ML 4: SUPPORT VECTOR MACHINE

Manuel, Ma. Janelle G.

AP 186

Objective

The main objective of this activity is to find the best decision line between the two classes using SVM algorithm.

SVM Algorithm

```
H = P
P = matrix(P)

q = np.ones((N, 1))
q *= -1.0
q = matrix(q)


G = np.zeros((N, N))
for i in range(N):
    for j in range(N):
        if i == j:
            G[i][j] = -1.0
G=matrix(G)


h = np.zeros((N, 1))
h = matrix(h)


sol = solvers.qp(P, q, G, h)
lagrange = np.array(sol['x']).reshape((H.shape[1],))


for i in range(N):
    lagrange[i] *= zset[i]

print('lagrange is: ')
print(lagrange)
weights = np.matmul(lagrange, dataset)
```

Figure 1. SVM Algorithm in Python

Results

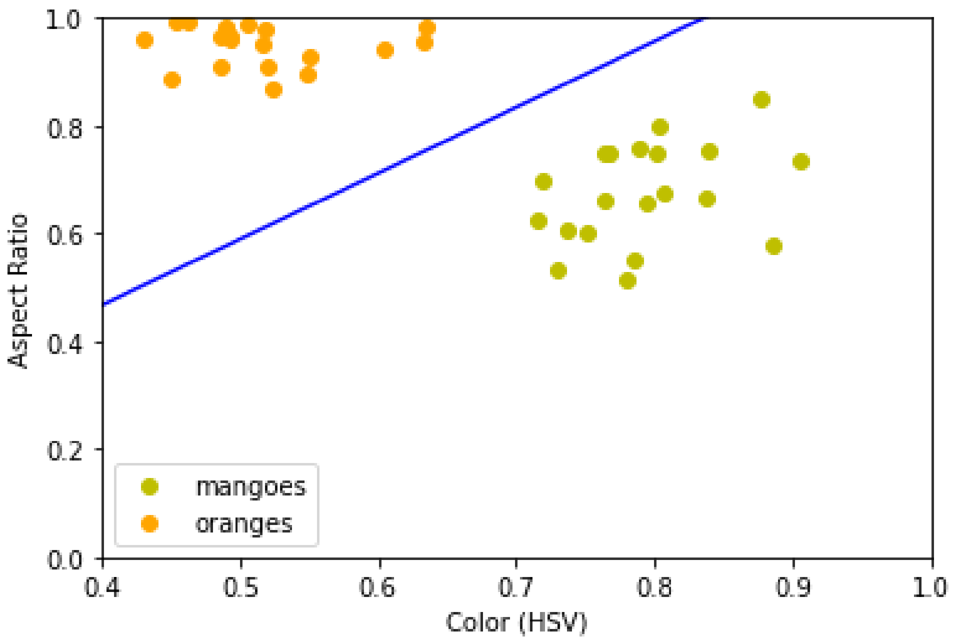

Figure 2. Best decision line between mangoes and oranges.

Using the algorithm in Figure 1, the weights and bias weight (w0) were obtained and found to be [-0.12114157, 6.53152757, -5.33773488] and -0.1211415708414493, respectively. The best decision line between mangoes and oranges was also plotted, shown in Figure 2, with a slope equal to 1.2236515539351738 and intercept equal to 0.022695314335630375.

Visually, the best decision line obtained for this dataset has equal margins between the two classes.

Results

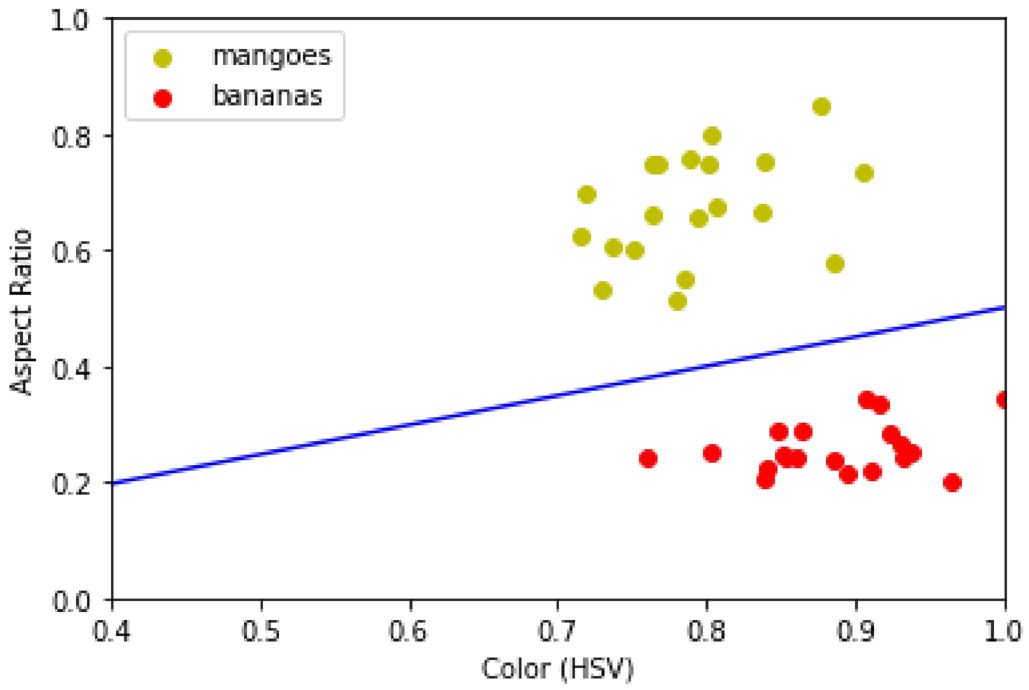

Figure 3. Best decision line between mangoes and bananas.

For the mangoes and bananas dataset, the weights and bias weight (w0) were found to be [0.04465778, -4.1368312, 8.16590155] and 0.04465778075472926, respectively. The best decision line between mangoes and oranges was also plotted, shown in Figure 2, with a slope equal to 0.5065982207073187 and intercept equal to -0.005468812042047929

Visually, the best decision line obtained for this dataset also has equal margins between the two classes.

Comparison with Perceptron

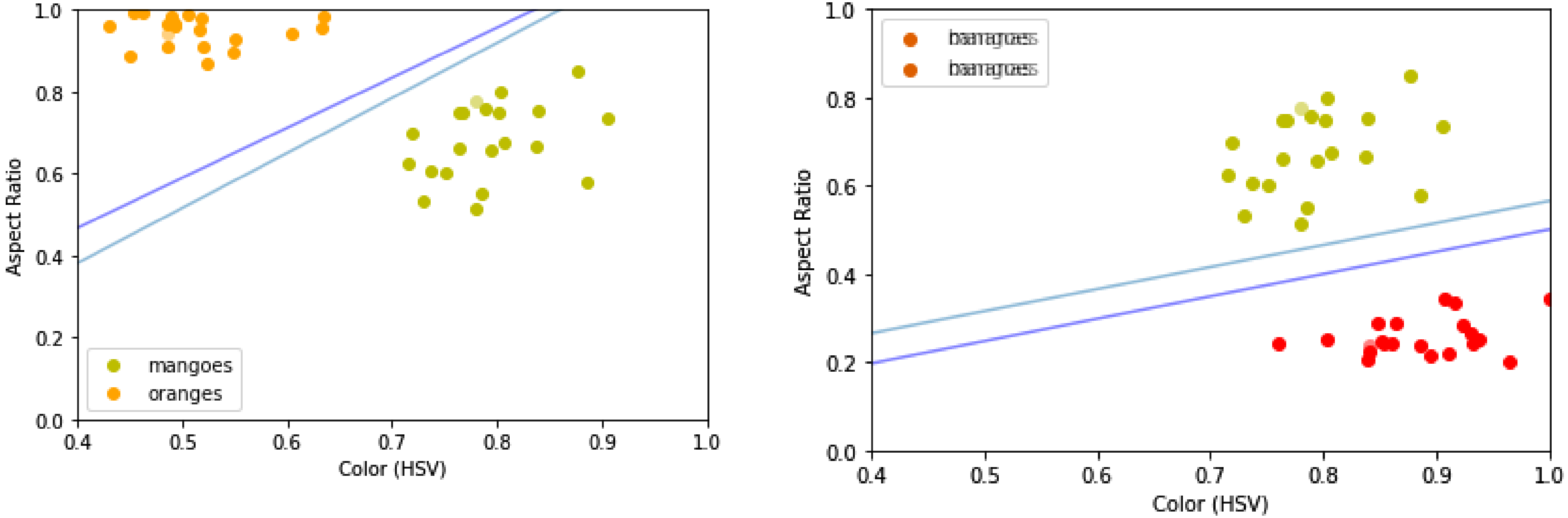

Figure 4. SVM results overlapped with the Perceptron results. Dark blue line is the SVM decision line while light blue line is the Perceptron decision line.

Analysis

SVM was successfully executed. It produced decision lines that have visually equidistant margins from the nearest feature vectors.

As compared to the Perceptron results, the decision lines obtained are independent to the initial weights and are indeed better in separating the two classes.

Reflection

Rating: 12/10

This one was really intimidating!!! It took me 2 days to actually understand how SVM works and how to make the algorithm in Python. I'm so relieved that the result is accurate! The margin between the classes ~seems~ equidistant!

References

- CVXOPT User's Guide:

https://cvxopt.org/userguide/coneprog.html

- Quadratic Programming with Python and CVXOPT:

https://courses.csail.mit.edu/6.867/wiki/images/a/a7/Qp-cvxopt.pdf

- Soriano, M. (2020). ML4 – Support Vector Machines.