Data Wrangling in R: Additional Examples with Answers

Clear your R workspace using the following command

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

We use the "airquality" dataset for this exercise. The dataset is part of the latest distribution of R 3.4.1 or can be accessed from the "datasets" package which you can install using install.packages() command). I have also uploaded a copy on the course portal, under datasets, just in case (most likely you already have it in your base). To check the availability of the dataset simply type *summary(airquality)* into the R-studio console (case sensitive). If you don't get an error message, it means that you have the dataset.

The following is the description of the dataset:

Format

A data frame with 154 observations on 6 variables.

[,1]	Ozone	numeric	Ozone (ppb)
[,2]	Solar.R	numeric	Solar R (lang)
[,3]	Wind	numeric	Wind (mph)
[,4]	Temp	numeric	Temperature (degrees F)
[,5]	Month	numeric	Month (1--12)
[,6]	Day	numeric	Day of month (1--31)

Details

Daily readings of the following air quality values for May 1, 1973 to September 30, 1973.

- **Ozone:** Mean ozone in parts per billion from 1300 to 1500 hours at Roosevelt Island
- **Solar.R:** Solar radiation in Langleys in the frequency band 4000–7700 Angstroms from 0800 to 1200 hours at Central Park
- **Wind:** Average wind speed in miles per hour at 0700 and 1000 hours at LaGuardia Airport
- **Temp:** Maximum daily temperature in degrees Fahrenheit at La Guardia Airport.

Source

The data were obtained from the New York State Department of Conservation (ozone data) and the National Weather Service (meteorological data).

Tasks:

1. **What are the variable names in the "airquality" dataframe?**

```
> colnames(airquality)
[1] "Ozone"   "Solar.R" "Wind"    "Temp"    "Month"   "Day"
```

2. **How many observations (rows) are in the in the "airquality" dataframe?**

```
> nrow(airquality) #number of rows
[1] 153
```

3. **What are the ranges and distribution of the variables in the "airquality" dataframe?**

```
> summary(airquality)
     Ozone           Solar.R          Wind             Temp           Month      
 Min.   :  1.00   Min.   :  7.0   Min.   : 1.700   Min.   :56.00   Min.   :5.000  
 1st Qu.: 18.00   1st Qu.:115.8   1st Qu.: 7.400   1st Qu.:72.00   1st Qu.:6.000  
 Median : 31.50   Median :205.0   Median : 9.700   Median :79.00   Median :7.000  
 Mean   : 42.13   Mean   :185.9   Mean   : 9.958   Mean   :77.88   Mean   :6.993  
 3rd Qu.: 63.25   3rd Qu.:258.8   3rd Qu.:11.500   3rd Qu.:85.00   3rd Qu.:8.000  
 Max.   :168.00   Max.   :334.0   Max.   :20.700   Max.   :97.00   Max.   :9.000  
 NA's   :37       NA's   :7                                                       
      Day       
 Min.   : 1.0  
 1st Qu.: 8.0  
 Median :16.0  
 Mean   :15.8  
 3rd Qu.:23.0  
 Max.   :31.0  
```

4. **Calculate the variance of all variables in the "airquality" dataframe using the sapply()?**

```
> sapply(airquality,var)
    Ozone   Solar.R      Wind      Temp     Month       Day 
       NA        NA 12.411539 89.591331  2.006536 78.579721 
```

We are getting NA values for numerical calculations even if we have a single missing value. To avoid this, we use the rm.na=TRUE option:

```
> sapply(airquality,var, na.rm = TRUE)
      Ozone     Solar.R        Wind        Temp       Month         Day
1088.200525 8110.519414   12.411539   89.591331    2.006536   78.579721
```

This removes the NA (missing) values from the caluculations but doesn't remove them permanently from the data frame.

5. **Select the days where the Temperature was above 77 and store the records in a new dataframe called "High_Temp". Do this using the base R and "dplyr" package.**

With Base R:

```
> High_Temp<-airquality[airquality$Temp>77,]
```

With dplyr:

#note make sure that dplyr is installed otherwise install using

install.packages('dplyr')

#while you are connected to the internet.

```
> library(dplyr)
> High_Temp<-filter(airquality,Temp>77)
```

6. **What is the average wind speed for days in June? Do this in an step by step fashion (i.e. store the filter results in a temporary intermediate dataframe, T, first)**

```
> T=filter(airquality,Month==6)
> mean(T$Wind)
[1] 10.26667
```

7. **Repeat the above question without using an intermediate temporary variable.**

with filter function from dplyr

```
> mean(filter(airquality,Month==6)$Wind)
[1] 10.26667
```

#Even more elegant solution with piping would be

```
> airquality %>% filter(Month==6) %>%  summarize(mean(Wind))
  mean(Wind)
1   10.26667
```

8. How many days in each month is reported in the dataset?

Best is to use group_by function, and summarise with count(i.e. n()) function.

```
> summarise(group_by(airquality,Month),n_days=n())
# A tibble: 5 x 2
  Month n_days
  <int>  <int>
1     5     31
2     6     30
3     7     31
4     8     31
5     9     30
```

Or using piping

```
> airquality %>% group_by(Month) %>% summarise(n_days=n())
# A tibble: 5 x 2
  Month n_days
  <int>  <int>
1     5     31
2     6     30
3     7     31
4     8     31
5     9     30
```

9. What is the variance of the temperature for days between 5th June and 16th July?

Solution 1: Many intermediate dataframes

```
> June_data<- airquality[airquality$Month==6,]
> June_data_filtered<- June_data[June_data$Day>4,]
> July_data<-airquality[airquality$Month==7,]
> July_data_filtered<- July_data[July_data$Day<17,]
> June_July_filtered_combined=rbind(June_data_filtered, July_data_filtered)
> var(June_July_filtered_combined$Temp)
[1] 40.58827
```

Solution 2: More elegant and efficient

```
> airquality %>% filter(Month==6&Day>4 |Month==7&Day<17)%>% summarise(var(Temp))
  var(Temp)
1  40.58827
```

10. Can you see a big difference between the average Solar Radiation on even days and odd days?

In R, **x %% y** is the modulus (x mod y). Example: 5%%2 is 1. For even days the modulus will be zero and for odd days it will be 1. We can use the filter function to select the even days and calculate the mean of Solar.R for those records and do the same for odd days.

```
> airquality %>% filter((Day %% 2) == 0)%>%summarise(Av_Sol_R=mean(Solar.R,na.rm=TRUE))
  Av_Sol_R
1 188.5833
> airquality %>% filter((Day %% 2) == 1)%>%summarise(Av_Sol_R=mean(Solar.R,na.rm=TRUE))
  Av_Sol_R
1 183.3514
```

11. Solve question 10 by first creating a new variable called "Odd_Even_Flag" in the dataframe and then use the Group_by and summerise functions to calculate the mean of the Solar Radiation for each group.

First, we create the new variable in the dataframe and then we use group_by()and summerise () functions.

```
> airquality$Odd_Even_Flag= airquality$Day %% 2  #1 for odd and 0 for even days
> airquality %>% group_by(Odd_Even_Flag) %>% summarise(Av_Sol_R=mean(Solar.R,na.rm=TRUE))
# A tibble: 2 x 2
  Odd_Even_Flag Av_Sol_R
          <dbl>    <dbl>
1             0 188.5833
2             1 183.3514
```