

## Clustering Group T4-2

1.0

Generated by Doxygen 1.9.1



<b>1 Hierarchical Index</b>	<b>1</b>
1.1 Class Hierarchy	1
<b>2 Class Index</b>	<b>3</b>
2.1 Class List	3
<b>3 Class Documentation</b>	<b>5</b>
3.1 clustering.Clustering Class Reference	5
3.1.1 Member Function Documentation	5
3.1.1.1 cluster()	5
3.1.1.2 house_load()	6
3.1.1.3 load_data()	6
3.1.1.4 pyc_metric()	6
3.2 dbscan.DBSCANClustering Class Reference	6
3.2.1 Member Function Documentation	7
3.2.1.1 cluster()	7
3.2.1.2 package()	7
3.3 indices.Indices Class Reference	8
3.4 kmeans.kmeansClustering Class Reference	8
3.4.1 Member Function Documentation	8
3.4.1.1 cluster()	9
3.5 kmedians.kmediansClustering Class Reference	9
3.5.1 Member Function Documentation	9
3.5.1.1 cluster()	10
3.6 kmedoids.kmedoidsClustering Class Reference	10
3.6.1 Member Function Documentation	10
3.6.1.1 cluster()	11
3.6.1.2 package()	11
<b>Index</b>	<b>13</b>



# Chapter 1

## Hierarchical Index

### 1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

clustering.Clustering . . . . .	5
dbscan.DBSCANClustering . . . . .	6
kmeans.kmeansClustering . . . . .	8
kmedians.kmediansClustering . . . . .	9
kmedoids.kmedoidsClustering . . . . .	10
indices.Indices . . . . .	8



## Chapter 2

# Class Index

### 2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">clustering.Clustering</a>	5
<a href="#">dbscan.DBSCANClustering</a>	6
<a href="#">indices.Indices</a>	8
<a href="#">kmeans.kmeansClustering</a>	8
<a href="#">kmedians.kmediansClustering</a>	9
<a href="#">kmedoids.kmedoidsClustering</a>	10



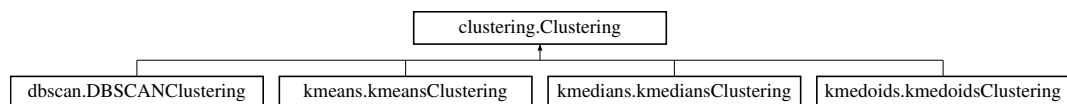


## Chapter 3

# Class Documentation

### 3.1 clustering.Clustering Class Reference

Inheritance diagram for clustering.Clustering:



#### Public Member Functions

- `def __init__(self, metric, dataset)`
- `def pyc\_metric(self, metric)`
- `def load\_data(self)`
- `def house\_load(self, path, skip=1)`
- `def cluster(self)`

#### Public Attributes

- `data`
- `dataset`
- `labels`

#### 3.1.1 Member Function Documentation

##### 3.1.1.1 `cluster()`

```
def clustering.Clustering.cluster (
    self )
```

does nothing in the meta class.  
needs to be implemented in the inheriting cluster algorithm classes

### 3.1.1.2 house\_load()

```
def clustering.Clustering.house_load (
    self,
    path,
    skip = 1 )

loads the housevotes dataset and encodes it using One-Hot-Encoding
@param path filepath to the dataset
@param skip number of lines that get skipped when reading in a file
@return One-Hot-Encoded housevotes dataset
```

### 3.1.1.3 load\_data()

```
def clustering.Clustering.load_data (
    self )

loads in a dataset, standardises it and sets it as self.data attribute
```

### 3.1.1.4 pyc\_metric()

```
def clustering.Clustering.pyc_metric (
    self,
    metric )

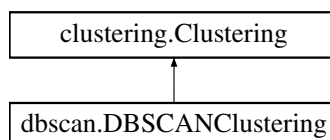
returns a distance metric which is usable by the pyclustering algorithms
@param distance metric string. allowed: "euclidean", "manhattan", "chebyshev", "cosine"
@return pyclustering distance_metric object, None when distance is not supported
```

The documentation for this class was generated from the following file:

- clustering.py

## 3.2 dbscan.DBSCANClustering Class Reference

Inheritance diagram for dbscan.DBSCANClustering:



## Public Member Functions

- `def __init__ (self, metric, dataset)`
- `def cluster (self, eps, minPts)`
- `def package (self, labels)`

## Public Attributes

- `metric`

### 3.2.1 Member Function Documentation

#### 3.2.1.1 cluster()

```
def dbscan.DBSCANClustering.cluster (
    self,
    eps,
    minPts )
```

clustering method. Will execute clustering on the data saved in `self.data` with the metric given in `self.metric`  
params are the same as in the DBSCAN paper  
@param eps Distance for the Eps-Neighbourhood  
@param minPts Minimal number of points in a cluster  
@returns formatted clustered data

#### 3.2.1.2 package()

```
def dbscan.DBSCANClustering.package (
    self,
    labels )
```

rearranges the result to a format similar to the one of the pyclustering algorithms  
allows for easier access in the streamlit interface  
@param labels cluster labels DBSCAN assigns to a point  
@returns clusters as list of lists of indices of points and noise as list of indices of points

The documentation for this class was generated from the following file:

- `dbscan.py`

### 3.3 indices.Indices Class Reference

#### Public Member Functions

- `def __init__(self, cluster_calc, cluster_label)`
- `def index_external(self, index)`
- `def index_internal(self, index)`

#### Public Attributes

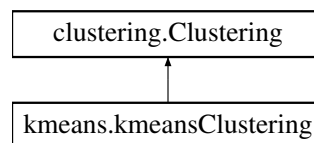
- `cluster_calc`
- `cluster_label`

The documentation for this class was generated from the following file:

- `indices.py`

### 3.4 kmeans.kmeansClustering Class Reference

Inheritance diagram for `kmeans.kmeansClustering`:



#### Public Member Functions

- `def __init__(self, metric, dataset)`
- `def cluster(self, k, plusplus=True)`

#### Public Attributes

- `data`
- `metric`

#### 3.4.1 Member Function Documentation

### 3.4.1.1 cluster()

```
def kmeans.kmeansClustering.cluster (
    self,
    k,
    plusplus = True )

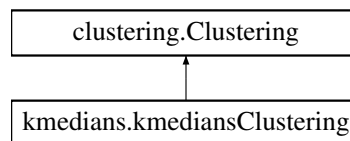
clustering method. Will execute clustering on the data saved in self.data with the metric
given in self.metric
@param k number of clusters that are generated
@param plusplus will use k++ initialiser if true
@returns clusters as list of lists of indices of points and final cluster centers
```

The documentation for this class was generated from the following file:

- kmeans.py

## 3.5 kmedians.kmediansClustering Class Reference

Inheritance diagram for kmedians.kmediansClustering:



### Public Member Functions

- def `__init__` (self, metric, dataset)
- def `cluster` (self, k)

### Public Attributes

- data
- metric

### 3.5.1 Member Function Documentation

### 3.5.1.1 cluster()

```
def kmedians.kmediansClustering.cluster (
    self,
    k )
```

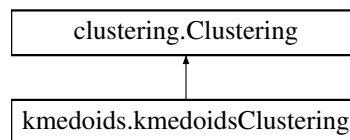
clustering method. Will execute clustering on the data saved in self.data with the metric given in self.metric  
@param k number of clusters that are generated  
@returns clusters as list of lists of indices of points and final cluster medians

The documentation for this class was generated from the following file:

- kmedians.py

## 3.6 kmedoids.kmedoidsClustering Class Reference

Inheritance diagram for kmedoids.kmedoidsClustering:



### Public Member Functions

- def `__init__` (self, metric, dataset)
- def `cluster` (self, k, init="k-medoids++")
- def `package` (self, labels)

### Public Attributes

- `data`
- `metric`

### 3.6.1 Member Function Documentation

### 3.6.1.1 cluster()

```
def kmedoids.kmedoidsClustering.cluster (
    self,
    k,
    init = "k-medoids++" )

clustering method. Will execute clustering on the data saved in self.data with the metric
given in self.metric
@param k number of clusters that are generated
@param init initialisation parameter. Standard: "k-medoids++"
@returns clusters as list of lists of indices of points, final cluster centers
```

### 3.6.1.2 package()

```
def kmedoids.kmedoidsClustering.package (
    self,
    labels )

rearranges the result to a format similar to the one of the pyclustering algorithms
allows for easier access in the streamlit interface
@param labels labels returned from the KMedoids algorithm
@returns clusters formatted similarly to the pyclustering algorithms
```

The documentation for this class was generated from the following file:

- kmedoids.py





# Index

## cluster

- clustering.Clustering, [5](#)
- dbscan.DBSCANClustering, [7](#)
- kmeans.kmeansClustering, [8](#)
- kmedians.kmediansClustering, [9](#)
- kmedoids.kmedoidsClustering, [10](#)

## clustering.Clustering, [5](#)

- cluster, [5](#)
- house\_load, [5](#)
- load\_data, [6](#)
- pyc\_metric, [6](#)

## dbscan.DBSCANClustering, [6](#)

- cluster, [7](#)
- package, [7](#)

## house\_load

- clustering.Clustering, [5](#)

## indices.Indices, [8](#)

## kmeans.kmeansClustering, [8](#)

- cluster, [8](#)

## kmedians.kmediansClustering, [9](#)

- cluster, [9](#)

## kmedoids.kmedoidsClustering, [10](#)

- cluster, [10](#)
- package, [11](#)

## load\_data

- clustering.Clustering, [6](#)

## package

- dbscan.DBSCANClustering, [7](#)
- kmedoids.kmedoidsClustering, [11](#)

## pyc\_metric

- clustering.Clustering, [6](#)