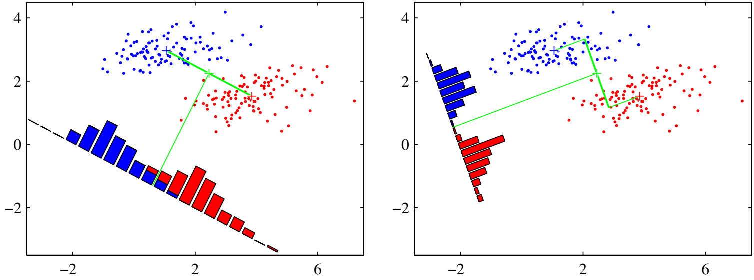
Lecture 3 & Assignment 1

Stephanie Brandl

Linear Discriminant Analysis

View classification in terms of dimensionality reduction



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

Maximizes mean class difference, and

Minimizes variance in each class

Linear Discriminant Analysis

Goal: Find a (normal vector of a linear decision boundary) $\mathbf{w}$ that

Maximizes mean class difference, $\mathbf{w}^\top S_B \mathbf{w}$ and

Minimizes variance in each class, $\mathbf{w}^\top S_W \mathbf{w}$

$\rightarrow$ maximize the *Fisher criterion*

$$\underset{\mathbf{w}}{\operatorname{argmax}} \frac{\mathbf{w}^\top S_B \mathbf{w}}{\mathbf{w}^\top S_W \mathbf{w}} \tag{1}$$

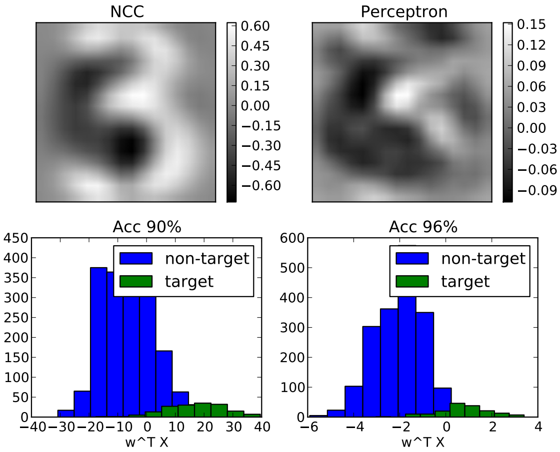
Linear Discriminant - a Probabilistic View

If we assume equal covariance in each class, $S_W = 2S_\Delta = 2S_o$, the optimal classification boundary is linear and given by

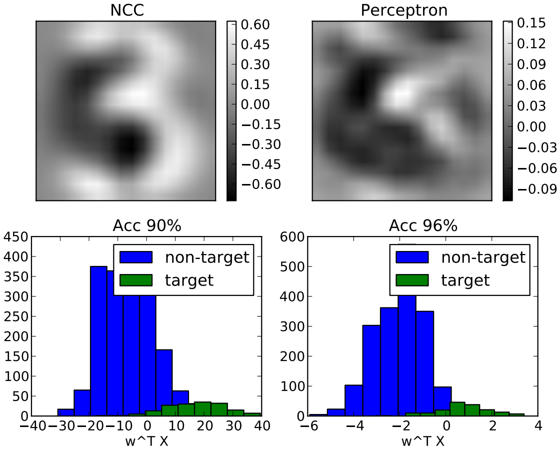
$$
\begin{aligned}
\mathbf{w} &= S_W^{-1}(\mathbf{w}_o - \mathbf{w}_\Delta) \\
\beta &= \frac{1}{2}\mathbf{w}_o S_W^{-1}\mathbf{w}_o - \frac{1}{2}\mathbf{w}_\Delta S_W^{-1}\mathbf{w}_\Delta + \log\frac{p(\Delta)}{p(o)} \\
&= \frac{1}{2}\mathbf{w}^T(\mathbf{w}_o + \mathbf{w}_\Delta) + \log\frac{p(\Delta)}{p(o)}
\end{aligned}
$$

$\Rightarrow$ Linear decision boundaries arise from simple assumption about the distribution of the data.

Assignment 2 - Hand written digit recognition



High accuracy but "lousy" weight vector - why?



High accuracy but "lousy" weight vector - why?

Misconception: signal channels (here: pixels) with large classifier weights are strongly related to the class label

High accuracy but "lousy" weight vector - why?

Misconception: signal channels (here: pixels) with large classifier weights are strongly related to the class label

Toy example 1:
Data points $\mathbf{x} \in \mathbb{R}^2$ with class labels y and two features $x_1 = y + d$ and $x_2 = d$ where d is a distractor / noise.

What is the optimal weight vector $\mathbf{w}$ such that $y = \mathbf{w}^\top \mathbf{x}$?

High accuracy but "lousy" weight vector - why?

Misconception: signal channels (here: pixels) with large classifier weights are strongly related to the class label

Toy example 1:
Data points $\mathbf{x} \in \mathbb{R}^2$ with class labels y and two features $x_1 = y + d$ and $x_2 = d$ where d is a distractor / noise.

What is the optimal weight vector $\mathbf{w}$ such that $y = \mathbf{w}^\top \mathbf{x}$?

$\rightarrow$ All class-related information is contained in x_1, but optimal weigth vector is $\mathbf{w} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$.

High accuracy but "lousy" weight vector - why?

Misconception: signal channels (here: pixels) with large classifier weights are strongly related to the class label

Toy example 1:
Data points $\mathbf{x} \in \mathbb{R}^2$ with class labels y and two features $x_1 = y + d$ and $x_2 = d$ where d is a distractor / noise.

What is the optimal weight vector $\mathbf{w}$ such that $y = \mathbf{w}^\top \mathbf{x}$?

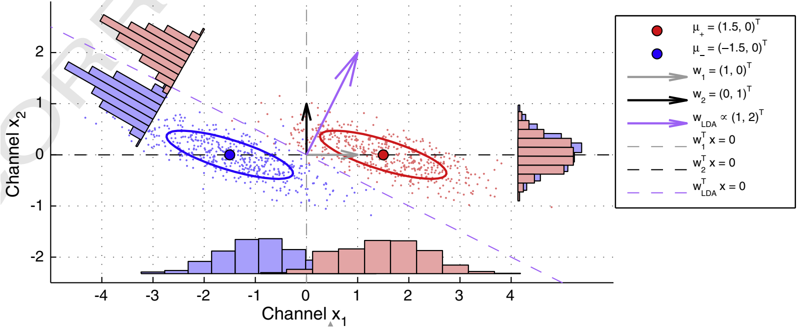
$\rightarrow$ All class-related information is contained in x_1, but optimal weigth vector is $\mathbf{w} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$.

The purpose of the weight vector is two-fold: amplify the signal of interest, while at the same time surpress signals of no interest.

High accuracy but "lousy" weight vector - why?

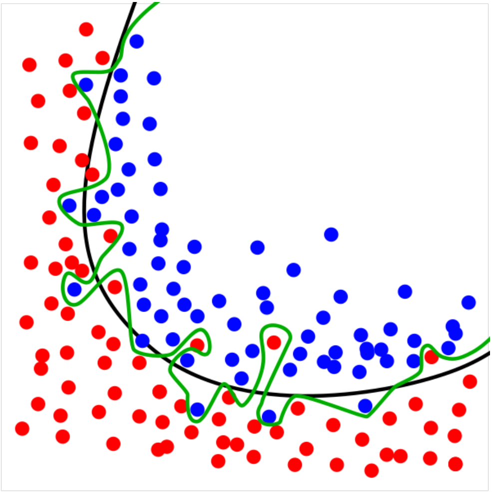
The purpose of the weight vector is two-fold: amplify the signal of interest, while at the same time surpress signals of no interest.

Toy example 2: Remember LDA



(see e.g. S. Haufe et al, "On the interpretation of weight vectors of linear models in multivariate neuroimaging", Neuroimage, 2013)

Why we need to test on new data - Overfitting



Generalization and Model Evaluation

The goal of classification is **generalization**: Correct categorization/prediction of new data

How can we estimate generalization performance?

Generalization and Model Evaluation

The goal of classification is **generalization**: Correct categorization/prediction of new data

How can we estimate generalization performance?

$\rightarrow$ **Cross-validation**:

- Train model on part of data
- Test model on other part of data
- Repeat on different cross-validation *folds*
- Average performance on test set across all folds

Cross-Validation

Algorithm 1: Cross-Validation

Require: Data $(x_1, y_1) \ldots, (x_N, y_N)$, Number of CV folds F

\# Split data in F **disjunct** folds
for folds $f = 1, \ldots, F$ **do**
    \# Train model on folds $\{1, \ldots, F\} \setminus f$
    \# Compute prediction error on fold f
end for
\# Average prediction error