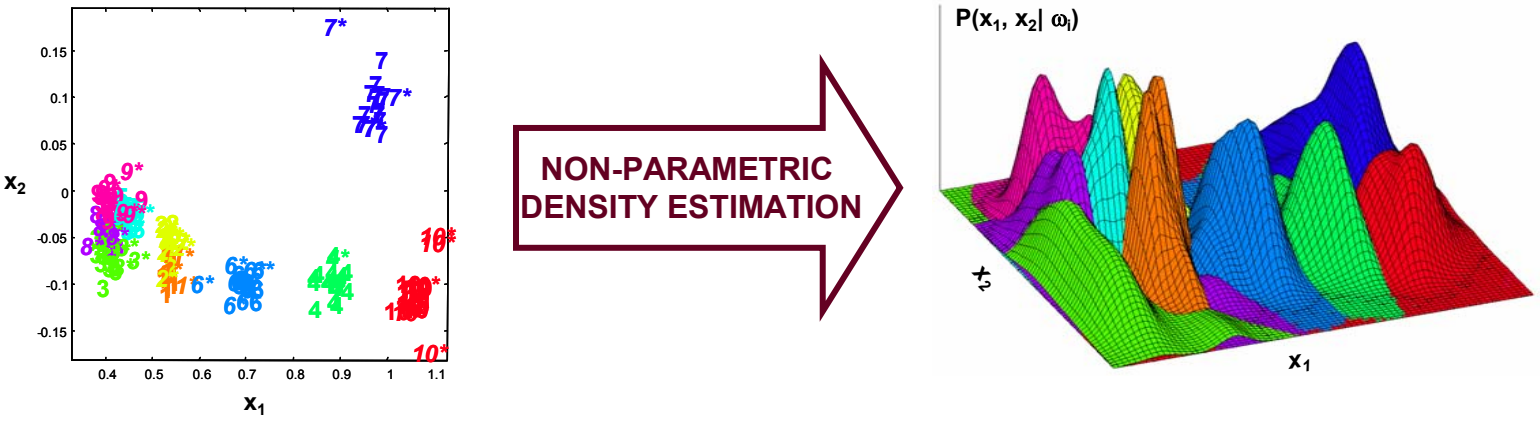
LECTURE 7: Kernel Density Estimation

- Non-parametric Density Estimation
- Histograms
- Parzen Windows
- Smooth Kernels
- Product Kernel Density Estimation
- The Naïve Bayes Classifier

Non-parametric density estimation

- **In the previous two lectures we have assumed that either**
  - The likelihoods $p(x|\omega_i)$ were known (Likelihood Ratio Test) or
  - At least the parametric form of the likelihoods were known (Parameter Estimation)
- **The methods that will be presented in the next two lectures do not afford such luxuries**
  - Instead, they attempt to estimate the density directly from the data without making any parametric assumptions about the underlying distribution
  - Sounds challenging? You bet!



The histogram

- **The simplest form of non-parametric D.E. is the familiar 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

$$P_H(x) = \frac{1}{N} \frac{[\text{number of } x^{(k} \text{ in same bin as } x]}{[\text{width of bin containing } x]}$$

    - The histogram requires two "parameters" to be defined: bin width and starting position of the first bin
- **The histogram is a very simple form of D.E., but it has several drawbacks**
  - The final shape of the density estimate depends on the starting position of the bins
    - For multivariate data, the final shape of the density is also affected by the orientation of the bins
  - The discontinuities of the estimate are not due to the underlying density, they are only an artifact of the chosen bin locations
    - These discontinuities make it very difficult, without experience, to grasp the structure of the data
  - A much more serious problem is the curse of dimensionality, since the number of bins grows exponentially with the number of dimensions
    - In high dimensions we would require a very large number of examples or else most of the bins would be empty
- **All these drawbacks make the histogram unsuitable for most practical applications except for rapid visualization of results in one or two dimensions**
  - Therefore, we will not spend more time looking at the histogram

Non-parametric density estimation, general formulation (1)

- **Let us return to the basic definition of probability to get a solid idea of what we are trying to accomplish**
  - 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 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

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

  - 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 $\Re$

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

From [Bishop, 1995]

Non-parametric density estimation, general formulation (2)

- 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

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

- This estimate becomes more accurate as we increase the number of sample points N and shrink the volume V

- **In practice the value of N (the total number of examples) is fixed**
  - In order to improve the accuracy of the estimate p(x) we could let V approach zero but then the region $\Re$ would then become so small that it would enclose no examples
  - This means that, in practice, we will have to find a compromise value for the volume V
    - Large enough to include enough examples within $\Re$
    - Small enough to support the assumption that p(x) is constant within $\Re$

From [Bishop, 1995]

Non-parametric density estimation, general formulation (3)

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

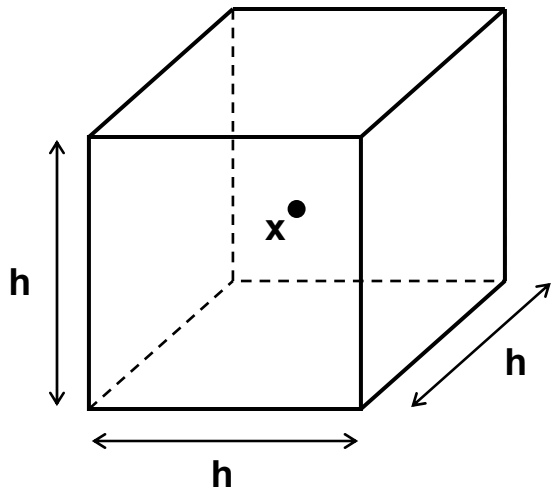
$$p(x) \cong \frac{k}{NV} \quad \text{where} \begin{cases} V \text{ is the volume surrounding } x \\ N \text{ is the total number of examples} \\ k \text{ is the number of examples inside } V \end{cases}$$

- **When applying this result to practical density estimation problems, two basic approaches can be adopted**
  - We can choose a fixed value of the volume V and determine k from the data. This leads to methods commonly referred to as **Kernel Density Estimation (KDE)**, which are the subject of this lecture
  - We can choose a fixed value of k and determine the corresponding volume V from the data. This gives rise to the **k Nearest Neighbor (kNN)** approach, which will be covered in the next lecture
- **It can be shown that both kNN and KDE converge to the true probability density as $N \to \infty$, provided that V shrinks with N, and k grows with N appropriately**

From [Bishop, 1995]

Parzen windows (1)

- **Suppose that the region $\Re$ that encloses the k examples is a hypercube with sides of length h centered at the estimation point x**
  - Then its volume is given by $V=h^D$, where D is the number of dimensions
- **To find the number of examples that fall within this region we define a kernel function K(u)**



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

  - 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 the point $x^{(n}$ is inside a hypercube of side h centered on x, and zero otherwise

From [Bishop, 1995]

Parzen windows (2)

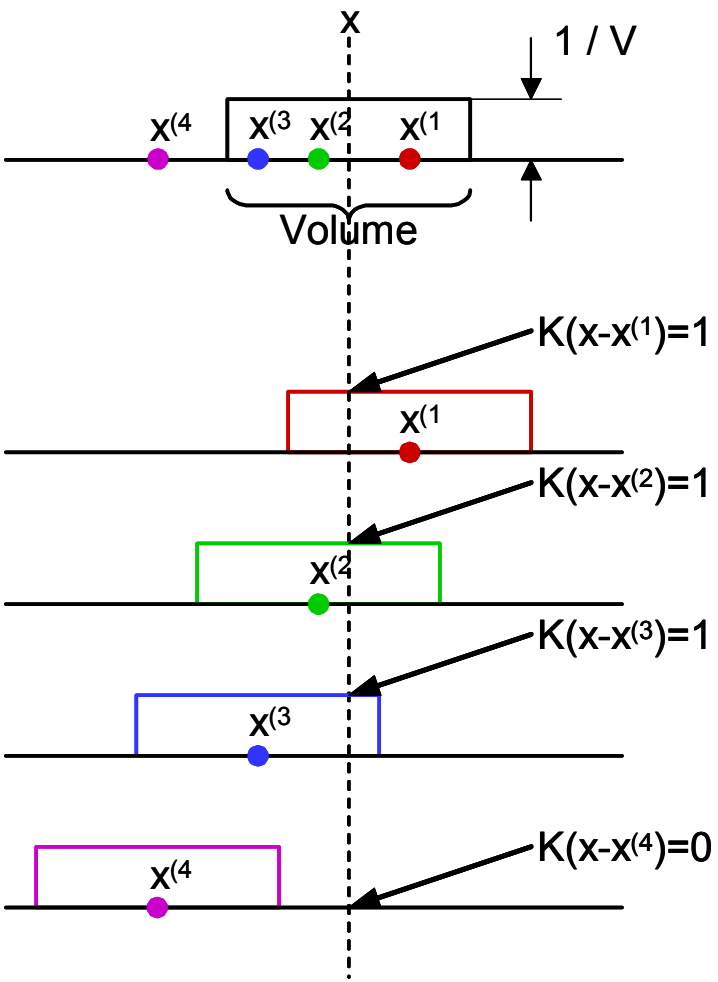
- **The total number of points inside the hypercube is then**

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

- **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 that the Parzen window density estimate resembles the histogram, with the exception that the bin locations are determined by the data points



From [Bishop, 1995]

Parzen windows (3)

- **To understand the role of the kernel function we compute the expectation of the probability estimate p(x)**

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

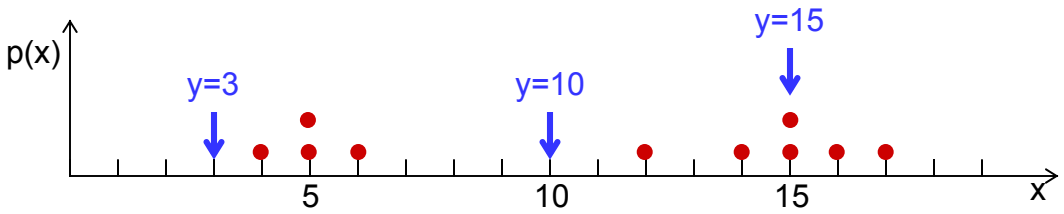
$$= \frac{1}{h^D} E\left[K\left(\frac{x - x^{(n}}{h}\right)\right] = \frac{1}{h^D}\int K\left(\frac{x - x'}{h}\right) p(x')dx'$$

  - where we have assumed that the vectors $x^{(n}$ are drawn independently from the true density p(x)
- **We can see that the expectation of the estimated density $p_{KDE}(x)$ is a convolution of the true density p(x) with the kernel function**
  - The width *h* of the kernel plays the role of a smoothing parameter: the wider the kernel function, the smoother the estimate $p_{KDE}(x)$
- **For $h \to 0$, the kernel approaches a Dirac delta function and $p_{KDE}(x)$ approaches the true density**
  - However, in practice we have a finite number of points, so *h* cannot be made arbitrarily small, since the density estimate $p_{KDE}(x)$ would then degenerate to a set of impulses located at the training data points

From [Bishop, 1995]

Numeric exercise

- **Given the dataset below, use Parzen windows to estimate the density p(x) at y=3,10,15. Use a bandwidth of h=4**
  - $X = \{x^{(1}, x^{(2},\ldots x^{(N}\} = \{4, 5, 5, 6, 12, 14, 15, 15, 16, 17\}$
- **Solution**
  - Let's first draw the dataset to get an idea of what numerical results we should expect



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

$$p_{KDE}(y=3)=\frac{1}{Nh^D}\sum_{n=1}^{N}K\left(\frac{y-x^{(n}}{h}\right)=\frac{1}{10\times 4^1}\left[\underbrace{K\left(\frac{3-4}{4}\right)}_{-1/4}+\underbrace{K\left(\frac{3-5}{4}\right)}_{-1/2}+\underbrace{K\left(\frac{3-5}{4}\right)}_{-1/2}+\underbrace{K\left(\frac{3-6}{4}\right)}_{-1}+\ldots+\underbrace{K\left(\frac{3-17}{4}\right)}_{-13/4}\right]=$$

$$=\frac{1}{10\times 4^1}[1+0+0+0+0+0+0+0+0+0]=\frac{1}{10\times 4}=0.025$$

  - Similarly

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

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

Smooth kernels (1)

- **The Parzen window has several drawbacks**
  - Yields density estimates that have discontinuities
  - Weights equally all the points x_i, regardless of their distance to the estimation point x
- **It is easy to to overcome some of these difficulties by generalizing the Parzen window with a smooth kernel function K(u) which satisfies the condition**

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

  - Usually, but not not always, K(u) will be a radially symmetric and unimodal probability density function, such as the multivariate Gaussian density function

$$K(x) = \frac{1}{(2\pi)^{D/2}} \exp\left(-\frac{1}{2}x^T x\right)$$

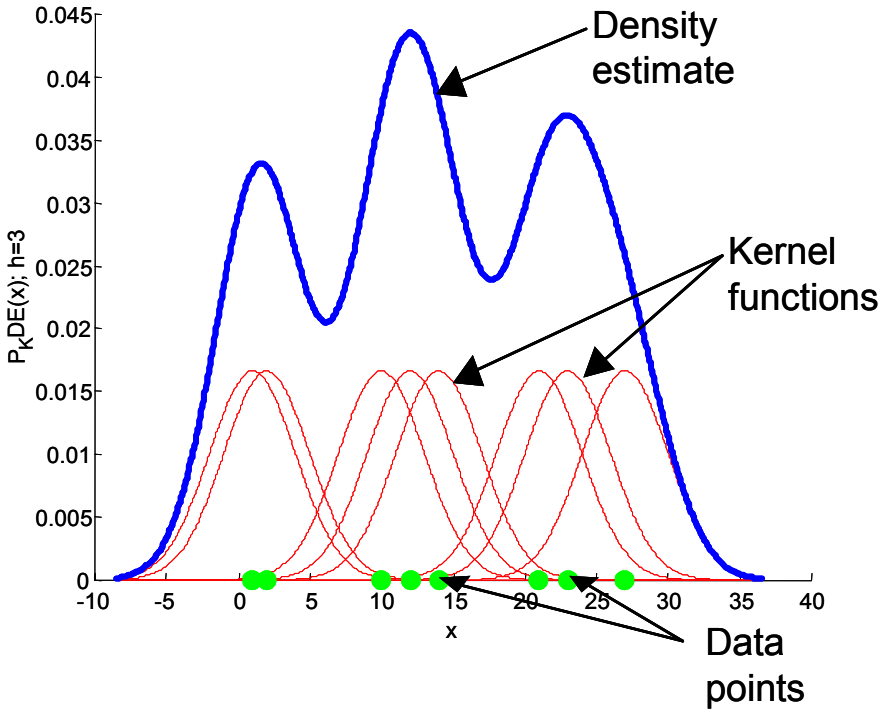
  - where the expression of the density estimate remains the same as with Parzen windows

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

From [Bishop, 1995]

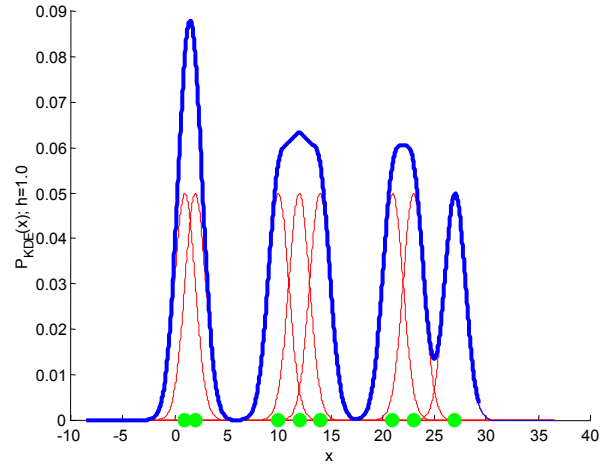
Smooth kernels (2)

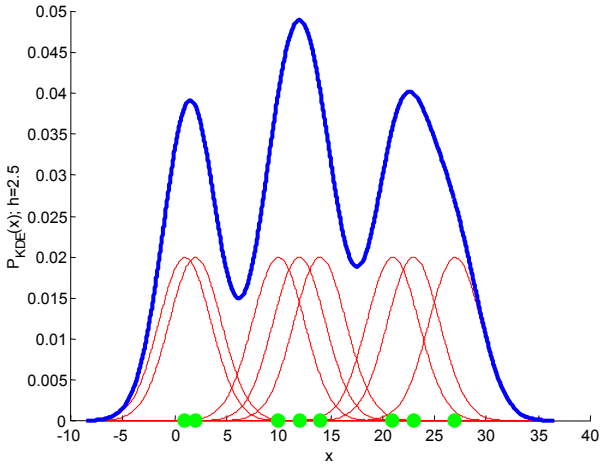
- **Just as the Parzen window estimate can be considered a sum of boxes centered at the observations, the smooth kernel estimate is a sum of "bumps" placed at the data points**
  - The kernel function determines the shape of the bumps
  - The parameter h, also called the **smoothing parameter** or **bandwidth**,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

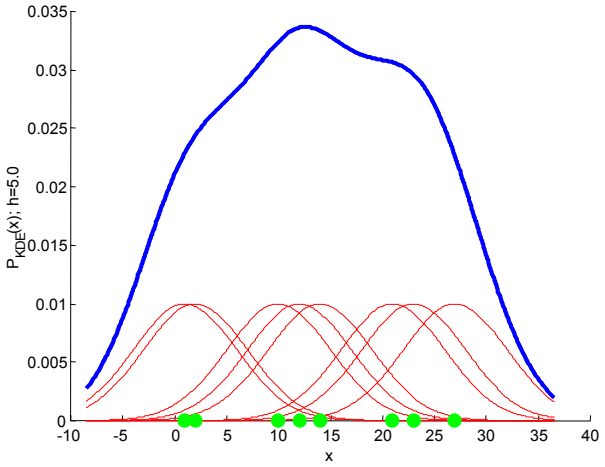


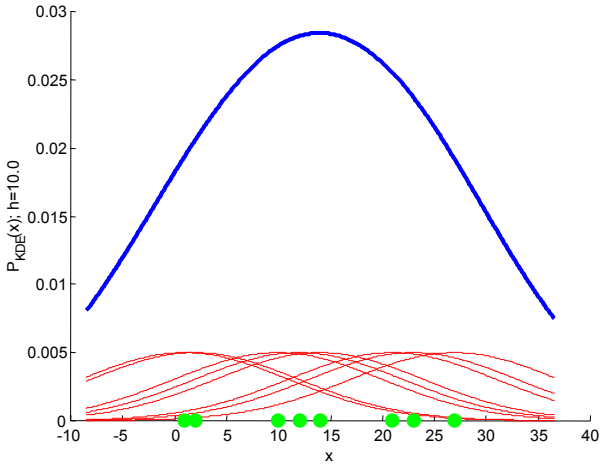
Choosing the bandwidth: univariate case (1)

- **The problem of choosing the bandwidth is crucial in density estimation**
  - A large bandwidth will over-smooth the density and mask the structure in the data
  - A small bandwidth will yield a density estimate that is spiky and very hard to interpret









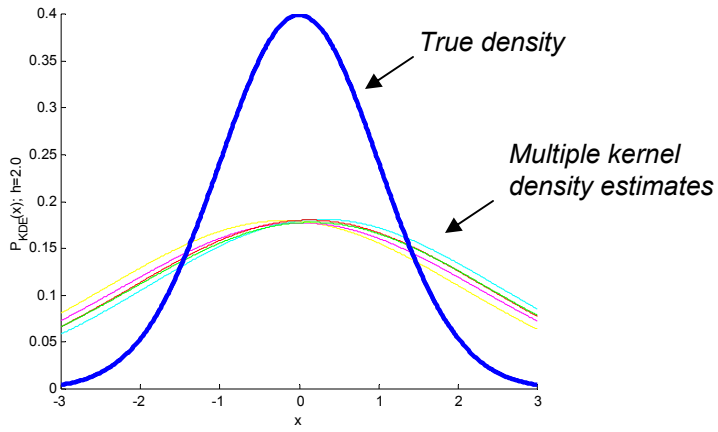
Choosing the bandwidth: univariate case (2)

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

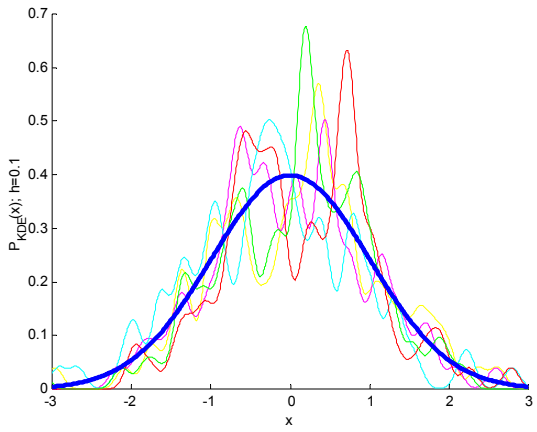
$$\mathrm{MSE}_x(p_{KDE}) = E\left[(p_{KDE}(x) - p(x))^2\right] = \underbrace{\{E[p_{KDE}(x) - p(x)]\}^2}_{\text{bias}} + \underbrace{\mathrm{var}(p_{KDE}(x))}_{\text{variance}}$$

- **This expression is an example of the bias-variance tradeoff that we saw earlier in the course: 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**
  - 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



VARIANCE



Bandwidth selection methods, univariate case (3)

- **Subjective choice**
  - The natural way for choosing the smoothing parameter is to plot out several curves and choose the estimate that is most in accordance with 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)

$$h_{opt} = \underset{h}{\operatorname{argmin}}\{\mathrm{MISE}(p_{KDE}(x))\} = \underset{h}{\operatorname{argmin}}\left\{E\left[\int (p_{KDE}(x) - p(x))^2 dx\right]\right\}$$

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

$$h_{opt} = 1.06\sigma N^{-1/5}$$

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

From [Silverman, 1986]

Bandwidth selection methods, univariate case (4)

- Better results can be obtained if we use a robust measure of the spread instead of the sample variance and we reduce the coefficient 1.06 to better cope with multimodal densities. The optimal bandwidth then becomes

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

  - IQR is the interquartile range, a robust estimate of the spread. It is computed as one half the difference between the 75th percentile (Q3) and the 25th percentile (Q1). The formula for semi-interquartile range is therefore: (Q3-Q1)/2
    - A percentile rank is the proportion of examples in a distribution that a specific example is greater than or equal to

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

$$h_{MLCV} = \underset{h}{\operatorname{argmax}}\left\{\frac{1}{N}\sum_{n=1}^{N}\log p_{-n}\left(x^{(n)}\right)\right\}$$

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

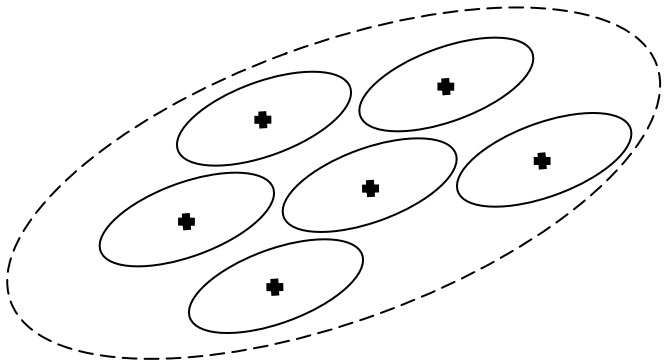
From [Silverman, 1986]

Multivariate density estimation

- **The derived expression of the estimate $P_{KDE}(x)$ for multiple dimensions was**

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

  - Notice that the bandwidth h is the same for all the axes, so this density estimate will be weight all the axis equally
- **However, if the spread of the data is much greater in one of the coordinate directions 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 kernel density estimate**
  - **Pre-scale each axis** (normalize to unit variance, for instance)
  - **Pre-whiten the data** (linearly transform to have unit covariance matrix), estimate the density, and then transform back [Fukunaga]
    - The whitening transform is simply $y=\Lambda^{-1/2}M^T x$, where Λ and M are the eigenvalue and eigenvector matrices of the sample covariance of x
    - Fukunaga's method is equivalent to using a hyper-ellipsoidal kernel



Product kernels

- **A very popular method for performing multivariate density estimation is the product kernel, defined as**

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

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

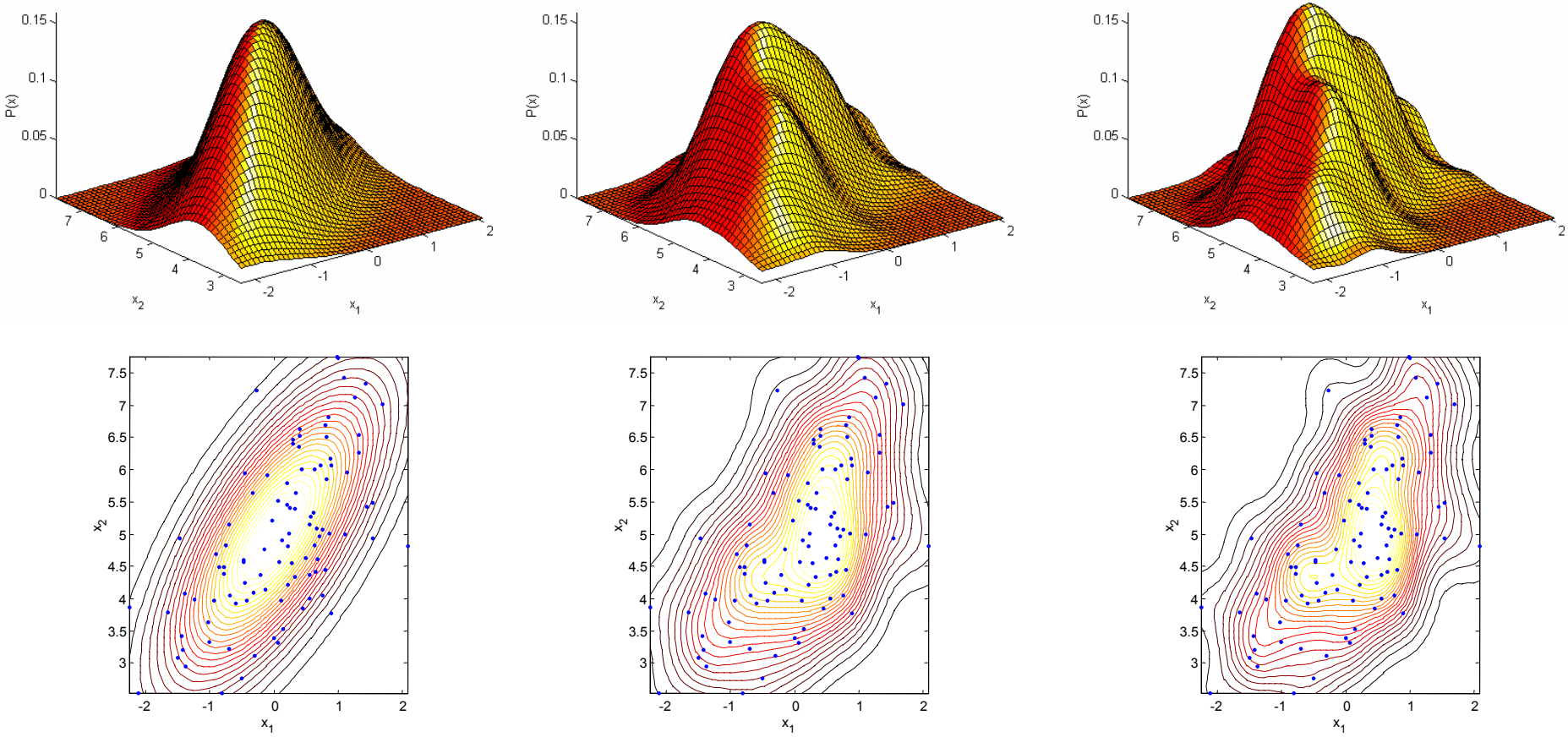
  - The product kernel consists of the product of one-dimensional kernels
- **Typically the same kernel function is used in each dimension ($K_d(x)=K(x)$), and only the bandwidths are allowed to differ**
  - Bandwidth selection can then be performed with any of the methods presented for univariate density estimation
- **It is important to notice that although the expression of $K(x,x^{(n)},h_1,\ldots h_D)$ uses kernel independence, this does not imply that any type of feature independence is being assumed**
  - A density estimation method that assumed feature independence would have the following expression

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

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

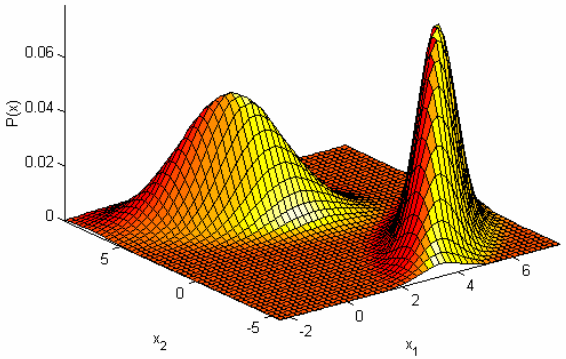
Product kernel, example 1

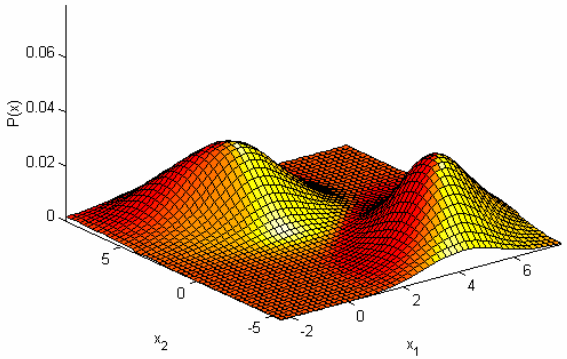
- **This example shows the product kernel density estimate of a bivariate unimodal Gaussian distribution**
  - 100 data points were drawn from the distribution
  - The figures show the true density (left) and the estimates using $h=1.06\sigma N^{-1/5}$ (middle) and $h=0.9AN^{-1/5}$ (right)

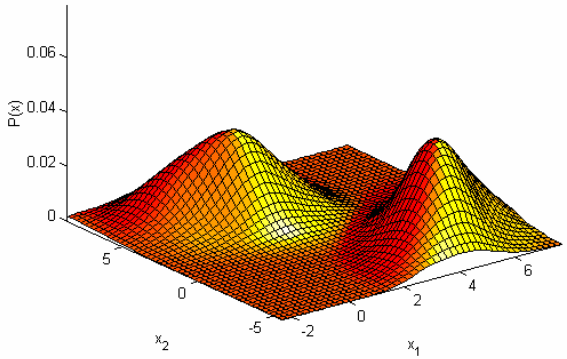


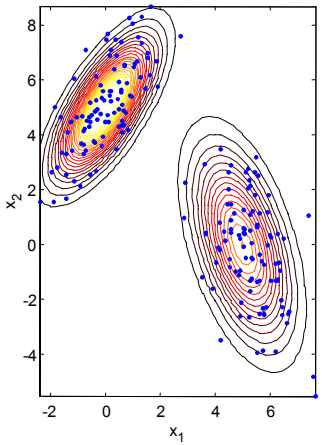
Product kernel, example 2

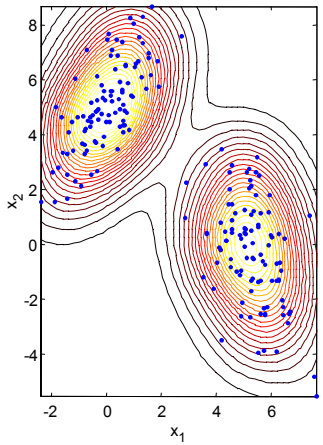
- **This example shows the product kernel density estimate of a bivariate bimodal Gaussian distribution**
  - 100 data points were drawn from the distribution
  - The figures show the true density (left) and the estimates using $h=1.06\sigma N^{-1/5}$ (middle) and $h=0.9AN^{-1/5}$ (right)

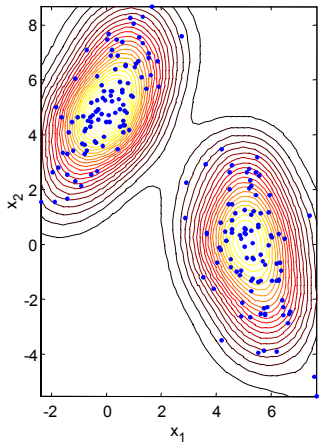












Naïve Bayes classifier

- **Recall that the Bayes classifier is given by the following family of discriminant functions**

$$\text{choose } \omega_i \text{ if } g_i(x) > g_j(x)\ \forall j \neq i$$

$$\text{where } g_i(x) = P(\omega_i \mid x)$$

- **Using Bayes rule, these discriminant functions can be expressed as**

$$g_i(x) = P(\omega_i \mid x) \propto P(x \mid \omega_i) P(\omega_i)$$

  - where $P(\omega_i)$ is our prior knowledge and $P(x|\omega_i)$ is obtained through density estimation
- **Although we have presented density estimation methods that allow us to estimate the multivariate likelihood $P(x|\omega_i)$, the curse of dimensionality makes it a very tough problem!**
- **One highly practical simplification of the Bayes classifier is the so-called Naïve Bayes classifier**
  - The Naïve Bayes classifier makes the assumption that the features are class-conditionally independent

$$P(x \mid \omega_i) = \prod_{d=1}^{D} P(x(d) \mid \omega_i)$$

    - It is important to notice that this assumption is not as rigid as assuming independent features $\mathbf{P(x) = \prod_{d=1}^{D} P(x(d))}$
  - Merging this expression into the discriminant function yields the decision rule for the Naïve Bayes classifier

$$g_{i,NB}(x) = P(\omega_i) \prod_{d=1}^{D} P(x(d) \mid \omega_i) \qquad \textbf{Naïve Bayes Classifier}$$

- **The main advantage of the Naïve Bayes classifier is that we only need to compute the univariate densities *P(x(d)|ω_i)*, which is a much easier problem than estimating the multivariate density *P(x|ω_i)***
  - Despite its simplicity, the Naïve Bayes has been shown to have comparable performance to artificial neural networks and decision tree learning in some domains