HƯỚNG DẪN BÀI THỰC HÀNH 3

Dataset: **Iris flower dataset** có sẵn trong thư viện scikit-learn. Hoặc download trên elearning.

Trước tiên, chúng ta cần khai các thư viện.:

In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
import sys
print(sys.version)

from sklearn import neighbors, datasets
```

```
2.7.13 (v2.7.13:a06454b1afa1, Dec 17 2016, 12:39:47)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)]
```

Tiếp theo, load dữ liệu và hiện thị vài dữ liệu mẫu.

Các class được gán nhãn là 0, 1, và 2.

In [2]:

```
iris = datasets.load_iris()
iris_X = iris.data
iris_y = iris.target
print 'Number of classes: %d' %len(np.unique(iris_y))
print 'Number of data points: %d' %len(iris_y)

X0 = iris_X[iris_y == 0,:]
print '\nSamples from class 0:\n', X0[:5,:]

X1 = iris_X[iris_y == 1,:]
print '\nSamples from class 1:\n', X1[:5,:]

X2 = iris_X[iris_y == 2,:]
print '\nSamples from class 2:\n', X2[:5,:]
```

```
Number of classes: 3
Number of data points: 150

Samples from class 0:
[[ 5.1  3.5  1.4  0.2]
 [ 4.9  3.   1.4  0.2]
 [ 4.7  3.2  1.3  0.2]
 [ 4.6  3.1  1.5  0.2]
 [ 5.   3.6  1.4  0.2]]

Samples from class 1:
[[ 7.   3.2  4.7  1.4]
 [ 6.4  3.2  4.5  1.5]
 [ 6.9  3.1  4.9  1.5]
 [ 5.5  2.3  4.   1.3]
```

```
 [ 6.5  2.8  4.6  1.5]]

Samples from class 2:
[[ 6.3  3.3  6.   2.5]
 [ 5.8  2.7  5.1  1.9]
 [ 7.1  3.   5.9  2.1]
 [ 6.3  2.9  5.6  1.8]
 [ 6.5  3.   5.8  2.2]]
```

Tách training và test sets

Giả sử chúng ta muốn dùng 50 điểm dữ liệu cho test set, 100 điểm còn lại cho training set. Scikit-learn có một hàm số cho phép chúng ta ngẫu nhiên lựa chọn các điểm này, như sau:

In [3]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(
    iris_X, iris_y, test_size=50)

print "Training size: %d" %len(y_train)
print "Test size    : %d" %len(y_test)
```

```
Training size: 100
Test size    : 50
```

Giả sử xét trường hợp đơn giản K=1, tức là với mỗi điểm test data, ta chỉ xét 1 điểm train gần nhất và lấy label của điểm đó để dự đoán cho điểm test này.

In [4]:

```
clf = neighbors.KNeighborsClassifier(n_neighbors = 1, p = 2)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

print "Print results for first 20 test data points:"
print "Predicted labels: ", y_pred[20:40]
print "Ground truth    : ", y_test[20:40]
```

```
Print results for first 20 test data points:
Predicted labels:  [2 0 2 2 0 0 1 2 1 2 1 0 2 0 2 0 0 0 1 0]
Ground truth    :  [2 0 2 1 0 0 1 2 1 2 1 0 2 0 2 0 0 0 2 0]
```

Kết quả cho thấy kết quả dự đoán gần giống với label thật của test data, chỉ có 2 điểm trong số 20 điểm được hiển thị có kết quả sai lệch. Ở đây chúng ta làm quen với khái niệm mới: *ground truth*. Một cách đơn giản, *ground truth* chính là dữ liệu *thực sự* của các điểm trong test data. Khái niệm này được dùng nhiều trong Machine Learning, hy vọng lần tới các bạn gặp thì sẽ nhớ ngay nó là gì.

Phương pháp đánh giá (evaluation method)

Để đánh giá độ chính xác của thuật toán KNN classifier này, chúng ta xem xem có bao nhiêu điểm trong test data được dự đoán đúng. Số lượng này chia cho tổng số lượng trong tập test data sẽ ra độ chính xác. Scikit-learn cung cấp hàm số accuracy_score để thực hiện công việc này.

In [5]:

```
from sklearn.metrics import accuracy_score
print "Accuracy of 1NN: %.2f %%" %(100*accuracy_score(y_test,
y_pred))
```

```
Accuracy of 1NN: 94.00 %
```

1NN đã cho chúng ta kết quả là 94%

Tham số sử dụng p = 2 nghĩa là gì?

Các bạn cũng có thể thử bằng cách thay p = 1? (Xem thêm sklearn.neighbors.KNeighborsClassifier)

Nhận thấy rằng chỉ xét 1 điểm gần nhất có thể dẫn đến kết quả sai nếu điểm đó là nhiễu. Một cách có thể làm tăng độ chính xác là tăng số lượng điểm lân cận lên, ví dụ 10 điểm, và xem xem trong 10 điểm gần nhất, class nào chiếm đa số thì dự đoán kết quả là class đó. Kỹ thuật này được gọi là major voting.

In [6]:

```
clf = neighbors.KNeighborsClassifier(n_neighbors = 10, p = 2)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

print "Accuracy of 10NN with major voting: %.2f %%"
%(100*accuracy_score(y_test, y_pred))
```

```
Accuracy of 10NN with major voting: 96.00 %
```

Kết quả đã tăng lên 98%,