# Progetto Data Mining

## INTRODUZIONE

Il seguente studio ha lo scopo prendere in analisi un dataset con l'obiettivo di ricercare, attraverso la cluster analysis, le similarità e le dissimilarità tra le variabili prese in considerazione. Le suddette variabili, che corrispondono ai valori nutrizionali, verranno selezionate in un insieme di caratteristiche generali, in questo caso relative a diverse marche di cereali da colazione. Successivamente si procederà con l'analisi delle componenti principali (PCA), al fine di ridurre il numero delle variabili latenti, limitando il più possibile la perdita di informazioni.

### *1) Cluster Analysis*

Prima di iniziare e far vedere come si effettua una Cluster Analysis vediamo cos'è.
Il clustering (o analisi dei gruppi) è un metodo di data analysis usato in diversi campi.
Avendo a disposizione un certo numero *n* di oggetti statistici, caratterizzati da *p* caratteristiche, si vuole cercare di determinare una partizione di *k* gruppi in modo tale che gli oggetti appartenenti a un gruppo siano vicini e quelli appartenenti a gruppi diversi siano lontani.
Il concetto di vicinanza o distanza, in questo caso, è associato a una quantità misurabile e oggettiva. Questo concetto viene anche inteso come concetto di similarità o dissimilarità per cui due oggetti simili sono più vicini rispetto a due oggetti dissimili che sono più lontani.
Parlando di distanza, possiamo dire che ci sono diversi tipi di distanze che permettono di fare questo studio, ad esempio la distanza Euclidea, Manhattan, Mahalanobis, ecc
In questo studio verrà utilizzata la distanza Euclidea.
Inoltre, possiamo già anticipare che utilizzeremo diversi algoritmi di classificazione per mostrare i differenti abbinamenti che si possono creare.
Quindi lo scopo del seguente studio è creare *k* cluster formati da *p* caratteristiche simili o dissimili su un *n* totale di oggetti statistici, dove *n* può essere inteso come i diversi tipi di cereali prodotti in America (prima colonna) e le *p* caratteristiche corrispondono a valori nutrizionali e altre informazioni relative ai vari tipi di cereali (prima riga).
Dal momento che questi dati vengono analizzati attraverso il software Rstudio, la prima cosa da fare è caricare il dataset scelto in R, inserendo il separatore dei valori, indicando che la prima riga è formata da testo per poi successivamente visualizzare la tabella.

**cereali<-read.csv("cereal.csv", header = TRUE, sep=",")**
**view(cereali)**

| | name | mfr | type | calories | protein | fat | sodium | fiber | carbo | sugars | potass | vitamins | shelf | weight | cups | rating |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 100% Bran | N | C | 70 | 4 | 1 | 130 | 10.0 | 5.0 | 6 | 280 | 25 | 3 | 1.00 | 0.33 | 68.40297 |
| 2 | 100% Natural Bran | Q | C | 120 | 3 | 5 | 15 | 2.0 | 8.0 | 8 | 135 | 0 | 3 | 1.00 | 1.00 | 33.98368 |
| 3 | All-Bran | K | C | 70 | 4 | 1 | 260 | 9.0 | 7.0 | 5 | 320 | 25 | 3 | 1.00 | 0.33 | 59.42551 |
| 4 | All-Bran with Extra Fiber | K | C | 50 | 4 | 0 | 140 | 14.0 | 8.0 | 0 | 330 | 25 | 3 | 1.00 | 0.50 | 93.70491 |
| 5 | Almond Delight | R | C | 110 | 2 | 2 | 200 | 1.0 | 14.0 | 8 | -1 | 25 | 3 | 1.00 | 0.75 | 34.38484 |
| 6 | Apple Cinnamon Cheerios | G | C | 110 | 2 | 2 | 180 | 1.5 | 10.5 | 10 | 70 | 25 | 1 | 1.00 | 0.75 | 29.50954 |
| 7 | Apple Jacks | K | C | 110 | 2 | 0 | 125 | 1.0 | 11.0 | 14 | 30 | 25 | 2 | 1.00 | 1.00 | 33.17409 |
| 8 | Basic 4 | G | C | 130 | 3 | 2 | 210 | 2.0 | 18.0 | 8 | 100 | 25 | 3 | 1.33 | 0.75 | 37.03856 |
| 9 | Bran Chex | R | C | 90 | 2 | 1 | 200 | 4.0 | 15.0 | 6 | 125 | 25 | 1 | 1.00 | 0.67 | 49.12025 |
| 10 | Bran Flakes | P | C | 90 | 3 | 0 | 210 | 5.0 | 13.0 | 5 | 190 | 25 | 3 | 1.00 | 0.67 | 53.31381 |
| 11 | Cap'n'Crunch | Q | C | 120 | 1 | 2 | 220 | 0.0 | 12.0 | 12 | 35 | 25 | 2 | 1.00 | 0.75 | 18.04285 |
| 12 | Cheerios | G | C | 110 | 6 | 2 | 290 | 2.0 | 17.0 | 1 | 105 | 25 | 1 | 1.00 | 1.25 | 50.76500 |
| 13 | Cinnamon Toast Crunch | G | C | 120 | 1 | 3 | 210 | 0.0 | 13.0 | 9 | 45 | 25 | 2 | 1.00 | 0.75 | 19.82357 |
| 14 | Clusters | G | C | 110 | 3 | 2 | 140 | 2.0 | 13.0 | 7 | 105 | 25 | 3 | 1.00 | 0.50 | 40.40021 |
| 15 | Cocoa Puffs | G | C | 110 | 1 | 1 | 180 | 0.0 | 12.0 | 13 | 55 | 25 | 2 | 1.00 | 1.00 | 22.73645 |
| 16 | Corn Chex | R | C | 110 | 2 | 0 | 280 | 0.0 | 22.0 | 3 | 25 | 25 | 1 | 1.00 | 1.00 | 41.44502 |
| 17 | Corn Flakes | K | C | 100 | 2 | 0 | 290 | 1.0 | 21.0 | 2 | 35 | 25 | 1 | 1.00 | 1.00 | 45.86332 |
| 18 | Corn Pops | K | C | 110 | 1 | 0 | 90 | 1.0 | 13.0 | 12 | 20 | 25 | 2 | 1.00 | 1.00 | 35.78279 |
| 19 | Count Chocula | G | C | 110 | 1 | 1 | 180 | 0.0 | 12.0 | 13 | 65 | 25 | 2 | 1.00 | 1.00 | 22.39651 |
| 20 | Cracklin' Oat Bran | K | C | 110 | 3 | 3 | 140 | 4.0 | 10.0 | 7 | 160 | 25 | 3 | 1.00 | 0.50 | 40.44877 |
| 21 | Cream of Wheat (Quick) | N | H | 100 | 3 | 0 | 80 | 1.0 | 21.0 | 0 | -1 | 0 | 2 | 1.00 | 1.00 | 64.53382 |
| 22 | Crispix | K | C | 110 | 2 | 0 | 220 | 1.0 | 21.0 | 3 | 30 | 25 | 3 | 1.00 | 1.00 | 46.89564 |
| 23 | Crispy Wheat & Raisins | G | C | 100 | 2 | 1 | 140 | 2.0 | 11.0 | 10 | 120 | 25 | 3 | 1.00 | 0.75 | 36.17620 |
| 24 | Double Chex | R | C | 100 | 2 | 0 | 190 | 1.0 | 18.0 | 5 | 80 | 25 | 3 | 1.00 | 0.75 | 44.33086 |
| 25 | Froot Loops | K | C | 110 | 2 | 1 | 125 | 1.0 | 11.0 | 13 | 30 | 25 | 2 | 1.00 | 1.00 | 32.20758 |
| 26 | Frosted Flakes | K | C | 110 | 1 | 0 | 200 | 1.0 | 14.0 | 11 | 25 | 25 | 1 | 1.00 | 0.75 | 31.43597 |
| 27 | Frosted Mini-Wheats | K | C | 100 | 3 | 0 | 0 | 3.0 | 14.0 | 7 | 100 | 25 | 2 | 1.00 | 0.80 | 58.34514 |
| 28 | Fruit & Fibre Dates; Walnuts; and Oats | P | C | 120 | 3 | 2 | 160 | 5.0 | 12.0 | 10 | 200 | 25 | 3 | 1.25 | 0.67 | 40.91705 |

Showing 1 to 29 of 77 entries

È sicuramente utile capire che tipo di dati vogliamo analizzare, quindi procediamo a fare un analisi descrittiva ed esplorativa della nostra tabella di dati.

La nostra tabella è formata da 77 righe che rappresentano i diversi tipi di cereali da colazione prodotti in America e 16 colonne che, oltre al Nome dei cereali, forniscono diverse informazioni.
Queste informazioni sono presentate nel seguente ordine:
-Manufacturer (mfr): indica il produttore
A = American Home Food Products
G = General Mills
K = Kellogg's
N = Nabisco
P = Post
Q = Quaker Oats
R = Ralston Purina
-Tipo (type): se sono Freddi(Cold) o Caldi(Hot)
-Calorie (calories)
-Proteine (protein)
-Grasso (fat)
-Sodio (sodium)
-Fibre (fiber)
-Carboidrati (carbohydrates)
-Zucchero (sugar)
-Potassio (potassium)
-Vitamine (vitamines)
-Scaffale (shelf): espositore (1, 2 o 3, contando dal pavimento)
-Peso (weight)
-Porzioni (cups)
-Valutazione (rating)
La presenza nel dataset di variabili irrilevanti ai fini dello studio comporta la scelta di creare una nuova tabella chiamata C in cui vengono considerate tutte le righe ma solo alcune colonne, corrispondenti solo ai valori nutrizionali di ogni marca.
**C=cereali[,4:12]**

| | calories | protein | fat | sodium | fiber | carbo | sugars | potass | vitamins |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 70 | 4 | 1 | 130 | 10.0 | 5.0 | 6 | 280 | 25 |
| 2 | 120 | 3 | 5 | 15 | 2.0 | 8.0 | 8 | 135 | 0 |
| 3 | 70 | 4 | 1 | 260 | 9.0 | 7.0 | 5 | 320 | 25 |
| 4 | 50 | 4 | 0 | 140 | 14.0 | 8.0 | 0 | 330 | 25 |
| 5 | 110 | 2 | 2 | 200 | 1.0 | 14.0 | 8 | -1 | 25 |
| 6 | 110 | 2 | 2 | 180 | 1.5 | 10.5 | 10 | 70 | 25 |
| 7 | 110 | 2 | 0 | 125 | 1.0 | 11.0 | 14 | 30 | 25 |
| 8 | 130 | 3 | 2 | 210 | 2.0 | 18.0 | 8 | 100 | 25 |
| 9 | 90 | 2 | 1 | 200 | 4.0 | 15.0 | 6 | 125 | 25 |
| 10 | 90 | 3 | 0 | 210 | 5.0 | 13.0 | 5 | 190 | 25 |

A questo punto tutte le informazioni necessarie per eseguire la cluster analysis possono essere reperite.
Si utilizzano i metodi:
**names(C)** che restituisce tutte le variabili selezionate nella nuova tabella;
**str(C)** che restituisce il nome, il tipo di variabili e la grandezza del dataframe, in questo caso 77 oggetti con 9 variabili di tipo quantitativo.

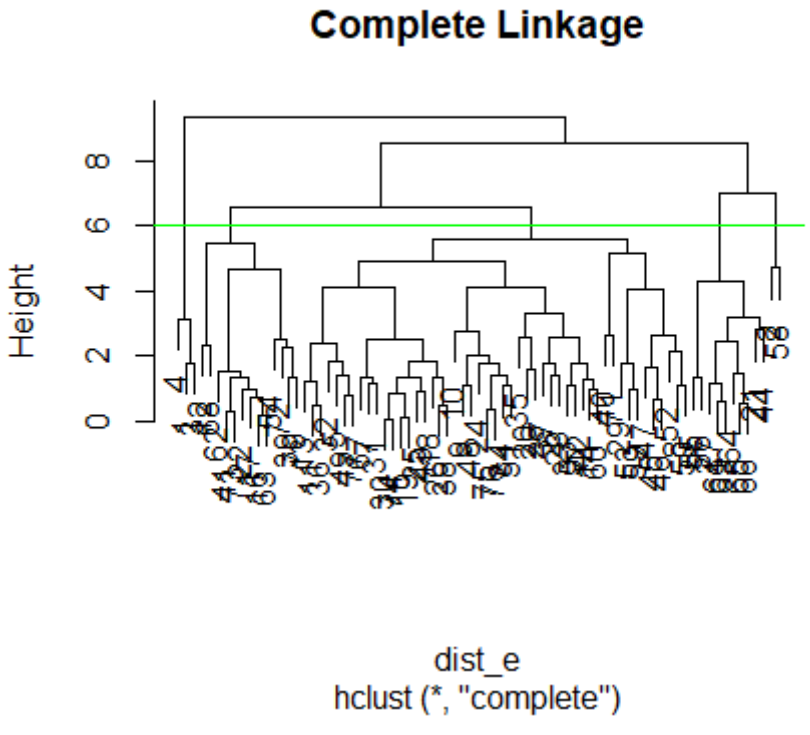
```
> names(C)
[1] "calories" "protein"  "fat"      "sodium"   "fiber"    "carbo"    "sugars"   "potass"   "vitamins"
> str(C)
'data.frame':   77 obs. of  9 variables:
 $ calories: int  70 120 70 50 110 110 110 130 90 90 ...
 $ protein : int  4 3 4 4 2 2 2 3 2 3 ...
 $ fat     : int  1 5 1 0 2 2 0 2 1 0 ...
 $ sodium  : int  130 15 260 140 200 180 125 210 200 210 ...
 $ fiber   : num  10 2 9 14 1 1.5 1 2 4 5 ...
 $ carbo   : num  5 8 7 8 14 10.5 11 18 15 13 ...
 $ sugars  : int  6 8 5 0 8 10 14 8 6 5 ...
 $ potass  : int  280 135 320 330 -1 70 30 100 125 190 ...
 $ vitamins: int  25 0 25 25 25 25 25 25 25 25 ...
>
```

C, può essere considerata come tabella di partenza da cui iniziare lo studio.
La funzione di base per implementare il clustering in R è *hclust()*. Questa funzione ha bisogno della matrice di dissimilarità che otteniamo dalla funzione *dist()* e dal *method()*, ovvero il tipo di distanza che verrà utilizzato per formare le coppie.
Per ottenere la matrice di dissimilarità, però, occorre standardizzare i dati con il metodo **standardizza=scale(C)**.
A questo punto possiamo ottenere la nostra matrice delle distanze
**dist_e=dist(standardizza)**

```
> dist_e
          1         2         3         4         5         6         7         8         9
 10
2  6.51509455
3  1.77895754 6.99132803
4  2.77081089 8.82998134 3.13537443
5  6.60758804 4.67826767 6.61258811 8.59054634
6  5.72310884 4.06540097 5.79529226 7.95038059 1.40157385
7  6.34874029 5.78015732 6.62737065 8.40986589 2.69293480 2.35828344
8  6.26818732 4.63825252 6.07244221 8.17310632 2.22698885 2.34772279 3.53601085
9  4.64920030 5.35951389 4.50920759 6.27883223 2.64695744 2.38937352 3.21166221 2.75164121
10 3.65060308 6.14112813 3.36739773 5.14716991 4.04431920 3.54424308 3.88440252 3.62802918 1.76416015
```

Il modo in cui i Cluster o Gruppi di Similarità/Dissimilarità si creano, spesso è dato dal tipo di legame che si sceglie; qui utilizzeremo i legami Completo, Medio, Singolo, Centroide e li confronteremo.
Una volta creato il primo cluster, attraverso il metodo *summary()* è utile capire l'altezza (height) e soprattutto il merge, ovvero come i dati si sono accoppiati nel nostro dendrogramma. È anche possibile accedere alla variabile $merge per vedere più nel dettaglio le coppie create.
Infine, possiamo visualizzare il plot, ovvero la rappresentazione grafica delle varie coppie.

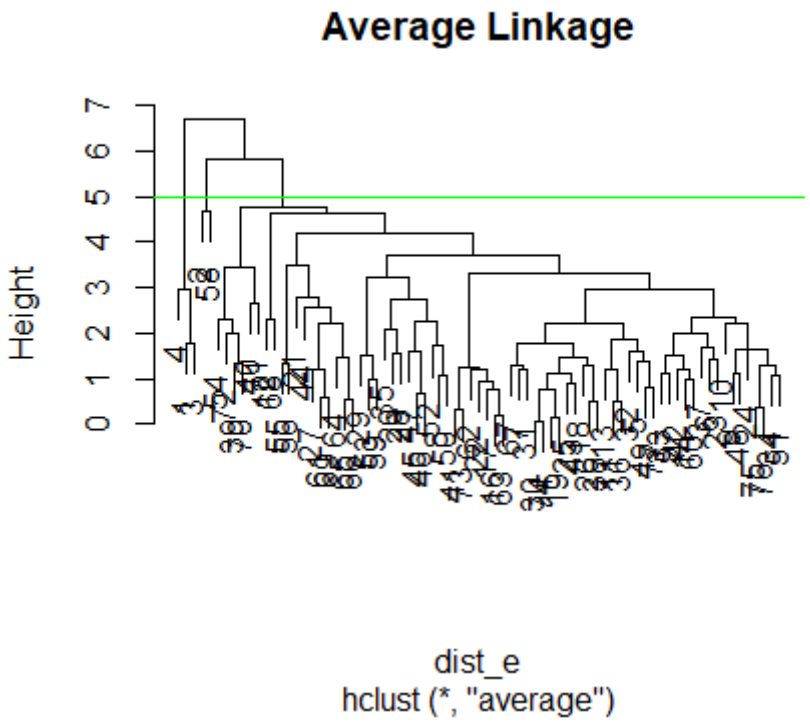
***-Metodo del legame completo***
Il metodo del legame completo (complete linkage) si basa su un criterio di distanza massima tra gli elementi che compongono i cluster.
Questo algoritmo di aggregazione predilige le differenze tra gli elementi, ovvero predilige la differenza tra alcuni elementi anziché la similarità tra gli elementi.
**legCompleto=hclust(dist_e,method = "complete")**
**summary(legCompleto)**
**legCompleto$merge**
**plot(legCompleto,main = "Complete Linkage")**
**Clusterone=cutree(legCompleto,6)**
**abline(h=6,col="green")**



***-Metodo del legame medio***

Il metodo del legame medio (average linkage) si basa su un criterio secondo cui la distanza tra cluster è definita come la media aritmetica delle distanze tra gli elementi che appartengono a due cluster diversi.

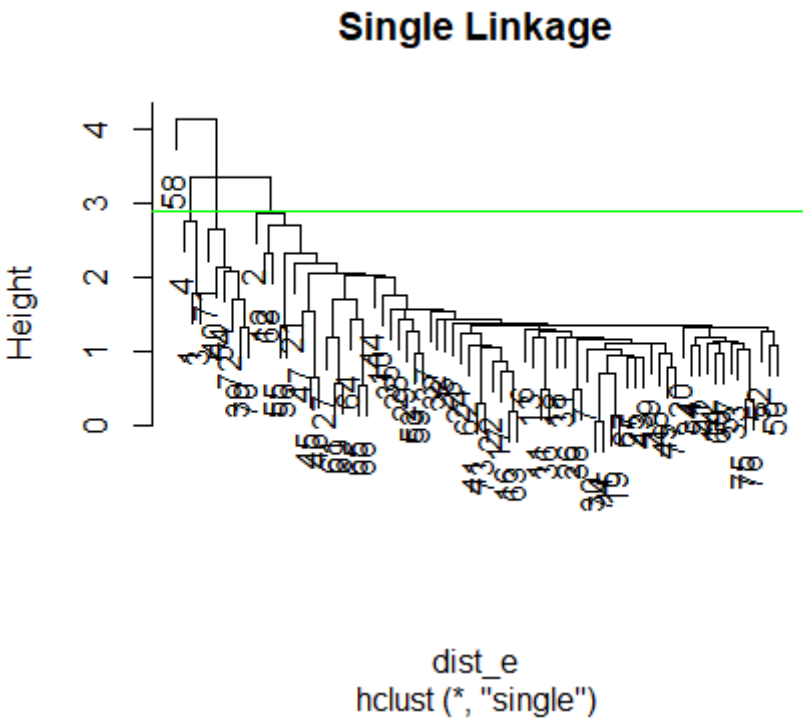
**legMedio=hclust(dist_e,method = "average")**
**summary(legMedio)**
**legMedio$merge**
**plot(legMedio,main = "Average Linkage")**
**Clustertwo=cutree(legMedio,5)**
**abline(h=5,col="green")**



***-Metodo del legame singolo***

Il metodo del legame singolo (single linkage) si basa su un criterio di distanza minima tra gli elementi che compongono i cluster. Questo algoritmo di aggregazione predilige elementi più simili tra loro e li accoppia.

**legSingolo=hclust(dist_e,method = "single")**
**summary(legSingolo)**
**legSingolo$merge**
**plot(legSingolo,main = "Single Linkage")**
**Clusterthree=cutree(legSingolo,2.9)**
**abline(h=2.9,col="green")**

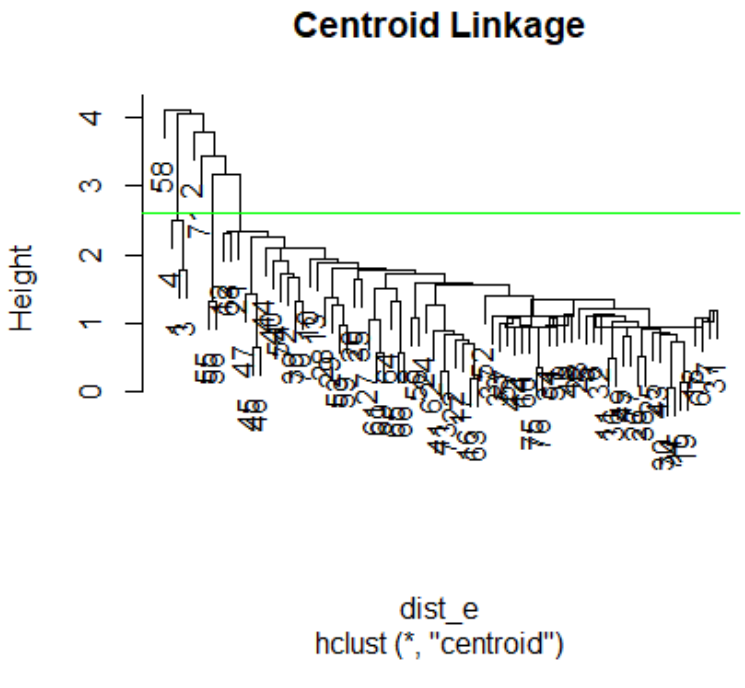



***-Metodo del legame centroide***

Il metodo del legame centroide (centroid linkage) si basa su un criterio in cui la distanza è data dai centroidi (baricentri) tra due diversi cluster.

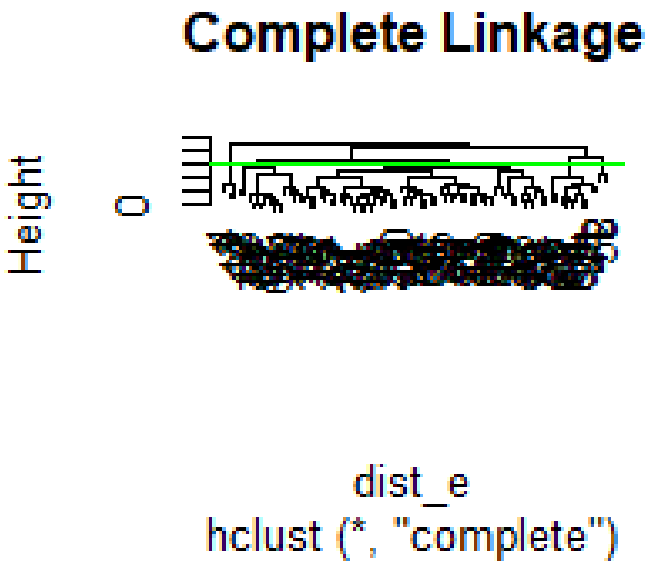
Per centroide s'intende il punto medio della nuvola dei punti che forma un determinato gruppo.

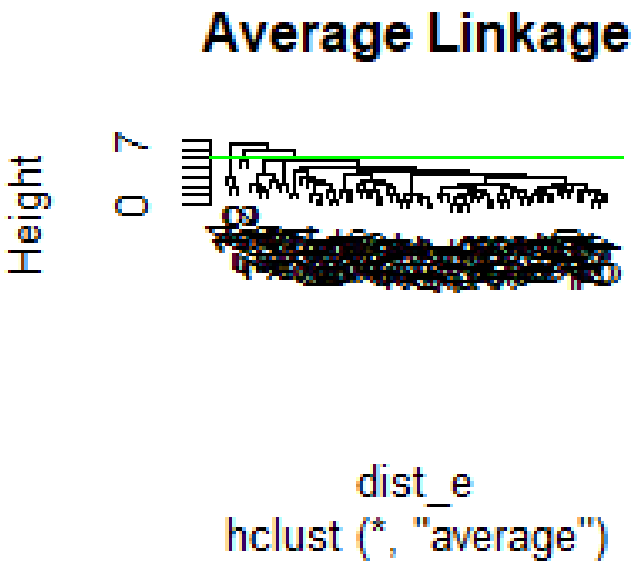
**legCentroid=hclust(dist_e, method = "centroid")**
**summary(legCentroid)**
**legCentroid$merge**
**plot(legCentroid,main="Centroid Linkage")**
**Clusterfour=cutree(legCentroid,2.6)**
**abline(h=2.6,col="green")**

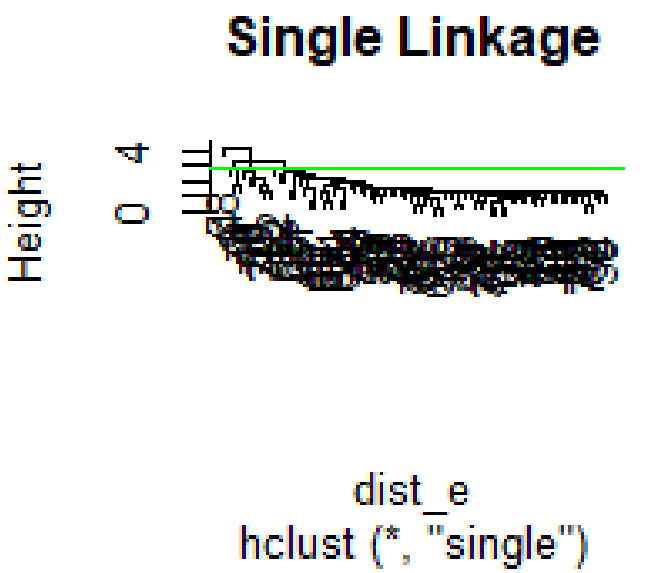


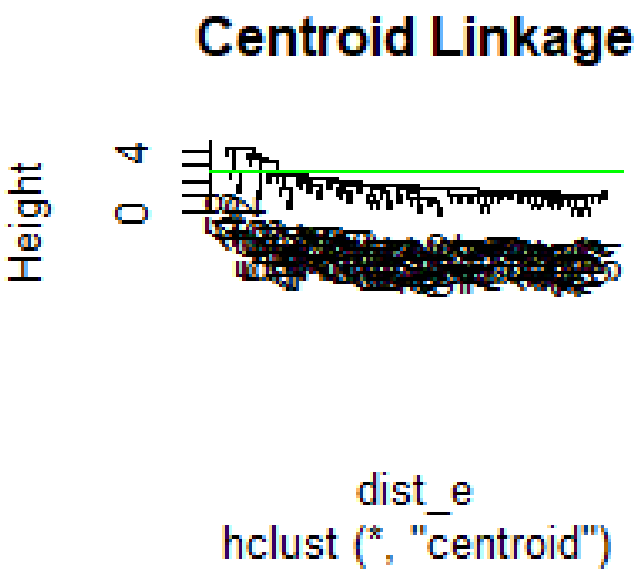
A questo punto è utile anche visualizzare tutti i plot insieme per notare le differenze e per fare ciò si divide l'output di R in 2 righe e 2 colonne affinché si possano inserire tutti i plot.

```
par(mfrow=c(2,2))
plot(legCompleto,main = "Complete Linkage")
Clusterone=cutree(legCompleto,6)
abline(h=6,col="green")
plot(legMedio,main = "Average Linkage")
Clustertwo=cutree(legMedio,5)
abline(h=5,col="green")
plot(legSingolo,main = "Single Linkage")
Clusterthree=cutree(legSingolo,2.9)
abline(h=2.9,col="green")
plot(legCentroid,main="Centroid Linkage")
Clusterfour=cutree(legCentroid,2.6)
abline(h=2.6,col="green")
```




**Chiave di Lettura del Dendrogramma**

Il dendrogramma non è altro che un diagramma ad albero in cui vengono rappresentati tutti gli step per formare le coppie. Questo grafico presenta nella parte bassa (o radici) tutti gli elementi distinti e nella parte alta vi è l'ultimo gruppo che si viene a creare attraverso l'accoppiamento, per questo il dendrogramma si legge dalle radici fino alla cima e non viceversa.

La lunghezza dei rami è proporzionale al livello di distanza tra un elemento e un altro e sulla sinistra dello schema è indicata l'altezza del dendrogramma.

La linea verde che invece appare su ogni grafico sta a indicare il livello (o la distanza) in cui si vuole tagliare il dendrogramma per dare significatività al test ma anche per visualizzare più velocemente i cluster creati.

Tagliare quindi il dendrogramma ottenuto attraverso il legame completo ad una distanza pari a 6, permette di dire che i cluster creati sono concretamente 5; differentemente, tagliando lo stesso albero ad altezza 4 i cluster creati sono ben 13.
A seconda che si voglia una maggiore o minore distanza, il taglio sarà più o meno profondo e i cluster saranno diversi.
Naturalmente sarebbe stato possibile tagliare ancora più in profondità per ottenere distanze minori ma a quel punto sarebbe stato più difficile contare i cluster che emergono dal taglio.
La domanda che ora sorge spontanea è: a quale livello tagliare il dendrogramma?
In generale, dato l'interesse ad avere il minor numero di gruppi con massima omogeneità, si cerca di tagliare alle radici (cioè in basso) dell'insieme con i rami più lunghi (cioè le verticali più lunghe), per questo abbiamo tagliato i grafici ad altezze diverse.
Visionando tutti i grafici costruiti possiamo notare le differenze che si hanno tra i differenti concetti di distanza.
Una differenza evidente si può notare tra il legame completo ed il legame singolo, in cui il primo algoritmo, utilizzando una distanza maggiore tra gli elementi dei cluster, predilige elementi dissimili tra di loro; invece il secondo algoritmo, utilizzando una distanza minore tra gli elementi dei cluster, predilige elementi simili tra di loro.
Nonostante la differenza tra i metodi però, le coppie di elementi formate sono spesso simili tra loro in tutti gli schemi presentati, come si evince dal *merge*.

## 2) *l'Analisi delle Componenti Principali (PCA)*

L'Analisi delle Componenti Principali o PCA è una tecnica usata per la semplificazione dei dati ed è utilizzata nell'ambito della statistica multivariata.
Lo scopo di questa tecnica è quello di ridurre il numero di variabili, che può essere più o meno elevato, a un numero minore di variabili latenti, limitando il più possibile la perdita di informazioni.
Lo studio della PCA avviene tramite una trasformazione lineare delle variabili che proietta quelle originarie in un nuovo sistema cartesiano in cui la nuova variabile con la maggiore varianza viene proiettata sul primo asse, la variabile nuova, seconda per dimensione della varianza, sul secondo asse e così via.
Prima di procedere con l'analisi dei componenti principali è utile ricordare che il dataset utilizzato è lo stesso di quello della Cluster Analysis.
Come prima cosa vengono create della variabili specifiche che conterranno le righe e le colonne.
**righe=nrow(C)**
**colonne=ncol(C)**
Successivamente, si crea, attraverso il metodo *cor()*, la matrice di correlazione
**MCorrelazione=cor(C)** e si visualizza **MCorrelazione**

```
> MCorrelazione
            calories      protein          fat       sodium       fiber      carbo      sugars
calories  1.00000000  0.019066068  0.498609814  0.300649227 -0.29341275  0.2506809  0.56234029
protein   0.01906607  1.000000000  0.208430990 -0.054674348  0.50033004 -0.1308636 -0.32914178
fat       0.49860981  0.208430990  1.000000000 -0.005407464  0.01671924 -0.3180435  0.27081918
sodium    0.30064923 -0.054674348 -0.005407464  1.000000000 -0.07067501  0.3559835  0.10145138
fiber    -0.29341275  0.500330043  0.016719237 -0.070675009  1.00000000 -0.3560827 -0.14120539
carbo     0.25068091 -0.130863648 -0.318043492  0.355983473 -0.35608274  1.0000000 -0.33166538
sugars    0.56234029 -0.329141777  0.270819175  0.101451381 -0.14120539 -0.3316654  1.00000000
potass   -0.06660886  0.549407400  0.193278602 -0.032603467  0.90337367 -0.3496852  0.02169581
vitamins  0.26535630  0.007335371 -0.031156266  0.361476688 -0.03224268  0.2581475  0.12513726
              potass     vitamins
calories -0.06660886  0.265356298
protein   0.54940740  0.007335371
fat       0.19327860 -0.031156266
sodium   -0.03260347  0.361476688
fiber     0.90337367 -0.032242679
carbo    -0.34968522  0.258147549
sugars    0.02169581  0.125137260
potass    1.00000000  0.020698687
vitamins  0.02069869  1.000000000
```

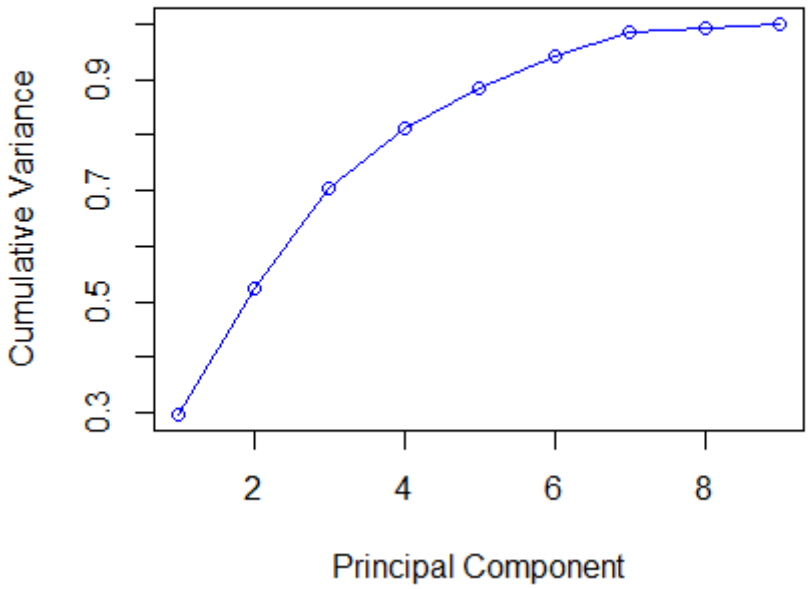
Essa è utile perché permette di trovare i suoi autovalori attraverso la funzione *eigen()*.
Per comodità verranno create due variabili apposite che conterranno gli autovalori e gli autovettori
**av=eigen(MCorrelazione)**
**autoValori=av$values**
**autoVettori=av$vectors**

```
> autoValori
[1] 2.67188758 2.04141952 1.61212066 0.99433925 0.64371166 0.52968361 0.37659685 0.07404297 0.05619791
> autoVettori
             [,1]        [,2]        [,3]       [,4]        [,5]       [,6]        [,7]         [,8]
 [1,] -0.24557838 -0.56316367 -0.06553063  0.2562895 -0.018384088  0.41655733  0.05117600  0.525817680
 [2,]  0.40215052 -0.10910761 -0.29986416  0.4668255  0.149100444  0.02468568  0.65868233 -0.156362253
 [3,]  0.08921231 -0.50016587  0.20414942  0.4654339  0.008067623 -0.43087077 -0.46670767 -0.174956270
 [4,] -0.20626796 -0.21979378 -0.47840417 -0.1959903 -0.635518295 -0.45177770  0.18138122  0.003570421
 [5,]  0.54231498 -0.05801277 -0.20037987 -0.2531918 -0.126148899  0.14587079 -0.25860915  0.494614884
 [6,] -0.34859841  0.15068669 -0.49073812  0.2565741 -0.091117804  0.46211011 -0.33738229 -0.330031975
 [7,] -0.14112655 -0.49352318  0.32300117 -0.4392656 -0.044013360  0.28945271  0.26042510 -0.358231624
 [8,]  0.51850187 -0.22326270 -0.19161468 -0.1768799 -0.105325198  0.23685991 -0.24188684 -0.431784182
 [9,] -0.15190006 -0.23018187 -0.46175507 -0.3273707  0.732281313 -0.24780710 -0.07594361  0.010569041
              [,9]
 [1,] -0.315591749
 [2,]  0.193323692
 [3,]  0.222326504
 [4,] -0.031032033
 [5,]  0.499556212
 [6,]  0.323299178
 [7,]  0.396690030
 [8,] -0.548784168
 [9,] -0.005470505
```

Adesso, si può procedere con lo studio delle varianze.
Per calcolare la varianza spiegata, si dividono gli autovalori per il numero di colonne (ovvero le variabili) e di conseguenza, si calcola la varianza cumulata della varianza spiegata, ottenuta attraverso il metodo *cumsun()*
**varSpiegata=autoValori/colonne**
**varSpiegata**
**varSpiegataCumulata=cumsum(varSpiegata)**
**varSpiegataCumulata**

```
> varSpiegata
[1] 0.296876398 0.226824391 0.179124517 0.110482139 0.071523517 0.058853734 0.041844094 0.008226997
[9] 0.006244212
> varSpiegataCumulata
[1] 0.2968764 0.5237008 0.7028253 0.8133074 0.8848310 0.9436847 0.9855288 0.9937558 1.0000000
```

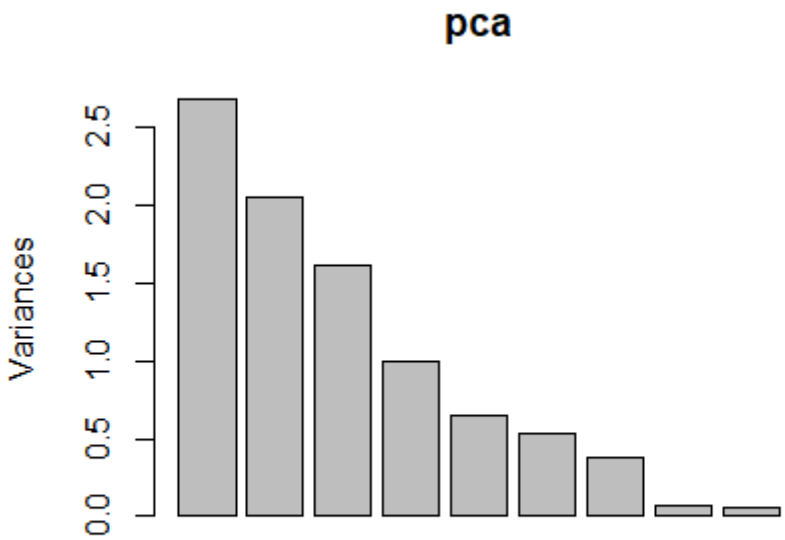
Volendo si può creare un grafico, ad esempio quello della varianza cumulata.
**plot(varSpiegataCumulata,type="o", ylab="Cumulative Variance", xlab="Principal Component", col="blue")**



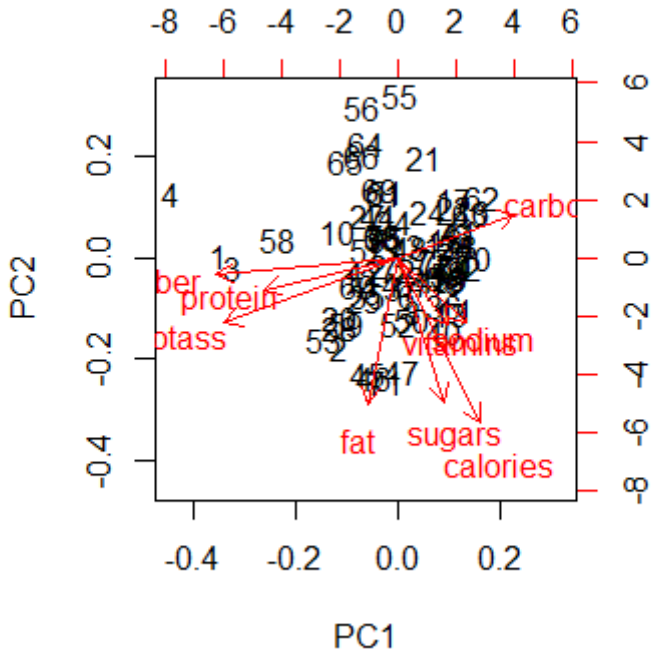

In verità, tutto questo studio può essere facilmente riassunto in un solo passaggio, utilizzando la funzione *prcomp()* che è la funzione ideale per una PCA. Essa restituisce, oltre ai valori appena calcolati, la deviazione standard, la matrice di rotazione e soprattutto permette di costruire vari tipi di grafici.
**pca=prcomp(C,scale. = TRUE)**
Possiamo quindi, attraverso il *summary()*, notare proprio come lo studio delle varianze è identico a quello fatto in precedenza.
**summary(pca)**

```
> summary(pca)
Importance of components:
                          PC1    PC2    PC3    PC4     PC5     PC6     PC7     PC8     PC9
Standard deviation     1.6346 1.4288 1.2697 0.9972 0.80232 0.72779 0.61367 0.27211 0.23706
Proportion of Variance 0.2969 0.2268 0.1791 0.1105 0.07152 0.05885 0.04184 0.00823 0.00624
Cumulative Proportion  0.2969 0.5237 0.7028 0.8133 0.88483 0.94368 0.98553 0.99376 1.00000
```

Perciò, una volta ottenute tutte queste informazioni, si procede con la creazione dei grafici, ad esempio un classico istogramma
**plot(pca)**




Oppure un biplot, che è lo scopo di questo studio.
**biplot(pca)**




Il biplot è una rappresentazione grafica dell'informazione contenuta in una matrice X di dimensione $n \times p$.
L'idea alla base del biplot consiste nell'aggiungere l'informazione sulla relazione tra variabili al grafico delle componenti principali.
Il suffisso *bi* indica le due informazioni contenute in X e rappresentate nel grafico. Le righe di X rappresentano le osservazioni campionarie e le colonne rappresentano le variabili.
Il biplot mostra quindi quali sono le variabili tra loro più vicine e come vengono raggruppate nei quadranti.
Osservando questo biplot si può concludere lo studio della PCA e, soprattutto, si può notare quale tra le variabili è posta nel primo quadrante e quindi qual è quella con varianza maggiore delle altre (carbo) e successivamente, in senso orario, tutte le altre che avranno varianza maggiore della successiva ma minore della precedente.

**CONCLUSIONE**

Questo lavoro ha voluto mostrare come approcciarsi ad un dataset, attraverso un'analisi esplorativa dello stesso e conseguentemente, illustrare come scegliere e analizzare le più rilevanti tra un'ampia gamma di variabili, sulla base dell'utilità delle variabili stesse ai fini dello studio.
Le due analisi effettuate, cluster analysis e PCA, hanno avuto lo scopo, rispettivamente, di spiegare i vari algoritmi da utilizzare nel software Rstudio e la conseguente formazione dei cluster e di classificare le variabili scelte con l'obiettivo di comprenderne la rilevanza ai fini dell'analisi.

***Sitografia***

Il seguente dataset è stato preso dalla seguente pagina: https://www.kaggle.com/crawford/80-cereals?select= cereal.csv