Resampling Techniques

Professor Ernesto Lee

Evaluate ML Algorithms

Better ways to Train Data

- ˆ Train and Test Sets.
- ˆ k-fold Cross-Validation.
- ˆ Leave One Out Cross-Validation.
- ˆ Repeated Random Test-Train Splits.

Split into Train and Test Sets

```
from pandas import read_csv

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LogisticRegression

names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']

dataframe = read_csv('pima-indians-diabetes.data.csv', names=names)

array = dataframe.values

X = array[:,0:8]

Y = array[:,8]

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.33,
random_state=7)

model = LogisticRegression(solver='liblinear')

model.fit(X_train, Y_train)

result = model.score(X_test, Y_test)

print("Accuracy: %.3f%%" % (result*100.0))
```

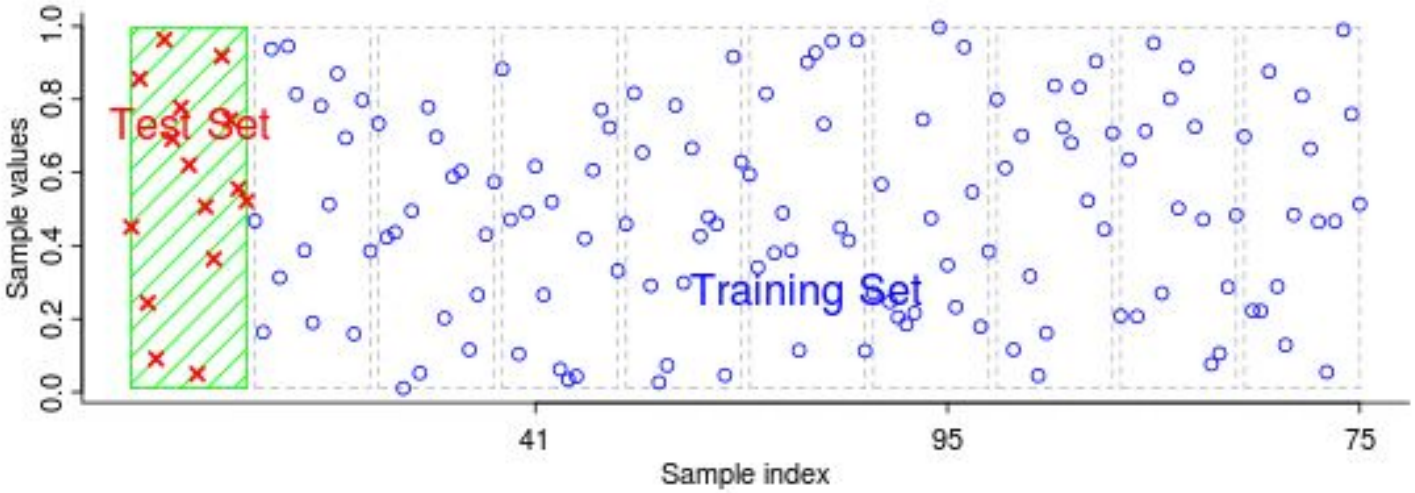
Exercise

https://archive-beta.ics.uci.edu/ml/datasets/statlog+german+credit+data

Download and split the data into a train and test set.

K-Fold Cross Validation

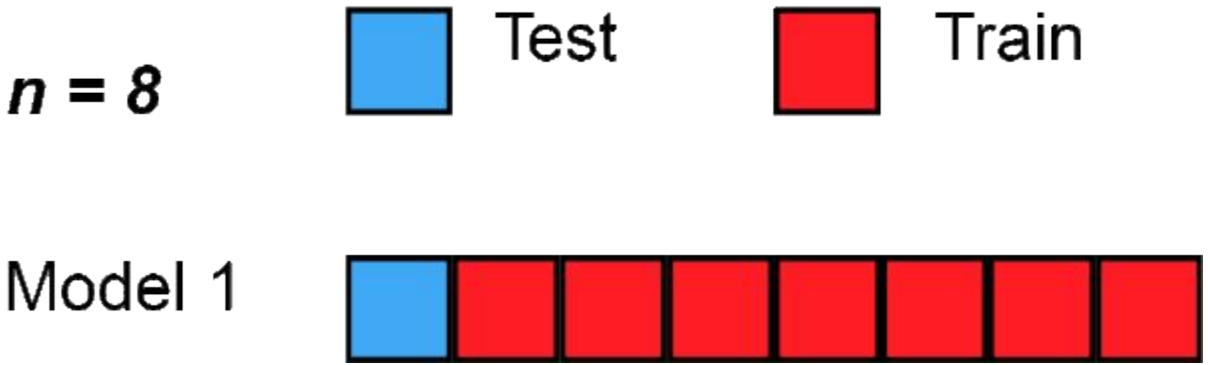
```
# Evaluate using Cross Validation
from pandas import read_csv
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
filename = 'pima-indians-diabetes.data.csv'
names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
dataframe = read_csv(filename, names=names)
array = dataframe.values
X = array[:,0:8]
Y = array[:,8]
kfold = KFold(n_splits=10, random_state=7)
model = LogisticRegression(solver='liblinear')
results = cross_val_score(model, X, Y, cv=kfold)
print("Accuracy: %.3f%% (%.3f%%)" % (results.mean()*100.0, results.std()*100.0))
```


Test Set
Training Set
Sample values
0.0
0.2
0.4
0.6
0.8
1.0
41
95
75
Sample index

Exercise

Use K-Fold Cross Validation to test and train (using Log Regression) on this dataset:

https://archive-beta.ics.uci.edu/ml/datasets/hepatitis



Leave One Out Cross Validation

```
# Evaluate using Leave One Out Cross Validation
from pandas import read_csv
from sklearn.model_selection import LeaveOneOut
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
filename = 'pima-indians-diabetes.data.csv'
names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
dataframe = read_csv(filename, names=names)
array = dataframe.values
X = array[:,0:8]
Y = array[:,8]
loocv = LeaveOneOut()
model = LogisticRegression(solver='liblinear')
results = cross_val_score(model, X, Y, cv=loocv)
print("Accuracy: %.3f%% (%.3f%%)" % (results.mean()*100.0, results.std()*100.0))
```

Repeated Random Test-Train Splits

```
# Evaluate using Shuffle Split Cross Validation
from pandas import read_csv
from sklearn.model_selection import ShuffleSplit
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
filename = 'pima-indians-diabetes.data.csv'
names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
dataframe = read_csv(filename, names=names)
array = dataframe.values
X = array[:,0:8]
Y = array[:,8]
n_splits = 10
test_size = 0.33
seed = 7
kfold = ShuffleSplit(n_splits=n_splits, test_size=test_size, random_state=seed)
model = LogisticRegression(solver='liblinear')
results = cross_val_score(model, X, Y, cv=kfold)
print("Accuracy: %.3f%% (%.3f%%)" % (results.mean()*100.0, results.std()*100.0))
```

What Techniques to Use When

- K-fold cross validation is the gold standard
- Use train/test for speed with slow algorithms
- Consider the others only when trying to balance variance, performance and speed

Summary

- Train and Test Sets.
- Cross-Validation.
- Leave One Out Cross-Validation.
- Repeated Random Test-Train Splits.