Naive Bayes

CS21206: Foundations of AI and ML

Abir Das & Ayan Chaudhury

IIT Kharagpur

Feb 06, 2026

Agenda

§ Understand the philosophy of generative and discriminative schools of Machine Learning.

§ Learn about the "Naive Bayes" assumption

§ Employ the naivity assumption for discrete and continuous features

Resources

- Machine Learning: A Probabilistic Perspective by Kevin P. Murphy.
- Andrew Ng's CS229 Lecture Notes

Two Ways to Classify

We want to learn a mapping from features $\mathbf{x}$ to class labels $y \in \{0, 1\}$.

Discriminative (e.g., Logistic Reg)

- Learn $P(y|\mathbf{x})$ directly.
- Tries to find the **decision boundary** separating classes.
- *"Given features $\mathbf{x}$, which class is it?"*

Generative (e.g., Naive Bayes)

- Learn $P(\mathbf{x}|y)$ and $P(y)$.
- Models the **distribution** of data for each class separately.
- *"What does $\mathbf{X}$ look like if it is a 'cat'?"*

Prediction using Bayes Rule:

$$\hat{y} = \arg\max_y P(y|\mathbf{x}) = \arg\max_y \frac{P(\mathbf{x}|y)P(y)}{P(\mathbf{x})} = \arg\max_y \underbrace{P(\mathbf{x}|y)}_{\text{Likelihood}} \underbrace{P(y)}_{\text{Prior}}$$

Motivation: The “Ideal” Generative Model

Ideally, we would just learn the full **Joint Distribution** $P(x_1, \ldots, x_D, y)$. If we had this, we could answer ANY query (classification, missing data, *etc.*).

Motivation: The “Ideal” Generative Model

Ideally, we would just learn the full **Joint Distribution** $P(x_1, \ldots, x_D, y)$. If we had this, we could answer ANY query (classification, missing data, *etc.*).
Example 1: Imagine a simple world with 3 binary variables:

- x_1: Fever (0/1)
- x_2: Cough (0/1)
- y: Flu (0/1)

Motivation: The "Ideal" Generative Model

Ideally, we would just learn the full **Joint Distribution** $P(x_1, \ldots, x_D, y)$. If we had this, we could answer ANY query (classification, missing data, *etc.*).

Example 1: Imagine a simple world with 3 binary variables:

- x_1: Fever (0/1)
- x_2: Cough (0/1)
- y: Flu (0/1)

The Full Joint Distribution $P(x_1, x_2, y)$ is a table of $2^3 = 8$ entries:

x_1 (Fever)	x_2 (Cough)	y (Flu)	Prob
0	0	0	0.40
0	0	1	0.01
0	1	0	0.05
0	1	1	0.05
1	0	0	0.05
1	0	1	0.10
1	1	0	0.04
1	1	1	0.30

Motivation: The "Ideal" Generative Model

Ideally, we would just learn the full **Joint Distribution** $P(x_1, \ldots, x_D, y)$. If we had this, we could answer ANY query (classification, missing data, *etc.*).

Example 1: Imagine a simple world with 3 binary variables:

- x_1: Fever (0/1)
- x_2: Cough (0/1)
- y: Flu (0/1)

The Full Joint Distribution $P(x_1, x_2, y)$ is a table of $2^3 = 8$ entries:

x_1 (Fever)	x_2 (Cough)	y (Flu)	Prob
0	0	0	0.40
0	0	1	0.01
0	1	0	0.05
0	1	1	0.05
1	0	0	0.05
1	0	1	0.10
1	1	0	0.04
1	1	1	0.30

We can answer ANY query!

- **Prediction:** $P(\text{Flu}|\text{Fever}, \text{Cough})$?
  $\frac{P(1,1,1)}{P(1,1,0)+P(1,1,1)} = \frac{0.30}{0.04+0.30} \approx 0.88$
- **Marginal:** Probability of Fever?
  Sum rows where $x_1 = 1$:
  $0.05 + 0.10 + 0.04 + 0.30 = 0.49$
- **Missing Data:** $P(\text{Flu}|\text{Cough})$?
  Sum over Fever cases (0/1) and normalize.

Missing Data

- x_1: Fever (0/1)
- x_2: Cough (0/1)
- y: Flu (0/1)

$$P(\text{Flu}|\text{Cough}) = P(y=1|x_2=1) = \frac{P(y=1, x_2=1)}{P(x_2=1)}$$

$$= \frac{\sum_{x_1} P(y=1, x_1, x_2=1)}{\sum_{y}\sum_{x_1} P(y, x_1, x_2=1)}$$

$$= \frac{0.05+0.30}{0.05+0.05+0.04+0.30} = \frac{0.35}{0.44}$$

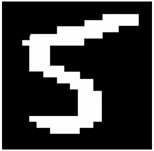
$$= 0.795$$

Takeaway

Even though we didn't measure Fever (x_1), we could still use the Joint Distribution to provide an exact probability. This is the power of Generative Models!

Example 2: Image Classification

Now imagine a tiny 30×30 binary image ($D = 900$ pixels).



30×30 image

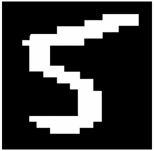
The Joint Distribution Table:

- Number of rows $= 2^{900} \approx 10^{270}$
- Assuming 1 row takes 1 nanosecond to read...
- It would take billions of years just to scan the table!

Prediction: $P(Y = 5|X_1, X_2, \cdots X_{900}) = \frac{P(X_1, X_2, \cdots X_{900}|Y=5)P(Y=5)}{P(X_1, X_2, \cdots X_{900})}$

Example 2: Image Classification

Now imagine a tiny 30×30 binary image ($D = 900$ pixels).



30×30 image

The Joint Distribution Table:

- Number of rows $= 2^{900} \approx 10^{270}$
- Assuming 1 row takes 1 nanosecond to read...
- It would take billions of years just to scan the table!

Prediction: $P(Y = 5|X_1, X_2, \cdots X_{900}) = \frac{P(X_1, X_2, \cdots X_{900}|Y=5)P(Y=5)}{P(X_1, X_2, \cdots X_{900})}$

Conclusion

For real-world problems (Text, Images, Genetics), we **cannot** estimate the full joint distribution table. We neither have enough data nor enough time to fill and read 10^{270} rows.

We need a simplifying assumption! $\rightarrow$ **Naive Bayes**.

The "Naive" Solution

To make learning feasible, we make a strong assumption to reduce parameters.

Naive Bayes Assumption: The features $x_1, \ldots, x_D$ are **conditionally independent** given the class y.

$$P(\mathbf{x}|y) = P(x_1, x_2, \ldots, x_D|y) = \prod_{j=1}^{D} P(x_j|y)$$

The “Naive” Solution

To make learning feasible, we make a strong assumption to reduce parameters.
Naive Bayes Assumption: The features $x_1, \ldots, x_D$ are **conditionally independent** given the class y.

$$P(\mathbf{x}|y) = P(x_1, x_2, \ldots, x_D|y) = \prod_{j=1}^{D} P(x_j|y)$$

The Parameter Savings:

- Instead of one giant table of size 2^D, we learn D small tables.
- For binary features: we need just 1 parameter per feature per class.
- Total parameters: $2D + 1$ (Linear $O(D)$ complexity instead of Exponential $O(2^D)$!).

The Naive Bayes Classifier

§ Given:

- Prior $P(y)$
- D conditionally independent features $x_1, \ldots, x_D$, given the class y
- For each feature, we specify $P(x_j|y)$

§ Classification decision rule:

- Prior $y^* = \arg\max_y P(y)P(x_1, \ldots, x_D|y) = \arg\max_y P(y)\prod_j P(x_j|y)$

The Naive Bayes Classifier

- Given:
  - Prior $P(y)$
  - D conditionally independent features $x_1, \ldots, x_D$, given the class y
  - For each feature, we specify $P(x_j|y)$
- Classification decision rule:
  - Prior $y^* = \arg\max_y P(y)P(x_1, \ldots, x_D|y) = \arg\max_y P(y)\prod_j P(x_j|y)$
- Let us take an example involving two classes. In case of two classes, $y \in \{0, 1\}$, we predict that $y = 1$ if:

$$\frac{P(y=1)\prod_j P(x_j|y=1)}{P(y=0)\prod_j P(x_j|y=0)} > 1 \tag{1}$$

Naive Bayes: Two Classes

- Assuming Boolean features $x_j \in \{0, 1\}$, let

$$p_j = P(x_j = 1|y = 1), \text{ then } 1 - p_j = P(x_j = 0|y = 1)$$

- Hence: $P(x_j|y = 1) = p_j^{x_j}(1 - p_j)^{(1-x_j)}$

Naive Bayes: Two Classes

- Assuming Boolean features $x_j \in \{0, 1\}$, let

$$p_j = P(x_j = 1|y = 1), \text{ then } 1 - p_j = P(x_j = 0|y = 1)$$

- Hence: $P(x_j|y = 1) = p_j^{x_j}(1 - p_j)^{(1-x_j)}$
- Similarly,

$$q_j = P(x_j = 1|y = 0), \text{ then } 1 - q_j = P(x_j = 0|y = 0)$$

- Hence: $P(x_j|y = 0) = q_j^{x_j}(1 - q_j)^{(1-x_j)}$

Naive Bayes: Two Classes

- Assuming Boolean features $x_j \in \{0,1\}$, let

$$p_j = P(x_j = 1|y = 1), \text{ then } 1 - p_j = P(x_j = 0|y = 1)$$

- Hence: $P(x_j|y = 1) = p_j^{x_j}(1 - p_j)^{(1-x_j)}$
- Similarly,

$$q_j = P(x_j = 1|y = 0), \text{ then } 1 - q_j = P(x_j = 0|y = 0)$$

- Hence: $P(x_j|y = 0) = q_j^{x_j}(1 - q_j)^{(1-x_j)}$
- Then eqn. (1) implies,

$$\frac{P(y=1)\prod_j p_j^{x_j}(1-p_j)^{(1-x_j)}}{P(y=0)\prod_j q_j^{x_j}(1-q_j)^{(1-x_j)}} > 1$$

$$\frac{P(y=1)\prod_j \left(\frac{p_j}{1-p_j}\right)^{x_j}(1-p_j)}{P(y=0)\prod_j \left(\frac{q_j}{1-q_j}\right)^{x_j}(1-q_j)} > 1 \tag{2}$$

Naive Bayes: Two Classes

§ Take logarithm; we predict $y = 1$, if:

$$\log \frac{P(y=1)}{P(y=0)} + \underbrace{\sum_j \log \frac{1-p_j}{1-q_j}}_{b_j} + \sum_j \underbrace{\left(\log \frac{p_j}{1-p_j} - \log \frac{q_j}{1-q_j}\right)}_{\theta_j} x_j > 0 \quad (3)$$

Naive Bayes: Learning Parameters

- How do we estimate the probabilities from training data?
- We use **Maximum Likelihood Estimation (MLE)**, which boils down to simple counting.
  - **Prior Probabilities** $P(Y)$:

$$\hat{P}(Y=1) = \frac{\#\text{ examples with } Y=1}{\text{Total } \#\text{ examples}}$$

  - **Conditional Probabilities** $P(X_i = 1|Y = y)$:

$$\hat{P}(X_i=1|Y=1) = \frac{\#\text{ examples with } Y=1 \text{ AND } X_i=1}{\#\text{ examples with } Y=1}$$

Naive Bayes: Learning Parameters

- How do we estimate the probabilities from training data?
- We use **Maximum Likelihood Estimation (MLE)**, which boils down to simple counting.
  - **Prior Probabilities** $P(Y)$:

$$\hat{P}(Y=1) = \frac{\#\text{ examples with } Y=1}{\text{Total } \#\text{ examples}}$$

  - **Conditional Probabilities** $P(X_i = 1|Y = y)$:

$$\hat{P}(X_i = 1|Y = 1) = \frac{\#\text{ examples with } Y=1 \text{ AND } X_i = 1}{\#\text{ examples with } Y=1}$$

- Similarly for $P(X_i = 1|Y = 0)$.
- Since features are binary, $P(X_i = 0|Y) = 1 - P(X_i = 1|Y)$.

The Dataset: Will I Play Tennis Today?

Day	Sky	Temp	Humid	Play?
D1	Sunny	Warm	Normal	Yes
D2	Rainy	Warm	High	Yes
D3	Sunny	Cold	Normal	Yes
D4	Rainy	Cold	High	No
D5	Sunny	Warm	High	No

- **Training Data ($N = 5$):**
- **Test Instance:** $\mathbf{x} =$ (Sunny, Warm, High)
- **Goal:** Predict Play? (Yes/No)

The Dataset: Will I Play Tennis Today?

Day	Sky	Temp	Humid	Play?
D1	Sunny	Warm	Normal	Yes
D2	Rainy	Warm	High	Yes
D3	Sunny	Cold	Normal	Yes
D4	Rainy	Cold	High	No
D5	Sunny	Warm	High	No

- **Training Data ($N = 5$):**
- **Test Instance:**
  $\mathbf{x} = (\text{Sunny, Warm, High})$
- **Goal:** Predict Play? (Yes/No)

Class: Yes ($N_{yes} = 3$)

- $P(\text{Yes}) = 3/5$
- $P(\text{Sunny}|\text{Yes}) = 2/3$
- $P(\text{Warm}|\text{Yes}) = 2/3$
- $P(\text{High}|\text{Yes}) = 1/3$

$$\text{Score}_{Yes} = \frac{3}{5} \times \frac{2}{3} \times \frac{2}{3} \times \frac{1}{3} = \mathbf{0.089}$$

The Dataset: Will I Play Tennis Today?

Day	Sky	Temp	Humid	Play?
D1	Sunny	Warm	Normal	Yes
D2	Rainy	Warm	High	Yes
D3	Sunny	Cold	Normal	Yes
D4	Rainy	Cold	High	No
D5	Sunny	Warm	High	No

- **Training Data ($N = 5$):**
- **Test Instance:** $\mathbf{x} = (\text{Sunny, Warm, High})$
- **Goal:** Predict Play? (Yes/No)

Class: Yes ($N_{yes} = 3$)

- $P(\text{Yes}) = 3/5$
- $P(\text{Sunny}|\text{Yes}) = 2/3$
- $P(\text{Warm}|\text{Yes}) = 2/3$
- $P(\text{High}|\text{Yes}) = 1/3$

$$\text{Score}_{Yes} = \frac{3}{5} \times \frac{2}{3} \times \frac{2}{3} \times \frac{1}{3} = \mathbf{0.089}$$

Class: No ($N_{no} = 2$)

- $P(\text{No}) = 2/5$
- $P(\text{Sunny}|\text{No}) = 1/2$
- $P(\text{Warm}|\text{No}) = 1/2$
- $P(\text{High}|\text{No}) = 2/2 = 1.0$

$$\text{Score}_{No} = \frac{2}{5} \times \frac{1}{2} \times \frac{1}{2} \times 1 = \mathbf{0.1}$$

The Dataset: Will I Play Tennis Today?

Day	Sky	Temp	Humid	Play?
D1	Sunny	Warm	Normal	Yes
D2	Rainy	Warm	High	Yes
D3	Sunny	Cold	Normal	Yes
D4	Rainy	Cold	High	No
D5	Sunny	Warm	High	No

- **Training Data ($N = 5$):**
- **Test Instance:** $\mathbf{x} = (\text{Sunny, Warm, High})$
- **Goal:** Predict Play? (Yes/No)

Class: Yes ($N_{yes} = 3$)

- $P(\text{Yes}) = 3/5$
- $P(\text{Sunny}|\text{Yes}) = 2/3$
- $P(\text{Warm}|\text{Yes}) = 2/3$
- $P(\text{High}|\text{Yes}) = 1/3$

$$\text{Score}_{Yes} = \frac{3}{5} \times \frac{2}{3} \times \frac{2}{3} \times \frac{1}{3} = \mathbf{0.089}$$

Class: No ($N_{no} = 2$)

- $P(\text{No}) = 2/5$
- $P(\text{Sunny}|\text{No}) = 1/2$
- $P(\text{Warm}|\text{No}) = 1/2$
- $P(\text{High}|\text{No}) = 2/2 = 1.0$

$$\text{Score}_{No} = \frac{2}{5} \times \frac{1}{2} \times \frac{1}{2} \times 1 = \mathbf{0.1}$$

Prediction: $0.1 > 0.089 \implies$ **No**

Scenario 2: Slight Data Change... Big Problem!

- Suppose, in our "Yes" examples ($N = 3$), the "Temp" column always sees **Cold** days.

Day	Sky	Temp	Humid	Play?
D1	Sunny	**Cold**	Normal	Yes
D2	Rainy	**Cold**	High	Yes
D3	Sunny	**Cold**	Normal	Yes
...	...	...	...	No

- **Test Input:** $\mathbf{x} = (\text{Sunny}, \textbf{Warm}, \text{High})$

Scenario 2: Slight Data Change... Big Problem!

- Suppose, in our "Yes" examples ($N = 3$), the "Temp" column always sees **Cold** days.

Day	Sky	Temp	Humid	Play?
D1	Sunny	**Cold**	Normal	Yes
D2	Rainy	**Cold**	High	Yes
D3	Sunny	**Cold**	Normal	Yes
...	...	...	...	No

- **Test Input:** $\mathbf{x} = (\text{Sunny}, \textbf{Warm}, \text{High})$

Calculation for $P(\text{Yes}|\mathbf{x})$

$$P(\text{Yes}) \times P(\text{Sunny}|\text{Yes}) \times P(\text{Warm}|\text{Yes}) \times \dots$$

$$= \frac{3}{5} \times \frac{2}{3} \times \frac{\mathbf{0}}{\mathbf{3}} \times \cdots = \mathbf{0}$$

Scenario 2: Slight Data Change... Big Problem!

§ Suppose, in our "Yes" examples ($N = 3$), the "Temp" column always sees **Cold** days.

Day	Sky	Temp	Humid	Play?
D1	Sunny	**Cold**	Normal	Yes
D2	Rainy	**Cold**	High	Yes
D3	Sunny	**Cold**	Normal	Yes
...	...	...	...	No

§ **Test Input:** $\mathbf{x} = (\text{Sunny}, \textbf{Warm}, \text{High})$

Calculation for $P(\text{Yes}|\mathbf{x})$

$$P(\text{Yes}) \times P(\text{Sunny}|\text{Yes}) \times P(\text{Warm}|\text{Yes}) \times \dots$$

$$= \frac{3}{5} \times \frac{2}{3} \times \frac{\mathbf{0}}{\mathbf{3}} \times \dots = \mathbf{0}$$

§ **The "Zero Frequency" Problem**: The probability is zero just because we haven't seen a "Warm" day for "Play=Yes" *yet*. This vetoes all other strong evidence (like Sunny!).

The Fix: Laplace Smoothing

- Are you thinking "such a complicated term this is!! LAPLACE SMOOTHING!"
- **Idea:** Add a "virtual count" of 1 to every value.

$$\hat{P}(x_i|y) = \frac{\text{Count}(x_i, y) + 1}{\text{Count}(y) + |V|}$$

where $|V|$ is the "**number**" of values that the feature can take.

The Fix: Laplace Smoothing

- Are you thinking “such a complicated term this is!! LAPLACE SMOOTHING!”
- **Idea:** Add a “virtual count” of 1 to every value.

$$\hat{P}(x_i|y) = \frac{\text{Count}(x_i, y) + 1}{\text{Count}(y) + |V|}$$

where $|V|$ is the “**number**” of values that the feature can take.
Re-calculating $P(\textbf{Warm}|\textbf{Yes})$:

  - Count(Warm, Yes) $= 0$
  - Count(Yes) $= 3$
  - Possible Temps $(|V|) = 2$ {Warm, Cold}

$$\hat{P}(\text{Warm}|\text{Yes}) = \frac{0+1}{3+2} = \frac{1}{5} = \mathbf{0.2}$$

Now the probability is small, but non-zero!

Handling Continuous Features

- So far, we have discussed Naive Bayes with discrete features (*e.g.*, word counts, sky condition).
- **Question:** What if our features $\mathbf{x} = (x_1, \ldots, x_d)$ are **continuous** real numbers?
  - Example: Classifying if a person is "Healthy" or "Sick".
  - Features: Height (cm), Weight (kg), Temperature (°F).
- We cannot use simple counting/tables because the probability of observing an exact real number (e.g., height = 170.0001 cm) is zero.

Handling Continuous Features

- So far, we have discussed Naive Bayes with discrete features (*e.g.*, word counts, sky condition).
- **Question:** What if our features $\mathbf{x} = (x_1, \ldots, x_d)$ are **continuous** real numbers?
  - Example: Classifying if a person is "Healthy" or "Sick".
  - Features: Height (cm), Weight (kg), Temperature (°F).
- We cannot use simple counting/tables because the probability of observing an exact real number (e.g., height = 170.0001 cm) is zero.
- **Solution:** We need a **probability density function (PDF)** to model $P(x_j|Y)$. The most common choice is the **Gaussian (Normal) Distribution**.

Gaussian Naive Bayes Assumption

We assume that for each class $y \in \{0, 1\}$, the continuous features x_i are distributed according to a Gaussian distribution.

Model Assumptions

- **Class Prior:** $Y \sim \text{Bernoulli}(\phi)$
- **Conditional Distributions:**

$$P(x_j|Y = y) = \frac{1}{\sqrt{2\pi\sigma_{jy}^2}} \exp\left(-\frac{(x_j - \mu_{jy})^2}{2\sigma_{jy}^2}\right)$$

Gaussian Naive Bayes Assumption

We assume that for each class $y \in \{0, 1\}$, the continuous features x_i are distributed according to a Gaussian distribution.

Model Assumptions

- **Class Prior:** $Y \sim \text{Bernoulli}(\phi)$
- **Conditional Distributions:**

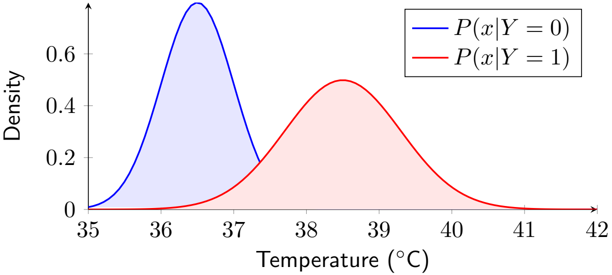
$$P(x_j|Y = y) = \frac{1}{\sqrt{2\pi\sigma_{jy}^2}} \exp\left(-\frac{(x_j - \mu_{jy})^2}{2\sigma_{jy}^2}\right)$$

This means for each feature j and each class y, we need to estimate two parameters:

- Mean μ_{jy}: The average value of feature j for class y.
- Variance σ_{jy}^2: How spread out feature j is for class y.

Visualizing Gaussian Naive Bayes

Imagine we have one feature x (e.g., Temperature) and two classes ($Y = 0$ Healthy, $Y = 1$ Sick).
We model each class as a "Bell Curve":



Learning Parameters (MLE)

- How do we estimate the parameters from data?
- We compute the sample mean and sample variance for each class.

Learning Parameters (MLE)

- How do we estimate the parameters from data?
- We compute the sample mean and sample variance for each class.
- Given dataset $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)})\}$:
  - **Means:**

$$\hat{\mu}_{jy} = \frac{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = y) \cdot x_j^{(i)}}{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = y)}$$

    (Average of feature j for all examples where class is y)
  - **Variances:**

$$\hat{\sigma}_{jy}^2 = \frac{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = y) \cdot (x_j^{(i)} - \hat{\mu}_{jy})^2}{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = y)}$$

    (Variance of feature j for all examples where class is y)

Concrete Example

Dataset (Height in feet):

- Class 0: $\{5.0, 5.5, 6.0\}$
- Class 1: $\{6.0, 6.2\}$

Concrete Example

Dataset (Height in feet):

- Class 0: {5.0, 5.5, 6.0}
- Class 1: {6.0, 6.2}

Step 1: Estimate Means

- $\hat{\mu}_0 = \frac{5.0+5.5+6.0}{3} = 5.5$
- $\hat{\mu}_1 = \frac{6.0+6.2}{2} = 6.1$

Concrete Example

Dataset (Height in feet):

- Class 0: {5.0, 5.5, 6.0}
- Class 1: {6.0, 6.2}

Step 1: Estimate Means

- $\hat{\mu}_0 = \frac{5.0+5.5+6.0}{3} = 5.5$
- $\hat{\mu}_1 = \frac{6.0+6.2}{2} = 6.1$

Step 2: Estimate Variances

- $\hat{\sigma}_0^2 = \frac{(5.0-5.5)^2+(5.5-5.5)^2+(6.0-5.5)^2}{3} = \frac{0.25+0+0.25}{3} = 0.167$
- $\hat{\sigma}_1^2 = \frac{(6.0-6.1)^2+(6.2-6.1)^2}{2} = \frac{0.01+0.01}{2} = 0.01$

Concrete Example

Dataset (Height in feet):

- Class 0: {5.0, 5.5, 6.0}
- Class 1: {6.0, 6.2}

Step 1: Estimate Means

- $\hat{\mu}_0 = \frac{5.0+5.5+6.0}{3} = 5.5$
- $\hat{\mu}_1 = \frac{6.0+6.2}{2} = 6.1$

Step 2: Estimate Variances

- $\hat{\sigma}_0^2 = \frac{(5.0-5.5)^2+(5.5-5.5)^2+(6.0-5.5)^2}{3} = \frac{0.25+0+0.25}{3} = 0.167$
- $\hat{\sigma}_1^2 = \frac{(6.0-6.1)^2+(6.2-6.1)^2}{2} = \frac{0.01+0.01}{2} = 0.01$

Prediction for new $x = 5.8$: Compute $P(x = 5.8|Y = 0)P(Y = 0)$ vs $P(x = 5.8|Y = 1)P(Y = 1)$ using Gaussian formula.

$$P(x = 5.8|Y = 0) = \frac{1}{\sqrt{2\pi(0.167)}} e^{-\frac{(5.8-5.5)^2}{2(0.167)}} \approx 0.98 \times e^{-0.27} \approx \mathbf{0.75}$$

$$P(x = 5.8|Y = 1) = \frac{1}{\sqrt{2\pi(0.01)}} e^{-\frac{(5.8-6.1)^2}{2(0.01)}} \approx 3.99 \times e^{-4.5} \approx \mathbf{0.04}$$

(Ignoring priors for simplicity, Class 0 is much more likely)

The "Linear" Connection

- **Question:** What does the decision boundary look like?
- If we assume the variance is **shared** across classes ($\sigma_{j0}^2 = \sigma_{j1}^2 = \sigma^2$), something magical happens.

The "Linear" Connection

- **Question:** What does the decision boundary look like?
- If we assume the variance is **shared** across classes ($\sigma_{j0}^2 = \sigma_{j1}^2 = \sigma^2$), something magical happens.
- Recall the decision rule involves the log-ratio:

$$\log \frac{P(Y=1|x)}{P(Y=0|x)} = \log \frac{P(Y=1)}{P(Y=0)} + \sum_j \log \frac{P(x_j|Y=1)}{P(x_j|Y=0)}$$

The "Linear" Connection

- **Question:** What does the decision boundary look like?
- If we assume the variance is **shared** across classes ($\sigma_{j0}^2 = \sigma_{j1}^2 = \sigma^2$), something magical happens.
- Recall the decision rule involves the log-ratio:

$$\log \frac{P(Y=1|x)}{P(Y=0|x)} = \log \frac{P(Y=1)}{P(Y=0)} + \sum_j \log \frac{P(x_j|Y=1)}{P(x_j|Y=0)}$$

- Substituting the Gaussian formula:

$$\log \frac{\exp(-(x_j-\mu_{j1})^2/2\sigma^2)}{\exp(-(x_j-\mu_{j0})^2/2\sigma^2)} = \frac{-(x_j-\mu_{j1})^2 + (x_j-\mu_{j0})^2}{2\sigma^2}$$

- The quadratic terms x_j^2 cancel out! We are left with terms linear in x_j.

The "Logistic Regression" Connection

Since the quadratic terms cancel (under shared variance assumption), the log-odds is a linear function:

$$\log \frac{P(Y=1|\mathbf{x})}{P(Y=0|\mathbf{x})} = \boldsymbol{\theta}^T \mathbf{x} + b$$

This implies the posterior probability is a **Sigmoid**:

$$P(Y=1|\mathbf{x}) = \frac{1}{1+e^{-(\boldsymbol{\theta}^T \mathbf{x}+b)}}$$

The "Logistic Regression" Connection

Since the quadratic terms cancel (under shared variance assumption), the log-odds is a linear function:

$$\log \frac{P(Y=1|\mathbf{x})}{P(Y=0|\mathbf{x})} = \boldsymbol{\theta}^T \mathbf{x} + b$$

This implies the posterior probability is a **Sigmoid**:

$$P(Y=1|\mathbf{x}) = \frac{1}{1+e^{-(\boldsymbol{\theta}^T \mathbf{x}+b)}}$$

This specific model—Naive Bayes with continuous features assumed to be Gaussian—is formally known as **Gaussian Discriminant Analysis (GDA)**.

Key Takeaway

Gaussian Naive Bayes (with shared variance) is a **Generative** classifier that produces a **Linear** decision boundary, exactly like Logistic Regression!

- GDA: Learns $P(X|Y)$ (Means/Variances) $\rightarrow$ gets $\boldsymbol{\theta}$.
- Logistic Regression: Learns $\boldsymbol{\theta}$ directly.

Summary: Naive Bayes & GDA

- **Generative Approach:** We model $P(\mathbf{x}|y)$ and $P(y)$ to estimate the joint distribution $P(\mathbf{x}, y)$, then use Bayes Rule for classification.
- **Discrete Features:** Use **Naive Bayes** (Bernoulli/Multinomial) with Laplace Smoothing to handle zero probabilities.
- **Continuous Features:** Use **Gaussian Discriminant Analysis (GDA)**. We estimate Means and Variances.
- **Key Insight:** If we assume shared covariance in GDA, the decision boundary is linear, and the posterior form is identical to **Logistic Regression**.
  - **GDA** is more data-efficient (lower variance) if the Gaussian assumption is true.
  - **Logistic Regression** is more robust (lower bias) if the assumption is wrong.
- GDA with shared covariance is often called **Linear Discriminant Analysis (LDA)**. Allowing different covariances, makes it **Quadratic Discriminant Analysis (QDA)**.

Next Class: Bias and Variance