Project 1

Naive Bayes Classifier

With python

Download Dataset

You can download the dataset from our course repository (under workshop I):

https://github.com/Kanchon-Gharami/ML_with_AI_Bangladesh/tree/main/Workshop%20II

Coding

After opening a new notebook, upload the **golf_data.csv** on your colab computer. Then run below.

Step 1: Import Libraries

First, you'll need to import the necessary libraries in your Colab notebook:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import confusion_matrix, accuracy_score,
classification_report
```

Step 2: Load the dataset

Load the dataset using pandas library:

```
data = pd.read_csv('golf_data.csv')
```

Step 3: Preprocess the dataset:

- Encode the categorical features ('Outlook', 'Temperature', 'Humidity', 'Windy') into numerical values since Gaussian Naive Bayes assumes that the features follow a normal distribution.
- Encode the label 'Play Golf' into numerical values.

```python
# Encode categorical features and label
outlook_encoder = LabelEncoder()
temperature_encoder = LabelEncoder()
humidity_encoder = LabelEncoder()
windy_encoder = LabelEncoder()
play_golf_encoder = LabelEncoder()

data['Outlook'] = outlook_encoder.fit_transform(data['Outlook'])
data['Temperature'] =
temperature_encoder.fit_transform(data['Temperature'])
data['Humidity'] = humidity_encoder.fit_transform(data['Humidity'])
data['Windy'] = windy_encoder.fit_transform(data['Windy'].astype(str))
data['Play Golf'] = play_golf_encoder.fit_transform(data['Play Golf'])
```

Step 4: Labels-Feature Split

Split the dataset into features (X) and the label (y).:

```python
# Split the dataset into features and labels
X = data[['Outlook', 'Temperature', 'Humidity', 'Windy']]
y = data['Play Golf']

# using the train test split function
X_train, X_test, y_train, y_test = train_test_split(X,y , random_state=104,
test_size=0.25, shuffle=True)
```

Step 5: Import Libraries

Train the Gaussian Naive Bayes classifier on the training dataset:

```python
# Train the Gaussian Naive Bayes classifier
model = GaussianNB()
model.fit(X_train, y_train)
```

Step 6: Testing

Test for a sample datat:

```python
# New input data
new_data = pd.DataFrame([['Sunny', 'Hot', 'Normal', False]],
                        columns=['Outlook', 'Temperature', 'Humidity',
'Windy'])

# Encode the input data
new_data['Outlook'] = outlook_encoder.transform(new_data['Outlook'])
new_data['Temperature'] =
temperature_encoder.transform(new_data['Temperature'])
new_data['Humidity'] = humidity_encoder.transform(new_data['Humidity'])
new_data['Windy'] = windy_encoder.transform(new_data['Windy'].astype(str))

# Predict the probability
probability = model.predict_proba(new_data)
print("The probability of playing golf under the given conditions is:",
probability[0][1])
```

Step 7: Evaluation

Train the Gaussian Naive Bayes classifier on the training dataset:

```python
# Predict on the test set
y_pred = model.predict(X_test)

# Calculate the confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred)
```

```python
# Calculate the accuracy
accuracy = accuracy_score(y_test, y_pred)

# Get precision, recall, f1-score, and support
class_report = classification_report(y_test, y_pred)

# Print the results
print("Confusion Matrix:")
print(conf_matrix)

print("\nAccuracy: {:.2f}%".format(accuracy * 100))

print("\nClassification Report:")
print(class_report)
```