pydist2 Documentation

Release 0.0.1

Mahmoud Harmouch

Oct 02, 2022

CONTENTS:

1 pydist2 **1**
- 1.1 Usage . . . 1
- 1.2 Installation . . . 1
- 1.3 Supported Python versions . . . 2
- 1.4 Progress & Features . . . 2
- 1.5 Todo list . . . 2

2 Installation Guide **3**
- 2.1 Stable release . . . 3
- 2.2 From sources . . . 3

3 pydist2 tutorial **5**

4 pydist2 guide **7**
- 4.1 Special Distances . . . 7
- 4.2 pdist1 and pdist2 . . . 8

5 pydist2 **11**
- 5.1 pydist2 package . . . 11
  - 5.1.1 Submodules . . . 11
  - 5.1.2 pydist2.custom_typing module . . . 11
  - 5.1.3 pydist2.distance module . . . 11
  - 5.1.4 Module contents . . . 11

6 Contributing **13**
- 6.1 Types of Contributions . . . 13
  - 6.1.1 Report Bugs . . . 13
  - 6.1.2 Fix Bugs . . . 13
  - 6.1.3 Implement Features . . . 13
  - 6.1.4 Write Documentation . . . 13
  - 6.1.5 Submit Feedback . . . 14
- 6.2 Get Started! . . . 14
- 6.3 Pull Request Guidelines . . . 15
- 6.4 Tips . . . 15
- 6.5 Deploying . . . 15

7 Credits **17**
- 7.1 Development Lead . . . 17
- 7.2 Contributors . . . 17

8 History **19**

8.1 0.0.1 (2021-03-04) 19
8.2 0.0.5 (2021-03-21) 19

9 Indices and tables **21**

Python Module Index **23**

Index **25**

CHAPTER

ONE

PYDIST2

pydist2 is a python library that provides a set of methods for calculating distances between observations. There are two main classes:

- **pdist1** which calculates the pairwise distances between observations in one matrix and returns a distance matrix.
- **pdist2** computes the distances between observations in two matrices and also returns a distance matrix.

1.1 Usage

```
pdist1(P, metric = "euclidean", matrix=False)
pdist2(P, Q, metric = "minkowski", exp = 3)
```

Arguments:

- two matrices P and Q.
- metric: The distance function to use.
- exp: The exponent of the Minkowski distance.

1.2 Installation

The pydist2 library is available on Pypi. Thus, you can install the latest available version using *pip*:

```
pip install pydist2
```

1.3 Supported Python versions

pydist2 has been tested with Python 3.7 and 3.8.

For more information, please checkout the documentation which is available at readthedocs.

This program and the accompanying materials are made available under the terms of the MIT License.

1.4 Progress & Features

- Commit the first code's version.
- Support the following list of distances.
- Display the distance in a matrix form(a combination for each pair of points):

```
>>> X = np.array([[100, 100],[0, 100],[100, 0], [500, 400], [300, 600]])
>>> pdist1(X,matrix=True) # by default, metric = 'euclidean'
array([[100.    , 100.    , 100.    ,   0.    , 100.    ],
       [100.    , 100.    , 100.    , 100.    ,   0.    ],
       [500.    , 100.    , 100.    , 500.    , 400.    ],
       [538.5165, 100.    , 100.    , 300.    , 600.    ],
       [141.4214,   0.    , 100.    , 100.    ,   0.    ],
       [583.0952,   0.    , 100.    , 500.    , 400.    ],
       [583.0952,   0.    , 100.    , 300.    , 600.    ],
       [565.6854, 100.    ,   0.    , 500.    , 400.    ],
       [632.4555, 100.    ,   0.    , 300.    , 600.    ],
       [282.8427, 500.    , 400.    , 300.    , 600.    ]])
```

where the first column represents the distance between each pair of observations. for instance, the euclidean distance between (100. , 100.) and (0. , 100.) is 100.

- Support numpy arrays of the same size only.

1.5 Todo list

- Re-validate the correctness of the distances equations.
- Performance tests & vectorization.
- Adding new distances.
- Adding a squared form of the distance.
- Support tuples and list.
- Remove numpy from dependencies.
- Write more test cases.
- Handling Exceptions.
- Restructure the docs.

CHAPTER

TWO

INSTALLATION GUIDE

2.1 Stable release

To install pydist2, run this command in your terminal:

```
$ pip install pydist2
```

This is the preferred method to install pydist2, as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

2.2 From sources

The sources for pydist2 can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/wiseaidev/pydist2
```

Or download the tarball:

```
$ curl -OJL https://github.com/wiseaidev/pydist2/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python -m pip install .
```

CHAPTER THREE

PYDIST2 TUTORIAL

This module contains two main interfaces which provides the following functionalities:

- **pdist1** calculates the pairwise distances between points in one vector and returns a vector which represents the distance between the observations.
- **pdist2** calculates the distances between points in two vectors and returns a vector which represents the distance between the observations.

The beginner will enjoy how the `distance` module lets you get started quickly.

```
>>> import numpy as np
>>> from pydist2.distance import Euclidean
>>> x = np.array([[1, 2, 3],
       [7, 8, 9],
       [5, 6, 7],], dtype=np.float32)
>>> Euclidean.compute(x)
array([10.39230485,  6.92820323,  3.46410162])
```

However, a better programmer can use the `pdist1` or `pdist2` class to compute the actual pairwise vectors distances *object* upon which he can then perform lots of operations. For example, consider this Python program:

```
import numpy as np
from pydist2.distance import pdist1
x = np.array([[1, 2, 3],
    [7, 8, 9],
    [5, 6, 7],], dtype=np.float32)
euclidean_distance = pdist1(x,'euclidean')
print(f"The euclidean distance between each observation in \n{x}\n is:\n{euclidean_
↪distance}")
```

this program will produce the following output:

```
The euclidean distance between each observation in
[[1. 2. 3.]
 [7. 8. 9.]
 [5. 6. 7.]]
 is:
[10.39230485  6.92820323  3.46410162]
```

Read *pydist2 guide* to learn more!

Warning: This module only handles numpy arrays as inputs; any other data types are right out(e.g. list, tuple…); this will be solved in future develpment of the package.

CHAPTER

FOUR

PYDIST2 GUIDE

Whether you need to compute the distances between observation of vectors, `distance` does it all!

4.1 Special Distances

There are several kinds of distance for which `distance` offers special support.

A. **pdist1** calculates the pairwise distances between observations in one vector with one of the following methods:

For a given a matrix of points $X_{m,n}$ (m rows and n columns), where each row is treated as a vector of points X_{1i} = $(x_{1i}, x_{2i}, \ldots, x_{ni})$, the various distances between the vector X_i and X_j are defined as follows:

1. **euclidean**: $d_{X_j,X_k} = \sqrt{\sum_{i=1}^{n}(x_{ij} - x_{ik})^2}$
2. **default**: the default method is the euclidean distance.
3. **seuclidean**: standardized euclidean distance. $d_{X_j,X_k} = \sqrt{\sum_{i=1}^{n} w_i(x_{ij} - x_{ik})^2}$ where w represents the inverse of the variance of the vector Xj over the m vectors.
4. **cityblock**: $d_{X_j,X_k} = \sum_{i=1}^{n} |x_{ij} - x_{ik}|$
5. **mahalanobis**: $d_{X_j,X_k} = \sum_{i=1}^{n} V_i(x_{ij} - x_{ik})^2$ where V is the inverse of the covariance matrix of X.
6. **minkowski**: $d_{X_j,X_k} = (\sum_{i=1}^{n} |x_{ij} - x_{ik})^p|^{1/p}$
7. **chebyshev**: $d_{X_j,X_k} = max|X_j - X_k|$
8. **cosine**: $d_{X_j,X_k} = 1 - cos(\boldsymbol{X}_j, \boldsymbol{X}_k) = 1 - \frac{\boldsymbol{X}_j \cdot \boldsymbol{X}_k}{||\boldsymbol{X}_j|| \cdot ||\boldsymbol{X}_k||} = 1 - \frac{\sum_{i=1}^{n} \boldsymbol{x}_{ij} \cdot \boldsymbol{x}_{ik}}{\sqrt{\sum_{i=1}^{n} x_{ij}^2} \cdot \sqrt{\sum_{i=1}^{n} x_{ik}^2}}$
9. **correlation**: $d_{X_j,X_k} = 1 - \frac{\sum_{i=1}^{n}(x_{ij} - (1/n) \cdot \sum_{j=1}^{n} x_{ij}) \cdot (x_{ik} - (1/n) \cdot \sum_{k=1}^{n} x_{ik})}{\sqrt{\sum_{i=1}^{n}(x_{ij} - (1/n) \cdot \sum_{j=1}^{n} x_{ij})^2} \cdot \sqrt{\sum_{i=1}^{n}(x_{ik} - (1/n) \cdot \sum_{k=1}^{n} x_{ik})^2}}$
10. **spearman**: $d_{X_j,X_k} = 1 - \frac{\sum_{i=1}^{n}(r_{ij} - \frac{(n+1)}{2}) \cdot (r_{ik} - \frac{(n+1)}{2})}{\sqrt{\sum_{i=1}^{n}(r_{ij} - \frac{(n+1)}{2})^2} \cdot \sqrt{\sum_{i=1}^{n}(r_{ik} - \frac{(n+1)}{2})^2}}$
11. **hamming**: $d_{X_j,X_k} = \frac{\|(X_j \otimes X_k) \cap mask_{X_j} \cap mask_{X_k}\|}{\|mask_{X_j} \cap mask_{X_k}\|}$
12. **jaccard**: $d_{X_j,X_k} = \frac{|X_j \bigcap X_k|}{|X_j \bigcup X_k|}$

B. **pdist2** calculates the distances between observations in two vectors with one of the following methods:

1. **manhattan**: The L1 distance between two vectors P and Q is defined as: $d(P, Q) = ||P - Q||_1 = \sum_{i=1}^{n} |p_i - q_i|$
2. **sqeuclidean**: Euclidean squared distance defined as: $d(P, Q)^2 = \sum_{i=1}^{n}(p_i - q_i)^2$
3. **euclidean**: Euclidean distance defined as: $d(P, Q) = \sqrt{\sum_{i=1}^{n}(P - Q)^2}$
4. **default**: the default method is the euclidean distance.

5. **chi-squared**: $d_{P,Q} = \sum_{i=1}^{n} \frac{(P_i - Q_i)^2}{P_i + Q_i}$
6. **cosine**: $1 - Cosine\_Similarity = 1 - cos(\boldsymbol{P}, \boldsymbol{Q}) = 1 - \frac{\boldsymbol{P} \cdot \boldsymbol{Q}}{||\boldsymbol{P}|| \cdot ||\boldsymbol{Q}||} = 1 - \frac{\sum_{i=1}^{n} \boldsymbol{P}_i \cdot \boldsymbol{Q}_i}{\sqrt{\sum_{i=1}^{n} P_i^2} \cdot \sqrt{\sum_{i=1}^{n} Q_i^2}}$
7. **earthmover**: $EMD(P, Q) = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij}}{\sum_{i=1}^{m} \sum_{j=1}^{b} f_{ij}}$

These are supported by simple classes that are available in the `distance` module, and also by a pair of classes of the main `pdist1` and `pdist2` classes.

4.2 pdist1 and pdist2

The library can compute distances between pair of observations in one vector using `pdist1`, and distances between pair of observations in two vectors using `pdist2`. Note that the two vectors must have the same shape!

```
>>> from pydist2.distance import pdist1, pdist2
>>> import numpy as np
>>> x = np.array([[1, 2, 3],
...        [7, 8, 9],
...        [5, 6, 7],], dtype=np.float32)
>>> y = np.array([[10, 20, 30],
...        [70, 80, 90],
...        [50, 60, 70]], dtype=np.float32)
>>> a = pdist1(x)
>>> a
array([10.39230485,  6.92820323,  3.46410162])
>>> pdist1(x, 'seuclidean')
array([3.40168018, 2.26778677, 1.13389339])
>>> pdist1(x, 'minkowski', exp=3)
array([8.65349742, 5.76899828, 2.88449914])
>>> pdist1(x, 'minkowski', exp=2)
array([10.39230485,  6.92820323,  3.46410162])
>>> pdist1(x, 'minkowski', exp=1)
array([18., 12.,  6.])
>>> pdist1(x, 'cityblock')
array([18., 12.,  6.])
>>> pdist2(x, y)
array([[ 33.67491648, 135.69819453, 101.26203632],
       [ 24.37211521, 125.35549449,  90.96153033],
       [ 27.38612788, 128.80217389,  94.39279634]])
>>> pdist2(x, y, 'manhattan')
array([[ 54., 234., 174.],
       [ 36., 216., 156.],
       [ 42., 222., 162.]])
>>> pdist2(x, y, 'sqeuclidean')
array([[ 1134., 18414., 10254.],
       [  594., 15714.,  8274.],
       [  750., 16590.,  8910.]])
>>> pdist2(x, y, 'chi-squared')
array([[ 22.09090909, 111.31927838,  81.41482329],
       [  8.48998061,  88.36363636,  59.6522841 ],
       [ 11.75121275,  95.51418525,  66.27272727]])
>>> pdist2(x, y, 'cosine')
```

(continues on next page)

(continued from previous page)

```
array([[-5.60424152e-09,  4.05881305e-02,  3.16703408e-02],
       [ 4.05880431e-02,  7.31070616e-08,  5.62480978e-04],
       [ 3.16703143e-02,  5.62544701e-04, -1.23279462e-08]])
>>> pdist2(x, y, 'earthmover')
array([[ 90., 450., 330.],
       [ 54., 414., 294.],
       [ 66., 426., 306.]])
```

CHAPTER

FIVE

PYDIST2

5.1 pydist2 package

5.1.1 Submodules

5.1.2 pydist2.custom_typing module

5.1.3 pydist2.distance module

5.1.4 Module contents

Top-level package for pydist2.

CHAPTER

SIX

CONTRIBUTING

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

6.1 Types of Contributions

6.1.1 Report Bugs

Report bugs at https://github.com/wiseaidev/pydist2/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

6.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” and “help wanted” is open to whoever wants to implement it.

6.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “enhancement” and “help wanted” is open to whoever wants to implement it.

6.1.4 Write Documentation

pydist2 could always use more documentation, whether as part of the official pydist2 docs, in docstrings, or even on the web in blog posts, articles, and such.

6.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/wiseaidev/pydist2/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

6.2 Get Started!

Ready to contribute? Here's how to set up *pydist2* for local development.

1. Fork the *pydist2* repo on GitHub.
2. Clone your fork locally:

   ```
   $ git clone git@github.com:your_name_here/pydist2.git
   ```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

   ```
   $ mkvirtualenv pydist2
   $ cd pydist2/
   $ python -m pip install -e ".[dev]"
   ```

4. Create a branch for local development:

   ```
   $ git checkout -b name-of-your-bugfix-or-feature
   ```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

   ```
   $ flake8 pydist2 tests
   $ python -m pip install -e ".[test]"
   $ tox
   ```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

   ```
   $ git add .
   $ git commit -m "Your detailed description of your changes."
   $ git push origin name-of-your-bugfix-or-feature
   ```

7. Submit a pull request through the GitHub website.

6.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 3.8, and for PyPy. Check https://app.travis-ci.com/github/wiseaidev/pydist2/pull_requests and make sure that the tests pass for all supported Python versions.

6.4 Tips

To run a subset of tests:

```
$ pytest tests.test_pydist2
```

6.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ bump2version patch # possible: major / minor / patch
$ git push
$ git push --tags
```

Travis will then deploy to PyPI if tests pass.

CHAPTER

SEVEN

CREDITS

7.1 Development Lead

- Mahmoud Harmouch <mahmoudddharmouchhh@gmail.com>

7.2 Contributors

None yet. Why not be the first?

CHAPTER

EIGHT

HISTORY

8.1 0.0.1 (2021-03-04)

- First release on PyPI.

8.2 0.0.5 (2021-03-21)

- Adding sqeuclidean metric.

CHAPTER

NINE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

p
pydist2, 11

INDEX

M

module
    pydist2, 11

P

pydist2
    module, 11