

# Cluster Analysis

Alice Cappella

Per effettuare la *cluster analysis* sul dataset `cars` possiamo percorrere due strade differenti:

1. selezionare le variabili quantitative e qualitative maggiormente legate alla variabile di interesse (`prezzo_auto`) ed applicare metodologie che consentono la presenza di variabili di natura mista;
2. selezionare solamente le variabili quantitative maggiormente correlate con `prezzo_auto` in modo tale da poter sfruttare diversi metodi di *clustering*.

Percorriamo innanzitutto la prima strada e, dopo aver calcolato la correlazione tra le variabili quantitative e l'indice  $\eta^2$  per quelle qualitative, selezioniamo solamente le variabili più legate con `prezzo_auto`.

```
library(dplyr)
library(reshape2)

cars = read.csv("cars-clean-v2-imputed.csv")
cars = cars %>%
  mutate_if(is.character, as.factor) %>%
  mutate(across(.cols = where( ~ n_distinct(.) < 6), as.factor))
cars_numeric = cars %>%
  select(where(is.numeric))
corr = cor(cars_numeric)
corr
```

```
##           prezzo_auto  anno_prod chilometraggio      kW      cv
## prezzo_auto    1.000000000  0.63752274   -0.51652262  0.52384490  0.52440789
## anno_prod      0.637522737  1.00000000   -0.73344857  0.01276064  0.01329855
## chilometraggio -0.516522624 -0.73344857    1.00000000  0.14327581  0.14368187
## kW             0.523844905  0.01276064    0.14327581  1.00000000  0.99996172
## cv             0.524407894  0.01329855    0.14368187  0.99996172  1.00000000
## cilindrata     0.158240706 -0.29655606    0.42460803  0.58181038  0.58228208
## peso           0.485241771  0.01230163    0.19755467  0.76182209  0.76259711
## emissioni      -0.003456499 -0.23685878    0.22524855  0.35679034  0.35668196
## consumi        -0.006565334 -0.17318064    0.06380238  0.20449605  0.20407614
##           cilindrata      peso      emissioni      consumi
## prezzo_auto    0.15824071  0.48524177 -0.003456499 -0.006565334
## anno_prod      -0.29655606  0.01230163 -0.236858781 -0.173180639
## chilometraggio 0.42460803  0.19755467  0.225248552  0.063802384
## kW             0.58181038  0.76182209  0.356790341  0.204496047
## cv             0.58228208  0.76259711  0.356681964  0.204076143
## cilindrata     1.00000000  0.66065242  0.287018630  0.057074325
## peso           0.66065242  1.00000000  0.304695758  0.098770857
## emissioni      0.28701863  0.30469576  1.000000000  0.730125752
## consumi        0.05707432  0.09877086  0.730125752  1.000000000
```

Prendendo come soglia 0.3 possiamo notare che le variabili più correlate con la variabile di interesse risultano: `anno_prod`, `chilometraggio` (negativamente), `kW` e `cv` (di cui ne verrà selezionata solo una dato che riportano informazioni simili) e `peso`.

Procediamo con il calcolo dell'indice  $\eta^2$ .

```
var_qualitative = names(select(cars,where(is.factor)))
eta2_results = sapply(var_qualitative,function(var) {
  eta2(cars$prezzo_auto,cars[[var]])})
eta2_df = data.frame(variabale = names(eta2_results),eta2 = eta2_results)
eta2_df[which(eta2_df$eta2 > 0.3),]
```

```
##      variabile      eta2
## marca      marca 0.3287687
## modello    modello 0.5750314
## marce      marce 0.3118093
```

Prendendo come soglia 0.3, le variabili con un valore maggiore dell'indice sono `marca`, `modello` e `marce`.

A questo punto possiamo selezionare le variabili individuate.

```
cars_cluster = cars[,c(rownames(eta2_df)[which(eta2_df$eta2 > 0.3)],
  names(which(abs(corr)[,1] > 0.3)))]
```

Come già specificato eliminiamo la variabile `kW`.

```
cars_cluster$kW = NULL
head(cars_cluster)
```

```
##   marca modello      marce prezzo_auto anno_prod chilometraggio cv peso
## 1  Fiat  Panda        5      6900      2013      32958 69  940
## 2  Fiat  Panda    6 o piu    11000      2022      20132 69  980
## 3  Fiat  Panda Automatico    3800      2004      116000 60  935
## 4  Fiat  Panda    6 o piu    10890      2022      27685 69 1055
## 5  Fiat  Panda        5      5900      2011      98000 69  930
## 6  Fiat  Panda        5      4900      2009      99817 60  930
```

Passiamo alla standardizzazione del nuovo insieme di dati. Procediamo a passi, iniziamo con le variabili qualitative ed effettuiamo un *One-Hot Encoding*, ossia trasformiamo le variabili categoriali in variabili *dummy* una per ogni modalità che ciascuna variabile presenta.

```
cars_cluster = dummy.data.frame(cars_cluster,names = c("marca","modello","marce"))
cars_cluster = cars_cluster %>%
  mutate_if(~ all(. %in% c(0,1)),as.factor)
head(cars_cluster)
```

```
##   marcaAlfa Romeo marcaAudi marcaBMW marcaCitroen marcaCUPRA marcaDacia
## 1           0           0           0           0           0           0
## 2           0           0           0           0           0           0
## 3           0           0           0           0           0           0
## 4           0           0           0           0           0           0
## 5           0           0           0           0           0           0
## 6           0           0           0           0           0           0
##   marcaFiat marcaFord marcaHyundai marcaJeep marcaKia marcaLancia
## 1           1           0           0           0           0           0
## 2           1           0           0           0           0           0
## 3           1           0           0           0           0           0
## 4           1           0           0           0           0           0
## 5           1           0           0           0           0           0
## 6           1           0           0           0           0           0
##   marcaMercedes-Benz marcaMG marcaMINI marcaNissan marcaOpel marcaPeugeot
## 1           0           0           0           0           0           0
## 2           0           0           0           0           0           0
```

## 3	0	0	0	0	0	0
## 4	0	0	0	0	0	0
## 5	0	0	0	0	0	0
## 6	0	0	0	0	0	0
##	marcaRenault	marcaSuzuki	marcaToyota	marcaVolkswagen	modello2008	modello208
## 1	0	0	0	0	0	0
## 2	0	0	0	0	0	0
## 3	0	0	0	0	0	0
## 4	0	0	0	0	0	0
## 5	0	0	0	0	0	0
## 6	0	0	0	0	0	0
##	modello3008	modello500	modello500X	modelloA1	modelloA3	modelloAustral
## 1	0	0	0	0	0	0
## 2	0	0	0	0	0	0
## 3	0	0	0	0	0	0
## 4	0	0	0	0	0	0
## 5	0	0	0	0	0	0
## 6	0	0	0	0	0	0
##	modelloAvenger	modelloAygo-X	modelloC3	modelloC3	Aircross	modelloCaptur
## 1	0	0	0		0	0
## 2	0	0	0		0	0
## 3	0	0	0		0	0
## 4	0	0	0		0	0
## 5	0	0	0		0	0
## 6	0	0	0		0	0
##	modelloClio	modelloCompass	modelloCorsa	modelloCountryman	modelloDuster	
## 1	0	0	0	0	0	
## 2	0	0	0	0	0	
## 3	0	0	0	0	0	
## 4	0	0	0	0	0	
## 5	0	0	0	0	0	
## 6	0	0	0	0	0	
##	modelloFiesta	modelloFocus	modelloFormentor	modelloGLA	modelloGolf	modelloIoi10
## 1	0	0	0	0	0	0
## 2	0	0	0	0	0	0
## 3	0	0	0	0	0	0
## 4	0	0	0	0	0	0
## 5	0	0	0	0	0	0
## 6	0	0	0	0	0	0
##	modelloIgnis	modelloJuke	modelloKuga	modelloMokka	modelloPanda	modelloPicanto
## 1	0	0	0	0	1	0
## 2	0	0	0	0	1	0
## 3	0	0	0	0	1	0
## 4	0	0	0	0	1	0
## 5	0	0	0	0	1	0
## 6	0	0	0	0	1	0
##	modelloPolo	modelloPuma	modelloQ3	modelloQashqai	modelloRenegade	
## 1	0	0	0	0	0	
## 2	0	0	0	0	0	
## 3	0	0	0	0	0	
## 4	0	0	0	0	0	
## 5	0	0	0	0	0	
## 6	0	0	0	0	0	
##	modelloSandro	modelloSportage	modelloT-Cross	modelloT-Roc	modelloTaigo	

```
## 1      0      0      0      0      0
## 2      0      0      0      0      0
## 3      0      0      0      0      0
## 4      0      0      0      0      0
## 5      0      0      0      0      0
## 6      0      0      0      0      0
##  modelloTiguan modelloTipo modelloTonale modelloTucson modelloVitara modelloX1
## 1      0      0      0      0      0
## 2      0      0      0      0      0
## 3      0      0      0      0      0
## 4      0      0      0      0      0
## 5      0      0      0      0      0
## 6      0      0      0      0      0
##  modelloYaris modelloYaris Cross modelloYpsilon modelloZS marce5 marce6 o piu
## 1      0      0      0      0      1      0
## 2      0      0      0      0      0      1
## 3      0      0      0      0      0      0
## 4      0      0      0      0      0      1
## 5      0      0      0      0      1      0
## 6      0      0      0      0      1      0
##  marceAutomatico prezzo_auto anno_prod chilometraggio cv peso
## 1      0      6900      2013      32958 69 940
## 2      0      11000      2022      20132 69 980
## 3      1      3800      2004      116000 60 935
## 4      0      10890      2022      27685 69 1055
## 5      0      5900      2011      98000 69 930
## 6      0      4900      2009      99817 60 930
```

Per le variabili quantitative effettuiamo la classica standardizzazione.

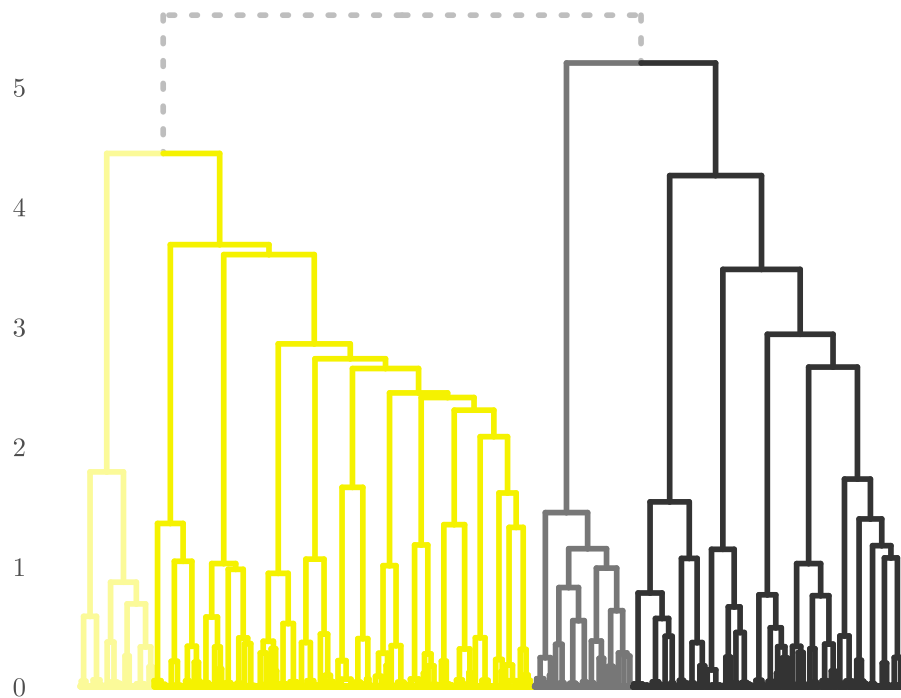
```
var_sel = c("prezzo_auto", "anno_prod", "chilometraggio", "cv", "peso")
cars_cluster[,var_sel] = scale(cars_cluster[,var_sel])
```

Possiamo ora calcolare una misura di dissimilarità per variabili di natura mista, il **coefficiente di Gower**. Si specifica che considerando sia variabili quantitative che qualitative sarà possibile solamente applicare i metodi di tipo gerarchico. In particolare, come metodo di agglomerazione utilizziamo il legame di *Ward*.

```
library(cluster)
gower = daisy(cars_cluster, metric = "gower")
hc = hclust(gower, method = "ward.D")
```

Rappresentiamo il dendrogramma da cui possiamo presumere la presenza di quattro gruppi.

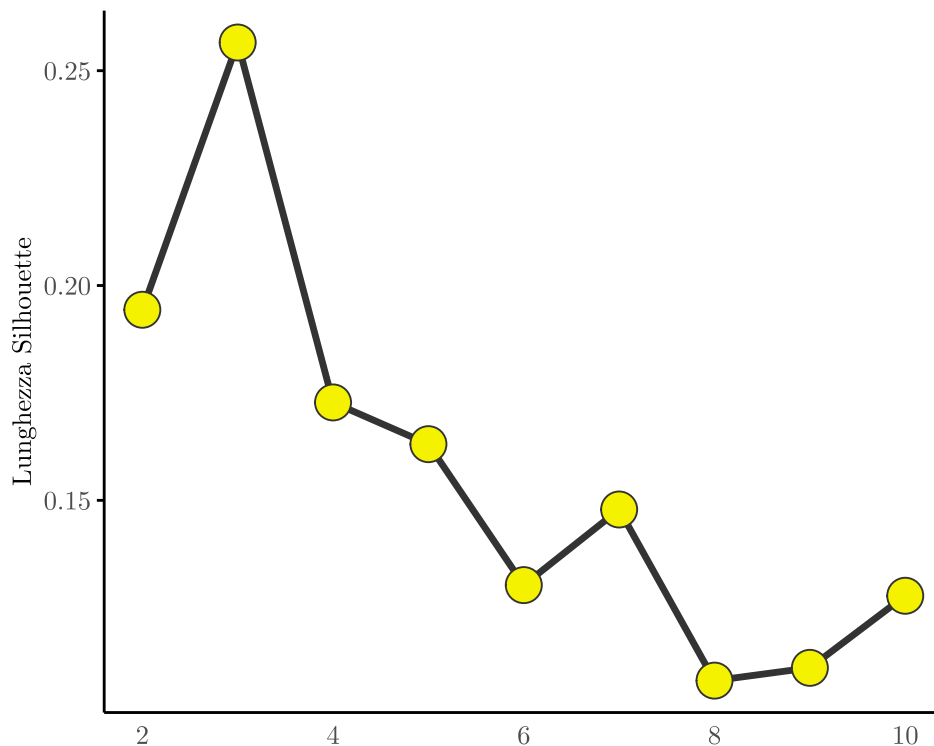
```
hc_data = dendro_data_k(hc, 4)
plot_ggdendro(hc_data, show_labels = F, direction = "tb", expand.y = 0,
              scale.color = c("grey", "#FBFA99", "#F5F200", "#777777", "#333333")) +
  theme_minimal() +
  theme(text = element_text(family = "CMUSerif"),
        axis.title = element_blank(),
        axis.text = element_text(size = 10),
        panel.grid = element_blank())
```



Non avendo la possibilità di realizzare lo *screeplot* della varianza spiegata al crescere del numero di gruppi, sfruttiamo il valore di *silhouette* che fornisce un'indicazione della coesione interna e della separazione esterna dei *cluster*.

```
sil_width = c(NA)
for(i in 2:10){
  pam_fit = pam(gower,
    diss = TRUE,
    k = i)
  sil_width[i] = pam_fit$silinfo$avg.width
}
silhouette.df = data.frame(x = 2:10,
  sil_width = sil_width[2:10])

silhouette.df %>%
  ggplot(aes(x,sil_width)) +
  geom_line(col = "#333333",size = 1.2) +
  geom_point(shape = 21,color="#333333",fill = "#F5F200",size = 6) +
  ylab("Lunghezza Silhouette") +
  theme_minimal()+
  theme(axis.text.x = element_text(size = 10),
    axis.ticks.x = element_blank(),
    axis.text.y = element_text(size = 10),
    axis.title.y = element_text(size = 10),
    axis.title.x = element_blank(),
    text = element_text(family = "CMUSerif"),
    panel.grid = element_blank(),
    axis.line = element_line(colour = "black"),
    axis.ticks = element_line(colour = "black"))
```



Utilizzando questa metrica, si può notare un picco in corrispondenza di tre gruppi.

Percorriamo ora la seconda strada e concentriamoci sulle variabili quantitative. Oltre a quelle già identificate in precedenza, creiamo un'altra variabile relativa alla somma del numero di *optional* che ogni modello riporta. Se questa nuova variabile presenta una correlazione con `prezzo_auto` superiore a 0.3 verrà inclusa nell'insieme di variabili da utilizzare per fare *clustering*.

```
cars = read.csv("cars-clean-v2-imputed.csv")
cars2 = cars[,names(which(cor(cars_numeric)[,1] > 0.3))]
cars2$kW = NULL
cars2$optional = NA
for(i in 1:nrow(cars2)){
  cars2$optional[i] = sum(as.numeric(cars[i,32:ncol(cars)]))
}
cor(cars2$prezzo_auto, cars2$optional)
```

```
## [1] 0.3134447
```

Includiamo anche la variabile `optional` appena creata e standardizziamo `cars2`.

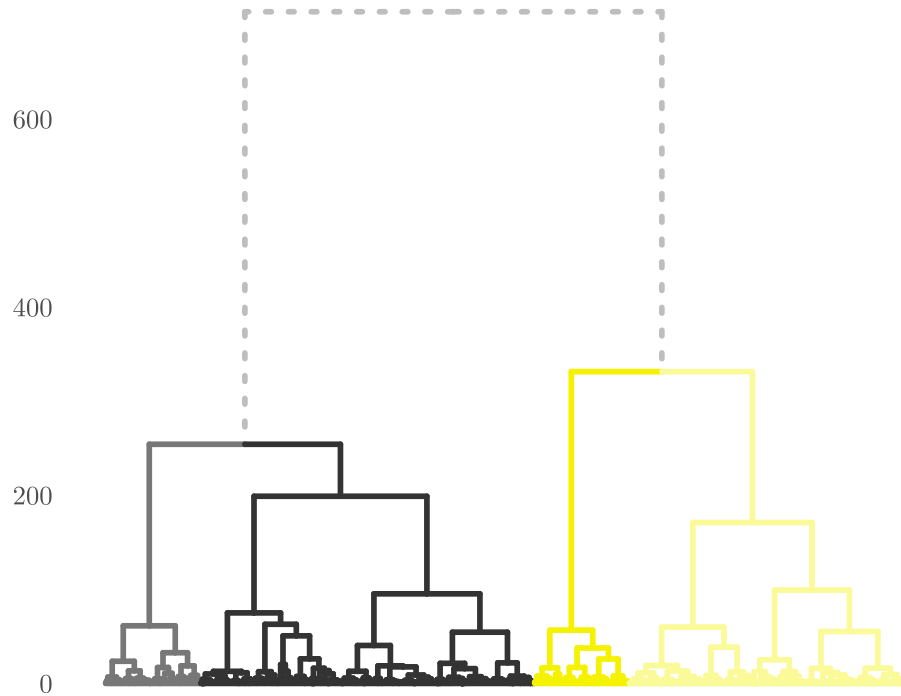
```
cars2 = scale(cars2) %>%
  as.data.frame()
```

Iniziamo con il metodo gerarchico dell'analisi dei gruppi. Calcoliamo le distanze tra le unità utilizzando la distanza euclidea mentre come metodo di agglomerazione il legame di *Ward*.

```
dist_df = dist(cars2)
hc2 = hclust(dist_df, method = "ward.D")
```

Rappresentiamo il dendrogramma da cui, anche in questo caso, si può notare la presenza di quattro gruppi.

```
hc2_data = dendro_data_k(hc2,4)
plot_ggdendro(hc2_data,show_labels = F,direction = "tb",expand.y = 0,
              scale.color = c("grey","#F5F200","#FBFA99","#333333","#777777")) +
  theme_minimal() +
  theme(text = element_text(family = "CMUSerif"),
        axis.title = element_blank(),
        axis.text = element_text(size = 10),
        panel.grid = element_blank())
```



Procediamo ora ai metodi non gerarchici e utilizziamo l'algoritmo *k-means*. Appliciamo questo algoritmo per un numero di gruppi che va da 1 a 10. In questo caso sarà possibile rappresentare lo *screplot* della varianza spiegata all'aumentare del numero di *cluster* e sfruttarlo per capire qual è il numero più adeguato di gruppi.

```
list_of_output = vector("list")
list_of_df = vector("list")
var_expl = vector()
tot_within = vector()
i = 1
n_max = 10

for(n_groups in 1:n_max) {
  list_of_output[[i]] = kmeans(cars2,centers = n_groups)
  temp = cars2
  temp$ID_Group = list_of_output[[i]]$cluster
  list_of_df[[i]] = temp
  var_expl[i] = list_of_output[[i]]$betweenss /
    list_of_output[[i]]$totss
  tot_within[i] = list_of_output[[i]]$tot.withinss
}
```

```

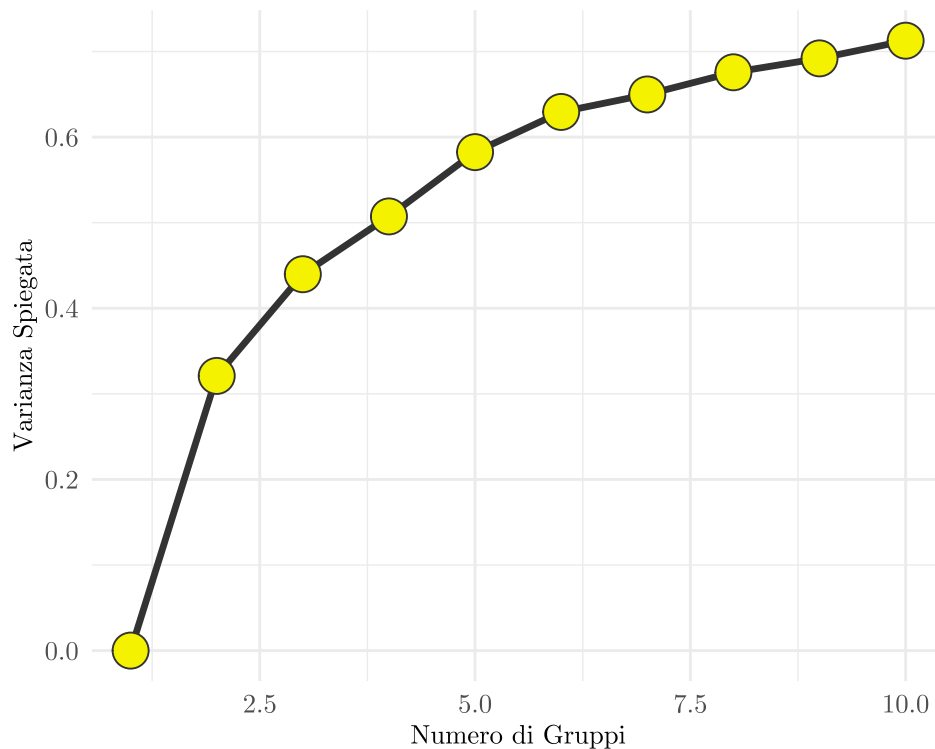
    i = i + 1
  }

  names(list_of_df) = 1:n_max
  names(list_of_output) = 1:n_max
  names(var_expl) = 1:n_max

  var_expl_df = data.frame(
    n_groups = names(var_expl),
    var_expl = unlist(var_expl)
  )
  var_expl_df$n_groups = as.numeric(as.character(var_expl_df$n_groups))

  ggplot(var_expl_df, aes(x = n_groups, y = var_expl)) +
    geom_line(col = "#333333",size = 1.2) +
    geom_point(shape = 21,color="#333333",fill = "#F5F200",size = 6) +
    labs(x = "Numero di Gruppi", y = "Varianza Spiegata")+
    theme_minimal()+
    theme(axis.text = element_text(size = 10),
          axis.title = element_text(size = 10),
          text = element_text(family = "CMUSerif"))

```

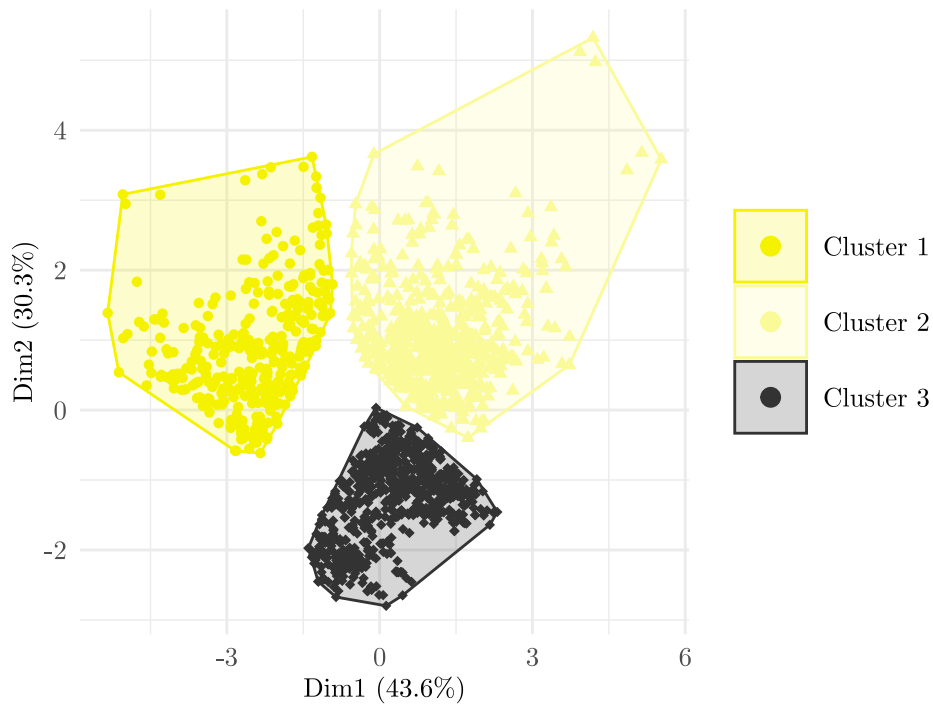


Lo *screeplot* in questa situazione non è molto di aiuto, infatti non si evidenzia la presenza di un gomito. Possiamo però provare a selezionare 3 gruppi e vedere come le unità sono suddivise in un piano a due dimensioni, ottenuto utilizzando l'analisi delle componenti principali.

```
cars2$cluster = list_of_output[[3]]$cluster
```



```
fviz_cluster(list_of_output[[3]],cars2,geom = "point",main = "")+
  scale_fill_manual(values = palette_function(3),
    labels = paste("Cluster",1:3,sep = " "))+
  scale_colour_manual(values = palette_function(3),
    labels = paste("Cluster",1:3,sep = " "))+
  scale_shape_manual(labels = rep(" ",3),values = c(16:(16 + 3)),guide = "none")+
  labs(paste("Cluster",1:3,sep = " "))+
  theme_minimal()+
  theme(axis.text.x = element_text(size = 10),
    axis.ticks.x = element_blank(),
    axis.text.y = element_text(size = 10),
    axis.title = element_text(size = 10),
    text = element_text(family = "CMUSerif"),
    legend.title = element_blank(),
    legend.text = element_text(size = 10),
    legend.key.size = unit(1,'cm'))
```



Possiamo notare che i gruppi individuati sono ben separati.

Per cercare di interpretare le due dimensioni effettuiamo l'analisi delle componenti principali e vediamo quali variabili sono rappresentate dalle prime due componenti.

```
prcomp(cars2[, -6])
```

```
## Standard deviations (1, ..., p=5):
## [1] 1.5743505 1.1475728 0.8842480 0.4939071 0.4226802
##
## Rotation (n x k) = (5 x 5):
##           PC1      PC2      PC3      PC4      PC5
## prezzo_auto 0.5443254 -0.2559500 0.32051698 -0.22680299 -0.695721733
## anno_prod   0.3084547 -0.7010317 0.25910436 0.20046724 0.553252194
## cv          0.5035831 0.4328125 0.05852018 -0.59093073 0.454371725
## peso        0.4935705 0.4432955 0.03905706 0.74722522 -0.002519635
## optional    0.3337032 -0.2433240 -0.90839624 -0.02878615 -0.058508683
```

La prima componente principale rappresenta tutte le variabili eccetto l'anno di produzione che è invece inclusa maggiormente nella seconda componente principale con segno negativo. Resta comunque complesso fornire un'interpretazione del grafico appena realizzato ma aiuta a visualizzare come le unità sono state suddivise.

Passiamo alle procedure di *clustering* basate sul modello. Stimiamo diverse mistura gaussiana con un numero di componenti che va da 1 a 5. Non consideriamo un numero superiori di componenti per parsimonia.

```
library(mclust)
```

```
## Package 'mclust' version 6.0.0
## Type 'citation("mclust")' for citing this R package in publications.
```

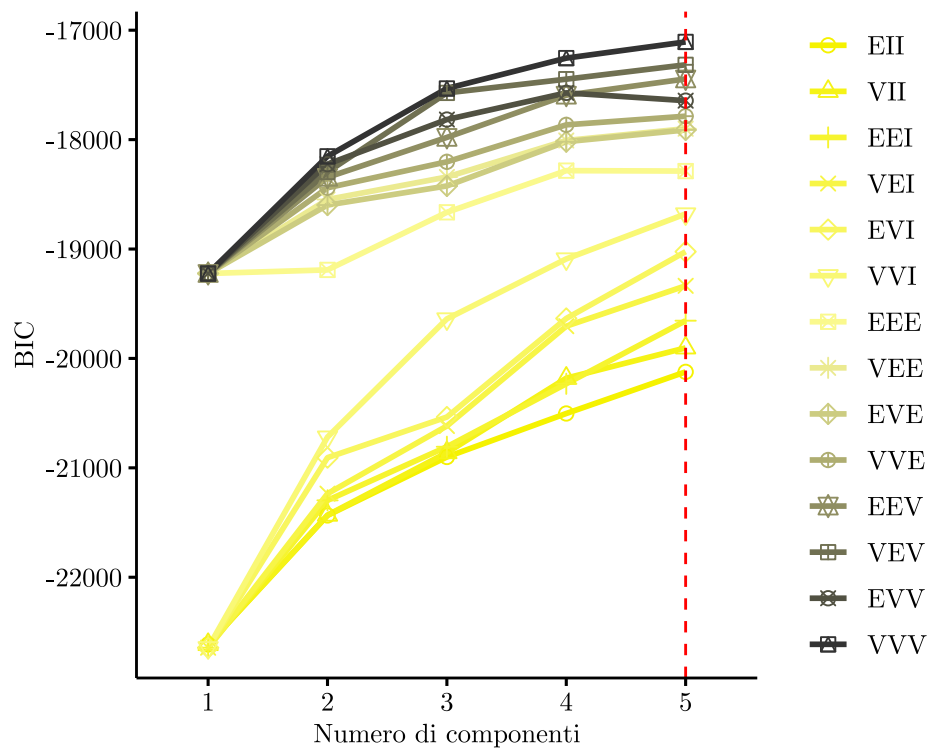
```
mbc = Mclust(cars2[, -6], G = 1:5, verbose = F)
summary(mbc)
```

```
## -----
## Gaussian finite mixture model fitted by EM algorithm
## -----
##
## Mclust VVV (ellipsoidal, varying volume, shape, and orientation) model with 5
## components:
##
## log-likelihood    n  df      BIC      ICL
##      -8170.11 1591 104 -17106.92 -17691.17
##
## Clustering table:
##   1  2  3  4  5
## 165 355 136 94 841
```

Vengono selezionate 5 componenti, ossia vengono individuati 5 gruppi. Dal *summary* possiamo poi notare che è stato selezionato il modello più flessibile (VVV), che prevede una forma ellissoidale delle componenti, con volume, forma e orientamento differente per ognuna di esse.

La struttura di matrice di varianze e covarianze delle componenti viene selezionata mediante l'indicatore BIC. Possiamo visualizzare l'andamento dell'indicatore di informazione automatica per le 14 possibili strutture di matrice di varianze e covarianze all'aumentare del numero di componenti.

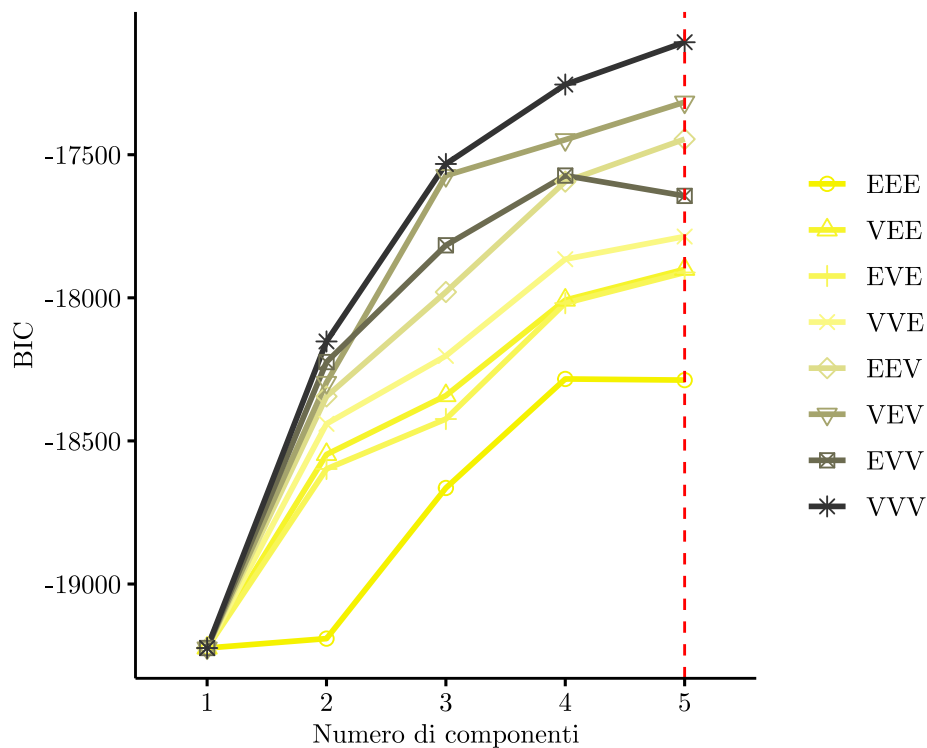
```
fviz_mclust_bic(mbc,legend = "right",shape = "model",size = 1,
               palette = palette_function(14))+
labs(x = "Numero di componenti")+
theme(legend.text = element_text(size = 10),
      axis.title = element_text(size = 10),
      axis.text = element_text(size = 10),
      title = element_blank(),
      text = element_text(family = "CMUSerif"))
```



Il BIC è riportato con segno negativo, cerchiamo quindi la struttura che massimizza l'indicatore.

Evidenziando due fasce di strutture separate, si può pensare di visualizzare solamente quelle con valore del BIC maggiore. Ristimiamo allora la mistura considerando solamente queste strutture per la matrice di varianze e covarianze delle componenti.

```
var_type = colnames(as.matrix(mbc[["BIC"]] > -2500))[7:14]
mbc2 = Mclust(cars2[, -6], G = 1:5, modelNames = var_type, verbose = F)
fviz_mclust_bic(mbc2,legend = "right",shape = "model",size = 1,
               palette = palette_function(8))+
labs(x = "Numero di componenti")+
theme(legend.text = element_text(size = 10),
      axis.title = element_text(size = 10),
      axis.text = element_text(size = 10),
      title = element_blank(),
      text = element_text(family = "CMUSerif"))
```



Il modello selezionato è, come già specificato, un modello VVV con 5 componenti. Potrebbe però essere adeguato considerare un modello di mistura con meno componenti, come ad esempio  $G = 3$  dato che il guadagno in termini di BIC ottenuto passando da 3 a 5 componenti non è elevato. In corrispondenza di tre componenti si può inoltre osservare un modello che presenta un valore del BIC molto simile a quello selezionato. Questo modello è EEV che, a differenza di VVV, prevede un uguale volume per le componenti.