HOMEWORK INSTRUCTIONS

You will most likely use RStudio to do the homework but, if you wish, you may use the copy of a virtual machine located at solace.ist.rit.edu/~mjmics. This virtual machine has a copy of R and all the packages used in this class preloaded. It will give you experience with working on a virtual machine which you may need in practice if you have to work with large data sets. RIT has a license for VMware you can use as students to host this virtual machine. In practice, you would likely use a virtual machine hosted remotely, often via Amazon Web Services or a similar service, but the mode of interaction is the same. I am enthusiastic about the use of virtual machines and am happy to help you set this up. INSTRUCTIONS FOR HOMEWORK I

R Markdown Files. For the homework assignment, you will turn in two files. You will turn in an R markdown file and a pdf file that results from rendering the R markdown file.

0. Zip any files you turn in. There should be a single zip file named with the number of the homework as a lowercase roman numeral. For example, homework ii should be enclosed in a zip file called ii.zip. MyCourses will add your name (plus a great many numbers) to the file name automatically.

1. Use a full-featured text editor that saves your files as UTF-8 or use R Studio.

2. Save the files with the extension .Rmd and make the file name match the roman numeral of the assignment but in lower case. For example, the second assignment will be called ii.Rmd and so on.

3. Whenever your answer includes R code, write the code flush left surrounded by code fences. A code fence consists of a line with three backticks flush left and no other text except the letter r in curly braces (you can put certain other code inside the curly braces as well and you will learn that later) for the opening code fence and none at all for the closing code fence. Here is an example.

```{r label} library(ISLR) pairs(Auto) with(Auto,(plot(mpg,cylinders))) sapply(Auto[,3:7],mean)

```

1. If you want to include mathematical expressions, you may write them in LaTeX and surround them with dolcannot understand LaTeX (it will be demonstrated forb lar signs. This will allow you to say things like *𝜇*0. If you

you and examples will be given), you can write the expressions out phonetically. For example, the above expression is pronounced *mu hat nought* and is written in LaTeX as $\widehat\mu\_0$.

R markdown is documented in Chapters 21, 23, and 24 of our textbook as well as in other publications we will discuss. You will need to fami

# PART 1. DESCRIBE THE 311 DATA

For this assignment, you will give a preliminary description of the 311 data. It should include pictures and tables and a data dictionary and be presented in an r markdown document called It should be included in a .zip file called ii.zip along with its rendered version, ii.pdf, and there should be no subdirectories in the .zip file. You will need to investigate the nature of the data using whatever means you can think of, such as googling. (A data dictionary gives definitions of columns and any specifications of limitations of what can be entered in the column. For example, a column containing zip codes consists of either exactly five digits or exactly nine digits with a dash between the fifth and sixth digit.

A column containing borough names consists only of the following entries: Bronx, Brooklyn, Manhattan, Queens, Staten

Island.)

Like all the remaining assignments, you are likely to find this difficult, time-consuming, and taxing to your imagination. The best way to learn is to throw yourself into a hard project like this. Following are some tips that you might discover in your own journery, but that I have collected here to make sure that you consider them. You may want to skim them a few times to make sure you understand.

**Usearemoteserver.** Reading and working with a 4.4 GB file can be daunting on many laptops and even lab machines. There is a wide range of available cloud solutions you can use instead. Personally, I find it convenient to process the file on a Macbook Pro with 16GB RAM and at least 40GB free storage. But if you have a 4GB laptop with maxed out storage, you’re going to want to explore one of the following options.

* solace: This is the machine from which I serve the data so if you use it via ssh you don’t have to copy the data. You can read it directly into R either giving the path or the URL to my copy. The main limitation for solace is that it doesn’t have a graphical interface except via the browser. You can actually create your graphics in the www directory and view them with a browser. Everyone in the class should be able to login to this machine with your RIT credentials. If it doesn’t work, talk to me.
* aws: Amazon Web Services offers a good deal for students and has machines with R preinstalled.
* google cloud: Google also offers a good deal for students and also has R installed on cloud-based machines. The main thing to keep in mind is to turn the service off when you are not using it so you won’t run out of free time during the semester. The same is true of Amazon.

**Use tinytex.** You are required to *render* your .Rmd files. This means to convert them to pdf and include all processing, graphical and otherwise. The easiest way to do this by far is install tinytex and ignore any suggestions about installing miktex or other tex-related packages. Tinytex is comparatively automated, which is what makes it easier. TeX, like R, relies on packages and tinytex will guess and install required TeX packages in a place where RStudio can find them. Installing tinytex is a two-step process as shown below.

**install.packages**('tinytex') tinytex::**install\_tinytex**()

After you install tinytex, you never refer to it explicitly. Instead you give the following commands that are not part of your .Rmd document but instead give the commands at the R console, referring to your .Rmd document.

**if** (!**require**(rmarkdown)) {

**install.packages**("rmarkdown",dependencies=TRUE) **library**(rmarkdown)

}

**render**("ii.Rmd",**pdf\_document**(latex\_engine="xelatex"))

**Use fread().** Reading a 4.4 GB file into R can be quite time-consuming. You can speed the process up a great deal by using the fread() function from the data.table package. Not only does it read data rapidly compared to read.csv() but it does sophisticated data wrangling. Read about it in by saying help(fread) after you enter the following code.

Notice the structure of the following if construct. This is a common construct for loading packages.

|  |
| --- |
| **if** (!**require**(data.table)) { **install.packages**("data.table",dependencies=TRUE) **library**(data.table)  } nyc311<-**fread**("311.csv") **names**(nyc311)<-**names**(nyc311) %>% stringr::**str\_replace\_all**("\\s", ".") |

**Avoid City.** The City column contains a vast number of misspellings and odd entries that don’t correspond to any definition of City. You can try to clean this column up or just drop it. Most students will not use it in any analysis anyway.

**Drop columns before you drop rows.** Identify columns you won’t analyze and drop them before you drop any rows that you consider problematic. This is faster than dropping rows first because it affects more data.

**Avoid duplicate rows.** As far as I know, there are no completely duplicated rows in the nyc311 data. It is often a good practice to check for duplicate rows, though, and to eliminate them if they exist. It is also the case with this data that you can remove the Unique Key column and find many rows that are otherwise duplicates that way. That is actually better than leaving the Unique Key column since you won’t need it for any analysis.

**if** (!**require**(dplyr)) {

**install.packages**("dplyr",dependencies=TRUE)

|  |
| --- |
| **library**(dplyr)  } nyc311nodups<-**distinct**(nyc311) **all\_equal**(nyc311nodups,nyc311) |

The last command above checks to see if two dataframes are equal. If they are, it pays to rm() one of them and save a lot of memory. If they are not, then be sure to save the smaller of the two, which should be nyc311nodups in this case.

**Usesectionheadings.** Rmarkdown uses section headings where # is a main section, ## is a subsection, and so on. You should use these to make it easier to follow your document. One rule you need to remember is that the section marker should be preceded by a blank line and should be the first thing on the line where it occurs. It should be followed by a space.

**Use chunk names.** When you render your .Rmd file, R will emit the name of each chunk as it is processed or “unnamed chunk” if you haven’t given it a name. Therefore it makes sense to give names to your chunks so you (and I) can better follow the execution process. You can not give the same name to two chunks in a document. You can give a chunk the label *Initialize* as follows.

```{r Initialize} initialization code here

```

**Sample 10,000 rows.** As you work with the nyc311 data, you are likely to make mistakes. Some of these mistakes will involve waiting a very long time for results. You should first create a sample of data, say 10,000 rows, and work with that. Only when you are sure your code works, then should you work with the entire data set.

sample<-nyc311[**sample**(**nrow**(nyc311),10000),]

**Use xtable().** There are several tools for making betterlooking tables than you will get by default. One example is xtable(). The argument to xtable() must be a valid data frame. The code fence enclosing the table must include the results="asis" option (line 1 in the following code block) or else the raw LaTeX will be rendered instead of the finished table. Note that xtable() will let you create a table too wide for the page. In that case, there are two easy alternatives. One is to make it a landscape table and the other is to use pander() instead of xtable().

```{r results="asis"} library(xtable) xtable(head(sample))

```

To print the table in landscape form, you can do something like the following.

```{r} library(kableExtra) landscape(knitr::kable(head(mtcars),"latex"))

```

**Use the sample file to start.** There is a sample file called ii.Rmd and a matching output file called ii.pdf on myCourses. You can download this and use it to start with. Make sure you have it in the working directory and give the following commands at the R console.

**library**(rmarkdown)

**render**("ii.Rmd",**pdf\_document**(latex\_engine="xelatex"))

You should be able to reproduce the ii.pdf exactly if you have installed all the relevant packages mentioned above in R. Otherwise you will see a lot of error messages. You may want to go back and skim these tips until you find you have installed all the relevant packages. Make sure you can properly render the file before you continue.

Next you should locate the parenthesized instructions in the ii.Rmd file and begin expanding the file. This will require you to explore R and ggplot2 and other packages. Try to really improve on this file before turning it in as your homework ii.

**Infer borough from zip code or lat / long.** There are many UNSPECIFIED borough entries. Googling *infer borough from zip code* immediately returns a list of zip codes per borough. Googling *inferboroughfromlat/long* is not as straightforward. There are some R packages you can use, sp and rgdal. There are publicly available ESRI shapefiles for zip codes. You can use these packages and shapefiles to determine some of the zip codes from lat / long coordinates. This is a somewhat difficult task and is rewarded as such. HW III. EXPLORE THE 311 DATA

For this assignment, you will conduct an exploratory data analysis of the 311 data. This is a much deeper dive than the previous assignment. Here you will look for connections between columns. These questions could include, for instance, what complaints are most associated with which agencies or which boroughs generate the most complaints of each major category. You will be graded on the questions you raise as well as the answers you discover. Turn in an R markdown file and a rendered pdf file recording your exploration.

One element you may wish to explore is the latitude and longitude variables assigned to each row. You can generate a map by modifying the following code. Note that you will have to get your own key for the Google Maps API instead of using the one below. You *must not* try to use the entire nyc311 data set as there are too many points to visualize. Instead, choose a sample and choose only one kind of complaint. Modify the following code to refer to the smaller data frame you generate. Do *not* try to run the following code on the entire data set.

)

|  |
| --- |
| *# Get NYC map in Black and White* **if**(!**require**("devtools")) { **install.packages**("devtools")  } **if**(!**require**("ggmap")) { devtools::**install\_github**("dkahle/ggmap", ref = |

"tidyup"

),

|  |
| --- |
| } key<-"AIzaSyAN47wCuHi93plFKGb6A1Kf8vc6bdfJHAE" **register\_google**(key = key) nyc\_map = **get\_map**(location = **c**(lon= -73.9, lat= 40.7 maptype = "terrain", zoom =12)  *# There is a new update regarding google maps requirement # for API key hence this code above is required.* map <- **ggmap**(nyc\_map) + **geom\_point**(data=bla,**aes**(x=bla$Longitude,y=bla$ size= 0.4, alpha=0.2, color= "red") +  **ggtitle**("Map of bla and bleah") + **theme**(plot.title = **element\_text**(hjust = 0.5)) + **xlab**("Longitude") + **ylab**("Latitude") map |

Latitude),