Assignment 3

(*Note:* 1. Each problem needs to be coded in python only.)
(*Note:* 2. Do not use the libraries available for the methods (e.g: classification trees, boosting) in Q1 and Q2. You are expected to code them. However, you can use the libraries and in-built commands in Q3 as mentioned in the question.)

1. (a) Construct a data set consisting of 200 points the same way as in Q1a, assignment 1. (You can use the same code if required.)
   (b) Construct the binary tree using *classification tree* method, solving for the three questions (i.e., which axis, where in the axis, and what classification). Use misclassification rule as the loss function.
   (c) Grow the tree to a maximum depth of 3. Recall that the depth of the tree is the maximum distance of a node from the root node. The branch can be stopped either if the misclassification is zero, or if the number of points in the rectangle is at most 10.
   (d) **Plot** the classifying rectangles along with the scatterplot of the generated features. Represent the features having different labels with different colors. **Print** the training error.

2. Generate 500 data points $X_1, X_2, \ldots, X_{500} \in \mathbb{R}^{10}$ such that they are i.i.d. $\sim \mathcal{N}(0, I_{10})$. Decide Y_i as follows:

$$Y_i = \begin{cases} +1 & \text{if } ||X_i||^2 \geq 9.34, \\ -1 & \text{otherwise.} \end{cases}$$

   Use a two-terminal classification tree to classify the data, and apply *boosting* on top of the classification tree. Compute the error each time after applying the boosting algorithm. **Plot** the training error as a function of the number of iterations the boosting algorithm was applied.

3. Consider the sales of car seats given in *Carseats.csv* file. The first column *Sales* is the dependent variable, and the other columns represent the independent variables. Convert the words in the data as follows:

$$\text{Good} \to 1,\ \text{Medium} \to 0,\ \text{Bad} \to (-1),\ \text{Yes} \to 1,\ \text{No} \to 0.$$

   (a) Split the data into training and testing data. The training data must contain 80% of the randomly selected data points, and the remaining data needs to be the test data.

(b) Construct a binary tree using *regression tree* method. Use the command *DecisionTreeRegressor* from *sklearn.tree*. Do not use maximum depth. Split the tree until there are 10 points in each node. Do not use pruning. Fit the tree in the training data, and then compute the test error in the test data. **Print** the constructed tree using *plot_tree* from *sklearn.tree*, and also **print** the test error.

(c) Perform *cost-complexity pruning* on the tree constructed in part (b). Choose the required set of α's using *cost_complexity_pruning_path* command on the classifier. Pick the value of α that gives the least test error, and then compute the test error in the test data. **Print** the tree constructed on the test data, and also **print** the test error.

(d) Build a *random forest* using the command *RandomForestRegressor* from *sklearn.ensemble*. Choose one-fourth of the training data at random, and choose four of the ten available parameters in the data set. **Print** the test error.