Non-Parametric Classifiers

K-NN density estimation,

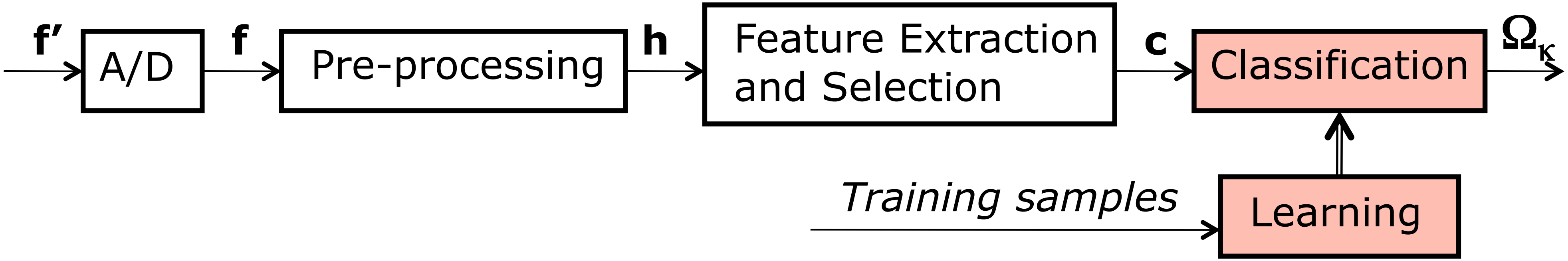
Parzen Windows

Dr. Elli Angelopoulou

Lehrstuhl für Mustererkennung (Informatik 5)

Friedrich-Alexander-Universität Erlangen-Nürnberg

Pattern Recognition Pipeline



- Classification
  - Statistical classifiers
    - Bayesian classifier
    - Gaussian classifier
  - Polynomial classifiers
  - Non-Parametric classifiers
    - k-Nearest-Neighbor density estimation
    - Parzen windows
    - Artificial neural networks

Probability Estimates

- A Bayesian classifier decides for the class with the highest posterior probability.

$$\delta(\Omega_\lambda|\vec{c}) = \begin{cases} 1 & \text{if } \lambda = \underset{\kappa}{\operatorname{argmax}}\, p(\Omega_\kappa|\vec{c}) \\ 0 & \text{otherwise} \end{cases}$$

- We can compute which class maximizes the posterior probability by exploiting the Bayesian rule and using the prior class probability $p\left(\Omega_\kappa\right)$ and the class-conditional likelihood $p\left(\vec{c}|\Omega_\kappa\right)$:

$$\underset{\kappa}{\operatorname{argmax}}\, p\left(\Omega_\kappa|\vec{c}\right) = \underset{\kappa}{\operatorname{argmax}}\, p(\Omega_\kappa)\, p(\vec{c}|\Omega_\kappa)$$

Probability Estimates – Special Cases

- In the special case of a Gaussian classifier, there exists a parametric density function (i.e. normal distribution) that describes the class-conditional density.

$$p(\vec{c}|\Omega_\kappa) \approx \mathcal{N}(\vec{c},\vec{\mu}_\kappa,\Sigma_\kappa)$$

- In that case one can use Maximum Likelihood Estimation to obtain values for the parameters of the probability density function (pdf): the mean $\vec{\mu}_\kappa$ and the covariance Σ_κ.

Probability Estimates – General

- Often, we have no information about the model of the underlying probability density function, about how the features are distributed.
- How can we obtain estimates of the posterior probability, or the class prior or the likelihood?
- We could try to approximate the distribution of the features with a more general model like a mixture of Gaussians, or we use a *non-parametric approach*.
- **Non-parametric classifiers** are specifically designed for handling non-parametric representations of probability densities.

Non-Parametric Density Estimators

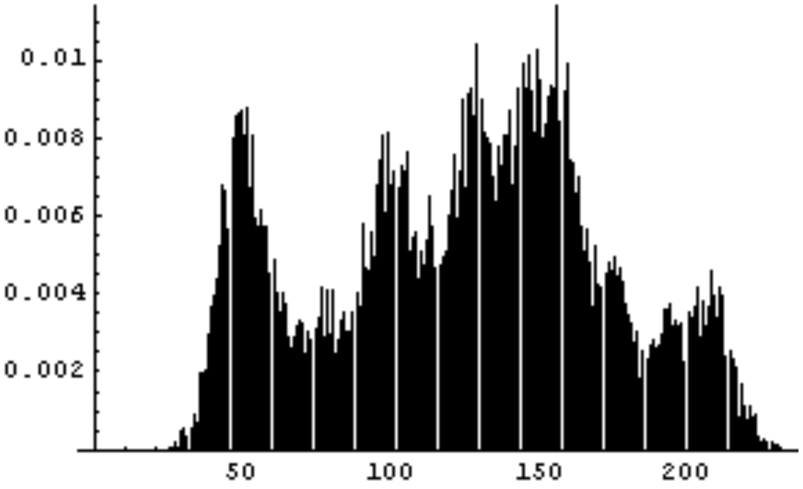
- The various types of non-parametric classifiers differ from one another by the kind of non-parametric density estimator that they use.
- A non-parametric density estimator is the term used for describing a methodology for estimating the probability density function of a random variable from a finite sample set.
- The simplest nonparametric density estimator is the **histogram estimator**, where we obtain pdf estimates by computing the relative frequencies in a histogram.

Histogram Estimator

- Formally, a histogram is a function *g*(*i*) that counts the number of observations that fall into each of *b* disjoint categories (known as bins). If *N* is the total number of observations then the histogram function must satisfy the following equation:

$$N = \sum_{i=1}^{b} g(i)$$

- The graph of a histogram is merely one way to represent a histogram.



Histogram of the Lena image courtesy of J.-M. Vezien http://www.limsi.fr/Individu/vezien/trima.html

Histogram and Relative Frequencies

- By counting how many samples fall within each bin one can compute relative frequencies.
- For scalar features, it is straightforward to obtain relative frequency estimates. The probability that the scalar feature c_1 has the particular value v is:

$$p(c_1 = v) = \frac{m}{N} = \frac{\#\,\text{samples in the bin of v}}{\text{total \# samples}}$$

- Estimating the probability of a particular value occurring simply involves counting.

A Realistic Example

- Consider a training data set of N=10^6 (1 million) samples. The Compaq skin database for example is composed of 2000 images with 22.669.739 skin pixels and 149.119.846 non-skin pixels.
- Assume a 15-dimensional feature vector, $\vec{c} \in R^{15}$.
- Let us construct a histogram with *b*=10 bins in each of the 15 dimensions.
- We have a total of 10^{15} bins.
- There are many more bins than feature vectors.
- Even in the best case scenario, where our training samples are nicely spread and we get no duplicates, we still have at least 10^{15}-10^6 empty bins.

Remarks on Histograms

- There is no "best" number of bins.
- Methods have been developed for determining the optimal number of bins, but they generally make strong assumptions about the shape of the distribution.
- Different bin sizes can reveal different characteristics of the data.
- The appropriate bin width is typically determined via experimentation.
- In a similar manner, the end points of the bins can affect the resulting estimated density.
- Lastly, histograms, unlike pdfs are discontinuous.

Region-Based Approach

- What is the probability that a particular feature vector $\vec{c}$ will fall within a specific sub-volume (region), say *R*, of the feature space?
- **If** we knew the probability density function it would be straightforward to compute such a probability.
- The probability of observing a feature $\vec{c}$ in a specific sub-region *R* of the feature space (if the density function is known) is:

$$P = p(\vec{c} \in R) = \int_R p(\vec{c})d\vec{c} \approx p(\vec{c})V$$

where *V* is the volume of region *R*.

Regions and Class-Conditional Probabilities

- Assume that all the features vectors that fall in region *R* are all associated with class Ω_κ and that all the features vectors that belong to class Ω_κ fall in region *R*.
- One can then get an estimate of the class-conditional probability for class Ω_κ as follows:

$$p(\vec{c}|\Omega_\kappa) = \frac{p(\vec{c} \in R)}{V} = \frac{g(\kappa)}{NV}$$

where $g(\kappa)$ is the number of samples in region *R*, i.e. in class Ω_κ.

Remarks on Region-Based Approaches

- Thus, using regions and knowing the pdf one can measure $P = p(\vec{c} \in R)$ and when regions are associated with classes $p(\vec{c} \mid \Omega_\kappa)$.
- Region-based density estimators provide good class-conditional approximations when the volumes are infinitesimally small, $V \rightarrow 0$, and $N \rightarrow \infty$.
- Histograms can be seen as a special case of a region-based approach, where all regions have V=1.
- How do we estimate P if we don't have the pdf?
- How do we compute the size of the volume V?

Relative Frequency

- Assume we have *N* training samples which are uniformly distributed.
- Using the binomial distribution we can compute the probability that *K* samples (out of the *N*) fall within the region *R* as:

$$P(|\vec{c} \in R| = K) = \binom{N}{K} P^K (1 - P)^{N-K}$$

where *P* in this equation is the probability of having 1 feature vector fall in region *R*. Recall that $P = p(\vec{c} \in R)$ and we are examining one feature at a time.

Relative Frequency – Mean Value

- According to the binomial distribution, the expected value of *K* is:

$$E\{K\} = NP \Rightarrow E\{K/N\} = P$$

- This equation indicates that when we compute the relative frequencies (i.e. how many samples fall within a region over the total number of samples), we get as a mean the *P* we were looking for.
- In other words from the relative frequencies we can get an unbiased estimate of *P*.

Relative Frequency - Variance

- Similarly, according to the binomial distribution the variance of *K* is:

$$E\{(K-NP)^2\} = NP(1-P)$$

divide both sides with N^2

$$\Rightarrow E\left\{\left({K}\!\big/\!{N} - P\right)^2\right\} = \frac{P(1-P)}{N}$$

- This last equation indicates that as $N \rightarrow \infty$ the variance in relative frequencies, *K/N*, approaches 0.

- So relative frequencies have a mean that is approximately *P* and a variance that approaches 0 for an infinitely large sample set.

Conclusions on Relative Frequencies

- For uniformly distributed training samples:

1. The expected value of relative frequencies is P.
2. If $N \rightarrow \infty$, the variance of the relative frequencies approaches 0.

- These two facts imply that the probability density function of the relative frequencies *p*(*K*/*N*) is sharply peaked.
- Recall that $P = p\left(\vec{c} \in R\right)$ and that given *P* one can estimate the likelihood $p(\vec{c}|\Omega_\kappa) = p\left(\vec{c} \in R\right)/V$.
- Thus, one can obtain density estimates of the class-conditional density by analyzing relative frequencies in different regions of feature space.

Density Estimation from Relative Frequencies

- Recall that: $P = p(\vec{c} \in R) = \int_R p(\vec{c})d\vec{c} \approx p(\vec{c})V$
- Thus:

$$p(\vec{c}) = \frac{P}{V}$$

- We have also shown that for uniform distributions from the relative frequencies we obtain an estimate of *P*:

$$P = \frac{K}{N}$$

- Hence, from the relative frequencies we can also get an estimate of $p(\vec{c})$:

$$p(\vec{c}) = \frac{K}{NV}$$

Remarks on Density Estimation

$$p(\vec{c}) = \frac{K}{NV}$$

- The larger the size of the training set *N*, the better.
- The smaller the volume *V*, the more accurate the estimate.
- So what is the right choice for *V*?
- Option 1: Use a fixed value for *K* and find the corresponding V from the data

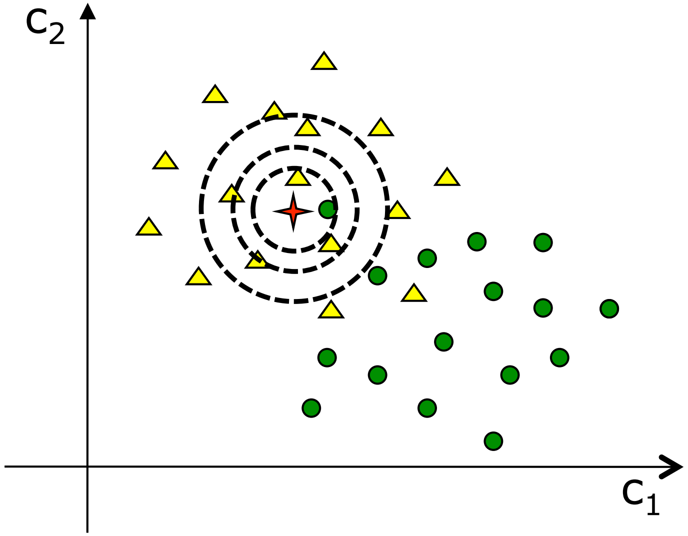
=> **K-nearest-neighbor** (fix *K*, look for a *V*)

- Option 2: Use a fixed volume *V* and find the corresponding value of K from the data

=> **kernel-based density estimation** (fix V, look for a K)

K-Nearest Neighbor Density Estimation

- A K-nearest neighbor classifier, assigns a feature vector $\vec{c}_{new}$ to the class that gets the majority vote among its *K* nearest neighbors in feature space.
- How does this relate to $p(\vec{c}) = K/(NV)$?



- Grow a sphere centered around $\vec{c}_{new}$. Stop when it is big enough to hold *K* samples. The volume of the sphere is the volume *V*.

Density Estimation – Fixed *K*

- The volume *V* is a function of *K*, $V(K)$.
- Thus, we now have:

$$p(\vec{c}) = \frac{K}{NV(K)}$$

- Different *K* values will give different pdf estimates.
- The larger the *K* the smoother the pdf estimate.
- From the simplest viewpoint, a classifier that uses K-nearest neighbor density estimation, is a K-nearest neighbor classifier.
- From a Bayesian viewpoint, such a classifier uses the K-nearest neighbors to obtain a posterior probability estimate.

K-NN Density Estimates and Bayes Classific.

- Assume we have *N* training samples $\vec{c}_1, \vec{c}_2, \ldots, \vec{c}_N$.
- Let N_κ of these *N* features belong to class Ω_κ.
- Assume *L* disjoint classes: $\sum_{\kappa=1}^{L} N_\kappa = N$
- Consider a sphere around $\vec{c}$ large enough to hold *K* features. Then

1. The class conditional density is $p(\vec{c}|\Omega_\kappa) = \dfrac{K_\kappa}{N_\kappa V}$

   where K_κ is the number of features in the sphere that belong to class Ω_κ.
2. The pdf of the feature space is $p(\vec{c}) = \dfrac{K}{NV}$
3. The class prior is $p(\Omega_\kappa) = \dfrac{N_\kappa}{N}$

K-NN Density Estimates and Bayes Classific.

- According to the Bayesian decision rule:

$$\lambda = \operatorname*{argmax}_{\kappa} p\left(\Omega_\kappa \middle| \vec{c}\right) = \operatorname*{argmax}_{\kappa} \frac{p(\Omega_k)\, p\left(\vec{c} \middle| \Omega_\kappa\right)}{p(\vec{c})}$$

$$= \operatorname*{argmax}_{\kappa} \frac{\frac{N_k}{N} \frac{K_k}{N_k V}}{\frac{K}{NV}}$$

$$= \operatorname*{argmax}_{\kappa} \frac{K_\kappa}{K}$$

- So to maximize the posterior probability one has to maximize the ratio K_κ/K. One must decide for the class that has the most samples in the sphere that includes just K features.

A K-NN Theorem

- Recall that p_B is the error probability of the ideal Bayesian classifier and is the lower limit in the probability of misclassification that we can achieve.
- Let p_{NN} be the error probability of the K-nearest neighbor classifier. Then:

$$p_B \le p_{NN} \le p_B\left(2 - \frac{K}{K-1} p_B\right)$$

where K is the number of neighbors in the K nearest neighbor classifier.

- Furthermore, when $K = N \rightarrow \infty$:

$$p_B \le p_{NN} \le 2p_B$$

Conclusions on the K-NN Classifier

- So the K-nearest neighbor classifier, though simple has a pretty good performance.
- So why bother with other more complex classifiers?
- We need to store all the training samples and use them during each classification decision.

Kernel Density Estimation

- Recall that for uniformly distributed samples

$$p(\vec{c}) = \frac{K}{NV}$$

- We have already examined how we can obtain an estimate of the pdf by selecting a value for K and allowing V to vary.
- We can also fix V and allow K to vary. This is called kernel density estimation.
- Kernel density estimation is a fundamental data *smoothing* problem, where inferences about the population are made based on finite set of data samples.
- It is also known as the Parzen-Rosenblatt window(s) method, or just Parzen window(s).

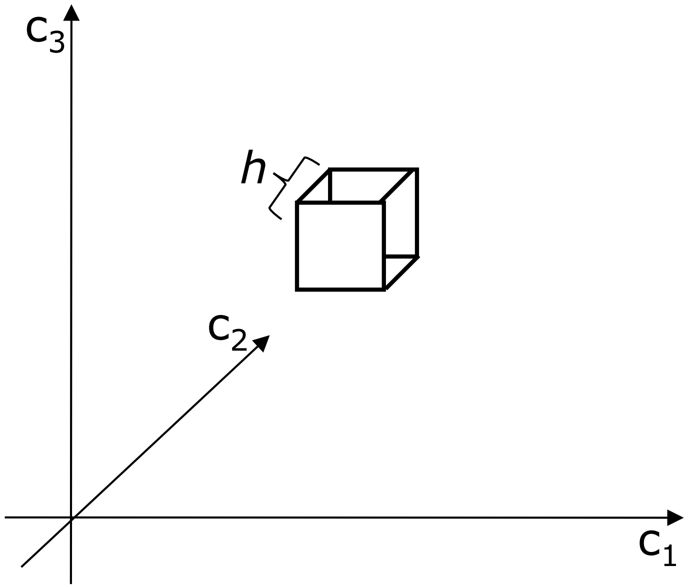
Main Concept of Parzen Windows

- Recall that we said that two of the problems with histograms is that the obtained estimates :
  - depend on the width of the bins
  - depend on the endpoints of bins
- Kernel density estimators, remove the dependence on the end points of the bins, by centering each of the bins (more appropriately hypecubes) at each data point.
- The width of the block can vary.
- So instead of the bins of the histogram we have hypercubes of side length *h*, which center around each feature vector $\vec{c}$.

Hypercube

- Our goal is to approximate $p(\vec{c}) = K/NV$, where *V* is the volume of a region *R* in which *K* samples exist.

- Let us assume that the region *R* we are considering is a *d*-dimensional hypercube (i.e. we are in *d*-dimensional feature space) with side length *h*.



- The volume of the hypercube is:

$$V = h^d$$

Kernel Function

- As a first step we need to measure distances within and around the hypercube,
- Given a new feature vector $\vec{c}$ and a training sample $\vec{c}_i$ compute a normalized distance vector $\vec{u}$ between them: $\vec{u} = d(\vec{c},\vec{c}_i)$
- The vector $\vec{u}$ is normalized by the length of the cube.
- A kernel function can then be defined as:

$$H(\vec{u}) = \begin{cases} 1 \;\; \text{if } |u_j| < 1/2 \;\; \text{for } j = 1,2,\ldots,d \\ 0 \;\; \text{otherwise} \end{cases}$$

- This uniform kernel function returns 1 if the sample is inside the hypercube of length 1 and 0 otherwise.

Use of the Hypercube and Kernel Function

- An equivalent way of defining the uniform kernel function is:

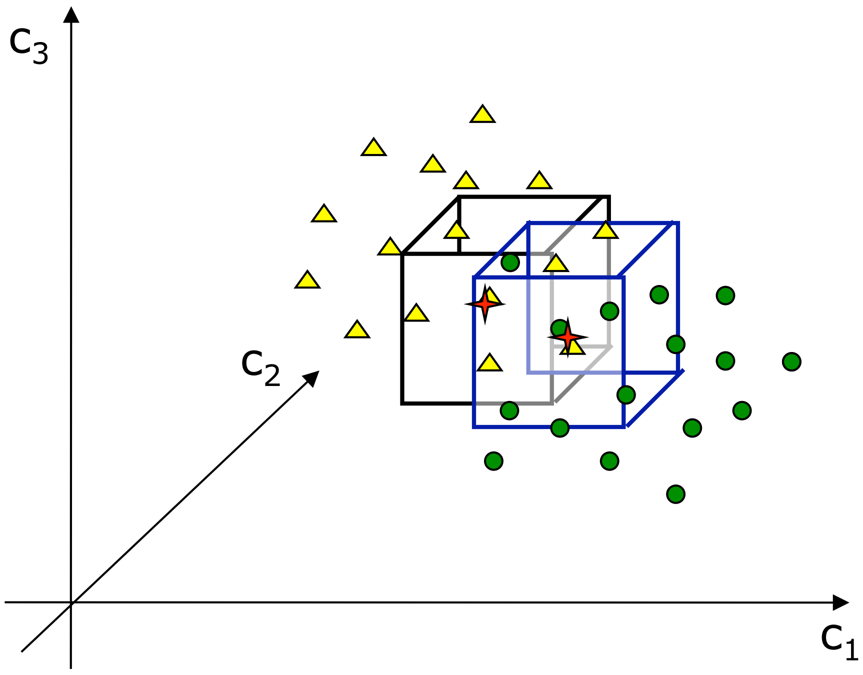
$$H\left(\frac{\|\vec{c}-\vec{c}_i\|}{h}\right)=\begin{cases}1 & \text{if } \|\vec{c}-\vec{c}_i\| < \left(h/2\right) \\ 0 & \text{otherwise}\end{cases}$$

- Use: Given a new feature vector $\vec{c}$, we center the hypercube at $\vec{c}$ and examine how many of the feature vectors in our training set fall inside the hypercube.
- The number of features that fall in a hypercube around $\vec{c}$ is:

$$K=\sum_{i=1}^{N} H\left(\|\vec{c}-\vec{c}_i\|/h\right)$$

Hypercubes in Feature Space

- Hypercubes can overlap.
- It depends on the data.



Density Estimation Using Kernel Functions

- Our goal is to estimate $p(\vec{c}) = {}^{K}\!/\!_{NV}$.
- We know *N*, the number of our training samples.
- We know *V*, $V = h^d$.
- We can use the kernel function to compute *K*:

$$K = \sum_{i=1}^{N} H\left(\left\|\vec{c} - \vec{c}_i\right\| / h\right)$$

- Thus, we can estimate the pdf as follows:

$$p(\vec{c}) = \frac{\sum_{i=1}^{N} H\left(\left\|\vec{c} - \vec{c}_i\right\| / h\right)}{Nh^d}$$

Kernel Functions

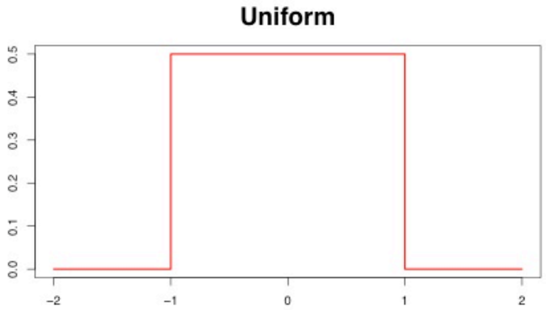
- Like the histogram, the uniform kernel function also has discontinuities.
- Thus, in practice other kernel functions are used that result in a smoother estimated density.
- For example, a Gaussian (a.k.a. normal) kernel is commonly used:

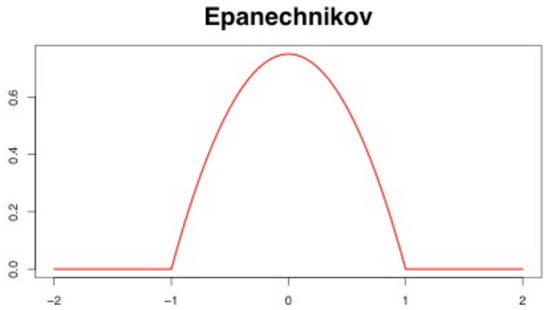
$$H\left(\frac{\left\|\vec{c}-\vec{c}_i\right\|}{h}\right)=\frac{1}{\sqrt{2\pi}}e^{-\frac{\left(\left\|\vec{c}-\vec{c}_i\right\|/h\right)^2}{2}}$$

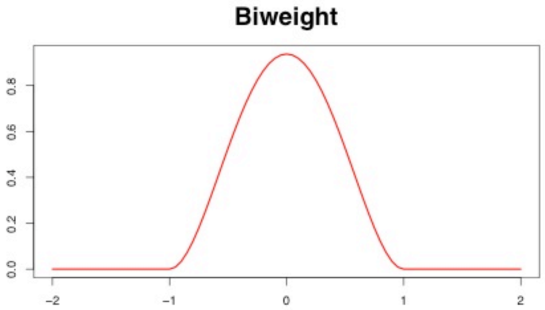
- Another widely used kernel is the biweight or quartic:

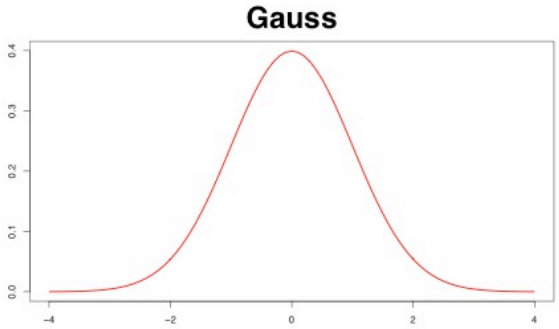
$$H\left(\frac{\left\|\vec{c}-\vec{c}_i\right\|}{h}\right)=\begin{cases}\frac{15}{16}\left(1-\left(\left\|\vec{c}-\vec{c}_i\right\|/h\right)^2\right)^2 & \text{if } \left(\left\|\vec{c}-\vec{c}_i\right\|/h\right)<1\\ 0 & \text{otherwise}\end{cases}$$

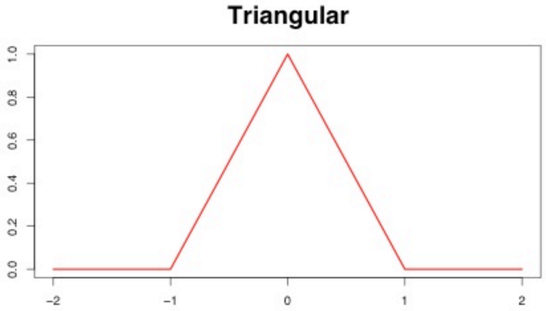
Plots of Different Kernel Functions











Density Estimation – Parzen Windows

- The number of samples *K*, that fall inside the hypercube is a function of $K(V)$.
- Thus, for Parzen windows we have:

$$p(\vec{c}) = \frac{K(V)}{NV}$$

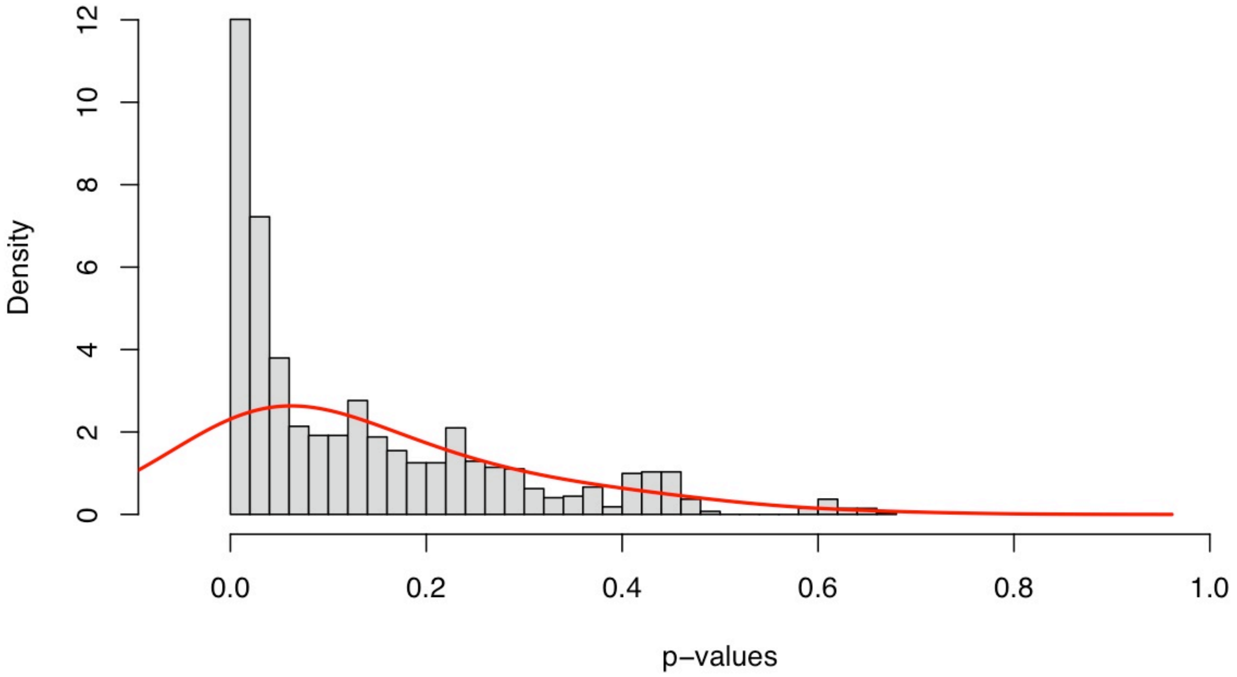
- Different hypercube sizes values will give different pdf estimates.
- A feature vector $\vec{c}_{new}$ will be recognized as belonging to the class that gets the majority vote in the hypercube centered at $\vec{c}_{new}$.
- Parzen windows are a general tool for estimating probability density functions from discrete samples.

Remarks on Kernel Density Estimation

- Computing K using a kernel function involves all samples in the training set. Thus, obtaining a pdf estimate can become a costly operation, especially as *N* becomes very very large (a desirable property).
- Kernel based density estimation is basically a superposition of (smeared) hypercubes.
- The bins are not predefined (as in the case of histograms), but depend on data.
- As in K-nearest neighbor the entire training data must be available at classification time.
- As in histogram the width of the bins can affect the resulting pdf estimates.

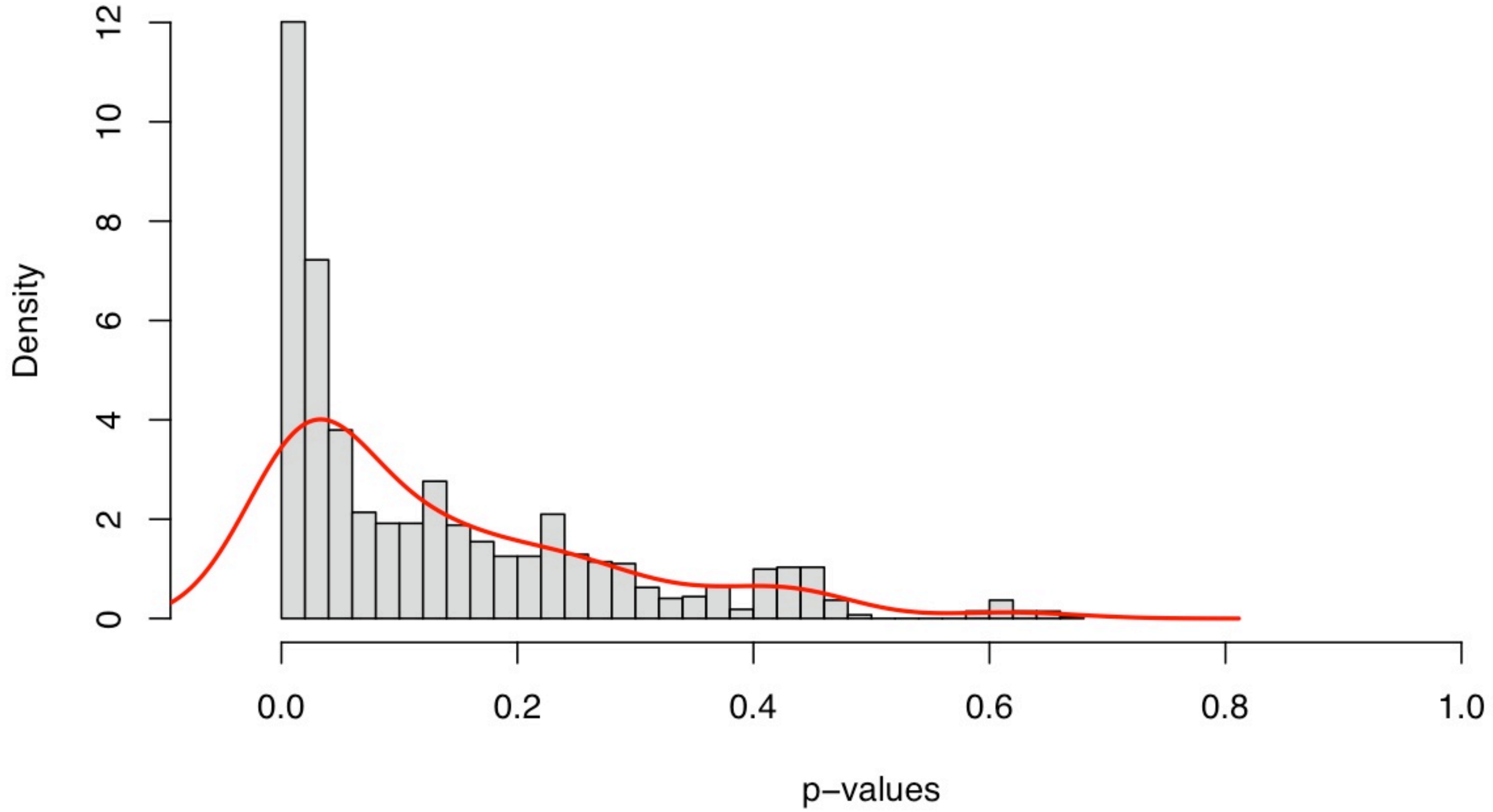
Hypecube Size

- The width of the hypercube *h* directly controls the smoothness of the resulting pdf.



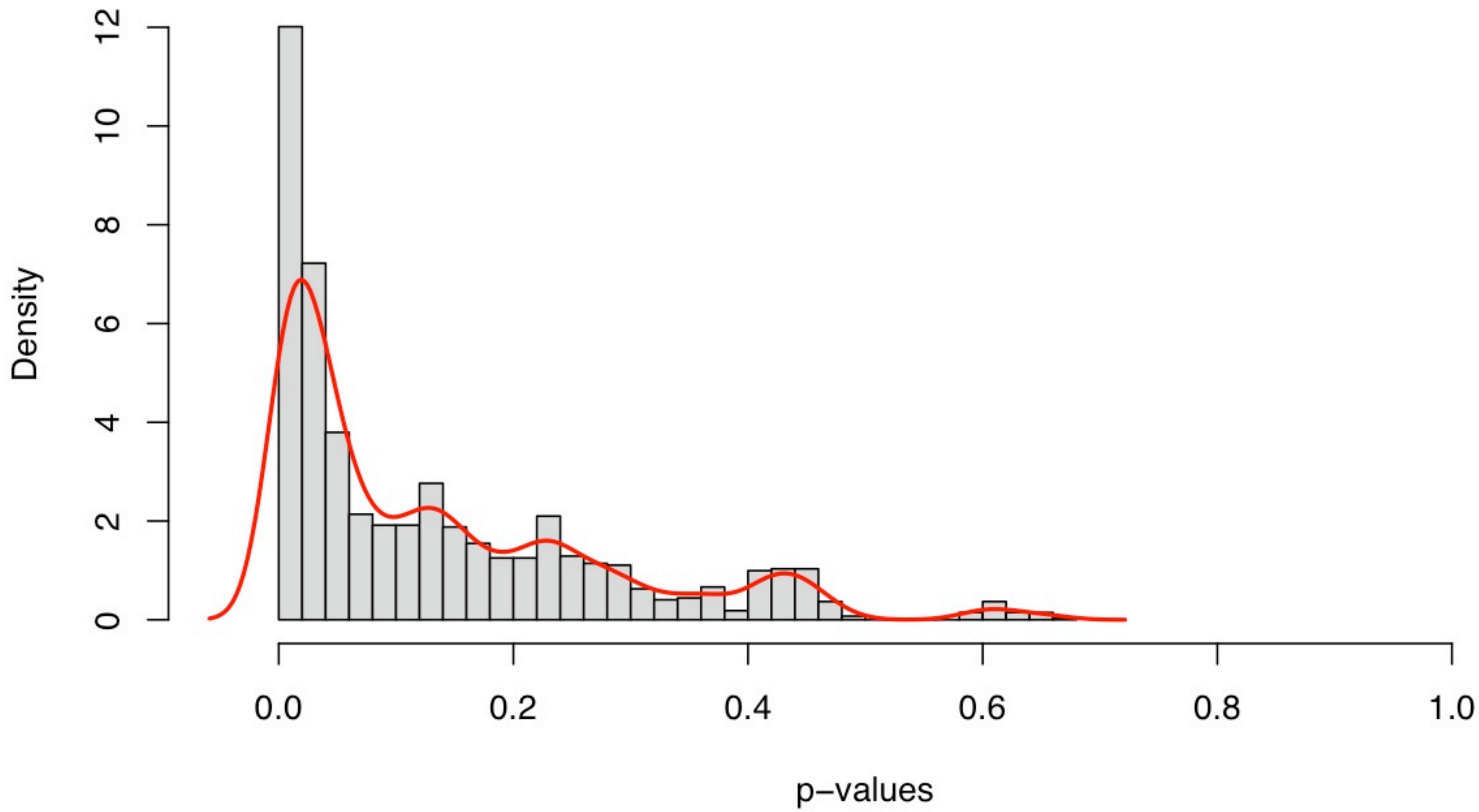
- A low *h*, in this case *h*=0.1 can result in underfitting or oversmoothing.

Hypecube Size - oversmoothing



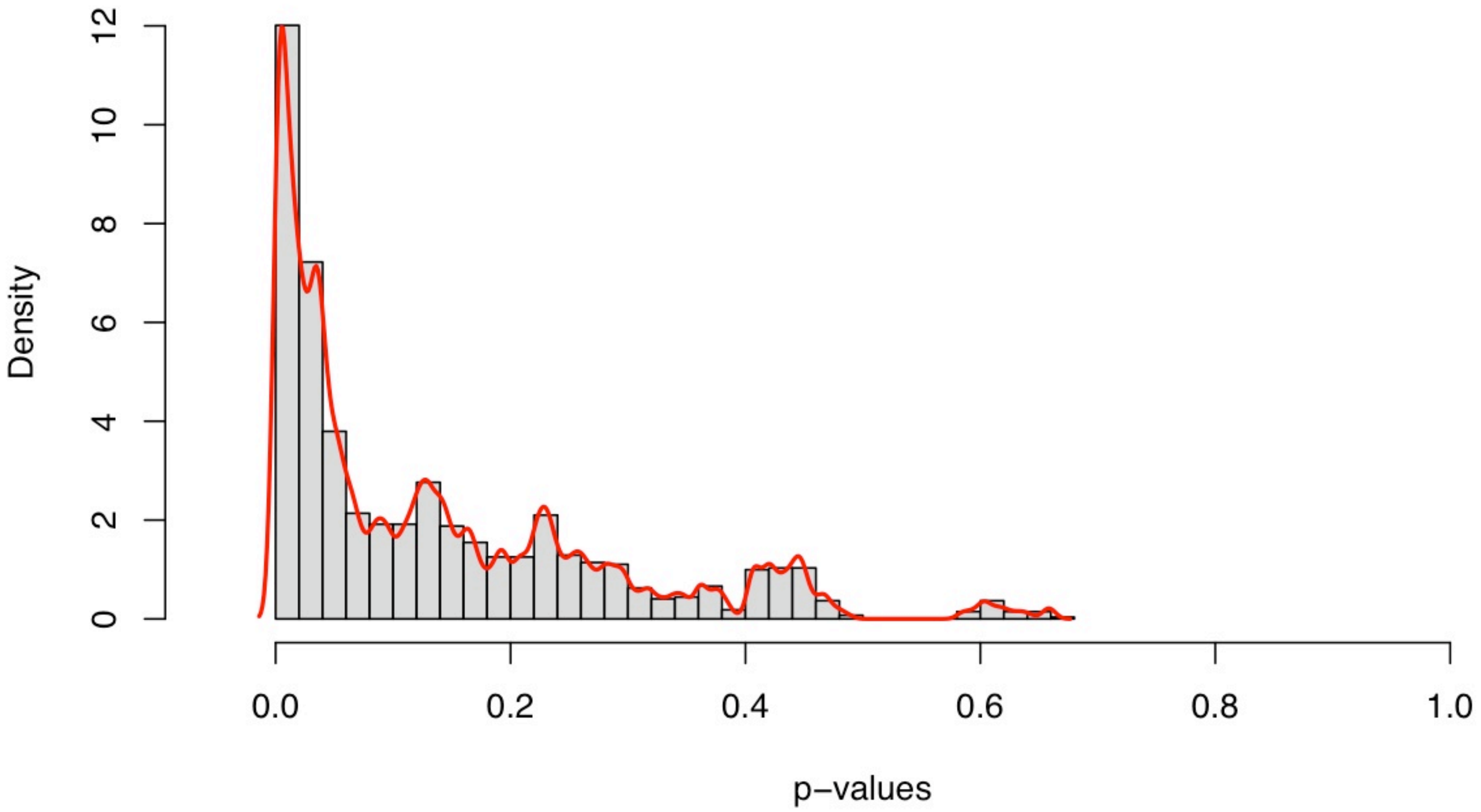
- As *h* decreases, in this case *h*=0.05 the amount of smoothing decreases.

Hypecube Size – reasonable smoothing



- As *h* decreases further, in this case *h*=0.02 the approximation better captures the attributes of the sample data.

Hypecube Size – overfitting



- As *h* decreases even further, in this case *h*=0.005 the approximation ends up overfitting the sample data.

References

1. The kernel density estimation plots and the effect of the hypecube width are adapted from the presentation of S. Scheid, “Introduction to Kernel Smoothing”, http://compdiag.molgen.mpg.de/docs/talk_05_01_04_stefanie.pdf