Homework assignment 3:
Decision trees and ROC curves

Objective: The objective of this exercise is to become familiar with fitting decision trees and making ROC curves in MATLAB.

Material: Pang-Ning Tan, Michael Steinbach, and Vipin Kumar, *Introduction to Data Mining*, section 4.1-4.6, as well as the included article on ROC curves.

Important: When handing in your homework assignment, please follow the instructions given in the assignment and on Blackboard. **Failure to do so will result in a lower grade.** Create a folder that contains the following:

- The answers to the questions and the plots/graphics in a structured PDF (not in your code!)
- A subfolder "data" containing the data file (e.g. "wine.mat")
- A subfolder "packages" that contains all external packages we provided
- The Matlab script file(s) with your code; use relative paths to import the data and packages, so we can run the code without changes

Put your name and that of your partner (if any) in the PDF and the scripts. Then zip (or tar) the folder mentioned above and rename the zip file to <HOMEWORK-NR_LANGUAGE-USED_NAME01_NAME02>. For example "HW03_Matlab_WoutMegchelenbrink_ElenaSokolova".

In addition, please:

- Write all your answers and any upload comments in English;
- If you are asked to make a plot, put the resulting plot in your PDF file;
- When working in pairs, put the name of your partner in the PDF answer file as well as in the upload comments; only one of you should submit the assignment.

3.1 Decision trees

In this part of the exercise we will fit decision trees using the MATLAB function `classregtree`. As a splitting criterion, the function uses one of the following two impurity measures:

$$\begin{aligned} \text{gdi}(t) &= 1 - \textstyle\sum_{i=1}^{C} p(i|t)^2 && \text{equivalent to Gini}(t) \\ \text{deviance}(t) &= -2 \textstyle\sum_{i=1}^{C} p(i|t) \log p(i|t) && \text{equivalent to Entropy}(t) \end{aligned}$$

We will analyze the wine data we have used previously.

The wine data set has the following attributes, all of which are continuous:

Wine data set

#	Attribute	Unit
1	Fixed acidity (tartaric)	g/dm^3
2	Volatile acidity (acetic)	g/dm^3
3	Citric acid	g/dm^3
4	Residual sugar	g/dm^3
5	Chlorides	g/dm^3
6	Free sulfur dioxide	mg/dm^3
7	Total sulfur dioxide	mg/dm^3
8	Density	g/cm^3
9	pH	pH
10	Sulphates	g/dm^3
11	Alcohol	% vol.

3.1.1 Load the wine data set `Data/wine.mat` into MATLAB. This contains the same data as used in the earlier assignment, but with outliers and the 12th attribute already removed.

Hints:

- The attributes are stored in matrix `X`, the class in vector `y`. Attribute names and class names are stored in the attributeNames and classNames variables.

3.1.2 (1.5 points) Fit a decision tree to the wine data in order to estimate if the wine is red or white. Use the Gini (gdi) splitting criterion. Use the following parameter values: `'prune'` equal to `'on'`, `'mergeleaves'` equal to `'off'` and `'minparent'` equal to 100. Explain what happens when you change the values of the parameter `'minparent'`. Visualize the decision tree, with the attribute names and class names shown in the tree plot.

Hints:

- Make sure you fit a classification tree, not a regression tree.

3.1.3 (1 point) Show that a wine with the following attribute values would be classified as white by the tree fitted in 3.1.2.

#	Attribute	Value
1	Fixed acidity (tartaric)	6.9 g/dm^3
2	Volatile acidity (acetic)	1.09 g/dm^3
3	Citric acid	0.06 g/dm^3
4	Residual sugar	2.1 g/dm^3
5	Chlorides	0.0061 g/dm^3
6	Free sulfur dioxide	12 mg/dm^3
7	Total sulfur dioxide	31 mg/dm^3
8	Density	0.99 g/cm^3
9	pH	3.5
10	Sulphates	0.64 g/dm^3
11	Alcohol	12 %

Which of the 11 attributes are actually used in classifying this particular wine?

Hints:

- If you don't know how to classify input values with a tree, see the help entry of the `classregtree` function.

3.1.4 (1 point) Classify all the wines in the wine data set. What percentage of the wine data is classified correctly by the tree?

Hints:

- To compare two cell arrays of strings, you can use the `strcmp` function.

3.2 Decision tree pruning using cross-validation

In this exercise we will use cross-validation to prune a decision tree. When applying cross-validation the observed data is split into training and test sets, i.e., `X_train`, `y_train` and `X_test` and `y_test`. We train the model on the training data and evaluate the performance of the trained model on the test data.

3.2.1 (2 points) Load the wine data set `Data/wine.mat` into MATLAB. Divide the data into a training and a test data set. Fit a decision tree to the training data using the Gini (`gdi`) splitting criterion and `minparent=10`.

Now, we want to choose the optimal level of pruning using cross-validation. For different levels of pruning from 0 to 10, compute the classification error on the training and test set. Plot the training and test classification error as a function of the pruning level.

Hints:

- Take a look at the function `cvpartition` and see how it can be used to partition the data into a training and a test set (holdout validation). Note that this function can also be used to partition data for K-fold cross-validation. Note also how the function can ensure that both training and test sets have roughly the same class proportions.

What appears to be the optimal pruning level? Do you get the same result when you run your code again, generating a new random split between training and test data?

3.2.2 (1.5 points) Repeat the exercise above, using 10-fold cross-validation. To do this, you must divide the data set into 10 random training and test folds. For each fold, fit a decision tree on the training set and evaluate its performance on the test set. Finally, compute the average classification error across the 10 cross-validation folds and plot it as a function of the pruning level.

Hints:

- As before, `cvpartition` can be used to partition the data into the 10 training and test partitions.

What appears to be the optimal pruning level? Do you get the same result when you run your code again, generating a new random split between training and test data? How about 100-fold cross-validation?

3.3 ROC curves, AUC scores, and the sign test

In this exercise we will use ROC curves and the sign test to compare classifiers. Study the lecture slides and the paper 'ROC Graphs: Notes and Practical Considerations for Researchers' by Tom Fawcett included with the homework assignment (ROC101.pdf). It describes all you need to know (and more) about ROC curves. The method explained for computing the area under the curve is unnecessarily complicated. A simpler formula is:

$$\mathrm{AUC} = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} \mathbf{1}_{p_i > p_j} .$$

Here i runs over all m data points with true label 1, and j runs over all n data points with true label 0; p_i and p_j denote the probability score assigned by the classifier to data point i and j, respectively. $\mathbf{1}$ is the indicator function: it outputs 1 if the condition (here $p_i > p_j$) is satisfied and 0 otherwise.

3.3.1 Load the file `Data/classprobs.xls`. The first column gives the true class label (either 0 or 1). The second and third column give the probabilistic scores for two different classifiers. The higher this probability, the more certain the classifier is that the example belongs to class 1 (instead of class 0).

3.3.2 (1 point) Calculate the ROC curves for the classifiers and plot them. Interpret the obtained results. Do both classifiers perform better than random guessing?

Hints:

- The function `roc` can be used for computing the ROC.

3.3.3 (1 point) Compute the AUC scores (area under the curve) of both classifiers using the formula given above. Do the AUC scores indicate that the classifiers are performing better than this baseline?

3.3.4 (1 point) Using a threshold of 0.5, translate the probability scores to predicted class labels, and compute the accuracy for each of the classifiers.