Applicazione di Machine Learning con Regressione Lineare
Stima del Prezzo di una Casa

Esercitazione di Informatica

1 Obiettivo del progetto

In questa esercitazione realizziamo un'applicazione di intelligenza artificiale capace di stimare il **prezzo di una casa** a partire da alcune sue caratteristiche.

L'applicazione:

- utilizza un dataset reale di abitazioni;
- addestra un modello di **regressione lineare**;
- salva il modello su file;
- mette il modello a disposizione tramite una pagina web in cui l'utente inserisce i dati della casa.

A differenza degli esercizi di classificazione, qui il risultato non è una categoria, ma un **numero reale**: il prezzo stimato.

2 Che cos'è la regressione lineare

La regressione lineare è un metodo matematico che serve a descrivere la relazione tra più grandezze numeriche.

Nel nostro caso:

- le **variabili di ingresso** sono le caratteristiche della casa;
- la **variabile di uscita** è il prezzo.

Il modello cerca di trovare una formula del tipo:

$$prezzo = a_1x_1 + a_2x_2 + a_3x_3 + \ldots + b$$

dove:

- x_1, x_2, x_3 sono le caratteristiche della casa,
- a_1, a_2, a_3 sono numeri che il modello impara dai dati,
- b è un valore costante.

3 Il dataset

Il dataset contiene informazioni su diverse abitazioni. Ogni riga rappresenta una casa.
Le variabili usate sono:

- **RM** – Numero medio di stanze
- **LSTAT** – Percentuale di popolazione a basso reddito nella zona
- **PTRATIO** – Rapporto studenti/docenti nelle scuole locali
- **TAX** – Tassa immobiliare locale
- **AGE** – Percentuale di abitazioni costruite prima del 1940

La variabile da prevedere è:

- **MEDV** – Valore medio delle abitazioni (in migliaia di dollari)

4 Fase 1: Addestramento del modello

Il file `train_model_linear.py` si occupa di insegnare al modello come stimare il prezzo.

4.1 Caricamento dei dati

```
housing = fetch_openml(name="boston", version=1, as_frame=True)
df = housing.frame
```

I dati vengono caricati e salvati in una tabella.

4.2 Scelta delle variabili

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

- `X` contiene le caratteristiche delle case
- `y` contiene i prezzi reali

4.3 Divisione in training e test

Una parte dei dati serve per addestrare il modello, l'altra per verificarne la precisione.

4.4 Creazione del modello

```
pipeline = Pipeline([
    ("scaler", StandardScaler()),
    ("model", LinearRegression())
])
```

La pipeline esegue due operazioni:

1. Normalizza i dati (StandardScaler)
2. Applica la regressione lineare

4.5 Addestramento

```
pipeline.fit(X_train, y_train)
```

Il modello analizza gli esempi e impara i coefficienti della formula matematica che collega caratteristiche e prezzo.

4.6 Valutazione

Il modello viene testato confrontando i prezzi stimati con quelli reali.

5 Salvataggio del modello

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

Il file `.pkl` contiene il modello già addestrato, pronto per essere riutilizzato.

6 Fase 2: Applicazione Web

Il file `app.py` crea un piccolo server web con Flask.

Quando l'utente visita il sito, viene mostrata una pagina con un modulo dove inserire i dati della casa.

6.1 Predizione del prezzo

Quando l'utente invia il modulo:

1. I dati vengono inviati al server
2. Il server li passa al modello caricato dal file `.pkl`
3. Il modello calcola il prezzo stimato
4. Il risultato viene rimandato al browser

7 Interfaccia utente

La pagina HTML contiene campi di input numerici. Il file JavaScript raccoglie i valori e li invia al server usando una richiesta HTTP.

Il risultato finale è una stima automatica del prezzo della casa.

Che cos'è una Pipeline nel Machine Learning

Una **pipeline**, in ambito machine learning, è una sequenza ordinata di operazioni applicate ai dati, dove l'uscita di un passaggio diventa l'ingresso del successivo. Serve a rendere il processo di analisi ripetibile, ordinato e meno soggetto a errori.

Si può immaginare come una linea di montaggio: il "prodotto grezzo" sono i dati, e in ogni stazione avviene una trasformazione precisa fino ad arrivare al modello che produce la previsione finale.

Le fasi principali di una pipeline

In un problema di classificazione, una pipeline tipica comprende tre fasi fondamentali.

1. Preparazione dei dati

In questa fase si sistemano i dati prima di usarli nel modello. Alcune operazioni comuni sono:

- gestione dei valori mancanti;
- trasformazione di dati categorici (parole) in numeri;
- ridimensionamento delle variabili numeriche (scaling).

Questo passaggio è molto importante perché gli algoritmi di machine learning lavorano solo con numeri e possono essere influenzati da grandezze espresse su scale molto diverse.

2. Trasformazione dei dati

Qui i dati vengono modificati per renderli più adatti all'apprendimento del modello. Per esempio:

- normalizzazione dei valori;
- standardizzazione;
- creazione di nuove caratteristiche a partire da quelle esistenti.

Queste trasformazioni aiutano il modello a individuare meglio le relazioni presenti nei dati.

3. Modello di apprendimento

Nell'ultima fase entra in gioco l'algoritmo di machine learning vero e proprio. Il modello impara dai dati di addestramento e poi è in grado di fare previsioni su dati nuovi e mai visti prima.

Perché la pipeline è utile

Il vantaggio principale di una pipeline è che **automatizza tutto il processo**. Quando vengono forniti nuovi dati, la pipeline applica esattamente le stesse trasformazioni usate durante l'addestramento, prima di effettuare la previsione.

Questo evita errori molto comuni, come:

- dimenticare di applicare una trasformazione ai dati di test;
- usare trasformazioni diverse tra dati di addestramento e dati di verifica.

Aspetto pratico

Nelle principali librerie di machine learning, una pipeline è un oggetto che contiene in ordine tutti i passaggi. Quando si esegue l'addestramento, ogni fase calcola i propri parametri usando solo i dati di training. Quando si fanno previsioni, quegli stessi parametri vengono riutilizzati automaticamente sui nuovi dati.

In sintesi

Una pipeline è un modo strutturato e sicuro per passare dai dati grezzi alle previsioni di un modello. Riduce gli errori, rende il lavoro più ordinato e garantisce che il processo sia sempre ripetibile. In pratica, si definiscono le regole una sola volta e il sistema le applica correttamente ogni volta.