nominally

Release 1.1.0

Matt VanEseltine

May 19, 2021

CONTENTS

Nominally simplifies and parses a personal name written in Western name order into six core fields: title, first, middle, last, suffix, and nickname.

Typically, *nominally* is used to parse entire lists or pd.Series of names en masse. This package includes a command line tool to parse a single name for convenient one-off testing and examples.

CHAPTER

ONE

FOR RECORD LINKAGE

Nominally is designed to assist at the front end of record linkage, during data preprocessing.

Varying quality and practices across institutions and datasets introduce noise into data and cause misrepresentation. This increases the challenges of deduplicating rows within data and and linking names across multiple datasets. We observe this by-no-means-exhaustive list:

- First and middle names split arbitrarily.
- Misplaced prefixes of last names (e.g., “van” and “de la”).
- Records with multiple last names partitioned into middle name fields.
- Titles and suffixes various recorded in fields and/or with separators.
- Inconsistent capture of accents and other non-ASCII characters.
- Single name fields concatenating name parts arbitrarily.

In attempting to match someone named Ramsay Jackson Canning across data, one may uncover

- R.J. CANNING JUNIOR
- Canning, Ramsay J.
- Ramsay “R.J.” Jackson Canning
- Dr. Ramsay Jackson Canning, M.D.
- Ramsay J. Canning, Jr.
- canning, jr., dr. ramsay

—and so on.

Nominally can’t fix *all* of your data problems (sorry).

But it can help by **consistently extracting the most useful features of personal names** under the highly restrictive case of a single string name field. *Nominally* *aggressively cleans*, scrapes titles, nicknames, and suffixes, and parses apart first, middle, and last names. In the list above (and many, many variations beyond), *nominally* correctly captures each Canning as a last name, each R(amsay) as a first, both types of suffix, and so forth.

CHAPTER TWO

CONTENTS

2.1 Use

2.1.1 Installation

Install in the normal way:

```
$ pip install nominally
```

Working on a project within a virtual environment is highly recommended:

```
$ python3 -m venv .venv
$ source ./.venv/bin/activate
(.venv) $ pip install nominally
Collecting nominally
  Downloading [...]/nominally-1.0.3-py3-none-any.whl
Collecting unidecode>=1.0 (from nominally)
  Downloading [...]/Unidecode-1.1.1-py2.py3-none-any.whl
Installing collected packages: unidecode, nominally
Successfully installed nominally-1.0.3 unidecode-1.1.1
```

Nominally requires Python 3.6 or higher and has one external dependency (unidecode).

2.1.2 parse_name() function

The *nominally.api.parse_name()* function returns the five core fields:

```
>>> from pprint import pprint
>>> import nominally
>>> parsed = nominally.parse_name('Samuel "Young Sam" Vimes II')
>>> pprint(parsed)
{'first': 'samuel',
 'last': 'vimes',
 'middle': '',
 'nickname': 'young sam',
 'suffix': 'ii',
 'title': ''}
```

2.1.3 Name() class

Additional features are exposed via the *nominally.parser.Name* class:

```
>>> from pprint import pprint
>>> from nominally import Name
>>> n = Name('Delphine Angua von Uberwald')
>>> pprint(n.report())
{'cleaned': {'delphine angua von uberwald'},
 'first': 'delphine',
 'last': 'von uberwald',
 'list': ['', 'delphine', 'angua', 'von uberwald', '', ''],
 'middle': 'angua',
 'nickname': '',
 'parsed': 'von uberwald, delphine angua',
 'raw': 'Delphine Angua von Uberwald',
 'suffix': '',
 'title': ''}
>>> n.raw
'Delphine Angua von Uberwald'
>>> n.cleaned
{'delphine angua von uberwald'}
>>> n.first
'delphine'
>>> n['first']
'delphine'
>>> n.get('first')
'delphine'
>>> pprint(dict(n))
{'first': 'delphine',
 'last': 'von uberwald',
 'middle': 'angua',
 'nickname': '',
 'suffix': '',
 'title': ''}
```

2.1.4 From the Console

For convenience, single names can be run at the command line.

```
$ nominally "St John Nobbs, Cecil (Nobby) Wormsborough"
       raw: St John Nobbs, Cecil (Nobby) Wormsborough
   cleaned: {'st john nobbs, cecil wormsborough', 'nobby'}
    parsed: st john nobbs, cecil (nobby) wormsborough
      list: ['', 'cecil', 'wormsborough', 'st john nobbs', '', 'nobby']
     title:
     first: cecil
    middle: wormsborough
      last: st john nobbs
    suffix:
  nickname: nobby
```

2.1.5 Extended Examples

See https://github.com/vaneseltine/nominally-examples/ for detailed examples of nominally usage.

2.2 FAQ

2.2.1 Input format

Nominally does one thing: take a name and parse it.

The name must be received as a Unicode string. If you are working with bytes as input, you will first need to decode them.

For assistance in working with Unicode strings, see:

- Python 3 Documentation, “Unicode HOWTO”
- Ned Batchelder, “Pragmatic Unicode”
- Joel Spolsky, “The Absolute Minimum Every Software Developer Absolutely, Positively Must Know About Unicode and Character Sets (No Excuses!)”

Nominally takes input one name at a time. For ideas about how to use Nominally on a larger scale, to work with entire lists, DataFrames, files, or databases, see https://github.com/vaneseltine/nominally-examples/.

2.2.2 Name cleaning

Nominally does not create or tag canonical names.

Strings are *aggressively* cleaned.

For specifics, see `nominally.parser.Name.clean()`

2.2.3 Name ordering

We only handle Western name order. No effort is made to disentangle or rearrange names based on their origins.

We do not preserve suffix or title ordering. Treat these as sets.

2.2.4 Titles and suffixes

We handle few suffixes:

- PhD
- MD
- Sr
- Junior, Jr, II, 2nd, III, 3rd, IV, 4th

We handle very few titles:

- Dr.
- Mr.
- Mrs.
- Ms.

These are treated as unordered sets.

2.2.5 Library

The Name class creates immutable instances.

2.2.6 See Also

More great Python packages in the record linkage community include:

- Python Record Linkage Toolkit by Jonathan de Bruin
- Dedupe Python Library by Forest Gregg and Derek Eder
- RLTK: Record Linkage ToolKit by the USC Center on Knowledge Graphs

2.3 Reference

Herein find documentation for the gory details of *nominally*.

2.3.1 API

`nominally.api.`**`parse_name`**`(`*s: str*`)` → Dict[str, Any]

Parse into Name, return core name attributes as a dict.

This is the simplest function interface to *nominally*.

`nominally.api.`**`cli`**`(`*arguments: Optional[Sequence[str]] = None*`)` → int

Simple CLI with a minimal set of options.

1. Report of a single name (parse into details).
2. Help via usage information. [help, -h, –help]
3. Version information. [-V, –version]

`nominally.api.`**`cli_help`**`()` → int

Output help for command line usage

`nominally.api.`**`cli_report`**`(`*raw_name: str*, *details: bool = True*`)` → int

Parse into Name, output (core or report) attributes.

`nominally.api.`**`cli_version`**`()` → int

Output version info and script location

2.3.2 Name

`class` `nominally.parser.`**`Name`**`(`*raw: str = ”*`)`

A personal name, separated and simplified into component parts.

`detail`

`classmethod clean``(`*s: str*, *, *condense: bool = False*, *final: bool = False*`)` → str

Clean this string to the simplest possible representation (but no simpler).

Note: Assumes that any nicknames have already been removed, along with anything else that would depend on special characters (other than commas).

static strip_pointlessness(*s: str*) → str

__eq__(*other: Any*) → bool
If Name is parsable and object dicts are identical, consider it equal.

__str__() → str
Output format: “last, title first middle suffix (nickname)”

- “organs, mr harry x, jr (snapper)”
- “organs, mr harry x, jr”
- “organs, mr harry x”
- “organs, harry x”
- “organs, harry”
- etc.

property parsable
Return true if any valid name values were created.

property raw
Return the original input string.

property cleaned
Return some set of cleaned string parts.

first

last

middle

nickname

report() → Dict[str, Any]
Return a more-or-less complete parsing dict.

suffix

title

2.4 About

Nominally is a program to separate commonly-used parts of personal names. Copyright (C) 2021 Matt VanEseltine.

2.4.1 GNU Affero General Public License

Nominally is free software: you can redistribute it and/or modify it under the terms of the GNU Affero General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

Nominally is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Affero General Public License for more details.

A copy of the GNU Affero General Public License is distributed along with **nominally** source code as the text file LICENSE. It is also available at the GNU Project and `here with this documentation`.

Logo

The nominally logo is based on a licensed copy of Head by Andrei Yushchenko at The Noun Project.

nameparser

Nominally began in mid-2019 as a fork of the Name Parser package (v. 1.0.4, ce92f37). Name Parser is copyright (C) 2014-2019 Derek Gulbranson and licensed herein under the GNU Lesser General Public License, version 2.1.

CHAPTER

THREE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX