4.3.5

Activity: Strings and Numbers

Maria has requested a grade-level analysis of the schools. But to provide that analysis, you need to first properly prepare the data. So, Maria asks you to ensure that the data types are appropriate before you dive into the analysis.

In this activity, you'll prepare and clean data by removing string characters from a column in the dataset and by converting data types.

Files

Download the following files to help you get started:

Strings and Numbers files (https://2u-data-curriculum-team.s3.amazonaws.com/dataviz-online/v2/module_4/4-3-Student_Data_Starter_Code.zip)

Using the Jupyter Notebook file in the `Unsolved` folder, navigate to the "Step 2" section to write your code. Alternatively, you can use the Jupyter Notebook file that you created in the previous lesson to write your code.

Instructions

1. Check for any `NaN` values in the data, as the following code shows:

```
student_df.isna().sum()
```

2. Drop any rows that contain `NaN` values, and then verify that no `NaN` values remain, as the following code shows:

```
student_df = student_df.dropna()
student_df.isna().sum()
```

3. Check for any duplicated rows in the data, as the following code shows:

```
student_df.duplicated().sum()
```

4. Drop the duplicated rows, and then verify that they have been dropped, as the following code shows:

```
student_df = student_df.drop_duplicates()
student_df.duplicated().sum()
```

5. Check the data types of the columns (paying special attention to the "grade" column), as the following code shows:

```
student_df.dtypes
```

6. Visually inspect the "grade" column to identify why it's type is `object` rather than `int`, as the following code shows:

```
student_df['grade']
```

7. Note the suffixes in the grade numbers (for example, "11th" instead of "11"), and use the `str.replace` function to remove them. Then review the column to verify the removal, as the following code shows:

```
student_df['grade'] = student_df['grade'].str.replace('th', '')
student_df['grade']
```

8. Change the type of the “grade“ column to int by using `astype`, and then verify the change by using `dtypes`, as the following code shows:

```
student_df['grade'] = student_df['grade'].astype(int)
student_df.dtypes
```

9. Preview the data by using `head` to ensure that the data is correct, as the following code shows:

```
student_df.head()
```

Solution

How did you do?

You can refer to the solution file in the `Solved` folder, which is in the zipped folder that you downloaded for this activity.

What’s Next?

You now know how to clean dirty data for an analysis. Your datasets are free of missing and duplicated values, and the data types are correct and ready for calculations. But, what do you do with these pristine datasets? You’ll next start to analyze them!