Chuong Vu

COMP 5800

Homework 4

Prof. Byung Kim

**PCA iris dataset**

**Perform the PCA clustering and plot clusters in the first two principal components.**

Do to the PCA clustering, mean that we have to find the k-mean, k-medoids, hierarchical, and density-based clustering.

First, let take a look at the iris data. By using R language, I can get the data from its library. Then we can explore the size and structure of data.

# Get data from the library

**> dim(iris)**

[1] 150 5

#Display the name of data header

**> names(iris)**

[1] "Sepal.Length" "Sepal.Width" "Petal.Length" "Petal.Width" "Species"

#data's structure

**> str(iris)**

'data.frame': 150 obs. of 5 variables:

$ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...

$ Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...

$ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...

$ Petal.Width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...

$ Species : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...

#data's attributes

**> attributes(iris)**

$names

[1] "Sepal.Length" "Sepal.Width" "Petal.Length" "Petal.Width" "Species"

$row.names

[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

[21] 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40

[41] 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60

[61] 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

[81] 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

[101] 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120

[121] 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140

[141] 141 142 143 144 145 146 147 148 149 150

$class

[1] "data.frame"

*Let first start with* ***correlation and covariance matrix***

# correlation matrix

**> cor(iris[1:4])**

Sepal.Length Sepal.Width Petal.Length Petal.Width

Sepal.Length 1.0000000 -0.1175698 0.8717538 0.8179411

Sepal.Width -0.1175698 1.0000000 -0.4284401 -0.3661259

Petal.Length 0.8717538 -0.4284401 1.0000000 0.9628654

Petal.Width 0.8179411 -0.3661259 0.9628654 1.0000000

# covariance matrix

**> cov(iris[1:4])**

Sepal.Length Sepal.Width Petal.Length Petal.Width

Sepal.Length 0.6856935 -0.0424340 1.2743154 0.5162707

Sepal.Width -0.0424340 0.1899794 -0.3296564 -0.1216394

Petal.Length 1.2743154 -0.3296564 3.1162779 1.2956094

Petal.Width 0.5162707 -0.1216394 1.2956094 0.5810063

We can see that it have really strong positive relatively correlation between Petal Length, Sepal Length and Petal Width. It also show that Petal Length have more than 3 times the variance of the other 3 variables.

But this not really show us much how the data looks when all 4 attributes are considered simultaneously. So we will compute the principal components, using both the covariance and correlation matrix to see what we can learn from the data.

**PCA on the Covariance Matrix *(Principal Components, Loadings, and Variance Explained)***

Let calculate the eigenvalues.

# calulate eigenvalues

**> covM <- cov(iris[1:4])**

**> eig <- eigen(covM,symmetric=TRUE,only.values=FALSE)**

**> c <- colnames(iris[1:4])**

**> eig$values**

[1] 4.22824171 0.24267075 0.07820950 0.02383509

**> rownames(eig$vectors) <- c(colnames(iris[1:4]))**

**> eig$vectors**

[,1] [,2] [,3] [,4]

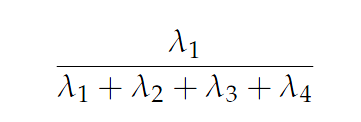
Sepal.Length 0.36138659 -0.65658877 -0.58202985 0.3154872

Sepal.Width -0.08452251 -0.73016143 0.59791083 -0.3197231

Petal.Length 0.85667061 0.17337266 0.07623608 -0.4798390

Petal.Width 0.35828920 0.07548102 0.54583143 0.7536574

Look at the eigenvalues, it tell us that how much the total variance in the data is directed along each eigenvector. This amount of variance along V1 is λ1 and the proportion of variance explained by the first principal component is

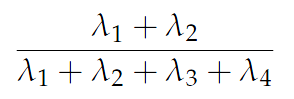


># get first component variation

**> eig$values[1]/sum(eig$values)**

[1] 0.9246187

0.92 mean 92% of the variation in the data. It is explained by the first component alone. So by using two dimensional from the first and second principal component directions representation, we can get the proportion of variance:



> # get second component variation

**> sum(eig$values[1:2])/sum(eig$values)**

[1] 0.9776852

With this two dimensions, we can explain 97.8% of the variance in these 4 variables. The entries in each eigenvector are called the *loadings* of the variables on the component. The loading give us an idea how important each variable is to each components.

**Principal Components in R**

**> irispca <- princomp(iris[1:4])**

# Variance Explained

**> summary(irispca)**

Importance of components:

Comp.1 Comp.2 Comp.3 Comp.4

Standard deviation 2.0494032 0.49097143 0.27872586 0.153870700

Proportion of Variance 0.9246187 0.05306648 0.01710261 0.005212184

Cumulative Proportion 0.9246187 0.97768521 0.99478782 1.000000000

# Eigenvectors

**> irispca$loadings**

Loadings:

Comp.1 Comp.2 Comp.3 Comp.4

Sepal.Length 0.361 -0.657 -0.582 0.315

Sepal.Width -0.730 0.598 -0.320

Petal.Length 0.857 0.173 -0.480

Petal.Width 0.358 0.546 0.754

Comp.1 Comp.2 Comp.3 Comp.4

SS loadings 1.00 1.00 1.00 1.00

Proportion Var 0.25 0.25 0.25 0.25

Cumulative Var 0.25 0.50 0.75 1.00

# Coordinates of the data along PCs:

**> irispca$scores[1:10, ]**

Comp.1 Comp.2 Comp.3 Comp.4

[1,] -2.684126 -0.31939725 -0.02791483 0.002262437

[2,] -2.714142 0.17700123 -0.21046427 0.099026550

[3,] -2.888991 0.14494943 0.01790026 0.019968390

[4,] -2.745343 0.31829898 0.03155937 -0.075575817

[5,] -2.728717 -0.32675451 0.09007924 -0.061258593

[6,] -2.280860 -0.74133045 0.16867766 -0.024200858

[7,] -2.820538 0.08946138 0.25789216 -0.048143106

[8,] -2.626145 -0.16338496 -0.02187932 -0.045297871

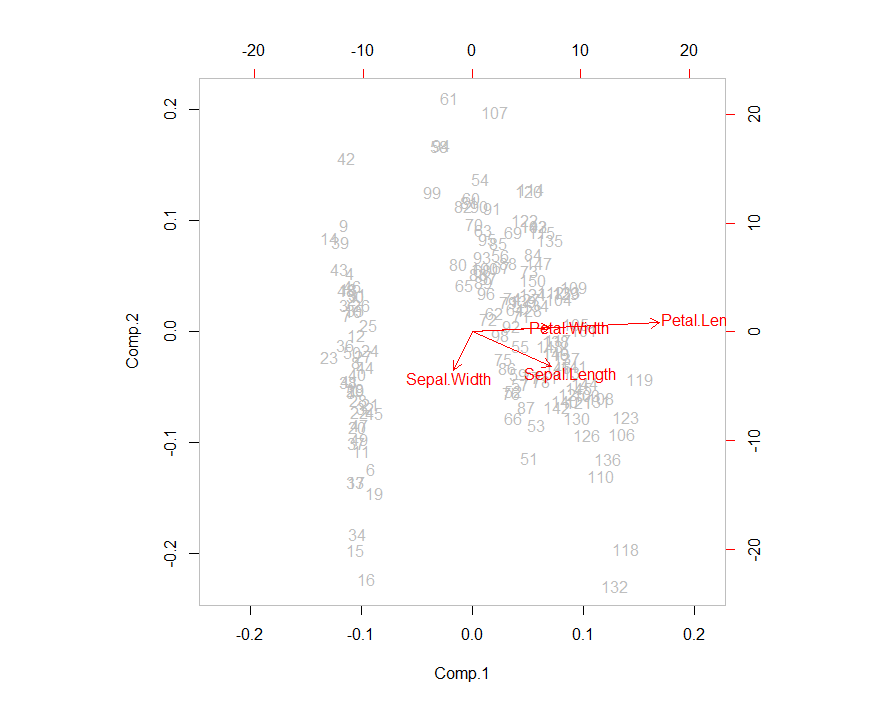
[9,] -2.886383 0.57831175 0.02075957 -0.026744736

[10,] -2.672756 0.11377425 -0.19763272 -0.056295401

After we get all of this information, we can see the variables on the 2 dimension graph by using biplot function in R. The PCA biplot allows us to see where our original variables fall in the space of the principal components. Uncorrelated variables will appear further apart.

> # Biplot graph

**> biplot(irispca, col = c("blue", "red"))**



Look at the graph, we can see how they align these projected variable directions. The arrow points down then the positive direction is down indicating observations which are grater then the mean. We pick number 42 and 132 which is seem like further apart from the data points to see what actual data points look like in the comparison to the rest of the sample population.

Now, with the function summary(), we can get the minimum, maximum, mean, median, the first (25%) and the third (75%) quartiles.

**> summary(iris[1:4])**

Sepal.Length Sepal.Width Petal.Length Petal.Width Species

Min. :4.300 Min. :2.000 Min. :1.000 Min. :0.100 setosa :50

1st Qu.:5.100 1st Qu.:2.800 1st Qu.:1.600 1st Qu.:0.300 versicolor:50

Median :5.800 Median :3.000 Median :4.350 Median :1.300 virginica :50

Mean :5.843 Mean :3.057 Mean :3.758 Mean :1.199

3rd Qu.:6.400 3rd Qu.:3.300 3rd Qu.:5.100 3rd Qu.:1.800

Max. :7.900 Max. :4.400 Max. :6.900 Max. :2.500

# Consider orientaion of outlying observations:

**> iris[42, ]**

Sepal.Length Sepal.Width Petal.Length Petal.Width Species

42 4.5 2.3 1.3 0.3 setosa

> **iris[132, ]**

Sepal.Length Sepal.Width Petal.Length Petal.Width Species

132 7.9 3.8 6.4 2 virginica

**Clustering with PCA**

1. ***The k-means clustering***

> # k-mean clustering

**> iris2 <- iris**

**> iris2$Species <- NULL**

**> (kmeans.result <- kmeans(iris2, 3))**

K-means clustering with 3 clusters of sizes 50, 38, 62

Cluster means:

Sepal.Length Sepal.Width Petal.Length Petal.Width

1 5.006000 3.428000 1.462000 0.246000

2 6.850000 3.073684 5.742105 2.071053

3 5.901613 2.748387 4.393548 1.433871

Clustering vector:

[1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

[42] 1 1 1 1 1 1 1 1 1 3 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3

[83] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 2 2 2 2 3 2 2 2 2 2 2 3 3 2 2 2 2 3 2 3 2

[124] 3 2 2 3 3 2 2 2 2 2 3 2 2 2 2 3 2 2 2 3 2 2 2 3 2 2 3

Within cluster sum of squares by cluster:

[1] 15.15100 23.87947 39.82097

(between\_SS / total\_SS = 88.4 %)

Available components:

[1] "cluster" "centers" "totss" "withinss" "tot.withinss"

[6] "betweenss" "size" "iter" "ifault"

Then we compare the clustering result with the class label (Species) to check whether similar objects are grouped together.

**> table(iris$Species, kmeans.result$cluster)**

1 2 3

setosa 50 0 0

versicolor 0 2 48

virginica 0 36 14

The result above shows that the cluster *setosa* can be easily separated from the other clusters, and the clusters *versicolor* and *virginica* are to as small degree overlapped with each other.

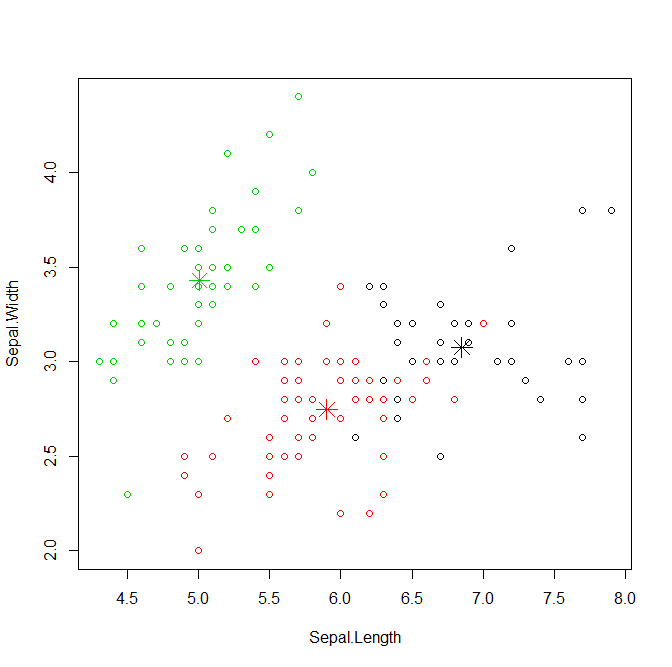
Now, let make a k-Means Clustering graph

# k-Means Clustering graph

**plot(iris2[c("Sepal.Length", "Sepal.Width")], col = kmeans.result$cluster)**

# plot cluster centers

**points(kmeans.result$centers[,c("Sepal.Length", "Sepal.Width")], col = 1:3, pch = 8, cex = 2)**



This graph show the clusters and their centers. Some black points close to the green center (star) are actually closer to the black center in the four dimensional space.

1. ***The k-medoids clustering***

The k-medoids clustering is similar to k-means. The major difference between them is that while a cluster is represented with its center in the k-mean algorithm, it is represented with the object closest to the center of the cluster in the k-medoids clustering. The k-medoids clustering is more robust than k-means in presence of outliers. PAM (Partitioning Around Medoids) is a classic algorithm for k-medoids clustering. While the PAM algorithm is inefficient for clustering large data, the CLARA algorithm is an enhanced technique of PAM by drawing multiple samples of data, applying PAM on each sample and then returning the best clustering.

Let demonstrate how to find clusters with PAM.

**> library(fpc)**

**> pamk.result <- pamk(iris2)**

# number of clusters

**> pamk.result$nc**

[1] 2

# check clustering against actual species

**> table(pamk.result$pamobject$clustering, iris$Species)**

setosa versicolor virginica

1 50 1 0

2 0 49 50

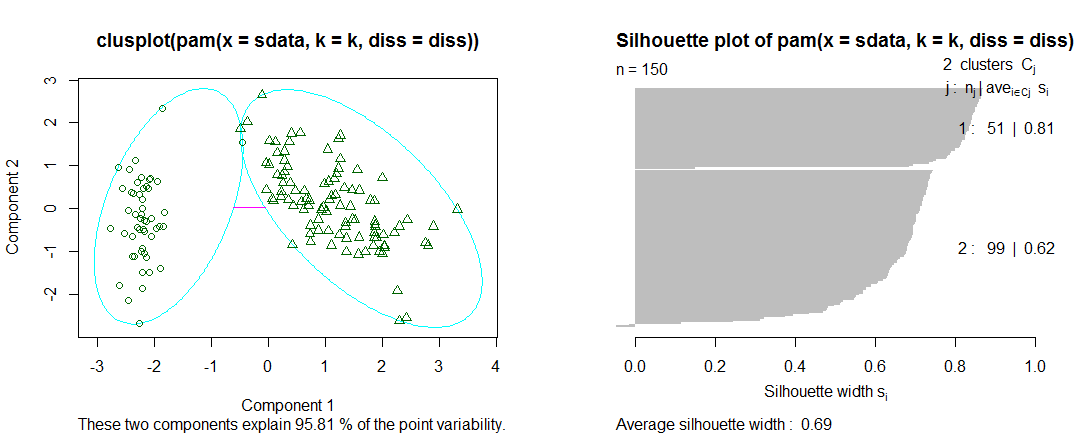
> # 2 graphs per page

**> layout(matrix(c(1,2),1,2))**

**> plot(pamk.result$pamobject)**

> # change back to one graph per page

**> layout(matrix(1))**



For the code above, pamk() function produces two clusters: one is *setosa*, and the other is a mixture of *versicolor* and *virginica.* In the figure above, the left chart it a 2-d *clusplot* (clustering plot) of the two clusters and the lines show the distance between clusters. The right figure shows their silhouettes. In the silhouettes, a large si (almost 1)suggests that the **corresponding** **observations** are very well clustered, a small si (almost 0) means that the observation lies between two clusters, and observations with a negative si are probably placed in the wrong cluster. Since the average Si are respectively 0.81 and 0.62 in the above silhouette, the identified two clusters are well clustered.

Now we can try PAM with k = 3

# try pam() with k = 3

**> library(cluster)**

**> pam.result <- pam(iris2, 3)**

**> table(pam.result$clustering, iris$Species)**

setosa versicolor virginica

1 50 0 0

2 0 48 14

3 0 2 36

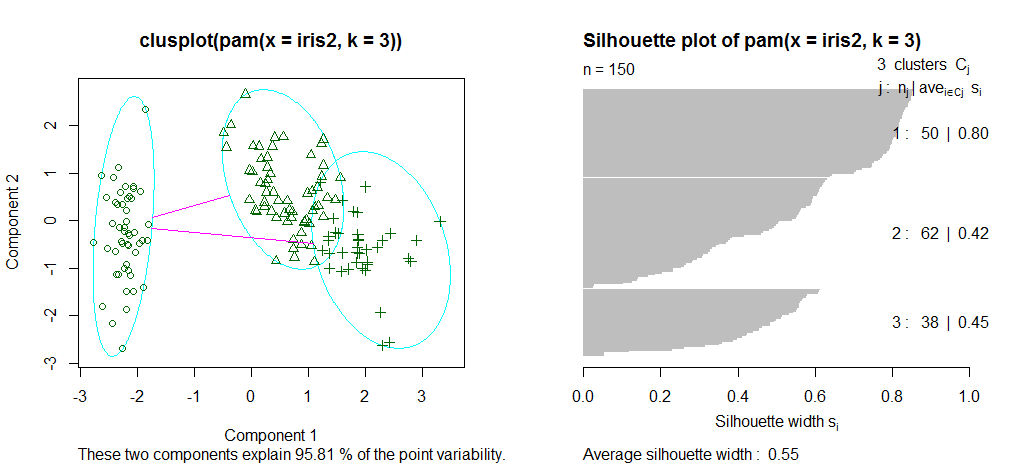
# 2 graphs per page

**> layout(matrix(c(1,2),1,2))**

**> plot(pam.result)**

# change back to one graph per page

**> layout(matrix(1))**



With the result above produced with pam() function, there are three clusters: first is cluster 1 is species *setosa* and it well separated from the other two; second is cluster 2 is mainly composed of *versicolor* plus some case from *virginica;* and third is the majority of the cluster 3 are *virginica* with two cases from *versicolor.*

**Hierarchical Clustering**

With hierarchical Clustering, we first draw a sample of 40 records from the iris data, so that the clustering plot will not be over crowed. Same as before, variable Species is removed from the data. After that, then we apply hierarchical clustering to the data.

# HIERACHICAL CLUSTERING

**> idx <- sample(1:dim(iris)[1], 40)**

**> irisSample <- iris[idx, ]**

**> irisSample$Species <- NULL**

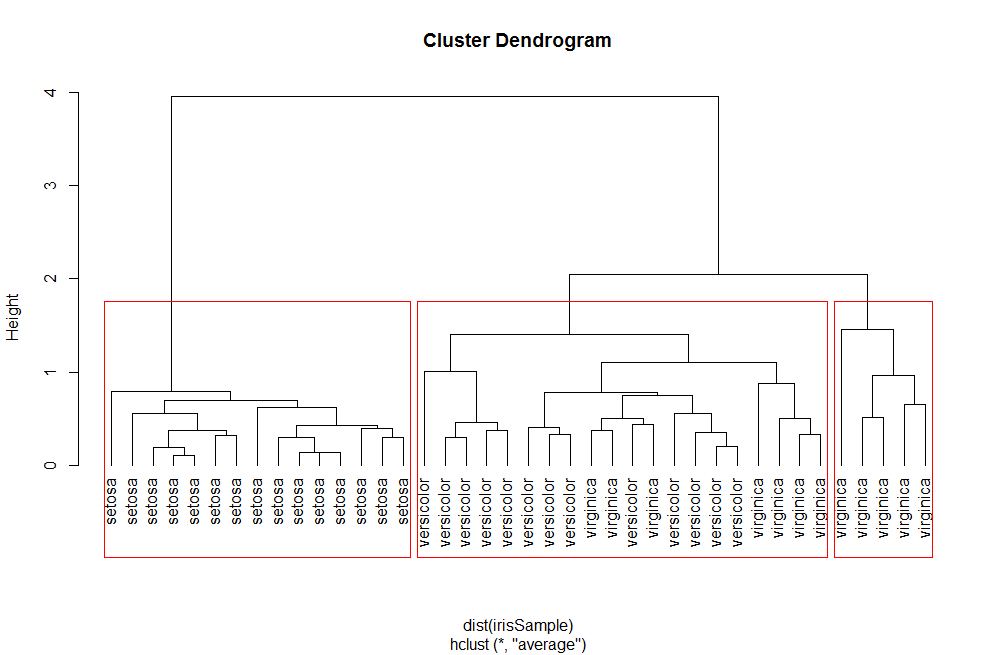
**> hc <- hclust(dist(irisSample), method="ave")**

**> plot(hc, hang = -1, labels=iris$Species[idx])**

> # cut tree into 3 clusters

**> rect.hclust(hc, k=3)**

**> groups <- cutree(hc, k=3)**



Similar to thee clustering of k-means. The figure above shows that cluster *setosa* can be easily separated from the other two clusters, and that clusters *versicolor* and *virginica* are to a small degree overlapped with each other.

**Variable Clustering with PCA**

The direction arrows on the biplot are merely the coefficients of the original variables when combined to make principal components. For example, we have the first principal component (the first column of **V**) **v**1 as:

**> eig$vectors[, 1]**

Sepal.Length Sepal.Width Petal.Length Petal.Width

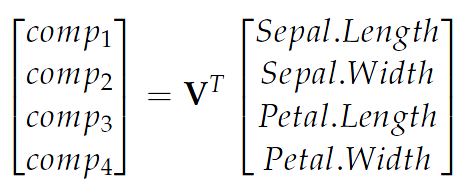
0.36138659 -0.08452251 0.85667061 0.35828920

This means that

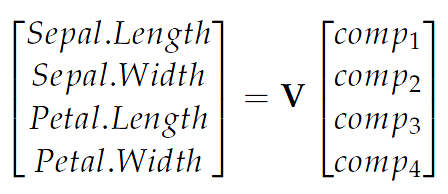
*comp1 = 0.36Sepal.Length - 0.08Sepal.Width + 0.85Petal.Length + 0.35Petal.Width*

the same equation could be written for each of the principal components, comp1,…, comp4.

Essentially, we have a system of equations telling us that the rows of **V**T (i.e. the columns of **V**) give us the weights of each variable for each principal component:



So if we want the coordinates of our original variables in term of Principal Components (so that we can plot them as we do in the biplot) we need to look no further than the rows of the matrix **V** as



means that the rows of V give us the coordinates of our original variables in the PCA space.

#First entry in each eigenvectors give coefficients for Variable 1:

**> eig$vectors[1,]**

[1] 0.3613866 -0.6565888 -0.5820299 0.3154872

Sepal.Length = 0.361comp1 - 0.657comp2 - 0.582comp3 + 0.315comp4

As we can see this on the biplot. The vector shown for Sepal.Length is (0.361, -0.656), which is the 2-d projection formed by throwing out components 3 and 4. Variables which lie upon similar directions in the PCA space tend to change in a similar fashion.

**Comparison with PCA on the Correlation Matrix**

We can complete the same analysis using the correlation matrix with loadings, scores and variance explained directly from eigenvectors and eigenvalues.

# Comparison with PCA and Correlation Matrix

**> irispca2 <- princomp(iris[1:4], cor = TRUE)**

**> summary(irispca2)**

Importance of components:

Comp.1 Comp.2 Comp.3 Comp.4

Standard deviation 1.7083611 0.9560494 0.38308860 0.143926497

Proportion of Variance 0.7296245 0.2285076 0.03668922 0.005178709

Cumulative Proportion 0.7296245 0.9581321 0.99482129 1.000000000

**> irispca2$loadings**

Loadings:

Comp.1 Comp.2 Comp.3 Comp.4

Sepal.Length 0.521 -0.377 0.720 0.261

Sepal.Width -0.269 -0.923 -0.244 -0.124

Petal.Length 0.580 -0.142 -0.801

Petal.Width 0.565 -0.634 0.524

Comp.1 Comp.2 Comp.3 Comp.4

SS loadings 1.00 1.00 1.00 1.00

Proportion Var 0.25 0.25 0.25 0.25

Cumulative Var 0.25 0.50 0.75 1.00

>

>

**> irispca2$scores[1:10, ]**

Comp.1 Comp.2 Comp.3 Comp.4

[1,] -2.264703 -0.4800266 0.12770602 0.02416820

[2,] -2.080961 0.6741336 0.23460885 0.10300677

[3,] -2.364229 0.3419080 -0.04420148 0.02837705

[4,] -2.299384 0.5973945 -0.09129011 -0.06595556

[5,] -2.389842 -0.6468354 -0.01573820 -0.03592281

[6,] -2.075631 -1.4891775 -0.02696829 0.00660818

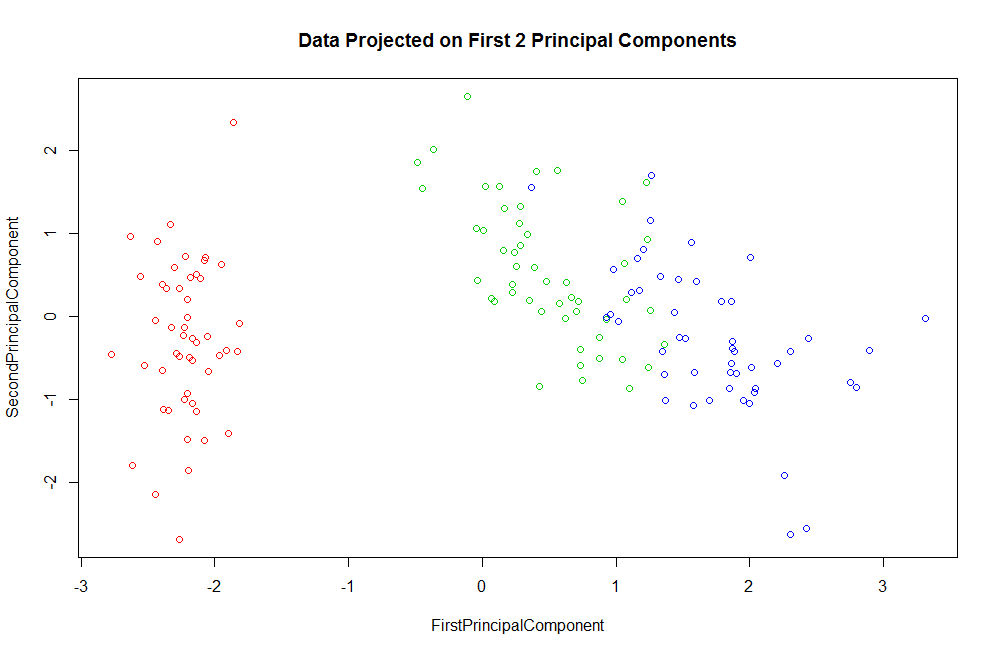
[7,] -2.444029 -0.0476442 -0.33547040 -0.03677556

[8,] -2.232847 -0.2231481 0.08869550 -0.02461210

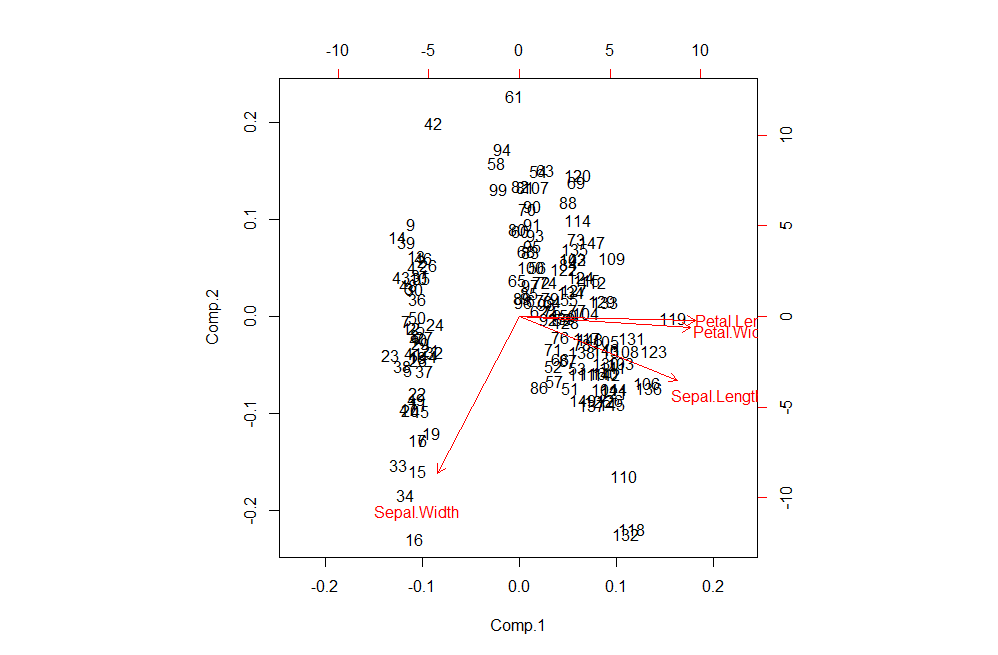
[9,] -2.334640 1.1153277 -0.14507686 -0.02685922

[10,] -2.184328 0.4690136 0.25376557 -0.03989929

**> plot(irispca2$scores[, 1], irispca2$scores[, 2], main = "Data Projected on First 2 Principal Components", xlab="FirstPrincipalComponent",ylab="SecondPrincipalComponent", col=c("red","green3","blue")[iris$Species])**



**> biplot(irispca2)**



As we can can see the direction vectors of the original variables are relatively uniform in length in the PCA space. This is due to the standardization in the correlation matrix. However, the general message is the same: Petal.Width and Petal.Length Cluster together, and many of the same observations appear "on the fray" on the PCA space - although not all of them!

My clustering result is seem like using the same algorithm from "The global k-means clustering algorithm." by *Aristidis Likas and Nikos A. Vlassis and Jakob J. Verbeek*. Where I use the k-means to get the clustering graph and the published article did the same too.

To find which data the PCA point corresponds to, I calculate the first four columns of the data set correspond to biometric measurements ("Sepal.Length" "Sepal.Width" "Petal.Length" "Petal.Width").

**> data(iris)**

**> dat <- as.matrix(iris[,-5])**

**> pca <- prcomp(dat, retx=TRUE, center=TRUE, scale=TRUE)**

>

>

> #Create new data sets for each of the three species.

> #Biometric values are based on the distributions of the original data means

> #and the covariances between these parameters.

**> setosa.mean <- apply(iris[iris$Species=="setosa",-5], 2, mean)**

**> setosa.cov <- cov(iris[iris$Species=="setosa",-5])**

>

**> versicolor.mean <- apply(iris[iris$Species=="versicolor",-5], 2, mean)**

**> versicolor.cov <- cov(iris[iris$Species=="versicolor",-5])**

>

**> virginica.mean <- apply(iris[iris$Species=="virginica",-5], 2, mean)**

**> virginica.cov <- cov(iris[iris$Species=="virginica",-5])**

>

>

> #Make new random data based on the calculated biometry info. each species

> #The MASS package allows for the calculation of correlated/covarying random

> #numbers using this information.

**> library(MASS)**

**> set.seed(1)**

**> n <- 30**

**> new.setosa <- mvrnorm(n, setosa.mean, setosa.cov)**

**> new.versicolor <- mvrnorm(n, versicolor.mean, versicolor.cov)**

**> new.virginica <- mvrnorm(n, virginica.mean, virginica.cov)**

>

>

> ###Predict PCs by projecting the new data using the predict.prcomp function

**> pred.setosa <- predict(pca, new.setosa)**

**> pred.versicolor <- predict(pca, new.versicolor)**

**> pred.virginica <- predict(pca, new.virginica)**

>

>

> ###Plot result

**> SPP <- iris$Species**

**> COLOR <- c(2:4)**

**> PCH <- c(1,16)**

>

**> pc <- c(1,2)**

**> plot(pca$x[,pc[1]], pca$x[,pc[2]], col=COLOR[SPP], cex=PCH[1], xlab=paste0("PC ", pc[1], " (", round(pca$sdev[pc[1]]/sum(pca$sdev)\*100,0), "%)"), ylab=paste0("PC ", pc[2], " (", round(pca$sdev[pc[2]]/sum(pca$sdev)\*100,0), "%)"))**

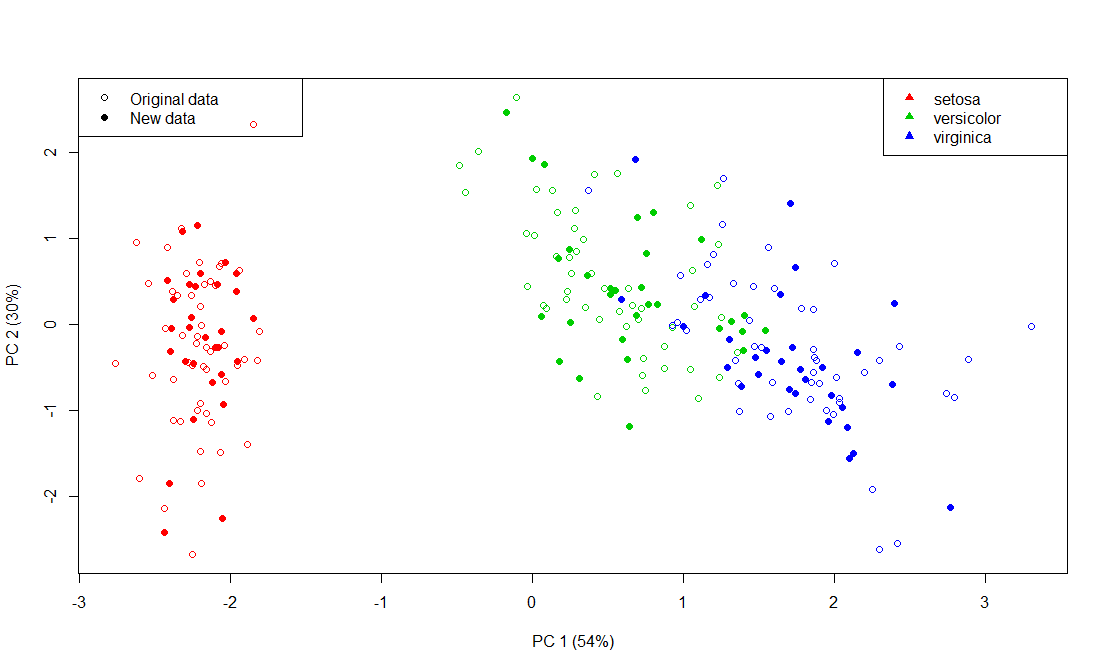
**> points(pred.setosa[,pc[1]], pred.setosa[,pc[2]], col=COLOR[levels(SPP)=="setosa"], pch=PCH[2])**

**> points(pred.versicolor[,pc[1]], pred.versicolor[,pc[2]], col=COLOR[levels(SPP)=="versicolor"], pch=PCH[2])**

**> points(pred.virginica[,pc[1]], pred.virginica[,pc[2]], col=COLOR[levels(SPP)=="virginica"], pch=PCH[2])**

**> legend("topright", legend=levels(iris$Species), col=COLOR, pch=17)**

**> legend("topleft", legend=c("Original data", "New data"), col=1, pch=PCH)**



By doing this performance, we can see the new and the original data on graph. The new data will be based on the original data with be calculated the correlated/covarying random number using MASS package from R and it will create a new data. The graph above is show how the new data lie in a similar area of the plot as the original data set.

# References

box, M. i. (2013, Oct 13). *http://stats.stackexchange.com/questions/72839/how-to-use-r-prcomp-results-for-prediction*.

Race, S. (n.d.). *http://www4.ncsu.edu/~slrace/LinearAlgebra2016/.* Retrieved from MSA Linear Algebra.

Verbeek, A. L. (n.d.). The global k-means clustering algorithm. Retrieved from The global k-means clustering algorithm: https://pdfs.semanticscholar.org/c82e/5c28fb271280a628588663b9eb66b262b6b0.pdf

Zhao, Y. (2013, April 26). *http://www.rdatamining.com/.* Retrieved from RDataMining.com: R and Data Mining: ftp://cran.r-project.org/pub/R/doc/contrib/Zhao\_R\_and\_data\_mining.pdf