Linear Discriminant Functions and Support Vector Machines

Biometrics
CSE 190
Lecture 11

Perceptron
Linear, threshold units

\[ \alpha(x_1, \ldots, x_n) = \begin{cases} 1 & \text{if } w_1 x_1 + \cdots + w_n x_n > \theta \\ -1 & \text{otherwise.} \end{cases} \]

\[ X_i : \text{inputs} \]
\[ W_i : \text{weights} \]
\[ \theta : \text{threshold} \]

How powerful is a perceptron?
Threshold = 0

Inverter

Boolean AND

Boolean OR

Boolean XOR

Concept Space & Linear Separability
Linear Separability

Training Perceptron

Perceptron Training Rule

Converges, if...

… training data linearly separable
… step size \( \eta \) sufficiently small
… no “hidden” units

Gradient Descent

Learn \( w_i \)'s that minimize squared error

\[ E(\vec{w}) = \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2 \]

\( D \) – training data
Perceptron Revisited: Linear Separators

- Binary classification can be viewed as the task of separating classes in feature space:

\[ w^T x + b > 0 \]
\[ w^T x + b < 0 \]

\[ f(x) = \text{sign}(w^T x + b) \]
Classification Margin
- Distance from example $\mathbf{x}_i$ to the separator is $w^T\mathbf{x}_i + b$.
- Training examples closest to the hyperplane are support vectors.
- Margin $\rho$ of the separator is the distance from the separator to support vectors.

Maximum Margin Classification
- Maximizing the margin is good according to intuition and PAC (Probably Approximately Correct) theory.
- Implies that only support vectors matter; other training examples are ignorable.

Linear SVM Mathematically
- Let training set $\{(\mathbf{x}_i, y_i)\}_{i=1,..n}, \mathbf{x}_i \in \mathbb{R}^d, y_i \in \{-1, 1\}$ be separated by a hyperplane with margin $\rho$. Then for each training example $(\mathbf{x}_i, y_i)$:
  - $w^T\mathbf{x}_i + b \leq \rho/2$ if $y_i = -1$
  - $w^T\mathbf{x}_i + b \geq \rho/2$ if $y_i = 1$
- For every support vector $\mathbf{x}_i$, the above inequality is an equality. After rescaling $w$ and $b$ by $\rho/2$ in the equality, we obtain that distance between each $\mathbf{x}_i$ and the hyperplane is $r = \frac{y_i(w^T\mathbf{x}_i + b)}{\|w\|}$.
- Then the margin can be expressed through (rescaled) $w$ and $b$ as: $\rho = 2r = \frac{2}{\|w\|}$.

Linear SVMs Mathematically (cont.)
- Then we can formulate the quadratic optimization problem:
  - Find $w$ and $b$ such that $\rho = \frac{2}{\|w\|}$ is maximized
  - and for all $(\mathbf{x}_i, y_i), i=1..n$: $y_i(w^T\mathbf{x}_i + b) \geq 1$
  - Which can be reformulated as:
  - Find $w$ and $b$ such that $\Phi(w) = \|w\|^2 - w^T\mathbf{w}$ is minimized
  - and for all $(\mathbf{x}_i, y_i), i=1..n$: $y_i(w^T\mathbf{x}_i + b) \geq 1$

Solving the Optimization Problem
- Need to optimize a quadratic function subject to linear constraints.
- Quadratic optimization problems are a well-known class of mathematical programming problems for which several (non-trivial) algorithms exist.
- The solution involves constructing a dual problem where a Lagrange multiplier $\alpha_i$ is associated with every inequality constraint in the primal (original) problem.
- Find $\alpha_1,..\alpha_n$ such that:
  - $\Phi(w) = \sum\alpha_i - \frac{1}{2}\sum\alpha_i\alpha_jy_iy_j\mathbf{x}_i^T\mathbf{x}_j$ is maximized and
  - $\sum\alpha_i = 0$
  - $\alpha_i \geq 0$ for all $\alpha_i$

The Optimization Problem Solution
- Given a solution $\alpha_1,..\alpha_n$ to the dual problem, solution to the primal is:
  - $w = \sum\alpha_iy_i\mathbf{x}_i, b = y_i - \sum\alpha_iy_i\mathbf{x}_i^T\mathbf{x}_k$ for any $\alpha_k > 0$
- Each non-zero $\alpha_i$ indicates that corresponding $\mathbf{x}_i$ is a support vector.
- Then the classifying function is (note that we don’t need $w$ explicitly):
  - $f(x) = \sum\alpha_iy_i\mathbf{x}_i^T\mathbf{x} + b$
- Notice that it relies on an inner product between the test point $\mathbf{x}$ and the support vectors $\mathbf{x}_i$ – we will return to this later.
- Also keep in mind that solving the optimization problem involved computing the inner products $\mathbf{x}_i^T\mathbf{x}_k$ between all training points.
Soft Margin Classification

- What if the training set is not linearly separable?
- Slack variables $\xi_i$ can be added to allow misclassification of difficult or noisy examples, resulting in a margin called "soft.

Soft Margin Classification Mathematically

- The old formulation:
  \[ \text{Find } w \text{ and } b \text{ such that } \Phi(w) = w^T w \text{ is minimized} \]
  \[ \text{and for all } (x_i, y_i), i = 1..n : \quad y_i(w^T x_i + b) \geq 1 \]

- Modified formulation incorporates slack variables:
  \[ \text{Find } w \text{ and } b \text{ such that } \Phi(w) = w^T w + C \sum \xi_i \text{ is minimized} \]
  \[ \text{and for all } (x_i, y_i), i = 1..n : \quad y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \]

- Parameter $C$ can be viewed as a way to control overfitting: it "trades off" the relative importance of maximizing the margin and fitting the training data.

Soft Margin Classification – Solution

- Dual problem is identical to separable case (would not be identical if the 2-norm penalty for slack variables $C \sum \xi_i^2$ was used in primal objective, we would need additional Lagrange multipliers for slack variables):
  \[ \text{Find } a_1, \ldots, a_n \text{ such that} \]
  \[ Q(a) = \sum a_i - \frac{1}{2} \sum a_i a_j y_i y_j x_i^T x_j \text{ is maximized and} \]
  \[ (1) \sum a_i y_i = 0 \]
  \[ (2) 0 \leq a_i \leq C \text{ for all } a_i \]

- Again, $x_i$ with non-zero $a_i$ will be support vectors.
- Solution to the dual problem is:
  \[ w = \sum a_i y_i x_i \]
  \[ b = \gamma(1 - \sum \xi_i) - \sum a_i y_i x_i^T x_k \text{ for any } k \text{ s.t. } a_k > 0 \]

Again, we don’t need to compute $w$ explicitly for classification:
\[ f(x) = \sum a_i y_i x_i^T x + b \]

Linear SVMs: Overview

- The classifier is a separating hyperplane.
- Most “important” training points are support vectors; they define the hyperplane.
- Quadratic optimization algorithms can identify which training points $x_i$ are support vectors with non-zero Lagrangian multipliers $a_i$.
- Both in the dual formulation of the problem and in the solution training points appear only inside inner products:

\[ f(x) = \sum a_i y_i x_i^T x + b \]

Non-linear SVMs

- Datasets that are linearly separable with some noise work out great:
  \[ \text{Datasets that are linearly separable with some noise work out great:} \]

- But what are we going to do if the dataset is just too hard?
  \[ \text{But what are we going to do if the dataset is just too hard?} \]

- How about… mapping data to a higher-dimensional space:
  \[ \text{How about… mapping data to a higher-dimensional space:} \]

Non-linear SVMs: Feature spaces

- General idea: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable:

Video Example:
http://www.youtube.com/watch?v=3lCbRZPrZA
The “Kernel Trick”

- The linear classifier relies on inner product between vectors $K(x_i,x_j)=x_i^T x_j$.
- If every datapoint is mapped into high-dimensional space via some transformation $\Phi$: $x \rightarrow \Phi(x)$, the inner product becomes:
  $K(x_i,x_j)=\Phi(x_i)^T \Phi(x_j)$.
- A kernel function is a function that is equivalent to an inner product in some feature space.  
- Example: 2-dimensional vectors $x=[x_1, x_2]$; let $K(x_i,x_j)=(1+ x_1 x_2)^2$.
  Need to show that $K(x_i,x_j)=\Phi(x_i)^T \Phi(x_j)$.
  $K(x_i,x_j)=(1+x_1^2)(1+x_2^2)=1+x_1^2+2 x_1 x_2+x_2^2+2 x_1 x_2+x_2^2$ = $1 x_1^2 \sqrt{2} x_1 x_2 \sqrt{2} x_2 x_2 \sqrt{2} x_2$ = $\Phi(x_i)^T \Phi(x_j)$, where $\Phi(x_i)=[1 x_1 x_2 x_1 x_2 x_2 x_2 x_2 x_2 x_2 x_2]$  
- Thus, a kernel function implicitly maps data to a high-dimensional space (without the need to compute each $\Phi(x)$ explicitly).

What Functions are Kernels?

- For some functions $K(x_i,x_j)$ checking that $K(x_i,x_j)=\Phi(x_i)^T \Phi(x_j)$ can be cumbersome.  
- Mercer’s theorem:  
  Every semi-positive definite symmetric function is a kernel  
- Semi-positive definite symmetric functions correspond to a semi-positive definite symmetric Gram matrix:

$$K=
\begin{pmatrix}
K(x_i,x_j) & K(x_i,x_j) & \cdots & K(x_i,x_j) \\
K(x_j,x_i) & K(x_j,x_j) & \cdots & K(x_j,x_j) \\
\vdots & \vdots & \ddots & \vdots \\
K(x_n,x_i) & K(x_n,x_j) & \cdots & K(x_n,x_n)
\end{pmatrix}$$

Examples of Kernel Functions

- Linear: $K(x_i,x_j)=x_i^T x_j$, where $\Phi(x)$ is $x$ itself.
- Polynomial of power $p$: $K(x_i,x_j)=(1+x_i^T x_j)^p$  
  - Mapping $\Phi$: $x \rightarrow \Phi(x)$, where $\Phi(x)$ has $\binom{p+d}{d}$ dimensions.
- Gaussian (radial-basis function): $K(x_i,x_j)=e^{\frac{-|x_i-x_j|^2}{\sigma^2}}$  
  - Mapping $\Phi$: $x \rightarrow \Phi(x)$, where $\Phi(x)$ is infinite-dimensional. Every point is mapped to a function (a Gaussian). Combination of functions for support vectors is the separator.
- Higher-dimensional space still has intrinsic dimensionality $d$ (the mapping is not onto), but linear separators in it correspond to non-linear separators in original space.

Non-linear SVMs Mathematically

- Dual problem formulation:
  Find $a_i\ldots a_n$ such that
  $$Q(a)=\sum_{i,j=1}^{n} a_i a_j y_i y_j K(x_i,x_j) \text{ is maximized and}$$
  $$\sum_{i=1}^{n} a_i = 0$$
  $$\sum_{i=1}^{n} a_i y_i \geq 0 \text{ for all } y_i$$

- The solution is:
  $$\hat{y}(x) = \sum_{i,j} a_i a_j y_i y_j K(x_i,x_j) + b$$

- Optimization techniques for finding $a_i$’s remain the same!

SVM applications

- SVMs were originally proposed by Boser, Guyon and Vapnik in 1992 and gained increasing popularity in late 1990s.
- SVMs are currently among the best performers for a number of classification tasks ranging from text to genomic data.
- SVMs can be applied to complex data types beyond feature vectors (e.g. graphs, sequences, relational data) by designing kernel functions for such data.
- SVM techniques have been extended to a number of tasks such as regression [Vapnik et al. ’97], principal component analysis [Schölkopf et al. ’99], etc.
- Tuning SVMs remains a black art: selecting a specific kernel and parameters is usually done in a try-and-see manner.
- Implementations: See libsvm, svmlight, and others.