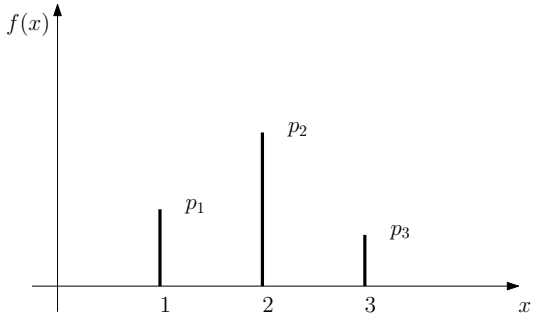
1 Introduction

Description of variable: value

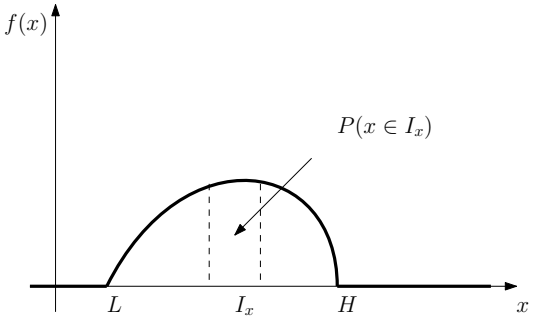
Description of random variable: Distribution = all possible values and their "probabilities"

Discrete random variable



where $p_1 \geq 0$ and $\sum p_1 = 1$ (are probabilities of the values).

Continuous random variable



where $f(x) \geq 0$,$x \in R$ and $\int_L^H f(x) = 1$.

Discretization

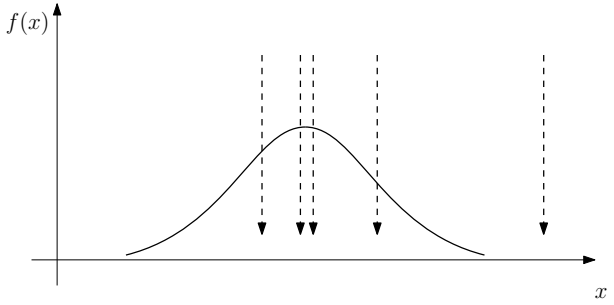
$$x = [x_1, x_2, \cdots x_n]\,; \text{ with } x_{i+1} - x_i = h$$

it is

$$P\left(x \in \left(x_i \pm \frac{h}{2}\right)\right) \doteq f(x_i)\, h$$

$\rightarrow f(x_i)$ points at the probability of x_i (of the neighborhood of x_i)

Basic property of distribution: The values are most probably near its center.



the closer the value x is to the top of distribution

$\Downarrow$

the higher is $f(x)$

$\Downarrow$

the greater is the probability that x was generated by this distribution

Model: Conditional probability distribution (density function)

$$f\,(\text{result}|\text{action})$$

mostly

$$f\,(y_t|\Theta) \quad \text{or} \quad f\,(y_t|x_t,\Theta)$$

where y_t is the target (modeled) variable, x_t is a vector of explanatory variables and Θ are model parameters.

For known parameters, we write $f\,(y_t|x_t)$, only.

Remark

$t = 1, 2, 3, \cdots$ is discrete time (instants of measurements).

y_t and $x_t = [x_1, x_2, \cdots, x_n]_t$ are the data examined; x_t should explain behaviour of y_t

Θ expresses the properties of the described system.

Example

$$y_t = 0.1y_{t-1} \;\rightarrow y = 10,\ 1,\ 0.1,\ 0.01,\ \cdots$$

$$y_t = 0.99y_{t-1} \rightarrow y = 10,\ 9.9,\ 9.81,\ 9.703,\ \cdots$$

Estimation: For model with unknown parameters, we need to estimate them.

$$f(\Theta|y) \propto f(y|\Theta) f(\Theta)$$

$\cdots$ this is, from the result we estimate the action which has evoke it.

Usually we use some dataset $y(T) = \{y_1, y_2, \cdots, y_T\}$. Then the estimation is recursive

$$f(\Theta|y(t)) \propto f(y_t|\Theta) f(y(t-1))$$

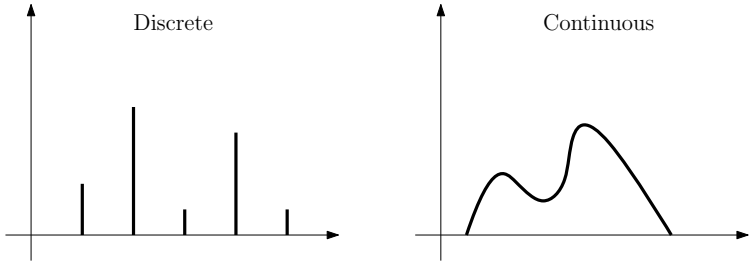
for $t = 0, 1, \cdots T$ where $y(0)$ are prior data.

Remark

Bayes rule

$$f(B|A, C) = \frac{f(A|B, C) f(B|C)}{f(A|C)}$$

Multimodal data - several working modes generating data with higher density (clusters).



For clusters we introduce discrete variable c_t - pointer. Its value at time t points at the active cluster.

Model of data in cluster

$$f\left(y_t|c_t\right)$$

Model of clusters in a dataset

$$f\left(c_t|y_t\right)$$

It holds

$$f\left(c_t|y_t\right) \propto f\left(y_t|c_t\right) f\left(c_t\right)$$

Details and programs of model estimation

The details of estimation can be found at

- Laboratory - Chapters 2 and 3 (on main webpage)
- on web: https://www.fd.cvut.cz/personal/nagyivan/MMADpilots/MMAD_pilots.html : PrgsScilab or in zipped file that can be downloaded at Programsto Scilab (these are the programs model*.sce and init*.sce)

2 Classification

Clustering: Grouping data into several classes. E.g. kids, youngsters, adults, old.

Classification: Sorting data into existing classes. E.g. The incoming person is to be assigned into one of the defined groups. (If according to the age, clear; if according to the appearance, uncertain)

***Example** 1: Let us have $y \in \{1, 2, 3\}$ with two equally probable classes with the model*

$f(y\|c)$	$c = 1$	$c = 2$
$y = 1$	*0.2*	*0.4*
$y = 2$	*0.7*	*0.3*
$y = 3$	*0.1*	*0.3*

Perform classification of the dataset $y = \{1, 3, 2, 3, 2\}$.

The classes are given by the maximum probability of $f(c|y) \propto f(y|c)$; $f(c)$ is uniform. This is equal to the transposition of the model

$f(c\|y)$	$y = 1$	$y = 2$	$y = 3$
$c = 1$	*0.2*	*0.7*	*0.1*
$c = 2$	*0.4*	*0.3*	*0.3*

From it, we have

$$y = 1 \to c = 2,\ y = 2 \to c = 1,\ y = 3 \to c = 2$$

For the dataset $y = \{1, 3, 2, 3, 2\}$, *the classification is* $c = \{2, 2, 1, 2, 1\}$. □

Example *2: Let us have multimodal system with two modes described by the models (components)* $f_1(y)$ *and* $f_2(y)$ *we denote* $f(y)$[1]

$$f(y|c = 1) = N_y(\mu = 5, r = 1)$$

$$f(y|c = 2) = N_y(\mu = 2, r = 1)$$

The probability of the first class is $f(c = 1) = 0.2$ *and the second is* $f(c = 2) = 0.8$. *Perform classification of the dataset* $y = \{4.3,\ 2.1,\ 3.4\}$

For the classification we maximize

$$f(c|y) \propto f(y|c) f(c)$$

with respect to c.

---

[1] *It is*

$$N_y(\mu, r) = \frac{1}{\sqrt{2\pi r}} \exp\left\{-\frac{1}{2r}(y - \mu)^2\right\}$$

From this we have

y	$f(y\|c=1)\,f(c=1)$	$f(y\|c=2)\,f(c=2)$	*class*
4.3	*0.062*	*0.022*	*1*
2.1	*0.001*	*0.317*	*2*
3.4	*0.022*	*0.119*	*2*

and we select the components with the grater value in the rows. □

***Example 3**: Why naive Bayes!*

We have 3-dimensional multinomial variable $y = [y_1, y_2, y_3]$ *with categorical distribution (i.e. each triple* $[y_1, y_2, y_3]$ *has its probability* $p_{1,2,3}$*) with* $y_1 \in \{1, 2, \cdots, 5\}$, $y_2 \in \{1, 2, \cdots 8\}$ *and* $y_3 \in \{1, 2, \cdots 6\}$. *The model for a single component is given by a vector of probabilities for each combination of the values of* y*, i.e.*

y_1	y_2	y_3	$p_{1,2,3}$
1	*1*	*1*	·
1	*1*	*2*	·
	···		
1	*2*	*1*	·
	etc.		

This table has 5·8·6 = 240 rows (only for one component).

If we assume independency of y, the description is given by three vectors with total dimension 5+8+6 = 19, which is substantially less then before. And the joint probability is given by the product of marginals. □

That is, why Naive Bayes method, who assumes y_i as independent, is so useful. And moreover, its quality is surprisingly good even if the condition if the independency is not absolutely true.

3 Naive Bayes

Mostly we use more than one variable $y_t = [y_1, y_2, \cdots, y_n]_t$ for $t = 1, 2, \cdots, T$.

y_t is a measurement at time t; $\quad y(t) = \{y_1, y_2, \cdots, y_t\}$ is a set of all measurements.

Naive Bayes assumption

Naive Bayes method assumes y **conditionally independent**, i.e.

$$f(y|c) = \prod_{i=1}^{n} f(y_i|c)$$

What is $y|c$? $\cdots$ Example: For the data $y = [y_1, y_2]$ and components c

y_1	y_2	c
5.2	3.8	1
1.4	8.3	2
1.6	7.9	2
4.9	3.5	1
1.5	8.1	2

$y\|1$	
5.2	3.8
4.9	3.5

$y\|2$	
1.4	8.3
1.6	7.9
1.5	8.1

Known parameters of the model

$$f(c|y) \propto f(y|c)\, f(c) = f(c) \prod_{i=1}^{n} f(y_i|c)$$

Example: (the previous one - Example 3)

We have 3-dimensional ***independent*** *multinomial variable* $y = [y_1, y_2, y_3]$ *with categorical distribution (i.e. each* ***variable*** y_i *has its probability* p_i*) with* $y_1 \in \{1, 2, \cdots, 5\}$, $y_2 \in \{1, 2, \cdots 8\}$ *and* $y_3 \in \{1, 2, \cdots 6\}$ *and two classes* $c \in \{1, 2\}$ *defined by the models* $f(c) = [0.2,\ 0.8]$ *and*

$f(y_1\|c)$	$c=1$	$c=2$
$y=1$	0.1	0.3
$y=2$	0.3	0.1
$y=3$	0.2	0.1
$y=4$	0.3	0.4
$y=5$	0.1	0.1

$f(y_2\|c)$	$c=1$	$c=2$
$y=1$	0.1	0.2
$y=2$	0.1	0.1
$y=3$	0.2	0.1
$y=4$	0.2	0.1
$y=5$	0.1	0.1
$y=6$	0.1	0.2
$y=7$	0.1	0.1
$y=8$	0.1	0.1

$f(y_3\|c)$	$c=1$	$c=2$
$y=1$	0.1	0.3
$y=2$	0.3	0.2
$y=3$	0.2	0.1
$y=4$	0.2	0.1
$y=5$	0.1	0.1
$y=6$	0.1	0.2

Classify the measured triple $[y_1, y_2, y_3] = [2, 5, 3]$

According to Naive Bayes we have

$$f\left(c|y_1=2,y_2=5,y_3=3\right)\propto f\left(c\right)f\left(y_1=2|c\right)f\left(y_2=5|c\right)f\left(y_3=3|c\right)\propto$$

$$=[0.2,\ 0.8]\times[0.3,\ 0.1]\times[0.1,\ 0.1]\times[0.2,\ 0.1]=[0.0012,\ \ 0.0008]$$

As the maximum is at the first entry, we classify to the first class.

Program: PrgsScilab (on web), Chapter 4, Section 4.1: `class_1.sce`

Unknown parameters of the model

(learning with a teacher)

• Naive Bayes learning with normal components

Teacher knows the switching of the components, i.e. actual values of the pointer c_t for learning data $\{y_t\}_{t=1}^{nL}$. We can estimate each component with its data separately.

Example: Estimate multi-model with two normal components using the following data

t	1	2	3	4	5	6	7	8	9	10
$y_{1;t}$	4.8	8.1	4.5	4.2	9.1	3.5	9.3	8.5	5.1	4.9
$y_{2;t}$	15.3	7.4	16.2	15.8	6.8	14.5	5.1	5.3	15.5	14.9
c_t	1	2	1	1	2	1	2	2	1	1

and classify $y = [5.3,\ 14.9]$.

The clusters (data from components) are $C^{\text{component}}_{\text{variable}}$

$$C_1^1 = \{4.8,\ 4.5,\ 4.2,\ 3.5,\ 5.1,\ 4.9\}, \quad C_1^2 = \{8.1,\ 9.1,\ 9.3,\ 8.5\}$$

$$C_2^1 = \{15.3,\ 16.2,\ 15.8,\ 14.5,\ 15.5,\ 14.9\},\ C_2^2 = \{7.4,\ 6.8,\ 5.1,\ 5.3\}$$

and

cluster	average	variance
C_1^1	4.5	0.34
C_1^2	8.75	0.3
C_2^1	15.37	0.37
C_2^2	6.15	1.27

The component estimates are

$$f\left(y_1|c=1\right)=N_{y_1}\left(4.5,\,0.34\right),\quad f\left(y_1|c=2\right)=N_{y_1}\left(8.75,\,0.3\right)$$

$$f\left(y_2|c=1\right)=N_{y_2}\left(15.37,\,0.37\right),\quad f\left(y_2|c=2\right)=N_{y_2}\left(6.15,\,1.27\right)$$

The model of the pointer is a normalized histogram of the pointer values

$$f\left(c\right)=[0.6,\,0.4]$$

Then

$$f\left(c|y=[5.3,\,14.9]\right)\propto$$

$$\propto\left[f\left(c=1\right)f\left(y_1=5.3|c=1\right)f\left(y_2=0.4|c=1\right);\;f\left(c=2\right)f\left(y_1=5.3|c=2\right)f\left(y_2=0.4|c=2\right)\right]=$$

$$[0.6 \times N_{y_1=5.3}\,(4.5,\, 0.34)\, N_{y_2=14.9}\,(15.37,\, 0.37)\,;\;\; 0.4 \times N_{y_2=5.3}\,(8.75,\, 0.3)\, N_{y_2=14.9}\,(6.15,\, 1.27)] =$$

$$\left[0.0779;\;\; 2 \times 10^{-23}\right]$$

Thus, the vector y is classified into the first component.

Program: PrgsScilab (on web), Chapter 4, Section 4.2: `class_3.sce`

• **Naive Bayes with categorical components**

Again learning with a teacher.

Example: Learn classification model from the data ($C^{\text{component}}_{\text{variable}}$)

$$C_1^1 = \{1,\, 1,\, 2,\, 1,\, 2,\, 1\}\,,\;\; C_1^2 = \{2,\, 2,\, 2,\, 1\}$$

$$C_2^1 = \{2,\, 1,\, 2,\, 2,\, 3,\, 1,\, 2,\, 1\}\,,\;\; C_2^2 = \{3,\, 1,\, 2,\, 3,\, 3\}$$

and classify the data record $y = [1, 3]$.

We have

var 1|clus 1		
values	1	2
freq.	4	2
prob.	2/3	1/3

var 1|clus 2		
values	1	2
freq.	1	3
prob.	1/4	3/4

var 2|clus 1			
values	1	2	3
freq.	3	4	1
prob.	3/8	1/2	1/8

var 2|clus 2			
values	1	2	3
freq.	1	1	3
prob.	1/5	1/5	3/5

From it, we get **models of components**

$f(y_1\|c)$	$c=1$	$c=2$
$y_1=1$	$2/3$	$1/4$
$y_1=2$	$1/3$	$3/4$

$f(y_2\|c)$	$c=1$	$c=2$
$y_2=1$	$3/8$	$1/5$
$y_2=2$	$1/2$	$1/5$
$y_2=3$	$1/8$	$3/5$

and model of the pointer $f(c) = [14/23;\ 9/23]$ $\cdots$ rel. freq. of $c=1$ and $c=2$.

Classification of $y=[1,3]$

$$f(c|y=[1,3]) \propto f(y_1=1|c)\, f(y_2|c)\, f(c) =$$

$$= [2/3,\ 1/4] \times [1/8,\ 3/5] \times [14/23,\ 9/23] = [0.0507,\ 0.0587]\,.$$

The second entry is greater, so we classify to the second component.

• **Naive Bayes with kernel estimation**

The kernel function is

$$\hat{k} = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h}K\left(\frac{y-y_i}{h}\right)$$

with $\frac{1}{h}K\left(\frac{y-y_i}{h}\right)$ defined in different ways; e.g. like the Gaussian bell function

$$\frac{1}{h}\exp\left\{-\frac{1}{2}\left(\frac{y-y_i}{h}\right)^2\right\}$$

where y_i are data from the dataset. Here, $h \to \sigma$ and $y_i \to \mu$. The parameter h is important and often is selected as

$$h = \frac{\sqrt{\sigma^2}}{N^{1/5}}$$

σ^2 is variance of y and N is the length of the dataset.

Example: Perform classification of multimodal data y with three components based on kernel approximation of the component models with the learning data sorted to the clusters $C_j, j = 1, 2, 3$.

$$C_1 = \{2.6,\ 3.1,\ 0.4,\ 2.9\}$$

$$C_2 = \{13.2,\ 11.3,\ 15.4,\ 12.8\}$$

$$C_3 = \{7.5,\ 8.2, 6.7,\ 9.1\}$$

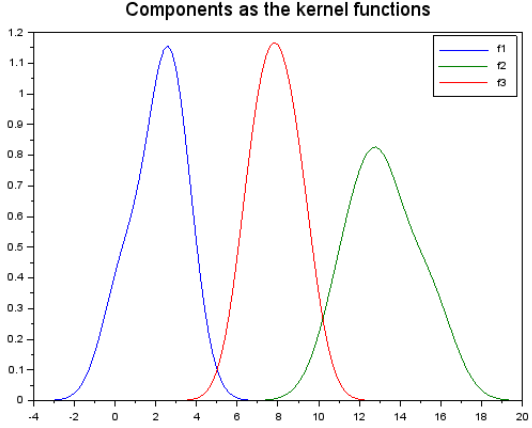
First we can see, that the estimate of the probabilities of components are uniform (why?). The density functions of the components will be kernel approximated:

$$\hat{k}_1(y) = \frac{1}{h}\exp\left\{-\frac{1}{2}\left(\frac{y-2.6}{h}\right)^2\right\} + \frac{1}{h}\exp\left\{-\frac{1}{2}\left(\frac{y-3.1}{h}\right)^2\right\} +$$

$$+\frac{1}{h}\exp\left\{-\frac{1}{2}\left(\frac{y-0.4}{h}\right)^2\right\} + \frac{1}{h}\exp\left\{-\frac{1}{2}\left(\frac{y-2.9}{h}\right)^2\right\}$$

where $h = 1.03$ for all kernels.

Similarly $\hat{k}_2(y)$ and $\hat{k}_3(y)$. We get



Remark: Notice, that the kernel functions are not Gaussian functions - they approximate the distribution of the points.

Now, the classification is very natural

$$f\left(c|y\right) \propto f\left(y|c\right), \, c = 1, 2, 3$$

We insert the value of the classified variable into each component and classify to that with the greatest value.

E.g. for $y = 6.4$ we have

$$f\left(c|6.4\right) = [0.0024,\, 0.0001,\, 0.6919]$$

and we classify into th third class.

For multivariate $y = [y_1,\, y_2, \cdots, y_n]$ perform the above procedure for each variable y_i and in the end, we perform the product

$$f\left(y|c\right) = \prod_{i=1}^{n} f\left(y_i|c\right),\ \forall c$$

Again, we classify to the class with the maximum probability.

Program: PrgsScilab (on web), Chapter 4, Section 4.4: `class_5b.sce`

Mixture estimation

(learning without a teacher)

Uses on-line estimation with weighted data in statistics update. The weights follow from probabilistic classification.

Algorithm (runs in time $t = 1, 2. \cdots$)

1. measure new data record y_t
2. construct proximities q_j (values of component models with currently estimated parameters and the measured data)
$$q_j = f_j\left(y_t | \hat{\Theta}_{j;t-1}\right), \ \forall \, \text{components } j$$
3. construct weights $w = q / \sum q_j$ (normalization)
4. update all component statistics with weighted data (e.g.)
$$S_{j;t} = S_{j;t-1} + w_j y_t$$
$$\kappa_{j;t} = \kappa_{j;t-1} + w_j$$

5. construct point estimates of parameters

$$\hat{\Theta}_{j;t} = S_{j;t}/\kappa_{j;t}$$

6. Increment time $t \to t+1$ and go to 1.

Remark: For on-line estimation consult the Laboratory, Chapter 2 - Models and their estimation.

Program: See PrgsScilab.pdf - class_6a.sce

For comparison of single model estimation, estimation with teacher and mixture estimation see: `Est1RegMod.sce`, `Est2Teacher.sce` and `Est3Mixture.sce`.

Program: PrgsScilab (on web), Chapter 4, Section 4.5: `class_6a.sce`

4 Regression

Logistic regression

For binomial data $c_t \in \{0, 1\}$.

Bernoulli distribution

$$f\left(c_t|p\right) = p^{c_t}\left(1-p\right)^{1-c_t},\ p \in (0,1)$$

where $p = P\left(c_t = 1\right)$.

With explanatory variables $x_t = \left[x_{1;t}, x_{2;t}, \cdots, x_{m;t}\right],\ t = 1, 2, \cdots, T$ we model $logit(p)$ by regression

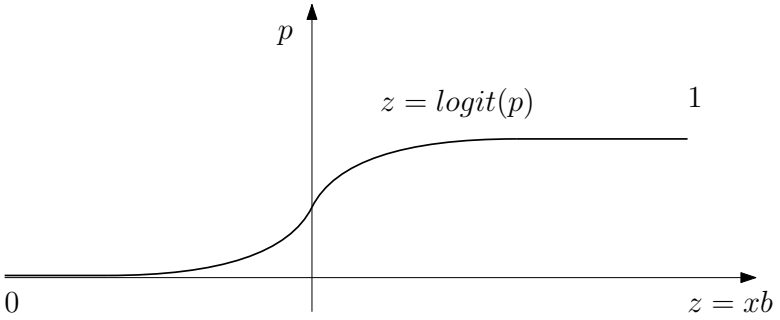
$$\underbrace{\ln\left(\frac{p}{1-p}\right)}_{logit(p)} = x_t b = \underbrace{b_0 + b_1 x_{1;t} + \cdots b_m x_{m;t}}_{z_t}$$

Remark: It holds

$$z_t = x_t b$$ – regression

$$\ln\left(\frac{p}{1-p}\right) = z_t \iff p = \begin{cases} \frac{\exp(z_t)}{1+\exp(z_t)} & \text{for } c_t = 1 \\ \frac{1}{1+\exp(z_t)} & \text{for } c_t = 0 \end{cases}$$ – normalization

The logit defines the function, i.e. transformation $R \to (0,1)$



How classification works

1. Measure x_t
2. Compute $x_t b = z_t$
3. Apply inverse *logit:* $p = \frac{\exp(z_t)}{1+\exp(z_t)}$
4. If $p = P(c_t = 1) > 0.5$ assign $c_t = 1$ otherwise $c_t = 0$

= classification of x_t to class 0 or 1.

Estimation

Model $f(c_t|p) = [1-p,\, p] = [P(c_t = 0),\, P(c_t = 1)]$

$$P(c_t) = \frac{\exp(c_t x_t b)}{1 + \exp(x_t b)},\ \forall c_t$$

Maximum likelihood

$$L(p) = \prod_{t=1}^{T} \frac{\exp(c_t x_t b)}{1 + \exp(x_t b)}$$

$\rightarrow$ numerical maximization

$$\hat{b} = \arg\max_{b} (L_p)$$

Program: in KNIME

see PrgsKNIME on web: `Task01_Logistic_Regresion`

Poisson regression

For nonnegative integer data $c_t = 0, 1, 2, \cdots$.

Poisson distribution

$$f\left(c_t|\lambda\right) = \exp\left(-\lambda\right)\frac{\lambda^{c_t}}{c_t!},\ \lambda > 0$$

The regression is

$$\ln\left(\lambda\right) = x_t b = b_0 + b_1 x_1 + b_2 x_2, \cdots, b_m x_m \iff \lambda = \exp\left(x_t b\right)$$

Model (in logarithm) with this regression is

$$\ln\left[f\left(c_t|x_t, b\right)\right] = -\exp\left(x_t b\right) + c_t x_t b - \ln\left(c_t\right)$$

logLikelihood

$$\ln L\left(b\right) = \sum_{t=1}^{T} \ln\left[f\left(c_t|x_t, b\right)\right]$$

and

$$\hat{b} = \arg\min\left(\ln L\right)$$

5 Classical clustering

K-means algorithm

The task: Divide the dataset $X = \{[x_1, x_2, \cdots, x_n]_t\}_{t=1}^{T}$ into m clusters according to their density.

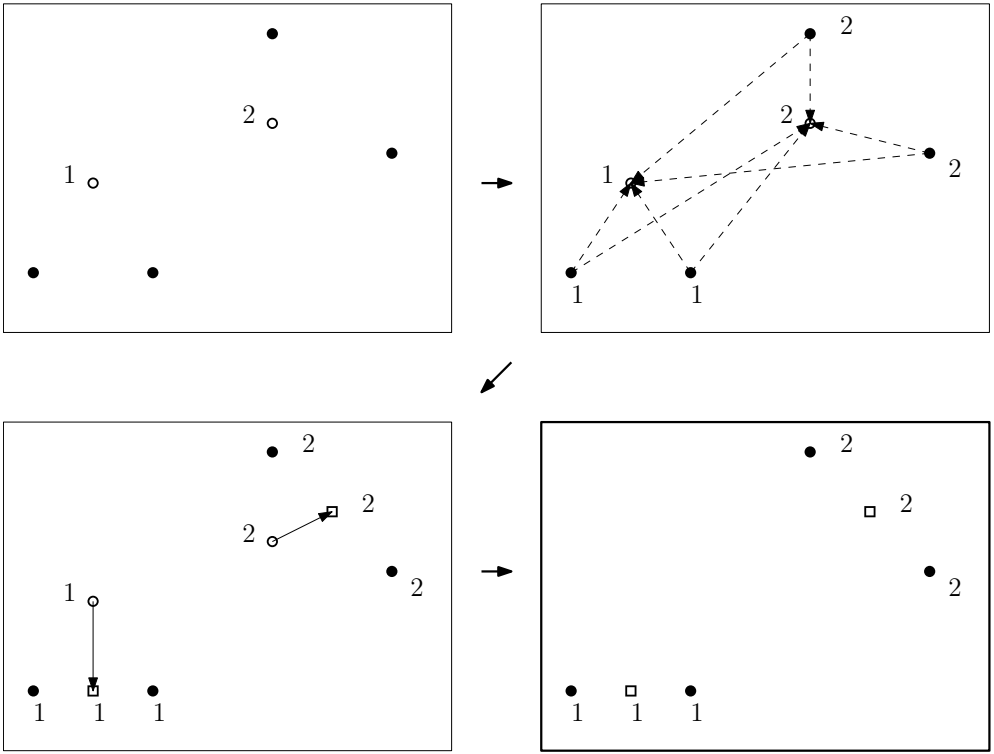
0. Determine the number m of clusters and set their initial centers.
1. Measure the distance from each data point to each cluster center and assign the point to the nearest center. The points form clusters.
2. Compute the average of points in each cluster and set it as its new center.
3. Check, if the centers changed. If yes, go to 1. If not, the algorithm ends.

Program: in KNIME

see PrgsKNIME on web: `Task02_k-Means_Clustering`

Remark: K-medoids algorithm is similar, but centers are always some of the points.

Example



C-means algorithm (fuzzy clustering)

The task: Divide the dataset $X = \{[x_1, x_2, \cdots, x_n]_t\}_{t=1}^{T}$ into m clusters according to their density.

In the c-means algorithm we minimize criterion

$$J = \sum_{i=1}^{N} \sum_{j=1}^{m} u_{ij}^{\omega} \|x_i - c_j\|^2, \ \omega \geq 1$$

where u_{ij} is a degree of membership of the point x_i to cluster c_j and $\|\cdot\|$ is a norm.

The algorithm runs in the iteration of the following two points:

- construct weights (are given as the membership functions)

$$u_{ij} = \frac{1}{\sum_{k=1}^{m} \left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|} \right)^{\frac{2}{\omega - 1}}} \tag{1}$$

- determine new centers (follows from minimization of the criterion)

$$c_j = \frac{\sum_{i=1}^{N} u_{ij}^{\omega} x_i}{\sum_{i=1}^{N} u_{ij}^{\omega}}$$

The algorithm starts with prior centers set by the user.

Remarks

1. The formula for c_j follows from minimization the criterion J.
2. u_{ij} says how strongly the point x_i belongs to the center c_j
3. $\frac{\|x_i - c_j\|}{\|x_i - c_k\|}$ is the distance of x_i from c_j relative to the distance of x_i from some other center c_k.
4. $\sum_{k=1}^{m} \left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|} \right)^{\frac{2}{\omega - 1}}$ is the share of the x_i to c_j distance in total distance of x_i from all centers.

Programs

see PrgsKNIME on web: `Task04_c-Means_Clustering`

and PrgsScilab on web: Section 4.6, C-means algorithm (program `Cmeans.sce`)

DBSCAN (density based clustering)

The task: In the dataset $X = \{[x_1, x_2, \cdots, x_n]_t\}_{t=1}^{T}$ find subsets of data with high density separated by areas with lover density.

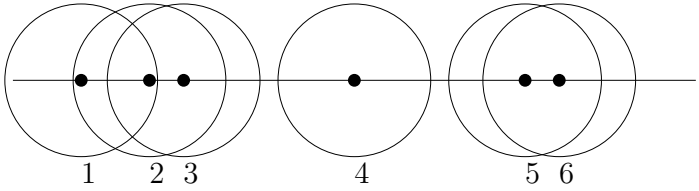
The basic notion is **ϵ-neighborhood** of a point. It decides whether the points are sufficiently dense.

- **Inner point** is such one that has in its neighborhood at least given number n_0 of points.
- A point x_i is **accessible** from the point x_j, if a sequence of inner points from x_i to x_j exists.
- A **connection** between points x_i a x_j exists, it both these points are accessible from some their inner point.

Clusters are formed by the connected points.

Remark : The choice of ϵ is critical.

Example



For $n_0 = 1$, the points 1,2,3 and 5,6 form clusters; the point 4 is a noise.

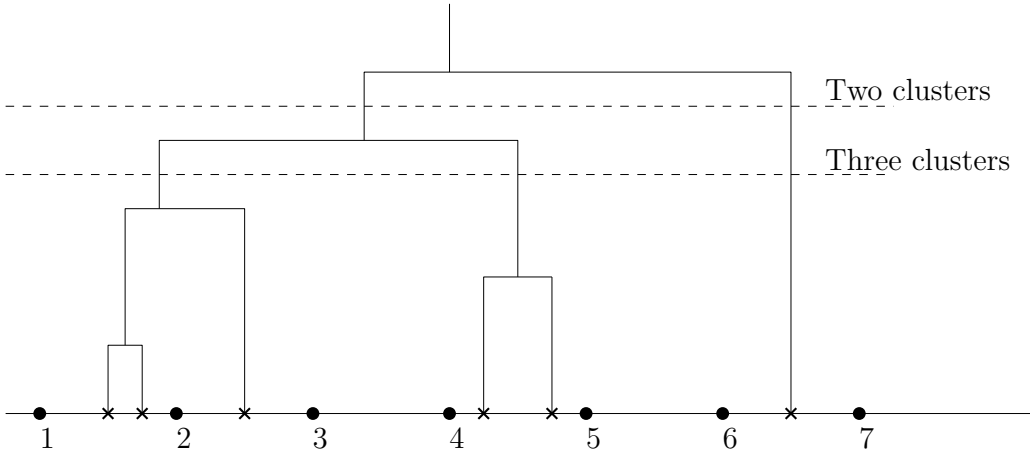
Program

see PrgsKNIME on web: `Task05_Density_Clustering`

Hierarchical clustering (agglomerative)

The task: In the dataset $X = \{[x_1, x_2, \cdots, x_n]_t\}_{t=1}^{T}$ construct a tree of clusters (dendrogram) and on its base cluster the data.

Example of a dendrogram



Creating dendrogram

At the beginning, all points are clusters with weight 1; then

1. Create new cluster by joining to nearest clusters,
   (a) the weight of the new cluster is sum of the weights of the joined clusters,
   (b) its position is $x_k = \frac{w_i x_i + w_j x_j}{w_i + w_j}$ where k is new cluster, i, j are joined clusters and w are weights.
2. Repeat, until all points are joined.

Cluster creation

After creating the histogram, the required number of clusters can be created by the horizontal line (see the dashed line in the picture).

Program: see PrgsKNIME on web: `Task06_Hierarchical_Clustrig`

Another approach is **divisive clustering.** Here we start with one cluster containing all points and it is subsequently divided until all points are clusters. However, this task is *np*-hard and is solved in an heuristic way.

6 Classical classification

K-nearest neighbour

The task: Classify a newly measured data record into one of the created components.

The procedure of classification is the following:

1. Compute the distance of the new point y from all points from $x_i \in X$.
2. Mark k points x_i, $i = 1, 2, \cdots, k$ nearest to y.
3. Assign y to the cluster to which majority of the nearest points belongs.

Remark: If there are more than one such cluster, take the first of them.

Program: see PrgsKNIME on web: `Task07_k-NearNeighb`

Decision trees

The task: In the learning dataset $X = \{[x_1, x_2, \cdots, x_n]_t\}_{t=1}^{T}$ with the pointer variable c_t (learning with a teacher) construct a classification tree and then perform classification for new data records.

The tree describes subsequent division of the remaining data according to the values of selected variables.

Example

Let us have the following data

t	x_1	x_2	c
1	*1*	*1*	*1*
2	*1*	*2*	2
3	*2*	*1*	1
4	*2*	*2*	*2*

where x_1, x_2 are the data records and c is the pointer variable.

For the selected variable x_1 we obtain two tables

$x_1 = 1$	
x_2	c
1	1
2	2

and

$x_1 = 2$	
x_2	c
1	1
2	2

As the assignment $x_1 \to c$ is not unique, we continue

$x_1 = 1$	
$x_2 = 1$	$x_2 = 2$
$c = 1$	$c = 2$

and

$x_1 = 2$	
$x_2 = 1$	$x_2 = 2$
$c = 1$	$c = 2$

As there is no other variable, we end. The tree is unique (the decision will be deterministic). E.g. for $x = [2, 1]$ we start at the top: $x_1 = 2$ points at the right part of the tree. Then $x_2 = 1$ leads us to the left part of the branch. Down we obtain $c = 1$. So, x is classified to the class 1.

Remark

If we start the tree from x_2, we are ready in one step. **The order of selection of the variables matters.**

Program: see PrgsKNIME on web: `Task08_Decision_Tree`

Tree in KNIME

For the above example

t	x_1	x_2	c
1	*1*	*1*	*1*
2	*1*	*2*	2
3	*2*	*1*	1
4	*2*	*2*	*2*

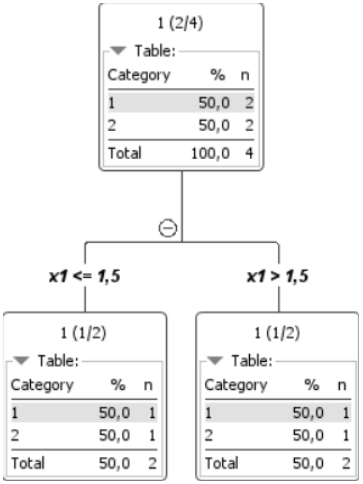
we have the **Tree 1**

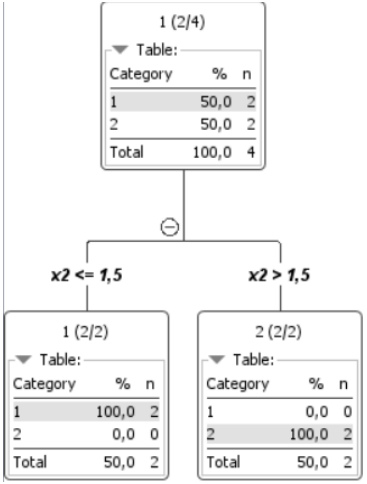
and for the extended data

t	x_1	x_2	c
1	*1*	*1*	*1*
2	*1*	*2*	2
3	*2*	*1*	1
4	*2*	*2*	*2*
5	1	1	1
6	1	1	2

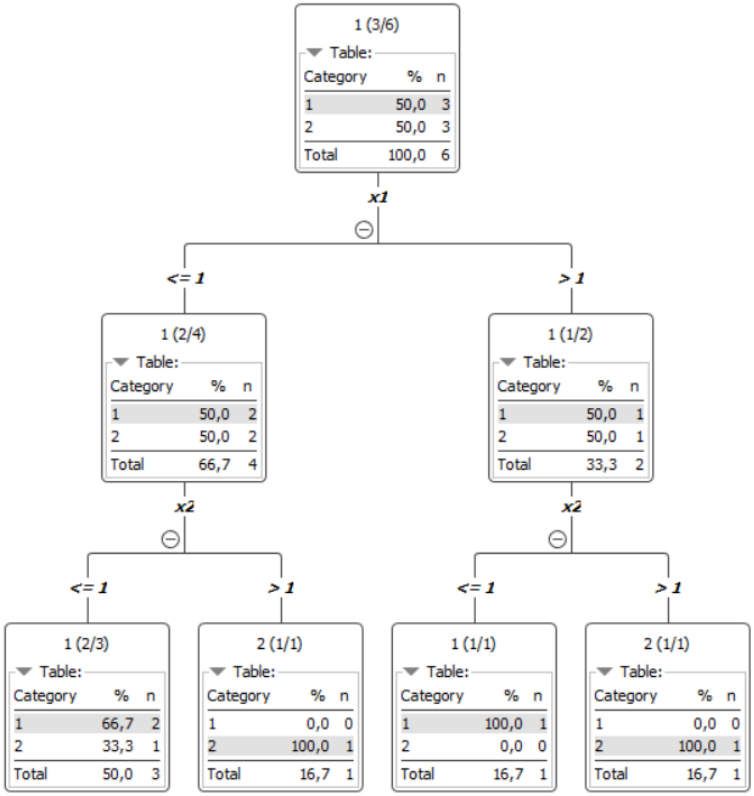
we get the **Tree 2**

Tree 1 $x_1,\ x_2$ $x_2,\ x_1$





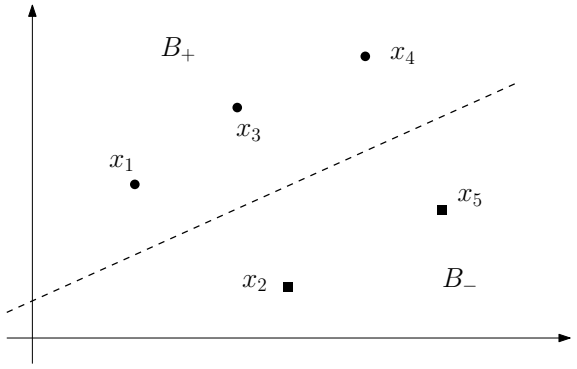
Tree 2


1 (3/6)
Table:
Category % n
1 50,0 3
2 50,0 3
Total 100,0 6
x1
<= 1
> 1
1 (2/4)
Table:
Category % n
1 50,0 2
2 50,0 2
Total 66,7 4
1 (1/2)
Table:
Category % n
1 50,0 1
2 50,0 1
Total 33,3 2
x2
x2
<= 1
> 1
<= 1
> 1
1 (2/3)
Table:
Category % n
1 66,7 2
2 33,3 1
Total 50,0 3
2 (1/1)
Table:
Category % n
1 0,0 0
2 100,0 1
Total 16,7 1
1 (1/1)
Table:
Category % n
1 100,0 1
2 0,0 0
Total 16,7 1
2 (1/1)
Table:
Category % n
1 0,0 0
2 100,0 1
Total 16,7 1

Support vector machines

Task: Linear classification into two classes. The goal is to find a hyperplane that optimally partitions the data points so that the training data belonging to different classes lie in opposite half-spaces. The optimal hyperplane is such that the value of the minimum of the distances of the points from the plane is as large as possible.

For two variables we have e.g. (dots one class, squares other one)



Program: see PrgsKNIME on web: `Task09_Support_Vec_Mach`

The derivation can be found in the Texts on web.