# Machine Learning - Lecture 8

## Linear Support Vector Machines

**24.05.2016**

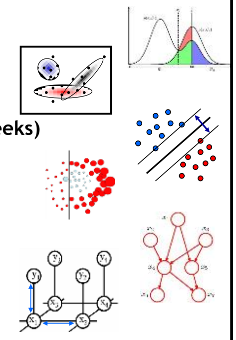
Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de/

leibe@vision.rwth-aachen.de



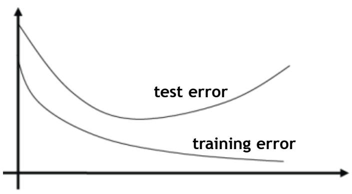
## Course Outline

- Fundamentals (2 weeks)
  - Bayes Decision Theory
  - Probability Density Estimation
- Discriminative Approaches (5 weeks)
  - Linear Discriminant Functions
  - Statistical Learning Theory & SVMs
  - Ensemble Methods & Boosting
  - Randomized Trees, Forests & Ferns
- Generative Models (4 weeks)
  - Bayesian Networks
  - Markov Random Fields





## Recap: Generalization and Overfitting




- Goal: predict class labels of new observations
  - Train classification model on limited training set.
  - The further we optimize the model parameters, the more the training error will decrease.
  - However, at some point the test error will go up again.
  - ⇒ *Overfitting to the training set!*



## Recap: Risk

- Empirical risk
  - Measured on the training/validation set

$$R_{emp}(\alpha) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, f(\mathbf{x}_i;\alpha))$$

- Actual risk (= Expected risk)
  - Expectation of the error on *all* data.

$$R(\alpha) = \int L(y_i, f(\mathbf{x};\alpha))dP_{X,Y}(\mathbf{x}, y)$$

  - $P_{X,Y}(\mathbf{x}, y)$ is the probability distribution of $(\mathbf{x}, y)$. It is fixed, but typically unknown.
  - ⇒ In general, we can't compute the actual risk directly!



## Recap: Statistical Learning Theory

- Idea
  - Compute an upper bound on the actual risk based on the empirical risk

$$R(\alpha) \cdot\ R_{emp}(\alpha) + \epsilon(N, p^*, h)$$

  - where

    $N$: number of training examples

    $p^*$: probability that the bound is correct

    $h$: capacity of the learning machine ("VC-dimension")



## Recap: VC Dimension

see Exercise 2.3

- Vapnik-Chervonenkis dimension
  - Measure for the capacity of a learning machine.
- Formal definition:
  - *If a given set of $\ell$ points can be labeled in all possible $2^\ell$ ways, and for each labeling, a member of the set $\{f(\alpha)\}$ can be found which correctly assigns those labels, we say that the set of points is shattered by the set of functions.*
  - *The VC dimension for the set of functions $\{f(\alpha)\}$ is defined as the maximum number of training points that can be shattered by $\{f(\alpha)\}$.*

## VC Dimension

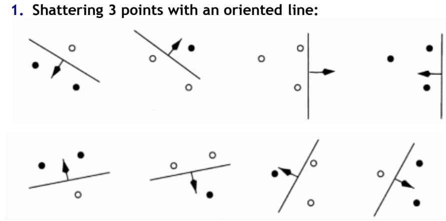
- Interpretation as a two-player game
  - Opponent's turn: He says a number $N$.
  - Our turn: We specify a set of $N$ points $\{\mathbf{x}_1,\ldots,\mathbf{x}_N\}$.
  - Opponent's turn: He gives us a labeling $\{\mathbf{x}_1,\ldots,\mathbf{x}_N\} \in \{0,1\}^N$
  - Our turn: We specify a function $f(\alpha)$ which correctly classifies all $N$ points.

⇒ If we can do that for all $2^N$ possible labelings, then the VC dimension is at least $N$.



## VC Dimension

- Example
  - The VC dimension of all oriented lines in $\mathbb{R}^2$ is 3.
    1. Shattering 3 points with an oriented line:



    2. More difficult to show: it is not possible to shatter 4 points (XOR)…
  - More general: the VC dimension of all hyperplanes in $\mathbb{R}^n$ is $n$+1.



## VC Dimension

- Intuitive feeling (unfortunately wrong)
  - The VC dimension has a direct connection with the number of parameters.
- Counterexample

$$f(x;\alpha) = g(\sin(\alpha x))$$

$$g(x) = \begin{cases} 1, & x > 0 \\ -1, & x \cdot 0 \end{cases}$$

  - Just a single parameter $\alpha$.
  - Infinite VC dimension
    - Proof: Choose $x_i = 10^{-i}, \quad i = 1,\ldots,\ell$

$$\alpha = \pi \left(1 + \sum_{i=1}^{\ell} \frac{(1-y_i)10^i}{2}\right)$$



## Upper Bound on the Risk

- Important result (Vapnik 1979, 1995)
  - With probability (1-$\eta$), the following bound holds

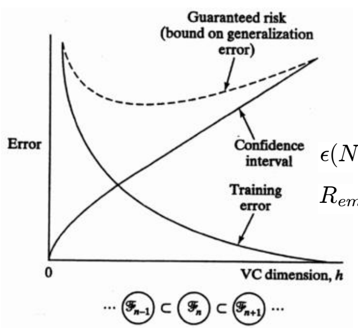
$$R(\alpha) \cdot \; R_{emp}(\alpha) + \underbrace{\sqrt{\frac{h(\log(2N/h)+1)-\log(\eta/4)}{N}}}_{\text{"VC confidence"}}$$

  - This bound is independent of $P_{X,Y}(\mathbf{x},y)$!
  - Typically, we cannot compute the left-hand side (the actual risk)
  - If we know $h$ (the VC dimension), we can however easily compute the risk bound

$$R(\alpha) \cdot \; R_{emp}(\alpha) + \epsilon(N, p^*, h)$$



## Upper Bound on the Risk




$\epsilon(N, p^*, h)$

$R_{emp}(\alpha)$



## Recap: Structural Risk Minimization

- How can we implement Structural Risk Minimization?

$$R(\alpha) \cdot \; R_{emp}(\alpha) + \epsilon(N, p^*, h)$$

- Classic approach
  - Keep $\epsilon(N, p^*, h)$ constant and minimize $R_{emp}(\alpha)$.
  - $\epsilon(N, p^*, h)$ can be kept constant by controlling the model parameters.
- Support Vector Machines (SVMs)
  - Keep $R_{emp}(\alpha)$ constant and minimize $\epsilon(N, p^*, h)$.
  - In fact: $R_{emp}(\alpha) = 0$ for separable data.
  - Control $\epsilon(N, p^*, h)$ by adapting the VC dimension (controlling the "capacity" of the classifier).

RWTH AACHEN UNIVERSITY

## Topics of This Lecture

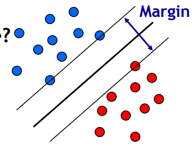
- Linear Support Vector Machines
  - Lagrangian (primal) formulation
  - Dual formulation
  - Discussion
- Linearly non-separable case
  - Soft-margin classification
  - Updated formulation
- Nonlinear Support Vector Machines
  - Nonlinear basis functions
  - The Kernel trick
  - Mercer's condition
  - Popular kernels
- Applications



RWTH AACHEN UNIVERSITY

## Revisiting Our Previous Example…

- How to select the classifier with the best generalization performance?
  - Intuitively, we would like to select the classifier which leaves maximal "safety room" for future data points.
  - This can be obtained by maximizing the margin between positive and negative data points.
  - It can be shown that the larger the margin, the lower the corresponding classifier's VC dimension.



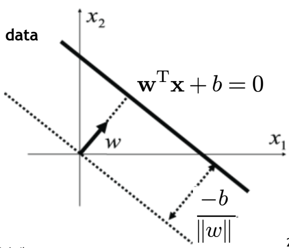

- The SVM takes up this idea
  - It searches for the classifier with maximum margin.
  - Formulation as a convex optimization problem
    ⇒ Possible to find the globally optimal solution!



RWTH AACHEN UNIVERSITY

## Support Vector Machine (SVM)

- Let's first consider linearly separable data
  - $N$ training data points $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ $\quad \mathbf{x}_i \in \mathbb{R}^d$
  - Target values $\quad t_i \in \{-1, 1\}$
  - Hyperplane separating the data

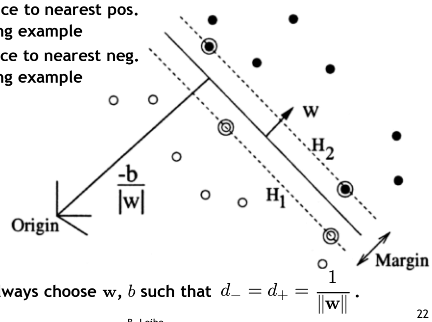





RWTH AACHEN UNIVERSITY

## Support Vector Machine (SVM)

- Margin of the hyperplane: $d_- + d_+$
  - $d_+$: distance to nearest pos. training example
  - $d_-$: distance to nearest neg. training example




  - We can always choose $\mathbf{w}, b$ such that $d_- = d_+ = \frac{1}{\|\mathbf{w}\|}$.



RWTH AACHEN UNIVERSITY

## Support Vector Machine (SVM)

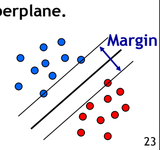
- Since the data is linearly separable, there exists a hyperplane with

$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b \geq +1 \quad \text{for} \quad t_n = +1$$
$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b \cdot \; -1 \quad \text{for} \quad t_n = -1$$

- Combined in one equation, this can be written as

$$t_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) \geq 1 \quad \forall n$$

⇒ Canonical representation of the decision hyperplane.
  - The equation will hold exactly for the points on the margin
    $$t_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) = 1$$
  - By definition, there will always be at least one such point.






RWTH AACHEN UNIVERSITY

## Support Vector Machine (SVM)

- We can choose $w$ such that
$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b = +1 \quad \text{for one} \quad t_n = +1$$
$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b = -1 \quad \text{for one} \quad t_n = -1$$
- The distance between those two hyperplanes is then the margin
$$d_- = d_+ = \frac{1}{\|\mathbf{w}\|}$$
$$d_- + d_+ = \frac{2}{\|\mathbf{w}\|}$$

⇒ We can find the hyperplane with maximal margin by minimizing $\|\mathbf{w}\|^2$.

## Support Vector Machine (SVM)

- **Optimization problem**
  - Find the hyperplane satisfying

$$\arg\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2$$

under the constraints

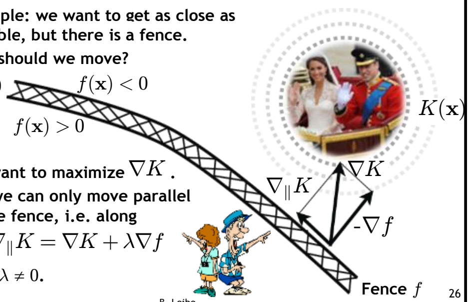
$$t_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) \geq 1 \quad \forall n$$

  - Quadratic programming problem with linear constraints.
  - Can be formulated using Lagrange multipliers.
- ***Who is already familiar with Lagrange multipliers?***
  - Let's look at a real-life example…



## Recap: Lagrange Multipliers

- **Problem**
  - We want to maximize $K(\mathbf{x})$ subject to constraints $f(\mathbf{x}) = 0$.
  - Example: we want to get as close as possible, but there is a fence.
  - How should we move?




  - We want to maximize $\nabla K$.
  - But we can only move parallel to the fence, i.e. along

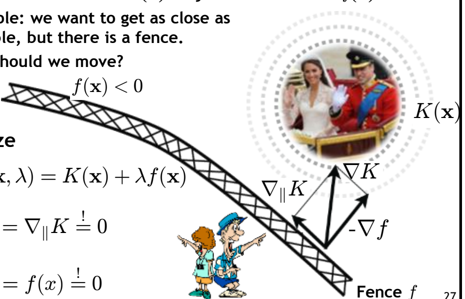
$$\nabla_{\|}K = \nabla K + \lambda \nabla f$$

with $\lambda \neq 0$.



## Recap: Lagrange Multipliers

- **Problem**
  - We want to maximize $K(\mathbf{x})$ subject to constraints $f(\mathbf{x}) = 0$.
  - Example: we want to get as close as possible, but there is a fence.
  - How should we move?




⇒ Optimize

$$\max_{\mathbf{x},\lambda} L(\mathbf{x},\lambda) = K(\mathbf{x}) + \lambda f(\mathbf{x})$$

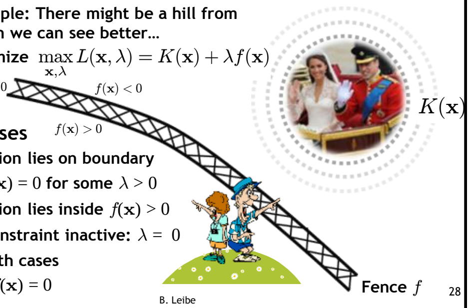
$$\frac{\partial L}{\partial \mathbf{x}} = \nabla_{\|}K \stackrel{!}{=} 0$$

$$\frac{\partial L}{\partial \lambda} = f(x) \stackrel{!}{=} 0$$



## Recap: Lagrange Multipliers

- **Problem**
  - Now let's look at constraints of the form $f(\mathbf{x}) \geq 0$.
  - Example: There might be a hill from which we can see better…
  - Optimize $\max_{\mathbf{x},\lambda} L(\mathbf{x},\lambda) = K(\mathbf{x}) + \lambda f(\mathbf{x})$

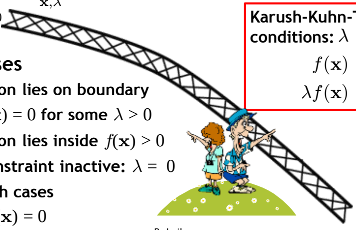



- **Two cases**
  - Solution lies on boundary
    ⇒ $f(\mathbf{x}) = 0$ for some $\lambda > 0$
  - Solution lies inside $f(\mathbf{x}) > 0$
    ⇒ Constraint inactive: $\lambda = 0$
  - In both cases
    ⇒ $\lambda f(\mathbf{x}) = 0$



## Recap: Lagrange Multipliers

- **Problem**
  - Now let's look at constraints of the form $f(\mathbf{x}) \geq 0$.
  - Example: There might be a hill from which we can see better…
  - Optimize $\max_{\mathbf{x},\lambda} L(\mathbf{x},\lambda) = K(\mathbf{x}) + \lambda f(\mathbf{x})$




Karush-Kuhn-Tucker (KKT) conditions:

$$\lambda \geq 0$$
$$f(\mathbf{x}) \geq 0$$
$$\lambda f(\mathbf{x}) = 0$$

- **Two cases**
  - Solution lies on boundary
    ⇒ $f(\mathbf{x}) = 0$ for some $\lambda > 0$
  - Solution lies inside $f(\mathbf{x}) > 0$
    ⇒ Constraint inactive: $\lambda = 0$
  - In both cases
    ⇒ $\lambda f(\mathbf{x}) = 0$



## SVM – Lagrangian Formulation

- **Find hyperplane minimizing $\|\mathbf{w}\|^2$ under the constraints**

$$t_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) - 1 \geq 0 \quad \forall n$$

- **Lagrangian formulation**
  - Introduce positive Lagrange multipliers: $a_n \geq 0 \quad \forall n$
  - Minimize Lagrangian ("primal form")

$$L(\mathbf{w},b,\mathbf{a}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n \left\{t_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) - 1\right\}$$

  - I.e., find $\mathbf{w}$, $b$, and $\mathbf{a}$ such that

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{n=1}^{N} a_n t_n = 0 \qquad \frac{\partial L}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{n=1}^{N} a_n t_n \mathbf{x}_n$$

## SVM – Lagrangian Formulation



- Lagrangian primal form

$$L_p = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n \{t_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) - 1\}$$
$$= \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n \{t_n y(\mathbf{x}_n) - 1\}$$

- The solution of $L_p$ needs to fulfill the KKT conditions
  - Necessary and sufficient conditions

$$a_n \geq 0$$
$$t_n y(\mathbf{x}_n) - 1 \geq 0$$
$$a_n \{t_n y(\mathbf{x}_n) - 1\} = 0$$

KKT:
$$\lambda \geq 0$$
$$f(\mathbf{x}) \geq 0$$
$$\lambda f(\mathbf{x}) = 0$$



## SVM – Solution (Part 1)



- Solution for the hyperplane
  - Computed as a linear combination of the training examples

$$\mathbf{w} = \sum_{n=1}^{N} a_n t_n \mathbf{x}_n$$

  - Because of the KKT conditions, the following must also hold

$$a_n \left(t_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) - 1\right) = 0$$

KKT: $\lambda f(\mathbf{x}) = 0$

  - This implies that $a_n > 0$ only for training data points for which

$$\left(t_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) - 1\right) = 0$$

⇒ *Only some of the data points actually influence the decision boundary!*



## SVM – Support Vectors



- The training points for which $a_n > 0$ are called "support vectors".
- Graphical interpretation:
  - The support vectors are the points on the margin.
  - They *define* the margin and thus the hyperplane.

⇒ Robustness to "too correct" points!

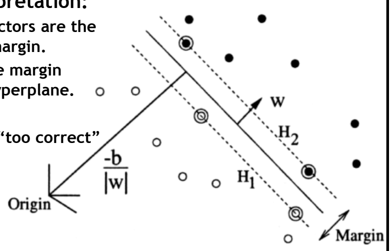




Image source: C. Burges, 1998

## SVM – Solution (Part 2)



- Solution for the hyperplane
  - To define the decision boundary, we still need to know $b$.
  - Observation: any support vector $\mathbf{x}_n$ satisfies

$$t_n y(\mathbf{x}_n) = t_n \left( \sum_{m \in \mathcal{S}} a_m t_m \mathbf{x}_m^{\mathrm{T}} \mathbf{x}_n + b \right) = 1$$

KKT: $f(\mathbf{x}) \geq 0$

  - Using $t_n^2 = 1$, we can derive: $b = t_n - \sum_{m \in \mathcal{S}} a_m t_m \mathbf{x}_m^{\mathrm{T}} \mathbf{x}_n$
  - In practice, it is more robust to average over all support vectors:

$$b = \frac{1}{N_{\mathcal{S}}} \sum_{n \in \mathcal{S}} \left( t_n - \sum_{m \in \mathcal{S}} a_m t_m \mathbf{x}_m^{\mathrm{T}} \mathbf{x}_n \right)$$



## SVM – Discussion (Part 1)



- Linear SVM
  - Linear classifier
  - Approximative implementation of the SRM principle.
  - In case of separable data, the SVM produces an empirical risk of zero with minimal value of the VC confidence (i.e. a classifier minimizing the upper bound on the actual risk).
  - SVMs thus have a "guaranteed" generalization capability.
  - Formulation as convex optimization problem.

  ⇒ Globally optimal solution!
- Primal form formulation
  - Solution to quadratic prog. problem in $M$ variables is in $\mathcal{O}(M^3)$.
  - Here: $D$ variables ⇒ $\mathcal{O}(D^3)$
  - Problem: scaling with high-dim. data ("curse of dimensionality")



## SVM – Dual Formulation



- Improving the scaling behavior: rewrite $L_p$ in a dual form

$$L_p = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n \{t_n(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b) - 1\}$$
$$= \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n t_n \mathbf{w}^{\mathrm{T}}\mathbf{x}_n - b \underbrace{\sum_{n=1}^{N} a_n t_n}_{=0} + \sum_{n=1}^{N} a_n$$

  - Using the constraint $\sum_{n=1}^{N} a_n t_n = 0$, we obtain $\quad \frac{\partial L_p}{\partial b} = 0$

$$L_p = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n t_n \mathbf{w}^{\mathrm{T}}\mathbf{x}_n + \sum_{n=1}^{N} a_n$$

## SVM - Dual Formulation

$$L_p = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n t_n \mathbf{w}^{\mathrm{T}}\mathbf{x}_n + \sum_{n=1}^{N} a_n$$

- Using the constraint $\mathbf{w} = \sum_{n=1}^{N} a_n t_n \mathbf{x}_n$ , we obtain $\frac{\partial L_p}{\partial \mathbf{w}} = 0$

$$L_p = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n t_n \sum_{m=1}^{N} a_m t_m \mathbf{x}_m^{\mathrm{T}}\mathbf{x}_n + \sum_{n=1}^{N} a_n$$

$$= \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m (\mathbf{x}_m^{\mathrm{T}}\mathbf{x}_n) + \sum_{n=1}^{N} a_n$$



## SVM - Dual Formulation

$$L = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m (\mathbf{x}_m^{\mathrm{T}}\mathbf{x}_n) + \sum_{n=1}^{N} a_n$$

- Applying $\frac{1}{2}\|\mathbf{w}\|^2 = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}$ and again using $\mathbf{w} = \sum_{n=1}^{N} a_n t_n \mathbf{x}_n$

$$\frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} = \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m (\mathbf{x}_m^{\mathrm{T}}\mathbf{x}_n)$$

- Inserting this, we get the Wolfe dual

$$L_d(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m (\mathbf{x}_m^{\mathrm{T}}\mathbf{x}_n)$$



## SVM - Dual Formulation

- Maximize

$$L_d(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m (\mathbf{x}_m^{\mathrm{T}}\mathbf{x}_n)$$

under the conditions

$$a_n \geq 0 \quad \forall n$$

$$\sum_{n=1}^{N} a_n t_n = 0$$

- The hyperplane is given by the $N_S$ support vectors:

$$\mathbf{w} = \sum_{n=1}^{N_S} a_n t_n \mathbf{x}_n$$



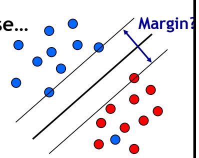
## SVM - Discussion (Part 2)

- Dual form formulation
  - In going to the dual, we now have a problem in $N$ variables ($a_n$).
  - Isn't this worse??? We penalize large training sets!
- However…
  1. SVMs have sparse solutions: $a_n \neq 0$ only for support vectors!
     ⇒ This makes it possible to construct efficient algorithms
     - e.g. Sequential Minimal Optimization (SMO)
     - Effective runtime between $\mathcal{O}(N)$ and $\mathcal{O}(N^2)$.
  2. We have avoided the dependency on the dimensionality.
     ⇒ This makes it possible to work with infinite-dimensional feature spaces by using suitable basis functions $\phi(\mathbf{x})$.
     ⇒ We'll see that in a few minutes…



## So Far…

- Only looked at linearly separable case…
  - Current problem formulation has no solution if the data are not linearly separable!
  - Need to introduce some tolerance to outlier data points.






## SVM - Non-Separable Data

- Non-separable data
  - I.e. the following inequalities cannot be satisfied for all data points

$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b \geq +1 \quad \text{for} \quad t_n = +1$$
$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b \cdot -1 \quad \text{for} \quad t_n = -1$$

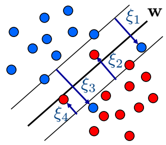
  - Instead use

$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b \geq +1 - \xi_n \quad \text{for} \quad t_n = +1$$
$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + b \cdot -1 + \xi_n \quad \text{for} \quad t_n = -1$$

with "slack variables" $\xi_n \geq 0 \quad \forall n$

## SVM - Soft-Margin Classification

- Slack variables
  - One slack variable $\xi_n \geq 0$ for each training data point.
- Interpretation
  - $\xi_n = 0$ for points that are on the correct side of the margin.
  - $\xi_n = |t_n - y(\mathbf{x}_n)|$ for all other points (linear penalty).




Point on decision boundary: $\xi_n = 1$

Misclassified point: $\xi_n > 1$

- We do not have to set the slack variables ourselves!

$\Rightarrow$ They are jointly optimized together with $\mathbf{w}$.

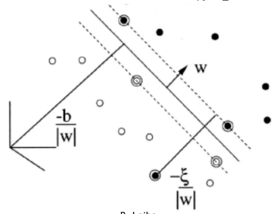
*How that?*



## SVM - Non-Separable Data

- Separable data
  - Minimize $\frac{1}{2}\|\mathbf{w}\|^2$
- Non-separable data
  - Minimize $\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{n=1}^{N}\xi_n$

Trade-off parameter!






## SVM - New Primal Formulation

- New SVM Primal: Optimize

$$L_p = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{n=1}^{N}\xi_n - \underbrace{\sum_{n=1}^{N} a_n\left(t_n y(\mathbf{x}_n) - 1 + \xi_n\right)}_{\text{Constraint}} - \underbrace{\sum_{n=1}^{N}\mu_n\xi_n}_{\text{Constraint}}$$

Constraint: $t_n y(\mathbf{x}_n) \geq 1 - \xi_n$

Constraint: $\xi_n \geq 0$

- KKT conditions

$$\begin{aligned} a_n &\geq 0 & \mu_n &\geq 0 \\ t_n y(\mathbf{x}_n) - 1 + \xi_n &\geq 0 & \xi_n &\geq 0 \\ a_n\left(t_n y(\mathbf{x}_n) - 1 + \xi_n\right) &= 0 & \mu_n\xi_n &= 0 \end{aligned}$$

KKT:

$$\begin{aligned} \lambda &\geq 0 \\ f(\mathbf{x}) &\geq 0 \\ \lambda f(\mathbf{x}) &= 0 \end{aligned}$$



## SVM - New Dual Formulation

- New SVM Dual: Maximize

$$L_d(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m (\mathbf{x}_m^{\mathrm{T}}\mathbf{x}_n)$$

under the conditions

$$0 \cdot a_n \cdot C$$

$$\sum_{n=1}^{N} a_n t_n = 0$$

This is all that changed!

- This is again a quadratic programming problem

$\Rightarrow$ Solve as before... (more on that later)

Slide adapted from Bernt Schiele



## SVM - New Solution

- Solution for the hyperplane
  - Computed as a linear combination of the training examples

$$\mathbf{w} = \sum_{n=1}^{N} a_n t_n \mathbf{x}_n$$

  - Again sparse solution: $a_n = 0$ for points outside the margin.

$\Rightarrow$ The slack points with $\xi_n > 0$ are now also support vectors!

  - Compute $b$ by averaging over all $N_{\mathcal{M}}$ points with $0 < a_n < C$:

$$b = \frac{1}{N_{\mathcal{M}}}\sum_{n\in\mathcal{M}}\left(t_n - \sum_{m\in\mathcal{M}} a_m t_m \mathbf{x}_m^{\mathrm{T}}\mathbf{x}_n\right)$$



## Interpretation of Support Vectors

- Those are the hard examples!
  - We can visualize them, e.g. for face detection

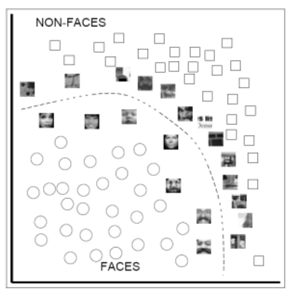
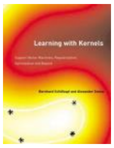



Image source: E. Osuna, F. Girosi, 1997

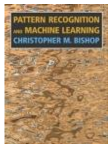
# References and Further Reading

- More information on SVMs can be found in Chapter 7.1 of Bishop's book. You can also look at Schölkopf & Smola (some chapters available online).




Christopher M. Bishop
Pattern Recognition and Machine Learning
Springer, 2006

B. Schölkopf, A. Smola
Learning with Kernels
MIT Press, 2002
[http://www.learning-with-kernels.org/](http://www.learning-with-kernels.org/)




- A more in-depth introduction to SVMs is available in the following tutorial:
  - C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, Vol. 2(2), pp. 121-167 1998.