OPEN SOURCE CANOPY CLASSIFICATION IN THE STATE OF GEORGIA

CREATING A REPRODUCIBLE METHOD FOR THE CLASSIFICATION OF CANOPY USING NAIP IMAGERY AND OPEN SOURCE PYTHON LIBRARIES – PRELIMINARY RESULTS

BACKGROUND

- **Deforestation:**
  - loss of forested lands leads to increased CO_2 being placed into the atmosphere while simultaneously eliminating carbon storage (Bala, Govindasamy, et al. 2007)
  - Smaller scales it leads to both increased runoff rates and subsequently increased erosion, especially in areas where no plant reclamation is initiated (Benito, E., et al, 2003)
- **Monitoring:**
  - Large scale monitoring is increasingly time consuming.
  - Commercial software dedicated to completing these tasks such as eCognition (Trimble Inc.) or Textron Systems Feature Analyst (Textron Systems 2010) are expensive and closed source.
- **Previous studies:**
  - GFC Canopy Study
    - Textrons Feature Analyst
  - PyTorch, Keras - Tensor Flow, Orfeo Toolbox

NAIP IMAGERY

- **National Agricultural Imagery Program**
- Collected by U.S. Department of Agriculture (USDA) aerial photography division during growing seasons.
- 1 m resolution.
  - Now 0.6m after 2019
  - 3-Band
    - Red, Green, Blue
  - 4-Band
    - Red, Green, Blue, Near Infrared (NIR)
- Preprocessing quality control removes any image that has more than 10% cloud cover per quarter quad rendering the need for a cloud mask negligible.

THE CASE FOR OPEN SOURCE DEVELOPMENT

- **What is open source?**
  - Open source products include permission to use the source code, design documents, or content of the product.
  - ‘Guarantees access to the source code for audit and modification and the ability to redistribute the software with no additional costs.’ per OSGeo
- **Open Source Geospatial Foundation (OSGeo)**
  - ‘A not-for-profit organization whose mission is to foster global adoption of open geospatial technology by being an inclusive software foundation devoted to an open philosophy and participatory community driven development.’
- Lack of insight into the inner workings of commercial software leads to uncertainty about validity of results (Sonnenburg 2007).
- Leads to increased collaboration between researchers, and greater transparency (Sonnenburg 2007).
- Allows for reproducibility and modification to fit different needs (Sonnenburg 2007).

SUPERVISED CLASSIFICATION

- Widely used robust method for approaches for classification
- Uses training data to create classifiers which are then in turn used to predict and learn the characteristics in unclassified data. (Belgiu & Drăguţ 2016)
- Popular types:
  - Support Vector Machines (SVM)
  - Artifical Nueral Network (ANN)
  - Random Forests (RF)

PYTHON & PYTHON PACKAGES

- A robust language suitable for automating, machine learning, and statistical analysis.
- Packages used:
  - Geospatial Data Abstraction Library (GDAL)
  - NumPy
  - Scikit-learn

PYTHON & PYTHON PACKAGES

- **Geospatial Data Abstraction Library (GDAL)**
  - 'GDAL is a C++ translator library for more than 200 raster and vector geospatial data formats.' – OSGeo
  - Core features as detailed by OSGeo
    - Reading and writing of raster and vector geospatial formats
    - Data format translation
    - Geospatial processing: subsetting, image warping, reprojection, mosaicing, tiling, DEM processing.
  - Python API

- **NumPy**
  - 'NumPy is the fundamental package for scientific computing with Python'
  - Creation and processing of arrays or matrices
  - Both GDAL's python API and Scikit-learn utilizes numpy

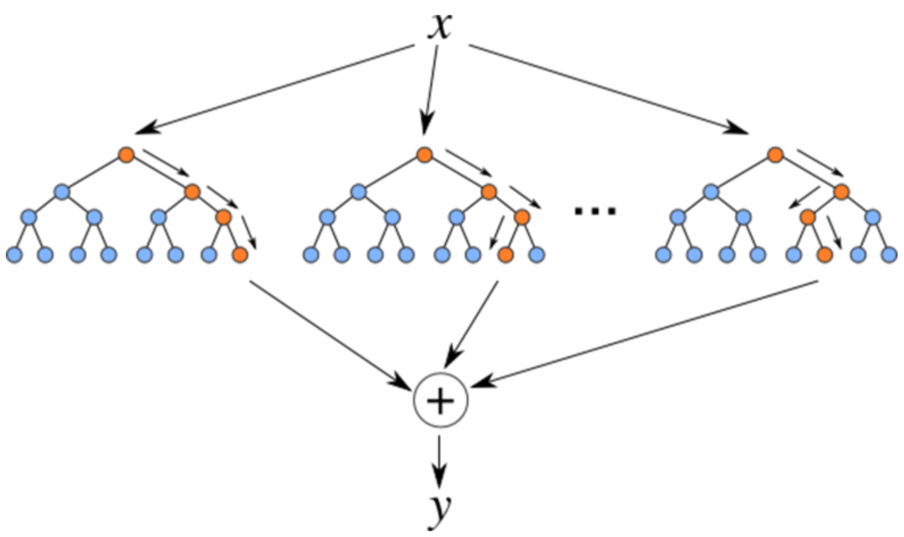
PYTHON & PYTHON PACKAGES

- **Scikit-learn**
  - Popular and robust machine learning (ML) python library capable of both regression and classification analysis.
  - Built on top of NumPy (van der Walt et al. 2011) and SciPy (Vertanen et al. 2019).

- **Why Scikit-learn?**
  - Other ML packages focus on ANN almost exclusively.
    - I.E. Keras-Tensorflow, Pytorch
  - Scikit-learn can run parallel across the central processing unit (CPU)
    - Others can run parallel across graphics processing units [GPU] , but only on Nvidia GPU's
  - Being built on NumPy and SciPy allows for increased efficiency when using geospatial data.
  - Thorough documentation

RANDOM FORESTS

- **Algorithm**
  - A combined multi tree predictor built upon bootstrap aggregating.
  - Each node is split using a random selection of features at the most optimal combination of features/split
  - The most popular class is chosen based of a vote after the specified number of trees are generated (Breiman 2001).
- **Reasons for choosing**
  - In cases of land-cover classification random forests is found to be as effective, if not more effective as other popular similar ensemble algorithms such as boosting and bagging (Breiman 2001, Gislason et al. 2006)
  - Considerably lighter load computationally than the popular Ada-boost algorithm (Freund & Schapire 1996).
  - `n_jobs` parameter allows for parallelzation across CPU cores
- **Coniderations**
  - Can use a considerable of memory as a matrix of number of samples (N) x number of trees (T) is stored in memory (Gislason et al. 2006)



EXTRA TREES CLASSIFIER

- **Algorithm**
  - Like RF in that it is a multi-tree predictor built using an ensemble of decision trees
  - ET classifier splits the nodes of the tree completely at random (Geurts et al. 2009)
  - ET uses the entirety of the sample and not just the bootstrap to grow trees, meaning each tree is independent or uncorrelated to the last (Guerts et al. 2009)
- **Reasons for choosing**
  - higher bias and lower variance than the standard RF
  - Suited for noisy or highly correlated datasets (Lawson et al. 2017, Xu et al. 2010).
  - ~ 3x faster computationally

WHY THE NIR BAND IS NEEDED

- **Two indices tested:**
  - Visually Atmospheric Resistant Index (VARI) – RGB index
  - Atmospheric Resistant Vegetation Index (ARVI) – NIR index
- Near Infra-Red band is absorbed by photosynthetically active vegetation, lesser by photosynthetically inactive vegetation, and reflected by bodies of water and impervious surfaces.
- 0.75 µm – 0.8 µm NIR wavelengths detects what RGB bands cannot (Tucker 1979).

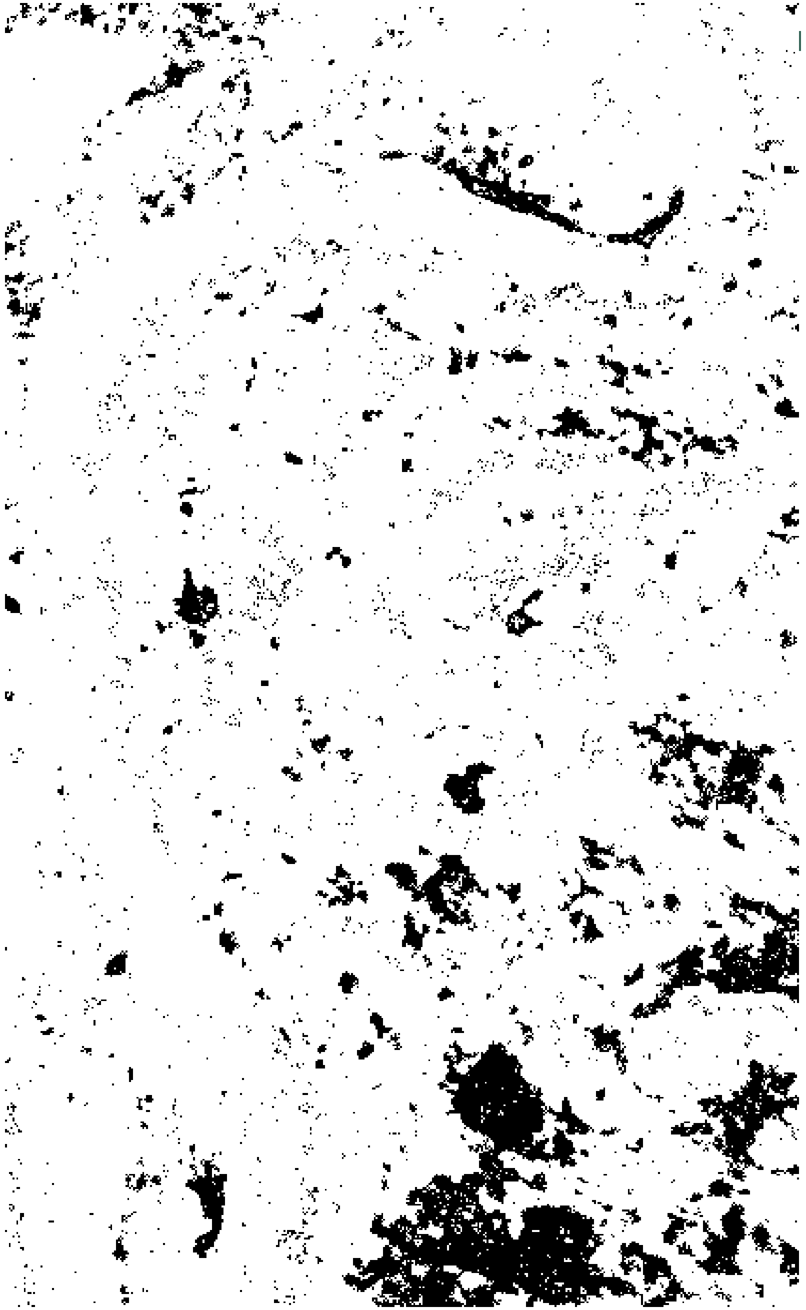
VISUALLY ATMOSPHERIC INDEX (VARI)

- Uses only visible light bands, making it potentially more accessible.
- Formula:

$$VARI = \frac{(Green - Red)}{(Green + Red - Blue)}$$

- Needs to be normalized between values 1 and -1 for classification:

```
def norm(array):
    array_min, array_max = array.min(), array.max()
    return ((1 - 0) * ((array - array_min) / (array_max - array_min))) + 1
```



Non-normalized VARI – No water detected

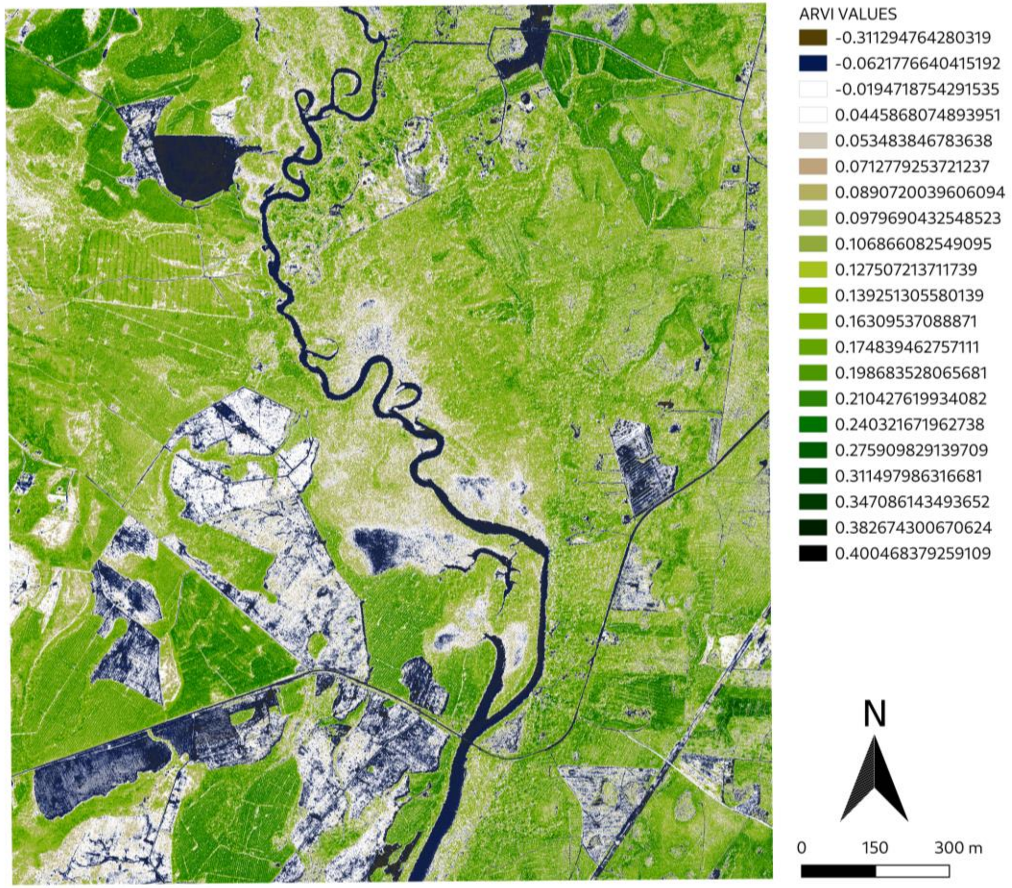
Normalized VARI – Water detected but has high error

ARVI – Little to no error with water detection

ATMOSPHERIC RESISTANT VEGETATION INDEX (ARVI)

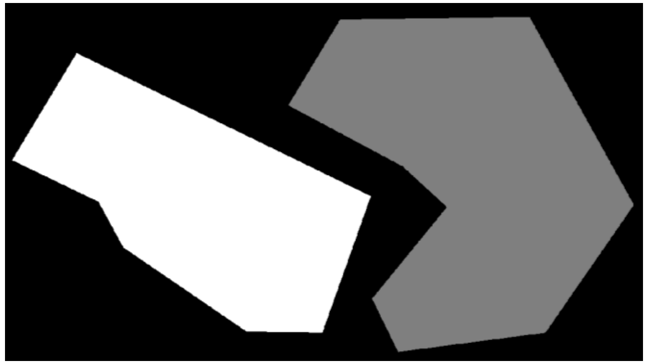
- **Atmospherically Resistant Vegetation Index (VARI)**
  - Creates an index that allows for higher variation between vegetation and other features to allow for more accurate identification
  - Near Infra-Red band is absorbed by photosynthetically active vegetation and reflected by bodies of water and impervious surfaces.
  - Formula:

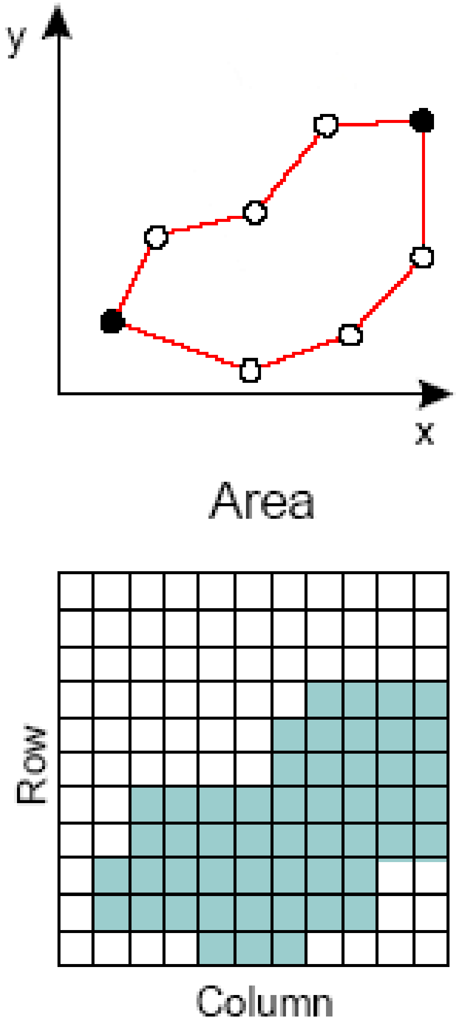
$$ARVI = \frac{(NIR - (2 * Red) + Blue)}{(NIR + (2 * Red) + Blue)}$$



METHODS – PREPARING DATA

- **Training Data**
  - Shapefile drawn in QGIS software with values of 1 and 2
    - 1: Non-canopy
    - 2: Canopy
  - Training data shapefile is rasterized with nodata values as zero





PARAMETER OPTIMIZATION

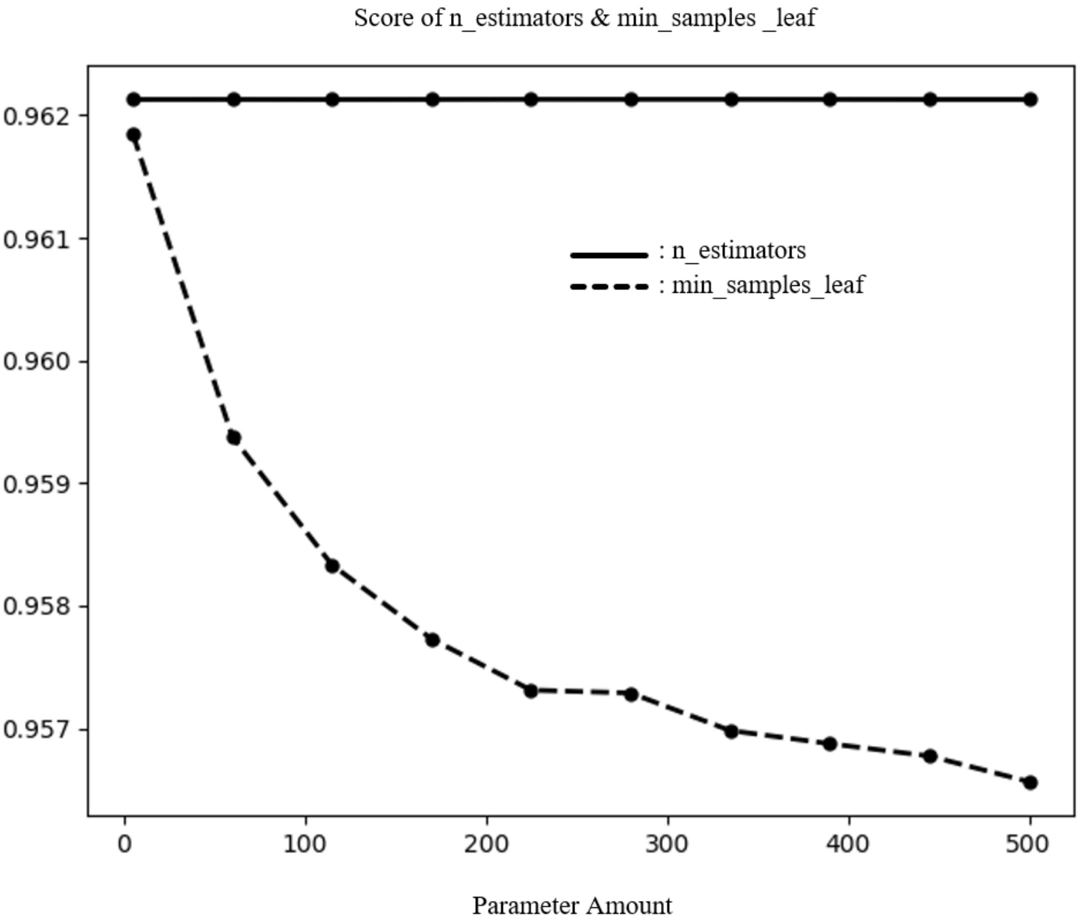
- 19 different parameters to adjust in the ET model
- `RandomizedSearchCV` or Randomized Search Cross-Validation used to find ideal parameters
- Parameters chosen:
  - n_estimators: number of trees generated in a forest
  - min_leaf_samples: samples required to split a node
- Training data split into test and training sets
  - Test: 33%

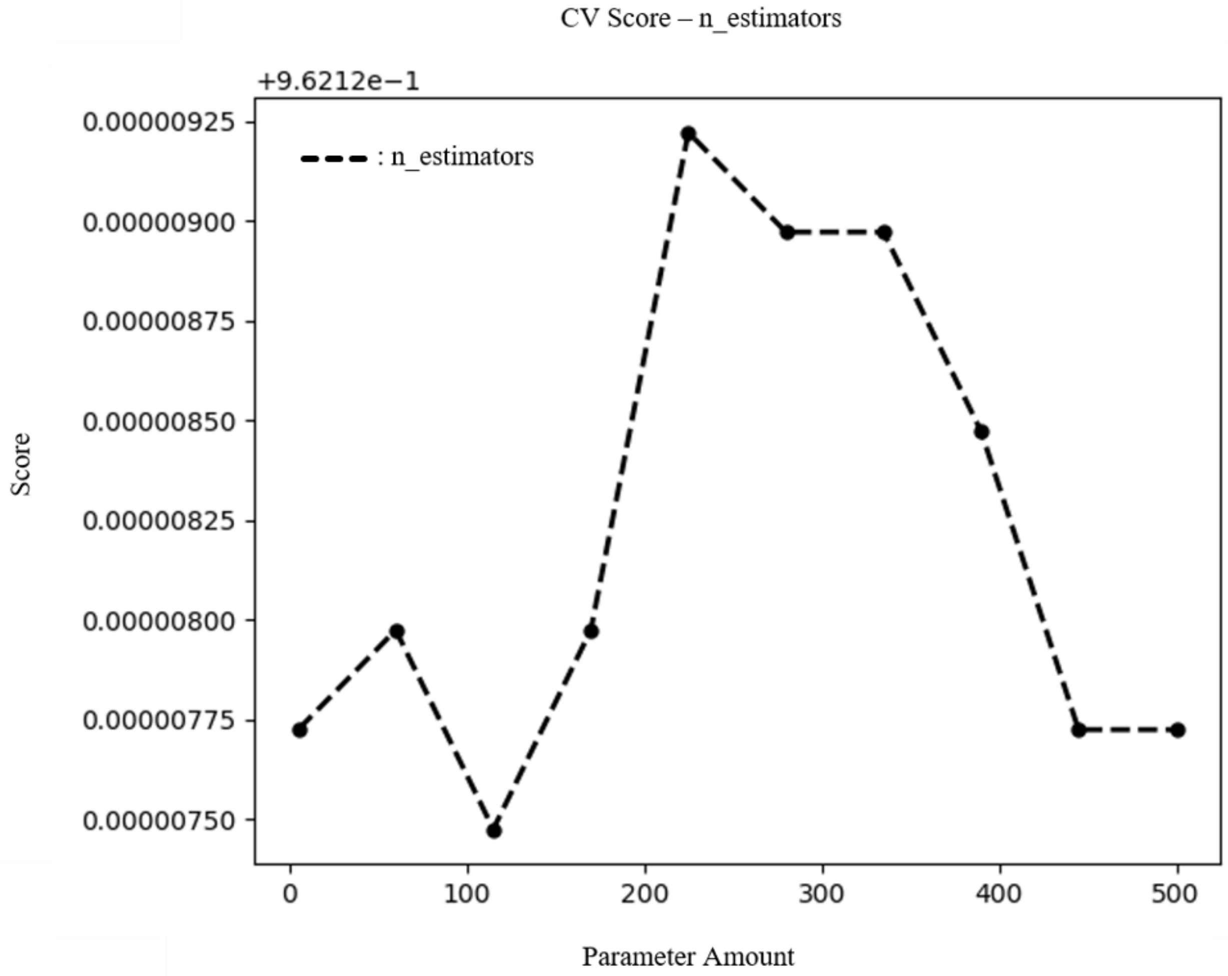
```python
def split_data(training_raster, training_fit_raster):

    y_raster = gdal.Open(training_raster)
    t = y_raster.GetRasterBand(1).ReadAsArray().astype(np.float32)
    x_raster = gdal.Open(training_fit_raster)
    n = x_raster.GetRasterBand(1).ReadAsArray().astype(np.float32)
    y = t[t > 0]
    X = n[t > 0]
    X = X.reshape(-1, 1)
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33)


    return X_train, X_test, y_train, y_test
```

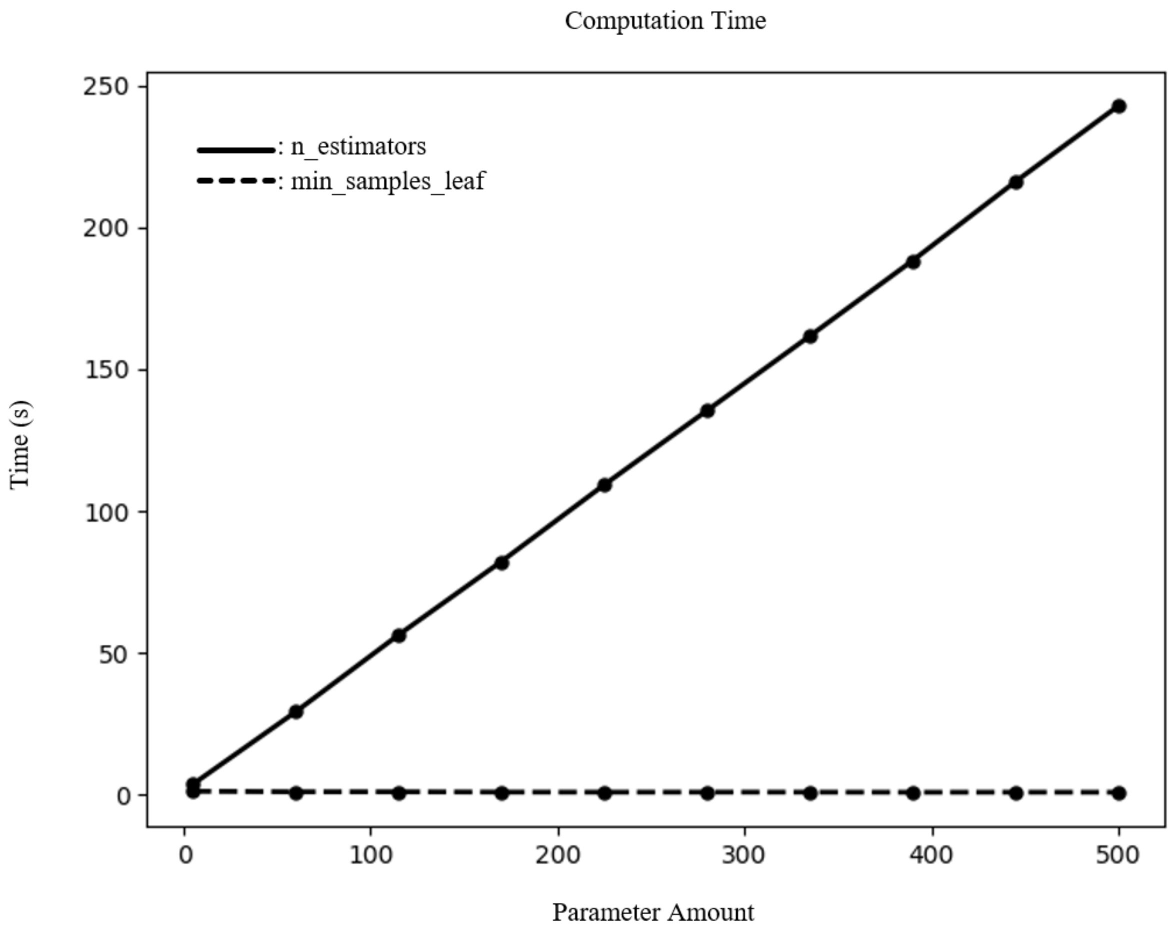
PARAMETER OPTIMIZATION

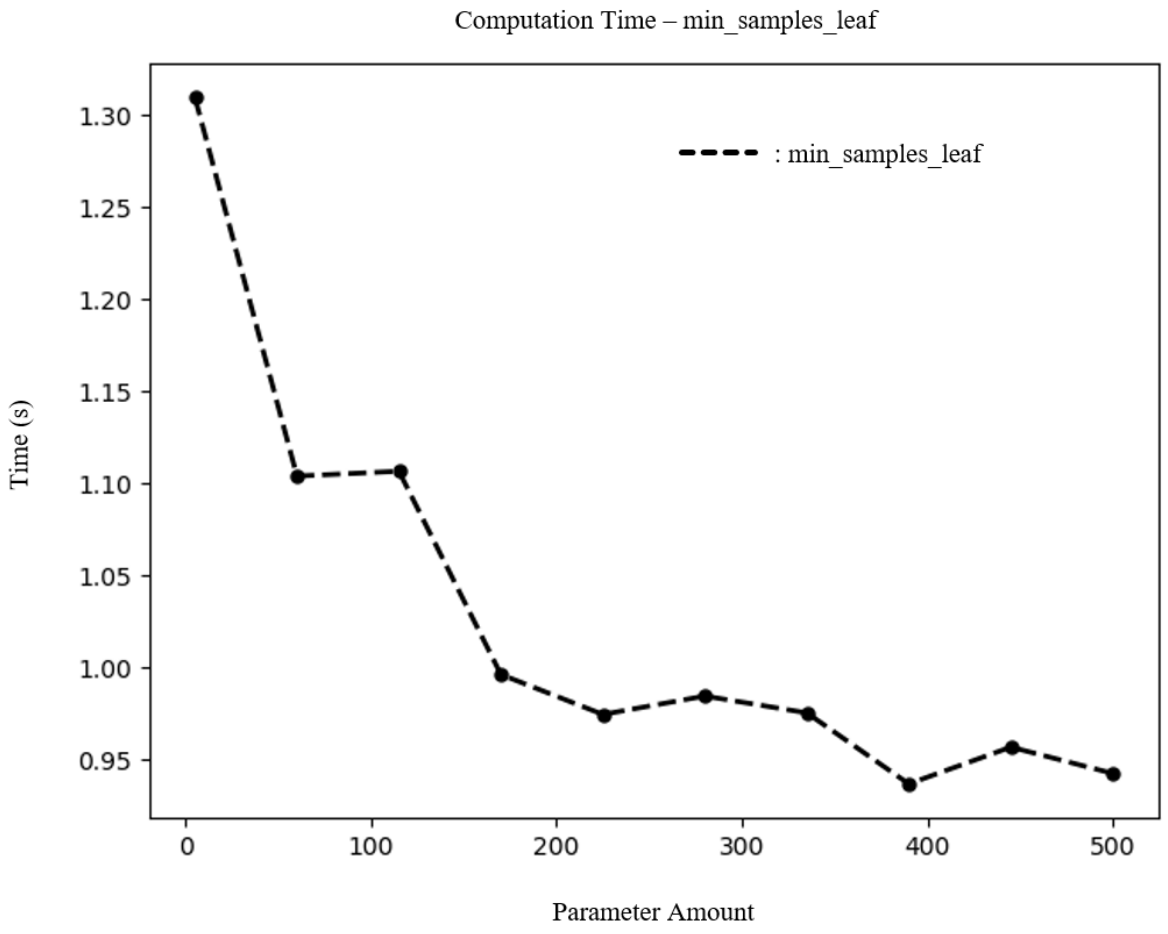
- A list of values is generated for each parameter.
- The values are then chosen at random and paired for cross validation.

```
def tune_hyperparameter(training_raster, training_fit_raster):

    y_raster = gdal.Open(training_raster)
    t = y_raster.GetRasterBand(1).ReadAsArray().astype(np.float32)
    x_raster = gdal.Open(training_fit_raster)
    n = x_raster.GetRasterBand(1).ReadAsArray().astype(np.float32)
    y = t[t > 0]
    X = n[t > 0]
    X = X.reshape(-1, 1)

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size =
    0.33)

    n_estimators = [int(x) for x in np.linspace(start=10, stop=500, num=10)]
    min_samples_leaf = [int(x) for x in np.linspace(start=10, stop=500, num=10)]
    random_grid = {
        'n_estimators': n_estimators,
        'min_samples_leaf': min_samples_leaf
    }
    etc = ExtraTreesClassifier(n_estimators=100, n_jobs=-1, max_features=None)
    clf = RandomizedSearchCV(etc, random_grid, random_state=0, verbose=3)
    clf.fit(X_test, y_test)

    print(clf.best_params_)
```


Score of n_estimators & min_samples _leaf
0.962
0.961
0.960
0.959
0.958
0.957
: n_estimators
: min_samples_leaf
0
100
200
300
400
500
Parameter Amount


CV Score – n_estimators
+9.6212e−1
0.00000925
0.00000900
0.00000875
0.00000850
0.00000825
0.00000800
0.00000775
0.00000750
: n_estimators
Score
0
100
200
300
400
500
Parameter Amount


Computation Time
: n_estimators
: min_samples_leaf
Time (s)
Parameter Amount
250
200
150
100
50
0
0
100
200
300
400
500


Computation Time – min_samples_leaf
: min_samples_leaf
Time (s)
Parameter Amount
1.30
1.25
1.20
1.15
1.10
1.05
1.00
0.95
0
100
200
300
400
500

METHODS – TRAINING MODEL

- **Extra Trees Classifier**
  - (X, y)
    - X contains features
    - y contains labels
  - Training data set is applied to proper ARVI raster, and subsequently applied to the rest of the dataset.

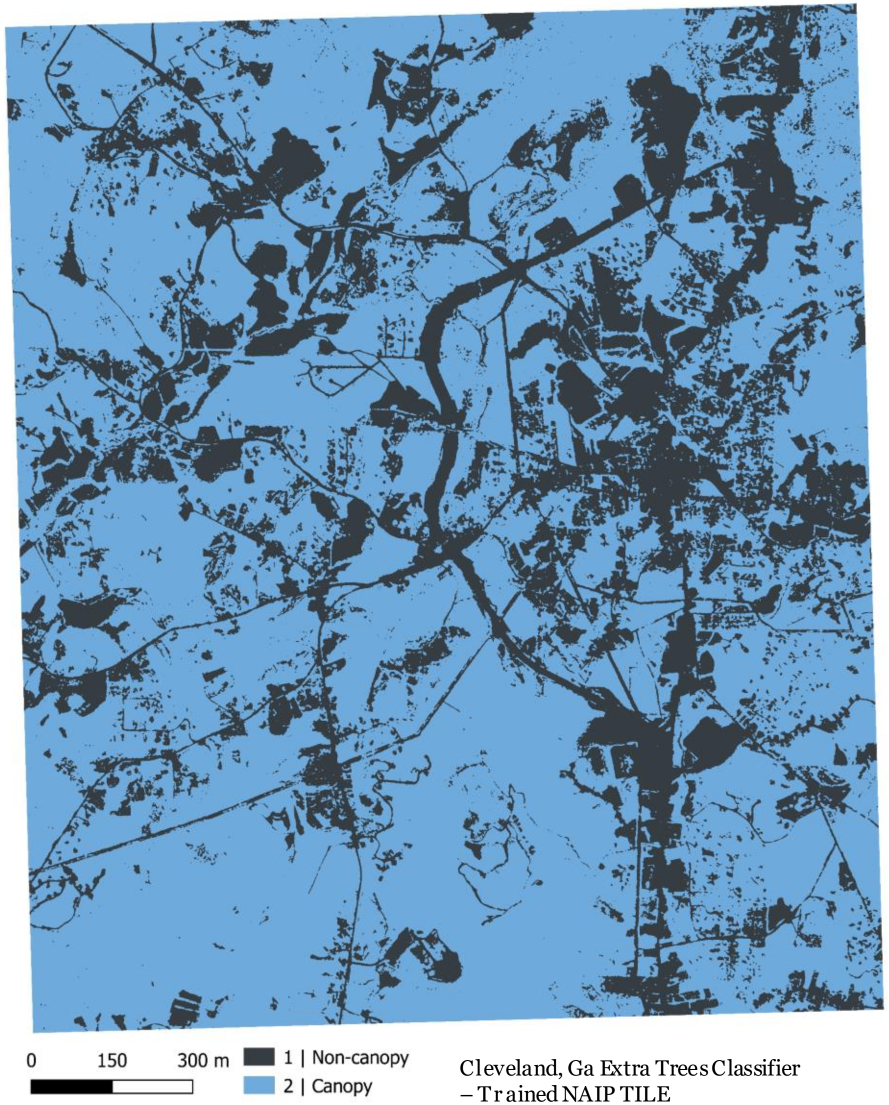
```
y_raster = gdal.Open(training_raster)
t = y_raster.GetRasterBand(1).ReadAsArray().astype(np.float32)
x_raster = gdal.Open(training_fit_raster)
n = x_raster.GetRasterBand(1).ReadAsArray().astype(np.float32)
y = t[t > 0]
X = n[t > 0]
X = X.reshape(-1, 1)
clf = ExtraTreesClassifier(n_estimators=41, n_jobs=-1,
                           max_features=None,
                           min_samples_leaf=5, class_weight={1: 2, 2: 0.5})
ras = clf.fit(X, y)
r = gdal.Open(in_raster)
class_raster = r.GetRasterBand(1).ReadAsArray().astype(np.float32)
class_array = class_raster.reshape(-1, 1)
ras_pre = ras.predict(class_array)
ras_final = ras_pre.reshape(class_raster.shape)
ras_byte = ras_final.astype(dtype=np.byte)
```

ADDITIONAL IMAGE PROCESSING

- 5x5 Median Filter applied to numpy array to smooth result and reduce noise.
- Boolean operator, only applied if `smoothing=True`.

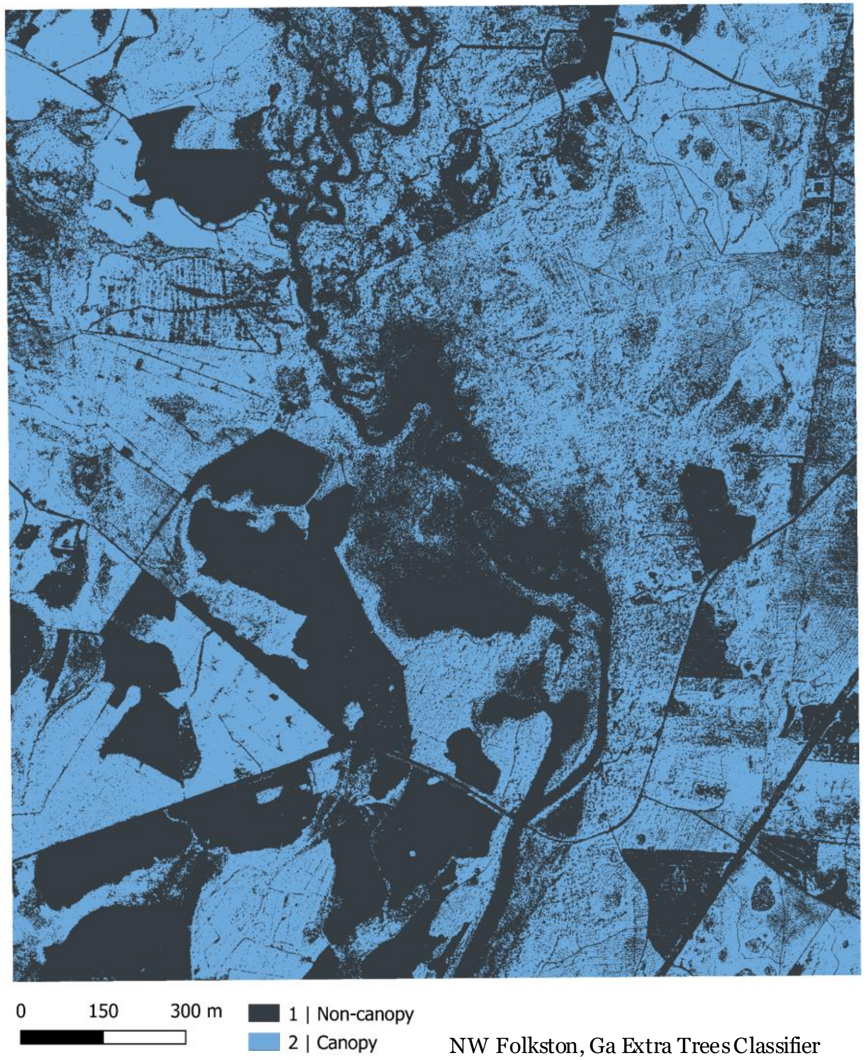
Smoothing=False

Smoothing=True

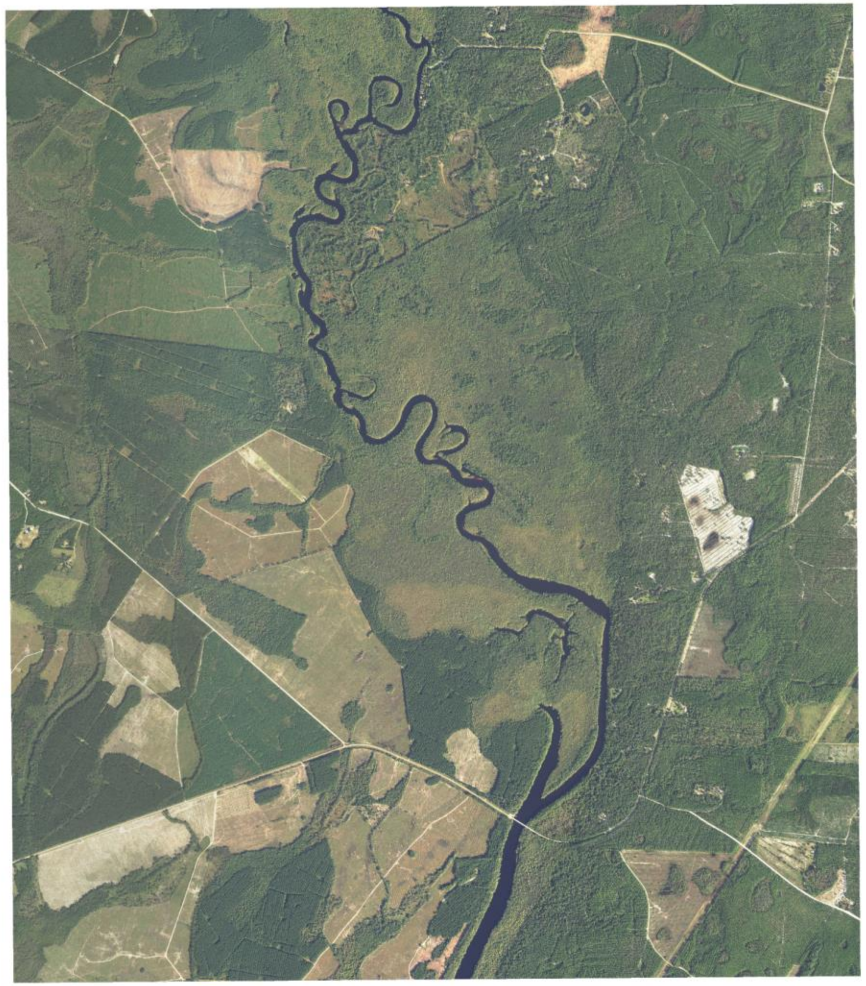


Cleveland, Ga Extra Trees Classifier – Trained NAIP TILE

2015 NAIP Imagery

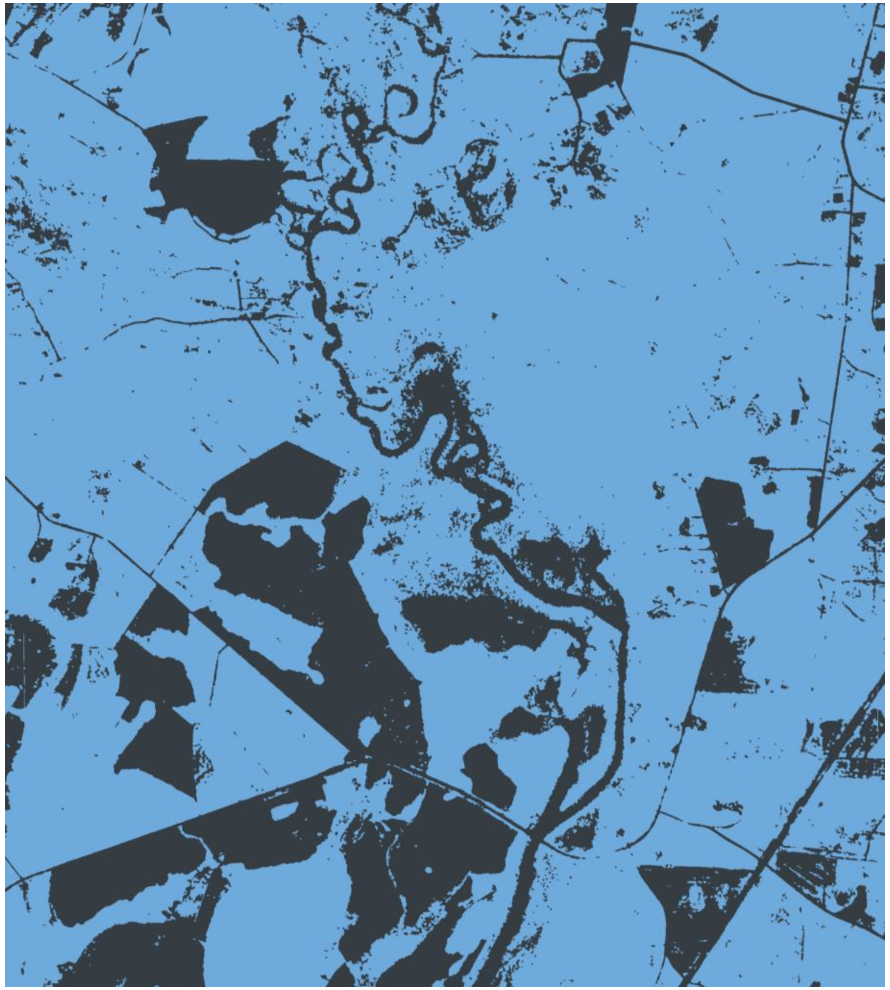


NW Folkston, Ga Extra Trees Classifier

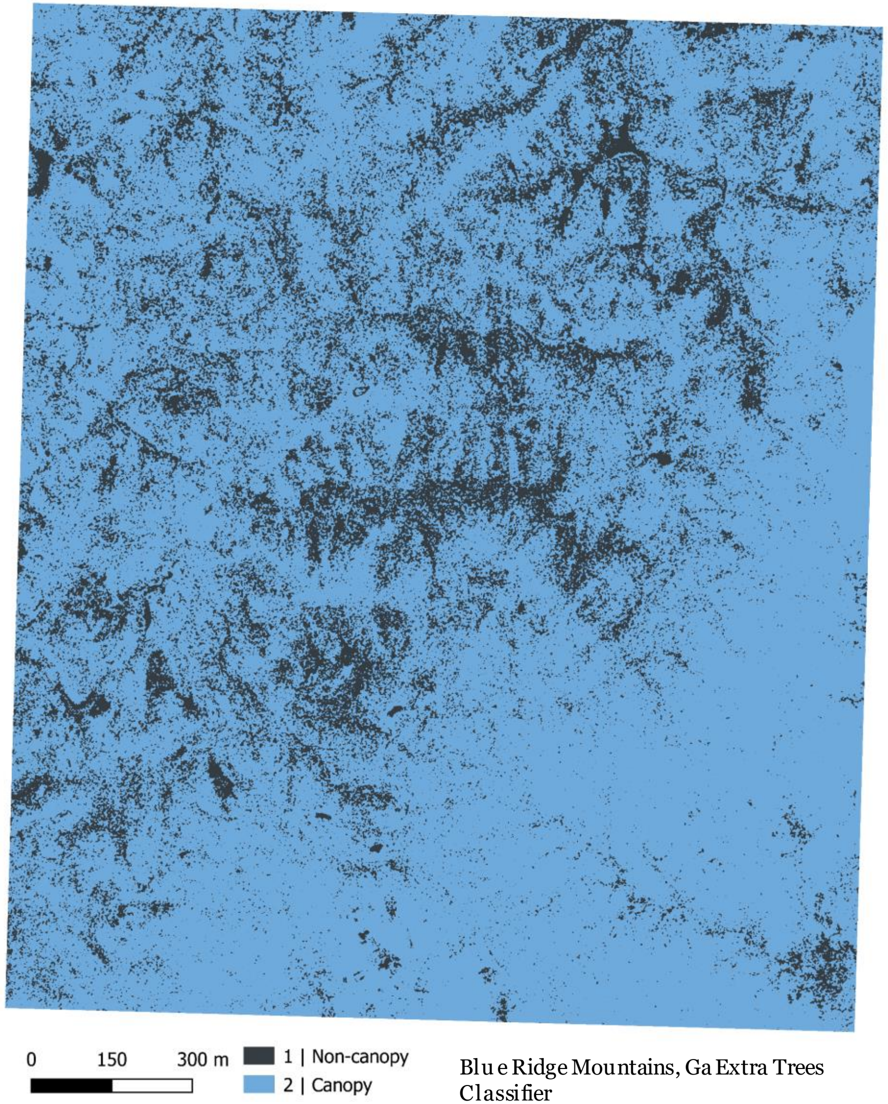


2015 NAIP Imagery

NW Folkston, Ga Extra Trees Classifier

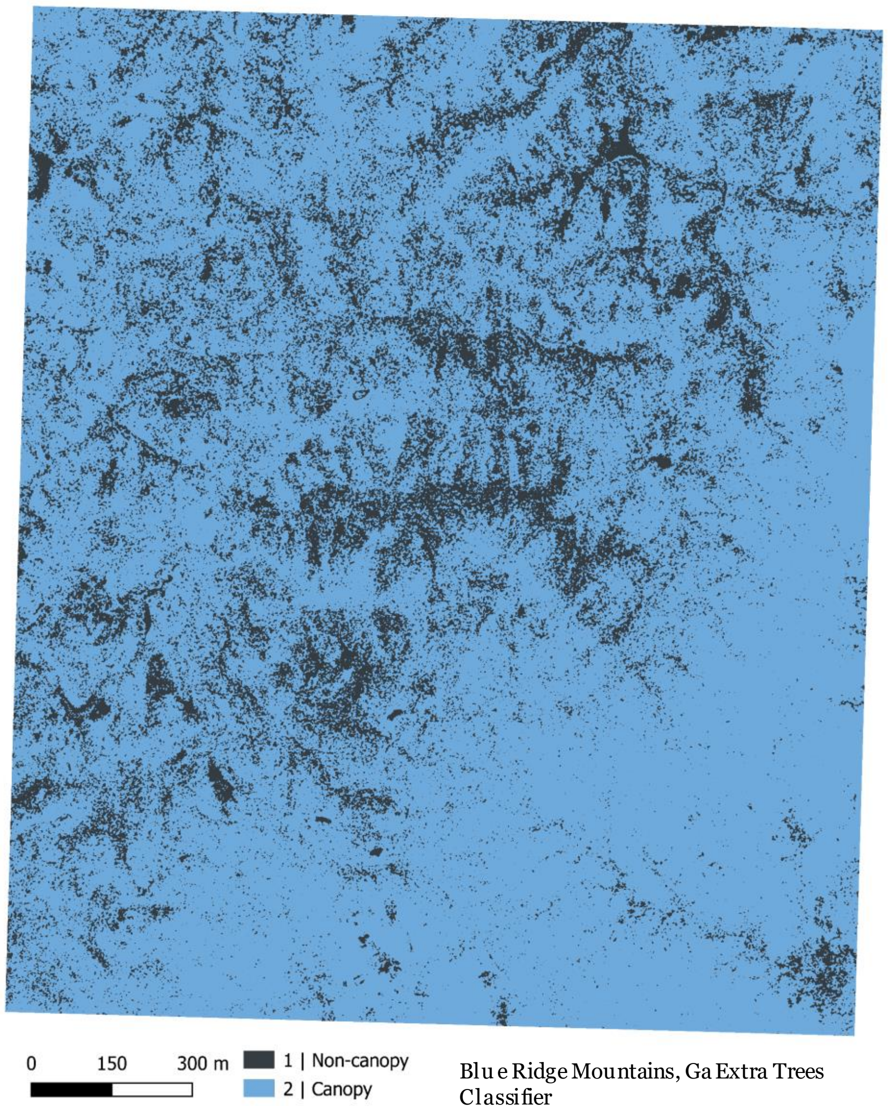


Textron Systems Feature Analyst

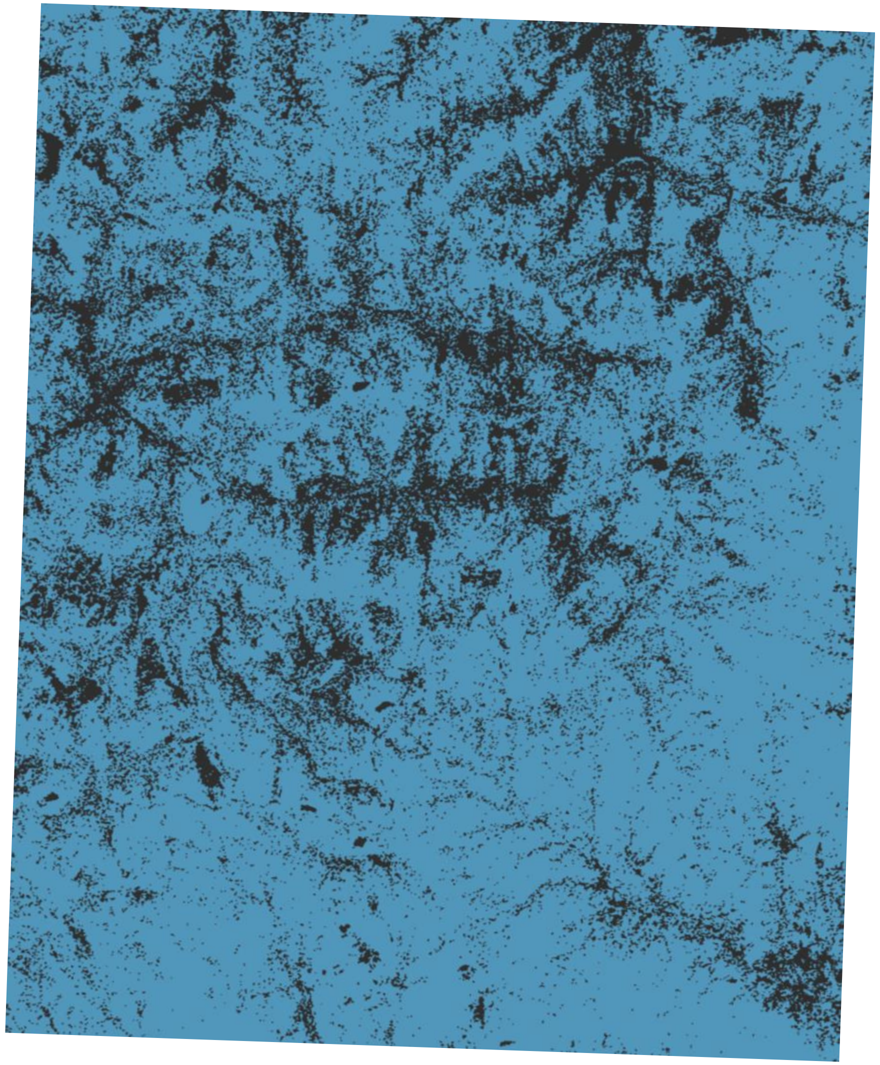


Blue Ridge Mountains, Ga Extra Trees Classifier

2015 NAIP Imagery



Blue Ridge Mountains, Ga Extra Trees Classifier



Textron Systems Feature Analyst

Atlanta, Ga Extra Trees Classifier

2015 NAIP Imagery

Atlanta, Ga Extra Trees Classifier

Textron Systems Feature Analyst

FUTURE - QUANTIFYING COMPARISONS

- Moving window comparison coefficient.
  - Fw = Index for moving window with window size w
  - w = window size
  - Tw = number of windows with window size w
  - a1-a2 = number of cells with category i in map 1 and map 2

$$F_w = \frac{1}{t_w} \sum_{s=1}^{t_w} \left[1 - \frac{\sum_{i=1}^{p} \left| a_{1i} - a_{2i} \right|}{2w^2} \right]$$

QUESTIONS?

CITATIONS

ABUJAYYAB, S. K., & KARAŞ, I. R. (2019). GEOSPATIAL MACHINE LEARNING DATASETS STRUCTURING AND CLASSIFICATION TOOL: CASE STUDY FOR MAPPING LULC FROM RASAT SATELLITE IMAGES. INTERNATIONAL ARCHIVES OF THE PHOTOGRAMMETRY, REMOTE SENSING & SPATIAL INFORMATION SCIENCES.
ANH, N. D., TUAN, V. A., & HANG, N. T. DEFORESTATION HOT-SPOT EXTRACTION FROM RADAR CHANGE RATIO (RCR) ANALYSIS OF SENTINEL-1 TIME SERIES DATA IN DAK G'LONG DISTRICT, DAK NONG PROVINCE.

BALA, G., CALDEIRA, K., WICKETT, M., PHILLIPS, T. J., LOBELL, D. B., DELIRE, C., & MIRIN, A. (2007). COMBINED CLIMATE AND CARBON-CYCLE EFFECTS OF LARGE-SCALE DEFORESTATION. PROCEEDINGS OF THE NATIONAL ACADEMY OF SCIENCES, 104(16), 6550-6555.

BASU, S., GANGULY, S., NEMANI, R. R., MUKHOPADHYAY, S., ZHANG, G., MILESI, C., ... & COOK, B. (2015). A SEMIAUTOMATED PROBABILISTIC FRAMEWORK FOR TREE-COVER DELINEATION FROM 1-M NAIP IMAGERY USING A HIGH-PERFORMANCE COMPUTING ARCHITECTURE. IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, 53(10), 5690-5708.

BENITO, E., SANTIAGO, J. L., DE BLAS, E., & VARELA, M. E. (2003). DEFORESTATION OF WATER-REPELLENT SOILS IN GALICIA (NW SPAIN): EFFECTS ON SURFACE RUNOFF AND EROSION UNDER SIMULATED RAINFALL. EARTH SURFACE PROCESSES AND LANDFORMS: THE JOURNAL OF THE BRITISH GEOMORPHOLOGICAL RESEARCH GROUP, 28(2), 145-155.

CUDA SEMANTICS — PYTORCH MASTER DOCUMENTATION. (2019). RETRIEVED JANUARY 22, 2020, FROM HTTPS://PYTORCH.ORG/DOCS/STABLE/NOTES/CUDA.HTML

DALLAQUA, F. B., FARIA, F. A., & FAZENDA, A. L. (2018, OCTOBER). ACTIVE LEARNING APPROACHES FOR DEFORESTED AREA CLASSIFICATION. IN 2018 31ST SIBGRAPI CONFERENCE ON GRAPHICS, PATTERNS AND IMAGES (SIBGRAPI) (PP. 48-55). IEEE.

ENVIRONMENTAL PROTECTION DIVISION, STATE OF GEORGIA'S ENVIRONMENT. (2009). RETRIEVED MARCH 21, 2020, FROM HTTPS://EPD.GEORGIA.GOV/SITES/EPD.GEORGIA.GOV/FILES/RELATED_FILES/SITE_PAGE/OBJECTIVE%202.PDF

GDAL/OGR CONTRIBUTORS (2019). GDAL/OGR GEOSPATIAL DATA ABSTRACTION SOFTWARE LIBRARY. OPEN SOURCE GEOSPATIAL FOUNDATION. URL HTTPS://GDAL.ORG

GEURTS, P., ERNST, D., & WEHENKEL, L. (2006). EXTREMELY RANDOMIZED TREES. MACHINE LEARNING, 63(1), 3-42.

GITELSON, A. A., KAUFMAN, Y. J., STARK, R., & RUNDQUIST, D. (2002). NOVEL ALGORITHMS FOR REMOTE ESTIMATION OF VEGETATION FRACTION. REMOTE SENSING OF ENVIRONMENT, 80(1), 76-87.

GITELSON, A. A., STARK, R., GRITS, U., RUNDQUIST, D., KAUFMAN, Y., & DERRY, D. (2002). VEGETATION AND SOIL LINES IN VISIBLE SPECTRAL SPACE: A CONCEPT AND TECHNIQUE FOR REMOTE ESTIMATION OF VEGETATION FRACTION. INTERNATIONAL JOURNAL OF REMOTE SENSING, 23(13), 2537-2562.

GPU SUPPORT | TENSORFLOW. (2020, JANUARY 13). RETRIEVED JANUARY 22, 2020, FROM HTTPS://WWW.TENSORFLOW.ORG/INSTALL/GPU

GRINGS, F., ROITBERG, E., & BARRAZA, V. (2019). EVI TIME-SERIES BREAKPOINT DETECTION USING CONVOLUTIONAL NETWORKS FOR ONLINE DEFORESTATION MONITORING IN CHACO FOREST. IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING.

JIANG, Z., HUETE, A. R., DIDAN, K., & MIURA, T. (2008). DEVELOPMENT OF A TWO-BAND ENHANCED VEGETATION INDEX WITHOUT A BLUE BAND. REMOTE SENSING OF ENVIRONMENT, 112(10), 3833-3845.

KAUFMAN, Y. J., & TANRE, D. (1992). ATMOSPHERICALLY RESISTANT VEGETATION INDEX (ARVI) FOR EOS-MODIS. IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, 30(2), 261-270.

LAWSON, E., SMITH, D., SOFGE, D., ELMORE, P., & PETRY, F. (2017). DECISION FORESTS FOR MACHINE LEARNING CLASSIFICATION OF LARGE, NOISY SEAFLOOR FEATURE SETS. COMPUTERS & GEOSCIENCES, 99, 116-124.

LI, M., LIEW, S. C., & KWOH, L. K. (2004, JULY). AUTOMATED PRODUCTION OF CLOUD-FREE AND CLOUD SHADOW-FREE IMAGE MOSAICS FROM CLOUDY SATELLITE IMAGERY. IN PROCEEDINGS OF THE XXTH ISPRS CONGRESS (PP. 12-13).

LIU, G. R., LIANG, C. K., KUO, T. H., & LIN, T. H. (2004). COMPARISON OF THE NDVI, ARVI AND AFRI VEGETATION INDEX, ALONG WITH THEIR RELATIONS WITH THE AOD USING SPOT 4 VEGETATION DATA. TERRESTRIAL, ATMOSPHERIC AND OCEANIC SCIENCES, 15(1), 15-31.

LO, C. P., & YANG, X. (2002). DRIVERS OF LAND-USE/LAND-COVER CHANGES AND DYNAMIC MODELING FOR THE ATLANTA, GEORGIA METROPOLITAN AREA. PHOTOGRAMMETRIC ENGINEERING AND REMOTE SENSING, 68(10), 1073-1082.

NAIP IMAGERY. (N.D.). RETRIEVED JANUARY 21, 2020, FROM HTTPS://WWW.FSA.USDA.GOV/PROGRAMS-AND-SERVICES/AERIAL-PHOTOGRAPHY/IMAGERY-PROGRAMS/NAIP-IMAGERY/

NAIP. RETRIEVED JANUARY 21, 2020, FROM HTTP://WWW.FSA.USDA.GOV/INTERNET/FSA_FILE/NAIP_2009_INFO_FINAL.PDF

NICKOLLS, J., BUCK, I., GARLAND, M., & SKADRON, K. (2008). SCALABLE PARALLEL PROGRAMMING WITH CUDA. QUEUE, 6(2), 40-53.

PAULI VIRTANEN, RALF GOMMERS, TRAVIS E. OLIPHANT, MATT HABERLAND, TYLER REDDY, DAVID COURNAPEAU, EVGENI BUROVSKI, PEARU PETERSON, WARREN WECKESSER, JONATHAN BRIGHT, STÉFAN J. VAN DER WALT, MATTHEW BRETT, JOSHUA WILSON, K. JARROD MILLMAN, NIKOLAY MAYOROV, ANDREW R. J. NELSON, ERIC JONES, ROBERT KERN, ERIC LARSON, CJ CAREY, İLHAN POLAT, YU FENG, ERIC W. MOORE, JAKE VANDERPLAS, DENIS LAXALDE, JOSEF PERKTOLD, ROBERT CIMRMAN, IAN HENRIKSEN, E.A. QUINTERO, CHARLES R HARRIS, ANNE M. ARCHIBALD, ANTÔNIO H. RIBEIRO, FABIAN PEDREGOSA, PAUL VAN MULBREGT, AND SCIPY 1.0 CONTRIBUTORS. (2019) SCIPY 1.0–FUNDAMENTAL ALGORITHMS FOR SCIENTIFIC COMPUTING IN PYTHON.

PEDREGOSA, F., VAROQUAUX, G., GRAMFORT, A., MICHEL, V., THIRION, B., GRISEL, O., ... & VANDERPLAS, J. (2011). SCIKIT-LEARN: MACHINE LEARNING IN PYTHON. JOURNAL OF MACHINE LEARNING RESEARCH, 12(OCT), 2825-2830.

QGIS DEVELOPMENT TEAM (2020). QGIS GEOGRAPHIC INFORMATION SYSTEM. OPEN SOURCE GEOSPATIAL FOUNDATION PROJECT. HTTP://QGIS.OSGEO.ORG

RAKSHIT, S., DEBNATH, S., & MONDAL, D. (2018). IDENTIFYING LAND PATTERNS FROM SATELLITE IMAGERY IN AMAZON RAINFOREST USING DEEP LEARNING. ARXIV PREPRINT ARXIV:1809.00340.

ŠIMIĆ DE TORRES, I. (2016). ANALYSIS OF SATELLITE IMAGES TO TRACK DEFORESTATION (BACHELOR'S THESIS, UNIVERSITAT POLITÈCNICA DE CATALUNYA).

SONNENBURG, S., BRAUN, M. L., ONG, C. S., BENGIO, S., BOTTOU, L., HOLMES, G., ... & RÄTSCH, G. (2007). THE NEED FOR OPEN SOURCE SOFTWARE IN MACHINE LEARNING. JOURNAL OF MACHINE LEARNING RESEARCH, 8(OCT), 2443-2466.

CITATIONS CONT.

STEFAN BEHNEL, ROBERT BRADSHAW, CRAIG CITRO, LISANDRO DALCIN, DAG SVERRE SELJEBOTN AND KURT SMITH. CYTHON: THE BEST OF BOTH WORLDS, COMPUTING IN SCIENCE AND ENGINEERING, 13, 31-39 (2011), DOI:10.1109/MCSE.2010.118

STÉFAN VAN DER WALT, S. CHRIS COLBERT AND GAËL VAROQUAUX. THE NUMPY ARRAY: A STRUCTURE FOR EFFICIENT NUMERICAL COMPUTATION, COMPUTING IN SCIENCE & ENGINEERING, 13, 22-30 (2011), DOI:10.1109/MCSE.2011.37

TEXTRON SYSTEMS (2010) FEATURE ANALYST RELEASE 5.0 PROVIDENCE, RI

TRIMBLE INC. (2019) ECOGNITION RELEASE 1.3 SUNNYDALE, CA

TUCKER, C. J. (1979). RED AND PHOTOGRAPHIC INFRARED LINEAR COMBINATIONS FOR MONITORING VEGETATION. REMOTE SENSING OF ENVIRONMENT, 8(2), 127-150.

WARMERDAM, F. (2008). THE GEOSPATIAL DATA ABSTRACTION LIBRARY. IN OPEN SOURCE APPROACHES IN SPATIAL DATA HANDLING (PP. 87-104). SPRINGER, BERLIN, HEIDELBERG.

XU, H., YANG, M., & LIANG, L. (2010, JUNE). AN IMPROVED RANDOM DECISION TREES ALGORITHM WITH APPLICATION TO LAND COVER CLASSIFICATION. IN 2010 18TH INTERNATIONAL CONFERENCE ON GEOINFORMATICS (PP. 1-4). IEEE.

ZHANG, J., HU, J., LIAN, J., FAN, Z., OUYANG, X., & YE, W. (2016). SEEING THE FOREST FROM DRONES: TESTING THE POTENTIAL OF LIGHTWEIGHT DRONES AS A TOOL FOR LONG-TERM FOREST MONITORING. BIOLOGICAL CONSERVATION, 198, 60-69