Assignment 1:

Extraction of Pixels of Ducks

Introduction

You are given an image of duck farm taken from a drone. Use the Bayes classifier to extract the pixels of duck bodies from the image. Convert all duck pixels into white and all non duck pixels into black.

Process

First, duck sample pixels were extracted and compiled into one image file. The same was done for non duck sample pixels. This was done by manually cropping out random pixels from the image. Each image was then read in and converted into an array of 3 dimensional ([blue, green, red]) feature vectors. The length of each array is equal to the product of the length and width of each image in pixels.

Next, the mean vector and covariance matrix were calculated for the duck and no duck pixel arrays created earlier. The mean vectors and covariance matrices for each were calculated using the following equations:

Mean vector = $\bar{X} = (\frac{1}{n})\sum_{i=1}^{n}(X_i)$

Covariance matrix = $(\frac{1}{n-1})\sum_{i=1}^{n}(X_i - \bar{X})(X_i - \bar{X})^T$

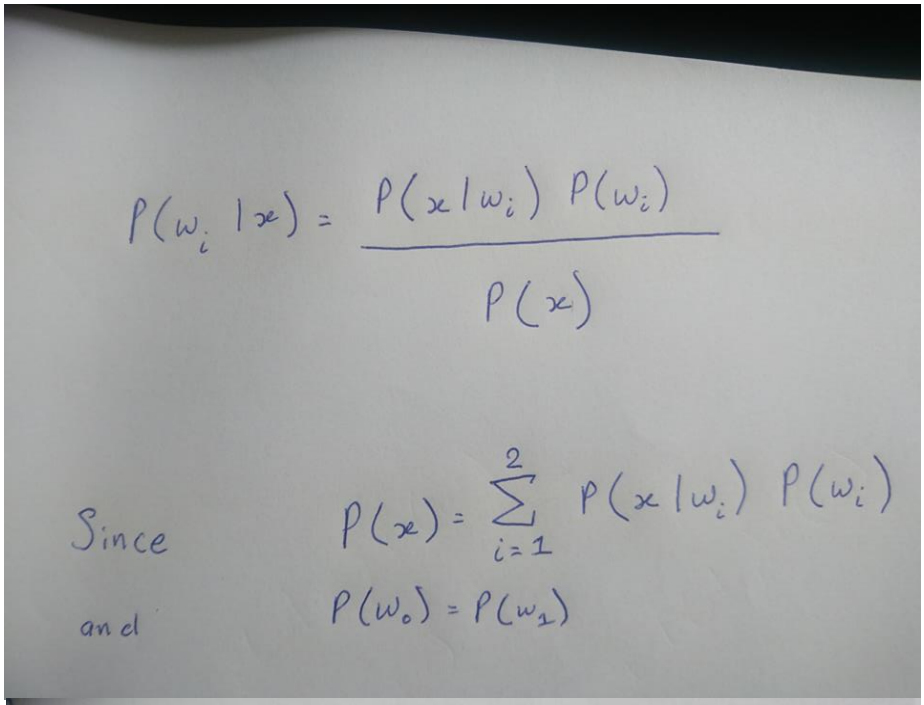
where X is the duck or no duck array, n is the length of each array.

Once the mean vector and covariance matrix were estimated for each sample, the Gaussian probabilistic models P(x | w=duck) and P(x | w = no ducks) were made using the equation:

$P(x \mid w) = \frac{1}{(2\pi)^{d/2}|\sum|^{1/2}} e^{[-\frac{1}{2}(x-\bar{x})^T\sum^{-1}(x-\bar{x})]}$

where the $\bar{x}$ and ∑ are the mean vector and covariance matrix for a particular w.

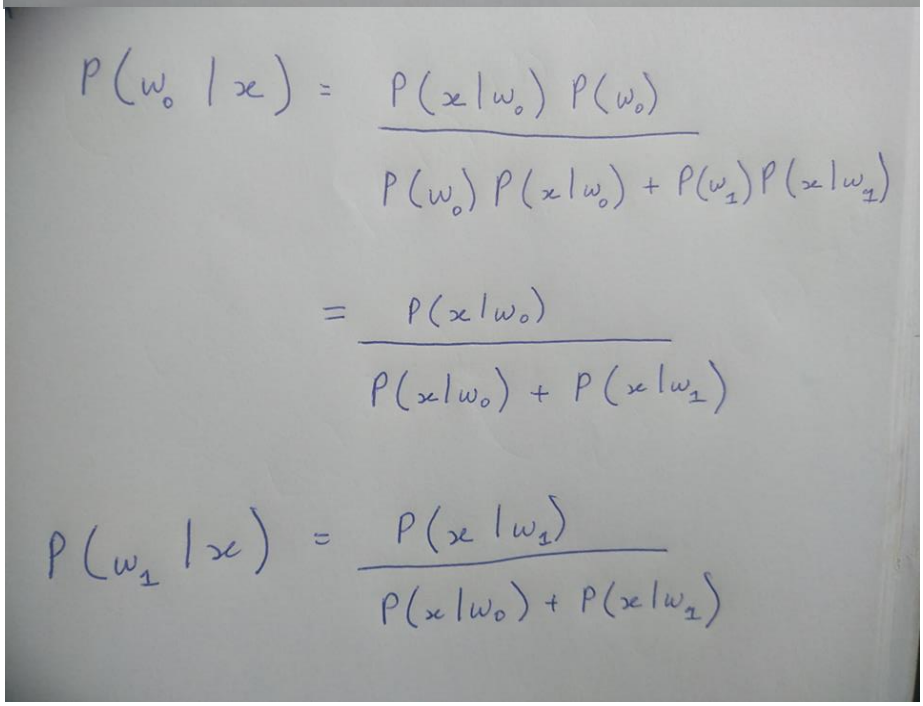
Next, the Gaussian probabilistic models were used to calculate the $P(w_i \mid x)$. Below shows the derivation of P(w | x) for w=ducks and w=no ducks



$$P(w_i \mid x) = \frac{P(x \mid w_i)\, P(w_i)}{P(x)}$$

Since $P(x) = \sum_{i=1}^{2} P(x \mid w_i)\, P(w_i)$

and $P(w_0) = P(w_1)$



$$P(w_0 \mid x) = \frac{P(x \mid w_0)\, P(w_0)}{P(w_0)\, P(x \mid w_0) + P(w_1) P(x \mid w_1)}$$

$$= \frac{P(x \mid w_0)}{P(x \mid w_0) + P(x \mid w_1)}$$

$$P(w_1 \mid x) = \frac{P(x \mid w_1)}{P(x \mid w_0) + P(x \mid w_1)}$$

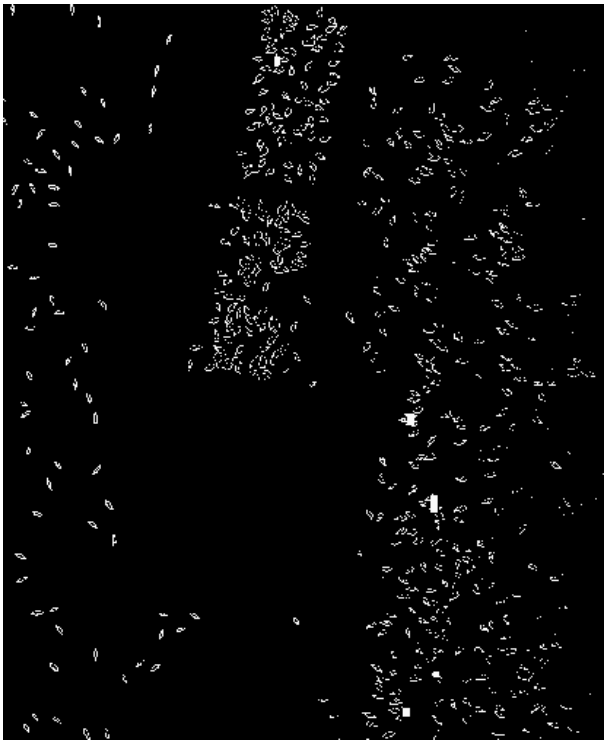
For our classification, we did $f(x) = P(w_1 \mid x) - P(w_0 \mid x) = \begin{cases} w1 \; if > 0 \\ w0 \; if \; not \end{cases}$

Since we are only checking if it is more than zero or not, the denominator is not important so $f(x) = P(w_1 \mid x) - P(w_0 \mid x) = P(x \mid w_1) - P(x \mid w_0)$

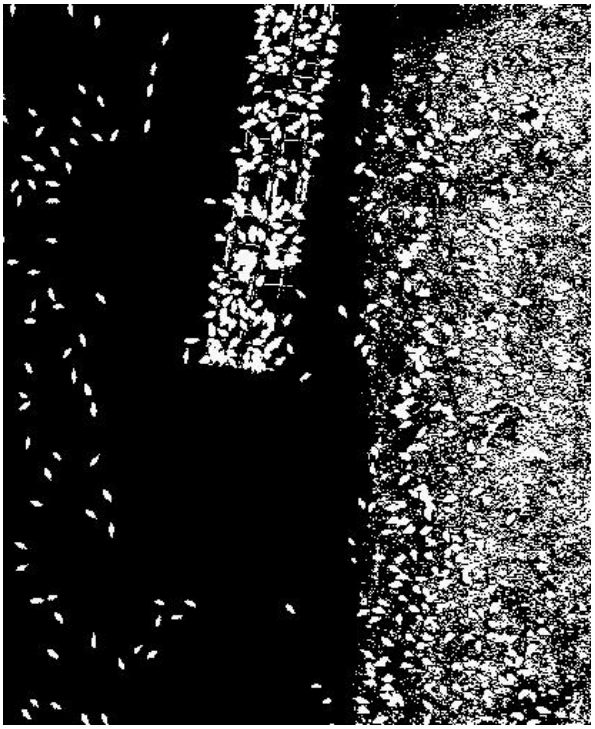
Finally, the pixel data for full_duck.jpg image was collected and classified based on the results of f(x).

Results

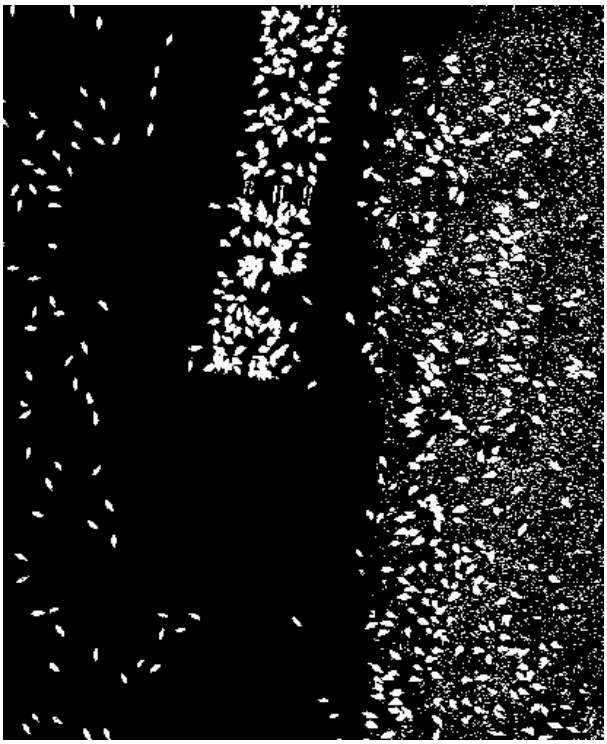
Attempt 1



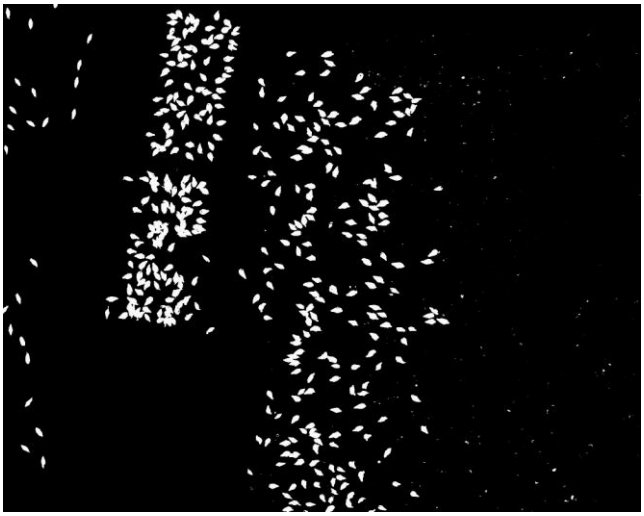
Attempt 2



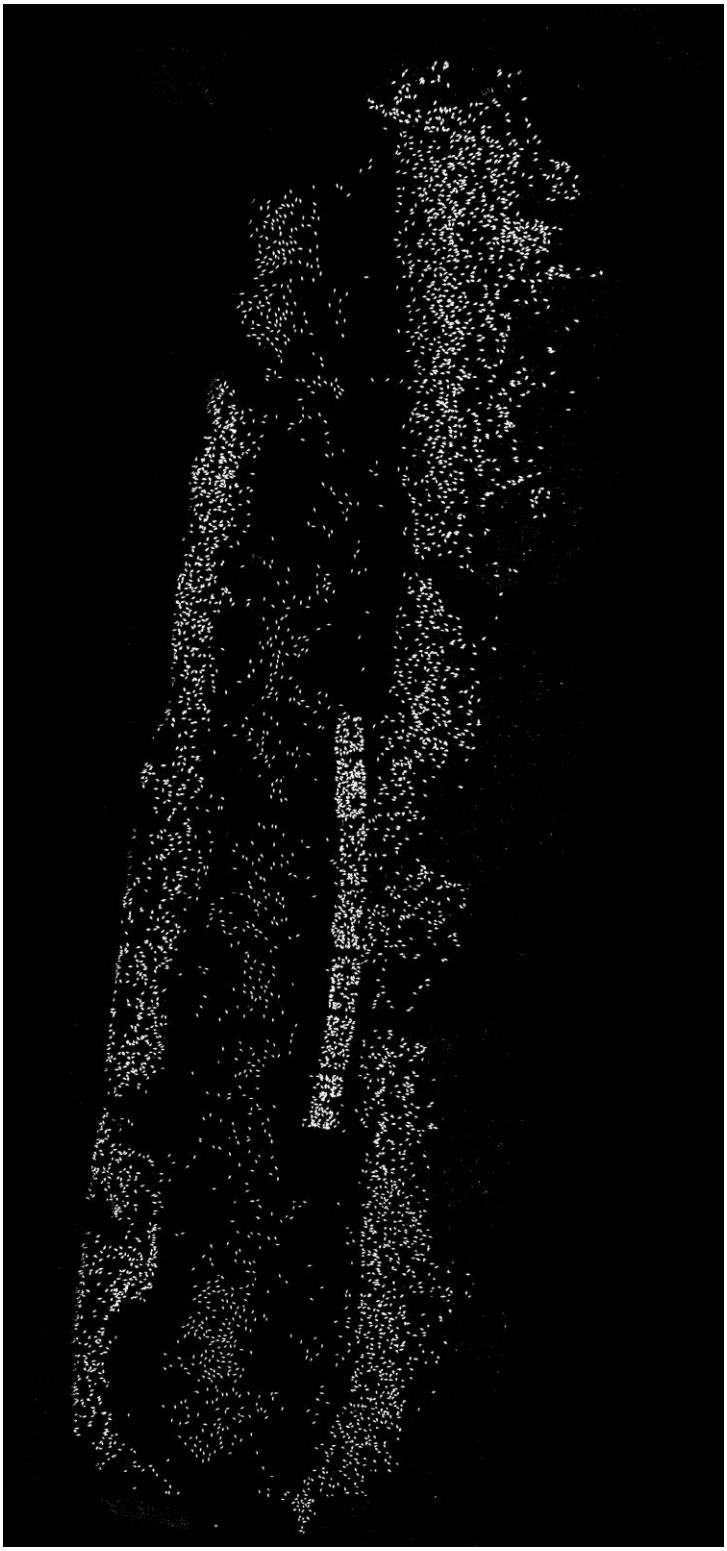
Attempt 3

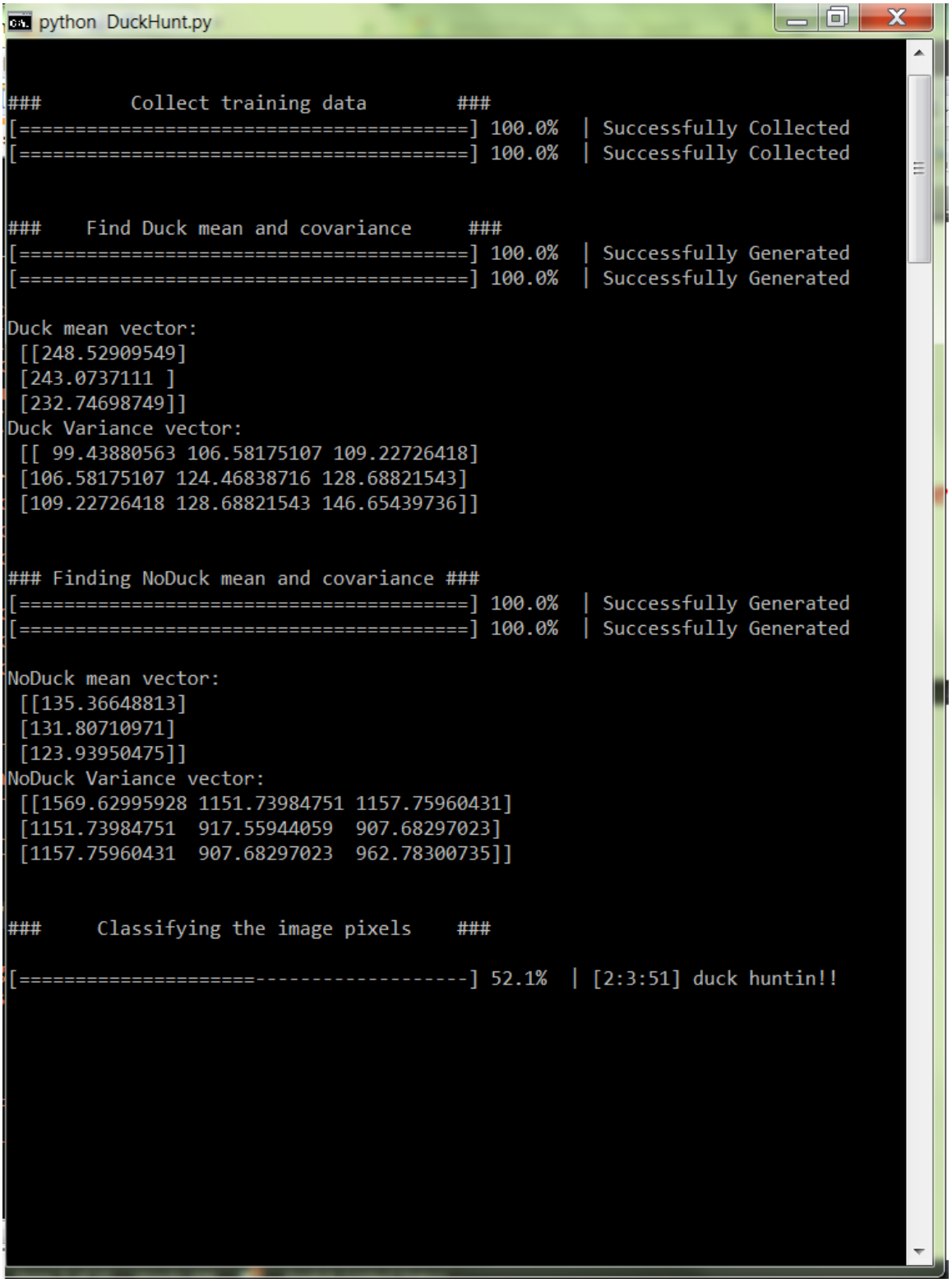


Attempt 4



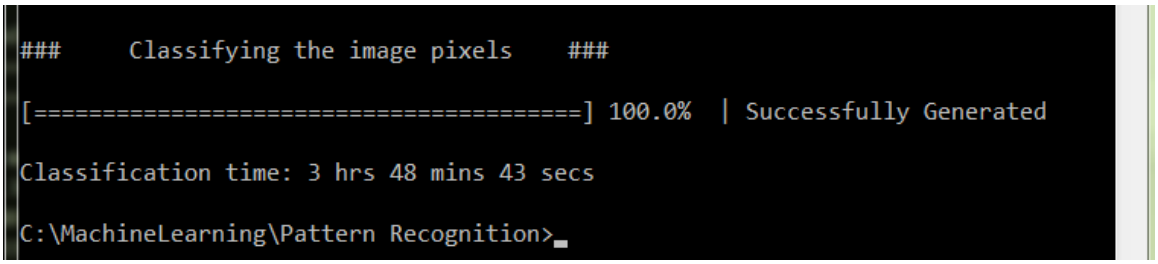
Final result





Mean vectors and covariance matrices for the duck and no duck training data

****The word"variance" changed to "covariance" in the code***


Classifying the image pixels
[==] 100.0% | Successfully Generated
Classification time: 3 hrs 48 mins 43 secs
C:\MachineLearning\Pattern Recognition>

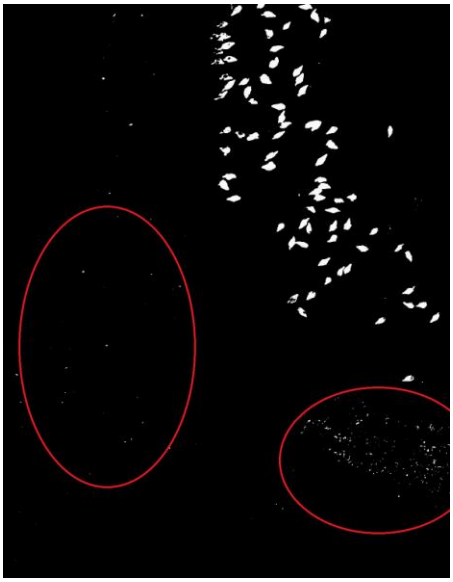
Discussion

While doing this assignment, I came across a few issues.

The first issue was that I had to wait very long for results while testing. To fix this, the main image was cropped into smaller test images to speed up testing and quickly find errors before doing a final run on the actual image.

Example of a test image cropped from the original for testing

Another issue was having rocks show up as ducks. Increasing the sample size of the ducks and the no duck helped to remove a lot of misclassifications however it was not perfect. Due to some rock pixels being similar or in some cases, exactly the same as duck pixels, there remained a few scattered white pixels that were not duck pixels.



Areas where the rock (non duck) pixels are misclassified as duck pixels

Attempts were made to fix this. The first attempt was to add more samples to the no duck and duck training data sets. This had zero affect on the results, even when the non duck objects were directly added to the no duck training data set.

Another attempt was made by using other pattern recognition techniques in conjunction with the Bayes Classifier. Tests were being done using k nearest neighbor techniques to identify clusters of duck pixels so that we can remove those duck pixels that are not in a large cluster.

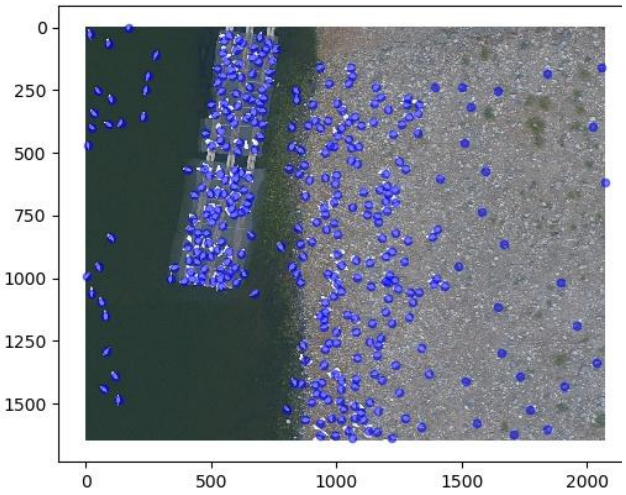


Image of duck pixel cluster center points

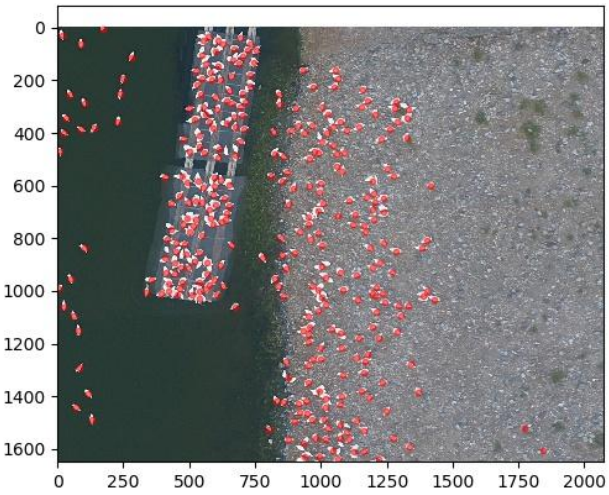


Image after clusters below a certain duck pixel count were removed and changed to non duck pixels

The above was done using the kmeans to find the center points of the clusters. It showed some success in the small scale image, but that was due to an educated guess on what value to set k to in the kmean classifier for this image. Finding a reasonable k value for the full image was still an issue that I was not able to solve in time to submit.

Since finding k was the issue, I tried using a clustering algorithm that doesn't require you to know k first (AgglomerativeClustering) but it was too inefficient to work with large data. It kept giving a memory issue.

An approach that calculated the distances of each pixel classified as a duck from 1 duck pixel at a time was also explored. The algorithm would calculate the distances from each point then find the average of the k closest points. If the average is below a certain value then it remains classified as a duck pixel, otherwise it is changed into a none-duck pixel. The test on this took too long so this approach was scrapped. I later realized the time complexity for the algorithm was O(n^2) which is terrible.

Summary

In conclusion, this assignment helped me to better understand Bayes classification and Gaussian probability. The lecture notes that were unclear to me became clearer after I implemented them myself.