Color Classification and Recycling Bin Detection

Lulua Rakla
PID A59012895
lrakla@ucsd.edu

I. Introduction

This report presents approaches to classify pixel color and identify blue recycling bins in images.

1. **Pixel Classification** This is a classic color identification problem. Given red, blue and green pixels as training data, the model learns to predict the color of the unseen image. For this, I have used Naive Bayes, a simple generative machine learning algorithm to perform multi-class classification. The images are trained in the RGB color space.
2. **Recycling Bin Detection** Given images of recycling bins as training data, the model learns to predict the blue recycling bins on unseen images. For this, I have again used Naive Bayes, a generative machine learning algorithm to perform multi-class classification. Here, instead of RGB color space, I used YCbCr color space. This makes the detection light invariant and performs well in images with sunlight and shadow. The use of 5 classes - blue, non-recycling bin blue, green, brown and black yields good results. Lastly, shape statistics like area, height and width are used to draw bounding boxes for recycling bins.

Both these problems have wide applications in object detection based on color and shape.

II. Problem Formulation

Gaussian Naive Bayes is a generative machine learning algorithm. It is based on Bayes Rule where given class conditional probability $P(X/Y)$ and prior probability $P(Y)$, posterior probability $P(Y/X)$ is calculated. The "Naive" assumption is that the features are independent of each other i.e Covariance Σ is a diagonal matrix.

The algorithmic approach is as follows :

1. Calculate Mean and Variance of each class and feature using *Maximum Likelihood Estimation*. Result is a *no of classes x no of features* matrix for Mean and Variance.
2. Calculate prior probability of each class
3. Calculate class label at test using formula in Fig. 2. The class which has the maximum posterior probability is selected.

GNB obtains the following MLE estimates of θ and ω:

$$\theta_k^{MLE} = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}\{y_i = k\}$$

$$\mu_{kl}^{MLE} = \frac{\sum_{i=1}^{n} x_{il}\mathbb{1}\{y_i = k\}}{\sum_{i=1}^{n}\mathbb{1}\{y_i = k\}} \quad \sigma_{kl}^{MLE} = \sqrt{\frac{\sum_{i=1}^{n}(x_{il} - \mu_{kl}^{MLE})^2\mathbb{1}\{y_i = k\}}{\sum_{i=1}^{n}\mathbb{1}\{y_i = k\}}}$$

Fig. 1. Gaussian Naive Bayes Parameters(from slides)

Given a test example $\mathbf{x}_* \in \mathbb{R}^d$, the GNB classifier produces the output:

$$y_* = \underset{y\in\{1,\ldots,K\}}{\arg\max} \log\theta_y^{MLE} + \sum_{l=1}^{d} \log\phi\left(x_{*l}; \mu_{yl}^{MLE}, \left(\sigma_{yl}^{MLE}\right)^2\right)$$

Fig. 2. Gaussian Naive Bayes Output(from slides [1])

θ_y : Probability of a class y
ϕ : Gaussian function
μ_{yl} : Mean of class y and feature l
σ_{yl} : Standard Deviation of class y and feature l

III. Technical Approach

In this section I will elaborate on the steps used to get the results.

1. Pixel Classification

The training data included images of Red, Blue and Green. There was almost equal number of samples as seen by prior probabilities. Using the mathematical equations defined in II, the mean, variance and prior probabilities was computed. These were then saved and reloaded in the classification module.

While I tried using YCbCr color space, the accuracy was better with RGB color space (It was 92% for blue color using YCbCr and 97% using RGB color space). Hence at testing, the BGR (default OpenCV color space) images were converted to RGB for classification.

TABLE I
Naive Bayes Parameters
(Order for Mean and Variance is RGB)

Class	Mean	Variance	Prior
Red	[0.75 0.35 0.35]	[0.037 0.062 0.062]	0.36
Green	[0.35 0.73 0.33]	[0.05 0.03 0.05]	0.32
Blue	[0.35 0.33 0.74]	[0.05 0.06 0.035]	0.30

2. Recycling Bin Detection

There were 60 training images given for this problem. To overcome the issues of lighting (shadows, sunlight, and different times of the day) I chose YCbCr color space. All the training images were converted to this before labelling. Using *roipoly* the training data labels were generated. Initially I used 2 classes - Recycling Bin Blue and Not Recycling Bin Blue. However, the results were poor.

Iteratively, I ended up using 5 classes which gave the best results. The data was labelled by looking at segmentation

masks on training images. For e.g if the road is being detected as blue, then I added that in the black class. If the sky was being detected as blue, I added it in the non recycling bin blue class. This greatly improved the performance of the model as it had more data to accurately formulate the distribution of the different classes.

After labelling, the Gaussian Naive Bayes was fit using the formulae in II. It gave the parameters as shown below.

TABLE II
NAIVE BAYES PARAMETERS
(ORDER FOR MEAN AND VARIANCE IS
BLUE, NON BLUE, GREEN, BROWN, BLACK)

Class	Mean	Variance	Prior
Blue	[0.34 0.43 0.66]	[0.03 0.002 0.009]	0.14
Non Recycling Blue	[0.54 0.41 0.62]	[0.039 0.002 0.004]	0.13
Green	[0.35 0.47 0.47]	[0.04 0.002 0.004]	0.18
Brown	[0.47 0.56 0.45]	[0.03 0.002 0.002]	0.2
Black	[0.27 0.47 0.52]	[0.01 0.00 0.00]	0.34

Apart from Naive Bayes, I tried using Gaussian Mixture Models, Multinomial Logistic Regression and Linear Discriminant Analysis. Without hyperparamter tuning Gaussian Mixture Models and Linear Discriminant Analysis had poor performance. Multinomial Logistic Regression had following parameters

TABLE III
LOGICTIC REGRESSION PARAMETERS

Class	coeff1	coeff2	coeff3	intercept
Blue	5.49	2.35489035e+01	1.02083255e+02	-68.95578341
Non Recycling Blue	5.77	-3.25898880e+01	6.51447289e+01	-22.02828778
Green	-6.67	-1.14112623e+02	-1.33554192e+02	128.24356146
Brown	-3.90058700e-02	1.14361968e+02	-1.12519841e+01	-49.57655652
Black	-4.54913038e+00	8.79	-2.24218080e+01	12.31706625

To identify bounding boxes from the segmented image shape statistics were used. This was mostly trial and error. The preprocessing done on the masks was as follows

1) Convert image to unint8 format which can be used by OpenCV
2) Erosion operation on the image using a (5,5) kernel. This separated the non recycling bin blue regions
3) Median Blurring using a (9,9) kernel. This was used to fill in the stray blue regions
4) Converting the image into black and white using thresholding
5) Finding all the contours in the image
6) Out of all the contours only the contours which have a boundingRect area greater than 10000 and height $>$ width was selected. This is to exclude small areas which are identified as blue regions. As all recycling bins are upright, the height has to be greater than width. This is also a limitation because the model does not detect horizontal or toppled bins.

Noteworthy Observations :

The case where two bins are to be detected (Validation Image 0067.png) was resolved by increasing the erosion iterations. Other cases were solved by putting the height $>$ width condition when including bounding boxes.

IV. RESULTS

A. Pixel Classification

The result of Naive Bayes is given in the table below

TABLE IV
PIXEL CLASSIFICATION ACCURACY ON VALIDATION DATA

Algorithm	Red	Green	Blue
Naive Bayes	100%	97%	97.05%

On test data(autograder), it gives an accuracy of 98%

B. Bin Detection

The result of Bin Detection on Validation Images is given in the table V. On Autograder, the model was able to detect 9.25/10 bins. It was unable to detect bin 3. Logistic Regression is unable to get bounding box for image 0062.png

In conclusion, having the YCbCr color space helped make the model light invariant. Further, using 5 classes instead of 2 helped the segmentation.

Lastly, the morphological operations were essential to get the correct bounding boxes.

Gaussian Naive Bayes, inspite of being a simple model was able to perform well due to the data and image processing. (I was unable to fix the formatting, inspite of many attempts. Results are on next page)

REFERENCES

[1] https://natanaso.github.io/ece276a/ref/ECE276A-5-GaussianDiscriminantAnalysis.pdf

TABLE V
BIN DETECTION CLASSIFICATION ACCURACY

Algorithm	Accuracy
Naive Bayes	100%
Multiclass Logistic Regression (trenary)	90%

TABLE VI
BOUNDING BOXES COORDINATES

Validation Image	Bounding Box Coordinates		
	Predicted	**Ground Truth**	**Accuracy**
0061	[203, 168, 309, 293]	[182, 101, 313, 295]	100%
0062	[29, 374, 133, 499]	[25, 347, 133, 497]	100%
0063	[174, 108, 266, 233]	[168, 64, 300, 239]	100%
0064	[359, 122, 458, 272]	[349, 104, 467, 264]	100%
0065	[814, 431, 928, 623]	[762, 416, 924, 622]	100%
0066	[]	[]	100%
0067	[584, 319, 695, 506], [709, 318, 824, 509]	[578, 305, 706, 504], [711, 305, 830, 509]	100%
0068	[]	[]	100%
0069	[]	[]	100%
0070	[]	[]	100%

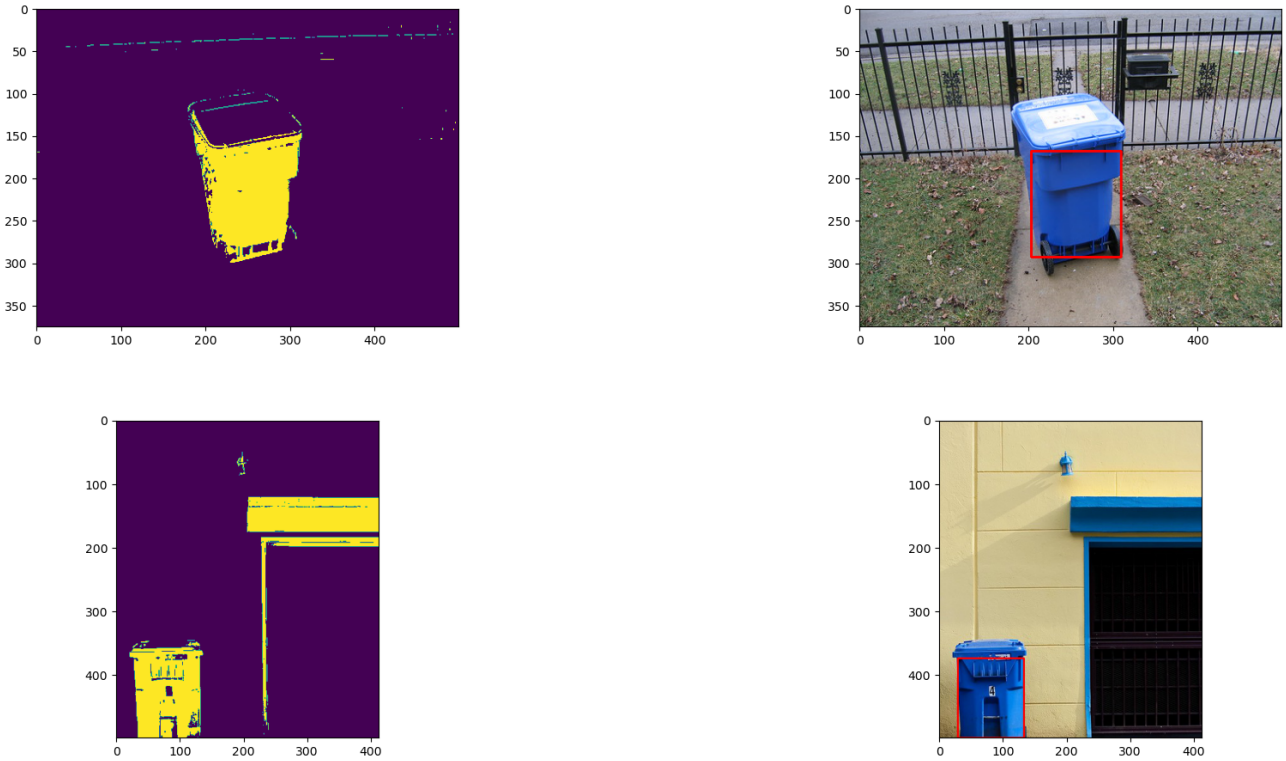

Fig. 3. Validation Images 0061 and 0062

Fig. 4. Validation Images 0063, 0064 and 0065

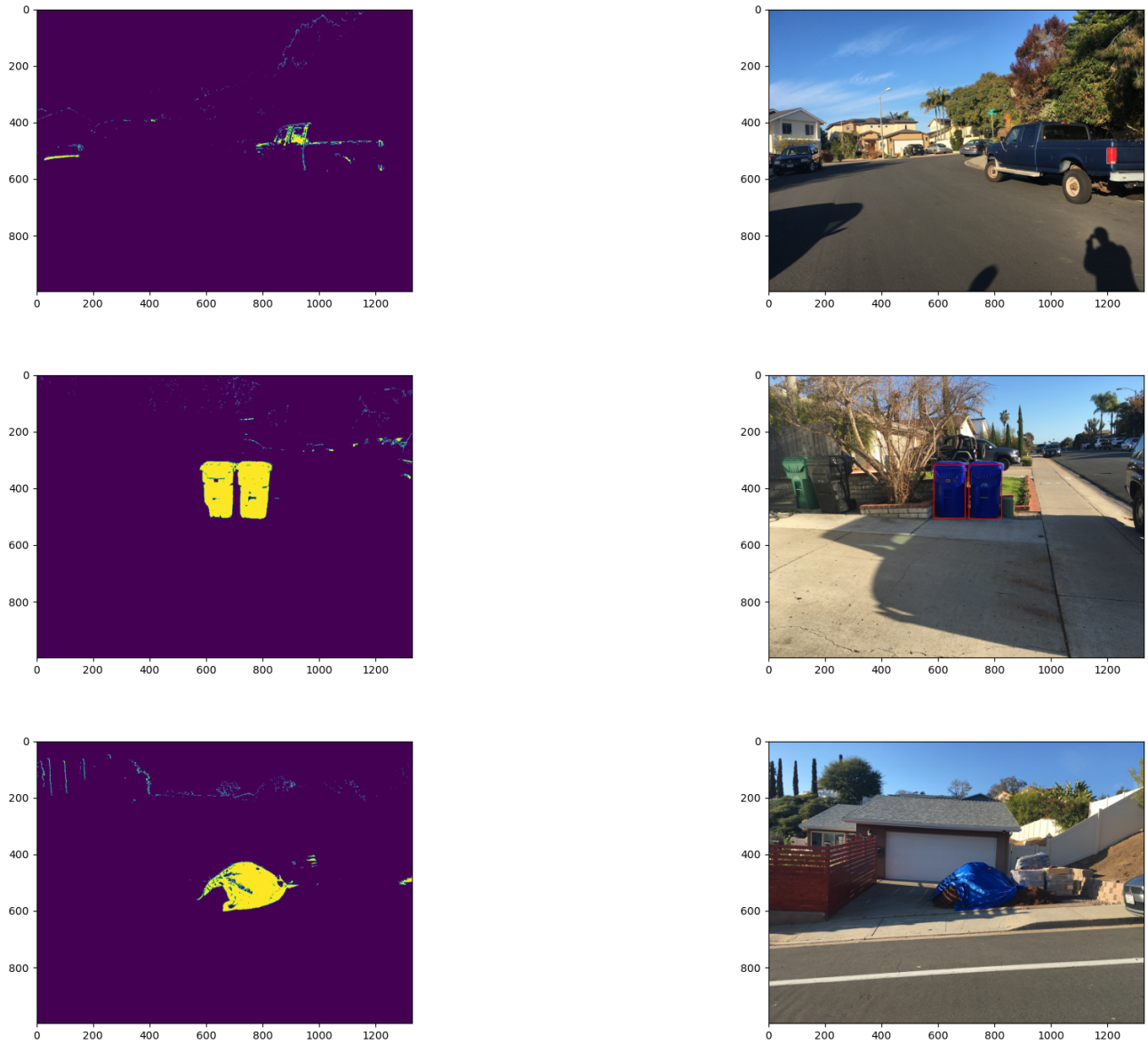

Fig. 5. Validation Images 0066.png - 0068.png in order

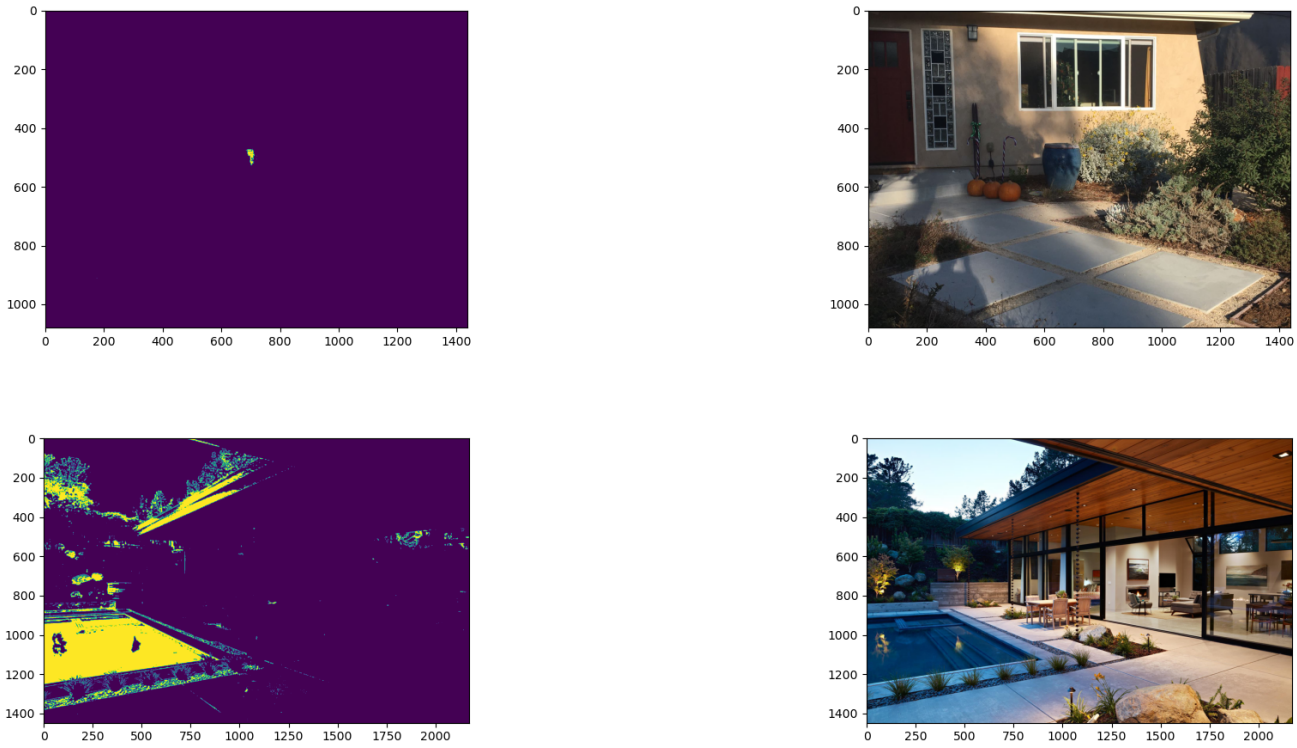

Fig. 6. Validation Images 0069.png and 0070.png in order

Fig. 7. An Additional Image from my kitchen was used to test the performance.