15.780, Fall 2021

Stochastic Models in Business Analytics

Problem Set 1 - Data Cleaning - Solutions

Due date: September 24, 2021

Instructions:

1. Submit a PDF file with your solutions to Canvas before the assigned deadline. Write your name and MIT ID on your submission.
2. All plots must have clear and easy-to-read axis labels and legends,
3. **Include relevant code in the PDF submission even if the question doesn't explicitly ask for it**. This means we can give you partial credit even if the output is wrong. if appropriate.

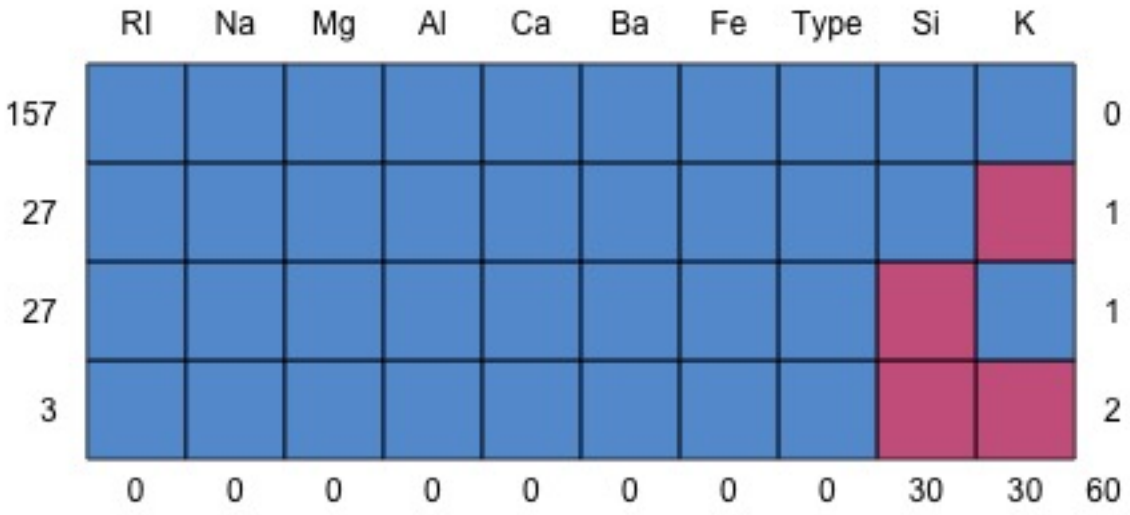
Problem 1. (**Missing Values**) (40 pts)
Problem 1 is based on the "Glass" dataset in mlbench package in R.
Hint: Load this dataset by the R commands:

```
install.packages("mlbench")
data(Glass, package = "mlbench")
```

1. (10 pts total) The original Glass dataset does not have missing values. Set seed to be 80 using the `set.seed()` command and randomly introduce 30 missing values each in the Si and K variables. Use the `md.pattern` function from the `mice` library to show the pattern of missing values. Include your code.

Table 1: Missing Values Pattern - Glass

	RI	Na	Mg	Al	Ca	Ba	Fe	Type	Si	K	
157	1	1	1	1	1	1	1	1	1	1	0
27	1	1	1	1	1	1	1	1	1	0	1
27	1	1	1	1	1	1	1	1	0	1	1
3	1	1	1	1	1	1	1	1	0	0	2
	0	0	0	0	0	0	0	0	30	30	60



2. (5 pts total) Use the `impute()` function from the `Hmisc` library to impute the variable Si with its mean value. What line(s) of code did you use? `impute(Glass$Si, mean)`

3. (10 pts total) Impute the variable Si with median value, and report the MAE, MSE and MAPE values to evaluate the accuracy of this imputation. Show your code along with the accuracy values.

```
MAE = function(actual, imputed) {
    return(mean(abs(actual - imputed)))
}

MSE = function(actual, imputed) {
    return(mean((actual - imputed)^2))
}

MAPE = function(actual, imputed) {
    return(mean(abs(actual - imputed) / abs(actual)))
}

org = original$Si
imp = impute(Glass$Si, median)
miss = is.na(Glass$Si)
c("MAE" = MAE(org[miss], imp[miss]),
  "MSE" = MSE(org[miss], imp[miss]),
  "MAPE" = MAPE(org[miss], imp[miss]))
```

Table 2: Accuracy of Median Impute - Glass - Si Variable

MAE	MSE	MAPE
0.6987	1.2681	0.0098

We will accept other numeric values.

4. (10 pts total) Impute the variable Si using KNN using the 5 nearest neighbours, and report the MAE, MSE and MAPE values to evaluate the accuracy of this imputation.

   You will need the `knnImputation` function from the `DMwR` package. Since that is no longer available in R's package repo, use the instructions on Canvas to install it.

```
library(DMwR)
```

```
imp_knn = knnImputation(Glass, k = 5)$Si
c("MAD"  = MAD(org[miss],  imp_knn[miss]),
  "MSE"  = MSE(org[miss],  imp_knn[miss]),
  "MAPE" = MAPE(org[miss], imp_knn[miss]))
```

Table 3: Accuracy of kNN - Glass - Si Variable

MAE	MSE	MAPE
0.6147	1.1383	0.0087

We will accept other numeric values.

5. (5 pts total) Based on the accuracies above, which of the two imputation techniques from question 3 and 4 do you think is better?

   All three metrics are lower for kNN, so it is a better imputation method.

Problem 2. (**Dealing with Inconsistencies**) (30 points)

This problem will help you get familiar with checking whether your data is consistent. This is often an important first step before proceeding with data analysis and fitting models. The dataset in this question (which can be obtained from Canvas) has been modified to contain some inconsistencies. It contains information about characteristics of flowers, such as petal lengths and widths. You can make use of the `editrules` library if you wish, which you can install and load by typing the following into the console:

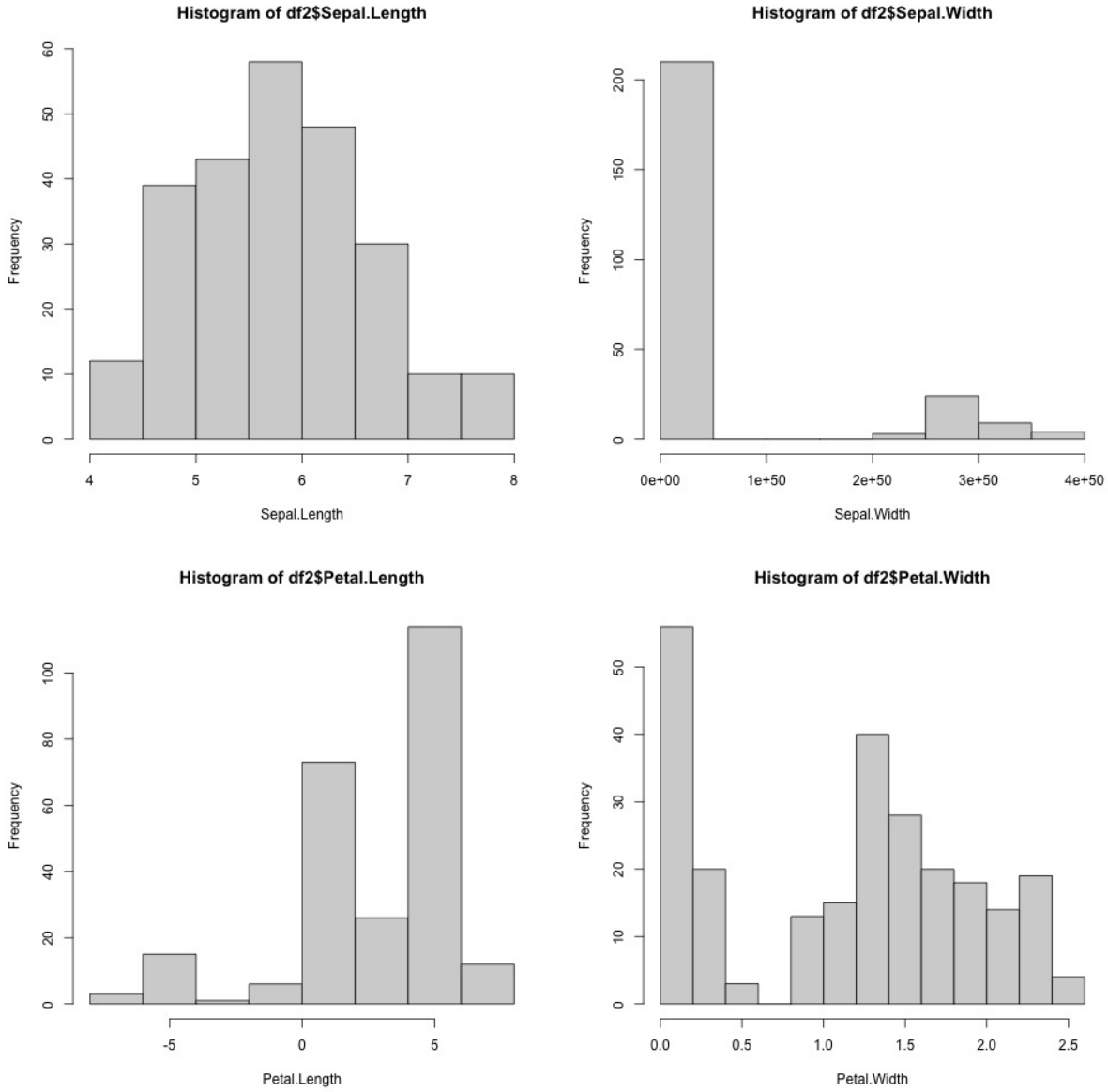
```
install.packages("editrules")
library(editrules)
```

1. (5pts) Load the *iris.csv* from Canvas (in the Files) into a dataframe. Use the **str()** function to get a first look at the data and look at the different variable types for each column. Is there any column that needs to be coerced into another variable type? If so, make the adjustments you deem necessary. Describe your operations in your Canvas submission.

   No values need to be coerced to a new datatype. They are all numeric already.

2. (10pts) Make a histogram for each numeric column. Do the values for each column make sense? Is there anything peculiar that you notice? Comment on the shape of any histogram which looks peculiar. Hint: Look at the minimum and maximum values for each histogram. Do the values for `length` and `width` make sense?

There are negative petal lengths in column `Petal.Length`; negative lengths are impossible! There are *very* large values for `Sepal.Wdith`. These values should not be this large. For histogram plots, see below.



3. (5pts) Based on your answer to the previous question, create a simple ruleset that contains the rules that the columns of the dataframe should obey. You can do this by using the **editset()** function from the editrules library. Apply your rule set to the dataframe by using the **violatedEdits()** function.

```
E = editset(c("Petal.Length > 0", "Sepal.Width < 100"))
v = violatedEdits(E,df)
```

4. (5pts) How many observations violate each of the rules from your ruleset? Remember that you can sum over a logical vector, since TRUE counts as a one and FALSE as a zero. Show your code.

   `sum(v[,1])` Number of violations for the first rule is 25

   `sum(v[,2])` Number of violations for the second rule is 40

5. (5pts) Filter your dataframe to contain only those observations that do not violated any of the rules. You can do so using the subset function and the logical vectors returned by your **violatedEdits()** function call. What are the dimensions of this new dataframe?

   `subset(df2, !v[,1] &!v[,2])` Dimension of 185 rows and 4 columns. We will accept other answers, so long as it aligns with the rule defined in part 3 above.