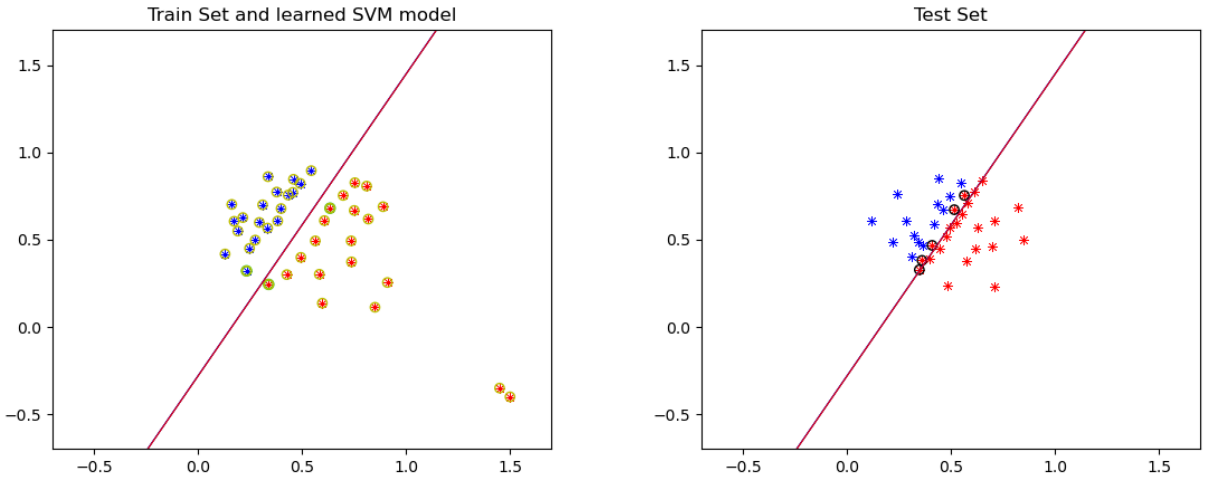
Question 1

(a)

Number of SV: 3
Accuracy on train data with C = 1000: 1.0
Accuracy on test data with C = 1000: 0.8648648648648649



Adjustments to `apply_a.py`

`python apply_a.py` threw a lot of errors, which were mainly due to not enforcing a datatype (i.e. matrix vs nparray) and not taking the dimensions requested in `simlin.py` into account.

For this reason, we have modified `apply_a.py`.

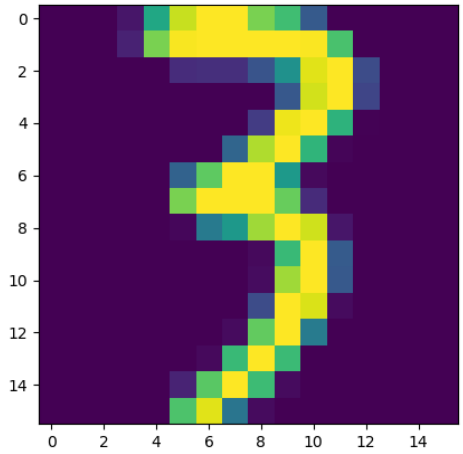
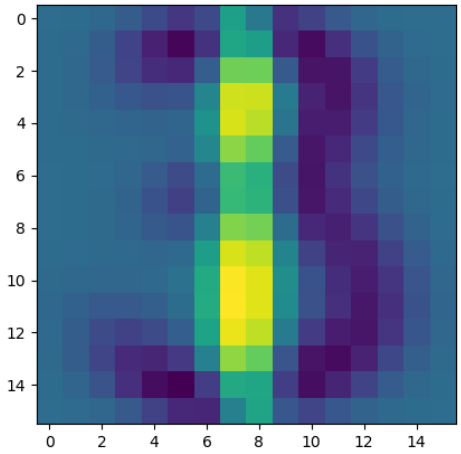
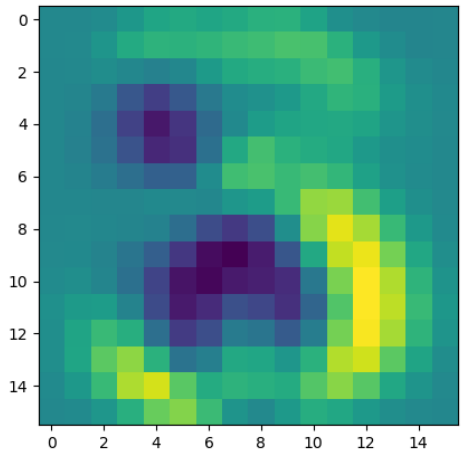
(b+c)

1-vs-3

Number of SV: 26
Width of margin: 0.000527852066940575
Train Error 1-vs-3: 7
Test Error 1-vs-3: 0.013953488372093023

3-vs-8

Number of SV: 89
Width of margin: 0.00010191438464165348
Train Error 3-vs-8: 66
Test Error 3-vs-8: 0.09939759036144578

Adjustments to `apply_bc.py`

As `python apply_a.py`, `python apply_bc.py` threw a lot of errors, which were mainly due to not enforcing a datatype (i.e. matrix vs nparray) and not taking the dimensions requested in `simlin.py` into account.

For this reason, we have modified `apply_bc.py`.

(d)

Accuracy of kernel SVM with C=1000 and norm=5: 0.40540540540540543

This shows, that the accuracy of the kernel SVM is much lower than the accuracy of the linear SVM.

Adjustments to `apply_d.py` and `svmkern.py`

There was a mismatch between the number of variables unpacked from the function `svmkern` in `apply_d.py` and the number of variables to be returned by this function, defined by its description. We have extended the function to return the same values as `svmlin`.
As with the previous adjustments that had to be made, we again had to enforce datatypes and dimensions.