R Cheatsheet

Copyright 2014, Bruce Beauchamp

Contents

[R Cheatsheet 1](#_Toc430274189)

[Best Subset Regression 4](#_Toc430274190)

[Control statements 4](#_Toc430274191)

[Iterating through a vector 4](#_Toc430274192)

[data.frames 5](#_Toc430274193)

[Aggregate a column by another column 5](#_Toc430274194)

[Append a column to a data.frame 5](#_Toc430274195)

[Append a row to a data.frame 6](#_Toc430274196)

[Browse a data.frame 6](#_Toc430274197)

[Convert a column to date data type 6](#_Toc430274198)

[Create a data.frame from scratch 6](#_Toc430274199)

[Edit a data.frame in table form 7](#_Toc430274200)

[Expose the fields in a data.frame to the development environment 7](#_Toc430274201)

[Get a cell of data from a data.frame 7](#_Toc430274202)

[Get the column names of a data.frame 7](#_Toc430274203)

[Get the number of rows in a data.frame 7](#_Toc430274204)

[Get a Row of Data from a data.frame 8](#_Toc430274205)

[Import a data.frame column as a formatted date 8](#_Toc430274206)

[Look at the first part of a dataframe 8](#_Toc430274207)

[Look at the structure of a data.frame 8](#_Toc430274208)

[Remove rows containing NA values from a data.frame 8](#_Toc430274209)

[Select Rows which meet criteria 8](#_Toc430274210)

[Select Rows which meet criteria using sqldf 9](#_Toc430274211)

[Select Columns from a data.frame by index 9](#_Toc430274212)

[Subsetting rows selecting on more than one value 9](#_Toc430274213)

[Data Types 10](#_Toc430274214)

[Get the type of an object 10](#_Toc430274215)

[Dates and Times 10](#_Toc430274216)

[Convert a String to a Date 10](#_Toc430274217)

[Environment 10](#_Toc430274218)

[Change the Working Directory 10](#_Toc430274219)

[Get the Working Directory 10](#_Toc430274220)

[List the Files in the Working Directory 10](#_Toc430274221)

[Load a Function into the Environment 10](#_Toc430274222)

[List the Functions Available in the Environment 10](#_Toc430274223)

[Files and Directories 11](#_Toc430274224)

[Create a directory 11](#_Toc430274225)

[Download a file from the web 11](#_Toc430274226)

[See if a directory or file exists 11](#_Toc430274227)

[Libraries 11](#_Toc430274228)

[Loading Libraries 11](#_Toc430274229)

[Linear Regression 12](#_Toc430274230)

[Confidence interval for a regression line fit 12](#_Toc430274231)

[Generate the Regression line 12](#_Toc430274232)

[Get more information about the linear regression 12](#_Toc430274233)

[Plot a regression line on a plot 12](#_Toc430274234)

[Matrices 13](#_Toc430274235)

[Create a matrix 13](#_Toc430274236)

[Misc 13](#_Toc430274237)

[Printing a variable to the console 13](#_Toc430274238)

[Packages 13](#_Toc430274239)

[Installing packages 13](#_Toc430274240)

[Plotting data 14](#_Toc430274241)

[Adding a legend to a base plot 14](#_Toc430274242)

[faceted plots (ggplot2) 14](#_Toc430274243)

[Line Plots 15](#_Toc430274244)

[Histograms 15](#_Toc430274245)

[Line plots with multiple lines 15](#_Toc430274246)

[Multiple plots in quadrants 17](#_Toc430274247)

[Plotting to a PNG file 17](#_Toc430274248)

[Scatter plot of y vs x 18](#_Toc430274249)

[Using Base Plot 18](#_Toc430274250)

[Reading Data 19](#_Toc430274251)

[Read a CSV file into a data.frame 19](#_Toc430274252)

[Read from an excel spreadsheet 19](#_Toc430274253)

[Strings 19](#_Toc430274254)

[Concatenate two strings 19](#_Toc430274255)

[Padding a string with zeros 19](#_Toc430274256)

[Vectors 20](#_Toc430274257)

[Concatenate two string vectors 20](#_Toc430274258)

[Creating a vector 20](#_Toc430274259)

[Iterating over a vector 20](#_Toc430274260)

[Length of a vector 20](#_Toc430274261)

[Return valid elements of a vector 20](#_Toc430274262)

# Best Subset Regression

Use the leaps() package.

library(leaps)

regfit.full = regsubsets(Salary~., data=Hitters)

# Control statements

## for-next loops

for (i in 1:50 ) {

foo.squared[i] = foo[i]^2

}

## Iterating through a vector

id = 1:3

for (i in id) {

cat(i, "\n")

}

|  |
| --- |
| 1  2  3 |
|  |
| |  | | --- | |  | |

# data.frames

## Aggregate a column by another column

> head(NEI)

fips SCC Pollutant Emissions type year

4 09001 10100401 PM25-PRI 15.714 POINT 1999

8 09001 10100404 PM25-PRI 234.178 POINT 1999

12 09001 10100501 PM25-PRI 0.128 POINT 1999

16 09001 10200401 PM25-PRI 2.036 POINT 1999

20 09001 10200504 PM25-PRI 0.388 POINT 1999

24 09001 10200602 PM25-PRI 1.490 POINT 1999

> test <- aggregate(Emissions~year, NEI, sum)

> class(test)

[1] "data.frame"

> head(test)

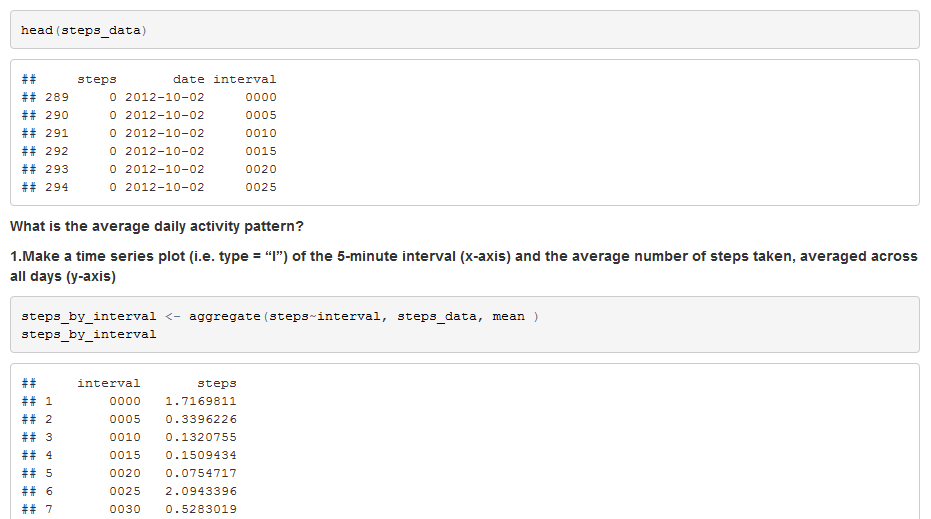
year Emissions

1 1999 7332967

2 2002 5635780

3 2005 5454703

4 2008 3464206



### Aggregate multiple columns

fatalities\_by\_event\_type = aggregate(cbind(FATALITIES, INJURIES)~EVTYPE, storm\_data, sum)

head(fatalities\_by\_event\_type)

## EVTYPE FATALITIES INJURIES

## 1 HAIL 4 225

## 2 TORNADO 2609 44405

## 3 TSTM WIND 148 2014

## Append a column to a data.frame

Note: The column data is inside a vector

|  |
| --- |
| > str(energy\_use\_subset)  'data.frame': 2880 obs. of 9 variables:  $ Date : chr "1/2/2007" "1/2/2007" "1/2/2007" "1/2/2007" ...  $ Time : chr "00:00:00" "00:01:00" "00:02:00" "00:03:00" ...  $ Global\_active\_power : num 0.326 0.326 0.324 0.324 0.322 0.32 0.32 0.32 0.32 0.236 ...  $ Global\_reactive\_power: num 0.128 0.13 0.132 0.134 0.13 0.126 0.126 0.126 0.128 0 ...  $ Voltage : num 243 243 244 244 243 ...  $ Global\_intensity : num 1.4 1.4 1.4 1.4 1.4 1.4 1.4 1.4 1.4 1 ...  $ Sub\_metering\_1 : num 0 0 0 0 0 0 0 0 0 0 ...  $ Sub\_metering\_2 : num 0 0 0 0 0 0 0 0 0 0 ...  $ Sub\_metering\_3 : num 0 0 0 0 0 0 0 0 0 0 ...  > str(dt)  POSIXlt[1:2880], format: "2007-02-01 00:00:00" "2007-02-01 00:01:00" "2007-02-01 00:02:00" ...  > energy\_use\_subset$date\_time <- dt  > str(energy\_use\_subset)  'data.frame': 2880 obs. of 10 variables:  $ Date : chr "1/2/2007" "1/2/2007" "1/2/2007" "1/2/2007" ...  $ Time : chr "00:00:00" "00:01:00" "00:02:00" "00:03:00" ...  $ Global\_active\_power : num 0.326 0.326 0.324 0.324 0.322 0.32 0.32 0.32 0.32 0.236 ...  $ Global\_reactive\_power: num 0.128 0.13 0.132 0.134 0.13 0.126 0.126 0.126 0.128 0 ...  $ Voltage : num 243 243 244 244 243 ...  $ Global\_intensity : num 1.4 1.4 1.4 1.4 1.4 1.4 1.4 1.4 1.4 1 ...  $ Sub\_metering\_1 : num 0 0 0 0 0 0 0 0 0 0 ...  $ Sub\_metering\_2 : num 0 0 0 0 0 0 0 0 0 0 ...  $ Sub\_metering\_3 : num 0 0 0 0 0 0 0 0 0 0 ...  $ date\_time : POSIXlt, format: "2007-02-01 00:00:00" "2007-02-01 00:01:00". |
|  |
|  |

## Append a row to a data.frame

> df

id nobs

1 1 2

> df <- rbind(df, data.frame(id=17, nobs=34))

> df

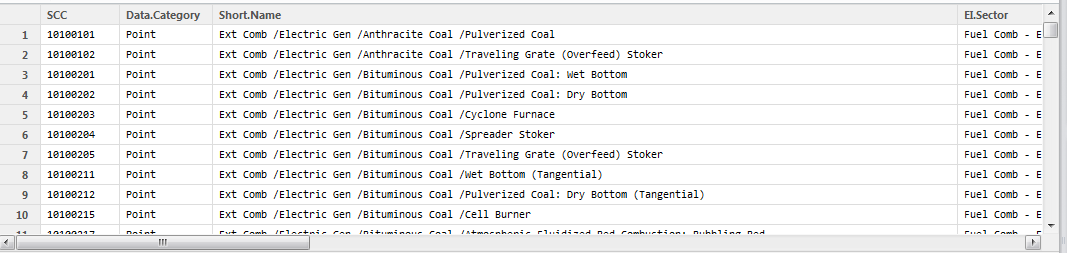
id nobs

1 1 2

2 17 34

## Browse a data.frame

View(SCCT)



## Convert a column to date data type

> energy\_use$Date <- as.Date(energy\_use$Date, format ="%d/%m/%Y")

> str(energy\_use)

'data.frame': 5258 obs. of 9 variables:

$ Date : Date, format: "2007-02-01" "2007-02-01" "2007-02-01" ...

## Create a data.frame from scratch

> df <- data.frame(id=integer(), nobs=integer())

> df <- rbind(df, data.frame(id=1, nobs=2))

> df

id nobs

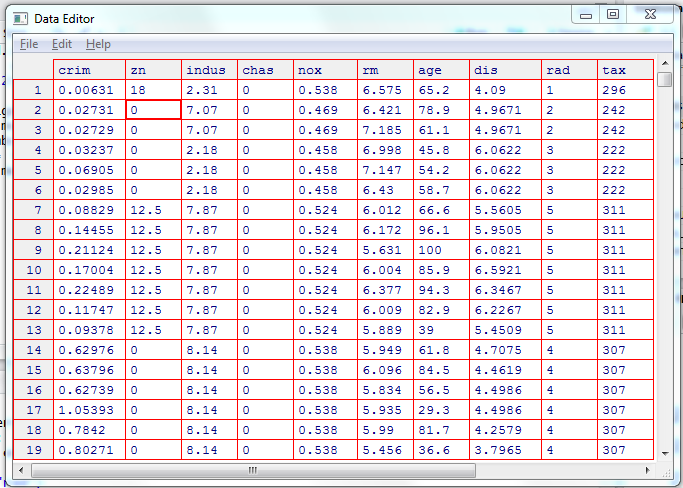
1 1 2

## Create a factor column based upon values in an existing column



## Edit a data.frame in table form

> fix(Boston)



## Expose the fields in a data.frame to the development environment

attach(Boston)

## Get a cell of data from a data.frame

> pollutantData[1,]

Date sulfate nitrate ID

1 2003-01-01 NA NA 1

> pollutantData[1,"ID"]

[1] 1

## Get the column names of a data.frame

> colnames(outcomeData)

## Get the number of rows in a data.frame

|  |
| --- |
| > nrow(data1)  [1] 153 |
|  |
| |  | | --- | |  | |

## Get a Row of Data from a data.frame

|  |
| --- |
| data1[1,]  Ozone Solar.R Wind Temp Month Day  1 41 190 7.4 67 5 1 |
|  |
| |  | | --- | |  | |

## Import a data.frame column as a formatted date

setClass('myDate')

setAs("character","myDate", function(from) as.Date(from, format="%d/%m/%Y") )

energy\_use <- read.csv("data/household\_power\_consumption\_short.txt",

sep=";", colClasses= c("myDate", "character", "numeric",

"numeric","numeric","numeric",

"numeric","numeric","numeric"))

## Look at the first part of a dataframe

head(df[whichrows, whichcols])

## Look at the structure of a data.frame

> str(energy\_use)

'data.frame': 5258 obs. of 9 variables:

$ Date : Factor w/ 4 levels "1/2/2007","2/2/2007",..: 1 1 1 1 1 1 1 1 1 1 ...

$ Time : Factor w/ 1440 levels "00:00:00","00:01:00",..: 1 2 3 4 5 6 7 8 9 10 ...

$ Global\_active\_power : num 0.326 0.326 0.324 0.324 0.322 0.32 0.32 0.32 0.32 0.236 ...

$ Global\_reactive\_power: num 0.128 0.13 0.132 0.134 0.13 0.126 0.126 0.126 0.128 0 ...

$ Voltage : num 243 243 244 244 243 ...

$ Global\_intensity : num 1.4 1.4 1.4 1.4 1.4 1.4 1.4 1.4 1.4 1 ...

$ Sub\_metering\_1 : num 0 0 0 0 0 0 0 0 0 0 ...

$ Sub\_metering\_2 : num 0 0 0 0 0 0 0 0 0 0 ...

$ Sub\_metering\_3 : num 0 0 0 0 0 0 0 0 0 0 ...

## Remove rows containing NA values from a data.frame

steps\_data <- na.omit(steps\_data)

## Select Rows which meet criteria

> test <- iris[iris$Species=="virginica", ]

> test

Sepal.Length Sepal.Width Petal.Length Petal.Width Species

101 6.3 3.3 6.0 2.5 virginica

102 5.8 2.7 5.1 1.9 virginica

103 7.1 3.0 5.9 2.1 virginica

Data2<-Data[which(Data$Date %in% c("01/02/2007","02/02/2007")),]

## Select Rows which meet criteria using sqldf

setwd("D:/One Drive/Coursera/Exploratory Data Analysis/Assignments/Week 1/Data/exdata-data-household\_power\_consumption")

install.packages("sqldf")

library(sqldf)

## Read subset of data using read.csv.sql

data<-read.csv.sql("household\_power\_consumption.txt",header=TRUE,sep=";",

sql="Select \* from file where Date='1/2/2007' OR Date='2/2/2007'")

## Select Columns from a data.frame by index

outcomeData <- outcomeData[, c(2, 7, column)]

Here, columns 2, 7, and one additional column are retained

## Subsetting rows selecting on more than one value

We use the %in% notation when we want to subset on multiple values of y. The x.sub5 data frame contains only the observations for which the values of variable y are equal to either 1 or 4.

x.sub5 <- x.df[x.df$y %in% c(1, 4), ]

x.sub5

V1 V2 V3 V4 V5 y

1 -1.6862356 1.395021 1.35898920 1.8492410 1.7536886 1

2 0.8610318 -0.569828 -0.01984841 0.3570547 -0.9326248 1

5 0.6296957 1.794336 2.16226397 0.1604166 0.3721850 4

# Data Types

## Get the type of an object

|  |
| --- |
| > x <- 4  > class(x)  [1] "numeric" |
|  |
|  |

# Dates and Times

## Convert a String to a Date

> strptime("1/2/2007 00:01:00", "%d/%m/%Y %H:%M:%S")

[1] "2007-02-01 00:01:00 PST"

> class(strptime("1/2/2007 00:01:00", "%d/%m/%Y %H:%M:%S"))

[1] "POSIXlt" "POSIXt"

# Environment

## Change the Working Directory

> setwd("c:/r/Prog3")

> getwd()

[1] "c:/r/Prog3"

## Get the Working Directory

> getwd()

[1] "C:/Users/Bruce/Documents"

>

## List the Files in the Working Directory

dir()

## Load a Function into the Environment

source(“myCode.R”)

## List the Functions Available in the Environment

ls()

# Files and Directories

## Create a directory

> dir.create("data")

## Download a file from the web

download.file()

## See if a directory or file exists

> file.exists("no such dir")

[1] FALSE

# Libraries

## Loading Libraries

> library(ISLR)

# Linear Regression

## Confidence interval for a regression line fit

> confint(fit)

2.5 % 97.5 %

(Intercept) 33.448457 35.6592247

lstat -1.026148 -0.8739505

## Generate the Regression line

> fit = lm(medv~lstat, Boston)

> fit

Coefficients:

(Intercept) lstat

34.55 -0.95

Note generates the regression line for mdev as a function of lstat

## Get more information about the linear regression

> summary(fit)

Residuals:

Min 1Q Median 3Q Max

-15.168 -3.990 -1.318 2.034 24.500

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 34.55384 0.56263 61.41 <2e-16 \*\*\*

lstat -0.95005 0.03873 -24.53 <2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

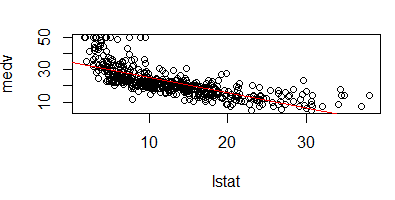
Residual standard error: 6.216 on 504 degrees of freedom

Multiple R-squared: 0.5441, Adjusted R-squared: 0.5432

F-statistic: 601.6 on 1 and 504 DF, p-value: < 2.2e-16

## Plot a regression line on a plot

> abline(fit, col="red")



# Matrices

## Create a matrix

x <- matrix(c(1,2,3,4), 2,2)

> x

[,1] [,2]

[1,] 1 3

[2,] 2 4

# Misc

## Printing a variable to the console

for (i in id) {

**cat(i, "\n")**

}

# Packages

## Installing packages

> install.packages("ISLR")

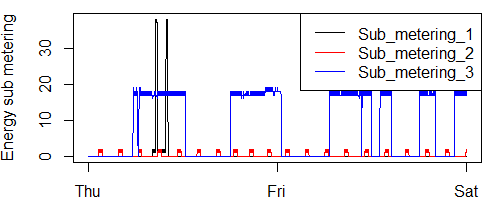
# Plotting data

## Adding a legend to a base plot

legend(x='topright',

legend=c("Sub\_metering\_1 ", "Sub\_metering\_2 ", "Sub\_metering\_3 "),

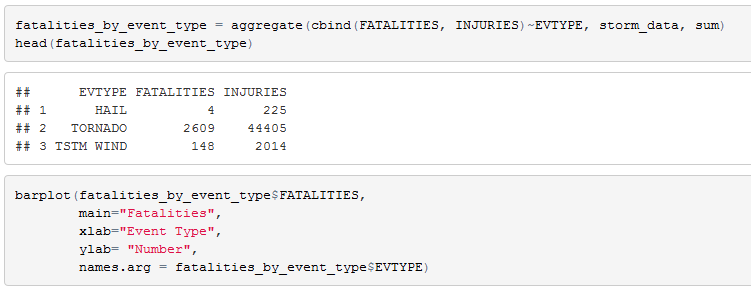
col=c('black', 'red', 'blue'), lty=c(1,1,1), cex=0.5 )

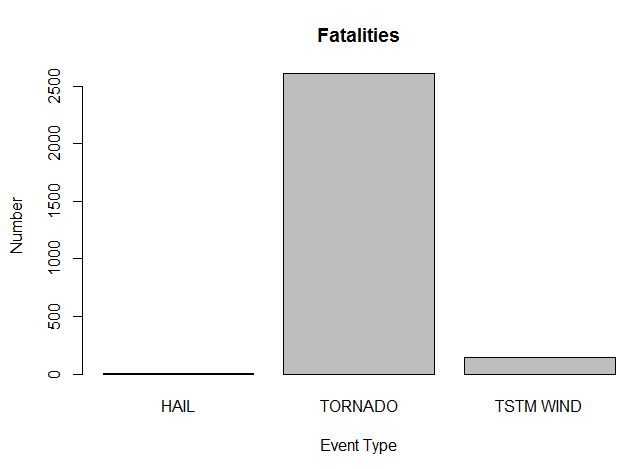


lty = line type. 0=blank, 1=solid

cex : Sets relative size of the legend

## Bar Plots





## faceted plots (ggplot2)

str(balt)

'data.frame': 2096 obs. of 6 variables:

$ fips : chr "24510" "24510" "24510" "24510" ...

$ SCC : chr "10100601" "10200601" "10200602" "30100699" ...

$ Pollutant: chr "PM25-PRI" "PM25-PRI" "PM25-PRI" "PM25-PRI" ...

$ Emissions: num 6.53 78.88 0.92 10.38 10.86 ...

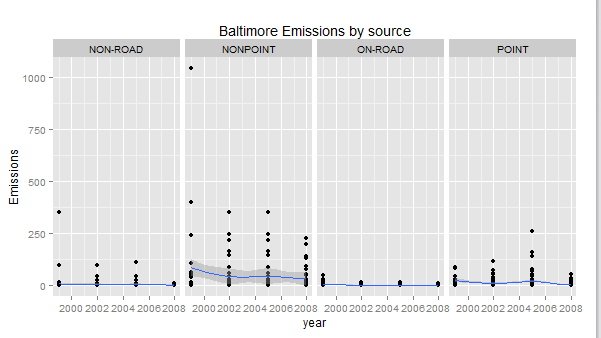
$ type : chr "POINT" "POINT" "POINT" "POINT" ...

$ year : int 1999 1999 1999 1999 1999 1999 1999 1999 1999 1999 ...

qplot(year, Emissions, data=balt, facets=.~type,

geom=c("point", "smooth"), method="loess") +

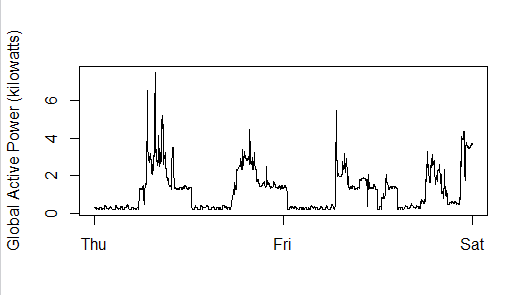
labs(title="Baltimore Emissions by source")



## Line Plots

plot(energy\_use\_subset$date\_time, energy\_use\_subset$Global\_active\_power, type="l",

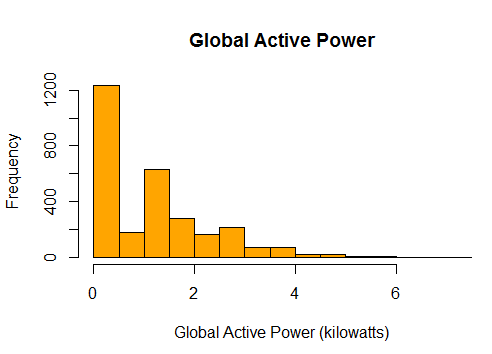
xlab='', ylab='Global Active Power (kilowatts)')



## Histograms

hist(energy\_use\_subset$Global\_active\_power, main='Global Active Power', col='orange',

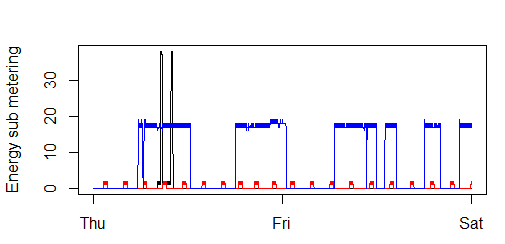
xlab='Global Active Power (kilowatts)')



## Line plots with multiple lines

plot(x,y1,type="l",col="red")

lines(x,y2,col="green")



## Multiple plots in a grid

require(gridExtra)

plot1 <- qplot(1)

plot2 <- qplot(1)

grid.arrange(plot1, plot2, ncol=2)

## Multiple plots in quadrants

par(mfrow=c(2,2), mar=c(4,4,2,1), oma=c(0,0,2,0))

with(energy\_use\_subset, {

plot(energy\_use\_subset$date\_time, energy\_use\_subset$Global\_active\_power, type="l",

xlab='', ylab='Global Active Power (kilowatts)')

plot(energy\_use\_subset$date\_time, energy\_use\_subset$Voltage, type="l",

xlab='datetime', ylab='Voltage')

plot(energy\_use\_subset$date\_time, energy\_use\_subset$Sub\_metering\_1, type="l",

xlab='', ylab='Energy sub metering')

lines(energy\_use\_subset$date\_time, energy\_use\_subset$Sub\_metering\_2, col='red')

lines(energy\_use\_subset$date\_time, energy\_use\_subset$Sub\_metering\_3, col='blue')

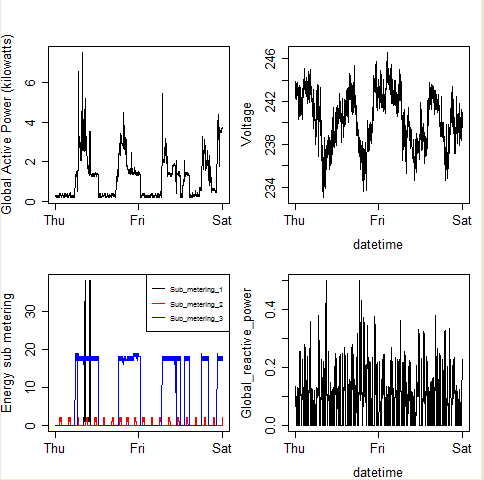
legend(x='topright', legend=c("Sub\_metering\_1 ", "Sub\_metering\_2 ", "Sub\_metering\_3 "),

col=c('black', 'red', 'blue'), lty=c(1,1,1), cex=0.5 )

plot(energy\_use\_subset$date\_time, energy\_use\_subset$Global\_reactive\_power, type="l",

xlab='datetime', ylab='Global\_reactive\_power')

})



## Plotting to a PNG file

hist(energy\_use\_subset$Global\_active\_power, main='Global Active Power', col='orange',

xlab='Global Active Power (kilowatts)')

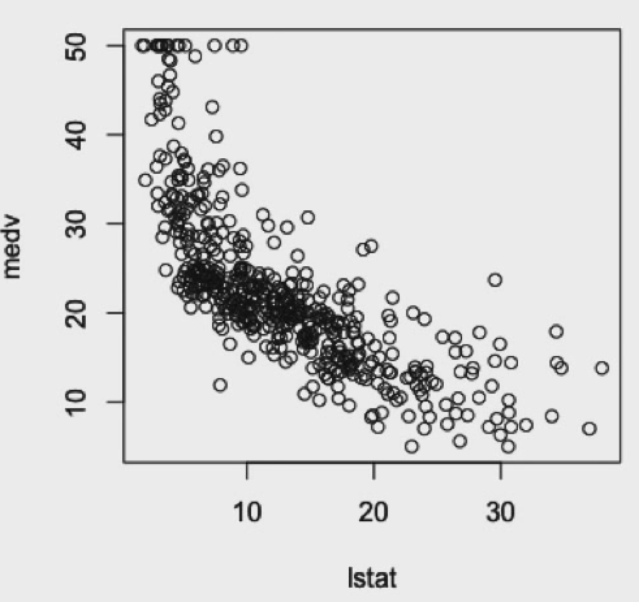
# Now copy the histogram to a png file

dev.copy(png, file='plot1.png', width = 480, height = 480 )

dev.off()

## Scatter plot of y vs x

plot(medv~lstat, Boston)



## Using Base Plot

NEI <- readRDS("summarySCC\_PM25.rds")

balt <- NEI[NEI$fips=="24510",]

yearlyEmissions <- aggregate(Emissions~year, balt, sum)

with(yearlyEmissions, plot(year, Emissions))

title(main= "Yearly Total PM2.5 Emissions for Baltimore")

# Reading Data

## Read a CSV file into a data.frame

> data1 <- read.csv("hw1\_data.csv")

> data1

Ozone Solar.R Wind Temp Month Day

1 41 190 7.4 67 5 1

2 36 118 8.0 72 5 2

3 12 149 12.6 74 5 3

## Read from an excel spreadsheet

library(xlsx)

read.xlsx()

# Strings

## Concatenate two strings

|  |
| --- |
| > paste("a","b", sep = "")  [1] "ab" |
|  |
| |  | | --- | | > |   > paste0("1","2")  [1] "12" |

## Padding a string with zeros

> sprintf("%03d", 17)

[1] "017"

steps\_data$interval = str\_pad(steps\_data$interval, 4, pad="0")

steps\_data

## steps date interval

## 289 0 2012-10-02 0000

## 290 0 2012-10-02 0005

## 291 0 2012-10-02 0010

## 292 0 2012-10-02 0015

# Vectors

## Concatenate two string vectors

> date\_part <- energy\_use\_subset[, 'Date']

> time\_part <- energy\_use\_subset[, 'Time']

> date\_time\_str <- paste(date\_part, time\_part, sep=' ')

> str(date\_time\_str)

chr [1:2880] "1/2/2007 00:00:00" "1/2/2007 00:01:00" "1/2/2007 00:02:00" ...

## Creating a vector

> a = 1:2

> a

[1] 1 2

|  |
| --- |
| > x <- c(1,3,2,5)  > x  [1] 1 3 2 5 |
|  |
| |  | | --- | |  | |

## Iterating over a vector

for (i in id) {

writeLines(paste(i))

}

|  |
| --- |
| 1  2 |
|  |
| |  | | --- | |  | |

## Length of a vector

> bad <- ozone\_col[is.na(ozone\_col)]

> length(bad)

[1] 37

## Return valid elements of a vector

|  |
| --- |
| > a  [1] 1 2 NA 4  > a[complete.cases(a)]  [1] 1 2 4 |
|  |
| |  | | --- | |  | |