German University in Cairo
Faculty of Media Engineering and Technology
Winter 2021

CSEN 1022 – Machine Learning

Assignment #2

(Due on December 20 at mid-night)
(This assignment can be done in teams of maximum 2 students)

Problem 1

You are required to design a Naïve Bayes classifier that can recognize images of three classes (airplane, automobile and frog). These classes are part of the machine learning benchmark CIFAR-10 dataset. The data is available on the GitHub assignment https://classroom.github.com/a/FRXkLpPg. In the Data folder, you will find two folders: "train" and "test". The "train" folder contains 3 sub-folders named "airplane", "automobile" and "frog", with each one containing 5000 different images of the respective classes. The "test" folder also contains the same 3 subfolders with 1000 different images of each class. The images in the "train" folder should be used to train a classifier for each class using the method given at the bottom of slide 18 in Lecture 3.pdf. After the classifiers are trained, test each classifier using the images given in the "test" folder. Assume that each pixel is distributed according to a Gaussian distribution whose parameters should be identified in the training phase of the Naïve Bayes classifier. In this assignment, you will apply the classifier to RGB images only. Repeat the same process for the images when resized to 16 by 16, 8 by 8 and 4 by 4.

Deliverables:

- Your code.
- 4 confusion matrices one for each scale (32 by 32, 16 by 16, 8 by 8 and 4 by 4).

Problem 2

For the same dataset, apply the maximum likelihood estimation of the Gaussian generative model to classify the images to the same 3 classes. In your impelementation when classifying a test image, identify the class by computing $\arg\max_k \Pr\left(\mathbf{x}\middle|C_k\right)\Pr\left(C_k\right)$ where

$$p(\mathbf{x}|\mathcal{C}_k) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\mathbf{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)^{\mathrm{T}}\mathbf{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_k)\right\}$$

Repeat the same process for the images when resized to 16 by 16, 8 by 8 and 4 by 4.

Deliverables:

- Your code.
- 4 confusion matrices one for each scale (32 by 32, 16 by 16, 8 by 8 and 4 by 4).
- Comparing the outcome of Problem 1 to Problem 2, **which** classification method is better? **Is** there a justification for this difference in accuracies?

German University in Cairo
Faculty of Media Engineering and Technology
Winter 2021

CSEN 1022 – Machine Learning

Assignment #2

(Due on December 20 at mid-night)
(This assignment can be done in teams of maximum 2 students)

Important Notes:

- Divide all pixel values by 255 to scale them between 0 and 1.
- A confusion matrix is a 3 by 3 matrix that shows how each image was classified to belong to which class. (For example: The first row would show how each of the 1000 airplane images were classified as airplane, automobile and frog, and so on for the other classes).
- To resize the images to 8 by 8, use *cv2.resize(image, (8,8), interpolation=cv2.INTER_AREA).*
- In problem 2, you can calculate the exponential part alone as the determinant will be 0.
- DO NOT use *np.cov()* in order to calculate the covariance matrix in problem 2.
- To get the inverse of the covariance matrix, use *np.linalg.pinv()* as it might not be a singular matrix.
- Compared to the explanation in the lecture, the problem to be solved here has 3 classes not 2. Therefore, in problem 2, add an additional term to the covariance matrix that represents the third class.
- Make sure you save all 8 confusion matrices in their correct order in their designated variables.
- Add the comment in the last cell where it says "*Edit Here*"
- Save the notebook results before you submit it on GitHub. DO NOT CLEAR THE OUTPUT.
- **You have to implement your own version of all needed functions. You are only allowed to use the mean and variance functions, functions that load images into Python and the libraries already imported in the first cell in the code provided on GitHub (numpy, matplotlib and opencv).**