CIS 471 Project Report

Building a Decision Tree Learning Agent

J.Xie

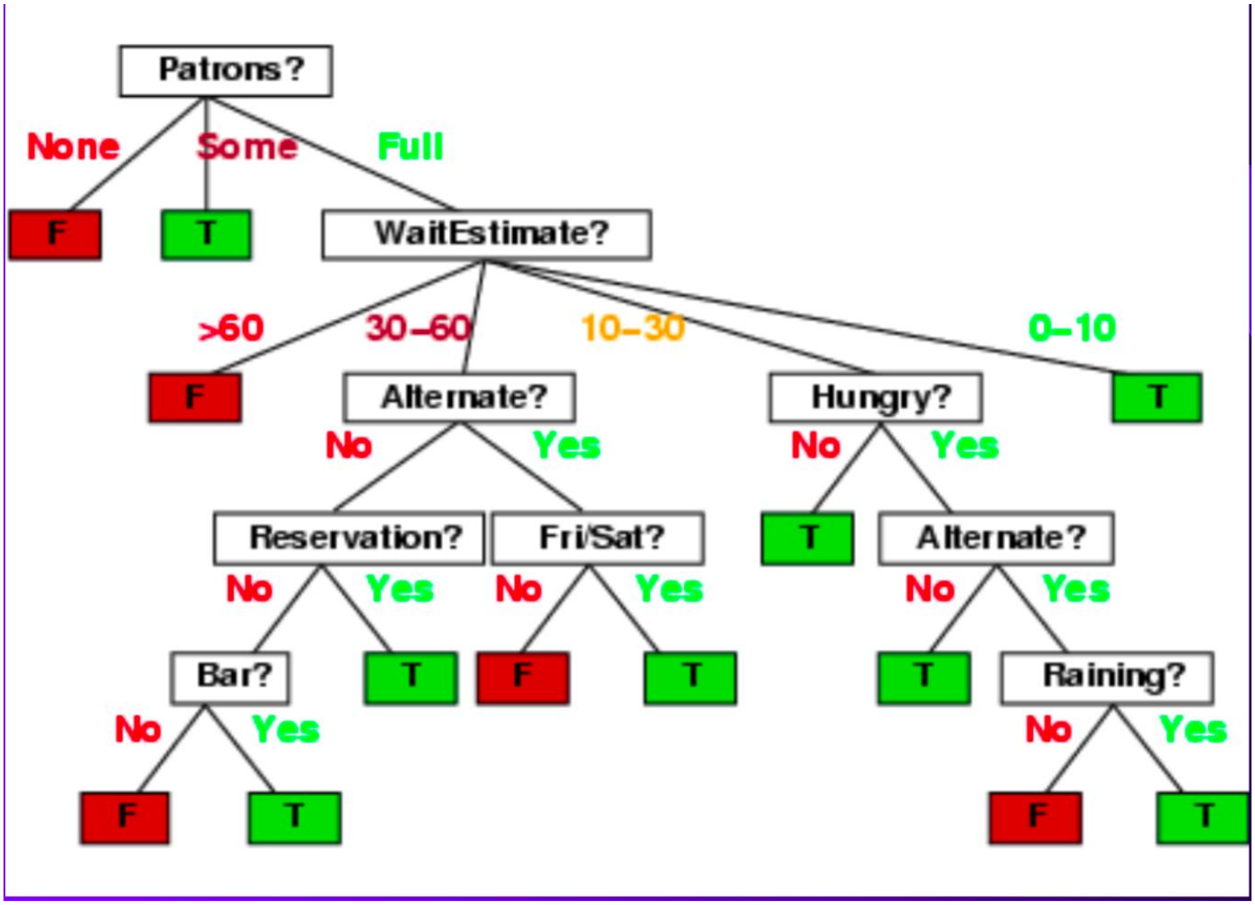
Today, researches in Artificial Intelligence, or AI, have been booming and people are trying their best to empower computers abilities to learn and think as humans. Recently, many algorithms have been developed in the machine learning, or ML, area and new applications based on the algorithms emerged. For example, the rapid progress of speech recognition algorithms has led to the prosperity of the smart speaker market such as Amazon’s Echo-dot, Google’s Home, and Apple’s HomePod. As a result, it would help me gain more practical know-how to implement an algorithm in this area for the course project.

Generally speaking, the machine learning area can be divided into symbolic and non-symbolic machine learning (Haugeland, 1985). The symbolic approach, as its name saying, uses human-readable and symbolic data to teach an AI, and eventually the AI would use logic and the training data to build a model where it could look up answers from. On the other hand, the non-symbolic approach uses raw data to train an AI so that the AI can analyze and build its own knowledgebase which we humans might not be able to understand. My personal project for this class would be based on symbolic learning since it seems more reasonable to do so for a two-week project.

During the class, I have learned about how the decision tree learning works and why it is useful in big data and data mining (Rokach, 2008). The general idea of decision trees can be demonstrated with an example from one of the lecture slides (Dou, 2018). Suppose you want to determine whether a person want to wait in the line for a resturant or not. You observed twelve people and kept track of a set of properity of them. Suppose the data you collected is in the table below.

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

From the table above, we can build one possible decision tree as the following one so that we could use it to predict future customers' decisions. Basically, it is a model for classification.



This is just one of billions decision trees one can generate with the given data and is not the most efficient one in terms of classification. Thus, an efficient method to find the best decision tree is needed.

For this course project, I am going to implement a notable decision-tree algorithm called Iterative-Dichotomiser-3. Such algorithm entitles the decision tree the ability to learn by splitting the training data into subsets according to the attribute value test, or information gain test. The program will keep splitting the source into subsets until it reaches some level that each leaf is pure, or definite. Such process is called recursive partitioning (Quinlan, 1986). If I have more time available after this implementation, I would spend some effort on its successor algorithm called C4.5.

The outline for such decision tree learning agent:

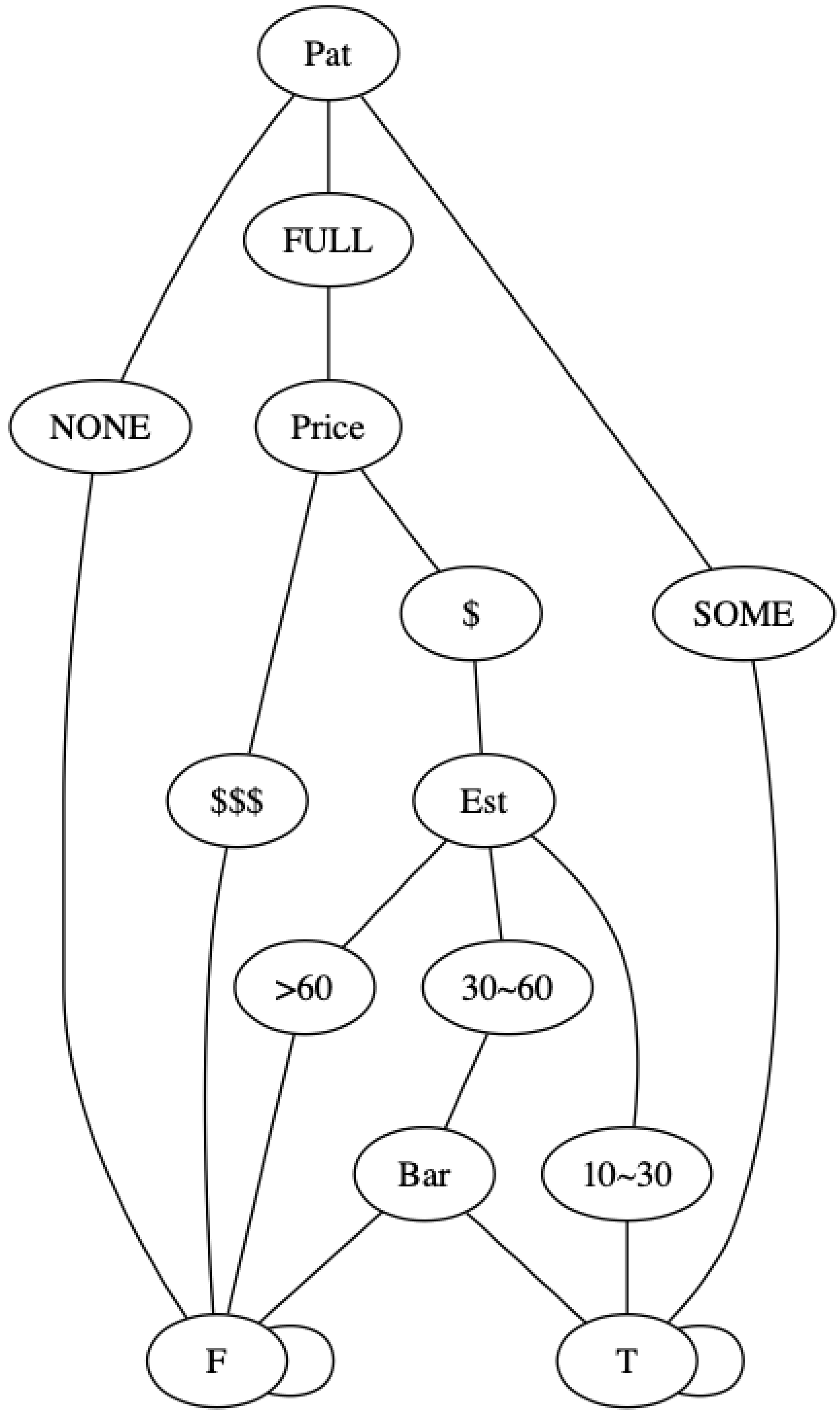
1. Input a set of training data, each entity should look like a tuple as this one ({Attribute x: value1, Attribute y: value2…}, Label), where the Label means the final value of this entity; using the previous restaurant waiting example, the Label would be True or False.
2. Compute the entropy of the set.
3. While (The division of data set is not pure):
    a. For each attribute:
        i. Compute the Information gain if choose such attribute
        ii. Keep track of the attribute with the greatest Information gain

b. Divide the current set with the attribute with the greatest Information gain.

4. Output the model.

In my demo, I have split the procedures into Sources File Handler layer, Samples Splicer layer, Computation layer, and Tree Generator layer. Each layer only focuses on one specific simple job. The combination of them is robust to any kind of legal sample-input. In a word, the source sample file will first go into Sources File Handler layer, and then it would be passed to Tree Generator layer; Tree Generator layer would keep asking the attribute with the best information gain from the Computation layer, which uses functionalities of Tree Generator layer.

The visualization, plotted by PyDot, of my model based on the samples in the above figure:



My implementation has been appended to this submission. I represented the decision tree as just one dictionary object in Python3 in the end. From this, one of the advantages of decision tree can be seem; that is its readability. Unlike a neural network, one can easily understand what a decision tree modal is doing. There is little complex computation going on in the ID3 algorithm while it is still able to generate rules to discover nonlinear relationships between samples and labels. However, the calculations used to find the next best attribute has a time complexity of O(N^2), which means the computations grow exponentially when the samples increase. Also, it is not robust to noise in the data since it uses the greedy strategy and has no back propagation. One mistake in the categorical value could lead to a totally different tree.

My thought for speeding up the finding the next best attribute process is using parallelization when doing the recursive construction of nodes. Once the samples are split, the algorithm would never go back to modify the earlier constructed nodes. As a result, create new threads when constructing new child nodes is safe. However, to implement this needs to use a real tree data structure and largely modify the demo, so I would do this experiment later myself. In theory, this could speed up the process to some extent.

References

Andrew Colin, D. D. (1996). *Building Decision Trees with the ID3 Algorithm.*

Dou, D. (2018). Lecture slides of Learning decision trees .

Ginsberg, M. (1993). *Essentials of Artificial Intelligence.*

Haugeland, J. (1985). *Artificial Intelligence: The Very Idea.* MIT Press.

Mitchell, T. (1997). *Machine Learning.*

Norvig, S. J. (1995). *Artificial Intelligence - A Modern Approach.* Prentice Hall.

Quinlan, R. (1986). *Induction of Decision Trees.*

Rokach, L. (2008). *Data mining with decision trees: theory and applications.* World Scientific Pub Co Inc.

Stubblefield, G. F. (1997). *Artificial Intelligence - Structures and Strategies for Complex Problem Solving.*

Utgoff, P. E. (1989). *Incremental Induction of Decision Trees.* Kluwer Academic Publishers.