Lab 06

Text classification: Naive Bayes, Rocchio and k-nearest neighbor

Group 1

Text classification: Naive Bayes, Rocchio and k-NN

Group 1

```
[153]: # Problem : Comparison between Naive Bayes, Rochhio Classifier and K-NN␣
        ↪algorithm on 20 Newsgroup dataset
       #
```

```
[6]: from sklearn.metrics import classification_report,f1_score
     from sklearn.datasets import fetch_20newsgroups
     from util import stop_words
```

```
[7]: twenty_train = fetch_20newsgroups(subset='train'␣
      ↪,shuffle=True,download_if_missing=True)
```

```
[8]: print(f"Total documents : {len(twenty_train.data)}")
     print(f"Number of classes : {len(twenty_train.target_names)}")
     twenty_train.target_names
```

```
Total documents : 11314
Number of classes : 20
```

```
[8]: ['alt.atheism',
      'comp.graphics',
      'comp.os.ms-windows.misc',
      'comp.sys.ibm.pc.hardware',
      'comp.sys.mac.hardware',
      'comp.windows.x',
      'misc.forsale',
      'rec.autos',
      'rec.motorcycles',
      'rec.sport.baseball',
      'rec.sport.hockey',
      'sci.crypt',
      'sci.electronics',
      'sci.med',
      'sci.space',
      'soc.religion.christian',
      'talk.politics.guns',
      'talk.politics.mideast',
      'talk.politics.misc',
 'talk.religion.misc']
```

[9]:
```
## WordCount

from sklearn.feature_extraction.text import CountVectorizer
# WordCount with filtering Stop Words
count_vect = CountVectorizer(stop_words=stop_words)

X_train_counts = count_vect.fit_transform(twenty_train.data)
```

[10]:
```
print(f"Number of Feature words (without stemming and a bit more cleaning) :␣
 ↪{X_train_counts.shape[1]}")
```

```
Number of Feature words (without stemming and a bit more cleaning) : 129156
```

[11]:
```
## DOCUMNETS AS tf-idf Feature Vectors

from sklearn.feature_extraction.text import TfidfTransformer
tfidf_transformer = TfidfTransformer()

# tf-idf TD Matrix
X_train_tfidf = tfidf_transformer.fit_transform(X_train_counts)
```

[12]:
```
# Importing Classifiers

from sklearn.naive_bayes import MultinomialNB
from sklearn.neighbors import KNeighborsClassifier,NearestCentroid
```

[13]:
```
# Classification Pipeline Module

from sklearn.pipeline import Pipeline
```

[14]:
```
# Pipeline =>
# WordCount ===> tf-idf vectorizaton ===> Classification

# pipeline for Multinomial Naive Bayes
text_clf = Pipeline([('vect', CountVectorizer(stop_words=stop_words)),('tfidf',␣
 ↪TfidfTransformer()),('clf', MultinomialNB())])
```

[15]:
```
# Training Naive Bayes
text_clf = text_clf.fit(twenty_train.data, twenty_train.target)
```

[16]:
```
## Evaluating model on test-set



import numpy as np
# test set
```

```
twenty_test = fetch_20newsgroups(subset='test', shuffle=True)

predicted = text_clf.predict(twenty_test.data)
# mean accuracy
acc =np.mean(predicted == twenty_test.target)
print(f"Acc ~ {round(acc*100)}%")
```

```
Acc ~ 82.0%
```

[17]:
```
print("Classification Report : ")
print(classification_report(twenty_test.
 ↪target,predicted,target_names=twenty_train.target_names))
```

```
Classification Report :
                          precision    recall  f1-score   support

             alt.atheism       0.81      0.70      0.75       319
           comp.graphics       0.78      0.72      0.75       389
 comp.os.ms-windows.misc       0.79      0.70      0.74       394
comp.sys.ibm.pc.hardware       0.67      0.81      0.73       392
   comp.sys.mac.hardware       0.86      0.81      0.84       385
          comp.windows.x       0.86      0.79      0.83       395
            misc.forsale       0.85      0.80      0.82       390
               rec.autos       0.89      0.91      0.90       396
         rec.motorcycles       0.93      0.95      0.94       398
      rec.sport.baseball       0.91      0.92      0.92       397
        rec.sport.hockey       0.88      0.98      0.93       399
               sci.crypt       0.77      0.96      0.85       396
         sci.electronics       0.84      0.63      0.72       393
                 sci.med       0.92      0.78      0.84       396
               sci.space       0.81      0.95      0.88       394
  soc.religion.christian       0.64      0.95      0.77       398
      talk.politics.guns       0.68      0.95      0.79       364
   talk.politics.mideast       0.94      0.95      0.95       376
      talk.politics.misc       0.94      0.53      0.68       310
      talk.religion.misc       0.93      0.27      0.42       251

                accuracy                           0.82      7532
               macro avg       0.84      0.80      0.80      7532
            weighted avg       0.83      0.82      0.81      7532
```

[18]:
```
# Pipeline for Rocchio Classifier
rocchio_clf = Pipeline([('vect',␣
 ↪CountVectorizer(stop_words=stop_words)),('tfidf', TfidfTransformer()),('clf',␣
 ↪NearestCentroid())])
```

[19]:
```python
# Training Rocchio Classifier
rocchio_clf = rocchio_clf.fit(twenty_train.data, twenty_train.target)
```

[20]:
```python
# Evaluating Rocchio classifier on test set
predicted = rocchio_clf.predict(twenty_test.data)
# mean accuracy
acc =np.mean(predicted == twenty_test.target)
print(f"Acc ~ {round(acc*100)}%")
```

```
Acc ~ 74.0%
```

[21]:
```python
print("Classification Report : ")
print(classification_report(twenty_test.
 ↪target,predicted,target_names=twenty_train.target_names))
```

```
Classification Report :
                          precision    recall  f1-score   support

             alt.atheism       0.79      0.54      0.64       319
           comp.graphics       0.56      0.79      0.65       389
 comp.os.ms-windows.misc       0.72      0.71      0.71       394
comp.sys.ibm.pc.hardware       0.69      0.63      0.66       392
   comp.sys.mac.hardware       0.77      0.74      0.75       385
          comp.windows.x       0.84      0.67      0.74       395
            misc.forsale       0.75      0.82      0.78       390
               rec.autos       0.88      0.81      0.85       396
         rec.motorcycles       0.97      0.90      0.93       398
      rec.sport.baseball       0.93      0.88      0.90       397
        rec.sport.hockey       0.95      0.88      0.91       399
               sci.crypt       0.97      0.71      0.82       396
         sci.electronics       0.33      0.80      0.47       393
                 sci.med       0.92      0.55      0.69       396
               sci.space       0.86      0.78      0.82       394
  soc.religion.christian       0.76      0.83      0.80       398
      talk.politics.guns       0.72      0.80      0.76       364
   talk.politics.mideast       0.98      0.70      0.82       376
      talk.politics.misc       0.63      0.60      0.61       310
      talk.religion.misc       0.59      0.47      0.53       251

                accuracy                           0.74      7532
               macro avg       0.78      0.73      0.74      7532
            weighted avg       0.79      0.74      0.75      7532
```

[28]:
```python
# Pipeline for KNN Classifier
knn_clf = Pipeline([('vect', CountVectorizer()),('tfidf',␣
 ↪TfidfTransformer()),('clf', KNeighborsClassifier(5,p=1))])
```

[29]:
```python
# Training KNN Classifier
knn_clf = rocchio_clf.fit(twenty_train.data, twenty_train.target)
```

[30]:
```python
# Evaluating KNN classifier on test set
predicted = knn_clf.predict(twenty_test.data)
# mean accuracy
acc =np.mean(predicted == twenty_test.target)
print(f"Acc ~ {round(acc*100)}%")
```

```
Acc ~ 74.0%
```

[31]:
```python
print("Classification Report : ")
print(classification_report(twenty_test.
 ↪target,predicted,target_names=twenty_train.target_names))
```

```
Classification Report :
                          precision    recall  f1-score   support

             alt.atheism       0.79      0.54      0.64       319
           comp.graphics       0.56      0.79      0.65       389
 comp.os.ms-windows.misc       0.72      0.71      0.71       394
comp.sys.ibm.pc.hardware       0.69      0.63      0.66       392
   comp.sys.mac.hardware       0.77      0.74      0.75       385
          comp.windows.x       0.84      0.67      0.74       395
            misc.forsale       0.75      0.82      0.78       390
               rec.autos       0.88      0.81      0.85       396
         rec.motorcycles       0.97      0.90      0.93       398
      rec.sport.baseball       0.93      0.88      0.90       397
        rec.sport.hockey       0.95      0.88      0.91       399
               sci.crypt       0.97      0.71      0.82       396
         sci.electronics       0.33      0.80      0.47       393
                 sci.med       0.92      0.55      0.69       396
               sci.space       0.86      0.78      0.82       394
  soc.religion.christian       0.76      0.83      0.80       398
      talk.politics.guns       0.72      0.80      0.76       364
   talk.politics.mideast       0.98      0.70      0.82       376
      talk.politics.misc       0.63      0.60      0.61       310
      talk.religion.misc       0.59      0.47      0.53       251

                accuracy                           0.74      7532
               macro avg       0.78      0.73      0.74      7532
            weighted avg       0.79      0.74      0.75      7532
```

1 Results

From above experiment we see that Naive Bayes(F-Score: 0.80) performs much better than Rocchio (F-Score: 0.74) and KNN (0.74) classifier. Rocchio and KNN have same performance.