AN OVERVIEW OF TEMPLATES FOR COURSEWORK 2

Files provided as the templates:

- Main_Program.R – Main entry point to the program. Can be used to import CSV dataset, locking aside validation set, train, test, cross-validate, plot learning curves for a multiclass perceptron.
- Perceptron.r – Basic implementation of a single layer perceptron algorithm. Can be used to model any classification problem with 4 features and class label.
- Evaluation_Cross_Validation.r - Function that performs 10 fold cross validation over training dataset.
- Evaluation_Validation.r - Function that computes accuracy, confusion matrix and precison over testing dataset. Can be used to test fitted model over unseen data (validation set).
- Evaluation_Curves.r - Function that plots learning curve for accuracy vs training set size. Can be used to plot various learning curves to decide optimum values of hyperparameters like learning rate and number of epochs (iterations).

To use these templates, all files including csv file should be uploaded in the same place (working directory).

Please note, the templates and examples of code for Coursework 2 are given for owl dataset - owls.csv. You can use them as a reference project for your own one, which should be tuned for penguin dataset.

Example dataset for these templates is an owl species dataset.

Owl dataset includes four parameters (body_length, wing_length, body_width and wing_width) from three species: barn owl, snowy owl and long eared owl.

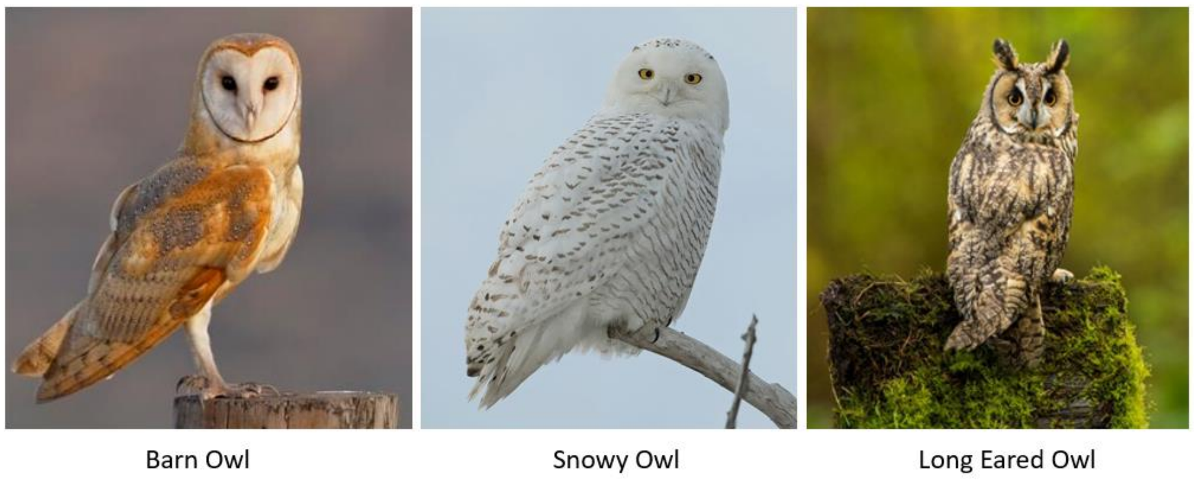

Figure 1. Owl species used in example dataset

The base model of a perceptron is developed taking into account following elements:

Synaptic Weights: Each input feature has a corresponding weight associated with it. Output class for every set of inputs is decided based on weighted sum of the inputs. We calculate the dot product of weight vector (w) with feature vector(X) to obtain this weighted sum: $f(x)= w\cdot x$ (see figure below).

Activation function: An activation function is used to fire a neuron based on inputs, in order to produce an output signal. For basic implementation, the unit step function has been employed as the activation function in Perceptron algorithm. It is a discontinuous function that outputs +1 for +ve argument and -1 for -ve argument.

Learning Rate: Optimum values of synaptic weights are achieved by training Perceptron by gradient descent. Learning rate denotes the step size during gradient descent. This parameter decides on how fast or slow does the Perceptron learn from training samples. Ideally, this parameter is tuned to attain a value that is high enough for Perceptron to make a nearly accurate guess about outcome and low enough that algorithm converges in fewer iterations to a meaningful predictor.

Epochs: A perceptron is trained through multiple passes over training dataset. Each pass over training dataset is referred to as an epoch. Number of iterations or epochs need to be tuned in order to let the algorithm learn sufficiently from the training instances.

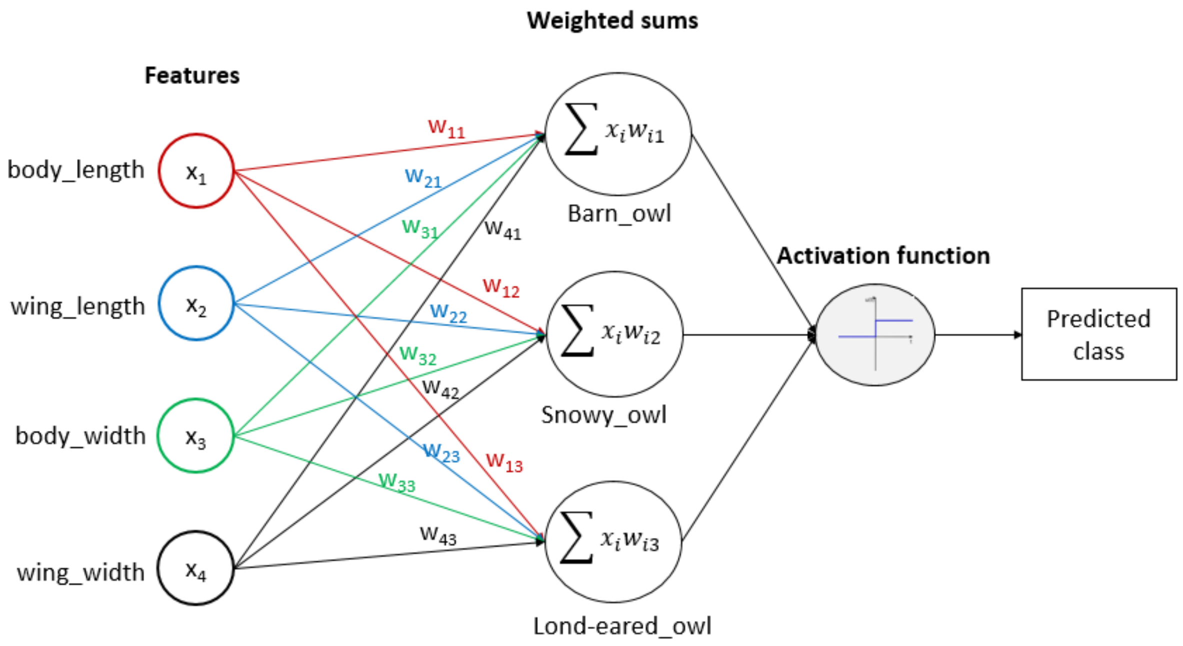

Figure 2. The structure of multiclass perceptron designed for owl dataset

The complete recommendations about individual task can be found in Module Handbook.