Machine Learning
Homework 1
(due Noon Jan 6)

Instructions

1. The deadline is a hard one. The form upload will close at sharp Noon Jan 6. There will be no extensions.
2. Total marks = 6
3. You have to type the assignment using a word processing engine, create a pdf and upload on the form. Please note that only pdf files will be accepted.
4. Name the submission as {branch}_{roll_number}_{name}.pdf
5. All code/Jupyter notebooks must be put up as **secret gists** and linked in the created pdf. Again, only secret gists. Not the public ones.
6. Any instances of cheating/plagiarism will not be tolerated at all.
7. Cite all the pertinent references in IEEE format.
8. The least count of grading would be 0.5 marks.
9. Some suggestions for plotting - WolframAlpha, Academo.Org, Geogebra, Matploltib, GNUplot, Matlab, Octave
10. You can find the course VM on the course web page. The root password is: 1234

1. Write a Jupyter notebook to create a decision tree from scratch using the CART algorithm. The code should be written in native Python and not use existing libraries. The code should work for regression and classification tasks. As a hint: you may want to use dictionaries to encode the nested relationship amongst the different nodes, eg. tree = {"feature1" :{"val1": ..., "val2", ...}}.
2. Show the usage of your decision tree on the IRIS dataset. The first 70% of

the data should be used for training purposes and the remaining 30% for test purposes. Show the accuracy of the decision tree you implemented on the test dataset.

3. Show the usage of your decision tree for the real estate price prediction regression problem: https://archive.ics.uci.edu/ml/datasets/Real+estate+valuation+data+set

4. Compare the performance of your model with the decision tree module from scikit learn.

5. Use dtreeviz in conjunction with your library to show the learnt decision trees for the above two problems.

6. For the IRIS dataset classification problem, consider three variants of the decision tree algorithm. In the best case, we do an exhaustive search over all possible tree orders and choose the one which gives us the best accuracy on the train set. We use this model to predict for the test set. The second variant that we build gives us the worst performing model from the exhaustive enumeration. Compare the performance of the best order with the greedy order and with worst order.

7. Create some fake data to do some experiments on the runtime complexity of your decision tree algorithm. Create a dataset with N samples and M binary features. Vary M and N to plot the time taken for: 1) learning the tree, 2) predicting for test data. How do these results compare with theoretical time complexity for decision tree creation and prediction.

Some useful references for the homework:

1. scikit learn page on decision trees: https://scikit-learn.org/stable/modules/tree.html