TECHNISCHE UNIVERSITÄT DARMSTADT

Statistical Machine Learning

Lecture 11: Support Vector Machines

Kristian Kersting

TU Darmstadt

Summer Term 2020

Today's Objectives

- Covered Topics
  - Linear Support Vector Classification
  - Features and Kernels
  - Non-Linear Support Vector Classification
  - Outlook on Applications, Relevance Vector Machines and Support Vector Regression

Outline

1. From Structural Risk Minimization to Linear SVMs

2. Nonlinear SVMs

3. Applications

4. Wrap-Up

TECHNISCHE UNIVERSITÄT DARMSTADT

Outline

1. From Structural Risk Minimization to Linear SVMs

2. Nonlinear SVMs

3. Applications

4. Wrap-Up

TECHNISCHE UNIVERSITÄT DARMSTADT

Structural Risk Minimization

- How can we implement structural risk minimization?

  $$R(\mathbf{w}) \leq R_{\text{emp}}(\mathbf{w}) + \epsilon(N, p^*, h)$$

  where N is the number of training examples, p^* is the probability that the bound is met and h is the VC-dimension

- Classical Machine Learning algorithms
  - Keep $\epsilon(N, p^*, h)$ constant and minimize $R_{\text{emp}}(\mathbf{w})$
  - $\epsilon(N, p^*, h)$ is fixed by keeping some model parameters fixed, e.g. the number of hidden neurons in a neural network (see later)

- **Support Vector Machines (SVMs)**
  - Keep $R_{\text{emp}}(\mathbf{w})$ constant and minimize $\epsilon(N, p^*, h)$
  - In practice $R_{\text{emp}}(\mathbf{w}) = 0$ with separable data
  - $\epsilon(N, p^*, h)$ is controlled by changing the VC-dimension (“capacity control”)

Support Vector Machines

- Linear classifiers (generalized later)
- Approximate implementation of the structural risk minimization principle
- If the data is linearly separable, the empirical risk of SVM classifiers will be zero, and the risk bound will be approximately minimized
- SVMs have built-in “guaranteed” generalization abilities

TECHNISCHE UNIVERSITÄT DARMSTADT

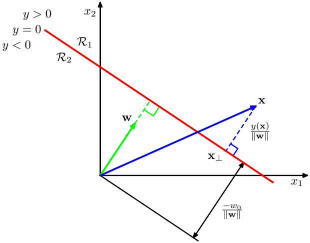
Support Vector Machines

- For now assume linearly separable data
- N training data points

$$\{\mathbf{x}_i, y_i\}_{i=1}^{N}, \text{ with } \mathbf{x}_i \in \mathbb{R}^d \text{ and } y_i \in \{-1, 1\}$$

- Hyperplane that separates the data

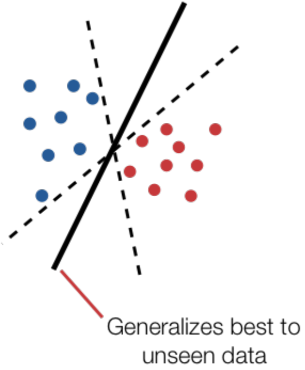
$$y(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$$

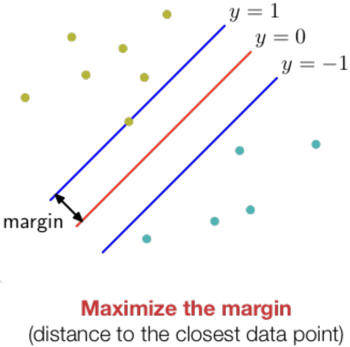


- Which hyperplane shall we use? How can we minimize the VC dimension?

Support Vector Machines

- Intuitively: We should find the hyperplane with the maximum "distance" to the data





TECHNISCHE UNIVERSITÄT DARMSTADT

Support Vector Machines

- Maximizing the margin
  - Why does that make sense?
  - Why does it minimize the VC dimension?
- Key result (from Vapnik)
  - If the data points lie in a sphere of radius R, $\|\mathbf{x}_i\| < R$, ...
  - ...and the margin of the linear classifier in d dimensions is γ, then

$$h \leq \min \left\{ d, \left\lceil \frac{4R^2}{\gamma^2} \right\rceil \right\}$$

- Maximizing the margin lowers a bound on the VC-dimension!

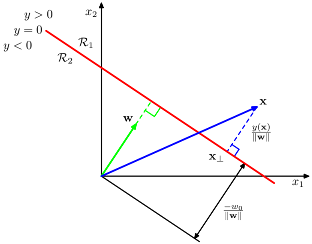
Support Vector Machines

- Find a hyperplane so that the data is linearly separated

$$y_i \left(\mathbf{w}^\intercal \mathbf{x}_i + b\right) \geq 1 \quad \forall i$$

- Enforce $y_i \left(\mathbf{w}^\intercal \mathbf{x}_i + b\right) = 1$ for at least one data point

Support Vector Machines



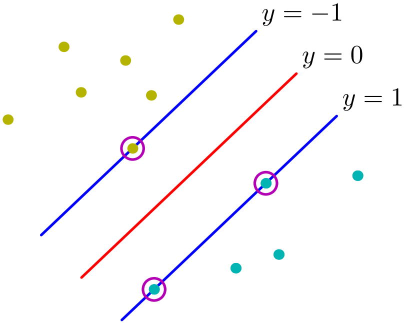
- We can easily express the margin
- The distance to the hyperplane is

$$\frac{y(\mathbf{x}_i)}{\|\mathbf{w}\|} = \frac{\mathbf{w}^\intercal \mathbf{x}_i + b}{\|\mathbf{w}\|}$$

  - (Note in the figure $b = w_0$)
- Hence the margin is $\frac{1}{\|\mathbf{w}\|}$

TECHNISCHE UNIVERSITÄT DARMSTADT

Support Vector Machines



- Support vectors: all points that lie on the margin, i.e., $y_i\left(\mathbf{w}^\intercal \mathbf{x}_i + b\right) = 1$

Support Vector Machines

- Maximizing the margin $1/\|\mathbf{w}\|$ is equivalent to minimizing $\|\mathbf{w}\|^2$
- Formulate as constrained optimization problem

$$\begin{aligned} \arg\min_{\mathbf{w},b} \quad & \tfrac{1}{2}\|\mathbf{w}\|^2 \\ \text{s.t.} \quad & y_i\left(\mathbf{w}^\intercal \mathbf{x}_i + b\right) - 1 \geq 0 \quad \forall i \end{aligned}$$

- Lagrangian formulation

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i \left(y_i\left(\mathbf{w}^\intercal \mathbf{x}_i + b\right) - 1\right)$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Support Vector Machines

$$\min L(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i \left(y_i \left(\mathbf{w}^\mathsf{T}\mathbf{x}_i + b\right) - 1\right)$$

$$\frac{\partial L(\mathbf{w}, b, \alpha)}{\partial b} = 0 \quad \Longrightarrow \quad \sum_{i=1}^{N} \alpha_i y_i = 0$$

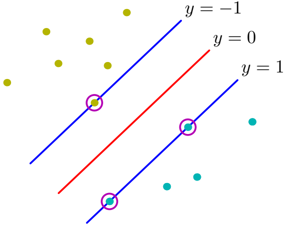
$$\frac{\partial L(\mathbf{w}, b, \alpha)}{\partial \mathbf{w}} = 0 \quad \Longrightarrow \quad \mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$$

- The separating hyperplane is a linear combination of the input data
- But what are the α_i?

TECHNISCHE UNIVERSITÄT DARMSTADT

Sparsity

- Important property
  - Almost all the α_i are zero
  - There are only a few support vectors



- But the hyperplane was written as

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$$

- SVMs are sparse learning machines
  - The classifier only depends on a few data points

TECHNISCHE UNIVERSITÄT DARMSTADT

Dual Form

- Let us rewrite the Lagrangian

$$
\begin{aligned}
L(\mathbf{w}, b, \alpha) &= \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i \left(y_i \left(\mathbf{w}^\mathsf{T} \mathbf{x}_i + b \right) - 1 \right) \\
&= \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i y_i \mathbf{w}^\mathsf{T} \mathbf{x}_i - \sum_{i=1}^{N} \alpha_i y_i b + \sum_{i=1}^{N} \alpha_i
\end{aligned}
$$

- We know that

$$
\sum_{i=1}^{N} \alpha_i y_i = 0
$$

- Hence we have

$$
\hat{L}(\mathbf{w}, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i y_i \mathbf{w}^\mathsf{T} \mathbf{x}_i + \sum_{i=1}^{N} \alpha_i
$$

Dual Form

$$\hat{L}(\mathbf{w}, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i y_i \mathbf{w}^\mathsf{T} \mathbf{x}_i + \sum_{i=1}^{N} \alpha_i$$

- Use the constraint $\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i$

$$\begin{aligned} \hat{L}(\mathbf{w}, \alpha) &= \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i y_i \sum_{j=1}^{N} \alpha_j y_j \mathbf{x}_j^\mathsf{T} \mathbf{x}_i + \sum_{i=1}^{N} \alpha_i \\ &= \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \left(\mathbf{x}_j^\mathsf{T} \mathbf{x}_i \right) + \sum_{i=1}^{N} \alpha_i \end{aligned}$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Dual Form

- We have also

$$\frac{1}{2}\|\mathbf{w}\|^2 = \frac{1}{2}\mathbf{w}^\mathsf{T}\mathbf{w} = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \left(\mathbf{x}_j^\mathsf{T}\mathbf{x}_i\right)$$

- Finally we obtain the Wolfe dual formulation

$$\tilde{L}(\alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \left(\mathbf{x}_j^\mathsf{T}\mathbf{x}_i\right)$$

- We can now solve the original problem by maximizing the dual function $\tilde{L}$

TECHNISCHE UNIVERSITÄT DARMSTADT

Support Vector Machines - Dual Form

$$\min \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \left(\mathbf{x}_j^\top \mathbf{x}_i \right)$$

$$\text{s.t. } \alpha_i \geq 0$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

- The separating hyperplane is given by the N_S support vectors

$$\mathbf{w} = \sum_{i=1}^{N_S} \alpha_i y_i \mathbf{x}_i$$

- b can also be computed, but we skip the derivation

TECHNISCHE UNIVERSITÄT DARMSTADT

Support Vector Machines so far

- Both the original SVM formulation (primal) as well as the derived dual formulation are quadratic programming problems (quadratic cost, linear constraints), which have unique solutions that can be computed efficiently
- Why did we bother to derive the dual form? To go beyond linear classifiers!

Outline

TECHNISCHE UNIVERSITÄT DARMSTADT

Nonlinear SVMs

- Nonlinear transformation ϕ of the data (features)

$$\mathbf{x} \in \mathbb{R}^d \quad \phi : \mathbb{R}^d \rightarrow H$$

- Hyperplane H (linear classifier in H)

$$\mathbf{w}^{\mathsf{T}} \phi(\mathbf{x}) + b = 0$$

- Nonlinear classifier in $\mathbb{R}^d$
- Same trick as in least-squares regression. So what is so special here?

TECHNISCHE UNIVERSITÄT DARMSTADT

Nonlinear SVMs

- Dual form

$$\min \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \left(\mathbf{x}_j^\mathsf{T} \mathbf{x}_i \right)$$

$$\text{s.t. } \alpha_i \geq 0$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

- With a nonlinear transformation, we obtain

$$\tilde{L}(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \left(\phi(\mathbf{x}_j)^\mathsf{T} \phi(\mathbf{x}_i) \right)$$

- $\phi(\mathbf{x}_i)$ only appears in scalar products with another $\phi(\mathbf{x}_j)$
- We only need to be able to evaluate scalar products

TECHNISCHE UNIVERSITÄT DARMSTADT

Nonlinear SVMs

- What about the discriminant function?

$$y(\mathbf{x}) = \mathbf{w}^{\mathsf{T}} \phi(\mathbf{x}) + b$$

- We can represent the weights differently and write the nonlinear discriminant function as

$$\mathbf{w} = \sum_{i=1}^{N_S} \alpha_i y_i \phi(\mathbf{x}_i)$$

$$y(\mathbf{x}) = \sum_{i=1}^{N_S} \alpha_i y_i \phi(\mathbf{x}_i)^{\mathsf{T}} \phi(\mathbf{x}) + b$$

  - where N_S is the number of support vectors

- The discriminant function can also be written with scalar products of the nonlinear features only

TECHNISCHE UNIVERSITÄT DARMSTADT

Nonlinear SVMs

- Both the dual optimization problem and the discriminant function can be written in terms of scalar products of the features
- We have already seen this when we talked about the dual version of the perceptron
- In fact the discriminant function even has the very same functional form

$$y(\mathbf{x}) = \sum_{i=1}^{N_S} \alpha_i y_i \phi(\mathbf{x}_i)^\intercal \phi(\mathbf{x}) + b$$

- **Key difference**: In an SVM the parameters α_i maximize the margin of the classifier, and have built-in generalization properties

TECHNISCHE UNIVERSITÄT DARMSTADT

Kernel Trick

- Kernel trick: replace every occurrence of a scalar product between features with a kernel function

$$K\left(\mathbf{x}_i, \mathbf{x}_j\right) = \phi\left(\mathbf{x}_i\right)^\intercal \phi\left(\mathbf{x}_j\right)$$

- If we can find a kernel function that is equivalent to this scalar product, we can avoid mapping into a high-dimensional space and instead compute the scalar-product directly
- What are examples of such kernels and when do they exist?

Polynomial Kernel

- Polynomial kernel of 2nd degree

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^\mathsf{T}\mathbf{y})^2 \quad \mathbf{x}, \mathbf{y} \in \mathbb{R}^2$$

- Equivalence to the dot product

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^\mathsf{T}\mathbf{y})^2 = x_1^2 y_1^2 + 2x_1x_2y_1y_2 + y_1^2y_2^2$$

$$\phi(\mathbf{x})^\mathsf{T} \phi(\mathbf{y}) = \begin{pmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{pmatrix}^\mathsf{T} \begin{pmatrix} y_1^2 \\ \sqrt{2}y_1y_2 \\ y_2^2 \end{pmatrix}$$

- Why is the kernel method an advantage?
  - Number of computations with kernel: 3 (dot product between **x** and **y**) + 1 (square the result) = 4
  - Number of computations with feature transformation and then dot product?

TECHNISCHE UNIVERSITÄT DARMSTADT

Polynomial Kernel

- We could also have used $\phi(\mathbf{x})$ as

$$\phi(\mathbf{x})^{\mathsf{T}}\phi(\mathbf{y}) = \frac{1}{\sqrt{2}}\begin{pmatrix} x_1^2 - x_2^2 \\ 2x_1x_2 \\ x_1^2 + x_2^2 \end{pmatrix}^{\mathsf{T}} \frac{1}{\sqrt{2}}\begin{pmatrix} y_1^2 - y_2^2 \\ 2y_1y_2 \\ y_1^2 + y_2^2 \end{pmatrix}$$

- $\phi(\mathbf{x})$ is not unique for a given kernel function $K(\mathbf{x}, \mathbf{y})$

TECHNISCHE UNIVERSITÄT DARMSTADT

Polynomial Kernel of Degree d

- Let $C_d(\mathbf{x})$ be the transformation that maps a vector into the space of all ordered monomials of degree d
- We can represent all polynomials of degree d as linear functions in this transformed space
- Example
  - Ordered monomials: $x_1^2, x_1x_2, x_2x_1, x_2^2$
  - Unordered monomials: x_1^2, x_1x_2, x_2^2
- The kernel $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^\top\mathbf{y})^d$ lets us compute arbitrary scalar products without doing the explicit mapping

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^\top\mathbf{y})^d = C_d(\mathbf{x})^\top C_d(\mathbf{y})$$

Polynomial Kernel of Degree d

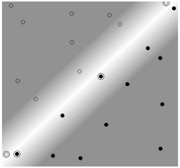
$$K(\mathbf{x}, \mathbf{y}) = \left(\mathbf{x}^\intercal \mathbf{y}\right)^d = C_d(\mathbf{x})^\intercal C_d(\mathbf{y})$$

- Dimensionality of the transformed space H: $\begin{pmatrix} d+N-1 \\ d \end{pmatrix}$
- Example

$$\begin{aligned} N &= 16 \times 16 = 256 \\ d &= 4 \\ \dim(H) &= 183181376 \end{aligned}$$

- The classifier has VC-dimension $\dim(H) + 1$!

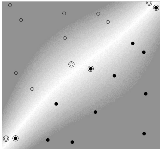
SVM - Linear Case



TECHNISCHE UNIVERSITÄT DARMSTADT

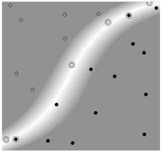
SVM with Kernels

- Polynomial kernel with degree 3



Linearly separable

Classifier almost linear



Not linearly separable

(in original space)

Constructing Kernels

- So far we identified some linear transformation $\phi(\mathbf{x})$ that we think will be useful
- Then we find a kernel $K(\mathbf{x}_i, \mathbf{x}_j)$ that allows us to compute the scalar product without making the mapping explicit

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^{\mathsf{T}} \phi(\mathbf{x}_j)$$

- What do kernels do?
  - They measure similarity (in a transformed space)
  - But what if we have a notion of similarity and want to encode this in a kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$ directly?

TECHNISCHE UNIVERSITÄT DARMSTADT

Radial Basis Functions

- **Radial Basis Function (RBF) kernel**

$$K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma^2}\right)$$

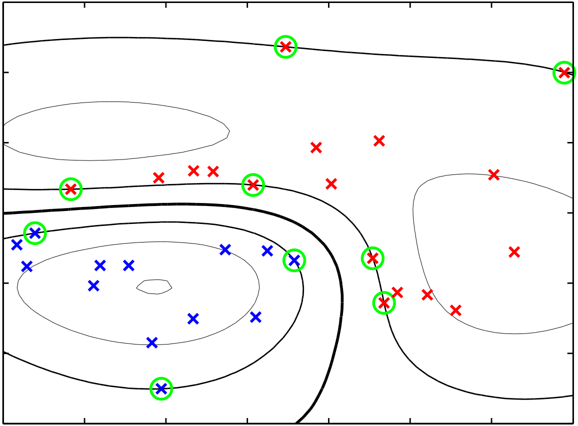
- Measures similarity between **x** and **y**
- Interesting property: *H* is infinite dimensional
  - Intuition given by Taylor series expansion

$$e^x = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \ldots + \frac{x^n}{n!} + \ldots$$

  - Since we only use the kernel function, it is not a problem
  - But the hyperplane also has infinite VC-dimension!

TECHNISCHE UNIVERSITÄT DARMSTADT

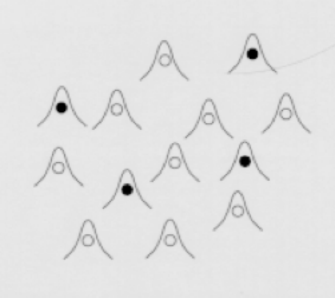
Radial Basis Function Kernel



TECHNISCHE UNIVERSITÄT DARMSTADT

VC-Dimension for RBF Kernel

- Intuition: If we can make the radius of the kernel arbitrarily small, then at some point every data point will have its "own" kernel



- But in contrast: If we bound the radius of the RBF, we can limit the VC-dimension!

TECHNISCHE UNIVERSITÄT DARMSTADT

Kernels

- Question: Is the Gaussian RBF kernel a valid kernel, i.e., is there a mapping $\{H, \phi\}$ so that

$$K(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^\intercal \phi(\mathbf{y}) \quad \text{with} \quad \phi : \mathbb{R}^d \to H$$

- How can we assess this more generally?

Mercer's Condition

- A function $K(\mathbf{x}, \mathbf{y})$ is a valid kernel, if for every $g(\mathbf{x})$ with

$$\int g(\mathbf{x})^2 \, \mathrm{d}\mathbf{x} < \infty$$

it holds that

$$\int \int K(\mathbf{x}, \mathbf{y}) \, g(\mathbf{x}) \, g(\mathbf{y}) \, \mathrm{d}\mathbf{x}\mathrm{d}\mathbf{y} \geq 0$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Kernels satisfying Mercer's condition

- Inhomogeneous polynomial kernel

$$K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^\intercal \mathbf{y} + c)^d$$

  - Can also represent polynomials of degree d

- Gaussian RBF kernel

$$K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|^2}{2\sigma^2}\right)$$

- Hyperbolic tangent kernel

$$K(\mathbf{x}, \mathbf{y}) = \tanh(a\mathbf{x}^\intercal \mathbf{y} + b)$$

TECHNISCHE UNIVERSITÄT DARMSTADT

Combining Kernels

- It may not be always easy to check if Mercer's condition is satisfied, but it is possible to construct new kernels out of known ones
- If $K_1(\mathbf{x}, \mathbf{y})$ and $K_2(\mathbf{x}, \mathbf{y})$ are valid kernels, then so are

$$
\begin{aligned}
& cK_1(\mathbf{x}, \mathbf{y}) \\
& K_1(\mathbf{x}, \mathbf{y}) + K_2(\mathbf{x}, \mathbf{y}) \\
& K_1(\mathbf{x}, \mathbf{y}) K_2(\mathbf{x}, \mathbf{y}) \\
& f(\mathbf{x}) K_1(\mathbf{x}, \mathbf{y}) f(\mathbf{y}) \\
& \ldots
\end{aligned}
$$

Non-separable data

- What if the data is not linearly separable?

- Simple solution: transform the features into a space so that they become linearly separable
  - E.g. RBF kernel with small kernel radius
- Problem: such a classifier will have a very high VC-dimension, and thus has a large capacity
  - It will lead to overfitting
  - Solution: allow for data points to "violate the margin"

SVMs with slack

- Instead of requiring that the data is perfectly linearly separable

$$\begin{aligned} \mathbf{w}^\intercal \mathbf{x}_i + b &\geq +1 \quad \text{for } y_i = +1 \\ \mathbf{w}^\intercal \mathbf{x}_i + b &\leq -1 \quad \text{for } y_i = -1 \end{aligned}$$

- Allow for small violations ξ_i from perfect separation

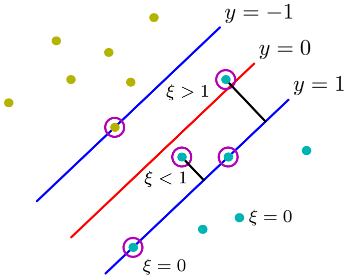
$$\begin{aligned} \mathbf{w}^\intercal \mathbf{x}_i + b &\geq +1 - \xi_i \quad \text{for } y_i = +1 \\ \mathbf{w}^\intercal \mathbf{x}_i + b &\leq -1 + \xi_i \quad \text{for } y_i = -1 \\ \xi_i &\geq 0 \quad \forall i \end{aligned}$$

SVMs with slack

- We require that

$$y_i \left(\mathbf{w}^\top \mathbf{x}_i + b\right) \geq 1 - \xi_i, \quad \xi_i \geq 0 \; \forall i$$

- ξ_i are called *slack variables*



TECHNISCHE UNIVERSITÄT DARMSTADT

SVMs with slack

- We have to penalize the deviations

$$\arg\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i$$

$$\text{s.t. } y_i\left(\mathbf{w}^\intercal\mathbf{x}_i + b\right) - 1 + \xi_i \geq 0$$

$$\xi_i \geq 0$$

- Maximize the margin while minimizing the penalty for all data points that are not outside the margin
- The weight C allows us to specify a trade-off. Typically determined through cross-validation
- Even if the data is separable, it may be better to allow for an occasional penalty

SVMs with slack

- Dual formulation

$$\max \ \tilde{L}(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \left(\mathbf{x}_j^T \mathbf{x}_i \right)$$

$$\text{s.t. } 0 \leq \alpha_i \leq C$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

  where $\alpha_i \leq C$ is called *box constraint*

- The separating hyperplane is given by the N_S support vectors

$$\mathbf{w} = \sum_{i=1}^{N_S} \alpha_i y_i \mathbf{x}_i$$

Outline

Text Classification

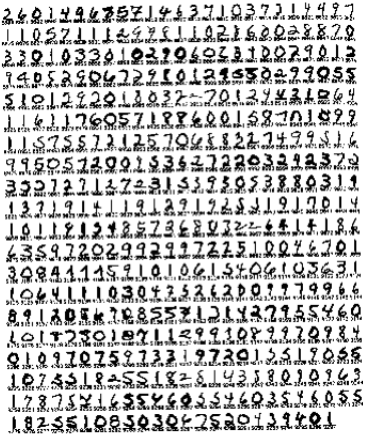
- Joachims, T., *Text categorization with Support Vector Machines: learning with many relevant features*, EMCL 1998
- Problem: Classify documents into a number of categories
- The text is represented using word statistics, i.e. histograms of the word frequency
  - We count how often every word occurs and ignore their order (“bag of words”)
  - Very high-dimensional feature space (roughly 10,000 dimensions)
  - Very few features that are not relevant (difficult to apply feature selection or dimensionality reduction)

Text Classification

	Bayes	Rocchio	C4.5	k-NN	SVM (poly) degree $d =$ 1	2	3	4	5	SVM (rbf) width $\gamma =$ 0.6	0.8	1.0	1.2
earn	95.9	96.1	96.1	97.3	98.2	98.4	**98.5**	98.4	98.3	**98.5**	98.5	98.4	98.3
acq	91.5	92.1	85.3	92.0	92.6	94.6	**95.2**	95.2	95.3	95.0	95.3	95.3	**95.4**
money-fx	62.9	67.6	69.4	78.2	66.9	72.5	75.4	74.9	**76.2**	74.0	75.4	**76.3**	75.9
grain	72.5	79.5	89.1	82.2	91.3	93.1	**92.4**	91.3	89.9	**93.1**	91.9	91.9	90.6
crude	81.0	81.5	75.5	85.7	86.0	87.3	88.6	**88.9**	87.8	**88.9**	89.0	88.9	88.2
trade	50.0	77.4	59.2	77.4	69.2	75.5	76.6	77.3	**77.1**	76.9	78.0	**77.8**	76.8
interest	58.0	72.5	49.1	74.0	69.8	63.3	67.9	73.1	**76.2**	74.4	75.0	**76.2**	76.1
ship	78.7	83.1	80.9	79.2	82.0	85.4	86.0	**86.5**	86.0	**85.4**	86.5	87.6	87.1
wheat	60.6	79.4	85.5	76.6	83.1	84.5	85.2	**85.9**	83.8	**85.2**	85.9	85.9	85.9
corn	47.3	62.2	87.7	77.9	86.0	86.5	85.3	**85.7**	83.9	**85.1**	85.7	85.7	84.5
microavg.	**72.0**	**79.9**	**79.4**	**82.3**	84.2	85.1	85.9	86.2	85.9	86.4	86.5	86.3	86.2
					combined: **86.0**					combined: **86.4**			

Handwritten Digit Classification

- U.S. Postal Service Database



TECHNISCHE UNIVERSITÄT DARMSTADT

Handwritten Digit Classification

- Human performance: 2.5% error
- Various learning algorithms
  - 16.2%:
  - 5.9%: 2-layer neural network
  - 5.1%: LeNet 1 - 5-layer neural network
- Various SVM results
  - 4.0%: Polynomial kernel ($p = 3$, 274 support vectors)
  - 4.1%: Gaussian kernel ($\sigma = 0.3$, 291 support vectors)

Handwritten Digit Classification

- Very little overfitting and good generalization

degree of polynomial	dimensionality of feature space	support vectors	raw error
1	256	282	8.9
2	≈ 33000	227	4.7
3	$\approx 1 \times 10^6$	274	4.0
4	$\approx 1 \times 10^9$	321	4.2
5	$\approx 1 \times 10^{12}$	374	4.3
6	$\approx 1 \times 10^{14}$	377	4.5
7	$\approx 1 \times 10^{16}$	422	4.5

TECHNISCHE UNIVERSITÄT DARMSTADT

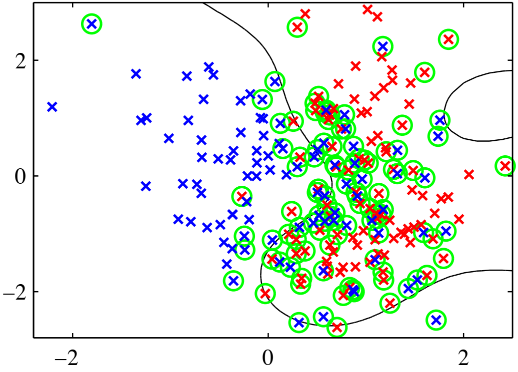
Handwritten Digit Classification

- To get even better results
  - Supply knowledge about invariances in the data: geometric deformations, etc.
  - 2.7% error: elastic matching (no learning)
    - Use knowledge of how digits can deform
    - Classify test digit by finding the template that required least deformation
- Recent results
  - With more training data, better modeling of invariances, etc.
  - Error down to about 0.5% with SVMs and 0.4% with neural networks

TECHNISCHE UNIVERSITÄT DARMSTADT

(Lack of) Sparseness

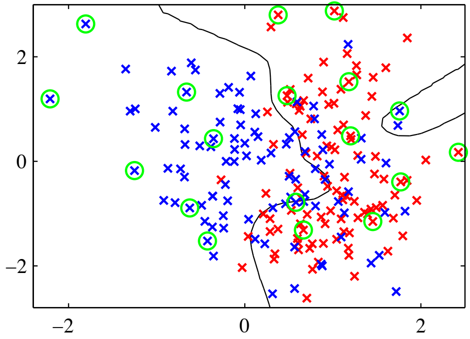
- If the classes overlap, SVMs may need many support vectors



TECHNISCHE UNIVERSITÄT DARMSTADT

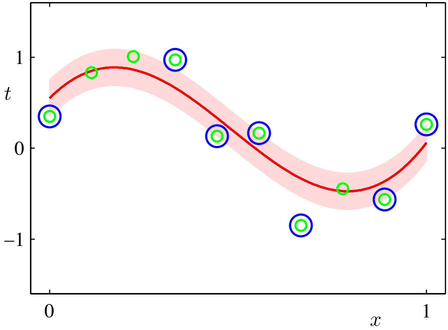
Relevance Vector Machines

- Probabilistic alternative to SVMs
- Much sparser results
- No notion of margin maximization



Support Vector Regression

- SVMs can also be adapted to regression tasks



Outline

4. Wrap-Up

You know now

- What the main idea behind SVMs is
- Why maximizing the margin is a good idea
- How to translate the SVM problem into a quadratic optimization problem
- How to interpret the support vectors
- How to use SVMs for data that is not linearly separable
- What the kernel trick is
- How to construct kernels
- How to formulate SVMs with slack variables

Self-Test Questions

- How did learning theory motivate support vector machines?
- What does maximum margin separation mean?
- Why did the SVM-craze drown the Neural-Networks-craze?
- What is a Kernel?
- How does a Kernel relate to features?
- How can I build Kernels from Kernels?
- What functions does the Radial Basis Function Kernel contain?
- How does support vector regression work?

Homework

- Reading Assignment for next lecture
  - Bishop 6.1, 6.3, 6.4