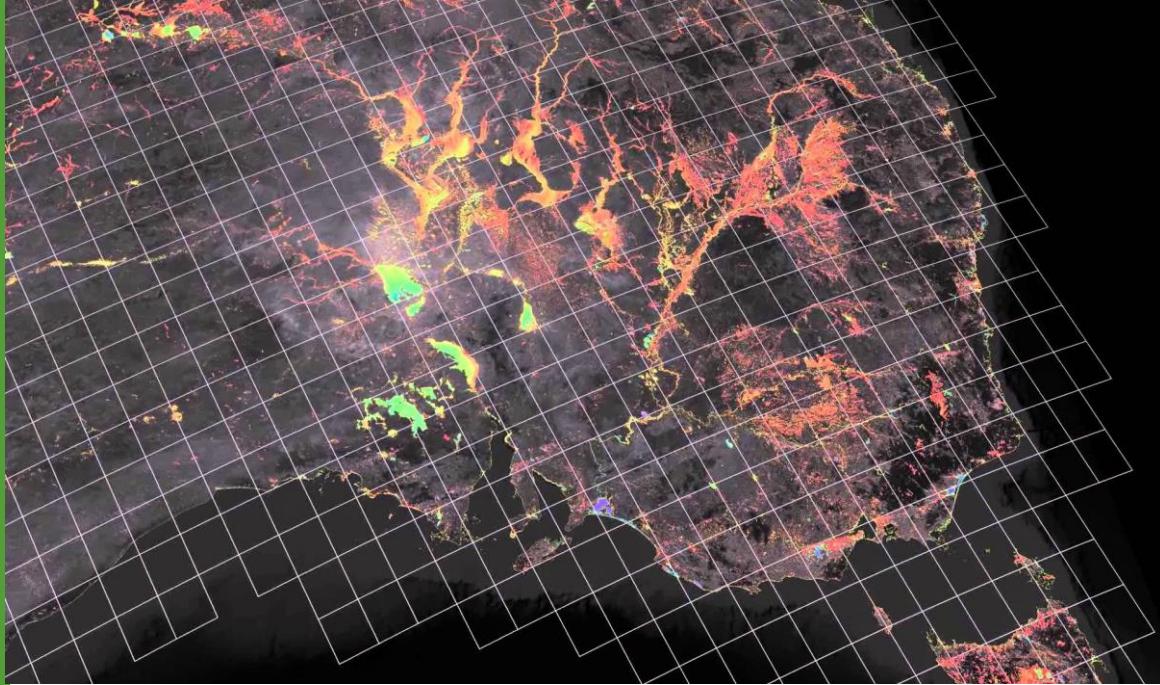


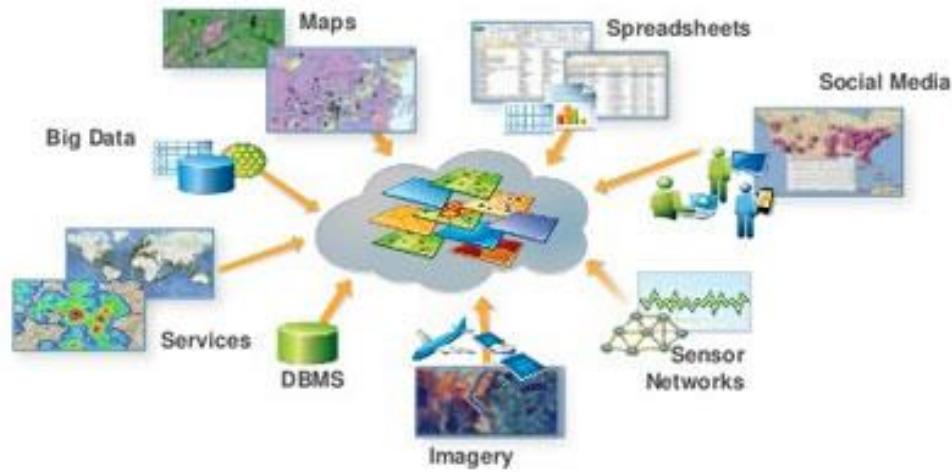
Tratamento de Dados Geospaciais em Plataforma de Nuvem

César Iván Alvarez
calvarezm@ups.edu.ec

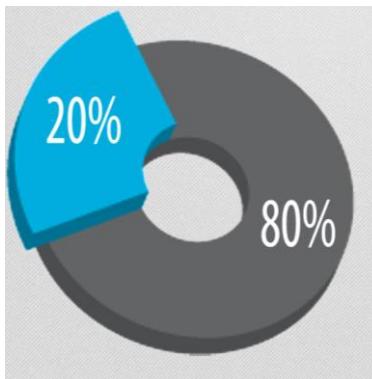


Dados Geoespaciais

Dados geoespaciais, ou geodados, são dados que contêm informações relacionadas a locais na superfície terrestre. É possível mapear objetos, eventos e outros fenômenos do mundo real de uma área geográfica específica identificada por coordenadas de latitude e longitude.

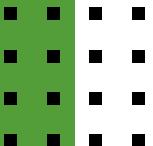


**Você sabia que mais de
80% da informação
mundial é geográfica?**



Tipos de dados geoespaciais

- 01 VECTOR**
Vertices and paths as points, lines and polygons.
- 02 RASTER**
Raster data is made up of pixels or grid cells.
- 03 DATABASES**
Geographic databases store vectors and rasters.
- 04 WEB**
Data built to serve and display geographic features over the internet.
- 05 MULTITEMPORAL**
Multitemporal geodata has a component of location and time.

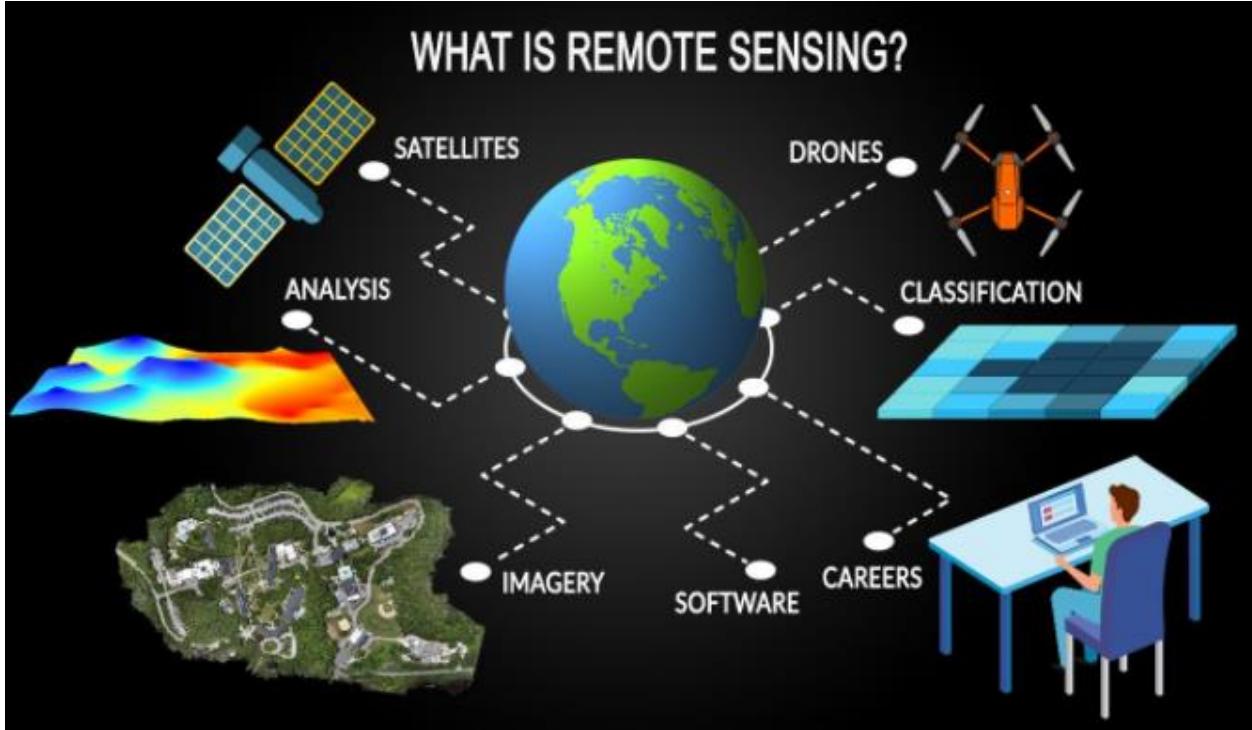




***Quem gerencia a informação geográfica consegue
organizar o território de forma mais eficaz***



Conceitos de Sensoriamento Remoto



Também conhecido como Sensoriamento Remoto, é um método de obtenção de informações sobre as propriedades de um objeto sem contato físico com ele.

Componentes do Sensoriamento Remoto



UAV-based remote sensing:
Height: < 3 km
Coverage: 0.1~100 km^2

Aerial remote sensing:
Height: < 30 km
Coverage: 10~100 km^2

Satellite remote sensing:
Height: > 150 km
Coverage: 10~1000 km^2

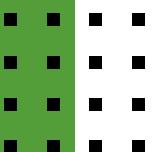


O espectro eletromagnético

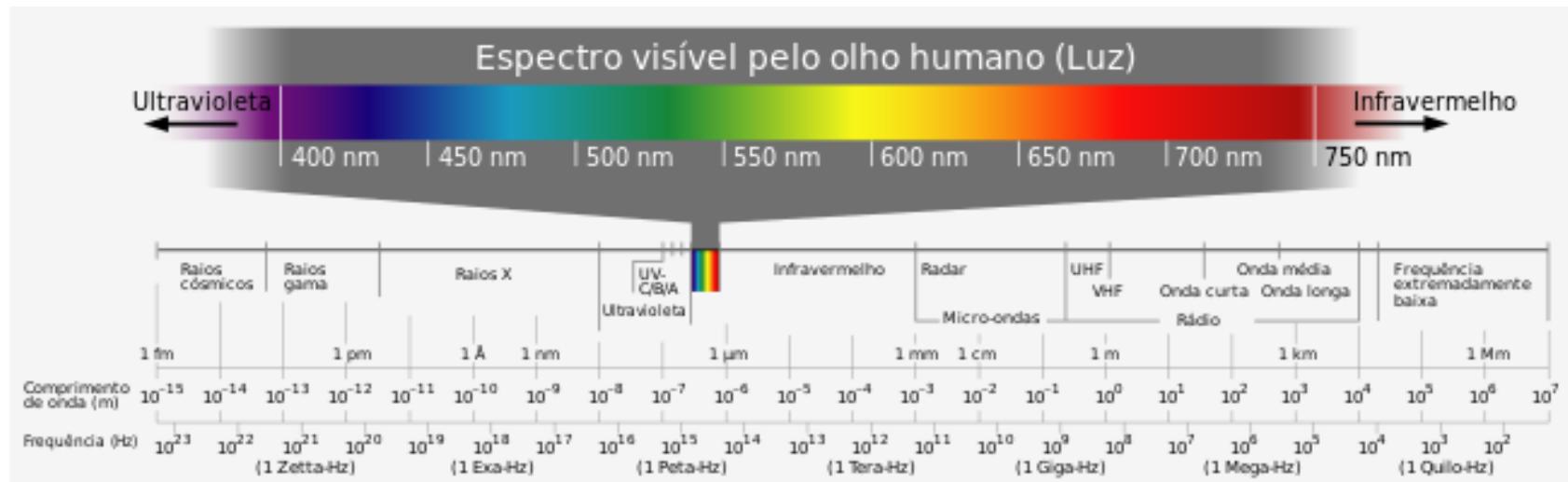
A radiação eletromagnética comprehende uma ampla variedade de frequências ou comprimentos de onda que vão desde raios gama até ondas de rádio. Todas estas emissões constituem assim o espectro electromagnético.

As radiações mais utilizadas em sensoriamento remoto são:

- As microondas são usadas em sensores de radar.
- Radiação infravermelha radiação emitida por corpos quentes.
- O espectro visível (RGB)
- A radiação ultravioleta é o principal componente da radiação solar.



O espectro eletromagnético



Tipos de Sensores



ACTIVE SENSING



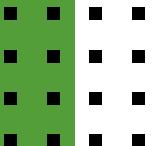
Remote Sensor emitting an energy source and detecting response off crop

Energy Source

PASSIVE SENSING

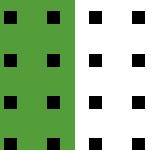


Remote Sensor collecting Red, Green, Blue, NIR spectrum



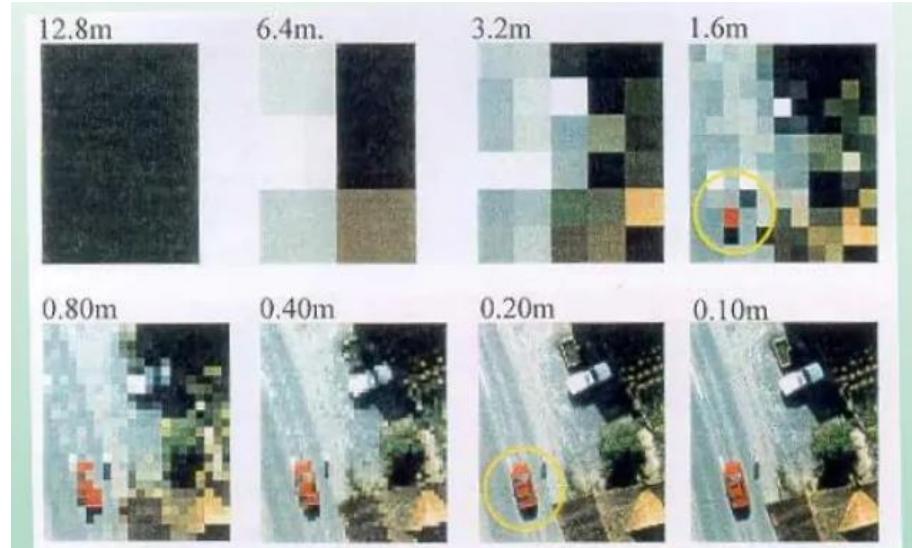
Tipos de resoluções

- Resolução Espacial - Geográfica
- Resolução Espectral
- Resolução Radiométrica
- Resolução temporária



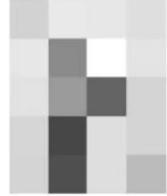
Resolução Espacial - Geográfica

Tamanho que a área abarcada em um pixel corresponde na realidade.

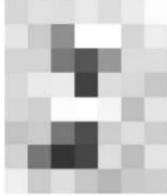


Resolução Espacial - Geográfica

1.6m



0.80m



0.40m



0.20m



0.10m



0.05m



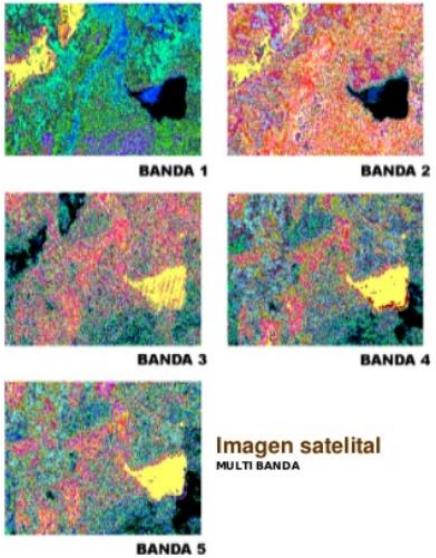
0.03m



0.01m



Resolução Espectral



Resolución Espectral

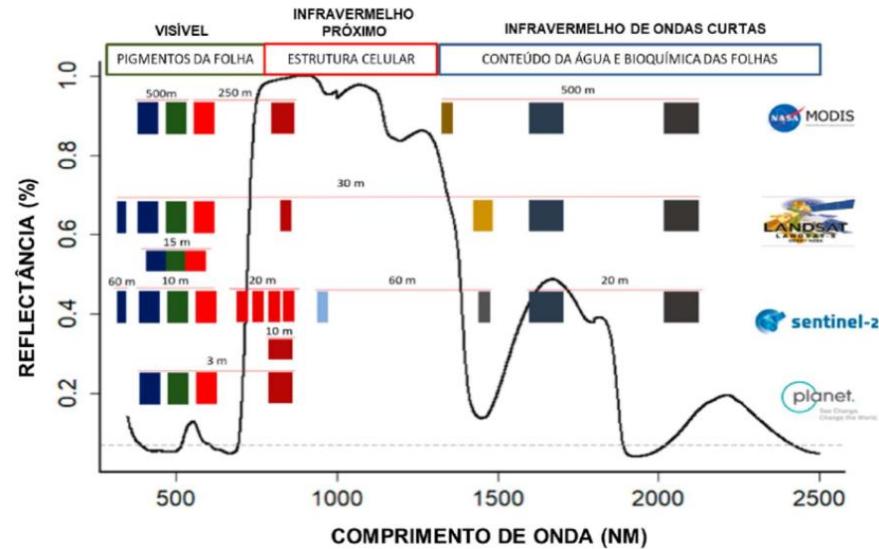
Cantidad de Bandas del sensor



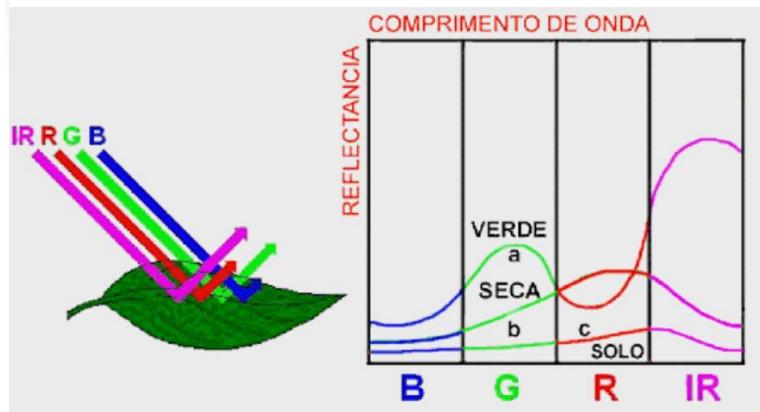
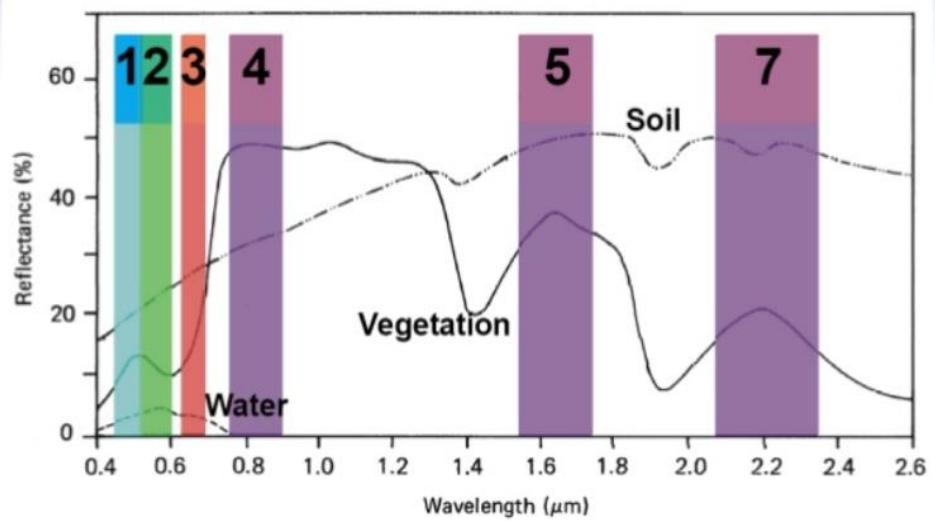
aerofotografía
MONOBANDA 1 banda)



Imagen satelital
MULTI BANDA

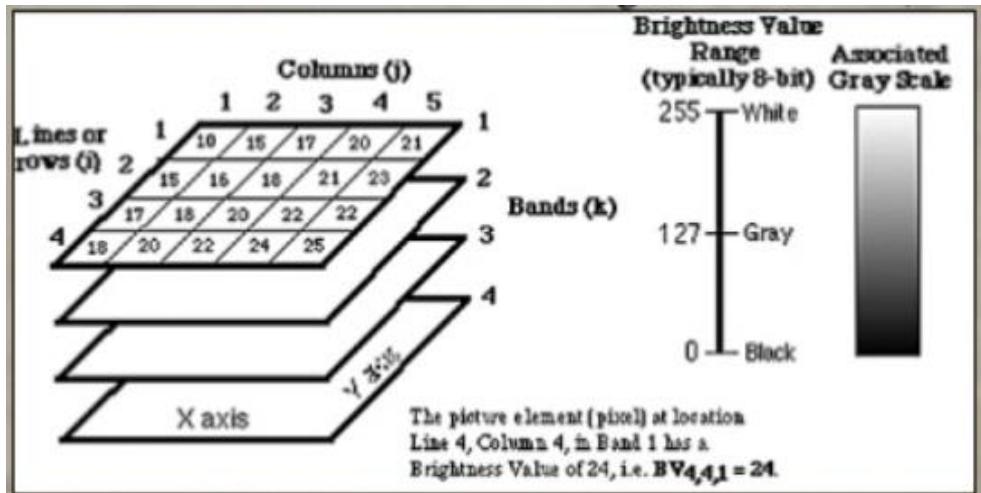


Resolução Espectral - Assinaturas Espectrais

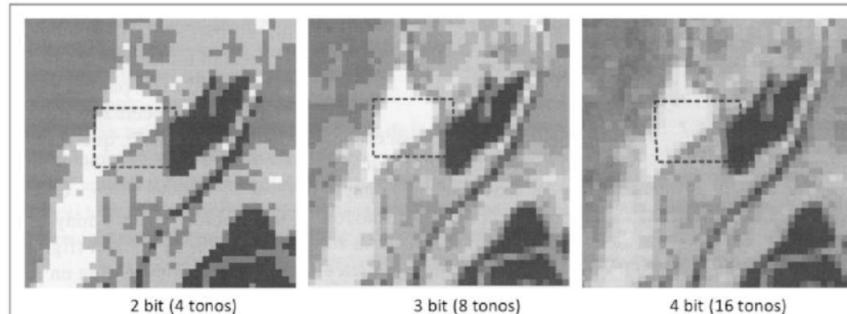


Resolução Radiométrica

Quanto maior o número de níveis que podem ser registrados, maior será a resolução radiométrica do sistema de sensores 8, 12, 16, 24, entre outros.

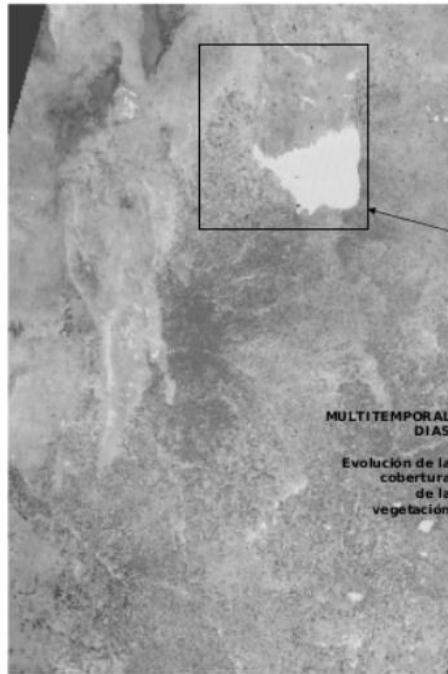
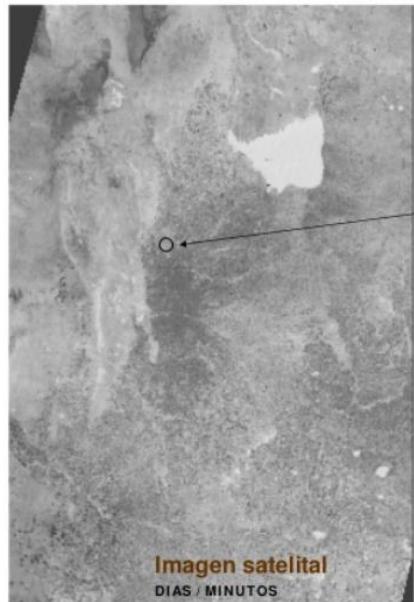


Resolução Radiométrica



Fuente: Adaptado de Chuvieco , 2002

Resolução temporária



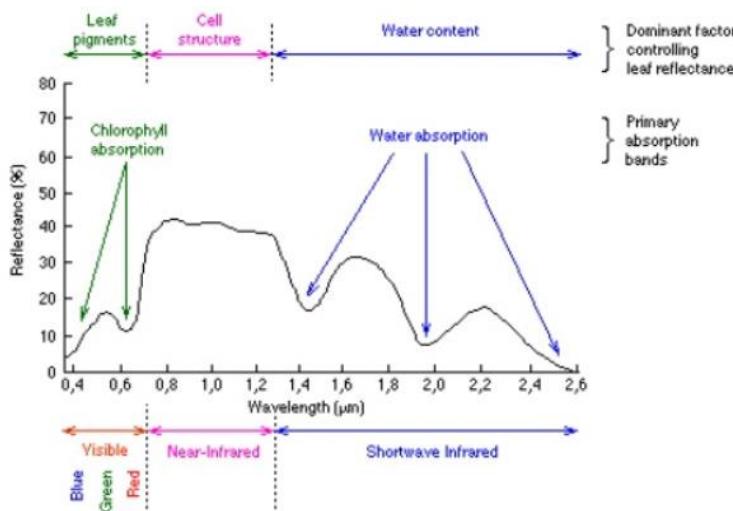
Intervalo de tempo que o satélite leva para voltar a recobrir a área de interesse

Índices multiespectrais

Existe um grande número de índices espectrais que podem analisar diversos aspectos como vegetação, recursos hídricos, neve, solo, fogo, entre outros. Ou seja, destacamos o elemento de interesse na superfície terrestre.

Radiância é o fluxo radiante emitido, refletido, transmitido ou recebido por uma determinada superfície, por unidade de ângulo sólido por unidade de área projetada.

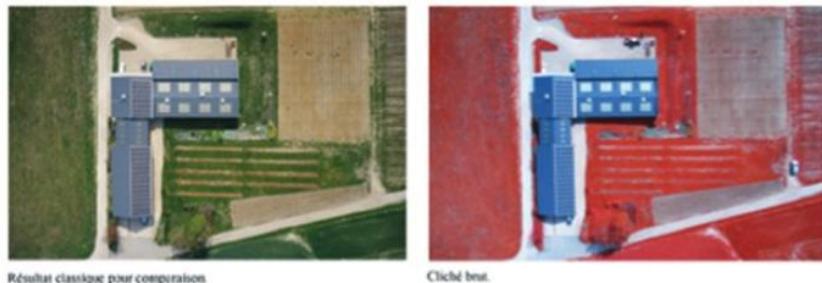
A refletância é a razão entre a radiância refletida e a produção total de energia.



NDVI

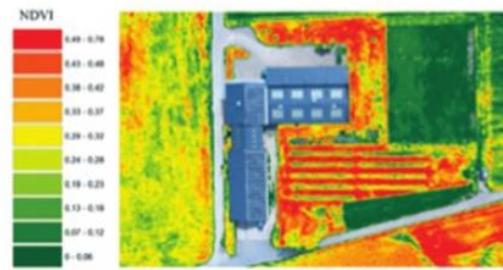
O Índice de Vegetação por Diferença Normalizada (NDVI) é um índice de vegetação utilizado para estimar a quantidade, qualidade e desenvolvimento da vegetação com base na medição da intensidade de radiação de determinadas bandas do espectro eletromagnético.

$$NDVI = \frac{NIR-RED}{NIR+RED}$$

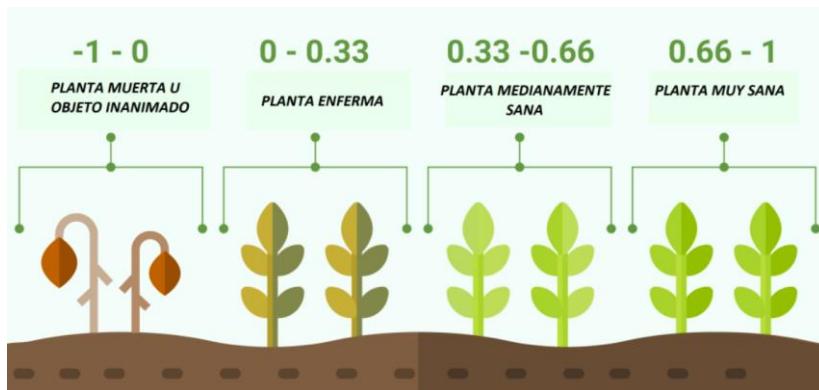
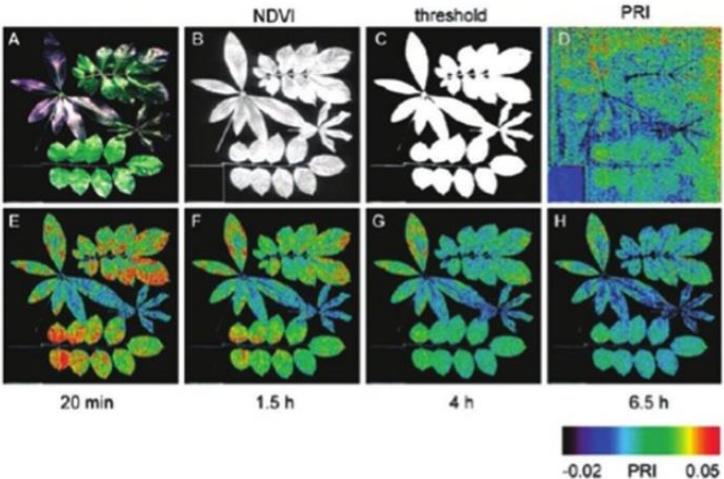
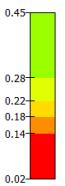
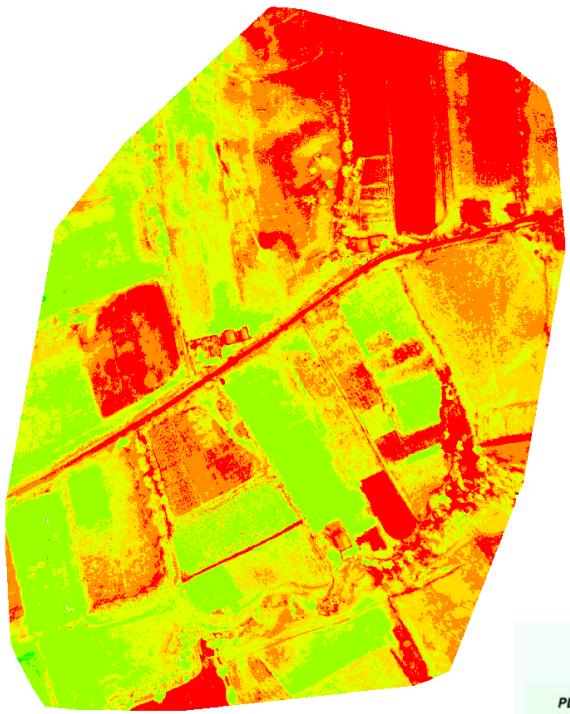


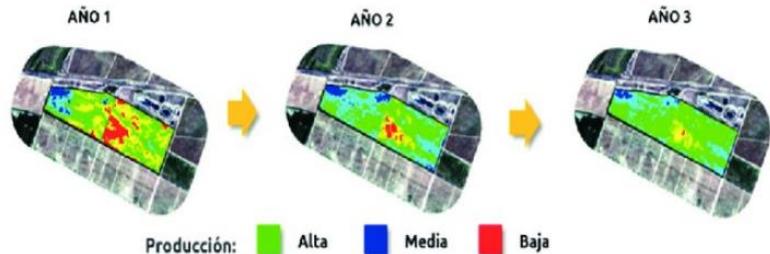
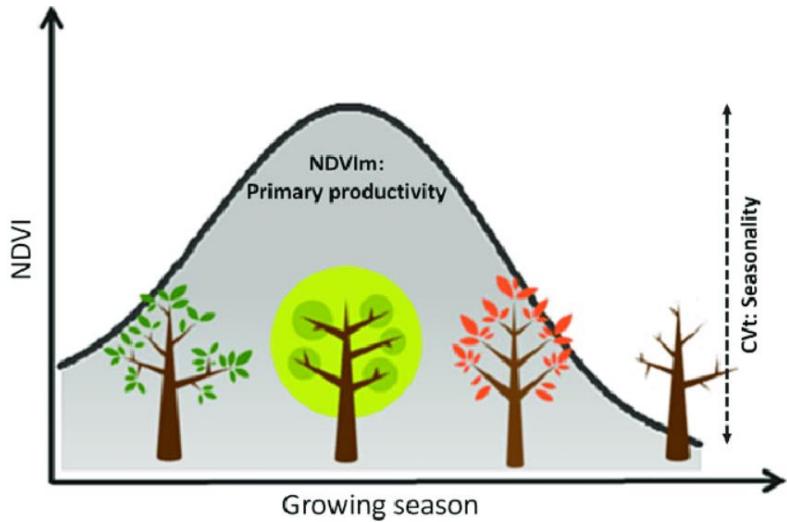
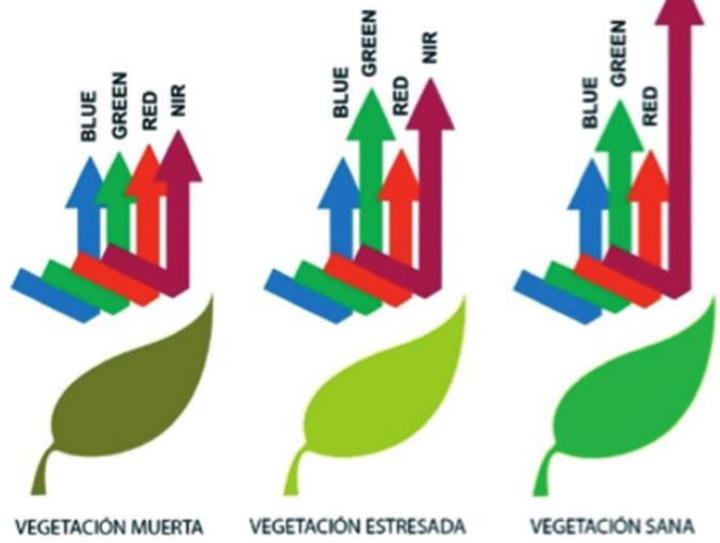
Résultat classique pour comparaison.

Cliché brut.



Carte d'activité végétale NDVI obtenue à partir du cliché brut.



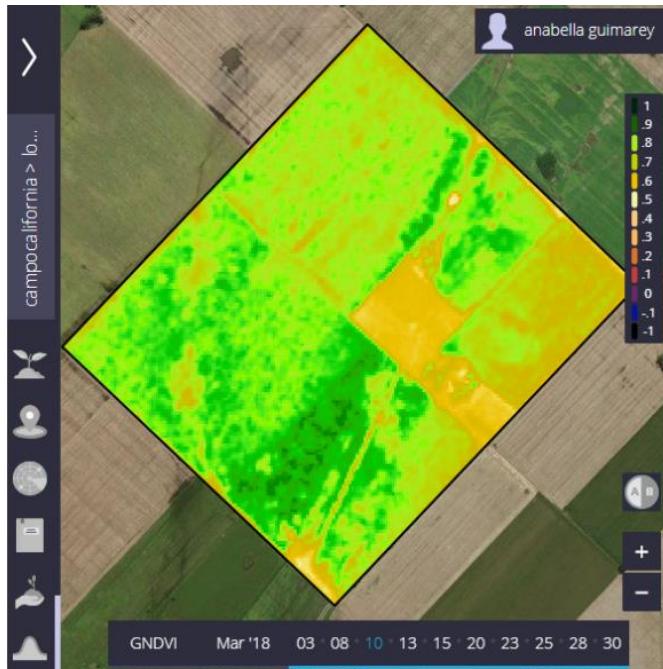


Producción: ■ Alta ■ Media ■ Baja

GNDVI

O Índice GNDVI (Green Normalized Difference Vegetation) é um índice de “verdura” da planta ou atividade fotossintética. É um dos índices de vegetação mais utilizados para determinar a absorção de água e nitrogênio na copa das culturas.

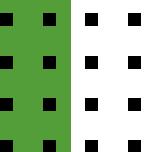
$$GNDVI = \frac{R_{NIR} - R_{Green}}{R_{NIR} + R_{Green}}$$



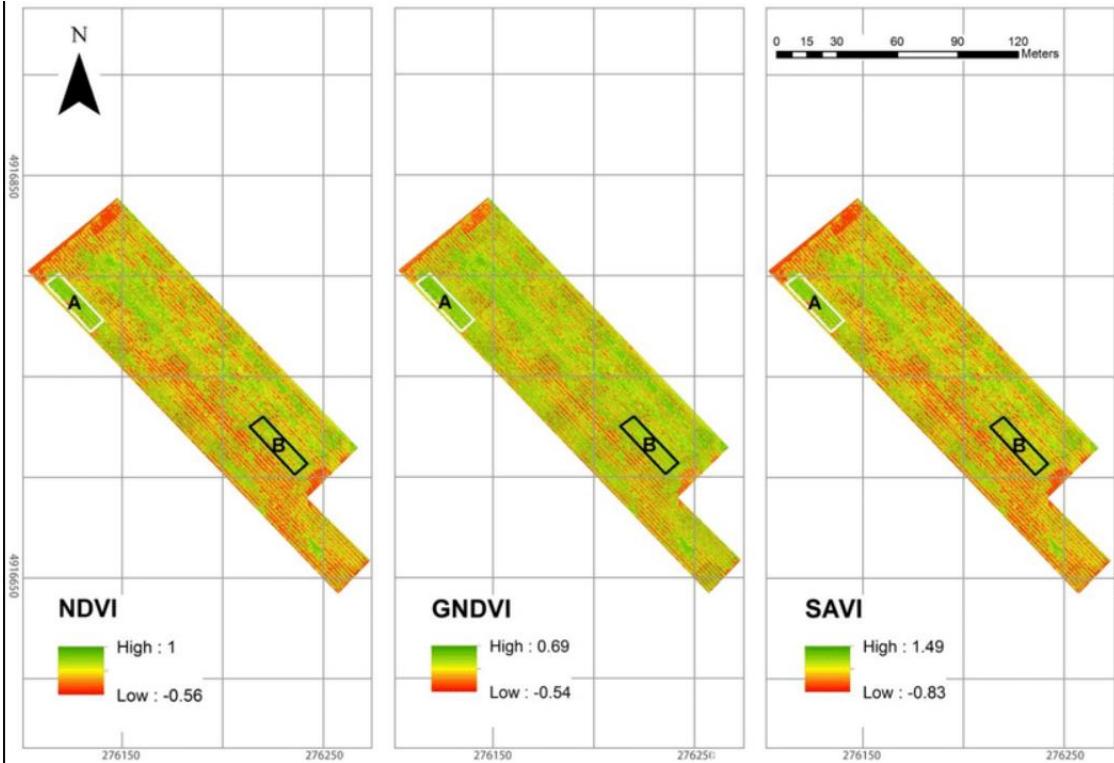
SAVI

$$\text{SAVI} = \frac{(1 + L)(\text{NIR} - \text{Red})}{(\text{NIR} + \text{Red} + L)}$$

O SAVI (Índice de Vegetação Ajustado ao Solo) apresenta uma ligeira variação em relação à fórmula tradicional do NDVI para evitar distorções nos valores de análise quando a vegetação é encontrada em solos expostos. Condições como temperatura ou umidade podem influenciar as faixas de trabalho analisadas e, portanto, os resultados oferecidos pelo indicador. Neste caso, o índice de vegetação SAVI tentará evitar esta influência do solo nos resultados adicionando um fator adicional (L) na equação NDVI que permitirá trabalhar em cenários onde o desenvolvimento das plantas é incipiente.

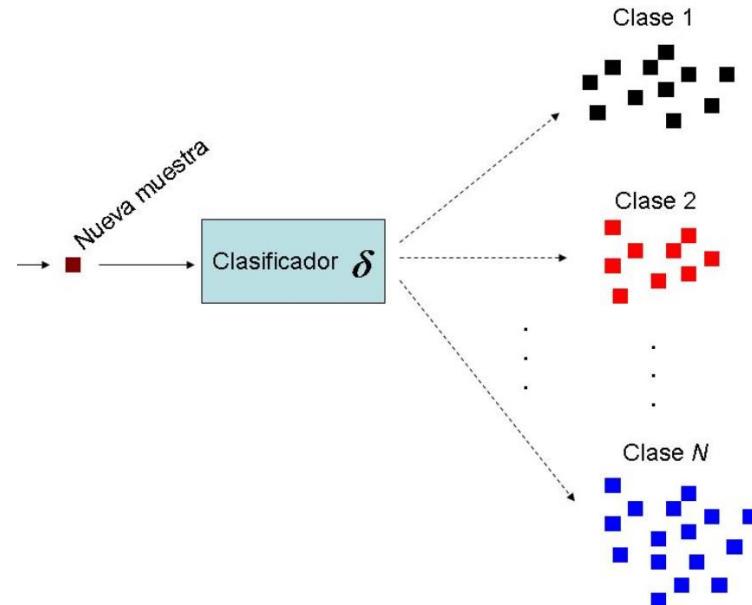


Indices multiespectraux

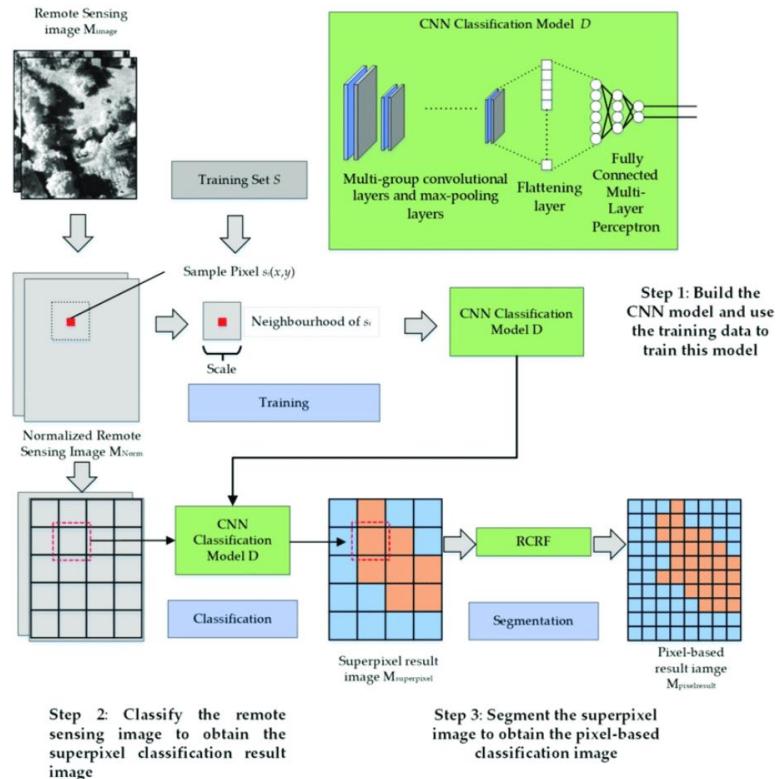
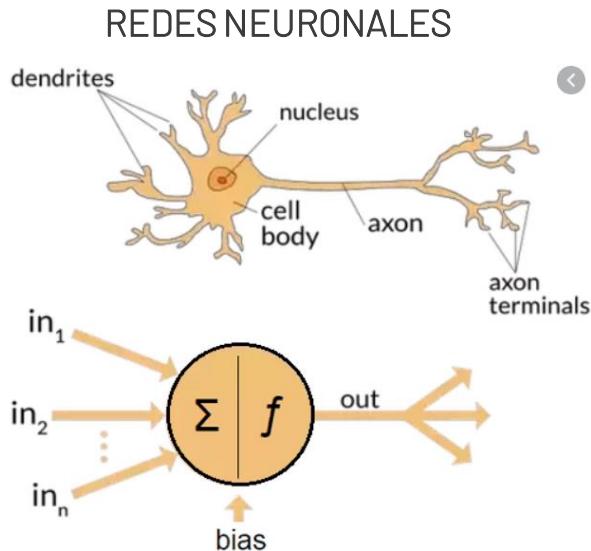


Classificação Supervisionada

A classificação supervisionada exige que você insira regiões de interesse (ROIs) para cada classe de cobertura do solo. São polígonos desenhados sobre áreas homogêneas que se sobrepõem a pixels pertencentes à mesma classe de cobertura do solo.

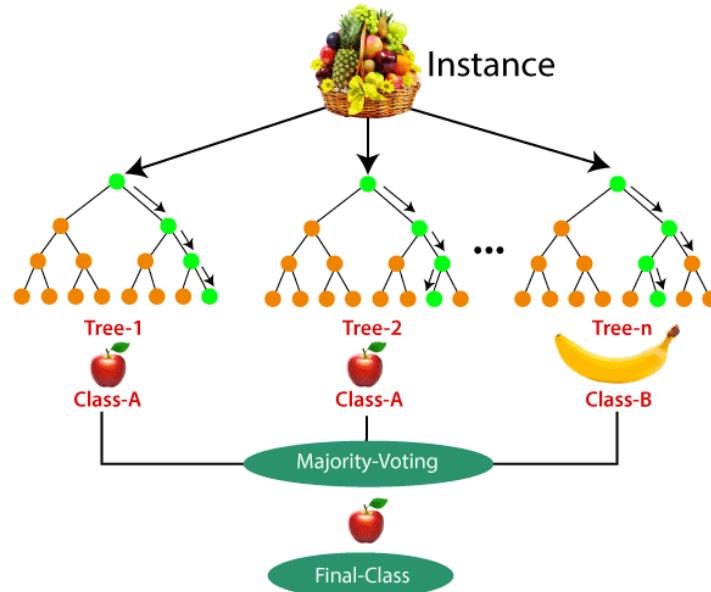
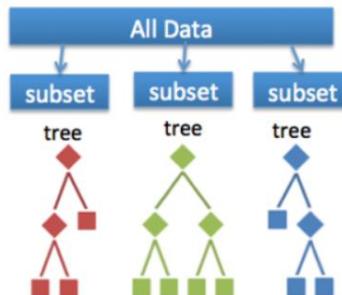


Classificação Supervisionada - Métodos

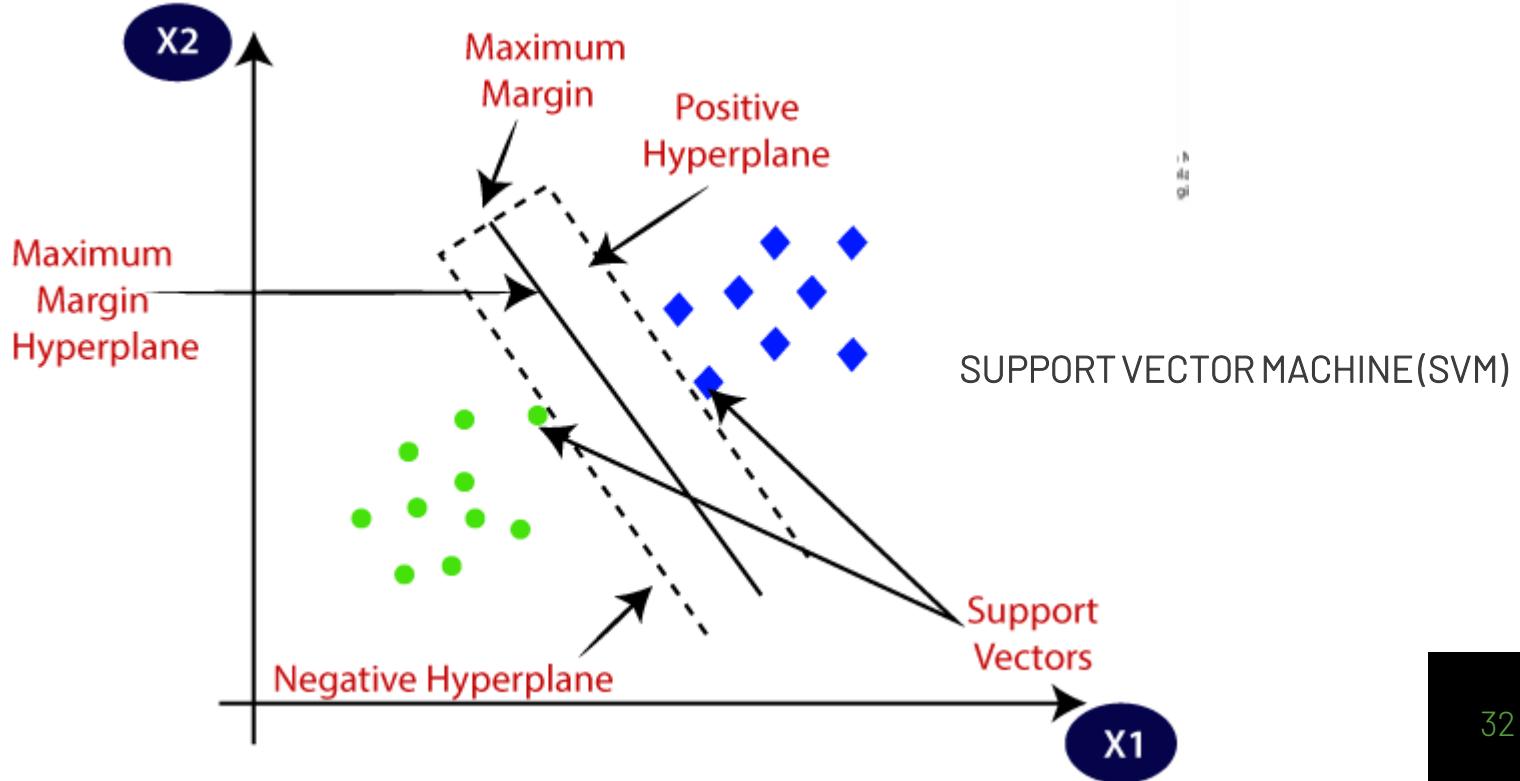


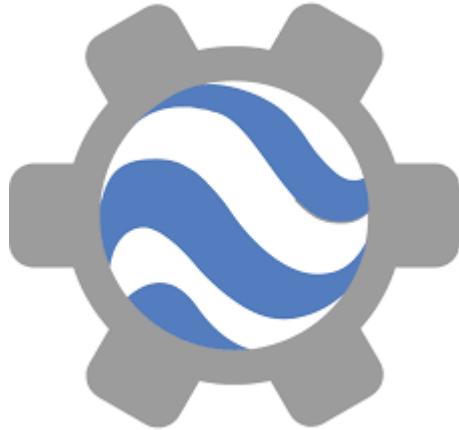
Classificação Supervisionada - Métodos

RANDOM FOREST



Classificação Supervisionada - Métodos





Google Earth Engine (GEE)

Google Earth Engine (ou simplesmente Earth Engine ou GEE), é uma plataforma em nuvem para análise científica e visualização de dados geoespaciais. O Google a define como “A plataforma de processamento geoespacial baseada em nuvem mais avançada do mundo!”



Vantagens de usar o GEE

- Carregue seus próprios dados raster e vetoriais (por exemplo, arquivos GeoTIFF ou Shapefile) para análise.
- Acesso a um catálogo de dados, que inclui todo o catálogo Landsat, MODIS, Sentinel, dados de temperatura, entre outros.
- Você pode exibir os resultados da análise no Google Maps ou em qualquer outra plataforma de mapeamento, como ArcGIS ou QGIS.
- O Earth Engine é gratuito para pesquisa, educação e uso sem fins lucrativos. Para aplicações comerciais, a avaliação do Earth Engine é permitida.
- Não é necessário baixar as imagens para realizar a análise, reduzindo assim os tempos de processamento.

Onde podemos usar o Google Earth Engine (GEE)?

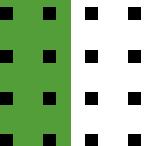
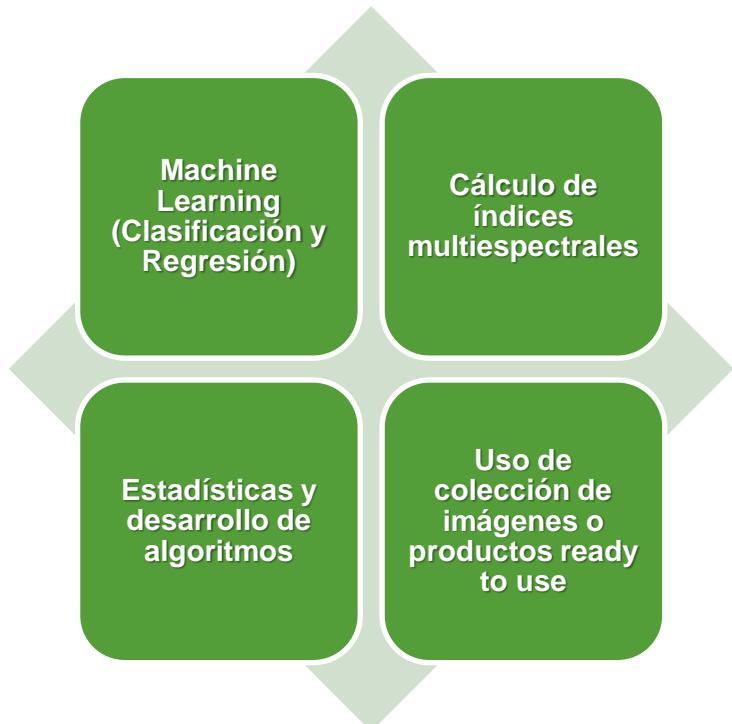
- GEE Javascript

- Earth Engine (EE) Python API

- rgee package R

- Google Earth Engine Complemento - QGIS

Qué podemos hacer en Google Earth Engine?

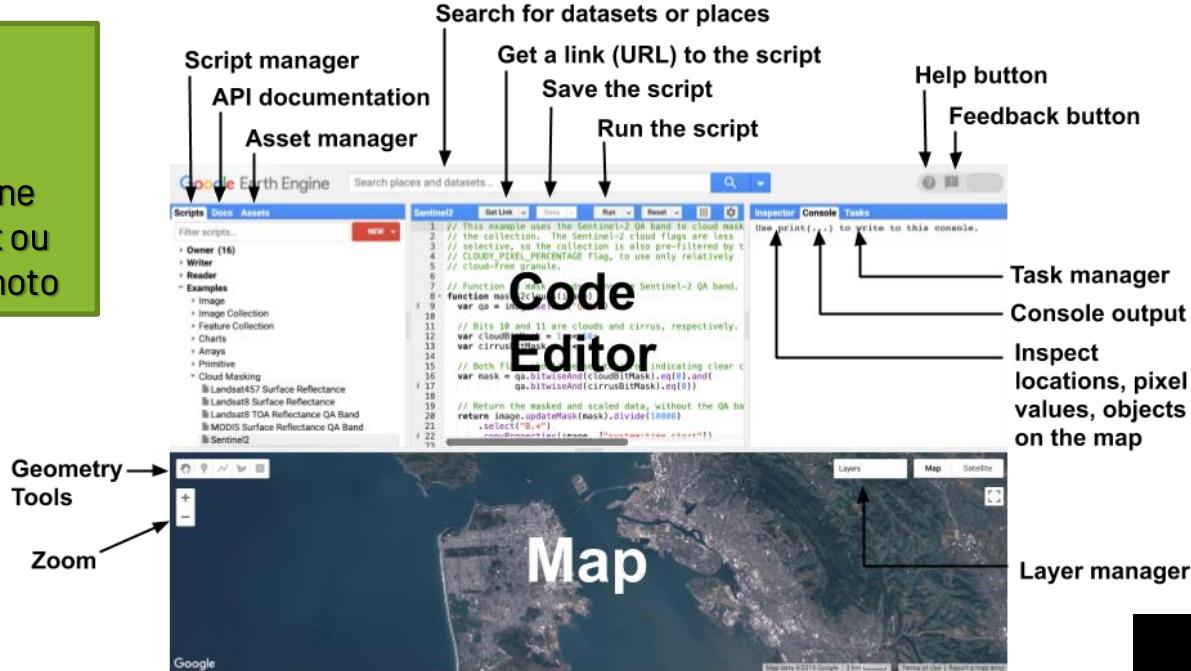


Serviço de geoprocessamento em nuvem - Google Earth Engine

O que eu preciso?

- conta GMAIL

Acesse o Google Earth Engine Conhecimento de Javascript ou Python e Sensoriamento Remoto



Entre no Editor de Código GEE

Google Earth Engine

Platform Datasets Noncommercial Commercial Timelapse Case Studies FAQ [Sign Up](#)

Earth Engine for commercial use: now generally available with Google Cloud. [Get more details here](#)

A planetary-scale platform for Earth science data & analysis

Powered by Google's cloud infrastructure

▶ Watch Video

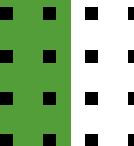


THANK YOU

Welcome to Earth Engine!

Check your inbox at cesarivanalvarezmendoza@gmail.com for information on how to get started.

[TRY THE CODE EDITOR](#) [LOG OUT](#)



01 – Alguns conceitos de Javascript

```
print('Hola amigos');

// Variables
var city = 'Quito';
var country = 'Ecuador';
print(city, country);

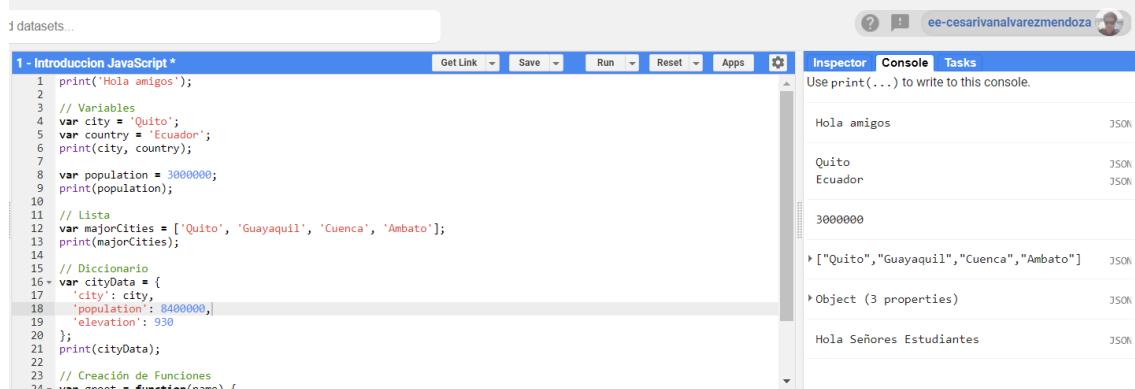
var population = 3000000;
print(population);

// Lista
var majorCities = ['Quito', 'Guayaquil', 'Cuenca'];
print(majorCities);

// Diccionario
var cityData = {
  'city': city,
  'population': 8400000,
  'elevation': 930
};
print(cityData);

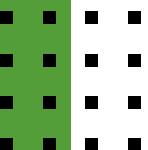
// Creación de Funciones
var greet = function(name) {
  return 'Hola ' + name;
};
print(greet('Señores Estudiantes'));

// Escritura de Comentarios
```



01 - Alguns conceitos de Javascript

| JS code editor | Python |
|---|--|
| <pre>var myFun = function(img) { return img; };</pre> | <pre># Define a function def my_fun(img): return img</pre> |
| <pre>// Define a variable var myVar = "var"; // Logical operators var match = this.and(that); var match = this.or(that); var match = this.not(that);</pre> | <pre># Define a variable my_var = "var" # Logical operators match = this.And(that) match = this.Or(that) match = this.Not(that)</pre> |



02 - Acesso ao catálogo GEE

Earth Engine Data Catalog

Search Español

Home View all datasets Browse by tags Landsat MODIS Sentinel API Docs

Earth Engine Data Catalog

Earth Engine's public data catalog includes a variety of standard Earth science raster datasets. You can import these datasets into your script environment with a single click. You can also upload your own [raster data](#) or vector data for private use or sharing in your scripts.

Looking for another dataset not in Earth Engine yet? Let us know by [suggesting a dataset](#).

Filter list of datasets

| | | | | |
|-----------------------------------|--|--|--|---------------------------------------|
| Canada AAFC Annual Crop Inventory | Allen Coral Atlas (ACA) - Geomorphic Zonation and Benthic Habitat - v1.0 | AHN Netherlands 0.5m DEM, Interpolated | AHN Netherlands 0.5m DEM, Non-Interpolated | AHN Netherlands 0.5m DEM, Raw Samples |
| | | | | |

<https://github.com/osoivan/GEEExercises>

02 - Trabalhar com coleta de imagens



Landsat 9 OLI-2/TIRS-2

2021–Present

Creación de una función llamada applyScaleFactors, donde se requiere de entrada la variable image

```
// Applies scaling factors.
function applyScaleFactors(image) {
  var opticalBands = image.select('SR_B*').multiply(0.0000275).add(-0.2);
  var thermalBands = image.select('ST_B*').multiply(0.00341802).add(149.0);
  return image.addBands(opticalBands, null, true)
    .addBands(thermalBands, null, true);
}
```

Que dará como resultado aplicar esta función

```
var dataset = ee.ImageCollection('LANDSAT/LC09/C02/T1_L2')
  .filterDate('2022-08-01', '2022-10-01');
```

variable
Librería ee
Función para filtrar por fechas
Fecha Inicio
Fecha Fin
Función ImageCollection
Catálogo

02 – Trabalhar com coleta de imagens

```
dataset = dataset.map(applyScaleFactors);  
var visualization = {  
bands: ['SR_B4', 'SR_B3', 'SR_B2'],  
min: 0.0,  
max: 0.3,  
};  
Map.setCenter(-78, 0.01, 7);  
Map.addLayer(dataset, visualization, 'True Color (432)');
```

A quién se aplicará la función

Función o algoritmo a aplicar

Función map, aplica un algoritmo sobre una colección de imágenes

Se crea una variable de visualización con las 3 bandas con colores en un mínimo y máximo

Variable Map por defecto (Mapa a visualizar)

Coordenadas de donde vamos a centrar la imagen y su zoom para visualización

Función addLayer, permite agregar al mapa a visualizar las características requeridas

03 - Filtrar imagem por coordenadas e máscaras

```
var geometry = ee.Geometry.Point([-78.5, 0.02]) ← Variable vectorial tipo geometría de punto  
/ Function to mask clouds using the Sentinel-2 QA band  
  
function maskS2clouds(image) { ← Función para enmascarar nubes y sombras Sentinel-2 usando la banda QA  
  var qa = image.select('QA60');  
  
  // Bits 10 and 11 are clouds and cirrus, respectively.  
  var cloudBitMask = 1 << 10;  
  var cirrusBitMask = 1 << 11;  
  
  // Both flags should be set to zero, indicating clear conditions.  
  var mask = qa.bitwiseAnd(cloudBitMask).eq(0)  
    .and(qa.bitwiseAnd(cirrusBitMask).eq(0));  
  
  return image.updateMask(mask).divide(10000);  
}
```

Que retornará la función de enmascarar nubes

03 - Filtrar imagem por coordenadas e máscaras

```
var dataset = ee.ImageCollection('COPERNICUS/S2_SR_HARMONIZED')
    .filterDate('2020-01-01', '2020-03-30')
    // Pre-filter to get less cloudy granules.
    .filter(ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE',20))
    Filtrar con la función  
maskS2clouds
    .map(maskS2clouds)
    .filter(ee.Filter.bounds(geometry));
Filtrar con la capa Cloudy Pixel
```

var visualization = {
 min: 0.0,
 max: 0.3,
 bands: ['B4', 'B3', 'B2'],
 //bands: ['B5', 'B4', 'B3'],
};
Filtrar con geometría
Se aplica la función media, para que nos genere
una imagen con la media de pixeles de las
fechas filtradas

- ▪ ▪ ▪ Map.centerObject(geometry, 10)
- ▪ ▪ ▪ //Map.addLayer(dataset.mean(), visualization);
- ▪ ▪ ▪ Map.addLayer(dataset.mean(), visualization, 'RGB');

03 - Filtrar imagem por coordenadas e máscaras

Google Earth Engine

Search places and datasets...

Scripts Docs Assets

Aplicación
borrador

Archive
No accessible repositories. Click Refresh to check again.

Examples
Image
Image Collection
Feature Collection
Charts
Arrays
Primitive
Cloud Masking
Code Editor

3 - Filtrar imagen Sentinel-2 *

Get Link Save Run Reset Apps

```
15 // Both flags should be set to zero, indicating clear conditions.
16 var mask = qa.bitwiseAnd(cloudBitMask).eq(0)
17 .and(qa.bitwiseAnd(cirrusBitMask).eq(0));
18
19 return image.updateMask(mask).divide(10000);
20 }
21
22 var dataset = ee.ImageCollection('COPERNICUS/S2_SR_HARMONIZED')
23 .filterDate('2020-01-01', '2020-03-30')
24 // Pre-filter to get less cloudy granules.
25 .filter(ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE',20))
26 .map(maskS2Clouds)
27 .filter(ee.Filter.bounds(geometry));
28
29 var visualization = {
30   min: 0.0,
```

Inspector Console Tasks

Point (-78.25381, 0.14379) at 19m/px

Pixels

Layer 1: Image (23 bands)

TCI R QA20

0.4
0.3
0.2
0.1

B4 B9 TCI R QA20

Layers Mapa Satélite

47

04 - Criação de mosaicos e composições



MOSAIC



MEDIAN COMPOSITE

Mosaic vs. Composite

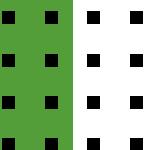
04 - Criação de mosaicos e composições

```
var geometry = ee.Geometry.Point([-78.5, 0.02])  
  
var filtered = ee.ImageCollection('COPERNICUS/S2_SR_HARMONIZED')  
    .filterDate('2020-01-01', '2022-01-01')  
    // Pre-filter to get less cloudy granules.  
    .filter(ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE',20))  
    .filter(ee.Filter.bounds(geometry));  
  
var mosaic = filtered.mosaic() //Crea un mosaico con los últimos pixeles de la  
colección encontrados ← Función mosaico  
  
var medianComposite = filtered.median(); //Genera una mediana de pixeles entre  
fechas ↑ Función mediana
```

- ▪ ▪
- ▪ ▪
- ▪ ▪
- ▪ ▪
- ▪ ▪
- ▪ ▪
- ▪ ▪
- ▪ ▪

04 - Criação de mosaicos e composições

```
var rgbVis = {  
    min: 0.0,  
    max: 3000,  
    bands: ['B4', 'B3', 'B2'],  
};  
  
// set styling  
Map.centerObject(geometry, 15) ← Agrega cada elemento  
                                como una capa en el  
                                visualizador  
Map.addLayer(filtered, rgbVis, 'Colección original');  
Map.addLayer(mosaic, rgbVis, 'Mosaico');  
Map.addLayer(medianComposite, rgbVis, 'Composición mediana')  
Map.addLayer(geometry,{color: 'red', pointSize: 30, pointShape: 'circle'}, 'Punto');  
  
Propiedades del Vector  
Cambiar nombre de capa
```



04 - Criação de mosaicos e composições

Screenshots illustrating the creation of a Sentinel-2 mosaic and median composition using the ArcGIS API for JavaScript.

The interface shows the following components:

- Scripts** tab selected in the top navigation bar.
- Owner (4)** dropdown menu showing items like `users/cesarivanalvarezmendoza/CAT`, `users/cesarivanalvarezmendoza/prueba`, etc.
- Code Editor**: A code editor window titled "4 - Mosaico y Mediana Sentinel-2" containing the following JavaScript code:

```
i 9 var mosaic = filtered.mosaic() //Crea un mosaico con los últimos pixeles de la colección encontrados
10
11 var medianComposite = filtered.median(); //Genera una mediana de pixeles entre fechas
12
13 var rgbVis = {
14   min: 0.0,
15   max: 3000,
16   bands: ['B4', 'B3', 'B2'],
17 };
18
19 // set styling
i 20 Map.centerObject(geometry, 15)
21 Map.addLayer(filtered, rgbVis, 'Colección original');
22 Map.addLayer(mosaic, rgbVis, 'Mosaico');
i 23 Map.addLayer(medianComposite, rgbVis, 'Composición mediana')
24 Map.addLayer(geometry, {color: 'red', pointSize: 30, pointShape: 'circle'}, 'Punto');
```
- Inspector** tab selected in the top navigation bar.
- Map View**: A satellite map showing a green landscape with a small red dot indicating the center of interest.
- Layers**, **Mapa**, and **Satélite** buttons in the bottom right corner of the map view.

05 - Coleções de vetores (coleções de recursos)

```
var country = 'BL'; //Choose the country initials [Two-letter FIPS country code]
https://en.wikipedia.org/wiki/List\_of\_FIPS\_country\_codes

// Load country features from Large Scale International Boundary (LSIB) dataset.
var countries = ee.FeatureCollection('USDOS/LSIB_SIMPLE/2017');  

var table = countries.filter(ee.Filter.eq('country_co', ee.String(country)));  
  

Map.centerObject(table,5);
Map.addLayer(table, {color: 'red'})
```

↑

Aregar al mapa en color rojo

↑

Feature collection disponible en GEE de límite de países

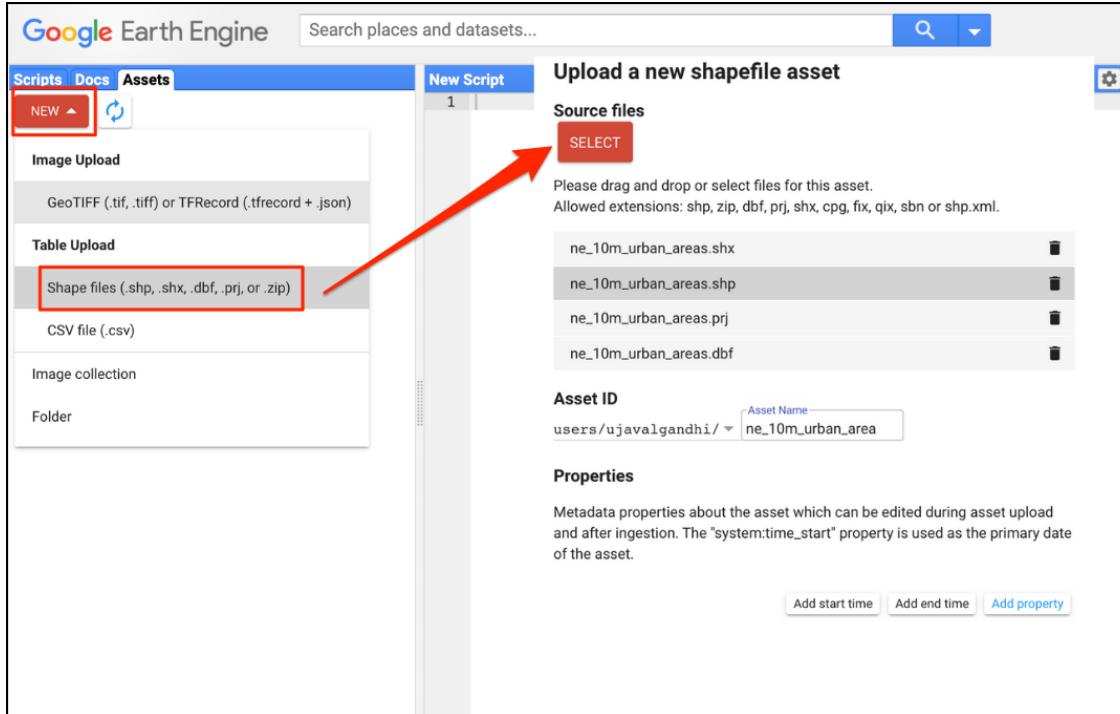
05 - Coleções de vetores (coleções de recursos)

The screenshot shows the Google Earth Engine interface. The top navigation bar includes 'Google Earth Engine', a search bar ('Search places and datasets...'), and user information ('ee-cesarivanalvarezmendoza'). The left sidebar has tabs for 'Scripts', 'Docs', and 'Assets', with 'Assets' selected. Under 'Assets', there are two sections: 'Owner (4)' and 'User (1)'. The 'Owner (4)' section lists 'users/cesarivanalvarezmendoza/CIAT' (ExtractData, Temperature) and 'users/cesarivanalvarezmendoza/prueba' (China_2020_HansenGlobalForestChange, SVMIL7, SVMIL7_China_2000, SVMIL7_China_2000_10Percentage, SVMIL7_Ecuador, Sentinel-5 NO2 Troposférico). The 'User (1)' section lists 'ee-cesarivanalvarezmendoza/5 - Vectors FeatureCollections'. The code editor contains the following script:

```
1 var country = 'BL'; //Choose the country initials [Two-letter FIPS country code]([https://en.wikipedia.org/wiki/List_of_U.S._states_by_FIPS_code])
2
3 // Load country features from Large Scale International Boundary (LSIB) dataset.
4 var countries = ee.FeatureCollection('USDOES/LSIB_SIMPLE/2017');
5 var table = countries.filter(ee.Filter.eq('country_co', ee.String(country)));
6
7 Map.centerObject(table,5);
8 Map.addLayer(table, {color: 'red'})
```

The main map view shows South America with Bolivia highlighted in red. The map includes labels for countries like Peru, Brazil, Argentina, Chile, Uruguay, and Paraguay, along with major cities and state/province boundaries. The bottom of the map displays standard Google Earth Engine controls and copyright information.

06 - Importar dados (Shapefiles, Geotiff, CSV)



06 - Importar dados (Shapefiles, Geotiff, CSV)

Google Earth Engine

Search places and datasets...

Scripts Docs Assets

users/cesarivanalvarezmendoza

- shapefiles
 - ChinaBoundary
 - EcuadorBound
 - EcuadorProvinces
 - PLOTS4326
 - Part1
 - TrainingAreas
 - Jianguo_province
 - CN015
 - JP01
 - JP02

5 - Vectores FeatureCollections *

Imports (1 entry) Get Link Save Run Reset Apps

```
1 var table: Table = users/cesarivanalvarezmendoza/shapefiles/PLOTS4326
2 Map.centerObject(table, 17);
3 Map.addLayer(table, {color: 'red'})
```

Inspector Console Tasks

Click on the map to inspect the layers.

Import into script

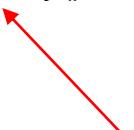
Layers Map Satélite

25

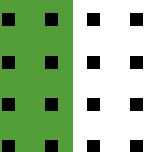
@Goolala

07 - Clipe de Coleção de Imagens

```
var s2 = ee.ImageCollection('COPERNICUS/S2_SR_HARMONIZED')
var urban = ee.FeatureCollection("users/cesarivanalvarezmendoza/shapefiles/PLOTS4326")

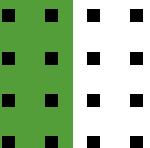
var geometry = urban.geometry()

var rgbVis = {
  min: 0.0,
  max: 3000,
  bands: ['B4', 'B3', 'B2'],
}
Generar la variable  
del polígono desde  
el featurecollection
};

var filtered = s2.filter(ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE', 30))
  .filter(ee.Filter.date('2021-01-01', '2022-01-01'))
  .filter(ee.Filter.bounds(geometry))
```



07 - Clipe de Coleção de Imagens

```
var image = filtered.median();
var clipped = image.clip(geometry) ← Función cortar
Map.centerObject(urban,17)           imagen en base a
Map.addLayer(clipped, rgbVis, 'Clipped')   geometría
```



07 - Clipe de Coleção de Imagens

Google Earth Engine

Search places and datasets...

Scripts Docs Assets

7 - Clip Imagen Sentinel-2

Get Link Save Run Reset Apps

Inspector Console Tasks

Use print(...) to write to this console.

```
5 var rgbVis = {  
6   min: 0.0,  
7   max: 3000,  
8   bands: ['B4', 'B3', 'B2'],  
9 };  
10  
11 var filtered = s2.filter(ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE', 30))  
12   .filter(ee.Filter.date('2021-01-01', '2022-01-01'))  
13   .filter(ee.Filter.bounds(geometry))  
14  
15 var image = filtered.median();  
16  
17 var clipped = image.clip(geometry)  
18  
19 Map.centerObject(urban, 17)  
20 Map.addLayer(clipped, rgbVis, 'Clipped')
```

Writer

No accessible contributor. Click Refresh to check again.

Layers Map Satélite

08 - Exportar imágenes

```
var country = 'BL'; //Choose the country initials [Two-letter FIPS country code]  
https://en.wikipedia.org/wiki/List\_of\_FIPS\_country\_codes
```

```
// Load country features from Large Scale International Boundary (LSIB) dataset.  
var countries = ee.FeatureCollection('USDOS/LSIB_SIMPLE/2017');  
var table = countries.filter(ee.Filter.eq('country_co', ee.String(country)));
```

```
// Make the clip boundary.
```

```
var clipToCol = function(image){  
    return image.clip(table);  
};
```

Función para
realizar clip

```
var collection = ee.ImageCollection('COPERNICUS/S5P/OFFL/L3_NO2')  
    .select('NO2_column_number_density')  
    .filterBounds(table)  
    .filterDate('2019-03-16', '2019-03-31')  
    .map(clipToCol);
```

08 – Exportar imagens

```
var band_viz = {  
  min: 0,  
  max: 0.00009,  
  palette: ['green','yellow', 'red']  
};  
  
Map.addLayer(collection.median(), band_viz, 'S5P N02');  
Map.addLayer(table);  
Map.centerObject(table,5);  
print(collection.size()) ← Ver resolución de  
la imagen  
  
// Export the image, specifying scale and region. ← Exportar a Google  
Export.image.toDrive({  
  image: collection.median(), ← Drive como  
  description: 'Bolivia_2019',  
  scale: 1000,  
  region: table  
});  
Exportar la  
mediana del image  
collection
```

08 - Exportar imágenes

Google Earth Engine

Search places and datasets...

Script Docs Assets

7 - Clip Imagen Sentinel-2 *

```
20 max: 0.0009,
21 palette: ['green','yellow', 'red']
22 };
23
24 Map.addLayer(collection.median(), band_viz, 'SSP N02');
25 Map.addLayer(table);
26 Map.centerObject(table,5);
27
28 // Export the image, specifying scale and region.
29 Export.image.toDrive({
30   image: collection,
31   description: 'Bolivia_2019',
32   scale: 1000,
33   region: table
34 });
```

Get Link Save Run Reset Apps

Inspector Console Tasks

Manage tasks.

Search or cancel multiple tasks in the Task Manager.

UNSUBMITTED TASKS

Bolivia_2019 RUN

Sentinel-2-CopernicusS2_SR_432bands_sk...

Brasil

Perú

Mapa Satélite

61

The screenshot shows the Google Earth Engine (GEE) web interface. On the left, there's a sidebar with tabs for 'Script', 'Docs', and 'Assets'. The 'Script' tab is active, displaying a code editor with a script titled '7 - Clip Imagen Sentinel-2 *'. The code uses the GEE API to process a Sentinel-2 collection, calculate a median image, clip it to a shapefile, and export it to Google Drive. The right side of the interface shows a map of South America with Bolivia highlighted in dark green. The map also shows parts of Peru, Brazil, and Argentina. The top right corner of the map has a status bar with 'PARAIBA' and 'Ceará'. Below the map, there are buttons for 'Mapa' and 'Satélite'. The bottom right corner of the map has a status bar with '61'. The top right of the interface has a user profile and a 'RUN' button for the task.

08 – Exportar imagens

Task: Initiate image export

Task name (no spaces) *

Bolivia_2019

Coordinate Reference System (CRS)

EPSG:3857

Scale (m/px)

1000

| DRIVE | CLOUD STORAGE | EE ASSET |
|-------|---------------|----------|
| <hr/> | <hr/> | <hr/> |

Drive folder

Drive folder name or blank for root

Filename *

Bolivia_2019

File format *

GEO_TIFF

[CANCEL](#)

[RUN](#)

The screenshot shows the Google Earth Engine interface. On the left, the 'Scripts' panel lists several projects and scripts, including '7 - Clip Imagen Sentinel-2'. The main workspace displays a map of South America with a large green polygon highlighting the borders of Bolivia. The map includes labels for countries like Peru, Brazil, and Paraguay, and cities like Lima, Brasília, and São Paulo. The top navigation bar includes a search bar, and the right side features an 'Inspector' panel with task management and a 'Source Script' button.

09 – Cálculo de índices - NDVI

- ```
■ ■ ■ ■ Map.addLayer(composite, rgbVis, 'RGB Ecuador')
```

## Generar una imagen compuesta por la mediana

## 09 – Cálculo de índices - NDVI

```
// Write a function that computes NDVI for an image and adds it as a band
function addNDVI(image) {
 var ndvi = image.normalizedDifference(['B8', 'B4']).rename('ndvi');
 return image.addBands(ndvi);
}
```



Función NDVI

```
// Map the function over the collection
var withNdvi = filtered.map(addNDVI);
var composite = withNdvi.median()
var ndviComposite = composite.select('ndvi').clip(geometry)
```

```
var palette = [
 'FFFFFF', 'CE7E45', 'DF923D', 'F1B555', 'FCD163', '99B718',
 '74A901', '66A000', '529400', '3E8601', '207401', '056201',
 '004C00', '023B01', '012E01', '011D01', '011301'];
```

```
var ndviVis = {min:0, max:1, palette: palette }
Map.addLayer(ndviComposite, ndviVis, 'NDVI')
```

## Función NDVI

## Paleta de colores NDVI

- ■ ■ Map.centerObject(table,7);
  - ■ ■ Map.addLayer(table, {color: 'red'}, 'Limite Ecuador')

# 09 - Cálculo de índices - NDVI

Google Earth Engine Search places and datasets...

Scripts Docs Assets

- ForestGainLossbyCountryUsingShapefiles 200...
- ForestLossbyYear
- HansenTreeCover2000 Export by Country
- LAI\_Modis\_Extraction\_by\_country
- S2
- exportSentinel2
- users/cesarivanalvarezmendoza/UASB**
  - 1 - Introducción JavaScript
  - 2 - Visualización Imagen Landsat 9
  - 3 - Filtrar Imagen Sentinel-2
  - 4 - Mosaico y Mediana Sentinel-2

9 - NDVI \*23 var withNdvi = filtered.map(addNDVI);  
i 24 var composite = withNdvi.median()  
i 25 var ndviComposite = composite.select('ndvi').clip(geometry)  
26  
var palette = [  
27 'FFFFFF', 'CE7E45', 'DF923D', 'F1B555', 'FCD163', '99B718',  
28 '74A901', '66A000', '529400', '3E8601', '207401', '056201',  
29 '004C00', '023B01', '012E01', '011D01', '011301'];  
30  
var ndviVis = {min:0, max:1, palette: palette }  
31 Map.addLayer(ndviComposite, ndviVis, 'NDVI')  
32  
33 Map.centerObject(table,);  
34  
35 Map.addLayer(table, {color: 'red'}, 'Limite Ecuador')  
36

Get Link Save Run Reset Apps

Inspector Console Tasks

Use print(...) to write to this console.

Layers Mapa Satélite

Combinaciones de teclas | Datos de mapas ©2022 Google | 50 km | Términos de uso

# 10 – Reductores (Reducers)

```
//Forest loss between 2000 to 2020
var year = 20; //Choose the year between 2000 (00) to 2020 (20)
var country = 'EC'; //Choose the country initials [Two-letter FIPS country
code](https://en.wikipedia.org/wiki/List_of_FIPS_country_codes

// Load country features from Large Scale International Boundary (LSIB) dataset.
var countries = ee.FeatureCollection('USDOS/LSIB_SIMPLE/2017');
var selected = countries.filter(ee.Filter.eq('country_co', ee.String(country)));

// Make the clip boundary.
var clipToCol = function(image){
 return image.clip(selected);
};

var gfc2020 = ee.Image('UMD/hansen/global_forest_change_2020_v1_8');

// Canopy loss during the selected year
var loss = gfc2020.select(['lossyear']).eq(year).clip(selected) //ClipGeometry can any geometry or feature or
shapfile
```

Catalog Global Forest

Filtrar por año

# 10 – Reductores (Reducers)

```
// Get the forest loss in square kilometers.
var arealmage = loss.multiply(ee.Image.pixelArea()).divide(1000000);

var stats = arealmage.reduceRegion({
 reducer: ee.Reducer.sum(),
 geometry: selected,
 scale: 30,
 maxPixels: 1e9
});
print('Pixels representing loss during the year choosed: ', stats.get('lossyear'), 'square kilometers in 20', year);

// Showing the map result
var losscolors = {
 bands: ['lossyear'],
 min: 0,
 max: 20,
 palette: ['black', 'green']
};
Map.addLayer(loss, losscolors,'tree cover');
Map.centerObject(selected,7);
```

Suma los píxeles dentro del polígono de featurecollection

Función get

# 10 – Reductores (Reducers)

10 - Reductores

```
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
```

Get Link Save Run Reset Apps Inspector Console Tasks

Use print(...) to write to this console.

Pixels representing loss during the year choosed: 515.1150369271691 square kilometers in 20 20

JSON JSON

```
10 - Reductores
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
```

Get Link Save Run Reset Apps Inspector Console Tasks

Use print(...) to write to this console.

Pixels representing loss during the year choosed: 515.1150369271691 square kilometers in 20 20

JSON JSON

Layers Map Satélite

# 11 – Serie de Tiempo

```
var s2 = ee.ImageCollection("COPERNICUS/S2_SR_HARMONIZED");
var geometry = ee.Geometry.Polygon([
 [82.60642647743225, 27.16350437805251],
 [82.60984897613525, 27.1618529901377],
 [82.61088967323303, 27.163695288375266], ← Creando un polígono de estudio o ROI
 [82.60757446289062, 27.16517483230927]
]);
Map.addLayer(geometry, {color: 'red'}, 'Farm')
Map.centerObject(geometry)
var rgbVis = {min: 0.0, max: 3000, bands: ['B4', 'B3', 'B2']};

var filtered = s2.filter(ee.Filter.date('2017-01-01', '2022-01-01'))
 .filter(ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE', 30))
 .filter(ee.Filter.bounds(geometry))

// Write a function for Cloud masking
function maskS2clouds(image) {
 var qa = image.select('QA60')
 var cloudBitMask = 1 << 10;
 var cirrusBitMask = 1 << 11;
 var mask = qa.bitwiseAnd(cloudBitMask).eq(0).and(
 qa.bitwiseAnd(cirrusBitMask).eq(0))
 return image.updateMask(mask)//.divide(10000)
 .select("B.*")
 .copyProperties(image, ["system:time_start"])
}
```

# 11 - Serie de Tiempo

```
var filtered = filtered.map(maskS2clouds)
// Write a function that computes NDVI for an image and adds it as a band
function addNDVI(image) {
 var ndvi = image.normalizedDifference(['B8', 'B4']).rename('ndvi');
 return image.addBands(ndvi);
}

// Map the function over the collection
var withNdvi = filtered.map(addNDVI);

// Display a time-series chart
var chart = ui.Chart.image.series({
 imageCollection: withNdvi.select('ndvi'),
 region: geometry,
 reducer: ee.Reducer.median(),
 scale: 10
}).setOptions({
 lineWidth: 1,
 title: 'Serie de Tiempo NDVI',
 interpolateNulls: true,
 vAxis: {title: 'NDVI'},
 hAxis: {title: '', format: 'YYYY-MMM'}
})
print(chart);
```

← Función NDVI

← Gráfico Serie de Tiempo

# 11 – Serie de Tiempo

```
var withNDVI = withNdvi.select('ndvi')
var withNDVI = ee.ImageCollection(withNDVI).toBands();

// Get Zonal Statistics
var reduced = withNDVI.reduceRegions({
 collection: geometry,
 reducer: ee.Reducer.mean(),
 scale: 100,
});
print(reduced);

// the resulting mean is a FeatureCollection
// so you can export it as a table
Export.table.toDrive({
 collection: reduced,
 description: 'NDVIpolygons',
 fileNamePrefix: 'NDVIpolygons',
 fileFormat: 'CSV'
})
```

← Cálculo de estadísticas NDVI

← Exportar a CSV

- ■ ■
- ■ ■
- ■ ■
- ■ ■
- ■ ■
- ■ ■
- ■ ■

# 11 - Serie de Tiempo

Google Earth Engine

Search places and datasets...

Scripts Docs Assets

11 - Series de Tiempo

Get Link Save Run Reset Apps

Inspector Console Tasks

Use print(...) to write to this console.

Serie de Tiempo NDVI

NDVI

2019-Jan 2020-Jan 2021-Jan 2021-Sep

ndvi

Script Content:

```
var cloudBitMask = 1 << 10;
var cirrusBitMask = 1 << 11;
var mask = qa.bitwiseAnd(cloudBitMask).eq(0).and(
 qa.bitwiseAnd(cirrusBitMask).eq(0));
return image.updateMask(mask).divide(10000)
 .select("B,*")
 .copyProperties(image, ["system:time_start"]);
```

```
i 28 var filtered = filtered.map(maskS2clouds);
// Write a function that computes NDVI for an image and adds it as a band
function addNDVI(image) {
 var ndvi = image.normalizedDifference(['B8', 'B4']).rename('ndvi');
 return image.addBands(ndvi);
}
```

```
i 34
// Map the function over the collection
var withNdvi = filtered.map(addNDVI);
```

```
i 37
// Display a time-series chart
var chart = ui.Chart.image.series({
```

Layers Mapa Satélite

# 12 – Clasificación Supervisada - SVM

```
var country = 'CH';
var countries = ee.FeatureCollection('USDOS/LSIB_SIMPLE/2017');
var table = countries.filter(ee.Filter.eq('country_co', ee.String(country)));

// Make the clip boundary.
var clipToCol = function(image){
 return image.clip(table);
};

// Make a cloud-free Landsat 7 TOA composite (from raw imagery).
var l8 = ee.ImageCollection('LANDSAT/LE07/C01/T1') ← Landsat 7 TOA
 .map(clipToCol);

var image = ee.Algorithms.Landsat.simpleComposite({
 collection: l8.filterDate('2000-01-01', '2000-12-31'),
 asFloat: true
});

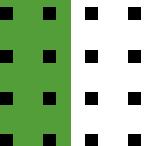
// Use these bands for prediction. We only used the multiespectral bands
var bands = ['B1','B2','B3','B4','B5','B7']; ← Bandas para la clasificación

// Manually created polygons. Only I add two polygons to each category forest or non-forest
var forest1 = ee.Geometry.Rectangle(106.024466, 28.470375, 106.114408, 28.385354); ← Área de Entrenamiento
var forest2 = ee.Geometry.Rectangle(123.926517, 52.129826, 124.431256, 51.743099);
var nonForest1 = ee.Geometry.Rectangle(82.564232, 40.174726, 85.900993, 38.822149);
var nonForest2 = ee.Geometry.Rectangle(100.003374, 37.005547, 100.315863, 36.751742);
```

# 12 – Clasificación Supervisada - SVM

```
// Make a FeatureCollection from the hand-made geometries.
var polygons = ee.FeatureCollection([
 ee.Feature(nonForest1, {'class': 0}),
 ee.Feature(nonForest2, {'class': 0}),
 ee.Feature(forest1, {'class': 1}),
 ee.Feature(forest2, {'class': 1}),
]);

// Get the values for all pixels in each polygon in the training.
var training = image.sampleRegions({
 // Get the sample from the polygons FeatureCollection. ← Reclasificación de categorías
 collection: polygons,
 // Keep this list of properties from the polygons. ← Áreas de entrenamiento en una sola variable
 properties: ['class'],
 // Set the scale to get Landsat pixels in the polygons.
 scale: 1000
});
```



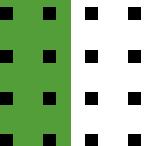
# 12 – Clasificación Supervisada - SVM

```
// Create an SVM classifier with custom parameters. ← Algoritmo de Machine Learning SVM
var classifier = ee.Classifier.libsvm({
 kernelType: 'RBF',
 gamma: 0.5,
 cost: 10
});

// Train the classifier. ← Áreas de entrenamiento
var trained = classifier.train(training, 'class', bands);

// Classify the image. ← Clasificación
var classified = image.classify(trained);

// Display the classification result and the input image.
Map.centerObject(table,4);
Map.addLayer(polygons, {}, 'training polygons');
Map.addLayer(classified,
 {min: 0, max: 1, palette: ['red', 'green']},
 'deforestation');
```



# 12 - Clasificación Supervisada - SVM

code.earthengine.google.com

Cartografía FCUP Otros Reproductor web d... Paper Checker | Onlin... Racurs :: PRODUCT... Mapas de distribuci... Principal compone... Atmospheric and R... Archivos de Inform... Otros marcadores

Google Earth Engine  ee-cesarivanalvarezmendoza

Scripts Docs Assets

- SVMI\_Ecuador
- Sentinel-5 NO2 Troposférico
- Sentinel5P
- extractLandsat7
- extractLandsat8
- extractLandsat9
- users/cesarivanalvarezmendoza/purdue
- ExtractNDVIGain
- ForestGainLossbyCountry 2000 - 2012
- ForestGainLossbyCountry 2000 - 2012 Export
- ForestGainLossbyCountryUsingShapefiles 200...
- ForestLossbyYear
- HansenTreeCover2000 Export by Country
- LAL.Modis\_Extraction\_by\_country
- S2
- exportSentinel2
- users/cesarivanalvarezmendoza/UASB
- 1 - Introducción\_JavaScript

12 - Clasificación SVM Get Link Save Run Reset Apps

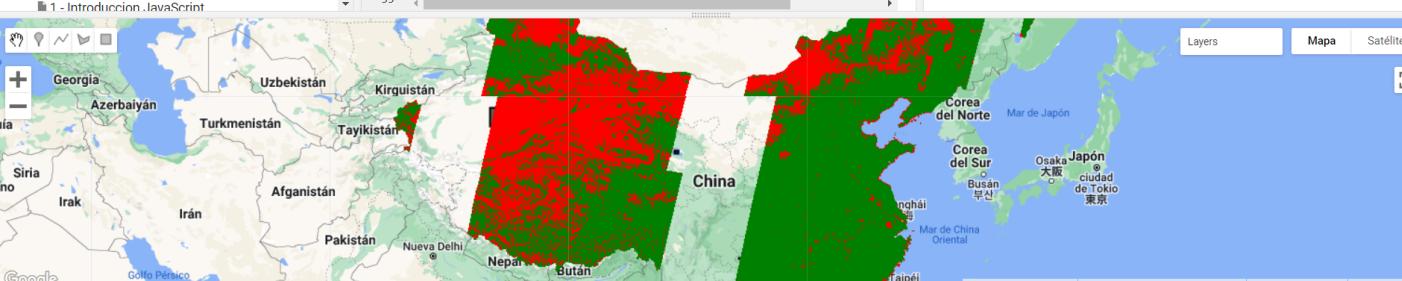
```
// Get the values for all pixels in each polygon in the training.
var training = image.sampleRegions({
 // Get the sample from the polygons FeatureCollection.
 collection: polygons,
 // Keep this list of properties from the polygons.
 properties: ['class'],
 // Set the scale to get Landsat pixels in the polygons.
 scale: 1000
});

// Create an SVM classifier with custom parameters.
var classifier = ee.Classifier.libSVM({
 kernelType: 'RBF',
 gamma: 0.5,
 cost: 10
});

// Train the classifier.
var trained = classifier.train(training, 'class', bands);
```

Inspector Console Tasks

Use print(...) to write to this console.



76

# 13 – Clasificación Supervisada Matriz de Confusión – Random Forest

```

// A Sentinel-2 surface reflectance image, reflectance bands selected,
// serves as the source for training and prediction in this contrived example.
var img = ee.Image('COPERNICUS/S2_SR/20210109T185751_20210109T185931_T10SEG')
 .select('B.*');

// ESA WorldCover land cover map, used as label source in classifier training.
var lc = ee.Image('ESA/WorldCover/v100/2020');

// Remap the land cover class values to a 0-based sequential series.
var classValues = [10, 20, 30, 40, 50, 60, 70, 80, 90, 95, 100];
var remapValues = ee.List.sequence(0, 10);
var label = 'lc';
lc = lc.remap(classValues, remapValues).rename(label).toByte();

// Add land cover as a band of the reflectance image and sample 100 pixels at
// 10 m scale from each land cover class within a region of interest.
var roi = ee.Geometry.Rectangle(-122.347, 37.743, -122.024, 37.838);
var sample = img.addBands(lc).stratifiedSample({
 numPoints: 100,
 classBand: label,
 region: roi,
 scale: 10,
 geometries: true
});

```

# 13 – Clasificación Supervisada Matriz de Confusión – Random Forest

```
// Add a random value field to the sample and use it to approximately split 80%
// of the features into a training set and 20% into a validation set.
sample = sample.randomColumn();
var trainingSample = sample.filter('random <= 0.8');
var validationSample = sample.filter('random > 0.8');

// Train a 10-tree random forest classifier from the training sample.
var trainedClassifier = ee.Classifier.smileRandomForest(10).train({
 features: trainingSample,
 classProperty: label,
 inputProperties: img.bandNames()
});

// Get information about the trained classifier.
print('Results of trained classifier', trainedClassifier.explain());

// Get a confusion matrix and overall accuracy for the training sample.
var trainAccuracy = trainedClassifier.confusionMatrix();
print('Training error matrix', trainAccuracy);
print('Training overall accuracy', trainAccuracy.accuracy());
```

# 13 – Clasificación Supervisada Matriz de Confusión – Random Forest

```

// Get a confusion matrix and overall accuracy for the validation sample.
validationSample = validationSample.classify(trainedClassifier);
var validationAccuracy = validationSample.errorMatrix(label, 'classification');
print('Validation error matrix', validationAccuracy);
print('Validation accuracy', validationAccuracy.accuracy());

// Classify the reflectance image from the trained classifier.
var imgClassified = img.classify(trainedClassifier);

// Add the layers to the map.
var classVis = {
 min: 0,
 max: 10,
 palette: ['006400', 'ffbb22', 'ffff4c', 'f096ff', 'fa0000', 'b4b4b4',
 'f0f0f0', '0064c8', '0096a0', '00cf75', 'fae6a0']
};
Map.setCenter(-122.184, 37.796, 12);
Map.addLayer(img, {bands: ['B11', 'B8', 'B3'], min: 100, max: 3500}, 'img');
Map.addLayer(lc, classVis, 'lc');
Map.addLayer(imgClassified, classVis, 'Classified');
Map.addLayer(roi, {color: 'white'}, 'ROI', false, 0.5);
Map.addLayer(trainingSample, {color: 'black'}, 'Training sample', false);
Map.addLayer(validationSample, {color: 'white'}, 'Validation sample', false);

```

# 13 – Clasificación Supervisada Matriz de Confusión – Random Forest

Google Earth Engine

Search places and datasets...

ee-cesarivanalvarezmendoza

Scripts Docs Assets

13 - Clasificacion RF y Matriz de Co... Get Link Save Run Reset Apps

Training error matrix JSON

List (9 elements) JSON

Training overall accuracy JSON

0.9696969696969697

Validation error matrix JSON

List (9 elements) JSON

0: [12,0,1,0,1,1,0,0,1]

13 - Clasificacion RF y Matriz de Co... 1 - Introducción JavaScript 10 - Reductores 11 - Series de Tiempo 12 - Clasificación SVM 13 - Clasificación RF y Mat... 2 - Visualización Imagen Landsat 9 3 - Filtrar imagen Sentinel-2

Inspector Console Tasks

```
53 // Classify the reflectance image from the trained classifier.
54 var imgClassified = img.classify(trainedClassifier);
55
56 // Add the layers to the map.
57 var classVis = {
58 min: 0,
59 max: 10,
60 palette: ['006400', 'ffbb22', 'ffff4c', 'f096ff', 'fa0000', 'b4b4b4',
61 'f0f0f0', '0064c8', '0096a0', '00cf75', 'fae6a0']
62 };
63 Map.setCenter(-122.184, 37.796, 12);
64 Map.addLayer(img, {bands: ['B11', 'B8', 'B3'], min: 100, max: 3500}, 'img');
```

Layers Map Satélite

Combinaciones de teclas Datos de mapas ©2022 Google 10 km Términos de uso

80

# Matriz de Confusión – Accuracy and Kappa Coefficients

|           |   | Truth          |                |
|-----------|---|----------------|----------------|
|           |   | P              | N              |
| Predicted | P | TP             | FP<br>(Type 1) |
|           | N | FN<br>(Type 2) | TN             |

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

- ■ ■
- ■ ■
- ■ ■
- ■ ■
- ■ ■
- ■ ■
- ■ ■
- ■ ■

# Matriz de Confusión – Accuracy and Kappa Coefficients

|           |   | Truth |             |
|-----------|---|-------|-------------|
|           |   | P     | N           |
| Predicted | P | 0     | 0           |
|           | N | 10    | $10^9 - 10$ |

**Out of 1 Billion People there  
are 10 terrorists**

$$\begin{aligned}\text{Accuracy} &= 10^9 - 10 / 10^9 \\&= 1 - 10^{-8} \\&= 0.99999 \\&\text{or } 99.9999\%\end{aligned}$$

- A 4x4 grid of black squares arranged in four rows and four columns, centered on a white background.

# Matriz de Confusión – Accuracy and Kappa Coefficients

|           |   | Truth |             |
|-----------|---|-------|-------------|
|           |   | P     | N           |
| Predicted | P | 0     | 0           |
|           | N | 10    | $10^9 - 10$ |

**Out of 1 Billion People there  
are 10 terrorists**

$$\begin{aligned}\text{Accuracy} &= 10^9 - 10 / 10^9 \\&= 1 - 10^{-8} \\&= 0.99999 \\&\text{or } 99.9999\%\end{aligned}$$

- A 3x3 grid of nine black squares arranged in three rows and three columns, centered on a white background.

# Matriz de Confusión – Accuracy and Kappa Coefficients

Classification results are evaluated based on the following metrics

- Overall Accuracy: How many samples were classified correctly.
- Producer's Accuracy: How well did the classification predict each class.
- Consumer's Accuracy (Reliability): How reliable is the prediction in each class.
- Kappa Coefficient: How well the classification performed as compared to random assignment.

|                       |   | Classification (Predicted) |      |      |      | Producer's Accuracy |
|-----------------------|---|----------------------------|------|------|------|---------------------|
|                       |   | 0                          | 1    | 2    | 3    |                     |
| Ground Truth (Actual) | 0 | 41                         | 4    | 0    | 0    | 0.91                |
|                       | 1 | 3                          | 43   | 0    | 0    | 0.93                |
|                       | 2 | 0                          | 0    | 48   | 3    | 0.94                |
|                       | 3 | 0                          | 0    | 0    | 37   | 1.00                |
|                       |   | 0.93                       | 0.91 | 1.00 | 0.93 | 0.94                |
| Consumer's Accuracy   |   |                            |      |      |      | Overall Accuracy    |

# Matriz de Confusión – Accuracy and Kappa Coefficients

|                               |              | Reference test information |        |          |        | Row total | User's Accuracy |
|-------------------------------|--------------|----------------------------|--------|----------|--------|-----------|-----------------|
|                               |              | Class                      | Road   | Building | Green  | Bare      |                 |
| Remote sensing classification | Road         | 101                        | 0      | 25       | 20     | 146       | 69.18%          |
|                               | Building     | 0                          | 128    | 0        | 17     | 145       | 88.28%          |
|                               | Green        | 10                         | 0      | 104      | 1      | 115       | 90.43%          |
|                               | Bare         | 2                          | 4      | 2        | 105    | 113       | 92.92%          |
|                               | Column total | 113                        | 132    | 131      | 143    | 519       |                 |
| Producer's accuracy           |              | 89.38%                     | 96.97% | 79.39%   | 73.43% |           |                 |

Overall accuracy = 84.4%, Kappa coefficient: 0.825.

- ▪ ▪ ▪ ▪

Muito  
obrigado  
pela sua  
atenção



César Iván Alvarez Mendoza  
[calvarezm@ups.edu.ec](mailto:calvarezm@ups.edu.ec)