Introduzione all'Ecosistema Python:
Distinzione tra Linguaggio, Intelligenza Artificiale e Machine Learning

Dispensa del Corso

1 Introduzione

Nel panorama tecnologico odierno, termini come "Python", "Intelligenza Artificiale" (AI) e "Machine Learning" (ML) vengono spesso usati in modo intercambiabile. Questa sovrapposizione crea confusione nei principianti. Lo scopo di questo documento è chiarire le definizioni, le gerarchie e le differenze sostanziali tra questi concetti, ponendo le basi per un utilizzo consapevole degli strumenti informatici.

2 La Distinzione Fondamentale: Strumento vs. Metodologia

Il primo passo per comprendere questo ecosistema è distinguere tra il mezzo (il linguaggio) e il fine (l'applicazione).

Concetto Chiave

Python è il linguaggio di programmazione (lo strumento).
Machine Learning è una tecnica di analisi dei dati (l'applicazione).

Per usare un'analogia: Python è la "penna", mentre il Machine Learning è la "poesia". Si può usare la penna (Python) per scrivere la lista della spesa (automazione di base), per risolvere equazioni (calcolo scientifico) o per scrivere poesie (Machine Learning). Il fatto di possedere una penna non rende automaticamente scrittori di poesie.

3 Gerarchia dell'Intelligenza Artificiale

È fondamentale visualizzare questi concetti come insiemi concentrici, dal più generale al più specifico:

1. **Intelligenza Artificiale (AI):** È il termine ombrello più ampio. Indica qualsiasi tecnica che permetta ai computer di imitare il comportamento intelligente umano. Questo include logiche semplici basate su regole ("SE succede X, ALLORA fai Y") fino a sistemi complessi.

2. **Machine Learning (ML):** È un sottoinsieme dell'AI. Invece di programmare esplicitamente ogni regola, forniamo alla macchina grandi quantità di dati e le permettiamo di "apprendere" le relazioni statistiche che legano questi dati.

3. **Deep Learning (DL):** È un sottoinsieme specializzato del Machine Learning ispirato alla struttura del cervello umano (reti neurali artificiali), capace di apprendere da enormi moli di dati non strutturati (immagini, audio, testo).

4 Il Paradigma: Programmazione Tradizionale vs. Machine Learning

La differenza cruciale per un ingegnere o uno sviluppatore risiede nel modo in cui si approccia la risoluzione del problema.

4.1 Programmazione Tradizionale (e Calcolo Scientifico)

In questo approccio, l'umano deve conoscere le regole a priori.

- **Input:** Dati + Regole (Formule, Leggi Fisiche).
- **Output:** Risultati.

Esempio: Nel calcolo di una funzione di trasferimento per un sistema di controllo, noi forniamo al computer l'equazione esatta ($G(s) = \frac{100}{s^2+6s+100}$). Python agisce come una calcolatrice avanzata. Non c'è apprendimento, il risultato è deterministico.

4.2 Machine Learning

In questo approccio, l'umano spesso non conosce le regole esatte o sono troppo complesse da scrivere.

- **Input:** Dati + Risultati desiderati (Esempi storici).
- **Output:** Regole (Il Modello).

Esempio: Forniamo al computer 10.000 registrazioni di un impianto in funzione e gli chiediamo di trovare il modello matematico che approssima quel comportamento. La macchina "impara" dai dati.

5 Perché Python?

Python è diventato il linguaggio standard "de facto" per la scienza dei dati e l'ingegneria non perché sia "intelligente" di per sé, ma per le sue librerie (moduli aggiuntivi scritti da altri sviluppatori).

5.1 Librerie per scopi diversi

Quando utilizziamo Python, possiamo caricare "pacchetti" diversi a seconda dell'obiettivo:

- **Per il Calcolo Scientifico (NON è ML):** Utilizziamo librerie come `numpy`, `scipy` o `control`. Queste librerie eseguono calcoli matematici precisi definiti dall'utente.

```
import numpy as np
# Calcolo deterministico: calcola il seno di 90 gradi
risultato = np.sin(np.deg2rad(90))
print(risultato) # Output: 1.0

```

Listing 1: Esempio di Calcolo Scientifico (No ML)

- **Per il Machine Learning:** Utilizziamo librerie come `scikit-learn`, `tensorflow` o `pytorch`. Queste librerie contengono algoritmi statistici per l'apprendimento dai dati.

```
from sklearn.linear_model import LinearRegression
# Dati di esempio (Input e Output noti)
X = [[1], [2], [3]]
y = [2, 4, 6]
# Il modello impara la regola (y = 2x)
model = LinearRegression().fit(X, y)

```

Listing 2: Concetto di Machine Learning

6 Conclusione

L'apprendimento di Python è il primo passo necessario per operare nel mondo moderno dell'ingegneria e dell'analisi dati. Tuttavia, scrivere codice in Python non significa automaticamente fare Intelligenza Artificiale.

- Se state scrivendo formule note per ottenere un grafico, state facendo **Programmazione/Calcolo Scientifico**.
- Se state usando dati storici per predire eventi futuri senza conoscere la formula esatta, state facendo **Machine Learning**.

Entrambi gli approcci sono validi e spesso complementari nella risoluzione di problemi complessi.

→ Clicca qui per la Lezione Python_2