RASTER ANALYTICS IN ARCGIS NOTEBOOKS

Internship Project
Student: Frida Ruiz Mendoza
Supervisor: Thomas Paschke
CONTENT

• About the project
  • ArcGIS Notebooks
• Methods
• Showcase: Simple Change detection in Bremen
• *Tips and tricks* (Story Map)
• Recommendations and Future work
The main objective of this project was to:

“Create a showcase for ArcGIS Notebooks and explore the advantages it offers to create end-to-end workflows for raster analysis”
ArcGIS Notebooks

Hosted Jupyter Notebook in your ArcGIS Enterprise portal and powered by the new ArcGIS Notebook Server.
Earth Observation programme by ESA that aims to provide global, continuous, autonomous, high quality.
Simple change detection for vegetation monitoring
METHOD

Different Methods for change detection

Credits: Herlod M, Remote Sensing techniques WUR, 2019

- Image Algebra (Image subtraction method) & Post-classification comparisons
- Commonly done for two points in time
- Straightforward and widely used

First Date

Second Date

Difference map
ARCGIS API FOR PYTHON

ArcGIS Notebooks sits at the intersection of ArcGIS and open data science

ArcGIS Notebooks
Module 1.

1. Understand GIS objects
2. Search for vector data (feature Layer)
3. Query attributes and display in a table (Pandas spatial DataFrame)
4. Select single Features and create a geometry using Geometry module
Module 2.

Filter Imagery Layer and query specific dates

- Filters images that intersect study area
- Filters images that have certain cloud cover

Natural Color with DRA representation of study area on 2018-08-22

Spectral Signature at point: x: 1443400.3748620122 and y: 6881093.89442059
Module 3

Example: Simple change detection in Bremen

```python
# display items on map
m1 = agol_gis.map('bremen')
m1.add_layer(old)
m2 = agol_gis.map('bremen')
m2.add_layer(new)

#m1.Layer = Layout(flex='1 1', padding='6px', height='450px')
#m2.Layer = Layout(flex='1 1', padding='6px', height='450px')
m1label = widgets.HTML('Study area NDVI on: ' + oldDate.value)
m2label = widgets.HTML('Study area NDVI on: ' + newDate.value)
btxt = widgets.HBox([m1label,m2label])
#btxt
b=widgets.HBox([m1,m2])
b
```
```python
### from bokeh.plotting import figure, show, output_file
from bokeh.models import NumeralTickFormatter

### Function to plot based on group size and data (counts), returns bokeh histogram.

def plot_histogram(counts, nGroups):
    # counts is a variable (topvalGrouped = stats históstos['Histogram'][0]['counts'])
    # stats histórico will change everytime a
    # nGroups is a number divisible by 256 of our choosing

    assert (256 % nGroups == 0), "number of groups must be a divisor of 256"  # This is making our custom error message: if

def divided_list_up_in_nGroups_sublists, taken from https://stackoverflow.com/questions/8671234/split-a-python-list-into-sublist
chunks = [counts[x:x+int(256/nGroups)] for x in range(0, len(counts), int(256/nGroups))]

# Sum all items in each sublist and store these in a new list.

topvalGrouped = []

i = 0
while i < len(chunks):
    topvalGrouped.append(sum(chunks[i]))
    i += 1

# create names for each bar.

start = -1.0

groupIncrement = 2/nGroups

xlabel = []  # creating empty list to input str from count plus each increment

while start < 1:
    xlabel.append(str(round(start, 2)) + " to " + str(round(start+groupIncrement, 2)))


start += groupIncrement

print(len(xLabel), groupIncrement, len(chunks))

# Plotting, dynamic width depending on amount of groups.

p = figure(title="NDVI distribution for chosen study area", plot_width=200, plot_height=200, x_range=xLabel)
p.xaxis.formatter = NumeralTickFormatter(format="0")
p.xaxis.major_label_orientation = 1  # change label x orientation

p.xaxis.axis_label = "Pixel Count"
p.xaxis.axis_label_text_font_style = "bold"
p.xaxis.axis_label_text_font_style = "bold"  # styling

if we have too many bars, remove labels on the x-axis:
    if nGroups > 22:
        p.xaxis.major_label_text_color = None

return p
```
NDVI Classification
```python
# Display difference in NDVI from two dates reclassified as loss, gain and no change
m = egol_gis.map('bremen')
print('Difference in NDVI classified as no change (gray), loss (red), gain (green)')
m.add_layer(remapdiff)
```

![Difference in NDVI classified as no change (gray), loss (red), gain (green)](image1)

```python
# Summary: What is the percentage of area that x = percentChange
data = pd.Series(x)
# making our dictionary a pandas series
# data['show df']
data.reset_index(name='Change class').rename(columns={})

<table>
<thead>
<tr>
<th>Change Class</th>
<th>Change class</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Loss</td>
</tr>
<tr>
<td>1</td>
<td>Gain</td>
</tr>
<tr>
<td>2</td>
<td>Stable</td>
</tr>
</tbody>
</table>
```

![Summary of change in NDVI classes between: 2018.05.29 and 2019.06.29](image2)
Story Map

Raster Analytics in ArcGIS Notebooks

The easiest way to automate your work-flow

Frida Ruiz Mendoza  |  August 06, 2019

https://storymaps.arcgis.com/stories/41ede17ec6754aae81ea87d072902535
Future work and lessons learned

Lessons learned:

• Working with ONE item (image) has no restrictions to calculate pixel values

• Working with MORE than one item requires understanding of **Mosaic_rule** and works best when using chained raster functions.

• Imagery Layers have **maximum Height and Width** -> # of pixels you can export or make calculations on -> modify pixel_size

• Way of reaching pixel values done by calculating statistics and deriving pixel count per value -> explore working with arrays
CONCLUSIONS

• Straightforward to use the service and quick to filter images
• Convenient to publish Web applications through the Esri Platforms
• API updated often (documentation is tricky)
• Sharing and reproducing your workflow easy
• Integration of both Esri and open source libraries
• User interface from ArcGIS Enterprise constantly updated and linked to Notebooks
THANK YOU!