Using Rasterio for Tiff image:

***Documentation for Functions and libraries used in codes has been provided in the end of the document for further reference.***

1>

im = rs.open("/content/image\_2016-01-03.tiff")

arr = im.read()

arr.shape

Here function read is used to access images Numpy array (N-dimensional array to be precise)

Shape method is used to images height, width and its content (RGB[It is a type of colour band] and other colour bands)

2>

from rasterio.plot import show

#to display RGB

dataset = rs.open('/content/image\_2016-01-03.tiff')

show(dataset.read([1,2,3]))

show(dataset.read(1))

here show is a function which is used for plotting the numpy array into a 2-d graph

3>

import rasterio

from rasterio.plot import show

import numpy as np

import os

%matplotlib inline

# Data dir

# Filepath

# Open the file:

raster = rs.open("/content/image\_2016-01-03.tiff")

# Plot band 1

show((raster, 1))

This is also similar to above function but will give more accurate plot of graph at only one colour band (Either Red or Green or Blue [show((raster, 1))])

4>

import matplotlib.pyplot as plt

%matplotlib inline

# Initialize subplots

fig, (ax1, ax2, ax3) = plt.subplots(ncols=3, nrows=1, figsize=(10, 4), sharey=True)

# Plot Red, Green and Blue (rgb)

show((raster, 4), cmap='Reds', ax=ax1)

show((raster, 3), cmap='Greens', ax=ax2)

show((raster, 1), cmap='Blues', ax=ax3)

# Add titles

ax1.set\_title("Red")

ax2.set\_title("Green")

ax3.set\_title("Blue")

Matplotlib is a python library used for plotting graphs,

Here we use 3 subplots for differentiating red, green and blue colour band and use the show function (Discussed earlier in point 2).

5>

dem = rasterio.open("/content/image\_2016-01-03.tiff")

fig, ax = plt.subplots(figsize=(10,10))

image\_hidden = ax.imshow(dem.read()[0])

fig.colorbar(image\_hidden, ax=ax)

rasterio.plot.show(dem, ax=ax)

This shows the colour gradient in an image using the colorbar function of rasterio

6>

def extract\_along\_line(xarr, line, n\_samples=256):

    profile = []

    for i in range(n\_samples):

        # get next point on the line

        point = line.interpolate(i / n\_samples - 1., normalized=True)

        # access the nearest pixel in the xarray

        value = xarr.sel(x=point.x, y=point.y, method="nearest").data

        profile.append(value)

    return profile

import rioxarray

import shapely.geometry

import matplotlib.pyplot as plt

sample\_tif = ('/content/image\_2016-01-03.tiff')

# Load xarray

tile = rioxarray.open\_rasterio(sample\_tif).squeeze()

# create a line (here its the diagonal of tile)

line = shapely.geometry.MultiLineString([[

            [tile.x[-1],tile.y[-1]],

            [tile.x[0], tile.y[0]]]])

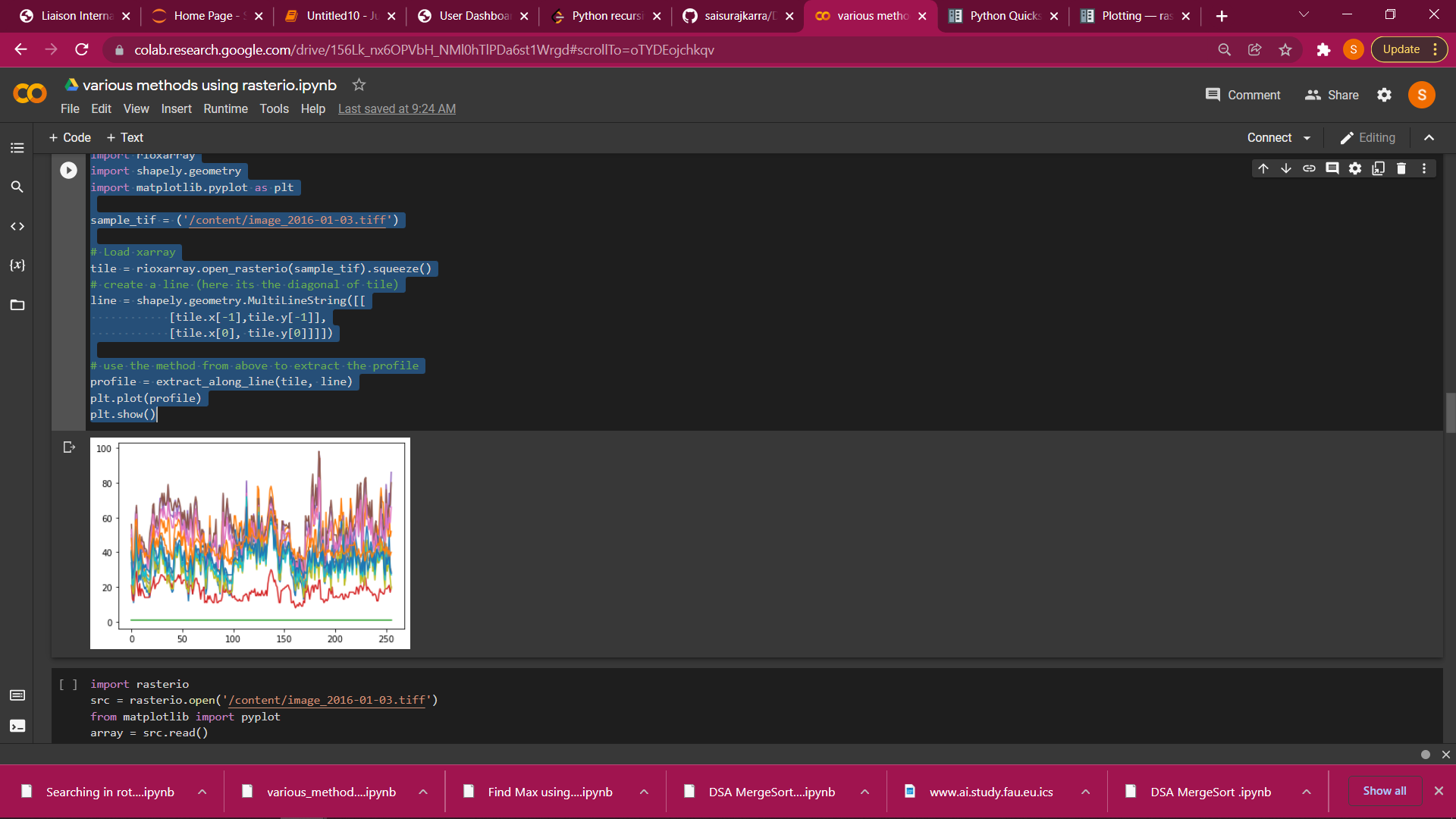
# use the method from above to extract the profile

profile = extract\_along\_line(tile, line)

plt.plot(profile)

plt.show()

The above code is an excellent source to view the histogram of color gradients that’s been used in the picture



7>

import numpy

ds = rs.open("/content/image\_2016-01-03.tiff")

array = ds.read(1) # Type depends on the type you want for the average.

avg = numpy.average(array)

print(avg)

This programs gives the average value of the pixels we get

8>

for x in range (0,340,1):

    for y in range(0,480,1):

        color = ds.read()

        print(color.mean())

here x is the height of image in pixels and y is the width of pixel, we find out the average value of the rgb

9>

pix\_val = list(ds.read())

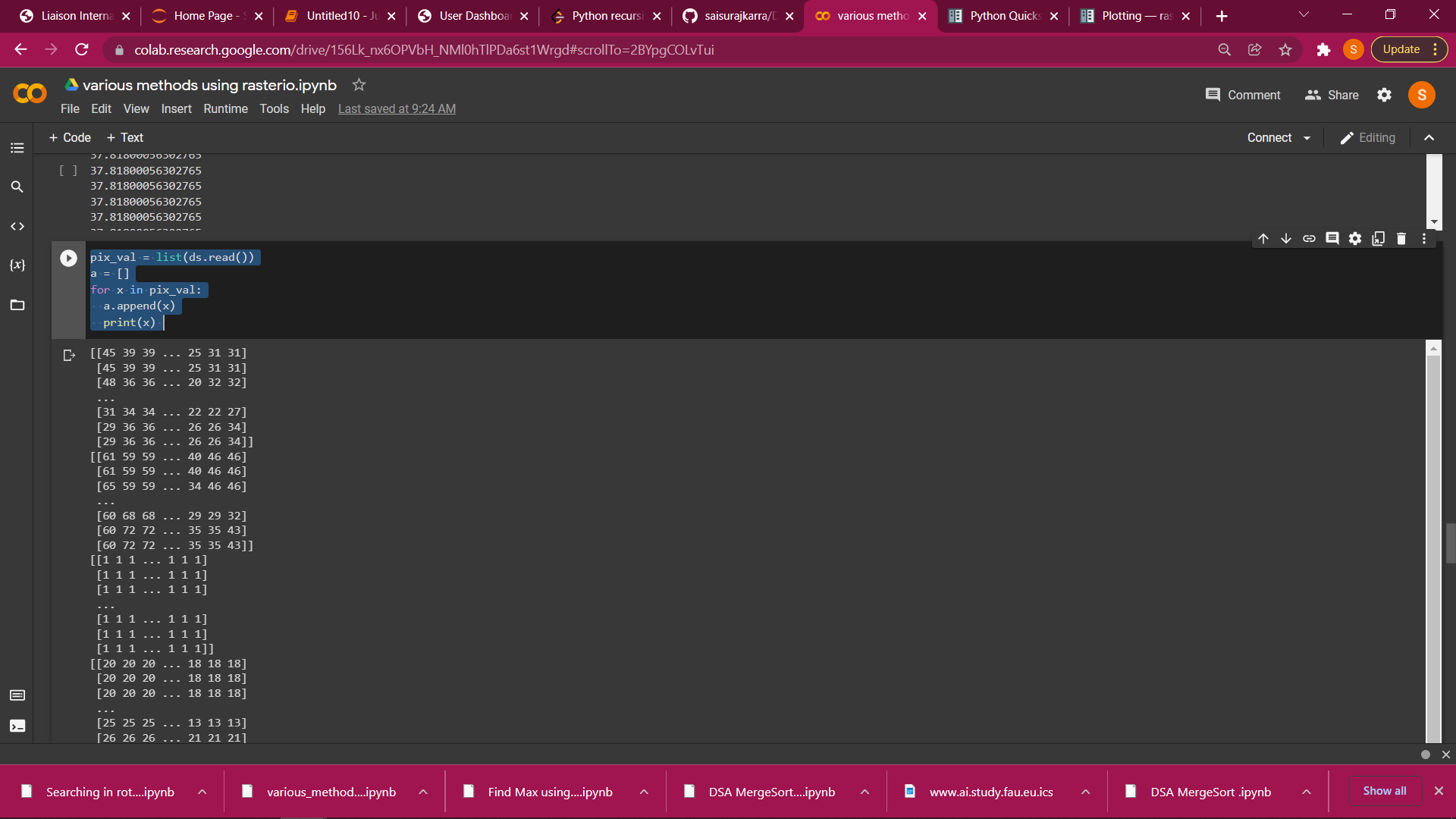
a = []

for x in pix\_val:

  a.append(x)

  print(x)

This code gives all the RGB and Colour band values in the form of array



10>

for x in range (0,340,1):

    for y in range(0,480,1):

        color = ds.read()

        print(color)

similar to 9 but gives all the value of colour bands (RGB and others) in form of array

Libraries documentation for further reading

<https://rasterio.readthedocs.io/en/latest/quickstart.html> (Rasterio Read,shape)

<https://rasterio.readthedocs.io/en/latest/topics/plotting.html> (Rasterio Plotting [Show function to be precise])

<https://numpy.org/doc/stable/> (Numpy)

<https://matplotlib.org/stable/users/index.html> (matplotlib)

<https://corteva.github.io/rioxarray/stable/> (Rioxarray)

<https://docs.python.org/3/library/os.html> (OS)

<https://www.geeksforgeeks.org/matplotlib-pyplot-colorbar-function-in-python/#:~:text=colorbar()%20in%20python,Axes%20or%20list%20of%20Axes>. (colorbar)

reference for the code work

* <https://geohackweek.github.io/raster/04-workingwithrasters/>
* <https://stackoverflow.com/questions/65316627/comparing-values-in-two-arrays-from-rasterio-and-performing-operations>
* <https://stackoverflow.com/questions/42060367/how-to-use-getpixel-function-in-gdal-python-for-geotiff-images>
* <https://stackoverflow.com/questions/60127026/python-how-do-i-get-the-pixel-values-from-an-geotiff-map-by-coordinate>
* <https://gis.stackexchange.com/questions/190423/getting-pixel-values-at-single-point-using-rasterio>
* <https://answerbun.com/geographic-information-systems/using-rasterio-to-show-rgb-image-with-nodata/>
* <https://automating-gis-processes.github.io/CSC/notebooks/L5/plotting-raster.html>
* <https://gis.stackexchange.com/questions/338074/plot-rgb-using-rasterio#:~:text=adjust%20('linear'%20%7C%20None),no%20adjustment%20will%20be%20applied>.