//follow the video instruction for creating “aoi” and training points

//cloud Mask

function maskabr(image) {

// Bits 3 and 5 are cloud shadow and cloud, respectively.

var cloudShadowBitMask = (1 << 3);

var cloudsBitMask = (1 << 5);

// Get the pixel QA band.

var qa = image.select('pixel\_qa');

// Both flags should be set to zero, indicating clear conditions.

var mask = qa.bitwiseAnd(cloudShadowBitMask).eq(0)

.and(qa.bitwiseAnd(cloudsBitMask).eq(0));

return image.updateMask(mask);

}

//load the data

var landsat8= ee.ImageCollection ("LANDSAT/LC08/C01/T1\_SR")

.filterDate('2018-05-03', '2018-10-30')

.map(maskabr).median().clip(aoi).select(['B1', 'B2', 'B3','B4','B5','B6','B7'])

print(landsat8, 'bands')

//visualzation parameters

var neshan = {

bands: ['B4', 'B3', 'B2'],

min: 0,

max: 3000,

gamma: 1.4,

};

//display the map

Map.addLayer(landsat8,neshan, 'ع' )

// calculate NDVI

var NDVI = (landsat8.select('B5').subtract(landsat8.select('B4')))

.divide(landsat8.select('B5').add(landsat8.select('B4')))

var NDVI2 = landsat8.normalizedDifference(['B5','B4']).rename('NDVI')

var NDBI= landsat8.normalizedDifference(['B6','B5']).rename('NDBI')

// visualize NDVI

var visparam = {min:-1, max:1, palette: ['blue', 'white', 'green']};

//show the NDVI

Map.addLayer(NDVI2,visparam, 'NDVI' )

Map.addLayer(NDBI,visparam, 'NDBI')

// add bands

var newdata = landsat8.addBands(NDVI2).addBands(NDBI)

print(newdata, 'newdata')

// merge the training

var training= water.merge(urban).merge(agriculture).merge(bare)

print(training, 'training')

//select bands for classifier

var Bands = newdata.bandNames()

print(Bands, 'Bands names')

// get training data samples

var training\_samp= newdata.select(Bands).sampleRegions({

collection: training,

properties: ['LC'],

scale: 30

})

// train your data

var classifier= ee.Classifier.smileRandomForest({numberOfTrees:50}).train({

features: training\_samp,

classProperty : 'LC',

inputProperties: Bands

})

//Run the classifier

var classfied = newdata.select(Bands).classify(classifier);

//print the model accuracy

print('RF error matrix: ', classifier.confusionMatrix());

print('RF accuracy: ', classifier.confusionMatrix().accuracy());

// show the classification result

var rang = [

'aec3d4', // water

'cdb33b', // croplands

'cc0013', // urban

'008000', // crop mosaic

'f7e084', // barren

];

Map.addLayer(classfied, {palette: rang, min:1, max:4}, 'classification')