Working with air quality and meteorological data exercises part 1: Solutions

Below are the solutions to these openair exercises

```
############### # # # Exercise 1 # # # ############### my1data <- importAURN(site = 'MY1', year = 2016, met = TRUE) ############### # # # Exercise 2 # # # ###############
summary(my1data)

############### # # # Exercise 3 # # # ###############
monthly_pm10 <- aggregate(my1data["pm10"], format(my1data["date"],"%Y-%m"), mean, na.rm = TRUE)
monthly_pm2.5 <- aggregate(my1data["pm2.5"], format(my1data["date"],"%Y-%m"), mean, na.rm = TRUE)
monthly_nox <- aggregate(my1data["nox"], format(my1data["date"],"%Y-%m"), mean, na.rm = TRUE)
monthly_no <- aggregate(my1data["no"], format(my1data["date"],"%Y-%m"), mean, na.rm = TRUE)
monthly_o3 <- aggregate(my1data["o3"], format(my1data["date"],"%Y-%m"), mean, na.rm = TRUE)
############### # # # Exercise 4 # # # ###############
daily_pm10 <- aggregate(my1data["pm10"], format(my1data["date"],"%Y-%j"), mean, na.rm = TRUE)
daily_pm2.5 <- aggregate(my1data["pm2.5"], format(my1data["date"],"%Y-%j"), mean, na.rm = TRUE) daily_nox <- aggregate(my1data["nox"], format(my1data["date"],"%Y-%j"), mean, na.rm = TRUE) daily_no <- aggregate(my1data["no"], format(my1data["date"],"%Y-%j"), mean, na.rm = TRUE) daily_o3 <- aggregate(my1data["o3"], format(my1data["date"],"%Y-%j"), mean, na.rm = TRUE) ############### # # # Exercise 5 # # # ###############

daily_max_nox <- aggregate(my1data["nox"], format(my1data["date"],"%Y-%j"), max, na.rm = TRUE)
daily_max_no <- aggregate(my1data["no"], format(my1data["date"],"%Y-%j"), max, na.rm = TRUE)
```

Sending Emails from R Solutions

Below are the solutions to these exercises on Sending Emails from R.

```
#####################  #  #  #  Exercise  1  #  #  #
#################### library(mailR) # Note that I am using environment variables. send.mail(from = Sys.getenv("sender") , to = Sys.getenv("recipient") , body = "Hello World. This is my email!" , html = TRUE , inline = FALSE , smtp = list(host.name = Sys.getenv("smtp_server"), port = 25) , authenticate = FALSE , send = TRUE)
```

```
##                          [1]                          "Java-Object{org.apache.commons.mail.HtmlEmail@5387f9e0}"
```

```
#####################  #  #  #  Exercise  2  #  #  #
#################### send.mail(from = Sys.getenv("sender") , to = Sys.getenv(c("recipient", "recipient2")) , body = "Hello World. This is my email!" , html = TRUE , inline = FALSE , smtp = list(host.name = Sys.getenv("smtp_server"), port = 25) , authenticate = FALSE , send = TRUE)
```

```
##                          [1]                          "Java-Object{org.apache.commons.mail.HtmlEmail@5f4da5c3}"
```

```
#####################  #  #  #  Exercise  3  #  #  #
#################### send.mail(from = Sys.getenv("sender") , to = Sys.getenv("recipient") , subject = "Email Testing" , body = "Hello World. This is my email!" , html = TRUE , inline = FALSE , smtp = list(host.name = Sys.getenv("smtp_server"), port = 25) , authenticate = FALSE , send = TRUE)
```

```
##                          [1]                          "Java-
Object{org.apache.commons.mail.HtmlEmail@6659c656}"

#####################  #  #  #  Exercise  4  #  #  #
#################### send.mail(from = Sys.getenv("sender") ,
to = Sys.getenv("recipient") , subject = "Email Testing" ,
body = "<span style='color:red;'>Hello World.</span> <span
style='color:blue;'>This is my email!</span>" , html = TRUE ,
inline  =  FALSE  ,  smtp  =  list(host.name  =
Sys.getenv("smtp_server"), port = 25) , authenticate = FALSE ,
send = TRUE)

##                          [1]                          "Java-
Object{org.apache.commons.mail.HtmlEmail@7591083d}"

#####################  #  #  #  Exercise  5  #  #  #
#################### send.mail(from = Sys.getenv("sender") ,
to = Sys.getenv("recipient") , subject = "Email Testing" ,
body = "my_email.html" , html = TRUE , inline = FALSE , smtp =
list(host.name = Sys.getenv("smtp_server"), port = 25) ,
authenticate = FALSE , send = TRUE)

##                          [1]                          "Java-
Object{org.apache.commons.mail.HtmlEmail@5fa7e7ff}"

#####################  #  #  #  Exercise  6  #  #  #
#################### send.mail(from = Sys.getenv("sender") ,
to = Sys.getenv("recipient") , subject = "Email Testing" ,
body = "default_knitr.html" , html = TRUE , inline = FALSE ,
smtp = list(host.name = Sys.getenv("smtp_server"), port = 25)
, authenticate = FALSE , send = TRUE)

##                          [1]                          "Java-
Object{org.apache.commons.mail.HtmlEmail@619a5dff}"

#####################  #  #  #  Exercise  7  #  #  #
#################### send.mail(from = Sys.getenv("sender") ,
to = Sys.getenv("recipient") , subject = "Email Testing" ,
body = "Hello World. This is my email!" , html = TRUE , inline
= FALSE , smtp = list(host.name = Sys.getenv("smtp_server"),
```

```
port = 25) , authenticate = FALSE , attach.files =
"mailr_six.r" , send = TRUE)

##                          [1]                          "Java-
Object{org.apache.commons.mail.HtmlEmail@4f2410ac}"

#####################  #  #  #  Exercise  8  #  #  #
#################### send.mail(from = Sys.getenv("sender") ,
to = Sys.getenv("recipient") , subject = "Email Testing" ,
body = "Hello World. This is my email!" , html = TRUE , inline
= FALSE , smtp = list(host.name = Sys.getenv("smtp_server"),
port = 25) , authenticate = FALSE , attach.files =
"mailr_six.r" , file.description = "Solutions to the first six
exercises." , file.names = "First Six Solutions" , send =
TRUE)

##                          [1]                          "Java-
Object{org.apache.commons.mail.HtmlEmail@2957fcb0}"

#####################  #  #  #  Exercise  9  #  #  #
#################### send.mail(from = Sys.getenv("sender") ,
to = Sys.getenv("recipient") , subject = "Email Testing" ,
body = "Hello World. This is my email!" , html = TRUE , inline
= FALSE , smtp = list(host.name = Sys.getenv("smtp_server"),
port = 25) , authenticate = FALSE , attach.files =
c("mailr_six.r", "mailr_eight.r") , file.description =
c("Solutions to the first six exercises.", "Solutions to the
first eight exercises.") , file.names = c("First Six
Solutions", "First Eight Solutions") , send = TRUE)

##                          [1]                          "Java-
Object{org.apache.commons.mail.HtmlEmail@593634ad}"

#####################  #  #  #  Exercise  10  #  #  #
#################### important_number <- sample(1:10, 1)
if(important_number  %%  2  ==  ){  send.mail(from  =
Sys.getenv("sender") , to = Sys.getenv("recipient") , subject
= "Even Alert!" , body = "The important number is even!" ,
html = TRUE , inline = FALSE , smtp = list(host.name =
Sys.getenv("smtp_server"), port = 25) , authenticate = FALSE ,
```

```
send = TRUE) } else { print("No need to send an email!") }

## [1] "No need to send an email!"
```

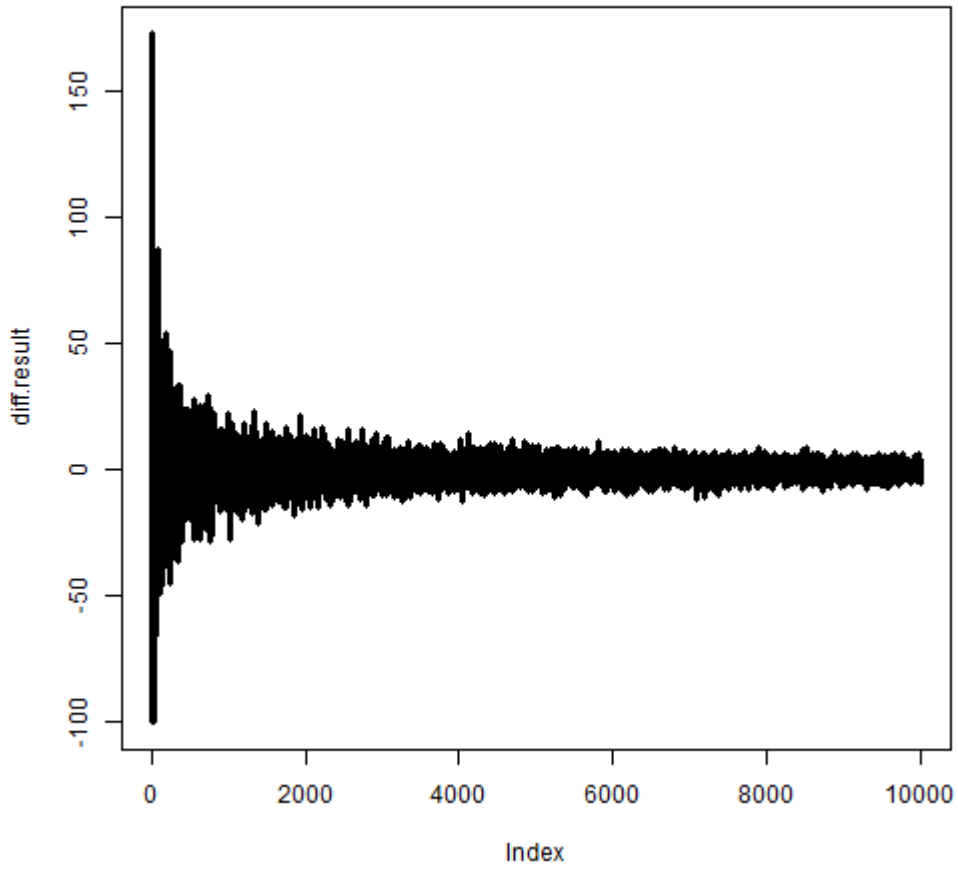
Hacking statistics or: How I Learned to Stop Worrying About Calculus and Love Stats Solutions (Part-6)

Below are the solutions to these exercises on statistical tests.

```
######################  #  #  #  Exercise  1  #  #  #
#####################  sample.norm<-function(size)  {  data<-
rnorm(size,mean=5+8/12,sd=2.94/12)
return(sum(data>=5+8/12&data<=5+9/12)/size)  }  set.seed(42)
sample.norm(200)

## [1] 0.135

######################  #  #  #  Exercise  2  #  #  #
######################                                prob<-
pnorm(5+9/12,mean=5+8/12,sd=2.94/12)-
pnorm(5+8/12,mean=5+8/12,sd=2.94/12) print(prob)

## [1] 0.133123

sample.diff<-function(size1,size2) { results<-NULL for(i in
size1:size2)  {  results<-c(  results,((sample.norm(i)-
prob)/prob)*100)  }  return(results)  }  set.seed(42)
diff.result<-sample.diff(1,10000) str(diff.result)
```

```
## num [1:10000] -100 -100 -100 -100 -100 ...
```

```
plot(diff.result,type='l',lwd=3)
```




```
#####################  #  #  #  Exercise  3  #  #  #
#################### #1 pnorm(,mean=,sd=1)
```

```
## [1] 0.5
```

```
x.norm <- seq(-4, 4, 0.01) plot(x.norm,dnorm(x.norm, mean=,
sd=1)) abline(v=,col="red") #2 1-pnorm(,mean=,sd=1)
```

```
## [1] 0.5
```

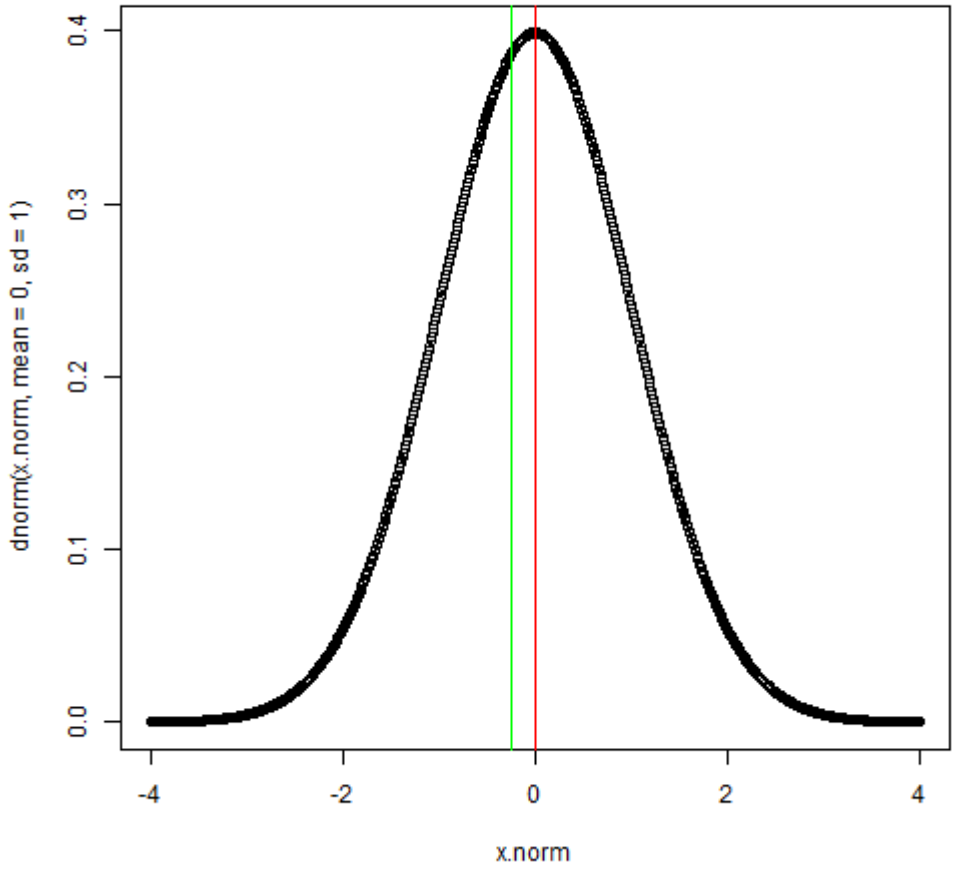
```
#or pnorm(,mean=,sd=1,lower.tail = FALSE)
```

```
## [1] 0.5
```

```
#3 pnorm(-0.25,mean=,sd=1)
```

```
## [1] 0.4012937
```

```
abline(v=-0.25,col="green")
```




```
#4 pnorm(,mean=,sd=1)-pnorm(-0.25,mean=,sd=1)
```

```
## [1] 0.09870633
```

```
#####################  #  #  #  Exercise  4  #  #  #
#####################  qnorm(0.99,mean=,sd=1)
```

```
## [1] 2.326348
```

```
qnorm(0.975,mean=,sd=1)
```

```
## [1] 1.959964
```

```
qnorm(0.95,mean=,sd=1)
```

```
## [1] 1.644854
```

```
qnorm(0.01,mean=,sd=1)
```

```
## [1] -2.326348
```

```
qnorm(0.025,mean=,sd=1)
```

```
## [1] -1.959964
```

```
qnorm(0.05,mean=,sd=1)
```

```
## [1] -1.644854
```

```
#####################  #  #  #  Exercise  5  #  #  #
#####################    #1    data.ex.5<-read.csv(
"http://www.r-exercises.com/wp-content/uploads/2017/08/data.ex
_.5.csv") str(data.ex.5)
```

```
## 'data.frame': 150 obs. of 1 variable: ## $ x: num 13.889
25.002 2.682 20.755 0.123 ...
```

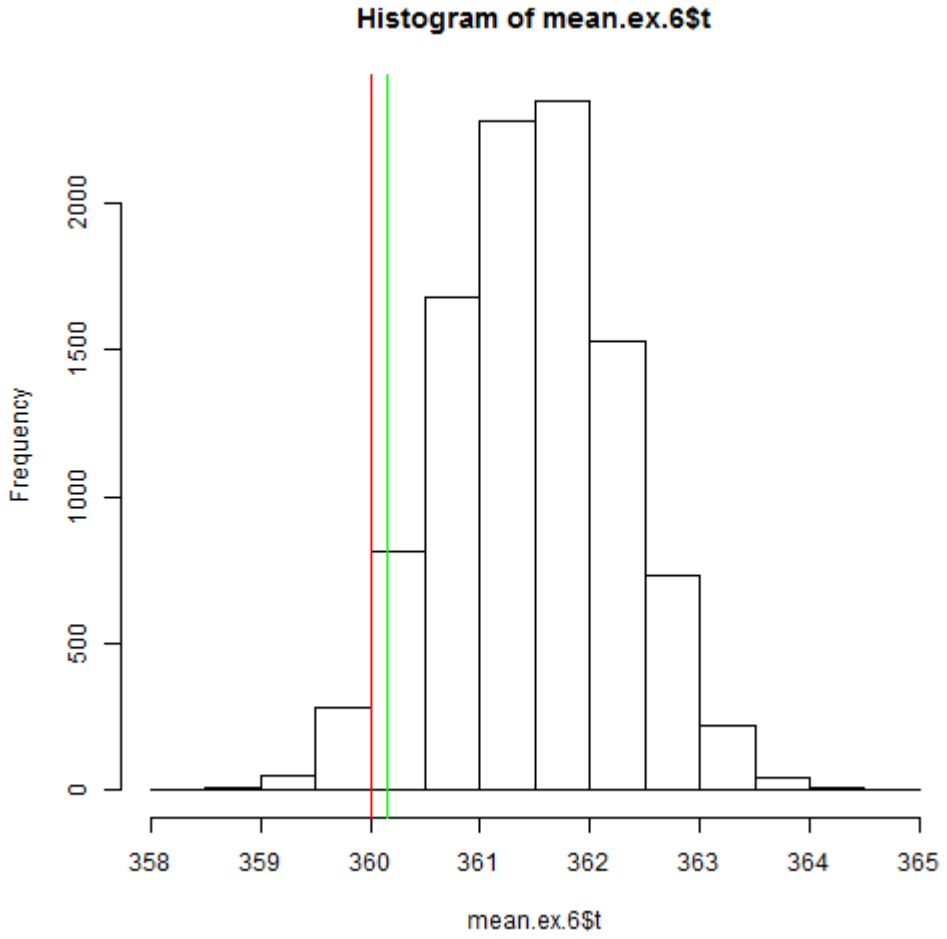
```
library(boot)   prob.boot<-function(data,indices)   {
return(sum(data[indices,1]>5&data[indices,1]<10)/length(data[i
ndices,1]))     }     set.seed(42)     prob.5.10<-
boot(data.ex.5,prob.boot,R=10000) print(prob.5.10)
```

```
## ## ORDINARY NONPARAMETRIC BOOTSTRAP ## ## ## Call: ##
boot(data = data.ex.5, statistic = prob.boot, R = 10000) ## ##
## Bootstrap Statistics : ## original bias std. error ## t1*
0.1733333 -0.0001986667 0.03074905
```

```
quantile(prob.5.10$t,c(0.025,0.975))
```

```
## 2.5% 97.5% ## 0.1133333 0.2333333
```

```
#2                    ((quantile(prob.5.10$t,0.975)-quantile(prob.5.10$t,0.025,0.975))/2)*100

## 97.5% ## 6

#####################  #  #  #  Exercise  6  #  #  ###################### #1 data.ex.6<-read.csv("http://www.r-exercises.com/wp-content/uploads/2017/08/data.ex_.6.csv") str(data.ex.6)

## 'data.frame': 200 obs. of 1 variable: ## $ x: num 373 357 367 363 346 ...

mean(data.ex.6$x)

## [1] 361.4657

#2 mean.boot<-function(data,indices) { return(mean(data[indices,])) } set.seed(42) mean.ex.6<-boot(data.ex.6,mean.boot,R=10000) print(mean.ex.6)

## ## ORDINARY NONPARAMETRIC BOOTSTRAP ## ## ## Call: ## boot(data = data.ex.6, statistic = mean.boot, R = 10000) ## ## ## Bootstrap Statistics : ## original bias std. error ## t1* 361.4657 0.003623713 0.8038724

(sum(mean.ex.6$t<=360)/length(mean.ex.6$t))*100

## [1] 3.39

#Since this probability is smaller than 5%, we reject the null hypothesis and conclude that the mean is bigger than 360. At a level of 1% we cannot reject the null hypothesis. ####################  #  #  #  Exercise  7  #  #  ##################### hist(mean.ex.6$t) abline(v=360,col="red") abline(v=quantile(mean.ex.6$t,0.05),col="green")
```




```
#####################  #  #  #  Exercise  8  #  #  #
#####################    #1    data.ex.8<-data.ex.6-
mean(data.ex.6$x)+363   #2   set.seed(42)   mean.ex.8<-
boot(data.ex.8,mean.boot,R=10000)   #3   endpoint.1<-
mean(data.ex.6$x)   endpoint.2<-363-endpoint.1+363   #4
(sum((mean.ex.8$t<=endpoint.1)|(mean.ex.8$t>=endpoint.2))/leng
th(mean.ex.8$t))*100
```

```
## [1] 5.56
```

```
#5 #Since 5.56% of the results are more far from 363 than our
average, we cannot say that our average is statistically
different from 363. We cannot reject the null hypothesis and
conclude  that  the  mean  is  equal  to  363  at  level  5%.
#####################  #  #  #  Exercise  9  #  #  #
#####################
(sum(mean.ex.8$t<=endpoint.1)/length(mean.ex.8$t))*100
```

```
## [1] 2.81
```

```
#Since only 2.81% of the results are as extreme as our average, we can claim that such a result would be rare if the mean was of 363. As a consequence, we reject the null hypothesis and conclude that the mean is significantly smaller than 363. ##################### # # # Exercise 10 # # # ##################### data.ex.9<-data.ex.5-mean(data.ex.5$x)+13 set.seed(42)   mean.ex.9<-boot(data.ex.9,mean.boot,R=10000) endpoint.1<-mean(data.ex.5$x)   endpoint.2<-13-endpoint.1+13 (sum((mean.ex.9$t<=endpoint.1)|(mean.ex.9$t>=endpoint.2))/length(mean.ex.9$t))*100
```

```
## [1] 0.03
```

---

ggvis Exercises (Part-1): Solutions

Below are the solutions to these exercises on visualizations with ggvis.

```
#####################  # # # Exercise 1 # # #  #####################  library(ggvis)  library(MASS) library(magrittr) attach(Cars93) plot1 <- ggvis(Cars93, x = ~Horsepower, y = ~MPG.city) ##################### # # # Exercise 2 # # # ##################### library(ggvis) library(MASS) library(magrittr) attach(Cars93) plot1 <- ggvis(Cars93, x = ~Horsepower, y = ~MPG.city) layer_points(plot1) ##################### # # # Exercise 3 # # #  #####################  library(ggvis) library(MASS) library(magrittr) attach(Cars93) Cars93 %>% ggvis(x = ~Horsepower, y = ~MPG.city) %>% layer_points() ######################  # # # Exercise 4 # # #  ######################  library(ggvis) library(MASS) library(magrittr) attach(Cars93) Cars93 %>% ggvis(~Horsepower, ~MPG.city, stroke = ~Cylinders) %>% layer_points() ######################  # # # Exercise 5 # # #
```

```
#####################  library(ggvis)  library(MASS)
library(magrittr) attach(Cars93) Cars93 %>% ggvis(~Horsepower,
~MPG.city, fill = ~Cylinders) %>% layer_points()
#####################  #  #  #  Exercise  6  #  #  #
#####################  library(ggvis)  library(MASS)
library(magrittr) attach(Cars93) Cars93 %>% ggvis(~Horsepower,
~MPG.city, size = ~EngineSize) %>% layer_points()
#####################  #  #  #  Exercise  7  #  #  #
#####################  library(ggvis)  library(MASS)
library(magrittr) attach(Cars93) Cars93 %>% ggvis(~Horsepower,
~MPG.city, shape = ~factor(Cylinders)) %>% layer_points()
#####################  #  #  #  Exercise  8  #  #  #
#####################  library(ggvis)  library(MASS)
library(magrittr) attach(Cars93) Cars93 %>% ggvis(~Horsepower,
~MPG.city, fill := "red", stroke := "black") %>%
layer_points() ################### # # # Exercise 9 # # #
#####################  library(ggvis)  library(MASS)
library(magrittr) attach(Cars93) Cars93 %>% ggvis(~Horsepower,
~MPG.city, size := 300, opacity := 0.5) %>% layer_points()
#####################  #  #  #  Exercise  10  #  #  #
#####################  library(ggvis)  library(MASS)
library(magrittr) attach(Cars93) Cars93 %>% ggvis(~Horsepower,
~MPG.city, shape := "cross") %>% layer_points()
```

More String Hacking with Regex and Rebus Solution

Below are the solutions to these exercises on string manipulation .

```
library(stringi)  library(stringr)  library(rebus)
############## # # # Exercise 1 # # # ############## x <-
c("stringer","stringi","rebus","redbus") stri_subset(x,regex =
START %R% "st") ############## # # # Exercise 2 # # #
##############  stri_subset(x,regex = "bus" %R% END)
```

```
############### # # # Exercise 3 # # # ############### m <- c("aba","aca","abba","accdda") pattern <- "a" %R% ANY_CHAR %R% "a" stri_subset(m,regex = pattern) ############### # # # Exercise 4 # # # ############### y <- c("brain","brawn","rain","train") reg = START %R% "br" %R% any_char(1,Inf) %R% "n" %R% END stri_subset(y,regex = reg) ############### # # # Exercise 5 # # # ############### rg <- START %R% or("br","tr") stri_subset(y,regex = rg)l ############### # # # Exercise 6 # # # ############### l <- c("Canada","america","france") rgx <- char_class("Cm") stri_subset(l,regex = rgx) ############### # # # Exercise 7 # # # ############### stri_extract("123abc",regex = one_or_more(DIGIT)) ############### # # # Exercise 8 # # # ############### vow <- c("blue","sue","CLUE","TRUE") str_match(vow,one_or_more(char_class("aeiouAEIOU"))) ############### # # # Exercise 9 # # # ############### str_match(vow,one_or_more(negated_char_class("aeiouAEIOU"))) ############### # # # Exercise 10 # # # ############### vow1 <- c("blue","sue","CLUE","TRUE","aue") stri_subset(vow,regex= exactly(one_or_more(char_class("aeiouAEIOU"))))
```

Soccer data sparring: Scraping, merging and analyzing Solutions

Below are the solutions to these exercises on Soccer data sparring: Scraping, merging and analyzing.

```
###################### # # # Exercise 1 # # # ###################### # Using package rvest library(rvest) # Webpage that contains links to the data page <- html("http://www.football-data.co.uk/germanym.php")

## Warning: 'html' is deprecated. ## Use 'read_html' instead.
```

```
## See help("Deprecated")


# Get all links contained in page all_links <- page %>% html_nodes("a") %>% # find all links html_attr("href") # get the urls # Remove page from workspace objects rm(page) ##################### # # # Exercise 2 # # # ##################### # Keep only files containg ".csv" all_links <- grep("\\.csv", all_links, value = TRUE) ##################### # # # Exercise 3 # # # ##################### # Starts at 93/94, so only need to rid of those starting 14 and after # Extract the year the seaons starts season_start <- as.numeric(gsub(".+/([0-9]{2})[0-9]{2}/.+", "\\1", all_links)) all_links <- all_links[!season_start %in% 14:50] # When this code was written the newest data was from season 17-18, so # %in% 14:17 would suffice rm(season_start) # Now limit to Bunedsliga 1 all_links <- all_links[gsub(".+(.{2}).csv$", "\\1", all_links) == "D1"] ##################### # # # Exercise 4 # # # ##################### # Open an empty list data_list <- list() # Load to all the csv files to our list, step by step for (csv_file in all_links) { data_id <- paste0("season_", gsub(".+/([0-9]{4})/.+", "\\1", csv_file)) cat("Starting to read in", data_id, "...") data_list[[data_id]] <- read.csv(paste0("http://www.football-data.co.uk/", csv_file), na.strings = c("", "NA")) cat(" Done, fantastisch!\n") }


## Starting to read in season_1314 ... Done, fantastisch! ##
Starting to read in season_1213 ... Done, fantastisch! ##
Starting to read in season_1112 ... Done, fantastisch! ##
Starting to read in season_1011 ... Done, fantastisch! ##
Starting to read in season_0910 ... Done, fantastisch! ##
Starting to read in season_0809 ... Done, fantastisch! ##
Starting to read in season_0708 ... Done, fantastisch! ##
Starting to read in season_0607 ... Done, fantastisch! ##
Starting to read in season_0506 ... Done, fantastisch! ##
Starting to read in season_0405 ... Done, fantastisch! ##
Starting to read in season_0304 ... Done, fantastisch! ##
Starting to read in season_0203 ... Done, fantastisch! ##
Starting to read in season_0102 ... Done, fantastisch! ##
Starting to read in season_0001 ... Done, fantastisch! ##
Starting to read in season_9900 ... Done, fantastisch! ##
```

```
Starting to read in season_9899 ... Done, fantastisch! ##
Starting to read in season_9798 ... Done, fantastisch! ##
Starting to read in season_9697 ... Done, fantastisch! ##
Starting to read in season_9596 ... Done, fantastisch! ##
Starting to read in season_9495 ... Done, fantastisch! ##
Starting to read in season_9394 ... Done, fantastisch!

rm(csv_file, data_id, all_links) ##################### # # # Exercise 5 # # # ##################### library(plyr) bundesl <- rbind.fill(data_list) ##################### # # # Exercise 6 # # # ##################### empty_row <- rowSums(is.na(bundesl)) == ncol(bundesl) sum(empty_row) # 1327 empty rows

## [1] 1327

empty_col <- sapply(bundesl, function(x) sum(is.na(x)) == nrow(bundesl)) sum(empty_col) # 21 empty columns

## [1] 21

# Continue without those bundesl <- bundesl[!empty_row, !empty_col] # Remove unnecessary stuff from workspace rm(empty_row, empty_col) ##################### # # # Exercise 7 #           #           #           ######################
summary(nchar(as.character(bundesl$Date)))

## Min. 1st Qu. Median Mean 3rd Qu. Max. ## 8 8 8 8 8 8

head(bundesl$Date)

## [1] 09/08/13 10/08/13 10/08/13 10/08/13 10/08/13 10/08/13
## 1802 Levels: 01/02/14 01/03/14 01/09/13 01/11/13 01/12/13 ... 30/10/93

bundesl$Date <- as.Date(bundesl$Date , "%d/%m/%y") # Sanity check, we know what range our dates should fall within...
summary(bundesl$Date)

## Min. 1st Qu. Median Mean 3rd Qu. ## "1993-08-07"
```

```
"1998-10-24" "2004-01-08" "2004-01-02" "2009-04-04" ## Max. ## "2014-05-10"


#####################  #  #  #  Exercise  8  #  #  ##################### full_col <- sapply(bundesl, function(x) sum(is.na(x)) == 0L) bundesl <- bundesl[, !names(bundesl) %in% c("Div", names(full_col[!full_col]))] # Clean up rm(full_col) ######################  #  #  #  Exercise  9  #  #  ###################### # One team has been entered with inconsistent capitalization in its name # Which can be fixed by just working with lowercase versions of the naems bundesl$HomeTeam <- tolower(as.character(bundesl$HomeTeam)) bundesl$AwayTeam <- tolower(as.character(bundesl$AwayTeam)) # Also be careful that some teams have no away, others no home wins.. tot_wins <- table(bundesl$HomeTeam, bundesl$FTR == "H")[, 2] + table(bundesl$AwayTeam, bundesl$FTR == "A")[, 2] sort(tot_wins, decreasing = TRUE)[1:3]

## bayern munich dortmund leverkusen ## 444 345 338

# The top three winners are Bayern Munich, Dortmund, Leverkusen ###################### # # # Exercise 10 # # # ###################### # Since we are considering both home and away games we need to reorder data # Take it to long form with the necessary variables # We will use rle and melt from data.table library(data.table) bundesl_long <- melt(bundesl, id.vars = c("FTR", "Date"), measure.vars = c("HomeTeam", "AwayTeam"))  bundesl_long$win  <- substring(bundesl_long$variable, 1, 1) == bundesl_long$FTR # Convert to data.table from data.frame setDT(bundesl_long) ## First some sanity checks... # half of non-draw games should result in a win summary(bundesl_long$win[bundesl_long$FTR != "D"])

## Mode FALSE TRUE ## logical 4772 4772

# We should get same results as in last exercise about number of wins bundesl_long[, sum(win) , by = value][order(V1, decreasing = TRUE)][1:3]
```

```
## value V1 ## 1: bayern munich 444 ## 2: dortmund 345 ## 3: leverkusen 338

# There should be an even number of entries per date bundesl_long[, .N %% 2L, by = Date][V1 != ]

## Empty data.table (0 rows) of 2 cols: Date,V1

# All sanity checks passed # Now order by team and date setkey(bundesl_long, value, Date) # Calculate streaks.. and selecting only winning streaks bundesl_long[, rle(win), by = value][values == TRUE, max(lengths), by = value][order(V1)]

## value V1 ## 1: braunschweig 1 ## 2: fortuna dusseldorf 1 ## 3: greuther furth 1 ## 4: leipzig 1 ## 5: uerdingen 1 ## 6: wattenscheid 1 ## 7: dusseldorf 2 ## 8: unterhaching 2 ## 9: aachen 3 ## 10: augsburg 3 ## 11: bochum 3 ## 12: cottbus 3 ## 13: dresden 3 ## 14: karlsruhe 3 ## 15: st pauli 3 ## 16: ulm 3 ## 17: bielefeld 4 ## 18: fc koln 4 ## 19: freiburg 4 ## 20: hamburg 4 ## 21: hansa rostock 4 ## 22: nurnberg 4 ## 23: duisburg 5 ## 24: hannover 5 ## 25: hertha 5 ## 26: hoffenheim 5 ## 27: munich 1860 5 ## 28: ein frankfurt 6 ## 29: m'gladbach 6 ## 30: schalke 04 6 ## 31: werder bremen 6 ## 32: kaiserslautern 7 ## 33: mainz 7 ## 34: dortmund 8 ## 35: leverkusen 8 ## 36: stuttgart 8 ## 37: wolfsburg 10 ## 38: bayern munich 19 ## value V1

# Again Bayern munich comes out on top with 19 games in a row!
```

Data visualization with googleVis solutions part 10

Below are the solutions to these exercises on visualizations with googleVis.

```
#################### # # # Exercise 1 # # # ####################
library(googleVis) TLC <- gvisTimeline(data=datTLc) ################## # # # Exercise 2 # # # ##################
library(googleVis) TLC <- gvisTimeline(data=datTLc) plot(TLC) ################## # # # Exercise 3 # # # ##################
library(googleVis) TLC <- gvisTimeline(data=datTLc, rowlabel="Name", barlabel="Position", start="start", end="end") plot(TLC)
#################### # # # Exercise 4 # # # ####################
library(googleVis) TLC <- gvisTimeline(data=datTLc, rowlabel="Name", barlabel="Position", start="start", end="end", options=list(timeline="{groupByRowLabel:false}")) plot(TLC)
#################### # # # Exercise 5 # # # ####################
library(googleVis) TLC <- gvisTimeline(data=datTLc, rowlabel="Name", barlabel="Position", start="start", end="end", options=list(timeline="{groupByRowLabel:false}", backgroundColor='white', height=400, colors="['red', 'green']")) plot(TLC) ################## # # # Exercise 6 # # # ####################
library(googleVis) Geo <- gvisGeoChart(Exports, "Country", "Profit", options=list(width=300, height=300)) Table <- gvisTable(Exports, options=list(width=220, height=300)) GeoTable <- gvisMerge(Geo,Table, horizontal=TRUE) plot(GeoTable) ################## # # # Exercise 7 # # # ####################
library(googleVis) MotionC=gvisMotionChart(Fruits, idvar = "Fruit", timevar = "Year" ) ################## # # # Exercise 8 # # # ####################
library(googleVis) MotionC=gvisMotionChart(Fruits, idvar = "Fruit", timevar = "Year" ) plot(MotionC) ################## # # # Exercise 9 # # # ####################
library(googleVis) MotionC=gvisMotionChart(Fruits, idvar = "Fruit", timevar = "Year", xvar = "Expenses", yvar = "Sales", sizevar ="Profit", colorvar = "Location") ################## # # # Exercise 10 # # # ####################
library(googleVis) MotionC=gvisMotionChart(Fruits, idvar = "Fruit", timevar = "Year", xvar = "Expenses", yvar = "Sales", sizevar ="Profit", colorvar = "Location") plot(MotionC)
```

R Markdown exercises part 2: solutions

Below are the solutions to these RMarkdown exercises.

```
#####################  #  #  #  Exercise  1  #  #  #  ##################### --- title: "R Markdown Example" author: "Makis" date: 18/Jul/2017 output: html_document --- ```{r,echo=FALSE}  summary(cars)  ```  ```{r,fig.width=5, fig.height=5,  echo=FALSE,dev='svg'}  plot(cars)  ```  ```{r, echo=FALSE} A <- c("Bob", "Tom", "Bill", "Joe") B <- c(1.78, 1.86, 1.85, 1.70) dataframe <- data.frame(A, B) dataframe ``` ```{r,echo=FALSE,warning=FALSE}  library(knitr) kable(dataframe, digits = 1) ``` ##################### # # # Exercise 2 # # # ##################### --- title: "**R Markdown Example**" author: "Makis" date: 18/Jul/2017 output: html_document  ---  ```{r,echo=FALSE}  summary(cars)  ```  ```{r,fig.width=5,  fig.height=5,  echo=FALSE,dev='svg'}  plot(cars)  ```  ```{r, echo=FALSE} A <- c("Bob", "Tom", "Bill", "Joe")  B  <-  c(1.78,  1.86,  1.85,  1.70)  dataframe  <- data.frame(A, B) dataframe ``` ```{r,echo=FALSE,warning=FALSE}  library(knitr)  kable(dataframe,  digits  =  1)  ```  #####################  #  #  #  Exercise  3  #  #  #  ##################### --- title: "**R Markdown Example**" author: "*Makis*" date: 18/Jul/2017 output: html_document --- ```{r,echo=FALSE}  summary(cars)  ```  ```{r,fig.width=5, fig.height=5,  echo=FALSE,dev='svg'}  plot(cars)  ```  ```{r, echo=FALSE} A <- c("Bob", "Tom", "Bill", "Joe") B <- c(1.78, 1.86, 1.85, 1.70) dataframe <- data.frame(A, B) dataframe ``` ```{r,echo=FALSE,warning=FALSE}  library(knitr) kable(dataframe, digits = 1) ``` ##################### # # # Exercise 4 # # # ##################### --- title: "**R Markdown Example**" author: "*Makis*" date: 18/Jul/2017 output: html_document --- # Summary ```{r,echo=FALSE} summary(cars) ```  ```{r,fig.width=5,  fig.height=5,  echo=FALSE,dev='svg'}  plot(cars)  ```  ```{r, echo=FALSE} A <- c("Bob", "Tom", "Bill",
```

"Joe") B <- c(1.78, 1.86, 1.85, 1.70) dataframe <- data.frame(A, B) dataframe ``` ```{r,echo=FALSE,warning=FALSE} library(knitr) kable(dataframe, digits = 1) ``` ########## # # # Exercise 5 # # # ########## --- title: "**R Markdown Example**" author: "*Makis*" date: 18/Jul/2017 output: html_document --- # Summary ```{r,echo=FALSE} summary(cars) ``` ### Plot ```{r,fig.width=5, fig.height=5, echo=FALSE,dev='svg'} plot(cars) ``` ```{r, echo=FALSE} A <- c("Bob", "Tom", "Bill", "Joe") B <- c(1.78, 1.86, 1.85, 1.70) dataframe <- data.frame(A, B) ``` ### Dataframe ```{r, echo=FALSE} dataframe ``` ### Table 1 ```{r,echo=FALSE,warning=FALSE} library(knitr) kable(dataframe, digits = 1) ``` ########## # # # Exercise 6 # # # ########## --- title: "**R Markdown Example**" author: "*Makis*" date: 18/Jul/2017 output: html_document --- # Summary ```{r,echo=FALSE} summary(cars) ``` ### Plot ```{r,fig.width=5, fig.height=5, echo=FALSE,dev='svg'} plot(cars) ``` ```{r, echo=FALSE} A <- c("Bob", "Tom", "Bill", "Joe") B <- c(1.78, 1.86, 1.85, 1.70) dataframe <- data.frame(A, B) ``` ### Dataframe ```{r, echo=FALSE} dataframe ``` ### Table 1 ```{r,echo=FALSE,warning=FALSE} library(knitr) kable(dataframe, digits = 1) ``` | A | B | |:---|:---:| | Bob | 1.78| | Tom | 1.86| | Bill | 1.85| | Joe | 1.70| ########## # # # Exercise 7 # # # ########## --- title: "**R Markdown Example**" author: "*Makis*" date: 18/Jul/2017 output: html_document --- # Summary ```{r,echo=FALSE} summary(cars) ``` ### Plot ```{r,fig.width=5, fig.height=5, echo=FALSE,dev='svg'} plot(cars) ``` ```{r, echo=FALSE} A <- c("Bob", "Tom", "Bill", "Joe") B <- c(1.78, 1.86, 1.85, 1.70) dataframe <- data.frame(A, B) ``` ### Dataframe ```{r, echo=FALSE} dataframe ``` ### Table 1 ```{r,echo=FALSE,warning=FALSE} library(knitr) kable(dataframe, digits = 1) ``` | A | B | |:---| ---:| | Bob | 1.78| | Tom | 1.86| | Bill | 1.85| | Joe | 1.70| ########## # # # Exercise 8 # # # ########## --- title: "**R Markdown Example**" author: "*Makis*" date: 18/Jul/2017 output: html_document --- # Summary ```{r,echo=FALSE} summary(cars) ``` ### Plot ```{r,fig.width=5, fig.height=5, echo=FALSE,dev='svg'} plot(cars) ``` ```{r, echo=FALSE} A <- c("Bob", "Tom", "Bill",

"Joe") B <- c(1.78, 1.86, 1.85, 1.70) dataframe <- data.frame(A, B) ``` ### Dataframe ```{r, echo=FALSE} dataframe ``` ### Table 1 ```{r,echo=FALSE,warning=FALSE} library(knitr) kable(dataframe, digits = 1) ``` | A | B | |:------| ---:| | Bob | 1.78| | Tom | 1.86| | Bill | 1.85| | Joe | 1.70| * Bob * Tom * Bill * Joe ##################### # # # Exercise 9 # # # ##################### --- title: "**R Markdown Example**" author: "*Makis*" date: 18/Jul/2017 output: html_document --- # Summary ```{r,echo=FALSE} summary(cars) ``` ### Plot ```{r,fig.width=5, fig.height=5, echo=FALSE,dev='svg'} plot(cars) ``` ```{r, echo=FALSE} A <- c("Bob", "Tom", "Bill", "Joe") B <- c(1.78, 1.86, 1.85, 1.70) dataframe <- data.frame(A, B) ``` ### Dataframe ```{r, echo=FALSE} dataframe ``` ### Table 1 ```{r,echo=FALSE,warning=FALSE} library(knitr) kable(dataframe, digits = 1) ``` | A | B | |:-----| ---:| | Bob | 1.78| | Tom | 1.86| | Bill | 1.85| | Joe | 1.70| 1. Bob 2. Tom 3. Bill 4. Joe ##################### # # # Exercise 10 # # # ##################### --- title: "**R Markdown Example**" author: "*Makis*" date: 18/Jul/2017 output: html_document --- # Summary ```{r,echo=FALSE} summary(cars) ``` ### Plot ```{r,fig.width=5, fig.height=5, echo=FALSE,dev='svg'} plot(cars) ``` ```{r, echo=FALSE} A <- c("Bob", "Tom", "Bill", "Joe") B <- c(1.78, 1.86, 1.85, 1.70) dataframe <- data.frame(A, B) ``` ### Dataframe ```{r, echo=FALSE} dataframe ``` ### Table 1 ```{r,echo=FALSE,warning=FALSE} library(knitr) kable(dataframe, digits = 1) ``` | A | B | |:------| ---:| | Bob | 1.78| | Tom | 1.86| | Bill | 1.85| | Joe | 1.70| 1. Bob 2. Tom 3. Bill 4. Joe [Link](https://www.r-exercises.com)

Parallel Computing Exercises:

Snow and Rmpi (Part-3) Solutions

Below are the solutions to these exercises on parallel computing with snow and Rmpi.

```
#####################  #  #  #  Exercise  1  #  #  #  ###################### data_large <- read.csv("InstEval_reduced.csv") ##################### # # # Exercise 2 # # # ##################### # create a smaller data set set.seed(1234) selected_rows <- sample(x = 1:nrow(data_large), size = nrow(data_large)/10) data_small <- data_large[selected_rows, ] # print the number of rows in both data sets print(nrow(data_large))
```

```
## [1] 10000
```

```
print(nrow(data_small))
```

```
## [1] 1000
```

```
#####################  #  #  #  Exercise  3  #  #  #  ##################### regress_resampled_data <- function(data) { rows_resampled <- sample(x = 1:nrow(data), size = nrow(data), replace = TRUE) data_resampled <- data[rows_resampled, ] fit <- lm(y ~ ., data = data_resampled) return(fit$coefficients) } ##################### # # # Exercise 4 # # # ##################### require(foreach) # run the function 10 times with the large dataset sequentially system.time( result_large <- foreach(1:10, .combine = rbind) %do% regress_resampled_data (data_large) )
```

```
## user system elapsed ## 6.297 0.149 6.447
```

```
# run the function 10 times with the large dataset sequentially system.time( result_small <- foreach(1:100, .combine = rbind) %do% regress_resampled_data (data_small) )
```

```
## user system elapsed ## 7.500 0.011 7.513

# The time spent is: # 6.447 s in the first case, and # 7.513 s in the second case # The difference may be attributed to  # the foreach loop overhead # (it is very likely that you have other figures due to difference # in hardware and software configuration). ##################### # # # Exercise 5 # # # ######################  require(snow)  require(doSNOW) cluster_snow  <-  makeCluster(2,  type="SOCK") registerDoSNOW(cluster_snow) ##################### # # # Exercise 6 # # # ##################### require(foreach) require(snow) system.time( result_large <- foreach(1:10, .combine = rbind) %dopar% regress_resampled_data (data_large) )

## user system elapsed ## 0.054 0.015 3.814

stopCluster(cluster_snow) # The execution time is 3.814 s. # It is 41% faster comparing to the task run sequentially. ######################  # # # Exercise 7 # # # ######################  require(snow)  require(doSNOW) require(foreach) # prepare a cluster cluster_snow <- makeCluster(2, type="SOCK") registerDoSNOW(cluster_snow) # run the task 100 times system.time( result_small <- foreach(1:100, .combine = rbind) %dopar% regress_resampled_data (data_small) )

## user system elapsed ## 0.114 0.014 6.253

# stop the cluster stopCluster(cluster_snow) # The execution time is 6.253 s. # It is only 17% faster comparing to the task run sequentially. # This reflect the fact that the overhead related to inter-process communication # can be significant (the shorter is the repeated task, the more noticeable is  # this overhead). # In some cases, a computation in parallel can even take more time # than a sequential one. ######################  # # # Exercise 8 # # # ######################  require(doMPI) cluster_mpi <- startMPIcluster(2)
```

```
## 2 slaves are spawned successfully. 0 failed.

registerDoMPI(cluster_mpi) ##################### # # # Exercise 9 # # # ###################### require(doMPI) closeCluster(cluster_mpi) ##################### # # # Exercise 10 # # # ##################### require(doMPI) require(foreach) # run the task 10 times with the data_large data set cluster_mpi <- startMPIcluster(count = 2)

## 2 slaves are spawned successfully. 0 failed.

registerDoMPI(cluster_mpi) system.time( result_large <- foreach(1:10, .combine = rbind) %dopar% regress_resampled_data (data_large) )

## user system elapsed ## 2.018 2.001 4.472

 closeCluster (cluster_mpi) # run the task 100 times with the data_small data set cluster_mpi <- startMPIcluster(count = 2)

## 2 slaves are spawned successfully. 0 failed.

registerDoMPI(cluster_mpi) system.time( result_small <- foreach(1:100, .combine = rbind) %dopar% regress_resampled_data (data_small) )

## user system elapsed ## 5.227 0.835 6.139

closeCluster(cluster_mpi) # when you finish work mpi.finalize()

## [1] 1

# for the large data set, parallel execution with Rmpi is # 31% faster than sequential one, but 17% slower # than parallel execution with snow # (note that in other cases the situation may be different, and  # computations with snow may be faster) # # for the small data set, parallel execution with Rmpi is 18% faster  # than sequential one, and 2% faster than parallel
```

execution with snow

Data wrangling : Transforming (3/3) Solutions

Below are the solutions to these exercises on data transformation.

```
#####################  #  #  #  Exercise  1  #  #  #  ###################  cars_inner <-  inner_join(mtcars, cars_table, by = 'ID') ; cars_inner
```

```
## mpg cyl disp hp drat wt qsec vs am gear carb ID ## 1 21.0 6 160.0 110 3.90 2.875 17.02 0 1 4 4 Mazda86 ## 2 22.8 4 108.0 93 3.85 2.320 18.61 1 1 4 1 Datsun73 ## 3 18.7 8 360.0 175 3.15 3.440 17.02 0 0 3 2 Hornet350 ## 4 18.1 6 225.0 105 2.76 3.460 20.22 1 0 3 1 Valiant392 ## 5 19.2 6 167.6 123 3.92 3.440 18.30 1 0 4 4 Merc217 ## 6 17.3 8 275.8 180 3.07 3.730 17.60 0 0 3 3 Merc172 ## 7 15.2 8 275.8 180 3.07 3.780 18.00 0 0 3 3 Merc175 ## 8 10.4 8 460.0 215 3.00 5.424 17.82 0 0 3 4 Lincoln219 ## 9 32.4 4 78.7 66 4.08 2.200 19.47 1 1 4 1 Fiat31 ## 10 21.5 4 120.1 97 3.70 2.465 20.01 1 0 3 1 Toyota-47 ## 11 13.3 8 350.0 245 3.73 3.840 15.41 0 0 3 4 Camaro54 ## 12 19.2 8 400.0 175 3.08 3.845 17.05 0 0 3 2 Pontiac306 ## 13 15.8 8 351.0 264 4.22 3.170 14.50 0 1 5 4 Ford-111 ## 14 19.7 6 145.0 175 3.62 2.770 15.50 0 1 5 6 Ferrari269 ## 15 15.0 8 301.0 335 3.54 3.570 14.60 0 1 5 8 Maserati168 ## 16 21.4 4 121.0 109 4.11 2.780 18.60 1 1 4 2 Volvo47 ## performance year ## 1 5.238095 2000 ## 2 4.078947 2007 ## 3 9.358289 2018 ## 4 5.801105 1999 ## 5 6.406250 2004 ## 6 10.404624 1985 ## 7 11.842105 1984 ## 8 20.673077 2009 ## 9 2.037037 2015 ## 10 4.511628 2006 ## 11 18.421053 2004 ## 12 9.114583 2003 ## 13 16.708861 1984 ## 14 8.883249 1998 ## 15 22.333333 2008 ## 16 5.093458 1998
```

```
#####################  #  #  #  Exercise  2  #  #  #  #####################  cars_left  <-  left_join(mtcars,  cars_table,  by  =  'ID');  cars_left
```

```
## mpg cyl disp hp drat wt qsec vs am gear carb ID ## 1 21.0 6 160.0 110 3.90 2.620 16.46 0 1 4 4 Mazda185 ## 2 21.0 6 160.0 110 3.90 2.875 17.02 0 1 4 4 Mazda86 ## 3 22.8 4 108.0 93 3.85 2.320 18.61 1 1 4 1 Datsun73 ## 4 21.4 6 258.0 110 3.08 3.215 19.44 1 0 3 1 Hornet6 ## 5 18.7 8 360.0 175 3.15 3.440 17.02 0 0 3 2 Hornet350 ## 6 18.1 6 225.0 105 2.76 3.460 20.22 1 0 3 1 Valiant392 ## 7 14.3 8 360.0 245 3.21 3.570 15.84 0 0 3 4 Duster128 ## 8 24.4 4 146.7 62 3.69 3.190 20.00 1 0 4 2 Merc-150 ## 9 22.8 4 140.8 95 3.92 3.150 22.90 1 0 4 2 Merc259 ## 10 19.2 6 167.6 123 3.92 3.440 18.30 1 0 4 4 Merc217 ## 11 17.8 6 167.6 123 3.92 3.440 18.90 1 0 4 4 Merc336 ## 12 16.4 8 275.8 180 3.07 4.070 17.40 0 0 3 3 Merc259 ## 13 17.3 8 275.8 180 3.07 3.730 17.60 0 0 3 3 Merc172 ## 14 15.2 8 275.8 180 3.07 3.780 18.00 0 0 3 3 Merc175 ## 15 10.4 8 472.0 205 2.93 5.250 17.98 0 0 3 4 Cadillac109 ## 16 10.4 8 460.0 215 3.00 5.424 17.82 0 0 3 4 Lincoln219 ## 17 14.7 8 440.0 230 3.23 5.345 17.42 0 0 3 4 Chrysler92 ## 18 32.4 4 78.7 66 4.08 2.200 19.47 1 1 4 1 Fiat31 ## 19 30.4 4 75.7 52 4.93 1.615 18.52 1 1 4 2 Honda207 ## 20 33.9 4 71.1 65 4.22 1.835 19.90 1 1 4 1 Toyota268 ## 21 21.5 4 120.1 97 3.70 2.465 20.01 1 0 3 1 Toyota-47 ## 22 15.5 8 318.0 150 2.76 3.520 16.87 0 0 3 2 Dodge212 ## 23 15.2 8 304.0 150 3.15 3.435 17.30 0 0 3 2 AMC148 ## 24 13.3 8 350.0 245 3.73 3.840 15.41 0 0 3 4 Camaro54 ## 25 19.2 8 400.0 175 3.08 3.845 17.05 0 0 3 2 Pontiac306 ## 26 27.3 4 79.0 66 4.08 1.935 18.90 1 1 4 1 Fiat83 ## 27 26.0 4 120.3 91 4.43 2.140 16.70 0 1 5 2 Porsche289 ## 28 30.4 4 95.1 113 3.77 1.513 16.90 1 1 5 2 Lotus139 ## 29 15.8 8 351.0 264 4.22 3.170 14.50 0 1 5 4 Ford-111 ## 30 19.7 6 145.0 175 3.62 2.770 15.50 0 1 5 6 Ferrari269 ## 31 15.0 8 301.0 335 3.54 3.570 14.60 0 1 5 8 Maserati168 ## 32 21.4 4 121.0 109 4.11 2.780 18.60 1 1 4 2 Volvo47 ## performance year ## 1 NA NA ## 2 5.238095 2000 ## 3 4.078947 2007 ## 4 NA NA ## 5 9.358289 2018 ## 6 5.801105 1999 ## 7 NA NA ## 8 NA NA ## 9 NA NA ## 10 6.406250 2004 ## 11 NA NA ## 12 NA NA ## 13 10.404624 1985 ## 14 11.842105 1984 ## 15 NA NA ## 16 20.673077 2009 ## 17 NA NA ## 18 2.037037 2015 ## 19 NA NA ## 20 NA NA ## 21 4.511628 2006 ## 22 NA NA ## 23 NA
```

NA ## 24 18.421053 2004 ## 25 9.114583 2003 ## 26 NA NA ## 27 NA NA ## 28 NA NA ## 29 16.708861 1984 ## 30 8.883249 1998 ## 31 22.333333 2008 ## 32 5.093458 1998

```
#####################  #  #  #  Exercise  3  #  #  #  #####################  cars_right  <-  right_join(mtcars,  cars_table,  by  =  'ID');  cars_right
```

mpg cyl disp hp drat wt qsec vs am gear carb ID ## 1 21.5 4 120.1 97 3.70 2.465 20.01 1 0 3 1 Toyota-47 ## 2 18.7 8 360.0 175 3.15 3.440 17.02 0 0 3 2 Hornet350 ## 3 19.7 6 145.0 175 3.62 2.770 15.50 0 1 5 6 Ferrari269 ## 4 15.8 8 351.0 264 4.22 3.170 14.50 0 1 5 4 Ford-111 ## 5 18.1 6 225.0 105 2.76 3.460 20.22 1 0 3 1 Valiant392 ## 6 21.4 4 121.0 109 4.11 2.780 18.60 1 1 4 2 Volvo47 ## 7 22.8 4 108.0 93 3.85 2.320 18.61 1 1 4 1 Datsun73 ## 8 32.4 4 78.7 66 4.08 2.200 19.47 1 1 4 1 Fiat31 ## 9 15.2 8 275.8 180 3.07 3.780 18.00 0 0 3 3 Merc175 ## 10 21.0 6 160.0 110 3.90 2.875 17.02 0 1 4 4 Mazda86 ## 11 15.0 8 301.0 335 3.54 3.570 14.60 0 1 5 8 Maserati168 ## 12 10.4 8 460.0 215 3.00 5.424 17.82 0 0 3 4 Lincoln219 ## 13 17.3 8 275.8 180 3.07 3.730 17.60 0 0 3 3 Merc172 ## 14 13.3 8 350.0 245 3.73 3.840 15.41 0 0 3 4 Camaro54 ## 15 19.2 6 167.6 123 3.92 3.440 18.30 1 0 4 4 Merc217 ## 16 19.2 8 400.0 175 3.08 3.845 17.05 0 0 3 2 Pontiac306 ## performance year ## 1 4.511628 2006 ## 2 9.358289 2018 ## 3 8.883249 1998 ## 4 16.708861 1984 ## 5 5.801105 1999 ## 6 5.093458 1998 ## 7 4.078947 2007 ## 8 2.037037 2015 ## 9 11.842105 1984 ## 10 5.238095 2000 ## 11 22.333333 2008 ## 12 20.673077 2009 ## 13 10.404624 1985 ## 14 18.421053 2004 ## 15 6.406250 2004 ## 16 9.114583 2003

```
#####################  #  #  #  Exercise  4  #  #  #  #####################  cars_full  <-  full_join(mtcars,  cars_table,  by  =  'ID');  cars_full
```

mpg cyl disp hp drat wt qsec vs am gear carb ID ## 1 21.0 6 160.0 110 3.90 2.620 16.46 0 1 4 4 Mazda185 ## 2 21.0 6 160.0 110 3.90 2.875 17.02 0 1 4 4 Mazda86 ## 3 22.8 4 108.0 93 3.85 2.320 18.61 1 1 4 1 Datsun73 ## 4 21.4 6 258.0 110 3.08 3.215 19.44 1 0 3 1 Hornet6 ## 5 18.7 8 360.0 175 3.15 3.440 17.02 0

0 3 2 Hornet350 ## 6 18.1 6 225.0 105 2.76 3.460 20.22 1 0 3 1 Valiant392 ## 7 14.3 8 360.0 245 3.21 3.570 15.84 0 0 3 4 Duster128 ## 8 24.4 4 146.7 62 3.69 3.190 20.00 1 0 4 2 Merc-150 ## 9 22.8 4 140.8 95 3.92 3.150 22.90 1 0 4 2 Merc259 ## 10 19.2 6 167.6 123 3.92 3.440 18.30 1 0 4 4 Merc217 ## 11 17.8 6 167.6 123 3.92 3.440 18.90 1 0 4 4 Merc336 ## 12 16.4 8 275.8 180 3.07 4.070 17.40 0 0 3 3 Merc259 ## 13 17.3 8 275.8 180 3.07 3.730 17.60 0 0 3 3 Merc172 ## 14 15.2 8 275.8 180 3.07 3.780 18.00 0 0 3 3 Merc175 ## 15 10.4 8 472.0 205 2.93 5.250 17.98 0 0 3 4 Cadillac109 ## 16 10.4 8 460.0 215 3.00 5.424 17.82 0 0 3 4 Lincoln219 ## 17 14.7 8 440.0 230 3.23 5.345 17.42 0 0 3 4 Chrysler92 ## 18 32.4 4 78.7 66 4.08 2.200 19.47 1 1 4 1 Fiat31 ## 19 30.4 4 75.7 52 4.93 1.615 18.52 1 1 4 2 Honda207 ## 20 33.9 4 71.1 65 4.22 1.835 19.90 1 1 4 1 Toyota268 ## 21 21.5 4 120.1 97 3.70 2.465 20.01 1 0 3 1 Toyota-47 ## 22 15.5 8 318.0 150 2.76 3.520 16.87 0 0 3 2 Dodge212 ## 23 15.2 8 304.0 150 3.15 3.435 17.30 0 0 3 2 AMC148 ## 24 13.3 8 350.0 245 3.73 3.840 15.41 0 0 3 4 Camaro54 ## 25 19.2 8 400.0 175 3.08 3.845 17.05 0 0 3 2 Pontiac306 ## 26 27.3 4 79.0 66 4.08 1.935 18.90 1 1 4 1 Fiat83 ## 27 26.0 4 120.3 91 4.43 2.140 16.70 0 1 5 2 Porsche289 ## 28 30.4 4 95.1 113 3.77 1.513 16.90 1 1 5 2 Lotus139 ## 29 15.8 8 351.0 264 4.22 3.170 14.50 0 1 5 4 Ford-111 ## 30 19.7 6 145.0 175 3.62 2.770 15.50 0 1 5 6 Ferrari269 ## 31 15.0 8 301.0 335 3.54 3.570 14.60 0 1 5 8 Maserati168 ## 32 21.4 4 121.0 109 4.11 2.780 18.60 1 1 4 2 Volvo47 ## performance year ## 1 NA NA ## 2 5.238095 2000 ## 3 4.078947 2007 ## 4 NA NA ## 5 9.358289 2018 ## 6 5.801105 1999 ## 7 NA NA ## 8 NA NA ## 9 NA NA ## 10 6.406250 2004 ## 11 NA NA ## 12 NA NA ## 13 10.404624 1985 ## 14 11.842105 1984 ## 15 NA NA ## 16 20.673077 2009 ## 17 NA NA ## 18 2.037037 2015 ## 19 NA NA ## 20 NA NA ## 21 4.511628 2006 ## 22 NA NA ## 23 NA NA ## 24 18.421053 2004 ## 25 9.114583 2003 ## 26 NA NA ## 27 NA NA ## 28 NA NA ## 29 16.708861 1984 ## 30 8.883249 1998 ## 31 22.333333 2008 ## 32 5.093458 1998

```
##################### # # # Exercise 5 # # # ##################### cars_semi <- semi_join(mtcars, cars_table, by = 'ID'); cars_semi
```

```
## mpg cyl disp hp drat wt qsec vs am gear carb ID ## 1 21.5 4
```

120.1 97 3.70 2.465 20.01 1 0 3 1 Toyota-47 ## 2 18.7 8 360.0 175 3.15 3.440 17.02 0 0 3 2 Hornet350 ## 3 19.7 6 145.0 175 3.62 2.770 15.50 0 1 5 6 Ferrari269 ## 4 15.8 8 351.0 264 4.22 3.170 14.50 0 1 5 4 Ford-111 ## 5 18.1 6 225.0 105 2.76 3.460 20.22 1 0 3 1 Valiant392 ## 6 21.4 4 121.0 109 4.11 2.780 18.60 1 1 4 2 Volvo47 ## 7 22.8 4 108.0 93 3.85 2.320 18.61 1 1 4 1 Datsun73 ## 8 32.4 4 78.7 66 4.08 2.200 19.47 1 1 4 1 Fiat31 ## 9 15.2 8 275.8 180 3.07 3.780 18.00 0 0 3 3 Merc175 ## 10 21.0 6 160.0 110 3.90 2.875 17.02 0 1 4 4 Mazda86 ## 11 15.0 8 301.0 335 3.54 3.570 14.60 0 1 5 8 Maserati168 ## 12 10.4 8 460.0 215 3.00 5.424 17.82 0 0 3 4 Lincoln219 ## 13 17.3 8 275.8 180 3.07 3.730 17.60 0 0 3 3 Merc172 ## 14 13.3 8 350.0 245 3.73 3.840 15.41 0 0 3 4 Camaro54 ## 15 19.2 6 167.6 123 3.92 3.440 18.30 1 0 4 4 Merc217 ## 16 19.2 8 400.0 175 3.08 3.845 17.05 0 0 3 2 Pontiac306

```
##################### # # # Exercise 6 # # # ##################### cars_anti <- anti_join(mtcars, cars_table, by = 'ID'); cars_anti
```

mpg cyl disp hp drat wt qsec vs am gear carb ID ## 1 30.4 4 95.1 113 3.77 1.513 16.90 1 1 5 2 Lotus139 ## 2 26.0 4 120.3 91 4.43 2.140 16.70 0 1 5 2 Porsche289 ## 3 27.3 4 79.0 66 4.08 1.935 18.90 1 1 4 1 Fiat83 ## 4 15.2 8 304.0 150 3.15 3.435 17.30 0 0 3 2 AMC148 ## 5 15.5 8 318.0 150 2.76 3.520 16.87 0 0 3 2 Dodge212 ## 6 33.9 4 71.1 65 4.22 1.835 19.90 1 1 4 1 Toyota268 ## 7 30.4 4 75.7 52 4.93 1.615 18.52 1 1 4 2 Honda207 ## 8 14.7 8 440.0 230 3.23 5.345 17.42 0 0 3 4 Chrysler92 ## 9 10.4 8 472.0 205 2.93 5.250 17.98 0 0 3 4 Cadillac109 ## 10 17.8 6 167.6 123 3.92 3.440 18.90 1 0 4 4 Merc336 ## 11 22.8 4 140.8 95 3.92 3.150 22.90 1 0 4 2 Merc259 ## 12 16.4 8 275.8 180 3.07 4.070 17.40 0 0 3 3 Merc259 ## 13 24.4 4 146.7 62 3.69 3.190 20.00 1 0 4 2 Merc-150 ## 14 14.3 8 360.0 245 3.21 3.570 15.84 0 0 3 4 Duster128 ## 15 21.4 6 258.0 110 3.08 3.215 19.44 1 0 3 1 Hornet6 ## 16 21.0 6 160.0 110 3.90 2.620 16.46 0 1 4 4 Mazda185

```
##################### # # # Exercise 7 # # # ##################### cars <- mtcars %>% sample_frac(size = 0.5, replace = FALSE) cars_inter <- intersect(mtcars, cars); cars_inter
```

mpg cyl disp hp drat wt qsec vs am gear carb ID ## 1 27.3 4 79.0 66 4.08 1.935 18.90 1 1 4 1 Fiat83 ## 2 10.4 8 472.0 205 2.93 5.250 17.98 0 0 3 4 Cadillac109 ## 3 13.3 8 350.0 245 3.73 3.840 15.41 0 0 3 4 Camaro54 ## 4 19.2 6 167.6 123 3.92 3.440 18.30 1 0 4 4 Merc217 ## 5 15.2 8 304.0 150 3.15 3.435 17.30 0 0 3 2 AMC148 ## 6 30.4 4 75.7 52 4.93 1.615 18.52 1 1 4 2 Honda207 ## 7 21.0 6 160.0 110 3.90 2.875 17.02 0 1 4 4 Mazda86 ## 8 21.5 4 120.1 97 3.70 2.465 20.01 1 0 3 1 Toyota-47 ## 9 32.4 4 78.7 66 4.08 2.200 19.47 1 1 4 1 Fiat31 ## 10 16.4 8 275.8 180 3.07 4.070 17.40 0 0 3 3 Merc259 ## 11 14.7 8 440.0 230 3.23 5.345 17.42 0 0 3 4 Chrysler92 ## 12 21.0 6 160.0 110 3.90 2.620 16.46 0 1 4 4 Mazda185 ## 13 18.7 8 360.0 175 3.15 3.440 17.02 0 0 3 2 Hornet350 ## 14 22.8 4 108.0 93 3.85 2.320 18.61 1 1 4 1 Datsun73 ## 15 18.1 6 225.0 105 2.76 3.460 20.22 1 0 3 1 Valiant392 ## 16 30.4 4 95.1 113 3.77 1.513 16.90 1 1 5 2 Lotus139

```
###################### # # # Exercise 8 # # # ###################### cars_union <- union(mtcars, cars); cars_union
```

mpg cyl disp hp drat wt qsec vs am gear carb ID ## 1 21.4 4 121.0 109 4.11 2.780 18.60 1 1 4 2 Volvo47 ## 2 15.0 8 301.0 335 3.54 3.570 14.60 0 1 5 8 Maserati168 ## 3 19.7 6 145.0 175 3.62 2.770 15.50 0 1 5 6 Ferrari269 ## 4 15.8 8 351.0 264 4.22 3.170 14.50 0 1 5 4 Ford-111 ## 5 30.4 4 95.1 113 3.77 1.513 16.90 1 1 5 2 Lotus139 ## 6 26.0 4 120.3 91 4.43 2.140 16.70 0 1 5 2 Porsche289 ## 7 27.3 4 79.0 66 4.08 1.935 18.90 1 1 4 1 Fiat83 ## 8 19.2 8 400.0 175 3.08 3.845 17.05 0 0 3 2 Pontiac306 ## 9 13.3 8 350.0 245 3.73 3.840 15.41 0 0 3 4 Camaro54 ## 10 15.2 8 304.0 150 3.15 3.435 17.30 0 0 3 2 AMC148 ## 11 15.5 8 318.0 150 2.76 3.520 16.87 0 0 3 2 Dodge212 ## 12 21.5 4 120.1 97 3.70 2.465 20.01 1 0 3 1 Toyota-47 ## 13 33.9 4 71.1 65 4.22 1.835 19.90 1 1 4 1 Toyota268 ## 14 30.4 4 75.7 52 4.93 1.615 18.52 1 1 4 2 Honda207 ## 15 32.4 4 78.7 66 4.08 2.200 19.47 1 1 4 1 Fiat31 ## 16 14.7 8 440.0 230 3.23 5.345 17.42 0 0 3 4 Chrysler92 ## 17 10.4 8 460.0 215 3.00 5.424 17.82 0 0 3 4 Lincoln219 ## 18 10.4 8 472.0 205 2.93 5.250 17.98 0 0 3 4 Cadillac109 ## 19 15.2 8 275.8 180 3.07 3.780 18.00 0 0 3 3 Merc175 ## 20 17.3 8 275.8 180 3.07 3.730 17.60 0 0 3 3 Merc172 ## 21 16.4 8 275.8

180 3.07 4.070 17.40 0 0 3 3 Merc259 ## 22 17.8 6 167.6 123 3.92 3.440 18.90 1 0 4 4 Merc336 ## 23 19.2 6 167.6 123 3.92 3.440 18.30 1 0 4 4 Merc217 ## 24 22.8 4 140.8 95 3.92 3.150 22.90 1 0 4 2 Merc259 ## 25 24.4 4 146.7 62 3.69 3.190 20.00 1 0 4 2 Merc-150 ## 26 14.3 8 360.0 245 3.21 3.570 15.84 0 0 3 4 Duster128 ## 27 18.1 6 225.0 105 2.76 3.460 20.22 1 0 3 1 Valiant392 ## 28 18.7 8 360.0 175 3.15 3.440 17.02 0 0 3 2 Hornet350 ## 29 21.4 6 258.0 110 3.08 3.215 19.44 1 0 3 1 Hornet6 ## 30 22.8 4 108.0 93 3.85 2.320 18.61 1 1 4 1 Datsun73 ## 31 21.0 6 160.0 110 3.90 2.875 17.02 0 1 4 4 Mazda86 ## 32 21.0 6 160.0 110 3.90 2.620 16.46 0 1 4 4 Mazda185

########## # # # Exercise 9 # # # ########## cars_dif <- setdiff(mtcars, cars); cars_dif

mpg cyl disp hp drat wt qsec vs am gear carb ID ## 1 21.4 6 258.0 110 3.08 3.215 19.44 1 0 3 1 Hornet6 ## 2 14.3 8 360.0 245 3.21 3.570 15.84 0 0 3 4 Duster128 ## 3 24.4 4 146.7 62 3.69 3.190 20.00 1 0 4 2 Merc-150 ## 4 22.8 4 140.8 95 3.92 3.150 22.90 1 0 4 2 Merc259 ## 5 17.8 6 167.6 123 3.92 3.440 18.90 1 0 4 4 Merc336 ## 6 17.3 8 275.8 180 3.07 3.730 17.60 0 0 3 3 Merc172 ## 7 15.2 8 275.8 180 3.07 3.780 18.00 0 0 3 3 Merc175 ## 8 10.4 8 460.0 215 3.00 5.424 17.82 0 0 3 4 Lincoln219 ## 9 33.9 4 71.1 65 4.22 1.835 19.90 1 1 4 1 Toyota268 ## 10 15.5 8 318.0 150 2.76 3.520 16.87 0 0 3 2 Dodge212 ## 11 19.2 8 400.0 175 3.08 3.845 17.05 0 0 3 2 Pontiac306 ## 12 26.0 4 120.3 91 4.43 2.140 16.70 0 1 5 2 Porsche289 ## 13 15.8 8 351.0 264 4.22 3.170 14.50 0 1 5 4 Ford-111 ## 14 19.7 6 145.0 175 3.62 2.770 15.50 0 1 5 6 Ferrari269 ## 15 15.0 8 301.0 335 3.54 3.570 14.60 0 1 5 8 Maserati168 ## 16 21.4 4 121.0 109 4.11 2.780 18.60 1 1 4 2 Volvo47

########## # # # Exercise 10 # # # ########## car_rows <- bind_rows(mtcars, cars); car_rows

mpg cyl disp hp drat wt qsec vs am gear carb ID ## 1 21.0 6

160.0 110 3.90 2.620 16.46 0 1 4 4 Mazda185 ## 2 21.0 6 160.0 110 3.90 2.875 17.02 0 1 4 4 Mazda86 ## 3 22.8 4 108.0 93 3.85 2.320 18.61 1 1 4 1 Datsun73 ## 4 21.4 6 258.0 110 3.08 3.215 19.44 1 0 3 1 Hornet6 ## 5 18.7 8 360.0 175 3.15 3.440 17.02 0 0 3 2 Hornet350 ## 6 18.1 6 225.0 105 2.76 3.460 20.22 1 0 3 1 Valiant392 ## 7 14.3 8 360.0 245 3.21 3.570 15.84 0 0 3 4 Duster128 ## 8 24.4 4 146.7 62 3.69 3.190 20.00 1 0 4 2 Merc-150 ## 9 22.8 4 140.8 95 3.92 3.150 22.90 1 0 4 2 Merc259 ## 10 19.2 6 167.6 123 3.92 3.440 18.30 1 0 4 4 Merc217 ## 11 17.8 6 167.6 123 3.92 3.440 18.90 1 0 4 4 Merc336 ## 12 16.4 8 275.8 180 3.07 4.070 17.40 0 0 3 3 Merc259 ## 13 17.3 8 275.8 180 3.07 3.730 17.60 0 0 3 3 Merc172 ## 14 15.2 8 275.8 180 3.07 3.780 18.00 0 0 3 3 Merc175 ## 15 10.4 8 472.0 205 2.93 5.250 17.98 0 0 3 4 Cadillac109 ## 16 10.4 8 460.0 215 3.00 5.424 17.82 0 0 3 4 Lincoln219 ## 17 14.7 8 440.0 230 3.23 5.345 17.42 0 0 3 4 Chrysler92 ## 18 32.4 4 78.7 66 4.08 2.200 19.47 1 1 4 1 Fiat31 ## 19 30.4 4 75.7 52 4.93 1.615 18.52 1 1 4 2 Honda207 ## 20 33.9 4 71.1 65 4.22 1.835 19.90 1 1 4 1 Toyota268 ## 21 21.5 4 120.1 97 3.70 2.465 20.01 1 0 3 1 Toyota-47 ## 22 15.5 8 318.0 150 2.76 3.520 16.87 0 0 3 2 Dodge212 ## 23 15.2 8 304.0 150 3.15 3.435 17.30 0 0 3 2 AMC148 ## 24 13.3 8 350.0 245 3.73 3.840 15.41 0 0 3 4 Camaro54 ## 25 19.2 8 400.0 175 3.08 3.845 17.05 0 0 3 2 Pontiac306 ## 26 27.3 4 79.0 66 4.08 1.935 18.90 1 1 4 1 Fiat83 ## 27 26.0 4 120.3 91 4.43 2.140 16.70 0 1 5 2 Porsche289 ## 28 30.4 4 95.1 113 3.77 1.513 16.90 1 1 5 2 Lotus139 ## 29 15.8 8 351.0 264 4.22 3.170 14.50 0 1 5 4 Ford-111 ## 30 19.7 6 145.0 175 3.62 2.770 15.50 0 1 5 6 Ferrari269 ## 31 15.0 8 301.0 335 3.54 3.570 14.60 0 1 5 8 Maserati168 ## 32 21.4 4 121.0 109 4.11 2.780 18.60 1 1 4 2 Volvo47 ## 33 27.3 4 79.0 66 4.08 1.935 18.90 1 1 4 1 Fiat83 ## 34 10.4 8 472.0 205 2.93 5.250 17.98 0 0 3 4 Cadillac109 ## 35 13.3 8 350.0 245 3.73 3.840 15.41 0 0 3 4 Camaro54 ## 36 19.2 6 167.6 123 3.92 3.440 18.30 1 0 4 4 Merc217 ## 37 15.2 8 304.0 150 3.15 3.435 17.30 0 0 3 2 AMC148 ## 38 30.4 4 75.7 52 4.93 1.615 18.52 1 1 4 2 Honda207 ## 39 21.0 6 160.0 110 3.90 2.875 17.02 0 1 4 4 Mazda86 ## 40 21.5 4 120.1 97 3.70 2.465 20.01 1 0 3 1 Toyota-47 ## 41 32.4 4 78.7 66 4.08 2.200 19.47 1 1 4 1 Fiat31 ## 42 16.4 8 275.8 180 3.07 4.070 17.40 0 0 3 3 Merc259 ## 43 14.7 8 440.0 230 3.23 5.345 17.42 0 0 3 4 Chrysler92 ## 44 21.0 6 160.0 110 3.90 2.620 16.46 0 1 4 4

Mazda185 ## 45 18.7 8 360.0 175 3.15 3.440 17.02 0 0 3 2
Hornet350 ## 46 22.8 4 108.0 93 3.85 2.320 18.61 1 1 4 1
Datsun73 ## 47 18.1 6 225.0 105 2.76 3.460 20.22 1 0 3 1
Valiant392 ## 48 30.4 4 95.1 113 3.77 1.513 16.90 1 1 5 2
Lotus139