Data science for economists - Assignment 1

Dan Boguslavsky and Anna Demidov

5/03/2020

Required packages installation:

```
#install.packages('magrittr')
#install.packages("microbenchmark")
#install.packages('data.table')
#install.packages('ggplot2')
#install.packages("reshape2")
#install.packages('lubridate')
#install.packages('stringr')
#install.packages('chron')
#install.packages("ggplot2")
#install.packages("corrplot")
#install.packages("leaflet")
#install.packages("rnaturalearthdata")
#install.packages('dplyr')
```

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

#Question 1 #a.

```
my_aggregation <- function(x, is.truncated = FALSE){
  if(is.truncated){
    x <- x[x>=quantile(x,0.05)&x<=quantile(x,0.95)] #Discard 5th quantile and 95th qu
antile from vector x.
    return(list("mean" = mean(x), "var" = var(x), "med" = median(x)))
  }#close-if
  list("mean" = mean(x), "var" = var(x), "med" = median(x))
}#close-func-"my_aggregation"
```

#b. We expect the aggregates to be very different when there is an extremely low or high value (or a few values) in the vector that is very different from most of the values. If those “extreme” values are “balanced”, meaning we have them on opposite sides, for example, a vector with a mean of 10 with most of the values in it, around its mean, and two values of 1,000,000 and (-1,000,000). In this case, the mean should not be very much changed, although, the variance will decrease greatly. On the other hand, if we have a vector that its variance is large enough because its values are “spreaded”, but imbalanced, we are likely to see the variance decreasing slightly, but the mean changed radically. The robustness of the median is based on the fact that we discard the values from both sides with the same percentage.

```
set.seed(256)
dis<-rlnorm(1000000,1,0.5)
my_aggregation(dis)
```

```
## $mean
## [1] 3.076621
##
## $var
## [1] 2.686372
##
## $med
## [1] 2.71703
```

```
my_aggregation(dis,is.truncated = TRUE)
```

```
## $mean
## [1] 2.932773
##
## $var
## [1] 1.343869
##
## $med
## [1] 2.71703
```

We can see that the mean has not changed much, but the variance had decreased by half, this is because we eliminated the large and small values on both sides somewhat eaqualy so the mean did not changed, but because the spead is smaller now, the variance is smaller. Also, we can see, as expected, that the median did not changed.

#c.

```
#Adjust the function to return the mean only:

my_aggregation_mean <- function(x, is.truncated = FALSE){
  if(is.truncated){
    x <- x[x>=quantile(x,0.05)&x<=quantile(x,0.95)]
    return(list("mean" = mean(x)))
  }#close-if
  list("mean" = mean(x))
}#close-func-"my_aggregation_mean"

#Now lets compare the run time:

library('microbenchmark')
microbenchmark(
my_aggregation_mean(dis, is.truncated = TRUE),
mean(dis,trim = 0.05),
times = 30 #Using 30 instead of default 100 just to save time.
)
```

expr <fctr>	time <dbl>
mean(dis, trim = 0.05)	30827467
my_aggregation_mean(dis, is.truncated = TRUE)	52164445
my_aggregation_mean(dis, is.truncated = TRUE)	56195073

expr <fctr>	time <dbl>
mean(dis, trim = 0.05)	30781908
my_aggregation_mean(dis, is.truncated = TRUE)	51868929
mean(dis, trim = 0.05)	34065485
my_aggregation_mean(dis, is.truncated = TRUE)	75344538
mean(dis, trim = 0.05)	30336050
mean(dis, trim = 0.05)	64030576
my_aggregation_mean(dis, is.truncated = TRUE)	43018132

1-10 of 60 rows   Previous **1** 2 3 4 5 6 Next

We can see that the R base function is much faster (almost twice as much) than ours. The reason is probably because R is using its own efficient algorithms to subset the data.

#Question 2

#a.

```
aq<-airquality
apply(aq,2,FUN = function(x) sum(is.na(x)))
```

```
##   Ozone Solar.R    Wind    Temp   Month     Day
##      37       7       0       0       0       0
```

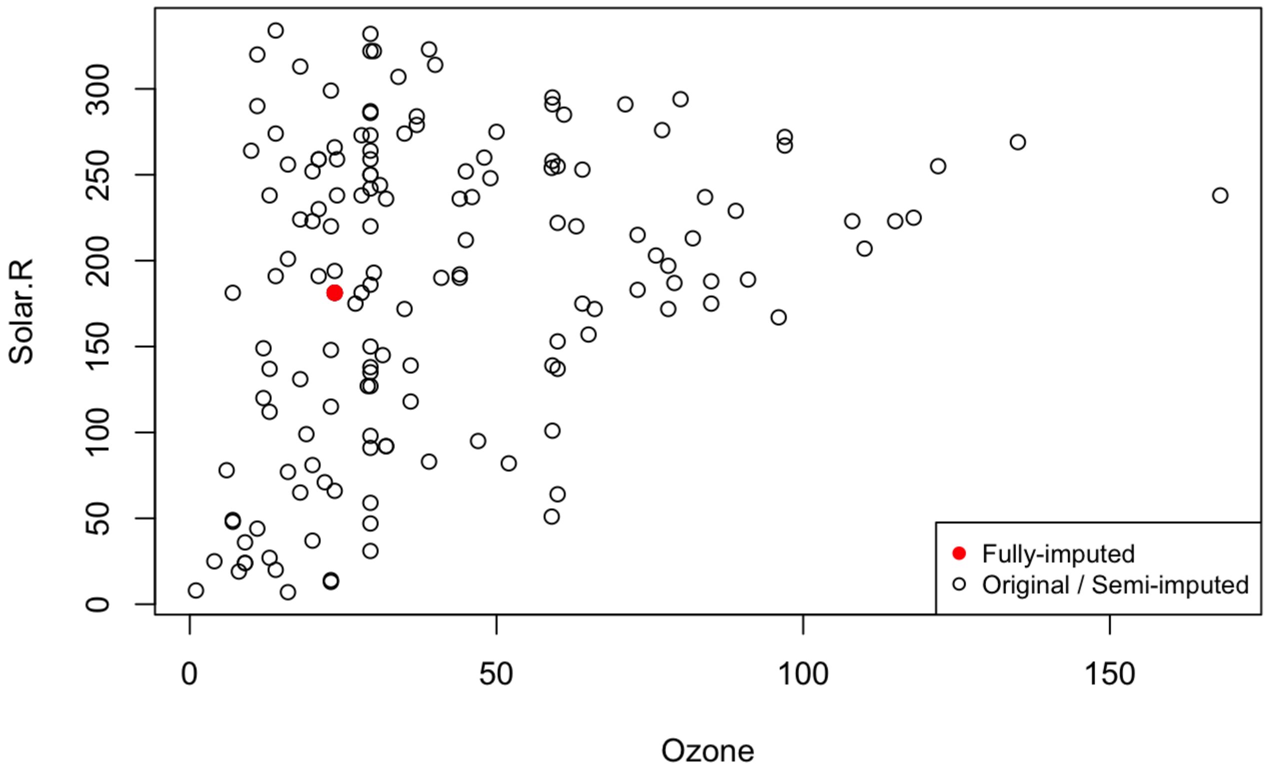
```
is.na_row <- apply(aq,1, FUN = function (x) anyNA(x))
```

#b.

```
library('data.table')
  airquality_imputed <- as.data.table(aq)
  airquality_imputed <-
    airquality_imputed[,lapply(.SD,FUN = function(x) ifelse(is.na(x),mean(x,na.rm = T
RUE),as.double(x))),by = Month]
```

#c.

```
plot(airquality_imputed$Ozone,airquality_imputed$Solar.R, xlab = "Ozone",ylab = "Sola
r.R")
imputed_points<-which(is.na(airquality$Ozone)&is.na(airquality$Solar.R))
points(airquality_imputed$Ozone[imputed_points],airquality_imputed$Solar.R[imputed_po
ints],col="red",pch = 19)
legend("bottomright", legend = c("Fully-imputed","Original / Semi-imputed"),col=c("re
d", "black"), pch=c(19,1), cex=0.8)
```



#Question 3

```
library('ggplot2')
data("diamonds")
```

#a.

```
dim(diamonds)
```

```
## [1] 53940    10
```

```
class(diamonds)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

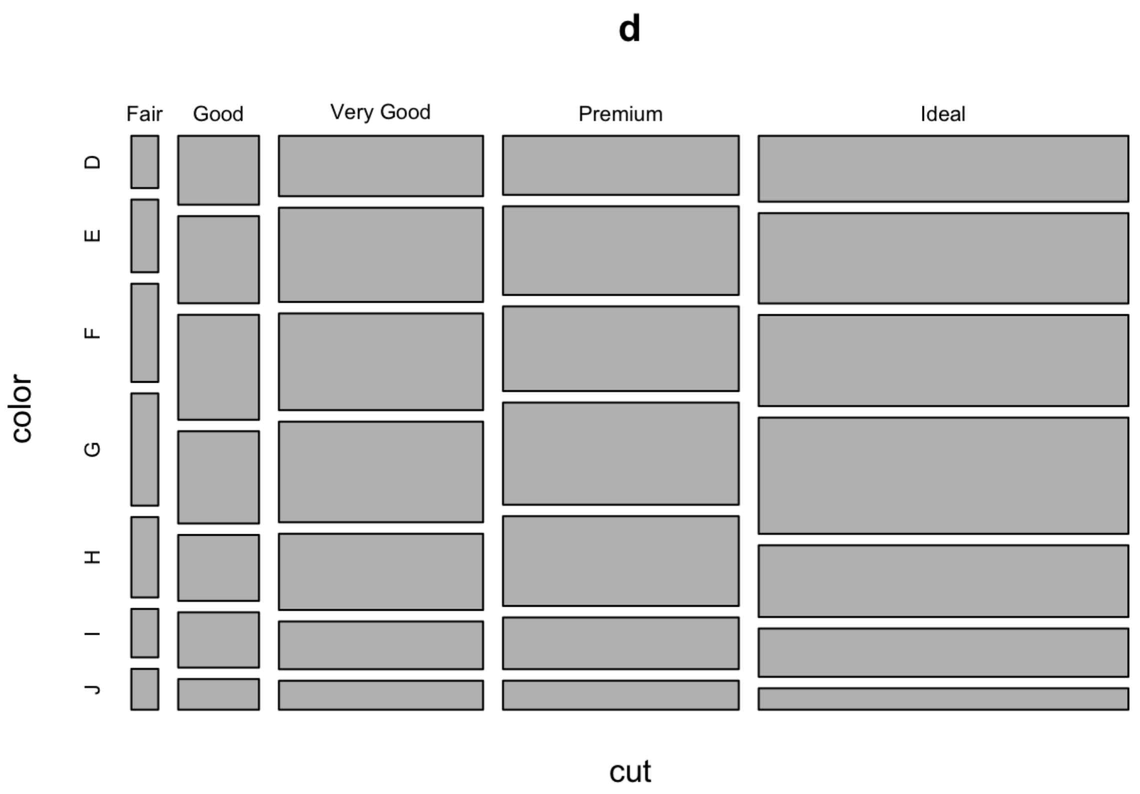
The class is - data frame, made out of a “tbl” class. The dimensions are 53940 rows over 10 columns.

#b.

```
library('magrittr')
library('data.table')
set.seed(256)
d <- diamonds[sample(nrow(diamonds),nrow(diamonds)*0.25),] %>% as.data.table()
```

#c.

```
mosaicplot(~cut+color,data = d)
```



In this plot we can observe the distribution of the entire data, based on the color and cut variables. For example, it is obvious that the “Ideal” cut is the most frequent one and the “Fair” cut is the less frequent one. Now, we can also observe the distribution of the color in each cut level. For example - the “G” color seems to be the most frequent in the “Ideal” cut group, but the “J” color - the less frequent one in that group.

#d.

```
#1.
d[,"logp":=log10(d$price)]
#2.
d[,"v":=d$x*d$y*d$z]
#3.
median_depth<-median(d$depth)
d[,"cond1":=(((d$cut=="Ideal") + (d$depth<median_depth) + (d$clarity!="I1") + (d$colo
r%in%c("D","E","F","G")))==3)]
```

#e.

```
d[,list(Vmean = mean(v),Vvar = var(v), LOGPmean = mean(logp), LOGPvar = var(logp)),by
= cond1]
```

cond1 <lgl>	Vmean <dbl>	Vvar <dbl>	LOGPmean <dbl>	LOGPvar <dbl>
FALSE	137.7239	6458.754	3.410909	0.1978538
TRUE	117.4117	4746.235	3.336896	0.1870643

2 rows

#f.

```r
library('magrittr')
color_cut<-expand.grid(unique(d$cut),unique(d$color)) %>% as.data.table()
colnames(color_cut)[1]<-"cut"
colnames(color_cut)[2]<-"color"
set.seed(256)
color_cut[,"some_feature":=rnorm(nrow(color_cut))]
d<-merge(d,color_cut,by=c("cut","color"),all = TRUE)
#1.
d[,list(some_feature_mean = mean(some_feature)),by = "clarity"]
```

clarity <ord>	some_feature_mean <dbl>
VVS2	-0.15472180
SI2	0.17363973
VS2	0.06292046
SI1	0.23581161
I1	0.13641033
VS1	0.09330214
VVS1	-0.06126381
IF	-0.13478275

8 rows

```r
#2.
d[some_feature>1,list(PRICE_sd = sd(price), PRICE_iqr = IQR(price), PRICE_mad = mad(p
rice)),by = "cut"]
```

cut <ord>	PRICE_sd <dbl>	PRICE_iqr <dbl>	PRICE_mad <dbl>
Fair	3603.874	3637.75	1888.832
Very Good	4048.386	4295.00	2718.347
Premium	4311.896	4569.50	2756.153
Ideal	4264.889	5256.00	3074.912

4 rows

```r
#3. we are looking for those who make (cond1 or cond2) but not both.
d[((1<carat&2>carat)|(5000<price&10000>price))&(!((1<carat&2>carat)&(5000<price&10000
>price))),.N,by="color"]
```

color <ord>	N <int>
D	178
E	282

color	N
<ord>	<int>
F	380
G	392
H	377
I	225
J	141

7 rows

#g.

```
library('reshape2')
```

```
##
## Attaching package: 'reshape2'
```

```
## The following objects are masked from 'package:data.table':
##
##     dcast, melt
```

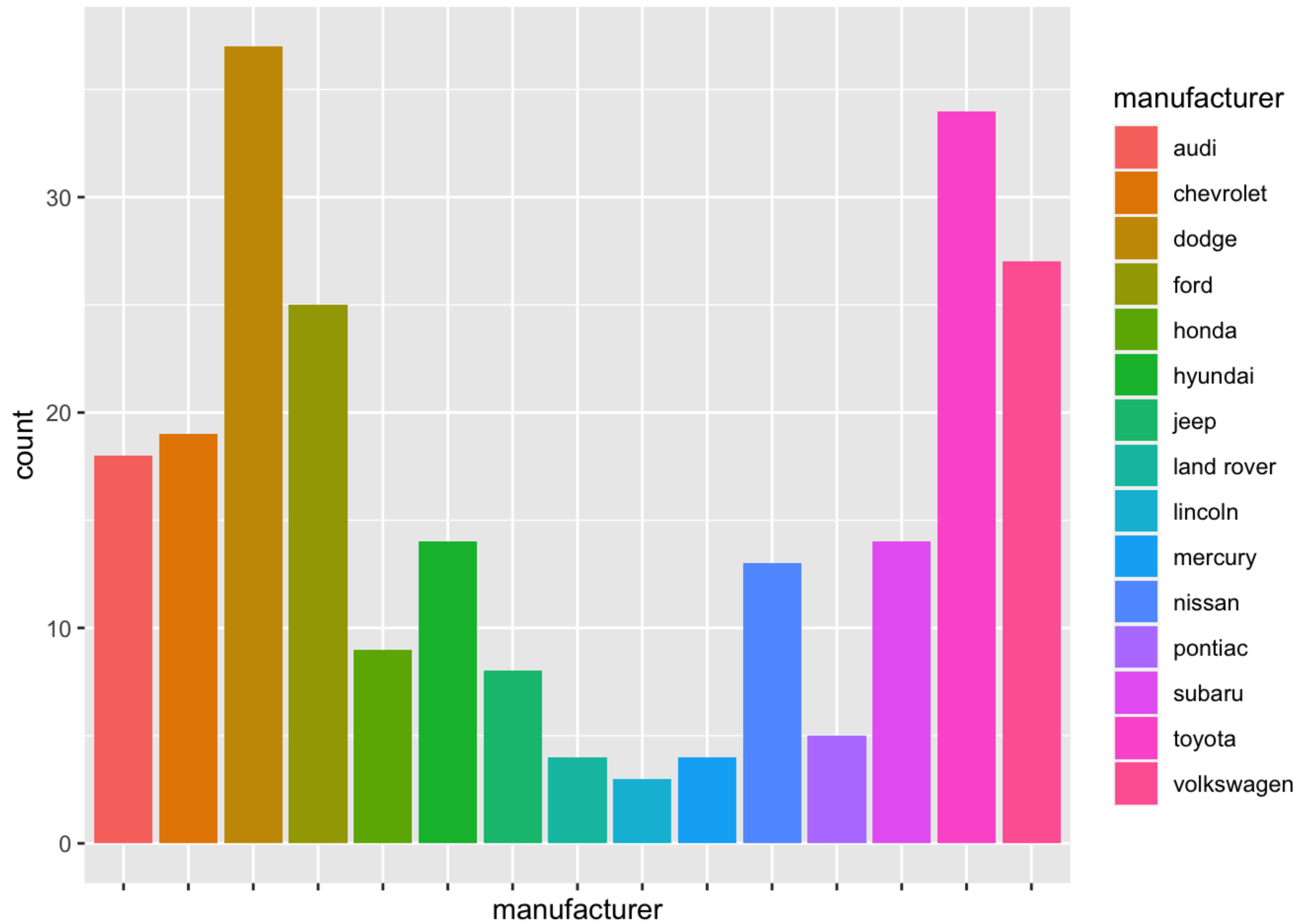
```
acast(d,cut~color,value.var = "price",mean)
```

```
##                  D        E        F        G        H        I        J
## Fair      3551.122 3387.158 3301.169 4211.125 5219.889 4069.526 4478.750
## Good      3121.198 3422.049 3831.166 4218.912 4336.910 5625.192 4528.819
## Very Good 3555.568 3230.214 3751.541 4198.918 4902.305 5455.137 5626.727
## Premium   3628.027 3627.814 4589.186 4545.345 5039.692 5693.104 6183.241
## Ideal     2529.812 2534.524 3376.194 3681.819 3947.424 4248.875 4437.435
```

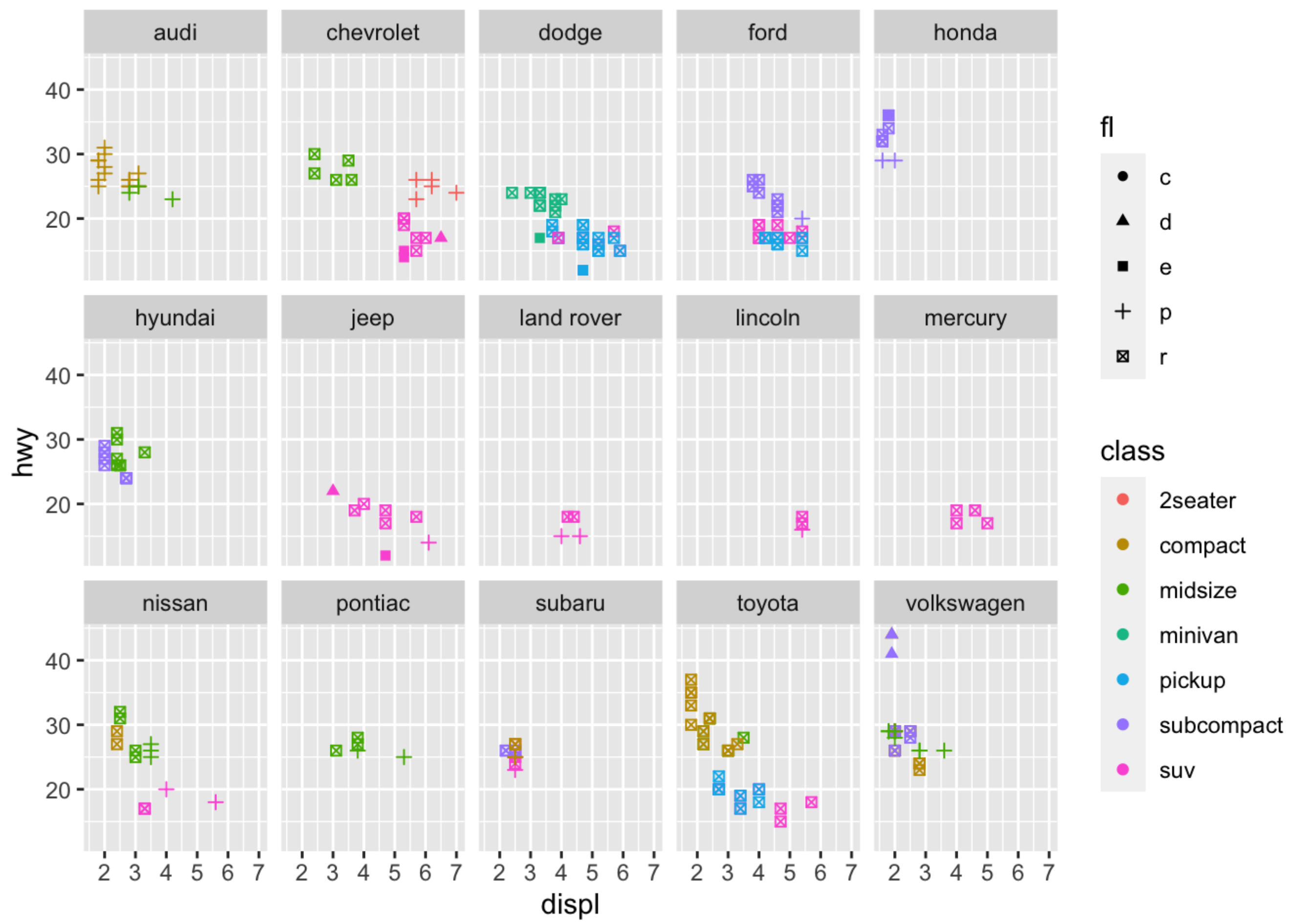
#Question 4

```
library('ggplot2')
data("mpg")
mpg$manufacturer<-factor(mpg$manufacturer)
mpg$class<-factor(mpg$class)
mpg$year<-factor(mpg$year)
ggplot(mpg,aes(x=manufacturer,fill=manufacturer))+geom_histogram(stat = "count") + sc
ale_x_discrete(labels = NULL)
```

```
## Warning: Ignoring unknown parameters: binwidth, bins, pad
```

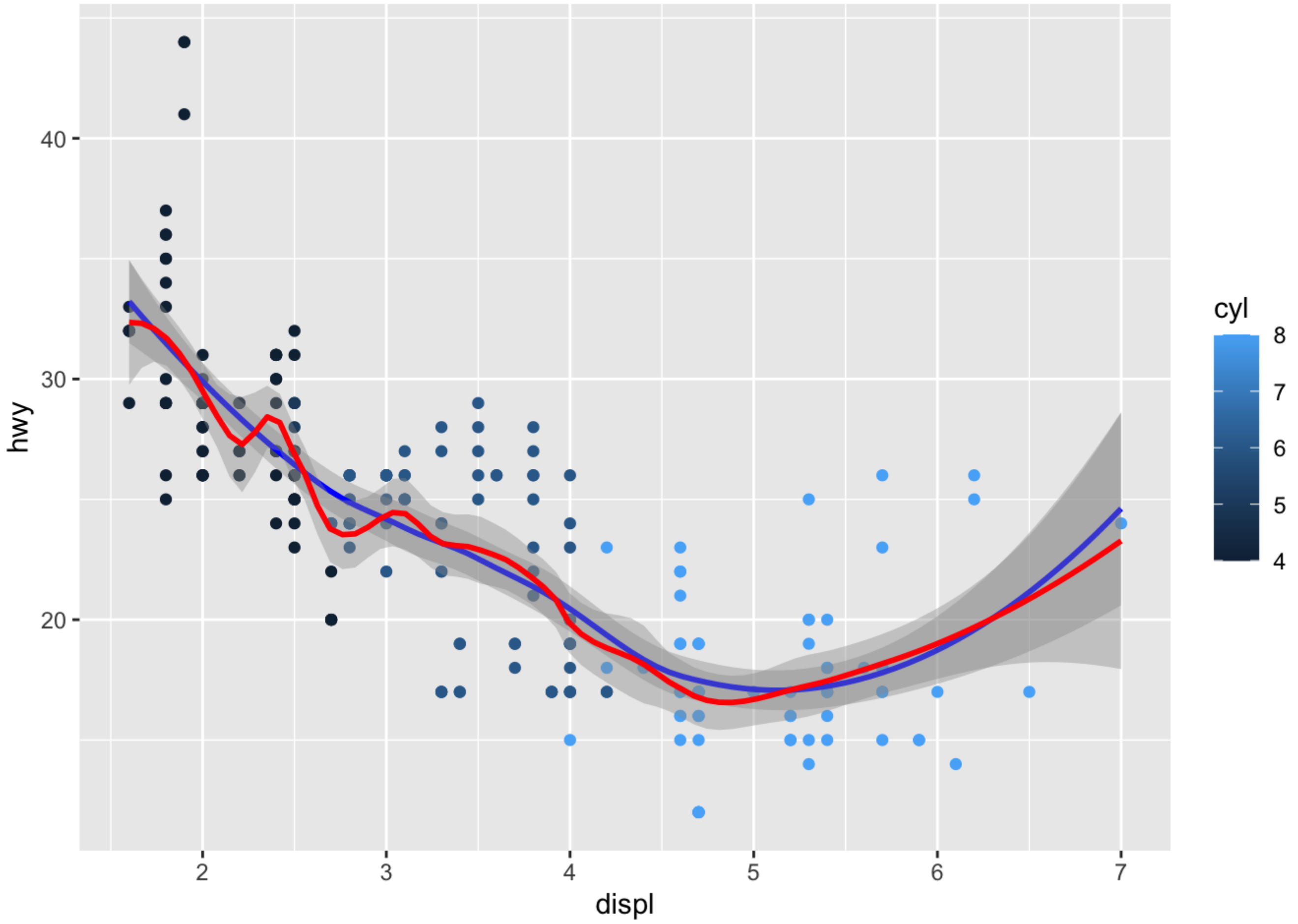



```
ggplot(mpg,aes(x=displ,y=hwy,color=class))+geom_point(aes(shape = fl)) + facet_wrap(~
manufacturer, ncol = 5)
```

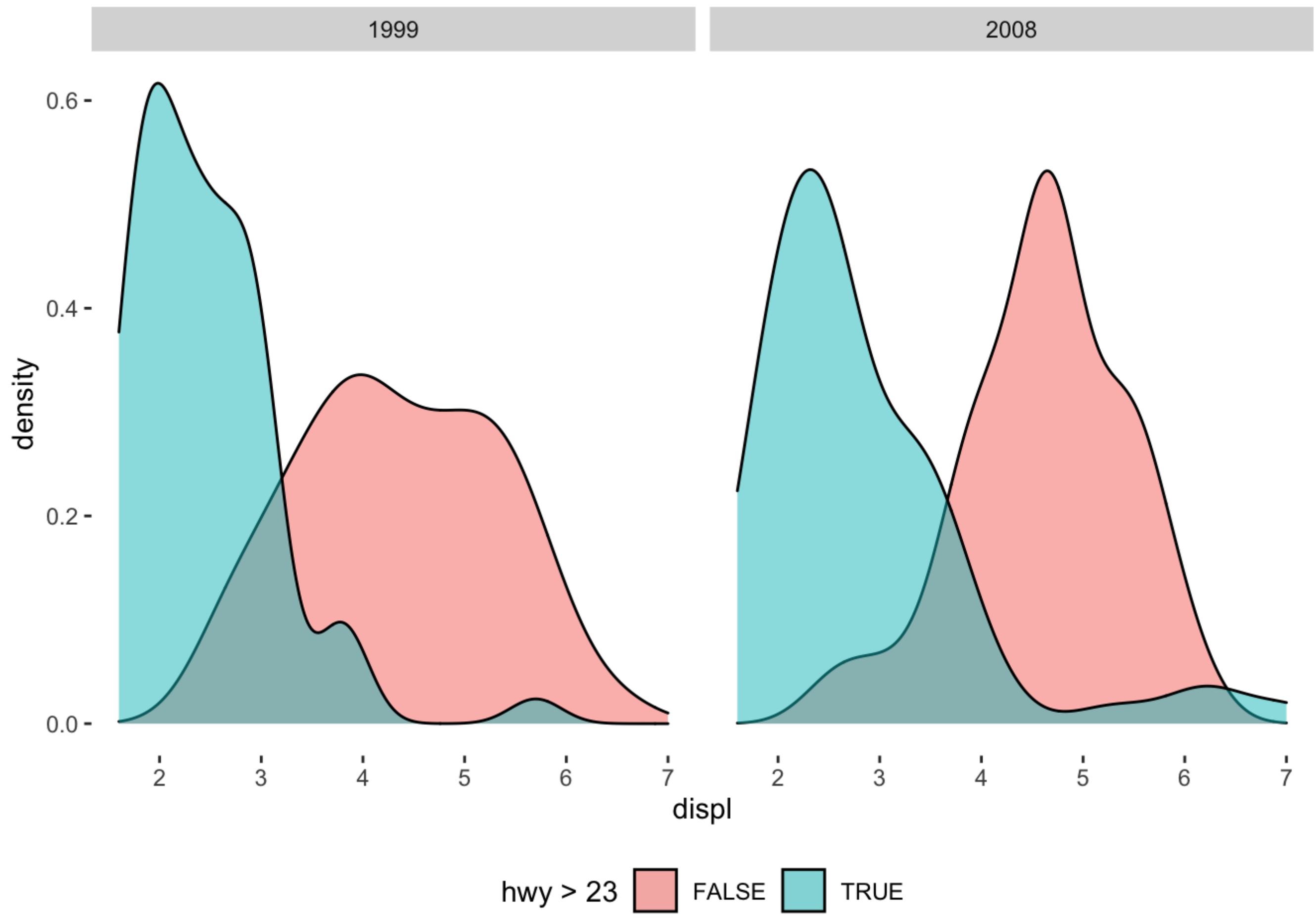


```
ggplot(mpg,aes(x=displ,y=hwy,color=cyl)) + geom_point() +
          geom_smooth(span=0.7,colour = "blue",level=0.95) + geom_smooth(span=0.3,col
our = "red",level=0.95)
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
ggplot(mpg,aes(x=displ)) + geom_density(aes(group = hwy>23,fill = hwy>23),alpha=0.5)
 +
          facet_wrap(vars(year)) +
          theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank
(),panel.background = element_blank(),
          legend.position="bottom")
```



#Question 5

#a.

```
airq<-airquality %>% as.data.table()
stock<-EuStockMarkets %>% as.data.table()
```

#b.

```
library('data.table')
airq[,"date":=as.Date(paste(airq$Day,airq$Month,"2019",sep = "-"),format = c("%d-%m-%
Y"))]
stock[,"date":=seq.Date(from = as.Date("2019-01-01",format = c("%Y-%m-%d")),by = "da
y",length.out = nrow(stock))]
```

#c.

```
setkey(airq,"date")
setkey(stock,"date")
stock<-stock[airq,,]
```

#d.

```
library('lubridate')
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:data.table':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
difference<-stock$date[stock$CAC==(min(stock$CAC))] %--%stock$date[stock$CAC==(max(st
ock$CAC))]
#The time period is:
as.period(difference, unit = 'month') #in months
```

```
## [1] "3m 8d 0H 0M 0S"
```

```
as.period(difference, unit = 'day') #in days
```

```
## [1] "100d 0H 0M 0S"
```

```
as.period(difference, unit = 'hour') #in hours
```

```
## [1] "2400H 0M 0S"
```

#e.

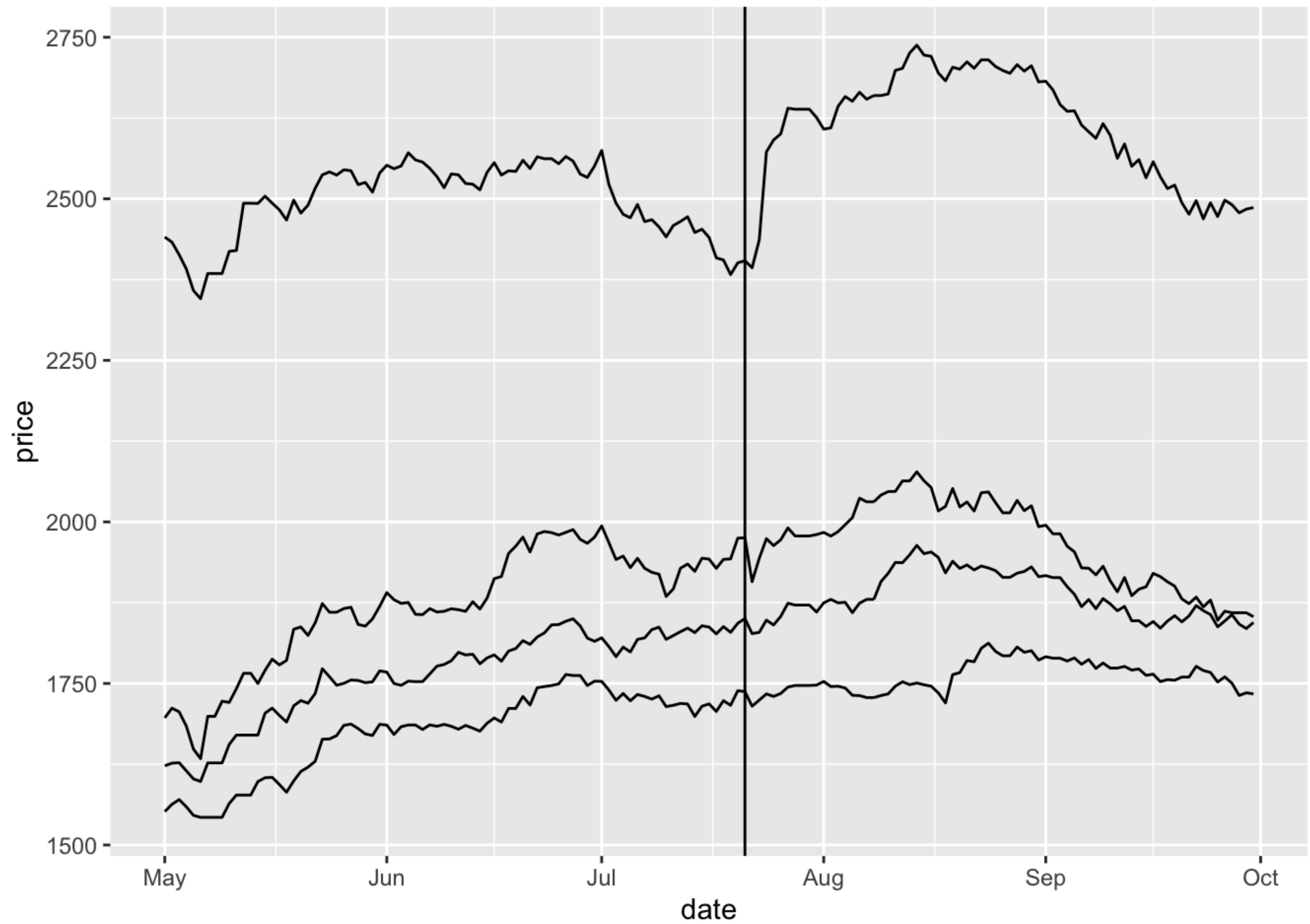
```
library('magrittr')
stock$Temp[stock$date %between% c(ymd(int_start(difference)),ymd(int_end(differenc
e)))] %>% mean()
```

```
## [1] 77.94059
```

#f.

```
library('ggplot2')

ggplot(data = stock,aes(x=date)) + geom_line(aes(date,CAC)) + geom_line(aes(date,DA
X)) + geom_line(aes(date,SMI)) +                         geom_line(aes(date,FTSE)) +
 ylab("price") +
      geom_vline(xintercept = as.numeric(stock[stock$Solar.R == min(stock$Solar.R,na.
rm = T),"date"]))
```



#g.

```
stock[,"week_num":=week(stock$date)]
stock[,list(CAC_mean = mean(CAC), TEMP_mean = mean(Temp)),by = "week_num"]
```

week_num <dbl>	CAC_mean <dbl>	TEMP_mean <dbl>
18	1680.167	66.16667
19	1730.671	66.14286
20	1791.914	63.28571
21	1856.557	60.85714
22	1863.329	74.57143
23	1863.200	84.85714
24	1882.429	82.14286
25	1970.486	74.14286
26	1980.243	78.57143
27	1939.986	85.14286

1-10 of 22 rows

Previous **1** 2 3 Next

```
##if only the MIN-MAX intervanl needed:
stock[stock$date %between% c(ymd(int_start(difference)),ymd(int_end(difference))),
      list(CAC_mean = mean(CAC), TEMP_mean = mean(Temp)),by = "week_num"]
```

week_num <dbl>	CAC_mean <dbl>	TEMP_mean <dbl>
18	1633.600	66.00000
19	1730.671	66.14286
20	1791.914	63.28571
21	1856.557	60.85714
22	1863.329	74.57143
23	1863.200	84.85714
24	1882.429	82.14286
25	1970.486	74.14286
26	1980.243	78.57143
27	1939.986	85.14286

1-10 of 16 rows Previous **1** 2 Next

#Question 6

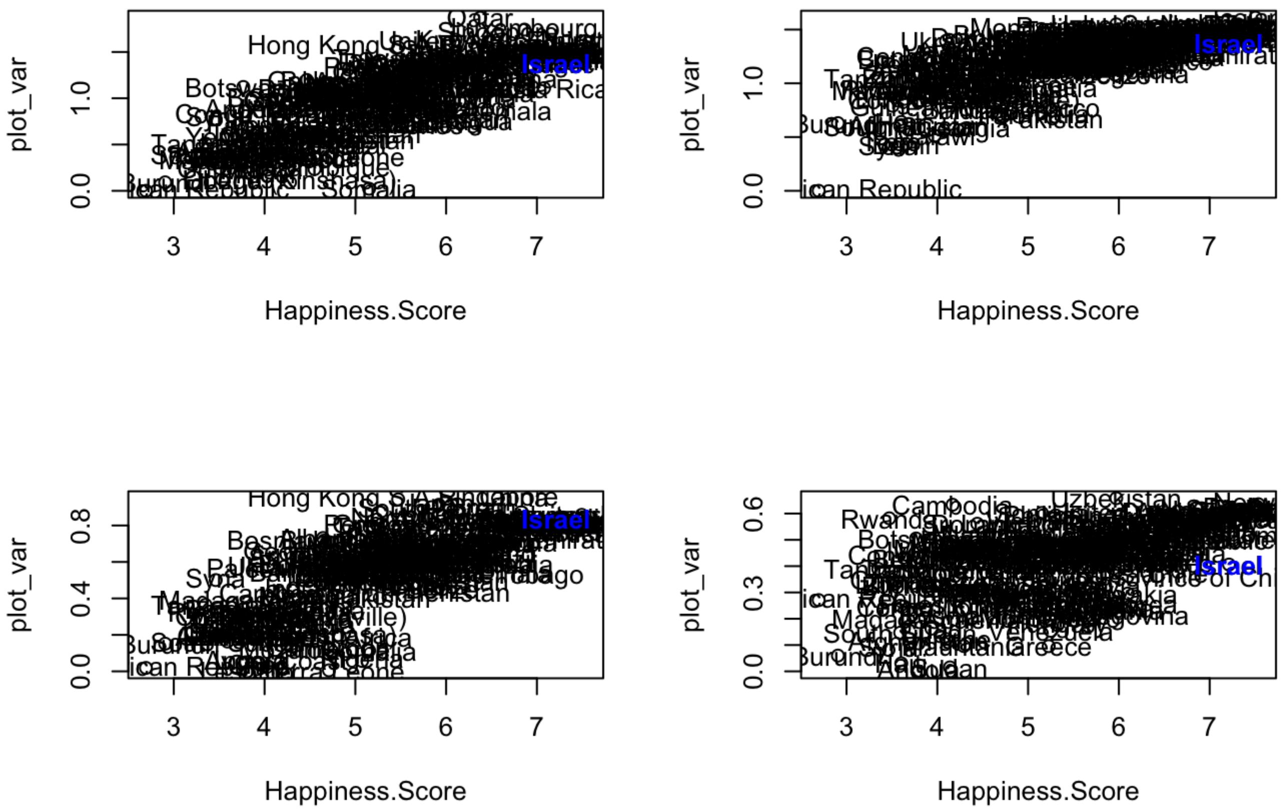
```
WHR2017<-read.csv("/Users/danboguslavsky/git/datascience/2017.csv")
```

#a.

```
#colnames(WHR2017)
for (i  in 1:NROW(colnames(WHR2017))){
  colnames(WHR2017)[i]<-gsub("..","-",colnames(WHR2017)[i],fixed = TRUE)
}#for_loop
```

#b.

```
attach(WHR2017)
to_plot<-list(`Economy-GDP.per.Capita.`,`Family`,`Health-Life.Expectancy.`,`Freedom`)
par(mfrow = c(2,2))
for(plot_var in to_plot){
  plot(x=`Happiness.Score`,y=plot_var)
  graphics::text(`Happiness.Score`,plot_var,Country)
  graphics::text(`Happiness.Score`,plot_var,ifelse(Country=='Israel','Israel',""),fon
t = 2, col="blue")
}#Close_for_loop
```



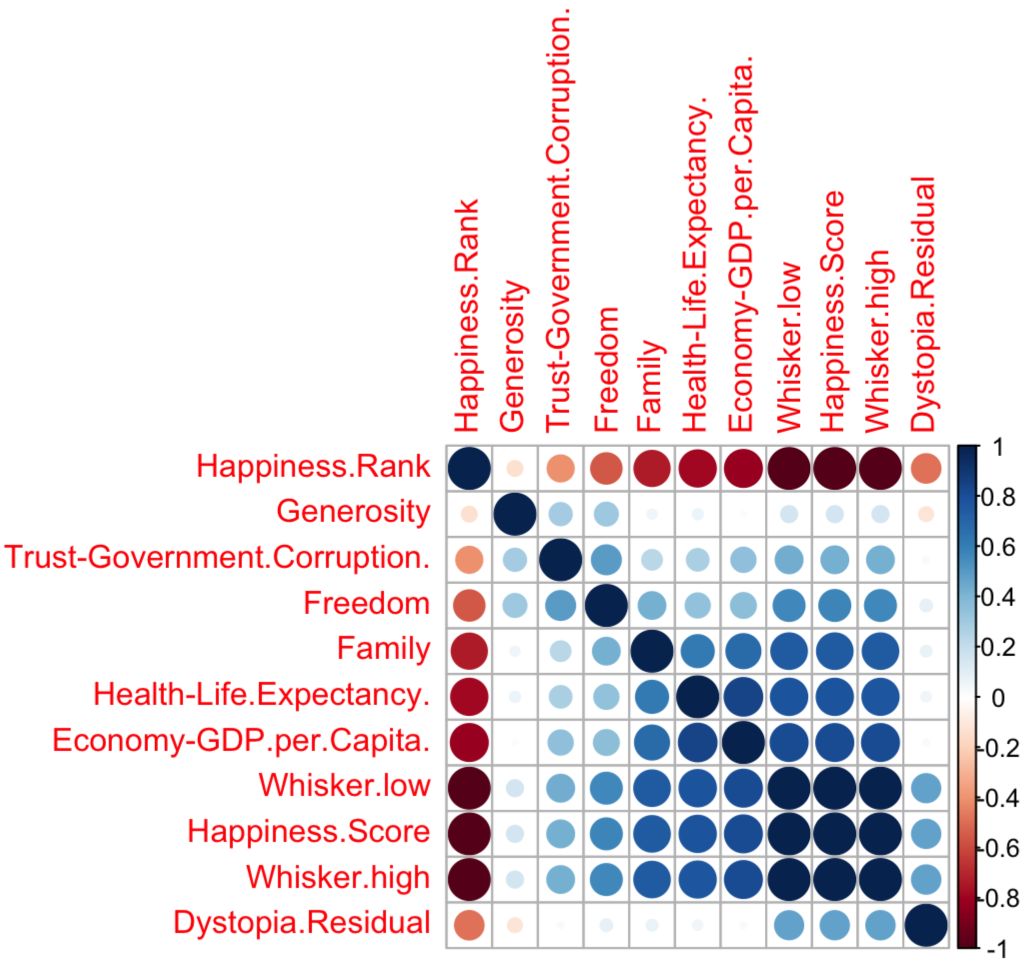

```
#graphics::text(Country)
detach(WHR2017)
```

#c.

```
library('corrplot')
```

```
## corrplot 0.84 loaded
```

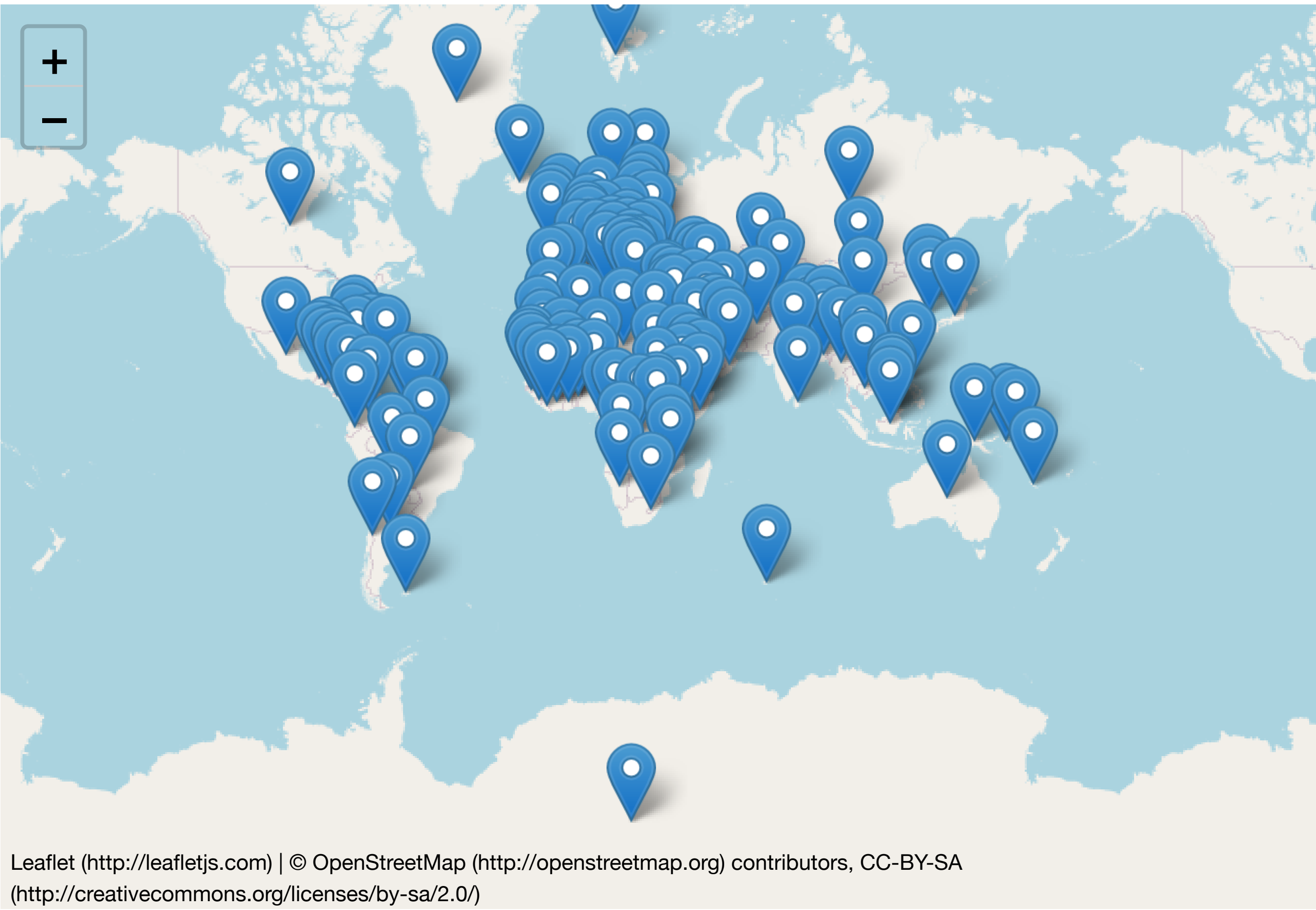
```
WHR2017_numerics<-WHR2017[,-c(1)]
corrplot(cor(WHR2017_numerics),order = "AOE")
```



We can see that the “Whisker.high”,“Whisker.low” and “Happiness.Score” are very highly and positivly correlated with each-other. “Happiness.Rank” is also highly correlated with the previous three but negativly. This makes very much sense as high “Happiness.Score” indicates lower rank - meaning higher position. “Economy-GDP.per.Capita.”,“Health-Life.Expectancy” and “Family” are also fairly correlated with the other four.

#d.Bonus.

```
#library("rnaturalearthdata")
library('leaflet')
library('magrittr')
contries_lng_lat<-data.frame(NA,NA,NA)
countries_data<-rnaturalearthdata::map_units110
country_num <- which(WHR2017$Country %in% countries_data$name_long)
for (i in country_num){
  new<-data.frame(NA,NA,NA)
  new[1,1]<-countries_data$name_long[i]
  new[1,2]<-countries_data@polygons[[i]]@labpt[1]
  new[1,3]<-countries_data@polygons[[i]]@labpt[2]
  contries_lng_lat<-rbind(contries_lng_lat,new)
}#Close_for
colnames(contries_lng_lat)<-c("Country","lng","lat")
contries_lng_lat<-contries_lng_lat[-1,]
contries_lng_lat<-merge(contries_lng_lat,subset(WHR2017,select=c("Country","Happines
s.Score")),by = "Country",all.x = T)
contries_lng_lat$Name_Score<-paste(contries_lng_lat$Country," - "," Score: ",round(co
ntries_lng_lat$Happiness.Score,3),sep = " ")
leaflet() %>% addTiles() %>% addMarkers(lng = contries_lng_lat$lng,lat = contries_lng
_lat$lat, label = contries_lng_lat$Name_Score)
```



#Question 7

```
library('data.table')
autos<-fread(file = "/Users/danboguslavsky/git/datascience/autos.csv", encoding = "La
tin-1")
```

```
## Warning in fread(file = "/Users/danboguslavsky/git/
## datascience/autos.csv", : Found and resolved improper
## quoting out-of-sample. First healed line 5263: <<2016-03-29
## 16:46:46,"_SPARDOSE"______Polo_1_4___6N1___60PS___5Tuerer____FESTPREIS,privat,Ange
bot,
## 500,control,limousine,1999,manuell,60,polo,150000,12,benzin,volkswagen,ja,
## 2016-03-25 00:00:00,0,59581,2016-03-30 11:46:58>>. If the fields are not quoted
## (e.g. field separator does not appear within any field), try quote="" to avoid
## this warning.
```

#a.

```
library('magrittr')
grep("Mazda",autos$name,ignore.case = TRUE) %>% length()
```

```
## [1] 5463
```

```
mazda <- autos[grep("Mazda",autos$name,ignore.case = TRUE),] %>% as.data.table()
```

#b.

```
mazda[,"is_3":= grepl("3",mazda$name)]
```

#c.

```
library('lubridate')
mazda[,list(Created_to_Seen_time =mean(difftime(as.POSIXct(lastSeen),as.POSIXct(dateC
reated),units = "hours")),
            Num_of_obs = .N, Diesel_sahre = (sum(fuelType == "diesel"))/.N),by = "is_
3"]
```

is_3	Created_to_Seen_time	Num_of_obs	Diesel_sahre
<lgl>	<time>	<int>	<dbl>
TRUE	206.3029 hours	1730	0.09710983
FALSE	215.8856 hours	3733	0.22769890

2 rows

#Question 8 #a.

```
zeros<-function(d){
  a<-matrix(0,d,d)
  a[c(1,d),] <-1
  a[,c(1,d)] <-1
  return(a)
}#close_function_"zeros"
```

#b.

```
same<-function(a,b){
  if(length(a)==length(b)){
    for (i in 1:length(a)) {
      if(a[i]!=b[i]){return(FALSE)}
    }#close_for
    return(TRUE) #no non identical values found -> the vectors are identical
  }#close_if
  return(FALSE) #not the same length -> not identical
}#close_function
```

#c.

```
library('stringr')
counter<-function(a,b){
 count = 0
 a<-str_split(a,"",simplify = T)
 for (char in a){
   if (char==b){count = count + 1}
 }#close for loop
 return(count)
}#close function
```

#d.

```
birthday <- function(birthday){
  birthday<-as.Date(birthday)
  print(weekdays(birthday))
  difference<-(Sys.Date()-birthday)
  print(difference)
  print(paste("Next birthday in: ",(ceiling(difference/365)-(difference/365))*365," d
ays",sep = ""))
}
```

#Question 9

```
library('ggplot2')
data("diamonds")
```

#a.

```
numeric_diamonds <- unlist(lapply(diamonds, is.numeric))
numeric_diamonds<-diamonds[,numeric_diamonds]
cor_mat<-matrix(NA,ncol(numeric_diamonds),ncol(numeric_diamonds))
for(i in 1:ncol(numeric_diamonds)){
  for(j in 1:ncol(numeric_diamonds)){
    cor_mat[i,j]<-(cor(numeric_diamonds[,i],numeric_diamonds[,j]))
  }#close_j_loop
}#close_i_loop
colnames(cor_mat)<-names(numeric_diamonds)
rownames(cor_mat)<-names(numeric_diamonds)
cor_mat
```

```
##            carat       depth      table      price           x           y
## carat 1.00000000  0.02822431  0.1816175  0.9215913  0.97509423  0.95172220
## depth 0.02822431  1.00000000 -0.2957785 -0.0106474 -0.02528925 -0.02934067
## table 0.18161755 -0.29577852  1.0000000  0.1271339  0.19534428  0.18376015
## price 0.92159130 -0.01064740  0.1271339  1.0000000  0.88443516  0.86542090
## x     0.97509423 -0.02528925  0.1953443  0.8844352  1.00000000  0.97470148
## y     0.95172220 -0.02934067  0.1837601  0.8654209  0.97470148  1.00000000
## z     0.95338738  0.09492388  0.1509287  0.8612494  0.97077180  0.95200572
##                z
## carat 0.95338738
## depth 0.09492388
## table 0.15092869
## price 0.86124944
## x     0.97077180
## y     0.95200572
## z     1.00000000
```

The importance of a correlation matrix in the context of date science is being expressed as it lets us see pattern between a large amount of variables. Thanks to it, we can show a very important information is a very simple and basic form.

#b. No, we cannot compute the Pearson correlation between ‘cut’ and ‘color’ as they are both “categorial” variables and cannot be used in Pearson’s correlation formula.

#c.

```r
library('magrittr')
library('dplyr')
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:lubridate':
##
##     intersect, setdiff, union
```

```
## The following objects are masked from 'package:data.table':
##
##     between, first, last
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library('data.table')
cut_by_color<-diamonds %>% group_by(cut, color) %>% summarise(n=n()) %>% dcast(color~
cut)
```

```
## Using n as value column: use value.var to override.
```

```r
cut_by_color<-as.data.table(cut_by_color)
cut_by_color[,.SD/sum(.SD),by = "color"]
```

color <ord>	Fair <dbl>	Good <dbl>	Very Good <dbl>	Premium <dbl>	Ideal <dbl>
D	0.02405904	0.09771218	0.2233210	0.2366052	0.4183026
E	0.02286414	0.09523323	0.2449730	0.2385424	0.3983873
F	0.03269755	0.09526305	0.2267868	0.2442884	0.4009642
G	0.02780730	0.07713425	0.2035955	0.2589444	0.4325186
H	0.03648844	0.08453757	0.2196532	0.2842004	0.3751204
I	0.03227591	0.09627444	0.2220583	0.2633714	0.3860199
J	0.04237892	0.10933048	0.2414530	0.2877493	0.3190883

7 rows

#d.

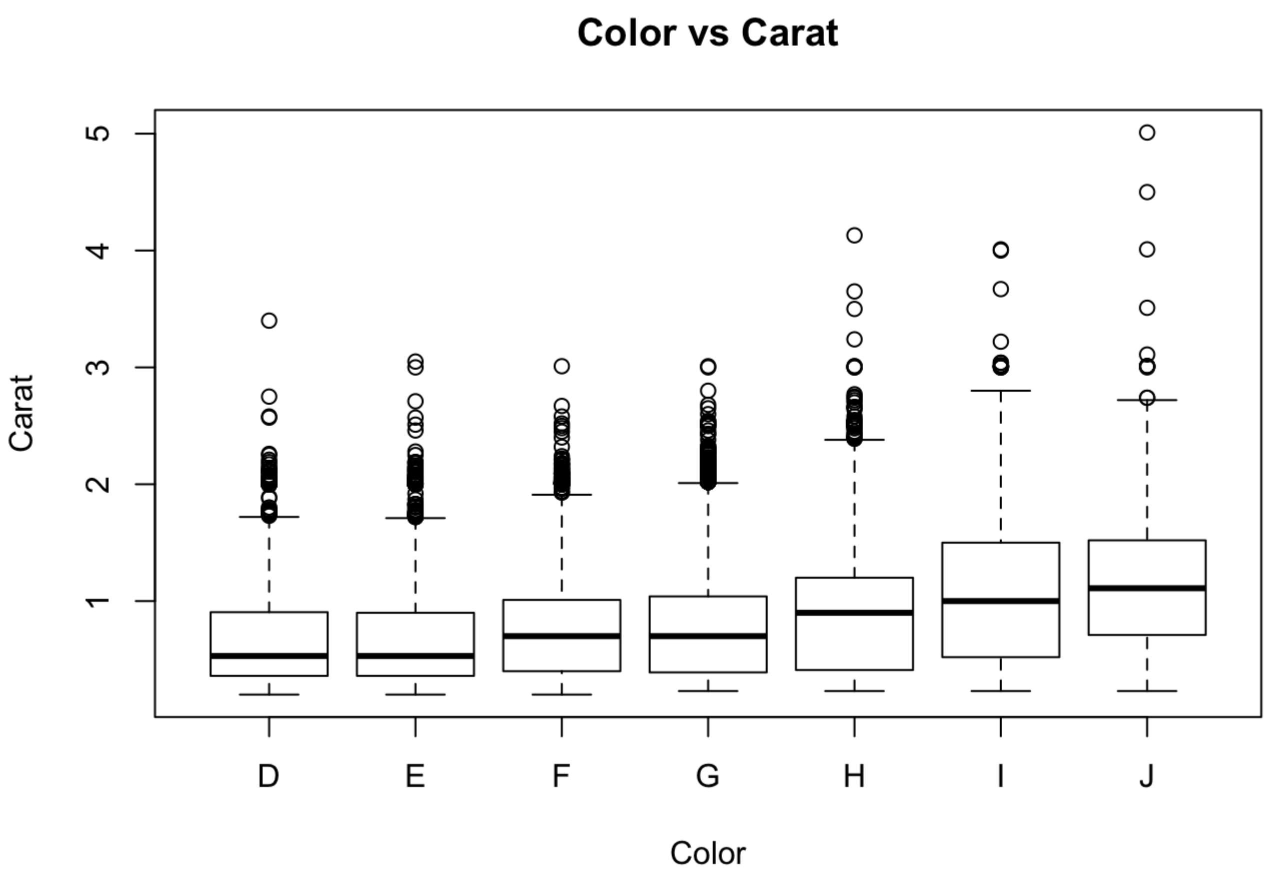
```
diamonds$color<-as.integer(diamonds$color)
cor(diamonds$color,diamonds$carat)
```

```
## [1] 0.2914368
```

This value does not have any meaning. A color can not be presented as a number value which will mean anything except a category.

#e. We can present a the carat to color relationship with a Boxplot. We can see for each color its carat specifications:

```
library('data.table')
data("diamonds")
boxplot(diamonds$carat~diamonds$color,data=diamonds, main="Color vs Carat", xlab="Col
or", ylab="Carat")
```



So we can see here that as we go from D to J the median of carat value is increasing.

#Question 10: #a.

```
MAD_comp <- function(x){
  vec_x <- sort(x)
  x_median <- median(x)
  deviations<-c()
  for (i in vec_x){
    deviations<-c(deviations, abs(i-x_median))
  }#close_for
  deviations<-sort(deviations)
  return(median(deviations)*1.4826)
}#close_function
```

#b.

```
set.seed(256)
vec_10_norm <- rnorm(10,mean = 1 , sd = 1)
sd(vec_10_norm)
```

```
## [1] 0.6417884
```

```
MAD_comp(vec_10_norm)
```

```
## [1] 0.7570071
```

#c.

```
set.seed(256)
vec_10_exp<- rexp(10,rate = 1)
sd(vec_10_exp)
```

```
## [1] 0.9655831
```

```
MAD_comp(vec_10_exp)
```

```
## [1] 0.8049221
```

#d. We would expect the 'MAD' to be closer when using with normal distrebution and the 'sd' to be more apart. This is because normal's distrebution Median and mean are close to each other. In the exponential distrebution, the mean is shifted but the median stays aproximatly the same.

```
paste("Difference in standard diviation: ", sd(vec_10_exp)-sd(vec_10_norm), " (Expone
ntian - Normal)",sep = "")
```

```
## [1] "Difference in standard diviation: 0.323794671332899 (Exponentian - Normal)"
```

```
paste("Difference in MAD: ", MAD_comp(vec_10_exp)- MAD_comp(vec_10_norm), " (Exponent
ian - Normal)",sep = "")
```

```
## [1] "Difference in MAD: 0.0479150585278129 (Exponentian - Normal)"
```

We can see that both results are greather within the Exponentian distrebution but the the ‘sd’ difference is much larger.

#e.

```
norm_diff_vec<-c()
exp_dif_vec<-c()
for(i in 1:1000){
  norm_diff_vec<-c()
  exp_dif_vec<-c()
  set.seed(256)
  vec_10_norm <- rnorm(10,mean = 1 , sd = 1)
  set.seed(256)
  vec_10_exp<- rexp(10,rate = 1)
  norm_difference <- abs(MAD_comp(vec_10_norm) - sd(vec_10_norm))
  exp_difference <- abs(MAD_comp(vec_10_exp) - sd(vec_10_exp))
  norm_diff_vec<-c(norm_diff_vec,norm_difference)
  exp_dif_vec<-c(exp_dif_vec,exp_difference)
}
mean(norm_diff_vec)
```

```
## [1] 0.1152187
```

```
mean(exp_dif_vec)
```

```
## [1] 0.1606609
```

#f. In clause ‘d’ as explained, due to the robustness of the Median, the difference in the MAD is much smaller the the difference in the Standard Deviation. Even though the tai is pulling the mean in the exponential distrebution, the median stays aproximatly the same.

In clause ‘e’ we can see that the average difference of the Exponential distrebution is greather, because in both distrebutions, the MAD is aproximatly the same but the standard deviation is greathet in the exponential distrebution, so the avarage difference in larger being calculated on the exponential distrebution.