

Find Neighbor Polygons in a Layer

QGIS Tutorials and Tips



Author

Ujaval Gandhi

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

Translations by

Christina Dimitriadou

Paliogiannis Konstantinos

Tom Karagkounis

Βρε■τε Γειτονικ■ Πολ■γωνα σε ■να Επ■πεδο

Υπ■ρχουν κ■ποιες περιπτ■σεις χρ■σης ■που θ■λετε να βρε■τε ■λα τα γειτονικ■ πολ■γωνα καθεν■ς απ■ τα πολ■γωνα σε ■να επ■πεδο . Με ■να μικρ■ σεν■ριο Python, μπορο■με να το πετ■χουμε αυτ■ και πολ■ περισσ■τερο στο QGIS. Εδ■ ε■ναι ■να παρ■δειγμα σεναρ■ου, που μπορε■τε να χρησιμοποι■σετε, για να βρε■τε ■λα τα πολ■γωνα που μοιρ■ζονται τα ■ρι■ τους, με κ■θε ■να απ■ τα πολ■γωνα σε ■να επ■πεδο και επ■σης μπορε■τε να προσθ■σετε τα ον■ματ■ τους στον π■νακα ιδιοτ■των . Ως πρ■σθετο πλεον■κτημα, επ■σης το σεν■ριο συνοψ■ζει μ■χρι ■να χαρακτηριστικ■ της επιλογ■ς σας απ■ ■λα τα γειτονικ■ πολ■γωνα .

Επισκ■ψηση του ■ργου

Για να αποδε■χθε■, π■ς λειτουργε■ το σεν■ριο, θα χρησιμοποι■σουμε ■να επ■πεδο των πολυ■νων της χ■ρας και να βρεθο■ν οι χ■ρες που μοιρ■ζονται τα σ■νορα . Θ■λουμε επ■σης να υπολογιστε■ ο συνολικ■ς πληθυσμ■ς των γειτ■νων της χ■ρας .

Αποκτ■στε τα δεδομ■να

Θα χρησιμοποι■σουμε τα [Admin 0 - Χ■ρες](#) dataset from Natural Earth.

Κατεβ■στε το [Admin 0 - χ■ρες shapefile](#) ..

Πηγ■ Δεδομ■νων [NATURALEARTH]

Αποκτ■στε το script

Κατεβ■στε το: λ■ψη : ``neighbors.py script< /static/find_neighbor_polygons/script/neighbors.py >`` και να το αποθηκε■σετε στο δ■σκο σας .

Διαδικασ■α

1. Τοποθετ■στε το `ne_10m_admin_0_countries` layer πηγ■νοντας στο *Layer ■ Add Vector Layer*.



2. Το σενάριο χρησιμοποιεί 2 πεδία για να εκτελέσει την ενέργεια. Ένα πεδίο όνομα και ένα πεδίο που θέλετε να συνοψίσει. Χρησιμοποιήστε το : guilabel : `εργαλείο Identify`, κάνετε κλικ σε οποιοδήποτε στοιχείο και εξετάστε τα χαρακτηριστικά. Στην περίπτωση αυτή, το πεδίο όνομα είναι **NAME** και θέλουμε να συνοψίσουμε τις εκτιμήσεις για τον πληθυσμό από το ****POP_EST**** πεδίο.



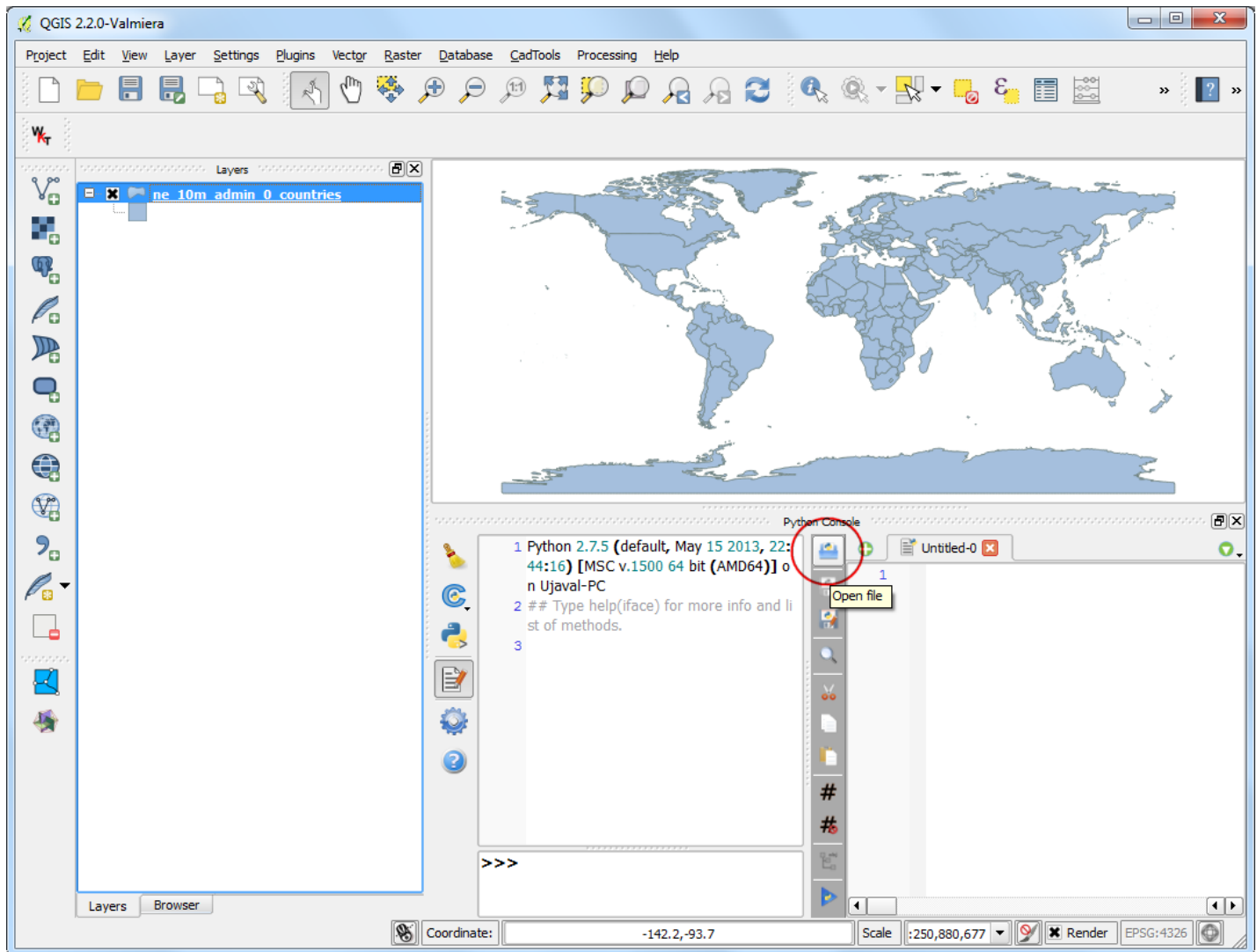
3. Μεταβασή σε: `menuselection:Plugins --> Python Console`.



4. Στο παρ■θυρο *Python Console* , κ■ντε κλικ στο κουμπ■ *Show Editor*.



5. Στον πίνακα: `guiabel:Editor` , κάντε κλικ στο κουμπι *Open file* και αναζητήστε στο script `neighbors.py` που έχετε κατεβάσει και κάντε κλικ στο *Open*.



6. Όταν φορτωθεί το script, μπορείτε να θέλετε να αλλάξετε το `_NAME_FIELD` και `_SUM_FIELD` για να ταιριάζει με τα χαρακτηριστικά από το δικό σας επίπεδο. Εάν εργάζεστε με το επίπεδο `ne_10m_admin_0_countries`, μπορείτε να αφίσετε αυτό όπως είναι. Κάντε κλικ στο κουμπί: `guilabel: `Save`` στον πίνακα: `guilabel: `Editor``, αν έχετε κάνει οποιοδήποτε αλλαγή. Τώρα κάντε κλικ στο κουμπί: `guilabel: Run script`, για να εκτελέσει το script.



7. Μόλις το script τερματιστεί, κάντε δεξί κλικ στο επώνυμο `` ne_10m_admin_0_countries`` και επιλέξτε: guilabel : *Open Attribute Table*.



8. Θα παρατηρήσετε 2 νέα χαρακτηριστικά που ονομάζονται `` NEIGHBORS`` και `` SUM``. Αυτά προστέθηκαν απ' το script.

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

Παρακάτω είναι το πλήρες script για την αναφορά. Μπορείτε να το τροποποιήσετε ώστε να ταιριάζει στις ανάγκες σας.

```
#####
# 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.'
```