Case Study 2 (Part 1) - County-level Time-series NDVI

1 U.S. county shapefile

We have prepared the 2022 US shapefile (cb_2022_us_county_500k.zip) on Canvas. Please download it first.

1.1 Upload shapefile to assets

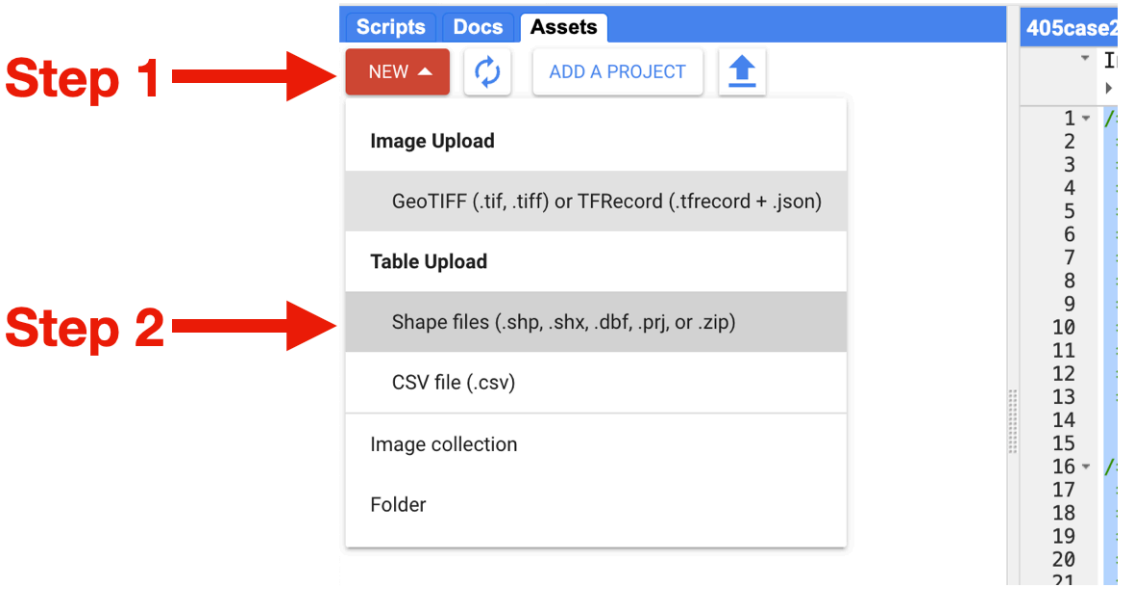

Fig. 1: This is in the top left corner of the code editor

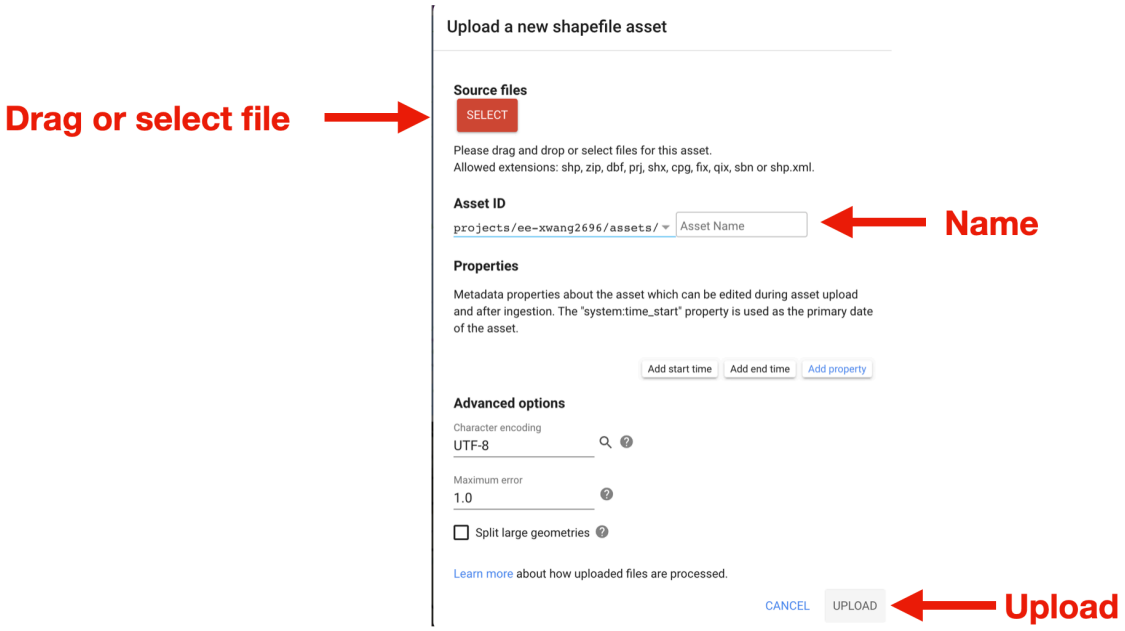

Fig. 2: After you complete the previous step, this will pop up

Fig. 3: This is in the top right corner of the code editor, you can check the upload progress in task manager

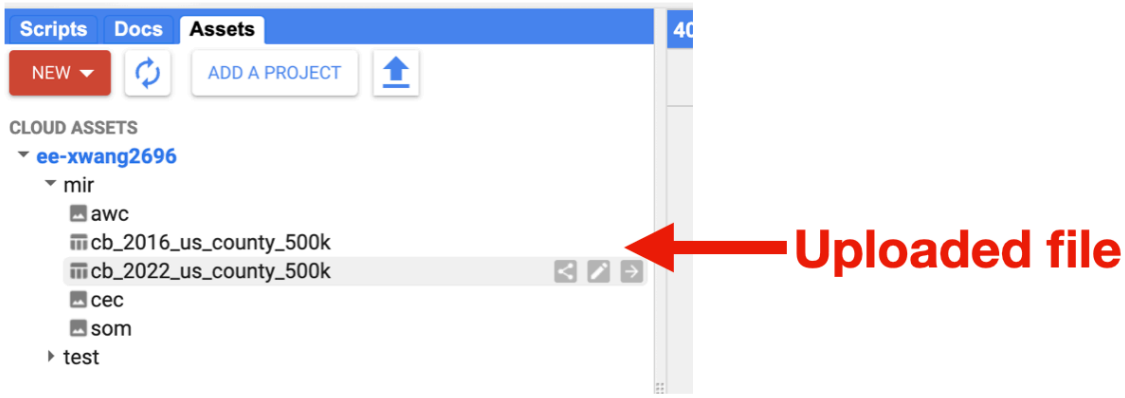

Fig. 4: This is in the top right corner of the code editor, you can find the uploaded files in the assets

1.2 Import shapefile to your code

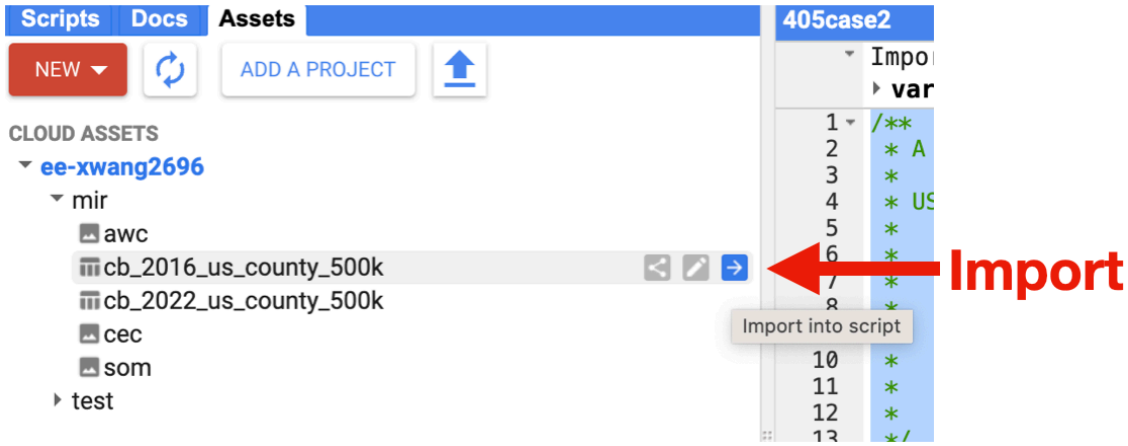

Fig. 5: This is in the top left corner of the code editor, you can click the right arrow to import a file into your script

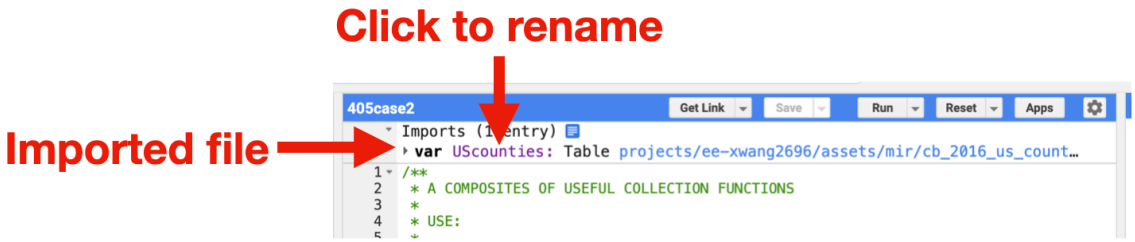

Fig. 6: This is in the top of the code editor, you can see the imported file and rename it

2 Code overview

You need to download the Case Study 2(Part 1).docx code file. Once you've uploaded and imported the shapefile, launch the code editor in your browser, copy and paste the code to your own code editor, and run the code.

3 Procedures

3.1 Visualize the U.S. counties shapefile

Code block 1:

```
// Import it from top to code
var counties = UScounties;
print(counties);
// Add it to map and check
Map.addLayer(counties, {}, 'ctn');
```

3.1.1 Visualization in map

You can show/hide the drawn map by clicking the 'layers' button on the top right corner of the map (Fig. 7).

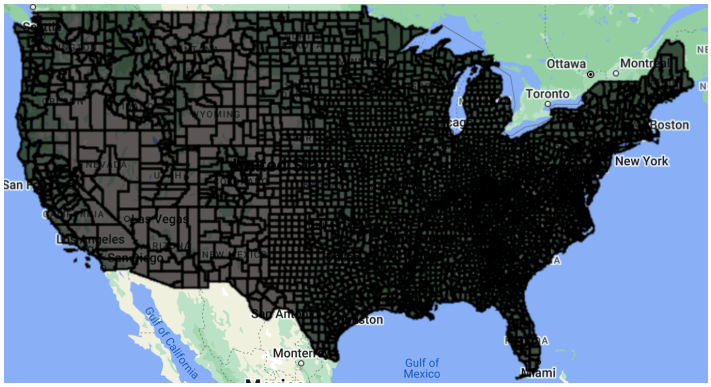

Fig. 7

3.2 Visualize the CDL crop mask: CDL doc link

Code block 2:

```
// Year
var year = 2012
var start_day = year + '-01-01'
var end_day = year + '-12-31'
// Apply CDL mask to filter the corn field
var dataset = ee.ImageCollection('USDA/NASS/CDL')
  .filter(ee.Filter.date(start_day, end_day))
  .first()
// Equal 1 is corn
var cropMask = dataset.select('cropland').eq(1);
print('cropMask', cropMask)
Map.addLayer(cropMask, {}, 'cropMask');
```

3.2.1 Visualization in map

You can show/hide the drawn map by clicking the 'layers' button on the top right corner of the map (Fig. 8).

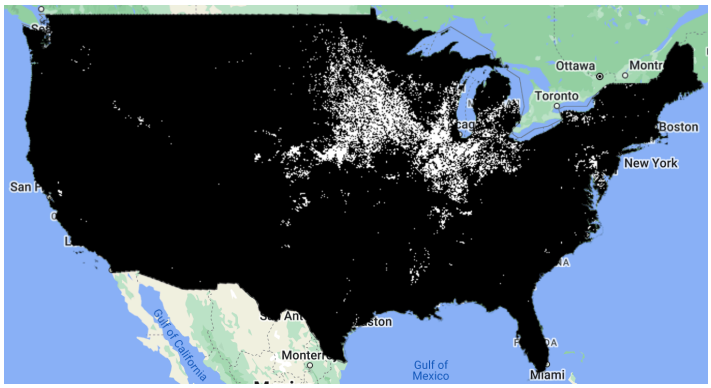

Fig. 8

3.3 Visualize the MODIS dataset: MODIS doc link

Code block 3:

```
var start = '-03-01';
var end = '-11-30';
// Get the MODIS dataset
```

```
var modVI = ee.ImageCollection('MODIS/061/MCD43A4')
  .filterDate(year+start, year+end)
print('modVI',modVI)
// Define a color to show the MODIS dataset
var trueColorVis = {
  min: 0.0,
  max: 4000.0,
  gamma: 1.4,
};
Map.addLayer(modVI, trueColorVis, 'modVI');
```

3.3.1 Visualization in map

You can show/hide the drawn map by clicking the 'layers' button on the top right corner of the map (Fig. 9).

Fig. 9

3.4 Use CDL to mask MODIS

```
var maskmodVI = modVI.map(function(img){return img.updateMask(cropMask)})
```

ee.Image.updateMask: doc link

```
Image.updateMask(mask)
'''
Updates an image`s mask at all positions where the existing mask is not zero. The output image retains the metadata and
    footprint of the input image.

Args:
  this.image: Image
  mask: Image
'''
```

ee.ImageCollection.map: doc link

```
ImageCollection.map(algorithm, dropNulls)
'''
Maps an algorithm over a collection.
Returns the mapped collection.

Args:
  this.collection: Collection
  algorithm: Function
  dropNulls: Boolean, optional
'''
```

3.4.1 Tutorial: Mapping over an ImageCollection

3.5 Clip and visualize CDL and MODIS in Wisconsin

Code block 4:

```
// FeatureCollection of counties in Wisconsin, USA.
var fipsWI = ee.FeatureCollection('TIGER/2018/States')
  .filter('STATEFP == "55"');

// Clip the WI region mask
var clipmask = cropMask.clip(fipsWI)
print('clipmask',clipmask)
Map.addLayer(clipmask, {}, 'clipmask');

// Clip the WI region VI
var clipmodVI = modVI.map(function(img) {return img.clip(fipsWI)})
print('clipmodVI',clipmodVI)

Map.addLayer(clipmodVI, trueColorVis, 'clipmodVI');

// Use the WI's mask to extract the WI's VI
var maskclipmodVI = clipmodVI.map(function(img){return img.updateMask(clipmask)})
print('maskclipmodVI',maskclipmodVI)
Map.addLayer(maskclipmodVI, trueColorVis, 'maskclipmodVI');
```

ee.FeatureCollection: doc link

```
ee.FeatureCollection(args, column)
'''
FeatureCollections can be constructed from the following arguments:
  - A string: assumed to be the name of a collection.

  - A single geometry.

  - A single feature.

  - A list of features.

  - A GeoJSON FeatureCollection

  - A computed object: reinterpreted as a collection.

Args:
  args: ComputedObject|Feature|FeatureCollection|Geometry|List
  column: String, optional
'''
```

ee.Image.clip: doc link

```
Image.clip(geometry)
'''
Clips an image to a Geometry or Feature.
The output bands correspond exactly to the input bands, except data not covered by the geometry is masked. The output image
     retains the metadata of the input image.

Use clipToCollection to clip an image to a FeatureCollection.

Returns the clipped image.

Args:
  this.image: Image
```

```
  geometry: Feature|Geometry|Object

'''
```

3.5.1 Visualization in map

You can show/hide the drawn map by clicking the 'layers' button on the top right corner of the map (Fig. 10 Fig. 11 Fig. 12).

Fig. 10: clipmask

Fig. 11: clipmodVI

Fig. 12: maskclipmodVI

4 To do task

4.1 Explore Different Cropland Types Using CDL and MODIS Datasets in GEE

Modify the Sec. 3.2 code. Expand your analysis by exploring a different crop type or a combination of crop types using the CDL to mask the MODIS dataset. Instead of focusing solely on corn fields (where cropland equals 1), select another crop type or multiple crop types and analyze their distribution and health in a state of your choice.

Hint: see CDL link for bands information, and how to choose other field types.

4.2 Analyze Crop Fields in a Different State Using MODIS and CDL Datasets in GEE

Modify the Sec. 3.5 code, choose a different state in the USA and apply a similar analysis using the MODIS and CDL datasets.

Hint: You can find the state code of states here state code link.