~~~
---
title: "Machine Learning 1"
author: "Ziyuan_Han"
date: "10/22/2021"
output:
  html_document:
    df_print: paged
---

#Clustering methods
Kmeans clustering in R is done with the `kmeans() `function.
Here we makeup some data to test and learn with .

```{r}
tmp <- c(rnorm(30,3),rnorm(30,-3))
data <-cbind(x = tmp, y=rev(tmp)) #rev function revers the order of the
dataset
data
hist(data)
plot(data)
```
Run `kmeans() ` set k to 2 nstart 20. The thing with Kmeans is you have to
tell it how many clusters you want.
```{r}
km<- kmeans(data, centers = 2, nstart =20) #the cluster means is the
central
km$cluster #explore the km output parameters using "$"
km$ifault
km$size #Q: how many points are in each cluster
#Q what component of your result object details cluster assignment/
membership?
km$cluster
#Q What 'component' of your result object details cluster center
km$centers
#Q plot x colored by the kmeans cluster assignment and add cluster centers
as blue points?
plot(data, col =km$cluster)
points(km$centers,col="blue",pch=15,cex=2)

```


#hclust: Hierarchical Clustering
```{r}
hc <- hclust(dist(data))
hc
plot(hc)
```
To find our membership vector we need to "cut" the tree and for this we use
the `cutress()` fucntion adn tell it the height to cut at.
```{r}
hc <- hclust(dist(data))
hc
plot(hc)
abline(h=7,col="red")
~~~

~~~
#we can also use cutree() data sate the number of k clusters we want
grps<-cutree(hc,k=2)
plot(data,col=grps)
#kmeans(x,centers=?)  hclust(dist(x))
```

#PCA : principle component analysis

##PCA UK food data
```{r}
url <- "https://tinyurl.com/UK-foods"
x <- read.csv(url,row.names = 1)
dim(x)
head(x)
rownames(x) <- x[,1]
x <- x[,-1]
x
barplot(as.matrix(x), col=rainbow(13),beside = TRUE)
##making a pairs plot
mycol<- rainbow(nrow(x))
pairs(x,col=mycol,pch=3)
```
##PCA to the rescue !
Here we will use the base R function for PCA, which is called `prcomp()`.

```{r}
t(x) #transpose the data ; needs to put country in the row
# prcomp(x)
pca <-prcomp(t(x))
summary(pca)
plot(pca)
attributes(pca)
plot(pca$x[,1:2])
text(pca$x[,1:2], labels=colnames(x))
#we can also examine the PCA "loadings", which tell us how much the
original variables contribute to each new PC
barplot(pca$rotation[,1],las=2)
```
##one more PCA for today
```{r}
url2 <- "https://tinyurl.com/expression-CSV"
rna.data <- read.csv(url2, row.names=1)
head(rna.data)
dim(rna.data)
head(rna.data)
## Again we have to take the transpose of our data
pca.rna <- prcomp(t(rna.data), scale=TRUE)
## Simple un polished plot of pc1 and pc2
plot(pca.rna$x[,1], pca.rna$x[,2], xlab="PC1", ylab="PC2")
summary(pca.rna)
plot(pca.rna$x[,1:2], main="Quick scree plot")
text(pca.rna$x[,1:2],labels = colnames(rna.data))
## Variance captured per PC
~~~

```
pca.rnavar <- pca.rna$sdev^2
## Percent variance is often more informative to look at
pca.rnavar.per <- round(pca.rnavar/sum(pca.rnavar)*100, 1)
pca.rnavar.per
```