Language Classification

In [1]:
```
from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.svm import LinearSVC

from sklearn.pipeline import Pipeline
from sklearn.datasets import load_files
from sklearn.model_selection import train_test_split
from sklearn import metrics
```

In [2]:
```
# The training data folder

languages_data_folder = 'data/languages/paragraphs'

dataset = load_files(languages_data_folder)
```

In [3]:
```
dataset.target_names
```

Out[3]:
```
['ar', 'de', 'en', 'es', 'fr', 'it', 'ja', 'nl', 'pl', 'pt', 'ru']
```

In [4]:
```
# Split the dataset in training and test set:

docs_train, docs_test, y_train, y_test = train_test_split(
    dataset.data, dataset.target, test_size=0.5, random_state=0)
```

- Build a vectorizer that splits strings into sequence of 1 to 3 characters instead of word tokens

In [5]:
```
vectorizer = TfidfVectorizer(ngram_range=(1, 3), analyzer='char',
                             use_idf=False)
```

- Build a vectorizer / classifier pipeline using the previous analyzer

In [6]:
```
clf = Pipeline([
    ('vec', vectorizer),
    ('clf', LinearSVC(C=10000)),
])
```

- Fit the pipeline on the training set

In [7]:
```
clf.fit(docs_train, y_train);
```

- Predict the outcome on the testing set

In [8]:
```
y_predicted = clf.predict(docs_test)
```

In [9]:
```
# Print the classification report

print(metrics.classification_report(
    y_test, y_predicted, target_names=dataset.target_names))
```

```
              precision    recall  f1-score   support

          ar       1.00      1.00      1.00        11
          de       1.00      1.00      1.00        82
          en       1.00      1.00      1.00        68
          es       1.00      1.00      1.00        66
          fr       1.00      1.00      1.00        65
          it       1.00      1.00      1.00        39
          ja       1.00      1.00      1.00        35
          nl       1.00      1.00      1.00        26
          pl       1.00      1.00      1.00        21
          pt       1.00      1.00      1.00        53
          ru       1.00      1.00      1.00        29

    accuracy                           1.00       495
   macro avg       1.00      1.00      1.00       495
weighted avg       1.00      1.00      1.00       495
```

In [10]:
```
# Show the confusion matrix

cm = metrics.confusion_matrix(y_test, y_predicted)
print(cm)
```

```
[[11  0  0  0  0  0  0  0  0  0  0]
 [ 0 82  0  0  0  0  0  0  0  0  0]
 [ 0  0 68  0  0  0  0  0  0  0  0]
 [ 0  0  0 66  0  0  0  0  0  0  0]
 [ 0  0  0  0 65  0  0  0  0  0  0]
 [ 0  0  0  0  0 39  0  0  0  0  0]
 [ 0  0  0  0  0  0 35  0  0  0  0]
 [ 0  0  0  0  0  0  0 26  0  0  0]
 [ 0  0  0  0  0  0  0  0 21  0  0]
 [ 0  0  0  0  0  0  0  0  0 53  0]
 [ 0  0  0  0  0  0  0  0  0  0 29]]
```

In [ ]:
```

```

In [11]:
```
# Predict the result on some short new sentences:

sentences = [
    'This is a language detection test.',
    'Ceci est un test de d\xe9tection de la langue.',
    'Dies ist ein Test, um die Sprache zu erkennen.',
    'Questo è un test di rilevamento della lingua.'
]

predicted = clf.predict(sentences)
```

In [12]:
```
for s, p in zip(sentences, predicted):
    print("The language of {} is '{}'".format(s, dataset.target_names[p]))
```

```
The language of This is a language detection test. is 'en'
The language of Ceci est un test de détection de la langue. is 'es'
The language of Dies ist ein Test, um die Sprache zu erkennen. is 'de'
The language of Questo è un test di rilevamento della lingua. is 'it'
```

In [ ]:
```

```