PIRANI MAX ENRICO

# STATISTICA SENZA MATEMATICA

# *Introduzione*

Ho scritto questo libro principalmente per i nostri studenti, la maggior parte dei quali non amava la matematica e non capiva perché dovessero imparare le formule matematiche quando il software del computer eseguiva i calcoli per loro. Non li convinceva l'argomentazione secondo cui l'esecuzione dei calcoli avrebbe permesso loro di comprendere il test, e non convinceva nemmeno noi. Volevamo che avessero una comprensione concettuale della statistica e che si divertissero ad analizzare i dati. Negli ultimi 20 anni abbiamo dovuto adattare il nostro insegnamento a gruppi numerosi di studenti, molti dei quali non avevano una formazione formale in matematica. Ci siamo resi conto che era difficile consigliare alcuni dei tradizionali libri di testo di statistica: o erano pieni di formule matematiche, e venivano percepiti dagli studenti come noiosi, o erano semplici ricette di cucina statistica, che mostravano loro come eseguire i calcoli, ma non davano loro una reale comprensione del significato delle statistiche. Abbiamo quindi deciso di scrivere questo libro, che cerca di dare agli studenti una comprensione concettuale della statistica evitando la distrazione delle formule e dei calcoli.

Abbiamo cercato di semplificare concetti complessi, a volte molto complessi. La semplificazione comporta un compromesso in termini di accuratezza. Ne eravamo consapevoli durante la stesura del libro e abbiamo cercato di essere il più accurati possibile, pur fornendo la spiegazione più semplice. Siamo anche consapevoli che alcuni studenti non utilizzano `R` per l'analisi dei dati. SPSS®, tuttavia, è ancora uno dei pacchetti statistici più utilizzati nelle scienze sociali ed è per questo che il testo è strettamente legato a `R`. Gli studenti che non utilizzano questo pacchetto dovrebbero comunque trovare il libro utile.

# *Statistica descrittiva*

## *Media*

La media è la somma di tutti i risultati in un campione diviso per il numerodei risultati nel campione. In altre parole, si calcola prendendo la somma ditutti i valori nel campione e dividendo per il numero totale di valori nel campione. Un esempio di calcolo di media in `R` :

```
# Inizializzazione dei valori
voti <- c(80, 85, 90, 95)

# Calcolo della media
media <- mean(voti)

# Visualizzazione della media
media
```

```
## [1] 87.5
```

Questo codice calcolerà la media dei voti presenti nel vettore `voti` e restituirà il risultato.

## *Mediana*

La mediana è il valore che si trova esattamente al centro di un insiemeordinato di numeri. Per calcolare la mediana, è necessario ordinare i numeriin ordine crescente o decrescente e trovare il valore centrale o la media deidue valori centrali se il numero di elementi è pari.

Ecco un esempio di calcolo della mediana in R utilizzando Rmarkdown:

```
# Dataset di esempio
dati <- c(2, 3, 5, 6, 8, 10, 12, 15)

# Calcolo della mediana
```

```r
mediana <- median(dati)

# Stampa il risultato
mediana
```

```
## [1] 7
```

Nell'esempio sopra, abbiamo un vettore di dati `dati` e utilizziamo la funzione `median()` di R per calcolare la mediana dei valori. Il risultato viene quindi visualizzato tramite la funzione `print()`.

### *Definizione di Ranking*

Il ranking è un metodo per ordinare una serie di oggetti in base a una determinata caratteristica o criterio. In sostanza, il ranking assegna un posto o una posizione unica a ciascun oggetto, in base a come si posiziona rispetto agli altri oggetti.Ecco un esempio di come calcolare il ranking di una lista di valori in `R`

Iniziamo creando un nuovo documento R Markdown (.Rmd) nel tuo ambiente R.

Prima di tutto, assicurati di avere installati i pacchetti necessari. In questo esempio utilizzeremo il pacchetto "dplyr" per la manipolazione dei dati e il pacchetto "tidyr" per la trasformazione dei dati in formato lungo.

Ecco un esempio di codice R Markdown:

```r
# Creazione dei dati di esempio
valori <- c(10, 5, 8, 2)

# Calcolo del ranking dei valori
ranking <- rank(-valori, ties.method = "min")

# Visualizzazione del ranking
ranking
```

```
## [1] 1 3 2 4
```

La funzione `rank()` viene utilizzata per calcolare il ranking dei valori nel vettore `valori`. L'argomento `ties.method = "min"` viene utilizzato per gestire il caso in cui ci siano valori duplicati nel vettore. In questo caso, il metodo "min" viene utilizzato per assegnare il ranking più basso ai valori duplicati (ad esempio, se ci sono due valori uguali al secondo posto, entrambi riceveranno il ranking 2).

Nell'esempio, il vettore `valori` contiene i valori [10, 5, 8, 2]. Il codice calcola il ranking di questi valori, ottenendo [1, 3, 2, 4]. Significa che il numero più alto (10) ha il ranking 1, il numero successivo più alto (8) ha il ranking 2, il terzo più alto (5) ha il ranking 3

e l'ultimo (2) ha il ranking 4. Questo viene visualizzato usando la funzione `print()`.

### *Definizione di Moda in statistica descrittiva*

La moda in statistica descrittiva è il valore che compare più frequentemente inun insieme di dati. In altre parole, è il valore che si presenta con la maggiore frequenza. Ecco un esempio in `R`:

```
# Esempio di calcolo della moda in R
x <- c(1, 2, 3, 3, 4, 4, 5, 5, 5)
moda <- unique(x)[which.max(tabulate(match(x, unique(x))))]
print(moda)
```

Nell'esempio fornito, l'insieme di dati `x` contiene i numeri da 1 a 5, ma con frequenze diverse. La moda sarà quindi il valore 5, poiché compare più frequentemente degli altri valori.

### *Problema del calcolo delle medie quando gli estremi sono troppo distanti tra loro*

Quando gli estremi in un insieme di dati sono troppo distanti tra loro, potrebbeesserci un problema nel calcolo della media. La media è influenzata dai valoriestremi perché tiene conto di tutti i valori nel calcolo. Se ci sono valoriestremamente alti o bassi, possono distorcere la media.

Hai ragione, la presenza di valori estremi in un insieme di dati può influenzare significativamente la media calcolata. In questi casi, può essere più appropriato considerare l'uso di altre misure statistiche, come la mediana o la moda, che possono essere meno influenzate da valori estremi. Un esempio pratico di questo potrebbe essere :

Ad esempio, se stiamo cercando di calcolare la media dei redditi delle persone in una determinata area, ma ci sono poche persone con redditi molto elevati o molto bassi che distorcono significativamente la media, allora potrebbe essere più utile considerare la mediana dei redditi, che rappresenta il valore centrale dell'insieme di dati. In questo modo, si riduce l'influenza dei valori estremi e si ottiene una misura più rappresentativa della distribuzione dei redditi. Oppure un altro esempio:

Se calcoliamo la media di questi voti, otterremo:

$(50 + 60 + 70 + 80 + 90 + 100 + 200) / 7 = 750 / 7 \quad 107.14$

Qui possiamo notare che il valore estremo di 200 ha fortemente influenzato la media, portandola ad essere piuttosto alta. Questo può avere un effetto distorto sulla rappresentazione complessiva dei dati.

In situazioni come questa, potrebbe essere più appropriato utilizzare altre misure di tendenza centrale come la mediana o la moda, che non sono influenzate in modo significativo dai valori estremi.

In conclusione, quando gli estremi sono troppo distanti tra loro, è importante considerare l'utilizzo di diverse misure di tendenza centrale e valutare le caratteristiche dei dati per ottenere una descrizione accurata e completa.

In caso di valori estremi che potrebbero influenzare significativamente la media, potrebbe essere più appropriato utilizzare la mediana o la moda come misure di tendenza centrale.

La mediana è il valore centrale di un insieme di dati ordinato, che non viene influenzato dai valori estremi. Ordinando i dati in ordine crescente o decrescente, la mediana sarà il valore esattamente al centro della lista.

La moda, d'altra parte, è il valore che compare più frequentemente nella lista di dati. La moda è anche meno influenzata dai valori estremi rispetto alla media, poiché si basa sulla frequenza di comparsa dei valori piuttosto che sulla loro somma.

Quindi, in caso di valori estremi, potrebbe essere utile considerare sia la mediana che la moda come alternative alla media per ottenere una visione più accurata del centro dei dati.

## *Definizione di errore di campionatura*

L'errore di campionatura è la discrepanza tra il valore stimato di una variabiledi interesse e il suo vero valore nella popolazione, causata dall'estrazione diun campione invece di raccogliere dati da tutta la popolazione. Questo errorepuò essere ridotto aumentando la dimensione del campione o utilizzando tecnichedi campionamento appropriate.

Un esempio pratico di errore di campionatura potrebbe essere l'indagine su un'elezione politica. Supponiamo di voler stimare la percentuale di voti che un candidato ha ottenuto nella popolazione generale. Selezioniamo un campione rappresentativo di elettori e chiediamo loro per chi hanno votato, potremmo ottenere una stima del risultato elettorale. Tuttavia, questa stima potrebbe deviare dal risultato reale a causa di variazioni casuali nel campione selezionato. Quanto più grande è il campione e quanto più accuratamente è selezionato, tanto più piccolo sarà l'errore di campionatura e tanto più accurata sarà la nostra stima.

### *Cosa significa misura di tendenza centrale*

La misura di tendenza centrale è un valore statistico che rappresenta un punto centrale o tipico di un insieme di dati. È utilizzato per indicare il valore intorno al quale si raggruppano i dati.

Ecco una definizione e un esempio pratico in R:

> Definizione: La misura di tendenza centrale è un valore che rappresenta la posizione centrale di un insieme di dati. Le tre misure di tendenza centrale comuni sono la media, la mediana e la moda.

Esempio pratico in R: Supponiamo di avere un vettore di dati numeric, ad esempio: 3, 5, 7, 9, 11.

Per calcolare la media:

```
dati <- c(3, 5, 7, 9, 11)
media <- mean(dati)
```

Per calcolare la mediana:

```
dati <- c(3, 5, 7, 9, 11)
mediana <- median(dati)
```

Per calcolare la moda:

```
dati <- c(3, 5, 5, 7, 9, 11)
moda <- table(dati)
moda <- as.numeric(names(moda)[moda == max(moda)])
```

Nell'esempio sopra, la media sarebbe 7, la mediana sarebbe 7 e la moda sarebbe 5. Queste tre misure di tendenza centrale ci danno informazioni diverse sulla distribuzione dei dati.

### *Come ottenere una misura di tendenza centrale utilizzando R*

I questo capitolo si descrivono i comandi per ottenere una misura di analisi centrale utilizzando R.

Ora iniziamo con l'imparare a creare un dataset utilizzando SQLite e creando il dataset partendo da una tabella già presente. Il database si chiama `dog_walking.db` mentre la tabella si chiama `walking_data`. Il tutto creando un codice in `R`.

Ecco un'esempio di codice in R che illustra come creare un dataset utilizzando SQLite e partendo da una tabella esistente:

```
# Installa il pacchetto RSQLite se non hai già fatto
# install.packages("RSQLite")

# Carica il pacchetto RSQLite
```

```
library(RSQLite)

# Imposta la connessione al database
con <- dbConnect(RSQLite::SQLite(), "dog_walking.db")

# Esegue la query per selezionare tutti i dati dalla tabella
query <- "SELECT * FROM walking_data"
walkin_dog <- dbGetQuery(con, query)

# Chiudi la connessione al database
dbDisconnect(con)

# Stampa il dataset appena creato
print(walkin_dog)
```

Assicurati di avere il pacchetto `RSQLite` installato sul tuo sistema prima di eseguire questo codice. Puoi installarlo utilizzando la funzione `install.packages("RSQLite")` se necessario.

Ora vediamo di utilizzare un funzione che sia simile al commando `explore` nel pacchetto SPSS analizzando creando delle misure di tendenza centrale sempre usando R.

```
library(summarytools)
descr(walkin_dog)
confidence_interval <- t.test(walkin_dog)$conf.init
confidence_interval
```

### *La rappresentazione grafica dei dati*

L'istogramma di frequenza è un utile strumento per illustrare graficamente i dati. I ricercatori sono spesso interessati alla frequenza di occorrenza dei valori nei loro dati di campionamento. Ad esempio, se si raccoglievano informazioni sulle occupazioni degli individui, si potrebbe essere interessati a scoprire quanti individui rientrano in ciascuna categoria di impiego. Per illustrare l'istogramma, consideriamo un istogramma di frequenza basato sui dati raccolti in uno studio del 2011 condotto da Armitage e Reidy (non pubblicato). In questo studio sulla paura del sangue, i ricercatori chiesero ai partecipanti di indicare da una lista di sette colori quale fosse il loro preferito. L'istogramma che rappresenta i dati è mostrato nella Figura . Dalla Figura si può notare che le persone in questo campione hanno valutato il blu come il colore preferito più spesso e il bianco come il meno preferito. L'istogramma di frequenza è un ottimo modo per ispezionare visivamente i dati. Spesso desideriamo sapere se ci sono

punteggi che potrebbero sembrare un po' fuori posto. Ora per importare il dataset nel mio database sqlite uso questo codice

```
library(haven)
library(DBI)
data_colors <- read_sav("/Users/enricopirani/Downloads/ch03_favourite_colours.sav")
con <- dbConnect(RSQLite::SQLite(), dbname= "~/Statistica/dog_walking.db")
dbWriteTable(con, "colors", data_colors, overwrite = TRUE)
query <- "SELECT * FROM colors"
colors_data <- dbGetQuery(con, query)
```

Ora ottengo l'istogramma dal dataset

```
library(ggplot2)
# Crea l'istogramma
grafico_istogramma <- ggplot(data_colors, aes(x = colours)) +
  geom_bar(fill = "green", color = "black") +
  theme(panel.background = element_rect(fill = "black"))
print(grafico_istogramma)
```

Nel dataset colors_data ho due colonne nella prima c'è l' ID del soggetto nella seconda un numero che varia da 1 a 7. Ogni numero corrisponde a 1=Rosso , 2=Verde, 3=Marrone, 4=Nero, 5=Bianco, 6=Blue, 7=Giallo. Ora devo creare un histogramma in cui nelle X c'è il colore preferito mentre nelle Y il numero di persone. Come posso farlo in R

```
conteggio_colore <- table(colors_data$COLOUR)
etichette_colori <- c("Rosso", "Verde", "Marrone", "Nero", "Bianco", "Blue", "Giallo")

barplot(conteggio_colore, main = "Preferenza di colore",
        xlab = "Colore", ylab = "Numero di persone",
        names.arg = etichette_colori)
```

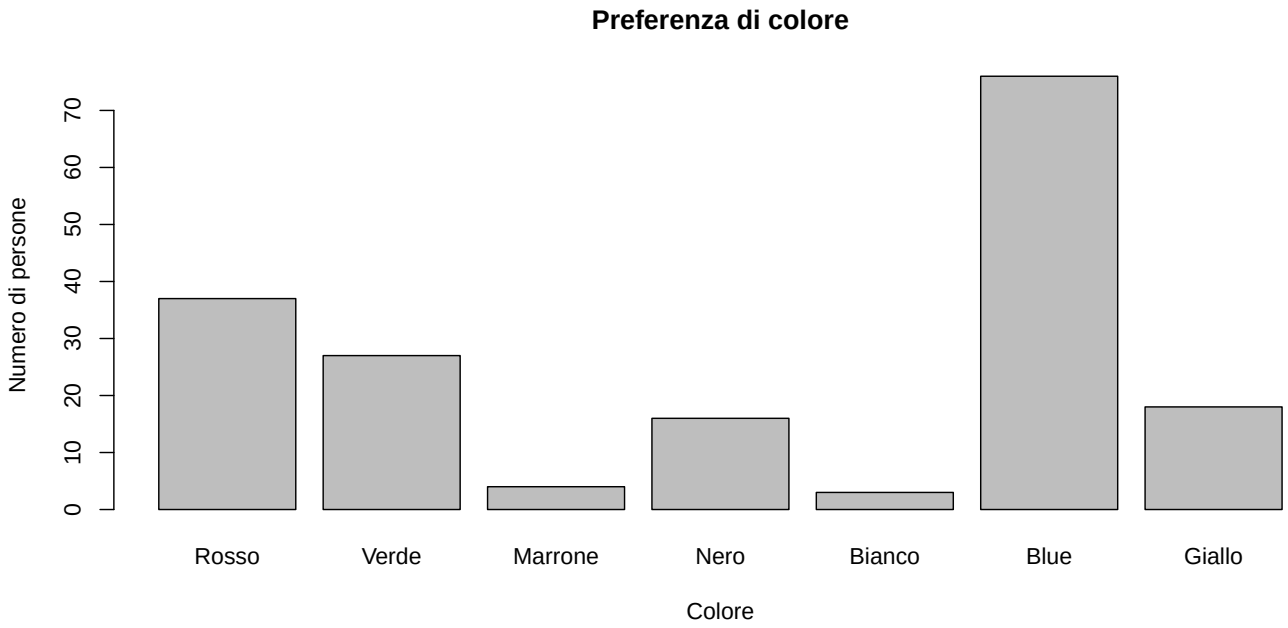



Figure 1: A full width figure.

# *Figures*

## *Margin Figures*

Images and graphics play an integral role in Tufte's work. To place figures in the margin you can use the **knitr** chunk option `fig.margin = TRUE`. For example:

```
library(ggplot2)
mtcars2 <- mtcars
mtcars2$am <- factor(
  mtcars$am, labels = c('automatic', 'manual')
)
ggplot(mtcars2, aes(hp, mpg, color = am)) +
  geom_point() + geom_smooth() +
  theme(legend.position = 'bottom')
```

Note the use of the `fig.cap` chunk option to provide a figure caption. You can adjust the proportions of figures using the `fig.width` and `fig.height` chunk options. These are specified in inches, and will be automatically scaled down to fit within the handout margin.

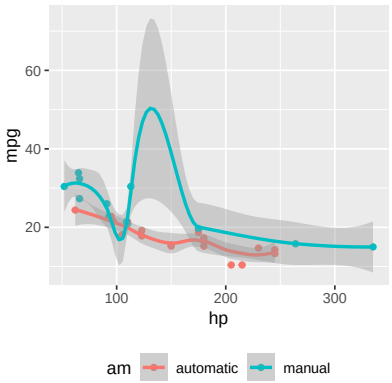



Figure 2: MPG vs horsepower, colored by transmission.

## *Arbitrary Margin Content*

In fact, you can include anything in the margin using the **knitr** engine named `marginfigure`. Unlike R code chunks ```` ```{r} ````, you write a chunk starting with ```` ```{marginfigure} ```` instead, then put the content in the chunk. See an example on the right about the first fundamental theorem of calculus.

For the sake of portability between LaTeX and HTML, you should keep the margin content as simple as possible (syntax-wise) in the `marginefigure` blocks. You may use simple Markdown syntax like `**bold**` and `_italic_` text, but please refrain from using footnotes, citations, or block-level elements (e.g. blockquotes and lists) there.

Note: if you set `echo = FALSE` in your global chunk options, you will have to add `echo = TRUE` to the chunk to display a margin figure, for example ```` ```{marginfigure, echo = TRUE} ````.

We know from *the first fundamental theorem of calculus* that for $x$ in $[a, b]$:

$$\frac{d}{dx}\left(\int_a^x f(u)\,du\right) = f(x).$$

### *Full Width Figures*

You can arrange for figures to span across the entire page by using the chunk option `fig.fullwidth = TRUE`.

```
ggplot(diamonds, aes(carat, price)) + geom_smooth() +
  facet_grid(~ cut)
```

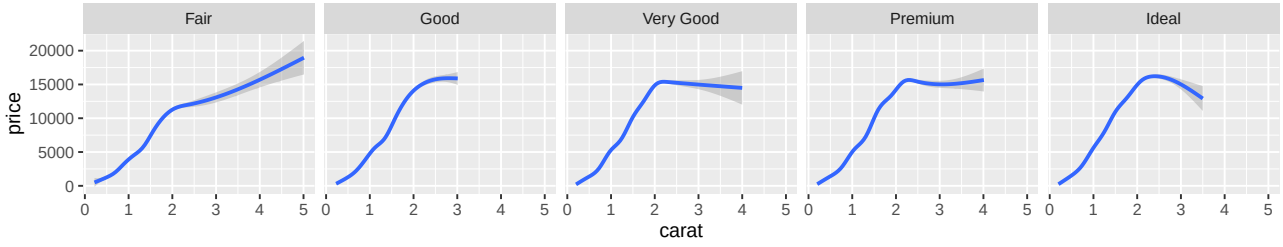



Figure 3: A full width figure.

Other chunk options related to figures can still be used, such as `fig.width`, `fig.cap`, `out.width`, and so on. For full width figures, usually `fig.width` is large and `fig.height` is small. In the above example, the plot size is $10 \times 2$.

### *Arbitrary Full Width Content*

Any content can span to the full width of the page. This feature requires Pandoc 2.0 or above. All you need is to put your content in a fenced `Div` with the class `fullwidth`, e.g.,

```
::: {.fullwidth}
Any _full width_ content here.
:::
```

Below is an example:

*R is free software and comes with ABSOLUTELY NO WARRANTY.* You are welcome to redistribute it under the terms of the GNU General Public License versions 2 or 3. For more information about these matters see `https://www.gnu.org/licenses/`.

### *Main Column Figures*

Besides margin and full width figures, you can of course also include figures constrained to the main column. This is the default type of figures in the LaTeX/HTML output.

```
ggplot(diamonds, aes(cut, price)) + geom_boxplot()
```

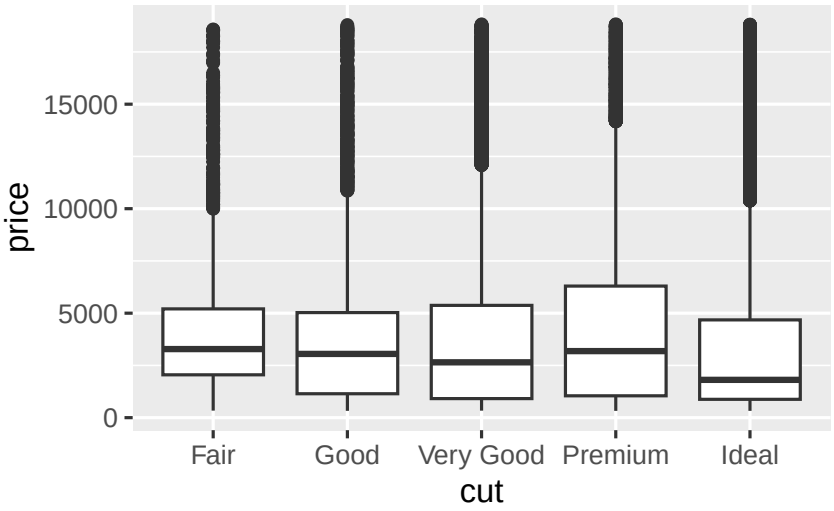



Figure 4: A figure in the main column.

# *Sidenotes*

One of the most prominent and distinctive features of this style is the extensive use of sidenotes. There is a wide margin to provide ample room for sidenotes and small figures. Any use of a footnote will automatically be converted to a sidenote. [1]

If you'd like to place ancillary information in the margin without the sidenote mark (the superscript number), you can use the `margin_note()` function from **tufte** in an inline R expression. This function does not process the text with Pandoc, so Markdown syntax will not work here. If you need to write anything in Markdown syntax, please use the `marginfigure` block described previously.

[1] This is a sidenote that was entered using a footnote.

This is a margin note. Notice that there is no number preceding the note.

# *References*

References can be displayed as margin notes for HTML output. For example, we can cite R here [R Core Team, 2023]. To enable this feature, you must set `link-citations: yes` in the YAML metadata, and the version of Pandoc should at least 2.11 or `pandoc-citeproc` should be available and at least 0.7.2. You can always install your own version of Pandoc from `https://pandoc.org/installing.html` if the version is not sufficient. To check the version of `pandoc-citeproc` in your system, you may run this in R:

```
system2('pandoc-citeproc', '--version')
```

If your version of `pandoc-citeproc` is too low, or you did not set `link-citations: yes` in YAML, references in the HTML output will be placed at the end of the output document.

# *Tables*

You can use the `kable()` function from the **knitr** package to format tables that integrate well with the rest of the Tufte handout style. The table captions are placed in the margin like figures in the HTML output.

```
knitr::kable(
  mtcars[1:6, 1:6], caption = 'A subset of mtcars.'
)
```

Table 1: A subset of mtcars.

| | mpg | cyl | disp | hp | drat | wt |
|---|---|---|---|---|---|---|
| Mazda RX4 | 21.0 | 6 | 160 | 110 | 3.90 | 2.620 |
| Mazda RX4 Wag | 21.0 | 6 | 160 | 110 | 3.90 | 2.875 |
| Datsun 710 | 22.8 | 4 | 108 | 93 | 3.85 | 2.320 |
| Hornet 4 Drive | 21.4 | 6 | 258 | 110 | 3.08 | 3.215 |
| Hornet Sportabout | 18.7 | 8 | 360 | 175 | 3.15 | 3.440 |
| Valiant | 18.1 | 6 | 225 | 105 | 2.76 | 3.460 |

# *Block Quotes*

We know from the Markdown syntax that paragraphs that start with `>` are converted to block quotes. If you want to add a right-aligned footer for the quote, you may use the function `quote_footer()` from **tufte** in an inline R expression. Here is an example:

> "If it weren't for my lawyer, I'd still be in prison. It went a lot faster with two people digging."
>
> — Joe Martin

Without using `quote_footer()`, it looks like this (the second line is just a normal paragraph):

> "Great people talk about ideas, average people talk about things, and small people talk about wine."
>
> — Fran Lebowitz

# *Responsiveness*

The HTML page is responsive in the sense that when the page width is smaller than 760px, sidenotes and margin notes will be hidden by default. For sidenotes, you can click their numbers (the superscripts) to toggle their visibility. For margin notes, you may click the circled plus signs to toggle visibility.

# *More Examples*

The rest of this document consists of a few test cases to make sure everything still works well in slightly more complicated scenarios. First we generate two plots in one figure environment with the chunk option `fig.show = 'hold'`:

```
p <- ggplot(mtcars2, aes(hp, mpg, color = am)) +
  geom_point()
p
p + geom_smooth()
```

Then two plots in separate figure environments (the code is identical to the previous code chunk, but the chunk option is the default `fig.show = 'asis'` now):

```
p <- ggplot(mtcars2, aes(hp, mpg, color = am)) +
  geom_point()
p
```

```
p + geom_smooth()
```

You may have noticed that the two figures have different captions, and that is because we used a character vector of length 2 for the chunk option `fig.cap` (something like `fig.cap = c('first plot', 'second plot')`).

Next we show multiple plots in margin figures. Similarly, two plots in the same figure environment in the margin:

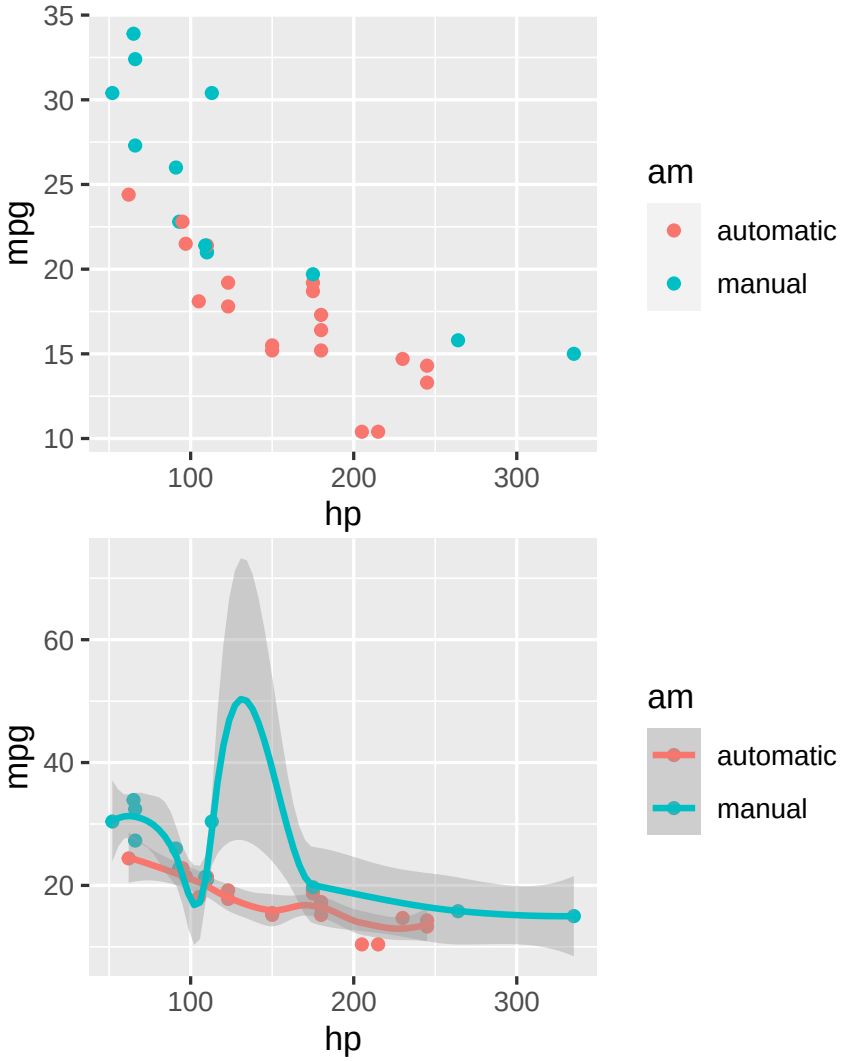



Figure 5: Two plots in one figure environment.

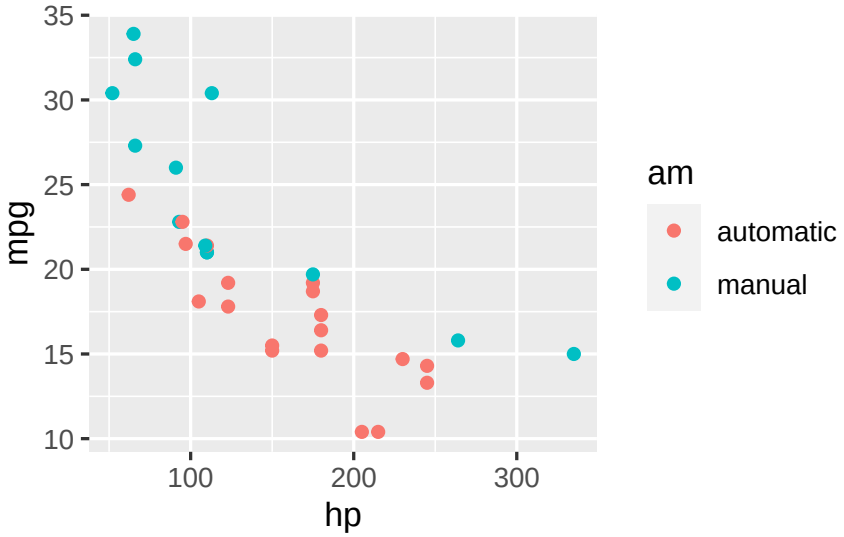



Figure 6: Two plots in separate figure environments (the first plot).

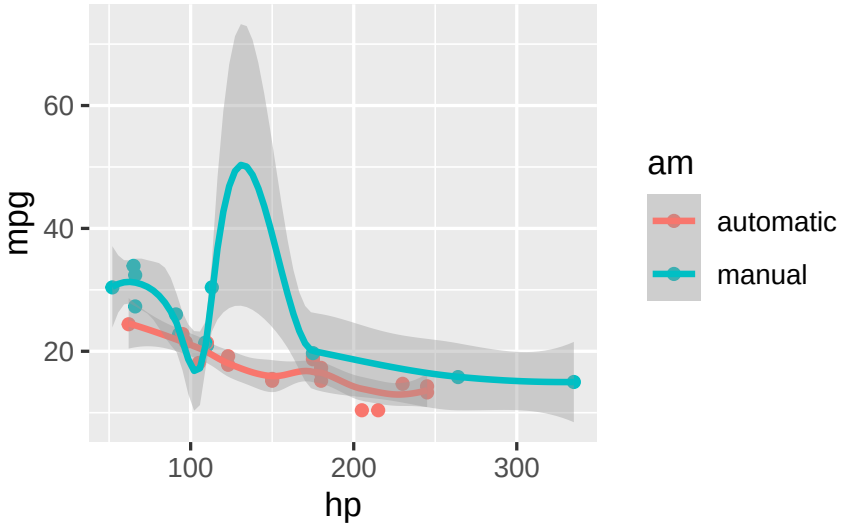



Figure 7: Two plots in separate figure environments (the second plot).

```
p
p + geom_smooth(method = 'lm')
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

Then two plots from the same code chunk placed in different figure environments:

```
knitr::kable(head(iris, 15))
```

| Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species |
|---|---|---|---|---|
| 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| 4.6 | 3.1 | 1.5 | 0.2 | setosa |
| 5.0 | 3.6 | 1.4 | 0.2 | setosa |
| 5.4 | 3.9 | 1.7 | 0.4 | setosa |
| 4.6 | 3.4 | 1.4 | 0.3 | setosa |
| 5.0 | 3.4 | 1.5 | 0.2 | setosa |
| 4.4 | 2.9 | 1.4 | 0.2 | setosa |
| 4.9 | 3.1 | 1.5 | 0.1 | setosa |
| 5.4 | 3.7 | 1.5 | 0.2 | setosa |
| 4.8 | 3.4 | 1.6 | 0.2 | setosa |
| 4.8 | 3.0 | 1.4 | 0.1 | setosa |
| 4.3 | 3.0 | 1.1 | 0.1 | setosa |
| 5.8 | 4.0 | 1.2 | 0.2 | setosa |

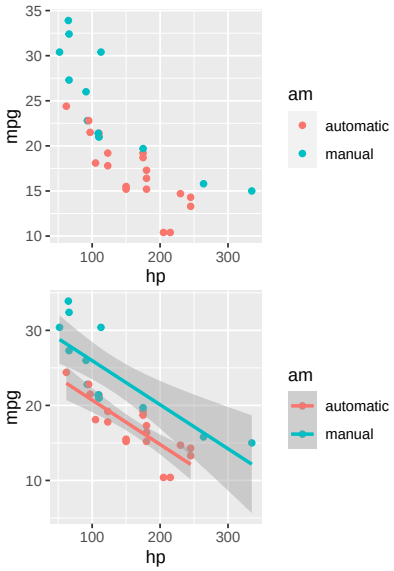



Figure 8: Two plots in one figure environment in the margin.

```
p
```

```
knitr::kable(head(iris, 12))
```

| Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species |
|---|---|---|---|---|
| 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| 4.6 | 3.1 | 1.5 | 0.2 | setosa |
| 5.0 | 3.6 | 1.4 | 0.2 | setosa |
| 5.4 | 3.9 | 1.7 | 0.4 | setosa |
| 4.6 | 3.4 | 1.4 | 0.3 | setosa |
| 5.0 | 3.4 | 1.5 | 0.2 | setosa |

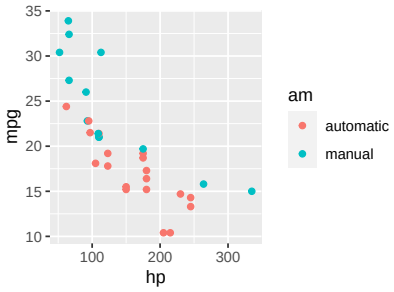



Figure 9: Two plots in separate figure environments in the margin (the first plot).

| Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species |
|---|---|---|---|---|
| 4.4 | 2.9 | 1.4 | 0.2 | setosa |
| 4.9 | 3.1 | 1.5 | 0.1 | setosa |
| 5.4 | 3.7 | 1.5 | 0.2 | setosa |
| 4.8 | 3.4 | 1.6 | 0.2 | setosa |

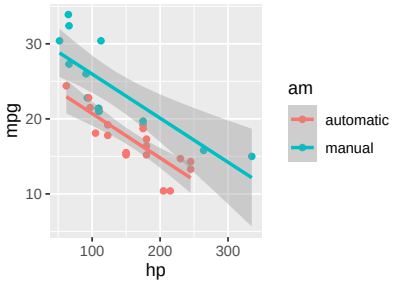



Figure 10: Two plots in separate figure environments in the margin (the second plot).

```
p + geom_smooth(method = 'lm')
```

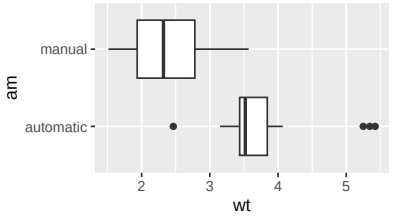
```
## `geom_smooth()` using formula = 'y ~ x'
```

```
knitr::kable(head(iris, 5))
```

| Sepal.Length | Sepal.Width | Petal.Length | Petal.Width | Species |
|---|---|---|---|---|
| 5.1 | 3.5 | 1.4 | 0.2 | setosa |
| 4.9 | 3.0 | 1.4 | 0.2 | setosa |
| 4.7 | 3.2 | 1.3 | 0.2 | setosa |
| 4.6 | 3.1 | 1.5 | 0.2 | setosa |
| 5.0 | 3.6 | 1.4 | 0.2 | setosa |

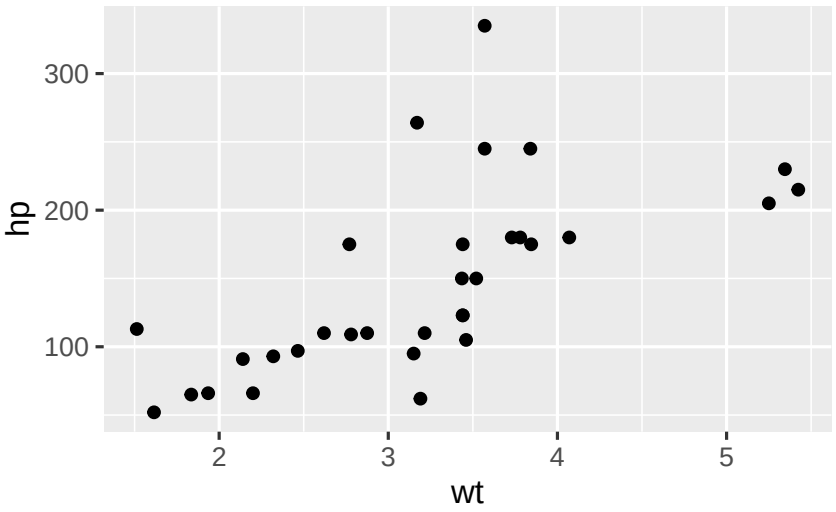
We blended some tables in the above code chunk only as *placeholders* to make sure there is enough vertical space among the margin figures, otherwise they will be stacked tightly together. For a practical document, you should not insert too many margin figures consecutively and make the margin crowded.

You do not have to assign captions to figures. We show three figures with no captions below in the margin, in the main column, and in full width, respectively.



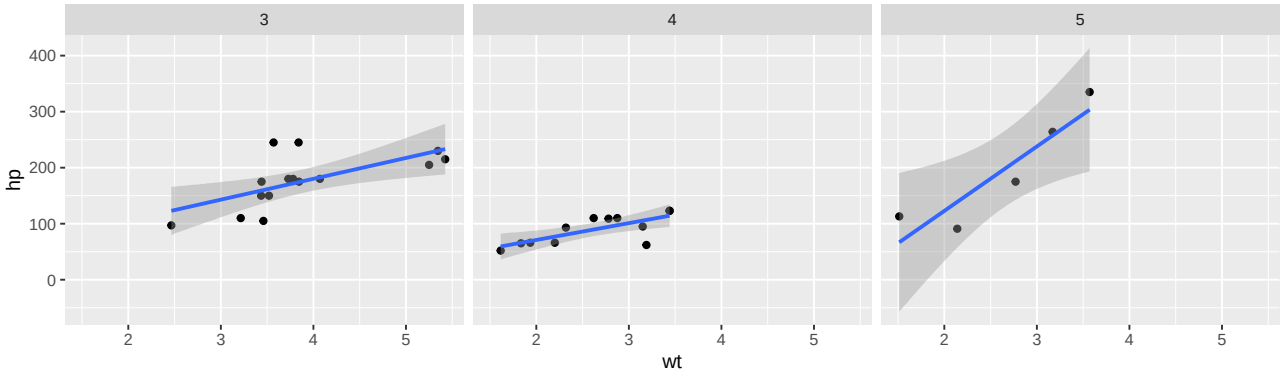

```
# a boxplot of weight vs transmission; this figure
# will be placed in the margin
ggplot(mtcars2, aes(am, wt)) + geom_boxplot() +
  coord_flip()
```

```
# a figure in the main column
p <- ggplot(mtcars, aes(wt, hp)) + geom_point()
p
```




```
# a fullwidth figure
p + geom_smooth(method = 'lm') + facet_grid(~ gear)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



# *Some Notes on Tufte CSS*

There are a few other things in Tufte CSS that we have not mentioned so far. If you prefer sans-serif fonts, use the function `sans_serif()` in **tufte**. For epigraphs, you may use a pair of underscores to make the paragraph italic in a block quote, e.g.

> *I can win an argument on any topic, against any opponent. People know this, and steer clear of me at parties. Often, as a sign of their great respect, they don't even invite me.*
>
> — Dave Barry

We hope you will enjoy the simplicity of R Markdown and this R package, and we sincerely thank the authors of the Tufte-CSS and Tufte-LaTeX projects for developing the beautiful CSS and LaTeX classes. Our **tufte** package would not have been possible without their heavy lifting.

You can turn on/off some features of the Tufte style in HTML output. The default features enabled are:

```
output:
  tufte::tufte_html:
    tufte_features: ["fonts", "background", "italics"]
```

If you do not want the page background to be lightyellow, you can remove `background` from `tufte_features`. You can also customize the style of the HTML page via a CSS file. For example, if you do not want the subtitle to be italic, you can define

```
h3.subtitle em {
  font-style: normal;
}
```

in, say, a CSS file `my_style.css` (under the same directory of your Rmd document), and apply it to your HTML output via the `css` option, e.g.,

```
output:
  tufte::tufte_html:
```

```
    tufte_features: ["fonts", "background"]
    css: "my_style.css"
```

There is also a variant of the Tufte style in HTML/CSS named "Envisoned CSS". This style can be used by specifying the argument `tufte_variant = 'envisioned'` in `tufte_html()`[2], e.g.

[2] The actual Envisioned CSS was not used in the **tufte** package. We only changed the fonts, background color, and text color based on the default Tufte style.

```
output:
  tufte::tufte_html:
    tufte_variant: "envisioned"
```

To see the R Markdown source of this example document, you may follow this link to Github, use the wizard in RStudio IDE (`File -> New File -> R Markdown -> From Template`), or open the Rmd file in the package:

```
file.edit(
  tufte:::template_resources(
    'tufte_html', '..', 'skeleton', 'skeleton.Rmd'
  )
)
```

This document is also available in Chinese, and its `envisioned` style can be found here.

# *Bibliography*

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2023. URL `https://www.R-project.org/`.