Linear Discriminant Analysis

Xiaochun MAI

Shenzhen University

Linear Discriminant Analysis

- Linear discriminant analysis (LDA) is an approach used in supervised machine learning to solve multi-class classification problems.
- LDA separates multiple classes with multiple features through data dimensionality reduction.
- This technique is important in data science as it helps optimize machine learning models.

Linear Discriminant Analysis

- Model

$$\underbrace{p(y=c \mid \boldsymbol{x}, \boldsymbol{\theta})}_{\text{Class posterior}} = \frac{\overbrace{p(\boldsymbol{x} \mid y=c, \boldsymbol{\theta})}^{\text{Class conditional density for class c}}\overbrace{p(y=c \mid \boldsymbol{\theta})}^{\text{Prior over class labels}}}{\sum_{c'} p(\boldsymbol{x} \mid y=c', \boldsymbol{\theta}) p(y=c' \mid \boldsymbol{\theta})}$$

- LDA is a generative classifier, since it specifies a way to generate the features **x** for each class c, by sampling from p(**x**|y = c, **θ**).
- By contrast, a discriminative classifier directly models the class posterior p(y|**x**, **θ**).

Reference: Kevin P. Murphy. Probabilistic Machine Learning. Chapter 9. Linear Discriminant Analysis

Linear Discriminant Analysis

- Model

$$\underbrace{p(y=c \mid \boldsymbol{x}, \boldsymbol{\theta})}_{\text{Class posterior}} = \frac{\underbrace{p(\boldsymbol{x} \mid y=c, \boldsymbol{\theta})}_{\text{Class conditional density for class c}} \underbrace{p(y=c \mid \boldsymbol{\theta})}_{\text{Prior over class labels}}}{\sum_{c'} p(\boldsymbol{x} \mid y=c', \boldsymbol{\theta}) p(y=c' \mid \boldsymbol{\theta})}$$

- If we choose the class conditional densities in a special way, the resulting posterior over classes is a linear function of **x**, i.e.,

$$\log p(y = c|\mathbf{x}, \boldsymbol{\theta}) = \mathbf{w}^T\mathbf{x} + \text{const},$$

where **w** is derived from **θ**. Thus the overall method is called **linear discriminant analysis** or **LDA**.

Gaussian discriminant analysis

- Consider a generative classifier where the class conditional densities are multivariate Gaussians:

$$p(\boldsymbol{x} \mid y = c, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$$

- The corresponding class posterior therefore has the form

$$p(y = c \mid \boldsymbol{x}, \boldsymbol{\theta}) \propto p(\boldsymbol{x} \mid y = c, \boldsymbol{\theta}) \underline{p(y = c \mid \boldsymbol{\theta})} \propto \underline{\pi_c} \mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$$

Prior over label c

$$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$$

Quadratic decision boundaries

- The log posterior over class labels is given by

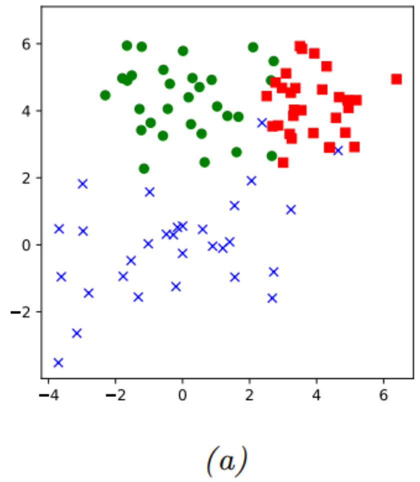
$$\log p(y = c|\boldsymbol{x}, \boldsymbol{\theta}) = \log \pi_c - \frac{1}{2}\log|2\pi\boldsymbol{\Sigma}_c| - \frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_c)^{\mathsf{T}}\boldsymbol{\Sigma}_c^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_c) + \text{const}$$

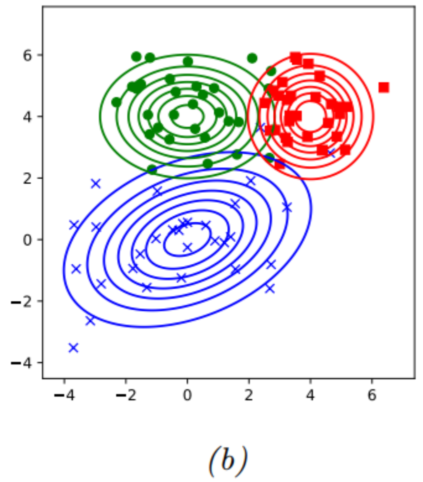
- This is called the discriminant function.
- For any two classes, say c and c', the decision boundary can be determined by

$$p(y = c|\boldsymbol{x}, \boldsymbol{\theta}) = p(y = c'|\boldsymbol{x}, \boldsymbol{\theta})$$

- The decision boundary between any two classes, will be a quadratic function of **x**. Hence this is known as quadratic discriminant analysis (QDA).

Quadratic decision boundaries

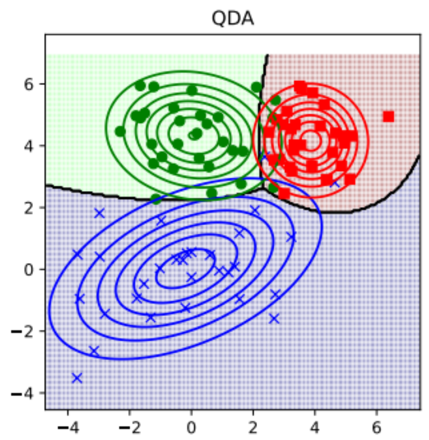




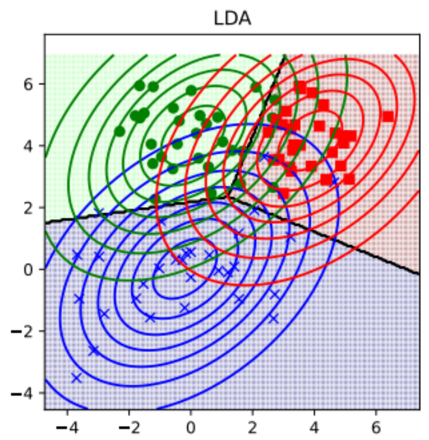
(a) Some 2d data from 3 different classes. (b) Fitting 2d Gaussians to each class.

- the features for the blue class are somewhat correlated
- the features for the green class are independent
- the features for the red class are independent and isotropic (spherical covariance).

Quadratic decision boundaries



(a)



(b)

Gaussian discriminant analysis fit to data. (a) Unconstrained covariances induce quadratic decision boundaries. (b) Tied covariances induce linear decision boundaries.

Linear decision boundaries

- Now we consider a special case of Gaussian discriminant analysis in which the covariance matrices are tied or shared across classes, so $\Sigma_c = \Sigma$. If Σ is independent of c, we can simplify the log posterior as

$$
\begin{aligned}
\log p(y=c|\boldsymbol{x},\boldsymbol{\theta}) &= \log \pi_c - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_c)^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_c) + \text{const} \\
&= \underbrace{\log \pi_c - \frac{1}{2}\boldsymbol{\mu}_c^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c}_{\gamma_c} + \boldsymbol{x}^\mathsf{T}\underbrace{\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c}_{\boldsymbol{\beta}_c} \underbrace{+\text{const} - \frac{1}{2}\boldsymbol{x}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{x}}_{\kappa} \\
&= \gamma_c + \boldsymbol{x}^\mathsf{T}\boldsymbol{\beta}_c + \kappa
\end{aligned}
$$

- The final term is independent of c, and hence is an irrelevant additive constant that can be dropped.
- Hence the discriminant function is a linear function of **x**, so the decision boundaries will be linear. Hence this method is called linear discriminant analysis or LDA.

The connection between LDA and logistic regression

$$\log p(y = c|\boldsymbol{x}, \boldsymbol{\theta}) \ = \gamma_c + \boldsymbol{x}^{\mathsf{T}}\boldsymbol{\beta}_c + \kappa$$

- We can write

$$p(y = c|\boldsymbol{x}, \boldsymbol{\theta}) = \frac{e^{\boldsymbol{\beta}_c^{\mathsf{T}}\boldsymbol{x}+\gamma_c}}{\sum_{c'} e^{\boldsymbol{\beta}_{c'}^{\mathsf{T}}\boldsymbol{x}+\gamma_{c'}}} = \frac{e^{\boldsymbol{w}_c^{\mathsf{T}}[1,\boldsymbol{x}]}}{\sum_{c'} e^{\boldsymbol{w}_{c'}^{\mathsf{T}}[1,\boldsymbol{x}]}} \qquad (9.8)$$

where $\boldsymbol{w}_c = [\gamma_c, \beta_c]$.

- The above equation has the same form as the multinomial logistic regression model.
- The key difference is that in LDA, we first fit the Gaussians (and class prior) to maximize the joint likelihood p(**x**, y|**θ**), and then we derive **w** from **θ**.
- By contrast, in logistic regression, we estimate **w** directly to maximize the conditional likelihood p(y|**x, w**).

The connection between LDA and logistic regression

- To gain further insight into Equation (9.8), let us consider the binary case. In this case, the posterior is given by

$$p(y=1|\boldsymbol{x},\boldsymbol{\theta}) = \frac{e^{\boldsymbol{\beta}_1^{\mathsf{T}}\boldsymbol{x}+\gamma_1}}{e^{\boldsymbol{\beta}_1^{\mathsf{T}}\boldsymbol{x}+\gamma_1} + e^{\boldsymbol{\beta}_0^{\mathsf{T}}\boldsymbol{x}+\gamma_0}} = \frac{1}{1+e^{(\boldsymbol{\beta}_0-\boldsymbol{\beta}_1)^{\mathsf{T}}\boldsymbol{x}+(\gamma_0-\gamma_1)}}$$
$$= \sigma\left((\boldsymbol{\beta}_1-\boldsymbol{\beta}_0)^{\mathsf{T}}\boldsymbol{x}+(\gamma_1-\gamma_0)\right)$$

where σ(η) refers to the sigmoid function.

The connection between LDA and logistic regression

$$\gamma_1 - \gamma_0 = -\frac{1}{2}\boldsymbol{\mu}_1^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_0^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_0 + \log(\pi_1/\pi_0)$$
$$= -\frac{1}{2}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0) + \log(\pi_1/\pi_0)$$

So if we define

$$\boldsymbol{w} = \boldsymbol{\beta}_1 - \boldsymbol{\beta}_0 = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)$$
$$\boldsymbol{x}_0 = \frac{1}{2}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0) - (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)\frac{\log(\pi_1/\pi_0)}{(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)}$$

then we have $\boldsymbol{w}^T\boldsymbol{x}_0 = -(\gamma_1 - \gamma_0)$, and hence

$$p(y = 1|\boldsymbol{x}, \boldsymbol{\theta}) = \sigma(\boldsymbol{w}^\mathsf{T}(\boldsymbol{x} - \boldsymbol{x}_0))$$

This has the same form as binary logistic regression.

The connection between LDA and logistic regression

- Hence the MAP decision rule is

$$\hat{y}(\boldsymbol{x}) = 1 \text{ iff } \boldsymbol{w}^{\mathsf{T}}\boldsymbol{x} > c$$

where $c = \boldsymbol{w}^T \boldsymbol{x}_0$. If $\pi_0 = \pi_1 = 0.5$, then the threshold simplifies to $c = \frac{1}{2}\boldsymbol{w}^T(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_0)$.

Model fitting

- Fit a GDA model using maximum likelihood estimation. The likelihood function is

$$p(\mathcal{D} \mid \boldsymbol{\theta}) = \prod_{n=1}^{N} \underset{\uparrow\ \text{Categorical distribution}}{\mathrm{Cat}}(y_n \mid \underset{\uparrow\ \text{Prior over class labels}}{\boldsymbol{\pi}}) \prod_{c=1}^{C} \mathcal{N}(\boldsymbol{x}_n \mid \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)^{\mathbb{I}(y_n=c)}$$

- The log-likelihood is

$$\log p(\mathcal{D} \mid \boldsymbol{\theta}) = \left[\sum_{n=1}^{N} \sum_{c=1}^{C} \underset{\uparrow\ \text{Choose the prior of the correct class}}{\mathbb{I}(y_n = c)} \log \pi_c\right] + \sum_{c=1}^{C} \left[\sum_{n:y_n=c} \log \mathcal{N}(\boldsymbol{x}_n \mid \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)\right]$$

Log sum of the prior prob. of classes + log-likelihood sum of intra-class points

Model fitting

- The log-likelihood is

$$\log p(\mathcal{D} \mid \boldsymbol{\theta}) = \left[\sum_{n=1}^{N} \sum_{c=1}^{C} \underset{\uparrow}{\mathbb{I}}(y_n = c) \log \pi_c\right] + \sum_{c=1}^{C}\left[\sum_{n:y_n=c} \log \mathcal{N}(\boldsymbol{x}_n \mid \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)\right]$$

Choose the prior of the correct class

Log sum of the prior prob. of classes + log-likelihood sum of intra-class points

- Through maximizing $\log p(\mathcal{D} \mid \boldsymbol{\theta})$, we can optimize π and the (μ_c, Σ_c) terms separately. The MLEs for the Gaussians are

$$\hat{\pi}_c = \frac{N_c}{N}$$

$$\hat{\boldsymbol{\mu}}_c = \frac{1}{N_c} \sum_{n:y_n=c} \boldsymbol{x}_n$$

$$\hat{\boldsymbol{\Sigma}}_c = \frac{1}{N_c} \sum_{n:y_n=c} (\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)^{\mathsf{T}}$$

Model fitting

- Unfortunately the MLE for $\hat{\Sigma}_C$ can easily overfit (i.e., the estimate may not be well-conditioned) if N_c is small compared to D, the dimensionality of the input features.
  - **Tied covariances.** Force $\Sigma_c = \Sigma$, i.e., sharing covariances across classes. This will results in a more reliable parameter estimate, since we can pool all the samples across classes.

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{N} \sum_{c=1}^{C} \sum_{n:y_n=c} (\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_c)^{\mathsf{T}}$$

  - **Diagonal covariances.** Force Σ_c to be diagonal, we reduce the number of parameters from $O(CD^2)$ to $O(CD)$, which avoids the overfitting problem. However, this loses the ability to capture correlations between the features. Despite this approximation, this approach scales well to high dimensions.

Model fitting

- Unfortunately the MLE for $\hat{\Sigma}_C$ can easily overfit (i.e., the estimate may not be well-conditioned) if N_c is small compared to D, the dimensionality of the input features.
  - **MAP estimation.** Forcing the covariance matrix to be diagonal is a rather strong assumption. An alternative approach is to perform MAP estimation of a (shared) full covariance Gaussian, rather than using the MLE. The MAP estimate is

$$\hat{\mathbf{\Sigma}}_{\text{map}} = \lambda \text{diag}(\hat{\mathbf{\Sigma}}_{\text{mle}}) + (1 - \lambda)\hat{\mathbf{\Sigma}}_{\text{mle}}$$

where λ controls the amount of regularization. This technique is known as **regularized discriminant analysis** or **RDA**.

Fisher's linear discriminant analysis

- An alternative approach is to reduce the dimensionality of the features $\boldsymbol{x} \in \mathbb{R}^D$ and then fit an multivariate Gaussian to the resulting low-dimensional features $\boldsymbol{z} \in \mathbb{R}^K$.
- Use a linear projection matrix, $\boldsymbol{z} = \boldsymbol{W}\boldsymbol{x}$, where $\mathbf{W} \in \mathbb{R}^{K \times D}$.
  - To find **W**, we can use principal components analysis (PCA). However, PCA is an unsupervised technique that does not take class labels into account. Thus the resulting low dimensional features are not necessarily optimal for classification.

Fisher's linear discriminant analysis

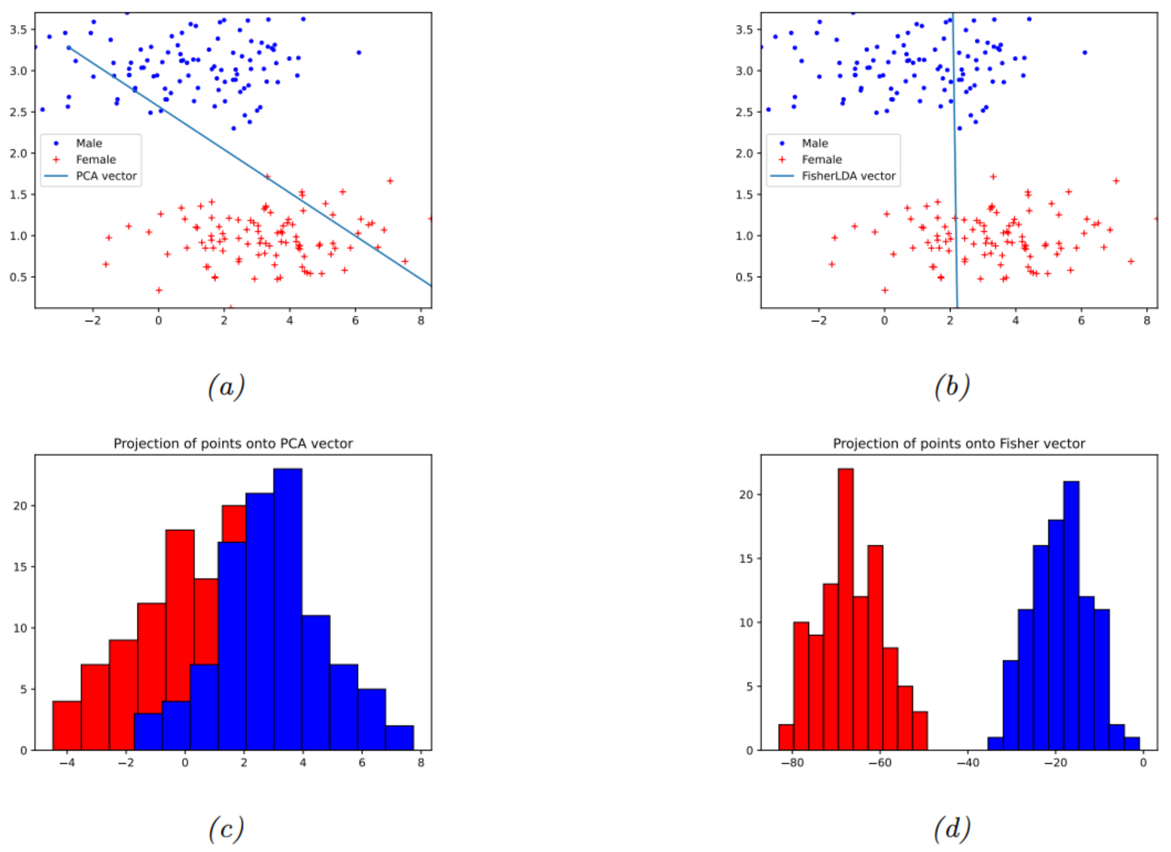


Figure 9.4: Linear discriminant analysis applied to two class dataset in 2d, representing (standardized) height and weight for male and female adults (a) PCA direction. (b) FLDA direction. (c) Projection onto PCA direction shows poor class separation. (d) Projection onto FLDA direction shows good class separation. Generated by fisher_lda_demo.ipynb.

Fisher's linear discriminant analysis

- An alternative approach is to reduce the dimensionality of the features $\boldsymbol{x} \in \mathbb{R}^D$ and then fit an multivariate Gaussian to the resulting low-dimensional features $\boldsymbol{z} \in \mathbb{R}^K$.
- Use a linear projection matrix, $\boldsymbol{z} = \boldsymbol{W}\boldsymbol{x}$, where $\mathbf{W} \in \mathbb{R}^{K \times D}$.
  - An alternative approach is to use gradient based methods to optimize the log likelihood, derived from the class posterior in the low dimensional space.
  - A third approach (which relies on an eigendecomposition, rather than a gradient-based optimizer) is to find the matrix **W** such that the low-dimensional data can be classified as well as possible using a Gaussian class-conditional density model. The assumption of Gaussianity is reasonable since we are computing linear combinations of (potentially non-Gaussian) features. This approach is called **Fisher's linear discriminant analysis**, or **FLDA**.

Fisher’s linear discriminant analysis

- FLDA is an interesting hybrid of discriminative and generative techniques. The drawback of this technique is that it is restricted to using $K \leq C - 1$ dimensions, regardless of D.
- In the two-class case, this means we are seeking a single vector **w** onto which we can project the data.
- Below we derive the optimal **w** in the two-class case. We then generalize to the multi-class case, and finally we give a probabilistic interpretation of this technique.

Derivation of the optimal 1d projection

- We now derive this optimal direction **w**, for the two-class case. Define the class-conditional means as

$$\boldsymbol{\mu}_1 = \frac{1}{N_1} \sum_{n:y_n=1} \boldsymbol{x}_n, \ \boldsymbol{\mu}_2 = \frac{1}{N_2} \sum_{n:y_n=2} \boldsymbol{x}_n$$

- Let $m_k = \boldsymbol{w}^T \boldsymbol{\mu}_k$ be the projection of each mean onto the line **w**. Also, let $z_n = \boldsymbol{w}^T \boldsymbol{x}_n$ be the projection of the data onto the line. The variance of the projected points is proportional to

$$s_k^2 = \sum_{n:y_n=k} (z_n - m_k)^2$$

Derivation of the optimal 1d projection

- The goal is to find **w** such that we maximize the distance between the means, $m_2 - m_1$, while also ensuring the projected clusters are "tight", which we can do by minimizing their variance. This suggests the following objective:

$$\boxed{J(w) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}}$$

Derivation of the optimal 1d projection

- rewrite the right hand side of the above in terms of **w**:

$$J(\boldsymbol{w}) = \frac{\boldsymbol{w}^{\mathsf{T}} \mathbf{S}_B \boldsymbol{w}}{\boldsymbol{w}^{\mathsf{T}} \mathbf{S}_W \boldsymbol{w}}$$

J(**w**) denotes the ratio of between-class divergence and within-class divergence.

- where **S**$_B$ is the between-class scatter matrix, **S**$_W$ is the within-class scatter matrix, given by

$$\mathbf{S}_B = (\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)^{\mathsf{T}}$$

$$\mathbf{S}_W = \sum_{n:y_n=1} (\boldsymbol{x}_n - \boldsymbol{\mu}_1)(\boldsymbol{x}_n - \boldsymbol{\mu}_1)^{\mathsf{T}} + \sum_{n:y_n=2} (\boldsymbol{x}_n - \boldsymbol{\mu}_2)(\boldsymbol{x}_n - \boldsymbol{\mu}_2)^{\mathsf{T}}$$

Derivation of the optimal 1d projection

To see this, note that

$$\boldsymbol{w}^{\mathsf{T}}\mathbf{S}_B\boldsymbol{w} = \boldsymbol{w}^{\mathsf{T}}(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)^{\mathsf{T}}\boldsymbol{w} = (m_2 - m_1)(m_2 - m_1)$$

and

$$\begin{aligned}\boldsymbol{w}^{\mathsf{T}}\mathbf{S}_W\boldsymbol{w} = & \sum_{n:y_n=1} \boldsymbol{w}^{\mathsf{T}}(\boldsymbol{x}_n - \boldsymbol{\mu}_1)(\boldsymbol{x}_n - \boldsymbol{\mu}_1)^{\mathsf{T}}\boldsymbol{w} + \\ & \sum_{n:y_n=2} \boldsymbol{w}^{\mathsf{T}}(\boldsymbol{x}_n - \boldsymbol{\mu}_2)(\boldsymbol{x}_n - \boldsymbol{\mu}_2)^{\mathsf{T}}\boldsymbol{w} \\ = & \sum_{n:y_n=1} (z_n - m_1)^2 + \sum_{n:y_n=2} (z_n - m_2)^2\end{aligned}$$

Derivation of the optimal 1d projection

- Take the derivative of J(**w**) with respect to w and equate to zero.
- Let $\boldsymbol{u} = \boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w}$, $\boldsymbol{v} = \boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w}$, then $J(\boldsymbol{w}) = \frac{\boldsymbol{u}}{\boldsymbol{v}}$

$$\frac{dJ(\boldsymbol{w})}{d\boldsymbol{w}} = \frac{\boldsymbol{v}\frac{d\boldsymbol{u}}{d\boldsymbol{w}} - \boldsymbol{u}\frac{d\boldsymbol{v}}{d\boldsymbol{w}}}{\boldsymbol{v}^2}$$

$$= \frac{(\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w})(2\boldsymbol{S}_B \boldsymbol{w}) - (\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w})(2\boldsymbol{S}_W \boldsymbol{w})}{(\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w})^2}$$

$$(\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w})(2\boldsymbol{S}_B \boldsymbol{w}) - (\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w})(2\boldsymbol{S}_W \boldsymbol{w}) = 0$$

$$\boxed{\boldsymbol{S}_B \boldsymbol{w} = \boldsymbol{\lambda} \boldsymbol{S}_W \boldsymbol{w}}$$

Generalized eigenvalue problem

$$\boldsymbol{\lambda} = \frac{\boldsymbol{w}^T \boldsymbol{S}_B \boldsymbol{w}}{\boldsymbol{w}^T \boldsymbol{S}_W \boldsymbol{w}}$$

Derivation of the optimal 1d projection

- If $\mathbf{S}_W$ is invertible, we can convert it to a regular eigenvalue problem:

$$\mathbf{S}_W^{-1}\mathbf{S}_B\boldsymbol{w} = \lambda\boldsymbol{w}$$

- However, in the two class case, there is a simpler solution. In particular, since

$$\mathbf{S}_B\boldsymbol{w} = (\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)^\mathsf{T}\boldsymbol{w} = (\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)(m_2 - m_1)$$

- Then, we have

$$\lambda\,\boldsymbol{w} = \mathbf{S}_W^{-1}(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)(m_2 - m_1)$$
$$\boldsymbol{w} \propto \mathbf{S}_W^{-1}(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)$$

Derivation of the optimal 1d projection

- Since we only care about the directionality, and not the scale factor, we can just set

$$\boxed{\boldsymbol{w} = \mathbf{S}_W^{-1}(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)}$$

- This is the optimal solution in the two-class case. If $\boldsymbol{S}_W \propto \boldsymbol{I}$, meaning the pooled covariance matrix is isotropic, then **w** is proportional to the vector that joins the class means. This is an intuitively reasonable direction to project onto.

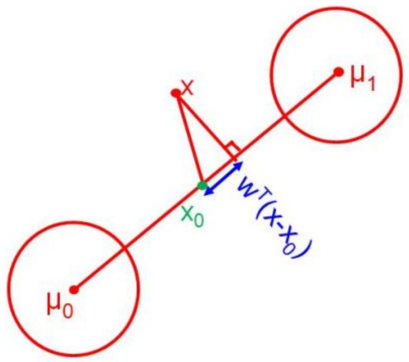


Figure 9.3: Geometry of LDA in the 2 class case where $\mathbf{\Sigma}_1 = \mathbf{\Sigma}_2 = \mathbf{I}$.

Extension to higher dimensions and multiple classes

- We can extend the above idea to multiple classes, and to higher dimensional subspaces, by finding a projection matrix **W** which maps from D to K.
- Let

$$\boldsymbol{z}_n = \boldsymbol{W}\boldsymbol{x}_n$$

be the low dimensional projection of the n'th data point. Let

$$\boldsymbol{m}_c = \frac{1}{N_c}\sum_{n:y_n=c} \boldsymbol{z}_n$$

be the corresponding mean for the c'th class and

$$\boldsymbol{m} = \frac{1}{N}\sum_{c=1}^{C} N_c\boldsymbol{m}_c$$

be the overall mean, both in the low dimensional space.

Extension to higher dimensions and multiple classes

- We define the following scatter matrices:

$$\tilde{\mathbf{S}}_W = \sum_{c=1}^{C} \sum_{n:y_n=c} (\boldsymbol{z}_n - \boldsymbol{m}_c)(\boldsymbol{z}_n - \boldsymbol{m}_c)^{\mathsf{T}}$$

$$\tilde{\mathbf{S}}_B = \sum_{c=1}^{C} N_c(\boldsymbol{m}_c - \boldsymbol{m})(\boldsymbol{m}_c - \boldsymbol{m})^{\mathsf{T}}$$

- Finally, we define the objective function as maximizing the following:

$$J(\mathbf{W}) = \frac{|\tilde{\mathbf{S}}_B|}{|\tilde{\mathbf{S}}_W|} = \frac{|\mathbf{W}^{\mathsf{T}}\mathbf{S}_B\mathbf{W}|}{|\mathbf{W}^{\mathsf{T}}\mathbf{S}_W\mathbf{W}|}$$

Extension to higher dimensions and multiple classes

- where $\mathbf{S}_W$ and $\mathbf{S}_B$ are defined in the original high dimensional space in the obvious way (namely using $\mathbf{x}_n$ instead of $\mathbf{z}_n$, $\boldsymbol{\mu}_c$ instead of $\mathbf{m}_c$, and $\boldsymbol{\mu}$ instead of $\mathbf{m}$).
- The solution can be shown to be $\boldsymbol{W} = \boldsymbol{S}_W^{-1/2}\boldsymbol{U}$, where **U** are the K leading eigenvectors of $\boldsymbol{S}_W^{-1/2}\boldsymbol{S}_B\boldsymbol{S}_W^{-1/2}$, assuming $\mathbf{S}_W$ is non-singular.
- (If it is singular, we can first perform PCA on all the data.)

Extension to higher dimensions and multiple classes

- Figure 9.5 gives an example of this method applied to some D = 10 dimensional speech data, representing C = 11 different vowel sounds. We project to K = 2 dimensions in order to visualize the data. We see that FLDA gives better class separation than PCA.

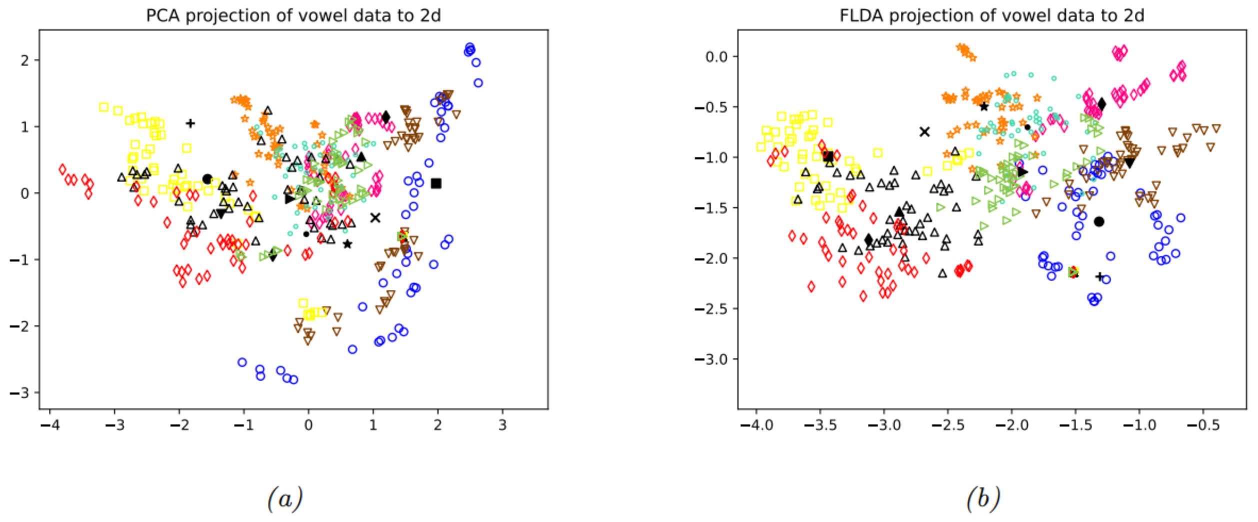


Figure 9.5: (a) PCA projection of vowel data to 2d. (b) FLDA projection of vowel data to 2d. We see there is better class separation in the FLDA case. Adapted from Figure 4.11 of [HTF09]. Generated by fisher_discrim_vowel.ipynb.

- Note that FLDA is restricted to finding at most a $K \leq C - 1$ dimensional linear subspace, no matter how large D, because the rank of the between class scatter matrix $\mathbf{S}_B$ is $C - 1$.
- (The -1 term arises because of the $\boldsymbol{\mu}$ term, which is a linear function of the $\boldsymbol{\mu}_c$.) This is a rather severe restriction which limits the usefulness of FLDA.

Generative vs discriminative classifiers

- **Generative classifier**: a model of the form p(**x**, y) = p(y)p(**x**|y). It can be used to generate examples **x** from each class y.
- **Discriminative classifier**: a model of the form p(y|**x**). It can only be used to discriminate between different classes.

Generative vs discriminative classifiers

- Advantages of discriminative classifiers
- **Better predictive accuracy**. Discriminative classifiers are often much more accurate than generative classifiers. The reason is that the conditional distribution $p(y|\mathbf{x})$ is often much simpler (and therefore easier to learn) than the joint distribution $p(y, \mathbf{x})$. In particular, discriminative models do not need to “waste effort” modeling the distribution of the input features.
- **Can handle feature preprocessing**. They allow us to preprocess the input in arbitrary ways. For example, we can perform a polynomial expansion of the input features, and we can replace a string of words with embedding vectors. It is often hard to define a generative model on such pre-processed data, since the new features can be correlated in complex ways which are hard to model.
- **Well-calibrated probabilities**. Some generative classifiers, such as naive Bayes, make strong independence assumptions which are often not valid. This can result in very extreme posterior class probabilities (very near 0 or 1). Discriminative models, such as logistic regression, are often better calibrated in terms of their probability estimates, although they also sometimes need adjustment).

Generative vs discriminative classifiers

- Advantages of generative classifiers
- **Easy to fit**. Generative classifiers are often very easy to fit. E.g., in Section 9.3.2, we show how to fit a naive Bayes classifier by simple counting and averaging. By contrast, logistic regression requires solving a convex optimization, and neural nets require solving a non-convex optimization problem, both of which are much slower.
- **Can easily handle missing input features**. Sometimes some of the inputs (components of **x**) are not observed. In a generative classifier, there is a simple method for dealing with this, as we show in Section 1.5.5. However, in a discriminative classifier, there is no principled solution to this problem, since the model assumes that **x** is always available to be conditioned on.

Generative vs discriminative classifiers

- Advantages of generative classifiers
- **Can fit classes separately.** In a generative classifier, we estimate the parameters of each class conditional density independently, so we do not have to retrain the model when we add more classes. In contrast, in discriminative models, all the parameters interact, so the whole model must be retrained if we add a new class.
- **Can handle unlabeled training data**. It is easy to use generative models for semi-supervised learning, in which we combine labeled data $\mathcal{D}_{xy} = \{(\boldsymbol{x}_n, y_n)\}$ and unlabeled data, $\mathcal{D}_x = \{\boldsymbol{x}_n\}$. However, this is harder to do with discriminative models, since there is no uniquely optimal way to exploit $\mathcal{D}_x$.
- **May be more robust to spurious features**. A discriminative model p(y|**x**) may pick up on features of the input **x** that can discriminate different values of y in the training set, but which are not robust and do not generalize beyond the training set. These are called spurious features. By contrast, a generative model p(**x**|y) may be better able to capture the causal mechanisms of the underlying data generating process; such causal models can be more robust to distribution shift.

Handling missing features

- Sometimes we are missing parts of the input **x** during training and/or testing. In a generative classifier, we can handle this situation by marginalizing out the missing values. (We assume that the missingness of a feature is not informative about its potential value.)
- By contrast, when using a discriminative model, there is no unique best way to handle missing inputs.

Handling missing features

- For example, suppose we are missing the value of x1. We just have to compute

$$\begin{aligned} p(y=c|\boldsymbol{x}_{2:D},\boldsymbol{\theta}) &\propto p(y=c|\boldsymbol{\pi})p(\boldsymbol{x}_{2:D}|y=c,\boldsymbol{\theta}) \\ &= p(y=c|\boldsymbol{\pi})\sum_{x_1} p(x_1,\boldsymbol{x}_{2:D}|y=c,\boldsymbol{\theta}) \end{aligned}$$

- In Gaussian discriminant analysis, we can marginalize out x_1.

Handling missing features

- If we make the naive Bayes assumption, we can just ignore the likelihood term for x1. This follows because

$$\sum_{x_1} p(x_1, x_{2:D}|y = c, \boldsymbol{\theta}) = \left[\sum_{x_1} p(x_1|\boldsymbol{\theta}_{1c})\right] \prod_{d=2}^{D} p(x_d|\boldsymbol{\theta}_{dc}) = \prod_{d=2}^{D} p(x_d|\boldsymbol{\theta}_{dc})$$

Features are independent.

where we exploited the fact that

$p(x_d|y = c, \boldsymbol{\theta}) = p(x_d|\boldsymbol{\theta}_{dc})$ and $\sum_{x_1} p(x_1|\theta_{1c}) = 1$.

Given class c and parameters θ, the conditional prob. of features x_d only relies on the parameters $\boldsymbol{\theta}_{dc}$ which are related to x_d .