How to Build a CBR System
Assoc. Prof. Ian Watson

Fortunately this is relatively easy
Most CBR systems use the k-NN algorithm
k-Nearest Neighbour Algorithm

Nearest Neighbour

\[
\text{Similarity}(T, S) = \sum_{i=1}^{n} f(T_i, S_i) \times w_i
\]

where:
- \( T \) is the target case
- \( S \) is the source case
- \( n \) is the number of attributes in each case
- \( i \) is an individual attribute from 1 to \( n \)
- \( f \) is a similarity function for attribute \( i \) in cases \( T \) and \( S \)
- \( w \) is the importance weighting of attribute \( i \)
Nearest Neighbour

- imagine a decision with two factors that influence it
- should you grant a person a loan?
  - net monthly income
  - monthly loan repayment

Nearest Neighbour

- these factors can be used as axes for a graph

Nearest Neighbour

- a previous loan can be plotted against these axes
Nearest Neighbour

- and a second loan

- and more loans

- and even more loans
Nearest Neighbour

- past cases (loans) may form clusters

![Diagram showing a scatter plot with net monthly income on the x-axis and monthly loan repayment on the y-axis. The plot illustrates clusters of data points representing good loans.]

Nearest Neighbour

- past cases (loans) may tend to form clusters

![Diagram showing a scatter plot with net monthly income on the x-axis and monthly loan repayment on the y-axis. The plot illustrates clusters of data points representing good loans and bad loans.]
Nearest Neighbour

- a new loan prospect can be plotted on the graph

net monthly income

monthly loan repayment

new case

and the distance to its nearest neighbours calculated
Nearest Neighbour

- and the distance to its nearest neighbours calculated

Net monthly income

Monthly loan repayment

Nearest Neighbour

- and the distance to its nearest neighbours calculated

Net monthly income

Monthly loan repayment

Nearest Neighbour

- the best matching past case is the closest
Nearest Neighbour

- the best matching past case is the closest

- this suggests a precedent

- the loan will be successful
Nearest Neighbour

- over time the prediction can be validated

Net monthly income vs. monthly loan repayment

It was a good loan

Nearest Neighbour

- the system is learning to differentiate good and bad loans better

Net monthly income vs. monthly loan repayment
Nearest Neighbour

- as more cases are acquired its performance improves

Euclidean Distance

\[ X^2 + Y^2 = H \]

The weight of the X axis (income) is increased
Nearest Neighbour

- Requires a unique similarity function for each attribute or feature (not always a trivial problem) – local similarity \( f(T, S) \)
- Local similarities are combined to give a global similarity – \( \text{sim}(T, S) \)
- \( k \)-NN Requires every feature of the query to be compared to every feature of every instance/case at run-time
- Not very efficient 😞

Nearest Neighbour

- distance weighted \( k \)-Nearest neighbour is a highly effective algorithm for many practical problems robust to noisy data if the training set is large enough
- bias is that the classification of an instance is most similar to other instances that are nearby in Euclidean distance
- But then again that’s the point

Nearest Neighbour

- because distance is calculated on all attributes - irrelevant attributes are a problem - curse of dimensionality
- some approaches weight attributes to overcome this - stretching the Euclidean space
- alternatively eliminate the least relevant attributes
Nearest Neighbour

- could locally stretch an axis...but more degrees of freedom...so more chance of overfitting...useful if problem space is not uniform...problem of over fitting
- much less common, but it is used in CBR
- efficient indexing of instances can be done with kd-trees (we’ll discuss later)
- possible to pre-compute a position of each instance in the Euclidean space then simply position query in the space

Enough Theory 😐

- How do I build a CBR system?
- Let’s consider an example
- Estimating the price of used cars
  - Cases have a description
    - The features that describe a car
  - Cases have an outcome/solution
    - The price the car sold for

1. Case Vocabulary

- Features used in retrieval should be predictive of the case outcome
  - Manufacturer e.g. Mazda
  - Model e.g. SP3
  - Engine size e.g. 2,500 cc
  - Body type e.g. 5 door hatch
  - Age e.g. 2005
  - Colour e.g. silver
  - …
2. Case Vocabulary

- Some features may not be used in retrieval but could be useful for other purposes
- Photograph

3. Case Vocabulary - outcome

|------------|---------------------|----------|-----------------|------------------|--------|--------------|--------------|

| Case ID 003 | Manufacturer: Mercedes | Model: S Class | Engine Size: 3500 | Body: 4 Door Sedan | Age: 2006 | Colour: Silver | Price: $35000 |
5. Local Similarity metrics

- For each feature $i$ used in retrieval
- Build a local similarity metric
- Manufacturer
- Model
- Engine Size
- Body
- Age
- Colour

This is the hardest part!!!
5. Local Similarity metrics

- Engine Size
  - A numeric feature
  - This is easy 😊
  - Consider the likely min and max values
  - 500cc to 7000cc
  - The feature Range is 7000 – 500 = 6500
  - $\text{sim}(f) = \frac{\text{Range} - \text{Diff}}{\text{Range}}$
  - (This normalises the result between 0 & 1)

- A simple linear function
- But STOP
- Isn’t a larger engine always better???
- More is perfect!!!

<table>
<thead>
<tr>
<th>Start Case</th>
<th>Target Case</th>
<th>Diff</th>
<th>Range</th>
<th>Sims</th>
<th>Sim-Normalised</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000cc</td>
<td>2000cc</td>
<td>1000</td>
<td>6500</td>
<td>0.88</td>
<td></td>
</tr>
<tr>
<td>500cc</td>
<td>1000cc</td>
<td>5000</td>
<td>6500</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>2000cc</td>
<td>3000cc</td>
<td>1000</td>
<td>6500</td>
<td>0.86</td>
<td></td>
</tr>
<tr>
<td>500cc</td>
<td>2000cc</td>
<td>1500</td>
<td>6500</td>
<td>0.54</td>
<td></td>
</tr>
</tbody>
</table>

Symmetric Similarity

- More is perfect
- Less is perfect
5. Local Similarity metrics

- Engine Size
  - Not necessarily a linear relationship
  
  ![Graph showing polynomial relationship between Engine Size and Symmetric Similarity]

- Not necessarily a linear relationship

- Symmetric Similarity
  - more is perfect: polynomial

5. Local Similarity metrics

- Body
  - Symbolic feature – treat like Model ???
  - 5 door hatch, 4 door sedan, 3 door coupe, 2 door sports,…
  - Will someone who wants a 2 door sport really be happy with a 3 door coupe ????
  - Not easy to put this into an ordered list
5. Local Similarity metrics

- **Body**
  - A decision table

<table>
<thead>
<tr>
<th>Car Type</th>
<th>4 door sedan</th>
<th>3 door coupe</th>
<th>2 door hatch</th>
<th>2 door sports</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distance (miles)</td>
<td>3.15</td>
<td>1.95</td>
<td>3.75</td>
<td>0.28</td>
</tr>
<tr>
<td>Age (years)</td>
<td>2.15</td>
<td>2.75</td>
<td>1.90</td>
<td>0.40</td>
</tr>
<tr>
<td>Colour spectrum</td>
<td>3.00</td>
<td>3.85</td>
<td>3.00</td>
<td>1.85</td>
</tr>
</tbody>
</table>

- 4 door sedan -> 3 door coupe = 0.5
- 2 door sports -> any other type = 0.0
- decision tables can model complex asymmetric similarities

- **Age**
  - Numeric feature, this is easy treat like Engine Size
  - Max age for a car???
  - In theory 100 years plus
  - But in practise say 20 years is Max Range
  - \( sim(f) = (\text{Range} – \text{diff})/\text{Range} \)

- **Colour**
  - Symbolic feature
  - Could use frequencies in colour spectrum

  Scientific, but does it model peoples' colour preferences???
  - Perhaps a hierarchy
5. Local Similarity metrics

- Colour
  - any colour 0.25
  - dark colours 0.75 light colours 0.75
  - black -> silver = 0.25
  - black -> blue = 0.75
  - Actually this isn't very good
  - Turns out colour is really hard to model 😞

6. Global similarity

- To get a similarity metric for a case against any other
  - Compute each local similarity
  - Multiply local similarity by feature weight
  - Sum the results (and normalise)

\[ \text{Similarity}(T, S) = \sum_{j=1}^{n} f(T_j, S_j) \times w_j \]

7. Feature Weights

- Usually set globally
- But can be over-ridden by a user at run-time
  - Manufacturer – very important \( w = 10.0 \)
  - Model – less important \( w = 1.0 \)
  - Engine Size – important \( w = 5.0 \)
  - Body – important \( w = 5.0 \)
  - Age – important \( w = 5.0 \)
  - Colour – less important \( w = 1.0 \)
- May take trial and error to approximate
8. We’re Almost Done 😊

Let’s go

Case ID 007
Manufacturer: BMW
Model: 320
Engine Size: 2000
Body: 3 Door Coupe
Age: 2004
Colour: Blue
Price: $???

Case ID 001
Manufacturer: Mazda
Model: SP3
Engine Size: 2500
Body: 5 Door Hatch
Age: 2005
Colour: Silver
Price: $25000

Case ID 002
Manufacturer: Ford
Model: Falcon XR6
Engine Size: 3000
Body: 4 Door Sedan
Age: 1995
Colour: Black
Price: $15000

Case ID 004
Manufacturer: Ford
Model: Focus
Engine Size: 1600
Body: 5 Door Hatch
Age: 2006
Colour: Black
Price: $14000

Case ID 001
Manufacturer: Mazda
Model: SP3
Engine Size: 2500
Body: 5 Door Hatch
Age: 2005
Colour: Silver
Price: $25000

Sim = 0.72

Repeat for every case in case base and sort cases by similarity

9. Result !!!!

Case ID 002
Manufacturer: Ford
Model: Falcon XR6
Engine Size: 3000
Body: 4 Door Sedan
Age: 1995
Colour: Black
Price: $15000

Sim = 0.54

Case ID 004
Manufacturer: Ford
Model: Focus
Engine Size: 1600
Body: 5 Door Hatch
Age: 2006
Colour: Black
Price: $14000

Sim = 0.62

Case ID 001
Manufacturer: Mazda
Model: SP3
Engine Size: 2500
Body: 5 Door Hatch
Age: 2005
Colour: Silver
Price: $25000

Sim = 0.72

Case ID 006
Manufacturer: Alpha Romeo
Model: Spider
Engine Size: 2500
Body: 2 Door Sports
Age: 2000
Colour: Red
Price: $19950

Sim = 0.72

Case ID 005
Manufacturer: BMW
Model: 330
Engine Size: 3000
Body: 3 Door Coupe
Age: 2005
Colour: Black
Price: $29500

Sim = 0.98

Case ID 005
Manufacturer: BMW
Model: 330
Engine Size: 3000
Body: 3 Door Coupe
Age: 2005
Colour: Black
Price: $29500
Summary

- CBR using k-NN is easy to implement
- Identify predictive case features
- Create a local similarity metric for each feature (the hardest part)
- Decide upon feature weights
- At retrieval compare features of query case to every feature of every source case
- Sum local features to get global similarity for each case
- Sort cases by similarity
- Select k best matching cases to inform result
- Adapt result (if necessary)

---

Result!!!

- Use the price from the best matching case

<table>
<thead>
<tr>
<th>Case ID 005</th>
<th>Sim = 0.98</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturer: BMW</td>
<td></td>
</tr>
<tr>
<td>Model: 330</td>
<td></td>
</tr>
<tr>
<td>Engine Size: 3000</td>
<td></td>
</tr>
<tr>
<td>Body: 3 Door Coupe</td>
<td></td>
</tr>
<tr>
<td>Age: 2005</td>
<td></td>
</tr>
<tr>
<td>Colour: Black</td>
<td></td>
</tr>
<tr>
<td>Price: $29500</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Case ID 00?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturer: BMW</td>
</tr>
<tr>
<td>Model: 320</td>
</tr>
<tr>
<td>Engine Size: 2000</td>
</tr>
<tr>
<td>Body: 3 Door Coupe</td>
</tr>
<tr>
<td>Age: 2004</td>
</tr>
<tr>
<td>Colour: Blue</td>
</tr>
<tr>
<td>Price: $29500</td>
</tr>
</tbody>
</table>

---

Adapt the Result!!!

- Consider the differences between the cases (engine size is less and car is older)

<table>
<thead>
<tr>
<th>Case ID 00?</th>
<th>Sim = 0.98</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturer: BMW</td>
<td></td>
</tr>
<tr>
<td>Model: 320</td>
<td></td>
</tr>
<tr>
<td>Engine Size: 2000</td>
<td></td>
</tr>
<tr>
<td>Body: 3 Door Coupe</td>
<td></td>
</tr>
<tr>
<td>Age: 2004</td>
<td></td>
</tr>
<tr>
<td>Colour: Blue</td>
<td></td>
</tr>
<tr>
<td>Price: $29500</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Case ID 005</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturer: BMW</td>
</tr>
<tr>
<td>Model: 330</td>
</tr>
<tr>
<td>Engine Size: 3000</td>
</tr>
<tr>
<td>Body: 3 Door Coupe</td>
</tr>
<tr>
<td>Age: 2005</td>
</tr>
<tr>
<td>Colour: Black</td>
</tr>
<tr>
<td>Price: $26000</td>
</tr>
</tbody>
</table>