DATA.STAT.770 Dimensionality Reduction and Visualization, Spring 2023, Exercise set 4

Part B: Linear Dimensionality Reduction, Continued

Problem B3: Principal Component Analysis of an Image

The file staircase.png contains a grayscale picture of the grand staircase of the RMS Olympic ocean liner; the RMS Olympic was owned by the White Star Line and was a sister ship of the Titanic. The image is 240 pixels wide and 200 pixels tall. Let's try using PCA to compress this image, just like was shown on the lecture.

A template code for this exercise, containing the parts unrelated to statistical analysis, is provided in this package in two programming languages, R (file image_compression_and_components_template.R) and Matlab (file image_compression_and_components_template.m).

1. Divide the image into 10 by 10 pixel blocks, so that the first block contains pixels in rows 1-10, columns 1-10, the second block contains pixels in rows 1-10, columns 11-20, and so on. Each block contains 100 pixel values and can be thought of as a 100-dimensional input vector, in total you have $24 \times 20 = 480$ such 100-dimensional input vectors.

2. Compute the first PCA component of this data set (480 items, 100 features) and project each input vectors onto that component - the result is one scalar value per input vector. Then project the scalar values back as a reconstruction of the origal features, as described on the lecture - the result is one 100-dimensional vector per input vector. Draw the resulting picture: for each pixel block, instead of the original pixel values, draw the approximated values.

3. Do the same for 2 components, 5, 10, 20, and 30 PCA components.

4. The PCA projection directions themselves are 100-dimensional vectors, and each element of the vectors corresponds to one pixel in the 10 by 10 image blocks. Therefore, each projection direction itself can be represented as a 10 by 10 block: it is like a filter for the image, showing which pixel values contribute positively to that principal component (high projection coefficients) and which pixel values contribute negatively (low projection coefficients). The provided template code also draws these filter images. Analyze the filters: have they identified important features of the image? Where do the features occur most strongly in the image?

Problem B4: Principal Component Analysis of an Audio File

Principal Component Analysis can be used to compress many kinds of data, not just images. The file the_entertainer.wav contains a part of "The Entertainer", a well-known 1902 piano rag by Scott Joplin (original audio available at `http://freemusicarchive.org/music/Scott_Joplin/Frog_Legs_Ragtime_Era_Favorites/04_-_scott_joplin_-_the_entertainer`). Just like an image can be divided into spatial blocks, audio waveforms can be divided into blocks over time; each block is a high-dimensional vector of numbers, and the resulting set of high-dimensional blocks can be reduced to a lower dimensionality by PCA.

The template codes musiccompression_template.R (R) and musiccompression_template.m (Matlab) contain code to load the audio file and to transform it into a feature data set of blocks. Your task is to reduce the dimensionality of the feature data set to 15 principal components, and then reconstruct the original data from the principal components, just like was done for the image in problem B3. The template code will then arrange the reconstructed data as a reconstructed audio waveform which can be plotted as a time series, and can also be played just like the original music. You do not need to play the file, but plot it, and save it to a file; provide it as part of your answer.

Note: if you choose to play any reconstructed audio, please do so at low volumes only! In case the audio waveform is not reconstructed well, it can contain clicks or scratches or other irritating or loud sounds which could harm your hearing if played overly loudly.

Problem B5: Independent Component Analysis for Separating Audio Mixtures

Consider a cocktail party situation where different microphones in the room record different mixtures of the ongoing independent audio sources in the room; the mixture that a microphone records is affected for example by how close the microphone is to each audio source.

a) Download and get to know a software package for independent component analysis. For example, use the FastICA software package; it is available at `http://research.ics.aalto.fi/ica/fastica/` for several programming languages such as Matlab, R, Python and C++. You can also use any other ICA software package.

b) The exercise pack includes four WAV audio files 'musicmix01.wav', 'musicmix02.wav', 'musicmix03.wav' and 'musicmix04.wav'. They each are a different mixture of four independent audio sources (same sources in each file, but mixed in a different way).

   Suppose that we are in a noisy environment where several different music pieces are being played at the same time. We want to separate the different pieces of music, for example to identify them. We will use independent component analysis (ICA) for this purpose.

Load the audio files as a time series data set (4 dimensions = sources recorded at each time instant). Plot these four mixed time series (waveforms). Use the ICA software package to recover the four original sources, and plot the source time series (waveforms).

Each of the four sources are public domain music recordings from `http://freemusicarchive.org`. The html file 'songpossibilities.html' in this exercise pack is a webpage that lists several possible songs, with direct links where you can listen to the songs. That webpage contains many more music tracks than what are used here. Compare the original sources recovered by ICA to the music tracks on the webpage (you can simply listen to the available music tracks on the webpage). Which four music pieces on the webpage correspond to the original sources?

Hint: the sources are all from near the beginning of the music tracks on the webpages, no need to listen all the way through each track.

Hint: in Matlab the command 'wavread' can be used to read in each audio file as a vector of amplitudes; this vector is suitable as a source time series. The command 'wavwrite' can be used to write the recovered sources as audio files.

c) Now consider another situation where four people are speaking at the same time, and each microphone records a different hard-to-comprehend mixture of the speakers. This is simulated in the audio files 'speechmix01.wav', 'speechmix02.wav', 'speechmix03.wav', and 'speechmix04.wav'. Apply ICA to separate the different speakers. What are they saying?

Hint: they are readings from four out of the following classic books available at Project Gutenberg, but which ones? The possible books are: "Alice's Adventures in Wonderland" by Lewis Carroll, "A Tale of Two Cities" by Charles Dickens, "Dracula" by Bram Stoker, "Frankenstein; Or, The Modern Prometheus" by Mary Wollstonecraft Shelley, "Grimms' Fairy Tales" by Jacob Grimm and Wilhelm Grimm "Heart of Darkness" by Joseph Conrad, "Moby Dick; Or, The Whale" by Herman Melville, "Pride and Prejudice" by Jane Austen, "The Adventures of Sherlock Holmes" by Arthur Conan Doyle, "The Adventures of Tom Sawyer" by Mark Twain, "War and Peace" by graf Leo Tolstoy.

Problem B6: Independent Component Analysis for Compressing Images

In problem B3, a template is provided for compressing 10 by 10 blocks of an image using principal component analysis. That template code can also be used with Independent Component Analysis. In this exercise we try to use independent component analysis for the same image as in B3.

- Download and get to know a software package for independent component analysis. For example, use the FastICA software package; it is available

at `http://research.ics.aalto.fi/ica/fastica/` for several programming languages such as Matlab, R, Python and C++. You can also use any other ICA software package.

- Form the 480 by 100 feature data matrix as in B3.
- Compute the mean of the data (a 1 by 100 vector), and substract the mean from the data.
- Give the resulting zero-mean feature data matrix as an input to an independent component analysis software, e.g. the same software as in B3. Use the software to extract the first 20 independent components (sources), which will be a matrix of size 480 by 20; the software should also provide their corresponding mixing matrix $\mathbf{W}$ which is a matrix of size 20 by 100.
- Project the sources with the mixing matrix, to form an reconstruction of the feature data matrix.
- Add the mean back to the reconstructed data.

The above steps can be inserted into the template code of B3, instead of the PCA solution, so that the ICA-reconstructed feature data matrix and the ICA mixing matrix are used in place of the PCA-reconstructed feature data matrix and the PCA principal components matrix, respectively.

Compare the ICA reconstruction to the PCA reconstruction with 20 components that you computed in B3; which one looks "better" to you? What about the resulting filter images, is there a difference?

Problem B7: Linear Discriminant Analysis

Consider the white wines data set used in Problem B2, which has wine varieties and their quality ratings (low, medium, or high). Suppose we want to use Linear Discriminant Analysis to reduce the data dimensionality to 2 dimensions in a way that tries to discriminate the classes.

a) Either implement Linear Discriminant Analysis (LDA; as discussed in Lecture 3, slide 21) in a programming language of your choice, or become familiar with a software package that implements it. Note that for the wine data, which has more than two classes, you need the multi-class case of LDA, the two-class case is not suitable here.

   Hint: if you implement LDA yourself, in Matlab the generalized eigenvalue problem $\Sigma_b \Phi = \lambda \Sigma_w \Phi$ can be solved by the function 'eig(Σ_b, Σ_w)'. If you use Octave instead of Matlab, note that the function 'eigs' is buggy in some versions and 'eig' is thus preferable.

b) Apply LDA to reduce the wine data to two dimensions. Plot the resulting two-dimensional data set as a scatter plot (color each data point by the class of the point). Does the result separate classes well? Does it separate them

better than PCA in Problem B1? (Note that this is a difficult problem and you should not expect very clear separation.)

c) Analyze the resulting projection matrix: which original features have the greatest influence on the projected coordinates?