Non parametric methods

Course of Machine Learning
Master Degree in Computer Science
University of Rome “Tor Vergata”

Giorgio Gambosi

a.a. 2018-2019

Probability distribution estimates

- The statistical approach to classification requires the (at least approximate) knowledge of $p(\mathcal{C}_i|\mathbf{x})$: in fact, an item $\mathbf{x}$ shall be assigned to the class $\mathcal{C}_i$ such that

$$i = \underset{k}{\operatorname{argmax}}\, p(\mathcal{C}_k|\mathbf{x})$$

- The same holds in the regression case, where $p(y|\mathbf{x})$ has to be estimated.

Probability distribution estimates: hypotheses

What do we assume to know of class distributions, given a training set $\mathbf{X}, \mathbf{t}$?

- Case 1. The probabilities $p(\mathbf{x}|\mathcal{C}_i)$ are known: an item is assigned $\mathbf{x}$ to the class $\mathcal{C}_i$ such that
$$i = \underset{j}{\operatorname{argmax}}\, p(\mathcal{C}_j|\mathbf{x})$$
where $p(\mathcal{C}_j|\mathbf{x})$ can be derived through Bayes' rule and prior probabilities, since $p(\mathcal{C}_k)|\mathbf{x}) \propto p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)$

Probability distribution estimates: hypotheses

- Case 2. The type of probability distribution $p(\mathbf{x}|\boldsymbol{\theta})$ is known: an estimate of parameter values $\boldsymbol{\theta}_i$ is performed for all classes, taking into account for each class $\mathcal{C}_i$ the subset of $\mathbf{X}_i, \mathbf{t}_i$ of items belonging to the class, that is such that $t = i$. Different approaches to parameter estimation:
  1. Maximum likelihood: $\boldsymbol{\theta}_i^{ML} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, p(\mathbf{X}_i, \mathbf{t}_i|\boldsymbol{\theta})$ is computed. Item $\mathbf{x}$ is assigned to class $\mathcal{C}_i$ if
  $$i = \underset{j}{\operatorname{argmax}}\, p(\mathcal{C}_j|\mathbf{x}) = \underset{j}{\operatorname{argmax}}\, p(\mathbf{x}|\boldsymbol{\theta}_j^{ML})p(\mathcal{C}_j)$$
  2. Maximum a posteriori: $\boldsymbol{\theta}_i^{MAP} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, p(\boldsymbol{\theta}|\mathbf{X}_i, \mathbf{t}_i)$ is computed. Item $\mathbf{x}$ is assigned to class $\mathcal{C}_i$ if
  $$i = \underset{j}{\operatorname{argmax}}\, p(\mathcal{C}_j|\mathbf{x}) = \underset{j}{\operatorname{argmin}}\, p(\mathbf{x}|\boldsymbol{\theta}_j^{MAP})p(\mathcal{C}_j)$$
  3. Bayesian estimate: the distributions $p(\boldsymbol{\theta}|\mathbf{X}_i, \mathbf{t}_i)$ are estimated for each class and, from them,
  $$p(\mathbf{x}|\mathcal{C}_i) = \int_{\boldsymbol{\theta}} p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{X}_i, \mathbf{t}_i)d\boldsymbol{\theta}$$
  Item $\mathbf{x}$ is assigned to class $\mathcal{C}_i$ if
  $$\begin{aligned} i &= \underset{j}{\operatorname{argmax}}\, p(\mathcal{C}_j|\mathbf{x}) = \underset{j}{\operatorname{argmax}}\, p(\mathcal{C}_j)p(\mathbf{x}|\mathcal{C}_j) \\ &= \underset{j}{\operatorname{argmax}}\, p(\mathcal{C}_j)\int_{\boldsymbol{\theta}} p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{X}_j, \mathbf{t}_j)d\boldsymbol{\theta} \end{aligned}$$

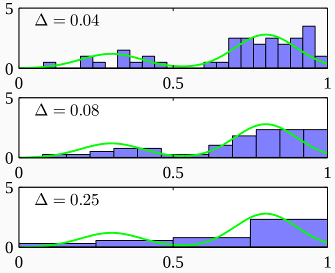
Probability distribution estimates: hypotheses

- Case 3. No knowledge of the probabilities assumed.
- The class distributions $p(\mathbf{x}|\mathcal{C}_i)$ are directly from data.
- In previous cases, use of (parametric) models for a synthetic description of data in $\mathbf{X}, \mathbf{t}$
- In this case, no models (and parameters): training set items explicitly appear in class distribution estimates.
- Denoted as non parametric models: indeed, an unbounded number of parameters is used

Histograms

- Elementary type of non parametric estimate
- Domain partitioned into m d-dimensional intervals (bins)
- The probability $P_{\mathbf{x}}$ that an item belongs to the bin containing item $\mathbf{x}$ is estimated as $\frac{n(\mathbf{x})}{n}$, where $n(\mathbf{x})$ is the number of element in that bin
- The probability density in the interval corresponding to the bin containing $\mathbf{x}$ is then estimated as the ratio between the above probability and the interval width $\Delta(\mathbf{x})$ (tipically, a constant Δ)

$$p_H(\mathbf{x}) = \frac{\frac{n(\mathbf{x})}{N}}{\Delta(\mathbf{x})} = \frac{n(\mathbf{x})}{N\Delta(\mathbf{x})}$$



Histograms: problems

- The density is a function of the position of the first bin. In the case of multivariate data, also from bin orientation.
- The resulting estimates is not continuous.
- Curse of dimensionality: the number of bins grows as a polynomial of order d: in high-dimensional spaces many bins may result empty, unless a large number of items is available.
- In practice, histograms can be applied only in low-dimensional datasets (1,2)

Kernel density estimators

- Probability that an item is in region $\mathcal{R}(\mathbf{x})$, containing $\mathbf{x}$

$$P_{\mathbf{x}} = \int_{\mathcal{R}(\mathbf{x})} p(\mathbf{z}) d\mathbf{z}$$

- Given n items $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n$, the probability that k among them are in $\mathcal{R}(\mathbf{x})$ is given by the binomial distribution

$$p(k) = \binom{n}{k} P_{\mathbf{x}}^K (1 - P_{\mathbf{x}})^{n-k} = \frac{n!}{k!(n-k)!} P_{\mathbf{x}}^K (1 - P_{\mathbf{x}})^{n-k}$$

- Mean and variance of the ratio $r = \dfrac{k}{n}$ are

$$E[r] = P_{\mathbf{x}} \qquad \text{var}[r] = \frac{P_{\mathbf{x}}(1 - P_{\mathbf{x}})}{n}$$

- $P_{\mathbf{x}}$ is the expected fraction of items in $\mathcal{R}(\mathbf{x})$, and the ratio r is an estimate. As $n \to \infty$ variance decreases and r tends to $E[r] = P_{\mathbf{x}}$. Hence, in general,

$$r = \frac{k}{n} \simeq P(\mathbf{x})$$

Nonparametric estimates

- Let the volume of $\mathcal{R}(\mathbf{x})$ be sufficiently small. Then, the density $p(\mathbf{x})$ is almost constant in the region and

$$P_{\mathbf{x}} = \int_{\mathcal{R}(\mathbf{x})} p(\mathbf{z})d\mathbf{z} \simeq p(\mathbf{x})V$$

  where V is the volume of $\mathcal{R}(\mathbf{x})$
- since $P_{\mathbf{x}} \simeq \frac{k}{n}$, it then derives that $p(\mathbf{x}) \simeq \frac{k}{nV}$

Approaches to nonparametric estimates

Two alternative ways to exploit the estimate $p(\mathbf{x}) \simeq \dfrac{k}{nV}$

1. Fix V and derive k from data (kernel density estimation)
2. Fix k and derive V from data (K-nearest neighbor).

It can be shown that in both cases, under suitable conditions, the estimator tends to the true density $p(\mathbf{x})$ as $n \to \infty$.

Kernel density estimation: Parzen windows

- Region associated to a point $\mathbf{x}$: hypercube with edge length h (and volume h^d) centered on $\mathbf{x}$.
- Kernel function $k(\mathbf{u})$ (Parzen window) used to count the number of items in the unit hypercube centered on the origin $\mathbf{0}$

$$k(\mathbf{u}) = \begin{cases} 1 & |u_i| \leq 1/2 \qquad\qquad i = 1, \dots, d \\ 0 & \text{otherwise} \end{cases}$$

- as a consequence, $k\left(\dfrac{\mathbf{x}-\mathbf{x}'}{h}\right) = 1$ iff $\mathbf{x}'$ is in the hypercube of edge length h centered on $\mathbf{x}$
- the number of items in the hypercube is then

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

Kernel density estimation: Parzen windows

- The estimated density is

$$p(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h^d}k\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right)$$

- Since

$$k(\mathbf{u}) \geq 0 \qquad \text{and} \qquad \int k(\mathbf{u})d\mathbf{u} = 1$$

  it derives

$$k\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right) \geq 0 \qquad \text{and} \qquad \int k\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right)d\mathbf{x} = h^d$$

Kernel density estimation: Parzen windows

As a consequence, it results that $p_n(\mathbf{x})$ is a probability density. In fact,

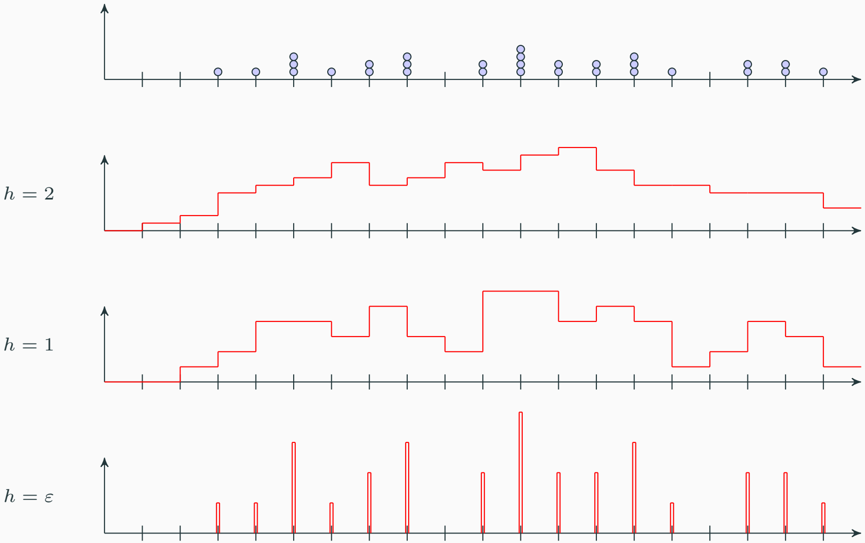
$$p(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h^d}k\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right) \geq 0$$

and

$$\begin{aligned}\int p(\mathbf{x})d\mathbf{x} = & \int \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h^d}k\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right)d\mathbf{x} \\ & \frac{1}{nh^d}\int\sum_{i=1}^{n}k\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right)d\mathbf{x} \\ & \frac{1}{nh^d}\sum_{i=1}^{n}\int k\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right)d\mathbf{x} = \frac{1}{nh^d}nh^d = 1\end{aligned}$$

Clearly, the window size has a relevant effect on the estimate

Kernel density estimation: Parzen windows



Kernels and smoothing

Drawbacks

1. discontinuity of the estimates
2. items in a region centered on $\mathbf{x}$ have uniform weights: their distance from $\mathbf{x}$ is not taken into account

Solution. Use of smooth kernel functions $\kappa_h(u)$ to assign larger weights to points nearer to the origin.

Assumed characteristics of $\kappa_h(u)$:

$$\int \kappa_h(\mathbf{x})d\mathbf{x} = 1$$

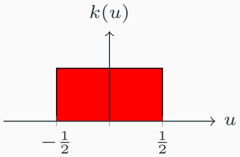
$$\int \mathbf{x}\kappa_h(\mathbf{x})d\mathbf{x} = 0$$

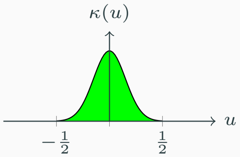
$$\int \mathbf{x}^2\kappa_h(\mathbf{x})d\mathbf{x} > 0$$

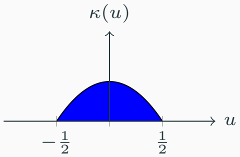
Kernels and smoothing

Usually kernels are based on smooth radial functions (functions of the distance from the origin)

1. gaussian $\kappa(u) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\frac{u^2}{\sigma^2}}$, unlimited support
2. Epanechnikov $\kappa(u) = 3\left(\frac{1}{2} - u^2\right)$, $|u| \leq \frac{1}{2}$, limited support
3. $\cdots$





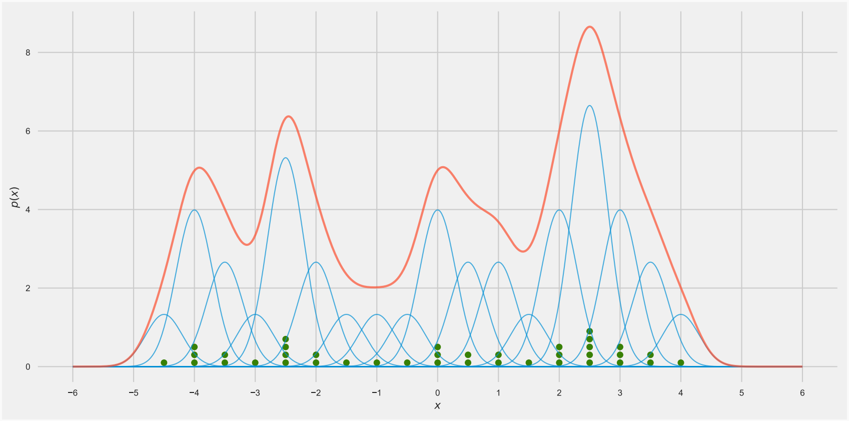


resulting estimate:

$$p(\mathbf{x}) = \frac{1}{nh}\sum_{i=1}^{n} \kappa\left(\frac{\mathbf{x}-\mathbf{x}_i}{h}\right) = \frac{1}{n}\sum_{i=1}^{n} \kappa_h\left(\mathbf{x}-\mathbf{x}_i\right)$$

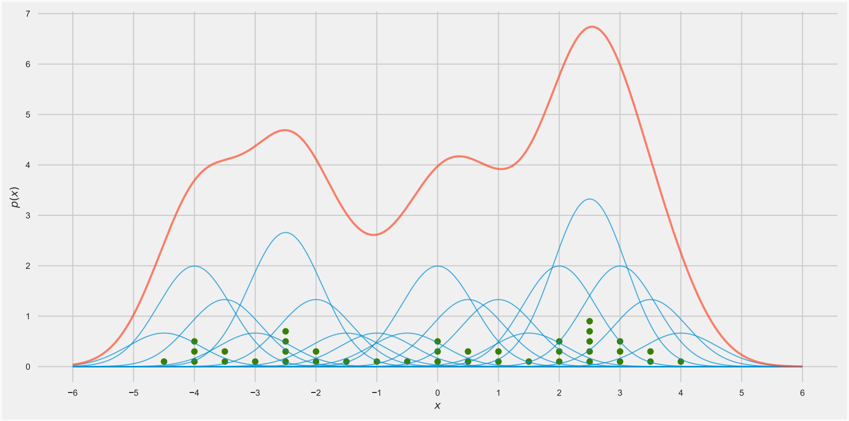
Kernels and smoothing

$h = 1$



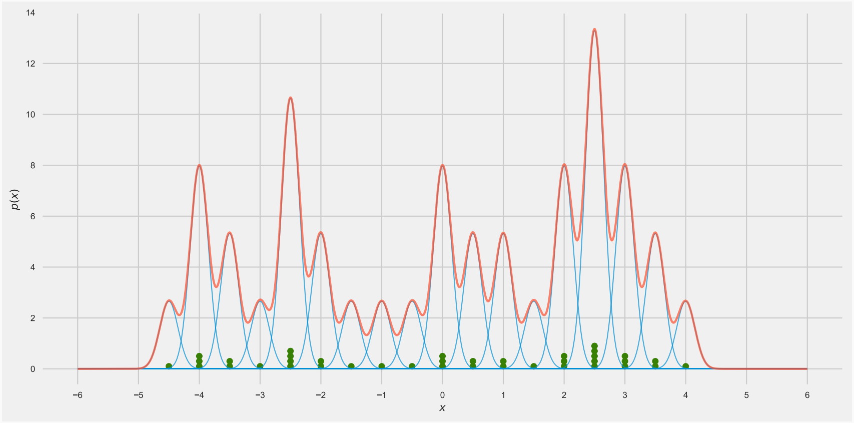
Kernels and smoothing

$h = 2$



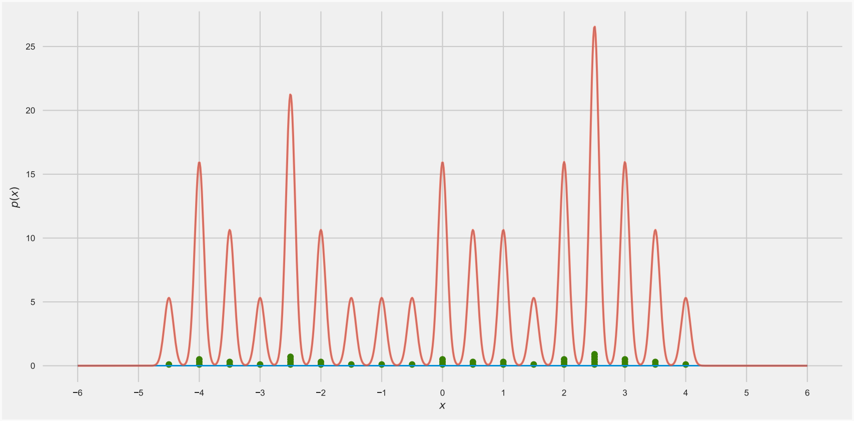
Kernels and smoothing

$h = .5$



Kernels e smoothing

$h = .25$



Kernels regression

Kernel smoothers methods can be applied also to regression: in this case, the value corresponding to any item $\mathbf{x}$ is predicted by referring to items in the training (and in particular to the items nearer to $\mathbf{x}$).

In this case, the conditional expectation

$$f(\mathbf{x}) = E[y|\mathbf{x}] = \int yp(y|\mathbf{x})dy = \int y\frac{p(\mathbf{x},y)}{p(\mathbf{x})}dy = \frac{\int yp(\mathbf{x},y)dy}{p(\mathbf{x})} = \frac{\int yp(\mathbf{x},y)dy}{\int p(\mathbf{x},y)dy}$$

should be returned.

Applying kernels, we have

$$p(\mathbf{x},y) \approx \frac{1}{n}\sum_{i=1}^{n}\kappa_h(\mathbf{x}-\mathbf{x}_i)\kappa_h(y-t_i)$$

Kernels regression

This results into

$$f(\mathbf{x}) = \frac{\int y \frac{1}{n} \sum_{i=1}^{n} \kappa_h(\mathbf{x}-\mathbf{x}_i)\kappa_h(y-t_i)dy}{\int \frac{1}{n} \sum_{i=1}^{n} \kappa_h(\mathbf{x}-\mathbf{x}_i)\kappa_h(y-t_i)dy} = \frac{\frac{1}{n} \sum_{i=1}^{n} \kappa_h(\mathbf{x}-\mathbf{x}_i) \int y\kappa_h(y-t_i)dy}{\sum_{i=1}^{n} \kappa_h(\mathbf{x}-\mathbf{x}_i) \int \kappa_h(y-t_i)dy}$$

and, since $\displaystyle\int \kappa_h(y-t_i)dy = 1$ and $\displaystyle\int y\kappa_h(y-t_i)dy = y_i$, we get

$$f(\mathbf{x}) = \frac{\sum_{i=1}^{n} \kappa_h(\mathbf{x}-\mathbf{x}_i)t_i}{\frac{1}{n} \sum_{i=1}^{n} \kappa_h(\mathbf{x}-\mathbf{x}_i)}$$

Kernels regression

By setting

$$w_i(\mathbf{x}) = \frac{\kappa_h(\mathbf{x} - \mathbf{x}_i)}{\sum_{j=1}^{n} \kappa_h(\mathbf{x} - \mathbf{x}_j)}$$

we can write

$$f(\mathbf{x}) = \sum_{i=1}^{n} w_i(\mathbf{x}) t_i$$

that is, the predicted value is computed as a linear of all target values, weighted by kernels (Nadaraya-Watson)

Locally weighted regression

In Nadaraya-Watson model, the prediction is performed by means of a weighted combination of constant values (target values in the training set).

Locally weighted regression improves that approach by referring to a weighted version of the sum of squared differences loss function used in regression.

If a value y has to be predicted for a provided item $\mathbf{x}$, a “local” version of the loss function is considered, with weight $\mathbf{w}_i$ dependent from the “distance” between $\mathbf{x}$ and $\mathbf{x}_i$.

$$L(\mathbf{x}) = \sum_{i=1}^{n} \kappa_h(\mathbf{x} - \mathbf{x}_i)(\mathbf{w}^T \overline{\mathbf{x}}_i - t_i)^2$$

Locally weighted regression

An instance of the weighted regression problem must be solved,

$$\hat{\mathbf{w}}(\mathbf{x}) = \underset{\mathbf{w}}{\operatorname{argmin}} \sum_{i=1}^{n} \kappa_h(\mathbf{x} - \mathbf{x}_i)(\mathbf{w}^T\overline{\mathbf{x}}_i - t_i)^2$$

which has solution

$$\hat{\mathbf{w}}(\mathbf{x}) = (\overline{\mathbf{X}}^T\Psi(\mathbf{x})\overline{\mathbf{X}})^T\overline{\mathbf{X}}^T\Psi(\mathbf{x})\mathbf{t}$$

where $\Psi(\mathbf{x})$ is a diagonal $n \times n$ matrix with $\Phi(\mathbf{x})_{ii} = \kappa_h(\mathbf{x} - \mathbf{x}_i)$.

The prediction is then performed as usual, as

$$y = \hat{\mathbf{w}}(\mathbf{x})^T\overline{\mathbf{x}}$$

The same considerations can be done if polynomial regression applied

Density estimation through kNN

- The region around $\mathbf{x}$ is extended to include k items
- The estimated density is

$$p(\mathbf{x}) \simeq \frac{k}{nV} = \frac{k}{nc_d r_k^d(\mathbf{x})}$$

where:

  - c_d is the volume of the d-dimensional sphere of unitary radius
  - $r_k^d(\mathbf{x})$ is the distance from $\mathbf{x}$ to the k-th nearest item (the radius of the smallest sphere with center $\mathbf{x}$ containing k items)

Classification through kNN

- To classify $\mathbf{x}_i$, let us consider a hypersphere of volume V with center $\mathbf{x}$ containing k items from the training set
- Let k_i be the number of such items belonging to class $\mathcal{C}_i$. Then, the following approximation holds:

$$p(\mathbf{x}|\mathcal{C}_i) = \frac{k_i}{n_i V}$$

  where n_i is the number of items in the training set belonging to class $\mathcal{C}_i$
- Similarly, for the evidence,

$$p(\mathbf{x}) = \frac{k}{nV}$$

- And, for the prior distribution,

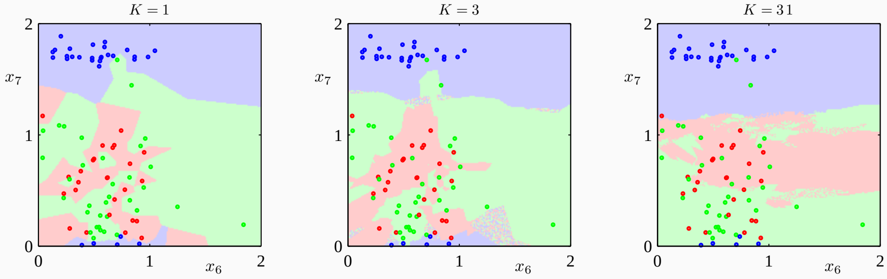
$$p(\mathcal{C}_i) = \frac{n_i}{n}$$

- The class posterior distribution is then

$$p(\mathcal{C}_i|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_i)p(\mathcal{C}_i)}{p(\mathbf{x})} = \frac{\frac{k_i}{n_i V} \cdot \frac{n_i}{n}}{\frac{k}{nV}} = \frac{k_i}{k}$$

Classification through kNN

- Simple rule: an item is classified on the basis of similarity to near training set items
- To classify $\mathbf{x}$, determine the k items in the training nearest to it and assign $\mathbf{x}$ to the majority class among them
- A metric is necessary to measure similarity.



Classification through kNN

- kNN is a simple classifier is simple and can work quite well, provided it is given a good distance metric and has enough labeled training data: it can be shown that it can result within a factor of 2 of the best possible performance as $n \to \infty$
- subject to the curse of dimensionality: due to the large sparseness of data at high dimensionality, items considered by kNN can be quite far away from the query point, and thus resulting in poor locality.

Local logistic regression

The same approach applied in the case of local regression can be applied for classification, by defining a weighted loss function to be minimized, with weights dependent from the item whose target must be predicted.

In this case, a weighted version of the cross entropy function is considered, which has to be maximized

$$L(\mathbf{x}) = \sum_{i=1}^{n} \kappa_h(\mathbf{x} - \mathbf{x}_i)(t_i \log p_i - (1 - t_i) \log(1 - p_i))$$

with $p_i = \sigma(\mathbf{w}^T \overline{\mathbf{x}}_i)$, as usual.

A suitable modification of the IRLS algorithm for logistic regression can be applied here to compute

$$\hat{\mathbf{w}}(\mathbf{x}) = \underset{\mathbf{w}}{\operatorname{argmax}} \sum_{i=1}^{n} \kappa_h(\mathbf{x} - \mathbf{x}_i)(t_i \log p_i - (1 - t_i) \log(1 - p_i))$$