Kingdom of Saudi Arabia
Ministry of Education
University of Jeddah
College of Computer Science and Engineering
Department of Computer Science and Artificial Intelligence

المملكة العربية السعودية
وزارة التعليم
جامعة جدّة
كلية علوم وهندسة الحاسب
قسم علوم الحاسب والذكاء الاصطناعي

Lab 5
CCAI 312 Pattern Recognition
Third Trimester 2023

Student Name: Ruba Khalid Alsulami

Student ID: 2110618

		Max Score	Student Score
PLO S2 / CLO 2 / SO 2	Task 1	2	
PLO C4 / CLO 3 / SO 7	Task 2	2	
Total			

Task 1: [PLO S2 / CLO 2 / SO 2] [2 marks]

1. Import the necessary libraries: `sklearn.datasets`, `sklearn.feature_extraction.text`, and `sklearn.naive_bayes`.

   from sklearn.datasets import fetch_20newsgroups

   from sklearn.feature_extraction.text import CountVectorizer

   from sklearn.naive_bayes import MultinomialNB

2. Load the "20 Newsgroups" dataset using the `fetch_20newsgroups` function and specify the categories to be used for training and testing.

```
In [54]:
from sklearn.datasets import fetch_20newsgroups
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB

categories = ['alt.atheism', 'talk.religion.misc', 'comp.graphics', 'sci.space','talk.politics.misc','sci.med'
              ,'rec.autos', 'rec.motorcycles']

train_data = fetch_20newsgroups(subset='train', categories=categories, shuffle=True, random_state=42)
test_data = fetch_20newsgroups(subset='test', categories=categories, shuffle=True, random_state=42)

In [55]:
print(train_data.filenames.shape)
print(test_data.filenames.shape)

(4285,)
(2853,)
```

3. Convert the text data into numerical features using the `CountVectorizer` class from Scikit-Learn.

```
In [14]:
vector=CountVectorizer()
train_vectors = vector.fit_transform(train_data.data)
test_vectors = vector.transform(test_data.data)
y_train = train_data.target
y_test = test_data.target
```

4. Train the Naive Bayes classifier on the training data using the `MultinomialNB` class from Scikit-Learn.

```
In [24]:
model = MultinomialNB()
model.fit(train_vectors, y_train)
```

5. Calculate the training accuracy

```
In [58]:
from sklearn.metrics import accuracy_score
predicted_labels = model.predict(train_vectors)
acc= accuracy_score(y_train, predicted_labels)
print('Accuracy = ',acc)

Accuracy =  0.9885647607934656
```

6. Evaluate the performance of the classifier on the test data

```
In [59]:
predictions = model.predict(test_vectors)
accuracy = accuracy_score(y_test, predictions)
```

7. Print the training and test accuracies of the classifier.

```
In [64]:
print("Train Accuracy: ",acc)

Y_pred_test = model.predict(test_vectors)
test_Accuracy=accuracy_score(y_test, Y_pred_test)
print("Test Accuracy: "+str(test_Accuracy))

Train Accuracy:  0.9885647607934656
Test Accuracy: 0.886435331230284
```

Task? 2: [PLO C4 / CLO 3 / SO 7]
[2 mark]

1. use GridSearchCV to perform hyperparameter tuning for the Naive Bayes model. Define a range of values for the alpha hyperparameter, which controls the smoothing of the probability estimates.

```
In [30]: print("Task 2")

         Task 2

In [44]: from sklearn.model_selection import GridSearchCV

         param_grid = {'alpha': [0.1, 1, 10]}
         grid = GridSearchCV(MultinomialNB(), param_grid, cv=5)
         grid.fit(train_vectors, train_data.target)

         print('Best hyperparameters :', grid.best_params_)
         print('Accuracy validation score:', grid.best_score_)

         Best hyperparameters : {'alpha': 0.1}
         Accuracy validation score: 0.9596857941685528
```

2. Evaluate the model with the best hyperparameters on the testing set and report the testing accuracy.

```
In [65]: best_clf = grid.best_estimator_
         y_test_pred = best_clf.predict(test_vectors)
         test= accuracy_score(y_test, y_test_pred)
         print("Best hyperparameters with Testing accuracy  =",test)

         Best hyperparameters with Testing accuracy  = 0.8878373641780581
```

3. Does the model improve with hyperparameters tuning?

Yes,the model was improve with hyperparameters tuning with Accuracy: 0.9606685951513537