Sapienza University of Rome

Master in Artificial Intelligence and Robotics

Machine Learning

A.Y. 2024/2025

Prof. Luca Iocchi

Sapienza University of Rome, Italy - Machine Learning (2024/2025)

9. Kernel Methods

Luca Iocchi

Summary

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

References

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

Linear models

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

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} (\, \mathbf{w}^T\mathbf{x}_n - t_n \,)^2 \, + \lambda \, \mathbf{w}^T\mathbf{w}$$

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

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

Solution

Optimal solution by solving $\nabla E(\mathbf{w}) = 0$

$$\mathbf{w}^* = -\frac{1}{\lambda}\sum_{n=1}^{N}(\mathbf{w}^T\mathbf{x}_n - t_n)\,\mathbf{x}_n$$

Let $\alpha_n = -\frac{1}{\lambda}(\mathbf{w}^T\mathbf{x}_n - t_n)$

Then

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

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

Sapienza University of Rome, Italy - Machine Learning (2024/2025)

Solution

Optimal solution by solving $\nabla E(\mathbf{w}) = 0$

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

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

Then

$$\mathbf{w}^* = \mathbf{X}^T\boldsymbol{\alpha}$$

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

Linear models

Linear model

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

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 (2024/2025)

Kernel functions

Similarity function

$$k : X \times X \to \mathbb{R}$$

$k(\mathbf{x}, \mathbf{x}')$ similarity of input samples $\mathbf{x}$ and $\mathbf{x}'$

Linear kernel

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

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

IDENTY MATRIX

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

Gram matrix

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

num. cols and num. rows as num ber of samples in the dataset

Sapienza University of Rome, Italy - Machine Learning (2024/2025)

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 $k(\mathbf{x},\mathbf{x}')$.

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

Kernels

Extend linear models to non-linear functions.
Allow input 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 a *kernel* function.

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

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!

Sapienza University of Rome, Italy - Machine Learning (2024/2025)

Input normalization

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

→ GARANTEES NO BIAS (because of dimension)

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.

Kernel families

Linear

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

Polynomial

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

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 SVM - classification

In SVM, solution has the form:

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

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

$$a_n = \ldots$$

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

Sapienza University of Rome, Italy - Machine Learning (2024/2025)

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

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

REGULARIZATION PARAMETER ⇒ CALLED "HYPERPARAMETER".

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

Kernelized linear regression

Apply the kernel trick:

MEAN THE OPTIMAL OF ERROR FUNCTION

"is not sparse matrix"

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

!! REMEMBER !!
BIG DATASET is A PROBLEM
DIMENSION is N×N

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

REGULARIZATION PARAMETER

Issue: computation of K requires $> N^2$ operations and K is not sparse.

Kernelized SVM - regression

Consider

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

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

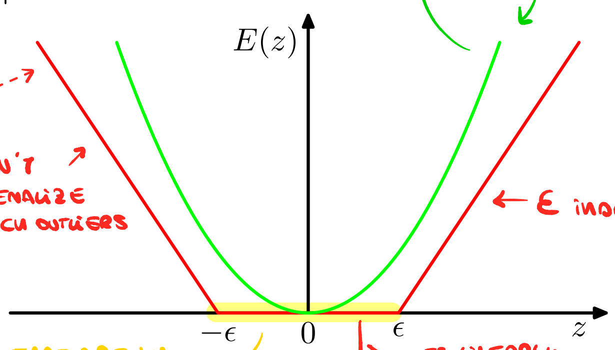
$$E_\epsilon(y,t) = \begin{cases} 0 & \text{if } |y-t| < \epsilon \\ |y-t| - \epsilon & \textit{otherwise} \end{cases}.$$

is an value that we put in order to relax limit of error

SHAPE OF QUADRATIC ERROR FUNCTION

WHAT WE SHOULD HAVE

WHAT WE REALLY USE



DOESN'T NOT PENALIZE TOO MUCH OUTLIERS

← ε insensitivity ERROR FUNCTION

WE USE THIS BECAUSE WE HAVE ALWAYS NOISE DATA

ERROR WILL BE =0 ⇐ POINT THAT ARE IN THE ε-TUBE

GOES LINEARLY AND NOT QUADRATIC

Not differentiable → difficult to solve.

Kernelized SVM - regression

↳ SIMILIAR SVM FOR CLASSIFICATION

DEFINE SOFT CONSTRAINS

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

HOW MUCH POINT ON N DATASET ARE OUT OF THIS INTERVAL

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

VALUE OF SAMPLE IN DATAS

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

Sapienza University of Rome, Italy - Machine Learning (2024/2025)

Kernelized SVM - regression

Loss function can be rewritten as:

ϵ IS A PARAMETER THAT CAN BE TUNE IF WE KNOW DATASET

=0 WHEN POINTS ARE INSIDE ϵ-TUBE

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

MEAN THAT NOT CONTRIBUTE

INVERSE OF $\lambda \Rightarrow \lambda^{-1}$

subject to the constraints:

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

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

Kernelized SVM - regression

Lagrangian problem

THIS PROBLE IS SPARSE

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

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

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

Sapienza University of Rome, Italy - Machine Learning (2024/2025)

Kernelized SVM - regression

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

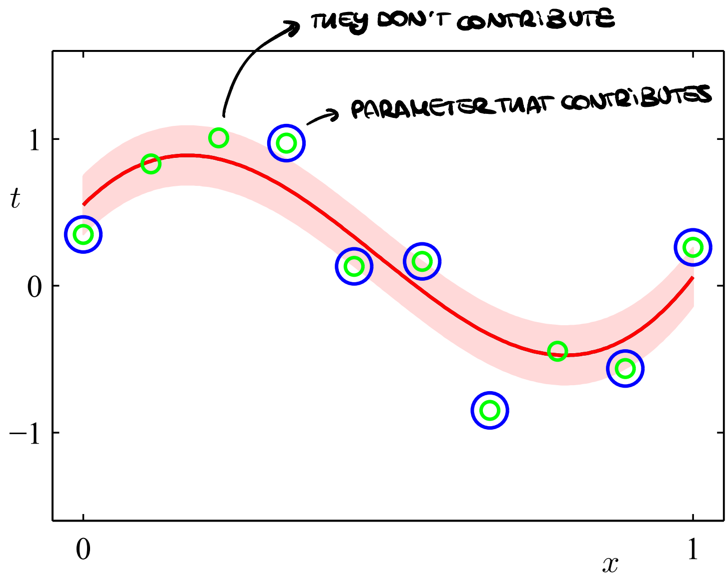
$\hat{a}_n > 0 \Rightarrow \epsilon + \xi_n + y_n - t_n = 0$
data point lies on or above upper boundary of the ϵ-tube

$\hat{a}'_n > 0 \Rightarrow \epsilon + \xi_n - y_n + t_n = 0$
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

EXERCISE 1

Consider a setting where the input space I is the set of finite strings over the characters a, b, c. Notice that input strings can be of different length. Given the dataset on the table on the right:

1. Identify the learning problem at hand, in particular the form of the target function.
2. Define a suitable kernelized linear model for this problem.
3. Define a kernel function suitable to measure the similarity of data sample for this problem.

x	t
b	1
a	2
ab	2
caba	4
abca	4
aabba	8
aaa	8
babaa	8
abcaaca	16
bcaaaca	16
abcbabacca	16

I : set of finite strings over the characters a, b, c

⇒ the position of a is not important, is relevant the frequency of a in dataset

```
def my_kernel (U,V)
    N = length (U)
    M = length (V)

    K = n.empty ((N,M))

    for i in range (0,N)

        for j in range (0,M)

            K[i,j] = k (U[i], V[j])

    return K
```

```
def k(u,v)
    d = abs ( count (u,"a") - count (v,"a"))

    # return 1.0/(d+1)**2
    return math.exp(-d)
```

CONSTRAINT FOR DEFINE A KERNEL:

1. must measure similarity
2. symmetric
3. must be positive

!! THERE IS ANOTHER SIMILAR EXERCISE ON CLASS ROOM WHERE WE HAVE TO DEFINE A KERNEL !!