Cheatography

R Cheat Sheet

by quantumrustler via cheatography.com/212850/cs/46340/

R Environment

`Ctrl + L` (Windows)	Clear command window
`ls()`	List objects in environment
`rm(obj)`	Remove object
`print( 'text')` `print(obj)`	Displays text or object

Operations and Special Characters

`+, -, *, /, ^`	Arithmetic operations
`%*%`	Matrix multiplication
`'`	Transpose
`==, !=, <, >, <=, >=`	Relational operators
`#`	Comment
`<-` or `=`	Assignment

Elementary Math Functions

`sqrt(x)`	Square root
`exp(x=3)`	Exponential of x
`abs(x=-1)`	Absolute value of x
`log(x= exp(1), b=exp(1))`	Logarithm with base b. If b is not specified, e is assumed by default

Vectors, Matrices, Arrays, Lists, Data Frames

`c(1,2,3)`	Combine values into vector
`m:n`	Sequence from m to n (can't do spacing)
`seq(fr om= 1,t o=1 0,by=2)`	Sequence with step. For decreasing step, by must be -ve
`seq(fr om= 3,t o=2 7,l eng th.o ut=40)`	Sequence with as many numbers specified
`rep(x= c(3 ,62 ,8.3 ), tim es= 3,e ach=2)`	Repeat values. The value for times provides the number of times to repeat x, and each provides the number of times to repeat each element of x.
`sort(x =c( 2.5 ,-1 ,-1 0,3.44 ),d ecr eas i ng =FALSE)`	Sort a vector in increasing or decreasing order
`length (x= c(3 ,2, 8,1))`	Determines how many entries exist in a vector given as the argument x
`myvec[1]` `myvec[c(1,3,5)]`	Retrieve specific elements from a vector
`myvec[-1]` `myvec[-c(1,3,5)]`	Delete elements by using negative versions of the indexes
`myvec[m:n]`	Retrieve elements from a vector with a sequence of indices from m to n
`prod(m yvec)`	Multiply all elements in a vector

Vectors, Matrices, Arrays, Lists, Data Frames (cont)

`matrix (da ta= c(- 3,2 ,89 3,0.17 ),n row =2, nco l=2 , by row =FALSE)`	Create a matrix filled in a column-by-column fashion
`rbind( 1:3 ,4:6)`	Bind together vectors as rows of a matrix
`cbind( c(1 ,4) ,c( 2,5 ),c (3,6))`	Bind together vectors as columns of a matrix

`dim(mymat)`
`nrow(m ymat)`
`ncol(m ymat)`
	Provides the dimensions of a matrix

`A[,n]`	Refers to the elements in all the rows of column n of the matrix A
`A[n,]`	Refers to the elements in all the columns of row n of the matrix A
`A[,m:n]`	Refers to the elements in all the rows between columns m and n of the matrix A
`A(m:n,)`	Refers to the elements in all the columns between rows m and n of the matrix A
`A[m:n,p:q]`	Refers to the elements in rows m through n and columns p through q of the matrix A.
Indexing can be done using individual indices in vectors. To delete or omit elements from a matrix, use negative indexes.	
`diag(x=3)`	Create an identity matrix of size 3 x 3
`diag(x=A)`	Identify the values along the diagonal of a square matrix
`t(A)`	Find the transpose of a matrix
`solve(A)`	Find the inverse of a matrix
`list(m atr ix( dat a=1 :4, nro w=2 ,nc ol= 2), c(T ,F, -T,T ),"h ell o")`	Create a list containing mixed object types. To name the components of a list as it's being created, assign a label to each component in the list command
`lst[[i]]`	Access the ith element of a list
`lst[1:2]`	Returns a sublist of selected elements
`names(lst)`	Name list components to make the elements more recognizable and easier to work with
`lst$name` `x[['na me']]`	Access element by name (or create new column)
`x$nested <- list(a =1:3)`	Add a nested list to an existing list

Vectors, Matrices, Arrays, Lists, Data Frames (cont)

`data.frame(person=c("Peter","Lois","Meg","Chris","Stewie"), age=c(42,40,17,14,1), gender=factor(c("M","F","F","M","M")), stringsAsFactors=TRUE)`	Create a data frame. stringsAsFactors is used to control automatic conversion of character strings to factors
`df[df$ gender == 'M', ]`	Logical Subset Subset rows where gender is M

Data frames are treated like matrices,
so you can also use functions like nrow(df).

Non-numeric Values

`TRUE` (or `T`) `FALSE` (or `F`)	Logical values
`any(mat)`	Returns TRUE if any of the logicals in the vector are TRUE and returns FALSE otherwise
`all(mat)`	Returns a TRUE only if all of the logicals are TRUE, and returns FALSE otherwise
`"This is a character string - "`	Character strings
`nchar( x=str)`	Returns the number of characters in a string. length(x=str) != nchar(x=str)
`cat("Hello", "worldd\b", ".\n", sep=" ")`	Sends output directly to the console screen and doesn't formally return anything
`paste("Hello", "world", ".", sep=" ")`	Concatenates and then returns the final character string as a usable R object
`substr(x=str, start=21, stop=27)`	Extracts a substring from x, starting at start and ending at stop
`sub(pattern="chuck", replacement="hurl", x=str)`	Replaces the first match of pattern in x with replacement
`gsub(pattern="chuck", replacement="hurl", x=bar)`	Replaces all matches of pattern in x with replacement
`factor(x=c("low", "medium", "high", "medium"))`	Converts a vector x into a categorical variable with labeled levels (similar to enums from other languages)

Non-numeric Values (cont)

`levels (x= myvec)`	Lists the categories (levels) in the factor x

Multidimensional Arrays

`array( dat a=1:24, dim=c(3, 4, 2))`	Creates a 3D array with 3 rows, 4 columns, and 2 layers
`array( dat a=r ep( 1:2 4,t ime s=3 ),d im= c(3 ,4, -2,3))`	Creates a 4D array with dimensions 3×4×2×3
`A[ , , n]`	All rows and columns in the n-th matrix (3rd dim) of A
`A[ , m, n]`	All rows in column m of the n-th matrix
`A[i, , ]`	All columns and layers of row i
`A[ , , , p]`	All rows, columns, and matrices in the p-th 4th dimension slice
`A[m:n, , , ]`	All columns and dimensions for rows m through n
`A[ , , m:n]`	All rows and columns for matrices m through n
`A[1:2, 2:3, 1, 1]`	A specific 2×2 submatrix from layer 1, 4th-dim slice 1

Statistics

`sum(xdata)`	Sum all elements in a vector
`mean(x data, na.rm= FALSE)`	Calculates the arithmetic mean
`median (xdata)`	Finds the median of a data
`table( xdata)`	Returns the frequencies
`xtab[x tab ==m ax( xtab)]`	Returns the mode, where xtab is a table of xdata
`min(xdata)`	Returns the smallest value
`max(xdata)`	Returns the largest value
`range( xdata)`	Returns the smallest and largest values
`round(x, n)`	Round to the specified number of decimal places (n)
`tapply(chickwts$weight, INDEX=chickwts$feed, FUN=mean)`	Applies mean to the numerical data for each grouping variable
`quanti le( x=x data, prob=0.8)` `quanti le( x=x data, prob=c (0.2 5, 0.5 ,0.75))`	Returns the quantile(s) of interest
`summar y(x data)`	Provides statistics automatically

`var(xdata)`
`sd(xdata)`
`IQR(xdata)`
	Direct R commands for computing measures of spread (variance, standard deviation, interquartile range)

Statistics (cont)

`cov(xd ata ,ydata)`	Computes the covariance between two numeric vectors
`cor(xd ata ,ydata)`	Computes the correlation coefficient between two numeric vectors
`plot(x, y, line="l ", xlab="x -ax is", yla b="y -ax - is")`	Creates a scatter plot of y versus x

Probability

Basic Probability Formulas

$Pr(A \cup B) = Pr(A) + Pr(B) - Pr(A \cap B)$	
$Pr(A \cap B) = 0$	If A and B are disjoint/mutually exclusive (cannot happen at the same time)
$Pr(A \cap B) = Pr(A) \times Pr(B)$	If A and B are independent (not related)
$Pr(A^C) = 1 - Pr(A)$	
$Pr(A \mid B) = Pr(A \cap B) / Pr(B)$	
$P(X > x) = 1 - P(X <= x)$	
cumsum(X.prob)	Calculates CDF of discrete RV
sum(X.prob*X.outcomes)	Calculates E[X] (X is discrete RV)
sum((X.outcomes*X.mean)^2* X.prob))	Calculates Var(X) (X is discrete RV) Alternative: $E[X^2] - (E[X])^2$
$F(x) = \int^x f(u)\ du$	CDF (continuous)

Plot Probabilities vs. Realizations

`barplot(height=X.prob, ylim=c(0,0.5), names.arg=X.outcomes, space=0, xlab="x", ylab="Pr(X = x)")`	PMF
`barplot(X.cumul, names.arg=X.outcomes, space=0, xlab="x", ylab="Pr(X <= x)")`	CDF (discrete)

Common Probability Distributions

X~Binomial(size, prob) (X is discrete RV)

dbinom(x=5,size=8,prob=1/6)	Calculates P(X = x) where x is no. of trials
sum(dbinom(x=0:5,size=8,prob=1/6)) pbinom(q=5,size=8,prob=1/6)	Calculates P(X <= q) where x is no. of trials
qbinom(p=0.95,size=8,prob=1/6)	Finds smallest x given P(X <= x) = p (inverse of CDF)

X~Pois(λ) (X is discrete RV)

Probability (cont)	
dpois(x=3,lambda=3.22)	Calculates P(X = x) where x is no. of events observed
Tip: P(X = 5) is meaningless so P(X < 5) = P(X <= 5)	
ppois(q=3,lambda=3.22)	Calculates P(X <= q) where q is no. of events
qpois(p=0.95,lambda=3.22)	Finds smallest x given P(X <= x) = p (inverse of CDF)
rpois(n=15,lambda=3.22)	Generates n random numbers from a Poisson distribution given lambda
X~Normal(μ, σ) (X is continuous RV)	
dnorm(x, mean, sd)	Returns the height of the normal distribution curve at x
pnorm(q, mean, sd) Default: μ = 0, σ = 1	Calculates P(X <= q) given μ and σ or P(Z <= z) if defaults are used
qnorm(p, mean, sd)	Finds smallest x given P(X <= x) = p (inverse of CDF)
qnorm(p, lower.tail=FALSE)	Finds smallest z given P(Z > z) = p Equal to P(Z <= z) = 1 - p
rnorm(n, mean, sd)	Generates n random numbers from a Normal distribution given μ and σ
QQ Plots and Histograms	
hist(chickwts$weight, main="", xlab="weight", xlim=c(xi,xf))	Draws a histogram of the given data
qqnorm(chickwts$weight, main="Normal QQ plot of weights")	Creates a QQ plot
qqline(chickwts$weight, col="gray")	Adds a reference line to the QQ plot