Lesson 7 R Activity 1

Rick Davila

5/8/2020

Lesson 7 - Install packages

Install necessary packages using library()

Perform data housekeeping - upload, name columns, display to make sure it reads properly, etc.

```
knitr::opts_chunk$set(echo = TRUE)

library(e1071)
library(xtable)
library("xlsx") # Needed to read data
```

```
## Warning: package 'xlsx' was built under R version 4.0.3
```

```
library(psych) # For geometric mean in Example 5.3
```

```
## Warning: package 'psych' was built under R version 4.0.3
```

```
rm(list = ls())
```

Read data file (data-ex-5-1.xlsx)

```
ex5_1 <- read.xlsx("data-ex-5-1.xlsx",
                   sheetIndex = 1,
                   colIndex = c(2,3),
                   as.data.frame = TRUE,
                   header = TRUE)
```

Assign labels to data columns using names() and attach() commands

```
names(ex5_1) <- c("Usage", "Demand")
attach(ex5_1)
```

Output data to make sure it reads properly

```
out <- as.data.frame(c(ex5_1))
colnames(out) <- c("Usage", "Demand")
tab <- (xtable(out, digits=c(0,0,2)))
print(tab, type="html")
```

	Usage	Demand

1	679	0.79
2	292	0.44
3	1012	0.56
4	493	0.79
5	582	2.70
6	1156	3.64
7	997	4.73
8	2189	9.50
9	1097	5.34
10	2078	6.85
11	1818	5.84
12	1700	5.21
13	747	3.25
14	2030	4.43
15	1643	3.16
16	414	0.50
17	354	0.17
18	1276	1.88
19	745	0.77
20	435	1.39
21	540	0.56
22	874	1.56
23	1543	5.28
24	1029	0.64
25	710	4.00
26	1434	0.31
27	837	4.20
28	1748	4.88
29	1381	3.48
30	1428	7.58
31	1255	2.63
32	1777	4.99
33	370	0.59
34	2316	8.19
35	1130	4.79
36	463	0.51
37	770	1.74
38	724	4.10
39	808	3.94
40	790	0.96
41	783	3.29
42	406	0.44
43	1242	3.24
44	658	2.14
45	1746	5.71
46	468	0.64

47	1114	1.90
48	413	0.51
49	1787	8.33
50	3560	14.94
51	1495	5.11
52	2221	3.85
53	1526	3.93

```
# Output data structure and dimensions
str(ex5_1)
```

'data.frame': 53 obs. of 2 variables: $ Usage : num 679 292 1012 493 582 … $ Demand: num 0.79 0.44 0.56 0.79 2.7 3.64 4.73 9.5 5.34 6.85 …

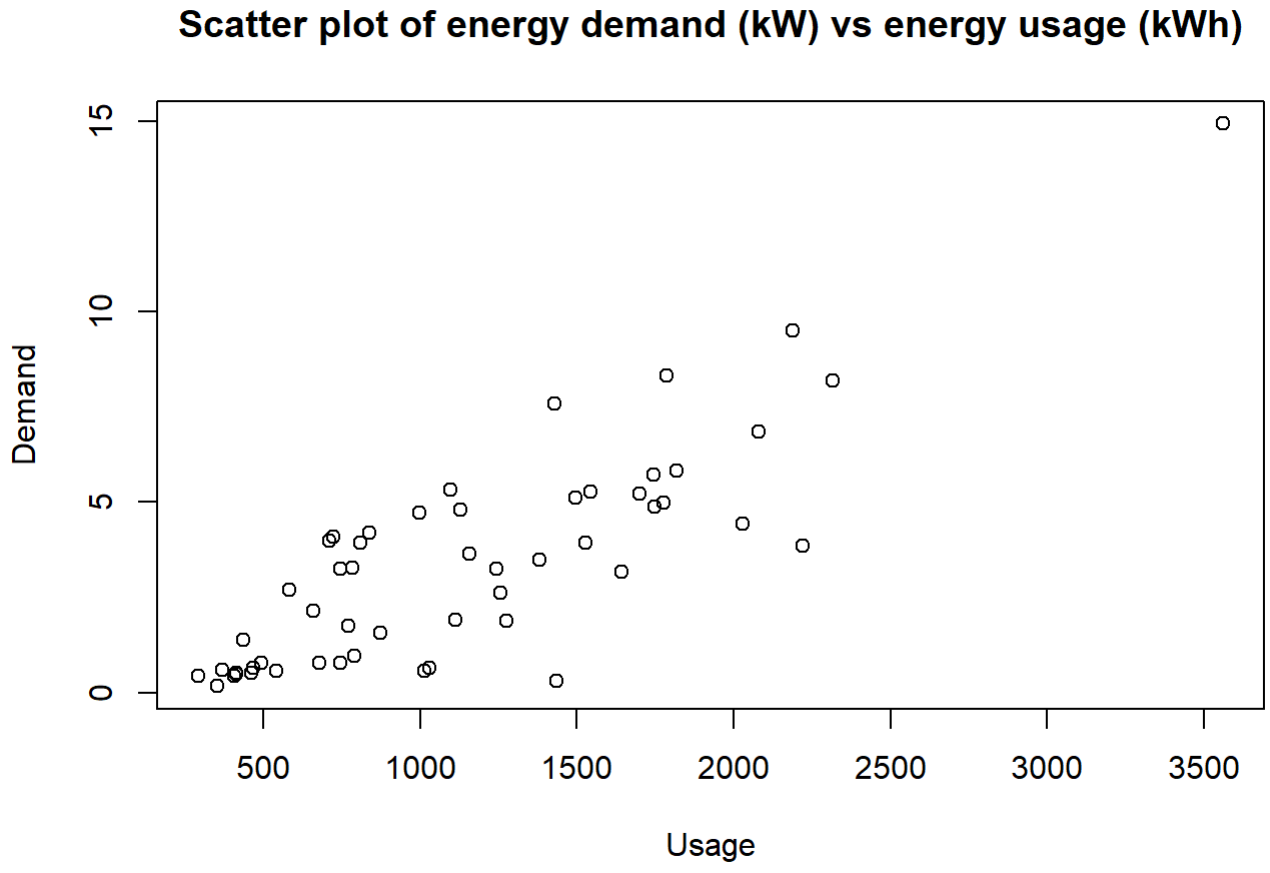
```
dim(ex5_1)
```

[1] 53 2

Example 5.1 (p. 173-175)

Create scatterplot of data

```
plot(Usage,Demand, main = "Scatter plot of energy demand (kW) vs energy usage (kWh)")
```



Create linear model and display ANOVA table. Compare to least squares fit on p. 173 and ANOVA table on p. 175

```
# Obtain regression estimates using lm command
model <- lm(Demand ~ Usage)
beta_0 <- model$coefficients[1]
beta_1 <- model$coefficients[2]

# show beta estimates, std. error, tvalues and p-values
xtable(model)
```

	Estimate <dbl>	Std. Error <dbl>	t value <dbl>	Pr(>|t|) <dbl>
(Intercept)	-0.831303660	0.441612128	-1.882429	6.548865e-02
Usage	0.003682843	0.000333897	11.029877	4.106229e-15

2 rows

The equation for the linear regression model:

$$\hat{y} = (-0.8313037) + (0.0036828)x$$

```
# ANOVA table
# summary(aov)
out <- anova(model)

xtable(out)
```

	Df <int>	Sum Sq <dbl>	Mean Sq <dbl>	F value <dbl>	Pr(>F) <dbl>
Usage	1	302.6331	302.633136	121.6582	4.106229e-15
Residuals	51	126.8660	2.487569	*NA*	*NA*

2 rows

Create plot of R-Student values versus fitted values

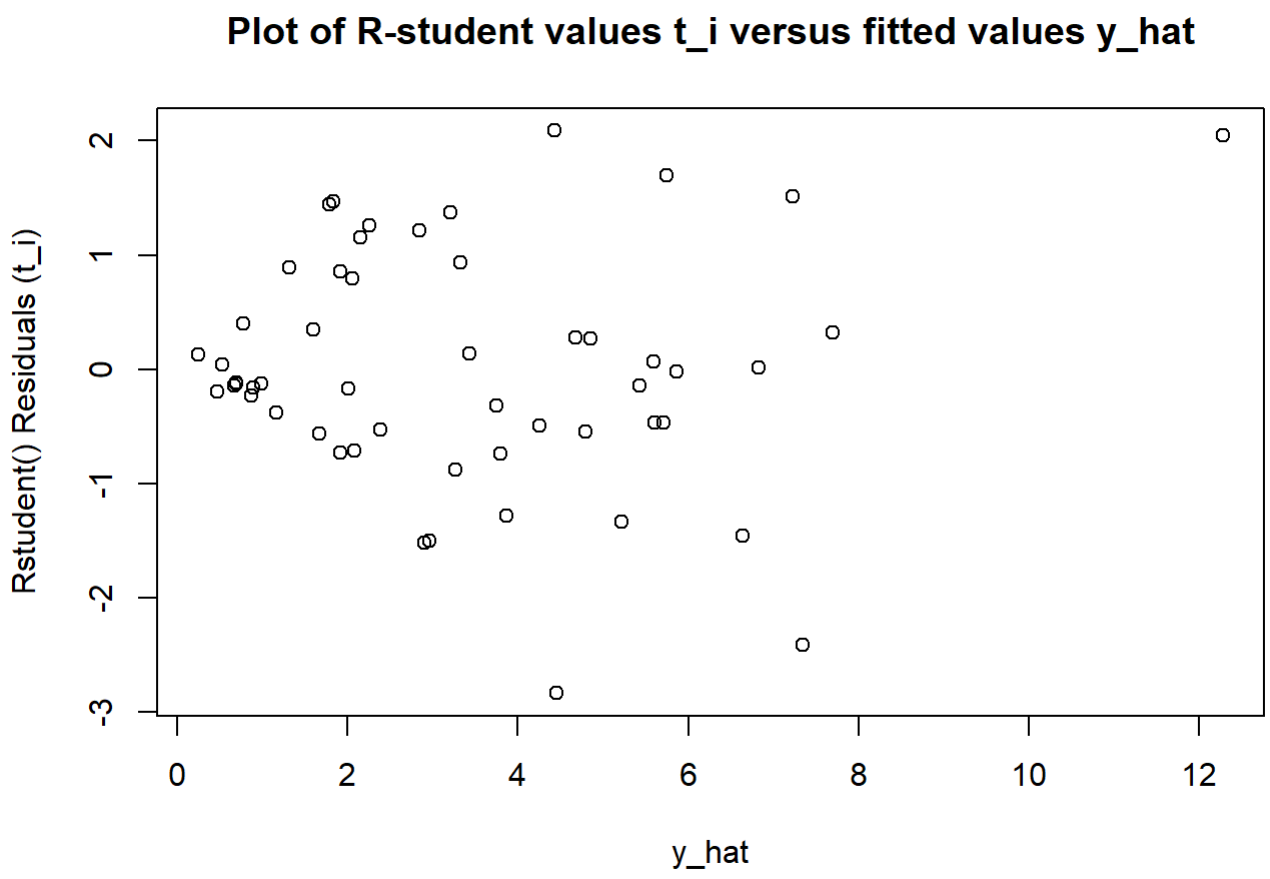
```
# Demand, given Usage
model <- lm(Demand ~ Usage)
beta_0 <- model$coefficients[1]
beta_1 <- model$coefficients[2]

# show beta estimates, std. error, tvalues and p-values
xtable(model)
```

	Estimate <dbl>	Std. Error <dbl>	t value <dbl>	Pr(>|t|) <dbl>
(Intercept)	-0.831303660	0.441612128	-1.882429	6.548865e-02

	Estimate <dbl>	Std. Error <dbl>	t value <dbl>	Pr(>|t|) <dbl>
Usage	0.003682843	0.000333897	11.029877	4.106229e-15

2 rows

```
# Calculate y_hat
y_hat <- beta_0 + beta_1*Usage

# Calculate R-Student residuals
residuals <- rstudent(model)

plot(y_hat, residuals, main='Plot of R-student values t_i versus fitted values y_hat', xlab = 'y
_hat', ylab = 'Rstudent() Residuals (t_i)')
```



Transform and regress on y* = sqrt(y).Compare to least squares fit on p. 173

```
y_star <- sqrt(Demand)

# Demand using sqrt transformation, given Usage
model.star <- lm(y_star ~ Usage)
beta_0 <- model.star$coefficients[1]
beta_1 <- model.star$coefficients[2]

# show beta estimates, std. error, tvalues and p-values
xtable(model)
```

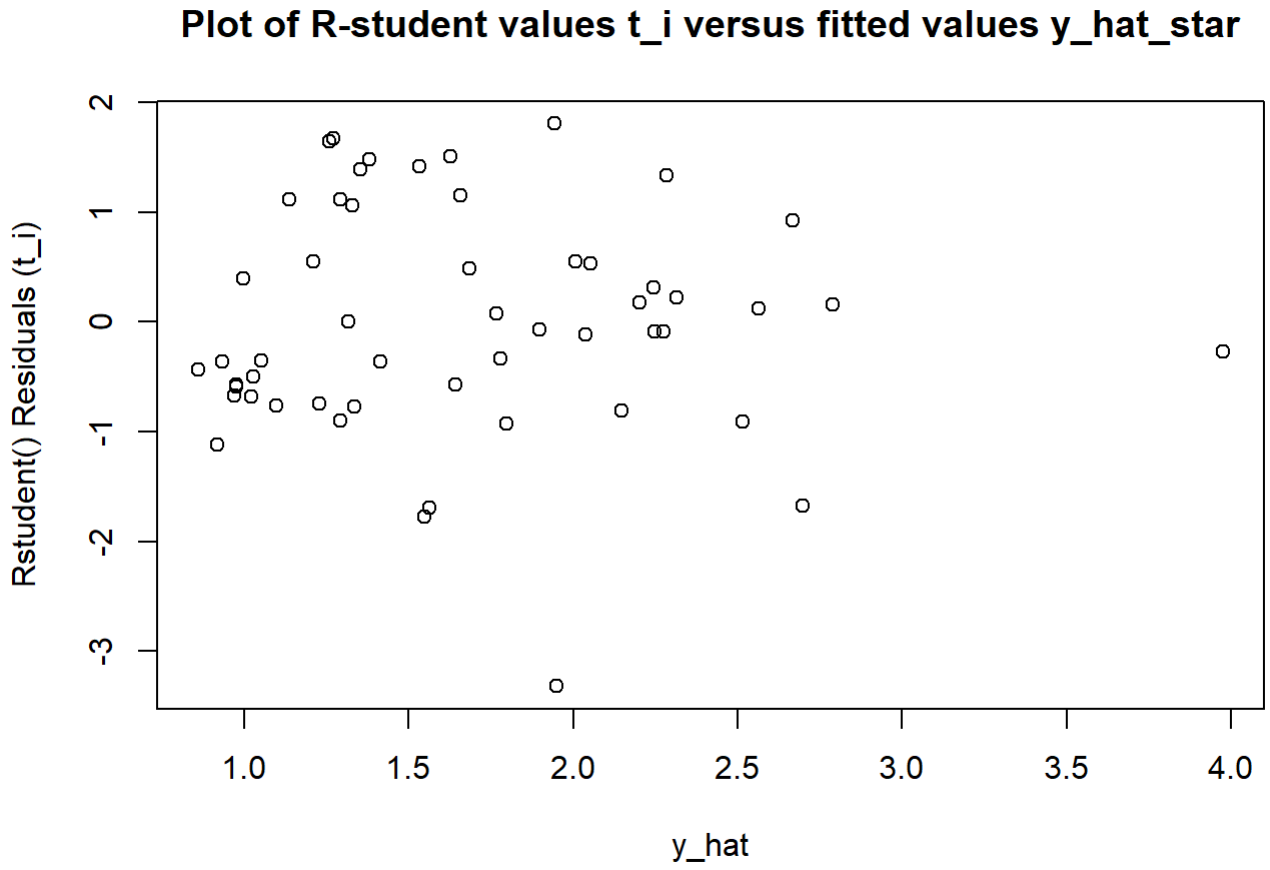
	Estimate <dbl>	Std. Error <dbl>	t value <dbl>	Pr(>|t|) <dbl>
(Intercept)	-0.831303660	0.441612128	-1.882429	6.548865e-02
Usage	0.003682843	0.000333897	11.029877	4.106229e-15

2 rows

```
# Calculate y_hat
y_hat_star <- beta_0 + beta_1*Usage
```

Create plot of R-Student values versus fitted values of transformed model

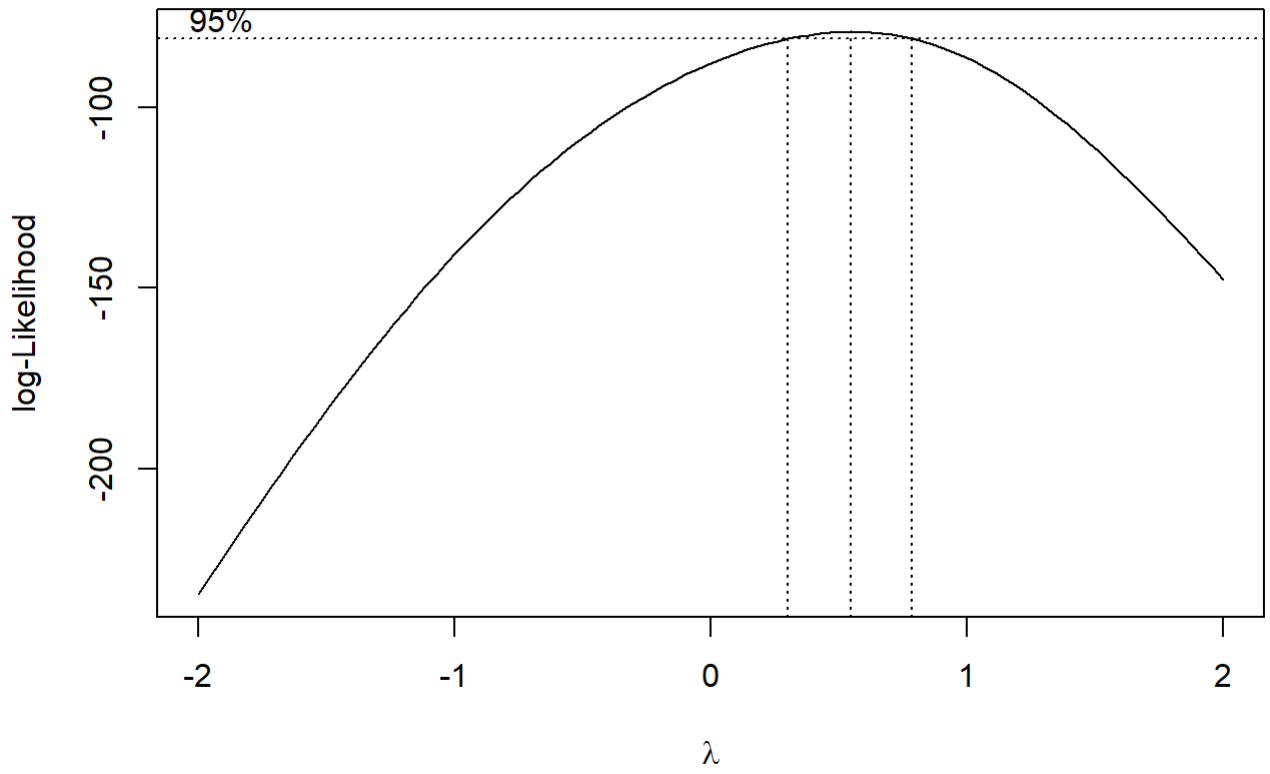
```
# Calculate R-Student residuals
residuals <- rstudent(model.star)

plot(y_hat_star, residuals, main='Plot of R-student values t_i versus fitted values y_hat_star',
xlab = 'y_hat', ylab = 'Rstudent() Residuals (t_i)')
```



Example 5.3 (p. 184)

Perform Box-Cox transformation of Electric Utility data using boxcox() function. boxcox() outputs a range of x and y values. Plot those values and compare to p. 185. Why are the plots different?

```
# create least sqares model
model <- lm(Demand ~ Usage)
library(MASS) # library containg boxcox
bc <- boxcox(Demand ~ Usage)
```



The boxcox() calcultes a log-likelihood for the y-axis versus the book using an approximate Confidence Interval for λ (based on the SS_{Res}).

Find value of lambda ("y" in R's boxcox notation) using which() and max() functions

```
# extract lambda
best.lambda <- bc$x[which(bc$y==max(bc$y))]
```

The best lambda is $\lambda = 0.5454545$

Transform y (Demand) using Box-Cox transformation Equation 5.1 on p. 182. Vary lambda values and compare SS_Residual to Table 5.7 on p. 185

```
# power transformation procedure

# define power transformation function

pow_trans <- function(Lambda_In, ydata) {
# function to perform eqn 5.1 in e-book

n <- length(ydata)
#y_dot <- exp(sum(log(ydata))/n)
y_dot <- geometric.mean(ydata)


  if(Lambda_In == 0){
    y.lambda <- y_dot*log(ydata)

  } else {
    numerator <- ydata^(Lambda_In) - 1
    denominator <- Lambda_In*y_dot^(Lambda_In-1)
    y.lambda <- numerator/denominator

  }
result <- y.lambda
}

lambdaList <- c(-2, -1, -0.5, 0, 0.125, .25, .375, .5, best.lambda, 0.625, .75, 1, 2)
desired_length <- length(lambdaList)
SS_Res_out <- rep(NA, desired_length) # initialize a list

count <- 1
for (val in lambdaList) {

  ydata_trans <- pow_trans(val,Demand)
  model <- lm(ydata_trans ~ Usage)
  SS_Res_out[count] <- anova(model)$'Sum Sq'[2]
  count = count+1
}

List1 <- data.frame(lambdaList)
List2 <- data.frame(SS_Res_out)
out <- as.data.frame(c(List1,List2))
colnames(out) <- c("Lambda","$SS_{Res}$(Lambda)")
tab <- (xtable(out, digits=c(0, 3, 4)))
print(tab, type="html")
```

	Lambda	SS_{Res}(Lambda)
1	-2.000	34100.6081
2	-1.000	986.0341
3	-0.500	291.5816
4	0.000	134.0935
5	0.125	118.1978
6	0.250	107.2054
7	0.375	100.2558

8	0.500	96.9493
9	0.545	96.6384
10	0.625	97.2887
11	0.750	101.6868
12	1.000	126.8660
13	2.000	1275.5606

```
```