pyms-nist-search

Release 0.6.3.post1

PyMassSpec extension for searching mass spectra using NIST's Mass Spectrum Search Engine.

Dominic Davis-Foster

Apr 29, 2024

Contents

PyMassSpec extension for searching mass spectra using NIST's Spectrum Search Engine

PyMassSpec NIST Search is Free Software licensed under the GNU Lesser General Public License Version 3.

A copy of the MassBank of North America database, in JSON, MSP and NIST Library formats, is included for the purposes of these tests. This library was created on 22 April 2020 using the "parse_mona_json.py" script and Lib2Nist. Licensed under the CC BY 4.0 License. For a list of contributors, see the file `MoNA_GCMS_Library/AUTHORS`.

Chapter

ONE

Installation

1.1 from PyPI

```
$ python3 -m pip install pyms-nist-search --user
```

1.2 from GitHub

```
$ python3 -m pip install git+https://github.com/domdfcoding/pynist@master --user
```

Chapter

TWO

Usage

You will need to supply your own copy of the NIST Mass Spectral library to use this software.

The main class in this library is the *Engine* class. This class performs the actual searching. Start by initialising the search engine as follows:

```
search = pyms_nist_search.Engine(
        FULL_PATH_TO_MAIN_LIBRARY,
        pyms_nist_search.NISTMS_MAIN_LIB,
        FULL_PATH_TO_WORK_DIR,
        )
```

Where `FULL_PATH_TO_MAIN_LIBRARY` is the path to the location of your mass spectral library, and `FULL_PATH_TO_WORK_DIR` is the path to the working directory to be used by the search engine.

A `pyms.Spectrum.MassSpectrum` object can then be searched as follows:

```
search.full_search_with_ref_data(mass_spec)
```

This will return a list of tuples consisting of *SearchResult* and *ReferenceData* objects for the possible identities of the mass spectrum.

A list of just the *SearchResult* objects can be obtained with this method:

```
hit_list = search.full_search(mass_spec)
```

For each of these hits, the reference data can be obtained as follows:

```
for hit in hit_list:
    ref_data = search.get_reference_data(hit.spec_loc)
```

Chapter

THREE

API Reference

PyMassSpec extension for searching mass spectra using NIST's Mass Spectrum Search Engine.

3.1 base

Base class for other PyMassSpec NIST Search classes.

class NISTBase(*name=''*, *cas='---'*)

Bases: object

Base class for other PyMassSpec NIST Search classes.

Parameters

- **name** (str) – The name of the compound. Default ''.
- **cas** (Union[str, int]) – The CAS number of the compound. Default '---'.

Methods:

__eq__(other)	Return self == other.
__str__()	Return str(self).
from_dict(dictionary)	Construct an object from a dictionary.
from_json(json_data)	Construct an object from json data.
from_pynist(pynist_dict)	Create an object from the raw data returned by the C extension.
to_dict()	Convert the object to a dictionary.
to_json()	Convert the object to json.

Attributes:

cas	The CAS number of the compound.
name	The name of the compound.

__eq__(*other*)

Return self == other.

Return type bool

__str__()

Return str(self).

Return type str

property cas
The CAS number of the compound.

Return type str

classmethod from_dict(*dictionary*)
Construct an object from a dictionary.

Parameters **dictionary** (Dict[str, Any])

classmethod from_json(*json_data*)
Construct an object from json data.

Parameters **json_data** (str)

classmethod from_pynist(*pynist_dict*)
Create an object from the raw data returned by the C extension.

Parameters **pynist_dict** (Dict[str, Any])

property name
The name of the compound.

Return type str

to_dict()
Convert the object to a dictionary.

New in version 0.6.0.

Return type Dict[str, Any]

to_json()
Convert the object to json.

Return type str

3.2 docker_engine

Search engine for Linux and other platforms supporting Docker.

Classes:

Engine(lib_path[, lib_type, work_dir, debug])	Search engine for Linux and other platforms supporting Docker.

Functions:

hit_list_from_json(json_data)	Parse json data into a list of SearchResult objects.
hit_list_with_ref_data_from_json(json_data)	Parse json data into a list of (SearchResult, ReferenceData) tuples.
require_init(func)	Decorator to ensure that functions do not run after the class has been uninitialised.

class Engine(*lib_path*, *lib_type=1*, *work_dir=None*, *debug=False*)

Bases: object

Search engine for Linux and other platforms supporting Docker.

The first time the engine is initialized it will download the latest version of the docker image automatically. This can also be performed manually, such as to upgrade to the latest version, with the following command:

```
$ docker pull domdfcoding/pywine-pyms-nist
```

The engine must be uninitialized when no longer required to shut down the underlying docker container. This is achieved with the *uninit()* method. Alternatively, this class can be used as a contextmanager to automatically uninitialize the engine upon leaving the with block:

```
with pyms_nist_search.Engine(
        FULL_PATH_TO_MAIN_LIBRARY,
        pyms_nist_search.NISTMS_MAIN_LIB,
        FULL_PATH_TO_WORK_DIR,
        ) as search:
    search.full_spectrum_search(ms, n_hits=5)
```

Changed in version 0.6.0: Added context manager support.

Methods:

full_search_with_ref_data(mass_spec[, n_hits])	Perform a Full Spectrum Search of the mass spectral library, including reference data.
full_spectrum_search(mass_spec[, n_hits])	Perform a Full Spectrum Search of the mass spectral library.
get_reference_data(spec_loc)	Get reference data from the library for the compound at the given location.
spectrum_search(mass_spec[, n_hits])	Perform a Quick Spectrum Search of the mass spectral library.
uninit()	Uninitialize the Search Engine.

full_search_with_ref_data(*mass_spec*, *n_hits=5*)
Perform a Full Spectrum Search of the mass spectral library, including reference data.

Parameters

- **mass_spec** (MassSpectrum) – The mass spectrum to search against the library.
- **n_hits** (int) – The number of hits to return. Default 5.

Return type List[Tuple[*SearchResult*, *ReferenceData*]]

Returns List of tuples containing possible identities for the mass spectrum, and the reference data.

full_spectrum_search(*mass_spec*, *n_hits=5*)
Perform a Full Spectrum Search of the mass spectral library.

Parameters

- **mass_spec** (MassSpectrum) – The mass spectrum to search against the library.
- **n_hits** (int) – The number of hits to return. Default 5.

Return type List[*SearchResult*]

Returns List of possible identities for the mass spectrum.

get_reference_data(*spec_loc*)
Get reference data from the library for the compound at the given location.

Parameters **spec_loc** (int)

Return type *ReferenceData*

spectrum_search(*mass_spec*, *n_hits=5*)
Perform a Quick Spectrum Search of the mass spectral library.

Parameters

- **mass_spec** (MassSpectrum) – The mass spectrum to search against the library.
- **n_hits** (int) – The number of hits to return. Default 5.

Return type List[*SearchResult*]

Returns List of possible identities for the mass spectrum.

uninit()
Uninitialize the Search Engine.

hit_list_from_json(*json_data*)
Parse json data into a list of SearchResult objects.

Parameters **json_data** (str) – str

Return type List[*SearchResult*]

hit_list_with_ref_data_from_json(*json_data*)
Parse json data into a list of (SearchResult, ReferenceData) tuples.

Parameters **json_data** (str) – str

Return type List[Tuple[*SearchResult*, *ReferenceData*]]

require_init(*func*)
Decorator to ensure that functions do not run after the class has been uninitialised.

Parameters **func** (Callable) – The function or method to wrap.

Return type Callable

3.3 reference_data

Class to store reference data from NIST MS Search.

class ReferenceData(*name=''*, *cas='---'*, *nist_no=0*, *id=''*, *mw=0.0*, *formula=''*, *contributor=''*, *mass_spec=None*, *synonyms=None*, *exact_mass=None*)
Bases: *NISTBase*

Class to store reference data from NIST MS Search.

Parameters

- **name** (str) – The name of the compound. Default ''.
- **cas** (Union[str, int]) – The CAS number of the compound. Default '---'.
- **nist_no** (Union[int, str]) – Default 0.
- **id** (Union[str, int]) – Default ''.
- **mw** (Union[float, str]) – Default 0.0.
- **formula** (str) – The formula of the compound. Default ''.
- **contributor** (str) – The contributor to the library. Default ''.
- **mass_spec** (Optional[MassSpectrum]) – The reference mass spectrum. Default None.
- **synonyms** (Optional[Sequence[str]]) – List of synonyms for the compound. Default None.

Methods:

__repr__()	Return a string representation of the *ReferenceData*.
from_jcamp(file_name[, ignore_warnings])	Create a ReferenceData object from a JCAMP-DX file.
from_json(json_data)	Construct an object from JSON data.
from_mona_dict(mona_data)	Construct an object from Massbank of North America json data that has been loaded into a dictionary.
from_pynist(pynist_dict)	Create a *ReferenceData* object from the raw data returned by the C extension.
to_dict()	Convert the object to a dictionary.
to_json()	Convert the object to JSON.
to_msp()	Returns the ReferenceData object as an MSP file similar to that produced by NIST MS Search's export function.

Attributes:

contributor	The name of the contributor to the library.
exact_mass	The exact mass of the compound.
formula	The formula of the compound.
id	The ID of the compound.
mass_spec	The mass spectrum of the compound.
mw	The molecular weight of the compound.
nist_no	The NIST number of the compund.
synonyms	A list of synonyms for the compound.

__repr__()
Return a string representation of the *ReferenceData*.

Return type str

property contributor
The name of the contributor to the library.

Return type str

property exact_mass
The exact mass of the compound.

Return type float

property formula
The formula of the compound.

Return type str

classmethod from_jcamp(*file_name*, *ignore_warnings=True*)
Create a ReferenceData object from a JCAMP-DX file.

Parameters

- **file_name** (Union[str, Path, PathLike]) – Path of the file to read.
- **ignore_warnings** (bool) – Whether warnings about invalid tags should be shown. Default True.

Authors Qiao Wang, Andrew Isaac, Vladimir Likic, David Kainer, Dominic Davis-Foster

Return type *ReferenceData*

classmethod from_json(*json_data*)
Construct an object from JSON data.

Parameters **json_data** (str)

Return type *ReferenceData*

classmethod from_mona_dict(*mona_data*)
Construct an object from Massbank of North America json data that has been loaded into a dictionary.

Parameters **mona_data** (Dict) – dict

Return type *ReferenceData*

classmethod from_pynist(*pynist_dict*)
Create a *ReferenceData* object from the raw data returned by the C extension.

Parameters **pynist_dict** (Dict[str, Any])

Return type *ReferenceData*

property id
The ID of the compound.

Return type str

property mass_spec
The mass spectrum of the compound.

Return type Optional[MassSpectrum]

property mw
The molecular weight of the compound.

Return type int

property nist_no
The NIST number of the compund.

Return type int

property synonyms
A list of synonyms for the compound.

Return type List[str]

to_dict()
Convert the object to a dictionary.

New in version 0.6.0.

Return type Dict[str, Any]

to_json()
Convert the object to JSON.

Return type str

to_msp()
Returns the ReferenceData object as an MSP file similar to that produced by NIST MS Search's export function.

Return type str

3.4 search_result

Class to store search results from NIST MS Search.

class SearchResult(*name=''*, *cas='---'*, *match_factor=0*, *reverse_match_factor=0*, *hit_prob=0.0*, *spec_loc=0*)

Bases: *NISTBase*

Class to store search results from NIST MS Search.

Parameters

- **name** (str) – The name of the compound. Default ''.
- **cas** (Union[str, int]) – The CAS number of the compound. Default '---'.
- **match_factor** (float) – Default 0.
- **reverse_match_factor** (float) – Default 0.
- **hit_prob** (float) – Default 0.0.
- **spec_loc** (float) – The location of the reference spectrum in the library. Default 0.

Methods:

from_pynist(pynist_dict)	Create a *SearchResult* object from the raw data returned by the C extension.
to_dict()	Convert the object to a dictionary.

Attributes:

hit_prob	Returns the probability of the hit being the compound responsible for the mass spectrum.
match_factor	Returns a score (out of 1000) representing the similarity of the searched mass spectrum to the search result.
reverse_match_factor	A score (out of 1000) representing the similarity of the searched mass spectrum to the search result, but ignoring any peaks that are in the searched mass spectrum but not in the library spectrum.
spec_loc	The location of the reference spectrum in the library.

classmethod from_pynist(*pynist_dict*)

Create a *SearchResult* object from the raw data returned by the C extension.

Parameters **pynist_dict** (Dict[str, Any])

Return type *SearchResult*

property hit_prob

Returns the probability of the hit being the compound responsible for the mass spectrum.

Return type float

property match_factor
Returns a score (out of 1000) representing the similarity of the searched mass spectrum to the search result.

Return type int

property reverse_match_factor
A score (out of 1000) representing the similarity of the searched mass spectrum to the search result, but ignoring any peaks that are in the searched mass spectrum but not in the library spectrum.

Return type int

property spec_loc
The location of the reference spectrum in the library.

This can then be searched using the `get_reference_data()` method of the search engine to obtain the reference data.

Return type int

to_dict()
Convert the object to a dictionary.

New in version 0.6.0.

Return type Dict[str, Any]

3.5 utils

General utilities.

Functions:

pack(mass_spec[, top])	Convert a pyms.Spectrum.MassSpectrum object into a string.
parse_name_chars(name_char_list)	Takes a list of Unicode character codes and converts them to characters, taking into account the special codes used by the NIST DLL.

pack(*mass_spec*, *top=20*)
Convert a pyms.Spectrum.MassSpectrum object into a string.

Adapted from https://sourceforge.net/projects/mzapi-live/

Parameters

- **mass_spec** (MassSpectrum)
- **top** (int) – The number of largest peaks to identify. Default 20.

Return type str

parse_name_chars(*name_char_list*)
Takes a list of Unicode character codes and converts them to characters, taking into account the special codes used by the NIST DLL.

Parameters **name_char_list** (Sequence[int])

Return type str

Returns The parsed name.

3.6 win_engine

Search engine for Windows systems.

class Engine(*lib_path*, *lib_type=1*, *work_dir=None*, *debug=False*)
Bases: object

Search engine for Windows systems.

Parameters

- **lib_path** (Union[str, Path, PathLike]) – The path to the mass spectral library.
- **lib_type** (int) – The type of library. One of NISTMS_MAIN_LIB, NISTMS_USER_LIB, NISTMS_REP_LIB. Default 1.
- **work_dir** (Union[str, Path, PathLike, None]) – The path to the working directory. Default None.

Methods:

full_search_with_ref_data(mass_spec[, n_hits])	Perform a Full Spectrum Search of the mass spectral library, including reference data.
full_spectrum_search(mass_spec[, n_hits])	Perform a Full Spectrum Search of the mass spectral library.
get_reference_data(spec_loc)	Get reference data from the library for the compound at the given location.
spectrum_search(mass_spec[, n_hits])	Perform a Quick Spectrum Search of the mass spectral library.
uninit()	Uninitialize the Search Engine.

full_search_with_ref_data(*mass_spec*, *n_hits=5*)
Perform a Full Spectrum Search of the mass spectral library, including reference data.

Parameters

- **mass_spec** (MassSpectrum) – The mass spectrum to search against the library.
- **n_hits** (int) – The number of hits to return. Default 5.

Return type List[Tuple[*SearchResult*, *ReferenceData*]]

Returns List of tuples containing possible identities for the mass spectrum, and the reference data

static full_spectrum_search(*mass_spec*, *n_hits=5*)
Perform a Full Spectrum Search of the mass spectral library.

Parameters

- **mass_spec** (MassSpectrum) – The mass spectrum to search against the library.
- **n_hits** (int) – The number of hits to return. Default 5.

Return type List[*SearchResult*]

Returns List of possible identities for the mass spectrum.

static get_reference_data(*spec_loc*)
Get reference data from the library for the compound at the given location.

Parameters **spec_loc** (int)

Return type *ReferenceData*

static spectrum_search(*mass_spec*, *n_hits=5*)
Perform a Quick Spectrum Search of the mass spectral library.

Parameters

- **mass_spec** (MassSpectrum) – The mass spectrum to search against the library.
- **n_hits** (int) – The number of hits to return. Default 5.

Return type List[*SearchResult*]

Returns List of possible identities for the mass spectrum.

uninit()
Uninitialize the Search Engine.

Chapter

FOUR

Contributing

`pyms-nist-search` uses tox to automate testing and packaging, and pre-commit to maintain code quality.

Install `pre-commit` with `pip` and install the git hook:

```
$ python -m pip install pre-commit
$ pre-commit install
```

4.1 Coding style

formate is used for code formatting.

It can be run manually via `pre-commit`:

```
$ pre-commit run formate -a
```

Or, to run the complete autoformatting suite:

```
$ pre-commit run -a
```

4.2 Automated tests

Tests are run with `tox` and `pytest`. To run tests for a specific Python version, such as Python 3.6:

```
$ tox -e py36
```

To run tests for all Python versions, simply run:

```
$ tox
```

4.3 Type Annotations

Type annotations are checked using `mypy`. Run `mypy` using `tox`:

```
$ tox -e mypy
```

4.4 Build documentation locally

The documentation is powered by Sphinx. A local copy of the documentation can be built with `tox`:

```
$ tox -e docs
```

Chapter

FIVE

License

`pyms-nist-search` is licensed under the GNU Lesser General Public License v3.0

Permissions of this copyleft license are conditioned on making available complete source code of licensed works and modifications under the same license or the GNU GPLv3. Copyright and license notices must be preserved. Contributors provide an express grant of patent rights. However, a larger work using the licensed work through interfaces provided by the licensed work may be distributed under different terms and without source code for the larger work.

Permissions

- Commercial use – The licensed material and derivatives may be used for commercial purposes.
- Modification – The licensed material may be modified.
- Distribution – The licensed material may be distributed.
- Patent use – This license provides an express grant of patent rights from contributors.
- Private use – The licensed material may be used and modified in private.

Conditions

- License and copyright notice – A copy of the license and copyright notice must be included with the licensed material.
- Disclose source – Source code must be made available when the licensed material is distributed.
- State changes – Changes made to the licensed material must be documented.
- Same license (library) – Modifications must be released under the same license when distributing the licensed material. In some cases a similar or related license may be used, or this condition may not apply to works that use the licensed material as a library.

Limitations

- Liability – This license includes a limitation of liability.
- Warranty – This license explicitly states that it does NOT provide any warranty.

See more information on choosealicense.com ⇒

```
                   GNU LESSER GENERAL PUBLIC LICENSE
                       Version 3, 29 June 2007

Copyright (C) 2007 Free Software Foundation, Inc. <https://fsf.org/>
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
```

(continues on next page)

(continued from previous page)

```
  This version of the GNU Lesser General Public License incorporates
the terms and conditions of version 3 of the GNU General Public
License, supplemented by the additional permissions listed below.

  0. Additional Definitions.

  As used herein, "this License" refers to version 3 of the GNU Lesser
General Public License, and the "GNU GPL" refers to version 3 of the GNU
General Public License.

  "The Library" refers to a covered work governed by this License,
other than an Application or a Combined Work as defined below.

  An "Application" is any work that makes use of an interface provided
by the Library, but which is not otherwise based on the Library.
Defining a subclass of a class defined by the Library is deemed a mode
of using an interface provided by the Library.

  A "Combined Work" is a work produced by combining or linking an
Application with the Library.  The particular version of the Library
with which the Combined Work was made is also called the "Linked
Version".

  The "Minimal Corresponding Source" for a Combined Work means the
Corresponding Source for the Combined Work, excluding any source code
for portions of the Combined Work that, considered in isolation, are
based on the Application, and not on the Linked Version.

  The "Corresponding Application Code" for a Combined Work means the
object code and/or source code for the Application, including any data
and utility programs needed for reproducing the Combined Work from the
Application, but excluding the System Libraries of the Combined Work.

  1. Exception to Section 3 of the GNU GPL.

  You may convey a covered work under sections 3 and 4 of this License
without being bound by section 3 of the GNU GPL.

  2. Conveying Modified Versions.

  If you modify a copy of the Library, and, in your modifications, a
facility refers to a function or data to be supplied by an Application
that uses the facility (other than as an argument passed when the
facility is invoked), then you may convey a copy of the modified
version:

   a) under this License, provided that you make a good faith effort to
   ensure that, in the event an Application does not supply the
   function or data, the facility still operates, and performs
   whatever part of its purpose remains meaningful, or

   b) under the GNU GPL, with none of the additional permissions of
   this License applicable to that copy.

  3. Object Code Incorporating Material from Library Header Files.
```

(continues on next page)

(continued from previous page)

```
  The object code form of an Application may incorporate material from
a header file that is part of the Library.  You may convey such object
code under terms of your choice, provided that, if the incorporated
material is not limited to numerical parameters, data structure
layouts and accessors, or small macros, inline functions and templates
(ten or fewer lines in length), you do both of the following:

   a) Give prominent notice with each copy of the object code that the
   Library is used in it and that the Library and its use are
   covered by this License.

   b) Accompany the object code with a copy of the GNU GPL and this license
   document.

  4. Combined Works.

  You may convey a Combined Work under terms of your choice that,
taken together, effectively do not restrict modification of the
portions of the Library contained in the Combined Work and reverse
engineering for debugging such modifications, if you also do each of
the following:

   a) Give prominent notice with each copy of the Combined Work that
   the Library is used in it and that the Library and its use are
   covered by this License.

   b) Accompany the Combined Work with a copy of the GNU GPL and this license
   document.

   c) For a Combined Work that displays copyright notices during
   execution, include the copyright notice for the Library among
   these notices, as well as a reference directing the user to the
   copies of the GNU GPL and this license document.

   d) Do one of the following:

       0) Convey the Minimal Corresponding Source under the terms of this
       License, and the Corresponding Application Code in a form
       suitable for, and under terms that permit, the user to
       recombine or relink the Application with a modified version of
       the Linked Version to produce a modified Combined Work, in the
       manner specified by section 6 of the GNU GPL for conveying
       Corresponding Source.

       1) Use a suitable shared library mechanism for linking with the
       Library.  A suitable mechanism is one that (a) uses at run time
       a copy of the Library already present on the user's computer
       system, and (b) will operate properly with a modified version
       of the Library that is interface-compatible with the Linked
       Version.

   e) Provide Installation Information, but only if you would otherwise
   be required to provide such information under section 6 of the
   GNU GPL, and only to the extent that such information is
   necessary to install and execute a modified version of the
   Combined Work produced by recombining or relinking the
```

(continues on next page)

(continued from previous page)

```
   Application with a modified version of the Linked Version. (If
   you use option 4d0, the Installation Information must accompany
   the Minimal Corresponding Source and Corresponding Application
   Code. If you use option 4d1, you must provide the Installation
   Information in the manner specified by section 6 of the GNU GPL
   for conveying Corresponding Source.)

  5. Combined Libraries.

  You may place library facilities that are a work based on the
Library side by side in a single library together with other library
facilities that are not Applications and are not covered by this
License, and convey such a combined library under terms of your
choice, if you do both of the following:

   a) Accompany the combined library with a copy of the same work based
   on the Library, uncombined with any other library facilities,
   conveyed under the terms of this License.

   b) Give prominent notice with the combined library that part of it
   is a work based on the Library, and explaining where to find the
   accompanying uncombined form of the same work.

  6. Revised Versions of the GNU Lesser General Public License.

  The Free Software Foundation may publish revised and/or new versions
of the GNU Lesser General Public License from time to time. Such new
versions will be similar in spirit to the present version, but may
differ in detail to address new problems or concerns.

  Each version is given a distinguishing version number. If the
Library as you received it specifies that a certain numbered version
of the GNU Lesser General Public License "or any later version"
applies to it, you have the option of following the terms and
conditions either of that published version or of any later version
published by the Free Software Foundation. If the Library as you
received it does not specify a version number of the GNU Lesser
General Public License, you may choose any version of the GNU Lesser
General Public License ever published by the Free Software Foundation.

  If the Library as you received it specifies that a proxy can decide
whether future versions of the GNU Lesser General Public License shall
apply, that proxy's public statement of acceptance of any version is
permanent authorization for you to choose that version for the
Library.
```

Chapter

SIX

Downloading source code

The `pyms-nist-search` source code is available on GitHub, and can be accessed from the following URL: https://github.com/domdfcoding/pynist

If you have `git` installed, you can clone the repository with the following command:

```
$ git clone https://github.com/domdfcoding/pynist
```

```
Cloning into 'pynist'...
remote: Enumerating objects: 47, done.
remote: Counting objects: 100% (47/47), done.
remote: Compressing objects: 100% (41/41), done.
remote: Total 173 (delta 16), reused 17 (delta 6), pack-reused 126
Receiving objects: 100% (173/173), 126.56 KiB | 678.00 KiB/s, done.
Resolving deltas: 100% (66/66), done.
```

Alternatively, the code can be downloaded in a 'zip' file by clicking:
Clone or download –> Download Zip

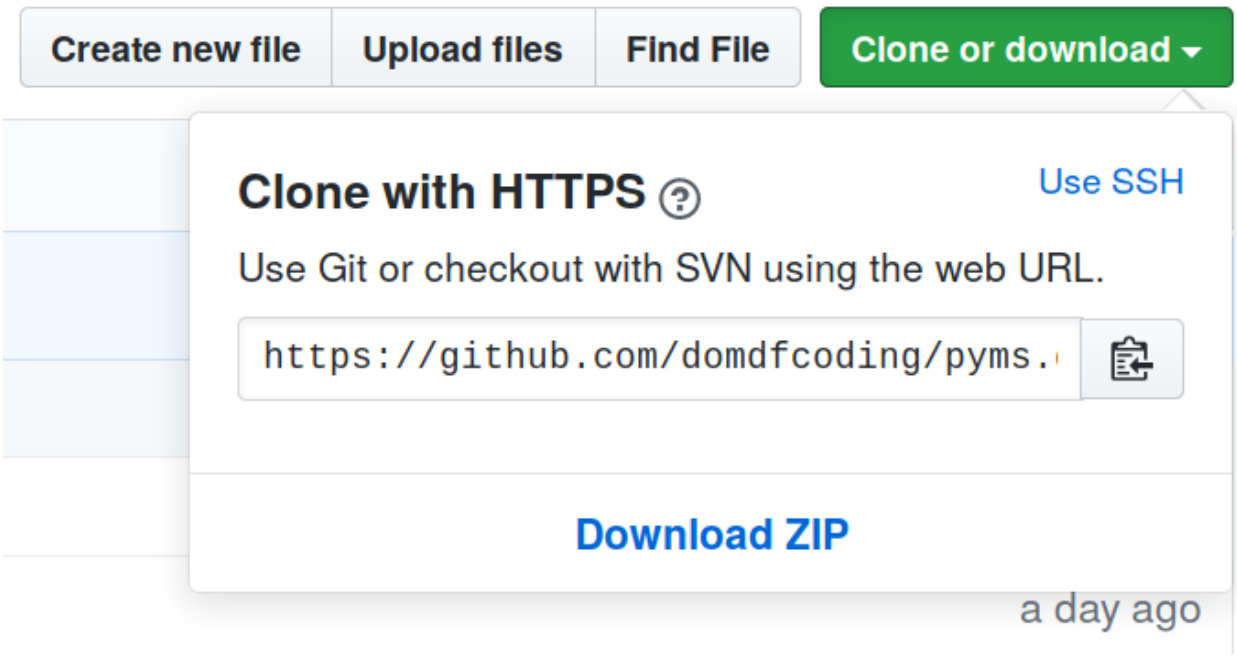

Fig. 1: Downloading a 'zip' file of the source code

6.1 Building from source

The recommended way to build `pyms-nist-search` is to use tox:

```
$ tox -e build
```

The source and wheel distributions will be in the directory `dist`.

If you wish, you may also use pep517.build or another **PEP 517**-compatible build tool.

Python Module Index

p

Index

Symbols

C

E

F

G

H

I

M

N

P

R

S

T

U