ISE 5103 Intelligent Data Analytics
Homework #4

Instructor: Charles Nicholson

See course website for due date

Learning objective: Data Wrangling!

Submission notes:

1. Team assignment! Include all team member names on the submitted work.
2. You will submit a PDF file with your solutions. Additionally, you will provide the R code you created to address the problems. The PDF is primarily what will be graded. The grader *may* view your R code, but should never *have* to in order to find your solutions.
3. In the PDF, clearly identify each problem (e.g. Problem 1a, Problem 2b, etc.) Also, note that only *relevant* and informative computer output should be provided.
4. Make sure to *provide comments* on what your R code is doing. Keep it clean and clear!
5. You will submit your complete R script. Note: include `library` commands to load *all* packages that are used in the completion of the assignment. Place these statements at the top of your script.
6. Do not zip your files for submission. Submit exactly two files. Name the files "LastName-HW1" with the appropriate file extension (that is, .pdf for the write-up and .R or .Rmd file for the code)

1 Data Quality Report

You will create a Data Quality Report for the Housing Data. The report has two parts: (i) a continuous valued data report, and (ii) a categorical valued data report.

In the first, for each numeric variable, you will show the number of observations, the number of missing observations, the percent of missing, the number of unique values, the percent of unique values, the mean, min, Q1, median, Q3, max, and standard deviation. See Figure 1 for an example.

variable	n	missing	missing_pct	unique	unique_pct	mean	min	Q1	median	Q3	max	sd
Id	1000	0	0.0	1000	100.0	500.500	1	251	500	750.2	1000	288.819
MSSubClass	1000	0	0.0	13	1.3	57.185	20	20	50	70.0	190	41.875
LotFrontage	1000	207	20.7	102	10.2	68.745	21	58	68	80.0	313	23.198
LotArea	1000	0	0.0	760	76.0	10424.881	1477	7500	9422	11423.5	215245	9940.619
OverallQual	1000	0	0.0	10	1.0	5.979	1	5	6	7.0	10	1.310

Figure 1: Excerpt of Data Quality Report: Continuous Valued Variables

The second part applies to the non-numeric data. For each variable, you will show the number of observations, the number of missing observations, the percent of missing, the number of unique values, the percent of unique values, the frequency ratio*, first mode, first mode frequency, second mode, second mode frequency, least common value, and least common frequency. See Figure 2 for an example.

*ratio of the first mode frequency to the second mode frequency

variable	n	missing	missing_pct	unique	unique_pct	freqRatio	1st mode	1st mode freq	2nd mode	2nd mode freq	least common	least common freq
MSZoning	1000	0	0.0	4	0.4	5.32	RL	803	RM	151	RH	10
Alley	1000	938	93.8	3	0.3	1.82	Grvl	40	Pave	22	Pave	22
LotShape	1000	0	0.0	4	0.4	1.92	Reg	633	IR1	330	IR3	7
LandContour	1000	0	0.0	4	0.4	22.62	Lvl	905	Bnk	40	Low	26

Figure 2: Excerpt of Data Quality Report: Categorical Valued Variables

You will use the `tidyverse` and specifically the `ddplyr` package to help you manipulate the data so as to create the report. I will provide some detailed instruction to help with this problem. Any problem part that does not have points assigned does not require you to show anything in your PDF submission.

(a) After loading the housing data into a data frame (or tibble) named `housingData`, run the code listed below to create three new variables.

```
housingData <- housingData %>%
       dplyr::mutate(age = YrSold - YearBuilt,
                     ageSinceRemodel = YrSold - YearRemodAdd,
                     ageofGarage = YrSold - GarageYrBlt))
```

(b) (2 points) Use the `dplyr` package to create a tibble named `housingNumeric` which contains all of the numeric variables from the original data. Please use the `dplyr::select` command along with the `is.numeric` function to complete this task.

(c) (2 points) Use the `dplyr` package to create a tibble named `housingFactor` which contains all of the numeric variables from the original data. You can use `dplyr::select` command here or, if you like, consider the `transmute` command to simultaneously keep only the character variables *and* change all character variables to factors.

(d) Try the `glimpse` command to take a look at your new tibbles.

(e) (4 points) Unfortunately, R does not have a method for extracting only Q1 or Q3. So, we will create our own user-defined functions to do this for us. Use the following code to create two new functions, `Q1` and `Q3`, respectively.

```
Q1<-function(x,na.rm=TRUE) {
    quantile(x,na.rm=na.rm)[2]
}

Q3<-function(x,na.rm=TRUE) {
    quantile(x,na.rm=na.rm)[4]
}
```

Briefly explain what these two new functions are doing.

(f) Next, we are going to create a new function that will apply several summary statistics to our data all at once. Create the new function `myNumericSummary` with the following code.

```
myNumericSummary <- function(x){
   c(length(x),  n_distinct(x), sum(is.na(x)), mean(x, na.rm=TRUE),
     min(x,na.rm=TRUE), Q1(x,na.rm=TRUE), median(x,na.rm=TRUE), Q3(x,na.rm=TRUE),
     max(x,na.rm=TRUE), sd(x,na.rm=TRUE))
}
```

This code accepts a numerical vector `x` as an input parameter and then returns a vector where the first element is the length of the input vector (i.e., the number of observations), the second element is the number of unique values, the third is the number of missing values, the forth is the mean value of non-missing numerics, etc. Notice the use of our new functions `Q1` and `Q3`.

(g) (8 points) Utitlize the `dplyr::summarize` command together with the new `myNumericSummary` function to apply the new function to every variable in the `housingNumeric` data set. You may need to look up some examples of how to use `summmarize` and the `across()` syntax from `dplyr` to do this efficiently. Save the results of this operation in a new tibble named `numericSummary`.

```
> glimpse(numericSummary)
Rows: 10
Columns: 40
$ stat            <chr> "n", "unique", "missing", "mean", "min", "Q1", "median", "Q3", "max", "sd"
$ Id              <dbl> 1000, 1000, 0, 500, 1, 251, 500, 750, 1000, 289
$ MSSubClass      <dbl> 1000.0, 13.0, 0.0, 57.2, 20.0, 20.0, 50.0, 70.0, 190.0, 41.9
$ LotFrontage     <dbl> 1000.0, 102.0, 207.0, 68.7, 21.0, 58.0, 68.0, 80.0, 313.0, 23.2
$ LotArea         <dbl> 1000, 760, 0, 10425, 1477, 7500, 9422, 11424, 215245, 9941
```

Figure 3: `glimpse(numericSummary)`

(h) Next, column bind some labels to our summary statistics with the following code.

```
numericSummary <-cbind(
                  stat=c("n","unique","missing","mean","min","Q1","median","Q3","max","sd"),
                  numericSummary)
```

If you `glimpse` the results, it should look something like Figure 3.

(i) While this is good data here, you need to perform a little trick on it so we can use the `kable` function and produce the table we want, i.e., need to "pivot" the data a couple of times. You also need to add a couple more computed values: percent missing and percent unique fields. Use the following code to accomplish this.

```
numericSummaryFinal <- numericSummary %>%
    pivot_longer("Id":"ageofGarage", names_to = "variable", values_to = "value") %>%
    pivot_wider(names_from = stat, values_from = value) %>%
    mutate(missing_pct = 100*missing/n,
           unique_pct = 100*unique/n) %>%
    select(variable, n, missing, missing_pct, unique, unique_pct, everything())
```

and finally, produce the first part of the Data Quality report,

```
library(knitr)
options(digits=3)
options(scipen=99)
numericSummaryFinal %>% kable()
```

(j) (30 points) Create the second part of the Data Quality report associated with the non-numeric data. See Figure 2 for a report excerpt.

Note: R does not have functions for identifying the first, second, or least commmon modes. Use the code below to accomplish this.

```
getmodes <- function(v,type=1) {

   tbl <- table(v)
   m1<-which.max(tbl)

   if (type==1) {
        return (names(m1))                    #1st mode
   }
   else if (type==2) {
        return (names(which.max(tbl[-m1])))  #2nd mode
   }
   else if (type==-1) {
        return (names(which.min(tbl)))       #least common mode
   }
   else {
        stop("Invalid type selected")
   }
}
```

Note: R does not have functions for identifying the frequencies of the first, second, or least commmon modes. Use the code below to accomplish this.

```
getmodesCnt <- function(v,type=1) {

   tbl <- table(v)
   m1<-which.max(tbl)

   if (type==1) {
       return (max(tbl))        #1st mode freq
   }
   else if (type==2) {
       return (max(tbl[-m1]))   #2nd mode freq
   }
   else if (type==-1) {
       return (min(tbl))        #least common freq
   }
   else {
       stop("Invalid type selected")
   }
}
```

2 Transformations

There are several data transformations that can be used to enhance data quality for modeling purposes. In this problem you will examine a few such transformations based on the `housingData` tibble from the first question.

(a) (8 points) Via visual inspection, identify two numeric variables that are highly skewed (e.g., not symmetric and far from normally distributed). Use a transformation method (e.g., ladder of powers or boxcox transformation) to transform these variables to be more normally distributed. Show visual depictions of distributions before/after transformations.

(b) (20 points) The variable `LotFrontage` has several missing values. Impute the missing values using:

i. mean value imputation

ii. regression with error

iii. predictive mean matching (Use the `mice` package and optionally see `https://datascienceplus.com/imputing-missing-data-with-r-mice-package/` for help)

iv. For all of the above show visual depictions of how the data was transformed (e.g., histogram or density plots)

(c) (10 points) Several categorical variables in the `housingData` have multiple factor levels.

Variables with many factor levels can pose a problem for supervised learning. Specifically, categorical variables are typically transformed into a series of so-called "dummy variables" – where each dummy variable takes on a binary value according to the observation's factor level. If a categorical variable has n factor levels, then $n-1$ dummy variables are required to represent the information. See Figure 4.

Color
red
yellow
blue
red
green
yellow

Red	Yellow	Blue
1	0	0
0	1	0
0	0	1
1	0	0
0	0	0
0	1	0

Figure 4: Example dummy variable encoding

Dummy variable encoding can *vastly* increase the dimensionality of the data and cause all kinds of problems. One method to address this is to "collapse" factor levels down to fewer levels.

Use the `forcats` package to do just that: Collapse the factor levels in the `Exterior1st` down to only five levels – the first four levels should be the most frequent levels and all other levels should be collapsed into a single "Other" level.

(d) (16 points) More fun with factors

i. Use `tidyverse` packages to compute the average `SalePrice` for each `Neighborhood` factor level.

ii. Create a parallel boxplot chart of this data, i.e., a boxplot associated with the sale prices for homes in each of the 18 neighborhoods.

iii. You should notice that there is a lot of variation in price by neighborhood. Using `forcats` re-order the factor levels of the `Neighborhood` variable in descending order of the median price per neighborhood (i.e., the neighborhood with the highest median price is `NoRidge`, the next highest median is `NridgHt`, etc., so `NoRidge` should be the first level and `NridgHt` should be the second factor level, etc.)

iv. If you have done re-ordering correctly, you should be able to produce a parallel boxplot of neighborhoods and sales prices in descending order (see Figure 5). Note: R orders values in graphs according to the ordering of the factors.

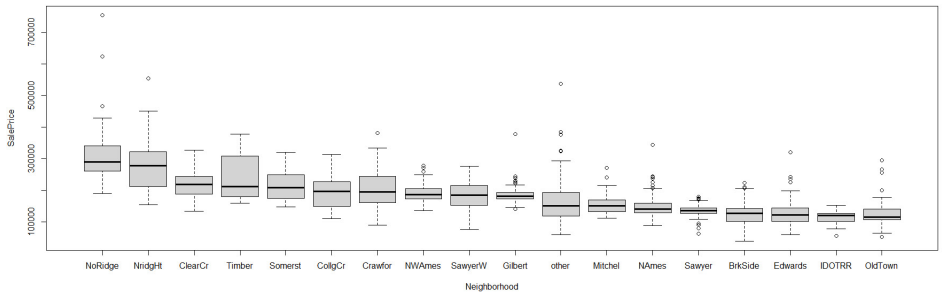

Figure 5: Ordered parallel boxplots