Naive Bayes Classifier

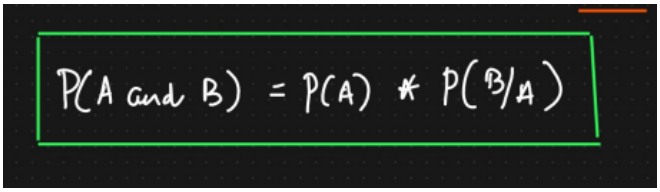
Naive Bayes is a powerful and simple probabilistic classifier based on Bayes' Theorem. It's widely used in various applications due to its effectiveness, speed, and ease of implementation.

Fundamental Probability Concepts

1. **Probability:** A measure of the likelihood of an event occurring. It ranges from 0 (impossible event) to 1 (certain event).
2. **Sample Space (S):** The set of all possible outcomes of an experiment.
3. **Event (A, B):** A subset of the sample space.
4. **Intersection (A ∩ B):** The event where *both* events A and B occur.
5. **Conditional Probability P(A|B):** The probability of event A occurring *given that* event B has already occurred.

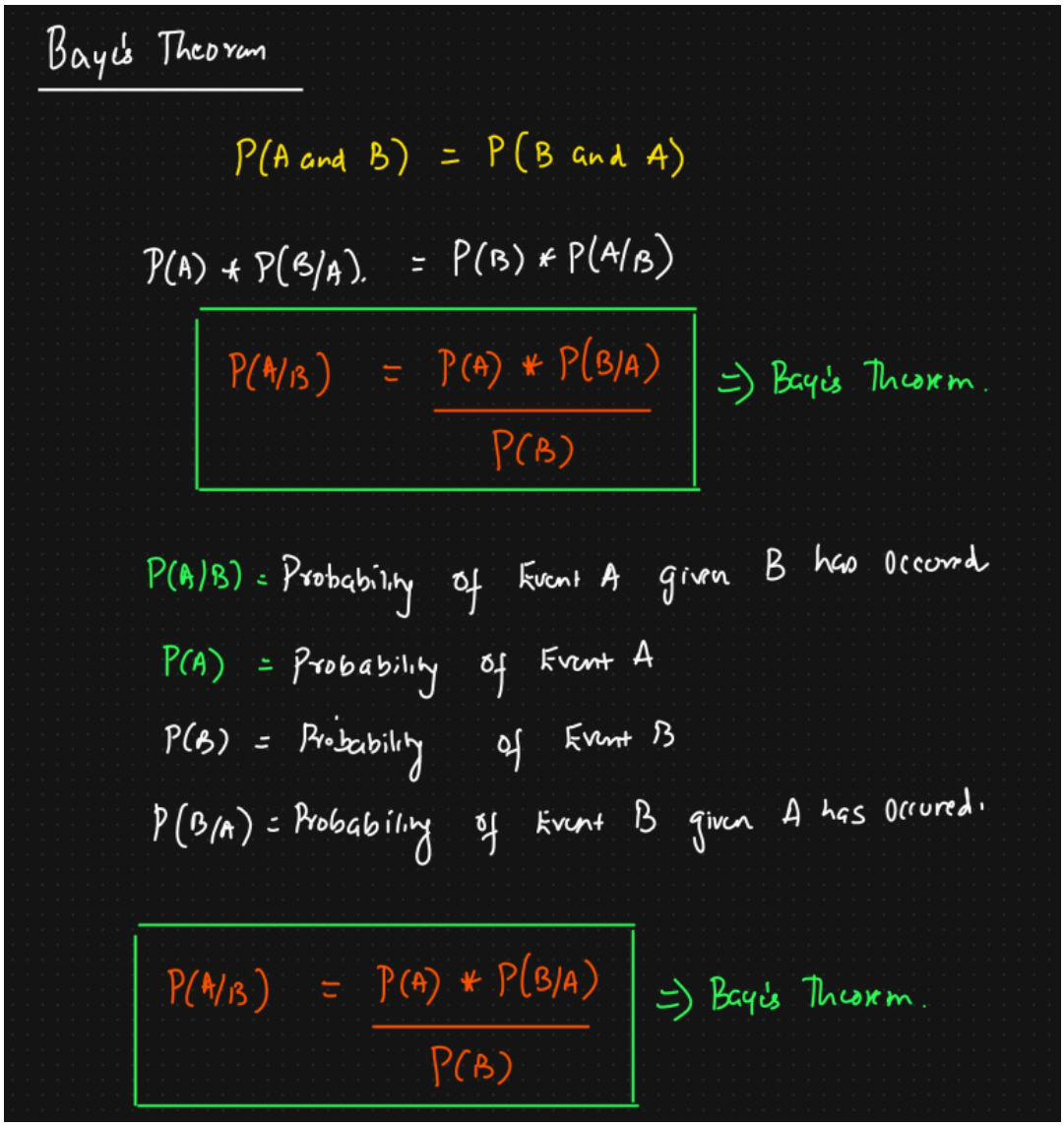
Independent vs. Dependent Events

1. **Independent Events:** Two events A and B are independent if the occurrence of one event does *not* affect the probability of the other event occurring.
    - **Mathematical Definition:** A and B are independent if and only if: $P(A \cap B) = P(A) \times P(B)$
    - **Conditional Probability Relationship:** If A and B are independent:
        - $P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{P(A)P(B)}{P(B)} = P(A)$
        - $P(B|A) = \frac{P(B \cap A)}{P(A)} = \frac{P(B)P(A)}{P(A)} = P(B)$ (Knowing B happened doesn't change the probability of A, and vice versa).
    - **Example:** Tossing a fair coin twice. The outcome of the first toss (Heads or Tails) does not influence the outcome of the second toss. $P(\text{Head on 2nd toss} \mid \text{Head on 1st toss}) = P(\text{Head on 2nd toss}) = 0.5$.
2. **Dependent Events:** Two events A and B are dependent if the occurrence of one event *does* affect the probability of the other event occurring.
    - **Mathematical Property:** $P(A \cap B) = P(A) \times P(B)$
    - **Conditional Probability Relationship:** The basic definition $P(A \mid B) = P(B)P(A \cap B)$ holds. Knowing B occurred changes the probability of A.
    - **Example:** Drawing two cards from a standard deck *without* replacement. Let A be "drawing a King first" and B be "drawing a King second".
        - $P(A) = 4/52$.
        - If event A occurs (a King is drawn first), then there are only 51 cards left, and only 3 Kings. So, $P(B \mid A) = 3/51$.
        - The probability of B depends on whether A happened.
    - **Calculating Joint Probability for Dependent Events:** From the conditional probability formula, we get: $P(A \cap B) = P(A \mid B) \times P(B) = P(B \mid A) \times P(A)$

- 

Bayes' Theorem

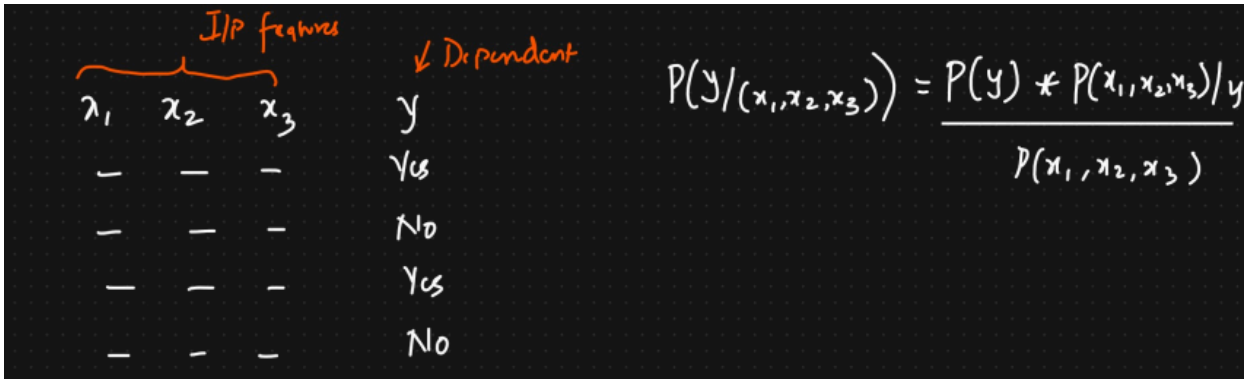
Bayes' Theorem provides a way to update the probability of a hypothesis based on new evidence. It relates the conditional probability of two events.



Derivation:

- From the definition of conditional probability, we have:
  - $P(A|B) = \frac{P(A \cap B)}{P(B)} \implies P(A \cap B) = P(A|B)P(B)$ **(Equation 1)**
  - $P(B|A) = \frac{P(B \cap A)}{P(A)} \implies P(B \cap A) = P(B|A)P(A)$ **(Equation 2)**
- Since the intersection is commutative ($P(A \cap B) = P(B \cap A)$), we can equate the right-hand sides of Equation 1 and Equation 2: $P(A|B)P(B) = P(B|A)P(A)$
- Dividing by $P(B)$ (assuming $P(B) \neq 0$), we get Bayes' Theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$



Naive Bayes Algorithm

1. Introduction:

- **What it is:** Naive Bayes is a simple yet effective and commonly used **probabilistic classifier** based on **Bayes' Theorem**.
- **Core Idea:** It calculates the probability of a data point belonging to each class and assigns the class with the highest probability.
- **"Naive" Aspect:** It makes a strong ("naive") assumption about the **independence of features**.

2. Bayes' Theorem:

- **Goal:** Given a set of features X=(X1,X2,...,Xn), we want to predict the class label C (from a set of possible classes Ck). We aim to find the class Ck that is most probable given the observed features. In other words, we want to maximize the posterior probability P(Ck|X1,...,Xn).

- **Applying Bayes' Theorem:**

$$P(C_k|X_1, ..., X_n) = \frac{P(X_1, ..., X_n|C_k)P(C_k)}{P(X_1, ..., X_n)}$$

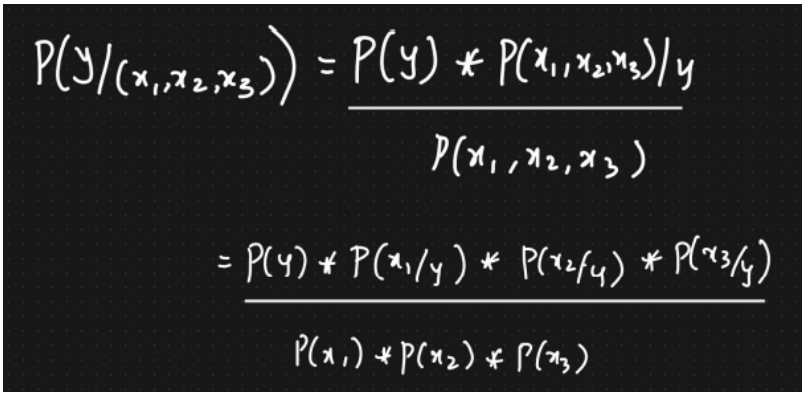
- $P(C_k|X_1, ..., X_n)$: Posterior probability of class C_k given features.
- $P(X_1, ..., X_n|C_k)$: Likelihood of observing the features given class C_k.
- $P(C_k)$: Prior probability of class C_k.
- $P(X_1, ..., X_n)$: Probability of observing the features (Evidence).

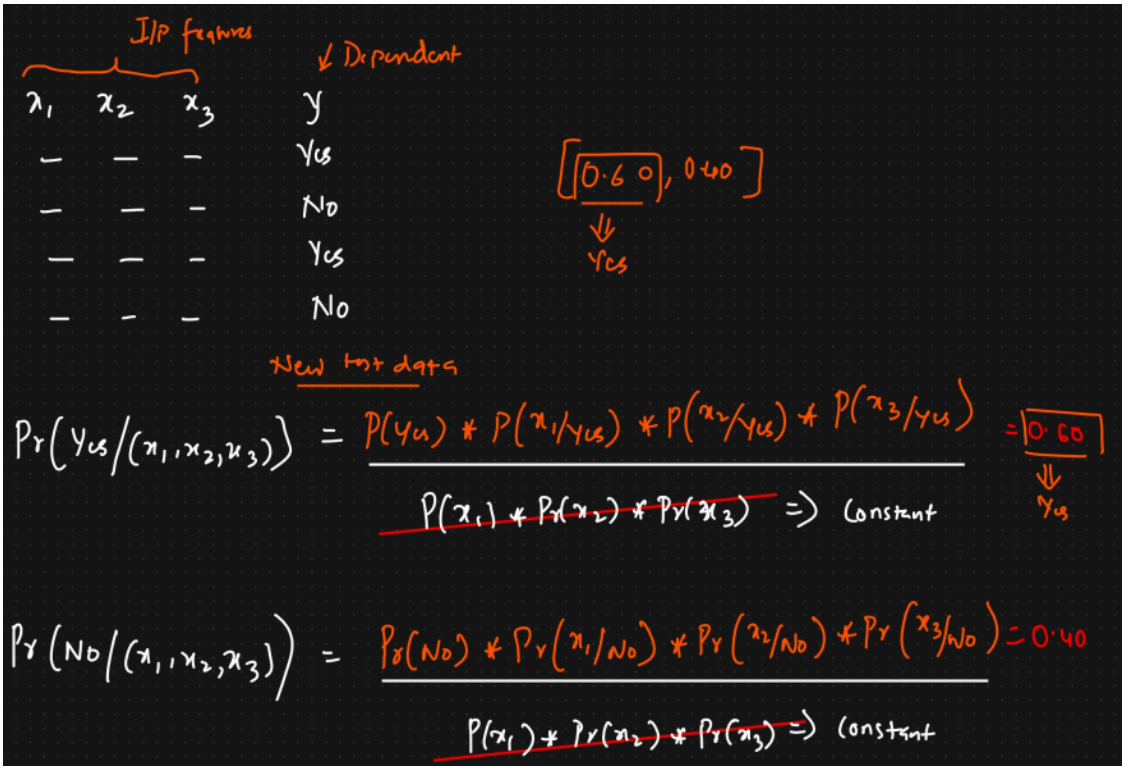
3. The "Naive" Independence Assumption:

- **Assumption:** Naive Bayes assumes that all features `X = (x1, x2, ..., xn)` are **conditionally independent** given the class `C`.
- **Meaning:** The presence or value of one feature does not affect the presence or value of another feature, given the class.
- **Mathematical Simplification:** This assumption dramatically simplifies the likelihood calculation: `P(X | C) = P(x1, x2, ..., xn | C) = P(x1 | C) * P(x2 | C) * ... * P(xn | C)`
- **Reality Check:** This independence assumption rarely holds true in real-world data. However, the algorithm often performs surprisingly well even when the assumption is violated.

4. How Naive Bayes Classification Works:

- **Goal:** For a new data point with features `X`, find the class `C` that maximizes the posterior probability `P(C | X)`.
- **Calculation:** Using Bayes' theorem and the naive assumption: `P(C | X) ∝ P(C) * P(x1 | C) * P(x2 | C) * ... * P(xn | C)`
    - We calculate this value for *every* class.
    - `P(X)` (the denominator) is constant for all classes for a given data point, so it can be ignored when just comparing the posterior probabilities.
- **Prediction:** The class `C` that yields the highest value from the calculation above is assigned as the predicted class for the data point `X`.
- **Training:** The training phase involves calculating the prior probabilities `P(C)` for each class and the likelihood probabilities `P(xi | C)` for each feature `xi` given each class `C`, usually based on frequencies in the training data.





Golf Dataset Example

Let's consider a dataset for predicting whether to play golf based on weather conditions:

Outlook	Temperature	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No

Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

The dataset is divided into two parts, namely, feature matrix and the response vector.

Assumptions of the **Naive Bayes** algorithm

1. **Feature Independence**:
   Each feature is assumed to be **independent** of the others given the class label.

2. **Equal Contribution**:
   All features contribute **equally and independently** to the outcome.

3. **No Feature Interaction**:
   Assumes **no interaction** between features (i.e., the presence/absence of one feature does not affect another).

4. **Data Distribution**:
   For continuous data, it assumes **normal (Gaussian) distribution** (in Gaussian Naive Bayes).

Now, with regards to our dataset, we can apply Bayes' theorem in following way:

Outlook

	Yes	No	P(yes)	P(no)
Sunny	3	2	3/10	2/4
Overcast	4	0	4/10	0/4
Rainy	3	2	3/10	2/4
Total	10	4	100%	100%

Temperature

	Yes	No	P(yes)	P(no)
Hot	2	2	2/9	2/5
Mild	4	2	4/9	2/5
Cool	3	1	3/9	1/5
Total	9	5	100%	100%

Humidity

	Yes	No	P(yes)	P(no)
High	3	4	3/9	4/5
Normal	6	1	6/9	1/5
Total	9	5	100%	100%

Wind

	Yes	No	P(yes)	P(no)
False	6	2	6/9	2/5
True	3	3	3/9	3/5
Total	9	5	100%	100%

Play		P(Yes)/P(No)
Yes	9	9/14
No	5	5/14
Total	14	100%

where, y is class variable and X is a dependent feature vector (of size n) where:

$$P(No|today) = \frac{P(SunnyOutlook|No)P(HotTemperature|No)P(NormalHumidity|No)P(NoWind|No)P(No)}{P(today)}$$

Just to clear, an example of a feature vector and corresponding class variable can be: (refer 1st row of dataset)

X = (Rainy, Hot, High, False)

y = No

So basically, P(y | X)here means, the probability of “Not playing golf” given that the weather conditions are “Rainy outlook”, “Temperature is hot”, “high humidity” and “no wind”.

Hence, we reach to the result:

$$P(y|x_1,\ldots,x_n) = \frac{P(x_1|y)P(x_2|y)\ldots P(x_n|y)P(y)}{P(x_1)P(x_2)\ldots P(x_n)}$$

which can be expressed as:

$$P(y|x_1,\ldots,x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_1)P(x_2)\ldots P(x_n)}$$

Now, as the denominator remains constant for a given input, we can remove that term:

$$P(y|x_1,\ldots,x_n) \propto P(y)\prod_{i=1}^{n} P(x_i|y)$$

We need to find$P(x_i|y_j)$for each x_i in X andy_j in y. All these calculations have been demonstrated in the tables below:

$P(Yes|today) \propto \frac{3}{9}.\frac{2}{9}.\frac{6}{9}.\frac{6}{9}.\frac{9}{14} \approx 0.02116$

So, in the figure above, we have calculated $P(x_i\ |y_j)$ for each x_i in X and y_j in y manually in the tables 1-4. For example, probability of playing golf given that the temperature is cool, i.e P(temp. = cool | play golf = Yes) = 3/9.

Lets Solve this Problem

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Outlook

	Yes	No	P(E/Yes)	Pr(E/No)
Sunny	2	3	2/9	3/5
Overcast	4	0	4/9	0/5
Rain	3	2	3/9	2/5

Temperature → Just (Sunny, Hot) → O/P

PLAY (Y/N)

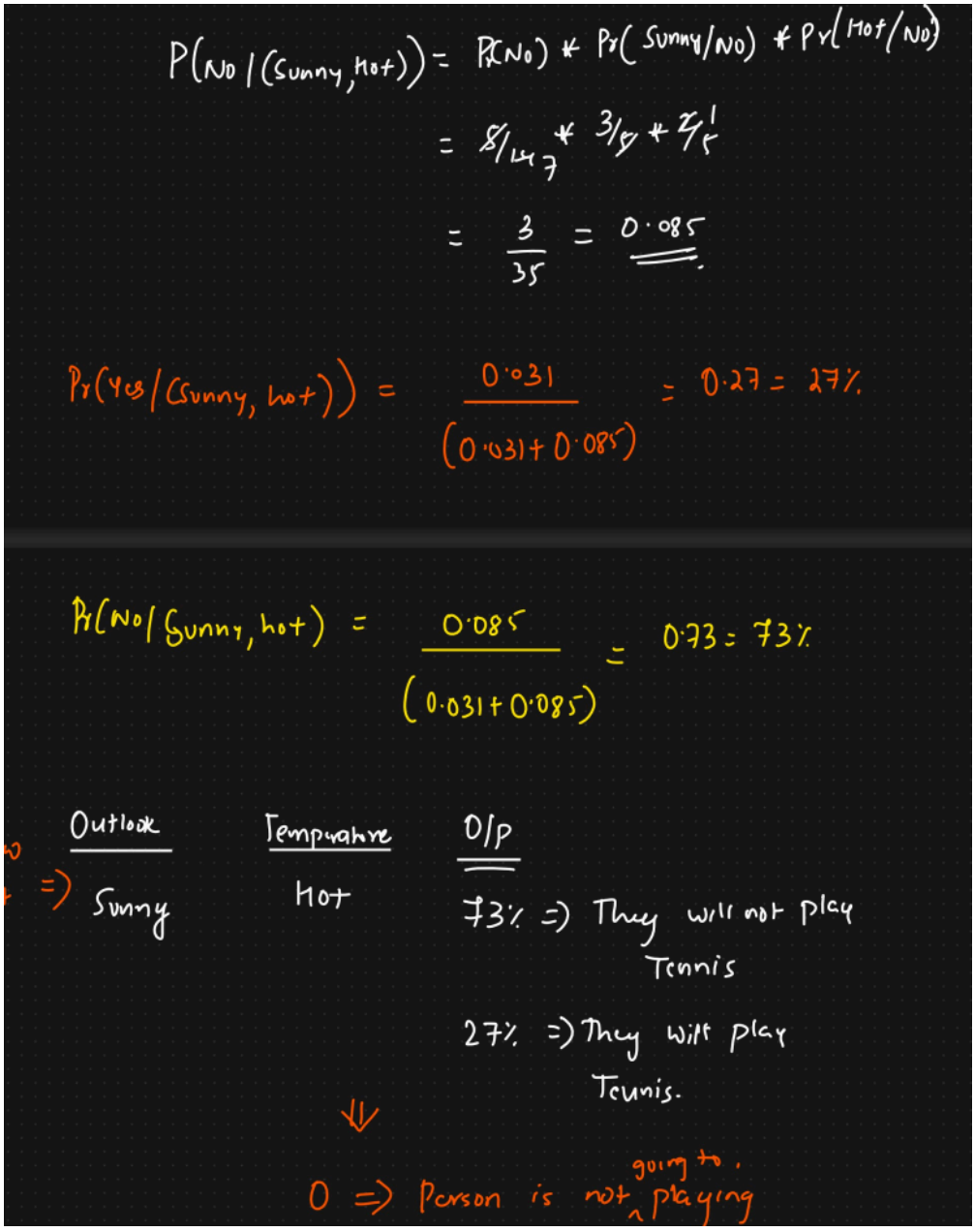
	Yes	No	P(E/Yes)	P(E/No)
Hot	2	2	2/9	2/5
Mild	4	2	4/9	4/5
Cool	3	1	3/9	3/5

		P(Yes)	P(No)
Yes	9	9/14	5/14
No	5		

$$P(Yes/Sunny, Hot) = \frac{P(Yes) * Pr(Sunny/Yes) * Pr(Hot/Yes)}{\cancel{Pr(Sunny) * Pr(Hot)}}$$

$$= A/\text{ll}_7 * \frac{2}{\cancel{9}} * \frac{2}{9}$$

$$= \frac{2}{63} = 0.031$$



5. Types of Naive Bayes Classifiers:

The choice depends on the nature of the features:

- **Gaussian Naive Bayes:**
  - Assumes features follow a **Gaussian (normal) distribution**.
  - Used for **continuous** features.
  - Calculates the mean and standard deviation of each feature for each class during training.
  - Use case: Sensor measurements, physical properties (density, hardness, etc.)
- **Multinomial Naive Bayes:**

    - Typically used for **discrete counts**.
    - Common in **text classification** (e.g., word counts in documents).
    - Often uses **Laplace (or Additive) smoothing** to handle cases where a feature count is zero in the training data for a given class.
    - Use case: Text classification, mineral composition percentages, frequency-based features
- **Bernoulli Naive Bayes:**
    - Used for **binary/boolean features** (feature is present or absent).
    - Also common in text classification (e.g., presence/absence of a word).
    - Use case: Presence/absence of minerals, binary sensor readings, pass/fail tests

6. Advantages:

- **Simple and Fast:** Easy to implement and computationally efficient for both training and prediction.
- **Requires Less Training Data:** Can perform well even with relatively small datasets.
- **Scales Well:** Handles high-dimensional data (many features) effectively, like in text classification.
- **Good Performance:** Often works surprisingly well even if the independence assumption isn't fully met.
- Handles different feature types through variants (Gaussian, Multinomial, Bernoulli).

7. Disadvantages:

- **Unrealistic Independence Assumption:** The core assumption is often violated in reality, which is its main theoretical limitation (though often not a practical one).
- **Zero-Frequency Problem:** If a specific feature value doesn't appear with a specific class in the training data, its conditional probability becomes zero, potentially wiping out the entire posterior probability. (Mitigated by smoothing techniques like Laplace smoothing).
- **Potentially Poor Probability Estimates:** While classification ranking might be good, the actual estimated probabilities can be inaccurate if the independence assumption is strongly violated.
- **Sensitivity to Feature Distribution:** Performance depends on features matching the distribution assumed by the chosen variant (e.g., Gaussian for Gaussian NB).

8. Common Applications:

- Text Classification (Spam Filtering, Sentiment Analysis, Topic Categorization)
- Medical Diagnosis
- Recommendation Systems
- Fraud Detection