Analysis methods

Beginning in March of 2024, the Spatial Analysis Lab began developing an iterative sampling protocol which integrates data collection and processing into predictive mapping outputs. This process would allow for more consistent predictive mapping at a faster pace than previous field seasons. Early modeling efforts used a manually harmonized Landsat/Sentinel-2 product at 30m resolution, which averaged images every 5-10 days. Machine learning models were implemented to process the surface reflectance, with Random Forest and a gradient boosted ensemble model. These methods were completed in Google Earth Engine, using built in modeling functions with outputs defined prior to processing.

Updated modeling methods now utilize cloud processing and NASA’s Application Programming Interface (API), **Land Processes Distributed Active Archive Center (LPDAAC)** and **SpatioTemporal Asset Catalogs (STAC)** to access **NASA’s Harmonized Landsat Sentinel (HLS)** product. HLS is a seamless surface reflectance product with observations once every 3 days at 30m resolution. Using these methods, greater amounts of data can be processed in parallel, with data input, model building or updating and output happening simultaneously compared to the original workflow. This workflow uses Google Colab, a hosted Jupyter Notebook service with access to Google’s cloud computing resources. Using Colab has a distinct advantage over Google Earth Engine, which is seamless integration with STAC allowing access to a greater diversity of NASA and other cloud-based imagery or datasets.

Image Processing

All imagery is gathered using STAC, OpenDataCube (citation needed) and Xarray (Hoyer, S., Hamman, J. (In revision). Xarray: N-D labeled arrays and datasets in Python), for each datapoint collected during the 2023-2024 field seasons. The data contains over 2,000 individual observations of presence and absence of Ventenata.

Using the STAC, OpenDataCube and Xarray query processes, polygons were grouped to reduce the number of API requests and streamline dataset compilation**. Polygons were grouped into K-Nearest Neighbor (KNN), then grouped into 34 units to make API requests**. The timeframe is from **April 2023 to October 2023**, with a metadata applied filter for images with less than 30% cloud cover. We then use HLS’s built-in quality band, **Fmask, to identify pixels with cloud shadows, adjacent to clouds and clouds using a simple bitmask classification system.** Identified pixels are back or forward-filled with the nearest cloud-free pixel up to 2 images, and the whole image is resampled to a weekly median. On average, this method resulted in just over 80 images per polygon (with some polygons overlapped in the same extent) for the growing season.

After the imagery has been processed and filtered, the built in bands are used to build the following vegetation indices, identified in the literature as being strong predictors in invasive species detection (Citations), phenological differences (Citations) and ecological bases (Citations).

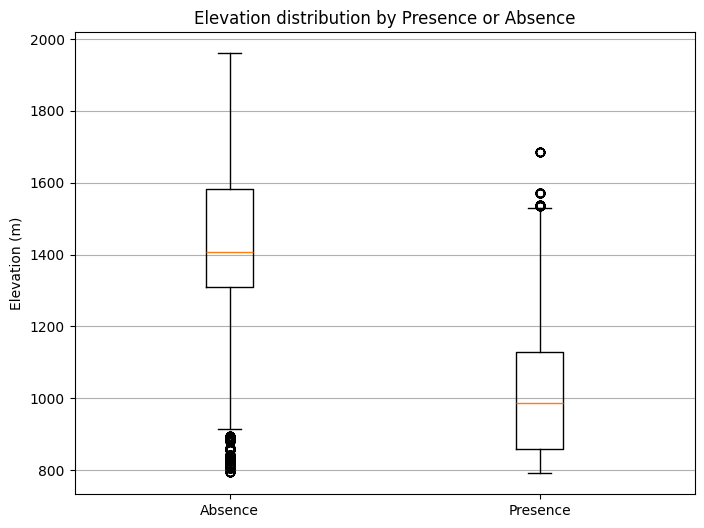
Table 1 shows the bands used to calculate the vegetation indices.

|  |  |  |
| --- | --- | --- |
| Band | Vegetation Indices | |
| Red | NDVI | Normalized Difference Vegetation Index |
| Green | EVI | Enhanced Vegetation Index |
| Blue | AVI | Advanced Vegetation Index |
| NIR | ARVI | Atmospherically Resistant Vegetation Index |
| SWIR | GNDVI | Green Normalized Difference Vegetation Index |
|  | SAVI | Soil Adjusted Vegetation Index |
|  | MSAVI | Modified Soil Adjusted Vegetation Index |
|  | RECL | Red-Edge Chlorophyll Index |
|  | NDWI | Normalized Difference Water Index |
|  | OSAVI | Optimized Soil Adjusted Vegetation Index |
|  | SIPI | Structure-Intensive Pigment Index |
|  | GCI | Green Chlorophyll Index |

Data description:

For all models, we split our 2023-2024 datasets into 70:30 train/test data randomly. The predictive class is binomial, either presence of or absence of Ventenata. The data was equally divided at 51% presence, 49% absence. Additionally, a secondary dataset not used for model training will be used to evaluate accuracy both in terms of model performance and using satellite imagery to test identification.

Using the 2,000 observations in the original dataset the spectral signatures were pulled out on a per-pixel basis plus the April-October timeframe yielding 90,000 observations.



A graph with lines and a line in the middle

Description automatically generated with medium confidence

A graph of a graph showing a slope by presence and presence

Description automatically generated

Model Building

The scikit-learn package is a machine learning toolkit within Python and is considered one of the most valuable open-source projects to date. Within the package there are multiple models that can be used for classification. This gives us a good opportunity to test multiple supervised and semi-supervised machine learning models to determine what model may be the most suitable for our purpose, data and long-term outputs.

Histogram Gradient Boosting Classifier (HBGC):

HGBC is a methodology that is based off Gradient Boosting, which is an ensemble decision tree algorithm that gradually improves predictions. HGBC uses a binning technique to accelerate tree training, which transforms the input variables to unique values. Typically, gradient boosting is a slow process because trees are added sequentially, as opposed to Random Forest which trains trees in parallel. The benefit of binned training is quick processing and adjustable hyperparameters. Because the dataset has multiple dimensions, we used three different cross validation techniques, and a grid search to tune hyperparameters to the optimal configuration using Mean Squared Error (MSE), Accuracy and Precision as evaluation metrics. The final selected model (see below) was based off the highest accuracy by cross validation type, lowest MSE, and highest precision.

A graph of a graph showing the amount of data

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

Initially models included topographical variables (elevation, slope, and aspect) as covariates, however multiple model iterations found topographical variables to have high permutation importance, thus impacting the identification of Ventenata using satellite imagery. Rather, topographical variables will be included in later modeling iterations using climate variables. The figure below shows permutation importance of model covariates which is calculated on training A graph of a number of objects

Description automatically generateddata to show how much the model relies on each feature during training.

Figure 1: HBGC permutation feature importance.

The feature ‘day of year’ (DOY) was found to be effective in identifying Ventenata presence on the ground, which is consistent with other modeling efforts. Figure 1 shows the permutation feature importance, which measures the error associated with a model after the feature’s values are permuted by shuffling the values for each observation being tested. For DOY, the error increased by 21% after 30 permutations.

A graph with blue lines

Description automatically generatedThe partial dependance plot displays how the DOY feature specifically impacts identification of Ventenata, In this case, positive values indicate class 1 (presence), negative values indicate class 0 (absence) with the magnitude reflecting the confidence in the model. HGBC outputs raw scores before converting to classified samples using a threshold value set in the hyperparameters. The partial dependance output uses the raw scores to calculate the partial dependence.

Figure 2: HBGC partial dependence plot of day of the year feature.

In this case, absence predictions are more confident in June-August and presence predictions are more confident at the beginning and end of the growing season. This is consistent with the phenological changes we see over time in the field.

Random Forest (RF)

Random forest is one of the most common machine learning models due to it’s ease of use and flexibility both in classification and regression. Similar to Gradient Boosting, random forest models are made up of multiple decision trees with the key difference being the feature bagging, or random feature subsetting that builds forests with low-correlation trees. These algorithms are prone to bias and overfitting, however using an ensemble method they can predict more accurately. The same process of hyperparameter tuning was used with the RF algorithm, tuning the maximum depth of the tree, or the number of splits a tree makes during the decision process, the minimum of samples to split, and the number of trees in the forest. The same cross-validation approach was used to evaluate out of box performance and accuracy to ensure the model is being evaluated on common use cases, i.e. predicting new areas.

Like the previously discussed models, the RF model performed well on the repeated stratified k-fold cross validation, the standard cross validation technique. Also similarly was the accuracy of the other methods of cross validation including the leave one group out (LOGO), and the Group K-Fold. These two techniques were used to evaluate model performance specifically using geography as an input for cross validation. LOGO uses all groups except one to train on, and then is tested on the one “left out” group. This is an effective tool for examining how well the data can be applied to spatial distinct or distant areas. While not technically accounting for spatial autocorrelation, the CV method can be used as a proxy to examine how distant values can be applied to nearby values. Group KFold is similar, only this time, instead of leaving one group out, each group acts as its own cross validation. This means one group trains a forest, followed by another group training a different forest and so on until all group both build one forest and test on all forests. This plays into the majority vote found in decision trees, where the majority of votes for the forest are used to classify presence or absence.

A graph of a bar chart

Description automatically generated with medium confidence

A screenshot of a computer program

Description automatically generatedA graph of a number of objects

Description automatically generatedAn interesting outcome of the analysis was the consistent importance of the DOY variable despite all other variables shifting in their importance.

RF modeling is a flexible classification method, but often overfits due to the algorithm’s data handling.

Adaptive Boosting (ADA)

Adaptive boosting uses a methodology well suited to ensemble modeling. Multiple weak learners (models) are adjusted to build a single strong output. ADA is thought of as less susceptible to overfitting because of the multiple weak learning method, however the output relies on the best choice of weak estimators. The following estimators were evaluated using a classifier comparison and adjusted hyperparameters using a grid search to examine the accuracy, MSE, and precision of each cross-validation method. Estimators that performed best with both the Group K-Fold and LOGO methods were selected to account for spatial differences in the data. The base model performed as shown below, but with selected weak estimators MSE and Accuracy increased by \_\_\_\_\_\_\_\_\_\_\_\_.

A graph of a graph

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

A diagram of a diagram of a robot

Description automatically generated with medium confidencePredictive Outputs

The models’ predictions were combined in different ways to evaluate individual model performance on untested data as well as combined model performance using sequential boosting and bagging.

Sequential Boosting

Sequential boosting is the process of using predictive output as an input for the next model. This creates a process by which all models’ strengths and predictive functionality are incorporated into the output. We used the following as inputs for each sequential boost, a high certainty prediction, and cross-validation based weighted residual.

In order to prevent subsequent models’ overfitting, multiple variables were evaluated based on the