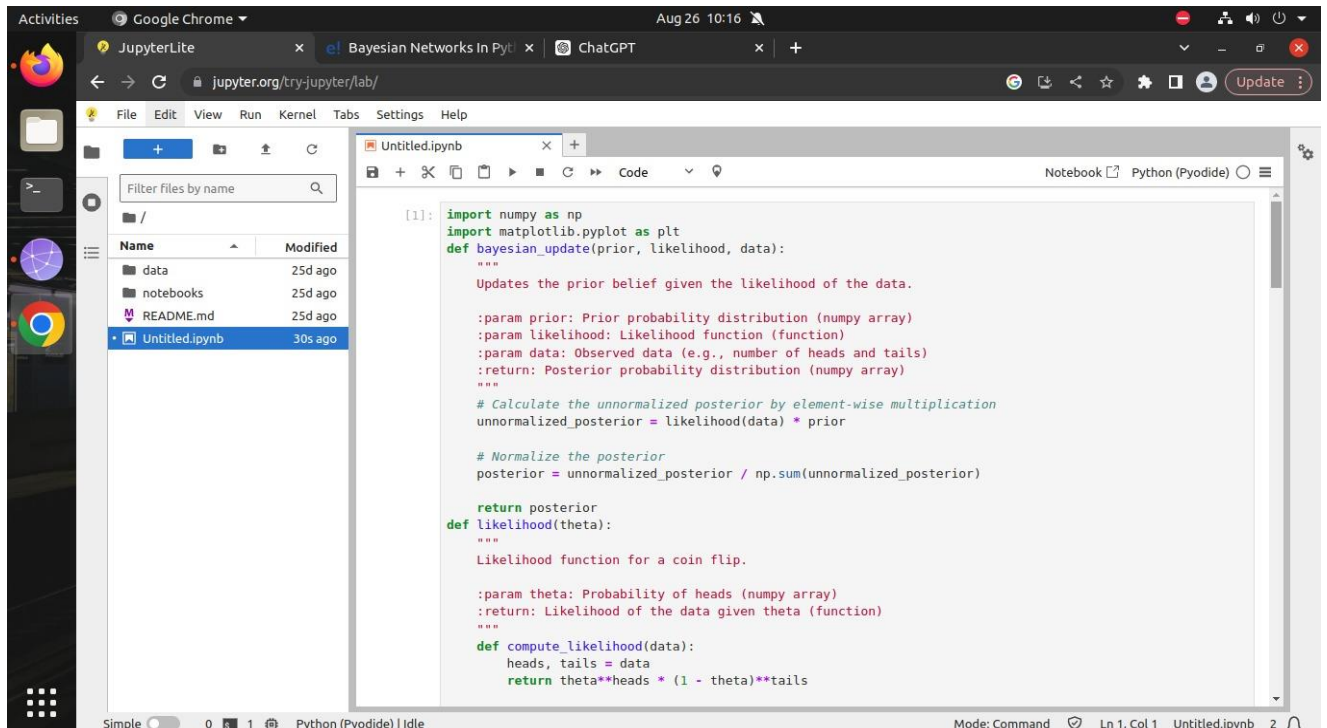


## EXPERIMENT NO: 4

### IMPLEMENTATION OF BAYESIAN ALGORITHM



The screenshot shows the JupyterLite interface with a file browser on the left and a code editor on the right. The file browser lists files: data, notebooks, README.md, and Untitled.ipynb. The code editor shows the following Python code:

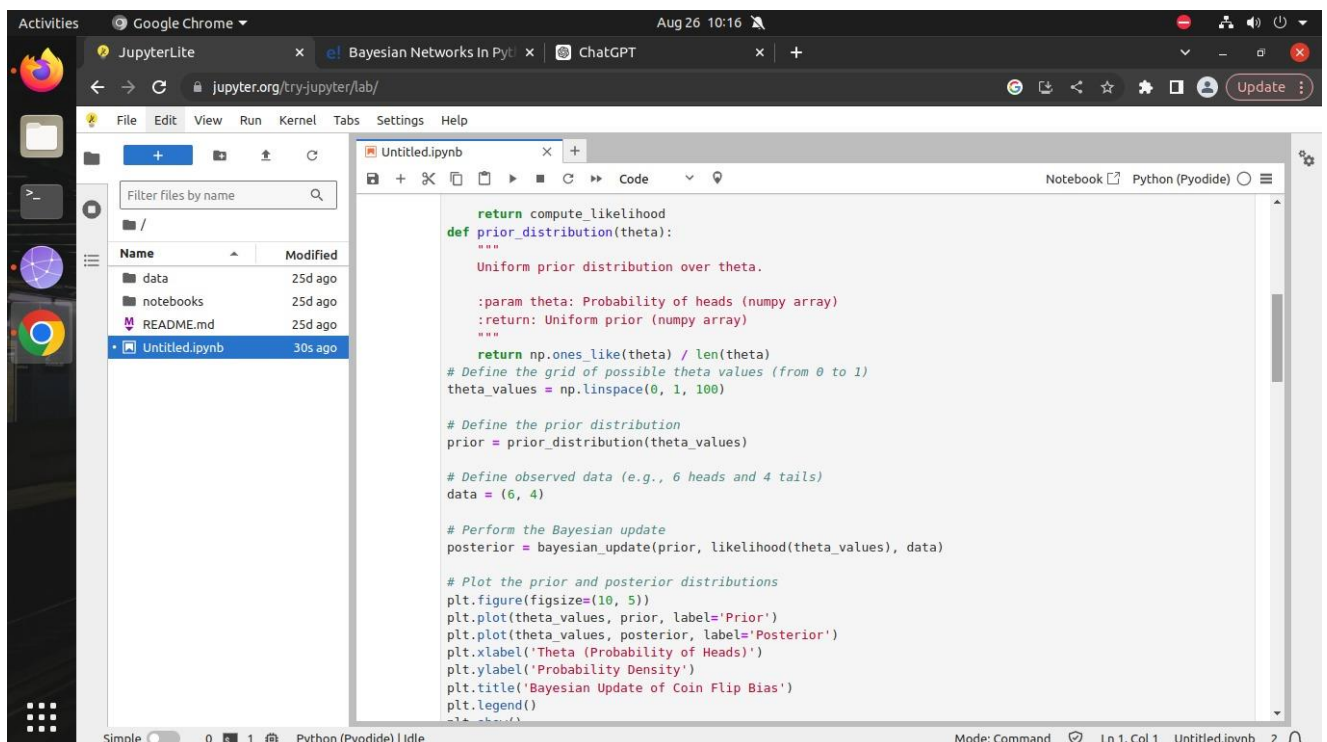
```
[1]: import numpy as np
import matplotlib.pyplot as plt
def bayesian_update(prior, likelihood, data):
    """
    Updates the prior belief given the likelihood of the data.

    :param prior: Prior probability distribution (numpy array)
    :param likelihood: Likelihood function (function)
    :param data: Observed data (e.g., number of heads and tails)
    :return: Posterior probability distribution (numpy array)
    """
    # Calculate the unnormalized posterior by element-wise multiplication
    unnormalized_posterior = likelihood(data) * prior

    # Normalize the posterior
    posterior = unnormalized_posterior / np.sum(unnormalized_posterior)

    return posterior
def likelihood(theta):
    """
    Likelihood function for a coin flip.

    :param theta: Probability of heads (numpy array)
    :return: Likelihood of the data given theta (function)
    """
    def compute_likelihood(data):
        heads, tails = data
        return theta**heads * (1 - theta)**tails
```



The screenshot shows the JupyterLite interface with the same file browser and code editor. The code editor shows the continuation of the Python code:

```
return compute_likelihood
def prior_distribution(theta):
    """
    Uniform prior distribution over theta.

    :param theta: Probability of heads (numpy array)
    :return: Uniform prior (numpy array)
    """
    return np.ones_like(theta) / len(theta)
# Define the grid of possible theta values (from 0 to 1)
theta_values = np.linspace(0, 1, 100)

# Define the prior distribution
prior = prior_distribution(theta_values)

# Define observed data (e.g., 6 heads and 4 tails)
data = (6, 4)

# Perform the Bayesian update
posterior = bayesian_update(prior, likelihood(theta_values), data)

# Plot the prior and posterior distributions
plt.figure(figsize=(10, 5))
plt.plot(theta_values, prior, label='Prior')
plt.plot(theta_values, posterior, label='Posterior')
plt.xlabel('Theta (Probability of Heads)')
plt.ylabel('Probability Density')
plt.title('Bayesian Update of Coin Flip Bias')
plt.legend()
```

Activities Google Chrome Aug 26 10:16

JupyterLite Bayesian Networks In PyL ChatGPT

jupyter.org/try-jupyter/lab/

File Edit View Run Kernel Tabs Settings Help

Filter files by name

| Name           | Modified |
|----------------|----------|
| /              |          |
| data           | 25d ago  |
| notebooks      | 25d ago  |
| README.md      | 25d ago  |
| Untitled.ipynb | 40s ago  |

```
plt.figure(figsize=(10, 5))
plt.plot(theta_values, prior, label='Prior')
plt.plot(theta_values, posterior, label='Posterior')
plt.xlabel('Theta (Probability of Heads)')
plt.ylabel('Probability Density')
plt.title('Bayesian Update of Coin Flip Bias')
plt.legend()
plt.show()

# Define the grid of possible theta values (from 0 to 1)
theta_values = np.linspace(0, 1, 100)

# Define the prior distribution
prior = prior_distribution(theta_values)

# Define observed data (e.g., 6 heads and 4 tails)
data = (6, 4)

# Perform the Bayesian update
posterior = bayesian_update(prior, likelihood(theta_values), data)

# Plot the prior and posterior distributions
plt.figure(figsize=(10, 5))
plt.plot(theta_values, prior, label='Prior')
plt.plot(theta_values, posterior, label='Posterior')
plt.xlabel('Theta (Probability of Heads)')
plt.ylabel('Probability Density')
plt.title('Bayesian Update of Coin Flip Bias')
plt.legend()
plt.show()
```

Simple 0 1 Python (Pyodide) | Idle Mode: Command Ln 1, Col 1 Untitled.ipynb 2

