Challenge 8

Student: Xabier Oyanguren Asua
NIU: 1456628 Date: 07/05/2022

1 Objectives

Using Python and mainly the sklearn library, we train several models to predict which pixels belong to tree regions and which to non-tree regions in a satellite image.

2 Data Acquisition

The image for the exercise was obtained from an online web map service with the request as:
https://www.ign.es/wms-inspire/pnoa-ma?REQUEST=GetMap&VERSION=1.1.0&SERVICE=WMS&SRS=EPSG:32631&BBOX=-706.70,4663987.54,-306.70,4664387.54&WIDTH=1024&HEIGHT=1024&LAYERS=OI.OrthoimageCoverage&STYLES=&FORMAT=JPEG&BGCOLOR=0xFFFFFF&TRANSPARENT=TRUE&EXCEPTION= INIMAGE%27.

This image can be seen in Figure 1.

Figure 1: Employed image for the exercise.

3 Procedure

First of all we select manually 100 points belonging to the "tree" class (label 1) and 100 points belonging to the "non-tree" class (label 0). The selected points can be seen in Figure 2. We then select an obvious feature for each of the points to be its RGB color triplet. We shuffle them (with their proper labels) and generate a train set with 100 samples and a test set with the rest of 100 samples.

Then, we instantiate a random forest classifier (with 100 trees), a logistic regressor and a Gaussian naive Bayes classifier. We train each of the models with the train data, and then test their accuracies over the test set.

4 Results

We get for the random forest a 96% accuracy, for the logistic a 97% accuracy and for the naive Bayes one a 94% accuracy.

Figure 2: Selected pixels with tree and non-tree label.

Binary predictions for the whole image by each model can be found in Figures 3,4,5. This classification result in color can be found in Figure 6,7,8.

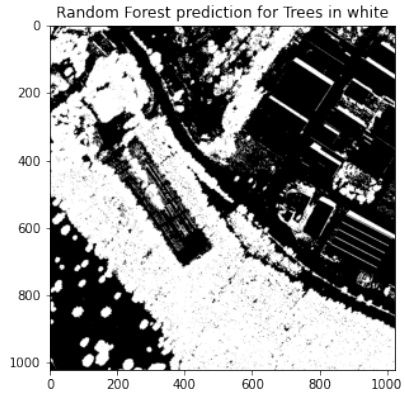

Figure 3: Binary prediction of whole image by Random Forest.

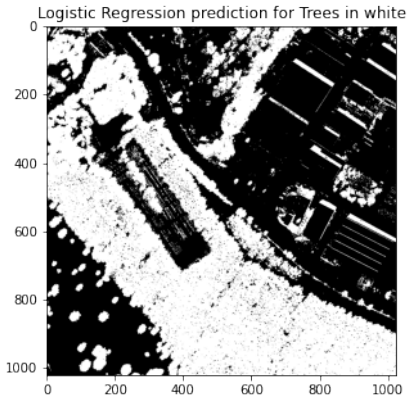

Figure 4: Binary prediction of whole image by Logistic Regression.

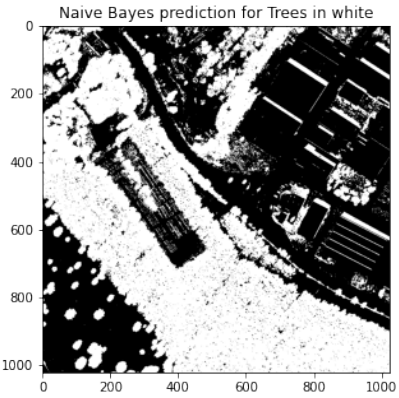

Figure 5: Binary prediction of whole image by Naive Bayes.

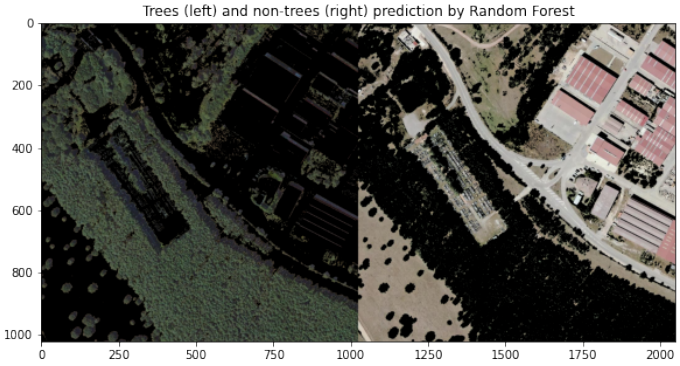

Figure 6: Colored prediction of whole image by Random Forest.

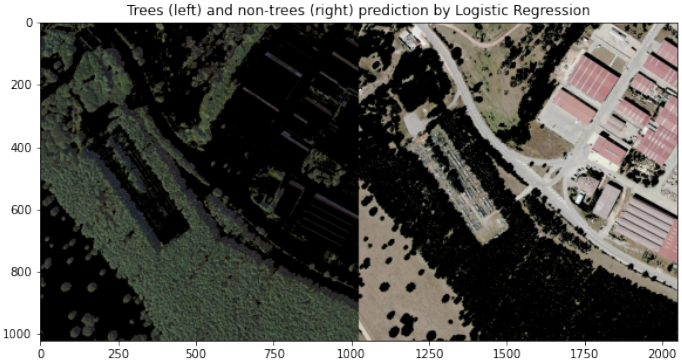

Figure 7: Colored prediction of whole image by Logistic Regression.

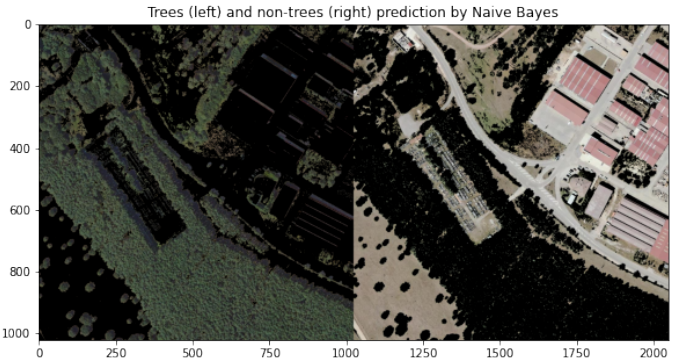

Figure 8: Colored prediction of whole image by Naive Bayes.

5 Packages

Employed packages and external functions were imported as:

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
import sklearn as sk
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import CategoricalNB, GaussianNB
```

6 Code

The image input, point selection and dataset generation was done as follows:

```
im_col = cv2.imread("map.jpeg")[:,:,::-1]
def choose_points(image, num_points=-1):
    %matplotlib tk
    plt.imshow(image)
    plt.title("Choose the points of correspondence")
    points = plt.ginput(n=num_points, timeout=-1, show_clicks=True)
    plt.show()
    return points

# Choose points manually:
points_tree = choose_points(im_col, num_points=100)
points_no_tree = choose_points(im_col, num_points=100)

# Get RGB features of each point:
points_tree_RGB = np.array([ im_col[round(coord[1]),round(coord[0]),:] for coord in points_tree])
points_no_tree_RGB = np.array([ im_col[round(coord[1]),round(coord[0]),:] for coord in points_no_tree])
points_tree = np.asarray(points_tree)
points_no_tree = np.asarray(points_no_tree)

# Prepare dataset
X = np.vstack((points_tree_RGB, points_no_tree_RGB))
y = np.hstack((np.ones(100), np.zeros(100)))

# Split train and test with random shuffle
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.5, shuffle=True)
```

As an example, the random forest was instantiated, trained, tested and employed to predict as:

```
random_forest = RandomForestClassifier(n_estimators=100)
random_forest.fit(X_train, y_train)
y_predict = random_forest.predict(X_test)
accuracy = np.sum(y_predict==y_test)/y_test.shape[0]
print(f"The accuracy on test set for the Random Forest is {accuracy*100}%")

y_predict_map = random_forest.predict(im_col.reshape(-1,3)).reshape(im_col.shape[:2])
plot(y_predict_map, title="Random Forest prediction for Trees in white",
     path="RF_prediction.png")

trees_im = np.where(y_predict_map[:,:,np.newaxis]==1, im_col, 0)
non_trees_im = np.where(y_predict_map[:,:,np.newaxis]==0, im_col, 0)
plot(np.hstack((trees_im, non_trees_im)), title="Trees (left) and non-trees (right) prediction by Random
    Forest", path="RF_prediction_col.png")
```