# Classificazione

- Classificatore di Bayes
  - Approccio parametrico (distribuzione Multinormale)
  - Approccio non parametrico (Parzen Window)
- Nearest Neighbor
  - *k*-NN
  - Metriche
- SVM
  - Lineari: pattern linearmente separabili e non
  - Non lineari
  - Caso multiclasse
- Multi classificatori
  - Fusione a livello di decisione
  - Fusione a livello di confidenza
  - Random Forest (Bagging)
  - AdaBoost (Boosting), Gradient Boosting

# Approccio Bayesiano

Il problema è posto in termini probabilistici. Se tutte le distribuzioni in gioco sono note l'approccio Bayesiano costituisce la migliore regola di classificazione possibile: soluzione OTTIMA !

- Sia $\mathbf{V}$ uno spazio di pattern $d$-dimensionali e $\mathrm{W} = \{w_1, w_2 \ldots w_s\}$ un insieme di $s$ classi disgiunte costituite da elementi di $\mathbf{V}$
- Per ogni $\mathbf{x} \in \mathbf{V}$ e per ogni $w_i \in \mathrm{W}$, indichiamo con $p(\mathbf{x}|w_i)$ la *densità di probabilità condizionale* (o condizionata) di $\mathbf{x}$ data $w_i$, ovvero la densità di probabilità che il prossimo pattern sia $\mathbf{x}$ sotto l'ipotesi che la sua classe di appartenenza sia $w_i$
- Per ogni $w_i \in \mathrm{W}$, indichiamo con $P(w_i)$ la *probabilità a priori* di $w_i$ ovvero la probabilità, indipendentemente dall'osservazione, che il prossimo pattern da classificare sia di classe $w_i$
- Per ogni $\mathbf{x} \in \mathbf{V}$ indichiamo con $p(\mathbf{x})$ la *densità di probabilità assoluta* di $\mathbf{x}$, ovvero la densità di probabilità che il prossimo pattern da classificare sia $\mathbf{x}$

$$p(\mathbf{x}) = \sum_{i=1}^{s} p(\mathbf{x}|w_i) \cdot P(w_i) \qquad \text{dove} \qquad \sum_{i=1}^{s} P(w_i) = 1$$

- Per ogni $w_i \in \mathrm{W}$ e per ogni $\mathbf{x} \in \mathbf{V}$ indichiamo con $P(w_i|\mathbf{x})$ la *probabilità a posteriori* di $w_i$ dato $\mathbf{x}$, ovvero la probabilità che avendo osservato il pattern $\mathbf{x}$, la classe di appartenenza sia $w_i$. Per il teorema di Bayes:

$$P(w_i|\mathbf{x}) = \frac{p(\mathbf{x}|w_i) \cdot P(w_i)}{p(\mathbf{x})}$$

# Classificatore di Bayes

Dato un pattern $\mathbf{x}$ da classificare in una delle $s$ classi $w_1, w_2 \ldots w_s$ di cui sono note:

- le probabilità a priori $P(w_1)$, $P(w_2)$ … $P(w_s)$
- le densità di probabilità condizionali $p(\mathbf{x}|w_1)$, $p(\mathbf{x}|w_2)$ … $p(\mathbf{x}|w_s)$

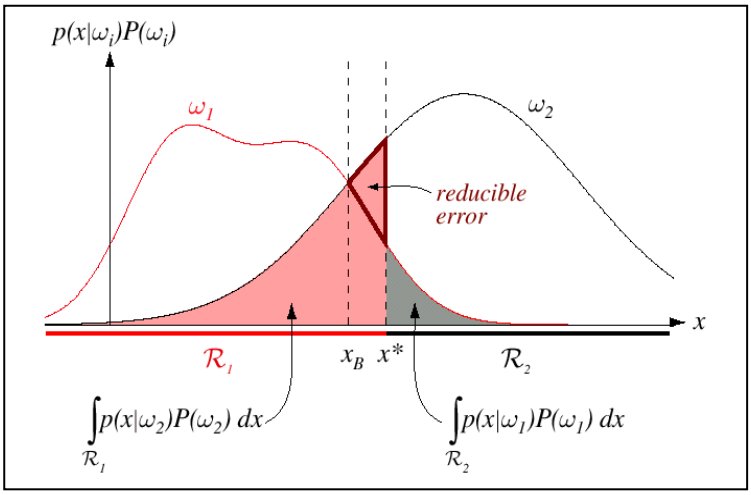
la regola di classificazione di Bayes assegna $\mathbf{x}$ alla classe $b$ per cui è massima la probabilità a posteriori:

$$b = \underset{i=1..s}{argmax} \{ P(w_i|\mathbf{x}) \}$$

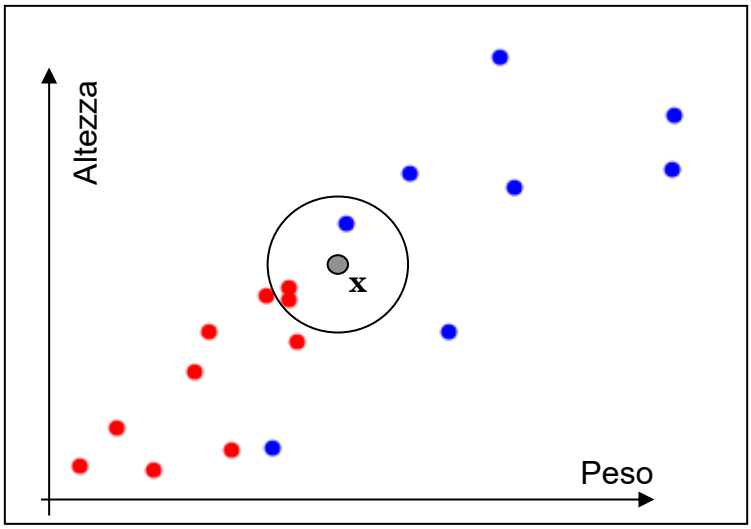
Massimizzare la probabilità a posteriori significa massimizzare la densità di probabilità condizionale tenendo comunque conto della probabilità a priori delle classi.

- La regola si dimostra ottima in quanto minimizza l'errore di classificazione. Ad esempio nel caso di 2 classi e $d$ = 1:

$$P(error) = \int_{\Re_1} p(x|w_2)P(w_2)dx + \int_{\Re_2} p(x|w_1)P(w_1)dx$$



# Esempio




Classificare le persone in maschi/femmine in base a peso e all'altezza, a partire dal training set in figura.

- $\mathbf{V}$ è uno spazio a 2 dimensioni ($d$ =2)
- $W = \{w_1, w_2\}$
  $w_1$ = maschi (blu),
  $w_2$ = femmine (rosso)

Una *stima (grossolana) delle probabilità a priori e delle densità* può essere effettuata a partire dal training set come segue (si vedranno in seguito tecniche più rigorose per effettuare tale stima):

- *Probabilità a priori*: si considera semplicemente l'occorrenza dei pattern nel training set: $P(w_1) = 8/18$ , $P(w_2) = 10/18$
- *Densità di probabilità condizionali* per un nuovo pattern $\mathbf{x}$ da classificare: si contano le occorrenze dei pattern del training set delle due classi in un intorno di $\mathbf{x}$ :
  - $p(\mathbf{x}|w_1) = 1/8$
  - $p(\mathbf{x}|w_2) = 2/10 = 1/5$
  - $p(\mathbf{x}) = \frac{1}{8} \times \frac{8}{18} + \frac{1}{5} \times \frac{10}{18} = \frac{1}{18} + \frac{2}{18} = \frac{1}{6}$

Si ottiene quindi:

$$P(w_1|\mathbf{x}) = \frac{p(\mathbf{x}|w_1) \cdot P(w_1)}{p(\mathbf{x})} = \frac{\mathbf{1/18}}{\mathbf{1/6}} = \mathbf{\frac{1}{3}}$$

$$P(w_2|\mathbf{x}) = \frac{p(\mathbf{x}|w_2) \cdot P(w_2)}{p(\mathbf{x})} = \frac{\mathbf{2/18}}{\mathbf{1/6}} = \mathbf{\frac{2}{3}}$$

L'approccio Bayesiano assegna il pattern $\mathbf{x}$ alla classe $w_2$ (femmine).

# Bayes: approccio parametrico e non-parametrico

- Mentre la stima delle probabilità a priori è abbastanza semplice (se non si hanno elementi si possono ipotizzare le classi equiprobabili), la conoscenza delle densità condizionali è possibile "solo in teoria"; nella pratica due soluzioni:
  - Approccio parametrico: si fanno ipotesi sulla forma delle distribuzioni (es. distribuzione multinormale) e si apprendono i parametri fondamentali (vettore medio, matrice di covarianza) dal training set.
  - Approccio non parametrico: si apprendono le distribuzioni dal training set (es. attraverso il metodo Parzen Window).

Generalmente l'approccio parametrico si utilizza quando, oltre ad avere una ragionevole certezza (o *speranza*) che la forma della distruzione sia adeguata, la dimensione del training set non è sufficiente per una buona stima della densità.

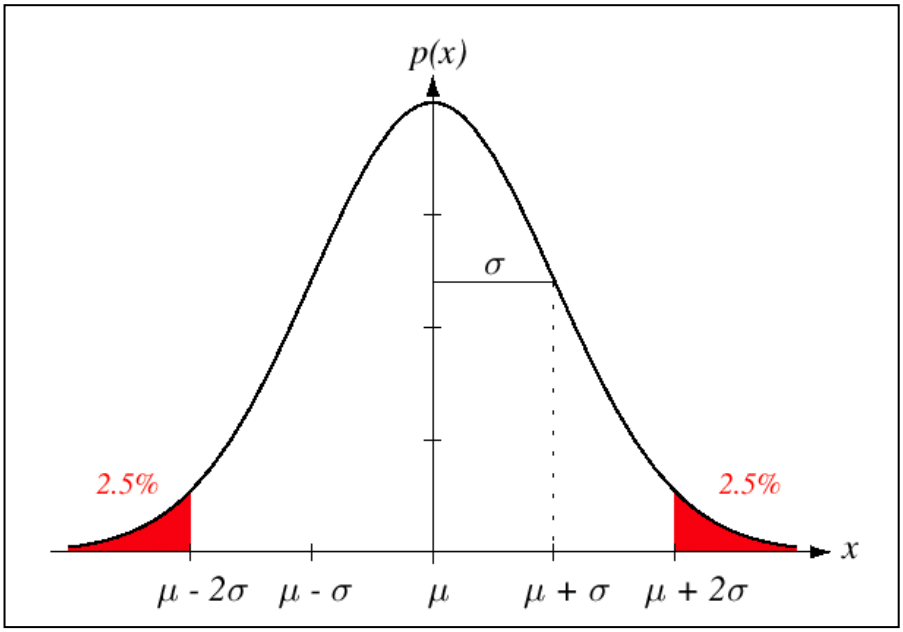
- L'approccio parametrico è infatti generalmente caratterizzato da un minor numero di gradi di libertà e il rischio di overfitting dei dati, quando il training set è piccolo, è minore.

# Distribuzione Normale (d=1)

- La densità di probabilità della distribuzione normale ($d = 1$) è:

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

dove $\mu$ è il valor medio è $\sigma$ la deviazione standard (o scarto quadratico medio) e il suo quadrato $\sigma^2$ la varianza.




Solo il 5% circa del "volume" è esterno all'intervallo $[\mu - 2\sigma, \mu + 2\sigma]$.

Solitamente si assume che la distribuzione valga 0 a distanze maggiori di $3\sigma$ dal valore medio.

# Esempio stima di $\mu$ e $\sigma$ (d=1)

- Dato un training set di pattern mono-dimensionali composto da $n = 10$ elementi:

$$\{3, 7, 9, -2, 15, 54, -11, 0, 23, -8\}$$


- La stima dei parametri per massima verosimiglianza (maximum likelihood) si dimostra [1] essere:
  - Stima per $\mu$: media campionaria dei valori.

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i = \frac{3+7+9+(-2)+15+54+(-11)+0+23+(-8)}{10} = \frac{90}{10} = 9$$

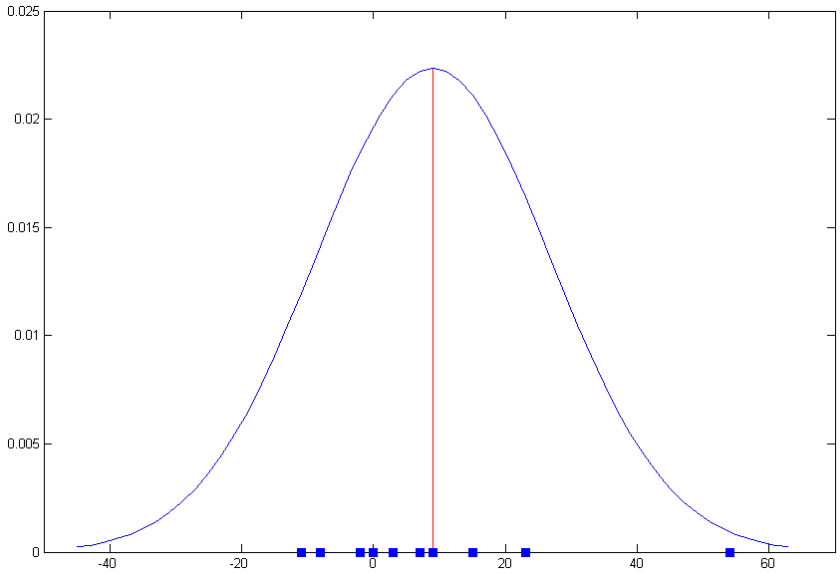
  - Stima per $\sigma^2$: varianza campionaria dei valori.

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2 =$$

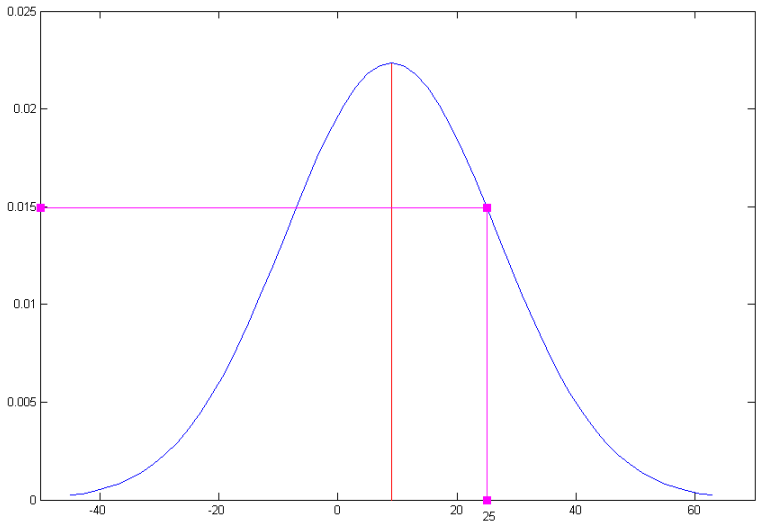
$$= \frac{(3-9)^2 + (7-9)^2 + (9-9)^2 + (-2-9)^2 + (15-9)^2 + (54-9)^2 + (-11-9)^2 + (0-9)^2 + (23-9)^2 + (-8-9)^2}{10} = 318.8$$

[1] https://it.wikipedia.org/wiki/Metodo_della_massima_verosimiglianza

# …in forma grafica




$$p(25) = \frac{1}{17.855 \cdot \sqrt{2 \cdot 3.1416}} \cdot e^{-\frac{(25-9)^2}{2 \cdot 318.8}} = \frac{1}{44.7559} \cdot e^{-0.4015} = 0.01495$$




ATTENZIONE SIAMO NEL CONTINUO:
$p$ è una densità di probabilità: $p(25)$ non è la probabilità del valore 25 (questa vale 0!) ma la densità di probabilità nel punto 25. Solo considerando un intervallo di valori (anche piccolo) sulla base possiamo parlare di probabilità.
In altre parole l'intervallo $[x, x + dx]$ ha probabilità $p(x)dx$.

# Distribuzione Normale Multivariata (Multinormale)

- *Notazione*: per evitare confusione utilizziamo a pedice l'indice del pattern e (ove necessario) ad apice la componente (scalare):
  - $\mathbf{x}_i$ pattern i-esimo (vettore)
  - $x_i^j$ componente j-esima del pattern i-esimo (scalare)

- La densità di probabilità nella distribuzione multinormale ($d > 1$):

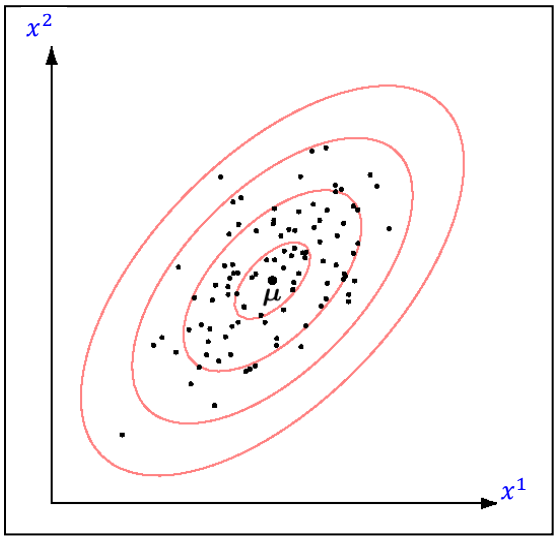
$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \; e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^t \; \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})}$$

dove $\boldsymbol{\mu} = [\mu^1, \mu^2 \ldots \mu^d]$ è il vettore medio è $\Sigma = [\sigma^{ij}]$ la matrice di covarianza $(d \times d)$.

- Si assume che i vettori siano di tipo «colonna». L'apice $t$ (trasposto) li trasforma in righe.
- $|\Sigma|$ e $\Sigma^{-1}$ sono rispettivamente il determinante e l'inversa di $\Sigma$.
- La matrice di covarianza è sempre simmetrica e definita positiva, pertanto ammette inversa. Essendo simmetrica il numero di parametri che la definisce è $d \cdot (d+1)/2$
- Gli elementi diagonali $\sigma^{ii}$ sono le varianze dei rispettivi $x^i$ (ovvero $(\sigma^i)^2$); gli elementi non diagonali $\sigma^{ij}$ sono le covarianze tra $x^i$ e $x^j$:
  - se $x^i$ e $x^j$ sono statisticamente indipendenti $\sigma^{ij} = 0$
  - se $x^i$ e $x^j$ sono correlati positivamente $\sigma^{ij} > 0$
  - se $x^i$ e $x^j$ sono correlati negativamente $\sigma^{ij} < 0$

# Rappresentazione grafica Normale Multivariata

- Per $d = 2$ la forma della distribuzione è quella di un'ellisse.




Le diverse ellissi individuano luoghi di punti a densità costante

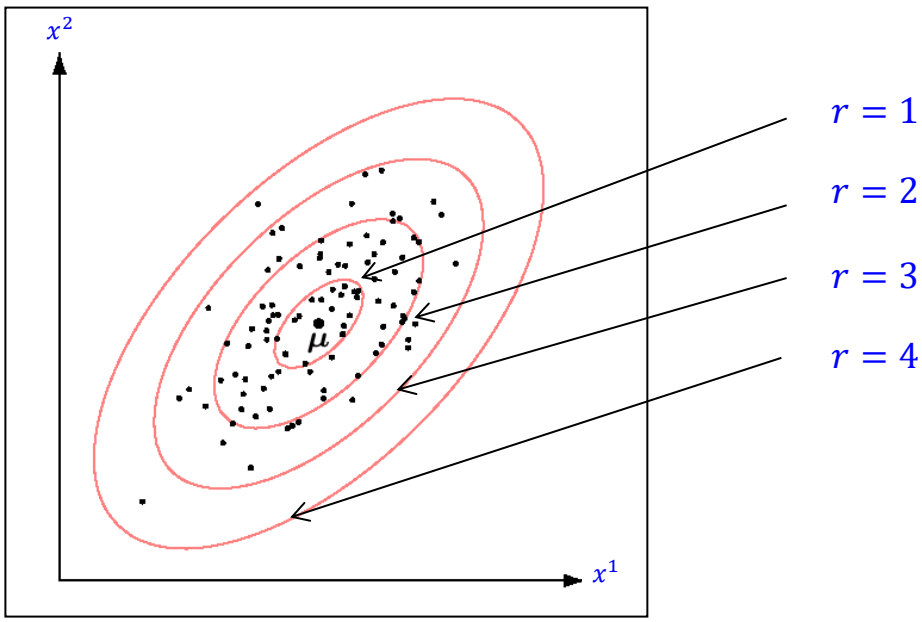
- $\boldsymbol{\mu} = [\mu^1, \mu^2]$ controlla la posizione del centro.
- $\sigma^{11}$ e $\sigma^{22}$ determinano l'allungamento sui due assi dell'ellisse.
- $\sigma^{12} = \sigma^{21}$ controlla la rotazione dell'ellisse rispetto agli assi cartesiani.
  - se = 0 (matrice di covarianza diagonale), la distribuzione multinormale è definita come prodotto di $d$ normali monodimensionali. In tal caso gli assi dell'ellisse sono paralleli agli assi cartesiani (es. Naive Bayes Classifier).
  - Se > 0 (*come nel caso della figura*) $x^1$ e $x^2$ sono positivamente correlate (quando aumenta $x^1$ aumenta anche $x^2$).
  - Se < 0 $x^1$ e $x^2$ sono negativamente correlate (quando aumenta $x^1$ cala $x^2$).
- Gli assi dell'ellisse sono paralleli agli autovettori di $\Sigma$.

# Distanza Mahalanobis

- La distanza di Mahalanobis $r$ tra $\mathbf{x}$ e $\boldsymbol{\mu}$, definita dall'equazione:

$$r^2 = (\mathbf{x} - \boldsymbol{\mu})^t \, \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu})$$

definisce i bordi a densità costante in una distribuzione multinormale. Tale distanza viene spesso utilizzata in sostituzione della distanza euclidea, essendo in grado di "pesare" le diverse componenti tenendo conto dei relativi spazi di variazione e della loro correlazione.

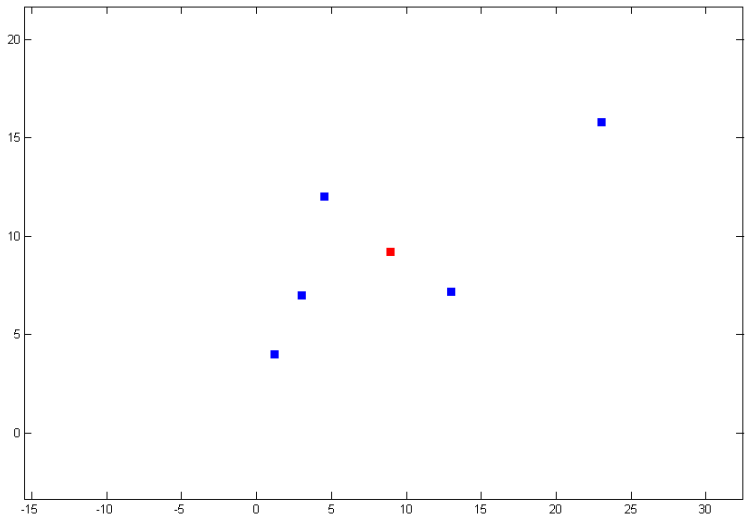



- Un semplice approccio di «anomaly detection» consiste nel fitting gaussiano (calcolo di $\boldsymbol{\mu}$ e $\Sigma$ ) dai soli pattern di funzionamento normale del sistema. I nuovi pattern, la cui distanza di Mahalanobis eccede una soglia data, sono considerati anomalie. È efficace però solo quando la distribuzione dei dati è gaussiana.

# Esempio stima di $\mu$ e $\sigma$ (d=2)

- Dato un training set di pattern bi-dimensionali composto da $n = 5$ elementi:

$$\left\{ [3,7]^t, [4.5,12]^t, [13,7.2]^t, [1,4]^t, [23,15.8]^t \right\}$$




- La stima dei parametri per massima verosimiglianza (maximum likelihood) è:

$$\boldsymbol{\mu} = \begin{bmatrix} \mu^1 \\ \mu^2 \\ \dots \\ \mu^d \end{bmatrix} \quad , \quad \mu^i = \frac{1}{n} \sum_{k=1\dots n} x_k^i \qquad \boldsymbol{\mu} = \begin{bmatrix} \frac{3+4.5+13+1+23}{5} \\ \frac{7+12+7.2+4+15.8}{5} \end{bmatrix} = \begin{bmatrix} 8.9 \\ 9.2 \end{bmatrix}$$

o, in notazione vettoriale:

$$\boldsymbol{\mu} = \frac{1}{n} \sum_{i=1\dots n} \mathbf{x}_i$$

# …prosegue

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma^{11} & \sigma^{12} & \dots & \sigma^{1d} \\ \sigma^{21} & \sigma^{22} & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \sigma^{d1} & \dots & \dots & \sigma^{dd} \end{bmatrix}, \qquad \sigma^{ij} = \frac{1}{n} \sum_{k=1\dots n} \left(x_k^i - \mu^i\right) \cdot \left(x_k^j - \mu^j\right)$$

o, in notazione vettoriale:

$$\boldsymbol{\Sigma} = \frac{1}{n} \sum_{i=1\dots n} (\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^t$$

*calcolata come somma di matrici, ciascuna ottenuta come vettore colonna per vettore riga.*

$$\boldsymbol{\Sigma} = \frac{1}{n} \boldsymbol{X}\boldsymbol{X}^{\boldsymbol{t}}$$

*…è possibile scriverla in modo ancora più compatto (moltiplicazione di matrici) dove **X** è la matrice contenente i pattern nelle colonne (a cui è stata già tolta la media).*

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sigma^{11} & \sigma^{12} \\ \sigma^{21} & \sigma^{22} \end{bmatrix} = \begin{bmatrix} 66.44 & 25.32 \\ 25.32 & 17.456 \end{bmatrix}$$

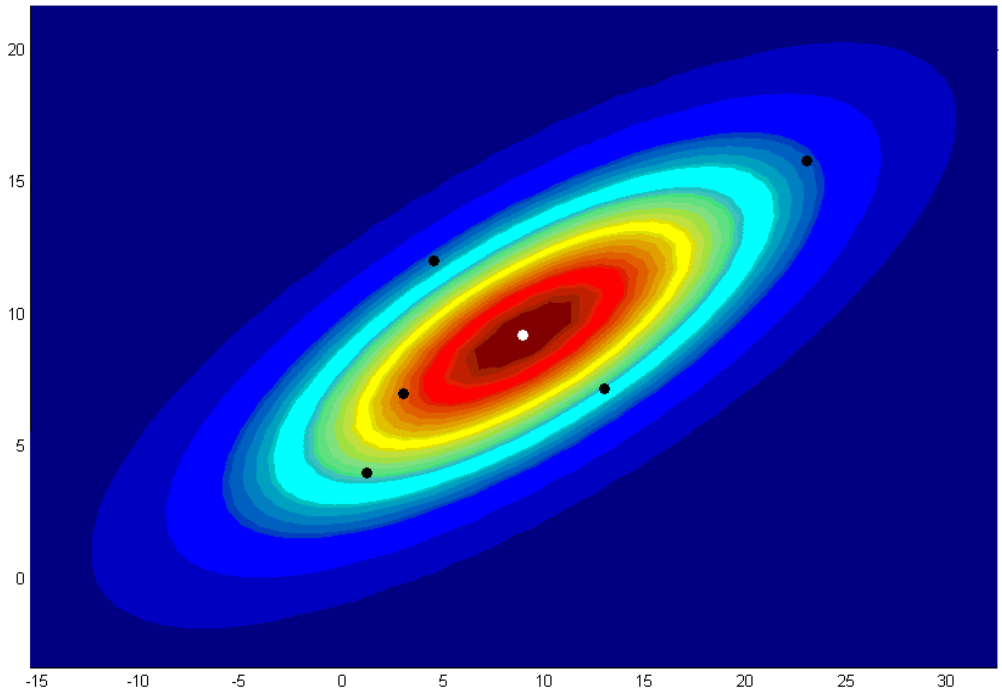
$$\sigma^{11} = \left(\sigma^1\right)^2 = \frac{(3-8.9)^2 + (4.5-8.9)^2 + (13-8.9)^2 + (1-8.9)^2 + (23-8.9)^2}{5} = 66.44$$

$$\sigma^{12} = \sigma^{21} = \frac{(3-8.9)\cdot(7-9.2) + (4.5-8.9)\cdot(12-9.2) + (13-8.9)\cdot(7.2-9.2) + (1-8.9)\cdot(4-9.2) + (23-8.9)\cdot(15.8-9.2)}{5} = 25.32$$

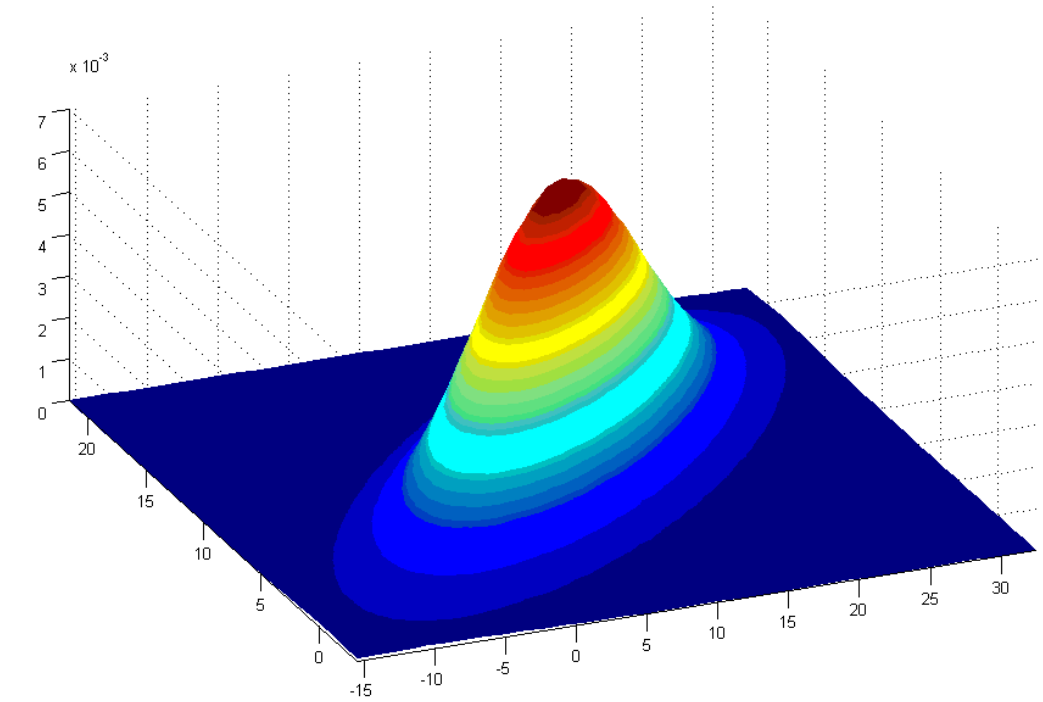
$$\sigma^{22} = \left(\sigma^2\right)^2 = \frac{(7-9.2)^2 + (12-9.2)^2 + (7.2-9.2)^2 + (4-9.2)^2 + (15.8-9.2)^2}{5} = 17.456$$

$$|\boldsymbol{\Sigma}| = (66.44 \cdot 17.456) - (25.32 \cdot 25.32) = 518.674 \qquad \boldsymbol{\Sigma}^{-1} = \begin{bmatrix} 0.0337 & -0.0488 \\ -0.0488 & 0.1281 \end{bmatrix}$$

# in forma grafica

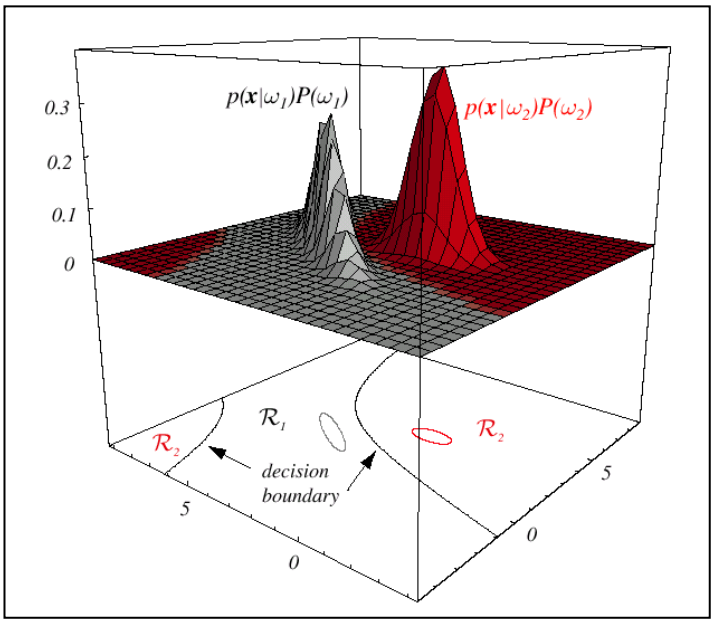



vista dall'alto



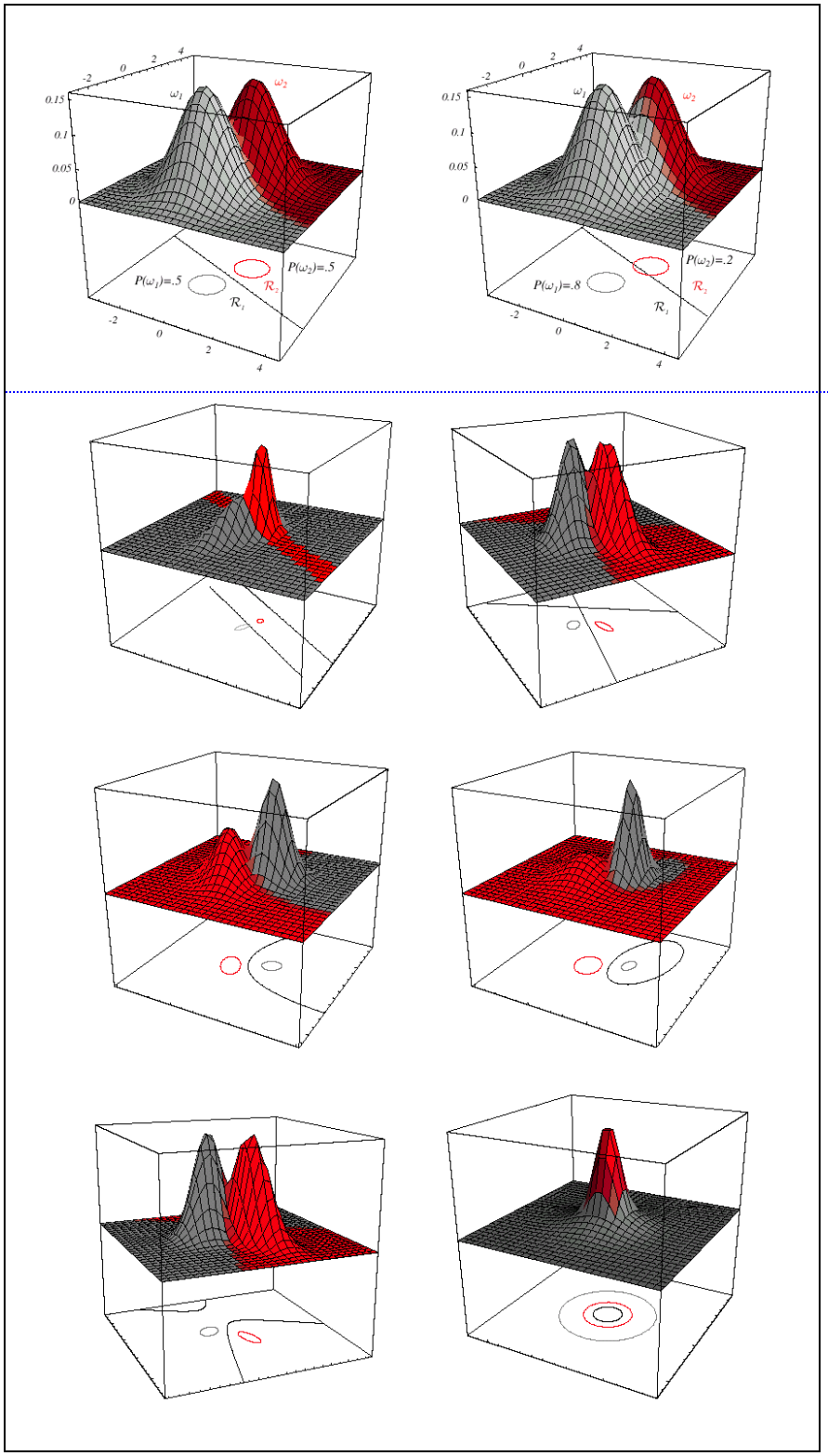

vista laterale

# Classificatore di Bayes con distribuzioni Multinormali




- Nell'esempio sono visualizzate le densità condizionali di 2 classi di pattern (distribuiti con distribuzione normale 2-dimensionale) corrette sulla base delle rispettive probabilità a priori.
- La classificazione è eseguita utilizzando la regola Bayesiana. Lo spazio è suddiviso in regioni non connesse. Nel caso specifico $\Re_2$ è costituita da due componenti disgiunte.
- Un decision boundary o decision surface (superficie decisionale) è una zona di confine tra regioni che il classificatore associa a classi diverse. Sul boundary la classificazione è ambigua.
- Le superfici decisionali possono assumere forme diverse. Nel caso specifico si tratta di due iperboli. In generale:
  - Se le 2 matrici di covarianza sono uguali tra loro: la superficie decisionale è un iper-piano.
  - Se le 2 matrici di covarianza sono arbitrarie: la superficie decisionale è un iper-quadratica.

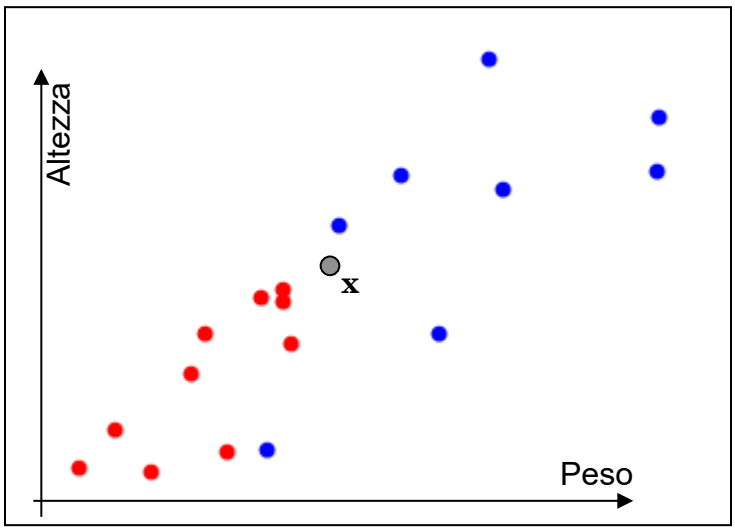
# …altri esempi di superfici decisionali




Stessa matrice di covarianza: iper-piani

Differenti matrici di covarianza: iper-quadratiche

# Maschi/Femmine
# con Bayes parametrico (multinormali)



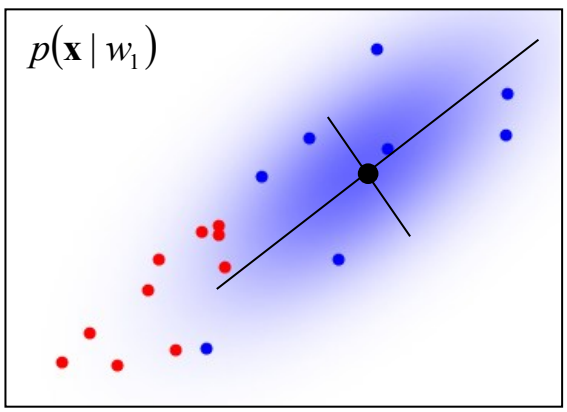

| Peso | Altezza | Classe |
|---|---|---|
| 72 | 173 | $w_1$ |
| 54 | 159 | $w_1$ |
| 65 | 172 | $w_1$ |
| 58 | 170 | $w_1$ |
| 62 | 165 | $w_1$ |
| 72 | 176 | $w_1$ |
| 60 | 173 | $w_1$ |
| 64 | 179 | $w_1$ |
| 55 | 166 | $w_2$ |
| 46 | 158 | $w_2$ |
| 52 | 158 | $w_2$ |
| 47 | 160 | $w_2$ |
| 55 | 167 | $w_2$ |
| 54 | 166 | $w_2$ |
| 55 | 164 | $w_2$ |
| 49 | 157 | $w_2$ |
| 51 | 165 | $w_2$ |
| 51 | 162 | $w_2$ |

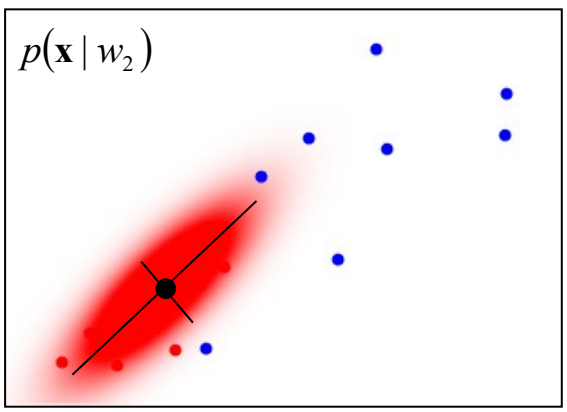
Stima dei parametri dal training set:

$$\boldsymbol{\mu}_1 = [63.4, 170.9]^T \quad \boldsymbol{\mu}_2 = [51.5, 162.3]^T \quad \Sigma_1 = \begin{bmatrix} 35.2 & 23.3 \\ 23.3 & 34.9 \end{bmatrix} \quad \Sigma_2 = \begin{bmatrix} 10.1 & 8.9 \\ 8.9 & 13.0 \end{bmatrix}$$

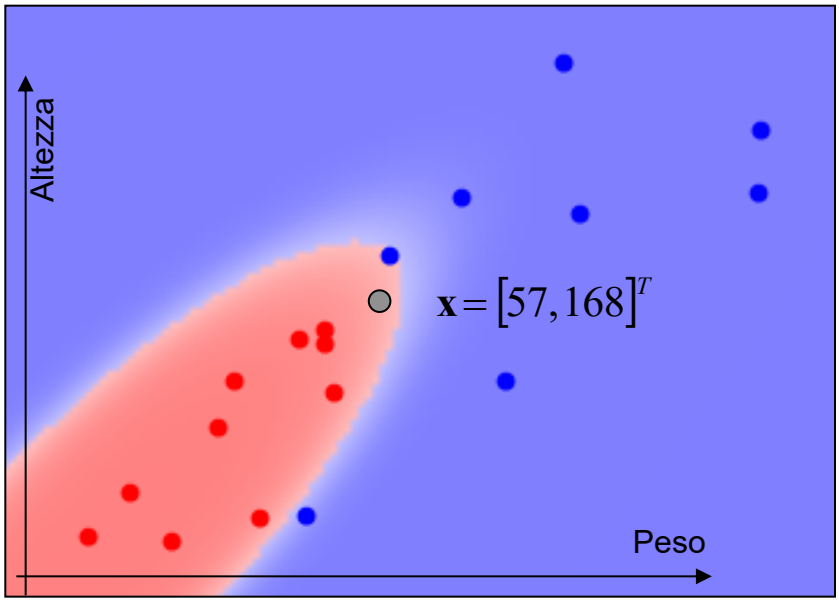
$$p(\mathbf{x} \mid w_1) = \frac{1}{(2\pi)^{d/2} \cdot |\Sigma_1|^{1/2}} \cdot \exp\left[ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_1)^t \, \Sigma_1^{-1} (\mathbf{x} - \boldsymbol{\mu}_1) \right]$$

$$p(\mathbf{x} \mid w_2) = \frac{1}{(2\pi)^{d/2} \cdot |\Sigma_2|^{1/2}} \cdot \exp\left[ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_2)^t \, \Sigma_2^{-1} (\mathbf{x} - \boldsymbol{\mu}_2) \right]$$



# …continua




Supponendo di non avere altre informazioni, si possono stimare le probabilità a priori come: $P(w_1) = 8/18$ , $P(w_2) = 10/18$

$$p(\mathbf{x} \mid w_1) = \frac{1}{(2\pi)^{d/2} \cdot |\Sigma_1|^{1/2}} \cdot \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_1)^t \Sigma_1^{-1}(\mathbf{x}-\boldsymbol{\mu}_1)\right] = 0.0033$$

$$p(\mathbf{x} \mid w_2) = \frac{1}{(2\pi)^{d/2} \cdot |\Sigma_2|^{1/2}} \cdot \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_2)^t \Sigma_2^{-1}(\mathbf{x}-\boldsymbol{\mu}_2)\right] = 0.0045$$

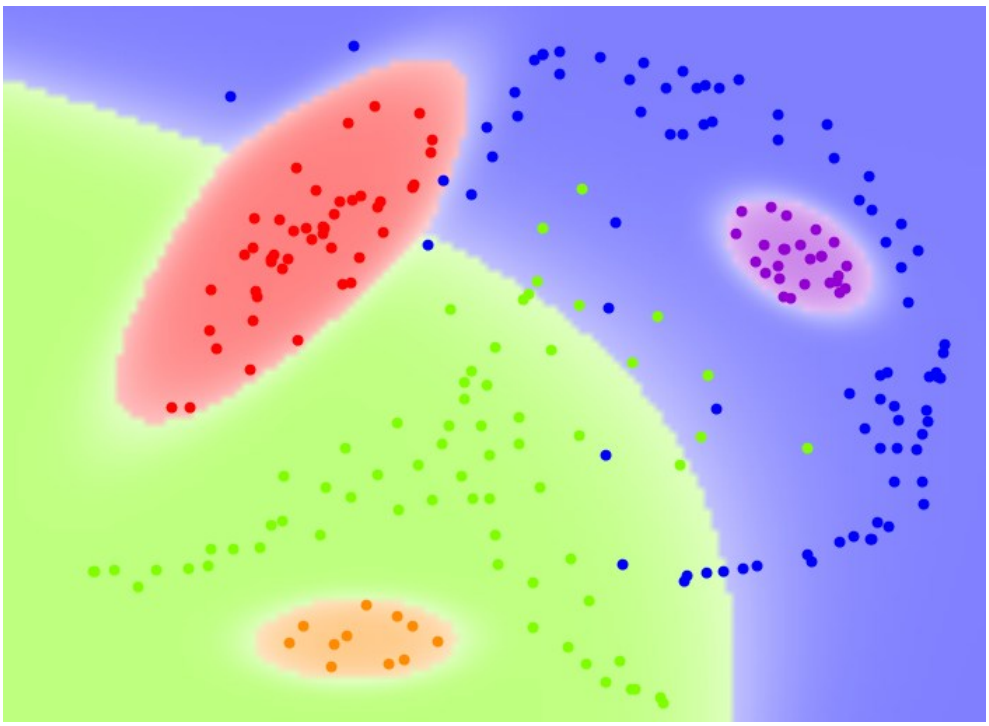
$$p(\mathbf{x}) = \sum_{i=1}^{s} p(\mathbf{x} \mid \mathrm{w}_i) \cdot P(\mathrm{w}_i) = 0.0040$$

$$P(\mathrm{w}_1 \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \mathrm{w}_1) \cdot P(\mathrm{w}_1)}{p(\mathbf{x})} \cong 0.36$$

$$P(\mathrm{w}_2 \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \mathrm{w}_2) \cdot P(\mathrm{w}_2)}{p(\mathbf{x})} \cong 0.64$$

# Bayes e confidenza di classificazione

Un grande vantaggio del classificatore di Bayes, rispetto ad altri classificatori, è legato al fatto che esso produce un valore di output probabilistico (un vero e proprio valore di probabilità tra 0 e 1, con somma 1 sulle diverse classi) che può essere utilizzato come confidenza (*visualizzata nella figura come sfumatura colore*):



Infatti, un classificatore può assegnare un pattern $\mathbf{x}$ a una classe $w_i$ con diversi livelli di certezza (o confidenza) che possono essere impiegati per:

- scartare pattern in applicazioni open-set con soglia
- costruire un multi-classificatore

Se non si è interessati alla confidenza, nella formula di Bayes non è necessario dividere per $p(\mathbf{x})$ il numeratore, e la regola di Bayes è semplicemente:

$$b = \underset{i=1..s}{argmax} \{ p(\mathbf{x}|w_i) \cdot P(w_i) \}$$

# Bayes parametrico in pratica

- Molto spesso si fanno ipotesi azzardate sulla normalità delle densità di probabilità delle classi del problema senza aver sperimentalmente eseguito nessuna verifica; ciò porta ad ottenere cattivi risultati di classificazione.

  Pertanto, dato un problema con $s$ classi e dato un training set (significativo), deve essere innanzitutto valutata la rispondenza alla "normalità" delle $s$ distribuzioni; questo può essere fatto:

  - in modo formale (es: test statistico di Malkovich - Afifi [1] basato sull'indice di Kolmogorov - Smirnov)
  - in modo empirico, visualizzando in vari modi le nuvole dei dati (esistono dei tool già predisposti per questo tipo di analisi fino a 3D) o gli istogrammi sulle diverse componenti e confrontandoli con le curve teoriche.
- Una volta provata una (seppur vaga) normalità delle distribuzioni, si stimano a partire dai dati, vettore medio $\mu$ e matrice di covarianza $\Sigma$ (maximum likelihood).
- Per quanto riguarda le probabilità a priori queste possono essere estratte dalle percentuale di campioni che nel training set appartengono alle diverse classi, o in caso di assenza di informazioni possono essere poste tutte uguali tra loro.
- Ogni nuovo pattern da classificare, è assegnato a una delle possibili classi in accordo con la regola di Bayes nella quale media e covarianza sono ora note.

[1] K. Fukunaga, *Statistical Pattern Recognition*, Academic Press, 1990.

# Approcci non parametrici e stima della Densità

Non vengono fatte ipotesi sulle distribuzioni dei pattern e le densità di probabilità sono stimate direttamente dal training set.

Il problema della stima accurata della densità è ritenuto da molti un problema più complesso della classificazione. *Pertanto perché risolvere come sotto-problema un problema che è più complesso dell'intero compito di classificazione* ?

In generale la stima della densità è affrontabile in spazi a dimensionalità ridotta (es. $d = 3$) e diventa critica al crescere della dimensionalità (curse of dimensionality): il volume dello spazio aumenta così tanto che i pattern diventato troppo sparsi.

Esempio:

- *In un cubo 3D di lato 1, loro distanza media di due punti scelti a caso è 0.66*
- *In un ipercubo con 1M di dimensioni, la distanza media di due punti scelti a caso è 408.25 !*

In uno spazio dove i pattern sono così sparsi occorrono molti più esempi per fare stime di densità affidabili.

La riduzione di dimensionalità (ne parleremo in seguito) è una delle tecniche possibili per contrastare la curse of dimensionality

# Stima della Densità

La probabilità che un pattern $\mathbf{x}$ cada all'interno di $\Re$ è:

$$P_1 = \int_{\Re} p(\mathbf{x}')\, d\mathbf{x}'$$

Dati $n$ pattern indipendenti, la probabilità che $k$ di questi cadano nella regione $\Re$ è calcolabile attraverso la distribuzione binomiale:

$$P_k = \binom{n}{k} {P_1}^k (1 - P_1)^{n-k}$$

il cui valor medio è $k = n\, P_1$ (e quindi $P_1 = k/n$)

Assumendo che la regione $\Re$ (di volume $V$) sia piccola e che quindi $p(\cdot)$ non vari significativamente all'interno di essa:

$$P_1 = \int_{\Re} p(\mathbf{x}')\, d\mathbf{x}' \approx p(\mathbf{x}) \cdot V$$

$$p(\mathbf{x}) = \frac{P_1}{V} = \frac{k}{n \cdot V}$$

# Parzen Window

La regione $\Re$, denominata finestra (Window), è costituita da un ipercubo $d$-dimensionale, definito dalla funzione $\varphi$ :

$$\varphi(\mathbf{u}) = \begin{cases} 1 & |u_j| \leq \frac{1}{2}, j = 1 \dots d \\ 0 & altrimenti \end{cases}$$

Dato un generico ipercubo centrato in $\mathbf{x}$ e avente lato $h_n$ (e quindi volume $V_n = {h_n}^d$) il numero di pattern del training set che cadono all'interno dell'iper-cubo è dato da:

$$k_n = \sum_{i=1}^{n} \varphi\left(\frac{\mathbf{x}_i - \mathbf{x}}{h_n}\right)$$

sostituendo $k_n$ (vedi lucido precedente) si ottiene:

$$p_n(\mathbf{x}) = \frac{1}{n \cdot V_n} \sum_{i=1}^{n} \varphi\left(\frac{\mathbf{x}_i - \mathbf{x}}{h_n}\right), \qquad \text{dove } V_n = {h_n}^d$$

Ovviamente, specie nel caso in cui il numero di pattern non sia elevato, la dimensione della finestra $V_n$ (e quindi il lato $h_n$) ha un forte impatto sul risultato, infatti:

- Se la finestra è piccola, la stima risulta piuttosto "rumorosa", molto attratta dai campioni e statisticamente instabile.
- Se la finestra è grande la stima è più stabile ma piuttosto vaga e sfuocata.

Si dimostra che per ottenere convergenza, la dimensione della finestra deve essere calcolata tenendo conto del numero di campioni del training set:

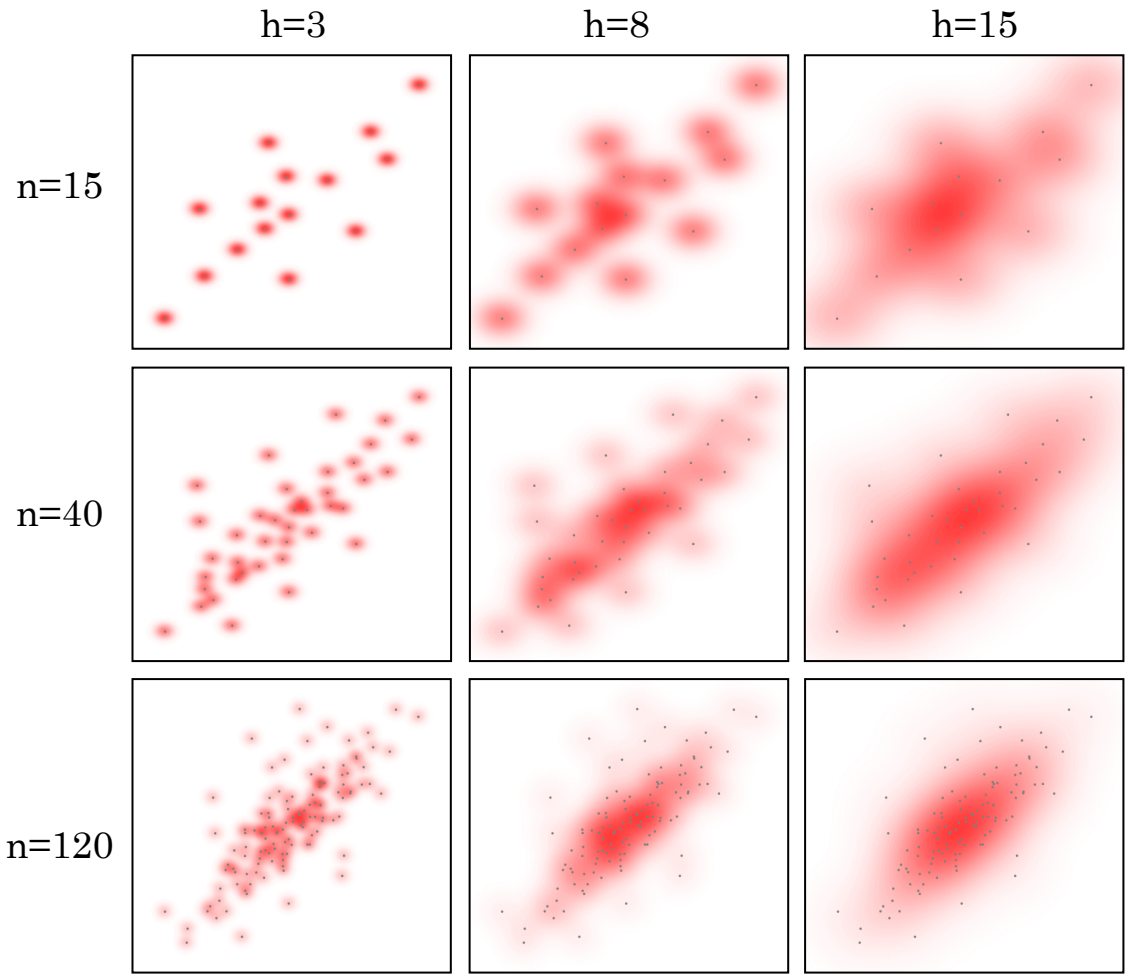
$$V_n = \frac{V_1}{\sqrt{n}}, \qquad \text{dove } V_1 \text{ (o } h_1\text{) è un iperparametro}$$

# Parzen Window con Soft kernel

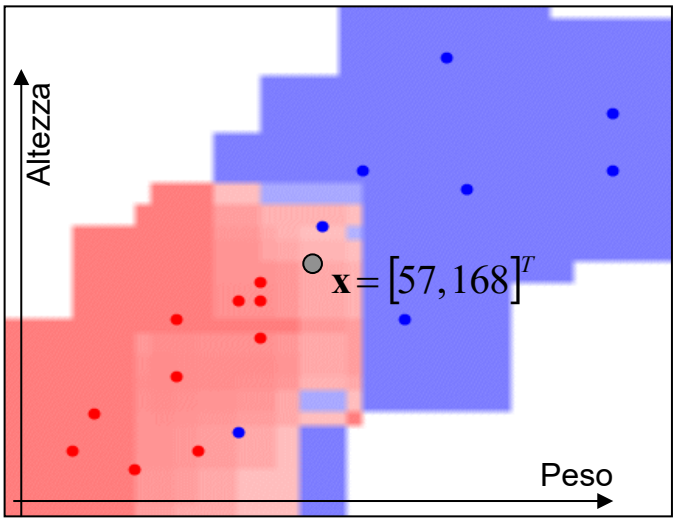
Nella pratica, invece di funzioni finestra ipercubo si utilizzano kernel function più soft grazie alle quali ogni pattern $\mathbf{x}_i$ contribuisce alla stima di densità in un intorno di $\mathbf{x}$ in accordo con la distanza da $\mathbf{x}$. In questo modo le superfici decisionali risultano molto più regolari (smoothed).

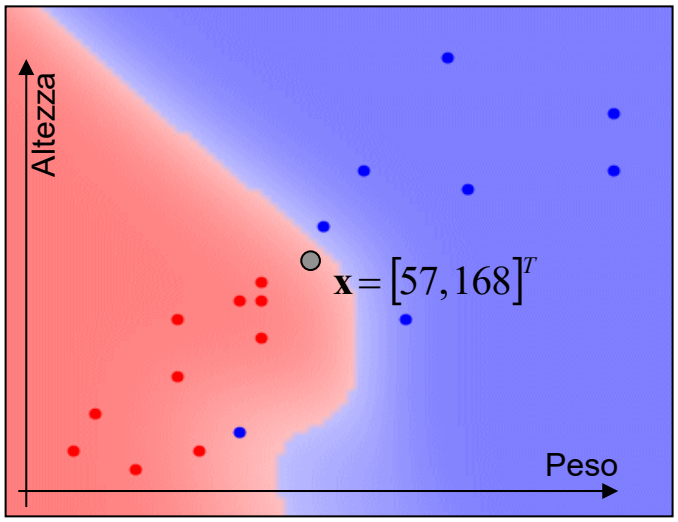
Le kernel function devono essere funzioni densità (sempre ≥ 0 e con integrale su tutto lo spazio uguale a 1). Utilizzando la funzione multinormale (con $\boldsymbol{\mu} = [0 \ldots 0]$ e $\Sigma = \mathrm{I}$):

$$\varphi(\mathbf{u}) = \frac{1}{(2\pi)^{d/2}} e^{-\frac{\mathbf{u}^t\mathbf{u}}{2}}$$



# Maschi/Femmine
# con Bayes + Parzen Window




Stima non-parametrica della densità attraverso Parzen Window (nell'ipotesi che $P(w_1) = 8/18$ , $P(w_2) = 10/18$)

- Funzione Kernel *ipercubo* con $h$ =10 (grafico a sinistra)

$$p(\mathbf{x} \mid w_1) = 0.0038 \qquad p(\mathbf{x} \mid w_2) = 0.0040 \qquad p(\mathbf{x}) = \sum_{i=1}^{s} p(\mathbf{x} \mid \mathrm{w}_i) \cdot P(\mathrm{w}_i) = 0.0039$$

$$P(\mathrm{w}_1 \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \mathrm{w}_1) \cdot P(\mathrm{w}_1)}{p(\mathbf{x})} \cong 0.43 \qquad \boxed{P(\mathrm{w}_2 \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \mathrm{w}_2) \cdot P(\mathrm{w}_2)}{p(\mathbf{x})} \cong 0.57}$$

- Funzione Kernel *normale* con $h$ =3 (grafico a destra)

$$p(\mathbf{x} \mid w_1) = 0.0024 \qquad p(\mathbf{x} \mid w_2) = 0.0041 \qquad p(\mathbf{x}) = \sum_{i=1}^{s} p(\mathbf{x} \mid \mathrm{w}_i) \cdot P(\mathrm{w}_i) = 0.0033$$

$$P(\mathrm{w}_1 \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \mathrm{w}_1) \cdot P(\mathrm{w}_1)}{p(\mathbf{x})} \cong 0.32 \qquad \boxed{P(\mathrm{w}_2 \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \mathrm{w}_2) \cdot P(\mathrm{w}_2)}{p(\mathbf{x})} \cong 0.68}$$

# Classificatore Nearest Neighbor (NN)

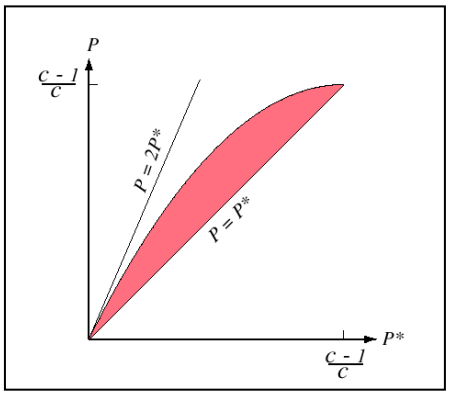
Data una metrica $dist(\cdot)$ nello spazio multidimensionale (es. distanza euclidea) il classificarore nearest neighbor (letteralmente "il più vicino tra i vicini"), classifica un pattern $\mathbf{x}$ con la stessa classe dell'elemento $\mathbf{x}'$ ad esso più vicino nel training set TS:

$$dist(\mathbf{x}, \mathbf{x}') = \min_{\mathbf{x}_i \in \text{TS}} \{ dist(\mathbf{x}, \mathbf{x}_i) \}$$

- Invece di derivare dai dati le distribuzioni condizionali delle classi per poi far uso della regola di Bayes per la classificazione, questo classificatore cerca in modo piuttosto pragmatico di massimizzare direttamente la probabilità a posteriori; infatti se $\mathbf{x}'$ è molto vicino a $\mathbf{x}$ è lecito supporre che:

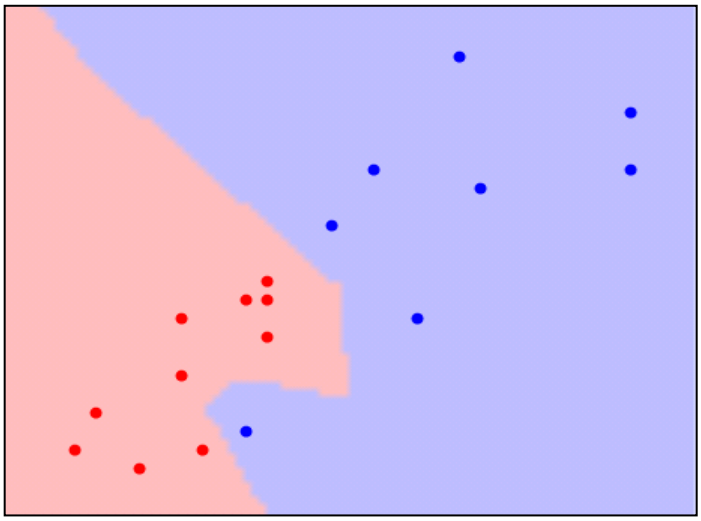
$$P(w_i|\mathbf{x}) \approx P(w_i|\mathbf{x}')$$

- In effetti, si può dimostrare (solo però nel caso di TS popolato da infiniti campioni) che la probabilità di errore *P* (nella figura sotto) della regola nearest neighbor non è mai peggiore del doppio del minimo errore possibile *P** (quello Bayesiano).

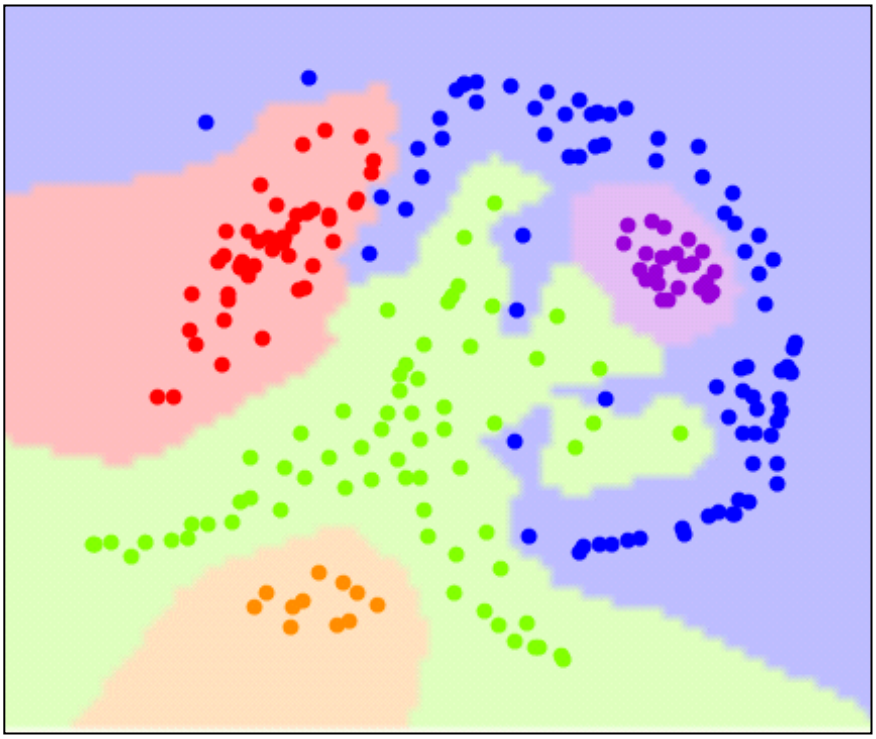



- Nella pratica, questo non significa però che l'approccio Bayesiano fornisca sempre risultati migliori di nearest neighbor, infatti se la stima delle densità condizionali è poco accurata i risultati del classificatore Bayesiano possono essere peggiori.

# Esempi NN

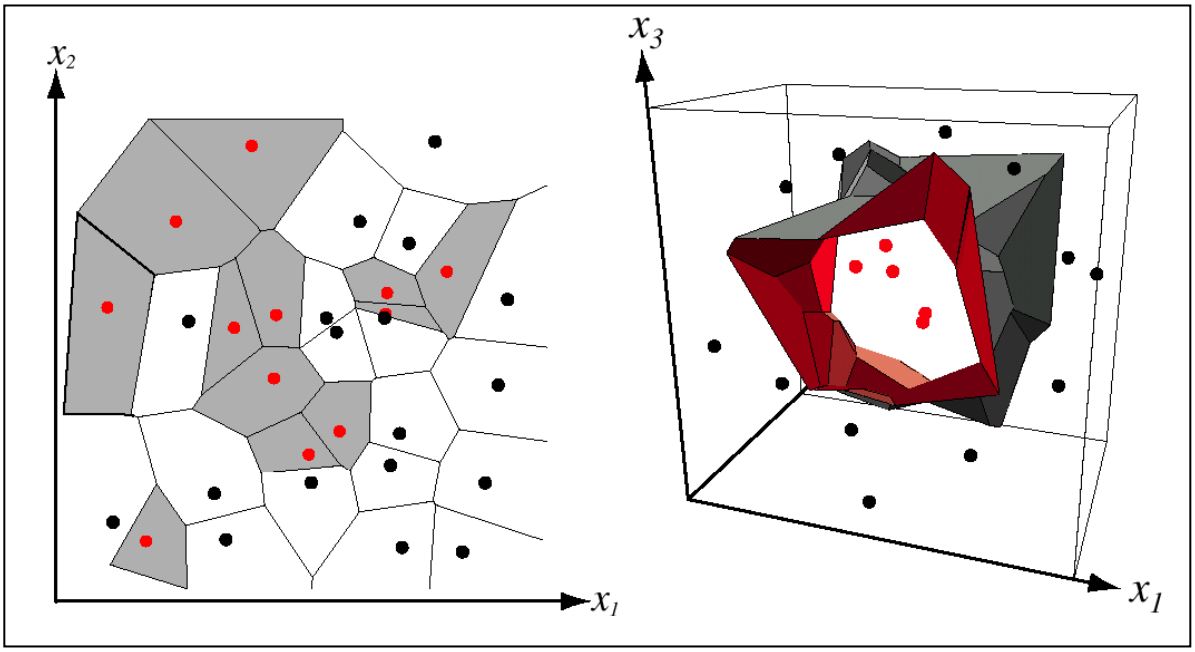


- Nell'esempio visto in precedenza, *la regola NN assegna il pattern* $\mathbf{x}$ *alla classe* $w_1$ *(maschi - blu)*
- La figura seguente mostra il partizionamento dello spazio operato dalla regola NN su un training set con 5 classi:



# Da NN a *k*-NN

- La regola nearest neighbor produce un partizionamento dello dello spazio, noto come *tassellazione di Voronoi*:

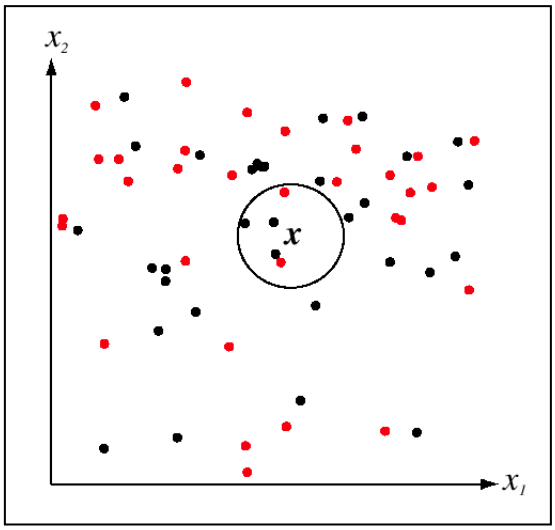



Ogni elemento $\mathbf{x}_i \in TS$ determina un tassello, all'interno del quale i pattern saranno assegnati alla stessa classe di $\mathbf{x}_i$.

- La regola di classificazione nearest neighbor è piuttosto radicale; infatti basta che un elemento del training set non sia molto "affidabile" (outlier) affinché tutti i pattern nelle sue vicinanze siano in seguito etichettati non correttamente.
- *Che errore commette il classificatore NN sul training set?*
- Un modo generalmente più robusto, che può essere visto come estensione della regola nearest-neighbor (in questo caso detta 1-nearest neighbor) è il cosiddetto classificatore *k*-nearest neighbor (*k*-NN).

# *k*-Nearest-Neighbor (*k*-NN)

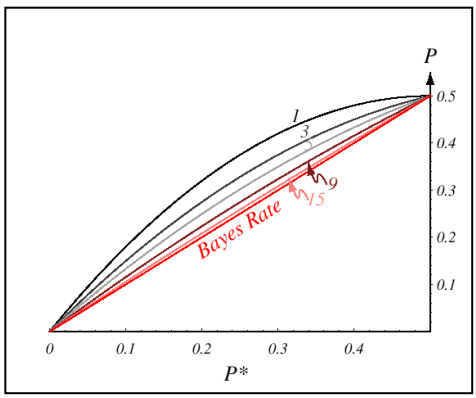
- La regola *k-Nearest Neighbor* (*k*-NN) determina i *k* elementi più vicini al pattern **x** da classificare (*k* è un iperparametro); ogni pattern tra i *k* vicini vota per la classe cui esso stesso appartiene; il pattern **x** viene assegnato alla classe che ha ottenuto il maggior numero di voti.




nella figura il classificatore 5-NN, assegna **x** alla classe "nera" in quanto quest'ultima ha ricevuto 3 voti su 5.

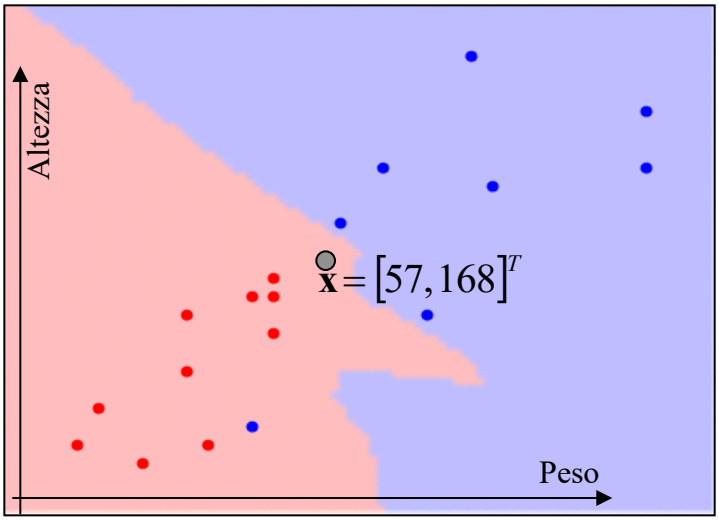
*Nel caso di 2 classi è bene scegliere k dispari per evitare pareggi.*

- Per TS infiniti la regola di classificazione *k*-NN si dimostra migliore di 1-NN, e all'aumentare di *k*, l'errore P converge all'errore Bayesiano.

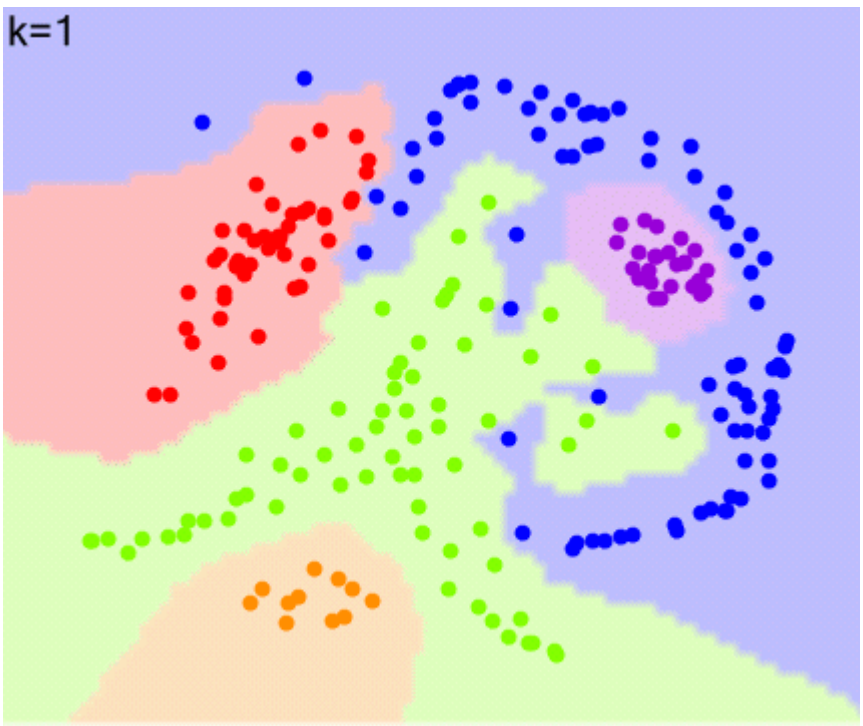



- Nella pratica (TS limitati), aumentare *k* significa estendere l'iper-sfera di ricerca andando a sondare la probabilità a posteriori lontano dal punto di interesse; il valore ottimale di *k* (solitamente < 10) deve essere determinato su un validation set separato.

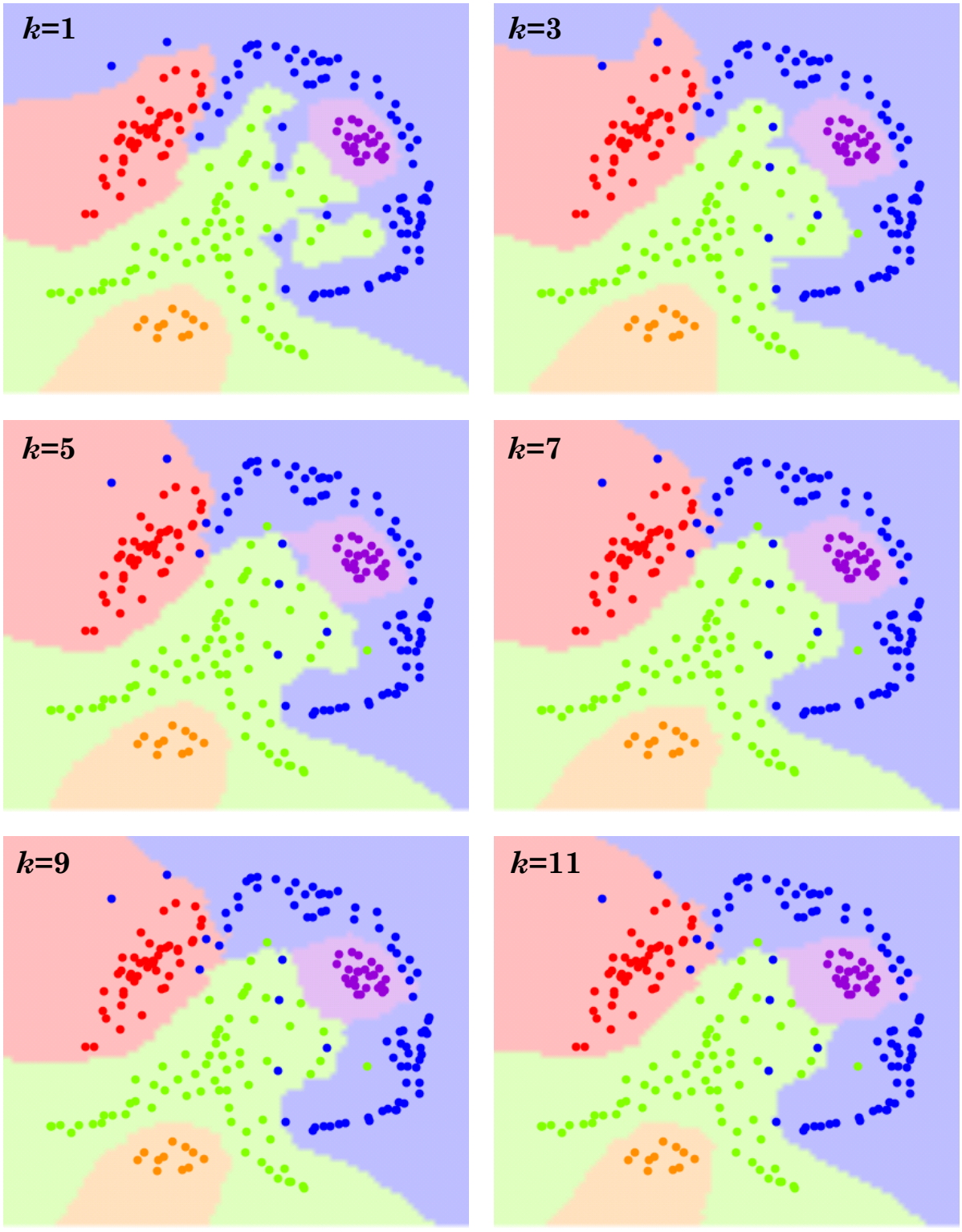
# Esempi *k*-NN




- Nell'esempio visto in precedenza, *la regola k-NN con k=3 assegna il pattern* $\mathbf{x}$ *alla classe* $w_2$ *(femmine - rossi)*
- L'animazione (scomposta nel lucido successivo) mostra il partizionamento dello spazio operato dalla regola *k*-NN sul training set con 5 classi visto in precedenza al variare di *k*



# Espansione dell'animazione lucido precedente (k=1,3,5,7,9,11)



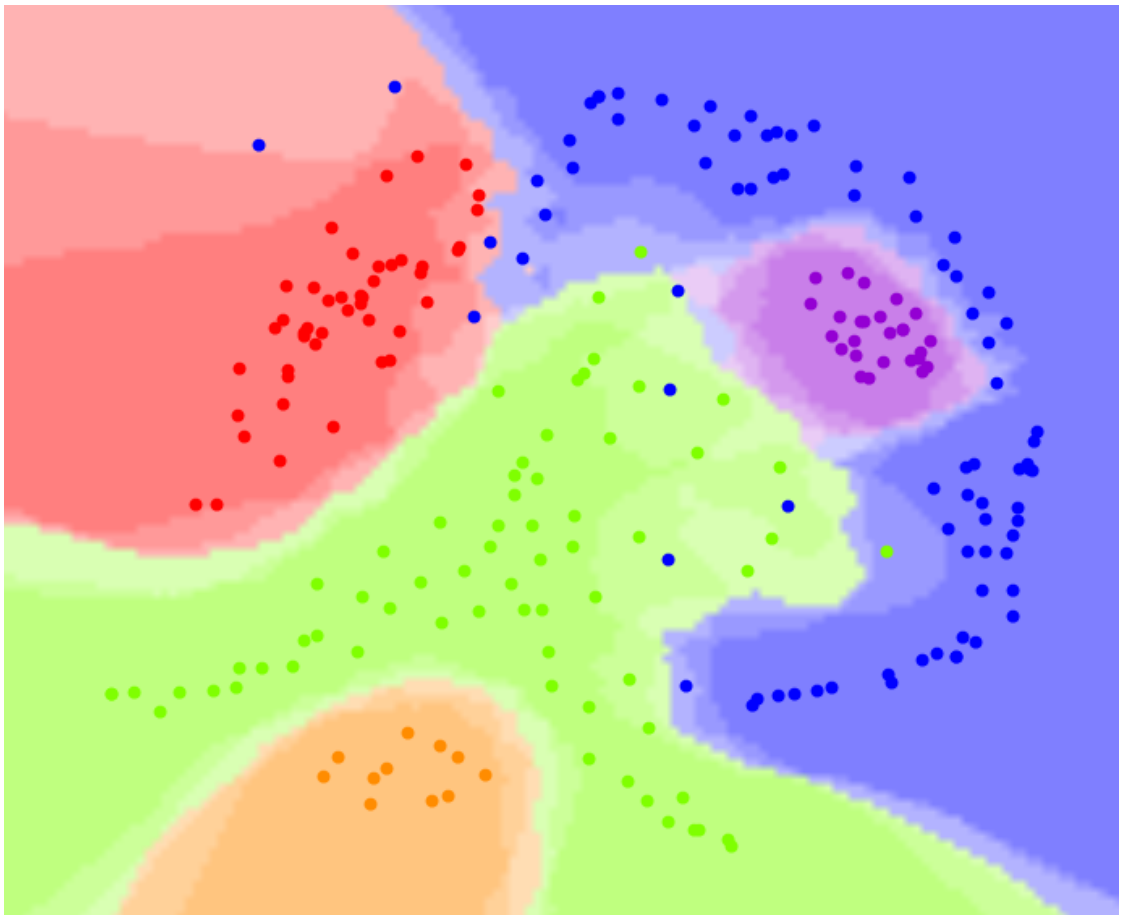
# *k*-NN e Confidenza di Classificazione

Da un classificatore *k*-NN risulta piuttosto semplice estrarre una *confidenza* (probabilistica) circa la classificazione eseguita; siano

$$[v_1, v_2 \dots v_s], \qquad \sum_{i=1}^{s} v_i = k$$

i voti ottenuti dal pattern $\mathbf{x}$, allora le confidenze (*vedi sfumature in figura sotto*) possono essere semplicemente ottenute dividendo per $k$ i voti ottenuti:

$$\left[\frac{v_1}{k}, \frac{v_2}{k} \dots \frac{v_s}{k}\right]$$



# NN e complessità computazionale

L'utilizzo di un classificatore NN o *k*-NN nel caso di training set di elevate dimensioni può diventare problematico:

- Necessario memorizzare tutti i pattern del Training Set
- Per ogni classificazione è necessario calcolare la distanza del pattern da classificare da tutti i pattern del training set e ordinare (parzialmente le distanze) per ottenere le più piccole

Tecniche di editing/condensing (lucido successivo) possono alleviare questo problema, ma quando l'efficienza è importante è consigliabile indicizzare i dati attraverso strutture spaziali (es. kd-tree) che consentono di individuare i vicini senza effettuare una scansione esaustiva.

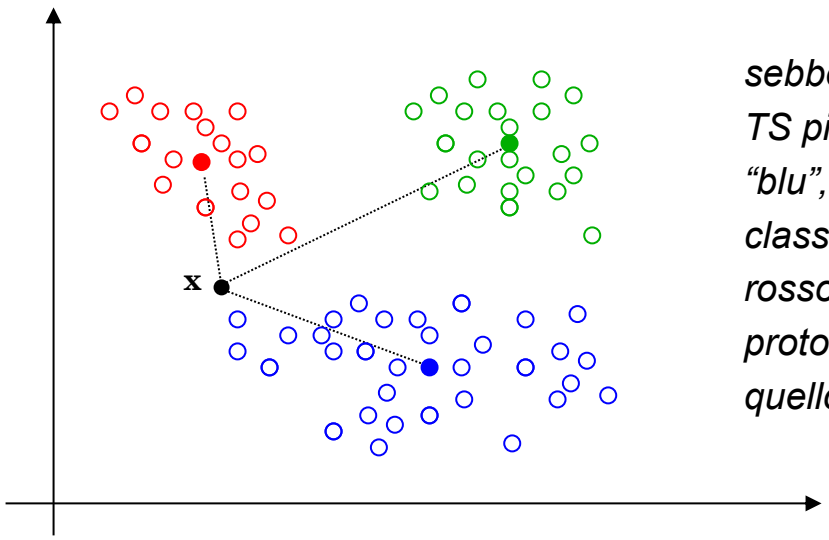
La libreria FLANN (C++) consente di effettuare ricerche nearest neighbor approssimate molto efficientemente.

*http://www.cs.ubc.ca/research/flann/*

# NN e Prototipi di Classi

Talvolta nella classificazione nearest neighbor invece di mantenere tutti i pattern del TS e calcolare la distanza da ciascuno di essi, si preferisce selezionare/derivare da essi uno (o più) *prototipi* per ciascuna classe e utilizzare questi ultimi per la classificazione come se fossero i soli elementi di TS: queste tecniche prendono il nome di:

- *editing*: quando si cancellano solo pattern dal training set, senza derivare nuovi pattern
- *condensing*: se i prototipi non appartenevano al TS e sono stati derivati




*sebbene il pattern di TS più vicino a* **x** *sia "blu",* **x** *viene classificato come rosso, in quanto il prototipo più vicino è quello rosso.*

Ciò comporta solitamente i seguenti vantaggi:

- non è necessario calcolare un elevato numero di distanze.
- i prototipi sono spesso più affidabili e robusti di singoli pattern (si riduce il rischio di affidarsi ad outlier).

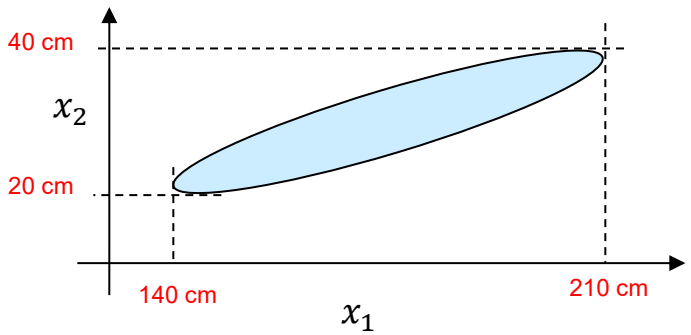
Un singolo prototipo di classe può essere derivato come vettore medio dei vettori di quella classe nel TS. Processi di vector quantization o clustering consentono di ottenere più prototipi per ogni classe.

# NN e Metriche

- Il comportamento della regola *k*-NN è strettamente legato alla metrica (funzione distanza) adottata.
- La distanza euclidea, che rappresenta il caso $L_2$ nella definizione di metriche di Minkowski, è sicuramente la metrica più spesso utilizzata.

$$L_k(\mathbf{a}, \mathbf{b}) = \left( \sum_{i=1}^{d} |a_i - b_i|^k \right)^{1/k}$$

- Nella pratica, prima di adottare semplicemente la distanza euclidea è bene valutare lo spazio di variazione delle componenti (o feature) e la presenza di eventuali forti correlazioni tra le stesse.
- Supponiamo ad esempio di voler classificare le persone sulla base dell'altezza e della lunghezza del piede. Ogni pattern $\mathbf{x}$ (bidimensionale) risulta costituito da due feature ($x_1$ = altezza, $x_2$ = lunghezza del piede).




Lo spazio di variazione dell'altezza (210-140 = 70 cm) risulta maggiore di quello della lunghezza del piede (40-20=20 cm). Pertanto se la similarità tra pattern venisse misurata con semplice distanza euclidea la componente altezza "peserebbe" più della componente lunghezza del piede.

# Normalizzazione

Per evitare i problemi legati a diversi spazi di variazioni delle feature, particolarmente fastidiosi per alcune tecniche (es. reti neurali), si consiglia di normalizzare i pattern.

Le normalizzazioni più comuni sono:

- Min-Max scaling: per ogni feature $i-esima$ si calcolano il massimo $max_i$ e il minimo $min_i$ e si applica una trasformazione lineare (scaling) che «tipicamente» mappa $min_i$ a 0 e $max_i$ a 1.

$$x' = (x - min_i)/(max_i - min_i)$$

- Standardization: per ogni feature $i-esima$ si calcola la media $mean_i$ e la deviazione standard $stddev_i$ e si trasformano i valori come:

$$x' = (x - mean_i)/stddev_i$$

  Dopo la trasformazione tutte le feature hanno (sul training set) media 0 e deviazione standard 1.

Attenzione: i parametri della normalizzazione (es. minimi, massimi) si calcolano sul solo training set e la trasformazione si applica sia a tutti i dati (training, validation, test). Nel caso di k-fold cross-validation la normalizzazione dovrebbe essere ripetuta k volte (uso di *pipeline* in scikit-learn).

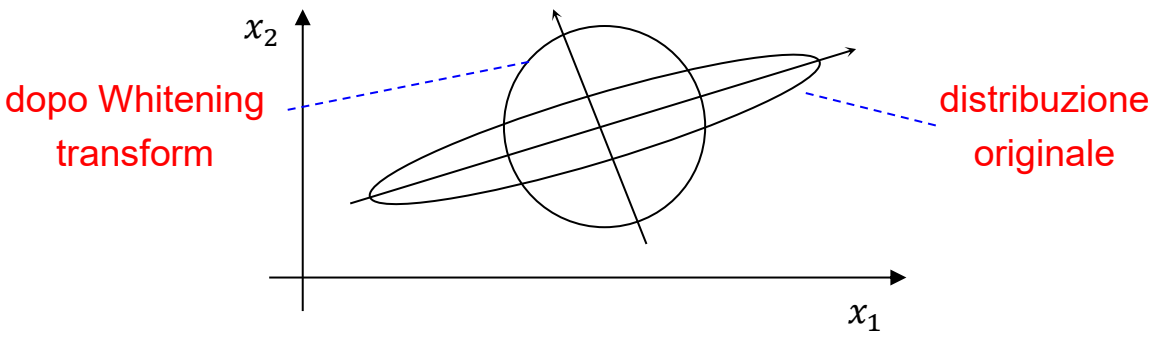
Le semplici tecniche sopra descritte operano sulle singole feature indipendentemente. Una tecnica di normalizzazione efficace (ma più costosa) che opera simultaneamente su tutte le feature tenendo conto della loro correlazione è la Whitening transform (o PCA whitening). Molto in voga prima del deep-learning, oramai poco utilizzata.

# Whitening transform

Un'efficace normalizzazione rispetto agli spazi di variazione, in grado anche di tener conto delle correlazioni tra feature è possibile:

- pre-normalizzando lo spazio delle feature attraverso Whitening transform (che vedremo meglio in seguito)
- utilizzando come metrica la *distanza di Mahalanobis*.

Le due alternative sono equivalenti. Nel primo caso l'ellissoide corrispondente allo spazio delle feature viene "sfericizzato" a priori e viene in seguito usata la distanza euclidea; nel secondo la distanza di Mahalanobis normalizza ogni componente sulla base della matrice di covarianza $\Sigma$.




Da non sottovalutare l'importanza della correlazione tra features come aspetto negativo per la classificazione. Infatti, l'utilizzo di feature correlate riduce (anche drasticamente) il potere discriminante. Nel caso ideale tutte le feature sono staticamente indipendenti (ellissoide assi paralleli a quelli cartesiani).

*Due feature altamente discriminanti se prese individualmente, ma tra loro fortemente correlate, sono nel complesso meno discriminanti di una terza feature leggermente più di discriminante di ognuna delle precedenti.*

La distanza di Mahalanobis (o la sfericizzazione dello spazio) tiene conto delle correlazioni e pesa maggiormente feature non correlate.

# Metric Learning

Un approccio più generale alla scelta della metrica da utilizzare in una determinata applicazione, consiste nel learning supervisionato della metrica stessa dai dati del training set.

Obiettivo è determinare una trasformazione degli input che:

- «avvicini» pattern della stessa classe
- «allontani» pattern di classi diverse

La distanza euclidea nella spazio originale è:

$$dist(a,b) = \sqrt{(\mathbf{a}-\mathbf{b})^t(\mathbf{a}-\mathbf{b})} = \|\mathbf{a}-\mathbf{b}\|_2$$

Un tipico approccio di metric learning lineare determina (con training supervisionato) una matrice $\mathbf{G}$ che trasforma gli input, e continuare ad applicare la distanza euclidea agli input trasformati

$$dist(a,b) = \|\mathbf{Ga}-\mathbf{Gb}\|_2$$

Vedremo una possibile soluzione di questo problema nell'ambito della riduzione di dimensionalità con LDA (Linear Discriminant Analysys).

Sono anche possibili approcci non lineari:

$$dist(a,b) = \|\mathbf{G}\phi(\mathbf{a})-\mathbf{G}\phi(\mathbf{b})\|_2$$

dove $\phi$ è una funzione non lineare.

# Similarità Coseno e Distanza Coseno

Una similarità/distanza piuttosto utilizzata in applicazioni di information retrieval, data mining e text mining è la similarità/distanza coseno.

Geometricamente, dati due vettori $\mathbf{a}$ e $\mathbf{b}$ la similarità coseno corrisponde al coseno dell'angolo tra di essi:

$$CosSimilarity(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a}^t \cdot \mathbf{b}}{\|\mathbf{a}\| \cdot \|\mathbf{b}\|}$$

è noto infatti che il prodotto scalare tra due vettori è:

$$\mathbf{a}^t \cdot \mathbf{b} = \|\mathbf{a}\| \cdot \|\mathbf{b}\| \cdot cos(\theta)$$

Due vettori identici hanno similarità 1 e due vettori opposti -1.

La distanza coseno è semplicemente:

$$CosDistance(\mathbf{a}, \mathbf{b}) = 1 - CosSimilarity(\mathbf{a}, \mathbf{b})$$

*Esempio Confronto di testi: Un testo può essere codificato da un vettore numerico dove ogni dimensione contiene il numero di occorrenze di una certa parola rispetto a un dato dizionario. La similarità di contenuto tra due testi non dipende dal numero assoluto di parole ma dalla frequenza relativa di ciascuna di esse. La distanza coseno «sconta» la lunghezza dei vettori.*

La distanza coseno non è una metrica (es. non rispetta la diseguaglianza triangolare). Se si è interessati a una metrica si può passare alla distanza angolare:

$$AngularDistance(\mathbf{a}, \mathbf{b}) = \frac{cos^{-1}\big(CosSimilarity(\mathbf{a}, \mathbf{b})\big)}{\pi}$$