Introduction to Data Science: Assignment 2

Eelke Landsaat (s4056868)
Jesse Reurink (s3771385)
Darragh Spillane (s5270855)

Group 10

September 21, 2022

2.1.1

Eigenvalues and eigenvectors We have the 2×2 matrix A;

$$A = \begin{bmatrix} 3 & 4 \\ 5 & 8 \end{bmatrix}$$

To calculate the eigenvalues, we first multiply λ with the identity matrix $I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$:

$$\lambda I = \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}$$

Then, we subtract this matrix from A:

$$A - \lambda I = \begin{bmatrix} 3 & 4 \\ 5 & 8 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} = \begin{bmatrix} 3-\lambda & 4 \\ 5 & 8-\lambda \end{bmatrix}$$

Taking its determinant:

$$|A - \lambda I| = (3-\lambda) \cdot (8-\lambda) - 4 \cdot 5$$
$$= 24 - 11\lambda + \lambda^2 - 20 = \lambda^2 - 11\lambda + 4$$

Setting this to 0 results,

$$|A - \lambda| = 0$$
$$= \lambda^2 - 11\lambda + 4 = 0$$

Using the quadratic formula,

$$\lambda = \frac{11 \pm \sqrt{(-11)^2 - 4 \cdot 1 \cdot 4}}{2 \cdot 1}$$
$$= \frac{11 \pm \sqrt{105}}{2}$$

Similarly for matrix B,

$$B = \begin{bmatrix} 4 & 2 \\ 3 & 1 \end{bmatrix}$$

Subtracting λI

$$B - \lambda I = \begin{bmatrix} 4-\lambda & 2 \\ 3 & 1-\lambda \end{bmatrix}$$

Then its determinant,

$$|B - \lambda I| = (4-\lambda)(1-\lambda) - 2 \cdot 3$$
$$= \lambda^2 - 5\lambda - 2$$

Setting it to 0,

$$\lambda^2 - 5\lambda - 2 = 0$$
$$\Rightarrow \lambda = \frac{5 \pm \sqrt{33}}{2}$$

To find the eigenvectors, we equate:

$$\begin{bmatrix} 3 & 4 \\ 5 & 8 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \frac{11 + \sqrt{105}}{2} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Resulting in the follow equations:

$$3x_1 + 4x_2 = \frac{11 + \sqrt{105}}{2} x_1,$$
$$5x_1 + 8x_2 = \frac{11 + \sqrt{105}}{2} x_2$$

2.1.2

Table of scores per group:

High	9, 7, 6.5, 8, 7.5, 7, 9.5, 8, 6.5
Medium	7.5, 8, 6, 7, 6.5, 7.5
Low	8, 6, 6, 6.5, 6.5

We denote by H the scores of the High group, by M the scores of the Medium group and by L the scores of the Low group. Then:

$\overline{H} = (9 + 7 + 6.5 + 8 + 7.5 + 7 + 9.5 + 8 + 6.5)/9 = \frac{69}{9}$
$\overline{M} = (7.5 + 8 + 6 + 7 + 6.5 + 7.5)/6 = \frac{85}{12}$
$\overline{L} = (8 + 6 + 6 + 6.5 + 6.5)/5 = \frac{33}{5}$

$$\begin{aligned} \sigma^2(H) &= \tfrac{1}{9-1} \textstyle\sum_{i=1}^{N} \left(x_i - \overline{H}\right)^2 \\ &= ((9-\tfrac{69}{9})^2 + (7-\tfrac{69}{9})^2 + (6.5-\tfrac{69}{9})^2 + (8-\tfrac{69}{9})^2 + (7.5-\tfrac{69}{9})^2 + (7-\tfrac{69}{9})^2 + (9.5-\tfrac{69}{9})^2 + (8-\tfrac{69}{9})^2 + (6.5-\tfrac{69}{9})^2)/8 \\ &= 1.125 \\ \sigma^2(M) &= \tfrac{1}{6-1} \textstyle\sum_{i=1}^{N} \left(x_i - \overline{M}\right)^2 \\ &= ((7.5 - \tfrac{85}{12})^2 + (8 - \tfrac{85}{12})^2 + (6 - \tfrac{85}{12})^2 + (7 - \tfrac{85}{12})^2 + (6.5 - \tfrac{85}{12})^2 + (7.5 - \tfrac{85}{12})^2)/5 \\ &= 0.5417 \\ \sigma^2(L) &= \tfrac{1}{5-1} \textstyle\sum_{i=1}^{N} \left(x_i - \overline{M}\right)^2 \\ &= ((8 - \tfrac{33}{5})^2 + (6 - \tfrac{33}{5})^2 + (6 - \tfrac{33}{5})^2 + (6.5 - \tfrac{33}{5})^2 + (6.5 - \tfrac{33}{5})^2)/5 \\ &= 0.54 \end{aligned}$$

$SS(W) = 9 \cdot 1.125 + 6 \cdot 0.5417 + 5 \cdot 0.54 = 16.0752$

Let $\bar{\bar{x}}$ denote the mean of the scores of all groups. Then:

$\bar{\bar{x}} = (9 \cdot \frac{69}{9} + 6 \cdot \frac{85}{12} + 5 \cdot \frac{33}{5})/(9 + 6 + 5) = 7.225$

$$\begin{aligned} SS(B) &= \textstyle\sum_{i=1}^{k} n_i \left(\bar{x}_i - \bar{\bar{x}}\right)^2 \\ &= 9(\tfrac{69}{9} - 7.225)^2 + 6(\tfrac{85}{12} - 7.225)^2 \\ &\quad +5(\tfrac{33}{5} - 7.225)^2 \\ &= 3.829 \end{aligned}$$

One-way ANOVA table:

Source	SS	df	MS	F
Between	3.829	2	1.946	2.300
Within	16.075	17	0.9456	NA
Total	19.904	19	NA	NA

2.2

The solutions to the sub-exercises of 2.2 can be found in the Matlab files provided.

2.2.3

See below code added to given script myEigenFaces:

# Step 1. Write code that projects the training data onto the first K principal components
# trainingVectors = ...
trainingVectors = transpose(pc(:,K))*trainingData; # projects the training data on to the k-th principal component
# note transpose of training data not needed as samples are already as
# columns and features as rows

# Step 2. Write code that subtracts the mean training face from the test data divided by sqrt(M-1) and projects the
# resulting matrix onto the first K prinicipal components
# testingVectors = ...
testingVectors = transpose(pc(:,K))*(testingData - repmat(meanTrainingFace,1,size(testingData,2)));
# test data projected on to k-th principalcomponent
#note transpose of testing data not needed as samples are already as
columns and features as rows

Note: line 31 in original script changed from original version to take the transpose of the data matrix A as A has samples as columns and features as rows (where it should be other way around)

Original: [pc, eigenvalues] = mypca(A)
edited: [pc, eigenvalues] = mypca(transpose(A))

Comparison between methods:
although principal component analysis seems to perform better it is more computationally expensive as it requires to calculate eigenvalues/vectors aswell as projecting data on to the principal components. this increase in computations may not justify the increase in accuracy