Decision Tree Induction: Using Entropy for Attribute Selection

Dr. Akinul Islam Jony

Assistant Professor

Dept. of Computer Science
Faculty of Science & Technology
American International University – Bangladesh

akinul@aiub.edu

Outline

- Attribute Selection: An Experiment
- Alternative Decision Trees
- Entropy
- Information Gain
- Using Entropy for Attribute Selection
- Maximising Information Gain

Objectives & Outcomes

- Get to know about some alternative strategies for selecting attributes at each stage of the TDIDT decision tree generation algorithm
- Get to know the risk of obtaining decision trees that are entirely meaningless
- Understand the importance of a good choice of attribute selection strategy
- Get to know one of the most widely used strategies which is based on minimising *entropy* (or equivalently maximising *information gain*)

Attribute Selection: An Experiment

- It was shown that the TDIDT algorithm is guaranteed to terminate and to give a decision tree that correctly corresponds to the data, provided that the *adequacy condition* is satisfied.
    - The *adequacy condition* is that no two instances with identical attribute values have different classifications.
- However, it was also pointed out that the TDIDT algorithm is *underspecified*.
- Provided that the *adequacy condition* is satisfied, any method of choosing attributes will produce a decision tree.
- We will begin this lecture by considering the decision trees obtained from using some poorly chosen strategies for *attribute selection* and then go on to describe one of the most widely used approaches and look at how the results compare.

Attribute Selection: An Experiment

- First we look at the decision trees produced by using the three attribute selection strategies listed below.
  - *takefirst* – for each branch take the attributes in the order in which they appear in the training set, working from left to right, e.g. for the *degrees* training set in the order *SoftEng*, *ARIN*, *HCI*, *CSA* and *Project*.
  - *takelast* – as for *takefirst*, but working from right to left, e.g. for the *degrees* training set in the order *Project*, *CSA*, *HCI*, *ARIN* and *SoftEng*.
  - *random* – make a random selection (with equal probability of each attribute being selected).
- *As always no attribute may be selected twice in the same branch*.
- ***Warning***: *these three strategies are given here for purposes of illustration only. They are not intended for serious practical use but provide a basis for comparison with other methods introduced later*.

Attribute Selection: An Experiment

- The table shows the results of running the TDIDT algorithm with attribute selection strategies *takefirst*, *takelast* and *random* in turn to generate decision trees for the seven datasets *contact lenses*, *lens24*, *chess*, *vote*, *monk1*, *monk2* and *monk3*.

Dataset	take first	take last	random					most	least
			1	2	3	4	5		
contact_lenses	42	27	34	38	32	26	35	42	26
lens24	21	9	15	11	15	13	11	21	9
chess	155	56	94	52	107	90	112	155	52
vote	40	79	96	78	116	110	96	116	40
monk1	60	75	82	53	87	89	80	89	53
monk2	142	112	122	127	109	123	121	142	109
monk3	69	69	43	46	62	55	77	77	43

Number of Branches Generated by TDIDT with Three Attribute Selection Methods

- In each case the value given in the table is the number of *branches* in the decision tree generated.

Attribute Selection: An Experiment

- The *random* strategy was used five times for each dataset.
- The last two columns record the number of *branches* in the largest and the smallest of the trees generated for each of the datasets.

Dataset	take first	take last	random 1	random 2	random 3	random 4	random 5	most	least
contact_lenses	42	27	34	38	32	26	35	42	26
lens24	21	9	15	11	15	13	11	21	9
chess	155	56	94	52	107	90	112	155	52
vote	40	79	96	78	116	110	96	116	40
monk1	60	75	82	53	87	89	80	89	53
monk2	142	112	122	127	109	123	121	142	109
monk3	69	69	43	46	62	55	77	77	43

Number of Branches Generated by TDIDT with Three Attribute Selection Methods

Attribute Selection: An Experiment

- In all cases there is a considerable difference. This suggests that although in principle the attributes can be chosen in any arbitrary way, the difference between a good choice and a bad one may be considerable.

Dataset	take first	take last	random					most	least
			1	2	3	4	5		
contact_lenses	42	27	34	38	32	26	35	42	26
lens24	21	9	15	11	15	13	11	21	9
chess	155	56	94	52	107	90	112	155	52
vote	40	79	96	78	116	110	96	116	40
monk1	60	75	82	53	87	89	80	89	53
monk2	142	112	122	127	109	123	121	142	109
monk3	69	69	43	46	62	55	77	77	43

Number of Branches Generated by TDIDT with Three Attribute Selection Methods

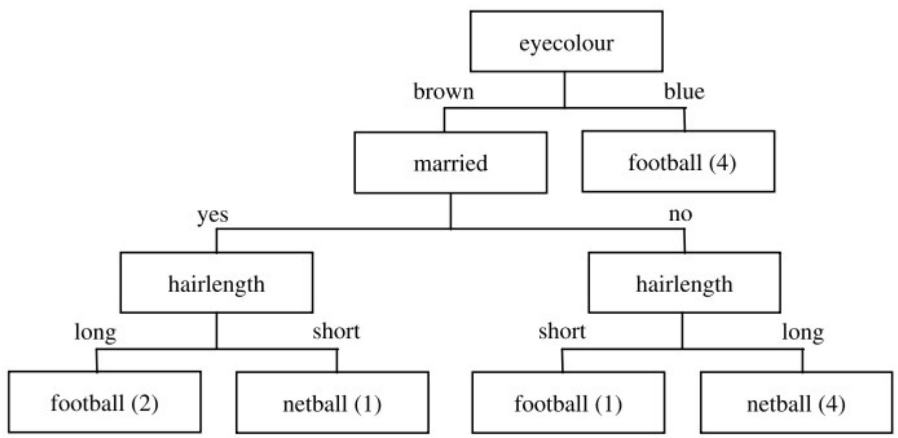
Alternative Decision Trees: The *Football/Netball* Example

- A fictitious university requires its students to enrol in one of its sports clubs, either the Football Club or the Netball Club. It is forbidden to join both clubs. Any student joining no club at all will be awarded an automatic failure in their degree.
- Table gives a training set of data collected about 12 students, tabulating four items of data about each one (eye colour, marital status, sex and hair length) against the club joined.
- **What determines who joins which club?**
- It is possible to generate many different trees from this data using the TDIDT algorithm.

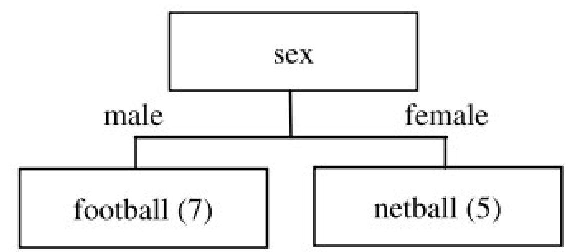
eyecolour	married	sex	hairlength	class
brown	yes	male	long	football
blue	yes	male	short	football
brown	yes	male	long	football
brown	no	female	long	netball
brown	no	female	long	netball
blue	no	male	long	football
brown	no	female	long	netball
brown	no	male	short	football
brown	yes	female	short	netball
brown	no	female	long	netball
blue	no	male	long	football
blue	no	male	short	football

Training Set for the *Football/Netball* Example

Alternative Decision Trees: The *Football/Netball* Example



Football/Netball Example: Decision Tree 1



Football/Netball Example: Decision Tree 2

Alternative Decision Trees: The *Football/Netball* Example

- Which one of the decision trees is correct?
- Although it is tempting to say that it is, it is best to avoid using terms such as ‘correct’ and ‘incorrect’ in this context.
- All we can say is that both decision trees are compatible with the data from which they were generated.
- The only way to know which one gives better results for unseen data is to use them both and compare the results.

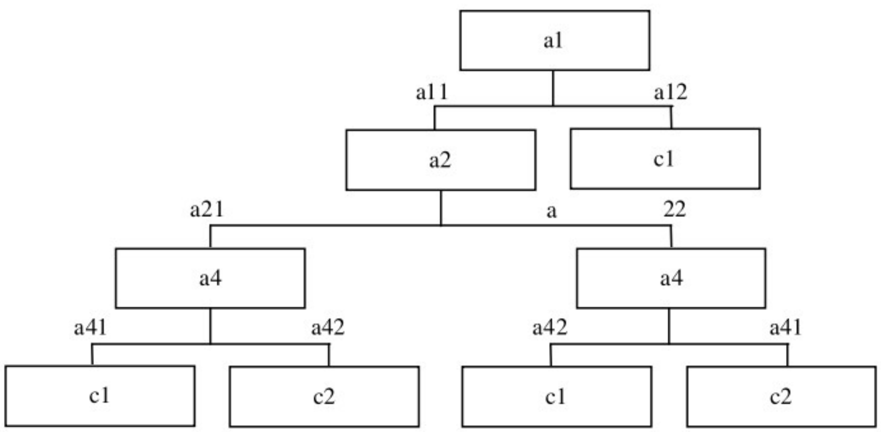
Alternative Decision Trees: The *anonymous* Dataset

- Now consider a different dataset.
- Here we have a training set of 12 instances. There are four attributes, a1, a2, a3 and a4, with values a11, a12 etc., and two classes c1 and c2.

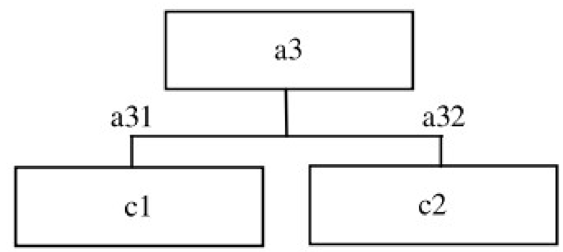
a1	a2	a3	a4	class
a11	a21	a31	a41	c1
a12	a21	a31	a42	c1
a11	a21	a31	a41	c1
a11	a22	a32	a41	c2
a11	a22	a32	a41	c2
a12	a22	a31	a41	c1
a11	a22	a32	a41	c2
a11	a22	a31	a42	c1
a11	a21	a32	a42	c2
a11	a22	a32	a41	c2
a12	a22	a31	a41	c1
a12	a22	a31	a42	c1

The *anonymous* Dataset

Alternative Decision Trees: The *anonymous* Dataset



Anonymous Data: Decision Tree 1



Anonymous Data: Decision Tree 2

Alternative Decision Trees: The *anonymous* Dataset

- Which tree is better?
- This *anonymous* dataset is actually the *football/netball* example in anonymised form.
- The effect of replacing meaningful attribute names such as *eyecolour* and *sex* with meaningless names such as *a1* and *a3* is considerable.
- Data mining algorithms generally do not allow the use of any background knowledge the user has about the domain from which the data is drawn, such as the 'meaning' and relative importance of attributes, or which attributes are most or least likely, to determine the classification of an instance.
- It is easy to see that a decision tree involving tests on *eyecolour*, *hairlength* etc. is meaningless when it is given in isolation, but if those attributes were part of a much larger number (possibly many thousands) in a practical application what would there be to prevent meaningless decision rules from being generated?
  - Apart from vigilance and a good choice of algorithm, the answer to this is 'nothing at all'.

Alternative Decision Trees

- That’s why, the quality of the strategy used to **select the attribute to split on** at each stage is clearly of vital importance.

- This is the topic to which we now turn.

Choosing Attributes to Split On: Using Entropy

- The attribute selection techniques (*takefirst*, *takelast* and *random*) were discussed for illustrative purposes only.
- For practical use several much superior methods are available.
- One commonly used method is to select the attribute that minimises the value of *entropy*, thus maximising the *information gain*.

Choosing Attributes to Split On: Using Entropy

- Table shows the size of the tree with most and least *branches* produced by the ***takefirst***, ***takelast*** and ***random*** attribute selection strategies for a number of datasets.
- The final column shows the number of branches generated by the '***entropy***' attribute selection method.
- In all cases the number of rules in the decision tree generated using the '*entropy*' method is less than or equal to the smallest number generated using any of the other attribute selection criteria introduced so far.
  - In some cases, such as for the chess dataset, it is considerably fewer.

Dataset	excluding entropy		entropy
	most	least	
contact lenses	42	26	**16**
lens24	21	**9**	**9**
chess	155	52	**20**
vote	116	40	**34**
monk1	89	53	**52**
monk2	142	109	**95**
monk3	77	43	**28**

A Comparison of Attribute Selection Methods

Choosing Attributes to Split On: Using Entropy

- There is no guarantee that using entropy will always lead to a small decision tree, but experience shows that it generally produces trees with fewer *branches* than other attribute selection criteria (not just the basic ones introduced so far).
- Experience also shows that small trees tend to give more accurate predictions than large ones, although there is certainly no guarantee of infallibility (steadiness).

Dataset	excluding entropy		entropy
	most	least	
contact lenses	42	26	**16**
lens24	21	**9**	**9**
chess	155	52	**20**
vote	116	40	**34**
monk1	89	53	**52**
monk2	142	109	**95**
monk3	77	43	**28**

A Comparison of Attribute Selection Methods

The *lens24* Dataset

- The *lens24* dataset is ophthalmological data about contact lenses.
- It comprises 24 instances linking the values of four attributes *age* (i.e. age group), *specRx* (spectacle prescription), *astig* (whether astigmatic) and *tears* (tear production rate) with one of three classes 1, 2 and 3 (signifying respectively that the patient should be fitted with hard contact lenses, soft contact lenses or none at all).

Value of attribute				Class
age	specRx	astig	tears	
1	1	1	1	3
1	1	1	2	2
1	1	2	1	3
1	1	2	2	1
1	2	1	1	3
1	2	1	2	2
1	2	2	1	3
1	2	2	2	1
2	1	1	1	3
2	1	1	2	2
2	1	2	1	3
2	1	2	2	1
2	2	1	1	3
2	2	1	2	2
2	2	2	1	3
2	2	2	2	3
3	1	1	1	3
3	1	1	2	3
3	1	2	1	3
3	1	2	2	1
3	2	1	1	3
3	2	1	2	2
3	2	2	1	3
3	2	2	2	3

classes
1: hard contact lenses
2: soft contact lenses
3: no contact lenses

age
1: young
2: pre-presbyopic
3: presbyopic

specRx
(spectacle prescription)
1: myopia
2: high hypermetropia

astig
(whether astigmatic)
1: no
2: yes

tears
(tear production rate)
1: reduced
2: normal

Training Set for *lens24* Data

Entropy

- ***Entropy*** is an information-theoretic measure of the 'uncertainty' contained in a training set, due to the presence of more than one possible classification.
- If there are K classes, we can denote the proportion of instances with classification i by p_i for i = 1 to K.
  - The value of p_i is the number of occurrences of class i divided by the total number of instances, which is a number between 0 and 1 inclusive.
- The entropy of the training set is denoted by E. It is measured in 'bits' of information and is defined by the formula

$$E = -\sum_{i=1}^{K} p_i \log_2 p_i$$

summed over the non-empty classes only, i.e. classes for which $p_i \neq 0$.

Entropy

- The value of $-\boldsymbol{p_i} \log_2 \boldsymbol{p_i}$ is positive for values of p_i greater than zero and less than 1.
  - When $p_i = 1$ the value of $-\boldsymbol{p_i} \log_2 \boldsymbol{p_i}$ is zero.
- This implies that E is positive or zero for all training sets.
- It takes its minimum value (zero) if and only if all the instances have the same classification, in which case there is only one non-empty class, for which the probability is 1.
- Entropy takes its maximum value when the instances are equally distributed amongst the K possible classes.
  - In this case the value of each p_i is $1/K$, which is independent of i. so

$$\begin{aligned} E &= -\sum_{i=1}^{K} (1/K)\log_2(1/K) \\ &= -K(1/K)\log_2(1/K) \\ &= -\log_2(1/K) = \log_2 K \end{aligned}$$

  - If there are 2, 3 or 4 classes this maximum value is 1, 1.5850 or 2, respectively.

Entropy

- For the *lens24* training set of 24 instances, there are 3 classes.
- There are 4 instances with classification 1, 5 instances with classification 2 and 15 instances with classification 3.
- So p_1 = 4/24, p_2 = 5/24 and p_3 = 15/24.
- We will call the *entropy* of the training set E_{start}.
- So $E_{start} = -(4/24) \log_2(4/24) - (5/24) \log_2(5/24) - (15/24) \log_2(15/24)$

  $= 0.4308 + 0.4715 + 0.4238$

  $= 1.3261$ bits

Value of attribute				Class
age	specRx	astig	tears	
1	1	1	1	3
1	1	1	2	2
1	1	2	1	3
1	1	2	2	1
1	2	1	1	3
1	2	1	2	2
1	2	2	1	3
1	2	2	2	1
2	1	1	1	3
2	1	1	2	2
2	1	2	1	3
2	1	2	2	1
2	2	1	1	3
2	2	1	2	2
2	2	2	1	3
2	2	2	2	3
3	1	1	1	3
3	1	1	2	3
3	1	2	1	3
3	1	2	2	1
3	2	1	1	3
3	2	1	2	2
3	2	2	1	3
3	2	2	2	3

classes
1: hard contact lenses
2: soft contact lenses
3: no contact lenses

age
1: young
2: pre-presbyopic
3: presbyopic

specRx
(spectacle prescription)
1: myopia
2: high hypermetropia

astig
(whether astigmatic)
1: no
2: yes

tears
(tear production rate)
1: reduced
2: normal

Training Set for *lens24* Data

Information Gain

- Information Gain, $IG = E_{start} - E_{new}$
  - E_{start} entropy of the training set measured by the formula

$$E = -\sum_{i=1}^{K} p_i \log_2 p_i$$

  - E_{new} entropy is calculated for each attribute

- We get an attribute's *IG* value by subtracting its E_{new} from E_{start}
- Attribute with highest *IG* value is selected

Using Entropy for Attribute Selection

- The process of decision tree generation by repeatedly splitting on attributes is equivalent to partitioning the initial training set into smaller training sets repeatedly, until the *entropy* of each of these subsets is zero (i.e. each one has instances drawn from only a single class).
- At any stage of this process, splitting on any attribute has the property that **the average entropy of the resulting subsets will be less than (or occasionally equal to) that of the previous training set**.

Using Entropy for Attribute Selection

- For the ***lens24*** training set, splitting on attribute ***age*** would give three subsets as shown below.

Value of attribute				Class
age	specRx	astig	tears	
1	1	1	1	3
1	1	1	2	2
1	1	2	1	3
1	1	2	2	1
1	2	1	1	3
1	2	1	2	2
1	2	2	1	3
1	2	2	2	1

Training Set 1 for *lens24*

Value of attribute				Class
age	specRx	astig	tears	
2	1	1	1	3
2	1	1	2	2
2	1	2	1	3
2	1	2	2	1
2	2	1	1	3
2	2	1	2	2
2	2	2	1	3
2	2	2	2	3

Training Set 2 for *lens24*

Value of attribute				Class
age	specRx	astig	tears	
3	1	1	1	3
3	1	1	2	3
3	1	2	1	3
3	1	2	2	1
3	2	1	1	3
3	2	1	2	2
3	2	2	1	3
3	2	2	2	3

Training Set 3 for *lens24*

Using Entropy for Attribute Selection

- **Training set 1 (*age* = 1)**

  - Entropy $E_1 = -(2/8)\log_2(2/8) - (2/8)\log_2(2/8) - (4/8)\log_2(4/8)$
    $= 0.5 + 0.5 + 0.5$
    $= 1.5$

Value of attribute				Class
age	specRx	astig	tears	
1	1	1	1	3
1	1	1	2	2
1	1	2	1	3
1	1	2	2	1
1	2	1	1	3
1	2	1	2	2
1	2	2	1	3
1	2	2	2	1

Training Set 1 for *lens24* Example

Using Entropy for Attribute Selection

- **Training set 2 (*age* = 2)**

  - Entropy $\boldsymbol{E_2} = -(1/8)\log_2(1/8) - (2/8)\log_2(2/8) - (5/8)\log_2(5/8)$
    $= 0.375 + 0.5 + 0.4238$
    $= 1.2988$

Value of attribute				Class
age	specRx	astig	tears	
2	1	1	1	3
2	1	1	2	2
2	1	2	1	3
2	1	2	2	1
2	2	1	1	3
2	2	1	2	2
2	2	2	1	3
2	2	2	2	3

Training Set 2 for *lens24* Example

Using Entropy for Attribute Selection

- **Training set 3 (*age* = 3)**

  - Entropy $\boldsymbol{E_3}$ = −(1/8) $\log_2$(1/8) − (1/8) $\log_2$(1/8) − (6/8) $\log_2$(6/8)
    = 0.375 + 0.375 + 0.3113
    = 1.0613

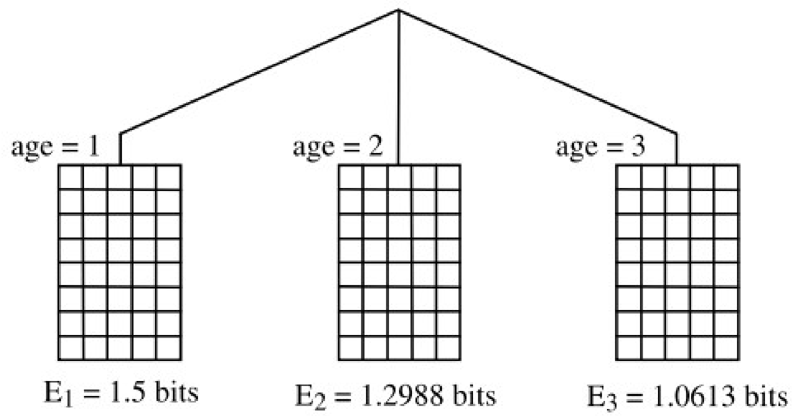
Value of attribute				Class
age	specRx	astig	tears	
3	1	1	1	3
3	1	1	2	3
3	1	2	1	3
3	1	2	2	1
3	2	1	1	3
3	2	1	2	2
3	2	2	1	3
3	2	2	2	3

Training Set 2 for *lens24* Example

Using Entropy for Attribute Selection

- Now, the values E_1, E_2 and E_3 need to be weighted by the proportion of the original instances in each of the three subsets.
- In this case all the weights are the same, i.e. 8/24.
- If the average entropy of the three training sets produced by splitting on attribute ***age*** is denoted by E_{new} ,
  - then E_{new} = (8/24)E_1+(8/24) E_2 +(8/24) E_3 = 1.2867 bits.
- We know ***Information Gain = E_{start} – E_{new}***
- So the ***information gain*** from splitting on attribute ***age*** is 1.3261 − 1.2867 = 0.0394 bits (see figure in the next slide).
- The 'entropy method' of attribute selection is to choose to split on the attribute that gives the greatest reduction in (average) *entropy*, i.e. the one that maximises the value of *Information Gain*.
  - This is equivalent to minimizing the value of E_{new} as E_{start} is fixed.

Using Entropy for Attribute Selection



Initial Entropy = 1.3261 bits
Average Entropy of Subsets = 1.2867 bits
Information Gain = 1.3261–1.2867 = 0.0394 bits

Information Gain for Splitting on Attribute ***age***

Maximising Information Gain

- The values of $\boldsymbol{E_{new}}$ and ***Information Gain*** for splitting on each of the four attributes *age*, *specRx*, *astig* and *tears* are as follows:

attribute *age*
$E_{new} = 1.2867$
Information Gain $= 1.3261 - 1.2867 = 0.0394$ bits

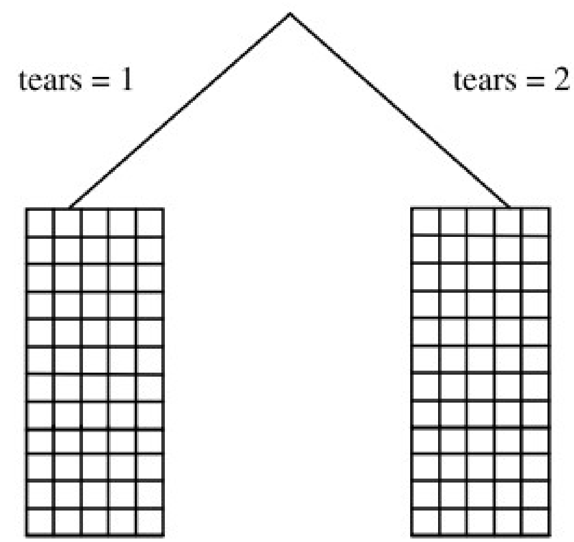
attribute *specRx*
$E_{new} = 1.2866$
Information Gain $= 1.3261 - 1.2866 = 0.0395$ bits

attribute *astig*
$E_{new} = 0.9491$
Information Gain $= 1.3261 - 0.9491 = 0.3770$ bits

attribute *tears*
$E_{new} = 0.7773$
Information Gain $= 1.3261 - 0.7773 = 0.5488$ bits

Maximising Information Gain

- Thus, the largest value of ***Information Gain*** (and the smallest value of the new entropy $\boldsymbol{E_{new}}$) is obtained by splitting on attribute ***tears*** (see Figure).
- The process of splitting on nodes is repeated for each branch of the evolving decision tree, terminating when the subset at every leaf node has *entropy* zero.



Splitting on Attribute *tears*

Exercises

- Calculate the following for the *degrees* dataset:
  - the initial entropy $\boldsymbol{E_{start}}$
  - the weighted average entropy $\boldsymbol{E_{new}}$ of the training (sub)sets resulting from splitting on each of the attributes *SoftEng*, *Arin*, *HCI*, *CSA* and *Project* in turn and the corresponding value of *Information Gain* in each case.
- Using these results, verify that the attribute that will be chosen by the TDIDT algorithm for the first split on the data using the entropy selection criterion is *SoftEng*.

SoftEng	ARIN	HCI	CSA	Project	Class
A	B	A	B	B	SECOND
A	B	B	B	A	FIRST
A	A	A	B	B	SECOND
B	A	A	B	B	SECOND
A	A	B	B	A	FIRST
B	A	A	B	B	SECOND
A	B	B	B	B	SECOND
A	B	B	B	B	SECOND
A	A	A	A	A	FIRST
B	A	A	B	B	SECOND
B	A	A	B	B	SECOND
A	B	B	A	B	SECOND
B	B	B	B	A	SECOND
A	A	B	A	B	FIRST
B	B	B	B	A	SECOND
A	A	B	B	B	SECOND
B	B	B	B	B	SECOND
A	A	B	A	A	FIRST
B	B	B	A	A	SECOND
B	B	A	A	B	SECOND
B	B	B	B	A	SECOND
B	A	B	A	B	SECOND
A	B	B	B	A	FIRST
A	B	A	B	B	SECOND
B	A	B	B	B	SECOND
A	B	B	B	B	SECOND

Classes
FIRST, SECOND
SoftEng
A,B
ARIN
A,B
HCI
A,B
CSA
A,B
Project
A,B

The *degrees* Dataset

Reference

- Max Bramer, “Chapter 5: Decision Tree Induction: Using Entropy for Attribute Selection”, *Principles of Data Mining* (4th Edition).