Comparing initialisation processes for the k-modes algorithm, and an alternative process utilising the hospital-resident assignment problem

Henry Wilde

December 15, 2017

1 The k-modes algorithm

The k-modes algorithm is a part of the family of clustering algorithms known as 'prototype-based clustering', and is an extension of the k-means algorithm for categorical data as set out in [3]. This work will outline the key differences between the two algorithms, and then aim to examine how the initial cluster selection process has an impact on the efficiency and quality of the k-modes algorithm.

1.1 Notation

We will use the following notation throughout this work to describe our data set, points, clusters and representative points:

- Our dataset has N elements and is denoted by $\mathbf{X}$.
- $\mathbf{X}$ is described by a set of $m \in \mathbb{Z}_+$ attributes $\mathbf{A} = \{A_1, \ldots, A_m\}$.
- Each attribute A_j draws its values from a set $dom(A_j) = \{a_1^{(j)}, \ldots, a_d^{(j)}\}$ where $d = |dom(A_j)| \in \mathbb{Z}_+$ is sometimes used as shorthand and is not necessarily consistent with the d associated with any other attributes.
- We write each data point $X^{(i)}$ as an m-dimensional vector:

$$X^{(i)} = [A_1 = x_1^{(i)}, A_2 = x_2^{(i)}, \ldots, A_m = x_m^{(i)}], \quad i = 1, \ldots, N$$

  where $x_j^{(i)}$ is the value of the j^{th} attribute of the i^{th} data point, $X^{(i)}$.
- Prototype-based clustering algorithms partition the elements of $\mathbf{X}$ into k distinct sets (clusters) denoted by $C_1, \ldots, C_k$, where $k \in \mathbb{Z}_+$ is a pre-determined, fixed integer such that $k \leq N$. That is:

$$C_1, \ldots, C_k \text{ are such that } \bigcup_{l=1}^{k} C_l = \mathbf{X} \text{ and } C_l \cap C_t = \emptyset \text{ for all } l \neq t$$

- Each cluster C_l has associated with it a representative point (defined in Section 1.3) which we denote by $\mu^{(l)} = [\mu_1^{(l)}, \ldots, \mu_m^{(l)}]$.

1.2 Dissimilarity measure

An immediate difference between the k-means and k-modes algorithms is that they deal with different types of data, and so the metric used to define the distance between two points in our space must be different. With k-means, where the data has all-numeric attributes, Euclidean distance is often used. However, we do not have this sense of distance with categorical data. Instead, we utilise a dissimilarity measure - defined below - as our metric. It can be easily checked that this is indeed a distance measure.

Definition 1.1. Let $\mathbf{X}$ be a data set and consider $X^{(a)}, X^{(b)} \in \mathbf{X}$. We define the *dissimilarity* between $X^{(a)}$ and $X^{(b)}$ to be:

$$d(X^{(a)}, X^{(b)}) = \sum_{j=1}^{m} \delta(x_j^{(a)}, x_j^{(b)}) \quad \text{where} \quad \delta(x, y) = \begin{cases} 0, & x = y \\ 1, & \text{otherwise} \end{cases}$$

1.3 Representative points

Now that we have defined a metric on our space, we can turn our attention to what we mean by the representative point $\mu^{(l)}$ of a cluster C_l. In k-means, we call $\mu^{(l)}$ a 'centroid' and define it to be the average of all points $X^{(i)} \in C_l$ by Euclidean distance. With categorical data, we use our revised distance measure defined in Definition 1.1 to specify a representative point. We call such a point a mode of $\mathbf{X}$.

Definition 1.2. We define a *mode* of our set $\mathbf{X}$ to be any vector $\mu = [\mu_1, \ldots, \mu_m]$ that minimises:

$$D(\mathbf{X}, \mu) = \sum_{i=1}^{n} d(X_i, \mu) \tag{1}$$

Note that μ is not necessarily in $\mathbf{X}$. We call such a mode a *virtual mode*.

Definition 1.3. Let $\mathbf{X}$ be a dataset with attributes $A_1, \ldots, A_m$. Then we denote by $n(a_s^{(j)})$ the *frequency* of the s^{th} category $a_s^{(j)}$ of A_j in $\mathbf{X}$. That is,

$$n(a_s^{(j)}) := |\{X^{(i)} \in \mathbf{X} : x_j^{(i)} = a_s^{(j)}\}|$$

We call $\frac{n(a_s^{(j)})}{N}$ the *relative frequency* of category $a_s^{(j)}$ in $\mathbf{X}$.

Remark. Note that we have $1 \leq n(a_s^{(j)}) \leq N$ for all s and $j = 1, \ldots, m$.

Theorem 1. Consider a dataset $\mathbf{X}$ and some $X^{(i)} \in \mathbf{X}$. Then:

$$D(\mathbf{X}, X^{(i)}) \text{ is minimised} \iff n(x_j^{(i)}) \geq n(a_s^{(j)}) \text{ for all } s \text{ and } j = 1, \ldots, m$$

A proof of this theorem can be found in the Appendix of [3].

1.4 The cost function

We can use Definitions 1.1 & 1.2 to determine a cost function for our algorithm. Let $\bar{\mu} = \{\mu^{(1)}, \ldots, \mu^{(k)}\}$ be a set of k modes of $\mathbf{X}$, and let $W = (w_{i,l})$ be an $n \times k$ matrix such that:

$$w_{i,l} = \begin{cases} 1, & X^{(i)} \in C_l \\ 0, & \text{otherwise} \end{cases}$$

Then we define our *cost function* to be the summed within-cluster dissimilarity:

$$\text{Cost}(W, \bar{\mu}) = \sum_{l=1}^{l=k} \sum_{i=1}^{i=n} \sum_{j=1}^{j=m} w_{i,l} \delta(x_{i,j}, \mu_{l,j}) \tag{2}$$

1.5 The k-modes algorithm

Below is a practical implementation of the k-modes algorithm [3]:

Algorithm 1 k-modes

$\bar{\mu} \leftarrow \emptyset$
for $l \in \{1, \ldots, k\}$ **do**
  $C_l \leftarrow \emptyset$
end for
Select k initial modes, $\mu^{(1)}, \ldots, \mu^{(k)}$.
$\bar{\mu} \leftarrow \{\mu^{(1)}, \ldots, \mu^{(k)}\}$
for $X_i \in \mathbf{X}$ **do**
  Select l^* that satisfies $d(X^{(i)}, \mu^{(l^*)}) = \min_{l \in \{1,\ldots,m\}} \{d(X^{(i)}, \mu^{(l)})\}$
  $C_{j^*} \leftarrow C_{j^*} \cup \{X^{(i)}\}$
  Update $\mu^{(l^*)}$
end for
repeat
  for $X^{(i)} \in \mathbf{X}$ **do**
    for $\mu^{(l)} \in \bar{\mu}$ **do**
      Calculate $d(X^{(i)}, \mu^{(l)})$
    end for
    if $d(X^{(i)}, \mu^{(l^*)}) > d(X^{(i)}, \mu^{(l')})$ for some $l' \neq l^*$ **then**
      $C_{l^*} \leftarrow C_{l^*} \setminus \{X^{(i)}\}$
      $C_{l'} \leftarrow C_{l'} \cup \{X^{(i)}\}$
      Update both $\mu^{(l^*)}$ and $\mu^{(j')}$
    end if
  end for
until No point changes cluster after a full cycle through $\mathbf{X}$

Remark. The processes by which the k initial modes are selected are detailed in Sections 2 & 4.

2 Initialisation processes

From the literature surrounding this topic, it has been established that the initial choice of clusters impacts the final solution of the k-modes algorithm [3][1]. While some works attempt to improve the quality of k-modes and similar algorithms by considering an alternative dissimilarity measure [2], this work will examine the way in which these k initial representative points are chosen. Two established methods of selecting these initial points are described in Sections 2.1 & 2.2.

2.1 Huang's method

In the standard form of the k-modes algorithm, the k initial modes are chosen at random from $\mathbf{X}$. Below is an alternative method of selecting these modes that forces some diversity between them, as described in [3]:

Algorithm 2 Huang's method

$\bar{\mu} \leftarrow \emptyset$
Let $P = (p_{s,j})$ be an empty $N \times m$ matrix.
for $j = 1, \ldots, m$ **do**
  $d \leftarrow |dom(A_j)|$
  for $s = 1, \ldots, d$ **do**
    Calculate $n(a_s^{(j)})$
  end for
  Sort $dom(A_j) = \{a_1^{(j)}, \ldots, a_d^{(j)}\}$ into descending order by frequency, breaking ties arbitrarily.
  Call this arrangement $dom^*(A_j)$.
  for $a_s^{(j)} \in dom^* A_j$ **do**
    $p_{s,j} \leftarrow \frac{n(a_s^{(j)})}{N}$
  end for
  $p_{s,j}$ is left empty for all $s > d$
end for
for $l = 1, \ldots, k$ **do**
  for $j = 1, \ldots, m$ **do**
    Take the nonempty entries of $p_{*,j}$ as a vector and consider it as a probability distribution.
    Sample $a_{s^*}^{(j)}$ from $dom^*(A_j)$ according to this probability distribution.
    $\mu_j^{(l)} \leftarrow a_{s^*}^{(j)}$
  end for
  $\bar{\mu} \leftarrow \bar{\mu} \cup \mu^{(l)}$
end for
for $l = 1, \ldots, k$ **do**
  Select $X^{(i)} \in \mathbf{X}$ such that $d(X^{(i)}, \mu^{(l)}) = \min_{t \in \{1,\ldots,m\}}\{d(X^{(t)}, \mu^{(l)})\}$ and $X^{(i)} \neq \mu^{(l')}$ for all $\mu^{(l')} \in \bar{\mu}$
  $\mu^{(l)} \leftarrow X^{(i)}$
  $\bar{\mu} \leftarrow \mu^{(l)}$
end for

In the original statement of Huang's method [3], the algorithm states that the most frequent categories should be assigned 'equally' to the k initial modes. How the categories should be distributed 'equally' is not well-defined or easily seen from the example given. In Section 5, an implementation of the k-modes algorithm (written in Python) is used to compare the quality of the initialisation processes discussed throughout this piece of work when applied to a collection of datasets. That implementation distributes the attribute values in a random way (with replacement) according to the probability distribution formed by the relative frequencies of each category for each attribute.

In our examples, we will assign the categories to our initial modes in the same way, as it is described in Algorithm 2. This ambiguity in the definition of Huang's method means that a probabilistic element must be introduced, and unless seeded pseudo-random numbers are used, results are not necessarily reproducible.

A small example of this method is given below.

Skip this example and replace with the toy example

Example 1. Below are the first five rows of a random sample of 250 records from a data set used to determine the acceptability of a car. This dataset was chosen primarily for its number of attributes. However, it should be noted that one downfall of this particular data set is that some of the attributes could be considered as ordinal rather than purely categorical since there are clearly established and easily understandable differences between "high" and "low" prices, for instance.

Price	Maintenance	Doors	Passengers	Luggage	Safety
low	vhigh	2	5+	med	med
vhigh	high	2	4	big	med
high	med	2	2	small	low
vhigh	med	3	2	big	low
low	med	5+	2	big	low

The frequencies of our attributes' categories are given below:

Price	Maintenance	Doors	Passenger	Luggage	Safety
$f(c_{\text{low}}) = 61$	$f(c_{\text{low}}) = 53$	$f(c_2) = 71$	$f(c_2) = 81$	$f(c_{\text{small}}) = 88$	$f(c_{\text{low}}) = 76$
$f(c_{\text{med}}) = 63$	$f(c_{\text{med}}) = 66$	$f(c_3) = 71$	$f(c_4) = 85$	$f(c_{\text{med}}) = 78$	$f(c_{\text{med}}) = 91$
$f(c_{\text{high}}) = 63$	$f(c_{\text{high}}) = 50$	$f(c_4) = 53$	$f(c_{5+}) = 84$	$f(c_{\text{big}}) = 84$	$f(c_{\text{high}}) = 83$
$f(c_{\text{vhigh}}) = 63$	$f(c_{\text{vhigh}}) = 81$	$f(c_{5+}) = 55$			

Table 1: Frequencies of all attribute categories, $f(c_{s,j})$

Thus, from Table 1 we see that our category matrix is:

$$\begin{pmatrix} \text{vhigh} & \text{vhigh} & 3 & 4 & \text{small} & \text{med} \\ \text{high} & \text{med} & 2 & 5+ & \text{big} & \text{high} \\ \text{med} & \text{low} & 5+ & 2 & \text{med} & \text{low} \\ \text{low} & \text{high} & 4 & & & \end{pmatrix}$$

Acceptability is an attribute of this data which has been removed but indicates whether a car is one of 'very good', 'good', 'acceptable' or 'unacceptable'. From this we can suppose that we are looking for $k = 4$ clusters, and so, by distributing the most frequent categories 'equally' our initial set of modes is:

$$\begin{aligned} \bar{\mu} = \{&\mu^{(1)} = [\text{vhigh}, \text{med}, 5+, 4, \text{big}, \text{low}], \quad \mu^{(2)} = [\text{high}, \text{low}, 4, 5+, \text{med}, \text{med}], \\ &\mu^{(3)} = [\text{med}, \text{high}, 3, 2, \text{small}, \text{high}], \quad \mu^{(4)} = [\text{low}, \text{vhigh}, 2, 4, \text{big}, \text{med}]\} \end{aligned} \tag{3}$$

Now we would select the least dissimilar point in our data set to replace each $\mu^{(l)} \in \bar{\mu}$ in numerical order according to Definition 1.1 and continue with the rest of the algorithm.

2.2 Cao's method

Cao's method selects representative points by the average density of a point in the dataset. As will be seen in the following definition, this average density is in fact the average relative frequency of all the attribute values of that point. This method is considered deterministic as there is no probabilistic element - unlike the standard or Huang's method - and so results are completely reproducible.

Definition 2.1. Consider a data set $\mathbf{X}$ with attribute set $\mathbf{A} = \{A_1, \ldots, A_m\}$. Then the *average density* of any point $X_i \in \mathbf{X}$ with respect to $\mathbf{A}$ is defined [1] as:

$$\text{Dens}(X^{(i)}) = \frac{\sum_{j=1}^{m} \text{Dens}_{A_j}(X^{(i)})}{m}, \quad \text{where} \quad \text{Dens}_{A_j}(X^{(i)}) = \frac{|\{X^{(t)} \in \mathbf{X} : x_j^{(i)} = x_j^{(t)}\}|}{N} = \frac{n(x_j^{(i)})}{N}$$

Observe that:

$$|\{X^{(t)} \in \mathbf{X} : x_j^{(i)} = x_j^{(t)}\}| = n(x_j^{(i)}) = \sum_{t=1}^{N} (1 - \delta(x_j^{(i)}, x_j^{(t)}))$$

and so, we can find an alternative definition for $\text{Dens}(X^{(i)})$:

$$\begin{aligned}\text{Dens}(X^{(i)}) &= \frac{1}{mN}\sum_{j=1}^{m}\sum_{t=1}^{N}(1-\delta(x_j^{(i)},x_j^{(t)})) \\ &= \frac{1}{mN}\sum_{j=1}^{m}\sum_{t=1}^{N}1 - \frac{1}{mN}\sum_{j=1}^{m}\sum_{t=1}^{N}\delta(x_j^{(i)},x_j^{(t)}) \\ &= \frac{mN}{mN} - \frac{1}{mN}\sum_{t=1}^{N}d(X^{(i)},X^{(t)}) \\ &= 1 - \frac{1}{mN}D(\mathbf{X},X^{(i)})\end{aligned} \tag{4}$$

Remark. It is worth noting that we have $\frac{1}{N} \leq \text{Dens}(X^{(i)}) \leq 1$, since for any $A_j \in \mathbf{A}$:

- If $n(x_j^{(i)}) = 1$, then $\text{Dens}(X^{(i)}) = \frac{\sum_{j=1}^{m}\frac{1}{N}}{m} = \frac{m}{mN} = \frac{1}{N}$
- If $n(x_j^{(i)}) = N$, then $\text{Dens}(X^{(i)}) = \frac{\sum_{j=1}^{m}1}{m} = \frac{m}{m} = 1$

With this alternative definition, we see – since m and N are fixed positive integers – that $\text{Dens}(X^{(i)})$ is maximised when $D(\mathbf{X}, X^{(i)})$ is minimised. Then by Theorem 1 we have that such an $X^{(i)}$ with maximal average density in $\mathbf{X}$ with respect to $\mathbf{A}$ is, in fact, a mode of $\mathbf{X}$. This notion helps justify the method proposed by Cao et al. as discussed below.

Cao's selection process is as follows:

Algorithm 3 Cao's method

$\bar{\mu} \leftarrow \emptyset$
for $X^{(i)} \in \mathbf{X}$ **do**
  Calculate $\text{Dens}(X^{(i)})$
end for
Select $X^{(i_1)} \in \mathbf{X}$ which satisfies $\text{Dens}(X^{(i_1)}) = \max_{X^{(i)}}\{\text{Dens}(X^{(i)})\}$
$\bar{\mu} \leftarrow \bar{\mu} \cup X^{(i_1)}$
Select $X^{(i_2)} \in \mathbf{X}$ such that:

$$\text{Dens}(X^{(i_2)}) \times d(X^{(i_1)},X^{(i_2)}) = \max_{X^{(i)}\in\mathbf{X}}\{d(X^{(i)},X^{(i_1)})\}$$

$\bar{\mu} \leftarrow \bar{\mu} \cup X^{(i_2)}$
while $|\bar{\mu}| < k$ **do**
  Select $X^{(i_t)} \in \mathbf{X}$ such that for all $\mu^{(l)} \in \bar{\mu}$:

$$d(X^{(i_t)},\mu^{(l)}) \times \text{Dens}(X_{i_t}) = \max_{X^{(i)}\in\mathbf{X}}\{\min_{\mu^{(l)}\in\bar{\mu}}\{d(X^i,\mu^{(l)}) \times \text{Dens}(X^{(i)})\}\}$$

  $\bar{\mu} \leftarrow \bar{\mu} \cup X^{(i_t)}$
end while

3 Matching games

The primary motivation for this work is to move away from the greedy approaches defined above by incorporating some techniques from game theory, namely: matching games. The purpose of solving a matching

game is to link the elements of two sets in a 'fair' way. By considering the initial modes found by Huang's method with some suitable subset of our dataset as a matching game, we can hope to find a closer-to-optimal set of initial modes for the k-modes algorithm.

Definition 3.1. Consider two sets S, R each of size N, and let us call them 'suitors' and 'reviewers'. Each element of S and R has associated with it a preference list of the other set's elements. These preference lists are ranked in descending order. We consider the preference lists as functions, f and g, respectively:

$$f : S \to R^n, g : R \to S^n$$

This construction of sets and preference lists is called a *matching game* of size N and we denote the game itself by (S, R).

A matching, M, is defined to be any bijection between S and R. If $s \in S$ and $r \in R$ are matched by M, then we write $M(s) = r$. A matching M is considered to be either stable or unstable based on the preference lists of the suitors and reviewers, and the presence of blocking pairs.

Definition 3.2. Let (S, R) be a matching game of size N with some matching M. A pair $(s, r) \in S \times R$ is said to *block* M if $M(s) \neq r$ but s prefers r to $M(s) = r'$ and r prefers s to $M^{-1}(r) = s'$. That is, r appears before r' in $f(s)$ and s appears before s' in $g(r)$.

Definition 3.3. Let (S, R) be a matching game of size N with some matching M. Then we say M is a *stable matching* if there are no blocking pairs, and unstable otherwise.

Example 2.

3.1 The Gale-Shapley algorithm

The Gale-Shapley algorithm is known to find a unique stable matching for any matching game of size N. This matching is also considered to be suitor-optimal. That is, each suitors is matched with the best possible reviewer that ensures a stable matching, but is in fact the worst possible matching for the reviewers [!!! cite or maybe have these theorems stated/proven !!!].

As was discussed at the start of Section 3, the hope for this proposed method is to consider our virtual modes with some subset of the data as a matching game and then solve it. It should be noted, however, that in this method we may not have equally sized sets of suitors and reviewers. As a result of this, the Gale-Shapley algorithm becomes inapplicable as the matching produced M would not be a bijection of our suitors and reviewers.

3.2 The capacitated Gale-Shapley algorithm for the hospital-resident problem

The situation where a large set of suitors are to be matched with a number reviewers is not unheard of. A practical example of this problem is how to best assign a cohort of medical students to a group of hospitals. Here, we have all of the requisite components of a matching game:

- A set of reviewers (the hospitals) and a set of suitors (the potential residents)
- A ranking of the students/residents by the hospitals, and vice versa

The only issue stopping us from using the Gale-Shapley algorithm is the disparity in the sizes of our sets. In reality, hospitals need not always take at most one resident on from a cohort of medical students. So each hospital has a capacity associated with it and we can consider our matching game to 'capacitated'. By this we mean that each reviewer may be matched with any number of suitors up to the capacity associated with it, making our matching $M : S \to R$ surjective.

Research surrounding the hospital-resident assignment problem is well-documented [cite] and an extension of the Gale-Shapley algorithm was developed to solve it, awarding the authors the 2012 Nobel

Prize in Economic Sciences. This algorithm is currently used by the National Resident Matching Program (`http://www.nrmp.org`).

As before, we consider a set of suitors and reviewers denoted by S and R. These sets are no longer (necessarily) the same size. We also have our preference lists f, g, and a set $C = \{c_1, \ldots, c_{|R|}\}$ of reviewer capacities. Finally, let $S_u \subset S$ denote the set of suitors that are currently unmatched. The capacitated Gale-Shapley algorithm is given below.

Algorithm 4 Capacitated Gale-Shapley

for $s \in S$ **do**
  $M(s) \leftarrow \emptyset$
end for
for $r \in R$ **do**
  $M^{-1}(r) \leftarrow \emptyset$
end for
$S_u \leftarrow S$
while $|S_u| > 0$ **do**
  Select any $s \in S_u$
  if $|f(s)| = 0$ **then**
    $S_u \leftarrow S_u \setminus \{s\}$
  else
    Select s's most preferred reviewer $r \in R$
    if $|M(r)| < c_r$ **then**
      $M(r) \leftarrow M(r) \cup s$
      $S_u \leftarrow S_u \setminus \{s\}$
    else
      for $s' \in M(r)$ **do**
        if $s \notin M(r)$ **then**
          if r prefers s to s' **then**
            $M(r) \leftarrow M(r) \cup s$
            $S_u \leftarrow S_u \cup \{s\}$
            $M(r) \leftarrow M(r) \setminus s'$
            $S_u \leftarrow S_u \cup \{s'\}$
          else
            $f(s) \leftarrow f(s) \setminus \{r\}$
          end if
        end if
      end for
    end if
  end if
end while

Remark. This implementation requires all residents to be ranked by all hospitals, and will produce a matching such that no hospital is left without at least one resident.

4 The proposed method

Now that we have defined what we mean by a matching game, with the algorithm described above, we can build an alternative initialisation process for the k-modes algorithm.

Let $\mathbf{X}$ be a dataset with attribute set $\mathbf{A}$, and let $\bar{\mu}$ be the set of virtual modes found by the Huang method up to Step 3. Then we construct the following capacitated matching game:

- The set of hospitals H is $\bar{\mu}$, and each hospital has capacity 1.
- The set of residents, R, is made up of the k least dissimilar points $X_{l,1}, \ldots, X_{l,k} \in \mathbf{X}$ to each $\mu^{(l)} \in \bar{\mu}$.
- Each hospital's preference list is simply their addition to the set of residents in descending order of similarity.
- The preference lists of the residents is more complicated. In this initial implementation, we take their preference list to be the set of hospitals in ascending order with respect to dissimilarity. Though, as will be seen in Section 5, other ways of generating these lists (such as randomly) can provide different results.

Now, by applying the capacitated Gale-Shapley algorithm to this game, we find a resident-optimal matching M. Let our set of modes $\bar{\mu} := M^{-1}(H)$. That is, the l^{th} mode is the resident matched with $\mu^{(l)}$ when the algorithm concludes.

5 Experimental results

To give comparative results on the quality of the initialisation processes defined in Sections 2 & 4, four well-known, categorical, labelled datasets - soybean, mushroom, breast cancer, and zoo - will be clustered with the k-modes algorithm. Then the typical performance measures of accuracy, precision, and recall will be calculated and summarised below. As a general rule, each algorithm will be trained on approximately two thirds of the respective dataset and tested against the final third.

Definition 5.1. Let a dataset $\mathbf{X}$ have k classes $C_1, \ldots, C_k$, let the number of objects correctly assigned to C_i be denoted tp_i, let fp_i denote the number of objects incorrectly assigned to C_i, and let fn_i denote the number of objects incorrectly not assigned to C_i. Then our performance measures are defined as follows:

$$\textit{Accuracy}: \quad \frac{\sum_{i=1}^{k} tp_i}{|\mathbf{X}|}, \quad \textit{Precision}: \quad \frac{\sum_{i=1}^{k} \frac{tp_i}{tp_i + fp_i}}{k}, \quad \textit{Recall}: \quad \frac{\sum_{i=1}^{k} \frac{tp_i}{tp_i + fn_i}}{k}$$

5.1 The datasets

A bit on the structure of each dataset and links to access them.

5.2 Results

Tables of results for each dataset and each initialisation process. Credit to `https://github.com/nicodv/kmodes` for the Python implementation of both the Huang and Cao processes, as well as the k-modes algorithm itself.

6 Resident preference lists

Some examples and hopefully some mathematical reasoning to justify that certain choices of preference lists reduce down to near equivalent results of the Huang method (or others). This then suggests the proposed method is in fact a generalisation of the other method(s).

References

[1] Bai L Cao F Liang J. "A new initialization method for categorical data clustering". In: *Expert Systems with Applications* 36 (2009), pp. 10223–10228. URL: https://pdfs.semanticscholar.org/1955/c6801bca5e95a44e70ce14180f00fd3e55b8.pdf.

[2] Huang Z He Z Ng MK Li MJ. "On the impact of dissimilarity measure in k-modes clustering algorithm". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29.3 (Mar. 2007).

[3] Huang Z. "Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Values". In: *Data Mining and Knowledge Discovery* 2.3 (Sept. 1998), pp. 283–304. DOI: 10.1023/A:1009769707641. URL: https://doi.org/10.1023/A:1009769707641.