Lecture Notes on

Pattern Recognition

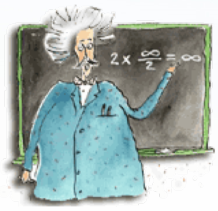
Session 05(A): Non-Parametric Estimation (I)

Gang Li
School of Information Technology
Deakin University, VIC 3125, Australia

Road map

- Non-Parametric Estimation
  - Non-Parametric Estimation
    - Histogram
  - Density Estimation
    - Two Assumptions
    - Density Estimation
  - Parzen-Window Estimation
    - Window Function
    - Parzen-Window Estimation
    - Convergence
  - Probabilistic Neural Network

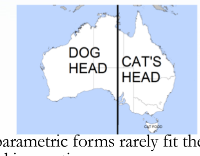
Non-Parametric Estimation



- Non-Parameter Estimation
- Histogram

Motivation

- Classical parametric densities are ***unimodal***.
  - Many practical problems involve ***multimodal*** densities.
    - Example: model probability p(x,y) for a raindrop to hit a location(x,y) in Australia



    - Common parametric forms rarely fit the densities actually encountered in practice.

Non-Parameter Estimation

- Based on a set of observations
  - Estimate **class-conditional densities**

$$p(\mathbf{x} \mid \omega_i)$$

  - Estimate **posterior probabilities**

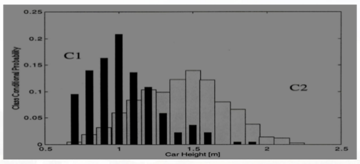
$$P(\omega_i \mid \mathbf{x})$$

Non-Parameter Estimation

- Task: model the likelihood function without making any assumption about its functional form.
  - Need to deal with the choice of smoothing parameters that govern the smoothness of the estimated density.
- Three main types of methods:
  - Histograms
  - Parzen Window (Kernels)
  - K-nearest neighbors

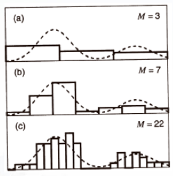
Histogram

- Suppose each data point **x** is represented by an n-dimensional feature vector $(x_1, x_2, \ldots, x_n)$.
- The histogram is obtained by dividing each x_i-axis into a number of bins M and approximating the density at each value of x_i by the ***fraction*** of the points that fall inside the corresponding bin.



Histogram

- The number of bins M (or bin size) is acting as a **smoothing** parameter.
  - If bin width is ***small*** (big M), then the estimated density is very spiky (i.e., noisy).
  - If bin width is ***large*** (small M), then the true structure of the density is smoothed out.
- In practice, we need to find an optimal value for M that compromises between these two issues.



Histogram

Pros

- Once the histogram has been constructed, the data is not needed anymore
  - i.e., memory efficient
- Retain only info on the size and location of histogram bins.
- Histogram can be built sequentially
  - i.e., consider the data one at a time and then discard

Cons

- The estimated density is not smooth and has discontinuities at the boundaries of the histogram bins.
- Histograms do not generalize well in high dimensions.
  - Consider a d-dimensional feature space; if we divide each variable in M intervals, we will end up with M^d bins.
  - A huge number of examples would be required to obtain good estimates (i.e., otherwise, most bins would be empty and the density will be approximated by zero).

Density Estimation

- Two Approximations
- Density Estimation

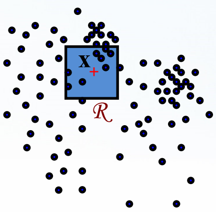
Density Estimation

- Approximation 1: Assume **p(x)** is continuous & R is small

$$P(\mathbf{X} \in \mathcal{R}) = \int_{\mathcal{R}} p(\mathbf{x}')d\mathbf{x}' = p(\mathbf{x})\int_{\mathcal{R}} d\mathbf{x}'$$
$$= p(\mathbf{x})V_{\mathcal{R}} = P_{\mathcal{R}}$$

- Approximation 2: Randomly take **n** samples, let **K** denote the number of samples inside ***R***.

$$\Rightarrow K \sim B(n, P_{\mathcal{R}})$$
$$P(K = k) = \binom{n}{k} P_{\mathcal{R}}^{\,k}(1-P_{\mathcal{R}})^{n-k}$$



n samples

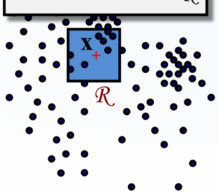
Density Estimation

$$E[K/n] = P_{\mathcal{R}}$$

- Approximation 1: Assume **p(x)** is continuous & R is small

$$P(\mathbf{X} \in \mathcal{R}) = \int_{\mathcal{R}} p(\mathbf{x}')d\mathbf{x}' = p(\mathbf{x})\int_{\mathcal{R}} d\mathbf{x}'$$
$$= p(\mathbf{x})V_{\mathcal{R}} = P_{\mathcal{R}}$$

- Approximation 2: Randomly take **n** samples, let **K** denote the number of samples inside ***R***.
  - $E[k] = nP_{\mathcal{R}}$
  - $E[k/n] = P_{\mathcal{R}}$
  - $Var[k/n] = E[(k/n - P_{\mathcal{R}})^2] = \frac{P_{\mathcal{R}}(1-P_{\mathcal{R}})}{n}$



n samples

Density Estimation

- Combining these two approximations we have:

$$p(\mathbf{x}) \approx \frac{k_{\mathcal{R}}/n}{V_{\mathcal{R}}}$$

- The above approximation is based on ***contradictory*** assumptions:
  - R is relatively ***small*** so that $p(\mathbf{x})$ is approximately constant inside the integration region – **Approximation 1**
  - R is relatively ***large*** (i.e., it contains many samples so that P_k is sharply peaked) – **Approximation 2**
- We need to choose an optimum R in practice …

Notation

- Suppose we form regions R_1, R_2, ... containing $\mathbf{x}$.
  - R_1 contains 1 sample, R_2 contains 2 samples, etc.
  - R_i has volume V_i and contains k_i samples.
- The n-th estimate $p_n(\mathbf{x})$ of $p(\mathbf{x})$ is given by:

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

Main conditions for convergence

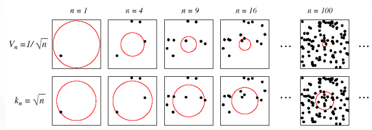
- The following conditions must be satisfied in order for $p_n(\mathbf{x})$ to converge to $p(\mathbf{x})$:

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

$\lim_{n\to\infty} V_n = 0$	• **Approximation 1**
$\lim_{n\to\infty} k_n = \infty$	• **Approximation 2**
$\lim_{n\to\infty} k_n/n = 0$	• **to allow $p_n(\mathbf{x})$ to converge**

Density Estimation

- How this process can be controlled?
  - choose the optimum values for V_n and k_n
- Two leading approaches:
  - Fix the volume V_n and determine k_n from the data
    - **Parzen Windows (Kernel-based Method)**
  - Control k_n
    - **KNN**



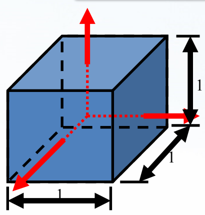
Parzen Windows

- Window Function
- Parzen Window Estimation
- Convergence

Window Function

$$\phi(\mathbf{u}) = \begin{cases} 1 & |u_j| \le 1/2, \quad j = 1,2,\ldots,d \\ 0 & \text{otherwise} \end{cases}$$

$$\boxed{\int \varphi(\mathbf{u})d\mathbf{u} = 1}$$

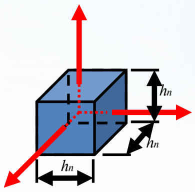


Window Function

$$\boxed{\int \varphi(\mathbf{u})d\mathbf{u}=1}$$

$$\phi(\mathbf{u})=\begin{cases}1 & |u_j|\le 1/2, \quad j=1,2,\ldots,d\\ 0 & \text{otherwise}\end{cases}$$

$$\varphi\left(\frac{\mathbf{x}}{h_n}\right)=\begin{cases}1 & |x_j|\le h_n/2\\ 0 & \text{otherwise}\end{cases}$$



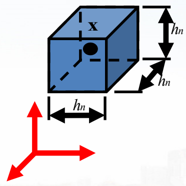
Window Function

$$\boxed{\int \varphi(\mathbf{u})d\mathbf{u}=1}$$

$$\phi(\mathbf{u})=\begin{cases}1 & |u_j|\le 1/2, \quad j=1,2,\ldots,d\\ 0 & \text{otherwise}\end{cases}$$

$$\varphi\left(\frac{\mathbf{x}}{h_n}\right)=\begin{cases}1 & |x_j|\le h_n/2\\ 0 & \text{otherwise}\end{cases}$$

$$\varphi\left(\frac{\mathbf{x}-\mathbf{x}'}{h_n}\right)=\begin{cases}1 & |x_j-x_j'|\le h_n/2\\ 0 & \text{otherwise}\end{cases}$$



Parzen-Window Estimation

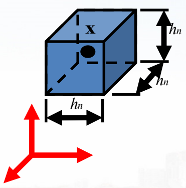
$$\boxed{\int \varphi(\mathbf{u})d\mathbf{u}=1}$$

$$\mathcal{D}=\{\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_n\}$$

k_n: # samples inside hypercube centered at $\mathbf{x}$.

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

$$\varphi\left(\frac{\mathbf{x}-\mathbf{x}'}{h_n}\right)=\begin{cases}1 & |x_j-x_j'|\le h_n/2\\ 0 & \text{otherwise}\end{cases}$$



Parzen Window Estimation

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

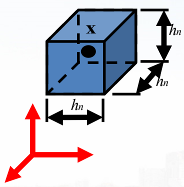
Parzen-Window Estimation

$$\boxed{\int \varphi(\mathbf{u})d\mathbf{u}=1}$$

The density estimate is a superposition of kernel functions and the samples $\mathbf{x}_i$.

- $\varphi(\mathbf{u})$ interpolates the density between samples.
- Each $\mathbf{x}_i$ contributes to the estimate based on its distance from x.

$$\varphi\left(\frac{\mathbf{x}-\mathbf{x}'}{h_n}\right)=\begin{cases}1 & |x_j-x_j'|\le h_n/2\\ 0 & \text{otherwise}\end{cases}$$



Parzen Window Estimation

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

Parzen-Window Estimation

$$\boxed{\int \varphi(\mathbf{u})d\mathbf{u}=1}$$

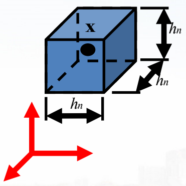
Prove that $\int p_n(\mathbf{x})d\mathbf{x}=1$

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

$$=\frac{1}{n}\sum_{i=1}^{n}\int\frac{1}{V_n}\varphi\left(\frac{\mathbf{x}-\mathbf{x}_i}{h_n}\right)d\mathbf{x} \qquad \text{Set } \mathbf{x}/h_n=\mathbf{u}.$$

$$=\frac{1}{n}\sum_{i=1}^{n}\int\phi(\mathbf{u})d\mathbf{u}=\frac{1}{n}n=1$$

$$\varphi\left(\frac{\mathbf{x}-\mathbf{x}'}{h_n}\right)=\begin{cases}1 & |x_j-x_j'|\le h_n/2\\ 0 & \text{otherwise}\end{cases}$$



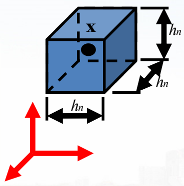
$$\boxed{p_n(\mathbf{x})=\frac{1}{n}\sum_{i=1}^{n}\frac{1}{V_n}\varphi\left(\frac{\mathbf{x}-\mathbf{x}_i}{h_n}\right)}$$

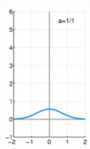
Parzen-Window Estimation

$$\boxed{\int \varphi(\mathbf{u})d\mathbf{u}=1}$$

- Parzen-Window
  - The window is ***not*** necessary a hypercube.
  - h_n is an important parameter.
    - It depends on sample size.
    - As $h_n \rightarrow 0$, $\delta_n(\mathbf{x})$ is a *Dirac delta function.*

$$\varphi\left(\frac{\mathbf{x}-\mathbf{x}'}{h_n}\right)=\begin{cases}1 & |x_j-x_j'|\le h_n/2\\ 0 & \text{otherwise}\end{cases}$$

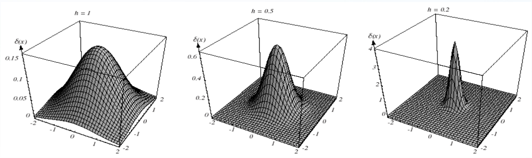




$$p_n(\mathbf{x})=\frac{1}{n}\sum_{i=1}^{n}\boxed{\frac{1}{V_n}\varphi\left(\frac{\mathbf{x}-\mathbf{x}_i}{h_n}\right)}$$

Parzen-Window Estimation

- $\delta_n(\mathbf{x})$ as a function of h_n

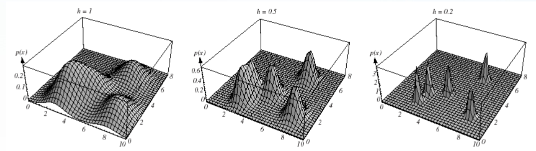


$$p_n(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} \boxed{\frac{1}{V_n}\varphi\left(\frac{\mathbf{x}-\mathbf{x}_i}{h_n}\right)}$$

TULIP Team for Universal Learning and Intelligent Processing — FLIP: Pattern Recognition (G. Li) — 25

Parzen-Window Estimation

- $p_n(\mathbf{x})$ as a function of h_n



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

TULIP Team for Universal Learning and Intelligent Processing — FLIP: Pattern Recognition (G. Li) — 26

Parzen-Window Estimation

- **Expected Value/Variance of $p_n(\mathbf{x})$**
  - The expected estimates approaches $p(\mathbf{x})$ as $V_n \to 0$

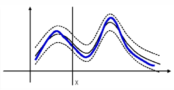
$$E[p_n(\mathbf{x})] = \int \frac{1}{V_n}\varphi\left(\frac{\mathbf{x}-\mathbf{x}'}{h_n}\right)p(\mathbf{x}')d\mathbf{x}'$$

  - The variance of the estimate is given by:

$$Var[p_n(\mathbf{x})] \le \frac{\sup(\varphi(.))E[p_n(\mathbf{x})]}{nV_n}$$

    - variance can be decreased when $nV_n \to \infty$ $(e.g., V_n = 1/\sqrt{n})$

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



At each point x, we prove the convergence of mean and the variance decreases to zero

TULIP Team for Universal Learning and Intelligent Processing — FLIP: Pattern Recognition (G. Li) — 27

Parzen-Window Estimation

- **Convergence Conditions**
  - $\lim_{n\to\infty} E[p_n(\mathbf{x})] = p(\mathbf{x})$
  - $\lim_{n\to\infty} Var[p_n(\mathbf{x})] = 0$
- We have the following additional constraints:

$$\sup_{\mathbf{u}} \varphi(\mathbf{u}) < \infty \qquad \lim_{\|\mathbf{u}\|\to\infty} \varphi(\mathbf{u})\prod_{i=1}^{d} u_i = 0$$

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

$$\lim_{n\to\infty} V_n = 0$$

$$\lim_{n\to\infty} nV_n = \infty$$

TULIP Team for Universal Learning and Intelligent Processing — FLIP: Pattern Recognition (G. Li) — 28

Parzen-Window Estimation

- **Choosing the Window Function**
  - V_n must approach 0 when $n \to \infty$, but at a rate slower than $1/n$, e.g.,

$$V_n = V_1/\sqrt{n}$$

    - The value of initial volume V_1 is important.
    - In some cases, a cell volume is proper for one region but unsuitable in a different region.

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

$$\lim_{n\to\infty} V_n = 0$$

$$\lim_{n\to\infty} nV_n = \infty$$

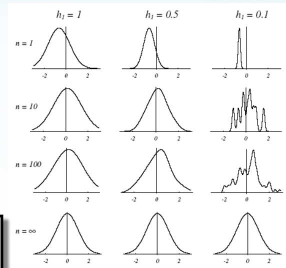
TULIP Team for Universal Learning and Intelligent Processing — FLIP: Pattern Recognition (G. Li) — 29

Parzen-Window Estimation

- 1D Illustration

$$\varphi(\mathbf{u}) = \frac{1}{\sqrt{2\pi}}e^{-u^2/2}$$

$$h_n = h_1/\sqrt{n}$$



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

TULIP Team for Universal Learning and Intelligent Processing — FLIP: Pattern Recognition (G. Li) — 30

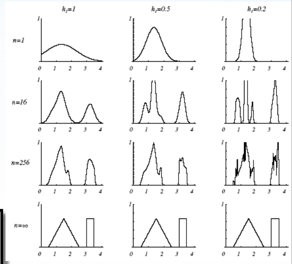
Parzen-Window Estimation

- 1D Illustration

$$\varphi(\mathbf{u}) = \frac{1}{\sqrt{2\pi}} e^{-u^2/2}$$

$$h_n = h_1 / \sqrt{n}$$

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



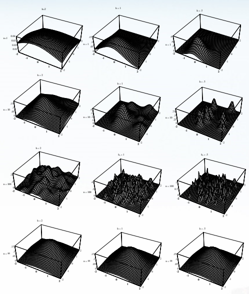
Parzen-Window Estimation

- 2D Illustration

$$\phi(\mathbf{u}) = N(u,1)$$

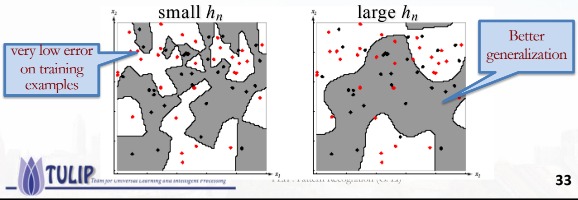
$$h_n = h_1 / \sqrt{n}$$

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



Classification using Parzen-Window Estimation

- Estimate density for each class.
- Classify a test point by computing the posterior probabilities and picking the max.
- The decision regions depend on the choice of the kernel function and h_n.



Pros and Cons of Parzen-Window Estimation

Pros

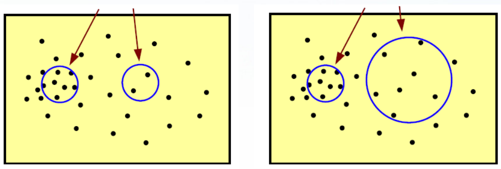
- Generality: no assumption on the form of the density
- It approximates to the true density when the samples are infinity

Cons

- Require a ***large number*** of samples to be ***stored***.
- Evaluation of the density could be very ***slow*** if the number of data points is large.
  - ***Possible solution***: use fewer kernels and adapt the positions and widths in response to the data (e.g., mixtures of Gaussians!)

Pros and Cons of Parzen-Window Estimation

- When the p(x) is not even, one window size may not fit all.



PNN (Probabilistic Neural Network)

- PNN
- Theory and Model

Probabilistic Neural Network

- The PNN is a hardware implementation of the kernel-based method of density estimation and Bayesian optimal classification (providing minimization of the average probability of the classification error)
  - Optimal Bayes' classification rule
  - (Parzen-Window) Estimation of a probability density function

$$p_n(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} \boxed{\frac{1}{V_n}\varphi\left(\frac{\mathbf{x}-\mathbf{x}_i}{h_n}\right)} \qquad p(\mathbf{x}\,|\,\omega_i)$$

Probabilistic Neural Network

$$\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}\ldots$$

$$\varphi\left(\frac{\mathbf{x}-\mathbf{x}_k}{h_n}\right) = \alpha\exp\left[\ldots -\frac{1}{2h_n^2}(\mathbf{x}^T\mathbf{x}-2\mathbf{x}^T\mathbf{x}_k-\mathbf{x}_i^T\mathbf{x}_k)^T\right]$$

Irrelevant for discriminant analysis

$$p_n(\mathbf{x}) = \frac{1}{n}\sum \ldots \mathbf{x}_k - \mathbf{x}_k^T\mathbf{x}_k)^T\Big]$$

$$= \sum_{k=1}^{n}\alpha'\exp\left[-\frac{1}{2h_n^2}(\mathbf{x}^T\mathbf{x}-2\mathbf{x}^T\mathbf{x}_k+\mathbf{x}_k^T\mathbf{x}_k)^T\right]$$

$$p_n(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} \boxed{\frac{1}{V_n}\varphi\left(\frac{\mathbf{x}-\mathbf{x}_i}{h_n}\right)} \qquad p(\mathbf{x}\,|\,\omega_i)$$

Probabilistic Neural Network

$$\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \qquad \mathbf{\Sigma} = \mathbf{I}$$

$$\varphi\left(\frac{\mathbf{x}-\mathbf{x}_k}{h_n}\right) = \alpha\exp\left[-\frac{1}{2h_n^2}(\mathbf{x}-\mathbf{x}_k)^T(\mathbf{x}-\mathbf{x}_k)\right] = \alpha\exp\left[-\frac{1}{2h_n^2}(\mathbf{x}^T\mathbf{x}-2\mathbf{x}^T\mathbf{x}_k-\mathbf{x}_i^T\mathbf{x}_k)^T\right]$$

$$p_n(\mathbf{x}) = \frac{1}{n}\sum_{k=1}^{n}\frac{1}{V_n}\alpha\exp\left[-\frac{1}{2h_n^2}(\mathbf{x}^T\mathbf{x}-2\mathbf{x}^T\mathbf{x}_k-\mathbf{x}_k^T\mathbf{x}_k)^T\right]$$

$$= \sum_{k=1}^{n}\alpha'\exp\left[-\frac{1}{2h_n^2}(\mathbf{x}^T\mathbf{x}-2\mathbf{x}^T\mathbf{x}_k+\mathbf{x}_k^T\mathbf{x}_k)^T\right] \propto \sum_{k=1}^{n}\exp\left[\frac{1}{h_n^2}(\mathbf{x}^T\mathbf{x}_k+\mathbf{x}_k^T\mathbf{x}_k/2)^T\right]$$

$$p_n(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} \boxed{\frac{1}{V_n}\varphi\left(\frac{\mathbf{x}-\mathbf{x}_i}{h_n}\right)} \qquad p(\mathbf{x}\,|\,\omega_i)$$

Probabilistic Neural Network

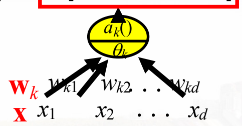
$$\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \qquad \mathbf{\Sigma} = \mathbf{I}$$

$$\varphi\left(\frac{\mathbf{x}-\mathbf{x}_k}{h_n}\right) = \alpha\exp\left[-\frac{1}{2h_n^2}(\mathbf{x}-\mathbf{x}_k)^T(\mathbf{x}-\mathbf{x}_k)\right] = \alpha\exp\left[-\frac{1}{2h_n^2}(\mathbf{x}^T\mathbf{x}-2\mathbf{x}^T\mathbf{x}_k-\mathbf{x}_i^T\mathbf{x}_k)^T\right]$$

$$p_n(\mathbf{x}) = \frac{1}{n}\sum_{k=1}^{n}\frac{1}{V_n}\alpha\exp\left[-\frac{1}{2h_n^2}(\mathbf{x}^T\mathbf{x}-2\mathbf{x}^T\mathbf{x}_k-\mathbf{x}_k^T\mathbf{x}_k)^T\right]$$

$$= \sum_{k=1}^{n}\alpha'\exp\left[-\frac{1}{2h_n^2}(\mathbf{x}^T\mathbf{x}-2\mathbf{x}^T\mathbf{x}_k+\mathbf{x}_k^T\mathbf{x}_k)^T\right] \propto \sum_{k=1}^{n}\boxed{\exp\left[\frac{1}{h_n^2}(\mathbf{x}^T\mathbf{x}_k+\mathbf{x}_k^T\mathbf{x}_k/2)^T\right]}$$

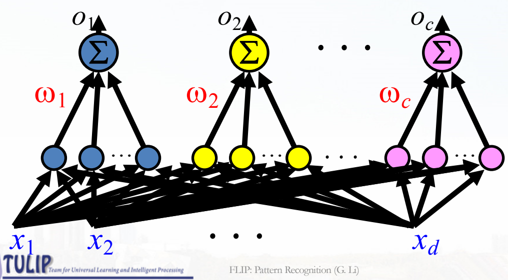
$$p_n(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} \boxed{\frac{1}{V_n}\varphi\left(\frac{\mathbf{x}-\mathbf{x}_i}{h_n}\right)}$$



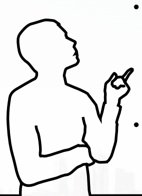
Probabilistic Neural Network

Assign patterns to the class with maximum output values.

$$p(\mathbf{x}\,|\,\omega_1) \propto o_1 \qquad p(\mathbf{x}\,|\,\omega_2) \propto o_2 \qquad p(\mathbf{x}\,|\,\omega_c) \propto o_c$$



Seminar S07



- Topics
  - Present one specific application of the Parzen-window (kernel-estimator) classifier.
  - Present/Design one method to incorporate Prior Probabilities into PNN
- Requirements
  - Prepare a **15 minutes** talk on your chosen topic
  - Make **ppt** to assist your talk
  - Prepare **at least 3 questions** to ask the audience after your talk
  - Get ready to **take questions** from the audience
- Hints
  - You can search for research articles from Google Scholar

Questions?