# Chapter 6 Nonparametric Techniques

## Introduction

- The assumed parametric form may not fit the ground-truth density encountered in practice
- $p\left(\mathbf{x} \mid \omega_j\right)$ doesn't have parametric form

**Two methods**

- Parzen Windows
- K-nearest-neighbor

## Density Estimation

- Feature space: $\mathcal{F}=\mathbf{R}^d$
- Feature vector: $\mathbf{x} \in \mathcal{F}$
- pdf function: $\mathbf{x} \sim p(\cdot)$
- The probability of a vector $\mathbf{x}$ falling into a region $\mathcal{R} \subset \mathcal{F}$: $P$

$$
\begin{gathered}
P=\operatorname{Pr}[\mathbf{x} \in \mathcal{R}]=\int_{\mathcal{R}} p\left(\mathbf{x}^{\prime}\right) d \mathbf{x}^{\prime} \simeq p(\mathbf{x}) \int_{\mathcal{R}} 1 d \mathbf{x}^{\prime} \\
P \simeq p(\mathbf{x}) V
\end{gathered}
$$

- the number of examples falling into $\mathcal{R}$ : $X$

$$
\begin{gathered}
X \sim \mathcal{B}(n, P) \\
P=\frac{\mathcal{E}[X]}{n}
\end{gathered}
$$

- from above we get

$$
p(\mathbf{x})=\frac{\mathcal{E}[X] / n}{V}=\frac{k / n}{V}
$$

  where we have Let $k$ to be the actual value of $X$ after observing the i.i.d. training examples $\left\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\right\}$
- To show the explicit relationships with n:

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

  where

  - $V_n$ : volume of $\mathcal{R}_n$
  - $n$ : # training examples
  - $k_n$ : training examples falling within $\mathcal{R}_n$
- Fix $V_n$ and determine $k_n$ $\rightarrow$ Parzen Windows
- Fix $k_n$ and determine $V_n$ $\rightarrow$ k-nearest-neighbor

## Parzen Windows

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

- Fix $V_n$, and then determine $k_n$

## Window function

- Assume $\mathcal{R}_n$ is a $d$-dimensional hypercube (超立方体)
- The length of each edge is $h_n$
- $V_n = h_n^d$

$$\varphi(\mathbf{u}) = \begin{cases} 1 & |u_j| \leq 1/2; \quad j = 1, \dots, d \\ 0 & \text{otherwise} \end{cases}$$

$\varphi(\mathbf{u})$ defines a unit hypercube **centered at the origin**

so we have

$$\varphi\left(\frac{\mathbf{x} - \mathbf{x}_i}{h_n}\right) = 1$$

which means $\mathbf{x}_i$ falls within the hypercube of volume $V_n$ centered at $\mathbf{x}$

then

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

so we have $p_n(x)$, an average of functions of $\mathbf{x}$ and $\mathbf{x_i}$

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

Note that $\varphi(\cdot)$ is not limited to be the hypercube window function, it could be **any pdf function**

window function (being pdf) $\varphi(\cdot)$ + window width $h_n$ + training data $\mathbf{x}_i$ $\Rightarrow$ Parzen pdf $p_n(\cdot)$

let

$$\delta_n(\mathbf{x}) = \frac{1}{V_n} \varphi\left(\frac{\mathbf{x}}{h_n}\right)$$

then

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

here

- $p_n(\mathbf{x})$ : superposition (叠加) of $n$ interpolations (插值)
- $\mathbf{x}_i$ : contributes to $p_n(\mathbf{x})$ based on its "distance" from $\mathbf{x}$ (i.e. " $\mathbf{x} - \mathbf{x}_i$")

## The effect of $h_n$ (window width)

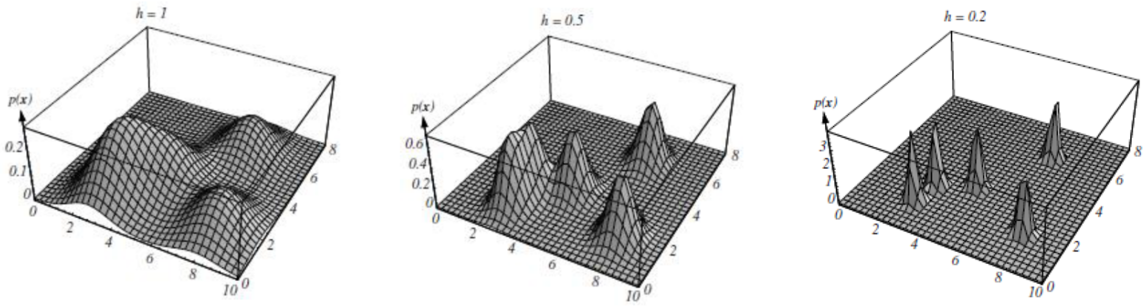
$$\delta_n(\mathbf{x}) = \frac{1}{V_n} \varphi\left(\frac{\mathbf{x}}{h_n}\right) = \frac{1}{h_n^d} \varphi\left(\frac{\mathbf{x}}{h_n}\right)$$

here

- $\frac{1}{h_n^d}$ Affects the amplitude (vertical scale, 幅度)
- $\frac{\mathbf{x}}{h_n}$ Affects the width (horizontal scale, 宽度)

- $h_n$ very large $\rightarrow \delta_n(\mathbf{x})$ being broad with small amplitude
- $h_n$ very small $\rightarrow \delta_n(\mathbf{x})$ being sharp with large amplitude
- A compromised value (折衷值) of $h_n$ should be sought for limited number of training examples

The shape of $p_n(x)$ with decreasing values of $h_n$




$h$=1.0 $h$=0.5 $h$=0.2

## $k_n$-Nearest-Neighbor

Fix $k_n$, and then determine $V_n$

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

### Step

- specify $k_n$
- center a cell about $\mathbf{x}$
- grow the cell until
  - the cell can be any shape: e.g. hypercube/sphere
- capturing $k_n$ nearest examples
- return cell volume as $V_n$

### The principled rule to specify $\mathrm{k}_n$

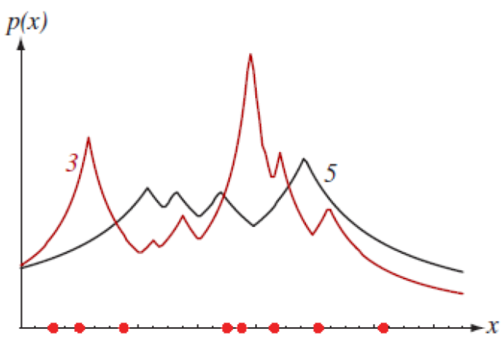
A rule-of-thumb choice for $k_n$ :

$$k_n = \sqrt{n}$$

Eight points in one dimension ($n$=8, $d$=1)

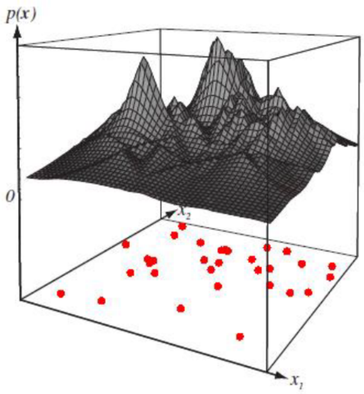
red curve: $k_n$=3

black curve: $k_n$=5



Thirty-one points in two dimensions ($n$=31, $d$=2)

black surface: $k_n$=5




**Note that they may not be normalized!**

# Nearest Neighbor Rule

## Nearest-Neighbor (NN) Rule

- Given the label space $\Omega = \{\omega_1, \omega_2, \ldots, \omega_c\}$ and a set of $n$ labeled training examples $\mathcal{D}^n = \{(\mathbf{x}_i, \theta_i) \mid 1 \leq i \leq n\}$, where $\mathbf{x}_i \in \mathbf{R}^d$ and $\theta_i \in \Omega$
- for test example $\mathbf{x}$, identify $\mathbf{x}' = \operatorname{argmin}_{\mathbf{x}_i \in \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}} D(\mathbf{x}_i, \mathbf{x})$ and then assign the label $\theta'$ associated with $\mathbf{x}'$ to $\mathbf{x}$
- $D(\mathbf{a}, \mathbf{b})$ : distance metric between two vectors $\mathbf{a}$ and $\mathbf{b}$, e.g. the Euclidean distance

### Voronoi tessellation

- Each training example $x$ leads to a cell in the Voronoi tessellation
- any point in the cell is closer to $\mathbf{x}$ than to any other training examples
- partition the feature space into $n$ cells
- any point in the cell shares the same class label as $\mathbf{x}$

### Error bounds of nearest neighbor rule

$$P^*(e) \leq P(e) \leq P^*(e)\left(2 - \frac{c}{c-1} P^*(e)\right)$$

where

- $c$ : the number of class labels
- $P(e \mid \mathbf{x})$ : The probability of making an erroneous classification on $\mathbf{x}$ based on nearest-neighbor rule
- $P(e)$ : The **average probability of error** based on nearest-neighbor rule: $P(e) = \int P(e \mid \mathbf{x}) p(\mathbf{x}) d\mathbf{x}$
- $P^*(e \mid \mathbf{x})$ : The **minimum** possible value of $P(e \mid \mathbf{x})$, i.e. the one given by Bayesian decision rule:

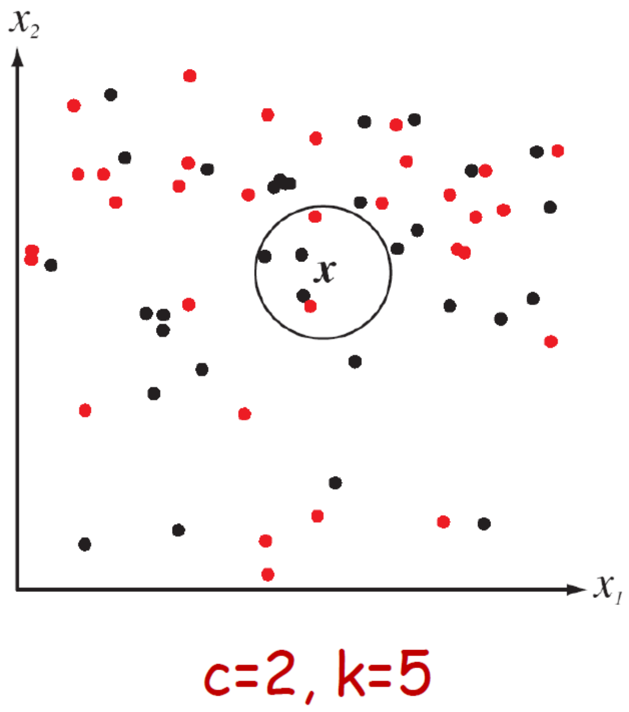
$$P^*(e \mid \mathbf{x}) = 1 - \max_{1<i<c} P(\omega_i \mid \mathbf{x})$$

- $P^*(e)$ : The **Bayes risk** (**under zero-one loss**): $P^*(e) = \int P^*(e \mid \mathbf{x}) p(\mathbf{x}) d\mathbf{x}$

## k-Nearest-Neighbor (kNN) Rule

- Given the label space $\Omega = \{\omega_1, \omega_2, \ldots, \omega_c\}$ and a set of $n$ labeled training examples $\mathcal{D}^n = \{(\mathbf{x}_i, \theta_i) \mid 1 \leq i \leq n\}$, where $\mathbf{x}_i \in \mathbf{R}^d$ and $\theta_i \in \Omega$
- for test example $\mathbf{x}$, identify $S' = \{\mathbf{x}_i \mid \mathbf{x}_i$ is among the $k\mathrm{NN}$ of $\mathbf{x}\}$ and then **assign the most frequent label** w.r.t. $S'$, i.e. $\arg\max_{\omega_i \in \Omega} \sum_{\mathbf{x}_i \in S'} 1_{\theta_i = \omega_i}$ to $\mathbf{x}$

- $1_\pi$ : an indicator function which returns 1 if predicate $\pi$ holds, and 0 otherwise
- For binary classification problem $(c = 2)$, an odd value of $\mathbf{k}$ is generally used to avoid ties




#### Computational Complexity

- Pre-structuring:
  - create some form of search tree, where nearest neighbors are recursively identified following the tree structure
- Editing/Pruning/Condensing:
  - eliminate “**redundant**” (“useless”) examples from the training set
  - e.g. example surrounded by training examples of the same class label

## Distance Metric

### Four properties

- non-negativity: $D(\mathbf{a}, \mathbf{b}) \geq 0$
- reflexivity: $D(\mathbf{a}, \mathbf{b}) = 0$ if and only if $\mathbf{a} = \mathbf{b}$
- symmetry: $D(\mathbf{a}, \mathbf{b}) = D(\mathbf{b}, \mathbf{a})$
- triangle inequality: $D(\mathbf{a}, \mathbf{b}) + D(\mathbf{b}, \mathbf{c}) \geq D(\mathbf{a}, \mathbf{c})$

### About Euclidean distance

#### Problem

Scaling the features $\longrightarrow$ **change the distance relationship**

#### Solution

**normalize** each feature into equal-sized intervals, e.g. [0,1]

## Types

### $L_k$ norm

$$L_k(\mathbf{a},\mathbf{b}) = \left(\sum_{j=1}^{d} |a_j - b_j|^k\right)^{\frac{1}{k}} \qquad (k>0)$$

- k=2: Euclidean distance
- k=1: Manhattan distance (city block distance)、

$$L_1(\mathbf{a},\mathbf{b}) = \sum_{j=1}^{d} |a_j - b_j|$$

- $k=\infty : L_\infty$ distance

$$L_\infty(\mathbf{a},\mathbf{b}) = \max_{1\le j\le d} |a_j - b_j|$$

#### Tanimoto distance

$$D_{\text{Tanimoto}}\ (S_1,S_2) = \frac{n_1+n_2-2n_{12}}{n_1+n_2-n_{12}}$$
$$(n_1 = |S_1|\,, n_2 = |S_2|\,, n_{12} = |S_1 \cap S_2|)$$

Example: treat each word as a set of characters

*Which word out of 'cat', 'pots' and 'patches' mostly resembles 'pat'?*

*cat*

$$S_1 = \{p,a,t\}$$
$$S_2 = \{c,a,t\}$$
$$S_3 = \{p,o,t,s\}$$
$$S_4 = \{p,a,t,c,h,e,s\}$$

$$D_{Tanimoto}(S_1,S_2) = \frac{3+3-2*2}{3+3-2} = 0.5$$
$$D_{Tanimoto}(S_1,S_3) = \frac{3+4-2*2}{3+4-2} = 0.6$$
$$D_{Tanimoto}(S_1,S_4) = \frac{3+7-2*3}{3+7-3} = 0.571$$

#### Hausdorff distance

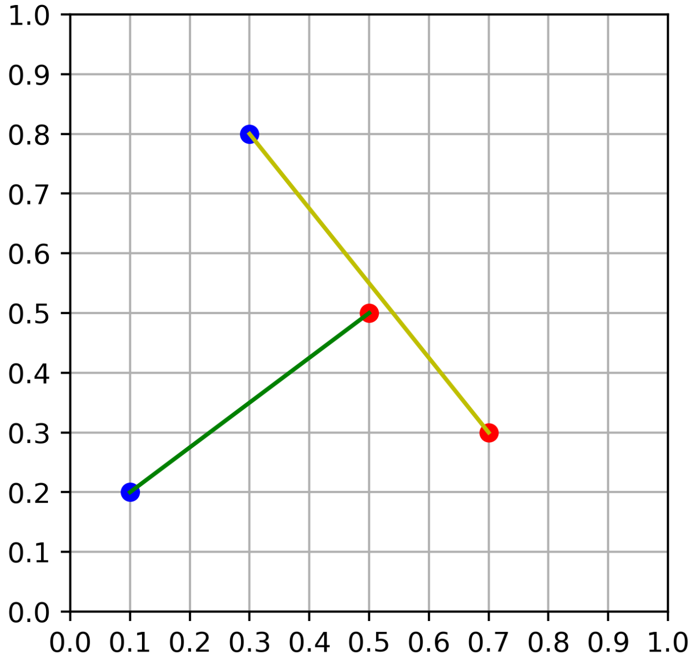
$$D_H\,(S_1,S_2) = \max\left(\max_{\mathbf{s}_1\in S_1 \mathbf{s}_2\in S_2} \min_{D}\,(\mathbf{s}_1,\mathbf{s}_2)\,, \max_{\mathbf{s}_2\in S_2} \min_{\mathbf{s}_1\in S_1} D\,(\mathbf{s}_2,\mathbf{s}_1)\right)$$

where $D\,(\mathbf{s}_1,\mathbf{s}_2)$ is any distance metric between $\mathbf{s}_1$ and $\mathbf{s}_2$

e.g. Hausdorff distance between two sets of feature vectors

$$S_1 = \{(0.1,0.2)^t,\ (0.3,0.8)^t\} \quad S_2 = \{(0.5,0.5)^t,(0.7,0.3)^t\}$$

$$\begin{aligned} D_H\,(S_1,S_2) &= \max(\max(0.5,0.36),\max(0.36,0.61)) \\ &= \max(0.5,0.61) \\ &= 0.61 \end{aligned}$$



1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0