Australian National University

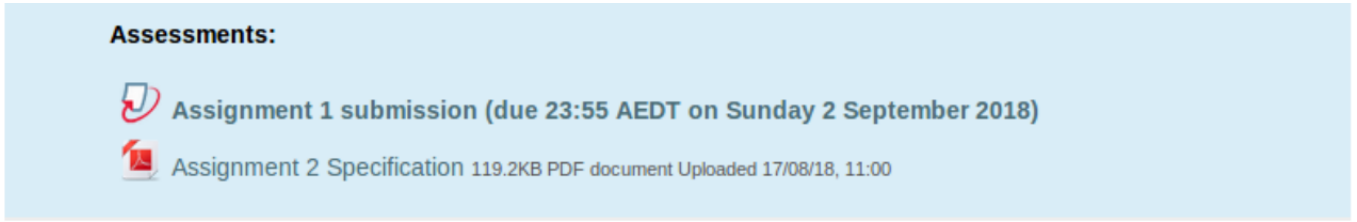
COMP3430 / COMP8430
Data wrangling

In person lecture week 6

(Lecturer: Thilina Ranbaduge)

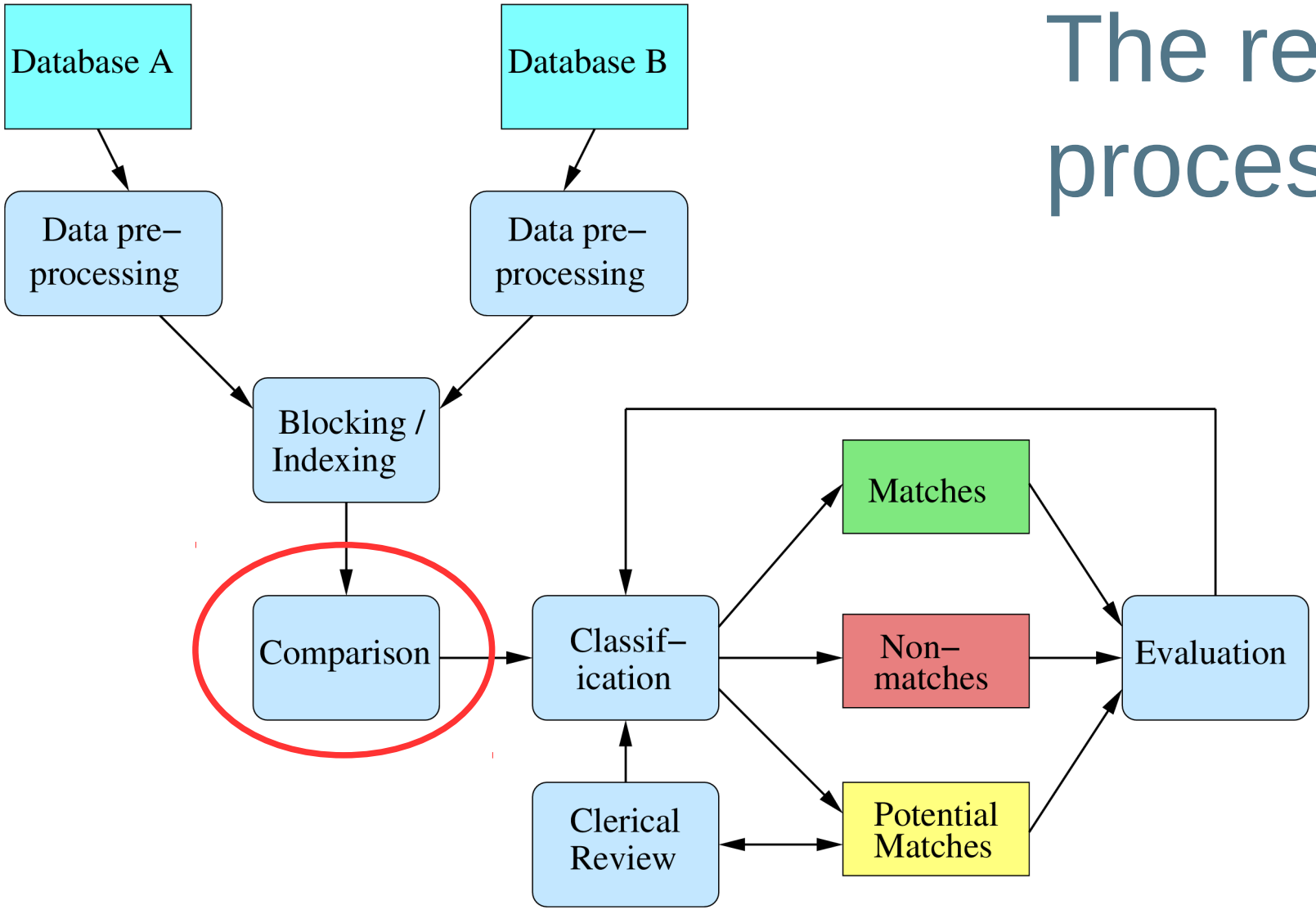
Some administrative things

- **Assignment 1 is due Sunday 2 Sept at 23:55! Assignment submission link is now available. See week 6.**



- **Assignment 2 specification is available in Wattle now.** Relevant data sets will be available by end of this week or the first week of September**.**

The record linkage process



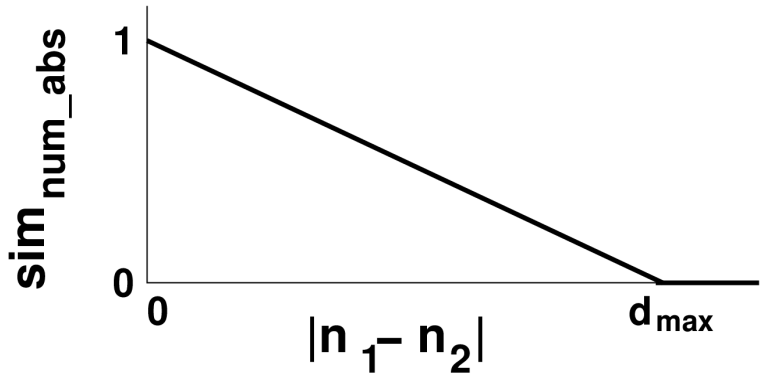
Comparing record pairs (2)

- Exact comparison of attribute values will not provide good linkage results
  - Even true matching record pairs often contain different attribute values
  - For example:
    ['peter', 'paul', 'meier', '2/21 main st', 'acton', 'act', ' 2601']
    ['peter', 'p', 'meyer', '21 main street', 'acton', 'act', ' 2602']

- Approximate comparison functions are required
  - To calculate similarities between attribute values, not only 'is the same or is different'
  - They need to be appropriate for the content of a certain attribute (text: names, addresses, dates, phone numbers; numerical: ages, salaries)

Numerical comparison functions (1)

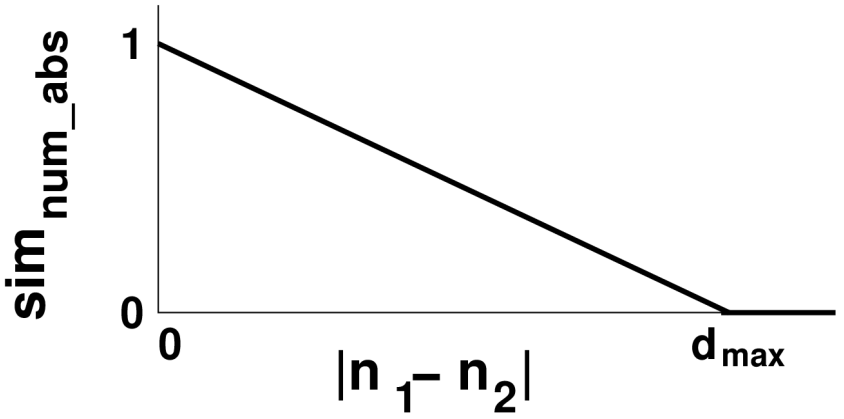
- For numerical values, we also want to have a comparison that calculates a similarity between 0 and 1
- We set a *maximum absolute difference* allowed, or a *maximum percentage difference* allowed
  - If two values differ more their similarity will be 0
- For absolute maximum difference of d_{max} and two values n_1 and n_2:
  - If $abs(n_1 - n_2) \geq d_{max}$: $sim_{num\_abs} = 0$
  - If $abs(n_1 - n_2) < d_{max}$: $sim_{num\_abs} = 1 - (abs(n_1 - n_2) / d_{max})$



Numerical comparison functions (2)

- Similar for maximum percentage difference
  - Similarity for income (salary) differences of maximum 5% is more suitable compared to a maximum difference of $10,000
  - Similarity of age difference by 10% is better than maximum age difference of 5 years (young compared to old people)

- **Question**: *Calculate similarities for absolute maximum difference of* $d_{max} = 5$, $n_1 = 42$ *and* $n_2 = \{37, 38, 40, 41, 49\}$



Then calculate percentage differences assuming these are ages

Q-gram based string comparison (1)

- Convert a string into its set of q-grams
  - Often with $q = 2$ (bigrams) or $q = 3$ (trigrams)
  - For example, with bigrams: “peter” → [‘pe’, ‘et’, ‘te’, ‘er’]
- Calculate the similarity between two strings based on counting the number of q-grams that occur in both strings
  - Jaccard similarity: $sim_{Jacc}(s_1,s_2) = |intersection(Q_1,Q_2)| / |union(Q_1,Q_2)|$
  - Dice coefficient: $sim_{Dice}(s_1,s_2) = 2* |intersection(Q_1,Q_2)| / (|Q_1|+|Q_2|)$

  where:
  - Q_x is the set of q-grams extracted from string s_x
  - $intersection(Q_1,Q_2)$ is the set of q-grams that occur in both strings
  - |..| denotes the number of elements in a set

Q-gram based string comparison (2)

- For example, with s_1 = “peter” and s_2 = “pete” and $q = 2$:
  - Q_1 = [‘pe’, ‘et’, ‘te’, ‘er’], Q_2 = [‘pe’, ‘et’, ‘te’], $|Q_1| = 4$, $|Q_2| = 3$
  - intersection(Q_1,Q_2) = [‘pe’, ‘et’, ‘te’] and *union*(Q_1,Q_2) = [‘pe’, ‘et’, ‘te’, ‘er’]
  - $sim_{Jacc}(s_1, s_2)$ = | [‘pe’, ‘et’, ‘te’] | / | [‘pe’, ‘et’, ‘te’, ‘er’] | = 3 / 4 = 0.75
  - $sim_{Dice}(s_1, s_2) = 2*3 / (3+4) = 6 / 7 = 0.857$

- **Questions**: *Which one is correct? Which one is better?*
  What are the Jaccard and Dice similarities between
  s_1 = 'peter' and s_2 = 'pedro' for q = *1, 2, and 3 ?*

Edit distance (1)

- Idea: Count how many basic *edit operations* are needed to convert one string into another (known as *Levenshtein* edit distance)
  - Insertion of a character: “pete” → “peter”
  - Deletion of a character: “miller” → “miler”
  - Substitution of a character: “smith” → “smyth”
  - Transpositions of two adjacent characters: “sydney” → “sydeny” (known as *Damerau-Levenshtein* edit distance)

- **Questions**: *What is the Levenshtein edit distance between “peter” and “petra”, and between “gayle” and “gail”?*

Edit distance (2)

- Convert an edit distance into a similarity $0 \leq sim_{edit\_dist} \leq 1$ by calculating $sim_{edit\_dist}(s_1, s_2) = 1 - edit\_dist(s_1, s_2) / max(len(s_1), len(s_2))$
- For example, with s_1 = “peter” and s_2 = “petra”:

  $sim_{edit\_dist}(s_1, s_2) = 1 - 2 / max(5, 5) = 1 - 2 / 5 = 3 / 5 = 0.6$

- Edit distance can be calculated using a dynamic programming algorithm based on the *edit matrix*
  - Which has a quadratic complexity in the lengths of the two strings (i.e. requires $len(s_1) * len(s_2)$ *computational steps*)

Edit distance (3)

- Matrix shows the number of edits between sub-strings
  (for example, between ‘ga’ and 'gayle’ → 3 inserts)

  “gail” → substitute ‘i’ with ‘y’,
  then insert ‘e’ → “gayle”
  (final edit distance is 2)

- **Question**: *Calculate edit distance between* s_1 *= “peter” and* s_2 *= “petra”*

		g	a	y	l	e
	0	1	2	3	4	5
g	1	**0**	1	2	3	4
a	2	1	**0**	1	2	3
i	3	2	1	**1**	2	2
l	4	3	2	2	**1**	**2**