Class 7 Lab: Machine Learning

Max Strul

10/19/22

Intro to machine learning

Unsupervised, supervised and reinforcement learning

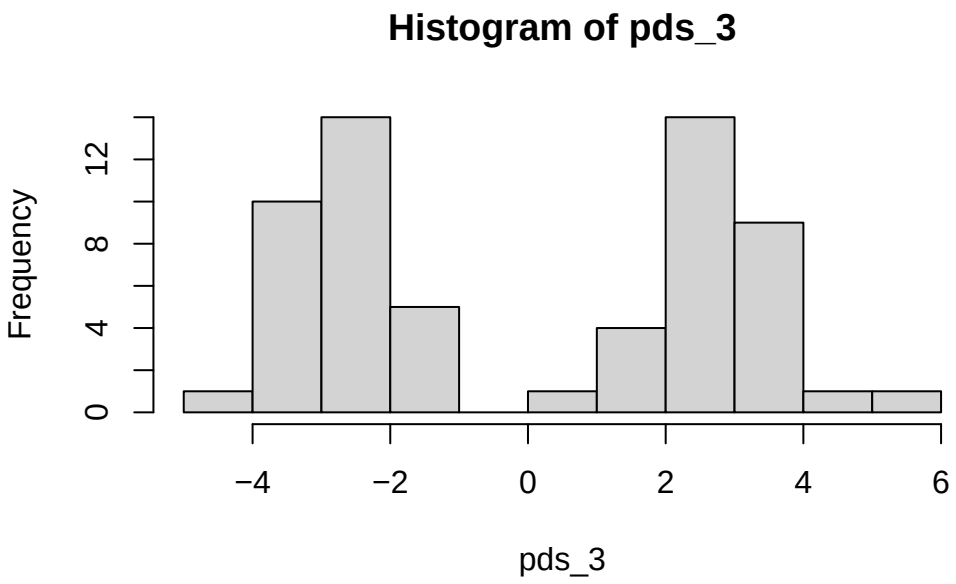
Clustering

K-means clustering

The problem with k-means clustering is that we have to first declare how many groups we want the data to be separated into.

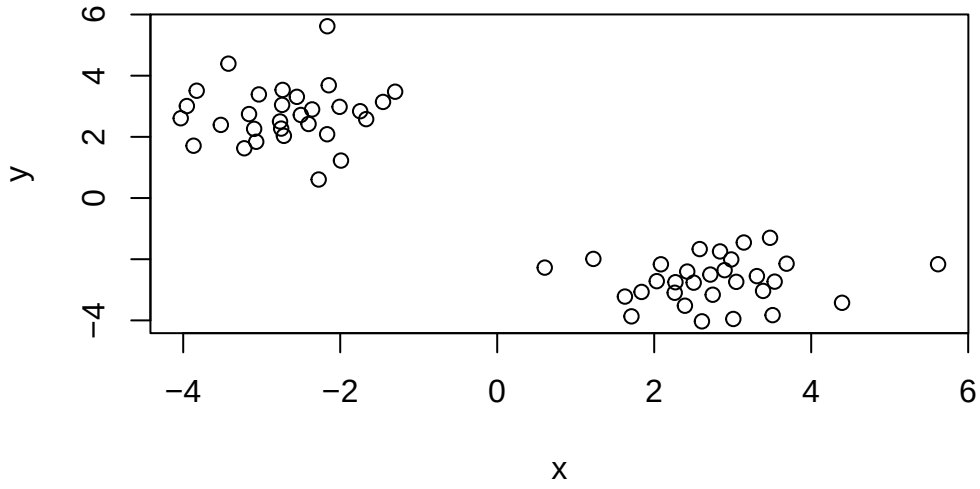
We will start by making up some data to cluster

```
pds_1 <- rnorm(30,-3)
pds_2 <- rnorm(30,3)
pds_3 <- c(pds_1,pds_2)
#you can use mean=3 or just a 3
hist(pds_3)
```



K=2 groups

```
clustering <- cbind(x=pds_3,y=rev(pds_3))
plot(clustering)
```



We can visually see that we have created two distinct groups.

How does k-means clustering in base R work?

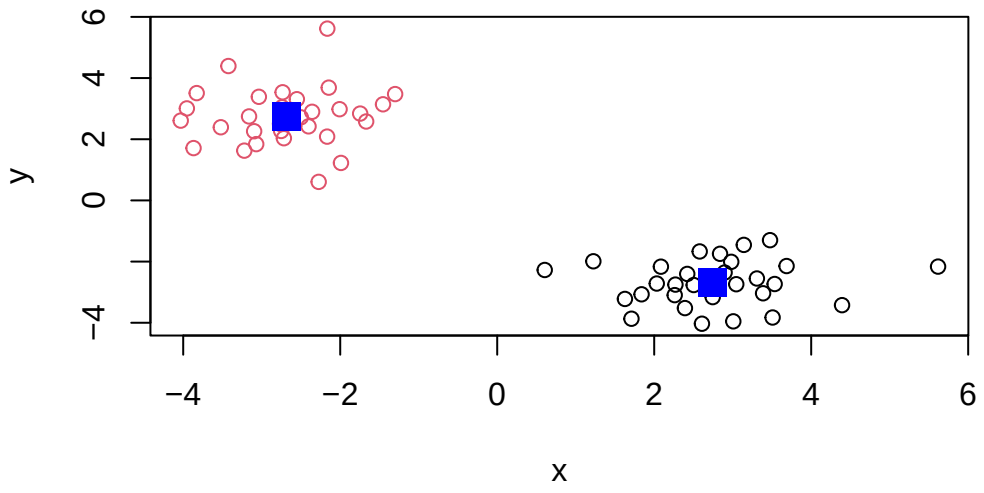
`kmeans()` is the function.

```
km_1<-kmeans(clustering, centers=2, nstart=10)
km_1
```

```
K-means clustering with 2 clusters of sizes 30, 30

Cluster means:
          x         y
1  2.748030 -2.688637
2 -2.688637  2.748030

Clustering vector:
 [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1
[39] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

Within cluster sum of squares by cluster:
[1] 42.35823 42.35823
 (between_SS / total_SS =  91.3 %)
```

```
Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
```

```r
plot(clustering, col=km_1$cluster)
points(km_1$centers,col="blue",pch=15,cex=2)
```

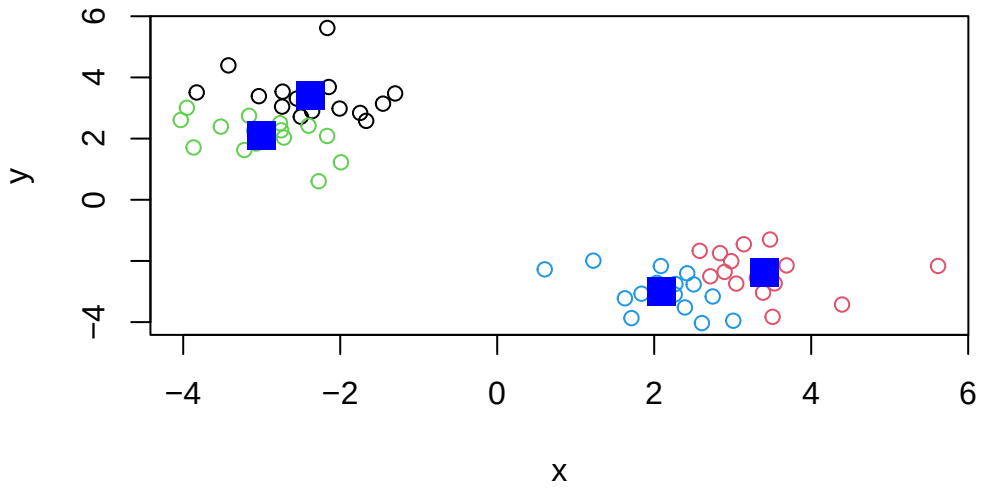



```r
km_2<-kmeans(clustering, centers=4, nstart=20)
km_2
```

```
K-means clustering with 4 clusters of sizes 15, 15, 15, 15

Cluster means:
          x         y
1 -2.377579  3.407220
2  3.407220 -2.377579
3 -2.999694  2.088840
4  2.088840 -2.999694
```

```
Clustering vector:
 [1] 1 1 3 3 1 1 1 3 3 3 3 1 1 1 1 1 3 1 1 1 3 3 3 3 3 3 1 3 3 1 2 4 4 2 4 4 4 4
[39] 4 4 2 2 2 4 2 2 2 2 2 4 4 4 4 2 2 2 4 4 2 2

Within cluster sum of squares by cluster:
[1] 15.18937 15.18937 11.23022 11.23022
 (between_SS / total_SS =  94.6 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
```

```
plot(clustering, col=km_2$cluster)
points(km_2$centers, col="blue",pch=15,cex=2)
```



library(ggplot2) masked_dataset <- cbind(data=clustering,mask=km_1*cluster*)*print*(*km*$_1$cluster) print(masked_dataset) ggplot(data=masked_dataset)+ aes(x,y,col=mask)+ geom_point()

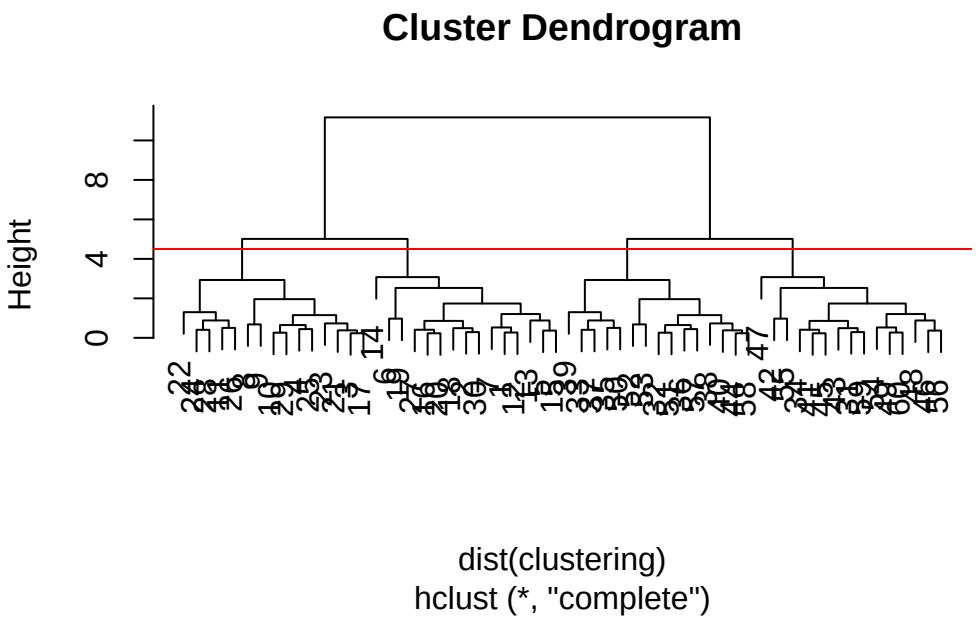
Some other ways you can go about performing clustering is through `hclust()`, which performs hierarchical clustering, as well as `dist()` which creates a distance matrix

Hierarchical clustering

```r
#there are two main functions:
#dist matrix is required for a hierarchical clustering
#hclust()
#dist()
hc <- hclust(dist(clustering))
hc
```

```
Call:
hclust(d = dist(clustering))

Cluster method   : complete
Distance         : euclidean
Number of objects: 60
```

```r
#hc has labeles as: hc$merge,height,order,labels,method,call
plot(hc)
abline(h=4.5,col="red")
```



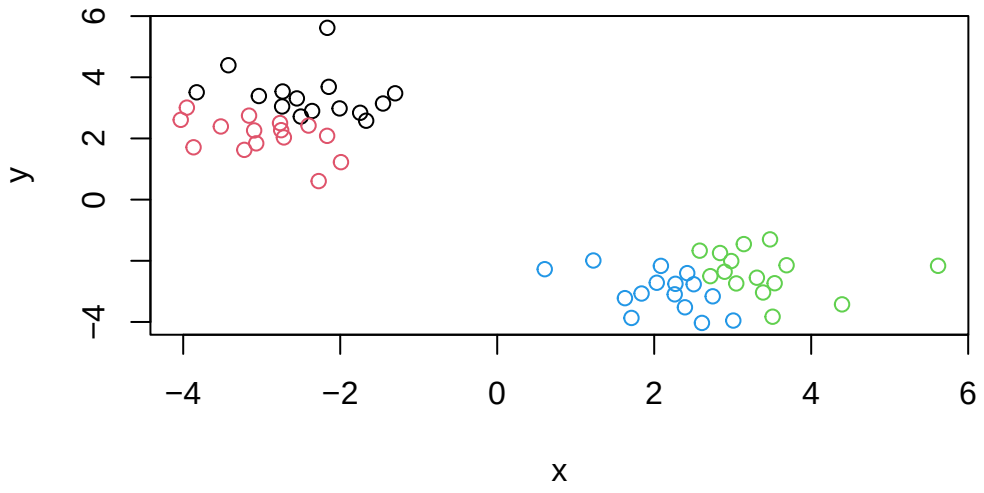
To get access to cluster memberships you can cut the tree which yields branches. To do so

you can utilize the height functionality and cut the data off into groups at a specific height. `cutree()`

```
cutvaluesmembershipvector <- cutree(hc,h=4.5)
cutvaluesmembershipvector
```

```
 [1] 1 1 2 2 1 1 1 2 2 2 2 1 1 1 1 1 2 1 1 1 2 2 2 2 2 2 1 2 2 1 3 4 4 3 4 4 4 4
[39] 4 4 3 3 3 4 3 3 3 3 3 4 4 4 4 3 3 3 4 4 3 3
```

```
#We can now plot this
plot(clustering, col=cutvaluesmembershipvector)
```



Dimensionality reduction, visualization and 'structure' analysis

Hands on with principal component analysis

You can use a principal component to describe your data in a better way that properly uses the coordinate system.

It does so in an orthogonal (perpendicular way)

You could *theoretically* utilize components that do not create orthogonal means of analysis.

PCA reduces dimensional Visualizes multidimensional data Chooses the most useful variables Identifies groupings of objects Helps you identify out liars.

Lab Worksheet

First we need to obtain the data

```
url <- "https://tinyurl.com/UK-foods"
uk_food <- read.csv(url)
```

Q1. How many rows and columns are in your new data frame named x? What R functions could you use to answer this questions?

```
ncol(uk_food)
```

```
[1] 5
```

```
nrow(uk_food)
```

```
[1] 17
```

```
dim(uk_food)
```

```
[1] 17  5
```

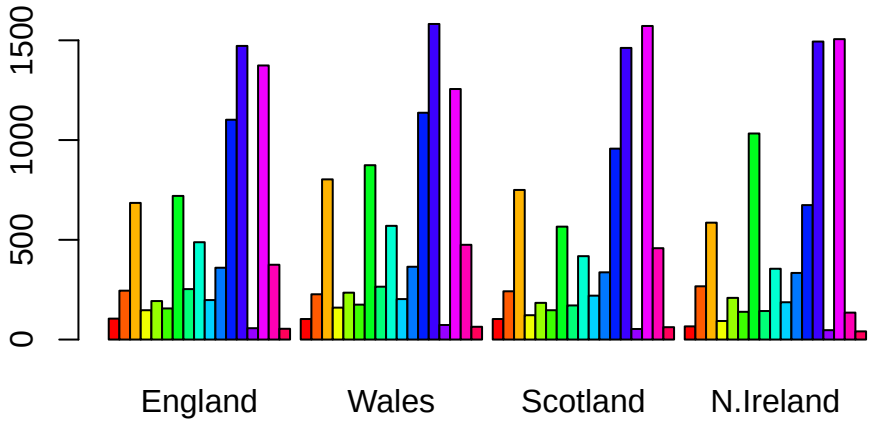
```
uk_food_w_headers <- read.csv(url,row.names = 1)
dim(uk_food_w_headers)
```

```
[1] 17  4
```

Q2. Which approach to solving the 'row-names problem' mentioned above do you prefer and why? Is one approach more robust than another under certain circumstances?

I prefer the row.names being called when you read the csv. I think this method is more robust because you can run it multiple times. If you perform the other one then it will start to remove many points off the top of your data.

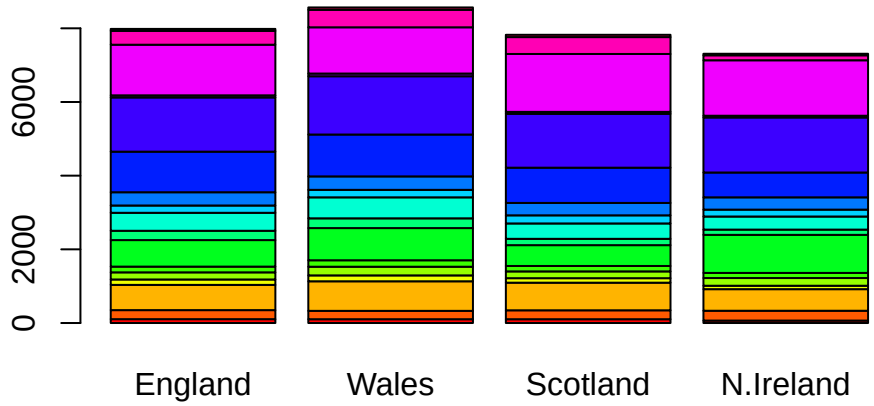
```
barplot(as.matrix(uk_food_w_headers), beside=T, col=rainbow(nrow(uk_food_w_headers)))
```



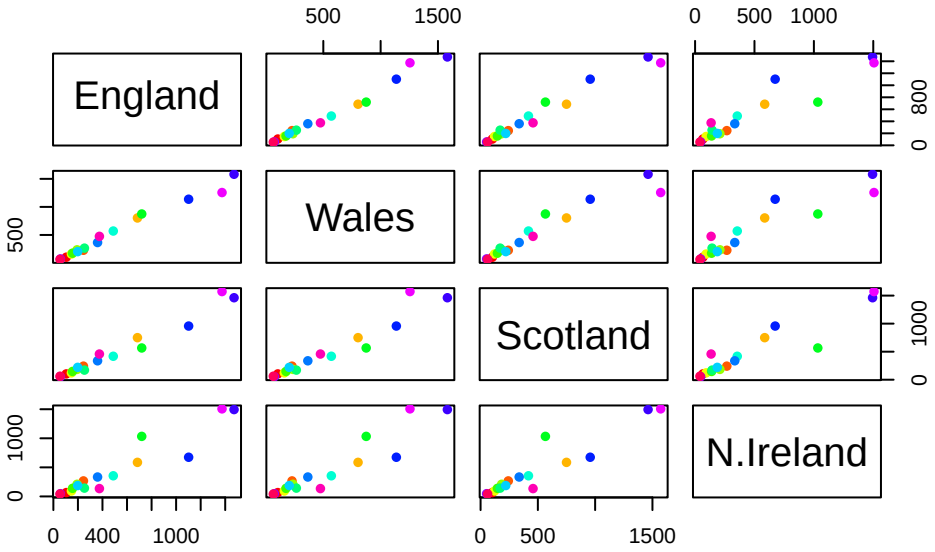
Q3: Changing what optional argument in the above barplot() function results in the following plot?

If you want to modify the plot to the desired plot, you need to plot it with beside= FALSE

```
barplot(as.matrix(uk_food_w_headers), beside=FALSE, col=rainbow(nrow(uk_food_w_headers)))
```




```
pairs(uk_food_w_headers, col=rainbow(17), pch=16)
```



Q4: Generating all pairwise plots may help somewhat. Can you make sense of the following code and resulting figure? What does it mean if a given point lies on the diagonal for a given plot?

This plot is essentially a fold change plot which compares two larger groups and analyzes specific characteristics within those two groups and measures which one is higher or lower when comparing the two.

This creates a matrix of scatter plots which looks for pair relationships between certain characteristics / labels within the data set and their larger groups. It looks at all possible pairs of the larger groups and compares if one has a specific characteristic/label that is more highly correlated with a specific larger group. As seen in plot 1,1 (bottom left), there is a blue point that is far off the line, which means that it is more important and has higher correlation in the larger group of England when compared to N.Ireland. Similarly there is an orange dot off the line, which means that the orange category is more tightly correlated and different than compared to England. This means that you can look at all possible pairs and try to see what specific variables are the most different between two countries.

This plot is limited in how many graphs you can show and analyze.

If a point is on a diagonal for a given plot it means the two values are equal compared between the two larger groups.

```
row.names(uk_food_w_headers)
```

```
 [1] "Cheese"              "Carcass_meat "       "Other_meat "
 [4] "Fish"                "Fats_and_oils "      "Sugars"
 [7] "Fresh_potatoes "     "Fresh_Veg "          "Other_Veg "
[10] "Processed_potatoes " "Processed_Veg "      "Fresh_fruit "
[13] "Cereals "            "Beverages"           "Soft_drinks "
[16] "Alcoholic_drinks "   "Confectionery "
```

```
col_vector<-rainbow(10)
col_vector
```

```
 [1] "#FF0000" "#FF9900" "#CCFF00" "#33FF00" "#00FF66" "#00FFFF" "#0066FF"
 [8] "#3300FF" "#CC00FF" "#FF0099"
```

Q5: What is the main differences between N. Ireland and the other countries of the UK in terms of this data-set?

We cant answer it with our current tool set, so we will answer it with PCA.

To do so you need the `prcomp()`. However you must first transpose the data.frame matrix with `t()` function which swaps Y with X

```
pca<-prcomp((t(uk_food_w_headers)))
summary(pca)
```

```
Importance of components:
                            PC1      PC2      PC3       PC4
Standard deviation     324.1502 212.7478 73.87622 5.552e-14
Proportion of Variance   0.6744   0.2905  0.03503 0.000e+00
Cumulative Proportion    0.6744   0.9650  1.00000 1.000e+00
```

This shows that our PC1 obtains 67% of the variance. The cumulative proportion shows that PC1 and PC2 gives us 96.5% of all variance.

A new plot against these two variables will cover about 96% of all variance.

Q6. Complete the code below to generate a plot of PC1 vs PC2. The second line adds text labels over the data points.

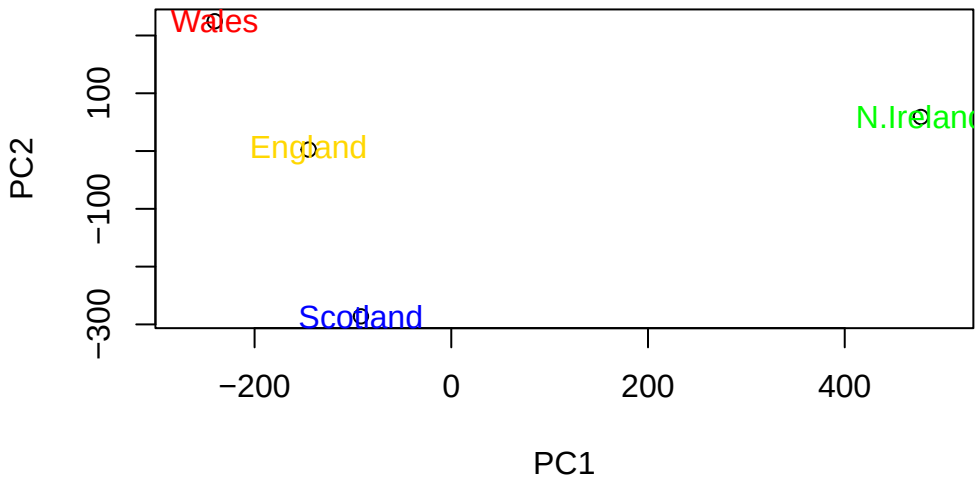
```
pca$x[,1] #gives us the values of PC1 for our data set
```

```
   England      Wales   Scotland  N.Ireland
-144.99315 -240.52915  -91.86934  477.39164
```

```
pca$x[1,] #gives us all englands values
```

```
          PC1           PC2           PC3           PC4
-1.449932e+02  2.532999e+00 -1.057689e+02  1.042460e-14
```

```
plot(pca$x[,1], pca$x[,2], xlab="PC1", ylab="PC2", xlim=c(-270,500))

#Q8. Customize your plot
#so that the colors of the country names
#match the colors in our UK and Ireland map
text(pca$x[,1], pca$x[,2], colnames(uk_food_w_headers),col =c("gold","red","blue","green")
```




```
par(mar=c(10, 3, 0.35, 0))
barplot( pca$rotation[,1], las=2 )
```