In [1]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
import numpy as np
from sklearn.datasets import load_digits
import matplotlib.pyplot as plt

digits = load_digits()
```

In [2]:

```
#Use train test split
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(digits.data,digits.target,test_size
=0.3)
```

Logistic

In [3]:

```
lr = LogisticRegression(solver='liblinear',multi_class='ovr')
lr.fit(X_train, y_train)
lr.score(X_test, y_test)
```

Out[3]:

```
0.9537037037037037
```

In [4]:

```
#SVM

svm = SVC(gamma='auto')
svm.fit(X_train, y_train)
svm.score(X_test, y_test)
```

Out[4]:

```
0.4074074074074074
```

In [5]:

```
#random forest

rf = RandomForestClassifier(n_estimators=40)
rf.fit(X_train, y_train)
rf.score(X_test, y_test)
```

Out[5]:

```
0.9592592592592593
```

In [6]:

```
#As every time we excute it change the value every time so we use
#Cross validation to solve this error
```

KFold cross validation

In [7]:

```
#Basic example
from sklearn.model_selection import KFold
kf = KFold(n_splits=3)
kf
```

Out[7]:

```
KFold(n_splits=3, random_state=None, shuffle=False)
```

In [8]:

```
for train_index, test_index in kf.split([1,2,3,4,5,6,7,8,9]):
    print(train_index, test_index)
```

```
[3 4 5 6 7 8] [0 1 2]
[0 1 2 6 7 8] [3 4 5]
[0 1 2 3 4 5] [6 7 8]
```

In [9]:

```
#Now Use KFold for our digits example

def get_score(model, X_train, X_test, y_train, y_test):
    model.fit(X_train, y_train)
    return model.score(X_test, y_test)
```

In [10]:

```
from sklearn.model_selection import StratifiedKFold
folds = StratifiedKFold(n_splits=3)

scores_logistic = []
scores_svm = []
scores_rf = []

for train_index, test_index in folds.split(digits.data,digits.target):
    X_train, X_test, y_train, y_test = digits.data[train_index], digits.data[test_index
], \
                                       digits.target[train_index], digits.target[test_i
ndex]
    scores_logistic.append(get_score(LogisticRegression(solver='liblinear',multi_class=
'ovr'), X_train, X_test, y_train, y_test))
    scores_svm.append(get_score(SVC(gamma='auto'), X_train, X_test, y_train, y_test))
    scores_rf.append(get_score(RandomForestClassifier(n_estimators=40), X_train, X_test
, y_train, y_test))
```

In [11]:

```
scores_logistic
```

Out[11]:

```
[0.8948247078464107, 0.9532554257095158, 0.9098497495826378]
```

In [12]:

```
scores_svm
```

Out[12]:

```
[0.3806343906510851, 0.41068447412353926, 0.5125208681135225]
```

In [13]:

```
scores_rf
```

Out[13]:

```
[0.9432387312186978, 0.9449081803005008, 0.9165275459098498]
```

In [14]:

```
#cross_val_score function
```

In [15]:

```
from sklearn.model_selection import cross_val_score
```

In [16]:

```
cross_val_score(LogisticRegression(solver='liblinear',multi_class='ovr'), digits.data,
digits.target,cv=3)
```

Out[16]:

```
array([0.89482471, 0.95325543, 0.90984975])
```

In [17]:

```
cross_val_score(SVC(gamma='auto'), digits.data, digits.target,cv=3)
```

Out[17]:

```
array([0.38063439, 0.41068447, 0.51252087])
```

In [18]:

```
cross_val_score(RandomForestClassifier(n_estimators=40),digits.data, digits.target,cv=3
)
```

Out[18]:

```
array([0.92153589, 0.94657763, 0.91819699])
```