Estrazione casuale del campione

```
# Va sempre rieseguito il set seed sennò cambia l'n
set.seed(matricola, sample.kind = "Rejection")
n <- sample(110:120, 1)
campione <- popolazione[sample(nrow(popolazione), n), ]
```

Test d'indipendenza tra due variabili categorizzate

```
# Dopo aver tagliato in classi v1_class e v2_class
tabella <- table(v1_class, v2_class)
chisq.test(tabella)
```

Cambio di tipologia

Mettiamo caso io abbia delle classi al posto di una variabile numerica:

- uno, due, tre, quattro al posto dei numeri corrispettivi
- basso, medio, alto, altissimo

Per i numeri andrò a convertirli

```
classi_num <- as.numeric(factor(classi_nomi, levels = c("uno", "due", "tre", "quattro"), ordered = TRUE))
```

se invece mi chiedono:

Sulla base dei dati campionari, determinare un intervallo di confidenza al 99% per la frazione di elementi della popolazione caratterizzati da un livello alto del parametro gamma .

```
successi <- sum(campione$gamma == 'alto')
binom.test(successi, length(campione), conf.level = 0.99) # O in base a quello che mi chiedono faccio il test
```

Frequenza assoluta, relativa e cumulata (Dopo aver diviso in classi)

è lo stesso anche senza classi

- Assoluta: `freq_assoluta <- table(campione$classi_gamma)`
- Relativa: `frequenza_relativa <- prop.table(frequenza_assoluta)`
- Cumulata: `frequenza_cumulata <- cumsum(frequenza_assoluta)`

Media della distribuzoine in classi

Sapendo già come si calcola la frequenza assoluta e come si divide in classi

```
breaks <- c(0, 3.5, 9.2, 23, 80) # Sono le classi in cui è diviso gamma
centri_classi <- (head(breaks, -1) + tail(breaks, -1)) / 2
frequenze <- as.numeric(freq_assoluta) # Converto freq_assoluta in vettore numerico
media_classi <- sum(frequenze * centri_classi) / sum(frequenze)
```

Densità di frequenza

Sapendo già come calcolare la frequenza e convertirla in vettore numerico e che breaks sono le classi

```
ampiezze <- diff(breaks)
densita <- frequenze / ampiezze
```

Mediana della distribuzione in classi

```
N <- sum(frequenze)
cum_freq <- cumsum(frequenze)
i_med <- which(cum_freq >= N / 2)[1]
L <- breaks[i_med]
F_prev <- ifelse(i_med == 1, 0, sum(frequenze[1:(i_med - 1)]))
f_med <- frequenze[i_med]
h <- diff(breaks)[i_med]
mediana <- L + ((N/2 - F_prev) / f_med) * h
```

Regressione Lineare

Supponiamo di dover predire v3 partendo da v1 e di dover valutare il modello ottenuto

```
linmod <- lm(v3 ~ v1, data = campione)
summary(linmod) # Stampo un riassunto e commento i 2 R^2
plot(campione$v1, linmod$residuals)
abline(h=0)
```

Effettuare una predizione

Dopo aver allenato il modello:

```
nuovo_dato <- data.frame(v1 = 10)
predizione <- predict(linmod, newdata = nuovo_dato)
```

Cambio di base

- Esponenziale

```
expmod <- lm(log(v3) ~ v1, data = campione)
predizione <- predict(expmod, newdata = nuovo_dato) |> exp() # Quando genero uso exp per annullare log
```

- Radice quadrata

```
sqrtmod <- lm(sqrt(v3) ~ v1, data = campione)
predizione_sqrt <- predict(sqrtmod, new_data = nuovo_dato) # la variabile predetta è sqrt(v3)
predizione <- predizione_sqrt^2
```

Intervallo di confidenza per una proporzione

Supponiamo di avere:

- `x` : numero di successi
- `n` : numero totale di prove
- `conf.level` : livello di confidenza (es. 0.95 per 95%)

```
prop.test(x, n, conf.level = 0.95, correct = FALSE)
```

Intervallo di confidenza per una media

Hai:

- `x` : un vettore con i dati numerici
- `conf.level` : livello di confidenza

```
t.test(x, conf.level = 0.95)
```

Intervallo di confidenza per una frazione di elementi che soddisfano una condizione

È simile al caso della proporzione.

Supponiamo tu abbia un vettore `x` e voglia sapere la frazione di elementi che soddisfano una certa condizione, ad esempio `x > 10` .

```
# Calcolo la frazione
x <- c(...)  # il tuo vettore
condizione <- x > 10
x_successi <- sum(condizione)
n_totale <- length(x)

prop.test(x_successi, n_totale, conf.level = 0.95, correct = FALSE)
```

Nota su `correct = FALSE`

L'opzione `correct = FALSE` evita la **correzione di continuità di Yates** (utile per grandi campioni). Per campioni piccoli, potresti volerla attivare mettendo `TRUE`.

Determinare con un livello di significatività per una media

Supponiamo tu abbia un vettore `kappa` (es. una colonna in un dataframe):

- `mu = 0` : valore della media sotto l'ipotesi nulla
- `alternative = "two.sided"` : test bilaterale (la media può essere maggiore o minore di 0)
- `conf.level = 0.90` : corrisponde a livello di significatività **α = 0.10**

```
t.test(kappa, mu = 0, alternative = "two.sided", conf.level = 0.90)
```

Valutazione risultato

$$p = 0.1424 > 0.10 = \alpha$$

Se p è maggiore di α non possiamo rifiutare l'ipotesi nulla. **Non ci sono prove sufficienti** per dire che la media sia diversa da 0.

Determinare con un livello di significatività per una proporzione

Stessa cosa della media però con la proporzione

Su un campione di 120 clienti, 72 si sono detti soddisfatti del servizio. Determinare, con un livello di significatività del 1%, se **più del 50%** dei clienti è soddisfatto.

```
prop.test(x = 72, n = 120, p = 0.5, alternative = "greater", conf.level = 0.99)
```

Alternative

La scelta di `alternative` dipende dalla **domanda di ricerca**:

Domanda	`alternative`
È diverso da...?	`"two.sided"`
È maggiore di...?	`"greater"`
È minore di...?	`"less"`