Algoritmi per l'intelligenza artificiale

Simone Colli
authoremail

Appunti del corso tenuto dal **Prof. Vincenzo Bonnici**

Università degli Studi di Parma
Anno Accademico 2025/2026

Indice

1 Introduzione

Nel campo dell'apprendimento automatico classico, le attività sono tradizionalmente suddivise in quattro rami principali:

- Apprendimento supervisionato (supervised).
- Apprendimento semi-supervisionato (semi-supervised).
- Apprendimento non supervisionato (unsupervised).
- Apprendimento per rinforzo (reinforcement learning).

La distinzione primaria tra queste metodologie di ML risiede nel livello di disponibilità dei "dati di verità di base" (ground truth). Il **ground truth** è definito come la conoscenza preliminare dell'output che il modello dovrebbe produrre per un dato input, basata sull'osservazione diretta in contrapposizione all'inferenza.

1.1 Apprendimento automatico supervisionato

L'apprendimento automatico supervisionato ha come obiettivo l'apprendimento di una funzione che, dato un **campione di dati** e i relativi output desiderati, riesca ad approssimare la funzione sottostante che mappa gli input agli output.

Questa metodologia è comunemente applicata in due principali contesti:

- **Classificazione**: quando si desidera mappare l'input a etichette di output discrete.
- **Regressione**: quando l'obiettivo è mappare l'input a un output continuo.

In entrambi i casi, lo scopo è identificare relazioni o strutture specifiche nei dati di input che consentano di generare output corretti in modo efficace. È fondamentale notare che la correttezza dell'output è determinata interamente dai dati di addestramento, i quali costituiscono la "verità di base" che il modello apprende.

Tuttavia, l'efficacia del modello può essere significativamente ridotta dalla presenza di etichette "rumorose" o "errate" all'interno dei dati stessi. Algoritmi notevoli nell'apprendimento supervisionato includono la regressione logistica, il classificatore bayesiano naif, le macchine a vettori di supporto, le reti neurali artificiali e le foreste casuali.

Il successo di un modello di ML dipende dalla sua capacità di generalizzazione. Questo concetto è strettamente connesso alla complessità del modello, che si riferisce alla complessità della funzione che si sta cercando di apprendere. Se si dispone di una quantità limitata di dati o se questi non sono distribuiti uniformemente, è cruciale optare per un modello a bassa complessità per evitare situazioni di **overfitting** (sovradattamento). L'overfitting si verifica quando il modello apprende la funzione adattandosi troppo bene ai soli dati di addestramento, senza cogliere la tendenza o la struttura effettiva che guida l'output, e quindi non riesce a generalizzare a nuovi punti dati.

La gestione della generalizzazione è formalizzata tramite il compromesso **bias-varianza** (bias-variance tradeoff). Così facendo il modello presenterà un equilibrio tra:

- **Bias** (distorsione): l'errore sistematico dovuto a ipotesi errate nel processo di apprendimento.
- **Varianza**: la quantità in base alla quale l'errore può variare tra diversi set di dati.

La difficoltà si presenta nel creare un modello che cattura accuratamente le regolarità dei dati di addestramento e che sia in grado di generalizzare bene a dati non visti in precedenza.

Generalmente, un aumento del bias (e una conseguente riduzione della varianza) porta a modelli con livelli di prestazione più stabili e garantiti, un fattore che può essere cruciale in certe applicazioni. Per ottenere una buona generalizzazione, la varianza del modello deve essere attentamente bilanciata in base alla dimensione e alla complessità dei dati di addestramento. Nello specifico, set di dati piccoli e semplici dovrebbero essere gestiti con modelli a bassa varianza, mentre set di dati grandi e complessi richiedono modelli con una varianza più elevata per poter catturare appieno la struttura sottostante dei dati.

1.2 Apprendimento automatico semi-supervisionato

L'apprendimento automatico semi-supervisionato (semi-supervised) mira a **etichettare i punti dati senza etichetta**. Per fare ciò, utilizza le conoscenze apprese da un piccolo numero di dati già etichettati.

Questa tecnica è utile in scenari dove ottenere dati etichettati è costoso o complesso. Ad esempio, nel rilevamento di messaggi inappropriati in un social network, è impraticabile etichettare manualmente ogni messaggio. Si può, invece, etichettare manualmente un piccolo sottoinsieme e usare tecniche semi-supervisionate per comprendere e classificare il resto dei contenuti.

Metodi comuni includono le **macchine vettoriali di supporto trasversali** e i **metodi basati su grafi** (come la propagazione delle etichette).

1.2.1 Presupposti dell'apprendimento semi-supervisionato

Per poter giustificare l'uso di pochi dati etichettati per trarre conclusioni su un grande insieme di dati non etichettati, i metodi semi-supervisionati si basano su alcuni presupposti fondamentali:

- **Continuità**: Si assume che punti dati "vicini" tra loro abbiano maggiori probabilità di condividere la stessa etichetta.
- **Ipotesi del cluster**: Si presume che i dati formino naturalmente dei cluster discreti. Di conseguenza, punti nello stesso cluster hanno maggiori probabilità di condividere un'etichetta.
- **Presupposto molteplice (manifold)**: Si ipotizza che i dati si trovino approssimativamente in uno spazio di dimensioni inferiori (un *manifold*) rispetto allo spazio di input originale. Questo è rilevante quando un sistema con pochi parametri, non osservabile direttamente, produce output osservabili ad alta dimensione.

1.3 Apprendimento automatico non supervisionato

L'apprendimento automatico non supervisionato (unsupervised) opera **senza output etichettati**. Il suo obiettivo principale è quindi quello di **dedurre la struttura naturale** presente all'interno di un insieme di dati.

Questi metodi cercano di trovare modelli (pattern) intrinseci nei dati. Le attività più comuni in questo ambito sono:

- Il **clustering** (raggruppamento).
- L'apprendimento della **rappresentazione** (representation learning).

- La **stima della densità** (density estimation).

In tutti questi casi, si desidera comprendere la struttura intrinseca dei dati senza usare etichette fornite esplicitamente.

Algoritmi comuni includono il **clustering**, l'analisi dei componenti principali (**PCA**) e gli **auto-codificatori** (autoencoders).

Dato che non vengono fornite etichette, nella maggior parte dei metodi di apprendimento non supervisionato non esiste un modo specifico per confrontare le prestazioni del modello.

Le due tecniche principali per affrontare problemi di apprendimento non supervisionato sono il clustering e la riduzione della dimensionalità dei dati.

1.3.1 Clustering

Il clustering è una **tecnica esplorativa** che permette di aggregare dati in gruppi (detti *cluster*) senza avere una precedente conoscenza della loro appartenenza a tali gruppi. Si applica a dataset dove i dati al loro interno presentano elementi simili tra loro. All'interno di ogni singolo cluster si troveranno quindi dati che hanno molte **caratteristiche simili** tra loro. È un'ottima tecnica per trovare relazioni tra i dati.

1.3.2 Riduzione della dimensionalità

La riduzione della dimensionalità senza supervisione è un approccio molto usato nella **pre-elaborazione delle features**. L'obiettivo principale di questa tecnica è di **eliminare il "rumore"** dai dati.

Questa operazione può talvolta causare una minore prestazione predittiva. Tuttavia, può anche rendere lo spazio dimensionale più compatto, aiutando a **mantenere le informazioni più rilevanti**. Inoltre, è molto utile per la **rappresentazione dei dati**: dati in uno spazio delle caratteristiche ad elevata dimensionalità possono essere proiettati su uno spazio 1D, 2D o 3D per l'analisi visiva.

1.3.3 Analisi esplorativa

L'apprendimento non supervisionato è estremamente utile nell'**analisi esplorativa dei dati** (exploratory data analysis), poiché è in grado di **identificare automaticamente la struttura** nei dati. Ad esempio, se un analista volesse segmentare i consumatori, i metodi di clustering sarebbero un ottimo punto di partenza per l'analisi.

In situazioni dove è impraticabile o impossibile per un essere umano proporre tendenze nei dati, l'apprendimento non supervisionato può fornire **informazioni iniziali** che possono poi essere usate per testare o verificare singole ipotesi.

1.4 Apprendimento per rinforzo

L'apprendimento con rinforzo (reinforcement learning) ha l'obiettivo di realizzare **agenti autonomi**. Questi agenti devono essere in grado di scegliere azioni da compiere per conseguire determinati obiettivi. Questo avviene tramite l'interazione con l'ambiente in cui sono immersi, con lo scopo di massimizzare una nozione di **premio cumulativo**.

2 Classificazione

La classificazione è un'attività dell'apprendimento supervisionato che consiste nell'assegnare un'etichetta (o classe) a un dato sulla base di sue caratteristiche osservabili.

Nell'ambito della classificazione si parla di:

- **Feature** (caratteristiche): un aspetto direttamente osservabile di un fenomeno per il quale si può registrare una misura, che sia quantitativa (numerica) o categoriale (come vero/falso, rosso/verde, ecc.).
- **Classe**: un concetto astratto e generale che "spiega" le osservazioni. L'assegnazione a una classe costituisce una sintesi delle feature osservate.
- **Label** (o etichetta): il nome specifico di una classe.

Tuttavia, alcuni dati possono rendere più complesso l'assegnazione delle classi; questi esempi sono tecnicamente noti come **outlier statistici**.

Definizione 2.1: Classificazione

ata una **collezione di dati**, definita come un insieme P di M-uple del tipo:

$$m_i = (x_{1i}, \ldots, x_{Mi}) \in D_1 \times \ldots \times D_M$$

dove ogni x_{ji} rappresenta una feature ed appartiene ad un possibile dominio di valori D_j.
L'insieme P è partizionato in k classi, le cui etichette compongono l'insieme $L = (A_1, \ldots, A_k)$.
Un **algoritmo di classificazione** è una funzione compputabile $f : P \mapsto L$, tale che:

$$f(m \in P) = f(x_1, \ldots, x_m) \in L$$

Tale funzione $f(m)$ assegna a ogni dato m un'etichetta A_i scelta tra quelle presenti in L cercando di stimare l'etichetta reale del stesso.
Lo schema di classificazione può produrre due tipi di risultati:

- **Successo** (hit) se l'etichetta stimata $f(m)$ coincide con l'etichetta reale del dato.
- **Fallimento** (miss) se l'etichetta stimata è errata.

È generalmente impossibile creare classificatori *error free*. È quindi fondamentale fornire stime sul tasso percentuale di hit/miss che lo schema può ottenere. Il livello tollerabile di errore dipende dalla **criticità dell'applicazione**: per applicazioni industriali si può richiedere un tasso $< 5\%$, mentre per applicazioni mediche un tasso $> 0.5\%$ potrebbe essere già inaccettabile.

Esempio 2.1: Problema di classificazione: salmoni e branzino

Si consideri il problema di distinguere tra salmoni e branzini (sea bass) basandosi su alcune caratteristiche osservabili. Le **feature** utilizzate potrebbero essere la lunghezza, il peso in grammi e il colore dominante (un attributo qualitativo scelto da un insieme predefinito come {blu, grigio, verde}). I dati vengono tipicamente organizzati in una tabella, dove ogni riga corrisponde a un pesce e le colonne ne descrivono le feature.

L'obiettivo è costruire un classificatore che, per ogni nuovo pesce osservato, sia in grado di riempire la colonna "specie" con l'etichetta corretta ("salmone" o "branzino"). È importante notare che gli errori non hanno lo stesso costo: confondere un salmone (pesce pregiato) con un branzino (meno pregiato) è un errore più grave del contrario.

Esempio 2.2: Problema di classificazione: studenti e carriera

Si consideri il problema di predire il futuro successo economico degli studenti universitari. Le **feature** raccolte per ogni studente includono dati anagrafici, il censo familiare e i voti conseguiti durante la carriera universitaria. L'obiettivo è costruire un classificatore che predica in quale **classe** di reddito si troverà lo studente dieci anni dopo la laurea. Le etichette (o **label**) potrebbero essere {"reddito basso", "reddito medio", "reddito alto"}.
È importante notare che, a causa dell'elevato numero di fattori non misurabili che influenzano la vita di un individuo, una predizione del genere ha un valore limitato se applicata al singolo studente, che ha un'alta probabilità di essere classificato erroneamente.
Tuttavia, questo tipo di analisi è estremamente utile a livello statistico e aggregato, per comprendere le tendenze generali di un'intera popolazione studentesca e informare politiche educative o economiche.

2.1 Costruire un classificatore

Il processo di costruzione di un classificatore automatico simula il fenomeno dell'apprendimento umano o animale, noto come **training** (addestramento). L'idea è **dedurre regole generali**, applicabili a record non ancora classificati, partendo dall'osservazione di esempi già noti e ben classificati.

Si definisce **universo delle osservazioni** l'insieme complessivo dei record (passati, presenti e futuri) relativi ad un fenomeno. Molti algoritmi iniziano esaminando un sottoinsieme di questo universo, già classificato e ben compreso.

Questo insieme di "allenamento", chiamato **Training Set (TS)**, è il deposito di informazioni iniziali da cui l'algoritmo ricava le "regole" di classificazione. Le regole ricavate saranno di vario tipo: statistiche, probabilistiche, fuzzy, funzioni discendenti, ecc.

2.2 Proprietà di un classificatore

Un buon insieme di regole di classificazione deve avere tre importanti proprietà:

- **Semplicità**: Le regole non devono essere troppo complicate, per garantire efficienza e basso costo computazionale in fase di classificazione.
- **Correttezza sul TS**: Le regole devono essere statisticamente sufficientemente corrette quando applicate al medesimo Training Set che le ha generate.
- **Generalizzabilità**: Le regole devono essere statisticamente corrette anche quando applicate al resto dei record dell'universo (dati nuovi, non visti).

Statisticamente corretto è un termine che indica che il tasso dei miss non deve superare certe soglie di tolleranza che dipendono dalla criticità delle applicazioni.

2.3 Il problema dell'overfitting

Le proprietà di correttezza sul TS e di generalizzabilità sono spesso in conflitto tra loro. Questo paradosso è noto come **overfitting** (sovradattamento).

L'overfitting si verifica quando un modello si adatta "troppo bene" ai dati del Training Set. Un modello molto complesso può imparare a memoria le peculiarità e persino il rumore casuale presente nel TS, ottenendo una correttezza perfetta su di esso. Tuttavia, tale modello non avrà appreso la "tendenza" generale dei dati e fallirà nel generalizzare a nuovi record, poiché la frontiera di decisione che ha appreso è eccessivamente complessa e specifica per il campione di training.

L'obiettivo non è quindi minimizzare l'errore sul TS (che porterebbe a un modello complesso e in overfitting), ma trovare un equilibrio: un modello (es. una retta o una curva semplice) che, pur commettendo qualche errore sul TS, catturi la struttura di fondo dei dati e possa quindi generalizzare meglio.

2.4 Validazione

Per "convalidare" la proprietà di generalizzazione di un insieme di regole, si utilizza un metodo che prevede, oltre al TS, un altro insieme di record già etichettati, detto **Control Set (CS)** o **Test Set**.

Il CS **non** viene utilizzato durante la fase di training (cioè per la sintesi delle regole). Viene usato solo dopo che le regole sono state definite. Se le regole mostrano sul CS un tasso di errore (miss) simile a quello ottenuto sul TS, allora si ritiene che le regole siano **generalizzabili**.

Poiché anche il CS è un campione casuale, per una stima più precisa è buona norma ripetere i test con diversi CS, spesso creati tramite strategie di randomizzazione nella selezione del TS e del CS dall'universo disponibile.

2.5 Gestione delle feature e del rumore

Nella costruzione di un classificatore è cruciale gestire sia la selezione delle feature che la presenza di rumore.

2.5.1 Selezione delle feature

Spesso si rilevano molte feature, ma non tutte sono utili; alcune possono essere sovrabbondanti o addirittura dannose. Combinare più feature (es. lunghezza e luminosità dei pesci) è spesso una strategia conveniente, ma non è detto che sia sempre la migliore. L'inclusione di troppe feature, specialmente se irrilevanti, può amplificare il "rumore" e confondere il classificatore.

Una buona pratica è scegliere feature che siano **invarianti** alle trasformazioni tipiche della situazione sperimentale (es. il peso di un pesce è invariante alle condizioni di luce, la luminanza no). Inoltre, deve esistere una probabile relazione tra le feature misurate e la classe da predire.

2.5.2 Rumore e outlier

I dati del mondo reale contengono inevitabilmente **rumore**, ovvero perturbazioni dovute a fenomeni non controllabili o non noti. Le cause di tale spostamento dai valori "ideali" possono essere:

- **Endogene**: Interne al fenomeno (es. un pesce con una dieta o storia anomala).
- **Esogene**: Dovute all'osservatore o allo strumento utilizzato (es. macchina fotografica starata, etichettatore distratto).

I dati molto "fuori norma" rispetto ai valori tipici di una classe sono definiti **outlier**. Un buon algoritmo di classificazione deve essere **robusto**, ovvero deve avere una forma di "protezione" o resistenza alle deviazioni che il rumore impone al processo decisionale.

2.6 Valutazione degli errori

Contare gli errori è essenziale, ma una singola percentuale di errore non è sufficientemente descrittiva. Questo perché non tutti gli errori sono uguali: i **costi degli errori** spesso **non sono uniformi** o simmetrici.

Ad esempio, in una diagnosi medica:

- Classificare un sano come malato (Falso Positivo) è un errore con un costo relativamente basso (paura, un test aggiuntivo).
- Classificare un malato come sano (Falso Negativo) è un errore gravissimo, che ritarda la diagnosi e può costare la vita al paziente.

Per analizzare questa asimmetria si usa la **matrice di confusione**. È una griglia quadrata che riporta quante istanze della classe "Reale" (sulle colonne) sono state assegnate alla classe "Prevista" (sulle righe).

Un classificatore perfetto ha come matrice di confusione la matrice identica (tutti i valori sulla diagonale principale, zero altrove). Un buon classificatore avrà valori percentuali bassi al di fuori della diagonale principale.

2.7 Fasi di un sistema di classificazione

Il processo di classificazione automatica si articola in diverse fasi:

1. **Sensing (o sampling)**: Raccolta dei dati dal mondo fisico e loro digitalizzazione.
2. **Segmentazione**: Partizione dei dati in unità significative, pulizia ed eliminazione di dati irrilevanti.
3. **Estrazione delle feature**: Misurazione delle caratteristiche (quantitative o qualitative). È cruciale scegliere feature invarianti e rilevanti.
4. **Classificazione**: Esecuzione dell'algoritmo scelto per l'assegnazione delle etichette.
5. **Post-processing**: Valutazione della qualità della classificazione e dei costi associati agli errori.
6. **Decisione**: Utilizzo effettivo del classificatore per risolvere il problema reale.

La costruzione di un classificatore è un **ciclo iterativo** che prevede la raccolta dei dati, la selezione delle feature, la scelta di un modello matematico, il training dell'algoritmo e infine la sua valutazione, ripetendo i passi per migliorare le prestazioni.