Introduzione alla classificazione funzionale di tessuti e geni con metodi di apprendimento automatico

Giorgio Valentini

e-mail: valentini@dsi.unimi.it

Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano

Sommario

- Classificazione funzionale di tessuti e geni su base bio-molecolare come problema di apprendimento automatico
- Metodi di apprendimento automatico per la classificazione di stati funzionali e per il supporto alla diagnosi bio-molecolare
- Stima della qualità dei classificatori per l' analisi dei dati di espressione genica

Prima parte:

Classificazione funzionale di tessuti e geni su base bio-molecolare come problema di apprendimento automatico

Classificazione bio-molecolare di tessuti e geni

- Diagnosi a livello bio-molecolare:
  - Identificazione e classificazione di pazienti sani e malati.
  - Classificazione di differenti tipologie patologiche
  - Diagnosi basata sulla integrazione di dati biologici eterogenei
- Predizione di esiti clinici:
  - Predizione delle risposte di pazienti a trattamenti farmacologici
  - Sviluppo di tool prognostici per uso clinico e per la risposta a terapie
- Identificazione di molecole target per lo sviluppo di farmaci
- Classificazione di classi funzionali di geni su base bio-molecolare.
- Analisi di qualità di prodotti agro-alimentari.

Classificazione di stati e classi funzionali come problema di apprendimento automatico (1)

I dati generati da bio-tecnologie high-throughput (ad es: DNA microarray) sono rappresentabili come insiemi di coppie (**x**,t):

- $\mathbf{x} \in R^d$, $\mathbf{x} = [x_1, x_2, \ldots, x_d]$ rappresenta i livelli di espressione genica di *d* geni
- t rappresenta un particolare stato funzionale

Es: $t \in C = \{ s, m \}$, s → paziente sano, m→malato

Classificazione di stati e classi funzionali come problema di apprendimento automatico (2)

Obiettivo dell' apprendimento automatico:

Apprendere la funzione non nota f :
f : $R^d \rightarrow$ C che mappa i livelli di espressione genica $\mathbf{x} \in R^d$
nella corrispondente classe funzionale $\mathrm{t} \in \mathrm{C}$ (es: paziente sano o malato)

tramite un algoritmo di apprendimento (learning machine) L che utilizza solo un training set $D = \{(\mathbf{x}_i, t_i)\}_{i=1}^{n}$ di campioni distribuiti in accordo alla distribuzione di probabilità congiunta P(**x**,t).

Algoritmi di apprendimento supervisionato e learning machine

• L' algoritmo di apprendimento L genera un' approssimazione $g : R^d \rightarrow$ C della funzione non nota f utilizzando il training set D: $L(D) \rightarrow g$.

• Si desidera che tale funzione sia la più "vicina" possibile ad f

• A tal fine si usa una funzione di perdita Loss$(f(\mathbf{x}),g(\mathbf{x}))$ che misuri quanto g differisca da f.

• Nei problemi di classificazione si usa la funzione di perdita 0/1:

$$Loss\big(g(\mathbf{x}), f(\mathbf{x})\big) = \begin{cases} 1 & se \quad g(\mathbf{x}) \neq f(\mathbf{x}) \\ 0 & se \quad g(\mathbf{x}) = f(\mathbf{x}) \end{cases}$$

Ma f non è nota (se lo fosse avremmo risolto il problema) ...

Addestramento delle learning machine

- Nella realtà si dispone spesso solo di un insieme relativamente limitato di dati (ad es: un insieme *D* di dati di espressione genica) e la learning machine viene addestrata ad approssimare *f* utilizzando tali dati come una serie di esempi da apprendere:

$(\mathbf{x}_1,t_1)$, $(\mathbf{x}_2,t_2)$, ... $(\mathbf{x}_n,t_n)$.

- La learning machine verrà addestrata ad apprendere una funzione *g* tale che $g(\mathbf{x}_1)=t_1$, $g(\mathbf{x}_2)=t_2$, ..., $g(\mathbf{x}_n)=t_n$, in modo da minimizzare il rischio empirico R_{emp} rispetto al training set $D=\{(\mathbf{x}_i,t_i)\}_{i=1}^{n}$:

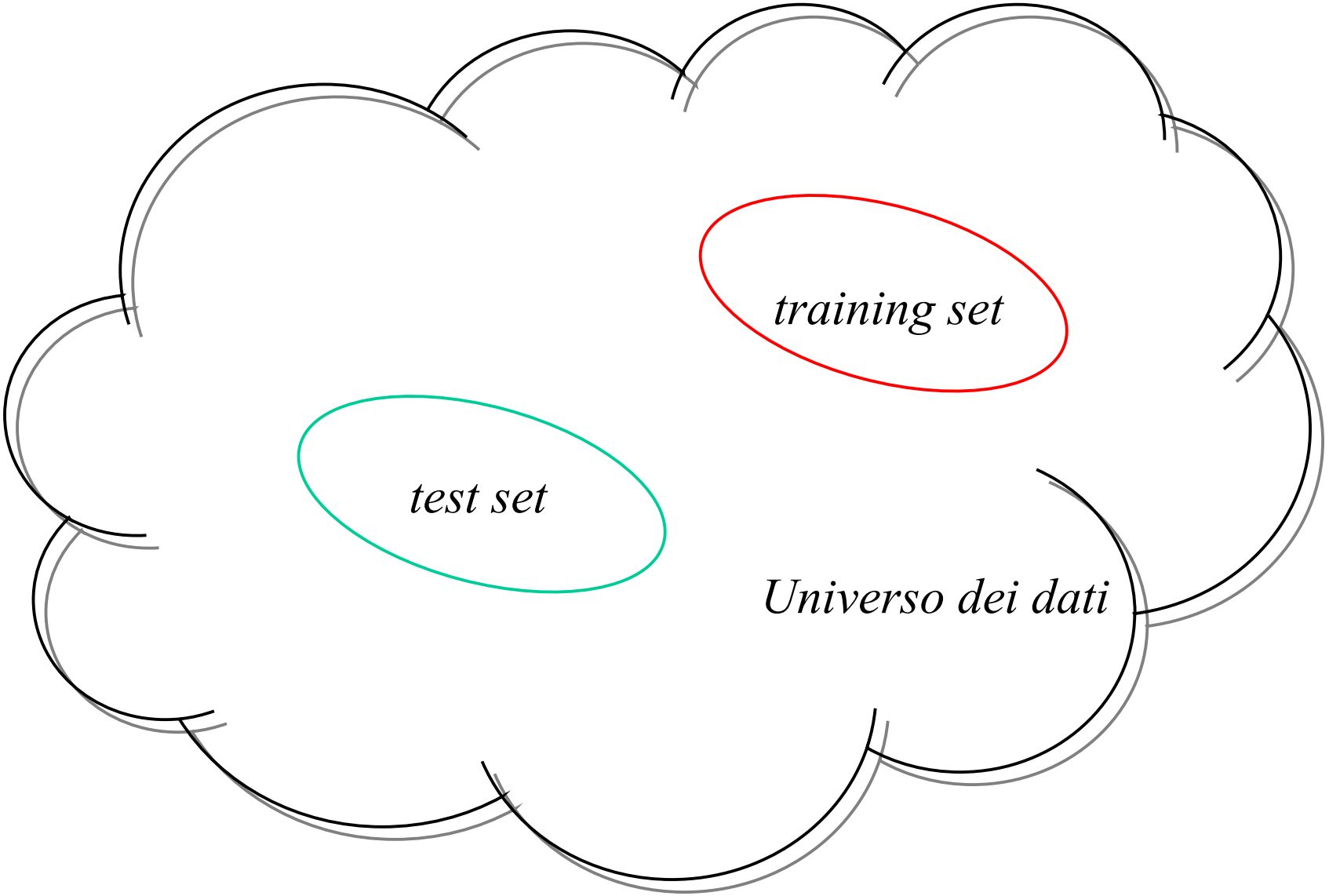
$$R_{emp}=\frac{1}{n}\sum_{i=1}^{n}Loss(g(\mathbf{x}_i),t_i)$$

Generalizzazione

- L viene addestrata su un *training set* $D \subset U$.
- La learning machine L è utile se può fare delle previsioni sull' Universo non noto U dei dati:

  vogliamo cioè che *generalizzi* correttamente su dati che non conosce.
- A questo fine L deve prevedere correttamente non tanto i dati D su cui è stata addestrata, ma i dati $(\mathbf{x},t) \in U$, $(\mathbf{x},t) \notin D$ che non conosce.
- Siccome di solito non si conosce a priori U o equivalentemente la distribuzione di probabilità congiunta $P_U(\mathbf{x},t)$, le capacità di generalizzazione di L vengono valutate rispetto ad un test set T separato da D, cioè tale che $T \subset U$ e $T \cap D = \varnothing$.

Universo dei dati e campioni



Apprendimento supervisionato

- *Apprendimento da dati "etichettati"*: ciascun campione viene etichettato (ad es: normale o malato)
- *Supervisionato* in quanto un "supervisore" assegna le etichette ai campioni da apprendere: cioè la learning machine è addestrata tramite un insieme di dati etichettati ($\mathbf{x}$,t)
- La learning machine impara ad associare un determinato campione $\mathbf{x}$ ad una classe t
- L' obiettivo della learning machine consiste nell' *assegnare un' etichetta corretta a campioni la cui classe di appartenenza non è nota a priori* (ad es: deve essere in grado di predire sulla base dei dati di espressione genica se un paziente sia sano o malato)

Obiettivo dell' apprendimento supervisionato

- Consiste nel predire correttamente la classe si appartenenza di campioni non noti (*generalizzazione*).

La generalizzazione dipende da:

- Accuratezza del classificatore sul training set
- Complessità della funzione computata dal classificatore
- Dimensione training set

Moreover from classical statistics we know that the *overfitting problem* may arise with small sized and high dimensional data

Caratteristiche dei dati di espressione genica

- Campioni di ridotta cardinalità
- Elevata dimensionalità dei dati
- Rumore
- Dati mancanti

Un problema complesso di apprendimento automatico

Seconda parte:

Metodi di apprendimento automatico per la classificazione di stati funzionali e per il supporto alla diagnosi bio-molecolare

Metodi di apprendimento supervisionato per l' analisi dei dati di espressione genica

Problemi biologici

- Predizione dello stato funzionale dei tessuti
- Predizione di classi funzionali di geni
- Diagnosi dei tumori
- Predizione della risposta a terapie
- Identificazione di target per i farmaci
- Identificazione di geni correlati a malattie

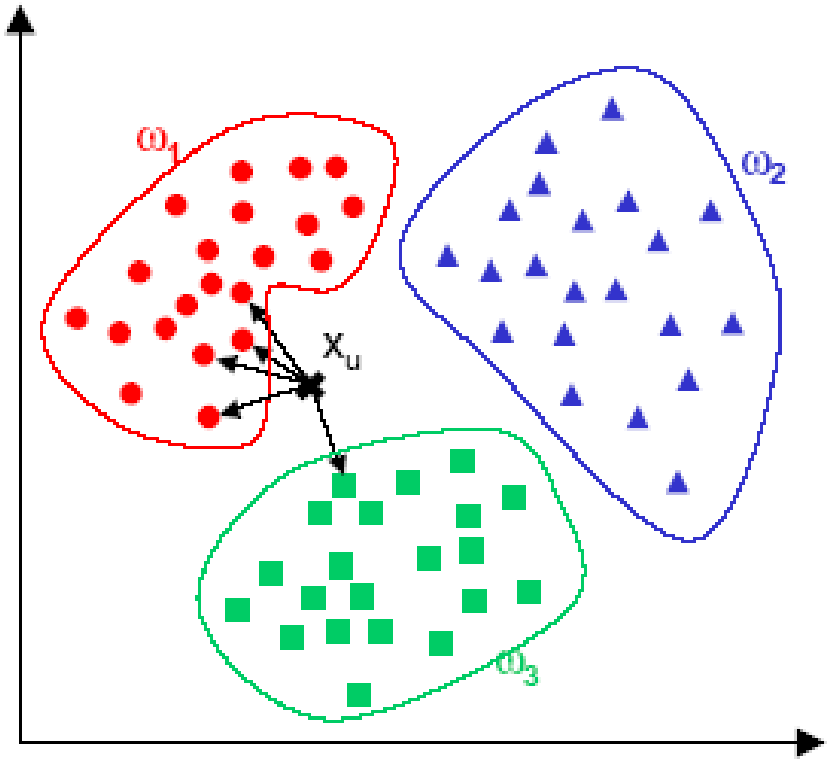
Metodi supervisionati

- Alberi di decisione
- Discriminanti lineari
- Percettroni multistrato
- Classificatori Nearest-Neighbours
- Support Vector Machine
- Metodi basati su kernel
- Metodi di ensemble

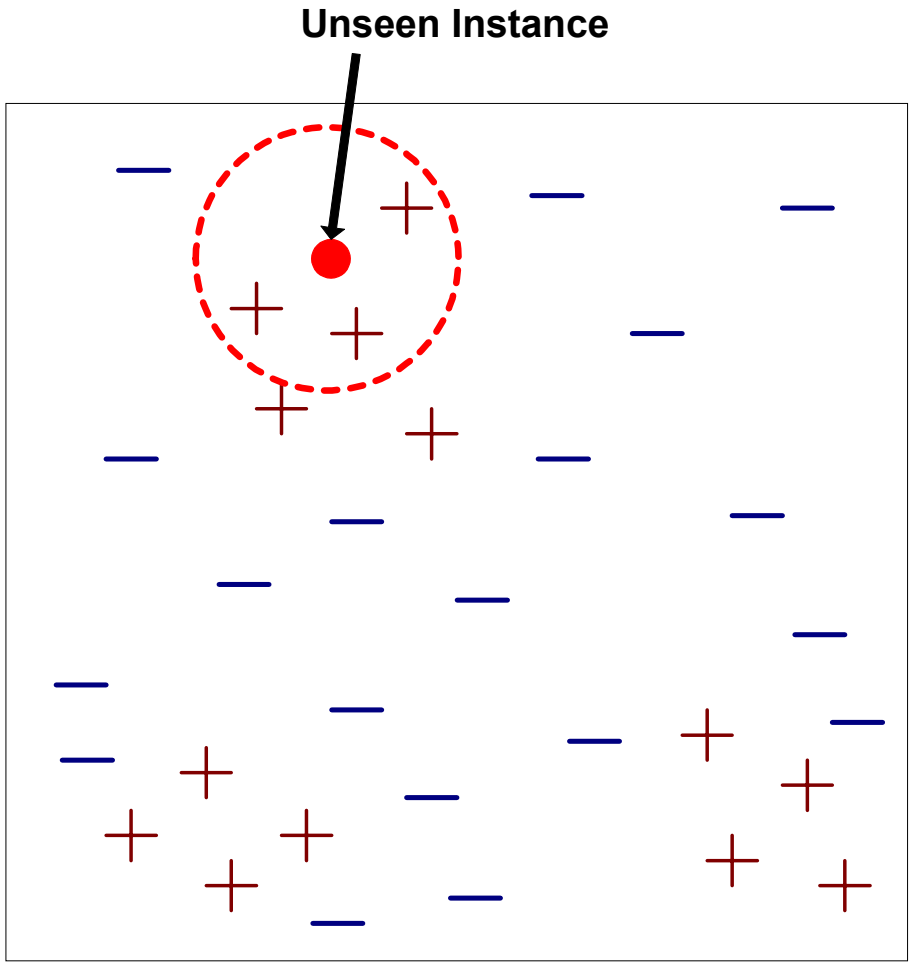
Metodi supervisionati per l' analisi di dati di espressione genica

- Discriminanti lineari e quadratici (Dudoit et al., 2002)
- K-Nearest Neighbours (Pomeroy et al., 2002)
- Reti neurali (Khan et al. 2001)
- Support Vector Machine (Brown et al. 2000, Furey et al. 2000)
- Alberi di decisione (Dudoit et al. 2000)
- Metodi di ensemble (Dudoit et al., 2002; Valentini et al., 2003)

K-nearest-neighbour



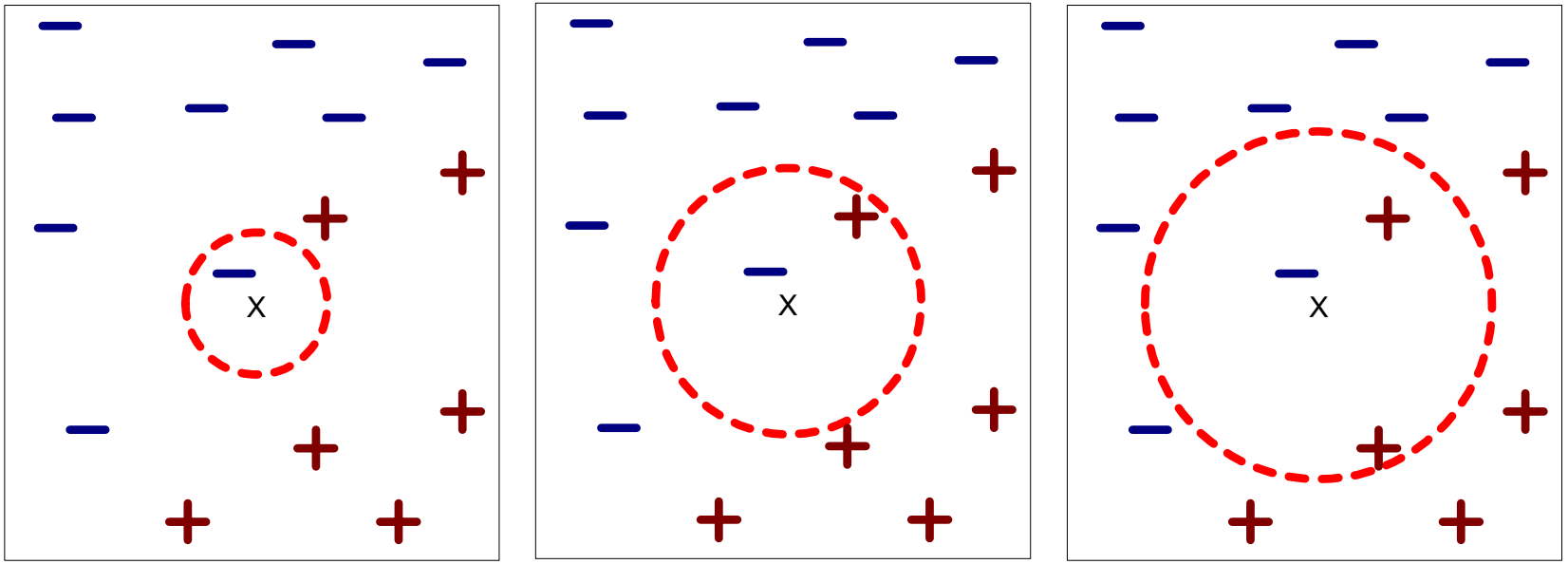
Classificatore Nearest-Neighbor



- L' *apprendimento* richiede:
  - Un insieme di pattern (**x**,*t*) di dati di espressione genica (training set)
  - Una metrica per calcolare le distanze fra pattern
  - Il valore di k, cioè del numero dei primi k vicini.

- La *classificazione* di un pattern **x** la cui classe *t* di appartenenza non è nota richiede :
  - La ricerca dei primi *k* vicini a **x**
  - Assegnamento ad **x** della classe *t* maggiormente frequente nei primi *k* vicini (voto a maggioranza)

Esempio di k-Nearest Neighbor

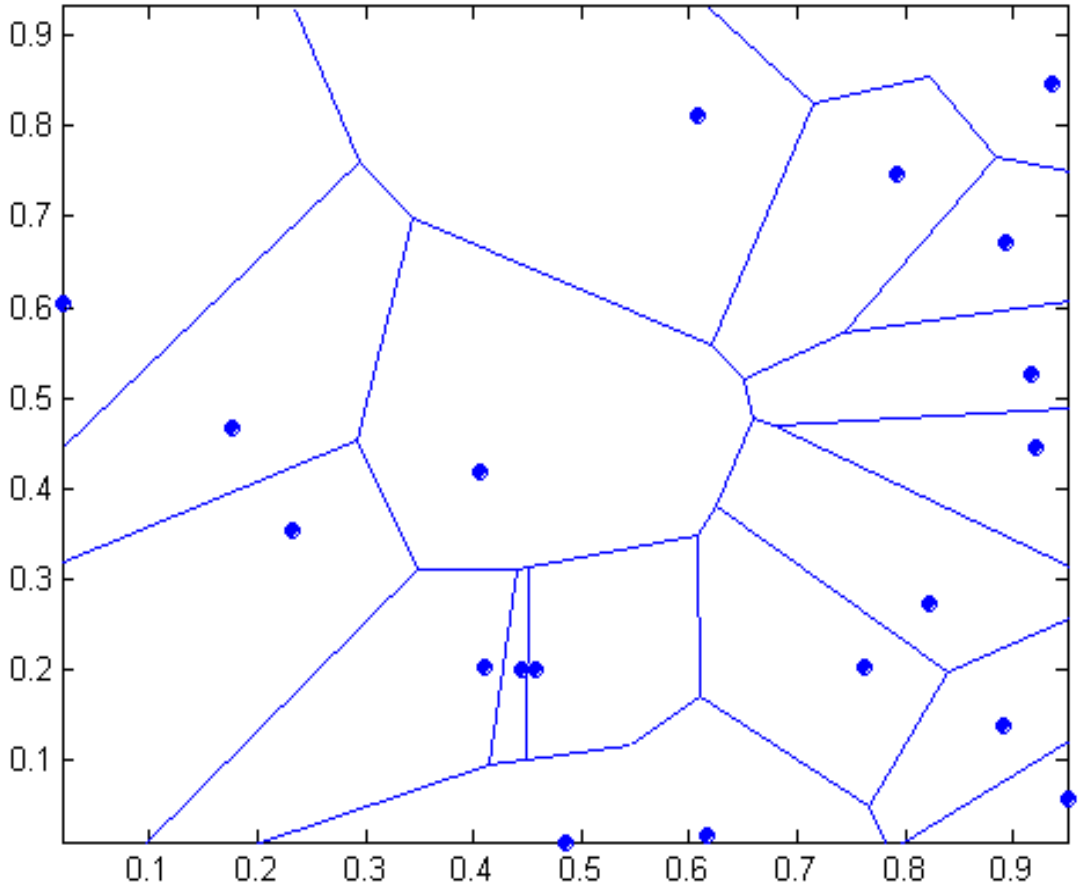


(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

I k-nearest neighbors di un campione x sono i campioni che hanno le k minori distanza da x.

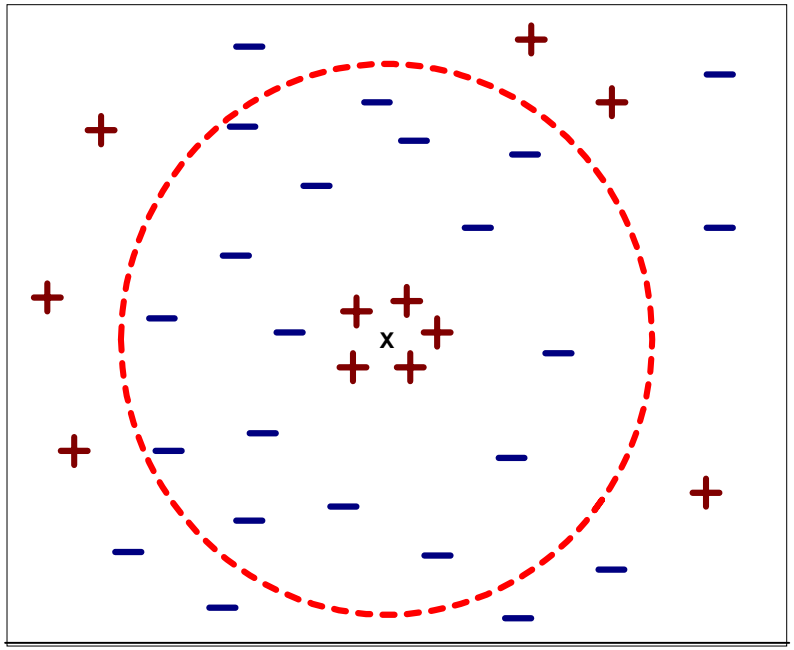
1- nearest-neighbor

E’ in relazione biunivoca con i diagrammi di Voronoi



Il parametro k in k-NN

- Se k è troppo piccolo la classificazione può essere sensibile a dati rumorosi
- Se k è troppo grande:
  - può essere computazionalmente costosa
  - l' intorno può includere campioni appartenenti ad altre classi

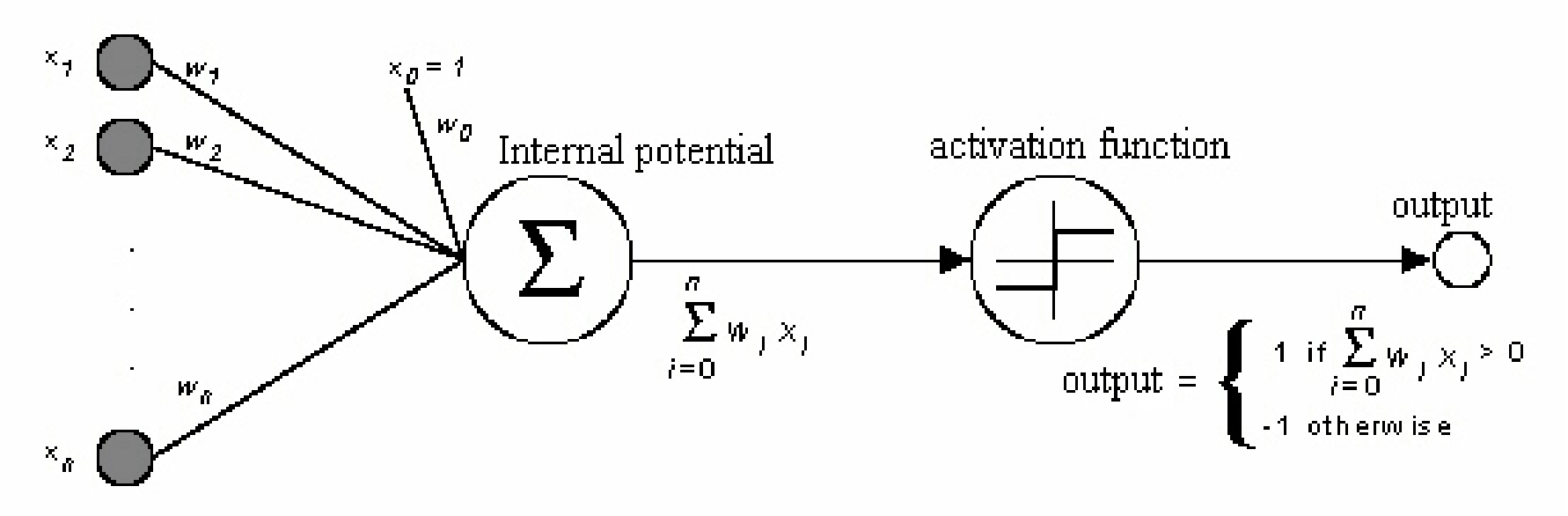


Limiti dei k-NN

- I classificatori k-NN sono “pigri”
  - non costruiscono esplicitamente un modello (come ad es: fanno gli alberi di decisione o le reti neurali)
  - la classificazione può essere realtivamente costosa

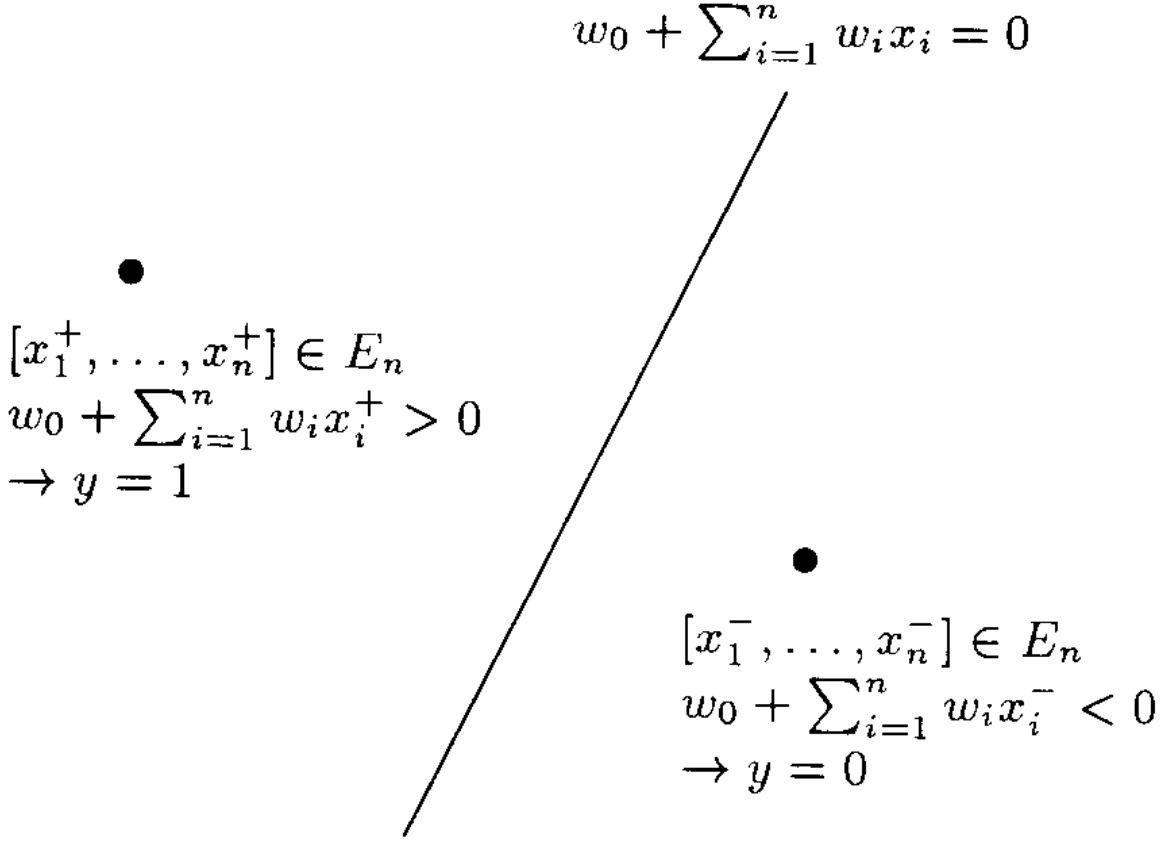
- I dati di espressione genica sono di elevata dimensionalità e k-NN è soggetto al problema del *curse of dimensionality*.

Percettrone (modello lineare di neurone)



- $x_1, ..., x_n$ - input
- $w_1, ..., w_n$ – pesi sinaptici
- w_0 – fattore costante (bias)
- $\sigma(\xi)$ – funzione di attivazione: $\sigma(\xi) = \mathrm{sgn}(\xi)$
- y- output: $y = \sigma(\xi)$

Interpretazione geometrica della funzione computata da un singolo percettrone

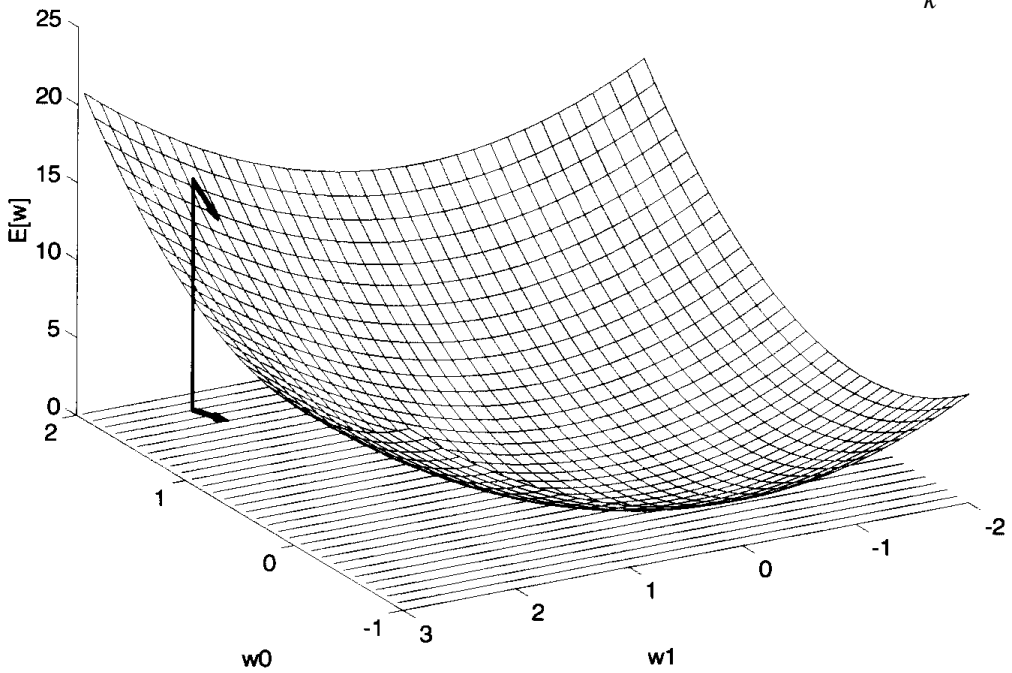


- Un percettrone effettua un classificazione in 2 classi utilizzando un iperpiano (una retta in 2 dimensioni) come superficie separatrice: $\boldsymbol{w}\,\boldsymbol{x} = 0$
- Può classificare correttamente solo insiemi linearmente separabili

Apprendimento del percettrone: minimizzazione dell' errore

- Come calcolare il vettore dei pesi ***w*** dell' iperpiano separatore ? Si minimizza l' errore della funzione $y_k = \mathrm{sgn}(w \cdot x_k)$ computata dal percettrone calcolata rispetto al training set $T = \{(\mathbf{x}_k, t_k)\}_{k=1}^{n}$:

$$E(w) \equiv \frac{1}{2} \sum_{t_k \in T} (t_k - y_k)^2$$



Il vettore dei pesi viene modificato in modo da minimizzare $E(w)$.

Minimizzazione del vettore dei pesi tramite discesa a gradiente

- Idea di fondo: spostarsi sulla superficie di errore verso il minimo.
- La direzione è determinata dalla derivata di E rispetto a ciascuna componente di $\boldsymbol{w}$.

  Gradiente di E: $\nabla E(w) \equiv \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \ldots, \frac{\partial E}{\partial w_n}\right]$

- Apprendimento per discesa a gradiente:

  $w_i \leftarrow w_i + \Delta w_i$, dove $\Delta w_i = -\eta \frac{\partial E}{\partial w_i}$

- Differenziando E rispetto ai pesi w_i:

$$\frac{\partial E}{\partial w_i} = \frac{\partial}{\partial w_i} \frac{1}{2} \sum_{t_k \in T} (t_k - y_k)^2 = \sum_{t_k \in T} (t_k - y_k)(-x_i)$$

Apprendimento del neurone: algoritmo iterativo di discesa a gradiente

1. Inizializzazione di ciascun peso w_i a valori casuali (piccoli)
2. Finchè non si raggiunge la condizione di stop:
   - $\Delta w_i = 0,\ 1 \leq i \leq d$
   - per ciascun $(\boldsymbol{x}_k, t_k) \in T$:
     - Calcola $y_k = \boldsymbol{w}\boldsymbol{x}_k$
     - Per ciascun peso w_i :

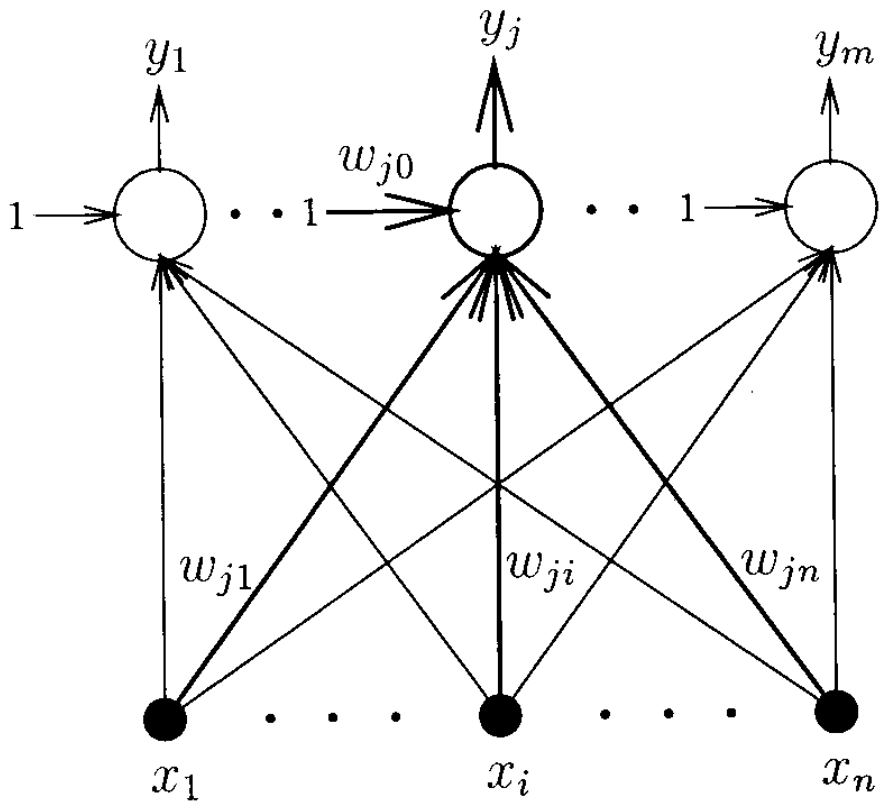
       $\Delta w_i \leftarrow \eta(t_k - y_k)\, x_i$

       $w_i \leftarrow w_i + \Delta w_i$

Possibili condizioni di stop:

- $E(\boldsymbol{w})$ < soglia prefissata
- $\Delta w_i \rightarrow 0,\ 1 \leq i \leq d$
- Numero di iterazioni > soglia prefissata

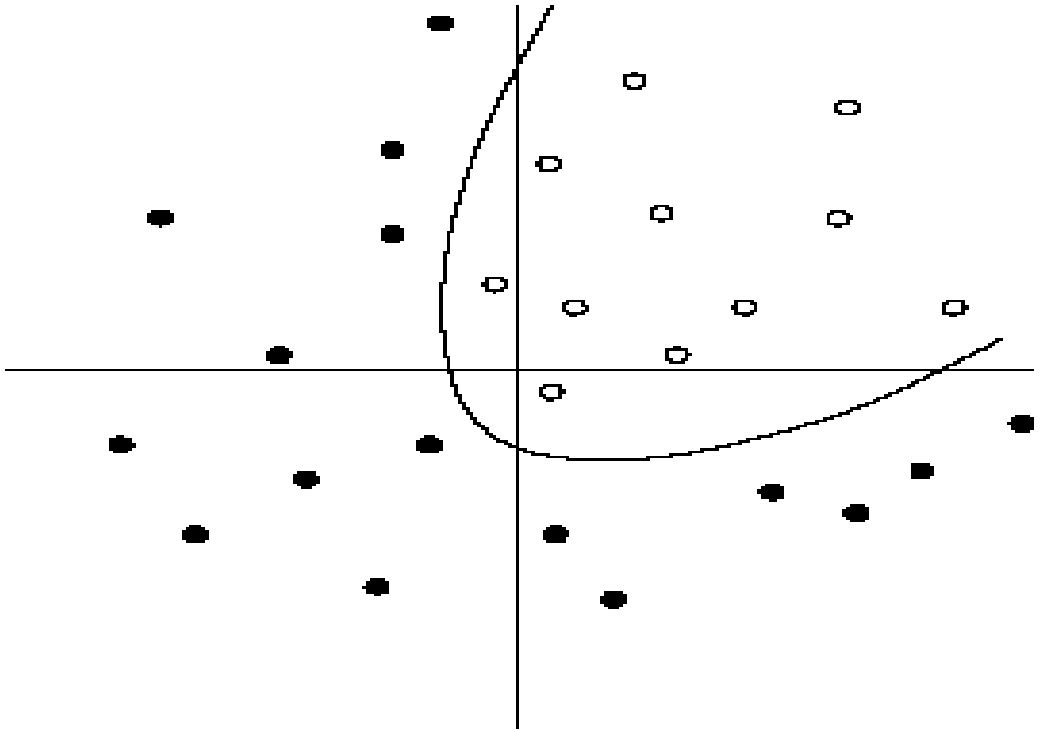
Percettrone a singolo strato per classificazione a più classi



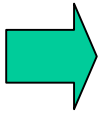
• Un neurone y_j per ogni classe j, $1 \leq j \leq m$ completamente connesso all' ingresso x.

• La classe di uscita viene computata tramite tecnica *WTA* (Winner Takes All)

Come classificare dati non linearmente separabili ?

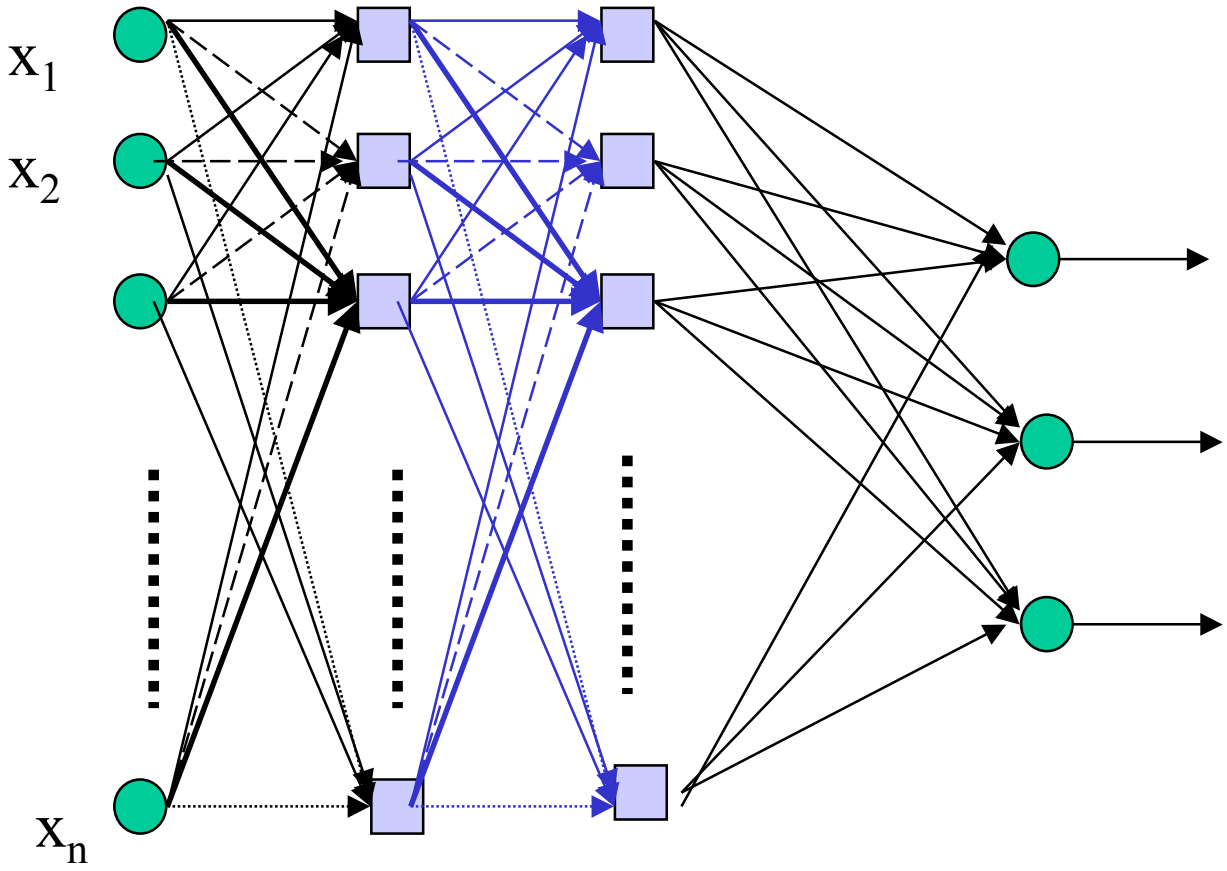


• E' impossibile separare le 2 classi con una retta (percettrone semplice)



• Si devono utilizzare percettroni a più strati …

Struttura di un percettrone multistrato (MLP)



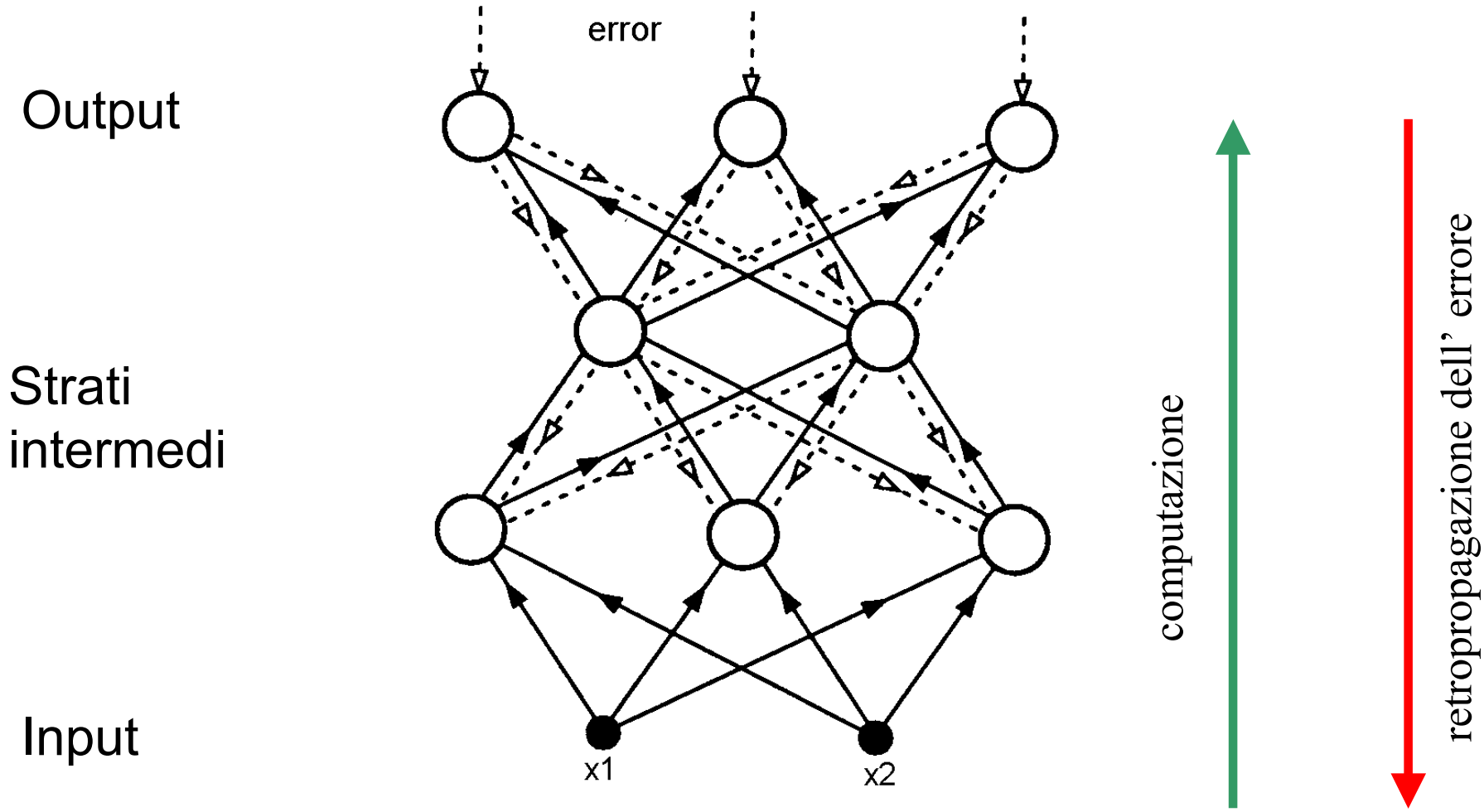
Come addestrare un MLP ?

Un algoritmo di discesa a gradiente non è direttamente applicabile:

- Ogni strato ha i suoi pesi che devono essere aggiornati
- Solo l' errore rispetto all' uscita è noto

Algortimo di backpropagation (retropropagazione)
(*Rumelhart* et al. 1986)

Visualizzazione dell' algoritmo di backpropagation in una rete neurale a 3 strati

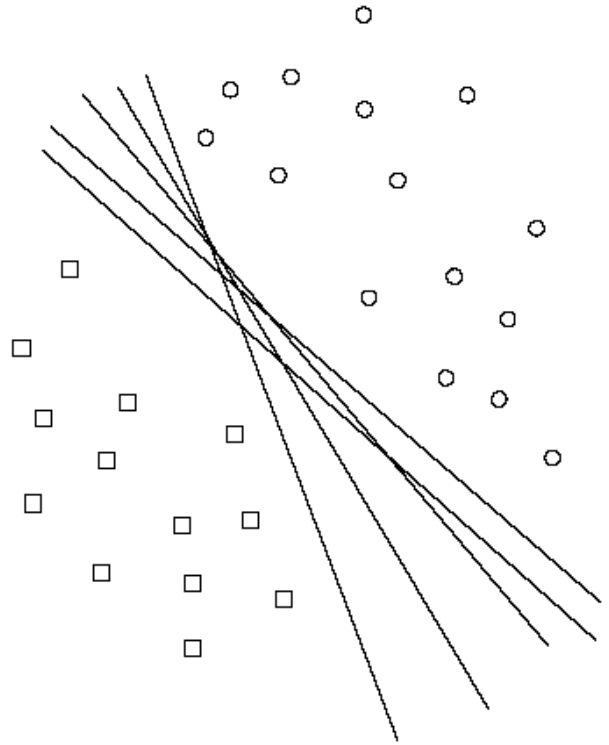


MLP: scelta del modello

- L’ apprendimento dipende dalle condizioni iniziali (valori casuali iniziali di pesi)
- Le capacità di generalizzazione dipendono:
  - dalla topologia
  - dal numero di neuroni degli strati intermedi
  - dalla *regolarizzazione* della rete
  - dalle condizioni di stop selezionate
  - dal coefficiente di apprendimento
  - dalla variante algoritmica utilizzata

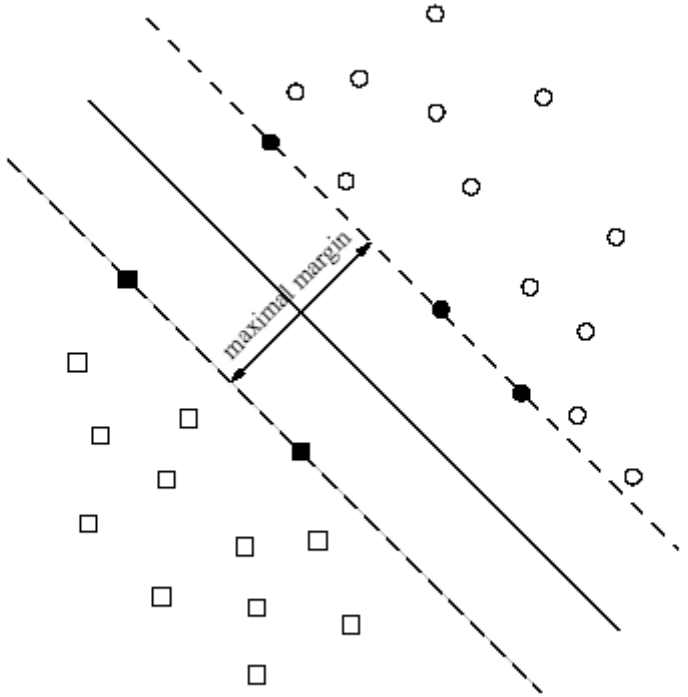
- Provare almeno:
  - diversi numeri di unità nascoste (neuroni degli strati intermedi)
  - diverse condizioni di stop
  - MLP regolarizzati (weight decay)

Support Vector Machine (SVM)



Come classifica una rete neurale
(non regolarizzata)

Soluzioni molteplici
(minimi locali multipli)



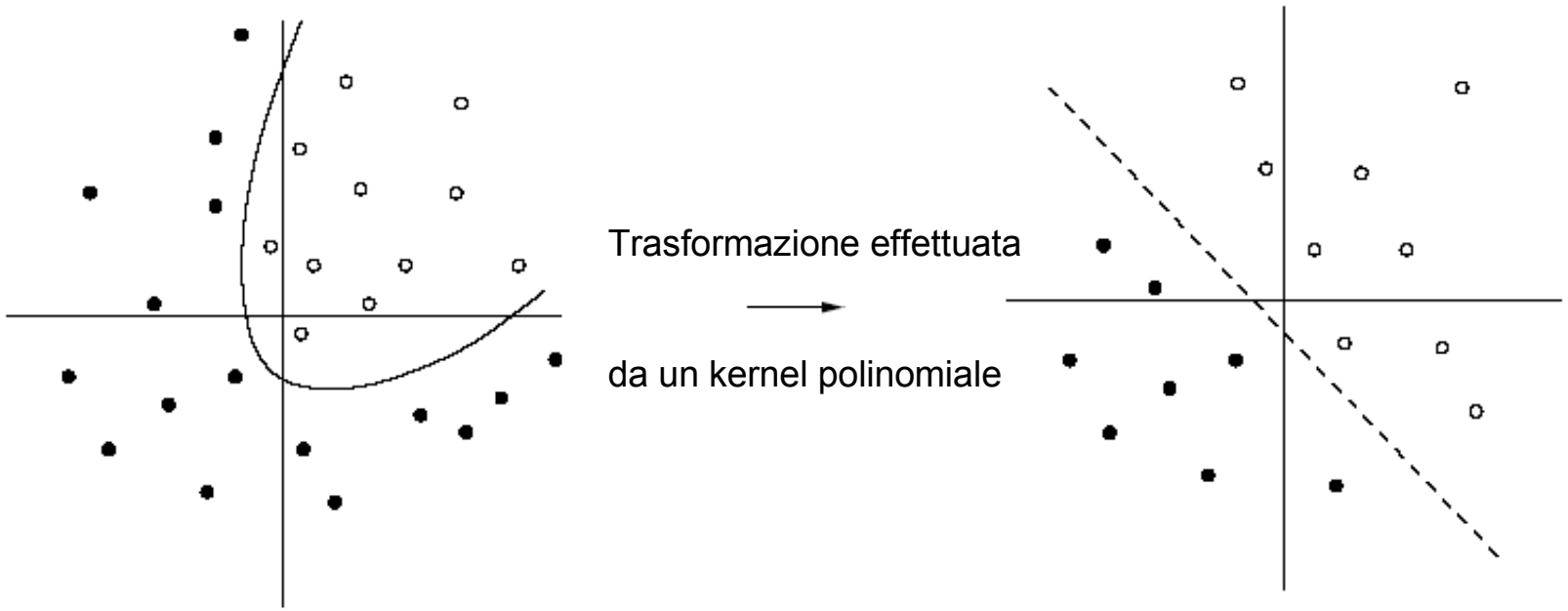
Come classifica una SVM lineare

Soluzione unica (minimo globale)

SVM per problemi non linearmente separabili

- Tramite funzioni kernel (es: funzioni gaussiane e polinomiali) i campioni da classificare sono proiettati in uno spazio iperdimensionale.

- In tale spazio è più probabile che un classificatore lineare riesca a classificare correttamente i campioni (*Teorema di Cover*)



Spazio di ingresso originale: i dati non sono linearmente separabili

La SVM calcola l' iperpiano di separazione ottimale nello spazio trasformato

Le SVM sono considerate lo "stato dell' arte" nella classificazione di dati di espressione genica

"General" motivations

•SVM are two-class classifiers theoretically founded on Vapnik's Statistical Learning Theory.

• They act as linear classifiers in a high dimensional feature space originated by a projection of the original input space.

• The resulting classifier is in general non linear in the input space.

• SVM achieves good generalization performances maximizing the margin between the classes.

• SVM learning algorithm has no local minima

"Specific" motivations

• Kernel are well-suited to working with high dimensional data.

• Small sample sizes require algorithms with good generalization capabilities.

• Automatic diagnosis of tumors requires high sensitivity and very effective classifiers.

• SVM can identify mis-labeled data (i.e. incorrect diagnosis).

• We could design specific kernel to incorporate "a priori" knowledge about the problem.

An example: SVM to classify cancerous and normal cells (lymphoma, Alizadeh et al. data)

We consider 3 standard SVM kernels:

- *Gaussian*
- *Polynomial*
- *Dot-product*

Varying:

- Values of the the kernel parameters
- The regularization factor C

Comparing them with:

- MLP
- LP

Varying:

- Number of hidden units
- Backpropagation parameters

Estimation of the generalization error through:

- 10-fold cross-validation
- leave-one-out

Results

Learning machine model	**Gen. error**	**St. dev.**	**Prec.**	**Sens.**
SVM-linear	1.04	3.16	98.63	100.0
SVM-poly	4.17	5.46	94.74	100.0
SVM-RBF	25.00	4.48	75.00	100.0
MLP	2.08	4.45	98.61	98.61
LP	9.38	10.24	95.65	91.66

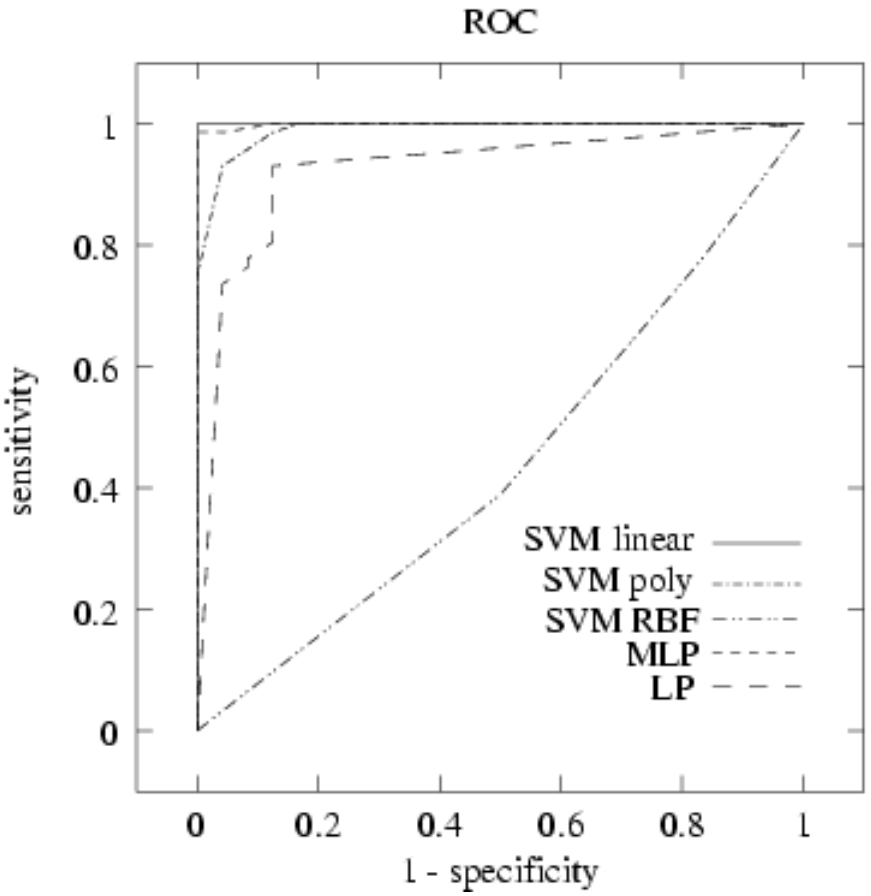
- 10-fold cross-validation ~ leave-one-out estimation of error
- SVM-linear achieves the best results.
- High sensitivity, no matter what type of kernel function is used.
- Radial basis SVM high misclassification rate and high estimated VC dimension

ROC analysis

• The ROC curve of the SVM-linear is ideal

• The polynomial SVM also achieves a reasonably good ROC curve

• The SVM-RBF show a diagonal ROC curve: the highest sensitivity is achieved only when it completely fails to correctly detect normal cells.



• The ROC curve of the MLP is also nearly optimal

• Linear perceptron shows a worse ROC curve, but with reasonable values lying on the highest and leftmost part of the ROC plane.

Summary of the results on the classification of malignant versus normal tissues

- Using hierarchical clustering 14,6% of the examples are misclassified (Alizadeh, 2000), against the 1.04% of the SVM, the 2.08% of the MLP and the 9.38% of the LP.
- Supervised methods exploit a priori biological knowledge (i.e. labeled data), while clustering methods use only gene expression data to group together different tissues, without any labeled data.
- Linear SVM achieve the best results, but also MLP and 2^{nd} degree polynomial show a relatively low generalization error.
- Linear SVM and MLP can be used to build classifiers with a high-sensitivity and a low rate of false positives.
- These results must be considered with caution because the size of the available data set is too small to infer general statements about the performances of the proposed learning machines.

SVM for gene expression data analysis: a bibliography

- M. Brown et al. *Knowledge-base analysis of microarray gene expression data by using support vector machines*. PNAS, 97(1):262--267, National Academy of Sciences Washington DC, 2000.
- T.S. Furey, N.Cristianini, N.Duffy, D.Bednarski, M.Schummer, and D.Haussler. *Support vector machine classification and validation of cancer tissue samples using microarray expression data*. Bioinformatics, 16(10):906--914, 2000.
- P. Pavlidis, J.Weston, J.Cai, and W.N. Grundy. *Gene functional classification from heterogenous data.* In Fifth International Conference on Computational Molecular Biology, ACM, Montreal, Canada, 2001.
- C. Yeang, S. Ramaswamy, P. Tamayo, S. Mukherjee, R. Rifkin, M Angelo, M. Reich, E. Lander, J. Mesirov, and T. Golub. *Molecular classification of multiple tumor types*. ISMB 2001, Proceedings of the 9th International Conference on Intelligent Systems for Molecular Biology, pages 316--322, Copenaghen, Denmark. Oxford University Press, 2001.
- S. Ramaswamy et al., *Multiclass cancer diagnosis using tumor gene expression signatures*, PNAS, 98(26), 15149--15154, 2001.
- I.Guyon, J.Weston, S.Barnhill, and V.Vapnik. *Gene Selection for Cancer Classification using Support Vector Machines*. Machine Learning, 46(1/3):389--422, 2002.
- J. Weston, F. Perez-Cruz, O. Bousquet, O. Chapelle, A. Elisseeff, and B. Scholkopf, *Feature selection and transduction for prediction of molecular bioactivity for drug design*, Bioinformatics, 1(1), 2002.
- G. Valentini, *Gene expression data analysis of human lymphoma using support vector machines and output coding ensembles*, Artificial Intelligence in Medicine, 26(3):283--306, 2002.
- G. Valentini, M. Muselli and F. Ruffino, *Bagged Ensembles of SVMs for Gene Expression Data Analysis*, The IEEE-INNS-ENNS International Joint Conference on Neural Networks, Portland, USA, 2003.

Terza parte:

Stima della qualità dei classificatori per l' analisi dei dati di espressione genica

Rischio atteso e rischio empirico

- L` apprendimento di una funzione non nota $f : R^d \rightarrow$ C avviene tramite un algoritmo L che genera un insieme di funzioni g che approssimano f utilizzando solo un training set $D = \{(\mathbf{x}_i, t_i)\}_{i=1}^n$ distribuito secondo un distribuzione di probabilità non nota $P(\mathbf{x},t)$:

$$g : \Re^d \times \Omega \rightarrow C$$

Ω rappresenta un insieme di parametri della learning machine (ad es., l' insieme dei pesi delle unità di calcolo di una rete neurale).

• Obiettivo dell' apprendimento non è minimizzare il rischio empirico $R_{emp}(\omega)$:

$$R_{emp}(\omega) = \frac{1}{n}\sum_{i=1}^{n} Loss\left(g(\mathbf{x}_i, \omega), t_i\right)$$

bensì il rischio atteso R(ω):

$$R(\omega) = \iint Loss\left(g(\mathbf{x}, \omega), t\right) p(\mathbf{x}, t)\, d\mathbf{x}\, dt$$

A parte le difficoltà matematiche della minimizzazione del funzionale R(ω), quasi sempre la funzione di densità di probabilità congiunta non è nota ...

Stima del rischio atteso

- Il rischio empirico non sempre converge al rischio atteso.
- La Teoria Statistica dell' Apprendimento di Vapnik ha mostrato che un limite superiore al rischio atteso può essere scomposto in due componenti:

$$R(\omega) \leq R_{emp}(\omega) + \Phi(h/m)$$

dove il primo termine dipende dal rischio empirico, mentre l' intervallo di confidenza Φ dipende principalmente dal rapporto fra la complessità h della learning machine e la cardinalità m del training set disponibile.

• Per valutare le capacità di generalizzazione delle learning machine è necessario stimare il rischio atteso e non semplicemente il rischio empirico.

• Il problema è: come stimare il rischio atteso ?

Due approcci principali alla stima del rischio atteso

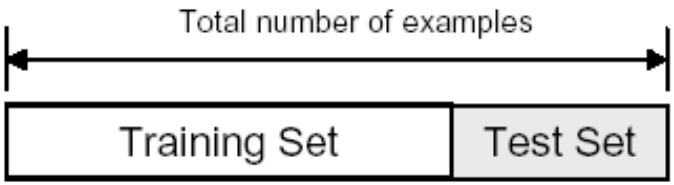
1. Stima teorica dei limiti superiori al rischio atteso (basati sull' errore empirico e sulla stima della complessità della learning machine)
2. Stima sperimentale (basata sul campionamento dei dati disponibili)

Metodi di stima sperimentale dell' errore di generalizzazione

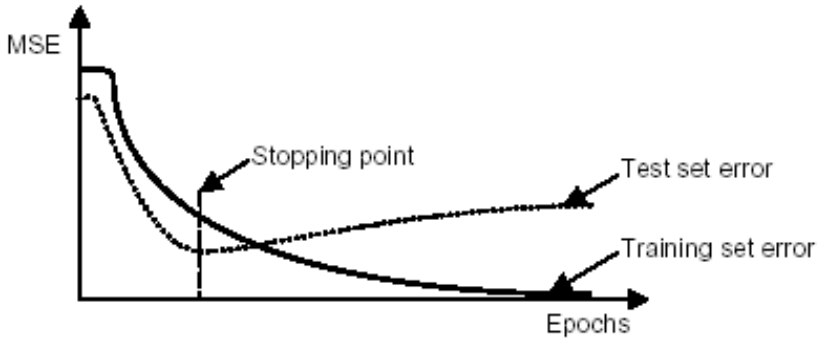
- Holdout
  - Suddivisione dei dati in *training* e *test* set (tipicamente 2/3 ed 1/3)
- Sottocampionamento casuale
  - Holdout ripetuto n volte
- Cross validation
  - Partizione dei dati in k sottoinsiemi disgiunti (fold)
  - k-fold: training con k-1 fold, test sul rimanente; il processo è ripetuto k volte utilizzando ognimvolta come test set un fold differente.
  - Leave-one-out: k = numero dei campioni disponibili
- Bootstrap
  - Campionamento con rimpiazzo
- Metodi out-of-bag
  - Training sui campioni estratti tramite bootstrap e testing sui rimanneti campioni non selezionati. Il proceso è ripetuto n volte.

Metodo di hold-out (1)

- **Split dataset into two groups**
  - Training set: used to train the classifier
  - Test set: used to estimate the error rate of the trained classifier



- **A typical application the holdout method is determining a stopping point for the back propagation error**

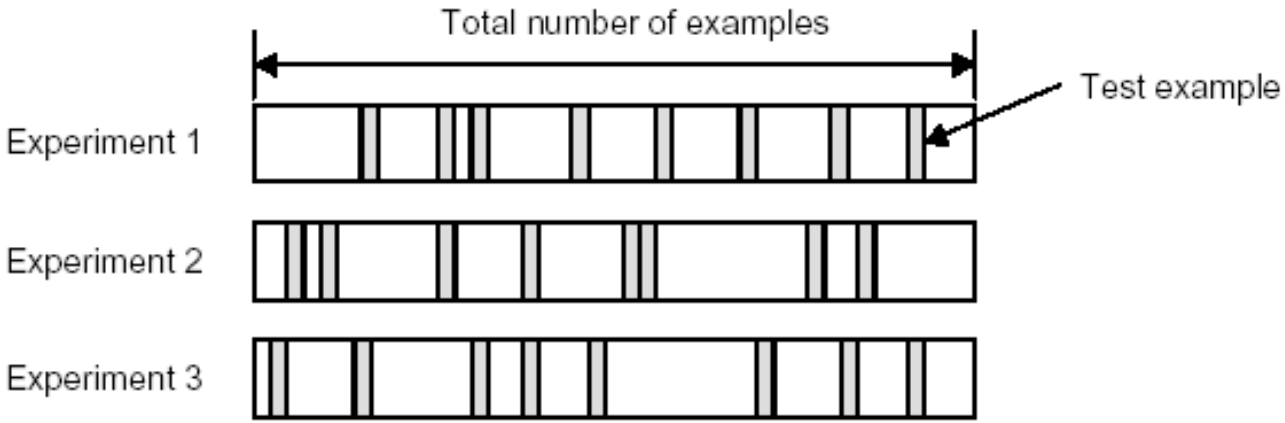


Metodo di hold-out (2)

- **The holdout method has two basic drawbacks**
  - In problems where we have a sparse dataset we may not be able to afford the "luxury" of setting aside a portion of the dataset for testing
  - Since it is a single train-and-test experiment, the holdout estimate of error rate will be misleading if we happen to get an "unfortunate" split
- **The limitations of the holdout can be overcome with a family of resampling methods at the expense of higher computational cost**
  - Cross Validation
    - Random Subsampling
    - K-Fold Cross-Validation
    - Leave-one-out Cross-Validation
  - Bootstrap

Campionamento casuale (holdout ripetuto)

- **Random Subsampling performs K data splits of the entire dataset**
  - Each data split randomly selects a (fixed) number of examples without replacement
  - For each data split we retrain the classifier from scratch with the training examples and then estimate E_i with the test examples

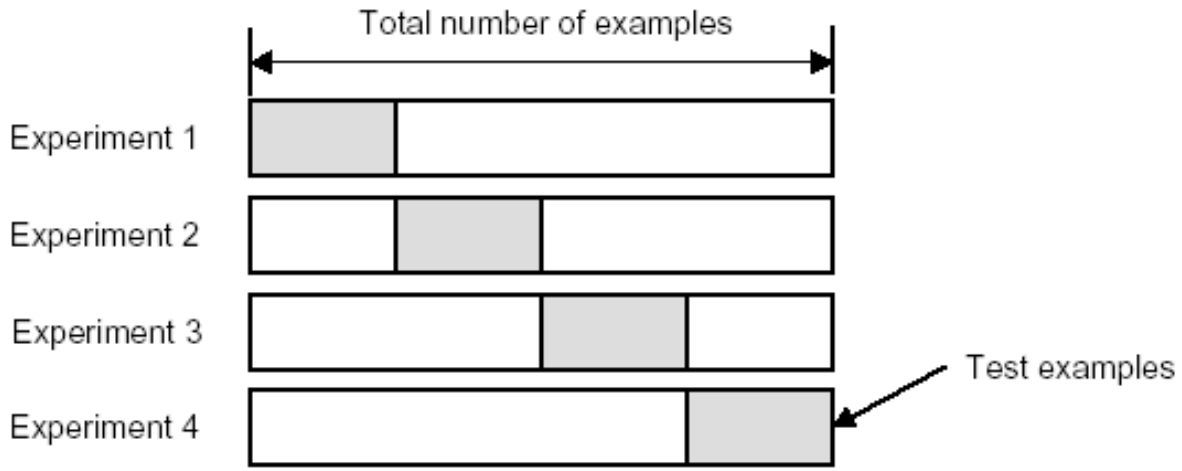


- **The true error estimate is obtained as the average of the separate estimates E_i**
  - This estimate is significantly better than the holdout estimate

$$E = \frac{1}{K}\sum_{i=1}^{K} E_i$$

K-fold cross validation

- **Create a K-fold partition of the the dataset**
  - For each of K experiments, use K-1 folds for training and a different fold for testing
  - This procedure is illustrated in the following diagram for K=4

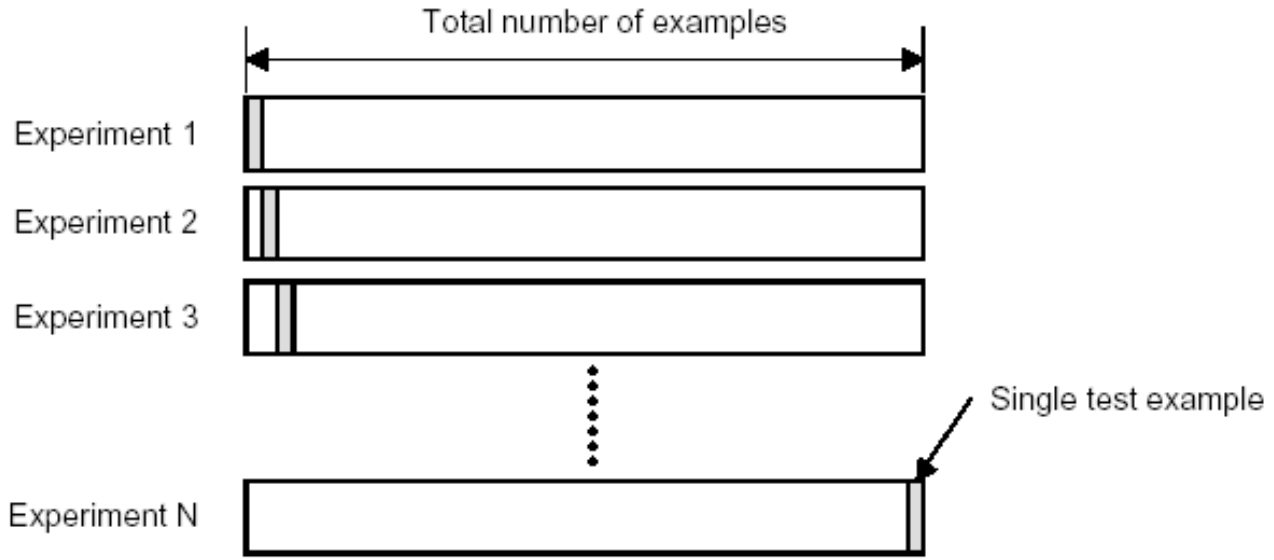


- **K-Fold Cross validation is similar to Random Subsampling**
  - The advantage of K-Fold Cross validation is that all the examples in the dataset are eventually used for both training and testing
- **As before, the true error is estimated as the average error rate on test examples**

$$E = \frac{1}{K}\sum_{i=1}^{K} E_i$$

Leave-one-out

- **Leave-one-out is the degenerate case of K-Fold Cross Validation, where K is chosen as the total number of examples**
  - For a dataset with N examples, perform N experiments
  - For each experiment use N-1 examples for training and the remaining example for testing



- **As usual, the true error is estimated as the average error rate on test examples**

$$E = \frac{1}{N}\sum_{i=1}^{N} E_i$$

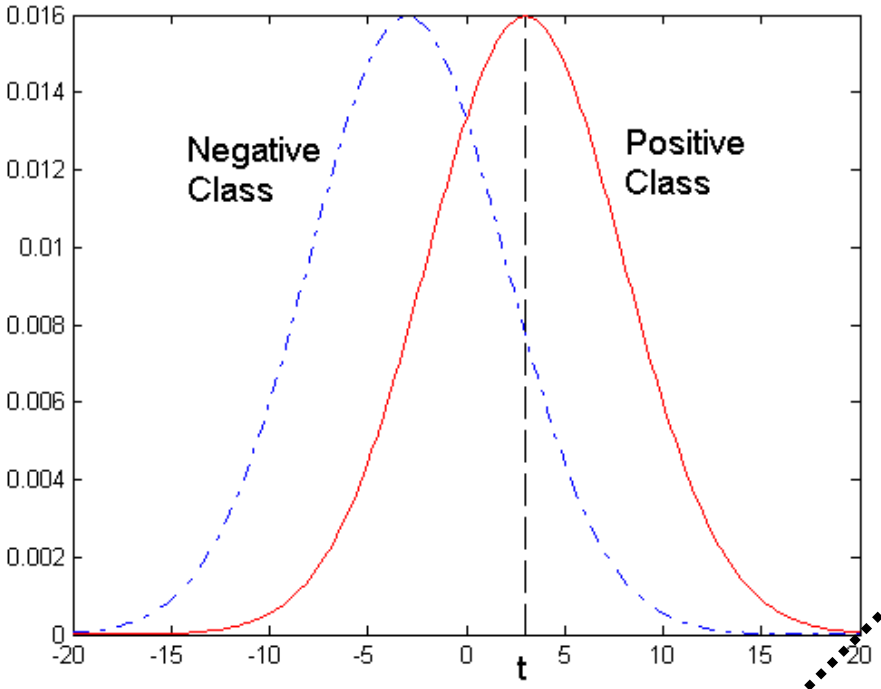
ROC (Receiver Operating Characteristic)

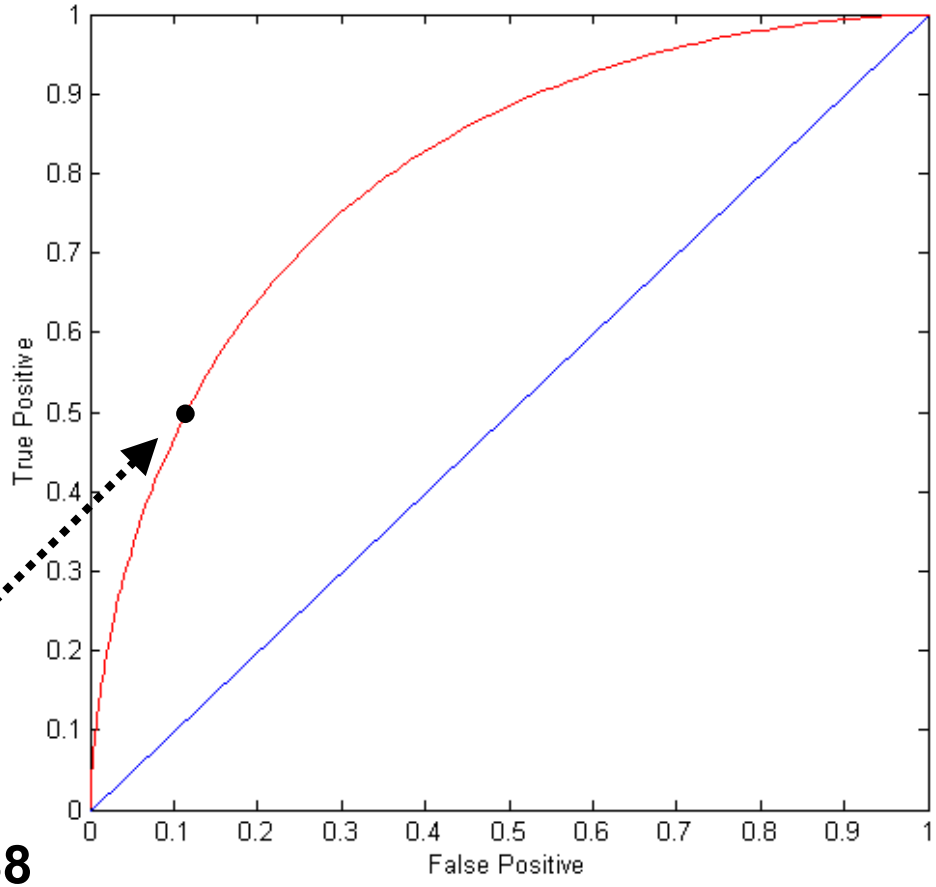
- Developed in 1950s for signal detection theory to analyze noisy signals
  - Characterize the trade-off between positive hits and false alarms
- ROC curve plots TP (on the y-axis) against FP (on the x-axis)
- Performance of each classifier represented as a point on the ROC curve
  - changing the threshold of algorithm, sample distribution or cost matrix changes the location of the point

ROC Curve

- 1-dimensional data set containing 2 classes (positive and negative)

- any points located at x > t is classified as positive





At threshold t:

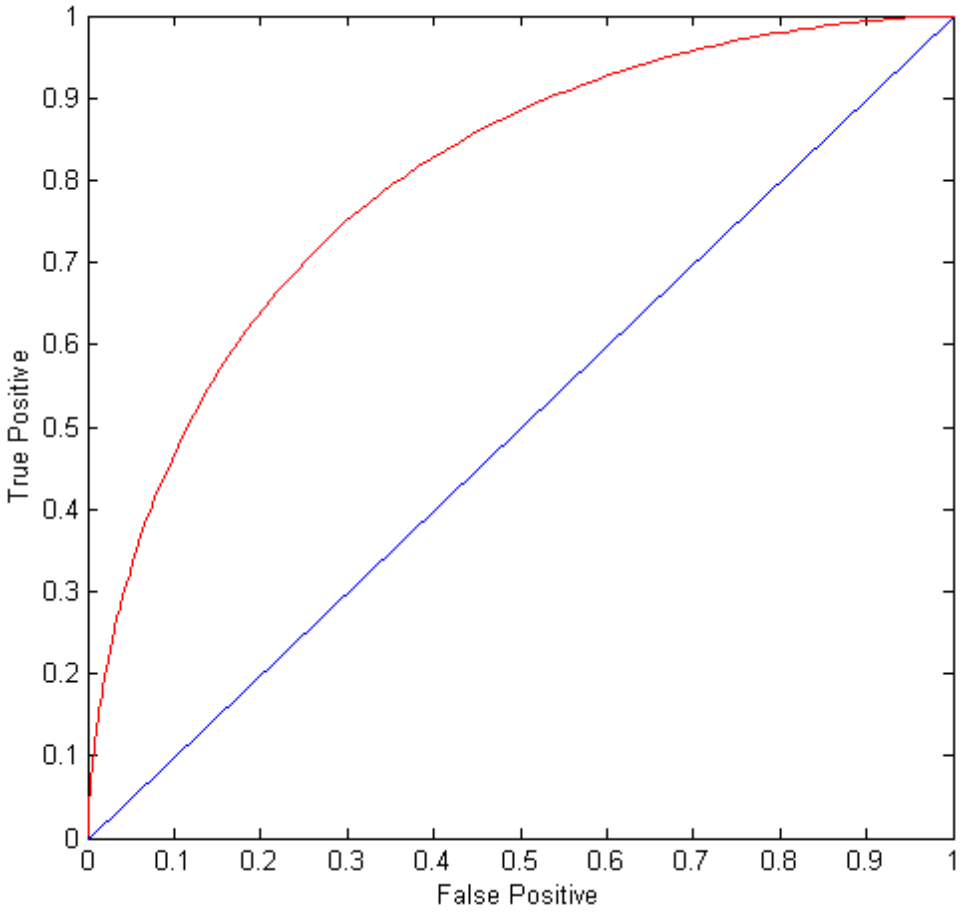
TP=0.5, FN=0.5, FP=0.12, FN=0.88

ROC Curve

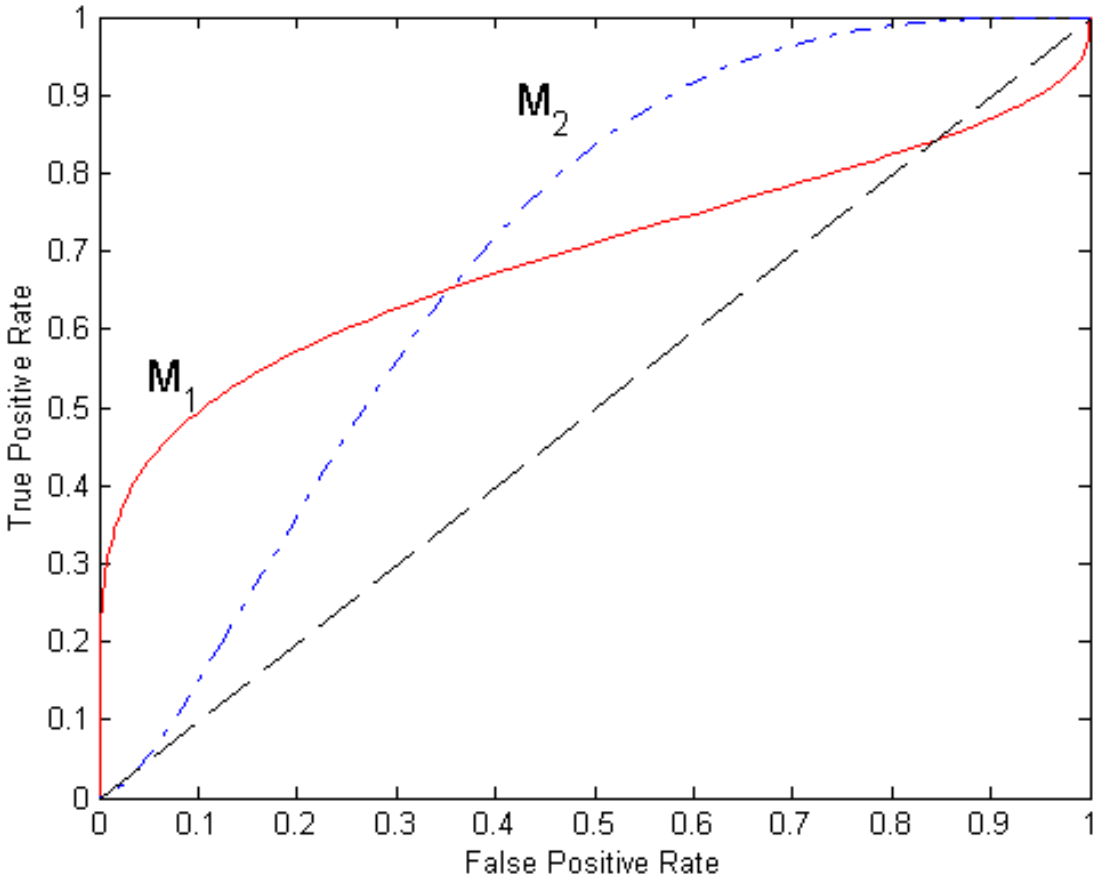
(TP,FP):

- (0,0): declare everything to be negative class
- (1,1): declare everything to be positive class
- (1,0): ideal

- Diagonal line:
  - Random guessing
  - Below diagonal line:
    - prediction is opposite of the true class



Using ROC for Model Comparison



- No model consistently outperform the other
  - M_1 is better for small FPR
  - M_2 is better for large FPR
- Area Under the ROC curve
  - Ideal:
    - Area = 1
  - Random guess:
    - Area = 0.5

Altre misure di stima della qualità dei classificatori per l' analisi dei dati di espressione genica (Guyon et al., 2002)

Accuratezza: $$Succ = \frac{1}{2n}\sum_{i=1}^{n} | h(x_i) + y_i |$$

Margini estremi: $$M_{ext} = \frac{\theta^+ - \theta^-}{\max(f(x_i)) - \min(f(x_i))}$$

$$\theta^+ = \min_{1 \le i \le n^+}(f(x_i)), \theta^- = \max_{n^+ + 1 \le i \le n}(f(x_i))$$

Margini mediani: $$M_{med} = \frac{\lambda^+ - \lambda^-}{\max(f(x_i)) - \min(f(x_i))}$$

dove λ^+ ed λ^- sono le mediane dei valori di f(x) per la classe positiva e negativa

Tasso di accettazione: $$Acc = \frac{\left|\{i \mid f(x_i) > \theta\}\right| + \left|\{i \mid f(x_i) < -\theta\}\right|}{n}$$

$$\theta = \max(|\theta^+|, |\theta^-|)$$

Predizione di tessuti tumorali con bagged ensemble di SVM e metodi di gene selection