Preparation

Download the dataset from

CTG_data.csv

The description of the original data (I extracted a subset of the columns, LB..tendency, and the NSP target) is in

https://archive.ics.uci.edu/ml/datasets/Cardiotocography

Create a new project in Orange.

Note: whenever you add an item to the report, in the text box label that item with the corresponding Part and Step number and a short title.

Part 1: reading and examining the dataset

1. Read in the CTG classification data file into Orange:
    a. The classification model you will be making will try to predict the "NSP" (normal, suspect, pathological, encoded as 1, 2, and 3, respectively) column from the feature columns
    b. Read the data file into Orange. Make sure the NSP target column is of categorial type and the target for the dataset.
    c. attach a Data Table widget and make sure the data was read in correctly.

Part 2: Tree based methods

2. Add a Test and Score widget, and connect it to the data file. Use 10-fold cross validation.
3. Create a preprocess + classify pipeline with a Tree classifier.
    a. use the preprocessor to normalize the data.
    b. set the Tree classifier to
        i. minimum 2 per leaf
        ii. don't split subsets < 5
        iii. max depth 3
        iv. leave "majority 95%" as is
    c. Connect this to the Test & Score widget as a Learner
4. Add an preprocess (same setting)+AdaBoost pipeline
    a. use 50 trees and a learning rate of 0.25
    b. connect to the Test & Score
5. Add a preprocess (same settings) + Random Forest pipeline
    a. Set the available tree hyperparameters to be the same as the Tree classifier in 2.2, but note that there is no "minimum samples per leaf" setting. Also, turn off "number of attributes considered at each split", and turn on replicable training.

    b. use 50 trees
    c. connect to the Test & Score
6. Set the Test & Score widget to class 3, and add a report for it. Discuss the following two metrics
    a. Do either of the more complex models work significantly better than the single tree if the goal is to have the highest classification accuracy?
    b. What metric would you use if the goal is to be sure to find all the cases of this class (3 == pathological)? Are there significant differences, and if so, which model is best, and is it one of the more complex ones?

Part 3: stacking

7. duplicate the entire workflow
8. Delete all three tree based pipelines
9. Create new pipelines with processor + classifier for each of the following methods, and connect them to Test & Score
    a. SVM (regular, linear kernel, C=1, eps=0.1)
    b. Naive Bayes (set a discretization preprocessor to a histogram with 10 bins, equal width)
    c. Logistic Regression (L2 regularization, C=1)
    d. Tree (min in leaves 2, don't split < 5, max depth 2)
    e. k-NN (10 neighbors, Euclidean distance)
10. Add a "Stacking classifier" and connect it to the Test & Score
    a. duplicate all of the classifiers above, and connect each duplicated pipeline to the Stacking Classifier as Learners, not Aggregate
11. Make sure the Test & Score class is still set to 3, and add a report
    a. compare the performance of the individual classifiers to the stacked combination in terms of general accuracy as represented by F1 and in terms of recall. Are the classifiers always in the same order, or does their relative performance depend on the metric? Is stacking more effective than the individual classifiers in this case? Does the difference seem significant?

Part 5: submitting

12. Save the workflow.
13. Open the report window, confirm that there are 2 test & score performance reports, and save it to html.
14. Upload both workflow and html report to canvas.