

# Find Neighbor Polygons in a Layer

QGIS Tutorials and Tips



Author

Ujaval Gandhi

<http://google.com/+UjavalGandhi>

Translations by

Marina Pavlova Ilya Trofimov Fayçal Fatihi

## Introduction to QGIS and Python

QGIS is a free and open source GIS. It is a desktop application that can be used to view, edit, and analyze spatial data. QGIS is written in Python and uses the Qt framework for its user interface. QGIS is a powerful tool for working with spatial data, and it is a great choice for anyone who is interested in GIS. In this tutorial, we will learn how to use QGIS and Python to work with spatial data. We will start by installing QGIS and Python, and then we will learn how to use QGIS to load and display spatial data. Finally, we will learn how to use Python to automate tasks in QGIS.

### Installing QGIS and Python

QGIS is available for Windows, Mac OS, and Linux. Python is available for Windows, Mac OS, and Linux. To install QGIS and Python, you need to download the software from the official websites. The QGIS website is <http://qgis.org> and the Python website is <http://python.org>. Once you have downloaded the software, you can install it on your computer. The installation process is straightforward and should not take too long.

### Getting Started with QGIS and Python

QGIS has a built-in Python console that allows you to run Python code directly within the application. To open the Python console, go to **Plugins > Manage and Install Plugins...** and check the **Python Console** checkbox. Once the console is open, you can run Python code to interact with QGIS. For example, you can use the `shapefile` module to load and display a shapefile. The following code snippet shows how to load a shapefile named `Admin 0 - Countries` from the `Natural Earth` dataset.

### Using the Python Console in QGIS

The `neighbors.py` script is a Python script that can be used to find the neighbors of a given country. To run the script, you need to save it to a file and then execute it from the Python console. The following code snippet shows how to run the `neighbors.py` script.

### Running the Python Script

1. In the Python console, enter the following code to load the `ne_10m_admin_0_countries` shapefile and add it to the QGIS map canvas as a **Vector Layer**.





3. ■■■■■■■■■■ ■■■■■ Plugins ■ Python Console.





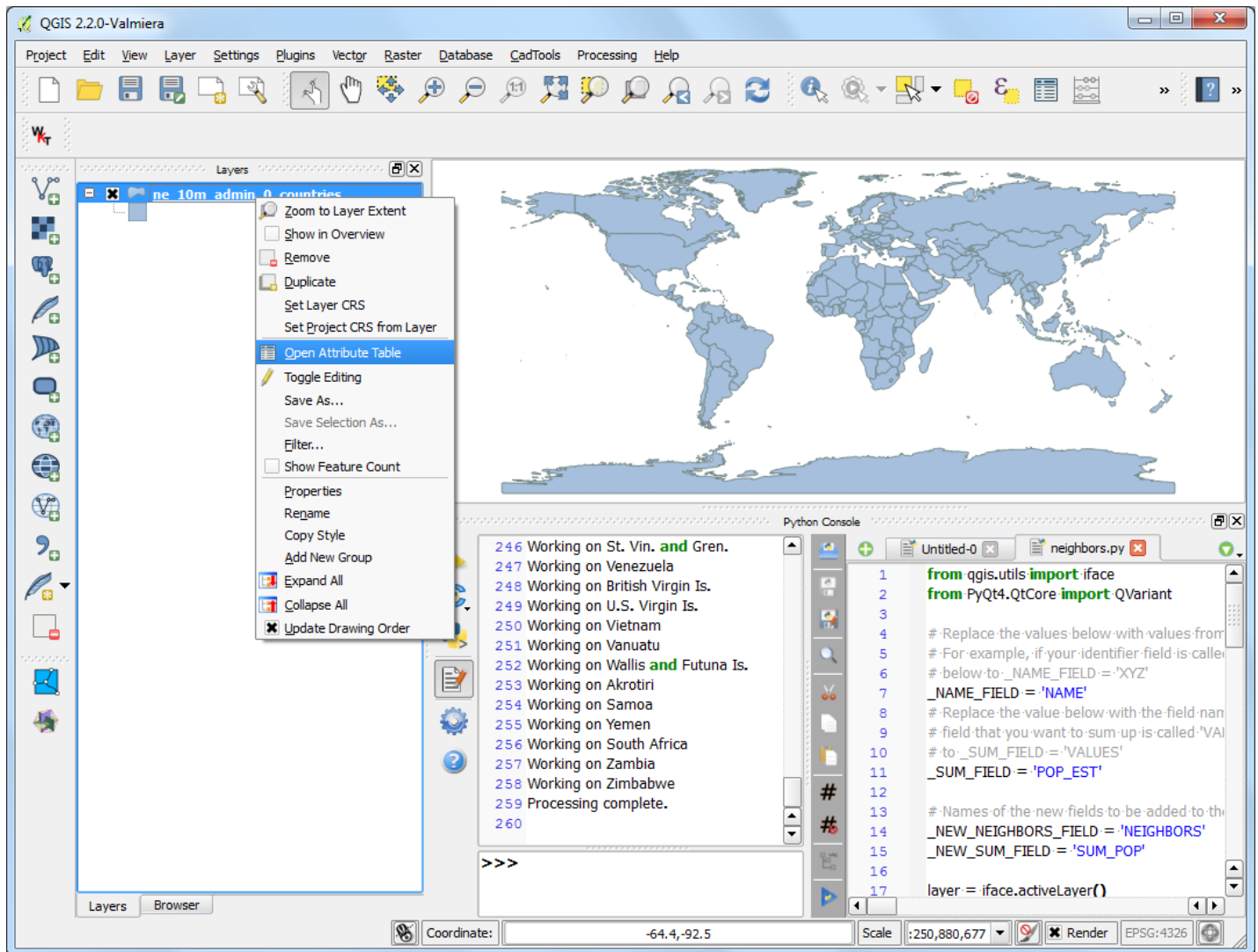


6. `ne_10m_admin_0_countries`, `ne_10m_admin_0_countries`, `ne_10m_admin_0_countries`  
`_NAME_FIELD` `_SUM_FIELD` `ne_10m_admin_0_countries`  
`ne_10m_admin_0_countries`, `ne_10m_admin_0_countries`, `ne_10m_admin_0_countries`  
`ne_10m_admin_0_countries`: `guilabel: save` `ne_10m_admin_0_countries`: `guilabel: Editor`, `ne_10m_admin_0_countries`  
`ne_10m_admin_0_countries`: `guilabel: Run script`, `ne_10m_admin_0_countries`



7. **ne\_10m\_admin\_0\_countries** **Open** **Table**.





8. `NEIGHBORS` `SUM`.

Attribute table - ne\_10m\_admin\_0\_countries :: Features total: 255, filtered: 255, selected: 0

	ION	REGION_WB	NAME_LEN	LONG_LEN	ABBREV_LEN	TINY	HOMEPART	NEIGHBORS	SUM
0		Latin America & ...	5.00	5.00	5.00	4.00	-99.00	NULL	0
1	sia	South Asia	11.00	11.00	4.00	-99.00	1.00	Iran,Turkmenista...	1621125240
2	a	Sub-Saharan Africa	6.00	6.00	4.00	-99.00	1.00	Namibia,Zambia,...	86676756
3		Latin America & ...	8.00	8.00	4.00	-99.00	-99.00	NULL	0
4	urope	Europe & Central...	7.00	7.00	4.00	-99.00	1.00	Macedonia,Greec...	15281164
5	urope	Europe & Central...	5.00	13.00	5.00	5.00	-99.00	NULL	0
6	urope	Europe & Central...	7.00	7.00	4.00	5.00	1.00	France,Spain	104582794
7	ia	Middle East & No...	20.00	20.00	6.00	-99.00	1.00	Saudi Arabia,Oman	32104718
8	ica	Latin America & ...	9.00	9.00	4.00	-99.00	1.00	Bolivia,Paraguay,...	235606259
9	ia	Europe & Central...	7.00	7.00	4.00	-99.00	1.00	Georgia,Turkey,I...	156089287
10		East Asia & Pacific	14.00	14.00	9.00	3.00	-99.00	NULL	0
11		Antarctica	10.00	10.00	4.00	-99.00	1.00	NULL	0
12	d Ne...	East Asia & Pacific	23.00	27.00	7.00	-99.00	-99.00	NULL	0
13	ope...	Sub-Saharan Africa	22.00	35.00	10.00	2.00	-99.00	NULL	0
14		Latin America & ...	17.00	19.00	6.00	4.00	1.00	NULL	0
15	d Ne...	East Asia & Pacific	9.00	9.00	4.00	-99.00	1.00	NULL	0
16	rope	Europe & Central...	7.00	7.00	5.00	-99.00	1.00	Italy,Hungary,Slo...	175681436
17	ia	Europe & Central...	10.00	10.00	4.00	-99.00	1.00	Georgia,Turkey,R...	290858866
18	ica	Sub-Saharan Africa	7.00	7.00	4.00	-99.00	1.00	Rwanda,Tanzani...	120214356
19	rope	Europe & Central...	7.00	7.00	5.00	-99.00	1.00	France,Netherla...	163595324
20	rica	Sub-Saharan Africa	5.00	5.00	5.00	-99.00	1.00	Nigeria,Niger,Bur...	186301451
21	rica	Sub-Saharan Africa	12.00	12.00	4.00	-99.00	1.00	Mali,Niger,Ghana...	87234511
22	sia	South Asia	10.00	10.00	5.00	-99.00	1.00	India,Myanmar	1214216958

Show All Features

#####  
#####

```
#####
# Copyright 2014 Ujaval Gandhi
#
#This program is free software; you can redistribute it and/or
#modify it under the terms of the GNU General Public License
#as published by the Free Software Foundation; either version 2
#of the License, or (at your option) any later version.
#
#This program is distributed in the hope that it will be useful,
#but WITHOUT ANY WARRANTY; without even the implied warranty of
#MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
#GNU General Public License for more details.
#
#You should have received a copy of the GNU General Public License
#along with this program; if not, write to the Free Software
#Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.
#
#####
from qgis.utils import iface
from PyQt4.QtCore import QVariant

# Replace the values below with values from your layer.
# For example, if your identifier field is called 'XYZ', then change the line
```

```

# below to _NAME_FIELD = 'XYZ'
_NAME_FIELD = 'NAME'
# Replace the value below with the field name that you want to sum up.
# For example, if the # field that you want to sum up is called 'VALUES', then
# change the line below to _SUM_FIELD = 'VALUES'
_SUM_FIELD = 'POP_EST'

# Names of the new fields to be added to the layer
_NEW_NEIGHBORS_FIELD = 'NEIGHBORS'
_NEW_SUM_FIELD = 'SUM'

layer = iface.activeLayer()

# Create 2 new fields in the layer that will hold the list of neighbors and sum
# of the chosen field.
layer.startEditing()
layer.dataProvider().addAttributes(
    [QgsField(_NEW_NEIGHBORS_FIELD, QVariant.String),
     QgsField(_NEW_SUM_FIELD, QVariant.Int)])
layer.updateFields()
# Create a dictionary of all features
feature_dict = {f.id(): f for f in layer.getFeatures()}

# Build a spatial index
index = QgsSpatialIndex()
for f in feature_dict.values():
    index.insertFeature(f)

# Loop through all features and find features that touch each feature
for f in feature_dict.values():
    print 'Working on %s' % f[_NAME_FIELD]
    geom = f.geometry()
    # Find all features that intersect the bounding box of the current feature.
    # We use spatial index to find the features intersecting the bounding box
    # of the current feature. This will narrow down the features that we need
    # to check neighboring features.
    intersecting_ids = index.intersects(geom.boundingBox())
    # Initialize neighbors list and sum
    neighbors = []
    neighbors_sum = 0
    for intersecting_id in intersecting_ids:
        # Look up the feature from the dictionary
        intersecting_f = feature_dict[intersecting_id]

        # For our purpose we consider a feature as 'neighbor' if it touches or
        # intersects a feature. We use the 'disjoint' predicate to satisfy
        # these conditions. So if a feature is not disjoint, it is a neighbor.
        if (f != intersecting_f and
            not intersecting_f.geometry().disjoint(geom)):
            neighbors.append(intersecting_f[_NAME_FIELD])
            neighbors_sum += intersecting_f[_SUM_FIELD]
    f[_NEW_NEIGHBORS_FIELD] = ','.join(neighbors)
    f[_NEW_SUM_FIELD] = neighbors_sum
    # Update the layer with new attribute values.
    layer.updateFeature(f)

```

```
layer.commitChanges()  
print 'Processing complete.'
```