UiT
THE ARCTIC UNIVERSITY OF NORWAY

FYS-2021 - Exercise Set 6

Decision Trees

Department of Physics and Technology
Faculty of Science and Technology

From book:

9.1, 9.6

Problem 1

In this problem, you will implement a Classification tree from scratch. For simplicity, you can assume continuous features and a binary split. We suggest using an object-oriented approach, where each (sub)tree is represented as an object:

```
class Tree:
  method init(...):
    // Initialize variables
  end

  // Implements the "GenerateTree"-function from Fig. 9.3 in the book.
  method fit(data, labels, ...):
    if impurity(labels) < minimum impurity: // Eq. 9.3 in the book
      // Declare a leaf node
      return
    end

    // Find the best split.
    split_index, threshold = find_best_split(data, labels)

    // Create branches
    branch1 = Tree(...)
    branch2 = Tree(...)

    // Generate sub-trees
    branch1.fit(data and labels assigned to branch 1)
    branch2.fit(data and labels assigned to branch 2)
  end

  method predict(data):
    for each row in data:
      // Find leaf corresponding to row
    end
  end
```

(1a) Implement your own version of the Classification Tree algorithm. Remember to include a parameter specifying the maximum depth of the tree, to prevent overfitting.

(1b) Test your implementation on the datasets in `blobs.csv` and `flame.csv`. Plot the data, and the regions found by the tree.

(1c) Draw or plot a diagram similar to Figure 9.6 in the book, illustrating the rules learned by the tree.

The file `tictac.csv` specifies the optimal move for a specified tic-tac-toe board layout. Each row corresponds to a different board. The first element is the label, which indicates the best move to make ($0 =$ top left, ..., $8 =$ bottom right). The rest of the row contains the (flattened) current board layout, with $1 = \times$, $-1 = \bigcirc$, and $0 =$ blank.

(1d) Split the data into training- and test-sets. Train your classification tree using the training set, and test it using the testing set. How is the generalization performance?

(1e) **Bonus:** Use your classification tree to create a program where the user can play tic-tac-toe against the computer.

Problem 2

(2a) Implement your own version of the Regression Tree algorithm. If you have implemented the Classification Tree, all you need to change is the impurity measure, and the creation of leaf-nodes. Remember to include a parameter specifying the maximum depth of the tree, to prevent overfitting.

(2b) Test your implementation using the dataset in `global-temperatures.csv`. Plot the predicted line together with the data for different values of the max-depth parameter. What do you observe? Plot/sketch a diagram of the resulting tree when the maximum depth equals 3.

(2c) Test your implementation using the dataset in `auto-mpg.csv`. Generate the tree for different maximum depths, and compute R^2 for the predictions made by each of the trees. Plot R^2 as a function of depth. The resulting plot should be monotonically increasing. Why is this the case?