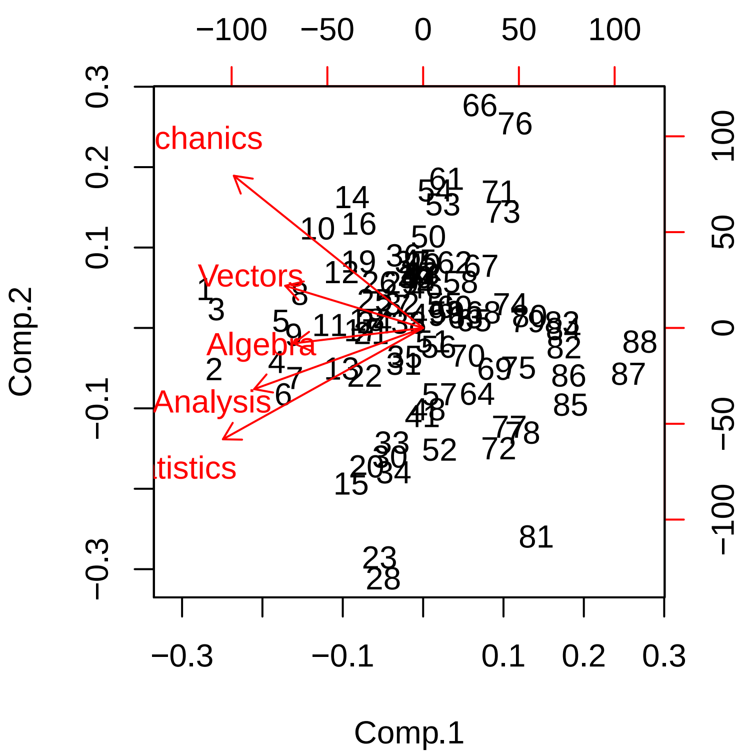
PCA: applicazioni

Dati Marks

1. Importare i dati ed effettuare l'analisi delle componenti principali dei dati centrati $\underset{n\times p}{\tilde{X}}$, calcolando la matrice dei punteggi $\underset{n\times p}{Y} = \underset{n\times p}{\tilde{X}}\underset{p\times p}{V}$ e la matrice dei pesi $\underset{p\times p}{V}$. Costruire il *biplot* e commentare.

```
rm(list=ls())

# carico i dati da una pagina internet
marks <- read.table("http://www.maths.leeds.ac.uk/~charles/mva-data/openclosedbook.dat",header = TRUE)

X = as.matrix(marks)
# assegno i nomi alle variabili
colnames(X) <- c("Mechanics", "Vectors", "Algebra", "Analysis", "Statistics")
n = nrow(X)
p = ncol(X)

# PCA
pca = princomp(X, cor=F)
# matrice dei pesi
V = pca$loadings
# matrice dei punteggi
Y = pca$scores

# biplot
biplot(pca)
```



2. Calcolare la correlazione tra il voto centrato sullo 0 dell'esame in `Statistics` e i punteggi della prima

componente principale

```
Xtilde <- scale(X,center=T,scale=F)[,]
S = (1/n)*t(Xtilde)%*%Xtilde

V[5,1]*pca$sdev[1]/sqrt(diag(S)[5])
```

```
##     Comp.1
## -0.8121103
```

```
cor(Xtilde[,"Statistics"],Y[,1])
```

```
## [1] -0.8121103
```

3. Scegliere il numero q di componenti principali utilizzando i criteri (i) proporzione di varianza spiegata dalle prime q componenti superiore all'80%; (ii) varianza spiegata da ciascuna componente maggiore di $c = \frac{1}{p}\sum_{j=1}^{p}\lambda_j$ (iii) utilizzando lo *scree plot*.

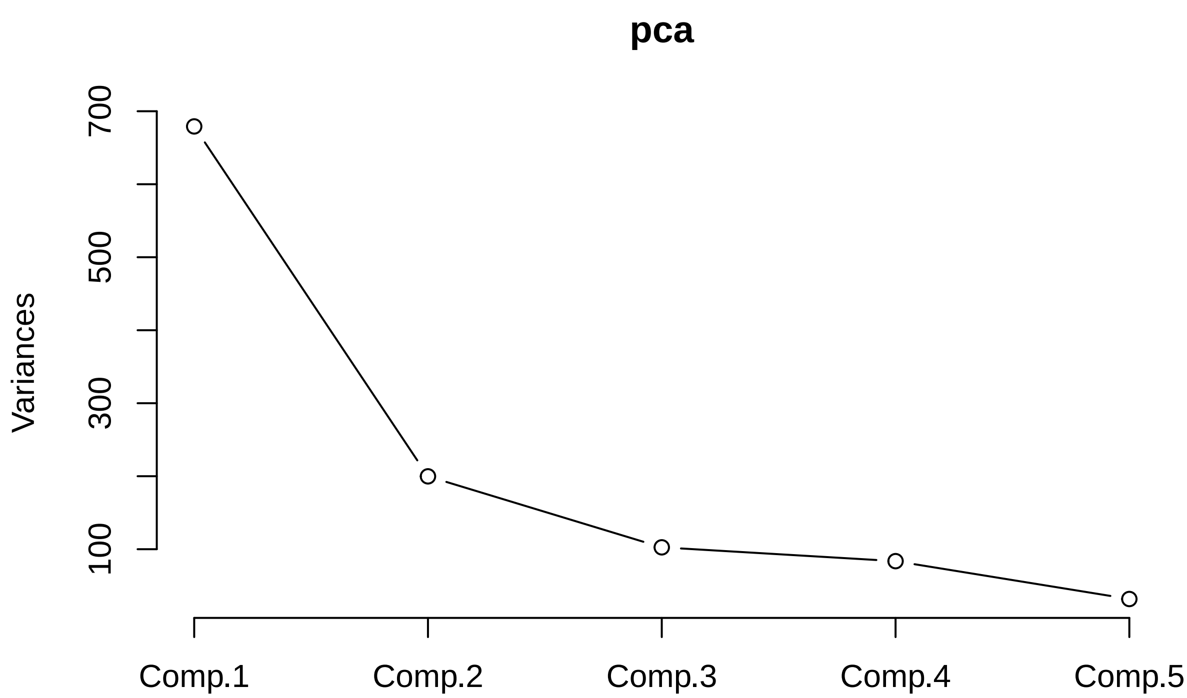
```
# scelta di q: proporzione di varianza spiegata > 80%
summary(pca)
```

```
## Importance of components:
##                           Comp.1     Comp.2      Comp.3     Comp.4
## Standard deviation     26.061142 14.1355705 10.12760414 9.14706148
## Proportion of Variance  0.619115  0.1821424  0.09349705 0.07626893
## Cumulative Proportion   0.619115  0.8012575  0.89475453 0.97102347
##                            Comp.5
## Standard deviation     5.63807655
## Proportion of Variance 0.02897653
## Cumulative Proportion  1.00000000
```

```
# scelta di q: varianza spiegata > c
c = mean(pca$sdev^2)
pca$sdev^2 > c
```

```
## Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
##   TRUE  FALSE  FALSE  FALSE  FALSE
```

```
# scelta di q: scree plot
plot(pca, type="line")
```



Dati Wine

1. Importare i dati ed effettuare l'analisi delle componenti principali dei dati standardizzati $\underset{n\times p}{Z}$ (escludendo la variabile `type`), calcolando la matrice dei punteggi $\underset{n\times p}{Y} = \underset{n\times p}{X}\underset{p\times p}{Z}$ e la matrice dei pesi $\underset{p\times p}{V}$. Calcolare la percentuale di varianza spiegata dalle prime q componenti principali, per $q = 1, \ldots, 10$.

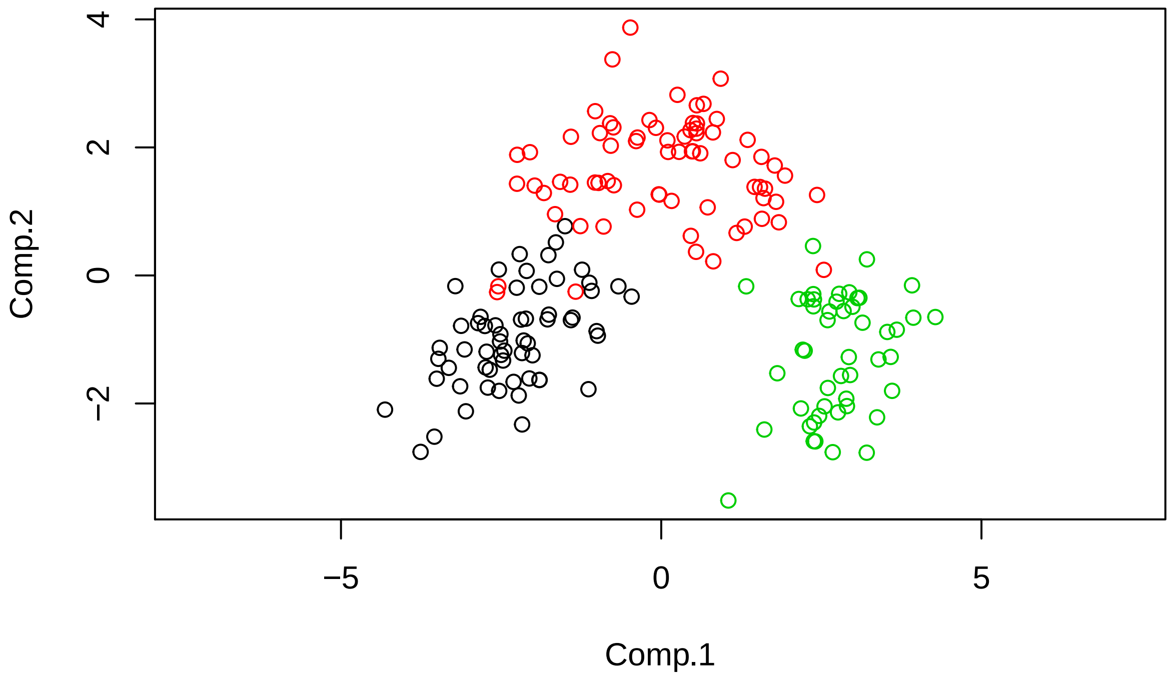
```
rm(list=ls())

# carico i dati da una pagina internet
load("/Users/aldosolari/Documents/mygithub/AE/data/wine/wine.Rdata")

X = as.matrix(wine[,-1])
n = nrow(X)
p = ncol(X)

# PCA
pca = princomp(X, cor=T)

V = pca$loadings
Y = pca$scores

# prop. di varianza spiegata dalle prime q cp
q <- 10
cumsum(pca$sdev^2/sum(pca$sdev^2))[1:q]
```

```
##    Comp.1    Comp.2    Comp.3    Comp.4    Comp.5    Comp.6    Comp.7
## 0.3619885 0.5540634 0.6652997 0.7359900 0.8016229 0.8509812 0.8933680
##    Comp.8    Comp.9   Comp.10
## 0.9201754 0.9423970 0.9616972
```

2. Costruire il diagramma di dispersione dei punteggi delle prime due componenti principali, colorando i

punti con colori diversi a seconda della tipologia di vino (variabile `type`).

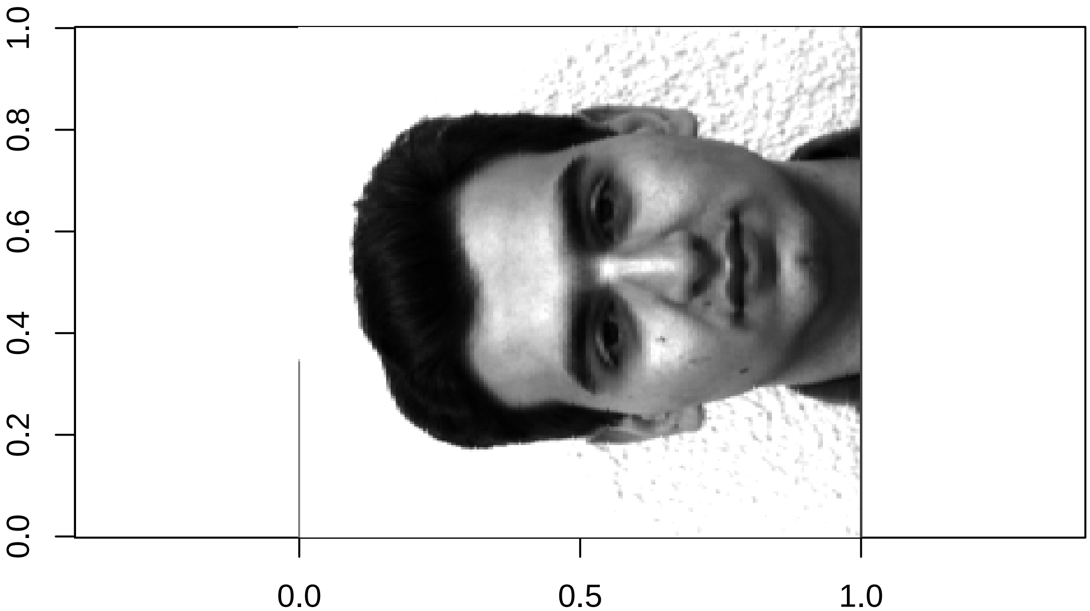
```
plot(Y[,1:2], col=as.numeric(wine$Type), asp=1)
```



Dati Face

1. Importare i dati e visualizzare l'immagine con il comando `image`. Effettuare l'analisi delle componenti principali dei dati centrati $\underset{n\times p}{\tilde{X}}$, calcolando la matrice dei punteggi $\underset{n\times p}{Y} = \underset{n\times p}{\tilde{X}}\underset{p\times p}{V}$ e la matrice dei pesi $\underset{p\times p}{V}$.

```
rm(list=ls())
load("/Users/aldosolari/Documents/mygithub/AE/data/face/face.Rda")

X = face
n = nrow(face)
p = ncol(face)

# visualizza immagine
image(X, col=gray(0:255/255), asp=p/n)
```



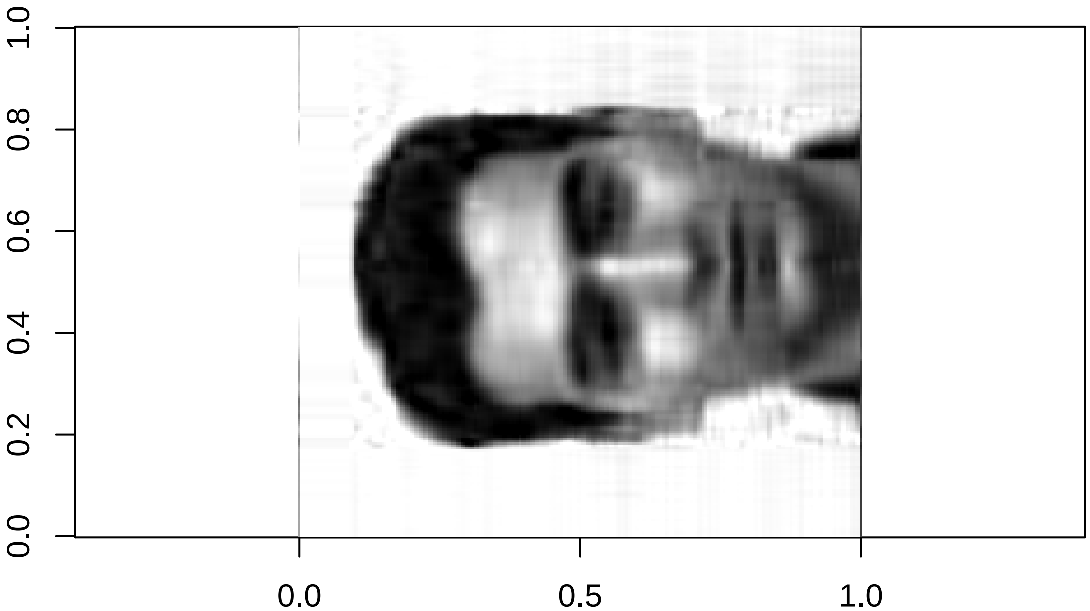

```
# PCA
pca = princomp(X, cor=F)
V = pca$loadings
Y = pca$scores
xbar = matrix(pca$center, ncol=1)
```

2. Ottenere la migliore approssimazione per $\underset{n\times p}{\tilde{X}}$ di rango q, $\underset{n\times p}{A_q} = \underset{n\times q}{Y_q}\underset{q\times p}{V_q'}$, con $q = 10$. Costruire l'immagine compressa $\underset{n\times p}{C} = \underset{n\times p}{A_q} + \underset{n\times 1}{1}\underset{1\times p}{\bar{x}}$, assicurandosi che tutti gli elementi di $\underset{n\times p}{C}$ siano compresi tra 0 e 1.

```
q = 10
Yq = Y[,1:q]
Vq = V[,1:q]

# migliore approssimazione di rango q
Aq = Yq %*% t(Vq)

# compressione immagine
one.n = matrix(rep(1,n), ncol=1)
face2 = Aq + one.n %*% t(xbar)

# forzo i valori tra 0 e 1
face2 <- pmax(pmin(face2, 1), 0)
```

3. Visualizzare l'immagine compressa e confrontarla con l'immagine originale calcolando il fattore di riduzione in termini di pixels e bytes (utilizzando il comando `object.size`)

```
# visualizza immagine compressa
image(face2, col=gray(0:255/255), asp=p/n)
```




```r
# salve immagine compressa
# library(png)
# writePNG(face,"face.png")

# confronta pixels utilizzati

pixels = prod(dim(face))
pixels2 = prod(dim(Yq)) + prod(dim(Vq)) + prod(dim(xbar))
round(pixels/pixels2, 2) # fattore di riduzione
```

```
## [1] 11.02
```

```r
# confronta memoria utilizzata
size = object.size(face)
size2 = object.size(Yq) + object.size(Vq) + object.size(xbar)
round( size/size2, 2) # fattore di riduzione
```

```
## 10.4 bytes
```