***Infrastructure:***

There are three folders in the Autoclean Data ver 1\_01 package, an auto-cleaning package for luminex data.

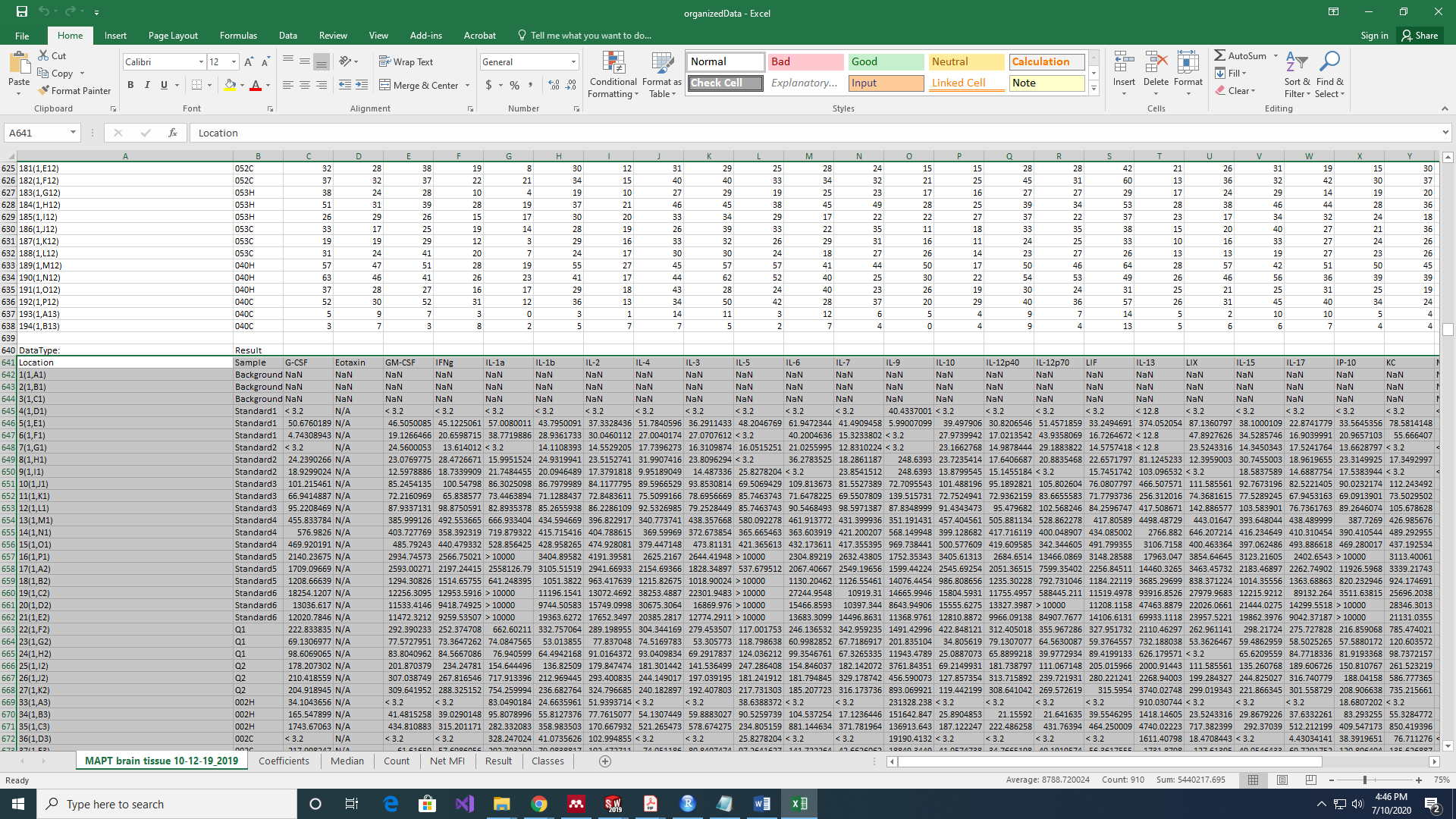
* R Scripts: Contains the two scripts that can be used, one is the package itself and the other is the usable script
* Example Files 1: Contains three excel documents (an input and output) as well as 2 log text files for those runs
* Example Files 2: Contains three excel documents (an input and output) as well as 2 log text files for those runs

***Prepping Luminex Data:***

Before running any of the R scripts or functions in the R scripts folder, the user must first prep the raw luminex data first. The prepping step only involves the user moving segments of the raw file into new sheets in the same excel file. Examples are found in both ‘Example Files 1’ and ‘Example Files 2’. A breakdown of the procedure is to have the user create 6 new sheets in the same excel file (a total of 7 excel sheets should exist). This six sheets should be named with the following names (include capitals as shown):

* Median
* Net MFI
* Count
* Result
* Classes
* Coefficients

The raw Luminex data should have headers such as Median or Count above each table within the initial sheet. Highlight all cells within that table (excluding the table name such as Median or Count) as shown in the picture below.



Note that in the picture, the user has highlighted all cells in the table Result, but excluded highlighting the row with the table name Result as well as the neighboring cell that held the value DataType. \***Note that if there are empty rows between valid data, these rows/columns must be removed and can be done by navigating to the special tab in excel, select blanks and then click ok. Then delete the highlighted cells.**

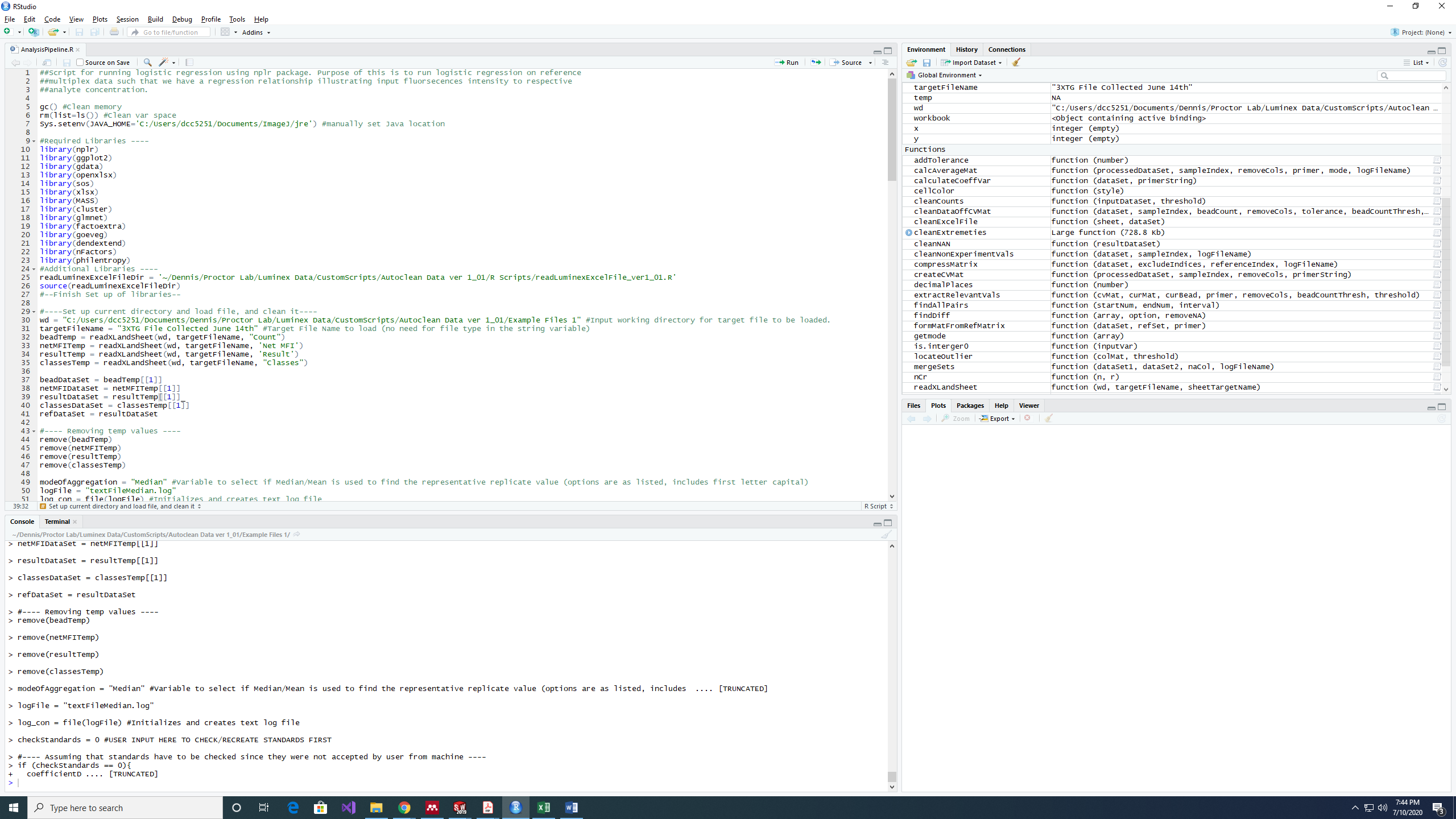
Copy each set of data into a brand new sheet with the respective name (e.g. the datatype Result should be copied into a new excel sheet with the name Result), do this for every datatype specified. For the class sheet, it is a user input where each row holds the classes for the observations. **Ensure that the file type for the excel file is an excel workbook and not a csv.** Once done, the excel file is now prepped.

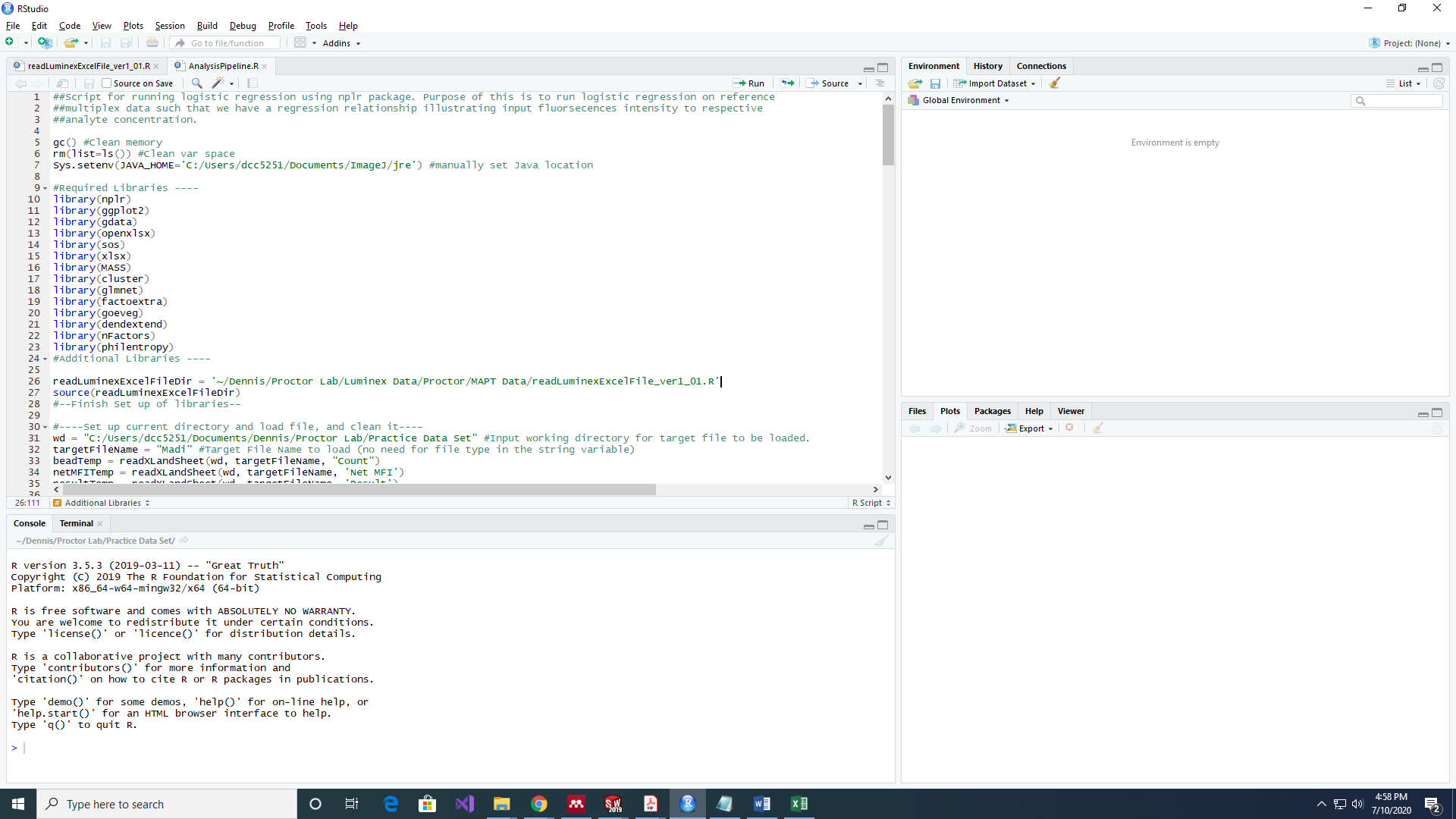
***Starting out:***

Navigate to the R scripts folder, there should be two R scripts there.

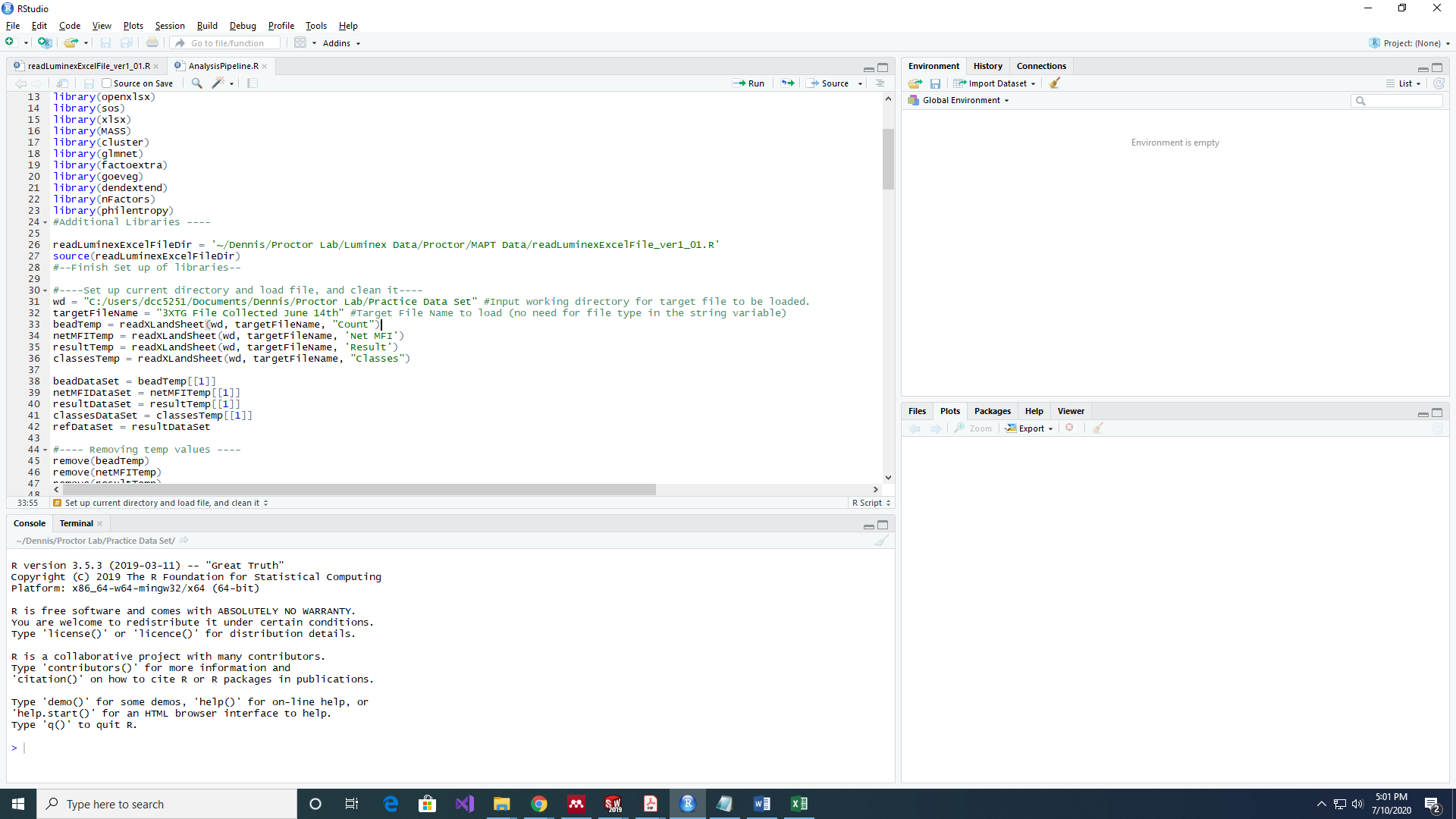
* AnalysisPipeline.R: This R script file is the only R script file that should be opened by the user when using this package to auto-clean luminex data.

Select the *AnalysisPipeline.R* file.

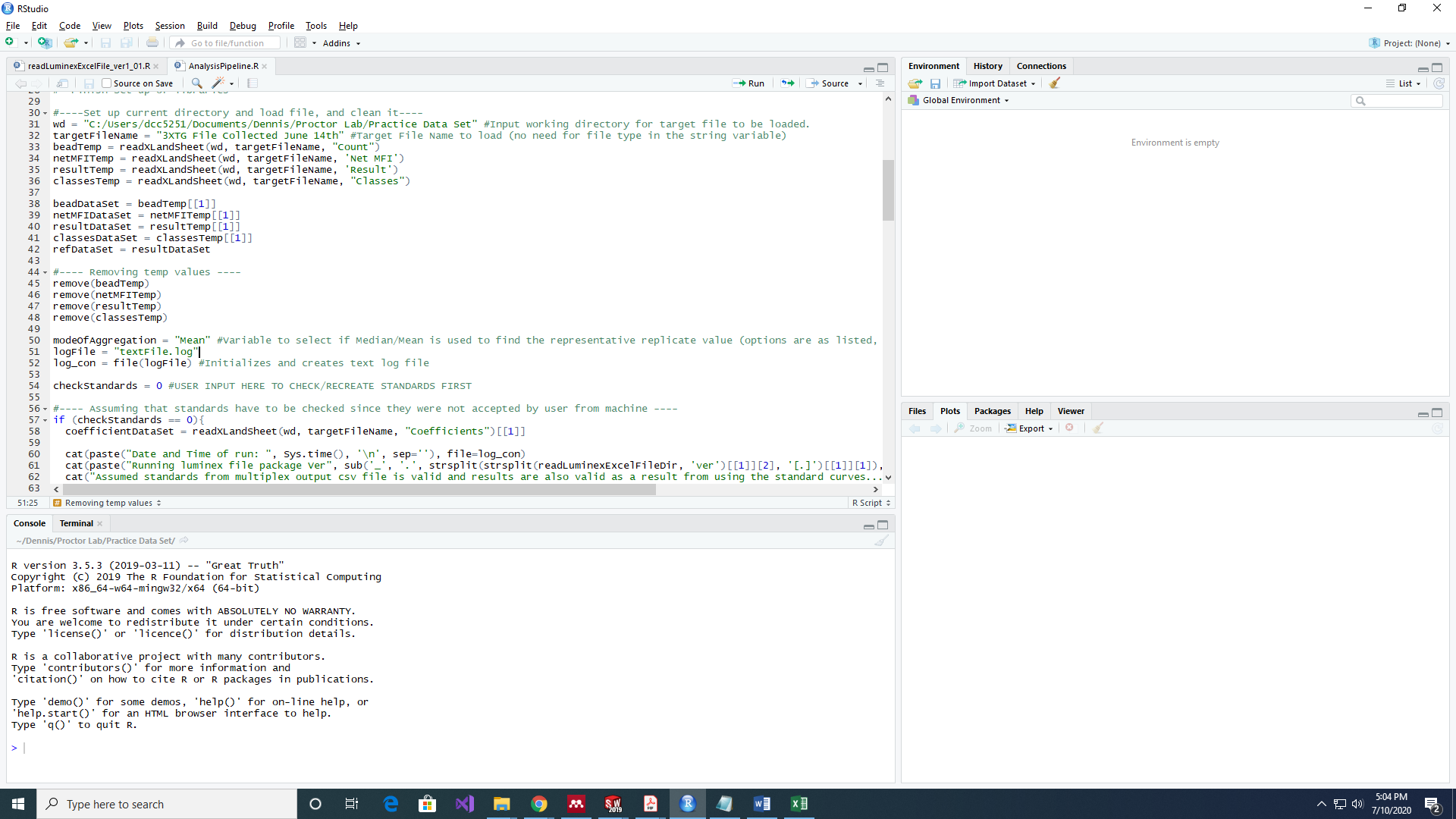
Open the file and navigate to the 7th line that has the Sys.setenv phrase as shown above. Change the value it is equal to another directory that is the jre (java) location. All R users should have Java installed by default. Navigate to the line that contains the variable ‘readLuminexExcelFileDir’. An image of the line is shown below:

This variable is used to store the location of the *readLuminexExcelFile\_ver1\_01.R* file, which is the package containing all the functions in the script. Input the respective location of the R file and store it in the variable.

Navigate to the lines that have the variables *wd* and *targetFileName* as shown below.

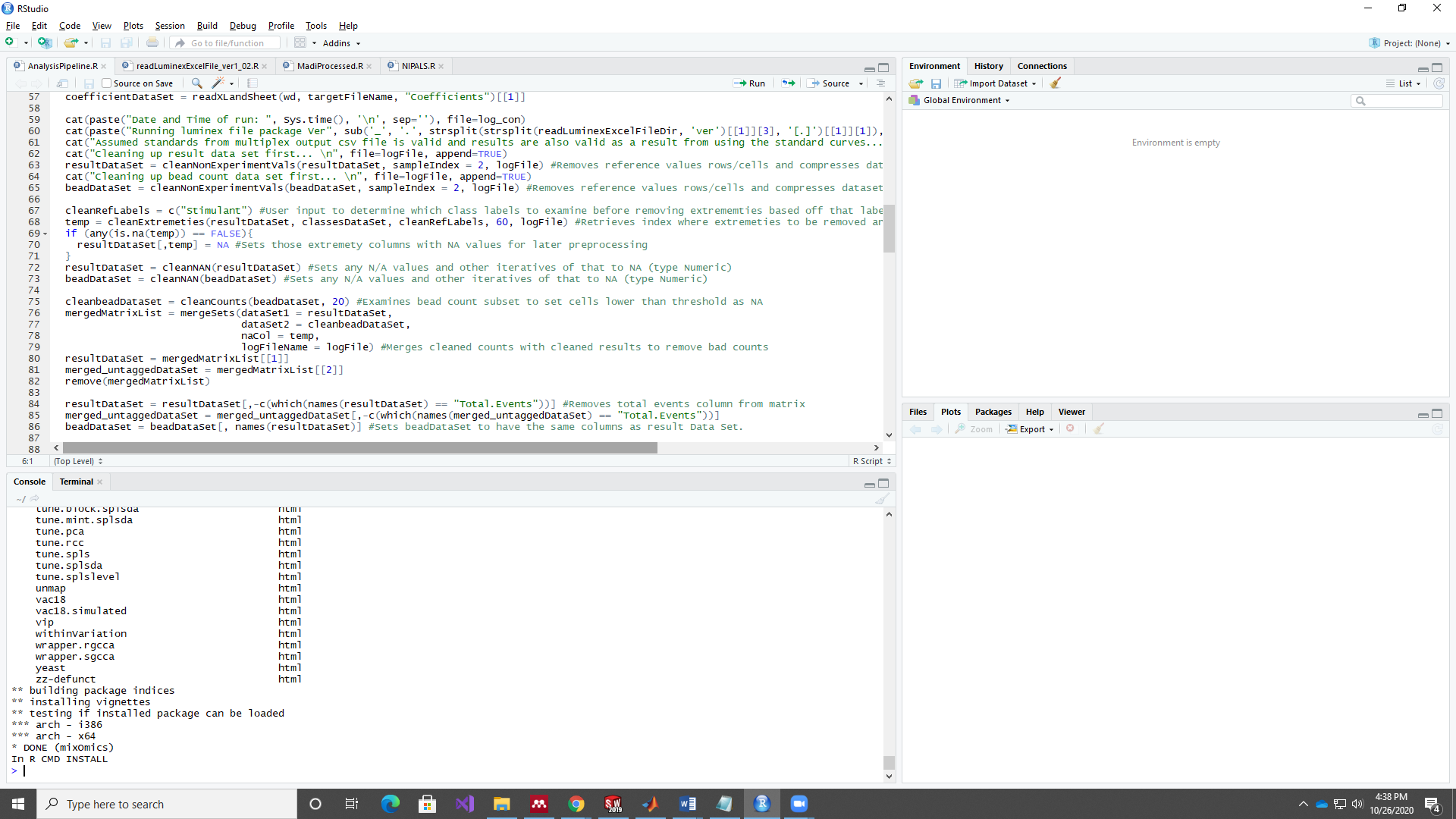


The variable *wd* stores the working directory (file directory) for where the raw luminex data excel sheet is located. The variable *targetFileName* is used to store the excel file’s name.

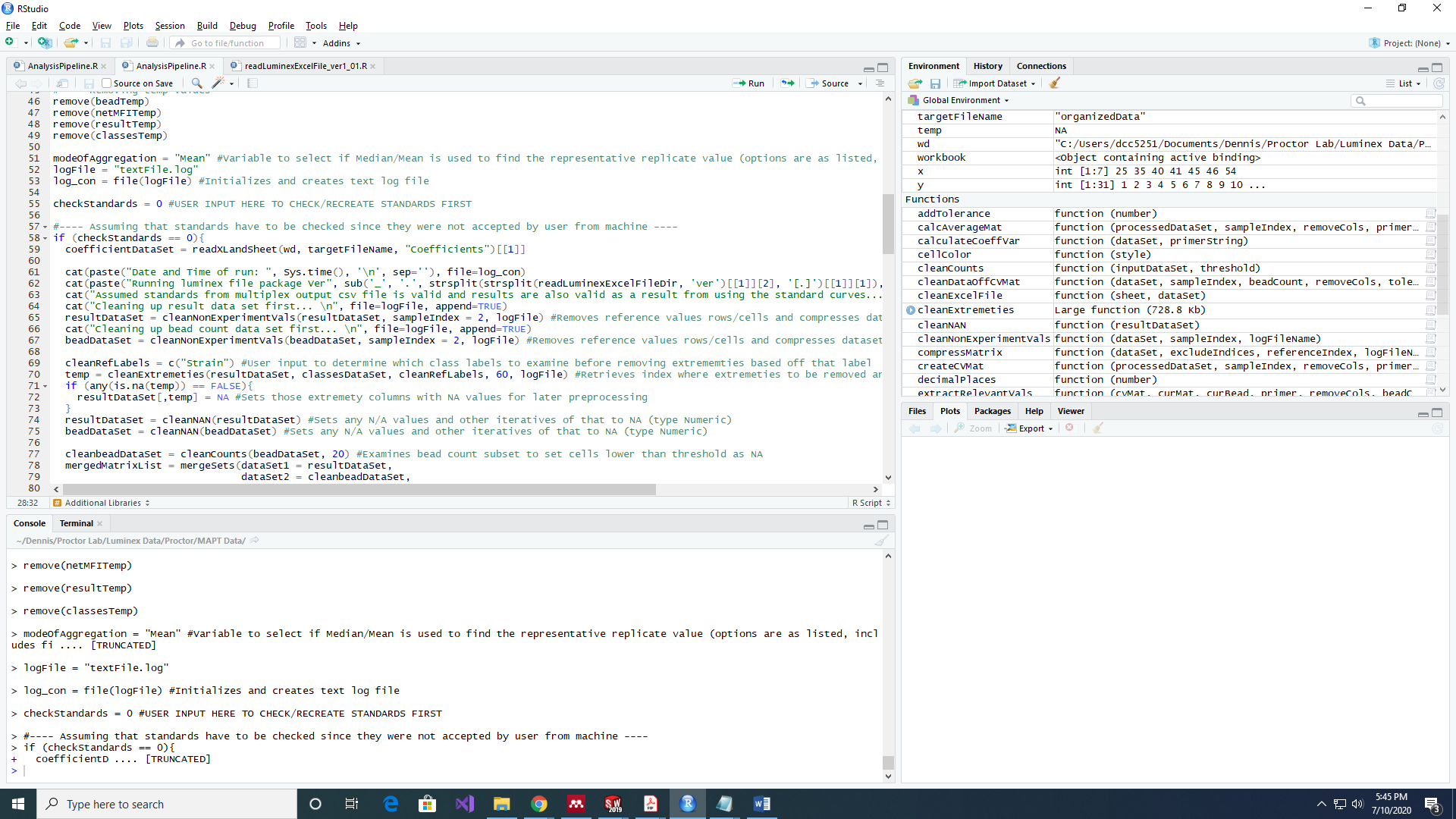
Navigate to the lines that set the variable *modeOfAggregation* and *logFile* as shown below. 

The variable *modeOfAggregation* allows the user to set either Mean or Median as the means of calculating the representative value from cleaned data (i.e. mean will result in outputting the mean concentration whilst median will result in outputting the median concentration). Variable options are Mean or Median (include first letter capital). The *logFile* variable allows the user to set a name for the log file that details the processes of the auto-clean script*.* **Note that the user must include the .log at the end of the file name as shown in the picture above.**

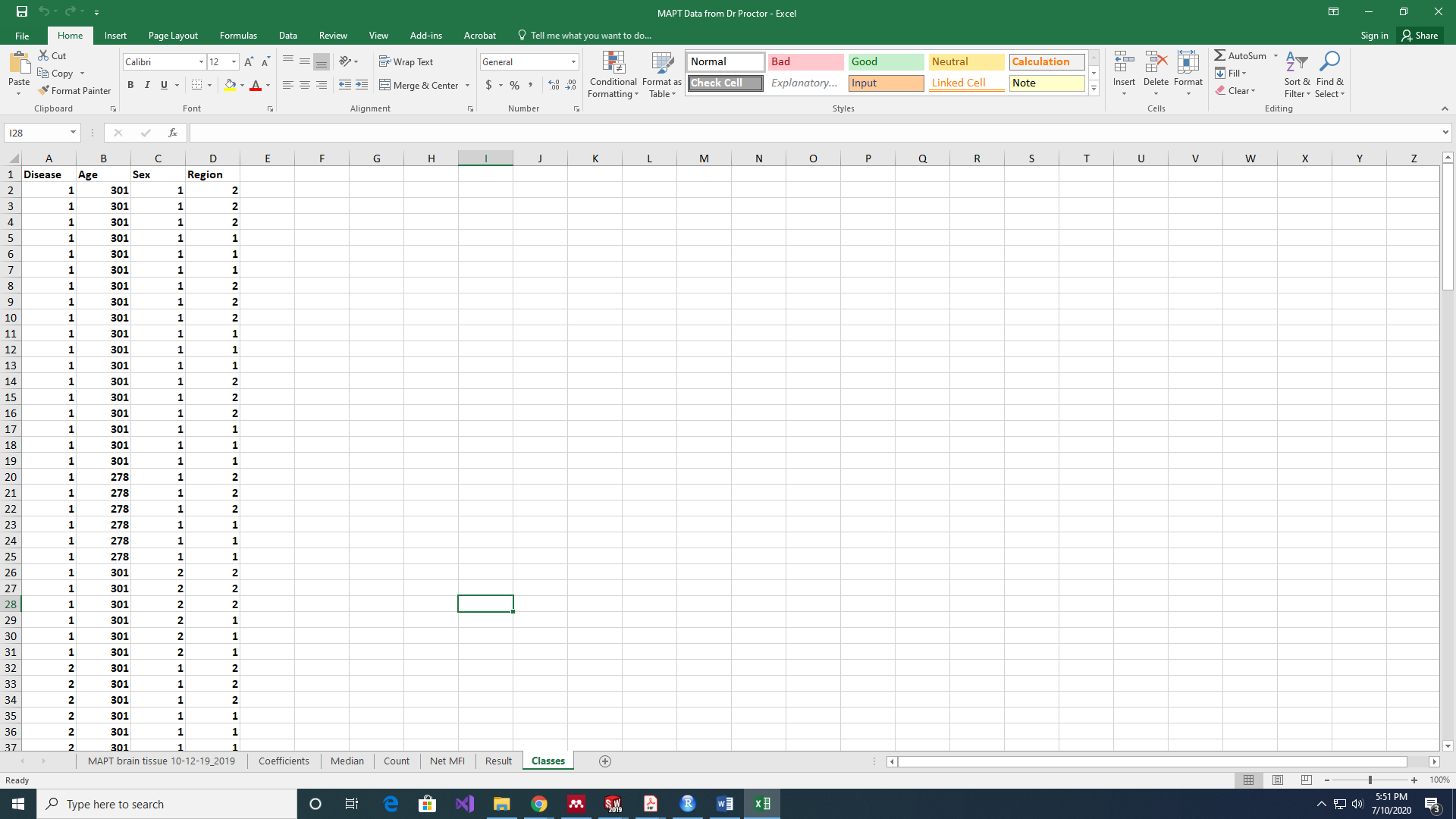
Another variable is the numeric variable at line 75 (shown below). The value 20 indicates that the threshold for determining if a well had sufficient bead counts was at least 20 beads, any well that had bead counts lower than the input (in this example it is 20) will be excluded from the calculation of the mean/median value of the technical triplicate. This value can be changed to the user’s preference.



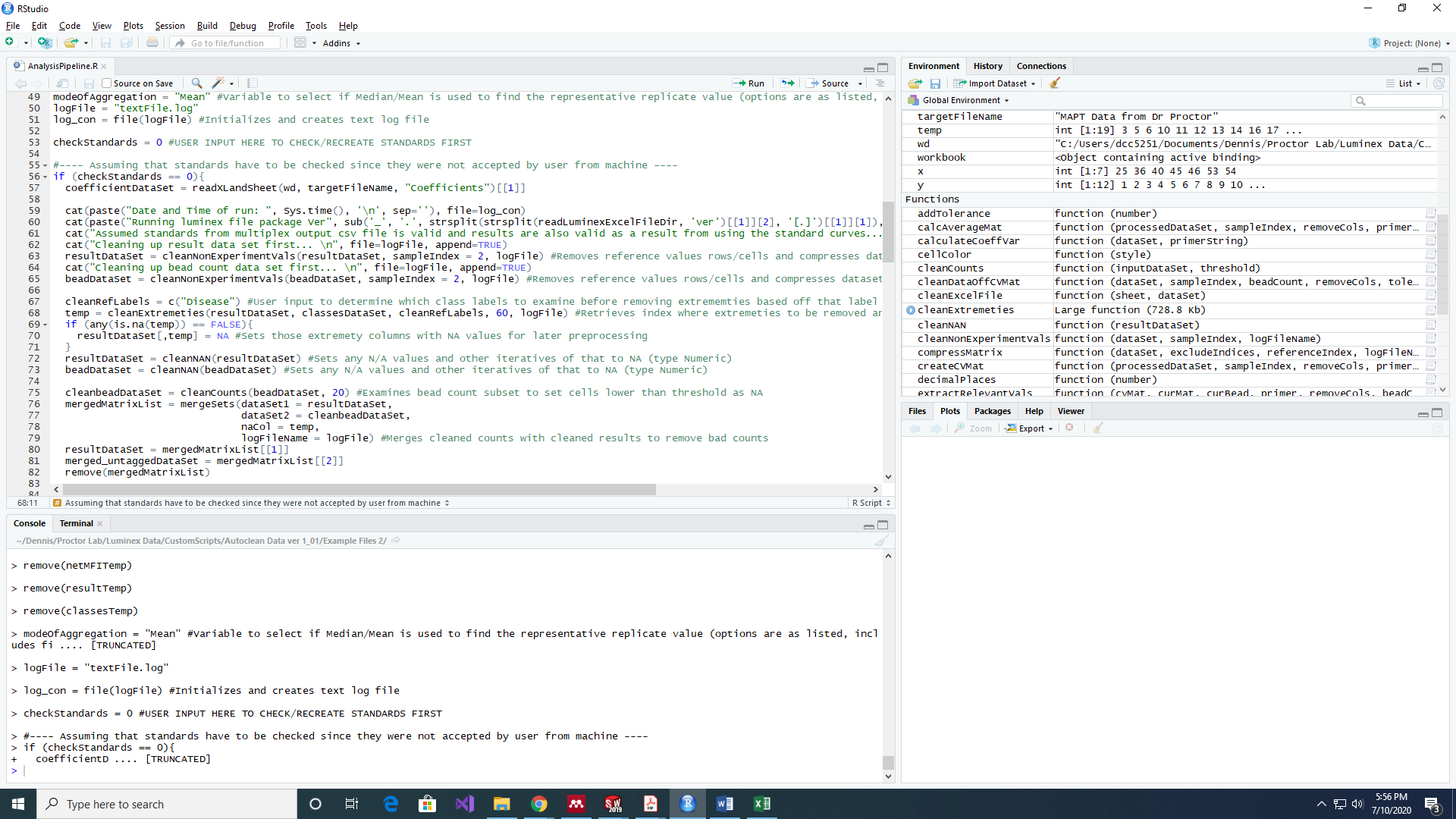
Last variable is found here in the line with the variable *cleanRefLabels*.



In order to determine a relevant value for the *cleanRefLabels*, navigate to your Classes sheet in the raw luminex excel file. In example file 2, there are four columns in the Classes sheet as shown:



The classes here are Disease, Age, Sex, and Region. The user can select one as the defining choice for cleaning data. This value is chosen since the auto-cleaning script determine which variables (columns) to discard based off if there are over 60% of lower/upper extreme values with respect to each class. For example, let’s say for variable A, over 65% of rows have concentrations that exceed the detectable range of the standard curve, and so all are shown to have the same value. This results in a relatively weak contributing factor to the model as there is little to no variation across all observations given that over 60% of observations for that class do not change. **The user can set the threshold by navigating to the following line shown:**

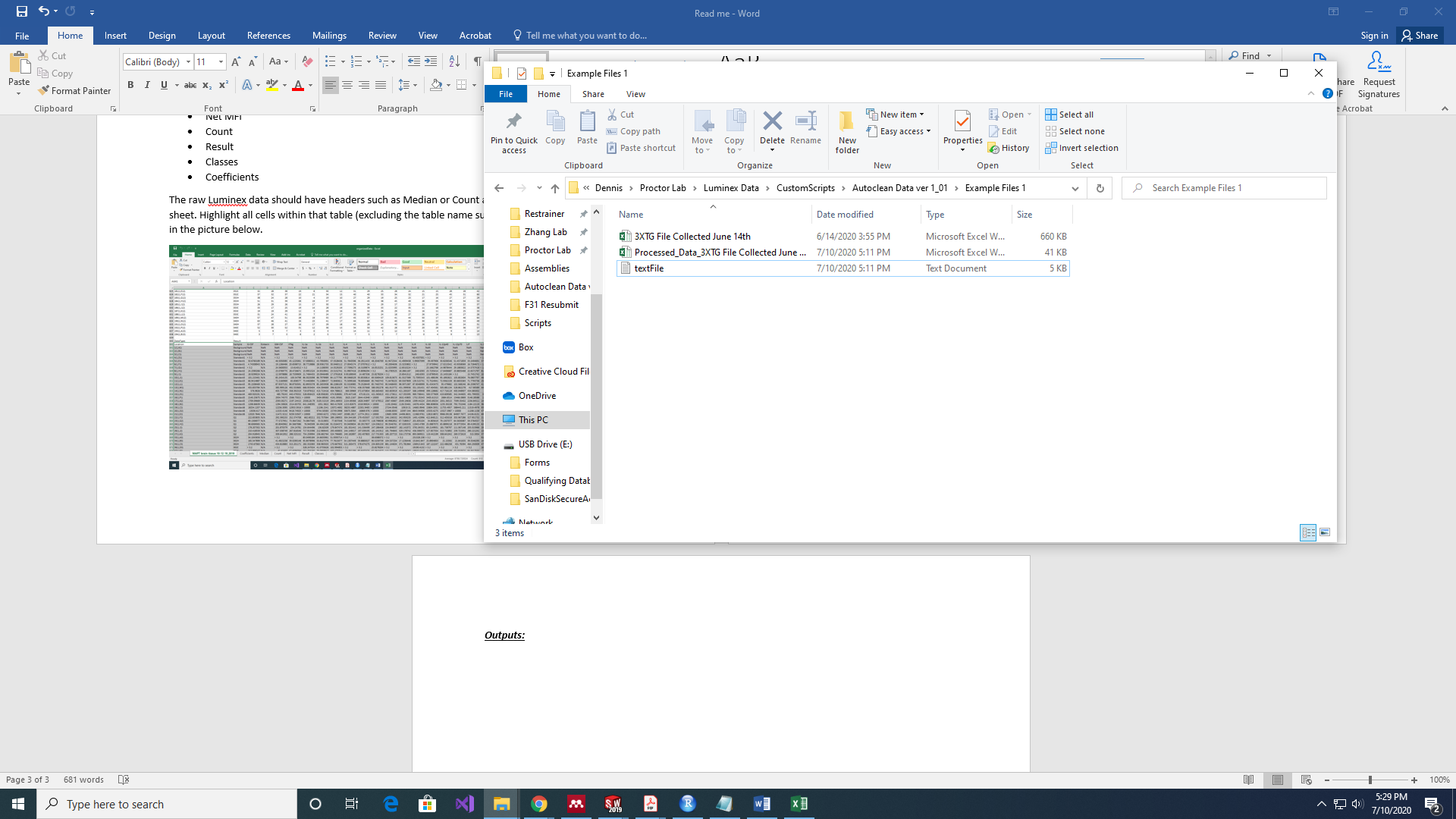


The user can change the number 60 to whatever percentage desired as the threshold for tossing, anything exceeding that value sets the variable to be discarded based off the criteria set by *cleanRefLabels*.

If the user desires not to use the *cleanExtremeties* function and have the pipeline remove columns based off upper/lower limit percentages, the user can input a ‘fake’ *cleanRefLabels* value instead. To give context, our example has four class labels the user can refer to if they desire to have the pipeline remove columns that had a certain proportion of values at either extremities within a specific class. Here, those would be *Disease, Age, Sex,* and *Region.* If the user here desires not use any classes and not use the function, the user will input a fake class, for example *State*. As *State* is not a valid class, the function will not remove any columns.

Thus the user should have now modified or agreed with the default settings of 8 variables. Once done, the user can run the script and the raw luminex file will be cleaned. This is only applicable if the standards setting is set to 0 as the pipeline only accounts for when the user agrees with the original standard curve. Thus the assumption is that the background values are not wrong.

***Outputs:***



As shown above, after running the script, there should be three files **(assuming this was not done in the example file directory and that the user only started with the raw luminex excel file first).** The first file here (3XTG File Collected June 14th) is the example raw luminex file, whilst the other two are the processed file and the log file respectively. All excel outputs will have the beginning name ‘Processed\_Data\_’ follow by the raw file name. The log file will retain the name inputted by the user for the *logFileName* variable.

The cleaned excel file should have four sheets in it:

* ProcessedData
* CleanedData\_ObsOnly
* CleanedData\_VarsOnly
* UncleanedData

ProcessedData sheet has the output where all numeric values (excluding column and row headers) have been cleaned, but cells that possess a string value (e.g. Low Bead Count) will notify the user that no representative value was found for that observation due to bad bead counts across all technical replicates.

CleanedData\_ObsOnly sheet has the output where only observations (rows) that had valid numeric values across all variables (columns) were retained and placed in the table. (**ONLY SHOWS DATA IF THERE WERE LOW BEAD COUNTS ACROSS ALL TECHNICAL REPLICATES FOR THE SAME OBSERVATION. IF THE DATASET HAS NO LOW BEAD COUNTS, THEN IT WILL BE EMPTY)**

CleanedData\_VarsOnly sheet has the output where only variables (columns) that had valid numeric values across all observations (rows) were retained. (**ONLY SHOWS DATA IF THERE WERE LOW BEAD COUNTS ACROSS ALL TECHNICAL REPLICATES FOR THE SAME OBSERVATION. IF THE DATASET HAS NO LOW BEAD COUNTS, THEN IT WILL BE EMPTY)**

UncleanedData sheet is in fact a **cleaned** sheet, and is similar to the ProcessedData sheet, with the exception where cells that had a string value are now replaced by a representative value where a mean/median (dependent on modeOfAggregation selection) was calculated across all low bead count replicates and used as the representative value. As shown in the Processed Data excel sheet in the Example File 2, the user will notice a number of cells highlighted as well in this UncleanedData sheet. The color meanings are explained below:

* Red = All replicates had unsatisfactory bead counts
* Orange = 1/2 Unsatisfactory Bead Counts in replicate subset, used the remaining satisfactory
* Yellow = Outlier found in replicate subset, used the remaining and calculated representative value

***Example Files:***

Example files are listed within 2 file directories. Both example file directories have 5 files:

* Raw Luminex Excel File
* Mean cleaned Excel file
* Median cleaned Excel file
* Log file for mean cleaned Excel file
* Log file for median cleaned Excel file

The first example (Example File 1) contains an excel file that had really good bead counts across replicates, and thus did not require much removal. As such in the cleaned file, the two ‘cleaned sheets’ (CleanedData\_ObsOnly and CleanedData\_VarsOnly) will not have any numeric values and are essentially empty. The second example (Example File 2) instead has quite the number of replicates with bad counts (across all three technical replicates for the same observation). And so, the cleaned file, the ‘cleaned sheets’ will have numeric values present.