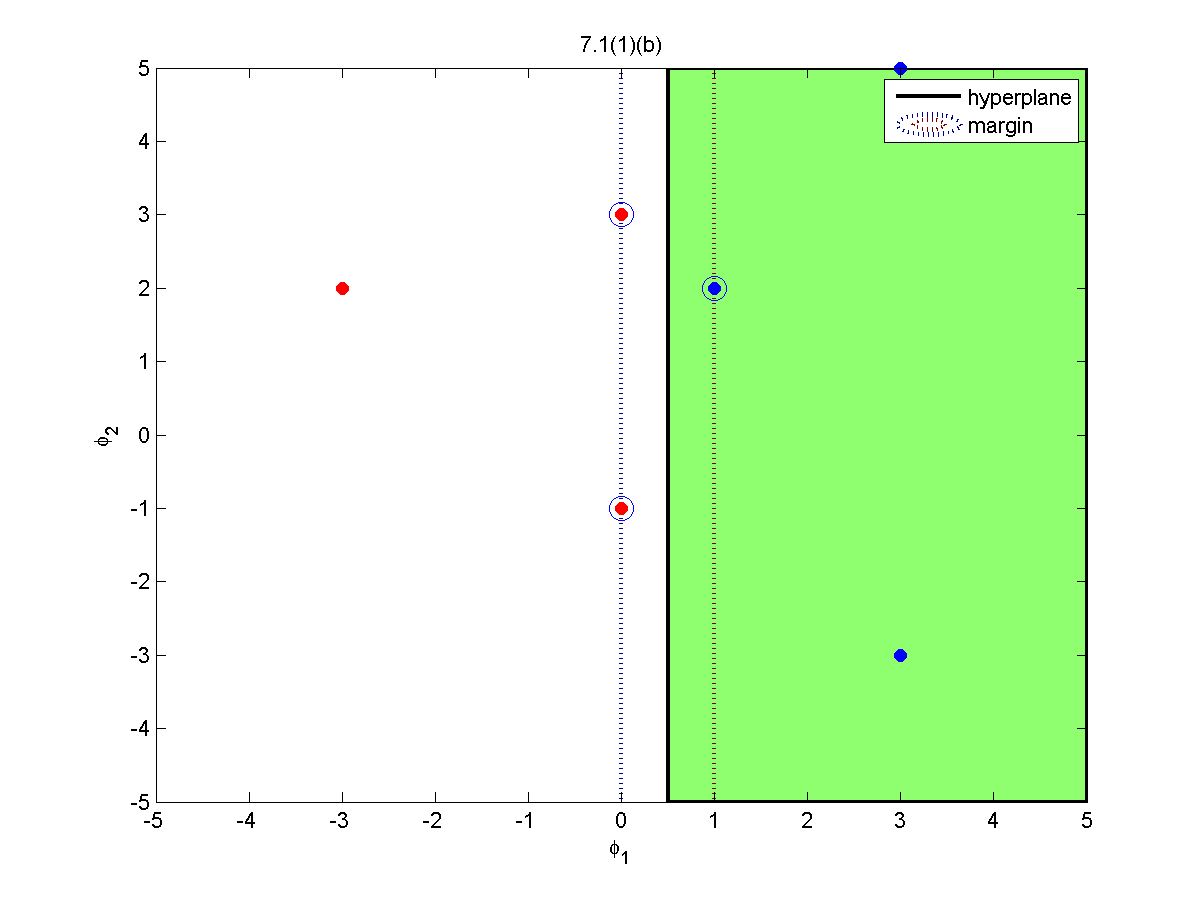
**Homework #7**

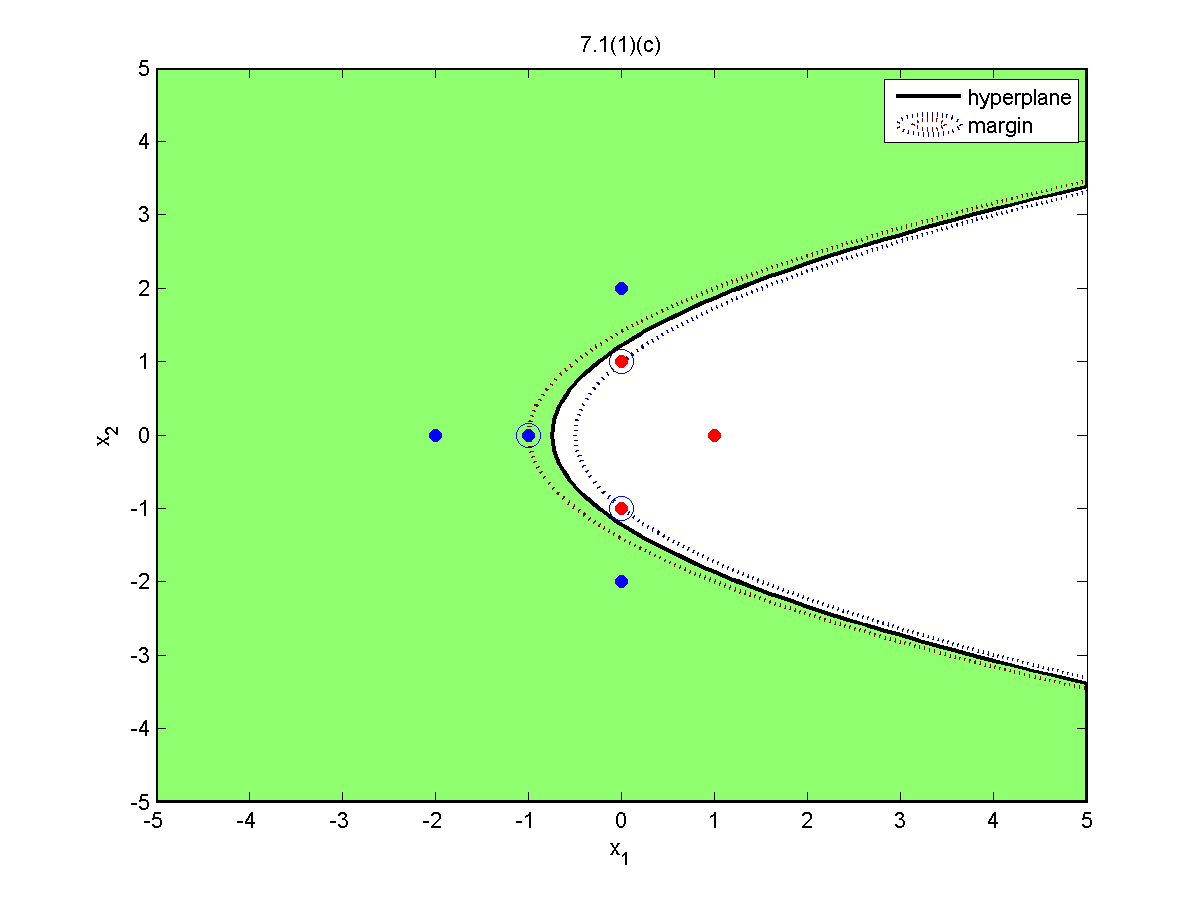
1. **Transforms: Explicit versus Implicit**
   * 1. The set would be

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

* + 1. The optimal separating hyperplane in would be , show as follow:



* + 1. The optimal separating hyperplane in would be , show as follow:

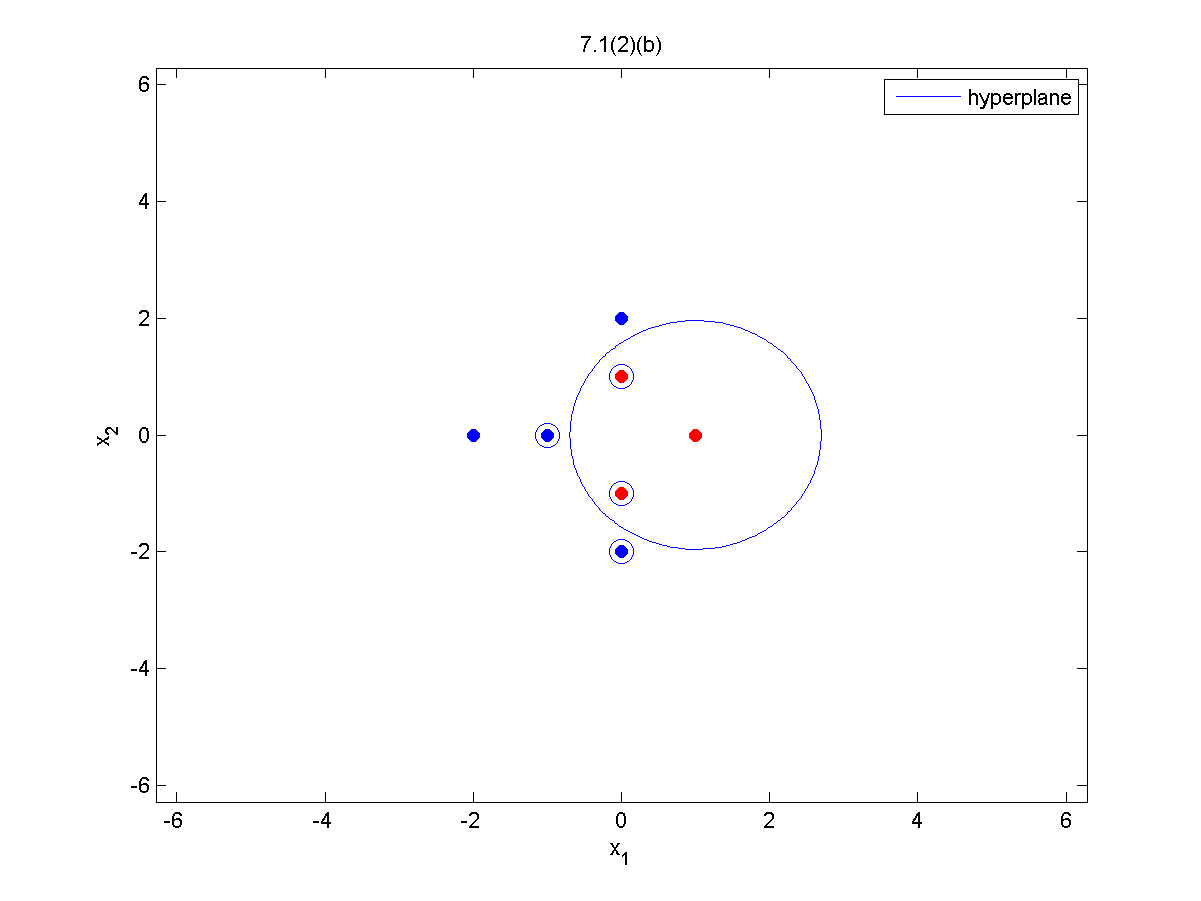


* + 1. The optimal would be

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |

That mean we could have 4 support vectors in the data.

* + 1. The equation in look like



Where we use function in MATLAB and get an ugly form of it

* 1. They aren’t the same curve, and they have a little chance to be the same curve. The kernel function is mapping to coordinate transformation that can represent all of quadratic functions, and the curve we got at problem 7.1(1) is a quadratic function too. Anyway, they are indeed the different function.

1. **A Leave-One-Out Bound of Support Vector Machine**
   1. Consider with

Therefore, the function when , and we could say that satisfies all constraints of because is optimal solution of .

* 1. Assume that is not the best solution, we got a better solution . Then use the combine the optimal , and get the optimal , which is a contradiction. Therefore, we have
  2. The leave-one-out error of SVM is upper bounded by the percentage of support vectors. This argument implies that if you take off the non-support vector, the training error wouldn’t be changed. By the definition of support vector, should be zero if the point isn’t a support vector. So if we take of a non-support vector, , that imply the inequality

Which means the point is in the right side and behind the max margin line at least 1 unit distance. So the leave-one-out error won’t affected by non-support vectors, but only affected by support vector, therefore

1. **Experiments with Linear Support Vector Machine**
   1. The error rates and the equations show as follow:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

We can find that when C = 0.01, the max margin line goes very up and loss its function. The result shows that the constraint of is really important to soft margin support vector machine because when C goes smaller, we loss the control under constraint .

* 1. The error rates and the equations show as follow:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

When C = 0.01, the max margin line still not work. However, in the other cases, the soft margin SVM works well.

1. **Experiments with Nonlinear Support Vector Machine**
   1. When kernel is . The result show as follow:
      1. Training error

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

* + 1. Testing error

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

* + 1. Number of support vector

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

We can see that when C goes up, which mean we have a thinner hyperplane, so our classifier can achieve less error in testing and training. In the other hand, when C drop down, the constraint become useless, it give us lots of support vectors; So we will get a bad classifier, just like we got in 7.3.

About d, the result seems like when we have a big d; thing will go easier. However the best testing error is in . The trend may over fitting in some sense.

* 1. When kernel is . The result show as follow:
     1. Training error

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

* + 1. Testing error

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

* + 1. Number of support vector

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Same situation, we almost fail on C = 0.001 in all of cases, and the trend is the same; C goes up, error goes down.

Take a look at , it don’t like d we look above because we didn’t observer the trend like goes up and error goes down. We still have the best testing error at , the same place as 7.4(1) in the table.