DAI\_ASSIGNMENT\_1\_Q3\_Report.pdf

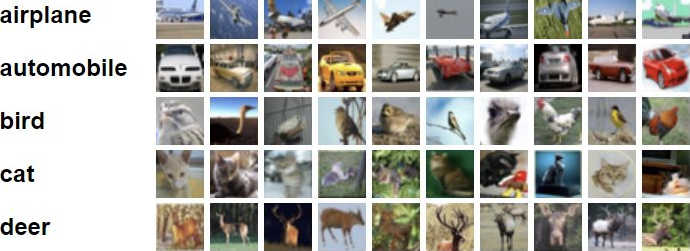
**Dependable AI CSL7370 – Assignment 1**

**Task 3:**

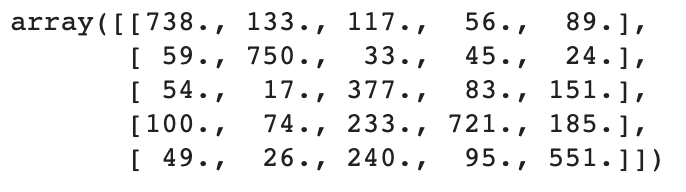
🡺 **Preprocessing Data:**

This step includes fetching of data from library of tensorflow. Steps included are as follows:

1. Importing the datasets from tensorflow.keras import datasets using this command and then load cifar10 dataset. This process is simple as you just have to load the dataset into variables (train\_images, train\_labels), (test\_images, test\_labels). Till this step we are splitting the dataset into variables. 50000 in training and 10000 in testing.
2. Now we need to drop 5 classes of choice. (In my case I have dropped bottom 5 classes)
3. In order to drop the classes, you can also find the indices of the classes which you need to incorporate for training and learning.
4. Indices of the requires classes are used to filter the dataset and fetch only those tuples which align to the respective index. This will get us all samples from the dataset of that particular class.
5. Similarly all training, testing datasets can be filtered as per requirement.
6. Once this process is done, you can concatenate for the sake of collaborative learning process. Such that, our X\_train, Y\_train, X\_test and Y\_test will have 25000, 25000, 5000, 5000 samples respectively.



1. Just to visualize the image we define a function : show\_label\_and\_image
2. Now that we have training and testing dataset, we can proceed to training process. The training process includes 3 layers in a simple Sequential Artificial Neural Network.
3. Additionally, the model uses Stochastic Gradient Descent to learn the updates. Loss used is CEL. This is preferred for classification datasets. To show the learning metrics we display accuracy measure. [Epochs = 10]
4. To print confusion matrix there are 3 methods. First: using inbuilt function (from sklearn), Second: using pandas.crosstab, Thirdly: from scratch using for loop
5. We shall focus on third method (I have solved with first 2 methods in notebook).
6. From scratch we just require to start with 2D array. Initialize 2D array using np.zeros. using : np.zeros((len(labels), len(labels)))
7. Once done we can start iterating rows and columns such that storing the counts of each comparison from predicted label and the truth label by comparing. The condition returns 0 if not match or 1 if match. np.sum((Y\_actual == labels[i]) & (Y\_pred == labels[j])) This line of code will store the sums of “1” in each iteration and updates the 2D array. We are checking 2 conditions here, first one will check whether the actual label and the label in our row iteration matches. Second condition checks whether predicted label matches with the label in our column iteration. (The comparison was possible because of comparing numpy array with numpy integer and not with native python integer)

****