605.649 — Introduction to Machine Learning

Programming Project #3

1 Project Description

The purpose of this assignment is to give you some hands-on experience using learning decision trees for classification and regression. Although we are not going to directly use anything from the module on rule induction; you might want to still examine the rules your trees learned to see if they "make sense." Specifically, you will be implementing a standard univariate (i.e. axis-parallel) decision tree and will compare the performance of the trees when grown to completion on trees that use reduced error pruning.

We need to stress that the number of children of a node in a decision tree is based upon the number of values the feature can take on. For numerical features (see below), we assume binary splits. For categorical (nominal and ordinal) features, the number of splits is based on the number of discrete values. When such features exist, then for classification, we choose the feature that maximizes the gain ratio:

$$gratio(f_i) = \frac{gain(f_i)}{IV(f_i)}$$

where

$$gain(f_i) = \underbrace{\left(-\sum_{\ell=1}^{k} \frac{c_{\pi,\ell}}{|\mathcal{D}_\pi|} \lg \frac{c_{\pi,\ell}}{|\mathcal{D}_\pi|}\right)}_{\mathcal{H}(\mathcal{D}_\pi)} - \underbrace{\left(\sum_{j=1}^{m_i} \frac{|\mathcal{D}_\pi^j|}{|\mathcal{D}_\pi|} \mathcal{H}(\mathcal{D}_\pi^j)\right)}_{E_\pi(f_i)}$$

with $\mathcal{D}_\pi$ being the subset of data in partition π, $c_{\pi,\ell}$ denoting the number of examples in partition π that occur in class ℓ, and $\mathcal{H}$, and m_i being the number of values in feature f_i. In addition,

$$IV(f_i) = -\sum_{j=1}^{m_i} \frac{|\mathcal{D}_\pi^j|}{|\mathcal{D}_\pi|} \lg \frac{|\mathcal{D}_\pi^j|}{|\mathcal{D}_\pi|}$$

For regression trees, we choose the split that minimizes the squared error:

$$Err'_\pi = \frac{1}{|\mathcal{D}_\pi|} \sum_j \sum_t (r^t - \hat{r}^t_{\pi j})^2 b_{\pi j}(\mathbf{x}^t)$$

where r^t is the ground truth response value for $\mathbf{x}^t$, $\hat{r}^t_{\pi j}$ is the predicted response value for $\mathbf{x}^t$ in partition π after following branch j, and

$$b_{\pi j}(\mathbf{x}^t) = \begin{cases} 1 & \text{if } \mathbf{x}^t \in \mathcal{D}_\pi \text{, i.e. } \mathbf{x}^t \text{ reaches node } \pi \text{ and takes branch } j \\ 0 & \text{otherwise.} \end{cases}$$

Now let's talk about the numeric attributes. There are two ways of handling them. The first involves discretizing (binning), similar to what you were doing in earlier assignments. This is *not the preferred approach*, so we ask that you not bin these attributes. Instead, the second and requested approach is to sort the data on the attribute and consider possible binary splits at midpoints between adjacent data points. Note that this could lead to a lot of possible splits. One way to reduce that is to consider midpoints between data where the class changes. For regression, there is no corresponding method, so you should consider splits near the middle of the sorted range and should not attempt to consider all possible splits.

For decision trees, it should not matter whether you have categorical or numeric attributes, but you need to remember to keep track of which is which. In other words, one-hot-coding will not be needed. In addition, as mentioned above, you need to implement that gain ratio criterion for splitting in your classification trees. Make sure to eliminate features that act as unique identifiers of the data points.

2 Data Sets

For this assignment, you will use three classification datasets and three regression data sets that you will download from the UCI Machine Learning Repository or from Canvas, namely:

1. Breast Cancer [Classification]

   `https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29`

   This breast cancer data set was obtained from the University of Wisconsin

2. Car Evaluation [Classification]

   `https://archive.ics.uci.edu/ml/datasets/Car+Evaluation`

   The data is on evaluations of car acceptability based on price, comfort, and technical specifications.

3. Congressional Vote [Classification]

   `https://archive.ics.uci.edu/ml/datasets/Congressional+Voting+Records`

   This data set includes votes for each of the U.S. House of Representatives Congressmen on the 16 key votes identified by the Congressional Quarterly Almanac.

4. Abalone [Regression]

   `https://archive.ics.uci.edu/ml/datasets/Abalone`

   Predicting the age of abalone from physical measurements.

5. Computer Hardware [Regression]

   `https://archive.ics.uci.edu/ml/datasets/Computer+Hardware`

   The estimated relative performance values were estimated by the authors using a linear regression method. The gives you a chance to see how well you can replicate the results with these two models.

6. Forest Fires [Regression]

   `https://archive.ics.uci.edu/ml/datasets/Forest+Fires`

   This is a difficult regression task, where the aim is to predict the burned area of forest fires, in the northeast region of Portugal, by using meteorological and other data .

3 Project Requirements

For this project, the following steps are required:

- Download the six (6) data sets from the UCI Machine Learning repository. You can find this repository at `http://archive.ics.uci.edu/ml/`. The data sets are also available in Canvas. All of the specific URLs are also provided above.
- Implement the decision tree algorithm for classification trees using gain ratio as the splitting criterion.
- Implement the decision tree algorithm for regression trees using mean squared error as the splitting criterion.
- Implement reduced-error pruning to be used as an option with your implementation of both the classification trees and the regression trees.
- Run your decision tree implementation on each of the data sets, comparing both pruned and unpruned versions of the trees. These runs should be done with 5×2 cross-validation so you can compare your results statistically.

- You should pull out 20% of the data to be used for pruning and then use the remaining 80% for cross validation. In other words, during cross validation, train a decision tree as far as you can on the folds. When pruning, use the 20% that was pulled out in other projects for hyperparameter tuning to evaluate the node pruning choices. Once the tree is pruned, use the held-out fold for testing.
- Write a very brief paper that incorporates the following elements, summarizing the results of your experiments. Your paper is required to be at least 5 pages and no more than 10 pages using the JMLR format You can find templates for this format at `http://www.jmlr.org/format/format.html`. The format is also available within Overleaf.
  1. Title and author name
  2. Problem statement, including hypothesis, projecting how you expect each algorithm to perform
  3. Brief description of your experimental approach, including any assumptions made with your algorithms
  4. Presentation of the results of your experiments
  5. A discussion of the behavior of your algorithms, combined with any conclusions you can draw
  6. Conclusion
  7. References (Only required if you use a resource other than the course content.)
- Submit your fully documented code, the outputs from running your programs, and your paper.
- For the video, the following constitute minimal requirements that must be satisfied:
  – The video is to be no longer than 5 minutes long.
  – The video should be provided in mp4 format. Alternatively, it can be uploaded to a streaming service such as YouTube with a link provided.
  – Fast forwarding is permitted through long computational cycles. Fast forwarding is *not permitted* whenever there is a voice-over or when results are being presented.
  – Be sure to provide verbal commentary or explanation on all of the elements you are demonstrating.
  – Provide sample outputs from one test set on one fold for a classification tree and a regression tree.
  – Show a sample classification tree without pruning and with pruning.
  – Show a sample regression tree without pruning and with pruning.
  – Demonstrate the calculation of information gain, gain ratio, and mean squared error.
  – Demonstrate a decision being made to prune a subtree.
  – Demonstrate an example traversing a classification tree and a class label being assigned at the leaf.
  – Demonstrate an example traversing a regression tree and a prediction being made at the leaf.

4 Project Evaluation

Your grade will be broken down as follows:

- Code structure – 10%
- Code documentation/commenting – 10%
- Proper functioning of your code, as illustrated by a 5 minute video – 30%
- Summary paper – 50%