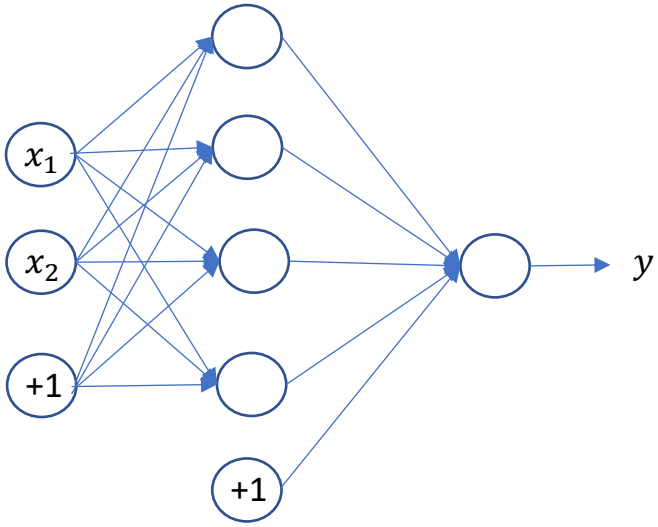

1. A feedforward network consisting of one hidden layer of perceptrons and a linear output neuron receives inputs two-dimensional inputs (x_1, x_2). Train the network to predict the following function:

$$y = sin(\pi x_1)cos(2\pi x_2)$$

where $-1.0 \leq x_1, x_2 \leq +1.0$.

Use data points distributed in an equally spaced 10x10 grid spanning the input space and the following procedures for training and testing:

a) Random subsampling with 70:30 data-split for training and testing.
b) Five-fold cross validation
c) Three-way data split (equal partitioning for train/validation/test)

Repeat each procedure in 10 different experiments and determine the optimal number of hidden neurons in the space of $\{2, 4, 6, 8, 10\}$ and estimate the error of the model.

Use a learning factor $\alpha = 0.05$ and early stopping to cease the learning epochs.

2. The CIFAR-10 dataset consists of 60,000 32x32 colour images from 10 classes, with 6,000 images per class:
   https://www.cs.toronto.edu/~kriz/cifar.html

   There are 50,000 training images and 10,000 test images. Read CIFAR-10 datasets from torchvision.datasets:

   ```
   from torchvision.datasets import CIFAR10
   ```

   Use batch size = 256; and early stopping with patience = 10 for all your experiments.

   a) Build three DNN of different complexity with three hidden layers to classify the images:
      i. A **small network** with 50 neurons in every hidden layer
      ii. A **medium network** with 100 neurons in every hidden layer
      iii. A **large network** with 500 neurons in every layer

      Plot cross-entropies against epochs for train and test datasets. Use the Adams optimizer with default parameter for training and early stopping criterion to terminate. Comment on the overfitting and underfitting of the networks if any.

   b) Using the **large network**, demonstrate how the following methods applied to all the layers could overcome overfitting of the network:
      i. Weight regularization with L2 norm. Use weight decay parameter $\beta = 0.001$.
      ii. Dropouts at a probability $p = 0.1$ to every layer.
      iii. Combining both weight regularization and dropouts from (i) and (ii), respectively.