scikit-query

Release 0.4

scikit-query developers

Nov 08, 2023

GENERAL INFORMATION

Clustering aims to group data into clusters without the help of labels, unlike classification algorithms. A well-known shortcoming of clustering algorithms is that they rely on an objective function geared toward specific types of clusters (convex, dense, well-separated), and hyperparameters that are hard to tune. Semi-supervised clustering mitigates these problems by injecting background knowledge in order to guide the clustering. Active clustering algorithms analyze the data to select interesting points to ask the user about, generating constraints that allow fast convergence towards a user-specified partition.

scikit-query is a library of active query strategies for constrained clustering inspired by scikit-learn and the now inactive active-semi-supervised-clustering library by Jakub Švehla.

It is focused on algorithm-agnostic query strategies, i.e. methods that do not rely on a particular clustering algorithm. From an input dataset, they produce a set of constraints by making insightful queries to an oracle. A variant for incremental constrained clustering is provided for applicable algorithms, taking a data partition into account.

Note: This project is under active development.

CHAPTER

ONE

THE BASICS OF SCIKIT-QUERY

This page explains how to install the library and use it in Python.

1.1 Installation

The library can be installed using `pip` :

```
pip install scikit-query
```

1.2 Using the library

The library has modules for each kind of constraint (pairwise or triplet), plus another one containing the oracles. They can be imported as below :

```
from skquery.pairwise import *
from skquery.triplet import *
from skquery.oracle import MLCLOracle
```

This will allow to use the constraint selection algorithms as well as the `MLCLOracle` to answer queries about pairwise constraints.

1.3 Making queries

All algorithms have a `fit` method taking as arguments a matrix of n points having m features and an oracle (typically from the `skquery.oracle` module). The oracle must have a `query` method returning a boolean.

```
qs = AIPC()
oracle = MLCLOracle()
constraints = qs.fit(dataset.data, oracle)
```

The oracle's `truth` attribute can support a ground truth labeling of the data, which will be used to automatically answer queries. If none is provided, it will ask queries to the user through the CLI.

```
oracle = MLCLOracle(truth=labels)
```

The constraints are returned as a dictionary of constraint types paired with lists of selected constraints. The table below describes how the constraint dictionary is structured.

Table 1: Conventions used for constraint storage

Type	Key	Constraint format
Must-link	ml	(int, int)
Cannot-link	cl	(int, int)
Triplet	triplet	(int, int, int)

CHAPTER

TWO

CONSTRAINTS IN CLUSTERING

Constrained clustering aims to integrate user knowledge of the data to produce partitions that better fit its expectations. This knowledge is expressed as relations involving the cluster labels of particular points or subsets of the data, named **constraints**. In the following, we note *y_x* the cluster label of a point *x* belonging to the dataset.

The constraints selected by the algorithms of this library are described below.

2.1 Pairwise constraints

Must-link and **cannot-link** (ML/CL) constraints, also referred to as pairwise constraints, establish a relation between two data points : they must be in the same cluster (must-link) or in separate clusters (cannot-link). Thanks to their simplicity and the fact that many complex constraints can be decomposed into a set of ML/CL constraints, they are most widely studied constraints ; most active clustering algorithms focus on selecting them efficiently.

Formally, they are expressed as an (in)equality between cluster labels :

$$y_i = y_j \quad (ML)$$
$$y_i \neq y_j \quad (CL)$$

Querying a pairwise constraint between two points *i* and *j* is simply asking : "Should *i" and *j* be in the same cluster ?". If the user answers "yes", then *(i,j)* is a *must-link* constraint. If they answer "no", it is a *cannot-link* constraint.

2.2 Triplet constraints

Triplet constraints, sometimes called relative constraints, define the relationship between three data points : a reference point *a*, a positive point *p* and a negative point *n*. The positive point *p* is assumed to be more similar to *a* than *n* is. Formally, it is expressed as follows:

$$y_a = y_n \implies y_a = y_p$$
$$y_a \neq y_p \implies y_a \neq y_n$$

Querying a triplet constraint *(i,j,k)* amounts to asking the user : "Is *i* more similar to *j* than to *k* ?" The answer to the query will determine the roles of *j* and *k* in the constraint. Indeed, "no" would mean that *j* corresponds to the negative point *n*, and *k* corresponds to *p*, while "yes" would mean the reverse.

CHAPTER

THREE

PAIRWISE CONSTRAINT SELECTION (`SKQUERY.PAIRWISE`)

This module regroups methods that query pairwise (i.e. *must-link/cannot-link*) constraints.

CHAPTER

FOUR

TRIPLET CONSTRAINT SELECTION (SKQUERY.TRIPLET)

This module regroups methods that query triplet constraints.

CHAPTER

FIVE

ORACLES (SKQUERY.ORACLE)

This module contains simple oracles that answer the queries made by active clustering algorithms. They can be used either for direct interaction with a user, or for producing automated answers based on a labeling of the data.

class skquery.oracle.**MLCLOracle**(*budget=10*, *truth=None*)

Oracle for pairwise queries.

5.1 Parameters

budget
[int, default=10] Maximum number of queries the oracle will answer.

truth
[array-like, default=None] Ground truth labeling that emulates a human oracle.

5.2 Attributes

queries
[int] Number of queries answered by the oracle so far.

budget
[int] Maximum number of queries the oracle will answer.

truth
[array-like, default=None] Ground truth labeling that emulates a human oracle.

query(*i*, *j*)

Query the oracle to find out whether two points should be in the same cluster.

5.2.1 Parameters

i
[int] Index of first data point.

j
[int] Index of second data point.

5.2.2 Returns

answer
[bool] Answer to the query.

5.2.3 Raises

NoAnswerError
If the oracle doesn't give an answer, i.e. they don't know what to answer.

class skquery.oracle.**TripletOracle**(*budget=10*, *truth=None*)

Oracle for triplet queries. Inspired from the formalization in [1].

5.3 Parameters

budget
[int, default=10] Maximum number of queries the oracle will answer.

truth
[array-like, default=None] Ground truth labeling that emulates a human oracle.

5.4 Attributes

queries
[int] Number of queries answered by the oracle so far.

budget
[int] Maximum number of queries the oracle will answer.

truth
[array-like, default=None] Ground truth labeling that emulates a human oracle.

5.5 References

[1] Xiong, S., Pei, Y., Rosales, R., Fern, X. Z. Active Learning from Relative Comparisons, IEEE Transactions on Knowledge and Data Engineering, vol. 27, n 12, p. 3166-3175, dec. 2015, doi: 10.1109/TKDE.2015.2462365.

query(*i*, *j*, *k*)

Query the oracle on the relation between a triplet of points.

5.5.1 Parameters

i
: [int] Index of first data point, the reference.

j
: [int] Index of second data point, the assumed positive example.

k
: [int] Index of third data point, the assumed negative example.

5.5.2 Returns

answer
: [bool] Answer to the query.

5.5.3 Raises

NoAnswerError
: If the oracle doesn't give an answer, i.e. they don't know what to answer.

CHAPTER

SIX

INFORMATIVE SUBSET SELECTION (SKQUERY.SELECT)

This module regroups preprocessing methods to select informative subsets of a dataset on which query strategies can be applied.

CHAPTER

SEVEN

CUSTOM EXCEPTIONS

exception skquery.exceptions.**EmptyBudgetError**

Exception to be raised when the number of queries made by an active method has reached the budget limit.

exception skquery.exceptions.**NoAnswerError**

Exception to be raised when the oracle doesn't answer a query, e.g. when they don't know what to answer.

exception skquery.exceptions.**QueryNotFoundError**

Exception to be raised when an active method is unable to perform further queries, e.g. when all points in the dataset have been selected for query.

PYTHON MODULE INDEX

S

INDEX