Lezione 6: Oltre la Linearità

Modellare il Diodo e la Regressione Polinomiale

Laboratorio di Sistemi e Automazione

1 Il Problema: Quando Ohm non basta

Fino ad ora abbiamo analizzato resistori, che seguono la Legge di Ohm ($V = R \cdot I$). La relazione è una **linea retta**. Ma in elettronica, i componenti più interessanti sono **Non-Lineari**.

Oggi analizzeremo un **Diodo**. La sua corrente non cresce in modo proporzionale alla tensione, ma segue l'equazione esponenziale di Shockley:

$$I = I_S \cdot (e^{\frac{V}{nV_T}} - 1)$$

Dove:

- I_S: Corrente di saturazione inversa (piccolissima).
- $V_T \approx 26mV$: Tensione termica a temperatura ambiente ($300K$).
- $n = 1$: Fattore di idealità.

Se provassimo a usare la `LinearRegression` (una retta) su un diodo, otterremmo un disastro.

2 L'Esperimento Simulato

Immaginiamo di raccogliere dati in laboratorio su un diodo al silicio. Per tensioni basse ($< 0.5V$) non passa quasi nulla. Poi, improvvisamente, la corrente schizza verso l'alto (il "ginocchio").

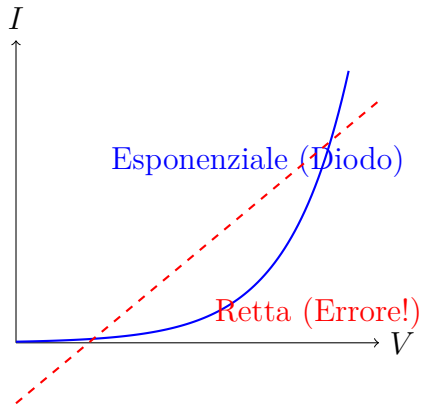

Fig. 1: Una linea retta non può approssimare una curva esponenziale.

3 La Soluzione: Regressione Polinomiale

Come facciamo a insegnare alla macchina a "curvare"? Usiamo un trucco matematico chiamato **Polynomial Features**.

Invece di dare alla macchina solo la tensione V, le diamo anche le sue potenze:

$$Input = [V, V^2, V^3, ...]$$

È come dire alla macchina: "Non cercare solo una retta, cerca una curva che combina parabole (x^2) e cubiche (x^3)". In Python useremo lo strumento `PolynomialFeatures`.

4 Implementazione in Python

4.1 1. Generazione Dati (Il Diodo)

Simuliamo il comportamento fisico del diodo usando i parametri reali ($V_T = 26mV$).

```python
import numpy as np
import matplotlib.pyplot as plt

# Dati di Tensione (Input) da 0V a 0.8V
V = np.linspace(0, 0.8, 20).reshape(-1, 1)

# Parametri fisici
Is = 1e-12  # 1 picoAmpere
Vt = 0.026  # 26 milliVolt
n = 1

# Equazione di Shockley (Output Corrente) + un po' di rumore reale
I = Is * (np.exp(V / (n * Vt)) - 1)

# Aggiungiamo rumore casuale (errori di misura)
I_noise = I + np.random.normal(0, 0.0005, size=I.shape)
```

Listing 1: Simulazione Dati Diodo

4.2 2. La Magia: Trasformazione Polinomiale

Qui avviene il salto di qualità. Trasformiamo l'input semplice in un input "potenziato" di grado 5.

```python
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.pipeline import make_pipeline

# Creiamo un modello "Intelligente" che prima eleva alla potenza (grado 5)
# e POI applica la regressione lineare su quei dati curvi.
grado = 5
modello_diodo = make_pipeline(PolynomialFeatures(grado), LinearRegression())

# Addestriamo il modello sui dati rumorosi
modello_diodo.fit(V, I_noise)

# Facciamo una predizione (disegniamo la curva appresa)
I_predetto = modello_diodo.predict(V)
```

Listing 2: Creazione del Modello Polinomiale

5 Visualizzazione: Retta vs Curva

Nel notebook vedrete il confronto. La retta ignorerà completamente la fisica del diodo. Il modello polinomiale si adagerà perfettamente sulla curva esponenziale.

```
plt.scatter(V, I_noise, color='black', label='Dati Lab')
plt.plot(V, I_predetto, color='red', linewidth=2, label=f'Modello Polinomiale (Grado {
    grado})')

plt.title("Caratteristica I-V del Diodo")
plt.xlabel("Tensione V [Volt]")
plt.ylabel("Corrente I [Ampere]")
plt.legend()
plt.grid(True)
plt.show()
```

Listing 3: Grafico Finale

6 Analisi Critica: Le "Oscillazioni Fantasma"

Osservate attentamente la linea rossa nel grafico generato, specialmente nella zona tra $0V$ e $0.6V$ (dove il diodo dovrebbe essere spento). Notate qualcosa di strano?

- La linea non è piatta a zero.
- Fa delle piccole onde (oscillazioni).
- A volte scende addirittura sotto lo zero (Corrente negativa?!).

6.1 Perché succede?

In Elettronica sappiamo che sotto la soglia $I \approx 0$. Ma la macchina **non sa cos'è un diodo**. La macchina sta solo cercando di piegare una curva matematica (un polinomio) per unire i punti. Per riuscire a fare quella curva ripida finale (il "ginocchio"), il polinomio "prende la rincorsa" e oscilla nella parte iniziale.

7 Conclusione: Il Ruolo dell'Ingegnere

Abbiamo scoperto un limite fondamentale dell'AI pura:

La Matematica non conosce la Fisica

L'algoritmo ci propone valori di corrente negativi o oscillanti prima della soglia di accensione. Un Data Scientist puro potrebbe accettarli, ma un **Elettronico sa che sono impossibili**.

Tocca a noi intervenire. Dobbiamo "filtrare" il modello matematico imponendo le nostre conoscenze fisiche (Logic-Constraint): *"Se la tensione è sotto la soglia, la corrente DEVE essere zero, qualunque cosa dica il polinomio".*

Nella prossima lezione impareremo a costruire questo **Modello Ibrido**.

→ LABORATORIO AVANZATO

Vedi le oscillazioni con i tuoi occhi su Colab:

[APRI SIMULAZIONE DIODO]

→ PROSSIMO STEP

Dobbiamo correggere gli errori della macchina.
Andiamo a implementare il "Filtro Fisico" nel codice.

Vai alla Lezione 7: Il Modello Ibrido (Fisica + AI)