Programming for Geo Informatics - Lab 4

Satellite image processing

Submitted By :

Ashwin E

SC24M136

1. Download Sentinel data for your city.

In [207…

```
import rasterio
from rasterio.merge import merge
from rasterio.plot import show
import numpy as np
#path = r"A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab\lab4\Browser_i
#path = r"A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab\lab4\S1A\s1a\m
#path = r"A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab\lab4\S1A\s1a\m
#path = r"A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab\lab4\data\2024
#path = r"A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab\lab4\s2qa\s2wa
#path = r"A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab\lab4\s2qa\s2wa
#path = r"A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab\lab4\data\2024
path = r"A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab\lab4\Browser_im
sentinel_data=rasterio.open(path)
```

2. Read the raster file and gather basic information (dimension of data, number of bands,spatial resolution, projection system).

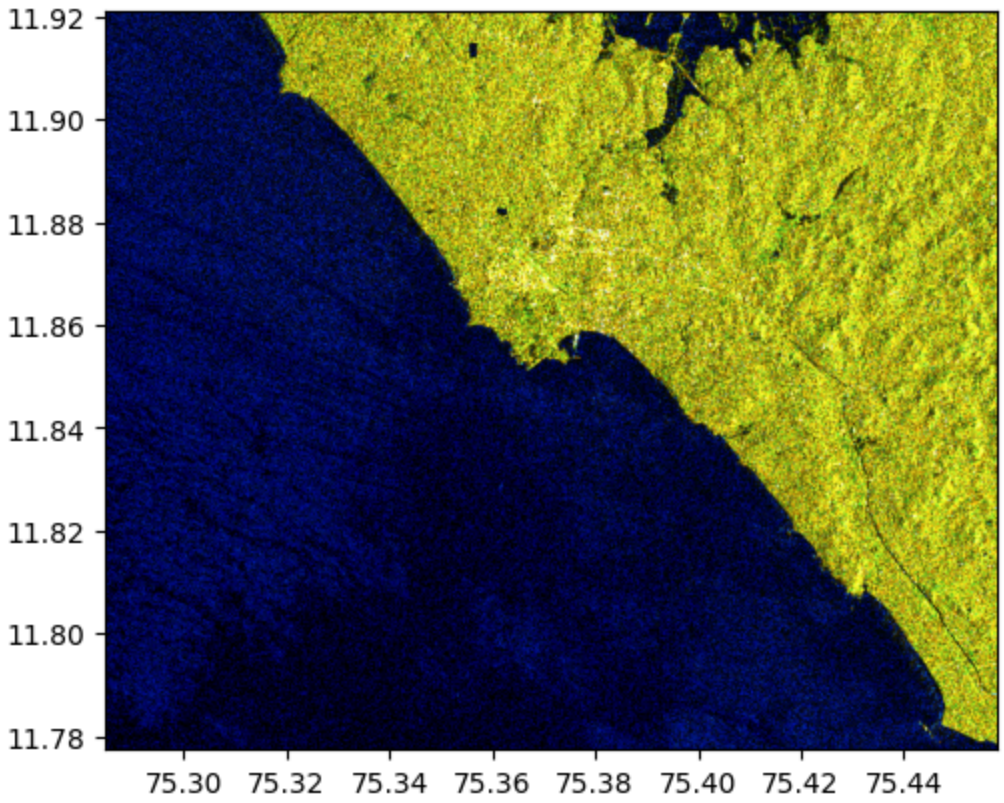
In [208…

```
print(f"Dimension : {sentinel_data.height} (Height)  X  {sentinel_data.width} (Widt
print(f"Projection System : {sentinel_data.crs}")
print(f"Spatial Resolution : {sentinel_data.res}")
print(f"Number of Bands : {sentinel_data.count}")
```

```
Dimension : 1057 (Height)  X  1250 (Width)
Projection System : EPSG:4326
Spatial Resolution : (0.00013897680000000037, 0.00013601797540208060)
Number of Bands : 3
```

In [209…

```
show(sentinel_data)
```



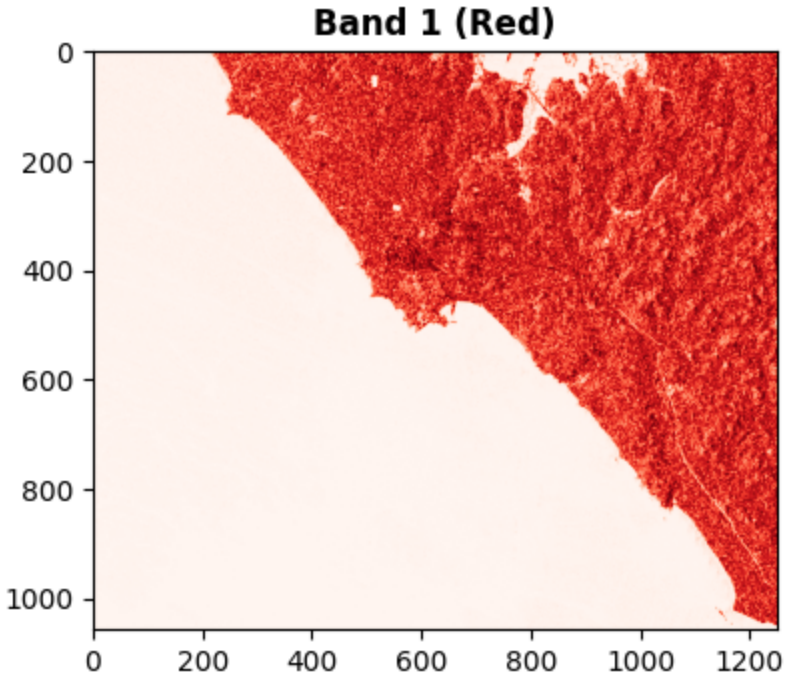
Out[209… <Axes: >

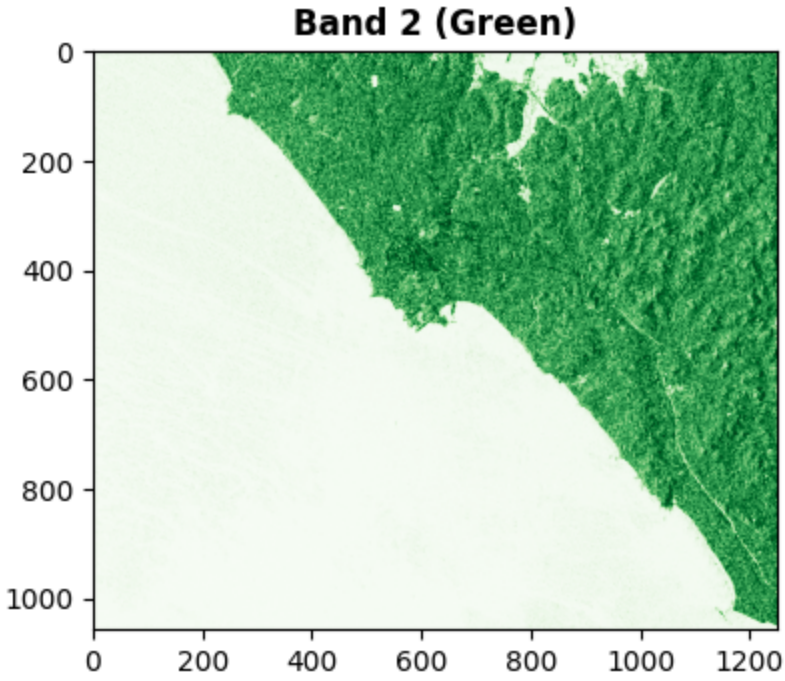
3. Stack the individual bands and form a single image file.

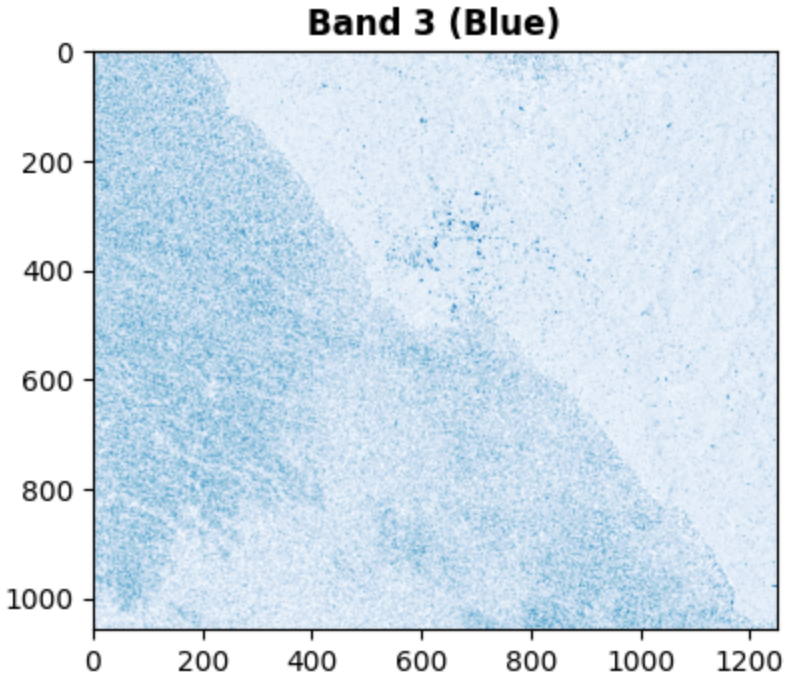
In [210…

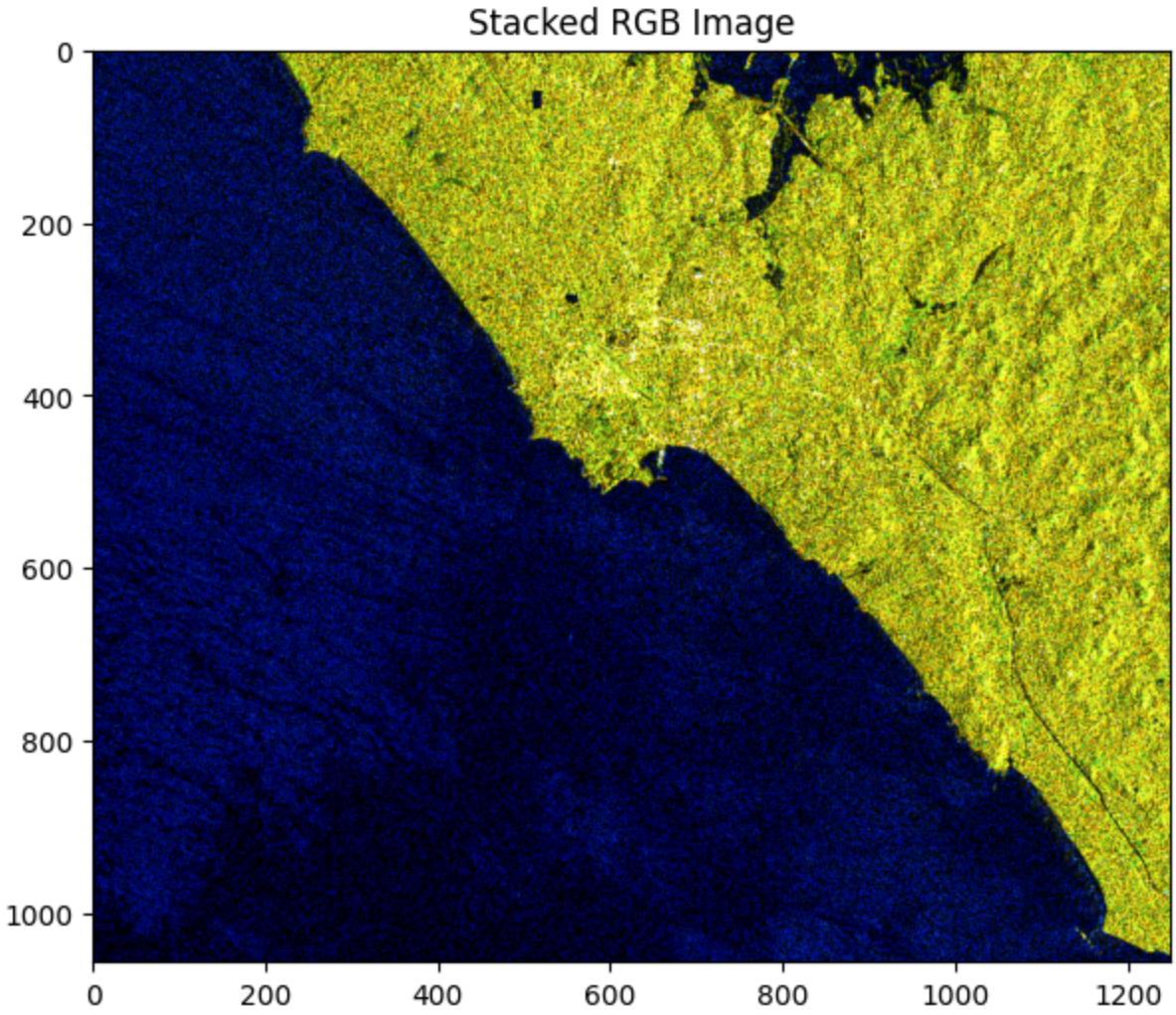
```
with rasterio.open(path) as dataset:
    num_bands = dataset.count
    print(f"Number of bands in the image: {num_bands}")
    band1 = dataset.read(1)
    band2 = dataset.read(2)
    band3 = dataset.read(3)
    plt.figure(figsize=(15,5))
    plt.subplot(1, 3, 1)
    show(band1, cmap='Reds', title='Band 1 (Red)')
    plt.figure(figsize=(15,5))
    plt.subplot(1, 3, 2)
    show(band2, cmap='Greens', title='Band 2 (Green)')
    plt.figure(figsize=(15,5))
    plt.subplot(1, 3, 3)
    show(band3, cmap='Blues', title='Band 3 (Blue)')
    plt.show()
    stacked_image = dataset.read([1, 2, 3])

plt.figure(figsize=(7, 7))
plt.imshow(stacked_image.transpose(1, 2, 0))
plt.title('Stacked RGB Image')
plt.show()
```

```
Number of bands in the image: 3
```


Band 1 (Red)
0
200
400
600
800
1000
0
200
400
600
800
1000
1200


Band 2 (Green)
0
200
400
600
800
1000
0
200
400
600
800
1000
1200





4. Create a function to create stack of bands which will take input of bands and provide stacked output.

In [211…

```python
#BANDS FROM DIFFERENT FILES
def stack_bands(band_paths, output_path):
    band_list = []
    with rasterio.open(band_paths[0]) as src:
        profile = src.profile
    for band_path in band_paths:
        with rasterio.open(band_path) as src:
            band_list.append(src.read(1))
    stacked_image = np.stack(band_list, axis=0)
    print("Shape of Stacked File : ", stacked_image.shape)
    profile.update(count=len(band_paths))
    with rasterio.open(output_path, 'w', **profile) as dst:
        dst.write(stacked_image)
    print(f"Stacked image saved at: {output_path}")

band_files = [
r"A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab\lab4\Browser_images\20
r"A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab\lab4\Browser_images\20
r"A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab\lab4\Browser_images\20
r"A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab\lab4\Browser_images\20
]

stack_bands(band_files,r"A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab
```

```
Shape of Stacked File :  (4, 1057, 1250)
Stacked image saved at: A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab\l
ab4\output.tiff
```

In [212…

```python
#INDIVIDUAL BANDS FROM A SINGLE FILE
def stack_bands(band_paths):
    bands = []
    for path in band_paths:
        with rasterio.open(path) as raster_data:
            bands.append(raster_data.read(1))
            stacked_image = np.stack(bands, axis=0)
    return stacked_image

stacked_image = stack_bands(band_files)
print("Shape of stacked image:", stacked_image.shape)
```

```
Shape of stacked image: (4, 1057, 1250)
```

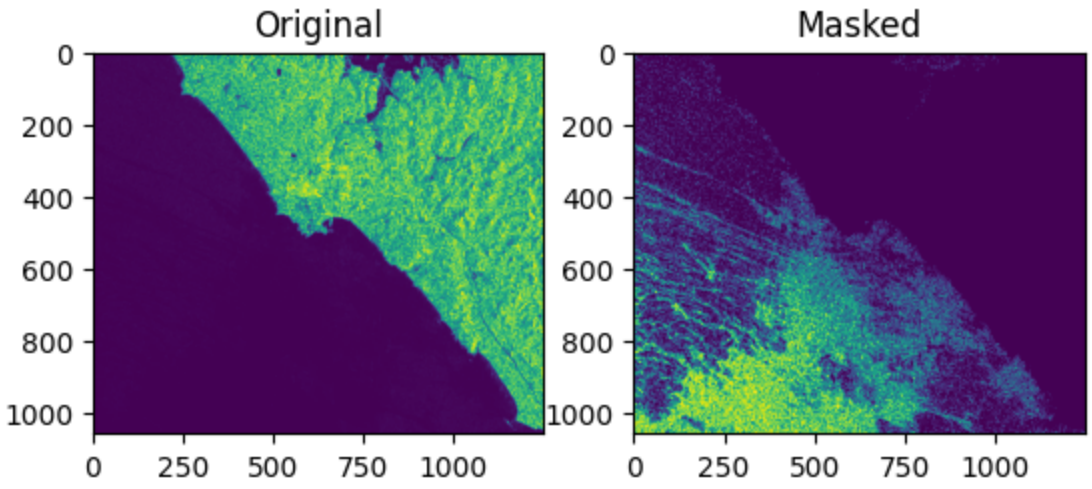
5. Find out the most repeating value in the stacked image or single band and generate binary mask which includes two classes, most repeating value and rest of the values. Copy location information(extent) from stacked image to this masked output.

In [213…

```python
import rasterio
import numpy as np
import rasterio.plot
```

```
import matplotlib.pyplot as plt

stacked_image_path=r"A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab\lab
with rasterio.open(stacked_image_path) as src:
    data = src.read(1)
    print(src.count)
    most_common_value = np.argmax(np.bincount(data.ravel()))
    binary_mask = np.where(data == most_common_value, 1, 0)
    plt.subplot(1,2,1)
    plt.title("Original")
    plt.imshow(src.read(1))
    plt.subplot(1,2,2)
    plt.title("Masked")
    plt.imshow(binary_mask)
```

```
4
```

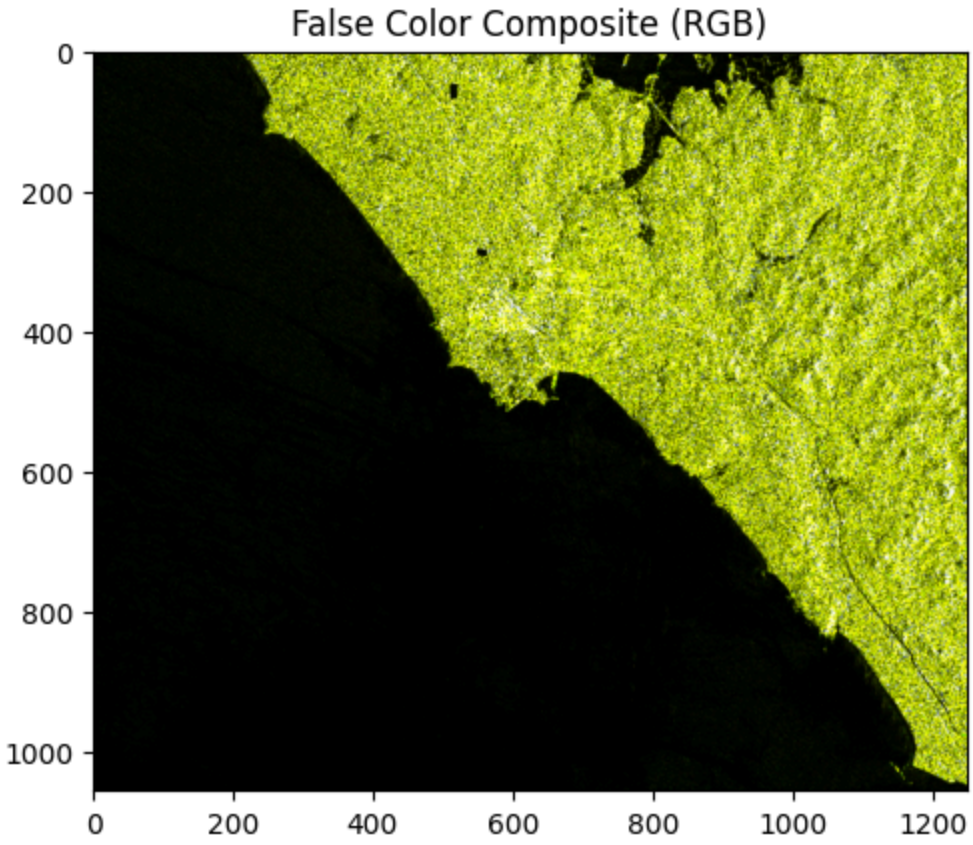


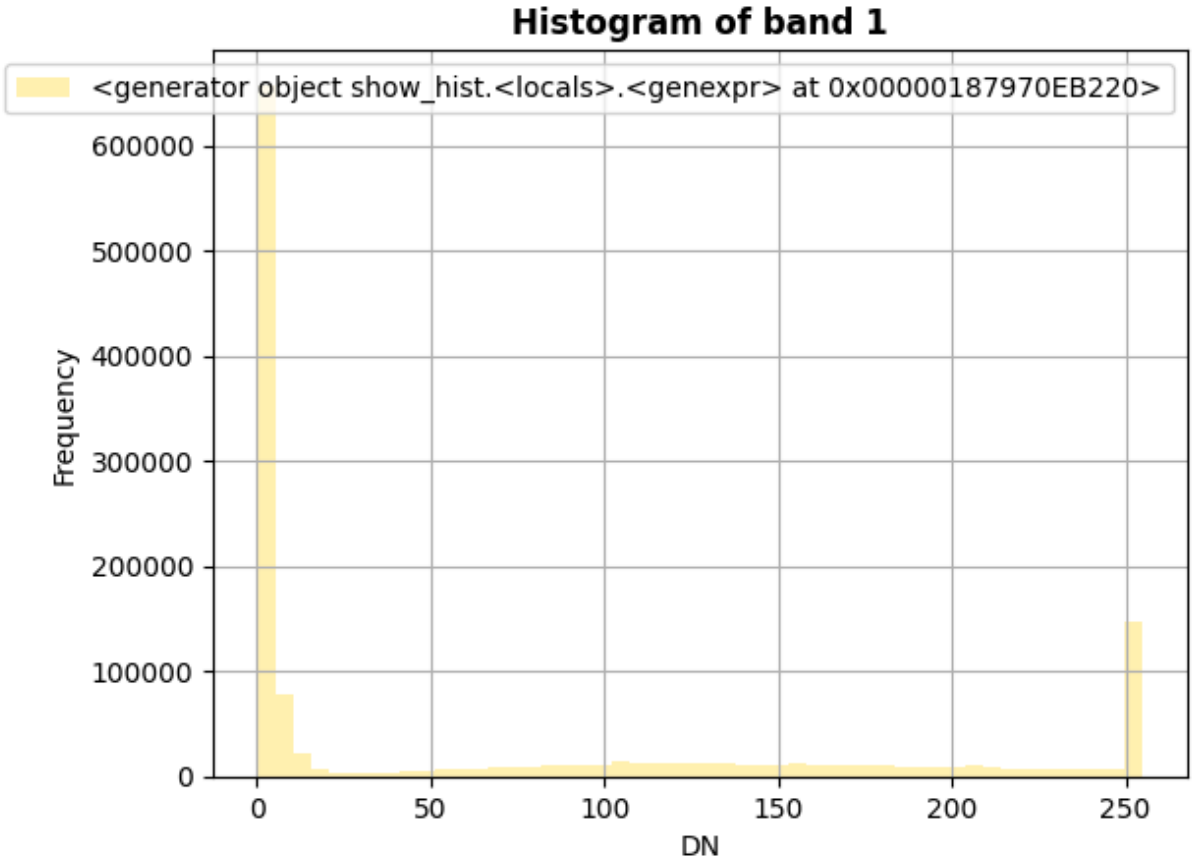
6. Display the histogram of each band and create a false color composite.

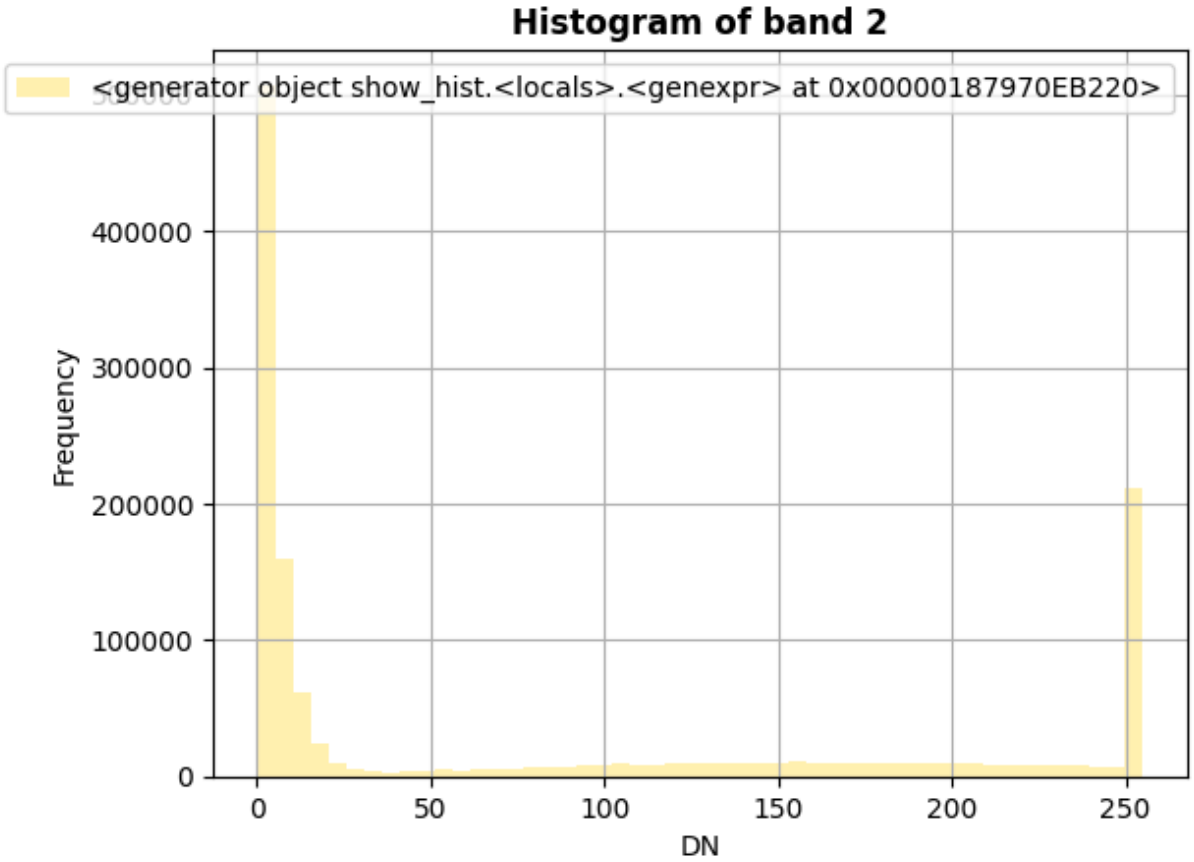
In [214…

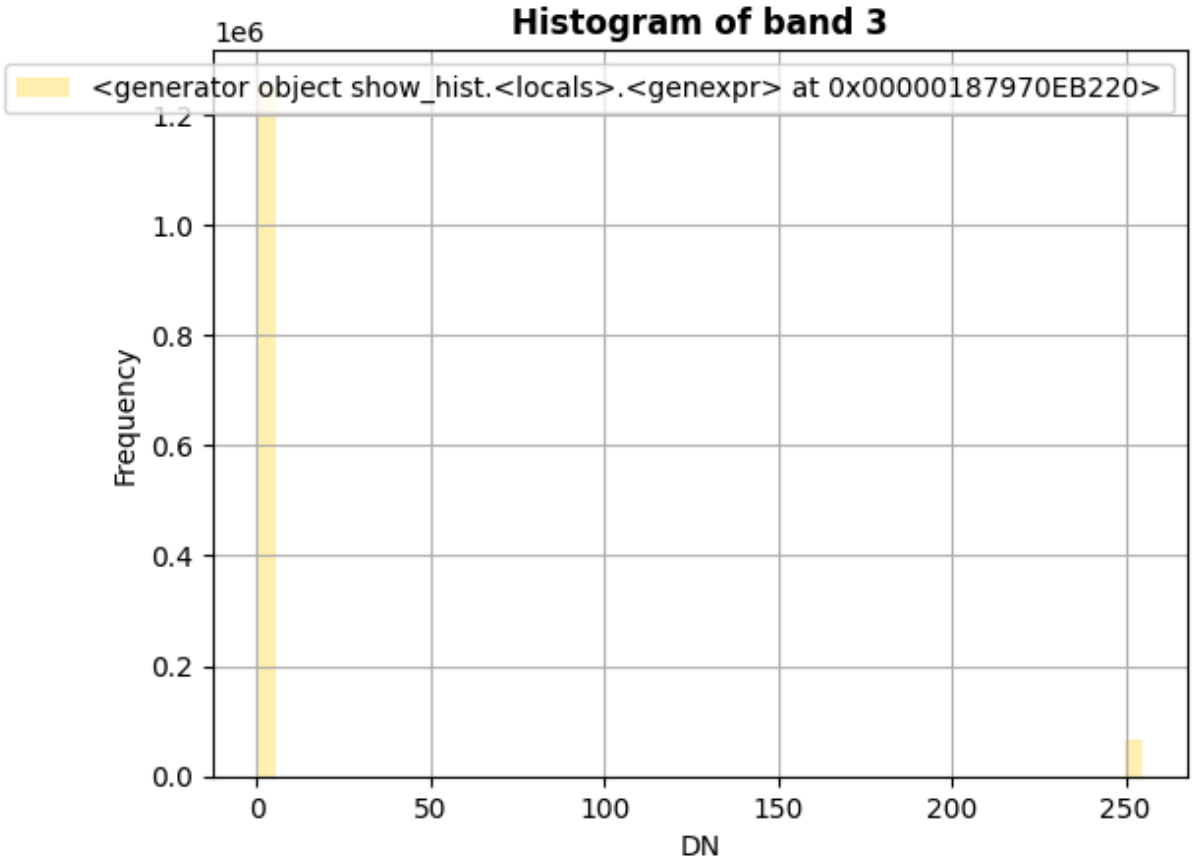
```
import rasterio
import numpy as np
import matplotlib.pyplot as plt
from rasterio.plot import show_hist
stacked_image_path=r"A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab\lab
dataset = rasterio.open(stacked_image_path)
red_band = dataset.read(1)
green_band = dataset.read(2)
blue_band = dataset.read(3)

def scale_min_max(array):
    return (array - np.nanmin(array)) / (np.nanmax(array) - np.nanmin(array))

red_normalized = scale_min_max(red_band)
green_normalized = scale_min_max(green_band)
blue_normalized = scale_min_max(blue_band)
rgb_normalized = np.dstack((red_normalized, green_normalized, blue_normalized))
plt.imshow(rgb_normalized)
plt.title("False Color Composite (RGB)")
```

```
plt.show()

for i in range(1,4):
    show_hist(dataset.read(i), bins=50, lw=0.0, stacked=False, alpha=0.3, histtype=
```




Histogram of band 1
<generator object show_hist.<locals>.<genexpr> at 0x00000187970EB220>
Frequency
600000
500000
400000
300000
200000
100000
0
0
50
100
150
200
250
DN


Histogram of band 2
<generator object show_hist.<locals>.<genexpr> at 0x00000187970EB220>
Frequency
400000
300000
200000
100000
0
0
50
100
150
200
250
DN



7. Clip and save a subset of data.

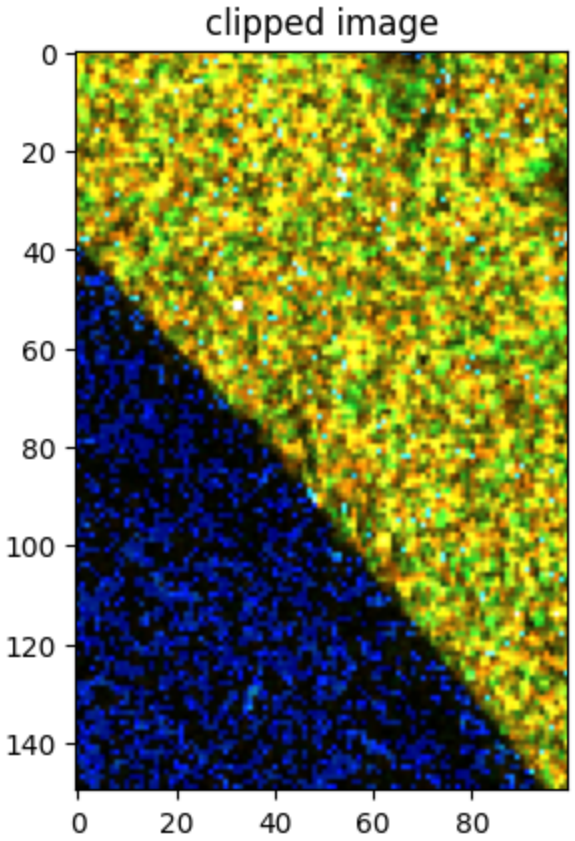
In [215…

```
import matplotlib.image
import rasterio
import matplotlib.pyplot as plt
import matplotlib
import numpy as np
filepath=r"A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab\lab4\Browser_
with rasterio.open(filepath) as dataset:
    subset_window = rasterio.windows.Window (300, 100, 100, 150)
    subset_data = dataset.read(window=subset_window)
print(subset_data.shape)
subset_data=np.transpose(subset_data, (1,2,0))
plt.title('clipped image')
plt.imshow(subset_data)
```

```
(3, 150, 100)
```

Out[215…

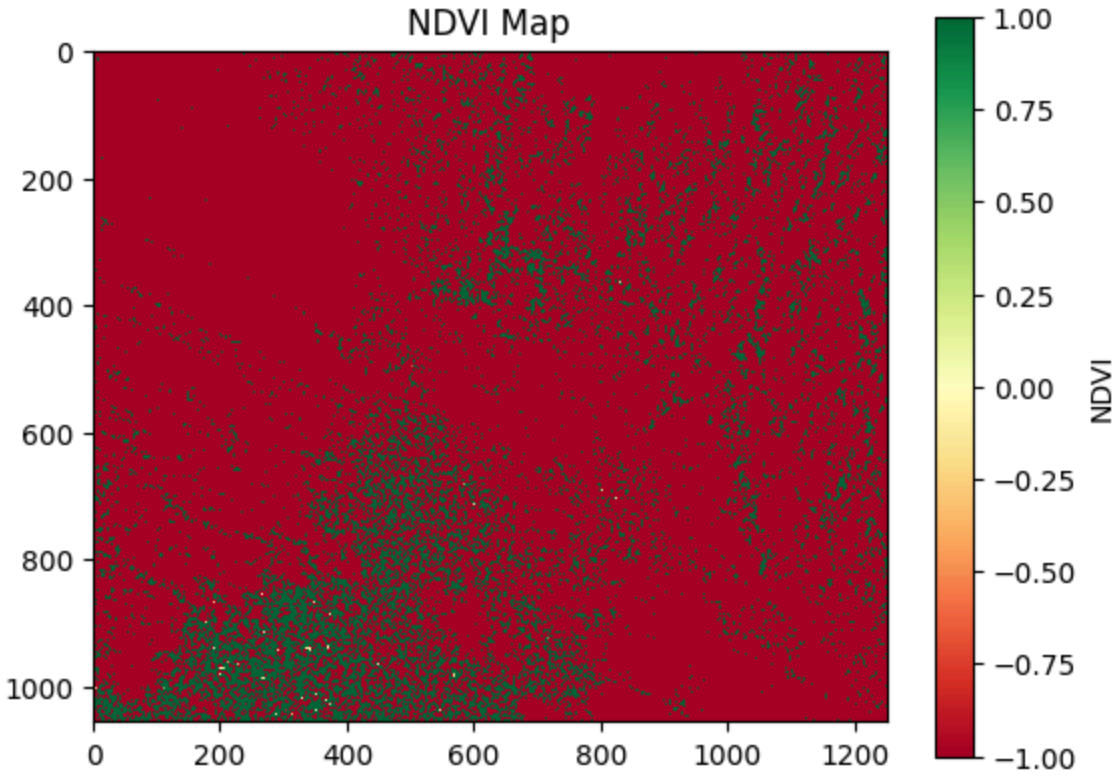
```
<matplotlib.image.AxesImage at 0x1879b4839b0>
```



8. Create a NDVI map of your study area and display the results. Derive basic statistics from your NDVI file.

In [220…

```
def calculate_ndvi(nir_band, red_band):
    nir_band = nir_band.astype(float)
    red_band = red_band.astype(float)
    denominator = nir_band + red_band
    with np.errstate(divide='ignore', invalid='ignore'):
        ndvi = (nir_band - red_band) / denominator
        ndvi = np.nan_to_num(ndvi, nan=-9999)
        ndvi = np.clip(ndvi, -1, 1)
        ndvi = ((ndvi + 1) * 32767.5).astype(np.int16)
    return ndvi

ndvi_map = calculate_ndvi(stacked_image[1], stacked_image[0])
plt.imshow(ndvi_map, cmap='RdYlGn', vmin=-1, vmax=1)
plt.colorbar(label='NDVI')
plt.title('NDVI Map')
plt.show()
print("NDVI Min:", np.nanmin(ndvi_map))
print("NDVI Max:", np.nanmax(ndvi_map))
print("NDVI Mean:", np.nanmean(ndvi_map))
```




```
NDVI Min: -32705
NDVI Max: 32767
NDVI Mean: -10628.964977861873
```

9. Create a function which will automate generation of NDVI map.

In [228…

```python
def generate_ndvi(band_paths):
    stacked_image = stack_bands(band_paths)
    ndvi_map = calculate_ndvi(stacked_image[3], stacked_image[2])
    with rasterio.open(band_paths[0]) as src:
        profile = src.profile
    with rasterio.open(r"A:\IIST GEO INFORMATICS\Programming for geoinformatics Lab
        ndvi_ds.write(ndvi_map, 1)
    return  True

if generate_ndvi(band_files):
    print("Generated NDVI Map")
```

```
Generated NDVI Map
```

Learning Outcomes : 1 : Learnt how to process and manipulate satellite image data using . 2 : Learnt to work with numpy, matplotlib and rasterio for satellite image processing . 3 : Learnt to create NDVI map, binary masks and False Color Composites .