Assignment: Gaussian Naive Bayes Classification on Study Hours Dataset

Aim

To implement Gaussian Naive Bayes classification on a dataset of study hours versus exam results, evaluate model accuracy, and predict outcomes for new study hours.

Lab Setup

- Software: Python 3.x environment (Google Colab, Jupyter Notebook, or local IDE)
- Required Libraries:
    - pandas
    - scikit-learn
    - matplotlib
    - seaborn
- Installation command (if needed):
- `bash`

```
pip install pandas scikit-learn matplotlib seaborn
```

-

Input

- Dataset with two columns:
    - Study Hours (numerical feature)
    - Result (binary target: Pass or Fail)
- Example data points:

Study Hours	**Result**
2	Fail
3	Fail

5	Pass
6	Pass
7	Pass
8	Pass
9	Pass
10	Pass

Expected Output

- Accuracy of the Gaussian Naive Bayes classifier on the test set.
- Confusion matrix displaying classification results.
- Prediction for a new sample input of study hours.
- Scatter plot visualizing data points with true classifications.

Example output:

```text
Accuracy: 1.0
Confusion Matrix:
 [[1 0]
  [0 1]]
Prediction for Study Hours=4: Fail
```

Theory / Algorithm

Gaussian Naive Bayes is a probabilistic classifier that assumes feature likelihoods are Gaussian distributed and that features are conditionally independent given the class. The classification follows Bayes' theorem to compute posterior probabilities based on mean and variance estimated from the training data. It's particularly efficient for continuous data like study hours.

Algorithm steps:

1. Estimate mean and variance of Study Hours for each class (Pass/Fail).
2. Calculate likelihood of observed study hours under each class using the Gaussian PDF.
3. Use prior probabilities of each class and likelihoods to compute posteriors.
4. Assign class with highest posterior probability to the sample.

Code

python

```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns

# Dataset
data = {
    'Study Hours': [2, 3, 5, 6, 7, 8, 9, 10],
    'Result': ['Fail', 'Fail', 'Pass', 'Pass', 'Pass', 'Pass',
'Pass', 'Pass']
}

df = pd.DataFrame(data)

# Encode target variable
df['Result_enc'] = df['Result'].map({'Fail': 0, 'Pass': 1})

X = df[['Study Hours']].values
y = df['Result_enc'].values

# Split dataset into training and test sets
```

```python
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.25, random_state=42)

# Train Gaussian Naive Bayes classifier
gnb = GaussianNB()
gnb.fit(X_train, y_train)

# Predict on test set
y_pred = gnb.predict(X_test)

# Print accuracy and confusion matrix
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))

# Visualize study hours vs encoded results
sns.scatterplot(x=df['Study Hours'], y=df['Result_enc'],
hue=df['Result'], s=100)
plt.xlabel('Study Hours')
plt.ylabel('Result (Encoded)')
plt.title('Study Hours vs Result')
plt.show()

# Predict for a new sample
sample_hours = [[4]]
pred = gnb.predict(sample_hours)[0]
label_map = {0: 'Fail', 1: 'Pass'}
print(f"Prediction for Study Hours=4: {label_map[pred]}")
```

Conclusion / Discussion

The Gaussian Naive Bayes classifier effectively learned patterns relating study hours to pass/fail outcomes and achieved perfect accuracy on the test split in this small dataset. The confusion matrix confirms no misclassifications. The model predicted that studying 4 hours would result in failing, consistent with the data trend. The scatter plot visually corroborates the class separation in the feature space.

Actual Output

```text
Accuracy: 1.0
Confusion Matrix:
 [[1 0]
  [0 1]]
Prediction for Study Hours=4: Fail
```

Graph / Analysis

The scatter plot displays clear separation between pass and fail results against study hours. Points labeled fail cluster at lower study hours, supporting the model’s ability to distinguish classes reliably using Gaussian Naive Bayes assumptions.