In [23]:
```
#importing modules
import pandas as pd
import numpy as np
import nltk
nltk.download('stopwords')
import re
from nltk.corpus import stopwords
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\DEBASMITA\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\stopwords.zip.
```

In [24]:
```
#loading dataset
df = pd.read_csv('SPAM text message 20170820 - Data.csv')
df.tail()
df.head()
```

Out[24]:

| | Category | Message |
|---|---|---|
| 0 | ham | Go until jurong point, crazy.. Available only ... |
| 1 | ham | Ok lar... Joking wif u oni... |
| 2 | spam | Free entry in 2 a wkly comp to win FA Cup fina... |
| 3 | ham | U dun say so early hor... U c already then say... |
| 4 | ham | Nah I don't think he goes to usf, he lives aro... |

In [25]:
```
#get necessary columns for processing
df = df[['Message', 'Category']]
# df.rename(columns={'Message': 'sms', 'Category': 'label'}, inplace=True)
df = df.rename(columns={'Message': 'sms', 'Category': 'label'})
df.head()
```

Out[25]:

| | sms | label |
|---|---|---|
| 0 | Go until jurong point, crazy.. Available only ... | ham |
| 1 | Ok lar... Joking wif u oni... | ham |
| 2 | Free entry in 2 a wkly comp to win FA Cup fina... | spam |
| 3 | U dun say so early hor... U c already then say... | ham |
| 4 | Nah I don't think he goes to usf, he lives aro... | ham |

In [26]:
```
#preprocessing the dataset
# check for null values
df.isnull().sum()
```

Out[26]:
```
sms      0
label    0
dtype: int64
```

In [28]:
```
STOPWORDS = set(stopwords.words('english'))

def clean_text(text):
    # convert to lowercase
    text = text.lower()
    # remove special characters
    text = re.sub(r'[^0-9a-zA-Z]', ' ', text)
    # remove extra spaces
    text = re.sub(r'\s+', ' ', text)
    # remove stopwords
    text = " ".join(word for word in text.split() if word not in STOPWORDS)
    return text
```

In [31]:
```
# clean the messages
df['clean_text'] = df['sms'].apply(clean_text)
df.head()
```

Out[31]:

| | sms | label | clean_text |
|---|---|---|---|
| 0 | Go until jurong point, crazy.. Available only ... | ham | go jurong point crazy available bugis n great ... |
| 1 | Ok lar... Joking wif u oni... | ham | ok lar joking wif u oni |
| 2 | Free entry in 2 a wkly comp to win FA Cup fina... | spam | free entry 2 wkly comp win fa cup final tkts 2... |
| 3 | U dun say so early hor... U c already then say... | ham | u dun say early hor u c already say |
| 4 | Nah I don't think he goes to usf, he lives aro... | ham | nah think goes usf lives around though |

In [32]:
```
#input split
X = df['clean_text']
y = df['label']
```

In [34]:
```
#model training
from sklearn.pipeline import Pipeline
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import classification_report
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer, TfidfTransformer

def classify(model, X, y):
    # train test split
    x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42, shuffle=True, stratify=y)
    # model training
    pipeline_model = Pipeline([('vect', CountVectorizer()),
                               ('tfidf', TfidfTransformer()),
                               ('clf', model)])
    pipeline_model.fit(x_train, y_train)

    print('Accuracy:', pipeline_model.score(x_test, y_test)*100)

#     cv_score = cross_val_score(model, X, y, cv=5)
#     print("CV Score:", np.mean(cv_score)*100)
    y_pred = pipeline_model.predict(x_test)
    print(classification_report(y_test, y_pred))
```

In [35]:
```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
classify(model, X, y)
```

```
Accuracy: 96.55419956927494
              precision    recall  f1-score   support

         ham       0.96      1.00      0.98      1206
        spam       0.99      0.75      0.85       187

    accuracy                           0.97      1393
   macro avg       0.98      0.87      0.92      1393
weighted avg       0.97      0.97      0.96      1393
```

In [36]:
```
from sklearn.naive_bayes import MultinomialNB
model = MultinomialNB()
classify(model, X, y)
```

```
Accuracy: 96.48241206030151
              precision    recall  f1-score   support

         ham       0.96      1.00      0.98      1206
        spam       1.00      0.74      0.85       187

    accuracy                           0.96      1393
   macro avg       0.98      0.87      0.91      1393
weighted avg       0.97      0.96      0.96      1393
```

In [38]:
```
from sklearn.svm import SVC
model = SVC(C=3)
classify(model, X, y)
```

```
Accuracy: 97.98994974874373
              precision    recall  f1-score   support

         ham       0.98      1.00      0.99      1206
        spam       0.99      0.86      0.92       187

    accuracy                           0.98      1393
   macro avg       0.99      0.93      0.95      1393
weighted avg       0.98      0.98      0.98      1393
```

In [39]:
```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
classify(model, X, y)
```

```
Accuracy: 97.27207465900933
              precision    recall  f1-score   support

         ham       0.97      1.00      0.98      1206
        spam       1.00      0.80      0.89       187

    accuracy                           0.97      1393
   macro avg       0.98      0.90      0.94      1393
weighted avg       0.97      0.97      0.97      1393
```

In [ ]:
```
```