Linear Regression with Weight Decay

[46]:
```
import numpy as np

in_sample_data = []
out_sample_data = []
with open('in_dta.txt', 'r') as f:
    for line in f:
        in_sample_data.append(line)

with open('out_dta.txt', 'r') as f:
    for line in f:
        out_sample_data.append(line)
```

[47]:
```
for i in range(len(in_sample_data)):
    act = in_sample_data[i].split(' ')
    act2 = []
    for val in act:
        if val != '':
            act2.append(float(val))
    in_sample_data[i] = act2
in_sample_data = np.asarray(in_sample_data)
```

[48]:
```
for i in range(len(out_sample_data)):
    act = out_sample_data[i].split(' ')
    act2 = []
    for val in act:
        if val != '':
            act2.append(float(val))
    out_sample_data[i] = act2
out_sample_data = np.asarray(out_sample_data)
```

[49]:
```
def non_linear_transform(X):
    x_transformed = []
    for x in X:
        x1 = x[0]
        x2 = x[1]
        x_transformed.append([1, x1, x2, x1**2, x2**2, x1*x2, abs(x1 - x2),␣
 ↪abs(x1 + x2)])
    return np.asarray(x_transformed)
```

[50]:
```
def linear_regression(X, y):
    X_plus = np.linalg.inv(X.transpose().dot(X)).dot(X.transpose())
    w = X_plus.dot(y)
    return(w)
```

```
[51]: def linear_regression_weight_decay(X, y, l):
          X_prime = np.linalg.inv(X.transpose().dot(X) + l*np.eye(X.shape[1])).dot(X.
       ↪transpose())
          w = X_prime.dot(y)
          return(w)
```

```
[54]: def get_error(X, w, y):
          correct_pos = []
          ct = 0
          for x in X:
              if np.sign(w.dot(x)) == y[ct]:
                  correct_pos.append(ct)
              ct += 1
          err = 1-len(correct_pos)/float(ct)
          return err
```

```
[56]: X = in_sample_data[:,:2]
      y = in_sample_data[:,2]
      Z = non_linear_transform(X)
      w = linear_regression(Z,y)
      in_sample_err = get_error(Z,w,y)
      test_X = out_sample_data[:,:2]
      test_Z = non_linear_transform(test_X)
      test_y = out_sample_data[:,2]
      out_sample_err = get_error(test_Z,w,test_y)
      print('Regular Linear Regression: ('+ str(round(in_sample_err, 2)) + ', ' +␣
       ↪str(round(out_sample_err,2)) + ')')

      for k in [-3, 3, 2, 1, 0, -1, -2]:
          l = 10**(k)
          X = in_sample_data[:,:2]
          X = non_linear_transform(X)
          y = in_sample_data[:,2]
          w = linear_regression_weight_decay(X,y,l)
          in_sample_err = get_error(X,w,y)
          test_X = non_linear_transform(out_sample_data[:,:2])
          test_y = out_sample_data[:,2]
          out_sample_err = get_error(test_X,w,test_y)
          print('Linear Regression with weight decay' + ' k = ' + str(k) + ': ('+␣
       ↪str(round(in_sample_err, 2)) + ', ' + str(round(out_sample_err,2)) + ')')
```

```
Regular Linear Regression: (0.03, 0.08)
Linear Regression with weight decay k = -3: (0.03, 0.08)
Linear Regression with weight decay k = 3: (0.37, 0.44)
Linear Regression with weight decay k = 2: (0.2, 0.23)
Linear Regression with weight decay k = 1: (0.06, 0.12)
Linear Regression with weight decay k = 0: (0.0, 0.09)
Linear Regression with weight decay k = -1, -2: (0.03, 0.06), (0.03, 0.08)
```

[57]:
```
min_err = 100
for k in range(-50,50):
    l = 10**(k)
    w = linear_regression_weight_decay(X,y,l)
    test_X = non_linear_transform(out_sample_data[:,:2])
    test_y = out_sample_data[:,2]
    out_sample_err = get_error(test_X,w,test_y)
    min_err = min(min_err, get_error(test_X,w,test_y))
print(min_err)
```

```
0.05600000000000005
```