Package 'lazyIris'

September 4, 2015

Title Lazy learning on the iris dataset

Description k-nn applied to iris dataset with visualisations.

Author Phil

Maintainer Phil <phil@parasec.net>

Date 2015-09-04

Version 0.1.0

URL https://github.com/phil8192/lazy-iris

License GPL (>= 2)

Depends R (>= 3.0.0)

Suggests knitr

LazyData true

VignetteBuilder knitr

NeedsCompilation no

R topics documented:

lazyIris-package *lazyIris.*

Description

Lazy learning on the iris dataset.

Details

k-nearest neighbour.

Functionality

- Read in the data.
- Accesses the quality of the data.
- Returns the ten most similar data points in the existing iris data given inputted arguments.
- Visualises the result.

Author(s)

phil <phil@parasec.net>

checkData *Check the iris data.*

Description

Performs any necessary data cleaning.

Usage

```
checkData(iris.data)
```

Arguments

iris.data	The iris.data

Value

Cleaned iris.data

Author(s)

phil

Examples

```
iris.data <- checkData(iris.data)
```

`classifier` *Majority voting.*

Description

Predict a class given a list of neighbours obtained from k-nn using majority voting.

Usage

```
classifier(cls, distance)
```

Arguments

`cls`	A list of neighbour classes found with knn.
`distance`	An equal length list of neighbour distances.

Details

Todo: Distance weighted majority voting/distance kernel.

Value

The majority prediction along with classification confidence.

Author(s)

phil

Examples

```
# form the query instance.
query <- list(
    sepal.length=5.84,
    sepal.width=3.05,
    petal.length=3.76,
    petal.width=1.20)

# get the 10-nearest neighbours.
top.10 <- knn(query, iris.data, 10)

# classify the instance.
prediction <- classifier(top.10$species, top.10$distance)

# print the result.
print(paste("prediction =", prediction$pred,
            "confidence =", prediction$conf))
```

`euclidean` *Euclidean distance metric.*

Description

Euclidean distance between two vectors.

Usage

```
euclidean(v1, v2)
```

Arguments

`v1`	Source vector.
`v2`	Target vector.

Details

Note: There is also a `dist` function in R with the options for euclidian, maximum, manhatten, canberra, binary and minkowski metrics.

Value

Euclidean distance.

Author(s)

phil

Examples

```
# (3,4) with (9,12)
euclidean(c(3, 4), c(9, 12))

# (3,4) with (44,66)
euclidean(c(3, 4), c(44, 66))

# (3,4) with (9,12) and (44,66)
m <- matrix(c(9, 12, 44, 66), ncol=2, byrow=TRUE)
apply(m, 1, euclidean, c(3, 4))
```

iris.data *Iris data.*

Description

The Iris dataset.

Usage

```
data(iris.data)
```

Format

A data.frame.

Details

Descripton from UCI Machine learning repository: This is perhaps the best known database to be found in the pattern recognition literature. Fisher's paper is a classic in the field and is referenced frequently to this day. (See Duda & Hart, for example.) The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are NOT linearly separable from each other.

Author(s)

phil

Source

https://archive.ics.uci.edu/ml/datasets/Iris

References

Lichman, M. (2013). UCI Machine Learning Repository https://archive.ics.uci.edu/ml. Irvine, CA: University of California, School of Information and Computer Science.

knn *K-Nearest-Neighbors.*

Description

Implementation of the k-NN method.

Usage

```
knn(query, feature.space, k = 1, d.fun = euclidean)
```

Arguments

query	A list of labeled parameters to query for.
feature.space	The feature space to query against.
k	The *k* nearest neighbours to return. Default: 1.
d.fun	The distance function to use. Default: euclidean.

Value

The top *k* nearest neighours in the feature space with respect to the query.

Author(s)

phil

Examples

```
# form the query instance.
query <- list(
    sepal.length=5.84,
    sepal.width=3.05,
    petal.length=3.76,
    petal.width=1.20)

# obtain the nearest-neighbour
top.1 <- knn(query, iris.data, 1)
print(top.1, row.names=FALSE)
```

---

loadData *Load data.*

Description

Loads the iris csv data from the inst/extdata directory.

Usage

```
loadData()
```

Value

Iris data as documented in iris.data

Author(s)

phil

Examples

```
iris.data <- loadData()
```

`visualise` *Produce a visualisation of a k-nearest neighbours.*

Description

Given a query, it's resulting neighbours and the original feature space, this function will plot a visualisation of the position of the query in relation to it's neighbours in the feature space.

Usage

```
visualise(feature.space, class.name, query = NULL, neighbours = NULL,
  plot.hist = F, plot.cor = F, ...)
```

Arguments

`feature.space`	The feature space.
`class.name`	The class name. E.g., *species*.
`query`	The query (optional).
`neighbours`	The resulting k-nearest neighbours (optional).
`plot.hist`	If `TRUE` plot a feature distribution histogram.
`plot.cor`	If `TRUE` plot feature correlation.
`...`	Additional arguments to `pairs` plotting function.

Details

Colour code:

- rediris setosa
- greeniris versicolour
- blueiris virginica
- blackquery

The visualisation makes use of R's `pairs` plotting function, more information on which can be viewed in the function's man page: `help(pairs)`.

Author(s)

phil

Examples

```
# plot all the data.
visualise(iris.data, class.name="species", main="iris data", plot.hist=TRUE,
    plot.cor=TRUE)

# do not plot the first 2 features and omit the histogram+correlation plots.
visualise(iris.data[, -(1:2)], class.name="species", main="iris data")

#### visualise k-nearest neighbours.

# form a query.
q <- list(
    sepal.length=5.84,
    sepal.width=3.05,
    petal.length=3.76,
    petal.width=1.20)

# get the 10-nearest neighbours
top.10 <- knn(q, iris.data, 10)

# visualise the neighbours and query point.
visualise(iris.data, class.name="species", query=q, neighbours=top.10,
    main="iris data neighbours", plot.hist=TRUE, plot.cor=FALSE)
```

Index