R Cheat Sheet - Ermano Buikis

Array e Matrici

```
m<-matrix ( nrow = 4 , byrow = T )
dim(m)       # mostra la dimensione di m
```

Libreria in uso

```
if(!require(dplyr)) install.packages("dplyr")
library(dplyr)
```

Unire vettori

```
cbind(x,y)    # unisce i 2 array per colonna
rbind(x,y)    # unisce i 2 array per riga
```

Creazione Vettore

```
v<-c(1,2,3)
v<-c('nome1','nome2')
```

Accesso elemento

```
v[row][col]
```

Richiama manuale

```
? X    # per comando X
```

attributes(x)

Operazioni Vettori

```
2*v , 2+v , 2%v , 2-v
mean(v)      # media
sd(v)        # deviazione standard
var(v)       # varianza
v[n]         # accesso agli elementi
length(v)    # lunghezza
vettore.mean[i] <- mean( v[ , i ] ) # media della colonna i
```

Operazioni Matrici

```
M1 (*) M2           # moltiplicazione elemento per elemento
M1%*%M2             # moltiplicazione riga per colonna
```

Inizializzazione vettore

```
vector.mean <- array( 0, dim(M[2]) )
# compone un vettore pieno di 0 grande come la lunghezza di tutte le colonne
```

Accedere ad una feature di un DataFrame

```
dataframe$NomeColonna
```

Print

```
paste( stringa1, stringa2 ,.. ) # print con concatenazione di stringhe
```

LOOP WORKFLOW

- IF

```
if ( v[ n ] > 2 ) { comandi }
else { comandi_else}
```

- FOR

```
for ( i in 1:length(v)) {  comandi }
```

- WHILE

```
while( i < length(v) ) {
comandi,  i= i+1}
```

Salva Dati su File

```
write( vettore , file = "/directori/FileName.txt" , sep= "\t" , ncol =dim(vettore[2] )
```

Lettura Dati

```
input_data <- read.table("./fileName.txt" ,head=T , sep= " " )
```

Filtra - Witch

Estrae valori da un vettore con una condizione
Seleziona le posizioni degli elementi con condizione positiva

```
v<-c(1,2,3)
pos = whitch ( v > 2)  # witch e' il comando per il filtro
pos
>>> [ F , F, T]        # output
```

Filtra – Subset

Funzione che prende una matrice e gli applica 2 condizioni, per ottenere un sottoinsieme.

```
subset(x, …)

subset( input.data[ , 1 ] < 7) & (input.data[ ,2 ] ==" A " )

subset(x, subset, select, drop = FALSE, …)              # method for matrix

subset(x, subset, select, drop = FALSE, …)              # method for data.frame

examples
subset(airquality, Temp > 80, select = c(Ozone, Temp))
subset(airquality, Day == 1, select = -Temp)
       subset(airquality, select = Ozone:Wind)
```

Campionamento

```
sample(vettore_input, numero di campionamenti, replace=True/False)
seq( 5, 100, by=5 ) # Sequenza
range( start , stop, steps )  # Range
```

Sommario

```
numsummary( x , quantiles = c( 0, 0.25, 0.5 ,0.75, 1)
```

Affidabilita'

```
reliability( )
```

Tavola di contingenza

table( ) restituisce una distribuzione di frequeze delle variabili

Intervallo di confidenza

```
tgamma <-  qt( 1 - alpha/2 , df = n-1 )
limsup <- media + tgamma * sqrt( S^2 / n)
```

TEST STATISTICI

- **prop.test**
- **binom.test**
- **t.test**
  - t.test(x1, x2, paired =T) #2 campioni
  - t.test(x , mu= 3, alternative =" two sided") # 1 campione
- **z.test**
- **saphiro.test**
  - per verificare che la distribuzione sia normale
- **perarson.test**(x, adjust =T) # chi-squared
- **chisq.test()** # test Chi-square
- **anova**(x)
- **f.test**(x)
- **wilcox.test**(x1, x2 , var.equal = T) # Wilcoxon rank test
  - H0 l'accoppiamento delle variabili segue una distribuzione simmetrica attorno allo 0
- **var.test**(x, y)
  - F-Test dove F= S_x^2 / S_y^2 e df (*degree of freedom*) = n+m-2
  - vedere se hanno la stessa varianza, comparo i 2 campioni con var.test( )
- **TukseyHSD**
  - plot(TukeyHSD(c))
  - TukeyHSD(c)

Prefissi Test Statistici su R

- Geom
- Norm
- Unif
- Chisq
- pois
- exp
- binom
- f
- gamma

Tipi di distribuzioni

1. **d** densita' in un punto
2. **p** funzione di ripartizione in un pt
3. **q** quantile
4. **r** genera valore casuale della distribuzione

```
dnorm(1)          # calcola il valore della densità nel punto 1
dnorm(c(-1,0,1))  # calcola il valore della densità nei punti -1, 0 e 1
pnorm(1)          # calcola il valore della funzione di ripartizione in 1
qnorm(0.10)       # calcola lo 0.10-quantile, cioè il decimo percentile
```

esempi

```
dnorm(c(1,2) )
qnorm( 0.10 )
dbinom( K, N , P )
```

Calcola il coefficente binomiale

```
choose( N , K )
```

PLOT

qqplot : misura i quantili empirici / teorici in x/y

ppplot : misura le percentuali empiriche / teoriche in x/y

ggplot

```
ggplot( (data, estetica) + hist( width= 1) + xlab + ylab + labs   )
```

Unique

```
unique(x)
```

Grep

```
grep ( a , X )  # trova pattern A in X, entrambe stringhe
```

Sovrappone plots

```
lines(  )
```

Fix dati

```
fix()
```

Fitting distribuzione

```
a<- fitdist( x, "norm" )

plot(a)
```

Boxplot

```
boxpot(vettore_input,main='titolo',ylab='titolo asse y',xlab='titolo asse x')

summary(sample.tempo)
```

Caratteristiche

Il baffo inferiore, e superiore sono valori estremi. La linea in mezzo e' la mediana, che divide in 2 il rettangolo, sono i 2 quartili(25 e 25) percentili, quartile(25%) = indici di posizione della distribuzione che ripartiscono I dati con percentuali precisi sul totale. Gli outliers sono i punti sopra il baffo superiore, da non buttare dai dati, si cerca di indagare la ragione di questo discostamento.

Istogramma

```
his(vettore,main=' titolo')
```

Visualizzare piu' boxplot in uno stesso grafico

```
Import.data ← read.table('/../...', head=T,sep=' ')
b ← boxplot(valorix~valoriy,data=input.data, main' ', ylab='',xlab='',plot=0)
# non visualizza con plot=0
names=paste(b$names,'(n=',b$n,')',las=p)
# attacca con paste la sintassi fra '' con I valori trovati su names
```

Plot Distribuzione Binomiale

```
plot( x, y,( type=h#barre verticali, type=p#punti, type=l#linea),
lvd= spessore barre,
ylab,xlab,main='',
cec.main=0.3 #grandezza titolo,
col=red #colore)
```

Plot Distribuzione Poisson

```
dpois(k,lambda*k)
```

Approssimazione bin con gauss

```
lines(x,norm.sample,lwd=2,col="blue") SOVRAPPORRE 2 grafici
```

Salvare plot

```
png("Gauss_binom.png")
```

Implementare Funzioni

```
nome ← funzione(a,b)
{ return }

nome(c,d)

Confrontare il p-Value e' il valore di alfa= livello di significativita'
se a < p_value rigettiamo H0
```

VERIFICA IPOTESI T TEST

per due campioni → **paired=TRUE/FALSE**

per un campione **t.test(vettore,mu=x)** con media =3

con varianze diverse si usa sempre t.test con opzione **var.equal=FALSE**

- The data must be sampled from a normally distributed population (or populations in case of a two-sample test).
- For two-sample tests, the two populations must have equal variances (i.e. homogeneity of variances).
- Each score (or difference score for the paired t-test) must be independent of all other scores.

Inizializza dataframe

```
data<-data.frame(c(),c())
attach(data)
# da valore ai nomi di colonne o righe cosi' che quando le richiamo mi tira fuori il vettore
```

Covarianza

```
cov(x,y)
```

Regressione lineare

```
lm ( x~y , data = )                # 2 variabili indipendenti
lm ( x ~ ( y + z )/j , data = )    # 3 variabili indipendenti
```

Residui = distanze fra modello e punto sperimentali

Rapporto di devianze, sopra modello, sotto dato. Quando $R^2=1$ il modello coincide con l' osservato, ma devo andare a vedere adjusted R^2 .
Overfitting, sovra fitto i miei dati se aggiungo tanti predittori, piu' R^2 diventa buono e cresce avvicinandosi ad 1. Con l' adjusted calcola anche i gradi di liberta' di X e Y, per abbattere la dipendenza di R dal num di predittori.

Generare un istogramma con gaussiana associata fittando dnorm

```
x <- f/10
h<-hist(x, breaks=5, col="red",
       main="Istogramma della frequenza del peso medio”,
       xlab = 'grammi/fagiolo')
xfit<-seq(min(x),max(x),length=40)
yfit<-dnorm(xfit,mean=mean(x),sd=sd(x))
yfit <- yfit*diff(h$mids[1:2])*length(x)
lines(xfit, yfit, col="blue", lwd=4)
```

Chi-square test

```
       # Example 1 (see the slides "Test del CHi-quadro in R")
       observed = c(160,190,240,115,295)
       expected.frequencies = c(.15,.20,.25,.10,.30)
       expected = 1000 * expected.frequencies
       expected

       # Manually compute the value of chi-squared.
       chisq = sum((observed - expected)^2/expected)

       # Manually find the p-value of the test, that is  the probability of getting a chi-
       squared at least this large with 5-1 = 4 degrees of freedom.


       pchisq(c(3.9), df=4, lower.tail=FALSE)


       # Using a built-in function o R
       chisq.test(observed, correct=FALSE, p=expected.frequencies)
```

Goodness of fit to a Poisson

```
       Creating frequency table

       data <- data.frame(c(0:4), c(2962,382,47,25,4))
       names(data) <- c('n', 'frequency' )
       print(data)
```

Esempio di analisi statistica con R

```
attach(data)
N <- sum(frequency)
RF <- frequency/N
DF <- data.frame(data, round(RF, 5)) #arrotona fino alla quinta cifra
MEAN <- sum(RF * n)
VAR <- (sum(n^2*frequency) - N*MEAN^2)/(N-1) # else use (MEAN+MEAN^2/R)
DISP <- VAR/MEAN # over dispersion
THETA <- 1/DISP
R <- MEAN*THETA/(1-THETA) # MEAN = R(1-THETA)/THETA #TRALASCIARE
cbind(MEAN,VAR,DISP,THETA,R)
x = n
(E_poi = round(N * dpois(x, lambda=MEAN),5))  calcola poisson per ogni N

#x=n, dpois sul vettore x calcolami la densita' secondo poisson, con la media
#calcolata dai dati, lambda=MEAN, *N per avere freq assolute confrontabili con altre

barplot(matrix(c(frequency,E_poi),nr=2, byrow = TRUE), beside=T,
        col=c("aquamarine3","coral"),
        names.arg=x) # sulle c vi sono i valori delle n, per ogni X da freq

legend("topright", c("Observed","Expected Poisson"), pch=15,  osservata con freq attesa
da pois
       col=c("aquamarine3","coral"),
       bty="n")

CT = chisq.test(rbind(frequency,E_poi));    prende freq attese ed osservate
C -> CT$statistic;

pVal = CT$p.value
RES = frequency - E_poi
par(mfrow=c(1,3))
plot(x,RES)
SR <- sum(RES)
MSE <- mean(RES^2)
SUMMARY <- matrix(c(SR, C, pVal, MSE), byrow = TRUE)
colnames(SUMMARY) <- c("Poisson Fit")
rownames(SUMMARY) <- c("sum of residuals", "Chi-square Statistic", "p-value", "MSE")
```

ANOVA

```
attach(InsectSprays) # Upload the dataset
data(InsectSprays)
str(InsectSprays)
tapply(count, spray, mean)        # Means
tapply(count, spray, var)         # Variances
tapply(count, spray, length)      # Sample size
boxplot(count ~ spray)
                ***Run 1-way ANOVA***
aov.out = aov(count ~ spray, data=InsectSprays)
summary(aov.out)
                ***Multiple comaprison***
pairwise.t.test(count, spray, p.adjust="bonferroni")
TukeyHSD(aov.out)
```

Bar Chart Plot

```
# Create the data for the chart
H <- c(7,12,28,3,41)
M <- c("Mar","Apr","May","Jun","Jul")

# Give the chart file a name
png(file = "barchart_months_revenue.png")

# Plot the bar chart
barplot(H,names.arg=M,xlab="Month",ylab="Revenue",col="blue",
main="Revenue chart",border="red")

# Save the file
dev.off()
```

Rename

```
colnames(dataframe) <- c("x1", "x2", "x3", "x4", "x5")
```

Add boxplots to a scatterplot

```
par(fig=c(0,0.8,0,0.8), new=TRUE)
plot(mtcars$wt, mtcars$mpg, xlab="Car Weight",
ylab="Miles Per Gallon")
par(fig=c(0,0.8,0.55,1), new=TRUE)
boxplot(mtcars$wt, horizontal=TRUE, axes=FALSE)
par(fig=c(0.65,1,0,0.8),new=TRUE)
boxplot(mtcars$mpg, axes=FALSE)
mtext("Enhanced Scatterplot", side=3, outer=TRUE, line=-3)
```