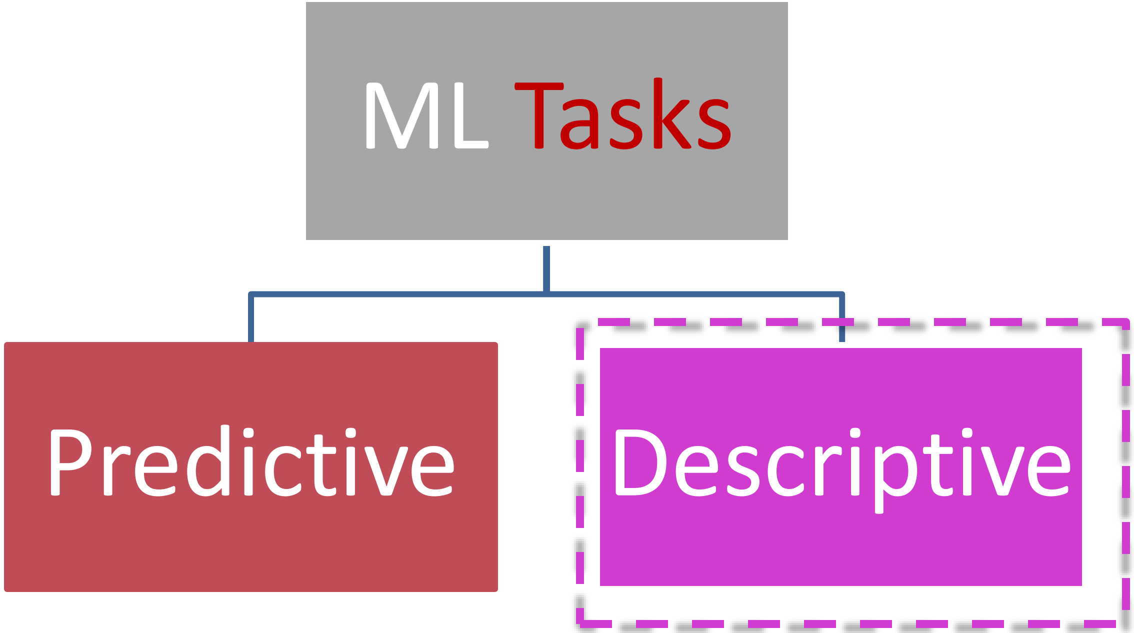
19.03.2019

Statistical Methods in AI (CSE/ECE 471)

Lecture-18: Kernel Density Estimation

Ravi Kiran

Center for Visual Information Technology (CVIT), IIIT Hyderabad



ML Tasks

Predictive

Descriptive

ML::Tasks → Descriptive

- Study/Exploit the ‘structure’ of data
  - Density Estimation
  - Clustering
  - Dimensionality Reduction
- Also studied as ‘Unsupervised Learning’
  - ‘Input’ data without paired ‘Output’

Unsupervised Learning → Density Estimation

Aka "learning without a teacher"

Feature Space $\mathcal{X}$

Words in a document ⇨ Word distribution (Probability of a word)

Task: Given $X \in \mathcal{X}$, learn $f(X)$.

$$\{x^{(1)}, \ldots, x^{(m)}\} \qquad x^{(i)} \in \mathbb{R}^n$$

The k-means clustering algorithm is as follows:

1. Initialize **cluster centroids** $\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$ randomly.

2. Repeat until convergence: {

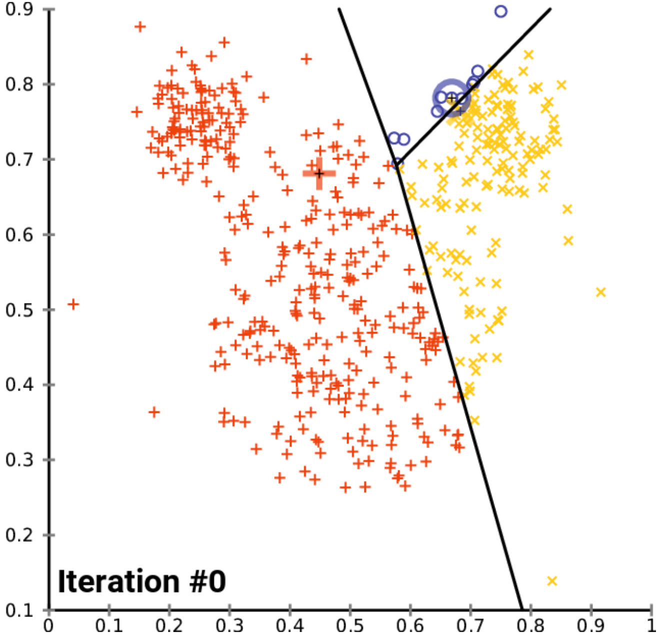
   For every i, set

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2.$$

   For each j, set

$$\mu_j := \frac{\sum_{i=1}^m 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^m 1\{c^{(i)} = j\}}.$$

}



GMM - Algorithm

- Initialize the means μ_k, covariances Σ_k and mixing coefficients π_k
- Iterate until convergence:
  - E-step: Evaluate the responsibilities given current parameters

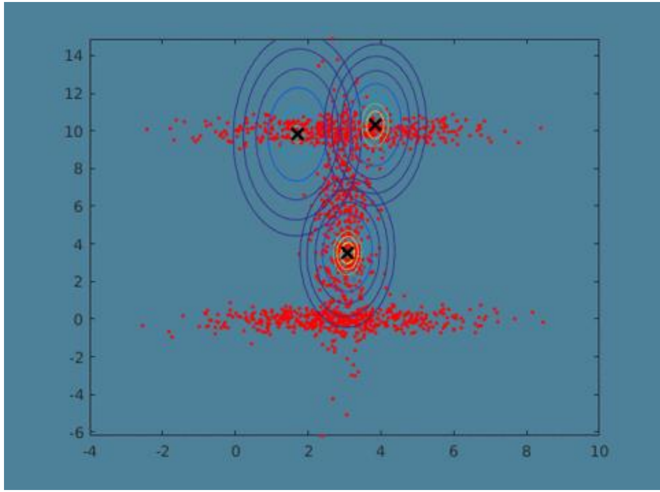
$$\gamma_k^{(n)} = p(z^{(n)}|\mathbf{x}) = \frac{\pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}^{(n)}|\mu_j, \Sigma_j)}$$

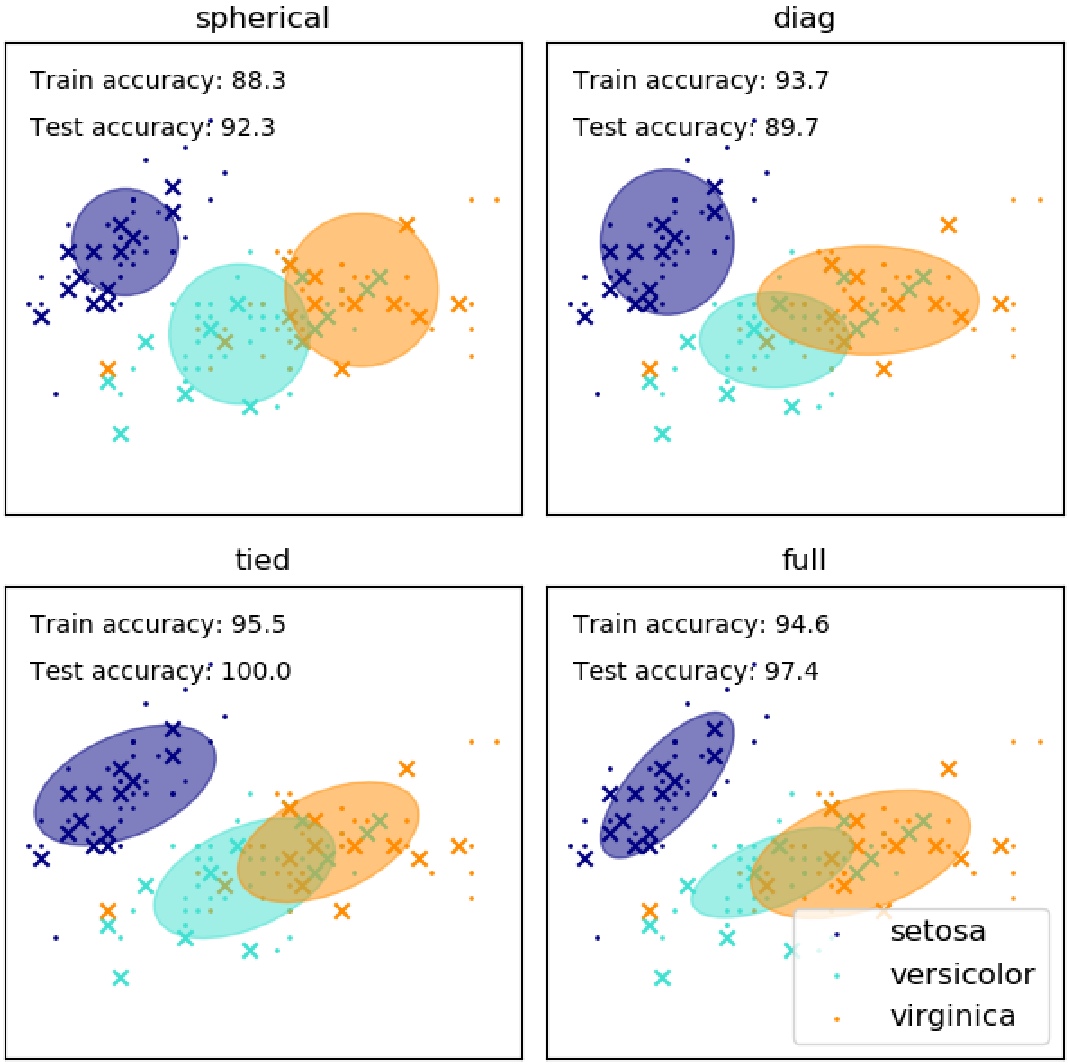
  - M-step: Re-estimate the parameters given current responsibilities

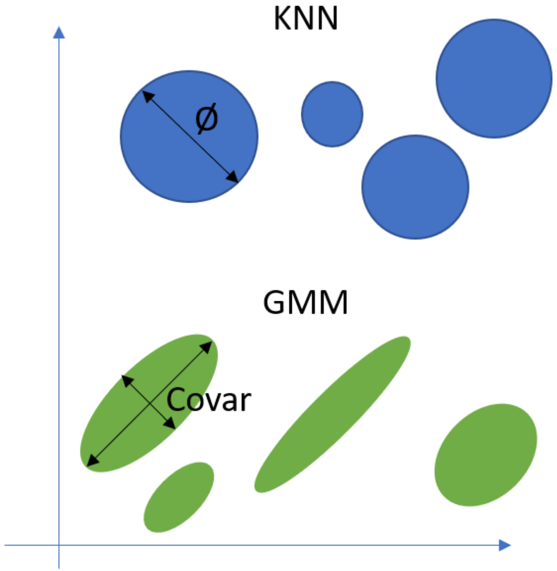
$$\begin{aligned} \mu_k &= \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k^{(n)} \mathbf{x}^{(n)} \\ \Sigma_k &= \frac{1}{N_k} \sum_{n=1}^{N} \gamma_k^{(n)} (\mathbf{x}^{(n)} - \mu_k)(\mathbf{x}^{(n)} - \mu_k)^T \\ \pi_k &= \frac{N_k}{N} \quad \text{with} \quad N_k = \sum_{n=1}^{N} \gamma_k^{(n)} \end{aligned}$$

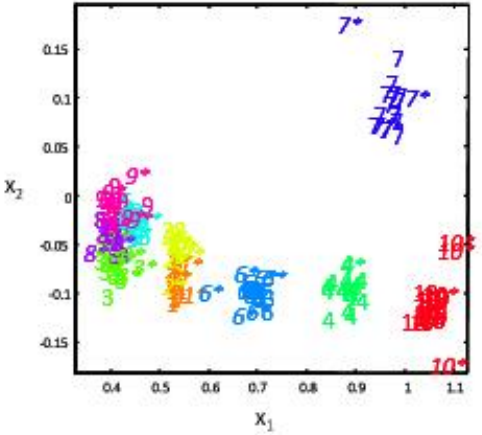
  - Evaluate log likelihood and check for convergence

$$\ln p(\mathbf{X}|\pi, \mu, \Sigma) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}^{(n)}|\mu_k, \Sigma_k) \right)$$

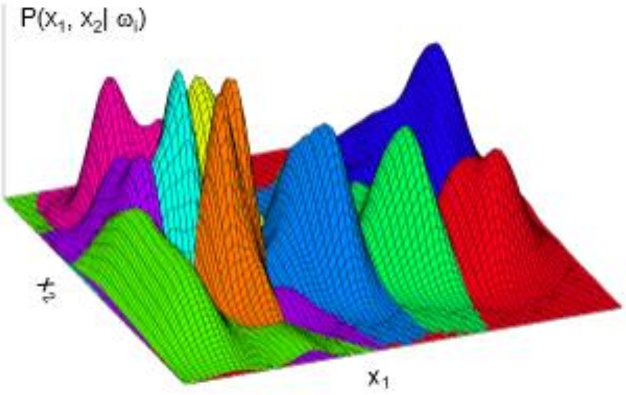



spherical
Train accuracy: 88.3
Test accuracy: 92.3
diag
Train accuracy: 93.7
Test accuracy: 89.7
tied
Train accuracy: 95.5
Test accuracy: 100.0
full
Train accuracy: 94.6
Test accuracy: 97.4
setosa
versicolor
virginica

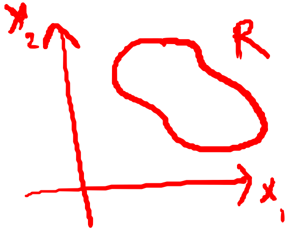

KNN
ø
GMM
Covar



non-parametric density estimation

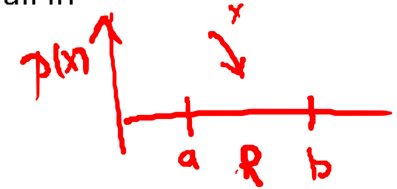


$\underline{p(x \in R)}$



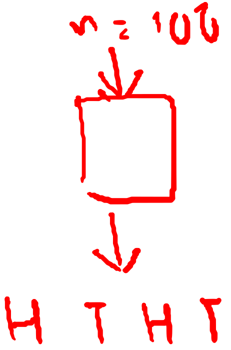
- The probability that a vector x, drawn from a distribution $p(x)$, will fall in a given region $\Re$ of the sample space is

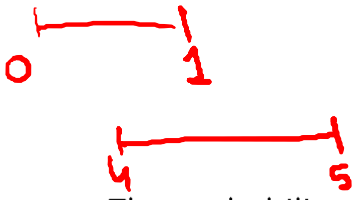
$$\underline{P = \int_{\Re} p(x')dx'}$$



$x \in R$

$P(a < x < b)$

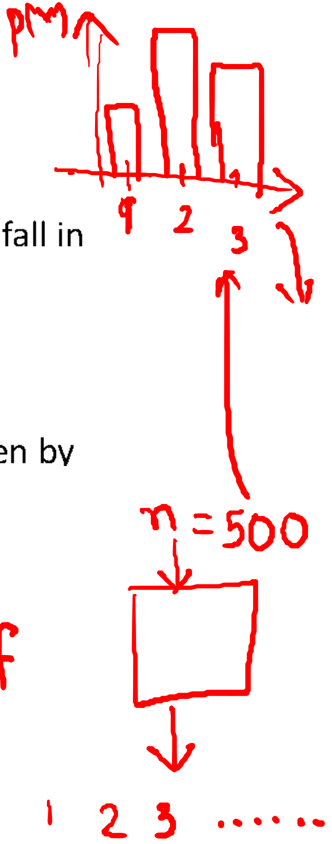




$$\int_{-\infty}^{a} p(x')dx'$$

p(X)

CDF : $P(X \leq a)$
$= P(X=1)+P(X=2)$

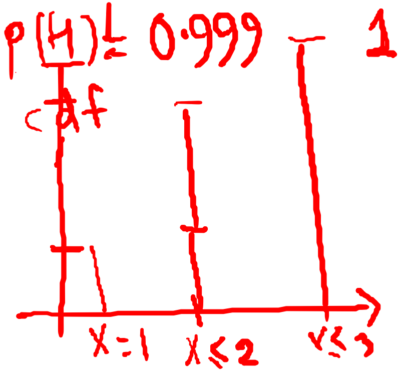


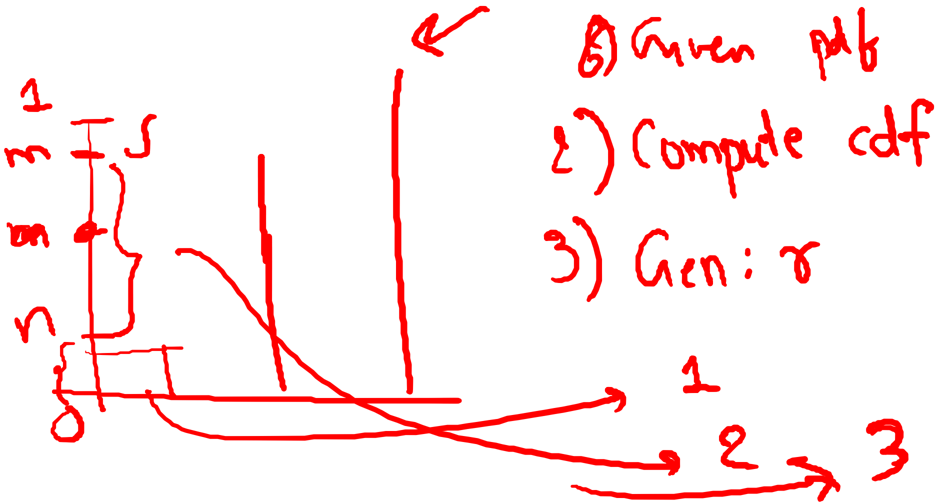
– The probability that a vector x, drawn from a distribution $p(x)$, will fall in a given region $\Re$ of the sample space is

$$P = \int_{\Re} p(x')dx'$$

– Suppose now that N vectors $\{x^{(1}, x^{(2}, \ldots x^{(N}\}$ are drawn from the distribution; the probability that k of these N vectors fall in $\Re$ is given by

$r < 0.5 \rightarrow H$
else → T





1) Given pdf
2) Compute cdf
3) Gen : r

- The probability that a vector x, drawn from a distribution $p(x)$, will fall in a given region $\mathfrak{R}$ of the sample space is

$$P = \int_{\mathfrak{R}} p(x')dx'$$

- Suppose now that N vectors $\left\{x^{(1}, x^{(2}, \dots x^{(N}\right\}$ are drawn from the distribution; the probability that k of these N vectors fall in $\mathfrak{R}$ is given by the binomial distribution

$$P(k) = \binom{N}{k} P^k (1 - P)^{N-k}$$

- The probability that a vector x, drawn from a distribution $p(x)$, will fall in a given region $\Re$ of the sample space is

$$P = \int_{\Re} p(x')dx'$$

- Suppose now that N vectors $\{x^{(1}, x^{(2}, \dots x^{(N}\}$ are drawn from the distribution; the probability that k of these N vectors fall in $\Re$ is given by the binomial distribution

$$P(k) = \binom{N}{k} P^k (1-P)^{N-k}$$

- It can be shown (from the properties of the binomial p.m.f.) that the mean and variance of the ratio k/N are

$$E\left[\frac{k}{N}\right] = P \quad \text{and} \quad var\left[\frac{k}{\mathrm{N}}\right] = E\left[\left(\frac{k}{N} - P\right)^2\right] = \frac{P(1-P)}{N}$$

- The probability that a vector x, drawn from a distribution $p(x)$, will fall in a given region $\Re$ of the sample space is

$$P = \int_{\Re} p(x')dx'$$

- Suppose now that N vectors $\{x^{(1}, x^{(2}, \dots x^{(N}\}$ are drawn from the distribution; the probability that k of these N vectors fall in $\Re$ is given by the binomial distribution

$$P(k) = \binom{N}{k} P^k (1-P)^{N-k}$$

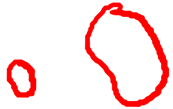
- It can be shown (from the properties of the binomial p.m.f.) that the mean and variance of the ratio k/N are

$$E\left[\frac{k}{N}\right] = P \quad \text{and} \quad var\left[\frac{k}{\mathrm{N}}\right] = E\left[\left(\frac{k}{N} - P\right)^2\right] = \frac{P(1-P)}{N}$$

- Therefore, as $N \to \infty$ the distribution becomes sharper (the variance gets smaller), so we can expect that a good estimate of the probability P can be obtained from the mean fraction of the points that fall within $\Re$

$$P \cong \frac{k}{N}$$

[Bishop, 1995]



- On the other hand, if we assume that $\Re$ is so small that $p(x)$ does not vary appreciably within it, then

$$\int_{\Re} p(x')dx' \cong p(x)V$$

$$\int_{R} 1 \, dx' = V$$

  - where V is the volume enclosed by region $\Re$

- On the other hand, if we assume that $\Re$ is so small that $p(x)$ does not vary appreciably within it, then

$$\int_{\Re} p(x')dx' \cong p(x)V$$

  - where V is the volume enclosed by region $\Re$

- Merging with the previous result we obtain

$$\left.\begin{array}{l} P = \int_{\Re} p(x')dx' \cong p(x)V \\ P \cong \frac{k}{N} \end{array}\right\} \Rightarrow p(x) \cong \frac{k}{NV}$$

The histogram

The simplest form of non-parametric DE is the histogram

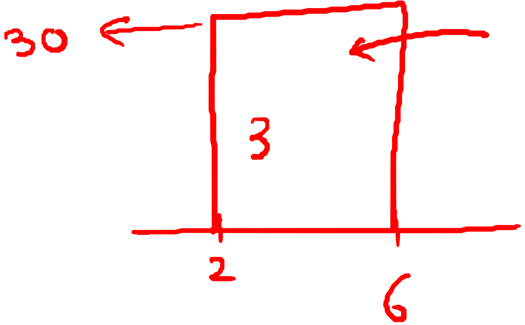
- Divide the sample space into a number of bins and approximate the density at the center of each bin by the fraction of points in the training data that fall into the corresponding bin

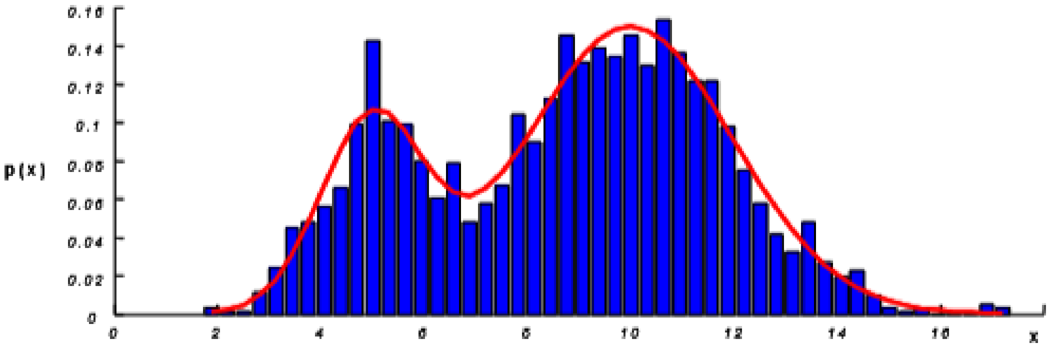
$$p(x) \cong \frac{k}{NV}$$

$$p_H(x) = \frac{1}{N}\frac{[\#\ of\ x^{(k}\ in\ same\ bin\ as\ x]}{[width\ of\ bin]}$$

$$p_H(x=3) = \frac{30}{100\ 4}$$

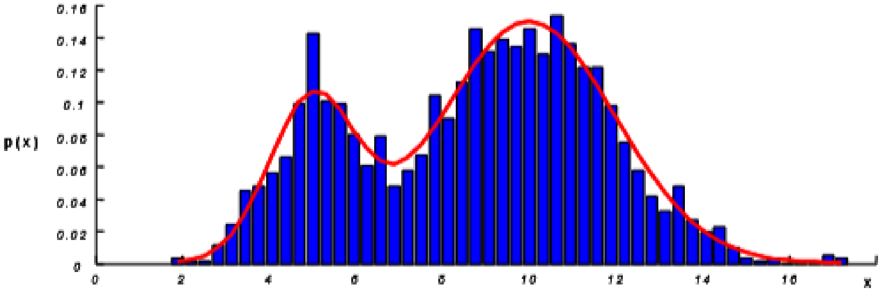
- The histogram requires two “parameters” to be defined: bin width and starting position of the first bin

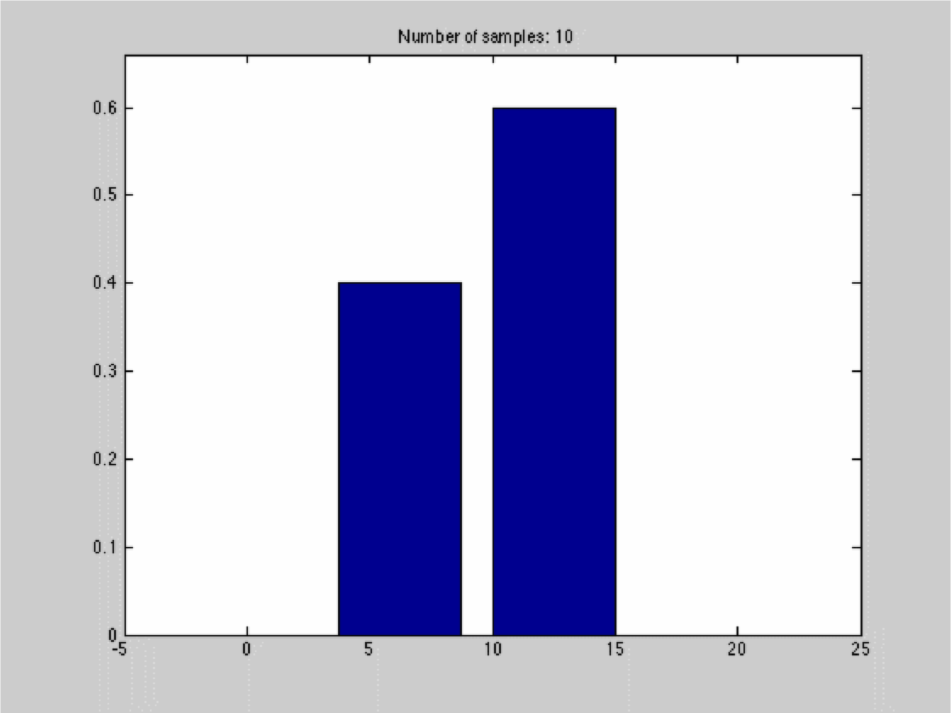




$$p_H(x) = \frac{1}{N}\frac{[\# \ of \ x^{(k} \ in \ same \ bin \ as \ x]}{[width \ of \ bin]}$$

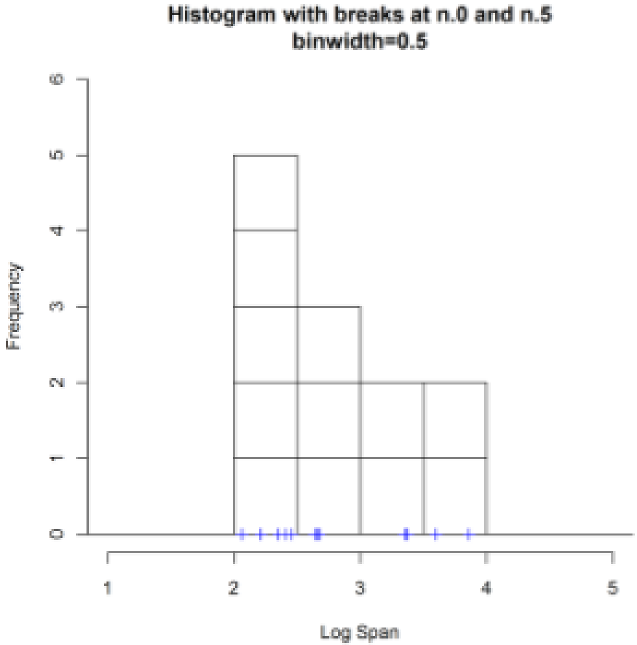
$$P_{dx} \approx f(x)dx = \frac{N_{dx}}{N}$$





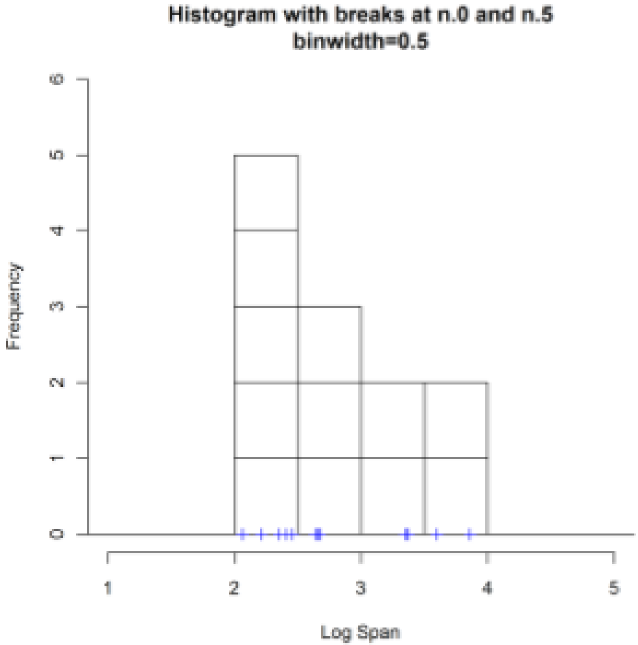
A toy example

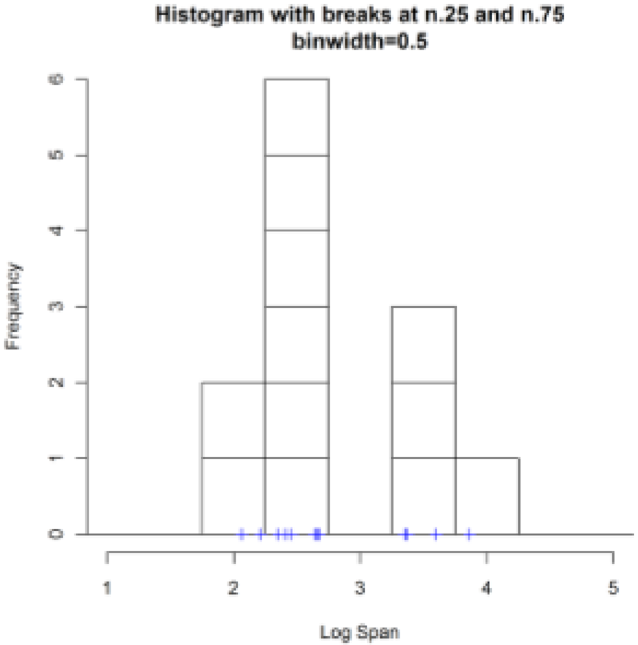
- (the log of) wing spans of aircraft built from 1956 – 1984
- Wing-spans: 2, 22, 42, 62, 82, 102, 122, 142, 162, 182, 202, 222



A toy example

- (the log of) wing spans of aircraft built from 1956 – 1984
- Wing-spans: 2, 22, 42, 62, 82, 102, 122, 142, 162, 182, 202, 222

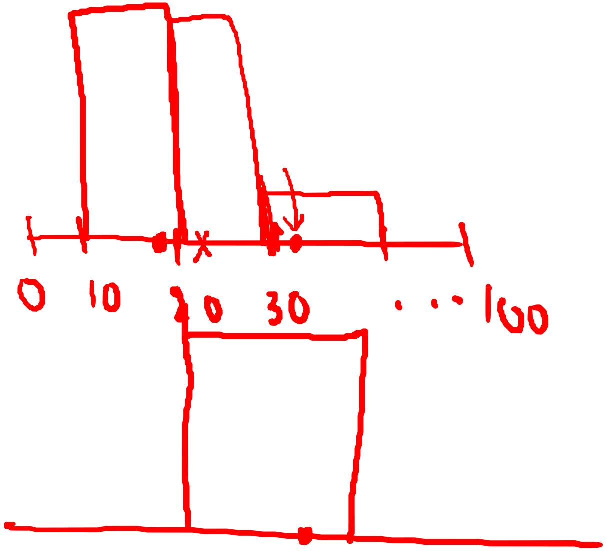


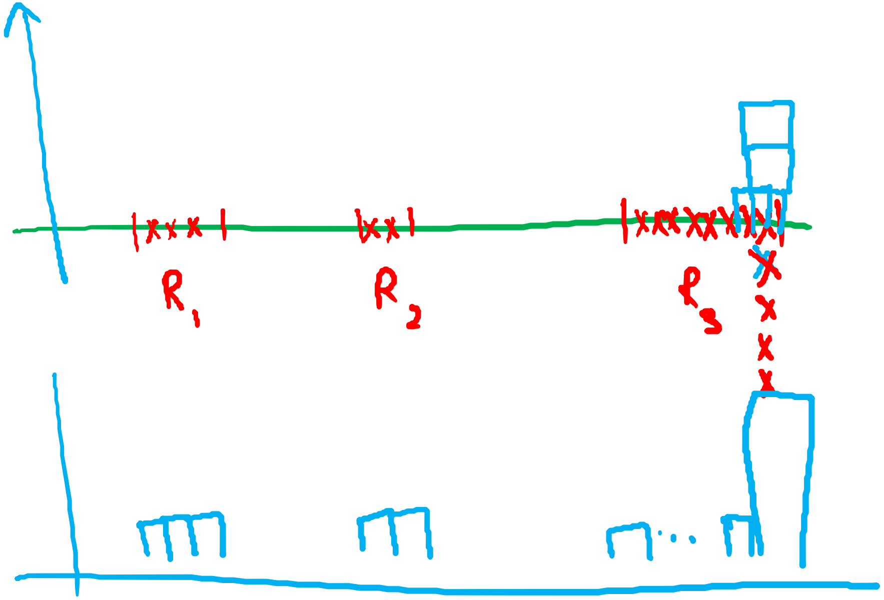


Issues with Histograms

- Not smooth
- Depend on end points of bins
- Depend on width of bins

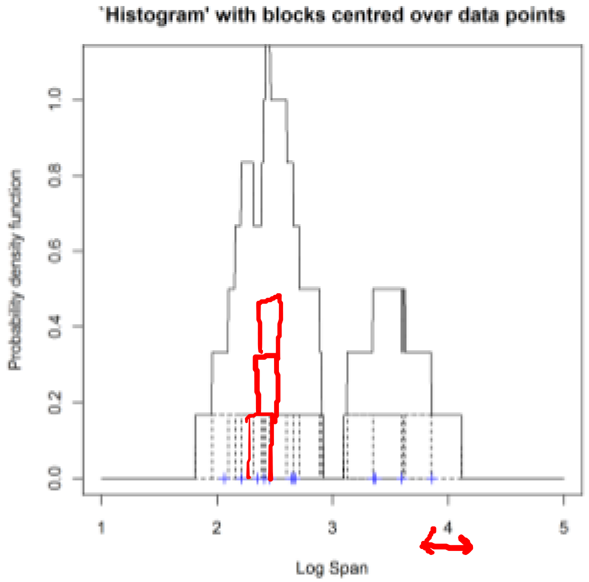
Characterizing density – a data-driven perspective





Box Kernel Density Estimate

- 12 data-points
- block width ½, height 1/6



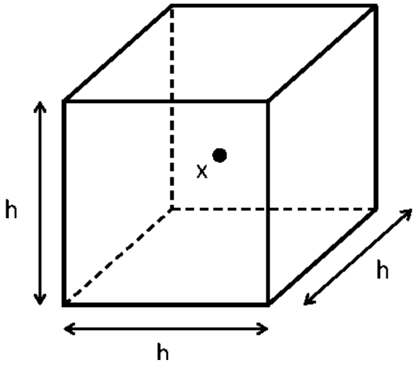
- In conclusion, the general expression for non-parametric density estimation becomes

$$p(x) \cong \frac{k}{NV} \text{ where } \begin{cases} V & \textit{volume surrounding } x \\ N & \textit{total \#examples} \\ k & \textit{\#examples inside } V \end{cases}$$

Parzen windows

Problem formulation

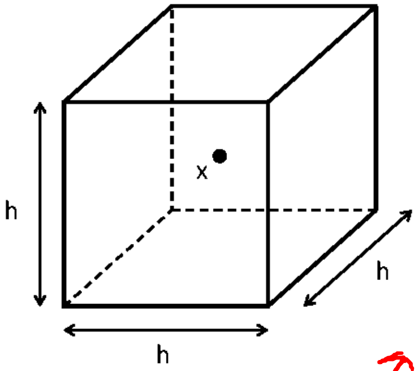
- Assume that the region $\Re$ that encloses the k examples is a hypercube with sides of length h centered at x
  - Then its volume is given by $V = h^D$, where D is the number of dimensions



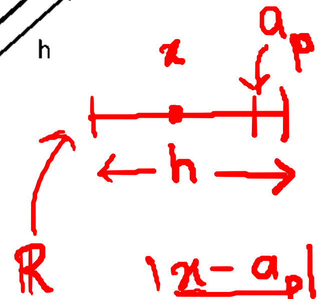
Parzen windows

Problem formulation

- Assume that the region $\Re$ that encloses the k examples is a hypercube with sides of length h centered at x
  - Then its volume is given by $V = h^D$, where D is the number of dimensions



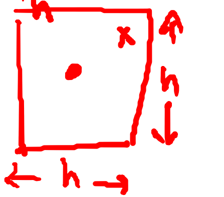
$a_1, a_2 \ldots a_n$



- To find the number of examples that fall within this region we define a kernel function $K(u)$

$$K(u) = \begin{cases} 1 & |u_j| < 1/2 \quad \forall j = 1 \ldots D \\ 0 & otherwise \end{cases}$$

$\left|\frac{x - a_p}{h}\right| < \frac{1}{2}$



  - This kernel, which corresponds to a unit hypercube centered at the origin, is known as a Parzen window or the naïve estimator
  - The quantity $K((x - x^{(n}))/h)$ is then equal to unity if $x^{(n}$ is inside a hypercube of side h centered on x, and zero otherwise

[Bishop, 1995]

$$K\left(\frac{x-x^{(n)}}{}\right) \quad K\left(\frac{x-x^{(n)}}{h}\right)$$

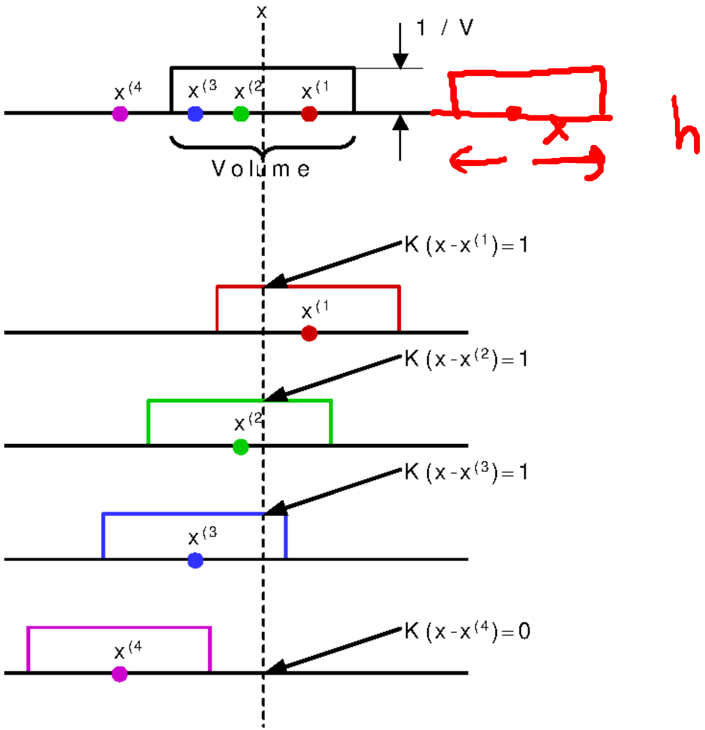
- The total number of points inside the hypercube is then

$$k = \sum_{n=1}^{N} K\left(\frac{x-x^{(n)}}{h}\right)$$

Substituting back into the expression for the density estimate

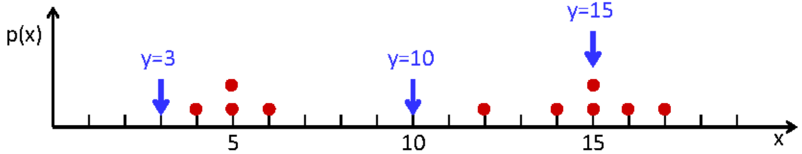
$$p_{KDE}(x) = \frac{1}{Nh^D}\sum_{n=1}^{N} K\left(\frac{x-x^{(n)}}{h}\right)$$

- Notice how the Parzen window estimate resembles the histogram, with the exception that the bin locations are determined by the data



Exercise

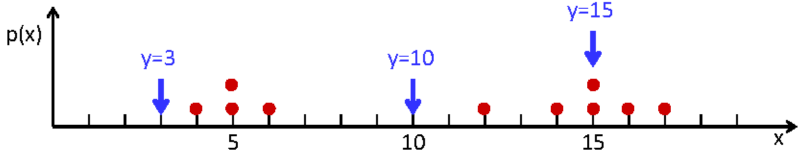
- Given dataset $X = \{4, 5, 5, 6, 12, 14, 15, 15, 16, 17\}$, use Parzen windows to estimate the density $p(x)$ at $y = 3{,}10{,}15$; use $h = 4$
- Solution
  - Let's first draw the dataset to get an idea of the data



  - Let's now estimate $p(y = 3)$

Exercise

- Given dataset $X = \{4, 5, 5, 6, 12, 14, 15, 15, 16, 17\}$, use Parzen windows to estimate the density $p(x)$ at $y = 3{,}10{,}15$; use $h = 4$
- Solution
  - Let's first draw the dataset to get an idea of the data

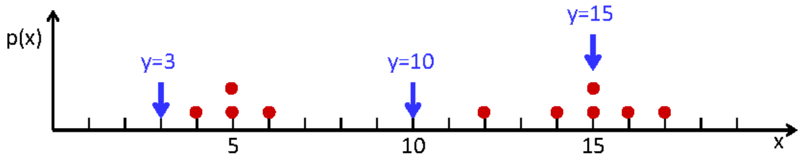


  - Let's now estimate $p(y = 3)$

$$p(y=3) = \frac{1}{Nh^D} \sum_{n=1}^{N} K\left(\frac{x - x^{(n}}{h}\right)$$

Exercise

- Given dataset $X = \{4, 5, 5, 6, 12, 14, 15, 15, 16, 17\}$, use Parzen windows to estimate the density $p(x)$ at $y = 3{,}10{,}15$; use $h = 4$
- Solution
  - Let’s first draw the dataset to get an idea of the data

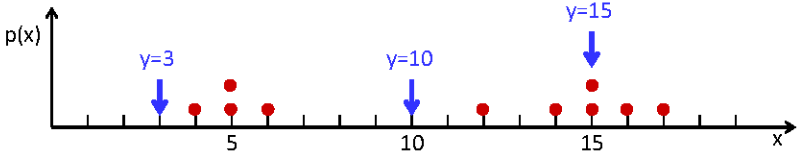


  - Let’s now estimate $p(y = 3)$

$$p(y = 3) = \frac{1}{Nh^D} \sum_{n=1}^{N} K\left(\frac{x - x^{(n}}{h}\right) = \frac{1}{10 \times 4^1}\left[K\left(\frac{3-4}{4}\right) + K\left(\frac{3-5}{4}\right) + \cdots K\left(\frac{3-17}{4}\right)\right] = 0.0025$$

Exercise

- Given dataset $X \ = \{4, 5, 5, 6, 12, 14, 15, 15, 16, 17\}$, use Parzen windows to estimate the density $p(x)$ at $y = 3{,}10{,}15$; use $h = 4$
- Solution
  - Let's first draw the dataset to get an idea of the data



  - Let's now estimate $p(y = 3)$

$$p(y = 3) = \frac{1}{Nh^D} \sum_{n=1}^{N} K\left(\frac{x - x^{(n}}{h}\right) = \frac{1}{10 \times 4^1}\left[K\left(\frac{3-4}{4}\right) + K\left(\frac{3-5}{4}\right) + \cdots K\left(\frac{3-17}{4}\right)\right] = 0.0025$$

  - Similarly

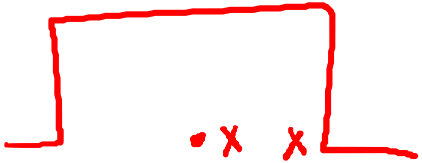
$$p(y = 10) = \frac{1}{10 \times 4^1}[0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0] = 0$$

$$p(y = 15) = \frac{1}{10 \times 4^1}[0 + 0 + 0 + 0 + 0 + 1 + 1 + 1 + 1 + 0] = 0.1$$

Smooth kernels

The Parzen window has several drawbacks

- It yields density estimates that have discontinuities
- It weights equally all points x_i, regardless of their distance to the estimation point x



Smooth kernels

The Parzen window has several drawbacks

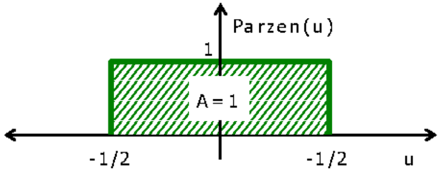
- It yields density estimates that have discontinuities
- It weights equally all points x_i, regardless of their distance to the estimation point x

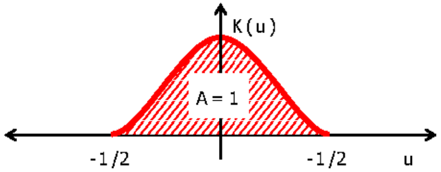
For these reasons, the Parzen window is commonly replaced with a smooth kernel function $K(u)$

$$\int_{R^D} K(x)dx = 1$$

- Usually, but not always, $K(u)$ will be a radially symmetric and unimodal pdf, such as the Gaussian $K(x) = (2\pi)^{-D/2}e^{-\frac{1}{2}x^T x}$
- Which leads to the density estimate

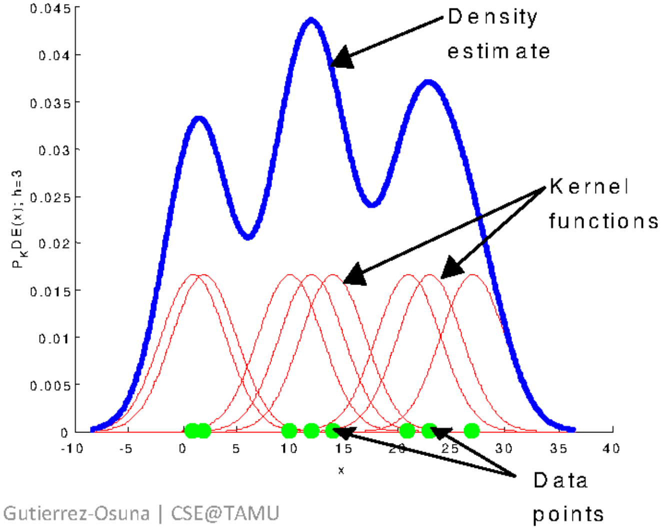
$$p_{KDE}(x) = \frac{1}{Nh^D}\sum_{n=1}^{N} K\left(\frac{x-x^{(k}}{h}\right)$$





Interpretation

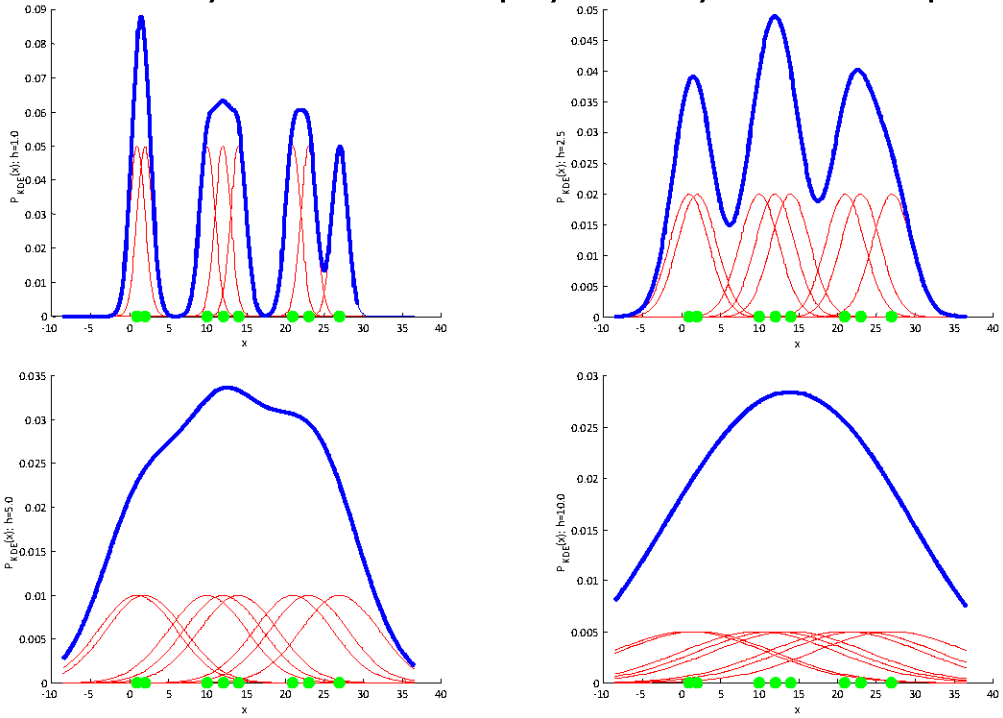
- Just as the Parzen window estimate can be seen as a sum of boxes centered at the data, the smooth kernel estimate is a sum of "bumps"
- The kernel function determines the shape of the bumps
- The parameter h, also called the smoothing parameter or bandwidth, determines their width



Bandwidth selection

The problem of choosing h is crucial in density estimation

- A large h will over-smooth the DE and mask the structure of the data
- A small h will yield a DE that is spiky and very hard to interpret



- We would like to find a value of h that minimizes the error between the estimated density and the true density
  - A natural measure is the MSE at the estimation point x, defined by

$$E[(p_{KDE}(x) - p(x))^2] = E[p_{KDE}(x) - p(x)]^2 + var(p_{KDE}(x))$$

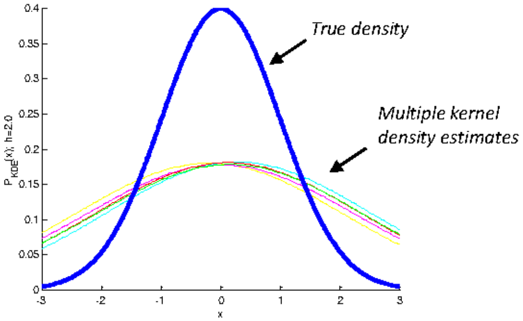
- We would like to find a value of h that minimizes the error between the estimated density and the true density
  - A natural measure is the MSE at the estimation point x, defined by

$$E[(p_{KDE}(x) - p(x))^2] = \underbrace{E[p_{KDE}(x) - p(x)]^2}_{bias} + \underbrace{var(p_{KDE}(x))}_{variance}$$

- This expression is an example of the bias-variance tradeoff that we saw in an earlier lecture: the bias can be reduced at the expense of the variance, and vice versa
  - The bias of an estimate is the systematic error incurred in the estimation
  - The variance of an estimate is the random error incurred in the estimation

- The bias-variance dilemma applied to bandwidth selection simply means that

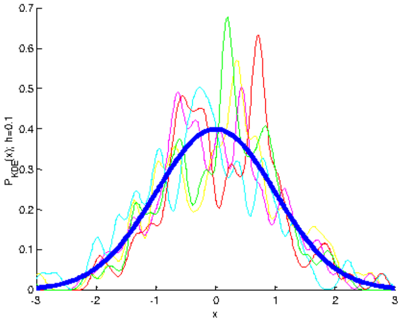
- The bias-variance dilemma applied to bandwidth selection simply means that
  - A large bandwidth will reduce the differences among the estimates of $p_{KDE}(x)$ for different data sets (the variance), but it will increase the bias of $p_{KDE}(x)$ with respect to the true density $p(x)$
  - A small bandwidth will reduce the bias of $p_{KDE}(x)$, at the expense of a larger variance in the estimates $p_{KDE}(x)$



BIAS

True density

Multiple kernel density estimates



VARIANCE

Bandwidth selection methods, univariate case

Subjective choice

- The natural way for choosing h is to plot out several curves and choose the estimate that best matches one's prior (subjective) ideas
- However, this method is not practical in pattern recognition since we typically have high-dimensional data

Reference to a standard distribution

- Assume a standard density function and find the value of the bandwidth that minimizes the integral of the square error (MISE)

$$h_{MISE} = \arg\min\left\{E\left[\int (p_{KDE}(x) - p(x))^2 dx\right]\right\}$$

- If we assume that the true distribution is Gaussian and we use a Gaussian kernel, it can be shown that the optimal value of h is

$$h^* = 1.06\sigma N^{-1/5}$$

  - where σ is the sample standard deviation and N is the number of training examples

- Better results can be obtained by
  - Using a robust measure of the spread instead of the sample variance, and
  - Reducing the coefficient 1.06 to better cope with multimodal densities
  - The optimal bandwidth then becomes

$$h^* = 0.9AN^{-1/5} \text{ where } A = \min\left(\sigma, \frac{IQR}{1.34}\right)$$

- IQR is the interquartile range, a robust estimate of the spread
  - IQR is the difference between the 75th percentile ($Q3$) and the 25th percentile ($Q1$): $IQR = Q3 - Q1$
  - A percentile rank is the proportion of examples in a distribution that a specific example is greater than or equal to

Maximum likelihood cross-validation

Maximum likelihood cross-validation

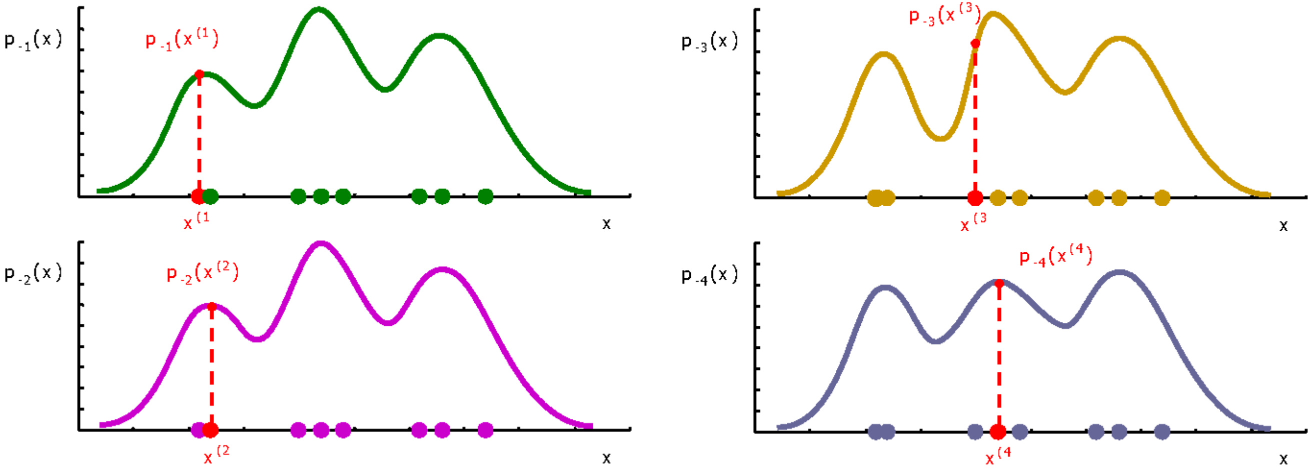
- The ML estimate of h is degenerate since it yields $h_{ML} = 0$, a density estimate with Dirac delta functions at each training data point

Maximum likelihood cross-validation

- The ML estimate of h is degenerate since it yields $h_{ML} = 0$, a density estimate with Dirac delta functions at each training data point
- A practical alternative is to maximize the “pseudo-likelihood” computed using leave-one-out cross-validation

$$h^* = \arg\max\left\{\frac{1}{N}\sum_{n=1}^{N} logp_{-n}\left(x^{(n)}\right)\right\}$$

$$where\ p_{-n}\left(x^{(n)}\right) = \frac{1}{(N-1)h}\sum_{\substack{m=1\\ m\neq \mathrm{n}}}^{N} K\left(\frac{x^{(n} - x^{(m}}{h}\right)$$



$p_{-1}(x)$
$p_{-1}(x^{(1)})$
$x^{(1}$
x
$p_{-3}(x)$
$p_{-3}(x^{(3)})$
$x^{(3}$
x
$p_{-2}(x)$
$p_{-2}(x^{(2)})$
$x^{(2}$
x
$p_{-4}(x)$
$p_{-4}(x^{(4)})$
$x^{(4}$
x

Multivariate density estimation

For the multivariate case, the KDE is

$$p_{KDE}(x) = \frac{1}{Nh^D} \sum_{n=1}^{N} K\left(\frac{x - x^{(n}}{h}\right)$$

- Notice that the bandwidth h is the same for all the axes, so this density estimate will be weight all the axis equally
- If one or several of the features has larger spread than the others, we should use a vector of smoothing parameters or even a full covariance matrix, which complicates the procedure

Multivariate density estimation

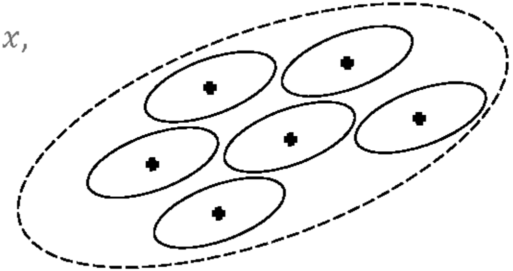
For the multivariate case, the KDE is

$$p_{KDE}(x) = \frac{1}{Nh^D} \sum_{n=1}^{N} K\left(\frac{x-x^{(n}}{h}\right)$$

- Notice that the bandwidth h is the same for all the axes, so this density estimate will be weight all the axis equally
- If one or several of the features has larger spread than the others, we should use a vector of smoothing parameters or even a full covariance matrix, which complicates the procedure

There are two basic alternatives to solve the scaling problem without having to use a more general KDE

- <u>Pre-scaling</u> each axis (normalize to unit variance, for instance)
- <u>Pre-whitening</u> the data (linearly transform so $\Sigma = I$), estimate the density, and then transform back [Fukunaga]
  - The whitening transform is $y = \Lambda^{-1/2} M^T x$, where Λ and M are the eigenvalue and eigenvector matrices of Σ
  - Fukunaga's method is equivalent to using a hyper-ellipsoidal kernel



Product kernels

A good alternative for multivariate KDE is the product kernel

$$p_{PKDE}(x) = \frac{1}{N}\sum_{i=1}^{N} K\left(x, x^{(n}, h_1, \dots h_D\right)$$

$$where\ \ K\left(x, x^{(n}, h_1, \dots h_D\right) = \frac{1}{h_1 \dots h_D}\prod_{d=1}^{D} K_d\left(\frac{x_d - x_d^{(n}}{h_d}\right)$$

- The product kernel consists of the product of one-dimensional kernels
  - Typically the same kernel function is used in each dimension ($K_d(x) = K(x)$), and only the bandwidths are allowed to differ
  - Bandwidth selection can then be performed with any of the methods presented for univariate density estimation

Product kernels

A good alternative for multivariate KDE is the product kernel

$$p_{PKDE}(x) = \frac{1}{N}\sum_{i=1}^{N} K\left(x, x^{(n}, h_1, \dots h_D\right)$$

$$where\ \ K\left(x, x^{(n}, h_1, \dots h_D\right) = \frac{1}{h_1 \dots h_D}\prod_{d=1}^{D} K_d\left(\frac{x_d - x_d^{(n}}{h_d}\right)$$

- The product kernel consists of the product of one-dimensional kernels
  - Typically the same kernel function is used in each dimension ($K_d(x) = K(x)$), and only the bandwidths are allowed to differ
  - Bandwidth selection can then be performed with any of the methods presented for univariate density estimation

$K\left(x, x^{(n}, h_1, \dots h_D\right)$ uses kernel independence

features are independent

Product kernels

A good alternative for multivariate KDE is the product kernel

$$p_{PKDE}(x) = \frac{1}{N}\sum_{i=1}^{N} K\left(x, x^{(n}, h_1, \dots h_D\right)$$

$$where\ \ K\left(x, x^{(n}, h_1, \dots h_D\right) = \frac{1}{h_1 \dots h_D}\prod_{d=1}^{D} K_d\left(\frac{x_d - x_d^{(n}}{h_d}\right)$$

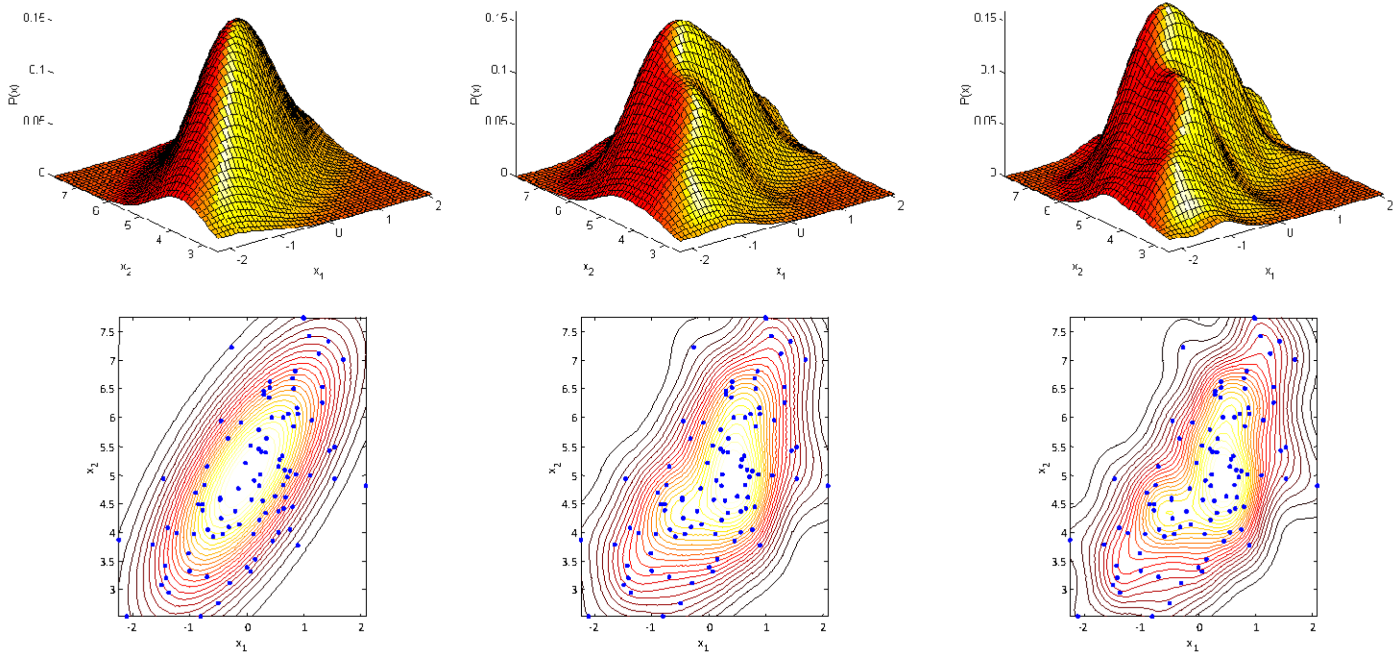
- The product kernel consists of the product of one-dimensional kernels
  - Typically the same kernel function is used in each dimension ($K_d(x) = K(x)$), and only the bandwidths are allowed to differ
  - Bandwidth selection can then be performed with any of the methods presented for univariate density estimation
- Note that although $K\left(x, x^{(n}, h_1, \dots h_D\right)$ uses kernel independence does not imply we assume the features are independent
  - If we assumed feature independence, the DE would have the expression

$$p_{FEAT-IND}(x) = \prod_{d=1}^{D}\frac{1}{Nh^D}\sum_{i=1}^{N} K_d\left(\frac{x_d - x_d^{(n}}{h_d}\right)$$

  - Notice how the order of the summation and product are reversed compared to the product kernel

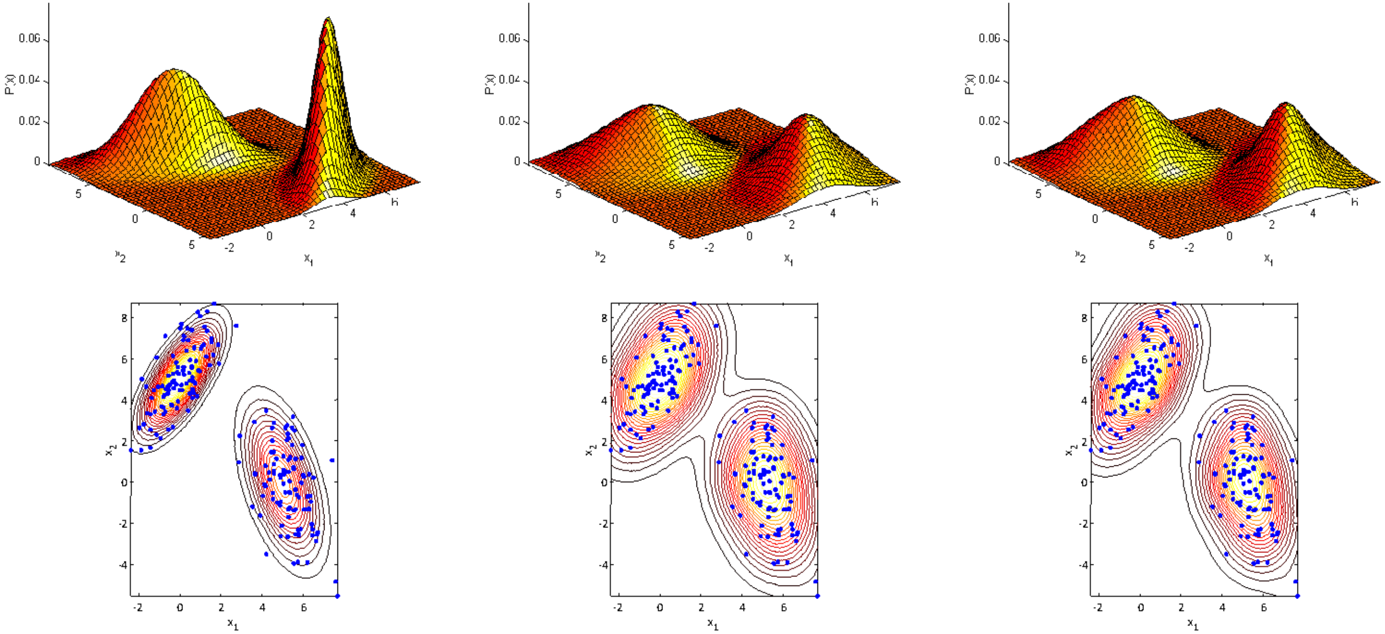
Example I

- This example shows the product KDE of a bivariate <u>unimodal</u> Gaussian
  - 100 data points were drawn from the distribution
  - The figures show the true density (left) and the estimates using $h = 1.06\sigma N^{-1/5}$ (middle) and $h = 0.9AN^{-1/5}$ (right)



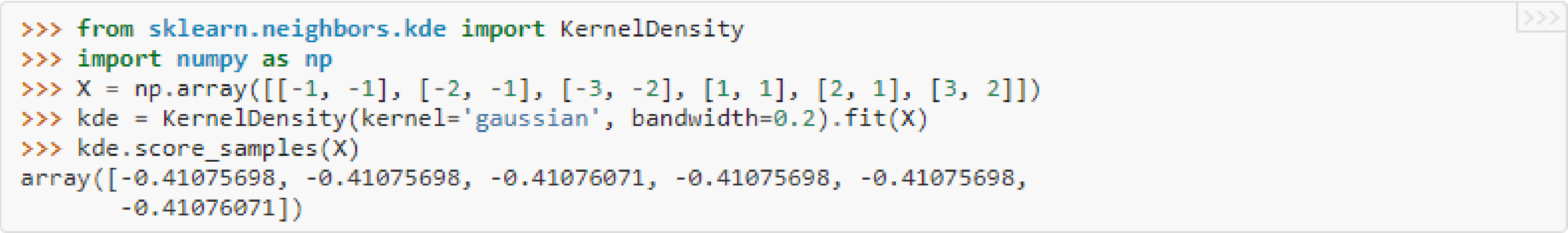
Example II

- This example shows the product KDE of a bivariate <u>bimodal</u> Gaussian
  - 100 data points were drawn from the distribution
  - The figures show the true density (left) and the estimates using $h = 1.06\sigma N^{-1/5}$ (middle) and $h = 0.9AN^{-1/5}$ (right)



KDE

- https://scikit-learn.org/stable/modules/density.html



```
>>> from sklearn.neighbors.kde import KernelDensity
>>> import numpy as np
>>> X = np.array([[-1, -1], [-2, -1], [-3, -2], [1, 1], [2, 1], [3, 2]])
>>> kde = KernelDensity(kernel='gaussian', bandwidth=0.2).fit(X)
>>> kde.score_samples(X)
array([-0.41075698, -0.41075698, -0.41076071, -0.41075698, -0.41075698,
       -0.41076071])
```

$$p(x) \cong \frac{k}{NV} \text{ where } \begin{cases} V & volume\ surrounding\ x \\ N & total\ \#examples \\ k & \#examples\ inside\ V \end{cases}$$

- We can fix V and determine k from the data. This leads to **kernel density estimation** (KDE), the subject of this lecture
- We can fix k and determine V from the data. This gives rise to the **k-nearest-neighbor** (kNN) approach