Xi'an Jiaotong-Liverpool University

Pattern Recognition

Lecture 11. non-Parametric methods: Kernel Density Estimation & KNN

Dr. Shanshan ZHAO & Dr. Yuxuan ZHAO

School of AI and Advanced Computing
Xi'an Jiaotong-Liverpool University

Academic Year 2023-2024

Table of Contents

Agenda

- Parametric Modeling
- Non-Parametric Modeling

Parametric Modeling

- Data availability in a Bayesian framework
  - We could design an optimal classifier if we knew $P(\omega_i)$ and $P(x|\omega_i)$
  - Unfortunately, we rarely have that much information available.
- Assumption
  - A prior information about the problem
  - The form of underlying density
    - Example: Normal density $P(x|\omega_i)$: decided by 2 parameters.
- Estimation techniques
  - Maximum-Likelihood(ML) and Maximum A Posteriori (MAP)
- Other techniques
  - Gaussian Mixture Model (GMM) and Hidden Markov Model (HMM)

Non-Parametric Modeling

- Non-Parametric Modeling tries to model arbitrary distributions without assuming a certain parametric form, i.e., without assuming forms of underlying densities.
- Two types:
  - Parzen window/kernel density
  - K-Nearest Neighbor

Probability Density Function(pdf)

$$P(a < x < b) = \int_a^b p(x)dx$$

$$\int_{-\infty}^{\infty} p(x)dx = 1$$

Density Estimation

The most fundamental techniques rely on the fact that the probability P that a vector x will fall in a region $\mathscr{R}$, $p(x)$ is given by

$$P = \int_{\mathscr{R}} p(x)dx$$

If we now assume that $p(x)$ is continuous and that the region $\mathscr{R}$ is so small that p does not vary appreciably within it, we can write

$$P = p(x) \int_{\mathscr{R}} dx = p(x)V$$

Given data $x_1, x_2, ..., x_n$, we can calculate the number of data points falling in the region $\mathscr{R}$ is k, and

$$p(x) = \frac{k}{nV}$$

Parzen windows

- The Parzen-window approach to estimating densities can be introduced by temporarily assuming that the region $\mathscr{R}$ is a d-dimensional hypercube.
- If h is the length of an edge of that hypercube, then its volume is given by

$$V = h^d$$

- We can obtain an analytic expression for k, the number of samples falling in the hypercube, by defining the following window function:

$$\phi(\frac{x - x_i}{h}) = \begin{cases} 1, \frac{|x_{[j]} - x_{i[j]}|}{h} \leq \frac{1}{2}, j = 1, 2, ..., d \\ 0, otherwise \end{cases}$$

[1]

[1] x is a data point with d dimentions, here we represent its jth dimension's coordiante with $x_{[j]}$.

Parzen windows

- The number of samples in this hypercube is therefore given by

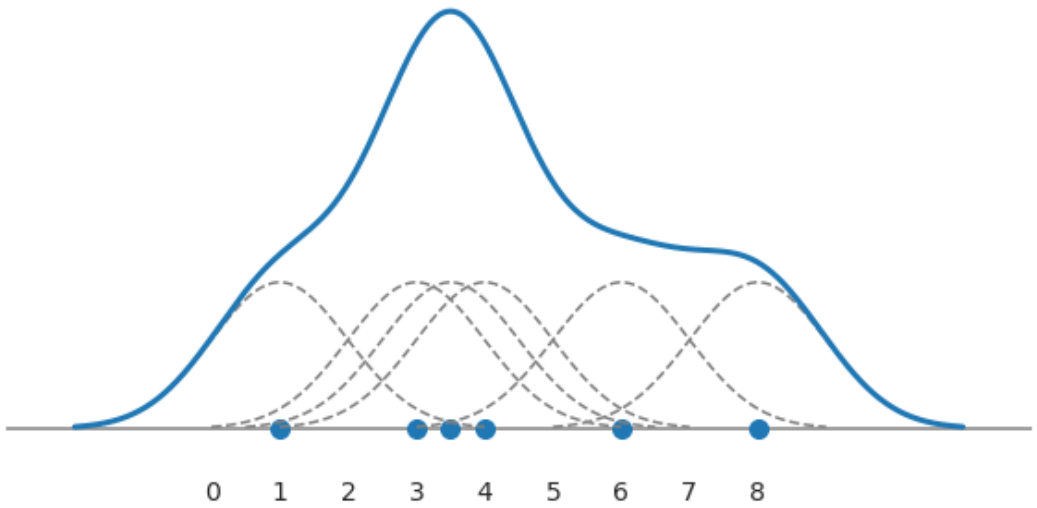
$$k = \sum_{i=1}^{n} \phi(\frac{x - x_i}{h})$$

- we obtain the estimate

$$p(x) = \frac{k}{nV} = \frac{1}{nV} \sum_{i=1}^{n} \phi(\frac{x - x_i}{h})$$

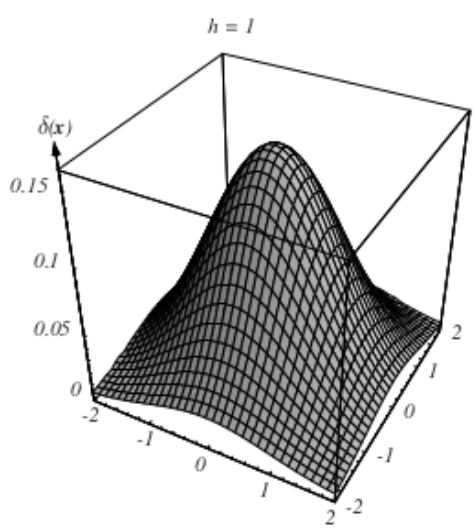
Play around

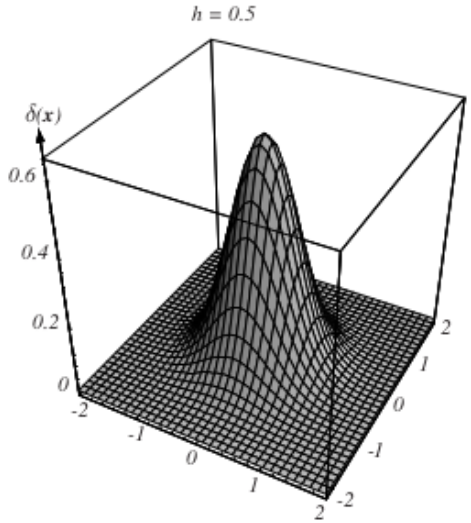
Lecture 11 KDEplot.ipynb

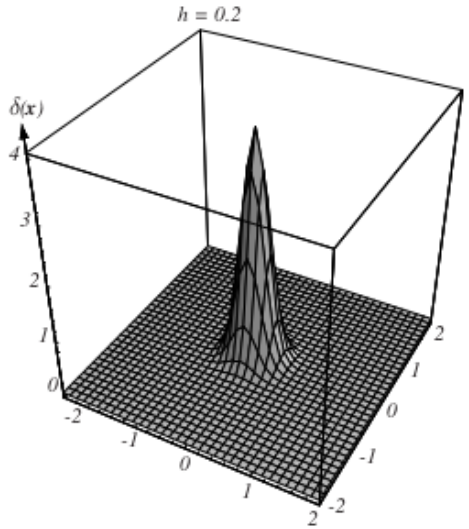


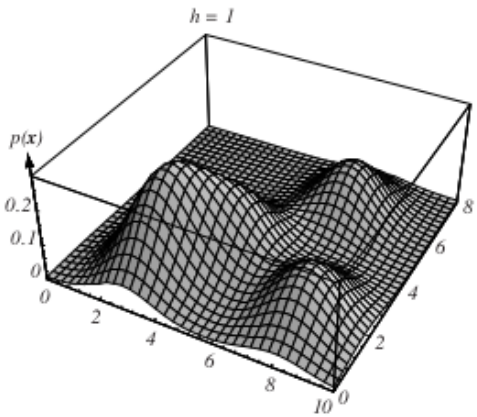
TASK

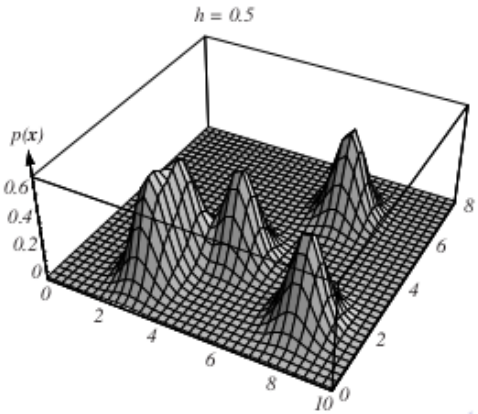
- Show that whether the choice of kernel matters much.
- Show that whether the bandwidth of kernel matters much.

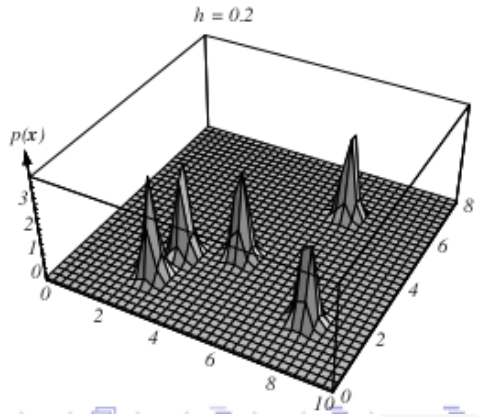

h = 1
δ(x)
0.15
0.1
0.05
0
-2
-1
0
1
2
-2
-1
0
1
2


h = 0.5
δ(x)
0.6
0.4
0.2
0
-2
-1
0
1
2
-2
-1
0
1
2


h = 0.2
δ(x)
4
3
2
1
0
-2
-1
0
1
2
-2
-1
0
1
2


h = 1
p(x)
0.2
0.1
0
0
2
4
6
8
10
0
2
4
6
8


h = 0.5
p(x)
0.6
0.4
0.2
0
0
2
4
6
8
10
0
2
4
6
8


h = 0.2
p(x)
3
2
1
0
0
2
4
6
8
10
0
2
4
6
8

TASK 2

Try the code with other dataset, e.g., iris

`https://archive.ics.uci.edu/ml/datasets/iris`

K-Nearest-Neighbor Estimation

- $$p_n(x) = \frac{k_n/n}{V_n}$$
- As we have seen, one of the problems encountered in the Parzen-window approach concerns the choice of the sequence of cell volumes sizes V_1 , V_2 , ... or overall window size (or indeed other window parameters, such as shape or orientation).
- For example, if we take $V_n = V_1/\sqrt{n}$, the results for any finite n will be very sensitive to the choice for the initial volume V_1 .
  - If V_1 is too small, most of the volumes will be empty, and the estimate $p(x)$ will be very erratic.
  - On the other hand, if V_1 is too large, important spatial variations in $p(x)$ may be lost due to averaging over the cell volume.

K-Nearest-Neighbor Estimation

- A potential remedy for the problem of the unknown "best" window function is to let the cell volume be a function of the training data, rather than some arbitrary function of the overall number of samples.
- For example, to estimate $p(x)$ from n training samples or prototypes we can center a cell about x and let it grow until it captures k_n samples, where k_n is some specified function of n.
- These samples are the k_n nearest-neighbors of x. It the density is high near x, the cell will be relatively small, which leads to good resolution. If the density is low, it is true that the cell will grow large, but it will stop soon after it enters regions of higher density.
- In either case, if we take

$$p_n(x) = \frac{k_n/n}{V_n} \tag{1}$$

K-Nearest Neighbor Density Estimation

$$p_n(x) = \frac{k_n/n}{V_n} \tag{2}$$

where p_n is the approximation to $p(x)$, which is the density function we want to estimate. n is the number of samples.

How to interpret it?

K-Nearest Neighbor Density Estimation

Let $x_1, x_2, ..., x_n$ be our random samples. Assume each observation has d different variables; namely, $x_i \in \mathscr{R}^d$. For each given point x, we first rank every observation based on its distance to x. Let $R_k(x)$ denotes the distance from x to its k-th nearest neighbor point.

For a given point x, the KNN density estimator estimates the density by

$$p_n(x) = \frac{k}{n} \frac{1}{V_d \cdot R_k^d(x)} \tag{3}$$

$$= \frac{k}{n} \cdot \frac{1}{\text{Volume of a } d\text{-dimensional ball with radius being } R_k(x)} \tag{4}$$

Where $V_d = \frac{\pi^d 2}{\Gamma(\frac{d}{2}+1)}$ is the volume of a unit d-dimensional ball and $\Gamma(x)$ is the Gamma function.

K-Nearest Neighbor Density Estimation

Here are the results when $d = 1, 2$, and 3

- $d = 1, V_1 = 2 : p_n(x) = \frac{k}{n} \frac{1}{2R_k(x)}$.
- $d = 2, V_2 = \pi : p_n(x) = \frac{k}{n} \frac{1}{\pi R_k^2(x)}$.
- $d = 3, V_3 = \frac{4}{3}\pi : p_n(x) = \frac{k}{n} \frac{3}{4\pi R_k^3(x)}$.

K-Nearest Neighbor Density Estimation

Example

We consider a simple example in $d = 1$. Assume our data is $X = \{1, 2, 6, 11, 13, 14, 20, 33\}$.

(i) What is the KNN density estimator at $x = 5$ with $k = 2$?

Solution: First, we calculate $R_2(5)$. The distance from $x = 5$ to each data point in X is $\{4, 3, 1, 6, 8, 9, 15, 28\}$. Thus, $R_2(5) = 3$ and

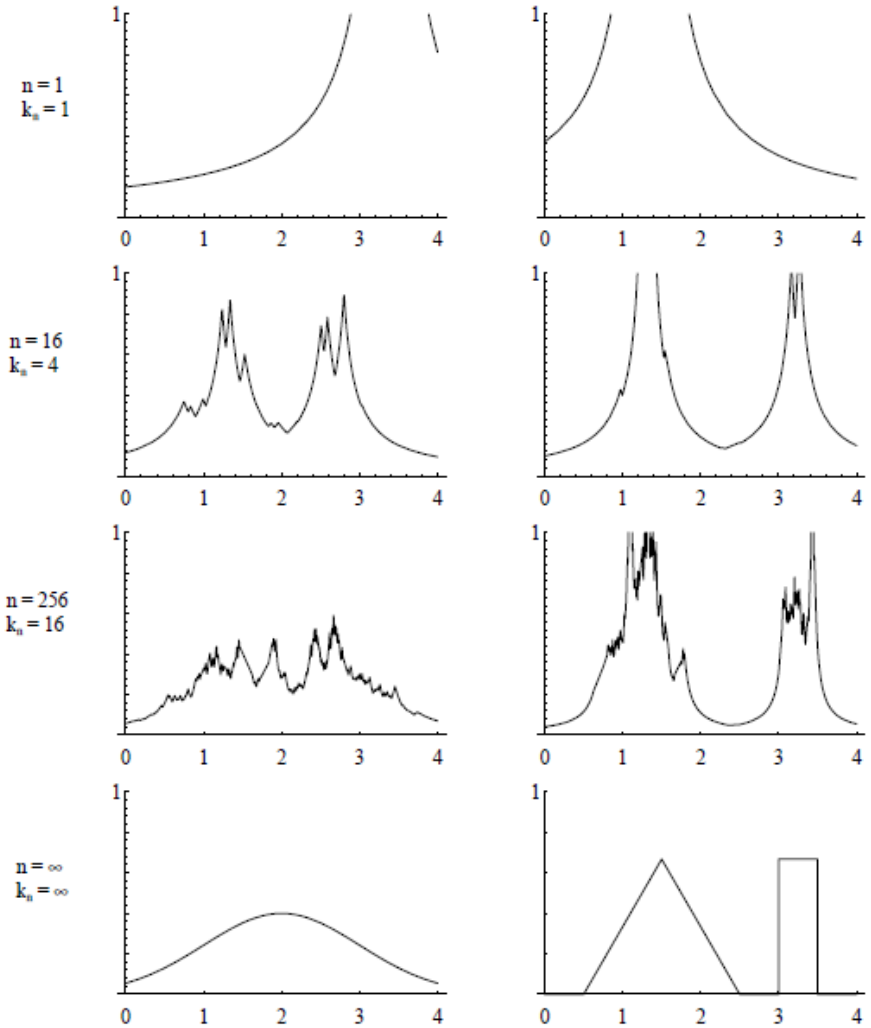
$$p(5) = \frac{2}{8} \frac{1}{2 \cdot R_2(5)} = \frac{1}{24} \tag{5}$$

(ii) What will the density estimator be when we choose $k = 5$?

In this case, $R_5(5) = 8$, so

$$p(5) = \frac{5}{8} \frac{1}{2 \cdot R_5(5)} = \frac{5}{128} \tag{6}$$

K-Nearest Neighbor Density Estimation



Several *k*-nearest-neighbor estimates of two unidimensional densities: a Gaussian and a bimodal distribution. Notice how the finite *n* estimates can be quite "spiky."[**duda1973pattern**]

k-NN Posterior Estimation for Classification

- We can directly apply the k-NN methods to estimate the posterior probabilities $P(\omega_i|x)$ from a set of n labeled samples.
- Place a window of volume V around x and capture k samples, with k_i turning out to be of label ω_i.
- The estimate for the joint probability is thus

$$p_n(x, \omega_i) = \frac{k_i}{nV} \tag{7}$$

- A reasonable estimate for the posterior is thus

$$P_n(\omega_i|x) = \frac{p_n(x, \omega_i)}{\sum_i^N p_n(x, \omega_i)} = \frac{k_i}{k} \tag{8}$$

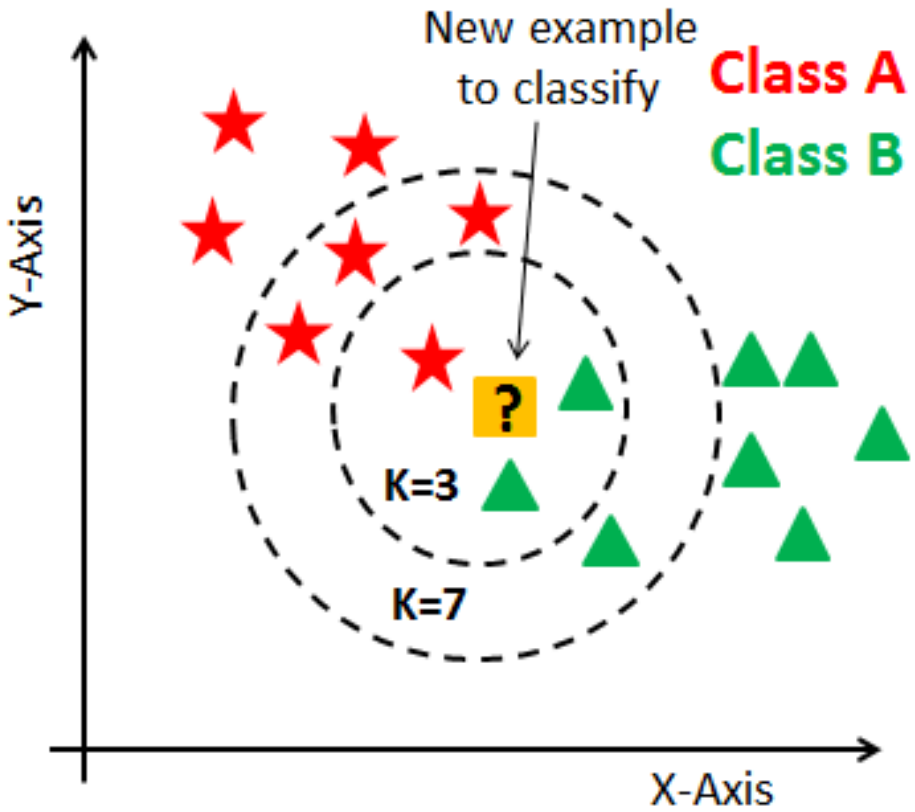
- The Posterior probability for ω_i is simply the fraction of samples within the window that are labeled ω_i. This is a simple as well as an intuitive result.

Pros and Cons

Pros	Cons
• Extremely easy to implement • It is a lazy algorithm and therefore requires no training prior to making predictions. This makes KNN much faster • There are only two parameters required to implement KNN, i.e., the value of k and the distance function (e.g., Euclidean or Manhattan etc.)	• KNN doesn't work well with high dimensional data because it becomes difficult to calculate the distance. • KNN has high prediction cost for large datasets. • KNN doesn't work well with categorical features since it is difficult to find the distance.

K-Nearest Neighbor Classification problem

Basic idea



Codes

Learn how to do classification with KNN.

Lecture 11 KNN.ipynb

Think: How to choose K?

Just as the bandwidth in the KDE, it is a very difficult problem in practice. However, according to theoretical analysis, a rough idea about how k should be changing with respect to the sample size n. (check the material provided on LMO if you are interested)

Thank You !

$\mathcal{Q}$ & $\mathcal{A}$