Exercise 2: Explore data surfaces in R

DATA 306

The main purpose of this exercise, like the last one, is to give you a sense of where we are going in this class. And like the last exercise, I take this opportunity to introduce you to some fundamental data carpentry in R. The exercise is **not** meant to *test* you on the particulars (we will go over these in more detail later). So: follow along with the commands I give you in your own script, tweak arguments, and take copious notes.

The only thing you need to supply is some light interpretation. If you do not give adequate interpretations where I ask for them, I will probably ask you to revise and resubmit.

In Exercise 1, we adopted a *deductive* approach to analyzing our NYC schools data. In this exercise, we will adopt a more *inductive* approach. Both approaches are widely used, often within the same analysis. The distinction between the two boils down to motive: do you want to know more about the sample (the data as given), or are you trying to understand patterns existing beyond the sample (the world outside the data)? In the last exercise, we were aiming for the world outside the data. In this exercise, we are mainly interested in getting a better grip on the data as given.

The deductive approach dominates formal scientific practice, but the inductive approach is the true analytic workhorse.

1. Make a few comment lines indicating who you are, the assignment name, date, etc. Same template as Exercise 1.

2. Read the data (`read.csv()`), assign it a name (`<-`), and get a look at its `str()`-ucture. This should be the same method and data you used in Exercise 1. For example:

   ```
   school <- read.csv("./data/NYCschools_r1.csv")
   str(school)
   ```

   We should see the same `data.frame` object of 469 rows and 25 columns as in Exercise 1. Some additional details about the `str()` output:

   - A `data.frame` is not the only kind of object that can hold "data."[1] But `data.frame` is a good place to start. It's the cornerstone of a great deal of R's modeling software.
   - The variable types (`int`, `num`, `chr`, etc.) do not come from thin air. By default, R decides how the variables should be classified based on the first five values. So if the first five rows in a given column look like "characters," then R will classify the column as `chr` column. This method does not work with every dataset, but there are ways to make the `read.csv()` function look at more values. The tradeoff is that looking further down the column means more processing time.

[1] The smarty pants `tidyverse` people decided to make their own version of a `data.frame` called a `tibble`, for example.

3. There are some unnecessary columns in `school`. If you examine `year`, you'll notice it is *invariant*, or that it does not change. And `X` is just the index number of all the rows, which is already implied by their order. We should *subset* the `data.frame` and create a new object with just columns we want.

   There are, as always, a lot of ways to do this. `subset()` requires no additional packages and is (relatively) intuitive:

   ```
   sch <- subset(school, select = -c(X, year))
   ```

   `c()` means **concatenate**, or turn the items enclosed in parentheses (`X` and `year` in this case) into a list of items. The minus symbol means select all BUT the list of concatenated items.

   We can check on the column names to make sure everything worked the way we expected it to:

   ```
   colnames(sch)
   ```

   It's good to look at your data like this. But your eyes can decieve. As the old proverb goes, *trust, but verify.* In this case, it is easy enough to come up with a quick "gut test": the difference in the number of columns in `sch` and `school` should be exactly two:

   ```
   ncol(school) - ncol(sch)
   ```

4. We can make summary statistics for our entire dataset with ease. But let's not summarize our identifier variables, since these are just unique codes associated with each school. We can exclude them from a given function call by using **indices** instead of column names within the `subset()` function. Indices, or index values, are integers that give the position of a row or column — you can think of them as coordinates, as in the game Battleship or Chess.

   With the `Extract[]` function, which is usually invoked with just square brackets (`[]`), I can reference columns by index number, rather than name. Again, the index values refer to the column's position in the sequence of columns, starting with 1 on the leftmost column. The colon signals a range.

   `c()` and the negative sign are doing the same thing here as above. `c()` is for concatenation of separate elements into a list. It's necessary here because we need to use the negative sign. Again, the negative sign means *exclude* all columns referenced after the sign. If we omitted the negative sign, we would *only include* columns 1 through 2. Compare the outputs between the first and second, commented line.

   Finally, notice that we include a comma *before* the concatenated list of index values. In `data.frame` (and `matrix`) objects, rows are always the first coordinate, followed by columns, e.g. `sch[X, Y]`, where X is a row number and Y is a column number. Leaving one of the coordinates blank implies we want all rows/columns.

   ```
   summary(sch[ , -c(1:2)])
   ## summary(sch[ , 3:23]) # equivalent to above
   ```

Write a few sentences comparing two distributions (columns, variables). Cite specific numbers in your interpretation.

5. By (my) design, all of our variables besides the school identifiers are *continuous* variables. Recall that continuous variables change across an unbroken surface, like a ramp. Height is usually coded as a continuous variable; the ramp glides from shortest to tallest. Education credentials ("highest degree obtained") are an example of an ordinal variable; the staircase steps unevenly from elementary to graduate degree. Continuous variables are meaningful at any point up or down the 'ramp,' whereas the space between two categories is not usually well-defined, even if they are ordered in steps.

   (a) Demographic categories like gender, race, ethnicity, and disability are normally understood as categorical variables. Yet they presented as continuous here, and it is not a bug. How does that make sense? Justify in a sentence or two. Refer to Exercise 1 for variable definitions.

   (b) Conceptually (not in R), how could these continuous variables be converted into categorical (ordinal or nominal) variables? Give an example in a few sentences or commented "pseudocode" if you wish.

6. It's important to understand how these variables "covary." We should know this before we pipe this data into a regression model, but also because we might be interested in understanding how various aspects of NYC schools are related to each other (or not).

   (a) A group of continuous variables can be compared in a correlation matrix. You can make one with the `cor()` command.

   ```
   cor.sch <- cor(sch[ , 3:7], use = "complete.obs")
   cor.sch
   ```

   (b) Adjust the code above so that columns 3 through 22 are included in the correlation matrix.

   (c) It's hard to see the inter-relations of 21 variables. The correlation matrix you just constructed should have 21 rows by 21 columns of correlations (check this with `dim(cor.sch)`), implying more than 21 × 21 = 441 total cells in the matrix. You can go to the terminal and type `cor.sch` and then hit ENTER to see all of them.

   We can get a better grip on these relations by abstracting them into some visualizations. Even with 441 statistics to summarize, this is an appropriate place for a stemplot, which conveys an enormous amount of information in a compressed space.

   To make this work, we need to do a few things to our correlation matrix. First, our correlation matrix includes *all* pair-wise correlations between variables. So for example, `cor.sch` has the correlation between `disability` and `ESL` (-0.028) and between `ESL` and `disability` (also -0.028). So it doubles every correlation.

   Another problem is that our matrix also reports all correlations of our variables with themselves. These all appear in what is called the "diagonal" of the matrix (look at `cor.sch` again and note the pattern of ones). This means the diagonal will always report a correlation of 1, or 100%.

   To fix both problems, we will ignore one half of the relations and the diagonal, taking only the *lower triangle* of the matrix (it would also be OK to take the `upper.tri()`):

```
cor.sch.lt <- cor.sch[lower.tri(cor.sch) == TRUE]
```

`lower.tri()` returns what is called a "logical." It looks at every cell in the matrix and asks if it is one of the cells in the lower triangle. If it is, then the cell gets the value `TRUE`. If not, it returns `FALSE`. When we put this in brackets and force it to only be exactly equal (double equal signs) to `TRUE`, we are telling R to only consider the lower triangle of the matrix.

Now we can take our new object `cor.sch.lt` and feed it into the `stemplot()` function.

```
stem(cor.sch.lt)
```

Think of the stemplot as a sideways histogram, where (the first line of output informs us) the tens place of the correlation is given by the first column of numbers, and the hundredths place by the second. The first correlation listed, then, is -0.82.

Again, this gives us a lot to work with, even though we don't know which variable-pairs are associated with a particular correlation coefficient. In comments in your script, write two sentences (minimum) describing this distribution. Feel free to produce a numerical summary of the `cor.sch.lt` distribution to support your interpretation.

7. Suppose we want to know which correlation coefficients go with which variable-pairs. We could examine the correlation matrix, but a more human-readable method is what is known as a *correlation plot*. Many correlation plotting functions have been implemented. Here, I show you one plot that I think works well.

   This plot is dense with information, including the sign (positive or negative) of the correlation and a color key. The color intensity and size are directly proportional to the size of the correlation (more intense and larger means higher correlation). The variables are reordered to cluster the most similarly-correlated variables together, and we only see the lower triangle of correlations.

```
## install.packages("corrplot") # uncomment to install the package, then re-comment
library(corrplot)             # load the package
corrplot(cor.sch, type = "lower", order = "hclust")
```

   What does the plot suggest about NYC schools? Write a few observations in comment form, pointing to specific relationships indicated by the correlation plot. It's OK if you are unsure about your interpretations at this point — just write down your intuitions and note how confident you are in them.

8. Let's take another look at this system of relations with what is called a *scatterplot matrix*. The scatterplot matrix is even more dense with information than the correlation plot, so we will "zoom in" on a few relations that stand out in the correlation matrix.

   Once we decide on some relations of interest, all we have to do is give `scatterplotMatrix()` the subset from our `data.frame`. I give some code using square brackets to do the subsetting. Not that the `scatterplotMatrix()` function takes the `data.frame` as an argument, rather than the correlation matrix.

   To begin, let's look at the relations between the proportion of students in elementary, middle, and high school grades and outsiderness (remember, the actual measure is the proportion of students, teachers, and parents who disagree with the idea that *students* feel they belong in their school). Later, I'll ask you to look at a set of variables that interests you.

```
## install.packages("car")
library(car)
scatterplotMatrix(sch[, c("outsiders_student", "outsiders_teacher",
                          "outsiders_parent", "total_enrollment")])
```

The diagonal of the scatterplot matrix shows the "kernel density" of each variable alone, along with a special kind of "strip chart" showing the density of observations along the x-axis. The off-diagonals give the pair-wise relations between each variable. So for example, looking at the top left diagonal (`outsiders_student`), we see a slightly right-skewed distribution in the first box.

To the right of the first box, we see a scatterplot of `outsiders_student` on the Y-axis and `outsiders_teacher` on the X-axis. The solid blue line running through the points is the line from a bivariate regression, while the dashed line (hard to see in this box) is a LOESS smoother (see Exercise 1). Remember that the top and bottom triangles give the same set of relationships, but flip the Y- and X-axes.

What kinds of relations do you see here? In particular, do you see any evidence of a relation between outsiderness and total enrollment? Write at least two full sentences of interpretation. In your interpretation, be sure to reference relevant correlation coefficients. The correlation coefficients of this set of variables can be viewed with the following code (I use their column and row indices rather than their names):

```
cor.sch[c(1, 17, 19, 21), c(1, 17, 19, 21)]
```

9. I know much less about NYC public schools than most of you. But it seems like they have different compositions of what would conventionally be understood as elementary-, middle-, and high-school aged students. (Tell me if I'm wrong.)

   This density plot illustrates the variation in composition pretty well (density plots are also shown in the diagonals of the scatterplot matrix above). The code is a bit complicated, so pay attention to the comments if you'd like a better sense of what's going on here. You can also feel free to just copy the code and not worry too much about its meaning for now.

```
hdens <- density(sch$high) # calculate kernel densities

plot(hdens, # ?plot to see the meaning of these arguments
     xlim = c(-.2, 1.23),
     main = "red = elementary, blue = middle, black = high",
     xlab = "")

## lines() plots another set of lines over an existing plot.
### densities are not saved as a named object for concision
lines(density(sch$elementary), col = "red")
lines(density(sch$middle), col = "blue")
```

   What does this mean? The Y-axis gives the "probability density," a concept that will require some explanation later in the course. For now, think of the plot as an abstracted histogram, where the curves show how age groups are distributed across schools.

High- and elementary-school kids seem the most segregated from others (the red and black peaks at zero on the x-axis). Middle school-aged kids are distributed more widely. In some cases they make up about a third of the population. But other schools are exclusively devoted to them (the blue peak at 1.0). This indicates there are a few different sorts of schools: middle school-only; elementary and high schools with some middle school-aged kids; and mixed grades. You can tell me if this guess matches with your knowledge.

In any case, it seems there are several "types" of schools with different mixes of grades. We should wonder if the relationship between total enrollment and outsiderness depends on the type of school — elementary, middle, or high. Technically, such a question would be called an *interaction* or *moderated* relation. Here's an (imperfect) example of how you might diagram it in `DiagrammeR`:[2]

Nota bene: Due to the idiosyncracies of document export, you will **NOT** be able to copy and paste this code from the PDF version of this exercise (the apostrophes will not work). Rewrite it or use the HTML version instead.

```
library(DiagrammeR)
grViz("digraph {
            node [shape = plaintext]
            ’+’ [shape = doublecircle]
            ’School size’ -> ’+’ -> ’Outsiders’
            ’Student age’ -> ’+’
            {rank = same; ’School size’ ’Outsiders’ ’+’}
            }")
```

Moderation implies that a third variable changes the relationship between two other variables. It is statistically equivalent to interaction, though in interaction the conceptual relations among the three variables are somewhat more ambiguous. Public health researchers and psychologists tend to use the language of moderation. Other analysts more often speak of "interaction."

10. How do student age groups change the relationship between enrollment and the number of outsiders? We can look at this by distinguishing between different kinds of schools in the scatterplot matrix we constructed in part 7.

    This step requires creating a new, "factor" (categorical) variable that distinguishes between different kinds of school. To do so, we will use `$` and our assignment operator `<-` to name a variable within `sch`. To begin, we'll fill this new column with only `NA` values.

```
sch$agemix <- NA
## defines a new column within sch, where each row has the value NA
```

    Then, we can assign a descriptive label for each row falling within the humps we found in the density plot. The elementary and high school humps were close to 50% of student enrollments, so we'll say that any school with more than 50% of these groups is an "Elementary" or "High"

[2] For more on causal diagrams in DiagrammeR, see: https://www.erikigelstrom.com/articles/causal-graphs-in-r-with-diagrammer/. You can play around more directly with graphviz and the dot coding language here: http://magjac.com/graphviz-visual-editor/.

school, respectively. The middle school hump was centered all the way out at 100%, so we'll take everything greater than 70% of enrollment in these grades as a "Middle" school. You can come up with your own cutoffs if you disagree.

To clarify the plots we produce below, I choose to ignore both middle school-only schools and "mixed" schools (a residual category). You can uncomment these if you want to see them, or choose a different mix of age groups to view.

```
sch$agemix[sch$elementary > 0.5] <- "Elementary"
## sch$agemix[sch$middle > 0.7] <- "Middle"
sch$agemix[sch$high > 0.5] <- "High"
## sch$agemix[is.na(sch$agemix) == TRUE] <- "Mixed"
```

It's important to check to see if our code did what we intended.

```
table(sch$agemix)
```

Your table should list 108 elementary schools and 72 high schools. The full scheme (if uncommented) classifies all 469 schools in `sch` in one of four categories. Elementary and high schools cover only about 38% of our full sample of schools (180/469 schools) and about 41% of total enrollment (calculation below – don't worry about the code unless you're interested).

```
sum(sch$total_enrollment[sch$agemix == "High"|
                         sch$agemix == "Elementary"],
    na.rm = TRUE)/
    sum(sch$total_enrollment,
        na.rm = TRUE)
```

Let's see how the categories we constructed vary in their relations between our independent (school size) and dependent (outsiderness) variables.

To do so requires some change in syntax. `scatterplotMatrix()` uses what is known as "formula syntax." In general, formula syntax looks somewhat like you might expect:

`<outcome> ~ <x1> + <x2> + <x3> | <x4>`

Where we replace everything in <> (including < and >) with a variable (column) name.

The formula we implement below is "one-sided" because it does not include an "outcome" to the left of the tilde (~). I broke the formula into two lines to make it easier to read in the PDF handout. The vertical line (AKA "bar") on the right hand side of the formula (|) tells the function to differentiate between age groups within the plot. Formula syntax requires us to specify our data in a separate argument (`data = sch`). The other arguments (`smooth` and `legend`) are for clarity.

```
scatterplotMatrix(~ outsiders_parent + outsiders_teacher +
                      outsiders_student + total_enrollment | agemix,
                  data = sch,
                  smooth = FALSE, # make TRUE to see LOESS
                  legend = list(coords = "bottomright"))
```

We will see formula syntax quite a bit more in this course. Make your XQuartz window larger if you can't see the details. RStudio users may suffer here from the application's graphics visualization defaults.

The results are interesting: the relationship between the proportion of various groups who believe students do not belong and total enrollment appears to vary in each type of school. Elementary schools show a flat or negative relationship between enrollment and outsiderness. High schools show either a flat or positive relation. The crossed lines give some evidence, within this subsample, of "crossover moderation." That is, of a positive relation existing for one group, and a negative for another. Importantly, this kind of moderation will be invisible (the contrasting relations "cancel out") if we keep these groups aggregated together as we did in our first scatterplot matrix.

11. Your turn. What other systems of associations exist in this data? Create your own scatterplot matrix including at least four variables other than those considered above. Include a two-three sentence interpretation of what you find, including discussion of the contrasting relations among different kinds of schools (either by the definition given above or your own).