INGI2262 – Machine Learning

Assignment 2

Decision Tree Learning

1 Written assignment

Q 1 Suppose you have the following training set with four boolean attributes x_1, x_2, x_3 and x_4 and a boolean output y.

x_1	x_2	x_3	x_4	y
0	0	0	1	0
0	1	1	1	1
1	0	1	1	1
1	1	1	1	0

Q 1.1 What is the tree learned by ID3 from this training set? You should be able to construct it from your general understanding of this algorithm without going into all the details of computing explicitly every step of this algorithm.

Q 1.2 Is there another binary decision tree which would perfectly classify the same training examples and would not be as deep as the one proposed by ID3? If so, what is this tree and explain why ID3 does not return it. If not so, argue why ID3 necessarily finds the minimum-depth tree on this training set.

Q 2 Suppose you have a training set with 10 positive and 10 negative examples and you need to choose, at the root node, between the two splits [8+,2-] (left) [2+,8-] (right) versus [10+,6-] (left) [0+,4-] (right).

Q 2.1 Evaluate the information gain of both splits and deduce from it which split would chosen by ID3.

Q 2.2 Suppose now that the mistakes on the positive examples are about 10 times as costly as mistakes to the negative. One way of dealing with such a cost imbalance is to replicate the positive examples 10 times each. What would then be the best splits, according to information gain, between [80+,2-] (left) [20+,8-] (right) versus [100+,6-] (left) [0+,4-] (right)?

Q 2.3 What do you conclude from the previous calculations in terms of splitting criteria and class cost imbalance? Do you see a numerically equivalent way to decide which is the best split in question 2.2 *without* replicating the positive examples?

Q 3 What is the total number of decision trees with d attributes, each having k possible values (including syntactically distinct trees possibly describing the same model)? Propose a general expression and a simple demonstration. The previous figure gives the size of the complete search space of the ID3 algorithm. What is the actual number of different trees considered in ID3 (without pruning)?

Q 4 Consider a classification problem based only on 2 continuous attributes (the instance space is the plane $\mathbb{R}^2$). C4.5 incorporates these attributes by defining threshold-based boolean attributes. In the induced tree, each node corresponds to a particular decision boundary splitting the examples into two regions. What are the shape (in the instance space) of the decision boundaries learned by C4.5? Into how many regions is the instance space divided before pruning? Does it depend on the attribute values of the training examples? Does it depend on the number of classes?

2 Mini-project

2.1 Task description

You are a data analyst in charge of predicting the housing values in Boston. The original data is briefly described in the appendix of this document, with 2 important modifications. The original dataset is made of 506 samples on 14 variables. Here, this dataset has been partitioned for you in two files ***BostonHouseTrain.csv*** (400 samples) and ***BostonHouseTest.csv*** (106 samples), used respectively for training and testing your models[1]. The last variable **`medv`** has been discretized as (and replaced by) a **`class`** value. Such **`class`** is either **`low`**, **`average`** or **`high`**. Your main task is to build a model from the training set to predict the **`class`** value on the test samples. Part of the work was done for you in the **`rpart`** package implemented in R.

2.2 Questions

Q 5 Use first the ***playTennis.csv*** toy datafile to check how the **`rpart`** implementation of the CART algorithm works. You are invited to use the **`rpart`** function to build a decision tree and the **`plot`** and **`text`** functions to represent the result graphically. Note that the **`rpart`** implementation include several meta-parameters (also called control parameters) that may strongly influence the result. You are invited to check at least the role of the arguments **`method`**, **`parms`** and **`control`**. For instance, this implementation assumes by default a minimum number of 20 observations to attempt a node split (how is this assumption specified concretely?). This might not be a relevant choice given the actual size of such a small dataset (why?). You are invited to consult the online help about **`rpart`** or the documentation provided in *An Introduction to Recursive Partitioning Using the RPART Routines*[2]. Are you able to reproduce the decision tree presented during the lecture using this toy example? If so, which are the values of the key control parameter(s) you had to choose? If not so, how do you explain the difference?

[1] Those files are available in a zip archive on Moodle.

[2] The document `RPART_intro.pdf` can be downloaded from Moodle.

Q 6 Use the ***BostonHouseTrain.csv*** training file and the **`rpart`** CART implementation to build a decision tree **without pruning**[3] to predict whether the house value is *low*, *high* or *average*. Report the number of nodes of such a tree learned without pre- or post-pruning. Test it both to classify the training data itself and the test data (***BostonHouseTest.csv***).
We recommend you to check the examples in the **`rpart.predict`** documentation to discover a simple way to build a **`table`** forming a confusion matrix between predicted class labels and actual class labels of test examples.
Is the training set accuracy guaranteed to be 100 % with CART without pruning? Do you observe this and why? Is the test accuracy worse or better than the training accuracy? Why?

Q 7 Use 25 % of the training data to build a decision tree with CART **without pruning** and all the test data to further assess the performance. The R function **`sample`** may be very useful to extract a random subset of examples from a training set. Rerun this experiment 5 times and notice the impact of different random selection of the training data. Report the number of nodes and classification accuracy (both on the 25% training and the whole test set) of these trees over 5 distinct runs.

Q 8 Measure the impact of training set size on the accuracy and the size of the learned tree (**without pruning**; use the whole test set for testing). Consider the following fractions of the total training set: 5%, 10%, 20%, 50%, 99%. Because of the high variance due to random splits, repeat the experience with 10 independent random samples for each training set size[4].
We recommend you to generate a **`data.frame`** with 4 columns corresponding respectively to the *random sample index*, *number of training instances*, *number of nodes of the learned tree*, *test set accuracy*.
Turn in a plot showing how the number of nodes varies with the training set size. The R **`boxplot`** function is a convenient tool for this task. Do you observe the expected evolution of the tree sizes as function of the number of training examples[5]?
Turn in a plot showing how accuracy on the test set varies with the training set size. This plot, called a *learning curve*, is fundamental for estimating whether a program learns, that is actually improves with experience. The R **`boxplot`** function could also be useful here.
Due to the random selection of training examples, results may vary even for a fixed number of training examples. How do you expect this variance to be a function of the training size? Do you confirm your expectation on the plot?

Q 9 Apply the CART algorithm to build a decision tree on the same data while studying the effect of pruning the tree. Recall that CART includes a pre-pruning mechanism, preventing some nodes to be further split while growing the tree, and a post-pruning mechanism to actually prune a tree after it was grown. Measure the performance of

[3] Among possibly other parameters, check the role of the **`cp`** argument to **`rpart.control`**.

[4] Without much experience in R you are likely to program this with 2 embedded **`for`** loops. This is functionally OK but **`for`** loops should ideally be avoided for speed concerns. More advanced R users would rely on an appropriate combination of **`expand.grid`** (to generate a bi-dimensional grid of experimental conditions) and **`apply`** (to apply a learning function in each element of the grid).

[5] To match your expectation make sure trees are learned without pre- or post-pruning.

CART with pruning, under the same conditions as in question 8. Use the same training set sizes (5%, 10%, 20%, 50%, 99%), but this time use some pruning mechanism(s). Report comparative plots of the learning curves and tree sizes of CART with and without pruning.

Q 10 Which are the 3 most important input features for the prediction task at hand? Build a decision tree on the full training set with appropriate control parameters based on your answers to the former questions. Analyze the structure of such a tree to support your claim about the most important features for this task.

Q 11 We will study here the effect of bagging decision trees. Consider the original training set (400 examples) and draw bootstrap samples from this set. A bootstrap sample has the same size as the original set but, as it is drawn with replacement, it can contain replicated examples while some original examples are no longer included. Build a decision tree from each bootstrap sample, considered as the training set for that tree. To get a simple model from each bootstrap sample, you are invited to use some control parameter(s) to make sure that the maximal depth of each tree is at most 2. You should end up with as many models, *i.e.* as many decision trees, as bootstrap samples you have considered. You now have an ensemble of models that you can use to classify the test set. For each test example, the predicted class is obtained by a majority vote among the specific predictions of all your models. If bagging works, one would expect to get a better test accuracy (as compared to using a single decision trees). Do you observe such benefit? Report an experimental curve to support your claim? How many bootstrap rounds (= how many trees) do you recommend to use? Why?

Submit a **PDF** report with answers to the questions of this assignment. Include a brief description of your use of R routines and decision trees classifiers. Add any comment you consider relevant about the use of this software on the task at hand.

Appendix

Boston {MASS} R Documentation

Housing Values in Suburbs of Boston

Description

The `Boston` data frame has 506 rows and 14 columns.

Usage

```
Boston
```

Format

This data frame contains the following columns:

`crim`
: per capita crime rate by town.

`zn`
: proportion of residential land zoned for lots over 25,000 sq.ft.

`indus`
: proportion of non-retail business acres per town.

`chas`
: Charles River dummy variable (= 1 if tract bounds river; 0 otherwise).

`nox`
: nitrogen oxides concentration (parts per 10 million).

`rm`
: average number of rooms per dwelling.

`age`
: proportion of owner-occupied units built prior to 1940.

`dis`
: weighted mean of distances to five Boston employment centres.

`rad`
: index of accessibility to radial highways.

`tax`
: full-value property-tax rate per \$10,000.

`ptratio`
: pupil-teacher ratio by town.

`black`
: *1000(Bk - 0.63)^2* where *Bk* is the proportion of blacks by town.

`lstat`
: lower status of the population (percent).

`medv`
: median value of owner-occupied homes in \$1000s.