60-473 Assignment 02
Zachary Rudzinski, 104064162

Submittables
a. The SVM tool used was scikit, a python library used for simple, painless machine learning.
I used it to create models for our SVMs (using all 3 kernels), to perform 10-fold cross validation, and to report the related metrics of classifier srength.

Please see the below pages; Included are the results from each step of the assignment as they were completed.

Please also note that my plots for these classifiers are included the submitted “./plots” directory. Please see them there at full resolution. Source code is included as well.

2. Using the SVM tool you installed, run three different classifiers with the following kernels and their parameters:
(a) SVM-L: linear kernel;
(b) SVM-P: polynomial kernel – degree 2;
(c) SVM-R: RBF

My plots for these classifiers are included the submitted "./plots" directory. Please see them there at full resolution.

```
-- SVM Linear Kernel, no CV --
Dataset: datasets/clusterincluster.csv	Score: 0.48787878787878786
	tn: 161  fp: 0  fn: 169  tp: 0
	specificity: 1.0
	sensitivity: 1.0
	ppv: 1.0
	npv: 1.0
	# Note: Divide by 0 exception here
Dataset: datasets/halfkernel.csv	Score: 0.7393939393939394
	tn: 150  fp: 24  fn: 62  tp: 94
	specificity: 0.8620689655172413
	sensitivity: 0.8620689655172413
	ppv: 0.8620689655172413
	npv: 0.8620689655172413
Dataset: datasets/twogaussians.csv	Score: 0.9545454545454546
	tn: 63  fp: 2  fn: 4  tp: 63
	specificity: 0.9692307692307692
	sensitivity: 0.9692307692307692
	ppv: 0.9692307692307692
	npv: 0.9692307692307692
Dataset: datasets/twospirals.csv	Score: 0.6393939393939394
	tn: 102  fp: 83  fn: 36  tp: 109
	specificity: 0.5513513513513514
	sensitivity: 0.5513513513513514
	ppv: 0.5513513513513514
	npv: 0.5513513513513514
```

```
-- SVM Polynomial Kernel, no CV --
Dataset: datasets/clusterincluster.csv Score: 0.6303030303030303
        tn: 160  fp: 0  fn: 122  tp: 48
        specificity: 1.0
        sensitivity: 1.0
        ppv: 1.0
        npv: 1.0
Dataset: datasets/halfkernel.csv        Score: 0.7454545454545455
        tn: 125  fp: 40  fn: 44  tp: 121
        specificity: 0.7575757575757576
        sensitivity: 0.7575757575757576
        ppv: 0.7575757575757576
        npv: 0.7575757575757576
Dataset: datasets/twogaussians.csv      Score: 0.9772727272727273
        tn: 73  fp: 1  fn: 2  tp: 56
        specificity: 0.9864864864864865
        sensitivity: 0.9864864864864865
        ppv: 0.9864864864864865
        npv: 0.9864864864864865
Dataset: datasets/twospirals.csv        Score: 0.5909090909090909
        tn: 91  fp: 92  fn: 43  tp: 104
        specificity: 0.4972677595628415
        sensitivity: 0.4972677595628415
        ppv: 0.4972677595628415
        npv: 0.4972677595628415
```

```
-- SVM RBF Kernel, no CV --
Dataset: datasets/clusterincluster.csv Score: 1.0
        tn: 168  fp: 0  fn: 0  tp: 162
        specificity: 1.0
        sensitivity: 1.0
        ppv: 1.0
        npv: 1.0
Dataset: datasets/halfkernel.csv        Score: 0.9818181818181818
        tn: 161  fp: 6  fn: 0  tp: 163
        specificity: 0.9640718562874252
        sensitivity: 0.9640718562874252
        ppv: 0.9640718562874252
        npv: 0.9640718562874252
Dataset: datasets/twogaussians.csv      Score: 0.9772727272727273
        tn: 67  fp: 2  fn: 1  tp: 62
        specificity: 0.9710144927536232
        sensitivity: 0.9710144927536232
        ppv: 0.9710144927536232
        npv: 0.9710144927536232
Dataset: datasets/twospirals.csv        Score: 0.9606060606060606
        tn: 162  fp: 6  fn: 7  tp: 155
        specificity: 0.9642857142857143
        sensitivity: 0.9642857142857143
        ppv: 0.9642857142857143
        npv: 0.9642857142857143
```

3. Run the three classifiers with default parameters on the 4 datasets using 10-fold cross validation, obtaining, for each classifier, the averages of the five measures of efficiency seen in class: PPV, NPV, specificity, sensitivity, accuracy, where class 1 corresponds to “positive” and class 2 to “negative”.

```
-- SVM Linear Kernel with 10-fold CV --
Dataset: datasets/clusterincluster.csv Score: 0.59
Averages of the measures of efficiency are below:
        tn: 287.6  fp: 212.4  fn: 234.8  tp: 265.2
        specificity: 0.5751999999999999
        sensitivity: 0.5304
        ppv: 0.0025084227257535195
        npv: nan
        # Had a runtime error (invalid value encountered in long_scalars) when calculating the i'th npv.
        # Not sure what happened here.
Dataset: datasets/halfkernel.csv        Score: 0.8
Averages of the measures of efficiency are below:
        tn: 434.1  fp: 65.9  fn: 197.9  tp: 302.1
        specificity: 0.8682000000000001
        sensitivity: 0.6042000000000001
        ppv: 0.0027202738879497095
        npv: 0.6868184024769269
Dataset: datasets/twogaussians.csv    Score: 1.0
Averages of the measures of efficiency are below:
        tn: 199.3  fp: 6.7  fn: 4.7  tp: 189.3
        specificity: 0.9674757281553397
        sensitivity: 0.9757731958762885
        ppv: 0.0051022811133977905
        npv: 0.976988495431651
Dataset: datasets/twospirals.csv        Score: 0.73
Averages of the measures of efficiency are below:
        tn: 306.9  fp: 193.1  fn: 165.9  tp: 334.1
        specificity: 0.6138
        sensitivity: 0.6682
        ppv: 0.0018968664194045596
        npv: 0.6491186528159893

Total elapsed time: 0:00:48.473416
```

-- SVM Polynomial Kernel with 10-fold CV --
Note: This one seemed to take a long time with 10 fold cross validation, and the half-kernel dataset had been running for over an hour so I let it be. Left it to classify the twospirals dataset while I went to do other work, and it took over 3 hours to complete, if that’s any estimate of how long I would be waiting for half kernel.

```
Dataset: datasets/clusterincluster.csv Score: 0.68
Averages of the measures of efficiency are below:
        tn: 450.0  fp: 50.0  fn: 316.7  tp: 183.3
        specificity: 0.9
        sensitivity: 0.36660000000000004
        ppv: 0.006319657038890991
        npv: nan
        # Had a runtime error (invalid value encountered in long_scalars) when calculating the i'th npv.
        # Not sure what happened here.
Total elapsed time: 0:00:43.384040

Dataset: datasets/twogaussians.csv    Score: 1.0
Averages of the measures of efficiency are below:
        tn: 201.6  fp: 4.4  fn: 3.1  tp: 190.9
        specificity: 0.978640776699029
        sensitivity: 0.9840206185567011
        ppv: 0.005120490299977576
        npv: 0.9848704108354376
Total elapsed time: 0:00:01.714871

Dataset: datasets/twospirals.csv        Score: 0.67
Averages of the measures of efficiency are below:
        tn: 280.9  fp: 219.1  fn: 165.6  tp: 334.4
        specificity: 0.5618000000000001
        sensitivity: 0.6688
        ppv: 0.001813611148033466
        npv: 0.6286496279884635
Total elapsed time: 3:17:12.482946
```

```
-- SVM RBF Kernel with 10-fold CV --
Dataset: datasets/clusterincluster.csv Score: 1.0
Averages of the measures of efficiency are below:
        tn: 500.0  fp: 0.0  fn: 0.0  tp: 500.0
        specificity: 1.0
        sensitivity: 1.0
        ppv: 0.0020000000000000005
        npv: 1.0
Dataset: datasets/halfkernel.csv        Score: 1.0
Averages of the measures of efficiency are below:
        tn: 498.9  fp: 1.1  fn: 0.0  tp: 500.0
        specificity: 0.9978
        sensitivity: 1.0
        ppv: 0.001995613553768956
        npv: 1.0
Dataset: datasets/twogaussians.csv    Score: 1.0
Averages of the measures of efficiency are below:
        tn: 201.4  fp: 4.6  fn: 3.0  tp: 191.0
        specificity: 0.9776699029126213
        sensitivity: 0.9845360824742269
        ppv: 0.005112613613619124
        npv: 0.9853225314066769
Dataset: datasets/twospirals.csv        Score: 0.97
Averages of the measures of efficiency are below:
        tn: 494.3  fp: 5.7  fn: 8.3  tp: 491.7
        specificity: 0.9886000000000001
        sensitivity: 0.9833999999999998
        ppv: 0.0020105215443909 43
        npv: 0.9835055522668631

Total elapsed time: 0:02:45.448946
```

4. Compare and comment on the performances you obtained for the three classifiers on the 4 datasets. Provide valid reasons for justifying why the classification is better for some kernels than others on each particular dataset.

Since Polynomial kernels with cross validation kills my laptop, I'll be comparing linear kernel w/ CV, polynomial kernel w/ no CV, and RBF kernel w/ CV. I've only been able to get a final result with the cluster in cluster and two gaussians datasets. The other two datasets have shown no result even after an hour of running.

-- Linear Kernel --
It seems that the linear kernel is more or less only good for linearly separable, or near-linearly seperable datasets. In all of the 4 datasets, there are clearly many points which are misclassified. Only two gaussians dataset seemed to be close to seperable, however the performance was beaten strongly by the RBF kernel. None of the other datasets are linearly seperable, so I didn't expect for this classifier to perform well.

In my opinion, I would only use the linear kernel when first testing out the dataset (to see if it can be cleanly linearly seperable), or if I wanted an SVM classifier very quickly. The linear kernel took a total of 48 seconds to classify and plot all 4 datasets, which is a lot quicker than the other two kernels. It's quick and easy, but not quite a good fit for the use cases presented to us in these 4 datasets.

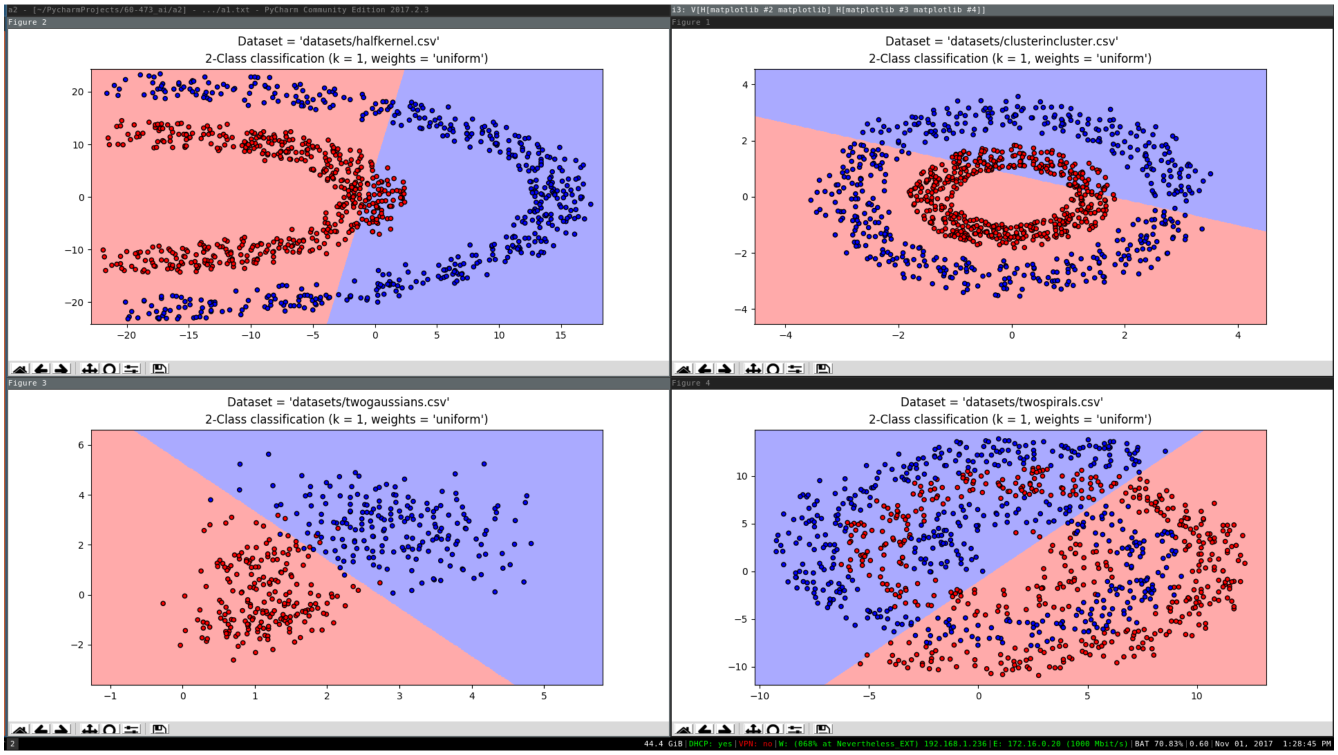

Illustration 1: 3_linearKernel.png
SVM-L with 10 fold cross validation on the 4 datasets.

-- Polynomial kernel (no CV) --
In contrast, the polynomial kernel had better, but still not stunning results. It was able to handle the two gaussians dataset a bit better than the linear kernel, but the rest of the datasets weren't anything to write home about. I was expecting the polynomial kernel to be able to properly classify all of the points in the half-kernel dataset since it's visually seperable by a quadratic line, however it came out with a strange pattern. I'm sure that this would have been resolved if I were able to wait long enough for polynomial kernel with 10-fold cross validation to work on this dataset.

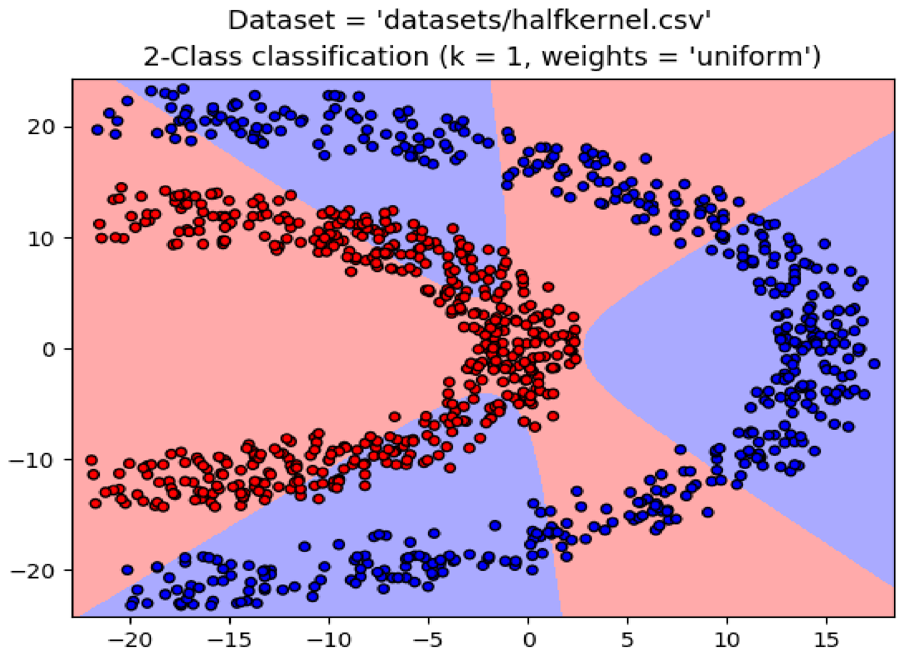

Illustration 2: 2b_poly_halfKernel.png
SVM with a polynomial kernel, no cross validation on half-kernel dataset.

Regardless of time spent building the model, this classifier would be better suited for the half kernel dataset since it's quadratically (?) sepearable. The other datasets would not perform well with this classifier, since they don't seem to cleanly separate in into side A vs side B.

In other news, I've been running the classifier with a polynomial kernel and 10-fold cross validation for about an hour, and I still don't have results for the half kernel dataset. I can only assume that the other datasets would take this long as well, which doesn't look promising to me. In all of the cases I ran the polynomial kernel classifier, the RBF classifier outperformed it. I would consider using the polynomial kernel if someone showed me that it was worth the time, but otherwise I think the time spent waiting for computation could be put to better use elsewhere.

-- RBF Kernel --
RBF seems to be the best choice for all of the databases to me. The shape of the decision boundary resembles the k-nearest neighbors decision boundary, which also did very well on all of the datasets. With average scores of 97%-100%, this is a strong kernel, for all of the datasets. My one concern with this kernel, similar to k-NN, is that new points that are visually far from the negative class may get falsly classified as such. For example, the half-kernel classifier has some gaps in the middle of the positive area, which could very well be the location of a new positive point. For that reason, I would consider using a polynomial classifier on the half kernel dataset.

The RBF kernel took a bit under 3 minutes to handle all 4 datasets, which was refreshing after the experience with the polynomial kernel. I would consider using this in a real life application right after the linear kernel, as it's a quick and strong classifier.

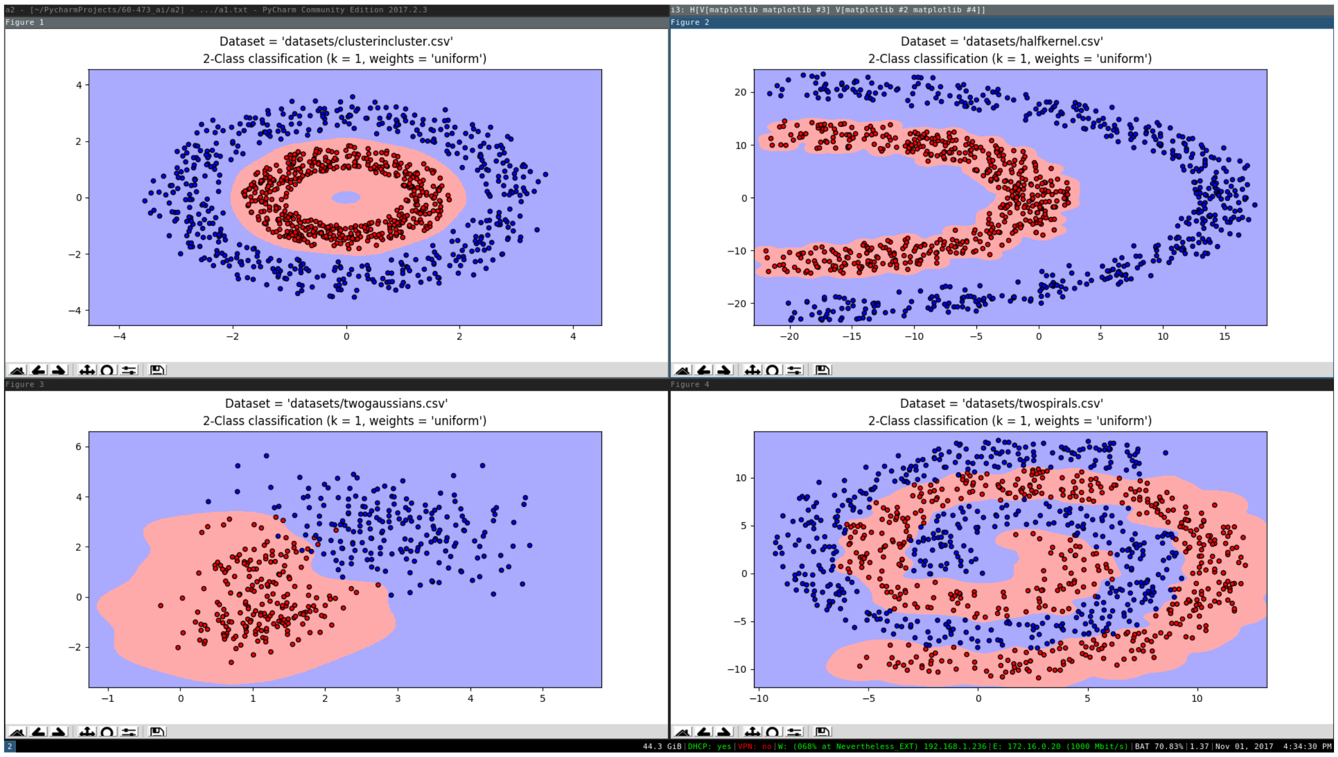

Illustration 3: 3c_rbfKernel.png
SVM-R classifier with 10 fold cross validation on the 4 datasets

5. For SVM-R, plot the ROC curve and find the AUC for each dataset. Note: for constructing the ROC curve, you can run SVM-R with different parameters. Eventually, you can apply grid search to obtain the best parameters (not required for this assignment).

I wasn't certain about what parameters I should be testing out for this one, so I stuck with showing the ROC curve for the 10 different K-folds in cross validation, like in the example I went off of to build the ROC curve. From taking a look at the documentation for SVM (http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html), it seems that the and class weight and gamma value are the only related parameter.

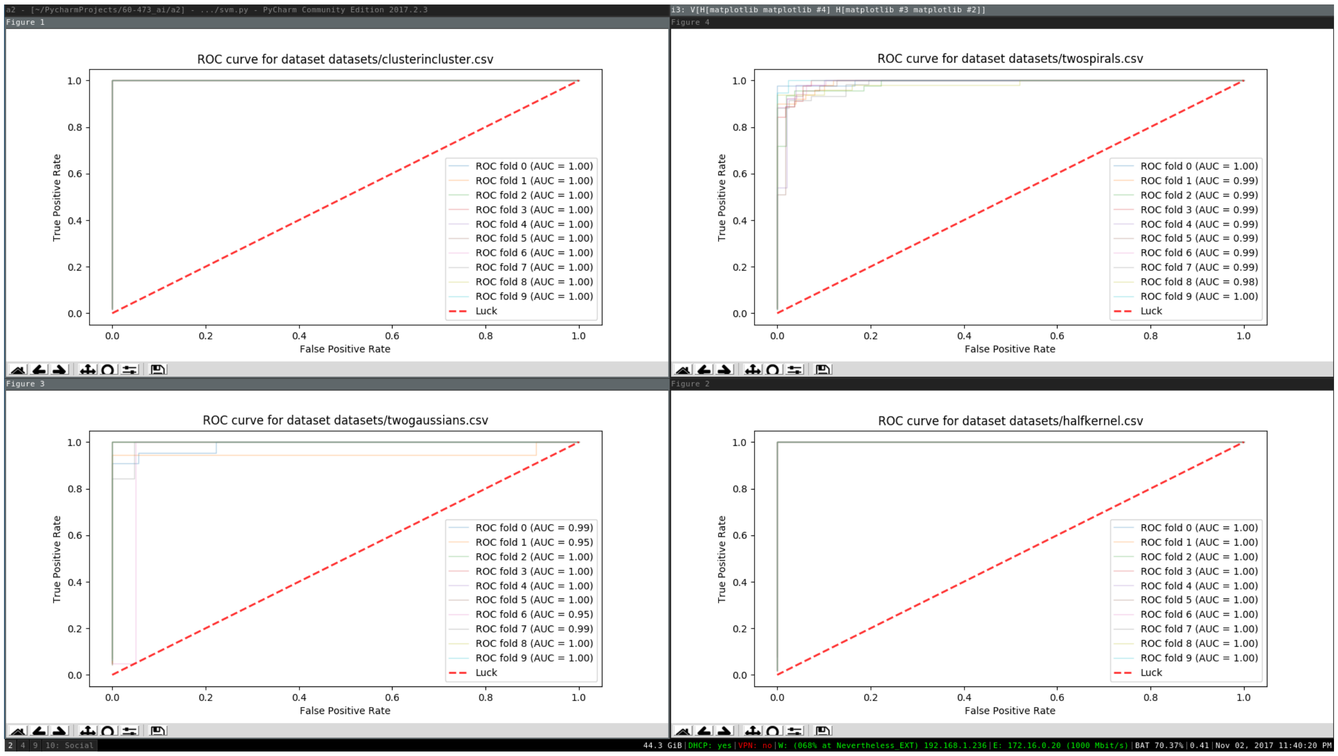

Illustration 4: 5_rocCurves.png
ROC Curves for SVM-R. Each of the 10 lines represents an ROC curve for each of the 10 folds of a 10-fold cross validation on each of the 4 datasets.

6. Compare and comment on the performance (both accuracy and AUC) of the classifiers.

Overall, the SVM classifiers worked fairly well. I didn't expect the linear kernel to perform, but it did build quickly. The polynomial kernel was dissapointingly slow when cross validation was used, and the performance was pretty bad. The RBF kernel however, worked very well. With scores either at 100% or at 97%, this classifier was able to handle whatever dataset I threw at it.

Extra
-- Comparison of SVM vs k-NN --

The scores and plots of k-NN are below, for reference. As an aside, my cross validation was broken in assignment 1 because I didn't shuffle the dataset. This is why there are errors showing that there only one class in the test set. This has been fixed in the current assignment

```
-- K-nearest Neighbors, cross validation --
Dataset: datasets/clusterincluster.csv  Score: 1.0
    Test set contains one class only. There is no false positive or false negative; Only the one class.
Dataset: datasets/halfkernel.csv    Score: 1.0
    Test set contains one class only. There is no false positive or false negative; Only the one class.
Dataset: datasets/twogaussians.csv  Score: 1.0
    tn: 6  fp: 0  fn: 0  tp: 34
    specificity: 1.0
    sensitivity: 1.0
    ppv: 1.0
    npv: 1.0
Dattaset: datasets/twospirals.csv   Score: 0.95
    Test set contains one class only. There is no false positive or false negative; Only the one class.
```

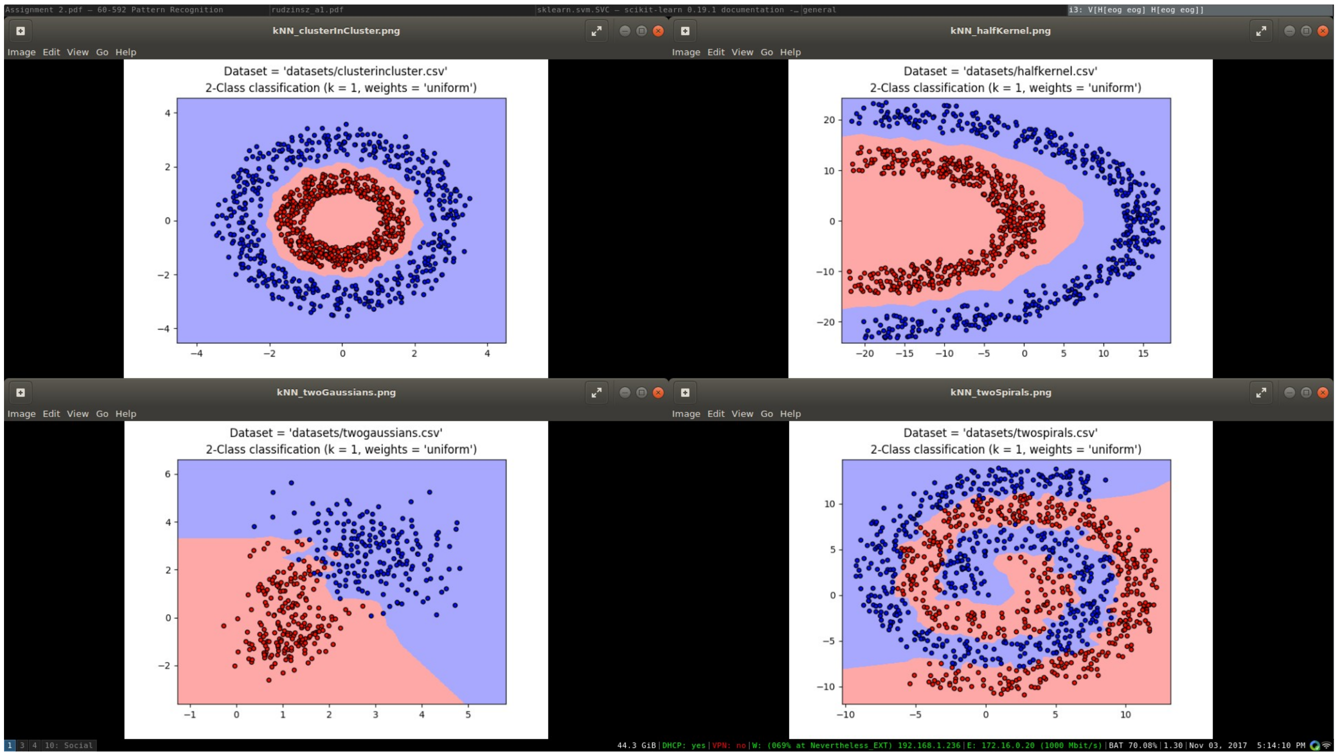
kNN_clusterInCluster.png
Image Edit View Go Help
Dataset = 'datasets/clusterincluster.csv'
2-Class classification (k = 1, weights = 'uniform')
kNN_halfKernel.png
Image Edit View Go Help
Dataset = 'datasets/halfkernel.csv'
2-Class classification (k = 1, weights = 'uniform')
kNN_twoGaussians.png
Image Edit View Go Help
Dataset = 'datasets/twogaussians.csv'
2-Class classification (k = 1, weights = 'uniform')
kNN_twoSpirals.png
Image Edit View Go Help
Dataset = 'datasets/twospirals.csv'
2-Class classification (k = 1, weights = 'uniform')

Illustration 5: extra_a1_kNN.png
The plots for my k=1 k nearest neighbors solution from assignment 1.

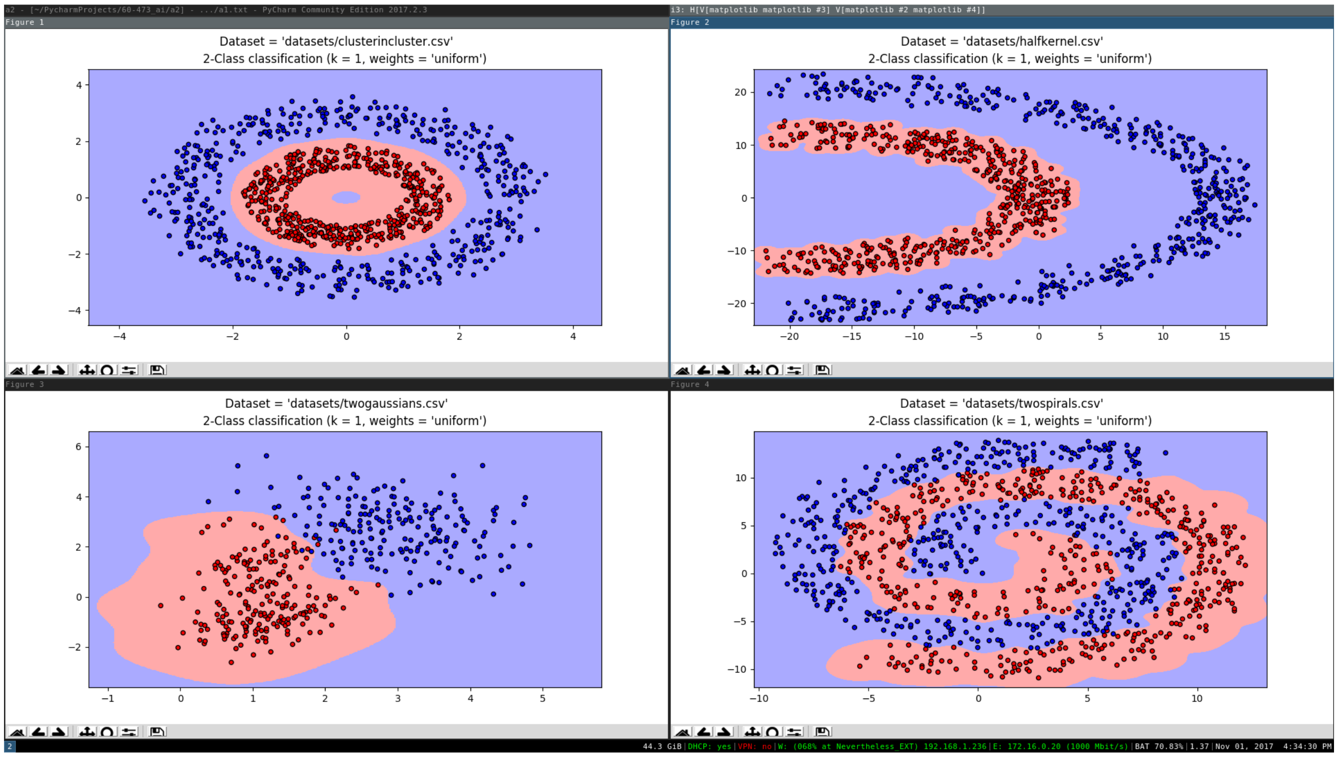

Illustration 6: 3c_rbfKernel.png
For quick reference, my plots for 10-fold cross validation with the RBF kernel from the current assignment.

k-NN seems to have the most similarity to SVM when the RBF kernel is applied. This is mainly true for the two spirals, and cluster in cluster. If the polynomial kernel finished running when cross validation was applied, I can predict that it would have a similar shape as the k-NN decision boundary, albeit smoother.

It seems that the SVM-R classifier is stronger than the k-NN approach, based on their scores. They had the same perfect score for cluster in cluster and half kernel, but SVM-R beats out k-NN for two gaussian and two spirals (Gaussian_k-NN: 96% vs Gaussian_SVM-R 100%, and TwoSpirals_k-NN: 92% vs TwoSpirals_SVM-R: 97%). That being said, my cross validation in assignment 1 was pretty broken, so this might have had an impact on scores and such.