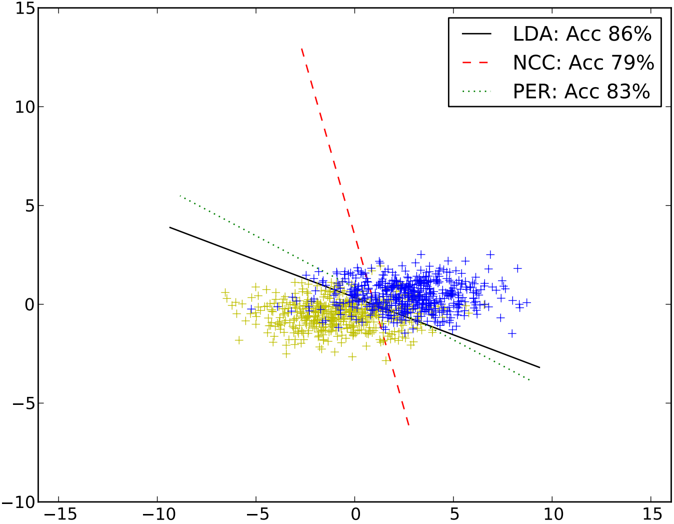
# Assignment 3 & Lecture 4

Stephanie Brandl

# Assignment 3 - Toy data



# Perceptron Algorithm with Stochastic Gradient Descent

**Computes:** Normal vector $\mathbf{w}$ of decision hyperplane for binary classification

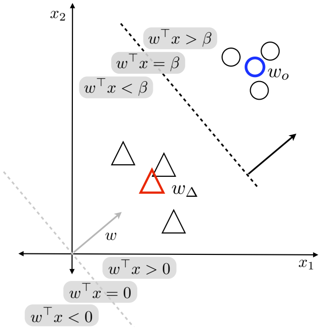
**Input:** Data $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}, \mathbf{x}_i \in \mathbb{R}^D,\ y_i \in \{-1, +1\}$,
learning rate $\eta$,
iterations $N_{it}$

**Algorithm:**

$\mathbf{w} = \mathbb{1}/D$
**for** $i = 1$ **to** $N_{it}$ **do**
  Pick a random data point $\mathbf{x}_i$
  **if** $\mathbf{w}^\top \mathbf{x}_i \cdot y_i < 0$ **then**
    $\mathbf{w} = \mathbf{w} + \eta/i \cdot \mathbf{x}_i \cdot \mathbf{y}_i$
  **end if**
**end for**

**Output:** $\mathbf{w}$

# Nearest Centroid Classifier (NCC)




Comparison of distance to class means is equivalent to linear classification

$$\begin{aligned} \|\mathbf{x} - \mathbf{w}_\Delta\| &< \|\mathbf{x} - \mathbf{w}_o\| \\ \Leftrightarrow 0 &< \mathbf{w}^\top \mathbf{x} - \beta \end{aligned}$$

where

$$\mathbf{w} = \mathbf{w}_o - \mathbf{w}_\Delta$$

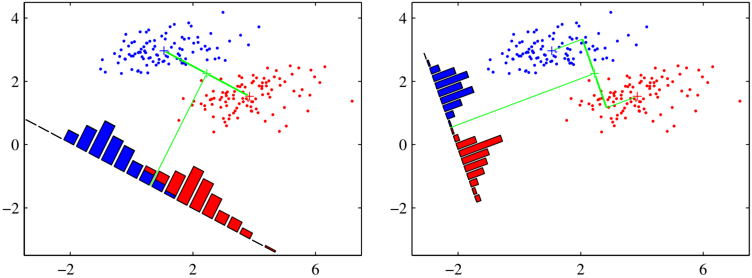
and

$$\begin{aligned} \beta &= 1/2(\mathbf{w}_o^\top \mathbf{w}_o - \mathbf{w}_\Delta^\top \mathbf{w}_\Delta) \\ &= 1/2\mathbf{w}(\mathbf{w}_o + \mathbf{w}_\Delta) \end{aligned}$$

This simple linear classification rule is often called **Nearest Centroid Classifier**.

# Linear Discriminant Analysis (LDA)

View classification in terms of dimensionality reduction




**Goal:** Find a (normal vector of a linear decision boundary) **w** that

Maximizes mean class difference, and

Minimizes variance in each class

# LDA vs. NCC

LDA:

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

NCC:

$$
\begin{aligned}
\mathbf{w}_{NCC} &= (\mathbf{w}_o - \mathbf{w}_\Delta) \\
\beta_{NCC} &= \frac{1}{2}\mathbf{w}_{NCC}^T(\mathbf{w}_o + \mathbf{w}_\Delta)
\end{aligned}
$$

$\Rightarrow$ same result if

# LDA vs. NCC

LDA:

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

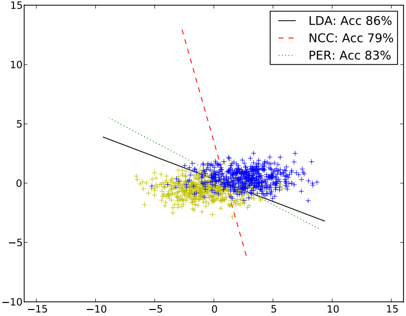
NCC:

$$
\begin{aligned}
\mathbf{w}_{NCC} &= (\mathbf{w}_o - \mathbf{w}_\Delta) \\
\beta_{NCC} &= \frac{1}{2}\mathbf{w}_{NCC}^T(\mathbf{w}_o + \mathbf{w}_\Delta)
\end{aligned}
$$

$\Rightarrow$ same result if

$$
S_W = \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}
$$

# LDA vs. NCC



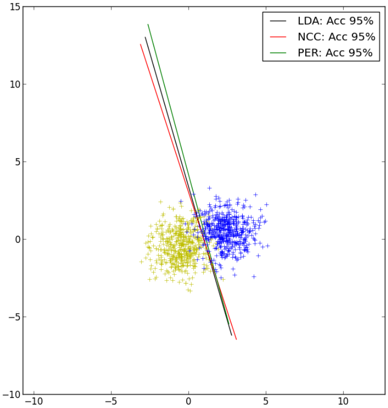

$$S_o = S_\Delta = \begin{bmatrix} 5 & 0 \\ 0 & 0.5 \end{bmatrix}$$

- Each class has the same covariance matrix
- In each class, $x$ and $y$ are uncorrelated
- In each class, $x$ and $y$ have different variance, $\text{Var}(x) = 5$ and $\text{Var}(y) = 0.5$

# LDA vs. NCC

Uncorrelated data with the same variance in each dimension.

$$S_o = S_\Delta = \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

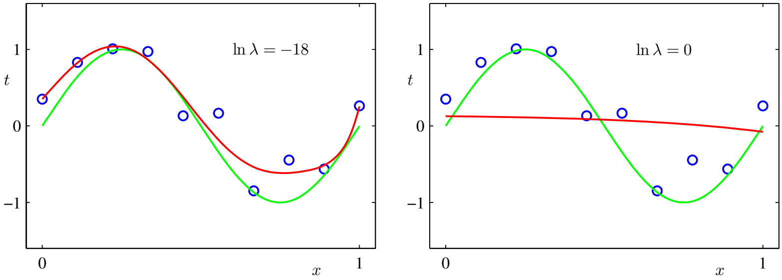


# Ridge Regression

Often it is important to **control the complexity** the solution of **w**.

This is done by constraining the norm of **w**,

$$\mathcal{E}_{RR}(\mathbf{w}) = ||y - \mathbf{w}^\top X||^2 + \lambda ||\mathbf{w}||^2$$



## Example: Polynomial Curve Fitting

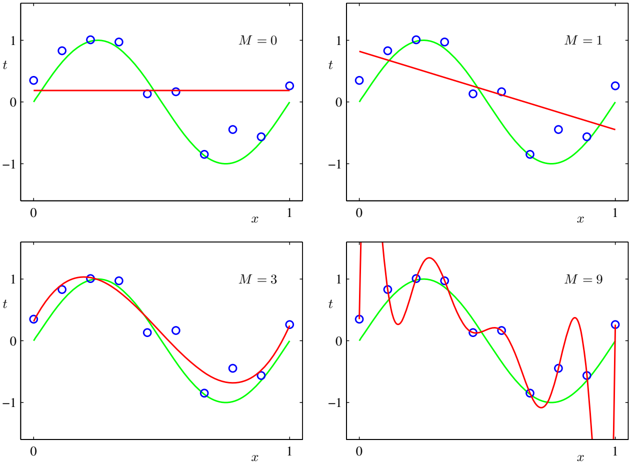
Toy data generated by $\sin(2\pi x)$ including random noise.
Traning set with $N$ samples:

$$\mathbf{x} = (x_1, \ldots, x_N)^\top \text{data points}$$
$$\mathbf{y} = (y_1, \ldots, y_N)^\top \text{observations}$$

We try to discover the underlying function $\sin(2\pi x)$ using a polynomial function

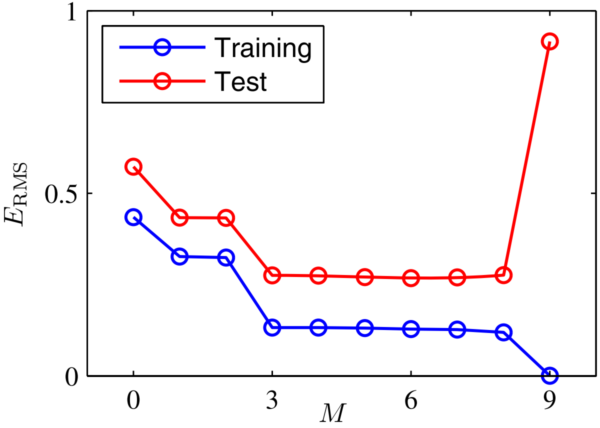
$$y(x, w) = w_0 + w_1 x \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j = w^\top X$$

# Example: Polynomial Curve Fitting




$$y(x, w) = w_0 + w_1 x \ldots + w_M x^M$$

# Example: Polynomial Curve Fitting

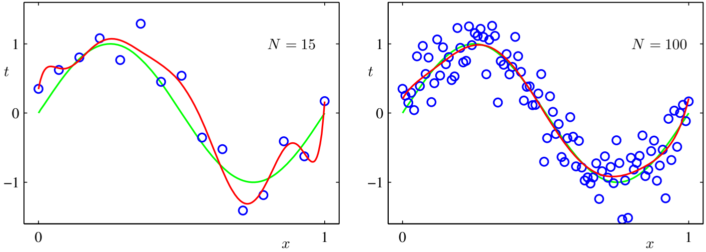


# Example: Polynomial Curve Fitting

| | $M=0$ | $M=1$ | $M=6$ | $M=9$ |
|---|---|---|---|---|
| $w_0^\star$ | 0.19 | 0.82 | 0.31 | 0.35 |
| $w_1^\star$ | | -1.27 | 7.99 | 232.37 |
| $w_2^\star$ | | | -25.43 | -5321.83 |
| $w_3^\star$ | | | 17.37 | 48568.31 |
| $w_4^\star$ | | | | -231639.30 |
| $w_5^\star$ | | | | 640042.26 |
| $w_6^\star$ | | | | -1061800.52 |
| $w_7^\star$ | | | | 1042400.18 |
| $w_8^\star$ | | | | -557682.99 |
| $w_9^\star$ | | | | 125201.43 |

# Example: Polynomial Curve Fitting

Increasing the size of the data set reduces the overfitting problem.



# Example: Polynomial Curve Fitting

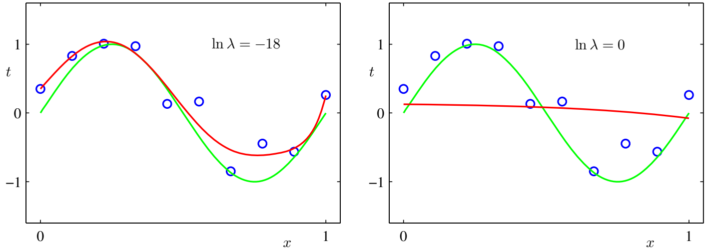
Instead of choosing $M$ beforehand, wen can control the complexitiy by regularization:

$$\mathcal{E}_{RR}(\mathbf{w}) = ||y - \mathbf{w}^\top X||^2 + \lambda||\mathbf{w}||^2$$

When adding the regularization term $\lambda||\mathbf{w}||^2$ to the error function, we penalize large values for $w$ and thus can control its complexity.

# Example: Polynomial Curve Fitting

Applying Ridge Regression



# Example: Polynomial Curve Fitting

| | $\ln \lambda = -\infty$ | $\ln \lambda = -18$ | $\ln \lambda = 0$ |
|---|---:|---:|---:|
| $w_0^\star$ | 0.35 | 0.35 | 0.13 |
| $w_1^\star$ | 232.37 | 4.74 | -0.05 |
| $w_2^\star$ | -5321.83 | -0.77 | -0.06 |
| $w_3^\star$ | 48568.31 | -31.97 | -0.05 |
| $w_4^\star$ | -231639.30 | -3.89 | -0.03 |
| $w_5^\star$ | 640042.26 | 55.28 | -0.02 |
| $w_6^\star$ | -1061800.52 | 41.32 | -0.01 |
| $w_7^\star$ | 1042400.18 | -45.95 | -0.00 |
| $w_8^\star$ | -557682.99 | -91.53 | 0.00 |
| $w_9^\star$ | 125201.43 | 72.68 | 0.01 |