**CART LULC Google Earth Engine**

1. **Image Collection and Filtering:**

*var collection = ee.ImageCollection('COPERNICUS/S2'). sort('CLOUDY\_PIXEL\_PERCENTAGE') .filterDate('2023-01-01', '2023-05-30') .filterBounds(geometry);*

* It creates an Image Collection object using the 'COPERNICUS/S2' dataset, which represents Sentinel-2 satellite images.
* The collection is sorted based on the 'CLOUDY\_PIXEL\_PERCENTAGE' property.
* Images within the specified date range ('2023-01-01' to '2023-05-30') are filtered.
* The collection is further filtered based on the specified geometry (e.g., a region of interest).

1. **Map Visualization:**

*Map.centerObject(geometry, 6); // Map.addLayer(geometry, {}, 'bounds'); print(collection);*

* It centers the map view on the specified geometry object.
* Optionally, it can add a layer representing the geometry (commented out in this code).
* It prints the collection to the console for inspection.

1. **Median Pixel Calculation:**

*var medianpixels = collection.median();*

* It calculates the median pixel values for each band in the collection, resulting in a single image representing the median values.

1. **Clipping and Rescaling:**

*var medianpixelsclipped = medianpixels.clip(geometry).divide(10000);*

* It clips the median pixel image to the specified geometry, limiting the image to the region of interest.
* The pixel values are divided by 10,000 to rescale them within the range of 0-1.

1. **Map Visualization of Sentinel-2 True Color Image:**

*Map.addLayer( medianpixelsclipped, { bands: ['B4', 'B3', 'B2'], min: 0, max: 1, gamma: 1.5 }, 'Sentinel\_2 true color' );*

* It adds a true color image layer to the map using the clipped and rescaled median pixel image.
* The specified bands ('B4', 'B3', 'B2') represent the red, green, and blue channels, respectively.
* The minimum and maximum pixel values are set to 0 and 1, respectively, with a gamma correction of 1.5.

1. **Training Data Preparation:**

*var trainingData = Water.merge(Barren).merge(Vegetation).merge(Street).merge(Buildings); print(trainingData);*

* It creates a training dataset by merging several feature collections: Water, Barren, Vegetation, Street, and Buildings.
* The resulting trainingData variable contains the combined features.
* The training data is printed to the console for inspection.

1. **Training Data Sampling:**

*var training = medianpixelsclipped .select(bands) .sampleRegions({ collection: trainingData, properties: [label], scale: 10 });*

* It selects the specified bands from the clipped and rescaled median pixel image.
* The sampleRegions() function samples pixels from the selected bands at the locations of the trainingData features.
* The 'label' property is used to store the land cover labels in the trainingData features.
* The 'scale' parameter defines the pixel resolution for sampling.

1. **Classifier Training:**

*var classifier = ee.Classifier.smileCart(15).train(training, label, bands);*

* It creates a CART (Classification and Regression Trees) classifier using the smileCart algorithm with a maximum of 15 nodes.
* The classifier is trained using the sampled training data, the 'label' property as the target variable, and the specified bands as the input features.

1. **Image Classification:**

*var classified = medianpixelsclipped .select(bands) .classify(classifier);*

* It selects the specified bands from the clipped and rescaled median pixel image.
* The classify() function applies the trained classifier to the selected bands, resulting in a classified image.

1. **Map Visualization of Classified Image:**

*Map.addLayer(classified, { min: 1, max: 5, palette: Palette }, "LULC Jeddah");*

* It adds a layer to the map representing the classified image.
* The minimum and maximum values are set to 1 and 5, respectively, corresponding to the land cover classes.
* The 'Palette' variable contains color codes for each land cover class.

1. **Confusion Matrix and Accuracy Calculation:**

*var trainAccuracy = classifier.confusionMatrix(); print('Training error matrix', trainAccuracy); print('Training overall accuracy', trainAccuracy.accuracy());*

* It calculates the confusion matrix and overall accuracy of the classifier using the training data.
* The confusion matrix represents the counts of true positive, true negative, false positive, and false negative predictions for each class.
* The overall accuracy is the proportion of correctly classified samples.

1. **Image Export (Optional):**

*// Export.image.toDrive({ // image: classified, // description: 'Classified\_CART', // folder: 'Jedah', // region: geometry, // scale: 10, // maxPixels: 1e13, // });*

* This section is currently commented out, but it shows how to export the classified image to Google Drive.
* The exported image would have the name 'Classified\_CART' and be saved in the 'Jedah' folder.
* The specified 'geometry' is used to define the export region.
* The 'scale' parameter defines the pixel resolution of the exported image.
* The 'maxPixels' parameter limits the maximum number of pixels in the exported image.