Sapienza University of Rome

Master in Artificial Intelligence and Robotics
Master in Engineering in Computer Science

Machine Learning

A.Y. 2020/2021

Prof. L. Iocchi, F. Patrizi

9. Kernel Methods

L. Iocchi, F. Patrizi

Summary

- Kernel functions
- Kernelized linear models
- Kernelized SVM - classification
- Kernelized SVM - regression

References

C. Bishop. Pattern Recognition and Machine Learning. Chap. 6, Sect. 7.1

Kernels

So far:
Objects represented as fixed-length feature-vectors $\mathbf{x} \in \mathbb{R}^M$ or $\phi(\mathbf{x})$.

Issue:
what about objects with variable length or infinite dimensions?

Examples:

- strings
- trees
- image features
- time-series
- ...

Kernels

Approach:

use a *similarity measure* $k(\mathbf{x}, \mathbf{x}') \geq 0$ between the instances $\mathbf{x}, \mathbf{x}'$

$k(\mathbf{x}, \mathbf{x}')$ is called a *kernel* function.

Note: If we have $\phi(\mathbf{x})$ a possible choice is $k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^T \phi(\mathbf{x}')$.

Sapienza University of Rome, Italy - Machine Learning (2020/2021)

Kernels

Definition

Kernel function: a real-valued function $k(\mathbf{x}, \mathbf{x}') \in \mathbb{R}$, for $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$, where $\mathcal{X}$ is some abstract space.

Typically k is:

- symmetric: $k(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}', \mathbf{x})$
- non-negative: $k(\mathbf{x}, \mathbf{x}') \geq 0$.

Note: Not strictly required!

Input normalization

Input data in the dataset D must be normalized in order for the kernel to be a *similarity measure*.

Several types of normalizations:

- min-max $\quad \bar{x} = \frac{x - min}{max - min}$
  min, max: minimum and maximum input values in D
- normalization (standardization) $\quad \bar{x} = \frac{x - \mu}{\sigma}$
  μ mean and σ standard deviation of input values in D
- unit vector $\quad \bar{x} = \frac{x}{||x||}$

In the following, we assume the use of normalized input data.

Sapienza University of Rome, Italy - Machine Learning (2020/2021)

Kernel families

Linear

$$k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$$

Polynomial

$$k(\mathbf{x}, \mathbf{x}') = (\beta \mathbf{x}^T \mathbf{x}' + \gamma)^d, \; d \in \{2, 3, \dots\}$$

Radial Basis Function (RBF)

$$k(\mathbf{x}, \mathbf{x}') = \exp(-\beta |\mathbf{x} - \mathbf{x}'|^2)$$

Sigmoid

$$k(\mathbf{x}, \mathbf{x}') = \tanh(\beta \mathbf{x}^T \mathbf{x}' + \gamma)$$

Kernelized linear models

Consider a linear model $y(\mathbf{x}; \mathbf{w}) = \mathbf{w}^T \mathbf{x}$ with dataset $D = \{(\mathbf{x}_n, t_n)_{n=1}^N\}$

Minimize $J(\mathbf{w}) = (\mathbf{t} - \mathbf{X}\mathbf{w})^T (\mathbf{t} - \mathbf{X}\mathbf{w}) + \lambda \|\mathbf{w}\|^2$

N INPUT INSTANCES

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_N^T \end{bmatrix} \text{ design matrix,} \qquad \mathbf{t} = \begin{bmatrix} t_1 \\ \vdots \\ t_N \end{bmatrix} \text{ output vector}$$

Optimal solution

$\hat{\mathbf{w}} = (\mathbf{X}^T\mathbf{X} + \lambda I_N)^{-1} \mathbf{X}^T \mathbf{t} = \mathbf{X}^T \overbrace{(\mathbf{X}\mathbf{X}^T + \lambda I_N)^{-1} \mathbf{t}}^{\alpha}$,
with I_N the $N \times N$ identity matrix.

Kernelized linear models

Weights Vector

Let $\boldsymbol{\alpha} = (\mathbf{X}\mathbf{X}^T + \lambda I_N)^{-1} \mathbf{t}$,

then $\hat{\mathbf{w}} = \mathbf{X}^T \boldsymbol{\alpha} = \sum_{n=1}^N \alpha_n \mathbf{x}_n$.

Column vector

Hence we have $y(\mathbf{x}; \hat{\mathbf{w}}) = \boxed{\hat{\mathbf{w}}^T} \mathbf{x} = \sum_{n=1}^N \boxed{\alpha_n \mathbf{x}_n^T} \mathbf{x}$.

If we consider a linear kernel $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$, we can rewrite the model as

Linear combination of input instances

$$y(\mathbf{x}; \hat{\mathbf{w}}) = \sum_{n=1}^N \alpha_n k(\mathbf{x}_n, \mathbf{x})$$

with

$\boldsymbol{\alpha} = (K + \lambda I_N)^{-1} \mathbf{t}$, and $K = \mathbf{X}\mathbf{X}^T$ **Gram matrix** Input data for itself.

Kernelized linear models

Linear model with linear kernel $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$

$$y(\mathbf{x}; \boldsymbol{\alpha}) = \sum_{n=1}^{N} \alpha_n \mathbf{x}_n^T \mathbf{x}$$

Solution

$$\boldsymbol{\alpha} = (K + \lambda I_N)^{-1} \mathbf{t}$$

Gram matrix

$$K = \begin{bmatrix} \mathbf{x}_1^T \mathbf{x}_1 & \cdots & \mathbf{x}_1^T \mathbf{x}_N \\ \vdots & \ddots & \vdots \\ \mathbf{x}_N^T \mathbf{x}_1 & \cdots & \mathbf{x}_N^T \mathbf{x}_N \end{bmatrix}$$

Sapienza University of Rome, Italy - Machine Learning (2020/2021)

Kernelized linear models

Linear model with any kernel k

$$y(\mathbf{x}; \boldsymbol{\alpha}) = \sum_{n=1}^{N} \alpha_n k(\mathbf{x}_n, \mathbf{x})$$

Solution

$$\boldsymbol{\alpha} = (K + \lambda I_N)^{-1} \mathbf{t}$$

Gram matrix

$$K = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \cdots & k(\mathbf{x}_1, \mathbf{x}_N) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_N, \mathbf{x}_1) & \cdots & k(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}$$

Kernel trick

Kernel trick or *kernel substitution*

If input vector $\mathbf{x}$ appears in an algorithm only in the form of an inner product $\mathbf{x}^T\mathbf{x}'$, replace the inner product with some kernel $k(\mathbf{x}, \mathbf{x}')$.

function that transform input space in feature space (for certain instances).

- Can be applied to any $\mathbf{x}$ (even infinite size)
- No need to know $\phi(\mathbf{x})$
- Directly extend many well-known algorithms

Kernelized SVM - classification

In SVM, solution has the form:

$$\hat{\mathbf{w}} = \sum_{n=1}^{N} \alpha_n \mathbf{x}_n$$

Linear model (with linear kernel)

$$y(\mathbf{x}; \boldsymbol{\alpha}) = \mathrm{sign}\left(w_0 + \sum_{n=1}^{N} \alpha_n \mathbf{x}_n^T \mathbf{x}\right)$$

Kernel trick

$$y(\mathbf{x}; \boldsymbol{\alpha}) = \mathrm{sign}\left(w_0 + \sum_{n=1}^{N} \alpha_n k(\mathbf{x}_n, \mathbf{x})\right)$$

Note: w_0 also estimated from $\boldsymbol{\alpha}$

Kernelized SVM - classification

Lagrangian problem for kernelized SVM classification

$$\tilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} a_n \, a_m \, t_n \, t_m \, k(\mathbf{x}_n, \mathbf{x}_m)$$

Solution

$$a_n = \ldots$$

$$w_0 = \frac{1}{|SV|} \sum_{\mathbf{x}_i \in SV} \left(t_i - \sum_{\mathbf{x}_j \in S} a_j t_j k(\mathbf{x}_i, \mathbf{x}_j) \right)$$

Kernelized linear regression

Linear model for regression $y = \mathbf{w}^T \mathbf{x}$ and data set $D = \{(\mathbf{x}_n, t_n)_{n=1}^{N}\}$

Minimize the regularized loss function

$$J(\mathbf{w}) = \sum_{n=1}^{N} E(y_n, t_n) + \lambda \|\mathbf{w}\|^2,$$

where $y_n = \mathbf{w}^T \mathbf{x}_n$.

Kernelized linear regression

Consider $E(y_n, t_n) = (y_n - t_n)^2$: i.e., regularized linear regression.

Solution

$$\hat{\mathbf{w}} = (\mathbf{X}^T\mathbf{X} + \lambda I_N)^{-1}\mathbf{X}^T\mathbf{t} = \mathbf{X}^T\boldsymbol{\alpha}$$

$$\boldsymbol{\alpha} = (\mathbf{X}\mathbf{X}^T + \lambda I_N)^{-1}\mathbf{t}$$

Predictions are made using:

$$y(\mathbf{x}; \hat{\mathbf{w}}) = \sum_{n=1}^{N} \alpha_n \mathbf{x}_n^T \mathbf{x}.$$

Sapienza University of Rome, Italy - Machine Learning (2020/2021)

Kernelized linear regression

Apply the kernel trick:

$$y(\mathbf{x}; \hat{\mathbf{w}}) = \sum_{n=1}^{N} \alpha_n k(\mathbf{x}_n, \mathbf{x})$$

$$\boldsymbol{\alpha}=(K + \lambda I_N)^{-1}\mathbf{t}$$

Issue: computation of K requires $|D|^2$ operations and K is not sparse.

Kernelized SVM - regression

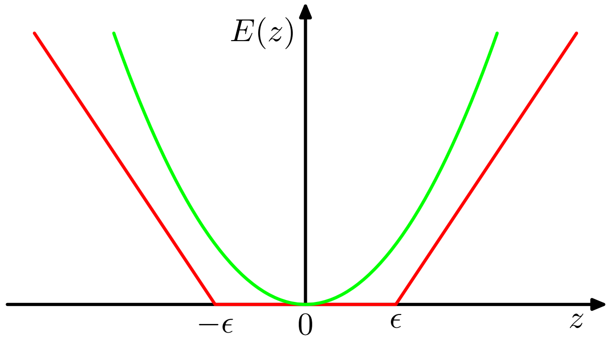
Consider

$$J(\mathbf{w}) = C \sum_{n=1}^{N} E_\epsilon(y_n, t_n) + \frac{1}{2}\|\mathbf{w}\|^2,$$

with C inverse of λ and an ϵ-insensitive error function:

$$E_\epsilon(y,t) = \begin{cases} 0 & \text{if } |y-t| < \epsilon \\ |y-t| - \epsilon & \textit{otherwise} \end{cases}.$$

Model admissible noise.



Not differentiable $\rightarrow$ difficult to solve.

Kernelized SVM - regression

Seen at 16:40 lesson

Introduce *slack variables* $\xi_n^+, \xi_n^- \geq 0$:

$$t_n \leq y_n + \epsilon + \xi_n^+$$
$$t_n \geq y_n - \epsilon - \xi_n^-$$

outside the tube.

Points inside the ϵ-tube $y_n - \epsilon \leq t_n \leq y_n + \epsilon \Rightarrow \xi_n = 0$

$\xi_n^+ > 0 \Rightarrow t_n > y_n + \epsilon$

$\xi_n^- > 0 \Rightarrow t_n < y_n - \epsilon$

with $y_n = y(\mathbf{x}_n; \mathbf{w})$

We are looking for the function that minimizes the error.

Kernelized SVM - regression

Loss function can be rewritten as:

minimize it.

$$J(\mathbf{w}) = C \sum_{n=1}^{N} (\xi_n^+ + \xi_n^-) + \frac{1}{2}\|\mathbf{w}\|^2,$$

subject to the constraints:

$$\begin{aligned} t_n &\leq y(\mathbf{x}_n;\mathbf{w}) + \epsilon + \xi_n^+ \\ t_n &\geq y(\mathbf{x}_n;\mathbf{w}) - \epsilon - \xi_n^- \\ \xi_n^+ &\geq 0 \\ \xi_n^- &\geq 0 \end{aligned}$$

This is a standard quadratic program (QP), can be "easily" solved.

Kernelized SVM - regression

Lagrangian problem

Lagrangian multipliers.

Maximize

$$\tilde{L}(\mathbf{a}, \mathbf{a}') = \ldots \sum_{n=1}^{N} \sum_{m=1}^{N} a_n a_m \ldots k(\mathbf{x}_n, \mathbf{x}_m) \ldots$$

from which we compute $\hat{a}_n$, $\hat{a}'_m$ (sparse values, most of them are zero) and

$$\hat{w}_0 = t_n - \epsilon - \sum_{m=1}^{N} (\hat{a}_m - \hat{a}'_m)\, k(\mathbf{x}_n, \mathbf{x}_m)$$

for some data point n such that $0 < a_n < C$

Prediction

only on the points outside the tube.

$$y(\mathbf{x}) = \sum_{n=1}^{N} (\hat{a}_n - \hat{a}'_n)\, k(\mathbf{x}, \mathbf{x}_n) + \hat{w}_0$$

Kernelized SVM - regression

From Karush-Kuhn-Tucker (KKT) condition (see Bishop Sect. 7.1.4)
Support vectors contribute to predictions

Points inside the tube have a,a'=0, outside the tube have or a or a'!=0

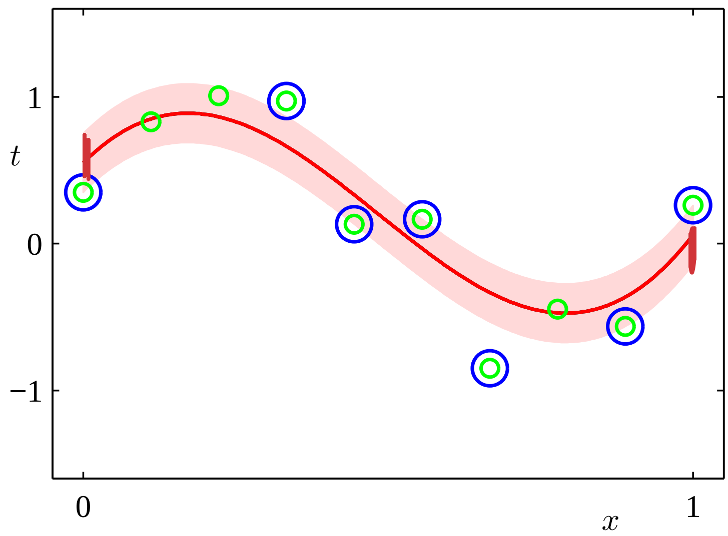
$\hat{a}_n > 0 \Rightarrow \epsilon + \xi_n + y_n - t_n = 0$
data point lies on or above upper boundary of the ϵ-tube

$\hat{a}'_n > 0 \Rightarrow \epsilon + \xi_n - y_n + t_n = 0$
data point lies on or below lower boundary of the ϵ-tube

All other data points inside the ϵ-tube have $\hat{a}_n = 0$ and $\hat{a}'_n = 0$ and thus do not contribute to prediction.

Kernelized SVM - regression

Example: support vectors and ϵ insensitive tube



Summary

- Kernel methods overcome difficulties in defining non-linear models
- Kernelized SVM is one of the most effective ML method for classification and regression
- Still requires model selection and hyper-parameters tuning