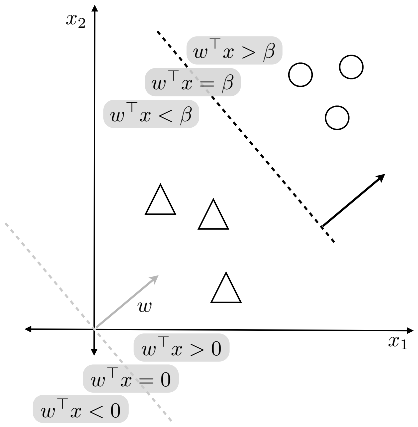
Cognitive Algorithms
Lecture 2

Linear Classification

Klaus-Robert Müller, Johannes Niediek,
Augustin Krause, Joanina Oltersdorff, Ken Schreiber

Technische Universität Berlin
Machine Learning Group

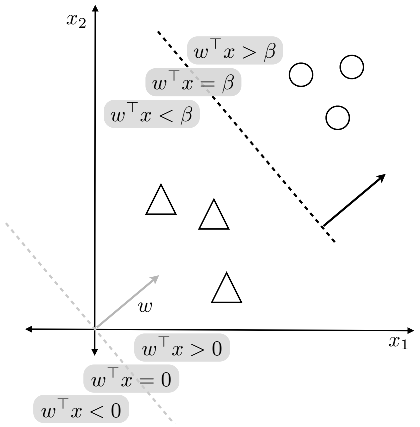
Recap: Linear Classification



Linear decision boundary:

$$\boldsymbol{w}^T \boldsymbol{x} - \beta = 0$$

Recap: Nearest Centroid Classifier



Comparison of distance between data point $\boldsymbol{x} \in \mathbb{R}^d$ to class means $\bar{\boldsymbol{x}}_\Delta, \bar{\boldsymbol{x}}_o \in \mathbb{R}^d$ is equivalent to linear classification with

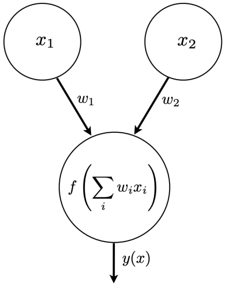
$$\boldsymbol{w} = \bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta$$

and

$$\beta = \frac{1}{2} \cdot \boldsymbol{w}^\top (\bar{\boldsymbol{x}}_o + \bar{\boldsymbol{x}}_\Delta)$$

Note notation change: $\boldsymbol{w}_o = \bar{\boldsymbol{x}}_o$, $\boldsymbol{w}_\Delta = \bar{\boldsymbol{x}}_\Delta$.

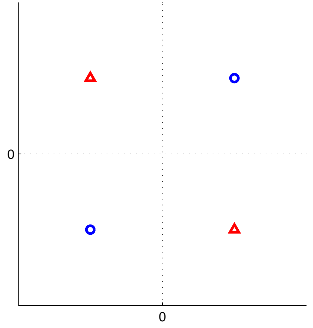
Recap: Perceptron



Problem	Classification
Model	$\hat{y} = f(\boldsymbol{w}^T x)$
Loss function	$-\sum_{m\in\mathcal{M}} \boldsymbol{w}^T \boldsymbol{x}_m y_m$
Optimization	stochastic gradient descent (SGD)

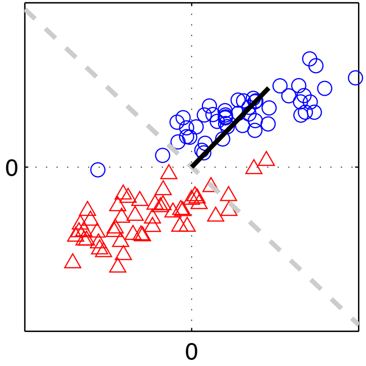
Problems with Nearest Centroid Classification

Not linearly separable data



Solution
Non-linear methods (later in this course)

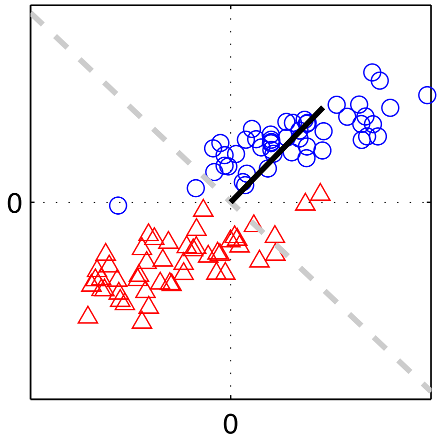
Correlated data



Solution
(Fisher's) Linear Discriminant Analysis

What is correlation?

Let's go through some definitions first
$\rightarrow$ They will be useful later



Random variables

Denote by Ω the sample space, the set of all possible outcomes of an experiment.
A mapping $X : \Omega \to \mathbb{R}$ which assigns a real value to every elementary event, is called a real-valued random variable.

Example: coin toss $\Rightarrow \Omega = \{\text{head}, \text{tail}\}$

$$X(\omega) = \begin{cases} 0, & \text{if } \omega = \text{tail} \\ 1, & \text{if } \omega = \text{head} \end{cases} \quad \text{for } \omega \in \Omega$$

Random variables

Denote by Ω the sample space, the set of all possible outcomes of an experiment.
A mapping $X : \Omega \to \mathbb{R}$ which assigns a real value to every elementary event, is called a real-valued random variable.

Example: coin toss $\Rightarrow \Omega = \{\text{head}, \text{tail}\}$

$$X(\omega) = \begin{cases} 0, & \text{if} \quad \omega = \text{tail} \\ 1, & \text{if} \quad \omega = \text{head} \end{cases} \qquad \text{for } \omega \in \Omega$$

- We use random variables to model the world
- In this course, we take a practical approach and introduce concepts when we need them

Probabilities and expected values

If X is a **discrete random variable**, i.e. if X takes on only finitely[1] many values, we can assign probabilities $p_i \in [0, 1]$ to the values x_i of X.
A probability of p_i means that out of very many trials, a fraction of p_i will have value x_i.

The **expected value** of X is given by

$$\mathbb{E}[X] = \sum_i p_i x_i \, .$$

Example: coin toss with $p_0 = p_1 = \frac{1}{2}$, then

$$\mathbb{E}[X] = 0 \cdot p_0 + 1 \cdot p_1 = \frac{1}{2} \, .$$

[1]Strictly speaking, a discrete random variable takes finitely many or countably many values.

Probability distributions and expected values

The probabilities of the values of a **continuous random variable** are described by a **probability density function**, a function $p : X(\Omega) \to \mathbb{R}_+$ with $\int_{X(\Omega)} p(x)\,\mathrm{d}x = 1$. The probability of observing a value in $[a, b] \subset \mathbb{R}$ is given by

$$\int_a^b p(x)\,\mathrm{d}x\,.$$

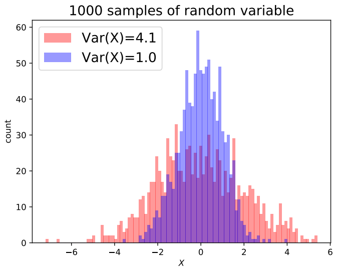
The expected value of X is given by

$$\mathbb{E}[X] = \int_{-\infty}^{\infty} x \cdot p(x)\,\mathrm{d}x\,.$$

Variance

measure of variability of X around its mean

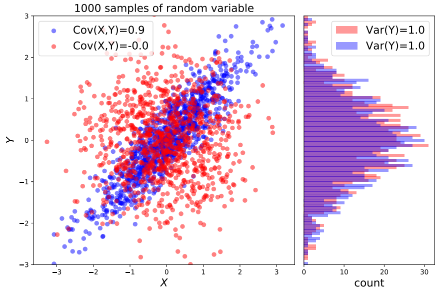
$$\mathrm{Var}(X) = \mathbb{E}[(X - \mathbb{E}[X])^2]$$



Covariance

measure of the joint variability of X and Y

$$\mathrm{Cov}(X, Y) := \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])]$$



Covariance

$$\mathrm{Cov}(X, Y) := \mathbb{E}[(X - \mathbb{E}(X))(Y - \mathbb{E}(Y))]$$

Correlation

$$\mathrm{Corr}(X, Y) := \frac{\mathrm{Cov}(X, Y)}{\sqrt{\mathrm{Var}(X)}\sqrt{\mathrm{Var}(Y)}} \in [-1, 1]\,.$$

normalized covariance

Covariance

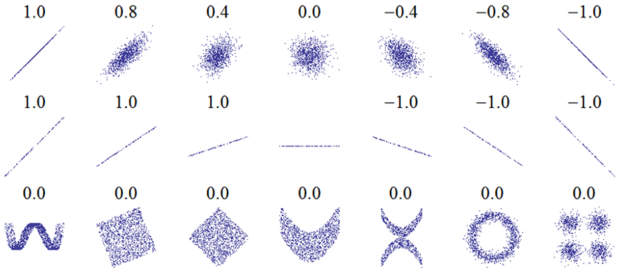
$$\mathrm{Cov}(X,Y) := \mathbb{E}[(X-\mathbb{E}(X))(Y-\mathbb{E}(Y))]$$

Correlation

$$\mathrm{Corr}(X,Y) := \frac{\mathrm{Cov}(X,Y)}{\sqrt{\mathrm{Var}(X)}\sqrt{\mathrm{Var}(Y)}} \in [-1,1]\,.$$

normalized covariance

Indicate the strength of a **linear** relationship



Correlation vs. dependence vs. causation

Consider two random variables X, Y.

- X and Y are called **independent** if $p(X, Y) = p(X) \cdot p(Y)$
- X and Y are called **uncorrelated** if $\mathrm{Corr}(X, Y) = 0$

We might call X and Y **causally related** if X influences Y or vice versa.

Note

- X and Y independent implies X and Y uncorrelated
- X and Y uncorrelated *does not* imply X and Y independent! (example $Y = X^2$ on $[-1, 1]$)
- X and Y dependent does not imply X and Y causally related

Normal distribution

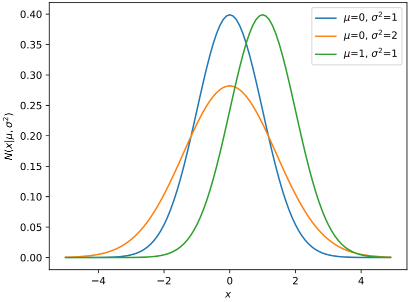
Parameters:

- Mean $\mu \in \mathbb{R}$
- Variance $\sigma^2 \in \mathbb{R}$

The probability density function

$$\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}\right)$$

defines the **normal distribution** or **Gaussian distribution** with parameters μ, σ^2.



The parameters μ and σ^2 are called 'mean' and 'variance', because the mean $\mathbb{E}[X]$ and variance $\mathbb{E}[(X-\mathbb{E}[X])^2]$ of the distribution are μ and σ^2

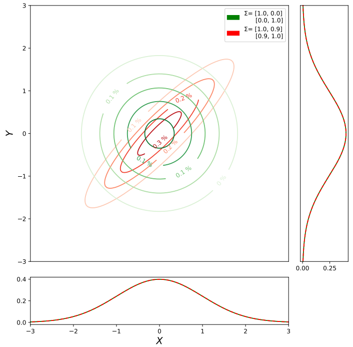
Multivariate normal distribution

For d dimensions:

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

Parameters:

- Mean $\boldsymbol{\mu} \in \mathbb{R}^d$
- Covariance matrix $\Sigma \in \mathbb{R}^{d \times d}$



Estimating the covariance matrices

Given n data points $\boldsymbol{x}_i \in \mathbb{R}^D$ in a data matrix $X \in \mathbb{R}^{D \times n}$
the empirical estimate of the **covariance matrix** is defined as

$$\hat{\Sigma} = \frac{1}{n} (X - \bar{X})(X - \bar{X})^\top ,$$

where the estimate of the expected value is given by the mean

$$\bar{\boldsymbol{x}} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i , \quad \bar{X} = (\bar{\boldsymbol{x}}, \bar{\boldsymbol{x}}, \ldots, \bar{\boldsymbol{x}}) \in \mathbb{R}^{D \times n}$$

The diagonal entries of $\hat{\Sigma}$ are estimates of the variance.

We call $(X - \bar{X})(X - \bar{X})^\top$ the *empirical scatter matrix*

Estimating the covariance matrices

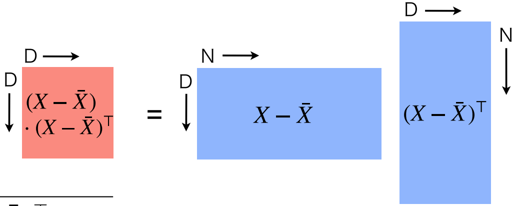
Given n data points $\boldsymbol{x}_i \in \mathbb{R}^D$ in a data matrix $X \in \mathbb{R}^{D \times n}$ the empirical estimate of the **covariance matrix** is defined as

$$\hat{\Sigma} = \frac{1}{n}(X - \bar{X})(X - \bar{X})^\top ,$$

where the estimate of the expected value is given by the mean

$$\bar{\boldsymbol{x}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i , \quad \bar{X} = (\bar{\boldsymbol{x}}, \bar{\boldsymbol{x}}, \dots, \bar{\boldsymbol{x}}) \in \mathbb{R}^{D \times n}$$

The diagonal entries of $\hat{\Sigma}$ are estimates of the variance.

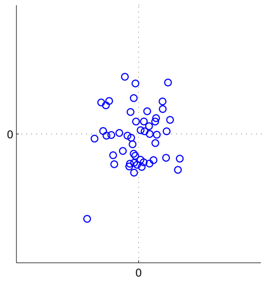


We call $(X - \bar{X})(X - \bar{X})^\top$ the *empirical scatter matrix*

Create correlated data from uncorrelated data

We can generate correlated data using a diagonal scaling matrix D and a rotation R. We assume centered data here (i.e. $\bar{X} = 0$), so $\hat{\Sigma} = \frac{1}{n}XX^\top$

Uncorrelated



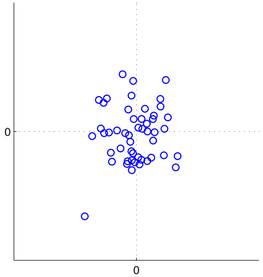
$$x \sim \mathcal{N}(0,1)$$

$$\frac{1}{n}XX^\top = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Create correlated data from uncorrelated data

We can generate correlated data using a diagonal scaling matrix D and a rotation R. We assume centered data here (i.e. $\bar{X} = 0$), so $\hat{\Sigma} = \frac{1}{n} XX^\top$

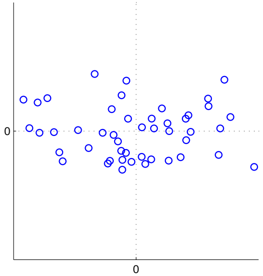
Uncorrelated



$x \sim \mathcal{N}(0,1)$

$\frac{1}{n} XX^\top = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

Uncorrelated, scaled



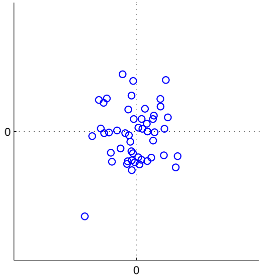
$\begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix} X$

$\frac{1}{n} XX^\top = \begin{bmatrix} 9 & 0 \\ 0 & 1 \end{bmatrix}$

Create correlated data from uncorrelated data

We can generate correlated data using a diagonal scaling matrix D and a rotation R. We assume centered data here (i.e. $\bar{X} = 0$), so $\hat{\Sigma} = \frac{1}{n} X X^\top$

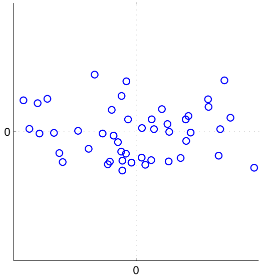
Uncorrelated



$x \sim \mathcal{N}(0, 1)$

$\frac{1}{n} X X^\top = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

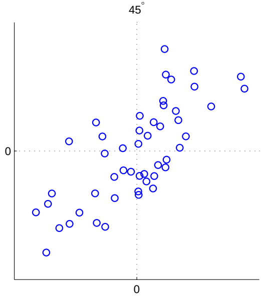
Uncorrelated, scaled



$\begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix} X$

$\frac{1}{n} X X^\top = \begin{bmatrix} 9 & 0 \\ 0 & 1 \end{bmatrix}$

Scaled, rotated by 45°



$\begin{bmatrix} cos(\phi) & -sin(\phi) \\ sin(\phi) & cos(\phi) \end{bmatrix} \begin{bmatrix} 3 & 0 \\ 0 & 1 \end{bmatrix} X$

$\frac{1}{n} X X^\top = \begin{bmatrix} 5 & 4 \\ 4 & 5 \end{bmatrix}$

Ronald A. Fisher

Ronald A. Fisher (1890 – 1962)

Founder of modern statistics

Interested in Biology

Suggested *Linear Discriminant Analysis* (LDA)

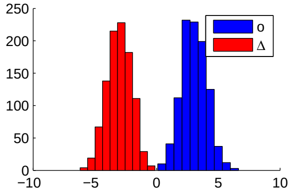
Held some very problematic opinions

https://priceonomics.com/why-the-father-of-modern-statistics-didnt-believe/, https://statmodeling.stat.columbia.edu/2020/08/01/ra-fisher-and-the-science-of-hatred/

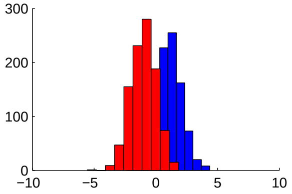
The Fisher Criterion - measure for class separability

Consider one dimensional data and two classes

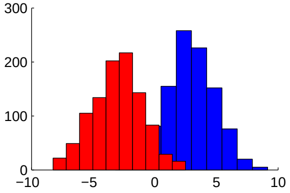
Good Class Separation



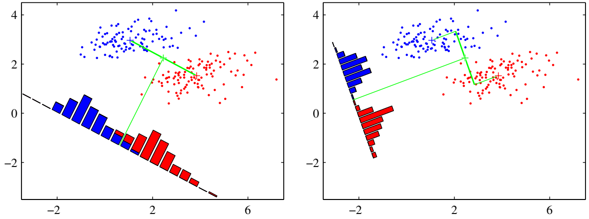
Bad Class Separation:
Close means



Bad Class Separation:
Large Variance per class



Linear Discriminant Analysis

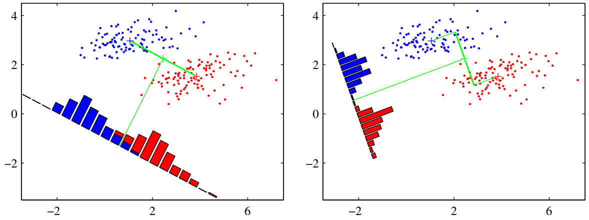


Goal: Find a (normal vector of a linear decision boundary) $\boldsymbol{w} \in \mathbb{R}^d$ that

Maximizes mean class difference, and

Minimizes variance in each class

Linear Discriminant Analysis



Maximize the **Fisher criterion**:

$$J(\boldsymbol{w}) = \frac{\text{between class variance}}{\text{within class variance}} = \frac{(\mu_o - \mu_\Delta)^2}{\sigma_o^2 + \sigma_\Delta^2}$$

where $\boldsymbol{x}_{1o}, \ldots, \boldsymbol{x}_{n_o o} \in \mathbb{R}^d$ and

$\mu_o = \frac{1}{n_o} \sum_{i=1}^{n_o} \boldsymbol{w}^\top \boldsymbol{x}_{io}$ and $\sigma_o^2 = \frac{1}{n_o} \sum_{i=1}^{n_o} (\boldsymbol{w}^\top \boldsymbol{x}_{io} - \mu_o)^2$ and similarly for Δ.

Linear Discriminant Analysis

Rewrite Fisher criterion to separate out $\boldsymbol{w}$-dependence using

$$\bar{\boldsymbol{x}}_o := \frac{1}{n_o}\sum_{i=1}^{n_o} \boldsymbol{x}_{io} \quad \Rightarrow \quad \mu_o = \frac{1}{n_o}\sum_{i=1}^{n_o} \boldsymbol{w}^\top \boldsymbol{x}_{io} = \boldsymbol{w}^\top \bar{\boldsymbol{x}}_o \,, \quad \sigma_o = \frac{1}{n_o}\sum_{i=1}^{n_o} (\boldsymbol{w}^\top \boldsymbol{x}_{io} - \mu_o)^2 = \frac{1}{n_o}\sum_{i=1}^{n_o} (\boldsymbol{w}^\top (\boldsymbol{x}_{io} - \bar{\boldsymbol{x}}_o))^2$$

Linear Discriminant Analysis

Rewrite Fisher criterion to separate out $\boldsymbol{w}$-dependence using

$$\bar{\boldsymbol{x}}_o := \frac{1}{n_o}\sum_{i=1}^{n_o} \boldsymbol{x}_{io} \quad \Rightarrow \quad \mu_o = \frac{1}{n_o}\sum_{i=1}^{n_o} \boldsymbol{w}^\top \boldsymbol{x}_{io} = \boldsymbol{w}^\top \bar{\boldsymbol{x}}_o \,, \quad \sigma_o = \frac{1}{n_o}\sum_{i=1}^{n_o} (\boldsymbol{w}^\top \boldsymbol{x}_{io} - \mu_o)^2 = \frac{1}{n_o}\sum_{i=1}^{n_o} (\boldsymbol{w}^\top (\boldsymbol{x}_{io} - \bar{\boldsymbol{x}}_o))^2$$

Hence,

$$(\mu_o - \mu_\Delta)^2 = (\boldsymbol{w}^\top (\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta))^2 = \boldsymbol{w}^\top \underbrace{(\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta)(\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta)^\top}_{S_B - \text{“between class scatter”}} \boldsymbol{w}\,.$$

Linear Discriminant Analysis

Rewrite Fisher criterion to separate out $\boldsymbol{w}$-dependence using

$$\bar{\boldsymbol{x}}_o := \frac{1}{n_o}\sum_{i=1}^{n_o}\boldsymbol{x}_{io} \quad\Rightarrow\quad \mu_o = \frac{1}{n_o}\sum_{i=1}^{n_o}\boldsymbol{w}^\top\boldsymbol{x}_{io} = \boldsymbol{w}^\top\bar{\boldsymbol{x}}_o\,, \quad \sigma_o = \frac{1}{n_o}\sum_{i=1}^{n_o}(\boldsymbol{w}^\top\boldsymbol{x}_{io}-\mu_o)^2 = \frac{1}{n_o}\sum_{i=1}^{n_o}(\boldsymbol{w}^\top(\boldsymbol{x}_{io}-\bar{\boldsymbol{x}}_o))^2$$

Hence,

$$(\mu_o-\mu_\Delta)^2 = (\boldsymbol{w}^\top(\bar{\boldsymbol{x}}_o-\bar{\boldsymbol{x}}_\Delta))^2 = \boldsymbol{w}^\top\underbrace{(\bar{\boldsymbol{x}}_o-\bar{\boldsymbol{x}}_\Delta)(\bar{\boldsymbol{x}}_o-\bar{\boldsymbol{x}}_\Delta)^\top}_{S_B-\text{“between class scatter”}}\boldsymbol{w}\,.$$

$$\sigma_o^2+\sigma_\Delta^2 = \frac{1}{n_o}\sum_{i=1}^{n_o}(\boldsymbol{w}^\top(\boldsymbol{x}_{io}-\bar{\boldsymbol{x}}_o))^2 + \frac{1}{n_\Delta}\sum_{j=1}^{n_\Delta}(\boldsymbol{w}^\top(\boldsymbol{x}_{j\Delta}-\bar{\boldsymbol{x}}_\Delta))^2$$

$$= \boldsymbol{w}^\top\underbrace{\left[\frac{1}{n_o}\sum_{i=1}^{n_o}(\boldsymbol{x}_{io}-\bar{\boldsymbol{x}}_o)(\boldsymbol{x}_{io}-\bar{\boldsymbol{x}}_o)^\top + \frac{1}{n_\Delta}\sum_{j=1}^{n_\Delta}(\boldsymbol{x}_{j\Delta}-\bar{\boldsymbol{x}}_\Delta)(\boldsymbol{x}_{j\Delta}-\bar{\boldsymbol{x}}_\Delta)^\top\right]}_{S_W-\text{“within class scatter”}}\boldsymbol{w}\,.$$

Linear Discriminant Analysis

Rewrite Fisher criterion to separate out $\boldsymbol{w}$-dependence using

$$\bar{\boldsymbol{x}}_o := \frac{1}{n_o}\sum_{i=1}^{n_o} \boldsymbol{x}_{io} \quad \Rightarrow \quad \mu_o = \frac{1}{n_o}\sum_{i=1}^{n_o} \boldsymbol{w}^\top \boldsymbol{x}_{io} = \boldsymbol{w}^\top \bar{\boldsymbol{x}}_o\,, \quad \sigma_o = \frac{1}{n_o}\sum_{i=1}^{n_o} (\boldsymbol{w}^\top \boldsymbol{x}_{io} - \mu_o)^2 = \frac{1}{n_o}\sum_{i=1}^{n_o} (\boldsymbol{w}^\top (\boldsymbol{x}_{io} - \bar{\boldsymbol{x}}_o))^2$$

Hence,

$$(\mu_o - \mu_\Delta)^2 = (\boldsymbol{w}^\top (\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta))^2 = \boldsymbol{w}^\top \underbrace{(\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta)(\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta)^\top}_{S_B - \text{“between class scatter”}} \boldsymbol{w}\,.$$

$$\sigma_o^2 + \sigma_\Delta^2 = \frac{1}{n_o}\sum_{i=1}^{n_o} (\boldsymbol{w}^\top (\boldsymbol{x}_{io} - \bar{\boldsymbol{x}}_o))^2 + \frac{1}{n_\Delta}\sum_{j=1}^{n_\Delta} (\boldsymbol{w}^\top (\boldsymbol{x}_{j\Delta} - \bar{\boldsymbol{x}}_\Delta))^2$$

$$= \boldsymbol{w}^\top \underbrace{\left[\frac{1}{n_o}\sum_{i=1}^{n_o} (\boldsymbol{x}_{io} - \bar{\boldsymbol{x}}_o)(\boldsymbol{x}_{io} - \bar{\boldsymbol{x}}_o)^\top + \frac{1}{n_\Delta}\sum_{j=1}^{n_\Delta} (\boldsymbol{x}_{j\Delta} - \bar{\boldsymbol{x}}_\Delta)(\boldsymbol{x}_{j\Delta} - \bar{\boldsymbol{x}}_\Delta)^\top\right]}_{S_W - \text{“within class scatter”}} \boldsymbol{w}\,.$$

And therefore ($\boldsymbol{w} \neq 0$),

$$J(\boldsymbol{w}) = \boldsymbol{w}^\top S_B \boldsymbol{w} \big/ \boldsymbol{w}^\top S_W \boldsymbol{w}\,.$$

Linear Discriminant Analysis

The optimal weight vector $\boldsymbol{w}$ is given by

$$\boldsymbol{w} = \underset{\boldsymbol{w}'}{\operatorname{argmax}}\, J(\boldsymbol{w}') = \underset{\boldsymbol{w}'}{\operatorname{argmax}} \frac{\boldsymbol{w}'^{\top} S_B \boldsymbol{w}'}{\boldsymbol{w}'^{\top} S_W \boldsymbol{w}'}$$

To optimize the Fisher criterion, we set its derivative (with respect to $\boldsymbol{w}$) to 0

Linear Discriminant Analysis

The optimal weight vector $\boldsymbol{w}$ is given by

$$\boldsymbol{w} = \underset{\boldsymbol{w}'}{\operatorname{argmax}}\, J(\boldsymbol{w}') = \underset{\boldsymbol{w}'}{\operatorname{argmax}} \frac{\boldsymbol{w}'^\top S_B \boldsymbol{w}'}{\boldsymbol{w}'^\top S_W \boldsymbol{w}'}$$

To optimize the Fisher criterion, we set its derivative (with respect to $\boldsymbol{w}$) to 0

$$0 \quad = \quad \frac{\partial}{\partial \boldsymbol{w}} J(\boldsymbol{w})\Big|_{\boldsymbol{w}} = \frac{(\boldsymbol{w}^\top S_W \boldsymbol{w}) S_B \boldsymbol{w} - (\boldsymbol{w}^\top S_B \boldsymbol{w}) S_W \boldsymbol{w}}{(\boldsymbol{w}^\top S_W \boldsymbol{w})^2}$$

Linear Discriminant Analysis

The optimal weight vector $\boldsymbol{w}$ is given by

$$\boldsymbol{w} = \underset{\boldsymbol{w}'}{\operatorname{argmax}}\, J(\boldsymbol{w}') = \underset{\boldsymbol{w}'}{\operatorname{argmax}} \frac{\boldsymbol{w}'^\top S_B \boldsymbol{w}'}{\boldsymbol{w}'^\top S_W \boldsymbol{w}'}$$

To optimize the Fisher criterion, we set its derivative (with respect to $\boldsymbol{w}$) to 0

$$\begin{aligned} 0 &= \frac{\partial}{\partial \boldsymbol{w}} J(\boldsymbol{w})\Big|_{\boldsymbol{w}} = \frac{(\boldsymbol{w}^\top S_W \boldsymbol{w}) S_B \boldsymbol{w} - (\boldsymbol{w}^\top S_B \boldsymbol{w}) S_W \boldsymbol{w}}{(\boldsymbol{w}^\top S_W \boldsymbol{w})^2} \\ (\boldsymbol{w}^\top S_B \boldsymbol{w}) S_W \boldsymbol{w} &= (\boldsymbol{w}^\top S_W \boldsymbol{w}) S_B \boldsymbol{w} \\ S_W \boldsymbol{w} &= S_B \boldsymbol{w} \underbrace{\frac{\boldsymbol{w}^\top S_W \boldsymbol{w}}{\boldsymbol{w}^\top S_B \boldsymbol{w}}}_{scalar \equiv \lambda} \end{aligned}$$

Linear Discriminant Analysis

$$\boldsymbol{w} = \operatorname*{argmax}_{\boldsymbol{w}'} \frac{\boldsymbol{w}'^\top S_B \boldsymbol{w}'}{\boldsymbol{w}'^\top S_W \boldsymbol{w}'}$$

$$\rightarrow S_W \boldsymbol{w} = S_B \boldsymbol{w} \lambda$$

Linear Discriminant Analysis

$$\boldsymbol{w} = \underset{\boldsymbol{w}'}{\operatorname{argmax}} \frac{\boldsymbol{w}'^\top S_B \boldsymbol{w}'}{\boldsymbol{w}'^\top S_W \boldsymbol{w}'}$$

$$\rightarrow S_W \boldsymbol{w} = S_B \boldsymbol{w} \lambda$$

Now we plug $S_B = (\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta)(\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta)^\top$ in

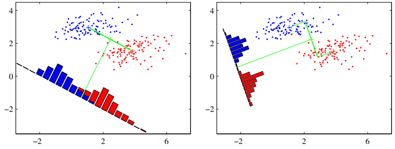
$$S_B \boldsymbol{w} = (\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta) \underbrace{(\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta)^\top \boldsymbol{w}}_{\text{scalar}}$$

finally, left multiplying with S_W^{-1} yields

$$\boldsymbol{w} \propto S_W^{-1}(\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta).$$

($\propto$ denotes proportionality, e.g. $x \propto 2x$)

Interim summary



Goal

Find $\boldsymbol{w} \in \mathbb{R}^d$ that

- maximizes mean class difference
- minimizes variance in each class

Formalization

Maximize the **Fisher criterion**

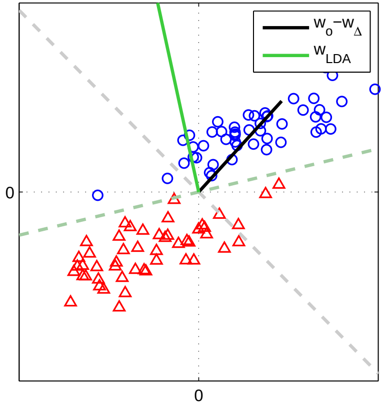
$$J(\boldsymbol{w}) = \frac{\text{between class variance}}{\text{within class variance}} = \frac{(\mu_o - \mu_\Delta)^2}{\sigma_o^2 + \sigma_\Delta^2}$$

Solution

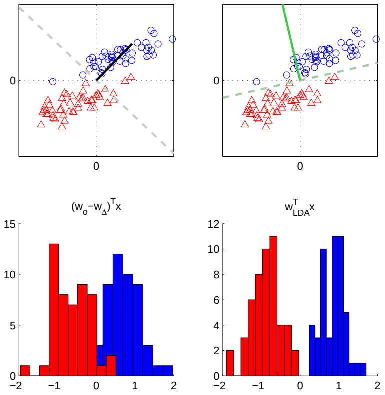
After some calculations...

$$\boldsymbol{w} \propto S_W^{-1}(\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta)$$

Linear Discriminant Analysis vs Nearest Centroid Classifier



Linear Discriminant Analysis vs Nearest Centroid Classifier



If correlated data are the problem, why don't we decorrelate the data and then apply the nearest-centroid classifier?

How can we decorrelate the data?

How can we decorrelate the data?

- *Decorrelating* refers to transforming to a diagonal empirical covariance matrix $\hat{\Sigma}$

How can we decorrelate the data?

- *Decorrelating* refers to transforming to a diagonal empirical covariance matrix $\hat{\Sigma}$
- *Whitening* transforms to a unit $\hat{\Sigma}$:

How can we decorrelate the data?

- *Decorrelating* refers to transforming to a diagonal empirical covariance matrix $\hat{\Sigma}$
- *Whitening* transforms to a unit $\hat{\Sigma}$:
  1. For a data matrix $X \in \mathbb{R}^{D \times n}$, calculate $\hat{\Sigma} = \frac{1}{n}(X - \bar{X})(X - \bar{X})^T$

How can we decorrelate the data?

- *Decorrelating* refers to transforming to a diagonal empirical covariance matrix $\hat{\Sigma}$
- *Whitening* transforms to a unit $\hat{\Sigma}$:
  1. For a data matrix $X \in \mathbb{R}^{D \times n}$, calculate $\hat{\Sigma} = \frac{1}{n}(X - \bar{X})(X - \bar{X})^T$
  2. Calculate eigenvalue decompostion $U \Lambda U^T = \hat{\Sigma}$ with Λ diagonal

How can we decorrelate the data?

- *Decorrelating* refers to transforming to a diagonal empirical covariance matrix $\hat{\Sigma}$
- *Whitening* transforms to a unit $\hat{\Sigma}$:
  1. For a data matrix $X \in \mathbb{R}^{D \times n}$, calculate $\hat{\Sigma} = \frac{1}{n}(X - \bar{X})(X - \bar{X})^T$
  2. Calculate eigenvalue decompostion $U \Lambda U^T = \hat{\Sigma}$ with Λ diagonal
  3. Transform X: $\tilde{X} = \Lambda^{-1/2} U^T X$

How can we decorrelate the data?

- *Decorrelating* refers to transforming to a diagonal empirical covariance matrix $\hat{\Sigma}$
- *Whitening* transforms to a unit $\hat{\Sigma}$:
  1. For a data matrix $X \in \mathbb{R}^{D \times n}$, calculate $\hat{\Sigma} = \frac{1}{n}(X - \bar{X})(X - \bar{X})^T$
  2. Calculate eigenvalue decompostion $U \Lambda U^T = \hat{\Sigma}$ with Λ diagonal
  3. Transform X: $\tilde{X} = \Lambda^{-1/2} U^T X$

New Covariance

How can we decorrelate the data?

- *Decorrelating* refers to transforming to a diagonal empirical covariance matrix $\hat{\Sigma}$
- *Whitening* transforms to a unit $\hat{\Sigma}$:
  1. For a data matrix $X \in \mathbb{R}^{D \times n}$, calculate $\hat{\Sigma} = \frac{1}{n}(X - \bar{X})(X - \bar{X})^T$
  2. Calculate eigenvalue decompostion $U \Lambda U^T = \hat{\Sigma}$ with Λ diagonal
  3. Transform X: $\tilde{X} = \Lambda^{-1/2} U^T X$

New Covariance

$$\tilde{\Sigma} = \frac{1}{n}(\tilde{X} - \bar{\tilde{X}})(\tilde{X} - \bar{\tilde{X}})^T$$

How can we decorrelate the data?

- *Decorrelating* refers to transforming to a diagonal empirical covariance matrix $\hat{\Sigma}$
- *Whitening* transforms to a unit $\hat{\Sigma}$:
  1. For a data matrix $X \in \mathbb{R}^{D \times n}$, calculate $\hat{\Sigma} = \frac{1}{n}(X - \bar{X})(X - \bar{X})^T$
  2. Calculate eigenvalue decompostion $U \Lambda U^T = \hat{\Sigma}$ with Λ diagonal
  3. Transform X: $\tilde{X} = \Lambda^{-1/2} U^T X$

New Covariance

$$\begin{aligned}\tilde{\Sigma} &= \frac{1}{n}(\tilde{X} - \bar{\tilde{X}})(\tilde{X} - \bar{\tilde{X}})^T \\ &= \Lambda^{-1/2} U^T \underbrace{\frac{1}{n}(X - \bar{X})(X - \bar{X})^T}_{\hat{\Sigma} = U \Lambda U^T} U \Lambda^{-1/2}\end{aligned}$$

How can we decorrelate the data?

- *Decorrelating* refers to transforming to a diagonal empirical covariance matrix $\hat{\Sigma}$
- *Whitening* transforms to a unit $\hat{\Sigma}$:
  1. For a data matrix $X \in \mathbb{R}^{D\times n}$, calculate $\hat{\Sigma} = \frac{1}{n}(X - \bar{X})(X - \bar{X})^T$
  2. Calculate eigenvalue decompostion $U\Lambda U^T = \hat{\Sigma}$ with Λ diagonal
  3. Transform X: $\tilde{X} = \Lambda^{-1/2}U^T X$

New Covariance

$$\tilde{\Sigma} = \frac{1}{n}(\tilde{X} - \bar{\tilde{X}})(\tilde{X} - \bar{\tilde{X}})^T$$
$$= \Lambda^{-1/2}U^T \underbrace{\frac{1}{n}(X - \bar{X})(X - \bar{X})^T}_{\hat{\Sigma}=U\Lambda U^T} U\Lambda^{-1/2}$$
$$= \Lambda^{-1/2}U^T U\Lambda U^T U\Lambda^{-1/2}$$

How can we decorrelate the data?

- *Decorrelating* refers to transforming to a diagonal empirical covariance matrix $\hat{\Sigma}$
- *Whitening* transforms to a unit $\hat{\Sigma}$:
  1. For a data matrix $X \in \mathbb{R}^{D \times n}$, calculate $\hat{\Sigma} = \frac{1}{n}(X - \bar{X})(X - \bar{X})^T$
  2. Calculate eigenvalue decompostion $U \Lambda U^T = \hat{\Sigma}$ with Λ diagonal
  3. Transform X: $\tilde{X} = \Lambda^{-1/2} U^T X$

New Covariance

$$
\begin{aligned}
\tilde{\Sigma} &= \frac{1}{n}(\tilde{X} - \bar{\tilde{X}})(\tilde{X} - \bar{\tilde{X}})^T \\
&= \Lambda^{-1/2} U^T \underbrace{\frac{1}{n}(X - \bar{X})(X - \bar{X})^T}_{\hat{\Sigma} = U \Lambda U^T} U \Lambda^{-1/2} \\
&= \Lambda^{-1/2} U^T U \Lambda U^T U \Lambda^{-1/2} \\
&= I
\end{aligned}
$$

How can we decorrelate the data?

- *Decorrelating* refers to transforming to a diagonal empirical covariance matrix $\hat{\Sigma}$
- *Whitening* transforms to a unit $\hat{\Sigma}$:
  1. For a data matrix $X \in \mathbb{R}^{D \times n}$, calculate $\hat{\Sigma} = \frac{1}{n}(X - \bar{X})(X - \bar{X})^T$
  2. Calculate eigenvalue decompostion $U \Lambda U^T = \hat{\Sigma}$ with Λ diagonal
  3. Transform X: $\tilde{X} = \Lambda^{-1/2} U^T X$

New Covariance

$$
\begin{aligned}
\tilde{\Sigma} &= \frac{1}{n}(\tilde{X} - \bar{\tilde{X}})(\tilde{X} - \bar{\tilde{X}})^T \\
&= \Lambda^{-1/2} U^T \underbrace{\frac{1}{n}(X - \bar{X})(X - \bar{X})^T}_{\hat{\Sigma} = U \Lambda U^T} U \Lambda^{-1/2} \\
&= \Lambda^{-1/2} U^T U \Lambda U^T U \Lambda^{-1/2} \\
&= I
\end{aligned}
$$

There is more than one way of whitening (we can multiply $\tilde{X}$ with any orthogonal matrix $OO^T = I$)

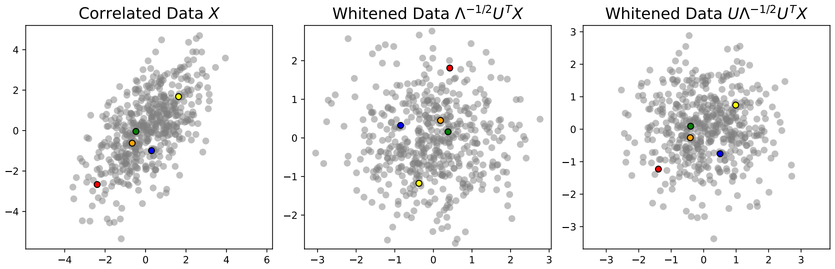
Whitening

Transforms data to data with covariance matrix that is the identity.

$\rightarrow$ Data are decorrelated after whitening

Often used as part of preprocessing

Leads to more numeric stability



Linear Discriminant Analysis

For centered data, we have

$$S = \frac{1}{n_\Delta + n_o} X X^T = S_W + \frac{n_\Delta n_o}{n_\Delta + n_o} S_B$$

Compare Subsection 4.1.5 in PRML[2] and Exercise 4.6 in PRML, a solution of the exercise is available here `https://github.com/zhengqigao/PRML-Solution-Manual`.

Then, for the LDA weight vector $\boldsymbol{w}$:

[2] C. M. Bishop, Pattern Recognition and Machine Learning, freely available here.

Linear Discriminant Analysis

For centered data, we have

$$S = \frac{1}{n_\Delta + n_o} X X^T = S_W + \frac{n_\Delta n_o}{n_\Delta + n_o} S_B$$

Compare Subsection 4.1.5 in PRML[2] and Exercise 4.6 in PRML, a solution of the exercise is available here `https://github.com/zhengqigao/PRML-Solution-Manual`.

Then, for the LDA weight vector $\boldsymbol{w}$:

$$S_W \boldsymbol{w} \propto S_B \boldsymbol{w}$$

[2] C. M. Bishop, Pattern Recognition and Machine Learning, freely available here.

Linear Discriminant Analysis

For centered data, we have

$$S = \frac{1}{n_\Delta + n_o} X X^T = S_W + \frac{n_\Delta n_o}{n_\Delta + n_o} S_B$$

Compare Subsection 4.1.5 in PRML[2] and Exercise 4.6 in PRML, a solution of the exercise is available here `https://github.com/zhengqigao/PRML-Solution-Manual`.

Then, for the LDA weight vector $\boldsymbol{w}$:

$$S_W \boldsymbol{w} \propto S_B \boldsymbol{w}$$

$$(S - \frac{n_\Delta n_o}{n_\Delta + n_o} S_B)\boldsymbol{w} \propto S_B \boldsymbol{w}$$

[2]C. M. Bishop, Pattern Recognition and Machine Learning, freely available here.

Linear Discriminant Analysis

For centered data, we have

$$S = \frac{1}{n_\Delta + n_o} X X^T = S_W + \frac{n_\Delta n_o}{n_\Delta + n_o} S_B$$

Compare Subsection 4.1.5 in PRML[2] and Exercise 4.6 in PRML, a solution of the exercise is available here `https://github.com/zhengqigao/PRML-Solution-Manual`.

Then, for the LDA weight vector $\boldsymbol{w}$:

$$S_W \boldsymbol{w} \propto S_B \boldsymbol{w}$$
$$(S - \frac{n_\Delta n_o}{n_\Delta + n_o} S_B)\boldsymbol{w} \propto S_B \boldsymbol{w}$$
$$S\boldsymbol{w} \propto S_B \boldsymbol{w} \propto \bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta$$

[2] C. M. Bishop, Pattern Recognition and Machine Learning, freely available here.

Linear Discriminant Analysis

For centered data, we have

$$S = \frac{1}{n_\Delta + n_o} XX^T = S_W + \frac{n_\Delta n_o}{n_\Delta + n_o} S_B$$

Compare Subsection 4.1.5 in PRML[2] and Exercise 4.6 in PRML, a solution of the exercise is available here `https://github.com/zhengqigao/PRML-Solution-Manual`.

Then, for the LDA weight vector $\boldsymbol{w}$:

$$\begin{aligned} S_W \boldsymbol{w} &\propto S_B \boldsymbol{w} \\ (S - \frac{n_\Delta n_o}{n_\Delta + n_o} S_B)\boldsymbol{w} &\propto S_B \boldsymbol{w} \\ S\boldsymbol{w} &\propto S_B \boldsymbol{w} \propto \bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta \\ \boldsymbol{w} &\propto S^{-1}(\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta) \end{aligned}$$

[2] C. M. Bishop, Pattern Recognition and Machine Learning, freely available here.

Linear Discriminant Analysis

The predictions of LDA then are:

$$\begin{aligned}\mathbf{x} &\mapsto \operatorname{sign}(\boldsymbol{w}^T\boldsymbol{x} - \beta)\\ \boldsymbol{w} &\propto S^{-1}(\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta)\end{aligned}$$

$$\boldsymbol{w}^T\boldsymbol{x} \propto$$

Linear Discriminant Analysis

The predictions of LDA then are:

$$\begin{aligned} \boldsymbol{x} &\mapsto \operatorname{sign}(\boldsymbol{w}^T\boldsymbol{x} - \beta) \\ \boldsymbol{w} &\propto S^{-1}(\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta) \end{aligned}$$

$$\boldsymbol{w}^T\boldsymbol{x} \propto (\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta)^T S^{-1}\boldsymbol{x} \propto$$

Linear Discriminant Analysis

The predictions of LDA then are:

$$\begin{aligned} \mathbf{x} &\mapsto \operatorname{sign}(\mathbf{w}^T\mathbf{x} - \beta) \\ \mathbf{w} &\propto S^{-1}(\bar{\mathbf{x}}_o - \bar{\mathbf{x}}_\Delta) \end{aligned}$$

$$\mathbf{w}^T\mathbf{x} \propto (\bar{\mathbf{x}}_o - \bar{\mathbf{x}}_\Delta)^T S^{-1}\mathbf{x} \propto \underbrace{(\bar{\mathbf{x}}_o - \bar{\mathbf{x}}_\Delta)^T U\Lambda^{-1/2}}_{\text{mean class difference of whitened } X} \underbrace{\Lambda^{-1/2}U^T\mathbf{x}}_{\text{whitened } x}$$

where $S = U\Lambda U^T$ is the eigenvalue decompostion of S

Linear Discriminant Analysis

The predictions of LDA then are:

$$\begin{aligned} \mathbf{x} &\mapsto \operatorname{sign}(\mathbf{w}^T\mathbf{x} - \beta) \\ \mathbf{w} &\propto S^{-1}(\bar{\mathbf{x}}_o - \bar{\mathbf{x}}_\Delta) \end{aligned}$$

$$\mathbf{w}^T\mathbf{x} \propto (\bar{\mathbf{x}}_o - \bar{\mathbf{x}}_\Delta)^T S^{-1}\mathbf{x} \propto \underbrace{(\bar{\mathbf{x}}_o - \bar{\mathbf{x}}_\Delta)^T U\Lambda^{-1/2}}_{\substack{\text{mean class difference}\\ \text{of whitened } X}} \underbrace{\Lambda^{-1/2}U^T\mathbf{x}}_{\text{whitened } \mathbf{x}}$$

where $S = U\Lambda U^T$ is the eigenvalue decompostion of S

Linear Discriminant Analysis

The predictions of LDA then are:

$$\begin{aligned} \mathbf{x} &\mapsto \operatorname{sign}(\boldsymbol{w}^T \boldsymbol{x} - \beta) \\ \boldsymbol{w} &\propto S^{-1}(\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta) \end{aligned}$$

$$\boldsymbol{w}^T \boldsymbol{x} \propto (\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta)^T S^{-1} \boldsymbol{x} \propto \underbrace{(\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta)^T U \Lambda^{-1/2}}_{\substack{\text{mean class difference} \\ \text{of whitened } X}} \underbrace{\Lambda^{-1/2} U^T \boldsymbol{x}}_{\text{whitened } \boldsymbol{x}}$$

where $S = U\Lambda U^T$ is the eigenvalue decompostion of S

Linear Discriminant Analysis

Alternative view: For centered data, LDA first whitens the data followed by nearest centroid classification:

$$\boldsymbol{w}^T\boldsymbol{x} = (\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta)^T S^{-1}\boldsymbol{x} = \underbrace{(\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta)^T U\Lambda^{-1/2}}_{\substack{\text{mean class difference}\\ \text{of whitened data}}}\underbrace{\Lambda^{-1/2}U^T\boldsymbol{x}}_{\text{whitened } \boldsymbol{x}}$$

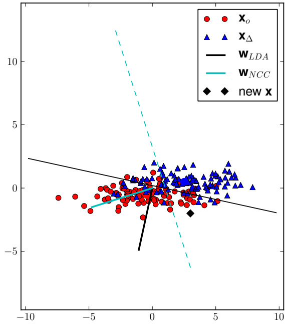
where $S = U\Lambda U^T$ is the eigenvalue decompostion of S

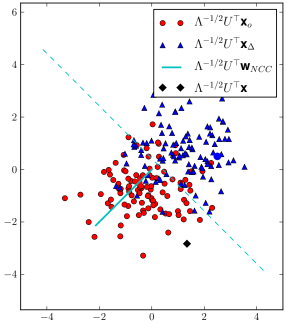
Linear Discriminant Analysis

Alternative view: For centered data, LDA first whitens the data followed by nearest centroid classification:

$$\mathbf{w}^T\mathbf{x} = (\bar{\mathbf{x}}_o - \bar{\mathbf{x}}_\Delta)^T S^{-1}\mathbf{x} = \underbrace{(\bar{\mathbf{x}}_o - \bar{\mathbf{x}}_\Delta)^T U\Lambda^{-1/2}}_{\text{mean class difference of whitened data}} \underbrace{\Lambda^{-1/2}U^T\mathbf{x}}_{\text{whitened } \mathbf{x}}$$

where $S = U\Lambda U^T$ is the eigenvalue decompostion of S



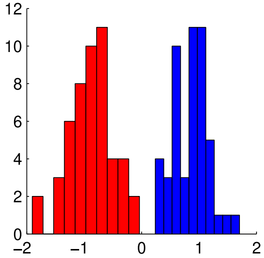


Discriminative and Generative Model

So far:

Find one-dimensional projection via $\boldsymbol{w}$ which best separates the two classes.

How can we build a discriminator, i.e. find a bias?



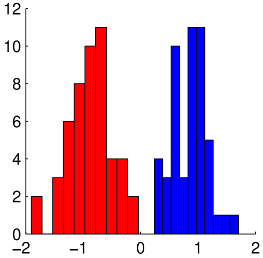
Can e.g. use center between projected means, i.e. $\beta = \frac{1}{2}(\mu_o + \mu_\Delta)$.

Discriminative and Generative Model

So far:

Find one-dimensional projection via $\boldsymbol{w}$ which best separates the two classes.

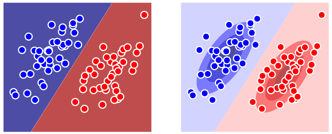
How can we build a discriminator, i.e. find a bias?



Can e.g. use center between projected means, i.e. $\beta = \frac{1}{2}(\mu_o + \mu_\Delta)$.

Generative approach:

Let's model how data was generated



Probabilistic modelling

Decision theory: The optimal classifier is Bayes classifier
For a new data point $\boldsymbol{x} \in \mathbb{R}^d$

$$\text{Decide class } \Delta \qquad \text{if } p(\Delta|\boldsymbol{x}) > p(o|\boldsymbol{x}).$$

Probabilistic modelling

Decision theory: The optimal classifier is Bayes classifier
For a new data point $\boldsymbol{x} \in \mathbb{R}^d$

$$\text{Decide class } \Delta \qquad \text{if } p(\Delta|\boldsymbol{x}) > p(o|\boldsymbol{x}).$$

Calculate $p(\Delta|\boldsymbol{x})$ with Bayes rule:

$$p(\Delta|\boldsymbol{x}) = \frac{p(\Delta)p(\boldsymbol{x}|\Delta)}{p(\boldsymbol{x})}$$

Probabilistic modelling

Decision theory: The optimal classifier is Bayes classifier
For a new data point $\boldsymbol{x} \in \mathbb{R}^d$

$$\text{Decide class } \Delta \qquad \text{if } p(\Delta|\boldsymbol{x}) > p(o|\boldsymbol{x}).$$

Calculate $p(\Delta|\boldsymbol{x})$ with Bayes rule:

$$p(\Delta|\boldsymbol{x}) = \frac{p(\Delta)p(\boldsymbol{x}|\Delta)}{p(\boldsymbol{x})}$$

For the decision, $p(\boldsymbol{x})$ is irrelevant:

$$p(\Delta|\boldsymbol{x}) > p(o|\boldsymbol{x}) \quad \Leftrightarrow \quad p(\Delta)p(\boldsymbol{x}|\Delta) > p(o)p(\boldsymbol{x}|o)\,.$$

Probabilistic modelling

The class probabilities $p(\Delta)$, $p(o)$ can be estimated using

$$p(\Delta) \approx \frac{n_\Delta}{n_\Delta + n_o} \quad \text{and similarly for } o\,.$$

Probabilistic modelling

The class probabilities $p(\Delta)$, $p(o)$ can be estimated using

$$p(\Delta) \approx \frac{n_\Delta}{n_\Delta + n_o} \quad \text{and similarly for } o\,.$$

Estimating $p(\boldsymbol{x}|\Delta)$ is difficult:

$\rightarrow$ if each dimension of $\boldsymbol{x}$ can take 2 values $\rightarrow$ 2^d possible values.

Probabilistic modelling

The class probabilities $p(\Delta)$, $p(o)$ can be estimated using

$$p(\Delta) \approx \frac{n_\Delta}{n_\Delta + n_o} \quad \text{and similarly for } o\,.$$

Estimating $p(\boldsymbol{x}|\Delta)$ is difficult:

$\rightarrow$ if each dimension of $\boldsymbol{x}$ can take 2 values $\rightarrow$ 2^d possible values.

One solution:

Choose distributions for $p(\boldsymbol{x}|\Delta)$, $p(\boldsymbol{x}|o)$ that are easy to deal with.

$\rightarrow$ Most popular: The Gaussian (or normal) distribution

$$\boldsymbol{x} \in \mathbb{R}^d \text{ in class } \Delta \;\sim \mathcal{N}(\bar{\boldsymbol{x}}_\Delta, S_\Delta) = \frac{1}{(2\pi)^{\frac{d}{2}}\sqrt{\det(S_\Delta)}} e^{-\frac{1}{2}(\boldsymbol{x}-\bar{\boldsymbol{x}}_\Delta)^\top S_\Delta^{-1}(\boldsymbol{x}-\bar{\boldsymbol{x}}_\Delta)}$$

and similarly for class o.

Linear discriminant - a probabilistic view

If we use equal covariance in each class, $\bar{S} = \frac{1}{n_\Delta + n_o}(n_\Delta S_\Delta + n_o S_o)$, the classification boundary is linear and given by

$$\boldsymbol{w} \quad = \quad \bar{S}^{-1}(\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta)$$

Linear discriminant - a probabilistic view

If we use equal covariance in each class, $\bar{S} = \frac{1}{n_\Delta + n_o}(n_\Delta S_\Delta + n_o S_o)$, the classification boundary is linear and given by

$$\begin{aligned} \boldsymbol{w} &= \bar{S}^{-1}(\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta) \\ \beta &= \frac{1}{2}\boldsymbol{w}^T(\bar{\boldsymbol{x}}_o + \bar{\boldsymbol{x}}_\Delta) + \underbrace{\log\frac{p(o)}{p(\Delta)}}_{\text{vanishes for } p(o)=p(\Delta)} \\ &= \frac{1}{2}(\mu_o + \mu_\Delta) + \log\frac{p(o)}{p(\Delta)} \end{aligned}$$

Linear discriminant - a probabilistic view

If we use equal covariance in each class, $\bar{S} = \frac{1}{n_\Delta + n_o}(n_\Delta S_\Delta + n_o S_o)$, the classification boundary is linear and given by

$$\begin{aligned}
\boldsymbol{w} &= \bar{S}^{-1}(\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta) \\
\beta &= \frac{1}{2}\boldsymbol{w}^T(\bar{\boldsymbol{x}}_o + \bar{\boldsymbol{x}}_\Delta) + \underbrace{\log\frac{p(o)}{p(\Delta)}}_{\text{vanishes for } p(o)=p(\Delta)} \\
&= \frac{1}{2}(\mu_o + \mu_\Delta) + \log\frac{p(o)}{p(\Delta)}
\end{aligned}$$

From Fisher criterion, we got

$$\boldsymbol{w} \propto S_W^{-1}(\bar{\boldsymbol{x}}_o - \bar{\boldsymbol{x}}_\Delta) \quad \text{with} \quad S_W = S_\Delta + S_o \quad \text{and (e.g.)} \quad \beta = \frac{1}{2}(\mu_o + \mu_\Delta)$$

$\Rightarrow$ Same as above if $n_\Delta = n_o$

LDA summary

Problem	Classification
Model	$y = \text{sign}(\boldsymbol{w}^T x - \beta)$
Error function	$\text{argmax}_{\boldsymbol{w}} \frac{\boldsymbol{w}^T S_B \boldsymbol{w}}{\boldsymbol{w}^T S_W \boldsymbol{w}}$
Optimization	Closed form

LDA algorithm

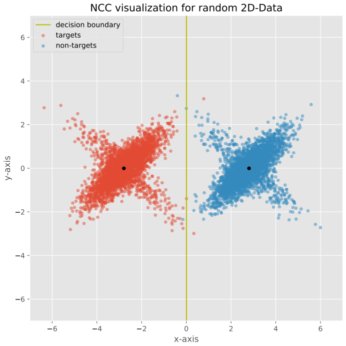
Computes: Normal vector $\boldsymbol{w}$ of decision hyperplane, threshold β

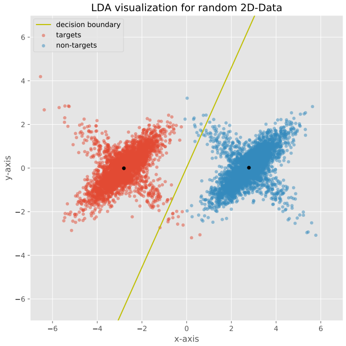
Input: Data $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}, x_i \in \mathbb{R}^d,\ y_i \in \{-1, +1\}$,

```
    Compute class mean vectors
    $\bar{\boldsymbol{x}}_- = 1/n_- \sum_{i \in \mathcal{Y}_-} \boldsymbol{x}_i$
    $\bar{\boldsymbol{x}}_+ = 1/n_+ \sum_{j \in \mathcal{Y}_+} \boldsymbol{x}_j$
    Compute averaged covariance matrix
    $\bar{S} = 1/(n_+ + n_-) \left[\sum_{i \in \mathcal{Y}_-} (\boldsymbol{x}_i - \bar{\boldsymbol{x}}_-)(\boldsymbol{x}_i - \bar{\boldsymbol{x}}_-)^\top + \sum_{j \in \mathcal{Y}_+} (\boldsymbol{x}_j - \bar{\boldsymbol{x}}_+)(\boldsymbol{x}_j - \bar{\boldsymbol{x}}_+)^\top\right]$
    Compute normal vector $\boldsymbol{w}$
    $\boldsymbol{w} = \bar{S}^{-1}(\bar{\boldsymbol{x}}_+ - \bar{\boldsymbol{x}}_-)$
    Compute threshold
    $\beta = 1/2\ \boldsymbol{w}^T(\bar{\boldsymbol{x}}_+ + \bar{\boldsymbol{x}}_-) + \log(n_-/n_+)$
```

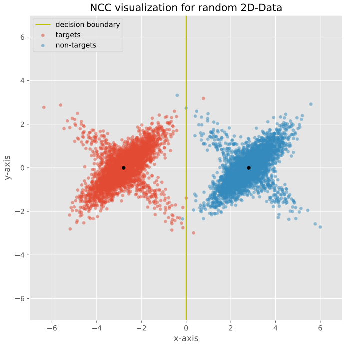
Output: $\boldsymbol{w}$, β

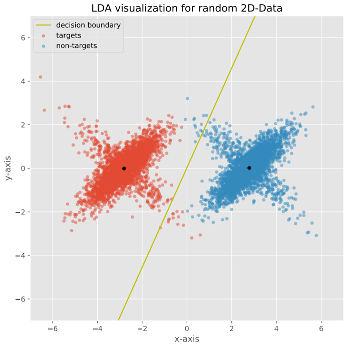
Is LDA always better than NCC?





Is LDA always better than NCC?

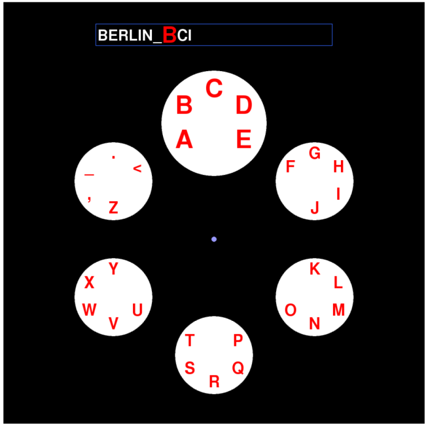


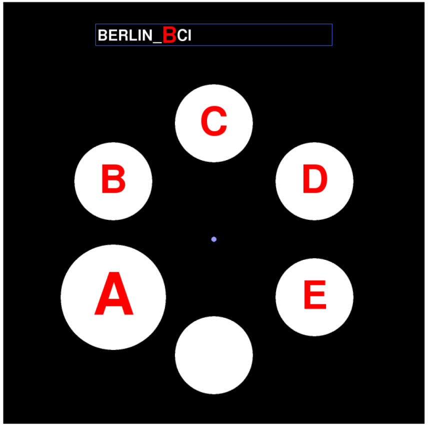


$\Rightarrow$ No, only if our assumption of equal covariance and Normal distribution for each class holds

Berlin Brain-Computer-Interface (BBCI)

Hex-o-spell: Writing with thoughts

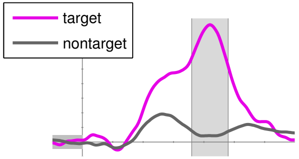


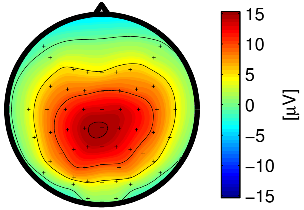


Demo: `http://iopscience.iop.org/1741-2552/8/6/066003/media`

BCI based on event-related potentials (ERPs)

- User concentrates on a symbol (the "target")
- The six circles are intensified randomly
- Intensified targets elicit ERPs that differ from non-targets
- Training data is collected and an LDA classifier is trained
- The trained classifier can now be used for spelling





This Video explains the data gathering [00:43 - 3:05]

BCI with ML: calibration and feedback

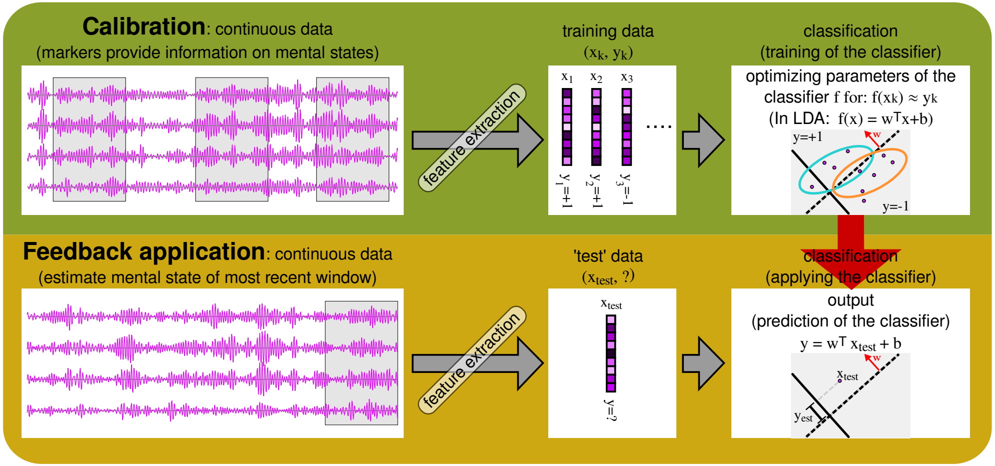

Calibration: continuous data
(markers provide information on mental states)
feature extraction
training data
(x_k, y_k)
x_1 x_2 x_3
y_1=+1 y_2=+1 y_3=-1
....
classification
(training of the classifier)
optimizing parameters of the classifier f for: $f(x_k) \approx y_k$
(In LDA: $f(x) = w^T x + b$)
y=+1
w
y=-1
Feedback application: continuous data
(estimate mental state of most recent window)
feature extraction
'test' data
$(x_{test}, ?)$
x_{test}
y=?
classification
(applying the classifier)
output
(prediction of the classifier)
$y = w^T x_{test} + b$
w
x_{test}
y_{est}

Illustration: single trials and ERPs

Illustration: single trials and ERPs

Continuous Signal (with markers)

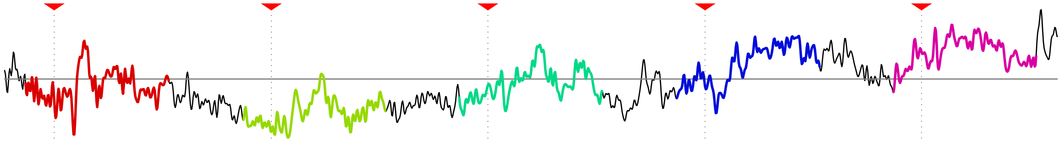
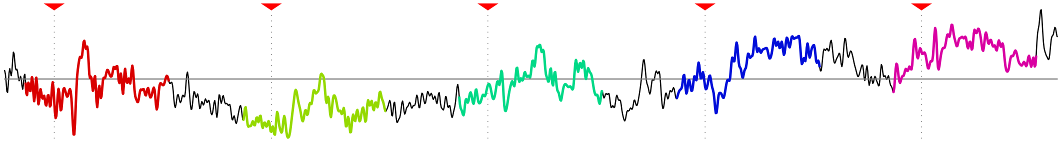
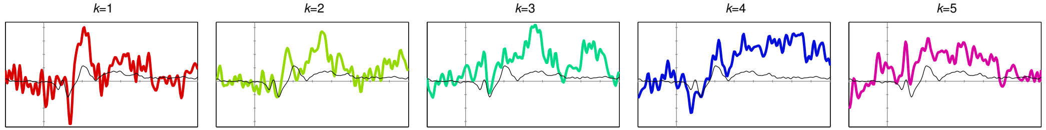


Illustration: single trials and ERPs

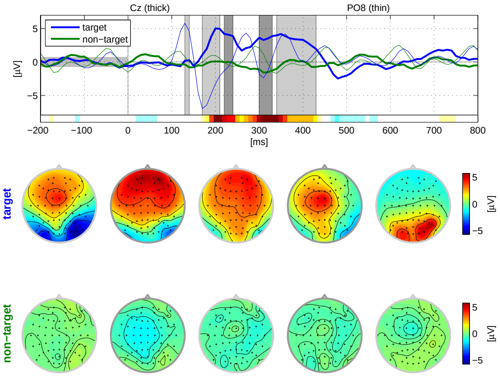
Continuous Signal (with markers)



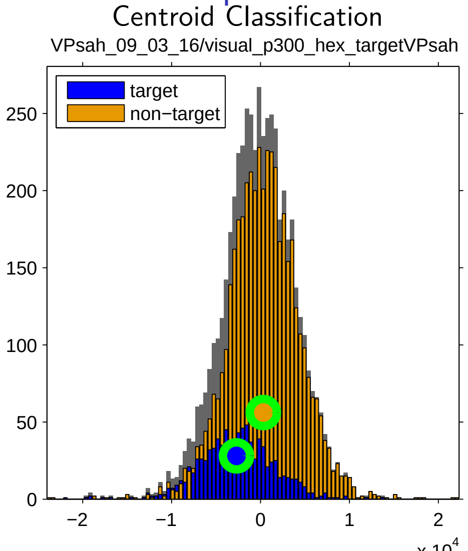
Segments (epochs) around stimulus markers



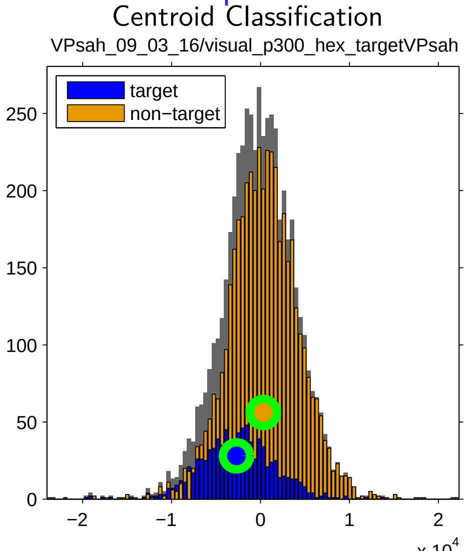
Scalp potentials in response to targets/non-targets

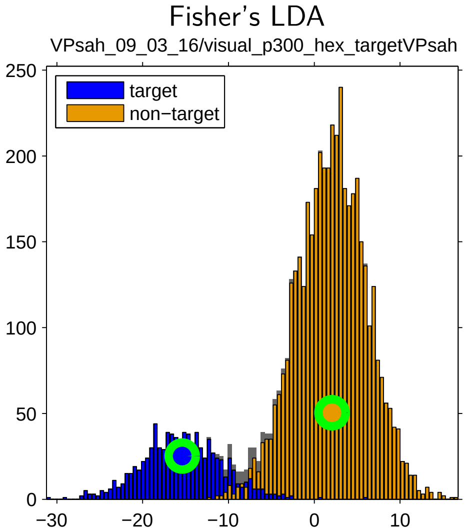


Berlin Brain-Computer-Interface



Berlin Brain-Computer-Interface





How can we properly evaluate a model?

If we use the following dataset $X \in \mathbb{R}^{d \times n}$:

$$p(\boldsymbol{x}) = \mathcal{N}(\mathbf{0}, I)$$
$$p(y = +1 | \boldsymbol{x}) = 0.5$$

with $n_{train} = 100$, $d = 300$.

How can we properly evaluate a model?

If we use the following dataset $X \in \mathbb{R}^{d \times n}$:

$$\begin{aligned} p(\mathbf{x}) &= \mathcal{N}(\mathbf{0}, I) \\ p(y = +1|\mathbf{x}) &= 0.5 \end{aligned}$$

with $n_{train} = 100$, $d = 300$.
We get the following accuracies

	Perceptron	NCC
train	100%	50%

How can we properly evaluate a model?

If we use the following dataset $X \in \mathbb{R}^{d \times n}$:

$$p(\boldsymbol{x}) = \mathcal{N}(\mathbf{0}, I)$$
$$p(y = +1|\boldsymbol{x}) = 0.5$$

with $n_{train} = 100$, $d = 300$.
We get the following accuracies

	Perceptron	NCC
train	100%	50%

- To estimate the performance of a model, let's estimate loss on samples that is has not trained on.
  - $\rightarrow$ This set of samples is called *test set*

How can we properly evaluate a model?

If we use the following dataset $X \in \mathbb{R}^{d \times n}$:

$$p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{0}, I)$$
$$p(y = +1|\boldsymbol{x}) = 0.5$$

with $n_{train} = 100$, $d = 300$.
We get the following accuracies

	Perceptron	NCC
train	100%	50%
test	50%	50%

- To estimate the performance of a model, let's estimate loss on samples that is has not trained on.
  - $\rightarrow$ This set of samples is called *test set*

How can we properly evaluate a model?

If we use the following dataset $X \in \mathbb{R}^{d \times n}$:

$$p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{0}, I)$$
$$p(y = +1 | \boldsymbol{x}) = 0.5$$

with $n_{train} = 100$, $d = 300$.
We get the following accuracies

	Perceptron	NCC
train	100%	50%
test	50%	50%

Overfitting

The production of an analysis which corresponds too closely or exactly to a particular set of data, and may therefore fail to fit additional data or predict future observations reliably.

https://www.lexico.com/definition/overfitting

Generalization and model evaluation

Generalization

Generalization is the correct categorization/prediction of new (unseen) data

How can we estimate generalization performance?

Generalization and model evaluation

Generalization

Generalization is the correct categorization/prediction of new (unseen) data

How can we estimate generalization performance?

- Train model and choose parameters on main part of data

Generalization and model evaluation

Generalization

Generalization is the correct categorization/prediction of new (unseen) data

How can we estimate generalization performance?

- Train model and choose parameters on main part of data
- Test model on other part of data, *that was not seen during training*, to estimate overall performance

Summary

Correlation. . .

- . . . is a measure of *linear relationship* between random variables
- . . . between features can affect classification accuracy

Linear Discriminant Analysis (LDA)

- LDA maximizes *between class variance* while minimizing *within class variance*
- For centered data, LDA is a NCC on whitened data
- If both classes follow a Gaussian with equal class covariances, then LDA is the optimal classifer

Model evaluation

- Only looking at performance on *training set* will give us an overly optimistic estimate of performance (*overfitting*)
- We want our model to *generalize* well