



Benvenuti nel Mondo dei Dati!

(Spoiler: È molto più cool di quanto pensiate)

 III Liceo Scientifico Biella - Scienze Applicate

 Python Biella Group



Recap: Cosa abbiamo fatto finora?

- Colpito alieni (e imparato PyGame Zero)
- Aiutato Tony a trovare la musica
- Creato account GitHub (benvenuti nel club dei developer!)
- Debuggato Stranger Stars (e sopravvissuto agli errori)

Oggi: Entriamo nel regno dove l'AI trova il suo cibo preferito... i DATI! 🍕



AI senza dati = ?



Un'AI senza dati è come:

- 🍕 Una pizza senza ingredienti
- 💡 Uno smartphone senza batteria
- 🎮 Un videogioco senza giocatori
- 🧠 Voi senza colazione la mattina



Perché i dati sono così importanti?

L'AI impara dai dati, proprio come voi imparate dagli errori di Python! 😅

Più dati = AI più intelligente

Dati di qualità = AI affidabile

Dati sporchi = AI confusa (garbage in, garbage out!)

Esempi reali:

- ChatGPT → addestrata su miliardi di testi
- Netflix → suggerimenti basati sui vostri gusti
- Spotify → playlist create analizzando milioni di ascolti



Il Viaggio di un DATO

1. RACCOLTA  → Dati grezzi (caotici, disordinati)
2. PULIZIA  → Rimuovere errori e duplicati
3. ANALISI  → Trovare pattern interessanti
4. VISUALIZZAZIONE  → Grafici che catturano l'attenzione
5. AI/ML  → Predizioni e decisioni intelligenti

Oggi ci concentriamo sui passi 1-3!



🔗 La catena

DATO → CSV → DataFrame

DATO

```
Character,Spell,Damage,Precision  
Harry,Expelliarmus,10,1  
Harry,Stupeficium,15,1  
...
```

CSV (file)

→ harry.csv (salvato sul disco)

DATAFRAME (in Python)

→ Caricato in memoria (pronto per l'analisi!)



Il percorso

Il percorso:

1.  **Raccogli dati**
2.  **Salvali in CSV** (formato leggero e universale)
3.  **Caricali in Python** come DataFrame con Polars
4.  **Analizza, filtra, trasforma!**



Cos'è davvero un file CSV?

CSV sta per *Comma Separated Values* (Valori Separati da Virgole).

È il formato più semplice per salvare dati in forma tabellare!

Immaginate che Excel sia un vestito elegante e costoso.

Il CSV è quel vestito ridotto ai minimi termini: solo l'intimo.

- Niente colori.
- Niente formule magiche.
- Niente celle giganti.
- Solo. Puro. Testo.



00 Sotto il cofano di un CSV

Se apri un file .csv con il Blocco Note (Notepad), non vedrai tabelle, ma qualcosa di simile a questo:

```
Character,Spell,Damage,Precision
```

```
Harry,Expelliarmus,10,1
```

```
Harry,Stupefy,15,1
```

```
Harry,Expecto Patronum,20,1
```

- Prima riga = nomi delle colonne (header)
- Ogni riga = un record
- Le virgole separano le colonne
- È un semplice file di testo (apribile con Notepad)

Perché usare CSV?

```
Character,Spell,Damage,Precision
Harry,Expelliarmus,10,1
Harry,Stupefy,15,1
Harry,Expecto Patronum,20,1
```

È *universale*: lo legge chiunque (Python, R, Excel, persino il tuo tostapane smart).

È *leggero*: occupa pochissimo spazio.

È "onesto": quello che vedi è quello che passi all'AI.



CSV vs Excel

File Excel (.xlsx)



- Formato binario complesso, può avere più fogli (sheets)
- Può contenere formule, grafici, colori, formattazione
- Più "pesante" (dimensioni maggiori)

File CSV (.csv)



- Semplice testo, un solo "foglio"
- Solo dati grezzi, niente formule o colori
- Leggerissimo e velocissimo da leggere!



CSV nella Vita Reale

Dove li trovate:

- Esportazioni da Excel (Salva come → CSV)
- Download di dati da siti web
- Dataset per Machine Learning
- File di configurazione di giochi
- Backup di contatti del telefono

Fun fact: Anche se si chiama "Comma" Separated,
in spesso si usa il ; (punto e virgola) invece della , (virgola)!



Cos'è un DataFrame?

nome	età	città	punteggio
Mario	16	Milano	95
Giulia	17	Roma	88
Luca	16	Torino	92
Sofia	17	Napoli	97

È una tabella che arriva in memoria dove ogni:

- Riga = un'osservazione/record
- Colonna = una caratteristica/variabile
 - ... immaginate Excel... ma SUPER POTENZIATO! 💪



Excel vs DataFrames

Excel



- Click, click, click... 🐀
- Ottimo per piccoli dataset
- Limite: ~1 milione di righe
- Velocità: 💰

DataFrames (con Polars)



- Codice riutilizzabile
- Gestisce milioni/miliardi di righe
- Velocità: 🚶



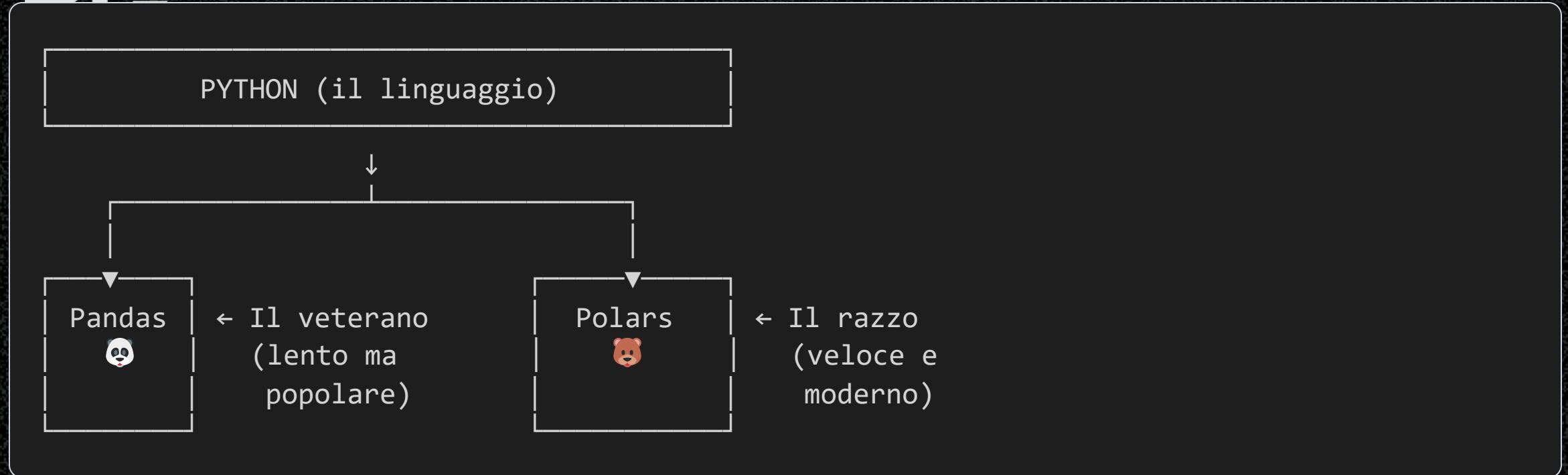
Polars: Cos'è?

Polars è una LIBRERIA Python , un **set di strumenti specializzati** che qualcun altro ha già costruito per voi!

```
import polars as pl # Importi la libreria  
df = pl.read_csv("dati.csv") # Usi le sue funzioni magiche!
```

- Python = La vostra cassetta degli attrezzi base 
- Polars = Set professionale per lavorare con tavole di dati 
- Altre librerie = Altri set specializzati (PyGame per giochi, Matplotlib per grafici...)

Polars si specializza in leggere, manipolare e analizzare dati tabulari (dataframes) velocemente!



Altre librerie utili per i dati:

NumPy → Calcoli matematici veloci, Matplotlib → Creare grafici

Scikit-learn → Machine Learning



Pandas vs Polars

Pandas (il classico)

- Libreria più famosa, esiste dal 2008
- Un po' lenta con grandi dataset

Polars (il nuovo campione)

- Libreria moderna (2020)
- VELOCISSIMA (scritta in Rust ) , usa tutti i core del vostro PC!
- Sintassi più chiara ed elegante

Oggi usiamo Polars perché siamo fighi così 😎



Polars: I Super Poteri

1. Velocità Supersonica

- 5-10x più veloce di Pandas!

2. Memoria Efficiente

- Consuma meno RAM

3. Lazy Evaluation

- Ottimizza le operazioni automaticamente

4. Multi-threading

- Usa tutti i core della CPU



Primi Passi con Polars

```
import polars as pl

# Leggere un CSV (facilissimo!)
df = pl.read_csv("spells.csv")

# Info sul DataFrame
print(df.describe())

# Vedere le prime righe
print(df.head())
```

È così semplice che potrebbe farlo anche il vostro gatto 😺
(ok, forse no... ma è comunque facilissimo!)



```
# Filtrare per una condizione
df.filter(pl.col("damage") >= 25)

# Filtrare con più condizioni (AND)
df.filter(
    (pl.col("precision") >= 0.8) &
    (pl.col("character") == "Harry")
)

# Filtrare con OR
df.filter(
    (pl.col("damage") > 30) |
    (pl.col("precision") > 0.8)
)
```

Attenzione: Si usa `&` per AND e `|` per OR (non `and` / `or` !)



SFIDA: Cosa fa questo codice?

```
import polars as pl

df = pl.read_csv("studenti.csv")
risultato = df.filter(pl.col("voto") > 9)
```

A) Elimina tutti i voti sopra 9

B) Seleziona solo gli studenti con voto > 9

C) Modifica tutti i voti a 9

D) Crasherà sicuramente *



Risposta SFIDA

B) Seleziona solo gli studenti con voto > 9

Il metodo `.filter()` è come un setaccio:

Lascia passare solo le righe che soddisfano la condizione

`pl.col("voto")` seleziona la colonna "voto"

`> 9` è la condizione da verificare

È come dire: "Ehi Polars, dammi solo i secchioni!" 😎



Selezionare Righe Specifiche

```
# Prima riga  
prima = df[0]  
  
# Prime 5 righe  
prime_cinque = df[0:5]  
  
# Ultima riga  
ultima = df[-1]  
  
# Righe dalla 10 alla 20  
intervallo = df[10:20]
```

È come slice delle liste Python! 🍕
(ma con DataFrame invece di liste)



Sample: pesca a caso!

```
# Pesca 5 righe casuali
df.sample(n=5)
# Pesca il 10% del dataset
df.sample(fraction=0.1)
# Pesca con "rimessa" (stessa riga può uscire più volte)
df.sample(n=100, with_replacement=True)
# Pesca riproducibile (stesso "caso" ogni volta)
df.sample(n=10, seed=42)
```

Quando è utile?

- 🎲 Selezionare un numero di record in modo casuale (piccolo dataset di test)
- 🔬 Analisi statistica (campionamento)
- 🎰 Quando vuoi un po' di caos controllato!



SFIDA: Debug Challenge!

Cosa c'è di sbagliato qui?

```
import polars as pl

df = pl.read_csv("studenti.csv")
risultato = df.filter(pl.col("voto") > 7 and pl.col("materia") == 'Inglese')
```

Suggerimento: Ricordate come si combinano le condizioni?

Tempo: 20 secondi... ⏳



Risposta SFIDA

Errore: Usare `and` invece di `&`!

Versione corretta:

```
df.filter(pl.col("voto") > 7 & pl.col("materia") == 'Inglese')
```

Perché?

`and` è per valori booleani semplici (True/False)

`&` è per operazioni vettoriali (su intere colonne)

Le parentesi sono obbligatorie!



Risorse Utili

- Documentazione Polars: <https://docs.pola.rs/>
- Polars Cheat Sheet: Cercate "Polars cheat sheet" su Google
- Confronto Pandas/Polars: Interessante per capire le differenze
- Il vostro GitHub: Per condividere e imparare dagli altri!
- Remember: Google e Stack Overflow sono vostri amici! 



SFIDA: Indovina il Dataset!

Scenario: Hai un file CSV con queste colonne:

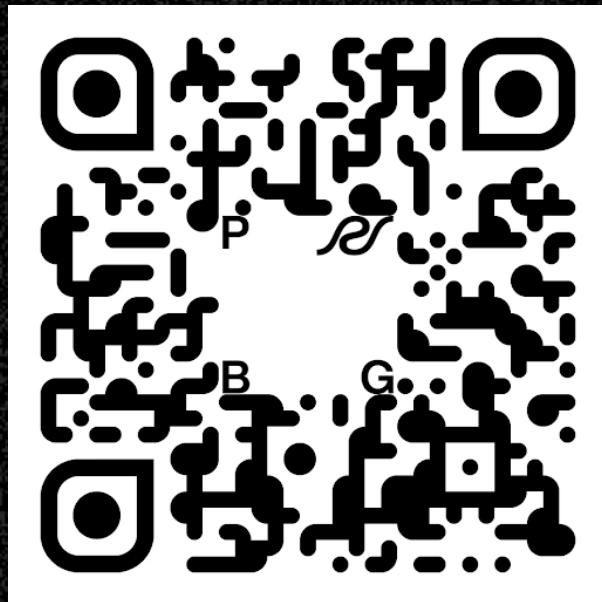
- character
- spell
- damage
- precision

Domanda: Che tipo di gioco potrebbe usare questi dati?

Pensateci 10 secondi, ma ne parleremo dopo... 

P S
BG

Grazie per l'attenzione...



"C'è sempre qualcosa da imparare per migliorarci e crescere...insieme!"