



# Naïve Bayes & kNN

---

Beatriz Sevilla Villanueva based on Tomas Aluja Slides  
and on Mario Martin Slides



# Contents

---

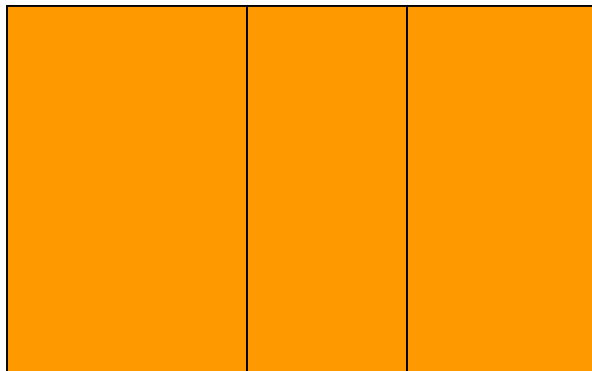
- ▶ Data
- ▶ Introduction
- ▶ Naive Bayes
- ▶ kNN
- ▶ Weighted kNN
- ▶ Distance Metrics

## ► Data in *Data Mining*:

- massive, secondary, non-random, with errors, missings ...

### Topics

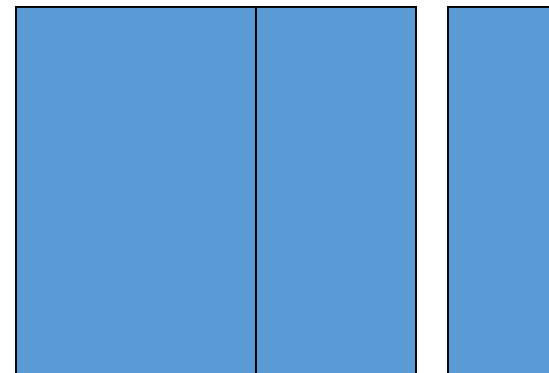
Sociodem.      Opinions      Products



Data to explore

Inputs

Output(s)

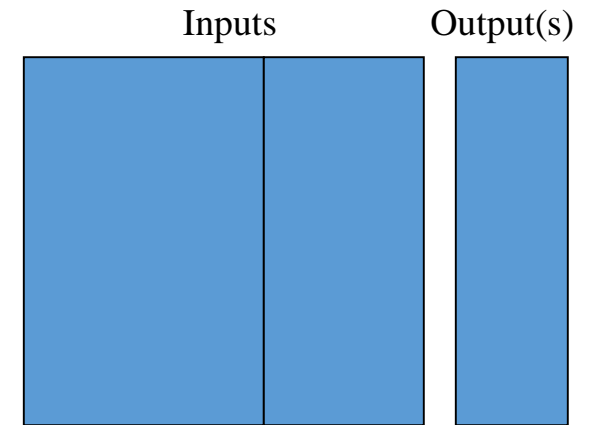


Data to modelize

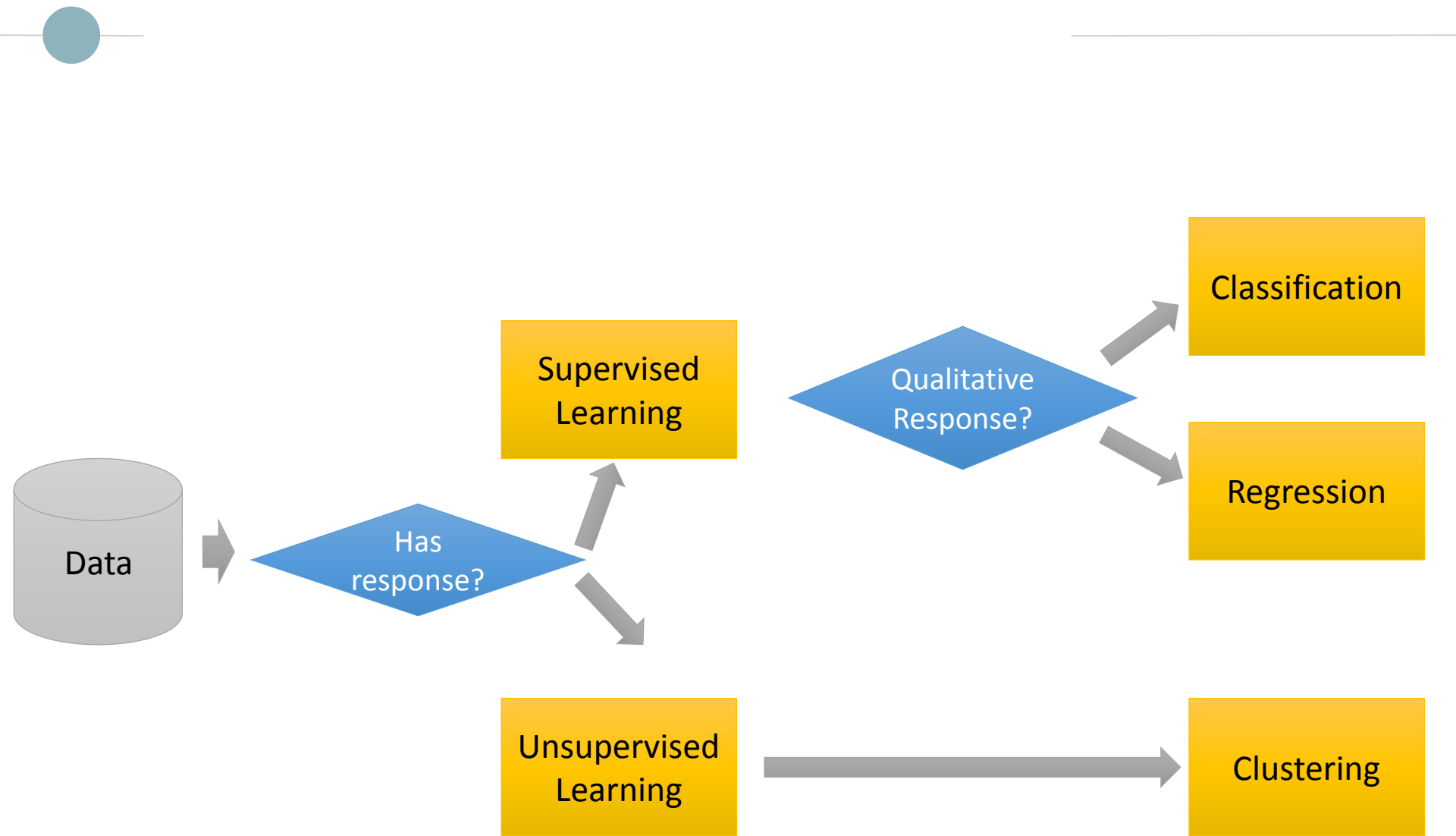
# Data

## ► Supervised Learning

- Input/Descriptors/Predictors
- Output/Response/Class
  - Binary
    - Classification
  - Qualitative or Categorical
    - Classification
  - Quantitative or Numerical
    - Regression



Data to modelize



# ● Introduction

---

- ▶ Data mining in supervised mode has a target column.
- ▶ We have an objective way to measure the success of classifiers / regressors:
  - Test new cases with the classifier / regressor and check predicted labels/values with reality.
- ▶ There are a lot of different measures to qualify success of a classifier.

# What is classification?

## Supervised:

Assigning one individual to a group (previously defined). Decisional goal

= Discriminant Analysis

= Pattern recognition ...

## Unsupervised:

Unveiling the true existing populations

Partitioning of a population

= Clustering

Definition of a classification rule (classifier):

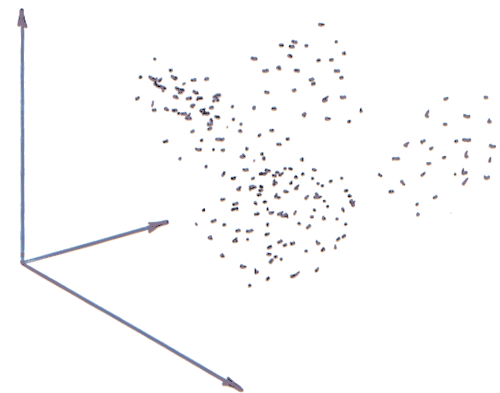
We need a training file  $X(n,p)$ , where for each individual we know to each class it belongs to.

Individuals  $x_i$  are points in  $\mathbb{R}^p$  space of variables (=features).

Response variable is categorical.

We want to partition the  $\mathbb{R}^p$  space into regions  $\Omega_k$ , corresponding each regions to one class (problem of *separability* versus *random fluctuation*)

One individual belongs to Class  $C_k$  if  $x_i \in \Omega_k$



# Types of classifiers

## ***Non parametric:***

(without prob. hypothesis)

*KNN, Kernel estimation of  $f(x/k)$*

## ***Parametric***

Linear:

*LDA, Logistic regression, ...*

Non linear:

*QDA*

## ***Original feature space***

*or*

***Expanded feature space***, introducing transformations of variables: *quadratic terms, interactions, basis expansions, the kernel trick* (to achieve separability).





# Assessment of the classification rules

---

*Not trivial*

The most obvious measure: estimate the **error rate (= 1-accuracy)**

Every classification rule has its own missclassification probability ( except for the case of perfect separability)

We can gain separability by means of increasing the complexity of the model

## ***Confussion matrix***

*Error rate*

*Recall*

*Precision*

*F-measure*

*AUC*

...



# Classifiers

---

## ► Simple algorithms but effective

- ***Naïve Bayes***

- Parametric: Builds a probabilistic model of your data following some assumptions.

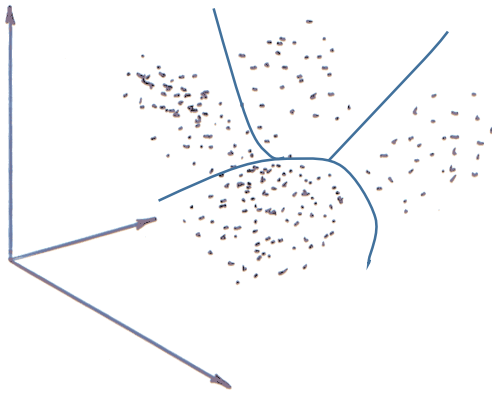
- ***k-NN***

- Non parametric method: In this case a lazy Instance Based Learning method that does not build any model.



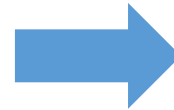
# Naive Bayes

# The four probabilities



$$f(k) = \pi_k$$

$$f(x/k)$$



$$f(k/x) = \frac{\pi_k f(x/k)}{f(x)}$$

$$f(x) = \sum_{k=1}^K \pi_k f(x/k)$$



Bayes formula (Thomas Bayes, 1701-1761, presbyterian priest)  
computing the probability of causes after having indirect evidences of facts

## Bayesian classifier

A region  $\Omega_k$  is defined by all points of  $\mathbb{R}^p$  with  $\max f(k/x)$

Hence the objective is to estimate  $f(k/x)$  for  $k=1..K$  in every point  $x$  of  $\mathbb{R}^p$

# ● The optimal rule

**Bayes rule:** Assign every point  $x$  of  $\mathbb{R}^p$  to the  $\operatorname{argmax}_k f(k/x)$

It leads to the definition of **decision surfaces**

*Important property: It minimises the missclassification error*

## **Discriminant function**

$$f(k / x) \equiv \pi_k f(x / k) \equiv \delta(\pi_k f(x / k)) = \delta_k(x)$$

Every monotonic function of the posterior probability

Assign every point  $x$  of  $\mathbb{R}^p$  to the  $\operatorname{argmax}_k \delta_k(x)$

# Naïve Bayes

*Data:*

$$\begin{matrix} & X_1 & \dots & X_M \\ \begin{matrix} i_1 \\ \vdots \\ i_n \end{matrix} & \begin{pmatrix} x_{11} & \dots & x_{1M} \\ \vdots & \ddots & \vdots \\ x_{n1} & \dots & x_{nM} \end{pmatrix} \end{matrix}$$

1. For learning, with the examples, we estimate the likelihood of our data:

$$p(i_i | c_j) = p(x_{i1}, \dots, x_{iM} | c_j)$$

that means, probability that observation  $i_i$  with the M values of the M variables  $(x_{i1}, x_{i2} \dots x_{iM})$  belongs to class  $c_j$

2. But for classifying, given an observation  $i_{i'}$   $(x_{i'1}, x_{i'2} \dots x_{i'M})$ , we look for the class that maximize the *a posterior* probability of belonging to the class:

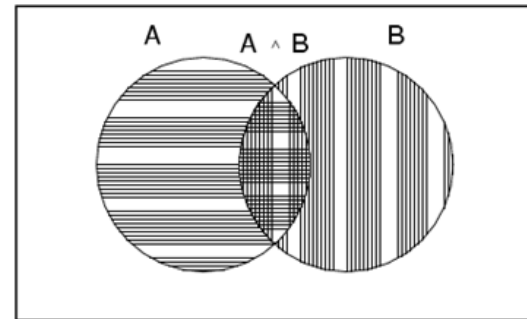
$$c_{MAP} = \operatorname{argmax}_{c_j \in C} P(c_j | x_{i'1}, x_{i'2} \dots x_{i'M})$$

# Naïve Bayes Classifier

- Bayes' theorem is used:

$$c_{MAP} = \operatorname{argmax}_{c_j \in C} P(c_j | x_{i'1}, x_{i'2} \dots x_{i'M}) = \operatorname{argmax}_{c_j \in C} \frac{P(x_{i'1}, x_{i'2} \dots x_{i'M} | c_j) P(c_j)}{P(x_{i'1}, x_{i'2} \dots x_{i'M})}$$

- Bayes' theorem



$$\begin{cases} P(A|B) = P(A \cap B) / P(B) \\ P(B|A) = P(A \cap B) / P(A) \end{cases}$$

$$\Rightarrow P(A \cap B) = P(A|B) P(B) = P(B|A) P(A)$$

$$\Rightarrow P(A|B) = \frac{P(B|A) P(A)}{P(B)}$$

# Computing Probabilities

►  $P(c_j)$  - Proportion of elements in class j

►  $P(x_{i'_1}, x_{i'_2} \dots x_{i'_M} | c_j)$

- Problem  $|X|^M |C|$  parameters!
- It can only be estimated from huge datasets.
  - Impractical

► Solution: **Independence assumption** (very Naïve)

- Variables values are independent. So in this case, we can easily compute:

$$P(x_{i'_1}, x_{i'_2} \dots x_{i'_M} | c_j) = \prod_{k=1..M} P(x_{i'_k} | c_j)$$



# Computing probabilities

## ► $P(x_{i'k}|c_j)$

- Only needed  $M|C|$  probability estimations
- Very easy. Number of values with property  $X_k$  in class  $c_j$  over the complete number of cases in class  $c_j$

► Solving  $P(x_{i'1}, x_{i'2} \dots x_{i'M}|c_j) = \prod_{k=1..M} P(x_{i'k}|c_j)$  the class assigned to a new observation is :

$$c_{NB} = \operatorname{argmax}_{c_j \in C} \frac{P(x_{i'1}, x_{i'2} \dots x_{i'M}|c_j)P(c_j)}{P(x_{i'1}, x_{i'2} \dots x_{i'M})} = \operatorname{argmax}_{c_j \in C} P(c_j) \prod_{k=1..M} P(x_{i'k}|c_j)$$

$P(x_{i'1}, x_{i'2} \dots x_{i'M})$

Constant for all  
classes

$\operatorname{argmax}_{c_j \in C} P(c_j) \prod_{k=1..M} P(x_{i'k}|c_j)$

Equation to solve



## Example: Learning to classify texts

---

- ▶ Training set:  $X$  document corpus
- ▶ Each document is labeled with  $f(x)=like/dislike$
- ▶ Goal: Learn function that permits given new document if you like it or not.
- ▶ Questions:
  - How do we represent documents?
  - How to compute probabilities?



# Example: Learning to classify texts

---

## ► How do we represent documents?

- Each document is represented as a *Bag of Words*
- Variables: All words that appear in the document
- So each document is represented as a Boolean vector with length  $N$ :
  - 0 – word does not appear ;
  - 1 – word appear

► Practical problem: A very huge table.

► Solution : Use sparse representation of matrixes



# Example: Learning to classify texts

---

## ► Some numbers

- 10.000 documents
- 500 words per document
- Maximum theoretical number of words: 50.000 (much less because of word repetitions)

## ► Reducing the number of variables

- Removing the number (sing/plural) and verbal forms (*stemming*)
- Remove conjunctions, propositions and articles (*stop words*)
- Now we have about. 10.000 variables

# Example: Learning to classify texts

- ▶ How to compute probabilities?

- ▶ For each word

$$V_{NB} = \operatorname{argmax}_{v \in \{\text{like}, \text{dislike}\}} P(v) \prod_i p(x_i = \text{word}_i | v)$$

- ▶ “a priori” probability for like and dislike classes **P(v<sub>i</sub>)**

$$P(v_{\text{like}}) = \frac{\text{\textit{\#documents like}}}{\text{\textit{total number of documents}}}$$

$$P(v_{\text{dislike}}) = \frac{\text{\textit{\#documents dislike}}}{\text{\textit{total number of documents}}}$$



## Example: Learning to classify texts

---

$$V_{NB} = \operatorname{argmax}_{v \in \{\text{like}, \text{dislike}\}} P(v) \prod_i p(x_i = \text{word}_i | v)$$

- ▶ Number of parameters  $P(x_i = \text{word}_i | v)$ 
  - 10.000 words and 2 classes (so about 20000)

$$P(\text{word}_k | v) = \frac{\text{\# docs } v \text{ in training where word}_k \text{ appears}}{\text{\# documents } v} = \frac{n_{v_k}}{n_k}$$

# Example: Learning to classify texts

## ► Problem:

$$P(\text{word}_k|v) = \frac{\text{\# docs } v \text{ in training where word}_k \text{ appears}}{\text{\# documents } v} = \frac{n_{v_k}}{n_k}$$

- When  $n_{v_k}$  is low, not an accurate probability
- when  $n_{v_k}$  is 0 for  $\text{word}_k$  for one class  $v$ , then any document with that word will never be assigned to  $v$  (independent of other appearing words)

## Example: Learning to classify texts

- Solution: More robust computation of probabilities (Laplace *smoothing*)

$$P(word_k | v) = \frac{n_{v_k+mp}}{n_v + m}$$

Where:

- $n_{v_k}$  is # of documents of class  $v$  in which word  $k$  appear
- $n_v$  is # of documents with label  $v$
- $p$  it's a likelihood estimation of “a priori”  $P(x_k | v)$  (f.i., uniform distribution)
- $m$  is the number of labels



# Example: Learning to classify texts

## ► *Smoothing*

$$P(x_k | v) = \frac{n_{v_k} + mp}{n_v + m}$$

More common “a priori” uniform distribution:

1. When two classes:  $p=1/2$ ,  $m=2$  (Laplace Rule)

$$P(x_k | v) = \frac{n_{v_k} + 1}{n_v + 2}$$

1. Generic case ( $c$  classes):  $p = 1/c$ ,  $m=c$

$$P(x_k | v) = \frac{n_{v_k} + 1}{n_v + c}$$



## Example: Learning to classify texts

---

- ▶ Naïve Bayes return good accuracy results even when independence assumption is not fulfilled
  - In fact, Spam/not Spam implementation of Thunderbird work in this way
  - Applied to document filtering (ex. *Newsgroups* or *incoming mails*)
- ▶ Learning and testing time are linear with the number of variables!

# Extension to Continuous Variables

- Assume each class follows a normal distribution for each variable:

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- For instance, with an average of 73 for the variable temperature in class x, and std=26.2, the conditional probability in the following:

$$p(\text{temperature} = 66 \mid x) = \frac{1}{\sqrt{2\pi}26.2} e^{-\frac{(66-73)^2}{2 \cdot 26.2^2}} = 0.034$$

# ● Example: Sex classification

## ► Training set:

Person	height (feet)	weight (lbs)	foot size(inches)
male	6	180	12
male	5.92 (5'11")	190	11
male	5.58 (5'7")	170	12
male	5.92 (5'11")	165	10
female	5	100	6
female	5.5 (5'6")	150	8
female	5.42 (5'5")	130	7
female	5.75 (5'9")	150	9

## ► Mean and variance assuming Gaussian distribution:

Person	mean (height)	variance (height)	mean (weight)	variance (weight)	mean (foot size)	variance (foot size)
male	5.855	$3.5033 \times 10^{-2}$	176.25	$1.2292 \times 10^2$	11.25	$9.1667 \times 10^{-1}$
female	5.4175	$9.7225 \times 10^{-2}$	132.5	$5.5833 \times 10^2$	7.5	1.6667

# Example: Sex classification

► Training set:

Person	height (feet)	weight (lbs)	foot size(inches)
male	6	180	12
male	5.92 (5'11")	190	11
male	5.58 (5'7")	170	12
male	5.92 (5'11")	165	10
female	5	100	6
female	5.5 (5'6")	150	8
female	5.42 (5'5")	130	7
female	5.75 (5'9")	150	9

► Mean and variance assuming Gaussian distribution:

Person	mean (height)	variance (height)	mean (weight)	variance (weight)	mean (foot size)	variance (foot size)
male	5.855	$3.5033 \times 10^{-2}$	176.25	$1.2292 \times 10^2$	11.25	$9.1667 \times 10^{-1}$
female	5.4175	$9.7225 \times 10^{-2}$	132.5	$5.5833 \times 10^2$	7.5	1.6667

► New instance to classify

Person	height (feet)	weight (lbs)	foot size(inches)
sample	6	130	8

► Determine which posterior probability is greater, male or female

$$\text{posterior (male)} = \frac{P(\text{male}) p(\text{height} \mid \text{male}) p(\text{weight} \mid \text{male}) p(\text{foot size} \mid \text{male})}{\text{evidence}}$$

$$\text{posterior (female)} = \frac{P(\text{female}) p(\text{height} \mid \text{female}) p(\text{weight} \mid \text{female}) p(\text{foot size} \mid \text{female})}{\text{evidence}}$$

$$\begin{aligned} \text{evidence} = & P(\text{male}) p(\text{height} \mid \text{male}) p(\text{weight} \mid \text{male}) p(\text{foot size} \mid \text{male}) \\ & + P(\text{female}) p(\text{height} \mid \text{female}) p(\text{weight} \mid \text{female}) p(\text{foot size} \mid \text{female}) \end{aligned}$$

# Example: Sex classification

## ► Training set:

Person	height (feet)	weight (lbs)	foot size(inches)
male	6	180	12
male	5.92 (5'11")	190	11
male	5.58 (5'7")	170	12
male	5.92 (5'11")	165	10
female	5	100	6
female	5.5 (5'6")	150	8
female	5.42 (5'5")	130	7
female	5.75 (5'9")	150	9

## ► Mean and variance assuming Gaussian distribution:

Person	mean (height)	variance (height)	mean (weight)	variance (weight)	mean (foot size)	variance (foot size)
male	5.855	$3.5033 \cdot 10^{-2}$	176.25	$1.2292 \cdot 10^2$	11.25	$9.1667 \cdot 10^{-1}$
female	5.4175	$9.7225 \cdot 10^{-2}$	132.5	$5.5833 \cdot 10^2$	7.5	1.6667

## ► New instance to classify

Person	height (feet)	weight (lbs)	foot size(inches)
sample	6	130	8

## ► Determine which posterior probability is greater, male or female

$$\text{posterior (male)} = \frac{P(\text{male}) p(\text{height} \mid \text{male}) p(\text{weight} \mid \text{male}) p(\text{foot size} \mid \text{male})}{\text{evidence}}$$

$$P(\text{male}) = 0.5$$

$$p(\text{height} \mid \text{male}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(6 - \mu)^2}{2\sigma^2}\right) \approx 1.5789,$$

$$p(\text{weight} \mid \text{male}) = 5.9881 \cdot 10^{-9}$$

$$p(\text{foot size} \mid \text{male}) = 1.3112 \cdot 10^{-3}$$

$$\text{posterior numerator (male)} = \text{their product} = 6.1984 \cdot 10^{-9}$$

# Example: Sex classification

► Training set:

Person	height (feet)	weight (lbs)	foot size(inches)
male	6	180	12
male	5.92 (5'11")	190	11
male	5.58 (5'7")	170	12
male	5.92 (5'11")	165	10
female	5	100	6
female	5.5 (5'6")	150	8
female	5.42 (5'5")	130	7
female	5.75 (5'9")	150	9

► Mean and variance assuming Gaussian distribution:

Person	mean (height)	variance (height)	mean (weight)	variance (weight)	mean (foot size)	variance (foot size)
male	5.855	$3.5033 \cdot 10^{-2}$	176.25	$1.2292 \cdot 10^2$	11.25	$9.1667 \cdot 10^{-1}$
female	5.4175	$9.7225 \cdot 10^{-2}$	132.5	$5.5833 \cdot 10^2$	7.5	1.6667

► New instance to classify

Person	height (feet)	weight (lbs)	foot size(inches)
sample	6	130	8

► Determine which posterior probability is greater, male or female

$$\text{posterior (female)} = \frac{P(\text{female}) p(\text{height} \mid \text{female}) p(\text{weight} \mid \text{female}) p(\text{foot size} \mid \text{female})}{\text{evidence}}$$

$$P(\text{female}) = 0.5$$

$$p(\text{height} \mid \text{female}) = 2.2346 \cdot 10^{-1}$$

$$p(\text{weight} \mid \text{female}) = 1.6789 \cdot 10^{-2}$$

$$p(\text{foot size} \mid \text{female}) = 2.8669 \cdot 10^{-1}$$

$$\text{posterior numerator (female)} = \text{their product} = 5.3778 \cdot 10^{-4} > \text{posterior numerator (male)} = 6.1984 \cdot 10^{-9}$$

The posterior probability of female is greater => it predicts the sample is female!



# k-NN

---

- ▶ Probably the simplest classifier
- ▶ Simple idea: a new instance is labeled with label of its closest neighbour/s.
- ▶ No model is learnt or trained
  - Lazy learning: generalization of the training data is delayed until a query is made
    - defers the majority of computation to consultation time



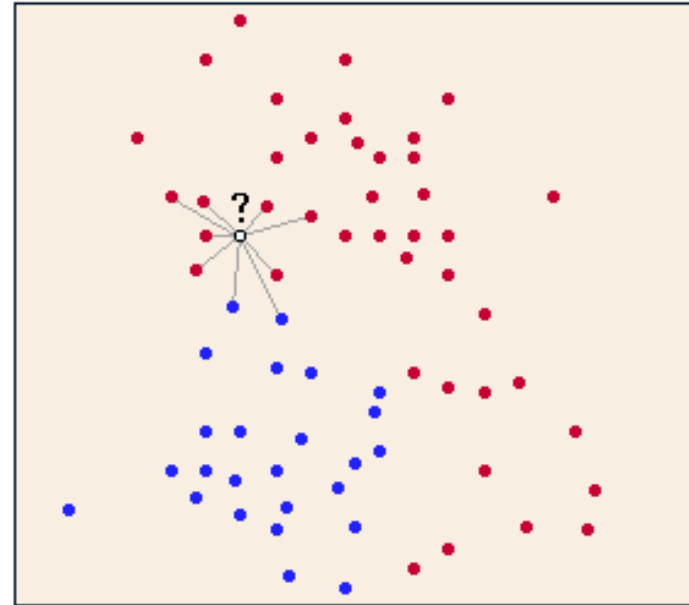
# ● k-NN origins

- ▶ k-nn (k nearest neighbours)
- ▶ First paper with the name:
  - [Cover TM, Hart PE \(1967\). "Nearest neighbor pattern classification" \(PDF\). IEEE Transactions on Information Theory. 13 \(1\): 21–27.](#)
  - That paper shows what happens if a very large aselectively chosen training set is used.
- ▶ Before Cover and Hart the rule was mentioned by [Nilsson \(1965\)](#) who called it “minimum distance classifier” and by [Sebestyen \(1962\)](#), who called it “proximity algorithm”. [Fix and Hodges in their very early discussion on non-parametric discrimination \(1951\)](#) already pointed to the NN rule as well.

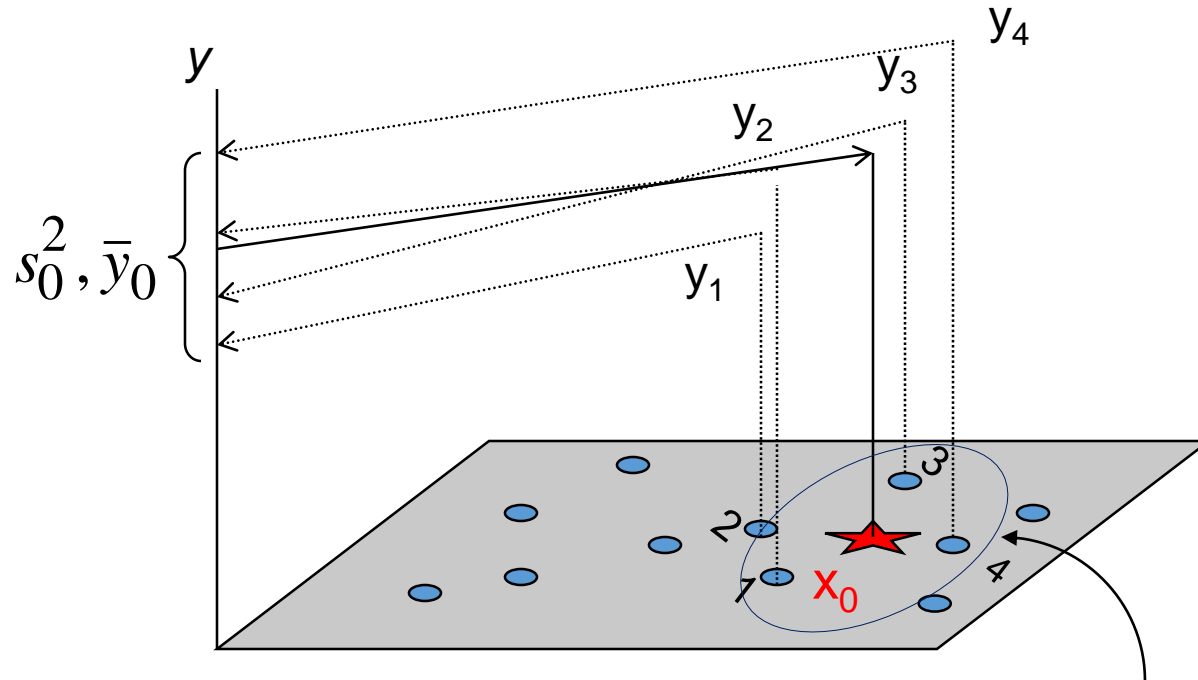
# Local Discrimination

## ► k-NN

- Universal Aproximator
- Requires a big dataset for training
- “*curse of dimensionality*”



# k-NN. Theoretical basis



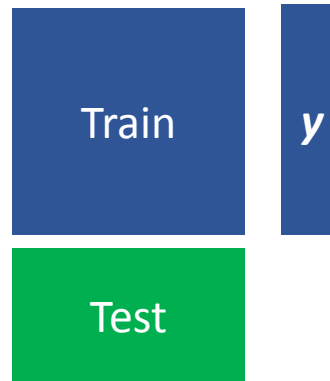
If  $K \rightarrow \infty$ ,  $K/n \rightarrow 0$

$V_0$  Neighbourhood of  $x_0$

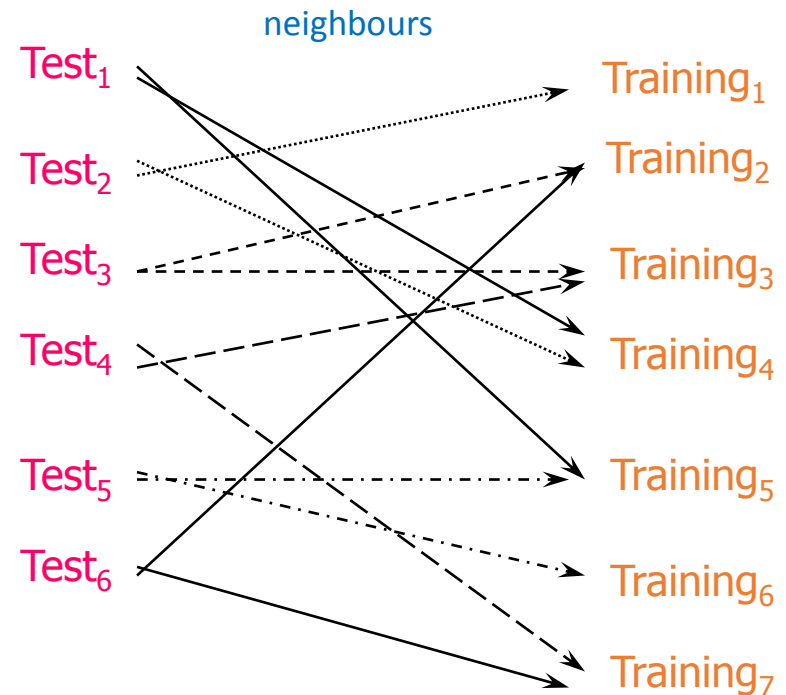
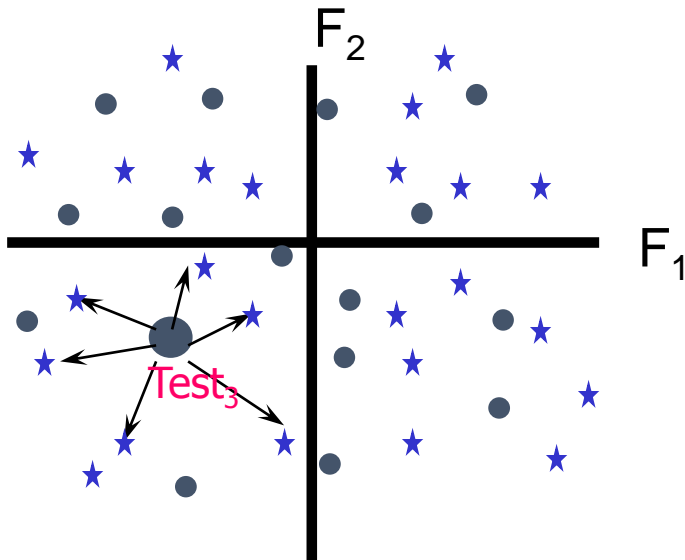
$$\bar{y}_0 \rightarrow E(y|x_0) \quad s_0^2 \rightarrow V(y|x_0) \quad (\text{numerical response})$$

$$\hat{f}(j|x) = \frac{k_j}{k \times V_0} \rightarrow f(j|x) \quad (\text{categorical response})$$

# Correspondence Graph of k-NN



`knn(train, test, y, k)`





## k-NN

---

- ▶ Non-parametric
- ▶ Lazy algorithm
- ▶ Instance-based Learning method
- ▶ Not builds any model
  
- ▶ Requeriments:
  - A training set
  - A similarity measure

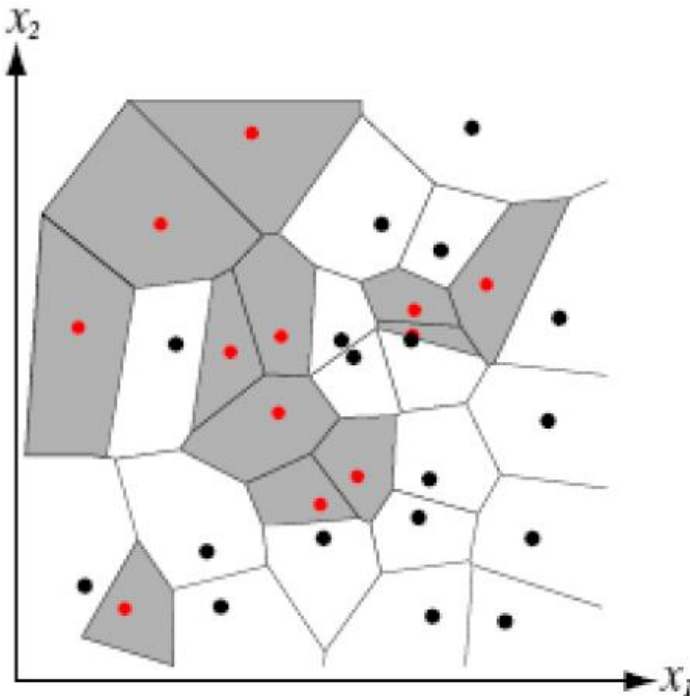


## ► K-Nearest neighbor algorithm

- It interprets each example as a point in a space defined by the features describing the data
- In that space a similarity measure allows as to classify new examples.
- Class is assigned depending on the  $K$  closest examples

# 1-NN Example

- ▶ Two real variables ( $x_1$ ,  $x_2$ ) define the space
- ▶ Each red point is a positive example. Black points are negative examples

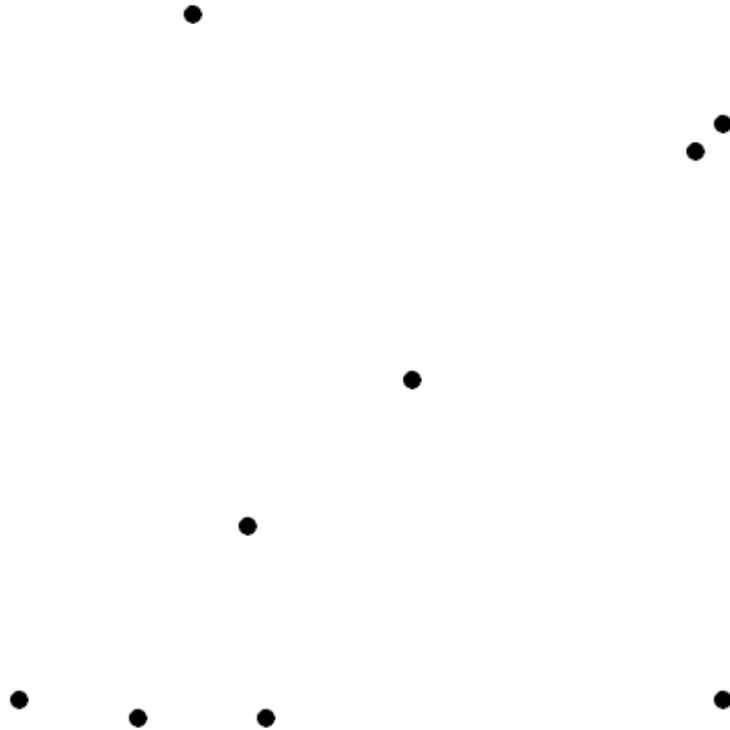


Equivalent to draw  
the Voronoi space of  
your data.



# Voronoi Space

---

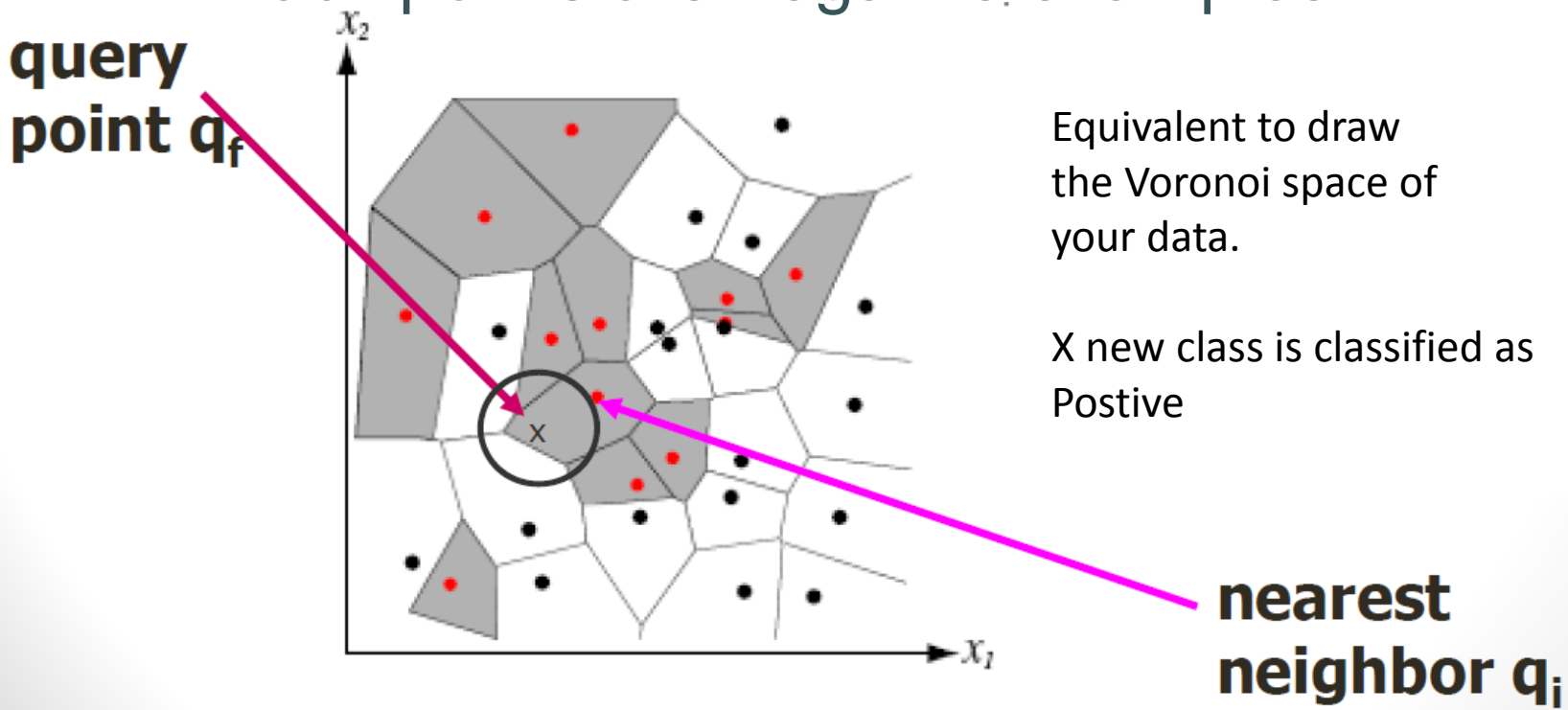




# 1-NN Example

- ▶ Two real variables ( $x_1$ ,  $x_2$ ) define the space
- ▶ Each red point is a positive example.

Black points are negative examples



# Distance Measure

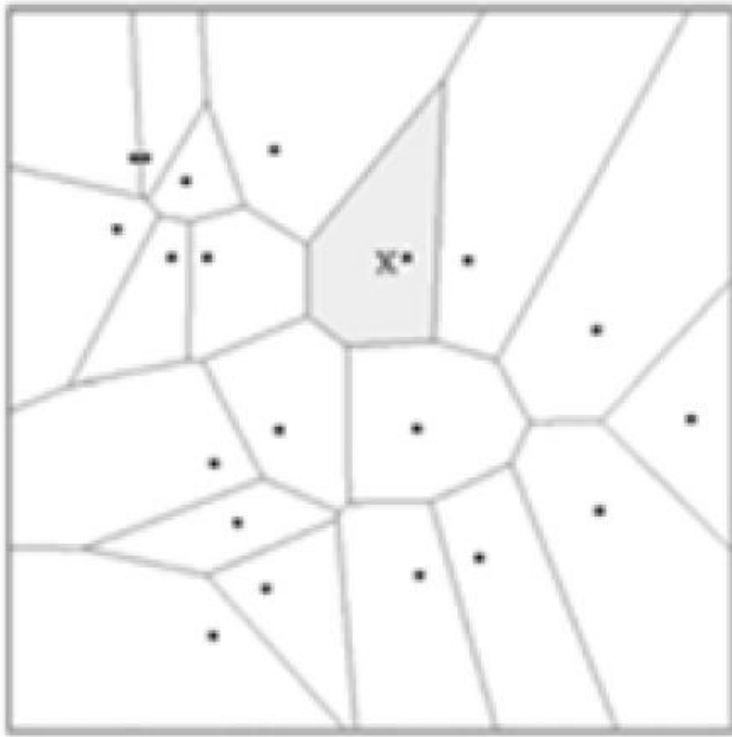
## ► Euclidean

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^M (x_{ik} - x_{jk})^2}$$

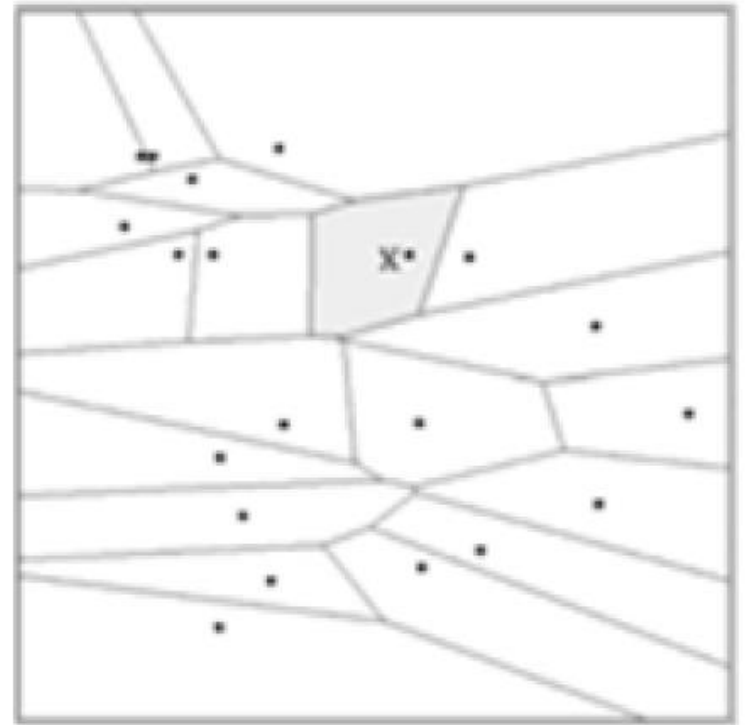
## ► Weighted Euclidean

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^M w_k (x_{ik} - x_{jk})^2}$$

## ● Weighting Effect



$$d^2(x_i, x_j) = (x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2$$



$$d^2(x_i, x_j) = 9(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2$$



## k-NN

---

### ► Advantages:

- Fast training
- Ability to learn very complex functions

### ► Problems:

- Very slow in testing. Needed some smart structure representation of data in trees
- Fooled by noise
- Fooled when irrelevant features

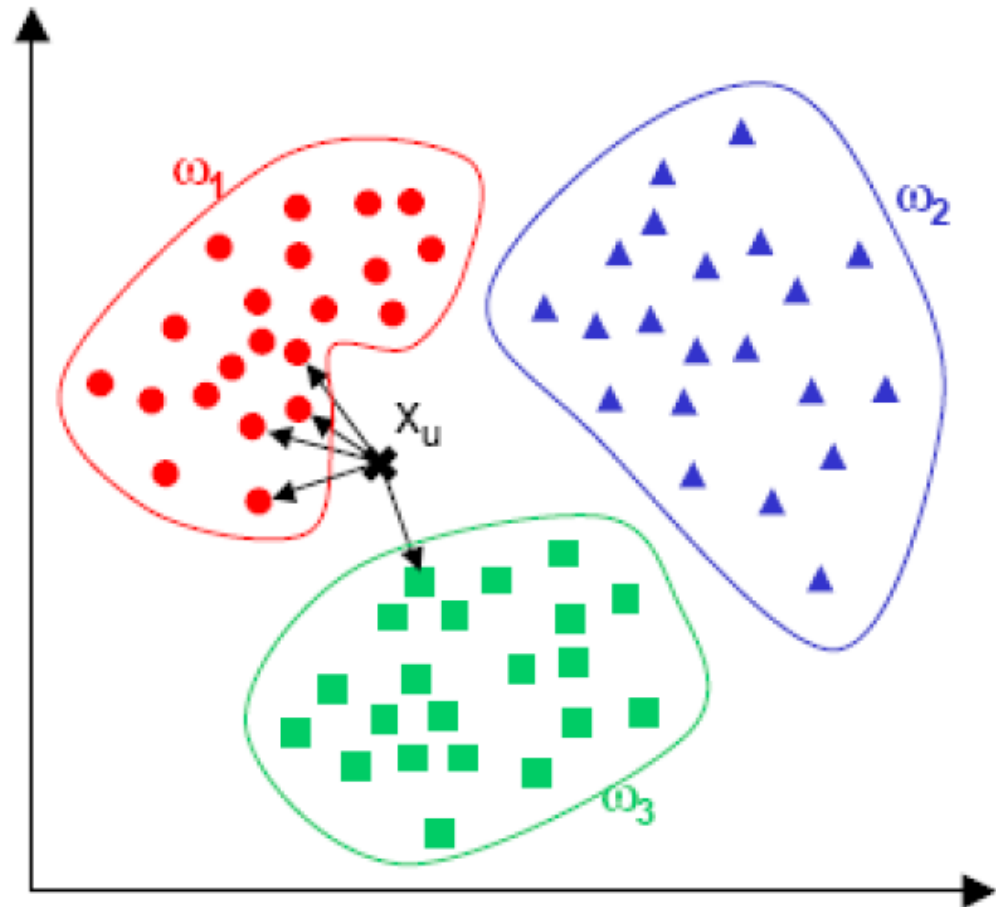


## ► Building more robust classifiers

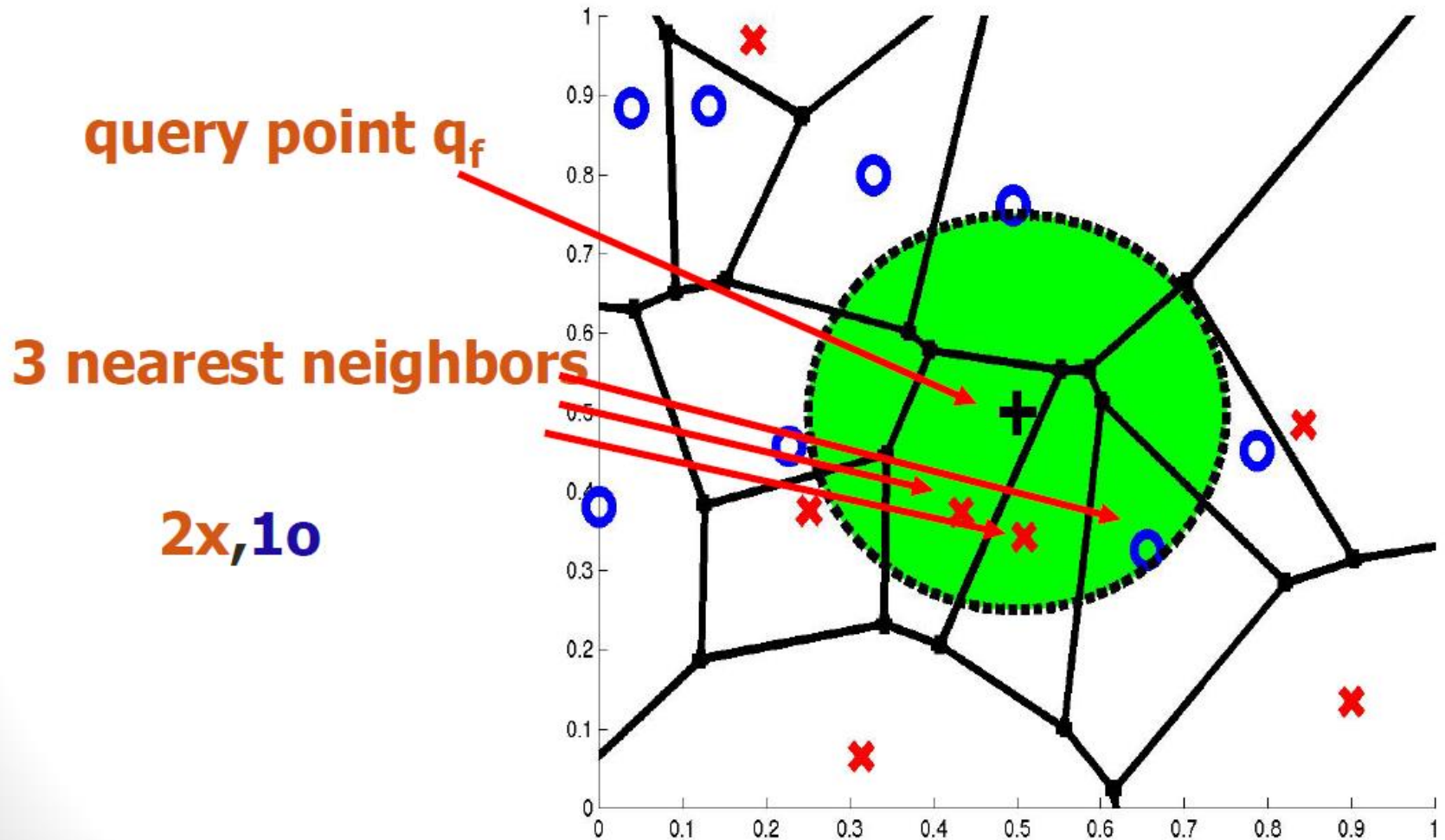
- Results do not depend on the closest example
- but on the  $k$  closest examples (so *k-nearest neighbours* (kNN) name)

## ● Example

► kNN;  $k = 5$



## 3-nearest neighbours

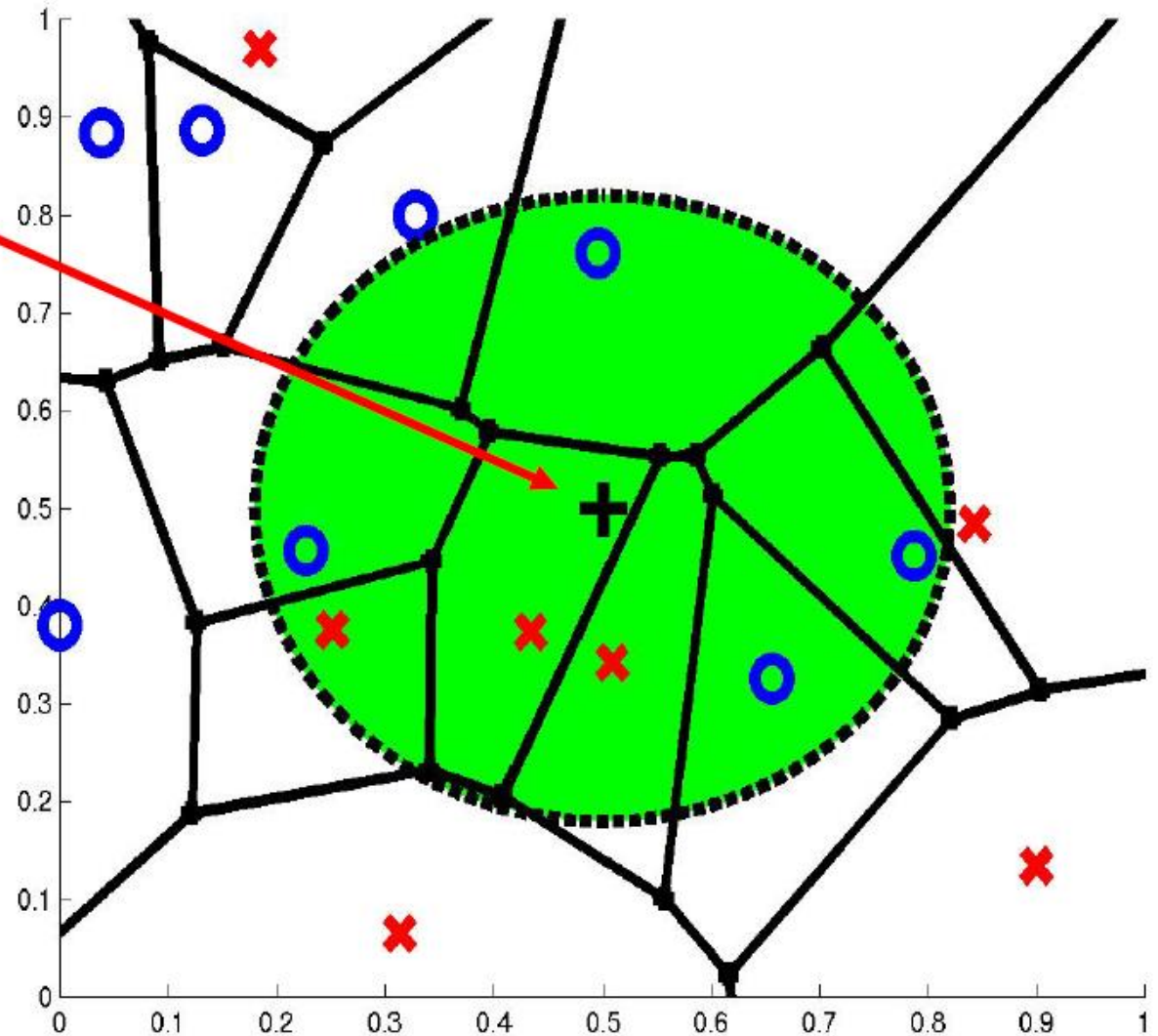


## 7-nearest neighbours

query point  $q_f$

7 nearest  
neighbors

3x, 4o





# k-NN Algorithm

---

## ► Parameters:

- Natural number  $k$  (*even number*)
- Training set
- Distance measure

## ► Algorithm:

1. Store all training set  $\langle i_i, \text{label}(i_i) \rangle$
2. Given new observation,  $i'$  compute the nearest  $k$  neighbors
3. Let vote the nearest  $k$  neighbors to assign the label to the new data.



## Drawbacks of k-NN

- ▶ The parameter that defines the complexity is the number of neighbours ***k***
- ▶ A higher ***k*** more bias
- ▶ A lower ***k*** more variance
- ▶ Problem with high dimensional data (*curse of dimensionality*)
- ▶ Problem when unbalanced classes (need of having balanced samples)
- ▶ Black hole effect. Tends to produce results a little worse than with other methods.

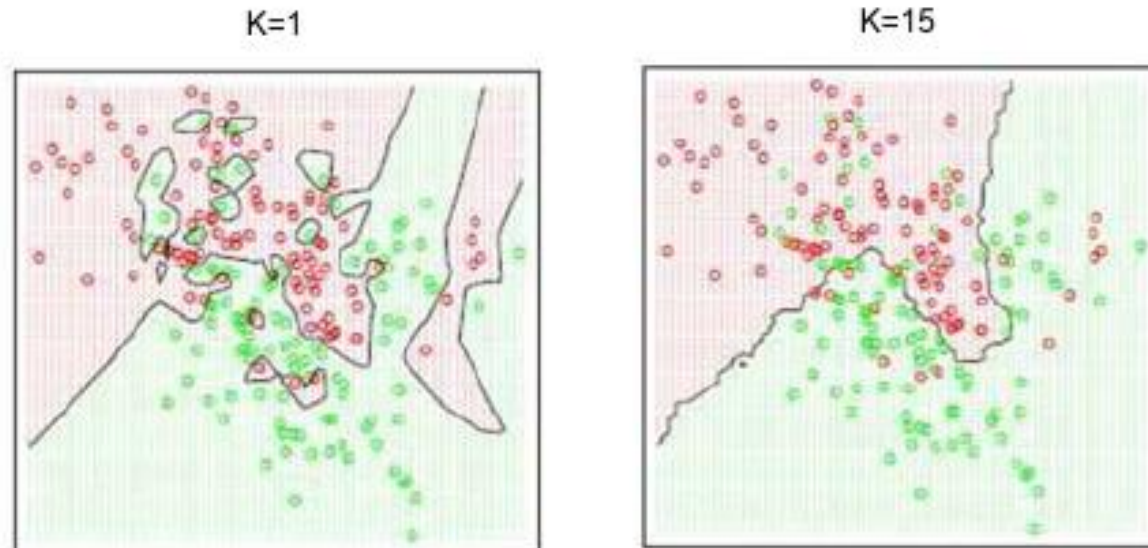
## ● How to select k?

---

- ▶ High number of k show two advantages:
  - Smother frontiers
  - Reduces sensibility to noise
- ▶ But too large values are bad because
  - We lose locality in the decision because very distant points can interfere in assigning labels
  - Computation time is increased
- ▶ K-value usually is chosen by cross-validation

## Decision boundary defined from data

### Effect of K



Figures from Hastie, Tibshirani and Friedman (Elements of Statistical Learning)

Larger  $k$  produces smoother boundary effect and can reduce the impact of class label noise.

But when  $k$  is too large, say  $k=N$ , we always predict the majority class

## ● Distance Weighted k-NN

---

- ▶ A smart variation of k-NN.
- ▶ When voting, all k neighbors have the same influence, but some of them are more distant than the others (so they should influence less in decisions)
- ▶ Solution: Given more weight to closest examples

## Distance Weighted kNN

- Label is assigned by the weighted addition:

$$f_l = \sum_i^k w_i l(x_i)$$

where  $l(x_i) \in \{-1, 1\}$  the label of example  $x_i$

$w_i = K(d(x_q, x_i))$  is the weight of example  $i$

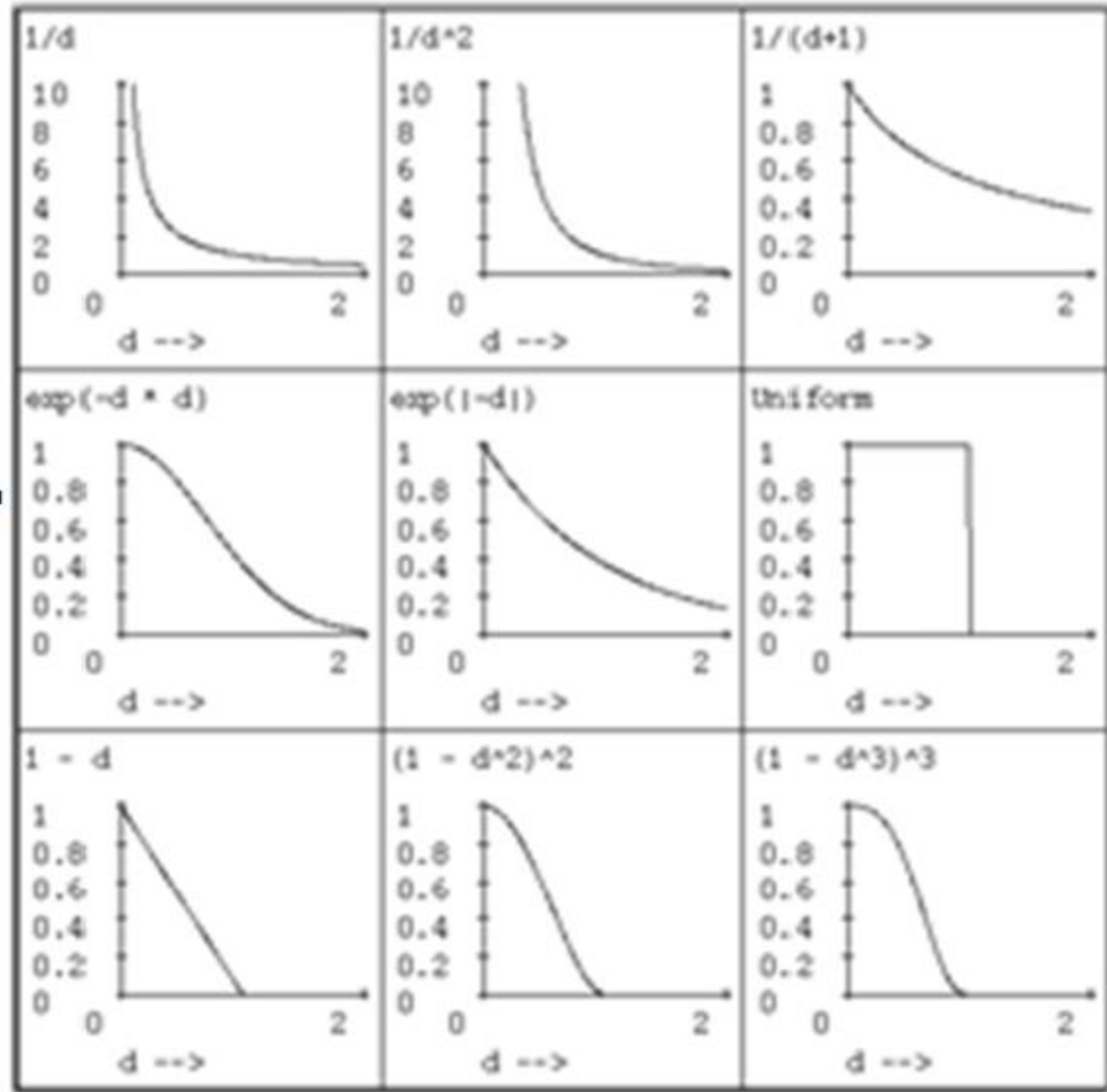
$K(d)$  is a weight functions depending on distance, and

$d(x_q, x_i)$  is distance from  $x_q$  to  $x_i$

# Distance Weighted kNN

$$d = d(x_i, x_{query})$$

Common  
Weighting  
Functions



## ● Problems with irrelevant features

- ▶ kNN is fooled when irrelevant features.
  - Lets assume examples described with 20 attributes, but only 2 are relevant to the classification
  - kNN will be lost easily in this case
- ▶ Solution: consist in feature removal:
  - Find weights  $z_1, \dots, z_n$ , one for each feature, that minimize error in a validation data set (so use cross-validation to choose weights)
  - Notice that setting  $z_j = 0$  means removing the feature

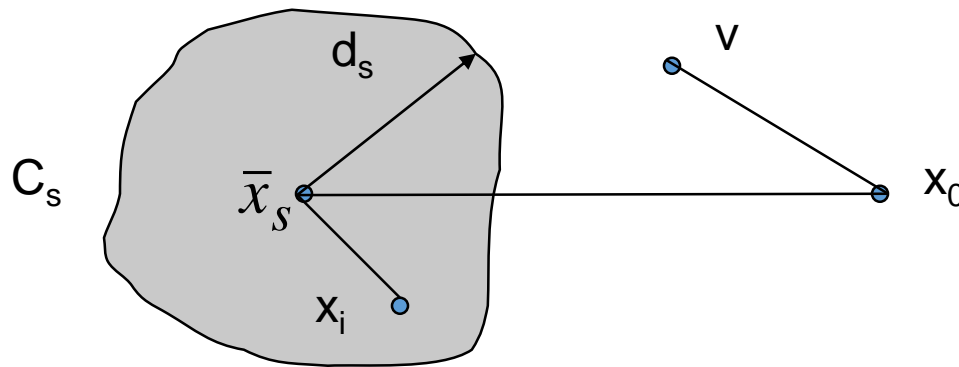




# Finding the k-NN

(Fukunaga&Narendra)

1. Classify the  $n$  individuals into a big number of classes (ex. using k-means)



$c_s$ : class  $s$   
 $\bar{x}_s$ : centroid of class  $c_s$   
 $d_s$ : diameter of class  $c_s$   
 $x_i$ : point in class  $c_s$   
 $v$ : point not in class  $c_s$   
 $x_0$ : point to classify

2. Algorithm branch&bound:

Discard of a class:

$$d(\bar{x}_s, x_0) - d_s > d(x_0, v)$$

Discarding individuals:

$$d(\bar{x}_s, x_0) - d(\bar{x}_s, x_i) > d(x_0, v)$$



# k-NN

---

## ► Advantages:

- Fast training
- Ability to learn very complex functions
- Incremental
  - Hability of adaptation to new cases

## ► Problems:

- Very slow in testing. Needed some smart structure representation of data in trees
- Fooled by noise
- Fooled when irrelevant features



# Management of heterogeneous data matrices

---

- ▶ **Heterogeneous matrices faced. Several approaches [Anderberg 73]:**
  - **Variables partitioning**
  - **Variables converting**
  - **Compatibility measures**
    - **Mixed metrics required**

# Distance between individuals

---

## ► Quantitative Variables

- Euclidean
- Manhattan
- Minkowski

## ► Qualitative Variables

- $\chi^2$  [Benzécri 80]

## ► Heterogeneous variables (compatibility measures):

- Gower [71]
- Gowda i Diday [91]
- Gibert's Mixed [91]
- Ichino i Yaguchi [94]
- Ralambondrainy [95]

# Distance for Numerical Variables

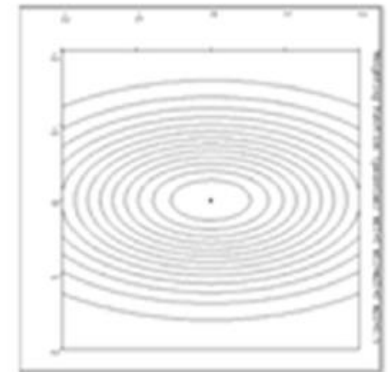
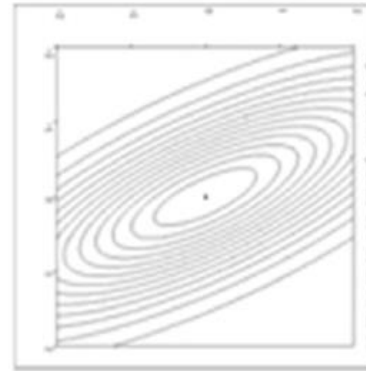
## ► *Minkowski*

$$d(x_i, x_j) = \sqrt[r]{\sum_{k=1}^M (x_{ik} - x_{jk})^r}$$

- $r = 1$  Manhattan
- $r = 2$  Euclidean

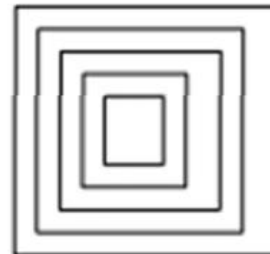
# Other Measures for Numerical

**Mahalanobis**

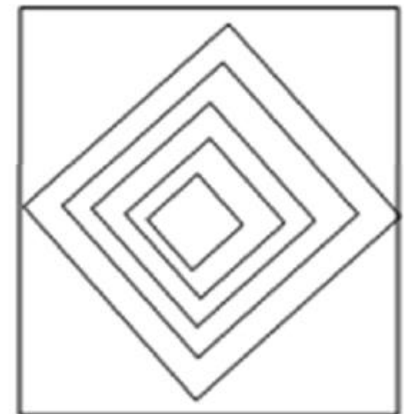


**Scaled Euclidian ( $L_2$ )**

**$L_{infinity}$  (max) norm**



**$L_1$  norm (absolute)**



## Distances for Qualitative Variables

### ► Chi-Square Distance

$$\chi^2 = \sum_{k=1}^p \sum_{j=1}^q \frac{\left(n_{kj} - \frac{n_k n_j}{n}\right)^2}{\frac{n_k n_j}{n}}$$

### ► Jaccard index (binary variables)

$$d(i, i') = \frac{n_{11} + n_{00}}{n}$$



## Mixed Metric - Gower

$$d_G(i, j) = \frac{\sum_k \delta_{ii'k} * d_k(i, i')}{\sum_k \delta_{ii'k}}$$

$$d_k(i, i') = \begin{cases} 0 & \text{if } X_k \text{ binary} \wedge x_{ik} = x_{i'k} = \text{TRUE} \\ 0 & \text{if } X_k \text{ categorical} \wedge x_{ik} = x_{i'k} \\ \frac{|x_{ik} - x_{i'k}|}{\max_k - \min_k} & \text{if } X_k \text{ es numerical} \\ \frac{|r_{ik} - r_{i'k}|}{\max_{r_k} - 1} & \text{if } X_k \text{ is ordered } (r_{ik} = \text{index of the category}) \\ 1 & \text{otherwise} \end{cases}$$

$$\delta_{ii'k} = \begin{cases} 0 & \text{if } x_{ik} = \text{MISSING} \text{ or } x_{i'k} = \text{MISSING} \\ 0 & \text{if } X_k \text{ assymetric binary} \wedge x_{ik} = x_{i'k} = 0 \text{ or } x_{ik} = x_{i'k} = \text{FALSE} \\ 1 & \text{otherwise} \end{cases}$$



# Mixed Metric - Gowda-Diday

## [Gow 91]

$$D(i, i') = \sum_{k=1}^K D_k(i_k, i'_k)$$
$$D_k(i, i') = D_p(i, i') + D_s(i, i') + D_c(i, i')$$

► Component Position

$$D_{kp}(i, i') = \begin{cases} \frac{|x_{ik} - x_{i'k}|}{R_k} & \text{if } (X_k \text{ numerical}) \text{ and } R_k \text{ is range of } X_k \\ 0 & \text{if } (X_k \text{ qualitative}) \end{cases}$$

► Component Span

$$D_{ks}(i, i') = \begin{cases} 0 & \text{if } X_k \text{ numerical} \\ 1 & \text{if } X_k \text{ qualitative and not multi-valued} \end{cases}$$

► Component Content

$$D_{kc}(i, i') = \begin{cases} 0 & \text{if } X_k \text{ numerical} \\ 0 & \text{if } (X_k \text{ qualitative}) \text{ and } (x_{ik} = x_{i'k}) \\ 1 & \text{if } (X_k \text{ qualitative}) \text{ and } (x_{ik} \neq x_{i'k}) \end{cases}$$

# Mixed Metric [Gibert 91]

$$d_{(\alpha,\beta)}^2(i, i') = \alpha d_{\zeta}^2(i, i') + \beta d_Q(i, i')$$

$$d_{\zeta}^2(i, i') = \sum_{\forall k \in \zeta} \frac{(x_{ik} - x_{i'k})^2}{s_k^2}$$

$$d_Q^2(i, i') = \begin{cases} 0 & \text{if } x_{ik} = x_{i'k} \\ \frac{1}{I_k^i} + \frac{1}{I_k^{i'}} & \text{otherwise, for compact } i \text{ and } i' \text{ w.r.t } X_k \\ \frac{(f_i^{k_s} - 1)^2}{I^{k_s}} + \sum_{j \neq s}^{n_k} \frac{(f_i^{k_j})^2}{I^{k_j}} & \text{if } x_{ik} = c_s^k, \text{ an extended } i' \text{ w.r.t } X_k \\ \sum_{j=1}^{n_k} \frac{(f_i^{k_j} - f_{i'}^{k_j})^2}{I^{k_j}} & \text{for } i, i' \text{ extended w.r.t } X_k \end{cases}$$

Proposal

$$\alpha = \frac{n_{\zeta}}{d_{\zeta}^2 \max^*}$$

$$\beta = \frac{n_Q}{d_Q^2 \max^*}$$

# Example: Sex classification

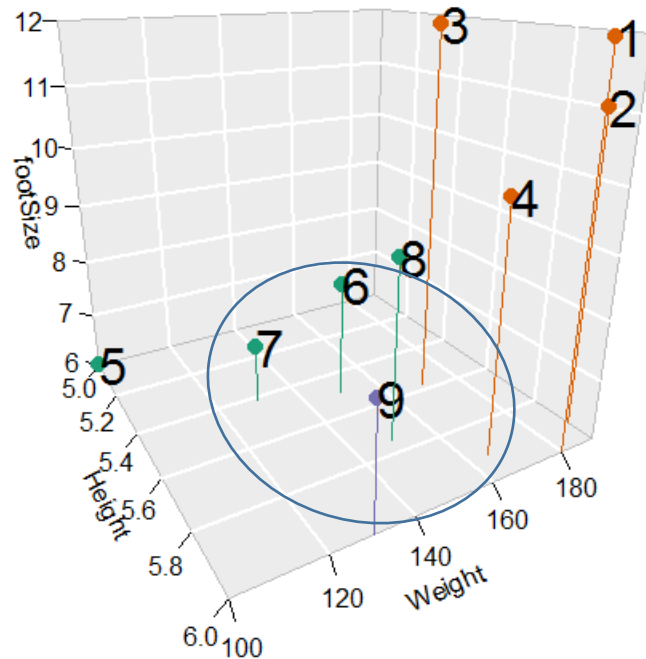
► Training set:

Person	height (feet)	weight (lbs)	foot size(inches)
male	6	180	12
male	5.92 (5'11")	190	11
male	5.58 (5'7")	170	12
male	5.92 (5'11")	165	10
female	5	100	6
female	5.5 (5'6")	150	8
female	5.42 (5'5")	130	7
female	5.75 (5'9")	150	9

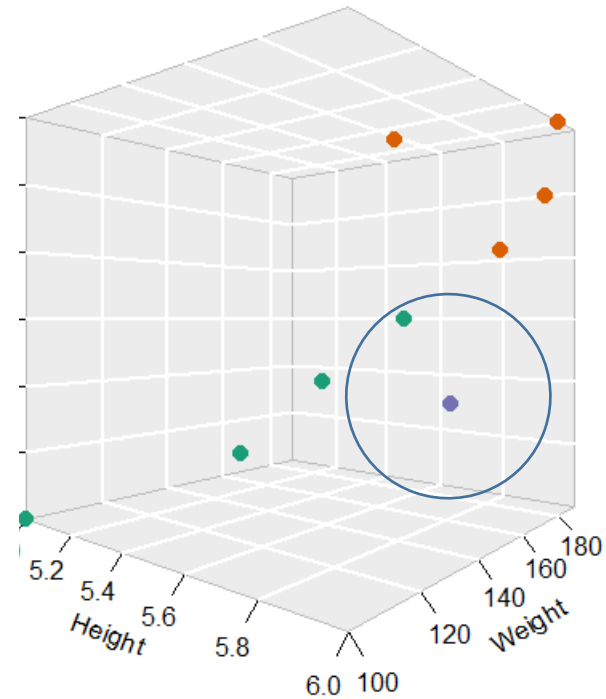
► New instance to classify

Person	height (feet)	weight (lbs)	foot size(inches)
sample	6	130	8

# Example: Sex classification



female male unknown



female male unknown

# ● Example: Sex classification

Person	height (feet)	weight (lbs)	foot size(inches)
sample	6	130	8

► 1 –  $NN$

► Class=female

	Class	Euclidean	Manhattan	Minkowski r=100	Gower
$i_1$	Male	50.16	54	50	0.41
$i_2$	Male	60.07	63.03	60	0.42
$i_3$	Male	40.20	44.42	40	0.51
$i_4$	Male	35.06	37.08	35	0.27
$i_5$	Female	30.08	33	30	0.56
$i_6$	Female	20.00	20.50	20	0.24
$i_7$	Female	<b>1.16</b>	<b>1.58</b>	<b>1</b>	0.25
$i_8$	Female	20.03	21.25	20	<b>0.22</b>

# Example: Sex classification

Person	height (feet)	weight (lbs)	foot size(inches)
sample	6	130	8

## ► 1 – $NN$

- Class=female

## ► 2-NN

- Class=female

	Class	Euclidean	Manhattan	Minkowski r=100	Gower
$i_1$	Male	50.16	54	50	0.41
$i_2$	Male	60.07	63.03	60	0.42
$i_3$	Male	40.20	44.42	40	0.51
$i_4$	Male	35.06	37.08	35	0.27
$i_5$	Female	30.08	33	30	0.56
$i_6$	Female	<b>20.00</b>	<b>20.50</b>	<b>20</b>	<b>0.24</b>
$i_7$	Female	<b>1.16</b>	<b>1.58</b>	<b>1</b>	0.25
$i_8$	Female	20.03	21.25	20	<b>0.22</b>

# Example: Sex classification

Person	height (feet)	weight (lbs)	foot size(inches)
sample	6	130	8

- ▶ 1 –  $NN$ 
  - Class=female
- ▶ 2- $NN$ 
  - Class=female
- ▶ 4- $NN$ 
  - Class=female
- ▶ 7- $NN$ 
  - Minkowsky
    - Class=female
  - Gower
    - Class=Male

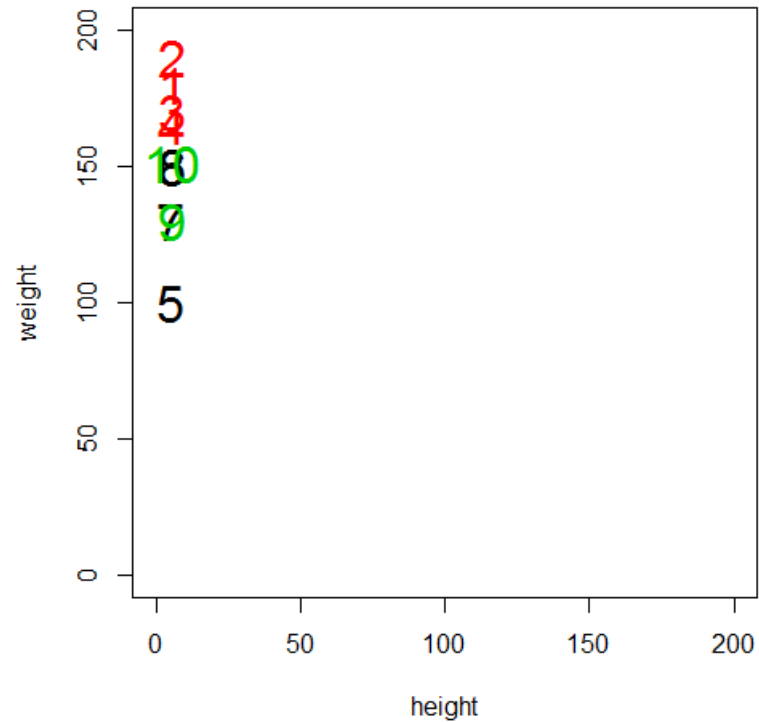
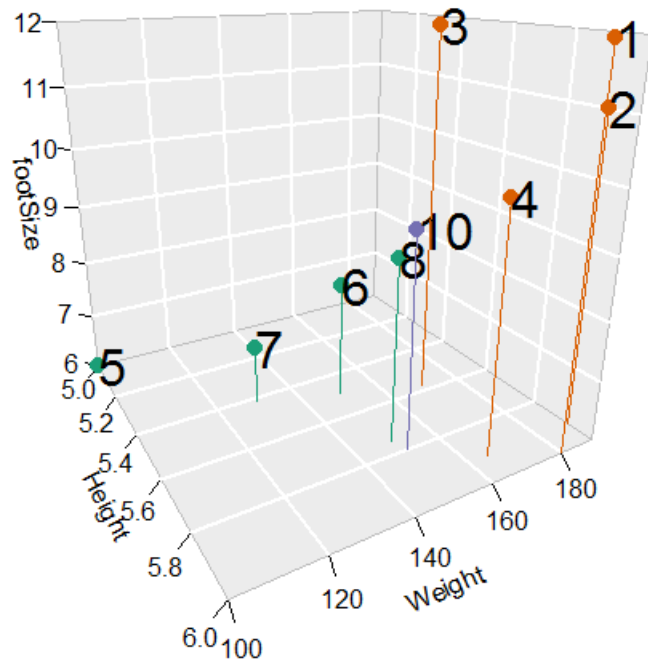
	Class	Euclidean	Manhattan	Minkowski r=100	Gower
$i_1$	Male	50.16	54	50	0.41
$i_2$	Male	60.07	63.03	60	0.42
$i_3$	Male	40.20	44.42	40	0.51
$i_4$	Male	35.06	37.08	35	0.27
$i_5$	Female	30.08	33	30	0.56
$i_6$	Female	20.00	20.50	20	0.24
$i_7$	Female	1.16	1.58	1	0.25
$i_8$	Female	20.03	21.25	20	0.22

# Example: Sex classification -

Range(Weight)=[100..190]

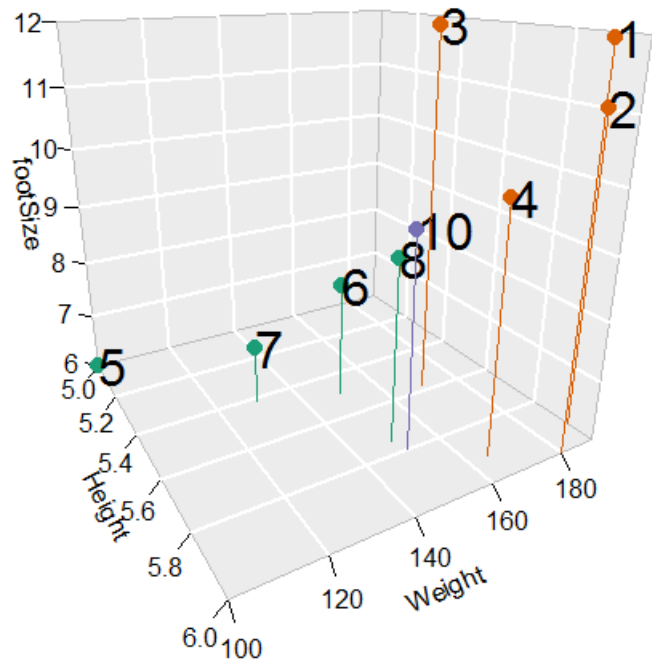
Range(Height)=[5..6]

Range(FoodSize)=[6..12]

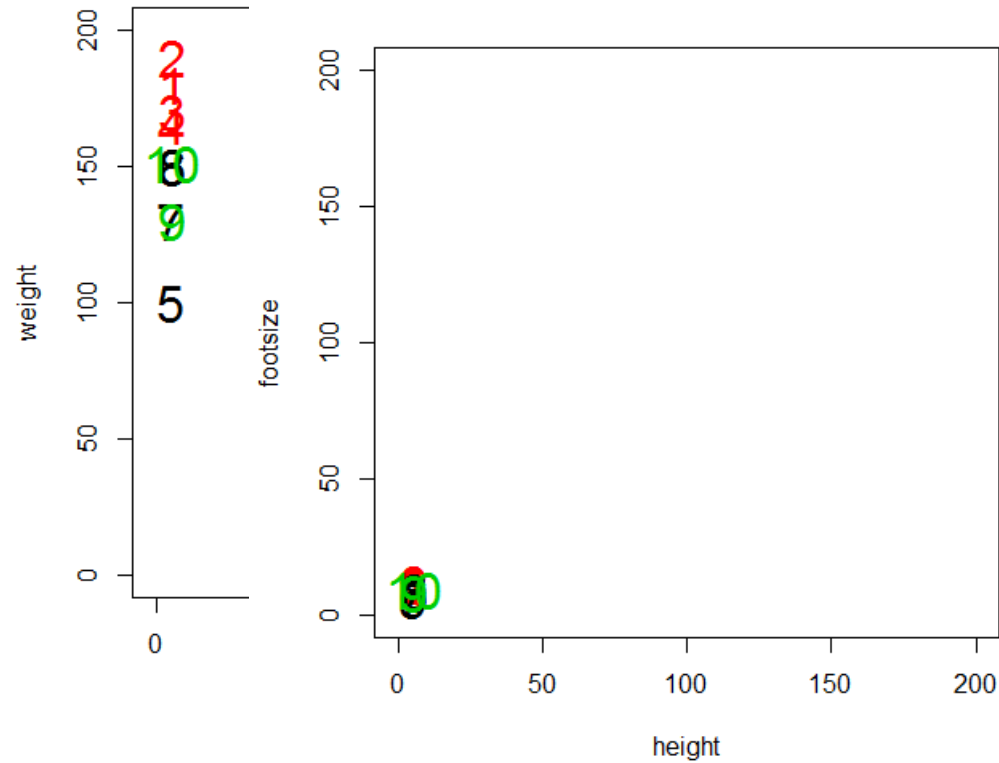




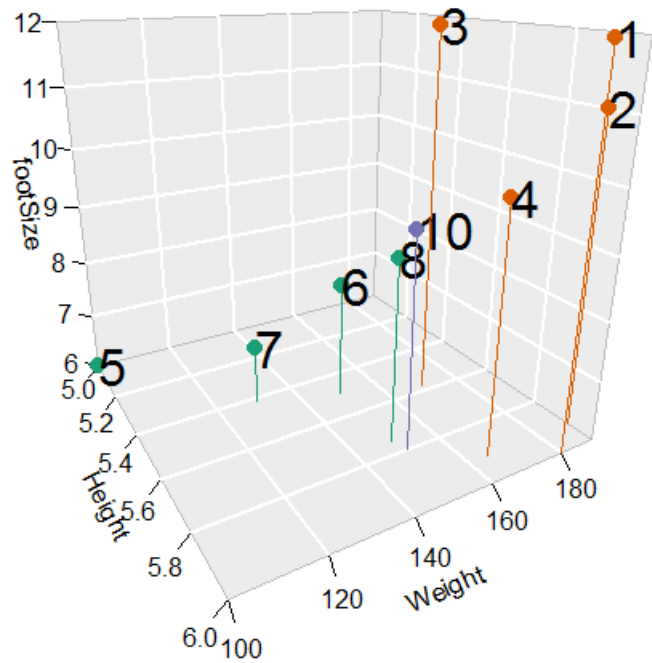
# Example: Sex classification -



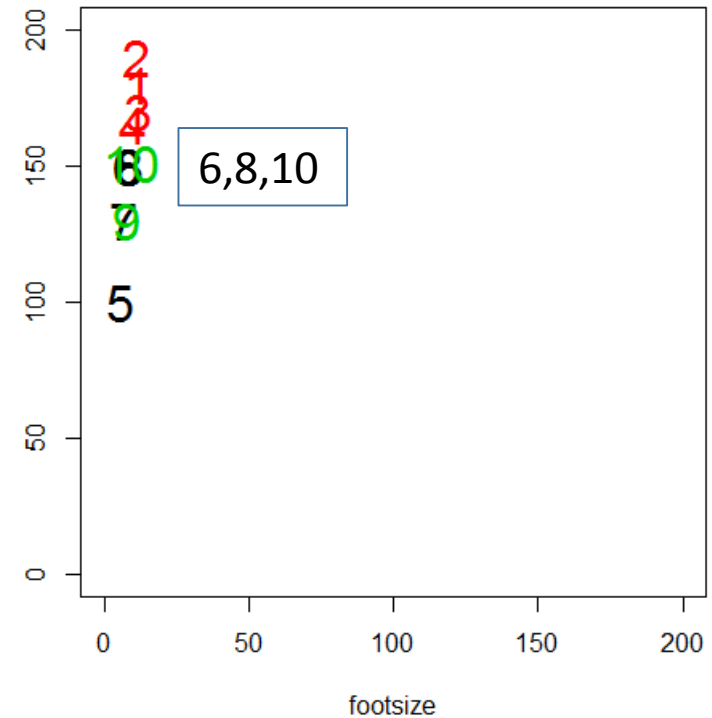
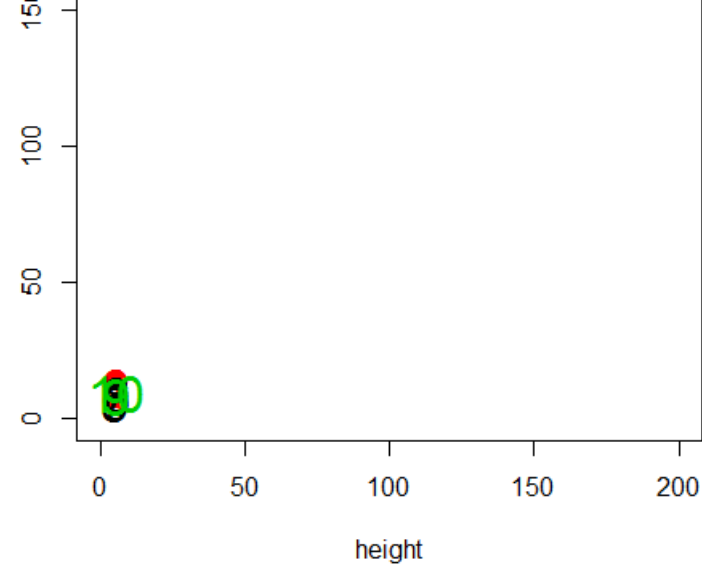
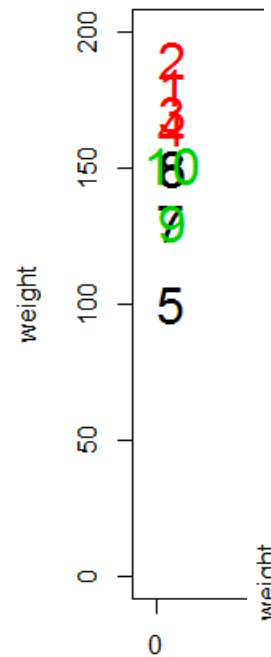
female      male      unknown



# Example: Sex classification



female male unknown



# Example: Sex classification

Person	height (feet)	weight (lbs)	foot size(inches)
sample	6	130	8

## ► 1 – NN

- Class=female
- Scaled centered
  - Class=Male

## ► 2-NN

- Euclidean=Female
- Rest: Tie

	Class	Euclidean	Standardized Euclidean	Normalize Euclidean	Gower	Standardized Gower
i <sub>1</sub>	Male	29.11	1.75	0.31	0.32	0.32
i <sub>2</sub>	Male	39.03	1.68	0.284	0.27	0.27
i <sub>3</sub>	Male	19.17	1.58	0.28	0.28	0.28
i <sub>4</sub>	Male	14.00	<b>0.69</b>	<b>0.10</b>	<b>0.12</b>	<b>0.12</b>
i <sub>5</sub>	Female	51.13	3.59	0.49	0.65	0.65
i <sub>6</sub>	Female	<b>1.83</b>	<b>1.19</b>	0.16	0.19	0.19
i <sub>7</sub>	Female	21.15	1.88	0.29	0.34	0.34
i <sub>8</sub>	Female	<b>1.12</b>	1.25	<b>0.05</b>	<b>0.05</b>	<b>0.05</b>

# Example: Sex classification

Person	height (feet)	weight (lbs)	foot size(inches)
sample	6	130	8

## ► 1 – NN

- Class=female
- Scaled centered
  - Class=Male

## ► 2-NN

- Euclidean=Female
- Rest: Tie

## ► 5-NN

- Euclidean
  - Class= Female
- Rest
  - Class=Male

	Class	Euclidean	Standardized Euclidean	Normalized Euclidean	Gower	Standardized Gower
i <sub>1</sub>	Male	29.11	1.75	0.31	0.32	0.32
i <sub>2</sub>	Male	39.03	1.68	0.284	0.27	0.27
i <sub>3</sub>	Male	19.17	1.58	0.28	0.28	0.28
i <sub>4</sub>	Male	14.00	<b>0.69</b>	<b>0.10</b>	<b>0.12</b>	<b>0.12</b>
i <sub>5</sub>	Female	51.13	3.59	0.49	0.65	0.65
i <sub>6</sub>	Female	<b>1.83</b>	<b>1.19</b>	0.16	0.19	0.19
i <sub>7</sub>	Female	21.15	1.88	0.29	0.34	0.34
i <sub>8</sub>	Female	<b>1.12</b>	1.25	<b>0.05</b>	<b>0.05</b>	<b>0.05</b>

# Example: Sex classification

Person	height (feet)	weight (lbs)	foot size(inches)
sample	6	130	8

- ▶ Weighted k-NN
- ▶ weight:  $1/(d+1)$
- ▶ Labels: male=-1 and female=1
- ▶ Euclidean
  - 2-NN:
    - $\frac{1}{0.05+1} * 1 + \frac{1}{0.1+1} * (-1) = 0,04$
  - 3-NN: 0.91
  - 4-NN: 0.12
- ▶ Gower:
  - 2-NN: 0.06
  - 3-NN: 0.89
  - 4-NN: 0.11
- ▶ All females

	Class	Normalized Euclidean	Gower
i <sub>1</sub>	Male	0.31	0.32
i <sub>2</sub>	Male	<b>0.284</b>	<b>0.27</b>
i <sub>3</sub>	Male	<b>0.28</b>	<b>0.28</b>
i <sub>4</sub>	Male	<b>0.10</b>	<b>0.12</b>
i <sub>5</sub>	Female	0.49	0.65
i <sub>6</sub>	Female	<b>0.16</b>	<b>0.19</b>
i <sub>7</sub>	Female	0.29	0.34
i <sub>8</sub>	Female	<b>0.05</b>	<b>0.05</b>

# ● Example: Sex classification

Person	height (feet)	weight (lbs)	foot size(inches)
sample	6	130	8

- ▶ Weighted k-NN
- ▶ weight:  $1/(d+1)$  or  $(1-d)$
- ▶ Labels: male=-1 and female=1

	$1/(d+1)$		$1-d$	
	Euclidean	Gower	Euclidean	Gower
2-NN	0.04	0.06	0.05	0.07
3-NN	0.91	0.89	0.89	0.88
4-NN	0.12	0.11	0.27	0.15

	Class	Normalized Euclidean	Gower
$i_1$	Male	0.31	0.32
$i_2$	Male	0.284	<b>0.27</b>
$i_3$	Male	<b>0.28</b>	0.28
$i_4$	Male	<b>0.10</b>	<b>0.12</b>
$i_5$	Female	0.49	0.65
$i_6$	Female	<b>0.16</b>	<b>0.19</b>
$i_7$	Female	0.29	0.34
$i_8$	Female	<b>0.05</b>	<b>0.05</b>

- ▶ All females

## ► Naïve Bayes

- Naivebayes from library e1071

## ► kNN

- knn, knn.cv from class library
  - Knn.dist , knn.predict from KODAMA library
  - Train(“knn”) caret library
  - kNN from VIM library
- 
- Dummy variables for categorical and then treat as numerical?

# Generalization error

To have an honest estimate of the error rate it must be computed on an independent data different from the training set. Called *test data* or *hold-out data*. Then, it is called the **Generalization error**

Let be a model:  $\hat{y} = f(X, w)$

- $f$  represents a class of functions to predict  $y$
- $X$  is the matrix of explanatory variables
- $w$  represents the coefficients of the model

Generalization error (=Loss function):  $L(y, f(X, w))$   
= Expected loss with independent data

$$= [y - \hat{y}]^2$$

(if regression)

$$= [y \neq \hat{y}]$$

(if classification)

$$Risk_{theoretical} = E[L] = \int L(y, f(X, w)) dP(X, y)$$



# Empirical Generalization error

Generalization error in the training:

$$Risk_{learn} = \frac{1}{n} \sum L(y, f(X, w))$$

If computed in the learning data we call it  $Risk_{learn}$

– If regression:

$$\frac{\sum (y - f(X, w))^2}{n}$$

– If classification: *percentage of misclassified* (error rate)

$$\frac{\sum (y \neq f(X, w))}{n}$$

What we expect from  $Risk_{learn}$ ?  
if  $n \rightarrow \infty$

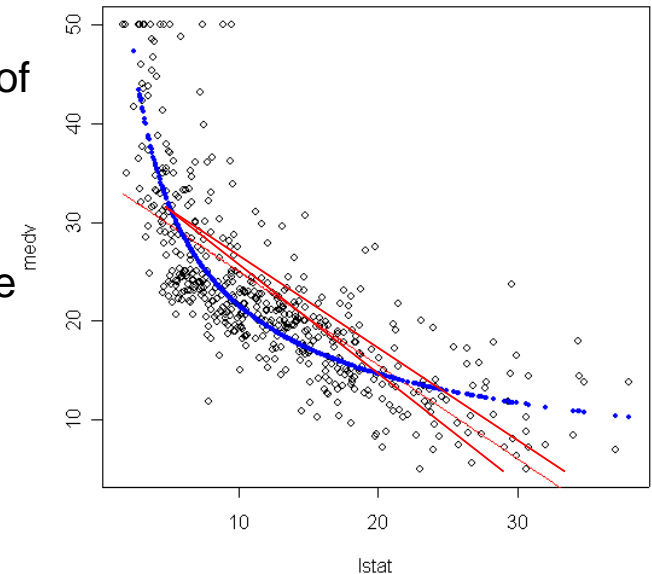
# Components of the generalization error

*Components of the Risk (Generalization Error) in regression:*

$$GE(\hat{y}) = E[y_0 - \hat{y}_0]^2 = E[\hat{y}_0 - r(x_0)]^2 + \sigma_0^2 = \sigma_0^2 + \text{var}(\hat{y}_0) + (E[\hat{y}_0] - r(x_0))^2$$

Risk = Generalization (Prediction) Error

1. **Random fluctuation** of the phenomenon (deviation of the actual data respect to the true model due to multiple small and unknown causes)
2. **Variance** of fit (deviation of the prediction trained model respect to its average, i.e. with another sample we would obtained a different fit)
3. **Bias** (deviation of the true model respect to the average prediction of the trained model)

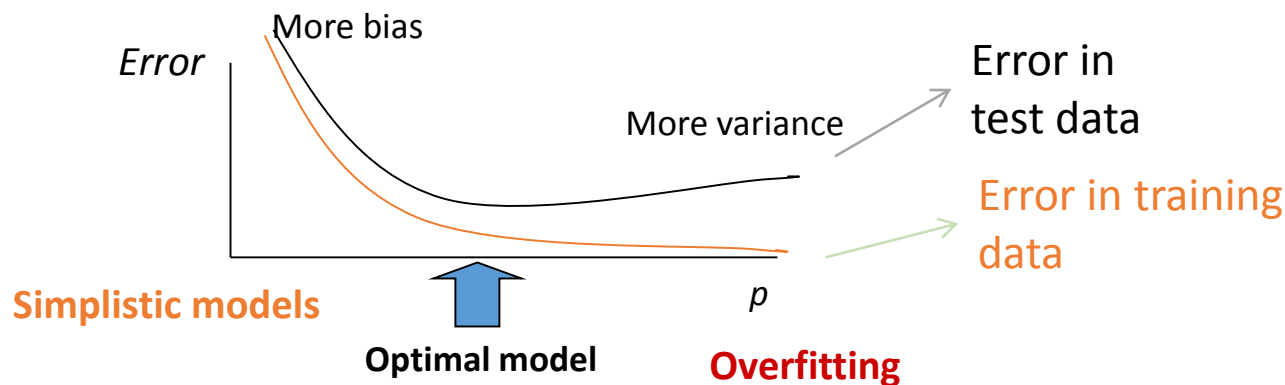


# Trade off of the generalization error

$$GE(\hat{y}) = E[y_0 - \hat{y}_0]^2 = \sigma_0^2 + \text{var}(\hat{y}_0) + (E[\hat{y}_0] - r(x_0))^2$$

The GE depends on the complexity of the model  
With  $p$  predictors we have  $2^p - 1$  possible models

## Trade-off between bias and variance



### Components of bias:

- Bad specification of the classifier function  $r$
- Not including all the relevant predictors in the model

### Components of prediction variance:

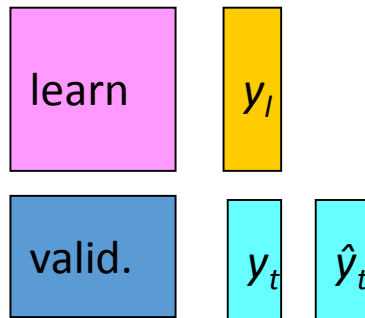
- Complexity of the model

## Finding the optimal model for prediction: Validation sample

To obtain reliable estimations we need to estimate it applying the model in an *independent* sample, from the one used to estimate the model, i.e. we divide the learning sample in two parts (learn and validation).

Instead of the  $GE$ , we can compute the  $R^2$  (if regression).

### Validation sample



$$GE_{valid}(\hat{y}) = \frac{1}{n_{valid}} \sum_{i=1}^{n_{test}} (y_i^{valid} - \hat{y}_i^{valid})^2$$
$$R_{valid}^2 = 1 - \frac{GE_{valid}(\hat{y})}{\frac{1}{n_{valid}} \sum_{i=1}^{n_{valid}} (y_i^{valid} - \bar{y}^{valid})^2}$$

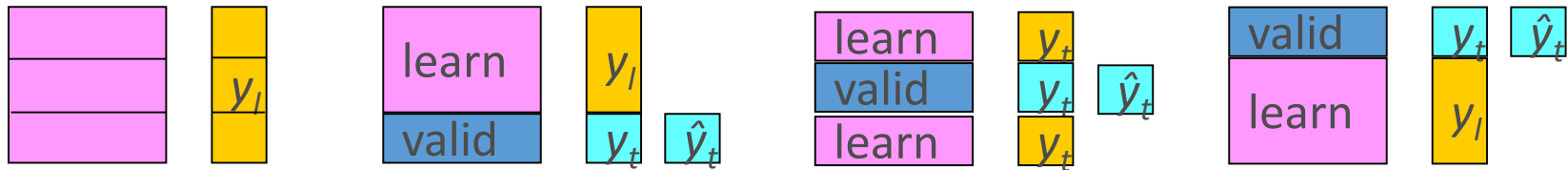
We split the available data in two parts at random, one used for learning (2/3) the other used to validate the model (1/3).

### Evaluation of the generalization error in **R** with a validation sample

```
> ypred = predict(l1, data=dd.valid)
> 1 - (sum((ypred-yvalid)^2) / sum((yvalid-mean(yvalid))^2))
```

# Finding the optimal model for prediction: Cross-validation

We split the learning data in  $k$  parts at random



$$GE_{cv}(\hat{y}) = PRESS = \frac{1}{n} \sum_{k=1}^R \sum_{i \in k} (y_i^{valid} - \hat{y}_{-i}^{valid})^2$$

$$R_{cv}^2 = 1 - \frac{GE_{cv}(\hat{y})}{\frac{1}{n} \sum_{i=1}^n (y_i - \bar{y})^2}$$

$$MGE_{cv}(\hat{y}) = \frac{1}{n} PRESS = \frac{1}{n} \sum_{i=1}^n \left( \frac{e_i}{1 - h_{ii}} \right)^2$$

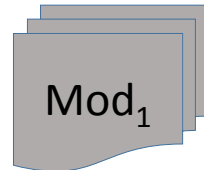
LOO (leave one out crossvalidation)  
each individual is a part,  $k=n$

```
> n      = length(medv)
> ECMPcv = sum((l1$residuals/(1-ls.diag(l1)$hat))^2)/n
> R2cv   = 1 - ECMPcv*n/(var(medv)*(n-1))
> R2cv
```

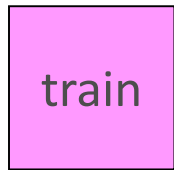
# The process of modeling

## 1. Selecting the optimal model

Competing models:



Define for each model the best parameters with the training data



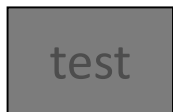
Selection of the optimal model:

- Crossvalidation
- Validation sample
- Likelihood penalization for complexity: AIC, BIC.

## 2. Estimation of the optimal model

Estimation of the selected optimal model in the whole training data

## 3. Computing honest estimates of the goodness of the model



Estimation of quality measures (error rate, ...) in the test data