Kernel Methods

Homework 2

Luis Antonio Ortega Andrés
Antonio Coín Castro

March 4, 2021

1 TODO

Contestar preguntas en notebook (y eliminar este pdf).

Pasar .pys a pdf.

Añadir referencias (documentación sklearn, paper PCA?) en formato Chicago.

Aclarar que eliminamos las componentes no nulas en KPCA.

Añadir gráficas sobre cómo es KPCA según gamma->0 o gamma->inf (están en la carpeta /img) (?)

En el notebook de CV:

- Comentar gráfica.

- Comentar mejores parámetros y resultado en test.

- Añadir gamma y nu,scale en Nyström en cv.

DUDA: En Random Features, n_components tiene que ser par (?). Cómo hacemos para pintar bien la gráfica del error? (ahora mismo np.repeat)

Kernel PCA

Why do the projections onto the first two KPCA principal components look different for the Sklearn and our implementation? Is any of the two incorrect?

Both representations are correct and valid, what makes them different is the decisions made at normalizing the eigenvectors.

These vectors are returned with $\|\cdot\|_{\mathcal{L}_2} = 1$ in $\mathbb{R}^N$ by Numpy and Scipy via specialized functions such as `eig` and `eigh` (version for symmetric cases). However, the kernel method needs these vectors to be unitary in the subjacent Hilbert space, not $\mathbb{R}^N$. This can be achieved by ensuring that their $\mathcal{L}_2$-norm in $\mathbb{R}^N$ equals the inverse of the square root of their corresponding eigenvalues $1/\sqrt{\lambda}$. The procedure is then to divide each eigenvector by this quantity, and it is in this step that generates discrepancy, as one may divide by $-1/\sqrt{\lambda}$ getting the same $\mathcal{L}_2$-norm.

Sklearn's implementation uses an auxiliary function `svd_flip` that deterministically assigns a sign for each normalized eigenvector (reference).

We have implemented this option as a parameter for our function in order to reproduce Sklearn's output perfectly. In our case, not using this function leads to a reflection of the second principal component in the Kernel PCA function. Linear PCA does not show this behavior by pure coincidence.

Vary the parameters of the kernel and comment on the behavior of the projections onto the first two KPCA components for the different values considered (e.g. $\gamma \in \{0.02, 0.2, 2.0, 20.0, 200.0, 2000.0\}$. In particular,

1. *What is the behavior in the limit in which the width of the kernel approaches ∞. Explain why one should expect such behavior.*
2. *What is the behavior in the limit in which the width of the kernel approaches 0. Explain why one should expect such behavior.*

First of all, to study the limit behavior of the projections we are using the explicit kernel formula:

$$\mathcal{K}(x,y) = Ae^{-\frac{\|x-y\|^2}{2\omega^2}},$$

where A is the output variance and ω is the kernel's width and γ is inversely proportional to ω . Using this formula, and given two fixed input values $x \neq y$,

$$\lim_{\gamma \to 0^+} = \lim_{\omega \to \infty} \mathcal{K}(x,y) = A.$$

$$\lim_{\gamma \to \infty} = \lim_{\omega \to 0^+} \mathcal{K}(x,y) = 0.$$

Let us begin with the upper limit, in this case, the kernel tends to 0 on any pair of non equal points. This does not apply when computing $\lim_{\gamma \to \infty} \mathcal{K}(x,x)$ as $\mathcal{K}(x,x) = 1$ is constant given any value of γ. Given this situation and considering a set of points $\boldsymbol{X}$, their kernel matrix converges to

$$\lim_{\gamma \to \infty} \mathcal{K}(\boldsymbol{X}, \boldsymbol{X}) = \boldsymbol{I}.$$

This leads to the following situation: the application of the kernel to any set of points results in an identity matrix, which has all of its eigenvalues equal to 1 and its eigenvector to settle a base in $\mathbb{R}^N$. There is no need for such base to be the usual base of $\mathbb{R}^N$ but it does when using Numpy's `eigh` function.

As a result, the kernel method performs the following matrix operation:

$$\mathcal{K}(\boldsymbol{X}_{test}, \boldsymbol{X}) \begin{pmatrix} e_1 & \dots & e_N \end{pmatrix}$$

Which would lead to a 0 matrix if $\boldsymbol{X}_{test}$ and $\boldsymbol{X}$ where disjoint. However, in our case, these matrix are equal, and their kernel a identity matrix:

$$\mathcal{K}(\boldsymbol{X}, \boldsymbol{X}) \begin{pmatrix} e_1 & \dots & e_N \end{pmatrix} = \boldsymbol{I}\boldsymbol{I} = \boldsymbol{I}.$$

Given this, plotting a projection over the first two components leads to plotting a point in $(1,0)$, a points in $(0,1)$ and the rest in the origin. Which is what can be seen at the end of the animation.

However, there is an interesting behaviour of the `eigh` function when its argument is near to an identity matrix but it is not. The returned eigenvalues are roughly equal to 1 and the eigenvector are nearly a base of $\mathbb{R}^N$, but they are not the usual base (neither near). As a result, the plot leads to the projection over $\mathbb{R}^2$ of a base in $\mathbb{R}^N$, which corresponds to the "spiked" appearance of the projection.

Using this theoretical results, increasing the value of gamma result in projecting all the datapoints into 0, and decreasing it

Nyström approximation

Comment on the values of the error for the different approximations, and their dependence with the number of sampled features.

(Extra point) Determine de dependence of the mean error with the number of features for the different random feature models. Provide an explanation of this behavior.