scrubadub Documentation

Release 0.1.0

Dean Malmgren

April 15, 2015

Contents

1 Quick start **3**

2 Related work **5**

2.1 Advanced usage 5

2.2 Contributing 6

2.3 Change Log 7

3 Indices and tables **9**

Warning: This package is a work in progress and is not yet available on pypi. This documentation should be considered more of a design document for what scrubadub will do someday rather than a specification of what it can do today.

Remove personally identifiable information from free text. Sometimes we have additional metadata about the people we wish to anonymize. Other times we don't. This package makes it easy to seamlessly scrub personal information from free text, without comprimising the privacy of the people we are trying to protect.

`scrubadub` currently supports removing:

- Names (proper nouns) via textblob
- Email addresses
- URLs
- Phone numbers via phonenumbers
- username / password combinations
- Skype usernames

CHAPTER 1

Quick start

Getting started with scrubadub is as easy as pip install scrubadub and incorporating it into your python scripts like this:

```
>>> import scrubadub

# John may be a cat, but he doesn't want other people to know it.
>>> text = u"John is a cat"

# Replace names with {{NAME}} placeholder. This is the scrubadub default
# because it maximally omits any information about people.
>>> placeholder_text = scrubadub.clean_with_placeholders(text)
>>> placeholder_text
u"{{NAME}} is a cat"
```

As a python package, scrubadub also has several more *advanced features* to allow users to fine-tune the manner in which scrubadub cleans dirty dirty text.

CHAPTER 2

Related work

scrubadub isn't the first package to attempt to remove personally identifiable information from free text. There are a handful of other projects out there that have very similar aims and which provide some inspiration for how scrubadub should work.

- MITRE gives the ability to replace names with a placeholder like [NAME] or alternatively replace names with fake names. last release in 8/2014. not on github. unclear what language although it looks like python. it is clear that the documentation sucks and is primarily intended for academic audiences (docs are in papers).
- physionet has a few deidentification packages that look pretty decent but are both written in perl and require advance knowledge of what you are trying to replace. Intended for HIPAA regulations. In particular, deid has some good lists of names that might be useful in spite of the fact it has 5k+ lines of gross perl.

Contents:

2.1 Advanced usage

By default, scrubadub aggressively removes content from text that may reveal personal identity, but there are certainly circumstances where such omissions are (i) not necessary and (ii) detrimental to downstream analysis. scrubadub allows users to fine-tune the manner in which content is deidentified using the specific methods in the scrubadub.scrubbers.Scrubber class.

class `scrubadub.scrubbers.`**`Scrubber`**

The Scrubber class is used to clean personal information out of dirty dirty text.

`clean_with_placeholders`(*text*)

This is the master method that cleans all of the filth out of the dirty dirty `text` using the default options for all of the other `clean_*` methods below.

`clean_proper_nouns`(*text*, *replacement='{{NAME}}'*)

Use part of speech tagging to clean proper nouns out of the dirty dirty `text`.

`clean_email_addresses`(*text*, *replacement='{{EMAIL}}'*)

Use regular expression magic to remove email addresses from dirty dirty `text`. This method also catches email addresses like `john at gmail.com`.

`clean_urls`(*text*, *replacement='{{URL}}'*, *keep_domain=False*)

Use regular expressions to remove URLs that begin with `http://`, `https://` or `www.` from dirty dirty `text`.

With `keep_domain=True`, this method only obfuscates the path on a URL, not its domain. For example, `http://twitter.com/someone/status/234978haoin` becomes `http://twitter.com/{{replacement}}`.

clean_phone_numbers(*text*, *replacement='{{PHONE}}'*, *region='US'*)
Remove phone numbers from dirty dirty `text` using python-phonenumbers, a port of a Google project to correctly format phone numbers in text.

`region` specifies the best guess region to start with (default: `"US"`). Specify `None` to only consider numbers with a leading + to be considered.

clean_credentials(*text*, *username_replacement='{{USERNAME}}'*, *password_replacement='{{PASSWORD}}'*)
Remove username/password combinations from dirty drity `text`.

clean_skype(*text*, *replacement='{{SKYPE}}'*, *word_radius=10*)
Skype usernames tend to be used inline in dirty dirty text quite often but also appear as `skype: {{SKYPE}}` quite a bit. This method looks at words within `word_radius` words of "skype" for things that appear to be misspelled or have punctuation in them as a means to identify skype usernames.

Default `word_radius` is 10, corresponding with the rough scale of half of a sentence before or after the word "skype" is used. Increasing the `word_radius` will increase the false positive rate and decreasing the `word_radius` will increase the false negative rate.

2.2 Contributing

The overarching goal of this project is to remove personally identifiable information from raw text as reliably as possible. In practice, this means that this project, by default, will preferentially be overly conservative in removing information that might be personally identifiable. As this project matures, I fully expect the project to become ever smarter about how it interprets and anonymizes raw text.

Regardless of which peraonl information is identified, this project is committed to being as agnostic about the manner in which the text is anonymized, so long as it is done with rigor and does not inadvertantly lead to improper anonymization. Replacing with placholders? Replacing with anonymous (but consistent) IDs? Replacing with random metadata? Other ideas? All should be supported to make this project as useful as possible to the people that need it.

Another important aspect of this project is that we want to have extremely good documentation and source code that is easy to read. If you notice a type-o, error, confusing statement etc, please fix it!

2.2.1 Quick start

1. Fork and clone the project:

```
git clone https://github.com/YOUR-USERNAME/scrubadub.git
```

2. Create a python virtual environment and install the requirements

```
mkvirtualenv scrubadub
pip install -r requirements/python-dev
```

3. Contribute! There are several open issues that provide good places to dig in. Check out the contribution guidelines and send pull requests; your help is greatly appreciated!

4. Run the test suite that is defined in `.travis.yml` to make sure everything is working properly

```
./tests/run.py
```

Current build status:

2.3 Change Log

This project uses semantic versioning to track version numbers, where backwards incompatible changes (highlighted in **bold**) bump the major version of the package.

2.3.1 latest changes in development for next release

2.3.2 0.1.0

- added skype username scrubbing (#10)
- added username/password scrubbing (#4)
- added phone number scrubbing (#3)
- added URL scrubbing, including URL path removal (#2)
- make sure unicode is passed to `scrubadub` (#1)
- several bug fixes, including:
  - accuracy issues with things like "I can be reached at 312.456.8453" (#8)
  - accuracy issues with usernames that are email addresses (#9)

2.3.3 0.0.1

- initial release, ported from past projects

CHAPTER 3

Indices and tables

- *genindex*
- *modindex*
- *search*

C

clean_credentials() (scrubadub.scrubbers.Scrubber method), 6
clean_email_addresses() (scrubadub.scrubbers.Scrubber method), 5
clean_phone_numbers() (scrubadub.scrubbers.Scrubber method), 5
clean_proper_nouns() (scrubadub.scrubbers.Scrubber method), 5
clean_skype() (scrubadub.scrubbers.Scrubber method), 6
clean_urls() (scrubadub.scrubbers.Scrubber method), 5
clean_with_placeholders() (scrubadub.scrubbers.Scrubber method), 5

S

Scrubber (class in scrubadub.scrubbers), 5