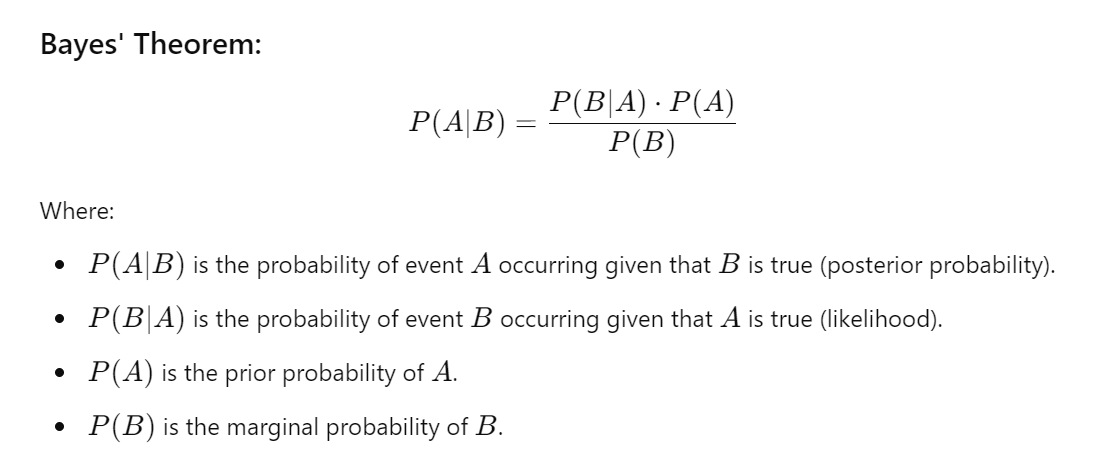
**Bayes Theorem**

**Aim: Implement Bayes Theorem using Python.**



Procedure:

To implement **Bayes' Theorem** on the **Iris dataset** (iris.csv), we first need to load the dataset and define events A and B. For the sake of this example, we can define:

* A: The class of the iris flower (e.g., setosa, versicolor, virginica).
* B: Some observed feature value (e.g., petal length, sepal width).

Let's assume we want to calculate the probability that a flower belongs to the class setosa given that its **sepal length** is greater than some threshold (say 5.0 cm).

**Steps:**

1. Load the iris.csv dataset.
2. Define the event (flower class) and feature (sepal length)
3. Compute prior P(A), likelihood P(B∣A), and marginal probability P(B).
4. Apply Bayes' Theorem.

**Events:**

A: Flower is Setosa

B: sepal length being greater than 5.0 cm

**Steps Breakdown:**

1. **Prior Probability P(A)**: The probability of the flower being setosa (without any conditions).
2. **Likelihood P(B∣A)**: The probability of the sepal length being greater than 5.0 cm, given the flower is setosa.
3. **Marginal Probability P(B)**: The probability of the sepal length being greater than 5.0 cm, across the whole dataset.
4. **Posterior Probability P(A∣B)**: The probability that a flower is setosa, given that its sepal length is greater than 5.0 cm.

import pandas as pd

def bayes\_theorem(prior\_A, likelihood\_B\_given\_A, marginal\_B):

"""

Calculate the posterior probability using Bayes' Theorem.

:param prior\_A: P(A) - Prior probability of A

:param likelihood\_B\_given\_A: P(B|A) - Likelihood of B given A

:param marginal\_B: P(B) - Marginal probability of B

:return: P(A|B) - Posterior probability of A given B

"""

return (likelihood\_B\_given\_A \* prior\_A) / marginal\_B

# Load the Iris dataset

def load\_iris\_dataset(file\_path):

return pd.read\_csv(file\_path)

# Calculate prior probability P(A)

def calculate\_prior(data, class\_col, class\_value):

return len(data[data[class\_col] == class\_value]) / len(data)

# Calculate likelihood P(B|A)

def calculate\_likelihood(data, class\_col, class\_value, feature\_col, feature\_condition):

subset = data[data[class\_col] == class\_value]

return len(subset[subset[feature\_col] > feature\_condition]) / len(subset)

# Calculate marginal probability P(B)

def calculate\_marginal(data, feature\_col, feature\_condition):

return len(data[data[feature\_col] > feature\_condition]) / len(data)

# Apply Bayes' Theorem on the Iris dataset

def apply\_bayes\_to\_iris(file\_path, class\_col, class\_value, feature\_col, feature\_condition):

# Load dataset

data = load\_iris\_dataset(file\_path)

# Calculate prior P(A)

prior\_A = calculate\_prior(data, class\_col, class\_value)

# Calculate likelihood P(B|A)

likelihood\_B\_given\_A = calculate\_likelihood(data, class\_col, class\_value, feature\_col, feature\_condition)

# Calculate marginal probability P(B)

marginal\_B = calculate\_marginal(data, feature\_col, feature\_condition)

# Apply Bayes' Theorem

posterior\_A\_given\_B = bayes\_theorem(prior\_A, likelihood\_B\_given\_A, marginal\_B)

return posterior\_A\_given\_B

# Example usage:

# Assume we want to calculate the probability P(Class='setosa' | SepalLength > 5.0)

file\_path = 'iris.csv' # Path to the iris dataset file

class\_col = 'species' # The column representing the class (A)

class\_value = 'setosa' # The class value we're interested in (A)

feature\_col = 'sepal\_length' # The feature we're using (B)

feature\_condition = 5.0 # The condition on the feature (B > 5.0)

# Calculate posterior probability P(setosa|sepal\_length > 5.0)

posterior\_probability = apply\_bayes\_to\_iris(file\_path, class\_col, class\_value, feature\_col, feature\_condition)

print(f"P({class\_value} | {feature\_col} > {feature\_condition}) = {posterior\_probability:.4f}")