Nearest Neighbours: nagdmc_knnc

Purpose

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

Declaration

```
#include <nagdmc.h>
void nagdmc_knnc(long rec1, long nvar, long nrec, long dblk, double data[],
                 long iproot, double prior[], double rho, long uc, int norm,
                 long k, long 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: **rec1** ≥ 0.

2: **nvar** – `long` *Input*

On entry: the number of variables in the data.

Constraint: **nvar** > 1.

3: **nrec** – `long` *Input*

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

Constraint: **nrec** > 1.

4: **dblk** – `long` *Input*

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

Constraint: **dblk** $\geq$ **rec1** + **nrec**.

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

On entry: the data values for the jth variable (for $j = 0, 1, \ldots,$ **nvar**-1) are stored in **data**[$i*$**nvar**$+j$], for $i = 0, 1, \ldots,$ **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: **prior**[c] – `double` *Input*

On entry: if **prior** is set to 0, uniform priors are used; otherwise **prior**[i] gives the prior probability for the ith of c categories on the dependent variable in the analysis, for $i = 0, 1, \ldots, c - 1$.

Constraints: if **prior** is not 0, **prior**[i] ≥ 0, for $i = 0, 1, \ldots, c - 1$, and the elements in **prior** must sum equal to one.

8: **rho** – `double` *Input*

On entry: the value of maximum probability of group membership that must be exceeded for classification. Each data record with a maximum probability of group membership less than or equal to **rho** is classified as **uc**.

Constraint: $0 \leq$ **rho** < 1.

9: **uc** – `double` *Input*

On entry: the value that should be assigned to data records if the value of **rho** is not exceeded.

10: **norm** – `int` *Input*

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

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

11: **k** – `long` *Input*

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

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

12: **res[nrec]** – `long` *Output*

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

13: **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 $i = 0, 1, \ldots, \mathbf{nrec} - 1$.

14: **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 $i = 0, 1, \ldots, \mathbf{nrec} - 1$.

15: **info** – `int *` *Output*

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

- 0: the function successfully completed its task.
- i; $i = i = 1, 2, 3, 4, 7, 8, 10, 11$: 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.

Notation

nrec	the number of data records to classify, n.
data	the data values, X.
prior	the prior probabilities p_l, for $l = 1, 2, \ldots, c$.
rho	the threshold for accepting classifications, ρ.
uc	the dummy value representing unclassified data records, z.
k	the number of nearest neighbours used in the calculations, k.
res	the nearest neighbour classifications $\hat{y}_i$, 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 categorical dependent variable y. The jth value of the ith data record is denoted by x_{ij}. Each member of X is to be classified into one of c categories where the prior probability of the lth category is p_l, for $l = 1, 2, \ldots, c$.

The k-nearest neighbour approach searches a set of training data records T (i.e., data records with known categories 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, and h_{il} be the number of members of S_i belonging to the lth category. The posterior probability θ_{il} of x_i belonging to the lth category is given by,

$$\theta_{il} = \frac{p_l h_{il}}{\sum_{m=1}^{c} p_m h_{im}}.$$

Let q denote the index of the maximum value in θ_{il}, for $l = 1, 2, \ldots, c$. Given a user-supplied value for ρ, x_i is classified by setting the ith value of the dependent variable, $\hat{y}_i$, to category value q if $\theta_{iq} > \rho$; otherwise x_i is unclassified and $\hat{y}_i$ is assigned a dummy value, say z.

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.
knnc_ex.c	the example calling program.