

Data science e classificazione

Università di Bologna

8 Aprile 2025

Introduzione a Data Science

Fissiamo le notazioni

Abbiamo già visto che le variabili possono essere divise in due categorie principali:

- **Numeriche o Quantitative:** Variabili rappresentanti numeri. Sono contraddistinte da due caratteristiche importanti:
 - ① Ammettono un ordinamento naturale.
 - ② Su di esse possono essere eseguite operazioni.
- **Categoriche o Qualitative:** Variabili rappresentanti un insieme finito di etichette (ad esempio il Genere, che ammette le etichette M e F). Le etichette rappresentabili da una variabile di questo tipo vengono spesso dette classi.

Esempio

In un data frame del tipo

	<i>Figura</i>	<i>Perimetro</i>	<i>Area</i>	<i>Num..Lati</i>
1	Triangolo	10	7	3
2	Quadrato	4	3	4
3	Pentagono	11	1	5

*le variabili corrispondenti alle colonne Figura e Num..Lati sono di tipo **Categorico**, mentre le colonne corrispondenti alle colonne Perimetro e Area sono di tipo **Numerico**.*

Example

In un dataset rappresentante la temperatura media di ogni stato ogni anno dal 2010 al 2019, la variabile corrispondente alle colonne Nome Stato e Anno sono di tipo **Categorico** mentre la variabile corrispondente alla colonna Temperatura è di tipo **Numerica**.

Definizione del problema

Consideriamo il seguente dataset

	Genere	Altezza	Peso	Col..Occhi
1	M	1.71	80	Marroni
2	F	1.60	60	Azzurri
3	M	1.80	74	Marroni
4	M	1.78	92	Verdi
5	F	1.55	56	Marroni

Potremmo chiederci se sia possibile sfruttare le informazioni contenute nel dataset per prevedere se una persona sia M o F conoscendo Altezza, Peso e Col..Occhi. Ponendoci questa domanda, abbiamo naturalmente diviso le variabili (le colonne) del dataset in due parti:

Definizione del problema

- L'insieme delle variabili che vogliamo prevedere (in questo caso, solo **Genere**). Queste sono dette **variabili target** o **output** e vengono solitamente indicate con la lettera y.
- L'insieme delle **variabili di input** (tutte le altre), contenenti le informazioni che possiamo sfruttare per prevedere le variabili di output. Queste vengono solitamente indicate con la lettera x.

Tipologia di Variabili

In genere, sia le variabili di input x che quelle di output y sono dei vettori, e sono quindi rappresentabili come:

$$x = (x_1, x_2, \dots, x_p)$$

$$y = (y_1, y_2, \dots, y_m)$$

Dove ciascuna x_i, y_j rappresenta una colonna differente del dataset.

Se x è una variabile scalare, il problema si dice univariato, mentre se $x = (x_1, x_2, \dots, x_p)$ è vettoriale, si dice multivariato.

variabile che
contiene un solo valore

Tipologia di Problemi di Machine Learning

I problemi di Machine Learning possono essere divisi in tre categorie principali, che dipendono fortemente dalla struttura del dataset:

- **Apprendimento Supervisionato:** quando sono presenti sia la variabile di input x sia la variabile di output y .
- **Apprendimento Non-Supervisionato:** quando la variabile di output y non è presente nel dataset. In questo caso, si vogliono semplicemente cercare delle strutture (dette *clusters*) tra le variabili di input.
- **Apprendimento Semi-Supervisionato:** quando le variabili di output y sono presenti soltanto in una frazione ridotta delle osservazioni.

Che tipologia di problemi affronteremo?

Per semplicità, supporremo sempre di avere a disposizione problemi di tipo Supervisionato. Ci mettiamo quindi nel caso in cui il dataset \mathcal{S} sia composto da N osservazioni $\{(x_i, y_i)\}_{i=1}^N$ con $x_i = (x_{i,1}, \dots, x_{i,p})$ variabile di input multivariata, di dimensione p e $y_i = (y_{i,1}, \dots, y_{i,m})$ variabile di output.

Tipologia dei Problemi di Machine Learning

Tra i problemi Supervisionati si riconoscono due classi, dipendenti dalla natura della variabile di output y :

- **Regressione:** quando la variabile di output y è di tipo *numerico*. Ad esempio, la Regressione Lineare (univariata o multivariata) che avete già studiato, è una tecnica di tipo regressiva.
- **Classificazione:** quando la variabile di output y è di tipo *categorico*. Ad esempio, la Regressione Logistica e le Support Vector Machines (che andremo a studiare in queste lezioni) sono algoritmi di Classificazione.

Esempio Classificazione

La domanda che ci eravamo posti in precedenza sul dataset:

	Genere	Altezza	Peso	Col..Occhi
1	M	1.71	80	Marroni
2	F	1.60	60	Azzurri
3	M	1.80	74	Marroni
4	M	1.78	92	Verdi
5	F	1.55	56	Marroni

In cui ci chiedavamo se fosse possibile predire il Genere dati Altezza, Peso, Col..Occhi è un problema di **Classificazione**, poiché la variabile di output è $y = \text{Genere}$, che è di tipo *categorico*, e le classi possibili per y sono $\mathcal{C} = \{M, F\}$.

Indicheremo sempre con \mathcal{C} l'insieme delle classi ammissibili per una variabile categorica.

Esempio Regressione

Un esempio di problema di *Regressione multivariata* è quello in cui ci chiediamo se, preso il dataset

	Stato	Mese	Temperatura
1	Italia	Febbraio	7
2	Francia	Marzo	14
3	Spagna	Giugno	30
4	Germania	Febbraio	0
5	Russia	Aprile	15

Sia possibile prevedere la temperatura ($y = \text{Temperatura}$) dati $x = \{\text{Stato}, \text{Mese}\}$.

numerica

categorica

Un po' di teoria


Introduzione

per ciascuna osservazione, $x \in \mathbb{R}^p$, $y \in \mathbb{R}$
input (features) output (target)



- Come già detto, in un problema di apprendimento supervisionato, vorremmo sfruttare le informazioni contenute in una variabile di input $x \in \mathbb{R}^p$ per cercare di prevedere una variabile di output $y \in \mathbb{R}^m$. Per semplicità, consideremo il caso $m = 1$ per cui $y \in \mathbb{R}$.
- La relazione tra x e y è modellizzata da una funzione $h : \mathbb{R}^p \rightarrow \mathbb{R}$ detta **predittore**, che associa ad ogni variabile x la corrispondente y .

$h_\theta \rightarrow$ funzione che dipende da parametri θ .

- In generale, il predittore dipenderà da un'insieme di parametri, identificati come θ . La relazione tra il predittore i suoi parametri definiscono l'algoritmo. 
- Chiaramente, la speranza è quella di far sì che il nostro predittore sia tale che

$$h_\theta(x_i) \approx y_i \quad \forall i = 1, \dots, N.$$

- Questo viene fatto fissando una misura di errore, chiamata **funzione di loss** $\ell(y, y')$, e scegliendo i parametri θ che risolvono:

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^N \ell(h_\theta(x_i), y_i) \quad (1) \quad \leftarrow$$

Trova il parametro θ minimo affinché $h_\theta(x_i) \approx y_i$

Empiric Risk minimization (ERM)

- La funzione

$$R(\theta) = \frac{1}{N} \sum_{i=1}^N \ell(h_{\theta}(x_i), y_i)$$

è detta **funzione di rischio empirico**, e la tecnica dell'ottimizzare θ tale che minimizzi $R(\theta)$, viene detta **minimizzazione del rischio empirico (ERM)**.

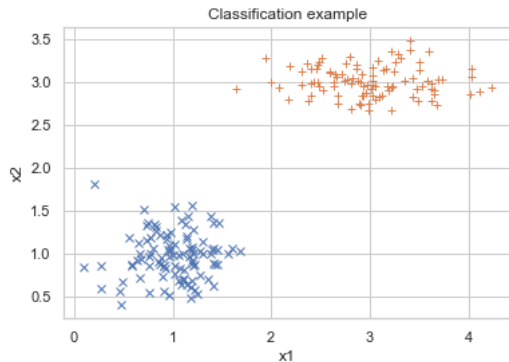
- Risolvere (1), viene solitamente detto **addestramento (training)**. \rightarrow risolvere la funzione di loss

Classificazione

Andremo ora ad introdurre in maniera più approfondita i problemi di classificazione. Ricordiamo che un problema di classificazione è un problema in cui la variabile di output y è di tipo Categorico. In questo caso, y potrà assumere soltanto un numero finito di valori, detti **classi**. Indichiamo con \mathcal{C} l'insieme delle classi ammissibili per y , e con C_k , $k = 1, \dots, K$ il valore della classe k-esima.

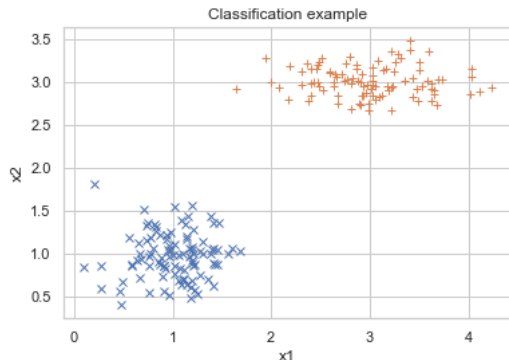
Introduzione

Consideriamo quindi il problema seguente, in cui $x = \{x_1, x_2\}$, y variabile categorica con $\mathcal{C} = \{+, x\}$. Rappresentando su un grafico bidimensionale i valori di x contenuti in \mathcal{S} , otteniamo il grafico:



Introduzione

Consideriamo quindi il problema seguente, in cui $x = \{x_1, x_2\}$, y variabile categorica con $\mathcal{C} = \{+, x\}$. Rappresentando su un grafico bidimensionale i valori di x contenuti in \mathcal{S} , otteniamo il grafico:



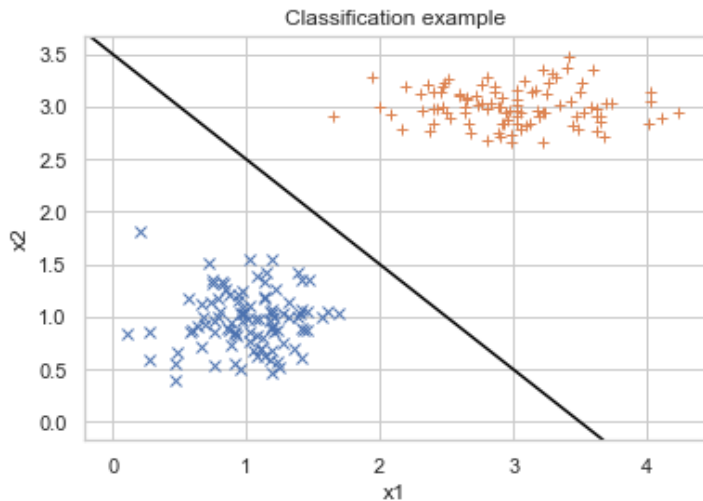
Come facciamo ad implementare un algoritmo che, sfruttando i dati presenti in \mathcal{S} , impari a classificare dei nuovi punti come $+$ o x , conoscendo x_1 e x_2 ?

Il modo più ovvio, è quello di tracciare una **retta che separi le due classi**, e poi assegnare alla classe + tutti i punti sopra la retta, alla classe x tutti i punti sotto di essa. In pratica, data una retta Π di equazione $ax_1 + bx_2 + c = 0$, definiamo il **predittore**:

$$h_{\theta}(x_1, x_2) = \begin{cases} + & \text{se } ax_1 + bx_2 + c > 0 \\ x & \text{se } ax_1 + bx_2 + c < 0 \end{cases} \quad (2)$$

questa volta, i parametri θ sono i coefficienti (a, b, c) della retta.

\Rightarrow un predittore che separa le classi attraverso una retta, è detto **predittore lineare**.



Notare come, in questo esempio, la retta tracciata separi perfettamente le due classi.

Definizione

*Dato un problema di classificazione di classi \mathcal{C} su un dataset \mathcal{S} , una retta (se esiste) Π di equazione $ax_1 + bx_2 + c = 0$ che separa perfettamente le due classi è detta **retta separatrice**.*

Definizione

*Un dataset \mathcal{S} per cui esiste almeno una retta separatrice Π , si dice **linearmente separabile**.*

La Regressione Logistica

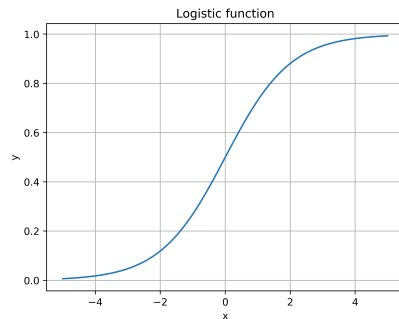
Nel mio caso ho addestrato il modello con SVM, potevo utilizzare anche la regressione logistica (ho scelto SVM perché visto più a lezione)

Definizione

- A differenza di quanto suggerisce il nome, la **regressione logistica** è uno dei più semplici **classificatori lineari**.

- Si basa su una funzione, detta **funzione logistica** o **sigmoide**, definita da:

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \quad (3)$$



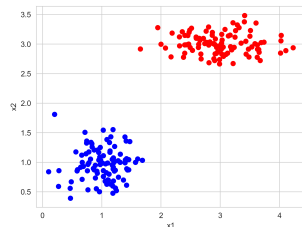
- In particolare, il predittore di una **Regressione Logistica** è definito da:

$$\wedge \quad h_{\theta}(x_1, x_2) = \frac{1}{1 + e^{-(ax_1 + bx_2 + c)}}. \quad (4)$$

- Nota che la funzione logistica assume valori compresi nell'intervallo $[0, 1] \implies$ il valore di $h_{\theta}(x_1, x_2)$ può essere visto come la probabilità che h_{θ} attribuisce al punto (x_1, x_2) di appartenere alla classe 1. Di conseguenza, (x_1, x_2) la classe assegnata a (x_1, x_2) sarà:

$$\begin{cases} 1 & \text{se } h_{\theta}(x_1, x_2) > 0.5, \\ 0 & \text{se } h_{\theta}(x_1, x_2) < 0.5. \end{cases} \quad (5)$$

- Consideriamo il dataset `Classification_data.csv`, presente su Virtuale.



- Si può definire il modello di Regressione Logistica con il comando:

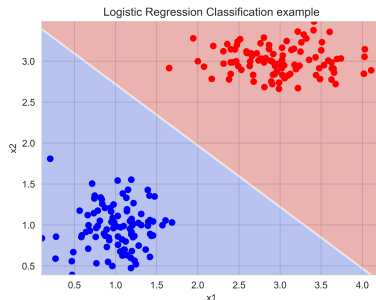
```
from sklearn.linear_model import LogisticRegression  
model = LogisticRegression()
```

- Per poi addestrarlo:

```
model.fit(df[["x1", "x2"]], df["class"])
```

Implementazione

- Una volta addestrato, per utilizzarlo per fare predizioni si utilizza la funzione:
`model.predict(test_data)`
- E' anche possibile visualizzare la retta separatrice esplicitamente:



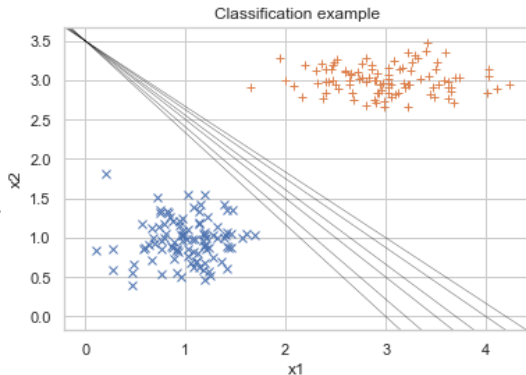
Classificatore a Massimo Margine (MMC)

È chiaro che, se S è linearmente separabile, allora esistono infinite rette separatrici. Quale scegliamo?

Se S è linearmente separabile, allora esistono infinite rette separatrici



Quale scegliamo?



Classificatore a Massimo Margine (MMC)

Per rispondere a questa domanda, dobbiamo ricordarci che lo scopo del classificatore è quello di predire correttamente la giusta classe di nuove osservazioni date in input. Per questo motivo, la cosa più naturale è scegliere, tra tutte le possibili rette separatrici, quella che massimizza la distanza tra le due classi, ovvero la retta Π tale che:

$$\Pi = \max_{\Pi} \min_{i=1,2} d(\Pi, C_i)$$

→ Trovo quella che massimizza la distanza tra le due classi

Classificatore a Massimo Margine (MMC)

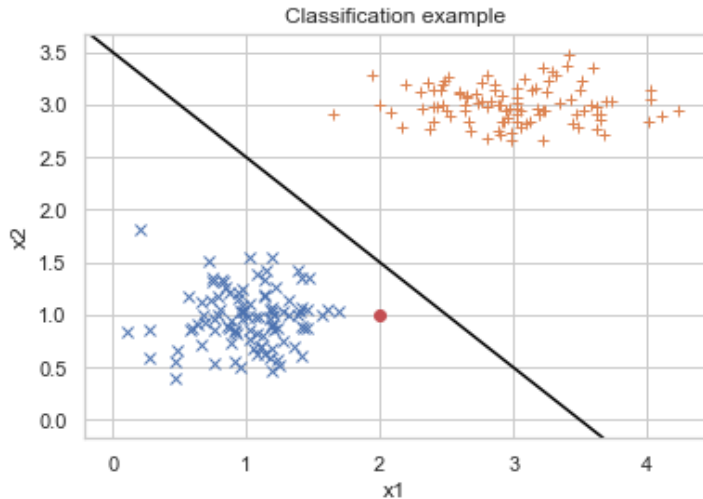
Una volta identificata la retta separatrice che massimizza la distanza tra le due classi, definiamo il predittore

$$h_{\theta}(x_1, x_2) = \begin{cases} + & \text{se } ax_1 + bx_2 + c > 0 \\ x & \text{se } ax_1 + bx_2 + c < 0 \end{cases} \quad (6)$$

Questo classificatore è detto **Classificatore a Massimo Margine (MMC)**.

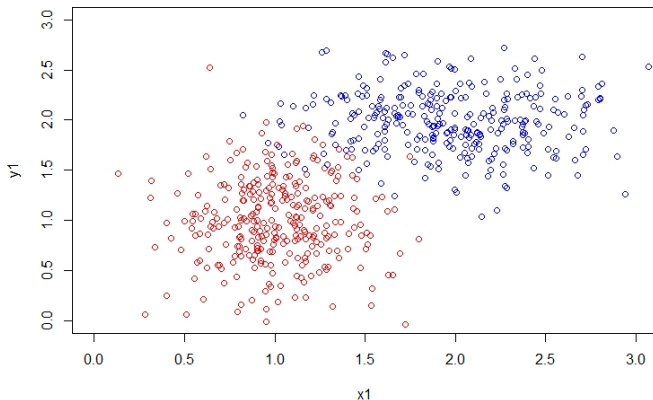
MMC \rightarrow consente di individuare la migliore distanza tra le due classi

Classificatore a Massimo Margine (MMC)



Support Vector Classifier (SVC)

Nella pratica, i dati non sono praticamente **mai** linearmente separabili. Infatti, come è possibile osservare nel seguente esempio, la maggior parte delle volte le classi si sovrappongono.



Support Vector Classifier (SVC)



In questo caso, non esiste nessuna retta separatrice tra le due classi, e quindi non è possibile definire il MMC. Per risolvere questo problema, si introduce un'iperparametro $C > 0$ detto **costo**, che controlla il numero massimo di punti che possono oltrepassare la linea di separazione.

Si definisce in questo modo una nuova retta $\Pi(C)$ di separazione e, di conseguenza, un predittore $h_{\theta}(x_1, x_2)$ che viene detto **Support Vector Classifier (SVC)**.

iperparametro costo $C \rightarrow C > 0$

Controlla il numero massimo di punti che possono oltrepassare la linea di separazione.

Per implementare MMC e SVC, è necessario utilizzare la libreria `sklearn`, che permette di lavorare con alcune funzioni di Machine Learning.

```
from sklearn.svm import SVC
```

dopodiché sarà sufficiente creare il modello con la funzione `SVC` di `sklearn.svm` nel seguente semplice modo:

```
model = SVC(kernel='linear')  
model.fit(df[['x1','x2']], df['class'])
```

Chiaramente, estendere un MMC al caso non linearmente separabile è banale. Basta cambiare il parametro di costo C della funzione SVC.

```
model = SVC(kernel='linear', C=10)
model.fit(df[['x1', 'x2']], df['class'])
```

(il dataframe `df` utilizzato negli esempi si trova nella cartella `data`, sotto il nome di `SVC_example.csv`).

Estensione al caso p -dimensionale

Fino ad ora abbiamo visto soltanto casi in cui la **variabile di input x aveva due dimensioni** ($p = 2$). L'estensione al caso in cui $p > 2$ è banale.

Definizione

In \mathbb{R}^p , un'**iperpiano** è il luogo dei punti che rispettano l'equazione

$$a_0 + \sum_{i=1}^p a_i x_i = a_0 + a_1 x_1 + a_2 x_2 + \cdots + a_p x_p = 0 \quad (7)$$

- L'iperpiano è l'estensione del concetto di retta in $p > 2$.
- Per $p = 3$, l'iperpiano è il piano.
- Un'equazione del tipo (7) è detta **Equazione Lineare**.

Linearmente Separabili

Usando la definizione di iperpiano, possiamo estendere in maniera naturale il concetto di linearmente separabili al caso p -dimensionale.

Definizione

Dato un problema di classificazione di classi \mathcal{C} su un dataset S , un iperpiano (se esiste) Π di equazione $a_0 + \sum_{i=1}^p a_i x_i = 0$ che separa perfettamente le due classi è detto iperpiano separatore.

Definizione

Un dataset S per cui esiste almeno un iperpiano separatore Π , si dice linearmente separabile.

Classificatore a Massimo Margine (MMC)

Di conseguenza, se S è linearmente separabile, è possibile definire (con un problema di minimo simile a quello visto per il caso bidimensionale) il concetto di **iperpiano a massimo margine**, da cui possiamo definire il **Classificatore a Massimo Margine (MMC)** per il caso p -dimensionale come

$$\mathcal{N} \quad h_{\theta}(x) = \begin{cases} + & \text{se } a_0 + \sum_{i=1}^p a_i x_i > 0 \\ x & \text{se } a_0 + \sum_{i=1}^p a_i x_i < 0 \end{cases} \quad (8)$$

dove l'iperpiano a massimo margine Π ha equazione

$$a_0 + \sum_{i=1}^p a_i x_i = 0.$$

Support Vector Classifier (SVC)

Nel caso di dati non linearmente separabili, aggiungendo il parametro di costo $C > 0$, è possibile definire anche il SVC in dimensione p allo stesso modo del MMC.
Anche dal punto di vista dell'implementazione, il codice è pressoché invariato.

Support Vector Machines (SVM)

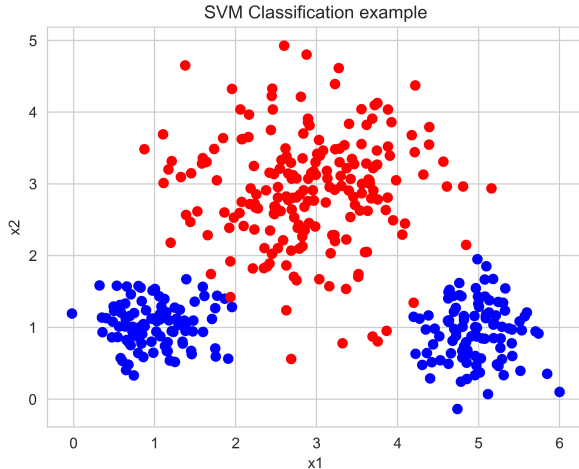
Dati non linearmente separabili

- Entrambi gli algoritmi visti fino ad ora hanno in comune il fatto che i loro predittori $h_\theta(x)$ distinguono le due classi basandosi sulla posizione dell'input rispetto ad un iperpiano. Nel caso $p = 2$, ad esempio, le due classi vengono separate da una retta.
- In casi come questo, in cui il predittore separa le classi con delle curve, tali curve vengono dette **curve separatrici**. In MMC e SVC, le curve separatrici sono delle rette.
- Ci sono casi di dataset che non sono linearmente separabili, ma per cui esistono delle curve "semplici" in grado di separare perfettamente le due classi.

↳ polinomio

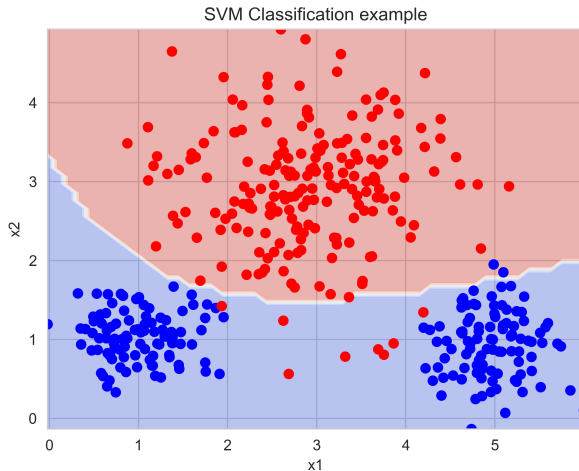
Dati separabili da polinomi

Esempio di dati separabili da un polinomio.



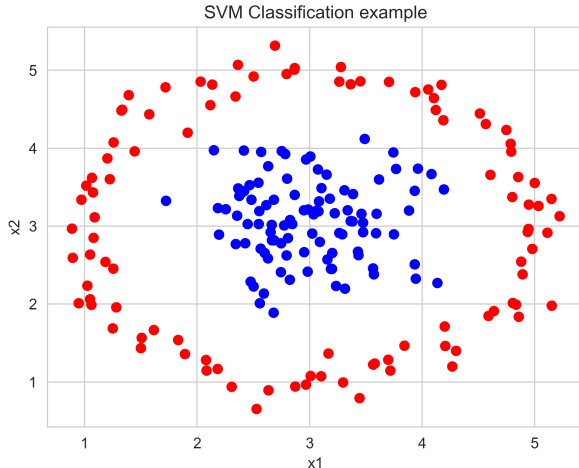
Dati separabili da polinomi

Esempio di dati separabili da un polinomio.



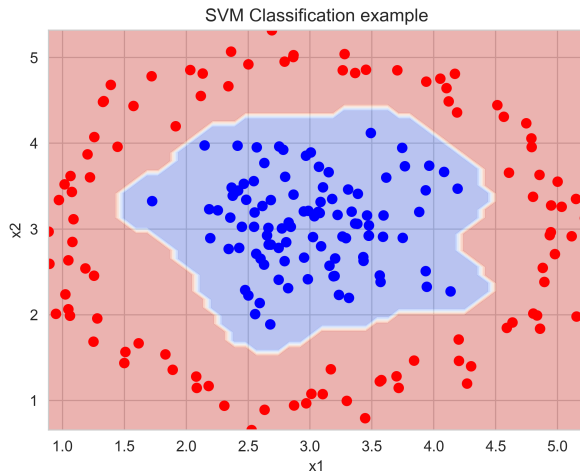
Dati separabili da circonferenze

Esempio di dati separabili da circonferenze.



Dati separabili da circonferenze

Esempio di dati separabili da circonferenze.



Support Vector Machines (SVM)

In questi casi, è possibile sfruttare esplicitamente la forma caratteristica della curva separatrice per migliorare di molto l'algoritmo di classificazione. L'idea è quella di fissare una funzione non lineare $\phi : \mathbb{R}^p \rightarrow \mathbb{R}^d$ tale che $\phi(\mathcal{S})$ (ovvero il dataset ottenuto trasformando \mathcal{S} con ϕ), sia linearmente separabile.

La forma di tale funzione, detta **kernel function**, esplicita la forma caratteristica del dataset, per semplificarlo e renderlo, appunto, linearmente separabile.

Support Vector Machines (SVM)

Una volta fissata la kernel function ϕ , data la lineare separabilità (o quasi) di $\phi(\mathcal{S})$, è possibile addestrare un SVC sul dataset trasformato, ovvero trovare un'iperpiano $\Pi(C)$ di equazione

$$a_0 + \sum_{i=1}^p a_i \phi(x_i) = 0 \quad (9)$$

con costo $C > 0$, tale che il classificatore:

$$h_{\theta}(x) = \begin{cases} + & \text{se } a_0 + \sum_{i=1}^p a_i \phi(x_i) > 0 \\ x & \text{se } a_0 + \sum_{i=1}^p a_i \phi(x_i) < 0 \end{cases} \quad (10)$$

abbia come curva di separazione non più una retta (in \mathbb{R}^p), ma una curva la cui forma dipende dalla scelta della funzione kernel ϕ .

Tipi di Kernel

La funzione `SVC()`. Questa prende in input il parametro `kernel` che descrive la forma della funzione ϕ che definisce la SVM. Le possibili scelte di ϕ sono:

- **linear**: $\phi(x) = x$, in questo caso si ottiene SVC, poiché la funzione ϕ è l'identità.
- **poly**: $\phi(x) = (1 + x)^d$, con **kernel polinomiale** è necessario inserire anche il parametro `degree` che definisce il **grado del polinomio (l'esponente d)**.
- **rbf**: $\phi(x) = \exp(-\frac{x^2}{\gamma})$, con **kernel radiale (esponenziale)** è necessario inserire anche il parametro **gamma** che definisce la **varianza della distribuzione**.
- **sigmoid**: $\phi(x) = \frac{1}{1 + \exp(-x)}$ \rightarrow **kernel iperbolico**