Share your accomplishment

Wrangle OpenStreetMap Data

Student Notes Project Review

Does Not Meet Specifications

Code Functionality

SPECIFICATION

Final project code functionality reflects the description in the project document.

MEETS SPECIFICATION

Reviewer Comments

Code submitted reflects work done in the submitted document PDF.

Code Readability

SPECIFICATION

Final project code follows an intuitive, easy-to-follow logical structure.

MEETS SPECIFICATION

Reviewer Comments

Distinct blocks separate distinct functions. There is no overlapping in function procedures. This makes the code extremely logical and easy to follow.

SPECIFICATION

Final project code that is not intuitively readable is well-documented with comments.

MEETS SPECIFICATION

Reviewer Comments

All critical areas are well commented.

Problems encountered in your map

SPECIFICATION

Student response shows understanding of the process of auditing, and ways to correct or standardize the data, including dealing with problems specific to the location, e.g. related to language or traditional ways of formatting.

DOES NOT MEET SPECIFICATION

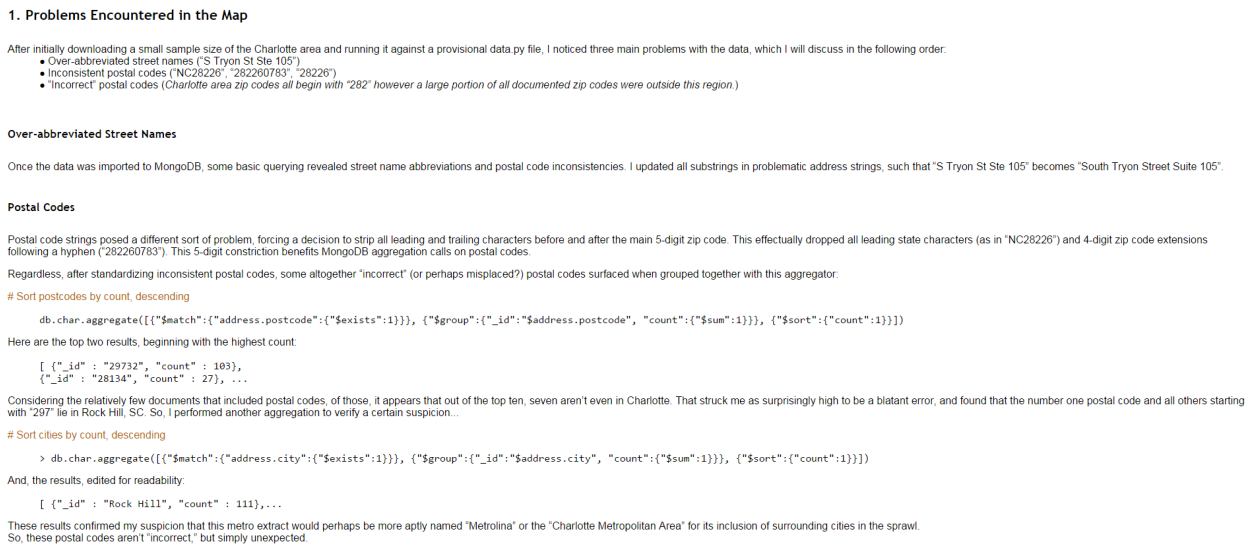
Reviewer Comments

In the **Problems Encountered..** section, there is only code. We want students to showcase the steps of wrangling. This includes auditing to find issues and then cleaning the issues found. I have attached the sample project to showcase the ideal format of this section. Please revise this area. Also, since you have provided the `.ipynb` file separately, there is not need to include large blocks of code in the presentation.

An example methodology would be this.....

1. Audit the data
    - From the audit, you should find issues
    - Showcase brief examples of the issues you found
2. Plan out action for cleaning some of the issues
    - Determine if the issues need to be solved by mappings, regex, imputing, etc.
    - Make sure the cleaning procedure doesn't require you to manually go in and fix the issue. **DO IT PROGRAMMATICALLY.**
    - Make sure to mention to the reader what your methods for cleaning were. No need to show the code if you can explain it.
3. Perform the cleaning procedures ✔ **Completed**
    - After performing the cleaning procedures, show before-and-after results of the cleaning you have done
4. (Extra Credit) Repeat Auditing Section to verify you have cleaned most of the data
    - Show how effective your cleaning procedure was.

Sample Project (Click on the Image to see a Larger Version)



1. Problems Encountered in the Map

After initially downloading a small sample size of the Charlotte area and running it against a provisional data.py file, I noticed three main problems with the data, which I will discuss in the following order:

- Over-abbreviated street names ("S Tryon St Ste 105")
- Inconsistent postal codes ("NC28226", "282260783", "28226")
- "Incorrect" postal codes (*Charlotte area zip codes all begin with "282" however a large portion of all documented zip codes were outside this region.*)

Over-abbreviated Street Names

Once the data was imported to MongoDB, some basic querying revealed street name abbreviations and postal code inconsistencies. I updated all substrings in problematic address strings, such that "S Tryon St Ste 105" becomes "South Tryon Street Suite 105".

Postal Codes

Postal code strings posed a different sort of problem, forcing a decision to strip all leading and trailing characters before and after the main 5-digit zip code. This effectually dropped all leading state characters (as in "NC28226") and 4-digit zip code extensions following a hyphen ("282260783"). This 5-digit constriction benefits MongoDB aggregation calls on postal codes.

Regardless, after standardizing inconsistent postal codes, some altogether "incorrect" (or perhaps misplaced?) postal codes surfaced when grouped together with this aggregator:

```
# Sort postcodes by count, descending

db.char.aggregate([{"$match":{"address.postcode":{"$exists":1}}}, {"$group":{"_id":"$address.postcode", "count":{"$sum":1}}}, {"$sort":{"count":1}}])
```

Here are the top two results, beginning with the highest count:

```
[ {"_id" : "29732", "count" : 103},
{"_id" : "28134", "count" : 27}, ...
```

Considering the relatively few documents that included postal codes, of those, it appears that out of the top ten, seven aren't even in Charlotte. That struck me as surprisingly high to be a blatant error, and found that the number one postal code and all others starting with "297" lie in Rock Hill, SC. So, I performed another aggregation to verify a certain suspicion...

```
# Sort cities by count, descending

> db.char.aggregate([{"$match":{"address.city":{"$exists":1}}}, {"$group":{"_id":"$address.city", "count":{"$sum":1}}}, {"$sort":{"count":1}}])
```

And, the results, edited for readability:

```
[ {"_id" : "Rock Hill", "count" : 111},...
```

These results confirmed my suspicion that this metro extract would perhaps be more aptly named "Metrolina" or the "Charlotte Metropolitan Area" for its inclusion of surrounding cities in the sprawl.
So, these postal codes aren't "incorrect," but simply unexpected.

https://docs.google.com/document/d/1F0Vs14oNEs2idFJR3C_OPxwS6L0HPliOii-QpbmrMo4/pub

- Notice how the samples project has used distinct formatting styles to distinguish different areas of the section. Using **bold**, *italics*, etc. are a great way to catch the employer's eyes and

make the entire document more readable.

SPECIFICATION

Some of the problems encountered during data audit are cleaned programmatically.

MEETS SPECIFICATION

Reviewer Comments

You have done a great job of demonstrating programmatic cleaning procedures. This is one of the most important advantages computers can give us.

Overview of the data

SPECIFICATION

The dataset is at least 50 MB.

MEETS SPECIFICATION

Reviewer Comments

```
charlotte.osm : 294.21 MB
charlotte.osm.json : 398.77 MB
```

SPECIFICATION

Student response provides a statistical overview of a dataset, like:

- size of the file
- number of unique users
- number of nodes and ways
- number of chosen type of nodes, like cafes, shops etc

MEETS SPECIFICATION

SPECIFICATION

Student response also includes the MongoDB queries used to obtain the statistics.

MEETS SPECIFICATION

Reviewer Comments

ADDITIONALLY (OPTIONAL)

Pipelines

I would recommend creating pipelines prior to database submission/query. This added step allows coders to review pipelines for errors that could potentially harm the database. Here is an example. Single Command Method

```
db.locations.aggregate([{"$group": {"_id": "$created.user", "count":
{"$sum":1}}}, {"$sort": {"count": -1}}, {"$limit": 5}])
```

Safer Method

```
pipeline = ([{"$group": {"_id": "$created.user", "count":
{"$sum":1}}},
{"$sort": {"count": -1}},
{"$limit": 5}]
db.locations.aggregate(pipeline)
```

This is just an example. This is particularly important when one is using `db.update()` or `db.insert()` to make database changes. Just something to think about and is optional.

Other ideas about the datasets

SPECIFICATION

Student proposes one or more additional ways of improving and analyzing the data.

DOES NOT MEET SPECIFICATION

Reviewer Comments

This rubric requirement and the next rubric requirements are related. This section require the students to have a thoughtful discussion about **additional** ways to improve data quality. These solutions do not necessarily have to deal with the issues you have discovered in the data and could deal with broader issues we talked about throughout the Lessons. I have provided a more expanded check list of requirements for this section.

Please include in the **Other Ideas about the datasets** section the following....

- Idea(s) concerning improving the data quality of OSM
  - You can reference problems you have encountered and talk about ways to prevent these issues in the future
  - Students are encouraged to think outside the box
  - Some ideas could include expanding upon
    - Gamification: encouraging user participation through incentives
    - Imputing missing values from other values within the same node
    - Cross-referencing/Cross-validating incorrect or missing data from other databases like Google API
      this is somewhat advanced, but fun!

SPECIFICATION

Student gives thoughtful discussion about the benefits as well as some anticipated problems in implementing the improvement.

DOES NOT MEET SPECIFICATION

Reviewer Comments

Of the idea(s) suggested, the rubric above also requires that students give a **"thoughtful discussion about the benefits and anticipated problems in implementing the improvement"**. What potential issues could you see that may arise from the implementation of this solution? It is very important for Data Scientist to be able to think several steps ahead. Many times we act as consultants to big decisions. Seeing potential issue and doing a cost/benefit analysis can make you a great asset to potential employers, not to mention a great leader of your own venture.

Thoroughness and Succinctness of Submission

SPECIFICATION

Student submission is long enough to thoroughly answer the questions asked without giving unnecessary detail. A good general guideline is that your question responses should take about 3-6 pages.

DOES NOT MEET SPECIFICATION

Reviewer Comments

Please understand that these projects will eventually be showcased in your Job-Ready portfolio. Employers, recruiters, etc. will be viewing them so the details I have outlined above are very important. Because of the nature of this project, it is extremely easy to clutter of the presentation to a point where it is not reader friendly. Currently, the format of the project does not meet the guidelines to be showcased in a Job-Ready portfolio. Please look at the sample project. It is not required to follow it exactly, but it does showcase the most ideal format of this project.

Sample Project

›bmrMo4/pub

How satisfied are you with this feedback?

- There are no excessive code blocks that deter from the information being conveyed
- Concise statements are used to convey the au ds with small samples to supplement the discussion
- Headlines, sub-headlines, and distinct areas are used to the reader clearly understands what part of the analysis they are on and what they are looking at
- It is not enough that the information is present, it needs to be easily readable

Resubmit Project

Download project

Learn the best practices for revising and resubmitting your project.

Have a question about your review? Email us at review-support@udacity.com and include the link to this review.