Già considerato l'approccio dei classificatori Bayesiani.

↳ particolare sviluppo: Naive Bayes

Machine learning

Linear classification

Corso di Laurea Magistrale in Informatica

Università di Roma Tor Vergata

Prof. Giorgio Gambosi

a.a. 2021-2022

potremmo avere situazioni dove ogni elem. non appartiene ad una sola classe.

- value t to predict are from a discrete domain, where each value denotes a class
- most common case: disjoint classes, each input has to assigned to exactly one class
- input space is partitioned into decision regions
- in linear classification models decision boundaries are linear functions of input $\mathbf{x}$ ($D-1$-dimensional hyperplanes in the D-dimensional feature space)
- datasets such as classes correspond to regions which may be separated by linear decision boundaries are said linearly separable

Caso binario: divido lo spazio in 2 iperpiani. ($D=2$).

lo spazio e' geometricamente diviso in regioni, una per ogni classe.

Ad ogni classe e' associata una regione connessa ed anche convessa.

Per semplicità faremo rifermiento a classificazioni dove D=2.

Se abbiamo quindi la separazione linerare, l'ideale sarebbe un iperpiano dove da una parte c'è la classe 0 e da una parte la classe 1.

Non è detto che ci sia sempre un dataset linearmente separabile.

insieme di valori discreto.

- Regression: the target variable $\mathbf{t}$ is a vector of reals
- Classification: several ways to represent classes (target variable values)
- Binary classification: a single variable $t \in \{0, 1\}$, where $t = 0$ denotes class C_0 and $t = 1$ denotes class C_1
- $K > 2$ classes: "1 of K" coding. $\mathbf{t}$ is a vector of K bits, such that for each class C_j all bits are 0 except the j-th one (which is 1)

La cosa più semplice nella classificazione: avviamo le classi con dei nomi, gli associo delle codifiche intere. Sugli interi è però definita una relazione d'ordine, ma qui è impropria e quindi la codifica non piace e si usano altri metodi, ovvero la codifica "1 su K".

Codifica "1 su K": supponiamo di avere 2 classi, ogni elemento è rappresentato come un vettore di due valori binari: il primo valore è associato alla prima classe, il secondo alla seconda.

Per i classi: il vettore della classe i-esima sarà: <0, … 1, … ,0> dove l'1 è in posizione 1-esima.

Interessante per i classificatori probabilistici: il risultato sarà una coppia di valori di probabilità (sempre caso a due dimensioni). Con la nostra notazione: nel caso M, F avremo probabilità 1 di avere un M e 0 di avere F (vettore <1,0>). Il tipo di rappresentazione è la stessa, vale anche per il caso k.

Approaches to classification

2 approcci generali: geometrico e probabilistico.

Three general approaches to classification

1. find $f : \mathbf{X} \mapsto \{1, \ldots, K\}$ (discriminant function) which maps each input $\mathbf{x}$ to some class C_i, such that $i = f(\mathbf{x})$
2. discriminative approach: determine the conditional probabilities $p(C_j|\mathbf{x})$ (inference phase); use these distributions to assign an input to a class (decision phase)
3. generative approach: determine the class conditional distributions $p(\mathbf{x}|C_j)$, and the class prior probabilities $p(C_j)$; apply Bayes' formula to derive the class posterior probabilities $p(C_j|\mathbf{x})$; use these distributions to assign an input to a class

- Approccio geometrico: applico l'algebra lineare per trovare le funzioni che separino le classi.
- Approccio probabilistico: 2 approcci
  - discriminativo
  - generativo.

Approccio generativo: sono fondamentalmente approcci Bayesiani. Concettualmente, cerchiamo il modo migliore (parametrico) per rappresentare le varie classi, a quel punto avrò lì i coefficienti

Approccio discriminativo: non cerco di caratterizzare $i(i|i\ \ )$, da cui poi verrà $i(i\ \ |i)$ che è quello che userò.
Qui, parto da fatto che mi interessi $i(i\ \ |i)$, quindi assumo che questo abbia una certa forma con dei parametri. A questo punto cerco di apprendere i parametri, salto la prima fase.

Nel caso 3) cerchiamo i valori di w che permettono meglio di rappresentare ogni classe (max likelihood, MAP), una volta modellate al meglio le varie classi applichiamo Bayes. Non ci confrontiamo mai con il target, o meglio guardiamo al target partizionando il dataset ma finisce lì. In 2), il learning è più "classico": per ogni valore di w vediamo le predizioni che ci fornisce il modello e vediamo qual è il costo

2) e 3) sono i così detti metodi generativi, cambiano al variare della struttura delle x e C mentre il primo aprroccio ha delle tecniche abbastanza semplici.

Cerchiamo di discriminare al meglio i punti nelle varie classi.

- Approaches 1 and 2 are discriminative: they tackle the classification problem by deriving from the training set conditions (such as decision boundaries) that , when applied to a point, discriminate each class from the others
- The boundaries between regions are specified by discrimination functions

3° approccio: cerchiamo di rappresentare al meglio le varie classi.

Comb. lineare delle feature + termine noto → nuovo valore per fare la predizione

- In linear regression, a model predicts the target value; the prediction is made through a linear function $y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$ (linear basis functions could be applied)
- In classification, a model predicts probabilities of classes, that is values in $[0, 1]$; the prediction is made through a generalized linear model $y(\mathbf{x}) = f(\mathbf{w}^T\mathbf{x} + w_0)$, where f is a non linear activation function with codomain $[0, 1]$
- boundaries correspond to solution of $y(\mathbf{x}) = c$ for some constant c; this results into $w^T\mathbf{x} + w_0 = f^{-1}(c)$, that is a linear boundary. The inverse function f^{-1} is said link function.

per la classificazione, anche con solo valori 0,1, combinando linearmente non è detto che ottenga 0-1 come valori del codominio. Applico la funzione di attivazione (che non è lineare).

Per ogni punto dello spazio, applico $y(x) = f(w^T x + w_0)$, per mandare i valori nelle mie classi avrò una superficie di separazione. Come è fatta:

$w^T x + w_0 = f^{-1}(c)$, ma è costante $\Rightarrow$ le superfici di separazione sono lineari.

Generative approaches

- Approach 3 is generative: it works by defining, from the training set, a model of items for each class
- The model is a probability distribution (of features conditioned by the class) and could be used for random generation of new items in the class
- By comparing an item to all models, it is possible to verify the one that best fits

Linear discriminant functions in binary classification

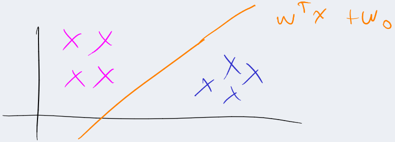
→ confine fra le classi

- Decision boundary: $D-1$-dimensional hyperplane of all points s.t. $y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0 = 0$
- Given $\mathbf{x}_1, \mathbf{x}_2$ on the hyperplane, $y(\mathbf{x}_1) = y(\mathbf{x}_2) = 0$. Hence,

$$\mathbf{w}^T\mathbf{x}_1 + w_0 - \mathbf{w}^T\mathbf{x}_2 - w_0 = \mathbf{w}^T(\mathbf{x}_1 - \mathbf{x}_2) = 0$$

  that is, vectors $\mathbf{x}_1 - \mathbf{x}_2$ and $\mathbf{w}$ are orthogonal → il vettore dei coefficienti è sempre ⊥ all'iperpiano.
- For any $\mathbf{x}$, the dot product $\mathbf{w} \cdot \mathbf{x} = \mathbf{w}^T\mathbf{x}$ is the length of the projection of $\mathbf{x}$ in the direction of $\mathbf{w}$ (orthogonal to the hyperplane $\mathbf{w}^T\mathbf{x} + w_0 = 0$), in multiples of $\|\mathbf{w}\|_2$
- By normalizing wrt to $\|\mathbf{w}\|_2 = \sqrt{\sum_i w_i^2}$, we get the length of the projection of $\mathbf{x}$ in the direction orthogonal to the hyperplane, assuming $\|\mathbf{w}\|_2 = 1$

Spazio 2D:

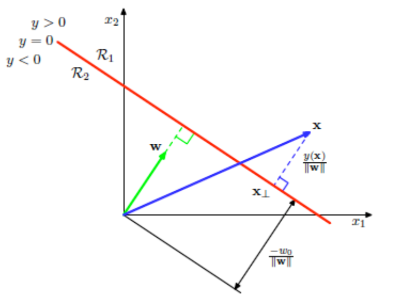


e qualunque iperpiano

- For any $\mathbf{x}$, $y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$ returns the distance (in multiples of $\|\mathbf{w}\|$) of $\mathbf{x}$ from the hyperplane
- The sign of the returned value discriminates in which of the regions separated by the hyperplane the point lies

ottengo:

- 0 se il punto e' sull' iperpiano
- la distanza fra il punto e l' iperpiano.



Linear discriminant functions in multiclass classification

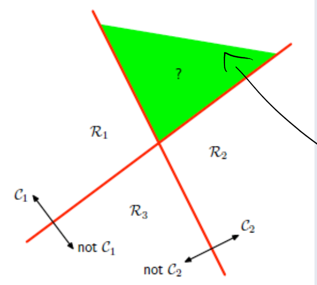
First approach

- Define $K-1$ discrimination functions
- Function f_i $(1 \leq i \leq K-1)$ discriminates points belonging to class C_i from points belonging to all other classes: if $f_i(\mathbf{x}) > 0$ then $\mathbf{x} \in C_i$, otherwise $\mathbf{x} \notin C_i$
- The green region belongs to both R_1 and R_2

Supponiamo di avere 2 classi.

$\neg C_i$

C_i



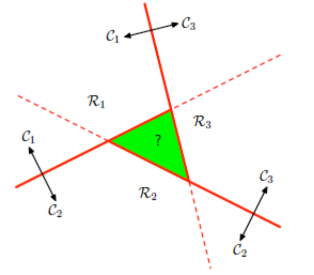
Ma la singola retta (iperpiano) non funziona.

Sia di C_1 che di C_2

Second approach

- Define $K(K-1)/2$ discrimination functions, one for each pair of classes
- Function f_{ij} $(1 \leq i < j \leq K)$ discriminates points which might belong to C_i from points which might belong to C_j
- Item $\mathbf{x}$ is classified on a majority basis
- The green region is unassigned

C'e' ancora lo stesso problema.



Dove K sono le classi.

Third approach

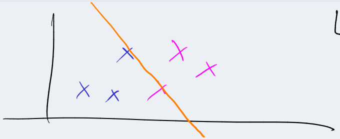
- Define K linear functions
$$y_i(\mathbf{x}) = \mathbf{w}_i^T\mathbf{x} + w_{i0} \qquad\qquad 1 \leq i \leq K$$
Item $\mathbf{x}$ is assigned to class C_k iff $y_k(\mathbf{x}) > y_j(\mathbf{x})$ for all $j \neq k$: that is,
$$k = \underset{j}{\operatorname{argmax}}\, y_j(\mathbf{x})$$
Assegno alla classe per cui la y è più alta.

- Decision boundary between C_i and C_j: all points $\mathbf{x}$ s.t. $y_i(\mathbf{x}) = y_j(\mathbf{x})$, a $D-1$-dimensional hyperplane
$$(\mathbf{w}_i - \mathbf{w}_j)^T\mathbf{x} + (w_{i0} - w_{j0}) = 0$$

$w_i^T x + w_{i,0} = w_j^T x + w_{j,0}$

↳ eq. di un iperpiano.

Domanda: come sono le regioni? Lineari



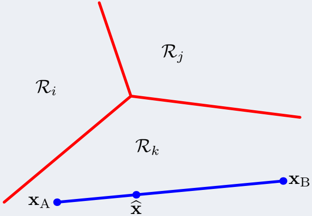
Nelle note, si può SALTARE

The resulting decision regions are connected and convex

- Given $\mathbf{x}_A, \mathbf{x}_B \in \mathscr{R}_k$ then $y_k(\mathbf{x}_A) > y_j(\mathbf{x}_A)$ and $y_k(\mathbf{x}_B) > y_j(\mathbf{x}_B)$, for all $j \neq k$
- Let $\hat{\mathbf{x}} = \lambda \mathbf{x}_A + (1-\lambda)\mathbf{x}_B$, $0 \leq \lambda \leq 1$
- Since y_i is linear for all i, $y_i(\hat{\mathbf{x}}) = \lambda y_i(\mathbf{x}_a) + (1-\lambda) y_i(\mathbf{x}_B)$
- Then, $y_k(\hat{\mathbf{x}}) > y_j(\hat{\mathbf{x}})$ for all $j \neq k$; that is, $\hat{\mathbf{x}} \in \mathscr{R}_k$

Convesse: due qualunque punti nella regione, considera la retta che li collega => tutti i punti sulla retta $\in$ alla regione.

Connesse: mai due regioni assegnate alla stessa classe "staccate" fra loro. Il cammino passa dentro.



- The definition can be extended to include terms relative to products of pairs of feature values (Quadratic discriminant functions)

$$y(\mathbf{x}) = w_0 + \sum_{i=1}^{D} w_i x_i + \sum_{i=1}^{D} \sum_{j=1}^{i} w_{ij} x_i x_j$$

  $\frac{d(d+1)}{2}$ additional parameters wrt the $d+1$ original ones: decision boundaries can be more complex

- In general, generalized discriminant functions through set of functions $\phi_i, \dots, \phi_m$

$$y(\mathbf{x}) = w_0 + \sum_{i=1}^{M} w_i \phi_i(\mathbf{x})$$

→ data una feature, devo prevedere K variabili.

- Assume classification with K classes
- Classes are represented through a 1-of-K coding scheme: set of variables $z_1, \dots, z_K$, class C_i coded by values $z_i = 1$, $z_k = 0$ for $k \neq i$
- K discriminant functions y_i are derived as linear regression functions with variables z_i as targets
- To each variable z_i a discriminant function $y_i(\mathbf{x}) = \mathbf{w}_i^T \mathbf{x} + w_{i0}$ is associated: $\mathbf{x}$ is assigned to the class C_k s.t.

$$k = \underset{i}{\operatorname{argmax}}\, y_i(\mathbf{x})$$

La parte non lineare: il valore più alto è 1, gli altri sono 0,

- Then, $z_k(\mathbf{x}) = 1$ and $z_j(\mathbf{x}) = 0$ ($j \neq k$) if $k = \underset{i}{\operatorname{argmax}}\, y_i(\mathbf{x})$
- Group all parameters together as

Decido di prevedere ogni "1-of-K" con K regressioni diverse ⇒ ottengo valori reali.

$$\mathbf{y}(\mathbf{x}) = \mathbf{W}^T \overline{\mathbf{x}} = \begin{pmatrix} w_{10} & w_{11} & \cdots & w_{1D} \\ w_{20} & w_{21} & \cdots & w_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ w_{K0} & w_{K1} & \cdots & w_{KD} \end{pmatrix} \begin{pmatrix} 1 \\ x_1 \\ \vdots \\ x_D \end{pmatrix} = \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_K \end{pmatrix} \quad \begin{matrix} y_1(x) \\ y_2(x) \\ \vdots \\ y_K(x) \end{matrix}$$

Linear discriminant functions and regression

- In general, a regression function provides an estimation of the target given the input $E[t|\mathbf{x}]$
- $y_i(\mathbf{x})$ can be seen as an estimate of the conditional expectation $E[z_i|\mathbf{x}]$ of binary variable z_i given $\mathbf{x}$
- If we assume z_i is distributed according to a Bernoulli distribution, the expectation corresponds to the posterior probability

$$\begin{aligned} y_i(\mathbf{x}) &\simeq E[z_i|\mathbf{x}] \\ &= P(z_i = 1|\mathbf{x}) \cdot 1 + P(z_i = 0|\mathbf{x}) \cdot 0 \\ &= P(z_i = 1|\mathbf{x}) \\ &= P(C_i|\mathbf{x}) \end{aligned}$$

- However, $y_i(\mathbf{x})$ is not a probability itself (we may not assume it takes value only in the interval $[0, 1]$)

- Given a training set $\mathbf{X}, \mathbf{t}$, a regression function can be derived by least squares
- An item in the training set is a pair $(\mathbf{x}_i, \mathbf{t}_i)$, $\mathbf{x}_i \in \mathbb{R}^D$ e $\mathbf{t}_i \in \{0, 1\}^K$
- $\overline{\mathbf{X}} \in \mathbb{R}^{n\times(D+1)}$ is the matrix of feature values for all items in the training set

$$\overline{\mathbf{X}} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1D} \\ 1 & x_{21} & \cdots & x_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{nD} \end{pmatrix}$$

- Then, for matrix $\overline{\mathbf{X}}\mathbf{W}$, of size $n \times K$, we have

$$(\overline{\mathbf{X}}\mathbf{W})_{ij} = w_{j0} + \sum_{k=1}^{D} x_{ik} w_{jk} = y_j(\mathbf{x}_i)$$

which is the estimate of $p(C_j|\mathbf{x}_i)$

- $y_j(\mathbf{x}_i)$ is compared to item $\mathbf{T}_{ij}$ in the matrix $\mathbf{T}$, of size $n \times K$, of target values, where row i is the 1-of-K coding of the class of item $\mathbf{x}_i$

$$(\overline{\mathbf{X}}\mathbf{W} - \mathbf{T})_{ij} = y_j(\mathbf{x}_i) - t_{ij} = \sum_{k=1}^{D} x_{ik} w_{jk} + w_{j0} - t_{ij}$$

Target: x_1 $\vdots$ x_n

codifica 1 su K dell'elemento.

- Let us consider the diagonal items of $(\overline{\mathbf{X}}\mathbf{W} - \mathbf{T})^T (\overline{\mathbf{X}}\mathbf{W} - \mathbf{T})$. Then,

$$((\overline{\mathbf{X}\mathbf{W}} - \mathbf{T})^T (\overline{\mathbf{X}\mathbf{W}} - \mathbf{T}))_{jj} = \sum_{i=1}^{n} (y_j(\mathbf{x}_i) - t_{ij})^2$$

That is,

$$((\overline{\mathbf{X}}\mathbf{W} - \mathbf{T})^T (\overline{\mathbf{X}}\mathbf{W} - \mathbf{T}))_{jj} = \sum_{\mathbf{x}_i \in C_j} (y_j(\mathbf{x}_i) - 1)^2 + \sum_{\mathbf{x}_i \notin C_j} y_j(\mathbf{x}_i)^2$$

- Summing all elements on the diagonal of $(\overline{\mathbf{X}}\mathbf{W}-\mathbf{T})^T(\overline{\mathbf{X}}\mathbf{W}-\mathbf{T})$ provides the overall sum, on all items in the training set, of the squared differences between observed values and values computed by the model, with parameters $\mathbf{W}$, that is

$$\sum_{j=1}^{K}\sum_{i=1}^{n}(y_j(\mathbf{x}_i)-t_{ij})^2$$

- This corresponds to the trace of $(\overline{\mathbf{X}}\mathbf{W}-\mathbf{T})^T(\overline{\mathbf{X}}\mathbf{W}-\mathbf{T})$. Hence, we have to minimize:

$$E(\mathbf{W})=\frac{1}{2}\mathrm{tr}\Big((\overline{\mathbf{X}}\mathbf{W}-\mathbf{T})^T(\overline{\mathbf{X}}\mathbf{W}-\mathbf{T})\Big)$$

- Standard approach, solve

$$\frac{\partial E(\mathbf{W})}{\partial \mathbf{W}}=\mathbf{0}$$

- It is possible to show that

$$\frac{\partial E(\mathbf{W})}{\partial \mathbf{W}} = \overline{\mathbf{X}}^T \overline{\mathbf{X}} \mathbf{W} - \overline{\mathbf{X}}^T \mathbf{T}$$

- From $\overline{\mathbf{X}}^T \overline{\mathbf{X}} \mathbf{W} - \overline{\mathbf{X}}^T \mathbf{T} = \mathbf{0}$ it results

$$\mathbf{W} = (\overline{\mathbf{X}}^T \overline{\mathbf{X}})^{-1} \overline{\mathbf{X}}^T \mathbf{T}$$

- and the set of discriminant functions

$$\mathbf{y}(\mathbf{x}) = \mathbf{W}^T \overline{\mathbf{x}} = \mathbf{T}^T \overline{\mathbf{X}} (\overline{\mathbf{X}}^T \overline{\mathbf{X}})^{-1} \overline{\mathbf{x}}$$

Effettua la predizione combinando linearmente i valori delle feature a partire dai dati.

Metodo di apprendimento non supervisionato (??), per classificatore lineare.

- The idea of *Linear Discriminant Analysis* (*LDA*) is to find a linear projection of the training set into a suitable subspace where classes are as linearly separated as possible
- A common approach is provided by Fisher linear discriminant, where all items in the training set (points in a D-dimensional space) are projected to one dimension, by means of a linear transformation of the type

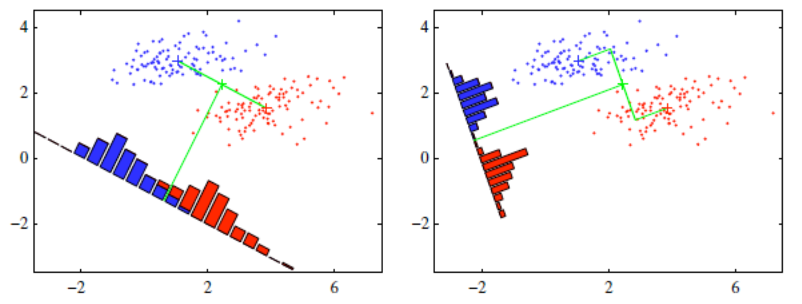
$$y = \mathbf{w} \cdot \mathbf{x} = \mathbf{w}^T \mathbf{x}$$

where $\mathbf{w}$ is the D-dimensional vector corresponding to the direction of projection (in the following, we will consider the one with unit norm).

Applicabile se il dataset è supervisionato (feature + target). Supponiamo classificazione binaria: come riduciamo la dimensionalità? 2D → retta così che tutti i punti di una classe siano da una parte e gli altri da un'altra.

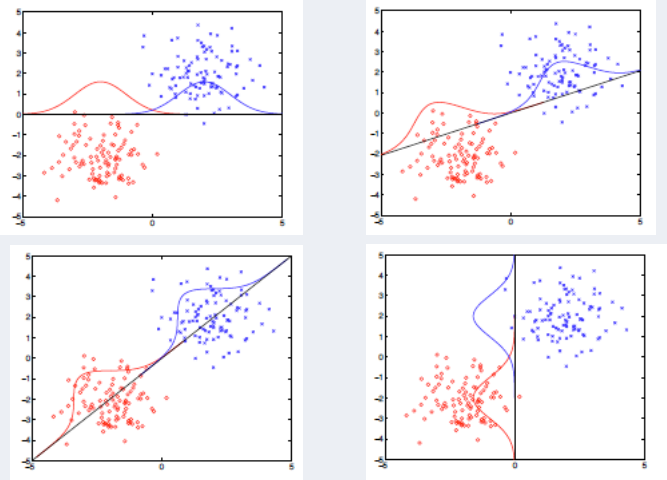
LDA

If $K = 2$, given a threshold $\tilde{y}$, item $\mathbf{x}$ is assigned to C_1 iff its projection $y = \mathbf{w}^T\mathbf{x}$ is such that $y > \tilde{y}$; otherwise, $\mathbf{x}$ is assigned to C_2.



Vorremmo almeno minimizzare la zona in cui rossi e blu si intrecciano (zona nel mezzo).
Iperpiano in questo caso è l'ascissa (1 punto) che separi gli insiemi.

Different line directions, that is different parameters $\mathbf{w}$, may induce quite different separability properties.



Let n_1 be the number of items in the training set belonging to class C_1 and n_2 the number of items in class C_2. The mean points of both classes are

$$\mathbf{m}_1 = \frac{1}{n_1} \sum_{\mathbf{x} \in C_1} \mathbf{x} \qquad\qquad \mathbf{m}_2 = \frac{1}{n_2} \sum_{\mathbf{x} \in C_2} \mathbf{x}$$

A simple measure of the separation of classes, when the training set is projected onto a line, is the difference between the projections of their mean points

$$m_2 - m_1 = \mathbf{w}^T(\mathbf{m}_2 - \mathbf{m}_1)$$

→ retta in cui le proiezioni dei punti medi sono il più distanti possibili.

where $m_i = \mathbf{w}^T \mathbf{m}_i$ is the projection of $\mathbf{m}_i$ onto the line.

- We wish to find a line direction $\mathbf{w}$ such that $m_2 - m_1$ is maximum
- $\mathbf{w}^T(\mathbf{m}_2 - \mathbf{m}_1)$ can be made arbitrarily large by multiplying $\mathbf{w}$ by a suitable constant, at the same time maintaining the direction unchanged. To avoid this drawback, we consider unit vectors, introducing the constraint $\|\mathbf{w}\|_2 = \mathbf{w}^T\mathbf{w} = 1$
- This results into the constrained optimization problem

$$\max_{\mathbf{w}} \mathbf{w}^T(\mathbf{m}_2 - \mathbf{m}_1)$$

$$\text{where } \mathbf{w}^T\mathbf{w} = 1$$

Considero vettori unitari (norma = 1)

Se consideriamo w molto grande ⇒ cresce la distanza fissati m_2 ed m_1, e' mal posto (la retta e' sempre la stessa).

- This can be transformed into an equivalent unconstrained optimization problem by means of lagrangian multipliers

$$\max_{\mathbf{w},\lambda} \mathbf{w}^T(\mathbf{m}_2 - \mathbf{m}_1) + \lambda(1 - \mathbf{w}^T\mathbf{w})$$

Cambio la f. obiettivo da massimizzare: il vincolo diviene parte della funzione obiettivo moltiplicandolo per il coefficiente λ, che e' il moltiplicatore lagrangiano.

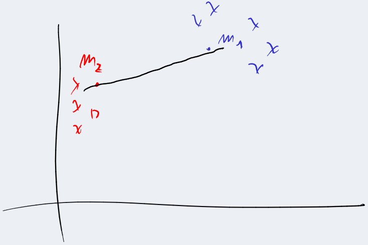
Deriving w in the binary case

Setting the gradient of the function wrt $\mathbf{w}$ to $\mathbf{0}$

$$\frac{\partial}{\partial \mathbf{w}}(\mathbf{w}^T(\mathbf{m}_2 - \mathbf{m}_1) + \lambda(1 - \mathbf{w}^T\mathbf{w})) = \mathbf{m}_2 - \mathbf{m}_1 + 2\lambda\mathbf{w} = \mathbf{0}$$

results into

$$\mathbf{w} = \frac{\mathbf{m}_2 - \mathbf{m}_1}{2\lambda}$$



Deriving w in the binary case

Setting the derivative wrt λ to 0

$$\frac{\partial}{\partial \lambda}(\mathbf{w}^T(\mathbf{m}_2 - \mathbf{m}_1) + \lambda(1 - \mathbf{w}^T\mathbf{w})) = 1 - \mathbf{w}^T\mathbf{w} = 0$$

results into

$$1 - \mathbf{w}^T\mathbf{w} = 1 - \frac{(\mathbf{m}_2 - \mathbf{m}_1)^T(\mathbf{m}_2 - \mathbf{m}_1)}{4\lambda^2} = 0$$

that is

$$\lambda = \frac{\sqrt{(\mathbf{m}_2 - \mathbf{m}_1)^T(\mathbf{m}_2 - \mathbf{m}_1)}}{2} = \frac{\|\mathbf{m}_2 - \mathbf{m}_1\|_2}{2}$$

Combining with the result for the gradient, we get

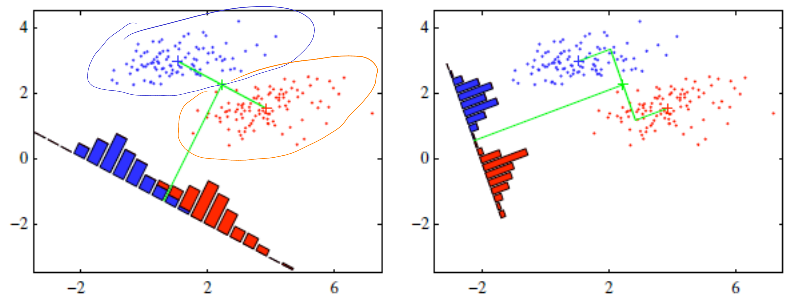
$$\boxed{\mathbf{w} = \frac{\mathbf{m}_2 - \mathbf{m}_1}{\|\mathbf{m}_2 - \mathbf{m}_1\|_2}}$$

La direzione ortogonale a quella che mi serve e' quella che congiunge m_1 ad m_2

The best direction $\mathbf{w}$ of the line, wrt the measure considered, is the one from $\mathbf{m}_1$ to $\mathbf{m}_2$.

However, this may result in a poor separation of classes.

Le due classi sono "allungate": effetto è che conviene mettersi in una direzione in base all'allungamento.



← l'intervallo è più stretto: è quello che si chiama dispersione delle classi.

Projections of classes are dispersed (high variance) along the direction of $\mathbf{m}_1 - \mathbf{m}_2$. This may result in a large overlap.

Tengo conto anche della varianza, distribuzione intorno alla media.

Deriving w in the binary case: refinement

- Choose directions s.t. classes projections show as little dispersion as possible
- Possible in the case that the amount of class dispersion changes wrt different directions, that is if the distribution of points in the class is elongated
- We wish then to maximize a function which:
  - is growing wrt the separation between the projected classes (for example, their mean points)
  - is decreasing wrt the dispersion of the projections of points of each class

Vorremmo una proiezione dove:
- le due classi sono molto separate (lontananza fra le medie)
- poca dispersione intorno alla media.

Deriving w in the binary case: refinement

- The within-class variance of the projection of class C_i ($i = 1, 2$) is defined as

$$s_i^2 = \sum_{\mathbf{x} \in C_i} (\mathbf{w}^T \mathbf{x} - m_i)^2$$

  The total within-class variance is defined as $s_1^2 + s_2^2$

- Given a direction $\mathbf{w}$, the Fisher criterion is the ratio between the (squared) class separation and the overall within-class variance, along that direction

$$J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}$$

← distanza elevata fra le medie

← così ho varianza piccola

- Indeed, $J(\mathbf{w})$ grows wrt class separation and decreases wrt within-class variance

```
Aspetto empirico: sto modellando perché l'obiettivo non è facilmente quantificabile, ma serve modellarlo
matematicamente con una scelta precisa.

Mi interessa mostrare che se modello l'obiettivo in modo diverso ottengo cose diverse: la funzione è come
quella di prima ma con denominatore e che ha l'andamento che ci piace. È il criterio di Fisher.
```

Let $\mathbf{S}_1, \mathbf{S}_2$ be the within-class covariance matrices, defined as

$$\mathbf{S}_i = \sum_{\mathbf{x} \in C_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^T$$

Then,

$$\begin{aligned}
s_i^2 &= \sum_{\mathbf{x} \in C_i} (\mathbf{w}^T \mathbf{x} - m_i)^2 = \sum_{\mathbf{x} \in C_i} (\mathbf{w}^T \mathbf{x} - \mathbf{w}^T \mathbf{m}_i)^2 \\
&= \sum_{\mathbf{x} \in C_i} (\mathbf{w}^T \mathbf{x} - \mathbf{w}^T \mathbf{m}_i)(\mathbf{w}^T \mathbf{x} - \mathbf{w}^T \mathbf{m}_i) \\
&= \sum_{\mathbf{x} \in C_i} (\mathbf{w}^T \mathbf{x} - \mathbf{w}^T \mathbf{m}_i)(\mathbf{x}^T \mathbf{w} - \mathbf{m}_i^T \mathbf{w}) \\
&= \sum_{\mathbf{x} \in C_i} \left(\mathbf{w}^T (\mathbf{x} - \mathbf{m}_i)\right)\left((\mathbf{x} - \mathbf{m}_i)^T \mathbf{w}\right) \\
&= \sum_{\mathbf{x} \in C_i} \mathbf{w}^T (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^T \mathbf{w} \\
&= \mathbf{w}^T \left(\sum_{\mathbf{x} \in C_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^T \right) \mathbf{w} = \mathbf{w}^T \mathbf{S}_i \mathbf{w}
\end{aligned}$$

→ nelle note

s_i^2 in funzione di dati e di $\mathbf{w}$

Deriving w in the binary case: refinement

dataset

Let also $\mathbf{S}_W = \mathbf{S}_1 + \mathbf{S}_2$ be the total within-class covariance matrix and

$$\mathbf{S}_B = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^T$$

be the between-class covariance matrix.

Then,

$$\begin{aligned}
J(\mathbf{w}) &= \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} = \cancel{\frac{(\mathbf{w}^T\mathbf{m}_2 - \mathbf{w}^T\mathbf{m}_1)^2}{\mathbf{w}^T\mathbf{S}_1\mathbf{w} + \mathbf{w}^T\mathbf{S}_2\mathbf{w}}} \\
&= \cancel{\frac{(\mathbf{w}^T\mathbf{m}_2 - \mathbf{w}^T\mathbf{m}_1)(\mathbf{w}^T\mathbf{m}_2 - \mathbf{w}^T\mathbf{m}_1)}{\mathbf{w}^T\mathbf{S}_1\mathbf{w} + \mathbf{w}^T\mathbf{S}_2\mathbf{w}}} \\
&= \cancel{\frac{\mathbf{w}^T(\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^T\mathbf{w}}{\mathbf{w}^T\mathbf{S}_1\mathbf{w} + \mathbf{w}^T\mathbf{S}_2\mathbf{w}}} \\
&= \frac{\mathbf{w}^T\mathbf{S}_B\mathbf{w}}{\mathbf{w}^T\mathbf{S}_W\mathbf{w}}
\end{aligned}$$

→ passaggi nelle note.

Deriving w in the binary case: refinement

As usual, $J(\mathbf{w})$ is maximized wrt $\mathbf{w}$ by setting its gradient to $\mathbf{0}$

$$\frac{\partial}{\partial \mathbf{w}} \frac{\mathbf{w}^T \mathbf{S}_B \mathbf{w}}{\mathbf{w}^T \mathbf{S}_W \mathbf{w}} = \cancel{2\frac{(\mathbf{w}^T \mathbf{S}_B \mathbf{w})\mathbf{S}_W \mathbf{w} - (\mathbf{w}^T \mathbf{S}_W \mathbf{w})\mathbf{S}_B \mathbf{w}}{(\mathbf{w}^T \mathbf{S}_W \mathbf{w})(\mathbf{w}^T \mathbf{S}_W \mathbf{w})^T}} = 0$$

which results into

$$(\mathbf{w}^T \mathbf{S}_B \mathbf{w})\mathbf{S}_W \mathbf{w} = (\mathbf{w}^T \mathbf{S}_W \mathbf{w})\mathbf{S}_B \mathbf{w}$$

Serie di considerazioni (c.t.*)

Observe that:

- $\mathbf{w}^T\mathbf{S}_B\mathbf{w}$ is a scalar, say c_B
- $\mathbf{w}^T\mathbf{S}_W\mathbf{w}$ is a scalar, say c_W
- $(\mathbf{m}_2 - \mathbf{m}_1)^T\mathbf{w}$ is a scalar, say c_m

Then, the condition $(\mathbf{w}^T\mathbf{S}_B\mathbf{w})\mathbf{S}_W\mathbf{w} = (\mathbf{w}^T\mathbf{S}_W\mathbf{w})\mathbf{S}_B\mathbf{w}$ can be written as

$$c_B\mathbf{S}_W\mathbf{w} = c_W\mathbf{S}_B\mathbf{w} = c_W(\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^T\mathbf{w} = c_W(\mathbf{m}_2 - \mathbf{m}_1)c_m$$

which results into

$$\mathbf{w} = \frac{c_W c_m}{c_B}\mathbf{S}_W^{-1}(\mathbf{m}_2 - \mathbf{m}_1)$$

Since we are interested into the direction of $\mathbf{w}$, that is in any vector proportional to $\mathbf{w}$, we may consider the solution

$$\hat{\mathbf{w}} = \mathbf{S}_W^{-1}(\mathbf{m}_2 - \mathbf{m}_1) = (\mathbf{S}_1 + \mathbf{S}_2)^{-1}(\mathbf{m}_2 - \mathbf{m}_1)$$

direzione ottima

Possible approach:

- model $p(y|C_i)$ as a gaussian: derive mean and variance by maximum likelihood

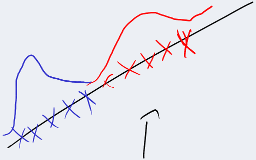
$$m_i = \frac{1}{n_i}\sum_{\mathbf{x}\in C_i} w^T\mathbf{x} \qquad \sigma_i^2 = \frac{1}{n_i - 1}\sum_{\mathbf{x}\in C_i}(w^T\mathbf{x} - m_i)^2$$

  where n_i is the number of items in training set belonging to class C_i
- derive the class probabilities

$$p(C_i|y) \propto p(y|C_i)p(C_i) = p(y|C_i)\frac{n_i}{n_1 + n_2} \propto n_i e^{-\frac{(y-m_i)^2}{2\sigma_i^2}}$$

- the threshold $\tilde{y}$ can be derived as the minimum y such that

$$\frac{p(C_2|y)}{p(C_1|y)} = \frac{n_2}{n_1}\frac{p(y|C_2)}{p(y|C_1)} > 1$$



dov'è il punto di separazione?

Mi comporto in modo Bayesiano: devo considerare il punto dove le due code si incrociano assumono lo stesso valore.

Funziona se le due classi hanno la stessa dimensione, altrimenti applico Bayes

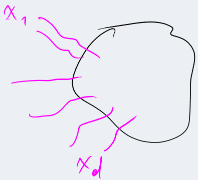
fra $p(C_i|y)$ e $p(y|C_i)$ cambia il prior.

"Ponte verso il futuro": riveste un'importanza significativa perché da questo concetto sono nate le reti neurali.

Mostrato qualche anno dopo che, se non c'è un separatore lineare buono, non funziona.

- Introduced in the '60s, at the basis of the neural network approach
- Simple model of a single neuron
- Hard to evaluate in terms of probability
- Works only in the case that classes are linearly separable

Singolo neurone: segnali da altri neuroni.



L'operazione che il neurone fa: in funzione dei segnali in ingresso, con il loro peso, quindi comb. lineare dei valori, supera una certa soglia $\Rightarrow$ il neurone va in condizione 1 e restituisce valore alto.

$x = x_1 \dots x_d$ associati $[w_1 \dots w_d]$ se $\sum_i w_i x_i > w_0 \Rightarrow$ ridai 1

Definition

It corresponds to a binary classification model where an item $\mathbf{x}$ is classified on the basis of the sign of the value of the linear combination $\mathbf{w}^T\mathbf{x}$. That is,

$$y(\mathbf{x}) = f(\mathbf{w}^T\mathbf{x})$$

$f()$ is essentially the sign function

$$f(i) = \begin{cases} -1 & \text{if } i < 0 \\ 1 & \text{if } i \geq 0 \end{cases}$$

The resulting model is a particular generalized linear model. A special case is ~~the one when f is the identity~~, that is $y(\mathbf{x}) = f(\mathbf{w}^T\mathbf{x})$.

By the definition of the model, $y(\mathbf{x})$ can only be ± 1: we denote $y(\mathbf{x}) = 1$ as $\mathbf{x} \in C_1$ and $y(\mathbf{x}) = -1$ as $\mathbf{x} \in C_2$.

To each element $\mathbf{x}_i$ in the training set, a target value is then associated $t_i \in \{-1, 1\}$.

Devo fare apprendimento: devo definire la funzione di costo, minimizzarla per i valori di w.
Come è fatta: la 0/1 non piace. Se abbiamo i valori di w, la funzione restituisce un valore di costo che voglio diminuire.

- A natural definition of the cost function would be the number of misclassified elements in the training set
- This would result into a piecewise constant function and gradient optimization could not be applied (we would have zero gradient almost everywhere)
- A better choice is using a piecewise linear function as cost function

Se ho un certo valore di w, li cambio di poco, continuerò a sbagliare come prima. Devo allora: cambiarli molto, la f. di costo è costante a tratti:

Applicando il gradiente, dove vado? Non ho una direzione dove miglioro nei punti piatti, nelle discontinuità → ∞.

We would like to find a vector of parameters $\mathbf{w}$ such that, for any $\mathbf{x}_i$, $\mathbf{w}^T\mathbf{x}_i > 0$ if $\mathbf{x}_i \in C_1$ and $\mathbf{w}^T\mathbf{x}_i < 0$ if $\mathbf{x}_i \in C_2$: in short, $\mathbf{w}^T\mathbf{x}_i t_i > 0$.

Each element $\mathbf{x}_i$ provides a contribution to the cost function as follows

1. 0 if $\mathbf{x}_i$ is classified correctly by the model
2. $-\mathbf{w}^T\mathbf{x}_i t_i > 0$ if $\mathbf{x}_i$ is misclassified

Let M be the set of misclassified elements. Then the cost is

nel caso precedente avremmo avuto: $\sum_{x \in M} 1$

è lineare in w, ci piace $\rightarrow$

$$E_p(\mathbf{w}) = -\sum_{\mathbf{x}_i \in M} t_i \mathbf{x}_i^T \mathbf{w}$$

The contribution of $\mathbf{x}_i$ to the cost is 0 if $\mathbf{x}_i \notin \mathcal{M}$ and it is a linear function of $\mathbf{w}$ otherwise

$x_i \in M$: se fisso w ho un certo valore della f. di costo. Cambiando di poco w i valori della f cambiano di poco. Problema: cambia l'insieme dei punti classificati male (M).

$x \in M$: se fisso $\mathbf{w}$ ho un certo valore della funzione di costo (espressione come sopra).

Quando M cambia:
- c'era un elemento classificato bene che ora è classificato male, quindi ho un addendo in più
- c'era un elemento che veniva classificato male che ora è classificato bene, quindi c'è un addendo in meno

Quindi, M e lineare e tutto funziona bene solo se l'insieme è sempre lo stesso. Siccome cambiando anche di poco cambia la somma, passiamo ad un'altra funzione linerare e quindi questo spiega perché è linerare a tratti

The minimum of $E_p(\mathbf{w})$ can be found through gradient descent

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \eta \frac{\partial E_p(\mathbf{w})}{\partial \mathbf{w}}\Big|_{\mathbf{w}^{(k)}}$$

the gradient of the cost function wrt to $\mathbf{w}$ is

$$\frac{\partial E_p(\mathbf{w})}{\partial \mathbf{w}} = -\sum_{\mathbf{x}_i \in M} \mathbf{x}_i t_i$$

Then gradient descent can be expressed as

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \eta \sum_{\mathbf{x}_i \in M_k} \mathbf{x}_i t_i$$

where M_k denotes the set of points misclassified by the model with parameter $\mathbf{w}^{(k)}$

Nel gradiente vedo i punti classificati male che entrano nelle modifica della soluzione.

Adoperata nel perceptrone, dove considero un elemento alla volta

Online (or stochastic gradient descent): at each step, only the gradient wrt a single item is considered

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \eta \mathbf{x}_i t_i$$

where $\mathbf{x}_i \in M_k$ and the *scale factor* $\eta > 0$ controls the impact of a badly classified item on the cost function

The method works by circularly iterating on all elements and applying the above formula.

Ho una certa soluzione in un certo punto: calcolo il gradiente su un solo punto. Qui, vuol dire che: considero un punto

- può essere ben classificato ⇒ contributo 0, gradiente = 0 quindi $\mathbf{w}$ non cambia
- mal classificato: contributo $\mathbf{x}_i t_i$

Modifico il valore dei coefficienti sommando il punto, sposta l'iperpiano di separazione nella direzione di giusta classificazione.

f: sgn (funzione segno)

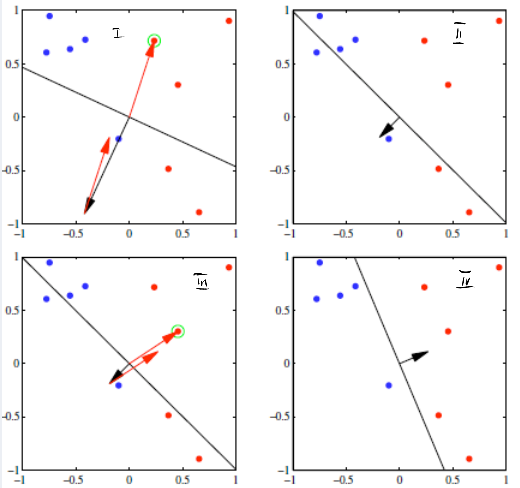
Initialize $\mathbf{w}^0$
$k := 0$ **repeat**
  $k := k + 1$
  $i := (k \bmod n) + 1$
  $y := f(\mathbf{w}^T \mathbf{x}_i) t_i$
  if $y > 0$ **then** $\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)}$ → elementi classificati bene.
  else $\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} + \eta \mathbf{x}_i t_i$
until all elements are well classified

↳ ip: c'è un separatore lineare, allora converge

I stante iniziale

Il punto rosso (O) e' mal classificato. Il vettore non rappresenta l' iperpiano di separazione.

Sommiamo i due vettori, otteniamo il 2° rosso => l'iperpiano si gira (II)



Geometricamente:

Sommo il vettore dei coefficienti col vettore dei punti.

In black, decision boundary and corresponding parameter vector $\mathbf{w}$; in red misclassified item vector $\mathbf{x}_i$, added by the algorithm to the parameter vector as $\eta\mathbf{x}_i$

At each step, if $\mathbf{x}_i$ is well classified then $\mathbf{w}^{(k)}$ is unchanged; else, its contribution to the cost is modified as follows

$$
\begin{aligned}
-\mathbf{x}_i^T \mathbf{w}^{(k+1)} t_i &= -\mathbf{x}_i^T \mathbf{w}^{(k)} t_i - \eta (\mathbf{x}_i t_i)^T \mathbf{x}_i t_i \\
&= -\mathbf{x}_i^T \mathbf{w}^{(k)} t_i - \eta ||\mathbf{x}_i||^2 \\
&< -\mathbf{x}_i^T \mathbf{w}^{(k)} t_i
\end{aligned}
$$

This contribution is decreasing, however this does not guarantee the convergence of the method, since the cost function could increase due to some other element becoming misclassified if $\mathbf{w}^{(k+1)}$ is used

L'effetto della modifica e' far si che diminuisca il costo al contributo del singolo elemento, ma localmente, non e' detto che globalmente sia così. Si cerca di classificare meglio il punto.

It is possible to prove that, in the case the classes are linearly separable, the algorithm converges to the correct solution in a finite number of steps.

~~Let $\hat{\mathbf{w}}$ be a solution (that is, it discriminates C_1 and C_2): if $\mathbf{x}_{k+1}$ is the element considered at iteration $(k+1)$ and it is misclassified, then~~

$$\mathbf{w}^{(k+1)} - \alpha\hat{\mathbf{w}} = (\mathbf{w}^{(k)} - \alpha\hat{\mathbf{w}}) + \eta\mathbf{x}_{k+1}t_{k+1}$$

where $\alpha > 0$ is a constant, to be specified later

E' possibile mostrare che:

1- se c'e' separabilità lineare, in un n° finito di passi, si trova il separatore lineare (l'ottimo)

2- c'e' un upper bound sul n° di passi necessari