ENGR421 HW2 Report

Berke Can Rizai - 69282

First, we do the imports.

With the genfromtxt with numpy, we take images to dataframe. Astype(‘uint8’) is to decrease memory and speed up the process later. Instead of taking precise floats we can get away with this.

Since labels are esentially just 1...5, I took them with uint8 again to decrease processing time and RAM usage.

A screenshot of a computer

Description automatically generated with medium confidence

I do the train test split with the following, :30000 means we take from 0 to 30000 elements of the dataframe. Same thing for the label.

In the test set, we take from 30000 to the rest of df.

Graphical user interface, text, application, chat or text message

Description automatically generated

Text

Description automatically generated

This splits the data into classes so that we can calculate the parameters following this. We printmean

Table

Description automatically generated



I found the deviations with the numpy std method.

Printing them, Table

Description automatically generated with medium confidence

I took the safelog method from lab material, because log 0 was printing error and wasn’t consistent.

We don’t need safelog for frequency however, it prints different if I don’t use it. Axis=0 is so that it makes it for each feature.

It is;







I found classpriors with the sum of of each label appearance time divided with all length of labels array.

My score function is as follows;

Here, freq returns the log of priors for each given class.

I divided the formula from the lecture to few parts to make it easier to understand and write. Text

Description automatically generated

piSigmaSq returns square root of 1/2pi\*std^2

Originally, in the first term which is everything except the freq, expotential is inside the logarithm. However, since equation simplifies if we take log of exponent, I have written the getExp as the, the results will be the same.

firstTerm is just the first term, as I have mentioned, it is simplified and I took the log of 1/piSigmaSq for the appropiate class i.

In the scoreFunc, I pass i and x as arguements. i is class number so that I can calculate the score function of an image for a given class which I will pass from the getResult. So, i is class number and x is image as an array.

Text

Description automatically generated

In the getResult, I return the class label that is most likely, that has largest score function. I first set an initial max and go over each of them to find max and return the label. I return +1 since labels are actually 0,1,2.. but they should be 1,2,3 etc.

Than I set up an empty list to add my guesses.

Graphical user interface, text

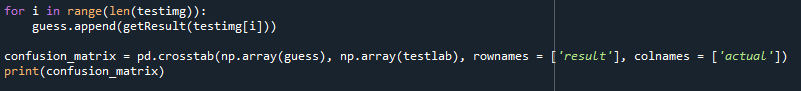
Description automatically generated

For each image, I add my guess to the list with getResult and pass image inside.

I set up confusion\_matrix with the guess as an np array because otherwise it doesn’t work. I also take trainlabels dataframe as an array and get the first column of it (index 0) for dimension match.

Than print it.

I, reinit the list with list().



I do the same steps for the test split of the dataframe and check the results with the confusion matrix. The output is this;

A screen shot of a computer

Description automatically generated with low confidence