Article

Assessing Wildfire Severity Using Sentinel-2 Satellite Imagery: A Case Study of the 2023 Tenerife Wildfires

Regina Deri[1]

[1] Master's student of Satellite Data Science, School of Geography, Geology and the Environment Sciences, University of Leicester

Abstract: Accurate assessment of wildfire severity is essential for effective wildfire management, damage assessment, planning and monitoring of vegetation restoration. In this study, remote sensing data derived from Sentinel-2 imagery is used to assess the severity of a wildfire that occurred in Tenerife, Spain, in 2023. The Normalized Burn Ratio (NBR) and differenced NBR (dNBR) indices are calculated to delineate areas of vegetation damage and map fire severity levels across the study area. Additionally, the changes in the Normalized Difference Vegetation Index (NDVI) was analyzed to evaluate post-fire vegetation recovery dynamics. Our results confirm the effectiveness of NBR and dNBR indices in assessing fire severity, with significant decreases in NBR values observed in burned areas. The generated fire severity map provides insights into the spatial distribution of fire impacts, highlighting areas of high burn severity of interest to land managers. Furthermore, our correlation analysis between dNBR and changes in NDVI offers insights into post-fire vegetation health and recovery. While a weak positive correlation was observed, further research is needed to understand the underlying factors influencing post-fire vegetation response. Overall, this study highlights the importance of remote sensing data analysis for comprehensive wildfire severity assessment.

Keywords: fire severity; wildfire; NBR; dNBR; NDVI; Sentinel-2; Tenerife; remote sensing

1. Introduction

Wildfires are a naturally occurring element of the Earth's system resulting to both positive and negative effect on the ecosystem [1]. They contribute positively by aiding in vegetation growth, nutrient release in forested areas, and maintaining ecological balance within forests [2]. However, they also pose significant negative impacts, ranking among the most destructive natural disasters globally [1]. Wildfires result to the destruction of infrastructure, economic losses, global warming, and the possible loss of human and lives [1], [2–4]. In August 2023, a devastating wildfire broke out on the Spanish island of Tenerife fuelled by extreme weather conditions and the rugged terrain of the island [5]. The wildfire, which ignited on August 15th spread to 1,800 hectares in 24 hours, destroyed a forested area of the national park surrounding the famous Mount Teide volcano, and forced the evacuation of residents and tourists from their homes and resorts [6]. The severity of wildfires such as the outbreak in Tenerife requires accurate and rapid mapping

of fire damaged areas to support fire management, planning and monitoring of vegetation restoration, and implementation of effective mitigation and planning strategies [7].

Various methods including from field-based assessments to remote sensing approaches, have been developed to assess wildfire severity [8]. The assessment of wildfire severity typically relies on traditional field methods, which have notable limitations such as limited spatial coverage and high expenses [9]. Although field surveys prove effective for small-scale assessments of wildfire damage, they become impractical and costly when addressing larger areas suggesting the need for alternative methods [9]. In addition, wildfires occur in remote areas, posing a challenge for early detection and monitoring through on-site observations [1].

In recent decades, there has been rapid advancement in mapping and assessing the effects and severity of wildfires using remote sensing [10]. Remote sensing has proven effective in providing opportunities to enhance wildfire prediction, detection, mapping, monitoring, and post-fire damage assessment over large and inaccessible areas [11]. In remote sensing fire severity mapping studies, image differencing approaches involving multi-date change detection (e.g., between pre- and post-fire images) and several reflectance indices have been widely used, including the differenced Normalised Burn Ratio (dNBR), the normalized difference vegetation index (NDVI), enhanced vegetation index (EVI), and integrated forest index [12-16].

The most commonly used index is the Normalized Burn Ratio (NBR), which has proven to be effective in producing reasonable mapping of the spatial variation in fire severity with roughly 53–80% accuracy compared to field validation [9,13,14,17,18]. The normalized burn ratio (NBR) uses near-infrared (NIR) and shortwave-infrared (SWIR) for the detection of change in reflectance as a result of fire, due to the sensitivity of these wavelengths in detecting changes due to fire [14]. The difference in reflectance before and after a fire provides a quantitative estimate of the absolute change in plant biomass due to fire [17]. Hence, to calculate dNBR, the pre-fire NBR is subtracted from the post-fire NBR to identify areas of significant change [17]. The selection of the pre-fire and post-fire images may vary depending on the objective of the research. However, to determine which image is more accurate, it is important to consider the time elapsed between the fire occurrence and the images acquisition [4]. Although dNBR has proven to be an effective indicator of fire severity in forests, when applied to a new area or vegetation type, the index is not always accurate and requires validation through field observations [14]. Additionally, a combination of multiple indices may provide more accurate and comprehensive results compared to the use of one index alone [12]. Other indices, such as the Normalized Difference Vegetation Index (NDVI), provide insights into the health and recovery of vegetation following the wildfire event and based on the difference in reflectance between the red and near-infrared spectral bands, the amount of photosynthetically active vegetation is measured [19]. In general both dNBR and dNDVI are effective at identifying biomass loss relative to the pre-fire state of an ecosystem [20]. In addition, these indices have been applied with success to satellite data, such as Landsat-7 and Landsat-8 [21-23] and Sentinel-2 [13,4] images to map and assess fire severity.

Satellite-based data, particularly Sentinel-2 MultiSpectral Instrument (MSI) data provided by the European Space Agency (ESA), has become a critical resource for mapping burned areas and assessing severity, due to its high spatial resolution ranging from 10 to 60 m and frequent revisit times [21]. In addition, the MSI sensor provides spectral information covering 13 spectral bands, including the near infrared (NIR) and the shortwave infrared (SWIR) which indicates the highest difference between burned and unburned areas [4,24].

Therefore, the aim of this report is to demonstrate the use of Sentinel-2 satellite imagery and remote sensing indices to assess the severity of the 2023 wildfire in Tenerife.

2. Materials and Methods

2.1 Study Area

The chosen wildfire occurred in Tenerife, the largest and most populous island of the Canary Islands archipelago, with an area of approximately 2,036 km² (Figure 1). The fire ignited on August 15th, 2023, and consumed approximately 1,800 hectares of the surrounding area within 24 hours, devastating a forested area of the national park surrounding the iconic Mount Teide volcano [6]. The island occasionally experiences drought conditions, particularly during the summer months, which heighten the risk of wildfires [25]. Tenerife's vegetation varies based on altitude and wind exposure, encompassing coastal shrublands, pine forests, and high-altitude alpine vegetation, all of which were impacted by the fire.

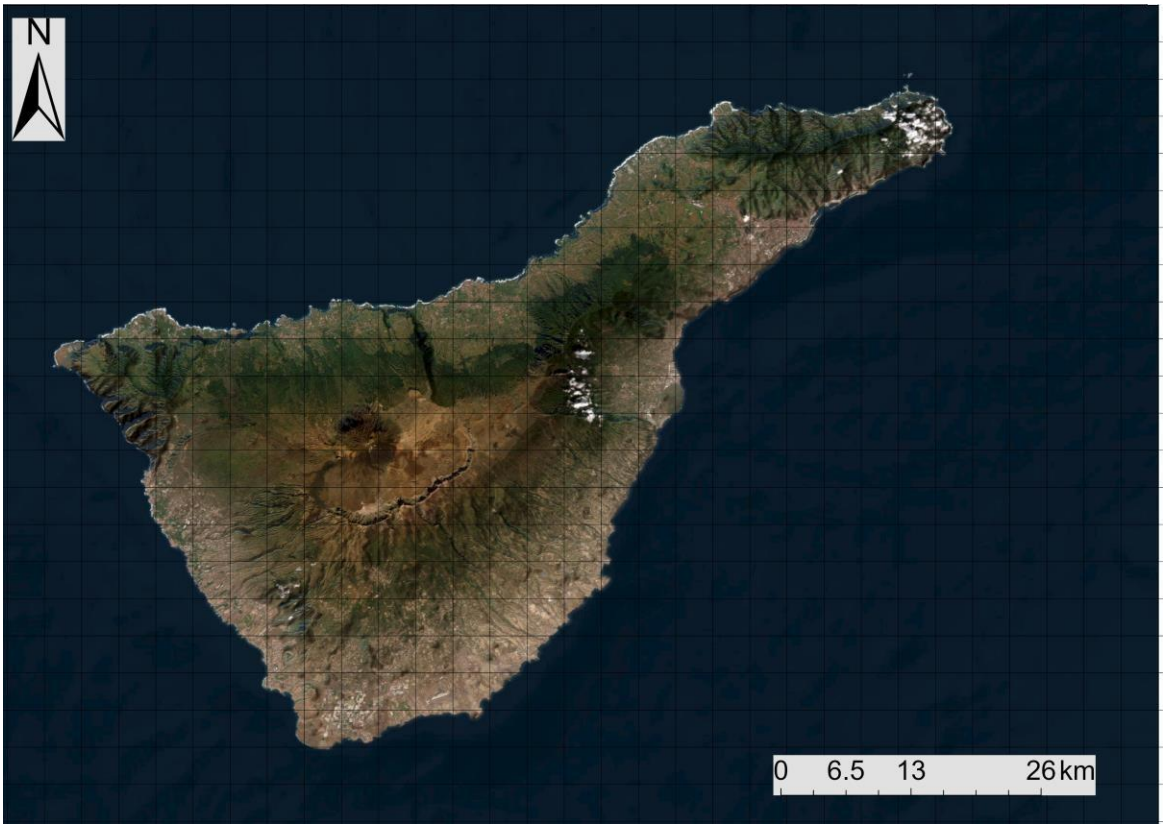


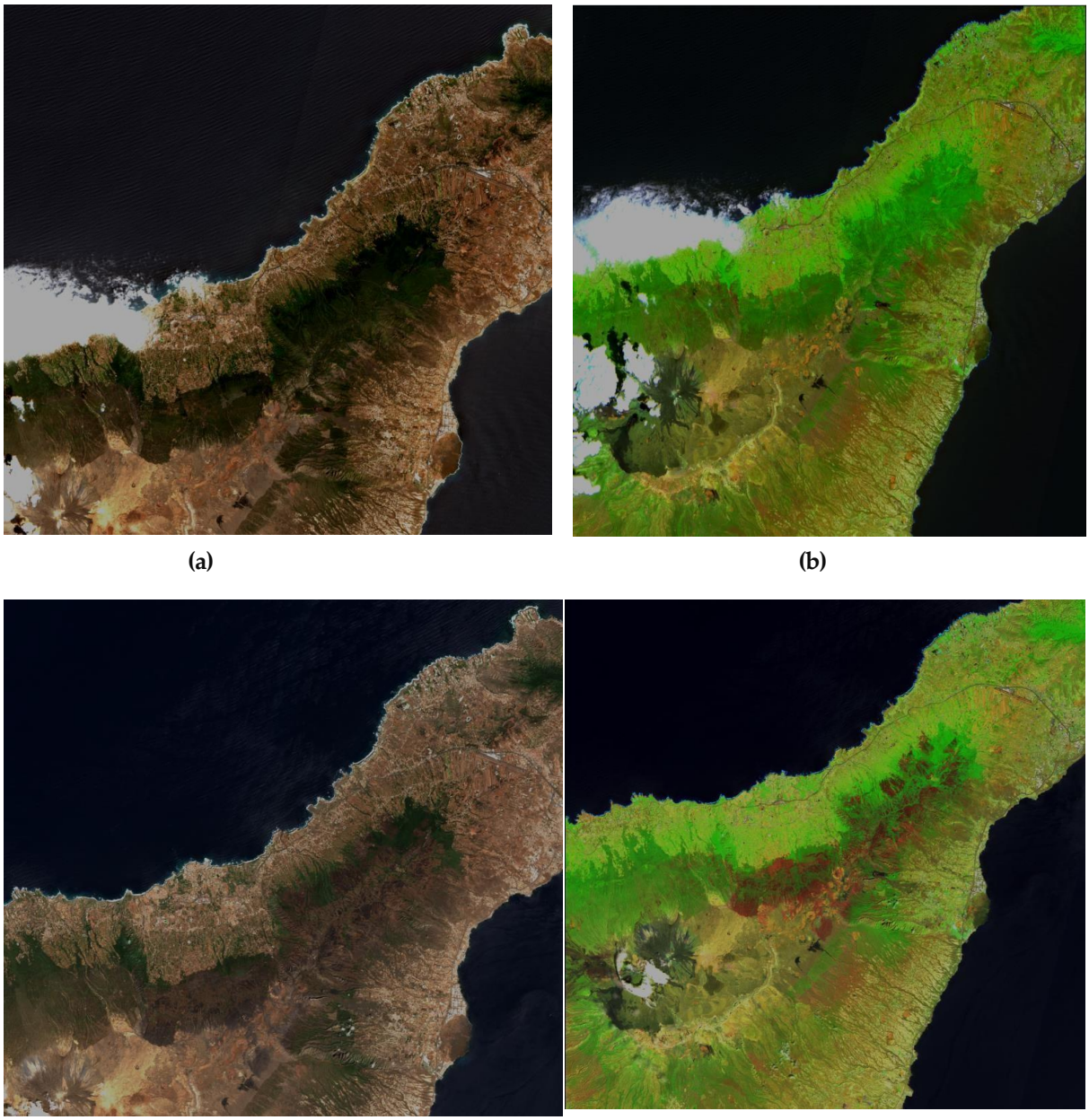
Figure 1. Location of the study area

2.2. Satellite Imagery and Pre-Processing

This study utilized data from the Sentinel-2A satellite. The Sentinel-2 imagery was downloaded from the Copernicus Data Space at level-2A (Atmospherically, radiometrically and geometrically corrected) with low cloud cover (< 10%) to minimize any potential obstruction of land surfaces, enabling accurate assessment of pre- and post-fire conditions [4]. The downloaded images are as close as possible to the fire occurrence to prevent a negative impact on their dNBR [4]. The pre-fire image was acquired on the 13th of August 2023, and the post-fire image was acquired on the7th of September 2023. The Sentinel-2 bands used in this study had a spatial resolution of 20 m including bands 2, 3, 4, and 8A, and 12. Table 1 shows a summary of the Sentinel-2 bands with their respective central wavelengths and spatial resolutions. Prior to analysis, the Sentinel-2A images underwent pre-processing procedures, including the correction of images with processing baseline 04.00 or later to remove the applied radiometric offset. This step was essential to fix the data inconsistency in the Sentinel-2 time series introduced in March 2022 [26]. A true and false color composite of the pre-fire and post-fire image was generated to aid in identifying the different features in the imagery based on the band combination as seen in figure 2.

Table 1. Sentinel-2 bands with their respective central wavelengths and spatial resolutions.

Spatial Resolution (m)	**Band Name**	**Spatial resolution (m)**	**Central wavelength (nm)**
Coastal aerosol	B1	60	443
Blue	B2	10	490
Green	B3	10	560
Red	B4	10	665
Red edge 1	B5	20	705
Red edge 2	B6	20	740
NIR (narrow 1)	B7	20	783
NIR	B8	10	842
NIR (narrow 2)	B8a	20	865
Water Vapour	B9	60	940
SWIR Cirrus	B10	60	1375
SWIR 1	B11	20	1610
SWIR 2	B12	20	2190



(a) **(b)**

(c) **(d)**

Figure 2. Satellite image of the pre- and post-fire scenario: (**a**) pre-fire true color representation (composite bands 4-3-2) ; (**b**) pre-fire false color representation (composite bands 12-8A-4); (**c**) post-fire true color representation (composite bands 4-3-2); (**d**) post-fire false color representation (composite bands 12-8A-4)

2.3 Spectral Indices

The assessment of wildfire severity involved the calculation of key indices, including preNBR, postNBR, dNBR, preNDVI, postNDVI, and dNDVI based on the derived pre- and post-fire imagery. The selection of these indices is based on their established effectiveness in previous studies for assessing wildfire severity. These analysis were

performed using python code on the goggle colab platform. The indices were calculated given by the following formulas:

$$preNBR = \frac{(preNIR - preSWIR)}{(preNIR + preSWIR)} \quad (1)$$

$$postNBR = \frac{(postNIR - postSWIR)}{(postNIR + postSWIR)} \quad (2)$$

$$\mathbf{dNBR} = preNBR - postNBR \quad (3)$$

$$preNDVI = \frac{(preNIR - preRed)}{(preNIR + preRed)} \quad (4)$$

$$postNDVI = \frac{(postNIR - postRed)}{(postNIR + postRed)} \quad (5)$$

$$\mathbf{dNDVI} = postNDVI - preNDVI \quad (6)$$

Burn severity map from dNBR image was obtained by applying the threshold levels according to the classification proposed by the United States Geological Survey (USGS) and was multiplied by a scale factor of 1000 to categorize the fire severity levels as shown in Table 2 [4, 27]. Furthermore, the derived NBR and dNBR results were imported into ArcGIS Pro to digitize burned and unburned polygons within the study area. Subsequently, the average NBR values for both burned and unburned polygons before and after the fire are computed and dNBR is determined for each polygon by subtracting the pre-fire NBR from the post-fire NBR values. Lastly, correlation analysis using pearson correlation was applied to explore the relationship between dNBR values, indicative of fire severity, and changes in NDVI, reflecting vegetation loss.

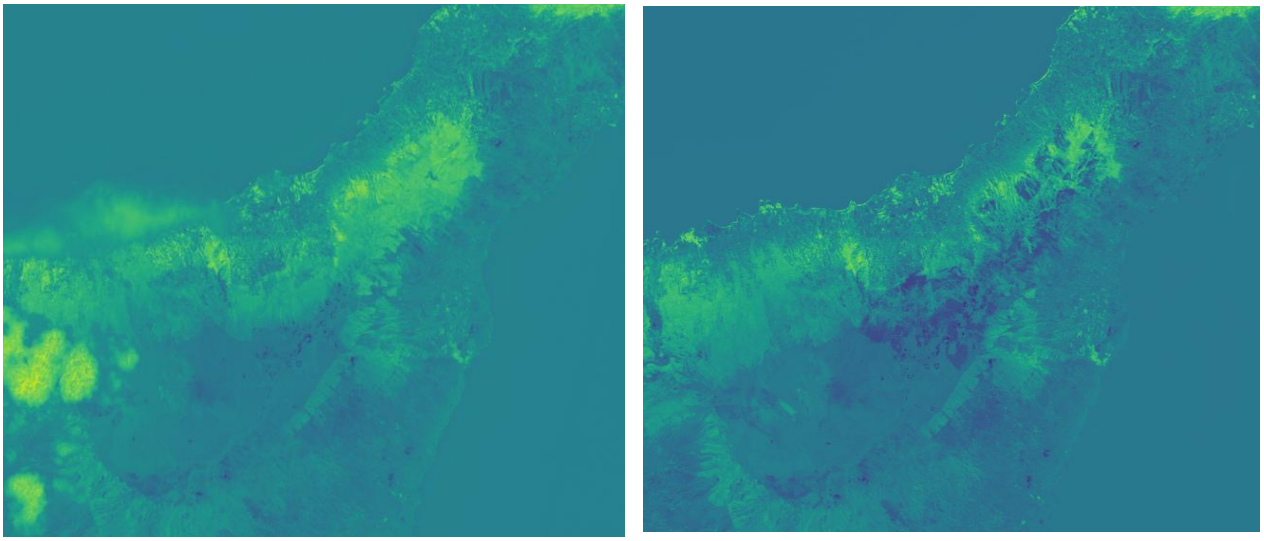
Table 2. Differenced Normalized Burn Ratio (dNBR) thresholds proposed by the United States Geological Survey (USGS)

Severity Level	dNBR Range (scaled by 10^3)	dNBR Range (not scaled)
Enhanced Regrowth, high (post-fire)	-500 to -251	-0.500 to -0.251
Enhanced Regrowth, low (post-fire)	-250 to -101	-0.250 to -0.101
Unburned	-100 to +99	-0.100 to +0.99
Low Severity	+100 to +269	+0.100 to +0.269
Moderate-low Severity	+270 to +439	+0.270 to +0.439
Miderate-high Severity	+440 to +659	+0.440 to +0.659
High Severity	+660 to +1300	+0.660 to +1.300

3. Results

3.1 Assessment of Wildfire Severity using NBR

The severity of the wildfire in Tenerife was analyzed by computing the NBR from pre-fire and post-fire Sentinel-2A images. A first assessment of each resulting NBR image can be made by means of visual image comparison between preNBR and postNBR as shown in Figure 3. Comparison of pre-fire and post-fire NBR images reveals the change in vegetation cover and condition within the impacted areas.



(a) **(b)**

Figure 3. Comparison between (**a**) preNBR and (**b**) postNBR

To further assess the impact of the wildfire, five polygons representing burned areas and five polygons representing unburned areas were digitized and the average pre-fire and post-fire NBR as well as the difference in NBR (dNBR) within each polygon was calculated. For the polygons representing burned areas, the average NBR before the fire exhibited relatively high values, indicating healthy vegetation cover prior to the wildfire. However, the average NBR after the fire showed a decrease, reflecting fire-induced damage to vegetation within these areas. The difference in NBR (dNBR) values for the burned area polygons showed negative values, indicating the severity of the fire. In contrast, the polygons representing pre-fire and post-fire NBR for unburned areas, showed higher values suggesting minimal changes in the health of the vegetation before and after the fire. As a result, the calculated dNBR values for unburned areas were close to zero, indicating minimal fire impact in these polygons (Table 3).

Table 3. Average NBR and dNBR Values for Burned and Unburned Polygons

Polygon	Type	Prefire NBR	Postfire NBR	dNBR
1	Burned	0.092904	-0.050336	-0.143241
2	Burned	0.094734	-0.010416	-0.105150
3	Burned	0.102745	0.019941	-0.082804

4	Burned	0.122793	-0.045659	-0.168452
5	Burned	0.128601	0.010591	-0.118009
6	Unburned	0.007058	0.009380	0.002322
7	Unburned	0.012222	0.010222	-0.002000
8	Unburned	0.055849	0.061954	0.006106
9	Unburned	0.110324	0.112563	0.002239
10	Unburned	0.060001	0.056428	-0.003573

3.2 Fire Severity Mapping

The dNBR calculated by subtracting pre-fire NBR values from post-fire values to identify areas of notable change attributed to the wildfire. The derived dNBR was used to create the fire severity map. According to the proposed fire severity ranges by USGS, dNBR values < 0.100 represented unburnt, values ranging from < 0.270 represented low fire severity, values ranging from < 0.440 represented moderate low severity, values ranging from < 0.660 represented moderate high severity and values < 1.300 represented high fire severity. Figure 4 shows the fire severity levels results for each index where higher dNBR values suggested heightened fire severity, whereas negative dNBR values suggested unburned areas or post-fire recovery following the fire.

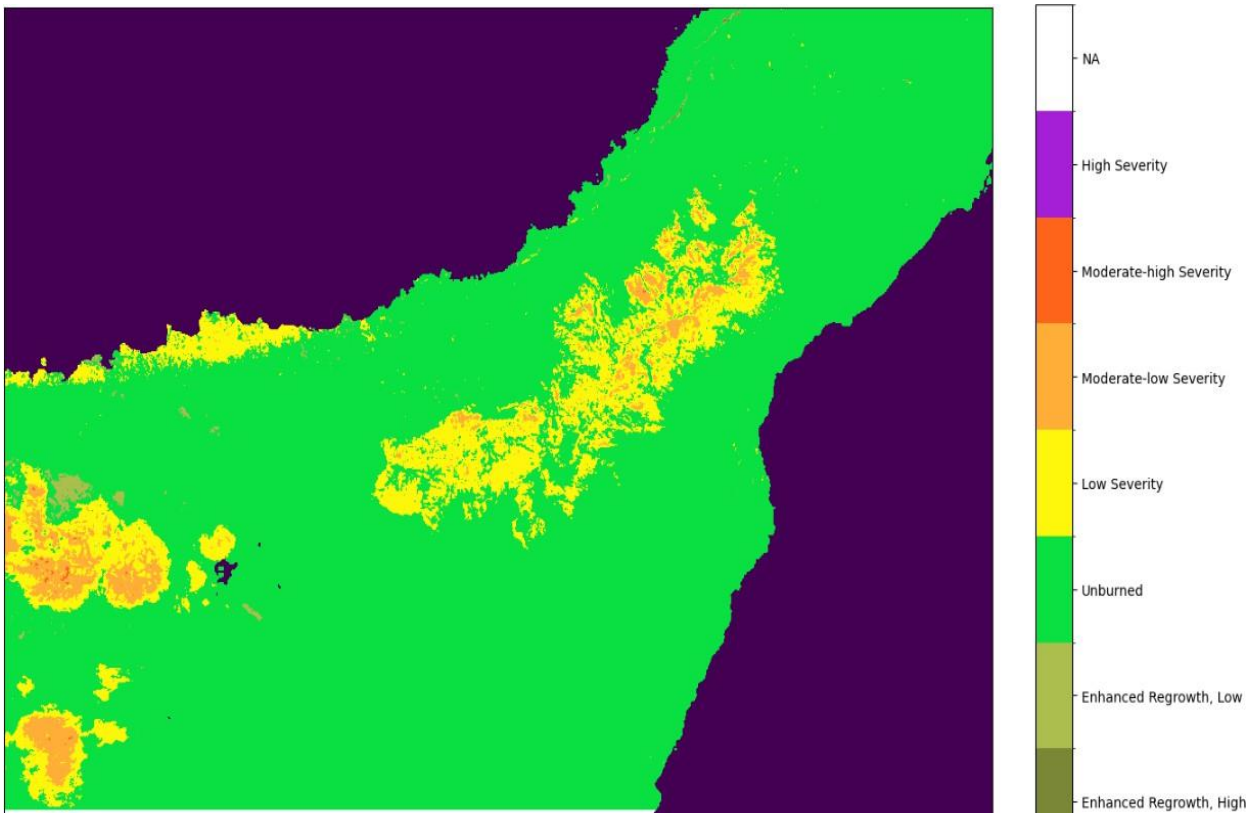
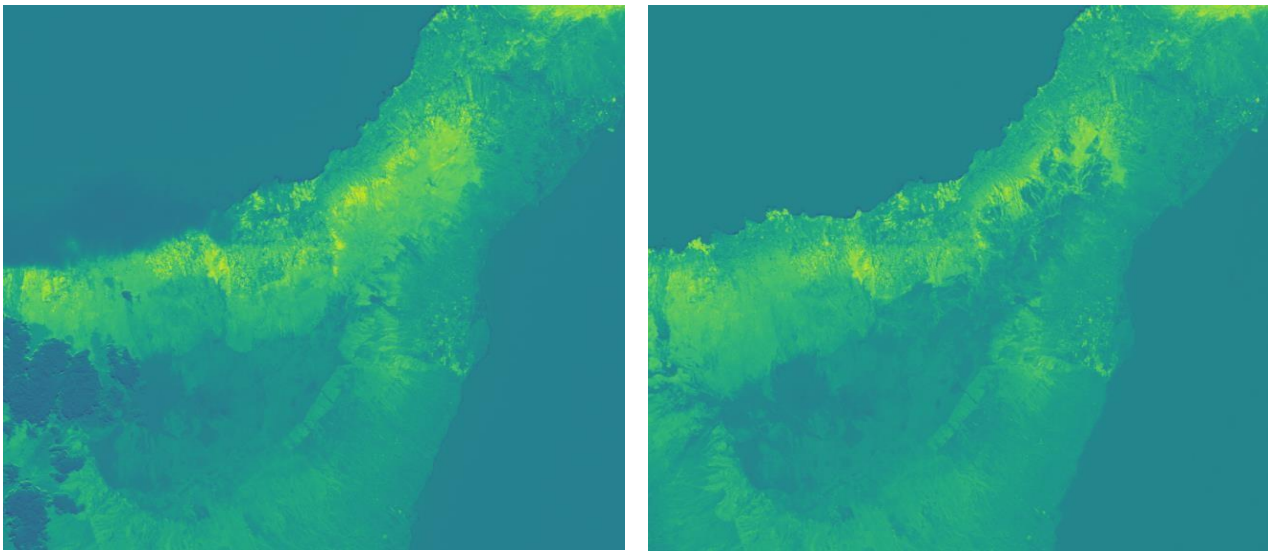


Figure 4. The dNBR fire severity map

3.3 Analysis of NDVI

In addition to NBR analysis, the NDVI from pre-fire and post-fire images was calculated. An initial assessment of the comparison between pre-fire and post-fire NDVI images revealed changes in vegetation health and recovery patterns. Areas with lower post-fire NDVI values indicated vegetation stress or damage, while regions with higher NDVI values suggested vegetation recovery or minimal fire impact (Figure 5).



(a) **(b)**

Figure 5. Comparison between (**a**) preNDVI and (**b**) postNDVI

3.4 Statistical Analysis

Statistical analysis was conducted to explore the relationship between dNBR values and changes in NDVI following the wildfire. Correlation analysis using 50 random points particular within high severity areas to assess the degree of correlation between fire severity, as indicated by dNBR, and vegetation recovery, as measured by changes in NDVI. The results of the Pearson's correlation coefficient R and P-values are shown in scatter plot chart (Figure 7). According to the result, preliminary analysis showed a weak positive correlation between the two indices was observed with an R-value of 0.289 and P-value of 0.8424.

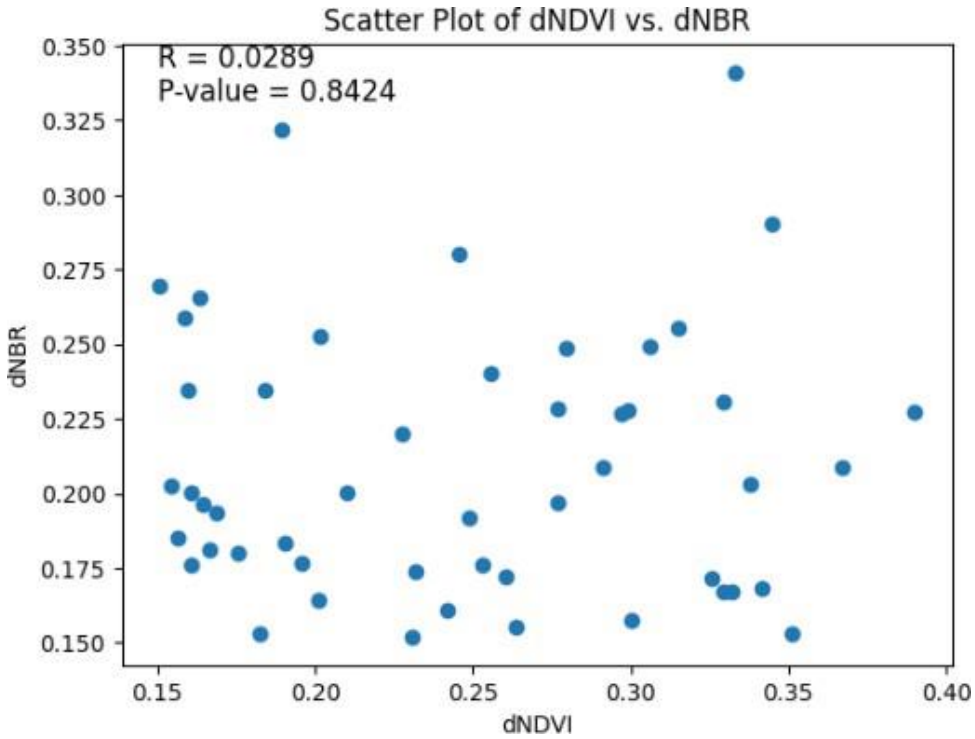


Figure 6. Scatterplot representing the correlation between the dNBR and dNDVI

4. Discussion

The results of this study confirm that remote sensing indices derived from Sentinel-2 imagery provide an overall acceptable assessment of fire severity. In particular, previous studies [4,10,12,17] have demonstrated significant success in utilizing the Normalized Burn Ratio (NBR). The utilization of NBR and dNBR indices for fire severity mapping in this study is consistent with the established success of these indices reported in the literature. As expected, the observed decrease in NBR values in burned areas, resulting in negative dNBR values, highlights the vegetation damage caused by the wildfire. This finding aligns with previous research indicating that lower NBR values correlate with greater fire severity and vegetation loss [17]. In this regard, dNBR plays a crucial role in monitoring fire severity. However, calibration using field data is essential for assessing the effectiveness and accuracy of remote sensing results [28]. This calibration is particularly crucial in ecosystems where no prior field calibration has been conducted [15].

The result of the dNDVI is consistent with existing literature highlighting the influence of wildfires on vegetation health and recovery [20]. This highlights the importance of NDVI for monitoring vegetation losses. Preliminary analysis of the correlation between dNBR values and changes in NDVI revealed a relatively weak positive correlation between the two indices, contrasting with previous research that reported a high positive correlation [20]. However, this discrepancy could be influenced by factors such as vegetation types in the study area. To improve our knowledge of how ecosystems respond to

fire and our capacity to forecast fire severity, more research could be done on the relationship between fire severity, environmental conditions, and vegetation recovery.

5. Conclusions

This study aimed at estimating the severity of the fire which occurred in August 2023 in Tenerife using remote sensing indices derived from sentinel-2 images. NBR and dNBR indices provided valuable insights into the spatial distribution and severity of the wildfire. The result showed a decrease in NBR values in burned areas, indicating vegetation damage caused by the wildfire in the study area. Additionally, the fire severity map generated from the dNBR values provided a comprehensive visualization of the severity levels across the study area. Areas with high burn severity are of particular interest to land managers as they indicate regions with the most significant changes to vegetation ecosystem properties due to fire. These findings can aid prioritization of post-fire management and restoration efforts, reducing the ecological effects of wildfires and promoting ecosystem resilience. Moreover, while a relatively weak positive correlation was observed between dNBR and dNDVI, further research is warranted to explore the relationship between wildfire, and post-fire vegetation recovery dynamics. In conclusion, this study demonstrates the effectiveness of remote sensing indices derived from Sentinel-2 imagery for assessing fire severity in the study area.

Funding: This research received no external funding

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: https://scihub.copernicus.eu/.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Rashkovetsky, D., Mauracher, F., Langer, M. and Schmitt, M. (2021). Wildfire Detection From Multisensor Satellite Imagery Using Deep Semantic Segmentation. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 14, pp.7001–7016. doi:https://doi.org/10.1109/jstars.2021.3093625.

2. Amos, C., Petropoulos, G.P. and Ferentinos, K.P. (2018). Determining the use of Sentinel-2A MSI for wildfire burning & severity detection. *International Journal of Remote Sensing*, 40(3), pp.905–930. doi:https://doi.org/10.1080/01431161.2018.1519284.

3. Phan, T.C. and Nguyen, T.T. (2019). Remote Sensing meets Deep Learning: Exploiting Spatio-Temporal-Spectral Satellite Images for Early Wildfire Detection. *Environ. Sci. (2019),* pp. 1-9. doi:https://infoscience.epfl.ch/record/270339.

4. Llorens, R., Sobrino, J.A., Fernández, C., Fernández-Alonso, J.M. and Vega, J.A. (2021). A methodology to estimate forest fires burned areas and burn severity degrees using Sentinel-2 data. Application to the October 2017 fires in the Iberian Peninsula. *International Journal of Applied Earth Observation and Geoinformation*, 95, p.102243. doi:https://doi.org/10.1016/j.jag.2020.102243.

5. Pons, C. and Pons, C. (2023). Wildfire on Spain's Tenerife spreads across island's north. *Reuters*. [online] 17 Aug. Available at: https://www.reuters.com/world/europe/wildfire-tenerife-national-park-prompts-village-evacuations-2023-08-16/.

6. France-Presse, A. (2023). Thousands evacuated on Tenerife as wildfire rages amid heatwave. *The Guardian*. [online] 4 Oct. Available at: https://www.theguardian.com/weather/2023/oct/05/tenerife-fires-wildfire-spain-canary-islands-heatwave-evacuation.

7. Alcaras, E., Costantino, D., Guastaferro, F., Parente, C. and Pepe, M. (2022). Normalized Burn Ratio Plus (NBR+): A New Index for Sentinel-2 Imagery. *Remote Sensing*, 14(7), p.1727. doi:https://doi.org/10.3390/rs14071727.

8. Barmpoutis, P., Papaioannou, P., Dimitropoulos, K. and Grammalidis, N. (2020). A Review on Early Forest Fire Detection Systems Using Optical Remote Sensing. *Sensors (Basel, Switzerland)*, [online] 20(22). doi:https://doi.org/10.3390/s20226442.

9. De Santis, A. and Chuvieco, E. (2007). Burn severity estimation from remotely sensed data: Performance of simulation versus empirical models. *Remote Sensing of Environment*, 108(4), pp.422–435. doi:https://doi.org/10.1016/j.rse.2006.11.022.

10. Howe, A.A., Parks, S.A., Harvey, B.J., Saberi, S.J., Lutz, J.A. and Yocom, L.L. (2022). Comparing Sentinel-2 and Landsat 8 for Burn Severity Mapping in Western North America. *Remote Sensing*, [online] 14(20), p.5249. doi:https://doi.org/10.3390/rs14205249.

11. Allison, R., Johnston, J., Craig, G. and Jennings, S. (2016). Airborne Optical and Thermal Remote Sensing for Wildfire Detection and Monitoring. *Sensors*, 16(8), p.1310. doi:https://doi.org/10.3390/s16081310.

12. Miller, J.D. and Thode, A.E. (2007). Quantifying burn severity in a heterogeneous landscape with a relative version of the delta Normalized Burn Ratio (dNBR). *Remote Sensing of Environment*, 109(1), pp.66–80. doi:https://doi.org/10.1016/j.rse.2006.12.006.

13. Gibson, R., Danaher, T., Hehir, W. and Collins, L. (2020). A remote sensing approach to mapping fire severity in south-eastern Australia using sentinel 2 and random forest. *Remote Sensing of Environment*, [online] 240, p.111702. doi:https://doi.org/10.1016/j.rse.2020.111702.

14. Giddey, B.L., Baard, J.A. and Kraaij, T. (2022). Verification of the differenced Normalised Burn Ratio (dNBR) as an index of fire severity in Afrotemperate Forest. *South African Journal of Botany*, 146, pp.348–353. doi:https://doi.org/10.1016/j.sajb.2021.11.005.

15. Delcourt, C.J.F., Combee, A., Izbicki, B., Mack, M.C., Maximov, T., Petrov, R., Rogers, B.M., Scholten, R.C., Shestakova, T.A., van Wees, D. and Veraverbeke, S. (2021). Evaluating the Differenced Normalized Burn Ratio for Assessing Fire Severity Using Sentinel-2 Imagery in Northeast Siberian Larch Forests. *Remote Sensing*, 13(12), p.2311. doi:https://doi.org/10.3390/rs13122311.

16. Chen, X., Vogelmann, J.E., Rollins, M., Ohlen, D., Key, C.H., Yang, L., Huang, C. and Shi, H. (2011). Detecting post-fire burn severity and vegetation recovery using multitemporal remote sensing spectral indices and field-collected

composite burn index data in a ponderosa pine forest. *International Journal of Remote Sensing*, 32(23), pp.7905–7927. doi:https://doi.org/10.1080/01431161.2010.524678.

17. Key, C.H. and Benson, N.C. (2006). Landscape assessment: ground measure of severity, the composite burn index; and remote sensing of severity, the normalized burn ratio. *FIREMON: Fire effects monitoring and inventory system USDA Forest Service, Rocky Mountain Res. Station 164. K*, 164.

18. Roy, D.P., Boschetti, L. and Trigg, S.N. (2006). Remote Sensing of Fire Severity: Assessing the Performance of the Normalized Burn Ratio. *IEEE Geoscience and Remote Sensing Letters*, 3(1), pp.112–116. doi:https://doi.org/10.1109/lgrs.2005.858485.

19. Veraverbeke, S., Lhermitte, S., Verstraeten, W.W. and Goossens, R. (2011). Evaluation of pre/post-fire differenced spectral indices for assessing burn severity in a Mediterranean environment with Landsat Thematic Mapper. *International Journal of Remote Sensing*, 32(12), pp.3521–3537. doi:https://doi.org/10.1080/01431161003752430.

20. Yilmaz, O.S., Acar, U., Sanli, F.B., Gulgen, F. and Ates, A.M. (2023). Mapping burn severity and monitoring CO content in Türkiye's 2021 Wildfires, using Sentinel-2 and Sentinel-5P satellite data on the GEE platform. *Earth Science Informatics*. doi:https://doi.org/10.1007/s12145-023-00933-9.

21. Quintano, C., Fernández-Manso, A. and Fernández-Manso, O. (2018). Combination of Landsat and Sentinel-2 MSI data for initial assessing of burn severity. *International Journal of Applied Earth Observation and Geoinformation*, [online] 64, pp.221–225. doi:https://doi.org/10.1016/j.jag.2017.09.014.

22. Epting, J., Verbyla, D. and Sorbel, B. (2005). Evaluation of remotely sensed indices for assessing burn severity in interior Alaska using Landsat TM and ETM+. *Remote Sensing of Environment*, 96(3-4), pp.328–339. doi:https://doi.org/10.1016/j.rse.2005.03.002.

23. Hoy, E.E., French, N.H.F., Turetsky, M.R., Trigg, S.N. and Kasischke, E.S. (2008). Evaluating the potential of Landsat TM/ETM+ imagery for assessing fire severity in Alaskan black spruce forests. *International Journal of Wildland Fire*, 17(4), p.500. doi:https://doi.org/10.1071/wf08107.

24. Bone, R.C., Balk, R.A., Cerra, F.B., Dellinger, R.P., Fein, A.M., Knaus, W.A., Schein, R.M.H. and Sibbald, W.J. (1992). Definitions for Sepsis and Organ Failure and Guidelines for the Use of Innovative Therapies in Sepsis. *Chest*, [online] 101(6), pp.1644–1655. doi:https://doi.org/10.1378/chest.101.6.1644.

25. Carrillo, J., Pérez, J.C., Expósito, F.J., Díaz, J.P. and González, A. (2022). Projections of wildfire weather danger in the Canary Islands. *Scientific Reports*, [online] 12(1), p.8093. doi:https://doi.org/10.1038/s41598-022-12132-5.

26. Sentinel Online. (n.d.). *Processing Baseline*. [online] Available at: https://sentinels.copernicus.eu/web/sentinel/technical-guides/sentinel-2-msi/processing-baseline [Accessed 26 Apr. 2024].

27. UN-SPIDER Knowledge Portal. (n.d.). *Normalized Burn Ratio (NBR)*. [online] Available at: https://un-spider.org/advisory-support/recommended-practices/recommended-practice-burn-severity/in-detail/normalized-burn-ratio.

28. Cardil, A., Mola-Yudego, B., Blázquez-Casado, Á. and González-Olabarria, J.R. (2019). Fire and burn severity assessment: Calibration of Relative Differenced Normalized Burn Ratio (RdNBR) with field data. *Journal of Environmental Management*, 235, pp.342–349. doi:https://doi.org/10.1016/j.jenvman.2019.01.077.

Appendix

Burn Severity Map of Tenerife wildfire, August 15 2023

Connect to Google Drive from Colab.

```
# Load the Drive helper and mount your Google Drive as a drive in the virtual machine
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

Import libraries

```
#import required libraries
!pip install rasterio
!pip install geopandas
!pip install rasterstats


import datetime
import geopandas as gpd
import glob
import json
import matplotlib.pyplot as plt
import matplotlib.colors
from matplotlib.colors import BoundaryNorm, ListedColormap
from matplotlib.cm import get_cmap
from matplotlib.axes import Axes
import math
from math import floor, ceil
import numpy as np
from osgeo import ogr
import os
from os import listdir
from os.path import isfile, isdir, join
import pandas as pd
from pprint import pprint
from pyproj import Proj
import rasterio
from rasterio import plot
from rasterio.plot import show
from rasterio.plot import show_hist
from rasterio.windows import Window
from rasterio.mask import mask
from skimage import exposure
import scipy.stats as stats
from scipy.stats import pearsonr
import requests
import shutil
import sys
from tempfile import TemporaryDirectory
import zipfile
```

```
# define the root directory where the data are stored
rootdir = '/content/drive/MyDrive/GY7709/week_7'
if rootdir not in sys.path:
  sys.path.append(rootdir)
# import the pygge library of functions for the analysis
import pygge

%matplotlib inline
```

```
Requirement already satisfied: rasterio in /usr/local/lib/python3.10/dist-packages (1.3.10)
Requirement already satisfied: affine in /usr/local/lib/python3.10/dist-packages (from rasterio) (2.4.0)
Requirement already satisfied: attrs in /usr/local/lib/python3.10/dist-packages (from rasterio) (23.2.0)
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from rasterio) (2024.2.2)
Requirement already satisfied: click>=4.0 in /usr/local/lib/python3.10/dist-packages (from rasterio) (8.1.7)
Requirement already satisfied: cligj>=0.5 in /usr/local/lib/python3.10/dist-packages (from rasterio) (0.7.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from rasterio) (1.25.2)
Requirement already satisfied: snuggs>=1.4.1 in /usr/local/lib/python3.10/dist-packages (from rasterio) (1.4.7)
Requirement already satisfied: click-plugins in /usr/local/lib/python3.10/dist-packages (from rasterio) (1.1.1)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from rasterio) (67.7.2)
Requirement already satisfied: pyparsing>=2.1.6 in /usr/local/lib/python3.10/dist-packages (from snuggs>=1.4.1->rasterio) (3.1.2)
Requirement already satisfied: geopandas in /usr/local/lib/python3.10/dist-packages (0.13.2)
Requirement already satisfied: fiona>=1.8.19 in /usr/local/lib/python3.10/dist-packages (from geopandas) (1.9.6)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from geopandas) (24.0)
Requirement already satisfied: pandas>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from geopandas) (2.0.3)
Requirement already satisfied: pyproj>=3.0.1 in /usr/local/lib/python3.10/dist-packages (from geopandas) (3.6.1)
Requirement already satisfied: shapely>=1.7.1 in /usr/local/lib/python3.10/dist-packages (from geopandas) (2.0.3)
Requirement already satisfied: attrs>=19.2.0 in /usr/local/lib/python3.10/dist-packages (from fiona>=1.8.19->geopandas) (23.2.0)
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from fiona>=1.8.19->geopandas) (2024.2.2)
Requirement already satisfied: click~=8.0 in /usr/local/lib/python3.10/dist-packages (from fiona>=1.8.19->geopandas) (8.1.7)
Requirement already satisfied: click-plugins>=1.0 in /usr/local/lib/python3.10/dist-packages (from fiona>=1.8.19->geopandas) (1.1.1)
Requirement already satisfied: cligj>=0.5 in /usr/local/lib/python3.10/dist-packages (from fiona>=1.8.19->geopandas) (0.7.2)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from fiona>=1.8.19->geopandas) (1.16.0)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.1.0->geopandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.1.0->geopandas) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.1.0->geopandas) (2024.1)
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.1.0->geopandas) (1.25.2)
Requirement already satisfied: rasterstats in /usr/local/lib/python3.10/dist-packages (0.19.0)
Requirement already satisfied: affine in /usr/local/lib/python3.10/dist-packages (from rasterstats) (2.4.0)
Requirement already satisfied: click>7.1 in /usr/local/lib/python3.10/dist-packages (from rasterstats) (8.1.7)
Requirement already satisfied: cligj>=0.4 in /usr/local/lib/python3.10/dist-packages (from rasterstats) (0.7.2)
Requirement already satisfied: fiona in /usr/local/lib/python3.10/dist-packages (from rasterstats) (1.9.6)
Requirement already satisfied: numpy>=1.9 in /usr/local/lib/python3.10/dist-packages (from rasterstats) (1.25.2)
Requirement already satisfied: rasterio>=1.0 in /usr/local/lib/python3.10/dist-packages (from rasterstats) (1.3.10)
Requirement already satisfied: simplejson in /usr/local/lib/python3.10/dist-packages (from rasterstats) (3.19.2)
Requirement already satisfied: shapely in /usr/local/lib/python3.10/dist-packages (from rasterstats) (2.0.3)
Requirement already satisfied: attrs in /usr/local/lib/python3.10/dist-packages (from rasterio>=1.0->rasterstats) (23.2.0)
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from rasterio>=1.0->rasterstats) (2024.2.2)
Requirement already satisfied: snuggs>=1.4.1 in /usr/local/lib/python3.10/dist-packages (from rasterio>=1.0->rasterstats) (1.4.7)
Requirement already satisfied: click-plugins in /usr/local/lib/python3.10/dist-packages (from rasterio>=1.0->rasterstats) (1.1.1)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from rasterio>=1.0->rasterstats) (67.7.2)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from fiona->rasterstats) (1.16.0)
Requirement already satisfied: pyparsing>=2.1.6 in /usr/local/lib/python3.10/dist-packages (from snuggs>=1.4.1->rasterio>=1.0->rasterstats) (3.1.2)
```

Set the directory paths on Google Drive.

```
print("Google Drive data directory: " + rootdir)

cd = "/content/work"
print("Temporary work directory: ", cd)

# Allow the notebook to connect to the account on the ESA Data Archive.
credentials = join(rootdir, 'sencredentials.txt')  # username and password to the Copernicus account

# directory for downloading the Sentinel-2 granules
downloaddir = join(cd, 'download') # To save the downloaded images
quickdir = join(cd, 'quicklooks')  # To save the quicklooks
outdir = join(cd, 'out')           # To save any other outputs

# create the new directories, unless they already exist
os.makedirs(cd, exist_ok=True)
os.makedirs(downloaddir, exist_ok=True)
os.makedirs(quickdir, exist_ok=True)
os.makedirs(outdir, exist_ok=True)

# check whether the file with the login details exists
if "sencredentials.txt" not in os.listdir(rootdir):
  print("\nERROR: File sencredentials.txt not found in directory: "+rootdir+
        "\nCannot log in to Copernicus Data Space Ecosystem.\n")
else:
  print("\nFound the file sencredentials.txt.")
```

```
Google Drive data directory: /content/drive/MyDrive/GY7709/week_7
Temporary work directory:  /content/work

Found the file sencredentials.txt.
```

```
'''
---------------------------------------------------------------
The following block of codes is modified from:

Balzter, H. (2024): Week 32. The Copernicus Data Space Ecosystem
https://blackboard.le.ac.uk/ultra/courses/_65627_1/outline/edit/document/_5159164_1
Downloaded 27 February 2024
---------------------------------------------------------------
'''
```

Download the Sentinel-2 Imagery

Firstly, the images from before and after the fire that will be used to calculate the burn severity is identified and downloaded from the Copernicus Data Space. The dates are as follows:

- 13 August 2023: before the ignition of the fire
- 7 September 2023: after the extinguishing of the fire

Download Postfire Image

```
[ ] # define the shapefile of the study area
    shapefile = join(rootdir, 'tenerife', 'POLYGON.shp')

    # check whether the shapefile exists
    if os.path.exists(shapefile):
      print('Shapefile found: '+shapefile)
    else:
      print('ERROR: Shapefile not found: '+shapefile)
      print('Upload a shapefile to your Google Drive directory: '+ rootdir)

    # Define a date range for the imagery search
    datefrom = '20230901' # start date for imagery search
    dateto   = '20230910' # end date for imagery search

    # Define the accepted cloud cover percentage in the imagery
    clouds = 10 # maximum cloud cover [%] for imagery search

    # Copernicus Data Space Ecosystem constants
    DATASPACE_API_ROOT = "http://catalogue.dataspace.copernicus.eu/resto/api/collections/Sentinel2/search.json"
    DATASPACE_DOWNLOAD_URL = "https://catalogue.dataspace.copernicus.eu/odata/v1/Products"
    DATASPACE_REFRESH_TOKEN_URL = "https://identity.dataspace.copernicus.eu/auth/realms/CDSE/protocol/openid-connect/token"
```

```
Shapefile found: /content/drive/MyDrive/GY7709/week_7/tenerife/POLYGON.shp
```

355

Get some information about the shapefile.

```
] # Get the shapefile layer's extent, CRS and EPSG code
  extent, outSpatialRef, epsg = pygge.get_shp_extent(shapefile)
  print("Extent of the area of interest (shapefile):\n", extent)
  print(type(extent))
  print("\nCoordinate referencing system (CRS) of the shapefile:\n", outSpatialRef)
  print('EPSG code: ', epsg)
```

```
Extent of the area of interest (shapefile):
 (-16.78106458215774, -16.24270191534427, 28.170017636424944, 28.53723397651825)
<class 'tuple'>

Coordinate referencing system (CRS) of the shapefile:
 GEOGCS["WGS 84",
    DATUM["WGS_1984",
        SPHEROID["WGS 84",6378137,298.257223563,
            AUTHORITY["EPSG","7030"]],
        AUTHORITY["EPSG","6326"]],
    PRIMEM["Greenwich",0,
        AUTHORITY["EPSG","8901"]],
    UNIT["degree",0.0174532925199433,
        AUTHORITY["EPSG","9122"]],
    AXIS["Latitude",NORTH],
    AXIS["Longitude",EAST],
    AUTHORITY["EPSG","4326"]]
EPSG code:  4326
```

356

Search for available image files on the Copernicus Data Space Ecosystem

```
# go to working directory
os.chdir(rootdir)

# load user credentials for the Copernicus Data Space Ecosystem
# from the saved file with two lines of text containing username and password
with open(join(rootdir, credentials)) as f:
  lines = f.readlines()
  f.close()
username = lines[0].strip()
password = lines[1].strip()

# convert date string to YYYY-MM-DD
date_object = datetime.datetime.strptime(datefrom, "%Y%m%d")
dataspace_datefrom = date_object.strftime("%Y-%m-%d")
date_object = datetime.datetime.strptime(dateto, "%Y%m%d")
dataspace_dateto = date_object.strftime("%Y-%m-%d")

# try reading the shapefile
try:
  location = gpd.read_file(os.path.abspath(shapefile))
except FileNotFoundError:
  print("ERROR: Shapefile does not exist: {}".format(os.path.abspath(shapefile)))

# pick a point from the shapefile and convert it to well-known text format (wkt)
point1 = location["geometry"].iloc[0]
point1wkt = point1.representative_point().wkt
print(point1wkt)

# limit the number of returned search results
max_records = 50
```

357

```
# call the query function and filter by Level 2A products only
try:
  products = pygge.query_dataspace_by_polygon(
                                max_cloud_cover = clouds,
                                start_date = dataspace_datefrom,
                                end_date = dataspace_dateto,
                                processing_level = "S2MSI2A",
                                area_of_interest = point1wkt,
                                max_records = max_records
                                )
except Exception as error:
  print(f"ERROR: Query_dataspace_by_polygon received this error: {error}")
  print(f"  max_cloud_cover={clouds}")
  print(f"  start_date={dataspace_datefrom}")
  print(f"  end_date={dataspace_dateto}")
  print(f"  area_of_interest={point1}")
  print(f"  max_records={max_records}")
  sys.exit(1)
```

```
POINT (-16.511883248751005 28.353625806471598)
type
id
geometry
properties
```

358

```
if len(products) == 0:
  print("No Sentinel-2 images found that match the search criteria.")
else:
  print('Search resulted in '+str(products.shape[0])+' satellite images with '+
      str(products.shape[1])+' attributes.')
  print('\nProduct titles:')
  for i in products.title:
    print(i)
  print('\nProduct uuids:')
  for i in products.uuid:
    print(i)
```

```
Search resulted in 1 satellite images with 34 attributes.

Product titles:
S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE

Product uuids:
bc1df4f1-666d-41bf-bbdc-29a75ddf3b39
```

359

Download the individual Sentinel-2 granules to Google Drive

```
# Download all selected images in the dataframe
pygge.download_s2_data_from_dataspace(
    product_df=products,
    l1c_directory=downloaddir,
    l2a_directory=downloaddir,
    dataspace_username=username,
    dataspace_password=password
    )
```

```
    Checking 1 of 1 : S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE
   Looking for file pattern: /content/work/download/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS*
/content/work/download/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE does not exist.
       Downloading  : S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE
Obtaining the download URL - via redirect from url constructed from uuid
response.status_code: 301
download url = response.headers['Location']: https://catalogue.dataspace.copernicus.eu/odata/v1/Products(bc1df4f1-666d-41bf-bbdc-29a75ddf3b39)/$value
Final response redirects to url: https://download.dataspace.copernicus.eu/odata/v1/Products(bc1df4f1-666d-41bf-bbdc-29a75ddf3b39)/$value
Refresh access token as url redirect may take longer than 600s token expiry time
refreshing access token...
Download the zipped image file
Downloaded file temporary_path: /content/work/tmp_zfwmcwq/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE.zip
Downloaded file destination path: /content/work/download
```

360

Download Prefire Image

```
[ ]  # define the shapefile of the study area
     shapefile = join(rootdir, 'tenerife', 'POLYGON.shp')

     # check whether the shapefile exists
     if os.path.exists(shapefile):
       print('Shapefile found: '+shapefile)
     else:
       print('ERROR: Shapefile not found: '+shapefile)
       print('Upload a shapefile to your Google Drive directory: '+ rootdir)

     # Define a date range for our search
     datefrom = '20230813' # start date for imagery search
     dateto   = '20230815' # end date for imagery search

     # Define the accepted cloud cover percentage in the imagery
     clouds = 10 # maximum cloud cover [%] for imagery search

     # Copernicus Data Space Ecosystem constants - DO NOT CHANGE THESE
     DATASPACE_API_ROOT = "http://catalogue.dataspace.copernicus.eu/resto/api/collections/Sentinel2/search.json"
     DATASPACE_DOWNLOAD_URL = "https://catalogue.dataspace.copernicus.eu/odata/v1/Products"
     DATASPACE_REFRESH_TOKEN_URL = "https://identity.dataspace.copernicus.eu/auth/realms/CDSE/protocol/openid-connect/token"
```

361

```
Shapefile found: /content/drive/MyDrive/GY7709/week_7/tenerife/POLYGON.shp
```

362

Search for available image files on the Copernicus Data Space Ecosystem

```
# go to working directory
os.chdir(rootdir)

# load user credentials for the Copernicus Data Space Ecosystem
# from the saved file with two lines of text containing username and password
with open(join(rootdir, credentials)) as f:
  lines = f.readlines()
  f.close()
username = lines[0].strip()
password = lines[1].strip()

# convert date string to YYYY-MM-DD
date_object = datetime.datetime.strptime(datefrom, "%Y%m%d")
dataspace_datefrom = date_object.strftime("%Y-%m-%d")
date_object = datetime.datetime.strptime(dateto, "%Y%m%d")
dataspace_dateto = date_object.strftime("%Y-%m-%d")

# try reading the shapefile
try:
  location = gpd.read_file(os.path.abspath(shapefile))
except FileNotFoundError:
  print("ERROR: Shapefile does not exist: {}".format(os.path.abspath(shapefile)))

# pick a point from the shapefile and convert it to well-known text format (wkt)
point1 = location["geometry"].iloc[0]
point1wkt = point1.representative_point().wkt
print(point1wkt)

# limit the number of returned search results
max_records = 50
```

363

```
# call the query function and filter by Level 2A products only
try:
  products = pygge.query_dataspace_by_polygon(
                                  max_cloud_cover = clouds,
                                  start_date = dataspace_datefrom,
                                  end_date = dataspace_dateto,
                                  processing_level = "S2MSI2A",
                                  area_of_interest = point1wkt,
                                  max_records = max_records
                                  )
except Exception as error:
  print(f"ERROR: Query_dataspace_by_polygon received this error: {error}")
  print(f"  max_cloud_cover={clouds}")
  print(f"  start_date={dataspace_datefrom}")
  print(f"  end_date={dataspace_dateto}")
  print(f"  area_of_interest={point1}")
  print(f"  max_records={max_records}")
  sys.exit(1)
```

364

```
POINT (-16.490916852836676 28.319954585491963)
type
id
geometry
properties
```

365

```
Search resulted in 1 satellite images with 34 attributes.
```

366

```
Product titles:
S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE

Product uuids:
5e2c6d74-f5f4-436e-a681-509448b748c2
```

367

Download the individual Sentinel-2 granules to Google Drive

```
pygge.download_s2_data_from_dataspace(
    product_df=products,
    l1c_directory=downloaddir,
    l2a_directory=downloaddir,
    dataspace_username=username,
    dataspace_password=password
    )
```

```
    Checking 1 of 1 : S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE
   Looking for file pattern: /content/work/download/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS*
/content/work/download/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE does not exist.
        Downloading  : S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE
Obtaining the download URL - via redirect from url constructed from uuid
response.status_code: 301
download url = response.headers['Location']: https://catalogue.dataspace.copernicus.eu/odata/v1/Products(5e2c6d74-f5f4-436e-a681-509448b748c2)/$value
Final response redirects to url: https://download.dataspace.copernicus.eu/odata/v1/Products(5e2c6d74-f5f4-436e-a681-509448b748c2)/$value
Refresh access token as url redirect may take longer than 600s token expiry time
refreshing access token...
Download the zipped image file
Downloaded file temporary_path: /content/work/tmpvkvxue5n/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE.zip
Downloaded file destination path: /content/work/download
```

368

Explore the downloaded file for the Prefire Images

```
# list of the downloaded Sentinel-2 images at 20m, prefire images
print("contents of the downloaded file :")
os.listdir('/content/drive/MyDrive/GY7709/week_7/Preimage/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE/GRANULE/L2A_T28RCS_A033611_20230813T115224/IMG_DATA/R20m')
```

```
contents of the downloaded file :
['T28RCS_20230813T115229_B01_20m.jp2',
 'T28RCS_20230813T115229_AOT_20m.jp2',
 'T28RCS_20230813T115229_B06_20m.jp2',
 'T28RCS_20230813T115229_B04_20m.jp2',
 'T28RCS_20230813T115229_B02_20m.jp2',
 'T28RCS_20230813T115229_B05_20m.jp2',
 'T28RCS_20230813T115229_B03_20m.jp2',
 'T28RCS_20230813T115229_B07_20m.jp2',
 'T28RCS_20230813T115229_TCI_20m.jp2',
 'T28RCS_20230813T115229_B11_20m.jp2',
 'T28RCS_20230813T115229_WVP_20m.jp2',
 'T28RCS_20230813T115229_B12_20m.jp2',
 'T28RCS_20230813T115229_SCL_20m.jp2',
 'T28RCS_20230813T115229_B8A_20m.jp2']
```

369

Filter the prefire image files to include only the bands needed

```
# Define the directory where the Sentinel-2 image files are located at 20m
image_folder = '/content/drive/MyDrive/GY7709/week_7/Preimage/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE/GRANULE/L2A_T28RCS_A033611_20230813T115224/IMG_DATA/R20m/'

# List all files in the image folder
image_files = os.listdir(image_folder)

# Filter the image files to include only the bands needed
required_bands = ['B02', 'B03', 'B04', 'B8A', 'B12', 'SCL']  # Sentinel-2 band identifiers

# Initialize a list to store the file paths for each band
band_paths = []

# Iterate over the image files and save the file paths for the required bands
for file in image_files:
    for band in required_bands:
        if band in file:
            band_paths.append(os.path.join(image_folder, file))

# Display the file paths for the required bands
for path in band_paths:
    print(path)
```

'0

```
/content/drive/MyDrive/GY7709/week_7/Preimage/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE/GRANULE/L2A_T28RCS_A033611_20230813T115224/IMG_DATA/R20m/T28RCS_20230813T115229_B04_20m.jp2
/content/drive/MyDrive/GY7709/week_7/Preimage/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE/GRANULE/L2A_T28RCS_A033611_20230813T115224/IMG_DATA/R20m/T28RCS_20230813T115229_B02_20m.jp2
/content/drive/MyDrive/GY7709/week_7/Preimage/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE/GRANULE/L2A_T28RCS_A033611_20230813T115224/IMG_DATA/R20m/T28RCS_20230813T115229_B03_20m.jp2
/content/drive/MyDrive/GY7709/week_7/Preimage/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE/GRANULE/L2A_T28RCS_A033611_20230813T115224/IMG_DATA/R20m/T28RCS_20230813T115229_B12_20m.jp2
/content/drive/MyDrive/GY7709/week_7/Preimage/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE/GRANULE/L2A_T28RCS_A033611_20230813T115224/IMG_DATA/R20m/T28RCS_20230813T115229_SCL_20m.jp2
/content/drive/MyDrive/GY7709/week_7/Preimage/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE/GRANULE/L2A_T28RCS_A033611_20230813T115224/IMG_DATA/R20m/T28RCS_20230813T115229_B8A_20m.jp2
```

371

```
'''
-----------------------------------------------------------------
The following block of codes is modified from:

Miguel, V. (2023): Burn Severity Map of Tenerife wildfire, Aug 2023
https://github.com/miguelvillasan/BurnSeverityMap-TenerifeWildfireAug2023
Downloaded 8 April 2024
-----------------------------------------------------------------
'''
```

372

Natural colour composite before the fire

```
# Open b2, b3 and b4
band2=rasterio.open(image_folder + 'T28RCS_20230813T115229_B02_20m.jp2')
band3=rasterio.open(image_folder + 'T28RCS_20230813T115229_B03_20m.jp2')
band4=rasterio.open(image_folder + 'T28RCS_20230813T115229_B04_20m.jp2')

# Extract the metadata from b2 and update the metadata
meta = band2.meta
meta.update({"count": 3})

# Write the natural colour composite image with metadata
prefire_rgb_path = '/content/drive/MyDrive/GY7709/week_7/Preimage/pre_RGB.tif'

with rasterio.open(prefire_rgb_path, 'w', **meta) as dest:
     dest.write(band2.read(1),1)
     dest.write(band3.read(1),2)
     dest.write(band4.read(1),3)


# open the tif file
img = rasterio.open(prefire_rgb_path)
# Read in the image as a numpy array and transpose the array
image = np.array([img.read(3), img.read(2), img.read(1)]).transpose(1,2,0)
# Rescale the image
p1, p97 = np.percentile(image, (1,97))
image = exposure.rescale_intensity(image, in_range=(p1, p97)) / 100000

# Plot the resulting image
fig = plt.figure(figsize=(20,12))
show(image.transpose(2,0,1), transform=img.transform)
```

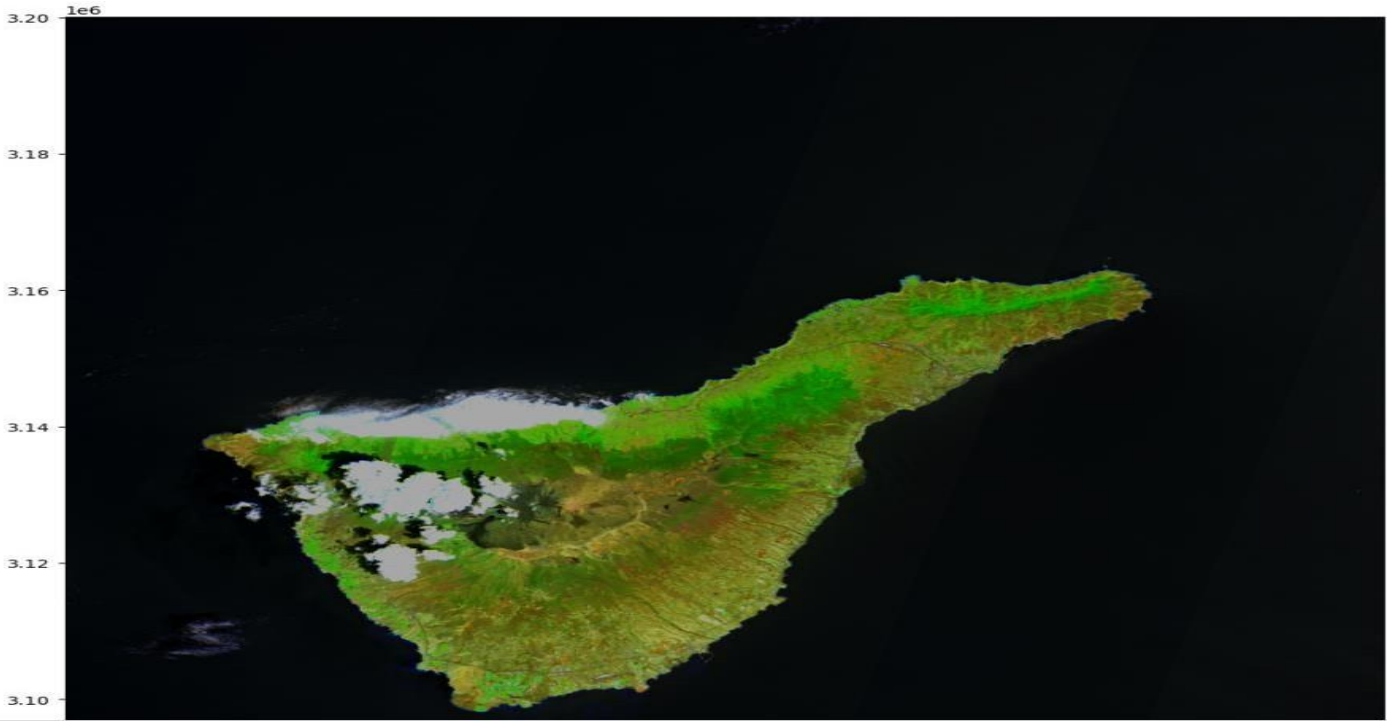
373

374

False colour composite before the fire

```
# Open b2, b3 and b4
band2=rasterio.open(image_folder + 'T28RCS_20230813T115229_B02_20m.jp2')
band8a=rasterio.open(image_folder + 'T28RCS_20230813T115229_B8A_20m.jp2')
band12=rasterio.open(image_folder + 'T28RCS_20230813T115229_B12_20m.jp2')

# Extract the metadata from b2 and update the metadata
meta = band2.meta
meta.update({"count": 3})

# Write the false colour composite image with metadata
prefire_false_path = '/content/drive/MyDrive/GY7709/week_7/Preimage/pre_false.tif'

with rasterio.open(prefire_false_path, 'w', **meta) as dest:
    dest.write(band2.read(1),1)
    dest.write(band8a.read(1),2)
    dest.write(band12.read(1),3)

# open the tif file
img = rasterio.open(prefire_false_path)
# Read in the image as a numpy array and transpose the array
image = np.array([img.read(3), img.read(2), img.read(1)]).transpose(1,2,0)
# Rescale the image
p1, p98 = np.percentile(image, (1,98))
image = exposure.rescale_intensity(image, in_range=(p1, p98)) / 100000

# Plot the resulting image
fig = plt.figure(figsize=(20,12))
show(image.transpose(2,0,1), transform=img.transform)
```

375



376

```
'''
-----------------------------------------------------------------
The following block of codes is modified from:

Balzter, H. (2024): Week 33. Zonal Statistics
https://blackboard.le.ac.uk/ultra/courses/_65627_1/outline/edit/document/_5159166_1
Downloaded 5 March 2024
-----------------------------------------------------------------
'''
```

377

Pre-processing

Before performing the analysis, the images are pre-processed. This includes removing radiometric offset and reprojection.

Remove radiometric offset from from the prefire images

```
[ ]  # get the baselines of each SAFE file
     # make an empty list to get all Sentinel 2 SAFE file directory names
     safe_ids = []
     # make an empty list where to store the processing baselines for each SAFE file
     baselines = []
     for i in band_paths:
       # get the part of the file path that contains the SAFE directory name only
       safe_id = i.split("/")[-6]
       if safe_id not in safe_ids: # if the directory name is not yet in the list
         safe_ids.append(safe_id) # remember the SAFE directory name
         baseline = safe_id.split("_")[3]
         print(safe_id+" has processing baseline "+baseline)
         baselines.append(baseline)
     print("\n")

     # Apply the radiometric correction to the files that need it. The call to the
     #   function below assumes an offset of 1000.
     prefire_radcor = [] # make an empty list of output file names
     for in_file in band_paths:
       # In the call to the pygge function below, the returned output file is called
       #   from the function and converted to a list object with the [] brackets
       #   Then appended to the prefire_radcor list.
       prefire_radcor = prefire_radcor + [
             pygge.remove_radiometric_offset_from_N0400(
             in_file,
             BOA_ADD_OFFSET = 1000
             )
           ]
```

378

```
S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE has processing baseline N0509


Removing radiometric offset from processing baseline N0400 onwards.
Removing radiometric offset from file: T28RCS_20230813T115229_B04_20m.jp2
Processing baseline: 0509
Output file: /content/drive/MyDrive/GY7709/week_7/Preimage/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE/GRANULE/L2A_T28RCS_A033611_20230813T115224/IMG_DATA/R20m/T28RCS_20230813T115229_B04_20m_A0509.jp2
Removing radiometric offset from processing baseline N0400 onwards.
Removing radiometric offset from file: T28RCS_20230813T115229_B02_20m.jp2
Processing baseline: 0509
Output file: /content/drive/MyDrive/GY7709/week_7/Preimage/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE/GRANULE/L2A_T28RCS_A033611_20230813T115224/IMG_DATA/R20m/T28RCS_20230813T115229_B02_20m_A0509.jp2
Removing radiometric offset from processing baseline N0400 onwards.
Removing radiometric offset from file: T28RCS_20230813T115229_B03_20m.jp2
Processing baseline: 0509
Output file: /content/drive/MyDrive/GY7709/week_7/Preimage/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE/GRANULE/L2A_T28RCS_A033611_20230813T115224/IMG_DATA/R20m/T28RCS_20230813T115229_B03_20m_A0509.jp2
Removing radiometric offset from processing baseline N0400 onwards.
Removing radiometric offset from file: T28RCS_20230813T115229_B12_20m.jp2
Processing baseline: 0509
Output file: /content/drive/MyDrive/GY7709/week_7/Preimage/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE/GRANULE/L2A_T28RCS_A033611_20230813T115224/IMG_DATA/R20m/T28RCS_20230813T115229_B12_20m_A0509.jp2
Removing radiometric offset from processing baseline N0400 onwards.
Removing radiometric offset from file: T28RCS_20230813T115229_SCL_20m.jp2
Processing baseline: 0509
Output file: /content/drive/MyDrive/GY7709/week_7/Preimage/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE/GRANULE/L2A_T28RCS_A033611_20230813T115224/IMG_DATA/R20m/T28RCS_20230813T115229_SCL_20m_A0509.jp2
Removing radiometric offset from processing baseline N0400 onwards.
Removing radiometric offset from file: T28RCS_20230813T115229_B8A_20m.jp2
Processing baseline: 0509
Output file: /content/drive/MyDrive/GY7709/week_7/Preimage/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE/GRANULE/L2A_T28RCS_A033611_20230813T115224/IMG_DATA/R20m/T28RCS_20230813T115229_B8A_20m_A0509.jp2
```

379

Explore the downloaded file for the Postfire Images

```
# list of the downloaded Sentinel-2 images at 20m, postfire images
print("contents of the downloaded file :")
os.listdir('/content/drive/MyDrive/GY7709/week_7/Postimage/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE/GRANULE/L2A_T28RCS_A042877_20230907T115311/IMG_DATA/R20m')
```

```
contents of the downloaded file :
['T28RCS_20230907T115221_B01_20m.jp2',
 'T28RCS_20230907T115221_AOT_20m.jp2',
 'T28RCS_20230907T115221_B12_20m.jp2',
 'T28RCS_20230907T115221_B8A_20m.jp2',
 'T28RCS_20230907T115221_B05_20m.jp2',
 'T28RCS_20230907T115221_B03_20m.jp2',
 'T28RCS_20230907T115221_B11_20m.jp2',
 'T28RCS_20230907T115221_B06_20m.jp2',
 'T28RCS_20230907T115221_WVP_20m.jp2',
 'T28RCS_20230907T115221_B02_20m.jp2',
 'T28RCS_20230907T115221_B07_20m.jp2',
 'T28RCS_20230907T115221_B04_20m.jp2',
 'T28RCS_20230907T115221_TCI_20m.jp2',
 'T28RCS_20230907T115221_SCL_20m.jp2']
```

380

Filter the postfire image files to include only the bands needed

```
np.array(img,np.float32)
```

```
# Define the directory where your Sentinel-2 image files are located
image_folder = '/content/drive/MyDrive/GY7709/week_7/Postimage/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE/GRANULE/L2A_T28RCS_A042877_20230907T115311/IMG_DATA/R20m/'

# List all files in the image folder
image_files = os.listdir(image_folder)

# Filter the image files to include only the bands needed
required_bands = ['B02', 'B03', 'B04', 'B8A', 'B12', 'SCL']  # Sentinel-2 band identifiers

# Initialize a list to store the file paths for each band
band_paths = []

# Iterate over the image files and save the file paths for the required bands
for file in image_files:
    for band in required_bands:
        if band in file:
            band_paths.append(os.path.join(image_folder, file))

# Display the file paths for the required bands
for path in band_paths:
    print(path)
```

381

```
/content/drive/MyDrive/GY7709/week_7/Postimage/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE/GRANULE/L2A_T28RCS_A042877_20230907T115311/IMG_DATA/R20m/T28RCS_20230907T115221_B12_20m.jp2
/content/drive/MyDrive/GY7709/week_7/Postimage/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE/GRANULE/L2A_T28RCS_A042877_20230907T115311/IMG_DATA/R20m/T28RCS_20230907T115221_B8A_20m.jp2
/content/drive/MyDrive/GY7709/week_7/Postimage/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE/GRANULE/L2A_T28RCS_A042877_20230907T115311/IMG_DATA/R20m/T28RCS_20230907T115221_B03_20m.jp2
/content/drive/MyDrive/GY7709/week_7/Postimage/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE/GRANULE/L2A_T28RCS_A042877_20230907T115311/IMG_DATA/R20m/T28RCS_20230907T115221_B02_20m.jp2
/content/drive/MyDrive/GY7709/week_7/Postimage/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE/GRANULE/L2A_T28RCS_A042877_20230907T115311/IMG_DATA/R20m/T28RCS_20230907T115221_B04_20m.jp2
/content/drive/MyDrive/GY7709/week_7/Postimage/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE/GRANULE/L2A_T28RCS_A042877_20230907T115311/IMG_DATA/R20m/T28RCS_20230907T115221_SCL_20m.jp2
```

```
'''
-----------------------------------------------------------------
The following block of codes is modified from:

Miguel, V. (2023): Burn Severity Map of Tenerife wildfire, Aug 2023
https://github.com/miguelvillasan/BurnSeverityMap-TenerifeWildfireAug2023
Downloaded 8 April 2024
-----------------------------------------------------------------
'''
```

Natural colour composite after the fire

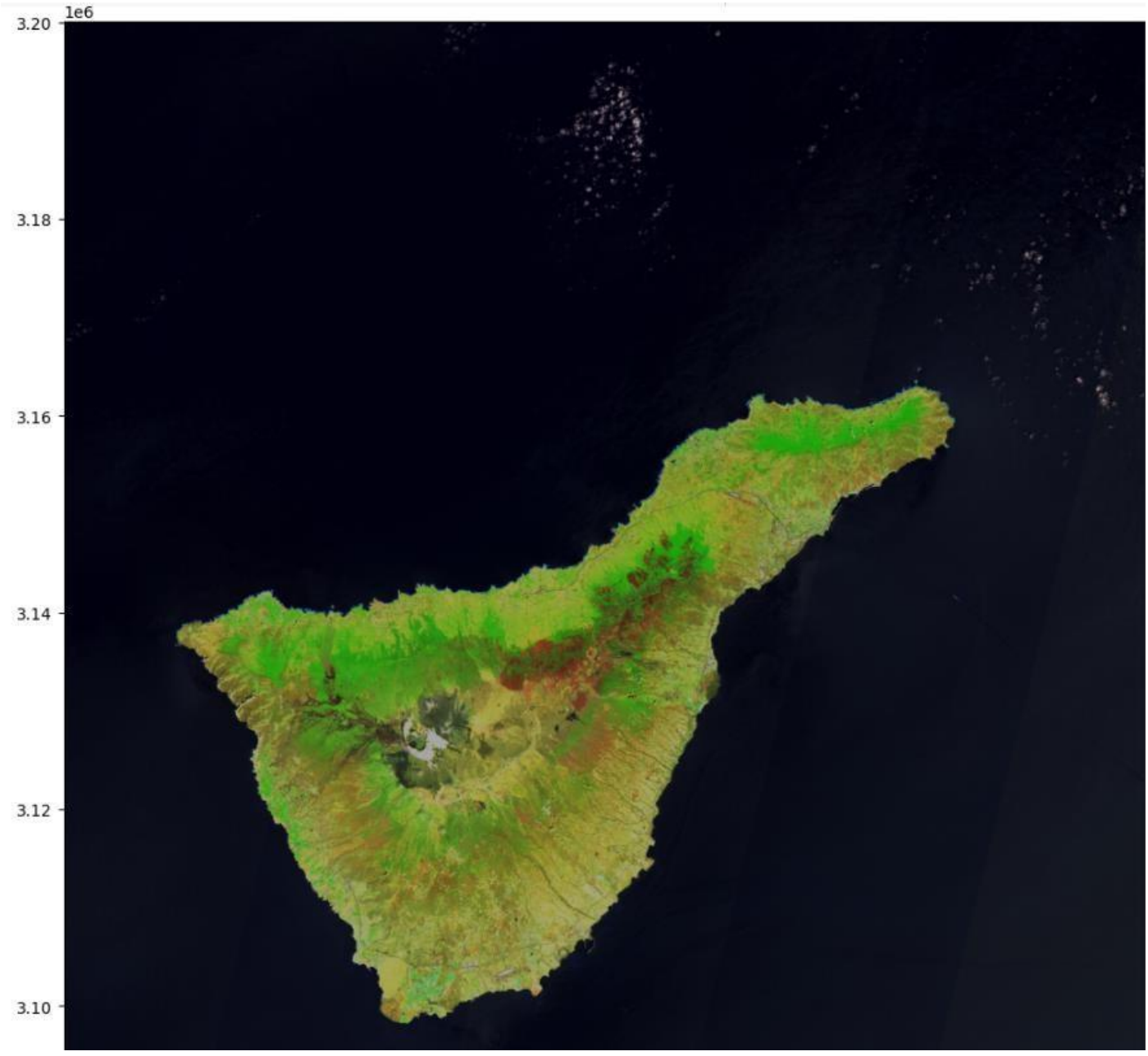
```
# Open b2, b3 and b4
band2=rasterio.open(image_folder + 'T28RCS_20230907T115221_B02_20m.jp2')
band3=rasterio.open(image_folder + 'T28RCS_20230907T115221_B03_20m.jp2')
band4=rasterio.open(image_folder + 'T28RCS_20230907T115221_B04_20m.jp2')

# Extract the metadata from b2 and update the metadata
meta = band2.meta
meta.update({"count": 3})

# Write the natural colour composite image with metadata
postfire_rgb_path = '/content/drive/MyDrive/GY7709/week_7/Postimage/post_RGB.tif'

with rasterio.open(postfire_rgb_path, 'w', **meta) as dest:
    dest.write(band2.read(1),1)
    dest.write(band3.read(1),2)
    dest.write(band4.read(1),3)


# open the tif file
img = rasterio.open(postfire_rgb_path)
# Read in the image as a numpy array and transpose the array
image = np.array([img.read(3), img.read(2), img.read(1)]).transpose(1,2,0)
# Rescale the image
p1, p97 = np.percentile(image, (1,97))
image = exposure.rescale_intensity(image, in_range=(p1, p97)) / 100000

# Plot the resulting image
fig = plt.figure(figsize=(20,12))
show(image.transpose(2,0,1), transform=img.transform)
```

385

False colour composite after the fire

```
# Open b2, b3 and b4
band2=rasterio.open(image_folder + 'T28RCS_20230907T115221_B02_20m.jp2')
band8a=rasterio.open(image_folder + 'T28RCS_20230907T115221_B8A_20m.jp2')
band12=rasterio.open(image_folder + 'T28RCS_20230907T115221_B12_20m.jp2')

# Extract the metadata from b2 and update the metadata
meta = band2.meta
meta.update({"count": 3})

# Write the false colour composite image with metadata
postfire_false_path = '/content/drive/MyDrive/GY7709/week_7/Postimage/post_false.tif'

with rasterio.open(postfire_false_path, 'w', **meta) as dest:
    dest.write(band2.read(1),1)
    dest.write(band8a.read(1),2)
    dest.write(band12.read(1),3)

# open the tif file
img = rasterio.open(postfire_false_path)
# Read in the image as a numpy array and transpose the array
image = np.array([img.read(3), img.read(2), img.read(1)]).transpose(1,2,0)
# Rescale the image
p1, p98 = np.percentile(image, (1,98))
image = exposure.rescale_intensity(image, in_range=(p1, p98)) / 100000

# Plot the resulting image
fig = plt.figure(figsize=(20,12))
show(image.transpose(2,0,1), transform=img.transform)
```

386




```
'''
-----------------------------------------------------------------
The following block of codes is modified from:

Balzter, H. (2024): Week 33. Zonal Statistics
https://blackboard.le.ac.uk/ultra/courses/_65627_1/outline/edit/document/_5159166_1
Downloaded 5 March 2024
-----------------------------------------------------------------
'''
```

Removing radiometric offset from the postfire images

```
# get the baselines of each SAFE file
# make an empty list to get all Sentinel 2 SAFE file directory names
safe_ids = []
# make an empty list where to store the processing baselines for each SAFE file
baselines = []
for i in band_paths:
  # get the part of the file path that contains the SAFE directory name only
  safe_id = i.split("/")[-6]
  if safe_id not in safe_ids: # if the directory name is not yet in the list:
    safe_ids.append(safe_id) # remember the SAFE directory name
    baseline = safe_id.split("_")[3]
    print(safe_id+" has processing baseline "+baseline)
    baselines.append(baseline)
print("\n")

# Apply the radiometric correction to the files that need it. The call to the
#   function below assumes an offset of 1000.
postfire_radcor = [] # make an empty list of output file names
for in_file in band_paths:
  #   In the call to the pygge function below, the returned output file is called
  #   from the function and converted to a list object with the [] brackets
  #   so we can append it to the postfire_radcor list.
  postfire_radcor = postfire_radcor + [
        pygge.remove_radiometric_offset_from_N0400(
        in_file,
        BOA_ADD_OFFSET = 1000
        )
      ]
```

392

```
S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE has processing baseline N0509


Removing radiometric offset from processing baseline N0400 onwards.
Removing radiometric offset from file: T28RCS_20230907T115221_B12_20m.jp2
Processing baseline: 0509
Output file: /content/drive/MyDrive/GY7709/week_7/Postimage/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE/GRANULE/L2A_T28RCS_A042877_20230907T115311/IMG_DATA/R20m/T28RCS_20230907T115221_B12_20m_A0509.jp2
Removing radiometric offset from processing baseline N0400 onwards.
Removing radiometric offset from file: T28RCS_20230907T115221_B8A_20m.jp2
Processing baseline: 0509
Output file: /content/drive/MyDrive/GY7709/week_7/Postimage/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE/GRANULE/L2A_T28RCS_A042877_20230907T115311/IMG_DATA/R20m/T28RCS_20230907T115221_B8A_20m_A0509.jp2
Removing radiometric offset from processing baseline N0400 onwards.
Removing radiometric offset from file: T28RCS_20230907T115221_B03_20m.jp2
Processing baseline: 0509
Output file: /content/drive/MyDrive/GY7709/week_7/Postimage/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE/GRANULE/L2A_T28RCS_A042877_20230907T115311/IMG_DATA/R20m/T28RCS_20230907T115221_B03_20m_A0509.jp2
Removing radiometric offset from processing baseline N0400 onwards.
Removing radiometric offset from file: T28RCS_20230907T115221_B02_20m.jp2
Processing baseline: 0509
Output file: /content/drive/MyDrive/GY7709/week_7/Postimage/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE/GRANULE/L2A_T28RCS_A042877_20230907T115311/IMG_DATA/R20m/T28RCS_20230907T115221_B02_20m_A0509.jp2
Removing radiometric offset from processing baseline N0400 onwards.
Removing radiometric offset from file: T28RCS_20230907T115221_B04_20m.jp2
Processing baseline: 0509
Output file: /content/drive/MyDrive/GY7709/week_7/Postimage/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE/GRANULE/L2A_T28RCS_A042877_20230907T115311/IMG_DATA/R20m/T28RCS_20230907T115221_B04_20m_A0509.jp2
Removing radiometric offset from processing baseline N0400 onwards.
Removing radiometric offset from file: T28RCS_20230907T115221_SCL_20m.jp2
Processing baseline: 0509
Output file: /content/drive/MyDrive/GY7709/week_7/Postimage/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE/GRANULE/L2A_T28RCS_A042877_20230907T115311/IMG_DATA/R20m/T28RCS_20230907T115221_SCL_20m_A0509.jp2
```

393

```
'''
-----------------------------------------------------------------
The following block of codes is modified from:

Miguel, V. (2023): Burn Severity Map of Tenerife wildfire, Aug 2023
https://github.com/miguelvillasan/BurnSeverityMap-TenerifeWildfireAug2023
Downloaded 8 April 2024
-----------------------------------------------------------------
'''
```

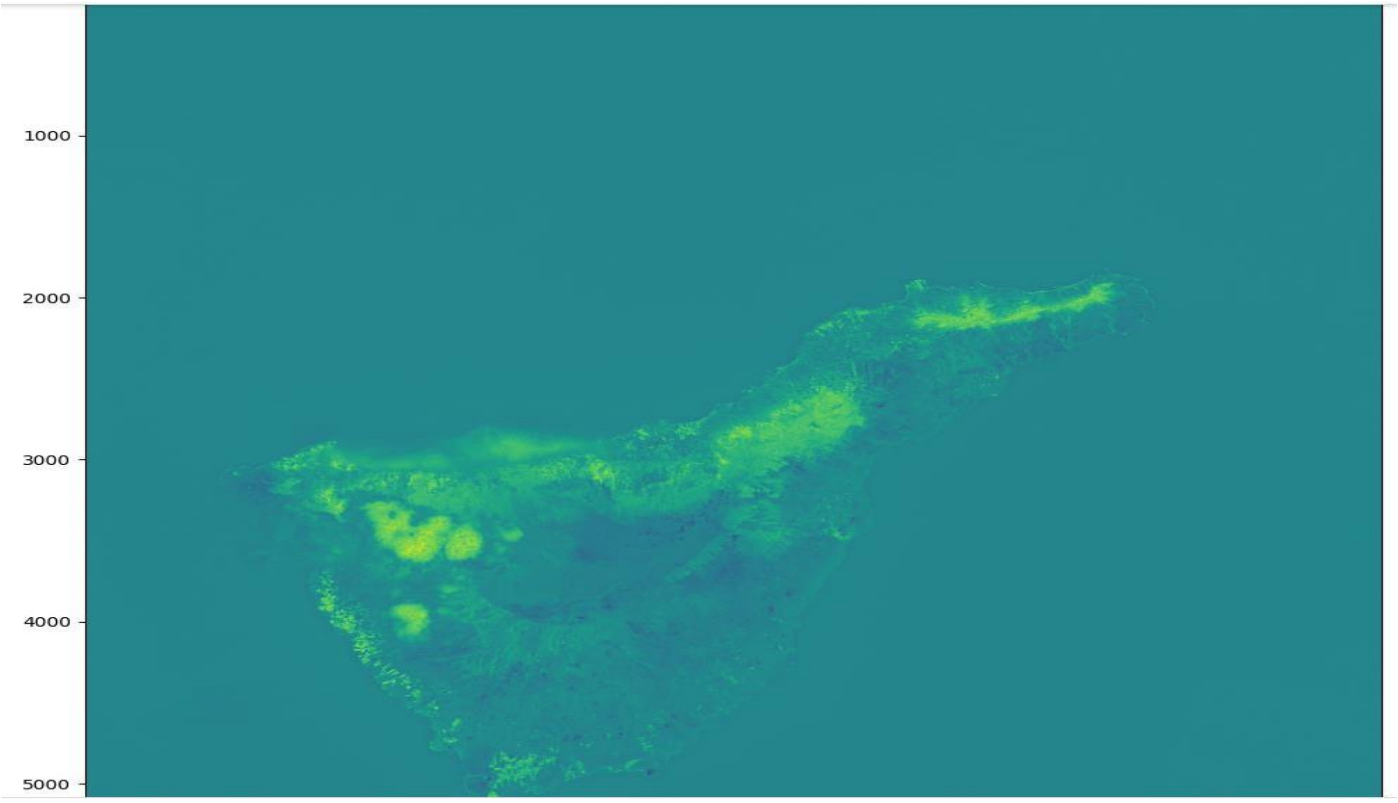
394

Calculate the Normalized Burn Ratio (NBR) and Differenced NBR images

NBR on 13 August 2023 (pre-fire)

```
prefire = '/content/drive/MyDrive/GY7709/week_7/Preimage/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE/GRANULE/L2A_T28RCS_A033611_20230813T115224/IMG_DATA/R20m/'
```

```
# Open b8a and b12
b8a=rasterio.open(prefire + 'T28RCS_20230813T115229_B8A_20m_A0509.jp2')
b12=rasterio.open(prefire + 'T28RCS_20230813T115229_B12_20m_A0509.jp2')

# read NIR(b8a) and SWIR(b12) as arrays
nir_band = b8a.read()
swir_band = b12.read()

# Calculate NBR
nbr = (nir_band.astype(float)-swir_band.astype(float)) / (nir_band+swir_band)

# Extract and update the metadata
meta = b12.meta
meta.update(driver='GTiff')
meta.update(dtype=rasterio.float32)

# Write the NBR image with metadata
prefire_nbr_path = '/content/drive/MyDrive/GY7709/week_7/Preimage/prefireNBR.tif'

with rasterio.open(prefire_nbr_path, 'w', **meta) as dest:
    dest.write(nbr.astype(rasterio.float32))

# Open and read the image
img = rasterio.open(prefire_nbr_path)
img_r = img.read()
image = np.asarray(img_r)

# Define a plot
fig = plt.figure(figsize=(20,12))

# Show the figure
```



398

```
'''
-----------------------------------------------------------------
The following block of codes is modified from:

Balzter, H. (2024): Week 33. Zonal Statistics
https://blackboard.le.ac.uk/ultra/courses/_65627_1/outline/edit/document/_5159166_1
Downloaded 5 March 2024
-----------------------------------------------------------------
'''
```

399

Warp the NBR image to the same projection as the shapefile.

```
# Warp the NBR image
# Generate the new filename for the warped output file
prewarped_image = os.path.splitext(prefire_nbr_path)[0] + "_warped.tif"

# Call the easy_warp function from pygge
pygge.easy_warp(prefire_nbr_path, prewarped_image, epsg, res="bilinear", skip=False,
                xres=None, yres=None)
```

400

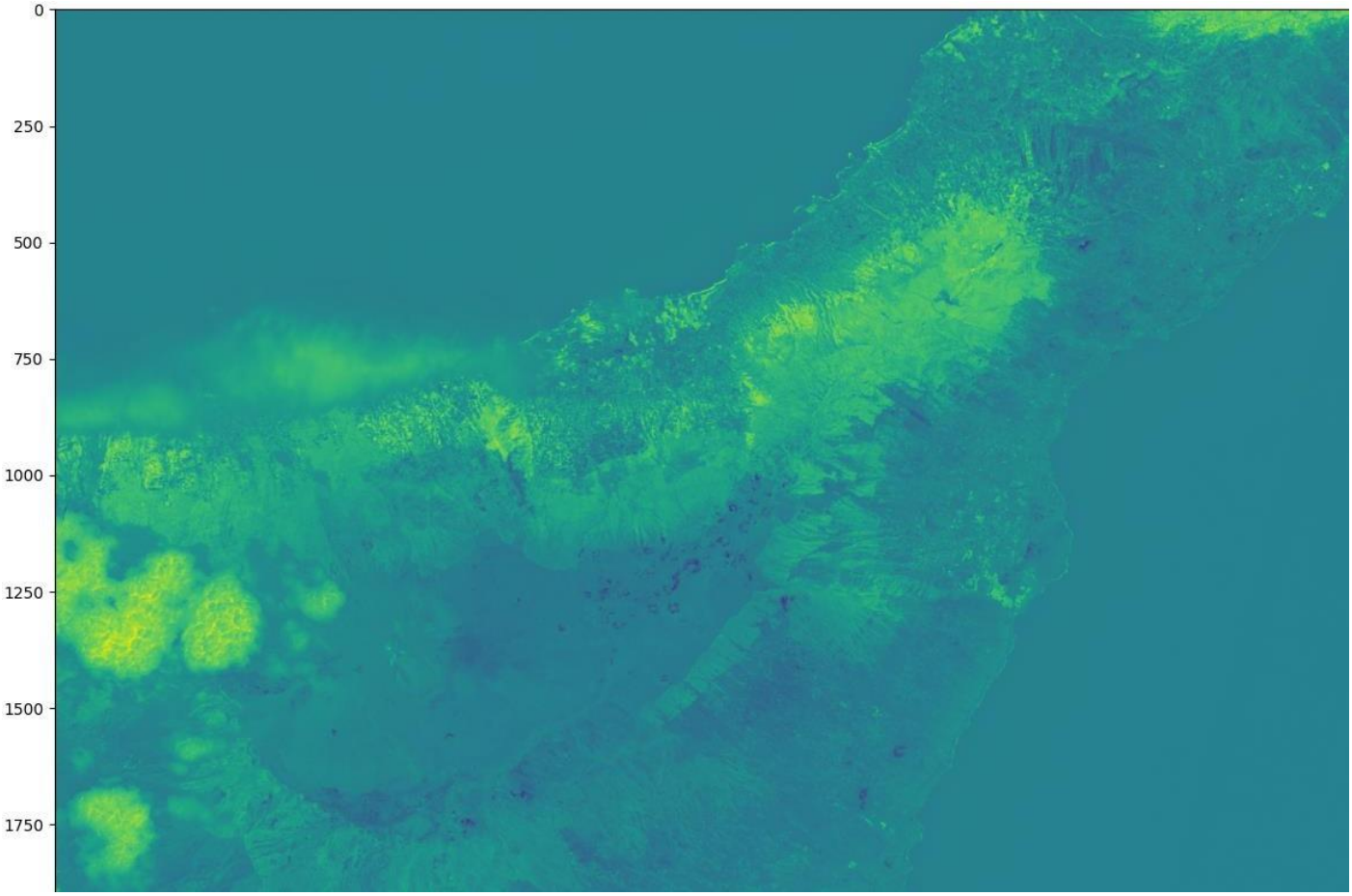
```
Warping file: /content/drive/MyDrive/GY7709/week_7/Preimage/prefireNBR.tif
  to output file:/content/drive/MyDrive/GY7709/week_7/Preimage/prefireNBR_warped.tif
rio warp /content/drive/MyDrive/GY7709/week_7/Preimage/prefireNBR.tif /content/drive/MyDrive/GY7709/week_7/Preimage/prefireNBR_warped.tif --dst-crs EPSG:4326 --resampling bilinear --overwrite
```

401

Clip the image to the extent of the shapefile

```
[ ]  # Define the filename for the clipped image
     preclip = os.path.splitext(prewarped_image)[0] + "_clipped.tif"

     # Clip the warped image to the shapefile extent
     pygge.easy_clip(prewarped_image, preclip, extent)

     # Open and read the warped image
     clip_img = rasterio.open(preclip)
     clipped_img = clip_img.read()
     preclipped_img = np.asarray(clipped_img)

     # Define a plot
     fig = plt.figure(figsize=(15,10))

     # Show the figure
     show(preclipped_img)
```

```
Window coordinates:  Window(col_off=1401, row_off=2009, width=2790, height=1904)
Window array size:  (1, 1904, 2790)
```

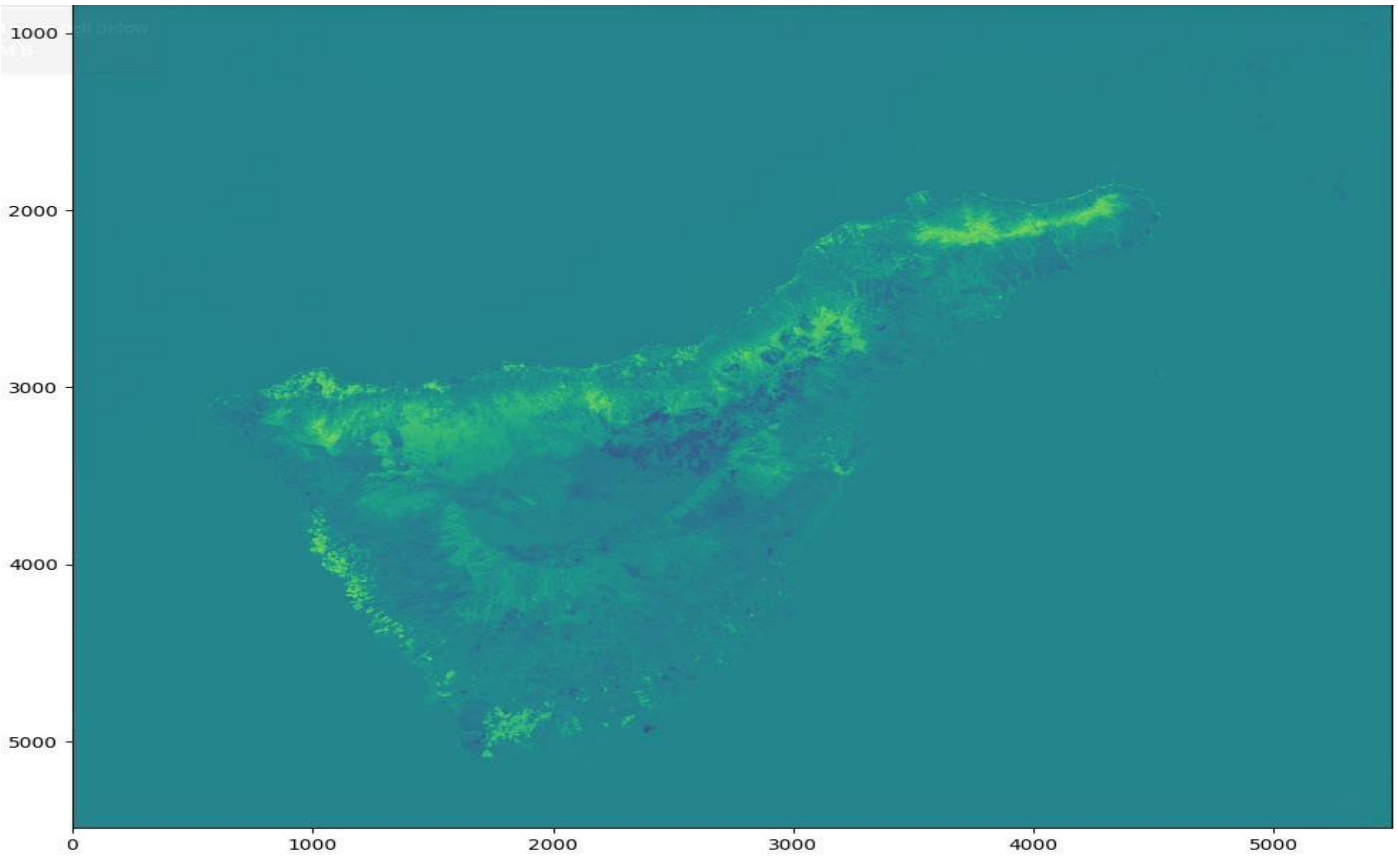


NBR on 7th September 2023 (post-fire)

```
postfire = '/content/drive/MyDrive/GY7709/week_7/Postimage/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE/GRANULE/L2A_T28RCS_A042877_20230907T115311/IMG_DATA/R20m/'
```

```
'''
---------------------------------------------------------------
The following block of codes is modified from:

Miguel, V. (2023): Burn Severity Map of Tenerife wildfire, Aug 2023
https://github.com/miguelvillasan/BurnSeverityMap-TenerifeWildfireAug2023
Downloaded 8 April 2024
---------------------------------------------------------------
'''
```

```
# Open b8a and b12
b8a=rasterio.open(postfire + 'T28RCS_20230907T115221_B8A_20m_A0509.jp2')
b12=rasterio.open(postfire + 'T28RCS_20230907T115221_B12_20m_A0509.jp2')

# read NIR(b8a) and SWIR(b12) as arrays
nir_band = b8a.read()
swir_band = b12.read()

# Calculate NBR
nbr = (nir_band.astype(float)-swir_band.astype(float)) / (nir_band+swir_band)

# Extract and update the metadata
meta = b12.meta
meta.update(driver='GTiff')
meta.update(dtype=rasterio.float32)

# Write the NBR image with metadata
postfire_nbr_path = '/content/drive/MyDrive/GY7709/week_7/Postimage/postfireNBR.tif'

with rasterio.open(postfire_nbr_path, 'w', **meta) as dest:
    dest.write(nbr.astype(rasterio.float32))

# Open and read the image
img = rasterio.open(postfire_nbr_path)
img_r = img.read()
image = np.asarray(img_r)

# Define a plot
fig = plt.figure(figsize=(20,12))

# Show the figure
show(image);
```



410

```
'''
-----------------------------------------------------------------
The following block of codes is modified from:

Balzter, H. (2024): Week 33. Zonal Statistics
https://blackboard.le.ac.uk/ultra/courses/_65627_1/outline/edit/document/_5159166_1
Downloaded 5 March 2024
-----------------------------------------------------------------
'''
```

411

Warp the NBR image to the same projection as the shapefile.

```
# Warp the NBR image
# Generate the new filename for the warped output file
postwarped_image = os.path.splitext(postfire_nbr_path)[0] + "_warped.tif"

# Call the easy_warp function from pygge
pygge.easy_warp(postfire_nbr_path, postwarped_image, epsg, res="bilinear", skip=False,
                xres=None, yres=None)
```

412

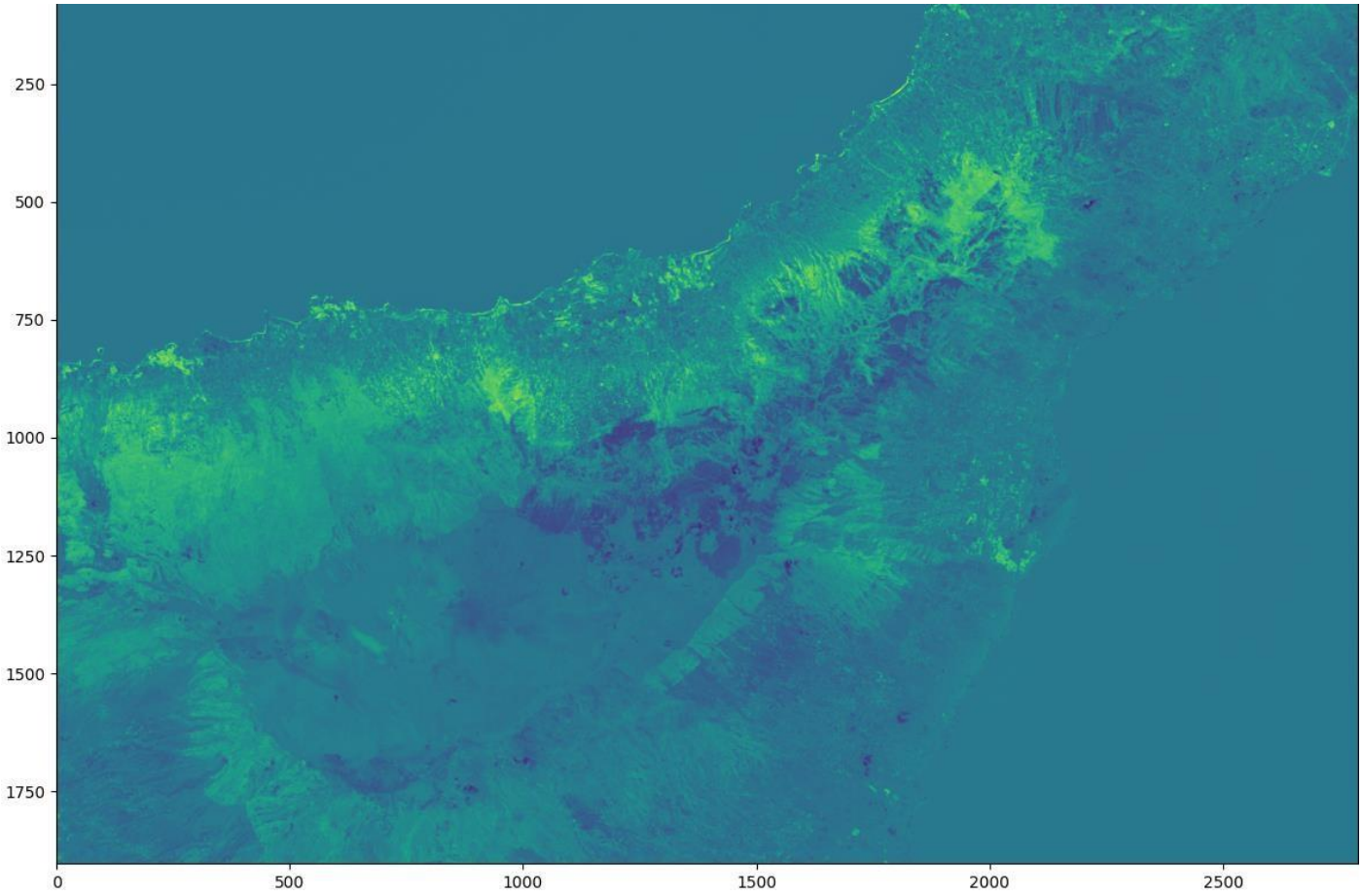
```
Warping file: /content/drive/MyDrive/GY7709/week_7/Postimage/postfireNBR.tif
  to output file:/content/drive/MyDrive/GY7709/week_7/Postimage/postfireNBR_warped.tif
rio warp /content/drive/MyDrive/GY7709/week_7/Postimage/postfireNBR.tif /content/drive/MyDrive/GY7709/week_7/Postimage/postfireNBR_warped.tif --dst-crs EPSG:4326 --resampling bilinear --overwrite
```

413

```
# Define the filename for the clipped image
postclip = os.path.splitext(postwarped_image)[0] + "_clipped.tif"

# Clip the warped image to the shapefile extent
pygge.easy_clip(postwarped_image, postclip, extent)

# Open and read the warped image
clip_img = rasterio.open(postclip)
clipped_img = clip_img.read()
postclipped_img = np.asarray(clipped_img)

# Define a plot
fig = plt.figure(figsize=(15,10))

# Show the figure
show(postclipped_img)
```

```
Window coordinates:  Window(col_off=1401, row_off=2009, width=2790, height=1904)
Window array size:  (1, 1904, 2790)
```

414



415

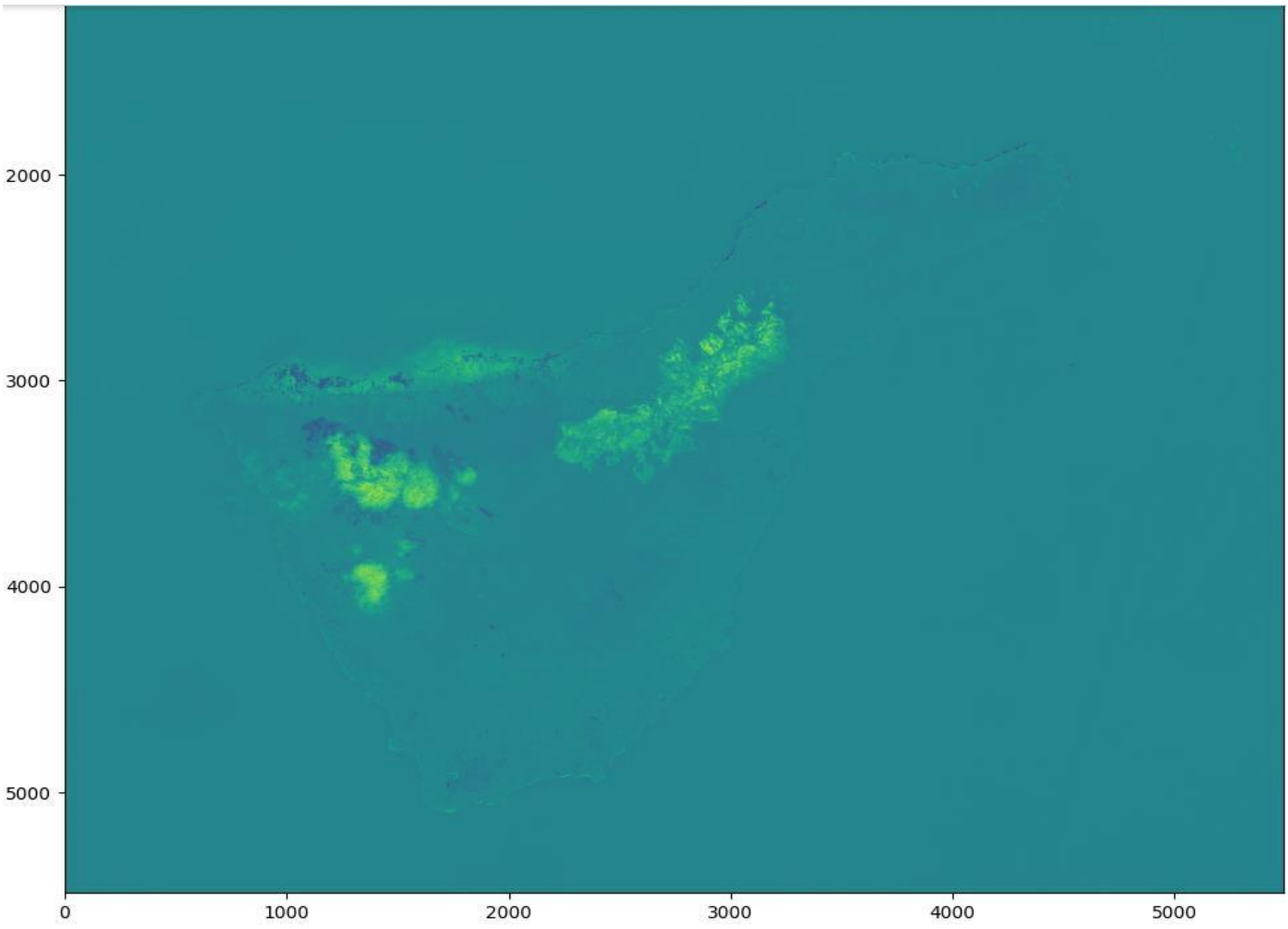
```
'''
---------------------------------------------------------------
The following block of codes is modified from:

Miguel, V. (2023): Burn Severity Map of Tenerife wildfire, Aug 2023
https://github.com/miguelvillasan/BurnSeverityMap-TenerifeWildfireAug2023
Downloaded 8 April 2024
---------------------------------------------------------------
'''
```

416

Differenced NBR (pre-fire minus post-fire NBR)

```
[ ]  # Open the NBR images from 13 August 2023 (pre-fire) and 7 September 2023 (post-fire)
     pre_nbr = rasterio.open(prefire_nbr_path)
     post_nbr = rasterio.open(postfire_nbr_path)

     # Read the images in as arrays
     prefire_nbr = pre_nbr.read()
     postfire_nbr = post_nbr.read()

     dnbr = prefire_nbr.astype(rasterio.float32)-postfire_nbr.astype(rasterio.float32)

     meta = pre_nbr.meta
     meta.update(driver='GTiff')
     meta.update(dtype=rasterio.float32)

     dnbr_path = '/content/drive/MyDrive/GY7709/week_7/NBR_Images/DNBR.tif'

     with rasterio.open(dnbr_path, 'w', **meta) as dest:
         dest.write(dnbr.astype(rasterio.float32))

     # Rescale the image
     img = rasterio.open(dnbr_path)
     img_r = img.read()
     image = np.asarray(img_r)

     fig = plt.figure(figsize=(20,12))
     show(image)
```

417



418

```
'''
-----------------------------------------------------------------
The following block of codes is modified from:

Balzter, H. (2024): Week 33. Zonal Statistics
https://blackboard.le.ac.uk/ultra/courses/_65627_1/outline/edit/document/_5159166_1
Downloaded 5 March 2024
-----------------------------------------------------------------
'''
```

419

```
# Warp the dNBR image
# Generate the new filename for the warped output file
dnbr_warped = os.path.splitext(dnbr_path)[0] + "_warped.tif"

# Call the easy_warp function from pygge
pygge.easy_warp(dnbr_path, dnbr_warped, epsg, res="bilinear", skip=False,
                xres=None, yres=None)
```

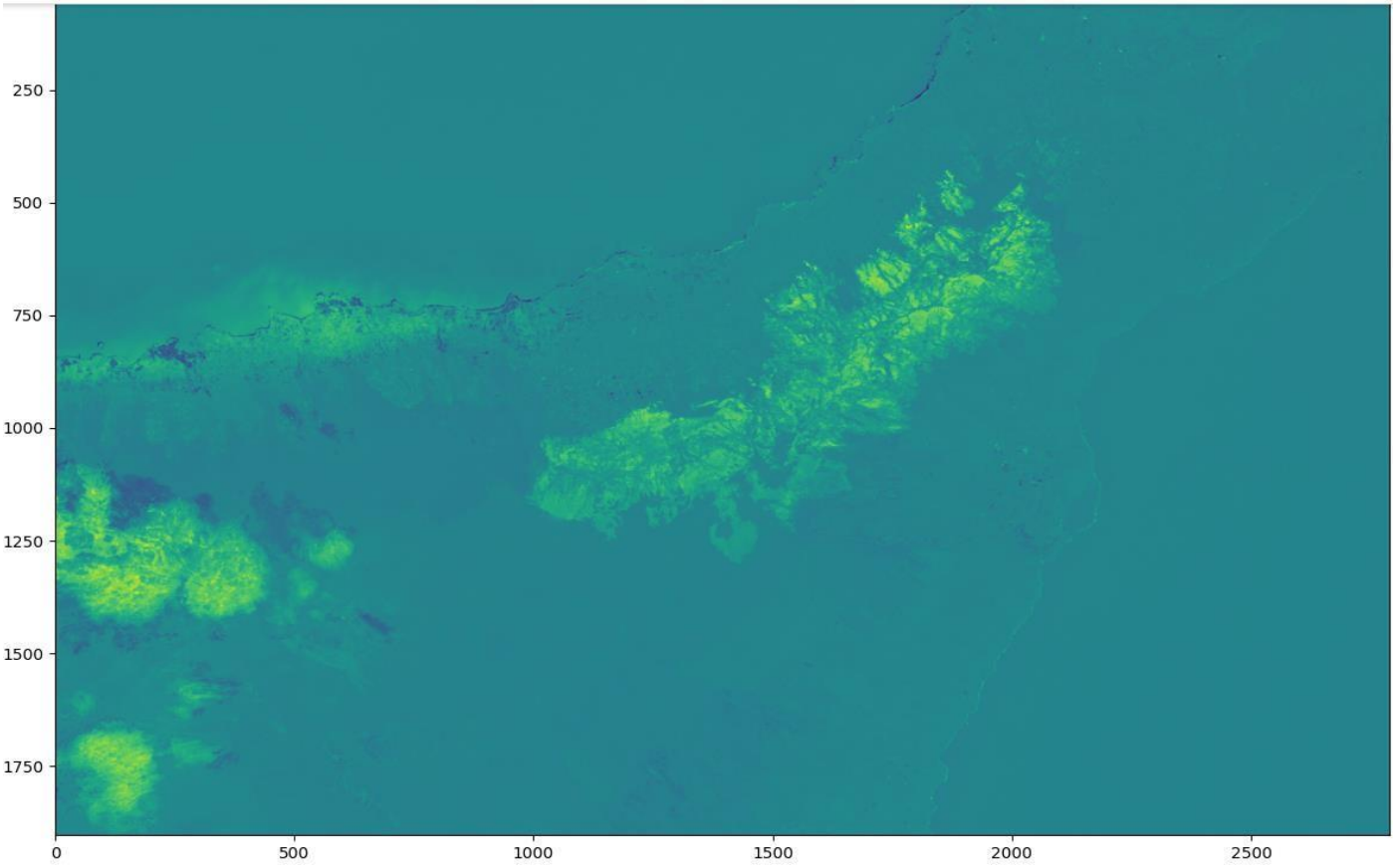
420

```
Warping file: /content/drive/MyDrive/GY7709/week_7/NBR_Images/DNBR.tif
  to output file:/content/drive/MyDrive/GY7709/week_7/NBR_Images/DNBR_warped.tif
rio warp /content/drive/MyDrive/GY7709/week_7/NBR_Images/DNBR.tif /content/drive/MyDrive/GY7709/week_7/NBR_Images/DNBR_warped.tif --dst-crs EPSG:4326 --resampling bilinear --overwrite
```

421

```
# Define the filename for the clipped image
dnbrclip = os.path.splitext(dnbr_warped)[0] + "_clipped.tif"

# Clip the warped image to the shapefile extent
pygge.easy_clip(dnbr_warped, dnbrclip, extent)

# Open and read the warped image
clip_img = rasterio.open(dnbrclip)
clipped_img = clip_img.read()
dbnr_clipped = np.asarray(clipped_img)

# Define a plot
fig = plt.figure(figsize=(15,10))

# Show the figure
show(dbnr_clipped)
```

422



423

```
'''
-----------------------------------------------------------------
The following block of codes is modified from:

Miguel, V. (2023): Burn Severity Map of Tenerife wildfire, Aug 2023
https://github.com/miguelvillasan/BurnSeverityMap-TenerifeWildfireAug2023
Downloaded 8 April 2024
-----------------------------------------------------------------
'''
```

424

Burn Severity Map

Using the calculated dNBR image to assess burn severity after the fire, where areas with higher dNBR values indicate more severe damage and areas with negative dNBR values showing increased vegetation productivity. The dNBR image is classified according to burn severity ranges proposed by the United States Geological Survey (USGS).

Firstly a water mask is created using the Scene Classification Layer from the postfire image (7 September, 2023) at 20m resolution coresponding to the resolution of the dNBR.

```
[ ]  scl_image = rasterio.open(postfire + 'T28RCS_20230907T115221_SCL_20m.jp2')
```

```
[ ]  # read the image as an array
     scl = scl_image.read()
     scl = np.array(scl)
     scl
```

```
array([[[6, 6, 6, ..., 6, 6, 6],
        [6, 6, 6, ..., 6, 6, 6],
        [6, 6, 6, ..., 6, 6, 6],
        ...,
        [6, 6, 6, ..., 6, 6, 6],
        [6, 6, 6, ..., 6, 6, 6],
        [6, 6, 6, ..., 6, 6, 6]]], dtype=uint8)
```

```
[ ]  # replace all pixels with nan except the water pixel value of 6
     water = np.where(scl != 6, np.nan, scl)
     water
```

```
array([[[6., 6., 6., ..., 6., 6., 6.],
        [6., 6., 6., ..., 6., 6., 6.],
        [6., 6., 6., ..., 6., 6., 6.],
        ...,
        [6., 6., 6., ..., 6., 6., 6.],
        [6., 6., 6., ..., 6., 6., 6.],
        [6., 6., 6., ..., 6., 6., 6.]]])
```

425

```
# plot the water pixel
fig = plt.figure(figsize=(10,10))
show(water);
```

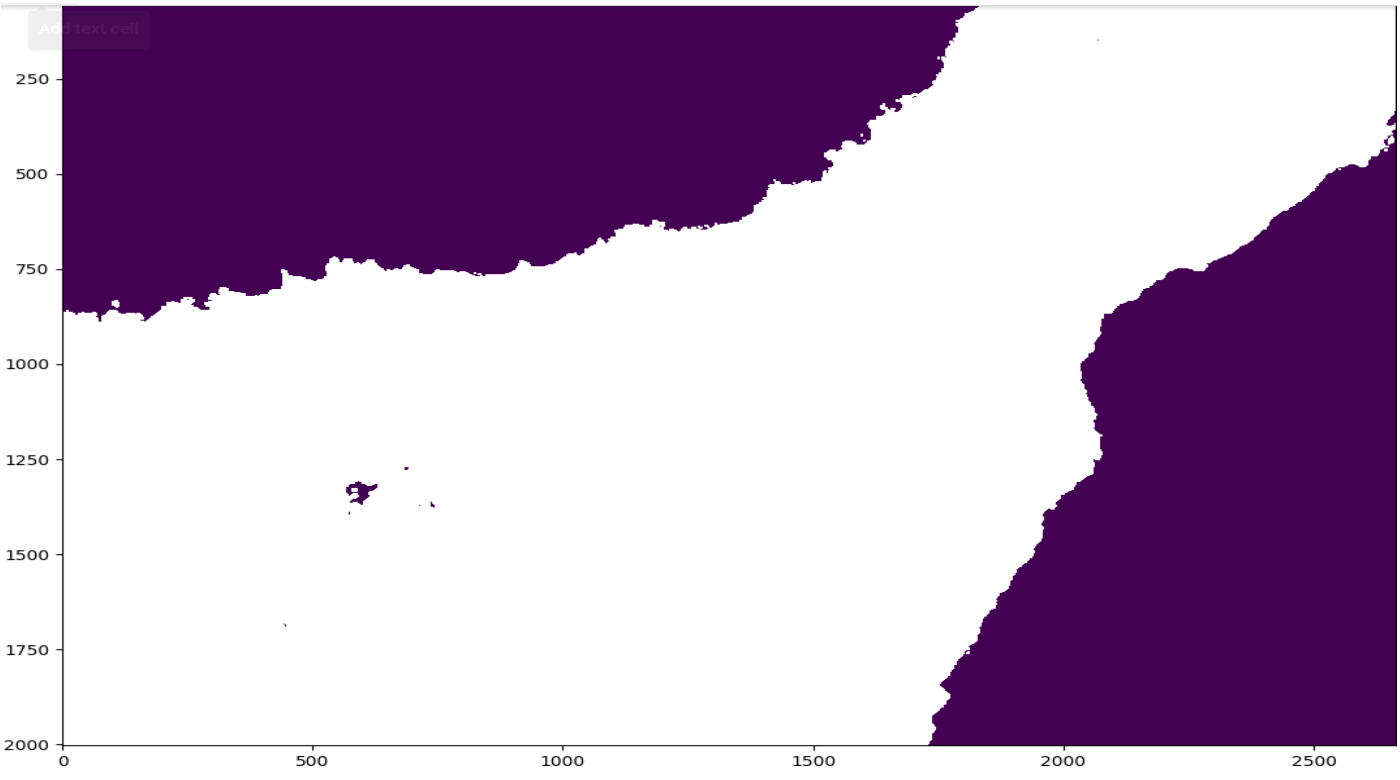
426

```
# Extract the metadata, update the driver to GTiff, and the data type to float32.
meta = scl_image.meta
meta.update(driver='GTiff')
meta.update(dtype=rasterio.float32)

# save the resulting water mask image
water_path = '/content/drive/MyDrive/GY7709/week_7/NBR_Images/water.tif'

with rasterio.open(water_path, 'w', **meta) as dest:
    dest.write(water.astype(rasterio.float32))
```

```
# Define the filename for the clipped image
waterclip = os.path.splitext(water_path)[0] + "_clipped.tif"

# Clip the warped image to the shapefile extent
pygge.easy_clip(water_path, waterclip, extent)

# Open and read the warped image
clip_img = rasterio.open(waterclip)
clipped_img = clip_img.read()
water_clipped = np.asarray(clipped_img)

# Define a plot
fig = plt.figure(figsize=(15,10))

# Show the figure
show(water_clipped)
```

```
Window coordinates:  Window(col_off=1257, row_off=2134, width=2665, height=2003)
Window array size:  (1, 2003, 2665)
```

427



428

The dNBR raster classification

Using the burn severity ranges proposed by the United States Geological Survey (USGS).

```
[ ]  # Open and read the dNBR image
     img = rasterio.open(dnbrclip)
     img_r = img.read(1)

     # multiply the array by 1000 to rescale it
     image = np.asarray(img_r)*1000
```

```
[ ]  # Open and read the water mask image
     water_img = rasterio.open(waterclip)
     water_r = water_img.read(1)
     # convert into a numpy array
     water = np.asarray(water_r)
```

```
[ ]  # set colours for plotting and classes
     cmap = ListedColormap([[122/255., 135/255., 55/255.],
                            [172/255., 190/255., 77/255.],
                            [10/255., 224/255., 66/255.],
                            [255/255., 247/255., 11/255.],
                            [255/255., 175/255., 56/255.],
                            [255/255., 100/255., 27/255.],
                            [164/255., 31/255., 214/255.],
                            [255/255., 255/255., 255/255.]])
```

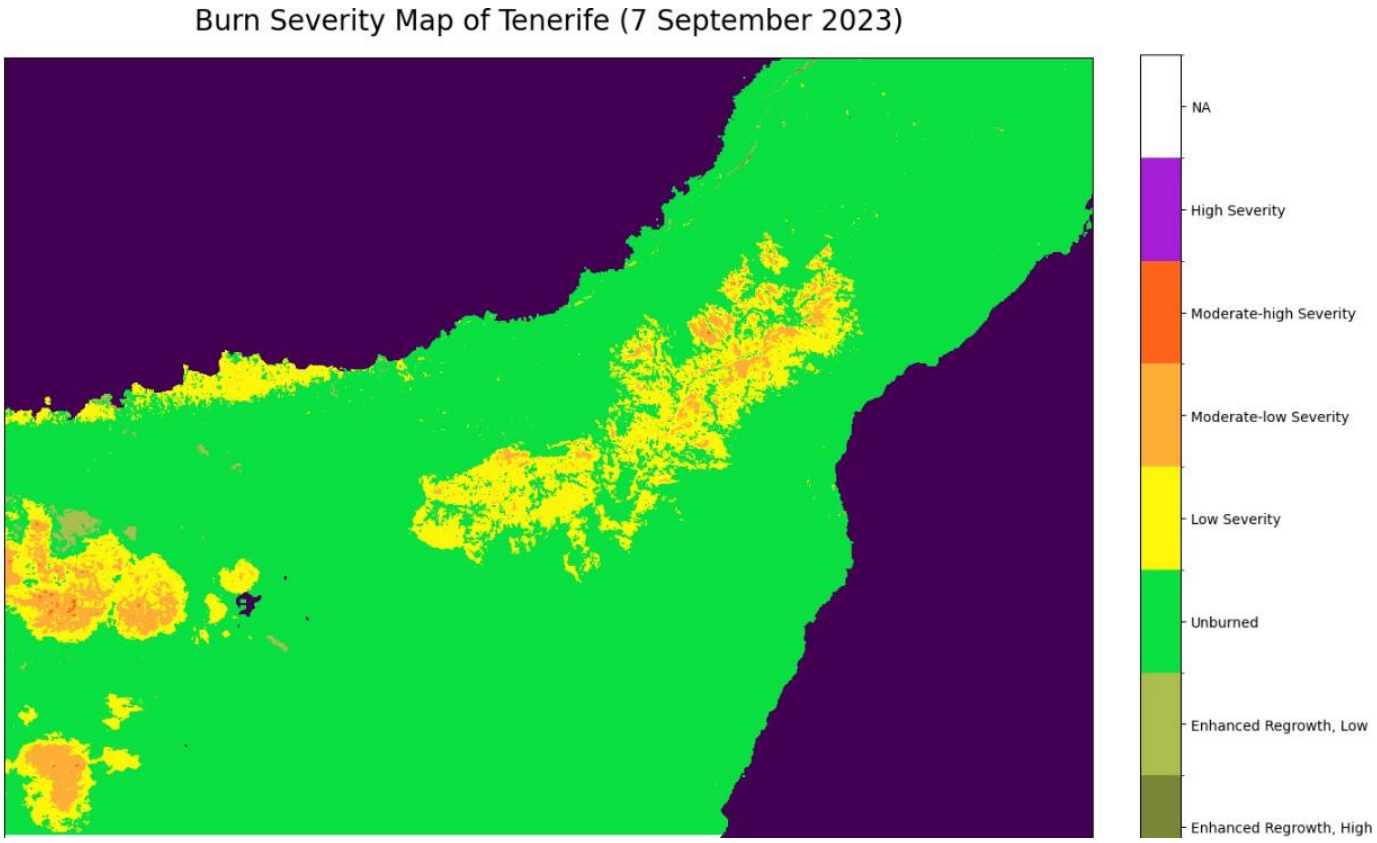
```
[ ]  # define the levels for the respective burn severity classes
     bounds = [-1000, -251, -101, 99, 269, 439, 659, 1300, 2000]
     norm = matplotlib.colors.BoundaryNorm(bounds, cmap.N)
```

429

Plot the severity map

```
# Define the figure
fig, ax = plt.subplots(figsize=(15, 15), subplot_kw={'xticks': [], 'yticks': []})

# Use `.imshow()` to plot the dNBR image with the custom colormap and norm
cax = ax.imshow(image, cmap=cmap, norm = norm)

# Add the water mask image to the figure using `.imshow()`
ax.imshow(water)

# Add a figure title
plt.title('Burn Severity Map of Tenerife (7 September 2023)', fontsize=20, pad=20.0)

# Add a colorbar
cbar = fig.colorbar(cax, ax=ax, fraction=0.035, pad=0.04, ticks=[-625, -176, -1, 184, 354, 549, 980, 1650])

# Set the tick labels to reflect the burn severity classes
cbar.ax.set_yticklabels(['Enhanced Regrowth, High',
         'Enhanced Regrowth, Low',
         'Unburned',
         'Low Severity',
         'Moderate-low Severity',
         'Moderate-high Severity',
         'High Severity',
         'NA'])

# Save the plot as an image file
plt.savefig('Burn_severity_map.png', dpi=300, bbox_inches='tight', pad_inches=0.1)

# Show the plot
plt.show()
```

430



431

Calculate the Normalized Difference Vegetation Index(NDVI) and NDVI Difference

NDVI on 13 August 2023 (pre-fire)

432

```
red_band = '/content/drive/MyDrive/GY7709/week_7/Preimage/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE/GRANULE/L2A_T28RCS_A033611_20230813T115224/IMG_DATA/R20m/T28RCS_20230813T115229_B04_20m_A0509.jp2'
nir_band = '/content/drive/MyDrive/GY7709/week_7/Preimage/S2B_MSIL2A_20230813T115229_N0509_R123_T28RCS_20230813T154508.SAFE/GRANULE/L2A_T28RCS_A033611_20230813T115224/IMG_DATA/R20m/T28RCS_20230813T115229_B8A_20m_A0509.jp2'
```

433

```
'''
------------------------------------------------------------
The following block of codes is modified from:

Balzter, H. (2024): Week 31. Time-series analysis of Normalised Difference Vegetation Index (NDVI)
https://blackboard.le.ac.uk/ultra/courses/_65627_1/outline/edit/document/_5159164_1
Downloaded 22 February 2024
------------------------------------------------------------
'''
```

434

```
# Define the output directory for the NDVI image
pre_ndvi = '/content/drive/MyDrive/GY7709/week_7/Preimage/preNDVI.tif'

 # calculate the NDVI raster
pygge.easy_ndvi(red_band, nir_band, pre_ndvi)
```

435

Warp the NDVI images to the same projection as the shapefile.

```
# Warp the NDVI image
# Generate the new filename for the warped output file
prewarped = os.path.splitext(pre_ndvi)[0] + "_warped.tif"

# Call the easy_warp function from pygge
pygge.easy_warp(pre_ndvi, prewarped, epsg, res="bilinear", skip=False,
                xres=None, yres=None)
```

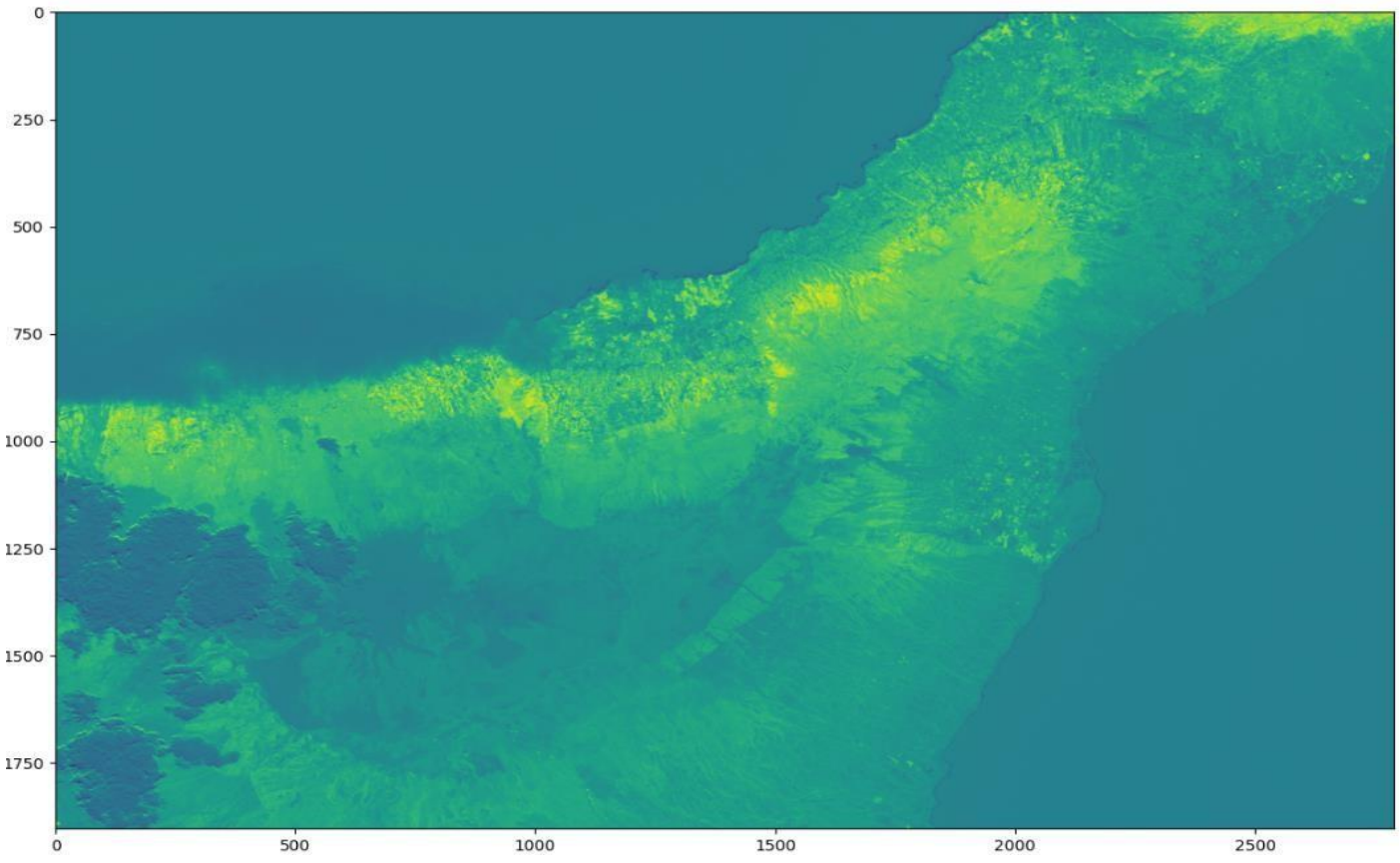
436

Clip the Prefire NDVI Image

```
preclip = os.path.splitext(prewarped)[0] + "_clipped.tif"

# Clip the warped image to the shapefile extent
pygge.easy_clip(prewarped, preclip, extent)

with rasterio.open(preclip) as src:
    # Read the clipped image
    clipped_image = src.read()

# Plot the clipped image
plt.figure(figsize=(15,10))
show(clipped_image)
```

```
Window coordinates:  Window(col_off=0, row_off=0, width=2790, height=1904)
Window array size:  (1, 1904, 2790)
```

437



438

NDVI on 7 September 2023 (post-fire)

```
red_band = '/content/drive/MyDrive/GY7709/week_7/Postimage/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE/GRANULE/L2A_T28RCS_A042877_20230907T115311/IMG_DATA/R20m/T28RCS_20230907T115221_B04_20m_A0509.jp2'
nir_band = '/content/drive/MyDrive/GY7709/week_7/Postimage/S2A_MSIL2A_20230907T115221_N0509_R123_T28RCS_20230907T194401.SAFE/GRANULE/L2A_T28RCS_A042877_20230907T115311/IMG_DATA/R20m/T28RCS_20230907T115221_B8A_20m_A0509.jp2'
```

```
# Define the output directory for the NDVI image
post_ndvi = '/content/drive/MyDrive/GY7709/week_7/Postimage/postNDVI.tif'

 # calculate the NDVI raster
pygge.easy_ndvi(red_band, nir_band, post_ndvi)
```

Warp the NDVI images to the same projection as the shapefile.

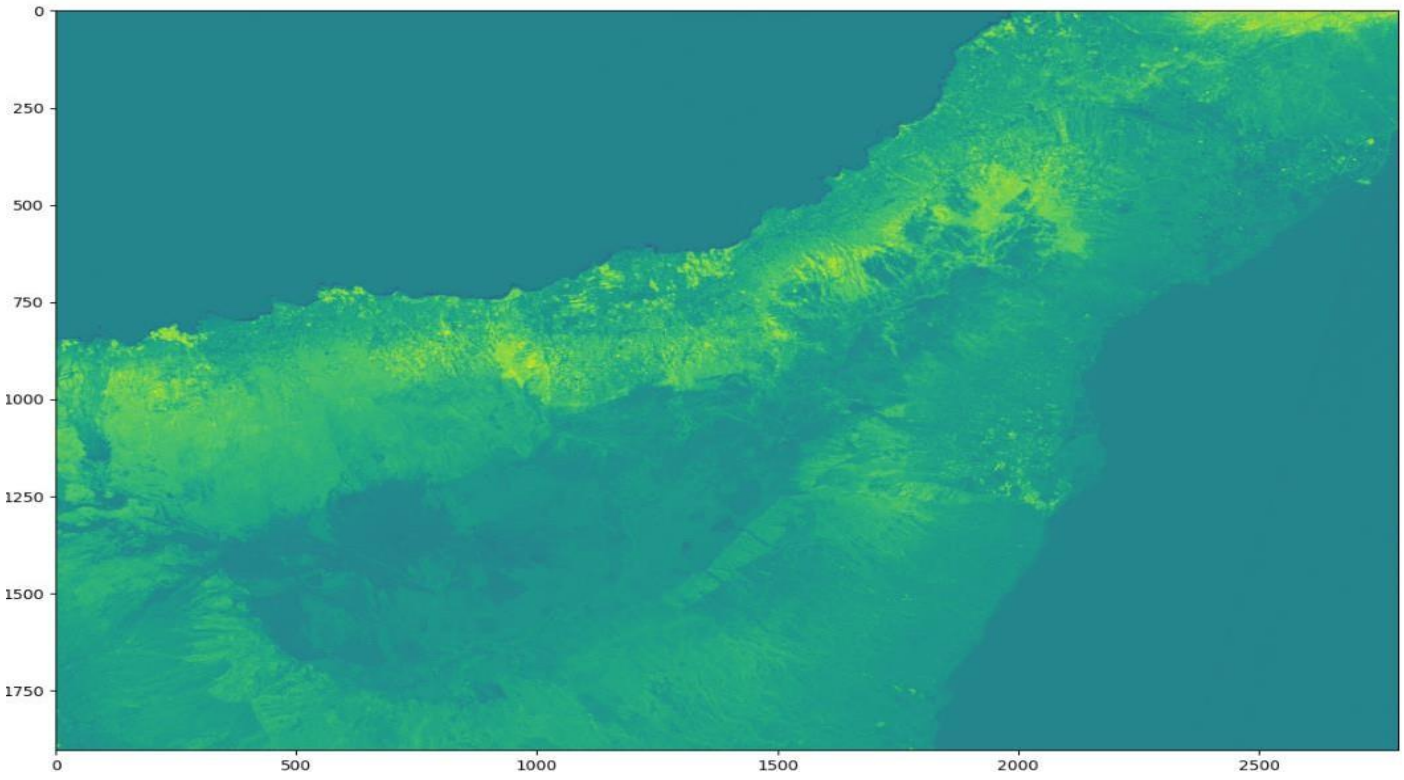
```
# Warp the NDVI image
# Generate the new filename for the warped output file
postwarped = os.path.splitext(post_ndvi)[0] + "_warped.tif"

# Call the easy_warp function from pygge
pygge.easy_warp(post_ndvi, postwarped, epsg, res="bilinear", skip=False,
                xres=None, yres=None)
```

```
Warping file: /content/drive/MyDrive/GY7709/week_7/Postimage/postNDVI.tif
  to output file:/content/drive/MyDrive/GY7709/week_7/Postimage/postNDVI_warped.tif
rio warp /content/drive/MyDrive/GY7709/week_7/Postimage/postNDVI.tif /content/drive/MyDrive/GY7709/week_7/Postimage/postNDVI_warped.tif --dst-crs EPSG:4326 --resampling bilinear --overwrite
```

Clip the Postfire NDVI Image

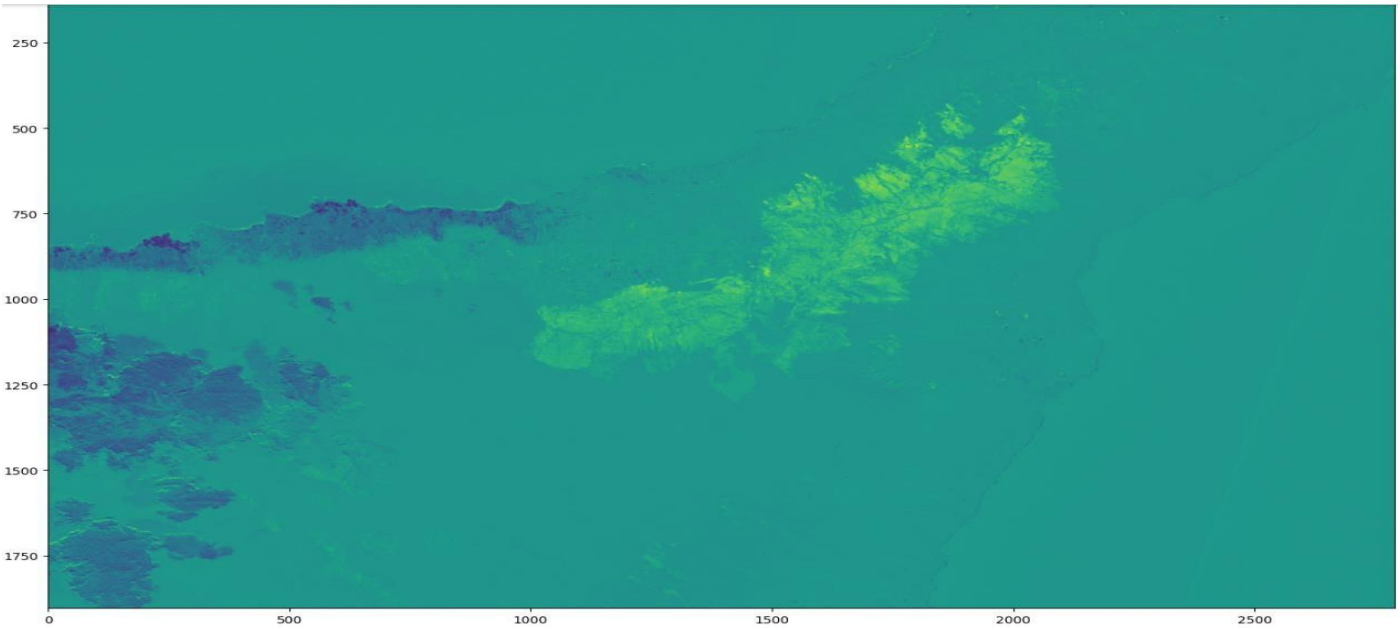
```
postclip = os.path.splitext(postwarped)[0] + "_clipped.tif"

# Clip the warped image to the shapefile extent
pygge.easy_clip(postwarped, postclip, extent)

with rasterio.open(postclip) as src:
    # Read the clipped image
    clipped_image = src.read()

# Plot the clipped image
plt.figure(figsize=(15,10))
show(clipped_image)
```

```
Window coordinates:  Window(col_off=1401, row_off=2009, width=2790, height=1904)
Window array size:  (1, 1904, 2790)
```



Differenced NDVI (pre-fire minus post-fire NDVI)

```
# Open the NDVI images from 13 August 2023 (pre-fire) and 7 September 2023 (post-fire)
ndvi_prefire = rasterio.open(preclip)
ndvi_postfire = rasterio.open(postclip)

# Read the images in as arrays
prefire_ndvi = ndvi_prefire.read()
postfire_ndvi = ndvi_postfire.read()

diff_ndvi = prefire_ndvi.astype(rasterio.float32)-postfire_ndvi.astype(rasterio.float32)

meta = ndvi_prefire.meta
meta.update(driver='GTiff')
meta.update(dtype=rasterio.float32)

dndvi_path = '/content/drive/MyDrive/GY7709/week_7/NBR_Images/dNDVI.tif'

with rasterio.open(dndvi_path, 'w', **meta) as dest:
    dest.write(diff_ndvi.astype(rasterio.float32))

# Rescale the image
img = rasterio.open(dndvi_path)
img_r = img.read()
image = np.asarray(img_r)

fig = plt.figure(figsize=(20,12))
show(image)
```



447

Average NBR and dNBR values

To further assess the impact of the wildfire, five polygons representing burned areas and five polygons representing unburned areas were digitized in a GIS and the average pre-fire and post-fire NBR as well as the difference in NBR (dNBR) within each polygon is calculated

448

```
# define the name of the shapefile containing the training polygons
polygon = '/content/drive/MyDrive/GY7709/week_7/Polygons/polygon.shp'

# Load the shapefile containing polygons
shapefile_gdf = gpd.read_file(polygon)

# Path to prefire NBR raster
prefire_nbr_path = '/content/drive/MyDrive/GY7709/week_7/NBR_file/prefireNBR_warped_clipped.tif'

# Path to postfire NBR raster
postfire_nbr_path = '/content/drive/MyDrive/GY7709/week_7/NBR_file/postfireNBR_warped_clipped.tif'

# Create empty lists to store NBR values for burned and unburned polygons
nbr_burned_prefire = []
nbr_burned_postfire = []
nbr_unburned_prefire = []
nbr_unburned_postfire = []
dnbr_burned = []
dnbr_unburned = []

'''
----------------------------------------------------------------
The following block of codes is modified from:

Alan, D. (2022): Masking a raster using a shapefile
https://github.com/rasterio/rasterio/blob/main/docs/topics/masking-by-shapefile.rst
Downloaded 18 April 2024
----------------------------------------------------------------
'''

# Iterate over the polygons in the shapefile
for idx, polygon in shapefile_gdf.iterrows():
    # Mask the prefire NBR raster using the polygon geometry
    masked_prefire_nbr, _ = mask(rasterio.open(prefire_nbr_path), [polygon.geometry], crop=True)
    # Mask the postfire NBR raster using the polygon geometry
    masked_postfire_nbr, _ = mask(rasterio.open(postfire_nbr_path), [polygon.geometry], crop=True)
```

449

```
        # Calculate the mean prefire NBR value within the masked area
        mean_nbr_prefire = np.nanmean(masked_prefire_nbr)
        # Calculate the mean postfire NBR value within the masked area
        mean_nbr_postfire = np.nanmean(masked_postfire_nbr)

        # Append the mean NBR values to the appropriate lists based on polygon classification
        if polygon['Class'] == 1:
            nbr_burned_prefire.append(mean_nbr_prefire)
            nbr_burned_postfire.append(mean_nbr_postfire)
        elif polygon['Class'] == 2:
            nbr_unburned_prefire.append(mean_nbr_prefire)
            nbr_unburned_postfire.append(mean_nbr_postfire)

        # Calculate the mean dNBR value within the masked area
        mean_dnbr = mean_nbr_postfire - mean_nbr_prefire

        # Append the mean dNBR value to the appropriate list based on polygon classification
        if polygon['Class'] == 1:
            dnbr_burned.append(mean_dnbr)
        elif polygon['Class'] == 2:
            dnbr_unburned.append(mean_dnbr)

    # Print the calculated NBR and dNBR values
    print("Mean prefire NBR for burned polygons:", nbr_burned_prefire)
    print("Mean postfire NBR for burned polygons:", nbr_burned_postfire)
    print("Mean prefire NBR for unburned polygons:", nbr_unburned_prefire)
    print("Mean postfire NBR for unburned polygons:", nbr_unburned_postfire)
    print("Mean dNBR for burned polygons:", dnbr_burned)
    print("Mean dNBR for unburned polygons:", dnbr_unburned)
```

450

```
Shapefile found: /content/drive/MyDrive/GY7709/week_7/Polygons/polygon.shp
Mean prefire NBR for burned polygons: [0.09290446, 0.09473369, 0.102745086, 0.12279252, 0.12860052]
Mean postfire NBR for burned polygons: [-0.050336115, -0.010416414, 0.019940637, -0.04565905, 0.010591455]
Mean prefire NBR for unburned polygons: [0.0070581464, 0.012222133, 0.055848587, 0.1103236, 0.060000986]
Mean postfire NBR for unburned polygons: [0.009380384, 0.010222499, 0.061954167, 0.11256265, 0.056427594]
Mean dNBR for burned polygons: [-0.14324057, -0.1051501, -0.08280445, -0.16845158, -0.11800907]
Mean dNBR for unburned polygons: [0.002322238, -0.0019996334, 0.0061055794, 0.0022390485, -0.0035733916]
```

451

```
data = {
    'Mean prefire NBR (burned)': nbr_burned_prefire,
    'Mean postfire NBR (burned)': nbr_burned_postfire,
    'Mean prefire NBR (unburned)': nbr_unburned_prefire,
    'Mean postfire NBR (unburned)': nbr_unburned_postfire,
    'Mean dNBR (burned)': dnbr_burned,
    'Mean dNBR (unburned)': dnbr_unburned
}

df = pd.DataFrame(data)
```

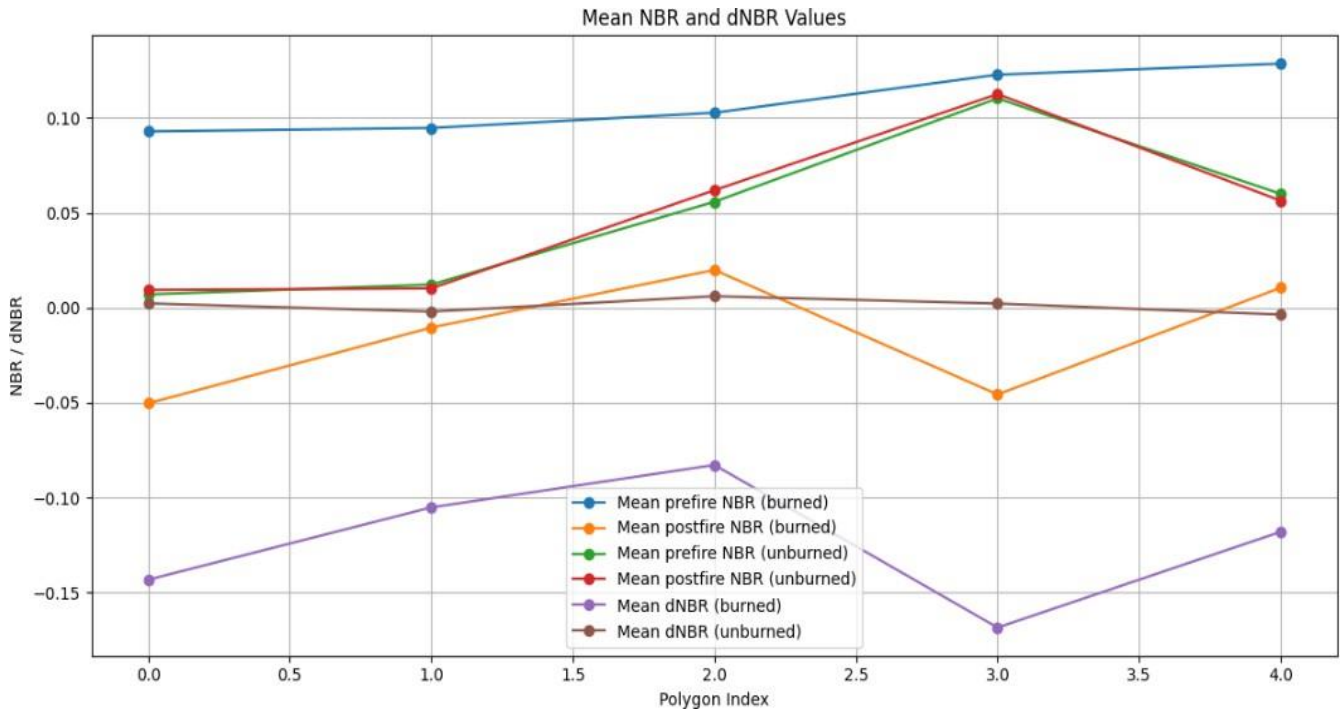
```
# Specify the path for the Excel file
excel_file_path = '/content/drive/MyDrive/GY7709/week_7/NBR_and_dNBR.xlsx'

# Save the DataFrame to an Excel file
df.to_excel(excel_file_path, index=False)

print('Data saved to Excel file:', excel_file_path)
```

```
Data saved to Excel file: /content/drive/MyDrive/GY7709/week_7/NBR_and_dNBR.xlsx
```

452

```
# Plot the NBR and dNBR values
plt.figure(figsize=(12, 6))

# Plot mean NBR values for burned and unburned polygons
plt.plot(df.index, df['Mean prefire NBR (burned)'], label='Mean prefire NBR (burned)', marker='o')
plt.plot(df.index, df['Mean postfire NBR (burned)'], label='Mean postfire NBR (burned)', marker='o')
plt.plot(df.index, df['Mean prefire NBR (unburned)'], label='Mean prefire NBR (unburned)', marker='o')
plt.plot(df.index, df['Mean postfire NBR (unburned)'], label='Mean postfire NBR (unburned)', marker='o')

# Plot mean dNBR values for burned and unburned polygons
plt.plot(df.index, df['Mean dNBR (burned)'], label='Mean dNBR (burned)', marker='o')
plt.plot(df.index, df['Mean dNBR (unburned)'], label='Mean dNBR (unburned)', marker='o')

# Add labels, title, and legend
plt.xlabel('Polygon Index')
plt.ylabel('NBR / dNBR')
plt.title('Mean NBR and dNBR Values')
plt.legend()

# Show the plot
plt.grid(True)
plt.tight_layout()
plt.show()
```

453



454

Correlation between dNBR and dNDVI

Correlation analysis using 50 random points particular within high severity areas to assess the correlation between dNBR and dNDVI

```
[ ] # Polygon of the high severity area
    burn_shapefile = '/content/drive/MyDrive/GY7709/week_7/Burnshapefile/POLYGON.shp'
```

```
[ ] # Get the shapefile layer's extent, CRS and EPSG code
    extent, outSpatialRef, epsg = pygge.get_shp_extent(burn_shapefile)
    print("Extent of the area of interest (shapefile):\n", extent)
    print(type(extent))
    print("\nCoordinate referencing system (CRS) of the shapefile:\n", outSpatialRef)
    print('EPSG code: ', epsg)
```

```
Extent of the area of interest (shapefile):
 (-16.462583697483247, -16.39097462762333, 28.371342537086203, 28.4343316684782)
<class 'tuple'>

Coordinate referencing system (CRS) of the shapefile:
 GEOGCS["WGS 84",
    DATUM["WGS_1984",
        SPHEROID["WGS 84",6378137,298.257223563,
            AUTHORITY["EPSG","7030"]],
        AUTHORITY["EPSG","6326"]],
    PRIMEM["Greenwich",0,
        AUTHORITY["EPSG","8901"]],
    UNIT["degree",0.0174532925199433,
        AUTHORITY["EPSG","9122"]],
    AXIS["Latitude",NORTH],
    AXIS["Longitude",EAST],
    AUTHORITY["EPSG","4326"]]
EPSG code:  4326
```

455

```
# Paths to dNBR and dNDVI images
dnbr_image_path = '/content/drive/MyDrive/GY7709/week_7/NBR_Images/DNBR_warped_clipped.tif'
dndvi_image_path = '/content/drive/MyDrive/GY7709/week_7/NBR_Images/dNDVI.tif'
```

```
# Paths to the images to be clipped
images_to_clip = [
    '/content/drive/MyDrive/GY7709/week_7/NBR_Images/DNBR_warped.tif',
    '/content/drive/MyDrive/GY7709/week_7/NBR_Images/dNDVI.tif'
]

# Path to the shapefile defining the extent
extent_shapefile = '/content/drive/MyDrive/GY7709/week_7/Burnshapefile/POLYGON.shp'

# Loop through each image and clip it
for image_path in images_to_clip:
    # Define the output path for the clipped image
    clipped_image = os.path.splitext(image_path)[0] + "_clipped.tif"

    # Clip the image to the shapefile extent using pygge
    pygge.easy_clip(image_path, clipped_image, extent)
```

```
Window coordinates:  Window(col_off=3051, row_off=2543, width=372, height=327)
Window array size:  (1, 327, 372)
Window coordinates:  Window(col_off=1650, row_off=534, width=372, height=327)
Window array size:  (1, 327, 372)
```

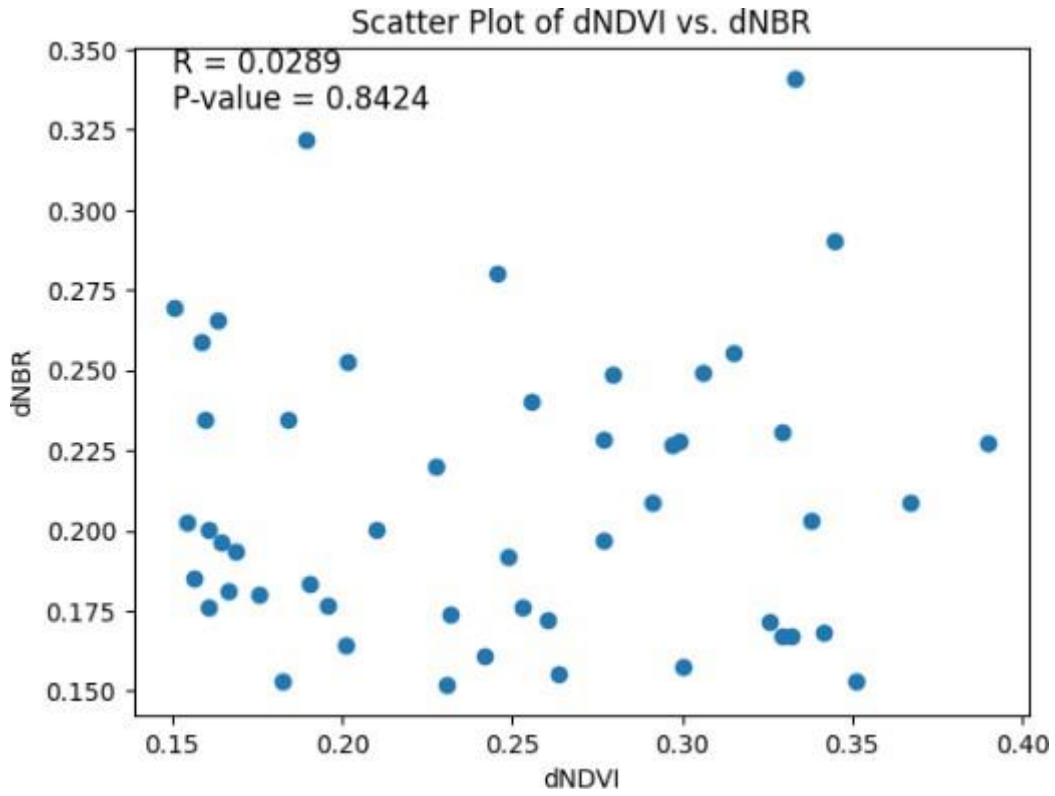
456

```
# Define the range of pixel values
min_value = 0.15
max_value = 0.4

# Paths to the clipped images
clipped_images = [
    '/content/drive/MyDrive/GY7709/week_7/NBR_Images/DNBR_warped_clipped.tif',
    '/content/drive/MyDrive/GY7709/week_7/NBR_Images/dNDVI_clipped.tif'
]

# Create an empty DataFrame to store pixel values
pixel_values_df = pd.DataFrame()

# Extract pixel values from each clipped image within the specified range
for clipped_image in clipped_images:
    with rasterio.open(clipped_image) as src:
        # Read the image data
        image_data = src.read()

        # Flatten the image data array
        flattened_data = image_data.flatten()

        # Filter pixel values within the specified range
        selected_pixel_values = flattened_data[(flattened_data >= min_value) & (flattened_data <= max_value)]

        # Randomly select 100 pixel values from the filtered values
        selected_pixel_values = np.random.choice(selected_pixel_values, size=50, replace=False)

        # Append the selected pixel values to the DataFrame
        pixel_values_df[os.path.basename(clipped_image)] = selected_pixel_values

# Save the DataFrame to an Excel file
output_excel_path = '/content/drive/MyDrive/GY7709/week_7/pixel50.xlsx'
pixel_values_df.to_excel(output_excel_path, index=False)

print("Pixel values saved to:", output_excel_path)
```

457

```
# Load the data from the Excel file
df = pd.read_excel('/content/drive/MyDrive/GY7709/week_7/pixel50.xlsx')

# Extract the dNDVI and dNBR columns
dndvi = df['DNBR_warped_clipped.tif']
dnbr = df['dNDVI_clipped.tif']

# Calculate the correlation coefficient (R-value) and p-value
r, p_value = stats.pearsonr(dndvi, dnbr)

# Create a scatter plot
plt.scatter(dndvi, dnbr)

# Add labels and title
plt.xlabel('dNDVI')
plt.ylabel('dNBR')
plt.title('Scatter Plot of dNDVI vs. dNBR')

# Add annotation for R-value and p-value
plt.text(0.15, 0.34, f'R = {r:.4f}\nP-value = {p_value:.4f}', verticalalignment='center', horizontalalignment='left', fontsize=12)

# Show the plot
plt.show()
```

458


Scatter Plot of dNDVI vs. dNBR
R = 0.0289
P-value = 0.8424
dNBR
dNDVI
0.350
0.325
0.300
0.275
0.250
0.225
0.200
0.175
0.150
0.15
0.20
0.25
0.30
0.35
0.40