



Tutorial para mapear distúrbios florestais no sudoeste da Amazônia usando CODED, LandTrendr e MTDD

SERVIR  **AMAZONIA**

Reygadas Langarica, Yunuen
UNIVERSITY OF RICHMOND

Índice

1. Histórico.....	1
2. Definições	1
3. Algoritmos	2
3.1. CODED	2
3.1.1. <i>Mapa de distúrbios</i>	2
3.1.2. <i>Mapa de degradação e desmatamento</i>	5
3.2. LandTrendr.....	8
3.2.1. Mapa de distúrbios	8
3.3. MTDD	11
3.3.1. <i>Amostras de treinamento</i>	11
3.3.2. <i>Mapa de degradação e desmatamento</i>	16
Referências	22

Tutorial para mapear distúrbios florestais no sudoeste da Amazônia usando CODED, LandTrendr e MTDD

1. Histórico

Devido à necessidade urgente de compreender os efeitos dos distúrbios florestais sobre os serviços dos ecossistemas, vários algoritmos de sensoriamento remoto focados na detecção de mudanças na vegetação foram desenvolvidos na última década. O desafio agora está em compreender qual algoritmo melhor se adapta à área de estudo do usuário e ao objetivo da pesquisa. Este tutorial compartilha nosso trabalho do Google Earth Engine (GEE) derivado da avaliação do desempenho de três algoritmos - Detecção de Degradação Contínua (CODED), Detecção de Tendências de Distúrbios e Recuperação (LandTrendr) baseada em Landsat, e Detecção de Perturbações de Tempo Multi-variável (MTDD) - para detectar e caracterizar perturbações florestais no sudoeste da Amazônia (SWA). Para todo o estudo, consulte Reygadas et al. (2021) (atualmente em revisão).

Na seção 3.3, compartilhamos códigos GEE baseados no MTDD para mapear distúrbios no SWA (isto é, Ucayali, Peru e Acre, Brasil). Tutoriais GEE completos para CODED e LandTrendr estão disponíveis gratuitamente online; neste tutorial, compartilhamos a configuração-parametrização específica para o SWA.

2. Definições

Embora seja difícil adotar um conjunto de definições que se harmonizem com a lógica fundamental por trás de cada algoritmo, nos referimos ao desmatamento, à degradação florestal e aos distúrbios florestais da seguinte forma:

- **Desmatamento.** Conversão de longo prazo ou permanente de terras florestadas em terras não florestadas (FAO, 2001).

- **Degradação das florestas.** Embora não haja uma definição amplamente aceita, geralmente é considerado um processo de longo prazo que não leva a uma mudança na cobertura da terra, mas afeta negativamente a estrutura e função da floresta (Sasaki e Putz, 2009; Schoene et al., 2007).

- **Distúrbios florestais.** Fatores que impulsionam o estado e função da floresta que podem variar de eventos de alto impacto, como incêndios ou desmatamento, a processos sutis e graduais, como os causados por secas prolongadas, insetos ou doenças (Cohen et al., 2017; McDowell et al., 2015).

Assim, consideramos tanto o desmatamento (ou seja, evento de alto impacto) quanto a degradação florestal (ou seja, processo sutil e gradual) tipos de distúrbios florestais.

3. Algoritmos

Os três algoritmos usam valores de reflexão de superfície do Landsat Thematic Mapper (TM), Landsat Enhanced Thematic Mapper+ (ETM+) e Landsat Operational Land Imager (OLI).

3.1. CODED

O CODED v0 utiliza séries temporais de frações de membros e NDFI para detectar e caracterizar distúrbios que podem mais tarde ser classificados como degradação ou desmatamento. Veja Bullock et al. (2020) para detalhes do algoritmo e Bullock (2020) para a implementação do GEE.

3.1.1. Mapa de distúrbios

Esta seção demonstra como executar este código para produzir um mapa de distúrbios florestais.

Insumos

O usuário tem que inserir as seguintes entradas:



```
CODED_DisturbancesMap
Get Link Save Run Reset Apps

Imports (2 entries)
var studyArea: Table users/retinta/Tutorials/CaseStudy1
var trainingPts: Table users/retinta/Tutorials/TrainingPointsCODED

1 ////////////////////////////////////////////////// CODED ALGORITHM //////////////////////////////////////
2 // Retrieves a forest disturbances map //////////////////////////////////
3 // See Bullock et al., 2020 for algorithm details, and Bullock, 2020 for GEE implementation
4 // This particular piece of code to obtain CODED outputs was written by V. Reygadas and V. Galati
5
6 ////////////////////////////////////////////////// User-defined information //////////////////////////////////
7
8 // Defines the study area
9 var saveRegion = ee.FeatureCollection(studyArea);
10
11 // Defines training data
12 var trainingData = trainingPts;
13
14 // Defines parameters
15 var params = ee.Dictionary({
16   'cFThreshold': .01, // Minimum threshold to remove clouds based on cloud fraction
17   'consec': 4, // Number of consecutive observations below the change threshold needed to declare a disturbance
18   'thresh': 3, // Change threshold (observation residual normalized by the training model RMSE)
19   'start': 2000, // Start year of the study period
20   'end': 2018, // End year of the study period
21   'trainDataEnd': 2016, // End year of the training period
22   'trainDataStart': 2013, // Start year of the training period
23   'trainLength': 3, // Number of years in the training period
24   'soil': [2000, 3000, 3400, 5000, 6000, 5000], // Soil endmember reflectance value for each band
25   'gv': [100, 900, 400, 0100, 3000, 1000], // Green vegetation endmember reflectance value for each band
26   'npv': [1400, 1700, 2200, 3000, 5500, 3000], // Non-photosynthetic vegetation endmember reflectance value for each band
27   'shade': [0, 0, 0, 0, 0, 0], // Shade endmember reflectance value for each band
28   'cloud': [9000, 9600, 8000, 7800, 7200, 6800], // Cloud endmember reflectance value for each band
29   'forestLabel': 1, // Label assigned to forest in the training data
30   'window': 2, // Maximum number of years to use in the monitoring period at any given time
31   'minYears': 2, // Minimum years between disturbances
32   'numChanges': 1, // Maximum number of changes to output
33   'minObs': 5, // Minimum number of observations needed to fit a model for training
34 });
35
36 // Defines output name
37 var outputName = 'CODED_Disturbances';
38
```

1. Study Area: polygon of the study area.

2. Training data: CODED requires the user to collect land-cover and land-use samples from a selected training period within the entire study period. This is usually obtained through a stratified random sample based on a combination of land-use and land-cover maps and high spatial resolution imagery available in GEE. The field that contains the land-cover identifiers must be named "label".

3. Set of parameters

4. Output name: a desired name for the disturbances map.

Executando o Código

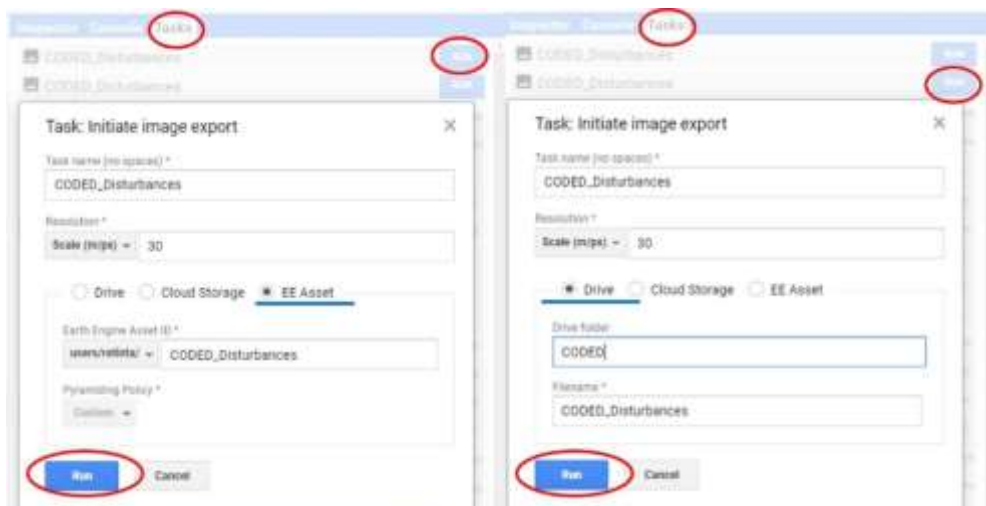
Execute o código clicando em run. Uma vez terminado, vá até o console de tarefas e clique em run para exportar os resultados como um ativo GEE e como um raster.

```
CODED_DisturbancesMap
Get Link Save Run Reset Apps

39 ////////////////////////////////////////////////////////////////////Gets CODED main results//////////////////////////////////////////////////////////////////
40
41 // Adds CODED utility functions
42 var codedUtils = require('users/retinta/Tutorials:CODED_changeDetection');
43 var dataUtils = require('users/retinta/Tutorials:CODED_dataUtils');
44
45 // Runs CODED
46 var results = codedUtils.submitCODED(saveRegion, params, trainingData);
47
48 // Turns array columns into images
49 var disturbances = dataUtils.makeImage(results, 0, 'dist_', params.get('start'), params.get('end'));
50 var magnitude = dataUtils.makeImage(results, 1, 'mag_', params.get('start'), params.get('end'));
51 var postChange = dataUtils.makeImage(results, 2, 'post_', params.get('start'), params.get('end'));
52 var difference = dataUtils.makeImage(results, 3, 'dif_', params.get('start'), params.get('end'));
53 var forestFlag = dataUtils.makeImage(results, 4, 'forest_', params.get('start'), params.get('end'));
54
55 var disturbanceBands = disturbances.addBands([magnitude, postChange, difference]);
56
57 var save_output = ee.Image(dataUtils.reduceBands(ee.Image(disturbanceBands), params)
58   .addBands(forestFlag.select(0)) // Forest flag for first year
59   .setMulti(params));
60
61 //////////////////////////////////////////////////////////////////// Exports the results ////////////////////////////////////////////////////////////////////
62
63 //Exports results as a GEE asset
64 = Export.image.toAsset({
65   image: save_output,
66   description: outputName,
67   assetId: outputName,
68   maxPixels: 1e11,
69   scale: 30,
70   region: saveRegion,
71   pyramidingPolicy: {
72     'default': "mode"
73   }
74 });
75
76 //Exports results as a raster to drive
77 = Export.image.toDrive({
78   image: save_output,
79   description: outputName,
80   region: saveRegion,
81   scale: 30
82 });
83
84 ////////////////////////////////////////////////////////////////////
85 print("All done");
```

Detects disturbances and estimates their characteristics

Exports the results as a GEE asset (used as input of the degradation/deforestation map) and as a raster (can be visualized in any GIS)



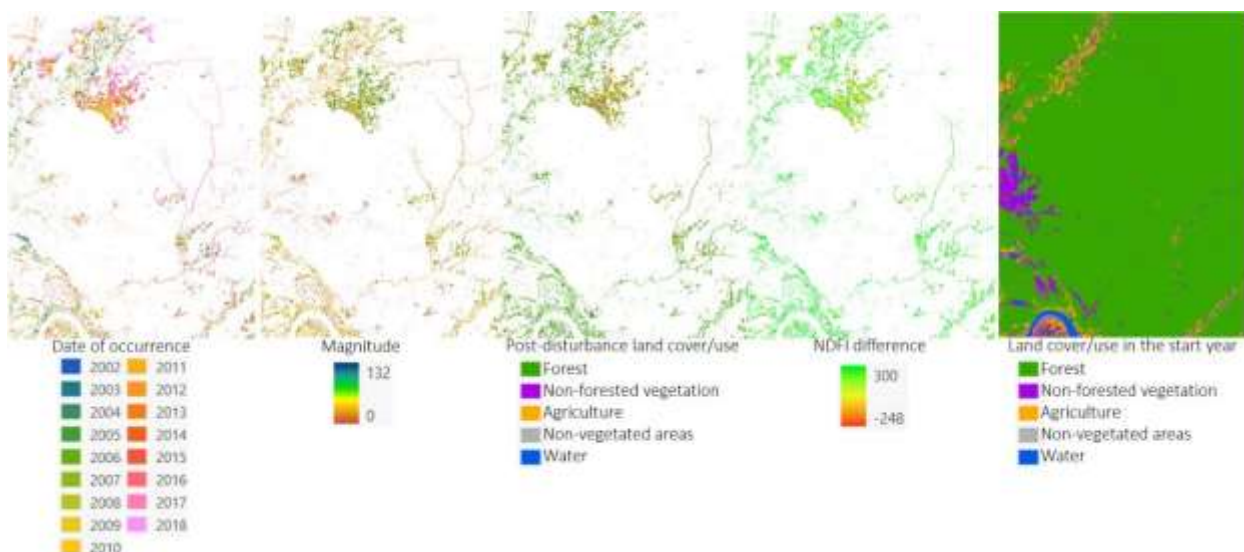
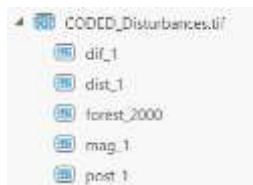
Output

Este código produz um mapa de perturbações com quatro camadas por perturbação (isto é, data de ocorrência, magnitude da mudança, cobertura da terra após a perturbação, diferença NDFI de antes e depois da perturbação) e uma última camada indicando o tipo de cobertura da terra no ano inicial.

GEE asset



Raster (".tif" file)

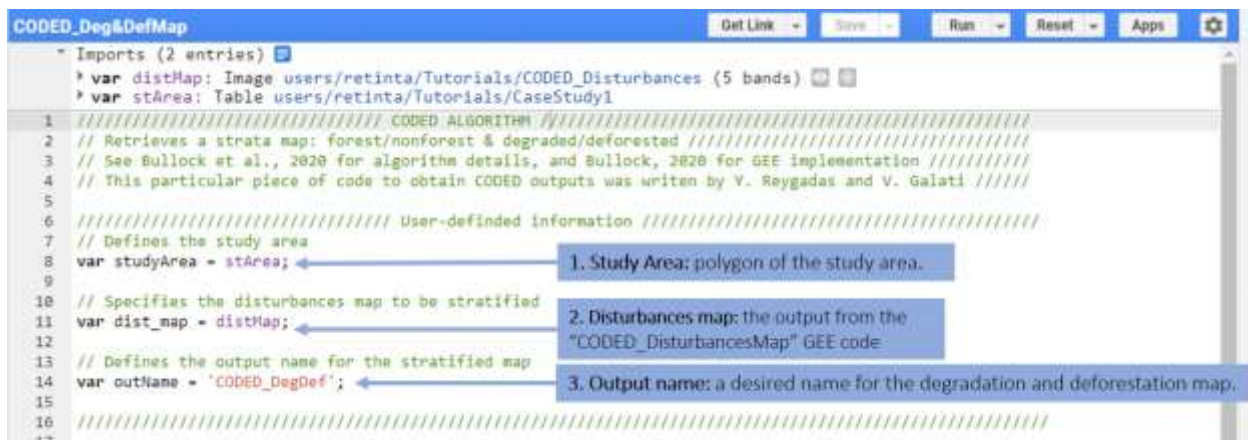


3.1.2. Mapa de degradação e desmatamento

Esta seção demonstra como executar este código para produzir um mapa classificado em floresta, não-floresta, degradação e desmatamento.

Insumos

Há três entradas que o usuário tem que entrar:



```
CODED_Deg&DefMap
Get Link Save Run Reset Apps

Imports (2 entries)
var distMap: Image users/retinta/Tutorials/CODED_Disturbances (5 bands)
var stArea: Table users/retinta/Tutorials/CaseStudy1

1 // ===== CODED ALGORITHM =====
2 // Retrieves a strata map: forest/nonforest & degraded/deforested
3 // See Bullock et al., 2020 for algorithm details, and Bullock, 2020 for GEE implementation
4 // This particular piece of code to obtain CODED outputs was written by V. Reygadas and V. Galati
5
6 // ===== User-defined information =====
7 // Defines the study area
8 var studyArea = stArea;
9
10 // Specifies the disturbances map to be stratified
11 var dist_map = distMap;
12
13 // Defines the output name for the stratified map
14 var outName = 'CODED_DegDef';
15
16
17
```

1. Study Area: polygon of the study area.

2. Disturbances map: the output from the "CODED_DisturbancesMap" GEE code

3. Output name: a desired name for the degradation and deforestation map.

Executando o Código

Execute o código clicando em run. Uma vez terminado, vá até o console de tarefas e clique em run para exportar os resultados como um raster.

The screenshot displays the Coded workspace interface. The top bar includes buttons for 'Get Link', 'Save', 'Run', 'Reset', and 'Apps'. The main area shows a code editor with a JavaScript function named `stratify` that processes land cover data. The code includes comments in green and logic for handling forest, non-forest, degradation, and deforestation based on a threshold. A blue callout box on the right states: "Produces and displays a map classified into forest, non-forest, degradation, and deforestation based on the conditions described in green".

Below the code editor, a 'Task: Initiate image export' dialog is open. It contains fields for 'Task name (for address)', 'Resolution', 'Scale (m/p)', and 'Parameter'. The 'Run' button is highlighted with a red circle. A blue callout box on the left of the dialog states: "Exports the results as a raster (can be visualized in any GIS)".

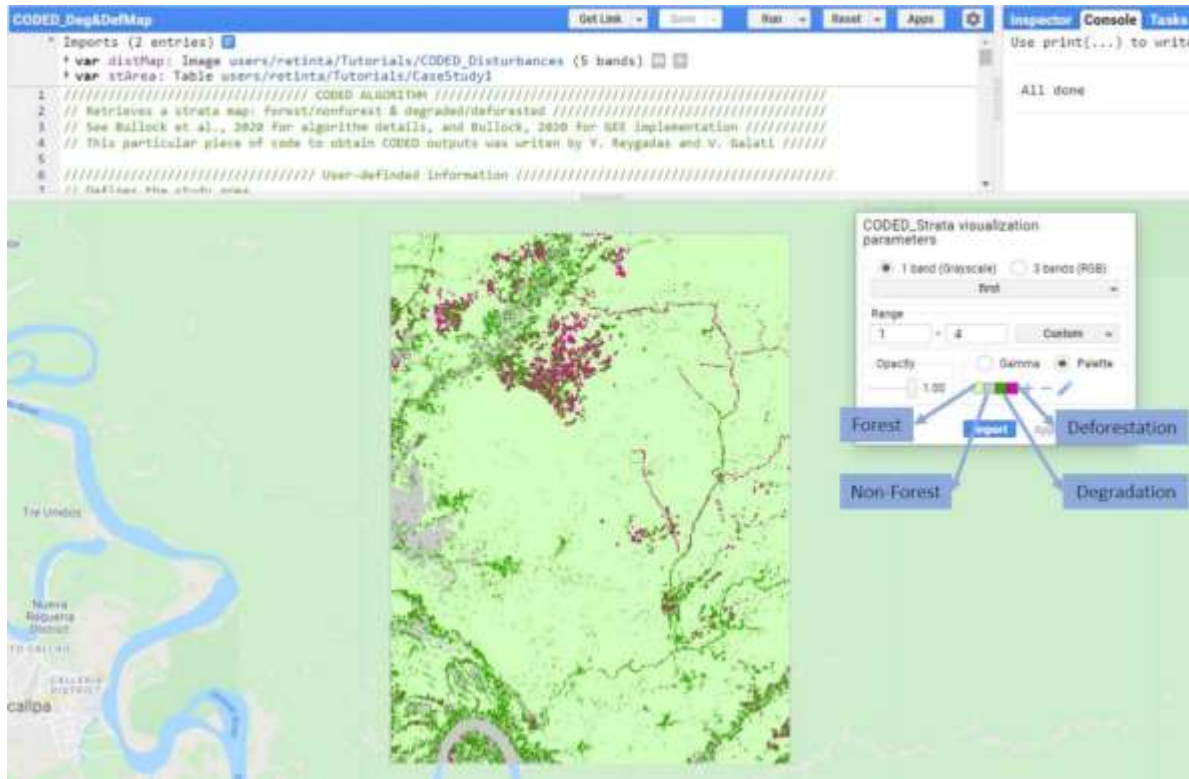
```
16 //////////////////////////////////////////////////////////////////Creates a stratified map //////////////////////////////////////////////////////////////////
17 // Function for stratifying the CODED results from the main function //
18 // (https://coded.readthedocs.io/en/latest/coded.html) //
19 var stratify = function(result) {
20   // The results are masked in areas of non change, so first unmask
21   var unmasked = result.unmask();
22   // Gets the band name of the original forest (depends on the start year)
23   var bands = result.bandNames();
24   // Gets FOREST
25   // If in the start year the land cover WAS forest (forest_startyear = 1)
26   // and there were no disturbances in the study period (dist_1=0), then forest
27   var forested = unmasked.select('forest_2000').eq(1)
28     .and(unmasked.select('dist_1').eq(0));
29   // Gets NON FOREST
30   // If in the start year the land cover WASN'T forest (forest_startyear != 1)
31   // and there were not disturbances, then nonForest and relabel
32   var nonForested = unmasked.select('forest_2000').neq(1)
33     .and(unmasked.select('dist_1').eq(0)).remap([1],[2]);
34   // For pixels that do not have enough observations post-disturbance, this sets
35   // a threshold to distinguish between DEGRADATION and DEFORESTATION
36   var magThreshold = 10;
37   // Gets DEGRADATION
38   // If after the first disturbance, the landcover is still forest (post_1=1),
39   // or if the mag_1 <= magThreshold, then degradation and relabel
40   var degradation = unmasked.select('post_1').eq(1)
41     .or(unmasked.select('mag_1').lte(magThreshold))
42     .remap([1],[3]);
43   // Gets DEFORESTATION
44   // If after the first disturbance, the landcover is not forest anymore (post_1!=1)
45   // or the mag_1 > magThreshold, then deforestation and relabel
46   var deforestation = unmasked.select('post_1').gt(1)
47     .or(unmasked.select('mag_1').gt(magThreshold))
48     .remap([1],[4]);
49   // Combines the strata into a single band image
50   var strata = ee.Image.cat([forested,nonForested,degradation,deforestation])
51     .select(0)
52     .reduce(ee.Reducer.firstNonNull());
53   // Creates and image palette and displays the combined map
54   var strataPalette = ['#013220', '#0b5d2e', '#5c6d5d', '#ff2079'];
55   //var stringRes = String(result);
56   //var stringInd = stringRes.search('CODED');
57   //var strataName = stringRes.substring(stringInd+14,stringInd+40);
58   //print (strataName);
59   Map.addLayer(strata,{min:1,max:4,palette:strataPalette},'CODED_Strata',false);
60   return strata;
61 };
62 // Runs the function for creating a strata map for an imported image and exports it
63 var toExp = stratify(dist_map);
64
```

```
76 //////////////////////////////////////////////////////////////////
77 // Exports
78 // Export Image to Drive
79 var toExp = function(image, description, outName,
80   region: studyArea,
81   scale: 30,
82   maxLineSize: 1000000) {
83   //
84   //
85   //
86   print('all done');
87 }
```


Saída

Este código produz um mapa classificado em floresta, nãofloresta, degradação e desmatamento.

Mapa exibido em GEE



Raster (".tif" file)



3.2. LandTrendr

LandTrendr detecta e caracteriza a perda ou ganho de vegetação segmentando e ajustando as trajetórias temporais de uma variável definida pelo usuário (possíveis variáveis de entrada: NDFI*added, NBR, NDVI, NDSI, NDMI, TCB, TCG, TCW, TCA, e Landsat TM - bandas equivalentes 1-5 e 7). Veja Kennedy et al. (2010) para detalhes do algoritmo e Kennedy et al. (2018) para implementação do GEE.

3.2.1. Mapa de distúrbios

Esta seção demonstra como executar este código para produzir um mapa de distúrbios florestais.

Insumos

O usuário tem que inserir as seguintes entradas:

The image shows a screenshot of the Google Earth Engine console with the LandTrendr code. Five blue callout boxes with white text and arrows point to specific parts of the code, explaining the inputs required for the script:

- 1. Input variable(s) to be segmented and from which vegetation changes will be detected.** Points to line 9: `var indices = ['NDFI'];`
- 2. Parameters to build an annual collection of Landsat surface reflectance from which input variable-time series are calculated.** Points to lines 11-16: `var startYear = 2000;`, `var endYear = 2018;`, `var startDay = '01-01';`, `var endDay = '12-31';`, `var aoi = studyArea;`, and `var maskThese = ['cloud', 'shadow', 'snow', 'water'];`
- 3. Parameters to control the segmentation of the input variable(s).** Points to lines 20-35: `maxSegments`, `spikeThreshold`, `vertexCountOvershoot`, `preventOneYearRecovery`, `recoveryThreshold`, `pvalThreshold`, `bestModelProportion`, and `minObservationsNeeded`.
- 4. Parameters to filter vegetation changes.** Points to lines 38-45: `delta`, `sort`, `year`, `mag`, `dur`, `preval`, and `mmu`.
- 5. Output name and folder; a desired name for the disturbances map and the desired output folder in Google drive.** Points to lines 48-51: `var outName = 'Disturbances';` and `var outFolder = 'LT';`

Executando o Código

Execute o código clicando em run. Uma vez terminado, vá até o console de tarefas e clique em run para exportar os resultados como um raster.

```

1 // DisturbancesMap
2
3 #####
4 // Require the landtrendr library that has NDFI added
5 var ltgee = require('users/retintat/Tutorials:LT_Utililities_NDFI');
6
7 #####
8 // Function for running LandTrendR per index
9 var doLT = function(indices) {
10   for(var i in indices) {
11     // Add index to changeParams object
12     changeParams.index = indices[i];
13     // Run LandTrendR
14     var LTresult = ltgee.runLT(startYear, endYear, startDay, endDay, aoi, indices[i], [], runParams, maskThese);
15     // Call function for getting the disturbances map
16     combinedMap(LTresult, indices[i]);
17   }
18 };
19
20 // Function for getting a disturbances map per index
21 var combinedMap = function(result, index) {
22   changeParams.year = {checked:true, start:startYear, end:endYear};
23   // Get the change map layers (yod, mag, dur, prevat, rate, dsnr)
24   var changing = ltgee.getChangeMap(result, changeParams);
25   // Display the change attribute map (specifically year of disturbance-yor-)
26   Map.addLayer(changing.select(['yor']).clip(aoi), yodVisParams, String(index));
27   //Export results
28   exportResults(changing, index, outName, outFolder);
29 }
30
31 // Function for exporting data
32 var exportResults = function(lmg, index, name, folder){
33   ExportImage.toDrive({
34     image: lmg.clip(aoi).unmask(0).short(),
35     description: 'LT_'+String(index)+'_'+name,
36     folder: folder,
37     region: aoi,
38     scale: 30,
39     maxPixels: 1000000000000
40   });
41 };
42
43 #####
44
45 // Visualization parameters (year of disturbances)
46 var palette = ['#548235', '#429982', '#8080ff', '#8080ff', '#ffff00', '#ffff00', '#ff0000'];
47 var yodVisParams = {
48   min: startYear,
49   max: endYear,
50   palette: palette
51 };
52
53 // Adjusts map
54 Map.centerOn(aoi, 10);
55
56 // Runs everything
57 doLT(indices);
58
59 print("All done");

```

Requires the Landtrendr library that also supports NDFI

Segments the input variable(s)

Produces a disturbances map with the user-defined filtering criteria

Exports the results as raster(s)

Sets the map visualization parameters

Runs all functions

Task: Initiate image export

Task name (no spaces) *

LT_NDFI_Disturbances

Resolution *

Scale (m/px) = 30

☒ Drive ☐ Cloud Storage ☐ EE Asset

Drive folder

LT

Filename *

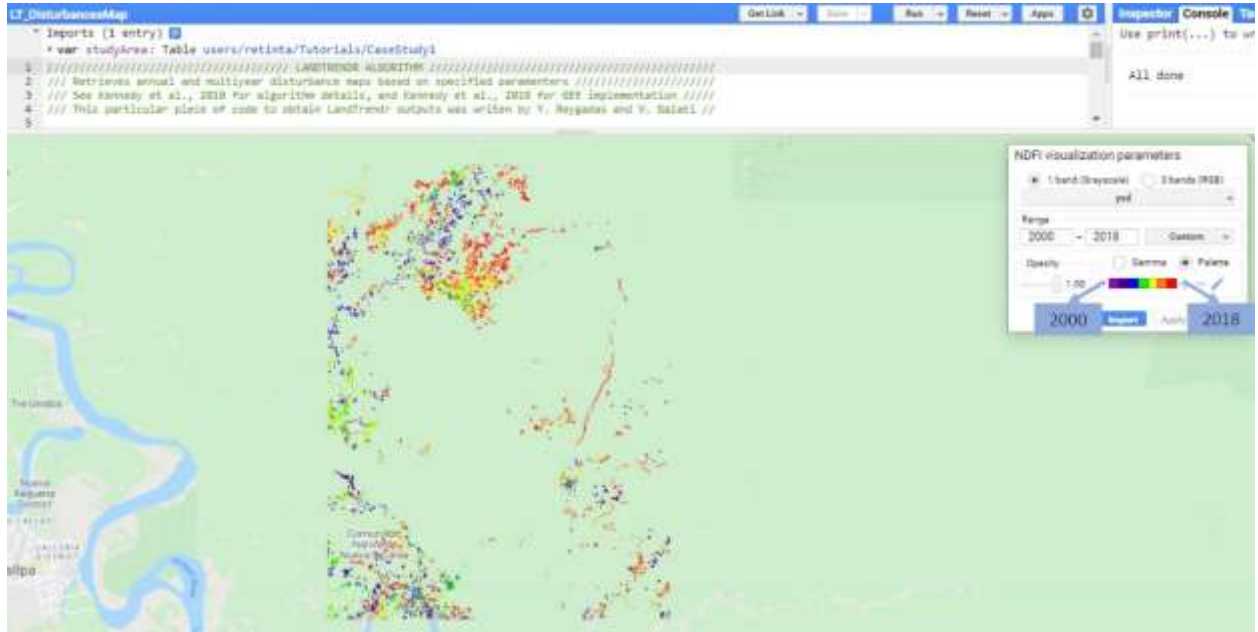
LT_NDFI_Disturbances

Run Cancel

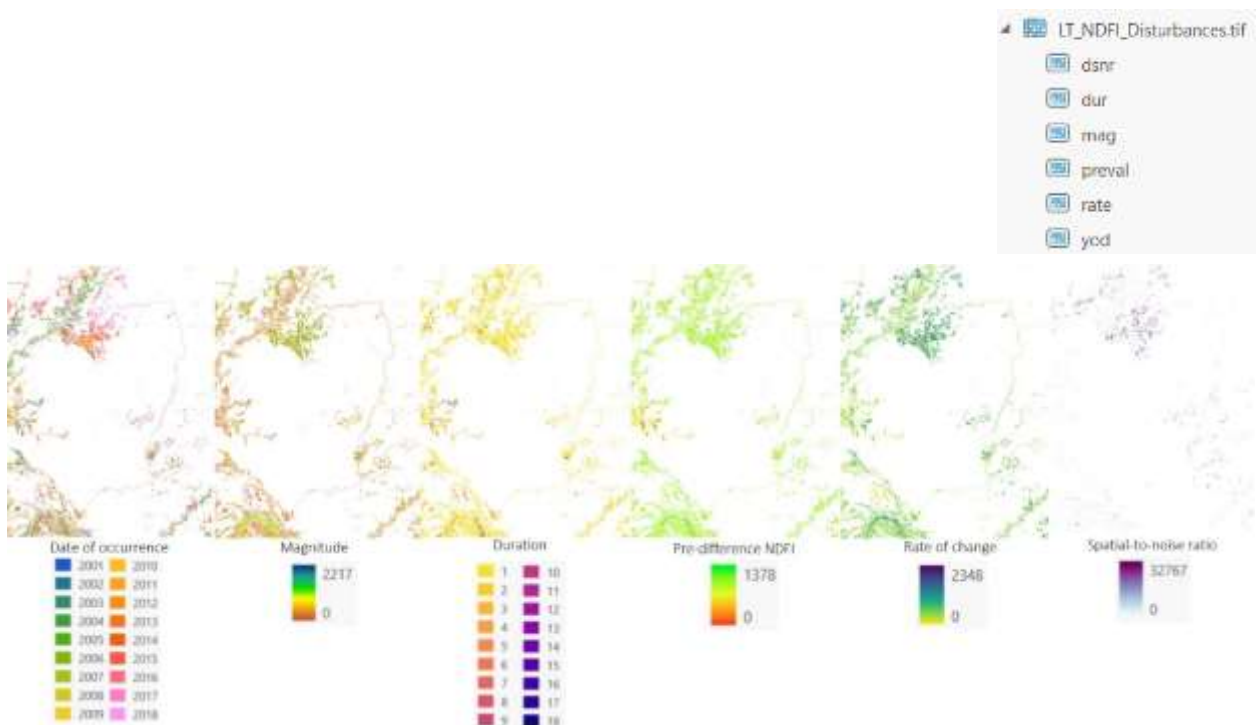
Saída

Este código produz um mapa de perturbações com seis camadas: data de ocorrência, magnitude da mudança, duração, valor espectral antes da mudança, taxa de mudança e relação sinal/ruído.

Mapa exibido em GEE (apenas a camada "ano de perturbação")



Raster (".tif" file)



3.3. MTDD

Este algoritmo baseado no MTDD classifica inicialmente as áreas florestais em intactas, degradadas e desmatadas através do treinamento de um modelo florestal aleatório com sessenta e seis métricas derivadas de seis séries temporais anuais (ou seja, NDVI, duas regiões espectrais SWIR, dois índices NDWI e SAVI) a partir das quais são calculadas onze estatísticas descritivas (ou seja, mínimo, máximo, intervalo, média, desvio padrão, coeficiente de variação, curtose, inclinação, inclinação, inclinação máxima de 5 anos e valor mais recente). Construímos este código MTDD GEE com base em Wang et al. (2019) e o adaptamos ao SWA.

3.3.1. Amostras de treinamento

Esta seção demonstra como executar este código para produzir amostras que são posteriormente utilizadas para treinar um classificador florestal aleatório.

As amostras são selecionadas aleatoriamente a partir do seguinte conjunto de condições:

Classe	Condição	Dados de referência
Intacta	Cobertura florestal em todo o período de estudo	MapBiomas (MapBiomas, 2020)
	Nenhum desmatamento em todo o período de estudo	RAISG (RAISG, 2020)
	Nenhuma perda florestal em todo o período de estudo	GFC (Hansen et al., 2013)
	Cobertura de árvores superior a 75% ^a em 2000	GFC (Hansen et al., 2013)
	Cobertura de árvores superior a 75% ^a em 2015	GFCC (Sexton et al., 2013)
Degradada	Cobertura florestal no final do ano	MapBiomas (MapBiomas, 2020)
	Desmatadas em algum momento do período de estudo	RAISG (RAISG, 2020)
	Cobertura de árvores superior a 30% ^b e inferior a 75% ^a em 2015	GFCC (Sexton et al., 2013)
Deforestada	Nenhuma cobertura florestal no final do ano	MapBiomas (MapBiomas, 2020)
	Desmatadas em algum momento do período de estudo	RAISG (RAISG, 2020)
	Perda florestal em algum momento do período de estudo	GFC (Hansen et al., 2013)

^a e ^b limiares de cobertura de árvores podem ser modificados no código GEE

Insumos

O usuário tem que inserir as seguintes entradas:

```
MTDD_Sampling
Imports (5 entries)
* var GFW: (Deprecated) Image "Hansen Global Forest Change v1.7 (2000-2019)" (13 bands)
* var USGS_TC: ImageCollection "Global Forest Cover Change (GFCC) Tree Cover Multi-Year Global 30m"
* var MAPBIOMAS: ImageCollection users/retinta/Machinelearning/MapBiomassCol
* var RAISG: Image users/retinta/Machinelearning/RAISG_2000_2018_fix1 (1 band)
* var studyArea: Table users/retinta/Tutorials/CaseStudy1

1 ////////////////////////////////////////////////// MTDD ALGORITHM //////////////////////////////////////////
2 ////////////////////////////////////////////////// Sampling //////////////////////////////////////////
3 // Outputs a user-defined number (90% of which are for training and 10% for validation) //
4 // of Intact Forest, Degraded Forest, and Deforested sampling points. If the output has //
5 // less points than expected, it is because the code ensures that the training samples //
6 // are uncorrelated with the validation samples by removing samples that are within some //
7 // distance to any other sample(s) //////////////////////////////////////////
8 // Based on Wang et al. (2019) and adapted by Raygades et al. 2021 to the Southwestern //
9 // Amazon to detect intact, disturbed, and deforested areas/////////////////////////////////
10
11 ////////////////////////////////////////////////// User-defined information //////////////////////////////////////////
12
13 // Defines the study period
14 var startYear = 2000;
15 var endYear = 2018;
16
17 // Defines the number of desired sampling points per class:
18 //intact forest, degraded forest, and deforested
19 // (90% will be used for training and 10% for validation)
20 var samPoints = 500;
21
22 // Defines the study area
23 var aoi = studyArea;
24
25 // Defines the datasets to build conditional to detect intact, degraded, and deforested areas
26 var gfw = GFW; // Global Forest Watch (GFW)
27 var usgs = USGS_TC; // Global Forest Cover Change (GFCC) Tree Cover Multi-Year Global 30m
28 var mapBiomass = MAPBIOMAS; // Annual Land Cover, MapBiomass
29 var raisg = RAISG; // Deforestation, RAISG
30
31 // Defines a and b (see table in section 3.3.1 of the tutorial) tree cover thresholds
32 var tcMaxThres = 75;
33 var tcMinThres = 30;
34
35 // Defines a desired name that will be added to identifiers for validationPts "outputName"
36 var outputName = "MTDD";
37
```

1. Study period: start and end year

2. Sampling points: number of desired sampling points per class

3. Study Area: polygon of the study area

4. Reference datasets: sources to build the conditions

5. Three cover thresholds: desired tree cover threshold for some of the conditions (see table above)

6. Output name: desired name for the training and validation points

Nota: Alguns dos conjuntos de dados usados para construir as condições datam de 2018. Portanto, o período de estudo tem que cair entre 2000 e 2018.

Executando o Código

Execute o código clicando em run. Uma vez terminado, vá até o console de tarefas e clique em run para exportar os resultados como um raster.

```
MTUD_Sampling
Get Link  Run  Reset  Apps

37 ##### All conditions #####
38
39 ##### Deforested at some point / not deforested (NAI20) #####
40 var rstartYear = NAI20.startsIn 2001, this conditional selects 2001 as startYear if the actual startYear is lower (2000)
41 if (startYear < 2001) {rstartYear = 2001;} else {rstartYear = startYear}
42 var def = raig.to(endYear).and(raig.gt(rstartYear))//Deforestation appears as 1
43 var noDef = def.not()// no deforestation appears as 1
44
45 ##### Forest loss at any time / no loss (UFW) #####
46 var endYear0 = endYear-2000// converts the end year to a number of two digits
47 var startYear0 = startYear-2000// converts the end start to a number of two digits
48 var loss = gfw.select("lossyear").lt(endYear0).and(gfw.select("lossyear").gt(startYear0)).unnest()
49 var NAI20 = loss.not()
50
51 ##### Tree cover greater than 75% in 2000 (TC75) #####
52 var tc75_2000 = gfw.select("treecover2000").gt(tcMaxThree);
53
54 ##### Tree Cover criteria (USAS) #####
55 // The USAS dataset has TC data for 2005, 2010, and 2015, so it sets a conditional
56 // to select the TC dataset closest to the endYear
57 var tcYear;
58 if (endYear < 2007) {tcYear = 2005;} // this conditional helps to select the TC dataset closest to the endYear
59 else if (endYear >= 2008 && endYear <= 2011) {tcYear = 2010;}
60 else if (endYear > 2011) {tcYear = 2015;}
61 // function for applying a filter to reduce noise in the TC data outputs (years 2005 and 2010 are specially noisy)
62 var filter = function (image) {
63   // Applies a 300 x 300 filter (default options)
64   var imageFiltered = image.focal_mean();
65   // This is only for visualization purposes. The resample is sufficient to force the computation to occur at native scale.
66   return imageFiltered.resample("1000x1000", null, 0);
67 }
68
69 // Tree cover greater than 75% in 2005 or around the end of the study period (USAS)
70 var tc75_endYear = usgs.filterBounds(aoi).select("tree_canopy_cover")
71   .filterMetadata("year", "equals", tcYear).mean().gt(tcMaxThree)// mean is only to convert it into a single image rather than image collection with only one
72 var tc75_endYear_fll = filter(tc75_endYear);
73
74 // Tree cover greater or equal than 10% and lower or equal than 75% in 2015 or around the end of the study period (USAS)
75 var tc10_endYear = usgs.filterBounds(aoi).select("tree_canopy_cover")
76   .filterMetadata("year", "equals", tcYear).mean().gt(tcMinThree)// mean is only to convert it into a single image rather than image collection with only one
77 var tc10_endYear_fll = usgs.filterBounds(aoi).select("tree_canopy_cover")
78   .filterMetadata("year", "equals", tcYear).mean().lt(tcMaxThree);
79 var tc10_75_endYear = tc10_endYear.and(tcLess75_endYear);
80 var tc10_75_endYear_fll = filter(tc10_75_endYear);
81
82 ##### Forest cover during all the study period (MapBiomas) #####
83 var rename = function (image){// function for assigning the same name to all scenes
84   return image.select(0).rename("a");
85 }
86 var mapBiomas_r = mapBiomas.map(rename);
87 var mapBiomas_fll = mapBiomas_r.filter((e, filter) => {//creates an collection containing the years within the study period (until 2019)
88   ee.filter.gt("year", startYear);
89   ee.filter.lt("year", endYear)});
90 var mapBiomas_minMax = mapBiomas_fll.reduce(ee.Reducer.minMax()).rename("min", "max");
91 var mapBiomas_range = mapBiomas_minMax.expression("b('min')-b('max')").rename("range");
92 var forestAll = mapBiomas_minMax.select("max").lt(60)//https://storage.googleapis.com/mapbiomas-public/NAI20/CLICAO2/LINENDA/codigo_de_iniciativa_1.pdf
93   .and(mapBiomas_range.gt(0)).rename("allForest");//Forest is 1 or 0 in the entire Acre-legal region during all years
94
95 ##### Forest / non-forest at the end of the study period (MapBiomas) #####
96 var forestEndYear = mapBiomas_r.filter((e, filter) => {year}).mean().lt(60)//mean is only to convert it into a single image rather than image collection with only one in
97 var noForestEndYear = forestEndYear.not();
98
99 ##### Selection of intact, disturbed, and deforested based on the above criteria #####
100
101 ##### Intact Forest (I)
102 //Criteria: never classified as deforested, TC75 in 2000,
103 //no forest loss, TC75 in 2015 (or around the end of the study period), classified as forest during all the study period
104 var intactForest = noDef.and(tc75_2000).and(noLoss)
105   .and(tc75_endYear).and(forestAll).clip(aoi);
106
107 ##### Degraded Forest (D)
108 //Criteria: Deforested at some point, TC<30<75% in 2015 (or around the end of the study period),
109 //and classified as forest at the end of the study period
110 var degradedForest = def.and(tc10_75_endYear_fll)
111   .and(forestEndYear).resample([0,1],[0,1]).clip(aoi);
112
113 ##### Deforested (F)
114 //Criteria: classified as deforested at some point, forest loss at some point,
115 // and classified as non-forest at the end of the study period
116 var deforested = def.and(loss)
117   .and(noForestEndYear).resample([0,1],[0,1]).clip(aoi);
118
119 ##### Merges classes and adds the result to the map (Using Image.cat to combine provokes an error in the sampling)
120 var IntactDegDef = intactForest.add(degradedForest).add(deforested).resample([1,1],[1,1]).rename("class");
121 Map.addLayer(IntactDegDef, [{"opacity":1,"bands":["class"],"palette":["#00FF00","#FF0000","#0000FF"],"Intact, Disturbed, and Deforested"}];
122 Map.centerOn(aoi, 0);
123
```

Builds each of the conditions described in the table above

Selects intact, degraded, and deforested areas based on the set of conditions described in green (also described in the table above)

Displays intact, degraded, and deforested areas

```

124 ////////////////////////////////////////////////// Random Sampling //////////////////////////////////////////
125
126 // Creates a random sample based on the user-defined number of sampling points per class
127 var samplingPts = IntactDefStratifiedSample({
128   numPoints: 0, // points for pixel values different than 1 and 2
129   seed: 0,
130   classValues: [1,2,3], // pixel values to be sampled
131   classPoints: {samPoints,samPoints,samPoints}, // number of sample points for each of the above classes
132   scale: 30, // landset spatial resolution
133   geometries: true, // retains the spatial information of the sample points needed for a spatial join
134 });
135
136 // Partitions the sample into training and validation points
137 samplingPts = samplingPts.randomColumn(); // adds column of uniform random numbers in a column named 'random'
138 var split = 0.9; // roughly 90% training, 10% validation
139 var training = samplingPts.filter(ee.Filter.lt('random', split));
140 var validation = samplingPts.filter(ee.Filter.gt('random', split));
141
142 // Ensures that the training samples are uncorrelated with the validation samples
143 // by removing samples that are within some distance to any other sample(s)
144 var distFilter = ee.Filter.withinDistance({//spatial join
145   distance: 500,
146   leftField: '.geo',
147   rightField: '.geo',
148   maxError: 10
149 });
150 var join = ee.Join.inverted();
151 training = join.apply(training, validation, distFilter); // applies the join
152
153 ////////////////////////////////////////////////// Prints and exports the training and validation points //////////////////////////////////////////
154
155 //Prints the results and adds them to the map
156 print("Training points", training.size());
157 //print(training);
158 Map.addLayer(training, {}, "Training Points");
159
160 print("validation points", validation.size());
161 //print(validation);
162 Map.addLayer(validation, {}, "Validation Points");
163
164 // Export the results as GEE assets (used as inputs of the map classified into intact, degraded, and deforested)
165 Export.table.toAsset({collection: training,
166   description: "TrainingPoints_" + outputName,
167   assetId: "trainingPts_" + outputName
168 });
169
170 Export.table.toAsset({collection: validation,
171   description: "ValidationPoints_" + outputName,
172   assetId: "validationPts_" + outputName
173 });
174
175 print("all done");

```

Randomly selects the user-defined number of samples per class

Divides the sampling points into training (90%) and validation (10%) datasets

Removes any points closer than 500 meters of each other to ensure training samples are uncorrelated with validation samples


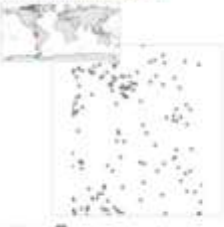
Prints the resulting number of training and validation points in the console and displays them in the map

Exports the results as GEE assets (used as inputs of the map classified into intact, degraded, and deforested)

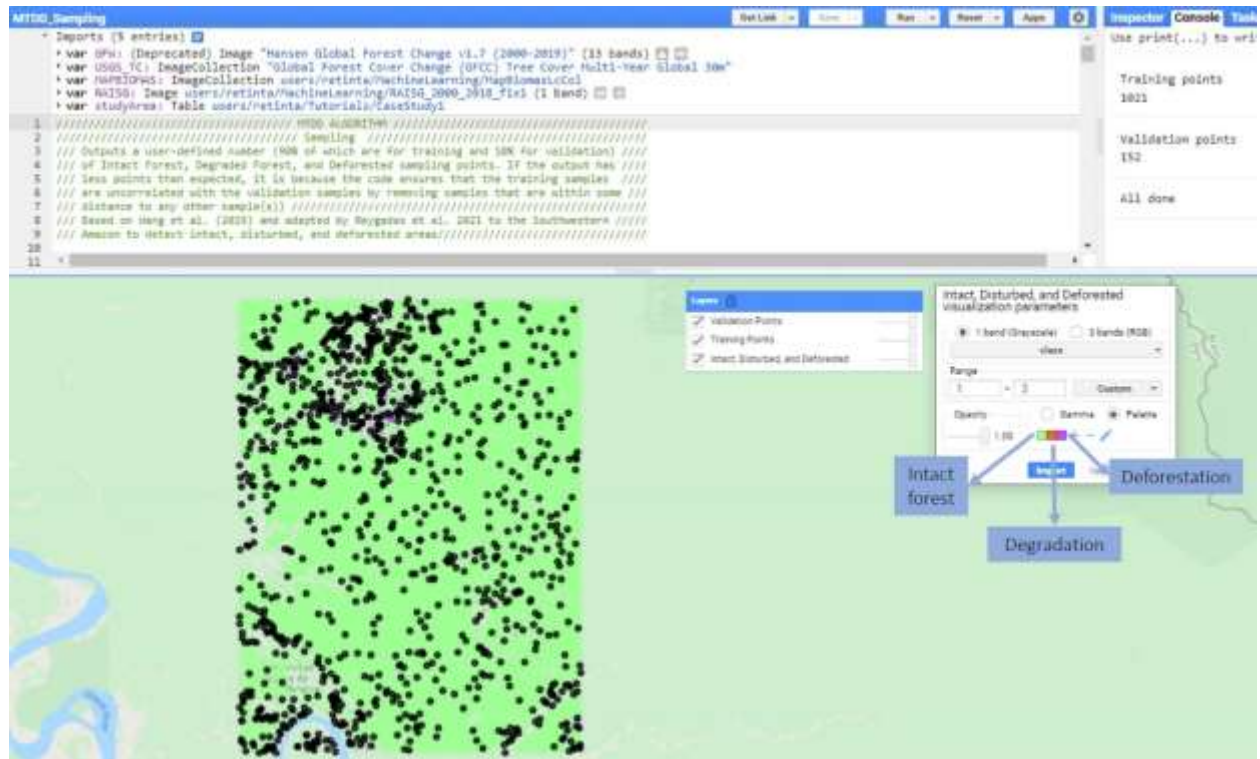
Saída

Este código produz dois conjuntos de dados: pontos de treinamento e pontos de validação.

GEE ativos

Table: trainingPts_MDTT	DESCRIPTION	FEATURES	PROPERTIES	Table: validationPts_MDTT	DESCRIPTION	FEATURES	PROPERTIES
	Feature Index	class (Long)	random (Float)		Feature Index	class (Long)	random (Float)
Table ID: users/rest/infra/tutorials/trainingPts_MDTT	0	1	0.53972374069840993	Table ID: users/rest/infra/tutorials/validationPts_MDTT	0	1	0.9287362220116006
	1	1	0.17310481412728573		1	1	0.9952952589520231
	2	1	0.3071274620055436		2	1	0.9668643293210513
	3	1	0.5262166033693878		3	1	0.9391284494480373
	4	1	0.06721850280471892		4	1	0.952914024892686

Mapa exibido em GEE



3.3.2. Mapa de degradação e desmatamento

Esta seção demonstra como executar este código para produzir um mapa classificado em floresta intacta, degradação e desmatamento.

Insumos

O usuário tem que inserir as seguintes entradas:

```
MTDD_ClassifiedMap
Imports (4 entries)
var mapBiomes: ImageCollection users/retinta/MachineLearning/MapBiomesCol
var tp: Table users/retinta/Tutorials/trainingPts_MDTT
var vp: Table users/retinta/Tutorials/validationPts_MDTT
var studyArea: Table users/retinta/Tutorials/CaseStudy1

1 ////////////////////////////////////////////////// MTDD ALGORITHM ///////////////////////////////////
2 ////////////////////////////////////////////////// Classification ///////////////////////////////////
3 // Outputs an image classified as Non Forest (0), Intact Forest (1), Disturbed Forest (2),
4 // and Deforestation (3), also VALIDATES the model and exports the results in csv format
5 //////////////////////////////////////////////////
6 // Based on Wang et al. (2018) and adapted by Rodrigues et al. 2021 to the Southwestern
7 // Amazon to detect intact, disturbed, and deforested areas
8 //////////////////////////////////////////////////
9
10 // Go to this link before starting: https://code.earthengine.google.com/accept_repo-users/emaplab/public;
11 // If you want to display the map and print the accuracies in the console,
12 // you have to uncomment lines 270-273 & 307-308, but if the study area is big, it usually produces a time computation or memory limit error
13
14 ////////////////////////////////////////////////// User-defined information ///////////////////////////////////
15
16 // Defines the years of the training data (possible years: 2000-2018)
17 var startYearT = 2000;
18 var endYearT = 2018;
19
20 // Defines years for classification (possible years: 2000-present)
21 var startYearC = 2000;
22 var endYearC = 2020;
23
24 // Defines parameters to build a Landsat collection
25 var startDayS = '01-01';
26 var endDayS = '12-31';
27 var sds = studyArea;
28 var maskTheseS = ['cloud', 'shadow', 'snow', 'water'];
29
30 // Defines training and validation samples
31 var trainingPts = tp;
32 var validationPts = vp;
33
34 // Defines datasets to define forest and non forest areas
35 var mapBiomes = mapBiomes; // Annual Land Cover
36
37 // Defines an output name that will be added to these identifiers: MTDD_Classified
38 var outfile = '2018';
```

1. Training period: start and end year used to generate the training points (start and end year in the "MTDD_Sampling" GEE code)

2. Classification period: start and end year for the classification (may or may not coincide with the training period)

3. Parameters: to build an annual Landsat surface reflectance collection from which the time series are calculated

4. Training and validation points: the output from the "MTDD_Sampling" GEE code

5. Land-use and land-cover maps: used to define a forest mask

6. Output name: a desired name for the output map

Executando o Código

Execute o código clicando em run. Uma vez terminado, vá até o console de tarefas e clique em run para exportar os resultados como um arquivo raster (mapa classificado) e csv (matriz geral de precisão e erro).

```
40 //////////////////////////////////////////////////
41 // Function for calculating ALL METRICS //////////////////////////////////////////////////
42 var metricsEval = function (startYear, endYear, startDay, endDay, aoi, maskThese) {
43   ////////////////////////////////////////////////// Time Series Trajectories //////////////////////////////////////////////////
44   // Builds an annual cloud, cloud shadow, snow, and water masked mosaic composite of Landsat
45   // Surface Reflectance TM-equivalent bands 1(Blue), 2(Green), 3(Red), 4(NIR), 5(SWIR1640), 7(SWIR2130)
46   var ltgee = require('users/remaprlab/public/modules/LandTrendr.js'); // Requires the LandTrendr library
47   var annualSRC = ltgee.buildSRCcollection(startYear, endYear, startDay, endDay, aoi, maskThese);
48   // Function for converting original Surface Reflectance (16-bit signed integer) values to 0-1 range
49   // This is not a crucial step but will keep a more homogeneous range of values between indices (NDVI, NDWI, SAVI) and bands (B1 and B7)
50   var multiply = function (image) {
51     var multiplication = image.multiply(oe.Image(0.0001));
52     return multiplication.copyProperties(image, ['system:time_start']);
53   };
54   var annualSRC_md = annualSRC.map(multiply);
55   // Function for adding a time band that will be needed to calculate temporal metrics
56   var createTimeBand = function (image) {
57     // Scales milliseconds by a large constant to avoid very small slopes
58     // in the linear regression output (temporal metrics)
59     var time = image.metadata('system:time_start').divide(1e10).rename('time');
60     return image.addBands(time);
61   };
62   var annualSRC_md_tm = annualSRC_md.map(createTimeBand);
63   // Function for adding indices as new bands to each image in the collection
64   var addIndices = function (image) {
65     var ndvi = image.expression('(b(\''B4\'')-b(\''B3\''))/(b(\''B4\'')+b(\''B3\''))').rename('NDVI');
66     var ndwi1640 = image.expression('(b(\''B4\'')-b(\''B7\''))/(b(\''B4\'')+b(\''B7\''))').rename('NDWI1640');
67     var ndwi2130 = image.expression('(b(\''B4\'')-b(\''B5\''))/(b(\''B4\'')+b(\''B5\''))').rename('NDWI2130');
68     var savj = image.expression('(1.5*((b(\''B4\'')-b(\''B3\''))/(b(\''B4\'')+b(\''B3\''))+0.5))').rename('SAVI');
69     return image.addBands([ndvi, ndwi1640, ndwi2130, savj]);
70   };
71   var annualSRC_md_tm_indices = annualSRC_md_tm.map(addIndices);
72 }
```

Start of the function that calculates all 66 metrics.

Builds a collection with annual time series of Landsat TM-equivalent bands 1-5 & 7, NDVI, NDWI1, NDWI2, and SAVI

```

74 /////////////////////////////////////////////////// DESCRIPTIVE STATISTICS //////////////////////////////////////////
75
76 // Location metrics: min, max, range, mean ////
77 var min = annualSRC_md_tm_indices.select(['B5', 'B7', 'NDVI', 'NDVI2130', 'NDVI1640', 'SAVI'])
78   .reduce(ee.Reducer.min());
79 var max = annualSRC_md_tm_indices.select(['B5', 'B7', 'NDVI', 'NDVI2130', 'NDVI1640', 'SAVI'])
80   .reduce(ee.Reducer.max());
81 var range = ee.subtract(min).rename(['B5_range', 'B7_range', 'NDVI_range', 'NDVI2130_range', 'NDVI1640_range', 'SAVI_range']);
82 var mean = annualSRC_md_tm_indices.select(['B5', 'B7', 'NDVI', 'NDVI2130', 'NDVI1640', 'SAVI'])
83   .reduce(ee.Reducer.mean());
84
85 // Scale metrics: stdDev, C.V., kurtosis, skewness ////
86 var stdDev = annualSRC_md_tm_indices.select(['B5', 'B7', 'NDVI', 'NDVI2130', 'NDVI1640', 'SAVI'])
87   .reduce(ee.Reducer.stdDev());
88 var cv = mean.divide(stdDev).rename(['B5_cv', 'B7_cv', 'NDVI_cv', 'NDVI2130_cv', 'NDVI1640_cv', 'SAVI_cv']);
89 var kurt = annualSRC_md_tm_indices.select(['B5', 'B7', 'NDVI', 'NDVI2130', 'NDVI1640', 'SAVI'])
90   .reduce(ee.Reducer.kurtosis());
91 var skew = annualSRC_md_tm_indices.select(['B5', 'B7', 'NDVI', 'NDVI2130', 'NDVI1640', 'SAVI'])
92   .reduce(ee.Reducer.skew());
93
94 // Temporal metrics ////
95 // Slope (output two bands: 'offset' (y-intercept) and 'scale' (slope))
96 var slopeB5 = annualSRC_md_tm_indices.select(['time', 'B5'])
97   .reduce(ee.Reducer.linearFit()).select(['scale'], rename('B5_slope'));
98 var slopeB7 = annualSRC_md_tm_indices.select(['time', 'B7'])
99   .reduce(ee.Reducer.linearFit()).select(['scale'], rename('B7_slope'));
100 var slopeNDVI = annualSRC_md_tm_indices.select(['time', 'NDVI'])
101   .reduce(ee.Reducer.linearFit()).select(['scale'], rename('NDVI_slope'));
102 var slopeNDVI2130 = annualSRC_md_tm_indices.select(['time', 'NDVI2130'])
103   .reduce(ee.Reducer.linearFit()).select(['scale'], rename('NDVI2130_slope'));
104 var slopeNDVI1640 = annualSRC_md_tm_indices.select(['time', 'NDVI1640'])
105   .reduce(ee.Reducer.linearFit()).select(['scale'], rename('NDVI1640_slope'));
106 var slopeSAVI = annualSRC_md_tm_indices.select(['time', 'SAVI'])
107   .reduce(ee.Reducer.linearFit()).select(['scale'], rename('SAVI_slope'));
108
109 // Max-slope (maximum absolute linear regression slope of 5-year window)
110 // Creates a moving 5-year window
111 var join = ee.Join.saveAll(); // Returns a join that pairs each element from the annual SR collection
112   matchesKey: 'images' // with a group of matching elements from the 5-year window collection.
113   // The list of matches is added to each result as an additional property
114 var diffFilter = ee.Filter.andDifference([ // Establishes the 5-year filter
115   difference: 7800000000, // 2.5 years in milliseconds, the filter selects 2 years before and after the target year
116   leftField: 'systemtime_start', // therefore, the extreme windows are made of only 3-4 years
117   rightField: 'systemtime_start'
118 ]);
119 var fiveWindowJoin = join.apply([ // Collection in which each image is associated with the 5-year window images (see proper lex
120   primary: annualSRC_md_tm_indices,
121   secondary: annualSRC_md_tm_indices,
122   condition: diffFilter
123 ]);
124 // Function for calculating the absolute maximum slope over 5-year moving windows
125 var maxAbsMovSlope = function (invar, outvar) {
126   // Calculates slopes over 5-year windows
127   var m5Slp = ee.ImageCollection(fiveWindowJoin.map(function(image) {
128     var annualSRC_md_tm_indices = ee.ImageCollection.fromImages(image.get('images'));
129     return ee.Image(image).addBands(annualSRC_md_tm_indices.select(['time', invar])
130       .reduce(ee.Reducer.linearFit()));
131   }));
132   // Gets rid of the extreme windows, which are made of only 3 or 4 years
133   var range1 = m5Slp.reduceColumns(ee.Reducer.minMax(), ['systemtime_start']); // Gets the data range of images in the collection
134   var filter1 = m5Slp.filter(ee.Filter.date(range1.get('min')).not());
135   var filter2 = filter1.filter(ee.Filter.date(range1.get('max')).not());
136   var range2 = filter2.reduceColumns(ee.Reducer.minMax(), ['systemtime_start']);
137   var filter3 = filter2.filter(ee.Filter.date(range2.get('min')).not());
138   var filter4 = filter3.filter(ee.Filter.date(range2.get('max')).not()); // Print this to see all 5-yr window slopes
139   // Converts slope values to absolute numbers
140   var abs = function (image) {return image.select('scale').abs();}
141   // Extracts the maximum absolute slope
142   return filter4.map(abs).reduce(ee.Reducer.max()).rename(outvar);
143 };
144 // Applies the absolute moving slope function to all variables
145 var m5SlpB5 = maxAbsMovSlope('B5', 'B5_m5Slp');
146 var m5SlpB7 = maxAbsMovSlope('B7', 'B7_m5Slp');
147 var m5SlpNDVI = maxAbsMovSlope('NDVI', 'NDVI_m5Slp');
148 var m5SlpNDVI2130 = maxAbsMovSlope('NDVI2130', 'NDVI2130_m5Slp');
149 var m5SlpNDVI1640 = maxAbsMovSlope('NDVI1640', 'NDVI1640_m5Slp');
150 var m5SlpSAVI = maxAbsMovSlope('SAVI', 'SAVI_m5Slp');
151
152 // Value at last year
153 var rangeDates = annualSRC_md_tm_indices.reduceColumns(ee.Reducer.minMax(), ['systemtime_start']);
154 var lastYearFil = annualSRC_md_tm_indices.filter(ee.Filter.date(rangeDates.get('max')));
155 var lastYear = lastYearFil.select(['B5', 'B7', 'NDVI', 'NDVI2130', 'NDVI1640', 'SAVI']).toBands();
156 var lastYearVal = lastYear.rename(['B5_last', 'B7_last', 'NDVI_last', 'NDVI2130_last', 'NDVI1640_last', 'SAVI_last']);
157
158 // All metrics ////
159 // Creates a collection from all metrics
160 var metricsCollection = ee.ImageCollection([min, max, range, mean, stdDev, cv, kurt, skew,
161   slopeB5, slopeB7, slopeNDVI, slopeNDVI2130, slopeNDVI1640, slopeSAVI,
162   m5SlpB5, m5SlpB7, m5SlpNDVI, m5SlpNDVI2130, m5SlpNDVI1640, m5SlpSAVI, lastYearVal]);
163
164 // Converts the metrics collection to a single multi-band image
165 var metrics = metricsCollection.toBands();
166 return metrics;
167 }
168 ///////////////////////////////////////////////////

```

Calculates 11 descriptive statistics (i.e., minimum, maximum, range, mean, standard deviation, coefficient of variation, kurtosis, skewness slope, maximum 5-year slope, and most recent value) for each of the 6 time series (i.e., NDVI, two SWIR spectral regions, two NDWI indices, and SAVI)

End of the function that calculates all 66 metrics


```

173 ////////////////////////////////////////////////// Masking: delimitation of forested areas ///////////////////////////////////
174
175 /// Selects areas that were forest at least 3 consecutive years during the study period///
176 var rename = function (image) { // Function for assigning the same name to all bands
177   return image.select(0).rename("b1");
178 }
179 var mapBiomass_r = mapBiomass.map(rename);
180 var mapBiomass_fil = mapBiomass_r.Filter(ee.Filter.and( //Creates an collection containing the years within the study period (until 2018)
181   ee.Filter.gte('year', startYearT),
182   ee.Filter.lte('year', endYearT)));
183 // Function for detecting pixels covered by forest
184 //https://storage.googleapis.com/mapbiomas-public/BAI5B/COLECA02/LPDHDA/codigo_de_la_leyenda_coleccion-2.pdf
185 //forest is 3 or 8 in the entire Acre-Ucayali region during all years
186 var forest = function (image) {
187   return image.eq(3).or(image.eq(8)).rename("forest").copyProperties(image, ['year']);
188 }
189 var forestCol = mapBiomass.map(forest);
190
191 // Function for adding a time band that will be needed to calculate land covers per year
192 var createTimeBandMask = function (image) {
193   var time = image.metadata('year').rename('time');
194   return image.addBands(time);
195 };
196 var forestCol_tm = forestCol.map(createTimeBandMask);
197
198 // Forest in three consecutive years
199 // Creates a moving 3-year window
200 var join1 = ee.Join.saveAll({matchesKey: 'images'});
201 var diffFilter1 = ee.Filter.maxDifference({difference: 1.5, leftField: 'year', rightField: 'year'});
202 var threeWindowJoin = join1.apply({primary: forestCol_tm, secondary: forestCol_tm, condition: diffFilter1});
203
204 // Function for detecting forest cover over 3-year moving windows
205 var forest3years = function (invar, outvar) {
206   // Calculates sum over 3-year windows
207   var f3y = ee.ImageCollection(threeWindowJoin.map(function (image) {
208     var forestCol_tm = ee.ImageCollection.fromImages(image.get('images'));
209     return ee.Image(image).addBands(forestCol_tm.select(['time', invar]))
210       .reduce(ee.Reducer.sum());
211   }));
212   // Extracts the maximum sum
213   return f3y.reduce(ee.Reducer.max());
214 };
215
216 // Applies the "detecting forest cover over 3-year" function to forest presence per year
217 var MaxForest3years = forest3years('forest', 'forest_3years').select('forest_sum_max');
218 // Gets the Forest mask (areas that were forest at least three consecutive years)
219 var forestMask = MaxForest3years.gte(3);
220

```

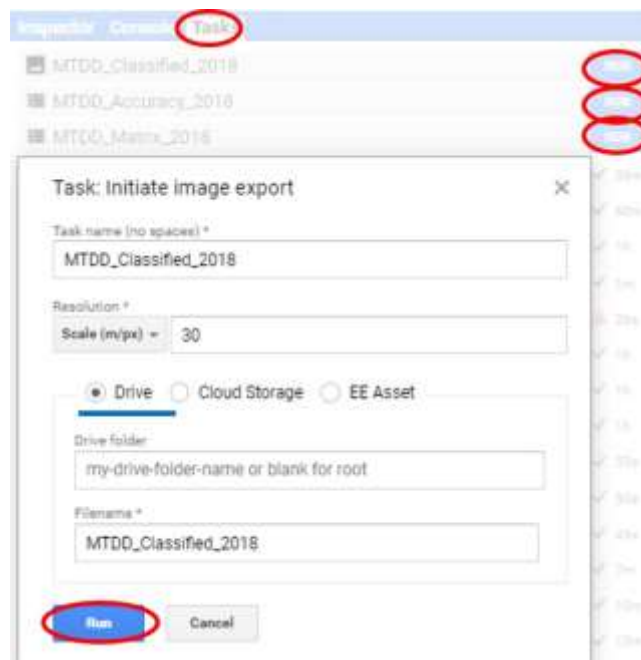
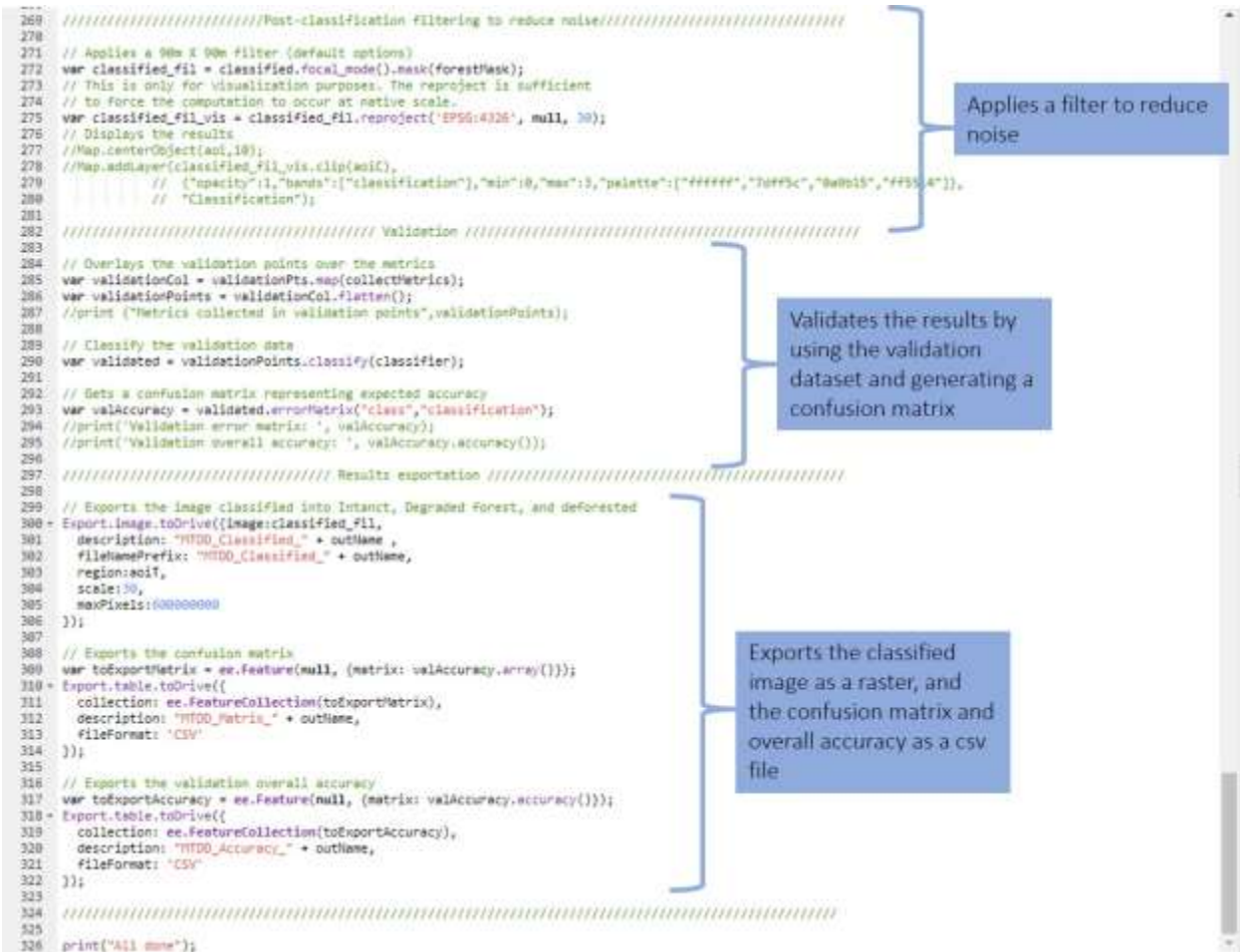
Creates a forest mask composed of all areas that were forest at least three consecutive years during the study period

```

218 ////////////////////////////////////////////////// Clasificador random forest ///////////////////////////////////
219
220 // *Crea una colección con todas las métricas con base en los años del muestreo
221 var metricsTraining = metricsGral(startYearT, endYearT, startDayB, endDayB, aoiB, maskTheseB);
222 print("Métricas muestreo", metricsTraining);
223
224 // **Crea una colección con todas las métricas con base en los años para la clasificación
225 var metricsClassification = metricsGral(startYearC, endYearC, startDayB, endDayB, aoiB, maskTheseB);
226 print("Métricas clasificación", metricsClassification);
227
228 // Sobrepone los puntos de entrenamiento con las métricas
229 var collectMetrics = function (feature) { // usar una función permite a GEE trabajar con más puntos antes de exceder su capacidad
230   return metricsTraining.sampleRegions({collection: feature,
231     properties: ['class'],
232     scale: 30,
233     tileScale: 2});
234 };
235 var trainingCol = trainingPts.map(collectMetrics);
236 var trainingPoints = trainingCol.flatten();
237
238 // Crea un clasificador random forest con 500 árboles y lo entrena
239 var classifier = ee.Classifier.smileRandomForest({numberOfTrees: 500})
240   .train({features: trainingPoints, classProperty: 'class'});
241
242 // inputProperties: ['0_B5_min', '0_B7_min', '0_HMVI_min', '0_HMVI1130_min', '0_HMVI1640_min', '0_SAVI_min',
243   '1_B5_max', '1_B7_max', '1_HMVI_max', '1_HMVI1130_max', '1_HMVI1640_max', '1_SAVI_max',
244   '2_B5_range', '2_B7_range', '2_HMVI_range', '2_HMVI1130_range', '2_HMVI1640_range', '2_SAVI_range',
245   '3_B5_mean', '3_B7_mean', '3_HMVI_mean', '3_HMVI1130_mean', '3_HMVI1640_mean', '3_SAVI_mean',
246   '4_B5_stdDev', '4_B7_stdDev', '4_HMVI_stdDev', '4_HMVI1130_stdDev', '4_HMVI1640_stdDev', '4_SAVI_stdDev',
247   '5_B5_cv', '5_B7_cv', '5_HMVI_cv', '5_HMVI1130_cv', '5_HMVI1640_cv', '5_SAVI_cv',
248   '6_B5_kurtosis', '6_B7_kurtosis', '6_HMVI_kurtosis', '6_HMVI1130_kurtosis', '6_HMVI1640_kurtosis', '6_SAVI_kurtosis',
249   '7_B5_skew', '7_B7_skew', '7_HMVI_skew', '7_HMVI1130_skew', '7_HMVI1640_skew', '7_SAVI_skew',
250   '8_B5_slp', '8_B7_slp', '8_HMVI_slp', '8_HMVI1130_slp', '8_HMVI1640_slp', '8_SAVI_slp',
251   '14_B5_mdslp', '14_B7_mdslp', '14_HMVI_mdslp', '14_HMVI1130_mdslp', '14_HMVI1640_mdslp', '14_SAVI_mdslp',
252   '20_B5_last', '20_B7_last', '20_HMVI_last', '20_HMVI1130_last', '20_HMVI1640_last', '20_SAVI_last']
253   });
254
255 // Clasifica las métricas deseadas
256 var metricsMask = metricsClassification.map(forestMask); // Mascarar las áreas no forestales
257 var classified = metricsMask.classify(classifier); // No bosque (0), bosque intacto (1), bosque degradado (2), áreas deforestadas (3)
258 print("Mapa clasificado");
259

```

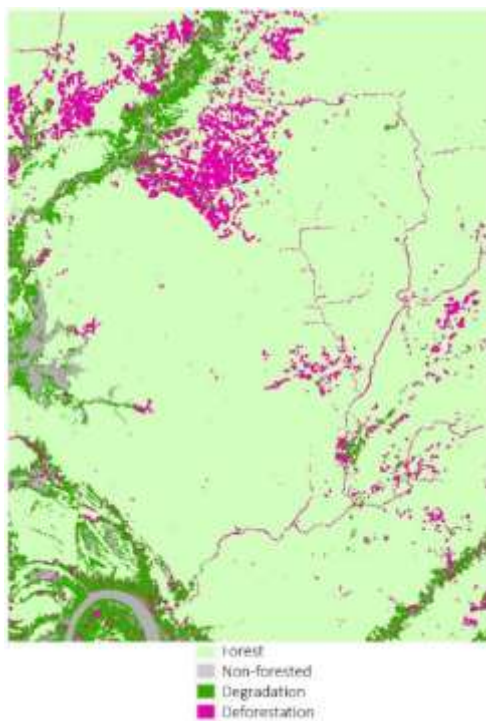
Entrena un clasificador random forest con las 66 métricas de los datos de entrenamiento y la clasifica las imágenes deseadas



Saída

Este código produz um mapa classificado em floresta não florestal, floresta intacta, degradação e desmatamento.

Raster (".tif" file)



Matriz geral de precisão e confusão (".csv" arquivos)

	A	B	C
1	system:in matrix	.geo	
2	0	0.927632	
3			

	A	B	C	D	E	F
1	system:in matrix	.geo				
2	0	[[0, 0, 0, 0], [0, 60, 0, 0], [0, 0, 47, 4], [0, 1, 6, 34]]				
3						
4						

Referências

- Bullock, E., 2020. Continuous Degradation Detection (CODED) — coded 0.2 documentation [WWW Document]. URL <https://coded.readthedocs.io/en/latest/old.html> (accessed 2.9.21).
- Bullock, E.L., Woodcock, C.E., Olofsson, P., 2020. Monitoring tropical forest degradation using spectral unmixing and Landsat time series analysis. *Remote Sens. Environ.* 238, 110968. <https://doi.org/10.1016/j.rse.2018.11.011>
- Cohen, W.B., Healey, S.P., Yang, Z., Stehman, S. V., Brewer, C.K., Brooks, E.B., Gorelick, N., Huang, C., Hughes, M.J., Kennedy, R.E., Loveland, T.R., Moisen, G.G., Schroeder, T.A., Vogelmann, J.E., Woodcock, C.E., Yang, L., Zhu, Z., 2017. How Similar Are Forest Disturbance Maps Derived from Different Landsat Time Series Algorithms? *Forests* 8, 98. <https://doi.org/10.3390/f8040098>
- FAO, 2001. Global Forest Resources Assessment 2000. Rome, Italy.
- Hansen, M.C., Potapov, P. V., Moore, R., Hancher, M., Turubanova, S.A., Tyukavina, A., Thau, D., Stehman, S. V., Goetz, S.J., Loveland, T.R., Kommareddy, A., Egorov, A., Chini, L., Justice, C.O., Townshend, J.R.G., 2013. High-resolution global maps of 21st-century forest cover change. *Science* (80-.). 342, 850–853. <https://doi.org/10.1126/science.1244693>
- Kennedy, R.E., Yang, Z., Cohen, W.B., 2010. Detecting trends in forest disturbance and recovery using yearly Landsat time series: 1. LandTrendr-Temporal segmentation algorithms. <https://doi.org/10.1016/j.rse.2010.07.008>
- Kennedy, R.E., Yang, Z., Gorelick, N., Braaten, J., Cavalcante, L., Cohen, W.B., Healey, S., 2018. Implementation of the LandTrendr algorithm on Google Earth Engine. *Remote Sens.* 10, 691. <https://doi.org/10.3390/rs10050691>
- MapBiomas, 2020. MapBiomas Amazonia Project-Collection 2.0 of annual land-cover and land-use maps [WWW Document]. URL <https://code.earthengine.google.com/toolkit-download>
- McDowell, N.G., Coops, N.C., Beck, P.S.A., Chambers, J.Q., Gangodagamage, C., Hicke, J.A., Huang, C. ying, Kennedy, R., Krofcheck, D.J., Litvak, M., Meddens, A.J.H., Muss, J., Negrón-Juarez, R., Peng, C., Schwantes, A.M., Swenson, J.J., Vernon, L.J., Williams, A.P., Xu, C., Zhao, M., Running, S.W., Allen, C.D., 2015. Global satellite monitoring of climate-induced vegetation disturbances. *Trends Plant Sci.* <https://doi.org/10.1016/j.tplants.2014.10.008>
- RAISG, 2020. Amazon Geo-Referenced Socio-Environmental Information Network [WWW Document]. URL <https://www.amazoniasocioambiental.org/> (accessed 3.30.21).
- Reygadas, Y., Spera, S., Galati, V., Salisbury, D.S., Silva, S., Novoa, S., 2021. Mapping Forest Disturbances Across the Southwestern Amazon: Evaluation and Comparison of Optical Remote Sensing Algorithms. *Remote Sens. Environ.* Under review.
- Sasaki, N., Putz, F.E., 2009. Critical need for new definitions of “forest” and “forest degradation” in global climate change agreements. *Conserv. Lett.* 2, 226–232. <https://doi.org/10.1111/j.1755-263X.2009.00067.x>

- Schoene, D., Killmann, W., von Lüpke, H., LoycheWilkie, M., 2007. Definitional Issues Related to Reducing Emissions from Deforestation in Developing Countries, Vol. 5. ed. Food and Agriculture Organization of the United Nations, Rome.
- Sexton, J.O., Song, X.-P., Feng, M., Noojipady, P., Anand, A., Huang, C., Kim, D.-H., Collins, K.M., Channan, S., Dimiceli, C., Townshend, J.R., 2013. International Journal of Digital Earth Global, 30-m resolution continuous fields of tree cover: Landsat-based rescaling of MODIS vegetation continuous fields with lidar-based estimates of error. *Int. J. Digit. Earth* 6, 427–448. <https://doi.org/10.1080/17538947.2013.786146>
- Wang, Y., Ziv, G., Adami, M., Mitchard, E., Batterman, S.A., Buermann, W., Schwantes Marimon, B., Marimon Junior, B.H., Matias Reis, S., Rodrigues, D., Galbraith, D., 2019. Mapping tropical disturbed forests using multi-decadal 30 m optical satellite imagery. *Remote Sens. Environ.* 221, 474–488. <https://doi.org/10.1016/j.rse.2018.11.028>