Machine Learning and Intelligent Systems

Kernel Machines

Maria A. Zuluaga

Dec 1, 2023

EURECOM - Data Science Department

Table of contents

Kernel Machines

Kernel Machines

- We have introduced kernels and showed that they can be a very powerful tool
- The remaining question is how can we use them?
- Given the linear models that we have seen, how can we integrate the use of kernels in them?

Kernelizing an algorithm

To kernelize an algorithm, three steps are required:

Kernelizing an algorithm

To kernelize an algorithm, three steps are required:

1. Demonstrate that the solution lies in the span of the training data, i.e.

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i \mathbf{x}_i \tag{1}$$

for some α_i.

Kernelizing an algorithm

To kernelize an algorithm, three steps are required:

1. Demonstrate that the solution lies in the span of the training data, i.e.

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i \mathbf{x}_i \tag{1}$$

   for some α_i.

2. Rewrite the algorithm and the model so that the train/text inputs are only accessed via inner-products, i.e.

$$\mathbf{x}_i^T \mathbf{x}_j$$

Kernelizing an algorithm

To kernelize an algorithm, three steps are required:

1. Demonstrate that the solution lies in the span of the training data, i.e.

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i \mathbf{x}_i \tag{1}$$

   for some α_i.

2. Rewrite the algorithm and the model so that the train/text inputs are only accessed via inner-products, i.e.

$$\mathbf{x}_i^T \mathbf{x}_j$$

3. Define a kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$

Example 1: Kernel Linear Regression (OLS)

Recap

- The OLS solution minimizes the quadratic loss:

$$\arg\min_{\mathbf{w}} \sum_{i=1}^{N} (y_i - \mathbf{w}^T \mathbf{x}_i)^2$$

- It was closed form solution

$$\hat{\mathbf{w}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$

- The prediction of an unseen point is done via:

$$h(\mathbf{x}^*) = \hat{\mathbf{w}}^T \mathbf{x}^*$$

Step 1: Prove the solution is a linear combination of the inputs

- Let us express $\mathbf{w}$ as a linear combination of the inputs

$$\begin{aligned}\mathbf{w} &= \sum_{i=1}^{N} \alpha_i \mathbf{x}_i \\ &= \mathbf{X}^T \vec{\alpha}\end{aligned}$$

Step 1: Prove the solution is a linear combination of the inputs

- Let us express $\mathbf{w}$ as a linear combination of the inputs

$$\begin{aligned} \mathbf{w} &= \sum_{i=1}^{N} \alpha_i \mathbf{x}_i \\ &= \mathbf{X}^T \vec{\alpha} \end{aligned}$$

- Since the squared loss is a convex function, last lecture we demonstrated that such a solution exists
- It is obtained by applying gradient descent and initializing $\vec{\alpha} = 0$

Step 2: Rewrite in terms of inner products

- Kernelization of the prediction step is trivial:

$$\begin{aligned} h(\mathbf{x}^*) &= \hat{\mathbf{w}}^T \mathbf{x}^* \\ &= \sum_{i=1}^{N} \alpha_i \mathbf{x}_i^T \mathbf{x}^* \end{aligned}$$

- Kernelization is trivial as it requires to replace inner products by $k(\cdot, \cdot)$

$$h(\mathbf{x}^*) = \sum_{i=1}^{N} \alpha_i k(\mathbf{x}_i, \mathbf{x}^*)$$

Closed form solution of $\vec{\alpha}$

As with $\hat{\mathbf{w}}$, the kernelized version of OLS allows for a closed form solution for $\vec{\alpha}$

Closed form solution of $\vec{\alpha}$

As with $\hat{\mathbf{w}}$, the kernelized version of OLS allows for a closed form solution for $\vec{\alpha}$

Theorem:

Kernel Ordinary Least Squares has the solution:

$$\vec{\alpha} = \mathbf{K}^{-1}\mathbf{y}$$

Closed form solution of $\vec{\alpha}$

As with $\hat{\mathbf{w}}$, the kernelized version of OLS allows for a closed form solution for $\vec{\alpha}$

Theorem:

Kernel Ordinary Least Squares has the solution:

$$\vec{\alpha} = \mathbf{K}^{-1}\mathbf{y}$$

Proof:

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

Closed form solution of $\vec{\alpha}$

As with $\hat{\mathbf{w}}$, the kernelized version of OLS allows for a closed form solution for $\vec{\alpha}$

Theorem:

Kernel Ordinary Least Squares has the solution:

$$\vec{\alpha} = \mathbf{K}^{-1}\mathbf{y}$$

Proof:

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \rightarrow \mathbf{X}^T\vec{\alpha}$$

Closed form solution of $\vec{\alpha}$

As with $\hat{\mathbf{w}}$, the kernelized version of OLS allows for a closed form solution for $\vec{\alpha}$

Theorem:

Kernel Ordinary Least Squares has the solution:

$$\vec{\alpha} = \mathbf{K}^{-1}\mathbf{y}$$

Proof:

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \rightarrow \mathbf{X}^T\vec{\alpha}$$

Exercise: Do the same exercise to obtain kernel ridge regression

Example 2: Kernel Hard Margin Support Vector Machines

Recap: Hard SVM

The solution of the hard SVM required solving a constrained optimization problem:

$$\begin{aligned} \arg\min_{\mathbf{w}, w_0} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{subject to} \quad & \forall i \ y_i(\mathbf{w}^T\mathbf{x}_i + w_0) \geq 1 \end{aligned}$$

Prediction of a new point at testing is:

$$h(\mathbf{x}^*) = \text{sign}(\hat{\mathbf{w}}^T\mathbf{x}^* + \hat{w}_0)$$

Formulating kernel SVM requires some manipulations

Dual Form of an Optimization Problem

- An optimization problem has a **dual form** if the function to be optimized and the constraints are strictly convex
- If this is the case, the dual form is also a solution of the primal form of the optimization problem

Dual Form of an Optimization Problem

- An optimization problem has a **dual form** if the function to be optimized and the constraints are strictly convex
- If this is the case, the dual form is also a solution of the primal form of the optimization problem
- Usually the term dual problem refers to the Lagrangian dual problem but other dual problems are used

Dual Form of an Optimization Problem

- An optimization problem has a **dual form** if the function to be optimized and the constraints are strictly convex
- If this is the case, the dual form is also a solution of the primal form of the optimization problem
- Usually the term dual problem refers to the Lagrangian dual problem but other dual problems are used
- The **Lagrangian dual problem** is obtained by forming the Lagrangian of a minimization problem by using **non-negative Lagrange multipliers** to add the constraints to the objective function, and then solving for the primal variable values that minimize the original objective function

Back to SVM

The solution of the hard SVM required solving a constrained optimization problem:

$$\begin{aligned} \underset{\mathbf{w}, w_0}{\arg\min} \quad & \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{subject to} \quad & \forall i \;\; y_i(\mathbf{w}^T\mathbf{x}_i + w_0) \geq 1 \end{aligned}$$

Dual Representation of the Maximum Margin Problem

The optimization of the dual hard SVM problem can be expressed as:

$$\begin{aligned} \arg\max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \left(\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \right) \\ \text{subject to} \quad & \forall i \ \alpha_i \geq 0, \qquad \sum_{i=1}^{N} \alpha_i y_i = 0 \end{aligned} \tag{2}$$

Dual Representation of the Maximum Margin Problem

The optimization of the dual hard SVM problem can be expressed as:

$$\begin{aligned} \underset{\boldsymbol{\alpha}}{\arg\max} \quad & \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\left(\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j\right) \\ \text{subject to} \quad & \forall i \ \alpha_i \geq 0, \qquad \sum_{i=1}^{N} \alpha_i y_i = 0 \end{aligned} \tag{2}$$

In this setting, the original parameters are expressed in terms of α via

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i \tag{3}$$

Reminder: There is no need to estimate $\mathbf{w}$.

Kernelizing the Dual Form of the SVM

- The learning process can be expressed in terms of inner products without formally expressing $\mathbf{w}$

Kernelizing the Dual Form of the SVM

- The learning process can be expressed in terms of inner products without formally expressing $\mathbf{w}$
- $\mathbf{w}$ can be expressed as a linear combination of the inputs

Kernelizing the Dual Form of the SVM

- The learning process can be expressed in terms of inner products without formally expressing $\mathbf{w}$
- $\mathbf{w}$ can be expressed as a linear combination of the inputs
- Since the original problem is convex, it is possible to train an SVM using gradient descent for a given set of α

Kernelizing the Dual Form of the SVM

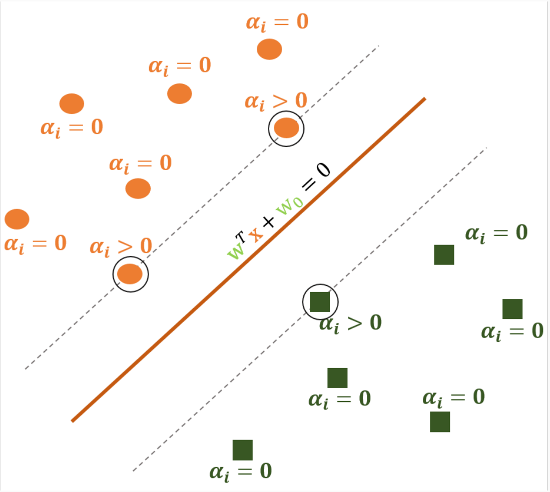
- The learning process can be expressed in terms of inner products without formally expressing $\mathbf{w}$
- $\mathbf{w}$ can be expressed as a linear combination of the inputs
- Since the original problem is convex, it is possible to train an SVM using gradient descent for a given set of α
- The dual representation is a quadratic problem can be solved through standard solvers

Kernelizing the Dual Form of the SVM

As we were able to express the learning process in terms of inner products, the kernelization is straightforward

$$
\begin{aligned}
\arg\max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\left(\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j) \right) \\
\text{subject to} \quad & \forall i \ \alpha_i \geq 0, \qquad \sum_{i=1}^{N} \alpha_i y_i = 0
\end{aligned} \tag{4}
$$

Interpretation: The Support Vectors



Support Vectors: Training points i whose $\alpha_i > 0$

Estimating w_0

A disadvantage of the dual formulation is that w_0 disappears from the optimization formulation.

Let us recall, from the primal formulation that if a given $\mathbf{x}$ is a support vector then $y_i(\hat{\mathbf{w}}\mathbf{x}_i + \hat{w}_0) = 1$.

If we replace accordingly in the previous term we obtain

$$\begin{aligned} y_i \left(\sum_{j=1}^{N} \alpha_j y_j \mathbf{x}_j^T \mathbf{x}_i + \hat{w}_0 \right) &= 1 \\ \sum_{j=1}^{N} \alpha_j y_j \mathbf{x}_j^T \mathbf{x}_i + \hat{w}_0 &= y_i \\ y_i - \sum_{j=1}^{N} \alpha_j y_j \mathbf{x}_j^T \mathbf{x}_i &= \hat{w}_0 \qquad (5) \end{aligned}$$

Estimating w_0

The obtained expression can be also kernelized

$$\hat{w}_0 = y_i - \sum_{j=1}^{N} \alpha_j y_j k(\mathbf{x}_j, \mathbf{x}_i) \tag{6}$$

How to estimate w_0 using the obtained expression?

- We know that only the support vectors have $\alpha > 0$
- It is possible to pick a random support vector to obtain w_0
- In practice it is better to obtain $\hat{w}_0$ by averaging over all points that are a support vector. **It is more stable.**

Prediction

The prediction of a new point is straightforward:

$$h(\mathbf{x}^*) = \text{sign}(\hat{\mathbf{w}}^T \mathbf{x}^* + \hat{w}_0) \tag{7}$$

It accounts to replacing Eqs. 3 ($\mathbf{w}$) and 5 ($\hat{w}_0$) in the term above.

Prediction

The prediction of a new point is straightforward:

$$h(\mathbf{x}^*) = \text{sign}(\hat{\mathbf{w}}^T \mathbf{x}^* + \hat{w}_0) \tag{7}$$

It accounts to replacing Eqs. 3 ($\mathbf{w}$) and 5 ($\hat{w}_0$) in the term above.

$$h(\mathbf{x}^*) = \text{sign}\left(\sum_{j=1}^{N} \alpha_j y_j k(\mathbf{x}_j, \mathbf{x}^*) + \left(y_i - \left(\sum_{j=1}^{N} \alpha_j y_j k(\mathbf{x}_j, \mathbf{x}_i)\right)\right)\right)$$

Example 3: Kernel Soft Margin Support Vector Machines

Recap: Soft SVM

The solution of the soft SVM required solving a constrained optimization problem:

$$\begin{aligned} \underset{\mathbf{w}, w_0}{\arg\min} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \xi_i \\ \text{subject to} \quad & \forall i \; y_i(\mathbf{w}^T\mathbf{x}_i + w_0) \geq 1 - \xi_i \\ & \forall i \; \xi_i \geq 0 \end{aligned}$$

That can also be presented in terms of the Hinge loss (unconstrained):

$$\underset{\mathbf{w}, w_0}{\arg\min} \; \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \max(1 - y_i(\mathbf{w}^T\mathbf{x}_i + w_0), 0)$$

Dual Form of the Soft SVM

$$
\begin{aligned}
\underset{\boldsymbol{\alpha}}{\arg\max} \quad & \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\left(\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)\right) \\
\text{subject to} \quad & \forall i \; 0 \leq \alpha_i \leq C, \\
& \sum_{i=1}^{N} \alpha_i y_i = 0
\end{aligned}
\tag{8}
$$

Dual Form of the Soft SVM

$$
\begin{aligned}
\arg\max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\left(\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)\right) \\
\text{subject to} \quad & \forall i \; 0 \leq \alpha_i \leq C, \\
& \sum_{i=1}^{N} \alpha_i y_i = 0
\end{aligned}
\tag{8}
$$

where

$$
\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i .
$$

Similarly, the expression for $\hat{w}_0$ remains the same as for the dual form of the hard SVM.

Optional: Exercise

- Study the Lagrange multiplies as a mechanism to transform the primal representation of an optimization problem into its dual one.
- Study the derivation of the dual form of the hard SVM
- Derive the dual form of the soft SVM
- **Deadline:** December 15th

Wrap-up

Wrap-up

- We presented the necessary steps to transform a given method to handle kernels
- We used the ordinary least squares as a first example
- We presented the primal and dual hard and soft SVM optimization problems (no proofs)
- We saw that the dual representation of the SVM provides a convenient interpretation of the support vectors

Key Concepts

- Kernel OLS
- Dual representation
- Kernel SVM

References

Further Reading and Useful Material

Source	Notes
Pattern Recognition and Machine Learning	Ch 7, appendix E
The Elements of Statistical Learning	Sec. 4.5, Ch 12