In [1]:

```
import numpy as np
import math
import random
import scipy as sp
import matplotlib.pyplot as plt
import scipy.io as scio
import pprint as pp

%matplotlib inline
```

Question 7 - Logistic Regression Implementation

Part A

In [2]:

```
mat = scio.loadmat('HW2_Data/data1.mat')

X_trn = mat['X_trn']
Y_trn = mat['Y_trn']
X_tst = mat['X_tst']
Y_tst = mat['Y_tst']
data = [X_trn,Y_trn,X_tst,Y_tst]
```

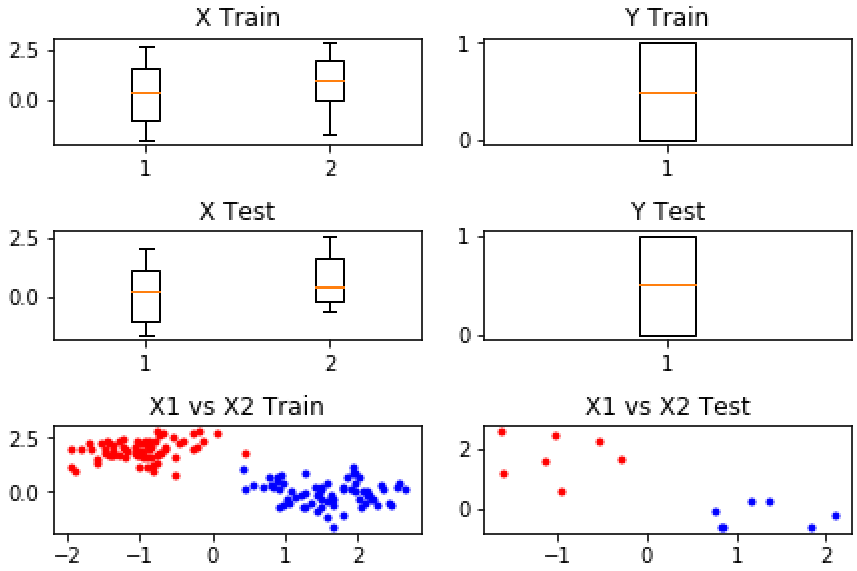
In [3]:

```
print('shape of the X data is [%d, %d]' % X_trn.shape)
print('shape of the Y data is [%d, %d]' % Y_trn.shape)
```

```
shape of the X data is [136, 2]
shape of the Y data is [136, 1]
```

In [4]:

```
data_labels = ['X Train', 'Y Train', 'X Test', 'Y Test']

for x in range(4):
    plt.subplot(3,2,x +1)
    plt.boxplot(data[x])
    plt.title(data_labels[x])


Y_trn = np.mat(Y_trn).A1.astype(int)


Y_tst = np.mat(Y_tst).A1.astype(int)

X_trn = np.mat(X_trn).A
X_tst = np.mat(X_tst).A

X_1a = []
X_2a = []
X_1b = []
X_2b = []
for i in range(len(X_trn)):
    if (Y_trn[i] == 1):
        X_1a.append(X_trn[i][0])
        X_2a.append(X_trn[i][1])
    else:
        X_1b.append(X_trn[i][0])
        X_2b.append(X_trn[i][1])

X_1atst = []
X_2atst = []
X_1btst = []
X_2btst = []
for i in range(len(X_tst)):
    if (Y_tst[i] == 1):
        X_1atst.append(X_tst[i][0])
        X_2atst.append(X_tst[i][1])
    else:
        X_1btst.append(X_tst[i][0])
        X_2btst.append(X_tst[i][1])

plt.subplot(3,2,5)
plt.plot(X_1a, X_2a, 'b.')
plt.plot(X_1b, X_2b, 'r.')
plt.title("X1 vs X2 Train")

plt.subplot(3,2,6)
plt.plot(X_1atst, X_2atst, 'b.')
plt.plot(X_1btst, X_2btst, 'r.')
plt.title("X1 vs X2 Test")

plt.tight_layout(pad=0.4, w_pad=0.5, h_pad=1.0)
```



In [5]:
```
def H(x, w):
    h =  x * w
    h = 1 + math.exp(h)
    h = 1/h
    return h
```

In [6]:
```
def gradientDescent(X, Y, w, alpha, ittr, batch):
   # print(X, Y, w, alpha)
    m = len(Y)

    for i in range(ittr):
        miniX = []
        miniY = []
        for j in range(batch):
            batchIdx = math.floor((sp.rand(1) * m)[0])

            miniX.append(X.A[batchIdx])
            miniY.append(Y.A[batchIdx])

        miniX = np.mat(miniX)
        miniY = np.mat(miniY)

        temp = (miniY.T - H(miniX,w))
        temp = temp * miniX
        temp = temp * alpha

        # - 2 lambda w?

        w = w - temp.T
        #print(H(miniX,w))

    return w
```

In [7]:

```
def logRegress(X_trn, Y_trn, ittr, lrnRate, batch):

    X = np.mat(X_trn)
    Y = np.mat(Y_trn).T
    w = [0,0]

    w = np.mat(w).T

    w = gradientDescent(X, Y, w, lrnRate, ittr, batch)


    return w
```

In [8]:

```
def errorF(Y, Ycomp):
    error = 0
    for i in range(len(Y)):
        if (Y[i] != Ycomp[i]):
            error += 1

    error = error / len(Y)
    return error
```

Part B

In [9]:

```
ittr = 1000
lrnRate = 0.00001
batch = 1

w = logRegress(X_trn, Y_trn, ittr, lrnRate, batch)

Y_sol = []
for i in range(len(X_trn)):
    Y_sol.append(np.round(H(X_trn[i], w)))

Y_sol = np.mat(Y_sol)
print("W: ", w)

error = errorF(Y_trn.T, Y_sol.T)
print("Training error: ", error * 100, "%")

Y_soltst = []
for i in range(len(X_tst)):
    Y_soltst.append(np.round(H(X_tst[i], w)))
Y_soltst = np.mat(Y_soltst)

errorTst = errorF(Y_tst.T, Y_soltst.T)
print("Testing error: ", errorTst * 100, "%")
```

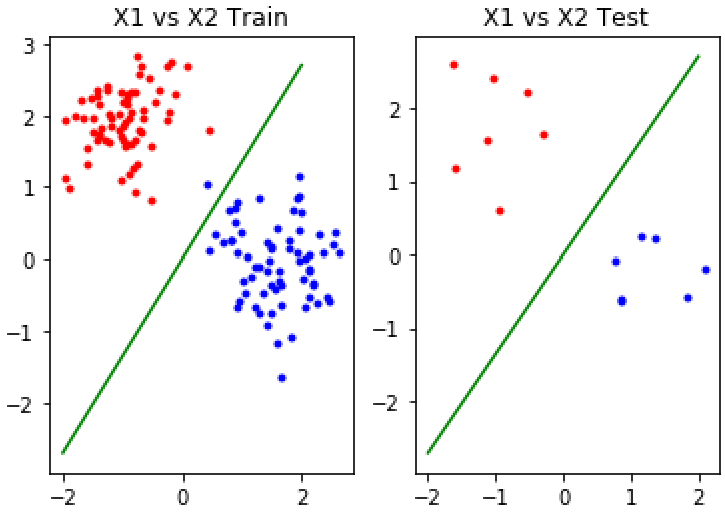
```
W:  [[-0.0064984 ]
 [ 0.00480926]]
Training error:  0.7352941176470588 %
Testing error:  0.0 %
```

In [10]:

```
X1_line = np.mat([-2,-1,0,1,2]).T
X2_line = (X1_line * -1 * w[0]) / w[1]

plt.subplot(1,2,1)
plt.plot(X1_line,X2_line, 'g-')
plt.plot(X_1a, X_2a, 'b.')
plt.plot(X_1b, X_2b, 'r.')
plt.title("X1 vs X2 Train")

plt.subplot(1,2,2)
plt.plot(X1_line,X2_line, 'g-')
plt.plot(X_1atst, X_2atst, 'b.')
plt.plot(X_1btst, X_2btst, 'r.')
plt.title("X1 vs X2 Test")
```

Out[10]:

```
<matplotlib.text.Text at 0x7f2f41cab0f0>
```



Part C

In [11]:

```
mat = scio.loadmat('HW2_Data/data2.mat')

X_trn = mat['X_trn']
Y_trn = mat['Y_trn']
X_tst = mat['X_tst']
Y_tst = mat['Y_tst']
data = [X_trn,Y_trn,X_tst,Y_tst]
```

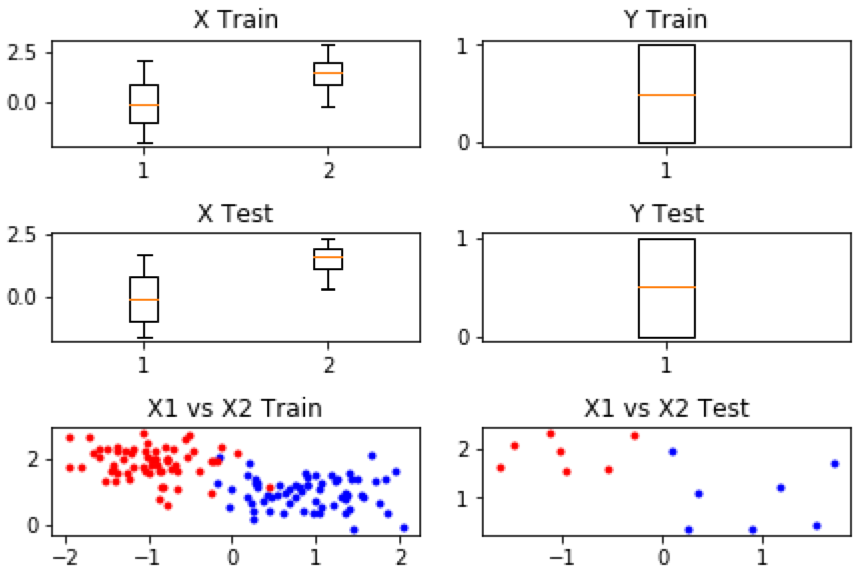
In [12]:

```
print('shape of the X data is [%d, %d]' % X_trn.shape)
print('shape of the Y data is [%d, %d]' % Y_trn.shape)
```

```
shape of the X data is [126, 2]
shape of the Y data is [126, 1]
```

In [13]:

```
data_labels = ['X Train', 'Y Train', 'X Test', 'Y Test']

for x in range(4):
    plt.subplot(3,2,x +1)
    plt.boxplot(data[x])
    plt.title(data_labels[x])


Y_trn = np.mat(Y_trn).A1.astype(int)


Y_tst = np.mat(Y_tst).A1.astype(int)

X_trn = np.mat(X_trn).A
X_tst = np.mat(X_tst).A

X_1a = []
X_2a = []
X_1b = []
X_2b = []
for i in range(len(X_trn)):
    if (Y_trn[i] == 1):
        X_1a.append(X_trn[i][0])
        X_2a.append(X_trn[i][1])
    else:
        X_1b.append(X_trn[i][0])
        X_2b.append(X_trn[i][1])

X_1atst = []
X_2atst = []
X_1btst = []
X_2btst = []
for i in range(len(X_tst)):
    if (Y_tst[i] == 1):
        X_1atst.append(X_tst[i][0])
        X_2atst.append(X_tst[i][1])
    else:
        X_1btst.append(X_tst[i][0])
        X_2btst.append(X_tst[i][1])

plt.subplot(3,2,5)
plt.plot(X_1a, X_2a, 'b.')
plt.plot(X_1b, X_2b, 'r.')
plt.title("X1 vs X2 Train")

plt.subplot(3,2,6)
plt.plot(X_1atst, X_2atst, 'b.')
plt.plot(X_1btst, X_2btst, 'r.')
plt.title("X1 vs X2 Test")

plt.tight_layout(pad=0.4, w_pad=0.5, h_pad=1.0)
```



In [14]:
```
ittr = 1000
lrnRate = 0.00001
batch = 1

w = logRegress(X_trn, Y_trn, ittr, lrnRate, batch)

Y_sol = []
for i in range(len(X_trn)):
    Y_sol.append(np.round(H(X_trn[i], w)))

Y_sol = np.mat(Y_sol)
print("W: ", w)

error = errorF(Y_trn.T, Y_sol.T)
print("Training error: ", error * 100, "%")

Y_soltst = []
for i in range(len(X_tst)):
    Y_soltst.append(np.round(H(X_tst[i], w)))
Y_soltst = np.mat(Y_soltst)

errorTst = errorF(Y_tst.T, Y_soltst.T)
print("Testing error: ", errorTst * 100, "%")
```

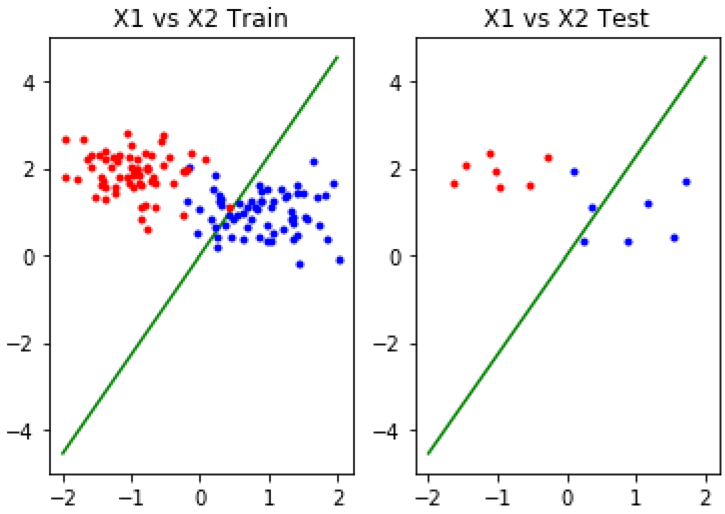
```
W:  [[-0.00463746]
 [ 0.0020435 ]]
Training error:  9.523809523809524 %
Testing error:  14.285714285714285 %
```

In [15]:

```
X1_line = np.mat([-2,-1,0,1,2]).T
X2_line = (X1_line * -1 * w[0]) / w[1]

plt.subplot(1,2,1)
plt.plot(X1_line,X2_line, 'g-')
plt.plot(X_1a, X_2a, 'b.')
plt.plot(X_1b, X_2b, 'r.')
plt.title("X1 vs X2 Train")

plt.subplot(1,2,2)
plt.plot(X1_line,X2_line, 'g-')
plt.plot(X_1atst, X_2atst, 'b.')
plt.plot(X_1btst, X_2btst, 'r.')
plt.title("X1 vs X2 Test")
```

Out[15]:

```
<matplotlib.text.Text at 0x7f2f41e0ef60>
```


X1 vs X2 Train
X1 vs X2 Test