Master M2 MSIAM and MOSIG

COSTA MAIA Isabella

Kernel Methods for Machine Learning

Data Challenge

17th February, 2021

This project had the aim of predicting whether a DNA sequence region is binding site to a specific transcription factor. Despite of a quick trial in support vector machines, the kernel ridge regression was the most explored model and the final code version chosen.

Ridge Regression

This model was implemented in several steps, the first of them being a simple ridge regression without a kernelization and using the numeric data. This part was based mainly in the functions ridgeEstimators(), which provided the regression coefficients for a given set of pairs of training samples and a given λ and predictRidge() that returned the predictions for a given feature matrix and a given vector of regression coefficients. This first version provided a score of 0.6366 in the public score using $\lambda = 0.001$.

Besides that, the kernelized ridge regression was implemented and three different kernels were tested. The estimators are provided by kernelRidgeEstimators() function and are calculated from the gram matrix computed at training time, its correspondent training labels and chosen λ value.

To compute the gram-matrix in train time, or the [K] matrix in test time, two different functions were implemented, both using multiprocessing to parallelize the computation within the rows of one of the input matrices, since computational resources were a major issue to overcome in order to provide predictions in a reasonable time. The function that provided the gram matrix at test time is an equivalent version of the other that computes the [K] matrix for test time, but that takes advantage of the symmetry to do only half of the computation.

Four different kernel functions were implemented. The first one, the Gaussian kernel, was used mostly as a test to verify the kernel version of the regression. Still using the numeric data tables, it provided worse accuracy in the public score than the non kernelized version using the same λ value and with $\sigma = 0.37$.

The second kernel implemented was the spectrum kernel that allowed to finally use the raw data. It was made by using a single loop to run through both input sequences and adding each one of the k-mers into the keys of a dictionary whose values were the counting of the presence of these k-mers in each sequence. The correspondent score was then the sum of product of the number of k-mers in each sequence. Using this simple yet time costing kernel provided a considerably better score than the previous tested versions, achieving an accuracy of 0.67733 in the private score when considering a k-mer of legth=8 .

The third kernel implemented was the mismatch kernel. It was actually a personalised version of it since it is based on a score created to penalise the kernel similarity with a higher number of mismatches. The function is as the following:

$$(\frac{counts}{k})^2 \exp^{(\frac{counts}{k})^2 - 1}$$

where *counts* is the number of common monomers placed at the same position within a k-mer and k is the length of the k-mer. However, as this kernel relies on a comparison not only between k-mers but also on their sub sequences, it is an extremely costly function. When computing the ridge regression with this kernel, even more than 30h running locally were not enough to provide predictions.

The final fourth implemented kernel and used as final submission is in fact a twisted version of the spectrum kernel, based simply in a linear combination of this function when considering different lengths of k-mers. Moreover, since it was added the information that the datasets came from different distributions, in this approach different λ values for each distribution and different choices of k-mer lengths in each one of the linear combinations were chosen. The values of λ and the lengths used in each combination were found assessing the accuracy in a split of each one of the train sets using the functions split() and evaluate().

One thing that is worth mentioning is that, unfortunately, in spite of writing functions for that, I didn't applied a re-scaling in the labels from the domain {0,1} to {-1,1}, what could have caused an important loss of accuracy when using the kernel ridge regression models implemented since the rounding between 0 and 1 can provoke bigger mistakes.

The final scores are the following:

- **Public Score:** 0.72000
- **Public rank:** 4th position
- **Private Score:** 0.69666
- **Private rank:** 11th position

The code can be downloaded from the link:

`https://gitlab.ensimag.fr/costamai/kernel-methods-data-challenge`