DataEng S22: Data Validation Activity

High quality data is crucial for any data project. This week you'll gain experience with validating a real data set.

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.

Initial Discussion Question - Discuss the following question among your working group members at the beginning of the week and place your responses 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 1: Jared- Never worked with data outside of school, so all the data was correct in school from memory

Response 2: Chiharu - machine learning images, some were incorrect. Didn't make any changes

Response 3: Deepa - data Pokeman database, some of the data was incorrect or missing, manually went through the data and manually changed it to have data

Response 4: Reeya - profit database, profit was in different types of currency, had to go through and manually change it into one type of currency

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.

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

Repeat this ABC loop as many times as needed to fully validate your data.

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. 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”
   Every crash data point has a record type
2. *limit* assertions. Example: “Every crash occurred during year 2019”
   Every record type is between 1 and 3
3. *intra-record* assertions. Example: “Every crash has a unique ID”
   Every crash happened on a date that exists
4. Create 2+ *inter-record check* assertions. Example: “Every vehicle listed in the crash data was part of a known crash”
   The total number of crashes any given month shouldn’t vary any more than 33% from the average.
5. Create 2+ *summary* assertions. Example: “There were thousands of crashes but not millions”
   Alcohol is involved with a lot of crashes, but not most. Weather condition is mostly cloudy or rainy
6. Create 2+ *statistical distribution assertions*. Example: “crashes are evenly/uniformly distributed throughout the months of the year.”
   Crashes happen more on certain days than other crashes. **The urban area with the most crashes is Portland, with code 57**

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

B. 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.

C. Run Your Code and Analyze the Results

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

- Some months were over 33% more crashes than the average month
- Only two urban area codes reported
- Most of the data did not have any problems

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: Need to revise assumptions. The assumption was that no month would exceed 33% difference from the average. Turns out that some months, particularly winter months, do have more than 33% difference from the average, but not 50%. So, the assumption is now that any given month should not be more than 50% above the average for all months.

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 ABC iteration?

Answer: I learned there's a lot of codes used to represent more general data. I learned that this organization is pretty good at not having errors in its data, at least none that I could manage to find that wasn't an error based on my own assumptions. I learned that crashes happen more on certain days and in certain months. I learned that some data is not filled out as well as other data. That there are only two urban area codes recorded.

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

E. 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.