Naive Bayes Example

Suppose we have a dataset about whether someone plays tennis based on **Outlook** and **Temperature**:

Outlook	Temperature	PlayTennis
Sunny	Hot	No
Sunny	Hot	No
Overcast	Hot	Yes
Rain	Mild	Yes
Rain	Cool	Yes

We want to predict **PlayTennis** for **Outlook = Sunny, Temperature = Mild**.

Step 1: Compute prior probabilities

$$P(Yes) = \frac{3}{5} = 0.6$$

$$P(No) = \frac{2}{5} = 0.4$$

Step 2: Compute likelihoods

For PlayTennis = Yes:

$$P(Outlook = Sunny|Yes) = \frac{0}{3} = 0$$

$$P(Temperature = Mild|Yes) = \frac{1}{3} \approx 0.33$$

For PlayTennis = No:

$$P(Outlook = Sunny|No) = \frac{2}{2} = 1$$

$$P(Temperature = Mild|No) = \frac{0}{2} = 0$$

To avoid zero probability, we can use **Laplace smoothing**.

Step 3: Compute posterior probabilities

$$P(Yes|Sunny, Mild) \propto P(Yes) \cdot P(Sunny|Yes) \cdot P(Mild|Yes) = 0.6 \cdot 0 \cdot 0.33 = 0$$

$$P(No|Sunny, Mild) \propto 0.4 \cdot 1 \cdot 0 = 0$$

> With Laplace smoothing (adding 1 to each count):

$$P(Sunny|Yes) = \frac{0+1}{3+3} = 1/6 \approx 0.167$$

$$P(Mild|Yes) = \frac{1+1}{3+3} = 2/6 = 0.333$$

$$P(Yes|Sunny, Mild) \propto 0.6 \cdot 0.167 \cdot 0.333 \approx 0.033$$

$$P(No|Sunny, Mild) \propto 0.4 \cdot \frac{2+1}{2+3} \cdot \frac{0+1}{2+3} = 0.4 \cdot 0.6 \cdot 0.2 = 0.048$$

Prediction: No, because 0.048 > 0.033.

Example: Predicting if a student will pass an exam based on study time and sleep

StudyTime	Sleep	Pass
High	Enough	Yes
High	Little	Yes
Medium	Enough	Yes
Low	Enough	No
Low	Little	No

We want to predict **Pass** for a student with **StudyTime = Medium** and **Sleep = Little**.

Step 1: Compute prior probabilities

- Total examples: 5
- Count of **Yes**: 3
- Count of **No**: 2

$$P(Yes) = \frac{3}{5} = 0.6$$

$$P(No) = \frac{2}{5} = 0.4$$

Step 2: Compute likelihoods

For Pass = Yes:

- $P(StudyTime = Medium|Yes) = \frac{1}{3} \approx 0.333$
- $P(Sleep = Little|Yes) = \frac{1}{3} \approx 0.333$

For Pass = No:

- $P(StudyTime = Medium|No) = \frac{0}{2} = 0$
- $P(Sleep = Little|No) = \frac{1}{2} = 0.5$

> To avoid zero probability, use **Laplace smoothing** (add 1 to each count and add number of categories to denominator).

Step 3: Compute posterior probabilities with Laplace smoothing

Yes class:

$$P(StudyTime = Medium|Yes) = \frac{1+1}{3+3} = \frac{2}{6} = 0.333$$

$$P(Sleep = Little|Yes) = \frac{1+1}{3+2} = \frac{2}{5} = 0.4$$

$$P(Yes|Medium, Little) \propto 0.6 \cdot 0.333 \cdot 0.4 \approx 0.08$$

No class:

$$P(StudyTime = Medium|No) = \frac{0+1}{2+3} = \frac{1}{5} = 0.2$$

$$P(Sleep = Little|No) = \frac{1+1}{2+2} = \frac{2}{4} = 0.5$$

$$P(No|Medium, Little) \propto 0.4 \cdot 0.2 \cdot 0.5 = 0.04$$

-Step 4: Predict class

- $P(Yes|Medium, Little) = 0.08$
- $P(No|Medium, Little) = 0.04$

Prediction: Yes, the student will pass.

KNN Example (Euclidean distance)

Suppose we have points in 2D space:

Point	X	Y	Class
A	1	2	Red
B	2	3	Red
C	3	3	Blue
D	6	5	Blue

We want to classify **Point P = (2,2)** using **k = 3**.

Step 1: Compute Euclidean distances

$$d(P, A) = \sqrt{(2-1)^2 + (2-2)^2} = \sqrt{1+0} = 1$$

$$d(P, B) = \sqrt{(2-2)^2 + (2-3)^2} = \sqrt{0+1} = 1$$

$$d(P, C) = \sqrt{(2-3)^2 + (2-3)^2} = \sqrt{1+1} = \sqrt{2} \approx 1.414$$

$$d(P, D) = \sqrt{(2-6)^2 + (2-5)^2} = \sqrt{16+9} = \sqrt{25} = 5$$

Step 2: Find k nearest neighbors

- k = 3 → nearest points: **A (Red), B (Red), C (Blue)**

Step 3: Majority vote

- Red: 2
- Blue: 1

Prediction: Red

Decision Tree: Predict if a candidate will get a promotion

Candidate	Experience	Performance	Promotion
1	High	Excellent	Yes
2	Low	Excellent	No
3	Medium	Good	Yes
4	Low	Good	No
5	High	Good	Yes

Target: **Promotion (Yes/No)**
Features: **Experience**, **Performance**

Step 1: Compute total entropy

- Total candidates = 5
- Yes = 3, No = 2

$$Entropy(S) = -\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5} \approx 0.971$$

Step 2: Compute entropy for each feature

Feature: Experience

Experience = High → 2 samples, Promotion = Yes:2, No:0

$$Entropy(High) = -1\log_2 1 - 0\log_2 0 = 0$$

Experience = Medium → 1 sample, Promotion = Yes:1, No:0

$$Entropy(Medium) = 0$$

Experience = Low → 2 samples, Promotion = Yes:0, No:2

$$Entropy(Low) = -0\log_2 0 - 1\log_2 1 = 0$$

Weighted Entropy for Experience:

$$E(Experience) = \frac{2}{5} \cdot 0 + \frac{1}{5} \cdot 0 + \frac{2}{5} \cdot 0 = 0$$

Information Gain for Experience:

$$IG(Experience) = Entropy(S) - E(Experience) = 0.971 - 0 = 0.971$$

Feature: Performance

Performance = Excellent → 2 samples, Promotion = Yes:1, No:1

$$Entropy(Excellent) = -0.5\log_2 0.5 - 0.5\log_2 0.5 = 1$$

Performance = Good → 3 samples, Promotion = Yes:2, No:1

$$Entropy(Good) = -\frac{2}{3}\log_2\frac{2}{3} - \frac{1}{3}\log_2\frac{1}{3} \approx 0.918$$

Weighted Entropy for Performance:

$$E(Performance) = \frac{2}{5} \cdot 1 + \frac{3}{5} \cdot 0.918 \approx 0.950$$

Information Gain for Performance:

$$IG(Performance) = 0.971 - 0.950 = 0.021$$

Step 3: Choose the best feature to split

- **Experience** has highest IG = 0.971 → split on **Experience**

Step 4: Build the tree

```
Experience
├─ High : Yes
├─ Medium : Yes
└─ Low : No
```

- All leaves are **pure**, so tree building stops.

Summary

1. Compute **total entropy** of dataset.
2. Compute **entropy for each feature**.
3. Compute **information gain**.
4. Split on the **feature with highest gain**.
5. Repeat recursively until leaves are pure.