Grace period: The project can be submitted until 11:59 PM of the day of the deadline with 20% penalty. Any change in the project after the deadline is considered late submission. One second late is late. The project is graded based on when it was submitted, not when it was finished. Homework late days cannot be used for the project.

The goal of this assignment is to use sklearn to train Perceptrons, Decision Trees, and Feedforward Neural networks. The best resource for learning sklearn is its documentation found at `https://scikit-learn.org/stable/`. You can to use Jupyter Notebooks and run your code and submit the notebook with the results.

1. **Perceptron Learning for the Iris Dataset**

   (a) Download the banknote authentication dataset from UCI Machine Learning Repository:

   `https://archive.ics.uci.edu/dataset/267/banknote+authentication`

   (b) Use the first 80% instances of the 0 class and the first 80% instances of the 1 class as your training set and the rest as your test set. (15 pts)

   (c) Use sklearn's linear model to train a Perceptron on your training data. Set shuffle = True and random_state = 42. If you are not sure how to set other hyperparameters of the algorithm, choose sklearn's default values.[1] If you feel you can have better results by changing the hyperparameters, you are welcome to change them and explain if they improved the results.[2] (15 pts)

   (d) Test it on your test data. (5 pts)

   (e) Report your train and test error. (5 pts)

2. **Decision Trees as Interpretable Models**

   (a) Download the car evaluation dataset from UCI Machine Learning Repository:

   `https://archive.ics.uci.edu/dataset/19/car+evaluation`.

   (b) Use the first 80% data points in each class (unacceptable, acceptable, good, very good) as training data and the rest as test data.

   (c) Build a decision tree on the training set and plot it. Use gini index. If you are unsure about other hyperparameters of the algorithm, use sklearn's default values.[3] (15 pts)

   (d) Convert the decision rules into a set of IF-THEN rules.[4] (10 pts)

[1] `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Perceptron.html#sklearn.linear_model.Perceptron`

[2] This dataset is not linearly separable, but it is close to linearly separable, so very good train and test results are expected. Because it is not separable, it will terminate at max-iter number of iterations, so you can change that number from the default value of 1000 to 50, 100, 500, 2000, 5000, 10000 and see if your results significantly change

[3] `https://scikit-learn.org/stable/modules/tree.html#decision-trees`

[4] You can use the code in `https://www.kdnuggets.com/2017/05/simplifying-decision-tree-interpretation-decision-rules-python.html` or use sklearn itself.

(e) Test the decision tree on your test set and report your test error. (5 pts)

3. **Feedforward Neural Networks for Regression**

   (a) Download the abalone dataset from UCI Machine Learning Repository:
   `https://archive.ics.uci.edu/dataset/1/abalone`

   (b) Select the first 3200 data points as the training set and the rest as the test set. The data has eight features. All features except sex are numeric. For simplicity, convert M in sex into 0, F into 1, and I into 2. (10 pts)

   (c) Use sklearn's neural network implementation[5] to train a Multilayer Perceptron that predicts the age of an abalone. Use a single hidden layer and one output layer. You are responsible to determine other architectural parameters of the network, including the number of neurons in the hidden and output layers, method of optimization, type of activation functions, and the L2 "regularization" parameter etc. Research what this means. You should determine the design parameters via trial and error, by testing your trained network on the test set and choosing the architecture that yields the smallest mean squared test error.[6] For this part, set early-stopping=False. (25 pts)

   Note: there are a lot of design parameters in a neural network. If you are not sure how they work, just set them as the default of sklearn, but if you use them masterfully, you can have better models.

4. **Extra Credit: Feedforward Neural Networks for Regression with Early Stopping** (20 pts):

   (a) Use the design parameters that you chose in the first part and train a neural network, but this time set early-stopping=True. Research what early stopping is, and compare the performance of your network on the test set with the previous network. You can leave the validation-fraction as the default (0.1) or change it to see whether you can obtain a better model.

   Note: To receive all the extra credit, you must get a performance on the test set that is better than the performance you see in part 3. This must be achieved by both early stopping and tweaking the design parameters of the network, rather than just using the default design parameters.

Note: You must submit Extra Credit along with your main project. If you submit Extra Credit Late, your whole project will be late.

[5] `https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html#sklearn.neural_network.MLPRegressor`

[6] Mean squared error is the residual sum of squares divided by the number of data points