# *Coursework 1 – Data Preparation and Classification*

Date Released: Friday, 18th October 2024

Date Due: Friday 8th November 2024

This coursework is worth 20% of your overall mark

## 1. Introduction

In this coursework, you are required to carry out a mini research project in Machine Learning classification. This will take the form of supervised image classification on the CIFAR-10 dataset. You will be required to load the data, select target classes, create training and testing subsets, and use these to train several machine learning models and perform inference on the testing data. You will be required to evaluate and compare the results, using this to determine the most suitable model.

The submission will include the code as a single .m file, a 1-page extended abstract and a data file as a .mat file, containing your indices and result measures. An abstract template is included for use and an example abstract accepted by a conference is included for reference.

## 2. Dataset

The CIFAR-10 dataset contains 60,000 images in 10 classes along with labels showing the object that they represent. The dataset is contained in the file `cifar-10-data.mat`. This is 174MB but we will not use the whole set for training and testing. This mat file contains:

- A 60000x32x32x3 variable `data` holding the images.
- A 60000x1 variable `labels` holding the corresponding image labels.
- A 10x1 variable `label_names` showing the categorical names of each indexed label.

Each image is of size 32 x 32 x 3 pixels. Since the first dimension of the data variable is the image index, to view an example image, you should use the `squeeze` command. E.g.
`imagesc(squeeze(data(120,:,:,:)));`

*Figure 1 - Example Data from CIFAR-10*

## 3. Data Preparation

**Data Loading:** To use the images with your method effectively, you will need to load the data into MATLAB, and convert the **images** to a double data type once imported.

**Image Visualisation:** Produce a 1-row and 4-column array of figures (Hint: the subplot command will be useful here), showing randomly-chosen examples and including the corresponding labels as the subfigure titles. Save this figure as a portable network graphic (png) file and use it in your report to show what your data looks like.

**Class Selection:** Since we have a lot of data, we will only work with three classes. Select these three classes pseudo-randomly using the `randperm` command with a Mersenne Twister. You must use your student ID as a seed for the random number generator. You will find the `rng` function useful for this. For example, if my ID is 200212365, I would use classes 6 (dog), 9 (ship) and 10 (truck). Store this list in a 3x1 vector called "classes"; you will need to submit this along with your code. Extract the data corresponding to these classes, while preserving the original order. Do not forget to ensure that your labels vector needs to correspond to your image data so this will also need to be extracted. You should now have a data matrix of size 18000x32x32x3 and a labels vector of size 18000x1.

**Training and Testing Split:** For this task, we will use a 50-50 pseudo-randomly-selected training-testing split. Split your dataset into training and testing subsets, each containing 50% of the data. Use your student ID again as a seed for a Mersenne Twister random number generator and the randperm command to select half of your images (i.e. 9000 images) for your training set. Note that you will need to set the seed each time you want to use it. Store the index resulting from randperm as a 9000x1 vector called "training_index"; you will need to submit this along with your code. Now, use this index to extract your training images and labels; the remaining 9000 images and labels should be set aside for testing. Be careful to ensure that the labels still correspond to the correct images.

**Data Format:** As in week 3, the image data needs to reshaped into a suitable format for training and testing and you will need four matrices: in this case, you should have a 9000x3072 matrix each for your training and testing data, and a 9000x1 matrix each for your training and testing labels.

Check your label distribution to examine the representation of each class in training and testing. You can comment on this in your report but must not change the data split.

## 4. Model Training and Evaluation

### 4.1. Model Training with *K*-Nearest Neighbour

In this section, you will be required to classify the test data using your own implementation of *k*-nearest neighbour (see Week 3 Lab for reference). This should be carried out separately using two different distance metrics to provide one set of results per distance metric:

1. $L^2$-distance, also known as Euclidean distance.
2. Another distance metric of your choice but not an $L^p$-distance. For example, you could use Bray-Curtis or Cosine Distance.

Be sure to save the prediction results and computation times for evaluation and comparison later.

### 4.2. Model Training with Existing Models

It is important to compare with existing models and we often do this by implementing the author's code. We will use some existing classification algorithms that are already implemented in MATLAB.

You will need to train and compare **any two** classification algorithms implemented in MATLAB, not including KNN. It is sufficient for this coursework to use the models' default parameters but you are welcome to optimise these parameters to improve performance. Please note that, given the dataset size, the models make take several minutes to run.

Table 1 shows some examples of classification algorithms in MATLAB and their function names as implemented in MATLAB's Statistics and Machine Learning Toolbox$^{TM}$. You may find it useful to have a look at this MATLAB documentation page on Supervised Learning Workflow and Algorithms for some ideas on how to structure your solution and select your algorithms. You are encouraged to make use of the suggestions on this page but feel free to explore further.

*Table 1: MATLAB Classification of Algorithms and their function Names*

| **Algorithm** | **Function** |
|---|---|
| SVM for Multiclass | fitcecoc() |
| Decision Tree | fitctree() |
| Ensembles | fitcensemble() |

## 4.3. Evaluation

It is important to evaluate methods from multiple points of view. For this work, please evaluate the models in terms of their accuracy, confusion matrix and the time taken to train and test. These can then be presented and discussed in your report to select the most suitable model for your problem. Please name the corresponding variables according using the format "modelname_measure", e.g. "knnL2_accuracy", "SVM_timetaken", "decisiontree_confusionmatrix". You will need to submit these in a data file.

Accuracy is measured by comparing the predictions from your models to the test labels in the dataset. Accuracy can be calculated as

$$accuracy = \frac{number\ of\ correct\ predictions}{total\ number\ of\ labels}$$

The time taken to train the models can be computed using the `tic` and `toc` commands.

## 5. Report

As part of this coursework, you are asked to write a 1-page report in the form of an Extended Abstract, which is a common way of submitting work to a conference or workshop. A template is provided in the file `SCC361-CW1-Report-Template.docx` and an example abstract accepted at BioMedEng23 is included in `Example-ExtendedAbstract-BioMedEng23.docx`.

The report must include the following sections:

1. **Introduction:** This should be a paragraph, describing the problem that you are solving and potential uses for it.
2. **Data and Preparation:** Describe the content of the dataset and the data preparation method. Please include the figure that you generated in Section 3.
3. **Methodology:** Describe the model training and evaluation methods used, including your reasons for choosing the models. You should briefly describe each method but you do not need to give a detailed technical description.
4. **Results:** This should include presentation of your results in a table and the confusion matrices as figures, along with your error analysis and observations

5. **Conclusion:** In this section, you should identify the model that you would recommend for use with your problem and justify your reasoning.

Most conference submissions will not permit you to change the margins, font or font size. Beyond the report title, please use the font Arial at size 10.

Ordinarily, in a paper submission, you would include the names and affiliations of the authors at the top, as in the template. Please do not do this; this submission should be anonymous.

## 6. Submission

Please submit the data file, code and report on Moodle on or before the **deadline of 4pm on Friday, 8th November, 2024**. The data file must be submitted as a single file called "cw1.mat" contain only the following variables; you do not need to submit the whole dataset:

- "classes" variable
- "training_index" variable
- Accuracy, confusion matrix and time taken measures.

You can do this with the `save` command, which takes the format
```
save('cw1','variable1','variable2','variable3');
```

You code should be submitted as a **single** MATLAB script (.m file), which is running without any errors, structured and includes documentation/ detailed comments.

Your 1-page report should be submitted as a pdf file, without your name.

There will be separate submission points for the data file, code and for the report. If for some reason, you cannot upload your submission, please zip it up and e-mail it to b.williams6@lancaster.ac.uk and scc-teaching-office@lancaster.ac.uk as soon as possible. You do not need to submit the rest of your data.

Your submission should be anonymous and not include your name in the data file, code or report.

## 7. Marking

Marks and feedback will be returned by 29th November 2024. There are a total of 20 marks for this coursework and the coursework mark constitutes 20% of your overall mark for this module.

- 10 marks are allocated to correct implementation and results.
- 5 marks are allocated to your code, including appropriate structure and suitable annotation/documentation of your code using comments.
- 5 marks are allocated to your report, with 1 marks per section mentioned above.