HW1_Hardik_2678294168

September 12, 2021

[128]:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from sklearn.linear_model import Ridge
from sklearn.linear_model import LassoCV
from sklearn.model_selection import GridSearchCV
import math
import matplotlib.pyplot as plt
import warnings

warnings.filterwarnings('ignore')
```

[70]:
```
def readingdataset(filename):
    file = "../data/HW1_p2_material/"
    filead=file+filename
    dataset = np.load(filead)
    Xtrain, Ytrain, Xtest, Ytest=␣
 ↪dataset['X_train'],dataset['y_train'],dataset['X_test'],dataset['y_test']
    #Xtrain=np.concatenate((np.ones((len(Xtrain),1)),Xtrain),axis=1)
    #Xtest=np.concatenate((np.ones((len(Xtest),1)),Xtest),axis=1)
    return Xtrain,Ytrain,Xtest,Ytest
```

[119]:
```
Xtrain_d1,Ytrain_d1,Xtest_d1,Ytest_d1=readingdataset("dataset1_dim9_Ntr10.npz")
Xtrain_d2,Ytrain_d2,Xtest_d2,Ytest_d2=readingdataset("dataset2_dim9_Ntr100.npz")
Xtrain_d3,Ytrain_d3,Xtest_d3,Ytest_d3=readingdataset("dataset3_dim9_Ntr1000.
 ↪npz")
Xtrain_d4,Ytrain_d4,Xtest_d4,Ytest_d4=readingdataset("dataset4_dim2_Ntr10.npz")
Xtrain_d5,Ytrain_d5,Xtest_d5,Ytest_d5=readingdataset("dataset5_dim2_Ntr30.npz")
Xtrain_d6,Ytrain_d6,Xtest_d6,Ytest_d6=readingdataset("dataset6_dim2_Ntr100.npz")
Xtrain_d7,Ytrain_d7,Xtest_d7,Ytest_d7=readingdataset("dataset7_dim2_Ntr10.npz")
Xtrain_d8,Ytrain_d8,Xtest_d8,Ytest_d8=readingdataset("dataset8_dim2_Ntr30.npz")
Xtrain_d9,Ytrain_d9,Xtest_d9,Ytest_d9=readingdataset("dataset9_dim2_Ntr100.npz")
```

[72]:
```
reg_coeff=np.logspace(-10, 10, num=11, endpoint=True, base=2.0)
reg_coeff
```

```
[72]: array([9.765625e-04, 3.906250e-03, 1.562500e-02, 6.250000e-02,
              2.500000e-01, 1.000000e+00, 4.000000e+00, 1.600000e+01,
              6.400000e+01, 2.560000e+02, 1.024000e+03])
```

```
[102]: def linear_reg(Xtrain,Ytrain,Xtest,Ytest):
           clf=LinearRegression(fit_intercept=True, normalize=False)
           clf=clf.fit(Xtrain,Ytrain)
           w_p=clf.coef_
           w0=clf.intercept_
           w0=np.array(w0, ndmin=1)
           w= np.concatenate((w0,w_p),axis=0)
           Ytrain_pred=clf.predict(Xtrain)
           Ytest_pred=clf.predict(Xtest)
           mse_train=mean_squared_error(Ytrain, Ytrain_pred)
           mse_test=mean_squared_error(Ytest, Ytest_pred)
           l1_norm=np.linalg.norm(w,ord=1)
           l2_norm=np.linalg.norm(w,ord=2)
           spars=0
           for weight in w:
               if weight==0:
                   spars=spars+1
           print ("Weight vector = {}".format(w))
           print ("MSE_train = {}".format(mse_train))
           print ("MSE_test = {}".format(mse_test))
           print ("L1_norm = {}".format(l1_norm))
           print ("L2_norm = {}".format(l2_norm))
           print ("Sparsity = {}".format(spars))

           return w,mse_train, mse_test, l1_norm, l2_norm, spars
```

```
[106]: def linear_reg_lasso(Xtrain,Ytrain,Xtest,Ytest):
           reg_coeff=np.logspace(-10, 10, num=11, endpoint=True, base=2.0)
           clf=LassoCV( alphas=reg_coeff, fit_intercept=True, normalize=False,␣
        ↪max_iter=1000, cv=5, positive=False )
           clf=clf.fit(Xtrain,Ytrain)
           w_p=clf.coef_
           w0=clf.intercept_
           w0=np.array(w0, ndmin=1)
           w= np.concatenate((w0,w_p),axis=0)
           Ytrain_pred=clf.predict(Xtrain)
           Ytest_pred=clf.predict(Xtest)
           mse_train=mean_squared_error(Ytrain, Ytrain_pred)
           mse_test=mean_squared_error(Ytest, Ytest_pred)
           l1_norm=np.linalg.norm(w,ord=1)
           l2_norm=np.linalg.norm(w,ord=2)
           spars=0
           for weight in w:
```

```
        if weight==0:
            spars=spars+1
    mse_lambda_array=clf.mse_path_
    mse_lambda_mean=np.mean(mse_lambda_array)
    mse_lambda_std=np.std(mse_lambda_array)
    best_alpha=math.log(clf.alpha_,2)
    print ("Best param log_lambda = {}".format(best_alpha))
    print ("Mean of MSE = {}".format(mse_lambda_mean))
    print ("STD of MSE = {}".format(mse_lambda_std))
    print ("Weight vector = {}".format(w))
    print ("MSE_train = {}".format(mse_train))
    print ("MSE_test = {}".format(mse_test))
    print ("L1_norm = {}".format(l1_norm))
    print ("L2_norm = {}".format(l2_norm))
    print ("Sparsity = {}".format(spars))
    return w,mse_train, mse_test, l1_norm, l2_norm, spars
```

[114]:
```
def linear_reg_ridge(Xtrain,Ytrain,Xtest,Ytest):
    model=Ridge(fit_intercept=True, max_iter=1000)
    parameters = {'alpha':np.logspace(-10, 10, num=11, endpoint=True, base=2.0)}
    clf=GridSearchCV(model, parameters, scoring='neg_mean_squared_error',␣
 ↪cv=5,return_train_score=True)
    clf=clf.fit(Xtrain,Ytrain)
    best_model = clf.best_estimator_
    w_p=best_model.coef_
    w0=best_model.intercept_
    w0=np.array(w0, ndmin=1)
    w= np.concatenate((w0,w_p),axis=0)
    Ytrain_pred=best_model.predict(Xtrain)
    Ytest_pred=best_model.predict(Xtest)
    mse_train=mean_squared_error(Ytrain, Ytrain_pred)
    mse_test=mean_squared_error(Ytest, Ytest_pred)
    l1_norm=np.linalg.norm(w,ord=1)
    l2_norm=np.linalg.norm(w,ord=2)
    spars=0
    for weight in w:
        if weight==0:
            spars=spars+1
    mse_lambda_array=clf.cv_results_['mean_test_score']
    mse_lambda_mean=np.mean(mse_lambda_array)
    mse_lambda_std=np.std(mse_lambda_array)
    best_alpha=math.log(clf.best_params_['alpha'],2)
    print ("Best param log_lambda = {}".format(best_alpha))
    print ("Mean of MSE = {}".format(mse_lambda_mean))
    print ("STD of MSE = {}".format(mse_lambda_std))
    print ("Weight vector = {}".format(w))
    print ("MSE_train = {}".format(mse_train))
    print ("MSE_test = {}".format(mse_test))
    print ("L1_norm = {}".format(l1_norm))
    print ("L2_norm = {}".format(l2_norm))
    print ("Sparsity = {}".format(spars))
    return w,mse_train, mse_test, l1_norm, l2_norm, spars
```

[103]:
```
#dataset1-9 feat - no regularizer
w_d1,mse_train_d1, mse_test_d1, l1_norm_d1, l2_norm_d1, spars_d1=␣
 ↪linear_reg(Xtrain_d1,Ytrain_d1,Xtest_d1,Ytest_d1)
```

```
Weight vector = [ -7.01477582   3.20265861  -2.01056618   4.61891474
-8.48679639
   5.34513234  -1.36854253 -20.00142649  13.2641012    3.11232438]
MSE_train = 5.275901734118127e-28
MSE_test = 480.8978021950014
L1_norm = 68.42523867301641
L2_norm = 27.802696240879936
Sparsity = 0
```

[107]:
```
#dataset1-9 feat - lasso regularizer
w_d1_lasso,mse_train_d1_lasso, mse_test_d1_lasso, l1_norm_d1_lasso,␣
 ↪l2_norm_d1_lasso, spars_d1_lasso=␣
 ↪linear_reg_lasso(Xtrain_d1,Ytrain_d1,Xtest_d1,Ytest_d1)
```

```
Best param log_lambda = 2.0
Mean of MSE = 1067.508880995702
STD of MSE = 1492.7569038540025
Weight vector = [ 0.12578696  2.26001059  0.         -3.34237423 -0.
5.01163416
  0.         -5.93509725 -0.          1.43300028]
MSE_train = 14.1078838492251
MSE_test = 233.38359844231735
L1_norm = 18.107903459292224
L2_norm = 8.870754296600296
Sparsity = 4
```

[115]:
```
#dataset1-9 feat - Ridge regularizer
w_d1_ridge,mse_train_d1_ridge, mse_test_d1_ridge, l1_norm_d1_ridge,␣
 ↪l2_norm_d1_ridge, spars_d1_ridge=␣
 ↪linear_reg_ridge(Xtrain_d1,Ytrain_d1,Xtest_d1,Ytest_d1)
```

```
Best param log_lambda = 4.0
Mean of MSE = -840.6303690331886
STD of MSE = 628.9771239660494
Weight vector = [-0.24139412  2.5665958  -0.28155627 -1.71113314 -1.61199141
2.81003838
  2.21325862 -3.03423719 -2.75553818  1.62177076]
MSE_train = 18.11719332610634
MSE_test = 264.8044400260602
```

```
L1_norm = 18.847513865491255
L2_norm = 6.669033494963576
Sparsity = 0
```

[112]:
```
#dataset2-9 feat - no regularizer
w_d2,mse_train_d2, mse_test_d2, l1_norm_d2, l2_norm_d2, spars_d2=␣
 ↪linear_reg(Xtrain_d2,Ytrain_d2,Xtest_d2,Ytest_d2)
```

```
Weight vector = [ 0.43392102  2.397075    0.5682055  -3.87069203  0.8554485
2.25097789
  2.04197312 -6.17726984 -1.80441184  1.25424529]
MSE_train = 86.3366112987716
MSE_test = 112.65154328000659
L1_norm = 21.654220029500788
L2_norm = 8.613675992794825
Sparsity = 0
```

[108]:
```
#dataset2-9 feat - lasso regularizer
w_d2_lasso,mse_train_d2_lasso, mse_test_d2_lasso, l1_norm_d2_lasso,␣
 ↪l2_norm_d2_lasso, spars_d2_lasso=␣
 ↪linear_reg_lasso(Xtrain_d2,Ytrain_d2,Xtest_d2,Ytest_d2)
```

```
Best param log_lambda = 0.0
Mean of MSE = 735.8136444968056
STD of MSE = 1269.9358862344855
Weight vector = [ 0.43208693  2.33887343  0.43071582 -2.94652499  0.
2.36208926
  1.92436118 -6.33525333 -1.61782688  1.14532181]
MSE_train = 87.63581771635758
MSE_test = 110.19640754070862
L1_norm = 19.533053620667054
L2_norm = 8.238431033695141
Sparsity = 1
```

[116]:
```
#dataset2-9 feat - Ridge regularizer
w_d2_ridge,mse_train_d2_ridge, mse_test_d2_ridge, l1_norm_d2_ridge,␣
 ↪l2_norm_d2_ridge, spars_d2_ridge=␣
 ↪linear_reg_ridge(Xtrain_d2,Ytrain_d2,Xtest_d2,Ytest_d2)
```

```
Best param log_lambda = 6.0
Mean of MSE = -121.8610900745945
STD of MSE = 27.571406540496987
Weight vector = [ 0.45904623  2.25533004  0.55844399 -2.57037539 -0.32269209
2.23048119
  2.05571123 -4.14875114 -3.77234633  1.17294188]
MSE_train = 89.14761102319815
MSE_test = 111.42028497489191
L1_norm = 19.54611950615057
L2_norm = 7.371538310088324
```

```
Sparsity = 0
```

[105]:
```
#dataset3-9 feat - no regularizer
w_d3,mse_train_d3, mse_test_d3, l1_norm_d3, l2_norm_d3, spars_d3=␣
 ↪linear_reg(Xtrain_d3,Ytrain_d3,Xtest_d3,Ytest_d3)
```

```
Weight vector = [ 1.71594731  1.90468457  0.41212604 -3.17204863  0.25311452
4.87289258
 -0.25297342 -8.71299177  0.80571383  0.89176542]
MSE_train = 98.21301479826998
MSE_test = 109.12481315987687
L1_norm = 22.994258103832294
L2_norm = 10.864521591717066
Sparsity = 0
```

[109]:
```
#dataset3-9 feat - lasso regularizer
w_d3_lasso,mse_train_d3_lasso, mse_test_d3_lasso, l1_norm_d3_lasso,␣
 ↪l2_norm_d3_lasso, spars_d3_lasso=␣
 ↪linear_reg_lasso(Xtrain_d3,Ytrain_d3,Xtest_d3,Ytest_d3)
```

```
Best param log_lambda = -2.0
Mean of MSE = 674.2737984543094
STD of MSE = 1094.9497290230072
Weight vector = [ 1.70180095  1.88836961  0.37895217 -2.91361253  0.
4.60921066
  0.         -7.90337403 -0.          0.87016587]
MSE_train = 98.45022018747636
MSE_test = 109.07211761162876
L1_norm = 20.265485823458295
L2_norm = 9.977982841156606
Sparsity = 3
```

[118]:
```
#dataset3-9 feat - Ridge regularizer
w_d3_ridge,mse_train_d3_ridge, mse_test_d3_ridge, l1_norm_d3_ridge,␣
 ↪l2_norm_d3_ridge, spars_d3_ridge=␣
 ↪linear_reg_ridge(Xtrain_d3,Ytrain_d3,Xtest_d3,Ytest_d3)
```

```
Best param log_lambda = 2.0
Mean of MSE = -102.13749808847291
STD of MSE = 2.223683320142983
Weight vector = [ 1.71554948  1.90414688  0.41169323 -3.16239173  0.24355474
4.83535943
 -0.21617519 -8.46183322  0.55511326  0.8910176 ]
MSE_train = 98.22296899901596
MSE_test = 108.98663170734264
L1_norm = 22.396834768429468
L2_norm = 10.626877909890354
Sparsity = 0
```

[120]:
```
#dataset4-2 feat - no regularizer
w_d4,mse_train_d4, mse_test_d4, l1_norm_d4, l2_norm_d4, spars_d4=␣
↪linear_reg(Xtrain_d4,Ytrain_d4,Xtest_d4,Ytest_d4)
```

```
Weight vector = [ 6.77265711 -2.4928513   7.23801612]
MSE_train = 95.38019904643618
MSE_test = 163.48761227397387
L1_norm = 16.503524531665292
L2_norm = 10.221157922129628
Sparsity = 0
```

[121]:
```
#dataset4-2 feat - lasso regularizer
w_d4_lasso,mse_train_d4_lasso, mse_test_d4_lasso, l1_norm_d4_lasso,␣
↪l2_norm_d4_lasso, spars_d4_lasso=␣
↪linear_reg_lasso(Xtrain_d4,Ytrain_d4,Xtest_d4,Ytest_d4)
```

```
Best param log_lambda = 2.0
Mean of MSE = 256.25146952709184
STD of MSE = 249.16970913724862
Weight vector = [5.53704387 0.         4.00141645]
MSE_train = 98.92934420027042
MSE_test = 134.48864379047166
L1_norm = 9.538460320163804
L2_norm = 6.831558271694046
Sparsity = 1
```

[122]:
```
#dataset4-2 feat - Ridge regularizer
w_d4_ridge,mse_train_d4_ridge, mse_test_d4_ridge, l1_norm_d4_ridge,␣
↪l2_norm_d4_ridge, spars_d4_ridge=␣
↪linear_reg_ridge(Xtrain_d4,Ytrain_d4,Xtest_d4,Ytest_d4)
```

```
Best param log_lambda = 4.0
Mean of MSE = -196.33770706675338
STD of MSE = 50.48813742731631
Weight vector = [4.61680832 1.55139276 2.16748412]
MSE_train = 102.92593770240387
MSE_test = 127.90734292133521
L1_norm = 8.335685196859789
L2_norm = 5.331015470702115
Sparsity = 0
```

[123]:
```
#dataset5-2 feat - no regularizer
w_d5,mse_train_d5, mse_test_d5, l1_norm_d5, l2_norm_d5, spars_d5=␣
↪linear_reg(Xtrain_d5,Ytrain_d5,Xtest_d5,Ytest_d5)
print(" ")
print("#####")

#dataset5-2 feat - lasso regularizer
```

```
w_d5_lasso,mse_train_d5_lasso, mse_test_d5_lasso, l1_norm_d5_lasso,␣
 ↪l2_norm_d5_lasso, spars_d5_lasso=␣
 ↪linear_reg_lasso(Xtrain_d5,Ytrain_d5,Xtest_d5,Ytest_d5)
print(" ")
print("#####")

#dataset5-2 feat - Ridge regularizer
w_d5_ridge,mse_train_d5_ridge, mse_test_d5_ridge, l1_norm_d5_ridge,␣
 ↪l2_norm_d5_ridge, spars_d5_ridge=␣
 ↪linear_reg_ridge(Xtrain_d5,Ytrain_d5,Xtest_d5,Ytest_d5)
```

```
Weight vector = [ 4.05510307  2.74884213 -0.29784002]
MSE_train = 87.12261767437649
MSE_test = 114.70433167932684
L1_norm = 7.101785217832998
L2_norm = 4.908024311927913
Sparsity = 0
#####
Best param log_lambda = 2.0
Mean of MSE = 139.29912344057894
STD of MSE = 75.98257910583324
Weight vector = [ 3.7532937   2.47956376 -0.        ]
MSE_train = 88.72353370540213
MSE_test = 105.57083378205994
L1_norm = 6.232857452937665
L2_norm = 4.498383042243042
Sparsity = 1
#####
Best param log_lambda = 6.0
Mean of MSE = -115.6039596435588
STD of MSE = 14.631848794699655
Weight vector = [ 3.79650626  2.41954902 -0.18839922]
MSE_train = 88.86631773419617
MSE_test = 106.00172830717882
L1_norm = 6.4044545005619415
L2_norm = 4.505904073798311
Sparsity = 0
```

[124]:
```
#dataset6-2 feat - no regularizer
w_d6,mse_train_d6, mse_test_d6, l1_norm_d6, l2_norm_d6, spars_d6=␣
 ↪linear_reg(Xtrain_d6,Ytrain_d6,Xtest_d6,Ytest_d6)
print(" ")
print("#####")

#dataset6-2 feat - lasso regularizer
```

```
w_d6_lasso,mse_train_d6_lasso, mse_test_d6_lasso, l1_norm_d6_lasso,␣
 ↪l2_norm_d6_lasso, spars_d6_lasso=␣
 ↪linear_reg_lasso(Xtrain_d6,Ytrain_d6,Xtest_d6,Ytest_d6)
print(" ")
print("#####")

#dataset6-2 feat - Ridge regularizer
w_d6_ridge,mse_train_d6_ridge, mse_test_d6_ridge, l1_norm_d6_ridge,␣
 ↪l2_norm_d6_ridge, spars_d6_ridge=␣
 ↪linear_reg_ridge(Xtrain_d6,Ytrain_d6,Xtest_d6,Ytest_d6)
```

```
Weight vector = [1.21077089 2.30240071 0.23152547]
MSE_train = 101.35833777888638
MSE_test = 101.44570933707959
L1_norm = 3.74469706915353
L2_norm = 2.6116315269679835
Sparsity = 0

#####
Best param log_lambda = -10.0
Mean of MSE = 137.055342060517
STD of MSE = 63.836299120708816
Weight vector = [1.2107991  2.30236633 0.23141756]
MSE_train = 101.35833791468285
MSE_test = 101.44595275106839
L1_norm = 3.744582986648264
L2_norm = 2.6116047303958663
Sparsity = 0

#####
Best param log_lambda = 6.0
Mean of MSE = -112.14403331581853
STD of MSE = 4.0246773291225395
Weight vector = [1.22012634 2.2180724  0.24090883]
MSE_train = 101.47660067151448
MSE_test = 101.30261474052405
L1_norm = 3.6791075627301373
L2_norm = 2.542949176508624
Sparsity = 0
```

[125]:
```
#dataset7-2 feat - no regularizer
w_d7,mse_train_d7, mse_test_d7, l1_norm_d7, l2_norm_d7, spars_d7=␣
 ↪linear_reg(Xtrain_d7,Ytrain_d7,Xtest_d7,Ytest_d7)
print(" ")
print("#####")

#dataset7-2 feat - lasso regularizer
```

```
w_d7_lasso,mse_train_d7_lasso, mse_test_d7_lasso, l1_norm_d7_lasso,␣
↪l2_norm_d7_lasso, spars_d7_lasso=␣
↪linear_reg_lasso(Xtrain_d7,Ytrain_d7,Xtest_d7,Ytest_d7)
print(" ")
print("#####")

#dataset7-2 feat - Ridge regularizer
w_d7_ridge,mse_train_d7_ridge, mse_test_d7_ridge, l1_norm_d7_ridge,␣
↪l2_norm_d7_ridge, spars_d7_ridge=␣
↪linear_reg_ridge(Xtrain_d7,Ytrain_d7,Xtest_d7,Ytest_d7)
```

```
Weight vector = [ 1.6193184   4.35846137 -2.05316003]
MSE_train = 25.417551693358597
MSE_test = 116.51141337592615
L1_norm = 8.030939797861016
L2_norm = 5.082700433707281
Sparsity = 0

#####
Best param log_lambda = 0.0
Mean of MSE = 80.56417433998845
STD of MSE = 88.39221271457387
Weight vector = [ 1.49398093  3.84541442 -1.4616377 ]
MSE_train = 26.522508540648026
MSE_test = 108.51285152596137
L1_norm = 6.801033058092814
L2_norm = 4.37670833943422
Sparsity = 0

#####
Best param log_lambda = 2.0
Mean of MSE = -70.89051049847157
STD of MSE = 13.911391429017938
Weight vector = [ 1.59580358  3.78666035 -1.49465134]
MSE_train = 26.617548498116584
MSE_test = 109.29761312532439
L1_norm = 6.877115268463562
L2_norm = 4.372569988665177
Sparsity = 0
```

[126]:
```
#dataset8-2 feat - no regularizer
w_d8,mse_train_d8, mse_test_d8, l1_norm_d8, l2_norm_d8, spars_d8=␣
↪linear_reg(Xtrain_d8,Ytrain_d8,Xtest_d8,Ytest_d8)
print(" ")
print("#####")

#dataset8-2 feat - lasso regularizer
```

```
w_d8_lasso,mse_train_d8_lasso, mse_test_d8_lasso, l1_norm_d8_lasso,␣
 ↪l2_norm_d8_lasso, spars_d8_lasso=␣
 ↪linear_reg_lasso(Xtrain_d8,Ytrain_d8,Xtest_d8,Ytest_d8)
print(" ")
print("#####")

#dataset8-2 feat - Ridge regularizer
w_d8_ridge,mse_train_d8_ridge, mse_test_d8_ridge, l1_norm_d8_ridge,␣
 ↪l2_norm_d8_ridge, spars_d8_ridge=␣
 ↪linear_reg_ridge(Xtrain_d8,Ytrain_d8,Xtest_d8,Ytest_d8)
```

```
Weight vector = [3.58068323 1.91863829 0.60434473]
MSE_train = 95.15432277075584
MSE_test = 109.24257017878496
L1_norm = 6.10366625566634
L2_norm = 4.107030295969647
Sparsity = 0

#####
Best param log_lambda = 0.0
Mean of MSE = 155.94480304519337
STD of MSE = 88.25024220759911
Weight vector = [3.51599517 1.88223308 0.593483  ]
MSE_train = 95.20146796600213
MSE_test = 109.46763164804094
L1_norm = 5.991711258559851
L2_norm = 4.032027469140141
Sparsity = 0

#####
Best param log_lambda = 6.0
Mean of MSE = -115.91126727230889
STD of MSE = 10.698199618203887
Weight vector = [3.20005109 1.41174703 0.97725369]
MSE_train = 95.90349733909382
MSE_test = 110.51198296942115
L1_norm = 5.589051809257596
L2_norm = 3.6315811189650824
Sparsity = 0
```

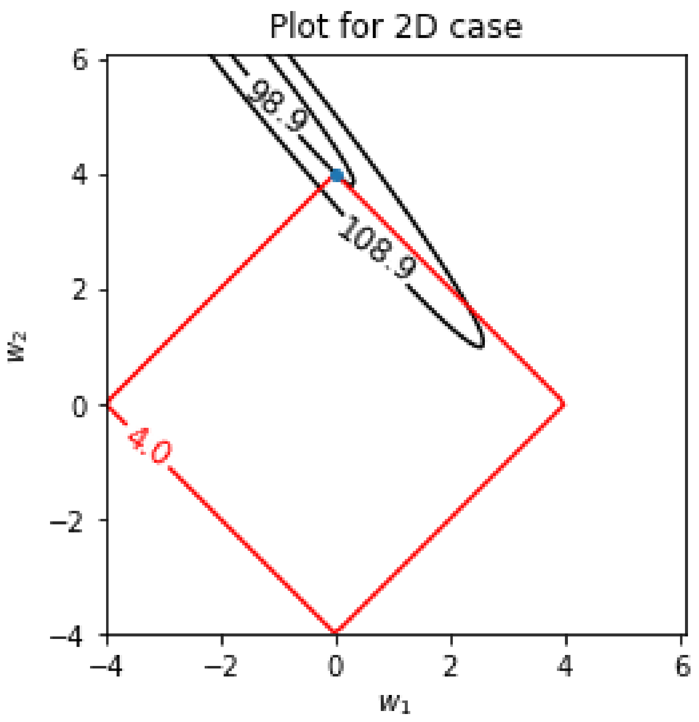
[127]:
```
#dataset9-2 feat - no regularizer
w_d9,mse_train_d9, mse_test_d9, l1_norm_d9, l2_norm_d9, spars_d9=␣
 ↪linear_reg(Xtrain_d9,Ytrain_d9,Xtest_d9,Ytest_d9)
print(" ")
print("#####")

#dataset9-2 feat - lasso regularizer
```

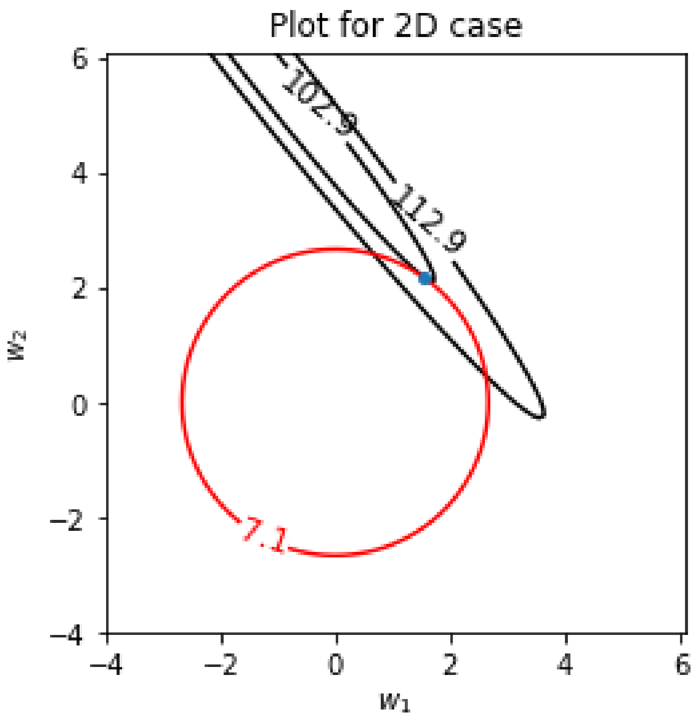
```
w_d9_lasso,mse_train_d9_lasso, mse_test_d9_lasso, l1_norm_d9_lasso,␣
 ↪l2_norm_d9_lasso, spars_d9_lasso=␣
 ↪linear_reg_lasso(Xtrain_d9,Ytrain_d9,Xtest_d9,Ytest_d9)
print(" ")
print("#####")

#dataset9-2 feat - Ridge regularizer
w_d9_ridge,mse_train_d9_ridge, mse_test_d9_ridge, l1_norm_d9_ridge,␣
 ↪l2_norm_d9_ridge, spars_d9_ridge=␣
 ↪linear_reg_ridge(Xtrain_d9,Ytrain_d9,Xtest_d9,Ytest_d9)
```

```
Weight vector = [ 4.11404128  3.04009919 -0.51630424]
MSE_train = 83.3240152571577
MSE_test = 111.41165530589785
L1_norm = 7.6704447096754755
L2_norm = 5.141411166840956
Sparsity = 0

#####
Best param log_lambda = 0.0
Mean of MSE = 123.04568773619322
STD of MSE = 57.15983617651494
Weight vector = [ 4.10151715  2.50447238 -0.        ]
MSE_train = 83.90836860287891
MSE_test = 109.50398425369923
L1_norm = 6.605989527505749
L2_norm = 4.805707525211649
Sparsity = 1

#####
Best param log_lambda = 4.0
Mean of MSE = -90.84543283804109
STD of MSE = 3.3415442744863117
Weight vector = [ 4.10848705  2.79878652 -0.28354144]
MSE_train = 83.44263614295154
MSE_test = 110.36009501840029
L1_norm = 7.19081501224555
L2_norm = 4.979283841239782
Sparsity = 0
```

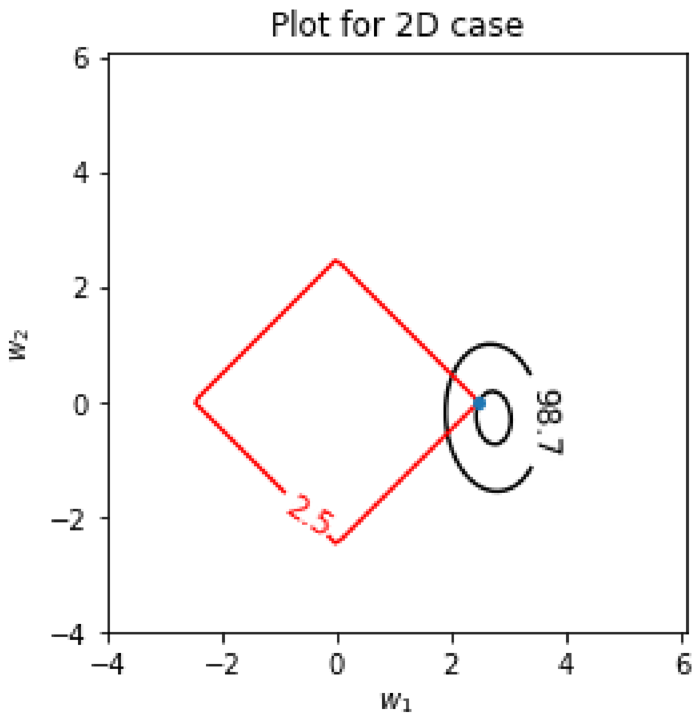
[131]:
```
def MSE(X,Y,w):
    return np.mean(np.square(np.dot(X,w)-Y))

def display(w,Xtest,Ytest,norm,levels=None,w1_range=(-4.0, 6.1, 100),␣
 ↪w2_range=(-4.0, 6.1, 100)):
    w = np.array(w)
    Xtest=np.concatenate((np.ones((len(Xtest),1)),Xtest),axis=1)
    w1list = np.linspace(w1_range[0], w1_range[1], w1_range[2])
    w2list = np.linspace(w2_range[0], w2_range[1], w2_range[2])
    W1, W2 = np.meshgrid(w1list, w2list)

    Z = np.stack((w[0]*np.ones(W1.shape),W1,W2),axis=0)
    Z = Z.reshape((Z.shape[0],-1))
    Z = np.matmul(Xtest,Z) - Ytest.reshape((len(Ytest),1))
    Z = np.square(Z)
    Z = np.sum(Z, axis=0, keepdims=False)/Xtest.shape[0]
    Z = Z.reshape(W1.shape)
    if norm == 'l2':
        W_norm = np.square(W1) + np.square(W2)
    elif norm == 'l1':
        W_norm = np.abs(W1) + np.abs(W2)
    else:
        raise RuntimeError('Unimplemented norm. Please enter "l1" or "l2".')
    plt.figure()

    mse_ori = MSE(Xtest,Ytest,w)
    levels = [mse_ori, mse_ori+10]
    contour = plt.contour(W1, W2, Z, levels, colors='k')
    plt.clabel(contour, colors = 'k', fmt = '%2.1f', fontsize=12)

    if norm == 'l2':
        levels = [np.sum(np.square(w[1:]))]
    elif norm == 'l1':
        levels = [np.sum(abs(w[1:]))]
    else:
        raise RuntimeError('Unimplemented norm. Please enter "l1" or "l2".')
    contour = plt.contour(W1, W2, W_norm, levels, colors='r')
    plt.clabel(contour, colors = 'r', fmt = '%2.1f', fontsize=12)
    plt.plot(w[1],w[2],marker = ".",markersize=8)

    plt.title('Plot for 2D case')
    plt.xlabel('$w_1$')
    plt.ylabel('$w_2$')
    plt.axis('square')
    return
```

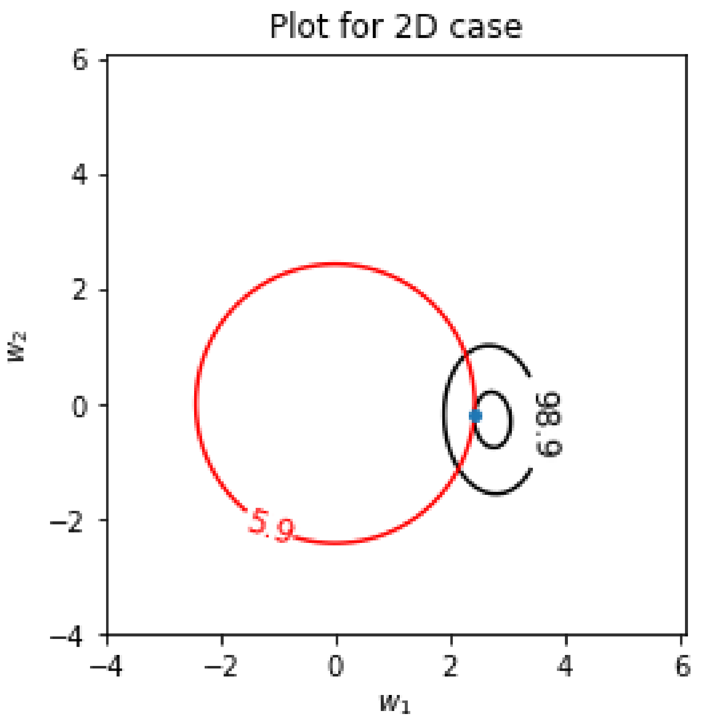
[132]:
```
# D4-l1 results
display(w_d4_lasso,Xtrain_d4,Ytrain_d4,norm='l1')

# D4-l2 results
display(w_d4_ridge,Xtrain_d4,Ytrain_d4,norm='l2')
plt.show()
```


Plot for 2D case
98.9
108.9
4.0
w_1
w_2

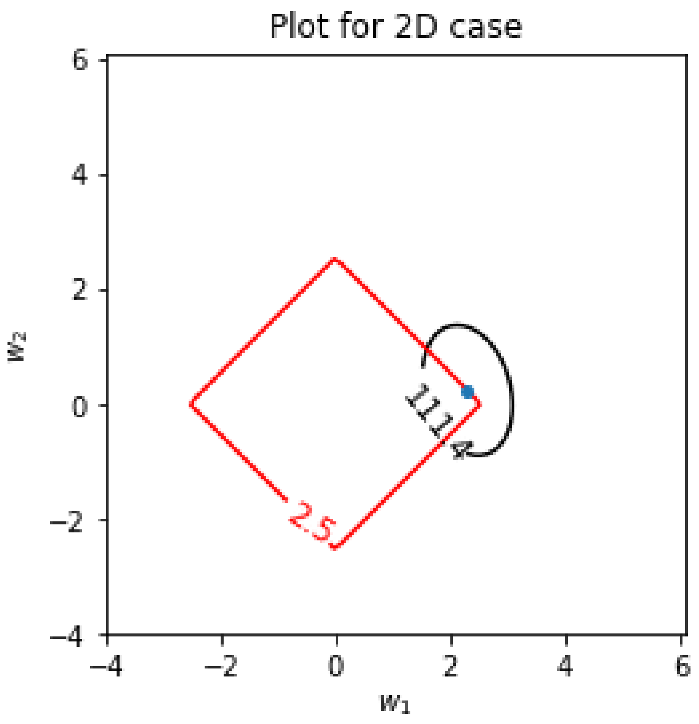

Plot for 2D case
102.9
112.9
7.1
w_1
w_2

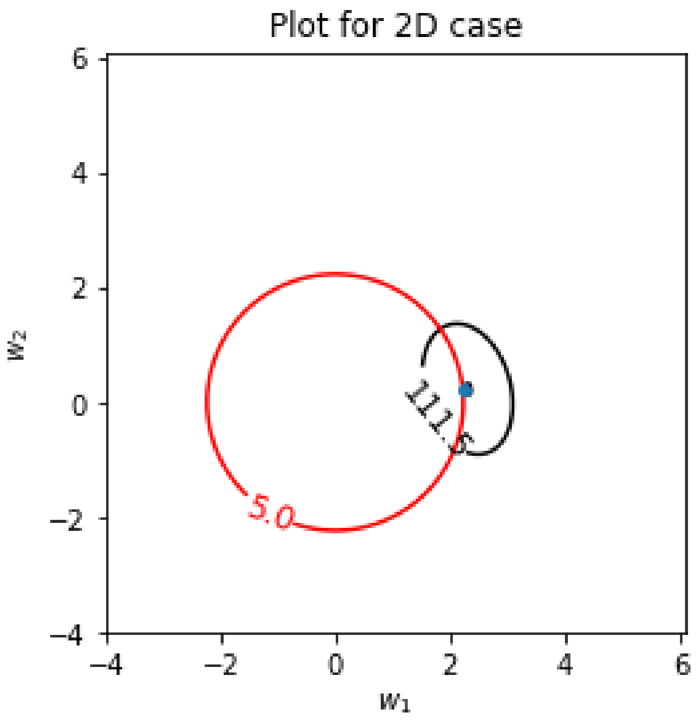
[133]:
```
# D4-l1 results
display(w_d5_lasso,Xtrain_d5,Ytrain_d5,norm='l1')

# D4-l2 results
display(w_d5_ridge,Xtrain_d5,Ytrain_d5,norm='l2')
plt.show()
```



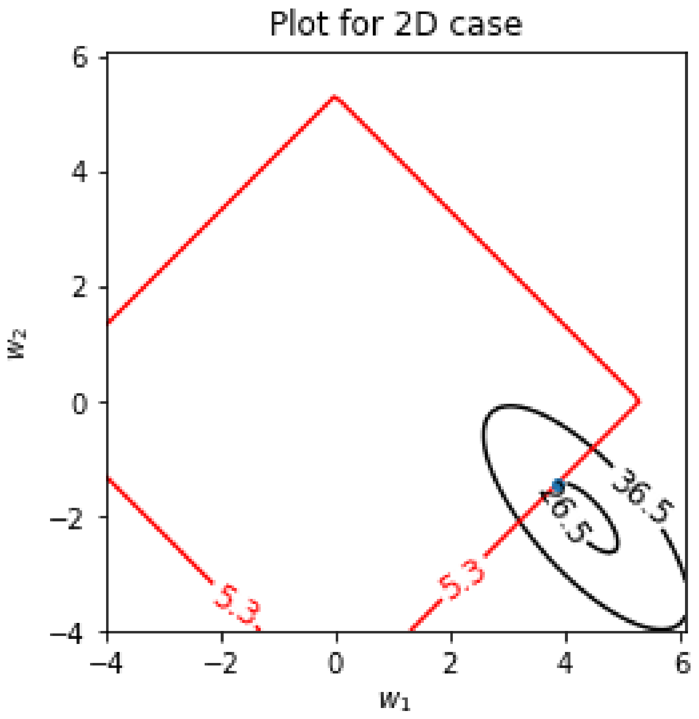


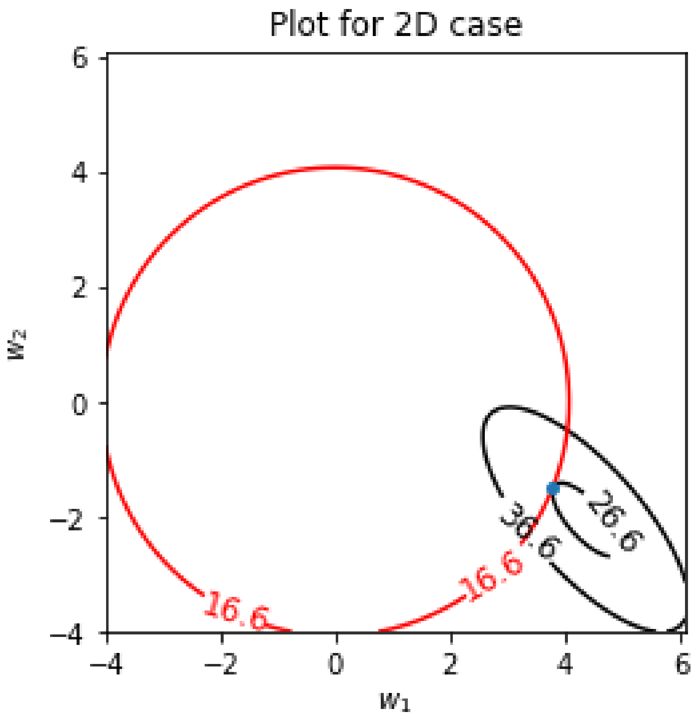
```
[134]: # D6-l1 results
       display(w_d6_lasso,Xtrain_d6,Ytrain_d6,norm='l1')

       # D6-l2 results
       display(w_d6_ridge,Xtrain_d6,Ytrain_d6,norm='l2')
       plt.show()
```


Plot for 2D case
6
4
2
0
−2
−4
−4
−2
0
2
4
6
w_1
w_2
2.5
111.4

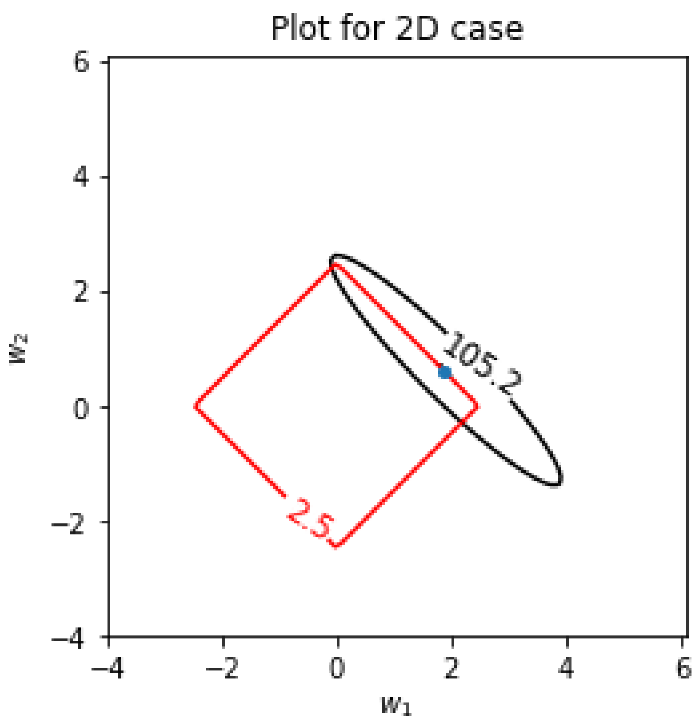

Plot for 2D case
6
4
2
0
−2
−4
−4
−2
0
2
4
6
w_1
w_2
5.0
111.5

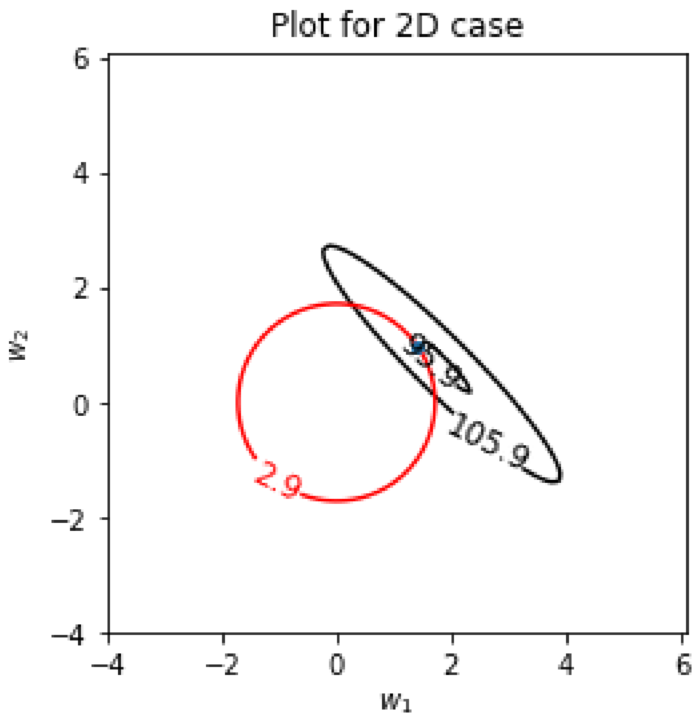
[135]:
```
# D7-l1 results
display(w_d7_lasso,Xtrain_d7,Ytrain_d7,norm='l1')

# D7-l2 results
display(w_d7_ridge,Xtrain_d7,Ytrain_d7,norm='l2')
plt.show()
```



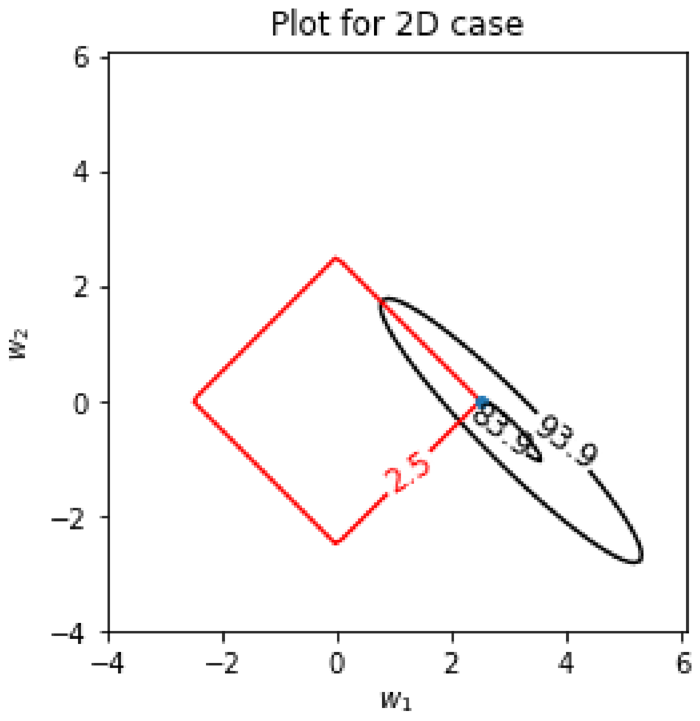


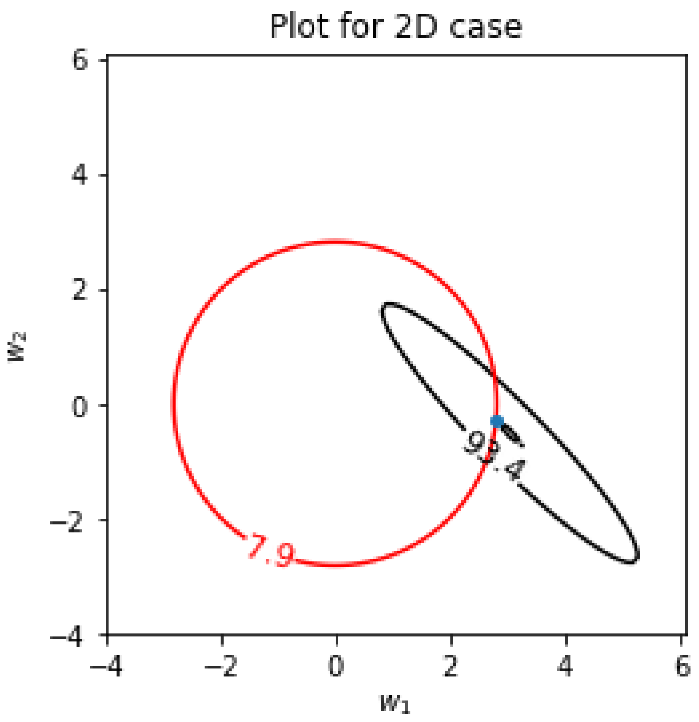
[136]:
```
# D8-l1 results
display(w_d8_lasso,Xtrain_d8,Ytrain_d8,norm='l1')

# D8-l2 results
display(w_d8_ridge,Xtrain_d8,Ytrain_d8,norm='l2')
plt.show()
```


Plot for 2D case
105.2
2.5
w_1
w_2
-4
-2
0
2
4
6


Plot for 2D case
95.9
105.9
2.9
w_1
w_2
-4
-2
0
2
4
6

[137]:
```
# D9-l1 results
display(w_d9_lasso,Xtrain_d9,Ytrain_d9,norm='l1')

# D9-l2 results
display(w_d9_ridge,Xtrain_d9,Ytrain_d9,norm='l2')
plt.show()
```




Plot for 2D case
w_1
w_2
7.9
93.4