

OBJETIVO.

Establecer los procedimientos para la clasificación de coberturas de bosque y sabana del Proyecto de Orinoco2 a través de la interfaz de Google Earth Engine, en tanto la metodología de clasificación se toma con base metodológica establecida por el IDEAM y su ajuste al análisis de coberturas establecido por la National Aeronautics and Space Administration (NASA) más procesos propios de ajuste, complemento y análisis de reconocimiento de la región de incidencia del proyecto Orinoco2.

ALCANCE

Este instructivo tiene como alcance la identificación de coberturas de bosque y sabana con imágenes satelitales tanto Landsat de mediana a alta resolución a una escala lo que se busca es que con este código prototipo de clasificación de coberturas de bosque y sabana se pueda reconocer las coberturas; es necesario establecer que este análisis, lo que se busca es encontrar y comprobar que estas zonas tienen presencia de bosque y sabana con el fin de que este se adopte a la realidad, este procedimiento o instructivo es de fácil uso esto es con el fin que se pueda aplicar para todos los funcionarios del área geoespacial de la Fundación Cataruben.

GENERALIDADES

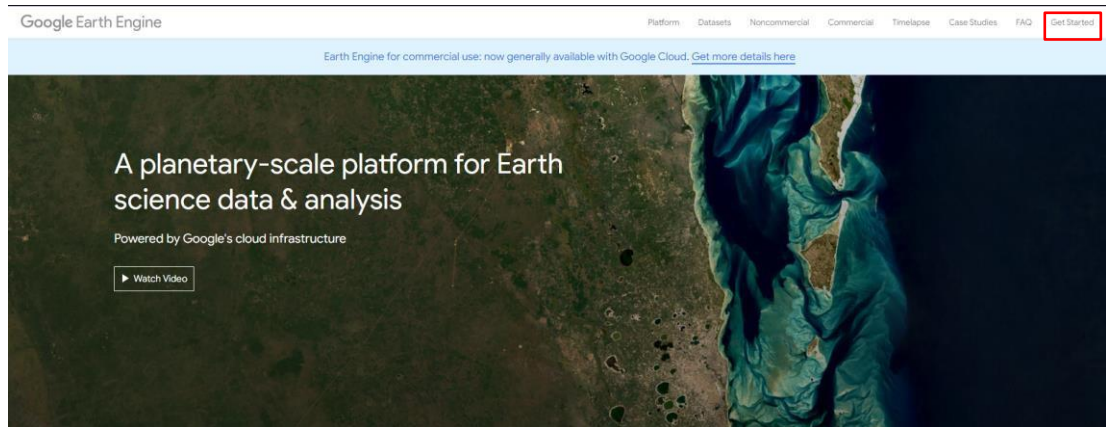
Desde el 2019 la National Aeronautics and Space Administration (NASA) y el Instituto de Hidrología, Meteorología y Estudios Ambientales (IDEAM), ha llevado a cabo la consolidación, entrenamiento y almacenamiento de información de imágenes satelitales, y shapes de coberturas nacionales de corine land cover con el fin de aportar insumos base que permite reconocer en temporalidad e histórico y cambios de coberturas en Colombia, dando estos datos que se generan a nivel nacional de sirvan como insumo base para distintas actividades.

Al aplicar la metodología de entrenamiento SAR para el análisis de coberturas, nos permite generar un modelo de coberturas naturales acoplado a la metodología de corine land cover, esto para poder analizar, describir y definir aquellas coberturas que han sido mantenidas o conservadas largo del tiempo; y con esto ser interpretadas a partir de imágenes satelitales Landsat.

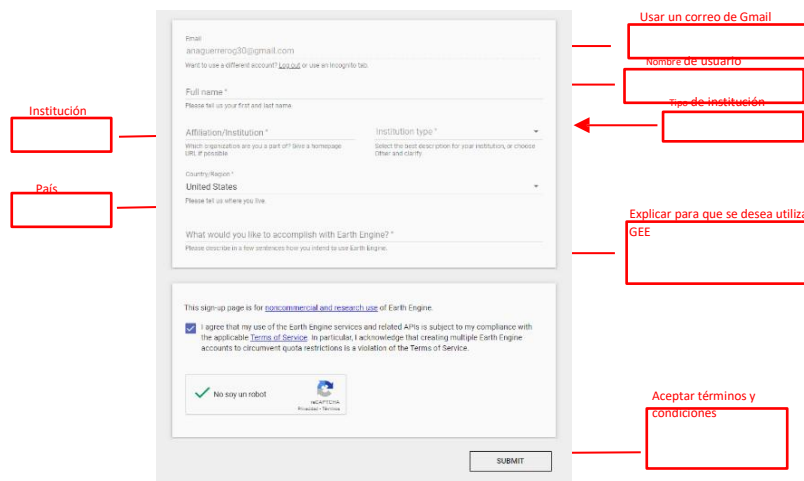
PROCEDIMIENTO

1. Registro de Google Earth Engine (GEE)

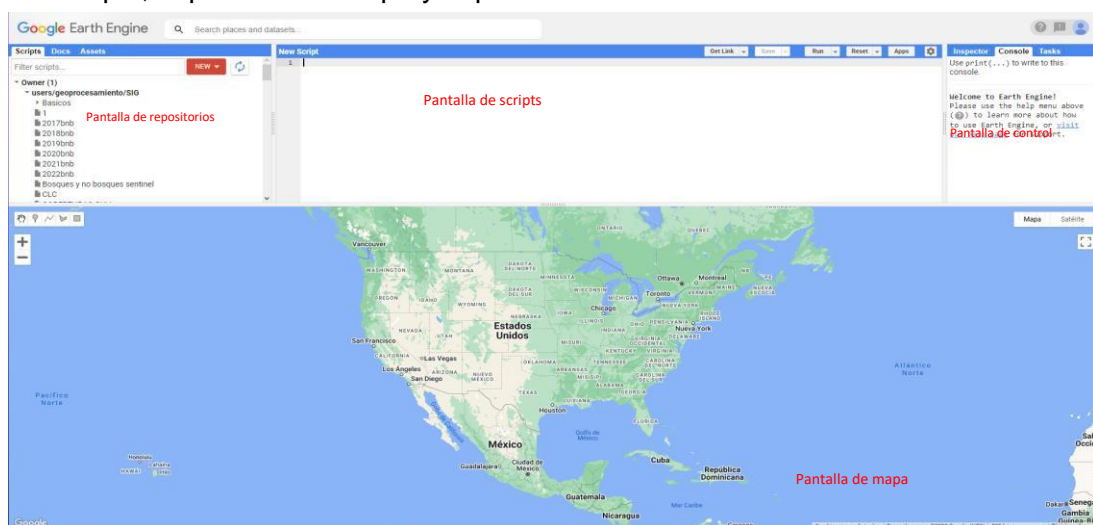
- 1.1 Ingresamos al siguiente enlace <https://earthengine.google.com>, siguiendo con esto le damos en la parte superior derecha le damos get started; tal como se muestra en la figura que se muestra a continuación.



1.2 Se ingresan los datos del formulario que le exige el programa; Cabe resaltar que para utilizar GEE se requiere de una cuenta de Google si ya la tiene creada salte al paso #4.



1.3 Una vez registrado, esperar a que le llegue un correo de confirmación por parte de Google; una vez confirmado, se procede a ingresar a la interfaz gráfica de GEE. Se observan cuatro pantallas de izquierda a derecha: pantalla de repositorios, la pantalla de scripts, la pantalla de mapa y la pantalla de control.



2) Descripción del entorno de google earth engine

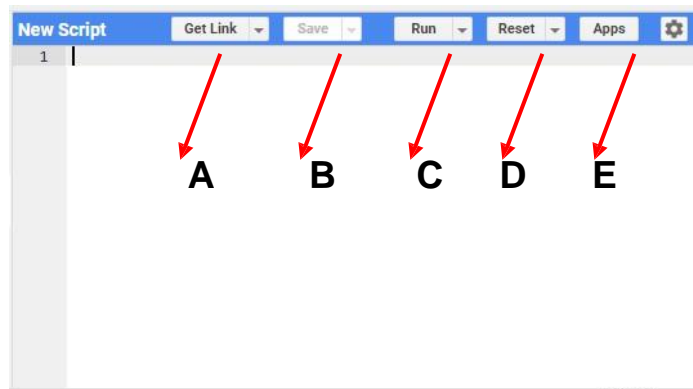
2.1 Pantalla de repositorios

En esta pantalla se crean los archivos o biblioteca de los códigos; para ello seleccionamos NEW y elegimos file para crear el archivo, si no tiene creado un repositorio y carpeta debe crear una a una para que al momento de guardar sus códigos queden guardados



2.2 Pantalla de scripts

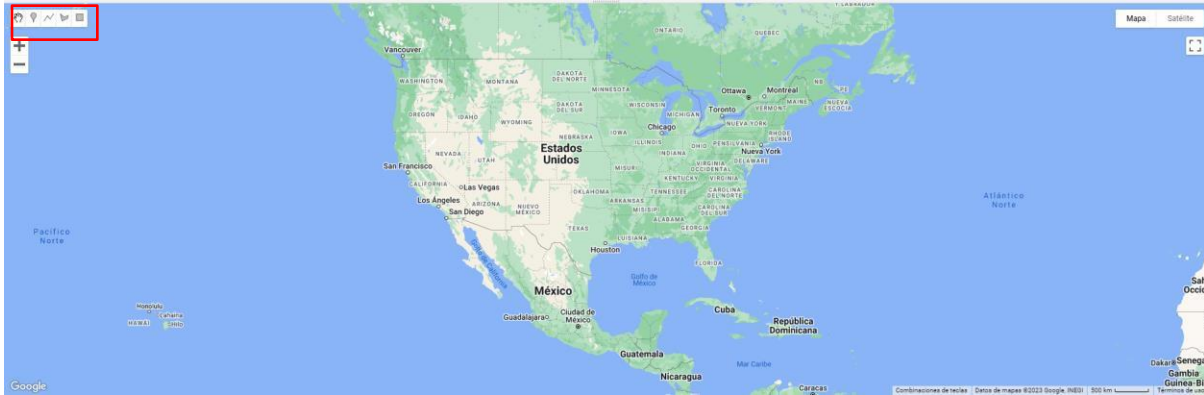
En esta pantalla se muestran todos los procesos ejecutados por los usuarios al ejecutar el código.



- A) Permite compartir el código por medio de un enlace.
- B) Permite guardar el código de cada vez que lo modifiques.
- C) Ejecuta todo el código que tienes en esta pantalla.
- D) Permite borrar o parar la ejecución del código.
- E) Permite crear aplicaciones a partir del código realizado.

2.3 Pantalla de Mapa

En esta pantalla es donde se visualiza lo indicado en la pantalla scripts, claramente hay que indicar con la función Map.addLayer para que esta se pueda visualizar.



Lo que se evidencia en el recuadro rojo son todos los tipos de figuras que se pueda utilizar para crear dentro de la pantalla de mapa.

2.4 Pantalla de control

En esta pantalla contiene 3 procesos que nos da las salidas al momento de realizar la ejecución del código; dentro de esta pantalla tenemos:

2.4.1 Consola

Nos permite visualizar si el código presenta algún error o algún fallo en la sintaxis.

2.4.2 Inspector

Permite consultar los valores que tienen los mapas al momento de ejecutar el código.

2.4.3 Tasks

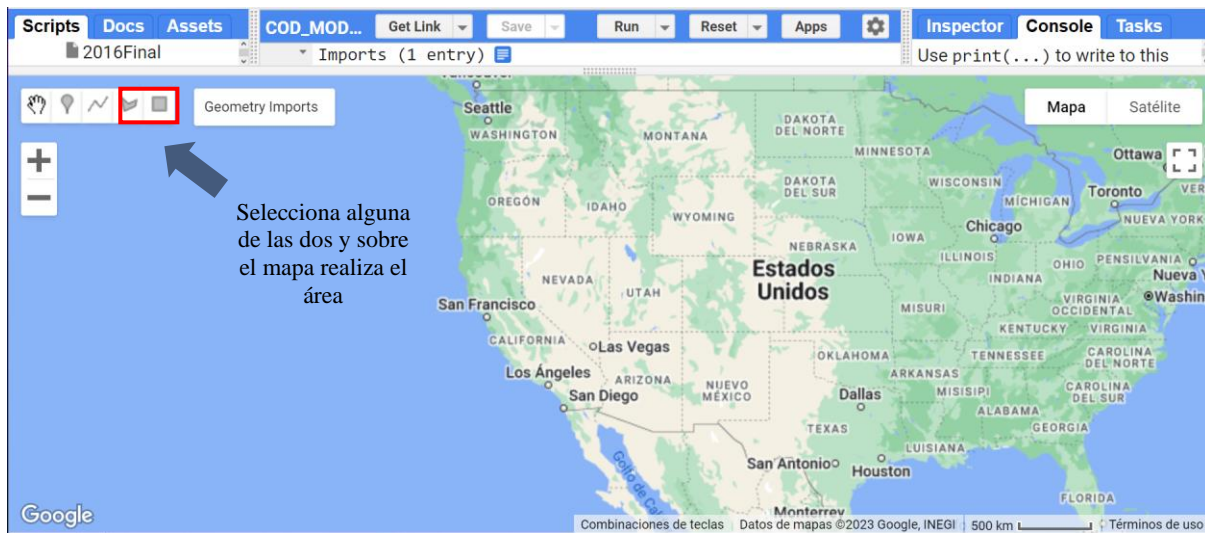
Esta nos da las salidas gráficas que se tiene en el código cuando se va a exportar o cuando se añade un diagrama que tiene por determinada el GEE.

3) Modelo clasificacion de coberturas

3.1 Selecciona, dibuja importa el área a analizar

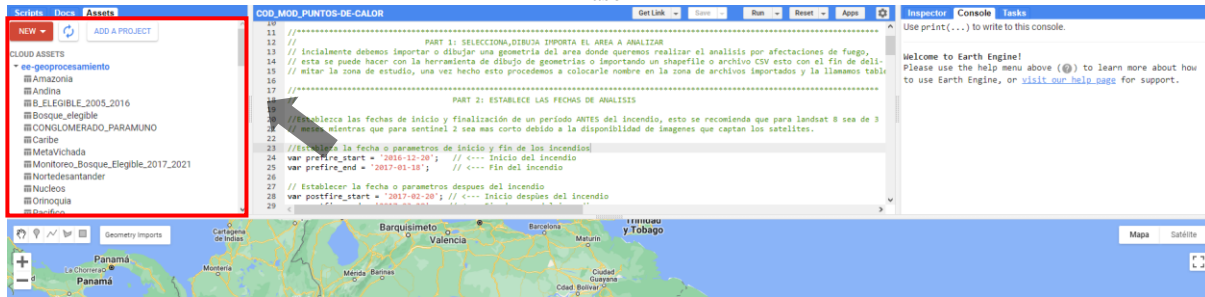
Inicialmente se procede a cargar el polígono o área a estudiar o analizar y lo llamamos table.

3.1.1 Creación de geometría con herramienta de Google earth engine



3.1.2 Importación de shapefile o CSV

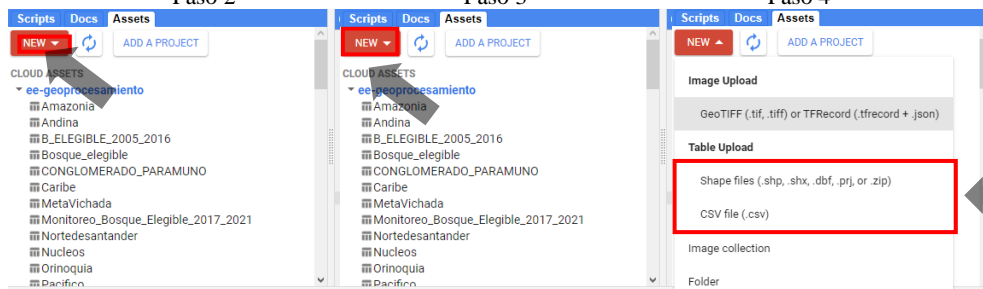
Paso 1



Paso 2

Paso 3

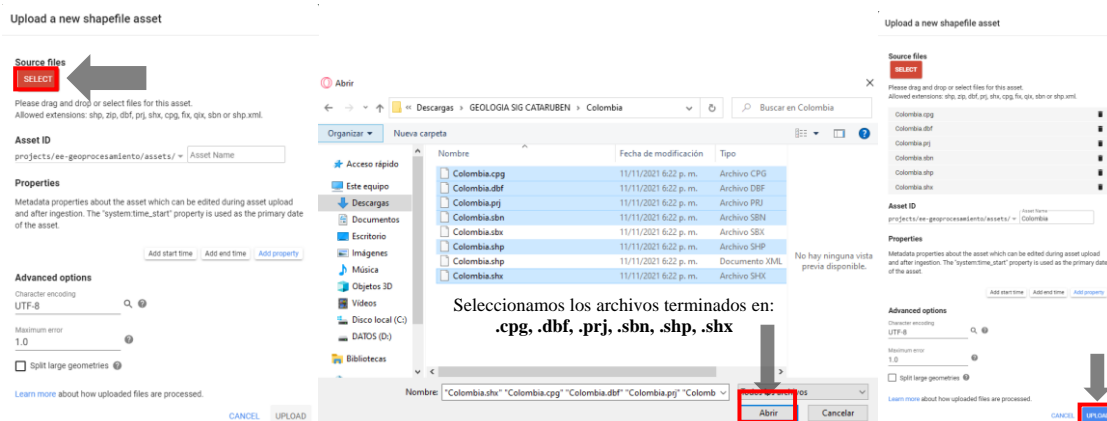
Paso 4



Paso 5

Paso 6

Paso 7



3.2 Definir los pacth y row de la zona

Una vez creado el poligono se procede a importar los pacth y row del satellite landsat y de las areas donde se encuentran los departamentos del area de interes

```
// Define las rutas y filas
var rutas = [4,4,5,5,5,6,6,6,7,7,7,8,8,8];
var filas = [56,57,56,57,58,56,57,58,57,58,59,57,58,59];
```

3.3 Definir los datos de entrada del usuario

En este paso se procede a importar las fechas de inicio y final de las colecciones de imagenes, el satellite esto con el fin de que se tengan a disponibilidad del analisis estos insumos, para ellos se procede a realizar el siguiente codigo.

```
// Define el rango de fechas
var fechaInicio = 'AAAA-MM-DD';
var fechaFin = 'AAAA-MM-DD';

// Define la colección
var coleccion = ee.ImageCollection('LANDSAT/LC08/C01/T1'); //seleccion de satellite

// Crea una ImageCollection vacía para almacenar las imágenes sin nubes
var coleccionSinNubes = ee.ImageCollection([]);
```

3.4 iteracion de imagenes por patch y row

Lo que hace es itera sobre una lista de rutas y filas y filtra una colección de imágenes satelitales en Google Earth Engine (GEE) en función de la ruta, fila y un rango de fechas especificado.

```
// Itera sobre cada par de ruta y fila
for (var i = 0; i < rutas.length; i++) {
  var ruta = rutas[i];
  var fila = filas[i];

  // Filtra la colección por ruta, fila y rango de fechas
  var coleccionFiltrada = coleccion
    .filter(ee.Filter.eq('WRS_PATH', ruta))
    .filter(ee.Filter.eq('WRS_ROW', fila))
    .filter(ee.Filter.date(fechaInicio, fechaFin));
```

3.5 Filtrado de toda la coleccion de imagenes

```
// Filtra la colección por cobertura de nubes
var imagenesSinNubes = coleccionFiltrada
  .filterMetadata('CLOUD_COVER', 'less_than', 10); // Ajusta el umbral de cobertura de nubes según sea necesario

// Agrega las imágenes sin nubes a la colección sin nubes
coleccionSinNubes = coleccionSinNubes.merge(imagenesSinNubes);

// Obtiene la primera imagen de la colección
var imagen = imagenesSinNubes.first();

// Definición de los parámetros de visualización
var parametrosVis = {
  bands: ['B4', 'B3', 'B2'], // Rojo, Verde, Azul
  min: 9.58005620496857,
  max: 36.533899965735
};

// Muestra la imagen con los parámetros de visualización
var vistaimagen = imagen.visualize(parametrosVis);
```


este fragmento de código filtra una colección de imágenes satelitales para seleccionar aquellas con una cobertura de nubes menor a 10, luego las agrega a una nueva colección `coleccionSinNubes`. Luego, se selecciona la primera imagen de esa colección y se define un conjunto de parámetros de visualización para mostrarla. Finalmente, se crea una vista de la imagen utilizando los parámetros de visualización definidos.

3.6 Visualizacion de las imagenes filtradas

En el siguiente codigo agrega las imagenes por individual ya seleccionada por fecha, nubosidad y prioridad para ser procesada al mapa de GEE con un nombre identificativo y un nivel de transparencia determinado, lo que permite visualizar la imagen en el entorno de GEE.

// Muestra la imagen en el mapa de Earth Engine

```
Map.addLayer(vistaimagen, {}, 'landsat ' + 'ruta: ' + ruta + ', fila: ' + fila, 0);
```



3.7 Imprimir la disponibilidad o cantidad de imagenes por cuadro

// Get the number of images in the collection

```
var num_images = cloud_free_images.size();
```

// Print the number of images

```
print('Number of images for path ' + path + ' and row ' + row + ': ' + num_images.getInfo());
```

```
}
```

Este fragmento de código calcula y muestra el número de imágenes disponibles en la colección `cloud_free_images` para una ruta y fila específicas. La información se imprime en la consola de GEE utilizando la función `print()`.

Inspector	Console	Tasks
Use print(...) to write to this console.		
Number of images for path 4 and row 56: 4		JSON
Number of images for path 4 and row 57: 2		JSON
Number of images for path 5 and row 56: 9		JSON
Number of images for path 5 and row 57: 6		JSON
Number of images for path 5 and row 58: 6		JSON
Number of images for path 6 and row 56: 4		JSON
Number of images for path 6 and row 57: 3		JSON
Number of images for path 6 and row 58: 1		JSON
Number of images for path 7 and row 57: 1		JSON
Number of images for path 7 and row 58: 1		JSON
Number of images for path 7 and row 59: 2		JSON
Number of images for path 8 and row 57: 0		JSON
Number of images for path 8 and row 58: 0		JSON
Number of images for path 8 and row 59: 1		JSON

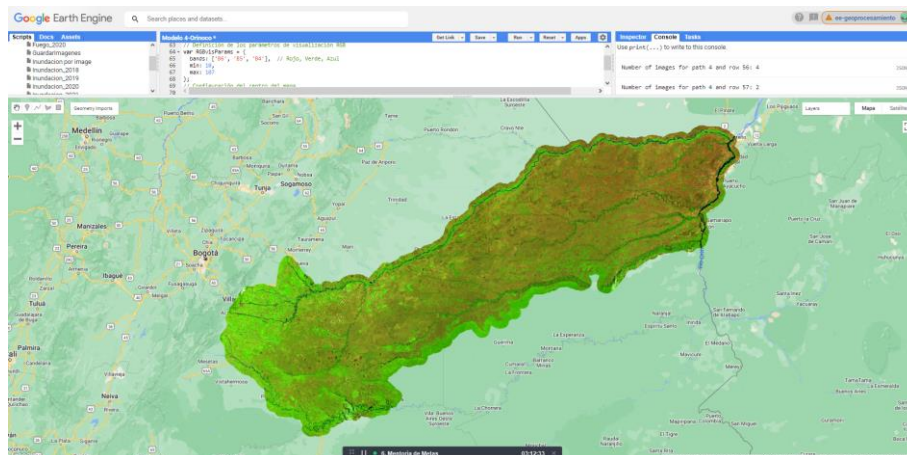
3.8 Crear un mosaico con todas las imagenes path y row

Este código crea un composite de una colección de imágenes sin nubes, selecciona las bandas relevantes, realiza un recorte del composite utilizando un polígono, y muestra el composite en el mapa de GEE utilizando una composición de color RGB.

```
// Create a simple composite of the cloud-free collection
var composite = ee.Algorithms.Landsat.simpleComposite(cloud_free_collection);

// Filtrado de bandas
composite = composite.select(['B2', 'B3', 'B4', 'B5', 'B6', 'B7']);
var composite = composite.clip(table);

// Definición de los parámetros de visualización RGB
var RGBvisParams = {
  bands: ['B6', 'B5', 'B4'], // Rojo, Verde, Azul
  min: 10,
  max: 107
};
// Configuración del centro del mapa
Map.setCenter(-71, 4, 7);
// Añadir el compuesto RGB al mapa
Map.addLayer(composite, RGBvisParams, 'Composite RGB');
```



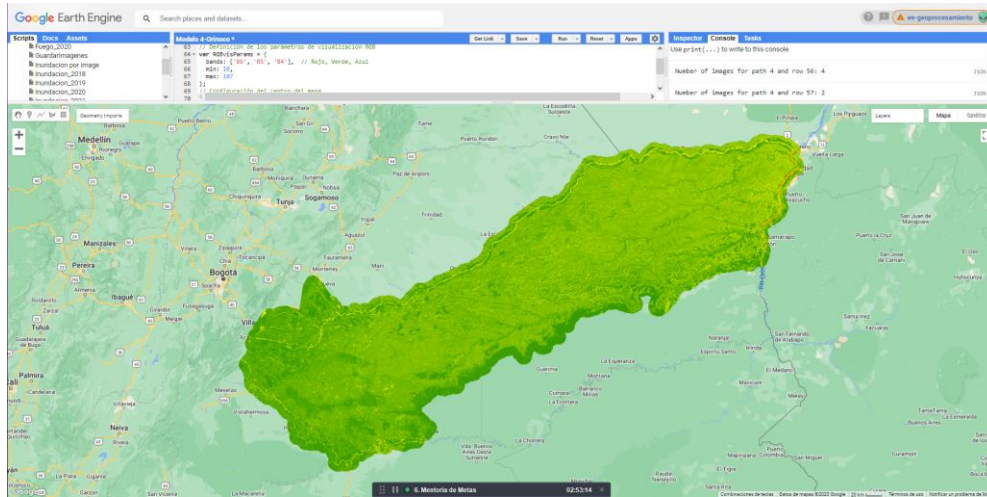
3.9 Creacion del indice NDVI o combinacion de bandas para apoyo

El siguiente código calcula el Índice de Vegetación de Diferencia Normalizada (NDVI) a partir de un composite de imágenes, define los parámetros de visualización para mostrar el NDVI en el mapa y agrega la capa del NDVI al mapa de GEE utilizando la paleta de colores definida. Esto permite visualizar y analizar la distribución de la vegetación en el área de interés.

```
// Cálculo del Índice de Vegetación de Diferencia Normalizada (NDVI)
var ndvi = composite.normalizedDifference(['B5', 'B4']).rename('NDVI');
// Definición de los parámetros de visualización para el NDVI
var ndviVisParams = {
  min: -1, // Valor mínimo del NDVI
  max: 1, // Valor máximo del NDVI
  palette: ['red', 'yellow', 'green'] // Paleta de colores para el NDVI
};
```



```
// Añadir la capa de NDVI al mapa
Map.addLayer(ndvi, ndviVisParams, 'NDVI');
```



3.10 Elección de puntos de control para coberturas

El siguiente código proporcionado realiza un muestreo aleatorio de una imagen compuesta utilizando una colección de polígonos que representan las 8 diferentes clases de coberturas en este caso se seleccionan las coberturas de 1.bosque, 2.herbazal, 3.arbustal, 4.transicional, 5.agua, 6.suelodescubierto, 7.cultivo y 8.atmosfera esto se identifica sobre la imagen de composición RGB y sobre el terreno. Luego, cuenta el número de puntos para cada clase de cobertura en la muestra aleatoria y muestra los resultados en la consola de GEE.

```
// Concatenación de las características
var poligonos = bosque.merge(herbazal).merge(arbustal).merge(transicional).merge(agua).merge(suelodescubierto).merge(cultivo).merge(atmosfera);

//.merge(herbazal).merge(agua).merge(agricola).merge(artificial).merge(nubes).merge(sombras);
// Muestreo de la imagen compuesta utilizando la colección de polígonos
var puntos = composite.sampleRegions({
  collection: poligonos,
  properties: ["landcover"],
  scale: 30
});

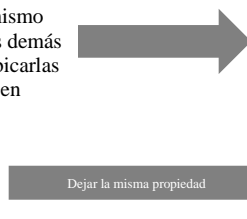
// Obtención de una muestra aleatoria de la colección de características, limitada a 5000 puntos
var puntos_final = puntos.randomColumn('random', 123)
  .sort('random')
  .limit(5000);

// Obtención de los valores únicos de la propiedad "landcover"
var uniqueLandcover = puntos_final.aggregate_array('landcover').distinct();

// Iteración sobre cada valor de landcover y conteo del número de puntos
uniqueLandcover.evaluate(function(landcoverValues) {
  landcoverValues.forEach(function(landcover) {
    var count = puntos_final.filter(ee.Filter.eq('landcover', landcover)).size();
    print('Cantidad de puntos para landcover ' + landcover + ':', count);
  });
});

print('Total de puntos', puntos_final); //muestra en la consola la cantidad puntos tomados para el analisis
```

Realizar el mismo
proceso con las demás
coberturas y ubicarlas
en la imagen



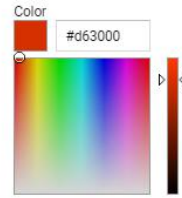
Configure geometry import

Name:

Import as:

Properties

Property	Value
landcover	1



Cambiar el valor por cada uno de los valores asignados a cada cobertura



Inspector Console Tasks		
Use print(...) to write to this console.		
Number of images for path 4 and row 56: 4		350N
Number of images for path 4 and row 57: 2		350N
Number of images for path 5 and row 56: 9		350N
Number of images for path 5 and row 57: 6		350N
Number of images for path 5 and row 58: 6		350N
Number of images for path 6 and row 56: 4		350N
Number of images for path 6 and row 57: 3		350N
Number of images for path 6 and row 58: 1		350N
Number of images for path 7 and row 57: 1		350N
Number of images for path 7 and row 58: 1		350N
Number of images for path 7 and row 59: 2		350N
Number of images for path 8 and row 57: 0		350N
Number of images for path 8 and row 58: 0		350N
Number of images for path 8 and row 59: 1		350N
Total de puntos		350N
FeatureCollection (5000 elements, 1 column)		350N

3.11 Estadística, validación y clasificación del método

Esta parte del código realiza una clasificación supervisada utilizando el clasificador de bosque aleatorio (Random Forest) en Earth Engine. Divide los datos en un conjunto de entrenamiento (80% de los puntos) y un conjunto de validación (20% de los puntos). Luego, entrena el clasificador utilizando el conjunto de entrenamiento y evalúa su rendimiento en el conjunto de entrenamiento y validación mediante la matriz de confusión y la precisión general.

```
// Añadir un campo de valor aleatorio a la muestra y utilizarlo para dividir aproximadamente el 80%
// de las características en un conjunto de entrenamiento y el 20% en un conjunto de validación.
var sample = puntos_final.randomColumn();
var trainingSample = sample.filter('random <= 0.8');
var validationSample = sample.filter('random > 0.8');

// Entrenamiento de un clasificador de bosque aleatorio de 100 árboles a partir de la muestra de entrenamiento.
var trainedClassifier = ee.Classifier.smileRandomForest(100).train({
  features: trainingSample,
  classProperty: 'landcover',
  inputProperties: composite.bandNames()
});

// Obtención de información sobre el clasificador entrenado.
print('Results of trained classifier', trainedClassifier.explain());

// Obtención de una matriz de confusión y precisión general para la muestra de entrenamiento.
var trainAccuracy = trainedClassifier.confusionMatrix();
print('Training error matrix', trainAccuracy);
print('Training overall accuracy', trainAccuracy.accuracy());

// Obtención de una matriz de confusión y precisión general para la muestra de validación.
validationSample = validationSample.classify(trainedClassifier);
var validationAccuracy = validationSample.errorMatrix('landcover', 'classification');
print('Validation error matrix', validationAccuracy);
print('Validation accuracy', validationAccuracy.accuracy());

// Clasificación de la imagen compuesta a partir del clasificador entrenado.
var imgClassified = composite.classify(trainedClassifier);

// Añadir las capas al mapa.
var classVis = {
  min: 1,
  max: 8,
  palette: ['green', 'yellow', 'white', 'grey', 'blue', 'pink', 'orange', 'black']
};

// Cálculo del Índice de Agua Normalizada Diferenciada (NDWI)
var ndwi = composite.normalizedDifference(['B3', 'B5']).rename('NDWI');

// Definición de los parámetros de visualización para el NDWI
var ndwiVisParams = {
  min: -1, // Valor mínimo del NDWI
  max: 1, // Valor máximo del NDWI
  palette: ['green', 'yellow', 'blue'] // Paleta de colores para el NDWI
};
```

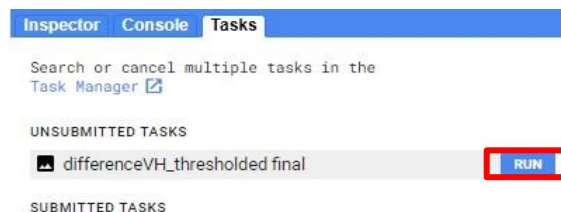
Number of images for path 6 and row 58: 1	JSON
Number of images for path 7 and row 57: 1	JSON
Number of images for path 7 and row 58: 1	JSON
Number of images for path 7 and row 59: 2	JSON
Number of images for path 8 and row 57: 0	JSON
Number of images for path 8 and row 58: 0	JSON
Number of images for path 8 and row 59: 1	JSON
Total de puntos	JSON
* FeatureCollection (5000 elements, 1 column)	JSON
Results of trained classifier	JSON
Object (5 properties)	JSON
Training error matrix	JSON
List (7 elements)	JSON
Training overall accuracy	JSON
0.9575056575308021	
Validation error matrix	JSON
List (7 elements)	JSON
Validation accuracy	JSON
0.9227761485826002	

3.12 Preparacion para exportacion de resultados

```
// Define los parámetros de exportación
var exportParams = {
  image: imgClassified,
  description: 'imagen_clasificada',
  folder: 'modelosOrinoco',
  scale: 30, // Especifica la escala de la imagen en metros
  maxPixels: 1e13,
  region: table // Especifica la región de interés para la exportación
};
```

```
// Exporta la imagen clasificada a Google Drive
Export.image.toDrive(exportParams);
```

Una vez le agregues este código anterior le das **RUN** y en la pantalla de scripts en la opción **tasks** procedemos a darle download y seleccionamos en qué parte del drive se va a guardar esta información y le cambiamos los nombres tal cual como aparece en las siguientes imágenes:



Le das Run en la imagen o capa que quieres descargar y te desplegara un cuadro donde te pedirá los datos algunos ya predeterminados y otros por definir tal como aparece en las señales (opcionales) de salida para que se realice la descarga, tal como se muestra a continuación:

Task: Initiate image export

Task name (no spaces) *
dNBR-Clasificacion

Coordinate Reference System (CRS)
EPSG:3857

Scale (m/px)
30

DRIVE CLOUD STORAGE EE ASSET

Drive folder
Drive folder name or blank for root

Filename *
dNBR-Clasificacion

File format *
GEO_TIFF

CANCEL RUN

Una vez dados los parámetros se procede a cerrar esa ventana y en la consola tasks aparecerá 4 opciones de que esta en proceso de descarga, descargado, cancelado y error en la descarga.

Inspector Console **Tasks**

Search or cancel multiple tasks in the Task Manager

UNSUBMITTED TASKS

dNBR-ColorverdaderoDespuesdelincendio RUN

SUBMITTED TASKS

dNBR-Clasificacion Descargando

myExportImageTask Descargado

dNBR-ColorverdaderoDespuesdelincendio Cancelado

dNBR-ColorverdaderoDespuesdelincendio Error en la descarga

Una vez descargado el archivo nos dirigimos al botón azul de su archivo descargado y le damos click este se desplegará y le damos en el botón (open to drive) y este lo dirigirá a la carpeta donde se encuentra el archivo descargado y de procede a descargarlo en su equipo para luego ser procesado en los softwares SIG.