Lecture: Naive Bayes Classification on Structured Data

1. Objective

Apply Naive Bayes to a **small structured dataset** and predict the label of a test observation.

2. Dataset (Tabular / Structured)

We consider a simple dataset for predicting whether a student will **Pass** or **Fail** based on three categorical features:

1. StudyTime = {Low, Medium, High}
2. Attendance = {Poor, Good}
3. Difficulty = {Easy, Medium, Hard}

Training Dataset

ID	StudyTime	Attendance	Difficulty	Label
S1	Low	Poor	Hard	Fail
S2	Medium	Poor	Hard	Fail
S3	High	Good	Medium	Pass
S4	Medium	Good	Easy	Pass
S5	High	Good	Hard	Pass
S6	Low	Good	Medium	Fail

Total: 6 records
Classes: Pass (3), Fail (3)

3. Step 1: Compute Priors

$$P(Pass) = \frac{3}{6} = 0.5$$
$$P(Fail) = \frac{3}{6} = 0.5$$

4. Step 2: Compute Likelihoods

Because Naive Bayes assumes **conditional independence of features**, we compute:

$$P(\text{StudyTime} = x \mid C)$$
$$P(\text{Attendance} = y \mid C)$$
$$P(\text{Difficulty} = z \mid C)$$

We use **Laplace smoothing (α = 1)**.

4.1 Likelihood Tables (per Feature, per Class)

StudyTime

StudyTime	Count in Pass	Count in Fail
Low	0	2
Medium	1	1
High	2	0

Total possible values = 3
Total Pass = 3
Total Fail = 3

Using Laplace smoothing:

For class Pass:

$$P(StudyTime = x \mid Pass) = \frac{\text{count} + 1}{3 + 3} = \frac{\text{count} + 1}{6}$$

For Fail:

$$P(StudyTime = x \mid Fail) = \frac{\text{count} + 1}{6}$$

Attendance

Attendance	Pass	Fail
Good	3	1
Poor	0	2

Total = 2 values (Good, Poor)

With smoothing:

$$P(Attendance = x \mid C) = \frac{\text{count} + 1}{3 + 2} = \frac{\text{count} + 1}{5}$$

Difficulty

Difficulty	Pass	Fail
Easy	1	0
Medium	1	1
Hard	1	2

Total = 3 values
Use smoothing:

$$P(Difficulty = x \mid C) = \frac{\text{count} + 1}{3 + 3} = \frac{\text{count} + 1}{6}$$

5. Step 3: Define the Test Instance

We want to predict the label for:

Feature	Value
StudyTime	High
Attendance	Good
Difficulty	Hard

6. Step 4: Compute Posterior for Each Class

We compute:

$$P(C \mid X) \propto P(C) \prod_i P(x_i \mid C)$$

6.1 Compute for Pass

StudyTime = High

Pass count = 2

$$P(High \mid Pass) = \frac{2+1}{6} = \frac{3}{6} = 0.5$$

Attendance = Good

Pass count = 3

$$P(Good \mid Pass) = \frac{3+1}{5} = \frac{4}{5} = 0.8$$

Difficulty = Hard

Pass count = 1

$$P(Hard \mid Pass) = \frac{1+1}{6} = \frac{2}{6} = 0.333$$

Now compute:

$$P(Pass \mid X) \propto 0.5 \times 0.5 \times 0.8 \times 0.333$$
$$= 0.5 \times 0.1332 = 0.0666$$

6.2 Compute for Fail

StudyTime = High

Fail count = 0

$$P(High \mid Fail) = \frac{0+1}{6} = 0.1667$$

Attendance = Good

Fail count = 1

$$P(Good \mid Fail) = \frac{1+1}{5} = 0.4$$

Difficulty = Hard

Fail count = 2

$$P(Hard \mid Fail) = \frac{2+1}{6} = 0.5$$

Now compute:

$$P(Fail \mid X) \propto 0.5 \times 0.1667 \times 0.4 \times 0.5$$
$$= 0.5 \times 0.03334 = 0.01667$$

7. Step 5: Final Comparison

$$P(Pass \mid X) \propto 0.0666$$
$$P(Fail \mid X) \propto 0.01667$$

Since:

$$0.0666 > 0.01667$$

Prediction: The student will PASS.

8. Slide-ready Summary for Final Frame

Decision Outcome

$$\hat{C} = Pass$$

Why?

The combination of:

- High Study Time
- Good Attendance
- Hard Difficulty

produces a higher posterior probability for **Pass** after combining priors and likelihoods.

1. Learning goals (Bloom-aligned)

By the end of this mini-lecture you should be able to:

- **Remember / Understand**
  - State Bayes' theorem and the Naive Bayes assumption for text.
- **Apply**
  - Compute priors and likelihoods from a labeled text corpus.
  - Use Naive Bayes to classify a new sentence as positive/negative.
- **Analyze / Evaluate**
  - Critically inspect the independence assumption in the context of sentiment words.
  - Identify limitations for higher-education NLP (e.g., student feedback).
- **Create**
  - Extend this toy example to a small sentiment classifier for course reviews or reflective writing in a personalized learning system.

2. Bayes' theorem and Naive Bayes for text

For a class $C \in \{\text{pos,neg}\}$and document d:

$$P(C \mid d) = \frac{P(d \mid C)\, P(C)}{P(d)}$$

Naive Bayes assumes:

- Represent text as a **bag of words**: $d = (w_1, w_2, \ldots, w_n)$.
- Words are **conditionally independent** given the class:

$$P(d \mid C) = \prod_{i=1}^{n} P(w_i \mid C)$$

So classification reduces to:

$$\hat{C} = \arg\max_{C}\; P(C) \prod_{i=1}^{n} P(w_i \mid C)$$

In NLP sentiment analysis, C is “positive” vs “negative” sentiment.

3. Tiny sentiment corpus (our toy dataset)

We build a very small labeled corpus.

ID	Sentence	Label
D1	I love this movie	pos
D2	This film is excellent and inspiring	pos
D3	What a terrible and boring movie	neg
D4	I hate this film	neg
D5	An excellent and enjoyable movie	pos
D6	This movie is boring and slow	neg

Total documents: 6
Classes: pos and neg.

This is analogous to a small set of labeled student comments about a course.

4. Preprocessing and vocabulary

We choose a very simple preprocessing pipeline (for teaching, not for production):

1. Lowercase text.
2. Tokenize into words.
3. Remove a small set of stopwords:
   {"i", "this", "is", "a", "and", "what"}.

Process each sentence:

- D1 (pos): "I love this movie"
  → tokens: ["love", "movie"]
- D2 (pos): "This film is excellent and inspiring"
  → tokens: ["film", "excellent", "inspiring"]
- D3 (neg): "What a terrible and boring movie"
  → tokens: ["terrible", "boring", "movie"]

- D4 (neg): "I hate this film"
  → tokens: ["hate", "film"]
- D5 (pos): "An excellent and enjoyable movie"
  → tokens: ["an", "excellent", "enjoyable", "movie"]
- D6 (neg): "This movie is boring and slow"
  → tokens: ["movie", "boring", "slow"]

Now construct the vocabulary V(unique tokens):

$$V = \{\text{an, boring, enjoyable, excellent, film, hate, inspiring, love, movie, slow, terrible}\}$$

Vocabulary size: $|V| = 11$.

5. Compute class priors

Number of documents per class:

- Positive: D1, D2, D5 → $N_{\text{pos}} = 3$
- Negative: D3, D4, D6 → $N_{\text{neg}} = 3$
- Total documents: $N = 6$

Class priors:

$$P(\text{pos}) = \frac{3}{6} = 0.5, P(\text{neg}) = \frac{3}{6} = 0.5$$

6. Word counts per class

6.1 Positive class

Positive documents (tokens):

- D1: ["love", "movie"]
- D2: ["film", "excellent", "inspiring"]
- D5: ["an", "excellent", "enjoyable", "movie"]

Counts in positive class:

- love: 1
- movie: 2

- film: 1
- excellent: 2
- inspiring: 1
- an: 1
- enjoyable: 1

Total token count in positive class:

$$N_{\text{words,pos}} = 1 + 2 + 1 + 2 + 1 + 1 + 1 = 9$$

6.2 Negative class

Negative documents (tokens):

- D3: ["terrible", "boring", "movie"]
- D4: ["hate", "film"]
- D6: ["movie", "boring", "slow"]

Counts in negative class:

- terrible: 1
- boring: 2
- movie: 2
- hate: 1
- film: 1
- slow: 1

Total token count in negative class:

$$N_{\text{words,neg}} = 1 + 2 + 2 + 1 + 1 + 1 = 8$$

Words that do not appear in a class have count 0 in that class.

7. Likelihoods with Laplace smoothing

We use multinomial Naive Bayes with **Laplace smoothing**:

$$P(w \mid C) = \frac{\text{count}(w, C) + \alpha}{N_{\text{words},C} + \alpha \mid V \mid}$$

Use $\alpha = 1$(standard choice).

So:

- For the positive class:

$$P(w \mid \text{pos}) = \frac{\text{count}(w,\text{pos}) + 1}{9 + 1 \cdot 11} = \frac{\text{count}(w,\text{pos}) + 1}{20}$$

- For the negative class:

$$P(w \mid \text{neg}) = \frac{\text{count}(w,\text{neg}) + 1}{8 + 1 \cdot 11} = \frac{\text{count}(w,\text{neg}) + 1}{19}$$

We only need words that will appear in our test sentence, but I will compute those explicitly.

Consider three words: "movie", "excellent", "enjoyable".

7.1 Likelihoods for the positive class

- movie in pos: count = 2

$$P(\text{movie} \mid \text{pos}) = \frac{2+1}{20} = \frac{3}{20} = 0.15$$

- excellent in pos: count = 2

$$P(\text{excellent} \mid \text{pos}) = \frac{2+1}{20} = 0.15$$

- enjoyable in pos: count = 1

$$P(\text{enjoyable} \mid \text{pos}) = \frac{1+1}{20} = \frac{2}{20} = 0.10$$

7.2 Likelihoods for the negative class

- movie in neg: count = 2

$$P(\text{movie} \mid \text{neg}) = \frac{2+1}{19} = \frac{3}{19} \approx 0.1579$$

- excellent in neg: count = 0

$$P(\text{excellent} \mid \text{neg}) = \frac{0+1}{19} = \frac{1}{19} \approx 0.0526$$

- enjoyable in neg: count = 0

$$P(\text{enjoyable} \mid \text{neg}) = \frac{0+1}{19} = \frac{1}{19} \approx 0.0526$$

8. Classifying a test sentence

Let the test sentence be:

"This movie is excellent and enjoyable"

Preprocess with the same pipeline:

- Lowercase and tokenize: ["this", "movie", "is", "excellent", "and", "enjoyable"]
- Remove stopwords {this, is, and, i, a, what}:
  → ["movie", "excellent", "enjoyable"]

So our document $d$is the bag of words:

$$d = (\text{movie,excellent,enjoyable})$$

8.1 Compute posterior for pos

$$\begin{aligned} P(\text{pos} \mid d) \quad &\propto P(\text{pos}) \cdot P(\text{movie} \mid \text{pos}) \cdot P(\text{excellent} \mid \text{pos}) \cdot P(\text{enjoyable} \mid \text{pos}) \\ &= 0.5 \times 0.15 \times 0.15 \times 0.10 \\ &= 0.5 \times 0.00225 \\ &= 0.001125 \end{aligned}$$

8.2 Compute posterior for neg

$$\begin{aligned} P(\text{neg} \mid d) \quad &\propto P(\text{neg}) \cdot P(\text{movie} \mid \text{neg}) \cdot P(\text{excellent} \mid \text{neg}) \cdot P(\text{enjoyable} \mid \text{neg}) \\ &= 0.5 \times 0.1579 \times 0.0526 \times 0.0526 \\ &\approx 0.5 \times 0.00043738 \\ &\approx 0.0002187 \end{aligned}$$

We only need relative comparison:

- $P(\text{pos} \mid d) \propto 0.001125$
- $P(\text{neg} \mid d) \propto 0.0002187$

Since $0.001125 > 0.0002187$:

$$\hat{C} = \text{pos}$$

Prediction: The test sentence is classified as **positive** sentiment.

9. Pipeline summary (text diagram)

Conceptually, the sentiment classifier behaves like:

Raw text

↓

Preprocessing (lowercase, tokenization, stopword removal)

↓

Bag-of-words representation

↓

Training:

- Estimate class priors P(C)

- Estimate word likelihoods P(w | C) with Laplace smoothing

↓

Naive Bayes classifier:

For new text d:

compute P(C | d) ∝ P(C) ∏ P(w_i | C)

↓

Predicted sentiment (pos / neg)

In your research context, **C** could be:

- “High satisfaction”, “Neutral”, “Low satisfaction” for course feedback.
- “Needs help”, “On track”, “Advanced” for reflective learning logs.

Agentic AI tutors can use such classifiers as sub-agents to monitor learner affect or engagement and adapt activities.