R Commands: Quick Reference Sheet[1]

to accompany Statistics: Unlocking the Power of Data by Lock[5]

CHAPTER 1

Loading Data	
Load a dataset from a google doc[2]	`google.doc("key")#key: between key= and # in url`
Load a dataset from the textbook	`data(dataname)`
Help for textbook datasets	`?dataname`
Type in a variable	`variablename = c(3.2, 3.3, 3.1)`
Variables	
Extract a variable from a dataset	`dataname$variablename`
Attach a dataset	`attach(dataname)`
Detach a dataset	`detach(dataname)`
Subsetting Data	
Take a subset of a dataset	`subset(dataname, condition)`
Random Sample	
Taking a random sample of size n	`sample(dataname, n) #use for data or variable`
n random integers 1 to max	`sample(1:max, n)`

CHAPTER 2

One Categorical (x)	
Frequency table	`table(x)`
Proportion in group A	`mean(x == "A")`
Pie chart	`pie(table(x))`
Bar chart	`barplot(table(x))`
Two Categorical (x1, x2)	
Two-way table	`table(x1, x2)`
Proportions by group	`mean(x1=="A"~x2)`
Difference in proportions	`diffProp(x1=="A"~x2)`
Segmented bar chart	`barplot(table(x1, x2), legend=TRUE)`
Side-by-side bar chart	`barplot(table(x1,x2),legend=TRUE,beside=TRUE)`
One Quantitative (y)	
Mean	`mean(y)`
Median	`median(y)`
Standard deviation	`sd(y)`
5-Number summary	`summary(y)`
Percentile	`percentile(y, 0.05)`
Histogram	`hist(y)`
Boxplot	`boxplot(y)`
One Quant. (y) and One Cat. (x)	
Means by group	`mean(y ~ x)`
Difference in means	`diffMean(y ~ x)`
Standard deviation by group	`sd(y ~ x)`
Side-by-side boxplots	`boxplot(y ~ x)`

[1] First time only, run `source("/shared/kari.lock.morgan@gmail.com/Lock5.R")`

[2] For your own google spreadsheet, within the google spreadsheet you first have to do File -> Publish to Web -> Start Publishing.

Two Quantitative (y1, y2)
Scatterplot
Correlation
	
	`plot(y1, y2)`
	`cor(y1, y2)`

Labels
Add a title
Label an axis
	`#optional arguments for any plot:`
	`main = "title of plot"`
	`xlab = "x-axis label", ylab = "y-axis label"`

CHAPTER 3

Repeat Code 1000 Times	`do(1000)*`
Sampling Distribution for Mean	`do(1000)*mean(sample(y, n))`
Bootstrap Distribution for Mean	`do(1000)*mean(sample(y, n, replace=TRUE))`

Generating a Sampling Distribution for any Statistic	`do(1000)*{`
	`  samp = sample(pop.data, n)`
	`  statistic(samp$var1, samp$var2)`
	`}`

Manually Generating a Bootstrap Distribution for any Statistic	`do(1000)*{`
	`  boot.samp = sample(data, n, replace=TRUE)`
	`  statistic(boot.samp$var1, boot.samp$var2)`
	`}`

Using a Bootstrap Distribution	`hist(boot.dist)`
	`sd(boot.dist)`
	`percentile(boot.dist, c(0.025, 0.975))`

Generate a Bootstrap CI	`bootstrap.interval(var1, var2) #level = .95`

CHAPTER 4

Randomization Statistic:
Shuffle one variable (x)
Proportion
Mean
	
	`shuffle(x)`
	`coin.flips(n, p)`
	`mean(sample(y + shift, n, replace=TRUE))`

Randomization Distribution	`do(1000)*one randomization statistic`

Finding p-value from a randomization distribution:
Lower-tailed test
Upper-tailed test
Two-tailed test
	`#rand.dist = randomization distribution`
	`#obs.stat = observed sample statistic`
	`tail.p(rand.dist, obs.stat, tail="lower")`
	`tail.p(rand.dist, obs.stat, tail="upper")`
	`tail.p(rand.dist, obs.stat, tail="two")`

Randomization Test via Reallocating	`randomization.test(y, x) #null = for one var` `#tail="lower", "upper", "two"`

CHAPTER 5

Normal Distribution:
Find a percentile for N(0,1)
Find the area beyond z on N(0,1)
Find percentiles or area for any normal
	`#tail="lower", tail="upper", tail="two"`
	`percentile("normal", 0.10)`
	`tail.p("normal", z, tail="lower")`
	`#add the optional arguments mean=, sd=`

CHAPTER 6

Normal Distribution: Find a percentile for N(0,1) Find the area beyond z on N(0,1)	`percentile("normal", 0.10)` `tail.p("normal", z, tail="lower")`
t-Distribution: Find a percentile for t Find the area beyond t	`percentile("t", df = 20, 0.10)` `tail.p("t", df = 20, t, tail="lower")`
Inference for Proportions: Single proportion Difference in proportions	`prop.test(count, n, p0) #delete p0 for CI` `prop.test(c(count1, count2), c(n1, n2))`
Inference for Means: Single mean Difference in means	`t.test(y, mu = mu0) #delete mu0 for CI` `t.test(y ~ x)`

Additional arguments
p-values using tail.p
p-values using prop.test or t.test
Intervals using prop.test or t.test
	`#tail="lower", "upper", "two"`
	`#alternative="two.sided", "less", "greater"`
	`#conf.level = 0.95 or confidence level`

CHAPTER 7

Chi-Square Distribution Find the area above χ^2 stat	`tail.p("chisquare", df = 2, stat, tail="upper")`
Chi-Square Test Goodness-of-fit Test for association	`chisq.test(table(x)) #if null probabilities not equal, use p = c(p1, p2, p3) to specify` `chisq.test(table(x1, x2))`
Randomization Test Goodness-of-fit Test for association	`chisq.test(table(x), simulate.p.value=TRUE)` `chisq.test(table(x1, x2), simulate.p.value=TRUE)`

CHAPTER 8

F Distribution Find the area above F-statistic	`tail.p("F", df1=3, df2=114, F, tail="upper")`
Analysis of Variance	`summary(aov(y ~ x))`
Pairwise Comparisons	`pairwise.t.test(y, x, p.adjust="none")`

CHAPTER 9

Simple Linear Regression
Plot the data
Fit the model
Give model output
Add regression line to plot
	
	`plot(y ~ x) # y is the response (vertical)`
	`lm(y ~ x)  # y is the response)`
	`summary(model)`
	`abline(model)`

Inference for Correlation	`cor.test(x, y)` `#alternative = "two.sided", "less", "greater"`

Prediction
Calculate predicted values
Calculate confidence intervals
Calculate prediction intervals
Prediction for new data
	
	`predict(model)`
	`predict(model, interval = "confidence")`
	`predict(model, interval = "prediction")`
	`predict(model, as.data.frame(cbind(x=1)))`

CHAPTER 10

Multiple Regression
Fit the model
Give model output
	
	`lm(y ~ x1 + x2)`
	`summary(model)`

Residuals
Calculate residuals
Residual plot
Histogram of residuals
	
	`model$residuals`
	`plot(predict(model), model$residuals)`
	`hist(model$residuals)`

Prediction
Calculate predicted values
Calculate confidence intervals
Calculate prediction intervals
Prediction for new data
	
	`predict(model)`
	`predict(model, interval = "confidence")`
	`predict(model, interval = "prediction")`
	`predict(model,as.data.frame(cbind(x1=1,x2=3)))`