DataEng: Data Validation Activity

Make a copy of this document and use it to record your results. Store a PDF copy of the document in your git repository along with any needed code before submitting for this week.

High quality data is crucial for any data project. This week you'll gain some experience and knowledge of analyzing data sets for quality.

The data set for this week is a listing of all Oregon automobile crashes on the Mt. Hood Hwy (Highway 26) during 2019. This data is provided by the Oregon Department of Transportation and is part of a larger data set that is often utilized for studies of roads, traffic and safety.

Here is the available documentation for this data: description of columns, Oregon Crash Data Coding Manual

Data validation is usually an iterative three-step process. First (part A) you develop assertions about your data as a way to make your assumptions explicit. Second (part B) you write code to evaluate the assertions and test the assumptions. This helps you to refine your existing assertions (part C) before starting the whole process over again by creating new assertions (part A again).

Submit: In-class Activity Submission Form

A. Create Assertions

Access the crash data, review the associated documentation of the data (ignore the data itself for now). Based on the documentation, create English language assertions for various properties of the data. No need to be exhaustive for this assignment, two or more assertions in each category are enough.

1. Create 2+ *existence* assertions. Example, "Every record has a date field".
   Answer:
   [1] 44% of the records have a 'Participant ID' attribute.
   [2] 81% of the records have a 'Vehicle ID' attribute.
   --- Part D: Iterate --
   [3] Every record has a 'Record Type' attribute.

2. Create 2+ *limit* assertions. The values of most numeric fields should fall within a valid range. Example: "the date field should be between 1/1/2019 and 12/31/2019 inclusive"
   Answer:

[1] The Week Day Code data field should be between 1 and 7.
[2] The County Code data field should be between 1 and 36.
--- Part D: Iterate ---
[3] The Collision Type data field should be between 0 and 9.

3. Create 2+ *intra-record check* assertions.
   Answer:
   [1] Week Day Code -1 = Crash Month + Crash Day + Crash Year
   Based on the formula we can get the week day in a week.
   [2] The Crash Severity is related to the Weather Condition.
   --- Part D: Iterate ---
   [3] The City Section ID is related to the County Code.

4. Create 2+ *inter-record check* assertions.
   Answer:
   [1] The Crash Hour at 14 should not be more than 10%.
   [2] The Urban Area Code is 57 should be more than 70%.
   --- Part D: Iterate ---
   [3] The County Code is 26 should be more than 50%.

5. Create 2+ *summary* assertions. Example: “every crash has a unique ID”
   Answer: Often uniqueness constraints
   [1] Every Vehicle has a unique ID.
   [2] Every Participant has a unique ID.
   --- Part D: Iterate ---
   [3] Not every crash has a unique crash serial number.

6. Create 2+ referential integrity insertions. Example “every crash participant has a Crash ID of a known crash”
   Answer:
   [1] Every crash participant has a Vehicle ID of a known vehicle.
   [2] Every crash vehicle has a Crash ID of a known crash.
   --- Part D: Iterate ---
   [3] Every crash has a Serial number of a known crash.

7. Create 2+ *statistical distribution assertions*. Example: “crashes are evenly/uniformly distributed throughout the year.”
   Answer: Such values should behave in such a statistical manner
   [1] No matter what the weather conditions are, crashes would happen.
   [2] Crashes can happen in the 24 hours every day.

--- Part D: Iterate ---
[3] No matter what the road surface conditions are, crashes would happen.

B. Validate the Assertions

1. Now study the data in an editor or browser. If you are anything like me you will be surprised with what you find. The Oregon DOT made a mess with their data!
2. Write python code to read in the test data and parse it into python data structures. You can write your code any way you like, but we suggest that you use pandas' methods for reading csv files into a pandas Dataframe
   Answer:
   Done

3. Write python code to validate each of the assertions that you created in part A. Again, pandas makes it easy to create and execute assertion validation code.
   Answer:
   Done
   --- Part D: Iterate --
   Done

4. If you are like me you'll find that some of your assertions don't make sense once you actually understand the structure of the data. So go back and change your assertions if needed to make them sensible.
   Answer:
   Done
   --- Part D: Iterate --
   Done

5. Run your code and note any assertion violations. List the violations here.
   Answer:
   Before I wrote: The Urban Area Code is 57 should be more than 90%.
   After I wrote: The Urban Area Code is 57 should be more than 70%. (After use my code to run to get the result)
   --- Part D: Iterate ---
   The City Section ID is related to the County Code.
   After I used the python script to get the City Section ID set and County Code set, I found some cities are not related to the County. There are three counties in the dataset and they are Multnomah, Clackamas and Hood River. For example, the county 'Greenhorn', I googled it to find this city is located in both Grant County and Baker County.

C. Evaluate the Violations

For any assertion violations found in part B, describe how you might resolve the violation. Options might include “revise assumptions/assertions”, “discard the violating row(s)”, “ignore”, “add missing values”, “interpolate”, “use defaults”, etc.

No need to write code to resolve the violations at this point, you will do that in step E.

If you chose to “revise assumptions/assertions” for any of the violations, then briefly explain how you would revise your assertions based on what you learned.

Answer:

Before I wrote: The Urban Area Code is 57 should be more than 90%.
After I wrote: The Urban Area Code is 57 should be more than 70%. (After use my code to run to get the result)
--- Part D: Iterate ---
Before I wrote: The City Section ID is related to the County Code.
After I wrote: Some of the City Section ID is related to the County Code.
(I get the City Section ID and County Code after running my python script code. I used the Crash Analysis and Coding Manual to find the corresponding county name and city name. Then, I found some city names not related to the county names.)

D. Learn and Iterate

The process of validating data usually gives us a better understanding of any data set. What have you learned about the data set that you did not know at the beginning of the current ABCD iteration?

Answer:

I have learned there are some kinds of assertions. When I need to write two more existence assertions, I read the slides and watch the class video to analyze what the existence assertion is. In addition, I also watch the class video and slides to learn what the limit assertions, intra-record check assertions, inter-record check assertions, summary assertions, referential integrity insertions and statistical distribution assertions are. I also read the dataset to analyze. To be honest, this dataset is so big and I have never seen it before. I spend a lot of time analyzing this dataset. I am looking at the dataset and looking at the crash analysis at the same time. Fortunately, I gradually understand the logic of this dataset.

I valid and evaluate my assertions to be more correct and reasonable. In addition, I learn more about the python pandas than before.

Next, iterate through the process again by going back to Step A. Add more assertions in each of the categories before moving to steps B and C again. Go through the full loop twice before moving to step E.

Answer:

Done

E. Resolve the Violations

For each assertion violation found during the two loops of the process, write python code to resolve the assertions. This might include dropping rows, dropping columns, adding default values, modifying values or other operations depending on the nature of the violation.

Note that I realize that this data set is somewhat awkward and that it might be best to “resolve the violations” by restructuring the data into proper tables. However, for this week, I ask that you keep the data in its current overall structure. Later (next week) we will have a chance to separate vehicle data and participant data properly.

E. Retest

After modifying the dataset/stream to resolve the assertion violations you should have produced a new set of data. Run this data through your validation code (Step B) to make sure that it validates cleanly.

Submit: In-class Activity Submission Form