ICS663: Pattern Recognition

Department of Information and Computer Sciences
University of Hawai`i at Manoa

Kyungim Baek

Announcement

- Project proposal **due today by 5:00 PM**
  - Presentation (~ 10 minutes)
    - Monday (9/28):
      - BJ, Thomas, Sharif, Danny, Jeremy
    - Wednesday (9/30):
      - Tetsuya, Kelly, Nurit
    - See the evaluation criteria for presentations posted
- **Exam 1**: Wednesday, **October 7**
  - Chap. 1, Sections 2.1 ~ 2.7, 2.9, 3.1 ~ 3.5, 4.1 ~ 4.6

Density Estimation

- Parametric Density Estimation
  - Maximum-likelihood Estimation
  - Bayesian Estimation
- Nonparametric Density Estimation
  - Histograms
  - Kernel Methods
  - K-NN Methods
- Semi-parametric Density Estimation
  - Mixture Density

Lecture 8

- Nonparametric Techniques for Density Estimation
  - Histograms
  - Parzen Windows

Parametric Techniques

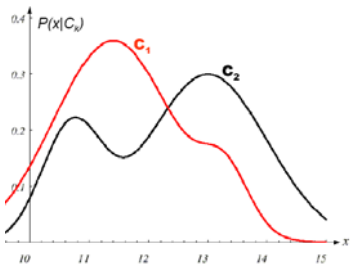
- **Parametric techniques** assume a specific functional form for the density model that can be parameterized. The parameters are then optimized by fitting the model to the data set.
- **Issues**:
  - The underlying models for probability functions are not usually known
  - In the real world, most distributions are multi-model, whereas most models are unimodal (i.e. assume that there is a single local maximum)
  - Approximating a multivariate distribution as a product of univariate distributions does not work well in practice

Nonparametric Techniques

- **Nonparametric techniques** do not assume a particular functional form for the density model, but allows the form of the density to be determined entirely by the data
  - *Histograms*: Simple approximation of the density
  - *Parzen Windows*: Estimate directly the class-conditional probabilities $p(\mathbf{x}|\omega_j)$ from the data
  - *Nearest Neighbor*: Estimate directly *a posteriori* probabilities $P(\omega_j|\mathbf{x})$ or bypass the estimation altogether by choosing directly a classification category (i.e. go directly to decision functions)

Density Estimation Problem

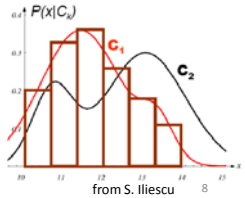
- Estimate the model of probability function $p(\mathbf{x})$ given a finite number of data points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ (while assuming that the estimation is driven entirely by the data)



from S. Iliescu

Histograms (I)

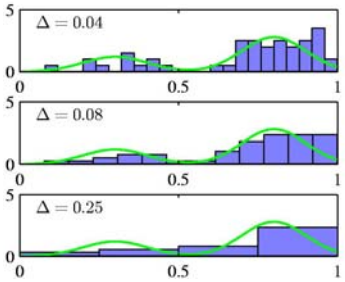
- Histograms based on the training set are the simplest methods for approximating (directly) the probability density functions. Histograms are "smoothed" by averaging over a local region of the feature space.
- **Method**: Divide the sample space into a number of bins and approximate the density at the center of each bin by the fraction of points in the training data that fall into the corresponding bin
- If *count*(**x**) is the number of samples (out of total *n*) in the same bin as **x** and $\Delta(\mathbf{x})$ is the width of the bin containing **x**, then: $P_H(\mathbf{x}) = \frac{count(\mathbf{x})}{n \cdot \Delta(\mathbf{x})}$



from S. Iliescu

Histograms (II)

An Illustration of the histogram approach to density estimation, in which a data set of 50 data points is generated from the distribution shown by the green curve. Histogram density estimates, based on the equation in the previous slide, with a common bin width Δ are shown for various values of Δ. (from *PRML* by C. Bishop)



- Issues
  - Artificial discontinuities at bin boundaries due to bin width and locations (Estimated density is not smooth)
  - Very sensitive to the size of bins (hence, the number of bins)
  - Poor generalization in higher dimensions

Mathematical Model (I)

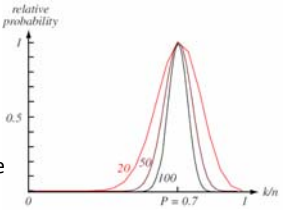
- Probability that a vector **x**, drawn from the unknown density function $p(\mathbf{x})$, will fall inside some region R is: $P = p(\mathbf{x} \in R) = \int_R p(\mathbf{x}')d\mathbf{x}'$
- If we have a set of n samples drawn independently from $p(\mathbf{x})$ then the probability that k of them will fall within the region R is:

# of unique splits k vs. $n-k$ → $P_k = \binom{n}{k} P^k (1-P)^{n-k}$ ← prob. of k samples falling in R; prob. that the rest are not

and the expected value for k is: $E(k) = nP$

- ML estimation of $P = \theta$, $\max_{\theta}(P_k \mid \theta)$ is reached for $\hat{\theta} = \frac{k}{n} \cong P$

  Therefore, the ratio k/n is a good estimate for the probability P

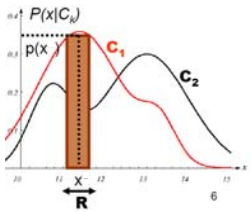


Mathematical Model (II)

- **Approximations** for the integral
  - If $p(\mathbf{x})$ is continuous and does not vary significantly within region R, then P can be approximated with the product between the (average value of) density function $p(\mathbf{x})$ within the region and the area/volume of the region

$$P = \int_R p(\mathbf{x}')d\mathbf{x}' \approx p(\mathbf{x})V \approx \frac{k}{n}$$

- **Density estimate**: $p(\mathbf{x}) \approx \frac{k}{nV}$
- **Observation**: estimate becomes more accurate as the number of sample points n increases while the region's volume V shrinks



partly from S. Iliescu

Density Estimation

- **General expression** for nonparametric density estimation is:

$$p(\mathbf{x}) \approx \frac{k}{nV}$$

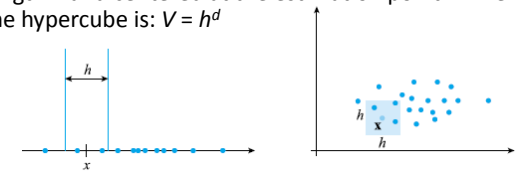
  - n = total number of samples
  - V = volume of the region R surrounding **x**
  - k = number of samples inside R (of volume V)

- **Two approaches**
  - Kernel-based methods or **Parzen Windows**: choose a fixed value for V and determine k from training data
  - **k-Nearest Neighbor** (**kNN**): choose a fixed value of k and determine the corresponding volume V from training data

from S. Iliescu

Parzen-Window Estimation (I)

- Assume that the region is a *d*-dimensional hypercube with sides of length *h* and centered at the estimation point **x**. The volume of the hypercube is: $V = h^d$



- The number *k* of samples falling within the hypercube will be found by defining the kernel function $\varphi(\mathbf{u})$ as follows:

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

unit hypercube centered at the origin

- This window function is also known as **Parzen window** or **naïve estimator**

Parzen-Window Estimation (II)

- The number of sample points within the hypercube with side of length *h* centered at the estimation point **x**:

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

- The density estimate is:

$K(\mathbf{x}, \mathbf{x}_i)$ Integrates to 1

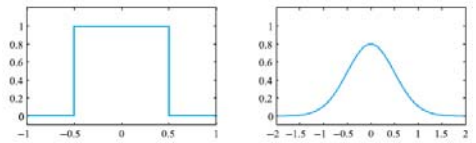
$$p_n(\mathbf{x}) = \frac{k_n}{nV_n} = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{V_n} \varphi\left(\frac{\mathbf{x} - \mathbf{x}_i}{h_n}\right)$$

- $p_n(\mathbf{x})$ estimates $p(\mathbf{x})$ as an average of functions of **x** and the samples $\mathbf{x}_i$ ($i = 1, \ldots, n$)

Parzen-Window Estimation (III)

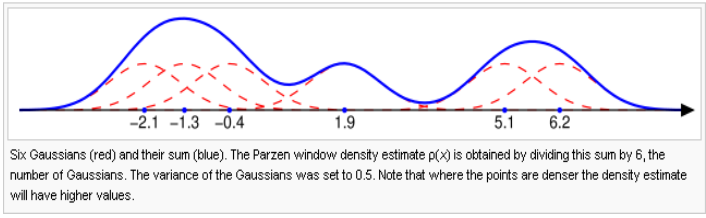
- The problem with hypercube window function is as in histograms – it is discontinuous
  - A smoother function can be used
- The window functions φ can be general satisfying the conditions $\varphi(\mathbf{x}) \ge 0$ and $\int \varphi(\mathbf{x}) d\mathbf{x} = 1$ (i.e. φ itself is a density function)
  - These conditions ensure the estimated $p_n(\mathbf{x})$ is a legitimate density function (i.e. it is non-negative and integrates to 1)

Examples of window functions



Parzen-Window Estimation (IV)

- Density estimate: $p(x) = \frac{k}{nV} = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{V} \varphi\left(\frac{x - x_i}{h}\right)$
- Consider a Gaussian window function: $\varphi(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}}$



Six Gaussians (red) and their sum (blue). The Parzen window density estimate p(x) is obtained by dividing this sum by 6, the number of Gaussians. The variance of the Gaussians was set to 0.5. Note that where the points are denser the density estimate will have higher values.

Figure is from http://en.wikipedia.org

Effect of Window Width (I)

- Parzen-window density estimation is an interpolation based on a windowing function $\varphi(\cdot)$
- By defining a function $\delta(\mathbf{x})$ as follows, the density function $p(\mathbf{x})$ can be rewritten:

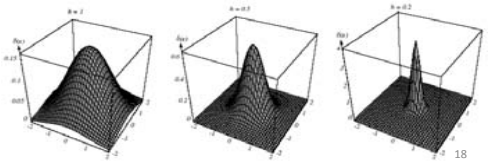
$$\delta(\mathbf{x}) = \frac{1}{V}\varphi\left(\frac{\mathbf{x}}{h}\right) \longrightarrow p(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\delta(\mathbf{x}-\mathbf{x}_i)$$

- $p(\mathbf{x})$ is the sum obtained by placing the center of $\delta(\mathbf{x})$ at data points
- h affects both the amplitude and width of $\delta(\mathbf{x})$

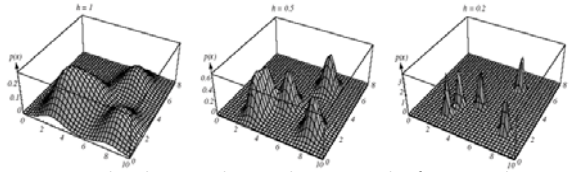
Effect of Window Width (II)

- If h is very large, then $\mathbf{x}$ should be far apart from $\mathbf{x}_i$ before $\delta(\mathbf{x}-\mathbf{x}_i)$ changes significantly from $\delta(\mathbf{0})$
  - $p(\mathbf{x})$ appears as a superposition of n broad and slowly changing functions
  - $p(\mathbf{x})$ is a very smooth "out-of-focus" estimate
- If h is very small, then the peak value of $\delta(\mathbf{x}-\mathbf{x}_i)$ is quite large, and it occurs in the vicinity $\mathbf{x} = \mathbf{x}_i$
  - $p(\mathbf{x})$ appears as a superposition of n sharp pulse centered at the samples
- If h approaches 0, then $\delta(\mathbf{x}-\mathbf{x}_i)$ approaches a *Dirac delta function* centered at $\mathbf{x}_i$

Examples of two-dimensional circularly symmetric normal Parzen windows for three different values of h

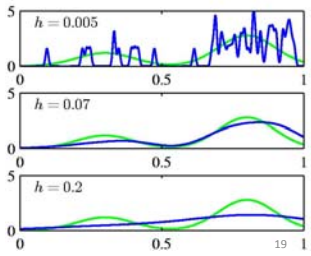


Effect of Window Width (III)



Three density estimates based on 5 samples using the Parzen window functions in the previous slide

Right: Illustration of the kernel density model applied to the same data set used to demonstrate the histogram approach earlier. We see that h acts as a smoothing parameter and that if it is set too large (bottom panel), then the binomial nature of the underlying distribution from which the data is generated (shown by the green curve) is washed out. The best density model is obtained for some intermediate value of h (middle panel). (from *PRML* by C. Bishop)



An Illustration

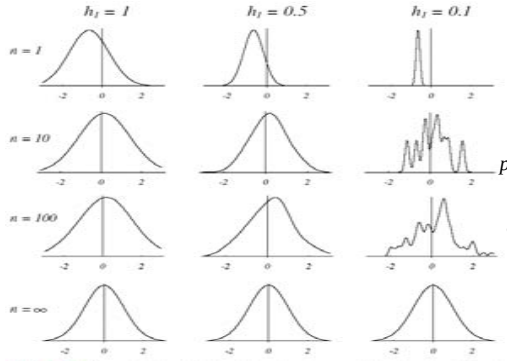


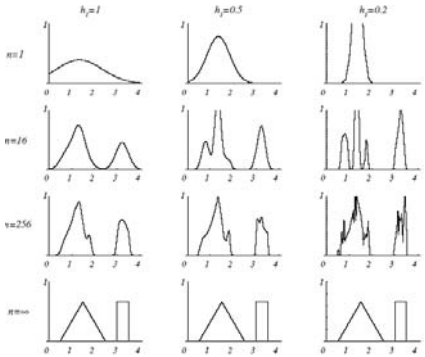
FIGURE 4.5. Parzen-window estimates of a univariate normal density using different window widths and numbers of samples. The vertical axes have been scaled to best show the structure in each graph. Note particularly that the $n = \infty$ estimates are the same (and match the true density function), regardless of window width. From: Richard

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

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

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

Another Example



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

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

Parzen-window estimates of a bimodal distribution (a mixture of a uniform and a triangle density) using different window widths and numbers of samples. Note that the $n = \infty$ estimates are the same (and match the true distribution), regardless of window width.

Classification Strategy

- **Strategy**: estimate the likelihood probabilities $p(\mathbf{x}|C_k)$ for each class by employing the Parzen-window method and use Bayes rule to classify the test data (i.e. compute the posterior probabilities and decide in favor of the class yielding the largest posterior probability)
- **Advantages**: generality
  - Method does not assume any prior knowledge about the underlying distribution
  - The same method can be applied to estimate different types of distribution (unimodal normal, bimodal mixture, etc.)

from S. Iliescu

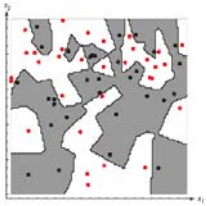
Classification Strategy (*cont'd*)

- **Disadvantages**: needs lots of data to ensure that estimates converge to the true distributions
  - Severe requirements for computation time and storage
  - Demand for large number of samples grows exponentially with the dimensionality of the feature space ("curse of dimensionality")
- **Danger of overfitting**: training error can be arbitrarily small (or even zero) by choosing small enough windows. This is not desirable since it will most likely cause overfitting and deteriorate the test performance.

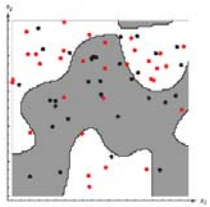
from S. Iliescu

Choice of Window Width

- The decision boundaries depend on the window width *h*. Apparently, for the data shown below, a small *h* would be appropriate for the upper region, while a large *h* would fit better the lower region; no single window width is ideal overall.



Small window width leads to more complicated boundaries and too much dissection of the feature space.



Large window width leads to higher training error, but most likely to have better generalization performance.