STAT8007

Worksheet 11 PCA

Let’s use R to analyse the data we used in class:

x1<- c(122, 21, 105, 101, 155, 131, 115, 53, 75, 45)

x2<-c(117, 32, 140, 105, 149, 146, 82, 60, 82, 37)

x1\_scaled<-(x1-mean(x1))/sd(x1)

x2\_scaled<-(x2-mean(x2))/sd(x2)

X<-cbind(x1\_scaled,x2\_scaled)

X

Plot this data:

windows(6,6)

plot(x2\_scaled ~ x1\_scaled)

We can use the prcomp function to run a PCA on our data

pca\_example<-prcomp(X, scale.= T, center = T)

Be sure to query the prcomp function to understand how it works.

summary(pca\_example)

We get the following output:

Importance of components:

PC1 PC2

Standard deviation 1.3839 0.29109

Proportion of Variance 0.9576 0.04237

Cumulative Proportion 0.9576 1.00000

The first row in the table shows the standard deviation of each PC

**NB the standard deviation is the square root of the eigenvalue associated with that component.**

The second row gives the proportion of the variation in the data accounted for by each PC. (i.e.λi/trace(S))

The third row gives the cumulative proportion of the variance accounted for by the PCs. An indication of the number of PCs required to adequately summarize the data can be inferred by examining the proportion of the variation explained by the PCs. In this case, the first PC accounts for 96% of the variation in the data. The second principal component only accounts for an additional 4% of the variation. This means that the first principal component can adequately describe this data set.

We can examine the principal components further:

pca\_example

We get the following output:

Standard deviations:

[1] 1.3839312 0.2910916

Rotation:

PC1 PC2

x1\_scaled -0.7071068 0.7071068

x2\_scaled -0.7071068 -0.7071068

The column labelled PC1 is the eigenvector of the data covariance matrix associated with the largest eigenvalue. Its elements are the coefficients, rotations or **loadings** of each original variable on the first PC.

Note that it matters if the loadings have opposite signs, but not which is positive and which is negative. The magnitudes of the loadings are also important.

We can also obtain the newly defined principal components using:

pca\_example$rotation

We can plot the principal components on our graph by plotting the eigenvectors:

slope1<-(-0.7071068/-0.7071068)

slope2<-(0.7071068/-0.7071068)

abline(0,slope1)

abline(0,slope2)

Recall that to view the original data in relation to the newly defined principal components we project the original data onto the principal components to obtain the **scores** which can be plotted in a score plot. The scores are contained in pca\_example$x

pca\_example$x

PC1 PC2

[1,] -0.8522751 0.13686738

[2,] 2.2116379 -0.16297023

[3,] -0.9431503 -0.52018369

[4,] -0.3074671 -0.01771826

[5,] -1.9220953 0.16609445

[6,] -1.4736639 -0.18478129

[7,] -0.1666352 0.58937611

[8,] 1.2235070 -0.08535832

[9,] 0.4994541 -0.07671312

[10,] 1.7306880 0.15538698

The first scaled observation had the coordinates (0.6994, 0.5059) in relation to the original variable axes (x1 and x2) but it now has the coordinates (-0.852, 0.137) in relation to the new principal components, PC1 and PC2.

The scores plot is given by:

plot(pca\_example$x)

**We can perform PCA ‘by hand’ on the same data set.**

Recall that the principal components are the eigenvectors of the correlation matrix associated with the data set.

cor\_matrix<-cor(X)

P <-eigen(cor\_matrix)

The eigen function returns the eigenvectors and the associated eigenvalues so we need to extract the eigenvectors (the principal components).

P<-P$vectors

We may now calculate the scores (the position of the original data in relation to the newly defined principal components). Recall from class, we calculate the scores (which we denoted as **T**) from our scaled data set **X** and the loadings (which we denoted **P**).

**T** = **XP**

T<-X%\*%P

Notice that we use the % symbol to ensure that matrix multiplication is performed.

Check that these scores correspond to pca\_example$x, the scores from the PCA model derived using the prcomp command.

Plot the scores.

To plot axes that cross at the origin try abline(v=0, h=0)

**Next, we calculate the residuals of a PCA model for determining model goodness of fit.**

First we calculate the X values modeled by using a PCA model using the identity

**X = TPT + E**

we can then calculate the residuals:

**E = X - TPT**

Remember that when using prcomp the scores, **T**, are given by pca$x and the loadings, **P**, are given by pca$rotation.

Let’s calculate the residuals for:

1. the model with 2 principal components (E2)
2. the model with 1 principal component (E1)

For two principal components we use both columns of the loadings matrix:

X\_mod2 <- pca\_example$x%\*%t(pca\_example$rotation)

The residuals are then:

E2<-X-X\_mod2

For just one principal component we use the first column of the loadings matrix only:

X\_mod1 <- pca\_example$x[,1]%\*%t(pca\_example$rotation[,1])

The residuals are then:

E1<-X-X\_mod1

Here pca\_example$x[,1], gives just the first column of the scores and pca\_example$rotation[,1] gives just the first column of the loadings (since we are using just 1 PC).

The symbols %\*% tell R that we wish to perform matrix multiplication, as opposed to element wise multiplication.

The function t() transposes a matrix.

To determine the goodness of fit of each model we use the formulae:

We need to determine and

This means square all the elements of the residual matrix **E** and sum over all rows and columns and do the same for the matrix of scaled data **X**.

The squared residuals are given by:

E2\_sq<-E2\*E2

(Note the element wise multiplication)

To calculate :

R2X <-1 – sum(sum(E2\_sq))/sum(sum(X\*X))

Now calculate the goodness of fit for the model with just one principal component.