手写字识别

In [1]:
```
import numpy as np
```

In [2]:
```
X = np.load('digits/digits_images.npy')    # images
y = np.load('digits/digits_lable.npy')       # labels
```

In [3]:
```
X
```

Out[3]:
```
array([[ 0.,  0.,  5., ...,  0.,  0.,  0.],
       [ 0.,  0.,  0., ..., 10.,  0.,  0.],
       [ 0.,  0.,  0., ..., 16.,  9.,  0.],
       ...,
       [ 0.,  0.,  1., ...,  6.,  0.,  0.],
       [ 0.,  0.,  2., ..., 12.,  0.,  0.],
       [ 0.,  0., 10., ..., 12.,  1.,  0.]])
```

In [4]:
```
y
```

Out[4]:
```
array([0, 1, 2, ..., 8, 9, 8])
```

In [5]:
```
from ML.model_selection import train_test_split
```

In [6]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size
=0.1)
```

In [7]:
```
from ML.knn import KNeighborsClassifier
```

In [8]:
```
X_train.shape[0]
```

Out[8]:
```
1618
```

In [14]:

```
%%time
best_acc_rate = 0
best_k = -1
best_p = -1

knn_clf = KNeighborsClassifier()
knn_clf.fit(X_train, y_train)
for k in range(1, 11):
    for p in range(1, 6):
        acc_rate = knn_clf.accuracy_rate(X_test, y_test, k, p)
        if acc_rate > best_acc_rate:
            best_acc_rate = acc_rate
            best_k = k
            best_p = p
print("best_accuracy_rate={}, best_k={}, best_p={}".format(best_acc
_rate, best_k, best_p))
```

```
best_accuracy_rate=0.9888268156424581, best_k=5, best_p=4
CPU times: user 2min 14s, sys: 415 ms, total: 2min 14s
Wall time: 2min 16s
```