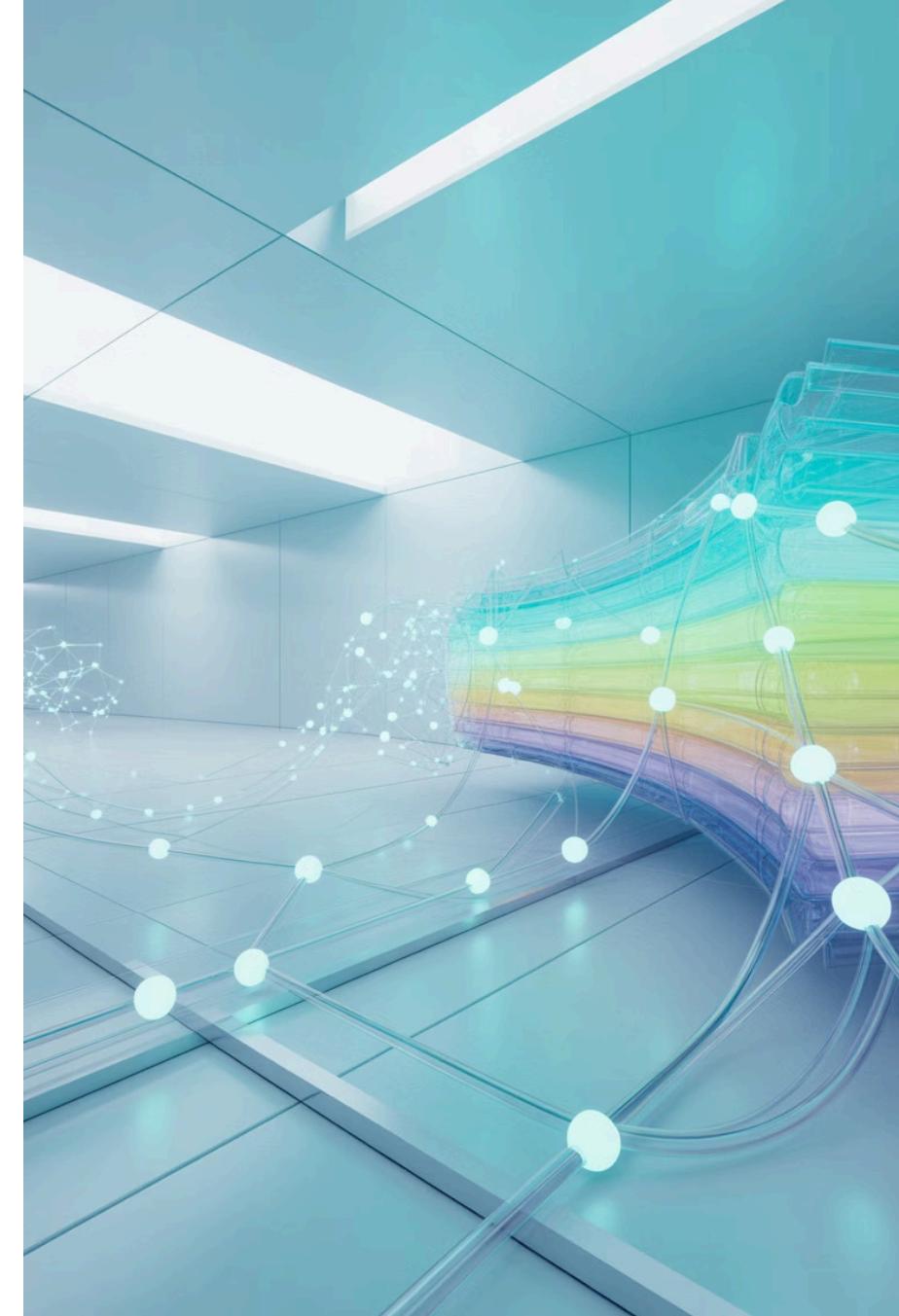


Classificatore Naive Bayes

Un approccio probabilistico potente e sorprendentemente semplice per il Machine Learning



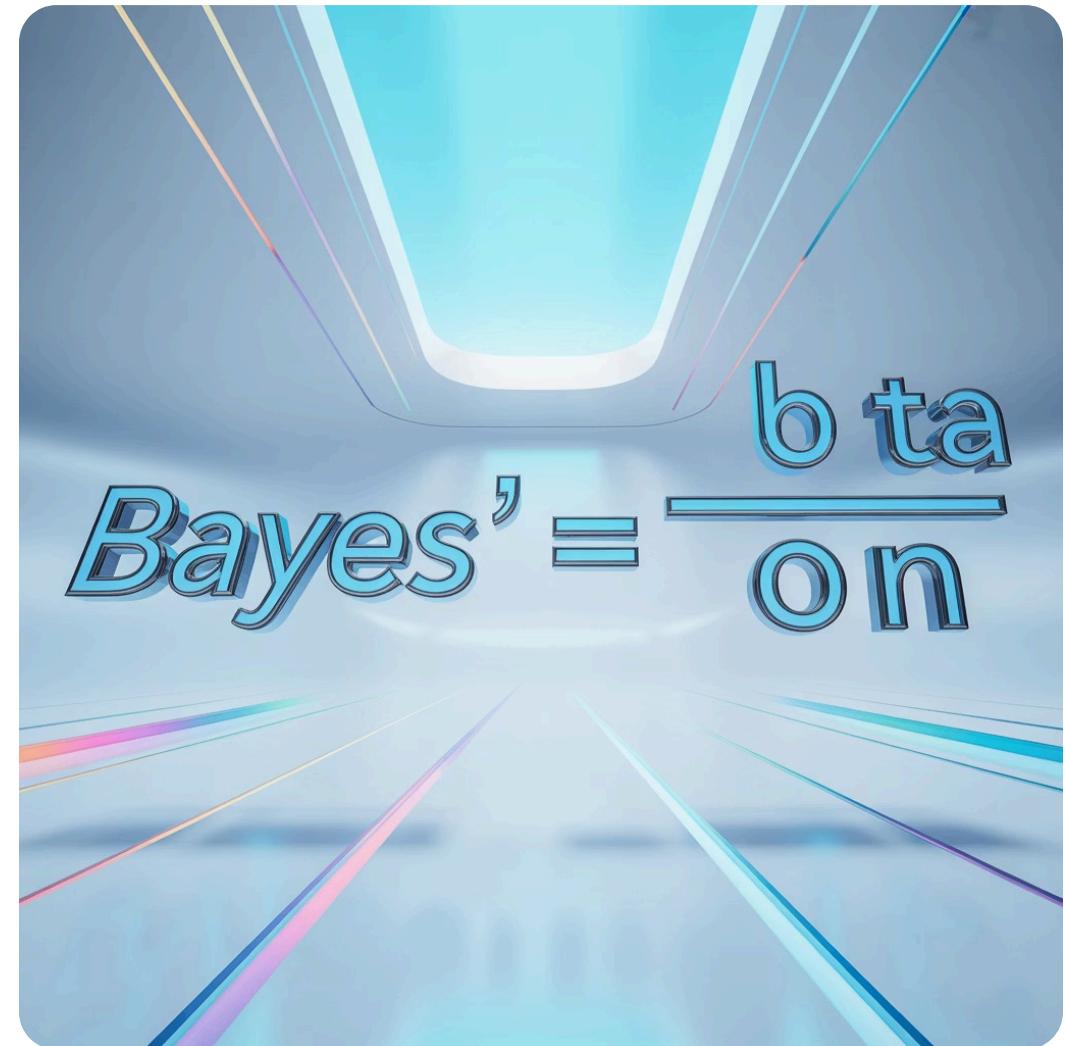
Cos'è un Classificatore Naive Bayes?

Un classificatore Naive Bayes è un modello di Machine Learning supervisionato che utilizza il **Teorema di Bayes** per fare predizioni. Il suo obiettivo è stimare la probabilità che un'osservazione appartenga a una determinata classe.

La formula fondamentale che il modello cerca di calcolare è:

$$P(\text{Classe} \mid \text{Osservazioni})$$

Questo modello si distingue per tre caratteristiche principali: è **velocissimo** da addestrare, è **concettualmente semplice** da comprendere, ed è **sorprendentemente efficace** in molte applicazioni reali.



Comprendere $P(\text{Classe} | \text{Osservazioni})$

Il Significato

Rappresenta la probabilità che qualcosa appartenga a una certa categoria (classe), dato quello che abbiamo osservato.

L'Idea Fondamentale

È il concetto cardine del Naive Bayes: combinare le evidenze disponibili per stimare la probabilità di appartenenza a una classe.

L'Applicazione

Questo approccio permette di trasformare osservazioni concrete in decisioni probabilistiche informate.

Esempio Pratico: Previsione della Pioggia

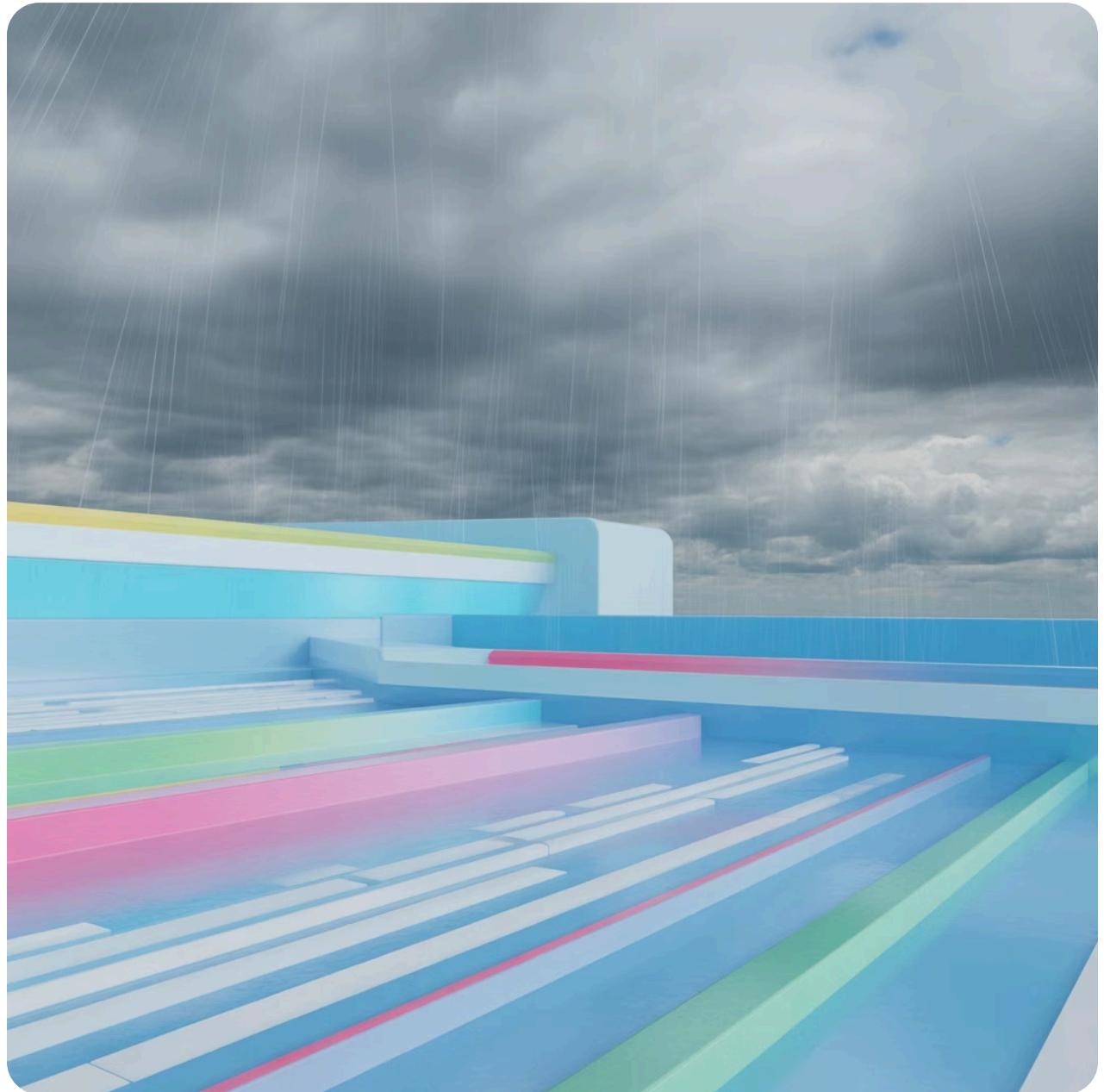
Immagina di voler capire se **oggi pioverà**. Hai alcune osservazioni dal mondo reale che possono aiutarti a fare questa predizione.

🔍 Le Classi

- **Pioggia**
- **Non pioggia**

🔍 Le Osservazioni

- Cielo grigio
- Umidità alta
- Presenza di vento



La domanda diventa quindi: *Qual è la probabilità che oggi piova, dato che vedo cielo grigio, umidità alta e vento?*

$$P(\text{Pioggia} \mid \text{Cielo grigio, umidità alta, vento})$$

Calcolo delle Probabilità

Cielo Grigio

Quando il cielo è grigio, piove il **70%** delle volte

Umidità Alta

Quando l'umidità è alta, piove il **60%** delle volte

Vento Presente

Quando c'è vento, piove il **30%** delle volte

Il Naive Bayes combina queste probabilità individuali (assumendo indipendenza tra le feature) per calcolare la probabilità finale. Confronta poi $P(\text{Pioggia} \mid \text{osservazioni})$ con $P(\text{Non pioggia} \mid \text{osservazioni})$ e sceglie la classe con probabilità più alta.

Altro Esempio: Il Medico Intelligente

Le Classi Diagnostiche

- **Malato**
- **Sano**

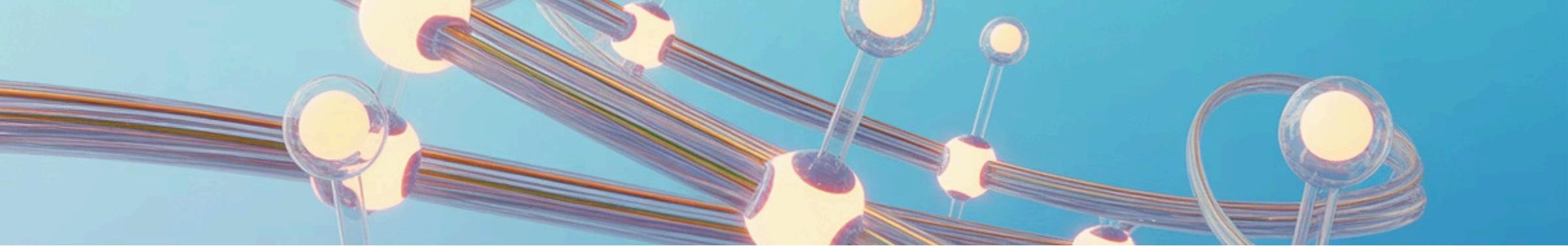
I Sintomi Osservati

- Febbre
- Tosse
- Mal di gola

$P(\text{Malato} \mid \text{febbre, tosse, mal di gola})$



Il modello risponde alla domanda: *Qual è la probabilità che tu sia malato **dati i sintomi** che ho osservato?* Questo è esattamente il tipo di ragionamento probabilistico che i medici fanno quotidianamente, ora automatizzato.



Perché Si Chiama "Naive"?

Il termine "naive" (ingenuo) deriva da un'assunzione **molto forte e quasi sempre falsa** che il modello fa:

Assume che tutte le feature siano indipendenti tra loro data la classe.

$$P(x_1, x_2, \dots, x_n | C) = \prod_i P(x_i | C)$$

L'Assunzione di Indipendenza nella Realtà

Testi

Le parole nei testi **non sono indipendenti**
- alcune parole tendono a comparire
insieme

Sintomi

I sintomi di una malattia **non sono indipendenti** - spesso si manifestano in combinazioni specifiche

Immagini

I pixel di un'immagine **non sono indipendenti** - i pixel vicini sono altamente correlati

- ❑ **👉 Nonostante questa assunzione irrealistica, il modello funziona molto bene!** Questo è uno dei motivi per cui Naive Bayes è così diffuso nel mondo reale.

Come Funziona: Fase di Training



Raccolta Dati

Il modello analizza un dataset di esempi già classificati per apprendere i pattern



Calcolo $P(C)$

Determina la probabilità di ogni classe (es. probabilità che un'email sia spam vs non spam)



Calcolo $P(\text{feature} | C)$

Calcola quanto ogni feature è probabile dentro una classe (es. probabilità che "free" appaia in uno spam)



Smoothing

Applica Laplace smoothing per evitare probabilità zero che invaliderebbero le predizioni

Come Funziona: Fase di Predizione

Per una nuova osservazione $x = (x_1, x_2, \dots, x_n)$, il modello esegue questi passaggi:



Calcolo

Calcola la probabilità per ogni classe usando la formula di Bayes

$$P(C | x) \propto P(C) \cdot \prod_i P(x_i | C)$$

Confronto

Confronta le probabilità calcolate per tutte le classi disponibili

Decisione

Sceglie la classe con la probabilità più alta come predizione finale

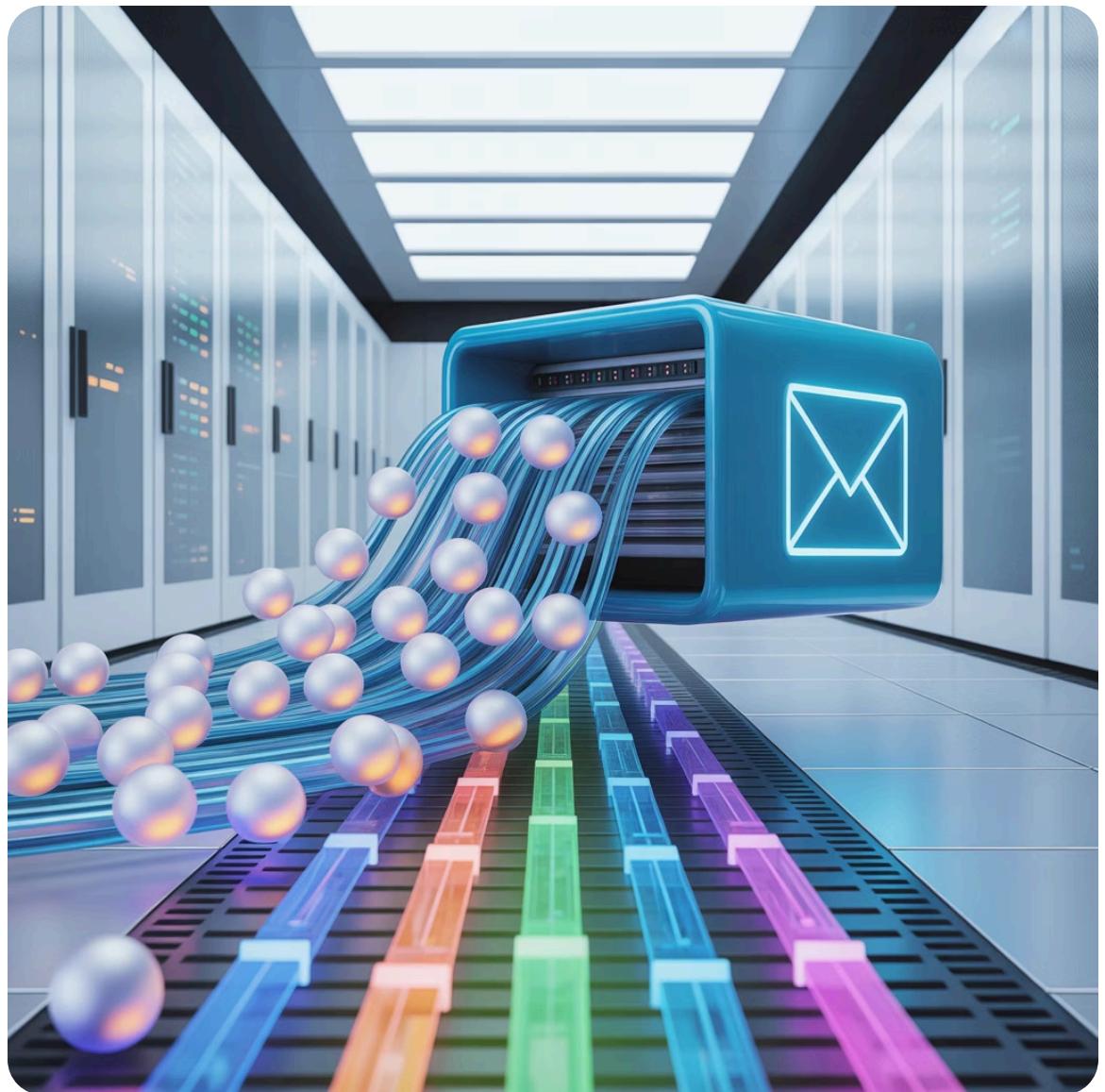
Esempio Completo: Filtro Spam

Nuova Email Ricevuta

"FREE money now!!"

Dati dal Training

- $P(\text{Spam}) = 0.2$
- $P(\text{Not Spam}) = 0.8$
- $P(\text{"free"} \mid \text{Spam}) = 0.6$
- $P(\text{"free"} \mid \text{Not Spam}) = 0.02$
- $P(\text{"money"} \mid \text{Spam}) = 0.4$
- $P(\text{"money"} \mid \text{Not Spam}) = 0.01$



Calcolo delle Probabilità Spam

Probabilità SPAM

$$P(\text{Spam} \mid \text{email}) \propto 0.2 \cdot 0.6 \cdot 0.4$$

Risultato: ≈ 0.048

Probabilità NON SPAM

$$P(\text{NotSpam} \mid \text{email}) \propto 0.8 \cdot 0.02 \cdot 0.01$$

Risultato: ≈ 0.00016

Il classificatore confronta i due valori: 0.048 è **molto più grande** di 0.00016, quindi l'email viene classificata come **SPAM**. La differenza è così netta che non ci sono dubbi sulla classificazione corretta.

Perché Funziona Così Bene?



Compensazione delle Dipendenze

Anche se l'indipendenza tra feature non è vera, le dipendenze reali spesso "si compensano" a vicenda. Il modello riesce comunque a catturare il segnale globale importante per la classificazione.



Linearità nel Log-Spazio

Il modello è "lineare nel log-spazio", cioè lavora con somme di log-probabilità invece che prodotti. Questo lo rende stabile numericamente ed estremamente efficiente computazionalmente.



Efficienza con Testi

Con grandi dataset testuali (email, recensioni, documenti) fa miracoli perché è molto veloce, gestisce facilmente migliaia di feature e non soffre del problema della dimensionalità.

Applicazioni nel Mondo Reale



Spam Filtering

Gmail, Outlook e altri servizi email



Classificazione Testi

Categorizzazione automatica di documenti



Sentiment Analysis

Analisi di opinioni e recensioni



Diagnosi Mediche

Supporto decisionale probabilistico



Riconoscimento Linguistico

Identificazione automatica della lingua



Sistemi di Raccomandazione

Suggerimenti personalizzati veloci

Esempio Pratico: Sentiment Analysis

Vediamo come il classificatore Naive Bayes può essere usato per la **sentiment analysis**, ovvero determinare se un testo esprime un sentimento positivo o negativo. Immagina di voler classificare le recensioni dei film.

🎬 Nuova Recensione

"Questo film è fantastico!"

📊 Dati dal Training

- $P(\text{Positivo}) = 0.5$
- $P(\text{Negativo}) = 0.5$
- $P(\text{"film"} | \text{Positivo}) = 0.08$
- $P(\text{"film"} | \text{Negativo}) = 0.07$
- $P(\text{"fantastico"} | \text{Positivo}) = 0.03$
- $P(\text{"fantastico"} | \text{Negativo}) = 0.001$



Il nostro obiettivo è calcolare quale sia più probabile: che la recensione sia positiva data le parole, o negativa date le parole. Assumiamo che le parole "Questo" e "è" non siano distinctive per il sentimento e le ignoriamo per questo esempio semplificato.

Esempio Pratico: Riconoscimento Linguistico

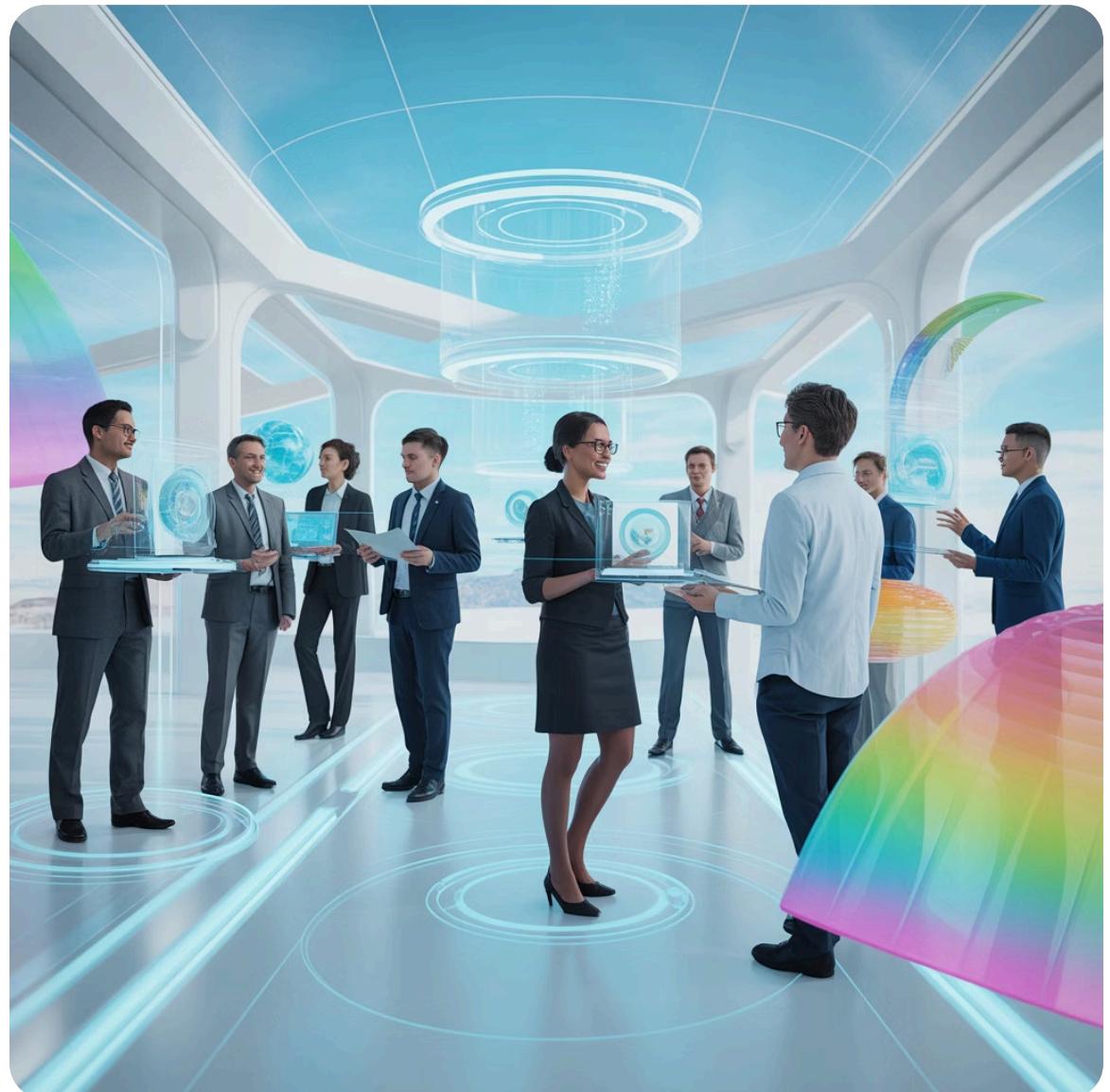
Immagina di ricevere un breve testo e voler sapere in quale lingua è scritto. Il classificatore Naive Bayes eccelle in questo compito, analizzando le parole e le loro probabilità di apparire in lingue diverse.

Nuovo Testo

"Un buon giorno per imparare"

Dati dal Training (Semplificati)

- $P(\text{Italiano}) = 0.5$
- $P(\text{Inglese}) = 0.5$
- $P(\text{"buon"} | \text{Italiano}) = 0.05$
- $P(\text{"buon"} | \text{Inglese}) = 0.0001$
- $P(\text{"giorno"} | \text{Italiano}) = 0.04$
- $P(\text{"giorno"} | \text{Inglese}) = 0.0001$
- $P(\text{"imparare"} | \text{Italiano}) = 0.03$
- $P(\text{"imparare"} | \text{Inglese}) = 0.0001$



Il modello calcolerà la probabilità che il testo sia italiano o inglese dati i termini "buon", "giorno" e "imparare". Le parole comuni come "un" e "per" (stopwords) vengono spesso ignorate perché non portano informazioni distintive sulla lingua.

Riepilogo dei Concetti Chiave



È Probabilistico ma Semplice

Usa il Teorema di Bayes per combinare evidenze in modo matematicamente rigoroso ma concettualmente chiaro



È Sorprendentemente Potente

Soprattutto per dati testuali, dove eccelle nonostante (o forse grazie a) la sua semplicità



È "Naive" per un Motivo

Finge che le feature siano indipendenti, un'assunzione quasi sempre falsa ma sorprendentemente efficace



È Velocissimo da Addestrare

Uno dei modelli più rapidi e facili da implementare, ideale per dataset di grandi dimensioni

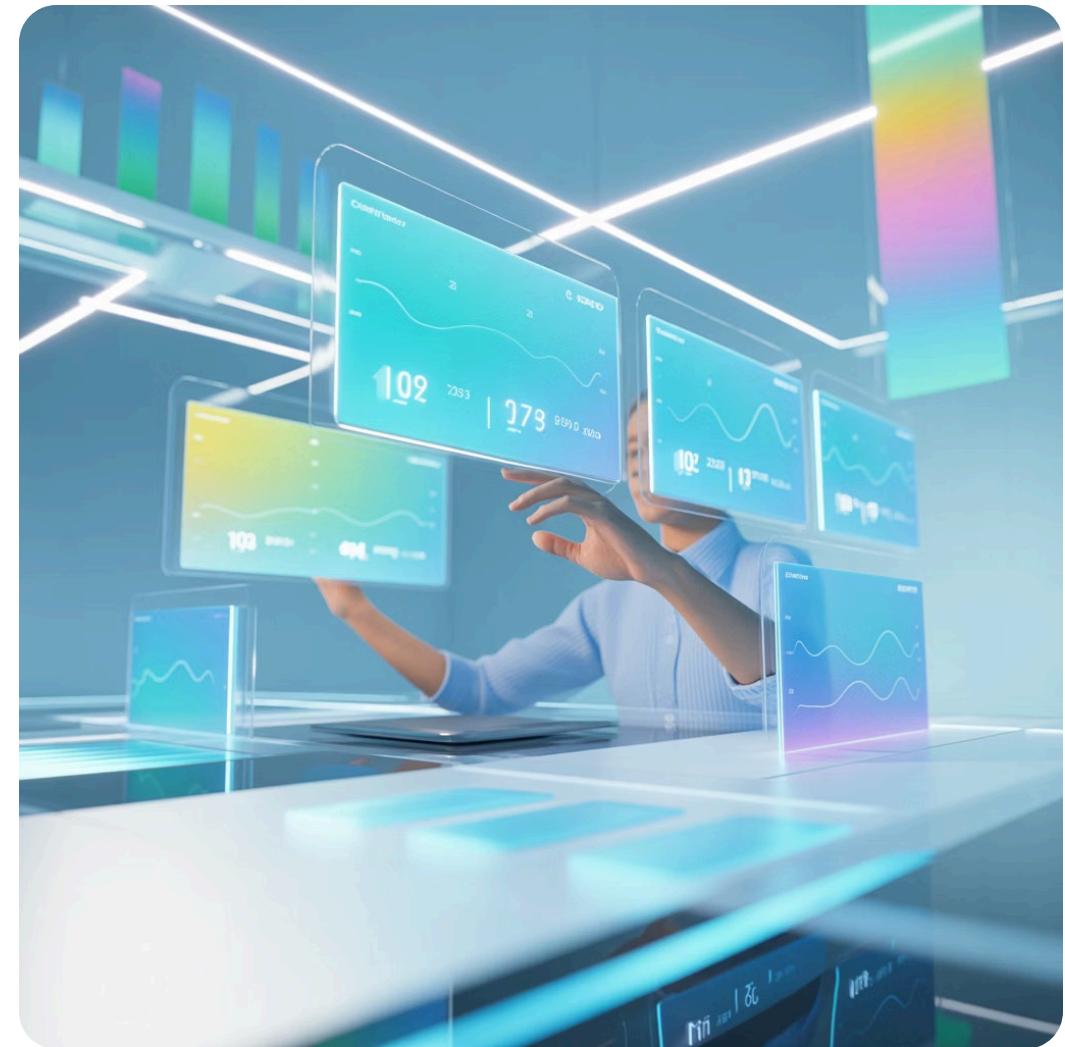
Il Training: Contare le Frequenze

Durante il training, un classificatore Naive Bayes esegue un'operazione fondamentale: **conta le cose**. Letteralmente.

Cosa Conta il Modello?

Per ogni classe, il modello conta quante volte compare ogni feature nel dataset di training. Ad esempio, in un filtro spam:

- Quante volte appare "free" nelle email spam
- Quante volte appare "free" nelle email non spam
- Quante volte appare "money" in ciascuna classe
- E così via per tutte le parole del vocabolario



È semplicemente **conteggio di frequenze** - niente di più complesso di questo nella fase di apprendimento.

Esempio: Tabella delle Frequenze

Conteggio delle Parole nel Dataset di Training

Parola	Contata in SPAM	Contata in NON SPAM
free	30	1
money	20	0
hello	2	50

La formula per calcolare la probabilità di una parola data una classe è:

$$P(x_i | C) = \frac{\text{count}(x_i, C) + 1}{\text{totale parole in } C + V}$$

dove **+1** è lo smoothing e **V** è il numero totale di parole uniche (vocabolario).

Il Problema dello Zero

⚠ Cosa Succede Senza Smoothing?

Supponiamo che nella classe NON SPAM, la parola "money" appaia **0 volte**. Senza l'applicazione dello smoothing, la sua probabilità sarebbe:

$$P(\text{money} \mid \text{NonSpam}) = \frac{\text{count}(\text{money}, \text{NonSpam})}{\text{totale parole in NonSpam}} = \frac{0}{\text{totale}} = 0$$

Questo diventa un problema critico, noto come "Problema dello Zero", perché se una singola parola all'interno di un documento ha una probabilità di 0 per una data classe (anche se il documento nel suo complesso potrebbe appartenere a quella classe), l'intera probabilità del documento per quella classe diventa 0. Matematicamente, questo si manifesta così:

$$P(\text{Documento} \mid \text{NonSpam}) = P(\text{parola}_1 \mid \text{NonSpam}) \times P(\text{parola}_2 \mid \text{NonSpam}) \times \cdots \times P(\text{money} \mid \text{NonSpam}) \times \cdots = \\ \text{QualsiasiValore} \times 0 \times \text{QualsiasiValore} = 0$$

Un prodotto di probabilità che include un termine zero annulla tutte le altre probabilità, rendendo impossibile distinguere tra diverse classi se anche solo una parola non è stata vista nel training set per quella classe. Per evitare ciò, è essenziale utilizzare tecniche di smoothing, come lo smoothing additivo (+1), per assegnare una probabilità non nulla anche a parole mai viste.

Come Funziona il Laplace Smoothing

01

Problema Identificato

Se una feature non compare mai in una classe durante il training, la sua probabilità diventa zero

02

Conseguenza Grave

Questo zero, moltiplicato per tutte le altre probabilità, annulla completamente la probabilità della classe

03

Aggiunta del +1

Lo smoothing aggiunge "1 fittizio" al conteggio di ogni feature, garantendo che nessuna probabilità sia mai esattamente zero

04

Normalizzazione

Il denominatore viene incrementato di V (dimensione vocabolario) per mantenere le probabilità valide e normalizzate

05

Risultato Finale

La probabilità diventa piccola ma esistente, permettendo al modello di fare predizioni corrette anche con feature rare



Riassunto Finale: I Punti Essenziali

Apprendimento per Conteggio

- 1 Naive Bayes impara semplicemente **contando le frequenze** con cui ogni feature appare nelle diverse classi del dataset di training

Il Problema dello Zero

- 2 Se una feature non compare mai in una classe, senza smoothing la probabilità diventa zero e quella classe viene ingiustamente eliminata dalle predizioni

La Soluzione Elegante

- 3 **Laplace smoothing** risolve il problema aggiungendo "1 fittizio" a ogni conteggio, garantendo che nessuna probabilità sia mai esattamente zero

Importanza Pratica

- 4 Questa tecnica è **fondamentale** nei problemi di classificazione testuale, dove il vocabolario può contenere migliaia o milioni di parole diverse