Machine Learning – Homework 1

資訊工程學系 碩一 608410117 沈濃翔

可至github觀看完整python code:

<https://github.com/Yuki23329626/ml-learning>

1. 首先引入幾個套件 numpy, random, matplotlib 跟 sklearn裡面跟error measurement 有關的套件

Linear\_regression function definition:

def linear\_regression ( x\_training, y\_training, x\_valid, y\_valid, sum\_training, sum\_valid):

  wlin = np.linalg.inv(x\_training.T.dot( x\_training)).dot(x\_training.T.dot(y\_training))

  return  wlin

y=2x+上生成20 data points(with equal spacing) x, with zero mean Gaussian noise

x = np.linspace(-3, 3, 20)

y = np.zeros((20,1))

mu, sigma = 0, 1 # mean and standard deviation

noise = np.random.normal(mu, sigma, 20) #Create Gaussian Noise

y = 2\*x + noise

分割data points為training and testing set(75% for training, 25% for testing)

x = np.reshape( np.append( x, np.ones(20)), (2, 20)).T

x\_training, x\_test, y\_training, y\_test = train\_test\_split(x, y, test\_size=0.25)

使用上述linear\_regression function求出回歸線後分別算出training error 與 testing error

def calculate\_error\_L1 ( x\_training, y\_training, x\_valid, y\_valid, sum\_training, sum\_valid):

    wlin = linear\_regression( x\_training, y\_training, x\_valid, y\_valid, sum\_training, sum\_valid)

    error\_training = abs((x\_training.dot(wlin) - y\_training).sum()/sum\_training)

    error\_testing = abs((x\_valid.dot(wlin) - y\_valid).sum()/sum\_valid)

    return  wlin, error\_training, error\_testing

cross-validation 則採用 sklearn 裡的 model\_selection (KFold and LeaveOneOut)

def leave\_one\_out(x ,y ):

    leaveOneOut = LeaveOneOut()

    leave\_one\_out = 0

    for train\_index, test\_index in leaveOneOut.split(x) :

        x\_training, X\_test = x[train\_index], x[test\_index]

        y\_training, Y\_test = y[train\_index], y[test\_index]

        leave\_one\_out += calculate\_error\_L1(x\_training, y\_training, X\_test, Y\_test, 14, 1)[2]

    return(leave\_one\_out/15)

def five\_fold(x, y):

    #### five fold error

    kf = KFold (n\_splits = 5)

    five\_fold\_error = 0

    for index\_training , index\_testing in kf.split( x\_training ):

        #print ("Train: " ,train\_index , "Test: ", test\_index)

        X\_training, X\_testing = x[index\_training], x[index\_testing]

        Y\_training, Y\_testing = y[index\_training], y[index\_testing]

        five\_fold\_error += calculate\_error\_L1(X\_training, Y\_training, X\_testing, Y\_testing,12, 3)[2]

    return (five\_fold\_error/5)

最後，印出各自的error並畫出回歸線

x\_line = np.linspace (-3, 3 ,1000)

y\_line = x\_line \* wlin[0] + wlin[1]

fig, ax = plt.subplots()

ax.plot (x\_line, y\_line )

ax.scatter (x\_training[:,0], y\_training, c='k')

plt.show()

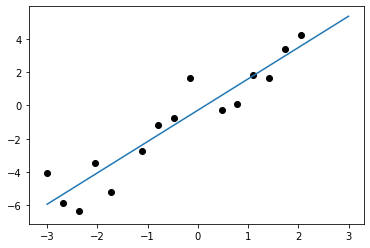
Question a:

Training error: 2.0724163126336256e-16

Testing error: 0.061130336971773266

Five Fold Error: 0.4987675459882105

Leave One out: 1.068429514830709



1. 求出regression function的degree分別在5, 10, 14下的error measurement

使用generator\_x()增加回歸線的degree

def generator\_x (x\_training, x\_test, p\_st, p\_end, s\_trn, s\_tst):

    for i in range (p\_st+1, p\_end+1) :

       x\_training = np.hstack(((x\_training[:,-2]\*\*i).reshape(s\_trn,1) ,x\_training))

       x\_test = np.hstack(((x\_test[:,-2]\*\*i).reshape(s\_tst,1) ,x\_test))

    return x\_training , x\_test

最後在帶回原本的calculate\_error\_L1() 和 cross-validation functions 求出error

並印出各degree下的error measurement和求出的線段

Question b:

--Degree 5--

Training error: 6.513308411134251e-16

Leave One out: 1.726853476750588

Five Fold Error: 0.729584053585077

Testing error: 0.16389489205215885

--Degree 10--

Training error: 9.238491453326484e-12

Leave One out: 5.457421307293006

Five Fold Error: 14.167276894786317

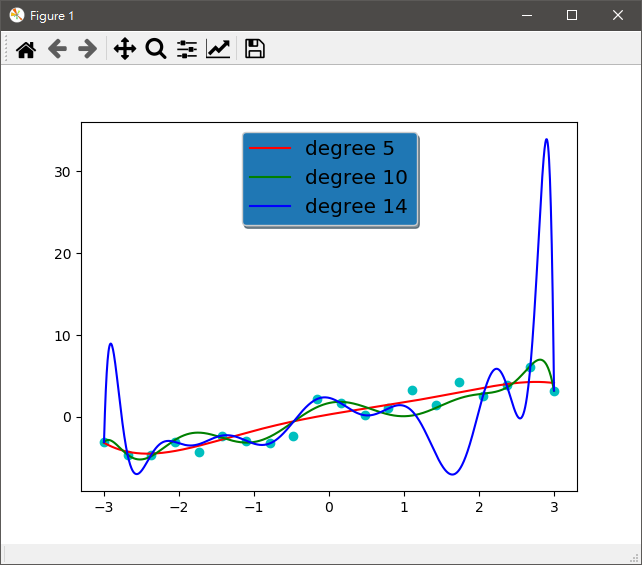
Testing error: 0.3016953241941689

--Degree 14--

Training error: 2.3982431057574988e-06

Leave One out: 224.74152737696875

Five Fold Error: 272.0571651777577

Testing error: 3.3286227531243187

可以發現degree越高，線段彎曲可能越多

1. 使用sin函數產生data points，可以想見印出的圖案可能會接近sin波

調整一些參數為y=sin(2x)+, noise ~ N(0, 0.04)(equal spacing) x

x = np.linspace(0, 1, 20)

y = np.zeros((20,1))

mu, sigma = 0, 0.04 # mean and standard deviation

noise = np.random.normal(mu, sigma, 20) #Create Gaussian Noise

y = np.sin(2\*np.pi\*x) + noise

一樣印出各個degree下的error與圖案

Question c:

--Degree 5--

Training error: 4.716471380117326e-11

Leave One out: 0.1527639901804861

Five Fold Error: 0.12600525818594915

Testing error: 0.016473033435046126

--Degree 10--

Training error: 0.019792418617954947

Leave One out: 0.7501776214034385

Five Fold Error: 7.964251484177825

Testing error: 0.38180491924213744

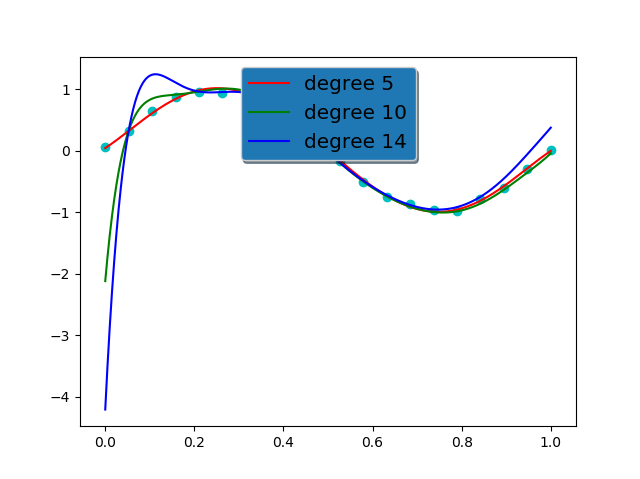
--Degree 14--

Training error: 0.05947466333058782

Leave One out: 9.83350281893189

Five Fold Error: 141.3765449250264

Testing error: 0.6991664286697561



Degree 越高未必越準確

1. 承接(b)，在degree 14的情況下變更data points的數量分別為60,160, 320

更改部分的code，新增size參數，在不同狀況下調整

size = 60

x = np.linspace(-3, 3, size)

y = np.zeros((size,1))

mu, sigma = 0, 1 # mean and standard deviation

noise = np.random.normal(mu, sigma, size) #Create Gaussian Noise

一樣印出各自的error measurement和圖示

Question d:

m = 60:

--Degree 14--

Training error: 2.8904975400855116e-09

Leave One out: 2.2553729354130363

Five Fold Error: 0.42400925924362587

Testing error: 2.8904975400855116e-09

m = 160:

--Degree 14--

Training error: 7.154951027950792e-09

Leave One out: 6.7255520589276125

Five Fold Error: 2.345233951988086

Testing error: 7.154951027950792e-09

m = 320:

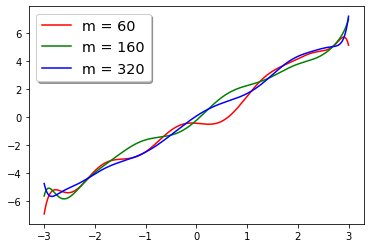
--Degree 14--

Training error: 6.173506787258513e-09

Leave One out: 13.500477548644637

Five Fold Error: 2.6007999254589436

Testing error: 6.173506787258513e-09



Data points 數量越多，應該能越接近原本的線

1. 實作redge\_regression function，在原本linear regression function的基礎上再加上一些變動

def ridge\_regression ( x\_training, y\_training, x\_valid, y\_valid, sum\_training, sum\_valid, lamda):

  wlin = np.linalg.inv(x\_training.T.dot( x\_training)+lamda\*np.eye(x\_training.shape[0])).dot(x\_training.T.dot(y\_training))

  return  wlin

其中 = 0, 0.001/m, 1/m, 1000/m(respectively); m=20

一樣計算error value與印出圖示

Question d λ=0:

--Degree 14--

Training error: 0.2179289957990372

Leave One out: 174.89921961491802

Five Fold Error: 109.12196671416223

Testing error: 0.2179289957990372

Question d λ=0.001/m:

--Degree 14--

Training error: 6.702936803056521e-06

Leave One out: 585.1958920932151

Five Fold Error: 477.53482544970257

Testing error: 6.702936803056521e-06

Question d λ=1/m:

--Degree 14--

Training error: 0.0005651681290709476

Leave One out: 86.3103599194199

Five Fold Error: 148.1696438047067

Testing error: 0.0005651681290709476

Question d λ=1000/m:

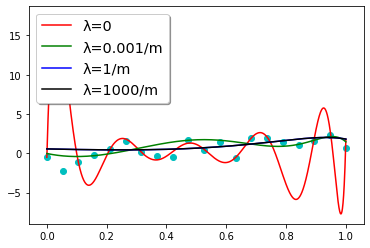
--Degree 14--

Training error: 0.39195893021513933

Leave One out: 227.596601952779

Five Fold Error: 147.71015134888972

Testing error: 0.39195893021513933



可以發現，λ的值越大，曲線的波動幅度會較低