**Land Cover Classification of Satellite Imagery**

Land cover classification of Krishna godavari mangroves satellite imagery using K-Nearest Neighbor(K-NNC), Support Vector Machine (SVM), and Gradient Boosting classification algorithms with Python.



Photo by [Paulo Simões Mendes](https://unsplash.com/@paulozono?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText) on [Unsplash](https://unsplash.com/?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText" \t "_blank)

*This article helps readers to better understand land cover classification on Krishna godavari mangroves satellite data using different classification algorithms with Python.*

**Table of Contents**

1. **Krishna godavari mangroves Satellite Imagery**
2. **K-Nearest Neighbor Classifier (K-NNC)**
3. **Support Vector Machine (SVM)**
4. **Gradient Boosting Classifier**
5. **Conclusion**
6. **References**

*Let’s Get Started…*

**Read Data**

let’s read the 12 bands using rasterio and stack them into an n-dimensional array using numpy.stack() method. The resultant data after stacking has the shape (12, 954, 298). The ground truth of the satellite image is read using the *loadmat*method from the *scipy.io*package. The ground truth has 6 classes which include water, plants, trees, bare land, e.t.c.

**Data Visualization**

These Krishna godavari mangroves data have multiple numbers of bands that contain the data ranging from visible to infrared. So it is hard to visualize the data for humans. By creating an RGB Composite Image **(displaying the image as a combination of these 3 colors)** makes it easier to understand the data effectively. To plot RGB composite images, you will plot the red, green, and blue bands, which are bands 4, 3, and 2, respectively. Since Python uses a zero-based index system, so you need to subtract a value of 1 from each index. Therefore, the index for the red band is 3, green is 2, and blue is 1.

The Composite images that we created can sometimes be dark if the pixel brightness values are skewed toward the value of zero. This type of problem can be solved by stretching the pixel brightness values in an image using the argument stretch=True to extend the values to the full 0-255 range of potential values to increase the visual contrast of the image. Also, the str\_clip argument allows you to specify how much of the tails of the data that you want to clip off. The larger the number, the more the data will be stretched or brightened.

Let’s see the code to plot the RGB composite image along with the stretch applied.

Let’s visualize the ground truth using the *plot\_bands*method from *eathpy.plot*package*.*

The below figure shows the composite image and the ground truth of the Krishna godavari mangroves satellite data.

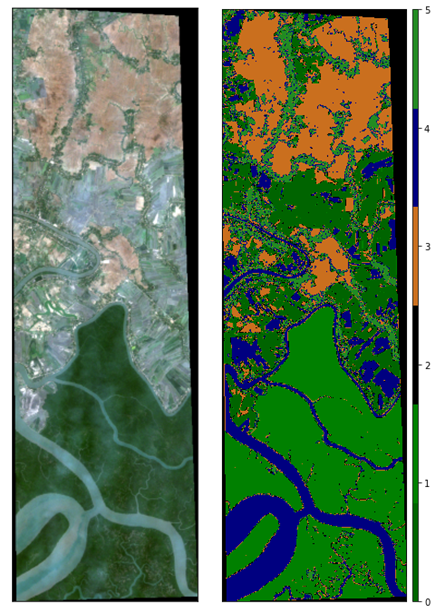
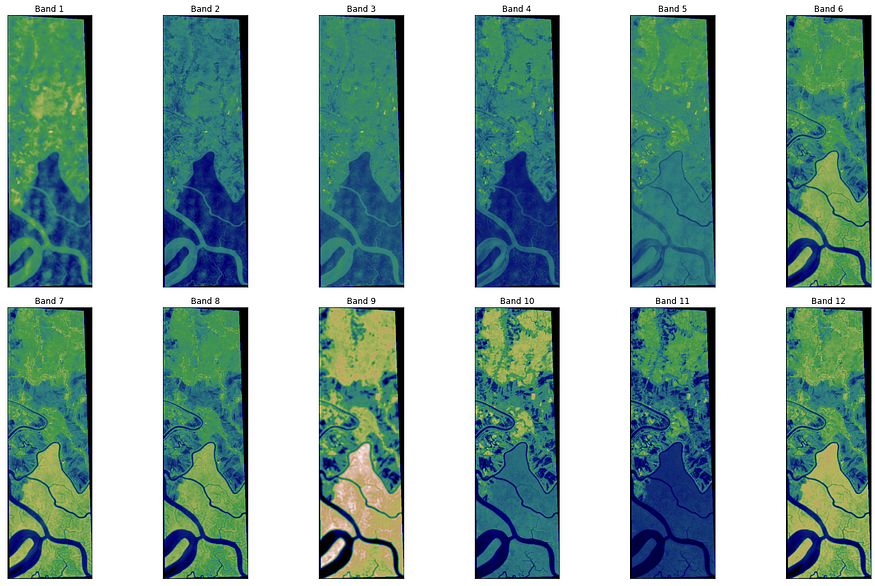


Image by Author

As we discussed, the data contains 12 bands. let’s visualize each band using the EarhPy package. the plot\_bands() the method takes the stack of the bands and plots along with custom titles which can be done by passing unique titles for each image as a list of titles using the title= parameter.



Visualization of Bands — Image by Author

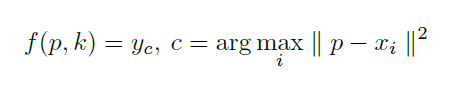
**Preprocessing**

Standardization is another **scaling** technique where the values are centered around the mean with a unit **standard** deviation. This means that the mean of the attribute becomes zero and the resultant distribution has a unit **standard** deviation. The scaled data is divided into train and test data in the ratio of 30:70. The below code is used to scale and split the data.

**K-Nearest Neighbor Classifier (K-NNC)**

k-Nearest neighbor classifier is one of the widely used classifiers in machine learning. The main objective of this method is that the data instances of the same class should be closer in the feature space.

Let us consider a dataset with n data points represented as f(x1, y1), (x2, y2)... (xi, yi)...(xn, yn). Where xi and yi are feature vector and corresponding class label respectively. For a new data point p, the class label can be predicted by k-NNC with a k value where k is the number of neighboring data points as follows:

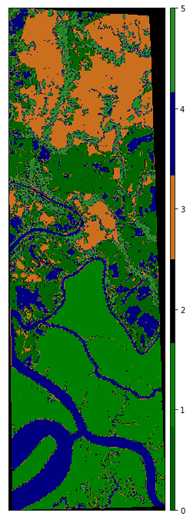


K-KNC, Image by author

We are going to implement k-NNC using the scikit learn package. The below code K-NNC instance with n\_neighbors as 6 and fits the train data, predicts the labels of the test data, shows the accuracy, and prints the classification report which includes precision, recall and F1-score of each class. The K-NNC has shown **98.94%** accuracy over the test data.

K-NNC Classification Report — Image by Author

Let’s visualize the classification map of K-NNC, the below code is used to predict the labels of the Krishna godavari mangroves data and plot the data using plot\_bands() method from the earthpy package.



Classification Map using K-NNC — Image by Author

**Support Vector Machine (SVM)**

The support vector machine (SVM) is a supervised learning method that generates input-output mapping functions from a set of labeled training data. The mapping function can be either a classification function, i.e., the category of the input data, or a regression function.

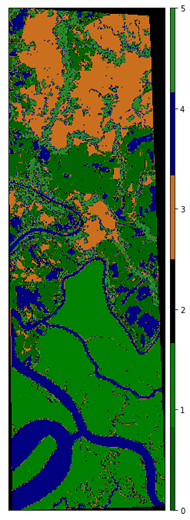
For classification, nonlinear kernel functions such as Radial Basis Function(RBF), Polynomial, Sigmoid, e.t.c are often used to transform input data to a high-dimensional feature space in which the input data become more separable compared to the original input space. Maximum-margin hyperplanes are then created. The model thus produced depends on only a subset of the training data near the class boundaries.

The below code is used to create an instance of SVM with the regularization parameter C as 3 and RBF kernel. Fits the data, predict the labels for test data, and prints the accuracy and classification report.

The Support Vector Machine (SVM) algorithm has shown **99.88**% accuracy on the test data. The classification report is shown below:

Image by Author

Let’s visualize the classification map of SVM, the below code is used to predict the labels of the Krishna godavari mangroves data and plot the data using plot\_bands() method from earthpy package.



Classification Map of Krishna godavari mangroves using SVM — Image by Author

**Gradient Boosting Classifier**

Gradient boosting is a technique attracting attention for its prediction speed and accuracy, especially with large and complex data. Gradient boosting is a type of machine learning boosting. It relies on the intuition that the best possible next model, when combined with previous models, minimizes the overall prediction error. The key idea is to set the target outcomes for this next model to minimize the error.

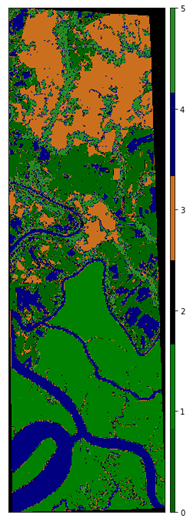
Today we are going to use **lightGBM**which is a gradient boosting framework that makes use of tree-based learning algorithms. LightGBM is called “**Light**” because of its computation power and giving results faster. It takes **less memory to run** and can **deal with large amounts of data**. It is one of the most widely used algorithms in competition because the motive of the algorithm is to get good accuracy of results.

The below code is used to create an instance of the lightgbm with parameters such as learning rate, maximum depth, number of classes, e.t.c

The lightbgm classifier has shown **98.55%** accuracy over the test data and the classification report is shown below.

Image by Author

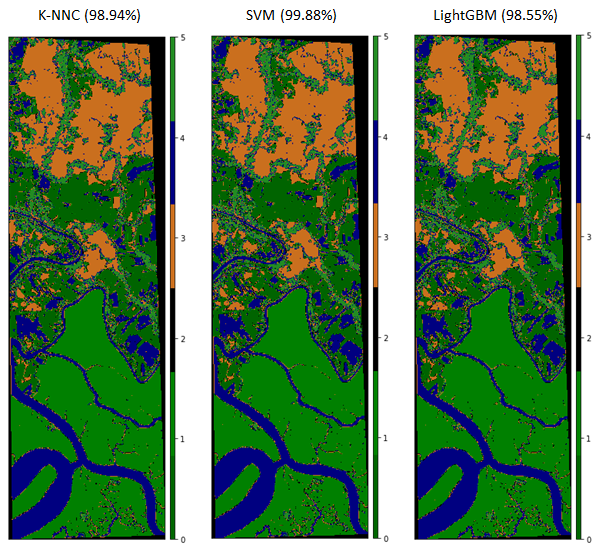
Let’s visualize the classification map generated using lightgbm classifier. The below code is used to predict the labels of the Krishna godavari mangroves data and plot the data using plot\_bands() method from the earthpy package.



Classification Map of Krishna godavari mangroves using lightgbm classifier — Image by Author

**Conclusion**

The article shows how to implement K-NNC, SVM, and LightGBM classifiers for land cover classification of Krishna godavari mangroves satellite data using Python. The Support Vector Machine has shown better performance compared to K-Nearest Neighbor Classifier (K-NNC) and LightGBM classifier. The below figure shows the classification maps of the above-mentioned three classifiers.



Comparison of the classifiers based on Accuracy — Image by Author

The code used in this article can be accessed from the below GitHub repository.

**Vegetation and Soil Indices**

Normalized Satellite indices are images that are computed from Multi-Spectral satellite images. These images emphasize a specific phenomenon that is present while mitigating other factors that degrade the effects in the image. For instance, a vegetation index will show healthy vegetation as bright in the index image, while unhealthy vegetation has lower values and barren terrain is dark. Since shading from terrain variation (hills and valleys) affect the intensity of images, the indices are created in ways that the color of an object is emphasized rather than the intensity or brightness of the object.

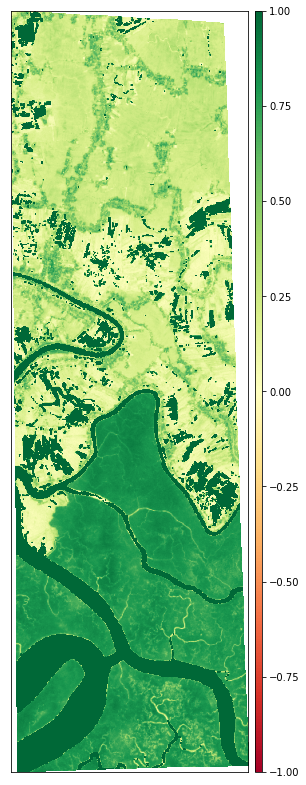
**Normalized Difference Vegetation Index (NDVI)**

To determine the density of green on a patch of land, researchers must observe the distinct colors (wavelengths) of visible(VIS) and near-infrared (NIR)sunlight reflected by the plants. The Normalized Difference Vegetation Index (NDVI) quantifies vegetation by measuring the difference between near-infrared which vegetation strongly reflects and red light (which vegetation absorbs). NDVI always ranges from -1 to +1.

NDVI = ((NIR - Red)/(NIR + Red))

For example, when you have negative values, it’s likely water. On the other hand, if you have an NDVI value close to +1, there’s a high possibility that it’s dense green leaves. But when NDVI is close to zero, there aren’t green leaves and it could even be an urbanized area.

Let’s see the code to implement NDVI on the Krishna godavari mangroves satellite data.



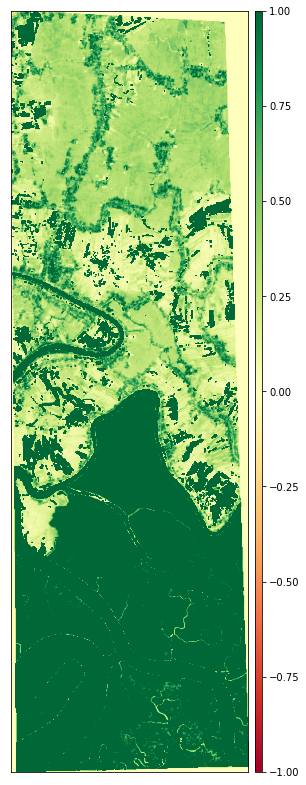
NDVI of Krishna godavari mangroves Data — Image by Author

**Soil Adjusted Vegetation Index (SAVI)**

The Soil-Adjusted Vegetation Index (SAVI) is a vegetation index that attempts to minimize soil brightness influences using a soil-brightness correction factor. This is often used in arid regions where vegetative cover is low.

SAVI = ((NIR - Red) / (NIR + Red + L)) x (1 + L)

The L value varies depending on the amount of green vegetative cover. Generally, in areas with no green vegetation cover, L=1; in areas of moderate green vegetative cover, L=0.5; and in areas with very high vegetation cover, L=0 (which is equivalent to the NDVI method). This index outputs values between -1.0 and 1.0. Let’s see the code for the implementation of SAVI.

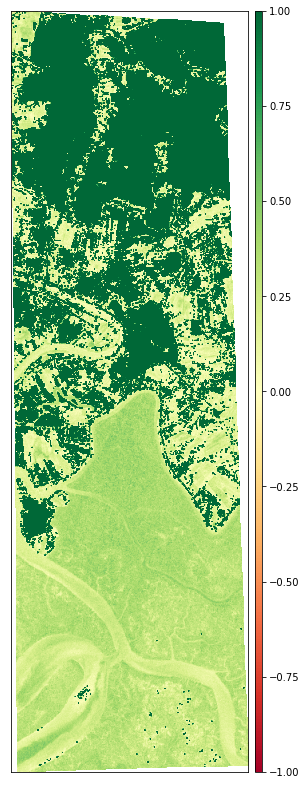


SAVI of Krishna godavari mangroves Data — Image by Author

**Visible Atmospherically Resistant Index (VARI)**

The Visible Atmospherically Resistant Index (VARI) is designed to emphasize vegetation in the visible portion of the spectrum while mitigating illumination differences and atmospheric effects. It is ideal for RGB or color images; it utilizes all three color bands.

VARI = (Green - Red)/ (Green + Red - Blue)



VARI of Krishna godavari mangroves Data — Image by Author

**Water Indices**

Surface water change is a very important indicator of environmental, climatic, and anthropogenic activities. Remote sensors, such as sentinel-2, Landsat, have been providing data for the last four decades, which are useful for extracting land cover types such as forest and water. Researchers have proposed many surface water extraction techniques, among which index-based methods are popular owing to their simplicity and cost-effectiveness.

**Modified Normalized Difference Water Index (MNDWI)**

The Modified Normalized Difference Water Index (MNDWI) uses green and SWIR bands for the enhancement of open water features. It also diminishes built-up area features that are often correlated with open water in other indices.

MNDWI = (Green - SWIR) / (Green + SWIR)

The below code serves the purpose of implementing MNDWI and the output is shown below.



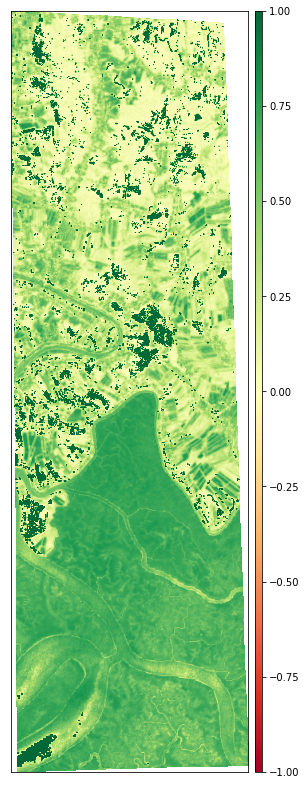
MNDWI of Krishna godavari mangroves Data — Image by Author

**Normalized Difference Moisture Index (NDMI)**

The Normalized Difference Moisture Index (NDMI) is sensitive to the moisture levels in vegetation. It is used to monitor droughts as well as monitor fuel levels in fire-prone areas. It uses NIR and SWIR bands to create a ratio designed to mitigate illumination and atmospheric effects.

NDMI = (NIR - SWIR1)/(NIR + SWIR1)

Let’s see the implementation and the output:



NDMI of Krishna godavari mangroves Data — Image by Author

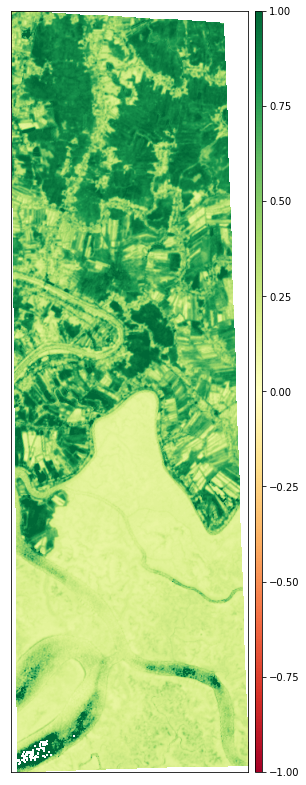
**Geology Indices**

Satellite imagery and aerial photography have proven to be important tools in support of mineral exploration projects. They can be used in a variety of ways. Firstly they provide geologists and field crews the location of tracks, roads, fences, and inhabited areas.

**Clay Minerals**

The clay ratio is a ratio of the SWIR1 and SWIR2 bands. This ratio leverages the fact that hydrous minerals such as clays, alunite absorb radiation in the 2.0–2.3 micron portion of the spectrum. This index mitigates illumination changes due to terrain since it is a ratio.

Clay Minerals Ratio = SWIR1 / SWIR2

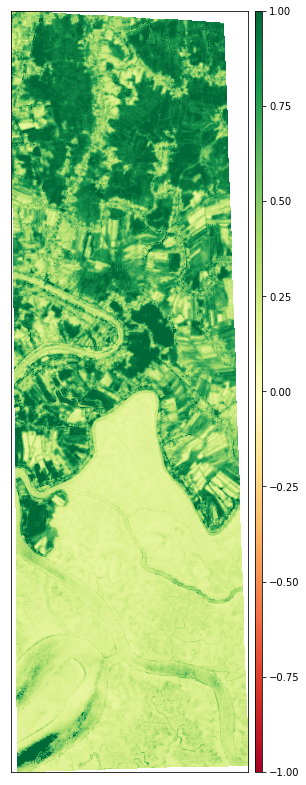


CMR of Krishna godavari mangroves Data — Image by Author

**Ferrous Minerals**

The ferrous minerals ratio highlights iron-bearing materials. It uses the ratio between the SWIR band and the NIR band.

Ferrous Minerals Ratio = SWIR / NIR



FMR of Krishna godavari mangroves Data — Image by Author

**Conclusion**

This article introduces different methods such as data visualization and normalized vegetation, water, and geogloy indices to analyze Krishna godavari mangroves *satellite data*using python.