CSE455/CSE552 – Machine Learning (Spring 2022)

Homework #2

Hand-in Policy: Via Teams. No late submissions will be accepted.
Collaboration Policy: No collaboration is permitted.
Grading: This homework will be graded on the scale 100.

Description: The aim of this homework is to get you acquainted with implementing a decision tree as discussed in class. Your implementation should be able to run on a data with two types of features (numeric and categorical).

Use the following data for testing your implementation: (ABALONE - https://archive.ics.uci.edu/ml/machine-learning-databases/abalone/. You can use Python libraries to read this data file.

This project is expecting you to write two functions. The first will take the training data including the type of features and returns a decision tree best modeling the classification problem. This should not do any pruning of the tree. An optional part of the homework will require the pruning step. The second function will just apply the decision tree to a given set of test data points.

You should follow the following for this homework:

1. Load the data and perform a quick analysis of what it is and what features it has. You will need to construct a vector indicating the type (values) of each of the features. In this case, you can assume that you have numeric (real or integer) and categorical values.
2. Implement the function “**build_dt(X, y, attribute_types, options)**”.
    a. X: is the matrix of features/attributes for the training data. Each row includes a data sample.
    b. y: The vector containing the class labels for each sample in the rows of X.
    c. attribute_types: The vector containing (1: integer/real) or (2: categorical) indicating the type of each attributes (the columns of X).
    d. options: Any options you might want to pass to your decision tree builder.
    e. Returns a decision tree of the structure of your choice.
3. Implement the function “**predict_dt(dt, X, options)**”.
    a. dt: The decision tree modeled by “build_dt” function.
    b. X: is the matrix of features/attributes for the test data.
    c. Returns a vector for the predicted class labels.
4. Report the performance of your implementation using an appropriate k-fold cross validation using confusion matrices on the given dataset.
5. Implement the pruning strategy discussed in the class. Repeat the steps 4 above. Indicate any assumptions you might have made. Repeat step 4.
6. Implement the function “**build_rdf(X, y, attribute_types, N, options)**”.
    a. X: is the matrix of features/attributes for the training data. Each row includes a data sample.

    b. y: The vector containing the class labels for each sample in the rows of X.
    c. attribute_types: The vector containing (1: integer/real) or (2: categorical) indicating the type of each attributes (the columns of X).
    d. N: The number of trees.
    e. options: Any options you might want to pass to your decision tree builder.
    f. Returns a decision tree of the structure of your choice.

7. Implement the function “**predict_rdf(rdf, X, options)**”.
    a. rdf: The decision tree modeled by “build_rdf” function.
    b. X: is the matrix of features/attributes for the test data.
    c. Returns a vector for the predicted class labels.
8. Report the performance of your implementation using an appropriate k-fold cross validation using confusion matrices on the given dataset.

What to hand in: You are expected to hand in one of the following

- **HW2_lastname_firstname_studentnumber_code.ipynb** (the Python notebook file containing the code and report output).

Your notebook should include something like the following:

Implementation of Decision Tree Modeling Function:

Implementation of Decision Tree Testing Function:

Results of k-fold cross validation:

Implementation of Decision Tree Testing Function with Pruning:

Results of k-fold cross validation:

Implementation of RDF:

Results of k-fold cross validation: