Cheatography

R programming Codes Cheat Sheet

by [deleted] via cheatography.com/77271/cs/18959/

Import Libraries to Read

library(readr)

library(ggplot2)

library(dplyr)

library(broom)

library(Tmisc)

library(caret)

library(caret)

library(splines)

library(party)

library(leaps)

library(glmnet)

Apply Functions

(m=matrix, a=array, l=list; v=vector, d=dataframe)

apply(x,index,fun) [input: m; output: a or l; applies function fun to rows/cols/cells (index) of x]

lapply(x,fun) [input l; output l; apply fun to each element of list x]

sapply(x,fun) [input l; output v; user friendly wrapper for lapply(); see also replicate()]

tapply(x,index,fun) [input l output l; applies fun to subsets of x, as grouped based on index]

Clustering

plot(1:nc, wss, type="b", xlab="Number of Clusters", ylab="Within groups sum of squares")}

wssplot <- function(data, nc=15, seed=1234){ wss <- (nrow(data)-1)*sum(apply(data,2,var)) for (i in 2:nc){ set.seed(seed) wss[i] <- sum(**kmeans(data, centers=i**)$withinss)}

GGplot

ggplot(mydata, aes(xvar, yvar)) + geom_point(aes(color=groupvar)) + geom_smooth(method="lm")

qplot(x = cty, y = hwy, data = mpg, geom = "point") [Creates a complete plot with given data, geom, and mappings. Supplies many useful defaults]

last_plot() [Returns the last plot]

ggsave("plot.png", width = 5, height = 5) [Saves last plot as 5' x 5' file named "plot.png" in working directory. Matches file type to file extension]

Setup

createDummyFeatures(obj=,target=,method=,cols=) [creates (0,1) flags for each non-numeric variable excluding **target**

**normalizeFeatures(obj=,target=,method=,cols=,range=,on.constant=)

center subtract mean

scale divide by std. deviation

standardize center and scale

range linear scale to given range

mergeSmallFactorLevels(task=,cols=,min.perc=) [combine infrequent factor levels into single merged level]

Basic Codes

read_csv("path/nhanes.csv")

View(df)

filter(df, ..,) [Filters data frame according to condition]

mean, median, range [na.rm=TRUE]

t.test(y~grp, data=df)

wilcox.test(y~grp, data=df)

anova(lmfit)

TukeyHSD(aov(lmfit)) [ANOVA Post-hoc pairwise contrasts]

xt <- xtabs(~x1+x2, data=df)

addmargins(xt)

prop.table(xt)

chisq.test(xt)

fisher.test(xt)

mosaicplot(xt)

factor(x, levels=c("wt", "mutant"))

relevel(x, ref="wildtype")

power.t.test(n, power, sd, delta)

power.prop.test(n, power, p1, p2)

tidy() augment() glance() [Model tidying functions in the broom package]

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

Cheatography

R programming Codes Cheat Sheet

by [deleted] via cheatography.com/77271/cs/18959/

Model Functions

aov(formula, data) [analysis of variance model]

lm(formula, data) [fit linear models]

glm(formula, family, data [fit generalized linear models]

nls(formula, data) [nonlinear least-squares estimates of the nonlinear model parameters]

lmer(formula, data) [fit mixed effects model]

(lme4); lme() or (nlme)

anova(fit, data...) [provides sequential sums of squares and corresponding F-test for objects]

contrasts(fit, contrasts = TRUE) [view contrasts associated with a factor]

contrasts(fit, how.many) <- value

glht(fit, linfct) [makes multiple comparisons using a linear function linfct (mutcomp)]

summary(fit) [summary of model, often w/ t-values]

confint(parameter) [confidence intervals for one or more parameters in a fitted model]

predict(fit,...) [predictions from fit]

Decision Tree

ctree(formula,data) [formula is a formula describing the predictor and response variables]

Data Information

is.na(x)	is.nan(x)
is.null(x)	is.array(x)
is.complex(x)	is.character(x)
is.data.frame(x)	is.numeric(x)
head(x)	tail(x)
summary(x)	str(x)
length(x)	dim(x)
dimnames(x)	attr(x,which)
nrow(x)	ncol(x)
NROW(x)	NCOL(x)
class(x)	unclass(x)

Data Splitting and Manipulating

createDataPartition(y,p=0.8) [createDaIt splits a vector 'y' with 80 percent data in one part and 20 percent in other parttaPartition(y,p=0.8)]

trainControl(summaryFunction = <R function>, classProbs = <logical>) [It is used for controlling training parameters like resampling, number of folds, iteration etc.]

densityplot.rfe(x,data,...) [Lattice functions for plotting resampling results of recursive feature selection]

featureplot(x,y,plot...) [A shortcut to produce lattice plots]

Polynomial regression

$medv = b0 + b1 * lstat + b2 * lstat^2$

lm(medv ~ lstat + I(lstat^2), data = train.data)

lm(medv ~ poly(lstat, 2, raw = TRUE), data = train.data)

Spline Model

spline(x,y) [cubic spline interpolation]

splineKnots(object)

knots <- quantile(train.data$lstat, p = c(0.25, 0.5, 0.75))

Step-wise Selection

null<- lm(Formula~1, data=dtrain)

full<-lm(Formula~.,data=dtrain)

step(null, scope=list(lower=null, upper=full), direction="forward")

step(full, scope=list(lower=full, upper=null), direction="backward")

Preprocessing

Transformations, filters, and other operations can be applied to the predictors with the **preProc** option.

train(, preProc = c("method1", "method2"), ...)

train determines the order of operations; the order that the methods are declared does not matter.

recipes package has a more extensive list of preprocessing operations.

Sponsored by **Readable.com**
Measure your website readability!
https://readable.com