National Technical University of Athens
MSc - Data Science and Machine Learning

Geospatial Big Data Analytics

Lab 1
Data retrieval and analysis of timeseries and geospatial products from Business Information Systems and Services

Vasileios Depastas
MSc student

A.M: 03400131
vasileiosdepastas@mail.ntua.gr

April 2022

Contents

List of Figures

List of Tables

1 Introduction

In this report, we experiment with existing business information systems and services that systematicall store all open earth observation data and other geospatial data and products. The data have worldwide coverage and the services aggregate data mainly from NASA/USGS (USA) as well as from European databases and services. There is a list of European Data and Information Access Services (DIAS) that offer data and services:

1. https://creodias.eu/
2. https://sobloo.eu/
3. https://www.wekeo.eu/
4. https://www.onda-dias.eu/cms/
5. https://mundiwebservices.com/

On top of those, Google and Amazon also offer similar services for earth observation data along with powerful high performance information systems.

6. Google Earth Engine https://earthengine.google.com/
7. Amazon https://aws.amazon.com/earth/

Google Earth Engine (GEE) was selected in order to perform a range of tasks that are detailed in this report.

2 Development Steps

Initially, a number of steps were taken in order to build the core code for the report that performs a number of steps including:

1. Registration on GEE
2. Code development on GEE using Javascript
   (a) Landsat 8 (L8) raw data loading
   (b) Polygon selection on the map in a region of Greece with crops
   (c) L8 image collection retrieval for the selected polygon for year 2019 with cloud cover $< 20\%$
   (d) Extraction of image with minimum cloud coverage and visualization. NDVI (normalized difference vegetation index) calculation and map visualization.
   (e) NDVI, EVI, NDWI indexes calculation for all images in L8 image collection
   (f) Maximum NDVI per pixel calculation as well as day-of-year(DOY) calculation this took place. Map visualization of results.
   (g) L8 image collection for the selected polygon with cloud cover $< 20\%$ for all available dates. NDVI calculation and time series visualization. Synthetic time series approximation/fitting.

(h) CART classification for different land usages (water, vegetation, streets, urban) on L8 data.

Finally, utilizing the code developed to answer the above, we do answer on a set of questions.

2.1 Step 1

Initially, we registered on the available services to experiment with the systems offered and the capabilities to retrieve images and data. Upon experimentation, a decision to register and work with GEE was taken. GEE offers a Javascript development environment that provides also a powerful API and a Console to inspect data and variables.
As a first step then, we do insert on GEE the entire set of multispectral data of satellite Landsat 8 (L8).

2.2 Step 2

Next, we draw a polygon in a region of interest within Greece that contains crops. We did select such a region from the map interface of GEE near the city of Larisa in northern Greece, and more specifically in a location with crops west of Larisa's city center.
The selected polygon can be seen in the image below with red colour:

Figure 1: Polygon - region of interest with crops near Larisa city

2.3 Step 3

Subsequently, an image collection (list of images) from satellite Landsat 8 (L8) that contains the selected polygon is retrieved for the year of 2019. Then, the image collection is filtered and images with cloud cover less than 20% are only kept.
We present below the GEE Console with the JSON results of the Javascript code. We can see there are 21 such images for the selected polygon region.

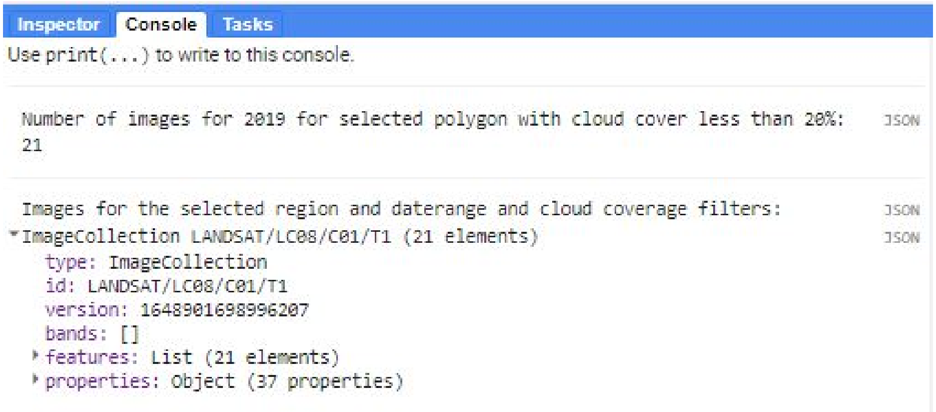

Figure 2: L8 image collection loading and filtering - GEE Console results

2.4 Step 4

Next, out of those 21 images within the filtered image collection, the image with the lowest cloud coverage was selected. As one can see in the console output below, the image was taken on 15-Oct-2019 and the cloud cover was approximately 0.03%, essentially negligible, on that day.
Also, one can extrapolate from the console output, that those L8 images have 12 spectral bands. This information is also available in the description of the dataset.

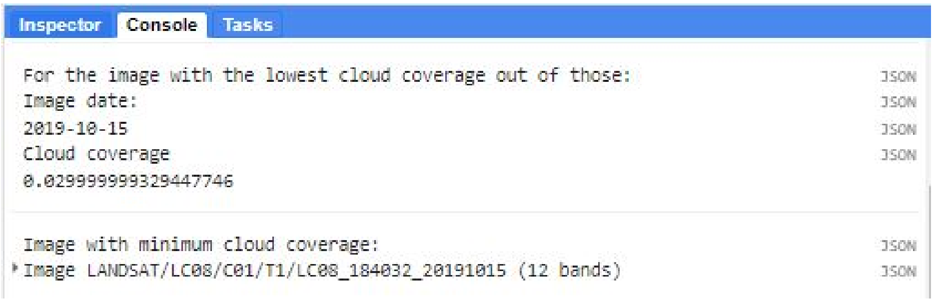

Figure 3: L8 image with lowest cloud cover - GEE Console results

Moving forward, we add an RGB layer on the map for the selected L8 image. Essentially, what we do is map the bands B4, B3 and B2 to colours Red (R), Green (G) and Blue (B) in order to add a natural RGB colour composite layer. In order for the image to look consistent across the different snippets presented in the next chapters, a rectangular was created on the map of the Code Editor and the images are clipped (cropped) to that rectangular that contains the polygon of interest.

Figure 4: L8 image with lowest cloud cover - RGB-432

Also, a false-coloured RGB composite is created with RGB-543. Essentially, this is a multispectral image interpretation using the standard visual RGB band range (red, green, and blue). This false color composite will give all the vegetation a distinct red color, allowing it to be more easily distinguished from its surroundings by the human eye. This is possible due to the high reflectance of plants in the near-infrared spectrum (band 5 is NIR). Additionally, the NIR, Red, Green scheme helps to distinguish clear water (darker shade of blue) from turbid water (cyan) in a false color image.

Figure 5: L8 image with lowest cloud cover - RGB-543

Additionally, the NDVI (normalized difference vegetation index) is calculated for the image and a layer is added on the Code Editor's map for visualization using a color palette from orangered to yellow to green.

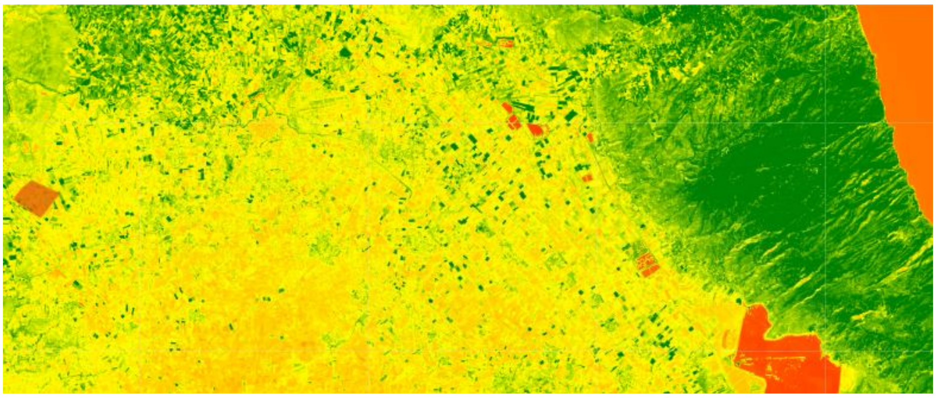

Figure 6: L8 image with lowest cloud cover - NDVI

In the previous figure, pixels with the minimum vegetation are orangered whereas areas with high vegetation are greener. In between, areas with medium vegetation (medium value of NDVI constructed band) are drawn with yellow.

2.5 Step 5

For the Image Collection of Landsat 8 images within 2019 with cloud cover $< 20\%$, the indexes NDVI, EVI and NDWI are calculated for each image and added as extra bands to each image.

NDVI helps to differentiate vegetation from other types of land cover (artificial) and determine its overall state. The index is calculated from the formula:

$$NDVI = \frac{Band5 - Band4}{Band5 + Band4} = \frac{NIR - Red}{NIR + Red} \tag{1}$$

, where NIR is near-infrared light and Red is visible red light.

EVI, Enhanced Vegetation Index, adjusts NDVI results to atmospheric and soil noises, particularly in dense vegetation areas, as well as mitigates saturation in most cases. The value range for EVI is -1 to $+1$, and for healthy vegetation, it varies between 0.2 and 0.8. The formula that gives the EVI is:

$$\begin{aligned} EVI = 2.5 * \frac{Band5 - Band4}{Band5 + 6 * Band4 - 7.5 * Band2 + 1} = \\ = 2.5 * \frac{NIR - Red}{NIR + 6 * Red - 7.5 * Blue + 1} \end{aligned} \tag{2}$$

NDWI, Normalized Difference Water Index, was initially elaborated to outline open water bodies and assess their turbidity, mitigating the reflectance of soil and land vegetation cover. NDWI is retrieved with a near-infrared and visible green band combination. This index is used in detection of flooded agricultural lands, allocation of flooding on the field, detection of irrigated farmland and allocation of wetlands. The formula for NDWI calculation is the following:

$$NDWI = \frac{Band5 – Band6}{Band5 + Band6} = \frac{NIR - SWIR1}{NIR + SWIR1} \tag{3}$$

Below is the output in the GEE Console showing the 21 images of the image collection with the 3 extra bands (NDVI, EVI, NDWI), a total of 15 bands from the initial 12.

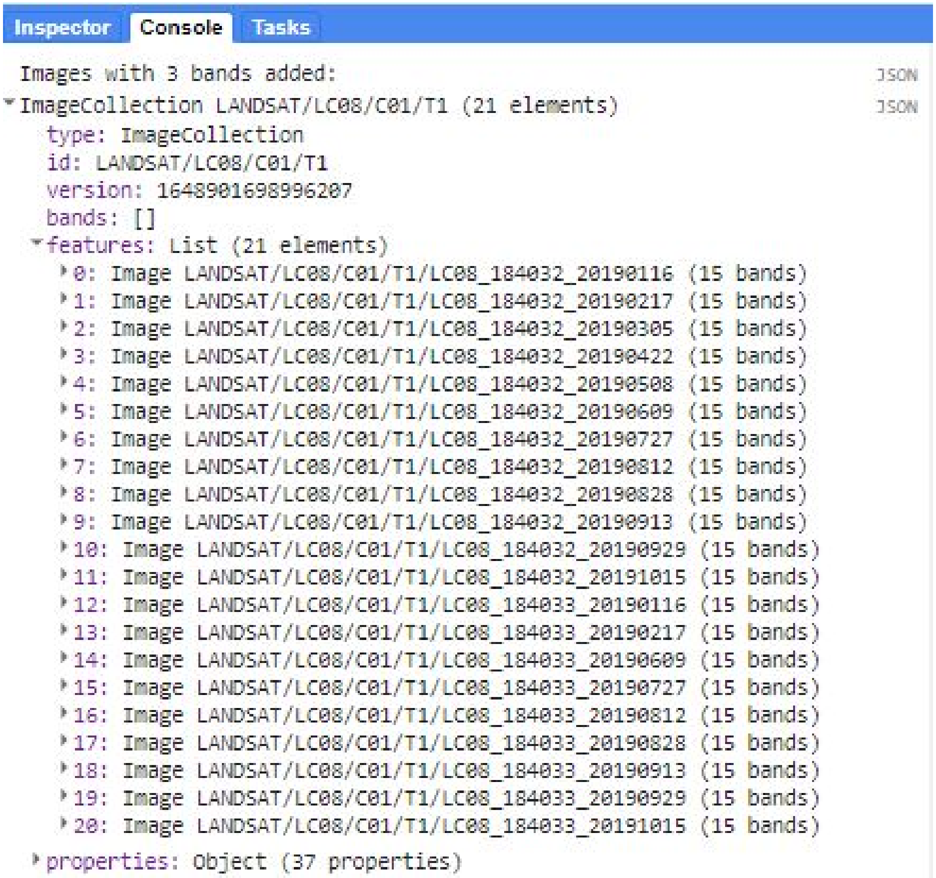

Figure 7: L8 image collection with NDVI, EVI and NDWI bands added

2.6 Step 6

In this section, we find the maximum NDVI index value for each pixel for the year 2019. Then, we also calculate the day-of-year for which this max NDVI value is found. The way we do this is by creating first an extra DOY band (on top of the previously created NDVI, EVI and NDWI bands) for the image collection and then create a mosaic image composed by pixels with the largest NDVI values. This way, each pixel then also has a DOY value that corresponds on the day of the year (value from 1 to 366) it was found.

The bands of the mosaic image are visible in the Google Earthe Engine Console snippet below:

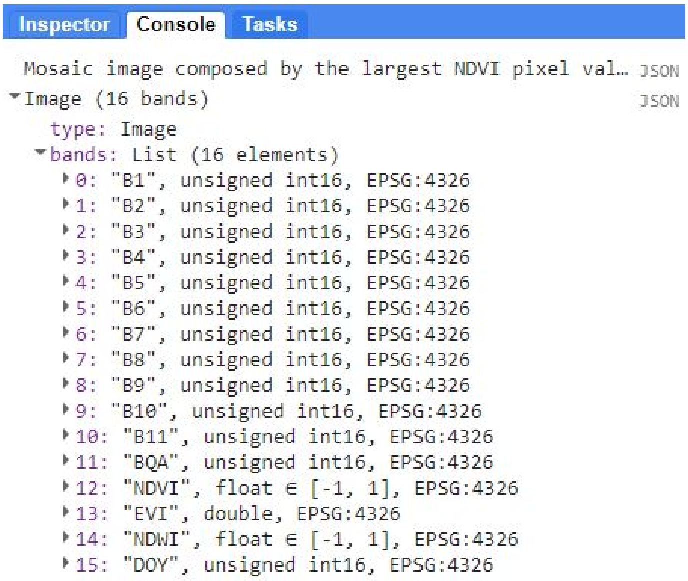

Figure 8: L8 mosaic image with largest NDVI pixel values in 2019 - Console

Below we present the resulting map layer for the NDVI band in an [orangered, yellow, green] color palette.

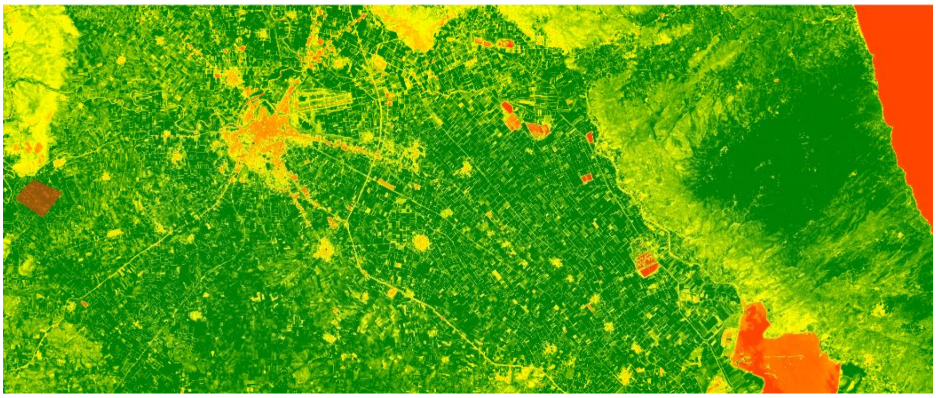

Figure 9: L8 mosaic image with largest NDVI pixel values in 2019 - NDVI band

We can see that water (aegean sea and a few lakes) is colored with red, the city of Larisa is colored with yellow-orange and most of the other space is towards green (crops, forest on mountains etc.) as we would expect since those are the highest NDVI pixel values within 2019.

Also, below is the map layer for the DOY band in an [yellow, lightgreen, green, red] color palette.

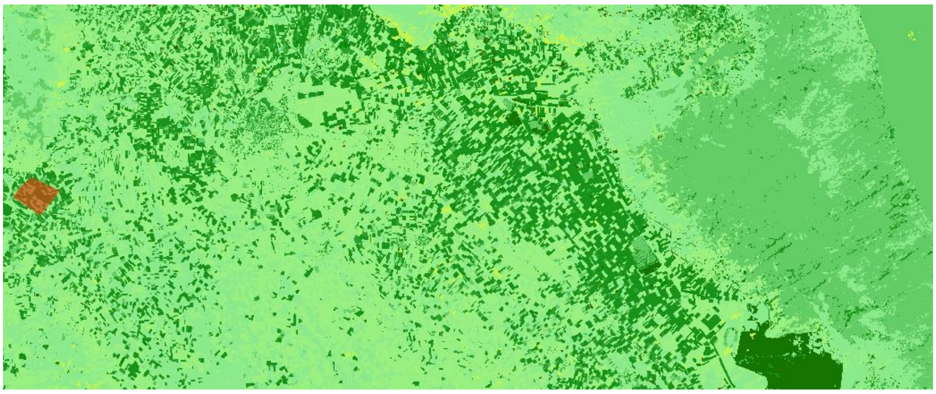

Figure 10: L8 mosaic image with largest NDVI pixel values in 2019 - DOY band

On the above map we can see that the colors are more concentrated in lightgreen and green colors that designate days of year mostly in the Spring and Summer period. This is something that we would expect, since there is normally more vegetation (higher NDVI index values) during the spring and summer months of the year.

2.7 Step 7

Next, we create an image collection with all available Landsat 8 images containing the selected polygon until the end of April 2022 and with cloud cover less than 20%. We also add an NDVI index band to the image collection. This collection has a total of 177 images.

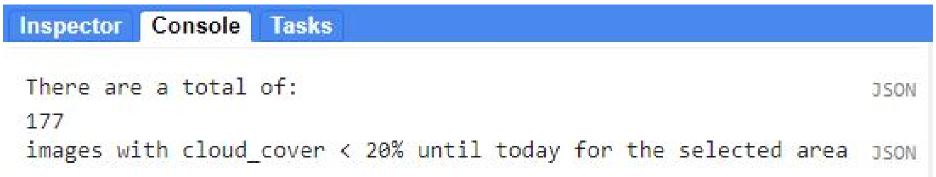

Figure 11: Alltime L8 image collection - Console

Given that those 177 images are satellite images since 2013 it becomes obvious that there are selected days only with data. This makes sense, since the satellite only generates image data for the selected polygon region every time it passes over that area. The date of every image observation is included along the x-axis, while the mean value of each image's NDVI pixel values defines the y-axis.

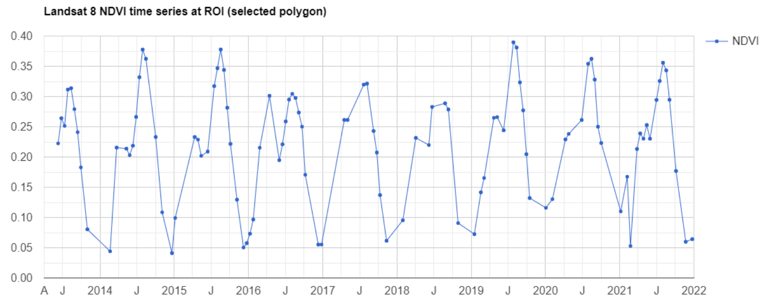

Figure 12: Alltime L8 image collection - NDVI time series

In order to fill in the values between the different observation/points we can apply a regression method.

First, we experiment with Linear Regression, although we observe on the time series that there is some periodicity on the NDVI mean values. More specifically, we observe peaks normally around the middle of the year (spring and summer months) and low values around autumn and winter months.

For Linear Regression, we first assume that the dependent variable y (mean value of the NDVI value of the image's pixels) is linearly dependent on the independent variable t (date).

$$y = y(t) = b_0 + b_1 \cdot t + \epsilon_t \tag{4}$$

, where ϵ_t is a random error that is assumed to follow normal distribution with mean 0 and constant variance. Therefore, for the expected value of y we get:

$$E[y] = b_0 + b_1 \cdot t \tag{5}$$

and the goal is to calculate the values of the coefficients b_0 and b_1 that define a line $\hat{y} = \hat{b_0} + \hat{b_1} \cdot t$ which minimizes the sum of square error (SSE):

$$SSE = \sum_{i=1}^{n}(y_i - \hat{y_i})^2 = \sum_{i=1}^{n}(y_i - (\hat{b_0} + \hat{b_1} \cdot t_i))^2 \tag{6}$$

where y_i is the known value of the mean NDVI pixel value and t_i is the corresponding date. The data we have available in the image collection dataset is actually (t_i, y_i) pairs, and minimizing the SSE gives the finally fitted line $\hat{y}$.

In the image below, we present the linear regression fitted line:

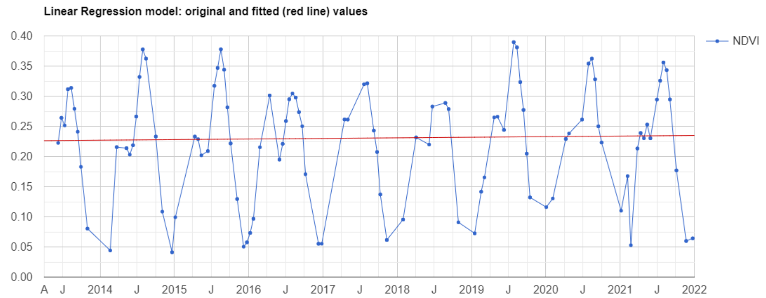

Figure 13: NDVI time series Linear Regression

As expected, the fitted line does not fit well the relationship of the available data. Hence, we will experiment with another regression model that has some periodic functionality. The harmonic regression can be formalized by the following equation:

$$y = y(t) = b_0 + b_1 \cdot t + b_2 \cdot \cos(2\pi\omega t) + b_3 \cdot \sin(2\pi\omega t) + \epsilon_t \tag{7}$$

By fitting the harmonic curve to the data, we get the following fitted curve:

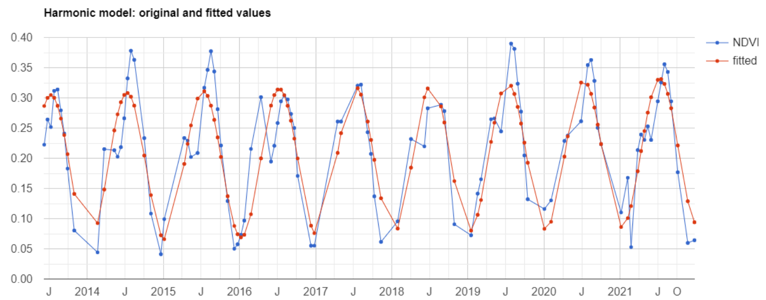

Figure 14: NDVI time series Harmonic Regression

The fit to the data seems to be much better compared to linear regression since we have inserted periodicity to the fitted curve. One observation we can make is that the amplitude of the curve is lower (in absolute value) compared to the real data resulting in lower (in absolute value) peaks.

2.8 Step 8

In this step, we do select a new polygon again in the region in and around the city of Larisa. This time, the polygon is much larger containing areas with different use of land (water, crops, streets, urban).

Here is the polygon area selected:

Figure 15: Polygon 2 - Larisa wider area

Next, we define 4 classes (water, crops/vegetation, streets, urban) for different land usage and we create sample points by adding appropriate markers in the map of the GEE editor. We have included points from different areas of Greece in order to provide a more wide description of the land usage for each category/class and have sufficient number of sample points too.

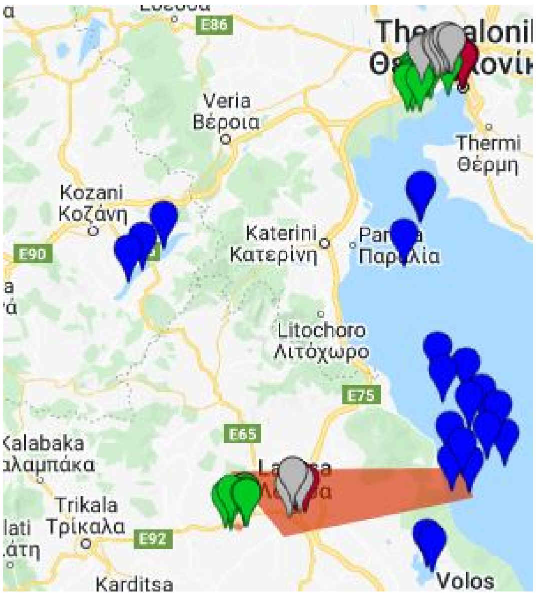

Figure 16: Sample points - 4 classes

This is the dataset that we split to training and test. We have selected the training set to contain 80% of the sample points and the remaining 20% is the test set. Then, we feed te training data points to a CART classifier

(Classification And Regression Tree). Subsequently, we evaluate the classifier accuracy (found to be approximately 90.5%) on the test set and based on the learned classes we classify the land usage in the selected polygon.

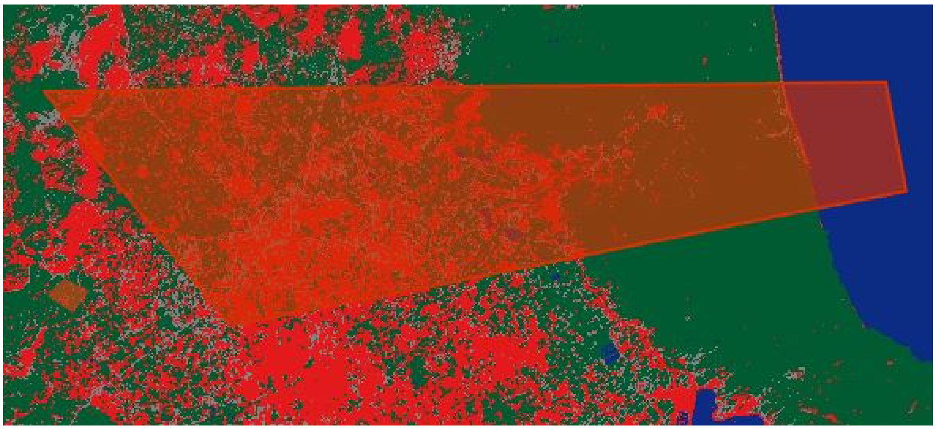

Figure 17: Classification - selected polygon

With color red is urban area, gray is streets, green is crop land or vegetation/forest and blue is water

Finally, we also present the confusion matrix for the classification performed.

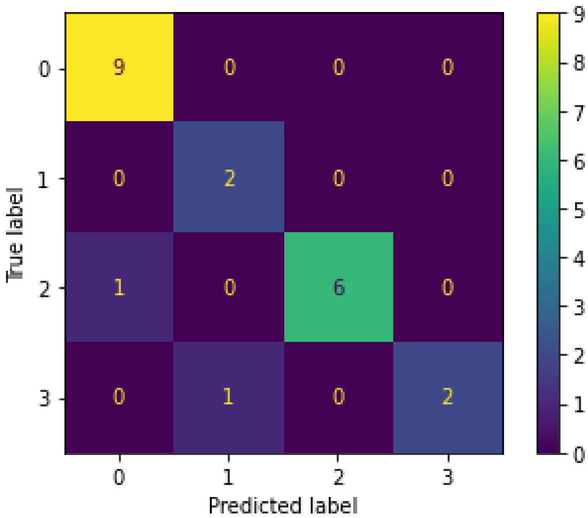

Figure 18: Classification - confusion matrix

3 Questions

In this section, we answer to the main questions of the exercise.

3.1 Step 1

1. Creodias

   Creodias is a platform that provides direct access to Earth Observation satellite data that also offers user friendly tools for cloud processing of the data. The platform contains the EO browser that offers the ability to visualize geospatial data. Also, there is a search capability for data as well as a web dashboard for managing available cloud computing resources. Some of the offered satelitte data include data from Landsat 5, Landsat 7 and Landsat 8.

2. Sobloo

   Sobloo is a Data and Information Access Service (DIAS) platform of Copernicus.

   It's essentially a one-stop shop for geospatial data. It provides under the same roof, business tools, integration through the generation of APIs and connectors, and cross-references of data sources or the AI associated with these tools to enrich raw data.

3. Wekeo

   WEkEO is the service for marine environmental data, virtual environments for data processing, and skilled user support.

   It is one of the Copernicus DIAS (Data and Information Access Services). It has public free service for discovery and access to data, and a commercial one with various analysis applications and cloud space.

   The service inlcudes an interactive map for data search and visualization from users.

4. Mundi web services

   Mundi Web Services is a platform on which users can directly use preconfigured geoservices. However, it also offers new geoservice providers the option of offering their services on the market. It contains all necessary components to implement new geo-based business ideas. This includes constantly updated data from the Sentinel satellites as well as historical data from the American Landsat program, geoservice applications, and flexible IT resources.

5. Google Earth Engine

   Google Earth Engine (GEE) offers users the ability to write Javascript code and interact directly with the GEE API in order to efficiently process geospatial data as well as visualize them in a Console and an interactive map.

   Google earth engine is available for free upon student registration and is very easy to share code and applications developed promoting collaboration and mutual learning.

3.2 Step 2

In 2.4 we already presented a natural RGB colour composite layer, a false RGB colour composite for the satellite image of 2019 with the lowest cloud cover as well as another layer with the NDVI value of pixels for that image.

3.3 Step 3

For a region of interest we generate four map layers. There are two images for 2019, one showing the NDVI band for a mosaic composite image with the largest NDVI pixel values and another one displaying the DOY band. Then this is repeated for 2018 in order to compare the two years.

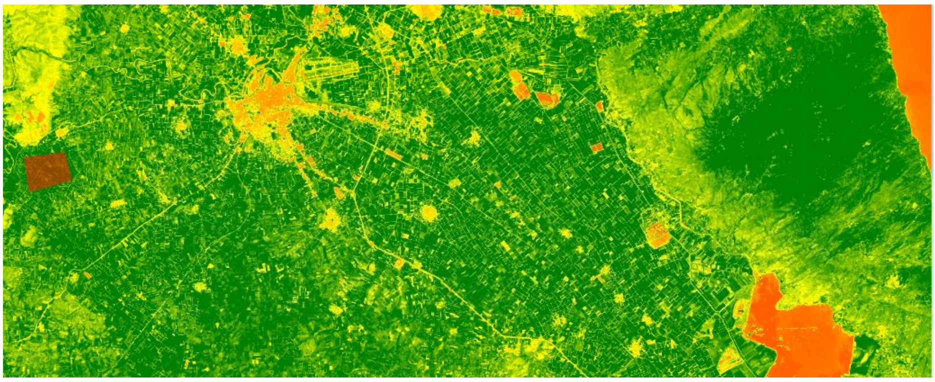

Figure 19: 2019 - max NDVI mosaic image - NDVI band

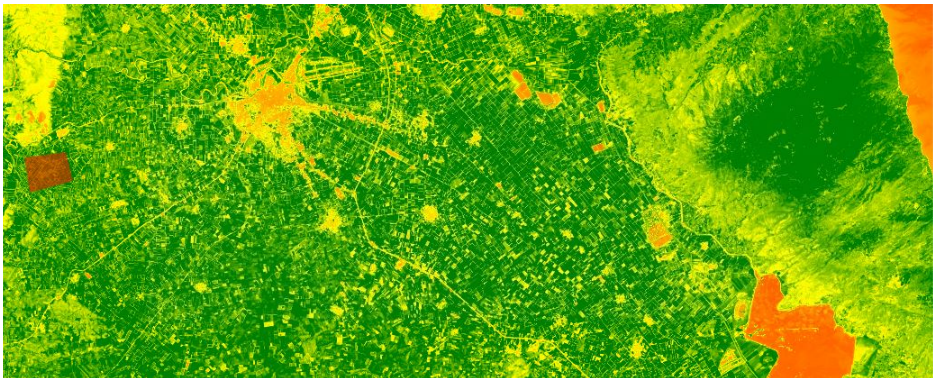

Figure 20: 2018- max NDVI mosaic image - NDVI band

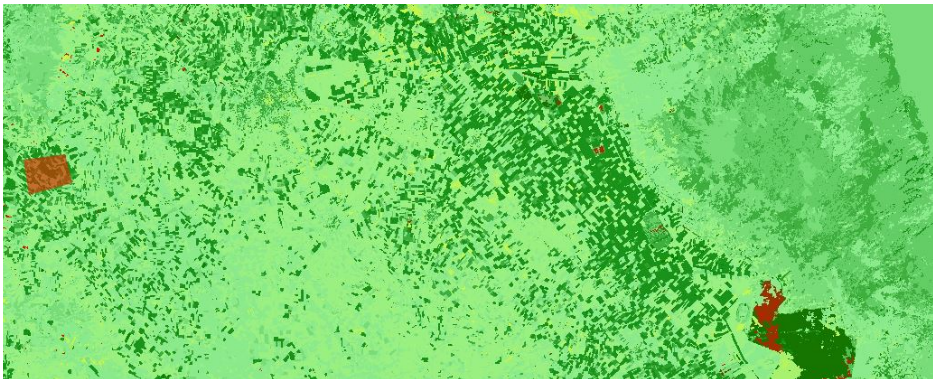

Figure 21: 2019 - max NDVI mosaic image - DOY band

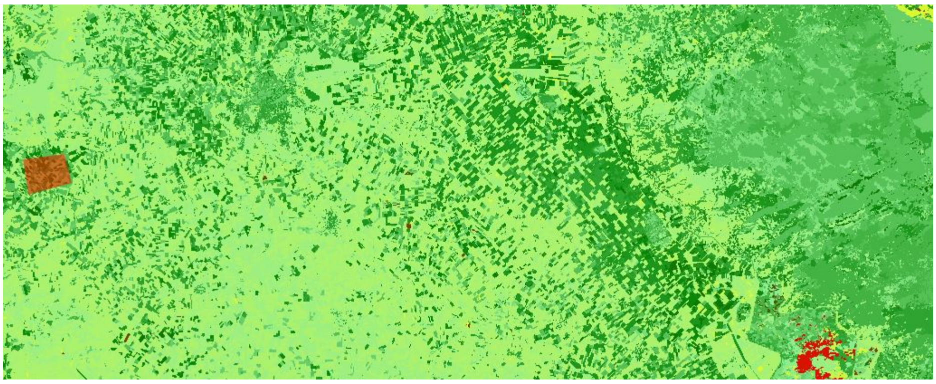

Figure 22: 2018 - max NDVI mosaic image - DOY band

On the figures with the NDVI band, we can observe that 2018 was more yellowish than green compared to 2019. This observation along with the color palette used for the map layer, [orangered, yellow, green], pinpoints to the conclusion that 2019 was a better year in terms of vegetation for the Larisa city and its surroundings. This might be due to better climate conditions (eg more rainy year) that are affecting vegetation.

Similarly comparing the two figures for the DOY band for 2018 and 2019, we can conclude that 2018 is more yellowish/lightgreen than 2019 and based on the color palette used, [yellow, lightgreen, green, red], this is interpreted for 2018 as having peak NDVI values for its pixels earlier in the year than in 2019 and more towards spring compared to 2019.

3.4 Step 4

We now draw 4 different polygons around Larisa city and also get all available L8 images with cloud cover less than 20%. We then add an NDVI band to the images of the returned image collection and we draw the time series of the mean pixel NDVI values of the 4 polygons along with the fitted harmonic regression curve.

Figure 23: 4 polygon regions near Larisa city

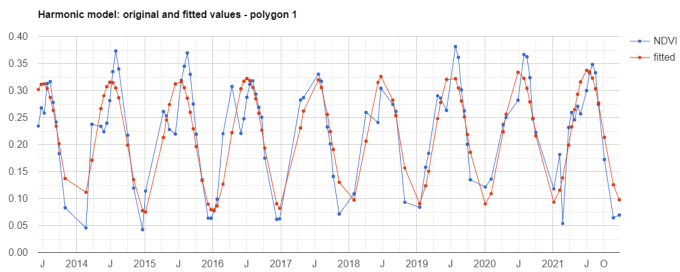

Figure 24: Polygon 1 - Time series and fitted harmonic regression

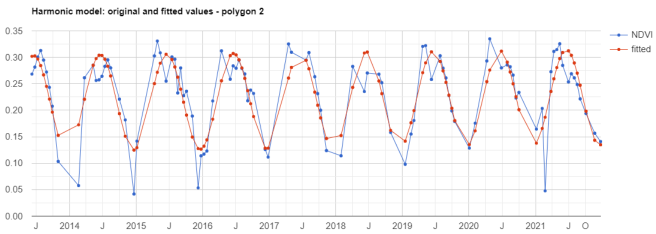

Figure 25: Polygon 2 - Time series and fitted harmonic regression

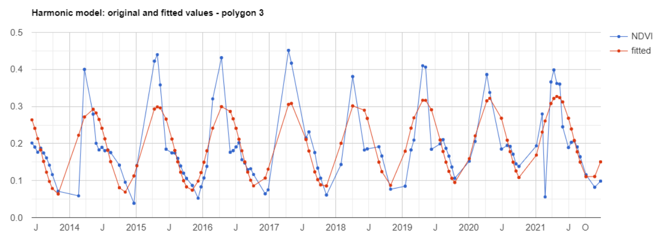

Figure 26: Polygon 3 - Time series and fitted harmonic regression

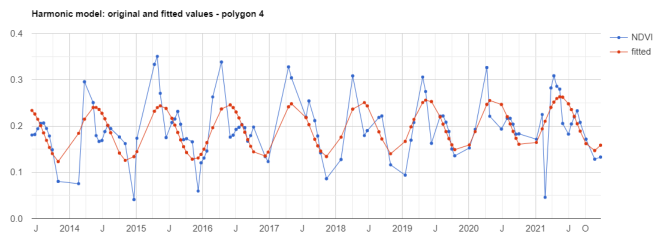

Figure 27: Polygon 4 - Time series and fitted harmonic regression

We can see that for the polygon 1 that is a pure crops region on the map, the curve seems to fit better on the data since they are more predictable presumably. This also applies on polygon 2 with some exceptions only. However, for polygons 3 and 4 that contain a mix of urban and crop lands, the real data seems to have spikes and not a very smooth periodic performance, hence the fit of the harmonic regression cannot follow the spikes of the data.

3.5 Step 5

We are now looking to classify different land areas based on their usage. Here we make a selection for three types of usage, water (class 0), urban (class 1), and vegetation (crops, forest areas etc - class 2). To classify the areas we select 40 points (markers) for each usage/class out of the three used. From those 80% are kept randomly as a training set and another 20% as a test set. For the classification task, two different classifiers, a CART and an SVM, are used.

In the image below, the selected sample points for the three classes can be seen. With red are markers in urban areas, with green areas with vegetation and with blue areas with water.

Figure 28: Classification - dataset sample points

Training the two models based on the values of all bands and the training set, we are able to classify the selected region and visualize it adding a map layer for each classifier.

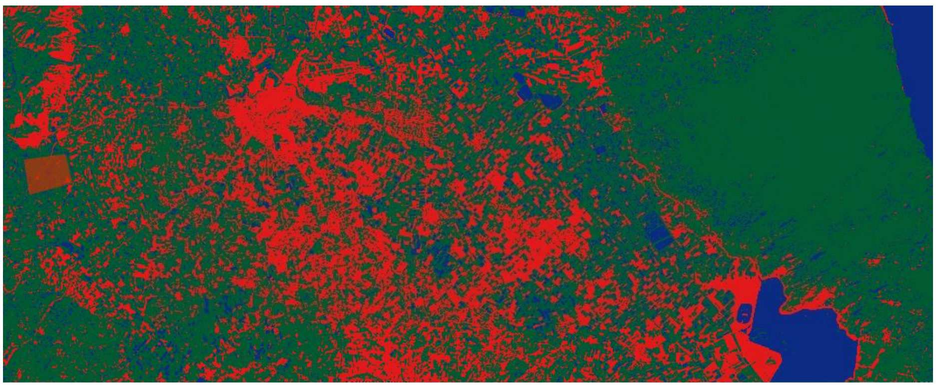

Figure 29: Classification - CART classification result around Larisa

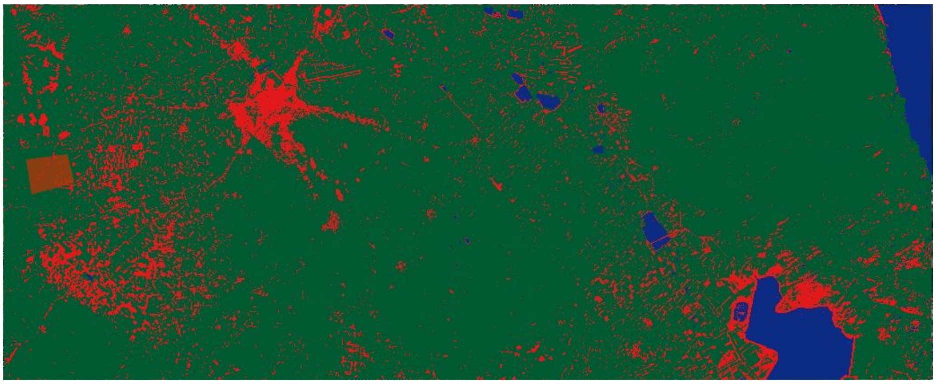

Figure 30: Classification - SVM classification result around Larisa

We can clearly see that CART classifies more regions as urban that the SVM classifies them as regions with vegetation. Both can efficiently distinguish water. Based on the satellite map, it seems that those regions outside the Larisa city are vegetation areas in the majority of them. Hence, we expect the SVM classifier to achieve better accuracy for the classification task at hand.

Indeed, printing the accuracy of the two classifiers in the Console of GEE we get:

Classifier	Accuracy
CART	85.7%
SVM	100%

Table 1: Classification accuracy

Also, we present the confusion matrices that are printed in the GEE Console:

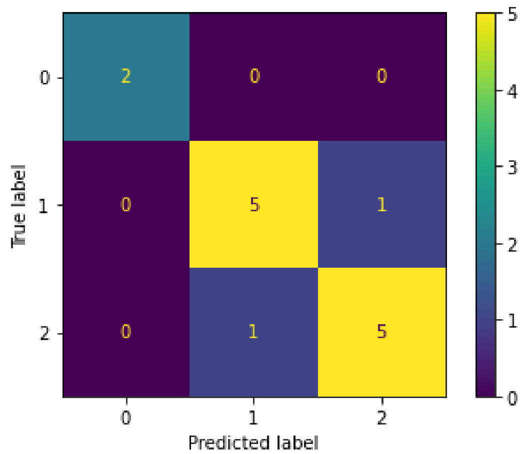

Figure 31: CART classification confusion matrix

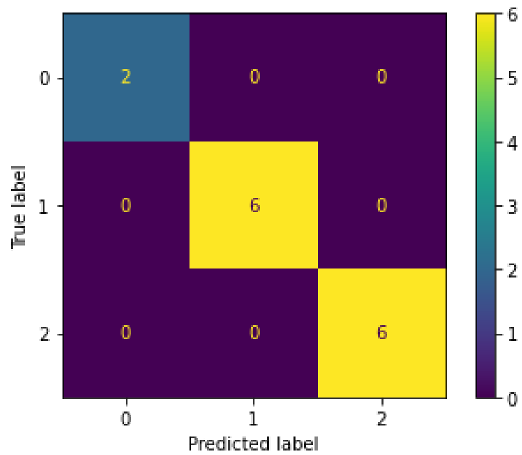

Figure 32: SVM classification confusion matrix

It is obvious from the confusion matrix of SVM, that there are no misclassified sample points from the test set. This is why the accuracy of the SVM classifier is 100%. However, we should note that the dataset size is quite small and hence we would need to get more points/markers for both training and test sets in order to get more accurate results. Also, we can see on the CART confusion matrix that the classifier mixes up the classes 1 and 2, meaning urban and vegetation areas, which is what we assumed before based on our observation of the map layer.

4 Code

Finally, below we share the code used to produce this report.

1. Development Steps Code

2. Questions Code
3. GEE app code
4. GEE app