Nearest Neighbours: nagdmc_knnp

Purpose

nagdmc_knnp computes k-nearest neighbour approximations given a binary tree computed by **nagdmc_kdtree** using training data.

Declaration

```
#include <nagdmc.h>
void nagdmc_knnp(long rec1, long nvar, long nrec, long dblk, double data[],
                 long iproot, int norm, long k, double res[], long nn[],
                 double dist[], int *info);
```

Parameters

1: **rec1** – `long` *Input*

On entry: the index in the data of the first data record used in the analysis.

Constraint: $\mathbf{rec1} \geq 0$.

2: **nvar** – `long` *Input*

On entry: the number of variables in the data.

Constraint: $\mathbf{nvar} > 1$.

3: **nrec** – `long` *Input*

On entry: the number of consecutive records, beginning at **rec1**, used in the analysis.

Constraint: $\mathbf{nrec} > 1$.

4: **dblk** – `long` *Input*

On entry: the total number of records in the data block.

Constraint: $\mathbf{dblk} \geq \mathbf{rec1} + \mathbf{nrec}$.

5: **data**[**dblk** ∗ **nvar**] – `double` *Input*

On entry: the data values for the jth variable (for $j = 0, 1, \ldots, \mathbf{nvar}-1$) are stored in $\mathbf{data}[i{*}\mathbf{nvar}+j]$, for $i = 0, 1, \ldots, \mathbf{dblk} - 1$.

6: **iproot** – `long` *Input*

On entry: the integer value of the root node of a binary tree as returned by **nagdmc_kdtree**.

7: **norm** – `int` *Input*

On entry: the norm used to compute distances. If $\mathbf{norm} = 1$, the ℓ_1-norm (or Manhattan distance) is used; otherwise $\mathbf{norm} = 2$ and the ℓ_2-norm (or Euclidean distance) is used.

Constraint: $\mathbf{norm} \in \{1, 2\}$.

8: **k** – `long` *Input*

On entry: the number of nearest neighbours used in the computation.

Constraint: $0 < \mathbf{k} < \mathbf{nrec}$.

9: **res**[**nrec**] – `double` *Output*

On exit: **res**[i] contains the k-nearest neighbour approximation for the ith data record, for $i = 0, 1, \ldots, \mathbf{nrec} - 1$.

10: **nn**[**nrec**∗**k**] – `long` *Output*

On exit: if **nn** is set to 0, it is not referenced; otherwise $\mathbf{nn}[i * \mathbf{k} + j]$ contains the index in the training data for the jth nearest neighbour to the ith data record, for $j = 0, 1, \ldots, \mathbf{k} - 1$; for $i = 0, 1, \ldots, \mathbf{nrec} - 1$.

11: **dist**[**nrec**∗**k**] – `double` *Output*

On exit: if **dist** is set to 0, it is not referenced; otherwise $\mathbf{dist}[i * \mathbf{k} + j]$ contains the distance from the ith data record to its jth nearest neighbour, for $j = 0, 1, \ldots, \mathbf{k} - 1$; for $i = 0, 1, \ldots, \mathbf{nrec} - 1$.

12: **info** – `int *` *Output*

On exit: **info** gives information on the success of the function call:

- 0: the function successfully completed its task.
- i; $i = i = 1, 2, 3, 4, 7, 8$: the specification of the ith formal parameter was incorrect.
- 57: information in the binary tree has been corrupted.
- 99: the function failed to allocate enough memory.
- 100: an internal error occurred during the execution of the function.
- > 100: an error occurred in a function specified by the user.

Notation

nrec the number of data records in the analysis, n.
data the data values, X.
k the number of nearest neighbours used in the calculations, k.
res the nearest neighbour approximations $\hat{y}_i$, for $i = 1, 2, \ldots, n$.

Description

Let X be a set of n data records x_i, for $i = 1, 2, \ldots, n$, on p independent variables and a continuous dependent variable y. The ith data record takes a value x_{ij} on the jth independent variable.

The k-nearest neighbour approach searches a set of training data records T (i.e., data records with known values for y) to find the k-nearest data records to x_i. Nearest neighbours are found by using a binary tree search, e.g., see Bentley (1975). The proximity of x_i to a member t of T is defined by a distance calculated over the independent variables and can defined by using one of:

(a) the ℓ_1-norm or Manhattan distance:

$$\sum_{j=1}^{p} |x_{ij} - t_j|,$$

where $|\cdot|$ denotes the modulus operator;

(b) the ℓ_2-norm or Euclidean distance:

$$\left[\sum_{j=1}^{p} (x_{ij} - t_j)^2\right]^{1/2}.$$

Let S_i be a set containing the k-nearest neighbours in T to x_i. Then the predicted value of y for x_i, $\hat{y}_i$, is given by the mean of its k-nearest neighbours:

$$\hat{y}_i = \frac{1}{k} \sum_{\forall t \in S_i} t_y.$$

References and Further Reading

Bentley J L (1975) Multi-dimensional binary search trees used for associative searching *Communications of the ACM* **18**(9) 509–517.

Duda R O and Hart P E (1972) *Pattern Classification and Scene Analysis* Wiley New York.

Storer J A and Cohn M (1993) Algorithms for fast vector quantization *Proc. Data Compression Conference* 381–390 IEEE Computer Society Press.

See Also

nagdmc_kdtree computes a binary tree for a nearest neighbour analysis.
nagdmc_free_kdtree frees the memory containing a binary tree.
nagdmc_load_kdtree loads a binary tree from a file into memory.
nagdmc_save_kdtree writes a binary tree to file.
knnp_ex.c the example calling program.