

Data Cleaning

ANLY-511-04 Group 3

12/06/2022

```
# load libraries for this assignment
library(tidyverse)
library(ggplot2)
library(readxl)
library(knitr)
library(kableExtra)
```

Functions for Use

```
calculate_proportions <- function(input_df, threshold) {

  # create data frame for printing proportions of of NA values
  df <- data.frame(colnames(input_df), round(colMeans(is.na(input_df)), 5), row.names = NULL)
  colnames(df) <- c('Column Name', 'Proportion of Values NA')
  df <- df[order(df$`Proportion of Values NA`, decreasing = TRUE),]
  df_table <- df[df$`Proportion of Values NA` > threshold,]

  return (df_table)
}
```

```
print_table <- function(input_table) {
  knitr::kable(df_table, row.names = FALSE) %>% kable_styling(bootstrap_options = "striped", full_width = F, position = "center")
}
```

```
print_shape <- function(input_df) {
  print(paste('Our dataset shape is now: (', nrow(input_df), ', ', ncol(input_df), ')', sep = ''))
}
```

Read in the Data

Let's first read in the data. We have five datasets to read in, all by the naming convention of `data/cardataXX.xlsx`, where `xx` \in {18, 19, 20, 21, 22}. We can also observe the *shape* of each dataset, indicated with the following syntax: (number of rows, number of columns).

```
# load in the data
cardata2018 <- read_excel('../data/cardata2018.xlsx')
cardata2019 <- read_excel('../data/cardata2019.xlsx')
cardata2020 <- read_excel('../data/cardata2020.xlsx')
cardata2021 <- read_excel('../data/cardata2021.xlsx')
cardata2022 <- read_excel('../data/cardata2022.xlsx')
```

Dataset Name	Shape
data/cardata2018.xlsx	(4727, 67)
data/cardata2019.xlsx	(4719, 67)
data/cardata2020.xlsx	(4450, 67)
data/cardata2021.xlsx	(4265, 67)
data/cardata2022.xlsx	(4455, 67)

The five datasets come from the same source, but we want to ensure that they are compatible with one-another. The best way to check this is to ensure that the columns are the same across all datasets. If they are, we can easily append the five datasets together, by row, in order to create one, larger dataset.

```
# extract column names for each dataset
colnames2018 <- colnames(cardata2018)
colnames2019 <- colnames(cardata2019)
colnames2020 <- colnames(cardata2020)
colnames2021 <- colnames(cardata2021)
colnames2022 <- colnames(cardata2022)

# ensure that the column names are all the same across all datasets
if ( mean(colnames2018 == colnames2019) == 1 &
      mean(colnames2018 == colnames2020) == 1 &
      mean(colnames2018 == colnames2021) == 1 &
      mean(colnames2018 == colnames2022) == 1 ) {
  print('The column names are the same and in the same order across all five datasets.')
} else {
  print('The column names are NOT the same and in the same order across all five datasets.')
}
```

```
## [1] "The column names are the same and in the same order across all five datasets."
```

Let's append these datasets together since we know now that they have the same structure and column names.

```
# bind datasets together by row
cardata <- rbind(cardata2018, cardata2019, cardata2020, cardata2021, cardata2022)
print_shape(cardata)
```

```
## [1] "Our dataset shape is now: (22616, 67)"
```

There are also some columns that we are not concerned with for the purpose of our analysis, so we will drop those before proceeding with the cleaning phase. These columns are:

- Test Vehicle ID
- Engine Code
- Shift Indicator Light Use Cd
- Test Originator
- Test Procedure Cd
- Test Category
- FE Bag 2
- Test Veh Configuration #
- Transmission Overdrive Code
- Shift Indicator Light Use Desc
- Analytically Derived FE?
- Test Procedure Description
- FE_UNIT
- FE Bag 3
- Actual Tested Testgroup
- Transmission Overdrive Desc
- Test Number
- ADFE Test Number
- Test Fuel Type Cd
- FE Bag 1
- FE Bag 4

```
# define columns to remove
remove <- c("Test Vehicle ID", "Test Veh Configuration #", "Actual Tested Testgroup",
            "Engine Code", "Transmission Overdrive Code", "Transmission Overdrive Desc",
            "Shift Indicator Light Use Cd", "Shift Indicator Light Use Desc", "Test Number",
            "Test Originator", "Analytically Derived FE?", "ADFE Test Number",
            "Test Procedure Cd", "Test Procedure Description", "Test Fuel Type Cd",
            "Test Category", "FE_UNIT", "FE Bag 1",
            "FE Bag 2", "FE Bag 3", "FE Bag 4")

# remove columns and report new shape
cardata <- cardata[, !( colnames(cardata) %in% remove)]
print_shape(cardata)
```

```
## [1] "Our dataset shape is now: (22616, 46)"
```

Address Missing Values

Now, with the columns that we do have, we can observe how many `NA` values exist in each column. We will set a threshold of **10%**; a column with more than 10 percent of its values being `NA` expresses a large number of missing values in our eyes and we will remove it from our analysis. Below, we can see all columns expressing more than 10% of their values as `NA`.

```
# print the proportions of NA values in each column using a threshold of 10%
df_table <- calculate_proportions(cardata, 0.10)
print_table(df_table)
```

Column Name	Proportion of Values NA
Averaging Group ID	0.98063
Averaging Weighting Factor	0.97073
ADFE Total Road Load HP	0.89123
ADFE Equiv. Test Weight (lbs.)	0.89123
ADFE N/V Ratio	0.89123
PM (g/mi)	0.82689
N2O (g/mi)	0.49823
CH4 (g/mi)	0.18739
NOx (g/mi)	0.14008
THC (g/mi)	0.13305
CO (g/mi)	0.13053

We can then remove these columns.

```
# remove high-probability NA columns from data frame
remove <- df_table$`Column Name`

# remove columns and report new shape
cardata <- cardata[, !(colnames(cardata) %in% remove)]
print_shape(cardata)
```

```
## [1] "Our dataset shape is now: (22616, 35)"
```

We still have columns expressing `NA` values, however. These columns, of course, have less than 10% of their values being `NA`, after the removal performed above. These columns can be seen below.

```
# print the proportions of NA values in each column using a threshold of 0%
df_table <- calculate_proportions(cardata, 0)
print_table(df_table)
```

Column Name	Proportion of Values NA
DT-Inertia Work Ratio Rating	0.07985
DT-Absolute Speed Change Ratg	0.07985
DT-Energy Economy Rating	0.07985
CO2 (g/mi)	0.05518
Aftertreatment Device Cd	0.04196
Aftertreatment Device Desc	0.04196

Column Name	Proportion of Values NA
# of Cylinders and Rotors	0.04059
RND_ADJ_FE	0.01375

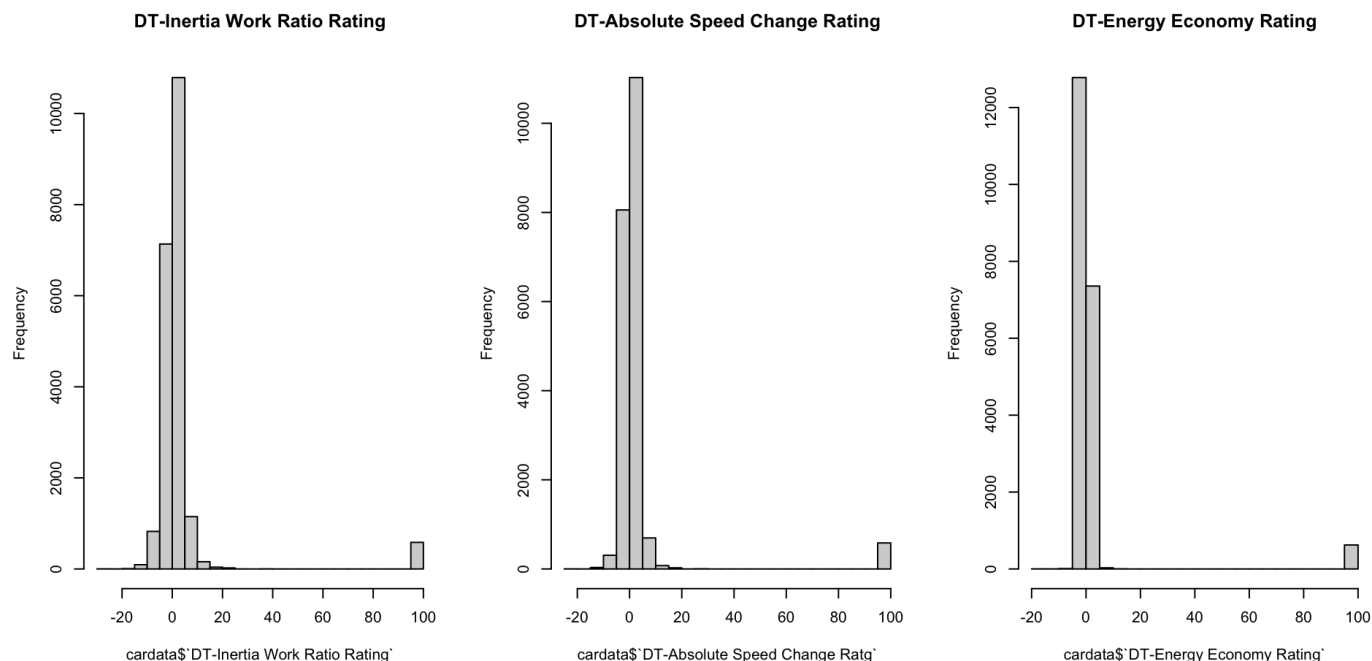
DT-Inertia Work Ratio Rating, DT-Absolute Speed Change Ratg, and DT-Energy Economy Rating

Let's start by observing the first three variables:

- DT-Inertia Work Ratio Rating
- DT-Absolute Speed Change Ratg
- DT-Energy Economy Rating

We can make histograms to see what their distributions are like.

```
par(mfrow = c(1, 3))
hist(cardata$`DT-Inertia Work Ratio Rating`, main = 'DT-Inertia Work Ratio Rating', breaks = 20)
hist(cardata$`DT-Absolute Speed Change Ratg`, main = 'DT-Absolute Speed Change Rating', breaks = 20)
hist(cardata$`DT-Energy Economy Rating`, main = 'DT-Energy Economy Rating', breaks = 20)
```



It is evident that each of these columns has a set of observations marked with a rating around 100. This is likely comparable to an `NA` value, as the distribution of values for these attributes is almost entirely hovering around a value of 0. Let's replace these values with `NA` values so that we don't confuse ourselves with them being valid.

```
# replace high outliers with NA
cardata$`DT-Inertia Work Ratio Rating`[cardata$`DT-Inertia Work Ratio Rating` > 50] <- NA
cardata$`DT-Absolute Speed Change Ratg`[cardata$`DT-Absolute Speed Change Ratg` > 50] <- NA
cardata$`DT-Energy Economy Rating`[cardata$`DT-Energy Economy Rating` > 50] <- NA
```

We can also observe our new, updated measurements of proportions of values in columns that are `NA`.

```
# create data frame for printing proportions of NA values
df_table <- calculate_proportions(cardata, 0)
print_table(df_table)
```

Column Name	Proportion of Values NA
DT-Energy Economy Rating	0.10745

Column Name	Proportion of Values NA
DT-Inertia Work Ratio Rating	0.10568
DT-Absolute Speed Change Ratg	0.10568
CO2 (g/mi)	0.05518
Aftertreatment Device Cd	0.04196
Aftertreatment Device Desc	0.04196
# of Cylinders and Rotors	0.04059
RND_ADJ_FE	0.01375

These columns, having been updated by our observations above, now have proportions of missing (or ambiguous) values greater than 10%. Keeping our threshold in mind, we will drop these columns as well.

```
# remove high-probability NA columns from data frame
remove <- df_table$`Column Name`[1:3]

# remove columns and report new shape
cardata <- cardata[, !(colnames(cardata) %in% remove)]
print_shape(cardata)
```

```
## [1] "Our dataset shape is now: (22616, 32)"
```

Again, we can observe the proportion of values in columns that are NA for columns that have at least one NA value.

```
# create data frame for printing proportions of NA values
df_table <- calculate_proportions(cardata, 0)
print_table(df_table)
```

Column Name	Proportion of Values NA
CO2 (g/mi)	0.05518
Aftertreatment Device Cd	0.04196
Aftertreatment Device Desc	0.04196
# of Cylinders and Rotors	0.04059
RND_ADJ_FE	0.01375

Aftertreatment Device Cd and Aftertreatment Device Desc

The following two variables have the same number of missing values in the dataset. Let's see if they are from the same rows.

- Aftertreatment Device Cd
- Aftertreatment Device Desc

```
# extract row names where these three variables are missing
after_cd_null <- rownames(cardata[is.na(cardata$`Aftertreatment Device Cd`), ])
after_desc_null <- rownames(cardata[is.na(cardata$`Aftertreatment Device Desc`), ])

# ensure that the column names are all the same across all datasets
if ( mean(after_cd_null == after_desc_null) == 1 ) {
  print('These variables are missing in the same rows in the dataset.')
} else {
  print('These variables are missing in different rows in the dataset.')
}
```

```
## [1] "These variables are missing in the same rows in the dataset."
```

The missing values in the dataset for these two variables *do* come from the same rows, which makes sense given their association with each other; the Aftertreatment Device Desc variable is just a description for the label of the Aftertreatment Device Cd variable. Since these are categorical variables, and there are only a small number of NA values (less than 5%), we will leave the NA values as they are and likely remove them if we engage in an analysis on both columns. We opt not to remove the rows corresponding to these missing values because there is valuable data in the other 30 columns present.

The last three variables containing missing values left to address are:

- CO2 (g/mi)
- # of Cylinders and Rotors
- RND_ADJ_FE

CO2 (g/mi)

Let's start with CO2 (g/mi) and observe its missing values.

```
# extract the rows where 'CO2 (g/mi)' is missing
na_co2 <- cardata[is.na(cardata$`CO2 (g/mi)`), ]

# proportion of electric vehicles missing this value
prop <- sum(na_co2$`Test Fuel Type Description` == 'Electricity') / sum(cardata$`Test Fuel Type Description` == 'Electricity')
print(paste("The proportion of Electric vehicles missing a value for 'CO2 (g/mi)' is:", prop))
```

```
## [1] "The proportion of Electric vehicles missing a value for 'CO2 (g/mi)' is: 1"
```

We can see that *all* electric vehicles are missing a value for the CO2 (g/mi) attribute. This may be indicative of an emission rate of **0.00 g/mi**, so we will fill these missing values with that value. Still, we will keep this assumption in mind when proceeding with this data, as our assumption might affect the results of future analysis.

```
# replace all electric vehicle CO2 emissions with 0 (all are missing, as discovered above)
cardata[cardata$`Test Fuel Type Description` == 'Electricity', ]$`CO2 (g/mi)` <- 0
```

of Cylinders and Rotors

Let's move on to # of Cylinders and Rotors.

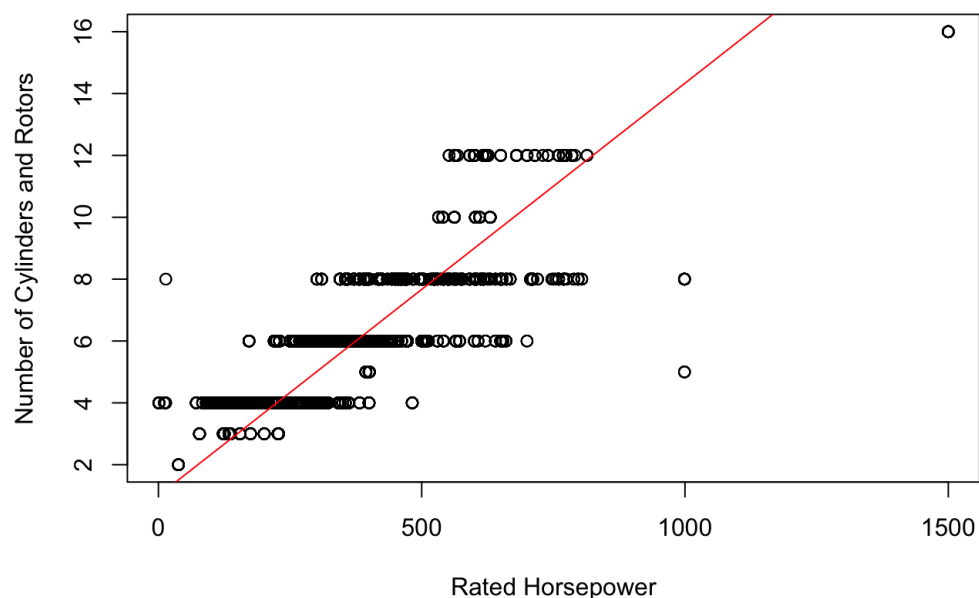
```
# table of number of cylinders
table(cardata$`# of Cylinders and Rotors`)
```

```
##
##      2      3      4      5      6      8     10     12     16
##    29   400 10807    45  6537  3333   166   350   31
```

We can see above that the number of cylinders and rotors is almost exclusively an even number, with values of 3 and 5 appearing some number of times.

```
# plot number of cylinders versus horsepower
plot(cardata$`Rated Horsepower`, cardata$`# of Cylinders and Rotors`, xlab = 'Rated Horsepower', ylab = 'Number of Cylinders and Rotors', main = 'Number of Cylinders and Rotors vs. Rated Horsepower')
abline(a = 1, b = 1/75, col = 'red')
```

Number of Cylinders and Rotors vs. Rated Horsepower



Above, we can see an approximation of the number of cylinders (# of Cylinders and Rotors) and rotors predicted by the horsepower (Rated Horsepower). Using this approximation, we will fill in the missing values of # of Cylinders and Rotors using the nearest even value. Note that the even values make up the vast majority in this dataset, as values of 1, 3, and 5, for instance, are very rare.

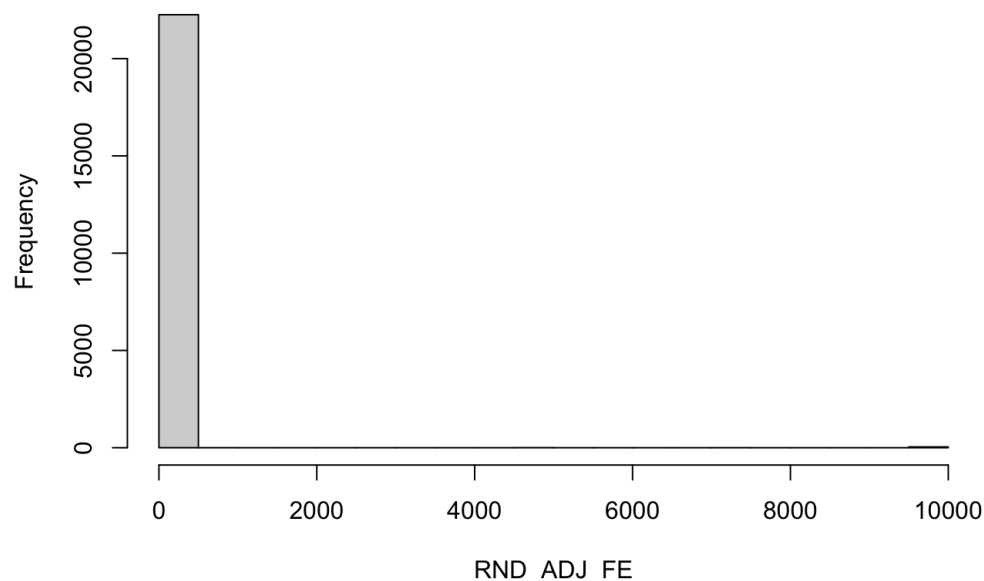
```
# replace missing cylinder values with the approximation above
for (i in 1:nrow(cardata)) {
  if (is.na(cardata$`# of Cylinders and Rotors`[i])) {
    val <- 1 + cardata$`Rated Horsepower`[i] / 75
    cardata$`# of Cylinders and Rotors`[i] <- round(val / 2) * 2
  }
}
```

RND_ADJ_FE

Finally, let's move on to RND_ADJ_FE .

```
# histogram of RND_ADJ_FE
hist(cardata$RND_ADJ_FE, xlab = 'RND_ADJ_FE', main = 'Histogram of RND_ADJ_FE')
```

Histogram of RND_ADJ_FE

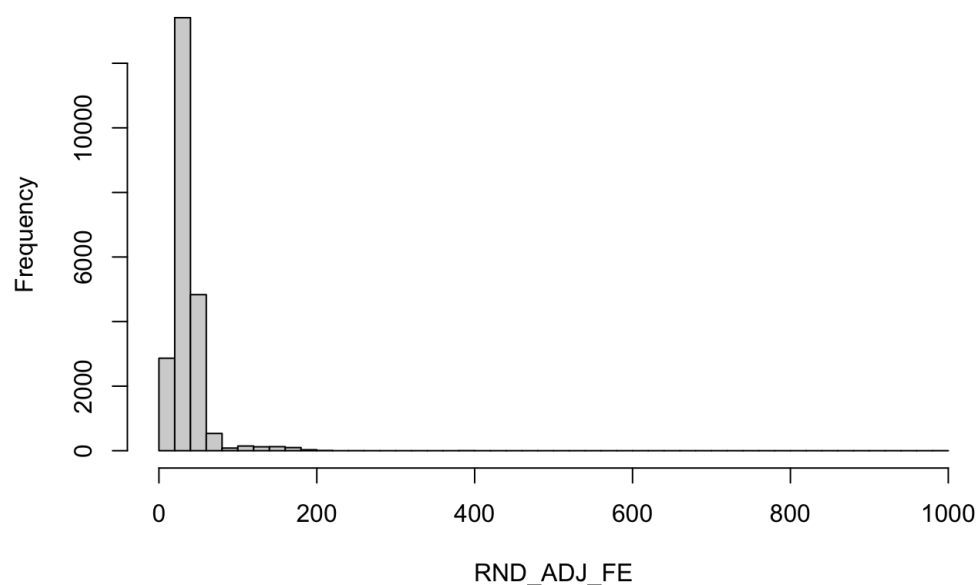


We can see that the distribution of `RND_ADJ_FE` is hidden by the outliers present in the data. Let's first replace those with missing values before proceeding.

```
# replace outliers with NA
cardata$RND_ADJ_FE[cardata$RND_ADJ_FE > 2000] <- NA

# re-view the histogram now
hist(cardata$RND_ADJ_FE, xlab = 'RND_ADJ_FE', main = 'Histogram of RND_ADJ_FE', breaks = 40)
```

Histogram of RND_ADJ_FE



This variable is a bit ambiguous, so we will replace the missing values with the median of the existing values. The values are all focused around a values of approximately 0 to 75, so this feels to us as a safe decision to proceed with.

```
# replace missing values with median of existing values
cardata$RND_ADJ_FE[is.na(cardata$RND_ADJ_FE)] <- median(cardata$RND_ADJ_FE, na.rm = TRUE)
```


Let's take a look at the proportions of missing values now, as we have addressed them all (and kept some).

```
df_table <- calculate_proportions(cardata, 0)
print_table(df_table)
```

Column Name	Proportion of Values NA
Aftertreatment Device Cd	0.04196
Aftertreatment Device Desc	0.04196
CO2 (g/mi)	0.01822

After addressing the missing values in the data, we find that we only have approximately **0.32%** of our data missing. This is a great improvement from the **15.64%** that we had initially and something that we can take with us as we look to analyze the data further.

Export Clean Data

Finally, we need to export this cleaned data so that we can use it in our analyses.

```
write.csv(cardata, '../data/cardata_clean.csv')
```