Data Mining
Classification

Kevin Swingler

What is Classification?

• Assigning an object to a certain class based on its similarity to previous examples of other objects
• Can be done with reference to original data or based on a model of that data
• E.g: Me: “It’s round, green, and edible”
      You: “It’s an apple!”

Usual Examples

• Classifying transactions as genuine or fraudulent – e.g. credit card usage, insurance claims, cell phone calls
• Classifying prospects as good or bad customers
• Classifying engine faults by their symptoms

Certainty

• As with most data mining solutions, a classification usually comes with a degree of certainty.
• It might be the probability of the object belonging to the class or it might be some other measure of how closely the object resembles other examples from that class

Techniques

• Non-parametric, e.g. K nearest neighbour
• Mathematical models, e.g. neural networks
• Rule based models, e.g. decision trees

Predictive / Definitive

• Classification may indicate a propensity to act in a certain way, e.g. A prospect is likely to become a customer. This is predictive.
• Classification may indicate similarity to objects that are definitely members of a given class, e.g. small, round, green = apple
Simple Worked Example

- Risk of making a claim on a motor insurance policy
  - This is a predictive classification – they haven’t made the claim yet, but do they look like other people who have?
  - To keep it simple, let’s look at just age and gender

The Data

<table>
<thead>
<tr>
<th>Age</th>
<th>Gender</th>
<th>Claim?</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>Female</td>
<td>No</td>
</tr>
<tr>
<td>31</td>
<td>Male</td>
<td>No</td>
</tr>
<tr>
<td>27</td>
<td>Male</td>
<td>No</td>
</tr>
<tr>
<td>26</td>
<td>Male</td>
<td>Yes</td>
</tr>
<tr>
<td>29</td>
<td>Female</td>
<td>No</td>
</tr>
<tr>
<td>32</td>
<td>Male</td>
<td>No</td>
</tr>
<tr>
<td>46</td>
<td>Male</td>
<td>No</td>
</tr>
<tr>
<td>45</td>
<td>Male</td>
<td>No</td>
</tr>
<tr>
<td>55</td>
<td>Male</td>
<td>No</td>
</tr>
<tr>
<td>25</td>
<td>Female</td>
<td>No</td>
</tr>
<tr>
<td>38</td>
<td>Female</td>
<td>No</td>
</tr>
<tr>
<td>21</td>
<td>Female</td>
<td>No</td>
</tr>
<tr>
<td>38</td>
<td>Female</td>
<td>No</td>
</tr>
<tr>
<td>42</td>
<td>Male</td>
<td>No</td>
</tr>
<tr>
<td>29</td>
<td>Male</td>
<td>No</td>
</tr>
<tr>
<td>37</td>
<td>Male</td>
<td>No</td>
</tr>
<tr>
<td>40</td>
<td>Female</td>
<td>No</td>
</tr>
</tbody>
</table>

K-Nearest Neighbour

- Performed on raw data
- Count number of other examples that are close
- Winner is most common

New person to classify

Rule Based

- If Gender = Male and Age < 30 then Claim
- If Gender = Male and Age > 30 then No Claim
- Etc …

New person to classify

Decision Trees

- A good automatic rule discovery technique is the decision tree
- Produces a set of branching decisions that end in a classification
- Works best on nominal attributes – numeric ones need to be split into bins

A Decision Tree

Note: Not all attributes are used in all decisions
Making a Classification

- Each node represents a single variable
- Each branch represents a value that variable can take
- To classify a single example, start at the top of the tree and see which variable it represents
- Follow the branch that corresponds to the value that variable takes in your example
- Keep going until you reach a leaf, where your object is classified!

Tree Structure

- There are lots of ways to arrange a decision tree
- Does it matter which variables go where?
- Yes:
  - You need to optimise the number of correct classifications
  - You want to make the classification process as fast as possible

A Tree Building Algorithm

- Divide and Conquer:
  - Choose the variable that is at the top of the tree
  - Create a branch for each possible value
  - For each branch, repeat the process until there are no more branches to make (i.e. stop when all the instances at the current branch are in the same class)
  - But how do you choose which variable to split?

The ID3 Algorithm

- Split on the variable that gives the greatest information gain
- Information can be thought of as a measure of uncertainty
- Information is a measure based on the probability of something happening

Information Example

- If I pick a random card from a deck and you have to guess what it is, which would you rather be told:
  - It is red (which has a probability of 0.5)
  - or
  - It is a picture card (which has a probability of $4/13 = 0.31$)

Calculating Information

- The information associated with a single event:
  $I(e) = -\log(p_e)$
  where $p_e$ is the probability of event $e$ occurring
- $I$(Red) = $-\log(0.5) = 1$
- $I$(Picture card) = $-\log(0.31) = 1.7$
Average Information

- The weighted average information across all possible values of a variable is called **Entropy**.
- It is calculated as the sum of the probability of each possible event times its information value:

$$H(X) = - \sum P(x_i) \log_2 P(x_i)$$

where \( \log \) is the base 2 log.

Entropy of IsPicture?

- \( I(\text{Picture}) = -\log(4/13) = 1.7 \)
- \( I(\text{Not Picture}) = -\log(9/13) = 0.53 \)
- \( H = 4/13*1.7 + 9/13*0.53 = 0.89 \)
- **Entropy** – \( H(X) \) – is a measure of uncertainty in variable \( X \)
- The more even the distribution of \( X \) becomes, the higher the entropy gets:

Unfair Coin Entropy

[Graph showing unfair coin entropy]

Conditional Entropy

- We now introduce conditional entropy: \( H(\text{outcome} | \text{known}) \)
- The uncertainty about the outcome, given that we know \( \text{known} \)

Information Gain

- If we know \( H(\text{Outcome}) \)
- And we know \( H(\text{Outcome} | \text{Input}) \)
- We can calculate how much \( \text{Input} \) tells us about \( \text{Outcome} \) simply as:

$$H(\text{Outcome}) - H(\text{Outcome} | \text{Input})$$

- This is the information gain of \( \text{Input} \)

Picking the Top Node

- ID3 picks the top node of the network by calculating the information gain of the output class for each input variable, and picks the one that removes the most uncertainty
- It creates a branch for each value the chosen variable can take
Adding Branches

• Branches are added by making the same information gain calculation for data defined by the location on the tree of the current branch
• If all objects at the current leaf are in the same class, no more branching is needed
• The algorithm also stops when all the data has been accounted for

Other Classification Methods

• You will meet a certain type of neural network in a later lecture – these too are good at classification
• There are many, many, many other methods for building classification systems