Indian Institute of Space Science and Technology

Estimation of Ayacut Area from Sentinel-2 Imagery using the Multi Layer Perceptron Algorithm for the Kanjirappuzha Irrigation Project

A. Yeshwanth Kumar
SC18M031

M. Tech. Geoinformatics

A report submitted for partial fulfilment of the requirements
of the Outreach Programme in the
Irrigation Design and Research Board, Government of Kerala

Declaration

All sentences or passages quoted in this document from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). Any illustrations that are not the work of the author of this report have been used with the explicit permission of the originator and are specifically acknowledged.

Name: A. Yeshwanth Kumar

Date: 26.07.2019

Abstract

The objective of this project was to estimate the area of ayacut (cultivated and cultivable regions) of the Kanjirappuzha Irrigation Project region from a Sentinel-2 multispectral satellite image using a simple machine learning algorithm. The algorithm of choice was the Multi Layer Perceptron. The MLP algorithm was used to generate the land use / land cover classification of the multispectral satellite image data, which was then reclassified based to visual interpretation to obtain the area estimate of the ayacut in the region of study.

Contents

List of Figures

Chapter 1

Introduction

Land Cover data illustrates how much of a region is covered by different natural earth features such as forests, wetlands, impervious surfaces, agriculture and land and water types. Land Use data on the other hand shows how people of a region use the landscape around them such as for the purposes of development, conservation, mixed uses, etc. The different types of land cover can be managed or used quite differently. In this project a satellite image obtained from the Sentinel-2 multi spectral sensor of the Kanjirappuzha Irrigation Project region was used for the land use / land cover classification. This classified raster was then reclassified to extract the ayacut from the various LULC classes.

1.1 Aims and Objectives

The main objectives of this project were as follows:

1. Use a machine learning algorithm to identify the ayacut regions in the KPIP region.

2. Calculate the area of the estimated ayacut regions to give an idea of the gross command area.

1.2 Software Used

The following software were used for the various processes required in this project:

1. Python 3.7 (open source)

2. ArcMap 10.4.1 (proprietary)

3. Sentinels Application Toolbox - SNAP toolbox (open source)

1.3 Data Used

The following data were used at various stages in this project:

1. Multi spectral images of 10m spatial resolution covering the Kanjirappuzha Irrigation Project obtained from the Sentinel - 2 optical sensor.

2. Shapefile containing the Left and Right Bank Canals obtained by courtesy of the Irrigation Design and Research Board, Government of Kerala.

3. Shuttle Radar Topography Mission (SRTM) 1 arc-second global DEM tiles covering the KPIP region obtained from the USGS Earth Explorer portal.

4. Training data was extracted from the satellite image through visual interpretation of land features.

1.4 Multi Layer Perceptron

A perceptron is a simple algorithm that can perform binary classification. It is a linear classifier i.e. it separates the two categories in the feature space by a straight line. A perceptron first computes the weighted summation of the input features with parameters called Weights and Bias and then passes this weighted summation to a signum activation function.

A Multi Layer Perceptron can be considered an artificial deep neural network which consists of more than one layer and each layer having more than one perceptron. It usually consists of an input layer to receive the inputs, an output layer to make a decision or prediction about the input and a number of layers between these two which perform several combinations of computation on the input called the 'hidden layers'.

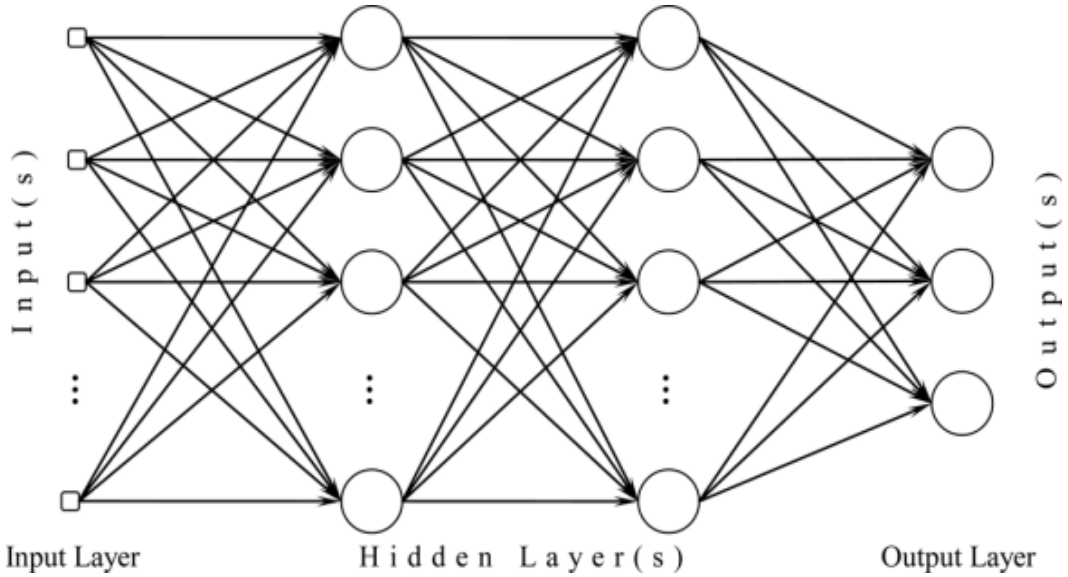


Figure 1.1: *Illustration of a Multi Layer Perceptron.*

Chapter 2

Methodology

2.1 Data Acquisition

The data used for the purpose of this project was obtained from the Sentinel - 2 Multi Spectral Instrument. The Sentinel - 2 MSI is an Earth Observation satellite sensor that acquires optical imagery of the Earth at a high spatial resolution of 10m. It can acquire multi spectral data in 13 bands in the visible, near-infrared and short wave infrared parts of the electromagnetic spectrum.

Two footprints of imagery from adjacent swaths which together envelope the entire Kanjirappuzha Irrigation Project Region were obtained from the open data portal of the European Space Agency's website.

Two tiles of the SRTM 1 arc-second global DEM from adjacent swaths which together envelope the entire Kanjirappuzha Irrigation Project Region were obtained from the USGS Earth Explorer portal.

2.2 Data Pre-processing

The data that was acquired was pre-processed in the following steps:

2.2.1 Mosaicking of DEMs:

The two DEM footprints obtained from USGS Earth Explorer were combined into a single raster by mosaicking them together. This was carried out using the Mosaic tool in ArcMap 10.4.1.

2.2.2 Generation of Watershed Boundary using Mosaicked SRTM DEMs:

In order to identify the area of study within the KPIP region, the watershed boundary was assumed to be a suitable choice. This watershed boundary was generated from the SRTM 1 arc second (30m) global DEM data in ArcMap 10.4.1 using the following procedure:

1. The mosaicked DEM raster was loaded into the ArcMap work space.

2. Small imperfections in DEM data were removed using the **Spatial Analyst Tools** → **Hydrology** → **Fill** tool in the ArcMap toolbox.

3. Next, a raster of flow direction from each cell to its steepest down-slope neighbour was calculated using the **Spatial Analyst Tools** → **Hydrology** → **Flow Direction** tool in the ArcMap toolbox.

4. Using the flow direction raster, a raster representative of the accumulated flow into each cell was calculated using the **Spatial Analyst Tools** → **Hydrology** → **Flow Accumulation** tool in the ArcMap toolbox.

5. Next, the Flow Direction raster was again used to generate a raster delineating all drainage basins in the extent of the DEM, using the **Spatial Analyst Tools** → **Hydrology** → **Basin** tool in the ArcMap toolbox.

6. Now, this raster containing the basin polygons was converted to polygon features (vector layer) using the **Conversion Tools** → **From Raster** → **Raster to Polygon** tool in the ArcMap toolbox.

7. Now, the relevant polygons which encompass the KPIP region from this basin polygon features layer were selected, combined and clipped out as a separate layer.

8. This clipped out layer was used as the watershed boundary layer which was required for further processing.

2.2.3 Atmospheric Correction of Sentinel 2 imagery:

The Sentinel 2 imagery obtained from the Open Access Hub of the European Space Agency was at a processing level of L1C (top of atmosphere reflectance). This data needed to be further processed to obtain a processing level of L2A (bottom of atmosphere reflectance). This was accomplished using the Sen2Cor v2.8 plugin available in the open source Sentinel Application Platform (SNAP) toolbox.

2.2.4 Clipping:

Now, the shapefile containing the Kanjirappuzha watershed boundary was used to clip out the only that portion of the image which contains the region of study. The remaining regions are not required and are hence removed from the image. This was carried out using the Clip tool in the software ArcMap 10.4.1.

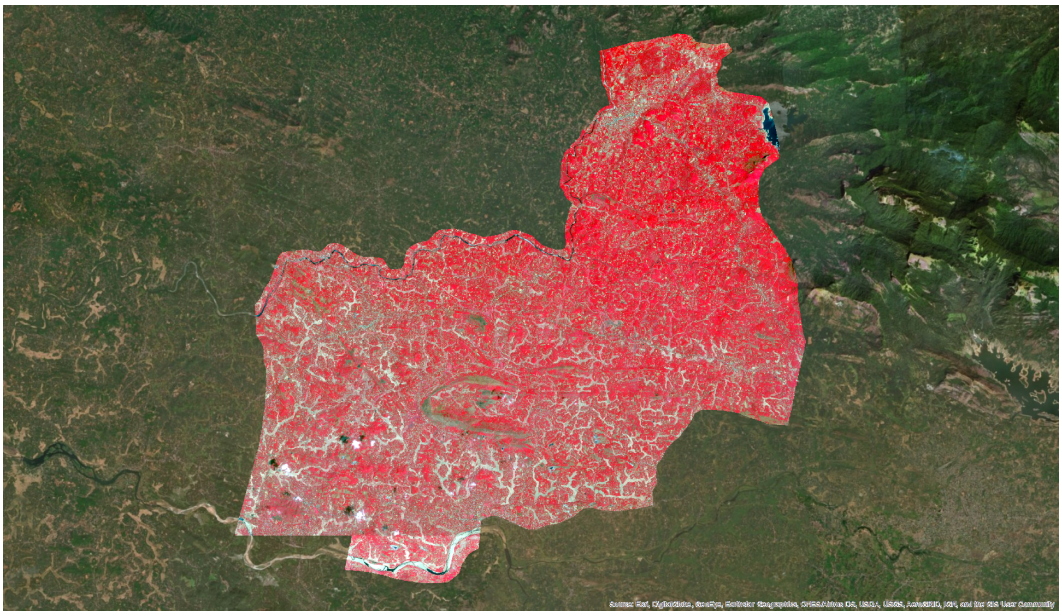


Figure 2.1: *The Pre-processed and Clipped Sentinel - 2 Image of the Kanjirappuzha Study Area displayed in Standard FCC (Red=NIR, Green=Red, Blue=Green).*

2.3 Extraction of Training Data

The training and testing data were simply extracted by creating regions of interest in the image where we know the ground class label and extracting these ROIs and saving them in CSV format. The ground class labels were assigned to these pixels simply by visual interpretation.

2.4 Defining the Classifier

Once the data was prepared, the classifier was now defined. The multi layer perceptron imported from open source Python libraries had the following structure:

1. Input layer with 10 nodes.
2. Hidden Layer 1 with 8 nodes.
3. Hidden Layer 2 with 16 nodes.
4. Hidden Layer 3 with 32 nodes.
5. Hidden Layer 4 with 64 nodes.
6. Hidden Layer 5 with 32 nodes.
7. Hidden Layer 6 with 16 nodes.
8. Hidden Layer 7 with 8 nodes.
9. Output Layer with 6 nodes.

The ReLU (Rectilinear Unit) activation unit was used in all the layers except the output layer where the sigmoid activation function was used.

The cross entropy loss function for 'M' classes was used for computing the cost for each iteration.

The gradient descent algorithm was used for back-propagation to compute the gradients and update the parameters in each iteration.

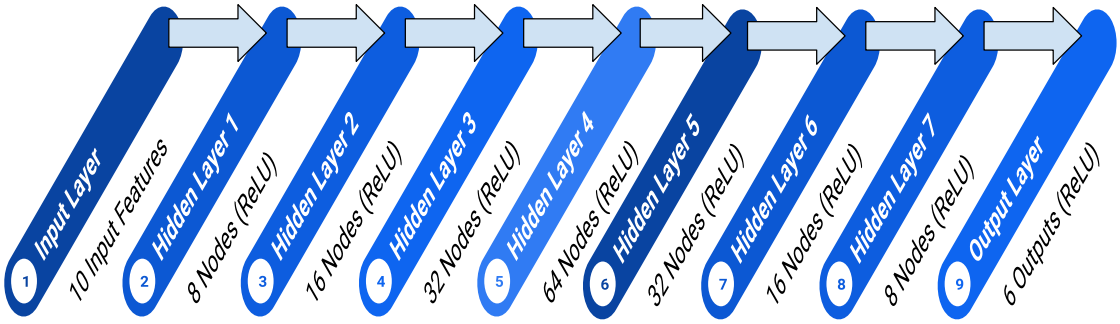


Figure 2.2: *Structure of the Multi Layer Perceptron Classifier.*

2.5 Training the Classifier

Once the classifier was defined, it was trained using the training data that was already prepared. The hyperparameters were adjusted each time to get the best possible training and testing accuracy.

2.6 Classifying the Data

Now that the classifier is trained on the training data and the optimum parameters have been computed, this trained classifier was used to classify all the pixels in the pre-processed raster. The output raster containing the class labels for each pixel was then georeferenced and exported using open source Python libraries.

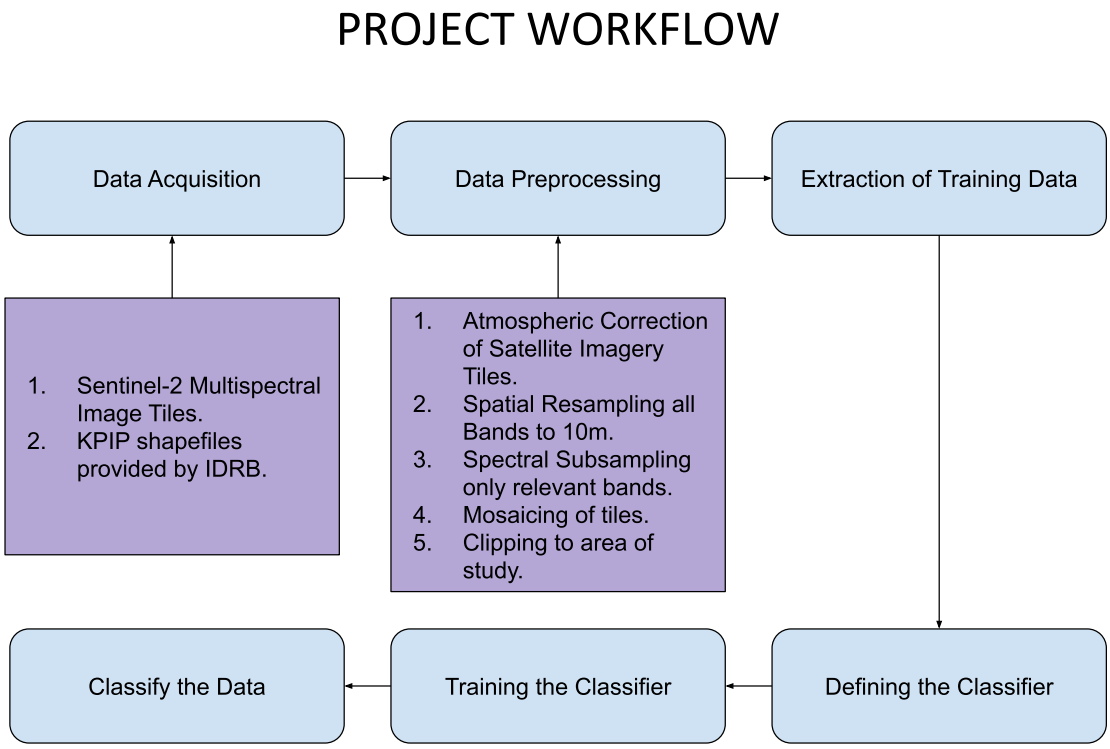


Figure 2.3: *Process Flow Chart of the Project.*

Chapter 3

Source Code

The source code for this project was prepared completely in Python 3.7.

```
# -*- coding: utf-8 -*-
"""
Created on Wed Jul  3 14:04:40 2019

@author: Yeshwanth
"""

import os
import glob
import time
import numpy as np
import pandas as pd
from pathlib import Path
from sklearn.neural_network import MLPClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, classification_repo
import matplotlib.pyplot as plt
from osgeo import gdal

os.chdir('H:\\M_Tech\\Lab_backup\\HP_Z4_workstation\\SC18M031\\Semester_3\\Outrea

#LOADING DATASET
kpip_watershed_dataset = gdal.Open("H:\\M_Tech\\Lab_backup\\HP_Z4_workstation\\SC
#kpip_watershed_dataset = gdal.Open("Data\\Sentinel2_Imagery\\final_imagery_kpip
kpip_img = kpip_watershed_dataset.ReadAsArray()
kpip_img1 = np.zeros((kpip_img.shape[1],kpip_img.shape[2],kpip_img.shape[0]))

for i in range(kpip_img.shape[0]):
```

```
    kpip_img1[:,:,i] = kpip_img[i,:,:]
#     kpip_img1[:,:,i] = (kpip_img1[:,:,i] - np.min(kpip_img1[:,:,i]))/(np.max(kp

kpip_img2 = kpip_img1.reshape(kpip_img.shape[1]*kpip_img.shape[2],kpip_img.shape

scaler = StandardScaler()
scaler.fit(kpip_img2)
kpip_img2 = scaler.transform(kpip_img2)


#LOADING LABELLED SAMPLES
tr_sample_files = glob.glob('ROIs\\new_trial\\*.csv')
tr_sample_paths = []
for string in tr_sample_files:
    path = os.path.abspath(string)
    path1 = Path(path)
    tr_sample_paths.append(path1)

labelled_samples = {}
cols = []
for i in range(1,11):
    cols.append('B'+str(i))
for i in range(len(tr_sample_paths)):
    class_name = os.path.splitext(os.path.basename(tr_sample_files[i]))[0]
    labelled_samples[class_name] = pd.read_csv(tr_sample_paths[i],
                    skiprows=[0,1,2,3,4,5,6,8],
                    skipinitialspace=True,
                    usecols=cols,
                    )

#Number of labelled samples, features and classes.
ns = 0
df_list = []
for key, df in labelled_samples.items():
    ns += len(df)
    df_list.append(df)

nf = labelled_samples['ayacut'].shape[1]
nc = len(labelled_samples)

#Sample Matrix.
data_x = np.array(pd.concat(df_list,ignore_index=True))
data_x = scaler.transform(data_x)

#Target Matrix.
```

```
data_y = np.zeros((ns,nc))

df = None
start = 0
end = 0
cc = []
for i in range(len(df_list)):
    df = df_list[i]
    end += len(df)
    data_y[start:end,i] = 1
    cc.append((start,end))
    start += len(df)

#Splitting into training and testing data.
x_train, x_test, y_train, y_test = train_test_split(data_x, data_y)

#Defining Classifier.
iters = 2000
clf = MLPClassifier(hidden_layer_sizes=(8,16,32,64,32,16,8),
                    max_iter=iters,
                    n_iter_no_change=50,
                    verbose=True)

st = time.time()
#Training classifier.
clf.fit(x_train, np.argmax(y_train,axis=1))
test_predictions = clf.predict(x_test)

print('Confusion Matrix:\n',confusion_matrix(np.argmax(y_test,axis=1), test_pred
print('Accuracy Score:',accuracy_score(np.argmax(y_test,axis=1), test_predictions

#Making Predictions.
preds = clf.predict(kpip_img2)
preds2 = preds.reshape(kpip_img.shape[1],kpip_img.shape[2])
plt.figure(figsize=(10,10))
plt.imshow(preds2)

#Writing as a Geotiff.
geo_trf = kpip_watershed_dataset.GetGeoTransform()
proj = kpip_watershed_dataset.GetProjection()

def write_geotiff(fname,data,geotransform,projection):
    driver = gdal.GetDriverByName('GTiff')
    rows,cols = data.shape
    dataset = driver.Create(fname,cols,rows,1,gdal.GDT_UInt32)
```

```
    dataset.SetGeoTransform(geotransform)
    dataset.SetProjection(projection)
    band = dataset.GetRasterBand(1)
    band.WriteArray(data)
    dataset = None

write_geotiff("Results\\new_trial1\\envi_mlp\\kpip_classified_mlp_{0}_iters.tiff"
              preds2,
              geo_trf,
              proj)

et = time.time()

print('Time elapsed: {0} seconds'.format(et-st))
```

Chapter 4

Results and Inferences

4.1 Inferences

1. It could be observed that upon increasing the number of iterations, the classification result becomes more representative of the ground truth. However, when the number of iterations is increased beyond a certain limit, there is no significant increase in accuracy. The optimal number of iterations for the defined classifier was found to be 2000.

2. Upon increasing the learning rate, the algorithm converges faster to the optimum parameter values. However, if the learning rate is increased too much, the algorithm fails to converge.

3. Increasing the number of layers or nodes per layer also improved the classification result however, at the expense of increased memory consumption (space complexity).

4.2 Results

Confusion Matrix:

Actual, Preds	Ayacut,	Barren,	Builtup,	Cloud,	Forest,	Water,	All
Ayacut,	2629,	0,	1,	0,	0,	0,	2630
Barren,	0,	2018,	2,	0,	1,	0,	2021
Builtup,	7,	11,	2091,	0,	0,	1,	2110
Cloud,	0,	0,	0,	464,	0,	0,	464
Forest,	0,	0,	0,	0,	2070,	0,	2070
Water,	0,	1,	0,	0,	0,	2018,	2019
All,	2636,	2030,	2094,	464,	2071,	2019,	11314

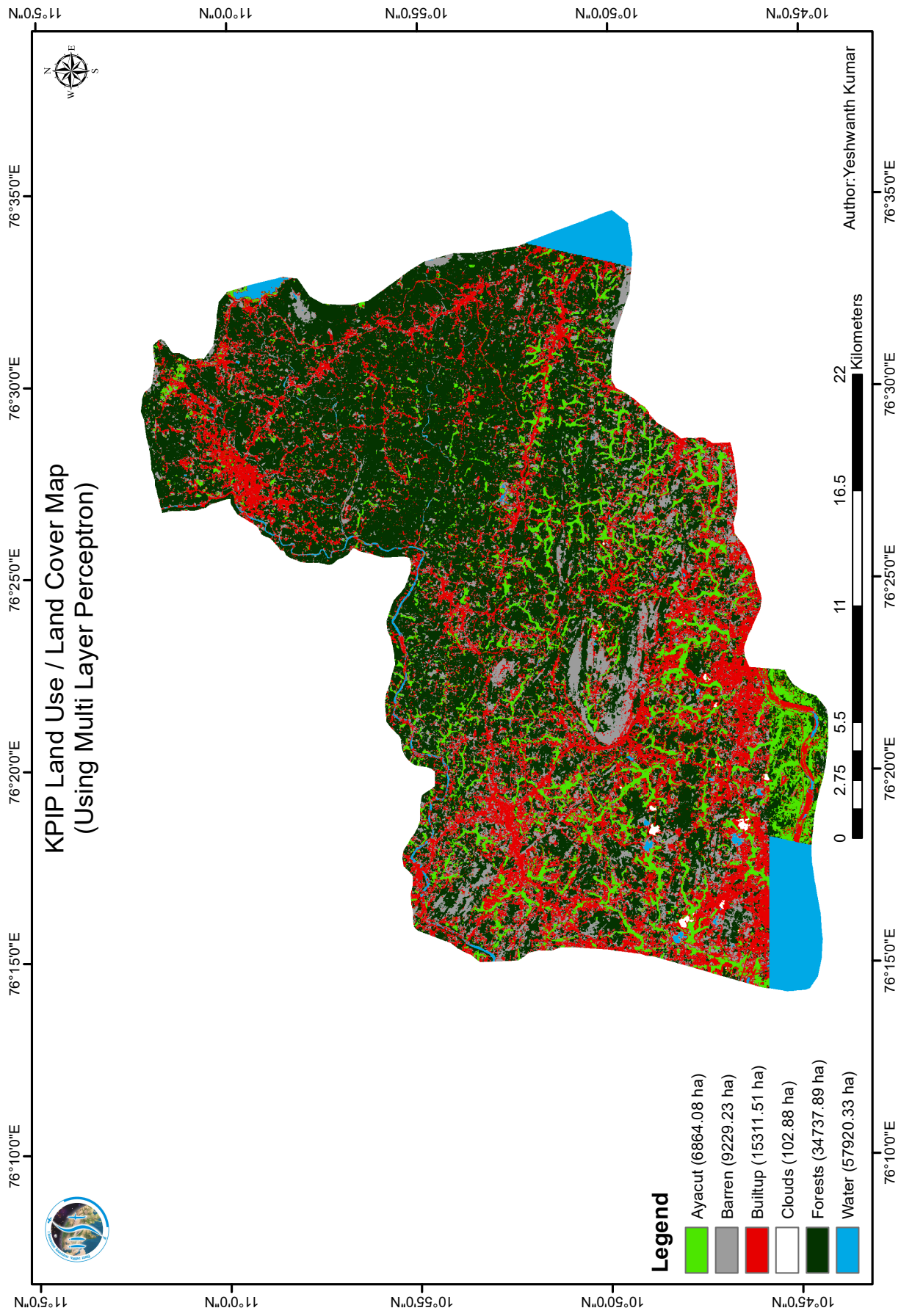
Classification Report:

Classification Report:

	precision	recall	f1-score	support
0	1.00	1.00	1.00	2630
1	0.99	1.00	1.00	2021
2	1.00	0.99	0.99	2110
3	1.00	1.00	1.00	464
4	1.00	1.00	1.00	2070
5	1.00	1.00	1.00	2019
micro avg	1.00	1.00	1.00	11314
macro avg	1.00	1.00	1.00	11314
weighted avg	1.00	1.00	1.00	11314

Accuracy Score: 91.27%

The results of the classification algorithm were published as maps using ArcMap 10.4.1.


KPIP Land Use / Land Cover Map
(Using Multi Layer Perceptron)
Author:Yeshwanth Kumar
Legend
Ayacut (6864.08 ha)
Barren (9229.23 ha)
Builtup (15311.51 ha)
Clouds (102.88 ha)
Forests (34737.89 ha)
Water (57920.33 ha)
0 2.75 5.5 11 16.5 22
Kilometers
76°10'0"E
76°15'0"E
76°20'0"E
76°25'0"E
76°30'0"E
76°35'0"E
10°45'0"N
10°50'0"N
10°55'0"N
11°0'0"N
11°5'0"N

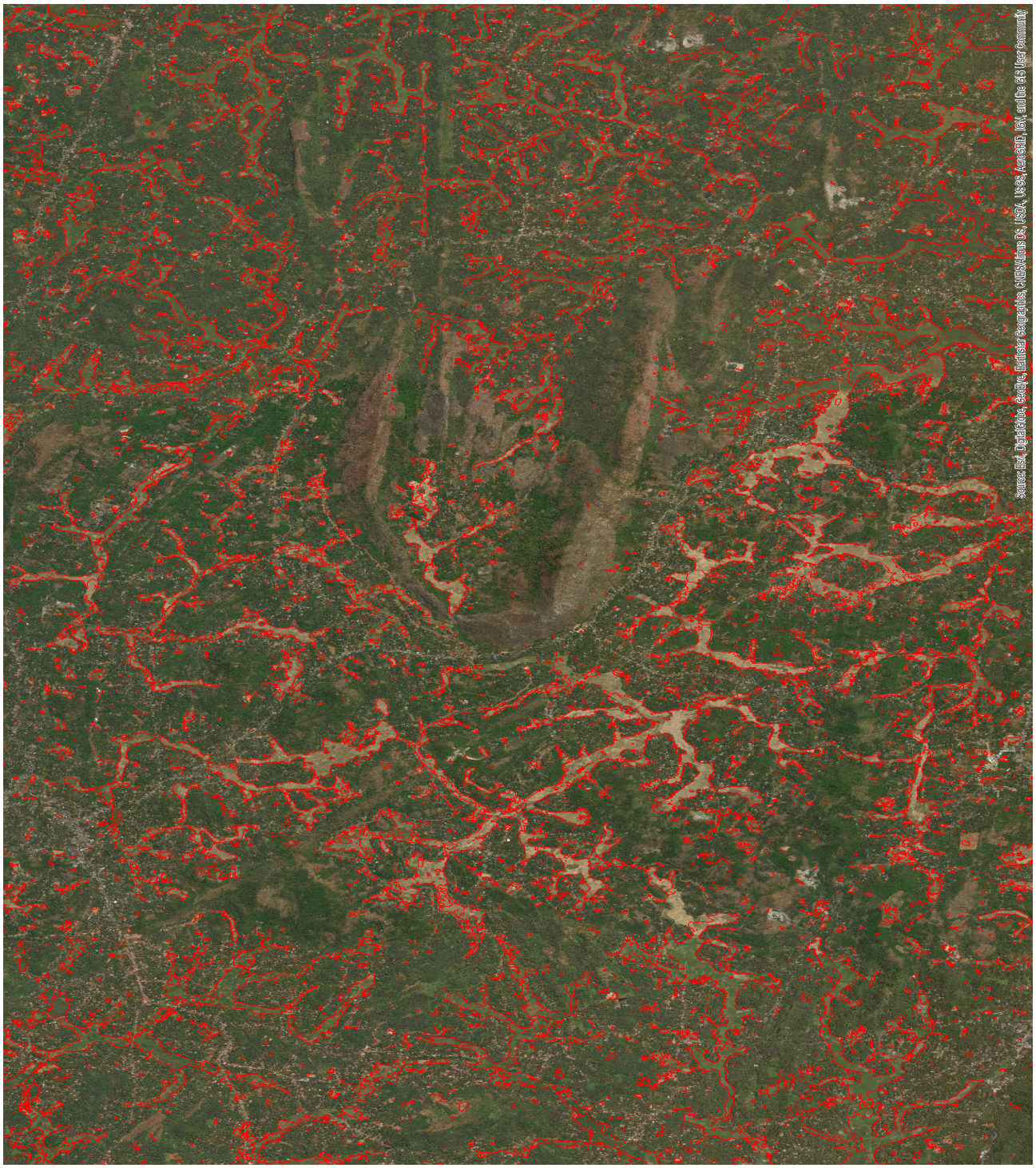


Figure 4.2: *Ayacut Regions in the Kanjirappuzha Study Region.*

Chapter 5

School Visits

In order to create awareness regarding space technology and research as a prospective career path to young minds, it was required that two government schools had to be visited and programmes must be conducted there to encourage students and teachers to foray into space technologies.

5.1 Schools Visited

The schools visited were:

1. Presidency Girls Higher Secondary School, Egmore, Chennai - 600008
Ph. No. : 044 2819 0096

2. Dr. Ambedkar Government Higher Secondary School, Egmore, Chennai - 600008
Ph. No. : 044 2841 3641

5.2 Activities Carried Out

In each school, students of classes XI and XII were given lectures explaining the various trends in the geospatial domain and how GIS technologies are influencing our daily lives (such as weather forecasting, disaster management, GPS routing services for package deliveries, BIM in construction industry, etc.).

Furthermore, they were made aware of open source geospatial data portals such as BHU-VAN, Earth Explorer, etc. They were also given a brief introduction as to how satellites work towards solving a lot of everyday problems.

The students were also introduced to various domains of geoinformatics such as remote sensing, GIS, LiDAR, Microwave remote sensing, etc. They were also given a hands on introduction on using open source software tools such as Python, QGIS, SNAP toolbox, Cloudcompare, etc.

Chapter 6

Conclusion

Hence, the multi layer perceptron algorithm was successfully implemented and the land use / land cover map of the Kanjirappuzha study region was generated using this algorithm. This output was in turn used to estimate the area of the ayacut lands in the region of study. The performance of the model for different combinations of hyper-parameters was observed and understood. The results of the classification were exported as maps and presented in the report with proper layout elements.

Also, two government schools were visited and the students were made aware of the various geospatial technologies and space research activities currently trending in India and the rest of the world with an intent to encourage students and teachers to take up space technology based research as a prospective career path.