USING GAUSSIAN NAIVE BAYES FOR COLOR CLASSIFICATION AND RECYCLING BIN DETECTION: A PROJECT REPORT

Renwei Bai

Department of Mechanical and Aerospace Engineering
University of California, San Diego
rbai@.ucsd.edu

1. INTRODUCTION

Color classification,which is to classify different colors by color space is a basic application in computer vision. Robots can recognize different objects by computer vision,which use vision sensors to obtain vision information and probabilistic model to recognize the objectives.

In this color classification project, I made a classification in RGB color space. RGB color space is a basic color space that the rod and cone cells receive on human's retina. By transforming RGB color space, we can generate more color spaces:Luma-Chroma Color Space (a linear transformation of RGB), HSV (cylindrical coordinates of RGB points) and LAB (nonlinear transformation of RGB).

The essential part of this project is how to combine a probabilistic model with the color data generated from pictures. I use a Gaussian Naive Bayes Mode to train generative classification.

The rest of paper is arranged as follows. First I give the detailed formulations of GND problems in Section 2.And then I will describe my technical approach to color classification and recycling bin detection in section 3. At least, I present your training, validation, and test results, and discuss them.

2. PROBLEM FORMULATIONS

2.1. Color Classification Problem

Every color in RGB space have three values:value R,value G and value B. The range of each value is [0,255]. The color classification problem is that: given a training set,which contain 28×28 labeled example images $y \in \{1,2,3\}^n$ with a single RGB value in the standard format $X \in R^{n\times 3}$ at all of their pixels, where 1 = red, 2 = green, 3 = blue and n are the number of examples, try to use GND to train a a probabilistic color model from pixel data to distinguish among red, green, and blue pixels.

Thanks to XYZ agency for funding.

The final formula to label pixel can be written as

$$y_* = \underset{k\in\{1,2,3\}}{\arg\min} \left\{ \log \frac{1}{\theta_k^2} + \sum_{l=1}^{d} \left(\log \sigma_{kl}^2 + \frac{\left(x_{*l-\mu_{kl}^2}\right)}{\sigma_{kl}^2} \right) \right\} (1)$$

where x_{*l} is the color values of testing color blocks, y_* is the label of the testing color. The parameters (θ, μ, σ) in GND formula come from the training data, where θ_k is the proportion of each label color in all colors, μ_{kl} is the mean of each color value in each color label and σ_{kl} is the variance of each color value in each color label.

2.2. Bin Detection Problem

The method of bin detection is similar with color classification. The difference is that we can label the different colors as we need. To make the model performs better, I labeled four color classes, $y \in \{1,2,3,4\}^n$. In my model , 1 = green, 2 = other blue, 3 = bin blue, 4 = gray. The formula to detect the color of pixels in the validation set can be written as

$$y_* = \underset{k\in\{1,2,3,4\}}{\arg\min} \left\{ \log \frac{1}{\theta_k^2} + \sum_{l=1}^{d} \left(\log \sigma_{kl}^2 + \frac{\left(x_{*l-\mu_{kl}^2}\right)}{\sigma_{kl}^2} \right) \right\} (2)$$

3. TECHNICAL APPROACH

In this section we discuss about the algorithms and the models used in this project.

3.1. GNB for Color Classification

Naive Bayes uses a generative model $p\left(y, X \mid \omega, \theta\right)$, for discrete labels $y \in \{1, \cdots, K\}^n$ and assumes that, when conditioned on y_i , the dimensions of an example x_{il} for $l = 1, ..., d$ are independent. In the color classification part, apparently, the RGB values of each pixel in each color label are independent So the naive Bayes algorithm can be applied to the color

classification problem.
Due to the conditional distribution, the joint pdf $p(y, X)$ can be written as the multiplication of pdf $p(y \mid X)$ of Y conditioned on $X = x$ and the marginal pdf $p(x)$ of X. In the color classification problem,

$$p(\mathbf{y}, X \mid \omega, \theta) = p(\mathbf{y} \mid \theta)\, p(X \mid \mathbf{y}, \omega)$$

$$= \left(\prod_{k=1}^{3} p(y_k \mid \theta) \right) \left(\prod_{k=1}^{3} \prod_{l=1}^{3} p(x_{kl} \mid y_k, \omega) \right) (3)$$

Another important part of GNB is that the RGB values X follows a Gaussian distribution and the color labels y follows a Categorical distribution, which gives that

$$p(y_i \mid \theta) := \prod_{k=1}^{3} \theta_k^{1\{y_i = k\}} (4)$$

$$p(x_{il} \mid y_i = k, \omega) := \phi\left(x_{il}; \mu_{kl}, \sigma_{kl}^2\right) (5)$$

And the MLE estimates of θ and $\omega := \left\{\mu_{kl}, \sigma_{kl}^2\right\}$ obtained from GNB are following

$$\theta_{kl}^{MLE} = \frac{1}{n} \sum_{i=1}^{n} 1\{y_i = k\} (6-1)$$

$$\mu_{kl}^{MLE} = \frac{\sum_{i=1}^{n} x_{il} 1\{y_i = k\}}{\sum_{i=1}^{n} 1\{y_i = k\}} (7-1)$$

$$\sigma_{kl}^{MLE} = \sqrt{\frac{\sum_{i=1}^{n} (x_{il} - \mu_{kl}^{MLE})^2 1\{y_i = k\}}{\sum_{i=1}^{n} 1\{y_i = k\}}} (8-1)$$

This parameters are calculated from the training data. They describe the distribution of each color value in each color class. Substitute formula (4) and (5) into (3) with MLE parameters(6) (7) (8). Given a validation example RGB value $x_* \in R^3$, we will have

$$y_* = \underset{y \in \{1,2,3\}}{\arg\max} \theta_y^{MLE} \prod_{l=1}^{3} \phi\left(x_{*l}; \mu_{yl}^{MLE}, \left(\sigma_{yl}^{MLE}\right)^2\right) (9)$$

Because $X \sim N(\mu, \sigma^2)$, we have

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2} \frac{(x-\mu)^2}{\sigma^2}\right) (10)$$

Substitute formula(10) into (9) and then take a $\log$ on the RHS. Formula (9) becomes

$$y_* = \underset{y \in \{1,2,3\}}{\arg\min} \frac{1}{\theta_y^2} + \sum_{l=1}^{3} (\log \sigma_y^2 + \frac{(x-\mu)^2}{\sigma_{yl}^2}) (11)$$

By calculating the minimum of formula (11), the label of x_* is y_*, which will tell us what the color of a testing pixel is according to the training set (where 1 = red, 2 = green, 3 = blue).

3.2. GNB for Bin Detection

This part is really similar with the previous one. What we need to do in algorithm is to adjust the color classes and collect new RGB data to generate a set of new parameter,which is training our program again.
We train our program with the same formulas:

$$\theta_{kl}^{MLE} = \frac{1}{n} \sum_{i=1}^{n} 1\{y_i = k\} (6-2)$$

$$\mu_{kl}^{MLE} = \frac{\sum_{i=1}^{n} x_{il} 1\{y_i = k\}}{\sum_{i=1}^{n} 1\{y_i = k\}} (7-2)$$

$$\sigma_{kl}^{MLE} = \sqrt{\frac{\sum_{i=1}^{n} (x_{il} - \mu_{kl}^{MLE})^2 1\{y_i = k\}}{\sum_{i=1}^{n} 1\{y_i = k\}}} (8-2)$$

With the new label of color ,$y \in \{1,2,3,4\}^n$,K=4,(1 = green, 2 = other blue, 3 = bin blue, 4 = gray) , and new parameter, the program can recognize the color of the different regions, by following formula:

$$y_* = \underset{y \in \{1,2,3,4\}}{\arg\min} \frac{1}{\theta_y^2} + \sum_{l=1}^{3} (\log \sigma_y^2 + \frac{(x-\mu)^2}{\sigma_{yl}^2}) (12)$$

After identifying the color regions, we need use shape statistics to filter the unwanted pixels,which makes the picture a binary image. Only the bin blue pixels will be left on the canvas. Then we can use the **regionprops** function from the **scikit-image package** to locate the objective pixels and draw a bounding box around the blue bin on the original image.

4. RESULTS

These figures are the segmented color images and their bounding boxes, which are the results of my bin detection program running locally. **Fig.1 -Fig.20** are the results of my 4-color-class GBN model. **Fig.21- Fig.26** are the results of 2-color-class GNB model.**Fig.27-Fig.28** are the results of 3-class-model GNB model.
By analysing the first 20 figures, it reveals some disadvantages of my GNB model with RGB color space. The illumination affects the accuracy of RGB model, which has been shown in **Fig.9,13,21,25**. When the bin blue regions are in the shadow, the model can not recognize them very well. Because RGB space can not describe the brightness of

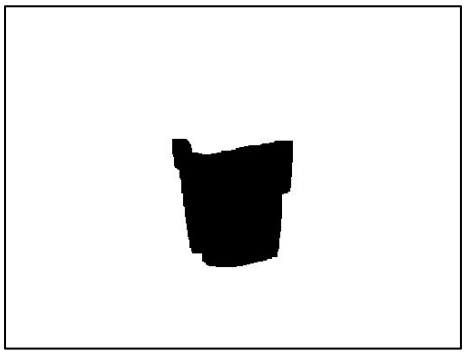

Fig.1. Segmented color images of image 0061

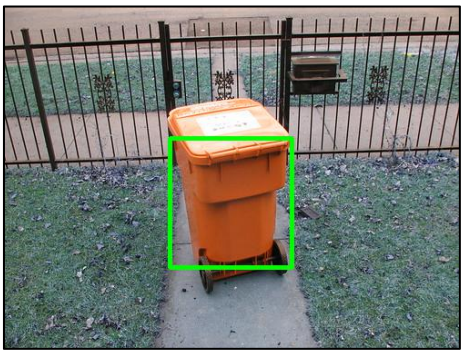

Fig.2. Bounding box of image 0061

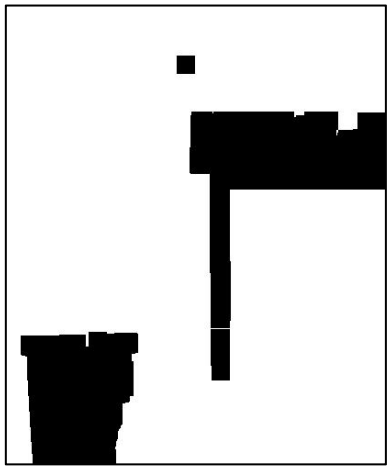

Fig.3. Segmented color images of image 0062

Fig.4. Bounding Box of image 0062

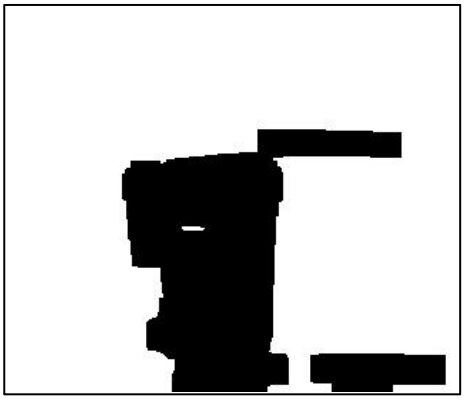

Fig.5. Segmented color images of image 0063

Fig.6. Bounding Box of image 0063

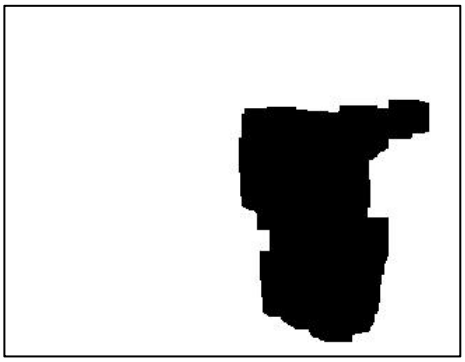

Fig.7. Segmented color images of image 0064

Fig.10. Bounding box of image 0065

Fig.8. Bounding box of image 0064

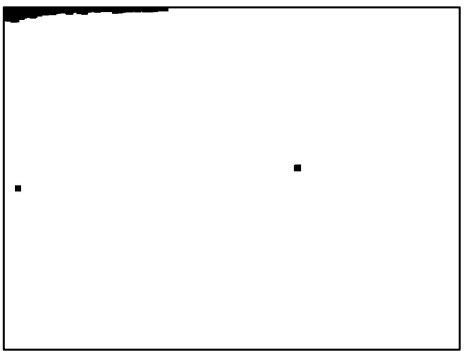

Fig.11. Segmented color images of image 0066

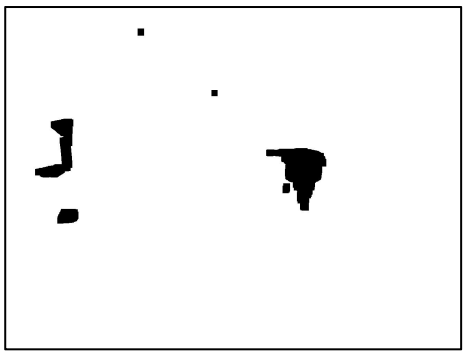

Fig.9. Segmented color images of image 0065

Fig.12. Bounding box of image 0066

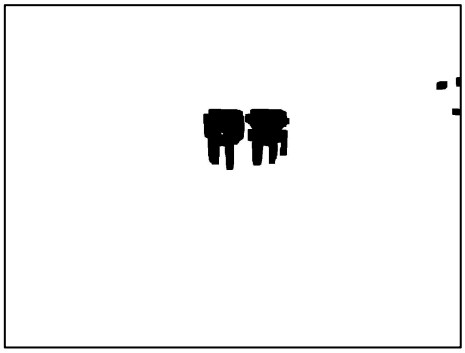

Fig.13. Segmented color images of image 0067

Fig.14. Bounding box of image 0067

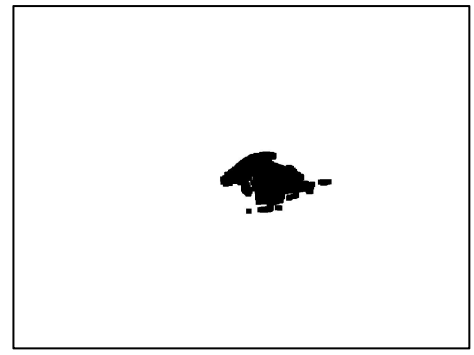

Fig.15. Segmented color images of image 0068

Fig.16. Bounding box of image 0068

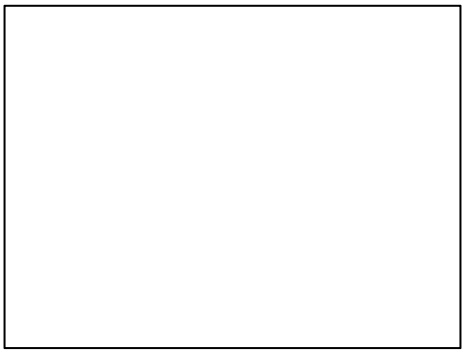

Fig.17. Segmented color images of image 0069

Fig.18. Bounding box of image 0069

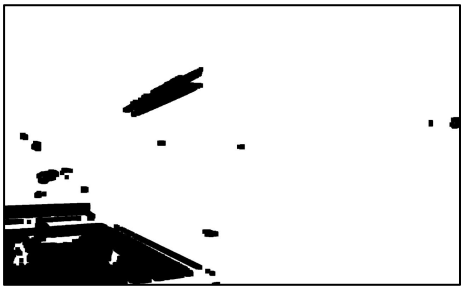

Fig.19. Segmented color images image of image 0070

Fig.20. Bounding box of image 0070

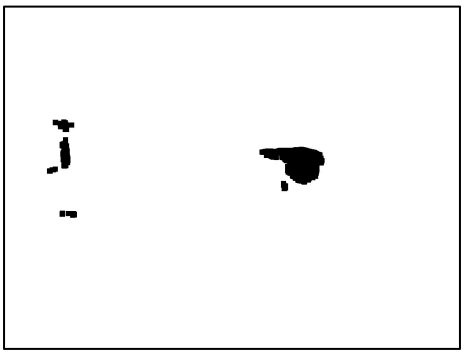

Fig.21. Segmented color images of image 0065

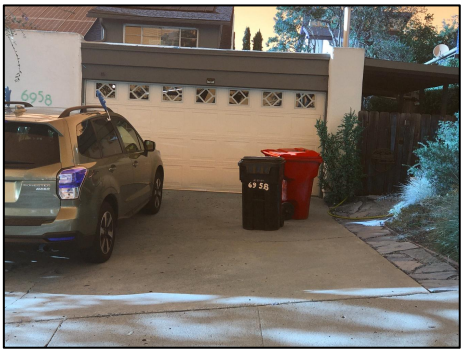

Fig.22. Bounding box of image 0065 2-class-model

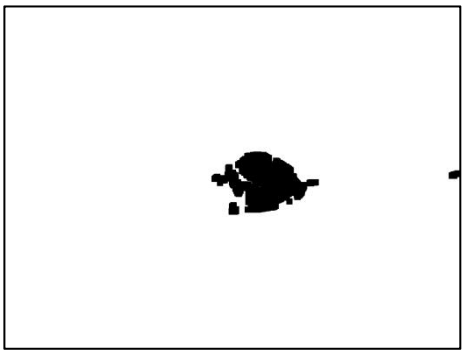

Fig.23. Segmented color images of image 0068 2-class-model

Fig.24. Bounding box of image 0068 2-class-model

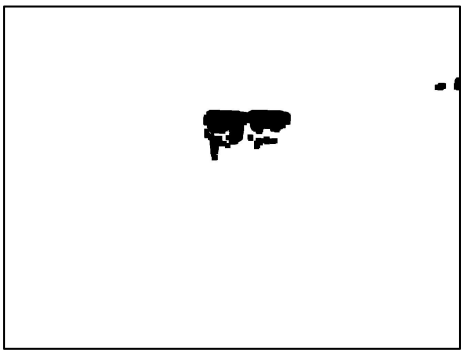

Fig.25. Segmented color images of image 0067
2-class-model

Fig.26. Bounding box of image 0067
2-class-model

Fig.27. Segmented color images of
image 0061

Fig.28. Segmented color images
of image 0061

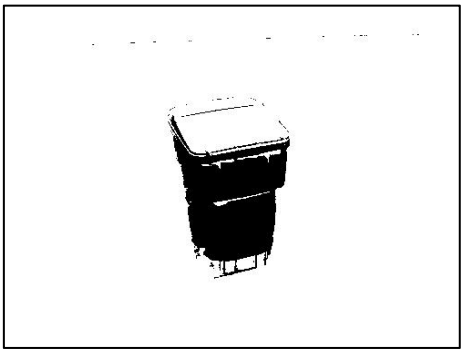

Fig.29. Segmented color images
of image 0061

a color, the RGB value of the same color in different lighting conditions vary greatly. It generates a bigger variance in bin blue data, which will misguide the system to recognize other colors which are very bright or dark as bin blue. Additionally, when the illumination condition is not applicable, my model probably fail to detect targets.

Another disadvantage of my model is that the parameter estimation strategy is MLE, which means that parameters are totally decided by the training data. In the bin detection part, we need to use the **roipoly** labeling tool to generate color model training data. Due to the θ part in formula(12), the number of the selected pixels do affect the accuracy of the model. Fig.27-Fig.28 are the mask image of validation image given by 3-color-class ($y \in \{1,2,3\}^n$, 1=green, 2=other blue, 3=bin blue) GNB model. As shown in Fig.27, apparently the model labeled ground gray as bin blue. After recollecting training data with more pixels in bin blue region. The accuracy improved slightly, as shown in Fig.28. But the model still can not tell the difference between the bin blue and road gray. To solve this problem, I added a new color class—gray.Comparing Fig.1 with Fig.27 and Fig.28, the 4-class model performs much better than 3-class model without changing the algorithm. And the 4-class model also performs better than 3-class.

However, what is interesting is that a 2-class model performs better than the 3-class model. In Fig.29, the 2-class (bin blue and not bin blue) model clearly identified bin blue and filtered other colors. By checking the parameters training from the 2-class model, I found that $\mu_{Binblue} = [48, 82, 180]$, $\mu_{Notblue} = [119, 119, 100]$. The RGB values of color in 'Not Blue' class are really close compared with the RGB value of 'Bin Blue' class. But the variance of 'Bin Blue' is much smaller than 'Not Blue'. The variance of 'Not Blue' is 92.6% , 14.1%,43.6% respectively larger in [R,G,B] than 'Bin Blue', which means that the 'Bin Blue' class is more concentrated and this model can recognize bin blue better with appropriate light condition than an incomplete 3-class model.

The parameters of GNB 4-class model in RGB color space are,

$$\theta^{MLE}_{Greenl} = 0.1671354091910917$$

$$\theta^{MLE}_{OtherBluel} = 0.21956985044735236$$

$$\theta^{MLE}_{BinBluel} = 0.4352497168086564$$

$$\theta^{MLE}_{Gray} = 0.17804502355289956$$

$$\mu^{MLE}_{Greenl} = [0.30109736, 0.38706712, 0.29420989]$$

$$\mu^{MLE}_{OtherBluel} = [0.40145388, 0.58459644, 0.82202932]$$

$$\mu^{MLE}_{BinBluel} = [0.17873478, 0.25588107, 0.60130495]$$

$$\mu^{MLE}_{Grayl} = [0.57775111, 0.56725118, 0.54256678]$$

$$\sigma = \begin{bmatrix} 0.02794657 & 0.03675439 & 0.02984469 \\ 0.02851198 & 0.02108734 & 0.01116654 \\ 0.02204423 & 0.02702302 & 0.03394953 \\ 0.03872675 & 0.02805644 & 0.02288151 \end{bmatrix}$$

The parameters of GNB 4-class model in RGB color space are,

$$\theta^{MLE}_{BinBluel} = 0.33499901602528237$$

$$\theta^{MLE}_{NotBinBlue} = 0.6650009839747176$$

$$\mu^{MLE}_{BinBluel} = [0.18792637, 0.32230437, 0.70552324]$$

$$\mu^{MLE}_{NotBinBluel} = [0.46629683, 0.46531039, 0.39169227]$$

$$\sigma = \begin{bmatrix} 0.03224907 & 0.04533985 & 0.04815073 \\ 0.06210578 & 0.05169293 & 0.06915282 \end{bmatrix}$$

5. CONCLUSION

In all, I implemented the GNB based probabilistic color model in RGB color space to classify colors and detect recycling bin. My model performs rather well. Its accuracy is 8/10 when I test it with validation set locally. But the accuracy given by autograder does not satisfied me. I tired a lot on collecting data set and adding the classes of colors of my model. But this not work. As I have analysed above, there are at least two shortcoming of my model, the light condition and MLE parameter estimation strategy due to the assumption of GNB. The RGB color space does not describe the brightness, which do affect the accuracy of the model. I think another color space like HSV space will be better. The reason why we use GNB is that we think the color distribution follows Gaussian distribution. But could it be that when I collect the training data, I am unintentionally affecting the distribution? A suggestive way to refine the result is to try another probabilistic model in different color space.

Exercis 4.

Solution: $X \sim N(u, \sigma^2)$ $\quad p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2}\frac{(x-u)^2}{\sigma^2}\right)$

(a) $\because \{x_i\}_{i=1}^n$ are n independent samples obtained from X

$$\therefore f(x_1, x_2, \dots, x_n) = f(x_1) f(x_2) \dots f(x_n)$$
$$= \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \exp\left[\sum_{i=1}^n \left(-\frac{1}{2}\frac{(x_i-u)^2}{\sigma^2}\right)\right]$$

Then transforming the objective by a monotone function (log)

Use MLE to estimate the mean

$$u^* \in \arg\max_u f(\underline{x}) = \arg\max_u \log(f(\underline{x}))$$

(b) $\log(f(\underline{x})) = \underbrace{-\frac{n}{2}\log 2\pi - \frac{n}{2}\log\sigma^2}_{\text{independent of } u} - \sum_{i=1}^n \frac{(x_i-u)^2}{2\sigma^2}$

$$\therefore u^* \in \arg\max_u (\log f(x)) = \arg\min_u \sum_{i=1}^n \frac{(x_i-u)^2}{2\sigma^2}$$

To solve the unconstrained optimization, set the gradient equal to 0:

$$0 = \nabla_u \left(\sum_{i=1}^n \frac{(x_i-u)^2}{2\sigma^2}\right)$$
$$= \frac{\partial \sum_{i=1}^n \frac{(x_i-u)^2}{2\sigma^2}}{\partial u}$$

$$= \frac{1}{2\sigma^2} \sum_{i=1}^n (2u - 2x_i)$$

$$\therefore u^* = \frac{\sum_{i=1}^n x_i}{n} = \bar{x}$$

$$\therefore \hat{u}_{MLE} = \bar{X}$$

(c) $u^* \in \arg\max_u p(u | x_1, \dots x_n) = \arg\max_u p(x_1, \dots x_n | u)\, p(u)$

$$= \arg\max_u \prod_{i=1}^n p(x_i | u)\, p(u)$$

$$\prod_{i=1}^{n} p(x_i|u) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_i-u)^2}{2\sigma^2}\right)$$

$$p(u) = \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp\left(-\frac{(u-u_0)^2}{2\sigma_0^2}\right)$$

$$\therefore u^* \in \underset{u^*}{\operatorname{argmax}} \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_i-u)^2}{2\sigma^2}\right) \cdot \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp\left(-\frac{(u-u_0)^2}{2\sigma_0^2}\right)$$

take log.

$$u^* \in \underset{u}{\operatorname{argmax}} \left(\sum_{i=1}^{n} -\log\sqrt{2\pi\sigma^2} - \frac{(x_i-u)^2}{2\sigma^2}\right) - \log\sqrt{2\pi\sigma_0^2} - \frac{(u-u_0)^2}{2\sigma_0^2}$$

$$\therefore u^* \in \underset{u}{\operatorname{argmin}} \left(\sum_{i=1}^{n} \log\sqrt{2\pi\sigma^2} + \frac{(x_i-u)^2}{2\sigma^2}\right) + \log\sqrt{2\pi\sigma_0^2} + \frac{(u-u_0)^2}{2\sigma_0^2}$$

$$\therefore \nabla_u \left[\left(\sum_{i=1}^{n} \log\sqrt{2\pi\sigma^2} + \frac{(x_i-u)^2}{2\sigma^2}\right) + \log\sqrt{2\pi\sigma_0^2} + \frac{(u-u_0)^2}{2\sigma_0^2}\right] = 0$$

$$\therefore \sum_{i=1}^{n} \frac{d(x_i-u)^2}{2\sigma^2 du} + \frac{d(u^2-2u_0u+u_0^2)}{2\sigma_0^2 du} = 0$$

$$\therefore \frac{-2\sum_{i=1}^{n} x_i + 2nu}{2\sigma^2} + \frac{2u-2u_0}{2\sigma_0^2} = 0$$

$$\therefore -\sigma_0^2 \sum x_i + \sigma_0^2 nu + \sigma^2 u - \sigma^2 u_0 = 0$$

$$\therefore u = \frac{\sigma_0^2 \sum_{i=1}^{n} x_i + \sigma^2 u_0}{\sigma_0^2 n + \sigma^2}$$