Investigation into the inner workings of Concept Learning and Decision Trees

Adriaan Louw (53031377)

April 15, 2018

Contents

1 Abstract **1**

2 Method **1**

3 Results **1**

3.1 Concept Learning 1

3.1.1 What is Concept Learning? 1

3.1.2 General-to-specific ordering of hypotheses 2

3.1.3 The FIND-S algorithm 2

3.1.4 Version Spaces 3

3.1.5 The CANDIDATE-ELIMINATION algorithm 3

3.1.6 Inductive bias 4

3.2 Decision Trees 4

3.2.1 The ID3 Algorithm 4

3.2.2 Working example of the ID3 Algorithm 5

1 Abstract

2 Method

3 Results

3.1 Concept Learning

3.1.1 What is Concept Learning?

We can describe Concept Learning in terms of trying to teach a machine to classify animals as either dogs or not.

We assuming we classify leaves by the following parameters

- Is the animal large or small? (Size)
- Is it brown or black? (Colour)
- Is the tail long or short? (Tail length)

- Does it have 2 or 4 legs?
- Does it bark?

We can then describe animals according to these parameters {size,colour,tail length,no of legs,does it bark}. This is called our hypothesis space.

Then a particular animal can be described as $\langle small, brown, short, 4, no \rangle$ for example. For our dataset we have a selection of animals and whether that animal is a dog or not.

The concept that we wish the machine to learn is which values of each parameter defines the Concept of a do. In our case that would be $\langle ?, ?, ?, 4, yes \rangle$ where ? means any value.

Formally we can say:

- We have X which is all the instances in the hypothesis space
- We also have $c(x)$ which is the function that returns whether an animal is a dog or not
- Training data $D = \{\langle x, c(x) \rangle : x \in X, c(x) \in \{0, 1\}\}$

(Chandola, 2018)

3.1.2 General-to-specific ordering of hypotheses

From the above example, the most general hypothesis would be $\langle ?, ?, ?, ?, ? \rangle$. This means any animal would satisfy the hypothesis. The most specific hypothesis is $\langle \emptyset, \emptyset, \emptyset, \emptyset, \emptyset \rangle$.

We can sort hypotheses based on whether they are more general or specific than other hypotheses.

Consider the following hypotheses: $h_1 = \langle ?, ?, ?, ?, yes \rangle$ and $h_2 = \langle ?, brown, ?, 4, yes \rangle$. h_1 is more general than h_2. In other words it less specific on what the parameters have to be to satisfy the hypothesis.

(Riedmiller, 2009)

3.1.3 The FIND-S algorithm

The goal of the FIND-S Algorithm is to find the "maximally specific hypothesis". This means it is the most specific hypothesis that can be found that represents the concept.

Assuming we have a the following dataset :

$$\begin{aligned} D &= \sum_{i=1}^{n} \{x_i, c(x_i)\} \\ &= \{\{\langle small, brown, short, 4, no \rangle, 0\}, \\ &\quad \{\langle small, brown, long, 4, yes \rangle, 1\}, \\ &\quad \{\langle large, black, short, 2, no \rangle, 0\}, \\ &\quad \{\langle large, black, long, 4, yes \rangle, 1\}\} \end{aligned} \tag{1}$$

The algorithm works as follows. We start with a empty hypothesis $h = \emptyset$. Then we iterate through each hypothesis in our data set (see Equation 1). If

$c(x_i) = 0$ then go to the next hypothesis. If not, then we do a pairwise and between h and the data x and update h. As in $h \leftarrow h \wedge x$. Then we go onto the next data point.

Given data set D from equation 1. We start with $h = \emptyset$. The first data point has $c(x) = 0$. So we ignore it. The second data point has $c(x) = 1$ so we do a pairwise and with h.

$$\begin{aligned} h \leftarrow & h \wedge x \\ \leftarrow & \emptyset \wedge \langle small, brown, long, 4, yes \rangle \\ \leftarrow & \langle small, brown, long, 4, yes \rangle \end{aligned} \tag{2}$$

Thus, $h = \{small, brown, long, 4, yes\}$. The next data point is also ignores because it does not define a positive match. Then we go onto the last data point.

$$\begin{aligned} h \leftarrow & h \wedge x \\ \leftarrow & \langle small, brown, long, 4, yes \rangle \wedge \langle large, black, long, 4, yes \rangle \\ \leftarrow & \langle ?, ?, long, 4, yes \rangle \end{aligned} \tag{3}$$

Thus we have the hypothesis we learned form the data set is $h = \langle ?, ?, long, 4, yes \rangle$.

(Chandola, 2018)

3.1.4 Version Spaces

Definition: A hypothesis h is consistent with the set of training examples D of target concept c if and only if $h(x) = c(x)$ for each training example $\langle x, c(x) \rangle$ in D.

$$Consistent(h, D) \equiv (\forall \langle x, c(x) \rangle \in D) h(x) = c(x) \tag{4}$$

Definition: the *version space*, $VS_{H,D}$, with respect to hypothesis space H and training examples D, is the subset of hypotheses from H consistent with all training examples in D.

$$VS_{H,D} \equiv \{h \in H | Consistent(h, D)\} \tag{5}$$

A version space is represented by a General and a Specific boundary. The General boundary is the set of maximally general members consistent with the training data. The Specific boundary is the set of maximally specific members consistent with the training data.

(Chandola, 2018; Riedmiller, 2009)

3.1.5 The CANDIDATE-ELIMINATION algorithm

The algorithm returns 2 sets. First G, the set of maximally general hypotheses. Secondly S, the set of maximally specific hypotheses.

As a worked example we will use the dataset from equation (1). Start with the initial state:

$$\begin{aligned} S_0 : & \{\langle \emptyset, \emptyset, \emptyset, \emptyset, \emptyset \rangle\} \\ G_0 : & \{\langle ?, ?, ?, ?, ? \rangle\} \end{aligned} \tag{6}$$

then using data point 1 ,$\{\langle small, brown, short, 4, no\rangle, 0\}$,we get

$$\begin{aligned} S_1 :&\{\langle\emptyset,\emptyset,\emptyset,\emptyset,\emptyset\rangle\}\\ G_1 :&\{\langle S,?,?,?,?\rangle,\langle ?,brown,?,?,?\rangle,\langle ?,?,short,?,?\rangle,\langle ?,?,?,4,?\rangle,\langle ?,?,?,?,no\rangle\} \end{aligned} \tag{7}$$

Using the positive point 2, $\{\langle small, brown, long, 4, yes\rangle, 1\}$,

$$\begin{aligned} S_2 :&\{\langle small, brown, long, 4, yes\rangle\}\\ G_2 :&\{\langle S,?,?,?,?\rangle,\langle ?,brown,?,?,?\rangle,\langle ?,?,?,4,?\rangle,\} \end{aligned} \tag{8}$$

Using the third data point ,$\{\langle large, black, short, 2, no\rangle, 0\}$, we get

$$\begin{aligned} S_3 :&\{\langle small, brown, long, 4, yes\rangle\}\\ G_3 :&\{\langle S,?,?,?,?\rangle,\langle ?,?,?,4,?\rangle,\} \end{aligned} \tag{9}$$

and the using the last data point ,$\{\langle large, black, long, 4, yes\rangle, 1\}$, we get

$$\begin{aligned} S_3 :&\{\langle ?,?,?,4,yes\rangle\}\\ G_3 :&\{\langle ?,?,?,4,?\rangle,\} \end{aligned} \tag{10}$$

(Chandola, 2018; Riedmiller, 2009)

3.1.6 Inductive bias

In our previous example we have only considered hypotheses made of conjunctions of attributes. This means that we cannot describe certain target concepts. In other words we have introduced a bias into our representation of hypothesis space. To include disjunctions and negations we need to create a much larger hypothesis space. The size of the current hypothesis space is 2^5 (there are 5 binary attributes). The new unbiased hypothesis space would be the power set of this space. Thus it will be the size of 2^{2^5}.

In such a hypothesis space the S boundary will be the disjunction of the positive data point and the G boundary will be the negated disjunction of negative data points. This means no meaningful deductions can be made from the training examples without a bias.

(Mitchell, 1997; Riedmiller, 2009)

3.2 Decision Trees

3.2.1 The ID3 Algorithm

The ID3 algorithm generates decision tree based on the information gain of an attribute A. Given a set S of object with attributes and each object can be classified into 2 categories. Category P for positive and category N for Negative. Then p will be the number of object in category P and n will; be the number of objects in category N. The information in the S can be seen as

$$I(p,n) = -\frac{p}{p+n}log_2\frac{p}{p+n} - \frac{n}{p+n}log_2\frac{n}{p+n} \tag{11}$$

For a specific attribute A and the number of options on that attribute v, the expected information required is:

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p+n} I(p_i, n_i) \tag{12}$$

The information gain can then be defined as

$$gain(A) = I(p,n) - E(A) \tag{13}$$

The attribute with the most information gain will be chosen as the root node. Then at each of the sub branches the algorithm will run again, on the other attributes of the reduced set, to determine which attribute has the most information gain. Then that attribute will be the root node of the sub-tree. (Mitchell, 1997; Quinlan, 1986)

3.2.2 Working example of the ID3 Algorithm

Using the *Play Tennis* data set from Quinlan (1986). Given the dataset size is 14 and the number of positive examples p is 9 and the number of negative examples n is 5:

$$\begin{aligned} I(p,n) &= -\frac{9}{14}log_2\frac{9}{14} - \frac{5}{14}log_2\frac{5}{14} \\ &= 0.940 \end{aligned} \tag{14}$$

To calculate $E(A) = E(outlook)$ (for the outlook attribute, we need to first calculate $I(p_i, n_i)$ for its 3 values namely: sunny($i = 1$),overcast($i = 2$) and rain($i = 3$)

$$\begin{aligned} I(p_1, n_1) =& I(2,3) = 0.971 \\ I(p_2, n_2) =& I(4,0) = 0 \\ I(p_3, n_2) =& I(3,2) = 0.971 \end{aligned} \tag{15}$$

Then,

$$\begin{aligned} E(outlook) &= \frac{5}{14}I(p_1,n_1) + \frac{4}{14}I(p_2,n_2) + \frac{5}{14}I(p_3,n_3) \\ &= 0.694 \end{aligned} \tag{16}$$

Calculating the gain of the attribute outlook gives

$$gain(outlook) = 0.940 - E(outlook) = 0.246 \tag{17}$$

Now we need to calculate the gain of the other attributes namely temperature, humidity and windy. The attribute temperature has 3 possible values hot,mild and cool. There are 2 positive and 2 negative objects in the dataset which has the value hot for the attribute temperature. This gives $p_1 = 2$ and $n_2 = 2$. Similarly for mild $p_2 = 4, N_2 = 2$ and cool $p_3 = 3, N_3 = 1$. This gives

$$\begin{aligned} gain(temperature) &= I(p,n) - E(temperature) \\ &= 0.940 - \frac{4}{14}I(p_1,n_1) - \frac{6}{14}I(p_2,n_2) - \frac{6}{14}I(p_3,n_3) \\ &= 0.940 - 0.911 \\ &= 0.029 \end{aligned} \tag{18}$$

Now for humidity we get high($p_1 = 3, n_1 = 4$), normal ($p_2 = 6, n_2 = 1$) which gives

$$\begin{aligned} gain(humidity) &= I(p,n) - E(humidity) \\ &= 0.940 - \frac{7}{14}I(p_1,n_1) - \frac{7}{14}I(p_2,n_2) \\ &= 0.940 - 0.788 \\ &= 0.151 \end{aligned} \tag{19}$$

And for windy we have true ($p_1 = 3, n_1 = 3$) and false ($p_2 = 6, n_2 = 2$) where

$$\begin{aligned} gain(windy) &= I(p,n) - E(windy) \\ &= 0.940 - \frac{6}{14}I(p_1,n_1) - \frac{8}{14}I(p_2,n_2) \\ &= 0.940 - 0.891 \\ &= 0.048 \end{aligned} \tag{20}$$

Conclusion

Now comparing the gains for the 4 attributes we find that the gain for outlook is the highest. We make this the root of our tree. We then subdivide the dataset according to the values of outlook. Then create a sub-trees for each.

When outlook = sunny we have 5 data points of which 2 are positive and 3 are negative. We need to calculate $I(p,n) = I(2,3)$.

$$\begin{aligned} I(p,n) &= -\frac{2}{5}log_2\frac{2}{5} - \frac{3}{5}log_2\frac{3}{5} \\ &= 0.971 \end{aligned} \tag{21}$$

Then for when temperature = hot we have $p_1 = 0$ and $n_1 = 2$, temperature = mild ($p_2 = 1, n_2 = 1$) and temperature = cool ($p_3 = 1, n_3 = 0$). Giving

$$\begin{aligned} gain(temperature) &= I(p,n) - E(temperature) \\ &= 0.971 - \frac{2}{5}I(p_1,n_1) - \frac{2}{5}I(p_2,n_2) - \frac{1}{5}I(p_3,n_3) \\ &= 0.971. - 0.4 \\ &= 0.571 \end{aligned} \tag{22}$$

For Humidity: high($p_1 = 0, n_1 = 3$) and normal($p_2 = 2, n_2 = 0$)

$$\begin{aligned} gain(humidity) &= I(p,n) - E(humidity) \\ &= 0.971 - \frac{3}{5}I(p_1,n_1) - \frac{2}{5}I(p_2,n_2) \\ &= 0.971. - 0.0 \\ &= 0.971 \end{aligned} \tag{23}$$

And Windy: true($p_1 = 1, n_1 = 1$) and false($p_2 = 1, n_2 = 2$)

$$\begin{aligned} gain(humidity) &= I(p,n) - E(humidity) \\ &= 0.971 - \frac{2}{5}I(p_1,n_1) - \frac{3}{5}I(p_2,n_2) \\ &= 0.971. - 0.951 \\ &= 0.020 \end{aligned} \tag{24}$$

This tells us the highest gain is for the attribute humidity. Therefore a new sub-tree is formed underneath our root tree at the node where outlook = sunny. The leaf nodes of the sub-tree will then be leaf nodes of the final tree since when outlook = sunny and humidity = high the class is always N and when outlook is sunny and humidity is normal the class is always P.

Going back to the root tree when outlook = overcast we see that the class is always P so this will be a leaf node of the final tree.

Finally we need to determine the sub-tree under outlook = rain. We need to calculate the gain again for temperature, humidity and windy. There are 5 data point for outlook = rain. There are 3 positive and 2 negative which gives

$$\begin{aligned} I(p,n) &= -\frac{3}{5}log_2\frac{2}{5} - \frac{2}{5}log_2\frac{3}{5} \\ &= 0.971 \end{aligned} \tag{25}$$

For Temperature: hot ($p_1 = 0, n_1 = 0$), mild($p_2 = 2, n_2 = 1$) and cool($p_3 = 1, n_3 = 1$)

$$\begin{aligned} gain(temperature) &= I(p,n) - E(temperature) \\ &= 0.971 - \frac{0}{5}I(p_1,n_1) - \frac{3}{5}I(p_2,n_2) - \frac{2}{5}I(p_3,n_3) \\ &= 0.971. - 0.951 \\ &= 0.010 \end{aligned} \tag{26}$$

For Humidity: high($p_1 = 1, n_1 = 1$) and normal($p_2 = 2, n_2 = 1$)

$$\begin{aligned} gain(humidity) &= I(p,n) - E(humidity) \\ &= 0.971 - \frac{2}{5}I(p_1,n_1) - \frac{3}{5}I(p_2,n_2) \\ &= 0.971. - 0.951 \\ &= 0.020 \end{aligned} \tag{27}$$

For Windy: true($p_1 = 0, n_1 = 2$) and false($p_2 = 3, n_2 = 0$)

$$\begin{aligned} gain(humidity) &= I(p,n) - E(humidity) \\ &= 0.971 - \frac{2}{5}I(p_1,n_1) - \frac{3}{5}I(p_2,n_2) \\ &= 0.971. - 0.0 \\ &= 0.971 \end{aligned} \tag{28}$$

Which means the attribute that has the highest gain when outlook = rainy is humidity. We create a new sub-tree of humidity and the location outlook = rainy. Additionally, when windy = true, the class is always N and when windy = false the class is always P. Which means the leaves of windy are leaf nodes of the final tree as well. This means our decision tree is complete.

References

Chandola, V. (2018). *Introduction to machine learning.* Retrieved from `https://www.cse.buffalo.edu// chandola/teaching/machinelearningdocs/parta2-handout.pdf`

Mitchell, T. M. (1997). *Machine Learning.* McGraw-Hill.

Quinlan, J. R. (1986). Induction of Decision Trees. *Machine Learning*, *1*(1), 81–106.

Riedmiller, M. (2009). *Concept Learning.* Retrieved from `http://ml.informatik.uni-freiburg.de/former/_media/documents/teaching/ss09/ml/version.pdf`