CITS5508 Machine Learning
Semester 1, 2022

Lab Sheet 3
(Not assessed)

1 Outline

In this labsheet, you are asked to train a few decision tree classifiers on the *Breast cancer wisconsin (diagnostic) dataset* available on Scikit-Learn and compare their performances. This labsheet is a good exercise to practise what you learn from the lecture notes and Chapter 6 of the textbook.

2 Tasks

Your tasks for this labsheet are listed below:

1. Full description about the *Breast cancer wisconsin (diagnostic) dataset* can be found in **Section 7.1.7** of the following web page:

   https://scikit-learn.org/stable/datasets/toy_dataset.html#breast-cancer-wisconsin-diagnostic-dataset

   There are two classes in the dataset:

   - *malignant* (212 instances, class value 0) and
   - *benign* (357 instances, class value 1).

   Follow the example code given on the web page:

   https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_breast_cancer.html#sklearn.datasets.load_breast_cancer

   to read in the dataset and separate it into a feature matrix `X` and a class vector `y`. Your feature matrix should have 569 (rows) $\times$ 30 (columns) and your class vector should have 569 elements.

2. Investigate whether some features can be dropped through some suitable visualisation of the data. If two features have a linear relationship, then it will be sufficient to keep only one of these features. If you use a different colour in the visualisation to show data points coming from each class, you will find that majorities of the points from the two classes are well separated. So we can assess that the classification task later on should not be too difficult.

   Determine from your visualisation what features can be dropped and write Python code to drop them from your feature matrix `X`.

   Split `X` and `y` into a training set and a validation set using the split ratio 85/15 and `random_state=123`.

3. Investigate a few decision tree classifiers with different hyperparameter values as follows:

   - Train a decision tree classifier using the default values for all the hyperparameters.
   - Use the trained classifier to perform predictions on the training set and the validation set. Compare the accuracy scores for the two sets.

- Do you think this classifier has an overfitting issue? Why?
- Follow the example code for Chapter 6 from the GitHub page for the textbook to display the decision tree built from the training process (like the one shown in Figure 6.1 of the textbook for the *iris* dataset). You will need to restrict the depth of the tree for the display (e.g., by setting `max_depth=3`); otherwise, the diagram will be too large. Study the diagram to see if it can help you to confirm whether the classifier has an overfitting issue.
- Construct a second decision tree classifier by setting the hyperparameter `max_depth=3`. Repeat all the steps above for this new classifier.

  Compare the accuracy scores for the training set and validation set with those from the previous classifier (note that your classification results may differ for different runs of the code).
- Construct a third decision tree classifier by setting the hyperparameter `min_samples_split=5`. Repeat all the steps above for this new classifier.

  Compare the accuracy scores for the training set and validation set with those from all the previous classifiers.
- Construct a fourth decision tree classifier by setting the hyperparameter `min_samples_leaf=5`. Repeat all the steps above for this new classifier.

  Compare the accuracy scores for the training set and validation set with those from all the previous classifiers.

4. Display the confusion matrices for the training set and validation set from the last decision tree classifier above.