8. Kernel methods

- motivation
- kernel formulations
- kernel functions

Linear-in-parameters model

Linear-in-parameters model (in the notation of 133A, lecture 9)

$$\theta^T F(x) = \theta_1 f_1(x) + \theta_2 f_2(x) + \cdots + \theta_p f_p(x)$$

- x is an independent variable, not necessarily a vector
- $F(x)$ is a *feature map:* maps x to a p-vector of features (possibly redundant)

$$F(x) = \big(f_1(x), f_2(x), \ldots, f_p(x)\big)$$

- the function $\theta^T F(x)$ is linear in the parameters θ

Training set: N data points $x^{(1)}, \ldots, x^{(N)}$ define an $N \times p$ data matrix

$$A = \begin{bmatrix} F(x^{(1)})^T \\ F(x^{(2)})^T \\ \vdots \\ F(x^{(N)})^T \end{bmatrix}$$

Kernel methods

Kernel matrix

$$Q = AA^T$$

Q is $N \times N$ and symmetric positive semidefinite with entries

$$Q_{ij} = F(x^{(i)})^T F(x^{(j)}), \quad i, j = 1, \ldots, N$$

Kernel function

$$\kappa(x, y) = F(x)^T F(y)$$

in this notation, the entries of the kernel matrix are

$$Q_{ij} = \kappa(x^{(i)}, x^{(j)}), \quad i, j = 1, \ldots, N$$

Kernel methods

- algorithms that use kernel matrix Q and function $\kappa(x, y)$, avoid $F(x)$, A, $A^T A$
- of interest if $N \ll p$ (including extensions to infinite-dimensional feature maps)

Polynomial kernel

$\theta^T F(x)$ is a polynomial of degree d or less in n variables

- here we assume x is an n-vector
- dimension of $F(x)$ is extremely large unless n or d is small:

$$p = \begin{pmatrix} n+d \\ n \end{pmatrix} = \frac{(n+d)!}{n!\,d!}$$

- with appropriately scaled (or repeated) monomials as features in $F(x)$,

$$\kappa(x,y) = F(x)^T F(y) = (1 + x^T y)^d$$

(see 133A, lecture 12)

Model fitting by regularized least squares

an example of a kernel method was discussed in 133A, lecture 12

$$\text{minimize} \quad \|A\theta - b\|^2 + \lambda\|\theta\|^2$$

- we fit a model $\hat{f}(x) = \theta^T F(x)$ to data points $x^{(1)}, \ldots, x^{(N)}$, $y^{(1)}, \ldots, y^{(N)}$
- b is the N-vector with entries $y^{(1)}, \ldots, y^{(N)}$
- second objective $\lambda\|\theta\|^2$ is added to avoid over-fitting
- optimal solution is $\hat{f}(x) = \hat{\theta}^T F(x)$ where

$$\hat{\theta} = (A^T A + \lambda I)^{-1} A^T b$$

Kernel method for regularized least squares fitting

via the “push-through” identity the solution $\hat{\theta}$ can be written as

$$\hat{\theta} = (A^T A + \lambda I)^{-1} A^T b = A^T (AA^T + \lambda I)^{-1} b$$

- can be computed as $\hat{\theta} = A^T \hat{w}$ where

$$\hat{w} = (Q + \lambda I)^{-1} b, \qquad Q = AA^T \text{ is the kernel matrix}$$

- fitted model $\hat{\theta}^T F(x)$ can be evaluated using $\hat{w}$ and the kernel function:

$$\begin{aligned} \hat{f}(x) = \hat{\theta}^T F(x) = \hat{w}^T A F(x) &= \hat{w}^T \begin{bmatrix} \kappa(x^{(1)}, x) \\ \vdots \\ \kappa(x^{(N)}, x) \end{bmatrix} \\ &= \sum_{i=1}^{N} \hat{w}_i \kappa(x^{(i)}, x) \end{aligned}$$

this method only requires kernel matrix Q and kernel function κ, not A, F, or $A^T A$

Principal components

another example is principal component analysis of the $N \times p$ data matrix A

- compute the leading right singular vectors $v_1, \ldots, v_k$ of A:

$$A = \begin{bmatrix} F(x^{(1)})^T \\ F(x^{(2)})^T \\ \vdots \\ F(x^{(N)})^T \end{bmatrix} = \sum_{i=1}^{\mathrm{rank}(A)} \sigma_i u_i v_i^T$$

- in feature space $\mathbf{R}^p$, principal components are linear functions $v_i^T y$ of $y \in \mathbf{R}^p$
- evaluated at $y = F(x)$, principal components are nonlinear functions

$$v_1^T F(x), \quad v_2^T F(x), \quad \ldots, \quad v_k^T F(x)$$

- using $A^T u_i = \sigma_i v_i$ the principal components can be written as

$$\frac{1}{\sigma_1} u_1^T A F(x), \quad \frac{1}{\sigma_2} u_2^T A F(x), \quad \ldots, \quad \frac{1}{\sigma_k} u_k^T A F(x)$$

Kernel PCA

- find leading singular values, left singular vectors of A via eigendecomposition

$$AA^T = Q = \sum_{i=1}^{\mathrm{rank}(A)} \sigma_i^2 u_i u_i^T$$

- right singular vectors v_i are given by

$$v_i = \frac{1}{\sigma_i} A^T u_i, \quad i = 1, \ldots, \mathrm{rank}(A)$$

- p.c.'s can be computed from left singular vectors and kernel function:

$$v_i^T F(x) = \frac{1}{\sigma_i} u_i^T A F(x) = \frac{1}{\sigma_i} u_i^T \begin{bmatrix} \kappa(x^{(1)}, x) \\ \vdots \\ \kappa(x^{(N)}, x) \end{bmatrix}$$

this method only requires kernel matrix Q and kernel function κ, not A, F, or $A^T A$

Exercises

1. modify the method on page 8.6 to solve

$$\text{minimize} \quad \|A\theta - b\|^2 + \lambda \sum_{i=2}^{p} \theta_i^2,$$

   assuming the elements in the first column of A are all ones

2. principal component analysis is usually applied to the centered data matrix

$$A_{\mathrm{c}} = (I - \frac{1}{N}\mathbf{1}\mathbf{1}^T)A$$

   what changes in the method on page 8.8 if we are interested in

$$v_1^T F(x), \quad v_2^T F(x), \quad , \ldots, \quad v_k^T F(x)$$

   where $v_1, \ldots, v_k$ are leading right singular vectors of A_{c}?

Outline

- motivation
- **kernel formulations**
- kernel functions

A general class of model fitting problems

we consider optimization problems in which the variable θ enters in only two ways

1. terms in objective and constraints that depend on model predictions on data set

$$A\theta = \begin{bmatrix} F(x^{(1)})^T\theta \\ \vdots \\ F(x^{(N)})^T\theta \end{bmatrix}$$

2. terms in objective that penalize $\|\theta\|$, or upper bounds on $\|\theta\|$ in the constraints

these properties imply that we can restrict θ to the row space of A

- $A\theta$ only depends on component of θ in the row space of A
- adding a nonzero component from the nullspace of A would only increase $\|\theta\|$

in machine learning, this is known as the *representer theorem*

Examples

Regularized least squares

$$\text{minimize} \quad \|A\theta - b\|^2 + \lambda\|\theta\|^2$$

Principal component analysis

- first right singular vector v_1 of A is solution of

$$\begin{array}{ll} \text{maximize} & \|A\theta\| \\ \text{subject to} & \|\theta\| \leq 1 \end{array}$$

- ith right singular vector v_i, where $i \leq \operatorname{rank}(A)$, is the solution of

$$\begin{array}{ll} \text{maximize} & \|A\theta\| \\ \text{subject to} & v_j^T\theta = 0, \quad j = 1, \ldots, i-1 \\ & \|\theta\| \leq 1 \end{array}$$

constraints $v_j^T\theta = 0$ are equivalent to $u_j^T A\theta = 0$, since $\sigma_j v_j = A^T u_j$

Factorization of kernel matrix

we discuss one approach to exploit the "representer theorem" on page 8.10

- denote by r the rank of the kernel matrix: $r = \mathrm{rank}(AA^T) = \mathrm{rank}(A)$
- the kernel matrix $Q = AA^T$ can be factored as

$$Q = BB^T$$

  where B is $N \times r$ with linearly independent columns

- the matrix $C = B^\dagger A$ has orthonormal columns and satisfies

$$A = BC$$

  (proof on next page)

- the rows of C are an orthonormal basis for the row space of A

$$\mathrm{range}(C^T) = \mathrm{range}(A^T) = \mathrm{span}(F(x^{(1)}), \ldots, F(x^{(N)}))$$

Proof: $C = B^\dagger A$ has orthonormal rows and satisfies $A = BC$

- the columns of B are a basis for $\mathrm{range}(AA^T) = \mathrm{range}(A)$
- the matrix $BB^\dagger$ projects on $\mathrm{range}(A)$; in particular,

$$BC = BB^\dagger A = A$$

- C has orthonormal rows because

$$CC^T = B^\dagger AA^T(B^\dagger)^T = B^\dagger BB^T(B^\dagger)^T = I$$

Reformulation of model fitting problem

every θ can be decomposed in components in the row space and nullspace of A:

$$\theta = C^T w + v, \qquad Cv = 0$$

- the vector $A\theta$ of model predictions only depends on w, and not on v:

$$A\theta = (BC)(C^T w + v) = Bw$$

- for given w, the Euclidean norm of θ is minimized by setting $v = 0$:

$$\|\theta\|^2 = \|C^T w\|^2 + \|v\|^2 = \|w\|^2 + \|v\|^2$$

therefore we can set $\theta = C^T w$ in any problem of the type described on page 8.10

Change of variables

we make the substitution

$$\theta = C^T w = (B^\dagger A)^T w$$

- $A\theta$ is replaced by Bw
- $\|\theta\|$ is replaced by $\|w\|$
- the r-vector w replaces the p-vector variable θ (a large reduction if $N \ll p$)
- the model function is linearly parametrized by the optimal solution $\hat{w}$:

$$\hat{f}(x) = \hat{\theta}^T F(x) = \hat{w}^T B^\dagger A F(x) = \hat{w}^T B^\dagger \begin{bmatrix} \kappa(x^{(1)}, x) \\ \kappa(x^{(2)}, x) \\ \vdots \\ \kappa(x^{(N)}, x) \end{bmatrix}$$

this formulation only requires B (computed from Q) and κ, not A, $A^T A$, F, or C

Regularized least squares

$$\text{minimize} \quad \|A\theta - b\|^2 + \lambda\|\theta\|^2$$

- variable θ is a p-vector
- solution $\hat{\theta}$ parametrizes the fitted model $\hat{f}(x) = \hat{\theta}^T F(x)$

Kernel method: solve reformulated problem

$$\text{minimize} \quad \|Bw - b\|^2 + \lambda\|w\|^2$$

- $N \times r$ matrix B is full-rank factor of kernel matrix $Q = BB^T$
- variable w is an r-vector, where $r = \operatorname{rank}(Q) \leq N$
- from solution $\hat{w}$, we obtain fitted model

$$\hat{f}(x) = \hat{w}^T B^\dagger \begin{bmatrix} \kappa(x^{(1)}, x) \\ \vdots \\ \kappa(x^{(N)}, x) \end{bmatrix}$$

Approximation problems

model fitting with non-quadratic penalty function h

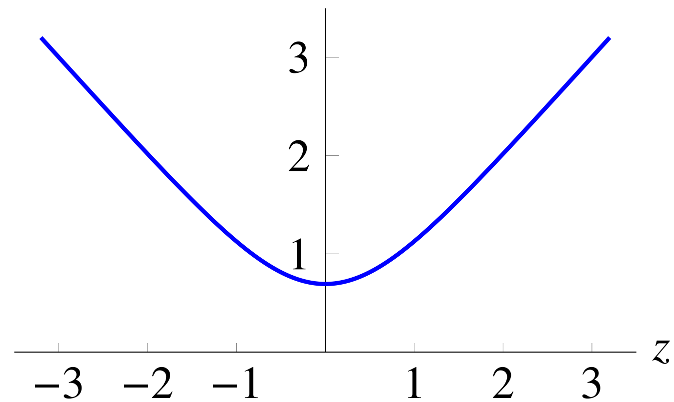
$$\text{minimize} \quad h(A\theta - b) + \lambda\|\theta\|^2$$

Examples

$h(u) = \|u\|_1$ or a smooth approximation

$$h(u) = \sum_{i=1}^{N} \phi(u_i)$$

$\phi(z) = \log(e^z + e^{-z})$



Kernel method

- solve problem in r-vector variable w (for example, using Newton's method)

$$\text{minimize} \quad h(Bw - b) + \lambda\|w\|^2$$

- no assumptions are made about h

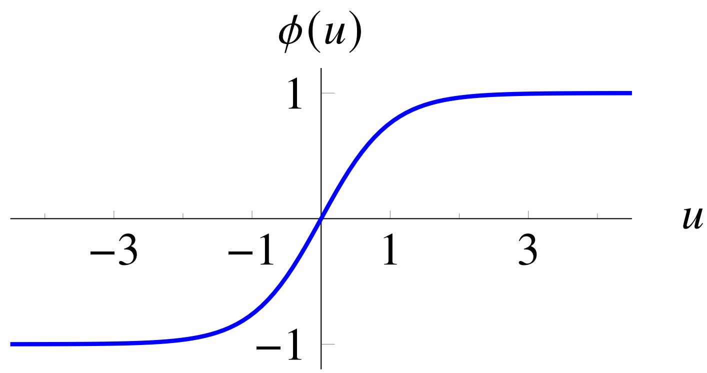
Nonlinear least squares example

another example from 133A (lecture 13)

$$\text{minimize} \quad \sum_{i=1}^{N} \left(\phi(F(x^{(i)})^T\theta) - y^{(i)}\right)^2 + \lambda\|\theta\|^2$$

- $y^{(i)} \in \{-1, 1\}$ are labels for two classes in a Boolean classification problem
- $\phi(u)$ is sigmoidal function (a smooth approximation of $\mathrm{sign}(u)$)

$$\phi(u) = \frac{e^u - e^{-u}}{e^u + e^{-u}}$$



Kernel method: solve the nonlinear least squares problem in r-vector variable w

$$\text{minimize} \quad \sum_{i=1}^{N} (\phi((Bw)_i) - y^{(i)})^2 + \lambda\|w\|^2$$

Boolean classification

the goal is to find a nonlinear decision function $\theta^T F(x)$ for a Boolean classifier:

$$\hat{f}(x) = 1 \quad \text{if } \theta^T F(x) > 0, \qquad \hat{f}(x) = -1 \quad \text{if } \theta^T F(x) < 0$$

Maximum-margin classifier

- given N examples $x^{(i)}$ with labels $y^{(i)} \in \{-1, 1\}$, find θ by solving

$$\begin{array}{ll} \text{minimize} & \|\theta\|^2 \\ \text{subject to} & \theta^T F(x^{(i)}) \geq 1 \quad \text{if } y^{(i)} = 1 \\ & \theta^T F(x^{(i)}) \leq -1 \quad \text{if } y^{(i)} = -1 \end{array}$$

- in matrix–vector form, if $d = (y^{(1)}, \ldots, y^{(N)})$ and A has rows $F(x^{(i)})^T$,

$$\begin{array}{ll} \text{minimize} & \|\theta\|^2 \\ \text{subject to} & \mathbf{diag}(d) A\theta \geq \mathbf{1} \end{array}$$

this is a *quadratic program*

Kernel formulation of maximum-margin classifier

solve a quadratic program in r-vector variable w:

$$\begin{array}{ll} \text{minimize} & \|w\|^2 \\ \text{subject to} & \mathbf{diag}(d)Bw \geq \mathbf{1} \end{array}$$

- B is computed from a kernel matrix factorization $Q = AA^T = BB^T$
- optimal solution $\hat{w}$ determines the nonlinear decision function $\tilde{f}(x) = \hat{\theta}^T F(x)$:

$$\tilde{f}(x) = \hat{w}^T B^{\dagger} \begin{bmatrix} \kappa(x^{(1)}, x) \\ \vdots \\ \kappa(x^{(N)}, x) \end{bmatrix}$$

- Boolean classifier returns

$$\hat{f}(x) = 1 \quad \text{if } \tilde{f}(x) > 0, \qquad \hat{f}(x) = -1 \quad \text{if } \tilde{f}(x) < 0$$

Support vector machine classifier

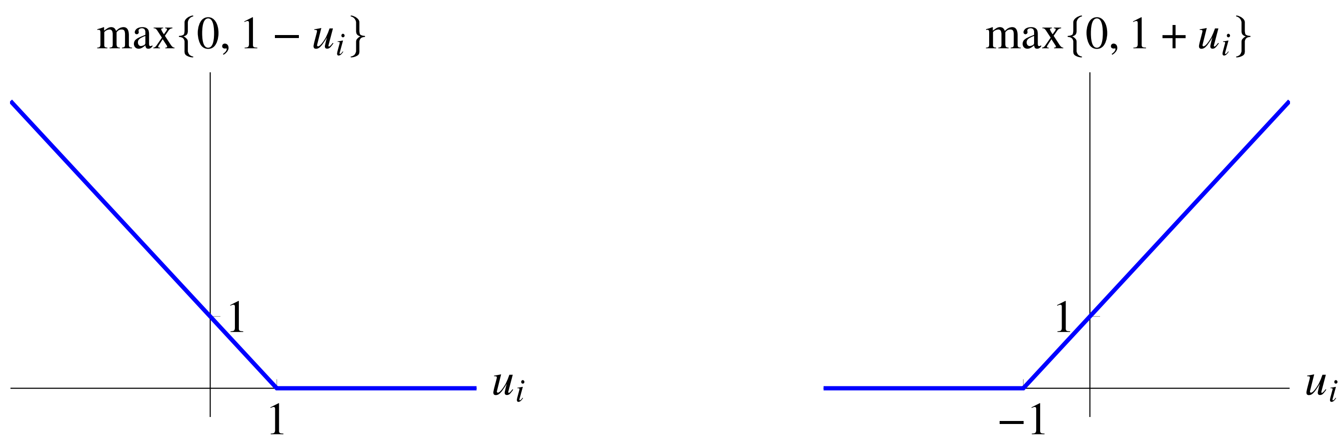
a variation on the maximum-margin classifier: compute θ from

$$\text{minimize} \quad \sum_{i=1}^{N} \max\left\{0, 1 - y^{(i)}(\theta^T F(x^{(i)}))\right\} + \lambda\|\theta\|^2$$

instead of imposing hard constraints

$$\theta^T F(x^{(i)}) \geq 1 \quad \text{if } y^{(i)} = 1, \qquad \theta^T F(x^{(i)}) \leq -1 \quad \text{if } y^{(i)} = -1$$

we impose a penalty on misclassified points:



Kernel formulation of support vector machine classifier

first term in support vector machine objective is a function of $A\theta$:

$$\text{minimize} \quad \sum_{i=1}^{N} \max\{0, 1 - y^{(i)}(A\theta)_i\} + \lambda\|\theta\|^2$$

Kernel formulation

$$\text{minimize} \quad \sum_{i=1}^{N} \max\{0, 1 - y^{(i)}(Bw)_i\} + \lambda\|w\|^2$$

- B is a full-rank factor of the kernel matrix $Q = BB^T$
- variable w is an r-vector
- from optimal $\hat{w}$ we directly find the decision function

$$\hat{\theta}^T F(x) = \hat{w}^T B^\dagger \begin{bmatrix} \kappa(x^{(1)}, x) \\ \vdots \\ \kappa(x^{(N)}, x) \end{bmatrix}$$

Outline

- motivation
- kernel formulations
- **kernel functions**

Kernel property

Kernel function: we require two properties of a kernel function

1. symmetry: $\kappa(x, y) = \kappa(y, x)$
2. for every finite set of points $x^{(1)}, \ldots, x^{(N)}$, the $N \times N$ matrix Q with entries

$$Q_{ij} = \kappa(x^{(i)}, x^{(j)}), \quad i, j = 1, \ldots, N$$

   is positive semidefinite

Properties: suppose κ_1, κ_2 are kernel functions

- $\kappa(x, y) = \alpha_1 \kappa_1(x, y) + \alpha_2 \kappa_2(x, y)$ is a kernel function, for all $\alpha_1, \alpha_2 \geq 0$
- $\kappa(x, y) = \kappa_1(x, y) \kappa_2(x, y)$ is a kernel function (see homework 2)

Examples

- polynomial kernel with degree d

$$\kappa(x, y) = (1 + x^T y)^d$$

- more generally,

$$\kappa(x, y) = q(x^T y)$$

where $q(t) = c_0 + c_1 t + \cdots + c_n t^n$ is a polynomial with nonnegative coefficients

- Gaussian kernel

$$\kappa(x, y) = \exp(-\frac{\|x - y\|^2}{2\sigma^2})$$

where $\sigma > 0$

From kernel to feature map

suppose κ is a function with the properties on page 8.23

- it can be shown that there exists a feature map F such that

$$\kappa(y, x) = \langle F(y), F(x) \rangle \quad \text{for all } x, y$$

  where $\langle \cdot, \cdot \rangle$ denotes an inner product

- however, in general the feature map $F(x)$ has infinite dimension

Finite-dimensional feature map

- for any given data set $x^{(1)}, \ldots, x^{(N)}$ we can construct a feature map F such that

$$\kappa(x^{(i)}, x) = F(x^{(i)})^T F(x) \quad \text{for all } x \text{ and for } i = 1, \ldots, N$$

- $F(x)$ can be chosen to have finite dimension $r = \operatorname{rank}(Q)$

Constructing a finite-dimensional feature map

we are given a kernel function κ and N points $x^{(1)}, \ldots, x^{(N)}$

- construct the $N \times N$ kernel matrix Q

$$Q_{ij} = \kappa(x^{(i)}, x^{(j)}), \quad i, j = 1, \ldots, N$$

- factor Q as $Q = BB^T$ with B an $N \times r$ matrix and $r = \operatorname{rank}(Q)$

- define the r-dimensional feature map

$$F(x) = B^\dagger \begin{bmatrix} \kappa(x^{(1)}, x) \\ \kappa(x^{(2)}, x) \\ \vdots \\ \kappa(x^{(N)}, x) \end{bmatrix}$$

on the next page we show that $F(x^{(i)})^T F(x) = \kappa(x^{(i)}, x)$ for all x and $i = 1, \ldots, N$

Proof

- the vectors $F(x^{(1)}), \ldots, F(x^{(N)})$ are the transposes of the rows of B:

$$F(x^{(i)}) = B^\dagger \begin{bmatrix} \kappa(x^{(1)}, x^{(i)}) \\ \vdots \\ \kappa(x^{(N)}, x^{(i)}) \end{bmatrix} = B^\dagger Q e_i = B^\dagger (BB^T) e_i = B^T e_i$$

- consider any x and define $d = \begin{bmatrix} \kappa(x^{(1)}, x) \\ \vdots \\ \kappa(x^{(N)}, x) \end{bmatrix}$

- by the kernel property the following matrix is positive semidefinite

$$\begin{bmatrix} Q & d \\ d^T & \kappa(x,x) \end{bmatrix} = \begin{bmatrix} BB^T & d \\ d^T & \kappa(x,x) \end{bmatrix}$$

- this implies that $d \in \mathrm{range}(B)$, *i.e.*, $BB^\dagger d = d$, and therefore

$$F(x^{(i)})^T F(x) = e_i^T BB^\dagger d = e_i^T d = \kappa(x^{(i)}, x)$$

References

- Bernhard Schölkopf and Alexander J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond* (2002).
- John Shawe-Taylor and Nello Cristianini, *Kernel Methods for Pattern Analysis* (2004).