Roi Herzig ID: 300360310 EMAIL:roeiherzig@mail.tau.ac.il

Moshe Raboh ID:300611878 EMAIL:shikorab@gmail.com

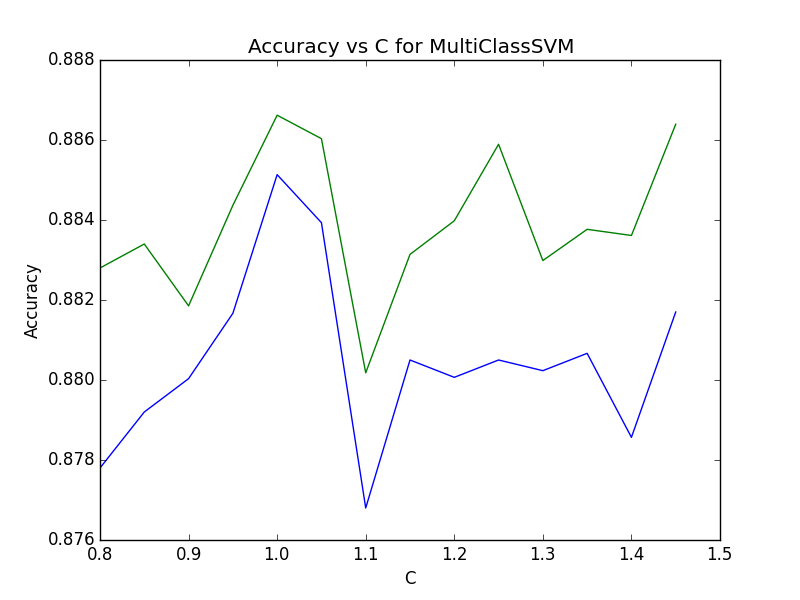
**Introduction To Machine Learning – EX 3**

1. **A:**Directory: " /specific/a/home/cc/students/csguests/roeiherzig/ML/EX3"See function "part\_a" in file "q6.py"

* We performed a grid search to find the best learning rate. The graph below shows the accuracy of both mean validation accuracy and mean training accuracy over the best range we found on the grid search.  
  We extracted the best learning rate from the graph.
* We used the best learning rate while the grid search for the best C. The graph below shows the accuracy of both mean validation accuracy and mean training accuracy over the best range we found on the grid search.  
  We extracted the best C from the graph.
* We used number of iterations**=**10000

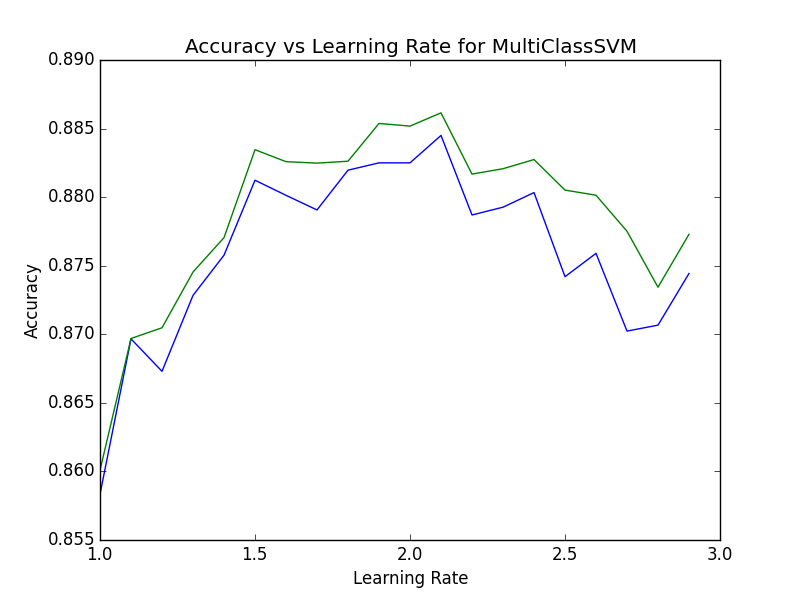
Trivially bigger number of iterations will slightly improve the results.

* The best results got with eta~2.2 and C~0.98 yields validation accuracy of ~0.89

Name of the image: “q6\_part\_a1.png”  


The green line represents training accuracy and the blue line represents validation accuracy.

Name of the image: “q6\_part\_a2.png”

****

The green line represents training accuracy and the blue line represents validation accuracy.

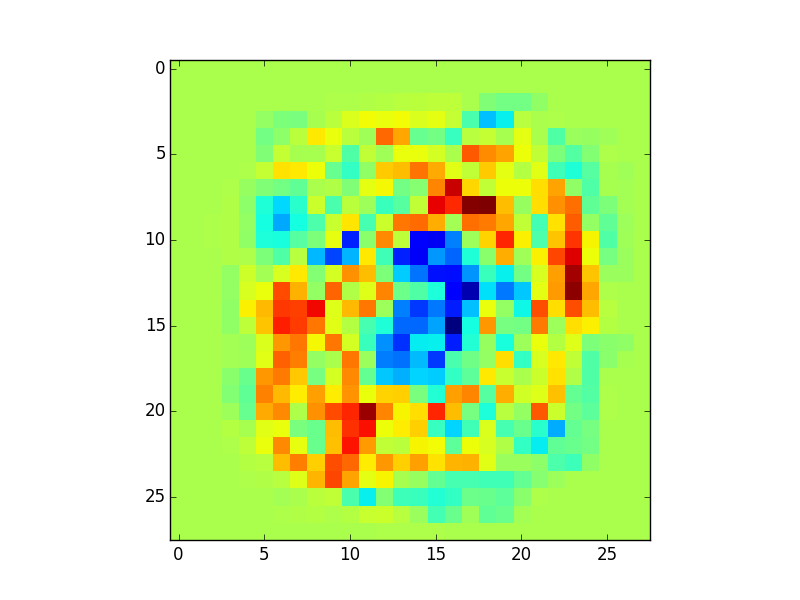
**B:**

Directory: " /specific/a/home/cc/students/csguests/roeiherzig/ML/EX3"See function "part\_b" in file "q6.py"

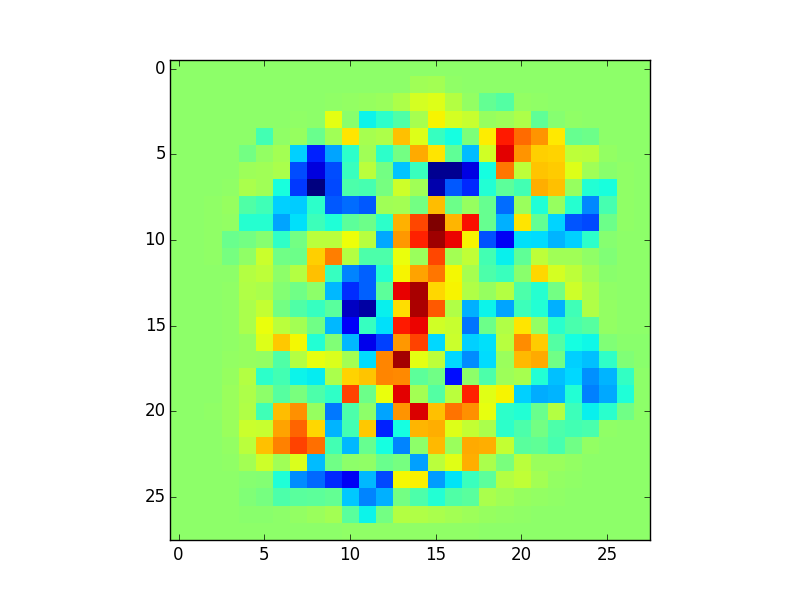
The images below are the weight matrix. As we can see, the weights that are related to the different features between specific class compare to the other classes.

As we can see the images reflects the digits that they should classify, however not as good as the binary classifier (In the multiclass version, we trained a binary svm just in case wyi\*x – wj\*x <=1)

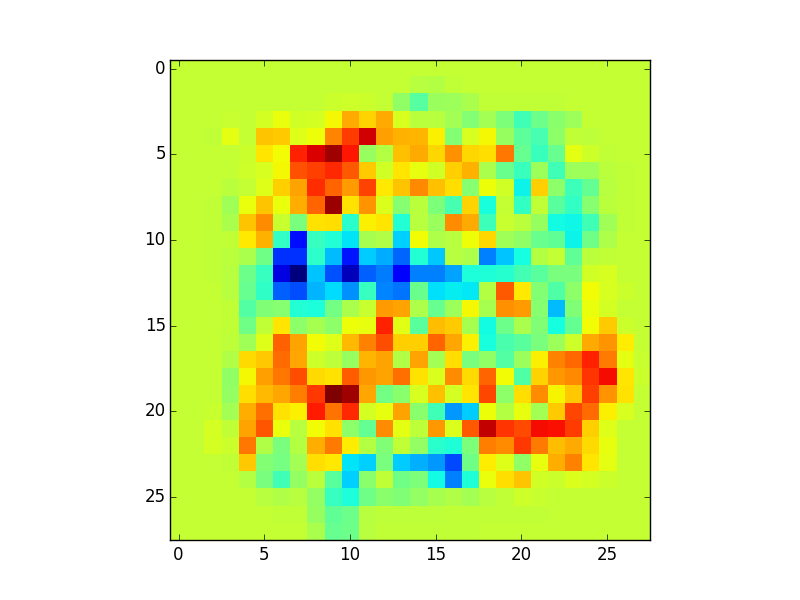
Name of the image: “q6\_partb\_class0.png”



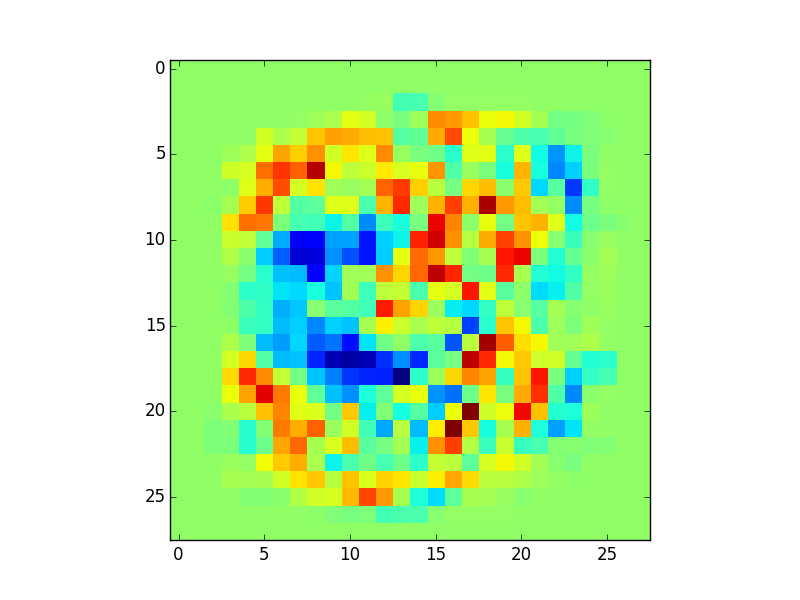
Name of the image: “q6\_partb\_class1.png”



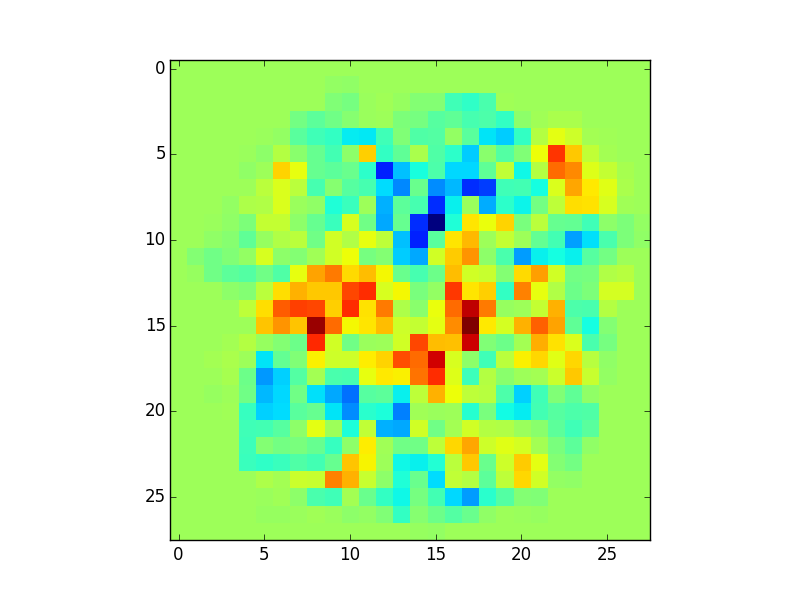
Name of the image: “q6\_partb\_class2.png”



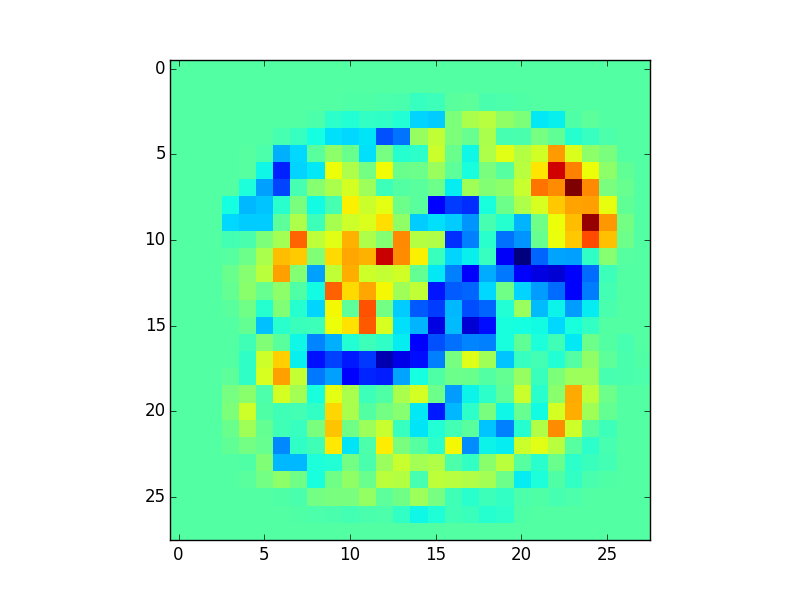
Name of the image: “q6\_partb\_class3.png”



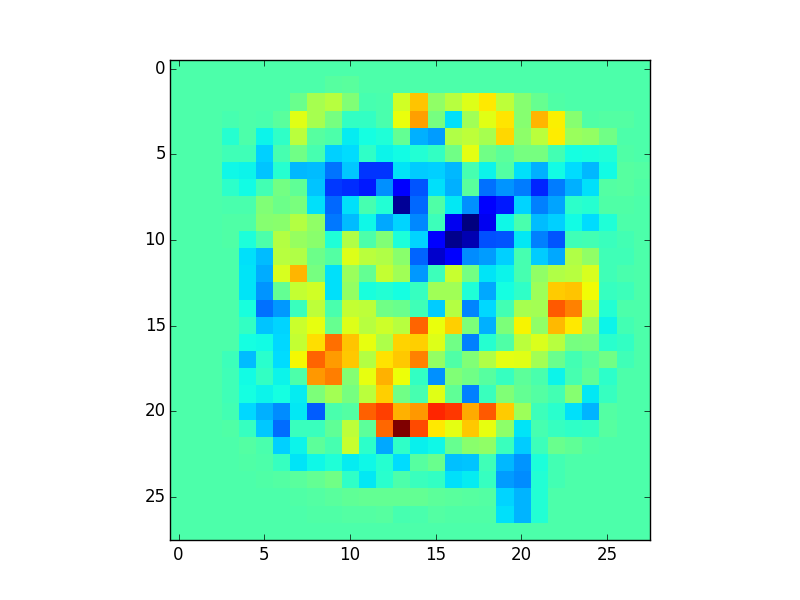
Name of the image: “q6\_partb\_class4.png”



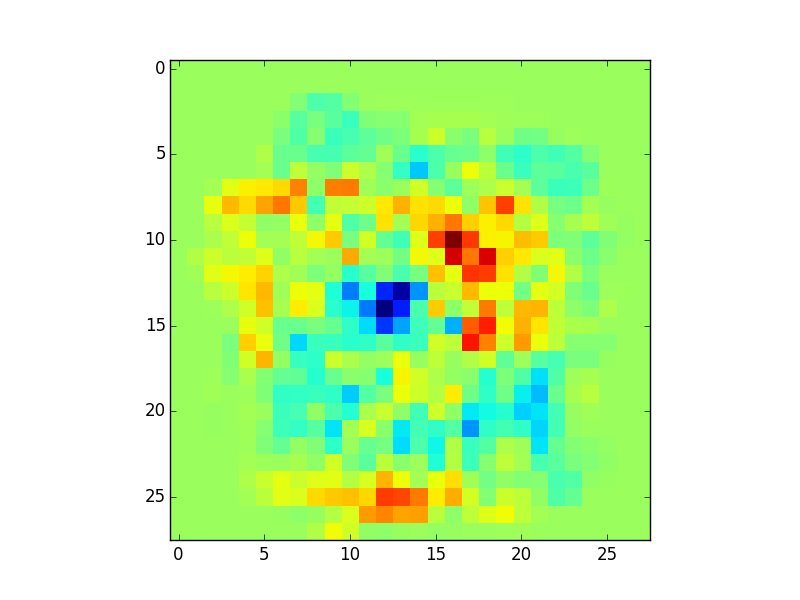
Name of the image: “q6\_partb\_class5.png”



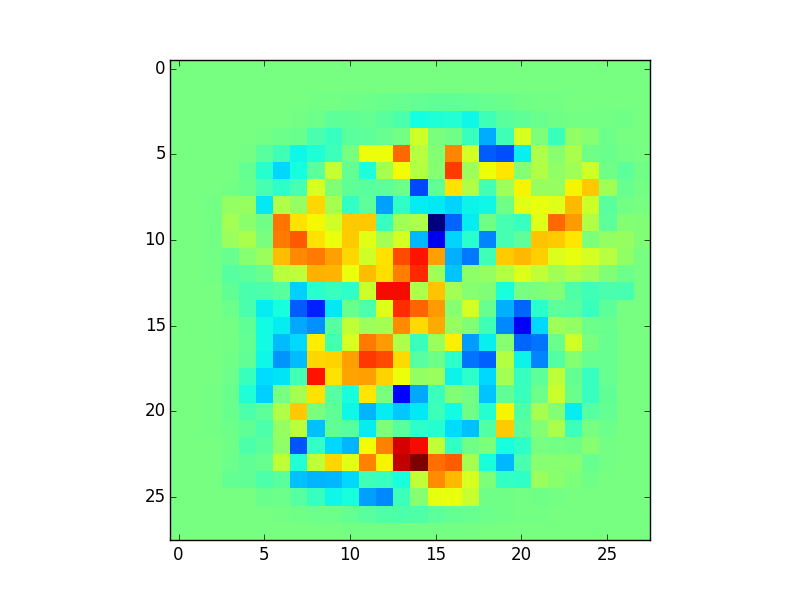
Name of the image: “q6\_partb\_class6.png”



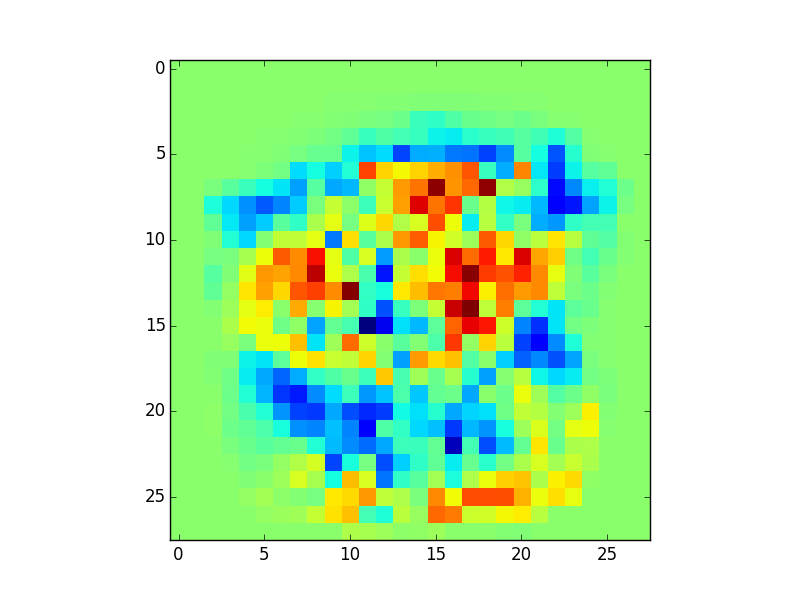
Name of the image: “q6\_partb\_class7.png”



Name of the image: “q6\_partb\_class8.png”



Name of the image: “q6\_partb\_class9.png”



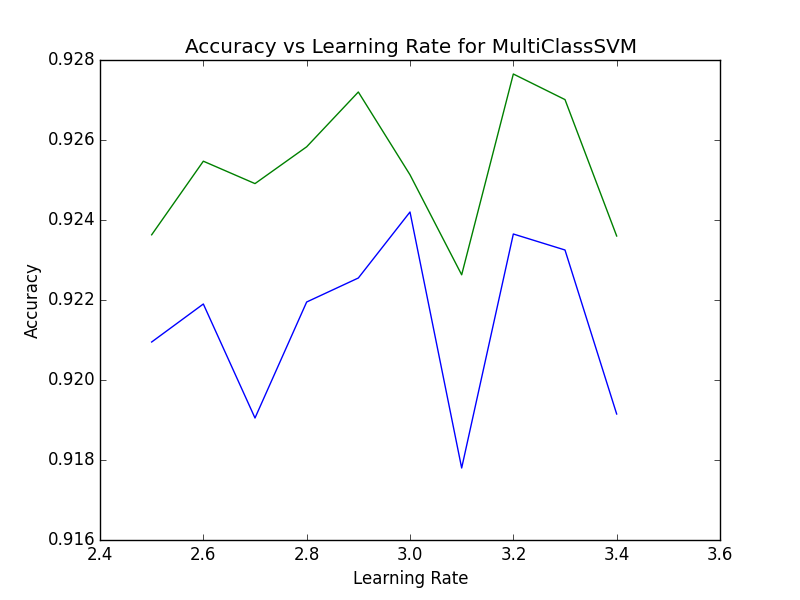
**C:**Directory: " /specific/a/home/cc/students/csguests/roeiherzig/ML/EX3"See function "part\_c" in file "q6.py"  
Mean accuracy of the full train samples: ~88%

1. **A:**Directory: " /specific/a/home/cc/students/csguests/roeiherzig/ML/EX3"See function "part\_a" in file "q7.py"

* We performed a grid search to find the best learning rate. The graph below shows the accuracy of both mean validation accuracy and mean training accuracy over the best range we found on the grid search.  
  We extracted the best learning rate from the graph.
* We used the best learning rate while the grid search for the best C. The graph below shows the accuracy of both mean validation accuracy and mean training accuracy over the best range we found on the grid search.  
  We extracted the best C from the graph.
* We used number of iterations**=**5000

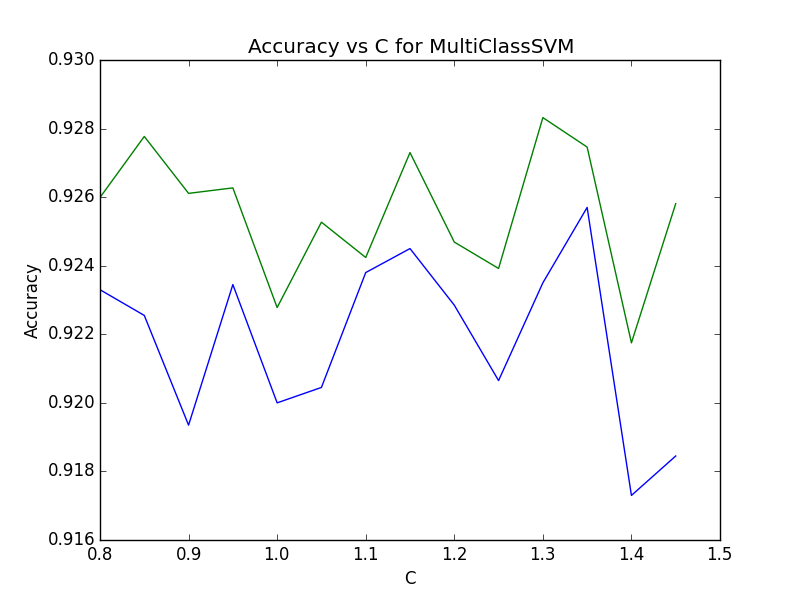
Trivially bigger number of iterations will slightly improve the results.

* The best results got with eta~3.2 and C~1.35 yields validation accuracy of ~0.92

Name of the image: “q7\_part\_a1.png”  


The green line represents training accuracy and the blue line represents validation accuracy.

Name of the image: “q7\_part\_a2.png”

****

The green line represents training accuracy and the blue line represents validation accuracy.

**B:**Directory: " /specific/a/home/cc/students/csguests/roeiherzig/ML/EX3"See function "part\_b" in file "q9.py"  
Mean accuracy of the full train samples: ~92%

C:  
Although we didn’t get to implement the new kernel, we might do better RBF kernel, since it defines a function space that is a lot larger compared the polynomial kernels. The RBF kernel gives you access to all analytic functions, so in some sense we can view this kernel as being as powerful as infinite order polynomial kernel.  
However, we will need to be careful to not overfit.