PROJECT: KERNELIZATION, KERNEL TRICKS

Project: **Kernelization, Kernel Tricks**

1. Check the Assignment Schedule for the DUE date.
2. For each Exercise, submit the script, the output of running this script, and the answers to the questions (if applicable).
3. Submit a single html file via Moodle (instructions below).

Programming Environment: Code your script using Python. You will likely want to use the scikit-learn package as well as numpy. Scikit-learn will give you access to various data mining functions, such as KMeans. Numpy is a generic package for scientific computing, which may be useful for dealing with matrices, for example.

Project Details:
For this project, we will using IPython notebooks. IPython notebooks can be thought of a more advanced interactive shell where you can intermix regular text, code snippets, and the results from that code. This makes it a really powerful tool for interactively exploring the data, trying different data mining techniques, and taking notes.

To run an IPython notebook, run the following command:

```
$ ipython notebook
```

A new window should be opened in your browser. From here you can open an existing IPython notebook or create your own notebook. Open the IPython notebook included with this project (the .ipynb file), there will be instructions there.

Once you have finished all your code and have answered all your questions, you will need to convert your notebook to an html file for submission. To do this run the following command:

```
$ ipython nbconvert P3.ipynb --to html
```

Exercise 1: ***Generating the data sets***. Write a script that generates three data sets in a 2-dimensional space, defined as follows:

1. **BAD_kmeans**: a data set for which the *k-means* clustering algorithm will not perform well.
2. **BAD_pca**: a data set for which the Principal Component Analysis (PCA) dimension reduction method upon projection of the original points onto 1-dimensional space (i.e., the first eigenvector) will not perform well.
3. **BAD_svm**: a data set for which the *linear* Support Vector Machine (SVM) supervised classification method using two classes of points (positive and negative) will not perform well.
4. Plot each data set in a 2-dimensional space.

Exercise 2: ***Evaluating the "badness" of the data mining methods***. Write a script that uses the BAD data sets in Exercise 1, runs the corresponding data mining method, produces the output of the method, and evaluates the performance of the method using (e.g., 10-fold) *cross-validation* and various performance metrics (e.g., variance, precision, recall, F1 measure). Note that not all metrics can be equally applied to every method. Report the performance metrics used and a summary of the results obtained. Reading the chapter "*Performance Metrics for Graph Mining Tasks*" by Kanchana Padmanabhan and John Jenkins in the resources for performance metrics is strongly encouraged for performing this exercise.

Exercise 3: ***Kernelizing the methods.*** Write a script that uses the *kernelized* version of each of the data mining methods in Exercise 2.

a. Choose at least two kernels for each method.
b. Use the same performance metrics as in Exercise 2 and compare the performance of the original un-kernelized version of the method versus the performance obtained after applying the kernel trick.
c. Do you observe a difference in performance when you use different kernels?
d. What are the best performance results you obtain by trying different kernels and kernel parameters? Also, make sure to report the number of support vectors for the SVM (a good rule of thumb is to strive for no more than 35%-50% support vectors to avoid model overfitting).

Exercise 4: ***Pipelining***. Dimension reduction is often used as a key data pre-processing step for other data mining methods downstream the end-to-end data analysis. In this exercise, you will use unsupervised *kernel* PCA as a pre-processing step for clustering. Later in the course, we will use *supervised dimension reduction methods* as a pre-processing step for supervised classification methods.

a. Generalize your BAD_kmeans data set to very high-dimensional space (d>>2).
b. Show that the *k-means* clustering algorithm does not perform well on that data.
c. Apply the *kernel* PCA method to this high dimensional data set and identify the number ($m<<d$) of principal components (i.e., eigenvectors) that provide a reasonably good low-dimensional approximation of your data (i.e., based on eigenvalue distribution). How much total variability of the data will be preserved upon using this low-dimensional representation?
d. Project your original data onto the top m eigenvectors corresponding to the largest eigenvalues.
e. Run the *k-means* clustering algorithm on the projected low dimensional data.
f. Compare the performance of the *k-means* clustering algorithm on the d-dimensional original data vs. the m-dimensional projected data. Has the performance improved?
g. If you run the *kernel k-means* clustering algorithm on the original data, will you get better or worse performance? Discuss the pros and cons of using *kernel k-means* on the original data directly versus applying *kernel* PCA as a pre-processing step and then running the *k-means* clustering algorithm on the low-dimensional data.