Università degli Studi di Salerno

Dipartimento di Informatica

Progetto di Statistica e Analisi dei Dati

Indagine statistica sulla raccolta differenziata nelle regioni italiane

Docente

Amelia Giuseppina Nobile

Studenti

Francesco Abate 0522500993

Vincenzo De Martino 0522500966

Sommario

Capitolo 1

Introduzione alla statistica

L'etimologia della parola "Statistica" fa riferimento alla constatazione per cui le prime informazioni su determinati fenomeni venivano raccolte da organismi statali, i quali ne erano i principali utilizzatori. Col tempo, il campo d'uso della statistica è andato crescendo quando si è capito come utilizzare un insieme di informazioni allo scopo di ricercare le cause del manifestarsi di determinati fenomeni.
Oggigiorno, la statistica è utilizzata in tutti i campi di studio e se ne fa sempre un utilizzo maggiore; gli organismi che istituzionalmente raccolgono e diffondono informazioni sono innumerevoli: tra questi, troviamo l'ente italiano ISTAT.

La statistica può essere, quindi, definita come un insieme di metodi per lo studio di determinati fenomeni, al fine di determinare quali elementi possono essere considerati cause principali e quali sono le relazioni tra loro. In generale, per fenomeno si indica tutto ciò che capita intorno a noi, quindi qualsiasi cosa percepibile e constatabile.
Esistono fenomeni che si presentano sempre con le stesse caratteristiche, detti "tipici"; i fenomeni che si presentano sempre con caratteristiche differenti sono detti "atipici".

Con modalità si definiscono gli attributi che un carattere può assumere, quindi numerici e non: in caso di attributi numerici si parla di modalità quantitative; in caso di attributi non numerici si parla di modalità qualitative. Infine, con frequenza si indica il numero che esprime quante volte una data modalità si presenta nella totalità delle unità rilevate.

Il progetto

Per l'esame universitario Statistica e Analisi dei Dati è necessario effettuare un'analisi statistica studiando i dati nel linguaggio di programmazione R.

La tematica che abbiamo deciso di trattare riguarda la produzione di rifiuti urbani, la quale in Italia è sempre cresciuta al passo dei bisogni del comune cittadino. La banca di dati è fornita dall'ente italiano ISTAT, la quale tratta il numero di tonnellate di rifiuti per ogni regione, aggiornata all'anno 2017.

La banca di dati mostra per ogni regione il numero di tonnellate di rifiuti suddivisi per raccolta indifferenziata e raccolta differenziata, la quale è suddivisa a sua volta in rifiuti organici, carta e cartone, vetro, plastica e altro. In quest'ultima categoria rientrano rifiuti speciali, metallo, legno, tessili, apparecchiature elettriche ed elettroniche, rifiuti ingombranti soggetti a recupero, rifiuti da costruzione e demolizione.

Nella prossima pagina segue uno screenshot della banca dati su Excel.

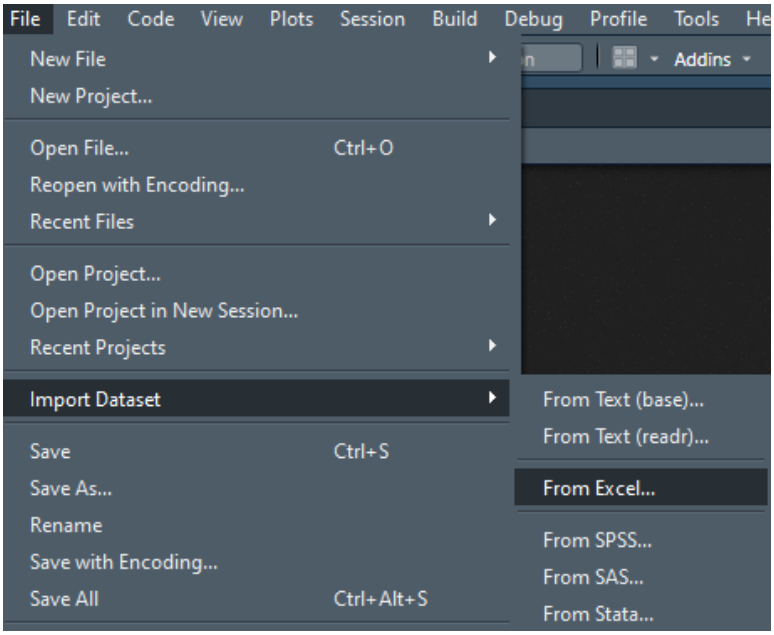
Infine, per semplificare la realizzazione del progetto ci siamo avvalsi dell'uso di RStudio, apposito IDE per R.

Regioni	Raccolta indifferenziata	Rifiuti organici	Carta e cartone	Vetro	Plastica	Altro
Piemonte	840.807	409.527	265.959	160.678	125.039	261.572
Valle d'Aosta /Vallée d'Aoste	28.649	14.581	9.387	6.685	6.160	8.259
Liguria	424.884	128.257	84.428	60.408	33.569	98.489
Lombardia	1.423.822	1.206.023	546.999	422.744	248.268	837.633
Trentino-Alto Adige/Südtirol	147.533	133.535	83.542	43.739	33.093	77.592
Veneto	615.317	764.526	286.931	222.674	117.036	328.309
Friuli-Venezia Giulia	203.354	166.913	68.954	45.153	26.348	78.296
Emilia-Romagna	1.034.391	708.244	385.188	160.642	137.038	434.262
Toscana	1.034.846	494.222	283.163	116.695	85.732	229.162
Umbria	172.706	116.919	57.202	27.106	22.719	54.178
Marche	300.266	232.084	111.478	61.059	24.788	87.310
Lazio	1.607.961	532.659	346.594	212.491	73.530	188.632
Abruzzo	262.624	149.314	77.083	49.747	15.088	42.889
Molise	80.819	14.953	6.802	6.672	4.157	3.253
Campania	1.209.747	678.908	180.335	136.738	137.860	217.411
Puglia	1.117.600	291.501	177.168	82.467	75.584	132.016
Basilicata	107.409	31.234	23.203	12.332	7.447	14.690
Calabria	466.847	126.580	76.408	45.004	16.489	42.462
Sicilia	1.800.509	208.309	123.274	56.831	38.333	72.939
Sardegna	267.312	213.663	83.012	70.519	45.376	43.590

Banca dati in R

Per poter analizzare i dati, è necessario importare la banca dati in R.
L'apposito IDE RStudio offre un'opzione che semplifica l'import della banca dati, al fine di evitare futili e complicate linee di codice.

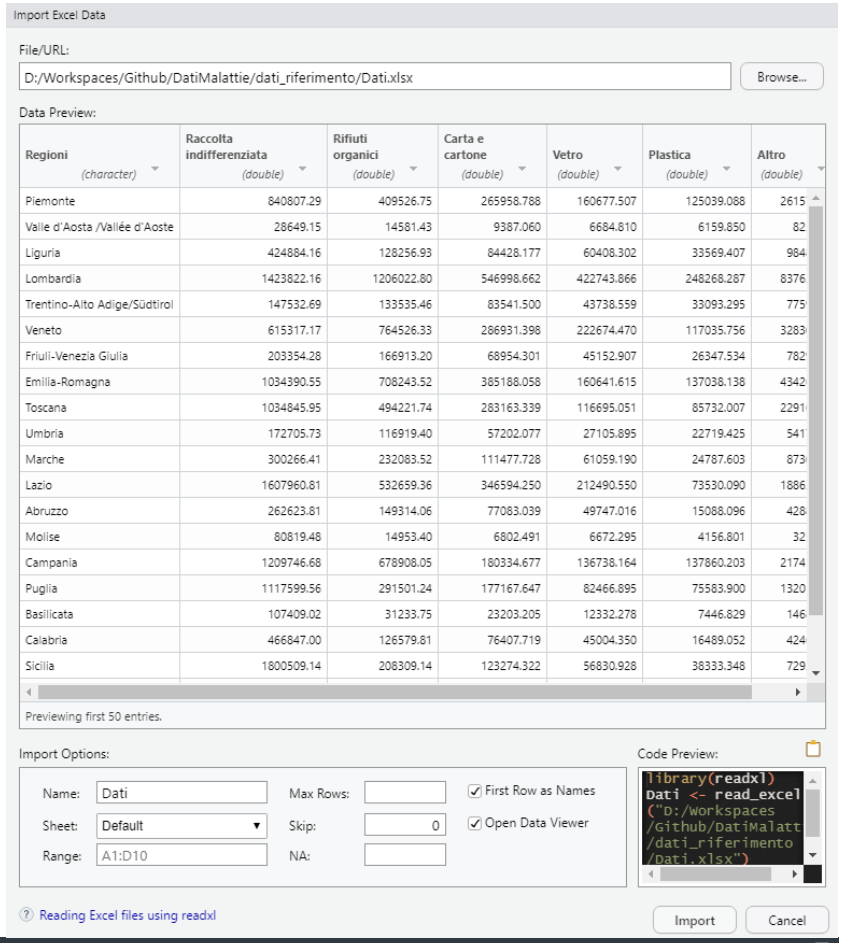
Tramite l'opzione
File -> Import Dataset -> From Excel
verrà caricato il file Excel in cui è presente la base di dati da importare.
Nel file Excel sono state incluse, per comodità, anche le regioni e le categorie di spazzatura, al fine di evitare l'import di molteplici tabelle Excel o di codificare manualmente tali record in appositi array.



Una volta selezionata l'apposita opzione, sarà necessario impostare l'URL della base di dati. Fatto ciò, verrà visualizzata un'anteprima della base di dati che si sta importando in modo da procedere con l'operazione di import in totale sicurezza.

Confermato l'import, i dati verranno caricati nel Global Environment di R, quindi verranno conservati in RAM.

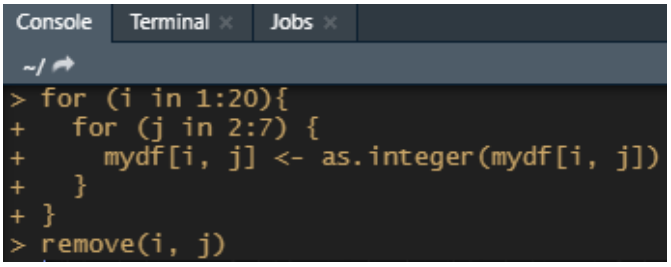
Non ci rimane altro che conservare i dati in una variabile in modo da trattarli, quindi basta assegnare il nome del file Excel.



Regioni (character)	Raccolta indifferenziata (double)	Rifiuti organici (double)	Carta e cartone (double)	Vetro (double)	Plastica (double)	Altro (double)
Piemonte	840807.29	409526.75	265958.788	160677.507	125039.088	2615
Valle d'Aosta /Vallée d'Aoste	28649.15	14581.43	9387.060	6684.810	6159.850	82
Liguria	424884.16	128256.93	84428.177	60408.302	33569.407	984
Lombardia	1423822.16	1206022.80	546998.662	422743.866	248268.287	8376
Trentino-Alto Adige/Südtirol	147532.69	133535.46	83541.500	43738.559	33093.295	775
Veneto	615317.17	764526.33	286931.398	222674.470	117035.756	3283
Friuli-Venezia Giulia	203354.28	166913.20	68954.301	45152.907	26347.534	782
Emilia-Romagna	1034390.55	708243.52	385188.058	160641.615	137038.138	4342
Toscana	1034845.95	494221.74	283163.339	116695.051	85732.007	2291
Umbria	172705.73	116919.40	57202.077	27105.895	22719.425	541
Marche	300266.41	232083.52	111477.728	61059.190	24787.603	873
Lazio	1607960.81	532659.36	346594.250	212490.550	73530.090	1886
Abruzzo	262623.81	149314.06	77083.039	49747.016	15088.096	428
Molise	80819.48	14953.40	6802.491	6672.295	4156.801	32
Campania	1209746.68	678908.05	180334.677	136738.164	137860.203	2174
Puglia	1117599.56	291501.24	177167.647	82466.895	75583.900	1320
Basilicata	107409.02	31233.75	23203.205	12332.278	7446.829	146
Calabria	466847.00	126579.81	76407.719	45004.350	16489.052	424
Sicilia	1800509.14	208309.14	123274.322	56830.928	38333.348	729

```
> mydf <- Dati
> mydf
# A tibble: 20 x 7
   Regioni                `Raccolta indifferenziata` `Rifiuti organici` `Carta e cartone`  Vetro Plastica  Altro
   <chr>                                      <dbl>               <dbl>            <dbl>  <dbl>    <dbl>  <dbl>
 1 Piemonte                                840807.             409527.          265959. 160678. 125039. 261572.
 2 Valle d'Aosta /Vallée d'Aoste            28649.              14581.           9387.   6685.   6160.   8259.
 3 Liguria                                 424884.             128257.          84428.  60408.  33569.  98489.
 4 Lombardia                              1423822.            1206023.          546999. 422744. 248268. 837633.
 5 Trentino-Alto Adige/Südtirol            147533.             133535.          83542.  43739.  33093.  77592.
 6 Veneto                                  615317.             764526.          286931. 222674. 117036. 328309.
 7 Friuli-Venezia Giulia                   203354.             166913.          68954.  45153.  26348.  78296.
 8 Emilia-Romagna                         1034391.            708244.          385188. 160642. 137038. 434262.
 9 Toscana                                1034846.            494222.          283163. 116695. 85732.  229162.
10 Umbria                                 172706.             116919.          57202.  27106.  22719.  54178.
11 Marche                                 300266.             232084.          111478. 61059.  24788.  87310.
12 Lazio                                 1607961.             532659.          346594. 212491. 73530. 188632.
13 Abruzzo                                262624.             149314.          77083.  49747.  15088.  42889.
14 Molise                                 80819.              14953.           6802.   6672.   4157.   3253.
15 Campania                              1209747.             678908.          180335. 136738. 137860. 217411.
16 Puglia                                1117600.             291501.          177168. 82467.  75584.  132016.
17 Basilicata                             107409.             31234.           23203.  12332.  7447.   14690.
18 Calabria                               466847.             126580.          76408.  45004.  16489.  42462.
19 Sicilia                               1800509.             208309.          123274. 56831.  38333.  72939.
20 Sardegna                               267312.             213663.          83012.  70519.  45376.  43590.
```

C'è solo un piccolo problema: nell'anteprima dell'import si nota che sulle colonne viene indicato il tipo double, quindi i dati presenti nelle colonne (eccetto i label) verranno trattati da tali. Per semplificare l'operato, tratteremo i dati come interi e lasciando gestire l'arrotondamento per difetto o eccesso a R stesso. Bisogna, quindi, convertire ogni valore di ogni colonna, tranne la prima siccome contiene le regioni, in intero.



Abbiamo, finalmente, un dataframe in R composto da soli numeri interi.

```
Console  Terminal  Jobs
~/
> mydf
# A tibble: 20 x 7
   Regioni                    `Raccolta indifferenziata` `Rifiuti organici` `Carta e cartone`  Vetro Plastica  Altro
   <chr>                                          <dbl>              <dbl>             <dbl>  <dbl>    <dbl>  <dbl>
 1 Piemonte                                      840807             409526            265958 160677   125039 261571
 2 Valle d'Aosta /Vallée d'Aoste                  28649              14581              9387   6684     6159   8259
 3 Liguria                                       424884             128256             84428  60408    33569  98488
 4 Lombardia                                    1423822            1206022            546998 422743   248268 837632
 5 Trentino-Alto Adige/Südtirol                  147532             133535             83541  43738    33093  77592
 6 Veneto                                        615317             764526            286931 222674   117035 328308
 7 Friuli-Venezia Giulia                         203354             166913             68954  45152    26347  78295
 8 Emilia-Romagna                               1034390             708243            385188 160641   137038 434261
 9 Toscana                                      1034845             494221            283163 116695    85732 229162
10 Umbria                                        172705             116919             57202  27105    22719  54177
11 Marche                                        300266             232083            111477  61059    24787  87309
12 Lazio                                        1607960             532659            346594 212490    73530 188631
13 Abruzzo                                       262623             149314             77083  49747    15088  42888
14 Molise                                         80819              14953              6802   6672     4156   3253
15 Campania                                     1209746             678908            180334 136738   137860 217410
16 Puglia                                       1117599             291501            177167  82466    75583 132015
17 Basilicata                                    107409              31233             23203  12332     7446  14689
18 Calabria                                      466847             126579             76407  45004    16489  42461
19 Sicilia                                      1800509             208309            123274  56830    38333  72939
20 Sardegna                                      267312             213663             83011  70519    45376  43589
```

Siccome gli studi verranno svolti su colonne, per semplificazione le colonne saranno contenute in semplici array.

```
# Getting the row arrays
datiRaccoltaIndifferenziata = mydf$`Raccolta indifferenziata`
datiCarta = mydf$`Carta e cartone`
datiPlastica = mydf$Plastica
datiVetro = mydf$Vetro
datiUmido = mydf$`Rifiuti organici`
datiAltro = mydf$Altro
```

Infine, riguardo la creazione e visualizzazione dei grafici, verranno utilizzate delle funzioni proxy per semplificare la loro creazione e visualizzazione, al fine di favorire il riutilizzo del codice. Segue, come esempio, la funzione proxy che permette la creazione e visualizzazione di un diagramma di Pareto, dipendente dall'array di valori passato come input alla funzione.

```
# createPareto : displays a Pareto diagram with frequencies
##  input -> name : gives a name to the Pareto diagram
##           arrayToAnalyze : array with indexes to display
createPareto <- function (name, arrayToAnalyze) {

  # Calculating frequencies
  mySum = sum(arrayToAnalyze)
  arrayToAnalyze <- arrayToAnalyze / mySum

  # Ordering
  ariOrdered <- order(arrayToAnalyze, decreasing = TRUE)

  # Creating graphic
  bp <- barplot(arrayToAnalyze[ariOrdered], main = name,
                col=rainbow(length(arrayToAnalyze)),
                names = mydf$Regioni[ariOrdered], las=2, ylim = c(0, 1.05))

  lines(bp, cumsum(arrayToAnalyze[ariOrdered]),
        type = "b", pch = 16)

  text(bp - 0.2, cumsum(arrayToAnalyze[ariOrdered]) + 0.03,
       paste(format(cumsum(arrayToAnalyze[ariOrdered]) * 100,
                    digits = 2), "%"))
}
```

Tali funzioni proxy verranno utilizzate per grafici generici; in caso di particolari esigenze, ovviamente si eviterà l'uso di tali codici.

Per il riuso del codice abbiamo provveduto alla creazione di due funzioni che restituiscono tutti i nomi delle regioni e dei rifiuti utilizzati nel nostro sistema, così dal ridurre la quantità

di codice e avere funzionalità già pronte:

```r
# namesGarbage : return names of garbage
namesGarbage<-function(){
  namesRifiuti <- c("Raccolta Indifferenziata","Rifiuti organici",
                    "Carta e cartone","Vetro","Plastica","Altro")
  return (namesRifiuti)
}

#namesRegion : rerurn names of Region
namesRegion<-function(){
  namesRegioni <- c("Piemonte","Valle d'Aosta /Vallée d'Aoste","Liguria",
                    "Lombardia","Trentino-Alto Adige/Südtirol","Veneto",
                    "Friuli-Venezia Giulia","Emilia-Romagna","Toscana",
                    "Umbria","Marche","Lazio","Abruzzo","Molise","Campania",
                    "Puglia","Basilicata","Calabria","Sicilia","Sardegna")
  return (namesRegioni)

}
```

Infine, nei prossimi capitoli verrà trattato un dataframe denominato come df: quest'ultimo sarà la base di dati comprendente solo ed esclusivamente i dati numerici, quindi non comprendendo le regioni nella prima colonna.

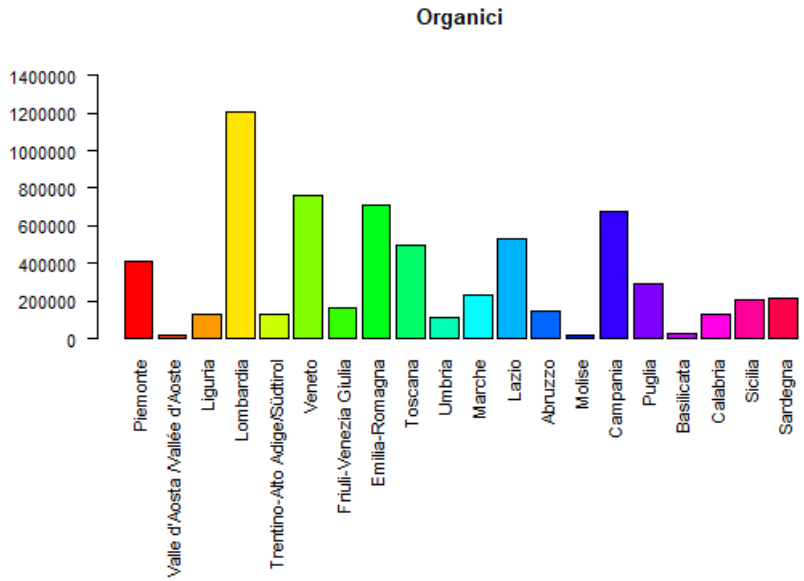
```r
for (i in 1:20){
  for (j in 2:7) {
    mydf[i, j] <- as.integer(mydf[i, j])
    df[i, j-1] <- mydf[i, j]
  }
}
remove(i, j)
```

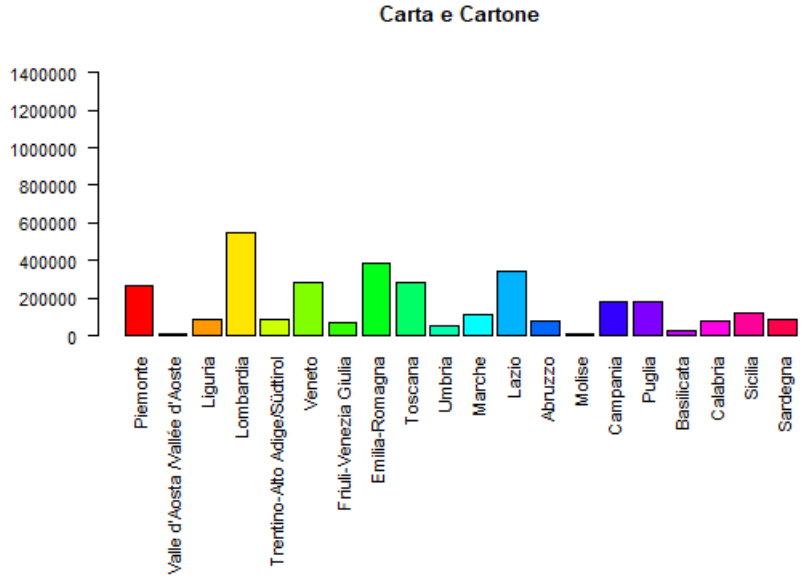
Per qualsiasi evenienza, la colonna delle regioni verrà salvata in un array denominato come namesRegioni.

Capitolo 2

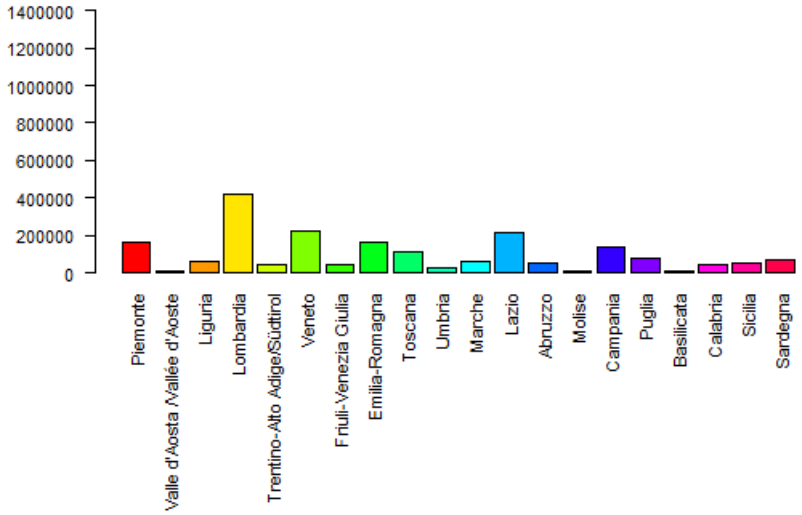
Prime considerazioni

Iniziamo con l'effettuare qualche piccola considerazione riguardo la raccolta differenziata, analizzando categoria per categoria quali regioni producono più rifiuti.

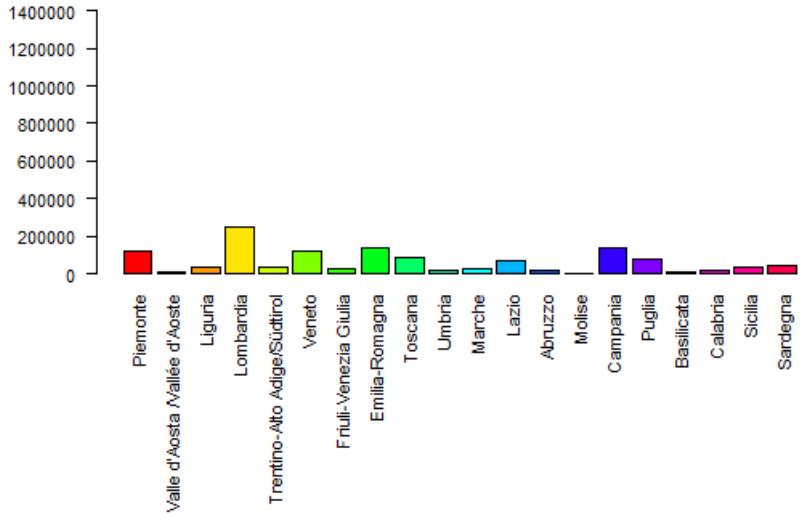




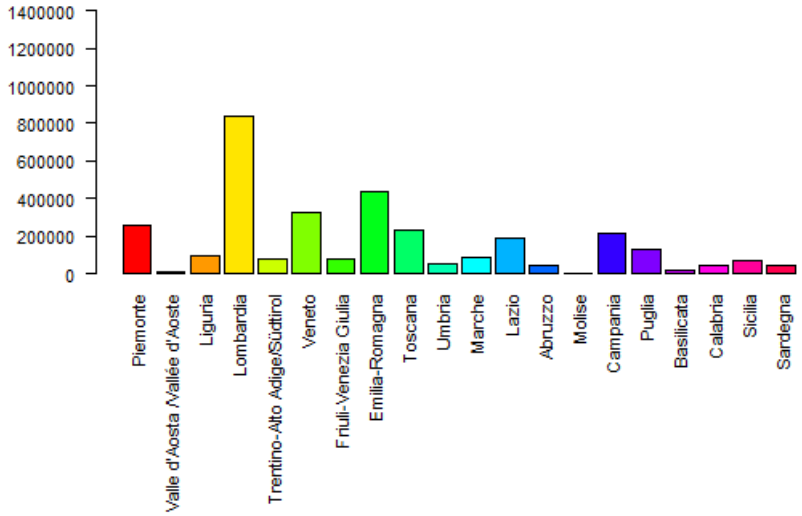
Vetro



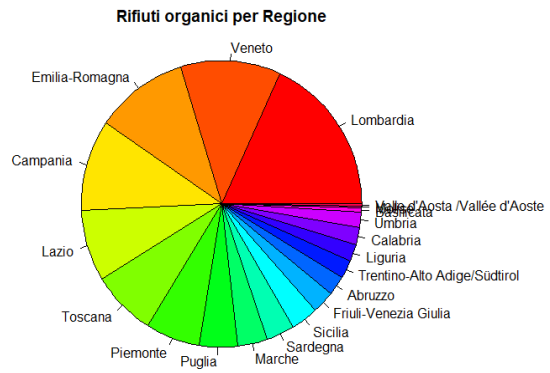
Plastica

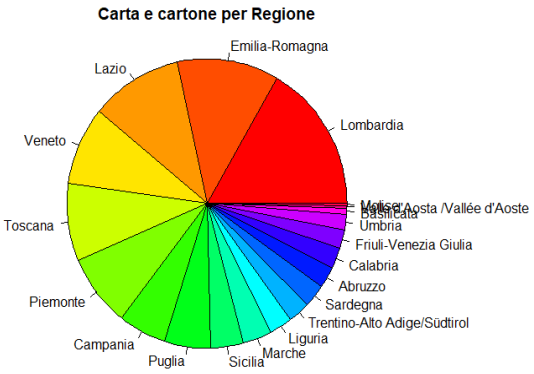


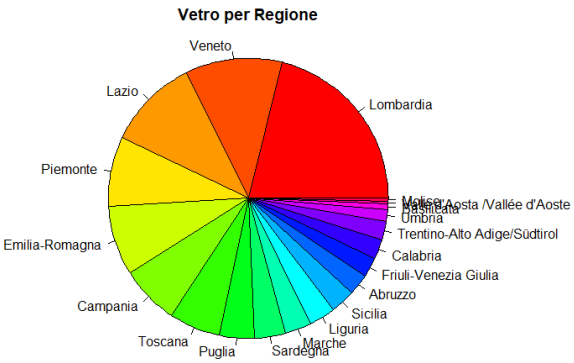
Altro

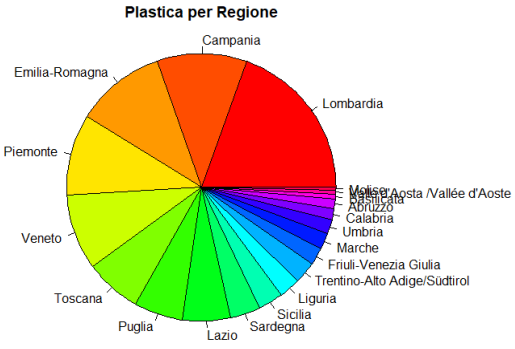


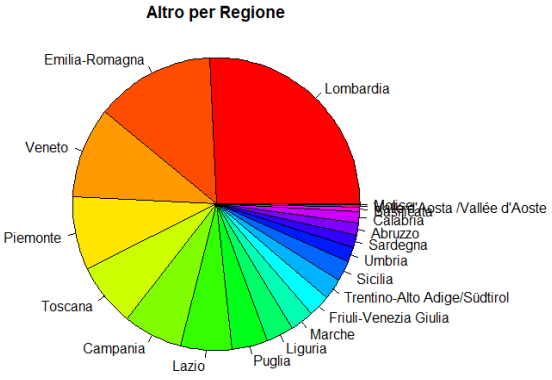
Dai grafici illustrati si evidenzia che la Lombardia è la regione producente più rifiuti in tutta Italia, di qualsiasi categoria; altre regioni con alti consumi sono il Veneto, l'Emilia-Romagna, il Lazio e la Campania. Diverse regioni consumano davvero poco rispetto alle regioni che consumano di più, come il Molise e la Valle d'Aosta: sarà sicuramente dovuto dal fatto che tali regioni hanno una popolazione decisamente inferiore rispetto alle altre, quindi i consumi sono decisamente minori. Infine, si nota che un po' tutte le regioni tendono a produrre più rifiuti organici rispetto alle altre categorie di rifiuti.
Osserviamo i consumi anche tramite diagrammi a torta, in modo da rilevare meglio la differenza riguardante i consumi tra regioni. Da ogni diagramma, difatti, si nota come la Lombardia sia sempre al primo posto in termini di consumi.











Seguono le funzioni con cui sono stati realizzati i diagrammi.

```
# Barplot section
createBarPlot_limit("Carta e Cartone", datiCarta, 1500000, 0)
createBarPlot_limit("Plastica", datiPlastica, 1500000, 0)
createBarPlot_limit("Vetro", datiVetro, 1500000, 0)
createBarPlot_limit("Organici", datiUmido, 1500000, 0)
createBarPlot_limit("Altro", datiAltro, 1500000, 0)
createBarPlot_limit("Rifiuti Indifferenziata",
                    datiRaccoltaIndifferenziata, 1500000, 0)

# Pie section
createPie("Rifiuti Indifferenziata",datiRaccoltaIndifferenziata)
createPie("Rifiuti Organici",datiUmido)
createPie("Carta e cartone",datiCarta)
createPie("Vetro",datiVetro)
createPie("Plastica",datiPlastica)
createPie("Altro",datiAltro)
```

È interessante, inoltre, osservare quanti rifiuti non soggetti a raccolta differenziata vengono prodotti dalle regioni. Prima di fare ciò, però, per ogni regione è necessario ricavare il numero di rifiuti soggetti a raccolta differenziata, sommando tutte le sottocategorie. Ciò è possibile tramite lo spezzone di codice mostrato a destra.

```
listDiff = rep(0, 20)
for (i in 1:20){
  mySum = 0
  for (j in 2:6) {
    mySum = mySum + df[i, j]
    df[i, j]
  }
  listDiff[i] = mySum
}
```

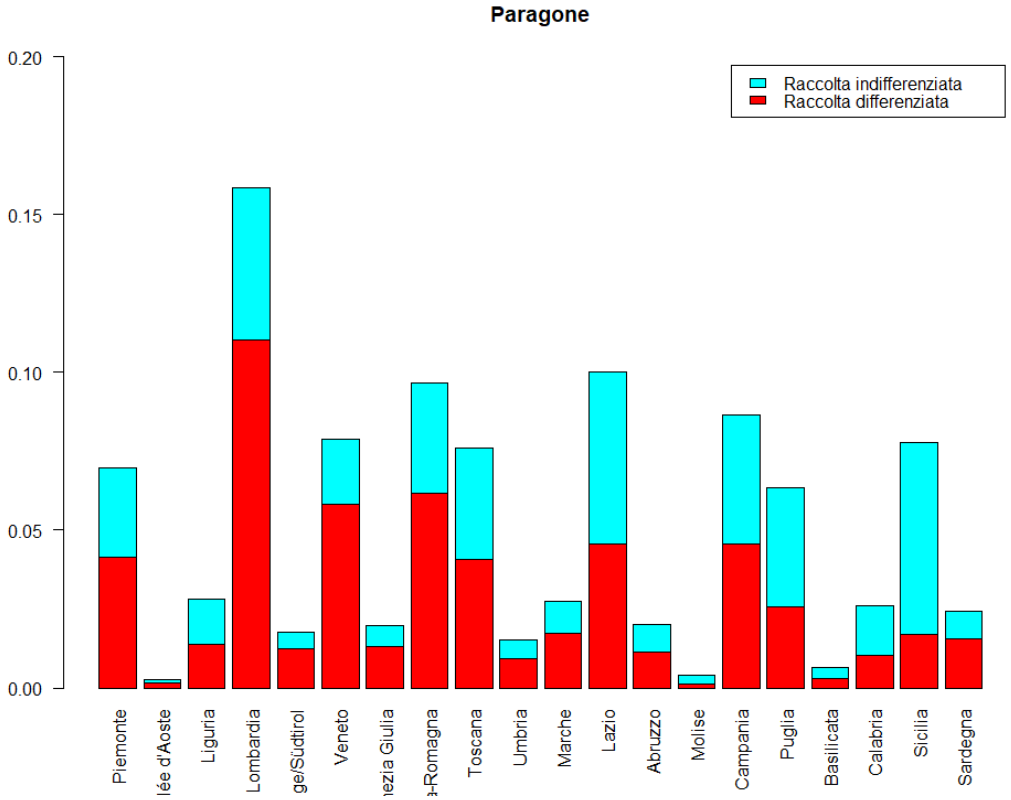
Infine, con le seguenti linee di codice è possibile creare un apposito data frame che mostri per ogni regione le tonnellate di spazzatura prodotta per raccolta differenziata e indifferenziata.

```
comparison = data.frame(listDiff, datiRaccoltaIndifferenziata)
rownames(comparison) <- namesRegion()
colnames(comparison) <- c("Raccolta differenziata",
                          "Raccolta indifferenziata")
comparison
```

Il risultato è il seguente.

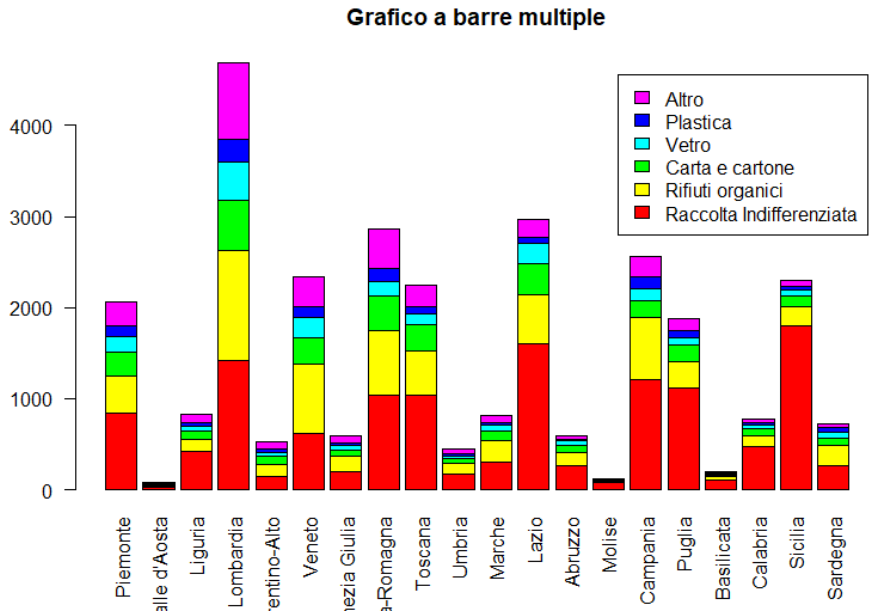
	Raccolta differenziata	Raccolta indifferenziata
Piemonte	1222771	840807
Valle d'Aosta /Vallée d'Aoste	45070	28649
Liguria	405149	424884
Lombardia	3261663	1423822
Trentino-Alto Adige/Südtirol	371499	147532
Veneto	1719474	615317
Friuli-Venezia Giulia	385661	203354
Emilia-Romagna	1825371	1034390
Toscana	1208973	1034845
Umbria	278122	172705
Marche	516715	300266
Lazio	1353904	1607960
Abruzzo	334120	262623
Molise	35836	80819
Campania	1351250	1209746
Puglia	758732	1117599
Basilicata	88903	107409
Calabria	306940	466847
Sicilia	499685	1800509
Sardegna	456158	267312

Interessante è anche il paragone tra le due tipologie di raccolta, siccome una visualizzazione in frequenze relative tramite grafico rende meglio l'idea della differenza per ogni regione.



Notiamo, purtroppo, che in moltissime regioni è ancora più che diffusa la raccolta indifferenziata, in particolare in Sicilia, nel Lazio, in Puglia, in Calabria e nel Molise: in queste regioni, la frequenza relativa di raccolta indifferenziata supera addirittura la metà della frequenza relativa della raccolta differenziata. Nel generale, in ogni caso, ogni regione ha comunque frequenze relative di raccolta indifferenziata abbastanza alte.

Di seguito troviamo un grafico rappresentante tutti i dati, relativi alla quantità di spazzatura e alla tipologia, per regione in un unico grafico a barre con barre sovrapposte per individuare e comprendere meglio i consumi per regione. I seguenti dati sono stati divisi per 1000.



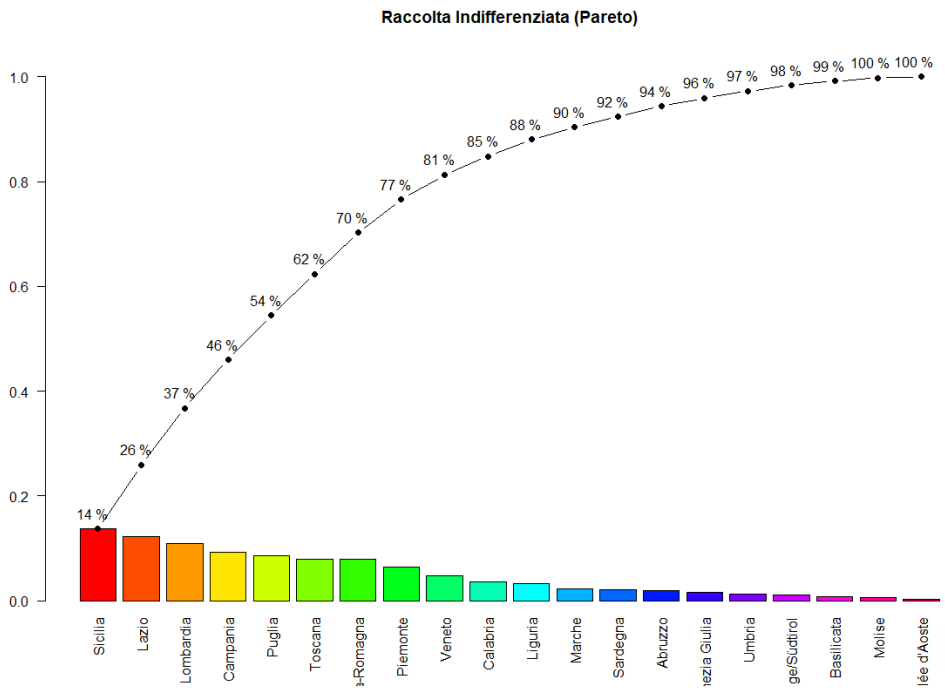
Capitolo 3

Eccessività della raccolta indifferenziata

In questa analisi prendiamo in oggetto la raccolta indifferenziata e vediamo quali regioni ne fanno un uso eccessivo. Ciò lo visioniamo tramite il diagramma di Pareto, il quale mostra la frequenza cumulata di uso di raccolta indifferenziata per ogni regione.
Prima di iniziare è necessario calcolare le singole frequenze per ogni regione, in modo da potersi ricavare le cumulate sommando le relative. Ciò che compone il diagramma di Pareto, infine, è un diagramma a barre che fa uso delle frequenze relative e una serie di punti collegati tra loro indicanti le frequenze cumulate.

```
createPareto <- function (name, arrayToAnalyze) {

  # Calculating frequencies
  mySum = sum(arrayToAnalyze)
  arrayToAnalyze <- arrayToAnalyze / mySum

  # Ordering
  ariOrdered <- order(arrayToAnalyze, decreasing = TRUE)

  # Creating graphic
  bp <- barplot(arrayToAnalyze[ariOrdered], main = name,
                col=rainbow(length(arrayToAnalyze)),
                names = mydf$Regioni[ariOrdered], las=2, ylim = c(0, 1.05))

  lines(bp, cumsum(arrayToAnalyze[ariOrdered]),
        type = "b", pch = 16)

  text(bp - 0.2, cumsum(arrayToAnalyze[ariOrdered]) + 0.03,
       paste(format(cumsum(arrayToAnalyze[ariOrdered]) * 100,
                    digits = 2), "%"))
}
```

Il diagramma di Pareto riguardante la raccolta indifferenziata è il seguente:



La maggior affluenza di dati in termini relativi è visibile in sette regioni di maggior rilievo, quali sono la Sicilia, il Lazio, la Lombardia, la Campania, la Puglia, la Toscana e l'Emilia-Romagna. Possiamo confermare ciò confrontando i valori della raccolta indifferenziata riguardanti tali regioni con la media campionaria. Va, quindi, calcolata la media campionaria: R rende possibile ciò tramite l'apposita funzione mean, la quale restituisce il valore di quest'ultima, pari a 657369.8 in tal caso.

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

```
mean(datiRaccoltaIndifferenziata)
```

Prima di effettuare un controllo con la media, è necessario comprendere se quest'ultima possa essere un buon indice di valutazione: per controllare ciò, è necessario calcolare il coefficiente di variazione rispetto alla media; se minore di 1, allora la media campionaria calcolata è un buon indice di valutazione.

$$CV = \frac{s}{|\overline{x}|}.$$

Il coefficiente di variazione è il rapporto tra la deviazione standard campionaria e il valore assoluto della media campionaria, quindi dobbiamo calcolare anche la deviazione standard. Anche per quest'ultima, R offre la funzione sd.

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2} \qquad (n = 2, 3, \ldots).$$

Riguardo il coefficiente di variazione, purtroppo R non fornisce alcuna funzione a riguardo, ergo ne costruiremo una adeguata.

```
sd(datiRaccoltaIndifferenziata)
cv(datiRaccoltaIndifferenziata)
```

```
cv <- function (x) {
  sd ( x) / abs ( mean (x) )
}
```

Fatto ciò, eseguiamo la funzione riguardante il coefficiente di variazione e notiamo che il suo valore è pari a 0.8471042.

Come detto poco fa, se il coefficiente di variazione è minore di 1, allora la media campionaria è un buon indice di valutazione. Per questo, quindi, possiamo confrontare la media campionaria con i singoli valori in tonnellate di spazzatura appartenente alla categoria di raccolta indifferenziata.

Riportiamo, di seguito, le tonnellate di rifiuti di raccolta indifferenziata per ogni regione, ordinati in maniera decrescente come fatto con il diagramma di Pareto.

Regioni	Raccolta indifferenziata
Sicilia	1800509.14
Lazio	1607960.81
Lombardia	1423822.16
Campania	1209746.68
Puglia	1117599.56
Toscana	1034845.95
Emilia-Romagna	1034390.55
Piemonte	840807.29
Veneto	615317.17
Calabria	466847.00
Liguria	424884.16
Marche	300266.41
Sardegna	267312.45
Abruzzo	262623.81
Friuli-Venezia Giulia	203354.28
Umbria	172705.73
Trentino-Alto Adige/Südtirol	147532.69
Basilicata	107409.02
Molise	80819.48
Valle d'Aosta /Vallée d'Aoste	28649.15

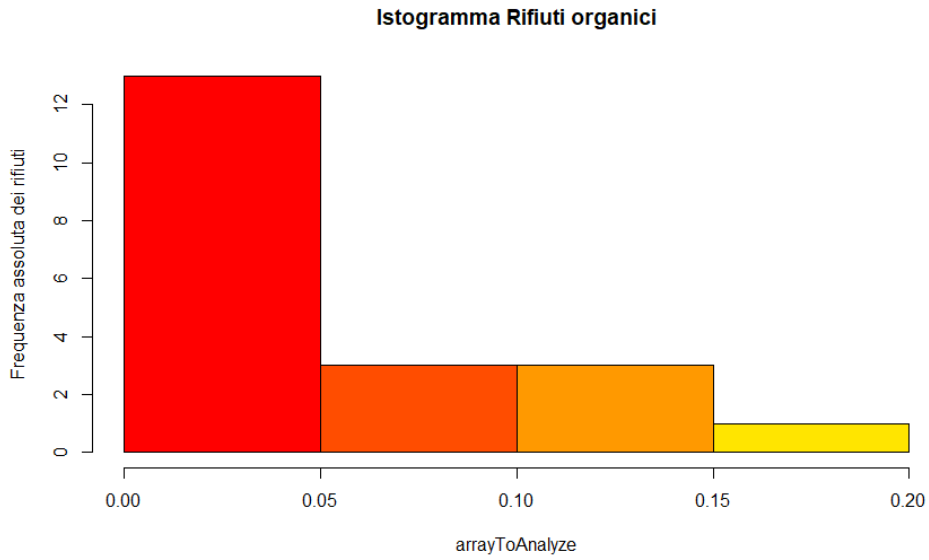
Notiamo subito come le prime sette regioni si distacchino notevolmente dalla media eccedendo in produzione di rifiuti appartenenti alla raccolta indifferenziata, in particolare la Sicilia.

La Sicilia si distacca notevolmente dalla media: dal dato di media campionaria notiamo che il valore per la regione Sicilia ha uno scarto molto elevato, difatti...

```
# Getting Sicily
SicilyValue = datiRaccoltaIndifferenziata[19]
waste = SicilyValue - mean(datiRaccoltaIndifferenziata)
waste # result is 1143139
```

Il valore dello scarto della media campionaria (1143139) testimonia riguardo l'enorme distacco della Sicilia rispetto alla media delle regioni: difatti, da qui si presume che il campione abbia un forte sbilanciamento verso destra nell'apposito istogramma.

Istogrammi



Gli istogrammi, che si utilizzano per variabili quantitative, sono una particolare rappresentazione grafica di una distribuzione di frequenza in classi. Il seguente è un istogramma riguardante i rifiuti organici per regione, come possiamo già notare nel grafico esiste solo una piccola quantità di regioni che producono molti rifiuti organici, la maggior parte hanno una quantità inferiore di consumi.
Tramite la funzione str () abbiamo dati come: fornisce i punti di suddivisione in classi (breaks), le frequenze assolute delle classi (counts), la densità delle classi (density) e i punti centrali delle classi (mids)

```
List of 6
 $ breaks  : num [1:5] 0 0.05 0.1 0.15 0.2
 $ counts  : int [1:4] 13 3 3 1
 $ density : num [1:4] 13 3 3 1
 $ mids    : num [1:4] 0.025 0.075 0.125 0.175
 $ xname   : chr "arrayToAnalyze"
 $ equidist: logi TRUE
 - attr(*, "class")= chr "histogram"
```

Oltre questo istogramma sono stati svolti tutti gli istogrammi per le tipologie di rifiuti

```
#histogram with garbage for region
createHisto("Istogramma Rifiuti Indifferenziata",datiRaccoltaIndifferenziata)
createHisto("Istogramma Rifiuti organici",datiUmido)
createHisto("Istogramma Carta e cartone",datiCarta)
createHisto("Istogramma Vetro",datiVetro)
createHisto("Istogramma Plastica",datiPlastica)
createHisto("Istogramma Altro",datiAltro)
```

Utilizzando il seguente codice

```
# createHisto: display histogram
# input -> name: gives a name to the Histogram
#          arrayToAnalyze: array with index to display
createHisto<-function(name,arrayToAnalyze){

  #calculating frequencies
  mySum = sum(arrayToAnalyze)
  arrayToAnalyze <- arrayToAnalyze / mySum
  h<-hist(arrayToAnalyze,freq=FALSE,main=name,ylab="Frequenza assoluta dei rifiuti",col=rainbow(length(arrayToAnalyze)))
  str(h)
}
```

BoxPlot

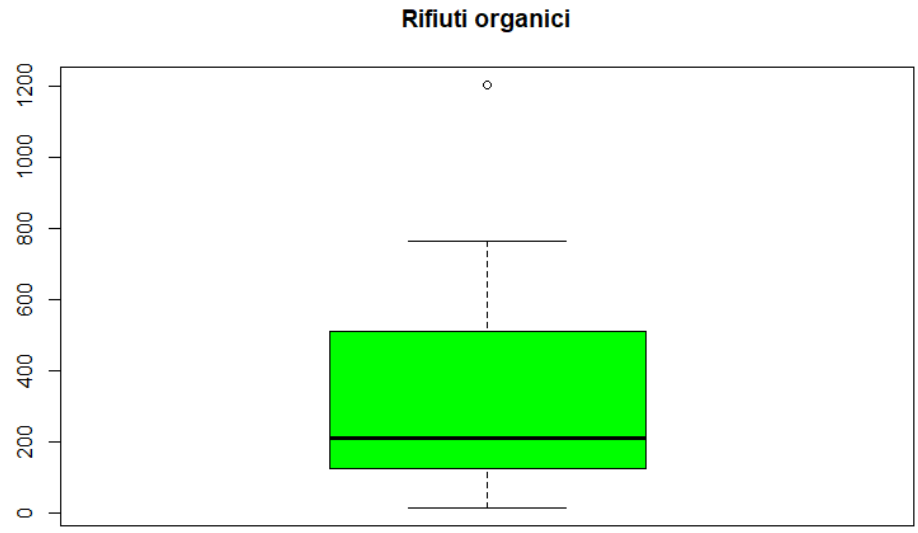
Il boxplot viene utilizzato per illustrare alcune caratteristiche di una distribuzione di frequenza come la centralità, la forma, la dispersione e la presenza di eventuali valori anomali, detti “outlier”.

Nell’immagine seguente abbiamo usato le funzioni quantile, summary e createBoxPlot_base su tutti i tipi di rifiuti per regione. Da notare che tutti i dati sono stati divisi per 1000.

La funzione Quantile ci ha permesso di capire i quantili, summary oltre i quantili ci ha permesso di ottenere la media e la mediana. Con la funzione createBoxPlot_base siamo riusciti a mostrare il grafico riguardante le varie tipologie di rifiuti.

```
quantile(datiRaccoltaIndifferenziata2)
quantile(datiUmido2)
quantile(datiCarta2)
quantile(datiVetro2)
quantile(datiPlastica2)
quantile(datiAltro2)

summary(datiRaccoltaIndifferenziata2)
summary(datiUmido2)
summary(datiCarta2)
summary(datiVetro2)
summary(datiPlastica2)
summary(datiAltro2)

#Display BoxPlot with garbage for region /1000
createBoxPlot_base("Rifiuti Indifferenziata",datiRaccoltaIndifferenziata2)
createBoxPlot_base("Rifiuti organici",datiUmido2)
createBoxPlot_base("Carta e cartone",datiCarta2)
createBoxPlot_base("Vetro",datiVetro2)
createBoxPlot_base("Plastica",datiPlastica2)
createBoxPlot_base("Altro",datiAltro2)
```

Nel seguente grafico troviamo il boxplot dei rifiuti organici per regione



Come possiamo notare nel grafico troviamo un valore anomalo, circa 1200, rispetto ai dati che sono compresi nel baffo.

Di seguito troviamo alcuni dati interessanti come i quantili, la media e la mediana relative ai dati della raccolta dell'umido.

```
  quantile(datiUmido2)
  0%    25%    50%    75%   100%
14.0  127.5  210.5  503.5 1206.0
  summary(datiUmido2)
 Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 14.0   127.5   210.5   330.6   503.5  1206.0
```

BoxPlot ad intaglio

I boxplot ad intaglio (notched boxplots) sono una rappresentazione grafica dei boxplot che permette di visualizzare anche l'intervallo di confidenza (intervallo di fiducia) del 95% per la mediana. Se si sceglie come grado di fiducia nella stima della mediana $1 - \alpha = 0.95$, si dimostra che per campioni numerosi l'intervallo di confidenza approssimato per la mediana

(M1, M2) = (M-1.57 $\frac{IRQ}{\sqrt{n}}$,M+ 1.57 $\frac{IRQ}{\sqrt{n}}$)

dove

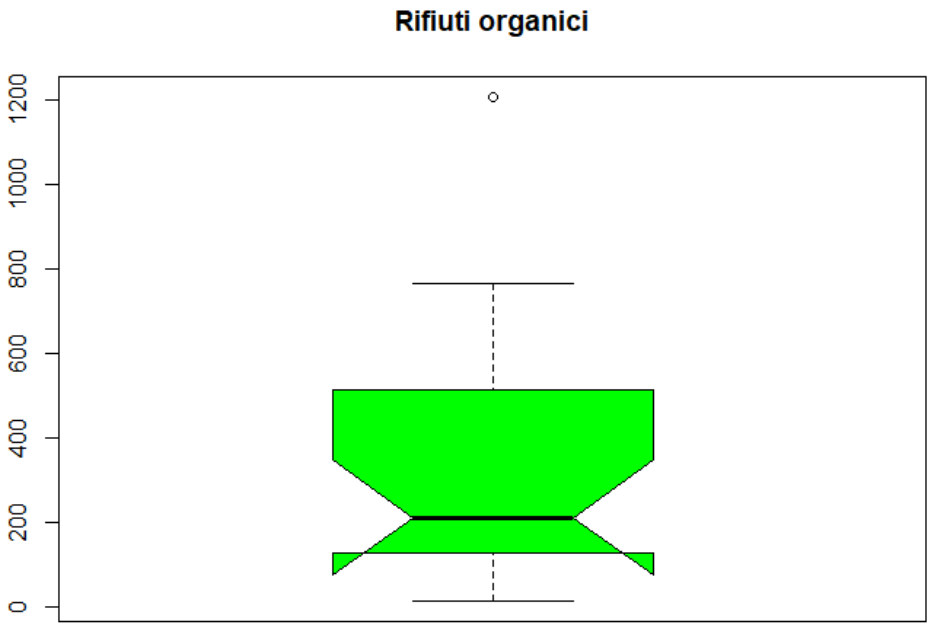
- M è la mediana;

- M1 è l'estremo inferiore dell'intervallo di confidenza per la mediana;

- M2 è l'estremo superiore dell'intervallo di confidenza per la mediana;

- IQR = Q3 − Q1 è lo scarto interquartile;

- n il numero di osservazioni nel campione.



Capitolo 4

Statistica descrittiva univariata

La statistica descrittiva è costituita da un insieme di metodi di natura logica e matematica atti a raccogliere, elaborare, analizzare ed interpretare dati allo scopo di descrivere fenomeni collettivi e di estendere la descrizione di certi fenomeni osservati ad altri fenomeni dello stesso tipo non ancora osservati.

Funzione di distribuzione empirica discreta

La funzione di distribuzione empirica F(x) è definita per ogni x reale ed è una funzione a gradini in cui ogni gradino indica quale proporzione di dati presenta un valore minore o uguale di quello indicato sull'asse delle ascisse.

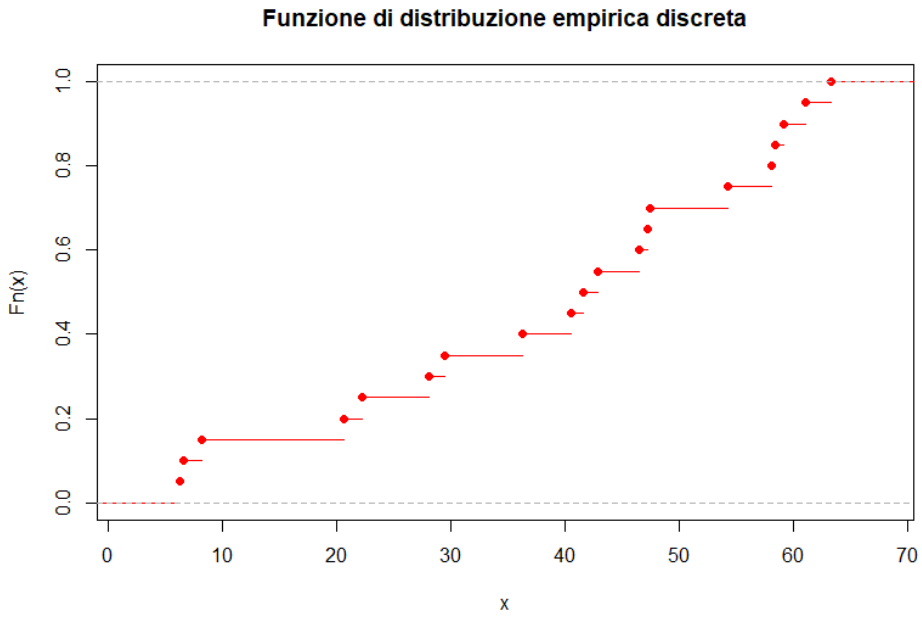
$$F(x) = \frac{\#\{x_i \leq x, i = 1, 2, \ldots, n\}}{n} = \begin{cases} 0, & x < z_1 \\ F_1, & z_1 \leq x < z_2 \\ \ldots & \\ F_i, & z_i \leq x < z_{i+1} \\ \ldots & \\ 1, & x \geq z_k \end{cases}$$

Il linguaggio R dispone della classe stepfun che implementa una serie di metodi per trattare funzioni a gradino.

Sono stati analizzati tutti i tipi di rifiuti e divisi per 1000.

```
#Display diagram of discrete empirical distribution

createEmpiricalDistribution(datiRaccoltaIndifferenziata2)
createEmpiricalDistribution(datiUmido2)
createEmpiricalDistribution(datiCarta2)
createEmpiricalDistribution(dativetro2)
createEmpiricalDistribution(datiPlastica2)
createEmpiricalDistribution(datiAltro2)
```

Un esempio è il seguente diagramma riguardante i dati della plastica.



Funzione di distribuzione empirica continua

Per fenomeni quantitativi continui occorre considerare la funzione di distribuzione empirica continua, ossia una funzione di distribuzione empirica strutturata in classi.

La funzione di distribuzione empirica continua è così definita:

$$F(x) = \begin{cases} 0, & x < z_0 \\ \ldots\ldots & \\ F_{i-1}, & x = z_{i-1} \\ \dfrac{F_i - F_{i-1}}{z_i - z_{i-1}}\, x + \dfrac{z_i F_{i-1} - z_{i-1} F_i}{z_i - z_{i-1}}, & z_{i-1} < x < z_i \\ F_i, & x = z_i \\ \ldots\ldots & \\ 1, & x \geq z_k, \end{cases}$$

Sono stati analizzati tutti i tipi di rifiuti e divisi per 1000.

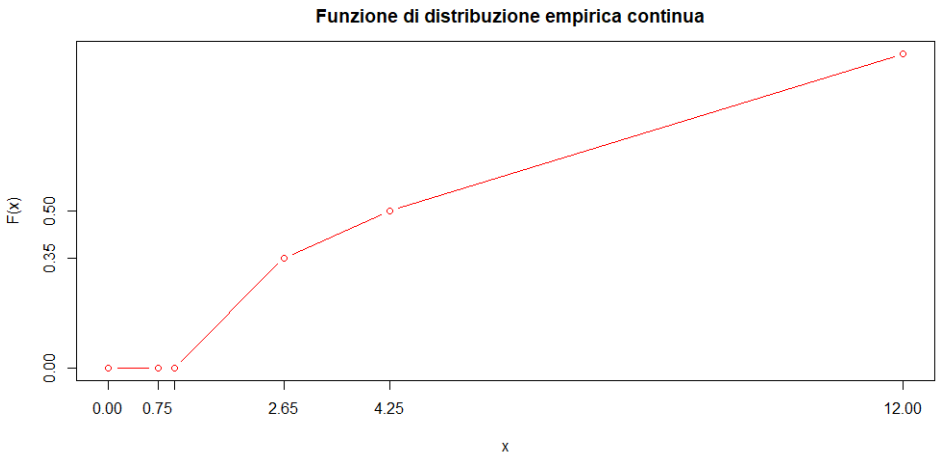
```
#Display diagram of continuous empirical distribution function
createContinuousEmpiricalDistribution(datiRaccoltaIndifferenziata2)
createContinuousEmpiricalDistribution(datiUmido2)
createContinuousEmpiricalDistribution(datiCarta2)
createContinuousEmpiricalDistribution(datiVetro2)
createContinuousEmpiricalDistribution(datiPlastica2)
createContinuousEmpiricalDistribution(datiAltro2)
```

Con il seguente codice per la funzione createContinuousEmpiricalDistribution:

```r
# createContinuousEmpiricalDistribution: display histogram
# input -> arrayToAnalyze: array with index to display
createContinuousEmpiricalDistribution<-function(arrayToAnalyze){

  #created relative frequency of values
  frequenza<-arrayToAnalyze/length(arrayToAnalyze)
  frequenza<-as.integer(frequenza)

  #m is the length of arrayToAnalyze
  m<-length(arrayToAnalyze)

  #I used quantiles to divide the problem into ranges
  q1<-quantile(frequenza,0.25)
  q2<-quantile(frequenza,0.5)
  q3<-mean(frequenza)
  q4<-quantile(frequenza,0.75)
  q5<-quantile(frequenza,1)
  classi<-c(q1,q2,q3,q4,q5)

  #create class closed on the right
  frelclassi <-table(cut(frequenza,breaks=classi,right=FALSE))/m

  #Fcum is the cumulative sum of relative frequencies
  Fcum<-cumsum(frelclassi)
  Fcum[4]<-Fcum[4]+frequenza[m]

  #max and min
  minn<-min(frequenza)
  #created abscissa and ordinate with previous values
  ascisse<-c(minn,q1,q2,q3,q4,q5)
  ordinate<-c(0,0,Fcum[1:3],1)
  #display plot
  plot(ascisse,ordinate,type="b",axes=FALSE,main="
  Funzione di distribuzione empirica continua",
       col="red",ylim=c(0,1),xlab="x",ylab="F(x)")
  axis(1,ascisse)
  axis(2,format(Fcum,digits=2))
  box()
}
```

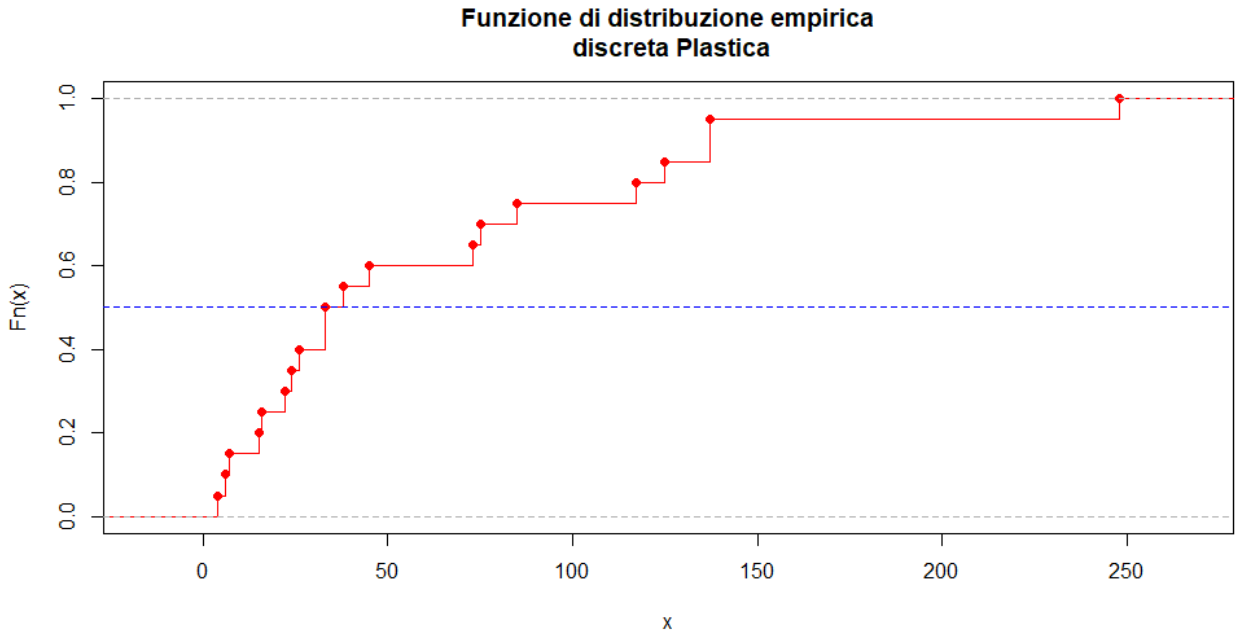
Si calcolano all'inizio le frequenze relative e le frequenze relative associate alle classi, nel nostro caso le classi sono chiuse a sinistra e aventi come valori i quantili e la media. Successivamente abbiamo disegnato il grafico contenente le frequenze, una linea che percorre i punti nel grafico e i cerchi che indicano~~ti~~ gli estremi.

Un esempio è il seguente diagramma riguardante i dati della plastica.



Abbiamo poi creato dei grafici che evidenziano i valori della mediana per le distribuzioni di frequenze. Svolto per tutti i tipi di rifiuti e diviso per 1000.

Nel nostro esempio troviamo i dati dei rifiuti della plastica.



Utilizzando il seguente snippet di codice.

```r
# createMedianGraph: display discrete empirical distribution function and value
#                    above and below the mean
# input ->  name : gives a name to the Pareto diagram
#           arrayToAnalyze: array with index to display
createMedianGraph<-function(name,arrayToAnalyze){

  #calculate cumulative sum and divide by length and round by 2
  Fdati1<-cumsum(arrayToAnalyze/length(arrayToAnalyze))
  round(Fdati1,2)

  #paste two string
  title<-paste("Funzione di distribuzione empirica\n discreta",name)

  #display value by color red and the line of media in blue
  plot(ecdf(arrayToAnalyze),main=title,verticals=TRUE,col="red")
  abline(h=0.5,lty=2,col="blue")

}
```

Quantili e algoritmi

In R esistono nove differenti algoritmi per calcolare i quantili ottenibili utilizzando la funzione quantile(v, probs =, type = j). dove v è un vettore numerico, probs è il vettore delle probabilità e j = 1, 2, ..., 9 denota il tipo di algoritmo selezionato. R utilizza di default per il calcolo dei quantili l'algoritmo di tipo 7, basato su tecniche di interpolazione tra i punti.

Nell'immagine seguente possiamo notare i dati dei rifiuti della plastica. Nella prima parte troviamo i dati grezzi, nella seconda sono i dati precedenti divisi per 1000.

Come possiamo notare sono stati utilizzati tutti e nove i tipi di algoritmi usati da R per calcolare i quantili in entrambi i tipi di dati. I diversi algoritmi forniscono diversi valori per i quantili, alcuni valori nei quantili sono simili come negli algoritmi di tipo 1,3 e 4 e quelli 2 e 5.

```
>   typesQuantiles(datiPlastica)
          0%      25%   50%       75%   100%
type 1 4156 16489.00 33569  85732.00 248268
type 2 4156 19604.00 35951 101383.50 248268
type 3 4156 16489.00 33569  85732.00 248268
type 4 4156 16489.00 33569  85732.00 248268
type 5 4156 19604.00 35951 101383.50 248268
type 6 4156 18046.50 35951 109209.25 248268
type 7 4156 21161.50 35951  93557.75 248268
type 8 4156 19084.83 35951 103992.08 248268
type 9 4156 19214.62 35951 103339.94 248268
>   typesQuantiles(datiPlastica2)
       0%    25%  50%      75% 100%
type 1  4 16.000 33.0  85.0000  248
type 2  4 19.000 35.5 101.0000  248
type 3  4 16.000 33.0  85.0000  248
type 4  4 16.000 33.0  85.0000  248
type 5  4 19.000 35.5 101.0000  248
type 6  4 17.500 35.5 109.0000  248
type 7  4 20.500 35.5  93.0000  248
type 8  4 18.500 35.5 103.6667  248
type 9  4 18.625 35.5 103.0000  248
```

Gli indici di posizione però non tengono conto della variabilità dei dati, infatti esistono distribuzioni di frequenze che sono molto diverse tra loro, pur avendo la stessa media campionaria. Indici significativi per misurare la variabilità di una distribuzione di frequenze sono la varianza campionaria e la deviazione standard campionaria, detta anche scarto quadratico medio campionario. La varianza e la deviazione standard sono tanto più grandi quando più i dati si discostano dalla media. In R la varianza campionaria di un vettore numerico v si calcola utilizzando la funzione var(v) e la deviazione standard campionaria si calcola utilizzando la funzione sd(v).

Assegnato un insieme di dati numerici x1, x2, . . . , xn, si definisce varianza campionaria, e si denota con s2, la quantità:

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2 \qquad (n = 2, 3, \ldots),$$

dove $\bar{x}$ denota la media campionaria dei dati. Inoltre, si definisce deviazione standard campionaria la radice quadrata della varianza campionaria, ossia:

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2} \qquad (n = 2, 3, \ldots).$$

La media campionaria e la deviazione standard campionaria sono i due indici di posizione e di dispersione dei dati maggiormente utilizzati. Per confrontare le variazioni esistenti tra diversi campioni di dati è utile introdurre il coefficiente di variazione.

Assegnato un insieme di dati numerici x1, x2, . . . , xn, si definisce coefficiente di variazione il rapporto tra la deviazione standard campionaria e il modulo della media campionaria, ossia:

$$CV = \frac{s}{|\overline{x}|}.$$

In R non è definita una funzione che calcola il coefficiente di variazione. Tale funzione può essere comunque facilmente implementata in R nel seguente modo:

```
# cv : calculate coefficient of variation
# input -> arrayToAnalyze: array with index to display
cv<-function(arrayToAnalyze){
  sd(arrayToAnalyze)/abs(mean(arrayToAnalyze))
}
```

Nell'esempio seguente troviamo la varianza, la deviazione standard campionaria e il coefficiente di variazione applicato ai dati dei rifiuti della plastica divisi per 1000.

```
> var(datiPlastica2)
[1] 3913.8
>   sd(datiPlastica2)
[1] 62.56037
>   cv(datiPlastica2)
[1] 0.9883155
```

Come possiamo notare il coefficiente di variazione è compreso tra 0 e 1.

Forma di una distribuzione di frequenze:

Skewness e curtosi campionaria

Esistono degli indici statistici che permettono di misurare quando una distribuzione di frequenze presenta simmetria o asimmetria oppure se essa è più o meno piccata. Un indice che permette di misurare la simmetria di una distribuzione di frequenze è la skewness campionaria (coefficiente di simmetria).

Skewness

Assegnato un insieme di dati numerici x1, x2,..., xn, si definisce skewness campionaria il valore:

$$\gamma_1 = \frac{m_3}{m_2^{3/2}}$$

Dove m3 denota il momento centrato campionario di ordine 3.
Si definisce momento campionario quando, assegnato un insieme di dati numerici

x1 , x2,..., xn, di ordine j e si denota con Mj la quantità:

$$M_j = \frac{1}{n}\sum_{i=1}^{n} x_i^j \qquad (j = 1, 2, \ldots).$$

Dalla definizione di skewness campionaria si nota che se la distribuzione di frequenze è simmetrica Y1, assume il valore 0, altrimenti Y1 > 0 per l'asimmetria positiva (ossia la distribuzione di frequenze ha la coda di destra più allungata) e Y1 < 0 per l'asimmetria negativa (ossia la distribuzione di frequenze ha la coda di sinistra più allungata).
Si nota che Y1 è un indice adimensionale, ossia è indipendente dall'unità di misura dei dati.

In R è stato scritto il seguente codice per rappresentare lo skewness

```
# skw: display skewness
# input ->  arrayToAnalyze: array with index to display
skw <-function (arrayToAnalyze){
  n<-length(arrayToAnalyze)
  m2<-(n-1)*var(arrayToAnalyze)/n
  m3<-(sum((arrayToAnalyze-mean(arrayToAnalyze))^3))/n
  m3/(m2^1.5)
}
```

Su tutti i tipi di rifiuti divisi per 1000.

```
#skewness /1000
skw(datiRaccoltaIndifferenziata2)
0.6411528
skw(datiUmido2)
1.282602
skw(datiCarta2)
1.112766
skw(datiVetro2)
1.840049
skw(datiPlastica2)
1.42234
skw(datiAltro2)
2.229513
```

Come possiamo notare tutti i dati sui rifiuti hanno asimmetria positiva, hanno quindi la coda di destra più allungata.

Curtosi campionaria

Un indice che permette di misurare la densità dei dati intorno alla media è la curtosi campionaria.

Assegnato un insieme di dati numerici x1, x2, . . . , xn, si definisce curtosi campionaria il valore:

$$\gamma_2 = \beta_2 - 3,$$

$$\beta_2 = \frac{m_4}{m_2^2},$$

Dove β è l'indice di Pearson, avendo denotato con m2 il momento centrato campionario di ordine 2 e con m4 il momento centrato campionario di ordine 4.

In R è stato scritto il seguente codice per rappresentare la curtosi campionaria.

```
# curt: display sample kurtosis
# input ->  arrayToAnalyze: array with index to display
curt <-function (arrayToAnalyze){
  n<-length(arrayToAnalyze)
  m2<-(n-1)*var(arrayToAnalyze)/n
  m4<-(sum((arrayToAnalyze-mean(arrayToAnalyze))^4))/n
  m4/(m2^2)-3
}
```

Su tutti i tipi di rifiuti e divisi di 1000.

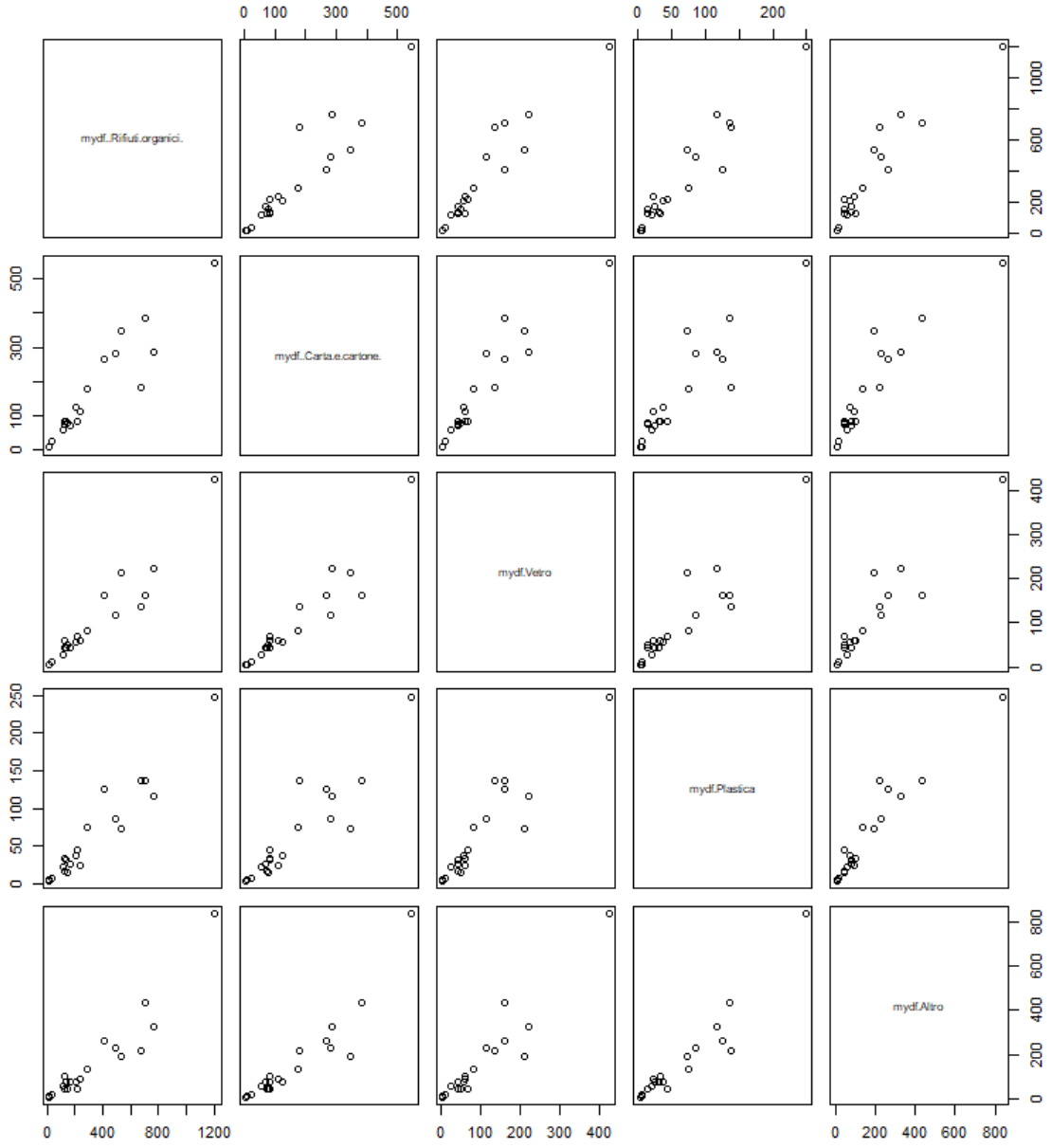
```
#sample kurtosis /1000
curt(datiRaccoltaIndifferenziata2)
-0.9023037
curt(datiUmido2)
1.078875
curt(datiCarta2)
0.5022303
curt(dativetro2)
3.461216
curt(datiPlastica2)
1.697922
curt(datiAltro2)
5.038462
```

Nella raccolta indifferenziata il valore è negativo, la distribuzione di frequenze è più piatta di una normale, ossia platicurtica. Per quanto riguarda invece la raccolta differenziata i valori sono tutti positivi, quindi la distribuzione di frequenze è più piccata di una normale, ossia leptocurtica.

Capitolo 5

Rifiuti organici correlati al cartone: studio della retta

La statistica descrittiva bivariata tende a descrivere le relazioni che intercorrono tra due variabili: in questo caso, studieremo le relazioni che intercorrono tra i rifiuti organici e gli altri tipi di rifiuti. Prima di cominciare, è bene specificare che le tonnellate di rifiuti sono state divise per 1000 al fine di rendere più leggibili i dati sui grafici.



Nel grafico soprastante vengono paragonate tutte le tipologie di rifiuti tra loro: la variabile posta sull'asse delle ascisse (quindi delle x) è indipendente, mentre la variabile posta sull'asse delle ordinate (quindi delle y) dipende da quella indipendente con cui si sta relazionando. Notiamo già, a primo impatto, che ogni variabile viene influenzata positivamente, siccome i

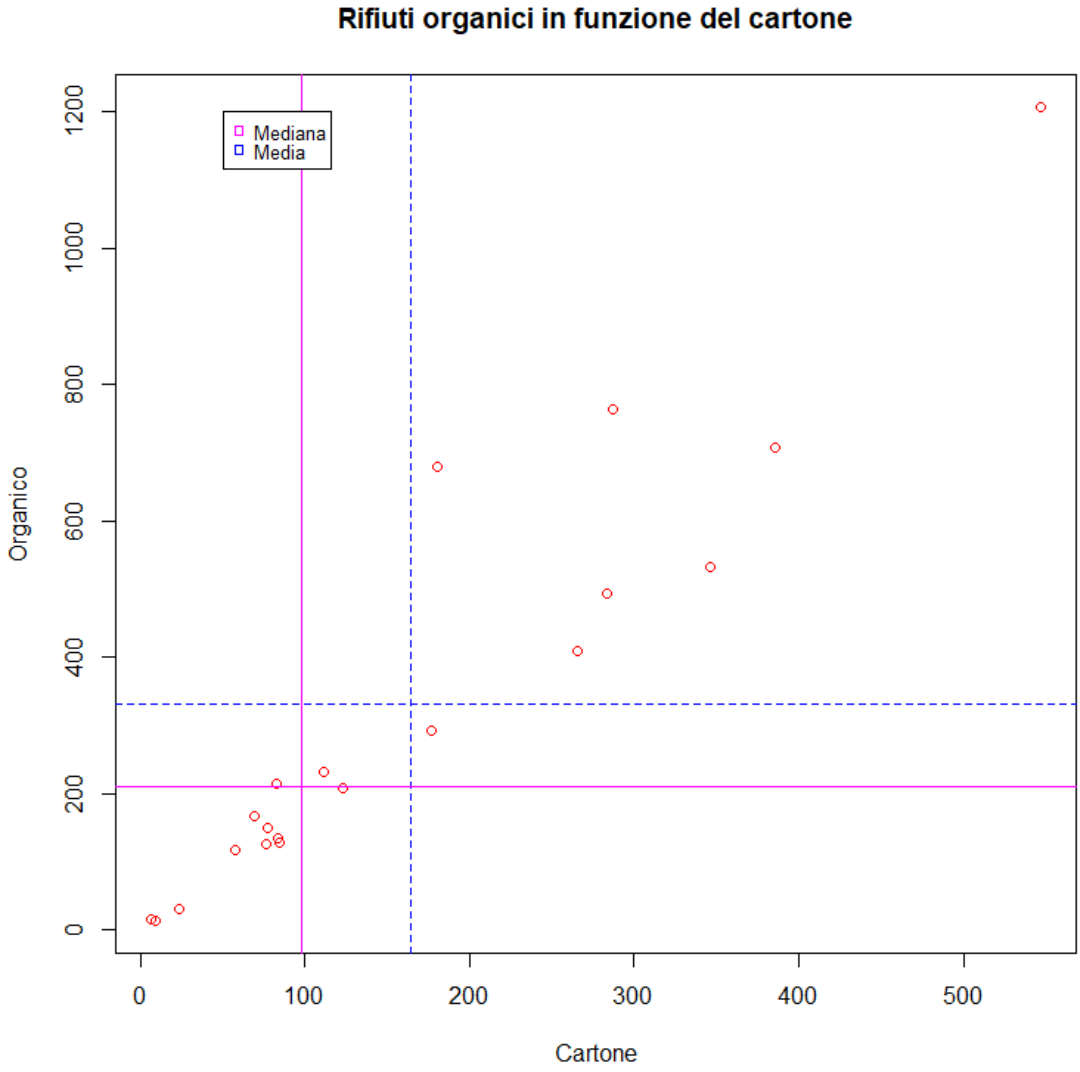
punti sui grafici sembrano definire delle rette ascendenti. Ovviamente è solo una deduzione: per essere certi, bisogna studiare il grafico.

```
> median(cartone)
[1] 97.9525
>   mean(cartone)
[1] 163.8551
>   median(umido)
[1] 210.986
>   mean(umido)
[1] 331.0972
```

Iniziamo con l'ipotizzare che i rifiuti organici, posti sulle y, vengano influenzati positivamente dal cartone, posto sulle x, quindi che con l'aumentare del cartone aumentano anche i rifiuti organici.

Regione	Carta	Organico
Molise	6802.491	14953.40
Valle d'Aosta /Vallée d'Aoste	9387.060	14581.43
Basilicata	23203.205	31233.75
Umbria	57202.077	116919.40
Friuli-Venezia Giulia	68954.301	166913.20
Calabria	76407.719	126579.81
Abruzzo	77083.039	149314.06
Sardegna	83011.918	213663.39
Trentino-Alto Adige/Südtirol	83541.500	133535.46
Liguria	84428.177	128256.93
Marche	111477.728	232083.52
Sicilia	123274.322	208309.14
Puglia	177167.647	291501.24
Campania	180334.677	678908.05
Piemonte	265958.788	409526.75
Toscana	283163.339	494221.74
Veneto	286931.398	764526.33
Lazio	346594.250	532659.36
Emilia-Romagna	385188.058	708243.52
Lombardia	546998.662	1206022.80

Segue un semplice plot mostrante i punti in cui vengono relazionati i dati, la media e la mediana dei dati.



Una prima deduzione, guardando la posizione dei punti di media e mediana, consiste nel fatto che i dati sembrano essere posizionati intorno ad una retta ascendente e ciò induce a pensare che esista una correlazione lineare positiva tra le variabili.

$$C_{xy} = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \overline{x})\,(y_i - \overline{y})$$

Per comprendere se tra le due variabili esiste una correlazione, iniziamo con il calcolare la covarianza: se assume valore positivo, significa che l'organico è correlato al cartone positivamente; se assume valore negativo, l'organico è correlato al cartone negativamente. Nel caso, invece, la covarianza assuma valore 0, allora le due variabili non sono correlabili tra loro.

$$r_{zw} = \frac{C_{zw}}{s_z\, s_w} = \frac{a\, c\, C_{xy}}{|a|\, s_x\, |c|\, s_y} = \frac{a\, c}{|a|\, |c|}\, r_{xy}$$

Per ottenere una misura quantitativa della correlazione tra le variabili, si considera il coefficiente di correlazione campionario, il quale può assumere un qualsiasi valore compreso tra -1 e 1. Se tale valore è pari a 0 non esiste alcuna correlazione tra le variabili considerate; se il valore è compreso tra 0 e 1 estremi esclusi, allora i punti saranno posizionati intorno ad una retta interpolante ascendente; se il valore è compreso tra -1 e 0, allora i punti saranno posizionati intorno ad una retta

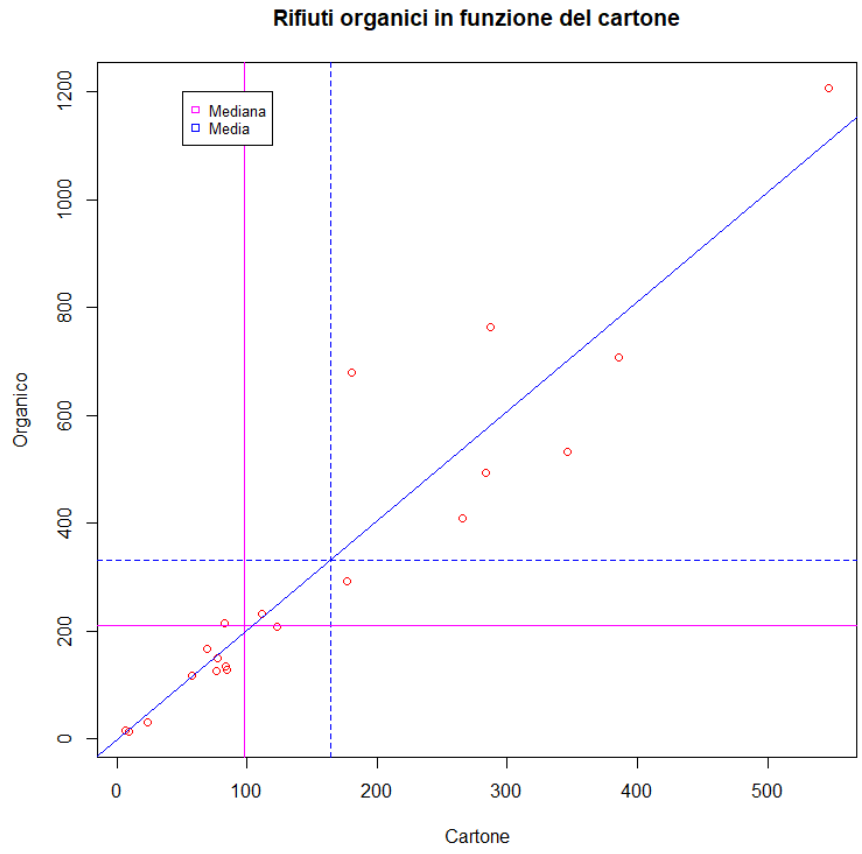
interpolante discendente; se il valore è 1, allora tutti i punti sono allineati su una retta ascendente; se il valore è -1 tutti i punti sono allineati su una retta discendente.

```
> cov(cartone, umido)
[1] 42420.59
>   cor(cartone, umido)
[1] 0.9409724
```

Osserviamo, quindi, i rispettivi valori della covarianza e del coefficiente di correlazione campionario: la covarianza assume un valore molto alto, quindi l'organico ha una forte correlazione positiva con il cartone. A testimonianza di ciò abbiamo il coefficiente di correlazione campionario, il quale assume un valore molto vicino ad 1, il quale indica la massima correlazione positiva.

Tramite il seguente snippet di codice creiamo il successivo grafico, il quale mostra la retta di regressione che interpola i punti.

```
plot(cartone, umido, main = "Rifiuti organici in funzione del cartone",
     xlab = "Cartone", ylab = "Organico", col = "red")
abline(v=median(cartone), lty=1, col="magenta")
abline(v=mean(cartone), lty=2, col="blue")
abline(h=median(umido), lty=1, col="magenta")
abline(h=mean(umido), lty=2, col="blue")
legend (50, 1200, c("Mediana", "Media"), pch =0, col=c("magenta", "blue"),
        cex =0.8)
abline(lm(umido~cartone), col="blue")
```



Notiamo subito che il punto della media si trova sulla retta di regressione ascendente: ciò sta a significare che la retta rappresenta al meglio il legame tra rifiuti organici e cartone; a testimonianza di ciò, difatti, abbiamo il coefficiente di correlazione campionario molto alto, vicino al valore massimo, il che indica una correlazione molto forte.

Come ben sappiamo, inoltre, la retta è raffigurabile tramite una semplice equazione

$$Y = \alpha + \beta X$$

in cui Y raffigura la variabile dipendente, X la variabile indipendente, α è detta intercetta e β è il coefficiente angolare, il quale indica la pendenza della retta di regressione.

Se il coefficiente angolare è positivo, allora la retta di regressione è crescente; se negativo, la retta di regressione è decrescente; se pari a zero, non c'è correlazione siccome la retta di regressione è orizzontale. L'intercetta, invece, corrisponde all'ordinata del punto di intersezione della retta interpolante. I rispettivi valori sono calcolabili nel seguente modo:

$$\beta = \frac{s_y}{s_x} r_{xy}, \qquad \alpha = \overline{y} - \beta \overline{x}.$$

```
>   c(alpha, beta)
[1] -1.572293  2.030266
```

Altro modo nel calcolare alpha e beta consiste nell'utilizzo di lm:

```
Call:
lm(formula = umido ~ cartone)

Coefficients:
(Intercept)      cartone
     -1.572        2.030
```

In sostanza, quindi, la retta di regressione ha equazione y = -1.572 + 2.030 x.

Rifiuti organici correlati al cartone: studio dei residui

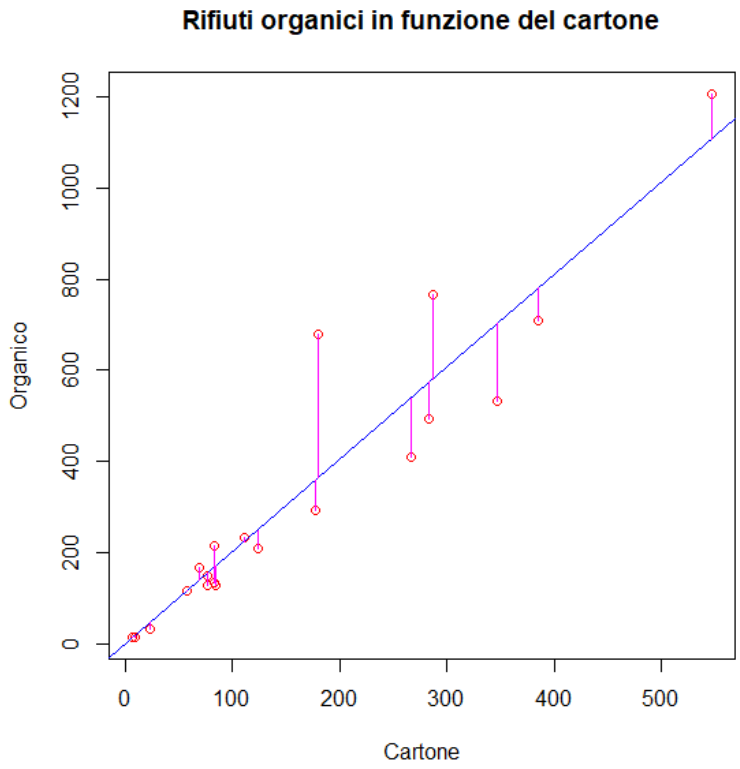
Una volta trovati i valori dell'intercetta e del coefficiente angolare, è possibile osservare quanto la retta di interpolazione si distacchi dai singoli punti che individuano le osservazioni.

Tra i punti riferenti alle osservazioni e la retta sono presenti degli scostamenti, detti residui, definiti nel seguente modo:

$$E_i = y_i - \widehat{y}_i = y_i - (\alpha + \beta\, x_i) \qquad (i = 1, 2, \ldots, n)$$

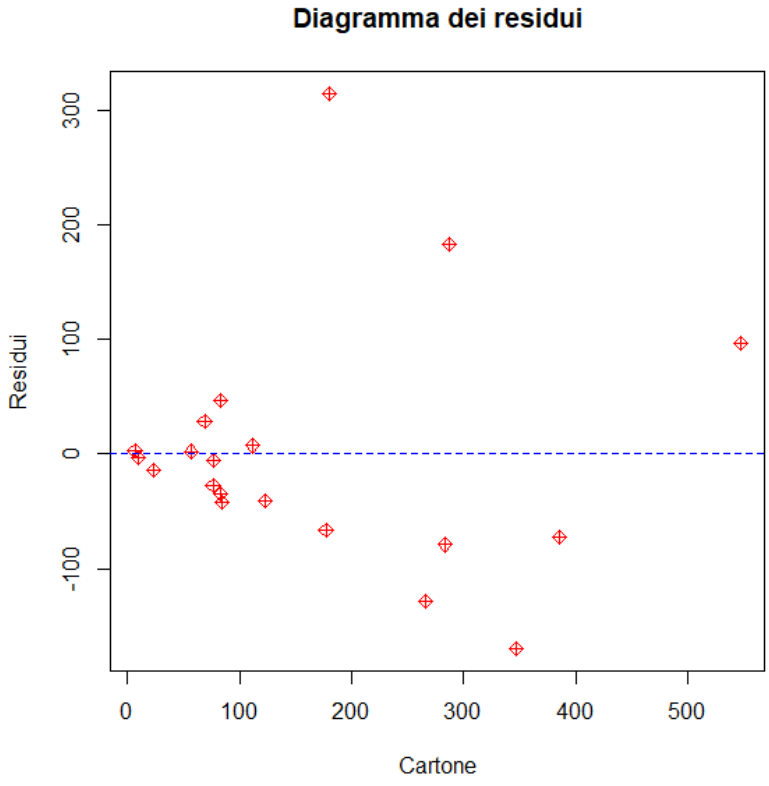
Mentre con (x_i, y_i) indichiamo un punto riferente alle osservazioni, con $(x_i, \hat{y}_i)$ indichiamo un punto sulla retta di interpolazione: la distanza dei due punti risulta essere il residuo. Siccome sull'asse delle x i punti assumono lo stesso valore, il residuo viene calcolato tenendo in considerazione solo l'asse delle y. Tramite il seguente snippet di codice vengono evidenziati tutti i residui sul grafico contenente la retta di regressione.

```
plot(cartone, umido, main = "Rifiuti organici in funzione del cartone",
     xlab = "Cartone", ylab = "Organico", col = "red")
abline(lm(umido~cartone), col="blue")
stime<-fitted(lm(umido~cartone))
segments(cartone, stime, cartone, umido, col="magenta")
```



Con il successivo snippet di codice, invece, si produce il diagramma dei residui.

```
residui <- resid(lm(umido~cartone))
plot(cartone, residui, main = "Diagramma dei residui",
     xlab = "Cartone", ylab = "Residui", col = "red", pch=9)
abline(h=0, col="blue", lty=2)
```



Il diagramma dei residui mostra quanto i residui si distaccano dalla loro media, la quale è nulla.

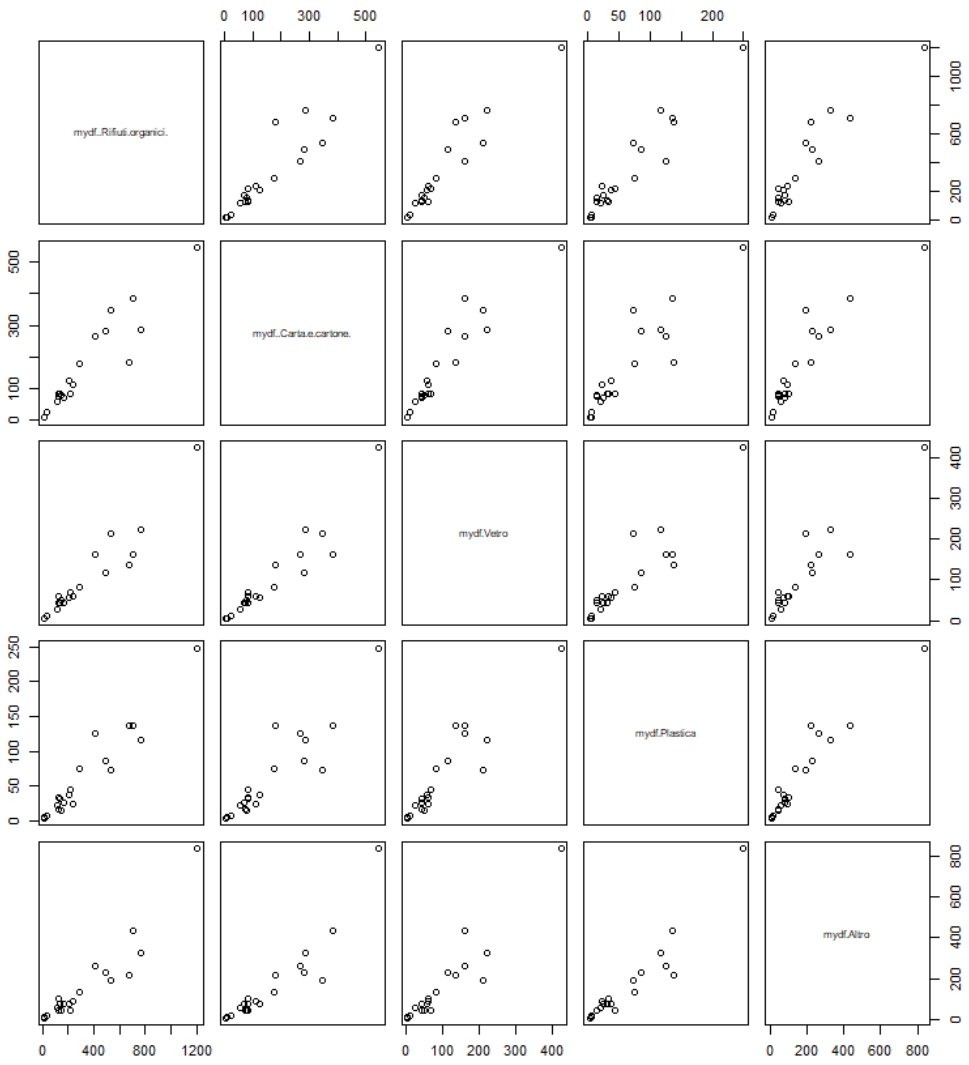
Poiché si è interessati a vedere quanto la retta si adatta ai dati, si stabilisce il coefficiente di determinazione, detto anche coefficiente di regressione, il quale è il rapporto tra la varianza dei valori stimati tramite la retta di regressione (quindi dei punti posti sulla retta) e la varianza dei valori osservati. Tale coefficiente può assumere valore tra 0 e 1: più si avvicina ad 1, più le due variabili di dati sono correlate tra loro.

```
>   summary (lm(umido~cartone))$r.square
[1] 0.885429
```

Con le verifiche fatte finora, quindi, possiamo affermare che tra i rifiuti organici ed il cartone è presente una forte relazione, in cui i rifiuti organici dipendono dalla produzione dei rifiuti carta e cartone.

Rifiuti organici correlati alle altre categorie di rifiuti

Nel precedente capitolo è stata studiata la correlazione tra i rifiuti organici ed il cartone; in questo capitolo si studieranno le correlazioni tra l'umido e tutte le altre categorie di rifiuti.



```
toAnalyze = data.frame(mydf$`Rifiuti organici`, mydf$`Carta e cartone`,
                       mydf$Vetro, mydf$Plastica, mydf$Altro)
toAnalyze = toAnalyze / 1000
pairs(toAnalyze)
```

La matrice di correlazione soprastante mostra tutte le variabili messe in corrispondenza tra loro.

Iniziamo con il ricavarci la matrice di covarianze campionarie e la matrice di coefficienti di correlazione campionari, per comprendere quanto siano forti le correlazioni tra variabili.

```
> cov(toAnalyze)
                    mydf..Rifiuti.organici. mydf..Carta.e.cartone. mydf.Vetro mydf.Plastica mydf.Altro
mydf..Rifiuti.organici.               97269.34              42420.586  29691.561     18766.030  57802.00
mydf..Carta.e.cartone.                42420.59              20894.099  13614.861      8283.951  26405.63
mydf.Vetro                            29691.56              13614.861   9953.606      5803.630  18541.16
mydf.Plastica                         18766.03               8283.951   5803.630      3910.508  11712.46
mydf.Altro                            57802.00              26405.634  18541.163     11712.462  38615.56
```

```
>   cor(toAnalyze)
                    mydf..Rifiuti.organici. mydf..Carta.e.cartone. mydf.Vetro mydf.Plastica mydf.Altro
mydf..Rifiuti.organici.               1.0000000             0.9409724  0.9542339     0.9622055  0.9431348
mydf..Carta.e.cartone.                0.9409724             1.0000000  0.9440851     0.9164507  0.9296161
mydf.Vetro                            0.9542339             0.9440851  1.0000000     0.9302355  0.9457265
mydf.Plastica                         0.9622055             0.9164507  0.9302355     1.0000000  0.9531263
mydf.Altro                            0.9431348             0.9296161  0.9457265     0.9531263  1.0000000
```

In generale sembra esserci forte correlazione in ogni coppia di variabili.

Nel nostro caso, però, studiamo i rifiuti organici in relazione a tutte le altre variabili, quindi si ricorre al modello di regressione lineare multipla con p variabili indipendenti, esprimibile tramite la seguente equazione:

$$Y = \alpha + \beta_1 X_1 + \beta_2 X_2 + \ldots \beta_p X_p.$$

Anche qui α risulta essere l'intercetta, mentre i diversi coefficienti angolari β_i risultano essere i regressori, quindi le inclinazioni di Y rispetto alla singola variabile X_i. Ricaviamo tramite il seguente snippet di codice l'intercetta e i regressori.

```
lm(toAnalyze$mydf..Rifiuti.organici.~toAnalyze$mydf..Carta.e.cartone.+
     toAnalyze$mydf.Vetro+toAnalyze$mydf.Plastica+toAnalyze$mydf.Altro)
```

Seguono i valori.

```
Call:
lm(formula = toAnalyze$mydf..Rifiuti.organici. ~ toAnalyze$mydf..Carta.e.cartone. +
    toAnalyze$mydf.Vetro + toAnalyze$mydf.Plastica + toAnalyze$mydf.Altro)
```

```
(Intercept)  toAnalyze$mydf..Carta.e.cartone.
    5.22277                            0.43426
```

```
toAnalyze$mydf.Vetro  toAnalyze$mydf.Plastica  toAnalyze$mydf.Altro
             1.01099                   2.60573                -0.07586
```

Pertanto, il modello di regressione multipla stimato è:

$$y = 5.22 + 0.43x_1 + 1.01x_2 + 2.61x_3 - 0.08x_4$$

Riguardo i residui, invece, è necessario ricavare, come prima, i valori stimati ottenuti mediante la regressione lineare multipla.

$$\widehat{y_i} = \alpha + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \ldots + \beta_p x_{i,p}$$

Da qui, si calcolano i residui.

$$E_i = y_i - \widehat{y}_i = y_i - (\alpha + \beta_1\, x_{i,1} + \beta_2\, x_{i,2} + \ldots \beta_p\, x_{i,p})$$

Anche in questo caso la media dei residui è nulla. Segue il calcolo dei valori stimati (cioè dei punti sulla retta) e dei residui.

```
stime = fitted(lm(toAnalyze$mydf..Rifiuti.organici.~
                  toAnalyze$mydf..Carta.e.cartone.+toAnalyze$mydf.Vetro+
                  toAnalyze$mydf.Plastica+toAnalyze$mydf.Altro))
residui = resid(lm(toAnalyze$mydf..Rifiuti.organici.~
                   toAnalyze$mydf..Carta.e.cartone.+toAnalyze$mydf.Vetro+
                   toAnalyze$mydf.Plastica+toAnalyze$mydf.Altro))
stime
residui
```

```
> stime
        1         2         3         4         5         6         7
 589.13595  31.47881 182.95890 1253.52829 166.06533 635.00314 143.52873
        8         9        10        11        12        13        14
 659.04210 452.17696 112.55595 173.32784 547.84999 125.05238  25.50459
       15        16        17        18        19        20
 564.50898 352.46601  46.05441 123.64671 210.56268 227.49626
>   residui
          1           2           3           4           5           6           7
-179.609947  -16.897807  -54.702899  -47.506290  -32.530330  129.522861   23.384271
          8           9          10          11          12          13          14
  49.200904   42.044045    4.363051   58.755164  -15.190994   24.261616  -10.551591
         15          16          17          18          19          20
 114.399025  -60.965015  -14.821411    2.932287   -2.253683  -13.833258
```

Concludiamo, infine, osservando il valore del coefficiente di determinazione, il quale ricordiamo permetta di stabilire quanto le variabili siano correlate tra loro.

```
> summary(lm(toAnalyze$mydf..Rifiuti.organici.~
+                toAnalyze$mydf..Carta.e.cartone.+toAnalyze$mydf.Vetro+
+                toAnalyze$mydf.Plastica+toAnalyze$mydf.Altro))$r.square
[1] 0.9556334
```

Capitolo 6

Analisi dei Cluster

L'analisi dei cluster è una metodologia che permette di raggruppare in sottoinsiemi, detti cluster, entità (unità) appartenenti ad un insieme più ampio. I vari metodi attraverso cui si attua l'analisi dei cluster hanno in comune uno scopo generale, di distribuire le osservazioni in gruppi in modo tale che il grado di naturale associazione sia alto tra i membri dello stesso gruppo e basso tra i membri di gruppi diversi. Gli individui che sono assegnati allo stesso cluster sono detti simili mentre gli individui che sono assegnati a differenti cluster sono detti dissimili. In questo modo si otterrà quindi un'alta omogeneità all'interno dei gruppi e un'alta eterogeneità tra gruppi distinti.

I metodi di analisi dei cluster nel loro complesso permettono di raggiungere i seguenti obiettivi come l'individuazione di una reale tipologia, previsioni basate su gruppi, esplorazione dei dati, generazione di ipotesi di ricerca, verifica di ipotesi, riduzione della complessità dei dati.

Sia I = {I1, I2, . . ., In} un insieme di n individui (entità o unità) appartenenti ad una popolazione.
Assumiamo che esista un insieme di caratteristiche (features) C = {C1, C2, . . . ,Cp} che sono osservabili e sono possedute da ogni individuo in I.
Il termine osservabile denota caratteristiche che danno origine a dati sia di tipo qualitativo che di tipo quantitativo (detti anche misure).
Denotiamo con il simbolo xij il valore della misura della caratteristica j esima relativa all'individuo Ii e con Xi = (xi1, xi2, . . . , xip) (i = 1, 2, . . . , n) il vettore di cardinalità 1×p di tali misure. Quindi:

$$X = \begin{pmatrix} x_{11} & x_{12} & \dots & x_{1p} \\ x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{np} \end{pmatrix} = \begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{pmatrix},$$

dove Xi = (xi1, xi2, . . . , xip) (i = 1, 2, . . . , n).

Le misure metriche di somiglianza sono soprattutto basate sulle funzioni distanza tra i vettori delle caratteristiche. Occorre quindi definire tale funzione.

Una funzione a valori reali d(Xi,Xj) è detta funzione distanza se e soltanto se essa soddisfa le seguenti condizioni:

$$(i)\ d(X_i, X_j) = 0\ se\ e\ solo\ se\ X_i = X_j,\ con\ X_i\ e\ X_j\ in\ E_p;$$

$$(ii)\ d(X_i, X_j) \geq 0\ per\ ogni\ X_i\ e\ X_j\ in\ E_p;$$

$$(iii)\ d(X_i, X_j) = d(X_j, X_i)\ per\ ogni\ X_i\ e\ X_j\ in\ E_p;$$

$$(iv)\ d(X_i, X_j) \leq d(X_i, X_k) + d(X_k, X_j)\ per\ ogni\ X_i,\ X_j\ e\ X_k\ in\ E_p.$$

La proprietà (i) implica che Xi è a distanza zero da sé stesso e che ogni due punti a distanza nulla debbono essere identici. La proprietà (ii) afferma che la funzione distanza è non negativa. La proprietà (iii) impone la simmetria richiedendo che la distanza tra Xi e Xj deve essere la stessa della distanza tra Xj e Xi. La proprietà (iv), nota come disuguaglianza triangolare, richiede che la distanza tra Xi e Xj debba essere sempre minore o uguale della somma delle distanze di ognuno dei due vettori considerati da qualunque altro terzo vettore Xk.

In generale, le distanze tra tutte le possibili coppie di unità sono inserite in una matrice simmetrica D di cardinalità n × n, ossia

$$D = \begin{pmatrix} 0 & d_{12} & \dots & d_{1n} \\ d_{21} & 0 & \dots & d_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ d_{n1} & d_{n2} & \dots & 0 \end{pmatrix},$$

dove dij = d(Xi,Xj) (i, j = 1, 2, . . . , n).

Dalla definizione segue che i termini sulla diagonale principale sono tutti uguali a zero mentre i termini simmetrici sono uguali a due a due.
Non esiste una sola funzione distanza ma esiste un'intera famiglia di funzioni che rispettano almeno le quattro proprietà della definizione della funzione distanza. Esistono quindi le seguenti proprietà:

(a) se d e d′ sono due metriche anche d + d′ è una metrica;

(b) se d è una metrica e c un numero reale positivo allora anche c d è una metrica;

(c) se d è una metrica e c un numero reale positivo allora anche d′ = d/(c+d) è una metrica.

Le opzioni disponibili per il calcolo della matrice delle distanze sono:

(1) metrica euclidea (euclidean);

(2) metrica del valore assoluto o metrica di Manhattan (manhattan);

(3) metrica del massimo o metrica di Chebycev (maximum);

(4) metrica di Minkowski (minkowski);

(5) distanza di Canberra (canberra);

(6) distanza di Jaccard (binary);

In R la funzione:

> dist (X,method = ”euclidean”, diag = FALSE, upper = FALSE)

ritorna la matrice delle distanze D calcolata utilizzando le misure di distanza tra le righe della matrice X dei dati, dove:

- X rappresenta una matrice numerica o un data frame;

- method seleziona la misura di distanza da utilizzare (di default è euclidean);

- diag è posta uguale a TRUE se si desidera che la matrice delle distanze contenga anche i valori nulli sulla diagonale (di default `e FALSE);

- upper è posta uguale a TRUE se si desidera che la matrice delle distanze

contenga anche i valori al di sopra della diagonale principale (di default è FALSE).

Nella funzione dist() potrebbe essere presente anche un ulteriore parametro finale r che indica la potenza della metrica di Minkowski (di default è r = 2, che corrisponde alla metrica euclidea).

Una volta calcolata una matrice d delle distanze, per visualizzare la matrice completa si può utilizzare la funzione as.matrix(d).

Matrice Euclidea

La metrica Euclidea è così definita:

$$d_2(X_i, X_j) = \left[\sum_{k=1}^{p}(x_{ik} - x_{jk})^2\right]^{1/2},$$

dove xik è il valore della k–esima caratteristica dell’individuo Ii. La distanza Euclidea usata su tutti i dati è fortemente influenzata dall’unità di misura in base alla quale è valutata ciascuna delle p caratteristiche.

Il seguente snippet di codice rappresenta il cluster utilizzando la distanza euclidea.

I dati di base sono stati trasformati in interi e divisi per 1000, abbiamo dato nome alle righe e colonne e successivamente calcolato il cluster con le distanze applicando il metodo euclideo.

```
distance<-round(df/1000,0)
rownames(distance)<-namesRegion()
colnames(distance)<-namesGarbage()
dist(distance,method="euclidean",diag=TRUE,upper=TRUE)
#scale and standardize data
d<-scale(distance)
#now we calculate again the data but by scale and standardize data
dist(d,method="euclidean",diag=TRUE,upper=TRUE)
```

Poiché la funzione distanza è legata alle unità di misura in maniera non invariante si utilizzano, per ovviare a questo inconveniente, dei metodi per scalare e standardizzare i dati.
In R per scalare e standardizzare le variabili si utilizza la funzione
scale(X, center = TRUE, scale = TRUE).
Dove:
- X rappresenta una matrice numerica o un data frame;
- center è posta uguale a TRUE se dagli elementi di ogni colonna della matrice X si sottrae il valore medio della corrispondente colonna (di default è TRUE);
- scale è posta uguale a TRUE se si dividono gli elementi centrati di ogni colonna della matrice X per la deviazione standard della corrispondente colonna (di default è TRUE).

Successivamente abbiamo usato di nuovo la funzione dist sui dati scalati e standardizzati.

Misure di Similarità

Nella maggior parte delle tecniche di clustering occorre inizialmente calcolare la matrice D delle distanze oppure una matrice S delle similarità. Una misura di similarità fornisce un valore numerico compreso tra 0 e 1 e permette di definire in modo quantitativo la somiglianza o differenza tra due individui Ii e Ij, con 0 l'assoluta assenza e con 1 la massima presenza di somiglianza.

Per definizione una funzione a valori reali sij = s(Xi,Xj) è detta misura di similarità se e soltanto se essa soddisfa le seguenti condizioni:

$$(i)\ s(X_i, X_i) = 1;$$

$$(ii)\ 0 \leq s(X_i, X_j) \leq 1;$$

$$(iii)\ s(X_i, X_j) = s(X_j, X_i)\ \textit{per ogni}\ X_i\ e\ X_j.$$

La proprietà (i) implica che la misura di similarità è unitaria se i due punti sono identici. La proprietà (ii) afferma che la misura di similarità è compresa tra 0 e 1. La proprietà (iii) impone la simmetria richiedendo che la misura di similarità tra Xi e Xj deve essere la stessa della misura di similarità tra Xj e Xi. La quantità sij è chiamata semplicemente coefficiente di

similarità e risulta essere l'elemento nella riga i-esima e colonna j-esima della matrice di similarità S, definita:

$$S = \begin{pmatrix} 1 & s_{12} & \dots & s_{1n} \\ s_{21} & 1 & \dots & s_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ s_{n1} & s_{n2} & \dots & 1 \end{pmatrix}.$$

Un esempio di misura di similarità per vettori binari è il complemento a 1 della distanza di Jaccard, detto coefficiente di similarità di Jaccard, ossia

$$s(X_i, X_j) = \frac{n_{11}}{n_{11} + n_{01} + n_{10}}.$$

Infatti, si nota che le condizioni (ii) e (iii) sono soddisfatte; inoltre, se Xi = Xj risulta che

n01 = n10 = 0 e quindi anche la prima condizione è valida.

Divisione in cluster tramite metodologie gerarchiche

Per dividere il dataset in cluster è possibile adottare dei metodi gerarchici, i quali non si occupano più di trovare sin da subito una suddivisione in cluster migliore, ma permettono di costruire un diagramma detto dendrogramma, il quale raggruppa iterativamente gli elementi formando cluster sempre più grandi posti ad albero: si parte dalla base in cui in ogni cluster sarà posto un singolo individuo, fino ad arrivare alla radice che avrà un singolo cluster per tutti gli individui.

Gli algoritmi di clustering che trattano la divisione gerarchica si suddividono in due categorie: gli algoritmi di tipo agglomerativi partono dalle foglie del dendrogramma per poi arrivare alla radice, quindi eseguono in bottom-up; gli algoritmi di tipo divisivo partono dalla radice per poi arrivare alle foglie, lavorando quindi in maniera opposta rispetto agli agglomerativi, quindi in top-down.

Riguardo l'analisi dei dati, utilizzeremo algoritmi agglomerativi che si baseranno su una metrica di distanza, cioè la distanza euclidea, e una metodologia di accorpamento, la quale funziona nel seguente modo:

- Si considera la matrice delle distanze (in base alla metrica scelta);
- Si individua la coppia di cluster meno distanti (o più somiglianti) e si raggruppano in un unico cluster;
- Si costruisce una nuova matrice di distanze che risulterà essere ridotta di una riga ed una colonna a causa dell'unione di due cluster;
- Si itera il procedimento fino ad arrivare ad avere una matrice di cardinalità 2x2: ergo, la procedura richiede n-1 iterazioni, dove n è il numero di righe e colonne;
- Si rappresenta graficamente il processo di agglomerazione tramite un dendrogramma.

I cinque passi elencati costituiscono la base di molti metodi gerarchici: le differenze si riscontrano nei primi due passi, siccome cambia la scelta della misura di distanza e il modo in cui si individuano i due cluster meno distanti.

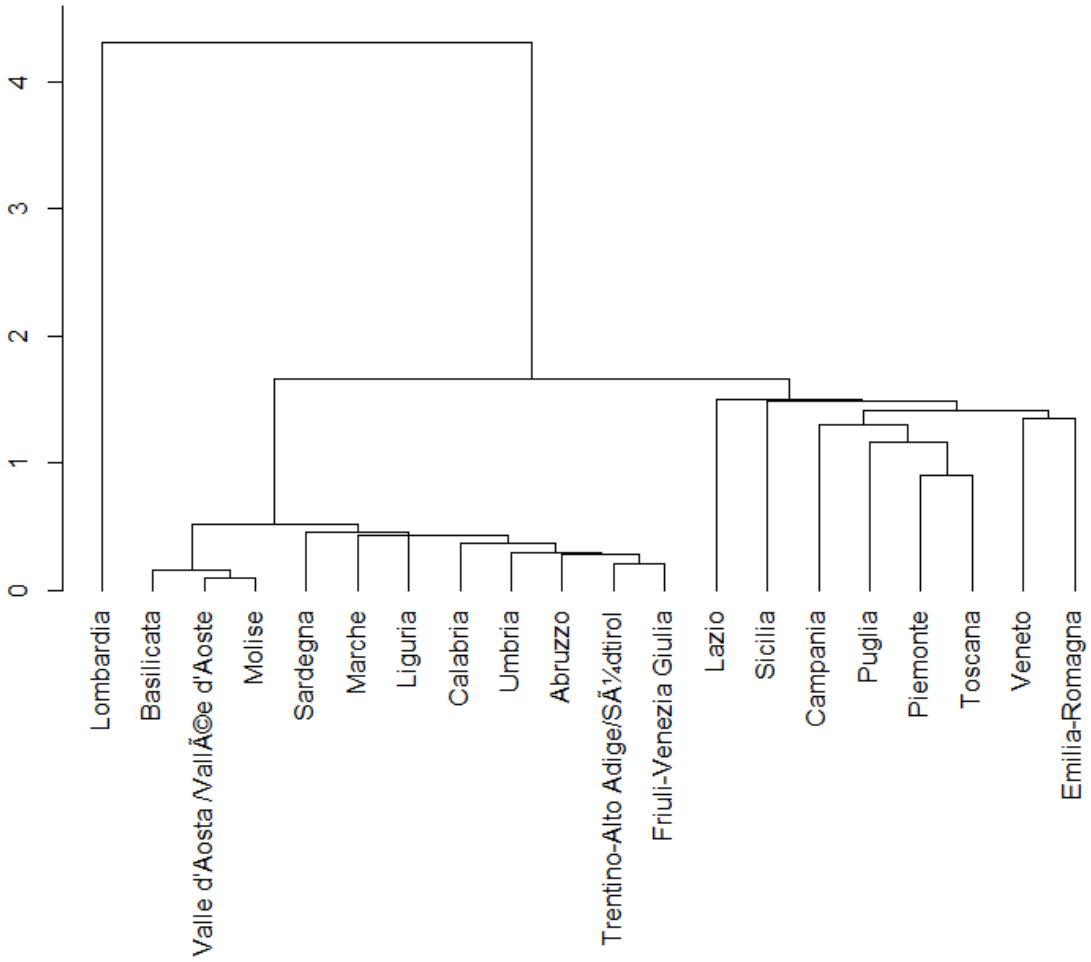
L'analisi agglomerativa gerarchica avviene tramite l'utilizzo di una funzione chiamata hclust, la quale specifica il preciso metodo gerarchico da utilizzare.

Metodo del legame singolo

Il primo metodo è quello del legame singolo, il quale accorpa i cluster basandosi sulla distanza più piccola. Una volta fatto l'accorpamento, si ricalcolano le distanze e si riconsidera la nuova distanza minima. Si continua fino ad avere un unico cluster.

Segue il dendrogramma generato dalle iterazioni del metodo del legame singolo.

```
set <- scale(df)
row.names(set) <- namesRegioni
d <- dist(set);
hlssingle <-hclust (d, method = "single");
str(hlssingle);

plot(hlssingle, hang =-1,
     xlab="Metodo del legame singolo")
```



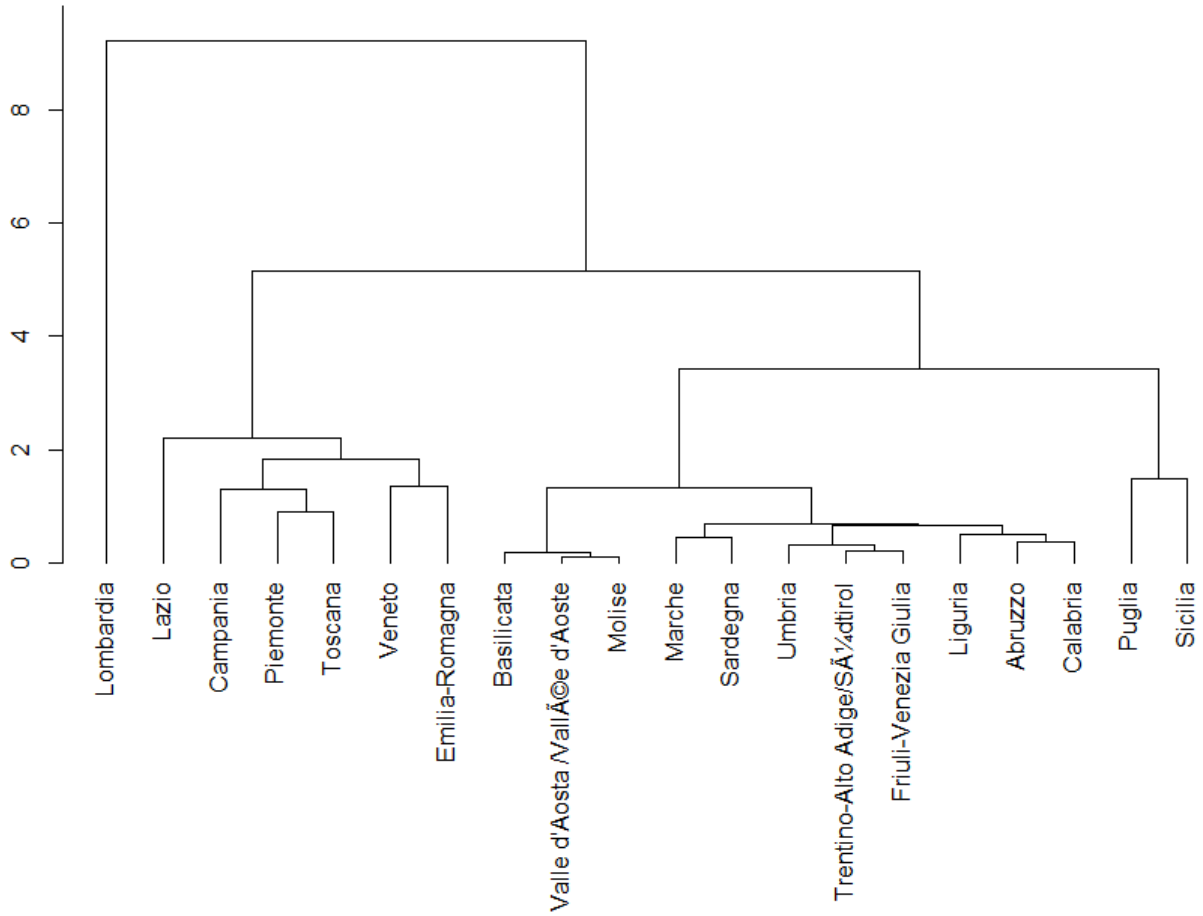
Dal dendrogramma notiamo che il dataset è suddivisibile in tre principali cluster, di cui uno composto solamente dalla Lombardia. È bene ribadire, però, che la suddivisione in cluster

dipende anche dal metodo utilizzato, quindi vediamo anche gli altri metodi che cluster producono.

Metodo del legame completo

Il metodo del legame completo effettua un'analisi opposta rispetto a quella del legame singolo, nel senso che considera le distanze massime tra gli elementi esterni al cluster.

```
hlsComplete <-hclust (d, method = "complete");
str(hlsComplete);

plot(hlsComplete, hang =-1,
     xlab="Metodo del legame completo")
```

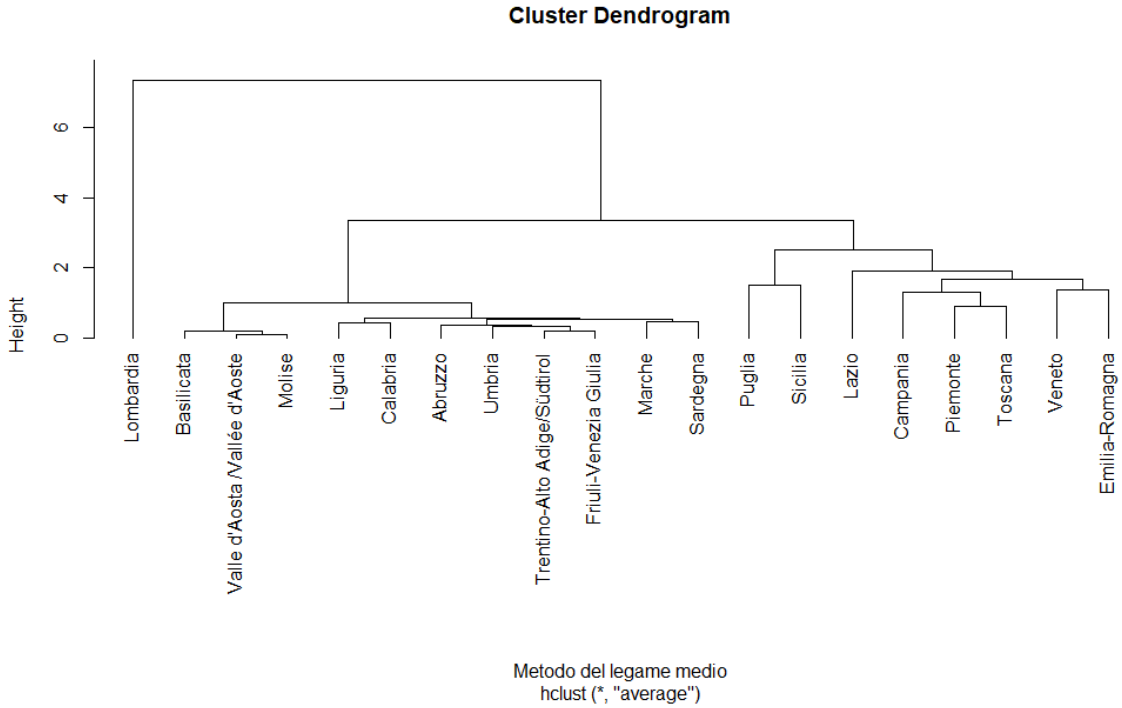


Con l'utilizzo del metodo del legame completo si notano subito grandi differenze riguardo la divisione in cluster: ora sono presenti ben 4 cluster principali e le città sono disposte in maniera differente nei cluster rispetto al precedente metodo. Nuovamente, la Lombardia è posta in un cluster composto da un unico elemento.

Metodo del legame medio

Il terzo metodo pone il calcolo della nuova distanza effettuando una media tra tutte le possibili distanze tra i valori interni nel cluster appena creato e quelli all'esterno. È importante sapere che il numero di individui in ogni cluster incide molto sul calcolo delle nuove distanze, difatti cluster più grandi avranno un peso maggiore rispetto agli altri.

```r
hlsAvarage<-hclust (d, method = "average");
str(hlsAvarage);

plot(hlsAvarage, hang =-1,
     xlab="Metodo del legame medio")
```

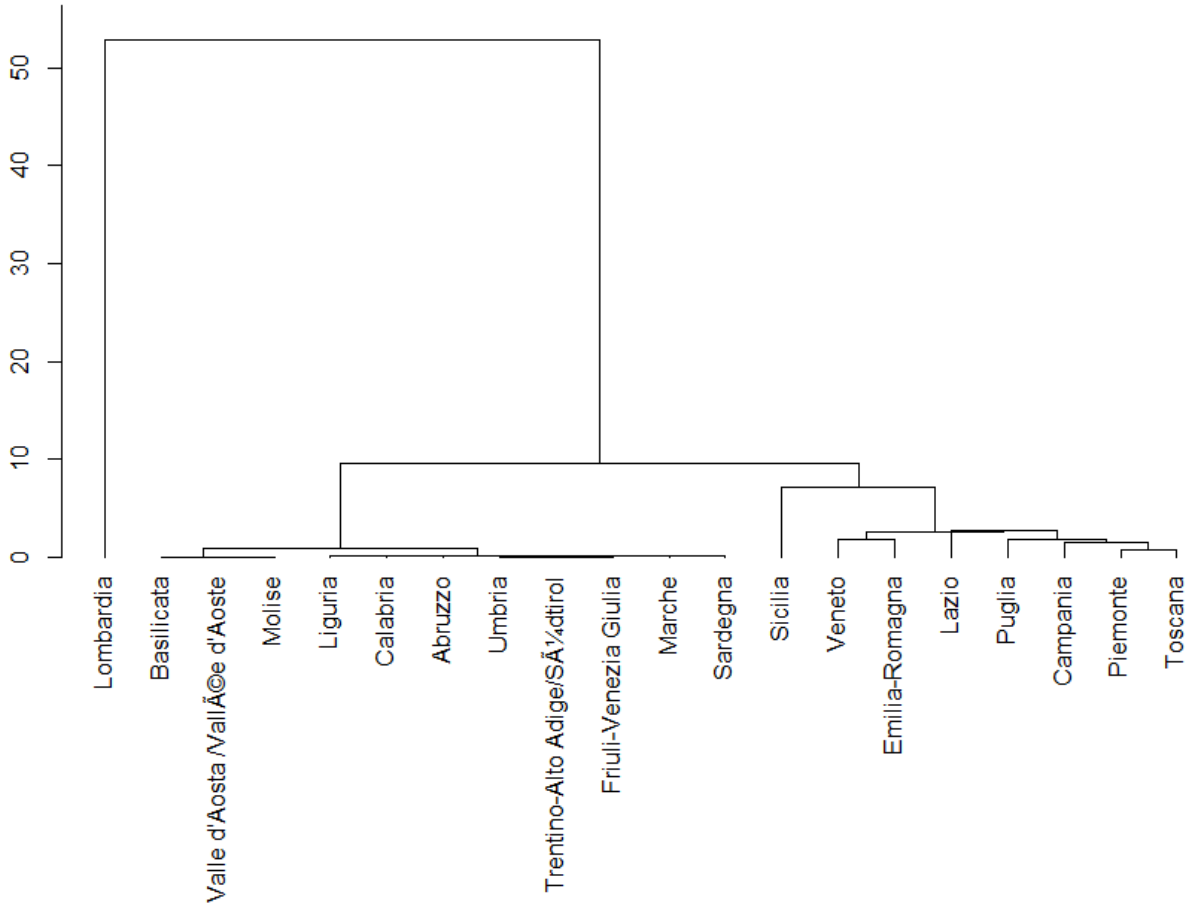


Quest'ultimo dendrogramma è molto simile a quello riprodotto dal metodo del legame singolo: i principali cluster individuati sono gli stessi e i raggruppamenti sono simili.

Metodo del centroide

Il metodo del centroide utilizza una metrica basata sulle distanze dei centroidi dei gruppi, dove il centroide è la media campionaria tra tutti gli elementi dei due cluster in considerazione. Come per il metodo precedente, anche in questo caso è rilevante il numero degli elementi appartenenti ad un singolo cluster, siccome la metrica perde l'appellativo di distanza dato che in questo nuovo caso si considera la distanza quadratica.

```r
hlsCentroid <-hclust (d^2, method = "centroid");
str(hlsCentroid);

plot(hlsCentroid, hang =-1,
     xlab="Metodo del centroide")
```

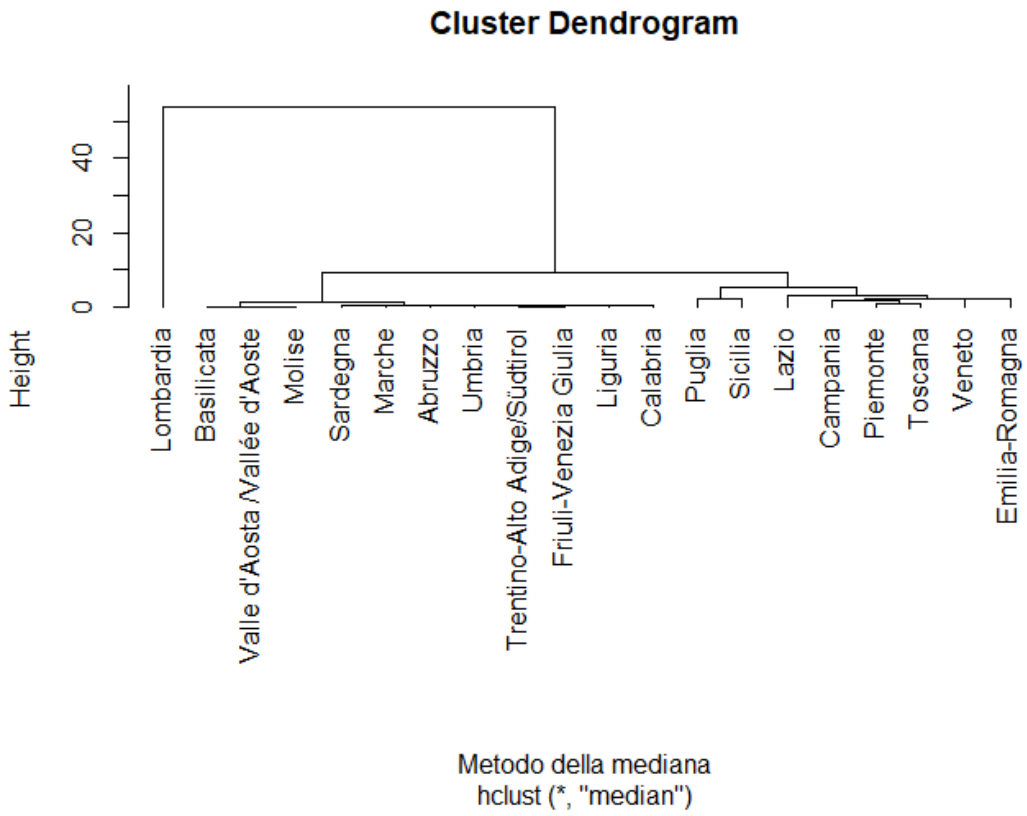


A parte le distanze quadratiche, anche questo dendrogramma risulta essere molto simile a quello del metodo del legame singolo, siccome i cluster sono molto simili.

Metodo della mediana

Oltre al metodo del centroide, anche il metodo della mediana si basa sulle distanze elevate al quadrato. Tale metodo è molto simile a quello del centroide, con la differenza che non si basa sulla numerosità dei cluster.

```
hlsMedian <-hclust (d^2, method = "median");
str(hlsMedian);

plot(hlsMedian, hang =-1,
     xlab="Metodo della mediana")
```



Anche il metodo della mediana sembra non influire moltissimo rispetto alla divisione precedente. In sostanza, l'unico metodo che sembra distaccarsi è quello del legame completo.

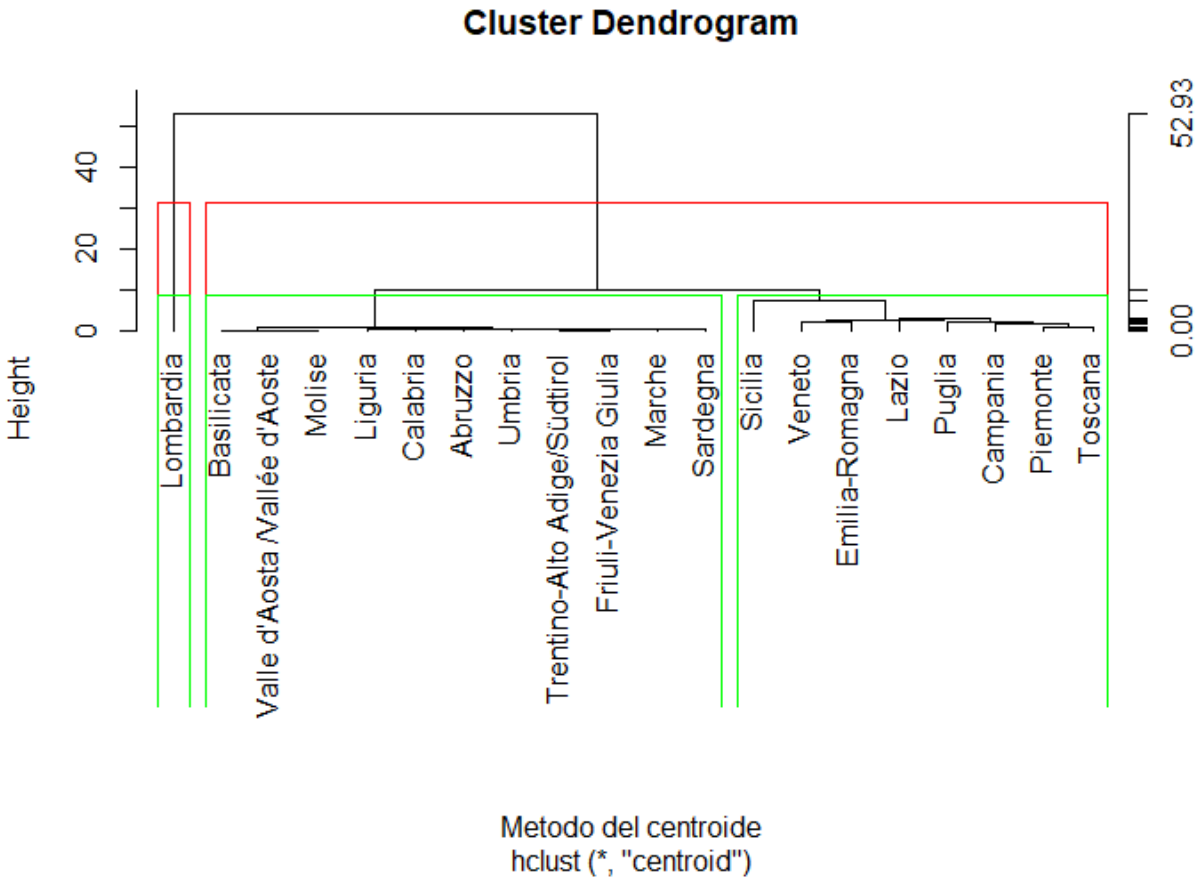
Analisi del dendrogramma

Ci proponiamo ora di analizzare il dendrogramma ottenuto con un particolare metodo gerarchico e di calcolare, fissato il numero di cluster, le misure di non omogeneità della partizione individuata. La funzione rect.hclust() permette di disegnare dei rettangoli intorno ai cluster, individuati in base all'altezza h alla quale si opera il taglio del dendrogramma oppure in base al numero k di cluster che si vogliono ottenere attraverso la funzione.

Nel nostro caso abbiamo partizionato il dendrogramma utilizzando il colore rosso per partizionare in due parti e col colore verde per partizionare in tre parti.

```
#partitions by means of rectangles
plot(hlsCentroid, hang =-1,
     xlab="Metodo del centroide")
axis(side=4,at=round(c(0,hlsCentroid$height),2))
rect.hclust(hlsCentroid,k=2,border="red")
rect.hclust(hlsCentroid,k=3,border="green")
```

E otteniamo il seguente grafico



Inserire gli individui nei cluster

Per ottenere una suddivisione degli individui in cluster in corrispondenza di un determinato livello di distanza oppure in corrispondenza di un prefissato numero di cluster, R utilizza anche la funzione cutree().

L'output della funzione cutree() è un vettore contenente numeri interi positivi associati ai cluster in cui sono stati inseriti i vari individui. Inoltre, per vedere come vengono classificati le regioni all'aumentare del numero di cluster si può considerare la funzione che produce la seguente matrice.

Nel nostro caso abbiamo analizzato il metodo il centroide.

cutree(hlsCentroid,k=1:20)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Piemonte	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Valle d'Aosta /Vallée d'Aoste	1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
Liguria	1	1	2	2	2	2	2	2	2	3	3	3	3	3	3	3	3	3	3	3
Lombardia	1	2	3	3	3	3	3	3	3	4	4	4	4	4	4	4	4	4	4	4
Trentino-Alto Adige/Südtirol	1	1	2	2	2	2	2	2	2	3	3	5	5	5	5	5	5	5	5	5
Veneto	1	1	1	1	4	4	4	4	4	5	5	6	6	6	6	6	6	6	6	6
Friuli-Venezia Giulia	1	1	2	2	2	2	2	2	2	3	3	5	5	5	5	5	5	7	7	7
Emilia-Romagna	1	1	1	1	4	4	5	5	5	6	6	7	7	7	7	7	7	8	8	8
Toscana	1	1	1	1	1	1	1	1	1	1	7	8	8	8	8	8	8	9	9	9
Umbria	1	1	2	2	2	2	2	2	2	3	3	5	5	5	5	5	9	10	10	10
Marche	1	1	2	2	2	2	2	2	2	3	3	5	9	9	9	9	10	11	11	11
Lazio	1	1	1	1	1	5	6	6	6	7	8	9	10	10	10	10	11	12	12	12
Abruzzo	1	1	2	2	2	2	2	2	2	3	3	5	5	5	5	11	12	13	13	13
Molise	1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	14
Campania	1	1	1	1	1	1	1	1	7	8	9	10	11	11	11	12	13	14	14	15
Puglia	1	1	1	1	1	1	1	7	8	9	10	11	12	12	12	13	14	15	15	16
Basilicata	1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	16	17
Calabria	1	1	2	2	2	2	2	2	2	3	3	3	3	3	13	14	15	16	17	18
Sicilia	1	1	1	4	5	6	7	8	9	10	11	12	13	13	14	15	16	17	18	19
Sardegna	1	1	2	2	2	2	2	2	2	3	3	5	9	14	15	16	17	18	19	20

Misure di sintesi associate ai cluster

In R è inoltre possibile ricavare misure di sintesi come la media campionaria, la varianza campionaria, la deviazione standard, etc, sui singoli cluster, ottenuti tagliando il dendrogramma tramite la funzione cutree(), utilizzando la funzione aggregate() e inserendo il tipo di funzione da effettuare.

Nel seguente codice come possiamo notare abbiamo tagliato in tre cluster il dendrogramma e trasformato in una lista di elementi e applicato successivamente il metodo aggregate per ottenere la media, la varianza e la deviazione standard sui gruppi.

```
#mean and median and sample standard deviation
cutT<-cutree(hlsCentroid,k=3,h=NULL)
listCut<-list(cutT)

#use method aggregate
aggregate(arrotondato,listCut,mean)
aggregate(arrotondato,listCut,var)
aggregate(arrotondato,listCut,sd)
```

Fornendo i seguenti dati:

```
  Group.1 mydf..Rifiuti.organici. mydf..Carta.e.cartone. mydf.Vetro mydf.Plastica mydf.Altro
1       1                511.1250              256.00000     143.75       98.87500        233
2       2                120.8182               61.81818      39.00       21.27273         50
3       3                1206.0000              547.00000     423.00      248.00000        838
>   aggregate(arrotondato,listCut,var)
  Group.1 mydf..Rifiuti.organici. mydf..Carta.e.cartone. mydf.Vetro mydf.Plastica mydf.Altro
1       1               40327.554               8059.714   3376.214     1286.9821   12610.86
2       2                5471.164               1165.964    508.800      170.4182    1060.40
3       3                      NA                     NA         NA            NA         NA
>   aggregate(arrotondato,listCut,sd)
  Group.1 mydf..Rifiuti.organici. mydf..Carta.e.cartone. mydf.Vetro mydf.Plastica mydf.Altro
1       1               200.81721               89.77591    58.1052      35.87453  112.29807
2       2                73.96731               34.14621    22.5566      13.05443   32.56378
3       3                      NA                     NA         NA            NA         NA
```

Il terzo cluster comprende solo la Lombardia e perciò la varianza e la deviazione standard per tipi di rifiuti ha come valore 0.

Metodi non gerarchici

L'obiettivo dei metodi non gerarchici è quello di ottenere un'unica partizione degli n individui di partenza in cluster. Gli algoritmi di tipo non gerarchico procedono, data una prima partizione, a riallocare gli individui nel gruppo con centroide più vicino, fino a che per nessun individuo si verifica che sia minima la distanza rispetto al centroide di un gruppo diverso da quello a cui esso appartiene. Il metodo più utilizzato prende il nome di k-means ed è dovuto a Hartigan e Wong.

Tale metodo richiede che il numero di cluster sia specificato a priori e fornisce in output un'unica partizione. Applicando il metodo k-means, considerando una suddivisione in tre cluster ed effettuando un'unica scelta casuale dei punti di riferimento con un numero massimo di iterazioni pari a 10 avremo:

```
K-means clustering with 3 clusters of sizes 3, 10, 7

Cluster means:
  mydf..Rifiuti.organici. mydf..Carta.e.cartone. mydf.Vetro mydf.Plastica mydf.Altro
1                20.33333                13.0000   8.666667      5.666667   8.666667
2               176.80000                94.1000  54.200000     33.100000  72.900000
3               685.00000               327.8571 204.857143    132.142857 356.714286

Clustering vector:
 [1] 3 1 2 3 2 3 2 3 3 2 2 3 2 1 3 2 1 2 2 2

Within cluster sum of squares by cluster:
[1]    416.6667  53193.9000 889412.0000
 (between_SS / total_SS =  70.9 %)

Available components:

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

Con misura di non omogeneità tra i cluster diviso la misura di non omogeneità totale pari al 70.9% quindi maggiore di 70%.

Abbiamo poi applicato il metodo str() per ottenere maggiori informazioni come la misura di non omogeneità totale, la somma delle misure di non omogeneità interne ai cluster(within) e la misura di non omogeneità tra i cluster(between):

```
List of 9
 $ cluster     : int [1:20] 3 1 2 3 2 3 2 3 3 2 ...
 $ centers     : num [1:3, 1:5] 20.3 176.8 685 13 94.1 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:3] "1" "2" "3"
  .. ..$ : chr [1:5] "mydf..Rifiuti.organici." "mydf..Carta.e.cartone." "mydf.Vetro" "mydf.Plastica" ...
 $ totss       : num 3243292
 $ withinss    : num [1:3] 417 53194 889412
 $ tot.withinss: num 943023
 $ betweenss   : num 2300270
 $ size        : int [1:3] 3 10 7
 $ iter        : int 2
 $ ifault      : int 0
 - attr(*, "class")= chr "kmeans"
```

E successivamente proviamo ad utilizzare il metodo k-means applicando il metodo scale, per scalare e standardizzare i dati, sul data frame:

```
#now we try to scale data and use kmeans
Z<-scale(arrotondato)
Z
km1<-kmeans(Z,center=3,iter.max = 10,nstart=10)
km1
str(km1)
```

Che produce:

```
K-means clustering with 3 clusters of sizes 6, 13, 1

Cluster means:
  mydf..Rifiuti.organici. mydf..Carta.e.cartone. mydf.Vetro mydf.Plastica mydf.Altro
1               0.8559880              0.8822572  0.6852747     0.7860875  0.5794353
2              -0.6108339             -0.6110594 -0.5651281    -0.5894581 -0.5317327
3               2.8049131              2.6502291  3.2350176     2.9464297  3.4359138

Clustering vector:
 [1] 1 2 2 3 2 1 2 1 1 2 2 1 2 2 1 2 2 2 2 2

Within cluster sum of squares by cluster:
[1] 5.043681 4.457470 0.000000
 (between_SS / total_SS =  90.0 %)

Available components:

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

Con misura di non omogeneità tra i cluster diviso la misura di non omogeneità totale pari al 90%.

Avente le seguenti proprietà:

```
>   str(km1)
List of 9
 $ cluster     : int [1:20] 1 2 2 3 2 1 2 1 1 2 ...
 $ centers     : num [1:3, 1:5] 0.856 -0.611 2.805 0.882 -0.611 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:3] "1" "2" "3"
  .. ..$ : chr [1:5] "mydf..Rifiuti.organici." "mydf..Carta.e.cartone." "mydf.Vetro" "mydf.Plastica" ...
 $ totss       : num 95
 $ withinss    : num [1:3] 5.04 4.46 0
 $ tot.withinss: num 9.5
 $ betweenss   : num 85.5
 $ size        : int [1:3] 6 13 1
 $ iter        : int 1
 $ ifault      : int 0
 - attr(*, "class")= chr "kmeans"
```

Confronto tra Metodi gerarchici e Metodi non gerarchici

Ora che abbiamo analizzato sia i metodi gerarchici che i metodi non gerarchici possiamo andarli a confrontare per vedere le differenze e capire quale dei due metodi offre una soluzione migliore:

Per prima cosa andiamo a validare la divisione in cluster in maniera ottimale usando le misure di non omogeneità. Per calcolare gli indici dobbiamo partire dal calcolo delle varianze delle singole colonne del dataset usando la funzione cov, per vedere quanto si discostano tra loro, nel dataset:

	Raccolta Indifferenziata	Rifiuti organici	Carta e cartone	Vetro	Plastica	Altro
Raccolta Indifferenziata	1.0000000	0.6543017	0.7179814	0.6310816	0.6405943	0.5566577
Rifiuti organici	0.6543017	1.0000000	0.9408432	0.9546156	0.9623982	0.9429913
Carta e cartone	0.7179814	0.9408432	1.0000000	0.9441357	0.9168512	0.9296047
Vetro	0.6310816	0.9546156	0.9441357	1.0000000	0.9306744	0.9461954
Plastica	0.6405943	0.9623982	0.9168512	0.9306744	1.0000000	0.9528598
Altro	0.5566577	0.9429913	0.9296047	0.9461954	0.9528598	1.0000000

Ora moltiplichiamo i singoli individui della matrice per N-1 dove N è il numero di individui del dataset, nel nostro caso sono 20 regioni quindi moltiplichiamo per 19:

	Raccolta Indifferenziata	Rifiuti organici	Carta e cartone	Vetro	Plastica	Altro
Raccolta Indifferenziata	19.00000	12.43173	13.64165	11.99055	12.17129	10.57650
Rifiuti organici	12.43173	19.00000	17.87602	18.13770	18.28557	17.91684
Carta e cartone	13.64165	17.87602	19.00000	17.93858	17.42017	17.66249
Vetro	11.99055	18.13770	17.93858	19.00000	17.68281	17.97771
Plastica	12.17129	18.28557	17.42017	17.68281	19.00000	18.10434
Altro	10.57650	17.91684	17.66249	17.97771	18.10434	19.00000

Ciò che risulta è la matrice di non omogeneità ora quindi possiamo calcolare il primo indice di stima necessario, ovvero la misura di non omogeneità totale:

$$\mathrm{tr}H_I = \sum_{r=1}^{p} h_{rr} = (n-1)\sum_{r=1}^{p} s_r^2.$$

dove P è il numero di categorie del dataset.

```
NHMS<-(19)*sum(apply(d,2,var))
NHMS
114
```

In R usiamo la funzione aggregate() che ci permette di calcolare una serie di indici, dato un dataset ed il relativo partizionamento ottenuto con la funzione cutree.

Ora mostriamo, in base al tipo di legame, i vari risultati della funzione aggregate:

```
#all type of division on cluster
  #centroid
  agvr<-aggregate(d,listCut,var)[,-1]
  agvr
Raccolta Indifferenziata Rifiuti organici Carta e cartone      Vetro   Plastica      Altro
              0.48180933       0.41459413      0.38540928 0.33888100 0.32875958 0.32636758
              0.06112056       0.05624721      0.05575548 0.05106982 0.04353332 0.02744303
                      NA               NA              NA         NA         NA         NA
  #single
  agvr1<-aggregate(d,listCut1,var)[,-1]
  agvr1
Raccolta Indifferenziata Rifiuti organici Carta e cartone      Vetro   Plastica      Altro
              0.48180933       0.41459413      0.38540928 0.33888100 0.32875958 0.32636758
              0.06112056       0.05624721      0.05575548 0.05106982 0.04353332 0.02744303
                      NA               NA              NA         NA         NA         NA
  #completed
  agvr2<-aggregate(d,listCut2,var)[,-1]
  agvr2
Raccolta Indifferenziata Rifiuti organici Carta e cartone      Vetro   Plastica      Altro
               0.3675694        0.1993109       0.2390706 0.17303314 0.18447747 0.21284397
               0.8077435        0.0740900       0.1047125 0.05833992 0.09763102 0.03668242
                      NA               NA              NA         NA         NA         NA
  #median
  agvr3<-aggregate(d,listCut3,var)[,-1]
  agvr3
Raccolta Indifferenziata Rifiuti organici Carta e cartone      Vetro   Plastica      Altro
              0.48180933       0.41459413      0.38540928 0.33888100 0.32875958 0.32636758
              0.06112056       0.05624721      0.05575548 0.05106982 0.04353332 0.02744303
                      NA               NA              NA         NA         NA         NA
  #average
  agvr4<-aggregate(d,listCut4,var)[,-1]
  agvr4
Raccolta Indifferenziata Rifiuti organici Carta e cartone      Vetro   Plastica      Altro
              0.48180933       0.41459413      0.38540928 0.33888100 0.32875958 0.32636758
              0.06112056       0.05624721      0.05575548 0.05106982 0.04353332 0.02744303
                      NA               NA              NA         NA         NA         NA
```

Come possiamo notare sono rappresentati in fila il metodo del centroide, metodo del legame singolo, metodo del legame completo, metodo della mediana, e metodo del legame medio.

I valori sono tutti uguali tranne con i valori del legame completo, per testare ciò abbiamo usato la funzione indentical().

Quindi abbiamo appreso che ci sono due gruppi da analizzare, uno che comprende i valori riguardanti tutti i gruppi tranne quello dove troviamo il metodo del legame completo e il gruppo contenente quest'ultimo.

Ora quindi andremo a calcolare, utilizzando i due gruppi creati, quanti elementi ci sono nei cluster trovati:

```
> num<-table(cutT)
>   num
cutT
 1  2  3
 8 11  1
>   num1<-table(cutT2)
>   num1
cutT2
 1  2  3
 6 13  1
```

Come possiamo notare i cluster sono differenti.

Da queste tabelle possiamo analizzare le misure di non omogeneità:

```
> #calculate again non-homogeneity matrix first
>   trh1<-(num[[1]]-1)*sum(agvr[1,])
>   trh1
[1] 15.93075
>   trh2<-(num[[2]]-1)*sum(agvr[2,])
>   trh2
[1] 2.951694
>   trh3<-(num[[3]]-1)*sum(agvr[3,])
>   trh3 #this is NA because is only Lombardia and value=
[1] NA
>   #calculate again non-homogeneity matrix second
>   trh11<-(num1[[1]]-1)*sum(agvr[1,])
>   trh11
[1] 11.3791
>   trh22<-(num1[[2]]-1)*sum(agvr[2,])
>   trh22
[1] 3.542033
>   trh33<-(num1[[3]]-1)*sum(agvr[3,])
>   trh33 #this is NA because is only Lombardia and value=0
[1] NA
```

Entrambe hanno nella terza riga il valore 0 poiché il cluster ha un unico valore e per definizione la misura di non omogeneità per un singolo elemento è uguale a 0.

La somma di non omogeneità tra i cluster prende il nome di misura di non omogeneità interna ai cluster. Se la misura di non omogeneità è più piccola della misura di non omogeneità totale significa che la scelta di divisione dei cluster è ottimale. La misura di non omogeneità tra i cluster serve per indicare una buona divisione in cluster ed è ottenuta sottraendo la misura di non omogeneità totale alla misura di non omogeneità interna.

```
>  #internal sum non non-homogeneity between cluster first(within)
>   trHS<-trh1+trh2
>   trHS
[1] 18.88244
>   #non-homogeneity between cluster first(between)
>   trHB<-NHMS-trHS
>   trHB
[1] 95.11756
>   #internal sum non non-homogeneity between cluster second(within)
>   trHS1<-trh11+trh22
>   trHS1
[1] 14.92114
>   #non-homogeneity between cluster second(between)
>   trHB1<-NHMS-trHS1
>   trHB1
[1] 99.07886
```

Nel nostro caso la divisione dei cluster è ottimale poiché la misura di non omogeneità, di entrambi i gruppi, è minore della misura di non omogeneità totale.

La relazione tra le misure di non omogeneità interna può essere vista anche in termini relativi, ottenuti effettuando il taglio e alla somma delle loro misure di non omogeneità (tr S) e alla misura di omogeneità tra i cluster (tr B):

$$1 = \frac{\operatorname{tr} S}{\operatorname{tr} T} + \frac{\operatorname{tr} B}{\operatorname{tr} T}.$$

```
>   #Relative measures first
>   #within divided total non-homogeneity
>   trHS/NHMS
[1] 0.1656354
>   #between divided total non-homogeneity
>   trHB/NHMS
[1] 0.8343646
>   #second
>   #within divided total non-homogeneity
>   trHS1/NHMS
[1] 0.1308872
>   #between divided total non-homogeneity
>   trHB1/NHMS
[1] 0.8691128
```

Consideriamo come trHS e trHS1 la somma delle misure di non omogeneità, trHB e trHB1 sono la misura di non omogeneità tra i cluster e NHMS è la misura di non omogeneità totale.

Per ogni fissata matrice X dei dati si ha che i cluster dovrebbero essere individuati in modo da minimizzare la misura di non omogeneità statistica all'interno dei cluster (within) e massimizzare la misura di non omogeneità statistica tra i gruppi (between).

Per quanto riguarda i metodi gerarchici il risultato migliore è quello ottenuto dal legame completo.

Per quanto riguarda i metodi non gerarchici abbiamo i seguenti valori:

```
> #relative internal sum non non-homogeneity between cluster
>   km1$tot.withinss/km1$totss
[1] 0.1000121
>   #relative non-homogeneity between cluster
>   km1$betweenss/km1$totss
[1] 0.8999879
```

Quindi possiamo affermare che, per quanto riguarda la suddivisione in tre cluster, i metodi non gerarchici ci consentono di avere il valore massimo dalle misure di non omogeneità tra i gruppi e il minimo valore nella misura di non omogeneità interna rispetto ai metodi non gerarchici.