# XV CONVEGNO NAZIONALE SETTORE EDUCATION AICQ

ORGANIZZATO DA

aicq

Associazione Italiana Cultura Qualità
Settore Education


# IA: il futuro non aspetta

*come affrontare consapevolmente le sfide*

**ITT Mazzotti** Treviso - **27 febbraio 2026**

# Come introdurre i LLM a scuola??

prof. Irene Campagnolo
discipline: Informatica, Sistemi e Reti
IIS KENNEDY - Monselice (PD)

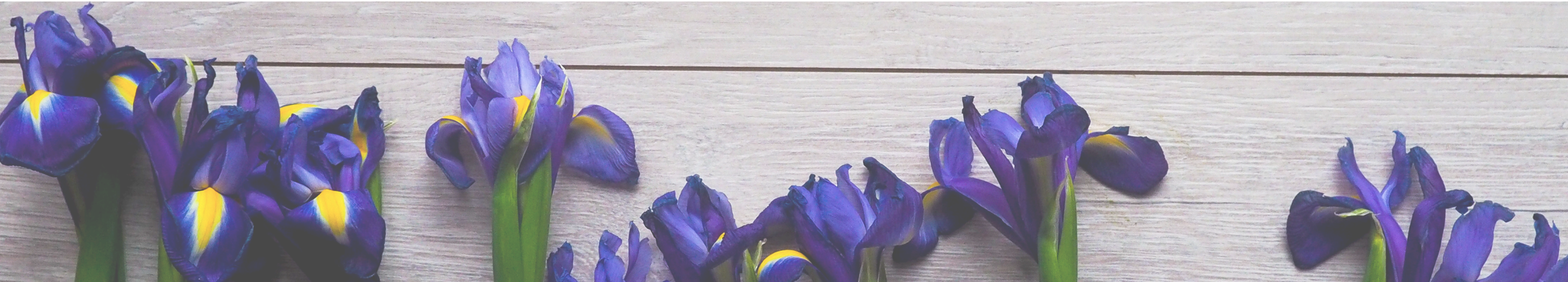
# con un fiore...

# Perché proprio gli iris?

Il dataset Iris è un dataset multivariato introdotto da Ronald Fisher nel 1936. Consiste in 150 istanze di Iris misurate da Edgar Anderson e classificate secondo tre specie: Iris **setosa**, Iris **virginica** e Iris **versicolor**. Le quattro variabili considerate sono la **lunghezza e la larghezza del sepalo e del petalo**.

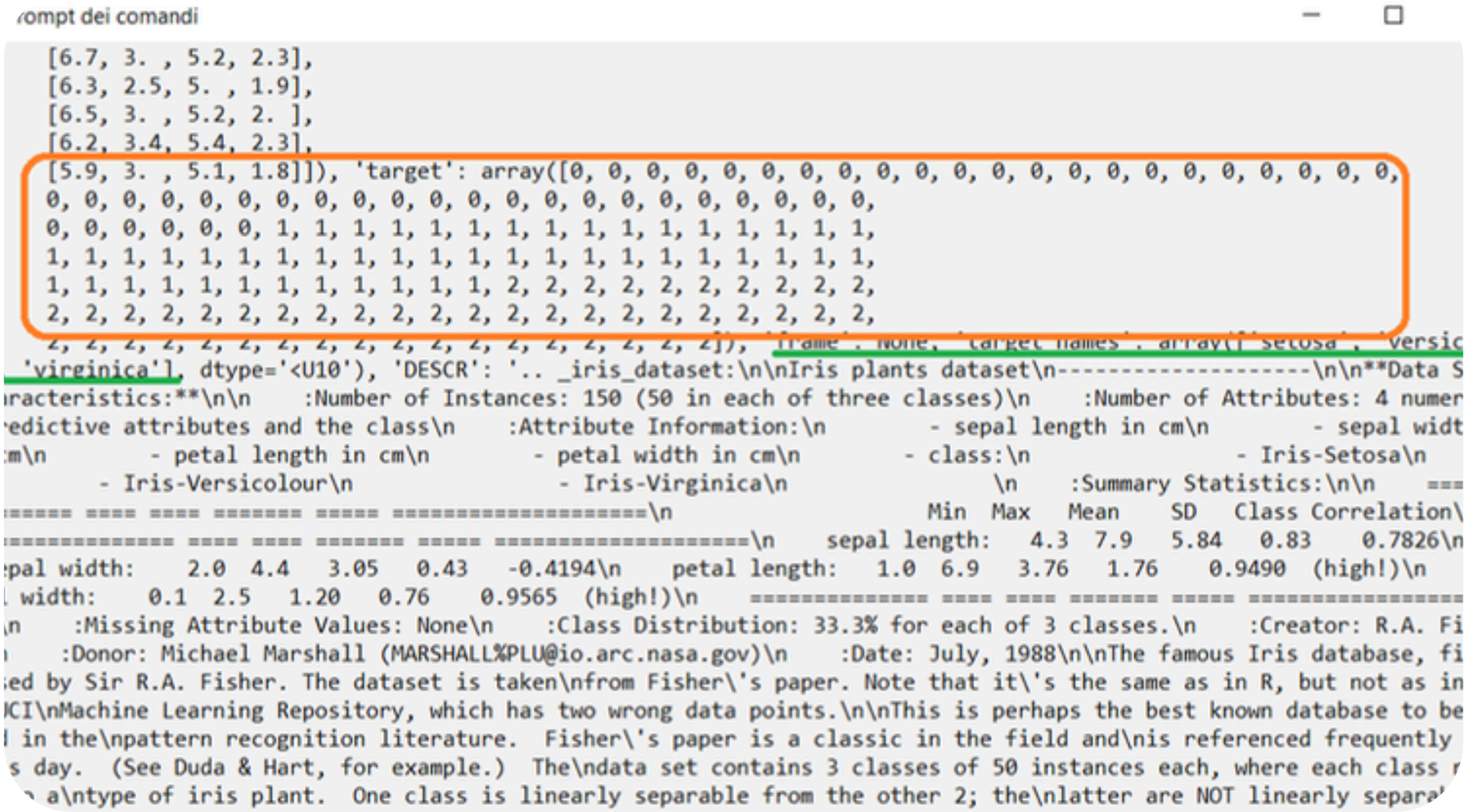

Sepal
Petal
Versicolor

quante misure
sono state
raccolte?



```
C:\Users\irene\EserciziPython>python carica_iris.py
{'data': array([[5.1, 3.5, 1.4, 0.2],
       [4.9, 3. , 1.4, 0.2],
       [4.7, 3.2, 1.3, 0.2],
       [4.6, 3.1, 1.5, 0.2],
       [5. , 3.6, 1.4, 0.2],
       [5.4, 3.9, 1.7, 0.4],
       [4.6, 3.4, 1.4, 0.3],
       [5. , 3.4, 1.5, 0.2],
       [4.4, 2.9, 1.4, 0.2],
       [4.9, 3.1, 1.5, 0.1],
       [5.4, 3.7, 1.5, 0.2],
       [4.8, 3.4, 1.6, 0.2],
       [4.8, 3. , 1.4, 0.1],
```

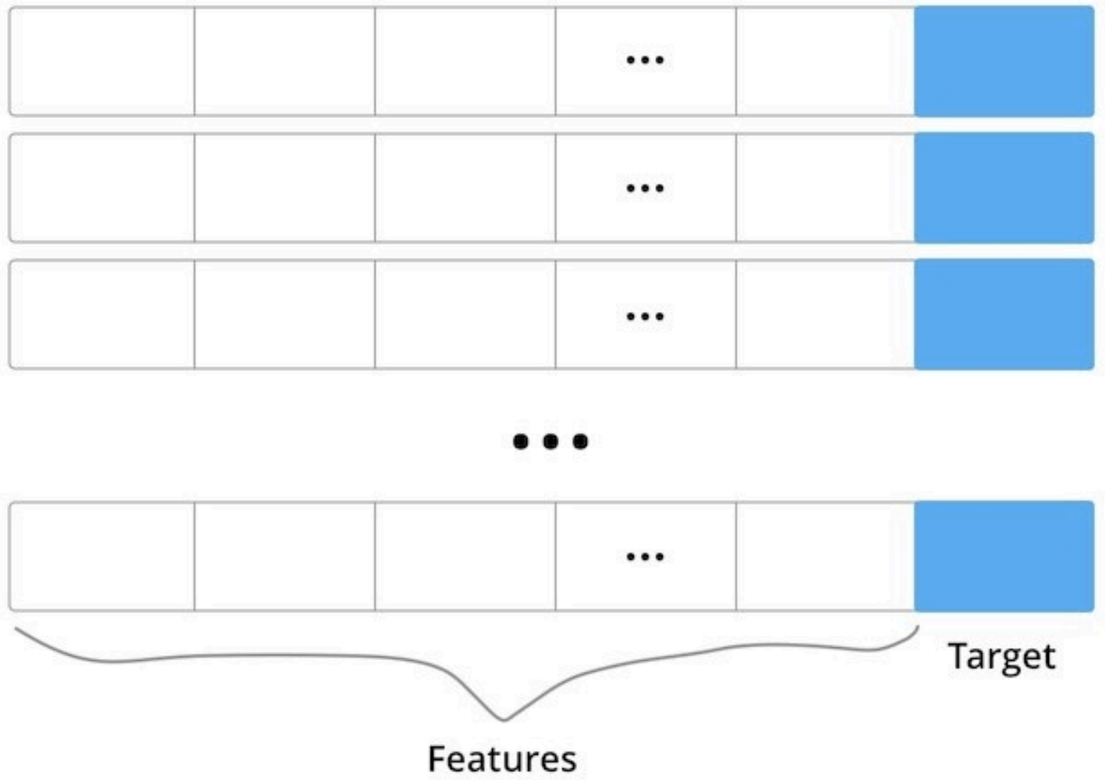
# ma ci sono altre informazioni...molto interessanti!

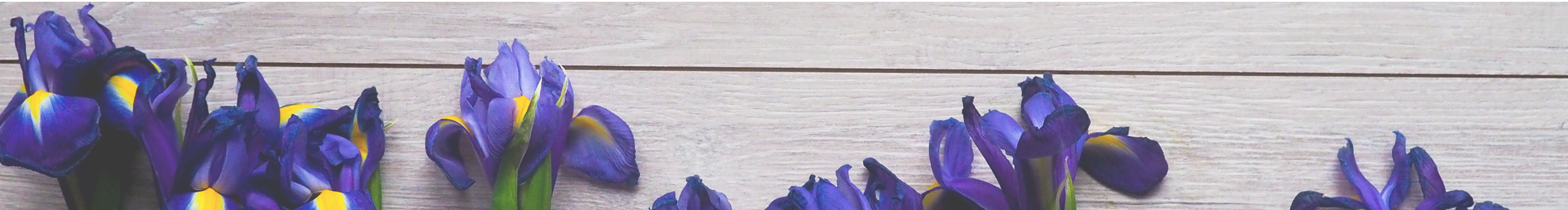


```
'target': array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]), 'frame': None, 'target_names': array(['setosa', 'versicol
r', 'virginica'], dtype='<U10'), 'DESCR': '.. _iris_dataset:\n\nIris plants dataset\n--------------------\n\n**Data Set
```


**dopo un lunghissimo array formato da 4 colonne, compare il 'target' dove i valori 0,1 e 2 corrispondono al tipo di iris: 'setosa', 'versicolor' e 'virginica'**

# I nostri dati sono già divisi in features e target



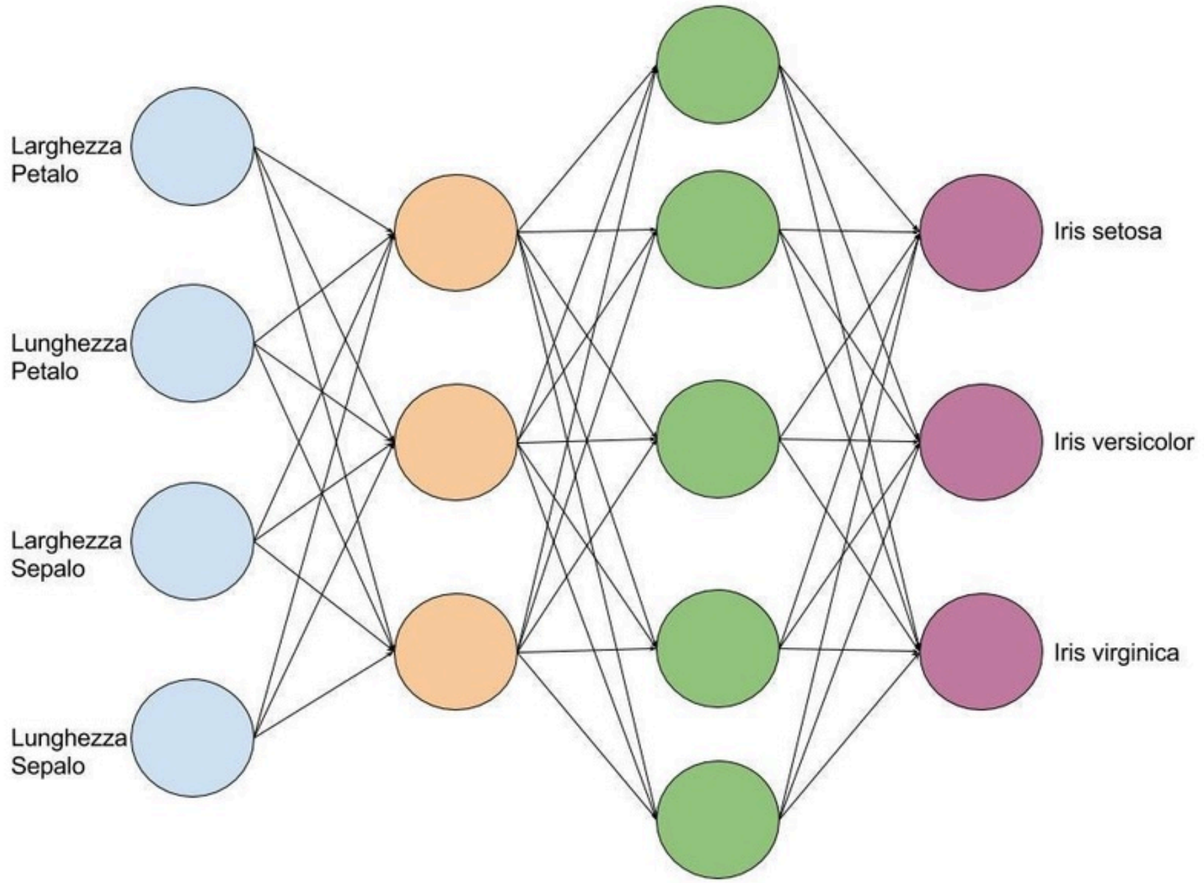
features

target

| sepal_length | sepal_width | petal_length | petal_width | species |
|---|---|---|---|---|
| 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| 4.6 | 3.1 | 1.5 | 0.2 | setosa |
| 5.0 | 3.6 | 1.4 | 0.2 | setosa |
| ... | ... | ... | ... | ... |
| 6.7 | 3.0 | 5.2 | 2.3 | virginica |
| 6.3 | 2.5 | 5.0 | 1.9 | virginica |
| 6.5 | 3.0 | 5.2 | 2.0 | virginica |
| 6.2 | 3.4 | 5.4 | 2.3 | virginica |
| 5.9 | 3.0 | 5.1 | 1.8 | virginica |

Così però i dati non ci dicono nulla...

**obiettivo:** inserire 4 valori per ottenere la specie di appartenenza



# Iris e Machine Learning....

# come dividiamo le misure?

training

test

Con il dataset di **training** verrà utilizzato
il metodo **fit** per eseguire l'apprendimento

con quello di **test** si proverà il modello
addestrato per valutarne l'efficacia

X y

112 75% train

38 25% test

Target

Features

# dove posso prendere i dati?

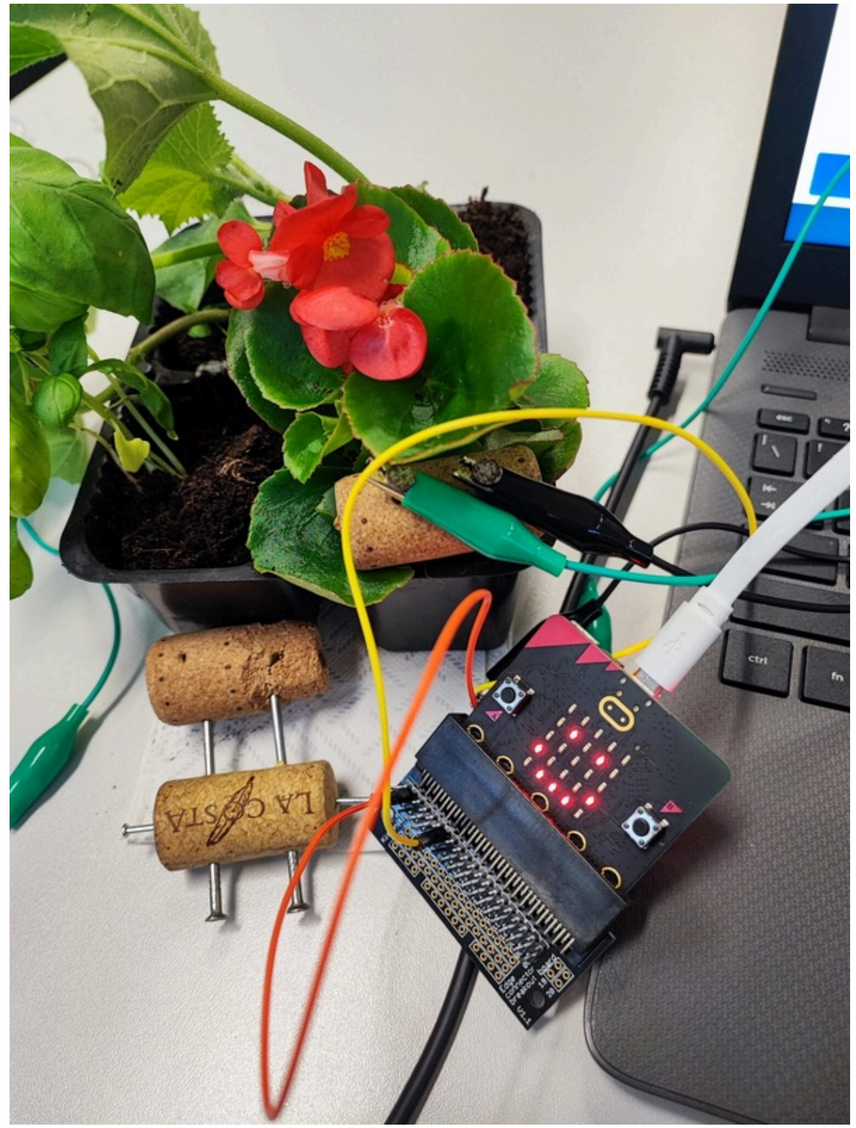
- 01 Kaggle
- 02 Datiopen.it
- 03 dait.interno.gov.it
- 04 Astro Pi (moodle)
- 05 Google Trends

# oppure...

- utilizzare una scheda micro:bit

- costruire un sensore

- salvare i dati in un file di testo



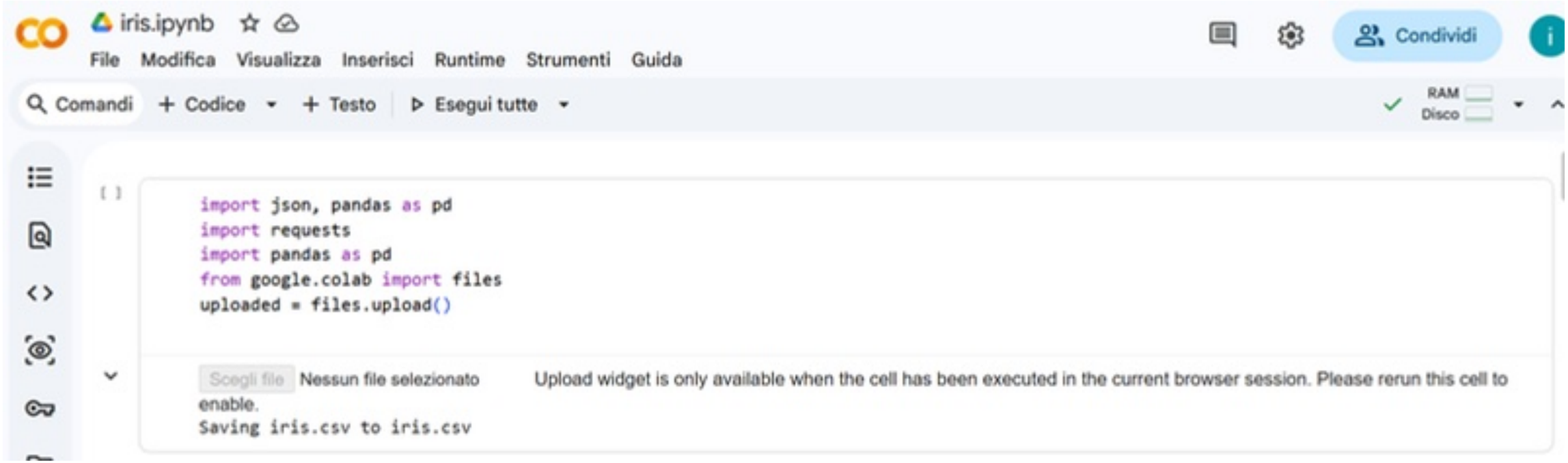
quale linguaggio di programmazione utilizzare?

# Python

- installare Python in laboratorio
- utilizzare colab


Google
colab

# si può utilizzare un file presente nel nostro drive



# oppure raggiungibile da un link

```python
#CARICAMENTO LIBRERIE
import pandas as pd
import seaborn as sns       # For visualizations
import matplotlib.pyplot as plt  # For plots
from sklearn.model_selection import train_test_split  # For splitting the data
from sklearn.svm import SVC  # The Support Vector Classifier
from sklearn.metrics import accuracy_score  # To check model accuracy
url = 'http://strafanti.altervista.org/iris.csv'
col_name = ['sepal-lenght','sepal-width','petal-lenght','petal-width','class']
dataset = pd.read_csv(url, names = col_name)
dataset['class'].value_counts()
```

```
...
```

| class | count |
|---|---|
| Iris-setosa | 50 |
| Iris-versicolor | 50 |
| Iris-virginica | 50 |

dtype: int64

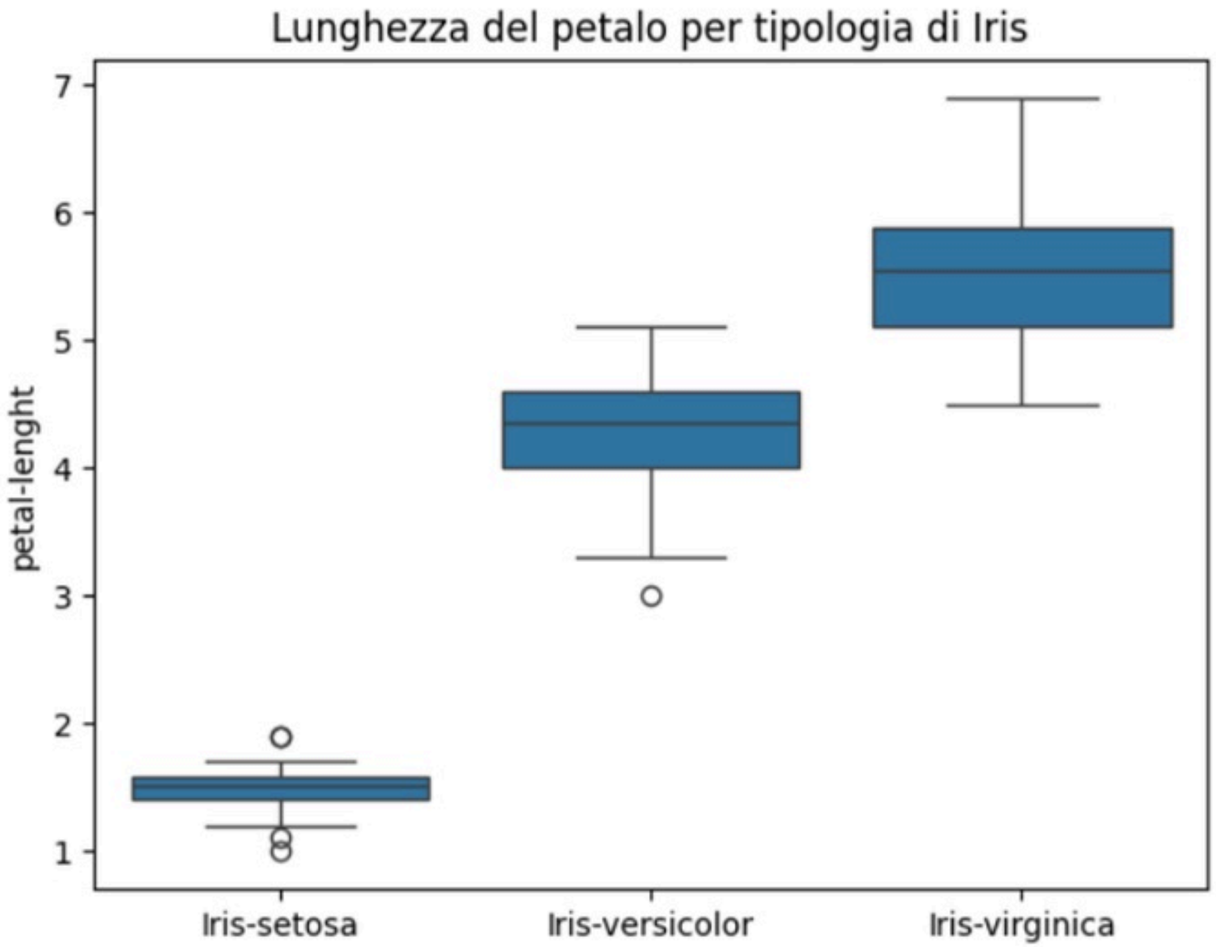
# come si presentano le prime 5 righe...

```
dataset.head()
```

| | sepal-lenght | sepal-width | petal-lenght | petal-width | class |
|---|---|---|---|---|---|
| **0** | 5.1 | 3.5 | 1.4 | 0.2 | Iris-setosa |
| **1** | 4.9 | 3.0 | 1.4 | 0.2 | Iris-setosa |
| **2** | 4.7 | 3.2 | 1.3 | 0.2 | Iris-setosa |
| **3** | 4.6 | 3.1 | 1.5 | 0.2 | Iris-setosa |
| **4** | 5.0 | 3.6 | 1.4 | 0.2 | Iris-setosa |

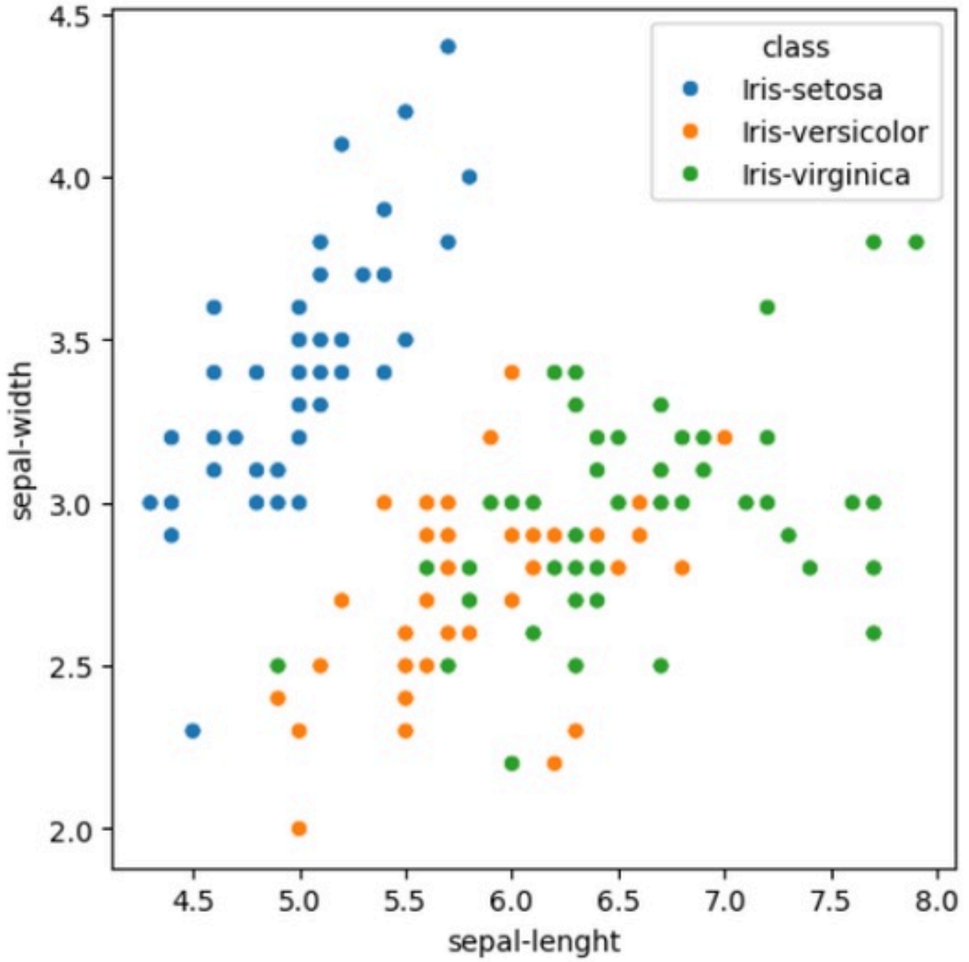
# qualche grafico…

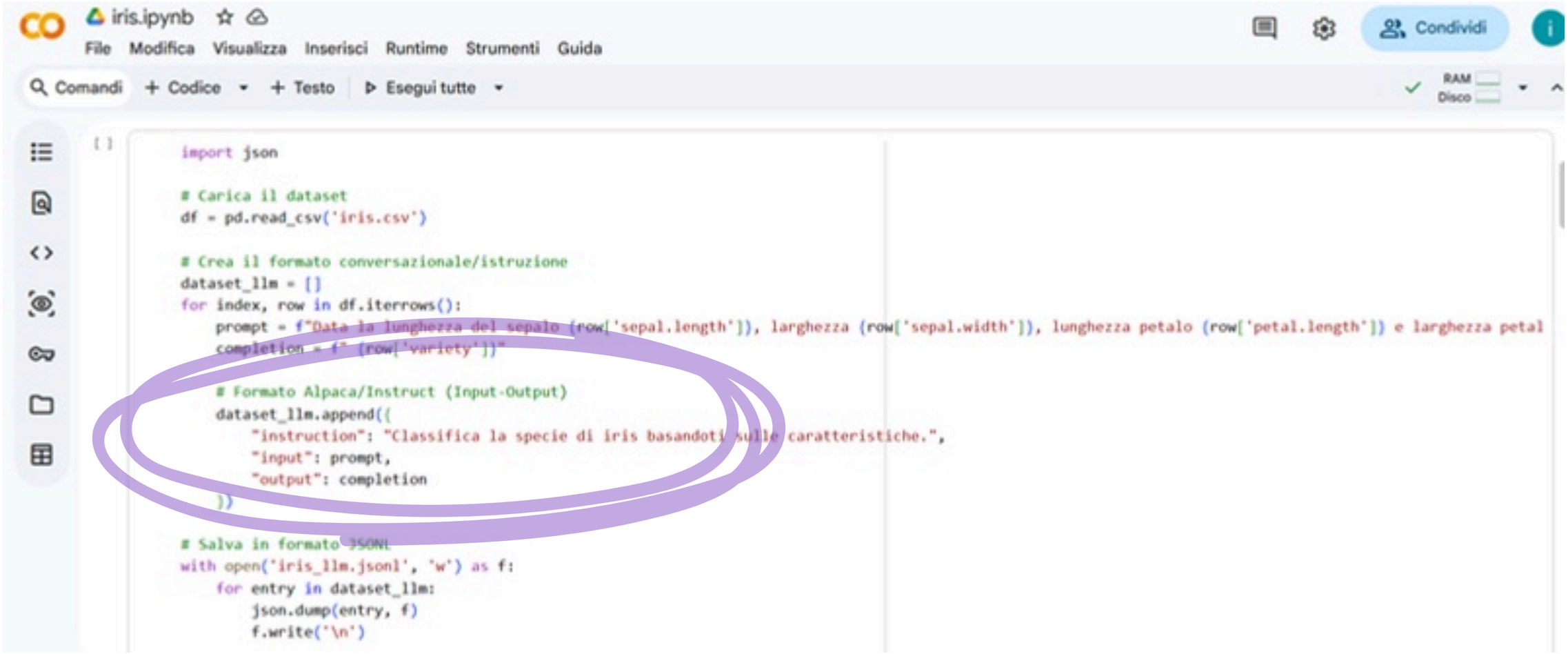
```
sns.boxplot(x='class', y='petal-lenght', data=dataset)
plt.title('Lunghezza del petalo per tipologia di Iris')
plt.show()
```



# qualche grafico...

```
fig, ax = plt.subplots(figsize=(5, 5))
sns.scatterplot(data=dataset, x="sepal-lenght", y="sepal-width", ax=ax, hue="class")
plt.tight_layout()
```




iris.ipynb
File Modifica Visualizza Inserisci Runtime Strumenti Guida
Condividi
Comandi + Codice + Testo Esegui tutte
RAM
Disco
[ ]
import json
# Carica il dataset
df = pd.read_csv('iris.csv')
# Crea il formato conversazionale/istruzione
dataset_llm = []
for index, row in df.iterrows():
prompt = f"Data la lunghezza del sepalo {row['sepal.length']}, larghezza {row['sepal.width']}, lunghezza petalo {row['petal.length']} e larghezza petal
completion = f" {row['variety']}"
# Formato Alpaca/Instruct (Input-Output)
dataset_llm.append({
"instruction": "Classifica la specie di iris basandoti sulle caratteristiche.",
"input": prompt,
"output": completion
})
# Salva in formato JSONL
with open('iris_llm.jsonl', 'w') as f:
for entry in dataset_llm:
json.dump(entry, f)
f.write('\n')

```
print(json.dumps(dataset_llm,indent=2))
```


```
[
  {
    "instruction": "Classifica la specie di iris basandoti sulle caratteristiche.",
    "input": "Data la lunghezza del sepalo 5.1, larghezza 3.5, lunghezza petalo 1.4 e larghezza petalo 0.2, che specie \u00e8?",
    "output": " Setosa"
  },
  {
    "instruction": "Classifica la specie di iris basandoti sulle caratteristiche.",
    "input": "Data la lunghezza del sepalo 4.9, larghezza 3.0, lunghezza petalo 1.4 e larghezza petalo 0.2, che specie \u00e8?",
    "output": " Setosa"
  },
  {
    "instruction": "Classifica la specie di iris basandoti sulle caratteristiche.",
    "input": "Data la lunghezza del sepalo 4.7, larghezza 3.2, lunghezza petalo 1.3 e larghezza petalo 0.2, che specie \u00e8?",
    "output": " Setosa"
  },
  {
    "instruction": "Classifica la specie di iris basandoti sulle caratteristiche.",
    "input": "Data la lunghezza del sepalo 4.6, larghezza 3.1, lunghezza petalo 1.5 e larghezza petalo 0.2, che specie \u00e8?",
    "output": " Setosa"
  },
```

```python
# Convert DataFrame to NumPy array
iris_data = dataset.values
# Split into features and labels
X = iris_data[:, 0:4]  # Features
y = iris_data[:, 4]    # Labels
```

```python
from sklearn.model_selection import train_test_split
# Split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```python
from sklearn.svm import SVC
# Create the SVM classifier
model = SVC()
# Train the model using the training data
model.fit(X_train, y_train)
```

```
SVC()
```

# inseriamo noi dei dati...

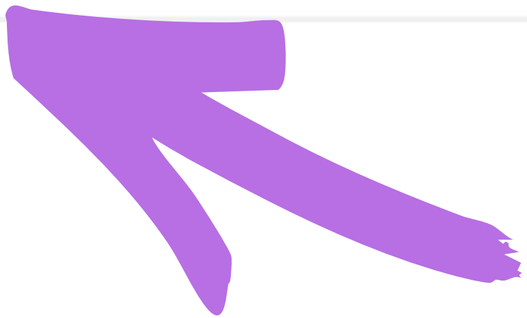
```
# Predict the test dataset
predictions = model.predict(X_test)
```

```
from sklearn.metrics import accuracy_score
# Calculate the accuracy of the model
accuracy = accuracy_score(y_test, predictions)
print("Accuracy:", accuracy)
```

```
Accuracy: 1.0
```

```
# Predicting a new data sample
sample = [[4.7, 3.2, 1.3, 0.2]]
prediction = model.predict(sample)
print("Predicted Iris Species:", prediction)
```

```
Predicted Iris Species: ['Iris-setosa']
```

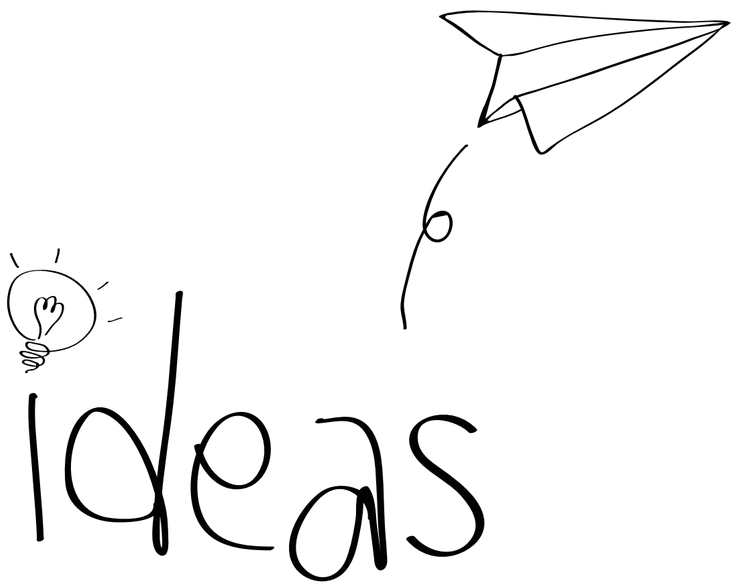


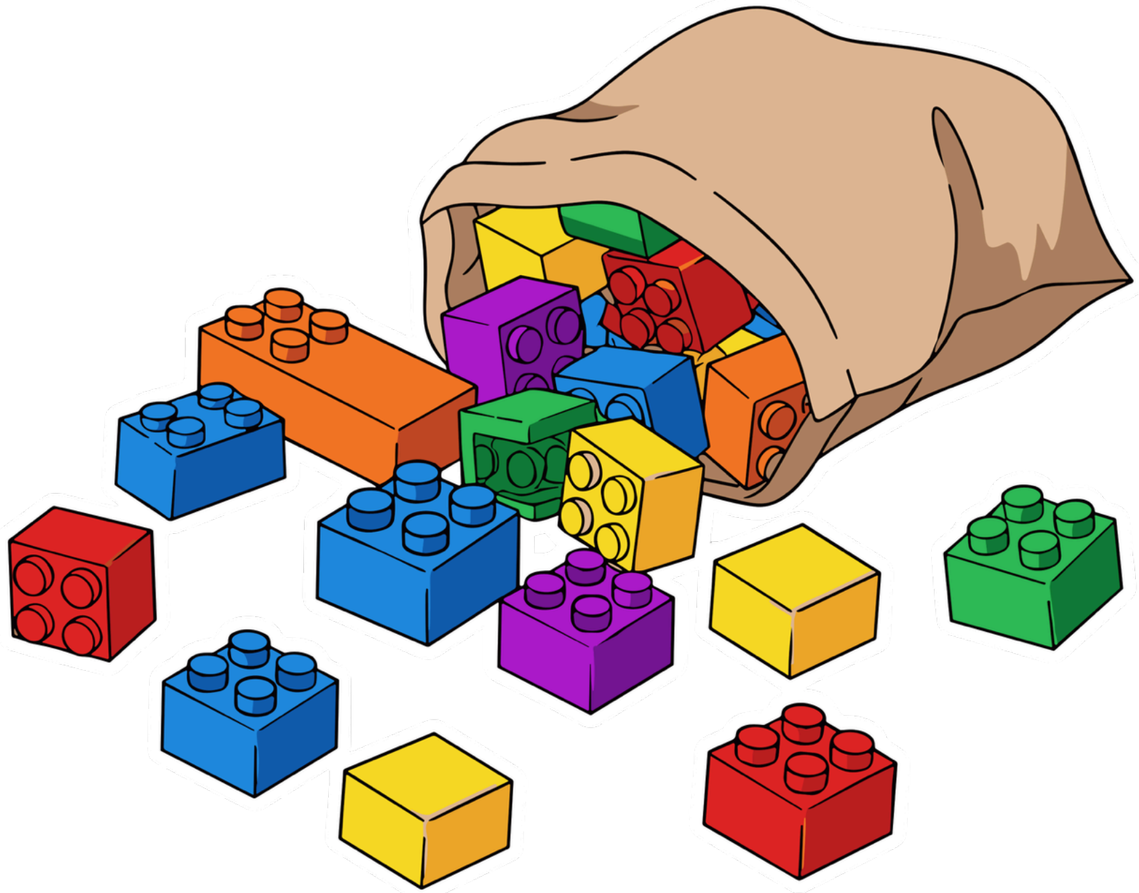
# e in locale?
# cosa posso utilizzare?


**Ollama is the easiest way to automate your work using open models, while keeping your data safe.**

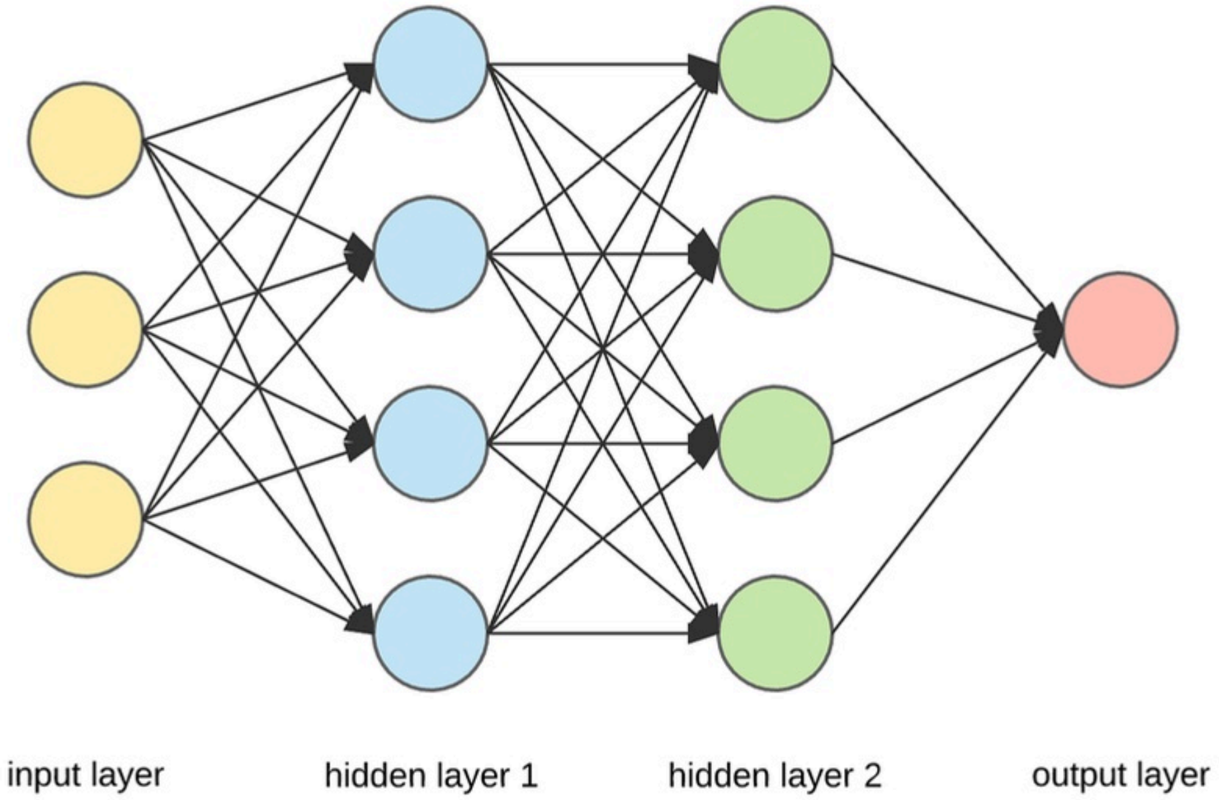

Models
Docs
Pricing
Search models
Sign in
Download
Download Ollama
macOS
Linux
Windows
irm https://ollama.com/install.ps1 | iex
paste this in PowerShell
or
Download for Windows
Requires Windows 10 or later

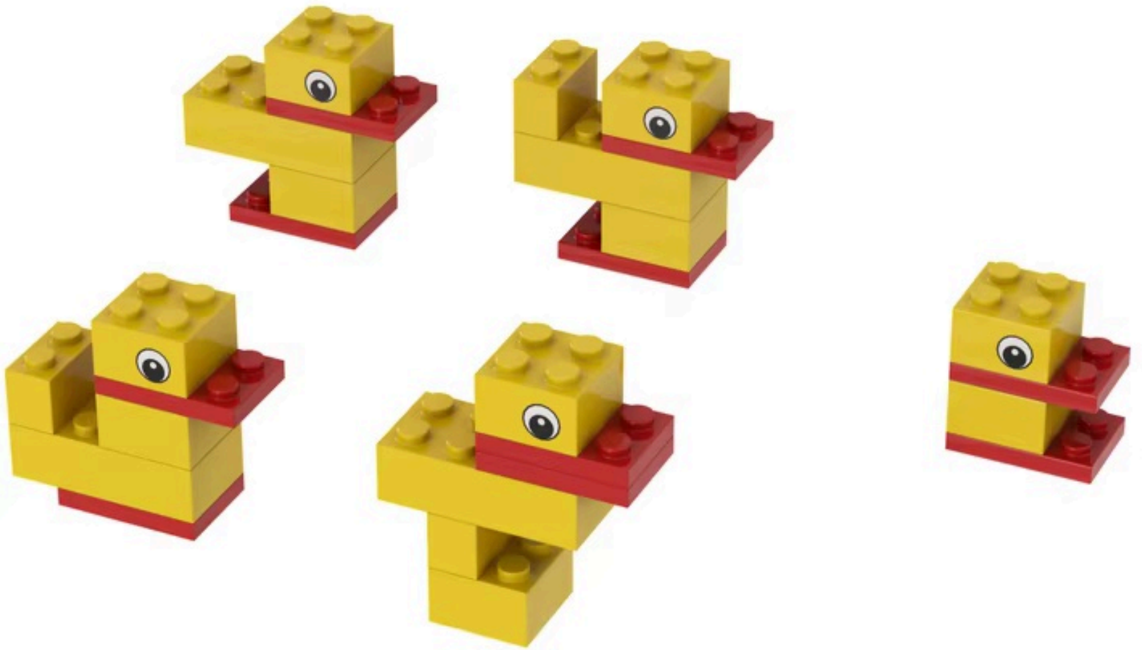
```
>>>
Ollama 0.16.3

▸ Run a model (llama3.2)
    Start an interactive chat with a model

  Launch Claude Code (not installed)
    Agentic coding across large codebases

  Launch Codex (not installed)
    OpenAI's open-source coding agent

  Launch OpenClaw (not installed)
    Personal AI with 100+ skills

  More...
    Show additional integrations
```


Ideas




# comporre una papera...


input layer
hidden layer 1
hidden layer 2
output layer




**https://irenecampagnolo.altervista.org/**