Chapter 1

Question 1

Implement least square regression with help of matrix inversion

For this we simply write the code :

```
i = 0
dict = {"Iris-setosa":'1', "Iris-versicolor": '2',
    "Iris-virginica": '3'}
fname = 'iris.data'
fout = open ('iris-svm-input.txt', 'w')
with open (fname) as f:
   content = f.readlines()
content = [line[:-1] for line in content]
#print (content)
for x in content:
    i = 0
    words = x.split(",")
    if len(words) < 2:
        break
    while i < 4:
        if words[i] == 0:
            break;
        else:
            fout.write (dict[words[4]]+' '+str(i+1)+':'+words[i]+' ')
            i = i+1
    fout.write('\n')
f.close()
fout.close()
```

Chapter 2

Question 2

Implement stochastic gradient descent algorithm with step size 0.1 to solve ridge regression

we simply use the formula:

```
def LeastSquare(X, Y):
   X_ = X.T
   return (matmul(inv( matmul(X_,X)) , matmul(X_, Y)))
```

Chapter 3

Question 3

The relation between x and y can be non-linear. Hence to catch non-linear relationship we will generate

Featurematrix=[1, x, x**2 , . . . , x**10]

for this we code:

```
x_ = genfromtxt('Regressiondata/x.txt', delimiter=',')[np.newaxis]
x = x_.T

Featurematrix = np.hstack((x**0, x, x**2, x**3, x**4, x**5, x**6,
    x**7, x**8, x**9, x**10))
```

Chapter 4

Question 4

Model selection: find optimal lamda and plot error vs lamda graphs.
For this we loop in following loop:

```
def CrossValidation(X, Y, fold):
   #X.shape[0] = X.shape[0]
   set_length = X.shape[0]/fold
   TrainError = []
   TestError = []
   ValidationError = []
   L = []

   for x in range(-10,11):
      lamda = 2**x
      L = hstack((L, x))
      i = 0
      sqError = 0
      sqTrainError = 0
      sqTestError = 0
      while i*set_length + set_length <= X.shape[0]:
      CvX = np.vstack((X[:i*set_length], X[(i+1)*set_length:]))
      CvY = np.vstack((Y[:i*set_length],Y[(i+1)*set_length:]))

      w = RidgeRegression(CvX, CvY, lamda)

      error1 = norm(x=(matmul(CvX, w) - CvY), ord =2)
      sqTrainError = sqTrainError + error1

      error2 = norm(x=(matmul(X, w) - Y), ord =2)
      sqTestError = sqTestError + error1
```

```
        error = norm(matmul(X[i*set_length:(i+1)*set_length],
            w)-y[i*set_length:(i+1)*set_length])
        sqError = sqError + error

        i = i + set_length

    sqError = sqError / set_length**2
    sqTrainError = sqTrainError /
        ((X.shape[0]-set_length)*set_length)
    sqTestError = sqTestError / (set_length * Y.shape[0])

    ValidationError.append(sqError)
    TrainError.append(sqTrainError)
    TestError.append(sqTestError)
    if x == -10:
        prevError = sqError
        lamdaMin = lamda
    elif sqError<prevError:
        prevError = sqError
        lamdaMin = lamda

    print ("optimal lamda: "+str(lamdaMin))
```

And to plot we do the following in the same function:

```
plt.plot(L, TrainError, 'r')
plt.plot(L, ValidationError, 'g')
plt.plot(L, TestError, 'b')
plt.show()
```

I am only a learner: so I dont know how to add an image to latex. Next submission, I will also add plots in documents.

Thank you