ps3_problem2

May 14, 2020

0.1 IDS/ACM/CS 158: Fundamentals of Statistical Learning

0.1.1 PS3, Problem 2: Leave-One-Out Cross Validation For Model Selection

Name: Li, Michael

Email address: mlli@caltech.edu

Notes: Please use python 3.6

You are required to properly comment and organize your code.

- Helper functions (add/remove part label according to the specific question requirements)

[1]:
```
import numpy as np
import numpy.random
import matplotlib.pyplot as plt
import numpy.matlib

def find_beta(data):
    """
    data - a matrix where each row corresponds to the
           p predictors in the first p columns and
           the observed output y in the final column

    returns the OLS estimate of the regression parameter
    """
    x = data[:,:-1]
    y = data[:,-1]

    # add bias term to training data
    bias = np.matlib.repmat(1, len(x), 1)
    x = np.concatenate((bias, x), axis=1)

    # calculate beta
    intermediate = np.matmul(x.transpose(), x)
    inverse_intermediate = np.linalg.inv(np.array(intermediate))
    pseudo_x = np.matmul(inverse_intermediate, x.transpose())

    return np.matmul(pseudo_x, y), np.matmul(x, pseudo_x)
```

```python
def predict(ols, data):
    """
    ols - ols estimate of the regression parameter
    data - a matrix where each row corresponds to the
           p predictors in the first p columns and
           the observed output y in the final column

    returns the predictions for the observations in data
    """

    x_with_bias_term = np.insert(data[:-1], 0, 1)
    return np.matmul(x_with_bias_term, ols)

def leave_one_out_cv(data):
    """
    data - a matrix where each row corresponds to the
           p predictors in the first p columns and
           the observed output y in the final column

    returns the leave one out cross validation of the data
    """
    ols, hat = find_beta(data)
    return np.mean([((data[i][-1] - predict(ols, data[i])) / (1-hat[i][i]))**2␣
 ↪for i in range(len(data))])
```

[2]:
```python
# reformat data so we have 3 models
f_1_data = np.genfromtxt('dataset5.csv', delimiter=',',skip_header =1)
f_2_data = np.array([[f_1_data[i][0], np.sin(f_1_data[i][1]), f_1_data[i][2]]␣
 ↪for i in range(len(f_1_data))])
f_3_data = np.delete(f_1_data, 1, axis=1)
```

[3]:
```python
# calculate the leave one out cross validation for each dataset
f_1_err = leave_one_out_cv(f_1_data)
f_2_err = leave_one_out_cv(f_2_data)
f_3_err = leave_one_out_cv(f_3_data)
```

[4]:
```python
print("The Leave One Out Cross Validation for Model 1 is {}".format(f_1_err))
print("The Leave One Out Cross Validation for Model 2 is {}".format(f_2_err))
print("The Leave One Out Cross Validation for Model 3 is {}".format(f_3_err))
```

```
The Leave One Out Cross Validation for Model 1 is 1.1074945247730847
The Leave One Out Cross Validation for Model 2 is 1.0802973038999084
The Leave One Out Cross Validation for Model 3 is 1.500086491089816
```

From the leave one out cross validations, it looks like model 2 has the lowest estimated test error. Thus, I would definitively select model $f_2(\mathrm{X})$ as the best model using this metric since each model was trained and tested on the same data.