Machine Learning for Physics and Astronomy

Juan Rojo
VU Amsterdam & Theory group, Nikhef

Natuur- en Sterrenkunde BSc (Joint Degree), Honours Track
Lecture 7, 12/10/2020

Today's lecture

- Kernel methods and the Dual Representation
- Gaussian processes
- Guest lecture by **Dr. Tristan Bereau** on applications of machine learning in condensed matter

Kernel Methods

Kernel Methods

most of the ML models we have considered for regression and classification are based on the **parametric mapping** (linear or non-linear) between inputs and outputs

$$y\left(\boldsymbol{x}, \boldsymbol{\theta}\right)$$

were the **training examples are discarded** after the model parameters (or their posterior distribution) have been determined

here we introduce models where the training dataset is also used in the **prediction phase**

recall discussion of **Support Vector Machines:** we introduced a **feature space mapping**

Example of non-linear feature-space transformation

$\boldsymbol{x}$	$\boldsymbol{\phi}\left(\boldsymbol{x}\right)$
Height	***Height/Weight***
Weight	***Weight+Height***
Age	***Age * Weight***

Kernel Methods

most of the ML models we have considered for regression and classification are based on the **parametric mapping** (linear or non-linear) between inputs and outputs

$$y\left(\boldsymbol{x}, \boldsymbol{\theta}\right)$$

were the **training examples are discarded** after the model parameters (or their posterior distribution) have been determined

here we introduce models where the training dataset is also used in the **prediction phase**

the key ingredient of **kernel methods** is the **kernel function** evaluated at the training points

$$k\left(\boldsymbol{x}, \boldsymbol{x}'\right) = \boldsymbol{\phi}(\boldsymbol{x})^T \boldsymbol{\phi}(\boldsymbol{x}')$$

kernel function ***feature space mapping***

the simplest kernel is the **linear kernel** (identity mapping in feature space)

$$k\left(\boldsymbol{x}, \boldsymbol{x}'\right) = \boldsymbol{x}^T \boldsymbol{x}'$$

Dual representations

many ML models for regression and classification can be reformulated in a **dual representation** where the kernel functions arise naturally

consider a linear regression model with a regularised least-squares error function

$$E_{\mathrm{tr}}\left(\boldsymbol{\theta}\right) = \frac{1}{2}\sum_{n=1}^{N}\left(\boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{x_n}) - t_n\right)^2 + \frac{\lambda}{2}\boldsymbol{\theta}^T\boldsymbol{\theta}$$

regulator

output from training examples

the model parameters are determined **analytically** by requiring the vanishing of the gradient

$$\boldsymbol{\theta} = \sum_{n=1}^{N} a_n\boldsymbol{\phi}(\boldsymbol{x_n})\,, \quad a_n = -\frac{1}{\lambda}\left(\boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{x_n}) - t_n\right)$$

recall that inner products take place in feature space!

one can formulate a dual representation of the problem in terms of the parameter vector

$$\boldsymbol{a} = \left(a_1, a_2, \ldots, a_N\right)^T \qquad \boldsymbol{t} = \left(t_1, t_2, \ldots, t_N\right)^T$$

Dual representations

with some algebra one can show that the original figure of merit of the model

$$E_{\mathrm{tr}}\left(\boldsymbol{\theta}\right) = \frac{1}{2}\sum_{n=1}^{N}\left(\boldsymbol{\theta}^T\boldsymbol{\phi}(\boldsymbol{x}_n) - t_n\right)^2 + \frac{\lambda}{2}\boldsymbol{\theta}^T\boldsymbol{\theta}$$

admits a **dual representation** in terms of **kernel function**

$$E_{\mathrm{tr}}(\boldsymbol{a}) = \frac{1}{2}\boldsymbol{a}^T\boldsymbol{K}\boldsymbol{K}\boldsymbol{a} - \boldsymbol{a}^T\boldsymbol{K}\boldsymbol{t} + \frac{1}{2}\boldsymbol{t}^T\boldsymbol{t} + \frac{\lambda}{2}\boldsymbol{a}^T\boldsymbol{K}\boldsymbol{a}$$

where $\boldsymbol{K}$ is known as the Gram matrix, an N x N symmetric matrix with entries

$$K_{nm} = k\left(\boldsymbol{x}_n, \boldsymbol{x}_m\right) = \boldsymbol{\phi}(\boldsymbol{x}_n)^T\boldsymbol{\phi}(\boldsymbol{x}_m)$$

given by the kernel function evaluated over two of the **input training examples**

the crucial property of this dual representation is that now the predictions for new inputs will explicitly **depend on the training examples**

Dual representations

the crucial property of this dual representation is that now the predictions for new inputs will explicitly **depend on the training examples**

$$y(\boldsymbol{x}) = \boldsymbol{k}(\boldsymbol{x})^T \left(\boldsymbol{K} + \lambda \boldsymbol{I}_N\right)^{-1} \boldsymbol{t}$$

output of trained model

new input

$$\boldsymbol{k}(\boldsymbol{x}) = \left(k(\boldsymbol{x}_1, \boldsymbol{x}), k(\boldsymbol{x}_2, \boldsymbol{x}), \ldots\right)$$

in the dual formulation we invert a *N x N* matrix (data space) rather than a *M x M* one (feature space), so this does not appear to be advantageous …

the main benefit is being able to **work directly with kernel functions** and bypass a choice of feature map

this allows one to work in feature spaces of **very high (even infinite) dimensionality**

Choosing kernel functions

recall that the kernel function is constructed from a **feature space mapping**

$$k\left(\boldsymbol{x}_n, \boldsymbol{x}_m\right) = \boldsymbol{\phi}(\boldsymbol{x}_n)^T \boldsymbol{\phi}(\boldsymbol{x}_m) = \sum_{i=1}^{M} \phi_i(\boldsymbol{x}_n)\phi_i(\boldsymbol{x}_m)$$

one can also construct kernel functions directly, provided they can be expressed as above

e.g., for a 2D input space

$$k(\boldsymbol{x}, z) = \left(\boldsymbol{x}^T z\right)^2 = \left(x_1 z_1 + x_2 z_2\right)^2 = \boldsymbol{\phi}(\boldsymbol{x})^T \boldsymbol{\phi}(z)$$

$$\boldsymbol{\phi}(\boldsymbol{x}) = \left(x_1^2, \sqrt{2} x_1 x_2, x_2^2\right)$$

Note that feature space dimension (M=3) is bigger than input space dimensionality (d=2)!

there exist more general methods to construct acceptable kernels e.g. using **basis functions**

a popular choice is Gaussian kernel, built from an **infinite-dimensional feature map**

$$k(\boldsymbol{x}, z) = \exp\left(-||\boldsymbol{x} - z||^2 / 2\sigma^2\right)$$

Gaussian Processes

Gaussian processes

the previous discussion was limited to **non-probabilistic models.** We now extend the kernel methods to **probabilistic discrimination processes**

in Gaussian processes we bypass the need of a **parametric classifier/regressor** and work directly in the **space of functions for the (prior) probability distributions**

we can illustrate the philosophy of the method by revisiting the simple linear regression example

$$y(\boldsymbol{x}) = \boldsymbol{\theta}^T \boldsymbol{\phi}(\boldsymbol{x}) = \sum_{i=1}^{M} \theta_i \phi_i(\boldsymbol{x})$$

predictor ***model parameters*** ***feature space map***

assume that the model parameters have as **prior distribution** an isotropic Gaussian

$$p(\boldsymbol{\theta}) = \mathcal{N}\left(\boldsymbol{\theta} \,|\, \mathbf{0}, \alpha^{-1}\mathbf{1}\right)$$

for a given choice of the model parameters we will have a different functional form of the predictor

Gaussian processes