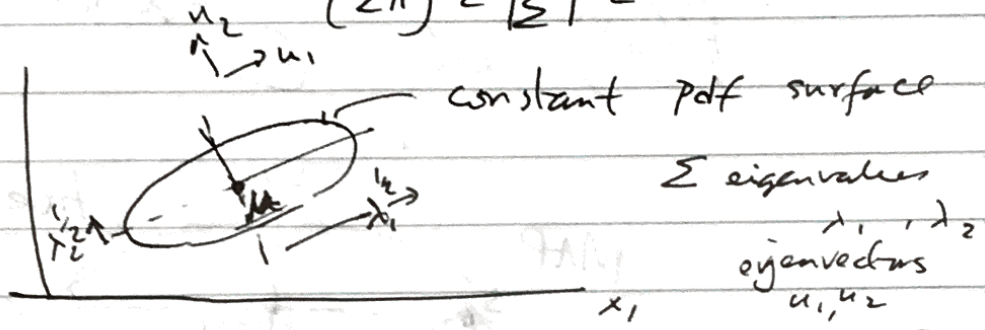
Gaussian Models & Classification

Multivariate Gaussian or multivariate normal (MVN) models.

$$\mathcal{N}(x|\mu,\Sigma) \triangleq \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)}$$



If $\Sigma u_i = \lambda_i u_i$ $\quad \Sigma^{-1} = \sum_i \frac{1}{\lambda_i} u_i u_i^T$

Define $y_i = u_i^T(x-\mu)$

$$(x-\mu)^T\Sigma^{-1}(x-\mu) = \sum_i \frac{y_i^2}{\lambda_i}$$

Mahalanobis distance.

In two-dim $\frac{y_1^2}{\lambda_1} + \frac{y_2^2}{\lambda_2} = \text{constant}$

→ Ellipses.

MLE for MVN

$$\hat{\mu} = \frac{1}{N}\sum_{i=1}^N x_i = \bar{x}$$

$$\hat{\Sigma} = \frac{1}{N}\sum_{i=1}^N (x_i-\bar{x})(x_i-\bar{x})^T$$

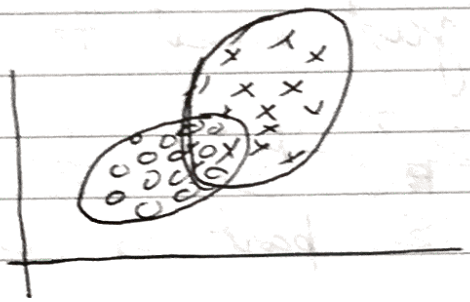
$$= \frac{1}{N}\sum_{i=1}^N x_i x_i^T - \bar{x}\bar{x}^T$$

(lookup derivation in the book!)

Gaussian discriminant analysis

$$P(x|y=c,\theta) = \mathcal{N}(x|\mu_c, \Sigma_c)$$

We will use this model for several supervised (in this lecture) and unsupervised tasks (in later ones).



Imagine we can estimate the parameters (say with labeled data)

Classification:

$$\hat{y}(x) = \underset{c}{\operatorname{argmax}} \left[\log p(y=c|\pi) + \log p(x|\theta_c) \right]$$

(prior prob of class c)

This is just $\hat{y}(x) = \underset{c}{\operatorname{argmax}} \left\{ (x-\mu_c)^T \Sigma_c^{-1} (x-\mu_c) + \log |\Sigma_c| \right\}$

for uniform π

The first term is the Mahalanobis distance from each centroid.

If Σ_c's are different for each class then classes are decided by thresholding some

quadratic function ⇒ Quadratic Discriminant analysis (QDA)

$$P(y=c|\theta) = \frac{\pi_c \frac{1}{(2\pi)^{D/2}|\Sigma_c|^{1/2}} e^{-\frac{1}{2}(x-\mu_c)^T\Sigma_c^{-1}(x-\mu_c)}}{\sum_{c'} \pi_{c'} \frac{1}{(2\pi)^{D/2}|\Sigma_{c'}|^{1/2}} e^{-\frac{1}{2}(x-\mu_{c'})^T\Sigma_{c'}^{-1}(x-\mu_{c'})}}$$

Life simplifies if we assume all $\Sigma_c = \Sigma$, namely, the covariance matrices are tied.

Then we just need to compare

$\log \pi_c - \frac{1}{2}(x-\mu_c)^T\Sigma^{-1}(x-\mu_c)$ for different classes [$|\Sigma|$ part is the same]

$$(x-\mu_c)^T\Sigma^{-1}(x-\mu_c)$$

$$= x^T\Sigma^{-1}x - 2\mu_c^T\Sigma^{-1}x + \mu_c^T\Sigma^{-1}\mu_c + 0$$

↑ independent of class

Call $\gamma_c = \log \pi_c - \frac{1}{2}\mu_c^T\Sigma^{-1}\mu_c$

$$\beta_c = \Sigma^{-1}\mu_c$$

$$P(y=c|x,\theta) = \frac{e^{\beta_c^T x + \gamma_c}}{\sum_{c'} e^{\beta_{c'}^T x + \gamma_{c'}}} = S(\eta)_c$$

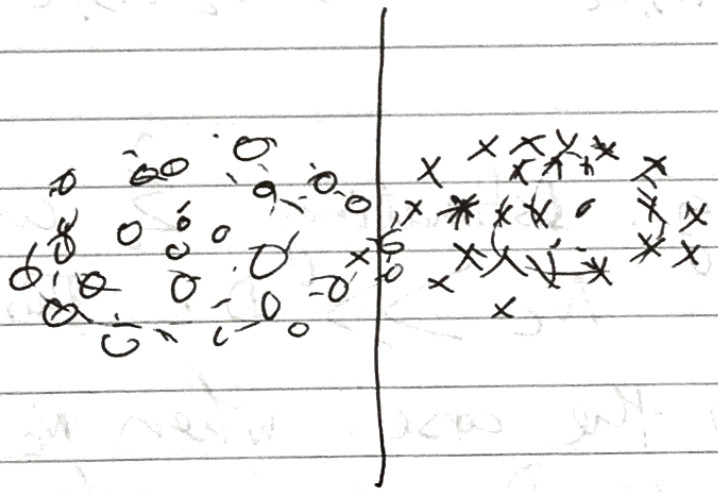
$$\eta = \left[\beta_1^T x + \gamma_1, \cdots ; \beta_C^T x + \gamma_C \right]$$

$$\xi(\eta)_c = \frac{e^{\eta_c}}{\sum_{c'} e^{\eta_{c'}}}$$ is the softmax function

If just have two classes

$c = 1, 2$

$$P(c=1|x,\theta) = \frac{e^{\beta_1^T x + \gamma_1}}{e^{\beta_1^T x + \gamma_1} + e^{\beta_2^T x + \gamma_2}} = \frac{e^{(\beta_1 - \beta_2)^T x + \gamma_1 - \gamma_2}}{1 + e^{(\beta_1 - \beta_2)^T x + \gamma_1 - \gamma_2}}$$



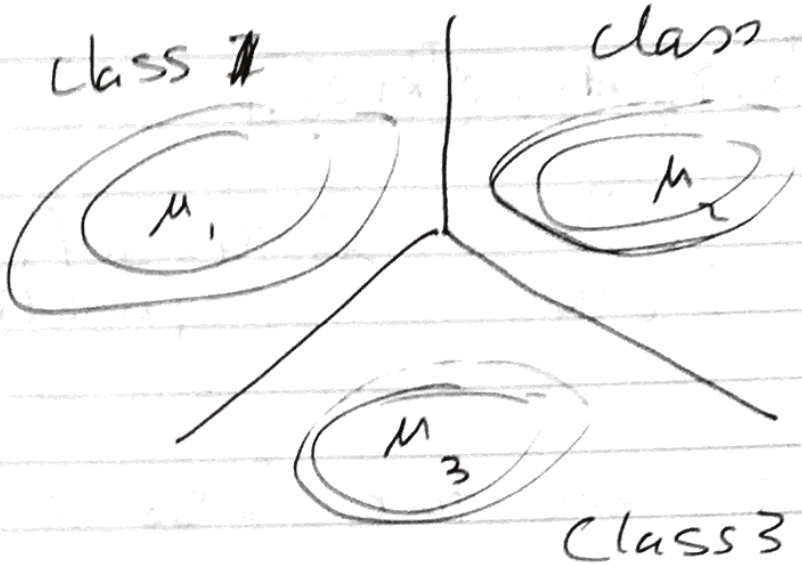
$$(\Delta\beta)^T x + (\Delta\gamma) > 0$$
$$< 0$$

decides class

$(\Delta\beta)^T x + \Delta\gamma = 0$ is the decision boundary.

Linear discriminant analysis (LDA)

Even in the multiclass case, we have



For MLE parameter fit with supervised learning for full-fledged gaussian model.

$$\hat{\mu}_c = \frac{1}{N_c} \sum_{i:y_i=c} x_i \qquad \hat{\Sigma}_c = \frac{1}{N_c} \sum_{i:y_i=c} (x_i - \hat{\mu}_c)(x_i - \hat{\mu}_c)^T$$

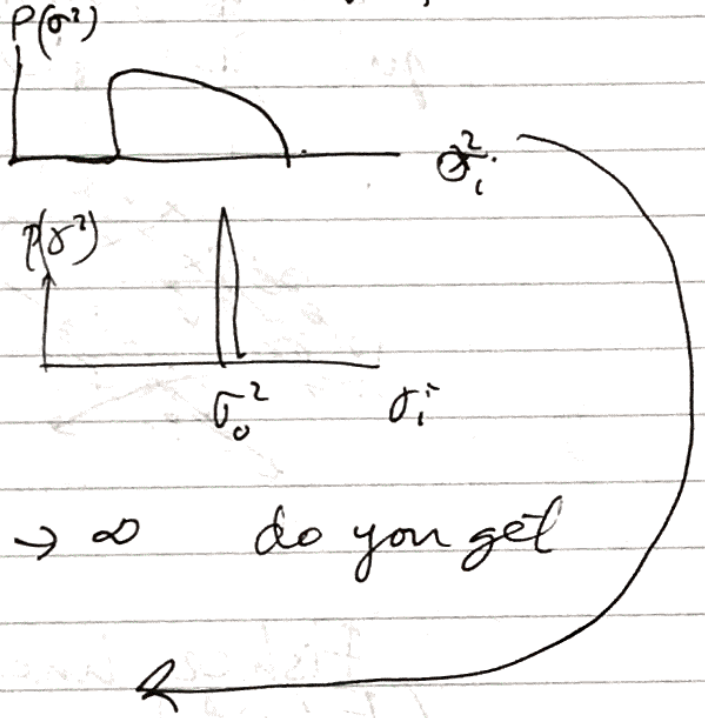
Overfitting: Estimating $\hat{\Sigma}_c$ well requires $N_c \gg D$. This is not always the case. When N_c is comparable to D we do not estimate $\hat{\Sigma}_c$ well.

For example, if I give you $x_i \sim N(0, \sigma_0^2 I_D)$, and you estimate $i = 1, \ldots, N$, x_i independ

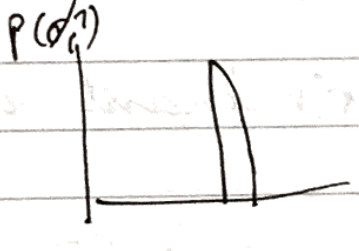
$$\hat{\Sigma} = \frac{1}{N} \sum_i (x_i - \bar{x})(x_i - \bar{x})^T$$

$\hat{\Sigma}$ would have eigenvalues distributed according to the Marchenko Pastur distribution for large N, D with N/D fixed

True $\Sigma = \sigma_0^2 I_D$



Only when $N/D \to \infty$ do you get

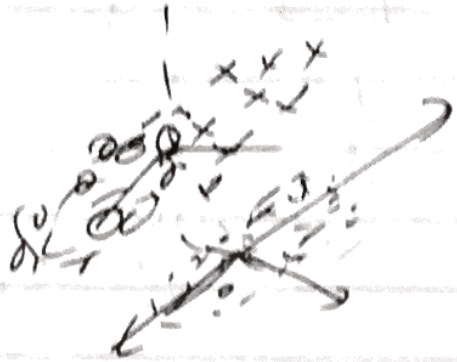


One more we could be fitting noise if $N \sim D$.

Various solutions: a) Parameter tying: $\Sigma_c = \Sigma$
(accept bias reduce variance)
b) Insist on diagonal Σ (like naive Bayes)
c) Integrate over Σ.
d) Project data to lower dim spaces.

The last idea: projecting to the maximally discriminative subspace, is an important topic in itself.

One approach could be to just find PCA components before classifying and keep a small number of principal components

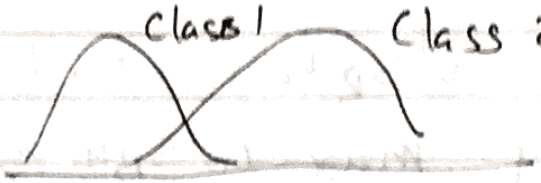


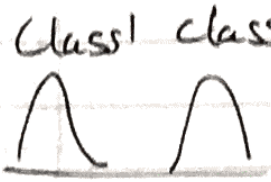
However, that approach does not pay attention to class labels.

Fisher linear discriminant analysis (FLDA)

For two classes, it is just a matter of finding a $w (= \beta_1 - \beta_2)$ weight vector that does the best discrimination.

For X_i from a particular class we expect $z_i = w^T x_i$ would be normally distributed





we should choose w s.t.

$$m_c = \frac{1}{N_c} \sum_{i: y_i = c} x_i \qquad s_c^2 = \sum_{i: y_i = c} (x_i - m_c)^2$$

↑ Sum sq. error

Construct a measure

$$J(w) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} = \left(\frac{\text{centroid dist}}{\text{intraclass variability}}\right)^2$$

This is proportional to the pooled t-statistic

$$J(w) = \frac{w^T S_B w}{w^T S_W w}$$

$$S_B = (\mu_2 - \mu_1)(\mu_2 - \mu_1)^T$$

$$S_W = \sum_{i: y_i = 1} (x_i - \mu_1)(x_i - \mu_1)^T + \sum_{i: y_i = 2} (x_i - \mu_2)(x_i - \mu_2)^T$$

$\arg\max_{\|w\|=1} J(w)$ We need the direction, so, we could choose different normalizations of w.

Max $w^T S_B w$ s.t. $w^T S_W w = 1$

$\max w^T S_B w - \lambda(w^T S_W w - 1)$

$\max_w \min_\lambda w^T S_B w - \lambda(w^T S_W w - 1)$

With the lagrange multiplier adjusted to satisfy the constraint

$$S_B w = \lambda S_W w$$

Generalized eigenvalue problem

If S_W is invertible.

$S_W^{-1} S_B w = \lambda w$ is a conventional eigenvalue problem

Find the largest eigenvalue.

Because S_B is only rank one for two classes, making further simplification possible

$$S_B w = (\mu_2 - \mu_1)(\mu_2 - \mu_1)^T w = (\mu_2 - \mu_1)(m_2 - m_1)$$

$\lambda w = S_W^{-1}(\mu_2 - \mu_1)(m_2 - m_1)$ ← scalar

$w \propto S_W^{-1}(\mu_2 - \mu_1)$

This method could be generalized to more classes. Since S_B is a rank $C-1$ matrix. We could only find $L \leq C-1$ dimensions. This could be a limitation in some applications.

For completeness: $z_i = W x_i$

[Rows of W are like w^T]

$z_i \in \mathbb{R}^L, x_i \in \mathbb{R}^D$

$$m_c = \frac{1}{N_c} \sum_{i:y_i=c} z_i \qquad m_c \in \mathbb{R}^L$$

$$m = \frac{1}{N} \sum_c N_c m_c \qquad \text{over all mean}$$

$$J(W) = \frac{\left| \sum_c N_c (m_c - m)(m_c - m)^T \right|}{\left| \sum_{c=1}^{C} \sum_{i:y_i=c} (z_i - m_c)(z_i - m_c)^T \right|}$$