```
import scipy
import numpy as np
from liblinearutil import *

data = np.genfromtxt('hw4_train.dat')
data_test = np.genfromtxt('hw4_test.dat')

X = data[:, :-1]
X_test = data_test[:, :-1]

phi = X
phi_test = X_test


d = X.shape[1]
for i in range(d):
    for j in range(i, d):
        # print(i, j)
        phi = np.c_[phi, np.multiply(np.expand_dims(X[:, i], axis=1), np.expand_dims(X[:, j],
axis=1))]
        phi_test = np.c_[phi_test, np.multiply(np.expand_dims(X_test[:, i], axis=1),
np.expand_dims(X_test[:, j], axis=1))]
phi = np.c_[np.ones(data.shape[0]), phi]
phi_test = np.c_[np.ones(data_test.shape[0]), phi_test]
phi = np.c_[phi, np.expand_dims(data[:, -1], axis=1)]
phi_test = np.c_[phi_test, np.expand_dims(data_test[:, -1], axis=1)]


print("Q16: --- E_out ---")
prob = problem(phi[:, -1], phi[:, 0:-1])
for lmd_power in [-4, -2, 0, 2, 4]:
    lmd = 10**lmd_power
    C = 1/(2*lmd)
    param = parameter(f'-s 0 -c {C} -e 0.0000001 -q')
    model = train(prob, param, 'q')
    p_labs, p_acc, p_vals = predict(phi_test[:, -1], phi_test[:, 0:-1], model)

print("Q17: --- E_in ---")
prob = problem(phi[:, -1], phi[:, 0:-1])
for lmd_power in [-4, -2, 0, 2, 4]:
    lmd = 10**lmd_power
    C = 1/(2*lmd)
    param = parameter(f'-s 0 -c {C} -e 0.0000001 -q')
    model = train(prob, param, 'q')
    p_labs, p_acc, p_vals = predict(phi[:, -1], phi[:, 0:-1], model)

print("Q18: ---E_out---")
prob = problem(phi[:120, -1], phi[:120, 0:-1])
for lmd_power in [-4, -2, 0, 2, 4]:
    lmd = 10**lmd_power
    C = 1/(2*lmd)
    param = parameter(f'-s 0 -c {C} -e 0.0000001 -q')
    model = train(prob, param, 'q')
    p_labs, p_acc, p_vals = predict(phi[120:, -1], phi[120:, 0:-1], model)

print("Q19: ---E_out---")
prob = problem(phi[:, -1], phi[:, 0:-1])
for lmd_power in [-4, -2, 0, 2, 4]:
    lmd = 10**lmd_power
    C = 1/(2*lmd)
    param = parameter(f'-s 0 -c {C} -e 0.0000001 -q')
    model = train(prob, param, 'q')
    p_labs, p_acc, p_vals = predict(phi_test[:, -1], phi_test[:, 0:-1], model)

print("Q19: ---E_out---")
prob = problem(phi[:, -1], phi[:, 0:-1])
for lmd_power in [-4, -2, 0, 2, 4]:
    lmd = 10**lmd_power
    C = 1/(2*lmd)
```

```
    param = parameter(f'-s 0 -c {C} -e 0.0000001 -q')
    model = train(prob, param, 'q')
    p_labs, p_acc, p_vals = predict(phi_test[:, -1], phi_test[:, 0:-1], model)

print("Q20: ---E_out---")
prob = problem(phi[:, -1], phi[:, 0:-1])
for lmd_power in [-4, -2, 0, 2, 4]:
    acc = 0
    for k in range(5):
        prob = problem(np.delete(phi[:, -1], slice(k*40, (k+1)*40), axis=0), np.delete(phi[:,
0:-1], slice(k*40, (k+1)*40), axis=0))
        lmd = 10**lmd_power
        C = 1/(2*lmd)
        param = parameter(f'-s 0 -c {C} -e 0.0000001 -q')
        model = train(prob, param, '-q')
        p_labs, p_acc, p_vals = predict(phi[k*40:(k+1)*40, -1], phi[k*40:(k+1)*40, 0:-1], model,
'-q')
        acc += p_acc[0]
    acc /= 5
    print('acc= ', acc)

# reference for multiple choices :
http://www.kaspercpa.com/statisticalreview.htm#:~:text=The%20variance%20of%20the%20sum,the%20rando
m%20variables%20are%20independent.&text=The%20covariance%20of%20a%20random,does%20not%20change%20t
heir%20covariances.
```