DataEng S23: Data Validation Activity

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.

**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: I have worked with a dataset from kaggle which has a variety of values, data types and extra characters which are not related to the database and it was also a mix of 2-3 database tables and I started cleaning it by removing the unwanted data and noise.

Response 2: During my undergraduate I worked on data science project prediction of cancer data set is divided into 30% for testing and 70% for training. We found some errors in data including missing values, incorrect labels. For missing values we used mean, median methods and we removed the incorrect labels which makes the data set good for analysis.

Response 3: While coordinating with other teams , we really didn't know the significance of each data value they asked us to populate for them. We ended up populating null instead of 0, which caused a data mapping error on their end. So we had to run a daily script to populate 0 instead of null based on a few conditions till we sent the fix to production.

Response 4: Yes, I have. Our team noticed that, for a particular record, values did not match between two different tables where they were supposed to. Then we looked into it further and found out that it was not the only record, there were more. We informed the team which handles data of those tables and they took care of it.

The data set for this week is [a listing of all Oregon automobile crashes on the Mt. Hood Hwy (Highway 26) during 2019](https://drive.google.com/file/d/1A_R4rDgJsII7wL-onaPeodvv07rPk1SX). This data is provided by the [Oregon Department of Transportation](https://www.oregon.gov/odot) and is part of a [larger data set](https://tvc.odot.state.or.us/tvc/) that is often utilized for studies of roads, traffic and safety.

Here is the available documentation for this data: [description of columns](https://docs.google.com/spreadsheets/d/1G5MV073INT8wSFCzP-sWcPwfiATJpbpj), [Oregon Crash Data Coding Manual](https://www.oregon.gov/ODOT/Data/documents/CDS_Code_Manual.pdf) Data validation is usually an iterative three-step process.

1. Create assertions about the data
2. Write code to evaluate your assertions. .
3. 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:

At Least one crash for every month in a year. (success)

1. *Limit* assertions:

Nearly 40 Crashes on highway 26 happened in rainy conditions. (success)

1. *Intra-record* assertions:

If there is a vehicle ID then there will be a crash ID.(success)

1. Create 2+ *inter-record check* assertions:(success)

All Participants involved in the crash will have Crash ID and the Crash Date.

Location of each crash will have Crash ID and that will be associated with the county.

Vehicles that are involved in a crash will have a crash ID and one or more vehicle type ID.

1. Create 2+ *summary* assertions:(success)

Number of Crashes which occurred at intersections will be in the hundreds.

Crashes involving drunk and driving will be more than 5%.

Hit and Run cases will be more than 60% in the vehicle dataset.

1. Create 2+ *statistical distribution assertions*:

crashes are evenly/uniformly distributed throughout the months of the year.

crash causes are evenly distributed throughout the jurisdiction or region.

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:

* Revise assertion - I have changed some of my assertions according to the data.
* Discard the violating rows - There were a lot of Null values which was making data go inaccurate and confusing.
* Interpolate - Predicted some of the data missing and which has different values and Interpolated to meet the assertions.

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.

## 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?

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.