DataEng S24: Data Validation Activity

Name: Monika Kamineni
Psu id: 920433615
Email: kamineni@pdx.edu

High quality data is crucial for any data project. This week you'll gain experience with validating a real data set provided by the Oregon Department of Transportation.

Due: this Friday at 10pm PT
Submit: 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 using the in-class activity submission form.

A. [MUST] Initial Discussion Question

Discuss the following question among your working group members at the beginning of the week and place your own response(s) in this space. Or, if you have no such experience with invalid data then indicate this in the space below.

Have you ever worked with a set of data that included errors? Describe the situation, including how you discovered the errors and what you did about them.

Response: The below are my team members experiences

Sriram:
Unreadable non-Ascii characters. Solution was to rewrite the characters. Other situations, there were cases where entries had values outside of acceptable limits. Those entries were removed.

Alex:
Within this project for DataEng, fields were marked as None when pulling from pubsub. Json module doesn't parse python None objects. Solution was to rewrite None as 'None' string.

Kirk:
Wafer/die data from Intel, a lot of die had empty values or -999 value for certain parameters. I removed those entries.

Monica: I worked for my UG project where the data had missing values,outliers and inconsistencies which required me to use outlier detection,standardizations and validations for generating accurate segmentation which is reliable for my clustering project.

Background

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 multi-step process.

B. Create assertions about the data
C. Write code to evaluate your assertions.
D. Run the code, analyze the results
E. Write code to transform the data and resolve any validation errors

B. [MUST] 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. Develop one or two assertions in each of the following categories during your first iteration through the ABC process.

1. *existence* assertions. Example: “Every crash occurred on a date”
2. *limit* assertions. Example: “Every crash occurred during year 2019”
3. *intra-record* assertions. Example: “If a crash record has a latitude coordinate then it should also have a longitude coordinate”
4. Create 2+ *inter-record check* assertions. Example: “Every vehicle listed in the crash data was part of a known crash”
5. Create 2+ *summary* assertions. Example: “There were thousands of crashes but not millions”
6. Create 2+ *statistical distribution assertions*. Example: “crashes are evenly/uniformly distributed throughout the months of the year.”

These are just examples. You may use these examples, but you should also create new ones of your own.

Answer:

Existence Assertions:

It verifies that each recorded crash within the dataset is invariably associated with a specific date.

Limit Assertions:
It establishes that all crashes documented in the dataset occurred exclusively during the calendar year 2019.

Intra-record Assertions:
It ensures that for every crash record containing latitude coordinates, corresponding longitude coordinates are present, and vice versa.

Inter-record Check Assertions:
- It confirms that each vehicle listed in the dataset has indeed been involved in a documented crash.
- A minimum of 10 crashes were identified to be attributed to Road Surface Condition Type 99.

Summary Assertions:
-The aggregate count of recorded crashes within the dataset consistently amounts to thousands, effectively indicating the absence of millions of reported incidents.
-The analysis confirms that the number of crashes involving alcohol does not exceed the total count of recorded crashes.
-The dataset reveals that the total reported accidents on Highway Number 26 during the year 2019 surpass 15 incidents.

Statistical Distribution Assertions:
-Statistical analysis indicates a uniform distribution of crashes across the months of the year, signifying a consistent occurrence rate throughout various periods.
-There should be at least one recorded accident on weekends where alcohol was involved.
-At least 10% of all crashes should occur without indicators for School Zone or Work Zone

C. [MUST] Validate the Assertions

1. Study the data in an editor or browser. Study it carefully, this data set is non-intuitive!.
2. Write python code to read in the test data. You are free to write your code any way you like, but we suggest that you use pandas' methods for reading csv files into a pandas Dataframe.
3. Write python code to validate each of the assertions that you created in part A. The pandas package eases the task of creating data validation code.
4. If needed, update your assertions or create new assertions based on your analysis of the data.

Answer:

The code is submitted in the git

D. [MUST] Run Your Code and Analyze the Results

Answer:

Result

Passed: If a crash record has a latitude coordinate then, it should also have a longitude coordinate
Passed: Thousands of crashes happened but not millions
Passed: Number of crashes involving alcohol should not exceed the total number of crashes
Passed: Total accidents on Highway Number 26 in 2019 should be more than 15
Passed: At least 10 vehicles met Crash with Road surface type 99
Passed: At least 10% of total crashes occurred without School Zone or Work Zone indicators

In this space, list any assertion violations that you encountered:

- **Failed:** Every crash occurred on a date
- **Failed:** Every crash occurred during year 2019
- **Failed**: Every vehicle listed was part of a known crash
- **Failed:** Crashes are evenly distributed throughout the months of the year
- **Failed:** At least one accident occurred on weekends with 'Alcohol-Involved Flag'

For each assertion violation, describe how to resolve the violation. Options might include:

- revise assumptions/assertions
- discard the violating row(s)
- Ignore
- add missing values
- Interpolate
- use defaults
- abandon the project because the data has too many problems and is unusable

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

Answer:

Failed: Every crash occurred on a date:

There appear to be some missing or incorrect crash dates in the dataset. To rectify this issue, we could consider removing rows where the crash date is missing or nonsensical. Alternatively, we might attempt to estimate or fill in missing dates if feasible.

Failed: Every crash occurred during the year 2019:
It appears that there are entries in the dataset that do not pertain to the year 2019. To address this discrepancy, we should verify the data source or our assumptions to ensure that only crashes from 2019 are being considered. If necessary, we may need to exclude rows corresponding to other years.

Failed: Every vehicle listed was part of a known crash:
It is perplexing why certain vehicles are not associated with known crashes in the dataset. To investigate this issue further, we need to determine why these vehicles are not linked. This may involve redefining the criteria for a "known" crash or assessing the integrity of our data.

Failed: Crashes are evenly distributed throughout the months of the year:
The distribution of crashes across different months does not appear to be as uniform as anticipated. To address this discrepancy, we may need to reassess our assumption of an even distribution or investigate any anomalies in the data that could be causing this irregularity.

Failed: At least one accident occurred on weekends with an 'Alcohol-Involved Flag':
It is concerning that there are no recorded accidents involving alcohol on weekends. To resolve this issue, we should review the dataset to ensure that weekend accidents with alcohol involvement are accurately documented. If necessary, we may need to adjust our analysis criteria or enhance our data collection methods.

E. [SHOULD] Resolve the Violations and Transform the Data

For each assertion violation write python code to resolve the violation according to your entry in the “how to resolve” section above.

Output the validated/transformed data to new files. There is no need to keep the same, awkward, single file format for the data. Consider outputting three files containing information about (respectively) crashes, vehicles and participants.

F. [ASPIRE] 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 ABC iteration?

Next, iterate through the process again by going back through steps B, C, D and E at least one more time.