Capitolo 1

Dare ai computer la capacità di apprendere dai dati

In questo capitolo

- **Costruire macchine intelligenti per trasformare i dati in conoscenza**
- **I tre diversi tipi di machine learning**
- **Introduzione alla terminologia e alla notazione di base**
- **Una roadmap per la realizzazione di sistemi di apprendimento automatico**
- **Usare Python per attività di machine learning**

È mia opinione che il *machine learning*, o apprendimento automatico, ovvero l'applicazione e lo studio degli algoritmi che estraggono informazioni utili dalla massa informe dei dati, sia il campo più interessante dell'informatica. Viviamo in un'era in cui i dati sono disponibili in una quantità sovrabbondante; utilizzando gli algoritmi di autoapprendimento nel campo del machine learning, siamo in grado di trasformare questi dati in *conoscenza*. Grazie alle tante e sempre più potenti librerie open source che sono state sviluppate nel corso degli ultimi anni, non vi è probabilmente mai stato un momento migliore per occuparsi di machine learning e per imparare a utilizzare questi potenti algoritmi per individuare schemi nei dati ed eseguire previsioni a proposito degli eventi futuri.

In questo capitolo parleremo dei concetti principali e dei vari tipi di machine learning. Dopo un'introduzione alla terminologia di base, getteremo le basi per l'utilizzo delle tecniche di machine learning per la risoluzione di problemi pratici.

In questo capitolo, affronteremo i seguenti argomenti.

- Concetti generali di machine learning.
- I tre tipi di apprendimento e la terminologia di base.
- Gli elementi costitutivi per la realizzazione di sistemi di machine learning.
- Installazione e configurazione di Python per l'analisi dei dati e il machine learning.

Costruire macchine intelligenti per trasformare i dati in conoscenza

In questa nuova era di tecnologie, vi è una risorsa che abbiamo disponibile in grande abbondanza: abbiamo una grande quantità di dati strutturati e non strutturati. Nella seconda metà del ventesimo secolo, il machine learning si è evoluto come una branca dell'*intelligenza artificiale*, che prevede lo sviluppo di algoritmi di autoapprendimento in grado di acquisire conoscenze dai dati, con lo scopo di effettuare previsioni. Invece di richiedere una presenza umana, in grado di individuare manualmente delle regole e costruire dei modelli per l'analisi di grandi quantità di dati, il machine learning offre un'alternativa più efficiente per catturare la conoscenza insita nei dati, col fine di migliorare gradualmente le prestazioni dei modelli previsionali e poi prendere decisioni guidate dai dati. Non solo il machine learning sta diventando sempre più importante nel campo della ricerca basata su computer, ma gioca anche un ruolo sempre più preminente nella nostra vita quotidiana. Grazie al machine learning, possiamo contare su solidi filtri in grado di eliminare lo spam dalla nostra posta elettronica, su comodi software per il riconoscimento del testo e della voce, su affidabili motori di ricerca per il Web, su giocatori artificiali di scacchi sempre più abili e, speriamo al più presto, su automobili a guida automatica, che siano sicure ed efficienti.

I tre diversi tipi di machine learning

In questo paragrafo esamineremo i tre diversi tipi di machine learning: *apprendimento con supervisione*, *apprendimento senza supervisione* e *apprendimento di rafforzamento*. Esamineremo le differenze fondamentali che esistono fra questi tre diversi tipi di apprendimento (Figura 1.1) e, utilizzando esempi concettuali, svilupperemo alcune idee relative ai domini pratici dei problemi ai quali tali tecniche possono essere applicate.

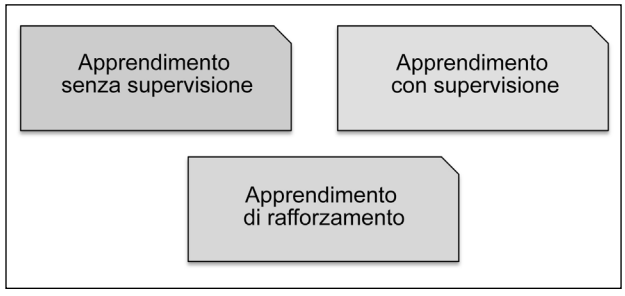


Figura 1.1

Effettuare previsioni sul futuro grazie all'apprendimento con supervisione

Lo scopo principale dell'apprendimento con supervisione consiste nel trarre un modello a partire da dati di *addestramento* etichettati, i quali ci consentono di effettuare previsioni

relative a dati non disponibili o futuri (Figura 1.2). Qui il termine *con supervisione* fa riferimento al fatto che nell'insieme di campioni i segnali di output desiderati (le etichette) sono già noti.
Considerando l'esempio del filtraggio dei messaggi spam di posta elettronica, possiamo addestrare un modello applicando un algoritmo di apprendimento con supervisione a un insieme di messaggi di posta elettronica già etichettati, che siano stati correttamente contrassegnati come spam oppure non-spam, per fargli determinare se un nuovo messaggio di posta elettronica appartiene all'una o all'altra categoria. Un compito di apprendimento con supervisione, sulla base di *etichette delle classi* discrete, come nel precedente esempio del filtraggio dello spam nella posta elettronica, è chiamato anche compito di *classificazione*. Un'altra sottocategoria di apprendimento con supervisione è la *regressione*, dove il segnale risultante è un valore continuo.

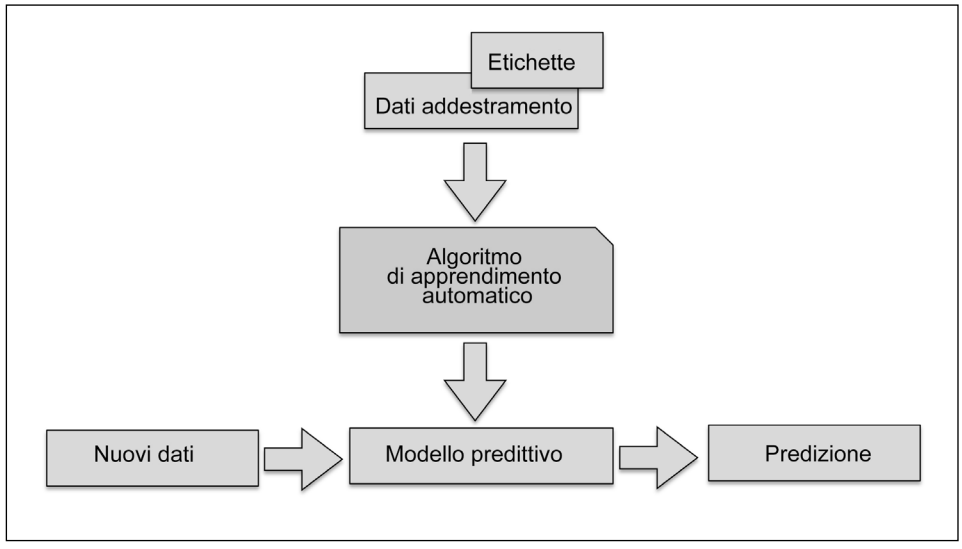


Figura 1.2

Classificazione per la predizione delle etichette delle classi

La classificazione è una sottocategoria dell'apprendimento con supervisione, dove l'obiettivo è quello di prevedere le etichette di categoria delle classi per le nuove istanze, sulla base delle osservazioni compiute nel passato. Queste etichette sono valori discreti, non ordinati, che possono essere considerati come *appartenenti a un gruppo* delle istanze. L'esempio menzionato in precedenza del rilevamento dello spam nella posta elettronica rappresenta un tipico esempio di *classificazione binaria*: l'algoritmo di apprendimento automatico impara un insieme di regole con lo scopo di distinguere fra due possibili classi: messaggi di posta elettronica che sono spam o che sono non-spam.
Tuttavia, l'insieme delle etichette delle classi non deve necessariamente avere una natura binaria. Il modello predittivo individuato da un algoritmo di apprendimento con supervisione può considerare ogni etichetta della classe che sia presente nel dataset di apprendimento di una nuova istanza che non sia dotata di etichetta. Un tipico esempio di un compito di *classificazione multiclasse* è il riconoscimento del testo scritto a mano.

Qui possiamo raccogliere un dataset di apprendimento che è costituito da più esempi di scrittura a mano di ciascuna lettera dell'alfabeto. Ora, se un utente fornisce un nuovo carattere scritto a mano tramite un dispositivo di input, il nostro modello predittivo sarà in grado di prevedere con una certa precisione la lettera corretta dell'alfabeto. Tuttavia, il nostro sistema di apprendimento automatico non sarebbe in grado di riconoscere correttamente le cifre da 0 a 9, per esempio, se queste non facevano già parte del dataset di apprendimento.

La Figura 1.3 illustra il concetto del compito di classificazione binaria sulla base di campioni di apprendimento: quindici di essi sono etichettati come *classe negativa* (O) e altrettanti campioni sono etichettati come *classe positiva* (+). In questa situazione, il nostro dataset è bidimensionale, il che significa che a ogni campione possono essere associati due colori: x_1 e x_2. Ora, possiamo utilizzare un algoritmo di apprendimento automatico con supervisione per trarre una regola (il confine decisionale è rappresentato dalla linea tratteggiata) che sia in grado di separare queste due classi e classificare i nuovi dati in ognuna di queste due categorie sulla base dei loro valori x_1 e x_2.

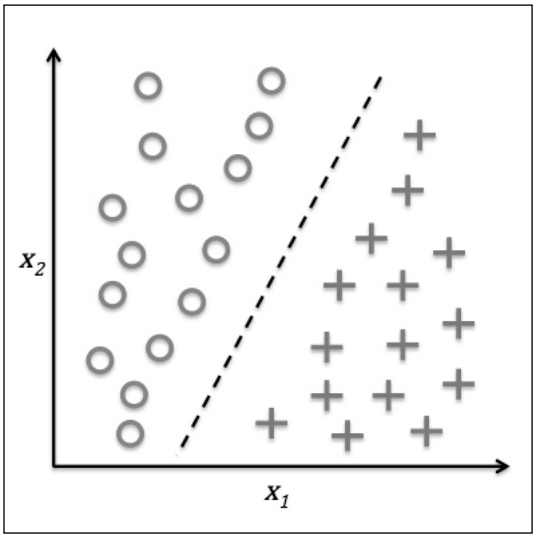


Figura 1.3

Regressione per la previsione di risultati continui

Nel paragrafo precedente abbiamo visto che il compito di classificazione consiste nell'assegnare alle istanze etichette di categorie non ordinate. Un secondo tipo di apprendimento con supervisione è la previsione di risultati continui, chiamata anche analisi di regressione. Nell'*analisi di regressione*, abbiamo un certo numero di variabili predittive (descrittive) e una variabile target continua (risultato): cerchiamo di trovare una relazione fra queste variabili, tale che ci consenta di prevedere un risultato.

Per esempio, supponiamo di essere interessati a prevedere le valutazioni di una prova scritta dei nostri studenti. Se vi è una relazione fra il tempo dedicato a studiare per il test e i risultati finali, potremmo utilizzare questi tempi come dati di apprendimento per derivare un modello che utilizzi proprio il tempo dedicato allo studio per prevedere le valutazioni dei futuri studenti che pensano di svolgere questo test.

NOTA

Il termine *regressione* è stato coniato da Francis Galton nel suo articolo *Regression Towards Mediocrity in Hereditary Stature,* scritto nel 1886. Galton ha descritto un fenomeno biologico in base al quale la varianza di *altezza* in una popolazione non aumenta nel corso del tempo. Ha osservato che la statura dei genitori non viene trasmessa ai figli, ma regredisce nella direzione della media della popolazione.

La Figura 1.4 illustra il concetto di *regressione lineare*. Data una variabile predittiva x e una variabile risposta y, tracciamo una linea retta attraverso questi dati in modo da minimizzare la distanza (si parla infatti di distanza quadratica media) fra i punti del campione e la linea. Ora possiamo utilizzare il punto di intersezione e la pendenza che abbiamo appreso da questi dati per prevedere la variabile target per nuovi dati.

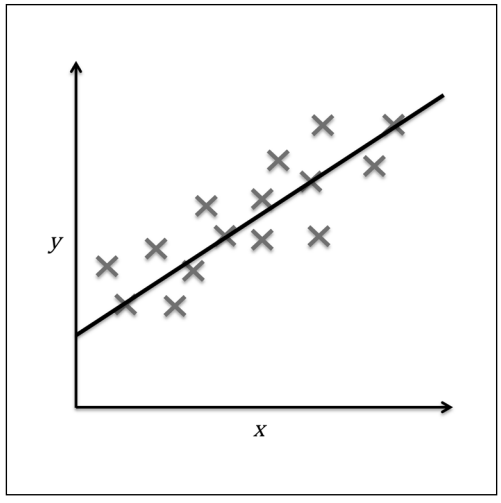


Figura 1.4

Risolvere problemi interattivi con l'apprendimento di rafforzamento

Un altro tipo di apprendimento automatico è l'apprendimento di rafforzamento. Qui l'obiettivo è quello di sviluppare un sistema (*agente*) che migliori le proprie prestazioni sulla base delle interazioni con l'*ambiente*. Poiché, tipicamente, le informazioni relative allo stato corrente dell'ambiente includono anche un cosiddetto segnale di *ricompensa* (*reward*), possiamo considerare l'apprendimento di rafforzamento come un esempio di apprendimento *con supervisione*. Tuttavia, nell'apprendimento di rafforzamento, questo feedback non è l'etichetta o il valore corretto di verità, ma una misura della qualità con cui l'azione è stata misurata da una funzione di *ricompensa*. Tramite l'interazione con l'ambiente, un agente può quindi utilizzare l'apprendimento di rafforzamento per imparare una serie di *azioni* che massimizzano questa *ricompensa* tramite un approccio esplorativo del tipo trial-and-error o una pianificazione deliberativa.
Un esempio classico di apprendimento di rafforzamento è il motore del gioco degli scacchi. Qui, l'agente decide come svolgere una serie di mosse a seconda dello stato della

scacchiera (l'ambiente) e la ricompensa può essere definita come la *vittoria* o la *sconfitta* alla fine del gioco (Figura 1.5).

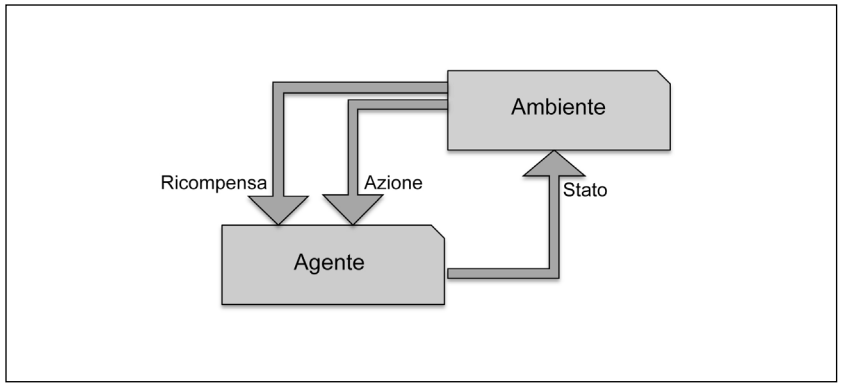


Figura 1.5

Scoprire le strutture nascoste con l'apprendimento senza supervisione

Nell'apprendimento con supervisione, conosciamo in anticipo la *risposta corretta* quando descriviamo il nostro modello, mentre nell'apprendimento di rafforzamento definiamo una misura, o *ricompensa*, per le specifiche azioni messe in atto dall'agente. Nell'apprendimento senza supervisione, al contrario, abbiamo a che fare con dati non etichettati o dati dalla *struttura ignota*. Utilizzando tecniche di apprendimento senza supervisione, siamo in grado di osservare la struttura dei nostri dati, per estrarre da essi informazioni cariche di significato senza però poter contare sulla guida né di una variabile nota relativa al risultato, né una funzione di ricompensa.

Ricerca di sottogruppi tramite il clustering

Il *clustering* è una tecnica esplorativa di analisi dei dati che ci consente di organizzare una serie di informazioni all'interno di gruppi significativi (i *cluster*) senza avere alcuna precedente conoscenza delle appartenenze a tali gruppi. Ogni cluster che può essere derivato durante l'analisi definisce un gruppo di oggetti che condividono un certo grado di similarità, ma che sono più dissimili rispetto agli oggetti presenti negli altri cluster, motivo per cui il clustering viene talvolta chiamato "classificazione senza supervisione". Il clustering è un'ottima tecnica per la strutturazione dell'informazione e per individuare relazioni significative nei dati. Per esempio, consente agli operatori di marketing di individuare dei gruppi di clienti sulla base dei loro interessi, in modo da sviluppare specifici programmi di marketing.
La Figura 1.6 illustra il modo in cui il clustering può essere applicato all'organizzazione di alcuni dati senza etichetta suddividendoli in tre gruppi distinti sulla base della similarità delle caratteristiche x_1 e x_2.

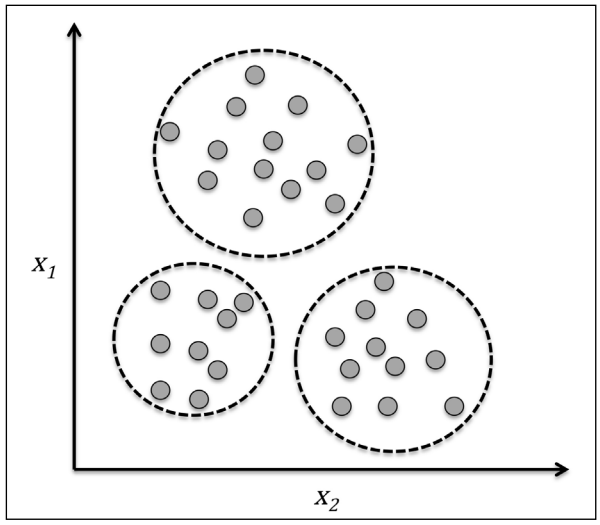


Figura 1.6

Riduzione dimensionale per la compressione dei dati

Un altro sottocampo dell'apprendimento senza supervisione è la *riduzione dimensionale*. Spesso ci troviamo a operare con dati a elevata dimensionalità (ogni osservazione fornisce un elevato numero di misure) il che può rappresentare una sfida in termini di spazio di memorizzazione disponibile e prestazioni computazionali degli algoritmi di apprendimento automatico. La riduzione dimensionale senza supervisione è un approccio comunemente utilizzato nella pre-elaborazione delle caratteristiche, e ha lo scopo di eliminare dai dati il "rumore", che può anche introdurre un degrado delle prestazioni predittive di alcuni algoritmi, e di comprimere i dati in un sottospazio dimensionale più compatto, mantenendo però la maggior parte delle informazioni rilevanti.
Talvolta la riduzione dimensionale può essere utile anche per rappresentare i dati: per esempio, un insieme di caratteristiche a elevata dimensionalità può essere proiettato su uno spazio di caratteristiche mono-, bi- o tri-dimensionale, in modo da visualizzarlo tramite diagrammi o programmi 3D o 2D. La Figura 1.7 mostra un esempio in cui la riduzione della dimensionalità non lineare è stata applicata in modo da comprimere un grafico 3D *Swiss Roll* in un nuovo sottospazio bidimensionale delle caratteristiche.

Introduzione alla terminologia e alla notazione di base

Ora che abbiamo introdotto le tre grandi categorie dell'apprendimento automatico (con supervisione, senza supervisione e di rafforzamento), diamo un'occhiata alla terminologia di base che ci troveremo a utilizzare nel corso dei prossimi capitoli. La seguente tabella rappresenta un estratto del dataset *Iris*, che è un classico esempio nel campo dell'apprendimento automatico. Il dataset Iris contiene la misurazione di centocinquanta specie di fiori iris di tre diverse specie: *Setosa*, *Versicolor* e *Viriginica*. Qui, ogni campione di fiore rappresenta una riga del dataset e la misurazione del fiore in centimetri viene indicata in colonne, che rappresentano le caratteristiche del dataset (Figura 1.8).

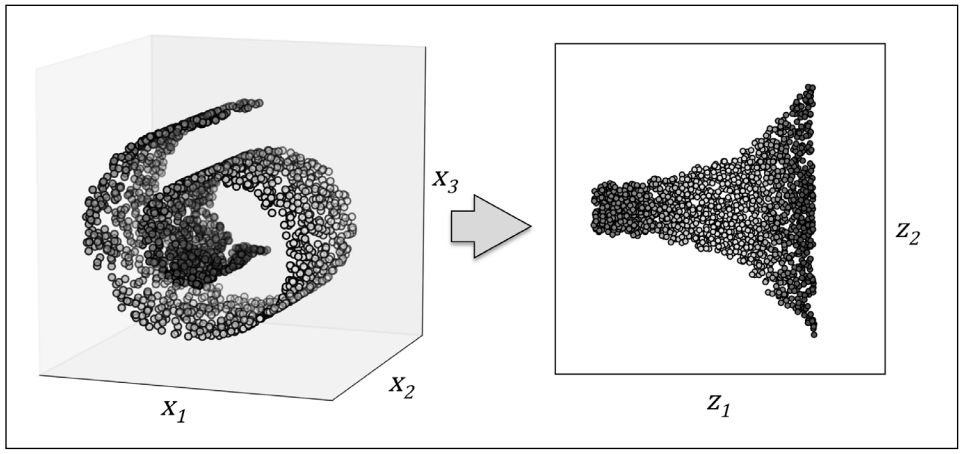
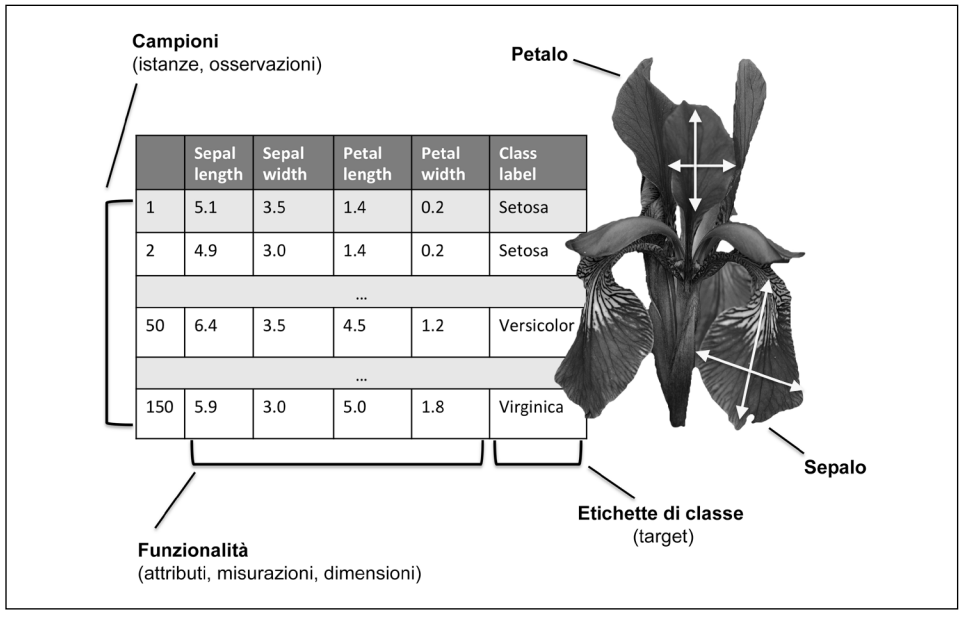


Figura 1.7



	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
...					
50	6.4	3.5	4.5	1.2	Versicolor
...					
150	5.9	3.0	5.0	1.8	Virginica

Figura 1.8

Per garantire la semplicità e l'efficienza della notazione e dell'implementazione, faremo uso di alcuni elementi di base dell'*algebra lineare*. Nei prossimi capitoli, utilizzeremo una notazione a *matrici* e *vettori* per far riferimento ai dati. Seguiremo la convenzione comune di rappresentare ciascun campione come una riga distinta nella matrice delle caratteristiche, *X*, dove ciascuna caratteristica è conservata come una colonna distinta. Il dataset Iris, costituito da centocinquanta campioni e quattro caratteristiche, può pertanto essere scritto come una matrice 150×4 $\boldsymbol{X} \in \mathbb{R}^{150\times4}$:

$$\begin{bmatrix} x_1^{(1)} & x_2^{(1)} & x_3^{(1)} & x_4^{(1)} \\ x_1^{(2)} & x_2^{(2)} & x_3^{(2)} & x_4^{(2)} \\ \vdots & \vdots & \vdots & \vdots \\ x_1^{(150)} & x_2^{(150)} & x_3^{(150)} & x_4^{(150)} \end{bmatrix}$$

Approfondimento

Per la parte rimanente del libro, utilizzeremo il numero in apice (*i*) per fare riferimento all'i-esimo campione di addestramento e il numero in pedice (*j*) per far riferimento alla j-esima dimensione del dataset di apprendimento.

Utilizzeremo lettere minuscole in grassetto per far riferimento ai vettori $\left(\boldsymbol{x} \in \mathbb{R}^{n\times 1}\right)$ e lettere maiuscole in grassetto per far riferimento alle matrici $(\boldsymbol{X} \in \mathbb{R}^{n\times m})$. Per far riferimento ai singoli elementi di un vettore o di una matrice, utilizzeremo lettere in corsivo ($x^{(n)}$ o $x_{(m)}^{(n)}$).

Per esempio, x_1^{150} fa riferimento alla prima dimensione del centocinquantesimo campione di fiori, la *larghezza del sepalo*. Pertanto, ciascuna riga di questa matrice delle caratteristiche rappresenta l'istanza di un fiore e può essere scritta come un vettore a colonna a 4 dimensioni $\boldsymbol{x}^{(i)} \in \mathbb{R}^{1\times 4}$, $\boldsymbol{x}^{(i)} = \begin{bmatrix} x_1^{(i)} & x_2^{(i)} & x_3^{(i)} & x_4^{(i)} \end{bmatrix}$.

Ogni dimensioni relativa a una caratteristica è invece un vettore a riga a 150 dimensioni $\boldsymbol{x}^{(i)} \in \mathbb{R}^{150\times 1}$, per esempio:

$$\boldsymbol{x}_j = \begin{bmatrix} x_j^{(1)} \\ x_j^{(2)} \\ \vdots \\ x_j^{(150)} \end{bmatrix}$$

Analogamente, memorizzeremo le variabili target (in questo caso le etichette delle classi) come un vettore a colonna a 150 dimensioni

$$\boldsymbol{y} = \begin{bmatrix} y^{(1)} \\ \dots \\ y^{(150)} \end{bmatrix} \left(y \in \{\text{Setosa, Versicolor, Virginica}\}\right)$$

Una roadmap per la realizzazione di sistemi di apprendimento automatico

Nei paragrafi precedenti abbiamo trattato i concetti di base dell'apprendimento automatico e i tre diversi tipi di apprendimento. In questo paragrafo ci concentreremo su altri elementi importanti di un sistema di apprendimento automatico che contraddistinguono l'algoritmo di apprendimento. Lo schema rappresentato nella Figura 1.9 rappresenta un

tipico diagramma di flusso per l'impiego dell'apprendimento automatico nella *modellazione predittiva*, di cui parleremo nei prossimi paragrafi.

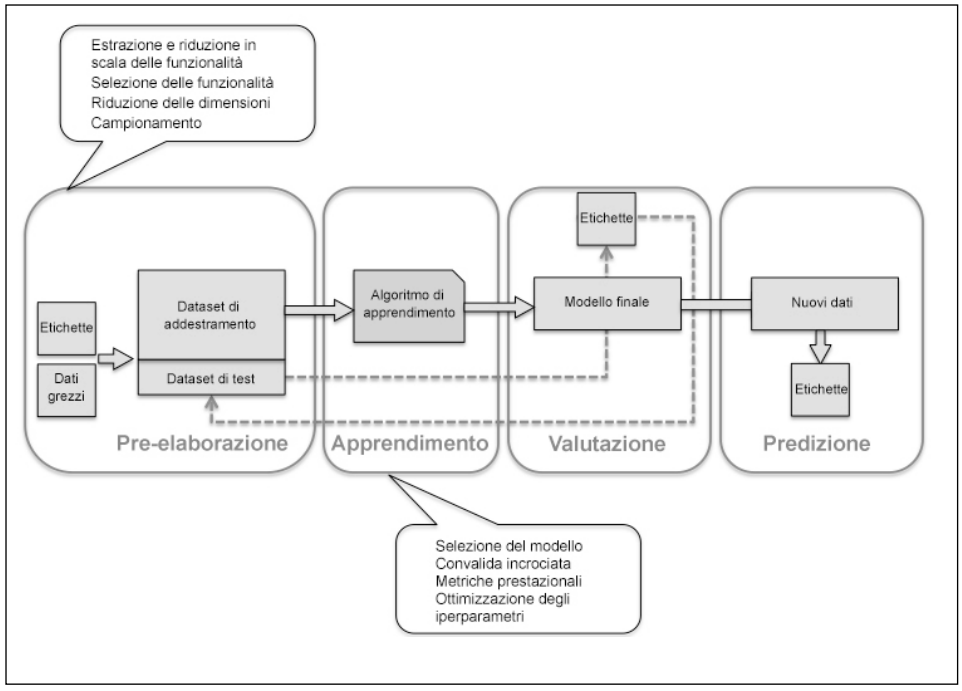


Figura 1.9

Pre-elaborazione: dare una "forma" ai dati

È raro che i dati abbiano una forma e un aspetto atti a garantire prestazioni ottimali dell'algoritmo di apprendimento. Pertanto, la *pre-elaborazione* dei dati è uno dei passi cruciali di qualsiasi applicazione di apprendimento automatico. Se prendiamo ad esempio il dataset del fiore iris del paragrafo precedente, possiamo considerare i dati grezzi come una serie di fotografie del fiore, dalle quali possiamo estrarre caratteristiche significative. Le caratteristiche significative possono essere il colore, la tonalità, l'intensità dei toni, l'altezza, la larghezza e la profondità. Molti algoritmi di apprendimento automatico richiedono anche che, per ottenere le massime prestazioni, le caratteristiche scelte adottino la stessa scala, il che spesso viene ottenuto trasformando le caratteristiche in un intervallo [0, 1] oppure in una distribuzione normale standard con media 0 e varianza 1, come vedremo nei prossimi capitoli.

Alcune delle caratteristiche scelte possono avere un'elevata correlazione e, pertanto, risultare in qualche modo ridondanti. In tali casi, le tecniche di riduzione delle dimensioni sono utili per comprimere le caratteristiche in un sottospazio dimensionale inferiore. La riduzione della dimensionalità dello spazio delle caratteristiche presenta anche il vantaggio di richiedere meno spazio di memorizzazione e di accelerare il funzionamento dell'algoritmo di apprendimento.

Per determinare se il nostro algoritmo di apprendimento automatico non solo si comporta bene sul set di addestramento, ma esegue generalizzazioni corrette sui nuovi dati, potremmo voler suddividere in modo casuale il dataset in due set distinti: di addestramento e di test. Utilizziamo il set di addestramento per informare e ottimizzare il modello di apprendimento automatico, mentre teniamo da parte fino all'ultimo il set di test, per valutare il modello finale.

Addestramento e selezione di un modello predittivo

Come vedremo nei prossimi capitoli, sono stati sviluppati vari algoritmi di apprendimento automatico con lo scopo di risolvere problemi differenti. Un elemento importante che può essere tratto dal noto *No Free Lunch Theorems* di David Wolpert è che non esiste alcun apprendimento "gratuito" (*The Lack of A Priori Distinctions Between Learning Algorithms*, D.H. Wolpert 1996; *No Free Lunch Theorems for Optimization*, D.H. Wolpert e W.G. Macready, 1997). Intuitivamente, possiamo correlare questo concetto con il noto detto "Suppongo che sia allettante, quando l'unico strumento che hai a disposizione è un martello, trattare tutto come se fosse un chiodo" (Abraham Maslow, 1966). Per esempio, ogni algoritmo di classificazione ha i suoi difetti intrinseci e nessun modello di classificazione può vantare una superiorità assoluta se non abbiamo alcuna informazione sul compito da svolgere. In pratica, è pertanto essenziale confrontare almeno un certo gruppo di algoritmi differenti, in modo da addestrarli e selezionare poi il modello che offre le migliori prestazioni. Ma prima di poter confrontare modelli differenti, dobbiamo decidere le metriche da impiegare per misurarne le prestazioni. Una metrica comunemente utilizzata è l'accuratezza della classificazione, che è definita come la proporzione tra le istanze classificate correttamente.
Sorge una domanda legittima: *come possiamo capire quale modello si comporta meglio sul dataset di test finale e sui dati reali se non utilizziamo questo dataset di test per la scelta del modello ma lo conserviamo per la valutazione finale del modello stesso?* Per poter risolvere il problema insito in questa domanda, possono essere utilizzate varie tecniche di convalida incrociata, nelle quali il dataset di addestramento viene ulteriormente suddiviso in *sottoinsiemi di addestramento di convalida*, in modo da stimare le *prestazioni di generalizzazione* del modello. Infine, non possiamo neppure aspettarci che i parametri standard dei vari algoritmi di apprendimento forniti dalle librerie di software siano ottimali per il nostro specifico problema. Pertanto, faremo frequentemente uso di *tecniche di ottimizzazione* degli iperparametri, che ci aiuteranno a ottimizzare le prestazioni del modello, come vedremo negli ultimi capitoli. Intuitivamente, possiamo considerare questi iperparametri come parametri che non vengono appresi dai dati, ma rappresentano le "manopole" del modello, sulle quali possiamo intervenire per migliorarne le prestazioni, come vedremo con maggiore chiarezza nei prossimi capitoli, quando le metteremo all'opera su esempi effettivi.

Valutazione dei modelli e previsione su istanze di dati mai viste prima

Dopo aver scelto un modello che possa adattarsi al dataset di addestramento, possiamo utilizzare il dataset di test per stimare la qualità della sua azione su dati mai visti prima, in modo da stimare l'errore di generalizzazione. Se siamo soddisfatti delle sue prestazioni, possiamo utilizzare questo modello anche per prevedere nuovi dati, futuri. È importante

notare che i parametri delle procedure di cui abbiamo appena parlato (riduzione della scala e delle dimensioni delle caratteristiche) si possono ottenere solo dal dataset di addestramento e che questi stessi parametri vengono poi riapplicati per trasformare il dataset di test e anche ogni nuovo campione dei dati. Le prestazioni misurate sui dati di test, altrimenti, potrebbero essere eccessivamente ottimistiche.

Usare Python per attività di machine learning

Python è uno dei più noti linguaggi di programmazione per l'elaborazione dei dati e pertanto gode di una grande quantità di utili librerie aggiuntive, sviluppate dalla sua ottima comunità.
Sebbene le prestazioni dei linguaggi interpretati, come Python, per compiti intensivi dal punto di vista computazionale siano inferiori rispetto a quelle dei linguaggi di programmazione a basso livello, alcune librerie di estensione, come *NumPy* e *SciPy*, sono state sviluppate proprio basandosi su implementazioni a basso livello, in Fortran e C, in modo da accelerare le operazioni sui vettori, che devono operare su array multidimensionali.
Per attività di programmazione nel campo del machine learning, faremo riferimento principalmente alla libreria *scikit-learn*, che, attualmente, è una delle librerie open source di machine learning più note e diffuse.

Installazione dei pacchetti Python

Python è disponibile per i tre principali sistemi operativi, Microsoft Windows, Mac OS X e Linux, e il suo installer, come la documentazione, può essere scaricato dal sito web ufficiale di Python: `https://www.python.org`.
Questo libro è stato scritto per le versioni di Python successive alla 3.4.3 e si consiglia vivamente di scaricare sempre la versione più aggiornata di Python 3, sebbene la maggior parte degli esempi di codice possa essere compatibile anche con le versioni di Python successive alla 2.7.10. Se decidete di utilizzare Python 2.7 per eseguire gli esempi di codice, assicuratevi di considerare le grandi differenze che esistono fra le due versioni di Python. Un buon riepilogo delle differenze fra Python 3.4 e 2.7 si trova all'indirizzo `https://wiki.python.org/moin/Python2orPython3`.
Gli altri pacchetti che ci troveremo a utilizzare nel corso del libro possono essere installati tramite il programma installer *pip*, che fa parte della libreria standard di Python fin da Python 3.3. Ulteriori informazioni su `pip` si trovano all'indirizzo `https://docs.python.org/3/installing/index.html`.
Dopo aver scaricato e installato Python, possiamo eseguire `pip` dalla riga di comando, in modo da installare ulteriori pacchetti Python:

```
pip install SomePackage
```

I pacchetti già installati possono invece essere aggiornati tramite il flag `--upgrade`:

```
pip install SomePackage --upgrade
```

Una distribuzione Python alternativa, altamente consigliata per il calcolo scientifico, è Anaconda di Continuum Analytics. Anaconda è una distribuzione Python di livello professionale completamente gratuita (includendo anche gli utilizzi commerciali) che offre tutti i pacchetti essenziali di Python per attività scientifiche, matematiche e ingegneristiche, il tutto incluso in un'unica comoda distribuzione multipiattaforma. L'installer di Anaconda può essere scaricato all'indirizzo `http://continuum.io/downloads#py34`, mentre una guida rapida all'utilizzo di Anaconda è disponibile all'indirizzo `https://store.continuum.io/static/img/Anaconda-Quickstart.pdf`.
Dopo aver installato Anaconda, possiamo installare i nuovi pacchetti Python utilizzando il seguente comando:

```
conda install SomePackage
```

I pacchetti preesistenti possono essere aggiornati utilizzando il seguente comando:

```
conda update SomePackage
```

Nel corso del libro, utilizzeremo principalmente gli array multidimensionali *NumPy* per conservare e manipolare i dati. Occasionalmente, faremo uso di *pandas*, una libreria basata su NumPy che fornisce ulteriori strumenti per la manipolazione ad alto livello dei dati, che semplificano ancora di più l'elaborazione di dati di tipo tabulare. Per migliorare l'esperienza di apprendimento e consentire una visualizzazione dei dati quantitativi, aspetto che spesso è estremamente utile per comprendere, intuitivamente, il senso delle operazioni, utilizzeremo la libreria *matplotlib*, che presenta il vantaggio di essere molto personalizzabile.
I numeri di versione dei principali pacchetti Python che sono stati utilizzati per scrivere questo libro sono i seguenti. Assicuratevi che i numeri di versione dei pacchetti che avete installato siano uguali o successivi rispetto a questi numeri di versione, in modo da garantire che gli esempi di codice funzionino correttamente:

- NumPy 1.9.1
- SciPy 0.14.0
- scikit-learn 0.15.2
- matplotlib 1.4.0
- pandas 0.15.2

Riepilogo

In questo capitolo abbiamo esplorato le tecniche di machine learning (apprendimento automatico) da un punto di vista davvero molto elevato, con il solo scopo di familiarizzare con l'argomento e con i principali concetti che andremo a esplorare più in dettaglio nel corso dei prossimi capitoli.
Come abbiamo visto, l'apprendimento con supervisione è costituito da due importanti sottocampi: la classificazione e la regressione. Mentre i modelli di classificazione ci consentono di catalogare gli oggetti in classi note, possiamo utilizzare l'analisi di regressione per prevedere i risultati continui delle variabili target. L'apprendimento senza

supervisione non solo ci offre tecniche utili per individuare le strutture all'interno di dati non etichettati, ma può essere utile anche per la compressione dei dati nei passi di pre-elaborazione delle caratteristiche.
Abbiamo proseguito con il tipico percorso di applicazione delle tecniche di machine learning alla soluzione dei problemi, che utilizzeremo come base per discussioni più approfondite e per esempi pratici, nel corso dei prossimi capitoli. Alla fine abbiamo configurato l'ambiente Python e abbiamo installato e aggiornato i pacchetti richiesti, in modo da prepararci a vedere in azione le attività di machine learning.
Nel prossimo capitolo, implementeremo uno dei primi algoritmi di classificazione ad apprendimento automatico, il che ci preparerà per il Capitolo 3, *I classificatori di machine learning di scikit-learn*, dove tratteremo algoritmi di apprendimento automatizzato più avanzati, che fanno uso della libreria open source di machine learning scikit-learn. Poiché gli algoritmi di machine learning apprendono dai dati, è fondamentale fornire loro informazioni utili e nel Capitolo 4, *Costruire buoni set di addestramento: la pre-elaborazione*, ci occuperemo delle principali tecniche di pre-elaborazione. Nel Capitolo 5, *Compressione dei dati tramite la riduzione della dimensionalità*, affronteremo l'argomento delle tecniche di riduzione della dimensionalità, che possono aiutarci a comprimere il dataset in un sottospazio delle caratteristiche dotato di un numero inferiore di dimensioni, il che può essere utile per migliorarne l'efficienza computazionale. Un aspetto importante della costruzione di modelli di machine learning consiste nella valutazione delle loro prestazioni e nella stima della qualità delle loro previsioni su dati nuovi, imprevisti. Nel Capitolo 6, *Valutazione dei modelli e ottimizzazione degli iperparametri*, apprenderemo tutte le migliori tecniche di ottimizzazione e valutazione dei modelli. In alcune situazioni potremmo non essere ancora soddisfatti delle prestazioni del nostro modello predittivo, sebbene abbiamo dedicato ore o magari giorni a ottimizzarlo e collaudarlo. Nel Capitolo 7, *Combinare più modelli: l'apprendimento d'insieme*, impareremo a combinare più modelli di machine learning, per realizzare sistemi predittivi ancora più potenti.
Dopo aver trattato tutti i concetti più importanti di una tipica catena di machine learning, implementeremo un modello per la previsione delle emozioni nel testo nel Capitolo 8, *Tecniche di machine learning per l'analisi del sentiment*, e poi, nel Capitolo 9, *Embedding di un modello in un'applicazione web*, includeremo il tutto in un'applicazione web aperta ai visitatori. Nel Capitolo 10, *Previsioni di variabili target continue: l'analisi a regressione* utilizzeremo gli algoritmi di machine learning per l'analisi della regressione, che ci consentirà di prevedere delle variabili di output continue, infine nel Capitolo 11, *Lavorare con dati senza etichette: l'analisi a cluster*, applicheremo gli algoritmi di clustering, che ci consentiranno di trovare quelle strutture che si nascondono all'interno dei dati. L'ultimo capitolo di questo libro affronterà le reti neurali artificiali, che ci consentiranno di affrontare problemi complessi, come il riconoscimento delle immagini e del parlato, uno degli argomenti attualmente più interessanti nella ricerca nell'ambito del machine learning.