svm

September 9, 2018

1 Importing the libraries

```
In [1]: import numpy as np
        import cvxpy as cp
```

2 Loading the dataset

```
In [2]: x = np.genfromtxt('Xsvm.csv' , delimiter = ",")
        y = np.genfromtxt('ysvm.csv' , delimiter = ",")
        print(x.shape,y.shape)
```

```
(500, 2) (500,)
```

```
In [3]: z = np.append(y.reshape(500,1),y.reshape(500,1),axis = 1)
        print(z.shape)
        xy = x*z ##for sigma_over_n(y*x_bar)
        # print(x[1:10],y[1:10],xy[1:10])
```

```
(500, 2)
```

3 Finding the optimal alphas

```
In [4]: alpha = cp.Variable(500)
        a = cp.vstack([alpha , alpha])
        #equation to be maximized ()cost
        objective = cp.Maximize(sum(alpha) - 0.5 * cp.sum_squares(cp.sum(cp.multiply(a.T , xy) ,
        constraints = [ 0 <= alpha , np.transpose(y)*alpha == 0]
        problem = cp.Problem(objective,constraints)
        result = problem.solve()  #maximizing the equation
        print(result) #optimal cost function
        print(alpha.value) #values of alphas
```

```
69.43702993663008
[ 1.38626419e-17  5.24868059e-17  5.26283784e-17  5.78074802e-17
  1.61518808e-17  2.55653038e-17  5.65591735e-17  1.48276379e-17
```

```
5.51148152e-17  3.49510018e-17  2.73489998e-17  2.36342925e-17
2.21433752e-17  5.41669588e-17  3.17332423e-17  4.42179939e-17
5.89161152e-17  2.70534277e-17  4.57736605e-17  2.59124319e-17
4.49198513e-17  4.09828421e-17  3.43082225e-17  3.11899435e-17
5.24993073e-17  3.43353275e-17  5.16031307e-17  5.70039545e-17
2.83925444e-17  4.92502948e-17  2.61272565e-17  2.13214521e-17
5.30666900e-17  3.53511117e+00  1.67121931e-17  2.47977366e-17
4.76503906e-17  4.74302145e-17  5.14331684e-17  3.96208486e-17
4.75067954e-17  4.37054008e-18  3.91125934e-17  5.00833641e-17
2.81030686e-17  4.81433739e-17  1.96172831e-17  3.55957126e-17
5.61383300e-17  5.56787940e-17  3.24219194e-17  9.11161835e-18
2.43233981e-17  1.51517254e-17  4.78068832e-17  6.83216775e-18
4.96590503e-17  1.95291682e-17  5.06952734e-17  2.41641559e-17
9.35632594e-18  2.30494606e-17  4.17486298e-18  5.47945067e-17
2.21249413e-17  5.61615740e-17  4.87975681e-18  5.56144173e-17
4.84912032e-17  1.46367293e-17  5.68148850e-17  3.45643066e-17
2.67482703e-17  3.62317931e-17  4.99532190e-17  4.83492815e-17
1.71913807e-17  2.37381900e-17  5.12216474e-17  5.48159123e-17
4.45342741e-17  2.80672837e-17  4.21799993e-17  4.99144254e-17
6.07248295e-17  2.61055554e-17  5.58712798e-17  5.79659762e-17
5.97947354e-17  5.16212320e-17  5.44531898e-17  5.00044398e-17
5.42581598e-17  4.87480075e-17  5.56149298e-17  2.82333022e-17
5.58284756e-17  3.57539267e-17  6.40653370e-18  3.37864502e-17
3.98241010e-17  5.31144128e-17  2.25345724e-17  1.09555241e-17
1.57681849e-17  1.55718537e-17  2.01390554e-17  3.00662815e-17
4.51794217e-17  5.28014408e-17  5.31987468e-17  4.47165634e-17
3.16044933e-17  1.66364273e-17  4.17417836e-17  2.49749762e-17
4.33342056e-18  2.36620766e-17  5.23357443e-17  4.30953897e-17
1.25818274e-17  5.07315779e-17  5.10669982e-17  2.73490241e-17
5.54005302e-17  1.23480463e-17  8.30774831e-18  2.99104274e-17
1.25920090e-18  5.79768632e-17  3.19399184e-17  2.44318183e-17
5.29633287e-18  5.06586064e-17  3.24140402e-17  1.20592586e-17
5.04607565e-17  5.09270359e-17  5.37108950e-17  5.49681225e-17
5.11571941e-17  1.47362256e-17  4.19216520e-17  3.45174095e-17
3.45115104e-17  1.50320758e-17  5.49335632e-17  4.97387227e-17
1.08393722e-17  4.16793540e-17  4.18389542e-17  4.88510923e-17
5.12794142e-17  6.11577273e-17  1.43250212e-17  8.51568004e-18
4.95312326e-17  1.63513965e-17  5.43210798e-17  5.84669128e-17
4.75089980e-17  5.12160812e-17  4.37283665e-17  4.46570770e-17
4.97896068e-17  2.91379334e-17  5.18924465e-17  3.49858489e-17
4.06237002e-17  2.99102762e-17  4.00612703e-17  5.14314770e-17
5.24444587e-17  4.35562915e-17  5.08831687e-17  5.52830194e-17
5.33122666e-17  5.16138371e-17  3.25630800e-17  3.03576608e-17
1.16992191e-17  5.37443076e-17  5.38281943e-17  5.25656011e-17
4.16130346e-17  5.55000891e-17  2.94191483e-17  5.62265290e-17
5.75247595e-17  2.72168627e-17  2.77022404e-17  5.36373610e-17
2.17145366e-17  3.73019850e-17  1.93868901e-17  4.10641573e-17
3.82621067e-17  5.76179484e-17  4.29743998e-17  2.22014298e-17
```

```
4.74572654e-17  5.76679496e-17  3.05925430e-17  5.43660857e-17
2.87449101e-17  6.51961260e-18  4.53180356e-17  1.26562983e-17
4.84126364e-17  4.99550726e-17  2.46030957e-17 -9.48171912e-19
2.85441311e-17  4.70915993e-17  5.21393056e-17  5.03324785e-17
2.97746855e-17  3.18823201e-17  1.00346714e-17  1.97599873e-17
3.68424633e-17  5.23521457e-17  1.01687664e-17  3.23973162e-17
5.30047606e-17  5.17068653e-17  2.02406993e-17  5.55820127e-17
5.27905754e-17  5.41384964e-17  5.18601414e-17  1.07640947e-17
5.37038454e-17  3.50807165e-17  5.58263217e-17  1.50365289e-17
1.28546892e-17  5.51689527e-17  4.83762161e-17  2.50505487e-17
5.19682463e-17  5.10265245e-17  4.38072242e-18  5.07359532e-17
1.16230854e-17  2.46667663e-17  5.29426282e-17  3.95144018e-17
6.12372922e-18  3.68069018e-17  4.95258065e-17  5.22334490e-17
2.56854271e-17  5.83323340e-17  6.62295061e-18  1.16162098e-17
5.21585815e-17  5.55574347e-17  4.83893697e-17  5.12737535e-17
6.10287238e-18  6.14797983e-17  3.18552151e-17  4.65108791e-17
3.87636269e-18  2.93920433e-17  5.02341915e-17  4.99708921e-17
5.18807410e-17  4.28530909e-17  1.67407592e-17  5.22413130e-17
3.27496819e-17  4.35530810e-17  1.86787706e-17  5.54814912e-17
5.48964747e-17  5.05268434e-17  4.94457993e-17  5.83735178e-17
4.72358410e-17  6.59019188e+01  4.09020137e-17  5.28002776e-17
5.42310878e-17  4.78160358e-17  5.30293914e-17  1.57789439e-17
8.22061472e-18  2.04957075e-18  2.14650699e-17  4.07931067e-17
4.37475577e-17  4.59871986e-17  3.01408204e-17  9.43162938e-18
3.47351271e-17  5.93177883e-17  5.78921895e-17  3.51552554e-17
5.05676335e-17  4.30928447e-17  3.00086492e-17  4.79440623e-17
1.49844192e-17  2.02949094e-17  4.42583915e-17  6.95837566e-18
5.67217260e-17  9.80451095e-19  5.08411757e-17  4.83155273e-17
5.29140582e-17  4.61190906e-17  1.60631328e-17  6.11366483e-17
6.65632002e-18  2.59801946e-17  1.99455710e-17  4.84931673e-17
5.20274734e-17  3.96836790e-17  3.66928887e-17  5.11722512e-17
5.36393520e-17  5.05456174e-17  5.19713290e-17  5.12517074e-17
1.67136541e-17  5.11221787e-17  4.66003646e-17  5.07794211e-17
4.70204930e-17  5.67701357e-17  5.45693944e-17  4.78804746e-17
5.59002706e-17  8.45723167e-18  4.37579353e-17  5.70751441e-17
8.59907848e-18  3.25906941e-17  1.64019460e-17  2.90798438e-17
2.28177005e-17  3.69145182e-17  5.47237725e-17  4.41541335e-17
3.62411534e-17  1.25607398e-17  1.96174775e-17  2.41506034e-17
1.20792854e-17  1.21786397e-17  2.77623519e-17  4.31925213e-17
3.44437478e-17  5.85564380e-17  4.29016074e-18  4.02856025e-17
5.20871350e-17  4.85514311e-17  5.27259330e-17  3.69170840e-17
5.77346532e-17  4.40677095e-17  2.86215219e-17  5.70096905e-17
3.17733139e-17  6.01744955e-17  5.63937873e-18  3.85957493e-17
3.48028897e-17  5.37770929e-17  4.75954487e-17  2.11091351e-17
5.76983306e-17  5.26453607e-17  5.78376755e-17  1.48891451e-17
5.02294744e-17  7.79722670e-18  4.96929495e-17  2.16613581e-17
5.29587173e-17  2.30867300e-17  2.39250435e-17  5.24719368e-17
1.05632876e-17  5.24794221e-17  5.51228264e-17  3.74273385e-17
```

```
  5.18840585e-17  5.69388523e-17  1.16585615e-17  4.39610810e-17
  1.40615103e-17  2.37147486e-17  2.32222553e-17  3.80961538e-17
  1.70766383e-17  4.62978202e-17  2.41043837e-18  5.95576820e-17
  7.98613470e-18  5.44863468e-17  1.23023065e-17  4.68235128e-17
  6.12127290e-17  4.74509074e-17  4.17966126e-17  4.41845219e-17
  5.04490918e-17  5.17955104e-17  2.50436481e-17  3.71135956e-17
  2.56617102e-17  4.76313339e-17  5.51782621e-17  2.32798535e-17
  2.97827047e-17  1.43464910e-17  4.71492420e-17  4.94406451e-17
  5.19876466e-17  1.10419215e-17  4.69349608e-17  9.20047187e-18
  9.22927099e-18  3.16416979e-17  4.21077019e-17  4.52838149e-17
  5.12483197e-17  4.91886796e-17  1.59856339e-17  2.51873717e-17
  5.33578991e-17  4.84531378e-17  3.93801634e-17  4.71153690e-17
  5.59742930e-17  3.43304803e-17  4.98613049e-17  4.47301159e-17
  5.65444460e-17  2.34239960e-17  3.42503582e-17  3.64536705e-17
  4.96693649e-17  5.63318192e-17  1.76471951e-17  1.01543094e-17
  4.86025249e-17  1.61876506e-17  3.95021305e-17  1.07614379e-17
  5.14056511e-17  2.68833207e-17  3.32984619e-17  4.99946339e-17
  4.46083382e-17  5.40403997e-17  3.23015076e-17  5.10311979e-17
  5.26866871e-17  4.43005064e-17  1.76901808e-17  4.92068481e-17
  5.20681060e-17  6.94370299e+01  5.37605526e-17  4.89475577e-17
  3.85162822e-17  2.17822993e-17  3.72207316e-17  3.92774568e-17
  5.02789152e-17  8.37715585e-18  2.92802349e-17  1.26504459e-17
  4.79777818e-17  9.81831195e-18  7.36495148e-18  4.65800358e-17
  4.87397391e-17  4.97862950e-17  3.87670039e-17  1.00170116e-17
  3.03530233e-17  2.99518303e-17  5.98009874e-17  4.76899455e-17
  3.76399956e-18  5.99613419e-17  9.64377295e-18  3.39694093e-17
  2.95038163e-17  1.64019460e-17  5.60905218e-18  2.47740196e-17]
```

4 Finding the values of W

```
In [5]: v = np.append(alpha.value.reshape(500,1),alpha.value.reshape(500,1),axis = 1)
        print(v.shape)
        w = np.sum(v * xy , axis = 0)
        print(w)
```

```
(500, 2)
[-0.11844048 11.78388865]
```

5 Finding W

```
In [6]: for i in range(0,500):
            if alpha.value[i] > 0.1:
                w0 = y[i] - np.matmul(w,np.transpose(x[i]))
                print(w0)
```

```
0.10229992556221768
0.10229992556221679
0.10229992556222078
```

6 Testing the Model

```
In [7]: x_test = np.array([[2,0.5],[0.8,0.7],[1.58,1.33], [0.008,0.001]])
        # print(x_test.shape)
        print("Predictions:")
        for i in range(0,4):
            y_test = w0 + np.matmul(w , np.transpose(x_test[i]))
            if(y_test >= 0):
                print("1")
            else:
                print("0")
```

```
Predictions:
1
1
1
1
```

7 Visualization

```
In [9]: import matplotlib.pyplot as plt

        x_train_1 = []
        x_train_0 = []
        for i in range(0,500):
            if y[i] == 1:
                x_train_1.append(x[i])
            else:
                x_train_0.append(x[i])
        x_train_1 = np.asarray(x_train_1)
        x_train_0 = np.asarray(x_train_0)
        print(x_train_1.shape)
        print(x_train_0.shape)

        plt.plot(x_train_1[:,0],x_train_1[:,1],'o',label = 'label = 1')
        plt.plot(x_train_0[:,0],x_train_0[:,1],'o', label = 'Label = -1')
        plt.plot(x_test[:,0],x_test[:,1],'ro',label = 'Test Sample')
        plt.legend()
        plt.show()
```

```
(228, 2)
```

```
(272, 2)
```