In [1]:

```
from sklearn.datasets import load_iris
```

In [2]:

```
iris = load_iris()
```

In [3]:

```
X,Y = iris.data,iris.target
```

In [4]:

```
from sklearn.model_selection import train_test_split
```

In [5]:

```
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,random_state=0)
```

In [6]:

```
X.shape
```

Out[6]:

```
(150, 4)
```

In [7]:

```
print(X_train.shape)
print(Y_train.shape)
```

```
(112, 4)
(112,)
```

In [8]:

```
from sklearn.linear_model import LogisticRegression
```

In [9]:

```
logreg = LogisticRegression(solver='lbfgs',multi_class='auto',max_iter=1000)
logreg.fit(X_train,Y_train)
logreg.score(X_test,Y_test)
```

Out[9]:

```
0.9736842105263158
```

In [10]:

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

In [11]:

```
scores=cross_val_score(logreg, X,Y,cv=5)
```

In [12]:

```
scores.mean()
```

Out[12]:

```
0.9733333333333334
```

In [13]:

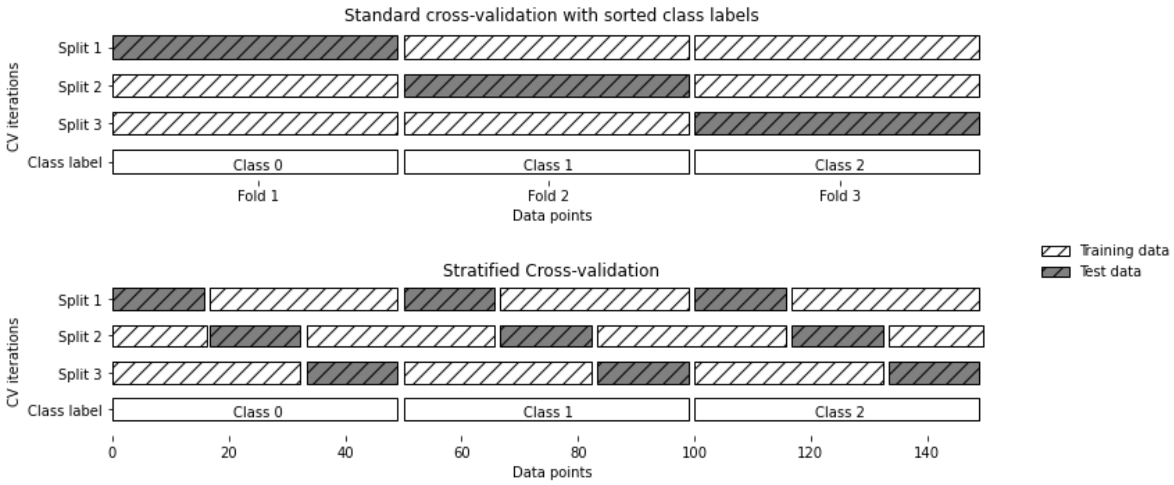
```
iris.target
```

Out[13]:

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

In [14]:

```
import mglearn
mglearn.plots.plot_stratified_cross_validation()
```



In [15]:

```
from sklearn.model_selection import KFold
```

In [16]:

```
kfold = KFold(n_splits=3,shuffle=True,random_state=0)
```

In [17]:

```
cross_val_score(logreg,iris.data,iris.target,cv=kfold)
```

Out[17]:

```
array([0.98, 0.96, 0.96])
```

In [18]:

```
from sklearn.model_selection import LeaveOneOut
loo=LeaveOneOut()
scores = cross_val_score(logreg,iris.data,iris.target,cv=loo)
scores.mean()
```

Out[18]:

```
0.9666666666666667
```

In [19]:

```
import numpy as np
from sklearn.svm import SVC
svm=SVC().fit(X_train,Y_train)
svm_params={"C":np.arange(1,3), "gamma" : ['scale'],
            "kernel":["rbf","linear","poly"]}
```

In [20]:

```
rbf = SVC(kernel='rbf')
linear = SVC(kernel='linear')
poly = SVC(kernel='poly')
```

In [21]:

```
rbf.fit(X_train, Y_train)
```

Out[21]:

```
SVC()
```

In [22]:

```
rbf.score(X_train, Y_train)
```

Out[22]:

```
0.9642857142857143
```

In [23]:

```
linear.fit(X_train, Y_train)
```

Out[23]:

```
SVC(kernel='linear')
```

In [24]:

```
linear.score(X_train, Y_train)
```

Out[24]:

```
0.9821428571428571
```

In [25]:

```
poly.fit(X_train, Y_train)
```

Out[25]:

```
SVC(kernel='poly')
```

In [26]:

```
poly.score(X_train, Y_train)
```

Out[26]:

```
0.9910714285714286
```