Frequentist Machine Learning

Assignment 3

Jonathan Lam, Tiffany Yu, Harris Paspuleti

Re-implement the example in section 7.10.2 using any simple, out of the box classifier (like K nearest neighbors from sci-kit). Reproduce the results for the incorrect and correct way of doing cross-validation.

In [ ]:

```
#setting up

import pandas as pd
import numpy as np
from sklearn import preprocessing
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import RepeatedKFold

# Consider a scenario with N = 50 samples in two equal-sized classes, and
# p = 5000 quantitative predictors (standard Gaussian) that are independent
# of the class labels.

# generate 50 random samples of N(0,1) data w/ 5000 features
X = np.random.normal(0,1,[50, 5000])
# The true (test) error rate of any classifier is 50%
Y = np.concatenate([np.zeros(25), np.ones(25)])
np.random.shuffle(Y)

#INCORRECT
CV_correct = []

# Screen the predictors: find a subset of “good” predictors that show fairly
# strong (univariate) correlation with the class labels
# preprocessing.MinMaxScaler needed because non-negative values needed for Sele
ctKBest
X_new = preprocessing.MinMaxScaler().fit_transform(X)
X_new = SelectKBest(chi2, k=100).fit_transform(X_new, Y)

# Using just this subset of predictors, build a multivariate classifier.
neigh = KNeighborsClassifier(n_neighbors=1)

# Use cross-validation to estimate the unknown tuning parameters and to estimat
e
# the prediction error of the final model.
# source: https://scikit-learn.org/stable/modules/generated/sklearn.model_selec
tion.RepeatedKFold.html
rkf = RepeatedKFold(n_splits=5, n_repeats=50)
for train_index, test_index in rkf.split(X_new):
    X_train, X_test = X_new[train_index], X_new[test_index]
    Y_train, Y_test = Y[train_index], Y[test_index]
    neigh.fit(X_train, Y_train)
    CV_correct.append(1-neigh.score(X_test, Y_test))

print("Average CV Error Rate:", np.around(np.array(CV_correct).mean(), 3))

"""
What has happened? The problem is that the predictors have an unfair
advantage, as they were chosen in step (1) on the basis of all of the samples.
Leaving samples out after the variables have been selected does not cor-
rectly mimic the application of the classifier to a completely independent
test set, since these predictors “have already seen” the left out samples.
"""

# CORRECT
CV_correct = []
kbest = SelectKBest(chi2, k=100)

# divide the samples into K CV folds at random
rkf = RepeatedKFold(n_splits=5, n_repeats=50)
for train_index, test_index in rkf.split(X):
```

```
Average CV Error Rate: 0.038
Average CV Error Rate: 0.547
```

The average CV error rate from the incorrect way using cross-validationis much lower than the average CV error rate using the correct way using cross-validation. The correct way has a higher average error because it has not seen the test samples, so it cannot use them as predictors.