

# Package ‘knn.covertree’

October 28, 2019

**Type** Package

**Title** An Accurate kNN Implementation with Multiple Distance Measures

**Version** 1.0

**Date** 2019-10-24

**Maintainer** Philipp Angerer <philipp.angerer@helmholtz-muenchen.de>

**Description** Similarly to the 'FNN' package, this package allows calculation of the k nearest neighbors (kNN) of a data matrix.  
The implementation is based on cover trees introduced by Alina Beygelzimer, Sham Kakade, and John Langford (2006) <doi:10.1145/1143844.1143857>.

**URL** <https://github.com/flying-sheep/knn.covertree>

**BugReports** <https://github.com/flying-sheep/knn.covertree/issues>

**License** AGPL-3

**Imports** Rcpp (>= 1.0.2), RcppEigen (>= 0.3.3.5.0), Matrix, methods

**Suggests** testthat, FNN

**LinkingTo** Rcpp, RcppEigen

**SystemRequirements** C++11

**NeedsCompilation** yes

**Encoding** UTF-8

**RoxygenNote** 6.1.1

**Author** Philipp Angerer [cre, aut] (<<https://orcid.org/0000-0002-0369-2888>>),  
David Crane [cph, aut]

**Repository** CRAN

**Date/Publication** 2019-10-28 16:00:02 UTC

## R topics documented:

find_knn . . . . .	2
knn.covertree . . . . .	3

<b>Index</b>	<b>4</b>
--------------	----------

---

find_knn	<i>kNN search</i>
----------	-------------------

---

### Description

k nearest neighbor search with custom distance function.

### Usage

```
find_knn(data, k, ..., query = NULL, distance = c("euclidean",
  "cosine", "rankcor"), sym = TRUE)
```

### Arguments

data	Data matrix
k	Number of nearest neighbors
...	Unused. All parameters to the right of the ... have to be specified by name (e.g. find_knn(data,k,distance = 'cosine'))
query	Query matrix. In knn and knn_asym, query and data are identical
distance	Distance metric to use. Allowed measures: Euclidean distance (default), cosine distance ( $1 - \text{corr}(c_1, c_2)$ ) or rank correlation distance ( $1 - \text{corr}(\text{rank}(c_1), \text{rank}(c_2))$ )
sym	Return a symmetric matrix (as long as query is NULL)?

### Value

A **list** with the entries:

**index** A  $nrow(data) \times k$  **integer matrix** containing the indices of the k nearest neighbors for each cell.

**dist** A  $nrow(data) \times k$  **double matrix** containing the distances to the k nearest neighbors for each cell.

**dist\_mat** A **dgMatrix** if sym == TRUE, else a **dsMatrix** ( $nrow(query) \times nrow(data)$ ). Any zero in the matrix (except for the diagonal) indicates that the cells in the corresponding pair are close neighbors.

### Examples

```
# The default: symmetricised pairwise distances between all rows
pairwise <- find_knn(mtcars, 5L)
image(as.matrix(pairwise$dist_mat))

# Nearest neighbors of a subset within all
mercedeses <- grepl('Merc', rownames(mtcars))
merc_vs_all <- find_knn(mtcars, 5L, query = mtcars[mercedeses, ])
# Replace row index matrix with row name matrix
matrix(
```

```
rownames(mtcars)[merc_vs_all$index],  
nrow(merc_vs_all$index),  
dimnames = list(rownames(merc_vs_all$index), NULL)  
)[, -1] # 1st nearest neighbor is always the same row
```

---

<code>knn.covertree</code>	<i>A not-too-fast but accurate kNN implementation supporting multiple distance measures</i>
----------------------------	---

---

**Description**

A not-too-fast but accurate kNN implementation supporting multiple distance measures

# Index

`dgCMatrix`, [2](#)

`double`, [2](#)

`dsCMatrix`, [2](#)

`find_knn`, [2](#)

`integer`, [2](#)

`knn.covertree`, [3](#)

`knn.covertree-package (knn.covertree)`, [3](#)

`list`, [2](#)

`matrix`, [2](#)