Applicazione di Machine Learning con Regressione Logistica
Stima del Rischio di Diabete

Prof. Fedeli Massimo

1 Obiettivo del progetto

In questa esercitazione realizziamo una piccola applicazione di intelligenza artificiale che utilizza un modello di regressione logistica per stimare la probabilità che una persona sia affetta da diabete.

L'applicazione:

- usa un dataset medico reale;
- addestra un modello di machine learning;
- salva il modello su file;
- mette il modello a disposizione tramite una pagina web in cui l'utente inserisce i propri dati.

2 Il dataset

Il modello viene addestrato usando un dataset pubblico contenente informazioni cliniche di diverse pazienti. Ogni riga del dataset rappresenta una persona, descritta da alcune misure mediche.

Le principali variabili sono:

- **Pregnancies** – Numero di gravidanze
- **Glucose** – Livello di glucosio nel sangue
- **BloodPressure** – Pressione sanguigna
- **SkinThickness** – Spessore della plica cutanea
- **Insulin** – Livello di insulina
- **BMI** – Indice di massa corporea
- **DiabetesPedigreeFunction** – Indicatore di familiarità
- **Age** – Età
- **Outcome** – 1 se la persona ha il diabete, 0 altrimenti

L'ultima colonna (Outcome) è quella che il modello deve imparare a prevedere.

3 Fase 1: Addestramento del modello

Il file `train_model.py` serve a creare e addestrare il modello.

3.1 Caricamento dei dati

```
df = pd.read_csv("diabetes.csv")
```

Qui leggiamo il file CSV che contiene il dataset e lo memorizziamo in una struttura dati chiamata DataFrame.

3.2 Separazione tra input e output

```
X = df[features]
y = df["Outcome"]
```

- `X` contiene le informazioni di ingresso (età, glucosio, ecc.)
- `y` contiene la risposta corretta (diabete sì/no)

3.3 Divisione in training e test

I dati vengono divisi in:

- Training set (80%) per insegnare al modello
- Test set (20%) per verificare se ha imparato correttamente

3.4 Standardizzazione e modello

La pipeline contiene due passaggi:

1. StandardScaler – normalizza i dati
2. LogisticRegression – modello matematico che stima una probabilità tra 0 e 1

3.5 Addestramento

Il modello osserva i dati di training e impara a collegare i valori clinici alla presenza o meno del diabete.

3.6 Salvataggio del modello

```
joblib.dump(pipeline, "modello_diabete.pkl")
```

Il modello viene salvato su file, così potrà essere riutilizzato nell'applicazione web senza doverlo riaddestrare ogni volta.

4 Che cos'è il file `.pkl`

Il file con estensione `.pkl` è un file in formato **Pickle**, un formato usato da Python per salvare oggetti su disco.

In questo progetto, l'oggetto salvato è l'intera pipeline di machine learning, che include:

- i parametri calcolati dallo StandardScaler
- i pesi appresi dal modello di regressione logistica

Salvare il modello in formato Pickle significa memorizzare su file lo stato interno del modello dopo l'addestramento. In questo modo:

joblib è una libreria Python usata per memorizzare su disco oggetti anche complessi, come modelli di machine learning che contengono molti parametri numerici.

dump è la funzione che effettua il salvataggio. In pratica "impacchetta" un oggetto Python e lo scrive in un file.

pipeline è l'oggetto che vogliamo salvare. Non è solo il modello di regressione lineare, ma tutta la pipeline, cioè:

lo StandardScaler con i parametri calcolati sui dati di training (media e deviazione standard),

il modello LinearRegression con i coefficienti che ha imparato.

- non è necessario riaddestrare il modello ogni volta che si avvia il programma;
- il modello può essere caricato rapidamente e usato per fare nuove previsioni.

Il caricamento avviene con:

```
model = joblib.load("modello_diabete.pkl")
```

A questo punto il modello è pronto per ricevere nuovi dati e calcolare la probabilità di diabete.

5 Fase 2: Applicazione Web con Flask

Il file `app.py` crea un piccolo server web che permette agli utenti di usare il modello tramite una pagina web.

Quando l'utente inserisce i dati:

1. i valori vengono inviati al server;
2. il server li passa al modello caricato dal file `.pkl`;
3. il modello restituisce una probabilità;
4. il risultato viene mostrato nella pagina.