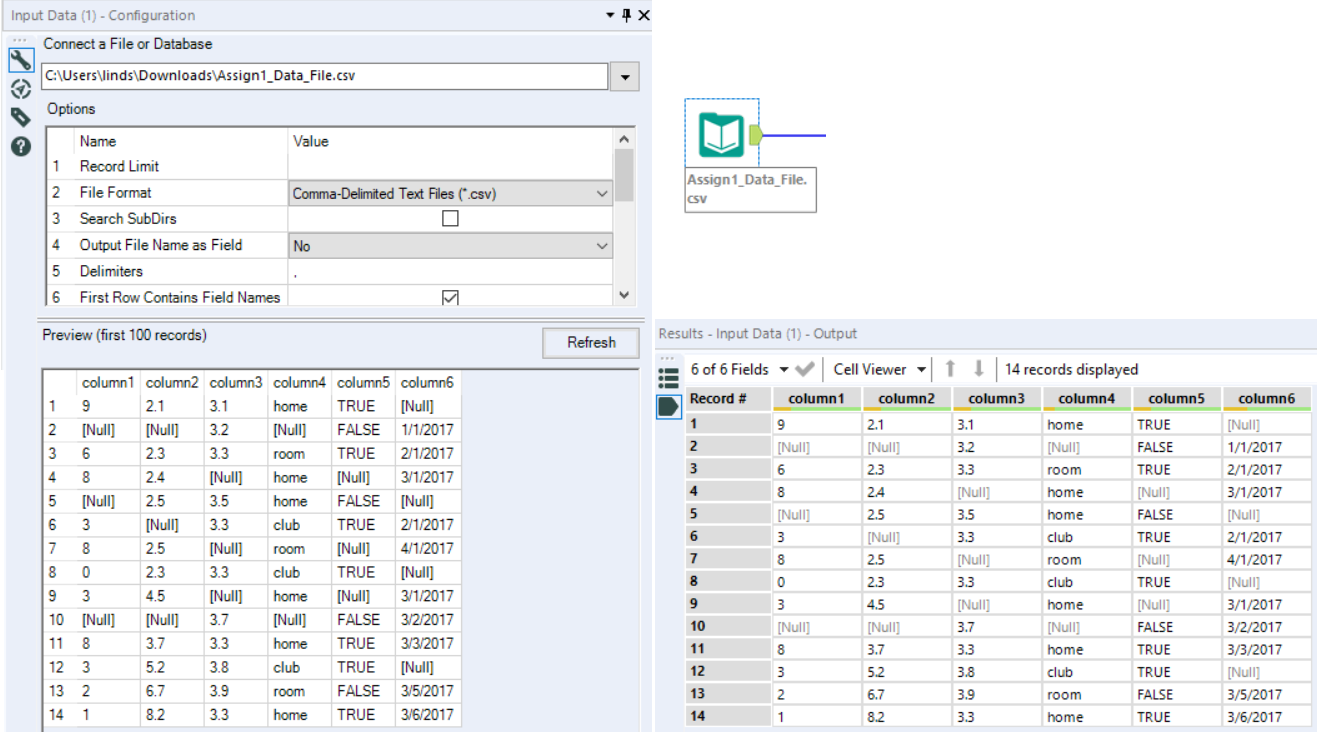
Lindsay Sheppard

DSC 609: Machine Learning

Programming Assignment 1

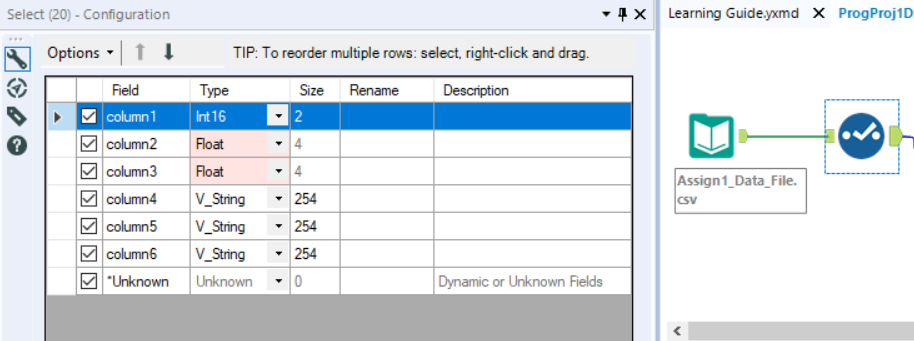
**Data Overview**

The file Assign1\_Data\_File.csv was the source of the data for the current project. The file was downloaded and then ‘drag-and-dropped’ into Alteryx Designer version 2018.3.5.52487, which cleaned the data as described. Both the Data Preview and Results window of the Input Data tool showed the uncleaned, initial data (Figure 1). The data contained six columns (*column1-6*). Columns 1 through 3 contained floats, or decimal values. The fourth column consisted of either “home”, “room”, or “club”, the fifth column were Boolean values (True or False), and the sixth column were dates from January to April 2017.



**Figure 1:** Initial data set as viewed in Alteryx’s Input Data Results window

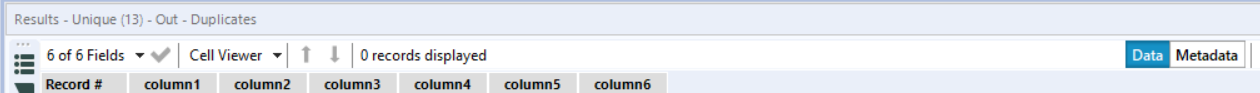
When Comma-Separated Values (CSV) format files are uploaded into Alteryx, the program defines all data as strings automatically. This can be solved two ways: by using the Auto-Field tool, which will automatically assign data types, or, by using the Select tool, which allows the user to choose data types for each field. The Select tool provides the advantage of allowing the user to rename and select or deselect out various columns as well. The Select tool was used to change column 1 to “Int16”, columns 2 and 3 to “Float”, and columns 4-6 to “V\_String”. Column 5 remained a string so that the tools would not correct the TRUE/FALSE values to 1s and 0s. Figure 2 provides a visual of the Select Tool configuration.



**Figure 2:** Select tool configuration window

**Duplicate Records**

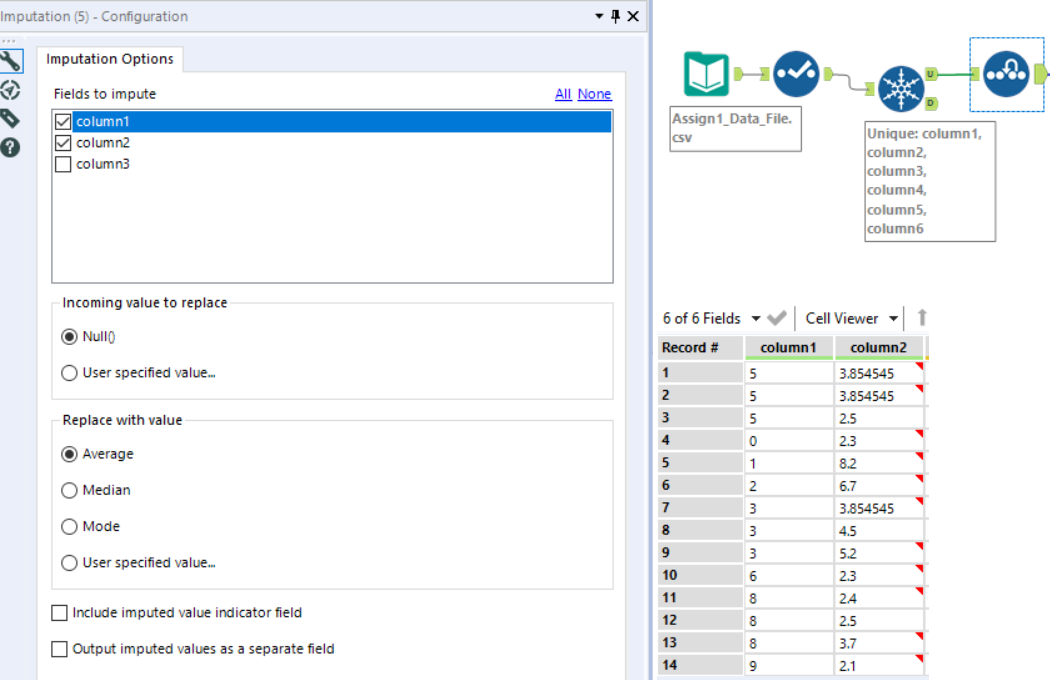
By selecting “all” in the Unique Tool, one can check the entire dataset for duplicates. However, there were no duplicate rows; all fourteen unique rows were found in the “U” output of the tool, and no duplicates were sorted into the “D” output (Figure 2). If there had been duplicate rows, they could have been filtered out this way or using a Select Records tool.



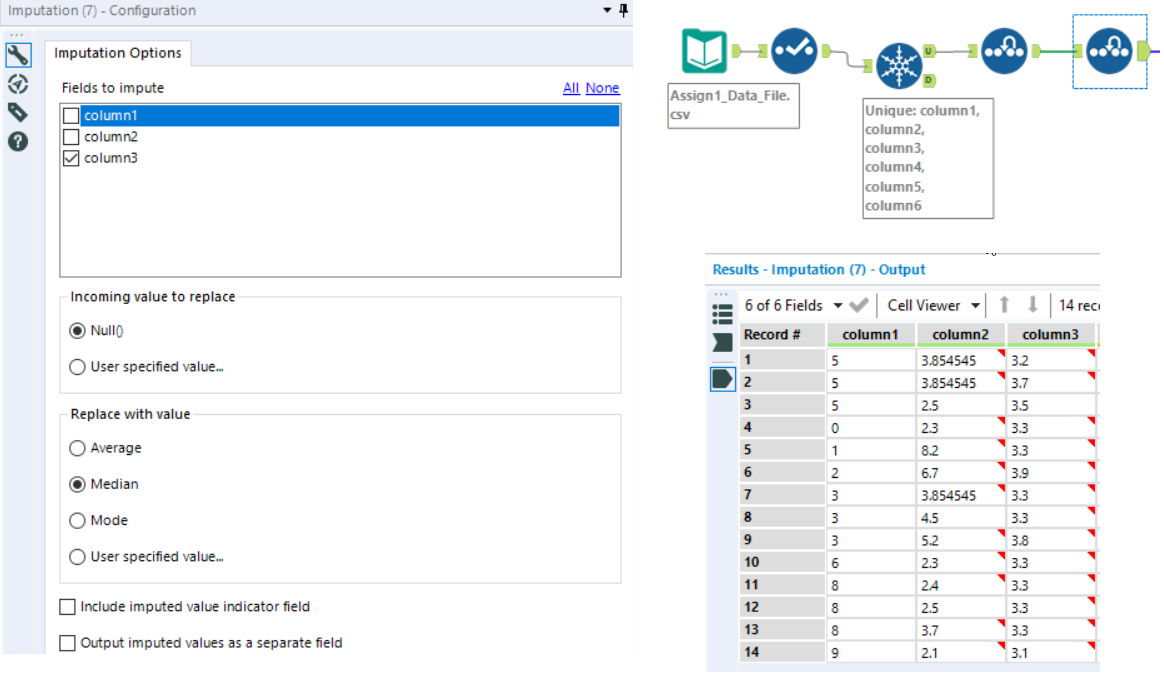
**Figure 3:**  Unique tool “Out” output

**Missing Data**

Columns 1 and 2 contained null values that were replaced with the respective mean of each column. This was handled easily using the Imputation tool. The tool allows the user to select one or more numeric columns and replace either nulls or a specified value with the average (mean), median, mode, or other specified value where necessary. For column 1, the imputed mean was “5”; for column 2, the mean was approximately “3.8545”. Column 3 replaced null values the same way but used a calculated median of 3.3 instead. The Imputation configuration window and the results are shown in Figures 4 and 5.



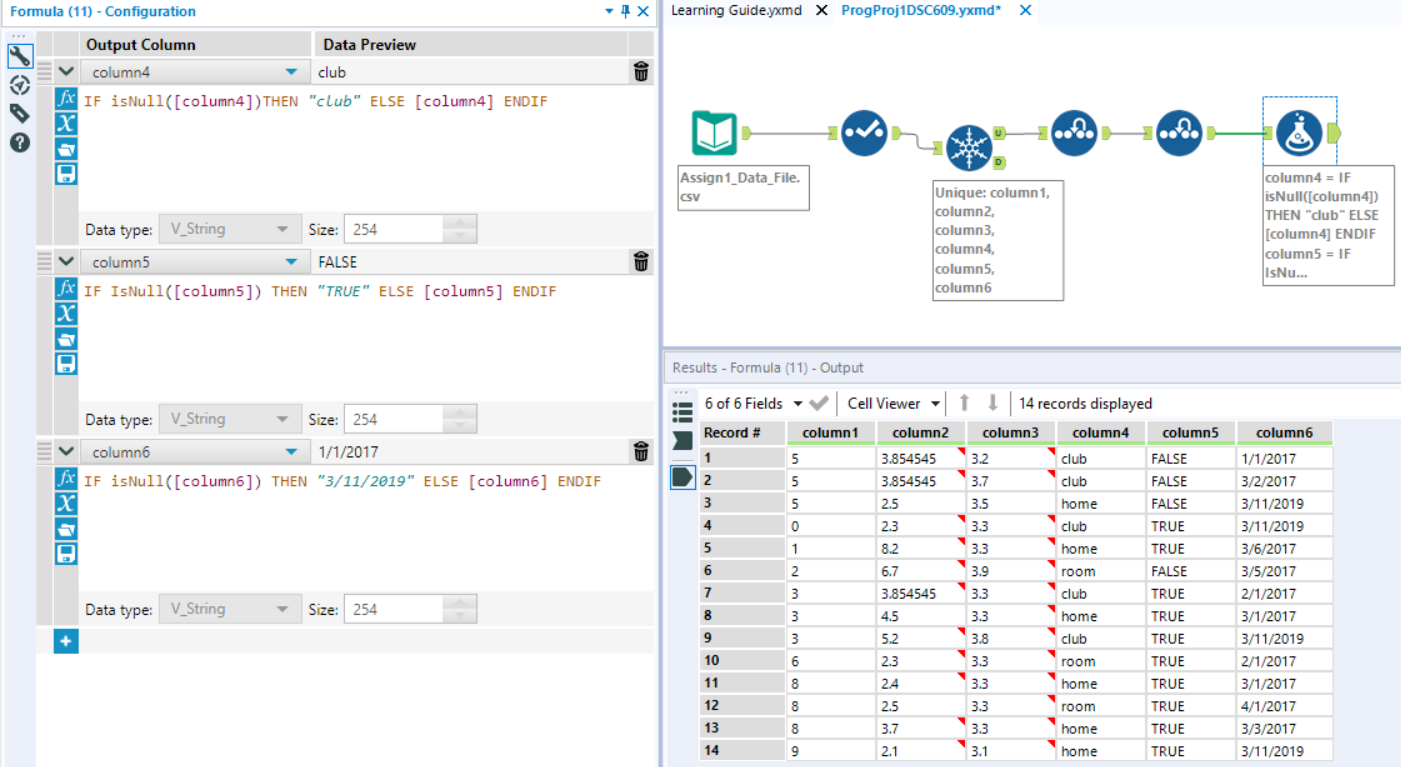
**Figure 4:** Imputation tool workflow, configuration, and output for Columns 1 and 2



**Figure 5:** Imputation of median, Column 3

For the columns with string type data, the Imputation tool does not work; the Formula tool was used instead. The null values were replaced with the string ‘club’ in the fourth column, “TRUE” in the fifth column, and “3/11/2019” in the sixth column. Instructions for the dataset determine the ‘club’ value for missing data in Column 4, but “TRUE” was chosen for Column 5 because the already present values of 3.3 in Column 3 corresponded to a value of “TRUE” in Column 5; the missing values in Column 3 that were converted to the value 3.3 were also missing in Column 5. The data cleaning date was chosen to replace null values in Column 6.

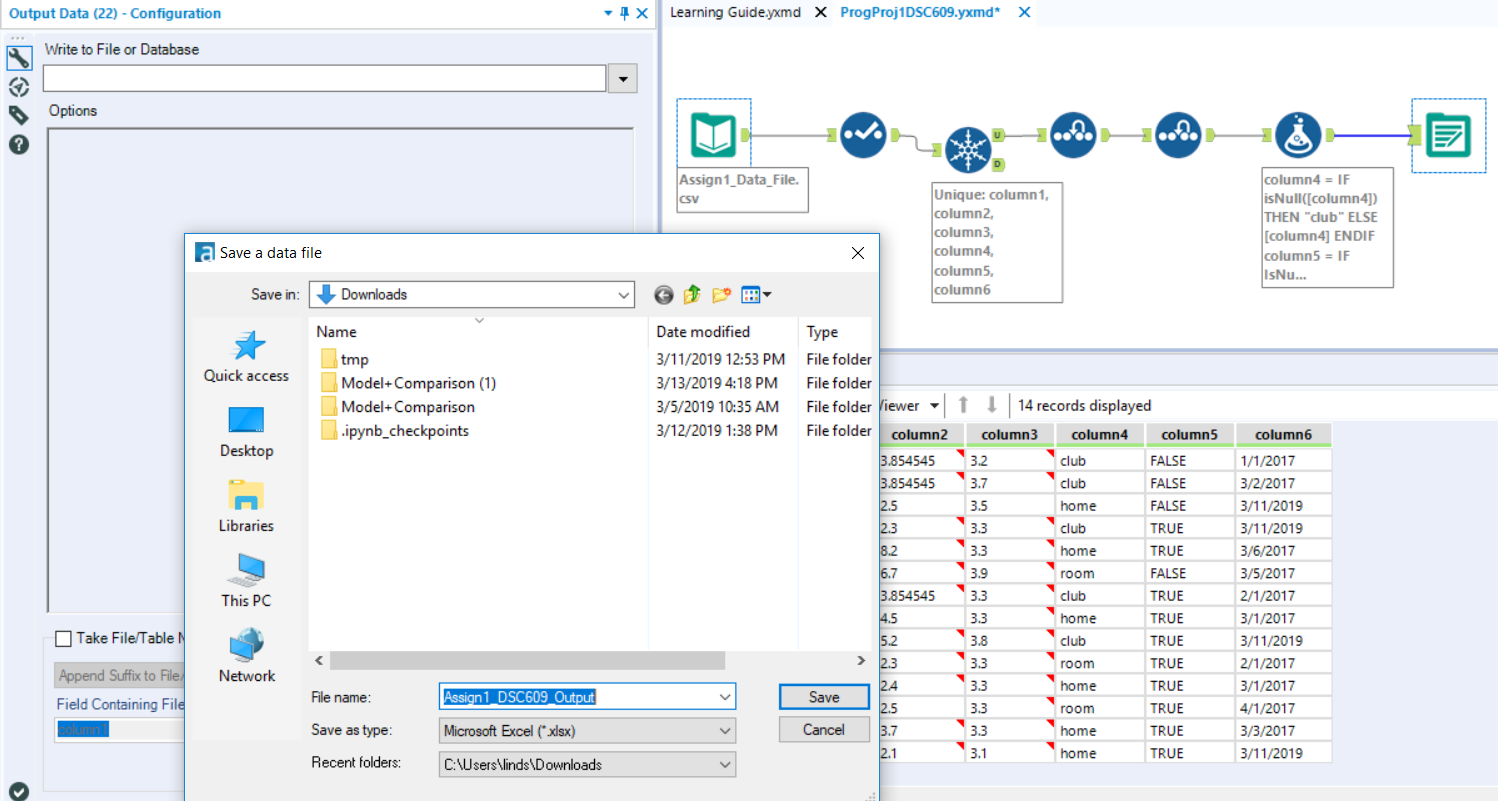
Unlike the Imputation tool, which requires multiple tools for different actions on each column, multiple formulas may be written into a single Formula tool. The Formula tool may also be used instead of the Imputation tool as desired. Three if-else statements were used to replace the missing values in each column. These are shown in Figure 6, along with the workflow and the resulting output data.

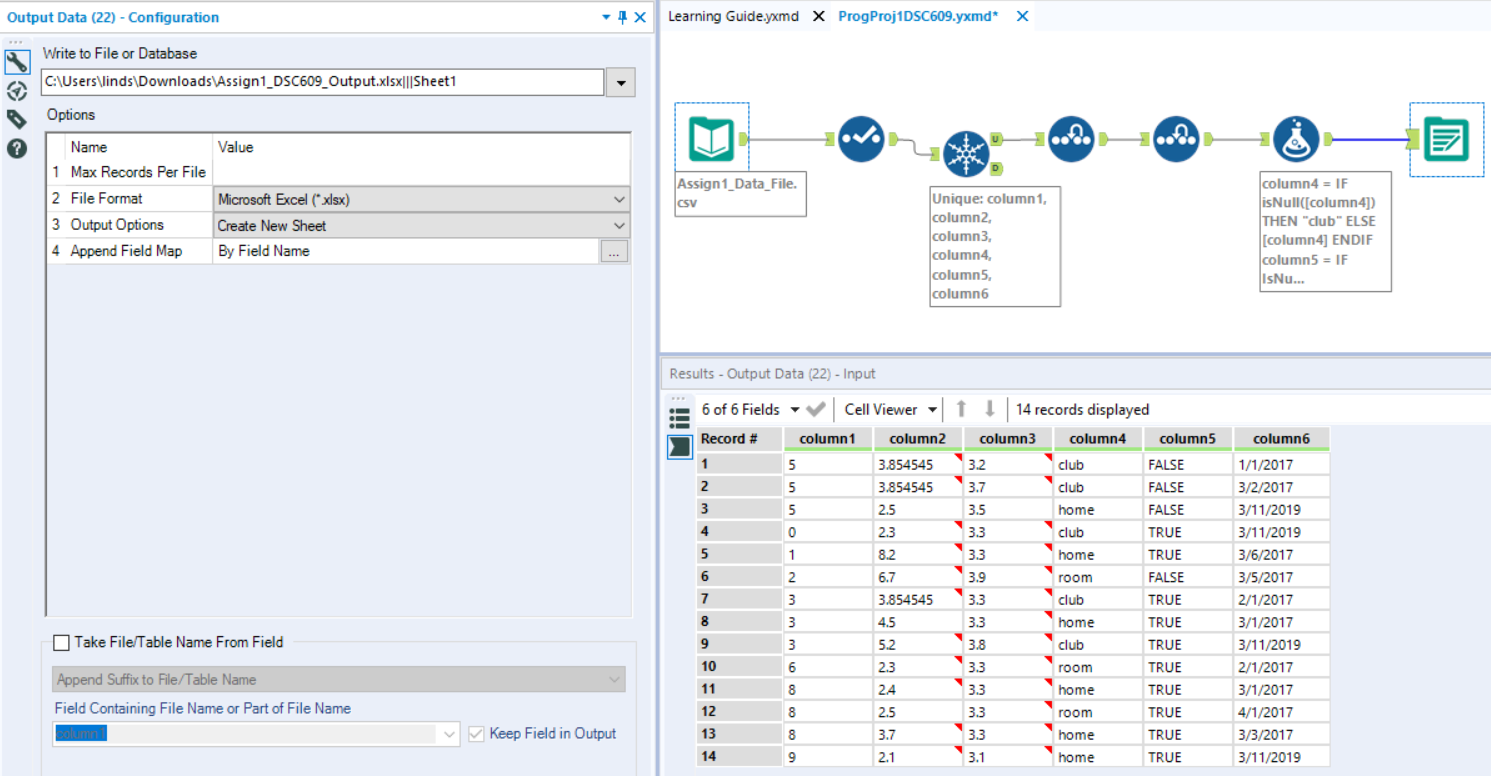


**Figure 6:** Formula tool expressions in the configuration window, workflow, and output

**File Saving**

To save the cleaned data for later use, the Output Data tool was used. Upon dragging and dropping the tool onto the workflow canvas, the tool configuration window was visible. Clicking the drop-down menu at the “Write to File or Database” and selecting ‘File...’ initialized a save window, where a user can choose to save the file in several different formats on his or her local drives. The file was saved as “Assign1\_DSC609\_Output” and saved as a Microsoft Excel (.xlsx) file in the current author’s Downloads file (*C:\Users\linds\Downloads\Assign1\_DSC609\_Output.xlsx|||Sheet1*). Figure 7 displays the process and the final workflow.



****

**Figure 7:** Saving the cleaned file; final workflow