Naive Bayes

Outline

Introduction to Naive Bayes

Math Example

Introduction to Naive Bayes

What is Naive Bayes?

- A probabilistic classification algorithm based on Bayes' Theorem.
- Called "naive" due to the strong (naive) assumption of feature independence.
- Performs well in many practical situations, especially text classification.

Bayes' Theorem

$$P(C|X) = \frac{P(X|C) \cdot P(C)}{P(X)}$$

- $P(C|X)$: Posterior probability of class C given data X
- $P(X|C)$: Likelihood of data X given class C
- $P(C)$: Prior probability of class C
- $P(X)$: Evidence (normalizing constant)

In classification:

$$P(C|X) \propto P(X|C) \cdot P(C)$$

Naive Independence Assumption

$$P(X|C) = \prod_{i=1}^{n} P(x_i|C)$$

- Assumes all features x_i are conditionally independent given class C.
- Simplifies computation dramatically.

Types of Naive Bayes

- **Gaussian NB:** Continuous data, assumes normal distribution.
- **Multinomial NB:** Discrete counts, great for text data.
- **Bernoulli NB:** Binary features (0/1), presence/absence.

Gaussian Naive Bayes

For continuous features:

$$P(x_i|C) = \frac{1}{\sqrt{2\pi\sigma_C^2}} \exp\left(-\frac{(x_i - \mu_C)^2}{2\sigma_C^2}\right)$$

- μ_C: Mean of feature for class C
- σ_C^2: Variance of feature for class C

Example: Spam Detection

- Features: Presence of words in email
- Classes: Spam vs Not Spam
- Learn:
  - $P(\text{spam})$, $P(\text{not spam})$
  - $P(\text{word}|\text{spam})$, etc.
- Predict using Bayes Rule

Pros and Cons

Pros:

- Simple, fast, efficient with high-dimensional data
- Requires small training data

Cons:

- Assumes feature independence (often unrealistic)
- Sensitive to correlated features

Python Example

```python
from sklearn.naive_bayes import GaussianNB
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split

X, y = load_iris(return_X_y=True)
X_train, X_test, y_train, y_test = train_test_split(X, y)

model = GaussianNB()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
```

Summary

- Naive Bayes is powerful despite its simplicity.
- Useful in classification tasks, especially with text data.
- Works best with large, clean, independent features.

Math Example

Dataset

Day	Outlook	Windy	Rain?
1	Sunny	False	No
2	Sunny	True	No
3	Overcast	False	Yes
4	Rainy	False	Yes
5	Rainy	True	No
6	Overcast	True	Yes
7	Sunny	False	No

Prediction Task

Given a new day with:

- Outlook = Rainy
- Windy = False

Predict whether it will **Rain** using Naive Bayes with Laplace Smoothing.

Step 1: Compute Class Priors

Total samples: 7

Yes $=3$, No $=4$, Number of classes $=2$

$$P(\text{Yes}) = \frac{3+1}{7+2} = \frac{4}{9} \quad P(\text{No}) = \frac{4+1}{7+2} = \frac{5}{9}$$

Step 2: Compute Likelihoods with Laplace Smoothing

For Class = Yes

$$P(\text{Outlook=Rainy — Yes}) = \frac{1+1}{3+3} = \frac{2}{6}$$

$$P(\text{Windy=False — Yes}) = \frac{2+1}{3+2} = \frac{3}{5}$$

For Class = No

$$P(\text{Outlook=Rainy — No}) = \frac{1+1}{4+3} = \frac{2}{7}$$

$$P(\text{Windy=False — No}) = \frac{2+1}{4+2} = \frac{3}{6}$$

Laplace Smoothing Formula

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

Where:

- x_i = feature value (e.g., Rainy, False)
- C = class label (Yes or No)
- $\text{count}(x_i, C)$ = number of times x_i appears in class C
- $\text{count}(C)$ = total samples in class C
- V = number of possible values the feature can take

Step 3: Compute Log Probabilities

Log Posterior (Yes):

$$\log\left(\frac{4}{9}\right) + \log\left(\frac{2}{6}\right) + \log\left(\frac{3}{5}\right) \approx -2.420$$

Log Posterior (No):

$$\log\left(\frac{5}{9}\right) + \log\left(\frac{2}{7}\right) + \log\left(\frac{3}{6}\right) \approx -2.534$$

Final Prediction

$$\log P(\text{Yes}|X) = -2.42 > \log P(\text{No}|X) = -2.53$$

Conclusion
Prediction: It Will Rain (Yes)

Thank you!