Text Classification using various classifiers ¶

In [1]:

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report
import pandas as pd
```

In [2]:

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

In [3]:

```
xtrain,xtest,ytrain,ytest = train_test_split(data['text'],data['label'],test_size = 0.2,random_state = 42)
```

In [4]:

```
#Feature extraction using tf-idf
tfidf_vectorizer = TfidfVectorizer(max_features=10000)
x_train_tfidf = tfidf_vectorizer.fit_transform(xtrain)
x_test_tfidf = tfidf_vectorizer.transform(xtest)
```

In [13]:

```
#Initialise and train Naive Bayes
nb_classifier = MultinomialNB()
nb_classifier.fit(x_train_tfidf,ytrain)
```

Out[13]:

```
MultinomialNB()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [14]:

```
y_pred = nb_classifier.predict(x_test_tfidf)
```

In [15]:

```
accuracy = accuracy_score(ytest,y_pred)
print("Accuracy: ",accuracy)
```

```
Accuracy:  0.49
```

In [16]:

```
print(classification_report(ytest,y_pred))
```

```
              precision    recall  f1-score   support

    negative       0.46      0.55      0.50        94
    positive       0.52      0.43      0.47       106

    accuracy                           0.49       200
   macro avg       0.49      0.49      0.49       200
weighted avg       0.50      0.49      0.49       200
```

recall value is how well your model is able to recognise your arguments

Using logistic Regression, SVM & Random Forest

In [45]:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
```

In [34]:

```
def load_data(file):
    data = pd.read_csv(file)
    return data
```

In [35]:

```
def preprocess(data):
    return data
```

In [36]:

```
def extract_features(train_texts,test_texts):
    tfidf_vectorizer = TfidfVectorizer(max_features=1000)
    x_train_tfidf = tfidf_vectorizer.fit_transform(train_texts)
    x_test_tfidf = tfidf_vectorizer.transform(test_texts)
    return x_train_tfidf,x_test_tfidf,tfidf_vectorizer
```

In [37]:

```
def train_classifier(classifier,xtrain,ytrain):
    classifier.fit(xtrain,ytrain)
    return classifier
```

In [42]:

```
def evaluate_classifier(classifier,xtest,ytest):
    y_pred = classifier.predict(xtest)
    accuracy = accuracy_score(ytest,y_pred)
    report = classification_report(ytest,y_pred)
    confusion_mat = confusion_matrix(ytest,y_pred)
    return accuracy,report, confusion_mat
```

In [47]:

```
def main():
    file = 'dataset.csv'
    data = load_data(file)

    data['text']=data['text'].apply(preprocess)

    xtrain,xtest,ytrain,ytest = train_test_split(data['text'],data['label'],test_size=0.2,random_state=42)

    x_train_tfidf,x_test_tfidf,tfidf_vectorizer=extract_features(xtrain,xtest)

    classifiers = {
        "Multinomial Naive Bayes":MultinomialNB(),
        "Logistic Regression":LogisticRegression(),
        "Support Vector Machine":SVC(),
        "Random Forest": RandomForestClassifier()
    }

    results = {}
    for clf_name,clf in classifiers.items():
        print(f"Training {clf_name}...")
        clf = train_classifier(clf,x_train_tfidf,ytrain)
        print("Evaluating...")
        accuracy,report,confusion_mat = evaluate_classifier(clf,x_test_tfidf,ytest)
        results[clf_name] = {"accuracy":accuracy,"report":report,"confusion matrix":confusion_mat}

    for clf_name,result in results.items():
        print(f"\n{clf_name}:")
        print("Accuracy:",result["accuracy"])
        print("Classification Report")
        print(result["report"])
        print("Confusion Matrix")
        print(result["confusion matrix"])

        plt.figure(figsize=(8,6))
        sns.heatmap(result["confusion matrix"],annot=True,cmap="Reds",fmt="g",cbar=False)
        plt.xlabel("Predicted labels")
        plt.ylabel("True labels")
        plt.title(f"Confusion matrix for {clf_name}")
        plt.show()

if __name__ == "__main__":
    main()
```

```
Training Multinomial Naive Bayes...
Evaluating...
Training Logistic Regression...
Evaluating...
Training Support Vector Machine...
Evaluating...
Training Random Forest...
Evaluating...

Multinomial Naive Bayes:
Accuracy: 0.49
Classification Report
              precision    recall  f1-score   support

    negative       0.46      0.55      0.50        94
    positive       0.52      0.43      0.47       106

    accuracy                           0.49       200
   macro avg       0.49      0.49      0.49       200
```

In [ ]: