भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

**Department of Electrical Engineering**
**IIT Hyderabad**
**EE5610-Pattern recognition & Machine Learning**

---

## 1. Data generation

(a) Use Gaussian distribution with appropriate parameters and produce a dataset with four classes and 30 samples per class: the classes must live in the 2D space and be centered on the corners of the unit square (0,0), (0,1) (1,1), (1,0), all with variance 0.3.
(b) Obtain a 2-class train set [X, Y] by having data on opposite corners sharing the same class with labels +1 and -1. (c) Generate a test set $[X_{te}, Y_{te}]$ from the same distribution, starting with 200 samples per class.
(d) Visualize both sets using scatter plot on a 2-D plane.

## 2. kNN classification

The k-Nearest Neighbors algorithm (kNN) assigns to a test point the most frequent label of its k closest examples in the training set.
(a) Write a function kNNClassify to generate predictions $Y_p$ for the 2-class data generated at Section 1. Pick a "reasonable" $k$.
(b) Evaluate the classification performance (prediction error) by comparing the predicted labels $Y_p$ to the true labels $Y_{te}$.
(c) Visualize the obtained results, e.g. by plotting the wrongly classified points using different colors/markers:
(d) Write a function to generate & visualize the decision regions of the 2D plane that are associated with each class, for a given classifier. Overlay the test points using scatter.

## 3. Parameter selection: What is a good value for k?

So far we considered an arbitrary choice for k. You will now use the function hold-outCVkNN for model selection
(a) Perform hold-out cross-validation by setting aside a fraction ($\rho$ of the training set for validation. Note: You may use $\rho = 0.3$, and repeat the procedure 10 times. The hold-out procedure may be quite unstable.

- Use a large range of candidate values for $k$ (e.g. $k = 1, 3, 5..., 21$). Notice odd

numbers are considered to avoid ties.

- Repeat the process for 10 times using a random cross-validation set each time with a $\rho = 0.3$
- Plot the training and validation errors for the different values of $k$.
- How would you now answer the question "what is the best value for k"?

(b) How is the value of k affected by $\rho$ (percentage of points held out) and number of repetitions? What does a large number of repetitions provide?
(c) Apply the model obtained by cross-validation (i.e., best $k$) to the test set and check if there is an improvement on the classification error over the result of Part 2.

**4. size of training data and kNN regression**

(a) Dependence on training size: Evaluate the performance as the size of the training set grows, e.g., n = {50, 100, 300, 500,...}. How would you choose a good range for k as n changes? What can you say about the stability of the solution? Check by repeating the validation multiple times.
(b) Try classifying more difficult datasets, for instance, by increasing the variance or adding noise by randomly flipping the labels on the training set.
(c) Modify the function kNNClassify to handle a) multi-class problems.

**5.Digit classification on MNIST data**

(a) Design a KNN classifier to classify the images in MNIST dataset as one of the 10 digits. The 28x28 images may be flattened to arrive at a 784 dimensional vector.
(b)Empirically determine the most suitable error function, and the corresponding $k$ to maximize the performance on the cross-validation experiments.
(c) Apply these values to evaluate the performance on the test dataset.
(d) Create a confusion matrix to understand the most confused classes (digits).
(e) Suggest alternate ways to improve the performance.

Note: First 4 questions are taken from: `http://lcsl.mit.edu/courses/cbmmss/machine_learning/labs/Lab1.html`