swachhdata Documentation

Release 1.0.0

Kritik Seth

Nov 30, 2020

INDEX:

- Simple and efficient tools for cleaning and transforming data
- Accessible to everybody, and reusable in various contexts
- Open source, commercially usable - MPL-2.0 license

CHAPTER ONE

ABOUT

- Simple and efficient tools for cleaning and transforming data
- Accessible to everybody, and reusable in various contexts
- Open source, commercially usable - MPL-2.0 license

1.1 Author

- Kritik Seth

CHAPTER

TWO

INSTALL

2.1 Conda

If you use *conda*, you can install it with:

```
conda install swachhdata
```

2.2 Pip

If you use *pip*, you can install it with:

```
pip install swachhdata
```

2.3 Dependencies

- regex >= 2019.12.20
- pandas >= 1.1.4
- tqdm >= 4.41.1
- bs4 >= 0.0.1
- beautifulsoup4 >= 4.6.3
- html5lib >= 1.0.1
- contractions >= 0.0.25
- emoji >= 0.6.0
- nltk >= 3.2.5
- spacy >= 2.2.4
- gensim >= 3.6.0
- num2words >= 0.5.10
- textblob >= 0.15.3

CHAPTER THREE

USER GUIDE

3.1 Quantitative

Coming Soon…

3.2 Qualitative

Coming Soon…

3.3 Text

3.3.1 urlRecast

Recast text data by removing or extracting URLs.

URLs supported:

- HTTP address: http://www.website.com
- HTTPS address: https://www.website.com
- www.website.com
- website.com
- www.website.gov.in/website.html
- IPv4 address: http://192.168.1.1/website.jpg
- Address with different Port: www.website.com:8080/website.jpg
- IPv4: 192.168.1.1/website.jpg
- Ipv6: 2001:0db8:0000:85a3:0000:0000:ac1f:8001/website.jpg
- Other permutations and combinations of above URLs.

process: string (‘remove’, ‘extract’, ‘extract_remove’), default=’remove’ verbose: int (0, 1, -1), default=0

get_regex_ [string] regex being used by recast

url_ [list of string(s)] extracted url(s)

```
>>> # process='remove'
>>> from swachhdata.text import urlRecast
>>> text = 'You can have a look at our catalogue at www.samplewebsite.com in␣
↪the services tab'
>>> url = urlRecast(process='remove')
>>> url.setup(text)
>>> url.recast()
'You can have a look at our catalogue at in the services tab'
>>> # OR
>>> url.setup_recast(text)
'You can have a look at our catalogue at in the services tab'
>>>
>>> # process='extract'
>>> from swachhdata.text import urlRecast
>>> text = 'You can have a look at our catalogue at www.samplewebsite.com in␣
↪the services tab'
>>> url = urlRecast(process='extract')
>>> url.setup(text)
>>> url.recast()
['www.samplewebsite.com']
>>> # OR
>>> url.setup_recast(text)
['www.samplewebsite.com']
>>>
>>> # process='extract_remove'
>>> from swachhdata.text import urlRecast
>>> text = 'You can have a look at our catalogue at www.samplewebsite.com in␣
↪the services tab'
>>> url = urlRecast(process='extract_remove')
>>> url.setup(text)
>>> url.recast()
'You can have a look at our catalogue at in the services tab'
['www.samplewebsite.com']
>>> # OR
>>> url.setup_recast(text)
'You can have a look at our catalogue at in the services tab'
['www.samplewebsite.com']
>>>
```

3.3.2 htmlRecast

Recast text data by removing HTML tags.

uses lxml from BeautifulSoup to clean up html tags

verbose: int (0, 1, -1), default=0

```
>>> from swachhdata.text import htmlRecast
>>> text = '<a href="www.samplewebsite.com">Click Here</a> to have a look at␣
↪the menu in the services tab'
>>> rec = htmlRecast()
>>> rec.setup(text)
>>> rec.recast()
'Click Here to have a look at the menu in the services tab'
>>> # OR
>>> rec.setup_recast(text)
'Click Here to have a look at the menu in the services tab'
```

3.3.3 EscapeSequenceRecast

Recast text data by removing Escape Sequences.

verbose: int (0, 1, -1), default=0

```
>>> from swachhdata.text import EscapeSequenceRecast
>>> text = 'To have a look at the menu\nClick Here'
>>> rec = EscapeSequenceRecast()
>>> rec.setup(text)
>>> rec.recast()
'To have a look at the menu Click Here'
>>> # OR
>>> rec.setup_recast(text)
'To have a look at the menu Click Here'
```

3.3.4 MentionRecast

Recast text data by removing or extracting Mentions.

Mentions supported:

- @jon_doe
- @123jon_doe
- @jon_doe123
- @jondoe
- @jon.doe
- @jon:doe
- @jon-doe

process: string (‘remove’, ‘extract’, ‘extract_remove’), default=’remove’ verbose: int (0, 1, -1), default=0

get_regex_ [string] regex being used by recast

mention_ [list of string(s)] extracted mention(s)

```
>>> # process='remove'
>>> from swachhdata.text import MentionRecast
>>> text = 'If you like the service we offer, post a review on google and␣
↪tag us @jondoe'
>>> rec = MentionRecast(process='remove')
>>> rec.setup(text)
>>> rec.recast()
'If you like the service we offer, post a review on google and tag us'
>>> # OR
>>> rec.setup_recast(text)
'If you like the service we offer, post a review on google and tag us'
>>>
>>> # process='extract'
>>> from swachhdata.text import MentionRecast
>>> text = 'If you like the service we offer, post a review on google and␣
↪tag us @jondoe'
>>> rec = MentionRecast(process='extract')
>>> rec.setup(text)
```

(continues on next page)

(continued from previous page)

```
>>> rec.recast()
['@jondoe']
>>> # OR
>>> rec.setup_recast(text)
['@jondoe']
>>>
>>> # process='extract_remove'
>>> from swachhdata.text import MentionRecast
>>> text = 'If you like the service we offer, post a review on google and tag us @jondoe'
>>> rec = MentionRecast(process='extract_remove')
>>> rec.setup(text)
>>> rec.recast()
'If you like the service we offer, post a review on google and tag us'
['@jondoe']
>>> # OR
>>> rec.setup_recast(text)
'If you like the service we offer, post a review on google and tag us'
['@jondoe']
```

3.3.5 ContractionsRecast

Recast text data by expanding Contractions

verbose: int (0, 1, -1), default=0

```
>>> # process='remove'
>>> from swachhdata.text import ContractionsRecast
>>> text = 'They're going to wildlife sanctuary, I guess Jon's going to be there too.'
>>> rec = ContractionsRecast()
>>> rec.setup(text)
>>> rec.recast()
'They are going to wildlife sanctuary, I guess Jon is going to be there too.'
>>> # OR
>>> rec.setup_recast(text)
'They are going to wildlife sanctuary, I guess Jon is going to be there too.'
```

3.3.6 CaseRecast

Recast text data by case formatting the text

Case formats supported:

- UPPER case (upper)
- lower case (lower)
- First Upper case (fupper)

process: str (‘lower’, ‘upper’, ‘fupper’), default=’lower’ verbose: int (0, 1, -1), default=0

```
>>> # process='lower'
>>> from swachhdata.text import CaseRecast
>>> text = 'You can have a look at our catalogue in the services tab'
```

(continues on next page)

(continued from previous page)

```
>>> rec = CaseRecast(process='lower')
>>> rec.setup(text)
>>> rec.recast()
'you can have a look at our catalogue in the services tab'
>>> # OR
>>> rec.setup_recast(text)
'you can have a look at our catalogue in the services tab'
>>>
>>> # process='upper'
>>> from swachhdata.text import CaseRecast
>>> text = 'You can have a look at our catalogue in the services tab'
>>> rec = CaseRecast(process='upper')
>>> rec.setup(text)
>>> rec.recast()
'YOU CAN HAVE A LOOK AT OUR CATALOGUE IN THE SERVICES TAB'
>>> # OR
>>> rec.setup_recast(text)
'YOU CAN HAVE A LOOK AT OUR CATALOGUE IN THE SERVICES TAB'
>>>
>>> # process='fupper'
>>> from swachhdata.text import CaseRecast
>>> text = 'You can have a look at our catalogue in the services tab'
>>> rec = CaseRecast(process='fupper')
>>> rec.setup(text)
>>> rec.recast()
'You Can Have A Look At Our Catalogue In The Services Tab'
>>> # OR
>>> rec.setup_recast(text)
'You Can Have A Look At Our Catalogue In The Services Tab'
```

3.3.7 EmojiRecast

Recast text data by removing, replaing or extracting Emoji(s).

process: string ('remove', 'replace', 'extract', 'extract_remove', 'extract_replace'), default='remove' space_out = bool (True, False), default=False verbose: int (0, 1, -1), default=0

emoji_ [list of emoji(s)] extracted emoji(s)

```
>>> # process='remove'
>>> from swachhdata.text import EmojiRecast
>>> text = 'Thanks a lot for your wishes! '
>>> rec = EmojiRecast(process='remove')
>>> rec.setup(text)
>>> rec.recast()
'Thanks a lot for your wishes!'
>>> # OR
>>> rec.setup_recast(text)
'Thanks a lot for your wishes!'
>>>
>>> # process='replace'
>>> from swachhdata.text import EmojiRecast
>>> text = 'Thanks a lot for your wishes! '
>>> rec = EmojiRecast(process='replace')
>>> rec.setup(text)
>>> rec.recast()
```

(continues on next page)

(continued from previous page)

```
'Thanks a lot for your wishes! smiling_face_with_smiling_eyes '
>>> # OR
>>> rec.setup_recast(text)
'Thanks a lot for your wishes! smiling_face_with_smiling_eyes '
>>>
>>> # process='extract'
>>> from swachhdata.text import EmojiRecast
>>> text = 'Thanks a lot for your wishes! '
>>> rec = EmojiRecast(process='extract')
>>> rec.setup(text)
>>> rec.recast()
['']
>>> # OR
>>> rec.setup_recast(text)
['']
>>> # process='extract_remove'
>>> from swachhdata.text import EmojiRecast
>>> text = 'Thanks a lot for your wishes! '
>>> rec = EmojiRecast(process='extract_remove')
>>> rec.setup(text)
>>> rec.recast()
'Thanks a lot for your wishes!'
['']
>>> # OR
>>> rec.setup_recast(text)
'Thanks a lot for your wishes!'
['']
>>> # process='extract_replace'
>>> from swachhdata.text import EmojiRecast
>>> text = 'Thanks a lot for your wishes! '
>>> rec = EmojiRecast(process='extract_replace')
>>> rec.setup(text)
>>> rec.recast()
'Thanks a lot for your wishes! smiling_face_with_smiling_eyes'
['']
>>> # OR
>>> rec.setup_recast(text)
'Thanks a lot for your wishes! smiling_face_with_smiling_eyes'
['']
```

3.3.8 HashtagRecast

Recast text data by removing or extracting Hashtag(s).

Hashtags supported:

- #sample_website
- #sample_website123
- #123sample_website
- #sample_website

process: string (‘remove’, ‘extract’, ‘extract_remove’), default=’remove’ verbose: int (0, 1, -1), default=0

get_regex_ [string] regex being used by recast

hashtag_ [list of string(s)] extracted hashtag(s)

```
>>> # process='remove'
>>> from swachhdata.text import HashtagRecast
>>> text = 'Post a photo with tag #samplephoto to win prizes'
>>> rec = HashtagRecast(process='remove')
>>> rec.setup(text)
>>> rec.recast()
'Post a photo with tag to win prizes'
>>> # OR
>>> rec.setup_recast(text)
'Post a photo with tag to win prizes'
>>>
>>> # process='extract'
>>> from swachhdata.text import HashtagRecast
>>> text = 'Post a photo with tag #samplephoto to win prizes'
>>> rec = HashtagRecast(process='extract')
>>> rec.setup(text)
>>> rec.recast()
['#samplephoto']
>>> # OR
>>> rec.setup_recast(text)
['#samplephoto']
>>>
>>> # process='extract_remove'
>>> from swachhdata.text import HashtagRecast
>>> text = 'Post a photo with tag #samplephoto to win prizes'
>>> rec = HashtagRecast(process='extract_remove')
>>> rec.setup(text)
>>> rec.recast()
'Post a photo with tag to win prizes'
['#samplephoto']
>>> # OR
>>> rec.setup_recast(text)
'Post a photo with tag to win prizes'
['#samplephoto']
```

3.3.9 ShortWordsRecast

Recast text data by removing (short) words of specified length.

min_length int (>0), default=3 verbose: int (0, 1, -1), default=0

```
>>> # min_length=3
>>> from swachhdata.text import ShortWordsRecast
>>> text = 'You can have a look at our catalogue in the services tab'
>>> rec = ShortWordsRecast(min_length=3)
>>> rec.setup(text)
>>> rec.recast()
'have look catalogue services'
>>> # OR
>>> rec.setup_recast(text)
'have look catalogue services'
```

3.3.10 StopWordsRecast

Recast text data by removing stop words.

> package : str ('nltk', 'spacy', 'gensim', 'custom'), default='nltk' stopwords : list (package='custom'), list of stopwords verbose : int (0, 1, -1), default=0

```
>>> from swachhdata.text import StopWordsRecast
>>> text = 'You can have a look at our catalogue in the services tab'
>>> rec = StopWordsRecast(package='nltk')
>>> rec.setup(text)
>>> rec.recast()
'You look catalogue services tab'
>>> # OR
>>> rec.setup_recast(text)
'You look catalogue services tab'
```

3.3.11 NumberRecast

Recast text data by removing, replacing or extracting numbers.

> Number formats supported: * 1234567 * 1,234,567 (use seperator=',') * 12,34,567 (use seperator=',') * 123.4567 (if not decimal, use seperator='.')
>
> process: string ('remove', 'replace', 'extract', 'extract_remove', 'extract_replace'), default='remove' seperator = str (',', '.'), default=None verbose: int (0, 1, -1), default=0
>
> **number_** [list of number(s)] extracted number(s)

```
>>> # process='remove'
>>> from swachhdata.text import NumberRecast
>>> text = 'The sales turnover of quarter 1 this year was $ 123456'
>>> rec = NumberRecast(process='remove')
>>> rec.setup(text)
>>> rec.recast()
'The sales turnover of quarter  this year was $ '
>>> # OR
>>> rec.setup_recast(text)
'The sales turnover of quarter  this year was $ '
>>>
>>> # process='replace'
>>> from swachhdata.text import NumberRecast
>>> text = 'The sales turnover of quarter 1 this year was $ 123456'
>>> rec = NumberRecast(process='replace')
>>> rec.setup(text)
>>> rec.recast()
'The sales turnover of quarter one this year was $ one hundred and twenty-
↪three thousand, four hundred and fifty-six'
>>> # OR
>>> rec.setup_recast(text)
'The sales turnover of quarter one this year was $ one hundred and twenty-
↪three thousand, four hundred and fifty-six'
>>>
>>> # process='extract'
>>> from swachhdata.text import NumberRecast
>>> text = 'The sales turnover of quarter 1 this year was $ 123456'
>>> rec = NumberRecast(process='extract')
```

(continues on next page)

(continued from previous page)

```
>>> rec.setup(text)
>>> rec.recast()
['1', '123456']
>>> # OR
>>> rec.setup_recast(text)
['1', '123456']
>>> # process='extract_remove'
>>> from swachhdata.text import NumberRecast
>>> text = 'The sales turnover of quarter 1 this year was $ 123456'
>>> rec = NumberRecast(process='extract_remove')
>>> rec.setup(text)
>>> rec.recast()
'The sales turnover of quarter  this year was $ '
['1', '123456']
>>> # OR
>>> rec.setup_recast(text)
'The sales turnover of quarter  this year was $ '
['1', '123456']
>>> # process='extract_replace'
>>> from swachhdata.text import NumberRecast
>>> text = 'The sales turnover of quarter 1 this year was $ 123456'
>>> rec = NumberRecast(process='extract_replace')
>>> rec.setup(text)
>>> rec.recast()
'The sales turnover of quarter one this year was $ one hundred and twenty-
↪three thousand, four hundred and fifty-six'
['1', '123456']
>>> # OR
>>> rec.setup_recast(text)
'The sales turnover of quarter one this year was $ one hundred and twenty-
↪three thousand, four hundred and fifty-six'
['1', '123456']
```

3.3.12 AlphabetRecast

Recast text data by removing all accented, non ascii characters and keeping only alphabets.

> process: string / list ('all', 'keep_alpha', 'rem_non_ascii', 'rem_acc_char', or combination in a list), default='all' verbose: int (0, 1, -1), default=0

```
>>> # process='all' (default)
>>> from swachhdata.text import AlphabetRecast
>>> text = 'It was past lunch time so the 3 of us dropped by The Main Street␣
↪Café  for a late lunch '
>>> rec = AlphabetRecast()
>>> rec.setup(text)
>>> rec.recast()
'It was past lunch time so the   of us dropped by The Main Street Cafe  for␣
↪a late lunch '
>>> # OR
>>> rec.setup_recast(text
'It was past lunch time so the   of us dropped by The Main Street Cafe  for␣
↪a late lunch '
```

3.3.13 PunctuationRecast

Recast text data by removing punctuations.

verbose: int (0, 1, -1), default=0

```
>>> from swachhdata.text import PunctuationRecast
>>> text = 'Have you fed that dog? I told you, "Don't feed that dog!"'
>>> rec = PunctuationRecast()
>>> rec.setup(text)
>>> rec.recast()
'Have you fed that dog I told you Don t feed that dog'
>>> # OR
>>> rec.setup_recast(text)
'Have you fed that dog I told you Don t feed that dog'
```

3.3.14 TokenisationRecast

Recast text data by tokenising it.

Tokenisation supported:

- word tokenisation
- sentence tokenisation

package: string (‘nltk’, ‘spacy’), default=’nltk’ method: string (‘word’, ‘sentence’), default=None verbose: int (0, 1, -1), default=0

```
>>> # method='word'
>>> from swachhdata.text import TokenisationRecast
>>> text = 'Grabbing her umbrella, Kate raced out of the house. Confused by␣
↪her sister’s sudden change in mood, Jill stayed quiet.'
>>> rec = TokenisationRecast(package='nltk', method='word')
>>> rec.setup(text)
>>> rec.recast()
['Grabbing', 'her', 'umbrella', ',', 'Kate', 'raced', 'out', 'of', 'the',
↪'house', '.', 'Confused', 'by', 'her', 'sister', '’', 's', 'sudden',
↪'change', 'in', 'mood', ',', 'Jill', 'stayed', 'quiet', '.']
>>> # OR
>>> rec.setup_recast(text)
['Grabbing', 'her', 'umbrella', ',', 'Kate', 'raced', 'out', 'of', 'the',
↪'house', '.', 'Confused', 'by', 'her', 'sister', '’', 's', 'sudden',
↪'change', 'in', 'mood', ',', 'Jill', 'stayed', 'quiet', '.']
>>>
>>> # method='sentence'
>>> from swachhdata.text import TokenisationRecast
>>> text = 'You can have a look at our catalogue at www.samplewebsite.com in␣
↪the services tab'
>>> rec = TokenisationRecast(package='nltk', method='sentence')
>>> rec.setup(text)
>>> rec.recast()
['Grabbing her umbrella, Kate raced out of the house.', 'Confused by her␣
↪sister’s sudden change in mood, Jill stayed quiet.']
>>> # OR
>>> rec.setup_recast(text)
['Grabbing her umbrella, Kate raced out of the house.', 'Confused by her␣
↪sister’s sudden change in mood, Jill stayed quiet.']
```

3.3.15 StemmingRecast

Recast text data by performing stemming on it.

> package: string (‘nltk’, ‘extract’, ‘extract_remove’), default=’nltk’ method: string (‘porter’, ‘snowball’) verbose: int (0, 1, -1), default=0

```
>>> # method='porter'
>>> from swachhdata.text import StemmingRecast
>>> text = 'You can have a look at our catalogue at www.samplewebsite.com in␣
↪the services tab'
>>> rec = StemmingRecast(method='porter')
>>> rec.setup(text)
>>> rec.recast()
'you can have a look at our catalogu at www.samplewebsite.com in the servic␣
↪tab'
>>> # OR
>>> rec.setup_recast(text)
'you can have a look at our catalogu at www.samplewebsite.com in the servic␣
↪tab'
>>>
>>> # method='snowball'
>>> from swachhdata.text import StemmingRecast
>>> text = 'You can have a look at our catalogue at www.samplewebsite.com in␣
↪the services tab'
>>> rec = StemmingRecast(method='snowball')
>>> rec.setup(text)
>>> rec.recast()
'you can have a look at our catalogu at www.samplewebsite.com in the servic␣
↪tab'
>>> # OR
>>> rec.setup_recast(text)
'you can have a look at our catalogu at www.samplewebsite.com in the servic␣
↪tab'
```

3.3.16 LemmatizationRecast

Recast text data by performing lemmatization on it.

> package: string (‘nltk’, ‘spacy’), default=’nltk’ verbose: int (0, 1, -1), default=0

```
>>> from swachhdata.text import LemmatizationRecast
>>> text = 'You can have a look at our catalogue at www.samplewebsite.com in␣
↪the services tab'
>>> rec = LemmatizationRecast()
>>> rec.setup(text)
>>> rec.recast()
'You can have a look at our catalogue at www.samplewebsite.com in the␣
↪service tab'
>>> # OR
>>> rec.setup_recast(text)
'You can have a look at our catalogue at www.samplewebsite.com in the␣
↪service tab'
```

3.3.17 TextRecast

TextRecast is a wrapper function for Recast classes.

text : string / list of strings / pandas.core.series.Series **_kwargs_

- url
- mention
- emoji
- hashtag
- token
- number

ntext [string / list of strings] Processed text

```
>>> { urlRecast = {'process': 'extract_remove'},
>>>   htmlRecast = True,
>>>   EscapeSequenceRecast = True,
>>>   MentionRecast = {'process': 'extract_remove'},
>>>   ContractionsRecast = True,
>>>   CaseRecast = {'process': 'lower'},
>>>   EmojiRecast = {'process': 'extract_remove', 'space_out': False},
>>>   HashtagRecast = {'process': 'extract_remove'},
>>>   ShortWordsRecast = {'min_length': 3},
>>>   StopWordsRecast = {'package': 'nltk', 'space_out': None},
>>>   NumberRecast = {'process': 'remove', 'seperator': None},
>>>   AlphabetRecast = {'process': 'all'},
>>>   PunctuationRecast = True,
>>>   StemmingRecast = {'package': 'nltk', 'method': 'porter'},
>>>   LemmatizationRecast = {'package':'nltk'},
>>>   TokenisationRecast = {'package': 'nltk', 'method': 'sentence' }
```

CHAPTER

FOUR

EXAMPLES

Coming Soon. . .

CHAPTER

FIVE

RELEASE NOTES

5.1 Text

In version *1.0.0* a lot of changes have been brought in, one of them is that all the classes now have three standard methods

- setup()
- recast()
- setup_recast()

New wrapper function for all Text Modules was introduced *TextRecast*

Note: (Classes may or may not have other attributes and parameters)

5.2 Quantitative

Alpha testing of Quantitative module has started

5.3 Qualitative

Qualitative module is under development

5.4 Legacy

Old text functions can still be accessed from the legacy module

```
>>> from swachhdata.legacy import *
```

CHAPTER SIX

CONTRIBUTE

Write a mail to sethkritik@gmail.com if you are interested in contributing