ASSESSMENT COVER SHEET

Assignment Details

Course: 3920_COMP_SCI_X_0009

Semester/Academic Year: Semester 2 - 2019

Assignment title: Assignment 3

Assessment Criteria

Assessment Criteria are included in the Assignment Descriptions that are published on each course's web site.

Plagiarism and Collusion

Plagiarism: using another person's ideas, designs, words or works without appropriate acknowledgement.

Collusion: another person assisting in the production of an assessment submission without the express requirement, or consent or knowledge of the assessor.

Consequences of Plagiarism and Collusion

The penalties associated with plagiarism and collusion are designed to impose sanctions on offenders that reflect the seriousness of the University's commitment to academic integrity. Penalties may include: the requirement to revise and resubmit assessment work, receiving a result of zero for the assessment work, failing the course, expulsion and/or receiving a financial penalty.

DECLARATION

I declare that all material in this assessment is my own work except where there is clear acknowledgement and reference to the work of others. I have read the University Policy Statement on Plagiarism, Collusion and Related Forms of Cheating:

http://www.adelaide.edu.au/policies/?230

I give permission for my assessment work to be reproduced and submitted to academic staff for the purposes of assessment and to be copied, submitted and retained in a form suitable for electronic checking of plagiarism.

MAX VERHAGEN - 7/11/2019

SIGNATURE AND DATE

Algorithmic description

Principal component analysis can be mathematically defined as an orthogonal linear transformation that is used to reduce the dimension of a large features dataset whilst keeping the largest amount of variation within the dataset. From the training data this transformation is then mapped and can be applied to other data using the created Mean vector and Projection matrix.

Firstly, in terms of PCA, when the data is innately loaded the mean of each feature is calculated into a matrix set, these averages then can be reapplied to the data in order to centralize it around the zero axis. If this average is not subtracted, then the first principal component regression line may correspond to the centre of the data set and/or not split the data along the main direction of the data spread.

Once centred a covariance matrix is created, showing the variation between two features. This is calculated using the formula $cov(x,y) = \frac{1}{n}\sum_{i=1}^{n}(X[i]-x)(Y[i]-y)$ which will create a symmetric matrix insuring that the eigenvectors created in the future will be real and non-negative. For example, the creation of a covariance matrix from a dataset this three features results in the following:

$$\begin{bmatrix} a1 & b1 & c1 \\ \dots & \dots & \dots \\ \dots & \dots & \dots \\ \dots & \dots & \dots \\ ai & bi & ci \end{bmatrix} \rightarrow \begin{bmatrix} a & a,b & a,c \\ a,b & b & b,c \\ a,c & b,c & c \end{bmatrix}$$

Eigenvalues are defined as the roots of the characteristic equation $\det(A - \lambda \mathrm{I}) = 0$. The equation is solved giving eigenvalues of λ, these values are then sorted from largest to smallest. The higher the value of the eigenvalue, the more information it is containing about the spread of data. Once sorted, the values can be divided by the sum for values to give the percentage coverage of variation. The selected eigenvalues are then placed back into the characteristic equation to produce eigenvectors. An eigenvectors direction stays unchanged once a linear transformation is applied to it.

Lastly the transpose of the newly created eigenvector matrix is projected onto the original sample dataset to create a new subspace containing the x-axis and y-axis location on the new PC1 vs PC2 plot. Essentially converting our multi-dimension data into a lower dimensional plain.

Understanding of PCA and KPCA

Alternatively, to PCA, kernel Principal component analysis which is less generally computationally expensive. Instead of data being directly centralized, the product of two samples are calculated by a kernel function. This new created kernel matrix is then normalized into a square matrix and used to once again find the eigenvalues. This is especially useful when dealing with non-linear separable data.

There are three main kernels used within this assignment these being a linear kernel, Gaussian-RBF kernel and Polynomial Kernel. Traditionally speaking a linear kernel is equivalent to the basic principal component analysis.

The Gaussian-RBF kernel also know as the radial basis function kernel where the value is determined by distance between the input and a fixed point. This kernel can be formulated by the following equation:

$$K(x, x') = \exp\left(-\frac{\left||x - x'|\right|^2}{2 * sigma^2}\right)$$

By contrast a polynomial kernel produces the delineation of the similarity of features across samples in refence to polynomial space that wouldn't be normally achieved with linear learning models. This produces an effect similar to polynomial regression without the excessive oversizing of values. This can be mathematically defined as:

$$K(x, y) = (x^T y + c)^d$$

where within this assignment d is set to two and c which offsets the influence of higher-order or low-order polynomials. However when c=0 the kernel is considered homogeneous.

Overall PCA and KPCA finds the variables that captures the largest variability in data to allow for visualization of data in addition to making computation easier due to reducing dimension size of the dataset by identifying the most relevant directions of variance.

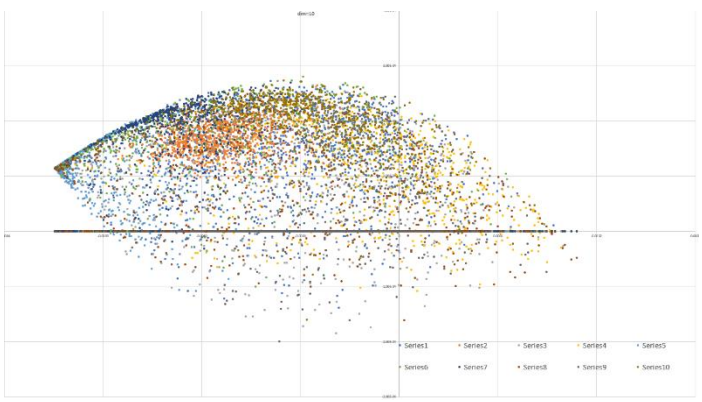
Analyses of implementation.

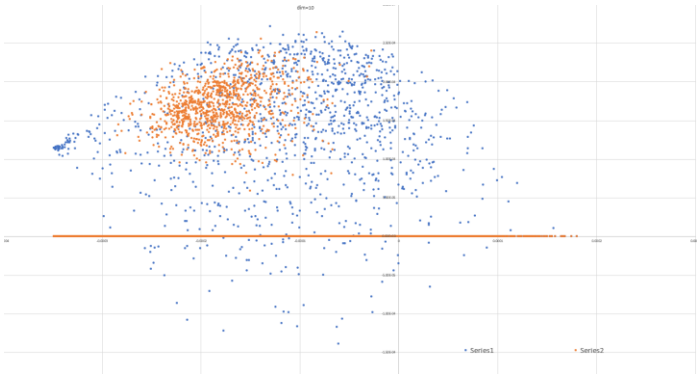
The command to compile the code is as following: **g++ -std=c++11 mian.cpp -o program** . Unlike both python and matlab, c++ does not always have the default installed library for calculating eigenvalues and eigenvectors, this error can be fixed by using: sudo apt install libeigen3-dev

No code needs to be modified in order to change tests dimension size or to switch between PCA and KPCA this is done purely through arguments. **./program train.cvs 11 0 test.cvs**

The program takes up to four arguments, by default when the only argument given is the train.cvs file then the dimension will be set to 256 and the system set to PCA. The dimension set can be changed by inputting a second argument value which is set as 11 in the above example. To change the Kernel type, the third argument is changed; 0=linear, 1=Gaussian-RBF kernel, 2=Polynomial kernel with d=2 permanently set. The last argument is the test file name.

Although desired c++ does not include any graphic default API therefore the output is placed into a csv file which can then be graphed using excels graphing tools. Graphing them all onto the same graph. **Please note the multiple points across the x-axis is a csv graphing error and not an error within the code and therefor should be ignored.** First row On the left is when the dimension is set to 10 and on the right is set to 256. The second row shows the same dimensions however with only two different classes displayed.





Within the system itself it follows the basic algorithmic steps broken down into classes. Firstly the csv is converted into a 2d vector and then run thought the selected form of PCA/KPCA. Throughout the running of the system versus comments are posted in command Cout to communicate the progress of the principal component analysis.