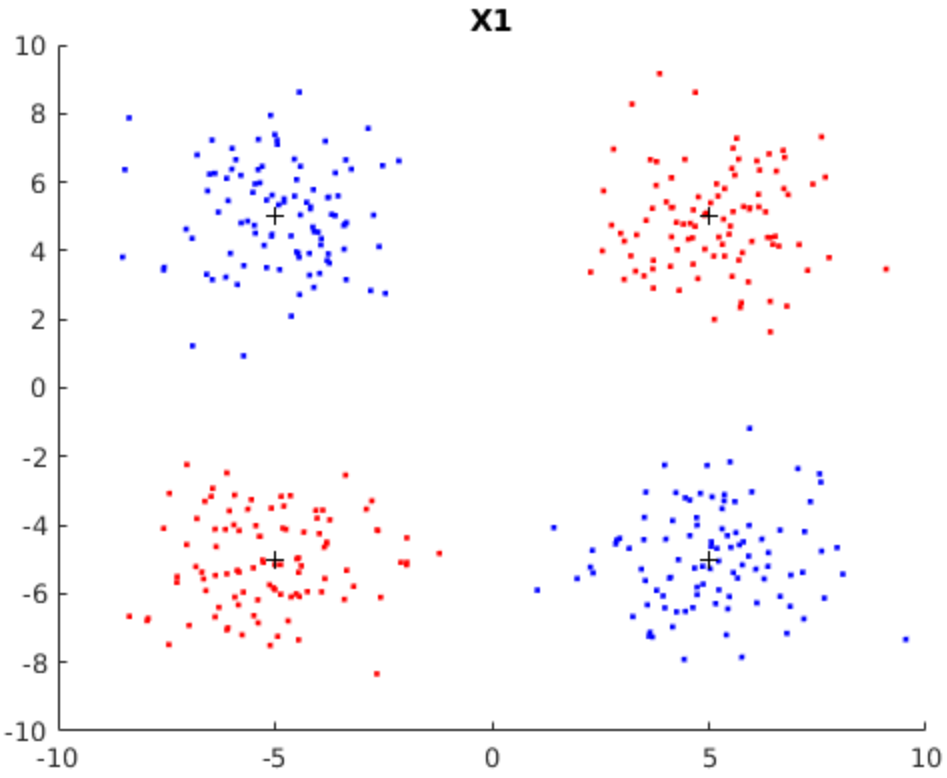
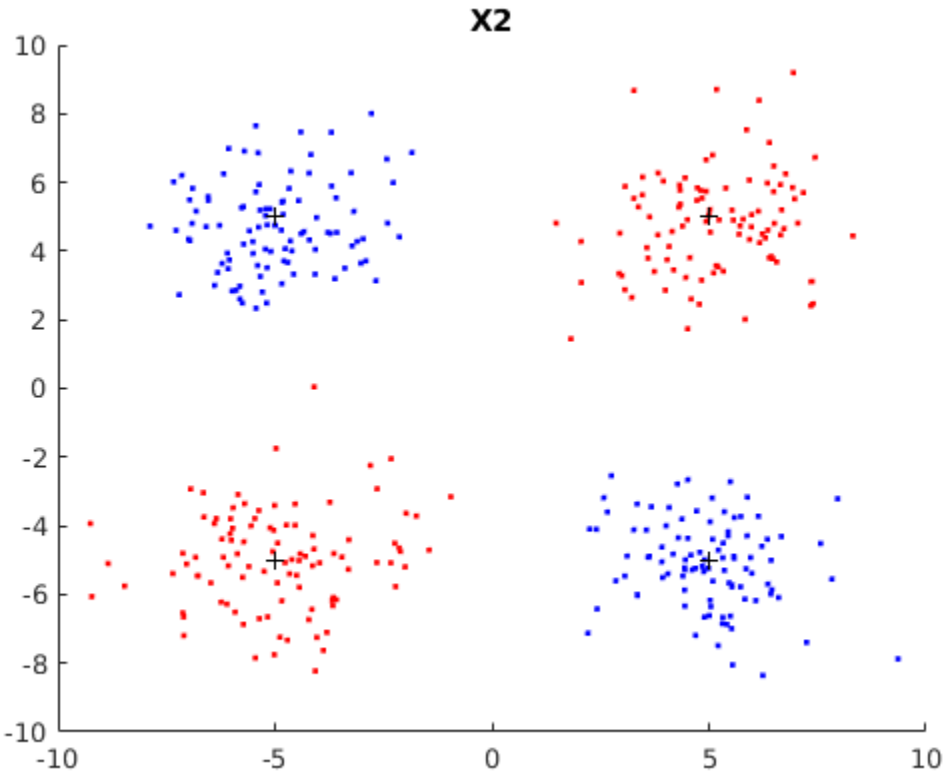
Table of Contents

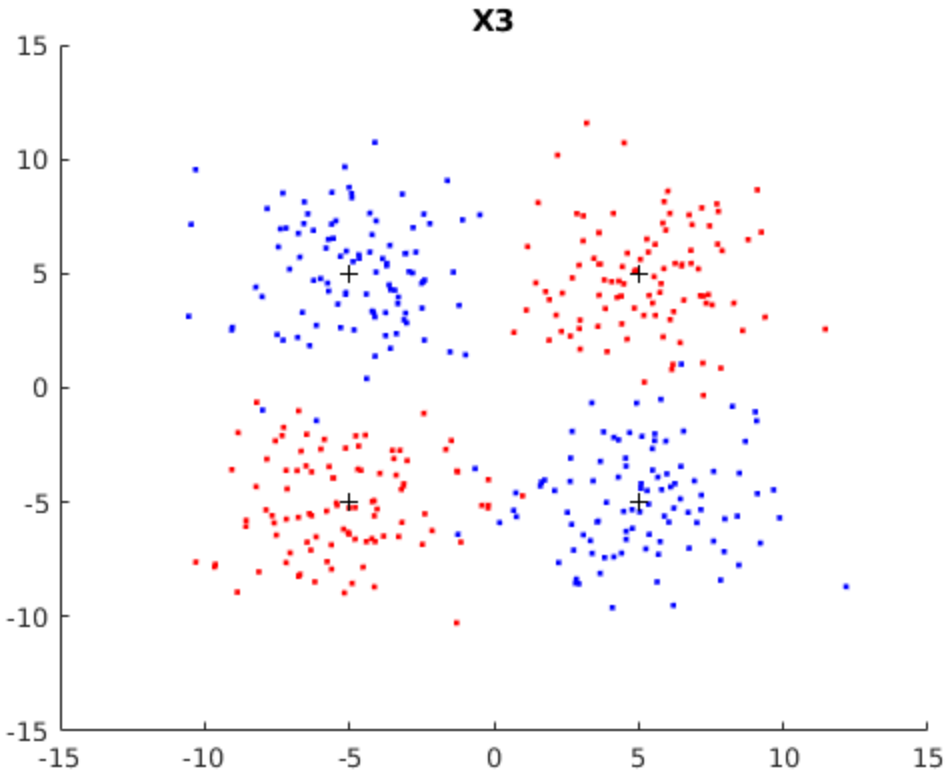

```
clear all
clc
warning('off', 'all')
```

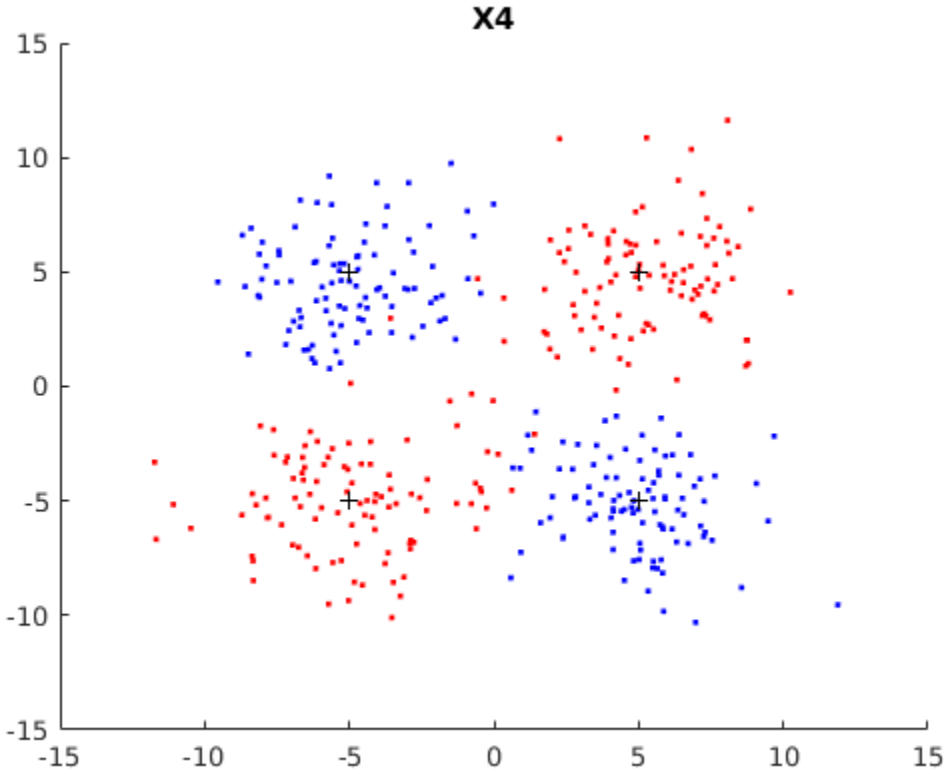
4.1

```
m = [-5 5 5 -5; 5 -5 5 -5];

randn('seed', 0);
[X1 Y1] = data_generator(m, 2, 100);

randn('seed', 10);
[X2 Y2] = data_generator(m, 2, 100);

randn('seed', 0);
[X3 Y3] = data_generator(m, 5, 100);

randn('seed', 10);
[X4 Y4] = data_generator(m, 5, 100);

plotData(X1, Y1, m , 'X1')
plotData(X2, Y2, m , 'X2')
plotData(X3, Y3, m , 'X3')
plotData(X4, Y4, m , 'X4')
```


X1
10
8
6
4
2
0
-2
-4
-6
-8
-10
-10
-5
0
5
10


X2
10
8
6
4
2
0
-2
-4
-6
-8
-10
-10
-5
0
5
10


X3
15
10
5
0
-5
-10
-15
-15
-10
-5
0
5
10
15


X4
15
10
5
0
-5
-10
-15
-15
-10
-5
0
5
10
15

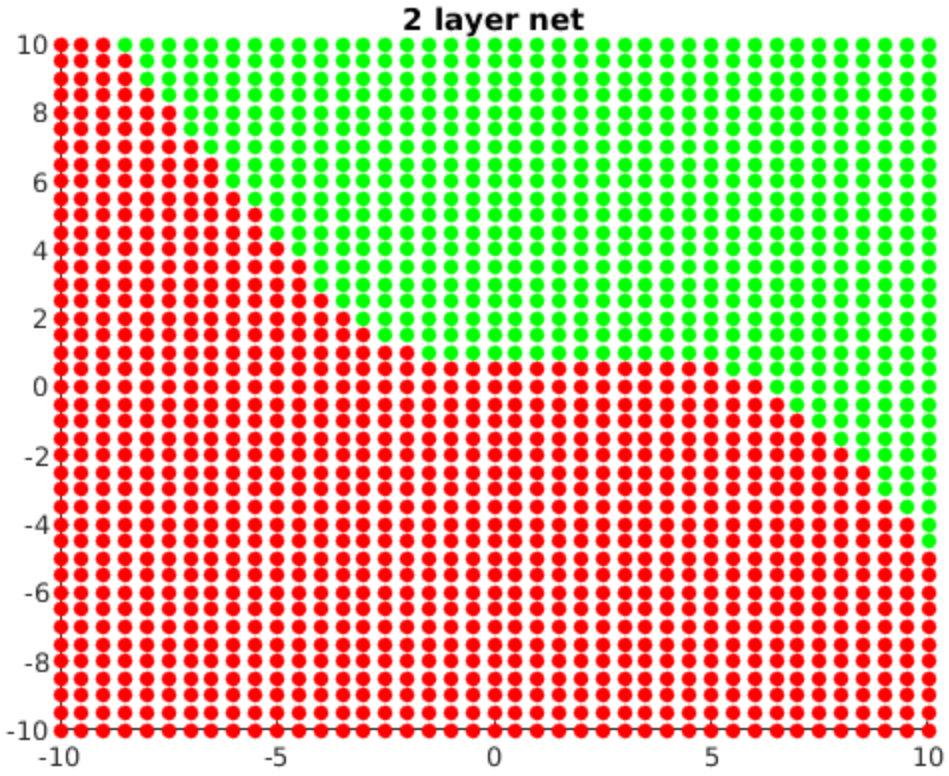
4.2

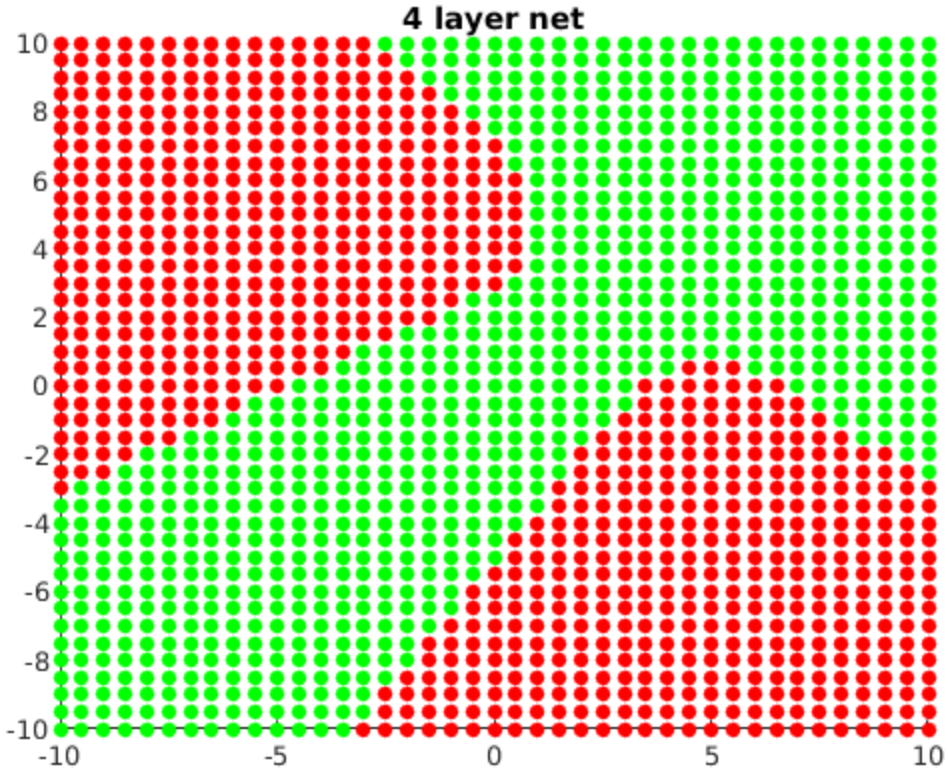
```
[net1 tr1] = NN_training(X1, Y1, 2, 1, 1000, [.01]);
[net2 tr2] = NN_training(X1, Y1, 4, 1, 1000, [.01]);
[net3 tr3] = NN_training(X1, Y1, 15, 1, 1000, [.01]);


% Performance of 2 node
perform(net1, Y1, net1(X1))
perform(net1, Y2, net1(X2))

% Performance of 4 node
perform(net2, Y1, net2(X1))
perform(net2, Y2, net2(X2))

% Performance of 15 node
perform(net3, Y1, net3(X1))
perform(net3, Y2, net3(X2))

xmin = -10;
ymin = -10;
xmax = 10;
ymax = 10;
dx = .5;
dy = .5;

% Plot decision surface of 2 node net
figure()
title('2 layer net')
xlim([xmin xmax])
ylim([ymin ymax])
hold on;
for x=xmin:dx:xmax
 for y=ymin:dy:ymax
  activation = net1([x;y]);
  if activation(1) > 0
   plot(x,y,'.r', 'markersize', 20)
  else
   plot(x,y,'.g', 'markersize', 20)
  end
  hold on;
 end
end

% Plot decision surface of 4 node net
figure()
title('4 layer net')
xlim([xmin xmax])
ylim([ymin ymax])
hold on;
for x=xmin:dx:xmax
 for y=ymin:dy:ymax
  activation = net2([x;y]);
```

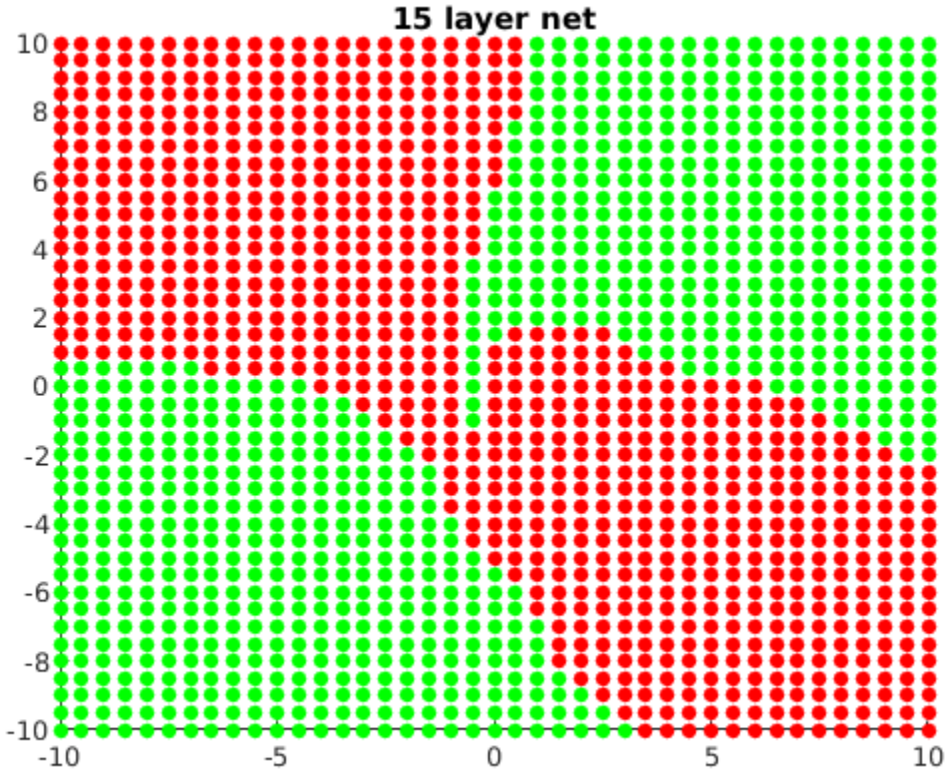
```
  if activation(1) > 0
   plot(x,y,'.r', 'markersize', 20)
  else
   plot(x,y,'.g', 'markersize', 20)
  end
  hold on;
 end
end

% Plot decision surface of 15 node net
figure()
title('15 layer net')
xlim([xmin xmax])
ylim([ymin ymax])
hold on;
for x=xmin:dx:xmax
 for y=ymin:dy:ymax
  activation = net3([x;y]);
  if activation(1) > 0
   plot(x,y,'.r', 'markersize', 20)
  else
   plot(x,y,'.g', 'markersize', 20)
  end
  hold on;
 end
end

%{
The performance of the neural nets increased as the number of nodes
 increased.
This was true for the test data set as well. What I observed was that
 the percent
difference between the training set performance and the testing set
 performance
was higher as the number of nodes increased. A NN with less nodes is
 more generalized
which makes it less prone to overfit to the training set.
%}
```

```
ans =

    0.5282


ans =

    0.5291


ans =

    0.0227
```

```
ans =

    0.0229

ans =

    0.0106

ans =

    0.0208
```




4 layer net
10
8
6
4
2
0
-2
-4
-6
-8
-10
-10
-5
0
5
10

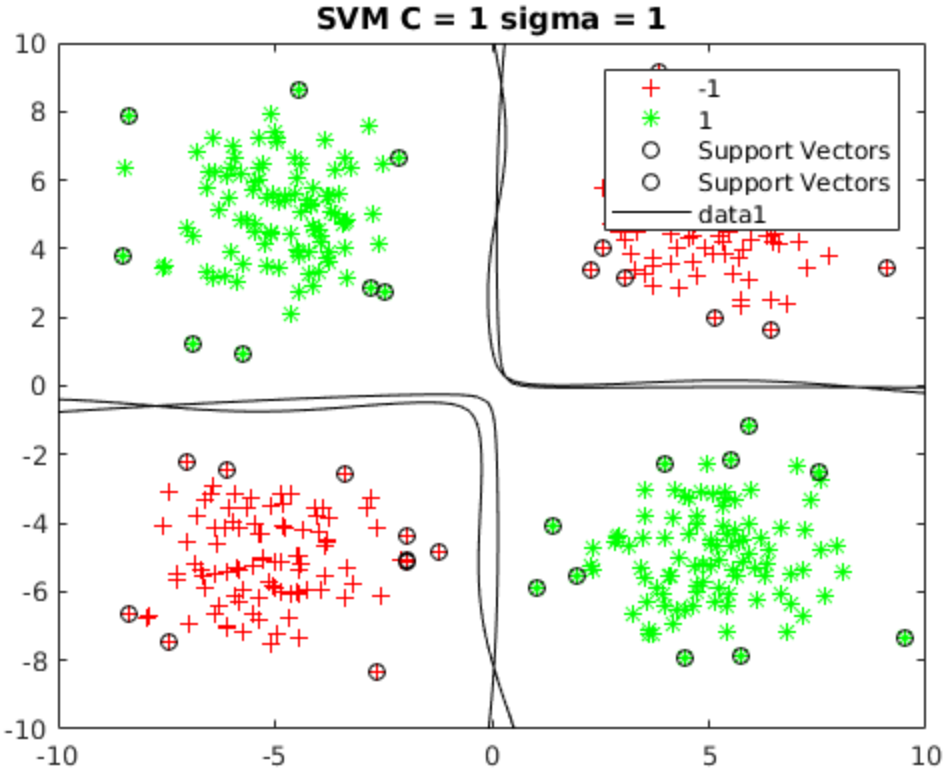

15 layer net
10
8
6
4
2
0
-2
-4
-6
-8
-10
-10
-5
0
5
10

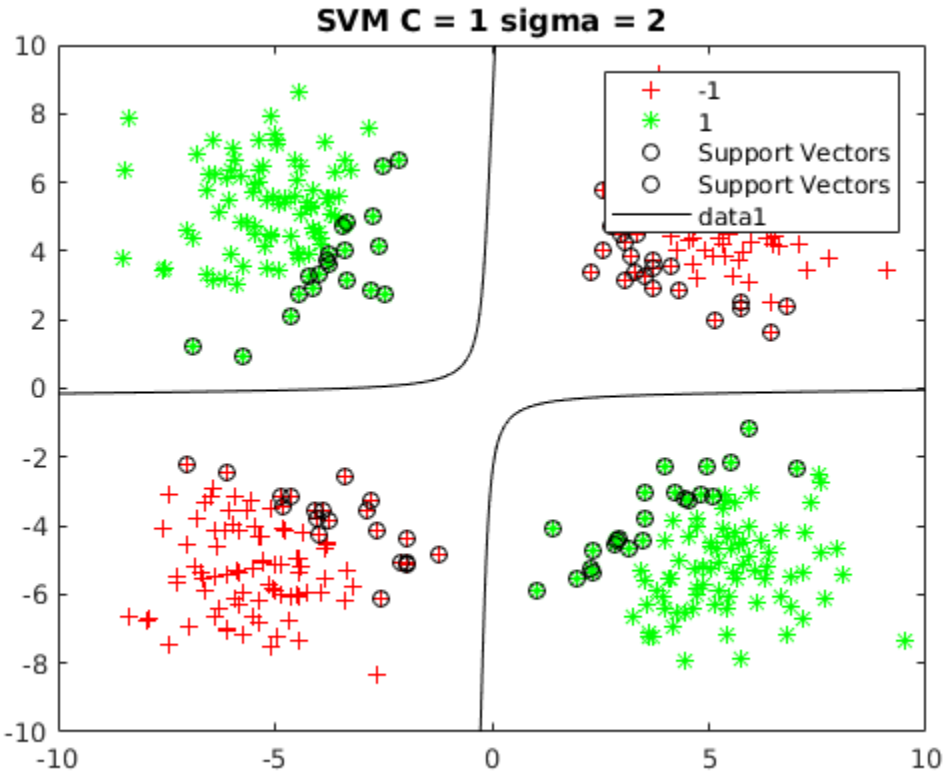
4.6

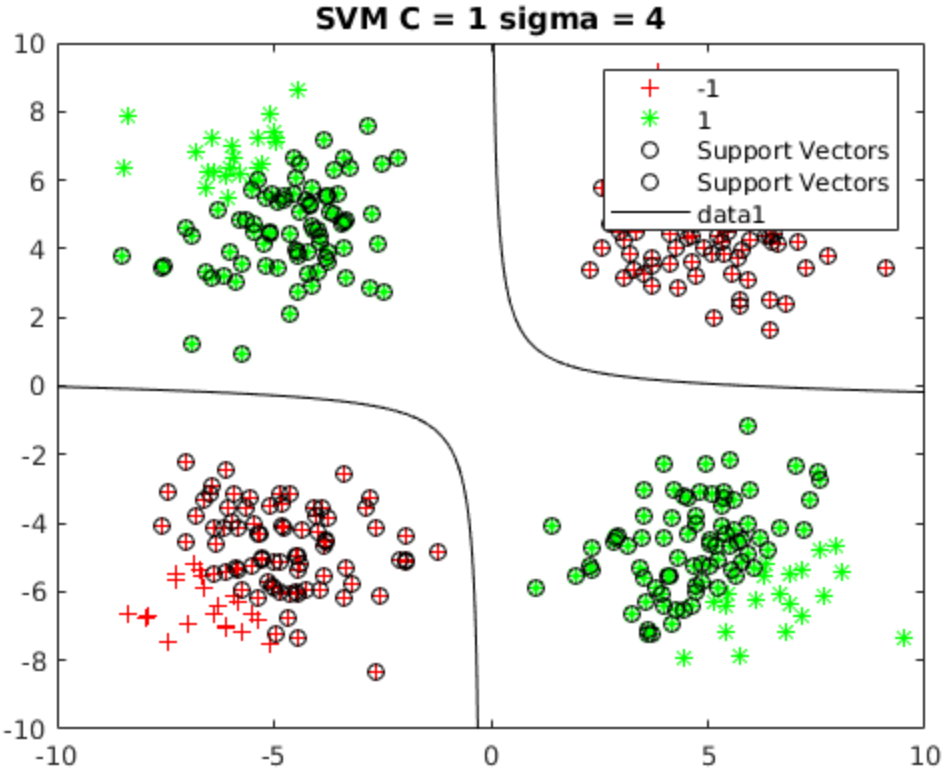
```
C = [1 100 1000];
sigma = [0.5 1 2 4];

for i = 1:3
 for j = 1:4
  hold on
  figure()
  [SVMstruct, svIndex, pe_tr, pe_te] = SVM_clas(X1, Y1, X2, Y2, .001,
 C(i), sigma(j));
  title(['SVM C = ' num2str(C(i)) ' sigma = ' num2str(sigma(j))])
  fprintf('Performance on training data with C = %i and sigma = %f for
 X1 data set: %f\n', ...
  C(i), sigma(j), pe_tr);
  fprintf('Performance on testing data with C = %i and sigma = %f for
 X1 data set: %f\n', ...
  C(i), sigma(j), pe_te);
  fprintf('\n')
  hold off
 end
end

%{
The training set error for every SVM converged to zero, and the
 testing set was at a constant
0.0025 performance error.
%}
```

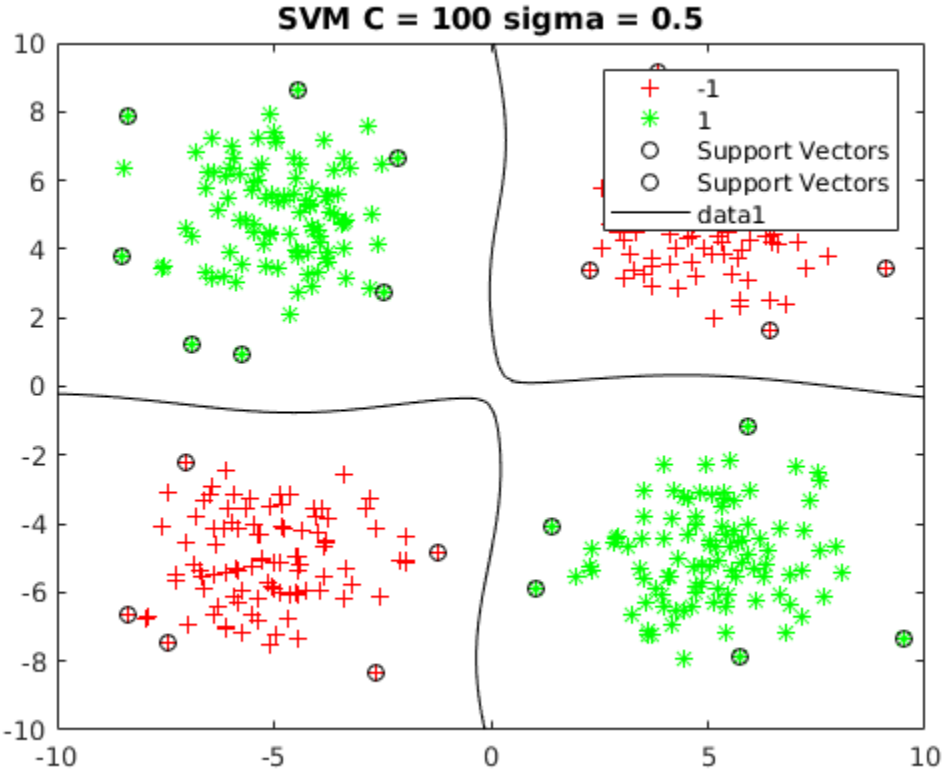
```
Performance on training data with C = 1 and sigma = 0.500000 for X1
 data set: 0.000000
Performance on testing data with C = 1 and sigma = 0.500000 for X1
 data set: 0.002500

Performance on training data with C = 1 and sigma = 1.000000 for X1
 data set: 0.000000
Performance on testing data with C = 1 and sigma = 1.000000 for X1
 data set: 0.002500

Performance on training data with C = 1 and sigma = 2.000000 for X1
 data set: 0.000000
Performance on testing data with C = 1 and sigma = 2.000000 for X1
 data set: 0.002500

Performance on training data with C = 1 and sigma = 4.000000 for X1
 data set: 0.000000
Performance on testing data with C = 1 and sigma = 4.000000 for X1
 data set: 0.002500

Performance on training data with C = 100 and sigma = 0.500000 for X1
 data set: 0.000000
Performance on testing data with C = 100 and sigma = 0.500000 for X1
 data set: 0.002500
```

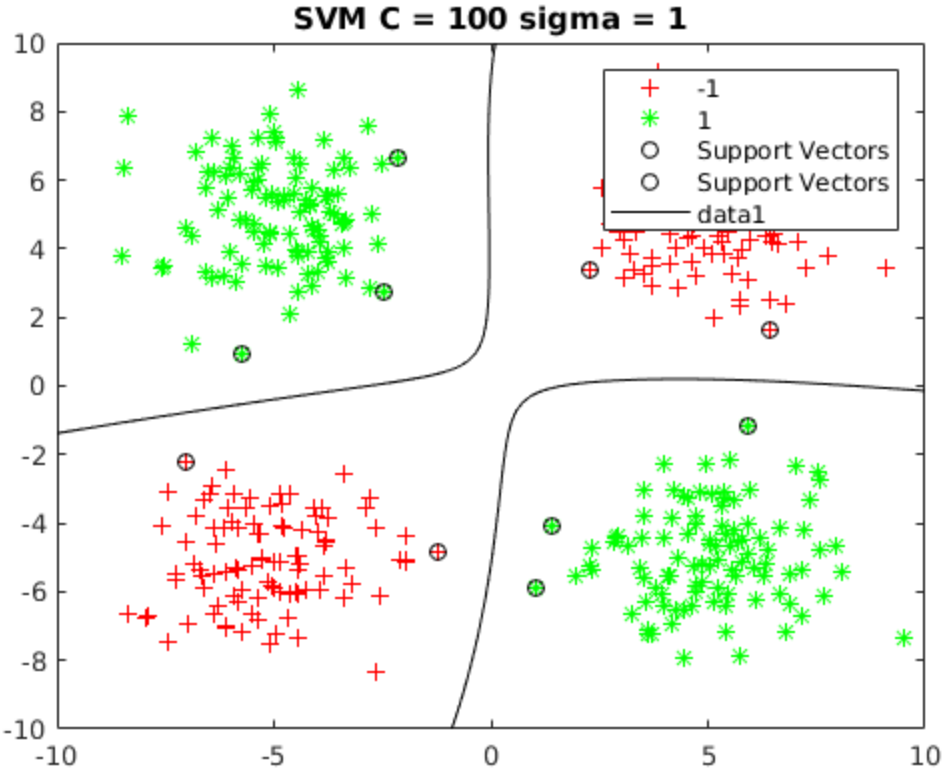
```
Performance on training data with C = 100 and sigma = 1.000000 for X1
 data set: 0.000000
Performance on testing data with C = 100 and sigma = 1.000000 for X1
 data set: 0.002500

Performance on training data with C = 100 and sigma = 2.000000 for X1
 data set: 0.000000
Performance on testing data with C = 100 and sigma = 2.000000 for X1
 data set: 0.002500

Performance on training data with C = 100 and sigma = 4.000000 for X1
 data set: 0.000000
Performance on testing data with C = 100 and sigma = 4.000000 for X1
 data set: 0.002500

Performance on training data with C = 1000 and sigma = 0.500000 for X1
 data set: 0.000000
Performance on testing data with C = 1000 and sigma = 0.500000 for X1
 data set: 0.002500

Performance on training data with C = 1000 and sigma = 1.000000 for X1
 data set: 0.000000
Performance on testing data with C = 1000 and sigma = 1.000000 for X1
 data set: 0.002500

Performance on training data with C = 1000 and sigma = 2.000000 for X1
 data set: 0.000000
Performance on testing data with C = 1000 and sigma = 2.000000 for X1
 data set: 0.002500

Performance on training data with C = 1000 and sigma = 4.000000 for X1
 data set: 0.000000
Performance on testing data with C = 1000 and sigma = 4.000000 for X1
 data set: 0.002500
```

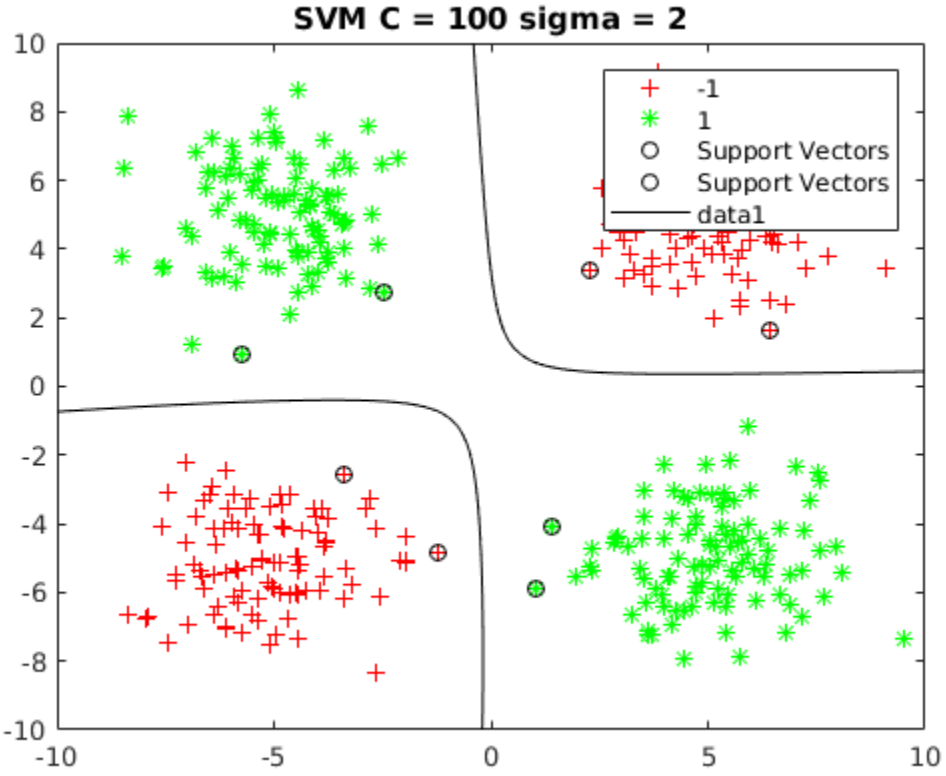

SVM C = 1 sigma = 1
-1
1
Support Vectors
Support Vectors
data1
10
8
6
4
2
0
-2
-4
-6
-8
-10
-10
-5
0
5
10

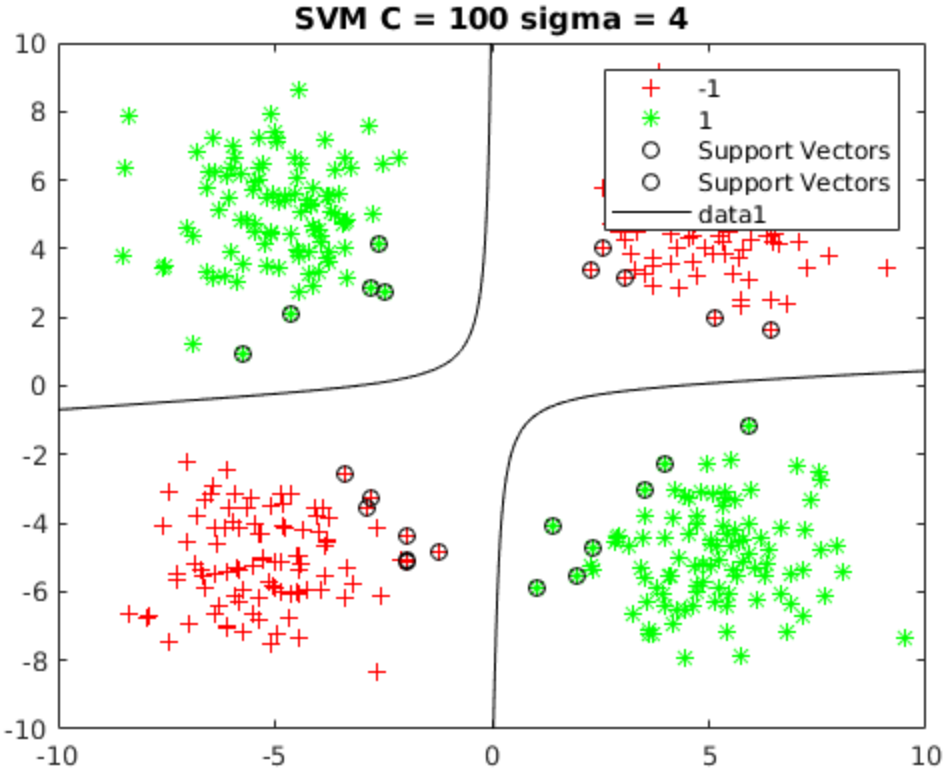

SVM C = 1 sigma = 2
-1
1
Support Vectors
Support Vectors
data1
10
8
6
4
2
0
-2
-4
-6
-8
-10
-10
-5
0
5
10

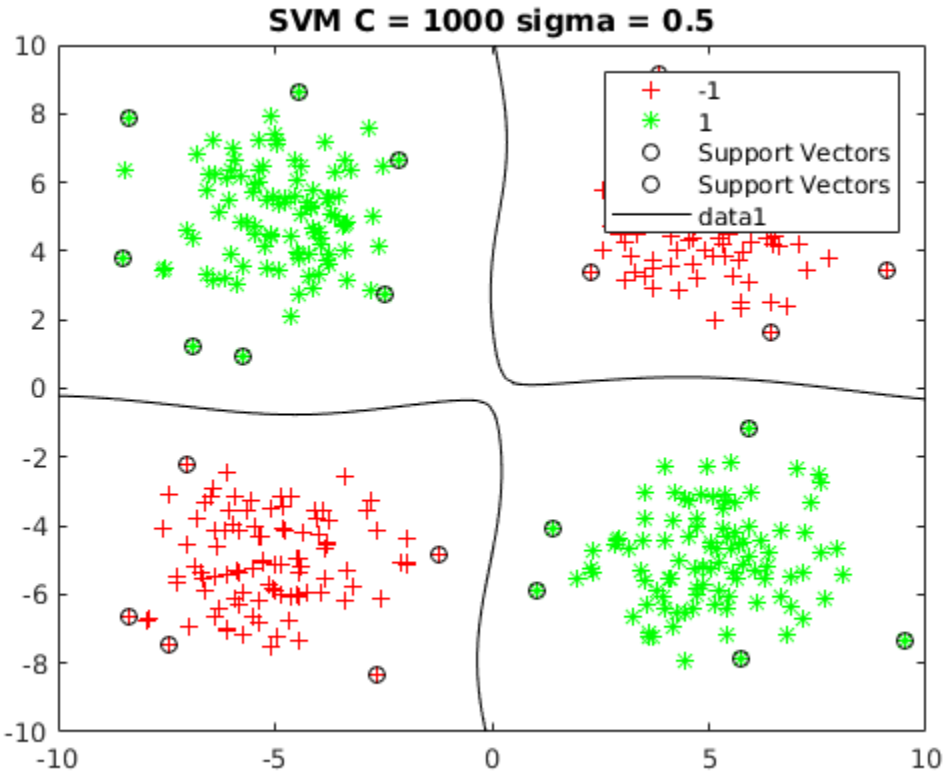

SVM C = 1 sigma = 4
-1
1
Support Vectors
Support Vectors
data1
10
8
6
4
2
0
-2
-4
-6
-8
-10
-10
-5
0
5
10

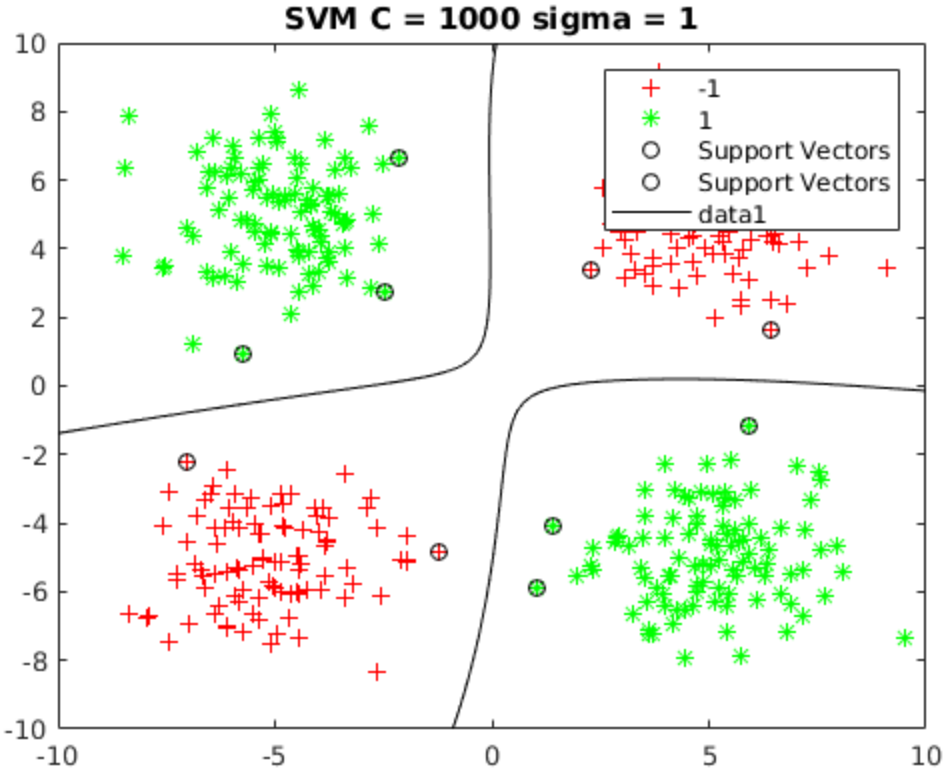

SVM C = 100 sigma = 0.5
-1
1
Support Vectors
Support Vectors
data1
10
8
6
4
2
0
-2
-4
-6
-8
-10
-10
-5
0
5
10

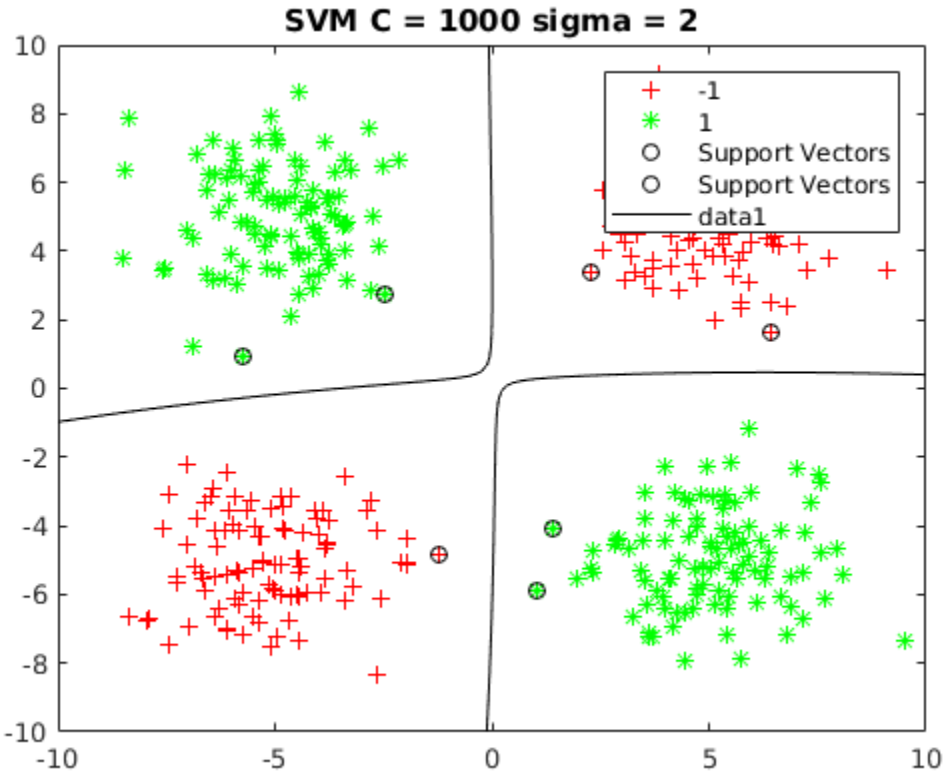

SVM C = 100 sigma = 1
-1
1
Support Vectors
Support Vectors
data1
10
8
6
4
2
0
-2
-4
-6
-8
-10
-10
-5
0
5
10

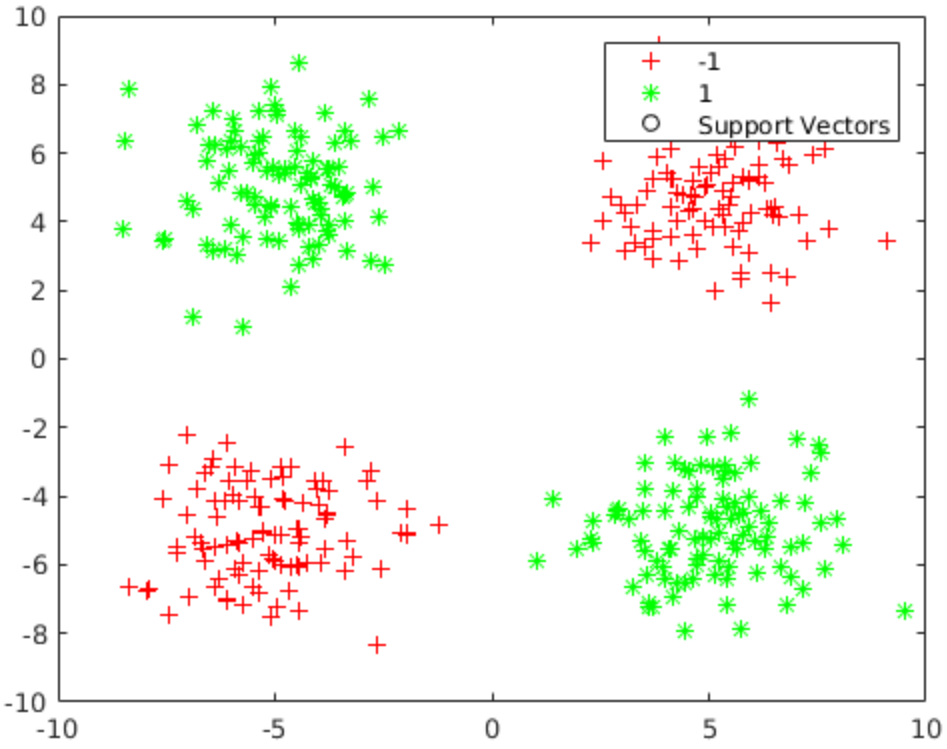

SVM C = 100 sigma = 2
-1
1
Support Vectors
Support Vectors
data1
10
8
6
4
2
0
-2
-4
-6
-8
-10
-10
-5
0
5
10

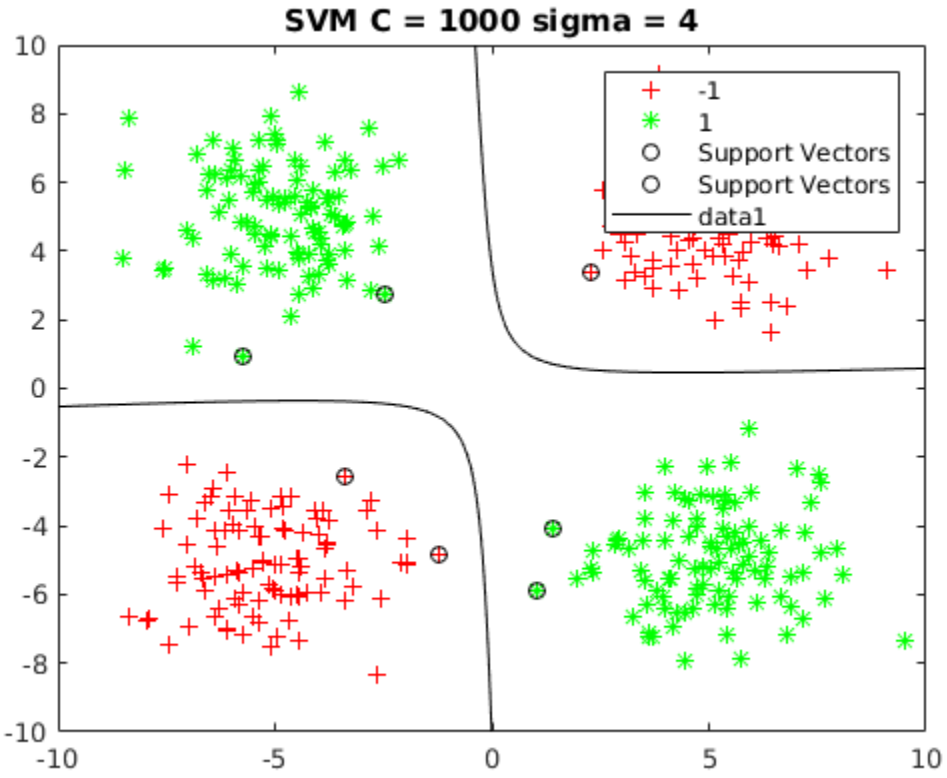

SVM C = 100 sigma = 4
-1
1
Support Vectors
Support Vectors
data1
10
8
6
4
2
0
-2
-4
-6
-8
-10
-10
-5
0
5
10


SVM C = 1000 sigma = 0.5
-1
1
Support Vectors
Support Vectors
data1
10
8
6
4
2
0
-2
-4
-6
-8
-10
-10
-5
0
5
10


SVM C = 1000 sigma = 1
-1
1
Support Vectors
Support Vectors
data1
10
8
6
4
2
0
-2
-4
-6
-8
-10
-10
-5
0
5
10


SVM C = 1000 sigma = 2
-1
1
Support Vectors
Support Vectors
data1
10
8
6
4
2
0
-2
-4
-6
-8
-10
-10
-5
0
5
10


-1
1
Support Vectors


SVM C = 1000 sigma = 4
-1
1
Support Vectors
Support Vectors
data1

Decision trees

```
dt_1 = classregtree(X1', Y1, 'prune', 'off');
dt_2 = classregtree(X2', Y2, 'prune', 'off');

fprintf('Error testing X2 on X1 learn: %f\n',computeError(dt_1(X2')',
 Y2))
fprintf('Error testing X1 on X2 learn: %f\n',computeError(dt_2(X1')',
 Y1))

dt_3 = classregtree(X1', Y1, 'prune', 'on');
dt_4 = classregtree(X2', Y2, 'prune', 'on');

fprintf('Error testing X2 on X1 learn pruned: %f
\n',computeError(dt_3(X2')', Y2))
fprintf('Error testing X1 on X2 learn pruned: %f
\n',computeError(dt_4(X1')', Y1))
```

```
Error testing X2 on X1 learn: 0.002500
Error testing X1 on X2 learn: 0.022500
Warning: classregtree will be removed in a future release. Use the
 predict
method of an object returned by fitctree or fitrtree instead.
Error testing X2 on X1 learn pruned: 0.002500
Warning: classregtree will be removed in a future release. Use the
 predict
method of an object returned by fitctree or fitrtree instead.
Error testing X1 on X2 learn pruned: 0.022500
```

Published with MATLAB® R2017a