

1 Last Week Review

One popular interpretation of kernel methods is that the kernel approaches map a data point x into its feature space

$$x \rightarrow \Phi(x),$$

then subsequent operations (mostly linear) are carried out in the feature space. The dimension of the feature space is usually very large or even infinite. Luckily, the computational burden can be (partially) overcome by something dubbed as “kernel trick”: in stead of doing computations in the feature space directly, we only need to consider the Gram matrix (with dimension being the number of data points),

$$K_{ij} = \langle \Phi(x_i), \Phi(x_j) \rangle,$$

which carries all the information necessary for computation. Note that it is truth simply because we only consider the second order relationship (correlations) in the feature space (even the feature map itself is usually nonlinear). It applies to many famous analysis methods, such as linear regression, PCA, ICA, Fisher discriminant analysis (FDA), and partial least square (PLS).

Analogs can be made between ordinary linear regression and kernel methods. For example, consider a linear regression

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon.$$

If the result of the model fitting is not satisfactory, then usually we may add more covariates (such as $\beta_3 X_1 X_2$) into the regression equation. These new covariates are just features in the context of kernel methods. Questions are how many new features and which features we should put into the regression equation. Kernel methods adopt an extreme approach: a large (or maybe infinite) number of features are added. In regression scenario, the functional class is so rich that they can fit anything; hence, regularization is used to find a tractable answer. The regularization is also used in classification case such as SVM.

In practice, we may not even know what the feature map Φ is only if the inner produce $K(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle$ can be carried out efficiently. Mercer’s theorem guarantees the existence of such feature map Φ for functions K satisfying certain criteria. It is a big issue in practice that how a good kernel function can be adapted chosen from data.

2 Semi-Definite Programming

A general SDP problem is of the form:

$$\begin{aligned} \min \quad & c'x \\ \text{subject to} \quad & x_1 F_1 + \cdots + x_n F_n \succeq 0 \\ & Ax = b \end{aligned}$$

The above so-called linear matrix inequality (LMI) constraint is actually a non-linear constraint, and the linearity here only refers to the fact that the LHS matrix is element-wise linear.

Some of the advantages of SDP are

1. it is fast, important for kernel methods especially when the dimension of the Gram matrix is very high;
2. it is a convex problem, hence has no local minima/maxima.

3 Learning Kernel Matrix

Usually people determine a kernel function before applying any kernel learning procedure. There are many possible kernels functions out there. Which one is the best (in certain sense)? Or is it possible to adapt the kernel function to the data, i.e., let the data determine the kernel function.

One of the most important properties of the Gram matrix K is that it is nonnegative semi-definite, i.e., the kernel matrix lives in a semi-definite cone. Next, we will discuss one of the approaches in the literature about learning the kernel matrix for classification (SVM, to be specific) through SDP techniques.

Goal: to find the kernel matrices from the kernel matrix cone such that it satisfies certain optimal criteria.

Things we'd like to know:

1. How to translate the above optimization into a SDP problem?
2. How to avoid overfitting because the cone space is so large that all possible candidates are all in?

Regarding the overfitting problem, in the paper, the search is restrained to a sub-space of linear combinations of some pre-selected kernel functions. **Question: can we use penalization terms to regularize this optimization problem instead of limiting the search space?** Sure, their method is another way of doing regularization.

3.1 Alignment

The alignment between two kernel matrix is defined as the correlation between two matrices (treat them as vectors instead):

$$A(K_1, K_2) = \frac{\text{cov}(K_1, K_2)}{\|K_1\|_2 \|K_2\|_2},$$

which serves as a dissimilarity measure between kernel matrices. **Question: is there any other ways to define the similarity measure?**

Given sample data, *target kernel matrix* serves as the “sample kernel matrix”

$$K_s = yy'.$$

Note that it is not a matrix induced by any kernel function, but from the truth directly. **Question: can the target matrix be fairly generalized to regression cases?**

The goal here is to

$$\begin{array}{ll} \min & A(K, K_s) \\ \text{subject to} & K \succeq 0 \end{array}$$

This optimization problem can be rewritten as an SDP problem.

3.2 Margin

For linear separable cases, the SVM can be written in the following form:

$$\begin{aligned} \min \quad & \frac{1}{2} \langle w, w \rangle \\ \text{subject to} \quad & y_i (\langle w, \Phi(x_i) \rangle + b) \geq 1, i = 1, \dots, n \end{aligned}$$

with the hard margin to be $\gamma = 1/\|w^*\|_2$ at the optimal point.

Similarly, we can translate the above problem into a semi-definite programming problem.

3.3 Transductivity

Transductivity: we have access to both labelled and unlabeled data, and hope that the kernel learned from the training data will also yield a good embedding for test data. It can be viewed as a problem with missing labels in my understanding. Question: is there any benefit to add unlabeled data in?

In their learning algorithm, the constraint on the kernel matrix search space is given by

$$K = \sum_{i=1}^m \mu_i K_i,$$

where K_i 's are induced by some pre-selected kernel functions. Note that they didn't put the positive constraints on the coefficients on μ_i . (**reasonable?**) That's why they need to explicitly specify

$$\sum_i \mu_i K_i \succeq 0,$$

where K_i 's are for both labelled/unlabeled data.

The error bounds for transduction mainly involves Rademacher complexity calculation and concentration inequality. We may touch some details later if we decide to.