Hogyan fejlesszünk Python csomagot?

Tanulságok egy klaszterező csomag létrehozásából

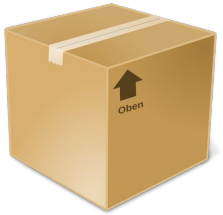
Fülöp Árpád, 

Tartalom

Célkitűzés

Eszközök

Csomagolás



Célkitűzés

Példa

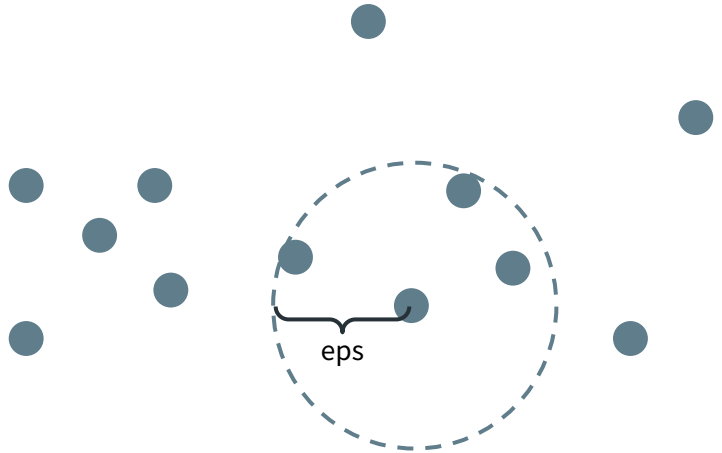
Újságcikkeket csoportosítunk tartalom alapján

Napi 300 újságcikket gyűjtünk

Feladat: Klaszterezzük a mai cikkeket!

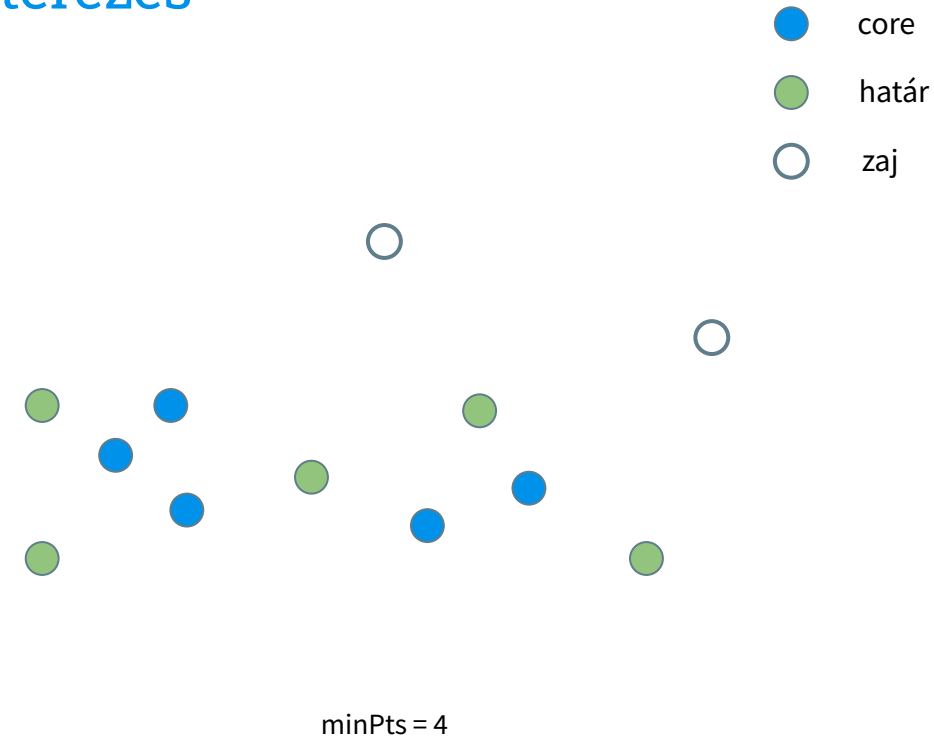
DBSCAN klaszterezés

DBSCAN klaszterezés

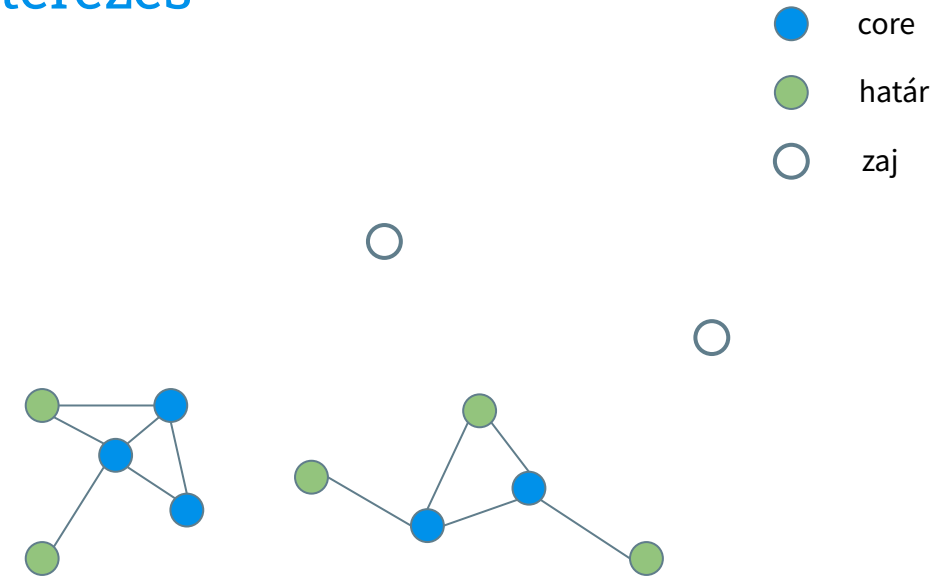


minPts = 4

DBSCAN klaszterezés



DBSCAN klaszterezés



minPts = 4

Példa (folyt.)

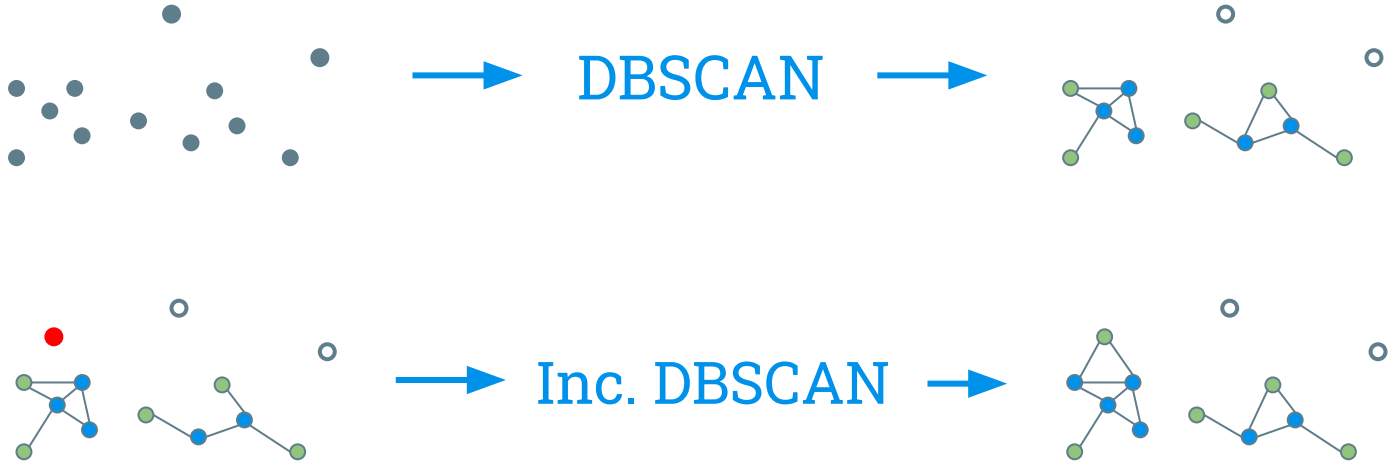
Újságcikkeket csoportosítunk tartalom alapján

Napi 300 újságcikket gyűjtünk

Feladat: Minden nap klaszterezzük az összes cikket!

- 1 hónap: ~10 ezer
- 1 év: ~120 ezer

IncrementalDBSCAN klaszterezés



Az ígéret

Gyorsabb x db pontot hozzátenni N db pont klaszterezéséhez, mint N+x db pontot klaszterezni

(Ha x sokkal kisebb, mint N)

Az ígéret másik oldala

elvenni

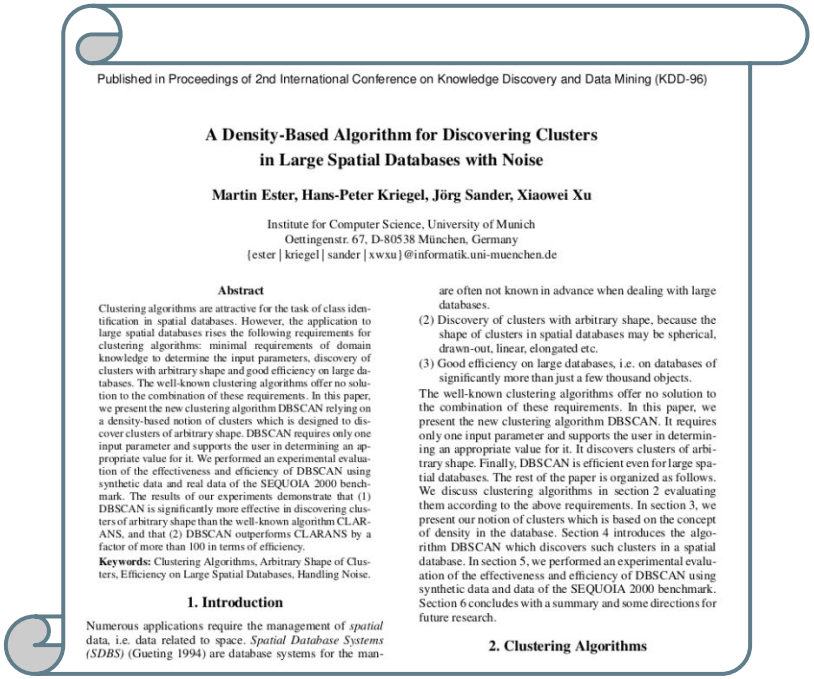
Gyorsabb x db pontot ~~hozzátenni~~ N db pont

ből N-x

klaszterezéséhez, mint ~~N+x~~ db pontot klaszterezni

(Ha x sokkal kisebb, mint N)

DBSCAN implementáció



Published in Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining (KDD-96)

A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu

Institute for Computer Science, University of Munich
Oettingenstr. 67, D-80538 München, Germany
{ester | kriegel | sander | xwxu}@informatik.uni-muenchen.de

Abstract

Clustering algorithms are attractive for the task of class identification in spatial databases. However, the application to large spatial databases rises the following requirements for clustering algorithms: minimal requirements of domain knowledge to determine the input parameters, discovery of clusters with arbitrary shape and good efficiency on large databases. The well-known clustering algorithms offer no solution to the combination of these requirements. In this paper, we present the new clustering algorithm DBSCAN relying on a density-based notion of clusters which is designed to discover clusters of arbitrary shape. DBSCAN requires only one input parameter and supports the user in determining an appropriate value for it. We performed an experimental evaluation of the effectiveness and efficiency of DBSCAN using synthetic data and real data of the SEQUOIA 2000 benchmark. The results of our experiments demonstrate that (1) DBSCAN is significantly more effective in discovering clusters of arbitrary shape than the well-known algorithm CLARANS, and that (2) DBSCAN outperforms CLARANS by a factor of more than 100 in terms of efficiency.

Keywords: Clustering Algorithms, Arbitrary Shape of Clusters, Efficiency on Large Spatial Databases, Handling Noise.

1. Introduction

Numerous applications require the management of *spatial* data, i.e. data related to space. *Spatial Database Systems (SDBS)* (Gueting 1994) are database systems for the manare often not known in advance when dealing with large databases.

(2) Discovery of clusters with arbitrary shape, because the shape of clusters in spatial databases may be spherical, drawn-out, linear, elongated etc.

(3) Good efficiency on large databases, i.e. on databases of significantly more than just a few thousand objects.

The well-known clustering algorithms offer no solution to the combination of these requirements. In this paper, we present the new clustering algorithm DBSCAN. It requires only one input parameter and supports the user in determining an appropriate value for it. It discovers clusters of arbitrary shape. Finally, DBSCAN is efficient even for large spatial databases. The rest of the paper is organized as follows. We discuss clustering algorithms in section 2 evaluating them according to the above requirements. In section 3, we present our notion of clusters which is based on the concept of density in the database. Section 4 introduces the algorithm DBSCAN which discovers such clusters in a spatial database. In section 5, we performed an experimental evaluation of the effectiveness and efficiency of DBSCAN using synthetic data and data of the SEQUOIA 2000 benchmark. Section 6 concludes with a summary and some directions for future research.

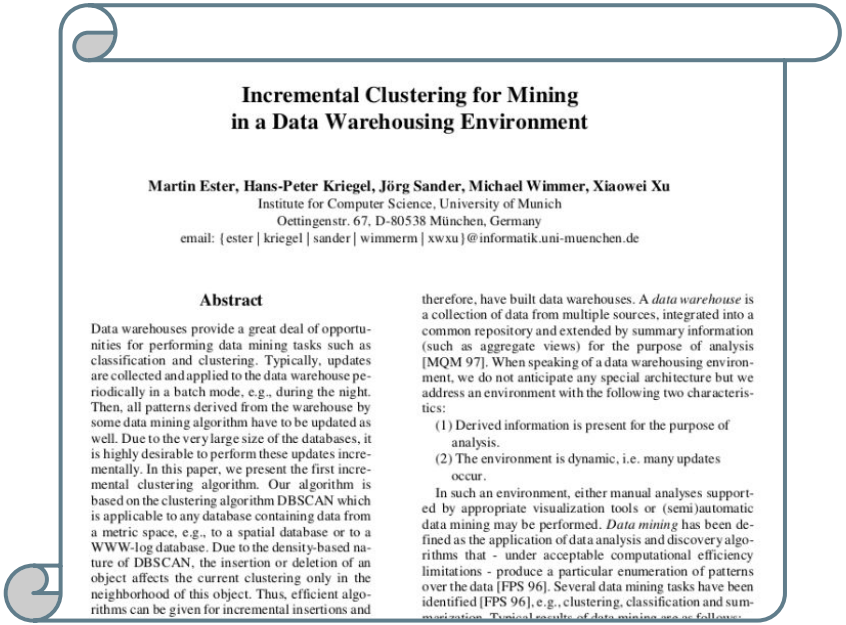
2. Clustering Algorithms

```
[1]: !pip install scikit-learn
```

...

```
[2]: from sklearn.cluster import DBSCAN
```

IncrementalDBSCAN implementáció



Incremental Clustering for Mining in a Data Warehousing Environment

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Michael Wimmer, Xiaowei Xu
Institute for Computer Science, University of Munich
Oettingenstr. 67, D-80538 München, Germany
email: {ester | kriegel | sander | wimmerm | xwxu}@informatik.uni-muenchen.de

Abstract

Data warehouses provide a great deal of opportunities for performing data mining tasks such as classification and clustering. Typically, updates are collected and applied to the data warehouse periodically in a batch mode, e.g., during the night. Then, all patterns derived from the warehouse by some data mining algorithm have to be updated as well. Due to the very large size of the databases, it is highly desirable to perform these updates incrementally. In this paper, we present the first incremental clustering algorithm. Our algorithm is based on the clustering algorithm DBSCAN which is applicable to any database containing data from a metric space, e.g., to a spatial database or to a WWW-log database. Due to the density-based nature of DBSCAN, the insertion or deletion of an object affects the current clustering only in the neighborhood of this object. Thus, efficient algorithms can be given for incremental insertions and

therefore, have built data warehouses. A *data warehouse* is a collection of data from multiple sources, integrated into a common repository and extended by summary information (such as aggregate views) for the purpose of analysis [MQM 97]. When speaking of a data warehousing environment, we do not anticipate any special architecture but we address an environment with the following two characteristics:

(1) Derived information is present for the purpose of analysis.
(2) The environment is dynamic, i.e. many updates occur.

In such an environment, either manual analyses supported by appropriate visualization tools or (semi)automatic data mining may be performed. *Data mining* has been defined as the application of data analysis and discovery algorithms that - under acceptable computational efficiency limitations - produce a particular enumeration of patterns over the data [FPS 96]. Several data mining tasks have been identified [FPS 96], e.g., clustering, classification and summarization. Typical results of data mining are as follows:

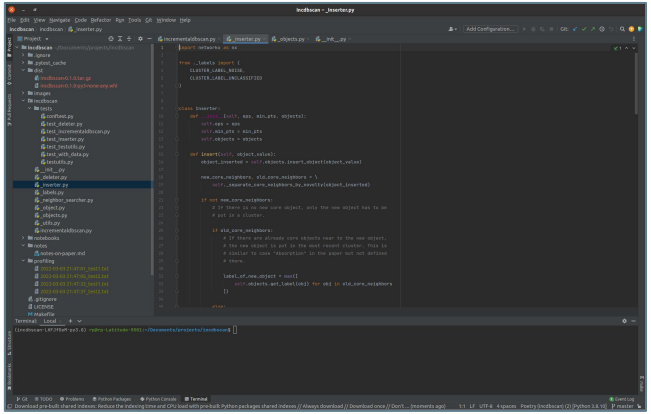
```
[1]: # ???
```

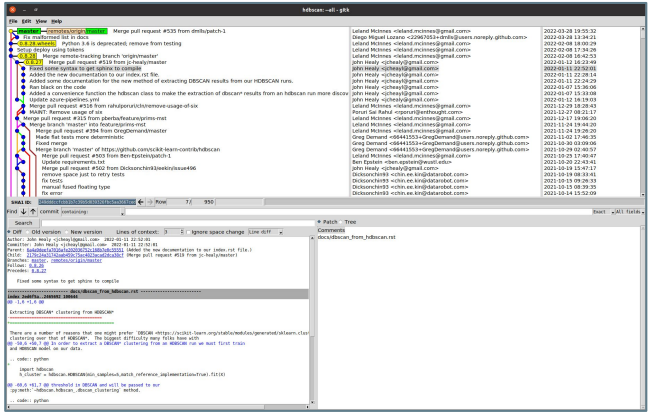
Tanács #1

Ha hasznos és népszerű csomagot akarsz csinálni, oldj meg egy általános problémát, amit más még nem oldott meg

Eszközök

Alapfelszerelés

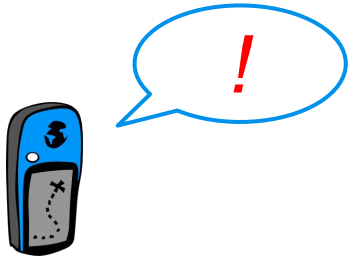




Fejlesztői környezet

Git verziókezelő

Tesztelők



- A helyesség ellenőrzése
- Tesztesetek felállítása
- Az elvárt eredményhez hasonlítás
  - ML kódoknál ez nem triviális

Pytest: https://pytest.org/

Az elvárt eredmény a cikk alapján

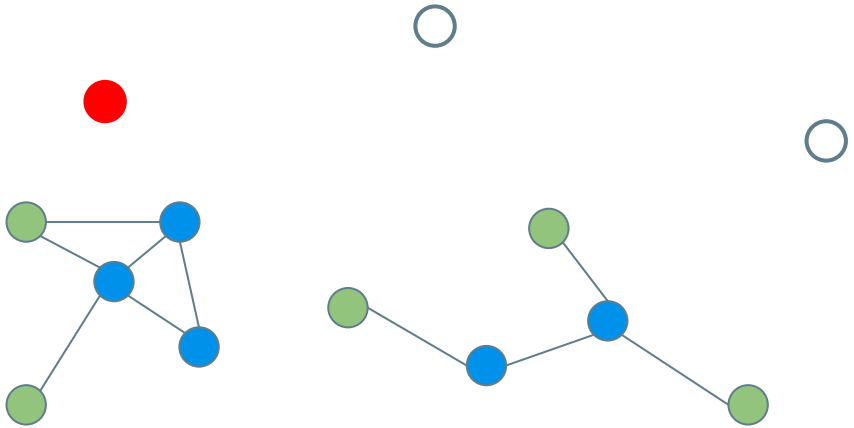
When inserting an object p into the database D, we can distinguish the following cases:

(1) (*Noise*)
$UpdSeed_{Ins}$ is empty, i.e. there are no "new" core objects after insertion of p. Then, p is a noise object and nothing else is changed.

(2) (Creation)
$UpdSeed_{Ins}$ contains only core objects which did not belong to a cluster before the insertion of p, i.e. they were noise objects or equal to p, and a new cluster containing these noise objects as well as p is created.

(3) (Absorption)
$UpdSeed_{Ins}$ contains core objects which were members of exactly one cluster C before the insertion. The object p and possibly some noise objects are absorbed into cluster C.

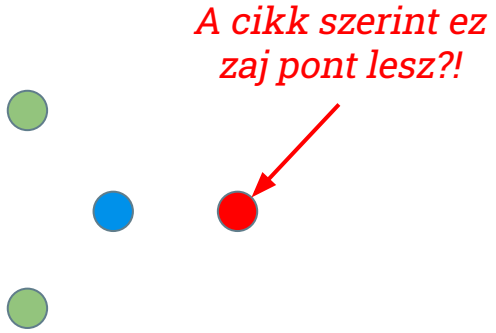
(4) (Merge)
$UpdSeed_{Ins}$ contains core objects which were members of several clusters before the insertion. All these clusters and the object p are merged into one cluster.



A probléma

A cikk megbízhatatlan…

(1) (*Noise*)
$UpdSeed_{Ins}$ is empty, i.e. there are no "new" core objects after insertion of p. Then, p is a noise object and nothing else is changed.



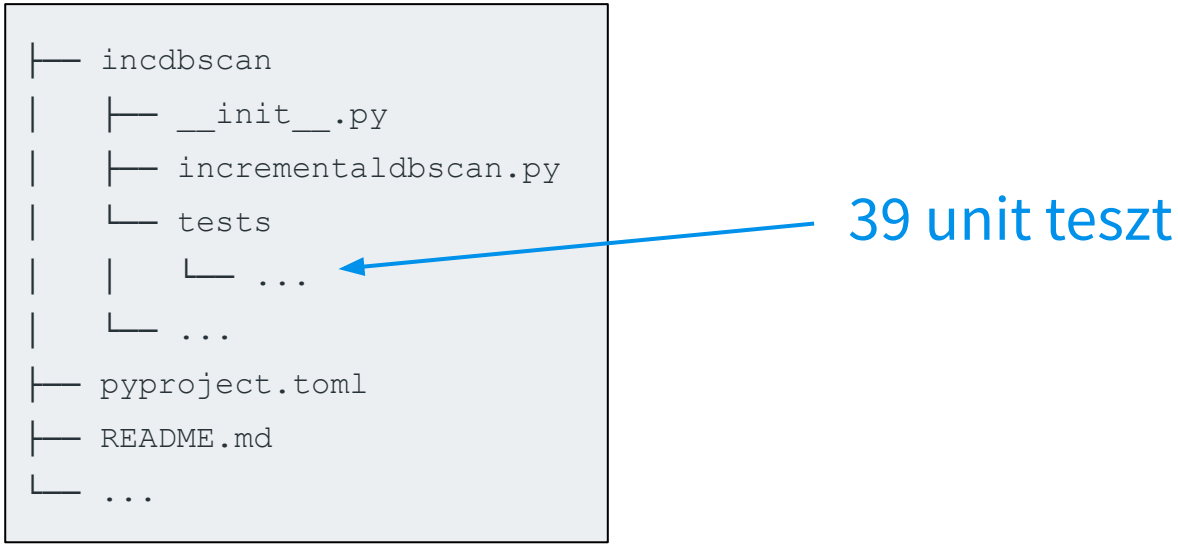
minPts = 3

Az elvárt eredmény inkább – a DBSCAN

Legyen az elvárt eredmény az, amit a DBSCAN csinálna

Ebben az esetben: az új pont legyen határ pont

Tesztek


├── incdbscan
│   ├── __init__.py
│   ├── incrementaldbscan.py
│   └── tests
│   │   └── ...
│   └── ...
├── pyproject.toml
├── README.md
└── ...
39 unit teszt

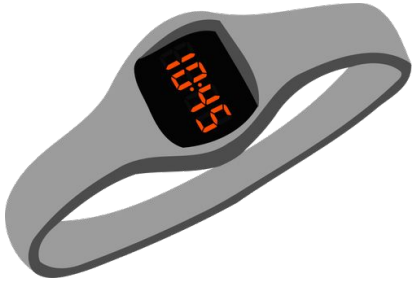
Tanács #2

Tesztelj sokat, a tesztek meg fogják menteni a projektedet

Tanács #3

Ne bízz a cikkben, amit implementálsz

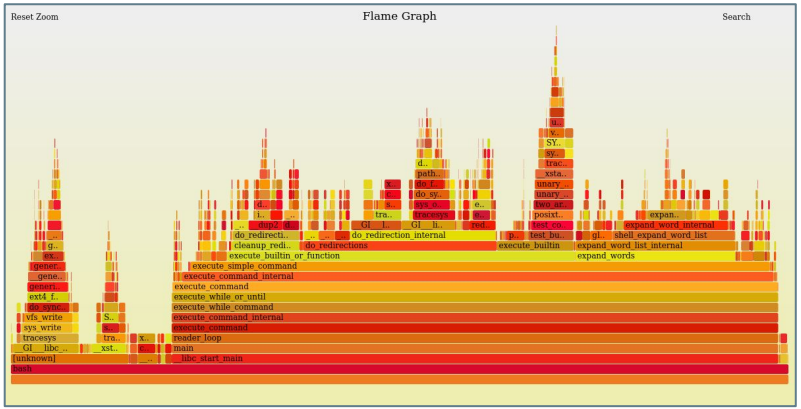
Profilozók



A kód hatékonyságának mérése

- Gyorsaság
- Memóriahasználat

line_profiler: [link]



https://github.com/brendangregg/FlameGraph

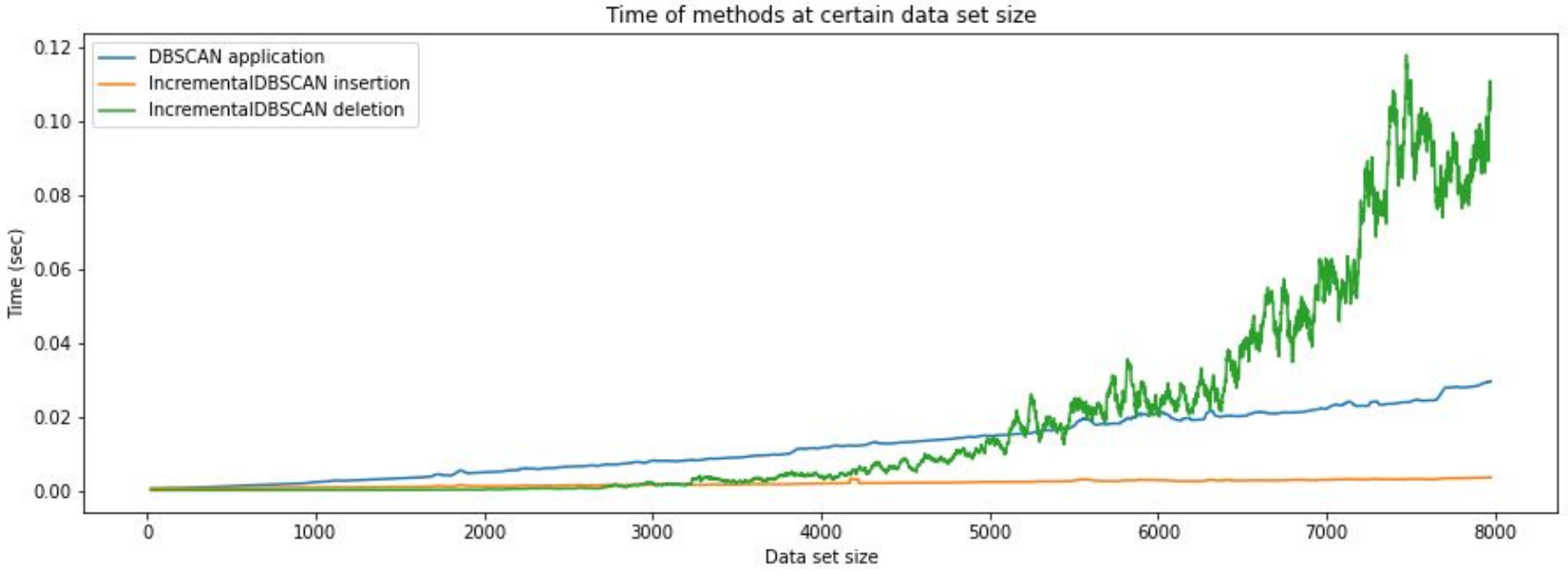
line_profiler – sorról sorra

```
Total time: 4.10178 s
File: /home/xyz/incdbscan/incdbscan/_inserter.py
Function: insert at line 15

Line #    Hits         Time  Per Hit   % Time  Line Contents
==============================================================
    15                                           def insert(self, object_value):
    16    1517   1605591.0   1058.4    39.1          object_inserted = self.objects.insert_object(object_value)
    17
    18    1517       806.0      0.5     0.0          new_core_neighbors, old_core_neighbors = \
    19    1517    335443.0    221.1     8.2              self._separate_core_neighbors_by_novelty(object_inserted)
    20
...
```

line_profiler – sorról sorra

```
Total time: 4.10178 s
File: /home/xyz/incdbscan/incdbscan/_inserter.py
Function: insert at line 15

Line #    Hits         Time  Per Hit   % Time  Line Contents
==============================================================
    15                                           def insert(self, object_value):
    16      1517    1605591.0   1058.4    39.1          object_inserted = self.objects.insert_object(object_value)
    17
    18      1517        806.0      0.5     0.0          new_core_neighbors, old_core_neighbors = \
    19      1517     335443.0    221.1     8.2              self._separate_core_neighbors_by_novelty(object_inserted)
    20
...
```

Teljesítménymérés



Fejlesztési irányok

- Pontok törlésének gyorsítása
  - Jobb adatszerkezettel
  - Alacsonyabb szintű kóddal (pl. Cython)
- Pontok hozzáadása batchekben

Tanács #4 (kutatóknak)

Legyen reprodukálható az eredményed

Statikus kódelemzők, linterek

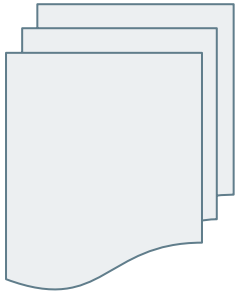
- Kódolási standardok ellenőrzése
- Egyszerűsítési lehetőségek jelzése
- Hibás vagy gyanús kód jelzése

Pylint: https://pylint.pycqa.org

Csomagolás

Csomagkészítés folyamata

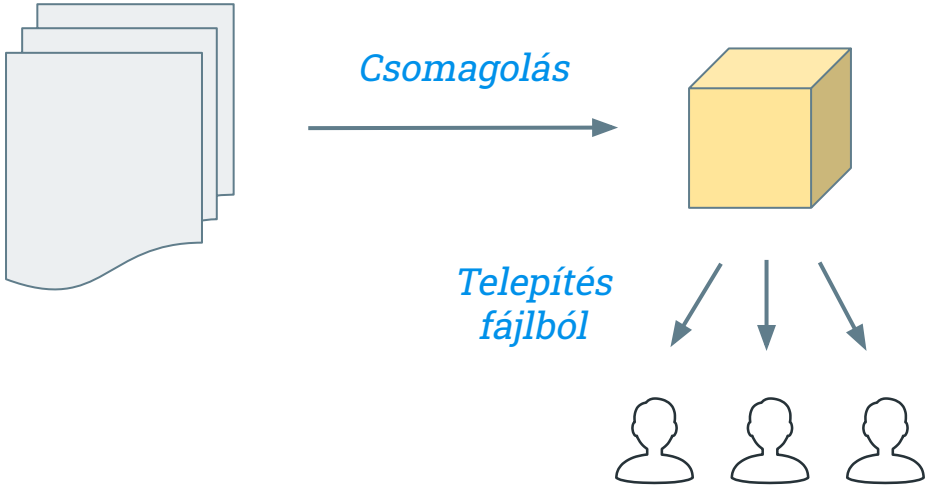
Forrásfájlok



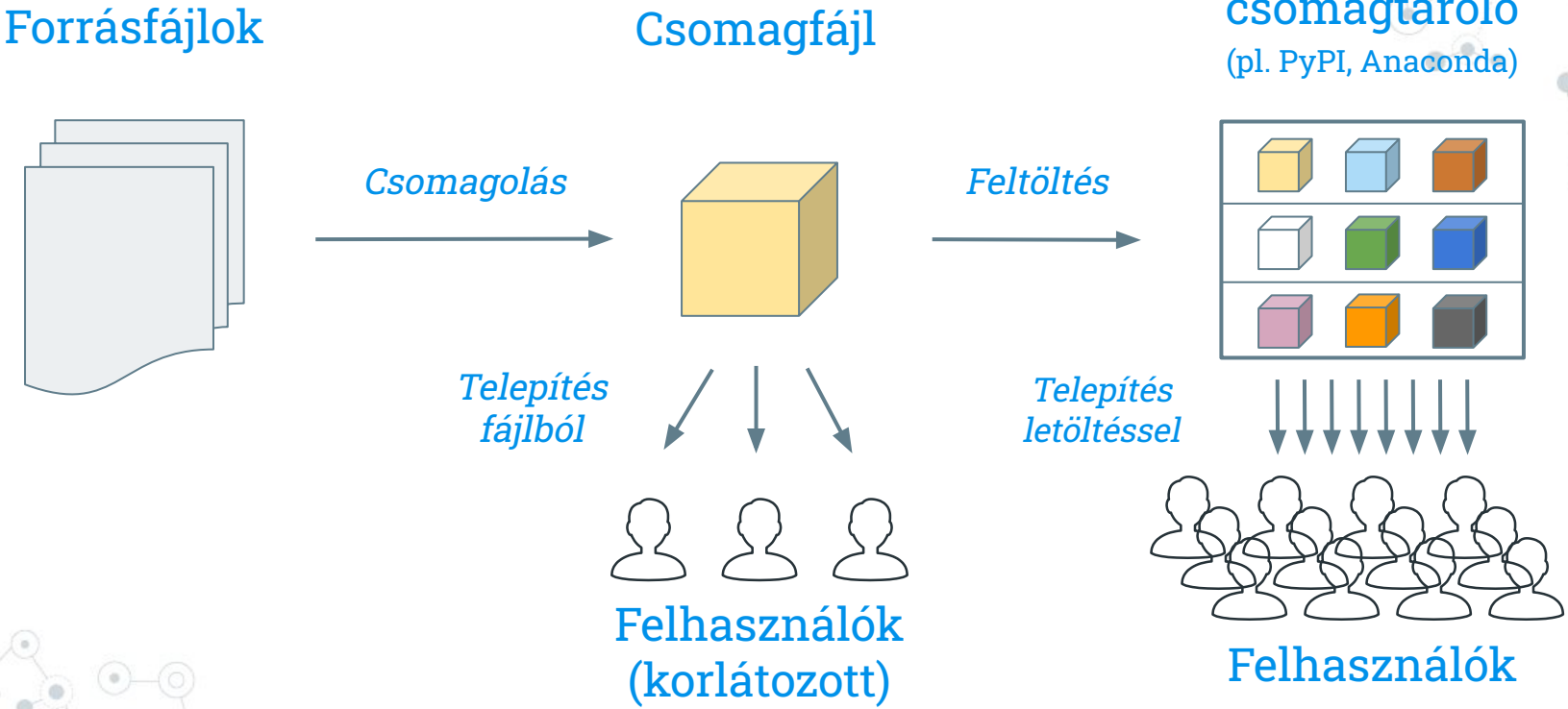
Csomagkészítés folyamata

Forrásfájlok

Csomagfájl



Csomagkészítés folyamata



Poetry

- Függőségkezelés
- Virtuális környezet kezelése
- Buildelés és csomagolás
- Publikálás

https://python-poetry.org/

Függőségkezelés Poetryvel

pyproject.toml

```
[tool.poetry.dependencies]
networkx = "^2.5.1"
numpy = "^1.20.3"
python = ">=3.7.1,<4.0"
scikit-learn = "^1.0"
sortedcontainers = "^2.4.0"
xxhash = "^2.0.0"
```

Python verzió meghatározása

Csomagolás és buildelés Poetryvel

pyproject.toml

```
[build-system]
requires = ["poetry_core>=1.0.0"]
build-backend = "poetry.core.masonry.api"
```

Csomagolás és buildelés: `poetry build`

Publikálás Poetryvel

Publikálás a PyPI-ra: `poetry publish`

Ezek után bárki egy paranccsal telepítheti a csomagot:

```
pip install incdbscan
```

Forráskód megosztása

GitHub, Gitlab, Bitbucket, …

Döntés a license-ről

- Pl. BSD 3-Clause License (pl. `pandas`, `scikit-learn`)
- `LICENSE` fájl

Tanács #5

Ne hagyd otthon a Poetryt, rengeteg mindenben segít

Hírverés

LinkedIn

reddit

May 3-4, 2022 · Machine Learning · Data Science · AI

BUDAPEST ML FORUM

in partnership with Continental AI Development Center

Letöltési statisztikák

```
arpad@Latitude:~$ pypistats python_minor incdbscan
| category | percent | downloads |
|:---------|--------:|----------:|
| null     |  86.84% |       101 |
| 3.8      |   6.14% |         7 |
| 3.7      |   3.51% |         4 |
| 3.10     |   2.63% |         3 |
| 3.9      |   0.88% |         1 |
| Total    |         |       116  |

Date range: 2022-03-06 - 2022-04-28
```

Kérdések?

Elérhetőségek

arpad.fulop.353@gmail.com

https://github.com/DataOmbudsman/incdbscan

https://pypi.org/project/incdbscan/