## Data Mining I

Classificatins

What is Data Mining?

* + Problem: the gap between the generation of data and our understanding of it
  + Definition: The process of extracting information from large databases and using it to make decisions
  + Knowledge Discovery in Databases (KDD)
* Methods of data mining
  + Prediction methods
  + Description methods
* Basic tasks
  + Classification (predictive)
  + Regression (predictive)
  + Clustering (descriptive)
  + Association rule discovery (descriptive)
* Examples
  + Beer and nappies
  + Advertisement
* Data Gathering, e.g., data warehousing
* Data preparation and Cleansing
* Pattern Extraction and Discovery (mining)
* Visualisation of the Data
* Analysis and Evaluation of Results
* Deductive learning
  + An Example: England  Scotland  Wales
* Inductive learning
  + An example: In a town, part strange  all strange
  + Supervised Learning: Input – Output
  + Unsupervised Learning: Input only
  + Reinforcement Learning: Input – Output (evaluative

output only)

* + For instance, the teacher  overall grade | detailed marks and correct answers for individual questions separately
* Classification – Predict the outcome of an

experiment with a nominal target class attribute

* + An Example: Yes/No: Is the weather suitable for playing tennis?
* Regression – also a prediction task, but the value

of the target class attribute is numerical

* + An example: currency exchange rate: how will the

Pound/EUR exchange rate develop?

* Classification rules:

IF buy\_time in December

AND cost > 500

AND type\_of\_item = electronics AND location = overseas

AND ...etc...

THEN possibly\_fraudulent = yes

# An Example

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Outlook** | **Temperature** | **Humidity** | **Windy** | **Play** |
| 1 | Sunny | Hot | High | False | No |
| 2 | Sunny | Hot | High | True | No |
| 3 | Overcast | Hot | High | False | Yes |
| 4 | Rain | Mild | High | False | Yes |
| 5 | Rain | Cool | Normal | False | Yes |
| 6 | Rain | Cool | Normal | True | No |
| 7 | Overcast | Cool | Normal | True | Yes |
| 8 | Sunny | Mild | High | False | No |
| 9 | Sunny | Cool | Normal | False | Yes |
| 10 | Rain | Mild | Normal | False | Yes |
| 11 | Sunny | Mild | Normal | True | Yes |
| 12 | Overcast | Mild | High | True | Yes |
| 13 | Overcast | Hot | Normal | False | Yes |
| 14 | Rain | Mild | High | True | No |

## Looking for Interest Patterns – Cont’d (2)

* A minimum coverage: a minimum number of instances
* Rule’s Accuracy: N1/N2
* **N1: the number of cases for THEN-part**
* **N2: the number of cases for IF-part**

**humidity = normal and windy = false and play = yes**

* **Consider a possible rule:**

**if humidity = normal and windy = false then play = yes**

**accuracy of 100%**

* **another possible rule: if humidity = normal**

**then windy = false and play = yes**

**accuracy of only 4/7 = 57.142%**

# Algorithms

**OneR (one-attribute-rule) algorithm - find one attribute to use that makes fewest prediction errors.**

**For example, consider outlook:**

**if outlook = sunny then play = no .. 2 errors in 5 if outlook = overcast then play = yes ..0 errors in 4 if outlook = rainy then play = yes .. 2 errors in 5**

**! a total of 4 errors in 14 cases**

**if humidity = high then play = no .. 3 errors in 7 if humidity = normal then play = yes .. 1 error in 7**

**! Also a total of 4 errors in 14 cases**

**Note: minimum coverage: 3**

# Description of OneR

For each attribute A:

For each value V of that attribute, create a rule:

* 1. count how often each class appears
  2. find the most frequent class, c
  3. make a rule "if A=V then C=c"

Calculate the error rate of this rule

Pick the attribute whose rules produce the lowest

error rate

**ID3 algorithm - *An example* (Fruit)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Conclusion | Skin | Colour | Size | Flesh |
| Safe | Hairy | Brown | Large | Hard |
| Safe | Hairy | Green | Large | Hard |
| Dangerous | Smooth | Red | Large | Soft |
| Safe | Hairy | Green | Large | Soft |
| Safe | Hairy | Red | Small | Hard |
| Safe | Smooth | Red | Small | Hard |
| Safe | Smooth | Brown | Small | Hard |
| Dangerous | Hairy | Green | Small | Soft |
| Dangerous | Smooth | Green | Small | Hard |
| Safe | Hairy | Red | Large | Hard |
| Safe | Smooth | Brown | Large | Soft |
| Dangerous | Smooth | Green | Small | Soft |
| Safe | Hairy | Red | Small | Soft |
| Dangerous | Smooth | Red | Large | Hard |
| Safe | Smooth | Red | Small | Hard |
| Dangerous | Hairy | Green | Small | Hard |

## ID3 Algorithm - Cont'd

Each instance can be viewed as a rule, the second line is :

if skin = hairy

and colour = green and size = large and flesh = hard

then conclusion = safe

*This kind of rules is not general*.

The decision tree is of the general form:

if attribute1 = value1 then <subtree 1> else if attribute1 = value2 then <subtree 2> else if ...

...........

else if attribute1 = valueN then <subtree N>

* This corresponds directly to a set of rules, with as many rules as there are leaf nodes in the whole tree
* Each rule is a tracing out of the path from the top of the tree to a leaf node

**[C1, C2, …, C16]**

**[10+, 6-]**

***Colou****r*

***Green***

***Brow***

***Red***

**[C2, C4, C8, C9, C12, C16] [2+, 4-]**

**Safe**

**[C3,C5,C6,C10,C13,C14,C15] [5+, 2-]**

***Size***

***Size***

***Large***

***Small***

**Safe Dangerous**

**Safe**

***Small***

***Large***

**[C3,C10,C14]**

**[1+, 2-]**

***Skin***

***Smooth***

***Hairy***

**Dangerous**

**Safe**

## ID3 - Cont'd

Decision trees do classification:

* Classify examples by sorting them down the tree from the root node to some leaf notes
* Learned function represented by tree
* Each node in tree is tested on some attribute of an instance
* Branches represent values of attributes
* Follow tree from root to leaves for output value

Example from *Conclusion* decision tree:

* A path for the negative decision (that is

*Conclusion* = *Dangerous*)

(*Colour = Red, Size = Large, Skin = Smooth*)

* conjunction and disjunction

(*Colour = Green*  *Size = Small*)

 (*Colour = Red*  *Size = Large*  *Skin = Smooth*)

* It is not difficult to write such rules in the If-Then form

How do you determine which attribute best classifies data?

Entropy: Given a target attribute *C*, which can have values c1, c2, …, cn (in the example above, n = 2 and the values are `safe' or `dangerous') and a certain attribute *A*,

which can take values a1, …, am. Then the entropy of

attribute *A* with respect to the target attribute *C* is

*Entropy* 

*m*



*j* 1

*n*

*p*(*aj* )

*i* 1

*p*(*ci*

| *aj* ) log2

*p*(*ci*

| *aj* )

where *p*(*ci*|*aj*): the probability that *ci*, given *aj*

*p*(*Conclusion = safe* | *Skin = Hairy*) = 6/8 = 3/4.

Entropy can provide the following:

* Entropy is a measure of 'degree of doubt'
* The higher it is, the more doubt there is about

the possible conclusions

* The attribute which has the lowest entropy is the

most useful determiner

## Fruit Example -- Cont'd

* Four attributes: *Skin*, *Colour*, *Size* and *Flesh*
* The target attribute: *Conclusion*

• *Set\_Cases* = {C1, C2, …, C16} = [10+, 6-]

*Step 1*: choose the most useful determiner:

Consider the attribute *Size*. We have

*p*(*safe|large*) = 5/7 *p*(*dangerous|large*) = 2/7

*p*(*large*) = 7/16 *p*(*safe|small*) = 5/9

*p*(*dangerous|small*) = 4/9 *p*(*small*) = 9/16

**Thus, the entropy for attribute *Size* with respect to *Conclusion* is**

**-7/16(5/7log(5/7) + 2/7(log(2/7)) - 9/16(5/9log(5/9) + 4/9log(4/9)) = 0.9350955…**

After all the calculations, *Colour* has the smallest entropy. So it can be put at the root of the tree

## Fruit Example - Cont'd

***Step 2*: partition *Set\_Cases* according to the possible values for the root node *Colour*, and apply the same process to each leaf node.**

**[C1, C2, …, C16]**

**[10+, 6-]**

***Colour***

***Green***

***Brow***

***Red***

C2,C4,C8,C9,C12,C16

[2+, 4-]

**Safe**

[C3,C5,C6,C10,C13,C14,C15] [5+, 2-]

*?*

*?*

## Fruit Example - Cont'd

Result In the form of set of rules: if colour = brown

then conclusion = safe

if colour = green and size = large

then conclusion = safe

if colour = green and size = small

then conclusion = dangerous

if colour = red and size = small

then conclusion = safe

if colour = red and size = large and skin = smooth

then conclusion = dangerous

if colour = red and size = large and skin = hairy

then conclusion = safe

## Summary of ID3 Algorithm

1. For each attribute, compute its entropy with

respect to the conclusion (the target attribute)

1. Select the attribute (say A) with lowest entropy
2. Divide the data into separate sets so that within a set, A has a fixed value (eg *Colour=green* in one set, *Colour=brown* in another, etc)
3. Build a tree with branches:

if *A=a1* then ... (subtree1) if *A=a2* then ... (subtree2)

...etc...

1. For each subtree, repeat this process from step 1
2. At each iteration, one attribute gets removed from consideration. The process stops when there are no attributes left to consider, or when all the data being considered in a subtree have the same value for the conclusion (e.g. they all say *Conclusion=safe*).

* It generates a detailed decision tree.
* With training data provided, it is always able to generate a tree.
* it is easily implemented
* The output is easily to be understood and

interpreted

* The process is simple process
* Its running time increases only linearly with the

complexity of the problem

* Wholly spurious correlations are possible, since the algorithm takes no account of any meaning that the data it works on may have
* The algorithm considers just one attribute at a time
* When inducing rules from large sets of examples in which there are a large number of possible outcomes, then the algorithm can be very sensitive to apparently trivial changes in the set of examples
* The algorithm cannot generate uncertain rules or handle uncertain data

## ID3 - Limitations

Consider the data at right

|  |  |  |
| --- | --- | --- |
| *Outcome* | X | Y |
| Yes | 3 | 3 |
| No | 2 | 1 |
| Yes | 4 | 4 |
| No | 2 | 4 |
| No | 1 | 3 |
| Yes | 1 | 1 |
| Yes | 2 | 2 |
| No | 2 | 3 |

A rule:

IF *X = Y*

THEN *Outcome = yes*

ELSE *Outcome = no*

# References

* I. Witten, E. Frank and M. Hall (2011)

*Data Mining: Practical Machine Learning Tools*

*and Techniques*,

Morgan Kaufmann Publishers.

* M. Berthold, C. Borgelt, F. Hoppner and F. Klawonn

Guide to Intelligent Data Analysis, Spriner-Verlag,

2010

* + Chapter 8, Section 8.1, 8.4
* <http://www.kdnuggets.com/>