Remote Sensing with Google Earth Engine – Part 2

César Iván Alvarez Mendoza PhD.
cesarivanalvarezmendoza@gmail.com

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')
                    .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'];

// 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);
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);
```

Landsat 7 TOA

Bandas para la clasificación

Área de Entrenamiento

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.
  collection: polygons,
  // Keep this list of properties from the polygons.
  properties: ['class'],
  // Set the scale to get Landsat pixels in the polygons.
  scale: 1000
});
```

Reclasificación de categorías

Áreas de entrenamiento en una sola variable

12 – Clasificación Supervisada - SVM

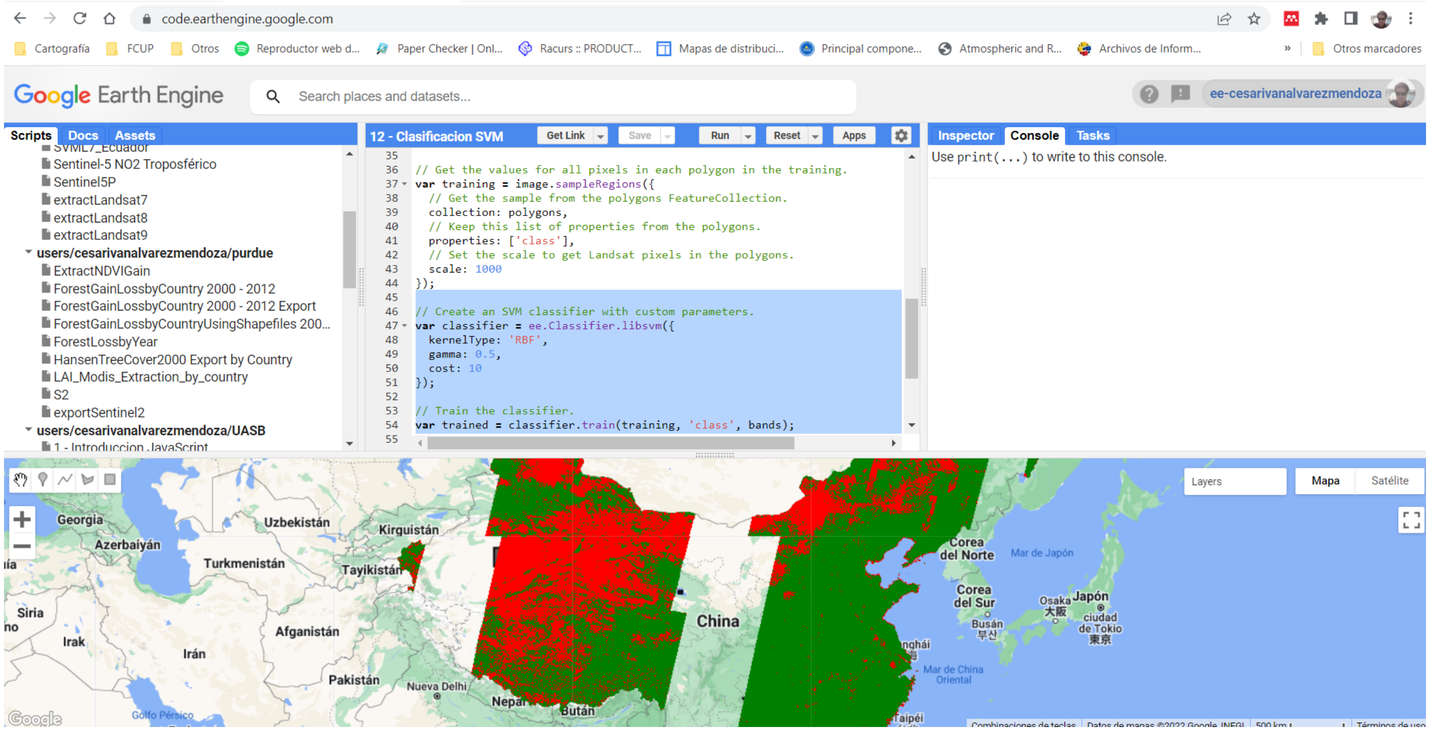
```
// Create an SVM classifier with custom parameters.
var classifier = ee.Classifier.libsvm({
  kernelType: 'RBF',
  gamma: 0.5,
  cost: 10
});

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

// Classify the image.
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');
```

Algoritmo de Machine Learning SVM

Áreas de entrenamiento

Clasificación

12 – Clasificación Supervisada - SVM



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
});
```

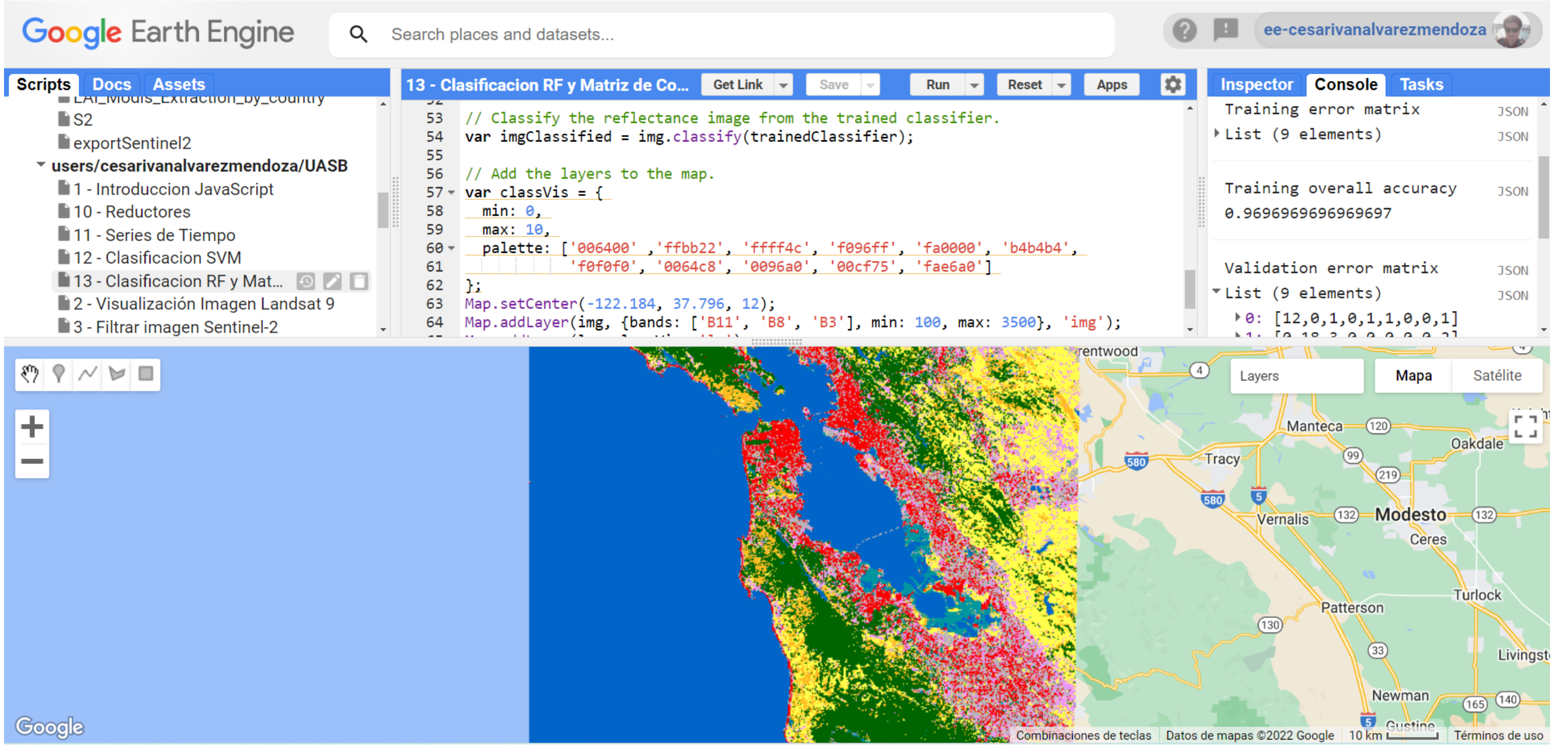
13 – Clasificación Supervisada Matriz de Confusión – Random Forest

```
// 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



Matriz de Confusión – Accuracy and Kappa Coefficients

		Truth	
		P	N
Predicted	P	TP	FP (Type 1)
	N	FN (Type 2)	TN

Accuracy = TP + TN / TP + TN + FP + FN

Matriz de Confusión – Accuracy and Kappa Coefficients

Truth

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

Out of 1 Billion People there are 10 terrorists

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

or 99.9999%

Matriz de Confusión – Accuracy and Kappa Coefficients

Truth

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

Out of 1 Billion People there are 10 terrorists

Accuracy = $10^9 - 10 / 10^9$
$= 1 - 10^{-8}$
$= 0.99999$
or 99.9999%

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.

Ground Truth (Actual) \ Classification (Predicted)	0	1	2	3	*Producer's Accuracy*
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***
Consumer's Accuracy	***0.93***	***0.91***	***1.00***	***0.93***	***0.94*** (*Overall Accuracy*)

Accuracy Assessment

Matriz de Confusión – Accuracy and Kappa Coefficients

	Class	Reference test information				Row total	User's Accuracy
		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.

César Iván Alvarez Mendoza
cesarivanalvarezmendoza@gmail.com

Gracias por su atención!