Collect Earth Online Module 5

Training a Machine Learning Land Cover Model with Collect Earth Online

SERVIR Science Coordination Office
Curriculum Development Team
Micky Maganini
Contact: mrm0065@uah.edu

Prepared for “From Data to Geo-Information for Natural Resources Management” at ITC
December 17, 2022

Prerequisites & Requirements

Requirements

Before taking this module, you will need the following:

- A basic understanding of optical remote sensing
- A computer
- Connection to the internet
- A Collect Earth Online Account
- Google Earth Pro downloaded to your computer (free to install)
    - You can find instructions on how to install Google Earth Pro on page 4 of the following document. Click here to view instructions to download Google Earth Pro.
- Membership of the “ITC: NRM2” Collect Earth Online Institution

Prerequisites

Before taking this module, you should complete the previous Collect Earth Online Modules, which are listed below.

- Collect Earth Online Module 1 : Getting Started with Collect Earth Online). Click here to view CEO Module 1.
- Collect Earth Online Module 2: Base Imagery Sources in Collect Earth Online. Click here to view CEO Module 2. .
- Collect Earth Online Module 3: Creating your own project in Collect Earth Online. Click here to view CEO Module 3.
- Collect Earth Online Module 4: Collecting Data for a Collect Earth Online Project. Click here to view CEO Module 4.

Learning Objectives

By the end of this module, you will be able to...

- Understand how to perform a stratified random sample based on land cover using Google Earth Engine.
- Export a feature collection containing your sampling locations from Google Earth Engine to Google Drive.
- Import a CSV file containing your sampling locations from Google Drive into Collect Earth Online.
- Download collected land cover reference data from Collect Earth Online
- Understand how to take land cover reference data from Collect Earth Online, and use it to create a Random Forest machine learning algorithm to automatically create a land cover map.
- Understand how Collect Earth Online can be used to assess the accuracy of a land cover map that was generated with machine learning

What is a Machine Learning Model?

In this module, we will use Collect Earth Online to collect land cover reference data, which can be used to both train and validate a machine learning model. To train the model, we will first obtain labeled data, where we manually label what land cover class is present at a specific location. Then, we will feed these labels to the model, as well as some kind of imagery. In this case, we will train our model using PlanetScope optical data. When we feed the imagery into our model, the model will note the spectral characteristics that are associated with each label. For example, we know that water often appears as blue. So when we see a blue linear feature in satellite imagery, we are pretty sure that we are looking at a river. When we feed this labeled imagery to the model, it will note that the image labeled “river” has strong reflectance in the blue band, and weak reflectance in the red and green bands. If you feed enough of these labeled images into the model, it will “learn” that “rivers” reflect strongly in the blue band, and poorly in the red and green bands, even though it has no knowledge of what a river is. We will feed labeled imagery for all different kinds of land cover classes into our model. We can also feed ancillary data, such as NDVI or other spectral indices, to our model so that it has more information than just the spectral reflectance to recognize patterns in which land cover classes correlate to what characteristics of the data.

After we train our model, we will ask it to create a land cover map for our entire study area based on the patterns it identified in the training data. Once we have our final product, we would like to obtain some idea of how accurate our model is. To do this we will use a separate set of labeled reference data in order to test the performance of our model. Thus, after we collect our reference data (i.e. a manually labeled dataset containing the coordinates of a point and a label that describes the land cover class you observed at that point. This entire workflow can be implemented in Collect Earth Online and Google Earth Engine as follows:

1. Plan out your project by selecting your classification schema, sampling and response design, and imagery as shown in Collect Earth Online Module 3.
    a. Click here to view Collect Earth Online Module 3.
2. Create a table of locations for our reference dataset in Google Earth Engine
3. Export reference dataset containing locations to label from Google Earth Engine to Google Drive
4. Download reference dataset from Google Drive

5. Import reference dataset into Collect Earth Online
6. Label the imagery at each location in your dataset using Collect Earth Online
7. Download the labeled reference dataset from Collect Earth Online.
8. Upload the labeled reference dataset to Google Earth Engine as a Google Earth Engine Asset.
9. Split the labeled reference dataset into two datasets: 70% to be used for training the model, and 30% to be used for testing the model
10. Train the machine learning model using the training set.
    a. imagery and ancillary datasets that you used to label the reference data
    b. In this case we will use PlanetScope and Landsat imagery for our optical dat and NDVI to train the model
11. Assess the validity of our machine learning model using a confusion matrix.

Below we will see an example of how to conduct the start-to-finish workflow described above. This is an example of how you can create a land cover map for an entire country of Ghana using the Random Forest machine learning algorithm.

Step 1: Project Planning

Your first step in creating a land cover map is planning out your sampling and response design, which we learned about in Collect Earth Online Module 3. Click here to view CEO Module 3. You may recall there are several steps to planning an image interpretation project, including setting the objectives for your project, along with the end product you will create to achieve those objectives. Then, you must select the classification schema, imagery, and ancillary you will use to interpret the imagery. Next, you will select your sampling and response design. Below I will list my decisions for this example project.

Project Objectives

Understand the relative proportions of land cover in the country of Ghana in 2015.

End Product

In this project we will create a wall-to-wall land cover map of Ghana for 2015 using a Random Forest machine learning model.

Classification Schema

In this case, our classification schema will be the same one used by the CGLS dataset. CGLS is a global land cover product, so there is likely a substantial error present in it. In a real scenario, we would work with the end users and stakeholders to obtain an official land cover classification used by the government, but in our case the CGLS classification will suffice. In this case, there are 14 Land Cover Classes. You can see an example of the Classification Schema shown below.

Land Cover Class	**Description**
Shrubs	Woody perennial plants with persistent and woody stems and without any defined main stem less than 5m tall. The shrub foliage can be either evergreen or deciduous
Herbaceous Vegetation	Plants without persistent stems or shoots above ground and lacking definite firm structure. Tree and shrub cover is less than 10%
Agriculture	Lands covered with temporary crops followed by harvest and a bare soil period. Note that perennial woody crops will be classified as the appropriate forest or shrub land cover type.
Urban/Built-Up	Land covered by buildings and other man-made structures
Bare Ground	Lands with exposed soil, sand, or rocks and never has more than 10% vegetated cover during any time of the year.
Permanent Water	Lakes, reservoirs, and rivers. Can be either fresh or salt-water bodies.
Wetland	Lands with a permanent mistrue of water and herbaceous or woody vegetation. The vegetation can be present in either salt,

	brackish, or fresh water.
Closed forest – evergreen broad leaf	Tree Canopy > 70%, almost all broadleaf trees remain green year round. Canopy is never without green foliage.
Closed forest – deciduous broadleaf	Tree canopy >7-%, consists of seasonal broadleaf tree communities with an annual cycle of leaf-on and leaf-off periods.
Other Closed Forest	Closed forest not matching any of the other definitions
Open Forest – Evergreen broad leaf	Top layer is 15-70% trees and the second layer is a mix of shrubs and grassland; almost all broadleaf trees remain green year round. Canopy is never without green foliage.
Open Forest – Deciduous Broad Leaf	Top layer trees 15-70% and second layer – mixed of shrubs and grassland, consists of seasonal broadleaf tree communities with an annual cycle of leaf-on and leaf-off periods.
Other Open Forest	Open forest not matching any of the other definitions,
Ocean	Oceans, seas. Can be either fresh or salt-water bodies.

Imagery and Ancillary Data

We will use Planet NICFI and Landsat Imagery for this project. We will also use NDVI time series data, which is offered in Collect Earth Online using the Geodash functionality.

Sampling Design

Our sampling design will consist of a random stratified sample. The strata we will use to conduct this sample are the land cover classes as shown in the CGLS dataset. Within each land cover class, we will randomly distribute an even number of points. This will allow our

model to obtain plenty of examples of what each land cover class looks like, which is why random stratified samples are often employed to train and assess the accuracy of machine learning models. For this project, we will use a sample size of 100, meaning we will randomly distribute 100 points in each land cover class (as predicted by CGLS). Since we have 14 land cover classes in our classification schema, this will result in a total of 1400 points in our reference dataset.

Response Design

Our response design will consist of a 5m x 5m square. This is because this corresponds to the approximate spatial resolution of the Plantescope imagery, which will be our primary method for training the machine learning model.

Step 2: Create Reference Dataset

Here we will use a stratified random sample to create our reference dataset locations. We can do this within Google Earth Engine by defining our region of interest, then using the *ee.stratifiedSample()* function. This function takes the following parameters/arguments:

- band: this is the band to use for stratification
- numPoints: the number of points to assign in each strata

```
// Perform Stratified Sample
var strat_sample = CGLS_discrete.stratifiedSample({
  'numPoints': 100,                                    //Create 100 points within each stratum
})
```

In this case we are using “CGLS_discrete” which is the previous land cover product as our band argument. We will use 100 as our ‘numPoints’ argument. This will create a feature collection of 1400 points called “strat_sample” which will be our reference dataset. For my purposes, I will use a pre-existing land cover product known as the Copernicus Global Land Cover System as our strata. You can view the entire script I used to create a stratified random sample by clicking here.

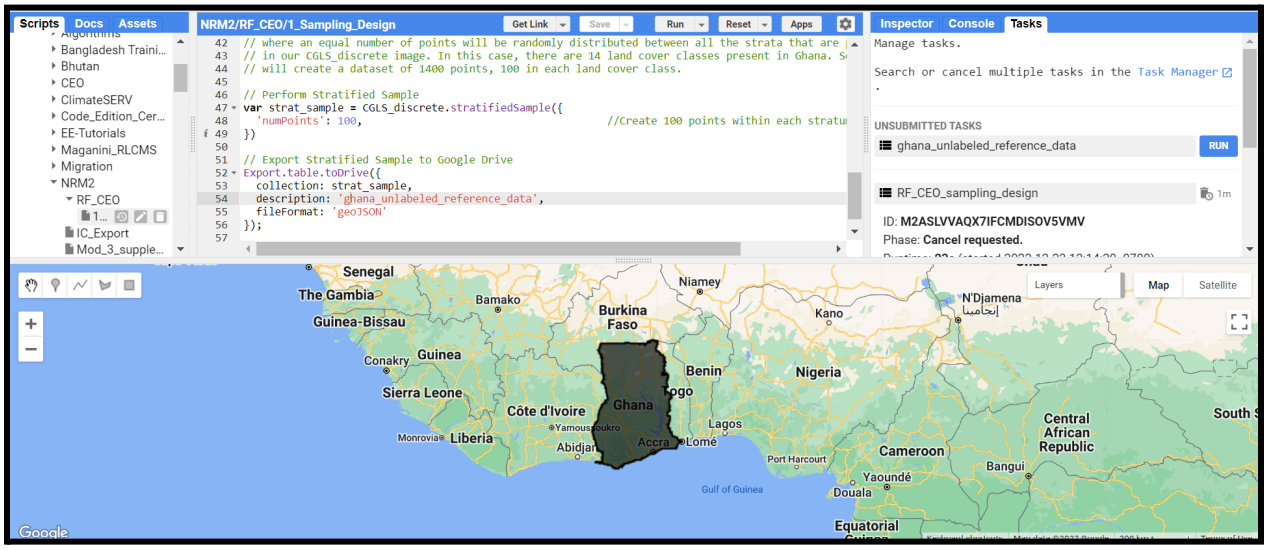
Step 3: Export Reference Dataset from GEE

Now we will export this dataset to Google Drive. We can do this using the *Export.table.toDrive()* function within GEE. This function takes the following arguments

- Collection: this is the dataset that we want to export to Google Drive. This will be our stratified sample we created in step 2.
- Description: this is the name we give to our dataset
- FileFormat: this is the file format we want to export our reference dataset in. In this case, we will export our reference dataset as a geoJSON file.

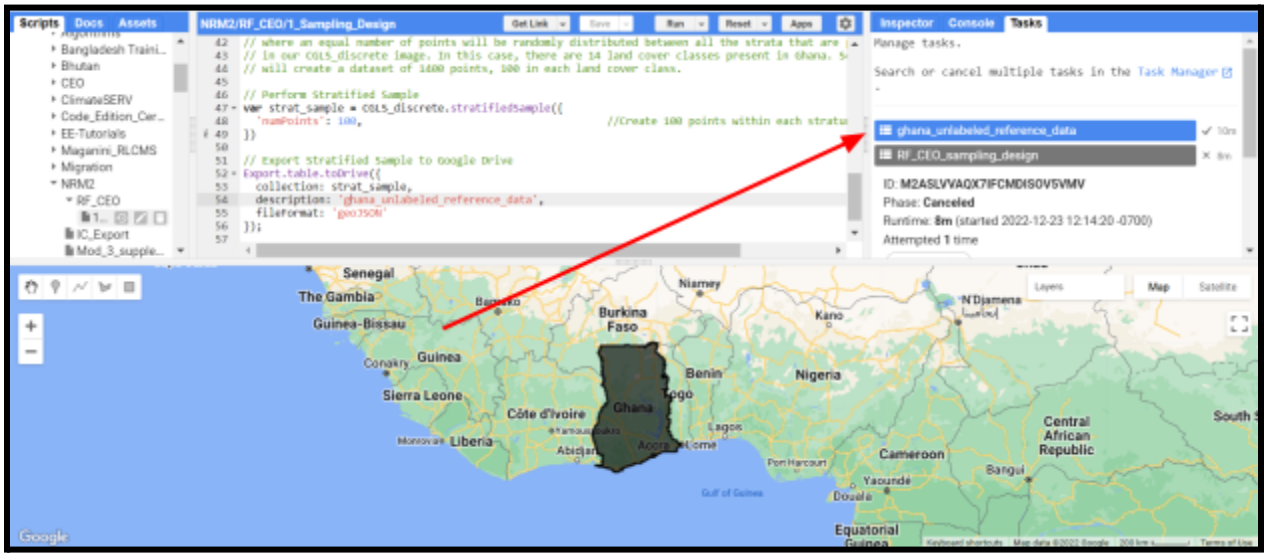
```
// Export Stratified Sample to Google Drive
Export.table.toDrive({
  collection: strat_sample,
  description: 'ghana_unlabeled_reference_data',
  fileFormat: 'geoJSON'
});
```

After running this script, a new task will appear on the right side of the screen. Click the **"tasks"** button to the right of the text that says **"Console".** You will then see your task under **"Unsubmitted Tasks"**, as seen in the screenshot below.



To export the task, we will click the blue Run button, then click **"Ok"**. Now a rotating gear will appear next to the task. This means our task is running.

It shouldn't take any longer than a minute for Google Earth Engine to export this simple dataset. When the task is complete, it will be highlighted in blue, as shown in the image below. If the task highlights in red, that means an error has occurred.



Once the task is complete, it will automatically appear in the Google Drive account that is associated with our Google Earth Engine account. Now my reference dataset will appear in Google Earth Engine with the file name we gave it in the "description" argument to the Export.table.toDrive method in GEE (in this case, it is "ghana_unlabeled_reference_data").

Step 4: Download and Convert Reference Dataset

Now we will download the dataset from our Google Drive. We can do this by right clicking the dataset in Google Drive, then clicking **"Download"**. After downloading our file as a .geojson, we can use this open source website to convert our .geojson file to a .csv file. We have to do this because Collect Earth Online requires you to upload our sampling locations as a .csv file.

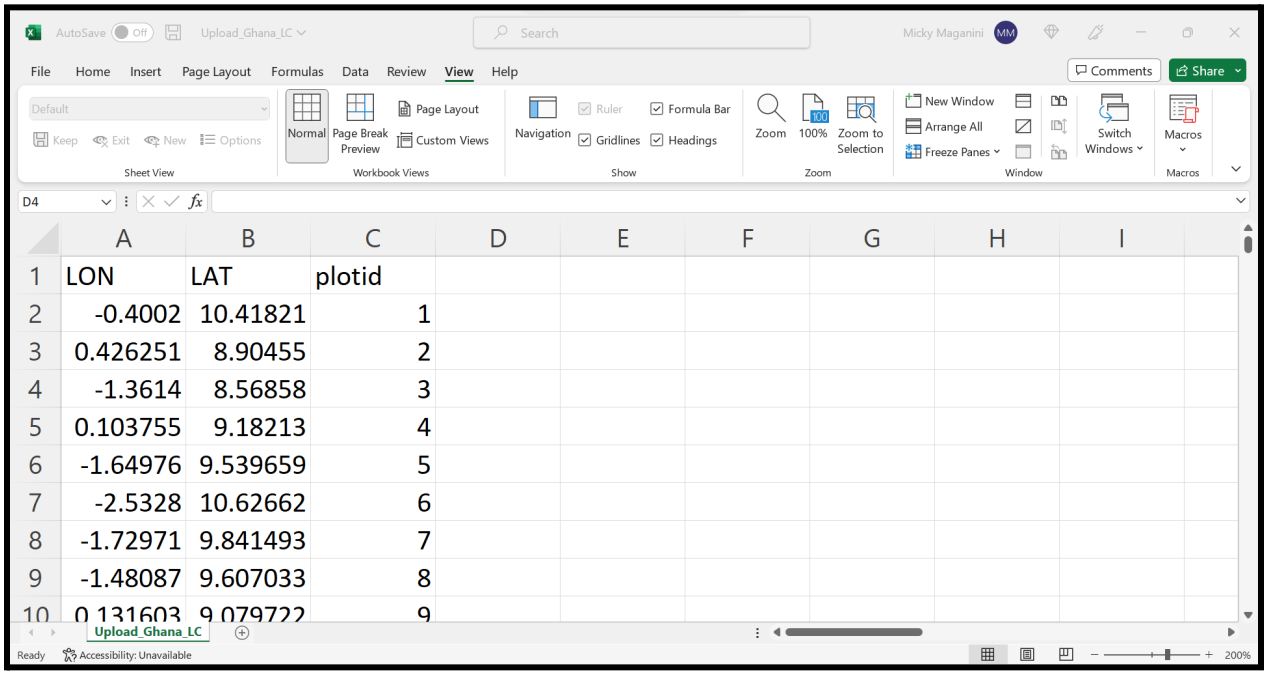
Step 5: Import reference dataset into CEO

Collect Earth Online requires us to upload our sampling design file as a .csv file with 3 columns, named PLOTID, LAT, and LON.

Our PLOTID (plot identification) column will contain positive integers starting with 1. This will help Collect Earth Online to give each location a name. Since our dataset doesn't currently have a column like this, we can create one. We can do this by typing "PLOTID" in the topmost row in a new column (i.e. in Cell C1). Under this cell we will type the letter 1. Then, in cell C3 wi will type "= C2 + 1". This will automatically enter "2" into cell C3. We can continue this pattern by clicking in the bottom right of cell C3 and dragging all the way to the bottom row of the dataset.

Our LAT column will contain the latitude of each location. Thus, we will change our previous column to have the header name LAT instead of latitude.

Our LON column will contain the longitude of each location. Thus, we will change our Column that has the name "longitude" to have the name "LON". Now, our dataset looks like the image below.



	A	B	C	D	E	F	G	H	I
1	LON	LAT	plotid						
2	-0.4002	10.41821	1						
3	0.426251	8.90455	2						
4	-1.3614	8.56858	3						
5	0.103755	9.18213	4						
6	-1.64976	9.539659	5						
7	-2.5328	10.62662	6						
8	-1.72971	9.841493	7						
9	-1.48087	9.607033	8						
10	0.131603	9.079722	9						

We are now ready to upload our reference dataset into Collect Earth Online to begin labeling the data. While creating our project as shown in module 4, when we get to the "sample design" step, we will select **"CSV"** file under the **"distribution"** option, and select our reference dataset file.

Step 6: Collect Data in CEO

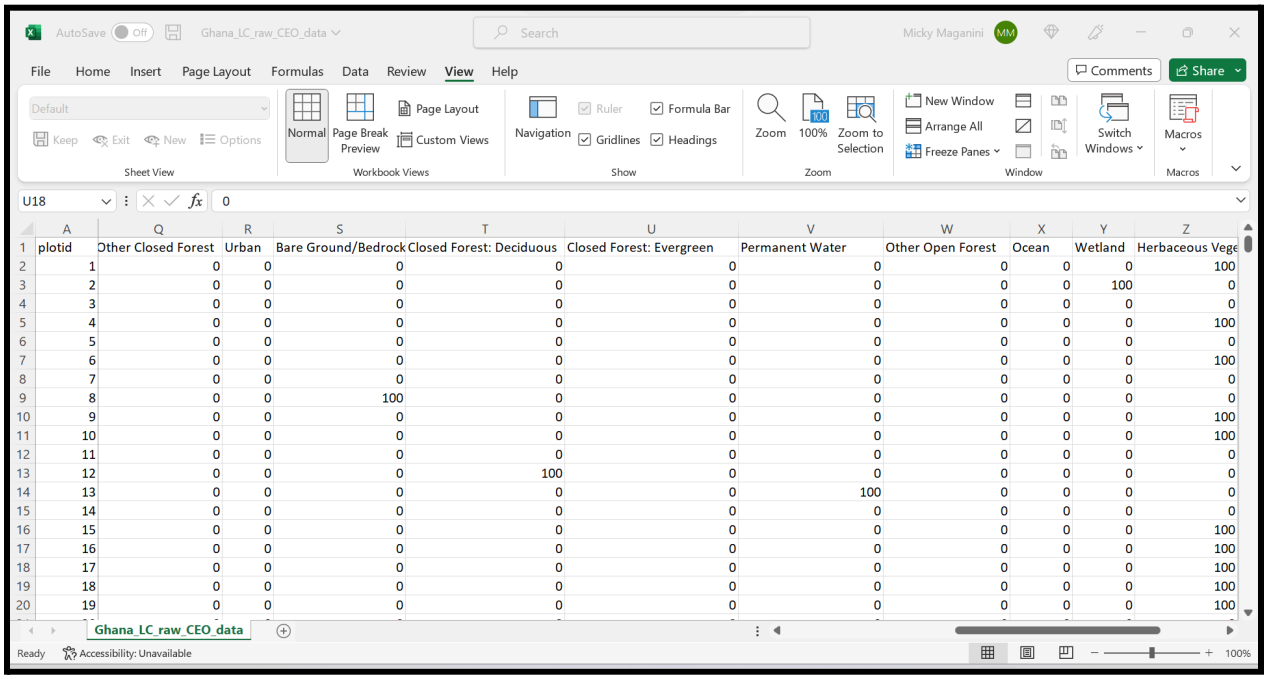
Our next step is to label the data by classifying each plot as one of our land cover classes in Collect Earth Online.

Step 7: Download Labeled Reference Data

To download the labeled reference dataset, we will visit our institution homepage, which looks like the following image. Then, we click the **"P"** button next to our project's name to download the labeled reference data we have just collected.

This will download the labeled reference data to my computer as a CSV file. Opening the file in Microsoft Excel, we can see that among other things, Collect Earth Online data will download with the PLOTID, latitude, longitude, and label for each reference data location as separate columns. There will also be a column for each land cover class specified in our response design. Since this study had 14 possible land cover classes, there will be 14

columns for each class, and each row will contain a percentage indicating the percent of the plot that was classified as that land cover class, as shown below. In this case, since there was only one sampling location per plot, each row will contain a ‘100’ for the land cover class that we interpreted for that plotid, and a ‘0’ for all land cover classes.



	A	Q	R	S	T	U	V	W	X	Y	Z
1	plotid	Other Closed Forest	Urban	Bare Ground/Bedrock	Closed Forest: Deciduous	Closed Forest: Evergreen	Permanent Water	Other Open Forest	Ocean	Wetland	Herbaceous Vege
2	1	0	0	0	0	0	0	0	0	0	100
3	2	0	0	0	0	0	0	0	0	100	0
4	3	0	0	0	0	0	0	0	0	0	0
5	4	0	0	0	0	0	0	0	0	0	100
6	5	0	0	0	0	0	0	0	0	0	0
7	6	0	0	0	0	0	0	0	0	0	100
8	7	0	0	0	0	0	0	0	0	0	0
9	8	0	0	100	0	0	0	0	0	0	0
10	9	0	0	0	0	0	0	0	0	0	100
11	10	0	0	0	0	0	0	0	0	0	100
12	11	0	0	0	0	0	0	0	0	0	0
13	12	0	0	0	100	0	0	0	0	0	0
14	13	0	0	0	0	0	100	0	0	0	0
15	14	0	0	0	0	0	0	0	0	0	0
16	15	0	0	0	0	0	0	0	0	0	100
17	16	0	0	0	0	0	0	0	0	0	100
18	17	0	0	0	0	0	0	0	0	0	100
19	18	0	0	0	0	0	0	0	0	0	100
20	19	0	0	0	0	0	0	0	0	0	100

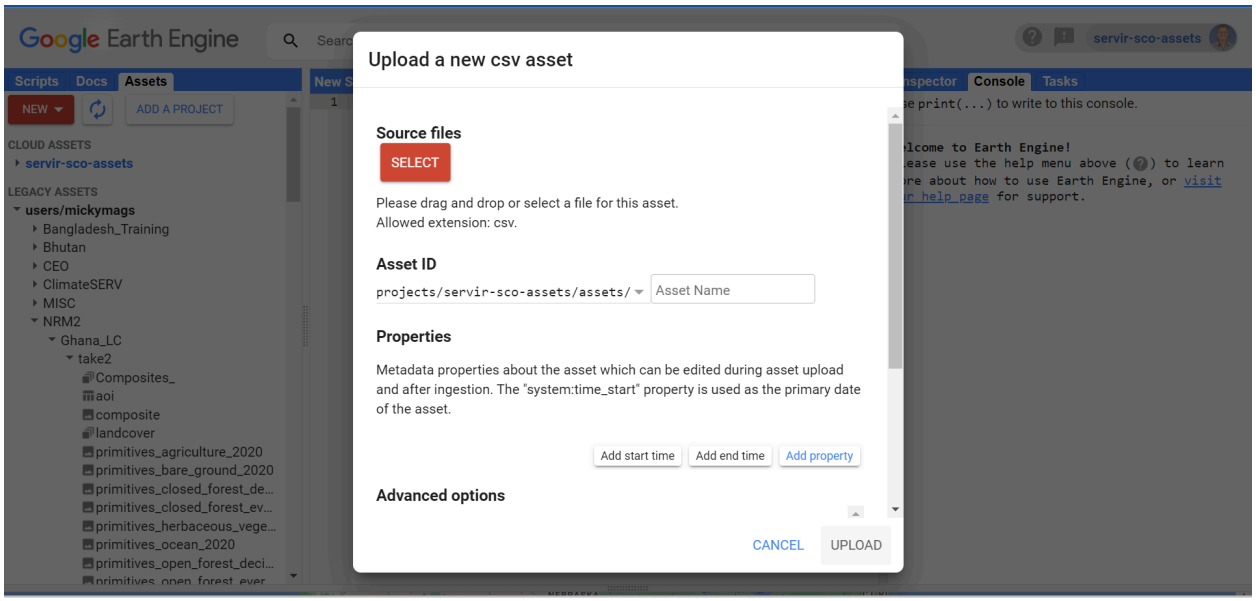
Using a python script, we can convert this dataset so that it contains one column called ‘land cover class’ that has a unique number value associated with each land cover class. This will make our lives easier when training the machine learning model in Google Earth Engine. The value associated with each land cover class can be seen in the following table. You can view the Google Colab Notebook by clicking here.

Land Cover Class	RLCMS Value
Oceans/Seas	1
Permanent Water	2
Closed Forest: Deciduous Broadleaf	3
Agriculture	4
Urban	5
Herbaceous Vegetation	6
Closed Forest: Evergreen Broadleaf	7
Open Forest: Deciduous Broadleaf	8
Shrubs	9
Bare Ground / Bedrock	10
Wetlands	11
Open Forest: Evergreen Broadleaf	12
Other Closed Forest	13
Other Open Forest	14

We are now ready to upload our reference dataset to Google Earth Engine.

Step 8: Upload the labeled reference dataset to GEE

Our next step is to upload our complete, labeled reference dataset to Google Earth Engine. We can do this by going to Google Earth Engine, then clicking the **“Assets”** tab on the left side of the screen. Then, we will click the red **“New”** button under the **“assets”** tab. This will open a dropdown menu. We will then select the **“CSV file”** button. This will bring up the following interface.



From here, we will click the red **“SELECT”** button, then select our reference dataset.

Step 9: Split Labeled Reference Dataset

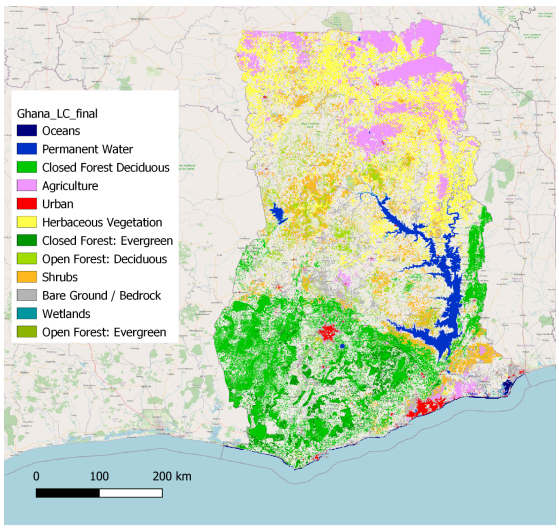
We will now split our data into a training set and a testing set. We will do this by allocating 70% of our reference data to be our training set, and the other 30% to be our testing set. We can do this in Google Earth Engine with the following function.

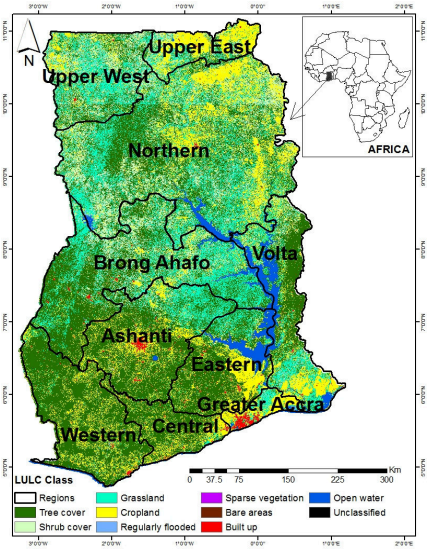
```
////////////////////////////////////////////////////////////////////////////////
//                      Step 1:  Split Reference Data                         //
////////////////////////////////////////////////////////////////////////////////


// Split the reference dataset, using 70% for training the model and 30% for testing

// Here is a function that splits data into training and validation groups. Set at 70% training/30% validation.

var splitData = function(data){
  var dict = {};
  var randomTpixels = data.randomColumn();
  var trainingData = randomTpixels.filter(ee.Filter.lt('random', 0.7));
  var valiData = randomTpixels.filter(ee.Filter.gte('random', 0.7));

  dict.training = trainingData;
  dict.validation = valiData;

  return dict;

};

// Run the function above on our reference data
var split_data = ee.Dictionary(splitData(ref_data));

var training_set = split_data.get('training')
var testing_set = split_data.get('validation')
```

Step 10: Train Machine Learning Model

Now we will use Earth Engine to train our machine learning model based on the training set of our reference data. We can use an off-the-shelf land cover machine learning algorithm such as SERVIR's Regional Land Cover Monitoring System, or RLCMS. RLCMS contains open source Google Earth Engine code that can be used to create land cover maps. In any machine learning land cover map, our first step is to create composites of our input imagery, then sampling our training data to create a final land cover map.

You can see the final model below,





Step 11: Assess Validity of your machine learning model

In order to assess the accuracy of our machine learning model, we will use our reference data that we created in Step 9. We can assess the performance of our model using confusion matrices, which contain the predicted class (i.e. the land cover class predicted by the model) on one axis, and the “true class” (i.e. the land cover class as classified by the interpreted in CEO) on the other axis. Since we have 14 land cover classes, our confusion matrix is a 14 x 14 grid, shown below

True Class

Predicted Class	Ocean	Permanent Water	Closed Forest: Deciduous	Agriculture	Urban	Herbaceous Vegetation	Closed Forest: Evergreen	Open Forest: Deciduous	Shrubs	Bare Ground	Wetlands	Open Forest: Evergreen	Other Closed Forest	Other Open Forest
Ocean	21	0	0	0	0	0	0	0	0	0	0	0	0	0
Permanent Water	0	55	0	0	0	1	0	0	0	0	0	0	0	0
Closed Forest: Deciduous	0	0	21	1	0	1	12	3	3	1	0	0	0	0
Agriculture	0	0	1	8	2	5	0	1	1	2	0	0	0	0
Urban	1	0	0	0	4	0	0	0	0	2	0	0	0	0
Herbaceous Vegetation	0	0	0	5	0	9	0	1	6	0	0	0	0	0
Closed Forest: Evergreen	0	0	3	0	0	0	1	0	0	0	0	0	1	0
Open Forest: Deciduous	0	0	2	1	0	2	0	3	1	2	0	0	0	0
Shrubs	0	0	2	0	0	4	0	2	3	0	0	0	0	0
Bare Ground / Bedrock	0	0	0	1	1	3	0	2	0	2	0	0	1	0
Wetlands	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Open Forest: Evergreen	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Other Closed Forest	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Other Open Forest	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The vertical axis of the matrix corresponds to the predicted class, and the horizontal axis corresponds to the actual class. The main diagonal of the matrix (in this case the top left and bottom right cell) shows the number of each class that was correctly identified by the model. That is, both the predicted class and the true class were the same for “X” number of points. The “off diagonal” shows the number of points were the classification of the model was different from the human interpretation of that point.

Data scientists often calculate multiple statistics to assess the performance of their model. You can learn more about confusion matrices and the statistical measurements associated with them by visiting the following page.