Machine Learning
Homework 1 : Decision Trees and Ensemble Methods
(due Noon Jan 15)

Instructions

1. In case you are unfamiliar with the Python data ecosystem (NumPy, Pandas), you are recommended to study the first four chapters of the Python data science handbook. A doubt clearing session would be organised in case you have any difficulties in the data science ecosystem.
2. The deadline for full score is Noon Jan 15. You can get 50% credit for late submission (Noon Jan 16).
3. Total marks = 25
4. 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.
5. All code/Jupyter notebooks must be put up as **secret gists** and linked in the created pdf submission. Again, only secret gists. Not 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.

1. (a) For the following dataset shown in Figure 1(same as the one we used in the class), now let us assume that date is also an input feature. Which attribute will be chosen as the root node for the decision tree? Do you think choosing this attribute is a good choice? Explain your answer. **[1 mark]**

   (b) Handling missing values in decision trees: Assume that the Outlook of D3 is missing. Could you still learn a decision tree from this example. The set of features of interest are: Outlook, Temperature, Humidity and Wind. Clarifiying again - Day is **not** to be used as a feature in this sub part of the question. Refer to Tom Mitchell Chapter 3.7.4 last paragraph and show the learnt decision tree despite the missing Outlook values from D3. **[1 mark]**

2. (a) 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', ...}}. **[4 marks]**

   (b) 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. **[1 mark]**

   (c) Use 5 fold cross-validation on the dataset. Using nested cross-validation find the optimum depth of the tree. **[2 marks]**

3. Show the usage of your decision tree for the real estate price prediction regression problem. **[1 mark]**
4. Compare the performance of your model with the decision tree module from scikit learn. **[1 mark]**
5. Use dtreeviz in conjunction with your library to show the learnt decision trees for the above two problems. What can you infer from this visualisation? **[2 marks]**

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Figure 1: PlayTennis Dataset

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. **[3 marks]**

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. **[2 marks]**

8. (a) Extend the decision tree you created in Q2 to now implement random forest. **[1 mark]**
   (b) Now use your random forest implementation on the IRIS dataset. Like before, the first 70% of the data should be used for trainining and 30% for test purposes. Compare the accuracy v/s decision tree for the following parameters: i) # of estimators = 20, ii) # feature choices to use = $\sqrt{(n)}$ where n indicates the total number of features available. **[1 mark]**
   (c) Now use 5-fold cross-validation on the dataset. Using nested cross-validation find the optimum number of estimators in the set of [1, 2, 5, 10, 50, 100] estimators. **[1 mark]**

9. Submit your score on Kaggle for the blue book for bulldozers competition using a) decision tree and b) random forests. **[2 marks]**

10. (a) Re-encode the IRIS dataset class as 'setosa' and 'not-setosa'. Now apply ADABoost (on depth-1 decision trees as weak learner) while considering only 'sepal length' and 'petal width' as the features. Train on 100% of the data and create a Matplotlib animation for the first 5 iterations of ADAboost. The title of the plot should show the iteration number and the accuracy on the train set. The plot should color the 'setosa' and 'not-setosa' points differently and their marker size should correspond to their weights as per the ADABoost algortihm. **[1 mark]**
    (b) Now, add some 'noise' to the above dataset, i.e. add some 'setosa' points to the vicinity of the 'not-setosa' points and vice-versa. A human should be able to recorgnise these as outliers. Run the animation on this dataset and comment on the behaviour of ADABoost. **[1 mark]**

Some useful references for the homework:

1. Scikit-learn page on decision trees
2. Scikit-learn page on ensemble methods