Team 41
Tree Segmentation from Multispectral Images

Jaelyn Litzinger, Adam Conrad, Kade Gonzalez

The Team

Jaelyn Litzinger

- Team Lead
- Visualizing the results of the classifier
- litzingj@oregonstate.edu

Adam Conrad

- Classifier architecture/design
- CGRB liaison
- conraada@oregonstate.edu

Kade Gonzalez

- Scrum Master
- Coordinate Wrangling
- gonzakad@oregonstate.edu

Project Partner

Bogdan Strimbu

- Assistant Professor at Oregon State University
- Department of Forest Engineering, Resources & Management
- https://directory.forestry.oregonstate.edu/people/strimbu-bogdan
- bogdan.strimbu@oregonstate.edu

Project Overview

- Currently, taking inventory of forests is an expensive and time consuming task that requires a trained specialist to manually inspect forests to count and identify the trees.
- With trees affecting our economy and sustainability, the matter of surveying is an important task that currently is daunting due to the mass size of forests and time required for manual inspection.

Project Overview

- Due to these current challenges with inventorying and identifying trees, this has sparked the need for alternative actions, such as remote imagery and analysis.
- Thus our team is currently working on developing an application that uses multispectral images to automate the task.
- Our implementation of tree segmentation aims to allow users to provide aerial multispectral images of saplings in Western Oregon.
- In return receive a list of coordinates for each tree in the image and the number of saplings in the image.

Project Overview

- To provide this outcome, we have focused on using a convolutional neural network. We train the model with hand labeled coordinates that are specified as particular classes, such as grass, road, dirt or saplings.
- An algorithm called “slic” then filters the image to obtain sapling candidates that can be classified by the model
- Then we run the model over the candidates to identify which are saplings and obtain their coordinates by their pixel location.
- Convert the image coordinates of the saplings into geographical coordinates using the Tif’s coordinate system.

Client Requests

- A model framework that can be trained to fit each new image
- A model with accuracy of 90%, with the commission and omission error below 10%
- The resulting coordinates in kml/kmz format so that they can be applied in Google Earth

System Overview

Build

- Python
  - Numpy
  - TensorFlow
  - Keras
  - MatPlotLib
  - Open CV
  - Tifffile
  - Scikit Learn, Scikit Image
  - Scipy
- GIS

The Classifier

- A convolutional neural network
- 3 inputs per pixel: RGB of the pixel
- Input is 30x30 block of pixels
- 3 hidden layers
- Outputs the class of the pixel block (ie sapling, grass, wood, etc)

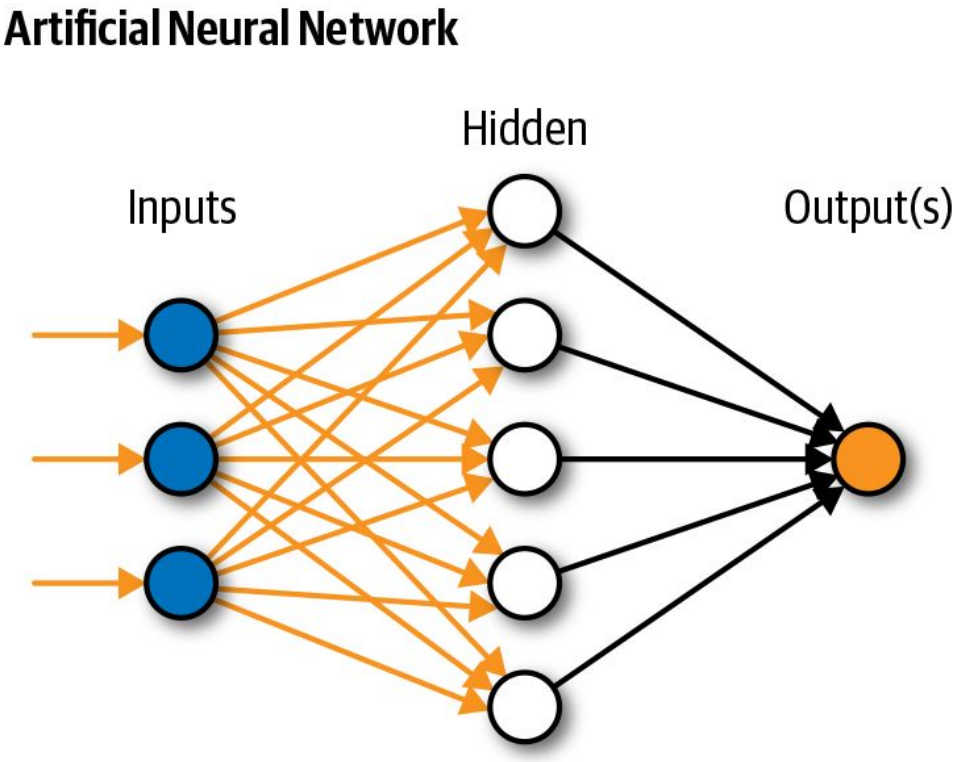
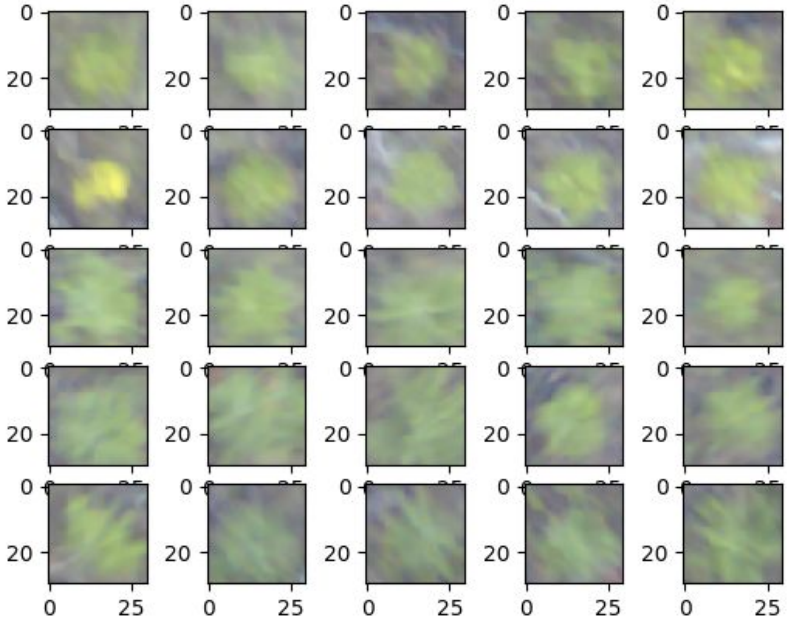


Image: https://www.innoarchitech.com/blog/artificial-intelligence-deep-learning-neural-networks-explained

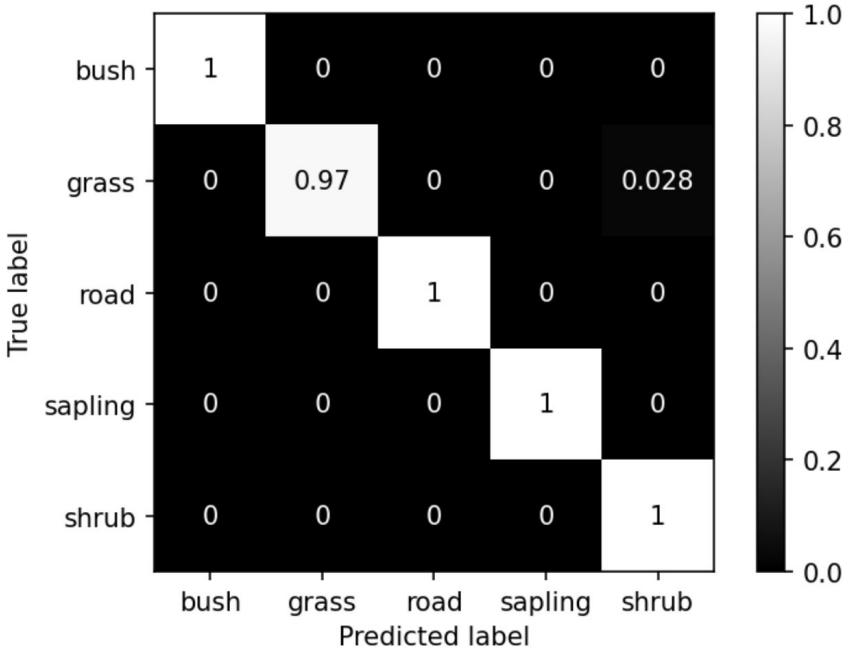
Training Data

- Labeled data of 90 coordinates per class
- Current model classes: sapling, road, grass, bush, and shrub
- Can add more classes like: wood, bark, leaves (on the ground), or any other distinctive objects in the image



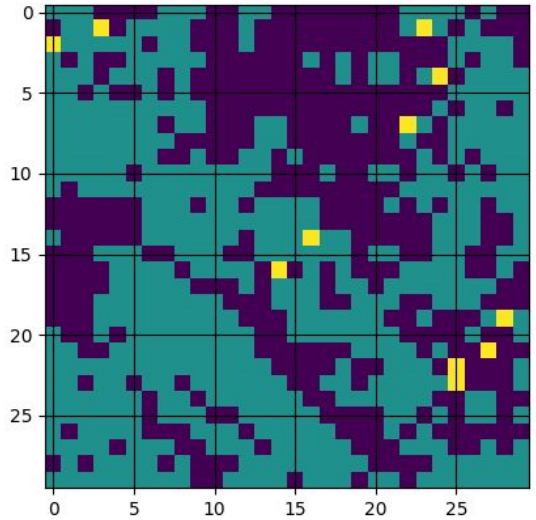
Evaluating Accuracy

- Model randomly splits labeled data into a 80/20 proportion for 80% training and 20% testing and validation
- Accuracy is evaluated based on the classification of the testing coordinates
    - False positives vs false negatives



Visualizing the Results

- A tool to view a graphical display of the results of the classifier on an unlabeled area of the image
- Graphs the classification of each 30x30 block in the region
- Blue: grass, purple: road, yellow: sapling



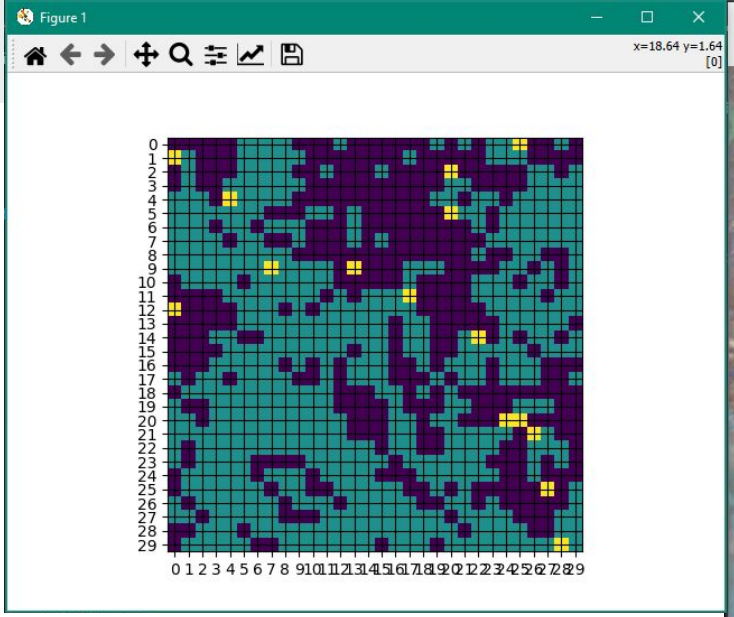
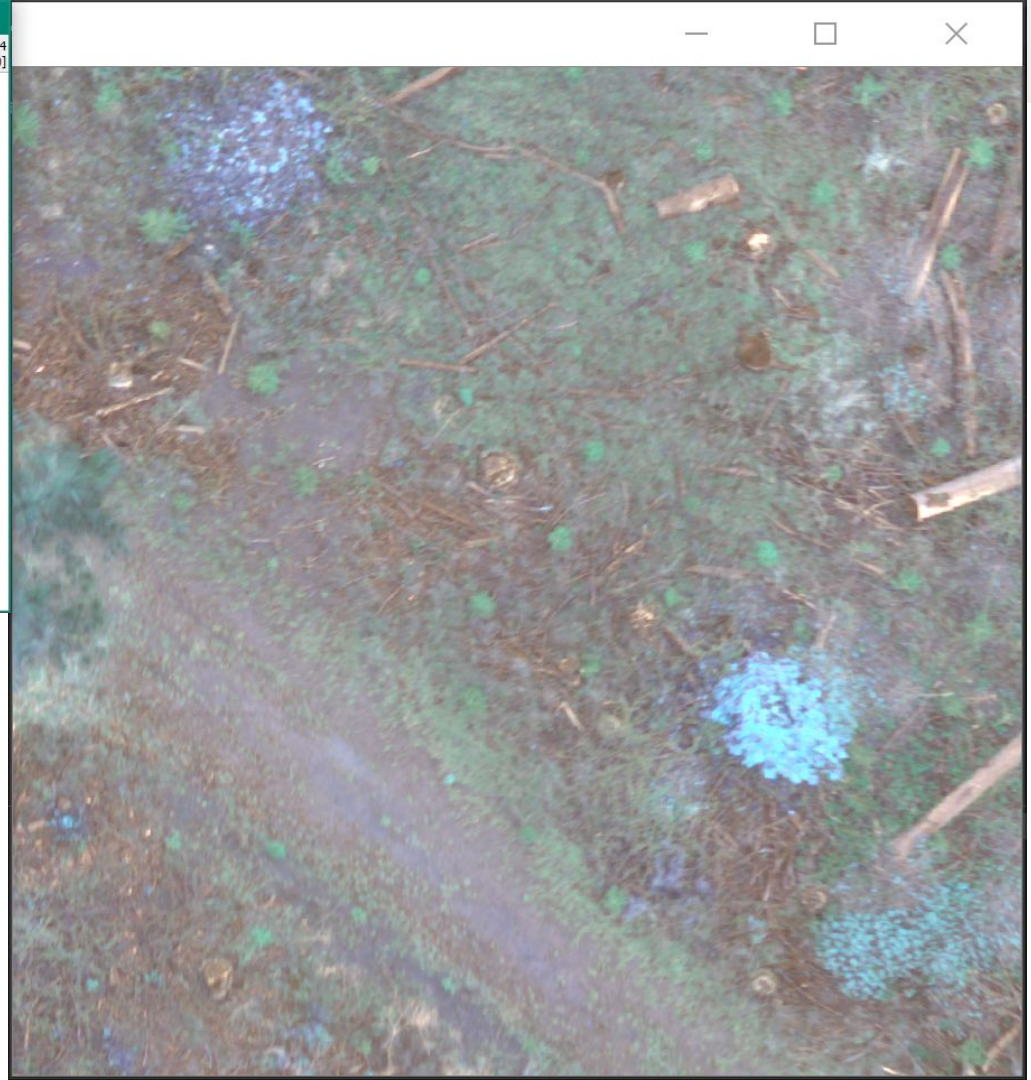


Figure 1
x=18.64 y=1.64
[0]
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29

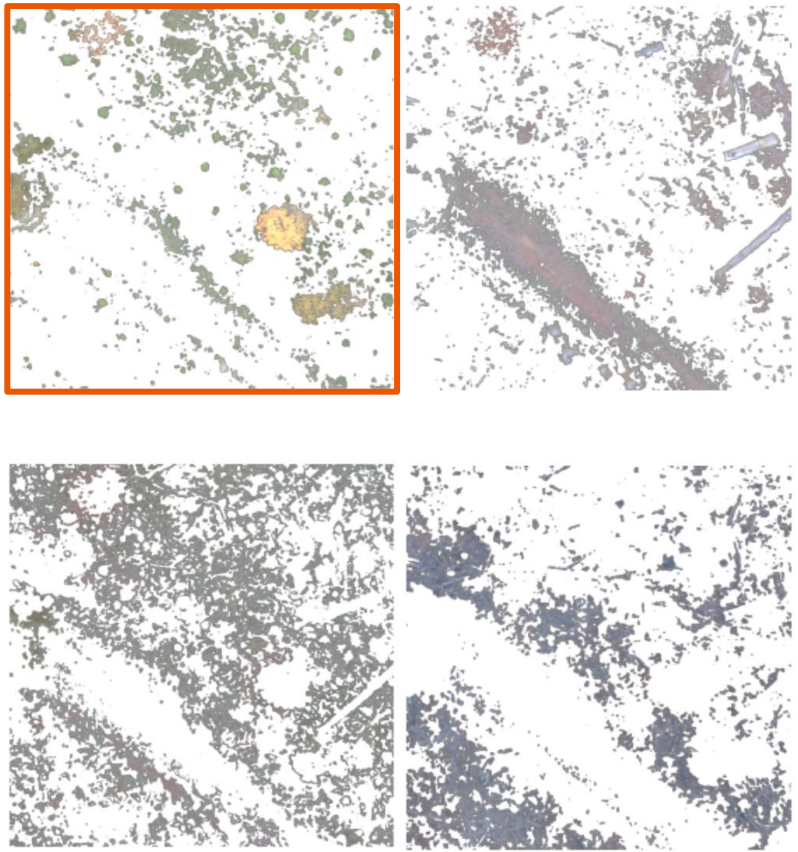


Finding More True Positives

- One reason the classifier on its own may be missing saplings is if the 30x30 pixel block only contains a portion of the sapling
  - Since the model was trained with the sapling centered in the block
- Thus, we need a way to feed the classifier only well centered blocks

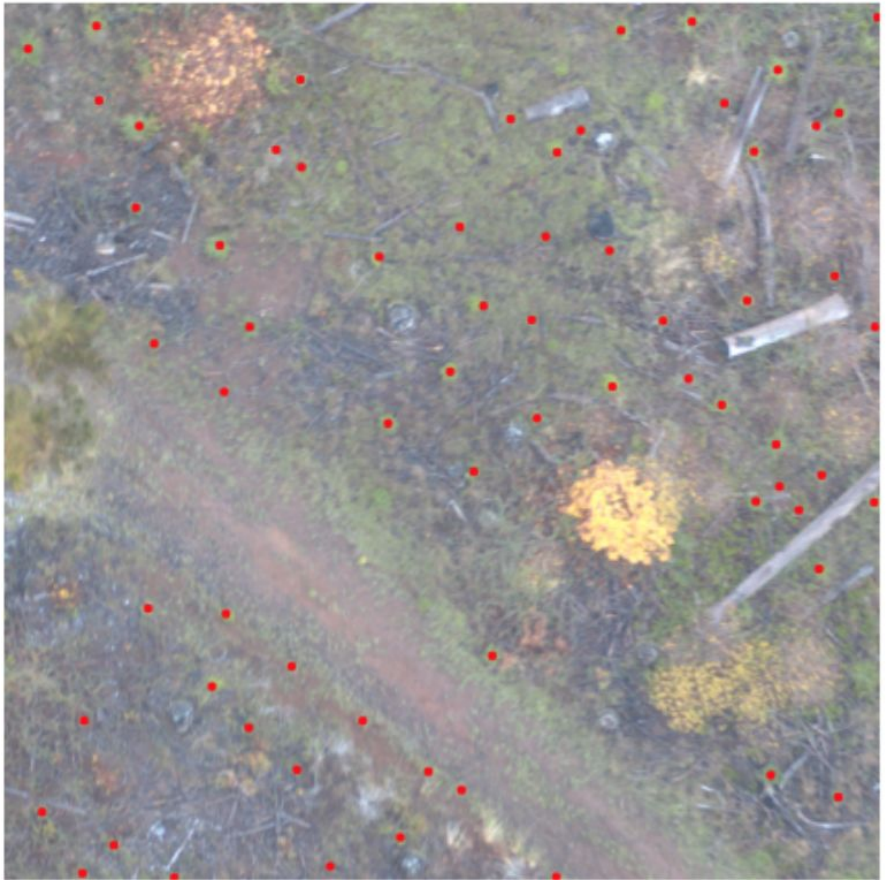
Slic

- Slic filters the image to identify regions of interest to classify over.
- We then feed those regions of interest to the classifier instead of the whole image
- This eliminates a lot of false positives by only giving the classifier potential saplings that are well centered



Sapling Coordinates

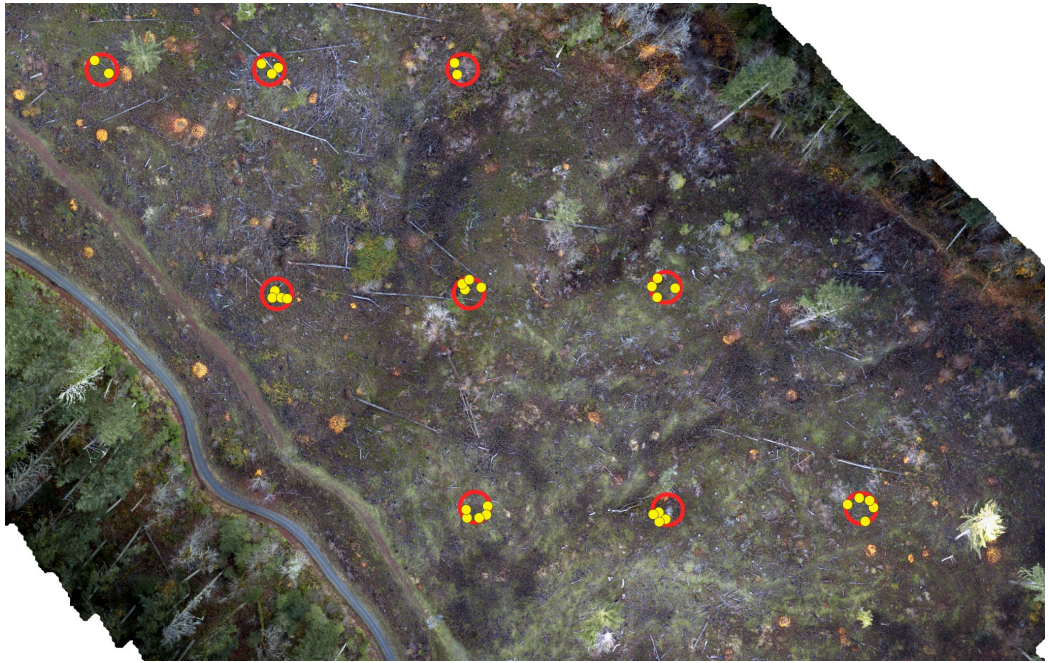
- The saplings identified by the classifier after slic are marked in red on the image



Validation

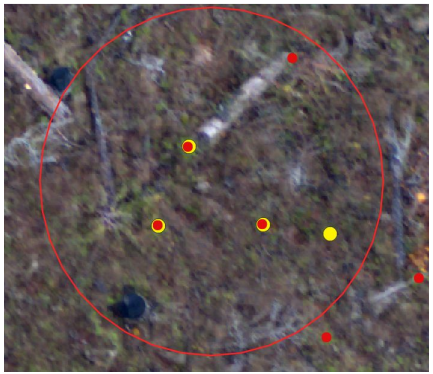
Due to the very large size of our input images calculating precision and recall statistics for every sapling is unrealistic.

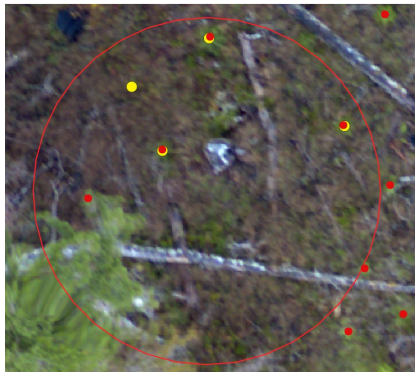
Instead, we are calculating precision and recall statistics on the validation circles shown on the right.

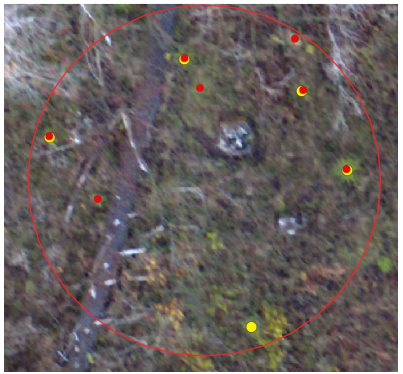


Validation Examples

	True Positives	False Positives	False Negatives	Recall	Precision
Plot 1	3	1	1	.75	.75
Plot 2	3	1	1	.75	.75
Plot 3	4	3	0	1	.571







User Stories

- ✅ **As an end user, I want to submit an image and receive a list of coordinates of all the trees in the image so that I know how many trees are in the region and where they are.**
  - Our program finds the locations of saplings in an input image.
- ✅ **As an end user, I want the resulting coordinates to be converted into a kml/kmz format so that I can apply them into Google Earth or similar programs.**
  - All sapling locations are converted to kmz format using the input image coordinate system.
- ✅ **As the project partner, I want the coordinate for a tree to be within the canopy of the tree so that the coordinates are accurate.**
  - The center of every identified region is found and are highly accurate.
- ⬛ **As the project partner, I want the accuracy of the model to be at least 90%, with the commission and omission error below 10% so that the model can be trusted.**
  - We were not able to achieve this level of accuracy. Our program finds saplings with ~10% omission error and ~26% commission error.

User Stories (Continued)

- ✅ **As the development team, we need a python library so that we can train a machine learning model with the multispectral images to produce an output file of coordinates.**
  - We used keras and tensorflow for building and training the model.
- ✅ **As the development team, we need to convert our x-y coordinates into a kml/kmz or geographical format so that it can be interpreted through a map application such as Google Earth.**
  - We used GIS packages that took the coordinates of the input image to place our found saplings.
- ✅ **As the development team, we need to create a training and testing dataset of images so that the model can be trained and then tested for accuracy.**
  - A program was developed to streamline the process of collecting training data.
- ✅ **As the development team, we need a way to assess the accuracy of our classifier so that we can judge the success of the algorithm and speak to its confidence.**
  - We tested the classification results in nine different sub-areas of the image due to its large size.

Current State

- The project is at a current state that our project partner is happy and it meets a majority of the requirements.
- Currently, we have a trained model that is fed a filtered image where it can classify the object as either a sapling or not.
- The project is in a state that it can be easily passed off, however, some additional documentation would be beneficial for another group to easily work on the project.
- In addition, the following group could work to refine the codebase as well as overall accuracy, and create a UI that would allow for easier usage.