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
    \[ R_{\text{emp}}(\alpha) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i; \alpha)) \]
- Actual risk (= Expected risk)
  - Expectation of the error on all data.
    \[ R(\alpha) = \int L(y, f(x; \alpha)) dP_{X,Y}(x, y) \]
  - \( P_{X,Y}(x, y) \) is the probability distribution of \((x, y)\).
    - 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
    \[ R(\alpha) \cdot R_{\text{emp}}(\alpha) + \epsilon(N, p^*, h) \]
  - where
    - \( N \): number of training examples
    - \( p^* \): probability that the bound is correct
    - \( h \): capacity of the learning machine (“VC-dimension”)

Recap: VC Dimension

- 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 \( \ell! \) 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 \( \{f(\alpha)\} \).
Interpretation as a two-player game
- Opponent's turn: He says a number $N$.
- Our turn: We specify a set of $N$ points $\{x_1, \ldots, x_N\}$.
- Opponent's turn: He gives us a labeling $\{y_1, \ldots, y_N\} \in \{0,1\}^N$.
- Our turn: We specify a function $f(x)$ 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

- Intuitive feeling (unfortunately wrong)
  - The VC dimension has a direct connection with the number of parameters.
- Counterexample
  $$f(x; \alpha) = g(\sin(\alpha x))$$
  $$g(x) = \begin{cases} 1, & x > 0 \\ -1, & x \leq 0 \end{cases}$$
  - Just a single parameter $\alpha$.
  - Infinite VC dimension
  - Proof: Choose $x_i = 10^{-i}$, $i = 1, \ldots, \ell$.
  - $$\alpha = \pi \left(1 + \sum_{i=1}^{\ell} \frac{(1-y_i)10^i}{2}\right)$$

### Upper Bound on the Risk

- Important result (Vapnik 1979, 1995)
  - With probability $(1-\eta)$, the following bound holds
    $$R(\alpha) \cdot R_{emp}(\alpha) + \sqrt{\frac{h(\log(2N/h) + 1) - \log(\eta/4)}{N}}$$
    “VC confidence”
  - This bound is independent of $P_{X,Y}(x,y)$!
  - 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
    $$R(\alpha) \cdot R_{emp}(\alpha) + \epsilon(N,p^*,h)$$

### 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).
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**

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.

Support Vector Machine (SVM)

- Let’s first consider linearly separable data
  - \(N\) training data points \((x_i, y_i)\) for \(i = 1, \ldots, N\)
  - Target values \(y_i \in \{-1, 1\}\)
  - Hyperplane separating the data
    \[ w^T x + b = 0 \]

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

Support Vector Machine (SVM)

- Since the data is linearly separable, there exists a hyperplane with
  \[ w^T x + b \geq 1 \quad \text{for} \quad t_n = +1 \]
  \[ w^T x + b \leq -1 \quad \text{for} \quad t_n = -1 \]
- Combined in one equation, this can be written as
  \[ t_n (w^T x_n + b) \geq 1 \quad \forall t_n \]
  - Canonical representation of the decision hyperplane.
- The equation will hold exactly for the points on the margin
  \[ t_n (w^T x_n + b) = 1 \]
  - By definition, there will always be at least one such point.

Support Vector Machine (SVM)

- We can choose \(w\) such that
  \[ w^T x_n + b = +1 \quad \text{for one} \quad t_n = +1 \]
  \[ w^T x_n + b = -1 \quad \text{for one} \quad t_n = -1 \]
- The distance between those two hyperplanes is then the margin
  \[ d_- = d_+ = \frac{1}{||w||} \]
  \[ d_- + d_+ = \frac{2}{||w||^2} \]

\Rightarrow We can find the hyperplane with maximal margin by minimizing \(||w||^2\).
Support Vector Machine (SVM)

- Optimization problem
  - Find the hyperplane satisfying
    \[ \arg \min_{w, b} \frac{1}{2}||w||^2 \]
    under the constraints
    \[ t_n(w^Tx_n + b) \geq 1 \quad \forall n \]
  - 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 \( K(x) \) subject to constraints \( f(x) = 0 \).
  - Example: we want to get as close as possible, but there is a fence.
  - How should we move?
    \[ f(x) = 0 \]
    \[ f(x) < 0 \]
  - Optimize
    \[ \max_{x, \lambda} L(x, \lambda) = K(x) + \lambda f(x) \]
    \[ \frac{\partial L}{\partial x} = \nabla K + \lambda f(x) = 0 \]
  - Two cases
    - Solution lies on boundary
      \( f(x) = 0 \) for some \( \lambda > 0 \)
    - Solution lies inside \( f(x) > 0 \)
    \( \lambda f(x) = 0 \)
  - In both cases
    \( \lambda f(x) = 0 \)

SVM - Lagrangian Formulation

- Find hyperplane minimizing \( ||w||^2 \) under the constraints
  \[ t_n(w^T x_n + b) - 1 \geq 0 \quad \forall n \]
- Lagrangian formulation
  - Introduce positive Lagrange multipliers: \( \alpha_n \geq 0 \quad \forall n \)
  - Minimize Lagrangian ("primal form")
    \[ L(w, b, \alpha) = \frac{1}{2} ||w||^2 - \sum_{n=1}^{N} \alpha_n \{ t_n(w^T x_n + b) - 1 \} \]
    - I.e., find \( w, b, \) and \( \alpha \) such that
      \[ \frac{\partial L}{\partial w} = 0 \Rightarrow \sum_{n=1}^{N} \alpha_n t_n x_n = 0 \]
      \[ \frac{\partial L}{\partial b} = 0 \Rightarrow w = \sum_{n=1}^{N} \alpha_n t_n x_n \]
Lagrangian primal form

\[ L_p = \frac{1}{2} \|w\|^2 - \sum_{n=1}^{N} a_n \left( t_n (w^T x_n + b) - 1 \right) \]

\[ = \frac{1}{2} \|w\|^2 - \sum_{n=1}^{N} a_n (t_n y(x_n) - 1) \]

The solution of \( L_p \) needs to fulfill the KKT conditions

- Necessary and sufficient conditions
  - \( a_n \geq 0 \)
  - \( t_n y(x_n) - 1 \geq 0 \)
  - \( f(x) \geq 0 \)
  - \( \lambda(x) = 0 \)

\[ \text{KKT:} \begin{cases} a_n \geq 0 \\ t_n y(x_n) - 1 \geq 0 \\ f(x) \geq 0 \\ \lambda(x) = 0 \end{cases} \]

Solution for the hyperplane

- To define the decision boundary, we still need to know \( b \).
- Observation: any support vector \( x_n \) satisfies

\[ t_n y(x_n) = t_n \left( \sum_{m \in S} a_m t_m x_n^T x_m + b \right) = 1 \]

\[ \text{KKT:} \begin{cases} f(x) \geq 0 \end{cases} \]

- Using \( t_n^2 = 1 \), we can derive:

\[ b = t_n - \sum_{m \in S} a_m t_m x_n^T x_m \]

- In practice, it is more robust to average over all support vectors:

\[ b = \frac{1}{NS} \sum_{n \in S} \left( t_n - \sum_{m \in S} a_m t_m x_n^T x_m \right) \]

Dual Formulation

\[ L_p = \frac{1}{2} \|w\|^2 - \sum_{n=1}^{N} a_n \left( t_n (w^T x_n + b) - 1 \right) \]

\[ = \frac{1}{2} \|w\|^2 - \sum_{n=1}^{N} a_n t_n w^T x_n + b N \sum_{n=1}^{N} a_n t_n + \sum_{n=1}^{N} a_n \]

Using the constraint \( \sum_{n=1}^{N} a_n t_n = 0 \), we obtain

\[ \frac{\partial L_p}{\partial b} = 0 \]

\[ L_p = \frac{1}{2} \|w\|^2 - \sum_{n=1}^{N} a_n t_n w^T x_n + \sum_{n=1}^{N} a_n \]

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!

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 \( O(M^3) \).
  - Here: \( D \) variables \( \Rightarrow O(D^3) \)
  - Problem: scaling with high-dim. data (“curse of dimensionality”)
Using the constraint \( t_n = \frac{1}{k} + k \), we obtain

\[
L_p = \frac{1}{2} \|w\|^2 - \sum_{n=1}^{N} a_n t_n w^T x_n + \sum_{n=1}^{N} a_n.
\]

Applying \( \frac{\partial L_p}{\partial w} = 0 \) and again using

\[
\frac{1}{2} w^T w = \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m t_n t_m (x_m^T x_n) + \sum_{n=1}^{N} a_n.
\]

Inserting this, we get the Wolfe dual

\[
L_d(a) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m t_n t_m (x_m^T x_n).
\]

The hyperplane is given by the \( N \) support vectors:

\[
w = \sum_{n=1}^{N} a_n t_n x_n.
\]

SVM - Non-Separable Data

- Non-separable data
  - I.e. the following inequalities cannot be satisfied for all data points
    \[
    w^T x_n + b \geq +1 \quad \text{for} \quad t_n = +1
    \]
    \[
    w^T x_n + b \cdot -1 \quad \text{for} \quad t_n = -1
    \]
  - Instead use
    \[
    w^T x_n + b \geq +1 - \xi_n \quad \text{for} \quad t_n = +1
    \]
    \[
    w^T x_n + b \cdot -1 + \xi_n \quad \text{for} \quad t_n = -1
    \]
  - with "slack variables" \( \xi_n \geq 0 \quad \forall n \)
**SVM - Soft-Margin Classification**

- Slack variables
  - One slack variable \( \xi_i \geq 0 \) for each training data point.
- Interpretation
  - \( \xi_i = 0 \) for points that are on the correct side of the margin.
  - \( \xi_i = y_i (x_i \cdot w) \) for all other points (linear penalty).
- We do not have to set the slack variables ourselves! ⇒ They are jointly optimized together with \( w \).

**SVM - Non-Separable Data**

- **Separable data**
  - Minimize \( \frac{1}{2} \|w\|^2 \)
- **Non-separable data**
  - Minimize \( \frac{1}{2} \|w\|^2 + \sum_{i=1}^{N} \xi_i \)

**New SVM Primal: Optimize**

\[
L_p = \frac{1}{2} \|w\|^2 + C \sum_{n=1}^{N} \xi_n - \sum_{n=1}^{N} a_n (t_n y_i (x_i \cdot w) - 1 + \xi_n) - \sum_{n=1}^{N} \mu_n \xi_n
\]

**KKT conditions**

\[
\begin{align*}
a_n & \geq 0 \quad \mu_n \geq 0 & \text{KKT:} & \lambda \geq 0 \\
\sum_{n=1}^{N} t_n y_i (x_i \cdot w) - 1 + \xi_n & \geq 0 & \xi_n & \geq 0 & f(x) & \geq 0 \\
a_n (t_n y_i (x_i \cdot w) - 1 + \xi_n) & = 0 & \mu_n \xi_n & = 0 & \lambda f(x) & = 0
\end{align*}
\]

**New SVM Dual: Maximize**

\( L_d(a) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m t_n t_m (x_i^T x_m) \)

under the conditions

\[
\begin{align*}
0 & \cdot a_n \cdot C & \text{This is all that changed!} \\
\sum_{n=1}^{N} a_n t_n & = 0 \\
\end{align*}
\]

**New Solution**

- Solution for the hyperplane
  - Computed as a linear combination of the training examples
  
  \[
  w = \sum_{n=1}^{N} a_n t_n x_n
  \]
  - Again sparse solution: \( a_n = 0 \) for points outside the margin.
  ⇒ The slack points with \( \xi_i > 0 \) are now also support vectors!
  - Compute \( b \) by averaging over all \( N_M \) points with \( 0 < a_n < C \):
  
  \[
  b = \frac{1}{N_M} \sum_{m \in M} \left( t_n - \sum_{m \in M} a_m t_m x_m^T x_n \right)
  \]

**Interpretation of Support Vectors**

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

- A more in-depth introduction to SVMs is available in the following tutorial: