How can computer learn with Logo

Dunja Mladenić
J. Stefan Institute
Artificial Intelligence Laboratory
Jamova 39, 61000 Ljubljana, Slovenia
E-mail: Dunja.Mladenic@ijs.si

Abstract

The paper presents a simple way of making your computer learn. Described learning is based on knowledge induction from examples, where entropy measure is used to guide the generalization process. The process is illustrated with the simple algorithm implemented in Logo, to help students to understand it and experiment by means of computer program. Implemented algorithm is one of the simple, well known machine learning algorithms from which some powerful algorithms have been developed and later successfully applied on many real-world problems, to induce knowledge about some real situation. One of the widely known real-world application is in medical diagnosis where machine learning system is used to predict a patient diagnosis from the symptoms.

Key words: teaching your computer, knowledge induction, machine learning

1 Introduction

Learning with knowledge induction is the ability that we often need in our everyday life. For example, observing the current weather situation, we can try to predict the weather for the next day. With days, or better with years of observation our predictions are becoming more and more correct. This process can be viewed as induction of knowledge contained in the weather examples. Each example consists of some values describing the weather situation at the appropriate day and is connected to prediction of the next-day weather. Such induction process can be used as the basis for programming the computer to induce knowledge. The idea of knowledge induction from examples is used in the *machine learning* [1], area of the computer science field called *artificial intelligence*.

Section 2 briefly describes the idea of learning from examples. Section 3 introduces implemented machine learning algorithm. The Logo program is illustrated and discussed in Section 4. Section 5 gives a brief discussion of the presented work.

2 Problem description

To enable our computer to learn from examples, we need a machine learning program and a set of examples of the target situation. This set is used as input to the program. Each example can be viewed as a n-tuple of *attributes* where each attribute describes one property of the concrete situation (e.g. weather situation at the appropriate day). To each example, a value called *class* is assigned (e.g. prediction of the next-day weather). The problem is to induce knowledge describing value of the class by means of the attribute values.

To illustrate the way of knowledge induction with computer, one of the simple machine learning algorithms ID3 [4] has been implemented in LOGO. The algorithm belongs to the family of machine learning algorithms called Top Down Induction of Decision Trees. The main characteristic of these algorithms is that knowledge induction is performed using top-down approach and is represented by means of decision tree. *Decision tree* consists of nodes containing tests of attribute values and leaves containing the predicted class values. Once generated, the tree can be used to classify new (unseen) examples, i.e. to decided which class value a new example has. This is done by proceeding the new example from the root of the tree to the appropriate leaf. Each step consists of testing the current node attribute value example has, and proceeding to the subtree according to the test result. The class value assigned to the final node (leaf) is the class predicted to the proceeded example.

2.1 An example problem domain

example	class	attributes			
number	suitable	outlook	temperature	humidity	windy
1	no	sunny	hot	high	no
2	no	sunny	hot	high	yes
3	yes	overcast	hot	high	no
4	yes	rain	mild	high	no
5	yes	rain	cool	normal	no
6	no	rain	cool	normal	yes
7	yes	overcast	cool	normal	yes
8	no	sunny	mild	high	no
9	yes	sunny	cool	normal	no
10	yes	rain	mild	normal	no
11	yes	sunny	mild	normal	yes
12	yes	overcast	mild	high	yes
13	yes	overcast	hot	normal	no
14	no	rain	mild	high	yes

Table 1: Examples for the 'Saturday morning' domain.

To describe learning from examples as performed by ID3 algorithm a simple 'Saturday morning' [4] domain is used. In this machine learning domain, 14 training examples are given (see Table 1). Each example is labelled with its class: suitable (no, yes), saying if a morning is suitable for some unspecified activity, and 4 attributes: outlook (sunny, overcast, rain), temperature (hot, mild, cool), humidity (high, normal) and windy (no, yes), describing a Saturday morning.

The set of examples given in Table 1 contains the information about the 'Saturday morning'. Some examples describe mornings that are suitable for some unspecified activity (class = `yes`) and the other describe unsuitable mornings (class = `no`). Looking at the first example in Table 1, (`no, sunny, hot, high, no`), one can see that morning is not suitable if `outlook` is `sunny`, `temperature` is `hot`, `humidity` is `high` and `windy` is `no`.

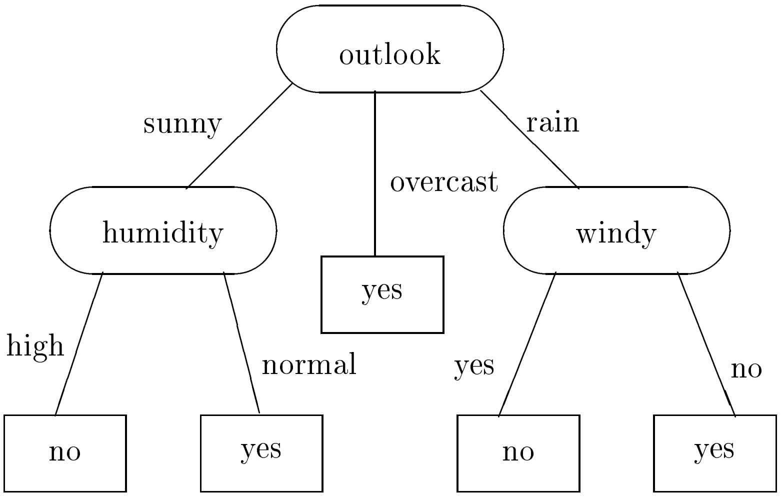

Figure 1: Desicion tree for 'Saturday morning' domain. Ovals labelled with the attribute name denote nodes that test attribute value, possible attribute values are given on the edges to the subtrees and rectangles labelled with the class denote leaves predicting the class.

To induce some knowledge about the morning suitability (class) from the given set of examples, one will search some regularities (correlations) between the class and the attribute values. Focusing the attribute `outlook` in Table 1, it is easy to see that always when `outlook` is `overcast` the class is `yes`. Notice that the middle leaf of the tree in Figure 1 is saying exactly the same. Described machine learning algorithm induces knowledge in the similar way. Figure 1 represents decision tree induced by the ID3 algorithm from the set of examples given in Table 1.Figure 1. This tree can be interpreted (starting with the most left leaf) in the following way:

if outlook = sunny and humidity = high **then** suitable = no
if outlook = sunny and humidity = normal **then** suitable = yes
if outlook = overcast **then** suitable = yes
if outlook = rain and windy = yes **then** suitable = no
if outlook = rainy and windy = no **then** suitable = yes

A new example about some similar situation (same class and attributes) can be classified using the above tree. Suppose that today's weather can be described as follows: `outlook` is `rain`, `temperature` is `hot`, `humidity` is `normal` and `windy` is `no`. We want to know if the day is suitable (if the class is `yes`). The class is predicted using the above tree so that example is proceeded from the root of the tree to the appropriate leaf. To proceed the example, we need to know only values of attributes that are on the path from the root to the appropriate leaf. `Outlook` is `rain` leads to the most right subtree having attribute windy in its root. `Windy` is `no` leads to the most right leaf labelled with the class `yes`. Now we can conclude that today's weather **is** suitable (without knowing values of attributes `temperature` and `humidity`).

3 Algorithm used for knowledge induction

Machine learning algorithm ID3 [4] induces knowledge from examples and represents it in the form of decision tree (see Sec. 2).
The main ID3 algorithm is as follows:

given: set of examples

$ExSet$:= set of examples;
if all examples in $ExSet$ have the same class C **then**
  Make a leaf labelled with the class C
else
  Find the most informative attribute to be the root of the current decision tree;
  According to the value of the chosen attribute split $ExSet$ into subsets;
  Repeat the whole process for each subset; {generate subtrees}
endif
output decision tree;

The main ID3 algorithm constructs decision tree from examples starting with the whole set of examples (root), splitting it into subsets according to the value of chosen attribute and preforming the same procedure to each subset. When some subset consists only of examples of the same class, the splitting process is stopped by construction of the leaf. Illustrated on the tree from Fig 1, to the root is assigned the whole set of examples. Then the `outlook` is chosen as the most informative attribute. To the left branch (attribute `humidity` in the root) only examples with `outlook = sunny` are assigned (see Table 1 ex.no: 1, 2, 8, 9, 11). To the most left leaf (labelled with class `no`) examples with the `outlook = sunny` and `humidity = high` are assigned (see Table 1 ex.no: 1, 2, 8; all having the same class and thus causing a leaf construction). The union of example sets assigned to the tree leaves gives the whole example set.

The most informative attribute is find according to the informativity calculated for each attribute using the following formula:

$$Informativity(Attribute) = Entropy(ExSet) - Entropy(Attribute)$$

$$Entropy(ExSet) = \sum_i p_i \log(p_i); Entropy(Attribute) = \sum_v p_v * Entropy(ExSet_v)$$

where p_i is the probability of i-th class in the set $ExSet$, p_v is the probability of the v-th value of attribute and $ExSet_v$ is the example set assigned to the v-th branch. Intuitively described, the most informative attribute has 'the greatest power' to lower the current entropy.

Illustrated on the examples give in Table 1, the entropy of the whole set is calculated from all 14 examples, 5 having class `no` and 9 having class `yes`. $Entropy(ExSet) = -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} = 0.940$ bits. Calculating the informativity for the `outlook` attribute that has three values (`sunny, overcast, rain`) 5 of the 14 examples have the first value, 2 of them from class `yes` and 3 from class `no`. The probability for the first branch is $p_1 = \frac{5}{14}$ and its entropy is $Entropy(ExSet_1) = -\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5} = 0.971$ bits. Entropy of the rest two branches is calculated in the same way. The entropy and informativity of attribute `outlook` are then calculated: $Entropy(outlook) = \frac{5}{14} * 0.971 + \frac{4}{14} * 0 + \frac{5}{14} * 0.971 = 0.694$; $Informativity(Attribute) = 0.940 - Entropy(outlook) = 0.246$

4 Description of the Logo implementation

Described machine learning algorithm is written in LogoS ([2], [3]), but can be easily adapted for any other Logo interpreter. Program consists of three main parts; the first reading domain (description of attributes and classes, and set of examples), the second generating decision tree and the third writing decision tree. Running the program[1] (procedure `To ID3`) illustrates knowledge induction process and influence of the example set changes to the generated decision tree. The main procedure `To ID3` repeats tree generation and display starting with a single example set and adding examples one by one. The last generated tree (from all 14 examples) is the same as given in Figure 1. What follows is the main procedure `To ID3`.

```
CmLn "ID3 End
To ID3
  TxScr
  Local "TrainExL
  Make "TrainExL []
  RdDomain ;read domain into Class, AttrL and ExL
  Repeat Length :ExL {
    Make "TrainExL AddLast ( Item GetIter :ExL ) :TrainExL
    Make "Tree ( GenTree :Class :AttrL :TrainExL ) ;generate decision tree
    Clr
    print "' ' type GetIter
    type "'. example added :' print ( Item GetIter :ExL ) print "' '
    WrTree :Tree 1 ;write decision tree
    Wait
  }
End
```

Domain reading is performed by the procedure `RdDomain` that sets values of three variables: `Class`, `AttrL` and `ExL`. `Class` is list of class values (`[ no yes ]`) and must contain all possible values of the class. `AttrL` is list of attributes, each attribute is represented with list of two elements (`[ outlook [ sunny overcast rain ]`) the first is attribute name and the second is list of possible attribute values. `ExL` is list of learning examples where each example is represented with list of values (`[ no sunny hot high no ]`), where the first (class) value is followed by the attribute values given in the same order as the attributes in the domain description (`AttrL`). Example set thus looks very similar to the examples given in Table 1.

Experiment with some other domain can be performed by simply changing the contents of these three variables in procedures `GetClass`, `GetAttrL` and `GetExL`.

```
To RdDomain
  Make "Class GetClass
  Make "AttrL GetAttrL
  Make "ExL GetExL
End

To GetClass
  Output [ false true ]
End

To GetAttrL
  Output [
    [ outlook [ sunny overcast rain ]]
    [ temperature [ hot mild cool ]]
    [ humidity [ high normal ]]
    [ windy [ false true ]]
  ]
End
```

```
To GetExL
  Output [
    [ false sunny hot high false ]
    [ false sunny hot high true ]
    [ true overcast hot high false ]
    [ true rain mild high false ]
    [ true rain cool normal false ]
    [ false rain cool normal true ]
    [ true overcast cool normal true ]
    [ false sunny mild high false ]
    [ true sunny cool normal false ]
    [ true rain mild normal false ]
    [ true sunny mild normal true ]
    [ true overcast mild high true ]
    [ false rain mild high true ]
    [ true overcast hot normal false ]
  ]
End
```

Decision tree is generated with the procedure `GenTree` that implements the main ID3 algorithm described in Section 3. It has class, attributes and examples as parameters. Procedure `GenTree` consists of three parts: two simple stopping criteria (example set is empty, all examples are of the same class) and subtrees generation. Subtrees generation is the central part of the whole algorithm and can be briefly described as follows. First, the most informative attribute is found and example set is splitted according to its values (`FindBestAttr`). Then, procedure `GenSubTrees` is called with the domain information (class, attributes) and the list of splited example set. Procedure `GenSubTrees` calls `GenTree` for each element of the list (each example set) and generates list of subtrees `TreeL`.

5 Discussion

Simple knowledge induction algorithm described in the paper, can be used as illustration of the process of learning from example situations. It is also implemented in a short LOGO program and can be used for simple experiments.[1] For the purpose of experiments and analysis, input data can be modified to describe some new situation. The idea is also for the students to get some opinions about way computer can perform learning, to try to be teachers to computer.

References

[1] Bratko, I (1990) *Prolog programming for artificial intelligence*, 2nd edition, Addison-Wesley, 1990.

[2] Batagelj, V., Grobelnik, M. and Zupanič, D. (1991) The design of LOGOS, *Third Europen Logo Conference Proceedings*, pp. 287-296, Parma, Italy.

[3] Grobelnik, M. and Zupanič, D. (1991) Inside LOGOS implementation, *Third Europen Logo Conference Proceedings*, pp. 297-311, Parma, Italy.

[4] Quinlan, J.R. (1986) Induction of decision trees. *Machine Learning 1*, pp. 81-106. Kluwer Academic Publishers.

[1] Program written in LOGOS is available by sending an e-mail to the author.