Perceptron learning

- **Problems:**

  - When the data are separable, there are many solutions, and which one is found depends on the starting values.
  - The "finite" number of steps can be very large. The smaller the gap, the longer the time to find it.
  - When the data are not separable, the algorithm will not converge, and cycles develop. The cycles can be long and therefore hard to detect.

Optimal Separating Hyperplanes

- **Separates the two classes and maximizes the distance to the closest point from either class**

$$\max_{\beta,\beta_0,||\beta||=1} M$$
$$\text{subject to } y_i(x_i^T\beta+\beta_0)\geq M,\ i=1,\ldots,N.$$

- **Get rid of the** $\|\beta\|=1$ **by** $\frac{1}{\|\beta\|}y_i(x_i^T\beta+\beta)\geq M,\ \text{and set } \|\beta\|=\frac{1}{M}$
- **Thus, criterion is equivalent to**

$$\min_{\beta,\beta_0}\frac{1}{2}||\beta||^2$$
$$\text{subject to } y_i(x_i^T\beta+\beta_0)\geq 1,\ i=1,\ldots,N.$$

Optimal Separating Hyperplanes

- **Then the Lagrange primal function, to be minimized w.r.t.** β_0, β **is**

$$L_P = \frac{1}{2}||\beta||^2 - \sum_{i=1}^{N} \alpha_i [y_i(x_i^T \beta + \beta_0) - 1].$$

- **Setting the derivatives to zero, we obtain:** $\beta = \sum_{i=1}^{N} \alpha_i y_i x_i, \ \sum_{i=1}^{N} \alpha_i y_i = 0$

- **And substituting these in primal function, we obtain the so-called Wolfe dual:**

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k \text{ subject to } \alpha_i \geq 0$$

Lagrange Duality

- **Primal** $minf(x)$

$$\text{subject to } g_i(x) \leq 0,\ i = 1, ...m$$
$$h_j(x) = 0,\ j = 1, ...p$$

Optimizer: x^*

Optimal value: p^*

- **Lagrangian**

$$\mathcal{L}(x, \alpha, \beta) = f(x) + \sum_i \alpha_i g_i(x) + \sum_j \beta_j h_j(x) \text{ where } \alpha_i \geq 0 \text{ for all } i$$

Easily check that $p^* \geq \min_x \mathcal{L}(x, \alpha, \beta) := \mathcal{D}(\alpha, \beta)$

- **Dual**

$$\max \mathcal{D}(\alpha, \beta) \text{ subject to } \alpha \geq 0$$

Optimal value: d^*

Weak duality: $d^* \leq p^*$

Karush-Kuhn-Tucker optimality condition

$$0 \in \partial\big(f(x) + \textstyle\sum_i \alpha_i g_i(x) + \sum_j \beta_j h_j(x)\big)$$ **1. Stationarity**

$$\alpha_i g_i(x) = 0 \text{ for all } i$$ **2. Complementary Slackness**

$$g_i(x) \leq 0, h_j(x) = 0 \text{ for all } i, j$$ **3. Primal Feasibility**

$$\alpha_i \geq 0 \text{ for all } i$$ **4. Dual Feasibility**

$$x^*, \alpha^*, \beta^* \text{ are primal and dual solutions}$$

$$\Leftrightarrow x^*, \alpha^*, \beta^* \text{ satisfy the KKT conditions}$$

Sufficiency

$$\begin{aligned}\mathcal{D}(\alpha^*, \beta^*) &= \min_x \mathcal{L}(x, \alpha^*, \beta^*) \\ &= \mathcal{L}(x^*, \alpha^*, \beta^*) \quad (\mathcal{L} \text{ is convex, and by 1}) \\ &= f(x^*) + \textstyle\sum_i \alpha_i^* g_i(x^*) + \sum_j \beta_j^* h_j(x^*) \\ &= f(x^*) \quad \text{(By 2,3)}\end{aligned}$$

Necessity

$$\begin{aligned}f(x^*) &= \mathcal{L}(\alpha^*, \beta^*) \\ &= \min_x \left(f(x) + \textstyle\sum_i \alpha_i^* g_i(x) + \sum_j \beta_j^* h_j(x) \right) \\ &\leq f(x^*) + \textstyle\sum_i \alpha_i^* g_i(x^*) + \sum_j \beta_j^* h_j(x^*) \text{ equality holds if } \mathcal{L} \text{ is convex, and 1 satisfy} \\ &\leq f(x^*) \quad \text{equality holds if 2,3 satisfy}\end{aligned}$$

Optimal Separating Hyperplanes

- **Stationarity:** $\beta = \sum_{i=1}^{N} \alpha_i y_i x_i,\ \sum_{i=1}^{N} \alpha_i y_i = 0$
- **Primal Feasibility:** $1 - y_i(x_i^T \beta + \beta) \leq 0$ for all i
- **Dual Feasibility:** $L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k$ subject to $\alpha_i \geq 0$

- **Thus, complementary slackness condition should be more satisfied.**

$$\alpha_i [y_i(x_i^T \beta + \beta_0) - 1] = 0 \text{ for all } i.$$

- if $\alpha_i > 0$, then $y_i(x_i^T \beta + \beta_0) = 1$, or in other words, x_i is on the boundary of the slab;
- if $y_i(x_i^T \beta + \beta_0) > 1$, x_i is not on the boundary of the slab, and $\alpha_i = 0$.

Optimal Separating Hyperplanes

- **Since $\beta = \sum_{i=1}^{N} \alpha_i y_i x_i$, the solution vector β is defined by linear-combination of the support vectors x_i , those points defined to be on the boundary of the slab via $\alpha_i > 0$. (robust to outliers)**
- **The optimal separating hyperplane produces $\hat{f}(x) = x^T \hat{\beta} + \hat{\beta}_0$ for classifying new observations:**

$$\hat{G}(x) = \text{sign}\hat{f}(x)$$