Recap: Classification as Dim. Reduction

- Classification as dimensionality reduction
  - Interpret linear classification as a projection onto a lower-dim. space.
  - $y = w^T x$

- Learning problem: Try to find the projection vector $w$ that maximizes class separation.

Recap: Fisher’s Linear Discriminant Analysis

- Maximize distance between classes
- Minimize distance within a class
- Criterion: $J(w) = \frac{w^T S_B w}{w^T S_W w}$
  - $S_B$ = between-class scatter matrix
  - $S_W$ = within-class scatter matrix
- The optimal solution for $w$ can be obtained as:
  $$w \propto S_B^{-1}(m_2 - m_1)$$
- Classification function:
  $$y(x) = w^T x + w_0 \geq 0$$

Recap: Probabilistic Discriminative Models

- Consider models of the form
  $$p(C_i|\phi) = y(\phi) = \sigma(w^T \phi)$$
  with
  $$p(C_2|\phi) = 1 - p(C_1|\phi)$$
- This model is called logistic regression.
- Properties
  - Probabilistic interpretation
  - But discriminative method: only focus on decision hyperplane
  - Advantageous for high-dimensional spaces, requires less parameters than explicitly modeling $p(\phi|C_i)$ and $p(C_i)$.

Recap: Logistic Regression

- Let’s consider a data set $\{\phi_n, t_n\}$ with $n = 1, \ldots, N$, where $\phi_n = \phi(x_n)$ and $t_n \in \{0, 1\}$, $t = (t_1, \ldots, t_N)^T$.
- With $y_n = p(C_t|\phi_n)$, we can write the likelihood as
  $$p(t|w) = \prod_{n=1}^{N} y_n^{t_n} (1 - y_n)^{1-t_n}$$
- Define the error function as the negative log-likelihood
  $$E(w) = -\ln p(t|w)$$
  $$= -\sum_{n=1}^{N} \left( t_n \ln y_n + (1 - t_n) \ln(1 - y_n) \right)$$
- This is the so-called cross-entropy error function.
Recap: Iterative Methods for Estimation

- Gradient Descent (1st order)
  \[ w^{(r+1)} = w^{(r)} - \eta \nabla E(w)|_{w^{(r)}} \]
  - Simple and general
  - Relatively slow to converge, has problems with some functions

- Newton-Raphson (2nd order)
  \[ w^{(r+1)} = w^{(r)} - \eta \nabla^2 E(w)|_{w^{(r)}} \]
  where \( \nabla^2 E(w) \) is the Hessian matrix, i.e. the matrix of second derivatives.
  - Local quadratic approximation to the target function
  - Faster convergence

Recap: Iteratively Reweighted Least-Squares (IRLS)

- Update equations
  \[
  w^{(r+1)} = w^{(r)} - (\Phi^T R \Phi)^{-1} \Phi^T (y - t) \\
  \quad = (\Phi^T R \Phi)^{-1} \left\{ \Phi^T \Phi w^{(r)} - \Phi^T (y - t) \right\} \\
  \quad = (\Phi^T R \Phi)^{-1} \Phi^T R z \\
  \text{with } z = \Phi w^{(r)} - R^{-1} (y - t)
  \]
- Very similar form to pseudo-inverse (normal equations)
  - But now with non-constant weighing matrix \( R \) (depends on \( w \)).
  - Need to apply normal equations iteratively.
  \( \Rightarrow \text{Iteratively Reweighted Least-Squares (IRLS)} \)

Topics of This Lecture

- Statistical Learning Theory
  - Generalization and overfitting
  - Empirical and actual risk
  - VC dimension
  - Structural Risk Minimization
- Linear Support Vector Machines (SVMs)
  - Linearly separable case
  - Lagrange multipliers
  - Lagrangian (primal) formulation
  - Dual formulation
  - Discussion

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!

Example: Linearly Separable Data

- Overfitting is often a problem with linearly separable data
  - Which of the many possible decision boundaries is correct?
  - All of them have zero error on the training set.
  - However, they will most likely result in different predictions on novel test data.
  - Different generalization performance

- How to select the classifier with the best generalization performance?

Cross-Validation

- Popular technique: Cross-Validation
  - Split the available data into training and validation sets.
  - Estimate the generalization error based on the error on the validation set.
  - Choose the model with minimal validation error.

- E.g. 5-fold cross-validation
  - Split
  - Repeat for all splits...
Statistical Learning Theory

- **Goal:** generalization ability
  - Choose the model that has the lowest probability of misclassification on all data, not just the training data.
- **Statistical learning theory**
  - Formal treatment of the question
  - "How can we control the generalization capability of a learning machine?"
  - Emphasis on theory in contrast to often-used heuristics.

Risk

- **Measuring the "optimality"**
  - Measure the optimality by the risk
  - Difficulty: how should the risk be estimated?
- **Practical way**
  - **Empirical risk** (measured on the training/validation set)
    \[ R_{\text{emp}}(\alpha) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i; \alpha)) \]
  - Example: quadratic loss function
    \[ R_{\text{emp}}(\alpha) = \frac{1}{N} \sum_{i=1}^{N} (y_i - f(x_i; \alpha))^2 \]

Summary: Risk

- **Actual risk**
  - Advantage: measure for the generalization ability
  - Disadvantage: in general, we don’t know \( P_{X,Y}(x, y) \)
- **Empirical risk**
  - Disadvantage: no direct measure of the generalization ability
  - Advantage: does not depend on \( P_{X,Y}(x, y) \)
  - We typically know learning algorithms which minimize the empirical risk.
  - Strong interest in connection between both types of risk.
Empirical Risk Minimization Principle

- Enforce conditions on learning machine
  - Necessary and sufficient condition: uniform convergence
    \[ p \left( \sup_{\alpha} \left| R(\alpha) - R_{\text{emp}}(\alpha) \right| > \epsilon \right) \rightarrow 0 \text{ as } N \rightarrow \infty \]
  - Compute the empirical risk
    - Based on training data \( \{(x_i, y_i)\}_{i=1}^N \)
      \[ R_{\text{emp}}(\alpha) = \frac{1}{N} \sum_{i=1}^N L(y_i, f(x_i; \alpha)) \]
    - Minimizing the empirical risk guarantees that we minimize the actual risk in the limit of \( N \rightarrow \infty \).

Statistical Learning Theory

- Idea
  - Compute an upper bound on the actual risk based on the empirical risk
    \[ R(\alpha) \leq R_{\text{emp}}(\alpha) + \epsilon(N, p^*, h) \]
    - Where
      \( N \): number of training examples
      \( p^* \): probability that the bound is correct
      \( h \): capacity of the learning machine ("VC-dimension")
- Side note:
  - (This idea of specifying a bound that only holds with a certain probability is explored in a branch of learning theory called "Probably Approximately Correct" or PAC Learning).

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 \( \{x_1, \ldots, x_N\} \).
  - Opponent's turn: He gives us a labeling \( \{x_1, \ldots, x_N\} \in \{0,1\}^N \)
  - 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

- 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 \( x_i = 10^{-i}, \ i = 1, \ldots, \ell \)
  \[ \alpha = \pi \left( 1 + \sum_{i=1}^{\ell} (1-y_i)10^i \right) \]

Upper Bound on the Risk

- Important result (Vapnik 1979, 1995)
  - With probability \( 1-\eta \), the following bound holds
  \[ R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\frac{h(\log(2N/h) + 1) - \log(\eta/4)}{N}} \]
  "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
  \[ R(\alpha) \leq R_{emp}(\alpha) + \epsilon(N,p^*, h) \]

Structural Risk Minimization

- Principle
  - Given a series of \( n \) models \( f_i(x; \alpha) \) with increasing VC dimension.
  - For each of the models: minimize empirical risk \( r_i \leq r_0 \leq \ldots \leq r_n \).
  - Choose the model that minimizes the upper bound (right-hand side of the equation).
  - This is in general not the model that minimizes the empirical risk.
- Remarks
  - This algorithm is formally justified.
  - The result is useful whenever the bound is indeed "tight".

Structural Risk Minimization

- How can we implement Structural Risk Minimization?
  \[ R(\alpha) \leq R_{emp}(\alpha) + \epsilon(N,p^*, h) \]
- 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).

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!
Support Vector Machine (SVM)

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

$$\sum_{i=1}^{N} t_i (w^T x_i + b) \geq 1 \quad \forall n$$

$\Rightarrow$ Canonical representation of the decision hyperplane.

• Combined in one equation, this can be written as
  $$t_i (w^T x_i + b) \geq 1 \quad \forall n$$
  - The equation will hold exactly for the points on the margin
  - $t_i (w^T x_i + b) = 1$
  - By definition, there will always be at least one such point.

• Optimization problem
  - Find the hyperplane satisfying
  $$\arg \min_{w,b} \frac{1}{2} \|w\|^2$$
  under the constraints
  $$t_i (w^T x_i + b) \geq 1 \quad \forall n$$

$\Rightarrow$ Quadratic programming problem with linear constraints.
  - Can be formulated using Lagrange multipliers.

• Who is already familiar with Lagrange multipliers?
  - Let’s look at a very current 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?
    - $f(x) = 0$
    - $f(x) > 0$
    - $f(x) < 0$
  - We want to maximize $\nabla f$
  - But we can only move parallel to the fence, i.e., along
    $\nabla f + \lambda \nabla K + \lambda \nabla f$
    with $\lambda \neq 0$. 

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?

$$f(x) = 0 \quad \Rightarrow \quad f(x) > 0$$

$$\Rightarrow \text{Optimize} \quad \max_{x, \lambda} L(x, \lambda) = K(x) + \lambda f(x)$$

Karush-Kuhn-Tucker (KKT) conditions: $\lambda \geq 0$

- Solution lies on boundary
  - $f(x) = 0$ for some $\lambda > 0$

- Solution lies inside $f(x) > 0$
  - $\lambda f(x) = 0$

In both cases

$\lambda f(x) = 0$

Example: There might be a hill from which we can see better…

In both cases

$\lambda f(x) = 0$

SVM - Lagrangian Formulation

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

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

- KKT:

$$a_n \geq 0$$

$$\lambda \geq 0$$

$$t_n (y(x_n) - 1) \geq 0$$

$$f(x) \geq 0$$

$$a_n (t_n y(x_n) - 1) = 0$$

$$\lambda f(x) = 0$$

SVM - Solution (Part 1)

- Solution for the hyperplane
  - Computed as a linear combination of the training examples
  $$w = \sum_{n=1}^{N} a_n t_n x_n$$

- Because of the KKT conditions, the following must also hold
  $$a_n (t_n (w^T x_n + b) - 1) = 0$$

- This implies that $a_n > 0$ only for training data points for which
  $$(t_n (w^T x_n + b) - 1) = 0$$

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

Recap: Lagrange Multipliers

- Problem
  - Now let’s look at constraints of the form $f(x) \geq 0$.
  - Example: There might be a hill from which we can see better…

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

- Two cases
  - Solution lies on boundary
    $$f(x) = 0$$
  - Solution lies inside $f(x) > 0$
  - Constraint inactive: $\lambda = 0$
  - 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: $a_n \geq 0 \quad \forall n$
  - Minimize Lagrangian (“primal form”)
    $$L(w, b, a) = \frac{1}{2} \|w\|^2 - \sum_{n=1}^{N} a_n \left\{ t_n (w^T x_n + b) - 1 \right\}$$

- I.e., find $w$, $b$, and $a$ such that
  $$\frac{\partial L}{\partial b} = 0 \quad \Rightarrow \quad \sum_{n=1}^{N} a_n t_n = 0$$
  $$\frac{\partial L}{\partial w} = 0 \quad \Rightarrow \quad w = \sum_{n=1}^{N} a_n t_n x_n$$
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 - Solution (Part 2)

- Solution for the hyperplane
  - To define the decision boundary, we still need to know \( b \).
  - Observation: any support vector \( x_n \) satisfies
  \[
  t_n y(x_n) = t_n \left( \sum_{m \in S} a_m t_m x_m^T x_n + b \right) = 1 \]
  - KKT: \( f(x) \geq 0 \)
  - Using \( t_n^2 = 1 \), we can derive:
  \[
  b = t_n - \sum_{m \in S} a_m t_m x_m^T x_n
  \]
  - In practice, it is more robust to average over all support vectors:
  \[
  b = \frac{1}{|S|} \sum_{n \in S} \left( t_n - \sum_{m \in S} a_m t_m x_m^T x_n \right)
  \]

SVM - Dual Formulation

- Improving the scaling behavior: rewrite \( L_p \) in a dual form
  \[
  L_p = \frac{1}{2} \|w\|^2 - \sum_{n=1}^{N} a_n \left( t_n (w^T x_n + b) - 1 \right)
  \]
  \[
  = \frac{1}{2} \|w\|^2 - \sum_{n=1}^{N} a_n t_n w^T x_n - \sum_{n=1}^{N} a_n + \sum_{n=1}^{N} a_n
  \]
  - Using the constraint \( \sum_{n=1}^{N} a_n t_n = 0 \), we obtain
  \[
  \frac{\partial L_p}{\partial b} = 0
  \]

- Applying \( \frac{1}{2} \|w\|^2 = \frac{1}{2} w^T w \) and again using \( w = \sum_{n=1}^{N} a_n t_n x_n \)
  \[
  \frac{1}{2} w^T w = \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m t_n t_m (x_n^T x_m)
  \]
  - Inserting this, we get the Wolfe dual
  \[
  L_d(a) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m t_n t_m (x_n^T x_m)
  \]
Perceptual and Sensory Augmented Computing

References and Further Reading

for many learning algorithms

0

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

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:

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

Slide adapted from Bernt Schiele

Christopher M. Bishop

A Tutorial on Support Vector Machines for Pattern Recognition

Springer, 2006


USPS benchmark

2.5% error: human performance

Different learning algorithms

16.2% error: Decision tree (C4.5)

5.9% error: (best) 2-layer Neural Network

4.1% error: Gaussian kernel (\( \sigma = 0.3, 291 \) support vectors)

(We will see those in the next lecture...)

Next lecture...

Next Lecture...

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.

Only looked at linear decision boundaries...

- This is not sufficient for many applications.
- Need to generalize the ideas to non-linear boundaries.

\[ \Rightarrow \text{Next Lecture...} \]

References and Further Reading

More information on SVMs can be found in Chapter 7.1 of Bishop's book.

Additional information about Statistical Learning Theory and a more in-depth introduction to SVMs are available in the following tutorial:


Machine Learning, Summer'11