COSC 522 – Machine Learning

Bayes Decision Theory – In-depth Discussion (Parametric vs. Non-Parametric)

Hairong Qi, Gonzalez Family Professor
Electrical Engineering and Computer Science
University of Tennessee, Knoxville
https://www.eecs.utk.edu/people/hairong-qi/
Email: hqi@utk.edu

Bayes' Formula (Bayes' Rule)

AICIP RESEARCH

conditional probability density function (pdf) or "likelihood"

from training data

prior probability (*a-priori* probability)

from domain knowledge

$$P(w_j|x) = \frac{p(x|w_j)\,P(w_j)}{p(x)}$$

j index for different classes
w_j: different classes
x: training sample

posterior probability (*a-posteriori* probability)

$$p(x) = \sum_{j=1}^{c} p(x|w_j)\,P(w_j)$$

normalization constant (evidence)

AICIP RESEARCH

Part I

In-Depth Discussion on Parametric Learning: Discriminant Functions (Three Cases with m-d Gaussian pdf)

Bayes Decision Theory

AICIP RESEARCH

$$P(\omega_j \mid x) = \frac{p(x \mid \omega_j) P(\omega_j)}{p(x)}$$

Maximum Posterior Probability

For a given x, if $P(\omega_1 \mid x) > P(\omega_2 \mid x)$, then x belongs to class 1, otherwise, 2.

Discriminant Function

The classifier will assign a feature vector x to class ω_i if $g_i(x) > g_j(x)$

Parametric Learning with Gaussian pdf

Case 1: Minimum Euclidean Distance (Linear Machine), $\Sigma_i = \sigma^2 I$

Case 2: Minimum Mahalanobis Distance (Linear Machine), $\Sigma_i = \Sigma$

Case 3: Quadratic classifier , Σ_i = arbitrary

All assuming Gaussian pdf

Multivariate Normal Density

AICIP RESEARCH

$$p(\vec{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\vec{x}-\vec{\mu})^T \Sigma^{-1}(\vec{x}-\vec{\mu})\right]$$

$\vec{x}$: d - component column vector

$\vec{\mu}$: d - component mean vector

Σ : d - by - d covariance matrix

$|\Sigma|$: determinant

Σ^{-1} : inverse

$$\vec{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix}, \vec{\mu} = \begin{bmatrix} \mu_1 \\ \vdots \\ \mu_d \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} \sigma_{11} & \cdots & \sigma_{1d} \\ \vdots & \ddots & \vdots \\ \sigma_{d1} & \cdots & \sigma_{dd} \end{bmatrix} = \begin{bmatrix} \sigma_1{}^2 & \cdots & \sigma_{1d} \\ \vdots & \ddots & \vdots \\ \sigma_{d1} & \cdots & \sigma_d{}^2 \end{bmatrix}$$

When $d = 1$, $p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}\right]$

Estimating Normal Densities

AICIP RESEARCH

- Calculate μ, Σ

$$\vec{\mu}_i = \begin{bmatrix} \mu_{i1} = \dfrac{1}{n_i}\sum_{k=1}^{n_i} x_{k1} \\ \vdots \\ \mu_{id} = \dfrac{1}{n_i}\sum_{k=1}^{n_i} x_{kd} \end{bmatrix}$$

$$\Sigma_i = \begin{bmatrix} \sigma_{11} & \cdots & \sigma_{1d} \\ \vdots & \ddots & \vdots \\ \sigma_{d1} & \cdots & \sigma_{dd} \end{bmatrix} = \frac{1}{n_i - 1}\sum_{k=1}^{n_i} \left(\vec{x}_k - \vec{\mu}_i\right)\left(\vec{x}_k - \vec{\mu}_i\right)^T$$

Covariance

For d sets of variates denoted $\{x_1\}, \cdots, \{x_p\}, \cdots, \{x_q\}, \cdots, \{x_d\}$, the covariance $\sigma_{pq} = \mathrm{cov}(x_p, x_q)$ of x_p and x_q is defined by

$$\begin{aligned}
\mathrm{cov}(x_p, x_q) = E[(x_p - \mu_p)(x_q - \mu_q)] \\
&= E[x_p x_q] - E[x_p \mu_q] - E[\mu_p x_q] + E[\mu_p \mu_q] \\
&= E[x_p x_q] - \mu_q E[x_p] - \mu_p E[x_q] + \mu_p \mu_q \\
&= E[x_p x_q] - \mu_q \mu_p - \mu_p \mu_q + \mu_p \mu_q \\
&= E[x_p x_q] - \mu_q \mu_p
\end{aligned}$$

When $p = q, \sigma_{pp} = \mathrm{cov}(x_p, x_p) = E[x_p x_p] - \mu_p \mu_p$

$$\begin{aligned}
&= E[x_p{}^2] - (E[x_p])^2 \\
&= \sigma_p^2
\end{aligned}$$

Discrimimant Function

$$P\left(w_j | x \right) = \frac{p\left(x | w_j \right) P\left(w_j \right)}{p(x)}$$

- One way to represent pattern classifier - use discriminant functions $g_i(x)$

$$g_i(x) = P(\omega_i | x)$$

$$g_i(x) = p(x|\omega_i)P(\omega_i)$$

$$g_i(x) = \ln p(x|\omega_i) + \ln P(\omega_i)$$

The classifier will assign a feature vector x to class ω_i if

$$g_i(x) > g_j(x)$$

- For two-class cases,

$$g(x) = g_1(x) - g_2(x) = P(\omega_1 | x) - P(\omega_2 | x)$$

Discriminant Function for Normal Density

$$p(\vec{x} \mid w) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\vec{x}-\vec{\mu})^T \Sigma^{-1} (\vec{x}-\vec{\mu})\right]$$

$$\begin{aligned}
g_i(\vec{x}) &= \ln p(\vec{x} \mid \omega_i) + \ln P(\omega_i) \\
&= -\frac{1}{2}(\vec{x}-\vec{\mu}_i)^T \Sigma_i^{-1} (\vec{x}-\vec{\mu}_i) - \frac{d}{2}\ln(2\pi) - \frac{1}{2}\ln|\Sigma_i| + \ln P(\omega_i) \\
&= -\frac{1}{2}(\vec{x}-\vec{\mu}_i)^T \Sigma_i^{-1} (\vec{x}-\vec{\mu}_i) - \frac{1}{2}\ln|\Sigma_i| + \ln P(\omega_i)
\end{aligned}$$

Case 1: $\Sigma_i = \sigma^2 I$

- The features are statistically independent, and have the same variance
- Geometrically, the samples fall in equal-size hyperspherical clusters
- Decision boundary: hyperplane of d-1 dimension

$$\Sigma = \begin{bmatrix} \sigma^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma^2 \end{bmatrix}, |\Sigma| = \sigma^{2d}, \Sigma^{-1} = \begin{bmatrix} \frac{1}{\sigma^2} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{1}{\sigma^2} \end{bmatrix}$$

Linear Discriminant Function and Linear Machine

AICIP RESEARCH

$\left\| \vec{x} - \vec{\mu}_i \right\|$: the Euclidean norm (distance)

$$\left\| \vec{x} - \vec{\mu}_i \right\|^2 = (\vec{x} - \vec{\mu}_i)^T (\vec{x} - \vec{\mu}_i)$$

$$g_i(\vec{x}) = -\frac{\left\| \vec{x} - \vec{\mu}_i \right\|^2}{2\sigma^2} + \ln P(\omega_i)$$

$$= -\frac{\vec{x}^T \vec{x} - 2\vec{\mu}_i^{\,T} \vec{x} + \vec{\mu}_i^{\,T} \vec{\mu}_i}{2\sigma^2} + \ln P(\omega_i)$$

$$\boxed{g_i(\vec{x}) = \frac{\vec{\mu}_i^{\,T}}{\sigma^2} \vec{x} - \frac{\vec{\mu}_i^T \vec{\mu}_i}{2\sigma^2} + \ln P(\omega_i)}$$

Minimum-Distance Classifier

AICIP RESEARCH

- When $P(\omega_i)$ are the same for all c classes, the discriminant function is actually measuring the minimum distance from each x to each of the c mean vectors

$$g_i(\vec{x}) = -\frac{\left\|\vec{x} - \vec{\mu}_i\right\|^2}{2\sigma^2}$$

Case 2: $\Sigma_i = \Sigma$

AICIP RESEARCH

- The covariance matrices for all the classes are identical but not a scalar of identity matrix.
- Geometrically, the samples fall in hyperellipsoidal
- Decision boundary: hyperplane of d-1 dimension

$$g_i(\vec{x}) = \ln p(\vec{x} \mid \omega_i) + \ln P(\omega_i)$$

$$= -\frac{1}{2}\underbrace{\boxed{(\vec{x}-\vec{\mu}_i)^T \Sigma_i^{-1} (\vec{x}-\vec{\mu}_i)}}_{\text{Squared Mahalanobis distance}} + \ln P(\omega_i)$$

$$= \vec{\mu}_i^T \left(\Sigma^{-1}\right)^T \vec{x} - \frac{1}{2}\vec{\mu}_i^T \Sigma^{-1} \vec{\mu}_i + \ln P(\omega_i)$$

Case 3: Σ_i = arbitrary

- The covariance matrices are different from each category
- Quadratic classifier
- Decision boundary: hyperquadratic for 2-D Gaussian

$$g_i(\vec{x}) = \ln p(\vec{x} \mid \omega_i) + \ln P(\omega_i)$$

$$= -\frac{1}{2}(\vec{x} - \vec{\mu}_i)^T \Sigma_i^{-1} (\vec{x} - \vec{\mu}_i) - \frac{1}{2}\ln|\Sigma_i| + \ln P(\omega_i)$$

$$= -\frac{1}{2}\vec{x}^T \Sigma_i^{-1} \vec{x} + \vec{\mu}_i^T (\Sigma_i^{-1})^T \vec{x} - \frac{1}{2}\vec{\mu}_i^T \Sigma_i^{-1} \vec{\mu}_i - \frac{1}{2}\ln|\Sigma_i| + \ln P(\omega_i)$$

Questions

- What is a discriminant function?
- What is a multivariate Gaussian (or normal density function)?
- What is the covariance matrix and what is its dimension?
- What would the covariance matrix look like if the features are independent from each other?
- What would the covariance matrix look like if the features are independent from each other AND have the same spread in each dimension?
- What is minimum (Euclidean) distance classifier? Is it a linear or quadratic classifier (machine)? What does the decision boundary look like?
- What are the assumptions made when using a minimum (Euclidean) distance classifier?
- What is minimum (Mahalanobis) distance classifier? Is it a linear or quadratic classifier (machine)? What does the decision boundary look like?
- What are the assumptions made when using a minimum (Mahalanobis) distance classifier?
- What does the decision boundary look like for a quadratic classifier?
- What are the cost functions for the discriminant functions? And what is the optimization method used to find the best solution?

AICIP RESEARCH

Part II

In-Depth Discussion on Non-Parametric Learning: Why kNN?

kNN in Classification

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

- Given c training sets from c classes, the total number of samples is

$$n = \sum_{m=1}^{c} n_m$$

- Given a point **x** at which we wish to determine the statistics, we find the hypersphere of volume ***V*** which just encloses k points from the combined set. If within that volume, k_m of those points belong to class m, then we estimate the density for class m by

$$p(x|w_m) = \frac{k_m/n_m}{V} \qquad P(w_m) = \frac{n_m}{n} \qquad p(x) = \frac{k/n}{V}$$

$$P(\omega_m \mid x) = \frac{p(x \mid \omega_m)P(\omega_m)}{p(x)} = \frac{\dfrac{k_m}{n_m V}\dfrac{n_m}{n}}{\dfrac{k}{nV}} = \frac{k_m}{k}$$

kNN Decision Boundary

AICIP RESEARCH

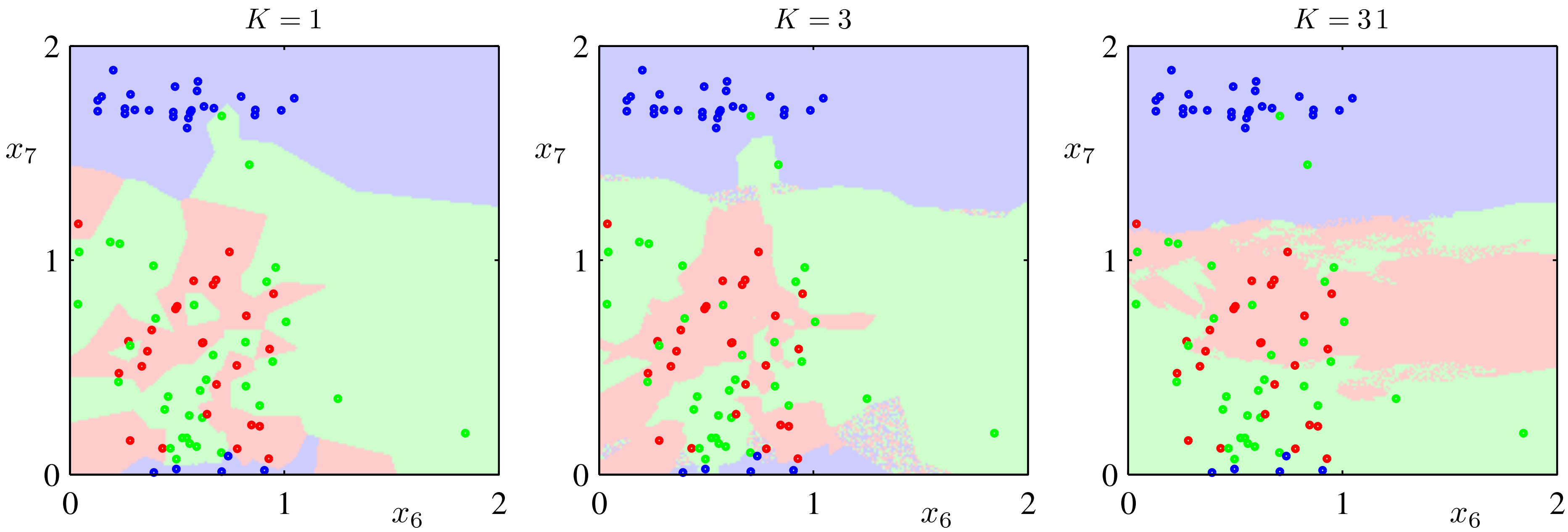


Figure 2.28 Plot of 200 data points from the oil data set showing values of x_6 plotted against x_7, where the red, green, and blue points correspond to the 'laminar', 'annular', and 'homogeneous' classes, respectively. Also shown are the classifications of the input space given by the K-nearest-neighbour algorithm for various values of K.

From [Bishop 2006]

Parzen Windows

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

AICIP RESEARCH

- The density estimation at *x* is calculated by counting the number of samples fall within a hypercube of volume *V* centered at *x*
- Let *R* be a *d*-dimensional hypercube, whose edges are *h* units long. Its volume is then $V=h^d$
- Introducing the “window” function

$$\varphi(u) = \begin{cases} 1 & |u_j| \le 0.5 \quad j=1,\ldots,d \\ 0 & otherwise \end{cases}$$

- Calculate k_n

$$k_n = \sum_{i=1}^{n} \varphi\left(\frac{x-x_i}{h}\right)$$

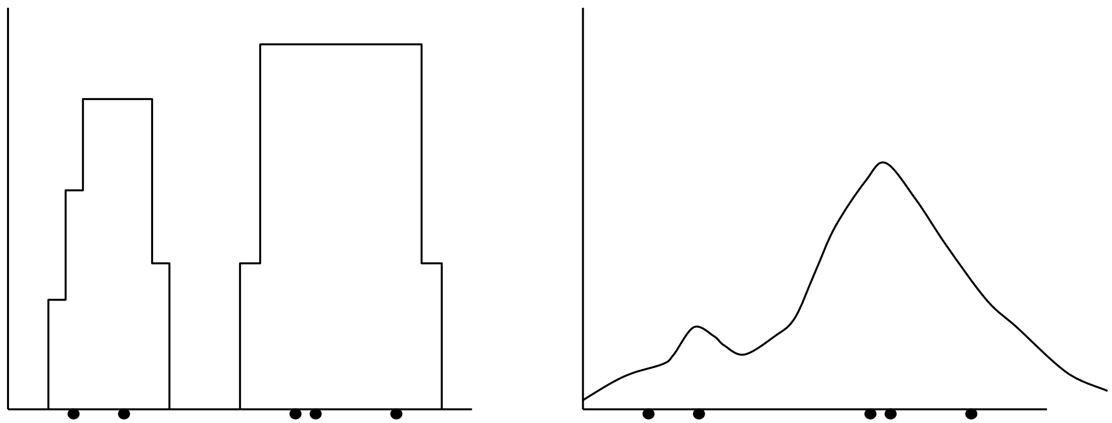
- Hence

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

Problems of Parzen Windows

AICIP RESEARCH

- Hypercube – why should a point just inside the hypercube contribute the same as a point very near to **x**, while a point just outside the hypercube contributes nothing? – Use a continuous window function



- How to choose *h*? – Depend on the number of samples. $h = \frac{1}{\sqrt{n}}$
- … but the hypercube is of fixed volume!
- How does kNN solve the “fixed” volume problem?

Potential Issues of kNN

AICIP RESEARCH

- What is a good value of "k"? $k_n = \sqrt{n}$
- What kind of distance should be used to measure "nearest"
  - Euclidean metric is a reasonable measurement
- Computation burden
  - Massive storage burden
  - Need to compute the distance from the unknown to all the neighbors