Statistica e analisi dei dati

Genito e Leone

27/02/2023

Dati sull'efficienza del carburante dal 1999 al 2008 per 38 modelli di auto

Descrizione

Il dataset si chiama "mpg" (Miles per Gallon) ed è fornito con il pacchetto `ggplot2` di R, contiene informazioni sull'efficienza del carburante di diverse automobili.

Il dataset include le seguenti variabili:

- `manufacturer`: il produttore dell'automobile.
- `model`: il modello dell'automobile.
- `displ`: il volume dei cilindri dell'automobile, in litri.
- `year`: l'anno di produzione dell'automobile.
- `cyl`: il numero di cilindri dell'automobile.
- `trans`: il tipo di trasmissione dell'automobile.
- `drv`: il tipo di trazione dell'automobile (anteriore, posteriore o integrale).
- `cty`: il consumo di carburante in città, in miglia per gallone.
- `hwy`: il consumo di carburante in autostrada, in miglia per gallone.
- `fl`: il tipo di carburante utilizzato dall'automobile.
- `class`: la classe di dimensioni dell'automobile.

Il dataset "mpg" è composto da 234 osservazioni, una per ogni automobile presente nel dataset. Utilizziamo questo dataset per esplorare la relazione tra l'efficienza del carburante e diverse caratteristiche delle automobili, come il produttore, il modello, il volume dei cilindri o il tipo di trasmissione.

Esplorazione dei dati

Installazione del pacchetto ggplot2

Il primo passo è installare il pacchetto **ggplot2** utilizzando il comando **install.packages("ggplot2")** e quindi caricarlo in R utilizzando il comando **library(ggplot2)**.

Caricamento delle librerie

Import del dataset

Eseguiamo il comando **data(mpg)** per caricare il dataset "mpg" in memoria.

```
data(mpg)
```

Panoramica del dataset

Utilizziamo il comando **head(mpg)** per visualizzare le prime osservazioni del dataset. Questo ci darà un'idea di come sono strutturati i dati e delle variabili incluse nel dataset.

```
head(mpg)
```

```
## # A tibble: 6 x 11
##   manufacturer model displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 audi         a4      1.8  1999     4 auto(l5)   f        18    29 p     compa~
## 2 audi         a4      1.8  1999     4 manual(m5) f        21    29 p     compa~
## 3 audi         a4      2    2008     4 manual(m6) f        20    31 p     compa~
## 4 audi         a4      2    2008     4 auto(av)   f        21    30 p     compa~
## 5 audi         a4      2.8  1999     6 auto(l5)   f        16    26 p     compa~
## 6 audi         a4      2.8  1999     6 manual(m5) f        18    26 p     compa~
```

Il comando **head()** è utile per esplorare i dati e avere una panoramica delle osservazioni presenti nel dataset. Vogliamo visualizzare anche le ultime osservazioni del dataset, quindi utilizziamo il comando **tail(mpg)**.

```
tail(mpg)
```

```
## # A tibble: 6 x 11
##   manufacturer model  displ  year   cyl trans      drv     cty   hwy fl    class
##   <chr>        <chr>  <dbl> <int> <int> <chr>      <chr> <int> <int> <chr> <chr>
## 1 volkswagen   passat   1.8  1999     4 auto(l5)   f        18    29 p     mids~
## 2 volkswagen   passat   2    2008     4 auto(s6)   f        19    28 p     mids~
## 3 volkswagen   passat   2    2008     4 manual(m6) f        21    29 p     mids~
## 4 volkswagen   passat   2.8  1999     6 auto(l5)   f        16    26 p     mids~
## 5 volkswagen   passat   2.8  1999     6 manual(m5) f        18    26 p     mids~
## 6 volkswagen   passat   3.6  2008     6 auto(s6)   f        17    26 p     mids~
```

Pulizia dei dati

La pulizia dei dati consiste nel processo di verifica e correzione dei dati contenuti in un dataset al fine di garantirne la qualità e la validità per un'analisi statistica. La pulizia dei dati può includere diverse attività, come:

- **Rimozione dei valori mancanti**: si verifica la presenza di valori mancanti (NA) nel dataset e si sceglie come gestirli, ad esempio eliminandoli o sostituendoli con valori sostitutivi.
- **Correzione dei valori errati:** si verifica la presenza di valori errati o anomali nel dataset e si sceglie come gestirli, ad esempio eliminandoli o sostituendoli con valori sostitutivi.
- **Trasformazione delle variabili**: si verifica il tipo di dato per ogni variabile (ad esempio, numerico, carattere, data) e si effettua eventualmente la trasformazione.
- **Creazione di nuove variabili:** si creano nuove variabili derivate dalle variabili esistenti, ad esempio calcolando il rapporto tra due variabili o creando una variabile binaria a partire da una variabile categoriale.

Verifica della presenza di valori mancanti

Verifichiamo la presenza di NA nel dataset, dunque procediamo con il comando **`sum(is.na(mpg))`** per contare il numero di valori mancanti.

```
sum(is.na(mpg))
```

```
## [1] 0
```

Non sono presenti valori nulli nel data frame. In caso contrario avremmo potuto scegliere di sostituirli con la media o la moda, nel caso fossero stati valori numerici.

Agli altri step non è dedicato nessun paragrafo in quanto non serve affrontarli. Non ci sono valori errati, come sarà possibile osservare nel summary mostrato successivamente nel report. Non c'è bisogno di effettuare nessuna trasformazione di tipo in quanto ognuno è coerente e non vediamo il bisogno di creare nuove variabili.

Descrizione del dataset

Utilizziamo il comando **`str(mpg)`** per visualizzare una descrizione delle variabili del dataset. Questo ci mostrerà il tipo di dato di ogni variabile (ad esempio, numerico o carattere) e il numero di osservazioni presenti per ogni variabile.

```
str(mpg)
```

```
## tibble [234 x 11] (S3: tbl_df/tbl/data.frame)
##  $ manufacturer: chr [1:234] "audi" "audi" "audi" "audi" ...
##  $ model       : chr [1:234] "a4" "a4" "a4" "a4" ...
##  $ displ       : num [1:234] 1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
##  $ year        : int [1:234] 1999 1999 2008 2008 1999 1999 2008 1999 1999 2008 ...
##  $ cyl         : int [1:234] 4 4 4 4 6 6 6 4 4 4 ...
##  $ trans       : chr [1:234] "auto(l5)" "manual(m5)" "manual(m6)" "auto(av)" ...
##  $ drv         : chr [1:234] "f" "f" "f" "f" ...
##  $ cty         : int [1:234] 18 21 20 21 16 18 18 18 16 20 ...
##  $ hwy         : int [1:234] 29 29 31 30 26 26 27 26 25 28 ...
##  $ fl          : chr [1:234] "p" "p" "p" "p" ...
##  $ class       : chr [1:234] "compact" "compact" "compact" "compact" ...
```

Il comando **str()** è utile per ottenere informazioni sulla struttura del dataset e sulla tipologia di dati contenuti in ogni variabile. Ad esempio, dalla descrizione delle variabili del dataset "mpg" mostrata sopra, possiamo vedere che la variabile **manufacturer** è di tipo carattere (**chr**), ovvero una stringa di testo, mentre la variabile **year** è di tipo intero (**int**), ovvero un numero intero.

In R, il tipo numerico (**numeric**), come la variabile **displ**, rappresenta i numeri con la virgola mobile, ovvero i numeri con decimale. Ad esempio, 3.141592 è un numero numerico.

Il tipo intero (**integer**) rappresenta invece i numeri interi, lo sono ad esempio **year cyl** ,ovvero i numeri senza decimale. Ad esempio, 3 è un numero intero.

Panoramica delle variabili

Utilizzando il comando **summary(mpg)** otteniamo una panoramica delle statistiche descrittive per ogni variabile del dataset.

```
summary(mpg)
```

```
##  manufacturer          model               displ            year
##  Length:234         Length:234         Min.   :1.600   Min.   :1999
##  Class :character   Class :character   1st Qu.:2.400   1st Qu.:1999
##  Mode  :character   Mode  :character   Median :3.300   Median :2004
##                                        Mean   :3.472   Mean   :2004
##                                        3rd Qu.:4.600   3rd Qu.:2008
##                                        Max.   :7.000   Max.   :2008
##       cyl           trans               drv                 cty
##  Min.   :4.000   Length:234         Length:234         Min.   : 9.00
##  1st Qu.:4.000   Class :character   Class :character   1st Qu.:14.00
##  Median :6.000   Mode  :character   Mode  :character   Median :17.00
##  Mean   :5.889                                         Mean   :16.86
##  3rd Qu.:8.000                                         3rd Qu.:19.00
##  Max.   :8.000                                         Max.   :35.00
##       hwy             fl                class
##  Min.   :12.00   Length:234         Length:234
##  1st Qu.:18.00   Class :character   Class :character
##  Median :24.00   Mode  :character   Mode  :character
##  Mean   :23.44
##  3rd Qu.:27.00
##  Max.   :44.00
```

La funzione **summary()** calcola automaticamente alcune statistiche di base per ogni variabile, come ad esempio il valore minimo, il valore massimo, la media, la mediana e la deviazione standard. Se la variabile è di tipo numerico, verranno calcolate anche il quartile e il range interquartile. Se la variabile è di tipo carattere o factor, verrà conteggiato il numero di osservazioni per ogni livello o valore univoco della variabile. La funzione **summary()** è utile per ottenere una panoramica delle caratteristiche dei dati contenuti nel dataset. Ad esempio, dall'output della funzione **summary()** per il dataset "mpg" possiamo vedere il numero di osservazioni per ogni variabile (ad esempio, "Length:234" indica che ci sono 234 osservazioni per ogni variabile), il tipo di dati contenuti in ogni variabile (ad esempio, "Class :character" indica che la variabile è di tipo carattere) e alcune statistiche di base per ogni variabile (ad esempio, "Min. :1.600" indica il valore minimo della variabile, "1st Qu.:2.400" indica il primo quartile, "Median :3.300" indica la mediana e così via).

Tabelle di frequenza

Utilizziamo il comando **`table(mpg$variabile)`** per ottenere il conteggio delle osservazioni per ogni categoria di una variabile. Ad esempio, **`table(mpg$cyl)`** ti fornirà il conteggio delle osservazioni per il numero di cilindri per ogni automobile.

La funzione **`table()`** è utile anche per ottenere una panoramica delle caratteristiche di una variabile e per individuare eventuali problemi o incongruenze nei dati. Ad esempio, se una variabile contiene molti valori mancanti o valori anomali, questi verranno segnalati nella tabella di frequenze.

Manufacturer

La variabile manufacturer del dataset mpg rappresenta il produttore del veicolo. Si tratta di una variabile categorica nominale (di tipo carattere o factor), in quanto i valori possibili sono limitati e non possono essere ordinati in base alla loro posizione nella scala di valori. I valori possibili per la variabile manufacturer sono 15 categorie diverse, ad esempio "audi", "chevrolet", "dodge", "Ford", "Toyota", "Nissan" e così via.

```
df <- data.frame(Freq = c(table(mpg$manufacturer)),
Perc_freq = c(table(mpg$manufacturer)/length(mpg$manufacturer)*100))

df
```

```
##            Freq Perc_freq
## audi         18  7.692308
## chevrolet    19  8.119658
## dodge        37 15.811966
## ford         25 10.683761
## honda         9  3.846154
## hyundai      14  5.982906
## jeep          8  3.418803
## land rover    4  1.709402
## lincoln       3  1.282051
## mercury       4  1.709402
## nissan       13  5.555556
## pontiac       5  2.136752
## subaru       14  5.982906
## toyota       34 14.529915
## volkswagen   27 11.538462
```

la tabella di frequenze mostra il numero di osservazioni per ogni produttore ad esempio sono presenti 18 auto del produttore audi, 25 del produttore ford, 8 del produttore jeep e così via.

Model

La variabile model del dataset mpg rappresenta il modello del veicolo. Si tratta di una variabile categoriale, in quanto i valori possibili sono discreti e limitati.

```
df <- data.frame(Freq = c(table(mpg$model)),
                 Perc_freq = c(table(mpg$model)/length(mpg$model)*100))
df
```

```
##                         Freq Perc_freq
## 4runner 4wd                6 2.5641026
## a4                         7 2.9914530
## a4 quattro                 8 3.4188034
## a6 quattro                 3 1.2820513
## altima                     6 2.5641026
## c1500 suburban 2wd         5 2.1367521
## camry                      7 2.9914530
## camry solara               7 2.9914530
## caravan 2wd               11 4.7008547
## civic                      9 3.8461538
## corolla                    5 2.1367521
## corvette                   5 2.1367521
## dakota pickup 4wd          9 3.8461538
## durango 4wd                7 2.9914530
## expedition 2wd             3 1.2820513
## explorer 4wd               6 2.5641026
## f150 pickup 4wd            7 2.9914530
## forester awd               6 2.5641026
## grand cherokee 4wd         8 3.4188034
## grand prix                 5 2.1367521
## gti                        5 2.1367521
## impreza awd                8 3.4188034
## jetta                      9 3.8461538
## k1500 tahoe 4wd            4 1.7094017
## land cruiser wagon 4wd     2 0.8547009
## malibu                     5 2.1367521
## maxima                     3 1.2820513
## mountaineer 4wd            4 1.7094017
## mustang                    9 3.8461538
## navigator 2wd              3 1.2820513
## new beetle                 6 2.5641026
## passat                     7 2.9914530
## pathfinder 4wd             4 1.7094017
## ram 1500 pickup 4wd       10 4.2735043
## range rover                4 1.7094017
## sonata                     7 2.9914530
## tiburon                    7 2.9914530
## toyota tacoma 4wd          7 2.9914530
```

Osservando l'intera tabella notiamo che del modello caravan 2wd ne sono presenti 11 veicoli mentre del modello a4 ne sono presenti 7 e così via.

Year

La variabile year del dataset mpg rappresenta l'anno di produzione del veicolo. Si tratta di una variabile quantitativa discreta, in quanto i valori possibili sono limitati e non possono assumere valori intermedi.

```
df <- data.frame(Freq =
c(table(mpg$year)), Perc_freq = c(table(mpg$year)/length(mpg$year)*100))
df
```

```
##      Freq Perc_freq
```

```
## 1999  117        50
## 2008  117        50
```

Dalla tabella di frequenza in alto possiamo notare che il 50% dei veicoli è stato prodotto nel 1999 mentre l'altro 50% nel 2008.

Cyl

La variabile cyl del dataset mpg rappresenta il numero di cilindri del motore del veicolo. Si tratta di una variabile quantitativa discreta, in quanto i valori possibili sono limitati e non possono assumere valori intermedi.

```
df <- data.frame(Freq =
c(table(mpg$cyl)), Perc_freq = c(table(mpg$cyl)/length(mpg$cyl)*100))
df
```

```
##   Freq Perc_freq
## 4   81 34.615385
## 5    4  1.709402
## 6   79 33.760684
## 8   70 29.914530
```

I valori elencati sono il numero di cilindri del motore dei veicoli, possiamo notare che il 35% dei veicoli ha un numero di cilindri pari a 4, 70 auto circa il 30% ha un motore a 8 cilindri, 4 veicoli sono dotati di un motore a 5 cilindri ed infine il 34% delle auto ha un motore a 6 cilindri.

Trans

La variabile trans del dataset mpg rappresenta il tipo di trasmissione del veicolo. Si tratta di una variabile categorica ordinale, in quanto i valori possibili sono limitati e possono essere ordinati in base alla loro posizione nella scala di valori.

```
df <- data.frame(Freq =
c(table(mpg$trans)), Perc_freq = c(table(mpg$trans)/length(mpg$trans)*100))
df
```

```
##            Freq  Perc_freq
## auto(av)      5  2.1367521
## auto(l3)      2  0.8547009
## auto(l4)     83 35.4700855
## auto(l5)     39 16.6666667
## auto(l6)      6  2.5641026
## auto(s4)      3  1.2820513
## auto(s5)      3  1.2820513
## auto(s6)     16  6.8376068
## manual(m5)   58 24.7863248
## manual(m6)   19  8.1196581
```

I valori presenti in questa tabella rappresentano il tipo di trasmissione dei veicoli: 83 veicoli ovvero 36% del dataset hanno il tipo di trasmissione auto(l4), circa il 25% sono caratterizzati dalla trasmissione manual(m5), 39 auto invece dalla trasmissione auto(l5).

Drv

La variabile drv del dataset mpg rappresenta il tipo di trazione del veicolo. Si tratta di una variabile categorica nominale, in quanto i valori possibili sono limitati e non possono essere ordinati in base alla loro posizione nella scala di valori. I valori possibili per la variabile drv sono "f", "r" e "4", che indicano rispettivamente trazione anteriore, trazione posteriore e trazione integrale.

```
 df <- data.frame(Freq =
c(table(mpg$drv)), Perc_freq = c(table(mpg$drv)/length(mpg$drv)*100))
df
```

```
##   Freq Perc_freq
## 4  103  44.01709
## f  106  45.29915
## r   25  10.68376
```

103 auto dunque il 44% possiedono la trazione integrale, il 45% dunque 106 auto hanno la trazione anteriore infine 25 veicoli il 10% sono caratterizzati dalla trazione posteriore.

Fl

La variabile fl del dataset mpg rappresenta il tipo di carburante utilizzato dal veicolo. Si tratta di una variabile categorica nominale, in quanto i valori possibili sono limitati e non possono essere ordinati in base alla loro posizione nella scala di valori. I valori possibili per la variabile fl sono "p", "c", "e", "r" e "d", che indicano rispettivamente: carburante premium, carburante normale (etanolo 85), elettricità, carburante regolare e diesel.

```
df <- data.frame(Freq =
c(table(mpg$fl)), Perc_freq = c(table(mpg$fl)/length(mpg$fl)*100))
df
```

```
##   Freq  Perc_freq
## c    1  0.4273504
## d    5  2.1367521
## e    8  3.4188034
## p   52 22.2222222
## r  168 71.7948718
```

Dalla tabella di frequenza possiamo notare che 1 auto usa il carburante normale per viaggiare, il gasolio premium lo usano il 22% circa di veicoli dunque 52 auto, il carburante regolare è quello più comune infatti 168 auto ovvero il 72% dei veicoli ne fanno uso infine 8 auto si riforniscono con l'elettricità e 5 veicoli con il diesel.

Class

La variabile class del dataset mpg rappresenta la classe di veicolo a cui appartiene il veicolo. Si tratta di una variabile categorica nominale, in quanto i valori possibili sono limitati e non possono essere ordinati in base alla loro posizione nella scala di valori. I valori possibili per la variabile class sono 19 categorie diverse, ad esempio "suv", "pickup", "minivan" e così via.

```r
df <- data.frame(Freq =
c(table(mpg$class)), Perc_freq = c(table(mpg$class)/length(mpg$class)*100))
df
```

```
##            Freq Perc_freq
## 2seater       5  2.136752
## compact      47 20.085470
## midsize      41 17.521368
## minivan      11  4.700855
## pickup       33 14.102564
## subcompact   35 14.957265
## suv          62 26.495726
```

La classe di veicolo più frequente è suv infatti di questa classe sono presenti 62 auto dunque il 26%, 5 veicoli circa il 2% appartiene alla classe di veicolo 2seater e così via.

Visualizzazione dei dati

Visualizzare i dati attraverso grafici serve a diverse **finalità**:

1. **Facilitare la comprensione**: i grafici possono aiutare a rappresentare in modo visivo i dati e a renderli più facili da comprendere. Ad esempio, un grafico a barre può mostrare facilmente le frequenze di una variabile categoriale, mentre un grafico a linee può mostrare l'evoluzione di una variabile nel tempo.
2. **Individuare pattern e tendenze**: i grafici possono aiutare a individuare pattern e tendenze nei dati, che altrimenti potrebbero essere difficili da rilevare dall'analisi dei dati in forma tabellare.
3. **Fare confronti**: i grafici consentono di fare confronti tra diverse categorie o gruppi di dati, ad esempio confrontando le frequenze di diverse variabili.
4. **Comunicare i risultati**: i grafici possono essere utilizzati per comunicare i risultati di un'analisi a un pubblico più ampio, anche a persone che non sono esperte di statistica o di analisi dei dati. Ad esempio, un grafico può essere incluso in un report o in una presentazione, come in questo caso, per illustrare in modo chiaro e conciso i risultati di un'analisi.

Variabile manufacturer

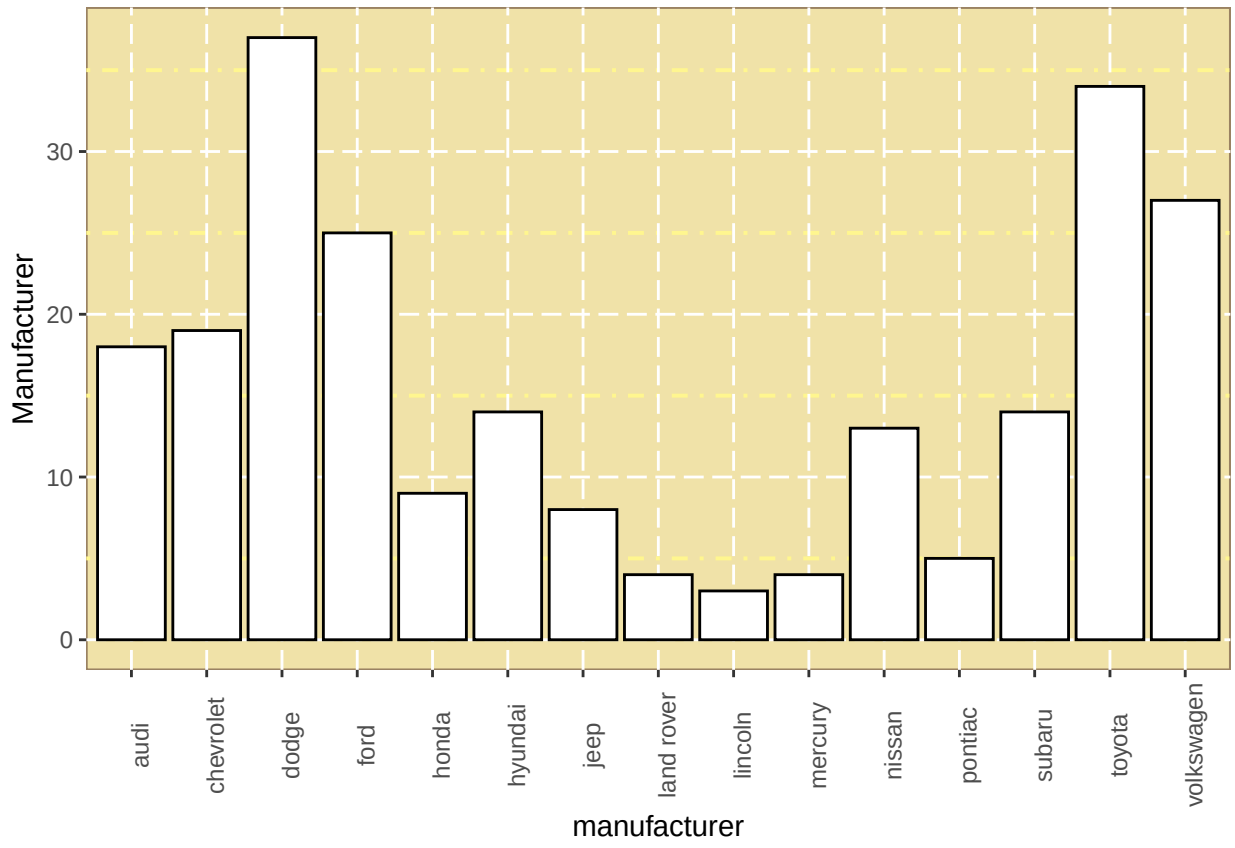
Per visualizzare la variabile "manufacturer" del dataset "mpg" con ggplot2, utilizziamo un grafico a barre e un grafico a torta. Entrambi questi grafici sono adatti per visualizzare le frequenze di una variabile di tipo carattere o factor, come la variabile "manufacturer".

Grafico a Barre

Il grafico a barre è un tipo di grafico utilizzato per visualizzare le frequenze di una variabile che può essere di tipo carattere o factor. In particolare, ogni barra del grafico rappresenta il numero di osservazioni per un determinato livello o valore univoco della variabile.

```r
p1 <- ggplot(mpg, aes(x = manufacturer)) +
      geom_bar(color = "black", fill = "white") +
  theme(axis.text.x = element_text(angle = 90)) + ylab("Manufacturer")
```

```
p1 <- p1 +
  theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
        panel.grid.major = element_line(color = '#ffffff', linetype = 'longdash'),
        panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p1
```



Il comando ha prodotto un grafico a barre che visualizza il numero di osservazioni per ogni produttore di auto presente nella variabile "manufacturer".

Il grafico a barre è utile per ottenere una panoramica delle caratteristiche di una variabile. Ad esempio, se un produttore di auto è presente in molti più casi rispetto agli altri, questo potrebbe indicare che è più popolare o diffuso.

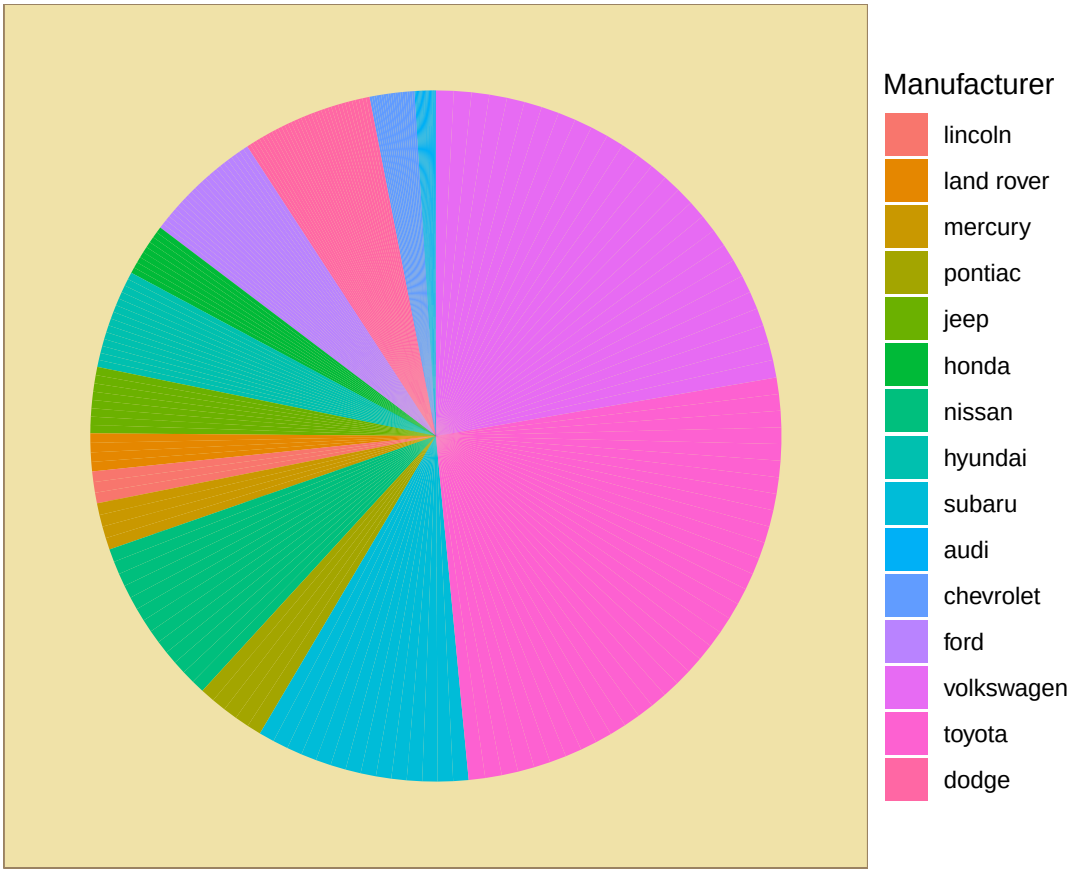
In questo caso dodge è l'osservazione più diffusa mentre lincoln quella meno frequente.

Grafico a Torta

Il grafico a torta (o grafico a settori) è un tipo di grafico utilizzato per visualizzare le frequenze di una variabile di tipo carattere o factor. In particolare, ogni settore della torta rappresenta la percentuale di osservazioni per un determinato livello o valore univoco della variabile.

```
Manufacturer <- factor(mpg$manufacturer,
                       levels = names(sort(table(mpg$manufacturer))))
```

```
p2 <- ggplot(mpg, aes(x = "", y = manufacturer, fill = Manufacturer)) +
  geom_bar(width = 1, stat = "identity") +
  coord_polar("y") + theme_void()

p2 <- p2 +
  theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
        panel.grid.major = element_line(color = '#f0e2a8', linetype = 'longdash'),
        panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p2
```



Il grafico a torta mostra la distribuzione delle osservazioni per ogni produttore di auto presente nella variabile 'manufacturer' del dataset 'mpg'.

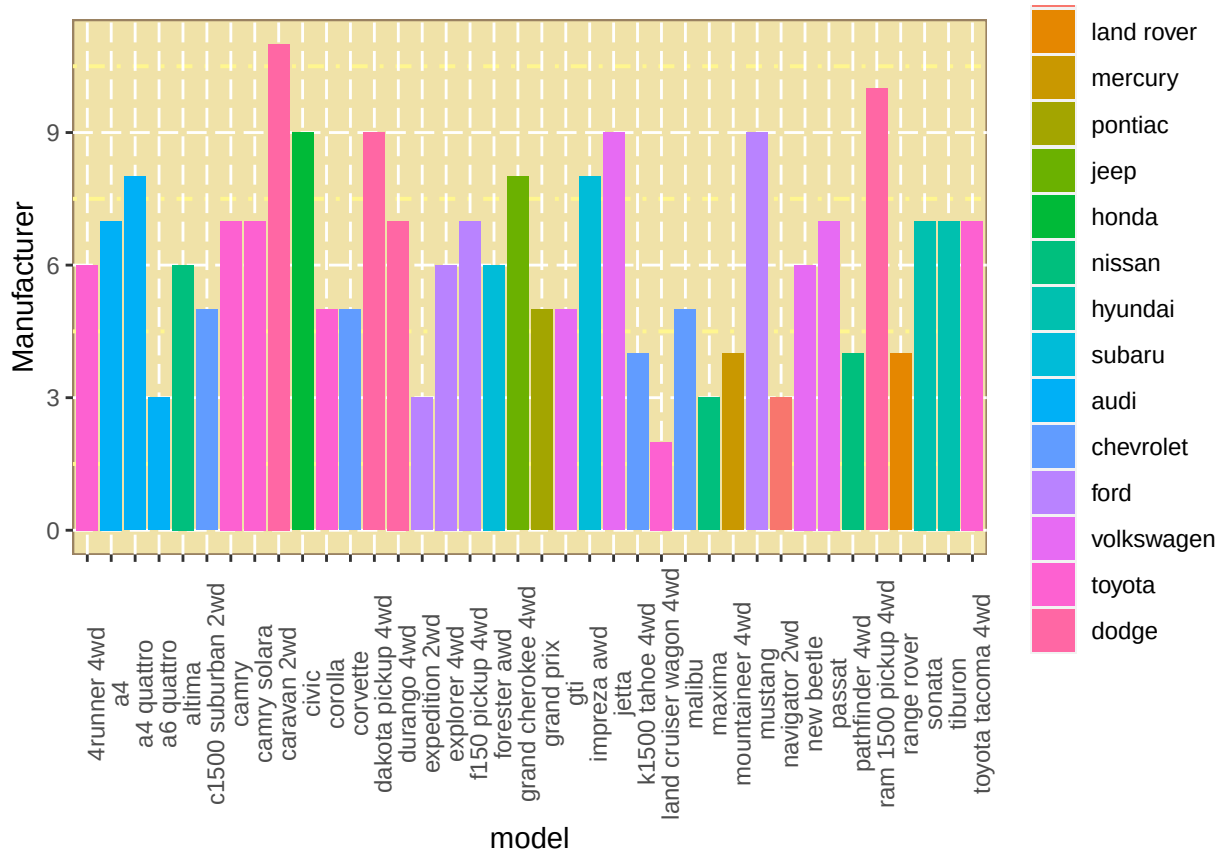
ll grafico mostra che il produttore 'toyota' è presente in circa il 15% delle osservazioni, seguito da 'dodge' con il 16% e così via...

Variabile model

Anche in questo caso essendo la variabile categoriale procediamo con la rappresentazione attraverso un grafico a barre, mentre non verrà mostrato il grafico a torta in quanto è irrilevante, la presenza di così tante categorie non permetterebbe di distinguere l'una dall'altra.

Grafico a barre

```
p3 <- ggplot(mpg, aes(x = model, fill = Manufacturer)) +
  geom_bar(position = "dodge")+ theme(axis.text.x = element_text(angle = 90))+
  ylab("Manufacturer")

p3 <- p3 +
  theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
        panel.grid.major = element_line(color = '#ffffff', linetype = 'longdash'),
        panel.grid.minor = element_line(color = '#fff68f',size = 0.7,linetype = "dotdash"))

p3
```



Il grafico a barre della variabile "model" condizionato a "manufacturer" mostra la distribuzione delle frequenze dei modelli di auto per ogni produttore. Ad esempio, possiamo vedere quanti modelli di auto di ogni produttore sono presenti nel dataset "mpg". Il grafico può essere utile per ottenere una panoramica della presenza di ogni produttore all'interno del dataset e per comparare le frequenze dei modelli di auto tra i diversi produttori. Con questo grafico sappiamo ogni modello a quale casa produttrice appartiene e conosciamo anche quanti di quei modelli sono presenti nel dataset.

Variabile displ

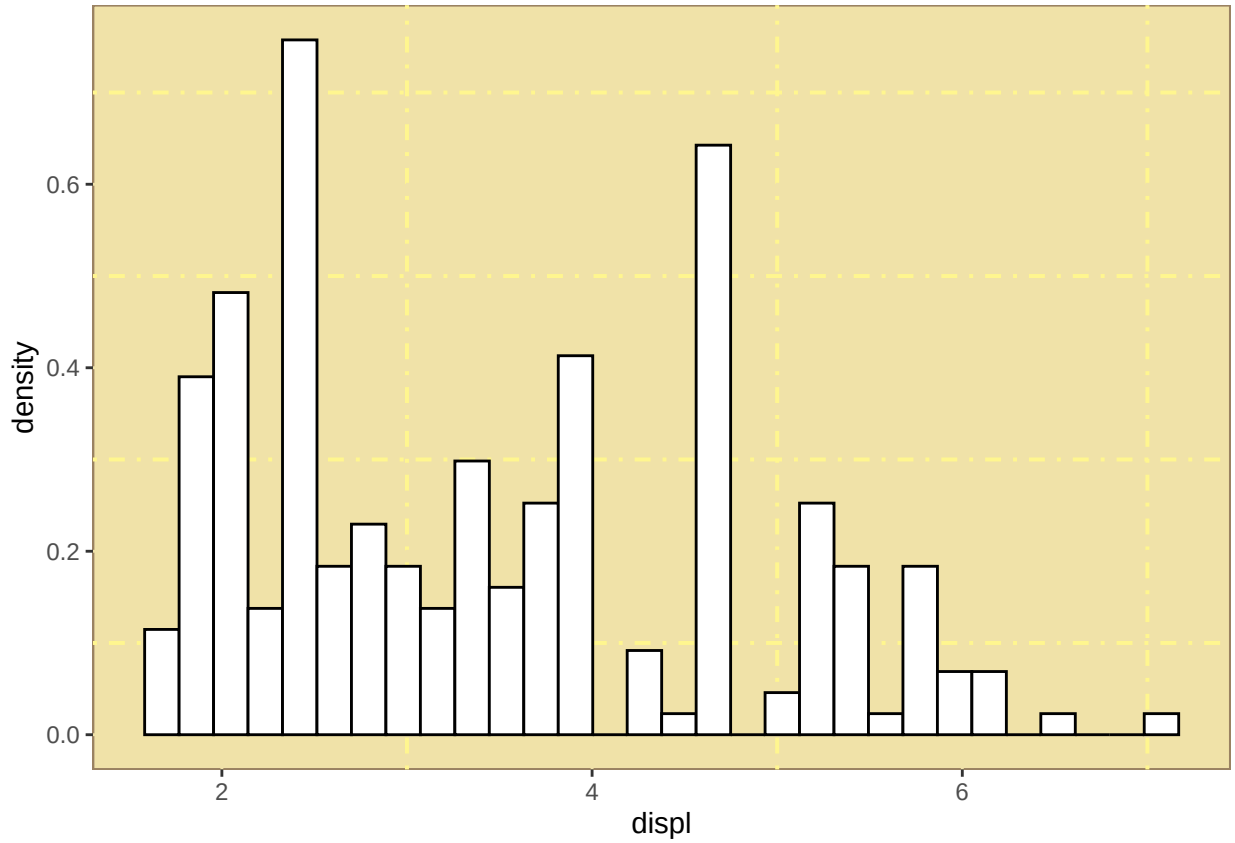
La variabile displ rappresenta il volume dei cilindri dell'automobile, in litri. Per visualizzare i dati della variabile "displ" del dataset "mpg", utilizziamo un istogramma.

Istogramma

Gli istogrammi sono formati da bins, ossia le barre verticali che dividono l'asse delle ascisse in intervalli di valori. Ogni barra rappresenta un bin, ovvero un intervallo di valori, e l'altezza della barra indica il numero di osservazioni che cadono all'interno di quell'intervallo.

L'istogramma è un grafico che mostra la distribuzione delle frequenze di una variabile numerica. Dall'istogramma possiamo vedere la forma della distribuzione, il valore medio ecc.

```
p4 <- ggplot(mpg, aes(x = displ)) +
  geom_histogram(bins = 30, aes(y = ..density..), color = "black", fill = "white")


p4 <- p4 + theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
                 panel.grid.major = element_line(color = '#f0e2a8', linetype = 'longdash'),
                 panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p4
```



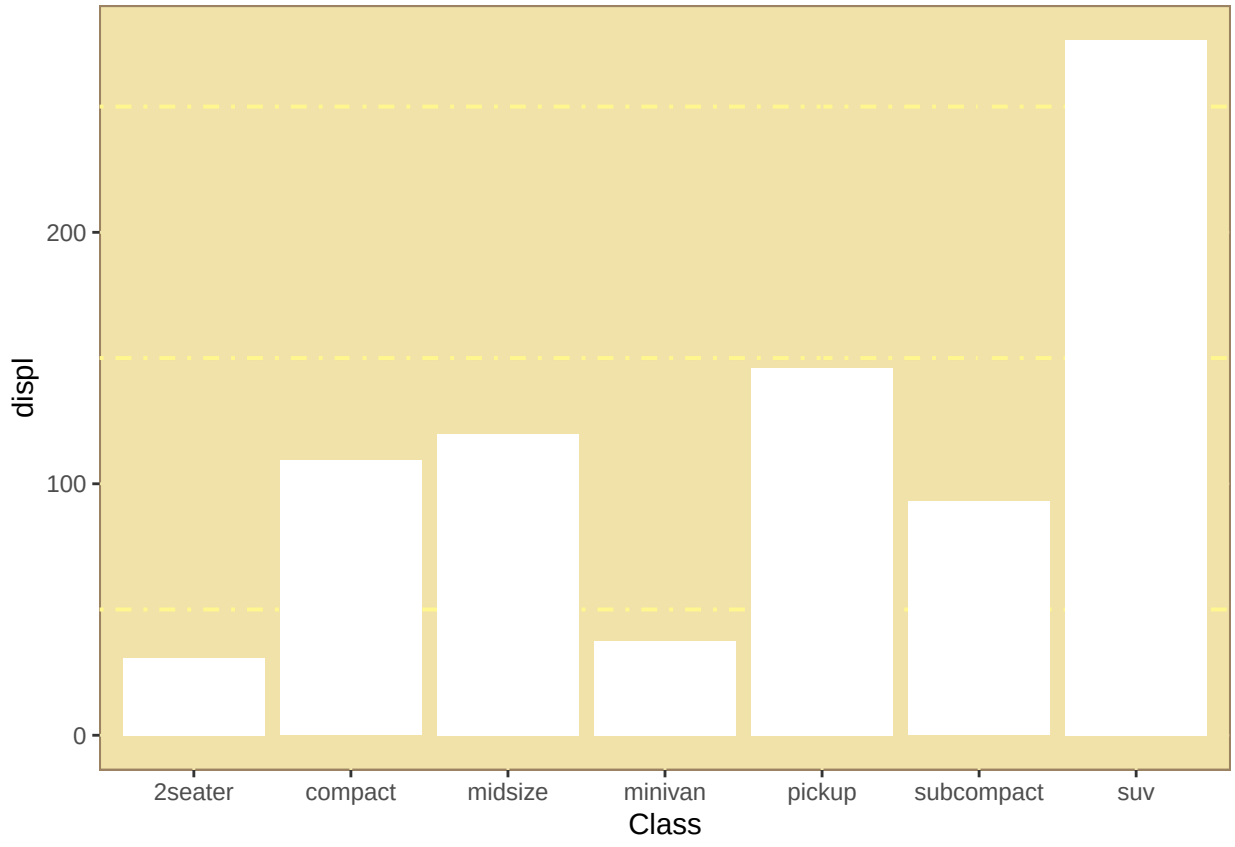
L'istogramma della variabile "displ" mostra la distribuzione dei valori della variabile. Ad esempio, possiamo vedere qual è la frequenza di ogni valore di "displ" all'interno del dataset "mpg". Il grafico visualizza i valori di "displ" come valori in ascissa e le frequenze come valori in ordinata.

Si può notare che la maggior parte dei valori di "displ" sono compresi tra 1 e 4, con un picco intorno a 2.5. Ciò indica che la distribuzione dei valori è sbilanciata a sinistra.

Grafico a barre

Vogliamo confrontare i valori della variabile "displ" tra le diverse categorie della variabile "class". Per farlo utilizziamo un grafico a barre con i valori di "displ" come valori in ascissa e le categorie di "class" come valori in ordinata:

```
macchina <- factor(mpg$class, ordered = T)

p5 <- ggplot(mpg, aes(x = macchina, y = displ)) +
  geom_bar(stat = "identity", fill = "white") +
  ylab("displ") + xlab("Class")


p5 <- p5 + theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
                 panel.grid.major = element_line(color = '#f0e2a8', linetype = 'longdash'),
                 panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p5
```

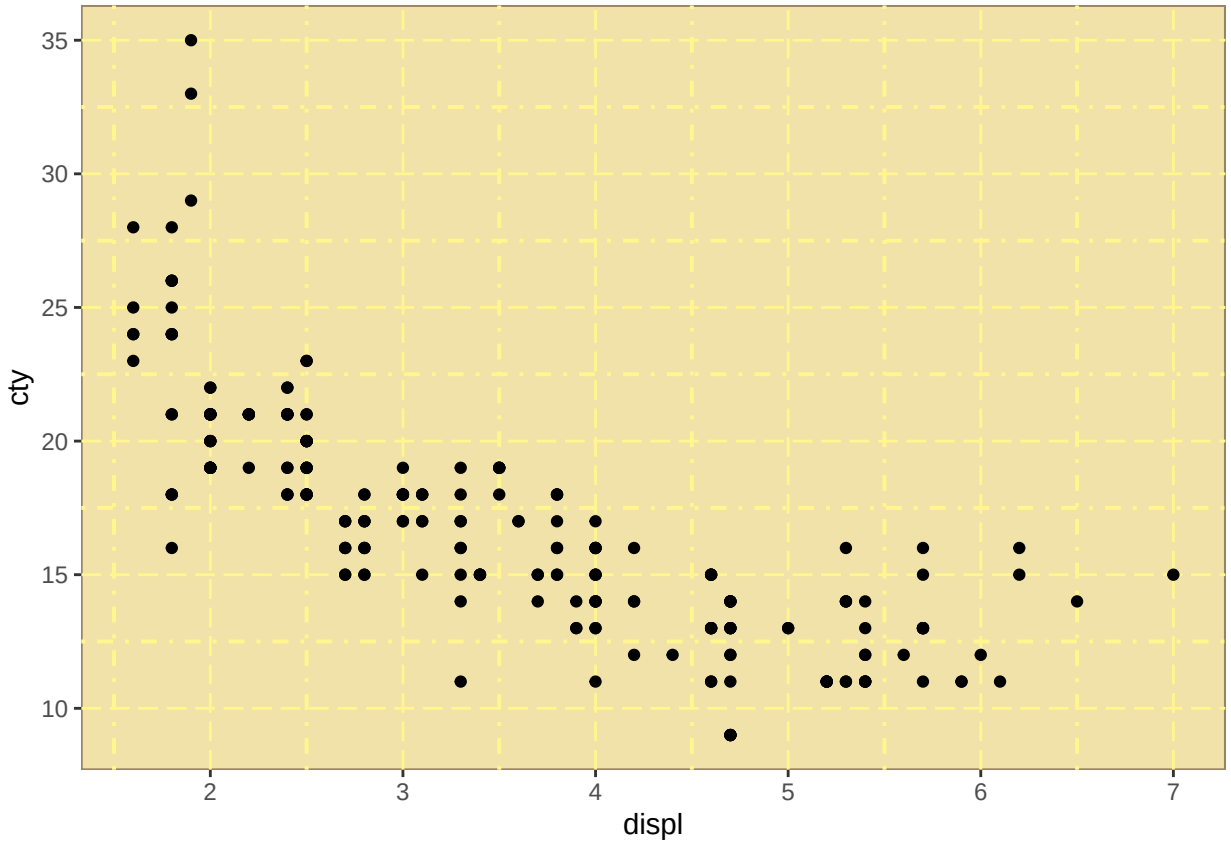


Dal grafico possiamo osservare sulle ascisse i tipi di veicoli, da notare inoltre che ogni veicolo ha un'altezza proporzionale al volume dei cilindri dell'automobile, in litri, in quanto sull'asse delle ordinate è presente la variabile displ.

Grafico a dispersione

Questo grafico mostra la relazione tra due variabili numeriche. Utilizziamo il grafico a dispersione per visualizzare come i valori di "displ" sono correlati ai valori di un'altra variabile numerica, come "cty" (l consumo di carburante in città, in miglia per gallone).

```
p6 <- ggplot(mpg, aes(x = displ, y = cty)) +
  geom_point()

p6 <- p6 + theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
                 panel.grid.major = element_line(color = '#fff68f', linetype = 'longdash'),
                 panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p6
```



Sull'asse delle ascisse ci sono le cilindrate del motore, ossia il numero di camere di combustione, sull'asse delle ordinate c'è il consumo di carburante in città, in miglia per gallone

Dal grafico vediamo che all'aumentare del volume dei cilindri ***aumenta*** il consumo di carburante in città.

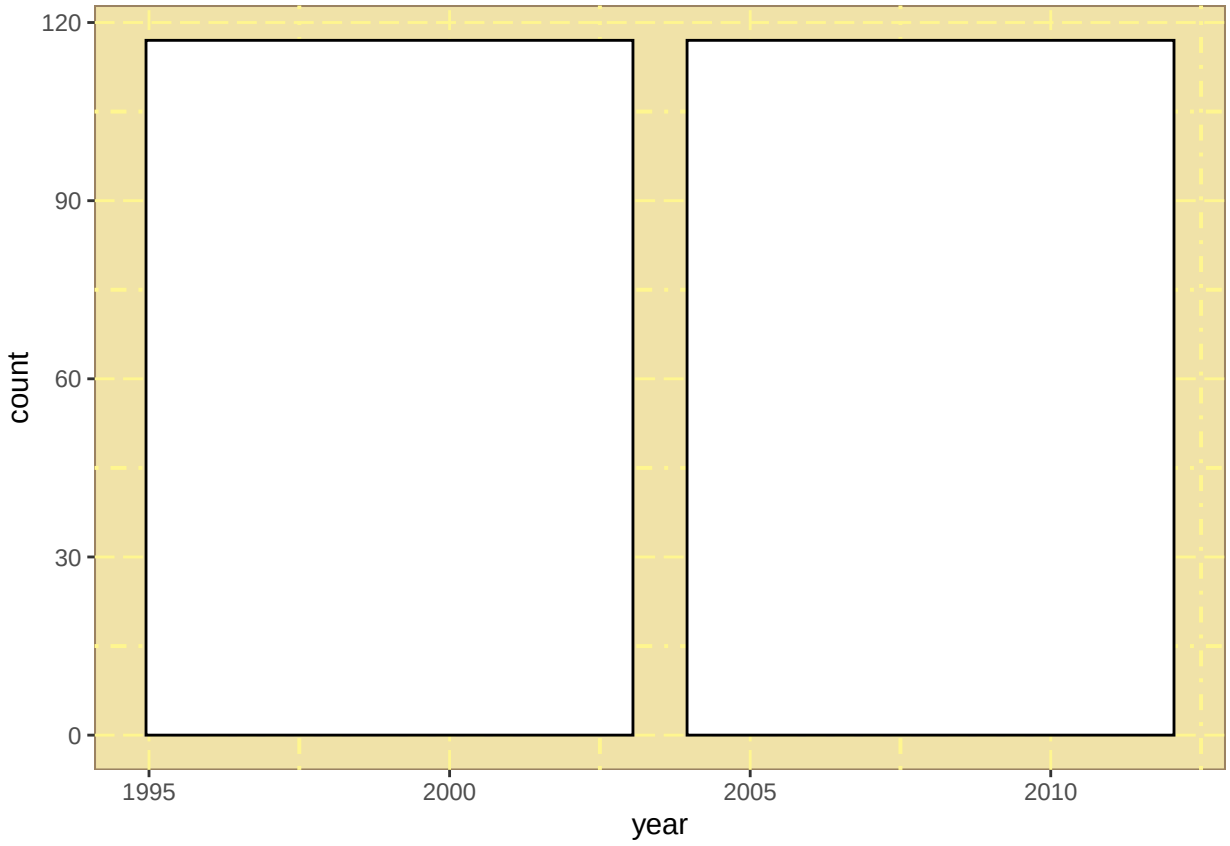
Variabile Year

Year indica l'anno di produzione dell'automobile. Le auto più vecchie del dataset sono del 1999, mentre le più recenti sono del 2008.

Barplot

Nel chunk successivo applichiamo la funzione **geom_histogram**, l'equivalente di **hist()**, per creare un istogramma della variabile **Year**. L'istogramma ci aiuterà a visualizzare la distribuzione dei valori della variabile e a identificare eventuali outlier.

```
p7 <- ggplot(mpg, aes(x = year)) +
  geom_bar(fill = "white", col = "black")

p7 <- p7 + theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
                 panel.grid.major = element_line(color = '#fff68f', linetype = 'longdash'),
                 panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p7
```

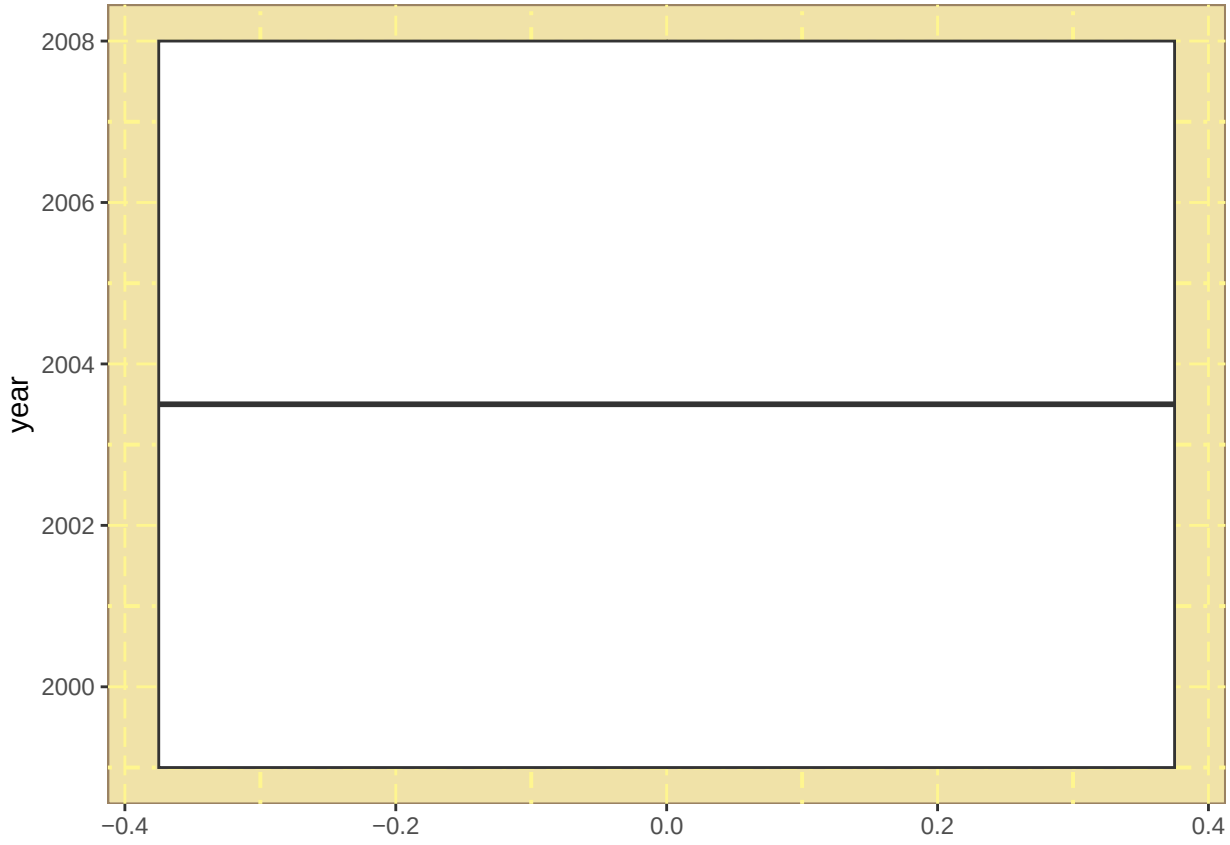


Da questo grafico capiamo che gli anni di produzione sono solo 2, il 1999 e il 2008. Quindi sono presenti due generazioni di veicoli, alcuni prodotti alla fine del 20esimo secolo e altri all'inizio del 21 secolo, discostanti da loro di 10 anni.

Boxplot

Con la funzione **`boxplot()`** creiamo un grafico a scatola a baffi della variabile **`Year`**.

```
p8 <- ggplot(mpg, aes(x = year)) +
  geom_boxplot() +
  coord_flip()

p8 <- p8 +
  theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
        panel.grid.major = element_line(color = '#fff68f', linetype = 'longdash'),
        panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p8
```



Il boxplot in questo caso appare anomalo, perché non ha i baffi, questo perché il primo quartile corrisponde al minimo e il terzo quartile corrisponde al massimo.

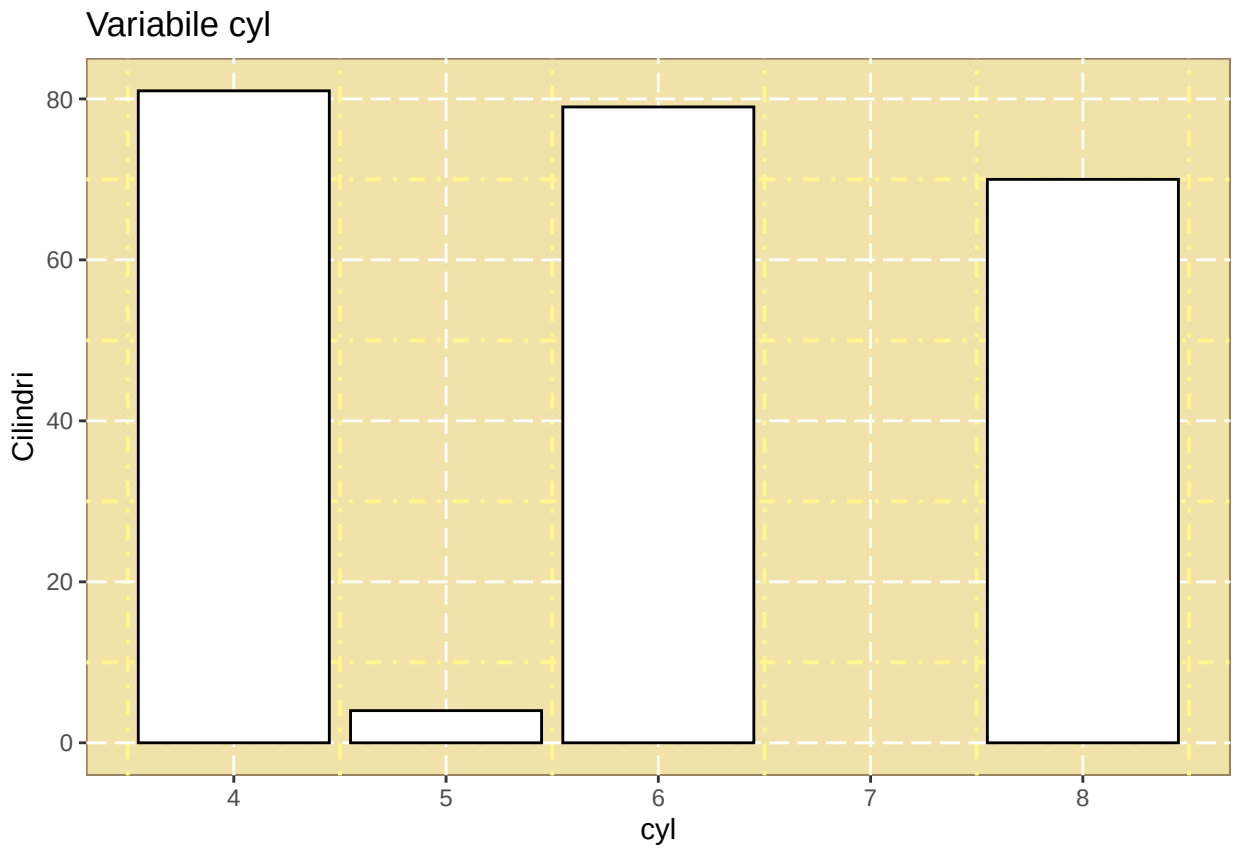
Variabile Cyl

La variabile **`cyl`** del dataset **`mpg`** rappresenta il numero di cilindri del motore dei veicoli. Si tratta di una variabile categoriale, in quanto i valori possibili sono discreti e limitati.

Barplot

Nel seguente chunk abbiamo creato un grafico a barre per visualizzare la distribuzione dei valori.

```
p9 <- ggplot(data = mpg, mapping = aes(cyl)) +
  geom_bar(stat = "count", fill= "white", col = "black") +
  ggtitle("Variabile cyl") + ylab("Cilindri")

p9 <- p9 +
  theme(panel.background = element_rect(fill = '#f0e2a8', color = "#9a8262"),
        panel.grid.major = element_line(color = '#ffffff', linetype="longdash"),
        panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p9
```



Nel grafico, "cyl" sarà mostrata sull'asse delle ascisse e il numero di osservazioni per ogni valore sarà mostrato sull'asse delle ordinate.

Il grafico a barre mostra che ci sono molte più macchine con un numero di cilindri pari a 4 rispetto a vetture con un numero di cilindri pari a 5 o 8.

Grafico a torta

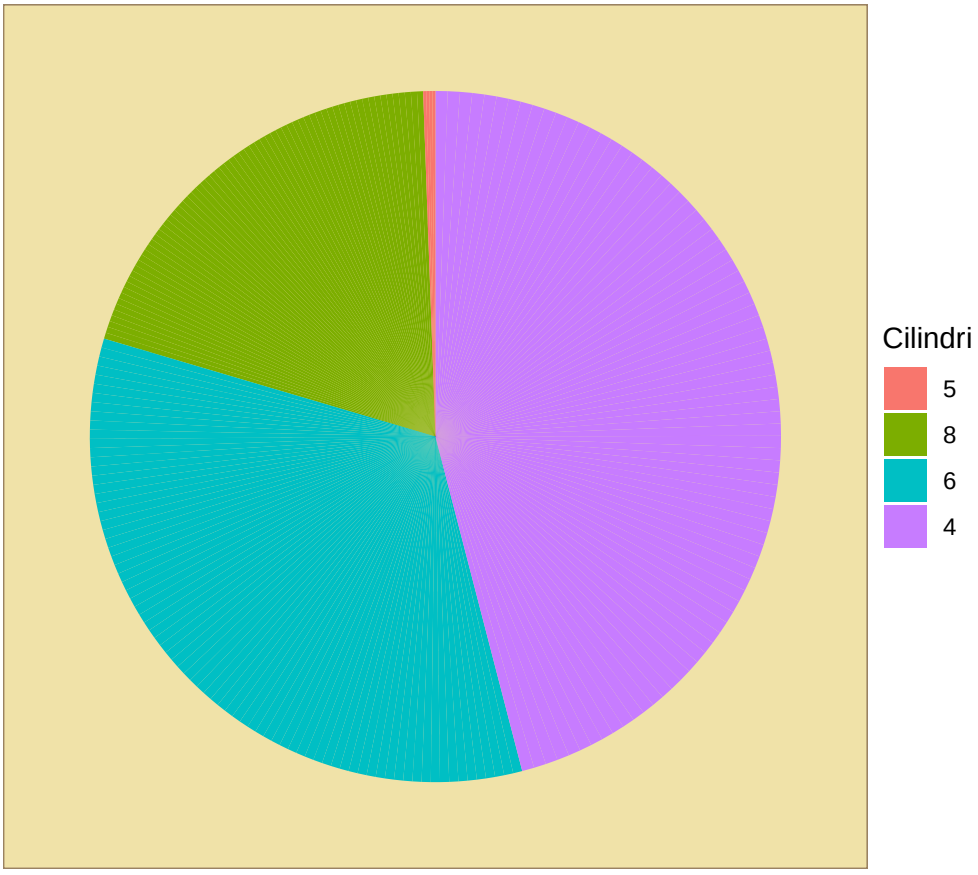
Utilizziamo un grafico a torta per mostrare le frequenze di ogni valore di "cyl" come percentuali dell'intero dataset.

Per avere un grafico a torta corretto, bisogna prima trasformare la variabile cyl in un fattore e riordinare i valori in base alla frequenza,

```
Cilindri <- factor(mpg$cyl, levels = names(sort(table(mpg$cyl))))
```

In questo modo, i valori di "Cilindri" verranno ordinati in base alla frequenza, dal valore più frequente al meno frequente

```
p10 <- ggplot(mpg, aes(x = "", y = Cilindri, fill = Cilindri)) +
  geom_bar(width = 1,stat = "identity") +
  coord_polar("y") + theme_void()

p10 <- p10 + theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
      panel.grid.major = element_line(color = '#f0e2a8', linetype = 'longdash'),
      panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p10
```



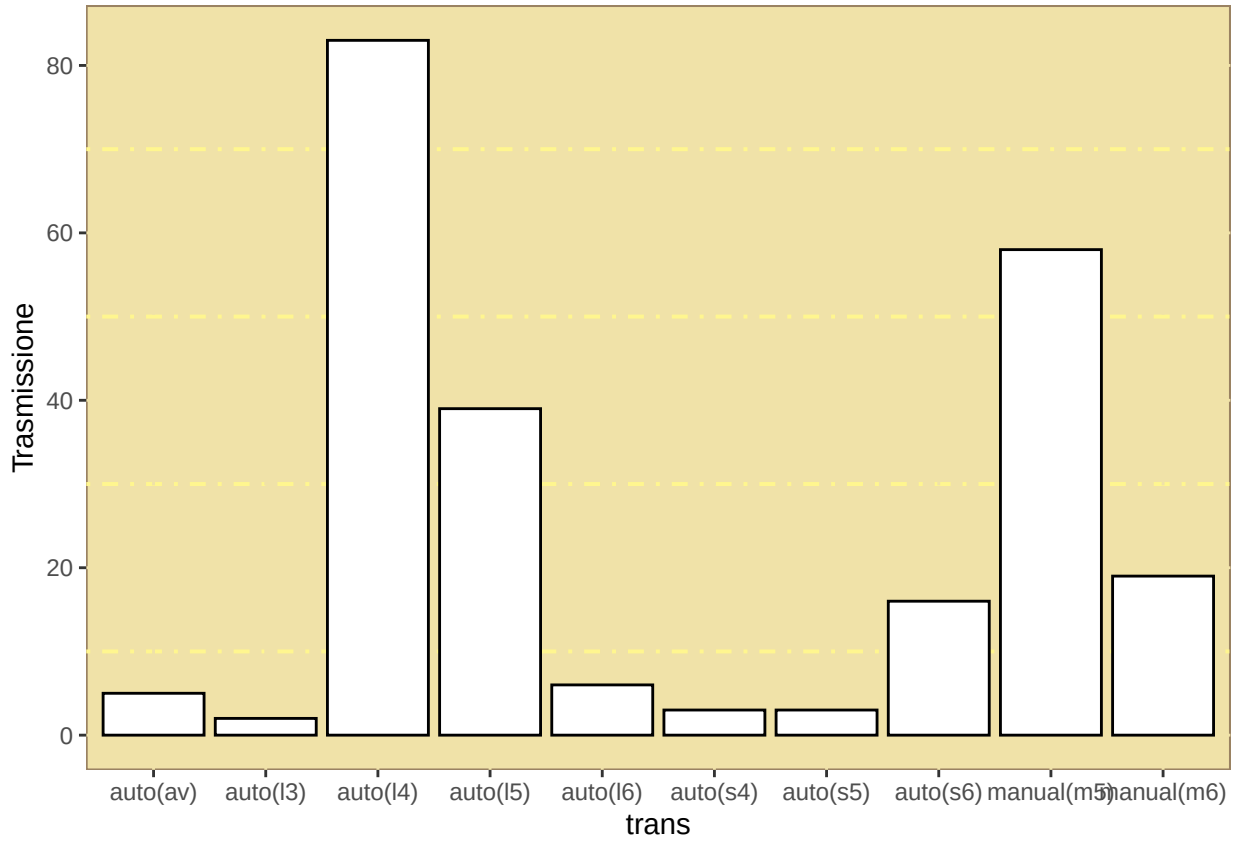
Il grafico a torta ci da una conferma di quello che avevamo già visto nel grafico a barre, la presenza di veicoli a 4 cilindri è superiore alle altre. In sostanza il dataset è formato quasi interamente da veicoli a 4 e 6 cilindri.

Variabile trans

La variabile **trans** del dataset **mpg** rappresenta il tipo di trasmissione dei veicoli. Si tratta di una variabile categoriale, in quanto i valori possibili sono discreti e limitati.

Barplot

```
p11 <- ggplot(mpg, mapping = aes(trans)) +
  geom_bar(fill = "white", col = "black") +
  ylab("Trasmissione")

p11 <- p11 +
  theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
        panel.grid.major = element_line(color = '#f0e2a8', linetype = 'longdash'),
        panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p11
```

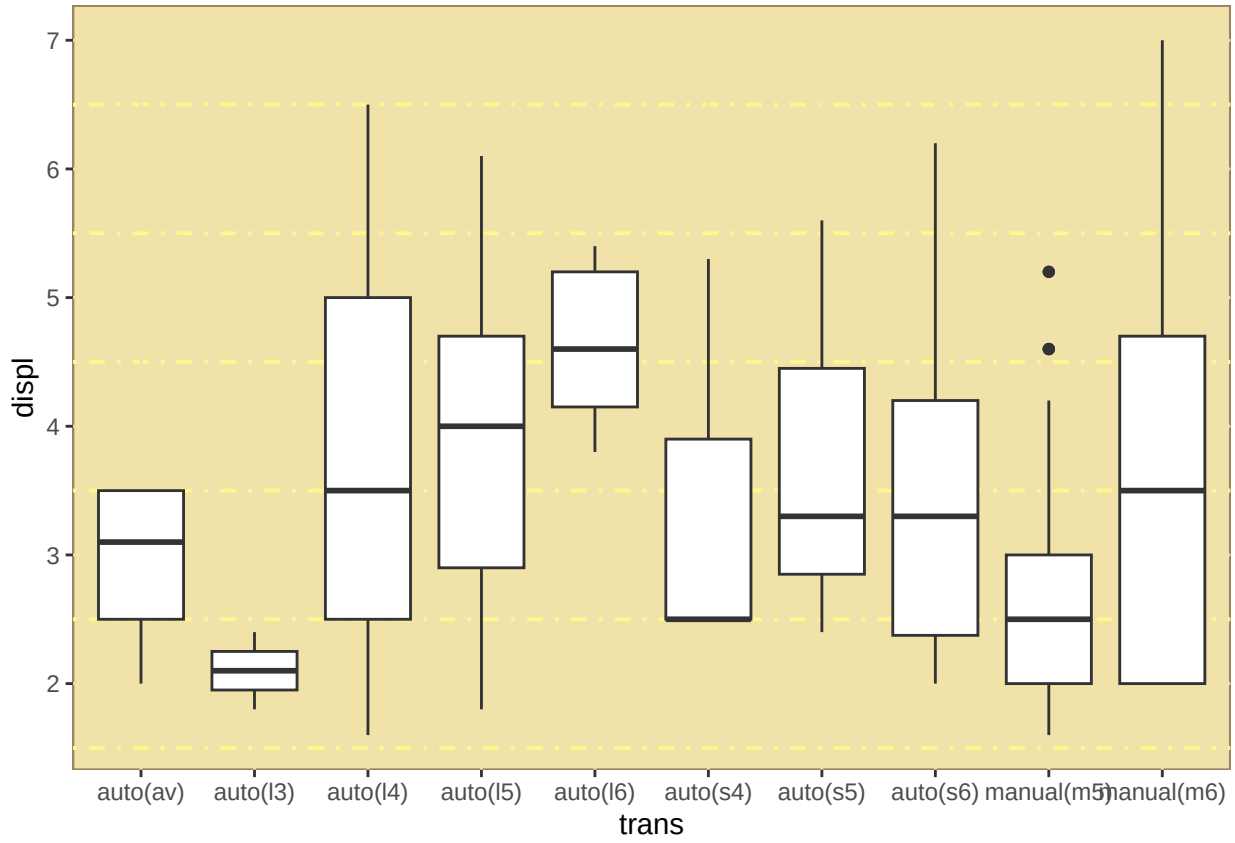


Dall'istogramma possiamo notare che ci sono molti veicoli con il tipo di trasmissione *auto(l4)* e il tipo di trasmissione *manual(m5)* .

Boxplot

```
p12 <- ggplot(mpg, aes(x = trans, y = displ)) +
  geom_boxplot()

p12 <- p12 +
```

```
  theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
        panel.grid.major = element_line(color = '#f0e2a8', linetype = 'longdash'),
        panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p12
```



Il boxplot è un grafico utile per visualizzare la distribuzione di una variabile continua stratificata per una variabile discreta, poiché permette di individuare facilmente le differenze tra le distribuzioni delle diverse categorie. Ad esempio, in questo caso, il grafico mostra come la distribuzione dei valori della variabile **displ** cambi in base al tipo di trasmissione del veicolo. Possiamo notare ad esempio che la mediana dei valori della variabile **displ** per il tipo di trasmissione "automatic" è leggermente più alta rispetto al tipo di trasmissione "manual". Oppure, che ci sono più outliers per il tipo di trasmissione "automatic" rispetto al tipo di trasmissione "manual".

Adesso ci soffermiamo a capire quei due outlier presenti nel boxplot *manual(m5)* a cosa si riferiscono:

```
outliers <- mpg %>%
  filter(trans == "manual(m5)") %>%
  filter(displ > 4.5 & displ < 5.5)
outliers
```

```
## # A tibble: 5 x 11
##   manufacturer model       displ  year   cyl trans drv     cty   hwy fl    class
##   <chr>        <chr>       <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <chr>
## 1 dodge        dakota pic~   5.2  1999     8 manu~ 4        11    17 r     pick~
```

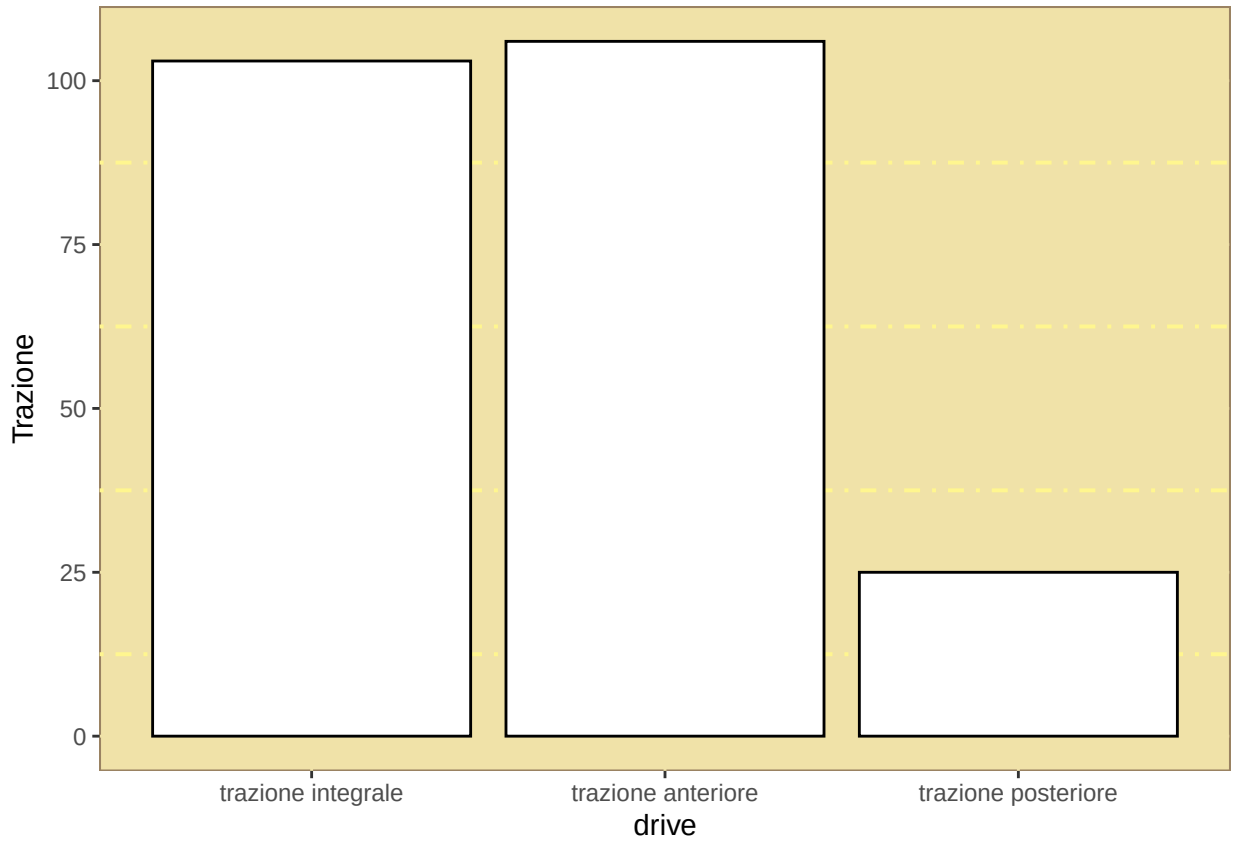
```
## 2 dodge        ram 1500 p~   5.2  1999     8 manu~ 4        11    16 r     pick~
## 3 ford         f150 picku~   4.6  1999     8 manu~ 4        13    16 r     pick~
## 4 ford         mustang       4.6  1999     8 manu~ r        15    22 r     subc~
## 5 ford         mustang       4.6  2008     8 manu~ r        15    23 r     subc~
```

Variabile Drv

La variabile Drv rappresenta il tipo di trazione dell'automobile (anteriore, posteriore o integrale).

Barplot

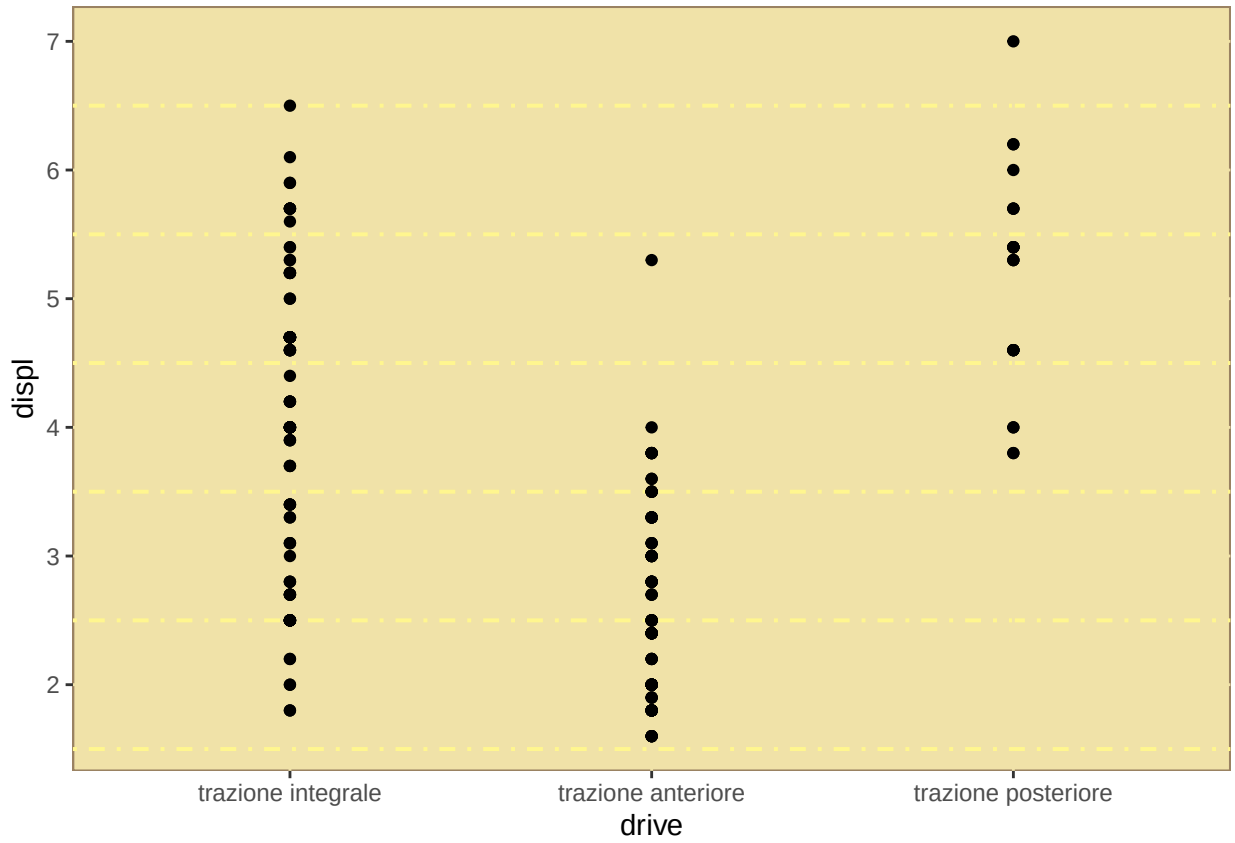
```
drive <- factor(mpg$drv, levels = c("4", "f", "r"),
        labels = c("trazione integrale",
                   "trazione anteriore", "trazione posteriore"))

p13 <- ggplot(mapping = aes(drive)) +
  geom_bar(fill = "white", col = "black") +
  ylab("Trazione")

p13 <- p13 +
  theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
        panel.grid.major = element_line(color = '#f0e2a8', linetype = 'longdash'),
        panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p13
```



La maggior parte dei veicoli possiede la trazione integrale e la trazione anteriore, sono pochi i veicoli con la trazione posteriore.

Scatterplot

```
p14 <- ggplot(mpg, aes(x = drive, y = displ)) +
  geom_point()

p14 <- p14 +
  theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
        panel.grid.major = element_line(color = '#f0e2a8', linetype = 'longdash'),
        panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p14
```

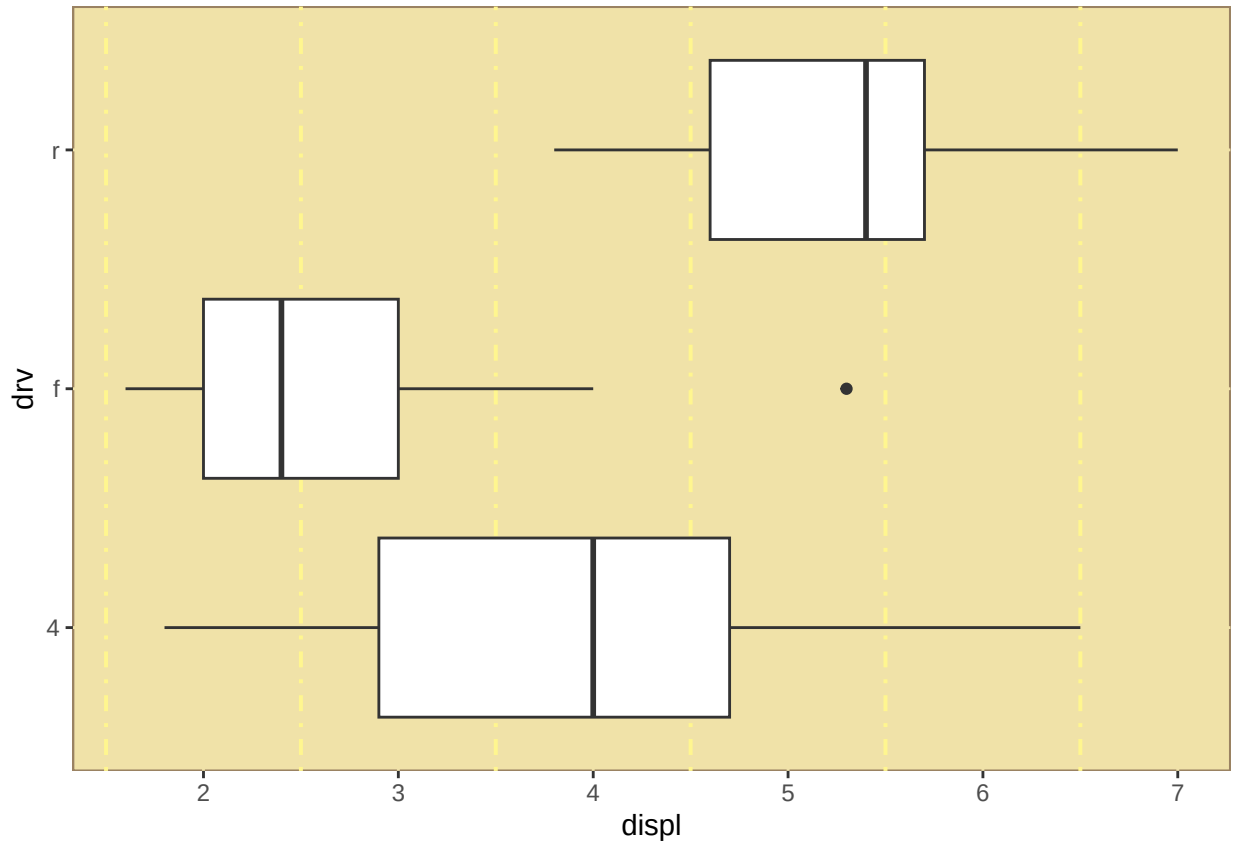


Dallo scatterplot possiamo osservare che i veicoli con la trazione integrale, possono avere qualsiasi volume dei cilindri (in litri). I veicoli con la trazione anteriore invece possiedono la cilindrata del motore bassa a differenza delle auto con la trazione posteriore che sono caratterizzate da una cilindrata alta.

Giungiamo alla medesima conclusione osservando il boxplot in basso.

Boxplot

```
p15 <- ggplot(mpg, aes(x = drv, y = displ)) +
  geom_boxplot() +
  coord_flip()

p15 <- p15 +
  theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
        panel.grid.major = element_line(color = '#f0e2a8', linetype = 'longdash'),
        panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p15
```



La centralità delle scatole, espressa dalla mediana vediamo che non è propriamente al centro, ciò sta a significare che non c'è simmetria nei dati, piuttosto la distribuzione è asimmetrica, come nel caso del primo boxplot che è fortemente asimmetrica a sinistra. Dall'ultimo boxplot notiamo dalla lunghezza dei baffi che c'è molta dispersione nei dati. Mentre nel secondo boxplot vediamo che c'è un outliers, ossia un valore anomalo che va molto al di fuori dei baffi.

Adesso ci soffermiamo a capire quell'outlier presente nel boxplot a cosa si riferisce:

```
outlier <- mpg %>%
  filter(drv == "f")%>%
  filter(displ > 5 & displ < 6)
outlier
```

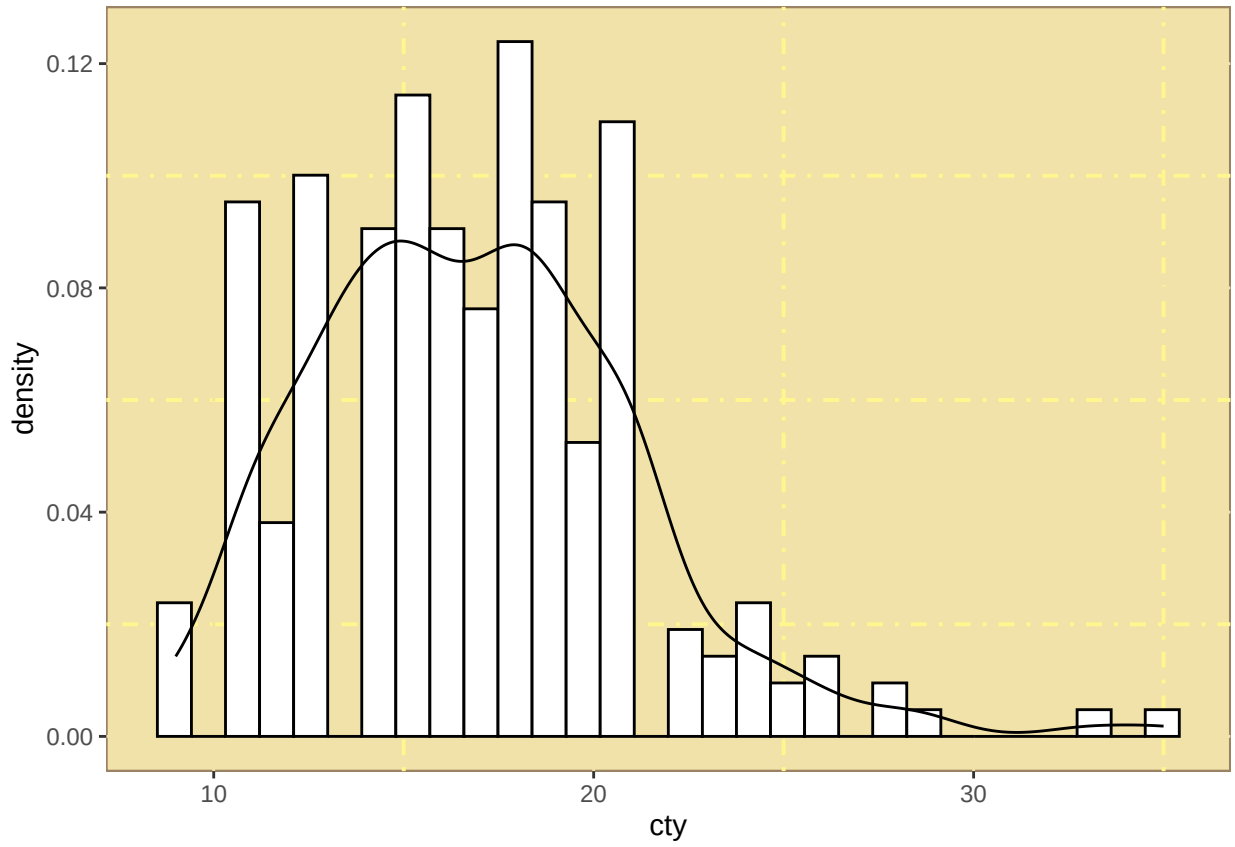
```
## # A tibble: 1 x 11
##   manufacturer model      displ  year   cyl trans  drv     cty   hwy fl    class
##   <chr>        <chr>      <dbl> <int> <int> <chr>  <chr> <int> <int> <chr> <chr>
## 1 pontiac      grand prix   5.3  2008     8 auto(~ f        16    25 p     mids~
```

Variabile Cty

La variabile Cty rappresenta il consumo di carburante in città, in miglia per gallone.

Istogramma e kernel density

```
p16 <- ggplot(data = mpg, mapping = aes(cty,..density..)) +
  geom_histogram(bins = 30, fill = "white", col = "black")+
  geom_density()

p16 <- p16 +
  theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
        panel.grid.major = element_line(color = '#f0e2a8', linetype = 'longdash'),
        panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p16
```



Dall'istogramma possiamo osservare come si distribuisce la variabile, è evidente che la maggior parte dei veicoli percorrono tra i 10 e i 20 miglia con un gallone.

Indici di sintesi

Alcuni indici di sintesi, detti anche statistiche, utili a descrivere dei dati numerici, sono media, mediana, moda, varianza, deviazione standard e coefficiente di variazione. La media, la mediana e la moda sono misure di centralità, mentre la varianza e la deviazione standard misurano la dispersione dei dati.

Media campionaria

La media campionaria è la media aritmetica di questi valori.

Definizione: Si definisce media campionaria e si denota con $\overline{x}$, la quantità:

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

La media campionaria è una media pesata dei valori distinti assunti dai dati. Ogni valore distinto usa come peso la sua frequenza relativa, ovvero la frazione dei dati uguale a tale valore numerico.

```
mean(mpg$cty)
```

```
## [1] 16.85897
```

La media campionaria utilizza tutti i dati ed è influenzata in maniera sensibile da valori eccezionalmente alti o bassi.

Per ogni valore x_i si definisce lo scarto dalla media campionaria la quantità:

$$s_i = x_i - \overline{x} \;\; con \;\; i = (1, 2, ..., n)$$

che indica il grado di scostamento del singolo valore x_i dalla media campionaria $\overline{x}$. Si nota immediatamente che la somma algebrica degli scarti dalla media campionaria è sempre nulla. Infatti, risulta:

$$\sum_{i=1}^{n} s_i = \sum_{i=1}^{n}(x_i - \overline{x}) = \sum_{i=1}^{n} x_i - n\overline{x} = n(\overline{x} - \overline{x}) = 0$$

Mediana Campionaria

Assegnato un insieme di dati di ampiezza n, lo si ordini in ordine crescente (dal valore più piccolo al valore più grande). Se n è dispari, si definisce mediana campionaria il valore che è in posizione $(n+1)/2$, mentre se n è pari la mediana campionaria è invece definita come la media aritmetica dei valori che occupano le posizioni $n/2$ e $n/2+1$.

```
median(mpg$cty)
```

```
## [1] 17
```

La mediana campionaria dipende solo da uno o da due valori centrali dei dati e non risente dei valori estremi. Inoltre, l'uso della mediana come indice per descrivere le caratteristiche dei dati ha lo svantaggio di dover prima riordinare i dati in ordine crescente, il che non è richiesto per il calcolo della media.

Moda campionaria

La moda campionaria di un insieme di dati, se esiste, è la modalità a cui è associata la frequenza (assoluta o relativa) più elevata. Se esistono più modalità con frequenza massima, ciascuna di esse è detto valore modale.

La distribuzione è detta bimodale se presenta due mode.

La moda ha alcune proprietà importanti tra cui:

- è possibile identificare la moda per qualsiasi tipo di variabile, quindi anche nelle variabili qualitative ordinabili e non ordinabili;
- indica sempre un valore realmente osservato nel campione;
- non è influenzata dai valori estremi;
- nel caso di distribuzioni di frequenze molto asimmetriche, la moda è il miglior indice per descrivere la tendenza centrale di un campione.

Non esiste in R una funzione per estrarre la moda o la classe modale di una distribuzione di dati poiché è facilmente ricavabile osservando il grafico delle frequenze assolute o l'istogramma delle frequenze relative.

Quartili, decili, percentili

- I *quartili,* come dice la parola stessa, si ottengono dividendo l'insieme di dati ordinati in 4 parti uguali. (quantili di ordine 1/4, 1/2, 3/4)

```
quantile(mpg$cty, probs = c(0.25,0.50,0.75))
```

```
## 25% 50% 75%
##  14  17  19
```

- I *decili,* infine, dividono la distribuzione in 10 parti uguali (quantili di ordine 1/10)

```
quantile(mpg$cty, probs=seq (0, 1, 0.1))
```

```
##   0%  10%  20%  30%  40%  50%  60%  70%  80%  90% 100%
##    9   11   13   14   15   17   18   19   20   21   35
```

- I *percentili*, invece, dividendo la distribuzione in 100 parti (quantili di ordine 1/100)

```
quantile(mpg$cty, probs=seq (0, 1, 0.01))
```

```
##    0%    1%    2%    3%    4%    5%    6%    7%    8%    9%   10%   11%   12%
##  9.00  9.00 10.32 11.00 11.00 11.00 11.00 11.00 11.00 11.00 11.00 12.00 12.00
##   13%   14%   15%   16%   17%   18%   19%   20%   21%   22%   23%   24%   25%
## 12.00 12.62 13.00 13.00 13.00 13.00 13.00 13.00 13.00 13.00 13.59 14.00 14.00
##   26%   27%   28%   29%   30%   31%   32%   33%   34%   35%   36%   37%   38%
## 14.00 14.00 14.00 14.00 14.00 14.23 15.00 15.00 15.00 15.00 15.00 15.00 15.00
##   39%   40%   41%   42%   43%   44%   45%   46%   47%   48%   49%   50%   51%
## 15.00 15.00 15.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 16.00 17.00 17.00
##   52%   53%   54%   55%   56%   57%   58%   59%   60%   61%   62%   63%   64%
## 17.00 17.00 17.00 17.00 17.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00 18.00
##   65%   66%   67%   68%   69%   70%   71%   72%   73%   74%   75%   76%   77%
## 18.00 18.00 18.00 19.00 19.00 19.00 19.00 19.00 19.00 19.00 19.00 19.08 20.00
##   78%   79%   80%   81%   82%   83%   84%   85%   86%   87%   88%   89%   90%
## 20.00 20.00 20.00 20.73 21.00 21.00 21.00 21.00 21.00 21.00 21.00 21.00 21.00
##   91%   92%   93%   94%   95%   96%   97%   98%   99%  100%
## 22.00 22.00 23.00 24.00 24.00 24.68 26.00 26.68 28.67 35.00
```

Varianza e deviazione standard

Assegnato un insieme di dati numerici $x_1, x_2, ..., x_n$, si definisce varianza campionaria, e si denota con s^2, la quantità:

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

con

$$(n = 2, 3, ...)$$

```
var(mpg$cty)
```

```
## [1] 18.11307
```

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

```
sd(mpg$cty)
```

```
## [1] 4.255946
```

Varianza campionaria e deviazione standard campionaria sono detti indici di dispersione o indici di variabilità poiché misurano la dispersione dei dati intorno alla media.

I valori della varianza campionaria e della deviazione standard campionaria dipendono dall'unità di misura dei dati.

Coefficiente di variazione

Per confrontare le variazioni esistenti tra diversi campioni di dati è utile introdurre coefficiente di variazione.

Assegnato un insieme di dati numerici $x_1, x_2, ..., x_n$, si definisce coefficiente di variazione il rapporto tra la deviazione standard campionaria e il modulo della media campionaria, ossia:

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

Si nota che il coefficiente di variazione è un numero puro, ossia è un indice indice *adimensionale* che non dipende dall'unità di misura utilizzata.

Il coefficiente di variazione è utilizzato quando è necessario confrontare tra loro le dispersioni (variabilità) di insiemi di dati espressi in differenti unità di misura oppure insiemi di dati aventi differenti range di variazione (il range di variazione è dato dalla differenza tra il massimo e il minimo dei dati).

In R si implementa nel seguente modo:

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

cv(mpg$cty)
```

```
## [1] 0.2524439
```

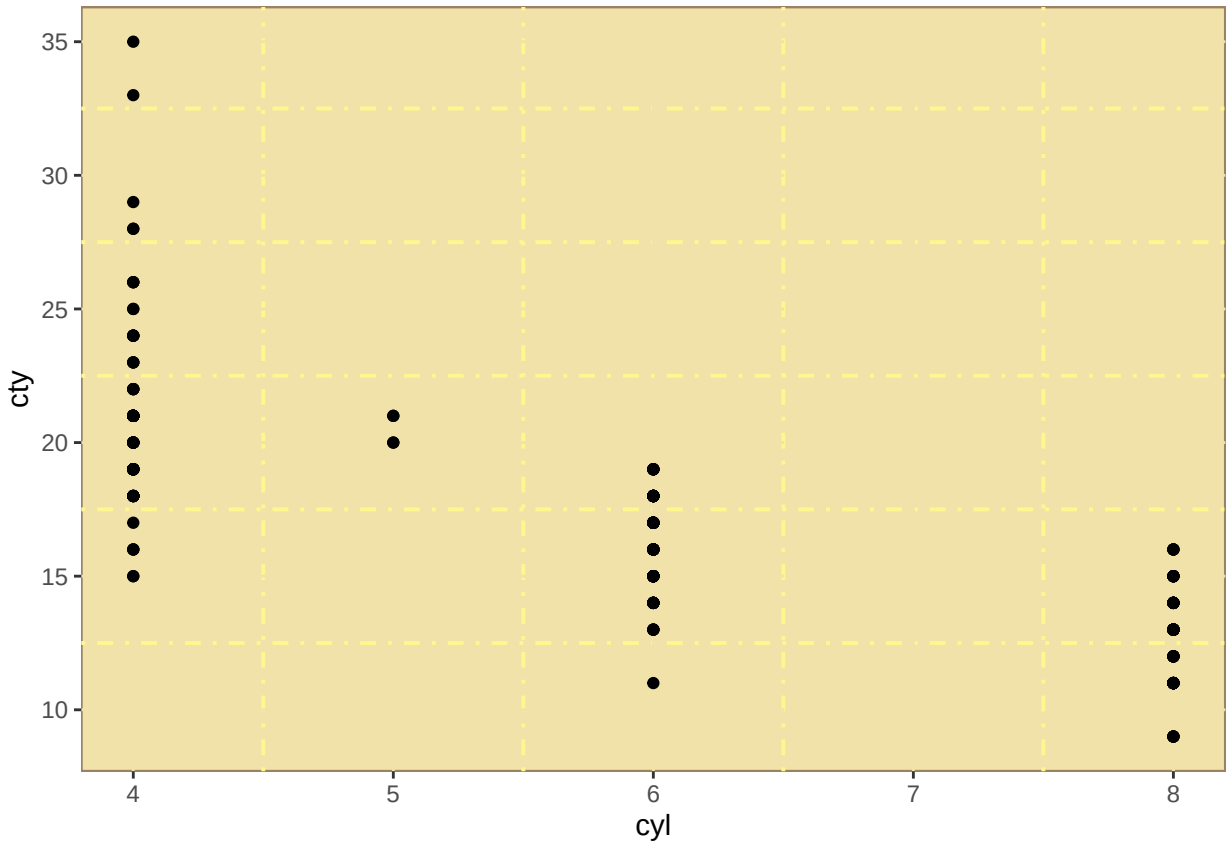
```
cv(mpg$displ)
```

```
## [1] 0.37213
```

Come è possibile osservare il coefficiente di variazione più alto si ottiene per la variabile *displ*, ciò vuol dire che in questo campione vi è una maggiore dispersione dei dati dalla media.

Scatterplot

```
p18 <- ggplot(mpg, aes(x = cyl, y = cty)) +
  geom_point()

p18 <- p18 +
  theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
        panel.grid.major = element_line(color = '#f0e2a8', linetype = 'longdash'),
        panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p18
```



All'aumentare del volume dei cilindri aumenta il consumo di carburante in città, infatti auto a 4 cilindri in media fanno più miglia per gallone rispetto alle auto con 8 cilindri.

Istogramma

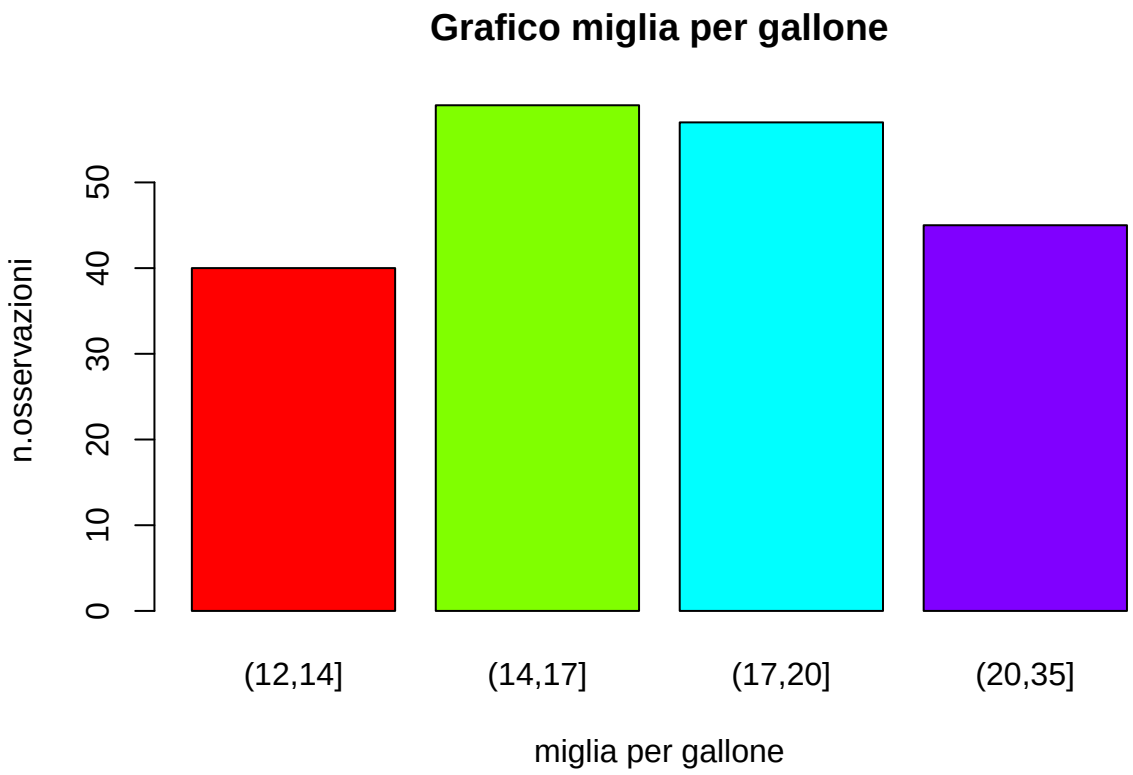
Utilizziamo la funzione **cut()** per raggruppare i dati e creare un istogramma con degli intervalli scelti.

```
fcut <- table (cut ( mpg$cty, breaks = c(12,14,17,20,35) ))
fcut
```

```
##
## (12,14] (14,17] (17,20] (20,35]
##      40      59      57      45
```

Gli intervalli sono scelti in modo da essere abbastanza equi.

```
barplot(fcut, col = rainbow(4),
        xlab = "miglia per gallone",
        main = "Grafico miglia per gallone",
        ylab = "n.osservazioni")
```



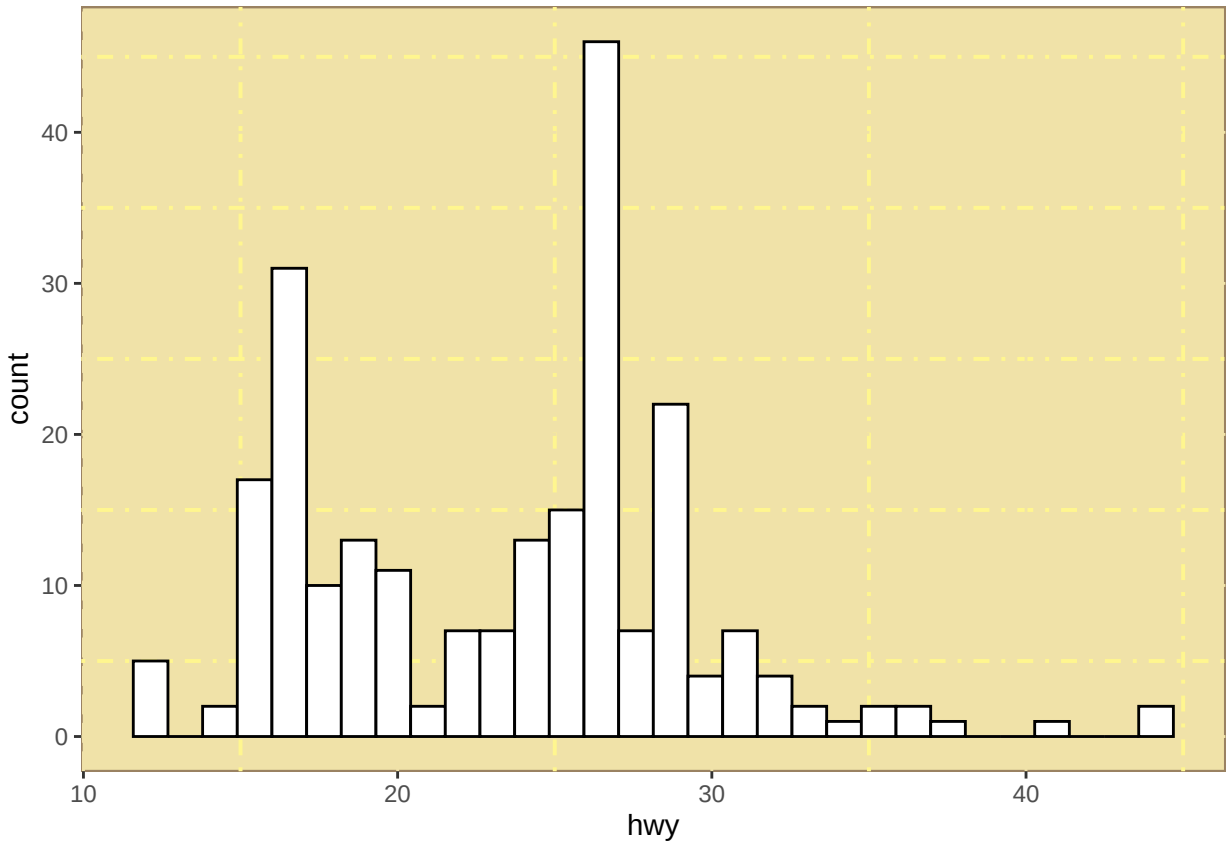
Variabile Hwy

La variabile hwy del dataset mpg rappresenta il numero di miglia per gallone che un veicolo può percorrere in autostrada.

Usando le funzioni di visualizzazione come **ggplot()** del pacchetto **ggplot2** è possibile creare grafici che mostrino le relazioni tra le variabili del dataset o la distribuzione di singole variabili. Ad esempio, abbiamo creato un istogramma per visualizzare i consumi di carburante **hwy** dei veicoli del dataset.

Istogramma

```
p19 <- ggplot(data = mpg, mapping = aes(hwy)) +
  geom_histogram(bins = 30, fill = "white", col = "black")

p19 <- p19 +
  theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
        panel.grid.major = element_line(color = '#f0e2a8', linetype = 'longdash'),
        panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p19
```



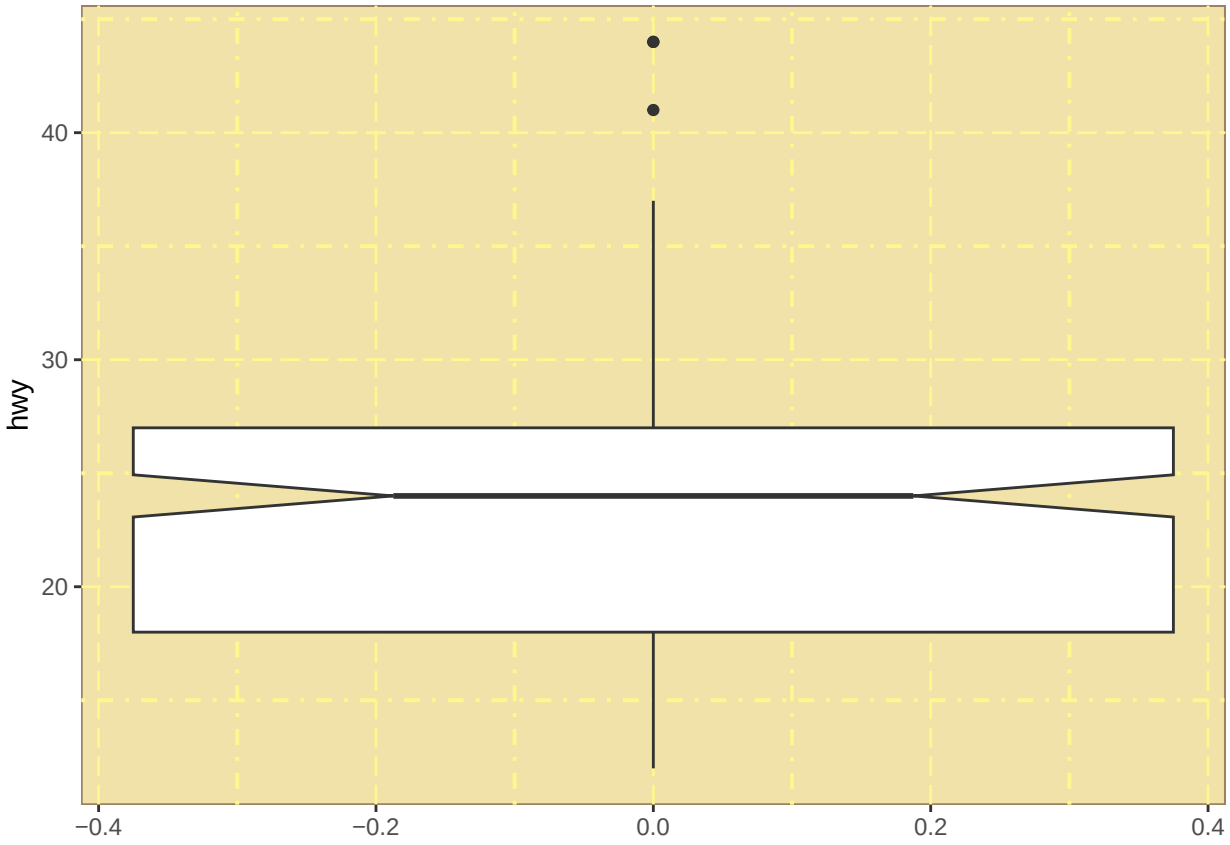
Tra 25 e 30 è presente il valore modale.

Boxplot ad intaglio

Possiamo utilizzare anche i boxplot ad intaglio i quali sono una rappresentazione grafica dei boxplot ma con l'aggiunta dell'intervallo di confidenza:

```
p8 <- ggplot(mpg, aes(x = hwy)) +
  geom_boxplot(notch = TRUE) +
  coord_flip()
```

```
p8 <- p8 +
  theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
        panel.grid.major = element_line(color = '#fff68f', linetype = 'longdash'),
        panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p8
```




```
IQR <- quantile(mpg$hwy,0.75) - quantile(mpg$hwy, 0.25)
M1 <- quantile(mpg$hwy,0.5) - 1.57 *IQR/sqrt(length(mpg$hwy))
M2 <- quantile(mpg$hwy,0.5) + 1.57 *IQR/sqrt(length(mpg$hwy))
c(M1 ,M2)
```

```
##      50%      50%
## 23.07629 24.92371
```

Il numero 1.57 è dovuto al rapporto $(1.25 \cdot 1.7)/1.35$ che appare nell'approssimazione della formula dell'intervallo di confidenza(che vedremo in seguito); quindi, con un grado di fiducia del 95% l'intervallo di confidenza approssimato per la mediana è $(23, 25)$.

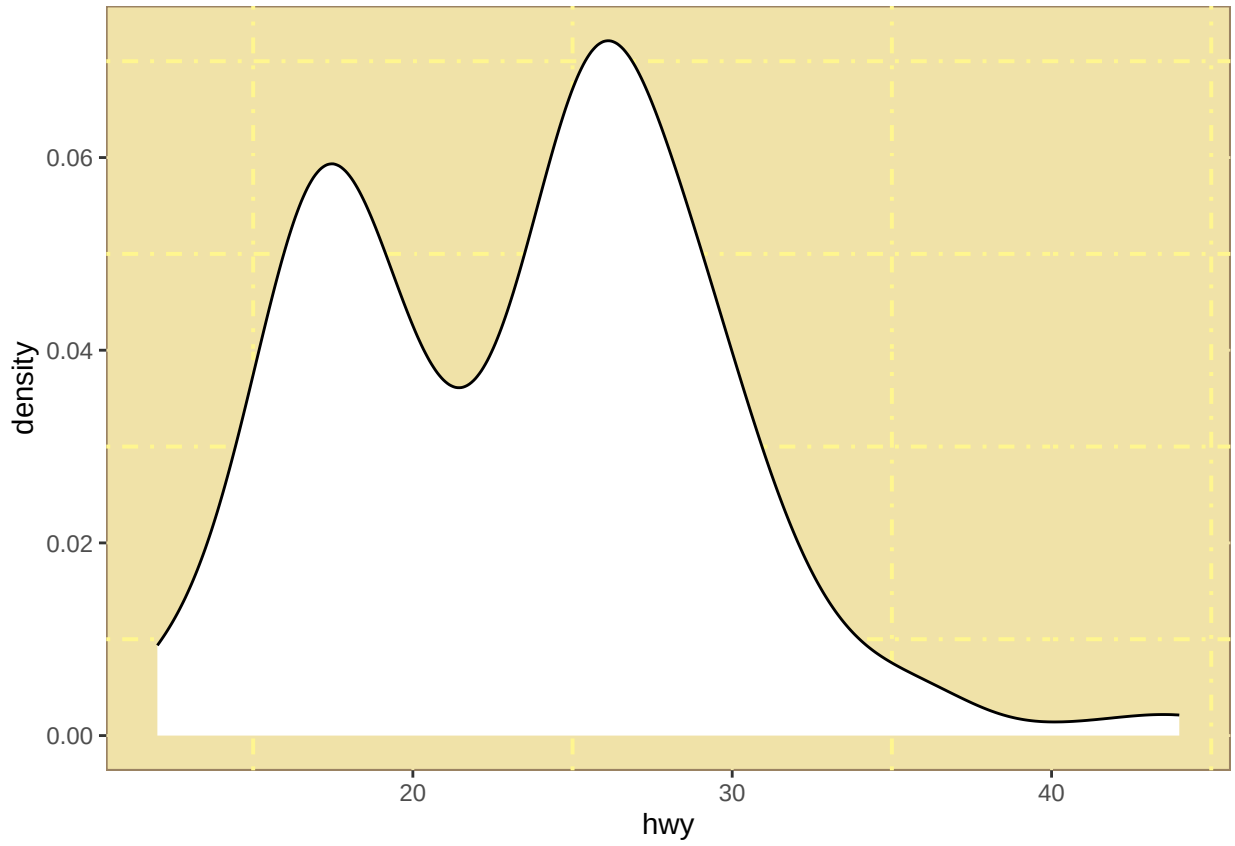
Kernel density plot

Il grafico kernel density plot è un grafico che mostra l'approssimazione della distribuzione dei valori di una variabile continua. È ottenuto tramite l'utilizzo di una tecnica di smoothing nota come "kernel density

estimation", che consiste nell'aggiungere una curva di densità di kernel sopra ogni osservazione. La curva di densità di kernel è una funzione continua che rappresenta la distribuzione di probabilità dei valori di una variabile.

Il grafico kernel density plot è spesso utilizzato per esplorare la distribuzione di una variabile e per confrontare le distribuzioni di diverse variabili. Ad esempio, abbiamo utilizzato il grafico kernel density plot per confrontare la distribuzione dei valori della variabile **hwy**.

```
p20 <- ggplot(data = mpg, mapping = aes(hwy)) +
  geom_density(fill = "white", col = "black")

p20 <- p20 +
  theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
        panel.grid.major = element_line(color = '#f0e2a8', linetype = 'longdash'),
        panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p20
```



La maggior parte dei veicoli presenti in questo dataset percorrono su strada tra i 20 e i 30 miglia con un gallone, possiamo notare anche che la distribuzione è bimodale (perchè presenta due mode), inoltre è presente un evidente asimmetria positiva perchè la coda di destra è più allungata rispetto alla coda di sinistra.

Introduciamo a tal proposito le definizioni di simmetria (skewness) e di curtosi di una distribuzione:

Coefficiente di simmetria

Assegnato un insieme di dati numerici $x_1, x_2, \dots x_n$ si definisce skewness campionaria il valore:

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

si nota che:

- $\gamma = 0$ se la distribuzione di frequenze è simmetrica
- $\gamma > 0$ se la distribuzione di frequenze è asimmetria positiva (ossia ha la coda di destra più allungata)
- $\gamma < 0$ se la distribuzione di frequenze è asimmetria negativa (ossia ha la coda di sinistra più allungata).

Si nota che γ è un indice adimensionale, ossia è indipendente dall'unità di misura dei dati.

Il calcolo della skewness campionaria può essere così implementato in R:

```
skw <- function (x){
n <-length (x)
m2 <-(n -1) *var (x)/n
m3 <- (sum ( (x-mean(x))^3) )/n
m3/(m2 ^1.5)
 }
```

```
skw(mpg$hwy)
```

```
## [1] 0.366865
```

il vettore presenta un'asimmetria positiva infatti ha la coda di destra più allungata.

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

Curtosi campionaria

Assegnato un insieme di dati numerici $x_1, x_2, ..., x_n$, si definisce curtosi campionaria il valore:

$$\gamma_2 = \beta_2 - 3$$

dove $\beta_2 = \frac{m_4}{m_2^2}$ è l'indice di Pearson che è un indice adimensionale, ossia è indipendente dall'unità di misura dei dati.

Gli indici γ_2 e β_2 permettono di confrontare la distribuzione di frequenze dei dati con una densità di probabilità normale standard, caratterizzata da $\beta_2 = 3$ e indice di curtosi $\gamma_2 = 0$. Se risulta:

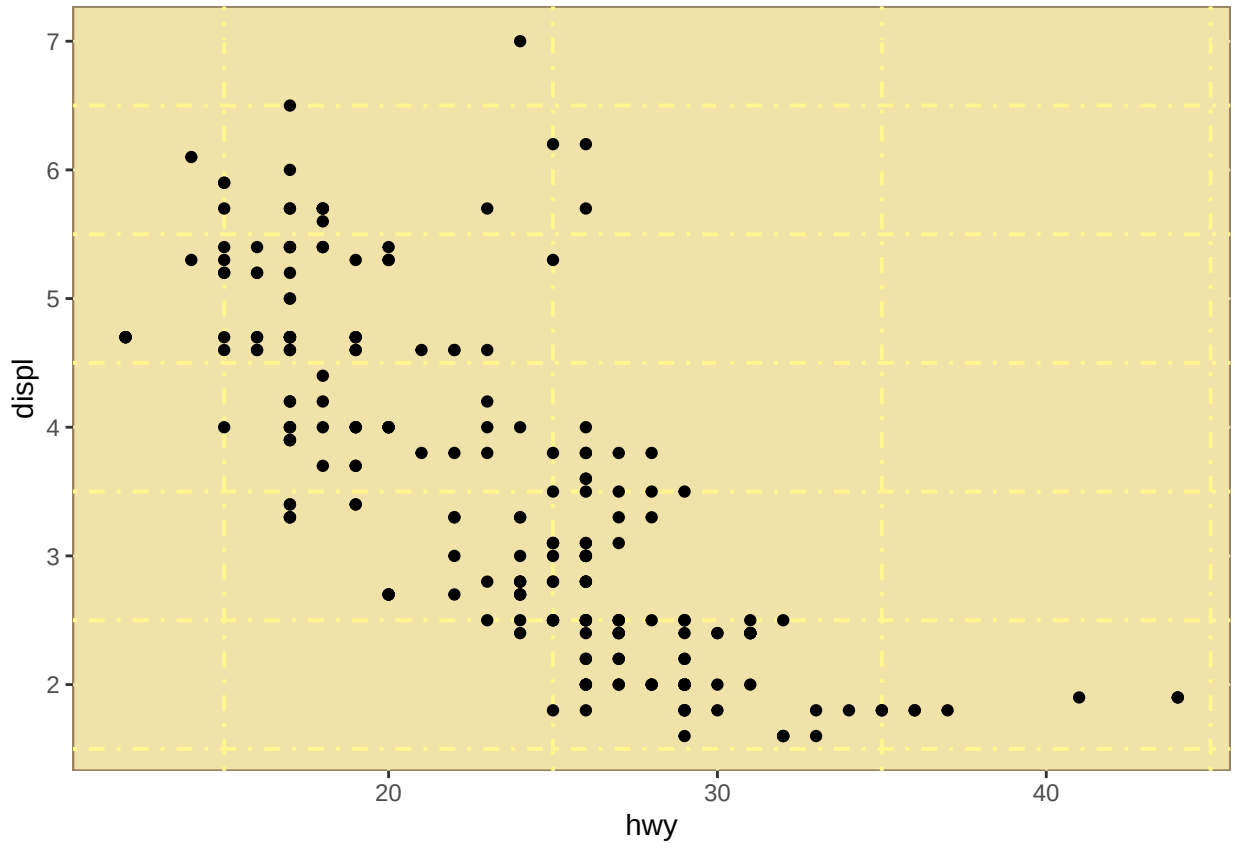
- $\beta_2 < 3$ ($\gamma_2 < 0$): la distribuzione di frequenze si definisce platicurtica, ossia la distribuzione di frequenze è più piatta di una normale;
- $\beta_2 > 3$ ($\gamma_2 > 0$): la distribuzione di frequenze si definisce leptocurtica, ossia la distribuzione di frequenze è più a punta di una normale;
- $\beta_2 = 3$ ($\gamma_2 = 0$): la distribuzione di frequenze si definisce normocurtica (mesocurtica), ossia piatta come una normale.

Il calcolo della curtosi campionaria ha significato soltanto per distribuzioni di frequenze unimodali, dato che tale indice è confrontato con quello di una normale standard.

Quindi nel nostro caso non ha senso calcolarlo dato che la nostra distribuzione è bimodale

Grafico a dispersione

```
p21 <- ggplot(data = mpg, mapping = aes(x = hwy, y = displ)) +
  geom_point()

p21 <- p21 +
  theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
        panel.grid.major = element_line(color = '#f0e2a8', linetype = 'longdash'),
        panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p21
```



Dal grafico possiamo notare che minore è il volume dei cilindri e minore è il consumo di carburante in miglia per gallone su autostrada, invece all'aumentare del numero di cilindri il numero di miglia possibili con un gallone diminuisce.

Funzione di distribuzione empirica continua

In statistica la funzione di distribuzione empirica, *viene usata per descrivere fenomeni quantitativi* comunque descritti con valori misurati su scale ordinali, intervallari o proporzionali, ma non se misurati con una scala nominale.

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

Come primo step creiamo le classi:

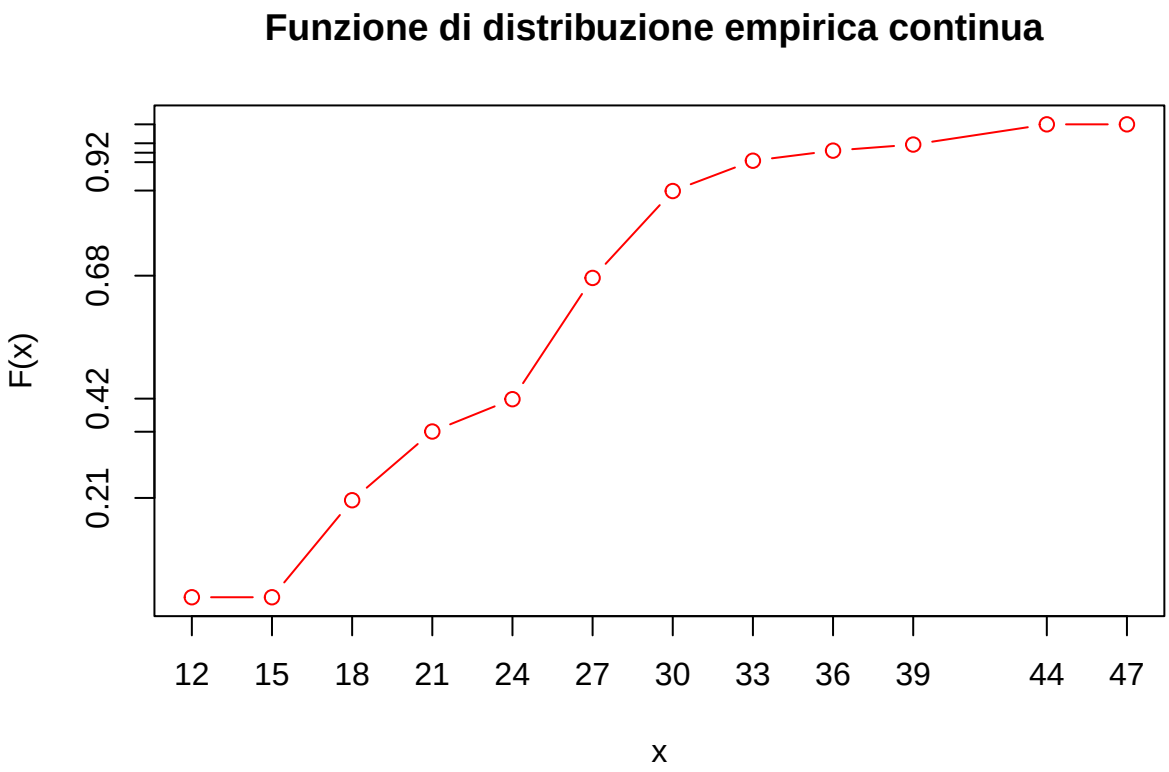
```
# Frequenza relativa delle osservazioni
freqrel <- table(mpg$hwy)/length(mpg$hwy)

# Lunghezza del vettore frequenza
m <- length(freqrel)

# Creazione delle classi
classi <-c(15 ,18 ,21 ,24 ,27, 30, 33, 36, 39, 44)

# Frequenze relative delle classi
freqClassi <- table(cut(mpg$hwy, breaks = classi, right = FALSE))/length(mpg$hwy)

# Frequenze cumulata
Fcum <- cumsum(freqClassi)
Fcum[9] <- 1
```

Per ottenere tali frequenze abbiamo utlizzato la funzione $cut()$ con l'opzione $right = FALSE$ che consente di costruire classi con intervalli chiusi a sinistra invece che a destra ed anche la funzione $cumsum()$ che permette di ottenere le frequenze cumulative.

Ricordiamo che in R la funzione $cut()$ trasforma dati numerici in dati qualitativi mediante la loro collocazione in opportune classi sulla base di quanto specificato nel parametro breaks.

Vogliamo ora scrivere delle linee di codice in R che consentono di ottenere il grafico della funzione di distribuzione empirica continua introducendo due classi fittizie $C_0 = (12, 15)$ e $C_5 = (44, 47)$ che ci serviranno per tracciare la linea $y = 0$ nell'intervallo *[12, 15)* e la linea $y = 1$ nell'intervallo *[44, 47)* nel grafico della funzione di distribuzione empirica continua.

```
ascisse <-c(12, 15 ,18 ,21 ,24 ,27, 30, 33, 36, 39, 44, 47)
ordinate <-c(0, 0, Fcum [1:9] ,1)
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()
```



Abbiamo realizzato il grafico della funzione di distribuzione empirica continua del vettore *hwy* utilizzando le classi:

```
[15,18)  [18,21)  [21,24)  [24,27)  [27,30)  [30,33) [33,36)  [36,39)  [39,44)
```

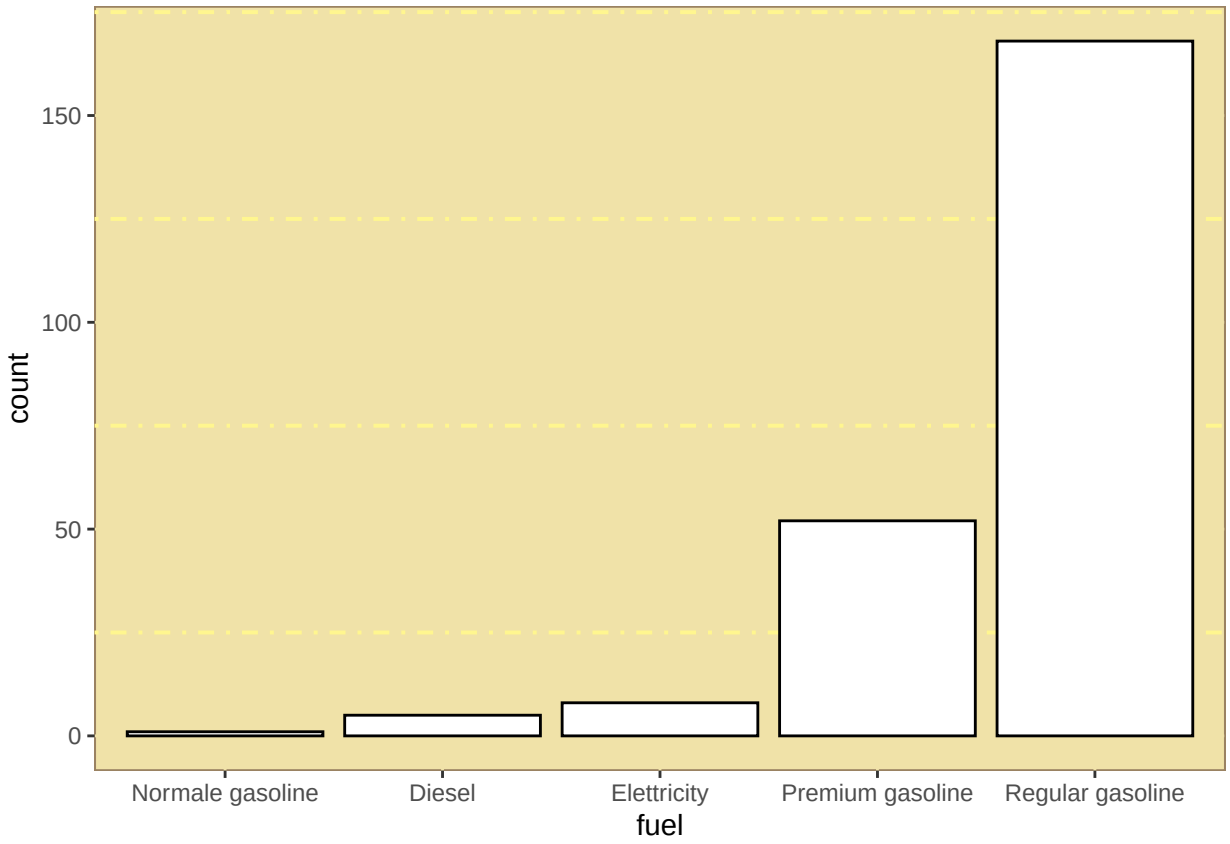
Variabile Fl

La variabile fl del dataset mpg rappresenta il tipo di carburante utilizzato dal veicolo. I valori possibili per la variabile fl sono "p", "c", "e", "r" e "d", che indicano rispettivamente: carburante premium, carburante normale (etanolo 85), elettricità, carburante regolare e diesel.

Grafico a barre

```
fuel <- factor(mpg$fl, levels = c("c","d","e","p","r"),
               labels = c("Normale gasoline", "Diesel",
                          "Elettricity","Premium gasoline",
                          "Regular gasoline"))

p22 <- ggplot(data = mpg, mapping = aes(x = fuel)) +
  geom_bar(col = "black", fill = "white")

p22 <- p22 +
  theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
```

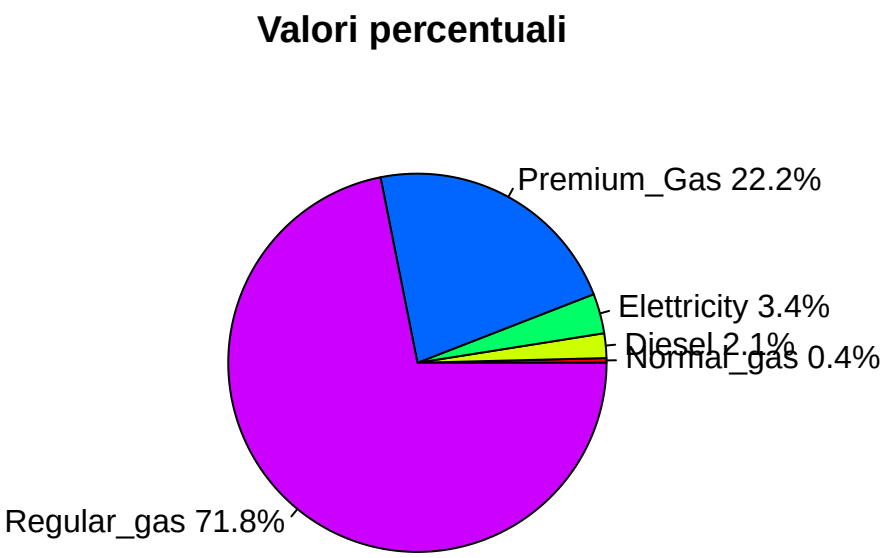
```
        panel.grid.major = element_line(color = '#f0e2a8', linetype = 'longdash'),
        panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p22
```



Possiamo notare che la maggiore parte delle auto presenti nel dataset di riferimento, circa il 72%, usa il gasolio regolare per viaggiare, il 22% si rifornisce con il carburante premium ed il restante 6% delle auto con gli altri tipi di risorse energetiche: elettricità, diesel o carburante normale.

Grafico a torta

```
freqRel <- table (mpg$fl )/ length (mpg$fl)
perc <- freqRel*100
perc <- round(perc,1)
LabelF <- c("Normal_gas", "Diesel", "Elettricity", "Premium_Gas", "Regular_gas")
LabelF <- paste(LabelF, perc)
LabelF <- paste(LabelF, "%", sep = "")
pie ( perc , label = LabelF , col = rainbow ( length ( LabelF )),
      main = "Valori percentuali ")
```

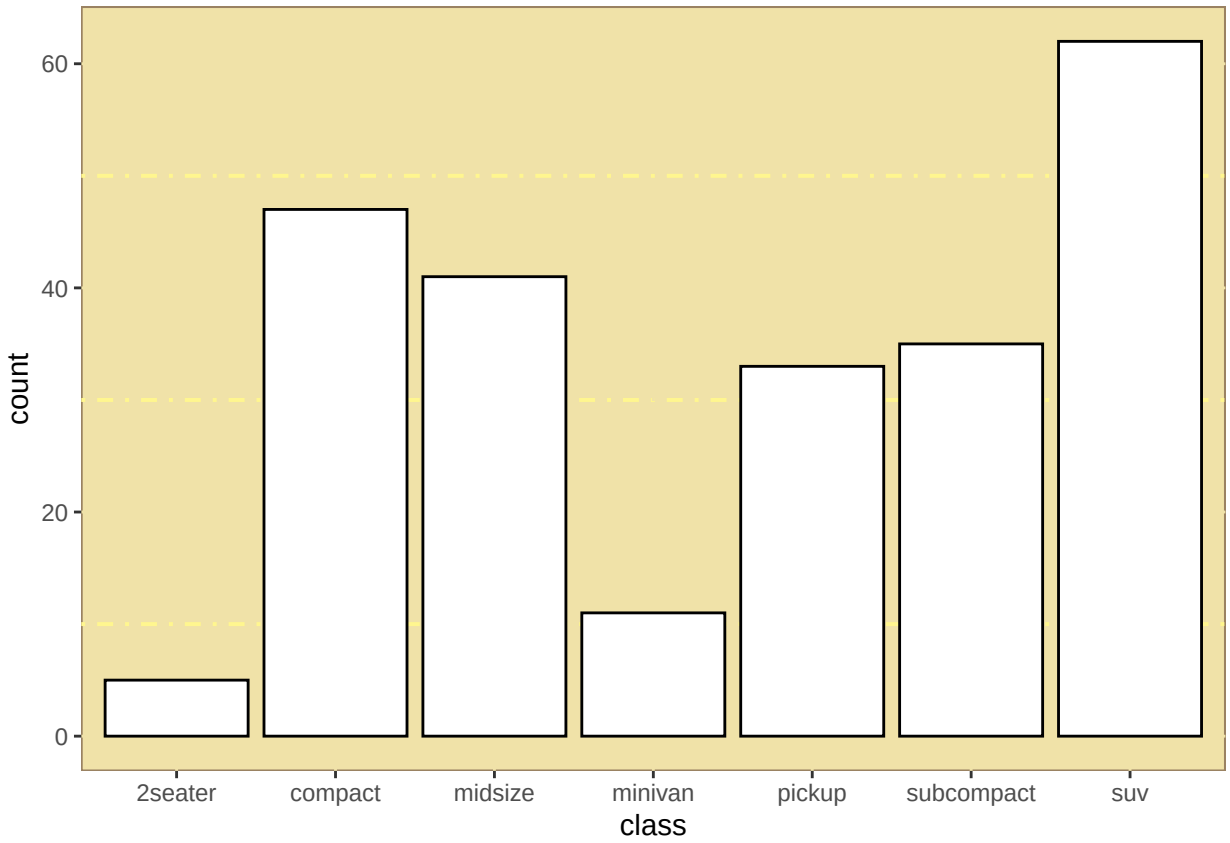


Variabile Class

La variabile Class rappresenta la classe di dimensioni dell'automobile.

Barplot

```
p23 <- ggplot(data = mpg, mapping = aes(class)) +
  geom_bar(fill = "white", col = "black")

p23 <- p23 +
  theme(panel.background = element_rect(fill = '#f0e2a8', color = '#9a8262'),
        panel.grid.major = element_line(color = '#f0e2a8', linetype = 'longdash'),
        panel.grid.minor = element_line(color = '#fff68f', size = 0.7,linetype = "dotdash"))

p23
```



Dall'istogramma possiamo notare che ci sono molti veicoli *suv* nel nostro dataset, l'altra classe di veicoli molto frequente è *compact* e *midsize*

Modello di regressione lineare

Il dataset "mpg" fornito con il pacchetto `ggplot2` in R potrebbe essere adeguato per eseguire un'analisi di regressione lineare multipla, a seconda della domanda di ricerca o dell'ipotesi che vogliamo verificare. La regressione lineare multipla è un modello statistico che permette di indagare le relazioni tra più variabili indipendenti (chiamate anche "predictor") e una variabile dipendente (chiamata anche "risposta").

Per eseguire un'analisi di regressione lineare multipla, è importante che il dataset contenga almeno una variabile dipendente e due o più variabili indipendenti. Nel dataset "mpg", le variabili "cty" (consumo di carburante in città) e "hwy" (consumo di carburante in autostrada) potrebbero essere utilizzate come variabile dipendente, mentre le altre variabili del dataset (ad esempio, "manufacturer", "displ", "year", "cyl", "trans", "drv" e "class") potrebbero essere utilizzate come variabili indipendenti.

Dato che lo scopo è quello di prevedere il consumo di carburante in base alle altre caratteristiche del veicolo (ad esempio, il volume del motore, il tipo di trasmissione o il tipo di carburante utilizzato), consideriamo l'utilizzo di un modello di regressione. Esistono diverse varianti di modelli di regressione, come ad esempio la regressione lineare semplice o la regressione multipla. In questo caso dato che ci sono molte variabili che andranno ad influire sulla variabile dipendente utilizzeremo il modello di regressione lineare multiplo e poi andremo a confrontarlo con il modello di regressione lineare semplice.

Per costruire un modello di regressione lineare multipla, è necessario avere una sufficiente quantità di dati che includano le variabili indipendenti e la variabile dipendente per ciascuna osservazione. È quindi possibile utilizzare questi dati per addestrare il modello e quindi utilizzarlo per fare previsioni sulla variabile dipendente per nuove osservazioni.

Se si definisce un data frame dfm, contenente n osservazioni delle p + 1 variabili $Y, X_1, X_2, ..., X_p$, allora $cov(dfm)$ e $cor(dfm)$ ci forniranno due matrici di dimensioni (p + 1) $\cdot$ (p + 1) i cui elementi solo le covarianze e le correlazioni tra coppie di variabili. In particolare, tali matrici sono simmetriche; la matrice delle covarianze contiene sulla diagonale principale la varianza delle singole colonne del data frame, mentre la matrice delle correlazioni contiene il numero 1 sulla diagonale principale. La matrice di correlazione evidenzia tutte le correlazioni lineari tra le coppie di variabili, ossia misura la forza del legame di natura lineare esistente tra tutte le coppie di variabili quantitative.

La funzione seguente ci permette di selezionare solo le variabili numeriche dunque le variabili di cui andremo a calcolare la correlazione e la covarianza.

```
dfm <- mpg %>%
  select_if(is.numeric)
```

Covarianza

La covarianza (cov) è una misura della correlazione tra due variabili. Indica se le due variabili cambiano in modo simile o in modo opposto e in che misura.

Assegnato un campione bivariato $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$ di una variabile quantitativa bidimensionale (X, Y), siano $\overline{x}$ e $\overline{y}$ rispettivamente le medie campionarie di $x_1, x_2, ..., x_n$ e di $y_1, y_2, ..., y_n$. La covarianza campionaria tra le due variabili X e Y è così definita:

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

con

$$(n = 2, 3, ...)$$

La covarianza campionaria può avere segno positivo, negativo o nullo. Quando $C_{xy} > 0$ si dice che le variabili sono correlate positivamente, se $C_{xy} < 0$ le variabili sono correlate negativamente e, infine, se $C_{xy} = 0$ le variabili sono non correlate.

In questo caso la matrice delle covarianze campionarie è

```
cov(dfm)
```

```
##            displ        year       cyl        cty         hwy
## displ  1.6691581  0.86137339  1.936767 -4.3906900 -5.89311104
## year   0.8613734 20.33690987  0.888412 -0.7145923  0.05793991
## cyl    1.9367668  0.88841202  2.597043 -5.5264664 -7.31139723
## cty   -4.3906900 -0.71459227 -5.526466 18.1130736 24.22543194
## hwy   -5.8931110  0.05793991 -7.311397 24.2254319 35.45777851
```

Lo svantaggio della covarianza è che non è racchiudibile in un range definito di valori, quindi non si riesce a capire quanto le variabili sono dipendenti tra loro, per questo introduciamo la correlazione.

Correlazione

Assegnato un campione bivariato $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$ di una variabile quantitativa bidimensionale (X, Y), siano $\overline{x}$ e s_x la media campionaria e la deviazione standard campionaria di $x_1, x_2, ..., x_n$ ed inoltre siano $\overline{y}$ e s_y la media campionaria e la deviazione standard campionaria di $y_1, y_2, ..., y_n$. Il coefficiente di correlazione campionario tra le due variabili X e Y è così definito:

$$r_{xy} = \frac{C_{xy}}{s_x s_y}$$

Si nota che il coefficiente di correlazione campionario

- è un indice adimensionale;
- non fa distinzione tra variabile dipendente e variabile indipendente, ossia $r_{xy} = r_{yx}$;
- può essere calcolato soltanto se entrambe le variabili sono quantitative;
- non cambia al variare dell'unità di misura con cui sono espresse le variabili;
- è fortemente influenzato dalla presenza di eventuali valori anomali, così come accade per la media campionaria e la varianza campionaria.

La correlazione (cor) è una misura della relazione lineare tra due variabili e può assumere valori compresi tra -1 e 1.

Il coefficiente di correlazione campionario ha lo stesso segno della covarianza. Quando:

- $r_{xy} > 0$ si dice che le variabili sono correlate positivamente
- $r_{xy} < 0$ le variabili sono correlate negativamente
- $r_{xy} = 0$ le variabili sono non correlate.

In questo caso la matrice delle correlazioni campionarie è

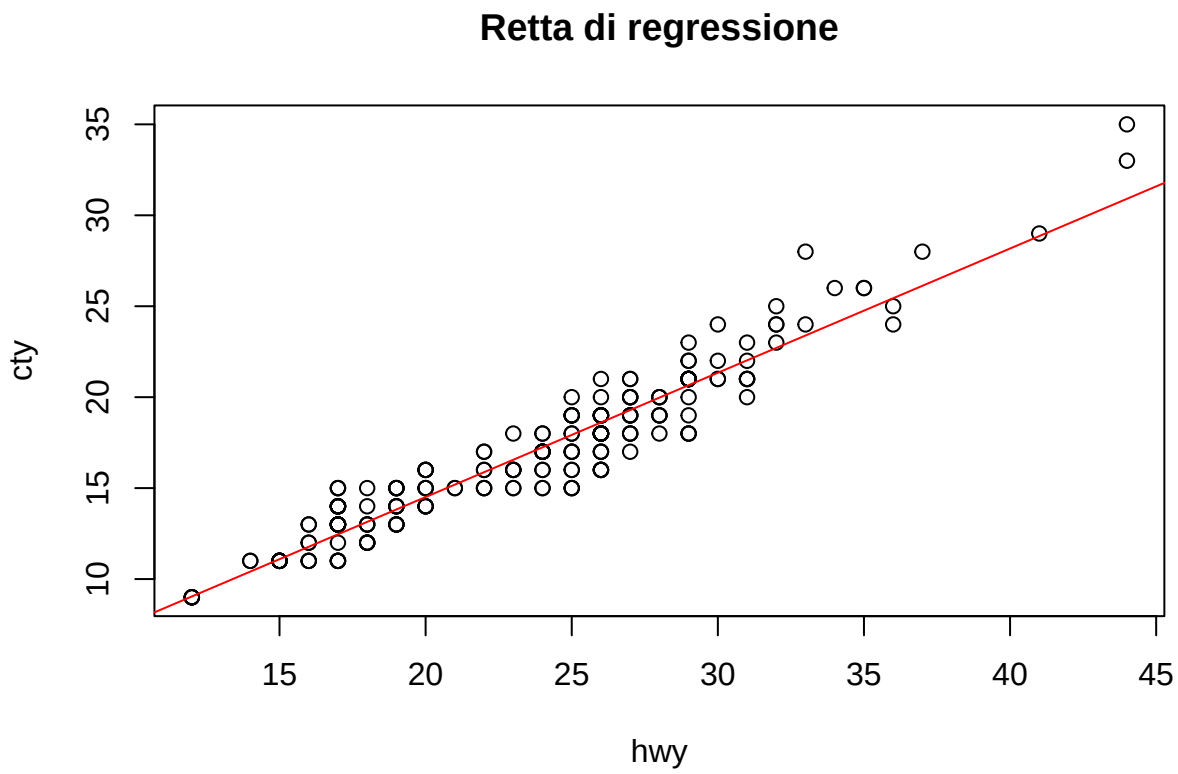
```
cor(dfm)
```

```
##            displ         year        cyl         cty          hwy
## displ  1.0000000  0.147842816  0.9302271 -0.79852397 -0.766020021
## year   0.1478428  1.000000000  0.1222453 -0.03723229  0.002157643
## cyl    0.9302271  0.122245347  1.0000000 -0.80577141 -0.761912354
## cty   -0.7985240 -0.037232291 -0.8057714  1.00000000  0.955915914
## hwy   -0.7660200  0.002157643 -0.7619124  0.95591591  1.000000000
```

Osservando i valori della correlazione possiamo notare che le variabili *displ* e *cyl* sono unite da un forte legame lineare, a differenza delle variabili *displ* e *cty*, *hwy* le quali sono correlate ma negativamente. Infine osservando i valori di *year* possiamo affermare che è correlata in modo molto debole con tutte le variabili.

Per avere un'ulteriore conferma della forte correlazione tra la variabile *hwy* e *cty* (0.95) rappresentiamo tale relazione in un diagramma a dispersione:

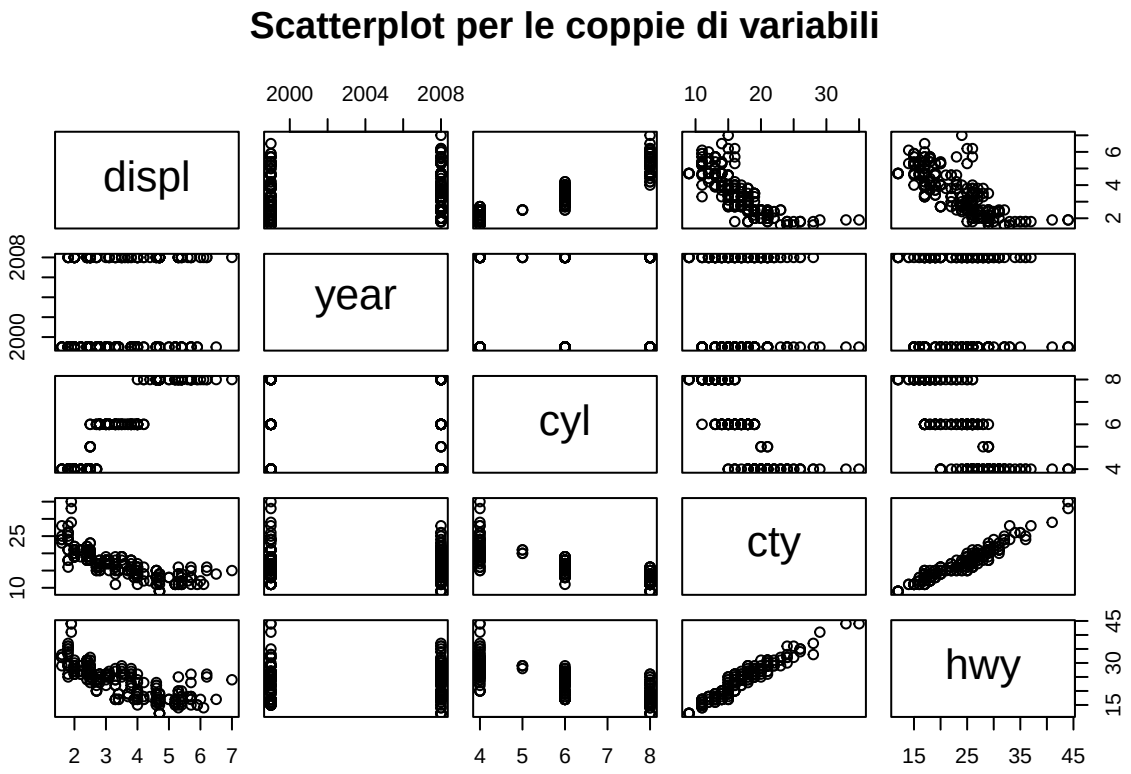
```
plot(mpg$hwy ,mpg$cty ,main =" Retta di regressione ", xlab="hwy",ylab="cty")
abline (lm( mpg$cty ~ mpg$hwy), col ="red")
```



Dal grafico è evidente che le due variabile sono fortemente legate linearmente, infatti lo scostamento delle osservazioni dalla retta di regressione lineare (retta interpolante ascendente) è minimo.

La funzione *pairs()* ci permette di visualizzare in un'unica finestra grafica una pluralità di scatterplot ottenuti mettendo in relazione tutte le coppie di variabili quantitative definite all'interno del data frame dfm.

```
pairs(dfm,  main = "Scatterplot per le coppie di variabili", col ="black")
```



Train set e test set

Il train set e il test set vengono utilizzati per valutare l'accuratezza di un modello di regressione. Il train set viene utilizzato per addestrare il modello, mentre il test set viene utilizzato per fare previsioni sulla variabile dipendente e quindi per valutare l'accuratezza del modello.

```
# Divisione del dataset mpg in un set di training (80%) e un set di test (20%)
set.seed(3)
partizioni <- createDataPartition(mpg$cty, p = 0.8, list = FALSE)

# Creazione del set di training utilizzando le partizioni ottenute
train_set <- mpg[partizioni, ]

# Creazione del set di test escludendo le partizioni utilizzate per il set di training
test_set <- mpg[-partizioni, ]
```

Modello di regressione lineare semplice

Il modello di regressione lineare semplice è uno dei metodi più comuni per analizzare la relazione tra una variabile dipendente (o di risposta) e una variabile indipendente (o predittiva). In particolare, il modello assume che la relazione tra le due variabili possa essere descritta da una linea retta.

In termini matematici, il modello di regressione lineare semplice è espresso come:

$$y = \beta_0 + \beta_1 X + \in$$

dove y è la variabile dipendente, x è la variabile indipendente, β_0 e β_1 sono i coefficienti di intercetta e di pendenza della retta di regressione, e $\in$ rappresenta l'errore residuo.

L'obiettivo della regressione lineare è quello di stimare i coefficienti β_0 e β_1 in modo da ottenere la linea retta che meglio si adatta ai dati. Questo viene fatto minimizzando la somma dei quadrati degli errori residui (SSE), ovvero la somma delle differenze tra i valori osservati di y e i valori predetti dalla retta di regressione.

Il modello di regressione lineare semplice può essere utilizzato per molteplici scopi, tra cui la previsione di valori futuri di y sulla base di x, la valutazione dell'effetto di x su y e l'identificazione di eventuali outlier o influenti nella relazione tra le due variabili.

```
modello_semplice <- lm(cty ~ displ, data = train_set)
summary(modello_semplice)
```

```
##
## Call:
## lm(formula = cty ~ displ, data = train_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.2708 -1.5193 -0.2695  0.9810 14.2315
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  25.5152     0.5243   48.66   <2e-16 ***
## displ        -2.4983     0.1407  -17.75   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.54 on 187 degrees of freedom
## Multiple R-squared:  0.6276, Adjusted R-squared:  0.6256
## F-statistic: 315.2 on 1 and 187 DF,  p-value: < 2.2e-16
```

Osservando il summary possiamo affermare che la variabile *displ* influenza la variabile dipendente ma negativamente (infatti ha segno negativo e dunque anche la pendenza della retta sarà decrescente), pertanto possiamo affermare che al decrescere di *displ* la variabile *cty* aumenta.

Previsioni

Dopo aver stimato il modello di regressione possiamo prevedere i valori della variabile dipendente su un set di dati che non è stato usato per addestrare il modello.

```
previsioni_semplice <- predict(modello_semplice, test_set)
```

Coefficiente di determinazione

Il coefficiente di determinazione per la regressione lineare semplice è *il rapporto tra la varianza dei valori stimati tramite la retta di regressione e la varianza dei valori osservati.* Pertanto, se si denota con $y_1, y_2, \ldots, y_n$

il vettore dei dati della variabile dipendente, con $\overline{y}$ la sua media campionaria e con $\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n$ i valori stimati attraverso la retta di regressione (la cui media campionaria è $\overline{y}$), coefficiente di determinazione (detto anche r-square) è così definito:

$$D^2 = \frac{\frac{1}{n-1}\sum_{i=1}^{n}(\hat{y}_i - \overline{y})^2}{\frac{1}{n-1}\sum_{i=1}^{n}(y_i - \overline{y})^2} = \frac{\sum_{i=1}^{n}(\hat{y}_i - \overline{y})^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2}$$

in R lo otteniamo con il seguente comando:

```
summary(modello_semplice)$r.squared
```

```
## [1] 0.6276219
```

Il valore ottenuto è molto alto e ciò indica che il regressore predice bene il valore della variabile dipendente.

Nel caso di regressione lineare semplice, il coefficiente di determinazione coincide con il quadrato del coefficiente di correlazione, ossia:

$$D^2 = r_{xy}^2$$

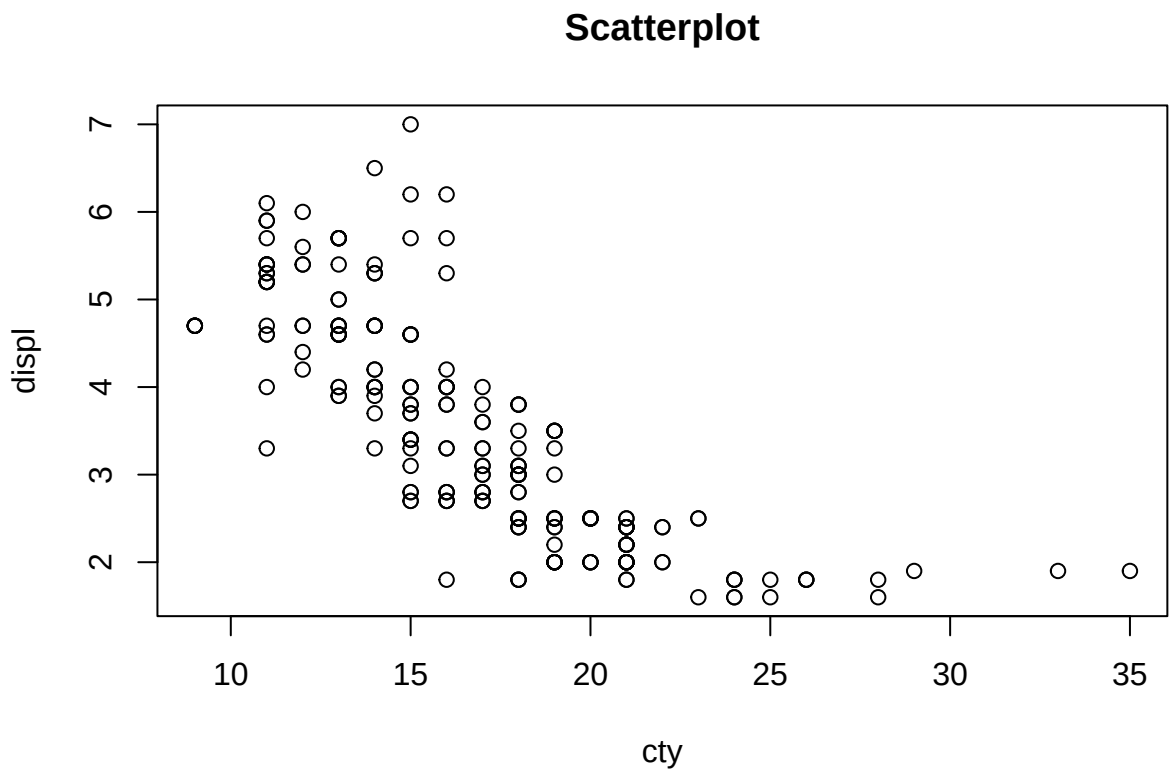
Modello di regressione con trasformazione semilogaritmica

Consideriamo il modello non lineare:

$$Y = e^{\alpha + \beta X}$$

Confrontiamo le variabili *cty* e *displ,* visualizzando i punti nel seguente scatterplot:

```
plot(mpg$cty, mpg$displ, main="Scatterplot", xlab="cty", ylab="displ")
```



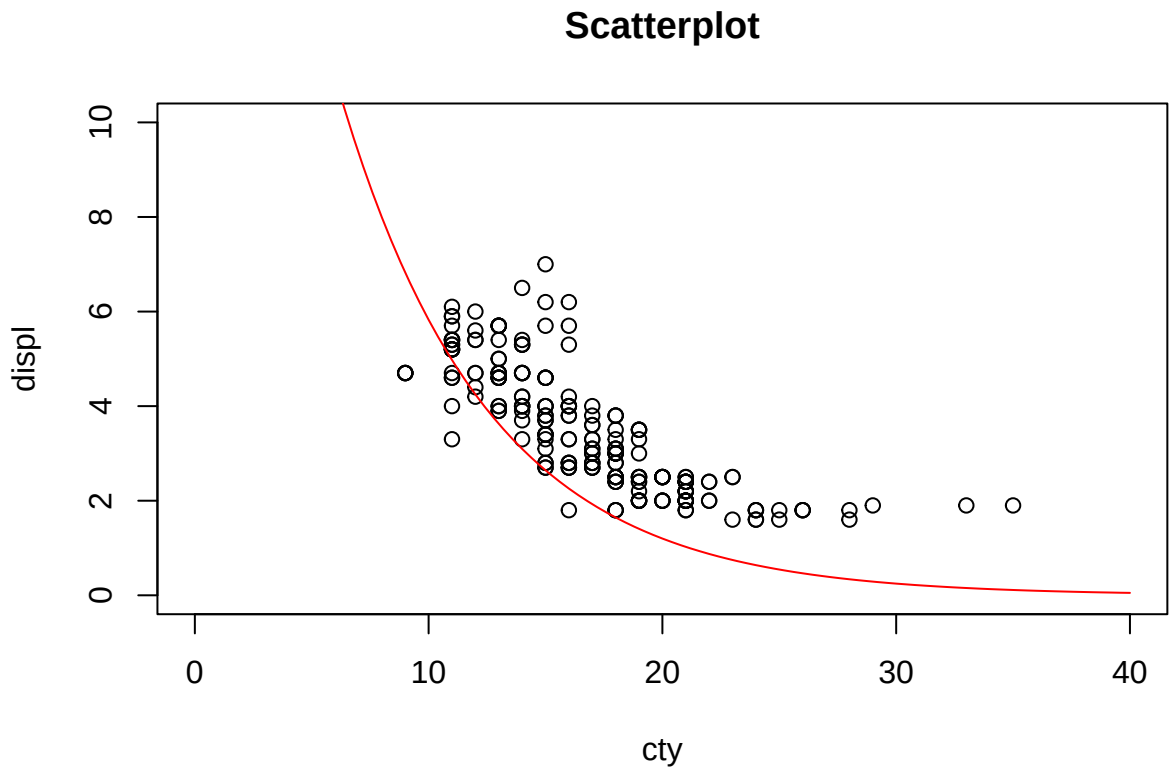
Osservando lo scatterplot si nota che un modello lineare non approssima bene i dati, consideriamo quindi il modello di regressione semilogaritmica:

```
modello_semilog <-lm(I(log(mpg$cty))~ mpg$displ)
summary(modello_semilog)
```

```
##
## Call:
## lm(formula = I(log(mpg$cty)) ~ mpg$displ)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.42329 -0.08219 -0.00317  0.08111  0.51292
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.342686   0.026879  124.36   <2e-16 ***
## mpg$displ   -0.158030   0.007258  -21.77   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1431 on 232 degrees of freedom
## Multiple R-squared:  0.6714, Adjusted R-squared:   0.67
## F-statistic: 474.1 on 1 and 232 DF,  p-value: < 2.2e-16
```

Disegniamo ora la curva stimata sullo scatterplot:

```
alpha <- modello_semilog$coefficients[[1]]
beta <- modello_semilog$coefficients[[2]]
plot(mpg$cty, mpg$displ, main="Scatterplot",
     xlim = c(0,40),
     ylim = c(0,10),
     xlab="cty", ylab="displ")
curve(exp(alpha + beta*x), add =TRUE, col="red")
```



Il coefficiente di determinazione è:

```
stime <- exp(alpha +beta*mpg$displ)

# Definizione 1
num1 <-sum ((( mpg$cty- stime )^2) )
den1 <-sum ((( mpg$cty- mean(mpg$cty))^2) )
d2 <-1- num1 /den1
d2
```

```
## [1] 0.6702734
```

Possiamo notare che il coefficiente del modello con trasformazione semilogaritmica è migliore rispetto al modello di regressione lineare semplice che era:

```
0.6276219
```

Modello di regressione lineare multiplo

Il modello di regressione lineare multipla è un tipo di modello di regressione che viene utilizzato per prevedere una variabile dipendente (chiamata anche variabile risposta o target) in base alla relazione con più variabili indipendenti (chiamate anche variabili esplicative o predictor). Il modello di regressione lineare multipla assume che ci sia una relazione lineare tra la variabile dipendente e le variabili indipendenti.

La regressione lineare multipla viene solitamente utilizzata quando ci sono più di una variabile indipendente che potrebbero influire sulla variabile dipendente. Ad esempio, potresti utilizzare il modello di regressione lineare multipla per prevedere il consumo di carburante di un'automobile (variabile dipendente) in base al volume del motore (variabile indipendente), al tipo di trasmissione (variabile indipendente) e al tipo di carburante utilizzato (variabile indipendente).

```
modello <- lm(cty ~ displ + trans + fl + cyl + drv, data = train_set)
summary(modello)
```

```
##
## Call:
## lm(formula = cty ~ displ + trans + fl + cyl + drv, data = train_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.2490 -0.9830 -0.2140  0.9035  6.6321
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      28.7981     2.2141  13.007  < 2e-16 ***
## displ            -0.7579     0.3200  -2.368  0.01899 *
## transauto(l3)    -1.2846     1.5752  -0.815  0.41593
## transauto(l4)    -1.9259     0.9166  -2.101  0.03708 *
## transauto(l5)    -1.5821     0.9499  -1.666  0.09763 .
## transauto(l6)    -2.0754     1.2150  -1.708  0.08942 .
## transauto(s4)     0.6542     1.6004   0.409  0.68322
## transauto(s5)    -0.6064     1.3541  -0.448  0.65482
## transauto(s6)    -0.5663     0.9823  -0.577  0.56502
## transmanual(m5)  -0.8105     0.9246  -0.877  0.38194
## transmanual(m6)  -0.5425     0.9502  -0.571  0.56879
## fld               3.6722     2.0891   1.758  0.08057 .
## fle              -6.2993     2.0396  -3.089  0.00235 **
## flp              -3.5576     1.9041  -1.868  0.06342 .
## flr              -3.0701     1.8918  -1.623  0.10646
## cyl              -1.0609     0.2310  -4.594 8.42e-06 ***
## drvf              2.3918     0.3785   6.320 2.20e-09 ***
## drvr              2.1235     0.5108   4.157 5.08e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.833 on 171 degrees of freedom
## Multiple R-squared:  0.8227, Adjusted R-squared:  0.805
## F-statistic: 46.66 on 17 and 171 DF,  p-value: < 2.2e-16
```

Nel modello di regressione lineare multiplo si è interessati a sottoporre a verifica l'ipotesi nulla, la quale equivale a dire che la variabile X_j non influenza la variabile Y.

- $H_0 : \beta_j = 0$
- $H_1 : \beta_j \neq 0$

la statistica test t è :

$$t = \frac{(\overline{\beta}_j)}{s.\sqrt{(c_{jj})}} \sim t_{n-p-1}$$

Fissato un livello di significatività α si determina il valore soglia $t_{\alpha/2}$ tale che:

$$P(|T| > t_{\alpha/2}) = P(T < -t_{\alpha/2}) + P(T > t_{\alpha/2}) = \alpha$$

Si rifiuta l'ipotesi *nulla* se:

$$|t| > t_{\alpha/2}$$

Con il seguente codice escludiamo le variabili indipendenti che non influenzano la variabile dipendente, ovvero le variabili *trans* e *fl*.

```
modello_aggiornato <- update(modello, ~ . - trans - fl)
summary(modello_aggiornato)
```

```
##
## Call:
## lm(formula = cty ~ displ + cyl + drv, data = train_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.9343 -1.1467 -0.1853  1.1047 13.5378
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  25.4744     0.8186  31.121  < 2e-16 ***
## displ        -0.5917     0.3656  -1.619 0.107259
## cyl          -1.3498     0.2643  -5.106 8.16e-07 ***
## drvf          2.5111     0.4156   6.042 8.27e-09 ***
## drvr          2.0206     0.5814   3.475 0.000637 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.174 on 184 degrees of freedom
## Multiple R-squared:  0.7315, Adjusted R-squared:  0.7257
## F-statistic: 125.3 on 4 and 184 DF,  p-value: < 2.2e-16
```

Osserviamo che i segni di entrambi i regressori *displ* e *cyl* sono negativi; quindi essi hanno un effetto negativo sulla variabile *cty* (all'aumentare di *displ* e *cyl* diminuisce *cty*). Ciò non vale per la variabile *drv* la quale ha segno positivo. Si nota anche che il regressore *displ* è prossimo allo zero indicando che non incide in modo significativo sulla variabile dipendente.

Pertanto possiamo dedurre che sono 3 le variabili che contribuiscono a definire la variabile dipendente *cty* (miglia per gallone) e sono: *displ*, *cyl* e *drv*.

Previsioni

Dopo aver stimato il modello di regressione possiamo prevedere i valori della variabile dipendente su un set di dati che non è stato usato per addestrare il modello.

```
previsioni_aggiornato <- predict(modello_aggiornato, test_set)
```

Misure di accuratezza

Per valutare l'accuratezza delle previsioni del modello di regressione lineare multipla, possiamo utilizzare diverse metriche di valutazione e poi confrontarle con il modello di regressione lineare semplice. Ad esempio:

- Errore quadratico medio (MSE): indica la deviazione media quadratica tra i valori previsti dal modello e i valori effettivi della variabile dipendente. Un valore di MSE piccolo indica che il modello fa previsioni accurate, mentre un valore di MSE grande indica che il modello fa previsioni meno accurate.

```
MSE_s <- mean((test_set$cty - previsioni_semplice)^2)
MSE_s
```

```
## [1] 7.327401
```

```
MSE_m <- mean((test_set$cty - previsioni_aggiornato)^2)
MSE_m
```

```
## [1] 5.571242
```

- Coefficiente di determinazione (R^2): indica la quantità di varianza nella variabile dipendente che viene spiegata dal modello. Un valore di R^2 vicino a 1 indica che il modello spiega in modo accurato la varianza nella variabile dipendente, mentre un valore di R^2 vicino a 0 indica che il modello non spiega in modo accurato la varianza nella variabile dipendente.

```
modello_test_s <- lm(cty ~ displ, data = test_set)
summary(modello_test_s)$r.squared
```

```
## [1] 0.7056375
```

```
modello_test_c <- lm(cty ~ displ + cyl + drv, data = test_set)
summary(modello_test_c)$r.squared
```

```
## [1] 0.7599574
```

- Errore assoluto medio (MAE): indica la deviazione media assoluta tra i valori previsti dal modello e i valori effettivi della variabile dipendente. Un valore di MAE piccolo indica che il modello fa previsioni accurate, mentre un valore di MAE grande indica che il modello fa previsioni meno accurate.

```
MAE_s <- mean(abs(test_set$cty - previsioni_semplice))
MAE_s
```

```
## [1] 1.815805
```

```r
MAE_c <- mean(abs(test_set$cty - previsioni_aggiornato))
MAE_c
```

```
## [1] 1.50492
```

	Modello di regressione semplice	Modello di regressione multiplo
Errore quadratico medio (MSE)	7.327401	5.571242
Coefficiente di determinazione (R^2)	0.7056375	0.7599574
Errore assoluto medio (MAE)	1.815805	1.50492

Osservando le misure di accuratezza possiamo affermare che il modello di regressione multiplo che abbiamo stimato è un buon modello ovvero è un modello che è in grado di fare previsioni accurate della variabile dipendente utilizzando le variabili indipendenti: *displ*, *cyl* e *drv*.

Residui

In generale, esisteranno degli scostamenti (residui) tra le ordinate dei punti y_i (valori osservati) e i corrispondenti valori stimati $\hat{y}_i$.

I residui sono così definiti:

$$E_i = y_i - \hat{y}_i = y_i - (\alpha + \beta x_i)$$

con

$$(i = 1, 2, ..., n)$$

e mostrano di quanto si discostano i valori osservati y_i dai valori stimati $\hat{y}_i$ con la regressione lineare multipla. In R per calcolare il vettore dei valori stimati $(\hat{y}_1, \hat{y}_2, ..., \hat{y}_n)$ tramite regressione lineare multipla si utilizza la funzione:

```r
# Visualizza il vettore delle stime
fit <- fitted(lm(data = mpg,cty ~ displ + cyl + drv))
```

Invece, per calcolare il vettore dei residui $(E_1, E_2, ..., E_n)$ si utilizza la funzione:

```r
# Visualizza i residui
residui <- resid(lm(data = mpg,cty ~ displ + cyl + drv))
```

L'analisi dei residui in un modello di regressione lineare multipla consiste nel verificare se i residui, ovvero la **differenza tra i valori osservati e quelli previsti dal modello**, seguono una distribuzione normale e se sono distribuiti in modo casuale. Ciò è importante perché una distribuzione normale e casuale dei residui indica che il modello è adeguato alla relazione tra le variabili dipendenti e indipendenti.

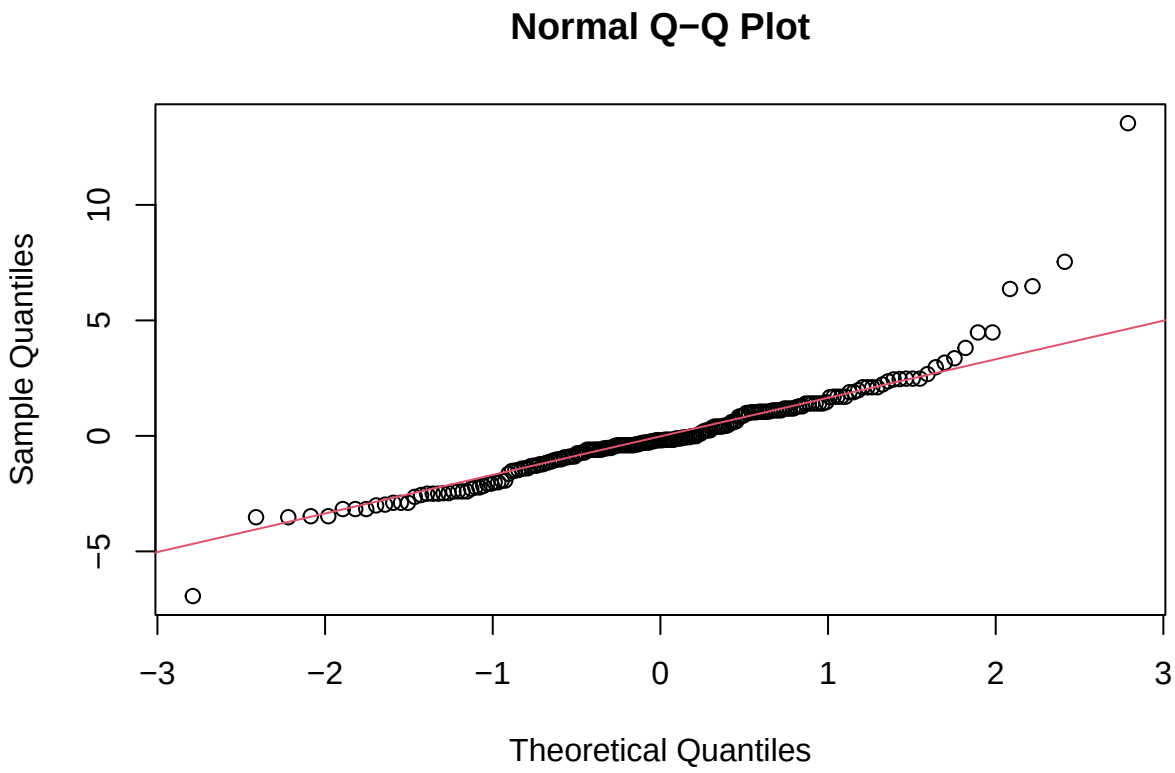
Per eseguire un'analisi dei residui, è possibile utilizzare alcuni strumenti come un grafico di istogramma dei residui, un grafico Q-Q plot e un grafico di dispersione dei residui sui valori previsti.

Il grafico di istogramma dei residui mostra la distribuzione dei residui, e dovrebbe essere simile ad una distribuzione normale. Il grafico Q-Q plot mostra se i residui seguono una distribuzione normale, in caso contrario i punti non si troverebbero sulla linea di regressione. Infine il grafico di dispersione dei residui sui valori previsti deve mostrare una distribuzione casuale, ovvero senza alcun schema visibile.

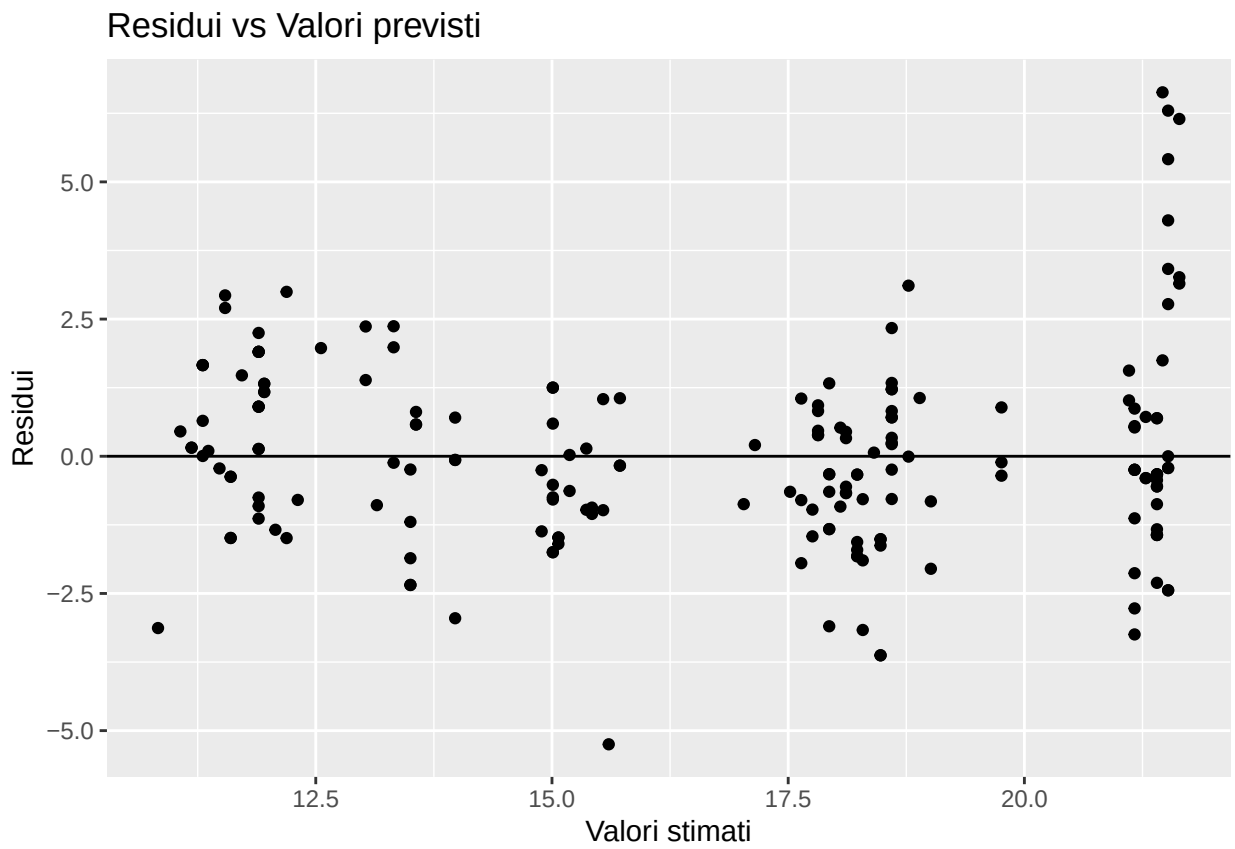
Nella realtà, spesso i dati non seguono perfettamente una distribuzione normale, ma è importante valutare se l'adesione al modello non sia troppo lontana dai valori attesi.

Si può usare la funzione plot() per creare questi grafici, e in R si può utilizzare la libreria **ggplot2**

```
# Q-Q plot
qqnorm(modello_aggiornato$residuals)
qqline(modello_aggiornato$residuals, col = 2)
```



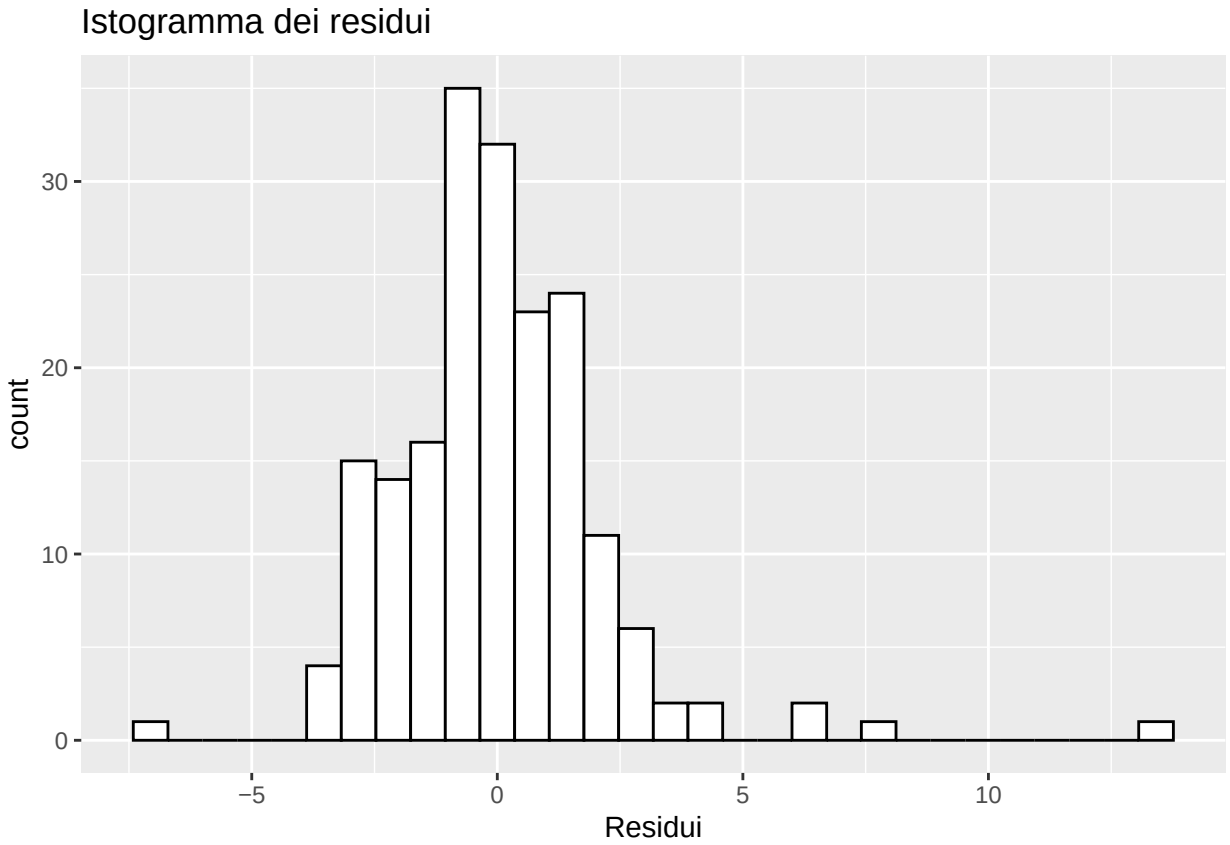

```
# Scatterplot dei residui vs valori stimati
ggplot(train_set, aes(x = modello_aggiornato$fitted.values,
                      y = modello$residuals))+
geom_point() +
ggtitle("Residui vs Valori previsti")+
geom_hline(yintercept = 0) +
xlab("Valori stimati") +
ylab("Residui")
```



La media campionaria dei residui $\overline{E}$ è nulla, ossia in media gli scostamenti positivi e negativi si compensano. Infatti:

$$\overline{E} = \frac{1}{n}\sum_{i=1}^{n} E_i = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i) = \overline{y} - \frac{1}{n}\hat{y} = 0$$

```
# Istogramma dei residui
ggplot(train_set, aes(modello_aggiornato$residuals)) +
  geom_histogram(bins = 30, col = "black", fill = "white") +
  ggtitle("Istogramma dei residui") + xlab("Residui")
```



Come possiamo osservare dal grafico la distribuzione è leggermente asimmetrica positiva a causa dei due outlier presenti dopo il valore *x=10.*

Se i residui non seguono una distribuzione normale e casuale, può essere necessario modificare il modello, ad esempio aggiungendo o rimuovendo delle variabili, o utilizzando una trasformazione delle variabili.

Coefficiente di determinazione

Il coefficiente di determinazione in un modello di regressione lineare multipla è *il rapporto tra la varianza dei valori stimati tramite la funzione di regressione multipla e la varianza i valori osservati della variabile dipendente.* Se si denota con $(y_1, y_2, ..., y_n)$ il vettore dei dati della variabile dipendente, con $\overline{y}$ la sua media campionaria e con $(\hat{y}_1, \hat{y}_2, ..., \hat{y}_n)$ i valori stimati attraverso la funzione di regressione (la cui media campionaria $\overline{y}$), il coefficiente di determinazione è:

$$D^2 = \frac{\frac{1}{n-1}\sum_{i=1}^{n}(\hat{y}_i - \overline{y}_i)^2}{\frac{1}{n-1}\sum_{i=1}^{n}(y_i - \overline{y}_i)^2} = \frac{\sum_{i=1}^{n}(\hat{y}_i - \overline{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y}_i)^2}$$

L'indice D^2 è adimensionale e risulta $0 \leq D^2 \leq 1$. Quando $D^2 = 0$ il modello di regressione multipla utilizzato non spiega per nulla i dati. Invece, quando $D^2 = 1$ il modello di regressione multipla utilizzato spiega perfettamente i dati. In R per calcolare l'indice D^2 per la regressione lineare multipla basta utilizzare la funzione:

```
summary(modello_aggiornato)$r.squared
```

```
## [1] 0.7315415
```

Il coefficiente di determinazione è *0.73*, ossia il modello di regressione multipla utilizzato spiega significativamente i dati ed è migliore perfino del modello con trasformazione semilogaritmica.

Modello di regressione semplice	Modello con trasformazione s emilogaritmica	Modello di regressione multiplo
`0.6276219`	`0.6702734`	`0.7315415`

Secondo il coefficiente di determinazione il modello di regressione multiplo è di gran lunga migliore.

Analisi dei cluster

L'analisi dei cluster è una metodologia che raggruppa le entità di un insieme più grande in sottoinsiemi detti cluster. L'obiettivo generale è di ottenere raggruppamenti basati sulla somiglianza in modo che gli elementi di un gruppo siano simili tra loro e quelli di gruppi diversi siano diversi tra loro. Questa tecnica viene utilizzata in molteplici discipline, ad esempio, nella finanza viene utilizzata per raggruppare gli investitori in base a preferenze simili e proporre loro investimenti adeguati, mentre nelle scienze sociali viene utilizzata per identificare gruppi di individui con tendenze sociali simili.

Ci sono due problemi principali nell'analisi dei cluster:

- La standardizzazione o meno delle variabili
- Il numero di caratteristiche da utilizzare.

La standardizzazione è importante in molti metodi di clustering, ma dipende dalla situazione specifica, serve a normalizzare le variabili o le caratteristiche in modo che abbiano una scala comune. La standardizzazione più comune consiste nell'utilizzare la media campionaria e la deviazione standard campionaria per ogni variabile. In pratica, ciò significa che ogni valore per ogni variabile viene sottratto dalla media campionaria e diviso per la deviazione standard campionaria. Il risultato è una variabile standardizzata che ha media uguale a 0 e deviazione standard uguale a 1. Questa tecnica è utile perché consente di comparare variabili su una scala comune, tuttavia non è sempre conveniente standardizzare. Inoltre, le tecniche di clustering sono computazionalmente onerose, dunque il numero di variabili da utilizzare è importante. È necessario selezionare solo le variabili più direttamente collegate al fenomeno osservato, per evitare tempi di calcolo lunghi e per farlo viene utilizzata la PCA.

Gli algoritmi di clustering (o di partizionamento) sono usualmente suddivisi tra approcci gerarchici e non gerarchici; gli algoritmi gerarchici sono a loro volta suddivisi tra metodi agglomerativi e divisivi. I metodi gerarchici operano aggregando (o separando) tutti gli elementi sequenzialmente. I metodi agglomerativi inizialmente hanno n cluster da n individui; in seguito, vengono aggregati in successione i cluster più "simili", fino ad ottenere un unico cluster da n individui. Gli algoritmi di clustering divisivi operano in maniera opposta: iniziano con unico cluster da n elementi; dividono quindi questo cluster in due cluster separati, e via di seguito; al termine della procedura ciascun individuo formerà il "suo" cluster.

In questa parte del corso vedremo quindi:

1. Algoritmi di **Clustering Gerarchico (Agglomerativo) (HC)**;
2. Algoritmi di **Clustering Non-Gerarchici (K-Means) (NHC)**;

Cluster gerarchico

Il punto di partenza di ogni metodo di Clustering Gerarchico (HC), è il calcolo della matrice di distanza o di similarità di ciascun individuo rispetto a tutti gli altri. Quale definizione di distanza usare (euclidea, Manhattan, Canberra, ecc.), spesso dipende dall'applicazione specifica o è una scelta soggettiva.

Distanza e similarità

La somiglianza può essere definita mediante un coefficiente di similarità $sij = s(Xi, Xj)$ oppure mediante una misura di distanza $dij = d(Xi, Xj)$ tra due individui Ii e Ij .

Mentre i coefficienti di similarità assumono valori compresi tra 0 e 1, le misure di distanza possono assumere qualsiasi valore non negativo, dunque reale maggiore o uguale a zero.

Le misure di similarità permettono di definire in modo quantitativo la somiglianza o differenza tra due individui Ii e Ij , intendendo ovviamente con 0 l'assoluta assenza e con 1 la massima presenza di somiglianza.

Le metriche di somiglianza invece sono soprattutto basate sulle funzioni distanza tra i vettori delle caratteristiche. Una funzione a valori reali $d(Xi, Xj)$ è detta funzione distanza se e soltanto se essa soddisfa le seguenti condizioni:

1. $d(Xi, Xj) = 0$ se e solo se $Xi = Xj$, con Xi e Xj in Ep;
2. $d(Xi, Xj) \geq 0$ per ogni Xi e Xj in Ep;
3. $d(Xi, Xj) = d(Xj, Xi)$ per ogni Xi e Xj in Ep;
4. $d(Xi, Xj) \leq d(Xi, Xk) + d(Xk, Xj)$ per ogni Xi, Xj e Xk in Ep.

La proprietà *1* implica che Xi è a distanza zero da se stesso e che ogni due punti a distanza nulla debbono essere identici.

La proprietà *2* afferma che la funzione distanza è non negativa.

La proprietà *3* impone la simmetria richiedendo che la distanza tra Xi e Xj deve essere la stessa della distanza tra Xj e Xi.

La proprietà *4*, 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.

Uno strumento generale in ***R*** per il calcolo delle distanze è la funzione `dist()`.

Innanzitutto creiamo una partizione selezionando il *10%* delle osservazioni del campione totale:

```
set.seed("7")
p_mpg <- createDataPartition(mpg$class, p = 0.1, list = FALSE)
part_mpg <- mpg[p_mpg, c(3,5,8,9) ]
part_mpg
```

```
## # A tibble: 28 x 4
##    displ   cyl   cty   hwy
##    <dbl> <int> <int> <int>
##  1   1.8     4    21    29
##  2   4.2     8    16    23
##  3   5.7     8    13    17
##  4   5.7     8    15    23
##  5   3.6     6    17    26
##  6   3.3     6    17    24
##  7   3.8     6    15    22
##  8   5.2     8    11    17
##  9   4.7     8    13    17
## 10   5.9     8    11    15
## # ... with 18 more rows
```

Mediante lo scalamento e la standardizzazione si ottengono dei nuovi dati le cui medie campionarie sono nulle e le varianze campionarie unitarie.

```
etichette_cluster <- mpg[p_mpg[,1],1]
```

```
mpg_std <- scale(part_mpg,center = TRUE,scale = TRUE)
rownames(mpg_std) <- as.list(etichette_cluster)$manufacturer
mpg_std
```

```
##                  displ         cyl         cty         hwy
## audi       -1.46775252 -1.31746510  0.67659764  0.62642230
## audi        0.34535353  1.14180309 -0.24304964 -0.17692844
## chevrolet   1.47854481  1.14180309 -0.79483800 -0.98027918
## chevrolet   1.47854481  1.14180309 -0.42697909 -0.17692844
## chevrolet  -0.10792298 -0.08783101 -0.05912018  0.22474693
## dodge      -0.33456124 -0.08783101 -0.05912018 -0.04303665
## dodge       0.04316919 -0.08783101 -0.42697909 -0.31082023
## dodge       1.10081439  1.14180309 -1.16269691 -0.98027918
## dodge       0.72308396  1.14180309 -0.79483800 -0.98027918
## dodge       1.62963698  1.14180309 -1.16269691 -1.24806276
## dodge       0.72308396  1.14180309 -0.79483800 -0.98027918
## ford        0.19426136 -0.08783101 -0.42697909 -0.71249560
## ford        0.19426136 -0.08783101 -0.61090855 -0.98027918
## ford        1.25190656  1.14180309 -0.79483800 -0.98027918
## ford        0.64753787  1.14180309 -0.42697909 -0.31082023
## honda      -1.46775252 -1.31746510  1.59624491  1.29588125
## hyundai    -0.33456124 -0.08783101  0.30873873  0.49253051
## hyundai    -1.31666035 -1.31746510  0.30873873  0.62642230
## lincoln     1.25190656  1.14180309 -1.16269691 -0.98027918
## pontiac     0.04316919 -0.08783101 -0.24304964  0.22474693
## pontiac     0.04316919 -0.08783101  0.12480927  0.49253051
## subaru     -0.93892992 -1.31746510  0.49266818  0.35863872
## toyota      0.19426136 -0.08783101 -0.24304964 -0.57860381
## toyota     -0.33456124 -0.08783101  0.12480927  0.35863872
## toyota     -1.46775252 -1.31746510  1.59624491  1.42977304
## toyota     -0.78783775 -1.31746510 -0.42697909 -0.57860381
## volkswagen -1.39220643 -1.31746510  2.88375109  2.63479915
## volkswagen -1.39220643 -1.31746510  2.14803327  2.23312378
## attr(,"scaled:center")
##     displ       cyl       cty       hwy
##  3.742857  6.142857 17.321429 24.321429
## attr(,"scaled:scale")
##    displ      cyl      cty      hwy
## 1.323695 1.626500 5.436867 7.468718
```

- Il parametro center = True indica che alla colonna si sottrae il valore medio della corrispondente colonna.
- Il parametro scale = True divide la matrice per la deviazione standard campionaria

Alla matrice *mpg_std* sono stati sostituiti agli indici delle righe i nomi dei produttori di auto

In seguito calcoliamo le metriche di distanze.

Euclidea

```
## Metrica Euclidea
d <- dist(mpg_std,diag = TRUE, upper = TRUE)
```

La distanza Euclidea è una funzione di distanza molto comune e utilizzata in molte applicazioni di clustering e altre tecniche di analisi dei dati. Matematicamente, la distanza Euclidea tra due punti p e q è data da:

$d(p,q) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + ... + (q_n - p_n)^2)}$

La matrice generata ci fornisce n(n-1)/2 distanze, dove n è la numerosità del campione.

Altre metriche

```
## Manhattan
dMh <- dist(mpg_std,method = "manhattan", diag = TRUE, upper = TRUE)

## Chebycev
dCb <- dist(mpg_std,method = "maximum", diag = TRUE, upper = TRUE)

## Minkowski
dMski <- dist(mpg_std,method = "minkowski",4, diag = TRUE, upper = TRUE)

## Canberra
dCa <- dist(mpg_std,method = "canberra", diag = TRUE, upper = TRUE)
```

La distanza di Manhattan, chiamata anche metrica "L-1", è calcolata come la somma delle differenze assolute tra le coppie di valori di due punti. Questa distanza è utile quando le dimensioni sono indipendenti.

$$d_1(X_i, X_j) = \sum_{k=1}^{p} |x_{ik} - x_{jk}|$$

La distanza di Chebycev è anche nota come distanza "L-infinito" e si basa sulla differenza massima tra le coppie di valori dei due punti. La metrica di Chebycev coinvolge anche una procedura di ordinamento.

$$d_\infty(X_i, X_j) = \max_{k=1,2,...,p} |x_{ik} - x_{jk}|$$

La distanza di Minkowski ha come caso particolare le distanze di Manhattan e di Euclidea, dove il parametro "p" determina se la distanza è più simile a Manhattan (p = 1) o a Euclidea (p = 2). Questa distanza è utile per adattarsi a situazioni dove le dimensioni sono importanti e le relazioni tra le dimensioni sono complesse.

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

La distanza di Canberra è una misura di dissimilarità tra due punti e si basa sulla somma delle differenze normalizzate tra le coppie di valori. Questa distanza è utile quando le dimensioni hanno scale diverse e non è possibile utilizzare la distanza Euclidea, perché è adimensionale. Però se uno dei due valori è uguale a zero la distanza totale è automaticamente 1, ossia la massima, e ciò non ha senso.

$$d_c(X_i, X_j) = \sum_{k=1}^{p} \frac{|x_{ik} - x_{jk}|}{|x_{ik}| + |x_{jk}|}$$

Similarità

Una funzione di similarità, o coefficiente di similarità, fornisce un valore numerico compreso tra 0 e 1 che quantifica la somiglianza o la differenza tra due punti del dataset. Questa funzione deve soddisfare tre proprietà:

- deve essere unitaria quando i due punti sono identici,
- compresa tra 0 e 1, e
- simmetrica (cioè la similarità tra i punti X1 e X2 deve essere la stessa della similarità tra i punti X2 e X1).

La funzione di similarità viene rappresentata come una matrice S con s_{ij} come elemento nella riga i-esima e colonna j-esima.

In alternativa, si può utilizzare una funzione di distanza, che misura la separazione tra i punti invece che la loro somiglianza. La trasformazione da una funzione di distanza a una funzione di similarità è sempre possibile, ma il contrario non è sempre vero a causa della disuguaglianza triangolare.

Un esempio di metrica di similarità è il coefficiente di Jaccard, che misura la somiglianza tra due collezioni di elementi. Il coefficiente di Jaccard tra due insiemi A e B è definito come:

$s(A, B) = |A \cap B|/|A \cup B|$

Dove $|A \cap B|$ è la cardinalità dell'intersezione tra gli insiemi A e B, e $|A \cup B|$ è la cardinalità dell'unione tra gli insiemi A e B. Questa metrica di similarità assume valori compresi tra 0 e 1, dove 0 significa assenza di somiglianza e 1 significa massima somiglianza.

Misure di non omogeneità statistica

Le misura di non omogeneità statistica sono metriche che vengono utilizzate in varie applicazioni statistiche e di intelligenza artificiale, come appunto i cluster, ma anche la PCA e nel machine learning.

Covarianza

Il calcolo delle misure di non omogeneità passano per la matrice di covarianza.

```
#il set di dati mpg_std è standardizzato
WI <- cov(mpg_std)
WI
```

```
##            displ        cyl        cty        hwy
## displ  1.0000000  0.9500750 -0.8351794 -0.8177630
## cyl    0.9500750  1.0000000 -0.7760244 -0.7478394
## cty   -0.8351794 -0.7760244  1.0000000  0.9723941
## hwy   -0.8177630 -0.7478394  0.9723941  1.0000000
```

La matrice di covarianza è una matrice quadrata che descrive la relazione lineare tra le diverse colonne di una matrice. Se la covarianza è positiva, significa che le due variabili tendono ad aumentare o diminuire insieme. Se la covarianza è negativa, significa che una variabile tende ad aumentare mentre l'altra tende a diminuire. Se la covarianza è vicina a zero, significa che le due variabili sono scarsamente correlate.

Ad esempio, la covarianza tra "displ" e "cyl" è 0.94, il che significa che le due variabili sono fortemente correlate. La covarianza tra "displ" e "cty" è -0.82, il che significa che le due variabili sono negativamente correlate.

Grazie alla matrice di covarianza calcoliamo la SSM, statistical scatter matrix (matrice statistica di non omogeneità), moltiplicandola per (n-1).

```
n = nrow(mpg_std)
HI <-(n-1)*WI
HI
```

```
##           displ       cyl       cty       hwy
## displ  27.00000  25.65203 -22.54984 -22.07960
## cyl    25.65203  27.00000 -20.95266 -20.19166
## cty   -22.54984 -20.95266  27.00000  26.25464
## hwy   -22.07960 -20.19166  26.25464  27.00000
```

Dalla matrice di non omogeneità otteniamo attraverso la traccia, ossia la somma della diagonale principale, la misura di non omogeneità.

```
trHI <-sum(diag(HI))
trHI
```

```
## [1] 108
```

Possiamo ottenere la stessa misura andando a sommare il quadrato della distanza euclidea dei punti, oppure andando a calcolare (n-1) per la somma della varianza delle colonne.

Questa misure è riferita a tutti gli elementi della popolazione, ad affiancarla c'è la misura di non omogeneità all'interno dei cluster, within, e la misura di non omogeneità all'esterno dei cluster, between.

L'obiettivo di queste misure è capire se il metodo utilizzato ha portato alla creazione di gruppi che hanno internamente individui simili tra loro ma che c'è dissimilarità all'esterno dei gruppi.

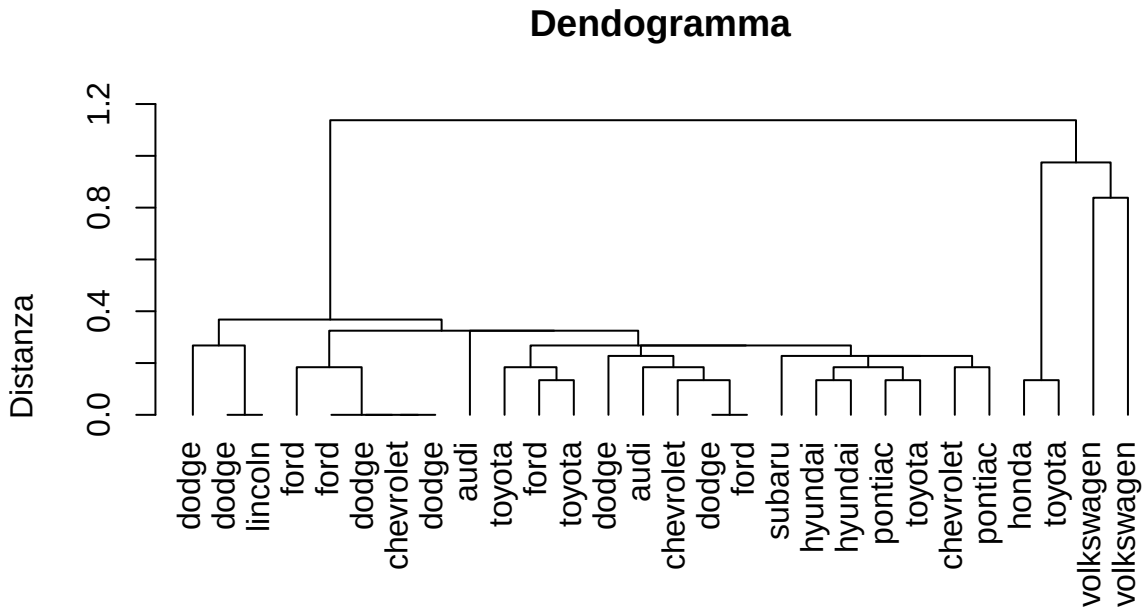
Metodi gerarchici

I metodi gerarchici sono algoritmi di clustering utilizzati per suddividere un insieme di elementi in gruppi omogenei in base alla loro somiglianza o dissomiglianza. Questi metodi creano una struttura ad albero gerarchica, chiamata dendrogramma, in cui i gruppi sono organizzati in modo gerarchico, in base alla loro vicinanza o distanza tra loro. Tali metodi sono distinti in *agglomerativi* che procedono per aggregazioni successive delle unità partendo da n gruppi formati da un singolo individuo e *divisivi* che partono da un solo gruppo formato da tutte le unità e procedono a divisioni successive fino a giungere a gruppi formati da una sola unità.

Il dendrogramma è una struttura ad albero utilizzata per rappresentare graficamente la sequenza di partizioni. Sull'asse delle ascisse sono rappresentate le partizioni, e sull'asse delle ordinate le distanze. Attraverso il dendrogramma possiamo ottenere una visione globale della divisione dei cluster ed è utile perché ci permette di scegliere dove tagliare per ottenere la partizione dell'insieme di individui in cluster.

```
mpg_2 <- mpg_std[,c("cty","hwy")]
d_2 <- dist(mpg_2, method = "euclidean")
hls <- hclust(d_2, method = "single")
plot(hls,  hang = -2,
```

```
    xlab=" Metodo gerarchico agglomerativo ",
    sub ="del legame singolo ",
    ylab = "Distanza",
    main = "Dendogramma")
```



L'obiettivo finale dei metodi gerarchici non è quello di ottenere una singola partizione degli n individui di partenza, ma di ottenere una sequenza di partizioni che possono essere rappresentate graficamente proprio con il dendrogramma.

Come possiamo vedere dal comando *hclust* va inserito un metodo, tutti i metodi funzionano pressoché allo stesso modo, che possiamo riassumere in 5 step:

1. Prendere una matrice X dei dati, e scegliere se scalarla o meno. Scegliere il metodo per calcolare la matrice delle distanze, D, o la matrice di similarità, S.
2. Prendere la distanza minore nella matrice delle distanze D, raggruppare in un cluster e ricalcolare la matrice delle distanze con il nuovo cluster.
3. Costruire una nuova matrice delle distanze con una riga ed una colonna in meno
4. Rifare dal passo 2 sulla nuova matrice fino ad ottenere una matrice 2x2, arrivando ad n-1 iterazioni
5. Rappresentare il processo mediante il dendrogramma

In questo algoritmo l'unica cosa che varia è il passo 1 e 2, ovviamente nel passo 1 a cambiare è la metrica della distanza, nel punto 2 invece a cambiare è il metodo con la quale si calcolano le distanza, ed è ciò che differenzia i vari metodi di clustering gerarchico.

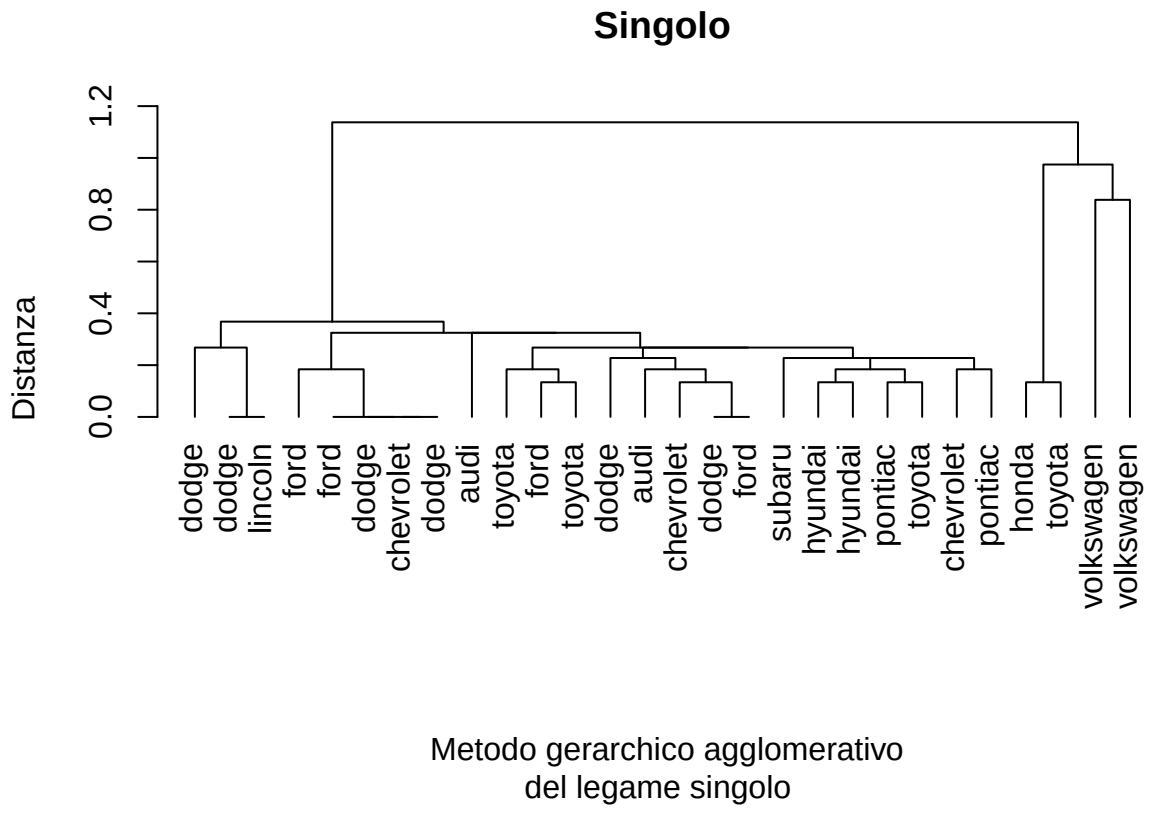
Metodo del legame singolo

Il metodo del legame singolo, o nearest neighbour method, è un algoritmo di clustering gerarchico che si basa sulla distanza minima tra tutte le possibili coppie di individui appartenenti a due gruppi. In questo metodo può essere utilizzata qualsiasi metrica di distanza. Come detto nell'algoritmo, calcolata la matrice D, prendiamo il punto con distanza minore e uniamo i due cluster. Una volta fatto ciò la nuova matrice delle distanze la calcoliamo come:

$$d_{((ij),k)} = min(d_{(i,k)}, d_{(j,k)})$$

Dove i e j indicano i due elementi che sono stati uniti e k indica uno degli altri con la quale calcolare la distanza.

```
single_hls <- hclust(d_2, method = "single")
plot(single_hls,  hang = -2,
     xlab=" Metodo gerarchico agglomerativo",
     sub ="del legame singolo ",
     ylab = "Distanza",
     main = "Singolo")
```



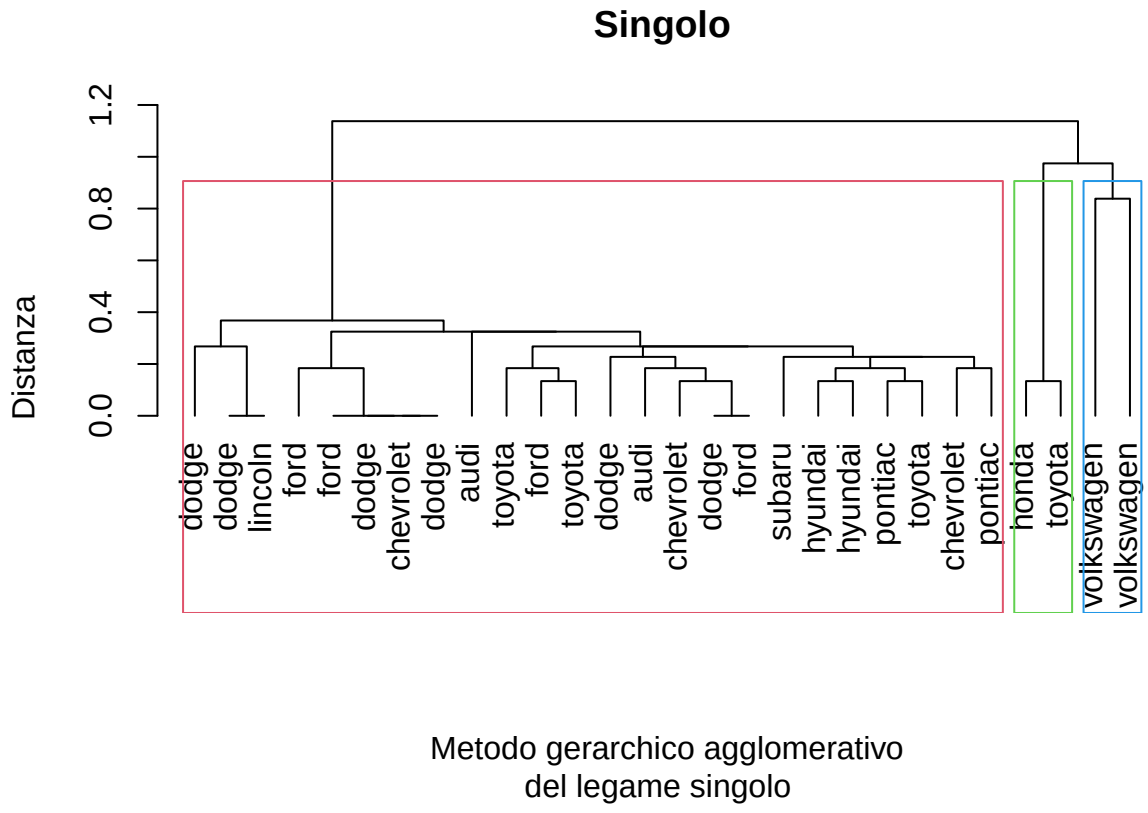
- Vantaggi: Consente di individuare outliers facilmente e di individuare gruppi di qualsiasi forma, tipo forme allungate.
- Svantaggi: Iniziano a formarsi cluster a catena e quindi si generano gruppi dissimili internamente tra loro

Dall'analisi dei cluster non gerarchici abbiamo visto che la partizione in 3 gruppi è ottimale, ora calcoliamo attraverso i vari legami le misure di non omogeneità statistica e vediamo cosa otteniamo dal confronto di ognuno di loro.

```
cut_single <- cutree(single_hls, k = 3)
```

Sulla base del dendrogramma proviamo a dividere il cluster in 3 gruppi, per farlo utilizziamo la funzione "**cutree**".

```
plot(single_hls,hang = -2,
     xlab=" Metodo gerarchico agglomerativo",
     sub ="del legame singolo ",
     ylab = "Distanza", main = "Singolo")

rect.hclust(single_hls , k = 3, border = 2:6)
```



Visualizziamo i 5 gruppi scelti sul dendrogramma, evidenziati da tre colori differenti.

```
sclus_1 = data.frame()
sclus_2 = data.frame()
sclus_3 = data.frame()

for (i in 1:nrow(mpg_2)) {
  if (cut_single[i]==1) {
```

```
    sclus_1 <- rbind(sclus_1, mpg_2[i,])
  }
  if (cut_single[i]==2) {
    sclus_2 <- rbind(sclus_2, mpg_2[i,])
  }
  if (cut_single[i]==3) {
    sclus_3 <- rbind(sclus_3, mpg_2[i,])
  }

}
names(sclus_1) <- c("cty","hwy")
names(sclus_2) <- c("cty","hwy")
names(sclus_3) <- c("cty","hwy")
```

Per vedere se la suddivisione dei cluster in 3 gruppi è corretta vediamo cosa ci dicono le statistiche di non omogeneità, per farlo abbiamo creato tre data-frame contenenti gli elementi dei rispettivi cluster.

```
sclust <- mpg_2
ns <- nrow(sclust)
trHs <-(n -1) *sum (apply (sclust,2, var ))
trHs
```

```
## [1] 54
```

In primis calcoliamo la misura di non omogeneità totale andando a determinare la traccia utilizzando il primo metodo visto, ossia moltiplicando (n-1) per la somma della varianza delle colonne.

Sappiamo dunque che la misura di non omogeneità totale è 54

Calcoliamo adesso la misura di non omogeneità dei tre cluster per poi poterci calcolare la misura di non omogeneità interna ai cluster ossia within.

```
Ws1 <- cov(sclus_1)
ns1 <- nrow(sclus_1)
Hs1 <- (ns1-1)*Ws1

if (ns1 > 1) {
  trHs1 <-sum(diag(Hs1))
}else{
  trHs1 <- 0
}
trHs1
```

```
## [1] 15.10049
```

La misura di non omogeneità statistica del primo cluster è:

```
15.10049
```

Questo cluster ha la misura di non omogeneità statistica più alta di tutti perchè è il gruppo che contiene più elementi al suo interno.

```
Ws2 <- cov(sclus_2)
ns2 <- nrow(sclus_2)
Hs2 <- (ns2-1)*Ws2
Hs2 <- (ns2-1)*Ws2
if (ns2 > 1) {
  trHs2 <-sum(diag(Hs2))
}else{
  trHs2 <- 0
}

trHs2
```

```
## [1] 0.008963506
```

La misura di non omogeneità statistica del secondo cluster è:

```
 0.008963506
```

```
Ws3 <- cov(sclus_3)
ns3 <- nrow(sclus_3)
Hs3 <- (ns3-1)*Ws3
if (ns3 > 1) {
  trHs3 <-sum(diag(Hs3))
}else{
  trHs3 <- 0
}
trHs3
```

```
## [1] 0.3513119
```

E la misura di non omogeneità statistica del terzo cluster è:

```
0.3513119
```

```
trWithin <- trHs1 +trHs2 + trHs3
trWithin
```

```
## [1] 15.46077
```

```
trBetween <- trHs - trWithin
trBetween
```

```
## [1] 38.53923
```

Essendo che la misura within è più piccola della between, possiamo dire che la suddivisione in tre cluster, applicando il metodo del legame singolo, è soddisfacente

```
s_perc <- as.character(round((trBetween/trHs),4)*100)
paste(s_perc,"%",sep="")
```

```
## [1] "71.37%"
```

Il rapporto tra la misura di non omogeneità tra cluster between e la traccia della matrice di non omogeneità è del **71.37%**, dunque abbastanza alta.

Controlliamo se abbiamo svolto bene i calcoli guardando se la misura di non omogeneità relativa all'unione dei cluster è uguale alla misura di non omogeneità totale

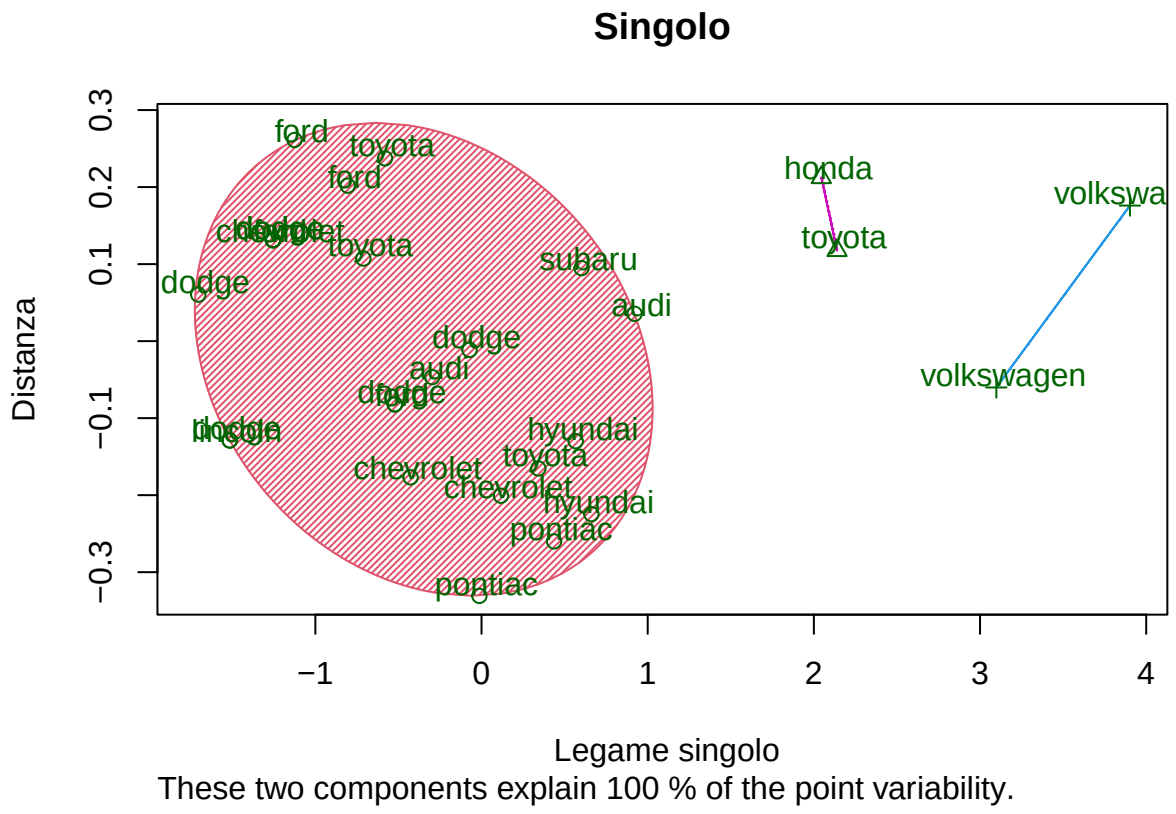
```
as.integer(round(trWithin+trBetween), digits = 0) == as.integer(round(trHs, digits = 0))
```

```
## [1] TRUE
```

Ed infatti ci restituisce vero il confronto.

Visualizziamo con un diagramma di Venn i tre cluster e i punti appartenenti agli insiemi

```
library(cluster)
clusplot(mpg_2, cut_single, color=TRUE,
         shade=TRUE, labels=3,
         lines=0, xlab=" Legame singolo",
         ylab = "Distanza", main = "Singolo")
```



Dal diagramma possiamo notare che vi è un singolo grande gruppo è gli altri 2 invece sono più piccoli, questo potrebbe essere un effetto collaterale del metodo a legame singolo, proprio perché viene a crearsi un effetto a catena che lega elementi dissimili perché si guarda solo alla distanza minima.

Inoltre possiamo vedere che la forma del cluster rosso è molto ovale e non ellissoidale, proprio perché con il metodo del legame singolo si possono creare cluster di forma allungata.

Possiamo notare che due auto volkswagen formano un singolo cluster che si differenzia dagli altri.

Metodo del legame completo

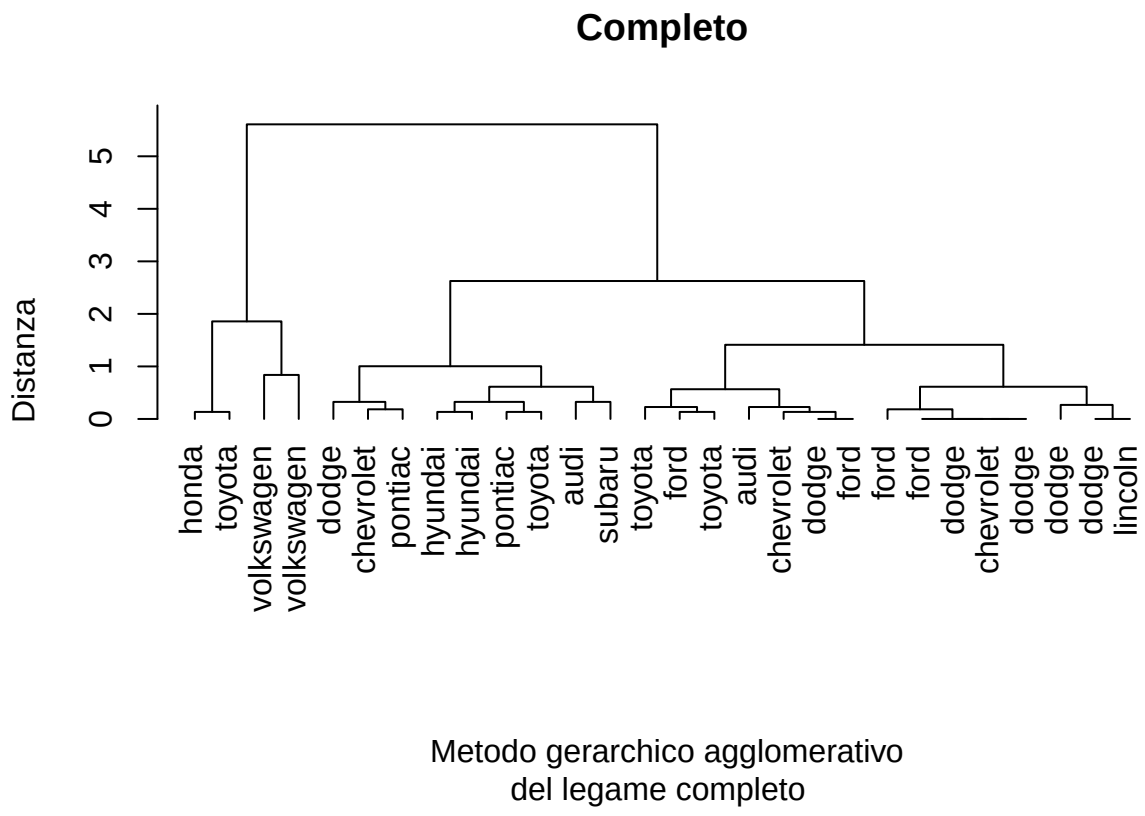
Il metodo del legame completo, utilizzato nell'algoritmo di clustering gerarchico, definisce la distanza tra due cluster come la massima tra tutte le distanze possibili tra ogni individuo del primo cluster e ogni individuo del secondo cluster. Questo metodo tende a formare gruppi di forma ellissoidale, privilegiando l'omogeneità all'interno dei gruppi a discapito della differenziazione tra gruppi.

Anche in questo caso, come dice l'algoritmo, prendiamo il minimo nella matrice delle distanze D e creiamo il primo cluster, successivamente calcoliamo la nuova matrice delle distanze, con questo nuovo gruppo, ed è questo che differenzia il metodo:

$$d_{(ij),k)} = max(d_{(i,k)}, d_{(j,k)})$$

Dove i e j indicano i due elementi che sono stati uniti e k indica uno degli altri con la quale calcolare la distanza.

```
complete_hls <- hclust(d_2, method = "complete")
plot(complete_hls,  hang = -2,
     xlab=" Metodo gerarchico agglomerativo",
     sub ="del legame completo ",
     ylab = "Distanza", main = "Completo")
```



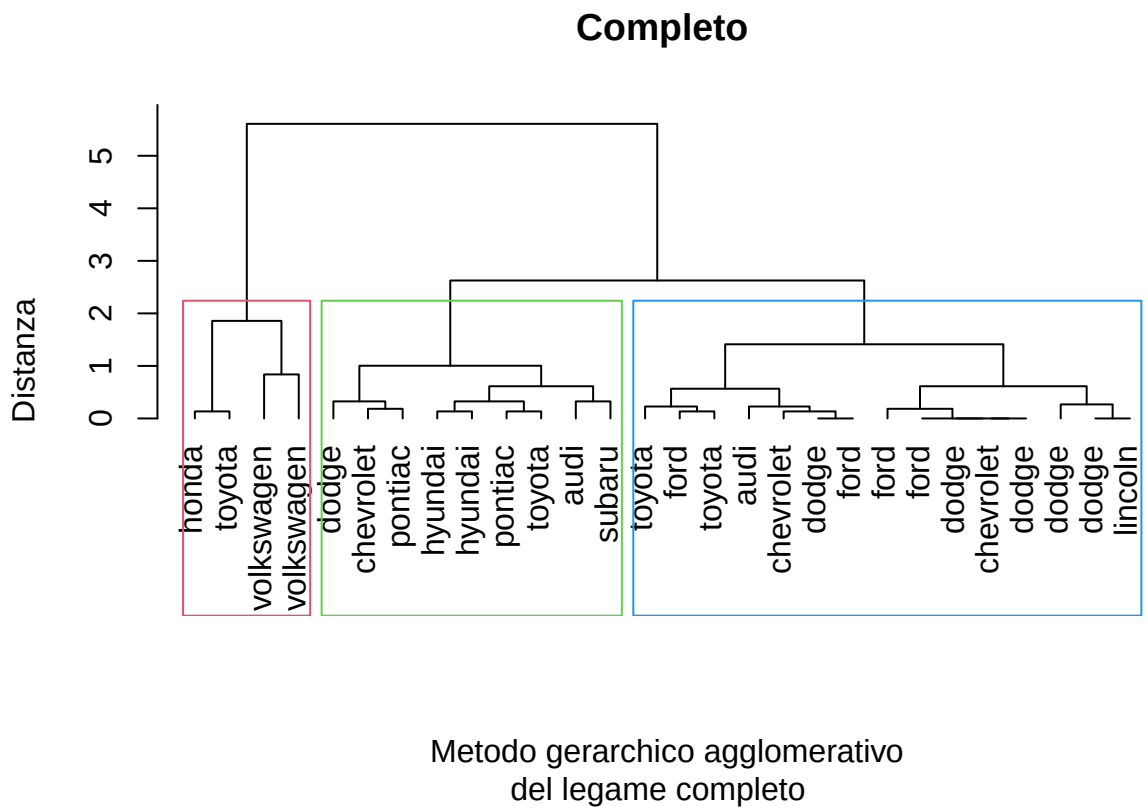
Il dendrogramma ottenuto con il legame completo ha rami più lunghi rispetto al dendrogramma ottenuto con il legame singolo, poiché i gruppi si formano a livelli di distanza maggiori. Infatti la scala delle ordinate va da 0 a 6 e non da 0 a 1.5 come nel metodo del legame singolo.

- Vantaggi: Utilizzato per identificare gruppi che si addensano intorno al nucleo. Questo a vantaggio di una maggiore omogeneità tra i gruppi, within, a discapito di una peggiore all'esterno dei gruppi, between.

Per confrontare i cluster tra loro divideremo tutti in 3 gruppi, quindi procediamo a farlo attraverso il metodo del legame completo

Vediamo come appare il dendrogramma selezionando tre cluster

```
complete_hls <- hclust(d_2, method = "complete")
plot(complete_hls,  hang = -2,
     xlab=" Metodo gerarchico agglomerativo",
     sub ="del legame completo ",
     ylab = "Distanza", main = "Completo")

rect.hclust(complete_hls , k = 3, border = 2:6)
```



Vediamo se le misure di non omogeneità statistica ci confermano se la divisione in due cluster è corretta

```
cut_complete <- cutree(complete_hls, k = 3)

cclus_1 = data.frame()
cclus_2 = data.frame()
cclus_3 = data.frame()

for (i in 1:nrow(mpg_2)) {
  if (cut_complete[i]==1) {
    cclus_1 <- rbind(cclus_1, mpg_2[i,])
  }
  if (cut_complete[i]==2) {
    cclus_2 <- rbind(cclus_2, mpg_2[i,])
  }
  if (cut_complete[i]==3) {
    cclus_3 <- rbind(cclus_3, mpg_2[i,])
  }
}

names(cclus_1) <- c("cty","hwy")
names(cclus_2) <- c("cty","hwy")
names(cclus_3) <- c("cty","hwy")

cclust <- mpg_2
nc <- nrow(cclust)
trHc <- (n - 1) * sum (apply (cclust, 2, var))
trHc
```

```
## [1] 54
```

```
Wc1 <- cov(cclus_1)
nc1 <- nrow(cclus_1)
Hc1 <- (nc1-1)*Wc1
trHc1 <-sum(diag(Hc1))
trHc1
```

```
## [1] 1.051076
```

```
Wc2 <- cov(cclus_2)
nc2 <- nrow(cclus_2)
Hc2 <- (nc2-1)*Wc2
trHc2 <-sum(diag(Hc2))
trHc2
```

```
## [1] 3.168649
```

```
Wc3 <- cov(cclus_3)
nc3 <- nrow(cclus_3)
Hc3 <- (nc3-1)*Wc3
trHc3 <-sum(diag(Hc3))
trHc3
```

```
## [1] 2.353355
```

```
trWithin <- trHc1 + trHc2+ trHc3
trWithin
```

```
## [1] 6.573081
```

```
trBetween <- trHc - trWithin
trBetween
```

```
## [1] 47.42692
```

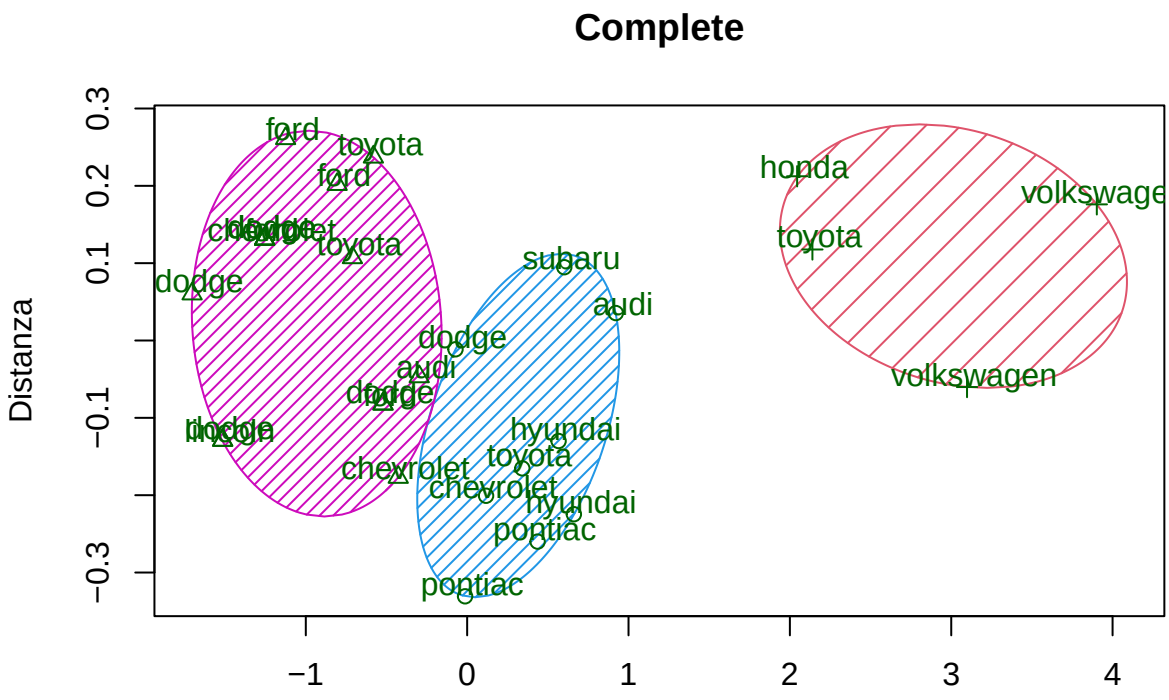
Anche in questo caso la misura di non omogeneità totale tra i cluster, within è minore di quella between, quindi la divisione in questi tre cluster è soddisfacente. Notiamo inoltre che la misura within è piccola della metà della misura del legame singola che vale *15.46*.

```
c_perc <- as.character(round((trBetween/trHc),4)*100)
paste(c_perc,"%",sep="")
```

```
## [1] "87.83%"
```

Infatti come era logico aspettarsi il rapporto tra between e totale è 88% dunque il 10% in più del legame singolo ciò significa che questo metodo massimizza di più la misura di non omogeneità statistica tra i cluster, between, rispetto a quella precedente.

```
clusplot(mpg_2, cut_complete, color=TRUE,
         shade=TRUE, labels=3, lines=0,
         xlab=" Legame completo",
         ylab = "Distanza", main = "Complete")
```



Legame completo
These two components explain 100 % of the point variability.

Notiamo che nel metodo del legame completo i gruppi hanno una forma ellissoidale, inoltre i primi due cluster sono molto distanti dal terzo e ciò si può notare dallo spazio presente nel grafico.

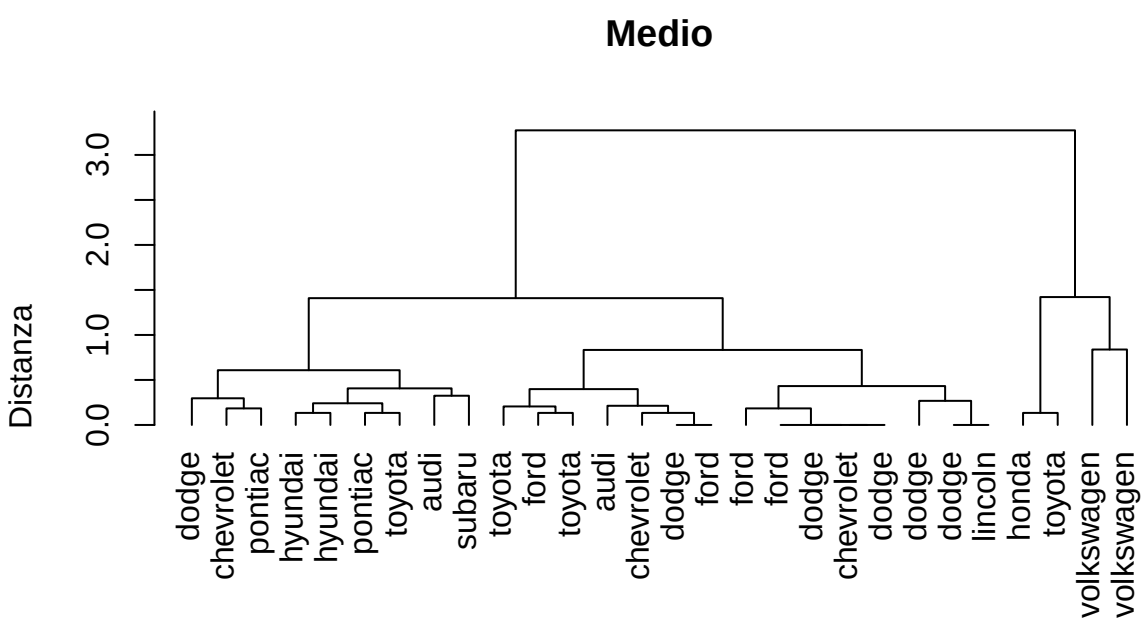
Metodo del legame medio

Il metodo del legame medio è un algoritmo di clustering gerarchico che definisce la distanza tra due gruppi come la media aritmetica delle distanze tra tutte le coppie di unità che compongono i due gruppi.

$$d_{((ij),k)} = \frac{1}{2}(d_{(i,k)}, d_{(j,k)})$$

Nella procedura si parte da un insieme di n cluster e ad ogni passo si uniscono i due cluster più vicini fino ad ottenere un unico cluster formato da tutti gli individui.

```
average_hls <- hclust(d_2, method = "average")
plot(average_hls,  hang = -2,
     xlab=" Metodo gerarchico agglomerativo",
     sub ="del legame medio ",
     ylab = "Distanza",
     main = "Medio")
```

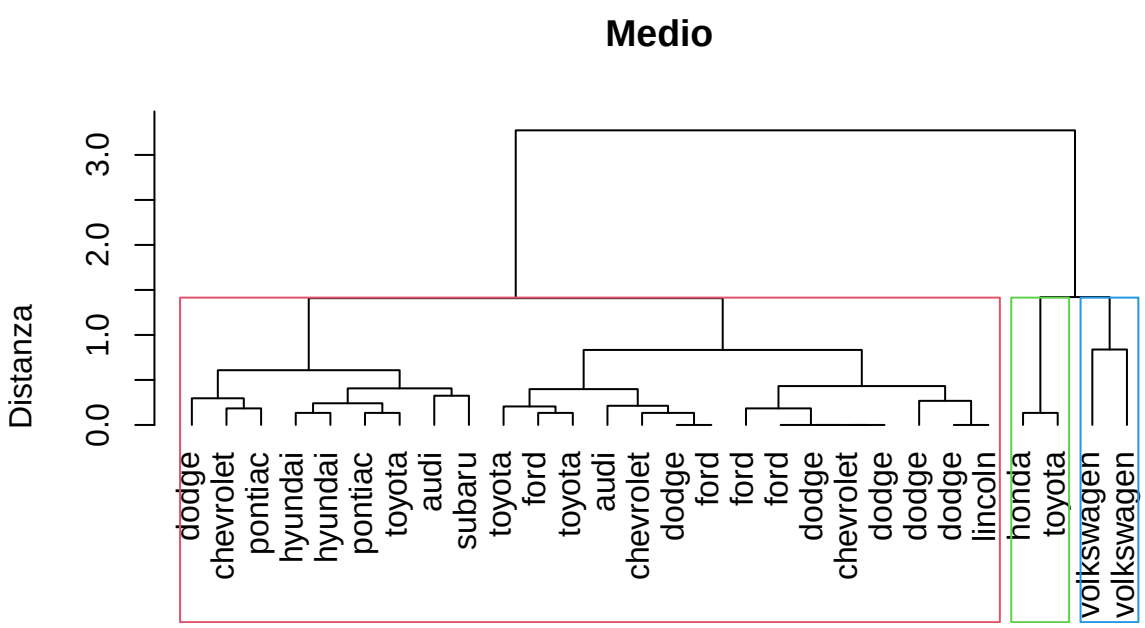


Il dendrogramma ottenuto con il legame medio ha rami più lunghi rispetto al dendrogramma ottenuto con il legame singolo, ma più corti rispetto a quello del legame completo, poiché i gruppi si formano a livelli di distanza medi. Infatti la scala delle ordinate va da 0 a 3 ossia il doppio del legame singolo e la metà del legame completo.

- Vantaggi: Meno sensibile alla presenza di valori anomali, e crea cluster di forma più compatta
- Svantaggi: Capita che anche se le distanze sono differenti l'elemento viene inglobato del cluster più numeroso.

Anche nel caso del metodo a legame medio la divisione avviene per tre gruppi

```
average_hls <- hclust(d_2, method = "average")
plot(average_hls,  hang = -2,
     xlab=" Metodo gerarchico agglomerativo",
     sub ="del legame medio ",
     ylab = "Distanza", main = "Medio")

rect.hclust(average_hls , k = 3, border = 2:6)
```



Metodo gerarchico agglomerativo
del legame medio

Notiamo che la partizione del legame medio è la stessa che otteniamo con il metodo del legame singolo, quindi ci aspettiamo di ottenere gli stessi risultati.

```
cut_average <- cutree(average_hls, k = 3)

aclus_1 = data.frame()
aclus_2 = data.frame()
aclus_3 = data.frame()

for (i in 1:nrow(mpg_2)) {
  if (cut_average[i]==1) {
    aclus_1 <- rbind(aclus_1, mpg_2[i,])
  }
  if (cut_average[i]==2) {
    aclus_2 <- rbind(aclus_2, mpg_2[i,])
  }
  if (cut_average[i]==3) {
    aclus_3 <- rbind(aclus_3, mpg_2[i,])
  }
}

names(aclus_1) <- c("cty","hwy")
names(aclus_2) <- c("cty","hwy")
names(aclus_3) <- c("cty","hwy")

aclust <- mpg_2
```

```
na <- nrow(aclust)
trHa <- (n - 1) * sum (apply (aclust, 2, var))
trHa
```

```
## [1] 54
```

```
Wa1 <- cov(aclus_1)
na1 <- nrow(aclus_1)
Ha1 <- (na1-1)*Wa1
trHa1 <-sum(diag(Ha1))
trHa1
```

```
## [1] 15.10049
```

```
Wa2 <- cov(aclus_2)
na2 <- nrow(aclus_2)
Ha2 <- (na2-1)*Wa2
trHa2 <-sum(diag(Ha2))
trHa2
```

```
## [1] 0.008963506
```

```
Wa3 <- cov(aclus_3)
na3 <- nrow(aclus_3)
Ha3 <- (na3-1)*Wa3
trHa3 <-sum(diag(Ha3))
trHa3
```

```
## [1] 0.3513119
```

```
trWithin <- trHa1 + trHa2 + trHa3
trWithin
```

```
## [1] 15.46077
```

```
trBetween <- trHa - trWithin
trBetween
```

```
## [1] 38.53923
```

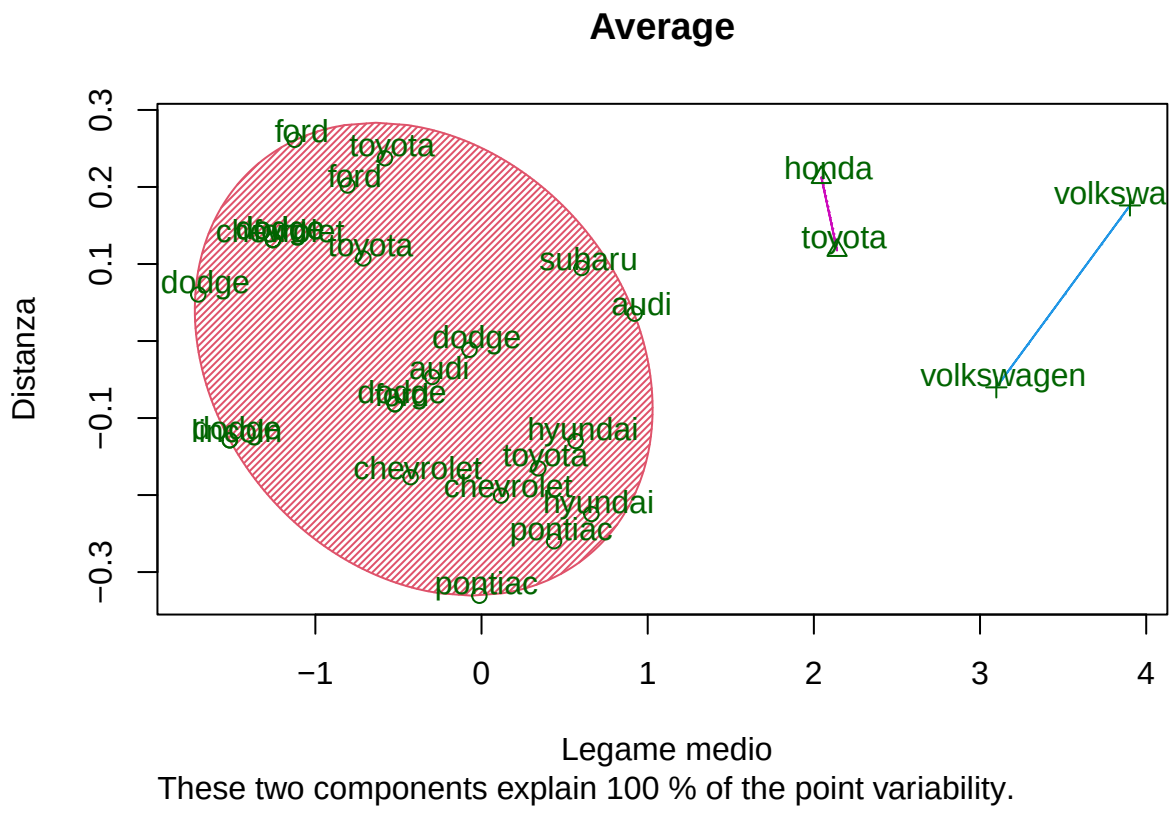
Con il metodo del legame medio la misura within torna ad essere uguale al legame singolo.

```
a_perc <- as.character(round((trBetween/trHa),4)*100)
paste(a_perc,"%",sep="")
```

```
## [1] "71.37%"
```

Il rapporto tra between e totale è *71%* come nel metodo del legame singolo.

```
clusplot(mpg_2, cut_average, color=TRUE,
         shade=TRUE, labels=3, lines=0,
         xlab=" Legame medio",
         ylab = "Distanza", main = "Average")
```



Dal grafico ottenuto deduciamo le medesime cose del legame singolo.

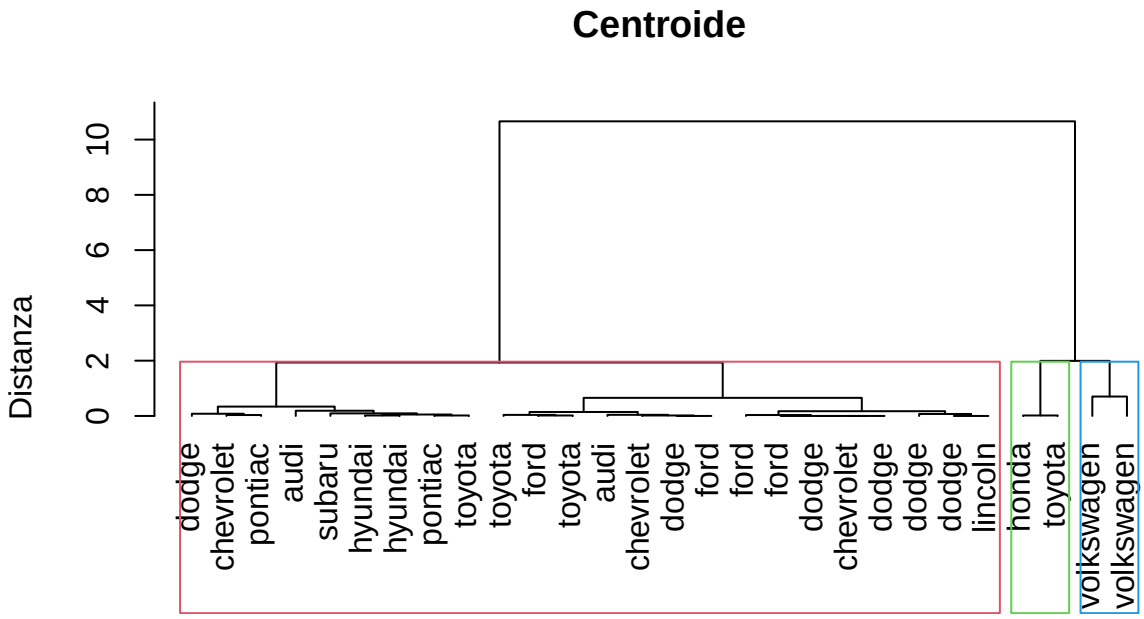
Metodo del centroide

Nel metodo del centroide e nel metodo della mediana si è obbligati ad utilizzare la distanza euclidea. Negli altri metodi poteva invece essere utilizzata qualsiasi altra metrica per calcolare la matrice delle distanze D.

Il metodo del centroide è un algoritmo di clustering gerarchico che calcola la distanza tra due cluster come la distanza tra i loro centroidi, ovvero le medie campionarie calcolate sui dati dei due cluster.

$$d_{((i,j),k} = \frac{1}{2}(d^2_{(i,k)} + d^2_{(j,k)}) - \frac{1}{4}d^2_{i,j}$$

```
d_quadro <- d_2^2
centroide_hls <- hclust(d_quadro, method = "centroid")
plot(centroide_hls,  hang = -2,
     xlab=" Metodo gerarchico agglomerativo",
     sub ="del centroide ", ylab = "Distanza", main = "Centroide")

rect.hclust(centroide_hls , k = 3, border = 2:6)
```



Metodo gerarchico agglomerativo
del centroide

- Vantaggi: Può dare origini a fenomeni gravitazionali, ossia un cluster inizia ad attirare a se gli altri più piccoli. Inoltre le distanze di unione diventano sempre più piccole.
- Svantaggi: Lo stesso del metodo della media, ossia che possono unirsi cluster distanti solo perché un cluster è molto numeroso.

Dal dendrogramma vediamo che la partizione che si viene a creare è la medesima del legame singolo e del legame medio.

```
cut_centroide <- cutree(centroide_hls, k = 3)

clus_1 = data.frame()
clus_2 = data.frame()
clus_3 = data.frame()


for (i in 1:nrow(mpg_2)) {
  if (cut_centroide[i]==1) {
    clus_1 <- rbind(clus_1, mpg_2[i,])
  }
  if (cut_centroide[i]==2) {
    clus_2 <- rbind(clus_2, mpg_2[i,])
  }
  if (cut_centroide[i]==3) {
    clus_3 <- rbind(clus_3, mpg_2[i,])
  }
```

```
}

names(clus_1) <- c("cty","hwy")
names(clus_2) <- c("cty","hwy")
names(clus_3) <- c("cty","hwy")

clust <- mpg_2
n <- nrow(clust)
tr_H <- (n - 1) * sum (apply (clust, 2, var))
tr_H
```

```
## [1] 54
```

```
W_1 <- cov(clus_1)
n_1 <- nrow(clus_1)
H_1 <- (n_1-1)*W_1
tr_H1 <-sum(diag(H_1))
tr_H1
```

```
## [1] 15.10049
```

```
W_2 <- cov(clus_2)
n_2 <- nrow(clus_2)
H_2 <- (n_2-1)*W_2
tr_H2 <-sum(diag(H_2))
tr_H2
```

```
## [1] 0.008963506
```

```
W_3 <- cov(clus_3)
n_3 <- nrow(clus_3)
H_3 <- (n_3-1)*W_3
tr_H3 <-sum(diag(H_3))
tr_H3
```

```
## [1] 0.3513119
```

```
trWithin <- tr_H1 + tr_H2 + tr_H3
trWithin
```

```
## [1] 15.46077
```

```
trBetween <- tr_H - trWithin
trBetween
```

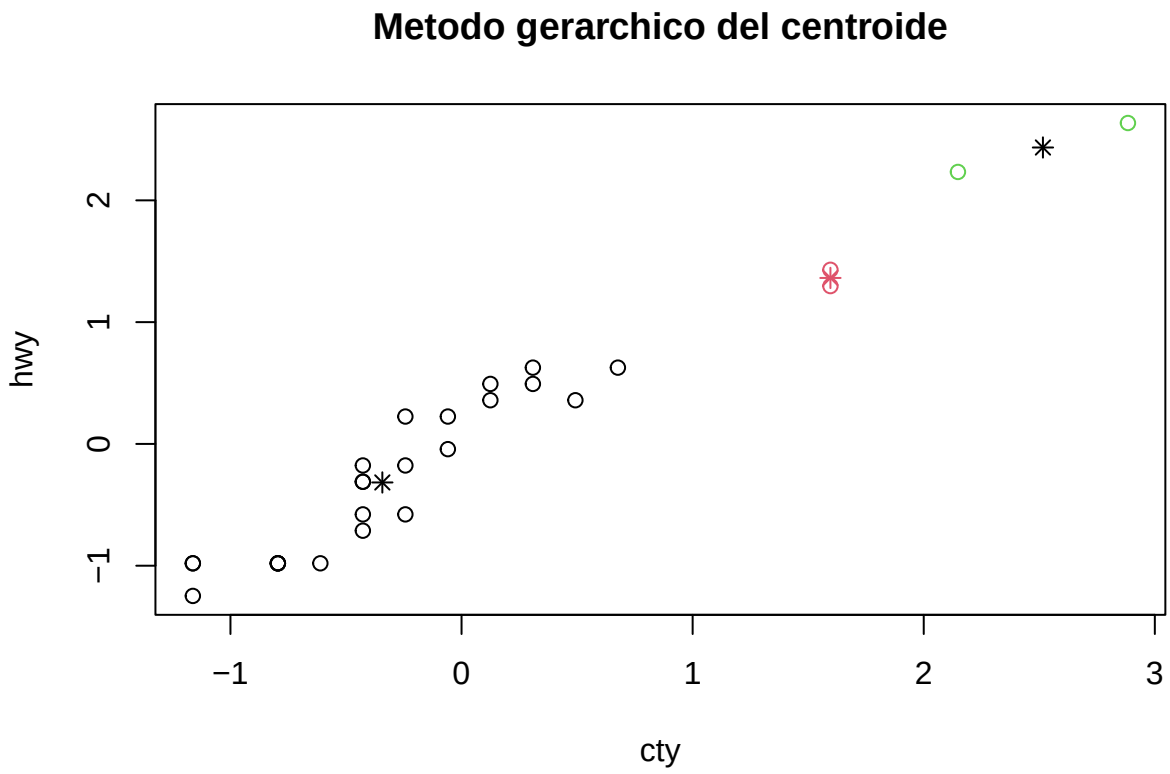
```
## [1] 38.53923
```

Le metriche ottenute ovviamente sono le stesse.

```
centroid_perc <- as.character(round((trBetween/tr_H),4)*100)
paste(centroid_perc,"%",sep="")
```

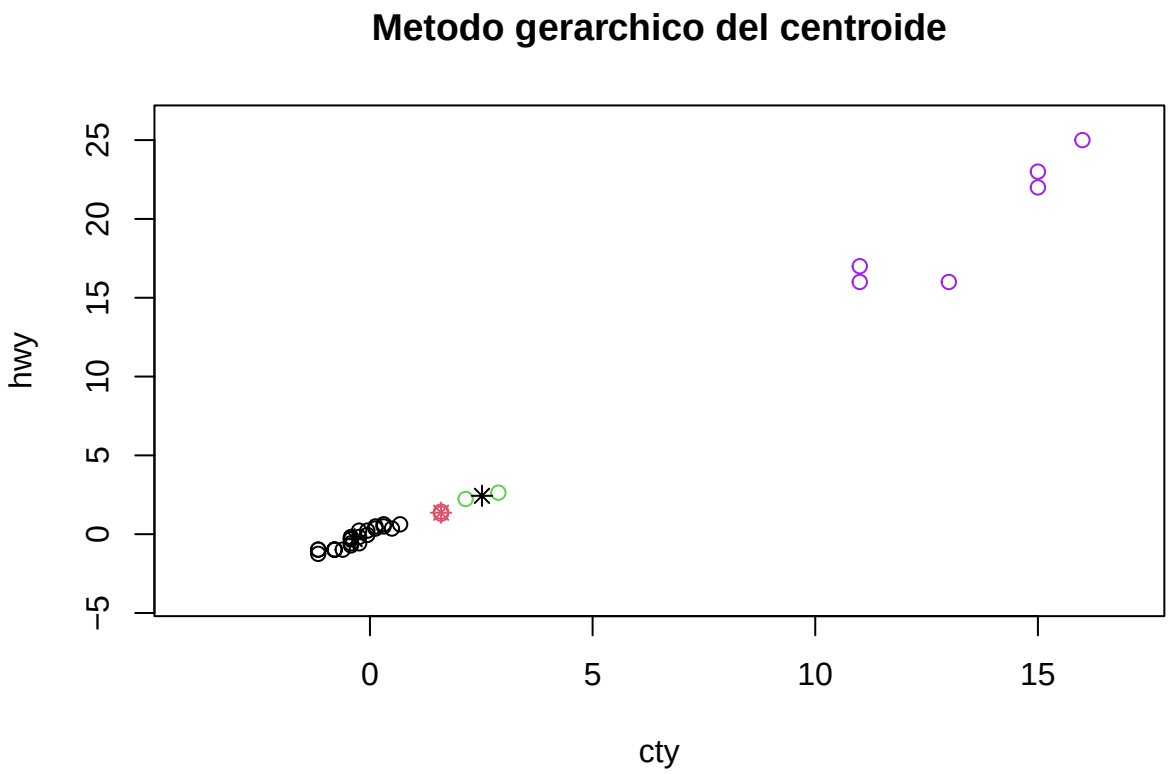
```
## [1] "71.37%"
```

```
cut_centroide_list <- list(cut_centroide)
agmean <- aggregate (mpg_2, cut_centroide_list , mean )[,-1]
plot(mpg_2, col = cut_centroide ,
     main = " Metodo gerarchico del centroide ")
points (agmean ,col = 1:2, pch =8, cex =1)
```



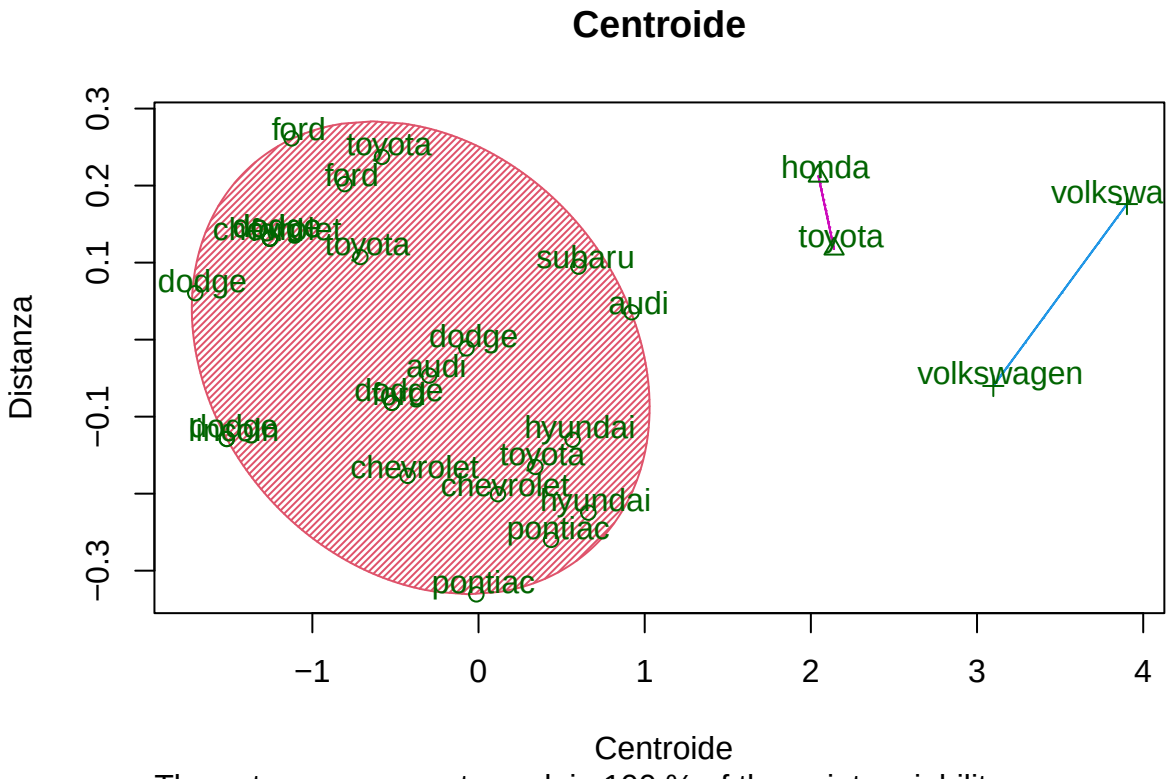
Con questo grafico riusciamo a vedere come si distribuiscono i punti dei vari gruppi, e possiamo vedere che la partizione a tre va bene in quanto si nota il distacco dai gruppi.

```
cut_centroide_list <- list(cut_centroide)
agmean <- aggregate (mpg_2, cut_centroide_list , mean )[,-1]
plot(mpg_2, col = cut_centroide ,
     main = " Metodo gerarchico del centroide ", xlim = c(-4,17),
     ylim = c(-4,26))
points (agmean ,col = 1:2, pch =8, cex =1)
points(outliers[,c(8,9)], col = "purple")
points(outlier[,c(8,9)], col = "purple")
```



In questo grafico abbiamo aggiunto gli outliers individuati durante l'esplorazione dei dati attraverso i boxplot costruiti precedentemente e possiamo notare che nella partizione non sono presenti.

```
clusplot(mpg_2, cut_centroide, color=TRUE,
         shade=TRUE, labels=3, lines=0,
         xlab=" Centroide", ylab = "Distanza", main = "Centroide")
```

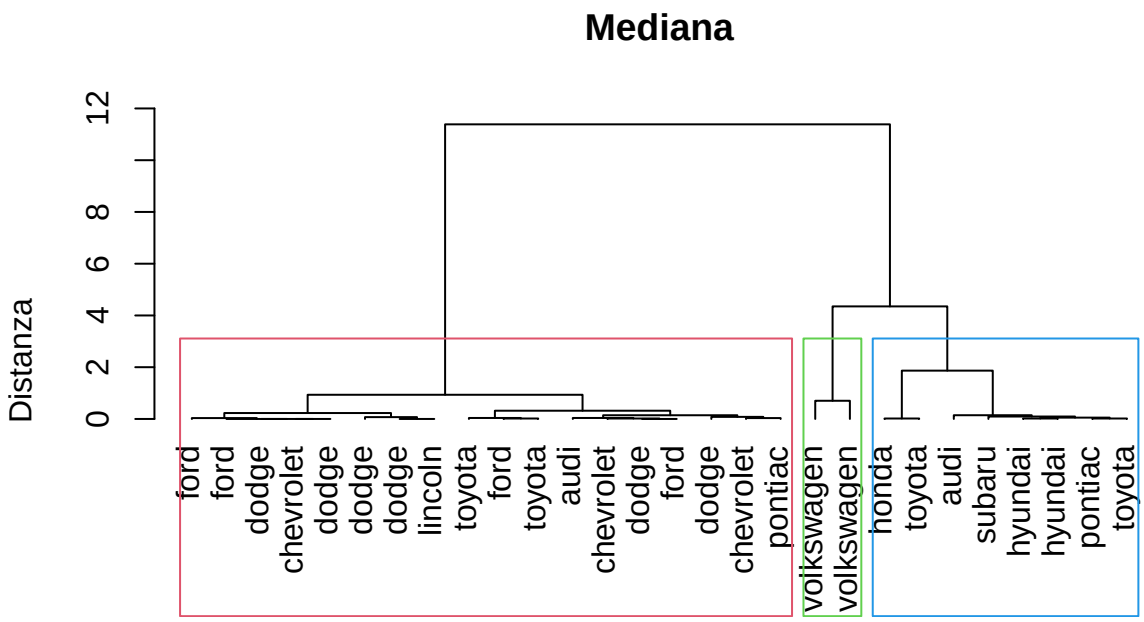


Proprio come il metodo del legame singolo e del legame medio si è formato un fenomeno gravitazionale che ha unito degli elementi dissimili tra loro in un unico cluster.

Metodo della mediana

Il metodo della mediana è simile a quello del centroide, con la differenza che la procedura è indipendente dalla numerosità dei cluster, quando due gruppi si aggregano il nuovo centroide è calcolato come la semisomma dei due centroidi precedenti.

```
Median_hls <- hclust(d_quadro, method = "median")
plot(Median_hls,  hang = -2, xlab=" Metodo gerarchico agglomerativo",
     sub ="della mediana ", ylab = "Distanza", main = "Mediana")

rect.hclust(Median_hls , k = 3, border = 2:6)
```



Metodo gerarchico agglomerativo
della mediana

- Svantaggi: Come il metodo del legame singolo, da formazione a catene, quindi ad un certo livello di distanza può capitare che nel cluster vengono uniti individui piuttosto dissimili

```
cut_mediana <- cutree(Median_hls, k = 3)

mclus_1 = data.frame()
mclus_2 = data.frame()
mclus_3 = data.frame()


for (i in 1:nrow(mpg_2)) {
  if (cut_mediana[i]==1) {
    mclus_1 <- rbind(mclus_1, mpg_2[i,])
  }
  if (cut_mediana[i]==2) {
    mclus_2 <- rbind(mclus_2, mpg_2[i,])
  }
  if (cut_mediana[i]==3) {
    mclus_3 <- rbind(mclus_3, mpg_2[i,])
  }
}

names(mclus_1) <- c("cty","hwy")
names(mclus_2) <- c("cty","hwy")
names(mclus_3) <- c("cty","hwy")
```

```r
mclust <- mpg_2
nm <- nrow(mclust)
trmH <- (nm - 1) * sum (apply (mclust, 2, var))
trmH
```

```
## [1] 54
```

```r
Wm1 <- cov(mclus_1)
nm1 <- nrow(mclus_1)
Hm1 <- (nm1-1)*Wm1
trmH1 <-sum(diag(Hm1))
trmH1
```

```
## [1] 3.817481
```

```r
Wm2 <- cov(mclus_2)
nm2 <- nrow(mclus_2)
Hm2 <- (nm2-1)*Wm2
trmH2 <-sum(diag(Hm2))
trmH2
```

```
## [1] 5.840899
```

```r
Wm3 <- cov(mclus_3)
nm3 <- nrow(mclus_3)
Hm3 <- (nm3-1)*Wm3
trmH3 <-sum(diag(Hm3))
trmH3
```

```
## [1] 0.3513119
```

```r
trWithin <- trmH1 +trmH2 + trmH3
trWithin
```

```
## [1] 10.00969
```

```r
trBetween <- trmH - trWithin
trBetween
```

```
## [1] 43.99031
```

La misura di non omogeneità statistica within è minore della misura di non omogeneità statistica between, quindi la divisione in cluster è soddisfacente.

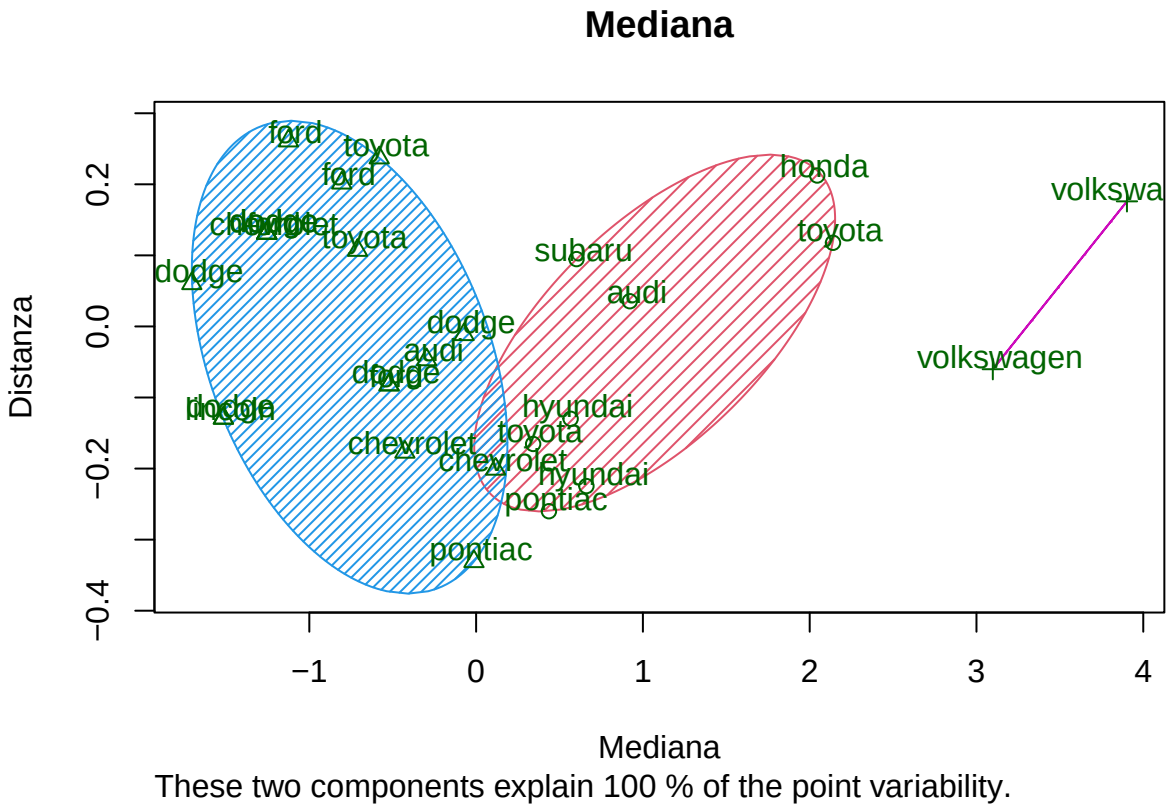
```r
m_perc <- as.character(round((trBetween/trmH),4)*100)
paste(m_perc,"%",sep="")
```

```
## [1] "81.46%"
```

Il rapporto tra la misura di non omogeneità statistica tra i cluster between e la misura di non omogeneità statistica totale è *81%*, quindi più elevata rispetto ai metodi del legame singolo, del legame medio e del centroide.

Vediamo come si presentano i cluster graficamente

```
clusplot(mpg_2, cut_mediana, color=TRUE,
         shade=TRUE, labels=3, lines=0,
         xlab="Mediana", ylab = "Distanza", main = "Mediana")
```



Rimane il gruppo che unisce le auto volkswagen,mentre riusciamo a vedere una distinzione tra il gruppo 1 e il gruppo 2 che prima non esisteva.

In conclusione possiamo dire che valutando tutti i metodi agglomerativi visti il metodo del legame completo è il migliore con un rapporto tra between e totale dell'*88%*, a seguire vi è il metodo del legame mediano con *81%* ed infine ex aequo troviamo il metodo del legame singolo, del legame medio e centroide.

Cluster non gerarchico

Negli ultimi decenni, i metodi di clustering non gerarchici sono diventati sempre più popolari nella comunità scientifica e in vari settori industriali. A differenza dei metodi di clustering gerarchici, che organizzano i dati in una struttura ad albero, nei metodi non gerarchici si assume che il numero di cluster in cui suddividere l'insieme sia fisso, e gli individui possono essere ricollocati in livelli precedenti. Questi metodi hanno dimostrato di essere estremamente utili in una vasta gamma di applicazioni, dall'analisi dei social network alla classificazione di immagini e al clustering di dati biologici. Esploreremo il principale metodo di clustering non gerarchico, ossia il *k-means* e analizzeremo le sue applicazioni.

L'algoritmo k-means è stato ideato nel 1975 da Hartigan e Wong due noti statistici

Gli algoritmi di clustering non gerarchici operano seguendo una procedura in cui, dopo aver creato una prima partizione, ogni punto viene riassegnato al cluster il cui centroide è più vicino, fino a quando non si verifica che nessun punto abbia una distanza inferiore dal centroide di un altro cluster rispetto a quello di appartenenza. In questo modo si giunge a una configurazione stabile in cui ciascun punto appartiene al cluster più vicino.

Il metodo del k-means richiede che il numero di cluster sia specificato a priori, e fornisce un'unica partizione. E funziona seguendo questo algoritmo:

- Step 1: Fissare il numero K di cluster, per una prima partizione provvisoria, specificando dei punti di riferimento.
- Step 2: Allocare gli individui nel cluster che ha i punti di riferimento più vicino
- Step 3: Calcolare il centroide per i k gruppi, i centroidi saranno i nuovi punti di riferimento.
- Step 4: Valutare la distanza minore tra gli elementi e i centroidi, cambiare gruppo all'individuo se non ha distanza minima con il centroide del gruppo in cui si trova.
- Step 5: Ricalcolare i centroidi dei k gruppi
- Step 6: Ripetere step 4 e 5 finché non ci sono più cambiamenti nei centroidi con l'iterazione precedente

Al fine di garantire la convergenza dell'algoritmo si utilizza la distanza euclidea e si utilizza la matrice contenente il quadrato delle distanze euclidee.

I benefici del metodo k-means includono la rapidità nell'esecuzione dei calcoli e l'ampia libertà lasciata ai punti per raggrupparsi e distanziarsi. Tuttavia, una limitazione dell'algoritmo è che la configurazione finale dei cluster può essere influenzata dalla scelta iniziale dei k centroidi, dall'ordine in cui vengono presi questi centroidi e dalle proprietà geometriche dei dati.

Visualizziamo le partizioni ottenuto utilizzando il k-means con la scelta casuale dei punti di riferimento e con 2 centroidi.

```
km_1 <- kmeans (mpg_2, center = 2, iter.max = 10, nstart = 1)
table(km_1$cluster)
```

```
##
##  1  2
## 24  4
```

```
km_1
```

```
## K-means clustering with 2 clusters of sizes 24, 4
##
## Cluster means:
##          cty        hwy
## 1 -0.3426781 -0.3163991
## 2  2.0560685  1.8983943
##
## Clustering vector:
##       audi       audi  chevrolet  chevrolet  chevrolet      dodge      dodge
##          1          1          1          1          1          1          1
##      dodge      dodge      dodge      dodge       ford       ford       ford
```

```
##          1          1          1          1          1          1          1
##       ford      honda    hyundai    hyundai    lincoln    pontiac    pontiac
##          1          2          1          1          1          1          1
##     subaru     toyota     toyota     toyota     toyota volkswagen volkswagen
##          1          1          1          2          1          2          2
##
## Within cluster sum of squares by cluster:
## [1] 15.100490  2.353355
##  (between_SS / total_SS =  67.7 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
km_1$centers
```

```
##          cty        hwy
## 1 -0.3426781 -0.3163991
## 2  2.0560685  1.8983943
```

```
km_1$withinss
```

```
## [1] 15.100490  2.353355
```

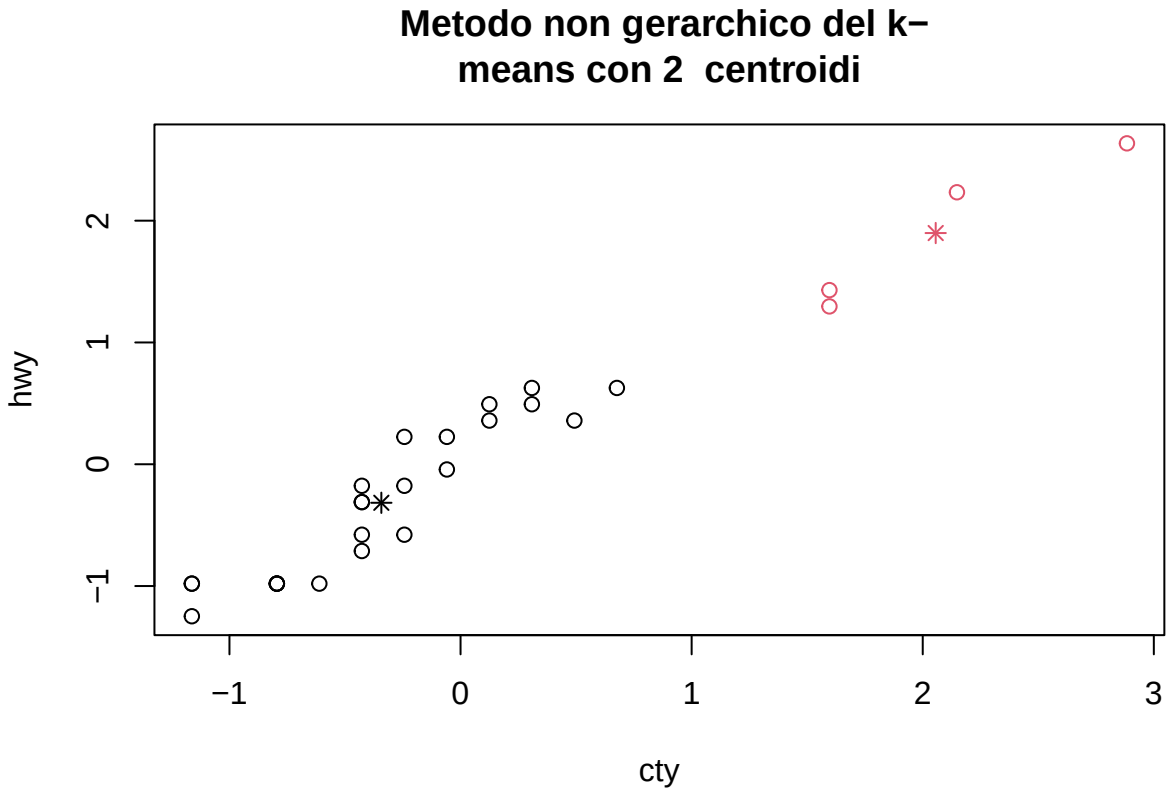
```
km_1$betweenss
```

```
## [1] 36.54615
```

Il metodo non gerarchico utilizzato divide la partizione in due, con 13 elementi nel cluster 1 e 15 elementi nel cluster 2. I centroidi hanno coordinata (0.86,0.93) e (-0.73,-0.80) circa, perché cambia ad ogni lancio, a causa della partizione iniziale che è sempre diversa.

La misura di non omogeneità statistica within è circa 15, mentre quella tra cluster, between è 39

```
plot(mpg_2, col = km_1$cluster , main = " Metodo non gerarchico del k-
means con 2  centroidi")
points (km_1$center , col = 1:2, pch = 8, cex =1)
```



Il grafico mostra dove sono posti i centroidi, e la distribuzione dei cluster dei due gruppi.

Vediamo adesso il k-means con 3 centroidi, come abbiamo fatto per i metodi gerarchici.

```
km_2 <- kmeans (mpg_2, center = 3, iter.max = 10, nstart = 10)
table(km_2$cluster)
```

```
##
##  1  2  3
## 10 14  4
```

```
km_2
```

```
## K-means clustering with 3 clusters of sizes 10, 14, 4
##
## Cluster means:
##          cty        hwy
## 1  0.1432022  0.3184712
## 2 -0.6897355 -0.7698778
## 3  2.0560685  1.8983943
##
## Clustering vector:
##       audi       audi  chevrolet  chevrolet  chevrolet      dodge      dodge
##          1          1          2          2          1          1          2
##      dodge      dodge      dodge      dodge       ford       ford       ford
```

```
##          2          2          2          2          2          2          2
##       ford      honda    hyundai    hyundai    lincoln    pontiac    pontiac
##          2          3          1          1          2          1          1
##     subaru     toyota     toyota     toyota     toyota volkswagen volkswagen
##          1          2          1          3          2          3          3
##
## Within cluster sum of squares by cluster:
## [1] 1.489533 2.654273 2.353355
##  (between_SS / total_SS =  88.0 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
km_2$centers
```

```
##          cty        hwy
## 1  0.1432022  0.3184712
## 2 -0.6897355 -0.7698778
## 3  2.0560685  1.8983943
```

```
km_2$withinss
```

```
## [1] 1.489533 2.654273 2.353355
```

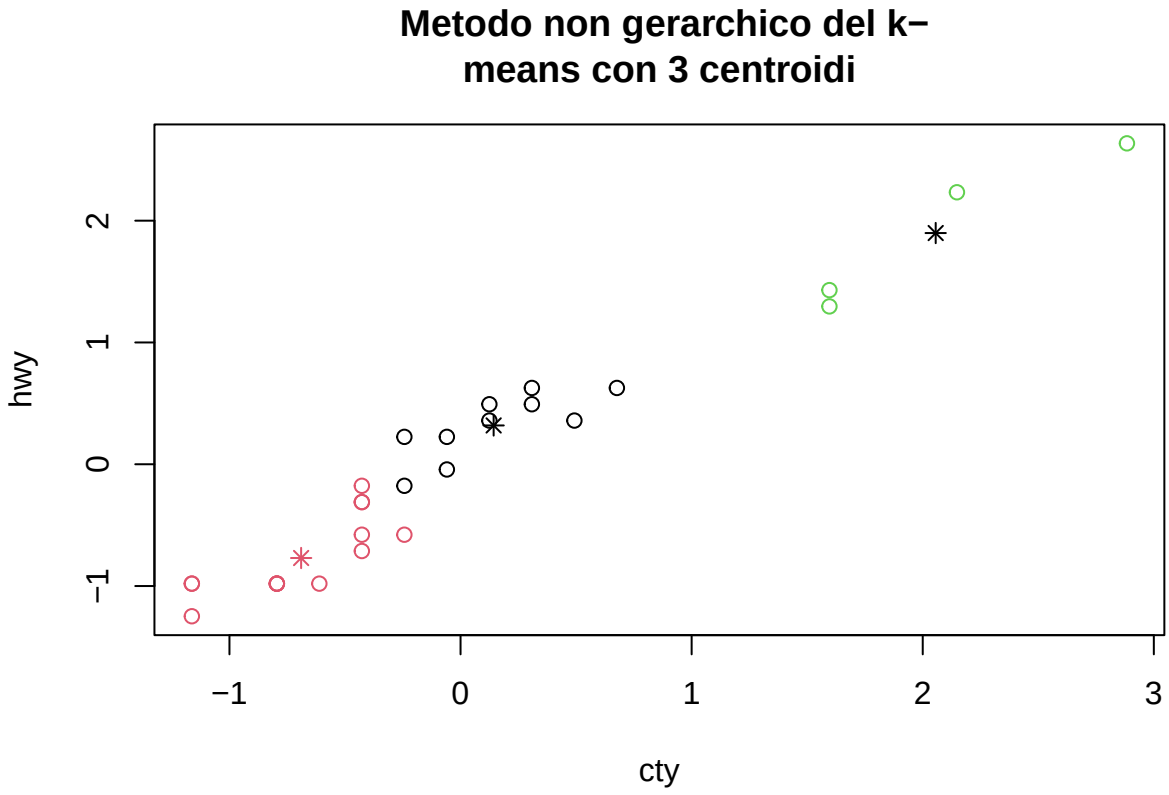
```
km_2$betweenss
```

```
## [1] 47.50284
```

Oltre a cambiare il numero dei centroidi abbiamo cambiato anche nstart, ossia richiediamo che l'algoritmo di agglomerazione venga ripetuto 10 volte, assicurandoci di minimizzare la misura within e massimizzare la between.

Anche in questo caso la misura within è circa 14, quindi diminuita e la between invece è cresciuta da 39 a 46. Dunque la partizione in 3 gruppi ci restituisce delle misure di non omogeneità statistica migliori

```
plot(mpg_2, col = km_2$cluster , main = " Metodo non gerarchico del k-
means con 3 centroidi")
points (km_2$center , col = 1:2, pch = 8, cex =1)
```



Vediamo utilizzando i centroidi calcolati con le misure gerarchiche:

```
d_c <-dist(mpg_2, method ="euclidean", diag=TRUE, upper = TRUE)
d_c <- d_c^2
tree <- hclust(d_c, method = "centroid")
taglio <- cutree(tree, k = 3, h =NULL )
tagliolist <-list(taglio)
centroidiIniziali <- aggregate(mpg_2, tagliolist, mean)[,-1]

km_c <- kmeans (mpg_2, centers = centroidiIniziali, iter.max = 10)

km_c
```

```
## K-means clustering with 3 clusters of sizes 14, 10, 4
##
## Cluster means:
##          cty        hwy
## 1 -0.6897355 -0.7698778
## 2  0.1432022  0.3184712
## 3  2.0560685  1.8983943
##
## Clustering vector:
##       audi       audi  chevrolet  chevrolet  chevrolet      dodge      dodge
##          2          2          1          1          2          2          1
##      dodge      dodge      dodge      dodge       ford       ford       ford
```

```
##          1          1          1          1          1          1          1
##       ford      honda    hyundai    hyundai    lincoln    pontiac    pontiac
##          1          3          2          2          1          2          2
##     subaru     toyota     toyota     toyota     toyota volkswagen volkswagen
##          2          1          2          3          1          3          3
##
## Within cluster sum of squares by cluster:
## [1] 2.654273 1.489533 2.353355
##  (between_SS / total_SS =  88.0 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
km_c$centers
```

```
##          cty        hwy
## 1 -0.6897355 -0.7698778
## 2  0.1432022  0.3184712
## 3  2.0560685  1.8983943
```

```
km_c$withinss
```

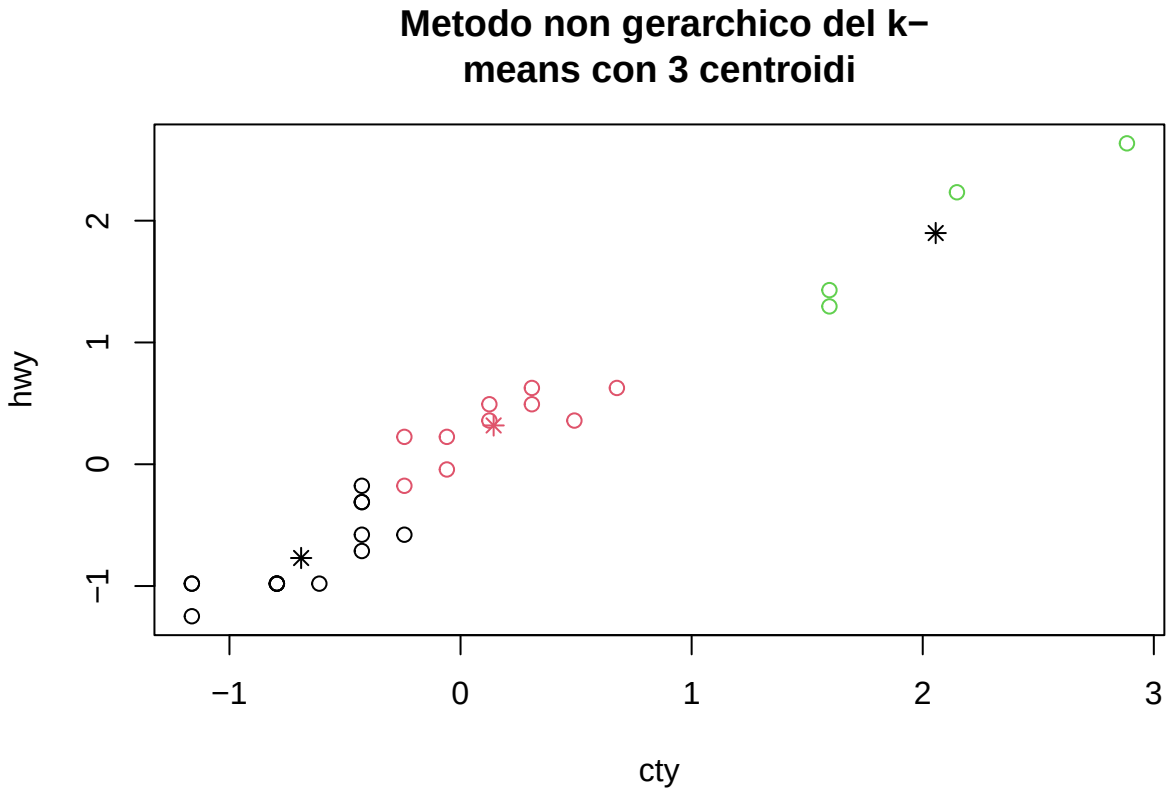
```
## [1] 2.654273 1.489533 2.353355
```

```
km_c$betweenss
```

```
## [1] 47.50284
```

Vediamo che la somma del withinss ci da 6, che è molto più piccola di quella ottenuta con i metodi precedenti, e la between è di 46, dunque abbiamo ottimizzato le misure di non omogeneità.

```
plot(mpg_2, col = km_c$cluster , main = " Metodo non gerarchico del k-
means con 3 centroidi")
points (km_c$center , col = 1:2, pch = 8, cex =1)
```



Fine prima parte