Optimizing Kernel Density Estimator in Python

Reynaldo Gil

July 25, 2019

Abstract

The work show the results of creating and optimizing python code for computing a Kernel Density Estimator over object of very high dimensionality.

1 Introduction

Kernel Density Estimation (KDE) is a non-parametric way to estimate the probability density function of a random variable ([1]).

2 Mathematical model

Given a collection of k points $z_i \in R^d$ the log-likelihood of a point $x \in R^d$ can be expressed :

$$\log p(x) = \log \sum_{i=1}^{k} p(z_i)p(x|z_i)$$

Assuming that $p(z_i) = \frac{1}{k}$ and $p(x|z_i)$ is described by the product of Gaussian components for each dimension:

$$p(x_j|z_i) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{(x_j - z_{ij})^2}{2\sigma^2}}$$

Then:

$$\log p(x) = \log \sum_{i=1}^{k} \frac{1}{k(\sigma\sqrt{2\pi})^d} \prod_{j=1}^{d} e^{-\frac{(x_j - z_{ij})^2}{2\sigma^2}}$$

Applying the exponential and logarithms rules this can be expressed as:

$$\log p(x) = -\log k - d\log\sigma - \frac{d}{2}\log 2\pi + \log\sum_{i=1}^{k} e^{-\sum_{j=1}^{d}\frac{(x-z_{ij})^2}{2\sigma^2}}$$

$$\log p(x) = -\log k - d\log\sigma - \frac{d}{2}\log 2\pi + \log\sum_{i=1}^{k} e^{-\frac{|x-z_i|^2}{2\sigma^2}}$$

This is equivalent to:

$$\log p(x) = \log\sum_{i=1}^{k} e^{\log\frac{1}{k} - \sum_{j=1}^{d}\left\{\frac{(x_j - z_{ij})^2}{2\sigma^2} + \frac{1}{2}\log 2\pi\sigma^2\right\}}$$

Unless both formulas are equivalent they have implication for computing due to the limit resolution and approximate computation in the computer. Eq 2 can have problems to compute when σ is small because the exponential can became 0. Eq. 2 can manage better this cases because when σ is small $\log 2\pi\sigma^2$ is negative and decrease the sum. However, when σ is big $d\log 2\pi\sigma^2$ can be so big again and the exponential become 0. So, depending of the σ is better use one or another.

3 Implementation

The interpreted and dynamic computation of python allow an easy programming but the execution is very inefficient because every object use a lot of memory and need to check its type in execution time. This problem increase with the size of the data. It is needed to avoid computation in Python to achieve a better performance. Due to this problem a lot of libraries implemented in C or other languages have been created. This libraries do the hard work and python works as the glue between then. In particular numpy is the basic

References

[1] Wikipedia. Kernel density estimation.