

Lab 8

Big Data Spring 2017

March 27, 2017

OpenRefine

- <http://openrefine.org/>
- Former Google project (now open source) that does data cleanup, transformation, etc. (data wrangling)
- Spreadsheet meets database
- User can filter the rows to display using ***facets*** that define filtering criteria (for example, showing rows where a given column is not empty)
- All actions that were done on a dataset are stored in a project and can be replayed on another dataset
- Web user interface (in browser), uses a web server on your local machine

Start OpenRefine

- Download the parking-violations-small.csv file from NYU Classes under Resources > Labs > Lab8
- Should have already download OpenRefine (<http://openrefine.org/download.html>)
- Start OpenRefine - should open browser console

Load Data into OpenRefine

- We will now load the CSV file into OpenRefine.
- Once the OpenRefine console opens in your browser, click “Choose Files” and select the `parking-violations-small.csv` file (wherever you saved it on your computer)
- Click **Next**

Data Preview

- You are now at the data preview page.
- Here you can set parameters and choose settings for data importing. You are shown a sample of the data.
- Keep all the defaults set
- Click “Create Project” in the upper right-hand corner

Checking for Duplicates

- summons_numbers must be unique - otherwise we have a functional dependency violation

Checking for Duplicates

- summons_numbers must be unique - otherwise we have a functional dependency violation
- Click drop-down arrow in the **summons_number** column
- Select **Facet>Customized Facet>Duplicates Facet**

Viewing range of values in a column

- First we will use a facet to view the different values in a given column
- Select the dropdown arrow next to `plate_type`.
- Select `Facet>Text Facet`

Viewing range of values in a column

- First we will use a facet to view the different values in a given column
- Select the dropdown arrow next to `plate_type`.
- Select `Facet>Text Facet`
- A facet window appears in the left side-pane.
 - We can see that there are 26 different values
 - Each value is listed along with the number of times it occurs in the column

Changing Values and Filtering Rows

- Notice that “999” is an entry.

Changing Values and Filtering Rows

- Notice that “999” is an entry.
- If we wanted to change this entry (make it blank/NULL, for example), you can hover over this row in the facet window, select “edit” and change this value to whatever you want.

Changing Values and Filtering Rows

- Notice that “999” is an entry.
- If we wanted to change this entry (make it blank/NULL, for example), you can hover over this row in the facet window, select “**edit**” and change this value to whatever you want.
- If we want to remove all rows with 999...
 - Click the drop-down arrow in **plate_type** column
 - Select “**Text Filter**”
 - Type “**999**” into the box that appears in the left pane
 - In the main pane, click the drop down arrow by “**All**” and select **Edit Rows>Remove All Matching Rows**
 - Click the X to close the text filter box in the left pane

Filtering rows with blank entries

- Suppose we are interested in analyzing violations based on what county they occur in.
- We therefore might want to exclude rows that have a blank entry in the **violation_county** column

Filtering rows with blank entries

- Suppose we are interested in analyzing violations based on what county they occur in.
- We therefore might want to exclude rows that have a blank entry in the **violation_county** column
- Let's see how many entries are blank:
 - Select the drop-down arrow for this column
 - Select **Facet>Customized Facet>Facet by blank**

Filtering rows with blank entries

- Suppose we are interested in analyzing violations based on what county they occur in.
- We therefore might want to exclude rows that have a blank entry in the **violation_county** column
- Let's see how many entries are blank:
 - Select the drop-down arrow for this column
 - Select **Facet>Customized Facet>Facet by blank**
- There is now a facet window open in the left pane
 - We see there are 467 rows with a blank entry in violation_county

Filtering rows with blank entries

- Suppose we are interested in analyzing violations based on what county they occur in.
- We therefore might want to exclude rows that have a blank entry in the **violation_county** column
- Let's see how many entries are blank:
 - Select the drop-down arrow for this column
 - Select **Facet>Customized Facet>Facet by blank**
- There is now a facet window open in the left pane
 - We see there are 467 rows with a blank entry in violation_county
- Hover over **“false”** in the facet window and select the **“include”** text that appears.
- We have now excluded rows that have blank entries in violation_county (we see there are 99533 rows left in the table)

Clustering

- Many times when data is input by humans, there are errors and inconsistencies.
- “Clustering” helps detect entries in a column that are close together (and thus represent the same value)

Clustering

- Many times when data is input by humans, there are errors and inconsistencies.
- “Clustering” helps detect entries in a column that are close together (and thus represent the same value)
- Let’s check for entry errors in **plate_id**:
 - Select the dropdown arrow in this column
 - Select **Edit Cells>Cluster and Edit...**

Clustering

- Many times when data is input by humans, there are errors and inconsistencies.
- “Clustering” helps detect entries in a column that are close together (and thus represent the same value)
- Let’s check for entry errors in **plate_id**:
 - Select the dropdown arrow in this column
 - Select **Edit Cells>Cluster and Edit...**
- We see that there are 3 instances of close entries in this column that are probably the same plate, but with human entry errors.
- To determine if these should be merged or not, you can click on **“Browse this cluster”** and look more closely at the data

Clustering

- Many times when data is input by humans, there are errors and inconsistencies.
- “Clustering” helps detect entries in a column that are close together (and thus represent the same value)
- Let’s check for entry errors in **plate_id**:
 - Select the dropdown arrow in this column
 - Select **Edit Cells>Cluster and Edit...**
- We see that there are 3 instances of close entries in this column that are probably the same plate, but with human entry errors.
- To determine if these should be merged or not, you can click on **“Browse this cluster”** and look more closely at the data
- Select merge on all three check boxes, keep suggested values, and click **“Merge Selected & Close”**.

Making text consistent

- Let's try clustering on the `street_name` column
 - Edit Cells>Cluster and Edit...

Making text consistent

- Let's try clustering on the **street_name** column
 - **Edit Cells>Cluster and Edit...**
- Yikes, 1439 clusters found!
- It would take a while to go through all of these
- Just by looking at the first few clusters, it is clear that we will need to make capitalization consistent and remove extraneous punctuation.
- Let's do this before we try to cluster - click "**Close**" on the clustering window

Making Text consistent

- To make everything capitalized, click **Edit cells> Common Transforms> To uppercase**

Making Text consistent

- To make everything capitalized, click **Edit cells> Common Transforms> To uppercase**
- To filter out punctuation, we will use a regular expression. Click **Edit cells> Transform...**
 - In the text box, write the command **value.replace(/[!@#%&';:]/, "")**
 - Click **“OK”**

Making Text consistent

- To make everything capitalized, click **Edit cells> Common Transforms> To uppercase**
- To filter out punctuation, we will use a regular expression. Click **Edit cells> Transform...**
 - In the text box, write the command **value.replace(/[!@#%&';:]/, "")**
 - Click **“OK”**
- Make sure there is no leading or trailing whitespace. Click **Edit cells> Common Transforms> trim leading and trailing whitespace**
- Click **Edit cells> Common Transforms> collapse consecutive whitespace**

Making Text Consistent

- Now let's try clustering again. Click **Edit cells> Cluster and Edit...**

Making Text Consistent

- Now let's try clustering again. Click **Edit cells> Cluster and Edit...**
 - Now there are only 15 clusters we have to look at!

Making Text Consistent

- Now let's try clustering again. Click **Edit cells> Cluster and Edit...**
 - Now there are only 15 clusters we have to look at!
- Look at the clusters that were found:
 - Sometimes it is clear that the entries should be merged
 - For example, "EAST 21 ST" and "EAST 21 ST ST" are probably the same street

Making Text Consistent

- Now let's try clustering again. Click **Edit cells> Cluster and Edit...**
 - Now there are only 15 clusters we have to look at!
- Look at the clusters that were found:
 - Sometimes it is clear that the entries should be merged
 - For example, "EAST 21 ST" and "EAST 21 ST ST" are probably the same street
 - Sometimes it is not as clear:
 - Are "AVE ST JOHNS" and "ST JOHNS AVE" the same thing?

Making Text Consistent

- Now let's try clustering again. Click **Edit cells> Cluster and Edit...**
 - Now there are only 15 clusters we have to look at!
- Look at the clusters that were found:
 - Sometimes it is clear that the entries should be merged
 - For example, "EAST 21 ST" and "EAST 21 ST ST" are probably the same street
 - Sometimes it is not as clear:
 - Are "AVE ST JOHNS" and "ST JOHNS AVE" the same thing?
 - Hover over this row and click "**Browse this cluster**"

Making Text Consistent

- Now let's try clustering again. Click **Edit cells> Cluster and Edit...**
 - Now there are only 15 clusters we have to look at!
- Look at the clusters that were found:
 - Sometimes it is clear that the entries should be merged
 - For example, "EAST 21 ST" and "EAST 21 ST ST" are probably the same street
 - Sometimes it is not as clear:
 - Are "AVE ST JOHNS" and "ST JOHNS AVE" the same thing?
 - Hover over this row and click "**Browse this cluster**"
 - We can see that the county is marked and being different for these two cases, so these entries should probably not be merged.
- **Determine which clusters should be merged. Merge them and close.**

More on Clustering

- Look at the **vehicle_color** column.
 - Just glancing at the entries, it is clear that there are many inconsistencies (e.g., “WHITE”, “WHT”, “WT”)

More on Clustering

- Look at the **vehicle_color** column.
 - Just glancing at the entries, it is clear that there are many inconsistencies (e.g., “WHITE”, “WHT”, “WT”)
- In this column, select **Edit Cells>Cluster and Edit**

More on Clustering

- Look at the **vehicle_color** column.
 - Just glancing at the entries, it is clear that there are many inconsistencies (e.g., “WHITE”, “WHT”, “WT”)
- In this column, select **Edit Cells>Cluster and Edit**
- Only 6 clusters were found. Select merge on all 6, then click **“Merge Selected and Re-Cluster”**

More on Clustering

- Now it says there are no more clusters - but there clearly are (e.g., “WHT” and “WHITE”).
- This is because we are only using key collision as the clustering method.

More on Clustering

- Now it says there are no more clusters - but there clearly are (e.g., “WHT” and “WHITE”).
- This is because we are only using key collision as the clustering method.
- In the “Method” drop-down, select “nearest neighbor”

More on Clustering

- Now it says there are no more clusters - but there clearly are (e.g., “WHT” and “WHITE”).
- This is because we are only using key collision as the clustering method.
- In the “Method” drop-down, select “nearest neighbor”
- Hmm...it says no clusters. Try reducing block chars to 4.

More on Clustering Parameters

- Key collision methods: very fast (linear runtime) but can be either too strict or too lax with no way to fine tune allowable distance between strings
- Nearest neighbor methods provide parameters where we can tune this
 - Radius: a distance threshold - any pair of strings closer than a certain value clustered together
 - Block chars: speeds up algorithm by first passing over sequence of strings to evaluate and obtains blocks in which all strings share a substring of given blocking size
 - a hybrid between key collision and nearest neighbor
- Read more about the parameters and clustering algorithms here: <https://github.com/OpenRefine/OpenRefine/wiki/Clustering-In-Depth>

Back to color merging...

- Again, whether or not we merge is subjective

Back to color merging...

- Again, whether or not we merge is subjective
 - depends on what we plan to do with the data

Back to color merging...

- Again, whether or not we merge is subjective
 - depends on what we plan to do with the data
- After you resolve these merges, tweak the Radius and Block Chars parameters to adjust the clustering and try to resolve more.
- You can also close the clustering, examine the column text facet to see what else you need to merge, and do this manually in the facet window

Editing History/Undoing Changes

- In the upper left-hand corner, there is an **undo/redo** tab. Click it.
- Here we can see all the steps of changes we've made to the table.
- You can click on a step to go back to that point!

OpenRefine Documentation

- Much more functionality - only covered a small subset today
- See the documentation and tutorials for more:

<http://openrefine.org/documentation.html>

Deliverable - due Wednesday, March 29, 6pm.

Complete the following three data cleaning tasks in OpenRefine using the parking-violation-small.csv dataset: (you can use the version you currently have - no need to revert changes you've made already during lab)

1. In the violation_time column, check if there are times that are invalid or blank (i.e., not a valid 12-hour clock time). Exclude these rows from the table.
2. In the registration_state column, check if there are invalid state entries. If so, exclude these rows from the table.
3. In the vehicle_make column, cluster similar items to make vehicle make labels consistent

Submission: In the top right of the OpenRefine console, click “Export”>Comma-separated value to save your cleaned data file as a csv file. Submit this CSV file to NYU Classes.