Pattern Recognition

Non-Parametric Methods for Estimating Probability Density Functions

Classification Problem (Review)

- The classification problem is to assign an arbitrary feature vector $x \in F$ to one of c classes.

- The classifier is a function from the feature space onto the set of classes,
  $\alpha : F \rightarrow \{ \omega_1 , \ldots , \omega_c \}$.   ($\alpha(x)$ is the classifier)

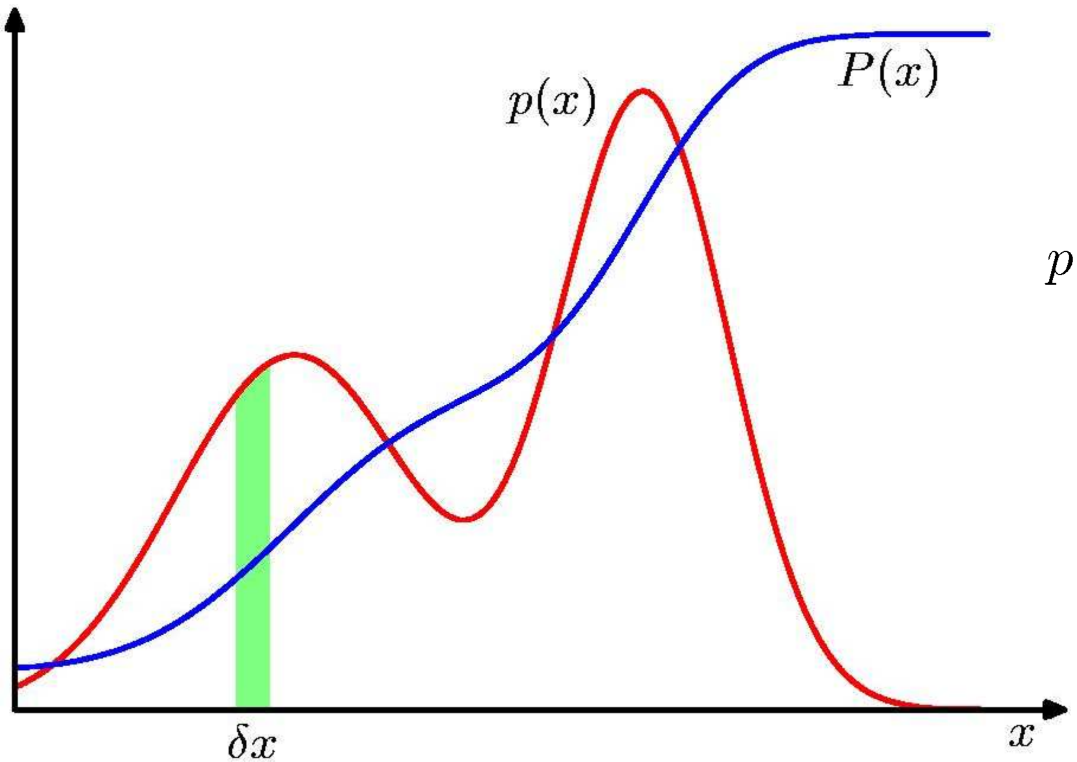
Classification Problem

- Feature vectors x that we aim to classify belong to the feature space *F*.
- The task is to assign an arbitrary feature vector $x \in F$ to one of the *c* classes
- We know the
  - 1. prior probabilities $P(\omega_1), \ldots, P(\omega_c)$ of the classes and
  - 2. the class conditional probability density functions $p(x \mid \omega_1), \ldots, p(x \mid \omega_c)$.

Probability Density Function

- A **probability density function** (**pdf**), or **density** of a continuous random variable, is a function that describes the relative likelihood for this random variable to take on a given value.

Probability Densities



$$p(x \in (a, b)) = \int_a^b p(x)\,\mathrm{d}x$$

$$P(z) = \int_{-\infty}^{z} p(x)\,\mathrm{d}x$$

$$p(x) \geqslant 0 \qquad \int_{-\infty}^{\infty} p(x)\,\mathrm{d}x = 1$$

Estimating Class Conditional PDFs and a-Priori Probabilities

- In practice, PDFs and a-priori probabilities are not known.
- PDFs and a-priori probabilities are estimated from training samples. (Supervised learning)
- We assume that the training samples are occurrences of the independent random variables. (That is: they were measured from different objects.)
- These random variables are assumed to be distributed according to $p(x|\omega_i)$. (independent and identically distributed (i.i.d.)).

Training Data Types

- Mixture Sampling: A set of objects are randomly selected, their feature vectors are computed and then the objects are hand-classified to the most appropriate classes.

- Separate Sampling: The training data for each class is collected separately.

Estimating a-Priori Probabilities

- For the classifier training, the difference of the two sampling techniques is:

  based on the mixed sampling we can deduce the a-priori probabilities $P(\omega_1), \ldots, P(\omega_c)$ as:

$$P(\omega_i) = \frac{n_i}{\sum_{j=1}^{c} n_j}.$$

Parametric Estimation of PDFs

- If we assume that $p(x|\omega_i)$ belongs to some family of parametric distributions, the class conditional pdfs $p(x|\omega_i)$ is reduced to the estimation of the parameter vector θ_i
- For example, we can assume that $p(x|\omega_i)$ is a normal density with unknown parameters $\theta_i = (\mu_i , \Sigma_i)$.

Maximum Likelihood Estimation

- The aim is to estimate the value of the parameter vector θ based on the training samples $D = \{ x_1 , . . . , x_n \}$.
- We assume that the training samples are occurrences of i.i.d. random variables distributed according to the density $p(x|\theta)$.

Maximum Likelihood Estimation

- The maximum likelihood estimate or the ML estimate θ’ maximizes the probability of the training data with respect to θ. Due to the i.i.d. assumption, the probability of D is

$$p(\mathcal{D}|\theta) = \prod_{i=1}^{n} p(\mathbf{x}_i|\theta).$$

Non-Parametric Estimation of Density Functions

- Often assuming class conditional pdfs to be members of a certain parametric family is not reasonable.
- Instead, we must estimate the class conditional pdfs non-parametrically.
- In non-parametric estimation (or density estimation), we try to estimate $p(x|\omega_i)$ in each point x whereas in parametric estimation we tried to estimate some unknown parameter vector.

Density Estimation Problem

- We are given the training data D = { x_1 , . . . , x_n }, where the samples are i.i.d. , and are all drawn from the unknown density p(x).

- The aim is to find an estimate p^(x) for p(x) in every point x.

Histogram

- Histograms are the simplest approach to density estimation.
- The feature space is divided into m equal sized cells or bins B_i .
- Then, the number of the training samples n_i , i = 1, . . . n falling into each cell is computed.

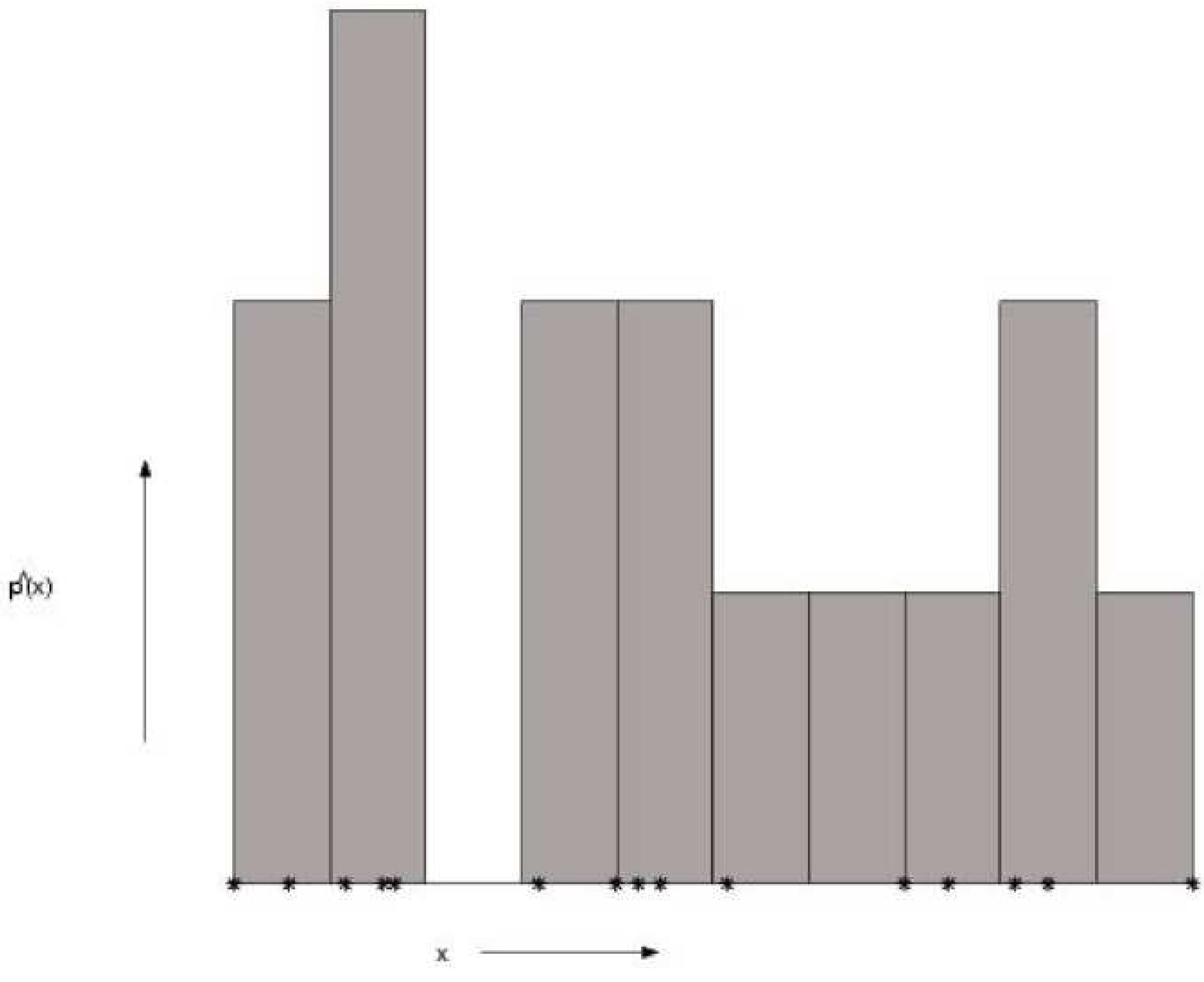
Histogram

- The density estimate is

$$\hat{p}(\mathbf{x}) = \frac{n_i}{Vn}, \quad \text{when} \quad \mathbf{x} \in \mathcal{B}_\mathrm{i}$$

V is the volume of the cell

Histogram



General Formulation of Density Estimation

- We are interested in how to select a suitable cell size/shape at the proximity of x, to produce an accurate pˆ(x).
- To estimate pˆ(x) at the point x and using the set (neighborhood) B surrounding x, the probability that a certain training sample x_j is in B is

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

General Formulation of Density Estimation

- We need: Probability that k out of n training samples fall into the set B

- Assuming: The training samples are independent, and each of them is in the set B with the probability P

General Formulation of Density Estimation

- The probability that there are exactly k samples in the set B is:

$$P_k = \binom{n}{k} P^k (1-P)^{n-k},$$

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

General Formulation of Density Estimation

- The expected value of k is:

$$E[k] = \sum_{k=0}^{n} k P_k = \sum_{k=1}^{n} \left(\begin{array}{c} n-1 \\ k-1 \end{array} \right) P^{k-1} (1-P)^{n-k} nP = nP.$$

General Formulation of Density Estimation

- Replacing E[k] with kˆ, an estimate for P is:

$$\hat{P} = \hat{k}/n.$$

General Formulation of Density Estimation

- If p(x) is continuous and B is small enough

$$\int_{\mathcal{B}} p(\mathbf{x}) d\mathbf{x} \simeq p(\mathbf{x}) V,$$

where V is the volume of B

Conclusion

- The obtained density estimate is a space averaged version of the true density.
- The smaller the volume V the more accurate the estimate is.
- However, if n is fixed, diminishing V will lead sooner or later to B which does not contain any training samples and the density estimate will become useless.
- The principal question is how to select B and V

Parzan Window

- Assume: region B_n is a d-dimensional hypercube. If h_n is the length of the side of the hypercube, its volume is given by $V_n = h_n^d$.

- Define a function that returns value 1 inside the hypercube centered at the origin, and value 0 outside the hypercube:

Parzan Window

$$\varphi(\mathbf{u}) = \begin{cases} 1 & \text{if} \quad |\mathrm{u_j}| \leq 1/2 \\ 0 & \quad \text{otherwise} \end{cases}$$

- If x_i is inside the hypercube: $\varphi((\mathbf{x} - \mathbf{x}_i)/h_n) = 1$
- If x_i is outside the hypercube: $\varphi((\mathbf{x} - \mathbf{x}_i)/h_n) = 0$

Parzan Window

- The density estimate becomes:

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

Parzen Estimates

- The Parzen-window density estimate at x using n training samples and the window function ϕ is defined by:

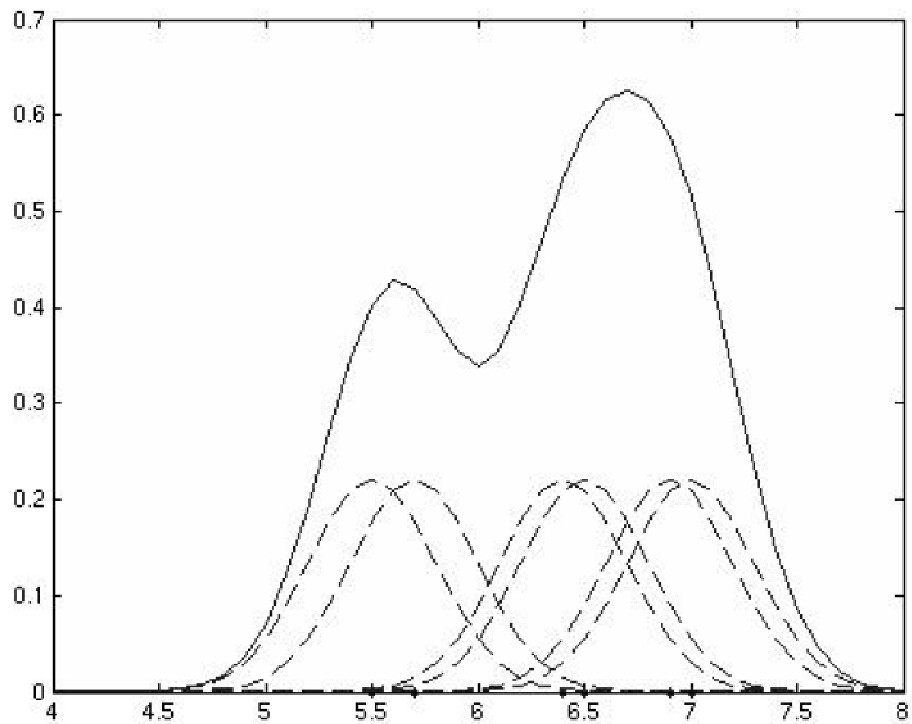
$$p_n(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{V_n} \varphi\left(\frac{\mathbf{x} - \mathbf{x}_i}{h_n}\right).$$

- Each training sample is contributing to the estimate in accordance with its distance from x

Parzen Estimates

- Using the normal density as window function:

$$\varphi(\mathbf{u}) \;=\; \frac{1}{(2\pi)^{d/2}} \exp[-0.5\mathbf{u}^T\mathbf{u}]$$



Disadvantage

- Every classification with Parzen classifiers requires n evaluations of a pdf, where n is the total number of training samples.

k-Nearest Neighbors Classifier

- The design of Parzen classification involves selecting window functions and suitable window widths.
- One possibility is to let them depend on the training data.
- This means fixing k_n and computing of suitable (small enough) V_n based on the selected k_n.

k-Nearest Neighbors Classifier

- To estimate p(x):
  - Place the center of the cell B_n at the test point x and let the cell grow until it encircles k_n training samples.
  - These k_n training samples are k_n nearest neighbors of x. Here, k_n is a given parameter.

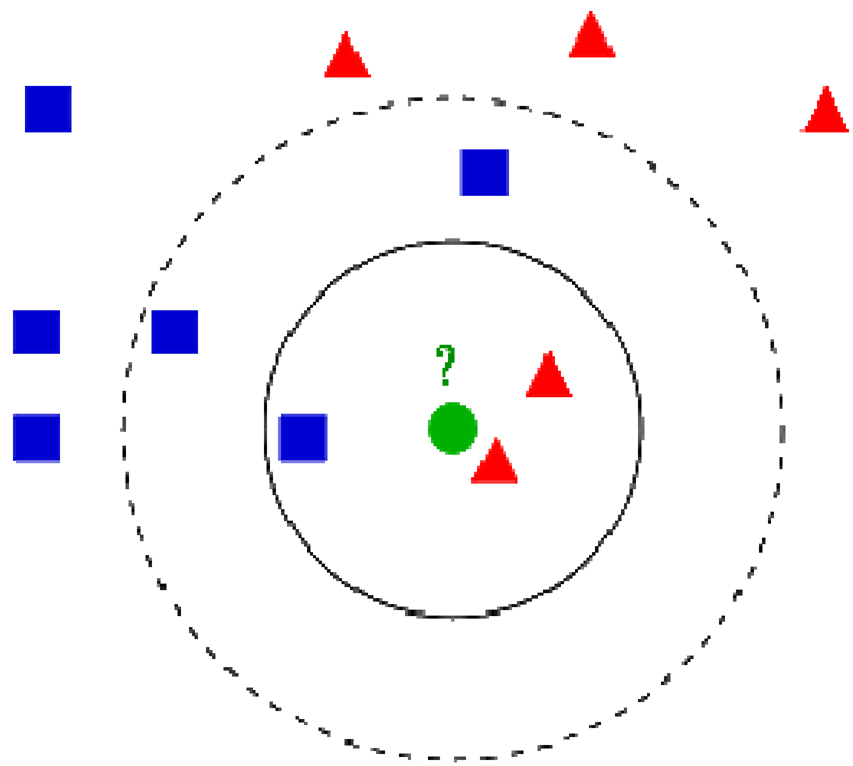
k-Nearest Neighbors Classifier

- The k_n nearest neighbor (KNN) density estimate is given by:

$$p_n(\mathbf{x}) = \frac{k_n}{nV_n},$$

- V_n is the volume of the smallest possible x centered cell that contains k_n training samples, and n is the total number of training samples

K-Nearest Neighbor Example



Disadvantages

- The distance to all sample points should be computed at each classification. This computation can be very time consuming
- The accuracy of the *k*-NN algorithm can be severely degraded by the presence of noisy or irrelevant features.

Questions?