```
Document Classifier
Design a document classifier using Naive Bayes Classifier
```

In [1]:
```
#Step 1: Import necessary librabries
import pandas as pd
documents=pd.read_csv("C:/1562/AlMlDatasets/final.csv")
```

In [2]:
```
documents.head()
```

Out[2]:

	Document	Class
0	MET is a great educational institution.	education
1	Educational greatness depends on ethics	education
2	A story of great ethics and educational greatness	education
3	Black is a great cinema	cinema
4	good movie depends on good story	cinema

In [4]:
```
#Step 2: Preprocess the data
#Encode class_label as 1 & 0
documents['Class']=documents.Class.map({'education':1,'cinema':0})
```

In [5]:
```
documents.head()
```

Out[5]:

	Document	Class
0	MET is a great educational institution.	1
1	Educational greatness depends on ethics	1
2	A story of great ethics and educational greatness	1
3	Black is a great cinema	0
4	good movie depends on good story	0

In [7]:
```
# prepare features and target
#convert the dataframe into an array
train_docs=documents.values
train_docs
```

Out[7]:
```
array([['MET is a great educational institution.', 1],
       ['Educational greatness depends on ethics', 1],
       ['A story of great ethics and educational greatness', 1],
       ['Black is a great cinema', 0],
       ['good movie depends on good story', 0]], dtype=object)
```

In [9]:
```
train_x=train_docs[:,0]
train_y=train_docs[:,1]
```

In [10]:
```
train_x
```

Out[10]:
```
array(['MET is a great educational institution.',
       'Educational greatness depends on ethics',
       'A story of great ethics and educational greatness',
       'Black is a great cinema', 'good movie depends on good story'],
      dtype=object)
```

In [11]:
```
train_y
```

Out[11]:
```
array([1, 1, 1, 0, 0], dtype=object)
```

In [12]:
```
# convert y into integer
train_y=train_y.astype('int')
```

In [13]:
```
train_y
```

Out[13]:
```
array([1, 1, 1, 0, 0])
```

In [15]:
```
# Step 3: prepare bag of words
#identify unique features from the document
from sklearn.feature_extraction.text import CountVectorizer
#CountVectorizer is class that allows you to find unique words from the given
#create a object
vec=CountVectorizer()
#apply this on train_x
vec.fit(train_x)
vec.vocabulary_
```

Out[15]:
```
{'met': 11,
 'is': 10,
 'great': 7,
 'educational': 4,
 'institution': 9,
 'greatness': 8,
 'depends': 3,
 'on': 14,
 'ethics': 5,
 'story': 15,
 'of': 13,
 'and': 0,
 'black': 1,
 'cinema': 2,
 'good': 6,
 'movie': 12}
```

In [25]:
```
# remove the stop words
vec=CountVectorizer(stop_words='english')
vec.fit(train_x)
vec.get_feature_names()
```

Out[25]:
```
['black',
 'cinema',
 'depends',
 'educational',
 'ethics',
 'good',
 'great',
 'greatness',
 'institution',
 'met',
 'movie',
 'story']
```

In [26]:
```
#prepare a sparse matrix
trans_x=vec.transform(train_x)
trans_x
```

Out[26]:
```
<5x12 sparse matrix of type '<class 'numpy.int64'>'
        with 20 stored elements in Compressed Sparse Row format>
```

In [27]:
```
print(trans_x)
```

```
  (0, 3)        1
  (0, 6)        1
  (0, 8)        1
  (0, 9)        1
  (1, 2)        1
  (1, 3)        1
  (1, 4)        1
  (1, 7)        1
  (2, 3)        1
  (2, 4)        1
  (2, 6)        1
  (2, 7)        1
  (2, 11)       1
  (3, 0)        1
  (3, 1)        1
  (3, 6)        1
  (4, 2)        1
  (4, 5)        2
  (4, 10)       1
  (4, 11)       1
```

In [29]:
```
trans_x.toarray()
```

Out[29]:
```
array([[0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0],
       [0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0],
       [0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1],
       [1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0],
       [0, 0, 1, 0, 0, 2, 0, 0, 0, 0, 1, 1]], dtype=int64)
```

In [31]:
```
# show bag of words
pd.DataFrame(trans_x.toarray(),columns=vec.get_feature_names())
```

Out[31]:

	black	cinema	depends	educational	ethics	good	great	greatness	institution	met	movie
0	0	0	0	1	0	0	1	0	1	1	0
1	0	0	1	1	1	0	0	1	0	0	0
2	0	0	0	1	1	0	1	1	0	0	0
3	1	1	0	0	0	0	1	0	0	0	0
4	0	0	1	0	0	2	0	0	0	0	1

In [34]:
```
#step4 : Prepare test data set
#load test data set
test_docs=pd.read_csv("C:/1562/AlMlDatasets/test.csv")
test_docs.head()
```

Out[34]:

	Document	Class
0	great great story	education

In [37]:
```
#encode test dataset
test_docs['Class']=test_docs.Class.map({'education':1,'cinema':0})
```

In [41]:
```
#convert array
test_arr=test_docs.values
test_x=test_arr[:,0]
test_y=test_arr[:,1]
```

In [42]:
```
#transform
test_trans_x=vec.transform(test_x)
```

In [43]:
```
#step 5: Build the model
from sklearn.naive_bayes import MultinomialNB
#create object
mnb=MultinomialNB()
# train the model
mnb.fit(trans_x,train_y)
```

Out[43]:
```
MultinomialNB()
```

In [45]:
```
#Step 6: test the model
prob=mnb.predict_proba(test_trans_x)
prob
```

Out[45]:
```
array([[0.36656891, 0.63343109]])
```

In [46]:
```
print("Document belogs to :",mnb.predict(test_trans_x))
```

```
Document belogs to : [1]
```

In [ ]: