CS559 Machine Learning
Nonparametric Methods

Tian Han

Department of Computer Science
Stevens Institute of Technology

Week 7

Outline

- Nonparametric Methods
- Kernel Density Estimator (Parzen window)
- K Nearest-neighbour

Nonparametric Methods

Nonparametric Approach

- Parametric approaches: use probability distribution having specific functional forms governed by a small number of parameters (e.g. $\mathbf{w}$).
  - $\mathbf{w}$ learned from data.
  - Limitation: chosen density might be poor model of the true distribution that generates the data.

Nonparametric Approach

- Parametric approaches: use probability distribution having specific functional forms governed by a small number of parameters (e.g. $\mathbf{w}$).
  - $\mathbf{w}$ learned from data.
  - Limitation: chosen density might be poor model of the true distribution that generates the data.
- Nonparametric approaches: make few assumptions about the form of the distribution.

Density Estimation

In practice, the underlying models can be so complicated, extremely **hard** to develop specific parametric functional form.

Density Estimation

In practice, the underlying models can be so complicated, extremely **hard** to develop specific parametric functional form.

- The model can be *multi-modality*.

Density Estimation

In practice, the underlying models can be so complicated, extremely **hard** to develop specific parametric functional form.

- The model can be *multi-modality*.
- Difficult to collect sufficient amount of data at each point.

Density Estimation

In practice, the underlying models can be so complicated, extremely **hard** to develop specific parametric functional form.

- The model can be *multi-modality*.
- Difficult to collect sufficient amount of data at each point.

Figure: $p(x, y)$ is the probability that a raindrop hits a position (x,y). [Stat 231, S.C Zhu]

Density Estimation—Frequentist method

Given: N observations $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$

Density Estimation—Frequentist method

Given: N observations $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$

Goal: estimate the density model $p(x)$.

Density Estimation—Frequentist method

Given: N observations $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$

Goal: estimate the density model $p(x)$.

Use **histogram**:

- n_i: the number of observations of x falling in bin i
- N: total number of observations
- Δ_i: the width of the bins to obtain probability values

Density Estimation—Frequentist method

Given: N observations $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$

Goal: estimate the density model $p(x)$.

Use **histogram**:

- n_i: the number of observations of x falling in bin i
- N: total number of observations
- Δ_i: the width of the bins to obtain probability values
- $\mathbf{p_i = \frac{n_i}{N\Delta_i}}$
- Easy to show: $\int p(x)dx = 1$

Different bin widths for histogram

Too small Δ: spiky. Too large Δ: too smooth

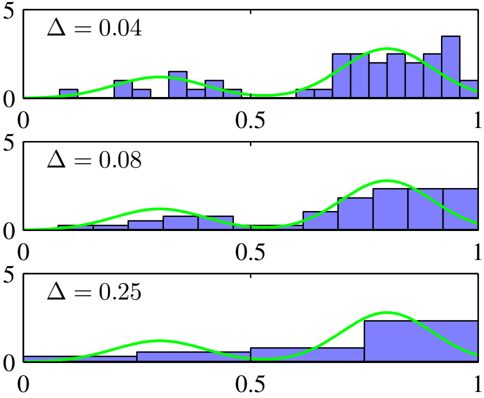
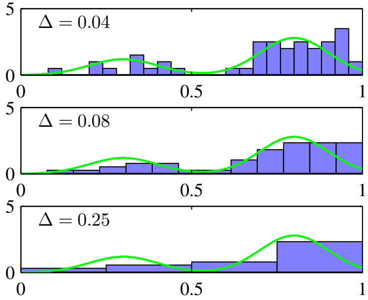


Figure: [C. Bishop, PRML]

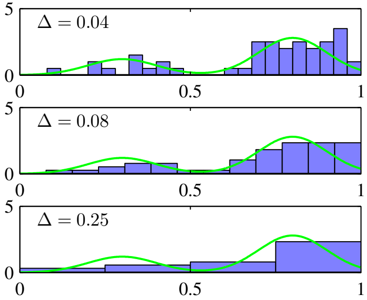
Property of histogram method

- Once the histogram computed, data can be discarded
- Easily applied if data arriving sequentially
- Bin edges introduce the discontinuities of estimated density
- Limitation: scaling with dimensionality, M^d bins for d-dimensional data.



Property of histogram method

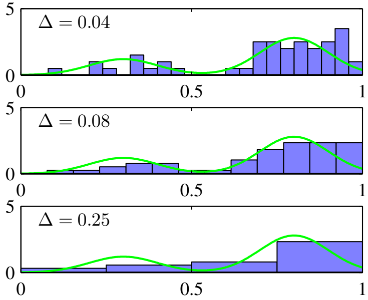
- Once the histogram computed, data can be discarded
- Easily applied if data arriving sequentially
- Bin edges introduce the discontinuities of estimated density
- Limitation: scaling with dimensionality, M^d bins for d-dimensional data.



1. Consider the data points that lie within some local neighborhood of that point

Property of histogram method

- Once the histogram computed, data can be discarded
- Easily applied if data arriving sequentially
- Bin edges introduce the discontinuities of estimated density
- Limitation: scaling with dimensionality, M^d bins for d-dimensional data.



1. Consider the data points that lie within some local neighborhood of that point
2. The value of the smoothing parameters (e.g., bin width) should be neither too large nor too small

Nonparametric method for density estimation

- Suppose we have collected a data set comprising N observations drawn from $p(\mathbf{x})$
- Consider small region $\mathcal{R}$ containing $\mathbf{x}$. The probability mass associated with this region is:

$$P = \int_{\mathcal{R}} p(\mathbf{x}) d\mathbf{x}$$

- The total number K of points lie inside region $\mathcal{R}$ follows binomial distribution:

$$Bin(K|N, P) = \frac{N!}{K!(N-K)!} P^K (1-P)^{N-K}$$

Nonparametric method for density estimation

- For large N, the distribution will be sharply peaked around the mean:

$$K \approx NP$$

- If the region $\mathcal{R}$ is small enough, $p(\mathbf{x})$ is roughly constant over the region: (V is the volume of the region $\mathcal{R}$)

$$P \approx p(\mathbf{x})V$$

- Our density estimate:

$$p(\mathbf{x}) = \frac{K}{NV}$$

Several problems

- If fix volume V, and take more and more samples N, can only get space-averaged value of $p(\mathbf{x})$:

$$p(\mathbf{x}) \approx \frac{P}{V} = \frac{\int_{\mathcal{R}} p(\mathbf{x}) d\mathbf{x}}{\int_{\mathcal{R}} d\mathbf{x}}$$

Need to let V approach 0.

Several problems

- If fix volume V, and take more and more samples N, can only get space-averaged value of $p(\mathbf{x})$:

$$p(\mathbf{x}) \approx \frac{P}{V} = \frac{\int_{\mathcal{R}} p(\mathbf{x}) d\mathbf{x}}{\int_{\mathcal{R}} d\mathbf{x}}$$

  Need to let V approach 0.

- If fix the number of samples N, and let V approach 0. The region will eventually become too small to contain *no samples*. $p(\mathbf{x}) \approx 0$ meaningless estimations.

Limiting behavior

Consider a sequence of regions $\mathcal{R}_1, \mathcal{R}_2, \ldots$ containing $\mathbf{x}$, $p_n(\mathbf{x})$ be the n-th estimate for $p(\mathbf{x})$:

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

- $\mathcal{R}_n$: the region used when we have n samples.
- V_n: volume of $\mathcal{R}_n$.
- k_n: number of points falling in $\mathcal{R}_n$.

Limiting behavior

Consider a sequence of regions $\mathcal{R}_1, \mathcal{R}_2, \ldots$ containing $\mathbf{x}$, $p_n(\mathbf{x})$ be the n-th estimate for $p(\mathbf{x})$:

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

- $\mathcal{R}_n$: the region used when we have n samples.
- V_n: volume of $\mathcal{R}_n$.
- k_n: number of points falling in $\mathcal{R}_n$.

Goal: design the sequence of regions $\mathcal{R}_1, \mathcal{R}_2, \ldots$ containing $\mathbf{x}$, such that:

$$n \to \infty,\ p_n(\mathbf{x}) \to p(\mathbf{x})$$

Three conditions

Three conditions for region design to ensure $p_n(\mathbf{x}) \to p(\mathbf{x})$:

Three conditions

Three conditions for region design to ensure $p_n(\mathbf{x}) \to p(\mathbf{x})$:

- $\lim_{n\to\infty} V_n = 0$: ensure space-averaged $\frac{P}{V}$ will converge to $p(\mathbf{x})$.

Three conditions

Three conditions for region design to ensure $p_n(\mathbf{x}) \to p(\mathbf{x})$:

- $\lim_{n\to\infty} V_n = 0$: ensure space-averaged $\frac{P}{V}$ will converge to $p(\mathbf{x})$.
- $\lim_{n\to\infty} k_n = \infty$: assume $p(\mathbf{x}) \neq 0$ and sufficient samples for each $\mathbf{x}$ to ensure the frequency ratio to converge to P.

Three conditions

Three conditions for region design to ensure $p_n(\mathbf{x}) \to p(\mathbf{x})$:

- $\lim_{n\to\infty} V_n = 0$: ensure space-averaged $\frac{P}{V}$ will converge to $p(\mathbf{x})$.
- $\lim_{n\to\infty} k_n = \infty$: assume $p(\mathbf{x}) \neq 0$ and sufficient samples for each $\mathbf{x}$ to ensure the frequency ratio to converge to P.
- $\lim_{n\to\infty} k_n/n = 0$: k_n grows slower than n.

Two common ways to design the regions sequences

The density estimate for n samples:

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

Two common ways to design the regions sequences

The density estimate for n samples:

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

- Kernel density (Parzen window): fixing region size for all $\mathbf{x}$, e.g., $V_n = 1/\sqrt{n}$, and determine k_n from the data.

Two common ways to design the regions sequences

The density estimate for n samples:

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

- Kernel density (Parzen window): fixing region size for all $\mathbf{x}$, e.g., $V_n = 1/\sqrt{n}$, and determine k_n from the data.
- K-nearest neighbours: fixing k_n, e.g., $k_n = \sqrt{n}$, let V_n grows until it encloses k_n neighbours of $\mathbf{x}$.

Two common ways to design the regions sequences

The density estimate for n samples:

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

- Kernel density (Parzen window): fixing region size for all $\mathbf{x}$, e.g., $V_n = 1/\sqrt{n}$, and determine k_n from the data.
- K-nearest neighbours: fixing k_n, e.g., $k_n = \sqrt{n}$, let V_n grows until it encloses k_n neighbours of $\mathbf{x}$.

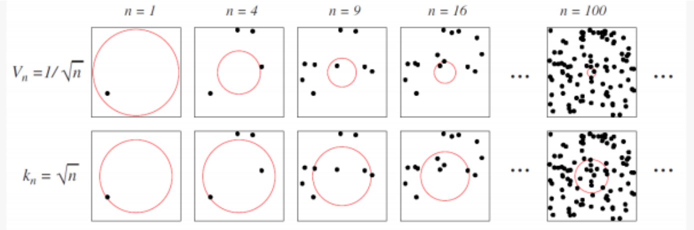


Figure: [R. Duda, Pattern Classification]

Parzen Window

Parzen window

Assume region $\mathcal{R}_n$ is d-dimensional hypercube centred around point $\mathbf{x}$. The effective volume of hypercube would be $V_n = h_n^d$.

Define window function (or *kernel function*):

$$k(u) = \begin{cases} 1, & |u_i| \leq 1/2, i = 1, ..., d \\ 0, & \text{otherwise} \end{cases}$$

Parzen window

Assume region $\mathcal{R}_n$ is d-dimensional hypercube centred around point $\mathbf{x}$. The effective volume of hypercube would be $V_n = h_n^d$.

Define window function (or *kernel function*):

$$k(u) = \begin{cases} 1, & |u_i| \leq 1/2, i = 1, ..., d \\ 0, & \text{otherwise} \end{cases}$$

- $k(u)$ defines an *unit hypercube* centred at the origin.

Parzen window

Assume region $\mathcal{R}_n$ is d-dimensional hypercube centred around point $\mathbf{x}$. The effective volume of hypercube would be $V_n = h_n^d$.

Define window function (or *kernel function*):

$$k(u) = \begin{cases} 1, & |u_i| \leq 1/2, i = 1, ..., d \\ 0, & \text{otherwise} \end{cases}$$

- $k(u)$ defines an *unit hypercube* centred at the origin.
- $k(\frac{\mathbf{x}-\mathbf{x}_i}{h_n})$ will be 1 if the data point $\mathbf{x}_i$ lies inside a cube of side h_n centred on $\mathbf{x}$ and zero otherwise.

Parzen window

The total number of data points lying inside this hypercube will therefore be:

$$k_n = \sum_{i=1}^{n} k(\frac{\mathbf{x} - \mathbf{x}_i}{h_n})$$

Parzen window

The total number of data points lying inside this hypercube will therefore be:

$$k_n = \sum_{i=1}^{n} k\left(\frac{\mathbf{x} - \mathbf{x}_i}{h_n}\right)$$

Recall density estimation:

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

Parzen window

The total number of data points lying inside this hypercube will therefore be:

$$k_n = \sum_{i=1}^{n} k(\frac{\mathbf{x} - \mathbf{x}_i}{h_n})$$

Recall density estimation:

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

Plug in expression for k_n: $(V_n = h^d)$

$$p_n(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{V_n} k(\frac{\mathbf{x} - \mathbf{x}_i}{h_n})$$

Parzen window

The total number of data points lying inside this hypercube will therefore be:

$$k_n = \sum_{i=1}^{n} k(\frac{\mathbf{x} - \mathbf{x}_i}{h_n})$$

Recall density estimation:

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

Plug in expression for k_n: $(V_n = h^d)$

$$p_n(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{V_n} k(\frac{\mathbf{x} - \mathbf{x}_i}{h_n})$$

The window function is used for **interpolation** – each sample contributing to the estimation according to its distance from $\mathbf{x}$.

Example in 1D

Suppose we have 7 samples $D = \{2, 3, 4, 8, 10, 11, 12\}$, window width $h = 3$, estimate the density at $x = 1$.

Choose $k(u)$

Want the estimation $p_n(\mathbf{x})$ to be legitimate density function. This can be assured by requiring $k(u)$ to be a density function, precisely:

$$
\begin{aligned}
k(u) &\geq 0 \\
\int_{\mathcal{R}^d} k(u)du &= 1
\end{aligned}
$$

Choose $k(u)$

Want the estimation $p_n(\mathbf{x})$ to be legitimate density function. This can be assured by requiring $k(u)$ to be a density function, precisely:

$$\begin{aligned} k(u) &\geq 0 \\ \int_{\mathcal{R}^d} k(u)du &= 1 \end{aligned}$$

- Unit hypercube $k(u)$ satisfies these conditions.

Choose $k(u)$

Want the estimation $p_n(\mathbf{x})$ to be legitimate density function. This can be assured by requiring $k(u)$ to be a density function, precisely:

$$\begin{aligned} k(u) &\geq 0 \\ \int_{\mathcal{R}^d} k(u)du &= 1 \end{aligned}$$

- Unit hypercube $k(u)$ satisfies these conditions.
- Gaussian kernel $k(u)$ is also possible.

Parzen window (Gaussian)

Use *smoother* window function: Gaussian

$$k(u) = \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{u^2}{2}\right)$$

Then:

$$p_n(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{(2\pi h^2)^{d/2}} \exp\{-\frac{||\mathbf{x} - \mathbf{x}_i||^2}{2h_n^2}\}$$

Parzen window (Gaussian)

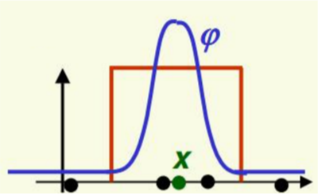
Use *smoother* window function: Gaussian

$$k(u) = \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{u^2}{2}\right)$$

Then:

$$p_n(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{(2\pi h^2)^{d/2}} \exp\left\{-\frac{||\mathbf{x}-\mathbf{x}_i||^2}{2h_n^2}\right\}$$

- Essentially, take a Gaussian centered at each data point and representing the unknown density as a mixture of these Gaussians. Counting the weighted average of potentially every single sample point.



Example in 1D using Gaussian window

- 7 samples $D = \{2, 3, 4, 8, 10, 11, 12\}$, $h = 1$

Example in 1D using Gaussian window

- 7 samples $D = \{2, 3, 4, 8, 10, 11, 12\}$, $h = 1$

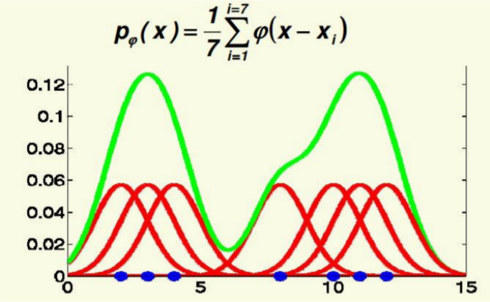


Figure: [source]

Example in 1D using Gaussian window

- 7 samples $D = \{2, 3, 4, 8, 10, 11, 12\}$, $h = 1$

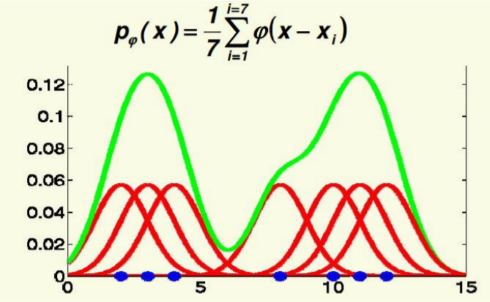


Figure: [source]

- The density is estimated by summation of 7 Gaussians, each centered at one of the sample point, and scaled by $\frac{1}{7}$

The effect of h_n on $p_n(\mathbf{x})$

Recall the estimation: $p_n(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{V_n}k(\frac{\mathbf{x}-\mathbf{x}_i}{h_n})$
Let $\delta_n(\mathbf{x}) = \frac{1}{V_n}k(\frac{\mathbf{x}}{h_n})$, then:

$$p_n(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\delta_n(\mathbf{x}-\mathbf{x}_i)$$

The effect of h_n on $p_n(\mathbf{x})$

Recall the estimation: $p_n(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{V_n}k(\frac{\mathbf{x}-\mathbf{x}_i}{h_n})$
Let $\delta_n(\mathbf{x}) = \frac{1}{V_n}k(\frac{\mathbf{x}}{h_n})$, then:

$$p_n(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\delta_n(\mathbf{x}-\mathbf{x}_i)$$

- If h_n is too large, the density estimate $p_n(\mathbf{x})$ is a superposition of n *broad, slowly changing* functions, thus will be very smooth and "out-of-focus".

The effect of h_n on $p_n(\mathbf{x})$

Recall the estimation: $p_n(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{V_n} k(\frac{\mathbf{x}-\mathbf{x}_i}{h_n})$
Let $\delta_n(\mathbf{x}) = \frac{1}{V_n} k(\frac{\mathbf{x}}{h_n})$, then:

$$p_n(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} \delta_n(\mathbf{x}-\mathbf{x}_i)$$

- If h_n is too large, the density estimate $p_n(\mathbf{x})$ is a superposition of n *broad, slowly changing* functions, thus will be very smooth and "out-of-focus".
- If h_n is too small, the estimate $p_n(\mathbf{x})$ will be just superposition of n *sharp* pulses centered at training samples,

The effect of h_n on $p_n(\mathbf{x})$

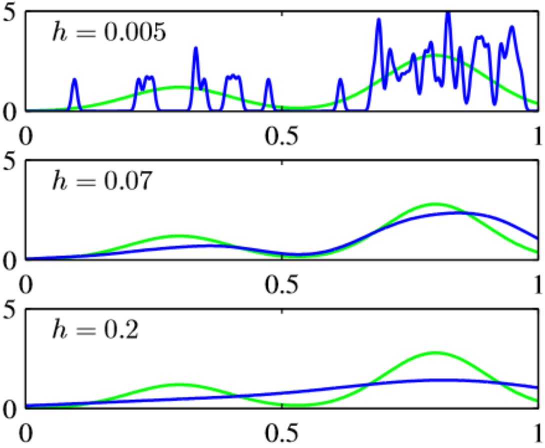


Figure: Green: underlying distribution. Blue: Parzen window estimated density using Gaussian [C. Bishop PRML]

Convergence of Parzen window estimator

- With an unlimited number of training samples it is possible to let V_n approach zero, and have $p_n(\mathbf{x})$ converge to $p(\mathbf{x})$.

Convergence of Parzen window estimator

- With an unlimited number of training samples it is possible to let V_n approach zero, and have $p_n(\mathbf{x})$ converge to $p(\mathbf{x})$.
- By convergence we mean the convergence in mean-variance sense i.e. that for all $\mathbf{x}$:
    1. $\lim_{n\to\infty} \mathbb{E}[p_n(\mathbf{x})] = p(\mathbf{x})$
    2. $\lim_{n\to\infty} \mathrm{Var}[p_n(\mathbf{x})] = 0$

Convergence of Parzen window estimator

- With an unlimited number of training samples it is possible to let V_n approach zero, and have $p_n(\mathbf{x})$ converge to $p(\mathbf{x})$.
- By convergence we mean the convergence in mean-variance sense i.e. that for all $\mathbf{x}$:
    1. $\lim_{n\to\infty} \mathbb{E}[p_n(\mathbf{x})] = p(\mathbf{x})$
    2. $\lim_{n\to\infty} \mathrm{Var}[p_n(\mathbf{x})] = 0$
- Expectation and variance are taken with respect to the sequence (of length n) of training samples.

Convergence of Parzen window estimator

Must place conditions on the unknown density $p(\mathbf{x})$, on the window function $k(u)$, and on the window width h_n:

- The density function $p(\mathbf{x})$ must be continuous.
- The window function $k(u)$ must be legitimate density.
- The values of the window function must be negligible at infinity. (check chapter 4.3 in *Pattern Classification* for details)
- $\lim_{n\to\infty} V_n = 0$
- $\lim_{n\to\infty} nV_n = \infty$

V_n must approach zero, but at a rate slower than $1/n$. E.g., $V_n = V_1/\sqrt{n}$

Example: univariate Gaussian

Underlying distribution is univariate Gaussian, the window volume decreases as n increase. $V_n = V_1/\sqrt{n}$

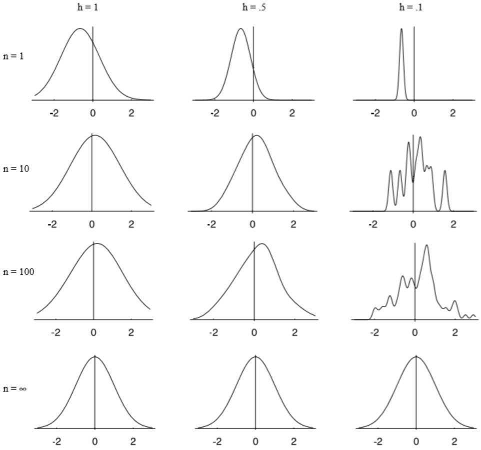


Figure: [R. Duda Pattern Classification]

Example: multi-modality

Underlying distribution is two modal, the window volume decreases as n increase. $V_n = V_1/\sqrt{n}$

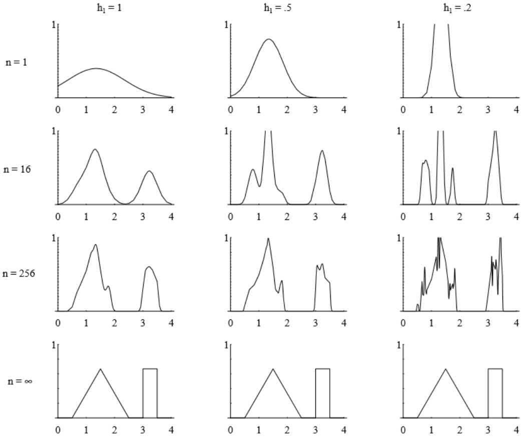


Figure: [R. Duda Pattern Classification]

Parzen window classification

- Estimate the density for each category $p(\mathbf{x}|C_i)$ using Parzen window (kernel density estimation), and class prior $p(C_i)$.

Parzen window classification

- Estimate the density for each category $p(\mathbf{x}|C_i)$ using Parzen window (kernel density estimation), and class prior $p(C_i)$.
- Classify the test point by the class label that has maximum posterior $p(C_i|\mathbf{x})$ through Bayes formula.

Parzen window classification

- Estimate the density for each category $p(\mathbf{x}|C_i)$ using Parzen window (kernel density estimation), and class prior $p(C_i)$.
- Classify the test point by the class label that has maximum posterior $p(C_i|\mathbf{x})$ through Bayes formula.
- The decision region for a Parzen window classifier depends upon the choice of window function and window width.

Parzen window: summary

- Advantages:

Parzen window: summary

- Advantages:
  - Can be applied to the data from any distribution.

Parzen window: summary

- Advantages:
  - Can be applied to the data from any distribution.
  - Theoretically, shown to converge as the number of samples goes to infinity.
- Disadvantages:

Parzen window: summary

- Advantages:
  - Can be applied to the data from any distribution.
  - Theoretically, shown to converge as the number of samples goes to infinity.
- Disadvantages:
  - In practice, the number of samples n is finite, so choose proper window size h_n can be difficult.

Parzen window: summary

- Advantages:
  - Can be applied to the data from any distribution.
  - Theoretically, shown to converge as the number of samples goes to infinity.
- Disadvantages:
  - In practice, the number of samples n is finite, so choose proper window size h_n can be difficult.
  - Computationally heavy. E.g., if we use Gaussian window function, at any $\mathbf{x}$, we need to compute n Gaussians.

K nearest neighbour

Parzen window vs K nearest neighbour

Recall density estimation:

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

Parzen window vs K nearest neighbour

Recall density estimation:

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

- Parzen window: fix the window size, e.g., $V_n = \frac{V_1}{\sqrt{n}}$, and $k_n = k_n(\mathbf{x})$ is function of $\mathbf{x}$:

$$p_n(\mathbf{x}) = \frac{k_n/n}{V_n} = \frac{k_n(\mathbf{x})}{V_1\sqrt{n}}$$

Parzen window vs K nearest neighbour

Recall density estimation:

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

- Parzen window: fix the window size, e.g., $V_n = \frac{V_1}{\sqrt{n}}$, and $k_n = k_n(\mathbf{x})$ is function of $\mathbf{x}$:

$$p_n(\mathbf{x}) = \frac{k_n/n}{V_n} = \frac{k_n(\mathbf{x})}{V_1\sqrt{n}}$$

- K NN: fix the number of sample inside window, e.g., $k_n = \sqrt{n}$, and $V_n = V_n(\mathbf{x})$ is a function of $\mathbf{x}$:

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

Parzen window vs K nearest neighbour

Parzen window: $V_n = \frac{V_1}{\sqrt{n}}$

$$p_n(\mathbf{x}) = \frac{k_n/n}{V_n} = \frac{k_n(\mathbf{x})}{V_1\sqrt{n}}$$

- The number of samples falling in a window can be counted explicitly:

$$k_n = \sum_{i=1}^{n} k(\frac{\mathbf{x} - \mathbf{x}_i}{h_n})$$

K nearest neighbour: $k_n = \sqrt{n}$

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

- The window size does not have an explicitly form, the diameter of the window is:

$$\begin{aligned} \phi(V_n(\mathbf{x})) &= 2|\mathbf{x} - \mathbf{x}_K^\star| \\ \mathbf{x}_K^\star &\in \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \end{aligned}$$

K nearest neighbour for density estimation

A good "rule of thumb" to choose K for n samples: $k_n = \sqrt{n}$.

K nearest neighbour for density estimation

A good "rule of thumb" to choose K for n samples: $k_n = \sqrt{n}$.

- Can prove the convergence if $n \to \infty$

K nearest neighbour for density estimation

A good "rule of thumb" to choose K for n samples: $k_n = \sqrt{n}$.

- Can prove the convergence if $n \to \infty$
- However, not too useful in practice.

K nearest neighbour for density estimation

A good "rule of thumb" to choose K for n samples: $k_n = \sqrt{n}$.

- Can prove the convergence if $n \to \infty$
- However, not too useful in practice.

$1D$ example: suppose we only have 1 sample x_1 $(n = 1)$. The estimated density is:

$$p_n(\mathbf{x}) = \frac{k_n/n}{V_n} = \frac{1}{2|x - x_1|}$$

However, it is poor estimate, not even a valid density function, i.e.,

$$\int \frac{1}{2|x - x_1|} dx = \infty \neq 1$$

Different K for density estimation

K governs the degree of smoothing.

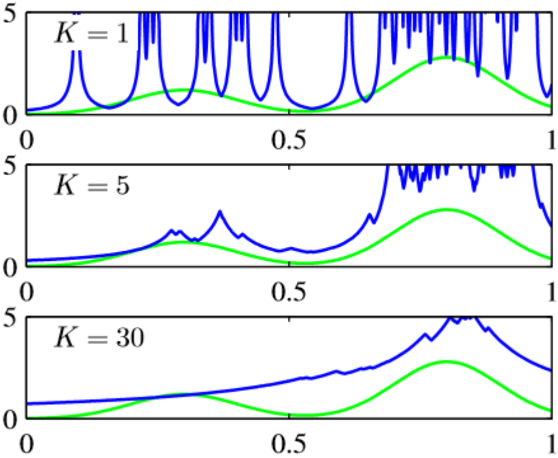


Figure: [C. Bishop PRML]

Example

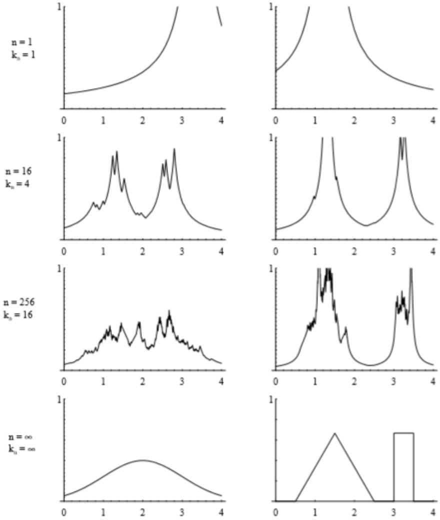


Figure: [R. Duda, Pattern Classification]

KNN for density estimation

The obtained density estimation $p_n(\mathbf{x})$ does not work very well using KNN, since

- the estimated density model is not a valid density.
- the estimated density has lots of discontinuities.

KNN for density estimation

The obtained density estimation $p_n(\mathbf{x})$ does not work very well using KNN, since

- the estimated density model is not a valid density.
- the estimated density has lots of discontinuities.

KNN for classification?

Multi-Class Classification - Posterior Probability

- Assuming: n_j points in class C_j with n points in total, so that $\sum_j n_j = n$. To classify a new point $\mathbf{x}$, we draw a sphere centred on $\mathbf{x}$ containing k points irrespective of their class. Suppose this sphere has volume V_n and contains k_j points from class C_j.

Multi-Class Classification - Posterior Probability

- Assuming: n_j points in class C_j with n points in total, so that $\sum_j n_j = n$. To classify a new point $\mathbf{x}$, we draw a sphere centred on $\mathbf{x}$ containing k points irrespective of their class. Suppose this sphere has volume V_n and contains k_j points from class C_j.
- An estimate of the density associated with each class

$$p(\mathbf{x}|C_j) = \frac{k_j/n_j}{V}$$

Multi-Class Classification - Posterior Probability

- Assuming: n_j points in class C_j with n points in total, so that $\sum_j n_j = n$. To classify a new point $\mathbf{x}$, we draw a sphere centred on $\mathbf{x}$ containing k points irrespective of their class. Suppose this sphere has volume V_n and contains k_j points from class C_j.
- An estimate of the density associated with each class

$$p(\mathbf{x}|C_j) = \frac{k_j/n_j}{V}$$

- The unconditional density is given by:

$$p(\mathbf{x}) = \frac{k/n}{V}$$

Multi-Class Classification - Posterior Probability

- Assuming: n_j points in class C_j with n points in total, so that $\sum_j n_j = n$. To classify a new point $\mathbf{x}$, we draw a sphere centred on $\mathbf{x}$ containing k points irrespective of their class. Suppose this sphere has volume V_n and contains k_j points from class C_j.
- An estimate of the density associated with each class

$$p(\mathbf{x}|C_j) = \frac{k_j/n_j}{V}$$

- The unconditional density is given by:

$$p(\mathbf{x}) = \frac{k/n}{V}$$

- The class priors are: $p(C_j) = \frac{n_j}{n}$

Multi-Class Classification - Posterior Probability

- Assuming: n_j points in class C_j with n points in total, so that $\sum_j n_j = n$. To classify a new point $\mathbf{x}$, we draw a sphere centred on $\mathbf{x}$ containing k points irrespective of their class. Suppose this sphere has volume V_n and contains k_j points from class C_j.
- An estimate of the density associated with each class

$$p(\mathbf{x}|C_j) = \frac{k_j/n_j}{V}$$

- The unconditional density is given by:

$$p(\mathbf{x}) = \frac{k/n}{V}$$

- The class priors are: $p(C_j) = \frac{n_j}{n}$
- Using Bayes' theorem, the posterior probability of class membership: $p(C_j|\mathbf{x}) = \frac{p(\mathbf{x}|C_j)p(C_j)}{p(\mathbf{x})} = \frac{k_j}{k}$

KNN classification

- Posterior probability:

$$p(C_j|\mathbf{x}) = \frac{p(\mathbf{x}|C_j)p(C_j)}{p(\mathbf{x})} = \frac{k_j}{k}$$

–The posterior is just the fraction of the samples within the cell which belong to class C_j.

KNN classification

- Posterior probability:

$$p(C_j|\mathbf{x}) = \frac{p(\mathbf{x}|C_j)p(C_j)}{p(\mathbf{x})} = \frac{k_j}{k}$$

–The posterior is just the fraction of the samples within the cell which belong to class C_j.

- The **bayesian decision rule**: choose class $C^\star$ that has maximum posterior $p(C^\star|\mathbf{x})$. Equivalently:

$$C^\star = \arg\max_i \{k_1, k_2, \ldots, k_C\}$$

–Choose the class which has the largest number of samples in the cell (majority voting).

KNN classification illustration

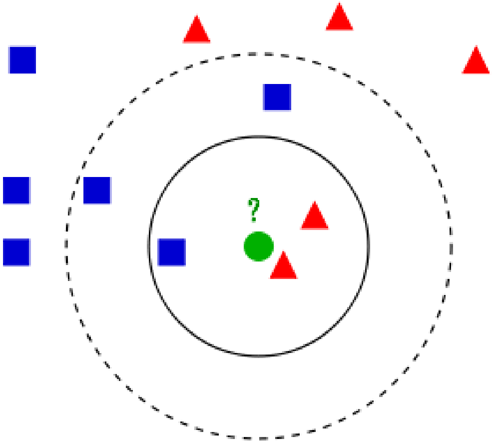


Figure: common values for k: 3,5

The nearest neighbour rule

- Let $\{(\mathbf{x}_1, \omega(\mathbf{x}_1), \ldots, (\mathbf{x}_n, \omega(\mathbf{x}_n))\}$ be labelled samples. For any test point $\mathbf{x}$:

$$\begin{aligned} \omega_{NN}(\mathbf{x}) &= \omega(\mathbf{x}^\star) \\ \mathbf{x}^\star &= \arg\min_j \{||\mathbf{x} - \mathbf{x}_j||, j = 1, \ldots, n\} \end{aligned}$$

The nearest neighbour rule

- Let $\{(\mathbf{x}_1, \omega(\mathbf{x}_1), \ldots, (\mathbf{x}_n, \omega(\mathbf{x}_n))\}$ be labelled samples. For any test point $\mathbf{x}$:

$$\begin{aligned} \omega_{NN}(\mathbf{x}) &= \omega(\mathbf{x}^\star) \\ \mathbf{x}^\star &= \arg\min_j \{||\mathbf{x} - \mathbf{x}_j||, j = 1, \ldots, n\} \end{aligned}$$

- Partition the space by *Voronoi diagram*.

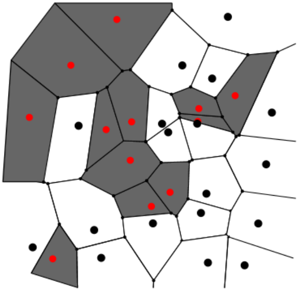


Figure: [R. Duda, Pattern Classification]

Choose k for classification

k controls the degree of smoothing: small k produces many small regions of each class, large k leads to fewer larger regions.

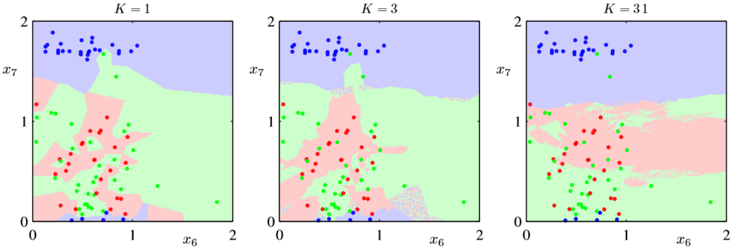


Figure: [R. Duda, Pattern Classification]

Choose k for classification

- k should be large so that error rate is minimized, better classification. (too small k leads to noisy decision boundaries)

Choose k for classification

- k should be large so that error rate is minimized, better classification. (too small k leads to noisy decision boundaries)
- k should be small enough so that only nearby samples are included. (too large k will lead to over smoothed boundaries)

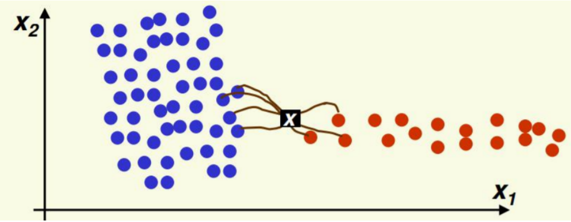


Figure: $k \leq 7$, $\mathbf{x}$ correctly classified. $k > 7$, $\mathbf{x}$ misclassified. [Source]

Distance measure

- Euclidean distance:

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{\sum_{m=1}^{D} (x_{im} - x_{jm})^2}$$

- Similarity: dot product
- Binary valued features: Hamming distince:

$$d(\mathbf{x}_i, \mathbf{x}_j) = \sum_{m=1}^{D} \mathbb{I}(x_{im} \neq x_{jm})$$

- Can assign weights to features:
  $d(\mathbf{x}_i, \mathbf{x}_j) = \sum_{m=1}^{D} w_m d(x_{im}, x_{jm})$

Distance may sensitive to transformations

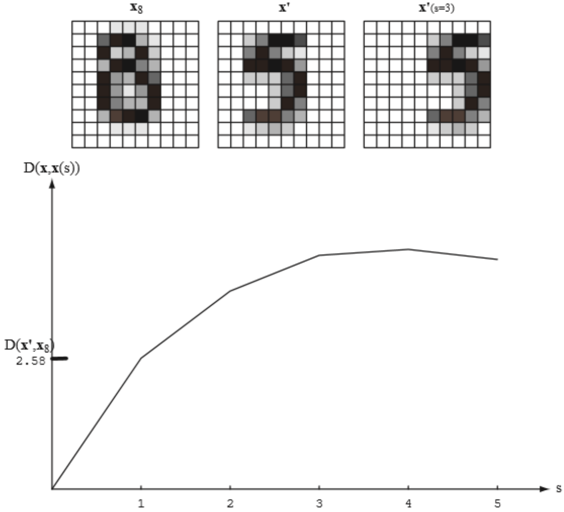


Figure: [R. Duda, Pattern Classification]

K nearest neighbour: summary

- Learning algorithm:
  - Store training examples

K nearest neighbour: summary

- Learning algorithm:
  - Store training examples
- Prediction algorithm: Look at the k (nearest neighbors).
  - Classification: assign the majority class label (majority voting) of these k neighbours.
  - Regression: assign average response of these k neighbours.

K nearest neighbour: summary

- Learning algorithm:
  - Store training examples
- Prediction algorithm: Look at the k (nearest neighbors).
  - Classification: assign the majority class label (majority voting) of these k neighbours.
  - Regression: assign average response of these k neighbours.
- **Limitations**:
  - Require the entire training data set to be stored, take a lot of memory.
  - Expensive computation if the data set if large.
  - Need to specify the distance function.

Acknowledgement and Further Reading

Part of the materials are based on lecture 17, 18 of S.C. Zhu's Stats 231, *Pattern Recognition and Machine Learning*.

A few slides are taken from Dr. Y. Ning's Spring 19 offering of CS-559.

Some examples are taken from machine learning slides of [Chengjiang Long]

Further Reading:
Chapter 2.5 of *Pattern Recognition and Machine Learning* by C. Bishop.

Chapter 4.1-4.5 of *Pattern Classification* by R.Duda.