techniques-for-cross-validation

October 29, 2023

1 Cross-Validation implementation

2 There are different Cross-Validation techniques

- Hold-out cross-validation
- K-folds cross-validation
- Leave-one-out cross-validation
- Leave-p-out cross-validation
- Stratified K-folds cross-validation
- Repeated K-folds cross-validation
- Group K-Fold Cross-Validation
- Time series CV cross-validation

Hold-Out based Validation

[4]:
```
import numpy as np
from sklearn.model_selection import train_test_split
X, y = np.arange(10).reshape((5, 2)), range(5)
X_train, X_test, y_train, y_test = train_test_split(X, y,test_size=0.
 ↪2,random_state=111)
```

K-folds cross-validation

[5]:
```
import numpy as np
from sklearn.model_selection import KFold

X = np.array([[1, 2], [3, 4], [1, 2], [3, 4]])
y = np.array([1, 2, 3, 4])
kf = KFold(n_splits=2)

for train_index, test_index in kf.split(X):
    print("TRAIN:", train_index, "TEST:", test_index)
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]
```

```
TRAIN: [2 3] TEST: [0 1]
TRAIN: [0 1] TEST: [2 3]
```

- Leave-one-out cross-validation

[6]:
```python
import numpy as np
from sklearn.model_selection import LeaveOneOut

X = np.array([[1, 2], [3, 4]])
y = np.array([1, 2])
loo = LeaveOneOut()

for train_index, test_index in loo.split(X):
    print("TRAIN:", train_index, "TEST:", test_index)
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]
```

```
TRAIN: [1] TEST: [0]
TRAIN: [0] TEST: [1]
```

- Leave-p-out cross-validation

[7]:
```python
import numpy as np
from sklearn.model_selection import LeavePOut

X = np.array([[1, 2], [3, 4], [5, 6], [7, 8]])
y = np.array([1, 2, 3, 4])
lpo = LeavePOut(2)

for train_index, test_index in lpo.split(X):
    print("TRAIN:", train_index, "TEST:", test_index)
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]
```

```
TRAIN: [2 3] TEST: [0 1]
TRAIN: [1 3] TEST: [0 2]
TRAIN: [1 2] TEST: [0 3]
TRAIN: [0 3] TEST: [1 2]
TRAIN: [0 2] TEST: [1 3]
TRAIN: [0 1] TEST: [2 3]
```

- Stratified k-Fold cross-validation

[8]:
```python
import numpy as np
from sklearn.model_selection import StratifiedKFold

X = np.array([[1, 2], [3, 4], [1, 2], [3, 4]])
y = np.array([0, 0, 1, 1])
skf = StratifiedKFold(n_splits=2)

for train_index, test_index in skf.split(X, y):
    print("TRAIN:", train_index, "TEST:", test_index)
    X_train, X_test = X[train_index], X[test_index]
    y_train, y_test = y[train_index], y[test_index]
```

```
TRAIN: [1 3] TEST: [0 2]
TRAIN: [0 2] TEST: [1 3]
```

- Repeated k-Fold cross-validation

```
[9]: import numpy as np
     from sklearn.model_selection import RepeatedKFold

     X = np.array([[1, 2], [3, 4], [1, 2], [3, 4]])
     y = np.array([0, 0, 1, 1])
     rkf = RepeatedKFold(n_splits=2, n_repeats=2, random_state=42)

     for train_index, test_index in rkf.split(X):
         print("TRAIN:", train_index, "TEST:", test_index)
         X_train, X_test = X[train_index], X[test_index]
         y_train, y_test = y[train_index], y[test_index]
```

```
TRAIN: [0 2] TEST: [1 3]
TRAIN: [1 3] TEST: [0 2]
TRAIN: [0 2] TEST: [1 3]
TRAIN: [1 3] TEST: [0 2]
```

- Group K-Fold Cross-Validation

```
[ ]:
```