Principal Component Analysis

Thileepan Paulraj

18 December 2018

UNDERSTANDING PCA (work reproduced from this web-page:https://goo.gl/Wgeieb)

While performing principal components analysis the original n-dimensional data get's projected onto several planes and the plane which has captured the maximum variance of the data becomes the first pricipal component. We select as many principal components as possible so that the maximum variance of the data is captured in minimal number of principal components.

Maximum variance is captured when the mean squared error between the original dataset and it's projection on to a plane is minimum.

Reading data

Let's use the diamonds data set now.

```
data = read.csv('diamonds.csv')
```

Viewing all the variable names in the dataset

```
colnames(data)
```

```
## [1] "X"       "carat"  "cut"     "color"   "clarity" "depth"   "table"
## [8] "price"   "x"       "y"       "z"
```

Taking only the numeric variables so we could use it in our analysis

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
data_for_pca <- select(data, -X, -cut, -color, -clarity)
```
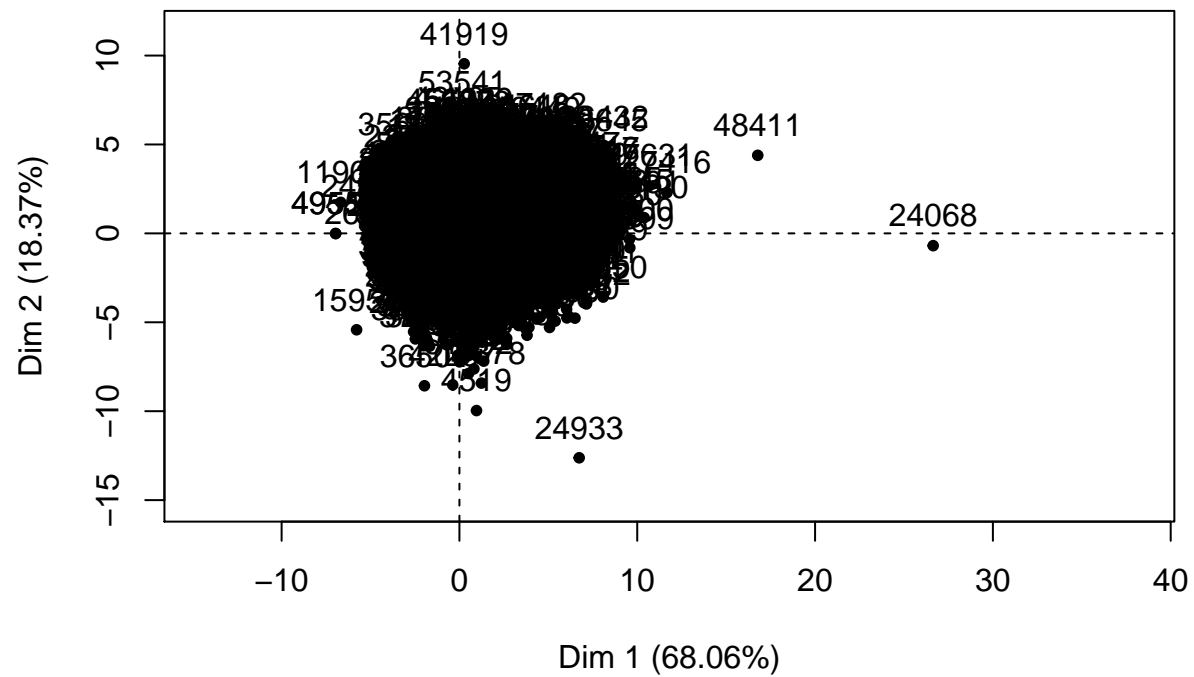
Principal components

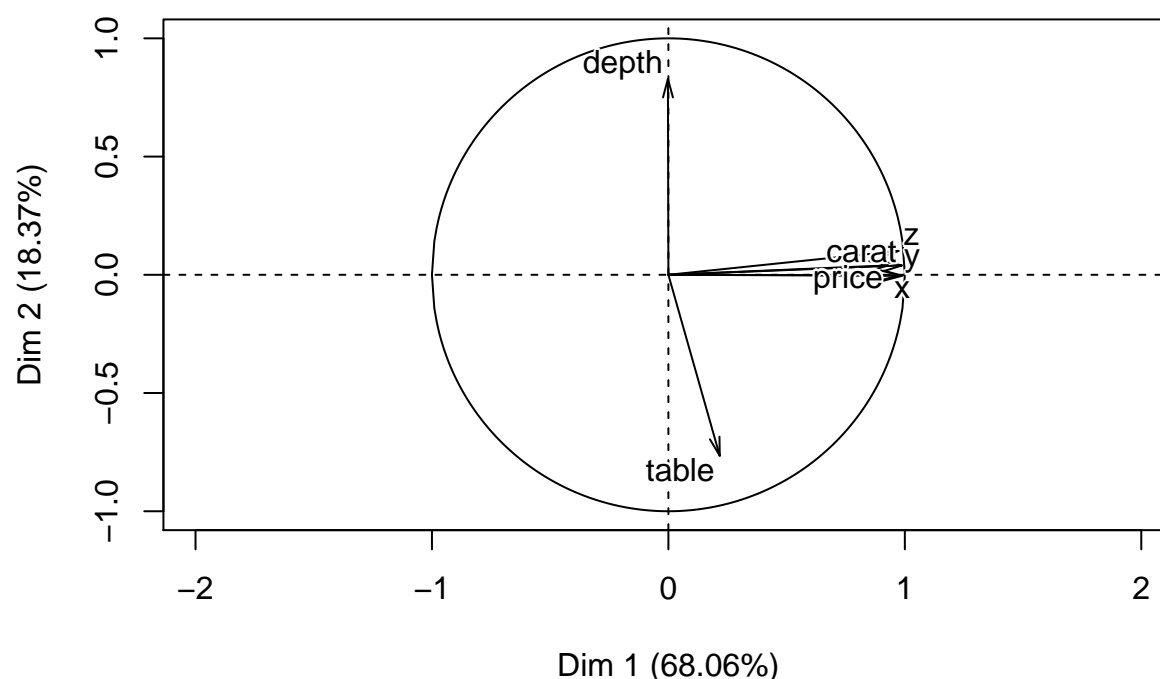Installing the factominer package for PCA

```r
library(FactoMineR)
```

```r
pca = PCA(data_for_pca)
```

Individuals factor map (PCA)

Variables factor map (PCA)

Here, the variables, 'price', 'carat', 'x', 'y', and 'z' form a composite variable called the Principal component 1 or Dim 1 which explains 68.06% of the variance in the data. Variable 'depth' explains 18.37% of the variance in the data along the second dimension. The variable 'table' is in the third dimension.

```
pca$eig
```

```
##        eigenvalue percentage of variance cumulative percentage of variance
## comp 1 4.76391480             68.0559258                          68.05593
## comp 2 1.28586808             18.3695440                          86.42547
## comp 3 0.69081126              9.8687323                          96.29420
## comp 4 0.17375333              2.4821905                          98.77639
## comp 5 0.04030722              0.5758174                          99.35221
## comp 6 0.03294659              0.4706656                          99.82288
## comp 7 0.01239871              0.1771245                         100.00000
```

Eigen values

In the table above, eigen values indicate how much variance each principal component explains. By doing this, the eigen values indicate which principal component is the most important. If we sort the eigen values and the corresponding eigen vectors in decending order, then we will know which principal components capture maximum variance. For example if we divide the first eigen value 4.763 by the total of all other eigen values then we will get a percentage of variance of 68.055. This indicates that the first prinipal component will capture 68.055% of total variance in the data.

Eigen Vectors

Eigen vectors transform our original data into principal components. Matrix multiplication of our scaled and centred original dataset with eigen vector number 1 will generate data for principal component 1. Each of these components are projected in a different direction in the 3-D space. The principal components are orthoginal to each other.

How much of each variable is represented in each principal component

Now, let's see how much variance of each variable is explained by each principal component.

```
Correlation_Matrix = as.data.frame(round(cor(data_for_pca,pca$ind$coord)^2*100,0))
Correlation_Matrix[with(Correlation_Matrix, order(-Correlation_Matrix[,1])),]
```

```
##        Dim.1 Dim.2 Dim.3 Dim.4 Dim.5
## carat     98     0     0     0     0
## x         98     0     0     1     0
## y         95     0     0     2     2
## z         95     1     0     2     1
## price     86     0     1    13     0
## table      5    59    37     0     0
## depth      0    69    31     0     0
```

This correlation matric tells us that 98% of the information in carat and X variables are loaded in the first dimension, 95% of the information from y and z variables are loaded in the first dimension. Information from the variables table and depth is spread between dimension 2 and 3. Information of the variable price is spread between 1st and 4th principal components.

If we discard the 5th principal component we will only loose 3% of the information from only 2 variables. Therefore it is safe is discard this dimension and only keep the 4 remaining dimensions.

COMPUTING PRINCIPAL COMPONENT ANALYSIS STEP BY STEP

Let's use the iris data set to perform pricipal component analysis

```
df <- iris[, -5]
```

Let's centre and scale the data

```
df.scaled <- scale(df, center = TRUE, scale = TRUE)
```

Computing the correlation matrix

The correlation matrix finds how each variable in our dataset is correlated with the other variables. The diagonals of the correlation matrix will always contain 1s because it's the value of correlation of the variable with itself.

The dimension of the correlation matrix will depend on the number of variables/ features present in our data set. If we have 4 variables then the correlation matrix will be [4,4] in dimension. If we have 16 variables then the correlation matrix will be [16,16] in dimension.

```
res.cor = cor(df.scaled)
round(res.cor, 2)
```

```
##            Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length        1.00       -0.12         0.87        0.82
## Sepal.Width        -0.12        1.00        -0.43       -0.37
## Petal.Length        0.87       -0.43         1.00        0.96
## Petal.Width         0.82       -0.37         0.96        1.00
```

Calculating Eigen Values and Eigen Vectors

```
res.eig = eigen(res.cor)

res.eig
```

```
## eigen() decomposition
## $values
## [1] 2.91849782 0.91403047 0.14675688 0.02071484
##
## $vectors
##            [,1]        [,2]       [,3]       [,4]
## [1,]  0.5210659 -0.37741762  0.7195664  0.2612863
## [2,] -0.2693474 -0.92329566 -0.2443818 -0.1235096
## [3,]  0.5804131 -0.02449161 -0.1421264 -0.8014492
## [4,]  0.5648565 -0.06694199 -0.6342727  0.5235971
```

Calculating principal components

As mentioned under the heading 'Eigen Vector' above, to calculate principal components, we need to multiply the scaled and centred data set with eigen vectors. Multiplying the first eigen vector with the data matrix gives us the first principal component and multiplying the data matrix with the second eigen vector gives us the second principal component and so on and so forth.

Transposing the eigen vectors

```
eigenvectors.t = t(res.eig$vectors)
```

Transposing the adjusted data

```
df.scaled.t = t(df.scaled)
```

Calculating the new principal components

```
pc = eigenvectors.t %*% df.scaled.t
pc = t(pc)
colnames(pc) <- c("PC1", "PC2", "PC3", "PC4")
head(pc)
```

```
##            PC1        PC2        PC3         PC4
## [1,] -2.257141 -0.4784238  0.12727962  0.024087508
## [2,] -2.074013  0.6718827  0.23382552  0.102662845
```

```
## [3,] -2.356335  0.3407664 -0.04405390  0.028282305
## [4,] -2.291707  0.5953999 -0.09098530 -0.065735340
## [5,] -2.381863 -0.6446757 -0.01568565 -0.035802870
## [6,] -2.068701 -1.4842053 -0.02687825  0.006586116
```