

Nonparametric Methods for Density Estimation

Nearest Neighbour Classification

Lecturer:
Ondřej Drbohlav

Authors:
Ondřej Drbohlav, Jiří Matas

Centre for Machine Perception
Czech Technical University, Prague
<http://cmp.felk.cvut.cz>

Last update: Oct 2022



Probability Density Estimation

Parametric Methods for Density Estimation

- ◆ Have been dealt with in the previous lecture
- ◆ Advantage: Low number of parameters to estimate
- ◆ Disadvantage: The resulting estimated density can be arbitrarily wrong if the underlying distribution does not agree with the assumed parametric model.

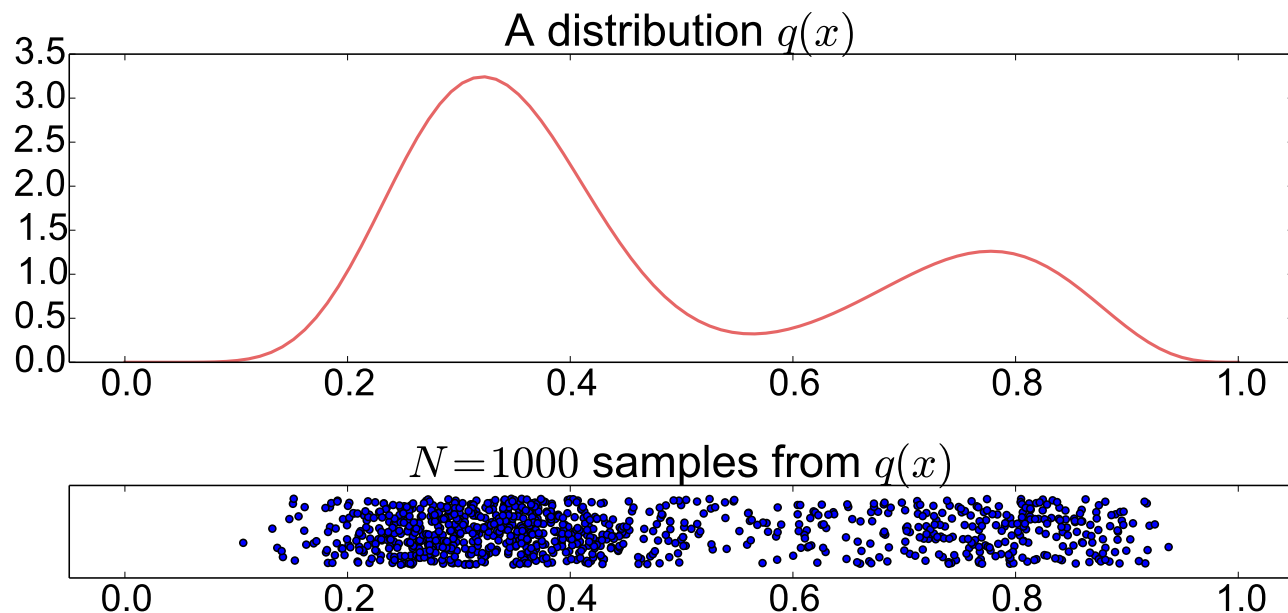
Non-Parametric Methods for Density Estimation

- ◆ Histogram
- ◆ Nearest Neighbor approach

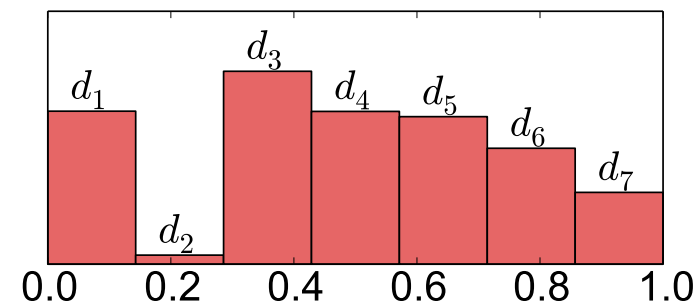
Histogram as piecewise constant density estimate:

Task formulation

Consider the following distribution $q(x)$ on the interval $[0, 1]$, and i.i.d. sampling from it. We will fit the distribution by a 'histogram' with B bins. More precisely, we will estimate a piecewise-constant function on the interval $[0, 1]$ with B segments of the same width. For a given B , the parameters of this piecewise-constant function are the heights d_1, d_2, \dots, d_B of the individual bins. This function is denoted $p(x|\{d_1, d_2, \dots, d_B\})$.



$p(x|\{d_1, d_2, \dots, d_B\})$ to be estimated



For the given number of bins B , d_1, d_2, \dots, d_B must conform to the constraint that the area under the function must sum up to one,

$$1 = \int_{-\infty}^{\infty} p(x|\{d_1, d_2, \dots, d_B\}) dx = \sum_{i=1}^B \int_{\frac{i-1}{B}}^{\frac{i}{B}} d_i dx = \sum_{i=1}^B d_i \overset{\text{bin width}}{\downarrow} w = \sum_{i=1}^B \frac{d_i}{B}. \quad (1)$$

Histogram as piecewise constant density estimate:

Finding d_i 's using Maximum Likelihood

Let us estimate $\{d_i, i = 1, 2, \dots, B\}$ by Maximum Likelihood (ML) approach. Let N_i denote the number of samples which belong the i -th bin (thus clearly, $\sum_{i=1}^B N_i = N$). The likelihood $L(\mathcal{T})$ of observing the samples $\mathcal{T} = \{x_1, x_2, \dots, x_N\}$ given the parameters $\boldsymbol{\theta} = \{d_1, d_2, \dots, d_B\}$ is

$$L(\mathcal{T}) = p(\mathcal{T}|\boldsymbol{\theta}) = \prod_{i=1}^N p(x_i|\boldsymbol{\theta}) = \prod_{j=1}^B \overbrace{\left(\prod_{k=1}^{N_j} d_j \right)}^{\text{points in } j\text{-th bin}} = \prod_{j=1}^B d_j^{N_j}. \quad (2)$$

The maximization task is then

$$\ell(\mathcal{T}) = \sum_{j=1}^B N_j \log d_j \rightarrow \max, \quad \text{subject to } \frac{1}{B} \sum_{j=1}^B d_j = 1, \quad (3)$$

where maximization has been formulated using the log-likelihood $\ell(\mathcal{T})$. The Lagrangian of the optimization task and the conditions of optimality (using the derivative $\partial/\partial d_k$) are then:

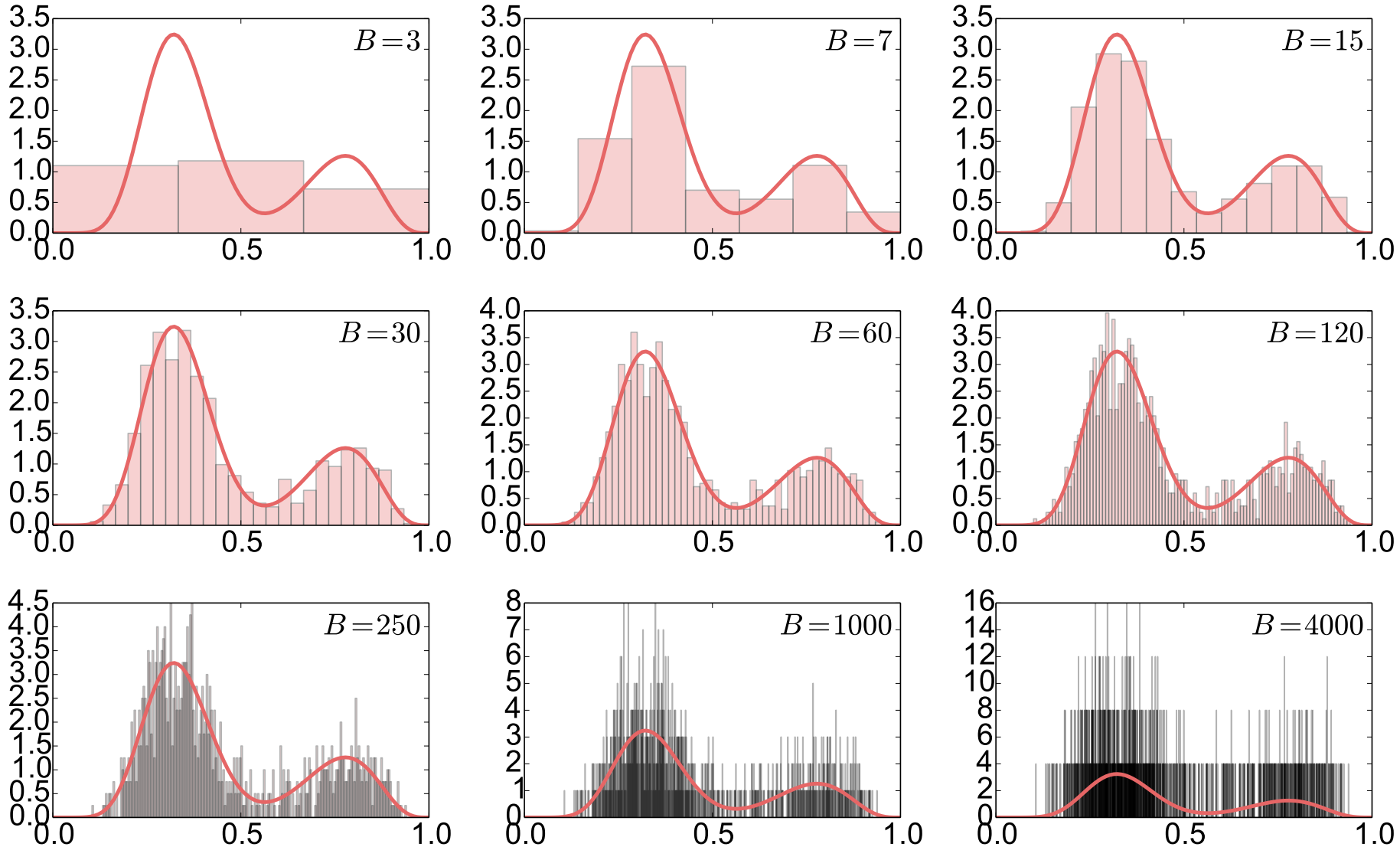
$$\text{Lagrangian: } \sum_{j=1}^B N_j \log d_j + \lambda \left(\frac{1}{B} \sum_{j=1}^B d_j - 1 \right) \quad (4)$$

$$\frac{N_k}{d_k} + \frac{\lambda}{B} = 0 \Rightarrow \frac{d_k}{N_k} = \text{const.} \Rightarrow d_k = B \frac{N_k}{N}. \quad (5)$$

Histogram as piecewise constant density estimate: Example, different number of bins

$$d_k = B \frac{N_k}{N} \quad (6)$$

This result is in line with the common use of histograms for approximating pdf's. Results for different B 's:



Histogram as piecewise constant density estimate: What number of bins produces closest pdf approximation?

Let us measure the differences between the (actual) source distribution $q(x)$ and the piecewise-constant density estimate $p(x) = p(x|\{d_1, d_2, \dots, d_B\})$ from the $N = 1000$ samples, using B bins.

Measures used:

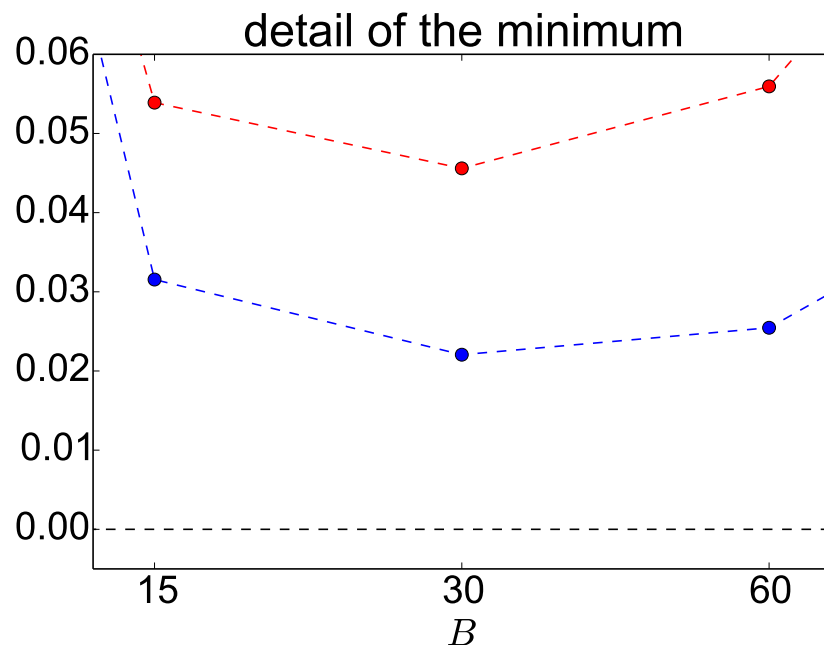
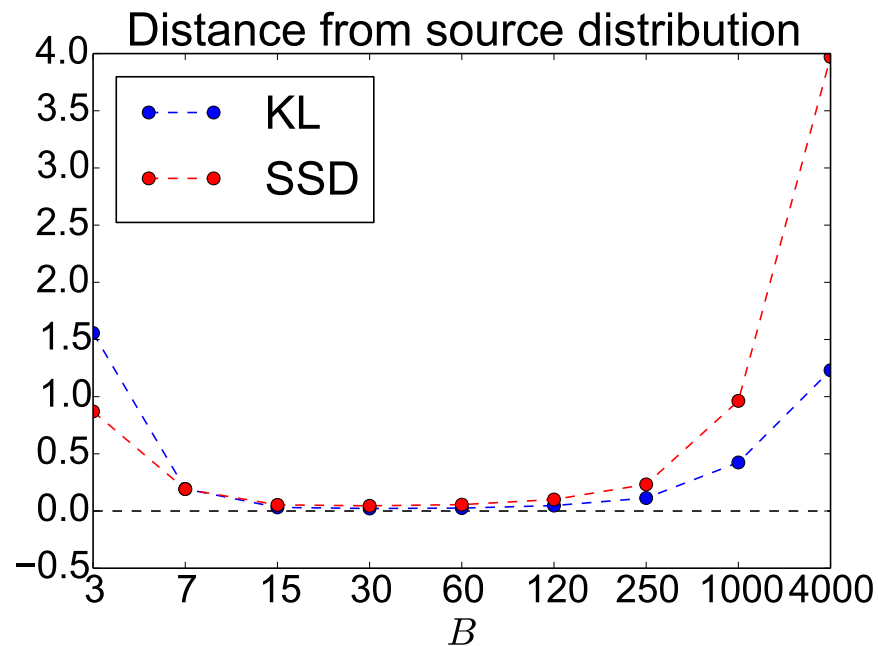
Kullback-Leibler divergence D_{KL} :

$$D_{\text{KL}}(p||q) = \int_{-\infty}^{\infty} p(x) \log \frac{p(x)}{q(x)} dx . \quad (7)$$

(Note that KL div. is not a metric.)

Sum of squared differences D_{SSD} :

$$D_{\text{SSD}}(p, q) = \int_{-\infty}^{\infty} (p(x) - q(x))^2 dx . \quad (8)$$



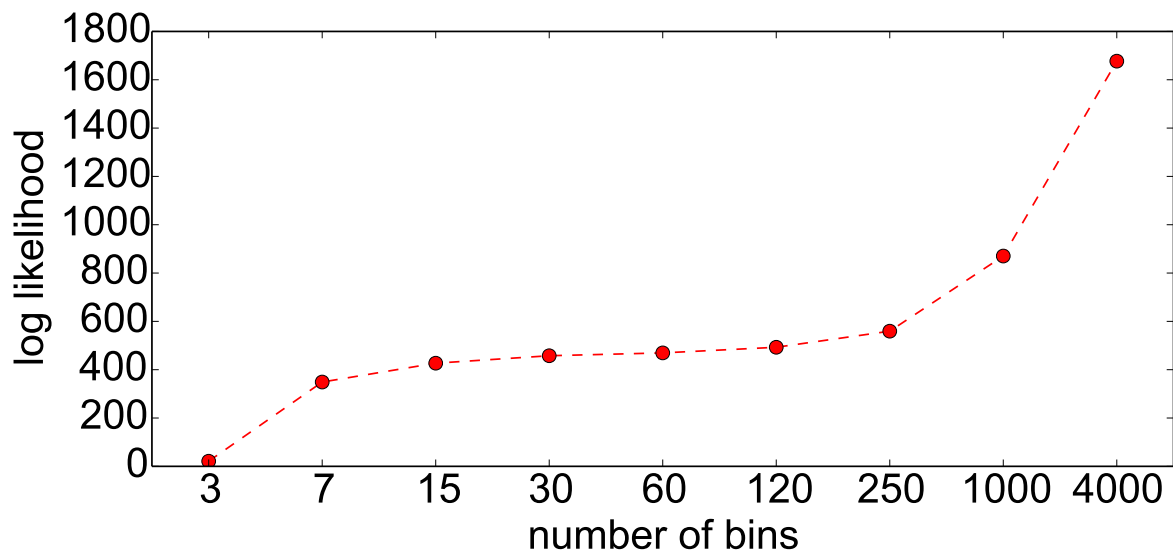
Histogram as piecewise constant density estimate: Choosing the number of bins B by ML

How can we find the optimal number of bins B ? Let us try to employ the ML approach again: find the B which maximizes the likelihood. Recall that:

parameters d_j : $d_j = B \frac{N_j}{N}$ (ML estimate) (9)

likelihood $L(\mathcal{T})$: $L(\mathcal{T}) = p(\mathcal{T} | \{d_1, d_2, \dots, d_B\}) = \prod_{j=1}^B d_j^{N_j} = \prod_{j=1}^B \left(\frac{BN_j}{N} \right)^{N_j}$ (10)

log-likelihood $\ell(\mathcal{T})$: $\ell(\mathcal{T}) = \sum_{j=1}^B N_j \log d_j = \sum_{j=1}^B N_j \log \frac{BN_j}{N}$ (11)



For $B = 4000$, the log-likelihood ℓ is the highest. But the pdf estimate with this B is poor, and very different from the source distribution as measured by D_{KL} or D_{SSD} . For $B = 10^5$, $\ell(\mathcal{T}) \sim 4600$. *What went wrong?*

Histogram, choosing the number of bins B :

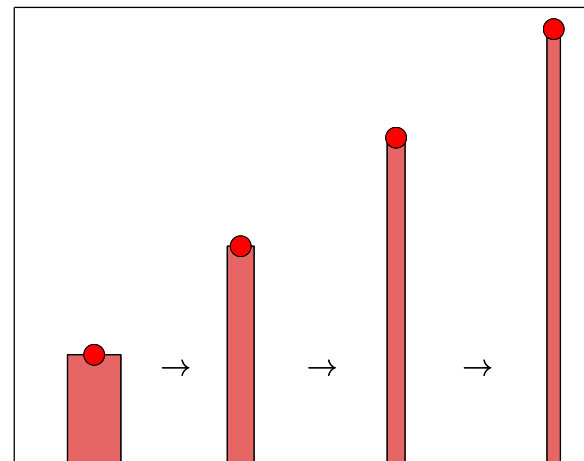
ML overfits and produces $B = \infty$

When B grows, eventually it will reach a number \hat{B} such that there is either *no* or *one* point in every bin (assuming no two points in the data are identical), and this will stay true for any $B > \hat{B}$.

In such cases,

$$d_j = \begin{cases} \frac{B}{N} & \text{if the bin is populated by a point,} \\ 0 & \text{if the bin is not populated.} \end{cases} \quad (12)$$

As the number of bins B grows, the widths of occupied bins get narrower and the heights d_j 's higher. If $B \rightarrow \infty$ then also $d_j \rightarrow \infty$ for the occupied bins, and therefore also $\ell(\mathcal{T}) \rightarrow \infty$. Thus, such an approach cannot produce a “reasonable” answer to choosing B , as the solution it provides is $B = \infty$.



The problem is that the log-likelihood ℓ is computed using the same data used for fitting the model (computing d_i 's). This is a similar concept to training a classifier on certain data and testing on the same data, which is prone to over-fitting and poor generalization.

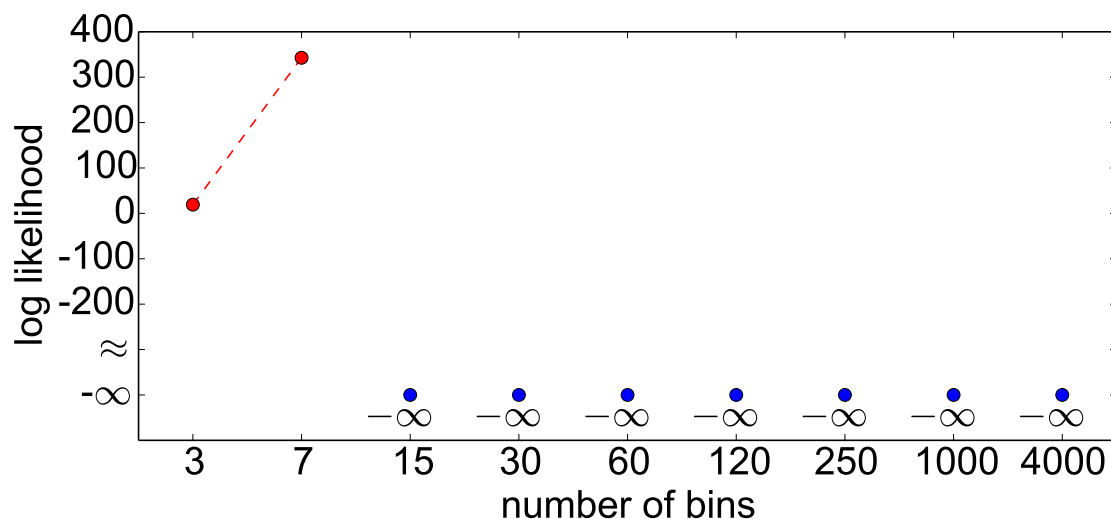
Histogram, choosing the number of bins B : Employing cross-validation

Let us compute the log likelihood using the following procedure: remove a given point from the dataset for computing d_i 's and evaluate its contribution to the log-likelihood. Do this for all the points. This approach is related to cross-validation technique (leave-one-out) for choosing parameters of a classifier.

Let the point in question belong to the j -th bin. The ML estimate for d_j , after removing this point from the dataset, is

$$d_j = B \frac{N_j - 1}{N - 1}, \quad (N_j \geq 1), \quad (13)$$

where the subtractions of 1 reflect the fact that the considered point is not used for estimating d_j . Computing the log likelihood ℓ this way produces the following result:



$$\ell = \sum_{\substack{j=1 \\ N_j \geq 1}}^B N_j \log d_j,$$

$$\text{with } d_j = B \frac{N_j - 1}{N - 1}$$

The 'failure' for $B > 7$ is caused by singly-occupied bins ($N_j = 1$) for which the modified ML estimate for d_j becomes zero. This will be fixed by using different estimates for d_j 's.

Histogram, choosing the number of bins B :

More suitable estimates for d_j 's

The problem of d_i being estimated as 0 is similar to the one encountered previously: Recall the example of tossing a coin three times, always getting *heads* ($\mathcal{T} = \{H, H, H\}$). The ML estimate is a fully unfair coin (probability of getting *heads* is 1, $\pi_{\text{head}} = 1$), thus making the likelihood of any sequence containing *tails* zero. We have seen before that employing the prior for the parameters to be estimated can mitigate this problem.

A (conjugate) prior for the histogram bin counts is the Dirichlet Distribution, with the pdf $p(d_1, d_2, \dots, d_B | \alpha_1, \alpha_2, \dots, \alpha_B) \sim \prod d_i^{\alpha_i - 1}$.

MAP Estimate:

$$d_i = B \frac{N_i + \alpha_i - 1}{N + \sum_{i=1}^B \alpha_i - B} \quad (14)$$

Bayes Estimate:

$$d_i = B \frac{N_i + \alpha_i}{N + \sum_{i=1}^B \alpha_i} \quad (15)$$

Interpretation: The parameters α_i 's can be interpreted as 'virtual' observations, as if α_k points have already been assigned to the k -th bin.

Example: The Bayes estimate using $\alpha_i = 1$ for all $i = 1, 2, \dots, B$ is

$$d_i = B \frac{N_i + 1}{N + B}. \quad (16)$$

Using this estimate will enable us to make reasonable computation of likelihood for all B 's.

Histogram, choosing the number of bins B :

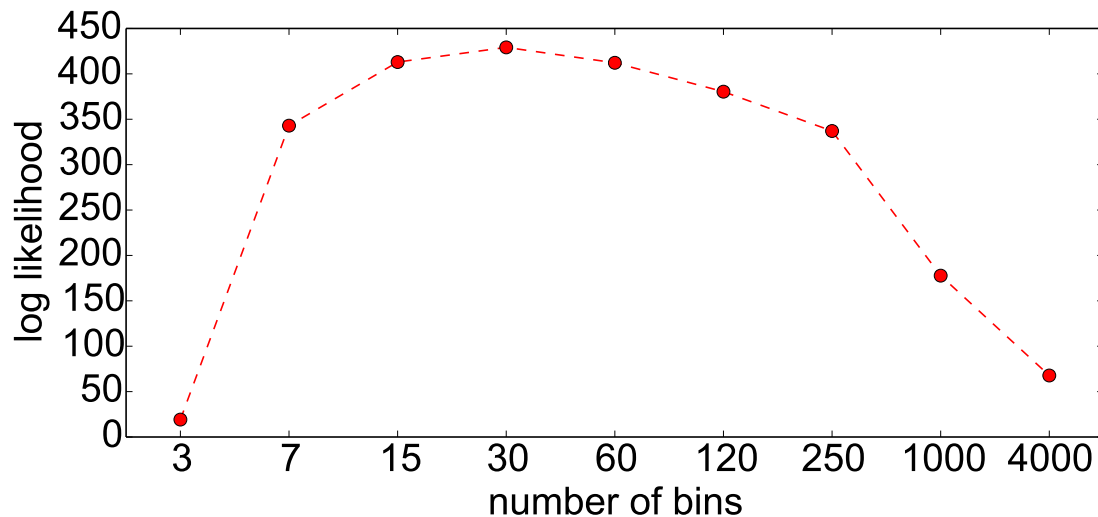
ML to find B , cross-validation, Bayes estimate for d_j 's

Let us now return to the previous task. Compute the log likelihood using the following procedure: remove a given point from the dataset for computing d_i 's and evaluate its contribution to the log-likelihood. Do this for all the points.

Use the Bayes estimate for d_j from the previous example, $d_j = B \frac{N_j + 1}{N + B}$. The modified estimation of d_j (omitting the point in question) will become

$$d_j = B \frac{N_j}{N - 1 + B}. \tag{17}$$

This leads to the following result:



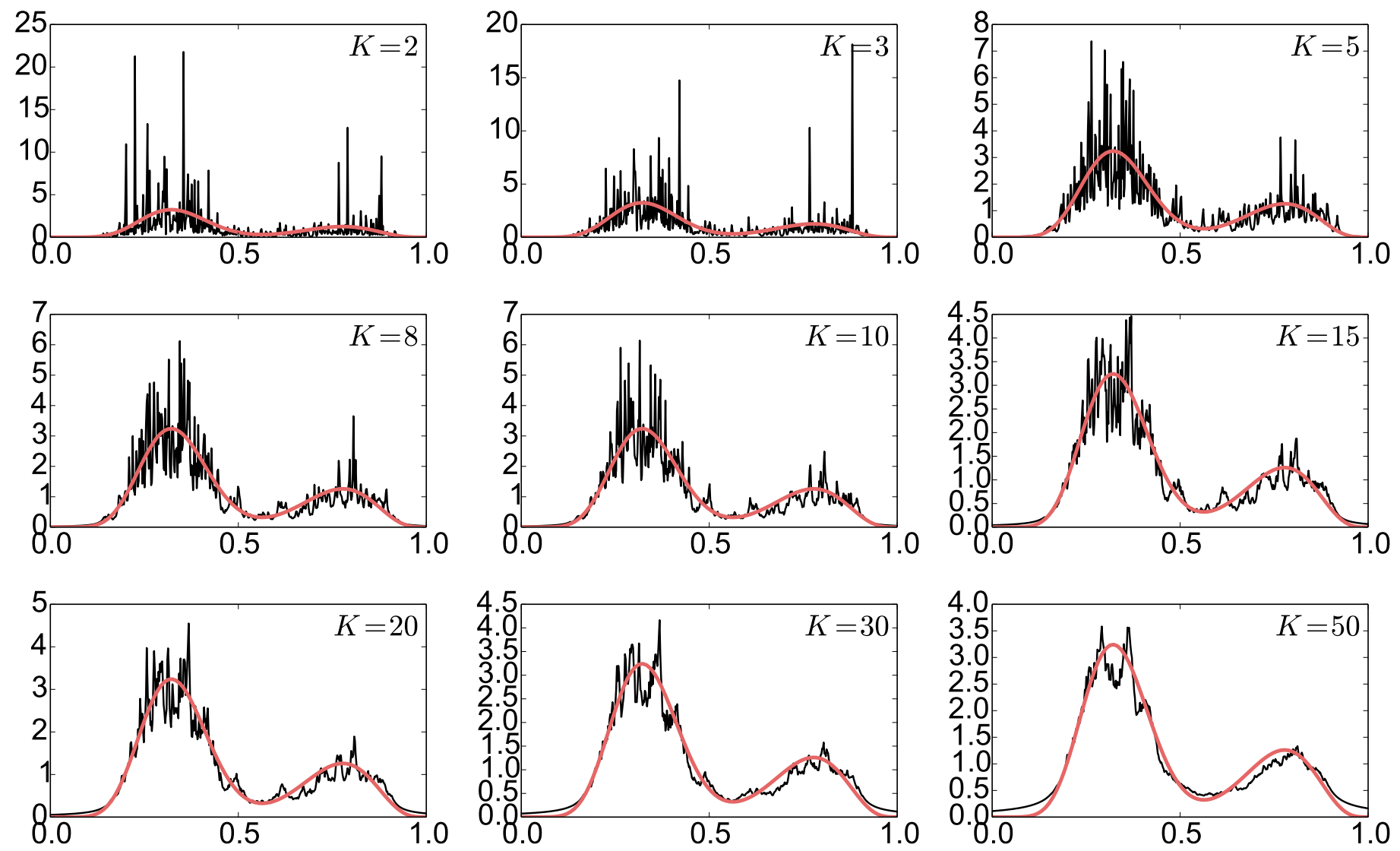
$$\ell = \sum_{j=1}^B N_j \log d_j,$$

$$\text{with } d_j = B \frac{N_j}{N - 1 + B}$$

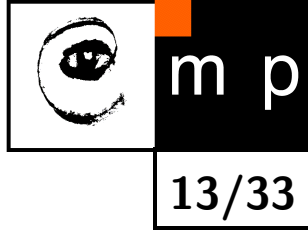
This result is in agreement with distribution differences as measured by D_{KL} or D_{SSD} . In particular, $B = 30$ is identified as the best-approximating number of bins.

K -Nearest Neighbor Approach to Density Estimation

Find K neighbors, the density estimate is then $p \sim 1/V$ where V is the volume of a minimum cell containing K NNs. Example ($p \sim$ inverse distance to K -th NN, same 1000 samples as before):



K -Nearest Neighbor Approach to Classification



Outline:

- ◆ Definition
- ◆ Properties
- ◆ Asymptotic error of NN classifier
- ◆ Error reduction by edit operation on the training class
- ◆ Fast NN search

K -NN Classification Definition

Assumption:

- ◆ Training set $\mathcal{T} = \{(x_1, k_1), (x_2, k_2), \dots, (x_N, k_N)\}$. There are R classes (letter K is reserved for K -NN in this lecture)
- ◆ A distance function $d : X \times X \mapsto \mathbb{R}_0^+$

Algorithm:

1. Given x , find K points $S = \{(x'_1, k'_1), (x'_2, k'_2), \dots, (x'_K, k'_K)\}$ from the training set \mathcal{T} which are closest to x in the metric d :

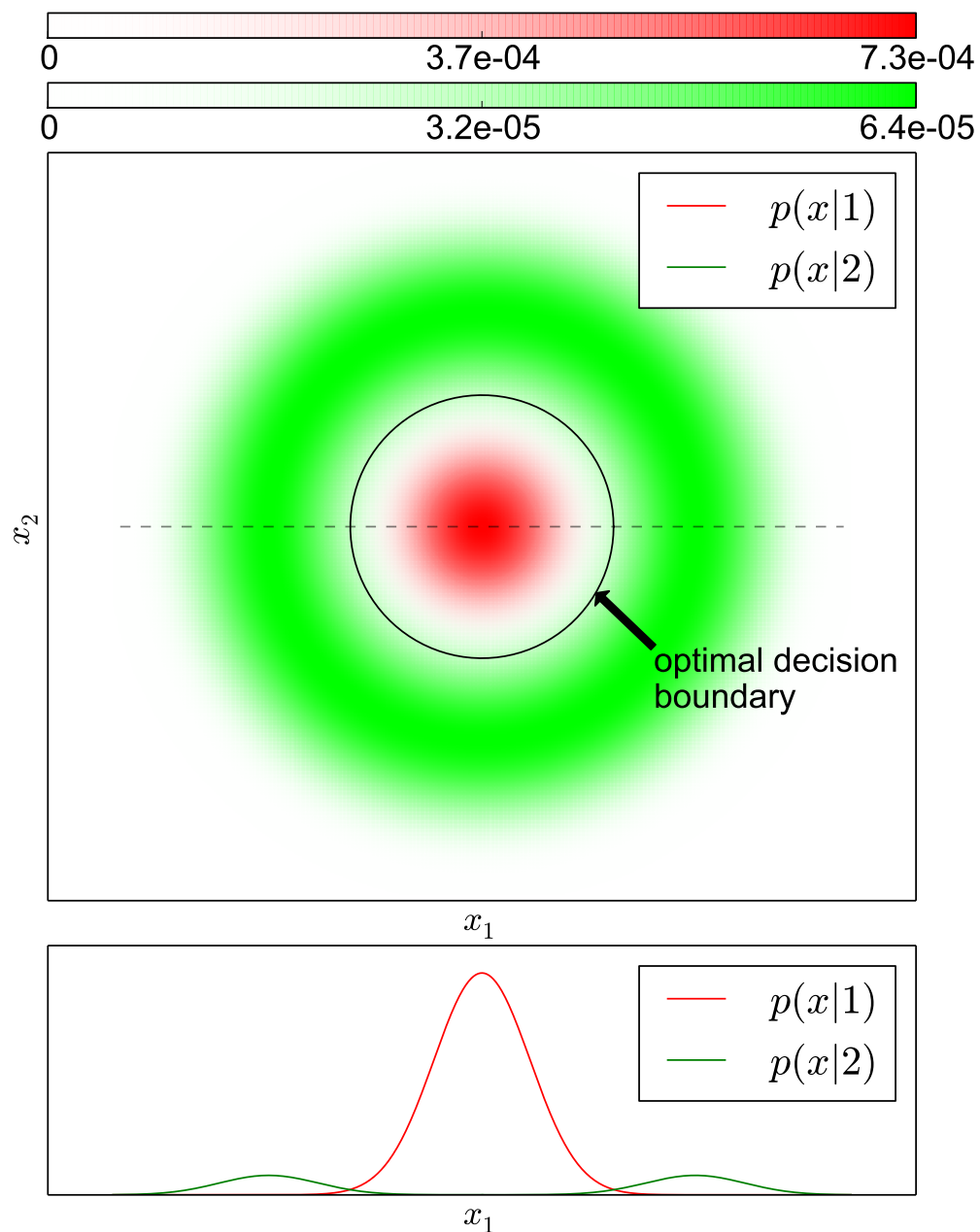
$$S = \{(x'_1, k'_1), (x'_2, k'_2), \dots, (x'_K, k'_K)\} \equiv \{(x_{r_1}, k_{r_1}), (x_{r_2}, k_{r_2}), \dots, (x_{r_K}, k_{r_K})\} \quad (18)$$

$$r_i: \text{the rank of } (x_i, k_i) \in \mathcal{T} \text{ as given by the ordering } d(x, x_i) \quad (19)$$

2. Classify x to the class k which has majority in S :

$$k = \operatorname{argmax}_{l \in R} \sum_{i=1}^K \mathbb{I}[k'_i = l] \quad (x'_i, k'_i) \in S \quad (20)$$

K-NN Example (1)



Consider the two distributions shown. The priors are assumed to be the same,

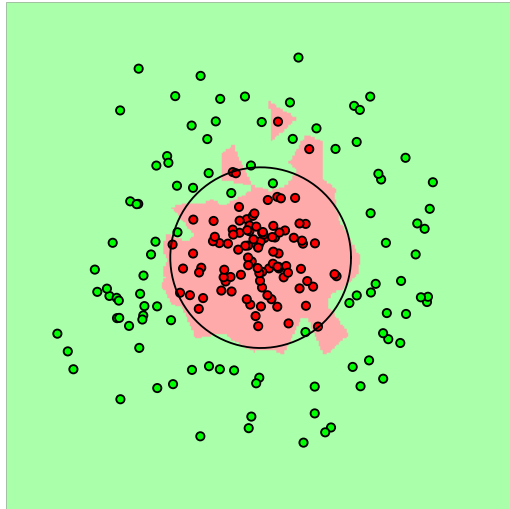
$$p(1) = p(2) = 0.5.$$

Bayesian optimal decision boundary is shown by the black circle.

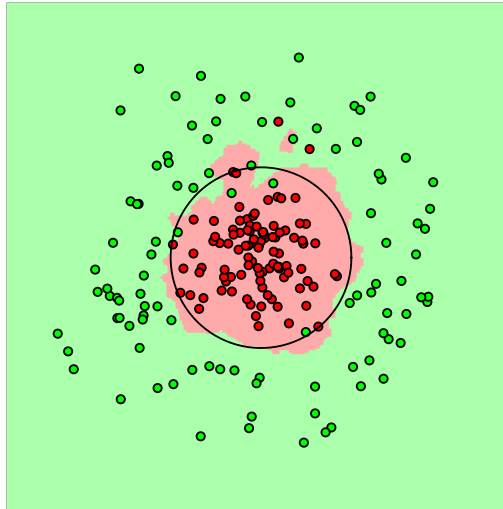
Bayesian error is $\epsilon_B = 0.026$.

K -NN Example (2)

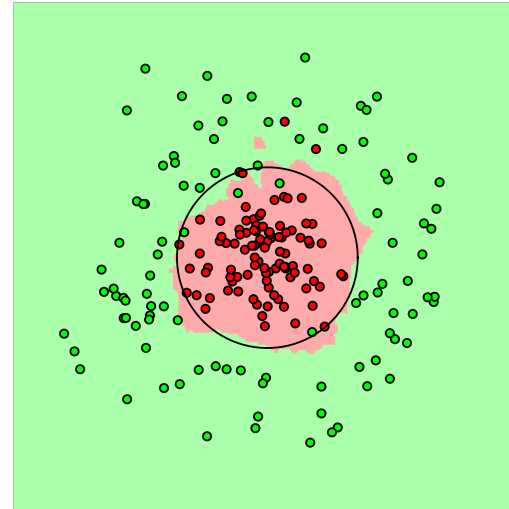
$K = 1$, error $\epsilon = 0.044$



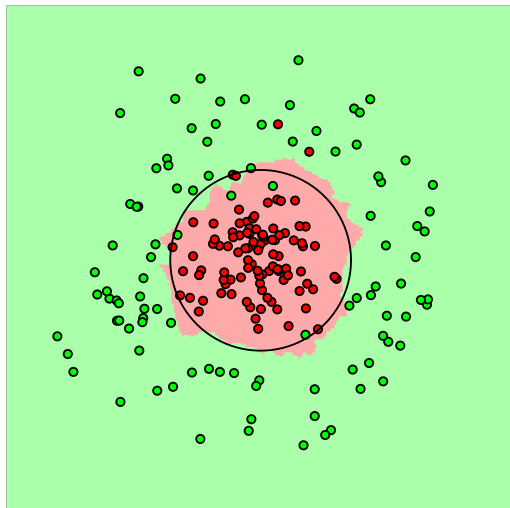
$K = 3$, error $\epsilon = 0.034$



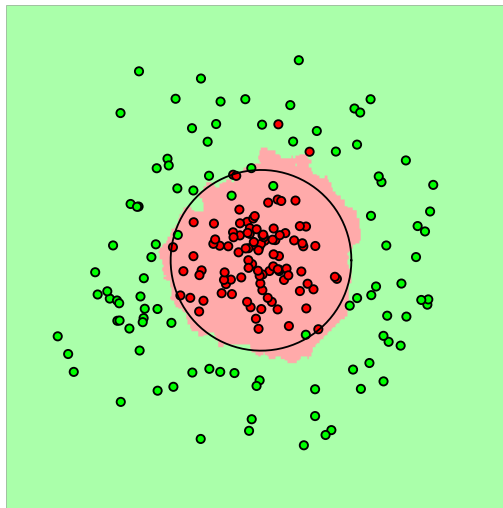
$K = 5$, error $\epsilon = 0.032$



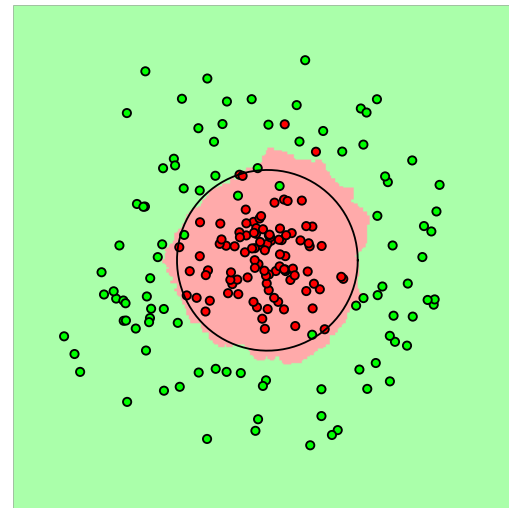
$K = 7$, error $\epsilon = 0.030$



$K = 9$, error $\epsilon = 0.031$

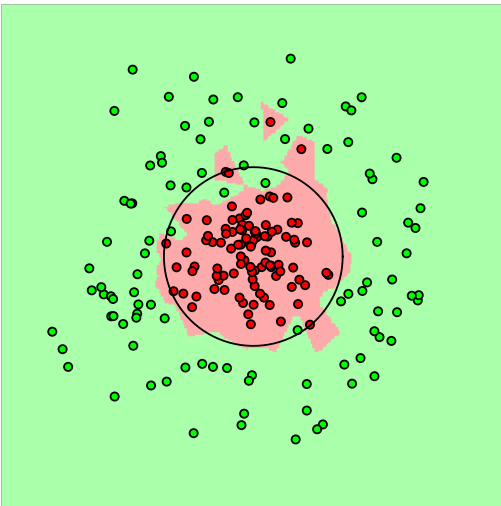
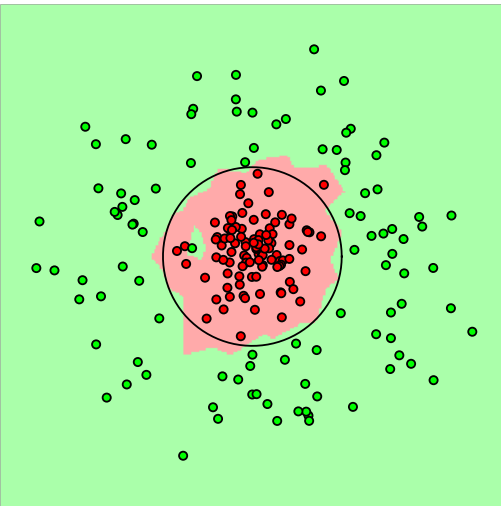
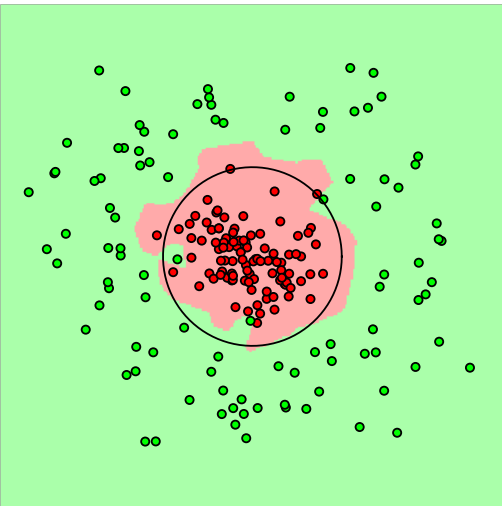
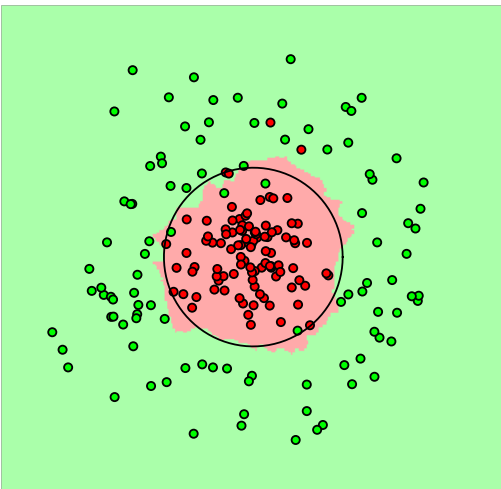
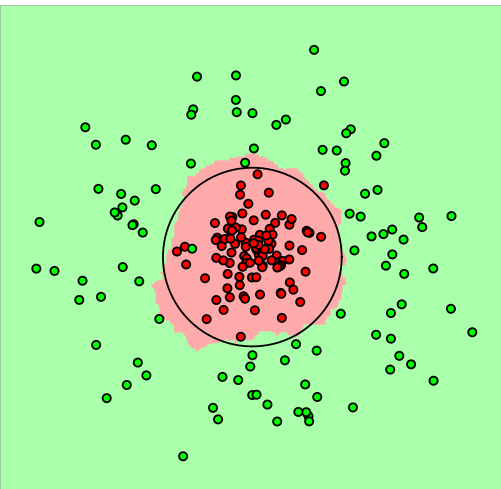
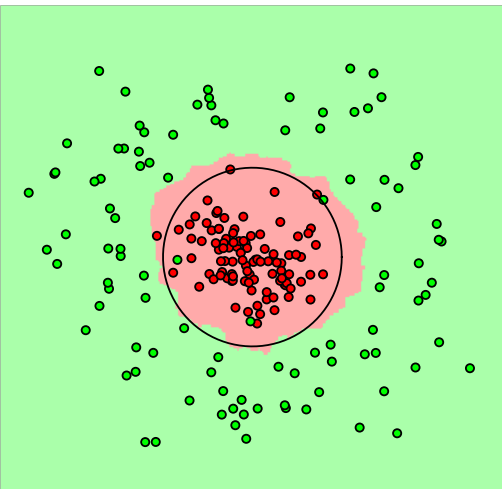


$K = 11$, error $\epsilon = 0.032$



$N = 100$ samples for each class. Bayes error $\epsilon_B = 0.026$.

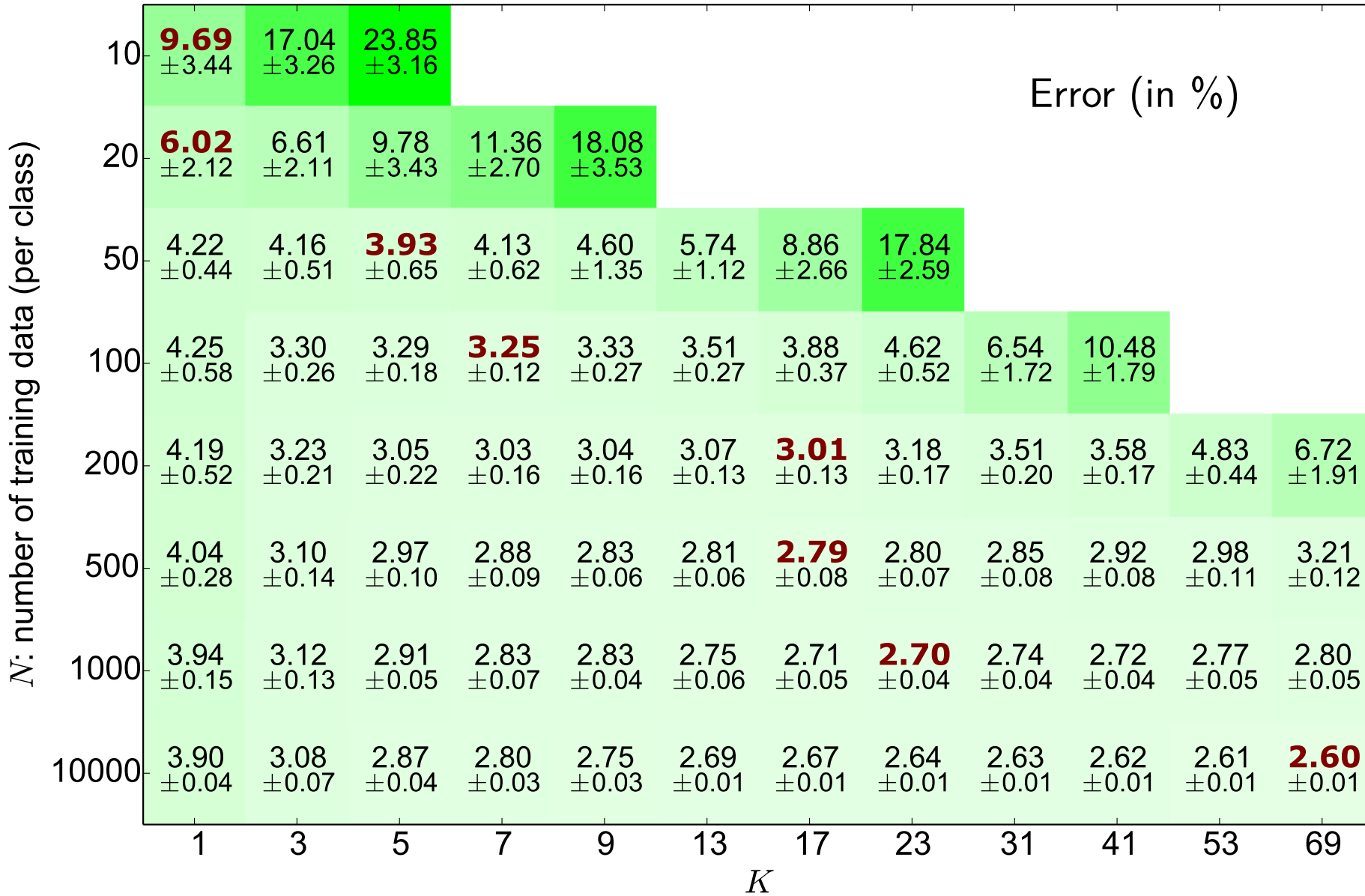
K-NN Example (3)

| \mathcal{T}_1 | \mathcal{T}_2 | \mathcal{T}_3 |
|--|---|--|
| $K = 1$, error $\epsilon = 0.044$ | $K = 1$, error $\epsilon = 0.038$ | $K = 1$, error $\epsilon = 0.043$ |
|  |  |  |
| $K = 7$, error $\epsilon = 0.030$ | $K = 7$, error $\epsilon = 0.031$ | $K = 7$, error $\epsilon = 0.036$ |
|  |  |  |

The results depend on the training set (result of a random process.)
 Each of the training sets \mathcal{T}_1 , \mathcal{T}_2 , \mathcal{T}_3 contain 100 points for each class.

K-NN Example (4)

K-NN error for different K and different sizes of the training set (N samples per class). 10 training sets have been generated randomly for each setting of K and N . Average error and its std is shown. Minimum average error is highlighted for each N . Bayes err. $\epsilon_B = 2.58\%$.



K-NN Properties

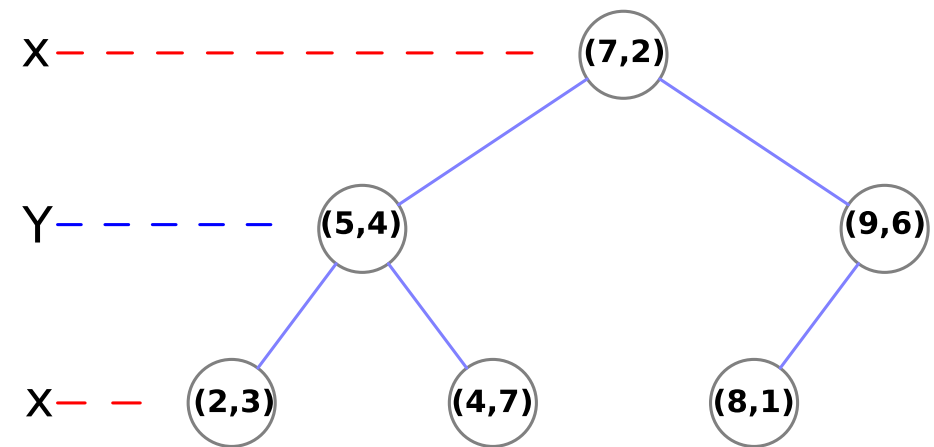
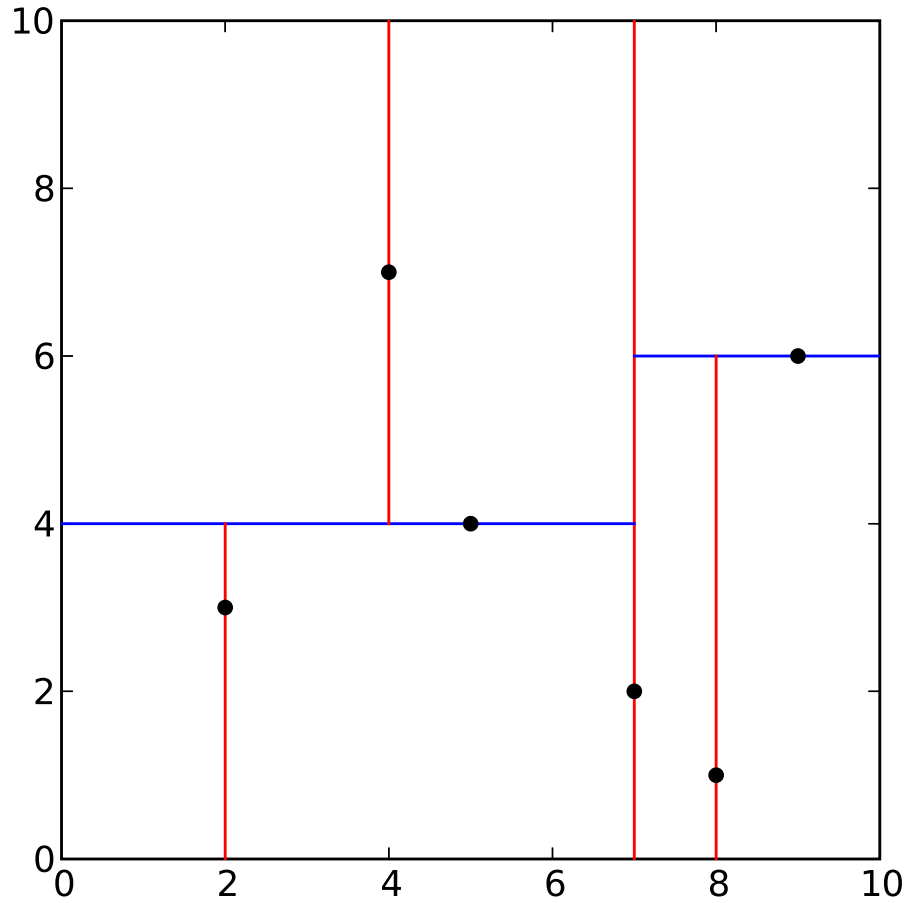
- ◆ Trivial implementation (\rightarrow good baseline method)
- ◆ 1-NN: Bayes error ϵ_B is the lower bound on error of classification ϵ_{NN} (in the asymptotic case $N \rightarrow \infty$.) Upper bounds can also be constructed, e.g. $\epsilon_{NN} \leq 2\epsilon_B$
- ◆ Slow when implemented naively, but can be sped up (Voronoi, k-D trees)
- ◆ High computer memory requirements (but training set can be edited and its cardinality decreased)
- ◆ How to construct the metric d ? (problem of scales in different axes)

K-NN : Speeding Up the Classification

- ◆ Sophisticated algorithms for NN search:
 - Classical problem in Comp. Geometry
 - k-D trees
- ◆ Removing the samples from the training class \mathcal{T} which do not change the result of classification
 - Exactly: using Voronoi diagram
 - Approximately: E.g. use Gabriel graph instead of Voronoi
 - Condensation algorithm: iterative, also approximate.

K-d Tree

k-d tree decomposition for the point set $(2,3)$, $(5,4)$, $(9,6)$, $(4,7)$, $(8,1)$, $(7,2)$



Condensation Algorithm

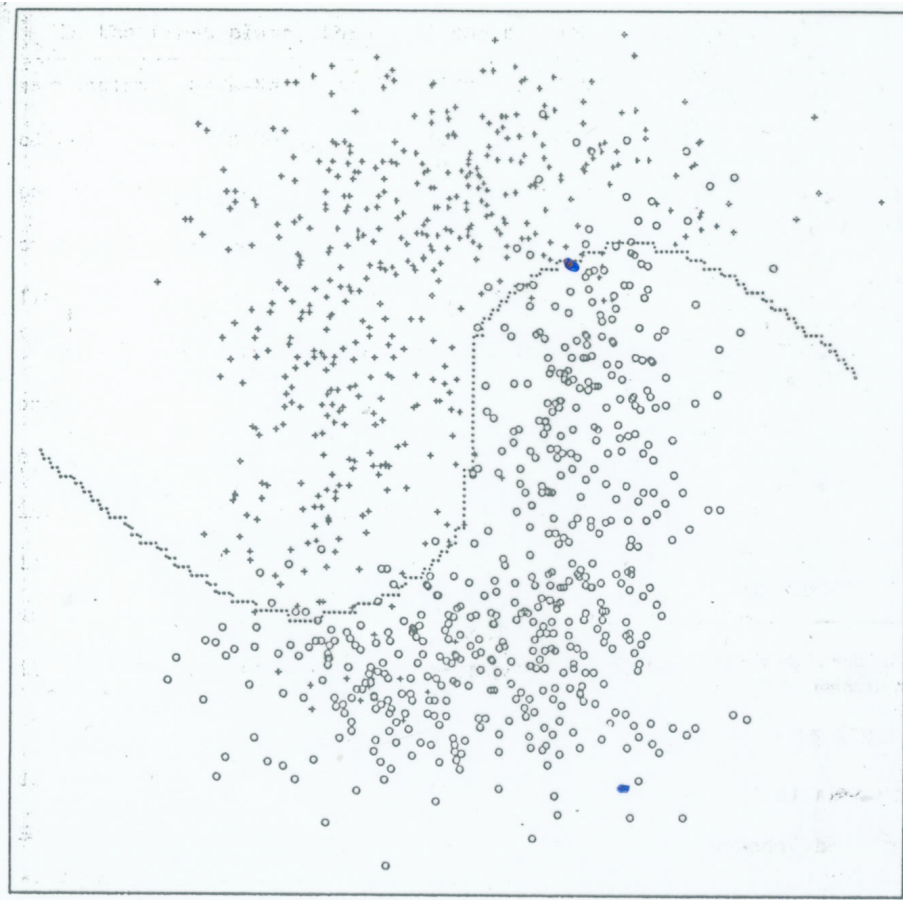
Input: The training set \mathcal{T} .

Algorithm

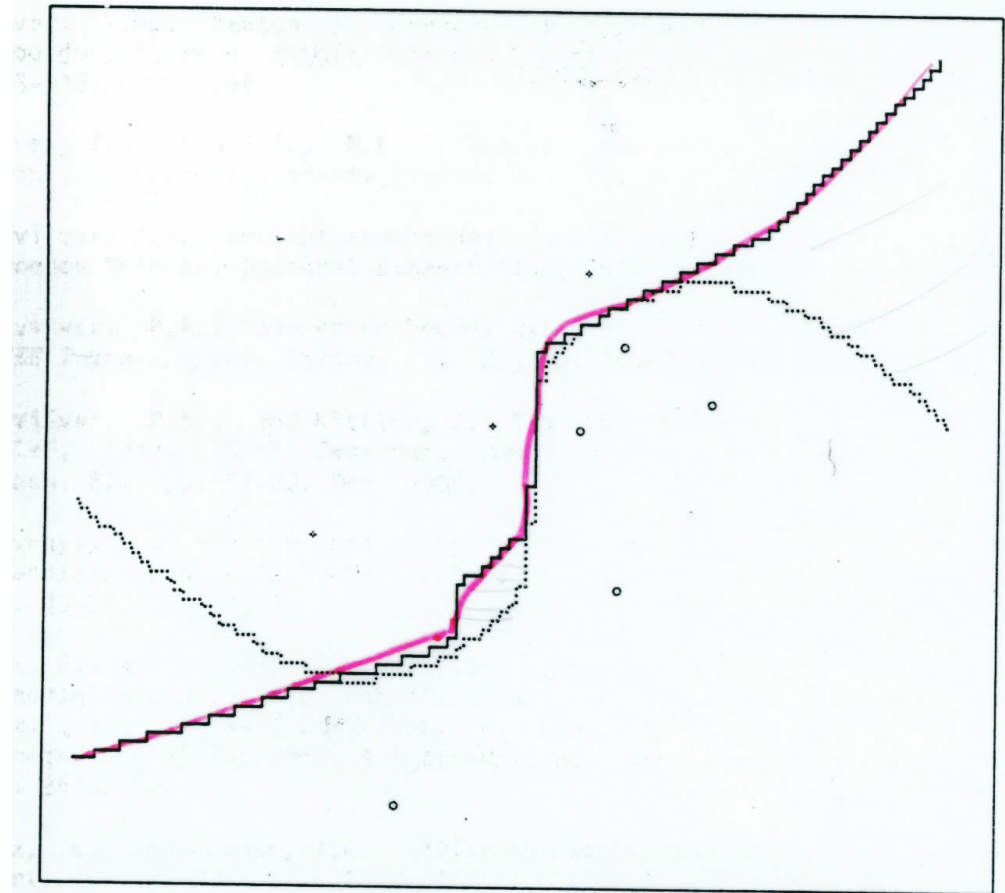
1. Create two lists, A and B . Insert a randomly selected sample from \mathcal{T} to A . Insert the rest of the training samples to B .
2. Classify samples from B using 1NN with training set A . If an $x \in B$ is mis-classified, move it from B to A .
3. If a move has been triggered in Step 2., goto Step 2.

Output: A (the condensed training set for 1NN classification)

Condensation Algorithm, Example



The training dataset



The dataset after the condensation.
Shown with the new decision boundary.

1-NN Classification Error

Recall that a classification error $\bar{\epsilon}$ for strategy $q: X \rightarrow R$ is computed as

$$\bar{\epsilon} = \int \sum_{k:q(x) \neq k} p(x, k) dx = \int \underbrace{\sum_{k:q(x) \neq k} p(k|x) p(x)}_{\epsilon(x)} dx = \int \epsilon(x) p(x) dx. \quad (21)$$

We know that the Bayesian strategy q_B decides for the highest posterior probability $q(x) = \operatorname{argmax}_k p(k|x)$, thus the partial error $\epsilon_B(x)$ for a given x is

$$\epsilon_B(x) = 1 - \max_k p(k|x). \quad (22)$$

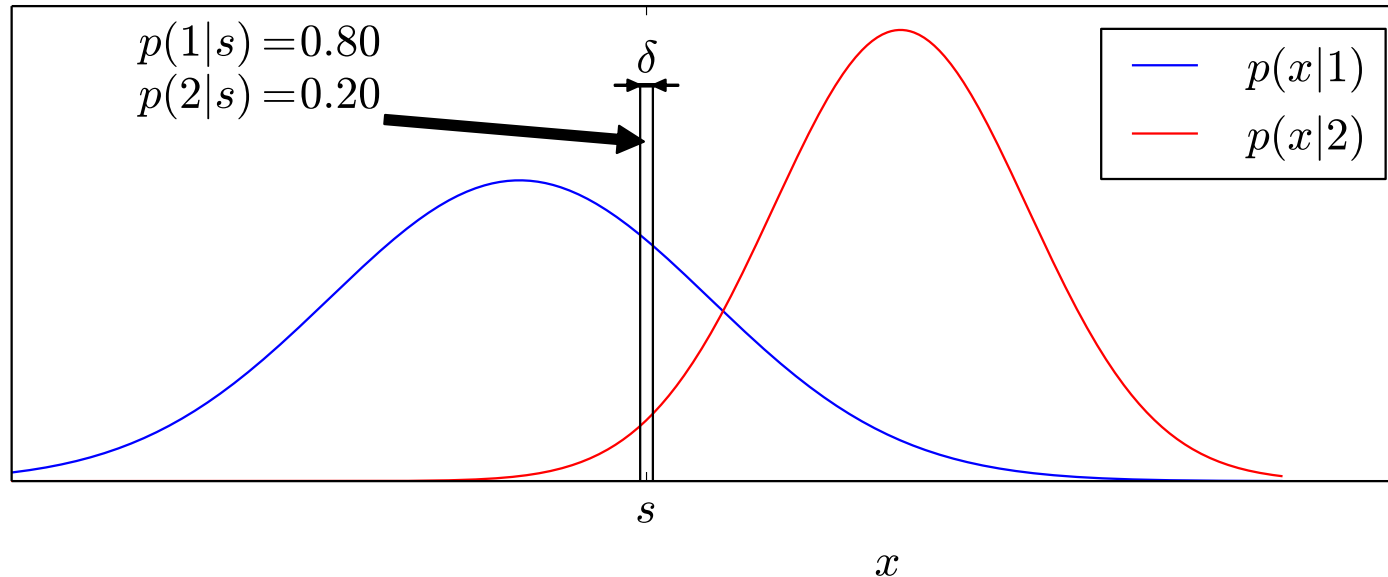
Assume the asymptotic case. We will show that the following bounds hold for the partial error $\epsilon_{NN}(x)$ and classification error $\bar{\epsilon}_{NN}$ in the 1-NN classification,

$$\epsilon_B(x) \leq \epsilon_{NN}(x) \leq 2\epsilon_B(x) - \frac{R}{R-1}\epsilon_B^2(x), \quad (23)$$

$$\bar{\epsilon}_B \leq \bar{\epsilon}_{NN} \leq 2\bar{\epsilon}_B - \frac{R}{R-1}\bar{\epsilon}_B^2, \quad (24)$$

where $\bar{\epsilon}_B$ is the Bayes classification error and R is the number of classes.

1-NN Classification Error, Example (1)



Consider two distributions as shown, a small interval δ on an x -axis, and a point $s \in \delta$. Let the class priors be $p(1) = p(2) = 0.5$. Assume $\delta \rightarrow 0$ and number of samples $N \rightarrow \infty$.

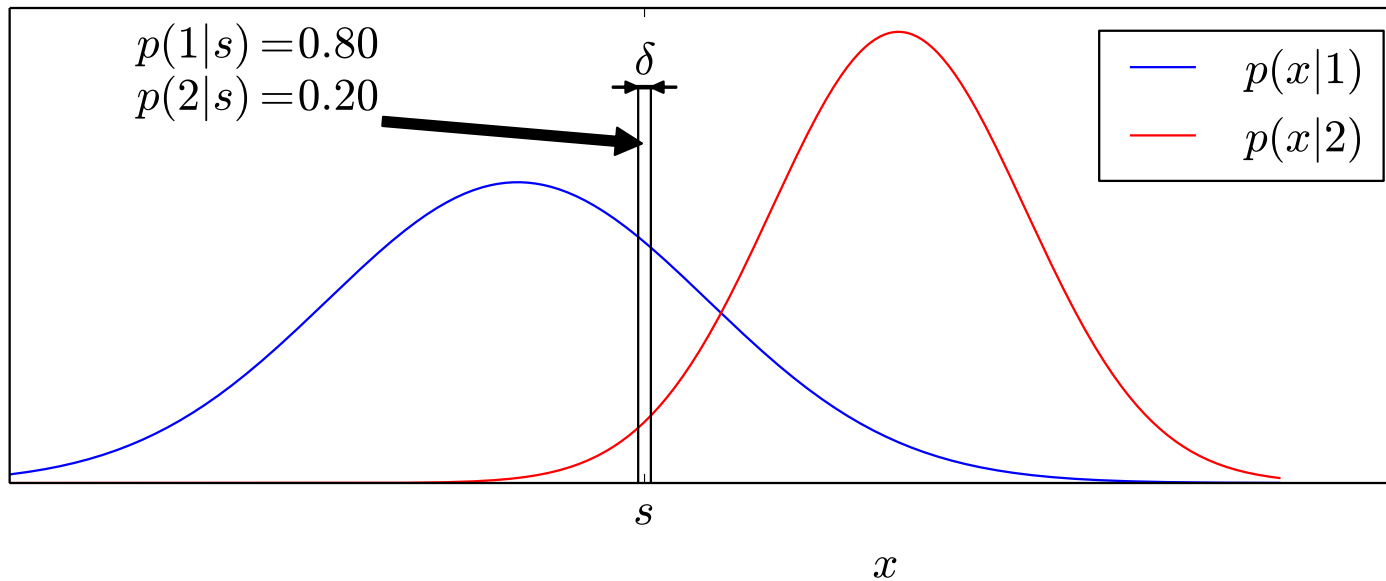
Observe the following:

$$p(1|s) = 0.8, \quad p(2|s) = 0.2, \quad (25)$$

$$p(NN=1|s) = p(1|s) = 0.8, \quad p(NN=2|s) = p(2|s) = 0.2, \quad (26)$$

where $p(NN=k|s)$ is the probability that the 1-NN of s is from class k ($k = 1, 2$) and thus s is classified as k .

1-NN Classification Error, Example (2)



The error $\epsilon_{NN}(s)$ at s is

$$\epsilon_{NN}(s) = p(1|s) p(NN=2|s) + p(2|s) p(NN=1|s) \quad (27)$$

$$= 1 - p(1|s) p(NN=1|s) - p(2|s) p(NN=2|s) \quad (28)$$

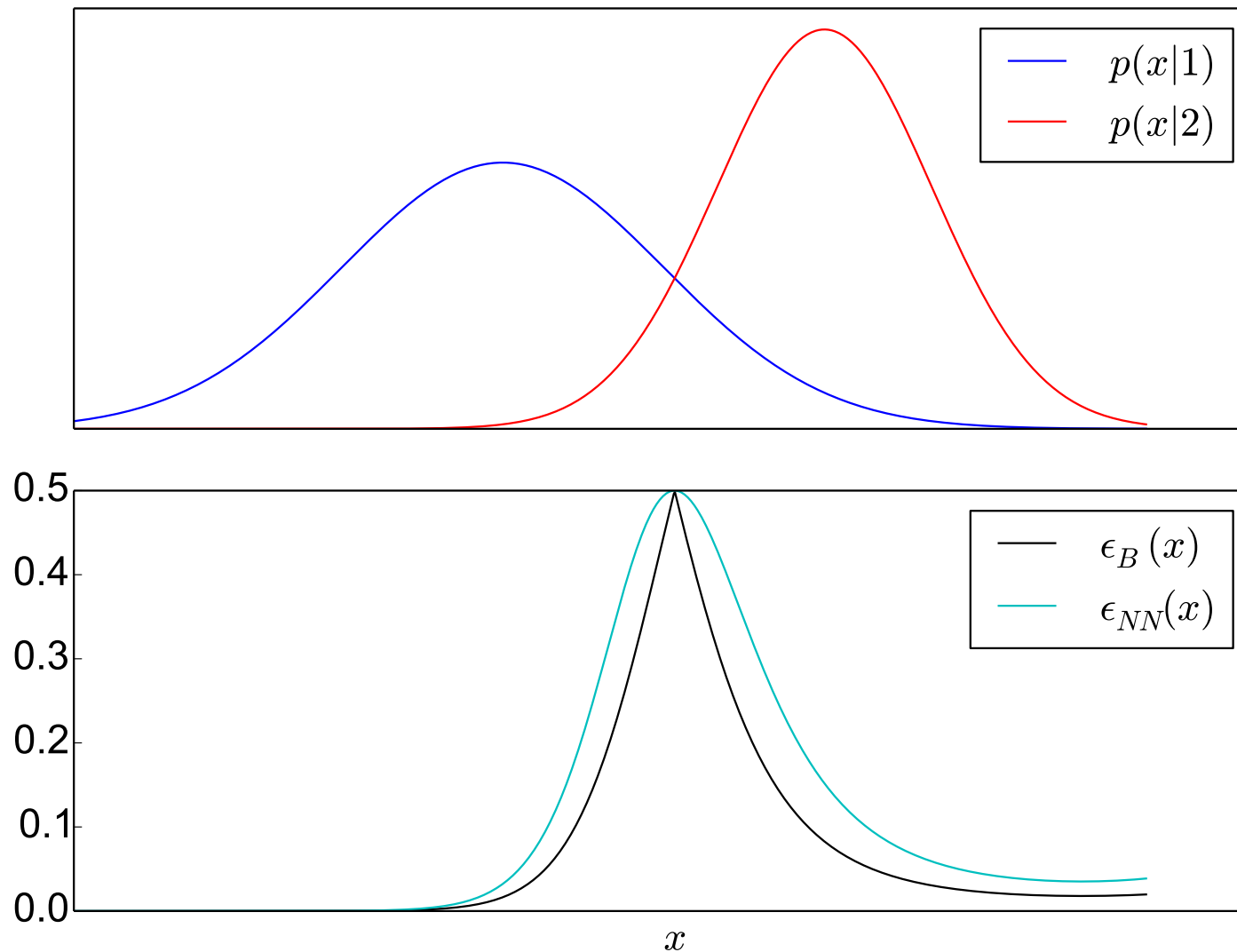
$$= 1 - p^2(1|s) - p^2(2|s). \quad (29)$$

Generally, for R classes, the error will be

$$\epsilon_{NN}(s) = 1 - \sum_{k \in R} p^2(k|s). \quad (30)$$

1-NN Classification Error, Example (3)

The two distributions and the partial errors
(the Bayesian error $\epsilon_B(x)$ and the 1-NN error $\epsilon_{NN}(x)$)



1-NN Classification Error Bounds (1)

Let us now return to the inequalities and prove them:

$$\epsilon_B(x) \leq \epsilon_{NN}(x) \leq 2\epsilon_B(x) - \frac{R}{R-1}\epsilon_B^2(x), \quad (31)$$

The **first** inequality follows from the fact that Bayes strategies are optimal.

To prove the **second** inequality, let $P(x)$ denote the maximum posterior for x :

$$P(x) = \max_k p(k|x) \quad (32)$$

$$\Rightarrow \epsilon_B(x) = 1 - P(x). \quad (33)$$

Let us rewrite the partial error $\epsilon_{NN}(x)$ using the Bayesian entities $P(x)$ and $q(x)$:

$$\epsilon_{NN}(x) = 1 - \sum_{k \in R} p^2(k|x) = 1 - P^2(x) - \sum_{k \neq q(x)} p^2(k|x). \quad (34)$$

We know that $p(q(x)|x) = P(x)$, but the remaining posteriors can be arbitrary. Let us consider the worst case. i.e. set $p(k|x)$ for $k \neq q(x)$ such that Eq. (34) is maximized. This will provide the upper bound.

1-NN Classification Error Bounds (2)

There are the following constraints on $p(k|x)$ ($k \neq q(x)$):

$$\sum_{k \neq q(x)} p(k|x) + P(x) = 1 \quad (\text{posteriors sum to 1}) \quad (35)$$

$$\sum_{k \neq q(x)} p^2(k|x) \rightarrow \min \quad (36)$$

It is easy to show that this optimization problem is solved by setting all the posteriors to the same number. Thus,

$$p(k|x) = \frac{1 - P(x)}{R - 1} = \frac{\epsilon_B(x)}{R - 1} \quad (k \neq q(x)) \quad (37)$$

The upper bound can then be rewritten in terms of the Bayes partial error $\epsilon_B(x) = 1 - P(x)$:

$$\epsilon_{NN}(x) \leq 1 - P^2(x) - \sum_{k \neq q(x)} p^2(k|x) = 1 - (1 - \epsilon_B(x))^2 - (R - 1) \frac{\epsilon_B^2(x)}{(R - 1)^2}. \quad (38)$$

1-NN Classification Error Bounds (3)

$$\epsilon_{NN}(x) \leq 1 - P^2(x) - \sum_{k \neq q(x)} p^2(k|x) = 1 - (1 - \epsilon_B(x))^2 - \frac{\epsilon_B^2(x)}{R-1}. \quad (39)$$

After expanding this, we get

$$\epsilon_{NN}(x) \leq 1 - (1 - \epsilon_B(x))^2 - \frac{\epsilon_B^2(x)}{(R-1)} \quad (40)$$

$$= 1 - 1 + 2\epsilon_B(x) - \epsilon_B^2(x) - \epsilon_B^2(x) \frac{R}{R-1} \quad (41)$$

$$= 2\epsilon_B(x) - \epsilon_B^2(x) \frac{R}{R-1} \quad (42)$$

Note that for $R = 2$, the bound is tight because using $\epsilon_B(x) = 1 - P(x)$ in Eq. (39) gives

$$\epsilon_{NN}(x) \leq 1 - P^2(x) - \frac{(1 - P(x))^2}{1} = \epsilon_{NN}(x). \quad (43)$$

1-NN Classification Error Bounds (4)

The inequality for the local errors has been proven:

$$\epsilon_{NN}(x) \leq 2\epsilon_B(x) - \epsilon_B^2(x) \frac{R}{R-1} \quad (44)$$

Is there a similar upper bound for the classification error $\bar{\epsilon}_{NN} = \int \epsilon_{NN}(x)p(x)dx$, based on the Bayes error $\bar{\epsilon}_B = \int \epsilon_B(x)p(x)dx$?

Multiplying Eq. (45) by $p(x)$, and integrating, gives

$$\bar{\epsilon}_{NN} \leq 2\bar{\epsilon}_B - \frac{R}{R-1} \int \epsilon_B^2(x)p(x)dx \quad (45)$$

Let us use the known identity and inequality (where $E(\cdot)$ is the expectation operator)

$$\text{var}(x) = E(x^2) - E^2(x), \text{var}(x) \geq 0 \quad \Rightarrow \quad E(x^2) \geq E^2(x) \quad (46)$$

Thus, $\int \epsilon_B^2(x)p(x)dx \geq (\int \epsilon_B(x)p(x)dx)^2$, and

$$\bar{\epsilon}_{NN} \leq 2\bar{\epsilon}_B - \frac{R}{R-1} \int \epsilon_B^2(x)p(x)dx \leq 2\bar{\epsilon}_B - \frac{R}{R-1} \bar{\epsilon}_B^2 \quad (47)$$

***K*-NN Classification Error Bound**

It can be shown that for *K*-NN, the following inequality holds:

$$\bar{\epsilon}_{KNN} \leq \bar{\epsilon}_B + \bar{\epsilon}_{1NN} / \sqrt{K \text{ const}} \quad (48)$$

Edit algorithm

The primary goal of this method is to reduce the classification error (not the speed-up of classification.)

Input: The training set \mathcal{T} .

Algorithm

1. Partition \mathcal{T} to two sets, A and B ($\mathcal{T} = A \cup B$, $A \cap B = \emptyset$.)
2. Classify samples in B using **K**-NN with training set A . Remove all samples from B which have been mis-classified.

Output: B the training set for **1**-NN classification.

Asymptotic property:

$$\bar{\epsilon}_{edit} = \bar{\epsilon}_B \frac{1 - \bar{\epsilon}_B}{1 - \bar{\epsilon}_{KNN}} \quad (49)$$

If $\bar{\epsilon}_{KNN}$ is small (e.g. 0.05) then the edited 1NN is quasi-Bayes (almost the same performance as Bayesian Classification.)