EZTracker User Manual

Tomas Baliu-Rodriguez, Michael Mott,
Claire Horack, Vinay Dawani

April 26, 2021

Vision Statement

For Data Analysts
who want to dynamically find similarities in attributes across geographic data,
the EZTracker
is an interactive plugin in QGIS
that highlights polygons that fall within an attribute value range similar to a user-selected polygon.
Unlike a static map that displays data ranges throughout the region,
our product allows a user to click the target polygon, providing a dynamic view of similar polygons
which supports our strategy to visualize data and better inform data analysts

Installation Guide

Method 1(recommended):

1) Download the file eztracker.zip
2) Open QGIS
3) Plugins>Manage and Install Plugins
4) Click on Install from ZIP tab
5) Click on the "..." button and navigate to extracker.zip
6) Click Install Plugin
7) You will now find the plugin on the plugin toolbar

Method 2:

1) Download the file eztracker.zip
2) Extract folder to one of the options below, where USER_NAME is the actual account name on the computer
    a) PC: C:\Users\USER_NAME\AppData\Roaming\QGIS\QGIS3\profiles\default\python\plugins\
    b) MAC: /Users/USER_NAME/Library/Application Support/QGIS/QGIS3/profiles/default/python/plugins/
3) Open QGIS.
4) Click on Plugins>Manage and Install Plugins>
5) Click on Installed Tab.
6) Select EZTracker.
7) Click close.
8) You will now find the plugin under the plugin toolbar.

Usage Guide

1) Click on the EZTracker plugin.
2) Load one or more polygonal .shp files into the layers window and set one as active.
3) Select a numerical attribute from the drop-down menu. If no attributes are listed then there are no numerical attributes for that .shp file.
4) Select a number for the bound. This number will be added and subtracted to the largest and smallest attribute values acquired from the polygon selections.
5) Click one or more polygons from the active layer and all polygons containing attribute values that fall within the bound will be highlighted. When multiple polygons are selected, all polygons containing an attribute value in between the attributes of the selections will be highlighted.
6) Clicking outside the layer will remove all polygon selections and highlights.
7) To change the layer you are working with, set the desired layer as active.
8) When you update the Attribute Value Range, the QGIS map will not display the update to the highlights until it is clicked on. It is recommended to do so using a middle mouse button as this will not affect the current selections made by the user. If you prefer, you may reselect the yellow polygons by clicking on them to see the updated result.

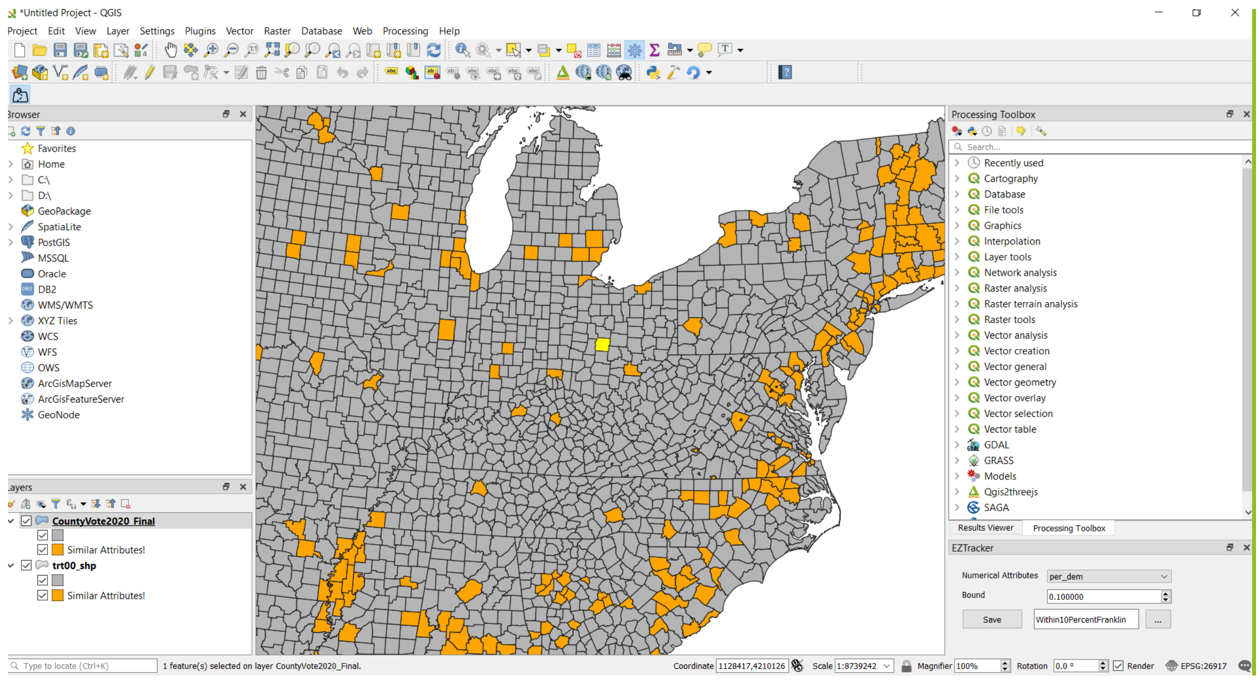
Saving a result:

1) Once you have the similar features that are desired, click the 'Select Folder' button and select the location you wish to have the file saved.
2) Enter a name for the file. If left blank nothing will happen. If the file name already exists in the location, the original file will be overwritten.
3) Click Save as Style.
4) This output is saved as a .qml file. This is saving the rule symbology that creates the output viewed on the map. It is important to note that the .qml file needs to be saved near to the .shp file used to create it.

Retrieving a saved output:

1) First load in the .shp file that was used to create the output.
2) Right-click on the layer and click Properties
3) Click Symbology
4) In the Symbology tab, at the bottom of the window click Style.
5) Click Load Style.
6) Click on the '...' next to File.
7) Navigate to the desired .qml file that was previously saved
8) Click Load Style
9) Click apply

Results:



CountyVote2020_Final
-
- Similar Attributes!

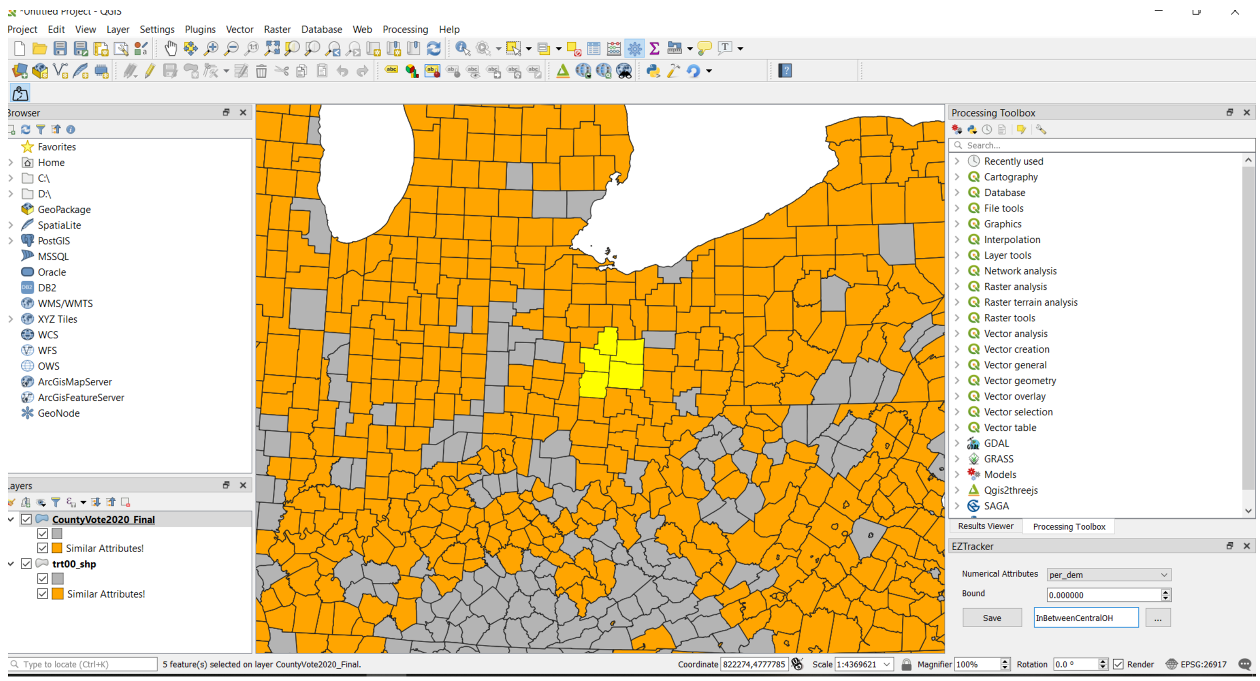
trt00_shp
-
- Similar Attributes!

Numerical Attributes: per_dem
Bound: 0.100000
Within10PercentFranklin

1 feature(s) selected on layer CountyVote2020_Final.

Coordinate 1128417,4210126 Scale 1:8739242 Magnifier 100% Rotation 0.0 ° Render EPSG:26917

Figure 1: One Selection with a Bound



CountyVote2020_Final
-
- Similar Attributes!

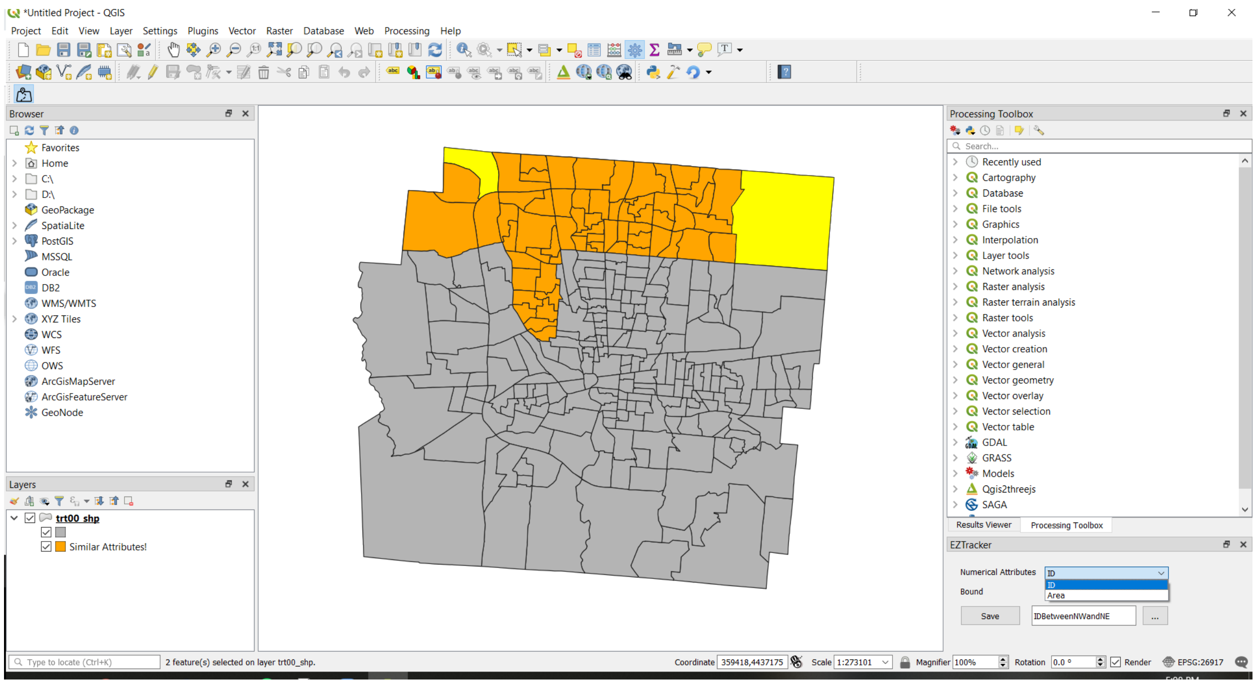
trt00_shp
-
- Similar Attributes!

Numerical Attributes: per_dem
Bound: 0.000000
InBetweenCentralOH

5 feature(s) selected on layer CountyVote2020_Final.

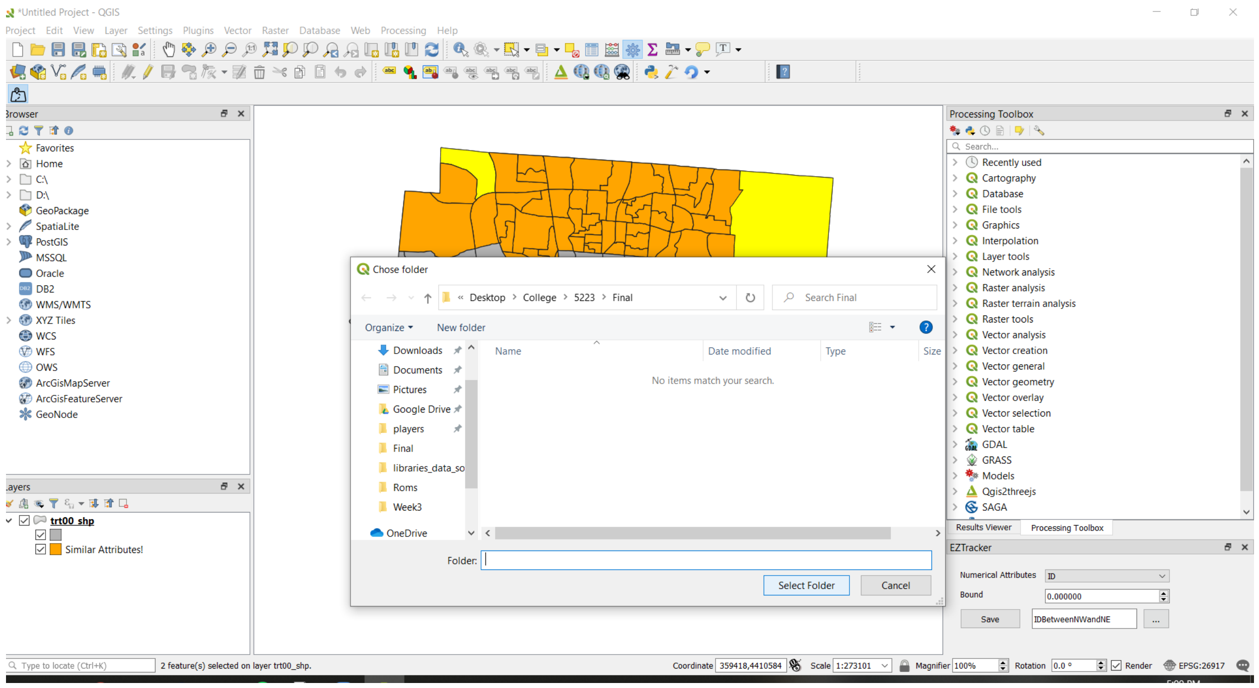
Coordinate 822274,4777785 Scale 1:4369621 Magnifier 100% Rotation 0.0 ° Render EPSG:26917

Figure 2: Four Selections with no Bound



2 feature(s) selected on layer trt00_shp.

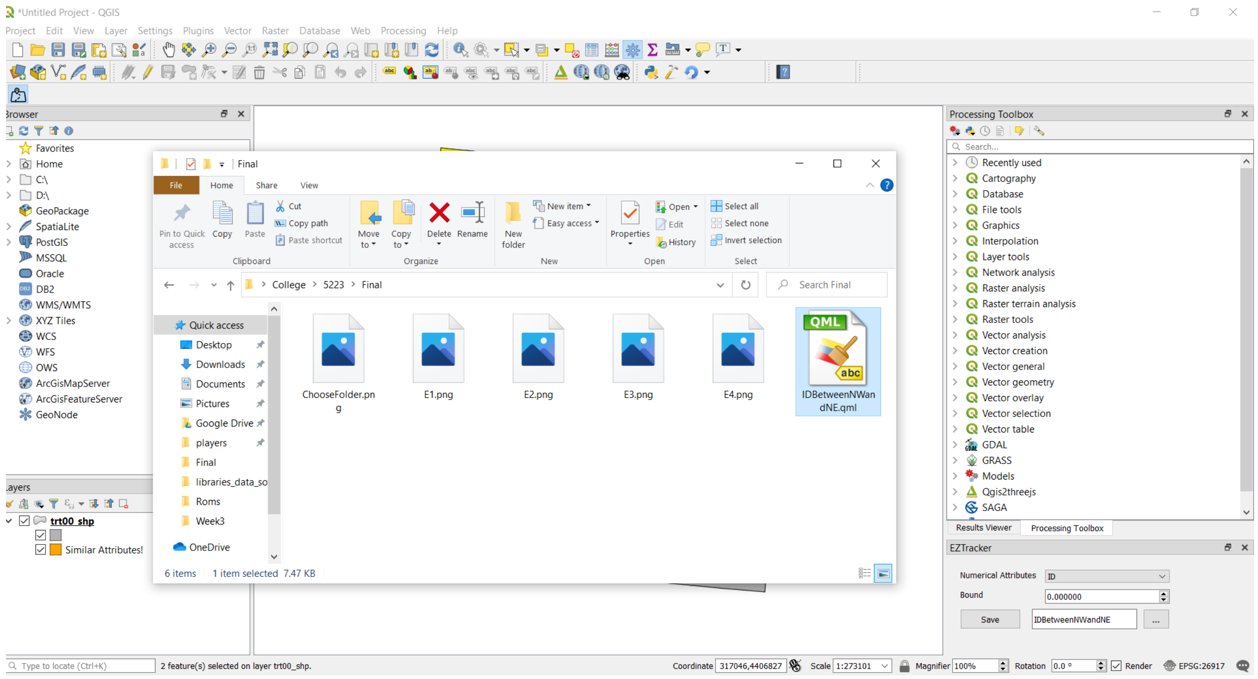
Figure 3: Two Selections with no Bound and Attribute List



No items match your search.

2 feature(s) selected on layer trt00_shp.

Figure 4: Radio Button folder selection



Numerical Attributes: ID
Bound: 0.000000
IDBetweenNWandNE

2 feature(s) selected on layer trt00_shp.
Coordinate 317046,4406827 Scale 1:273101 Magnifier 100% Rotation 0.0 ° EPSG:26917

Figure 5: The File Name and Saved Output

In Figure 1, you can see that Franklin County was selected along with the Percent of Democratic votes in the numerical attribute combo box with a bound of .1 meaning 10% in our data. Thus, all counties containing plus or minus 10% of the democratic votes as Franklin County was highlighted.

In Figure 2, you can see four counties in central Ohio were selected but no bounds, thus the bounds will be the highest and lowest attribute values (in this case percentages again) from the selection. So all highlighted counties fall within the voting percentages of the four selected counties. Giving a higher bound would highlight even more counties that fall within the natural extents plus and minus the bound.

In Figure 3, you can clearly see that all the area IDs in between the selection are highlighted. Also, the drop-down for the numerical attributes is shown.

In Figure 4, you see the result of clicking on the radio button next to the save button. This pulls up a file explorer and the user then clicks on choose folder once they find the area to save.

In Figure 5, you see that the user gave the output a valid name and hit save after choosing a folder. The output is clearly saved in the selected folder with the given name as a QML file.

Discussion

As this project was a QGIS plugin created in Python, an extensive software development process was done. Since we were learning pyQGIS during the development, we had to properly plan out what processes would be coded first. To do this, we began by taking our user stories and wrote down every part of the code that would need to be completed. They were: dockwidget plugin setup, interactivity, and select tool setup, layer update, value update, attribute selection drop-down, obtain bounds, search through features, highlight features, folder selection, and saving the output. After we did this, we ordered the list in how we thought it would make sense to complete it. We then reviewed the order by looking at the carmen QGIS modules and saw if we learned the material necessary in time. A couple of items in the list had to be re-ordered due to this.

After we created a timetable for the completion of the parts for our code, we split them into sprints. As our development team worked on the specific parts of the code, the scrum master made sure that the code would work for the end result and made sure each specific part properly fit into the entire code. Our product owner would spend time making sure the releases were met and would review and receive feedback to give to the development team. The development team followed a simple process of: create a skeleton code, fill out the skeleton, debug the code. After every release, we would take a day to fix the issues found in the code from the feedback and move on to the next section of the code. If a part of the project was not completed within our weekly scrum, we would find more time throughout the week to continue working so as to not fall behind schedule.