Marcel Rodriguez-Riccelli
1/26/2026
Satellite Image Analysis (GEOG 581)
<u>LAB 2 - Image enhancements, spatial filtering, and band combinations</u>

LINK TO CODE: [GOOGLE COLLAB](), [GITHUB]()

QUESTION 1. Using print() and the typeof() JavaScript function, what is the *type* of your buffered features?

I'm using the Google Earth Engine Python API to complete my assignments, so the functions for accomplishing this and results are a bit different. Printing the type after the buffer function has been applied with the following code using the Python type class:

print("Python Type:", type(water_buffer))

outputs the following:

Python Type: <class 'ee.geometry.Geometry'>

While using the Google Earth Engine Python API .getInfo() function to return the metadata dictionary as it would appear with the print() function in Google Earth Engine's Browser IDE:

water_info = water_buffer.getInfo()
print("Google Earth Engine Type:", water_info["type"])

outputs the following:

Google Earth Engine Type: Polygon

This means that in the frame of Python, the water_buffer variable is a Google Earth Engine ee.geometry.Geometry class object, while in the frame of Google Earth Engine, it is a Polygon.

QUESTION 2. Now that you have temporally filtered your image collection, use the print() function to see how many images result from these filtering commands. How many images are in S2_filtered?

Again, with the Google Earth Engine Python API we must use a slightly different method for finding the number of images that have passed our spatial and temporal filters. I used the following code:

n_images = S2_filtered.size().getInfo()
print("Number of images in S2_filtered collection:", n_images)

to get the following output:

Number of Images in S2_filtered collection: 64

From this, we can see that there are 64 images in the filtered collection.

QUESTION 3. By applying filterBounds() three times, do the images in S2_filtered include *all*, *some*, or *none* of the three buffered geometries?

Chaining the .filterBounds() functions for each of our areas of interest effectively acts as an and condition, meaning that our image should intersect the bounds of the water_buffer polygon *AND* the forest_buffer polygon *AND* the urban_buffer polygon. All resulting images in S2_filtered should include all three buffered geometries.

QUESTION 4. What is the date of the first image in this revised ImageCollection?

Getting a human readable datetime string using the Python API takes a bit extra effort as well, but is possible by the following code:

first_date = ee.Date(S2_first.get("system:time_start")).format("YYYY-MM-dd HH:mm:ss").getInfo()
print("Date of the selected image (first in S2_filtered):", first_date)

the output of which is:

Date of the selected image (first in S2_filtered): 2017-07-15 19:22:24

This tells us that the date of our selected image, the first and therefore earliest in the collection, corresponds to the earliest date specified by our filter, July 15th, 2017.

QUESTION 5. What region of the electromagnetic spectrum is Sentinel-2's Band 8 sensitive to?

The Earth Engine Catalogue entry for Harmonized Sentinel-2 MSI: MultiSpectral Instrument, Level-1C (TOA) lists Band 8 as corresponding to the Near Infrared region of the electromagnetic spectrum, with a central wavelength of 835.1nm for Sentinel-2 A and 833nm for Sentinel-2 B.

QUESTION 6. Describe each of your three Band 8 histograms in terms of skewness, kurtosis, minimum, maximum, and range.

As with most things, the Python API again requires another method for being able to generate histograms, as the "ui" functions are not accessible. To generate each histogram used the reduceRegion function instead by the following code:

```
water_hist = SR_NIR.reduceRegion(reducer=ee.Reducer.histogram(maxBuckets=256),
    geometry=water_buffer, scale=10, bestEffort=True, maxPixels=1e9,)
```

using the buffer polygon for each land cover type as the input to the geometry argument. Then, I used matplotlib to generate a histogram for each region *(Fig. 1)*.

Region 1 - Water

> Bimodal distribution clustered ~1300 and ~3200, high kurtosis with sharp peaks, a domain with minimum of ~200 and maximum of ~4800, and a range of 0 to ~80.

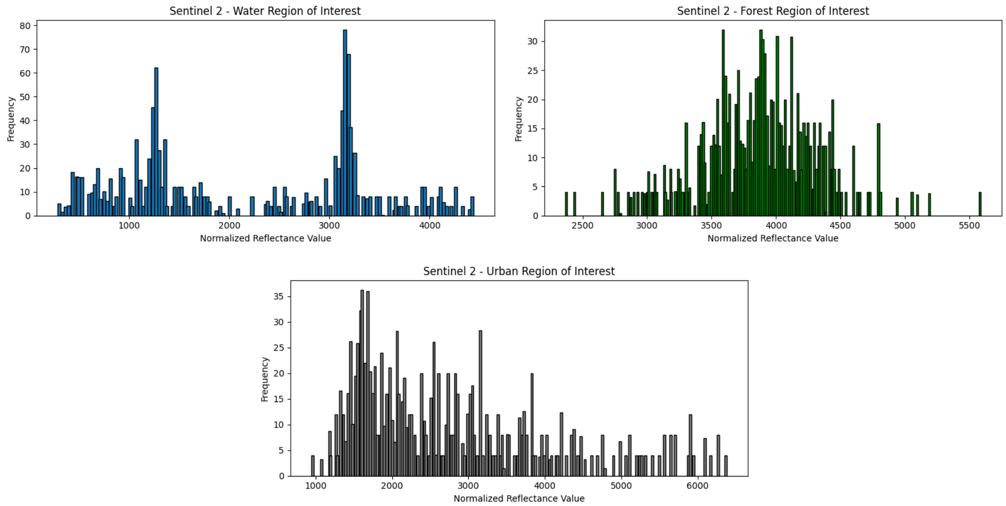

Fig. 1. A histogram showing the frequency of binned reflectance values for three regions of interest representing separate landcover types; Top Left: Water, Top Right: Forest, Bottom: Urban.

Region 2 - Forest

> A somewhat normal distribution, with a few local peaks mostly clustered around a mode at ~3900, low kurtosis with a gradual drop off from the peak area outward, a domain with a minimum of ~1500 and maximum of ~5600, and range of 0 - ~35.

Region 3 - Urban

> A positively skewed distribution with a mode at ~1600, low kurtosis with a gradual drop and scattered local peaks toward the mean, a domain with a minimum of ~900 and maximum of ~6600, and range of 0 - ~35.

QUESTION 7. What is the range of potential values for the raster “data_type” (Int, Float, etc.) used by this Sentinel-2 image’s Band 8?

By referencing the data_type > min and data_type > max keys in the .getInfo() dictionary for Band 8 (NIR), I was able to directly retrieve the range information:

Band 8 Range - Minimum: 0, Maximum: 65535

QUESTION 8. Are there distinct peaks or sections of the histogram that are associated with the dominant land cover within each of your buffered regions?

Using the inspector tool to query values at pixels within each region of interest, I can make the following observations of each:

Region 1 - Water

The buffered zone seems to extend from slightly offshore, through the surf zone, and onto the shoreface. The small amount of pixels that presumably constitute the deepest water furthest offshore tend to have lower reflectance values less than 1000, that becomes progressively higher as the region transitions into the surf zone in the cross-shore direction. This area, which occupies a large part of the center of the region, has values ~1300 that correspond to the lower peak of the bimodal distribution seen in the histogram, *(Fig. 1)*. Moving along further, the shoreface, which also occupies a large portion of the region, has values at around ~3600, corresponding to the second peak. Together, the region comprising disparate cover types explains the bimodal skew, and the uniformity of the material of these cover types explains the high kurtosis.

Region 2 - Forest

Pixels in this region of interest have values ranging from ~2000 to ~5000, with the majority of pixels showing values around ~3800. The higher range of this region of interest tracks logically with a mostly forested area, as healthy vegetation would produce high Near Infrared reflectance. This corresponds closely to the normality of the histogram *(Fig. 1)* for this region of interest.

Region 3 - Urban

Pixels in this region have a large range of values, and seem to vary across the image with high frequency, which we can expect from an urban area composed of structures and surfaces of varied materials. This would explain the highly skewed distribution seen in the histogram *(Fig. 1)* for this region of interest.

QUESTION 9. Why may we use the subtract sign in the denominator but must use subtract() in the numerator?

The variables being subtracted in the numerator are of the Earth Engine type Number, while the variable (in my case S2_NIR) being subtracted in the numerator is of the type image, meaning the subtraction operation is being applied to every pixel in the image.

QUESTION 10. What are the values of the stretched and unstretched Band 8 images at each of the three sites?

Point 1 - Water

Before Stretch: 2377
After Stretch: 6227.331401159305

Point 2 - Forest

Before Stretch: 4010
After Stretch: 10505.51069358385

Point 3 - Urban

Before Stretch: 1685
After Stretch: 4414.410353787728

QUESTION 11. What are the weights for all the cells within the kernel?

The weight of each cell in the original kernel *"two_d_kernel"* was 1, for a sum total of 8. The normalize argument, which we set to True, normalizes the cell values to sum to 1, causing each cell in the resulting kernel to have a value of 0.125.

QUESTION 12. What is the center value of the resulting kernel? What is the point of having this specific value at the center?

The center value is zero, meaning that the value of the pixel currently being operated on will not be included in the calculation of that pixel's value.

QUESTION 13. Is this kernel a low pass or high pass? Why?

This normalization causes this kernel to be a basic smoothing low pass filter; the weight of each cell becomes ⅛, and so when each weight is multiplied by the pixel value, and then those are summed, it becomes the summed value of all pixels divided by the number of pixels, or the average. This filters high frequencies in the spatial domain, such as hard edges (abrupt transitions between neighboring pixels).

QUESTION 14. For the other type of kernel (ie, low or high pass), what would a typical center value be?

A high pass kernel would feature a high center value, such as 9, and the neighboring pixels would be given values like -1. This would have the opposite effect of sharpening the image by emphasizing the differences between neighboring pixels.

QUESTION 15. In your own words, what does the convolve() function *do*?

Convolve slides the input kernel across the image and performs calculation for each pixel and outputs the resulting image.

QUESTION 16. What is the visual change to the output image compared to the band8 image?

The resulting image is quite a bit blurrier, both zoomed in and from afar; the most noticeable difference is the detail that has been lost at the edges of differing landcover types, such as the hard transitions from lakes to land.

QUESTION 17. In which of the three buffered regions do you see the largest visible change in landscape form or pattern? Does this region have higher or lower spatial frequency than the other regions?

I see the largest visible change in the Urban region. This region has a lot of high frequency spatial variation, with abrupt changes between surfaces of differing materials across small spans that is to be expected in a developed area. This is apparent both visually by seeing the tight clusters of neighboring pixels with drastically different values, and in the histogram that has a relatively large range of values with scattered and significant local peaks.

QUESTION 18. In your own words, what does the maximum filter *do* to an input image? How do the values at the three point locations change from the input Band 8 to the output_max_1 and output_max_3?

The maximum filter seems to, as it convolves across the image, for each pixel, finds the maximum value in it's neighborhood and assigns it that value. The values at each point location changes as follows from layer to layer:

Point 1 - Water

Original: 2377
Output Max 1: 6227.33154296875
Output Max 3: 6227.33154296875

Point 2 - Forest

Original: 4540
Output Max 1: 11894.01953125
Output Max 3: 11894.01953125

Point 3 - Urban

Original: 1685
Output Max 1: 4414.41015625
Output Max 3: 4414.41015625

QUESTION 19. Considering the entire image, what are the most prominent differences between `output_max_1` and `output_max_3`? Please comment on the kinds of features as well as the kinds of spectral values that show the most difference.

The most prominent difference between the 1 iteration and 3 iteration maximum filters when viewing the whole image zoomed out, are the extents of areas with uniform pixel values. Especially after the 3 iteration filter, all land features are nearly unrecognizable. Areas in the original image that had a large range of values and high frequency spatial variation such as urban areas have been resolved to smaller areas of contiguous pixel value, while other areas such as the ocean have become a giant contiguous extent of uniform value.

QUESTION 20. In your own words, why is there such a difference from only two additional applications of the maximum filter?

When the filter is applied recursively, a pixel with a high value reaches a greater extent. Upon first application, the maximum value can only reach its neighbors, but after its neighbors have been affected, the same value can then be applied in the calculation of the neighbors of each neighbor.

QUESTION 21. What are the kernel weights associated with this Laplacian kernel?

The weight of each neighbor is 1, and the weight of the central pixel is -8.

QUESTION 22. In your own words, how does a Laplacian filter "find edges" using this sort of kernel?

This is an edge finding kernel; it finds the difference between the pixel currently being operated on and the sum of its neighbors. If the difference is great, the pixel will be given a high value, and inversely a low value if the difference is low.

QUESTION 23. Describe the outputs at the three buffered geometries. In which does the Laplacian filter find the most pronounced edges?

Expectedly, the most pronounced edges are in the high spatial frequency urban areas, with abrupt changes in surface material means neighboring pixels have drastically different values. We can see structures clearly outlined, especially in downtown non-residential areas where roofing materials are more reflective.

QUESTION 24. Once you have calculated NDVI and EVI, you are left with two images, each of one band – what are the names of these bands? Where do these names come from?

The NDVI image lists a single band with the ID *"nd"*, while the EVI image lists a band with the ID *"B5"*. Presumably, the .normalizedDifference() function modifies the band in the resulting image to be *"nd"*, while the .expression() function uses the name first band provided, which in this case was Near Infrared Band 5 *"B5"*.

QUESTION 25. Based on your visual assessment of images, is there a strong positive correlation between NDVI and EVI for this image? What regions or features within the image show the greatest differences?

There does seem to be a strong positive correlation between the NDVI and EVI data for this image. The largest differences seem to be aquatic features, which is to be expected given the additional term including reflectance in the Blue visible band in the denominator.

QUESTION 26. Reference <u>this</u> script to include a legend in your script that distinguishes low, medium and high NDVI values. Choose colors that align with low, medium, high vegetation for cartographic integrity.

Below is my map of NDVI and EVI with the legend included, with NDVI on the right and EVI on the left *(Fig. 2)*.

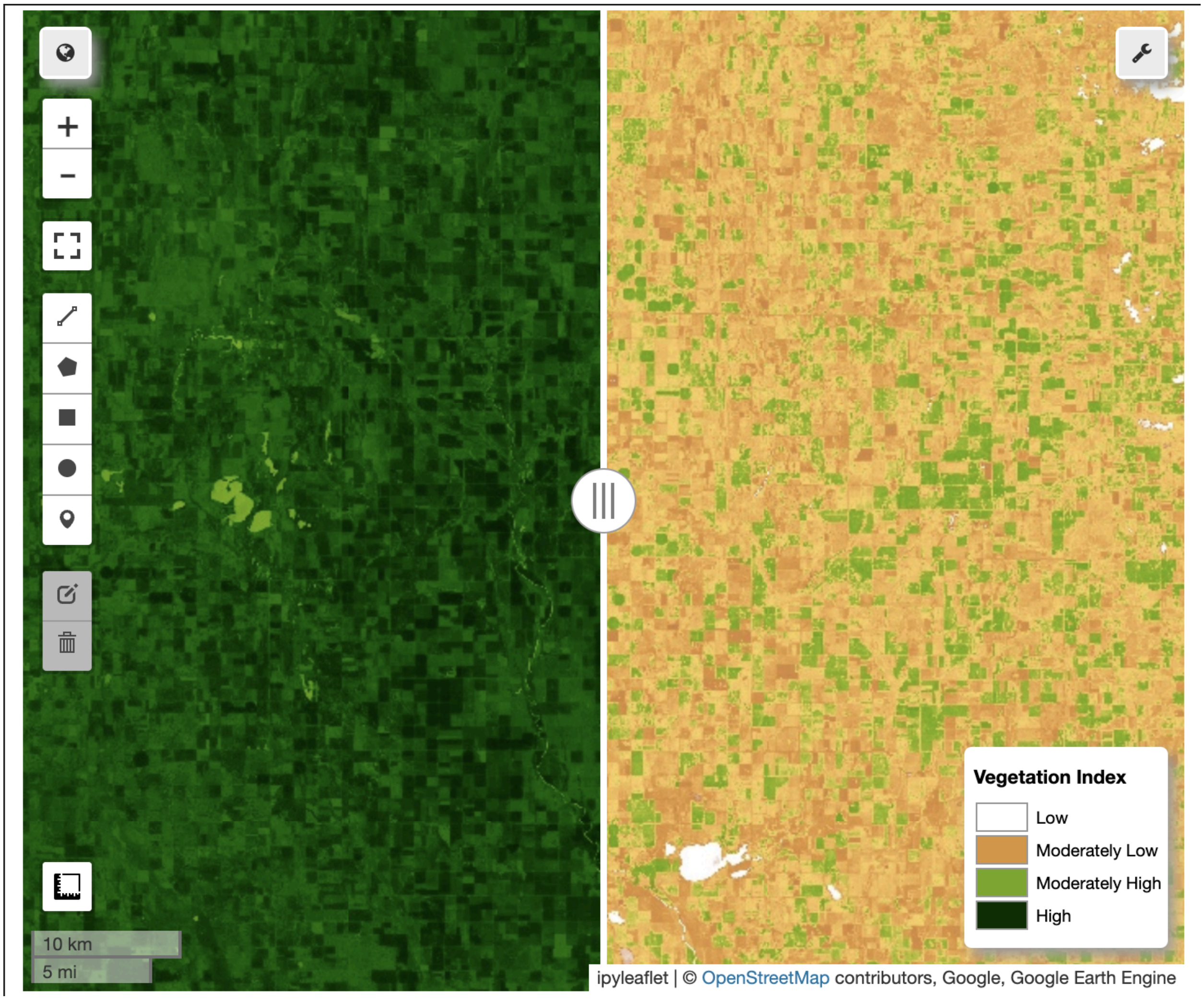

Fig. 2. Comparison of different vegetation indices derived from Landsat 8 imagery of Eastern South Dakota, Left: NDVI, Right: EVI

QUESTION 27. Based on your assessment of the scatter plot, how correlated are NDVI and EVI for this image? Are NDVI and EVI more or less correlated for low NDVI values? What about high NDVI values?

NDVI and EVI are very highly correlated for this image, with the points tightly grouped throughout and following a similar narrow trend *(Fig. 3)*. For low NDVI values, EVI remains relatively flat while NDVI values increase, but the trend switches as NDVI becomes high, creating a slight curve. This makes sense, as we know NDVI saturates at high value, and that EVI compensates for atmospheric constituents by including the blue band as a variable, and so the presence of aquatic bodies would result in larger relative EVI than NDVI at pertinent pixels.

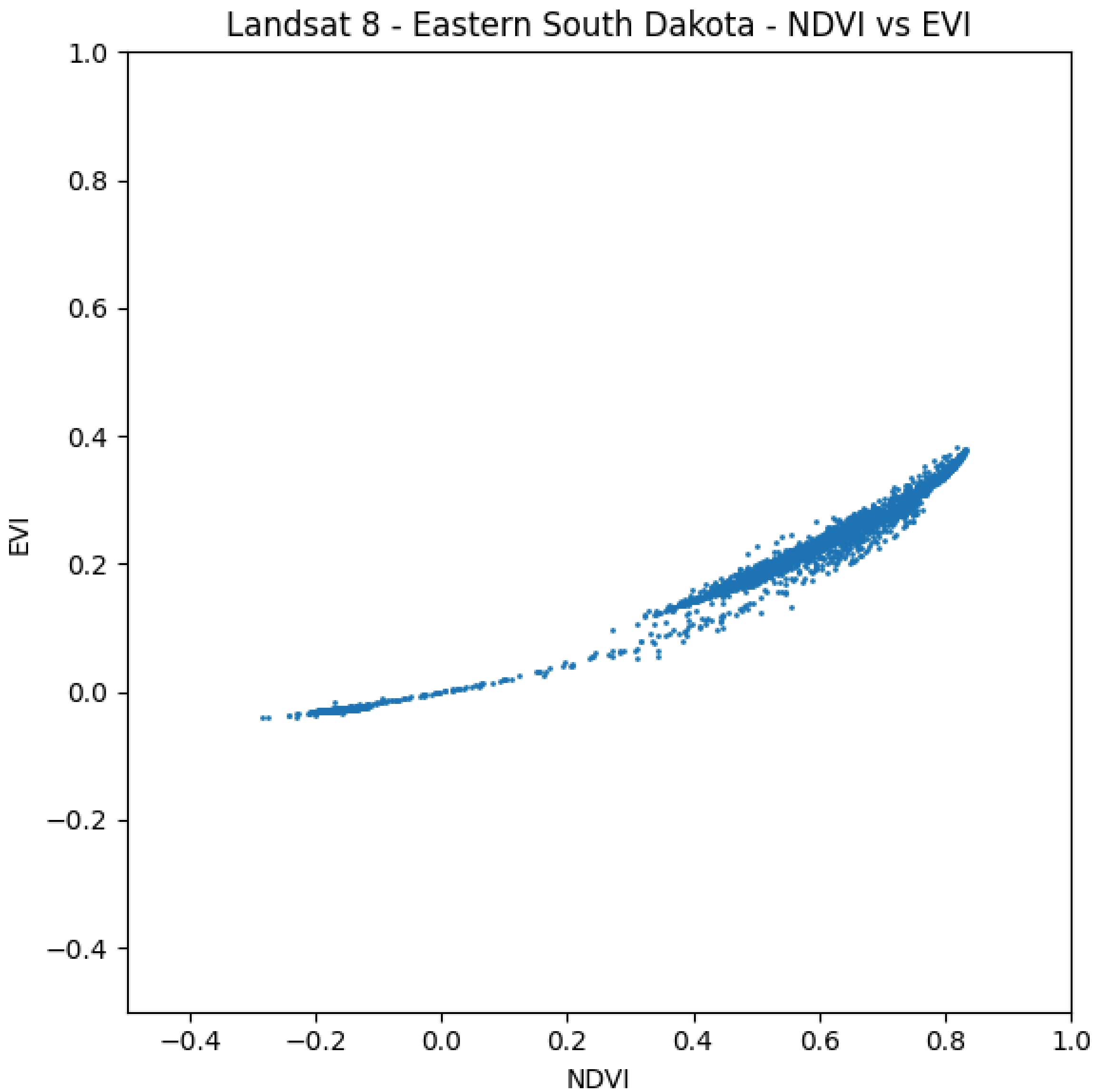

Fig. 3. Scatter plot showing the relationship between NDVI and EVI derived from Landsat 8 imagery of Eastern South Dakota.

QUESTION 28. Based on your visual assessment of this 'difference' image, what regions or features within the image show great difference?

We can see from the map below (Fig. 4) that our previous inferences and visual analyses were correct, and that the aquatic bodies show the greatest difference.

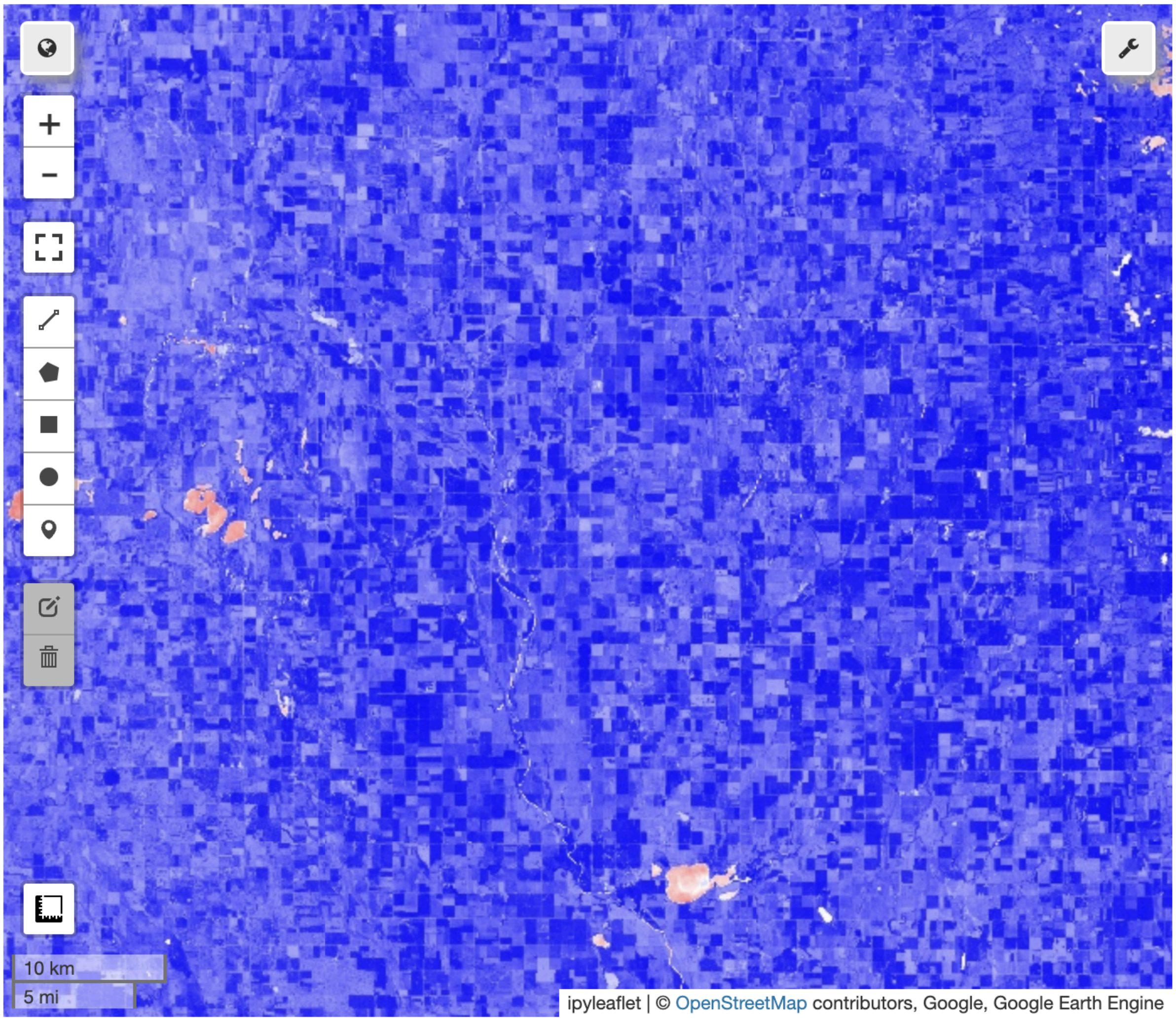

Fig. 3. A map showing the difference between NDVI and EVI values derived from Landsat 8 imagery of Eastern South Dakota (red symbolizes larger difference, while blue represents less difference).