ST1131 Rcode CheatSheet

Chin Sek Yi

2024-11-11

Table of Contents

Hope this cheatsheet helps you with your learning in ST1131! 🌟
Please reach out to me if you found a bug!

Prerequisite

Reserve letters: Don't use these alphabets as your variable names!

```
c q s t C D F I T
```

Clear Workspace

```
rm(list=ls()) #clear everything

x = 1:5
rm(list=c("x")) #clear one variable only
```

Logical Expressions

```
==   #Equality
!=   #Inequality
<    #Less than
>    #Greater than
<    #Less than or equal to
>=   #Greater than or equal to
y&x  #and
y|x  #or
!y   #not y
```

Data Structures

Vector, matrix, dataframe, list

```
vector= c(1,2,3)

mat = matrix(1:25, 5, 5)    #matrix(range, col, row, byrow = TRUE)

df = data.frame(A = c("apr", "may","jul"),
```

```
                B = c(1,2,3))

list = list("Cat", c(13,3,31), FALSE, 51.11, 23.4)
```

Seq, rep and sample:

```
#sequence
x = seq(1,5,1)            #1 2 3 4 5
x = seq(1,2,length = 3)   #1.0 1.5 2.0

#repeat a sequence
x = c(1,2,3)
rep(x, times = 3)         #1 2 3 1 2 3 1 2 3
rep(x, times = c(1,2,3))  #1 2 2 3 3 3

#sample
set.seed(10)              #for reusability
sample(1:6, size = 5)
sample(1:6, size = 5, replace = TRUE)

#create an empty array of size n
numeric(n)                #[1] 0 0 0 0 0 0 0 0 0 0
```

Loop

```
# for loop
x = c(1, 2, 3, 4, 5)

for (i in x) {
  print(i)
}

# while loop
i = 1
while (i <= 5) {
  print(i)
  i = i + 1
}
```

Dataframe methods

```
#read in df
df = read.csv(filename,sep=",",header=TRUE col.names=names) #names=c("colname
1","colname2")
df = read.csv()
```

```r
#df inspection
class(df)    #"data.frame"
names(df)    #column names
colnames(df) #column names
dim(df)      #number of rows/columns for matrix & df only
length(df)   #number of columns
nrow(df)     #number of rows
ncol(df)     #number of columns

head(df)
str(df)      #internal structure of df

#select a column
df[,"Height"]
df$Height

#or
attach(df)
Height
```

Dataframe operations

```r
colnames(df)[1] = "another name"   #change column name

df["Height">150 & "Weight"< 50, ]  #extract rows based on condition #remember
the comma!!
data[Gender == "M" & HW == "A",]

df[order(df$col), ]                #arrange rows based on values in column
df[order(rev(df$col)), ]           #for reverse

df = df[, c(5, 4, 1, 2, 3)]        #reorder columns by index or name
```

Table

```r
table(gender)
prop.table(table(gender))          #probability table

table(pmh,cancer)                  #contingency table

prop.table(tab,"pmh")*100          #row-wise probability
prop.table(tab,"cancer")*100       #col-wise probability
```

Factor

Rmbr to attach after factor!! Otherwise you'll get `levels(x) = null`

```
df$x = factor(df$x)
attach(df)

levels(x)                          #get the levels in x
levels(x) = c("name1","name2")     #change the levels name in x
```

or

```
df$x = as.factor(x)
attach(df)
```

Summary statistics

```
summary(df)

# Measures of Central Tendency
median()
mean()
mode()

# Measures of Spread and Variability
sd()
var()
range(), max(), min()
IQR()
quantile(df, c(0.25, 0.75))  # Quartiles

# Measures of Association
cor(x,y)
```

Plot

Barplot

What to report?

- Heights of the bars (counts or frequencies)
- Comparison of categories (which category is larger/smaller)
- Presence of any skew or imbalance in the data

```
barplot(data,
        xlab = "",
        ylab ="",
        main ="",
        beside = TRUE,  #bars are beside each other. Otherwise, they're stack
ed
        ylim = c(0,70), #the range of the y-axis. same for xlim = c()
        col = 5)        #colour
```

Boxplot

What to report?

- median, IQR, outliers
- outliers: number, which side, large/small

```
boxplot(age ~ cancer)
```

Outliers

```
boxplot(df$col)$out                         #gives values of outliers

out = which(boxplot(df$col)$out)            #give index of outliers
out = which(df$col %in% boxplot(col)$out)

new_df = data[!data %in% data[out]]         #create new df without outliers
```

Groups

```
grp = Boxplot()$group
which(grp == 1)
boxplot(age ~ cancer)$out[which(grp == 1)]  #give values of outliers in grp 1
```

Histogram

What to report?

- data cluster (where?)
- gaps (where?)

- deviation
- peak (how many? where?)
- symmetric/skewed

```
hist(df$col,
     freq = FALSE,  #to plot density or density = TRUE
     breaks = 10)   #affects the number of columns
```

Scatterplot

What to report?

- increasing/decreasing trend
- strength of trend. Can use `cor(x,y)` to confirm
- obvious outliers
- constant/increasing/decreasing variance

```
plot(x,y)

abline(h=1)                #to plot a horizontal line at y=1
abline(v=1)                #to plot a vertical line at v=1
abline(lm(y~x, data = df)) #to plot a best-fit line
```

Confidence Interval

for sig level 0.95

```
CI = c(p - qnorm(0.975) * sqrt(p * (1 - p) / n),    #n = sample size
       p + qnorm(0.975) * sqrt(p * (1 - p) / n))
```

Distributions

Overall:
Distributions:

1) Binomial dist
2) Normal dist
3) T dist

Tests:

1) Variance test
2) Hypothesis test (T.test)
3) Shapiro wilk test

Relevant Rcodes:

1) rbinom(), pbinom(), qbinom()

2) rnorm(), pnorm(), qnorm()

3) t.test()

4) shapiro.test()

5) wilcox.test()

6) qqnorm(), qqline()

Binomial

- rbinom(n, size, prob)
- pbinom(q, size, prob, lower.tail = TRUE)
- qbinom(p, size, prob, lower.tail = TRUE)

```r
# Generate 5 random numbers from a Binomial distribution with 10 trials and 0
.5 probability of success
rbinom(5, 10, 0.5)

# Calculate cumulative probability for 0.1 successes in 10 trials with a 0.5
probability of success
pbinom(0.1, 10, 0.5)

# Find the quantile for a 0.5 probability in 10 trials with a 0.5 probability
of success
qbinom(0.5, 10, 0.5)
```

Normal

- rnorm(n, mean = 0, sd = 1)
- pnorm(q, mean = 0, sd = 1, lower.tail = TRUE)
- qnorm(p, mean = 0, sd = 1, lower.tail = TRUE)

```r
# Generate 5 random numbers from a Normal distribution with a mean of 10 and
standard deviation of 2
rnorm(5, mean = 10, sd = 2)
```

```r
# Calculate cumulative probability for a value of 12 in a Normal distribution with a mean of 10 and standard deviation of 2
pnorm(12, mean = 10, sd = 2)

# Find the quantile for a probability of 0.25 in a Normal distribution with a mean of 10 and standard deviation of 2
qnorm(0.25, mean = 10, sd = 2)
```

Variance test, aka F-test

- if `variance_test$p.value` <= 0.05
  - Reject the null hypothesis: Population variance is not equal to the specified value.
- if `variance_test$p.value` > 0.05
  - Do not reject the null hypothesis: Insufficient evidence to conclude a difference in population variance.

```r
variance_test = var.test(data)

variance_test$p.value #get p-value
```

T.test (two-sample, Equal variance)

Assumptions:

- A quantitative response variable for two groups.
- Two independent samples, either from random sampling or randomized experiment.
- The population variance of each group is the same. Use `var.test()`.
- The population distribution of each group is approximately Normal.
  - If n is large then this assumption is not crucial.

Degrees of freedom:

- dof = n1 + n2 - 2 for independent two-sample t-test

Conclusion:

- E.g. Since p-value is smaller than 0.05, we reject H0 and conclude that the mean of sampleA is significantly different from the mean of sampleB at the 0.05 significance level.

Get **test statistic** and **p-value** from `summary(M1)`:

```r
t.test(sampleA, sampleB, alternative="two.sided/less/greater", var.equal = TRUE, conf.level= 0.95)
```

T.test (two-sample, Unequal variance), aka Welch's t-test

Same as above, except that `var.equal = FALSE`

```r
t.test(sampleA, sampleB, alternative="two.sided/less/greater", var.equal = FALSE, conf.level= 0.95)
```

T.test (Dependent samples)

Test is similar to one-sample t-test.

Conclusion:

- E.g. Since p-value is smaller than 0.05, we reject H0 and conclude that the mean is significantly different from {test value} at 0.05 significance level.

```r
t.test(sampleBefore, sampleAfter, alternative="two.sided/less/greater", paired = TRUE, conf.level= 0.95)
```

Calculation for **test statistic** (t.score):

```r
sample.mean = mean(x)
sample.sd = sd(x)
n = length(x)

t.score = (sample.mean-500)/(sample.sd/sqrt(n)) #500 is the test value. It will be given by the question.
```

Linear regression

To build a linear regression model, we need to include significant regressors, and ensure that our model is significant. This is to ensure our regressors have a true impact on the prediction of the response variable. Also, we need to make sure that the assumptions are met. In other words, check if the model is adequate. Then, check that goodness of fit of your model.

Table of Content:

  - summary(M1)
- **9.2 Tests for significance**
  - t-test
  - f-test
- **9.3 Model Assumptions/Adequacy:**
  - data is obtained by randomisation
  - relationship between X and Y is linear
  - $\varepsilon \sim N(0,\sigma 2)$ -> equal variance and normality
- **9.4 Goodness of fit**
  - R2 and adj R2

Building a linear model

Assuming you have a linear regression model named `M1`

```
M1 = lm(y~x, data=df)
summary_model = summary(M1)
```

99% CI of mean of y at x=x0 predict() -> generic function for any model

```
newdata = data.frame(x = x0)
predict(M1, newdata, interval="confidence", level=0.95)
```

99% CI of mean of y at x=x0 predict.lm() -> specifically for linear models created using the lm()

```
newdata = data.frame(x = x0)
predict.lm(m1, newdata, interval = "confidence", level=0.99)
```

Information to extract from `summary_model`:

```
#R-squared and Adjusted R-squared
summary_model$r.squared              # Extract the R-squared value
summary_model$adj.r.squared

#Residuals
summary_model$residuals              #the differences between the observed an
d predicted values.

#Degrees of freedom for the model
summary_model$df

#Information about the individual predictors
summary_model$coef
summary_model$std.error
summary_model$t.value
summary_model$p.value
```

```r
confint(M1, level=0.95)                #Confidence Intervals for Model Parameters
```

Tests for significance

- 9.2.1 T.test
- 9.2.2 F.test

T.test (for linear regression)

Purpose:

- for testing the significance of one regressor (or one coefficient)

Assumptions:

- data is obtained by randomisation
- relationship between X and Y is linear
- $\varepsilon \sim N(0,\sigma 2)$

Hypothesis:

- H0 : Beta1 = 0 or H0 : variable X is not significant.
- H1 : Beta1 ≠ 0 or H0 : variable X is significant.

Conclusion:

- Conclude whether the slope Beta1 significantly different from 0 at a pre-specified sig level.

- E.g. Since p-value is smaller than 0.05, we reject H0 and conclude that Beta1 is significantly different from 0 at the 0.05 significance level.

Get **test statistic** and **p-value** from `summary(M1)`:

```r
M1 = lm(y~x, data = df)
summary(M1)
```

F.test (for linear regression)

Purpose:

- To test if the whole model is significant

Hypothesis:

- H0 : all the coefficients, except intercept, are zero

- H1 : at least one of the coefficients, except intercept, are zero

Note:

In the simple model, the t-test to test the significance of the slope and the F-test to test the significance of model have the same p-value. Because simple model ony have one regressor.

Conclusion:

- if($p < 0.05$):
  - Data provide very strong evidence that the built model is significant. Thus, the whole model is significant.
- if($p > 0.05$):
  - regressor(s) used in the model is/are not significant. Thus, the whole model is not significant.

Get **test statistic** and **p-value** from `summary(M1)`:

```
M1 = lm(y~x, data = df)
summary(M1)
```

Check assumptions/adequacy

Assumptions:

- **Randomisation**: data is obtained by randomisation
- 9.3.1 **Linearity**: relationship between X and Y is linear
- 9.3.2 **Normality**: ensure residuals are normally distributed
- 9.3.3 **Equal variance**: $\varepsilon \sim N(0,\sigma 2)$
- 9.3.4 **Outliers & Influential points**

Linearity assumption

Scatterplot of Y~X

- if linear -> linearity assumption is met

```
plot(y~x)
```

Suggested fixes:

- Linearity assumption violated: add higher order terms in X, e.g. X2 to the model then plot() again to see if the linear regression assumptions are now met.

Normality assumption

Get residuals:

```
M1 = lm(y~x, data = car)

raw.res = M1$res     # lists all the raw residuals of model M
SR = rstandard(M1)   # lists all the standardized residuals of model M1
```

Residual plots:

- Histogram: `hist()`
  Is the histogram approximately normal or skewed?

```
hist(SR,
     prob = TRUE,
     main = "Histogram for Standard Residuals")
```

- `qqplot`
  Comment on the tails.

```
qqnorm(SR)
qqline(SR)
```

- `shapiro.test()`
  If(p < 0.05) -> reject H0 and conclude that x is not normal.
  - Not in lecture notes, so this test is optional.

```
shapiro.test(SR)
```

Equal variance assumption

1. Scatterplot for Y~X

- if variance is constant, the equal variance assumption is met.
- otherwise, the assumption is violated and the linear model is inadequate.

```
plot(y~x)
```

Suggested fixes:

- Variance not constant: transform the response variable by taking ln(Y), 1/Y, sqrt(Y).
- Then `plot()` again to see if the linear regression assumptions are now met.

2. Scatterplot for `plot(SR~Y)` and `plot(SR~X)`

We expect to see the points scatter randomly about 0, within the interval (-3, 3).
If there's a **funnel shape**, constant variance assumption is violated.

```
plot(SR~Y)
plot(SR~X)
```

```r
abline(h = 3) #to draw horizontal line at SR=3
abline(h = -3)
```

optional
To make abline dashed and slightly transparent `abline(h =3, col = adjustcolor("black", alpha = 0.5), lty = 2) abline(h =-3, col = adjustcolor("black", alpha = 0.5), lty = 2)`

Outliers and influential points

```r
# Residuals
raw.res <- residuals(M1) #or raw.res = M1$res
SR <- rstandard(M1)

# Check for outliers (values outside of -3 to 3)
outliers <- which(SR < -3 | SR > 3)

# Check for influential points using Cook's distance
cooksd <- cooks.distance(M1)
which(cooksd>1) # index of influential point
```

Goodness of fit

R2:

- the proportion of total variation of the response (about the sample mean) that is explained by the model.
- has the deficiency that the more variables we get, the larger it gets. This is not good as we could simply add higher powers of a particular explanatory variable until we get a maximal possible R2. Hence it is not reasonable to compare the fit of two models, using R2. Use **adj R2** instead.

Get **R2** and **adj R2** from:

```r
M1 = lm(y~x, data=df)
summary_model = summary(M1)

#R-squared and Adjusted R-squared
summary_model$r.squared
summary_model$adj.r.squared
```

This is the end!
All the best to your practicals and finals!
Do reach out to me if you have any questions!!
><