Experiment 3.2

Student Name: UTKARSH JOSHI **UID: 21BCS9158**
Branch: BE CSE **Section/Group:802A**
Semester: 5 SEM **DateofPerformance: 08/11/2023**
Subject Name: AI &ML with Lab **Subject Code: 21CSH-316**

1. **Aim:** Implement Naïve Bayes theorem to classify the English text

2. **Objective:** The objective is to assess how well the Naïve Bayes theorem to classify the English text and analyze its effectiveness in comparison to the algorithms or approaches.

3. **Input/Apparatus Used:** Google collab and python libraries.

4. **Hardwire Requirements:** Computer/Laptop minimum 4GB,windows and Power Supply

5. **Code:**

Task 1:

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
X, y = load_iris(return_X_y=True)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5,
random_state=0)
gnb = GaussianNB()
y_pred = gnb.fit(X_train, y_train).predict(X_test)
print("Number of mislabeled points out of a total %d points : %d"
      % (X_test.shape[0], (y_test != y_pred).sum()))
```

Task 2:

```
dataset = [("Buy cheap luxury watches", "spam"), ("Win a free iPhone X", "spam"),
 ("Get the best discounts", "spam"), ("Hello, how are you?", "not spam"),
  ("Meeting tomorrow?", "not spam"), ("Meeting today!", "spam"), ("Meeting
tomorrow?", "not spam"),
   ("buy healthy fruits", "not spam"), ("buy black watch", "spam"), ("buy belt", "not
spam")]
# Separate the dataset by class
class_data = {}  # Dictionary to hold separate classes
for text, label in dataset:
    if label not in class_data:
        class_data[label] = []
    class_data[label].append(text)
# Now, class_data will contain separate lists of text samples for each class
print("Spam class:")
for sample in class_data["spam"]:
    print(sample)
print("\nNot spam class:")
for sample in class_data["not spam"]:
    print(sample)
```

Task 3:

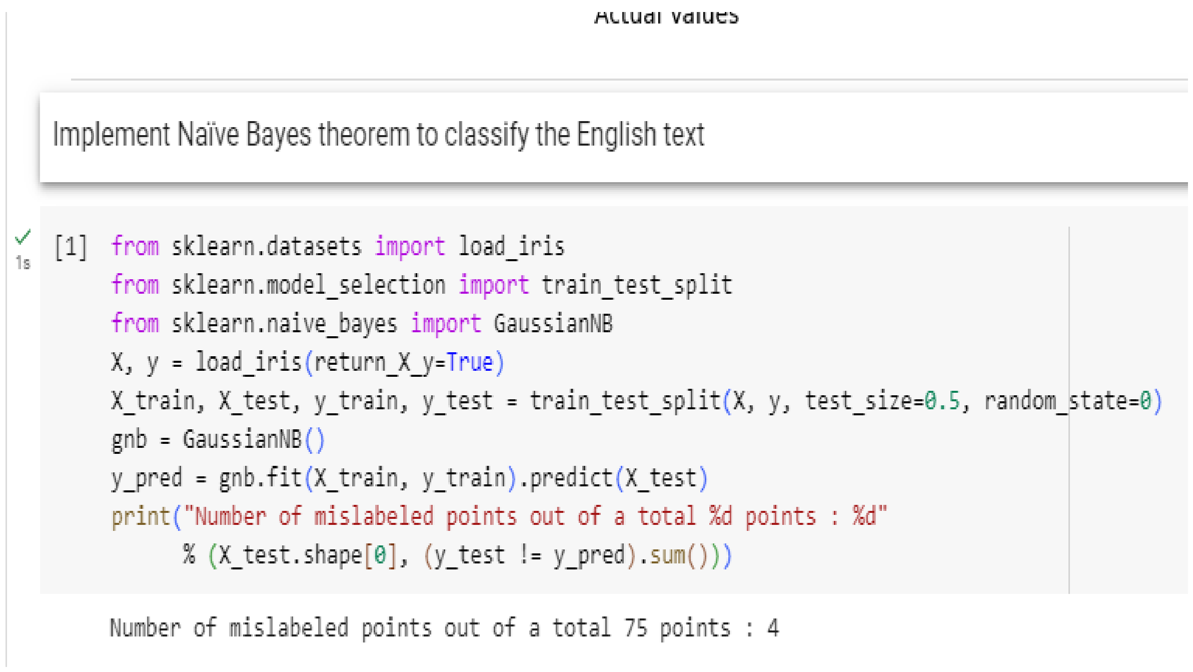
```
dataset = [
    ("Buy cheap luxury watches", "spam"), ("win a free iPhone X", "spam"),
    ("Get the best discounts", "spam"), ("Hello, how are you?", "not spam"),
    ("Meeting tomorrow?", "not spam"), ("Meeting today!", "spam"),
    ("Meeting tomorrow?", "not spam"), ("buy healthy fruits", "not spam"),
    ("buy black watch", "spam"), ("buy belt", "not spam")
]
# Calculate the number of samples
num_samples = len(dataset)
# Calculate the distribution of classes
class_distribution = {}
for text, label in dataset:
    if label not in class_distribution:
        class_distribution[label] = 0
    class_distribution[label] += 1
# Display dataset summary
print("Dataset Summary:")
print(f"Total samples: {num_samples}")
print("Class Distribution:")
for label, count in class_distribution.items():
    print(f"{label}: {count} samples")
```

Task 4:

```
import scipy.stats as stats
import matplotlib.pyplot as plt
import numpy as np
# Define mean and standard deviation
mu = 0
sigma = 5
# Create a range of x values
x = np.linspace(mu - 6 * sigma, mu + 6 * sigma, 250)
# Compute the Gaussian PDF
pdf = stats.norm.pdf(x, loc=mu, scale=sigma)
# Plot the PDF
plt.plot(x, pdf, label='Gaussian PDF', color="green")
plt.xlabel('x')
plt.ylabel('Probability Density')
plt.legend()
plt.title(f'Gaussian PDF (μ={mu}, σ={sigma})')
plt.grid(True)
plt.show()
```

6. Output:

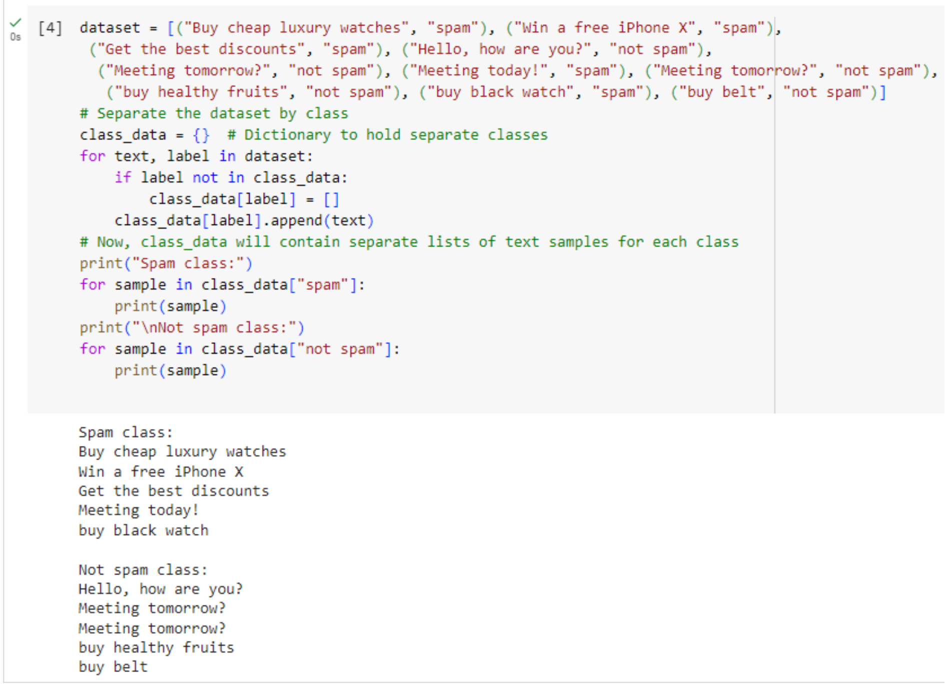
Task: 1


Implement Naïve Bayes theorem to classify the English text

```
[1] from sklearn.datasets import load_iris
    from sklearn.model_selection import train_test_split
    from sklearn.naive_bayes import GaussianNB
    X, y = load_iris(return_X_y=True)
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5, random_state=0)
    gnb = GaussianNB()
    y_pred = gnb.fit(X_train, y_train).predict(X_test)
    print("Number of mislabeled points out of a total %d points : %d"
          % (X_test.shape[0], (y_test != y_pred).sum()))
```

```
Number of mislabeled points out of a total 75 points : 4
```

Task: 2

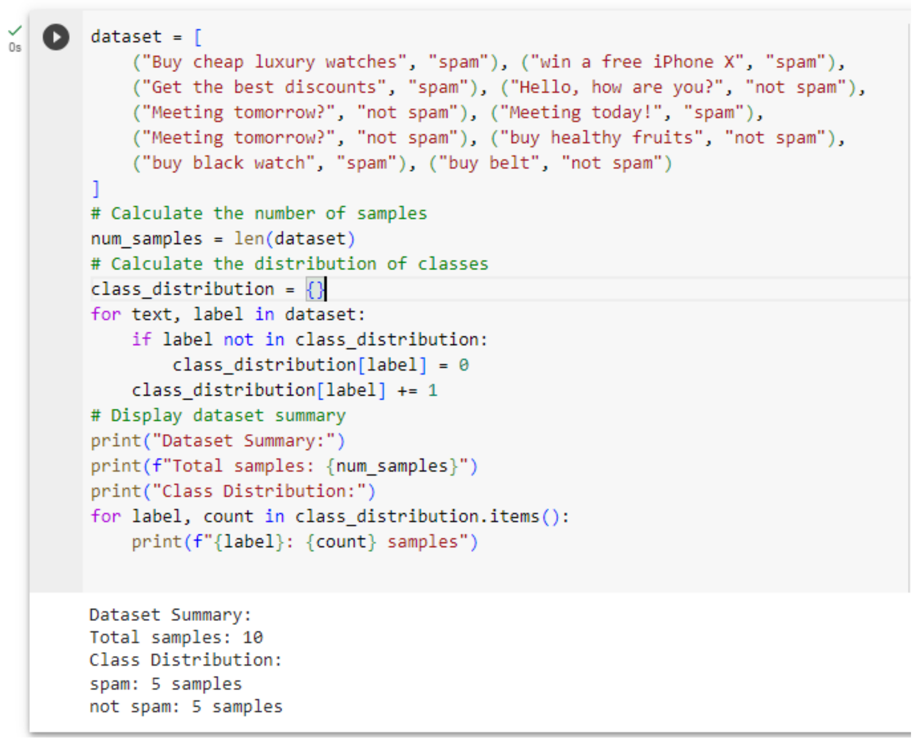


```
[4] dataset = [("Buy cheap luxury watches", "spam"), ("Win a free iPhone X", "spam"),
     ("Get the best discounts", "spam"), ("Hello, how are you?", "not spam"),
      ("Meeting tomorrow?", "not spam"), ("Meeting today!", "spam"), ("Meeting tomorrow?", "not spam"),
       ("buy healthy fruits", "not spam"), ("buy black watch", "spam"), ("buy belt", "not spam")]
    # Separate the dataset by class
    class_data = {}  # Dictionary to hold separate classes
    for text, label in dataset:
        if label not in class_data:
            class_data[label] = []
        class_data[label].append(text)
    # Now, class_data will contain separate lists of text samples for each class
    print("Spam class:")
    for sample in class_data["spam"]:
        print(sample)
    print("\nNot spam class:")
    for sample in class_data["not spam"]:
        print(sample)
```

```
Spam class:
Buy cheap luxury watches
Win a free iPhone X
Get the best discounts
Meeting today!
buy black watch

Not spam class:
Hello, how are you?
Meeting tomorrow?
Meeting tomorrow?
buy healthy fruits
buy belt
```

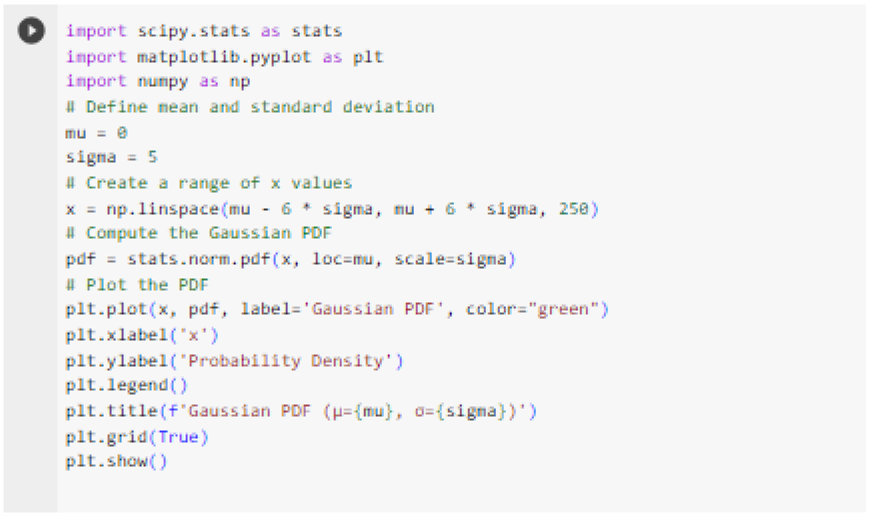
Task: 3



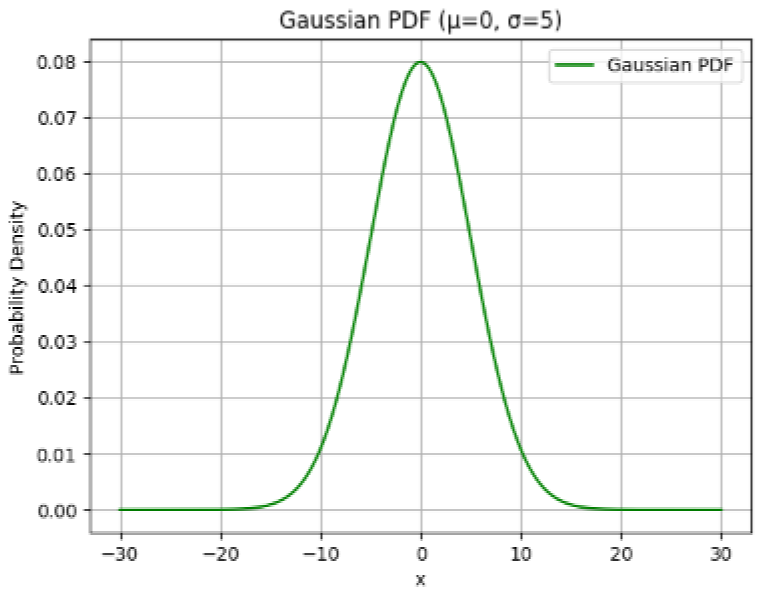
```
dataset = [
    ("Buy cheap luxury watches", "spam"), ("Win a free iPhone X", "spam"),
    ("Get the best discounts", "spam"), ("Hello, how are you?", "not spam"),
    ("Meeting tomorrow?", "not spam"), ("Meeting today!", "spam"),
    ("Meeting tomorrow?", "not spam"), ("buy healthy fruits", "not spam"),
    ("buy black watch", "spam"), ("buy belt", "not spam")
]
# Calculate the number of samples
num_samples = len(dataset)
# Calculate the distribution of classes
class_distribution = {}
for text, label in dataset:
    if label not in class_distribution:
        class_distribution[label] = 0
    class_distribution[label] += 1
# Display dataset summary
print("Dataset Summary:")
print(f"Total samples: {num_samples}")
print("Class Distribution:")
for label, count in class_distribution.items():
    print(f"{label}: {count} samples")
```

```
Dataset Summary:
Total samples: 10
Class Distribution:
spam: 5 samples
not spam: 5 samples
```

Task: 4



```
import scipy.stats as stats
import matplotlib.pyplot as plt
import numpy as np
# Define mean and standard deviation
mu = 0
sigma = 5
# Create a range of x values
x = np.linspace(mu - 6 * sigma, mu + 6 * sigma, 250)
# Compute the Gaussian PDF
pdf = stats.norm.pdf(x, loc=mu, scale=sigma)
# Plot the PDF
plt.plot(x, pdf, label='Gaussian PDF', color="green")
plt.xlabel('x')
plt.ylabel('Probability Density')
plt.legend()
plt.title(f'Gaussian PDF (μ={mu}, σ={sigma})')
plt.grid(True)
plt.show()
```



7. Learning Outcomes:

When implementing Naive Bayes theorem to classify English text
The learning outcomes should encompass both theoretical and practical aspects.
Here are some specific learning outcomes for this task:

- Understanding of Naive Bayes Classification
- Preprocessing and Text Tokenization.
- Building a Classification Model.
- Feature Extraction and Vectorization.
- Model Selection and Hyper parameter Tuning