4.1. Support Vector Machines

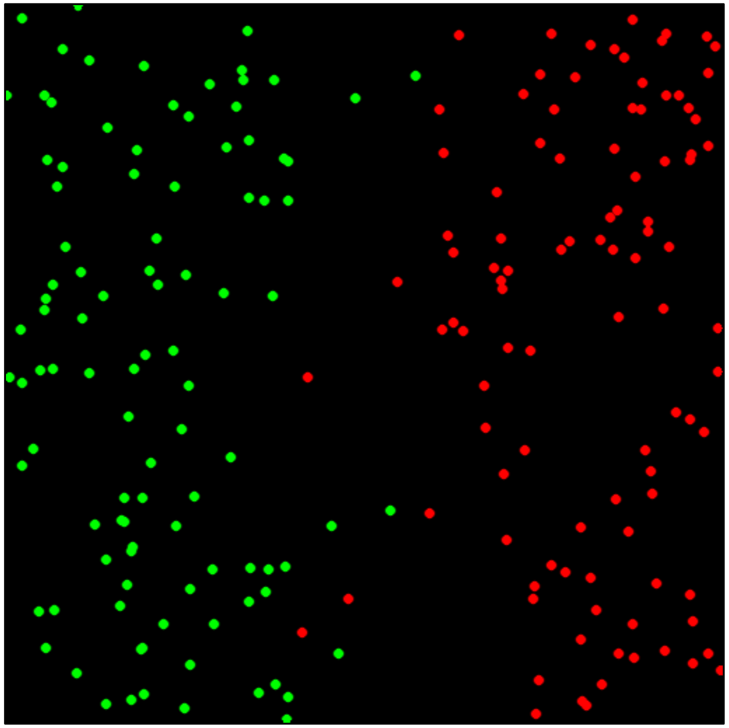
Multimedia Project SS 2016

Multimedia Computing and Computer Vision Lab

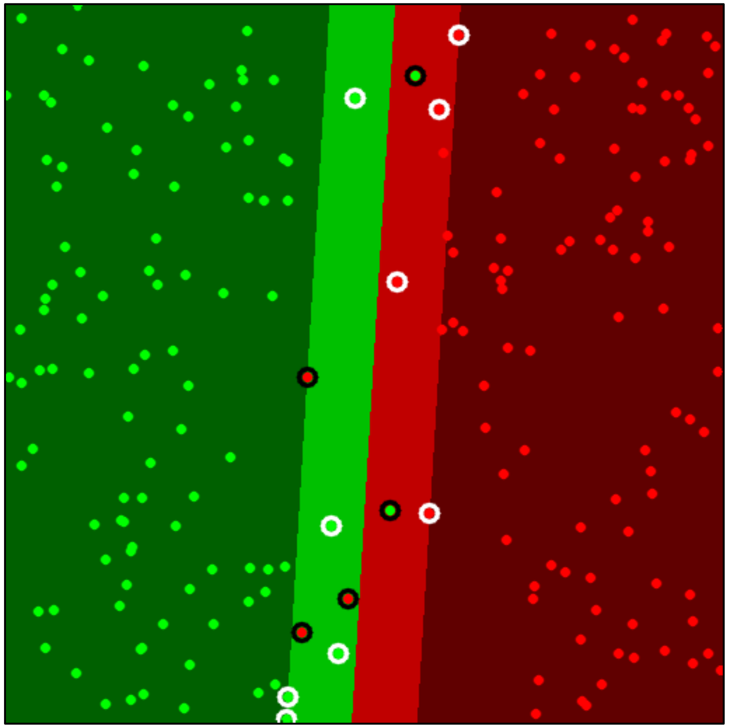
Anton Winschel

Support Vector Machines (SVM)

- Purpose: Binary classification with linear kernel
- Sparse solutions: Predictions only need the kernel function evaluated at a subset of training data points



2 class training data



maximum margin classification

Anton Winschel, Multimedia Computing and Computer Vision Lab, University of Augsburg
Universitätsstr. 6a, D-86159 Augsburg, Germany; email: anton.winschel@informatik.uni-augsburg.de

Part I: Hyperplane Classifiers

Maximum Margin Classifier

- Two class classification problem using linear decision model

$$y(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b$$

$$y : \mathbb{R}^M \rightarrow \mathbb{R}$$

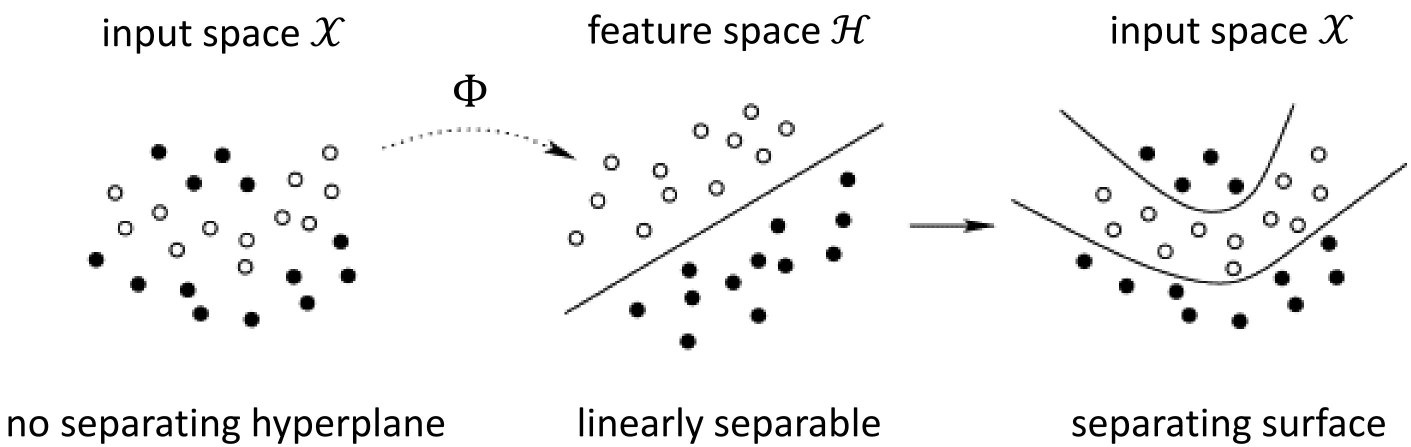
- Training data set
  - Examples $(\boldsymbol{x_1}, t_1), \dots, (\boldsymbol{x_N}, t_N) \in \mathbb{R}^M \times \{\pm 1\}$
  - Data points are represented by *pattern* $\boldsymbol{x_i}$ (→attributes), and target value (or *label)* t_i (→class)
  - Finding hypothesis that *optimally* splits data points according to their classes
  - If there are multiple solutions, try to find the one that gives the smallest generalization error
  - Convex optimization problem

Maximum Margin Classifier

- Classification
  - New data point $\boldsymbol{x}$ is classified according to the sign of $y(\boldsymbol{x})$
  - Only requires a small subset of training points – all other points can be discarded

Non-Linear Kernels

- In case of data that is not linearly separable, non-linear kernel functions can be used to map data points into higher-dimensional feature space
- Potentially better classification results but: (i) higher computational cost at test time, (ii) longer training time and (iii) potential overfitting of model to data
- Linear separation in feature space $\mathcal{H}$ gives non-linear decision surface in input space $\mathcal{X}$:



Anton Winschel, Multimedia Computing and Computer Vision Lab, University of Augsburg
Universitätsstr. 6a, D-86159 Augsburg, Germany; email: anton.winschel@informatik.uni-augsburg.de

Kernel Trick

- Kernel trick: Replace dot product by non-linear kernel (e.g. Gaussian)
  - Requires algorithm that depends on data only through dot products
  - By replacing the dot product with a kernel function, data points from the input domain are mapped into (possibly very high-dimensional) feature space
- Generates non-linear decision boundary in the input space
- Prediction: Linear combination of kernel functions

$$y(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i \, t_i k(\boldsymbol{x}, \boldsymbol{x_i}) + b \text{, with} \quad k(\boldsymbol{x}, \boldsymbol{x}') = \Phi(\boldsymbol{x})^T \Phi(\boldsymbol{x}')$$

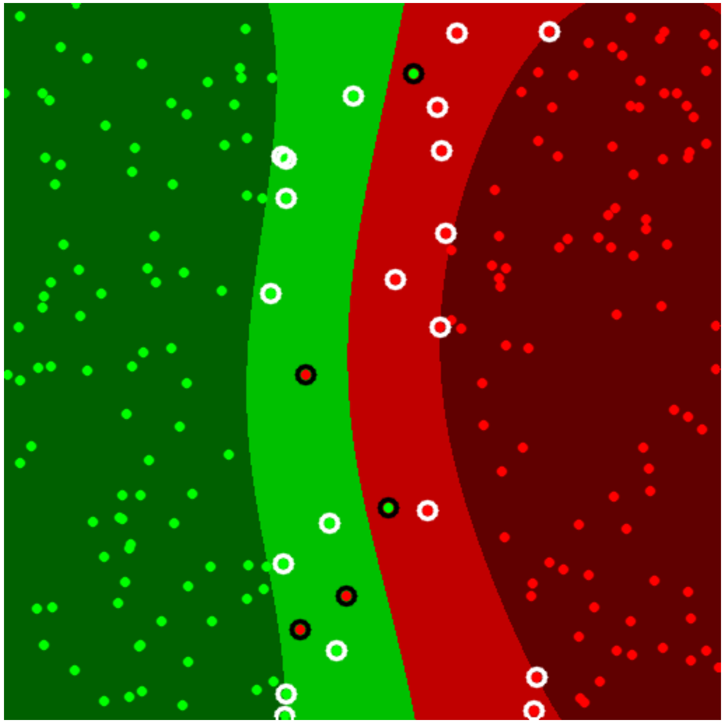
- Using identity mapping $\Phi(\boldsymbol{x}) \coloneqq \boldsymbol{x}$ the kernel is a simple dot product:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \langle \boldsymbol{x}, \boldsymbol{x}' \rangle$$

- For linear models, computing $y(x)$ is very efficient

Example: Non-Linear Decision Surfaces

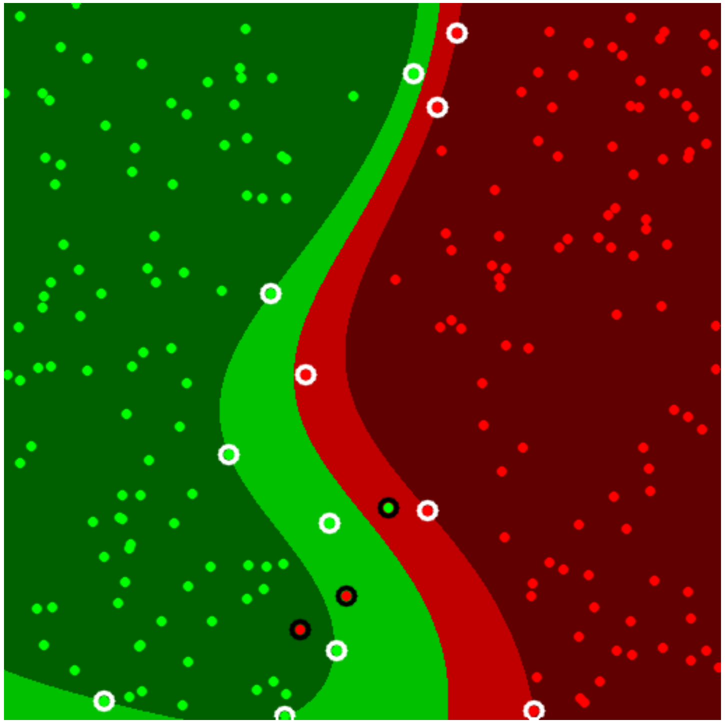
- RBF kernel (non-linear)



$C = 1$

Example: Non-Linear Decision Surfaces

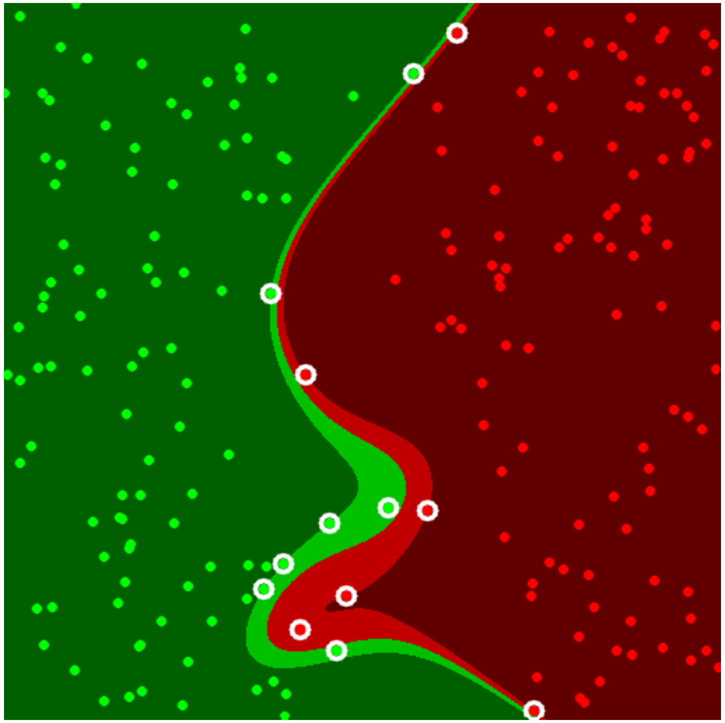
- RBF kernel (non-linear)



$C = 100$

Example: Non-Linear Decision Surfaces

- RBF kernel (non-linear)



$C = 10000$

Part II: Hard-Margin Formulation

Hyperplane Classifiers

- Hyperplane learning algorithm
  - A *hyperplane* in dot product space $\mathcal{H}$ is defined by

$$\{\boldsymbol{x} \in \mathcal{H} | \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b = 0\}, \text{ where } \boldsymbol{w} \in \mathcal{H}, b \in \mathbb{R}$$

  - A *linear decision function* corresponding to $\boldsymbol{w}$ is given by

$$y(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b$$

  - Assumption: N linearly separable training examples $\boldsymbol{x_1}, \dots, \boldsymbol{x_N}$, with target values $t_1, \dots, t_N$
    - There exists at least one $\boldsymbol{w}$ and b s.t. $y(\boldsymbol{x_i}) > 0$ for all points having $t_i = +1$, and $y(\boldsymbol{x_i}) < 0$ for those having $t_i = -1$
    - If hyperplane separates data perfectly, it holds $t_i y(\boldsymbol{x_i}) > 0$ for all training examples
  - Classification according to $\operatorname{sign}\big(y(\boldsymbol{x})\big)$

Anton Winschel, Multimedia Computing and Computer Vision Lab, University of Augsburg
Universitätsstr. 6a, D-86159 Augsburg, Germany; email: anton.winschel@informatik.uni-augsburg.de