# ECS607/766 Data Mining 2017/18

*Assignment 4: Dimensionality & Feature Selection*

## Introduction

The outcome of this lab is to get experience with manipulating the dimensions (columns) of high dimensional data, including feature selection and dimensionality reduction. This lab will use both matlab and weka. Start by downloading the zip file of lab materials from qmplus. Questions in ***red*** are assessed toward your final grade. You **MUST** hand in this sheet with your written answers to the TA by the end of the session.

## 1. Feature Selection

1. Invoke weka. Load the Labor.arff dataset.
   - This is about analyzing potential contracts debated between unions and management according to whether they were accepted or rejected by the parties debating. This is a useful capability because contracts could be checked automatically for credibility before wasting time debating.
2. Select each of the five attributes in the explorer. (The sixth is the class itself)
   - Which looks the most informative about class? (The colors=class in the histogram appear to overlap the least)? Which looks least informative about class?
3. Let's walk through backward feature selection.
   - Start by running Nearest Neighbor classification (Classifier => Lazy => IBk). Make sure 10-fold cross-validation is selected. Note the attributes used, and the classification accuracy.
   - Recall that backward feature selection greedily removes features to try and optimize cross-validation accuracy.
   - Remove each of the five attributes in turn (Tick box & Remove on the Preprocess screen) and rerun the NN classifier. Note the classification accuracy with each attribute removed.
     i. Note that you can hit undo after each removal and classify. And note that you can review the list of results on the classify screen.
   - Which attribute, when removed, gave the best accuracy?
   - This defines the best 4-attribute subset. What is it?
   - Now starting from this 4-attribute subset, find the best 3,2,1 attribute subset, filling in the table below.
   - ***Which sized subset, and which set of attributes gets the best accuracy?*** [1 mark]

| Subset Size | Attributes Selected | Accuracy | Attribs Removed |
|---|---|---|---|
| 5 | All: W, P, Hol, Vac, Health | 85.9% | None |
| 4 | | | |
| 3 | | | |
| 2 | | | |
| 1 | | | |

   - Its important to note that greedy feature selection does not try all combinations of features, so may not find the absolute best combination.
   - How many feature combinations did you try? How many combinations of features are there in total?
   - Give an example of a combination of features did you NOT try when doing backward selection?

4. Now let's explore ways to automate feature selection.
   - Go to Select Attributes. Evaluator => Information Gain. Search => Ranker.
   - This filter method ranks each attribute by their information provided about class.

- Which attribute was ranked most and least highly? Did those correspond to the first one eliminated and last one surviving in your backward selection?
- Compare the results for Chi squared (correlation based) ranking. Does it generally predict the same attributes as Information Gain?
- Now let's try to replicate the backward feature selection experiment. Select Evaluator => WrapperSubset, and Search=>GreedyStepwise. Set the Evaluator parameter (box to the right of Attribute-Choose to 10-fold CV as in Q3, and select KNN classifier (Lazy-IBk)). Set the Search method parameter (box to the right of Search-Choose) to backward feature selection. Make sure that Attribute Selection Mode box is set on Use Full training set.
- ***How many and which attributes are selected? Do they match the results from Q3?*** [1 mark]
- Try also forward selection? (Search Method options => SearchBackwards => False). Are the same set of attribute selected? If not, how is it possible for forward selection to get a different answer than backward selection?

5. Load the Iris flowertype data. (We explored it in matlab in previous lab, Open => iris.arff)
   - Run Naïve Bayes classifier (Classifier->Choose->Bayes->NaiveBayes) and note the performance. Run the logistic regression classifier (Choose->Functions->Logistic) and note the performance.
   - Recall that one drawback of Naïve Bayes is that if the feature independence assumption is not met, then performance may be significantly compromised.
   - To observe this, make 3 copies each of sepal length and sepal width.
     - i. (Filter => Unsuperv.=> Attribute => Copy, Parameters => Index [1,2] => Apply. )
     - ii. (Filter => Unsuperv.=> Attribute => Copy, Parameters => Index [6,7] => Apply. )
     - iii. (Filter => Unsuperv.=> Attribute => Copy, Parameters => Index [8,9] => Apply. )
     - iv. Now our database has 11 columns instead of 5.
   - Now re-try the classifiers NaïveByes, and Logistic. (Make sure 'class' is still set as the target attribute for the classifier) How do they each perform?
   - Now try to use the attribute selection tab to find the right attributes for naïve Bayes given the modified dataset with redundant columns
     - i. (Choose => Wrapper, Parameters: Classifier => NaïveBayes, Search => GreedyStepwise)
   - ***Which attributes does it pick (and hence which are discarded?)*** [1 mark]
   - Remove the rejected attributes and re-run NaiveBayes and Logistic classifier. How do they each perform now compared to the 11 column version, and to the initial four column version? What does this mean?

## 2. Dimensionality Reduction

1. Start matlab. Edit lab5.m
2. Run the first cell. A face database is loaded and some example faces are shown.
   - (Note that in matlab you can view any matrix as in image by `imagesc(variable)`, but you may need to reshape it first if the matrix has been flattened into a vector `imagesc(reshape(variable,[x,y]))`)
   - The loaded face database is matrix "faces". Try **size(faces)**. You will see that it has 165 rows (face images), and 4096 columns (pixels – one column per pixel).
3. Run the second cell. The SVD of data is taken. The 'U' output of SVD is the eigenvectors of the data covariance required for PCA. These shows the basis vectors / prototype faces in terms of which all other faces are to be encoded. (Every face will be a linear combination of these prototypes) Notice that some basis vectors encode lighting, others features such as glasses.
4. Cell 3 shows you how to take the PCA encoding of each face, and then decompress it back.
   - Recall that for a D-column input matrix/database, the goal of PCA is to find a K<=D-column encoding that most accurately encodes the data in D.

- You can see the original database faces, and the compressed database pcaFaces. You can compare their size with **size(faces)** and **size(pcaFaces)**. You can see the compressed version has only K=25 columns compared to the original D=4096 columns.

5. Starting from 1 dimensional encoding, increase the dimensions (set variable nPCA) and re-run the cell, observing how the facial encoding fidelity increases. At what number of encoding dimensions can you see features like glasses and facial expression?

6. How many PCs do you need to reach an encoding fidelity of 99%? (<1% reconstruction error). Using the **whos** command in matlab, compare the size in bytes of original and PCA data at this point.

7. Note that when using the full number (4096) of PCs, the encoding fidelity is 100%. Using all the PCs conveys exactly the same information as the original data.

8. The PCA process produces eigenvectors and eigenvalues. The eigenvectors give the new basis (~define the new database columns), and the eigenvalues explain how useful each column is for encoding the data.

    - Look at the eigenvalues of the basis. (Try **`plot(eigvals);`**) **.** The x-axis is the new PCs/database dimension, and the Y-axis is the eigenvalue / information content.

    - The first few dimensions are by far the most informative.

    - Make this a cumulative plot to see how variance each new dimension encodes, or how much each new dimension contributes to the reconstruction fidelity. Try:
      **`plot(cumsum(eigvals)/sum(eigvals)); ylim([0,1]);`**
      **`xlabel('Dimensions'); ylabel('Reconstruction Accuracy');`**

    - ***Use the data cursor on the plot to find out what # of PCs is required to explain 99% of the data variance (achieve 99% reconstruction accuracy). What # is this and does it match the value from Q2.6?*** [1 mark]

    - What does this tell you?

9. Now lets try to recognize the faces using a simple KNN classifier.

    - There are 15 people in this dataset. Cell 5 splits the data into train and test.

    - Change nPCA. Observe that classification using different numbers of PCA dimensions produces different results.

    - ***Which number of PCA dimensions gets maximum face recognition accuracy? Is it better or worse than accuracy classifying the raw images? *** [2 marks]

    - Why? (What factors contribute to this?)

Note: Accuracy was used in two different contexts in the above: (i) reconstruction accuracy in 2.1-2.8 (Unsupervised learning task: How accurately an image is encoded after compressing away some of the columns with PCA) , and (ii) face recognition accuracy in 2.9 (Supervised learning task: How accurately can we recognize faces given raw image or PCA compressed image).