**Downloading the sentinel data for Dharamapuri district and creating NDVI**

**Step 1 – installing the required library and importing it**

**1. Installing the required Libraries**

!pip·install·sentinelsat

!pip·install·rasterio

!pip·install·folium

!pip·install·geopandas

!pip·install·descartes·

**2.Importing**

import folium

import os

import numpy as np

from sentinelsat import SentinelAPI

import geopandas as gpd

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from shapely.geometry import MultiPolygon, Polygon

import rasterio as rio

from rasterio.plot import show

import rasterio.mask

**3.Loging in to the copernicus data hub**

user = 'ss-gis'

password = 'ssgis@3045'

api = SentinelAPI(user, password, 'https://scihub.copernicus.eu/dhus'

**Step – 2 Reading the Shapefile with Geopandas and visualize it with Folium python library.**

**1.Downloading the boundary file**

!wget https://www.dropbox.com/s/g24n4lme4722mfo/data.zip

!unzip 'data.zip'

**2.Reading the layer**

boundary = gpd.read\_file('data/Dharmapuri/Dharmapuri.shp')

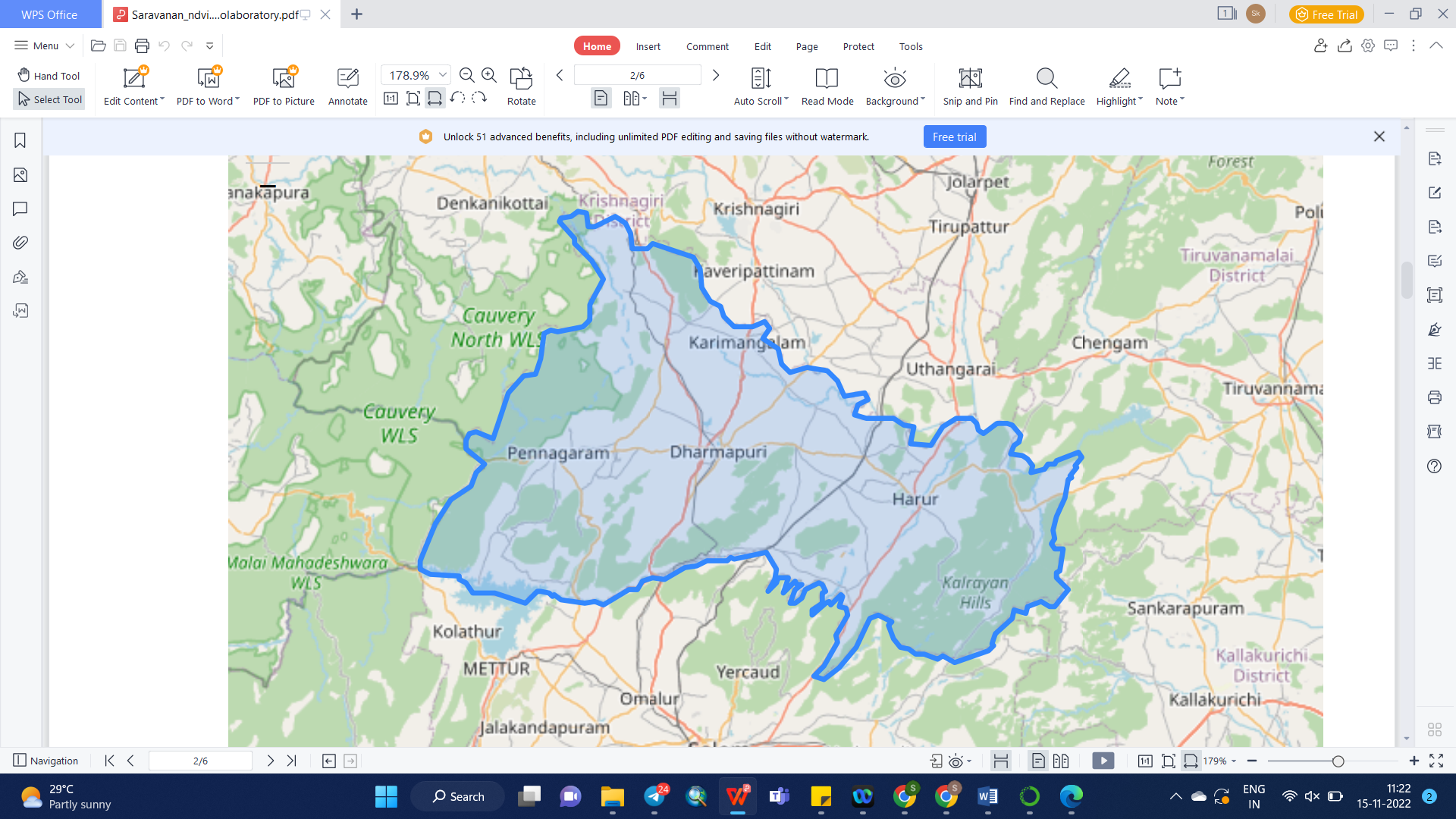
boundary

**3.Displaying the layer**

map = folium.Map([12.11, 78.2091], zoom\_start=9)

folium.GeoJson(boundary).add\_to(map)

map



**4.Creating the footprint**

footprint = None

for i in bound['geometry']:

footprint = i

**Step – 3 Downloading the Sentinel data**

**1.Searching the sentinel data**

Created a query for Sentinel 2 images Level 2A with cloud coverage between 0 and 10 that fall or intersect with the footprint (Area of study). For the time period, we are interested in Sentinel Level 2A satellite images taken between '20220501' and '20220530'

products = api.query(footprint,

date = ('20220501', '20220530'),

platformname = 'Sentinel-2',

processinglevel = 'Level-2A',

cloudcoverpercentage = (0,10))

**2.Createing a GeodataFrame or Dataframe from the product dictionary and sort**

**them according to cloud coverage percentage.**

products\_gdf = api.to\_geodataframe(products)

products\_gdf\_sorted = products\_gdf.sort\_values(['cloudcoverpercentage'], ascending=[True])

products\_gdf\_sorted

**3. Downloading the satellite image**

Let us say we are interested in the first satellite image since this has the least cloud coverage of all available images. we can simply call download and provide the product name

api.download("053a906a-6f8d-4769-9a20-47432d197d96")

**Step – 4 NDVI**

To calculate the NDVI, we need Red band and Near-Infrared Band (NIR).Sentinel Images have red in 4th band and NIR in the 8th band.

The formula for NDVI calculation is: nir - red /(nir + red). First we need to read the 4th and 8th bands as arrays and creating NDVI Index using the above

formula.

b4 = rio.open(source+'/T43PHP\_20220528T050701\_B04.jp2')

b8 = rio.open(source+'/T43PHP\_20220528T050701\_B08.jp2')

red = b4.read()

nir = b8.read()

ndvi = (nir.astype(float)-red.astype(float))/(nir+red)

meta = b4.meta

meta.update(driver='GTiff')

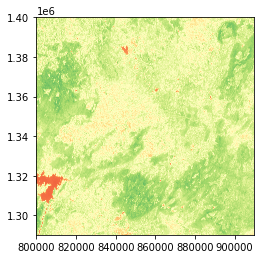
meta.update(dtype=rasterio.float32)

with rasterio.open('NDVI2.tif', 'w', \*\*meta) as dst:

dst.write(ndvi.astype(rasterio.float32))

ndvi = rio.open(r'NDVI2.tif')

show(ndvi, transform=ndvi.transform, cmap='RdYlGn')



**Step – 4 Mask Satellite image**

boundary\_proj = boundary.to\_crs({'init': 'epsg:32643'})

with rio.open("/content/NDVI2.tif") as src:

out\_image, out\_transform = rio.mask.mask(src, boundary\_proj.geometry,crop=True)

out\_meta = src.meta.copy()

out\_meta.update({"driver": "GTiff",

"height": out\_image.shape[1],

"width": out\_image.shape[2],

"transform": out\_transform})

with rio.open("NDVI\_masked.tif", "w", \*\*out\_meta) as dest:

dest.write(out\_image)

ndvi\_mask = rio.open(r'NDVI\_masked.tif')

show(ndvi\_mask, transform=ndvi\_mask.transform, cmap='RdYlGn')

