Support Vector Machines

Tobias Pohlen

Selected Topics in Human Language Technology and Pattern Recognition
February 10, 2014

Human Language Technology and Pattern Recognition
Lehrstuhl für Informatik 6
Computer Science Department
RWTH Aachen University, Germany

Outline

Literature

N. Christiani, J. Shawe-Taylor An introduction to support vector machines, Cambridge University Press, 2000.

- Indepth introduction to SVMs (theoretical and practical concepts)

V. N. Vapnik The nature of statistical learning theory, Springer, 1995

- Theoretical background of SVMs

C. J. C. Burges A Tutorial on Support Vector Machines for Pattern Recognition. Data Mining and Knowledge Discovery 2, 1998, pages 121-167

- Good and short introduction to SVMs (Only 40 pages)

Classification Problem

We want to solve the *binary classification problem*

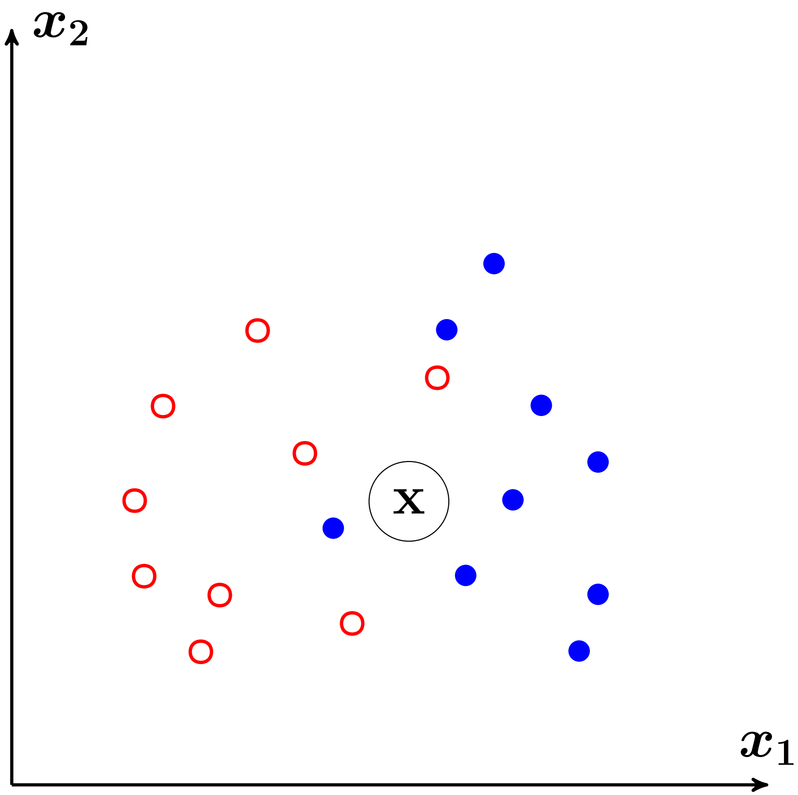
- ***Training set*** $X = \{(\mathrm{x}_1, y_1), ..., (\mathrm{x}_N, y_N)\} \subset \mathbb{R}^m \times \{-1, 1\}$
  - ***Input space*** $\mathbb{R}^m$
  - ***training examples*** x_i **and**
  - ***class labels*** y_i

Goal:

- **Assign the vector x to one of the classes -1 or 1**

We will consider the multiclass problem later.

Classification Problem - Visualization



Linear discriminant functions and linear classifiers

Definition 1. Linear discriminant function *defined as*

$$\boldsymbol{f} : \mathbb{R}^m \mapsto \mathbb{R}, \mathbf{x} \mapsto \mathbf{w}^T \mathbf{x} + \boldsymbol{b} = \sum_{i=1}^{m} \boldsymbol{w}_i \boldsymbol{x}_i + \boldsymbol{b} \tag{1}$$

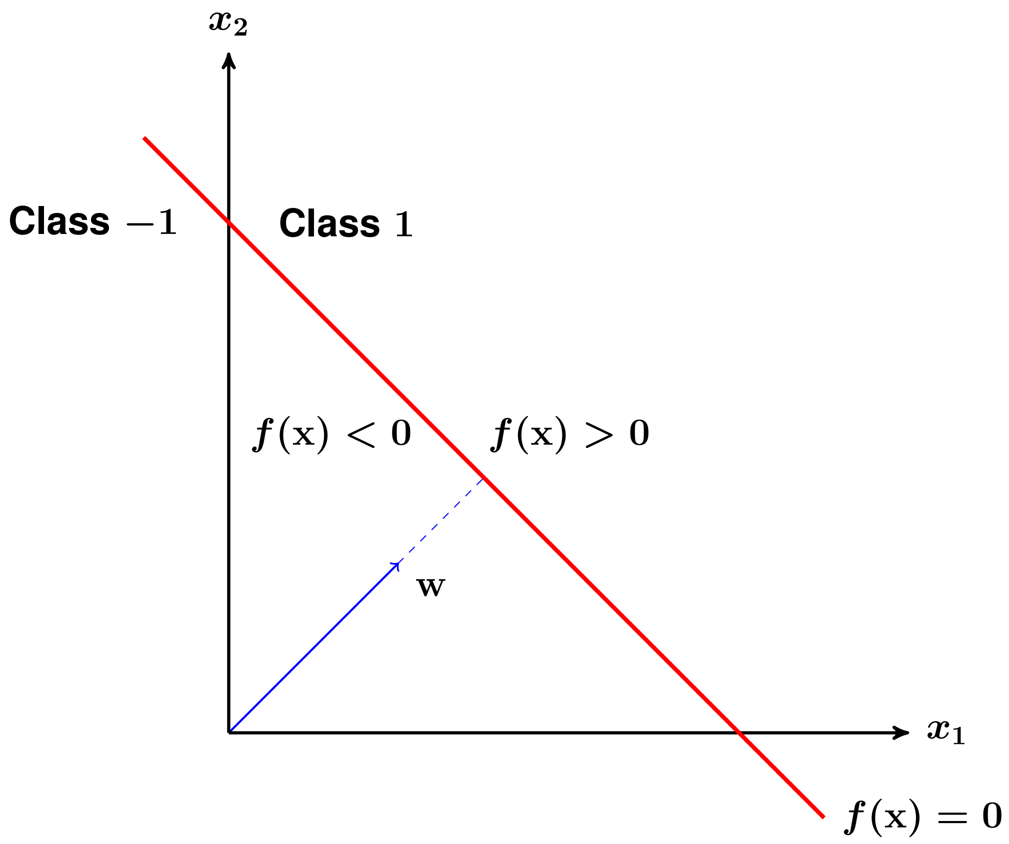
- $\mathbf{w} \in \mathbb{R}^m$ *is called* weight vector
- $\boldsymbol{b} \in \mathbb{R}$ *is called* bias

Definition 2. Linear classifier *defined as*

$$\boldsymbol{h_f} : \mathbb{R}^m \mapsto \{-1, 1\}, \quad \mathbf{x} \mapsto \boldsymbol{sign}(\boldsymbol{f}(\mathbf{x})) \quad = \begin{cases} 1 & \textit{if } \boldsymbol{f}(\mathbf{x}) \geq 0 \\ -1 & \textit{otherwise} \end{cases} \tag{2}$$

Decision surface and decision regions

Definition 3. *Decision surface of a linear classifier* h_f *is defined by:* $f(\mathrm{x}) = 0$



Decision surface of a linear classifier (red) is an $(m-1)$-dimensional hyperplane.

Linear separability

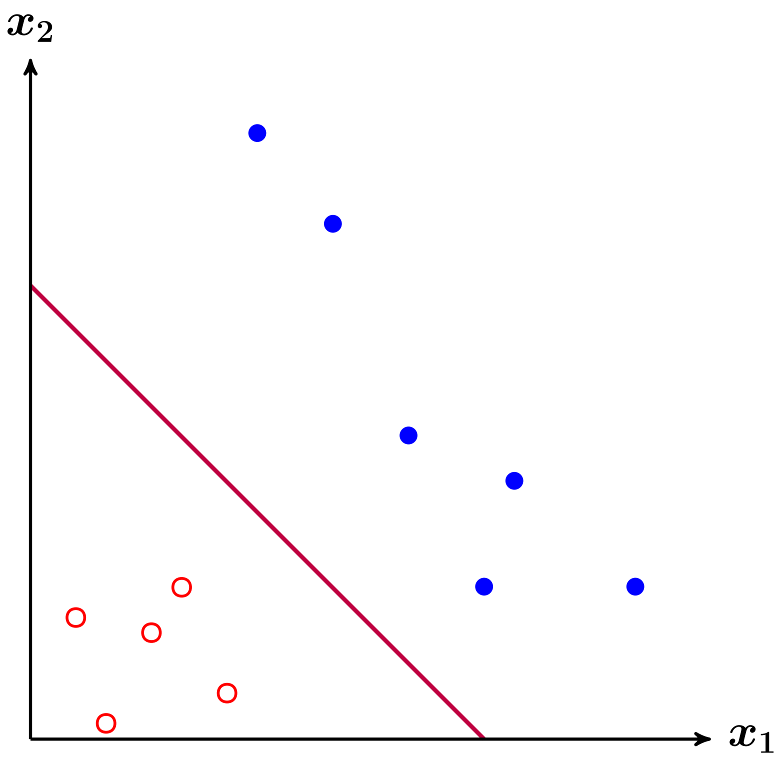
Definition 4. $X = \{(\mathrm{x}_1, y_1), ..., (\mathrm{x}_N, y_N)\}$ *is called* linearly separable *if*

$$\exists h_f : h_f(\mathrm{x}_i) = y_i, \forall i = 1, ..., N \quad (3)$$

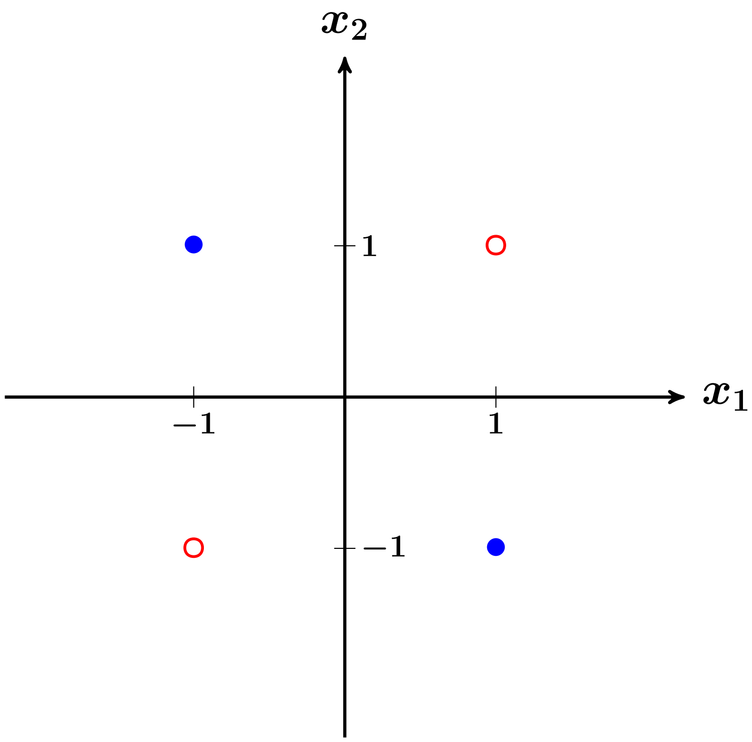
(i.e. there exists a classifier h_f that classifies all points correctly).

Linear separability - Visualization

Linearly separable training set



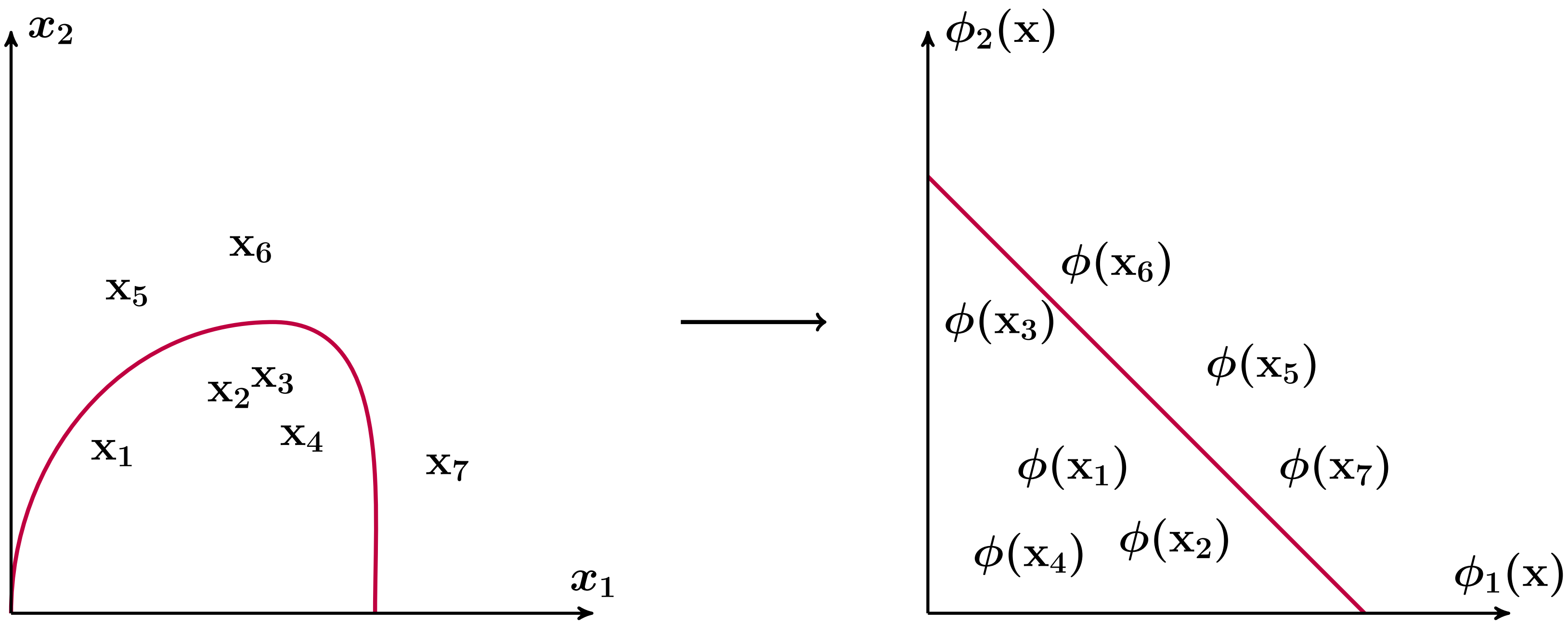
Linearly inseparable training set



Linear classifiers and inseparable training sets

How can we use linear classifiers for linearly inseparable training sets?

- **Idea: Map the data into another space in which it is linearly separable**



Feature Space Mappings

Definition 5. *Let $\mathcal{H}$ be a D-dimensional Hilbert space, $D \in \mathbb{N} \cup \{\infty\}$.*
A feature space mapping *is defined as*

$$\phi : \mathbb{R}^m \mapsto \mathcal{H}, \mathbf{x} \mapsto (\phi_i(\mathbf{x}))_{i=1}^{D} \tag{4}$$

- ϕ_i *are called* basis functions
- $\mathcal{H}$ *is called* feature space

Linear discriminant function with feature space mapping ϕ

$$f(\mathbf{x}) = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle + b, \qquad \mathbf{w} \in \mathcal{H} \tag{5}$$

$\langle \cdot, \cdot \rangle$ is the inner product of $\mathcal{H}$

A Hilbert space is a vector space with inner product $\langle \cdot, \cdot \rangle$.

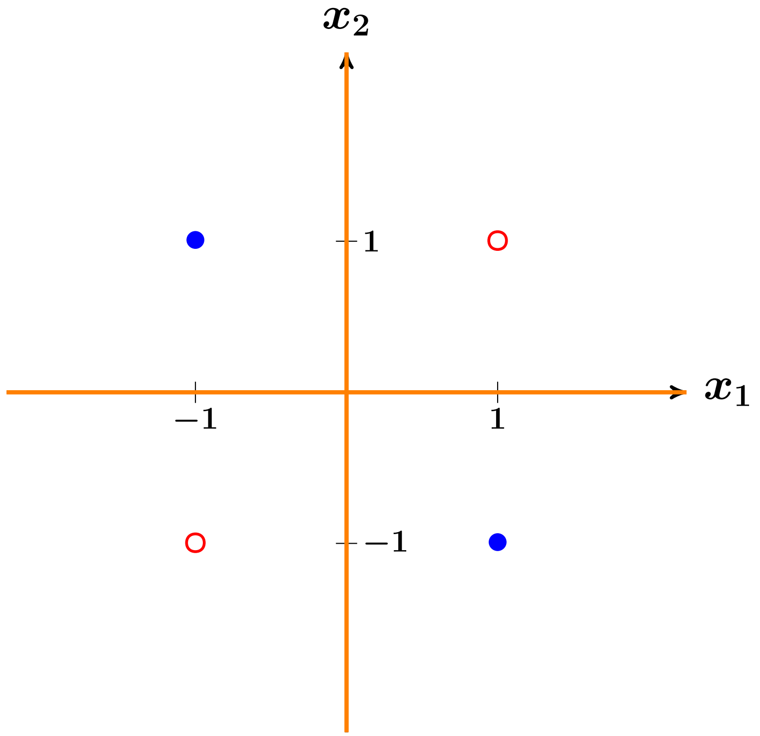
Separate the XOR Example

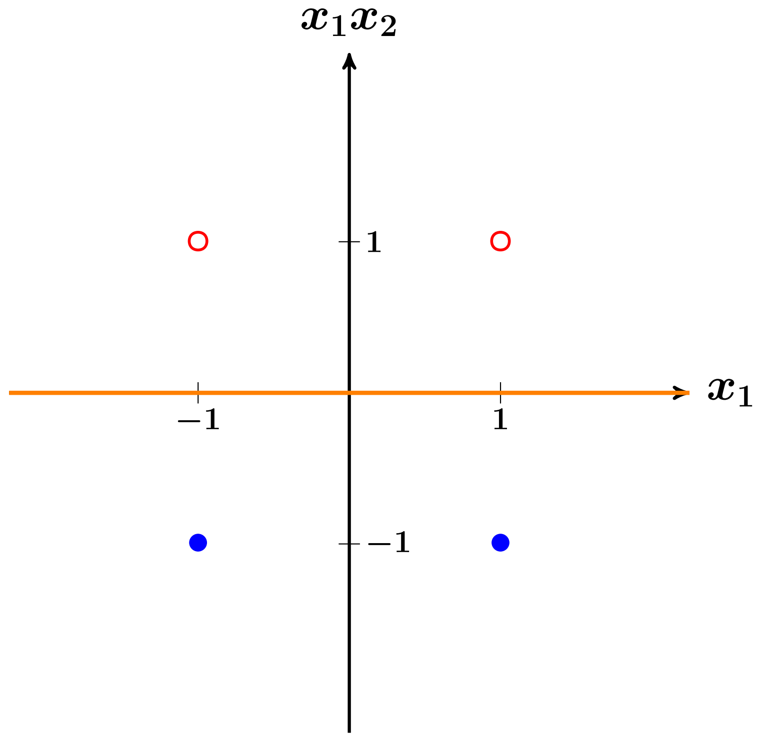
Let $\mathcal{H} \equiv \mathbb{R}^2$ with $\langle \mathrm{x}, \mathrm{y} \rangle \equiv \mathrm{x}^T \mathrm{y} = \mathrm{x} \cdot \mathrm{y}$ (dot product)

$$\phi : \mathbb{R}^2 \mapsto \mathcal{H}, \mathrm{x} \mapsto \begin{pmatrix} x_1 \\ x_1 x_2 \end{pmatrix} \tag{6}$$

$$f(\mathrm{x}) = (0 \;\; 1)^T \phi(\mathrm{x}) + 0 = x_1 x_2 \tag{7}$$

Decision surface in the input space | **Decision surface in the feature space**





Feature Space Mappings - Complexity

- **Evaluation of $f : \mathbb{R}^m \mapsto \mathbb{R}$ without feature space mapping**

$$f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b, \mathbf{w} \in \mathbb{R}^m \tag{8}$$

Complexity: $\mathcal{O}(m)$

- **Evaluation of f with feature space mapping**

$$f(\mathbf{x}) = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle + b, \mathbf{w} \in \mathcal{H} \tag{9}$$

Complexity: $\mathcal{O}(dim(\mathcal{H}))$
Typically $dim(\mathcal{H}) \gg m$**.**

Reducing complexity

Idea: The inner product $\langle \phi(\mathrm{x}), \phi(\mathrm{y}) \rangle$ can often be computed very efficiently.

Simple example:

$$\phi(\mathrm{x}) = \begin{pmatrix} x_1^2 \\ \sqrt{2} x_1 x_2 \\ x_2^2 \end{pmatrix} \tag{10}$$

Let $\mathrm{x}, \mathrm{y} \in \mathbb{R}^2$ arbitrary. Then

$$\begin{aligned} \langle \phi(\mathrm{x}), \phi(\mathrm{y}) \rangle &= \phi(\mathrm{x})^T \phi(\mathrm{y}) && (11) \\ &= x_1^2 y_1^2 + 2 x_1 x_2 y_1 y_2 + x_2^2 y_2^2 && (12) \\ &= (\mathrm{x}^T \mathrm{y})^2 && (13) \end{aligned}$$

- **Naive computation: 11 multiplications, 2 additions**
- **Optimized computation: 3 multiplications, 2 additions**

Kernel Functions

Definition 6. *Let* $\phi : \mathbb{R}^m \mapsto \mathcal{H}$ *be a feature space mapping.*

$$\boldsymbol{k}(\mathrm{x}, \mathrm{y}) = \langle \phi(\mathrm{x}), \phi(\mathrm{y}) \rangle \tag{14}$$

is called kernel function *or* kernel*.*

Interpretation: A kernel measures similarity.

Dual form

In order to use kernels, we need to formulate our algorithms in the so called *dual form*.

- $\phi(\mathrm{x})$ **occurs only as argument of an inner product** $\langle \cdot, \cdot \rangle$

Definition 7. *Let* $X = \{(\mathrm{x}_1, y_1), ..., (\mathrm{x}_N, y_N)\}$ *be a training set.*

- *Linear discriminant function* f *in* dual form*:*

$$f(\mathrm{x}) = \sum_{i=1}^{N} \theta_i k(\mathrm{x}_i, \mathrm{x}) + b, \qquad \theta_i \in \mathbb{R}, i = 1, ..., N \tag{15}$$

(16)

We will see later how to determine θ_i**.**

Kernel Functions - Complexity

- **Evaluation of $f : \mathbb{R}^m \mapsto \mathbb{R}$ without feature space mapping**

$$f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b, \mathbf{w} \in \mathbb{R}^m \quad (17)$$

  Complexity: $\mathcal{O}(m)$

- **Evaluation of f with feature space mapping**

$$f(\mathbf{x}) = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle + b, \mathbf{w} \in \mathcal{H} \quad (18)$$

  Complexity: $\mathcal{O}(dim(\mathcal{H}))$

- **Evaluation of f in the dual form**

$$f(\mathbf{x}) = \sum_{i=1}^{N} \theta_i k(\mathbf{x}, \mathbf{x}_i) + b, \theta_i \in \mathbb{R} \quad (19)$$

  Complexity: $\mathcal{O}(N)$
  (Assuming $k(\cdot, \cdot)$ can be evaluated efficiently)

Popular kernel functions

There are many known kernels. Popular examples:

- **Polynomial kernel**

$$k(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T \mathbf{y} + c)^d, \qquad c \in \mathbb{R}, d \in \mathbb{N}_0 \tag{20}$$

- **Gaussian kernel**

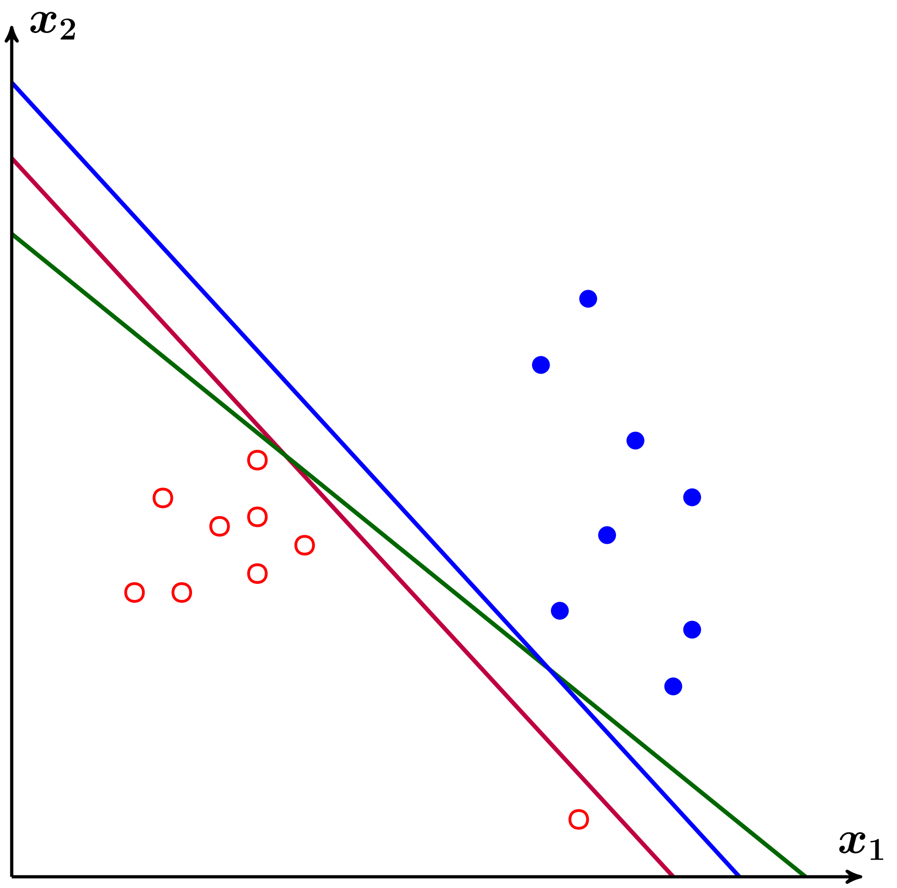
$$k(\mathbf{x}, \mathbf{y}) = \exp\left(\frac{-\|\mathbf{x} - \mathbf{y}\|_2^2}{\sigma^2}\right), \qquad \sigma \in \mathbb{R}_+ \tag{21}$$

Both kernels are widely used in practice.

SVMs - Motivation

Support Vector Machines (SVMs) **arose from the theoretical question:**

- **Given a training set. What is the *optimal* linear classifier?**

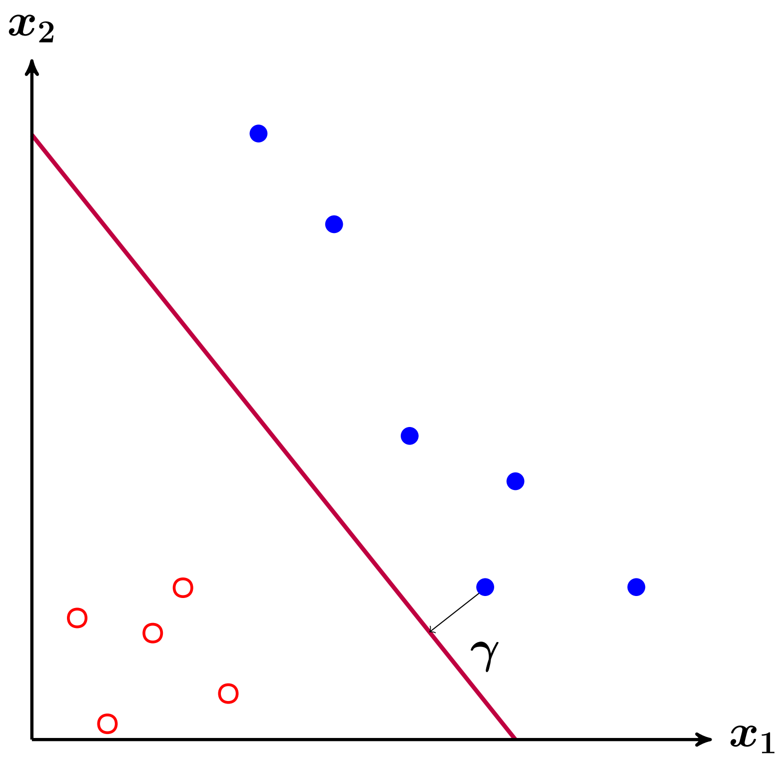


Concept of margins

Definition 8. ▶ h_f *linear classifier*

▶ X *training set*

▶ (geometric) margin γ *of* h_f *on* X *is the smallest distance from a point to the decision surface*



Expected generalization error

- **Vapnik answered the question using *statistical learning theory***
  - **Idea: The optimal classifier has the lowest expected error on *unknown* data**
- **Data (points and labels) is generated i.i.d. according to a fixed but unknown distribution $\mathcal{D}$**

Upper bound on the generalization error

$$\sum_{i \in \{-1,1\}} \int_{\mathbb{R}^m} \frac{1}{2} |1 - y h_f(\mathrm{x})| p(\mathrm{x}, y) dx \leq \epsilon(h_f, X) \tag{22}$$

Maximum margin classifiers

Given a linearly separable training set.
What is the *optimal* linear classifier in terms of the upper bound ϵ?

Result:

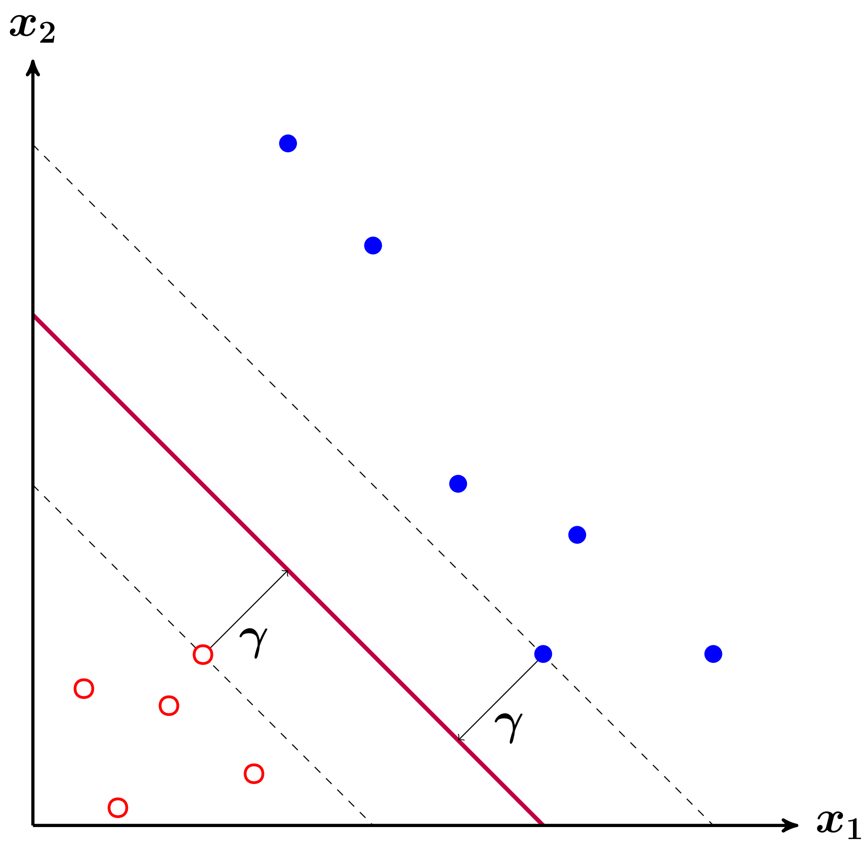
- **The linear classifier with the largest margin γ on the training set.**
  - **Such classifiers are called *maximum margin classifiers***

See

- **[Vapnik 95] for an introduction to statistical learning theory**
- **[Boser & Guyon$^+$ 92] and [Vapnik 82] for more theoretical background on SVMs**
- **[Christiani & Shawe-Taylor 00] for a good overview**
- **[Shawe-Taylor & Bartlett$^+$ 98] and [Bartlett & Shawe–Taylor 99] for Data Dependent Structural Risk Minimization**

Maximum margin visualized

Decision surface of a maximum margin classifier:



Computing margins

Lemma 1. *Margin of h_f on X can be computed by*

$$\gamma = \min_{i=1,\dots,N} \frac{y_i f(\mathrm{x}_i)}{\|w\|_2} \tag{23}$$

- $\frac{y_i f(\mathrm{x}_i)}{\|w\|_2}$ **is the euclidean distance from the point x_i to the decision surface**

SVM Learning

Assume the training set $X = \{(\mathrm{x}_1, y_1), ..., (\mathrm{x}_N, y_N)\}$ is linearly separable.

- **How can we determine w, b such that the margin is maximal?**

$$\max_{\mathrm{w},b} \gamma = \max_{\mathrm{w},b} \min_{i=1,...,N} \frac{y_i f(\mathrm{x}_i)}{\|w\|_2} \tag{24}$$

Observation:

- **If we scale w and b, the decision surface does not move:**

$$\lambda f(\mathrm{x}) = 0 \Leftrightarrow f(\mathrm{x}) = 0, \qquad \text{for } \lambda > 0 \tag{25}$$

  - (w, b) **and** $(\lambda \mathrm{w}, \lambda b)$ **induce the same decision surface**

Idea:

- **Scale w, b such that**

$$y_i f(\mathrm{x}_i) = 1 \tag{26}$$

for the training examples x_i that have the smallest distance to the decision surface

Optimization problem (primal form)

We can maximize γ by minimizing $\|w\|_2$:

$$\max_{\mathrm{w},b} \gamma = \max_{\mathrm{w},b} \min_{i=1,\ldots,N} \frac{y_i f(\mathrm{x}_i)}{\|w\|_2} = \max_{\mathrm{w},b} \frac{1}{\|w\|_2} \tag{27}$$

$$= \frac{1}{\min_{\mathrm{w},b} \|w\|_2} \tag{28}$$

Resulting optimization problem in the linearly separable case (primal form):

$$\min_{\mathrm{w}\in\mathbb{R}^m, b\in\mathbb{R}} \quad \frac{1}{2}\|\mathrm{w}\|_2^2 \tag{29}$$

$$\textbf{subject to} \quad y_i f(\mathrm{x}_i) \geq 1, \qquad i = 1, \ldots, N \tag{30}$$

Optimization theory: Lagrange multipliers

Introduce *Langrange function* and *Lagrange multipliers*.

Optimization problem:

$$\min_{\mathbf{x}\in\mathbb{R}^m} f(\mathbf{x}) \text{ subject to } \begin{cases} c_i(\mathbf{x}) = 0, i \in \mathcal{E} \\ c_i(\mathbf{x}) \geq 0, i \in \mathcal{I} \end{cases} \tag{31}$$

Lagrange function:

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) - \sum_{i\in\mathcal{E}\cup\mathcal{I}} \lambda_i c_i(\mathbf{x}) \tag{32}$$

- $\boldsymbol{\lambda} = (\lambda_1, ..., \lambda_l)$ **are called *Lagrange mutlipliers***

Minimizing f subject to the constraints is equivalent to

- **Minimizing L w.r.t. $\mathbf{x}$**
- **Maximizing L w.r.t. $\boldsymbol{\lambda}$**

Optimization theory: KKT conditions

First order necessary conditions

If x^* is local minimum of f (respecting the constraints), then there exists λ^* such that

$$\nabla_x \mathcal{L}(\mathrm{x}^*, \boldsymbol{\lambda}^*) = 0 \quad (33)$$
$$c_i(\mathrm{x}^*) = 0, \quad \forall i \in \mathcal{E} \quad (34)$$
$$c_i(\mathrm{x}^*) \geq 0, \quad \forall i \in \mathcal{I} \quad (35)$$
$$\lambda_i^* \geq 0, \quad \forall i \in \mathcal{I} \quad (36)$$
$$\lambda_i^* c_i(\mathrm{x}^*) = 0, \quad \forall i \in \mathcal{E} \cup \mathcal{I} \quad (37)$$

- **All these conditions are called *Karush-Kuhn-Tucker conditions (KKT conditions)***
- **The last conditions are called *complementary conditions***

See [Nocedal & Wrigt 06] for an in-depth introduction.

Lagrangian and KKT conditions

SVM training problem: $X = \{(x_1, y_1), ..., (\mathrm{x}_N, y_N)\}$ **is linearly separable**

$$\min_{\mathrm{w}\in\mathbb{R}^m, b\in\mathbb{R}} \quad \frac{1}{2}\|\mathrm{w}\|_2^2 \tag{38}$$

$$\text{subject to} \quad y_i(\langle \mathrm{w}, \phi(\mathrm{x}_i)\rangle + b) - 1 \geq 0, \qquad i = 1, ..., N \tag{39}$$

Lagrange function:

$$\mathcal{L}(\mathrm{w}, b, \alpha) = \frac{1}{2}\|w\|_2^2 - \sum_{i=1}^{N} \alpha_i(y_i(\langle \mathrm{w}, \phi(\mathrm{x}_i)\rangle + b) - 1) \tag{40}$$

KKT conditions:

$$\frac{\partial}{\partial \mathrm{w}}\mathcal{L}(\mathrm{w}, b, \alpha) = \mathrm{w} - \sum_{i=1}^{N} y_i\alpha_i\phi(\mathrm{x}_i) \stackrel{!}{=} 0 \Rightarrow \mathrm{w} = \sum_{i=1}^{N} y_i\alpha_i\phi(\mathrm{x}_i) \tag{41}$$

$$\frac{\partial}{\partial b}\mathcal{L}(\mathrm{w}, b, \alpha) = \sum_{i=1}^{N} y_i\alpha_i \stackrel{!}{=} 0 \tag{42}$$

Finding the dual (part 1)

KKT condition (part 1)

$$\frac{\partial}{\partial \mathrm{w}}\mathcal{L}(\mathrm{w}, b, \alpha) = \mathrm{w} - \sum_{i=1}^{N} y_i \alpha_i \phi(\mathrm{x}_i) \stackrel{!}{=} 0 \Rightarrow \mathrm{w} = \sum_{i=1}^{N} y_i \alpha_i \phi(\mathrm{x}_i) \tag{43}$$

Substituting this condition back into the Langrange function:

$$\mathcal{L}(\mathrm{w}, b, \alpha) = \frac{1}{2}\|w\|_2^2 - \sum_{i=1}^{N} \alpha_i (y_i(\langle \mathrm{w}, \phi(\mathrm{x}_i)\rangle + b) - 1) \tag{44}$$

$$= \frac{1}{2}\left\langle \sum_{i=1}^{N} y_i \alpha_i \phi(\mathrm{x}_i), \sum_{i=1}^{N} y_i \alpha_i \phi(\mathrm{x}_i) \right\rangle - \sum_{i=1}^{N} \alpha_i \left(y_i \left(\left\langle \sum_{i=j}^{N} y_j \alpha_j \phi(\mathrm{x}_j), \phi(\mathrm{x}_i) \right\rangle + b \right) - 1 \right) \tag{45}$$

$$= \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle \phi(\mathrm{x}_i), \phi(\mathrm{x}_j)\rangle - \sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle \phi(\mathrm{x}_i), \phi(\mathrm{x}_j)\rangle - b\sum_{i=1}^{N} \alpha_i y_i + \sum_{i=1}^{N} \alpha_i \tag{46}$$

$$= \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle \phi(\mathrm{x}_i), \phi(\mathrm{x}_j)\rangle - b\sum_{i=1}^{N} \alpha_i y_i \tag{47}$$

Finding the dual (part 2)

KKT condition (part 2)

$$\frac{\partial}{\partial b}\mathcal{L}(\mathbf{w}, b, \alpha) = \sum_{i=1}^{N} y_i \alpha_i \stackrel{!}{=} 0 \tag{48}$$

Substituting this condition back into the Langrange function:

$$\mathcal{L}(\mathbf{w}, b, \alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j)\rangle - b \underbrace{\sum_{i=1}^{N} \alpha_i y_i}_{=0} \tag{49}$$

$$= \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \underbrace{\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j)\rangle}_{=k(\mathbf{x}_i, \mathbf{x}_j)} \tag{50}$$

Optimization problem (dual form)

KKT-conditions for inequality constraints:

$$\lambda_i^* \geq 0, \quad \forall i \in \mathcal{I} \tag{51}$$

Optimization problem in the dual form: $X = \{(x_1, y_1), ..., (\mathrm{x}_N, y_N)\}$ **is linearly separable**

$$\max_{\alpha \in \mathbb{R}^N} \quad \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j k(\mathrm{x}_i, \mathrm{x}_j) \tag{52}$$

$$\text{subject to} \quad \sum_{i=1}^{N} y_i \alpha_i = 0 \tag{53}$$

$$\alpha_i \geq 0, i = 1, ..., N \tag{54}$$

- **This is a *quadratic programming problem***
  - **Quadratic objective function, Linear constraints**
  - **Convex problem $\Rightarrow$ Unique solution**
- **Can be solved using standard solvers**
  - **Complexity $\mathcal{O}(N^3)$, $N = \#$ training examples**

SVM classification

How do we classify new data points?

- **Need to determine θ_i for the dual form $f(\mathbf{x}) = \sum_{i=1}^{N} \theta_i k(\mathbf{x}_i, \mathbf{x})$**

KKT conditions yield:

$$\frac{\partial}{\partial \mathbf{w}} \mathcal{L}(\mathbf{w}, b, \alpha) = \mathbf{w} - \sum_{i=1}^{N} y_i \alpha_i \phi(\mathbf{x}_i) \stackrel{!}{=} 0 \tag{55}$$

$$\Leftrightarrow \mathbf{w} = \sum_{i=1}^{N} y_i \alpha_i \phi(\mathbf{x}_i) \tag{56}$$

Insert into linear discriminant function (primal form):

$$f(\mathbf{x}) = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle + b \tag{57}$$

$$= \left\langle \sum_{i=1}^{N} y_i \alpha_i \phi(\mathbf{x}_i), \phi(\mathbf{x}) \right\rangle + b \tag{58}$$

$$= \sum_{i=1}^{N} \underbrace{y_i \alpha_i}_{\theta_i} k(\mathbf{x}_i, \mathbf{x}) + b \tag{59}$$

Sparsity

Why is it called a *sparse* kernel machine?

- **Recall: Evaluating $f(\mathrm{x})$ in the dual form costs $\mathcal{O}(N)$**

$$f(\mathrm{x}) = \sum_{i=1}^{N} y_i \alpha_i k(\mathrm{x}, \mathrm{x}_i) + b, \alpha_i \in \mathbb{R} \tag{60}$$

Optimization problem (primal form):

$$\min_{\mathrm{w} \in \mathbb{R}^m, b \in \mathbb{R}} \quad \frac{1}{2} \|\mathrm{w}\|_2^2 \tag{61}$$

$$\text{subject to} \quad y_i f(\mathrm{x}_i) - 1 \geq 0, \qquad i = 1, ..., N \tag{62}$$

Complementary conditions:

$$\alpha_i (y_i f(\mathrm{x}_i) - 1) = 0, \quad \forall i = 1, ..., N \tag{63}$$

Meaning

- **Either $\alpha_i = 0$**
- **Or $y_i f(\mathrm{x}_i) = 1$**

Support vectors

$$\alpha_i(y_i f(\mathrm{x}_i) - 1) = 0, \quad \forall i = 1, ..., N \tag{64}$$
$$\Rightarrow \alpha_i = 0 \vee y_i f(\mathrm{x}_i) = 1 \tag{65}$$

$\Rightarrow$ **only the Lagrange multipliers of the points closest to the hyperplane are non-zero.**

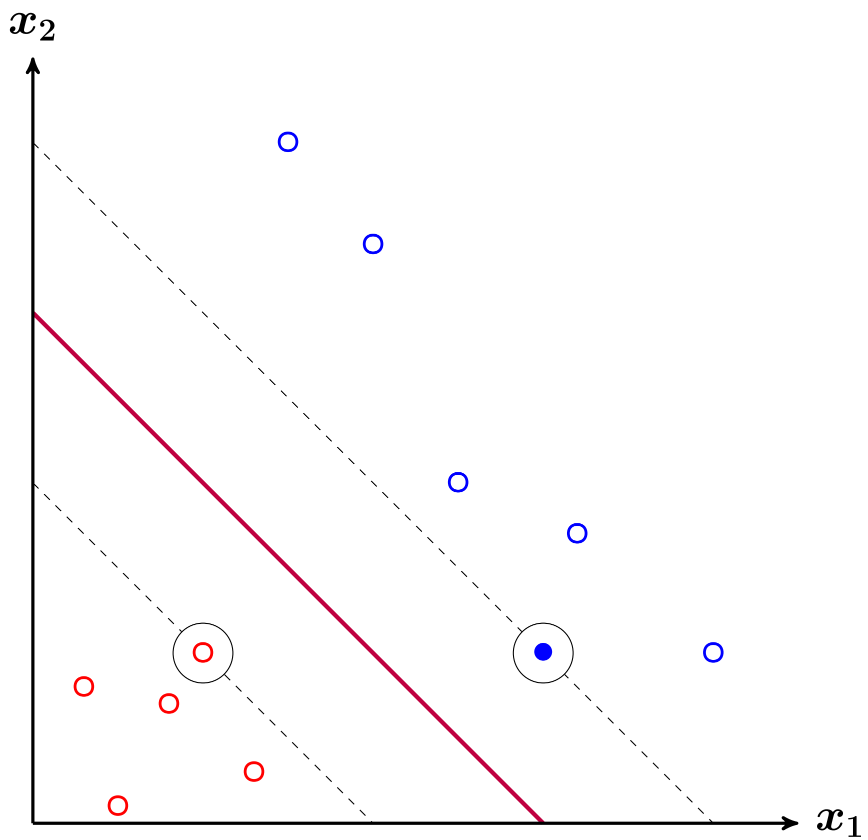
- **We call those points *support vectors***
- **Set of support vector indices is denoted by $\mathcal{SV}$**
- **Evaluating f in the dual form is linear in the number of support vectors**

$$f(\mathrm{x}) = \sum_{i=1}^{N} y_i \alpha_i k(\mathrm{x}, \mathrm{x}_i) + b = \sum_{i \in \mathcal{SV}} y_i \alpha_i k(\mathrm{x}, \mathrm{x}_i) + b \tag{66}$$

- **Typically:**

$$N \gg |\mathcal{SV}| \tag{67}$$

Support vectors visualized



- **Support vectors are circled**
- **Dashed lines are called *margin boundaries***

Support vectors - Implications

- **Only the support vectors influence the decision surface**
- **The support vectors are the points *hardest to classify***
  - **Give insight into the classification problem**
- **After learning, only the support vectors and their respective weights have to be saved**
  - **Modelsize is small compared to the size of the training set**

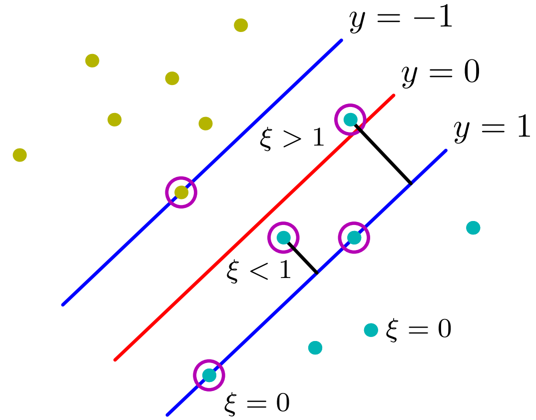
Remaining questions

- What do we do if X is *not* linearly separable?
- What do we do if we have more than two classes?

Learning inseparable case

X is *not* linearly separable.

- **Introduce a penalty for points that violate the maximum margin constraint**
- **Penalty increases linearly in the distance from the respective boundary**
  - **Introduce so called *slack variables* ξ_i**



Optimization problem (primal form):

$$\min_{\mathbf{w}\in\mathbb{R}^m, b\in\mathbb{R}, \boldsymbol{\xi}\in\mathbb{R}^N} \quad \frac{1}{2}\|\mathbf{w}\|_2^2 + C\|\boldsymbol{\xi}\|_1 \tag{68}$$

$$\text{subject to} \quad y_i(\langle \mathbf{w}, \phi(\mathbf{x}_i)\rangle + b) - 1 + \xi_i \geq 0, i = 1, ..., N \tag{69}$$

$$\xi_i \geq 0, i = 1, ..., N \tag{70}$$

Imagesource: C. M. Bishop http://research.microsoft.com/en-us/um/people/cmbishop/prml/webfigs.htm

Learning inseparable case - Dual form

The dual form can be found in the same manner as in the linearly separable case.
Result:

$$\max_{\alpha \in \mathbb{R}^N} \quad \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j k(\mathrm{x}_i, \mathrm{x}_j) \tag{71}$$

$$\text{subject to} \quad \sum_{i=1}^{N} y_i \alpha_i = 0 \tag{72}$$

$$0 \leq \alpha_i \leq C, i = 1, ..., N \tag{73}$$

- C **is a tradeoff parameter that determines how strongly a point is punished**
- b **can be calculated as before**
- **Points for which** $\alpha_i \neq 0$ **are still support vectors**

SVMs for multiclass problems

If we have K classes $1, ..., K$.

- Train K SVMs (one-against-all)
  - Get K linear discriminant functions $f_1, ..., f_K$
- Assign a point to the class whose hyperplane is furthest from it
- Resulting classifier

$$h_{f_1,...,f_K}(\mathrm{x}) = \underset{i=1,...,K}{\operatorname{argmax}} f_i(\mathrm{x}) \tag{74}$$

SVMs in practice

In MATLAB

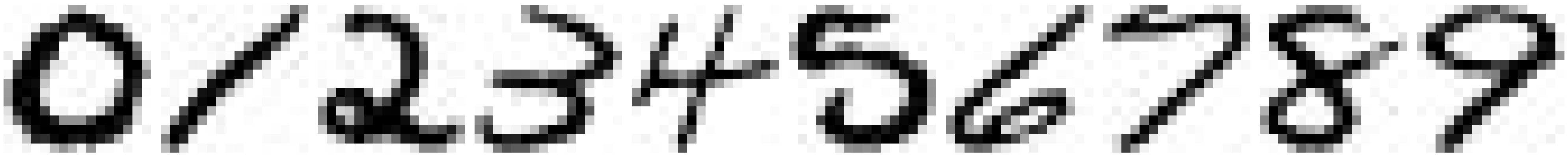
- Implement on your own (5 lines of code)
  - Use `quadprog` to solve the quadratic programming problem
- Or use the built in library (better optimizer)
  - `svmtrain` to train the SVM
  - `svmclassify` to classify new data points
- Documentation: `doc svmtrain` or `doc quadprog`

In C++

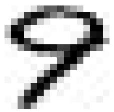
- Very good and easy to use libraries are available such as
  - LIBSVM http://www.csie.ntu.edu.tw/~cjlin/libsvm/
  - SVMLight http://svmlight.joachims.org/
- Highly optimized quadratic programming solvers
- Significantly faster than MATLAB

USPS Data Set

USPS Data set (9298 16×16 images)



Used feature vectors:



$$\rightarrow \begin{pmatrix} 5 \\ 4 \\ \vdots \\ 152 \\ 31 \end{pmatrix} \in \mathbb{R}^{256} \qquad (75)$$

Performance comparison

SVM performance benchmark as summarized by Vapnik in [Vapnik 95]

Classifier	Best parameter choice	$\|\mathcal{SV}\|$	Raw error in %
Human performance	-	-	2.5%
Decision tree, C4.5	-	-	16.2%
Best two-layer neural network	-	-	5.9%
Five-layer network (LeNet 1)	-	-	5.1%
SVM with polynomial kernel	$d = 3$	274	4.0%
SVM with Gaussian kernel	$\sigma^2 = 0.3$	291	4.1%

Limitations of SVMs

- It's not clear how an appropiate kernel should be chosen for a given problem
- Computationally expensive:
  - Training in the dual form (N training examples): $\mathcal{O}(N^3)$
    - Infeasible for large-scale applications
  - Training in the primal form (m-dimensional input space): $\mathcal{O}(m^3)$
  - Evaluation of f in the dual form more expensive than in the primal form
  - Evaluation practically infeasible if number of support vectors is very large

Conclusion

- SVMs use a simple linear model
- Feature space mappings enlarge the range of linearly separable training sets
  - They can efficiently be used by enabling kernel functions
- Good generalization performance
  - Margin concept
- Convex optimization problem
- In practice SVMs are good *blackbox classifiers*
  - They give reasonable good results without much effort
  - When dealing with new classification problems, it's often a good choice to try SVMs using different kernels

Support Vector Machines - References

See

- [Nocedal & Wrigt 06] for an indepth introduction to numerical optimization (theory and practice)
- [Bishop 06] and [Christiani & Shawe-Taylor 00] for the derivations of the dual forms
- [Burges 98] for some thoughts about limitations
- [Dalal & Triggs 05] for a practical application of SVMs in computer vision
- [Vapnik 95] For a performance comparison (Kernel SVM vs. Neural Network)
- [Shawe-Taylor & Cristianini 06] for more background on kernels
- [Mercer 09] for a proof of Mercer's theorem

Thank you for your attention

Tobias Pohlen

tobias.pohlen@rwth-aachen.de

http://www.geekstack.net/seminar-paper

GoBack

References

[Bartlett & Shawe–Taylor 99] P. Bartlett, J. Shawe–Taylor: Generalization Performance of Support Vector Machines and Other Pattern Classifiers. In B. Schölkopf, C.J.C. Burges, A.J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pp. 43–54, Cambridge, MA, 1999. MIT Press. 22

[Bishop 06] C.M. Bishop: *Pattern Recognition and Machine Learning*. Springer, New York, 2006. 47

[Boser & Guyon+ 92] B.E. Boser, I.M. Guyon, V.N. Vapnik: A Training Algorithm for Optimal Margin Classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pp. 144–152, New York, NY, USA, 1992. ACM. 22

[Burges 98] C.J.C. Burges: A Tutorial on Support Vector Machines for Pattern Recognition. *Data Min. Knowl. Discov.*, Vol. 2, No. 2, pp. 121–167, June 1998. 47

[Christiani & Shawe-Taylor 00] N. Christiani, J. Shawe-Taylor: *An introduction to support vector machines*. Cambridge University Press, 2000. 22, 47

[Dalal & Triggs 05] N. Dalal, B. Triggs: Histograms of Oriented Gradients for Human Detection. In C. Schmid, S. Soatto, C. Tomasi, editors, *International Conference on Computer Vision & Pattern Recognition*, Vol. 2, pp. 886–893, INRIA Rhône-Alpes, ZIRST-655, av. de l'Europe, Montbonnot-38334, June 2005. 47

[Mercer 09] J. Mercer: Functions of positive and negative type, and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society, London*, Vol. 209, pp. 415–446, 1909. 47

[Nocedal & Wrigt 06] J. Nocedal, S.J. Wrigt: *Numerical optimization*. Springer, second edition edition, 2006. 28, 47

[Shawe-Taylor & Bartlett$^+$ 98] J. Shawe-Taylor, P.L. Bartlett, R.C. Williamson, M. Anthony: Structural Risk Minimization Over Data-Dependent Hierarchies. *IEEE Transactions on Information Theory*, Vol. 44, No. 5, pp. 1926–1940, 1998. 22

[Shawe-Taylor & Cristianini 06] J. Shawe-Taylor, N. Cristianini: *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2006. 47

[Vapnik 82] V. Vapnik: *Estimation of Dependences Based on Empirical Data: Springer Series in Statistics (Springer Series in Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1982. 22

[Vapnik 95] V.N. Vapnik: *The Nature of Statistical Learning Theory*. Springer, 1995. 22, 44, 47