Case-Based Reasoning

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

How to Build a CBR System

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

Nearest Neighbour

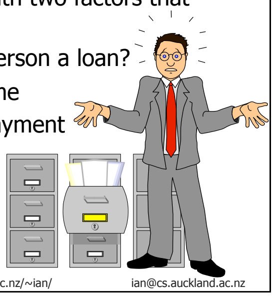
$$Similarity(T,S) = \sum_{i=1}^{n} f(T_i, S_i) \times w_i$$

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* and
w is the importance weighting of attribute *i*

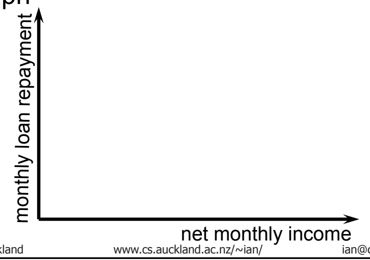
Nearest Neighbour

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



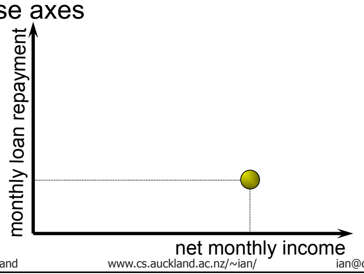
Nearest Neighbour

- these factors can be used as axes for a graph



Nearest Neighbour

- a previous loan can be plotted against these axes

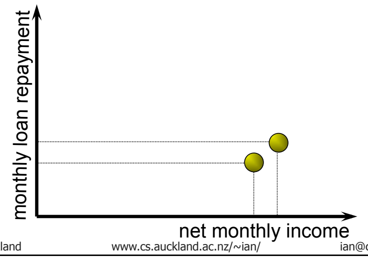


7

The University of Auckland

Nearest Neighbour

- and a second loan



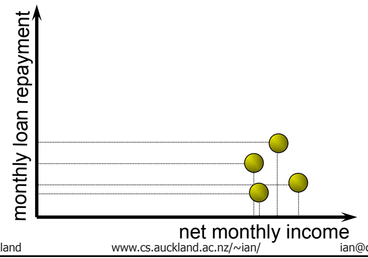
 www.cs.auckland.ac.nz/~ian/ ian@cs.auckland.ac.nz

8

The University of Auckland

Nearest Neighbour

- and more loans



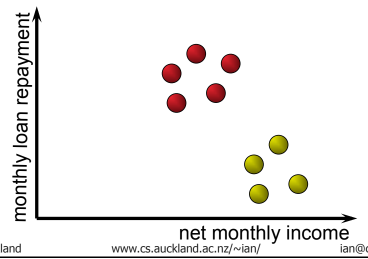
 www.cs.auckland.ac.nz/~ian/ ian@cs.auckland.ac.nz

9

The University of Auckland

Nearest Neighbour

- and even more loans



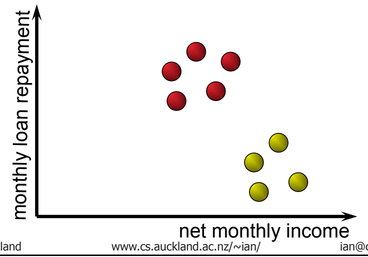
 www.cs.auckland.ac.nz/~ian/ ian@cs.auckland.ac.nz

10

The University of Auckland

Nearest Neighbour

- past cases (loans) may form clusters

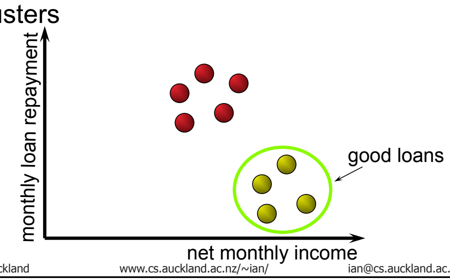


11

The University of Auckland

Nearest Neighbour

- past cases (loans) may tend to form clusters

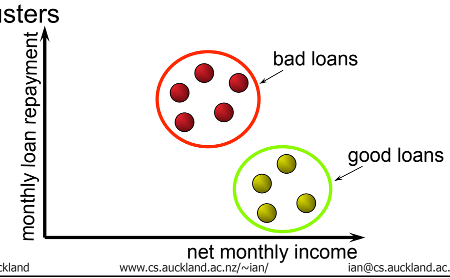


12

The University of Auckland

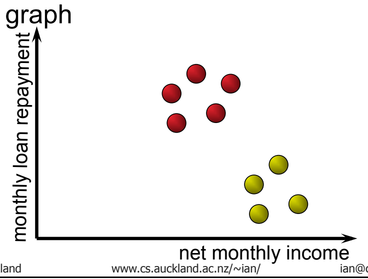
Nearest Neighbour

- past cases (loans) may tend to form clusters



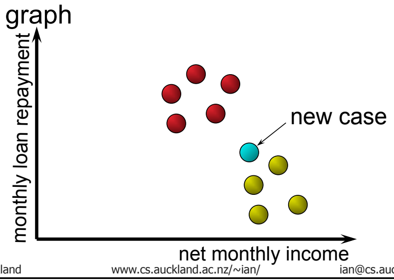
Nearest Neighbour

- a new loan prospect can be plotted on the graph



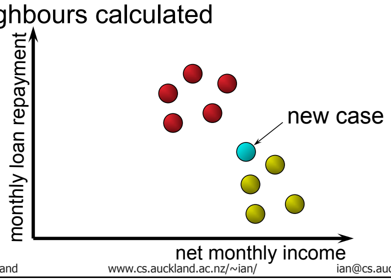
Nearest Neighbour

- a new loan prospect can be plotted on the graph



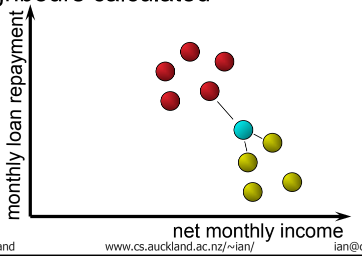
Nearest Neighbour

- and the distance to its nearest neighbours calculated



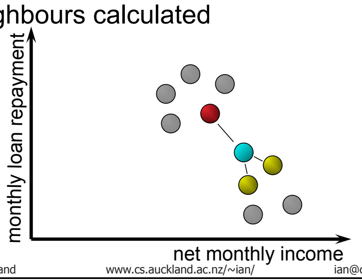
Nearest Neighbour

- and the distance to its nearest neighbours calculated



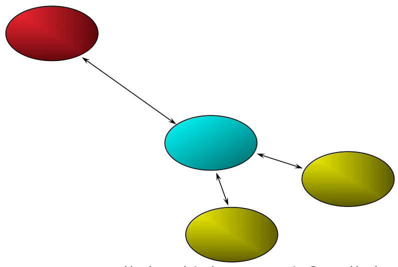
Nearest Neighbour

- and the distance to its nearest neighbours calculated



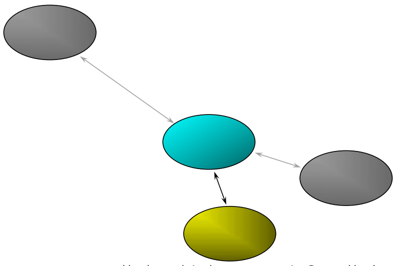
Nearest Neighbour

- the best matching past case is the closest



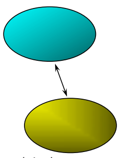
Nearest Neighbour

- the best matching past case is the closest



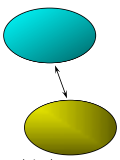
Nearest Neighbour

- this suggests a precedent



Nearest Neighbour

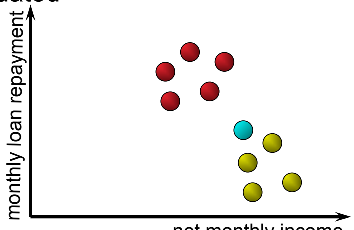
- this suggests a precedent
- the loan will be successful



22

Nearest Neighbour

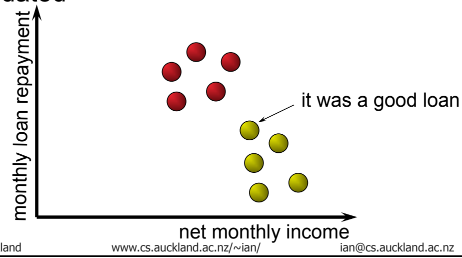
- over time the prediction can be validated



23

Nearest Neighbour

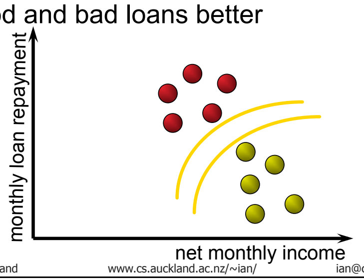
- over time the prediction can be validated



24

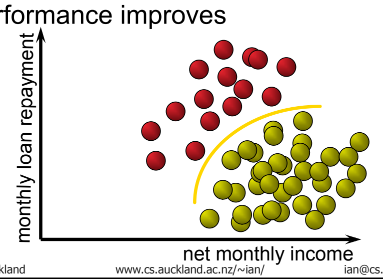
Nearest Neighbour

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

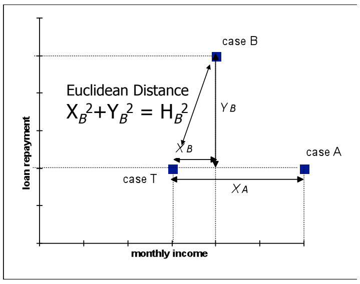


Nearest Neighbour

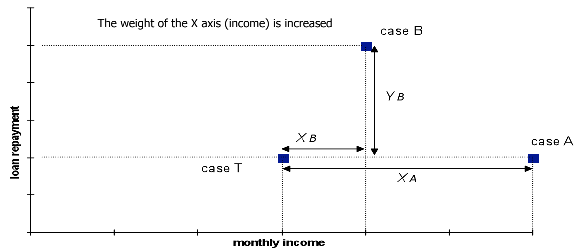
- as more cases are acquired its performance improves



Nearest Neighbour



Nearest Neighbour



28

Nearest Neighbour

- Requires a unique similarity function for each attribute or feature (not always a trivial problem) – *local similarity* $f(T_i,S_i)$
- Local similarities are combined to give a *global similarity* – 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 ☹

 www.cs.auckland.ac.nz/~ian/ ian@cs.auckland.ac.nz

29

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

 www.cs.auckland.ac.nz/~ian/ ian@cs.auckland.ac.nz

30

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

 www.cs.auckland.ac.nz/~ian/ ian@cs.auckland.ac.nz

31

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

32

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

33

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 001

Manufacturer: Mazda
Model: SP3
Engine Size: 2500
Body: 5 Door Hatch
Age: 2005
Colour: Silver

Price: $25000

4. Acquire a case-base

Case ID 001

Case ID 002

Case ID 003

Case ID 004

Case ID 005

Case ID 006

Manufacturer: Alpha Romeo
Model: Spider
Engine Size: 2500
Body: 2 Door Sports
Age: 2000
Colour: Red

Price: $19950

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!!!

Case ID 001

Manufacturer: Mazda
Model: SP3
Engine Size: 2500
Body: 5 Door Hatch
Age: 2005
Colour: Silver

Price: $25000

5. Local Similarity metrics

- Manufacturer
  - A symbolic feature
  - Mazda, Ford, Mercedes, BMW, Alpha Romeo, …
  - Is there a way of organising these to reflect their similarity wrteo ????
  - An ordered list (symbol set), a hierarchy,…
  - Note this may be very hard
  - An possible ordered list:
    - Mercedes – BMW – Alpha Romeo - Mazda - Ford

0 0.25 0.5 0.75 1.0

5. Local Similarity metrics

- Model
  - A symbolic feature
  - SP3, S Class, Falcon XR6, 330
  - Is there a way of organising these to reflect their similarity wrteo ????
  - An ordered list (symbol set), a hierarchy,…..
- **STOP !!!**
  - Actually model is a useful descriptor but is not very predictive of price after all
  - Model is superseded by Make, Engine Size and Body
  - We will not use Model as a feature for retrieval

The University of Auckland

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
  - *sim(f)* = (Range – Diff)/Range

(This normalises the result between 0 & 1)

The University of Auckland

5. Local Similarity metrics

- Engine Size

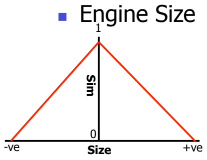
Source Case	Target Case	Diff	Range	Sim	Sim Normalised
1600	2500	900	6500	5600	0.86
2500	2500	0	6500	6500	1.00
3000	2500	500	6500	6000	0.92
3500	2500	1000	6500	5500	0.85

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

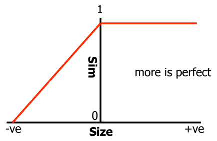
The University of Auckland

5. Local Similarity metrics

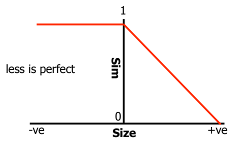
- Engine Size



Symmetric Similarity



Symmetric Similarity

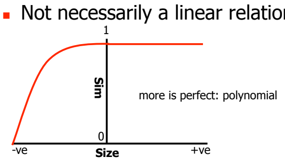


Symmetric Similarity

The University of Auckland

5. Local Similarity metrics

- Engine Size
  - Not necessarily a linear relationship

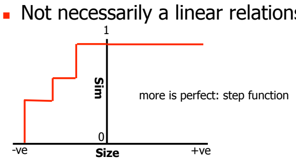


Symmetric Similarity

The University of Auckland

5. Local Similarity metrics

- Engine Size
  - Not necessarily a linear relationship



Symmetric Similarity

The University of Auckland

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

	4 door sedan	5 door hatch	3 door coupe	2 door sports
4 door sedan	1.00	0.80	0.50	0.20
5 door hatch	0.75	1.00	0.75	0.30
3 door coupe	0.75	0.75	1.00	0.40
2 door sports	0.00	0.00	0.00	1.00
target cases				

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

5. Local Similarity metrics

- 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)* = (Range – diff)/Range

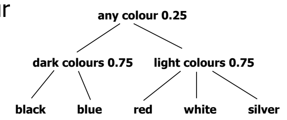
5. Local Similarity metrics

- 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



  - 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)

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

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

52

8. We're Almost Done ☺

- Let's go

Case ID 00?

Manufacturer: BMW
Model: 320
Engine Size: 2000
Body: 3 Door Coupe
Age: 2004
Colour: Blue

Price: $???

→ **$?**

53

8. We're Almost Done ☺

- Compare query against each case

Case ID 00?		Case ID 001	
Manufacturer: BMW	↔	Manufacturer: Mazda	= 0.5 x 10 = 5.0
Model: 320		Model: SP3	
Engine Size: 2000	↔	Engine Size: 2500	= 1.0 x 5 = 5.0
Body: 3 Door coupe	↔	Body: 5 Door Hatch	= 0.75 x 5 = 3.75
Age: 2004	↔	Age: 2005	= 0.95 x 5 = 4.75
Colour: Blue	↔	Colour: Silver	= 0.25 x 1 = 0.25
Price: $???		**Price: $25000**	

$\Sigma = 18.75/26 = 0.72$

sum of the feature weights

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

54

9. Result !!!

Case ID 002 Sim = 0.54

Case ID 004 Sim = 0.62

Case ID 001 Sim = 0.72

Case ID 006 Sim = 0.72

Case ID 003 Sim = 0.80

Case ID 005 Sim = 0.98

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 similiarity 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)

 www.cs.auckland.ac.nz/~ian/ ian@cs.auckland.ac.nz

Result!!!

- Use the price from the best matching case

Case ID 00?

Manufacturer: BMW
Model: 320
Engine Size: 2000
Body: 3 Door Coupe
Age: 2004
Colour: Blue

Price:

Case ID 005 **Sim = 0.98**

Manufacturer: BMW
Model: 330
Engine Size: 3000
Body: 3 Door Coupe
Age: 2005
Colour: Black

Price: $29500

 www.cs.auckland.ac.nz/~ian/ ian@cs.auckland.ac.nz

Adapt the Result!!!

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

Case ID 00?

Manufacturer: BMW
Model: 320
Engine Size: 2000
Body: 3 Door Coupe
Age: 2004
Colour: Blue

Price:

Case ID 005 **Sim = 0.98**

Manufacturer: BMW
Model: 330
Engine Size: 3000
Body: 3 Door Coupe
Age: 2005
Colour: Black

Price: $29500

 www.cs.auckland.ac.nz/~ian/ ian@cs.auckland.ac.nz