Crop Classification by using Sentinel- 2 images in Google Earth Engine (GEE): A case study in the state of Maharashtra

Abstract:

Creating crop type mapping of climate regions from satellite images it's not an easy task, due to the similarity of the spectral characteristics of satellite image. Crop type maps are an important data source for agricultural monitoring. In this study, multispectral sentinel 2 satellite images were used classes to identify crop types, Google Earth Engine, a cloud based computing platform, providing the remote sensing-based application aggregation methods derived. The S2 high-resolution, multispectral imaging mission with a global 5-day revisit frequency, Multispectral Instrument (MSI) samples 13 spectral bands. The classification process was performed by using Random Forest algorithm. We cannot classify a different crop types from satellite image without knowing spectral values, even on one provides open source spectral libraries for crop classification. In this study, I have collected different crop types from different research centres that are doing research on single crop. Used as a dummy spectral value. To fulfil the aims of this research, acquired dated during the rabi session in Maharashtra 2018/10/01 – 2019/06/30 Sentinel -2 images .

Study area: Is Maharashtra located in India; Maharashtra is a leading State in agriculture. Principal crops grown in the State are rice, jowar, bajra, wheat, tur, mung, urad, gram and other pulses. The State is a major producer of oilseeds. Groundnut, sunflower, soybean are the major oil seed crops. The important cash crops are cotton, sugarcane, turmeric and vegetables. Also many of agriculture research institutions are located in Maharashtra. To be better understood the data; I have generated six crop type maps based Maharashtra administrative divisions: Amravati. Aurangabad. Konkan. Nagpur. Nashik. Pune.

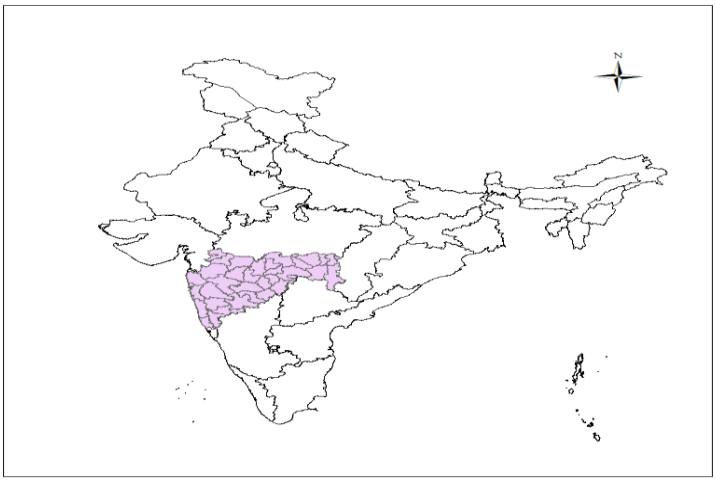

Fig: Study Area

Method:

The methodology consists of three main parts: pre-processing, classification, validation. Used GEE for collect train samples for classifier algorithms: Cotton, Rice, Sugarcane, Jowar, Wheat and other empty land for element. The classification process was performed with one of machine learning algorithm is Random Forest (FR), this algorithm is derived from statistical theories and used in classification studies.

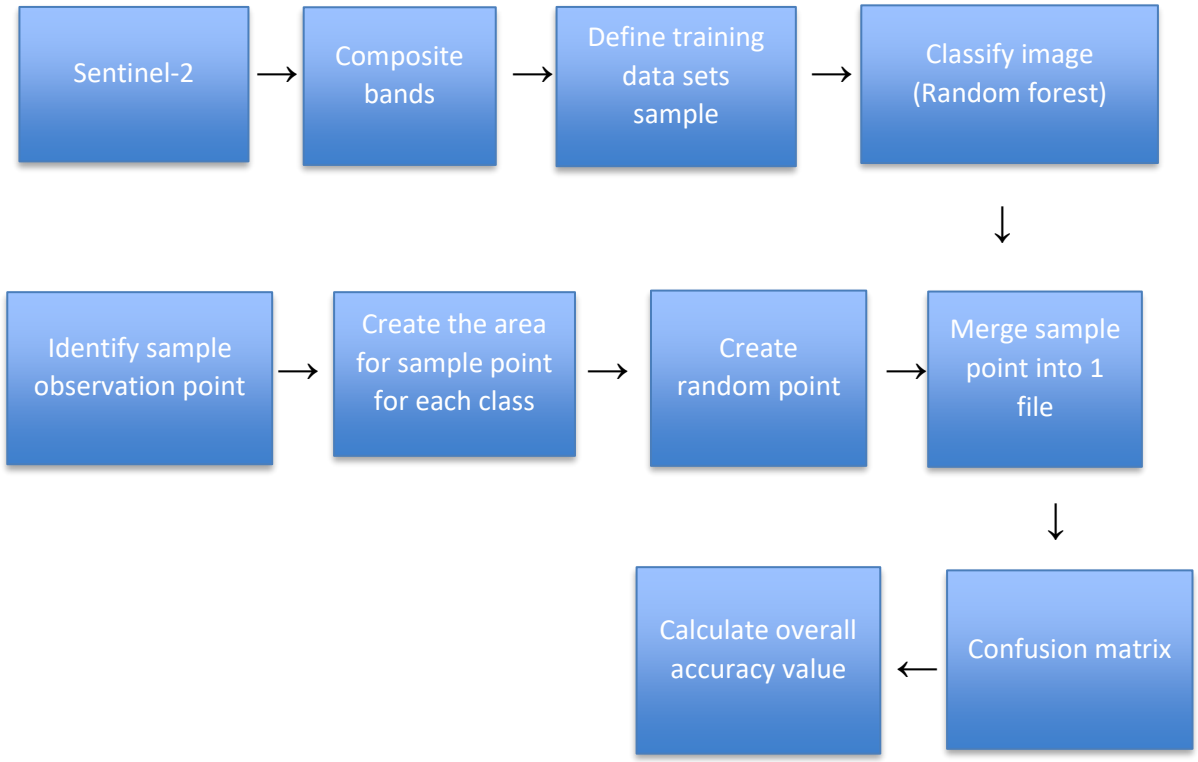

Fig: Flow Diagram

Results:

A measure of overall behaviour of the random forest algorithm can be determined by the overall accuracy, which is the total percentage of pixels correctly classified. The minimum acceptable overall accuracy is `0.9995250`. Histograms show spectral values for crop types based on R, G, B, NIR bands.

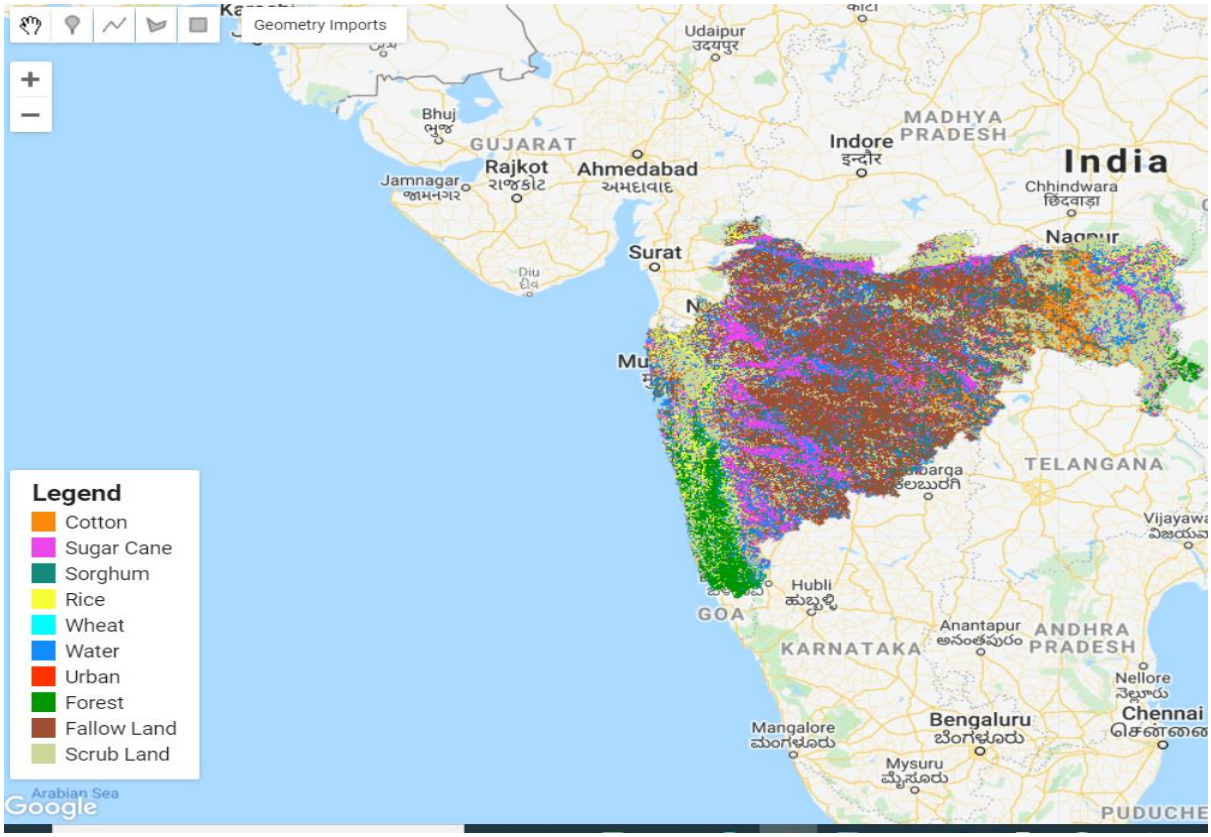

Fig: 1 Crop type map of Maharashtra

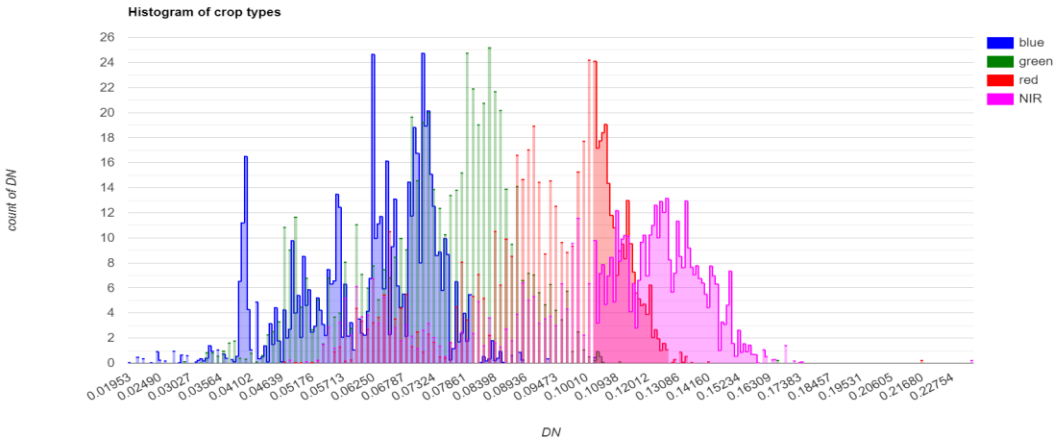

Fig: 2 Histogram of crop types

Division wise crop type maps

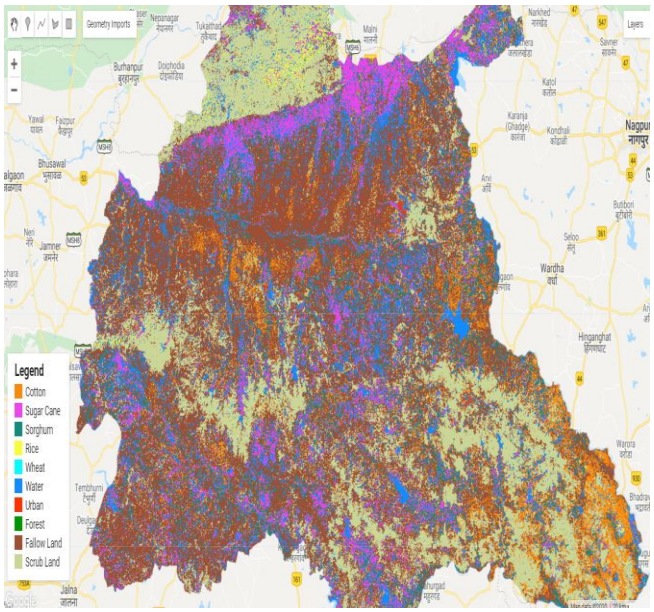

Fig: 3 Crop type map of Amaravati Division

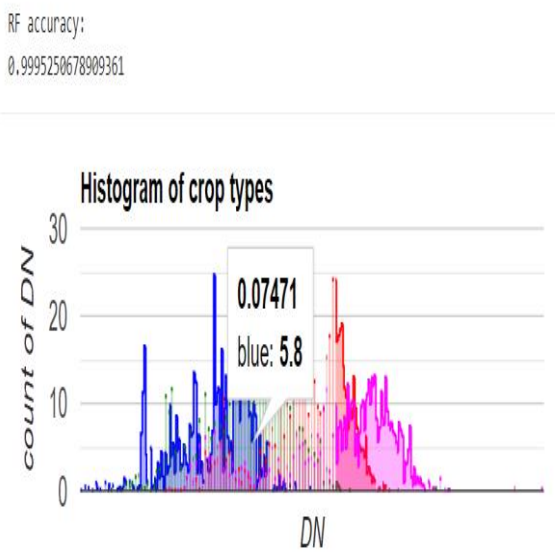

Fig: 4 Accuracy and histogram of crop types

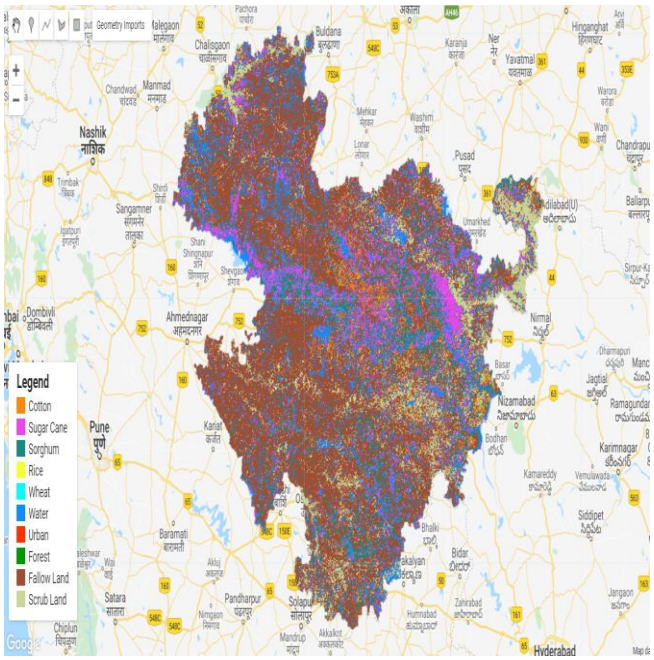

Fig: 5 Crop type map of Aurangabad Division

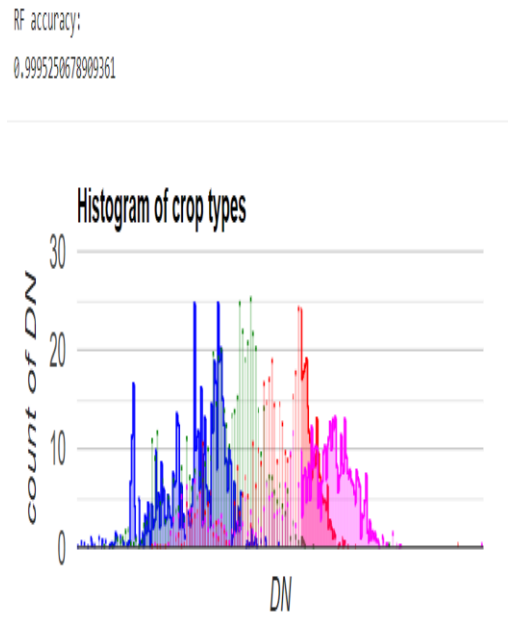

Fig: 6 Accuracy and histogram of crop types

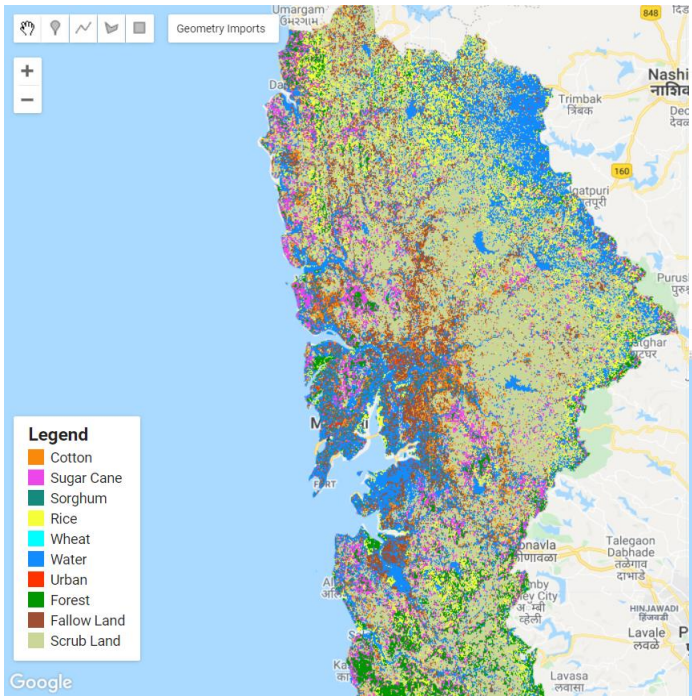

Fig:7 Crop type map of Konkan division.

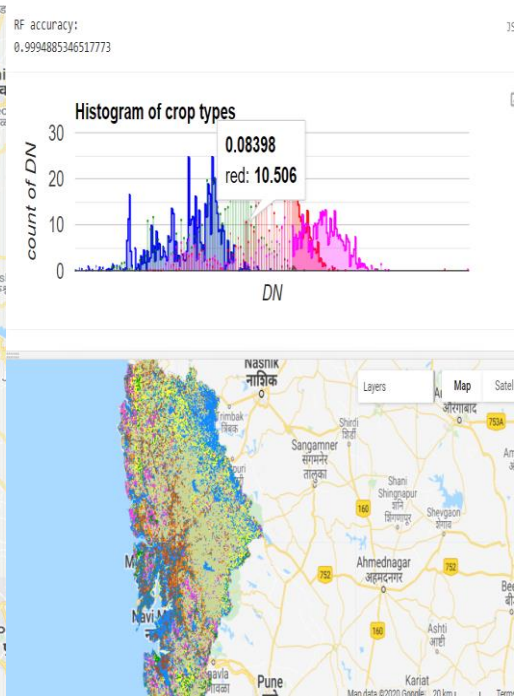

Fig: 8 Accuracy and histogram of crop types

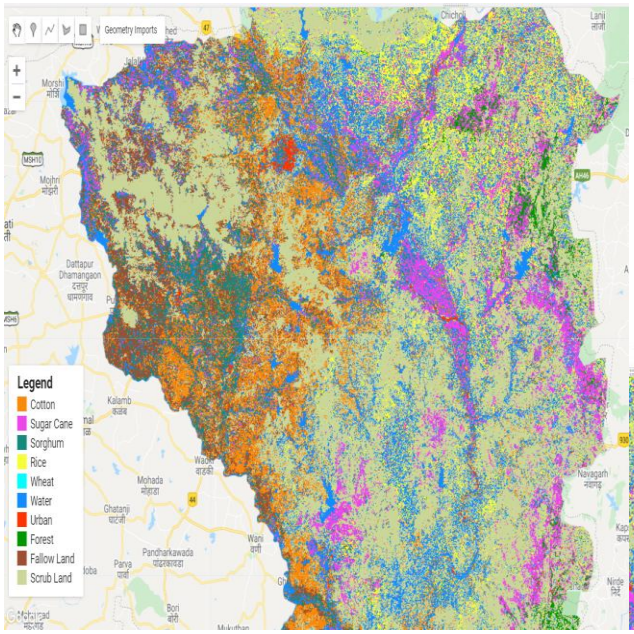

Fig: 9 Crop type map of Nagpur division.

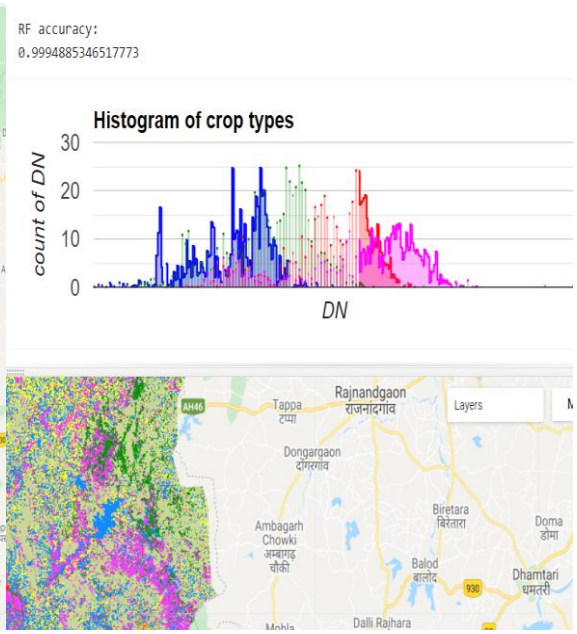

Fig: 10 Accuracy and histogram of crop types

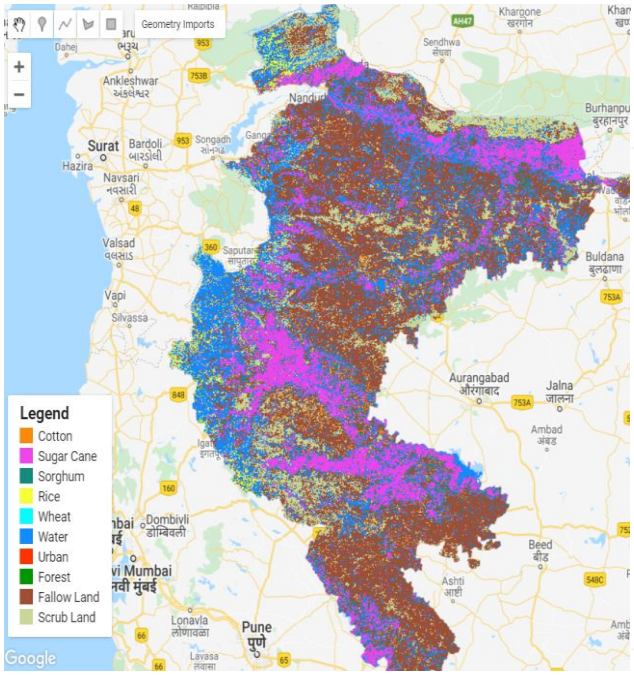

Fig: 11 Crop type map of Nashik division.

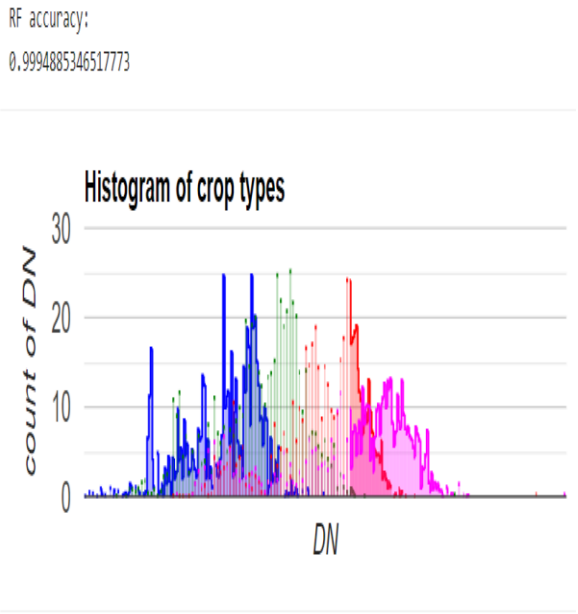

Fig: 12 Accuracy and histogram of crop types

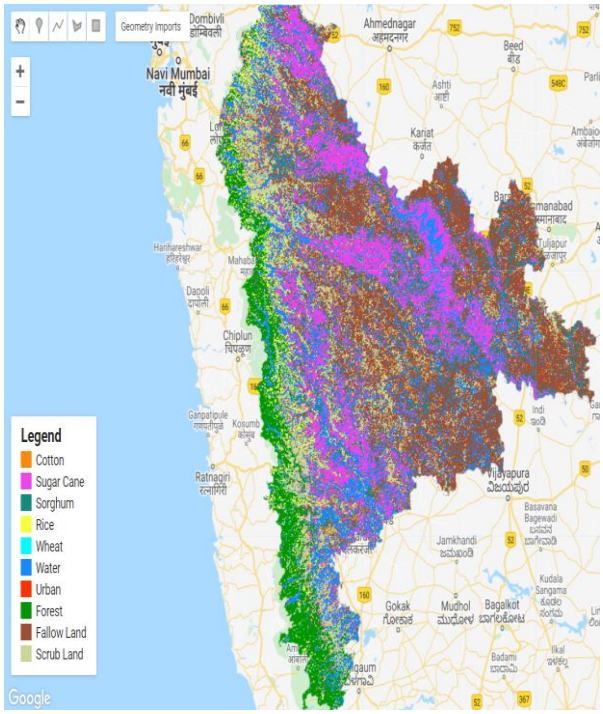

Fig: 13 Crop type map of Pune division.

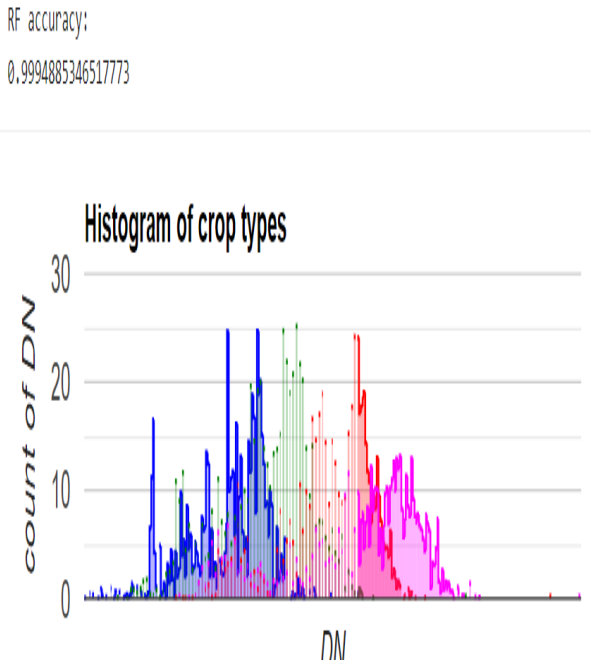

Fig: 14 Accuracy and histogram of crop types

Source Code:

```
/**
 * Function to mask clouds using the Sentinel-2 QA band
 * @param {ee.Image} image Sentinel-2 image
 * @return {ee.Image} cloud masked Sentinel-2 image
 */

function maskS2clouds(image) {
  var qa = image.select('QA60');

  // Bits 10 and 11 are clouds and cirrus, respectively.
  var cloudBitMask = 1 << 10;
  var cirrusBitMask = 1 << 11;

  // Both flags should be set to zero, indicating clear conditions.
  var mask = qa.bitwiseAnd(cloudBitMask).eq(0)
      .and(qa.bitwiseAnd(cirrusBitMask).eq(0));

  return image.updateMask(mask).divide(10000);
}

var sentinel_2 = ee.ImageCollection('COPERNICUS/S2_SR')
                  .filterBounds(AOI)
                  .filterDate('2018-10-01', '2019-06-30')

                  // Pre-filter to get less cloudy granules.
                  .filter(ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE',20))
```

```
                .map(maskS2clouds)
                .median()
print(sentinel_2)
var visualization = {
  min: 0.004,
  max: 0.39,
  bands: ['B11', 'B8', 'B4'],
};

var visual_rgb = {
  min: 0.004,
  max: 0.39,
  bands: ['B4', 'B3', 'B2'],
};
Map.setCenter(75.7139, 19.7515, 7);
var clip_aoi = sentinel_2.clip(AOI)

// Create a 3-band, 8-bit, color-IR composite to export.
var visualization = clip_aoi.visualize({
  bands: ['B4', 'B3', 'B2'],
  max: 0.4
});

// Create a task that you can launch from the Tasks tab.
Export.image.toDrive({
  image: visualization,
  description: 'Greenest_pixel_composite',
  scale: 30
});
```

```
//Amaravati Region//
var clip_amaravati = clip_aoi.clip(amaravati_region)
var clip_aurangabad = clip_aoi.clip(aurangabad_region)
var clip_kona = clip_aoi.clip(kona_region)
var clip_napur = clip_aoi.clip(napur_region)
var clip_nashik = clip_aoi.clip(nashik_region)
var clip_pune = clip_aoi.clip(pune_region)

//Map.addLayer(clip_aurangabad, visual_rgb,'Amaravati Region')

var crop_classes = cotton.merge(sugar_cane).merge(sorghum).merge(rice).merge(wheat).merge(water).merge(urban).merge(forest).merge(Fallow_land).merge(Scrub_Land)
// Select the bands for training
var bands = ['B2','B3','B4','B5','B6','B7','B8','B8A','B9','B11','B12'];


// Sample the input imagery to get a FeatureCollection of training data.
var training = clip_aoi.select(bands).sampleRegions({
  collection: crop_classes,
  properties: ['crop_type'],
  scale: 30
});


// ----------------------------------------------------------------------------------
// Random forest classification
// ----------------------------------------------------------------------------------

// Make a Random Forest classifier and train it.
var classifier = ee.Classifier.smileRandomForest(10).train({
```

```
  features: training,
  classProperty: 'crop_type',
  inputProperties: bands
});




// Define a palette for the  classification.
var palette = [
  'f98a0c', // cotton (0)
  'ed47ec', // sugar_cane (1)
  '158b7d', //sorghum(2)
  'f7ff39', //rice(3)
  '00ffff', //wheat(4)
  '128cff', //water(5)
  'ff3300', //urban(6)
  '009700', //forest(7)
  '9f4f33', // Fallow land(8)
  'cbd699' //scub land(9)
];


// Classify the Crop type of Maharashtra imagery.
//var classified_maha = clip_aoi.select(bands).classify(classifier);
//Map.addLayer(classified_maha, {min: 0, max: 9, palette: palette},'Classified Crop Types of Maharashtra')
//Classify the Region Based Crop Types
//var classified_amaravati = clip_amaravati.select(bands).classify(classifier)
//Map.addLayer(classified_amaravati, {min: 0, max: 9, palette: palette},'Classified Crop Types of Amaravati Region')
```

```
//var classified_aurangabad = clip_aurangabad.select(bands).classify(classifier)

//Map.addLayer(classified_aurangabad, {min: 0, max: 9, palette: palette},'Classified Crop Types of Aurangabad Region')


//var classified_kona = clip_kona.select(bands).classify(classifier)

//Map.addLayer(classified_kona, {min: 0, max: 9, palette: palette}, 'Classified Crop Types of Kona Region')


//var classified_napur = clip_napur.select(bands).classify(classifier)

//Map.addLayer(classified_napur, {min: 0, max: 9, palette: palette},'Classified Crop Types of Napur Region')


//var classified_nashik = clip_nashik.select(bands).classify(classifier)

//Map.addLayer(classified_nashik, {min: 0, max: 9, palette: palette},'Classified Crop Types of Nashik Region')


//var classified_pune = clip_pune.select(bands).classify(classifier)

//Map.addLayer(classified_pune, {min: 0, max: 9, palette: palette},'Classified Crop Types of Pune Region')


// Display the classification result and the input image.

Map.setCenter(75.7139, 19.7515, 7);



// // Print the confusion matrix.

// Get a confusion matrix representing resubstitution accuracy.

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

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


//Histogram

for(var a = 0; a < 9; a++){

  var His_class = classified_maha.eq(a)
```

```
  var Hist_Mask     = His_class.selfMask()


}
var Hist = Hist_Mask.addBands(sentinel_2).select('B[1-4]')
var crop_classes_hist = cotton.merge(sugar_cane).merge(sorghum).merge(rice).merge(wheat)
var options = {
  title: 'Histogram of crop types',
  fontSize: 20,
  hAxis: {title: 'DN'},
  vAxis: {title: 'count of DN'},
  series: {
    0: {color: 'blue'},
    1: {color: 'green'},
    2: {color: 'red'},
    3: {color: 'magenta'}}
};

// Make the histogram, set the options.
var histogram = ui.Chart.image.histogram(Hist, crop_classes_hist, 30)
    .setSeriesNames(['blue', 'green', 'red', 'NIR'])
    .setOptions(options);

// Display the histogram.
print(histogram);



// set position of panel
var legend = ui.Panel({
  style: {
    position: 'bottom-left',
```

```
    padding: '8px 15px'
  }
});

// Create legend title
var legendTitle = ui.Label({
  value: 'Legend',
  style: {
    fontWeight: 'bold',
    fontSize: '18px',
    margin: '0 0 4px 0',
    padding: '0'
    }
});

// Add the title to the panel
legend.add(legendTitle);

// Creates and styles 1 row of the legend.
var makeRow = function(color, name) {

      // Create the label that is actually the colored box.
      var colorBox = ui.Label({
        style: {
          backgroundColor: '#' + color,
          // Use padding to give the box height and width.
          padding: '8px',
          margin: '0 0 4px 0'
        }
      });
```

```
    // Create the label filled with the description text.
    var description = ui.Label({
      value: name,
      style: {margin: '0 0 4px 6px'}
    });

    // return the panel
    return ui.Panel({
      widgets: [colorBox, description],
      layout: ui.Panel.Layout.Flow('horizontal')
    });
};

// name of the legend
var names = ['Cotton','Sugar Cane','Sorghum','Rice','Wheat','Water','Urban','Forest','Fallow Land','Scrub Land'];

// Add color and and names
for (var i = 0; i < 10; i++) {
  legend.add(makeRow(palette[i], names[i]));
  }

// add legend to map (alternatively you can also print the legend to the console)
Map.add(legend);
```