stringcolorred!70!black

Il mio primo intelligente: un dottore che impara dai dati

Prof. Fedeli Massimo

Introduzione

Immagina di avere un foglio Excel con le cartelle cliniche di 300 pazienti: età, pressione, colesterolo, ecc. Il nostro obiettivo è **insegnare al computer a riconoscere** se un nuovo paziente è sano o a rischio di infarto, **senza scrivergli regole**, ma facendogli **imparare** dai casi passati.

1. I dati: il "libro" delle cartelle cliniche

Scarichiamo un file dal web che contiene 14 colonne:

age	età (anni)
sex	sesso (0 = donna, 1 = uomo)
cp	tipo di dolore al petto
trestbps	pressione a riposo
chol	colesterolo
fbs	glicemia a digiuno
...	...
target	grado di malattia (0 = sano, 1-4 = malato)

Alcune celle hanno il simbolo "?" al posto del numero: sono **dati mancanti**. Il computer non può lavorare con i buchi, quindi **cancelliamo le righe incomplete**:

```
df = df.dropna()
```

2. Trasformiamo il problema in "sì o no"

Nel file originale ci sono 5 livelli di malattia. Noi vogliamo solo due risposte:

- 0 = **sano**
- 1 = **malato**

Convertiamo quindi tutti i valori 1-4 in 1:

```
df['target'] = (df['target'] > 0).astype(int)
```

3. Dividiamo la classe e gli esercizi

- **X** = tutte le colonne tranne "target" (le *feature*)
- **y** = solo la colonna "target" (la *classe*)

Ora spacchiamo il foglio in due:

- **Training set** (80 %) = foglio su cui il computer **studia**
- **Test set** (20 %) = foglio su cui **interroghiamo** il computer

```
X_train, X_test, y_train, y_test = train_test_split(
X, y, test_size=0.2, random_state=42, stratify=y)
```

L'opzione `stratify` fa sì che in entrambi i fogli ci sia la stessa percentuale di sani e malati.

4. Standardizzare: mettere tutti "sulla stessa riga"

Le colonne hanno unità diverse: età (0-100), colesterolo (100-600), ecc. Per non far vincere il "più grande", **centriamo e rimpiccioliamo** tutti i numeri:

- media = 0

- deviazione standard = 1

```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
    # impara la media e la scala
X_test  = scaler.transform(X_test)     #
    applica la stessa trasformazione
```

Importante: impariamo la media e la scala **solo sul training**, altrimenti "bariamo".

5. L'apprendimento: il computer fa i compiti

Usiamo un algoritmo chiamato **Regressione Logistica**. Non è una retta, ma una **curva a S** che restituisce una **probabilità**:

$$P(\text{malato}) = \sigma(w_1 x_1 + w_2 x_2 + \cdots + w_n x_n)$$

σ è la funzione logistica; w_i sono i "pesi" che il computer aggiusta per avvicinarsi alla risposta corretta.

```
model = LogisticRegression(max_iter
   =1000)
model.fit(X_train, y_train)
```

6. L'interrogazione: quanto è bravo?

Facciamo rispondere il computer sul foglio che **non ha mai visto**:

```
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred
   )
```

L'accuratezza è la percentuale di risposte esatte. Un valore tipico con questi dati è $\approx 85\,\%$. Non è perfetto, ma è **meglio di un lancio di moneta**!

7. Prova con un nuovo paziente

Creiamo un paziente immaginario:

```
paziente = {'age':60, 'sex':1, 'cp':0, 'trestbps':150,
'chol':250, 'fbs':1, 'restecg':1, 'thalach':130,
'exang':1, 'oldpeak':2.3, 'slope':1, 'ca':1, 'thal':3}
```

Lo standardizziamo con la stessa "scheda" usata prima e chiediamo la previsione:

```
paziente_scaled = scaler.transform(pd.DataFrame([paziente]))
rischio = model.predict_proba(paziente_scaled)[0][1]
```

Il computer dice: "C'è un 73 % di probabilità che questo paziente sia a rischio."

8. Cosa abbiamo imparato?

- Il computer **non** ha bisogno di regole scritte a mano: impara *da solo* dai dati.
- I dati vanno **puliti** (no buchi) e **preparati** (scale simili).
- **Mai valutare** sullo stesso foglio su cui si è studiato: si "barerebbe".
- L'accuratezza è una **prima misura**, ma in medicina servono anche altri controlli (sensibilità, specificità, ecc.).

9. Provaci tu!

1. Cambia il paziente di esempio: più giovane, donna, colesterolo basso...
2. Osserva come cambia la probabilità.
3. Prova a modificare `test_size` o `random_state`: l'accuratezza cambia?

Link utili

`https://archive.ics.uci.edu/ml/datasets/heart+Disease` (dataset)
`https://scikit-learn.org` (documentazione di scikit-learn)