Gaussian Naive Bayes for Multiclass Classification

Edouard Clocheret, Eric Fan,
Shivam Hingorani, Junhui Huang

Overview of Gaussian Naive Bayes

Input:

1. Features X:
   - **Dimension**: $X \in \mathbb{R}^{n \times d}$, where:
     - n : Number of samples.
     - d : Number of features.
2. Labels y:
   - **Dimension**: $y \in \mathbb{R}^{n}$, where:
     - n : Number of samples.

Output:

Final prediction Class y

Strengths:

1. Simple and Fast
2. Works Well with Small and Large Datasets
3. Works Well with High-Dimensional Data

Mathematical Background

Bayes Theorem

Bayes' theorem provides a way to compute the probability of a class C given some features X:

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

- P(C | X) : **Posterior probability** — the probability of class C given the features X .
- P(X | C) : **Likelihood** — the probability of observing X given that the class is C .
- P(C) : **Prior probability** — the probability of class C before observing the data.
- P(X) : **Evidence** — the total probability of X across all classes (this is constant for all classes and is often ignored during prediction).

Mathematical background of the algorithm

The goal: Predict the class label based on the features among multiple classes.

$$C = \arg\max_{C_k} \boxed{P(C_k \mid X)} = \arg\max_{C_k} P(C_k) \cdot \prod_{i=1}^{n} P(x_i | C_k)$$

First, calculate the prior distribution.

which is defined as the probability of each class based on the training data.

$$P(C_k) = \frac{\text{Number of samples in class } C_k}{\text{Total number of samples}} = \frac{N_k}{N}$$

Second, calculate the likelihood.

which is the probability of observing **feature** X given that the data point belongs to a specific **Class** C_k.

Naive Assumption (features are independent)

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

Detailed look on likelihood (Gaussian distribution)

$$P(x_i|C_k) = \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp\left(-\frac{(x_i - \mu_k)^2}{2\sigma_k^2}\right)$$

Third, calculate the posterior likelihood.

Since $P(X)$ is the same for all classes, we can ignore it when comparing classes.

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

Finally, pick the class with the highest posterior probability

$$C = \arg\max_{C_k} P(C_k \mid X) = \arg\max_{C_k} P(C_k) \cdot \prod_{i=1}^{n} P(x_i | C_k)$$

Loss (1/2)

$$\text{Likelihood} = \prod_{i=1}^{N} P(y_i|X_i) \Rightarrow \text{NLL} = -\sum_{i=1}^{N} \log P(y_i|X_i)$$

Likelihood across all N samples, that has to be **maximized.**

It represents the **probability** that a label is **correctly assigned**.

Loss = Negative Log Likelihood, that has to be **minimized.**

We use the **log** loss because it is easier to calculate derivatives, and more efficient to **sum** log **probabilities**.

Loss (2/2)

Assumption: the features follow a Gaussian distribution.

$$P(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

So the log-Loss becomes :

$$\log L(\mu, \sigma^2) = -\frac{n}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i - \mu)^2$$

Maximum Likelihood Estimators (1/2)

$$P(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

How to compute **estimators** of the parameters **μ** and **σ^2** that **minimizes the Loss?**

Maximum Likelihood Estimators (2/2)

$$\log L(\mu, \sigma^2) = -\frac{n}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i - \mu)^2$$

Taking the derivative with respect to μ and σ^2 and setting them to zero, we obtain:

- The MLE for the mean:

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

- The MLE for the variance:

$$\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{\mu})^2$$

The result of MLE is the empirical mean and variance.

Model

Model Implementation (1/2)

```
Input:
  - Training data X_train (N samples, n features)
  - Labels y_train (corresponding to X_train)
  - Test data X_test

Output:
  - Predicted labels y_pred for X_test

Step 1: Compute Class Statistics
  - Separate X_train by class into subsets {X_C1, X_C2, ..., X_Ck}, one for each class.
  - For each class C_k:
      - Compute prior probability P(C_k):
          P(C_k) = N_Ck / N_total
      - For each feature i:
          - Compute mean μ_Ck,i:
              μ_Ck,i = mean(X_Ck[:, i])
          - Compute variance σ_Ck,i^2:
              σ_Ck,i^2 = variance(X_Ck[:, i])
```

Model Implementation (2/2)

```
Step 2: Define Gaussian Probability Density Function
  Function Gaussian(x, μ, σ^2):
      return (1 / sqrt(2 * π * σ^2)) * exp(-((x - μ)^2) / (2 * σ^2))

Step 3: Predict Class for Test Data
  For each sample x in X_test:
      For each class C_k:
          - Compute log of posterior probability:
              log P(C_k | x) = log P(C_k)
                              + Σ (log Gaussian(x_i, μ_Ck,i, σ_Ck,i^2) for i in features)
      - Assign x to the class C_k with the highest log P(C_k | x)

Step 4: Return Predictions
  Return y_pred (predicted labels for all samples in X_test)
```

Reproduction

Original Study

- Multiclass Classification with Iris Dataset using Gaussian Naive Bayes, *International Journal of Computer Science and Mobile Computing*
- Uses “the Iris dataset” from UCI Machine Learning repository, included in scikit-learn
- Train a Gaussian Naive Bayes classifier using 80% of data, then test using the remaining 20%

Data

- 150 examples, evenly distributed
- 3 labels / classes: Iris setosa, Iris versicolor, Iris virginica
- 4 features: sepal length, sepal width, petal length, petal width

Iris setosa

Iris versicolor

Iris virginica

Performance

1. Accuracy (A) = (TP + TN) / (TP + TN + FP + FN)
2. Precision (P) = (TP) / (TP + FP)
3. Recall (R) = (TP) / (TP + FN)
4. F1-score = 2 • (P • R) / (P + R)

Results

Accuracy	Precision	Recall	F1-score
0.95	0.95	0.95	0.95

- Our results closely mirror those produced by previous studies using this dataset and align with the scikit-learn implementation of GNB
- **90% accuracy is often considered ‘good’, and we are comfortably above that**
- 77% test accuracy on “Student Performance” dataset

Limitations

- Naive assumption (feature independence)
    - Poor Performance with Correlated Features
- Gaussian Distribution Assumption
- Sensitivity to Data Scaling
    - Features with larger variances can dominate the calculation of probabilities

Thank you for your attention!

Edouard Clocheret, Eric Fan, Shivam Hingorani, Junhui Huang

References

1. Wikipedia, 2023. Naive Bayes Classifier. [online] Available at: https://en.wikipedia.org/wiki/Naive_Bayes_classifier [Accessed 8 Dec. 2024].
2. GeeksforGeeks, 2023. Gaussian Naive Bayes. [online] Available at: https://www.geeksforgeeks.org/gaussian-naive-bayes/ [Accessed 8 Dec. 2024].
3. Brown University, n.d. DATA2060, Assignment 6. [online] Brown University. Available at: [Accessed 8 Dec. 2024].
4. Scikit-learn, 2024. Naive Bayes. [online] Available at: https://scikit-learn.org/1.5/modules/naive_bayes.html [Accessed 8 Dec. 2024].
5. Iqbal, Z. and Yadav, M., 2020. Multiclass Classification with Iris Dataset Using Gaussian Naive Bayes. International Journal of Computer Science and Mobile Computing, 9(4), pp.27–35. Available at: Link [Accessed 8 Dec. 2024].
6. UCI Machine Learning Repository, n.d. Iris Data Set. [online] Available at: https://archive.ics.uci.edu/ml/datasets/Iris [Accessed 8 Dec. 2024].