spammy Documentation

Release 1.0.3

Tasdik Rahman

May 07, 2017

Contents

1 1.1 Overview **3**

2 1.2 Features **5**

3 1.3 Example **7**

4 1.4 Installation **9**
- 4.1 1.4.1 Upgrading . . . 9
- 4.2 1.4.2 Installation behind a proxy . . . 9

5 1.5 Benchmarks **11**

6 1.6 Contributing **13**
- 6.1 1.6.1 Roadmap . . . 13

7 1.7 Licensing **15**

8 1.8 Credits **17**

9 2 Documentation **19**
- 9.1 API Reference . . . 19
  - 9.1.1 Spammy Classes . . . 19
  - 9.1.2 spammy.classifier module . . . 19
    - 9.1.2.1 References . . . 19
  - 9.1.3 spammy.exceptions module . . . 21
  - 9.1.4 spammy.train module . . . 21
  - 9.1.5 spammy.version module . . . 22
  - 9.1.6 Module contents . . . 22

10 3 Indices and tables **25**

Contents

- *1 spammy*
  - *1.1 Overview*
  - *1.2 Features*
  - *1.3 Example*
  - *1.4 Installation*
    - *1.4.1 Upgrading*
    - *1.4.2 Installation behind a proxy*
  - *1.5 Benchmarks*
  - *1.6 Contributing*
    - *1.6.1 Roadmap*
  - *1.7 Licensing*
  - *1.8 Credits*
- *2 Documentation*
- *3 Indices and tables*

Author Tasdik Rahman

Latest version 1.0.0

CHAPTER 1

1.1 Overview

spammy : Spam filtering at your service

CHAPTER 2

1.2 Features

- train the classifier on your own dataset to classify your emails into spam or ham
- Dead simple to use. See *usage*
- Blazingly fast once the classifier is trained. (See *benchmarks*)
- Custom exceptions raised so that when you miss something, spammy tells you where did you go wrong in a graceful way
- Written in uncomplicated `python`
- Built on top of the giant shoulders of nltk

CHAPTER 3

1.3 Example

[back to top]

```
>>> import os
>>> from spammy import Spammy
>>>
>>> directory = '/home/tasdik/Dropbox/projects/spamfilter/data/corpus3'
>>>
>>> # directory structure
>>> os.listdir(directory)
['spam', 'Summary.txt', 'ham']
>>> os.listdir(os.path.join(directory, 'spam'))[:5]
['4257.2005-04-06.BG.spam.txt', '0724.2004-09-21.BG.spam.txt', '2835.2005-01-19.BG.
↪spam.txt', '2505.2005-01-03.BG.spam.txt', '3992.2005-03-19.BG.spam.txt']
>>>
>>> # Spammy object created
>>> cl = Spammy(directory, limit=100)
>>> cl.train()
>>>
>>> SPAM_TEXT = \
... """
... My Dear Friend,
...
... How are you and your family? I hope you all are fine.
...
... My dear I know that this mail will come to you as a surprise, but it's for my
... urgent need for a foreign partner that made me to contact you for your sincere
... genuine assistance My name is Mr.Herman Hirdiramani, I am a banker by
... profession currently holding the post of Director Auditing Department in
... the Islamic Development Bank(IsDB)here in Ouagadougou, Burkina Faso.
...
... I got your email information through the Burkina's Chamber of Commerce
... and industry on foreign business relations here in Ouagadougou Burkina Faso
... I haven'disclose this deal to any body I hope that you will not expose or
... betray this trust and confident that I am about to repose on you for the
... mutual benefit of our both families.
```

```
...
... I need your urgent assistance in transferring the sum of Eight Million,
... Four Hundred and Fifty Thousand United States Dollars ($8,450,000:00) into
... your account within 14 working banking days This money has been dormant for
... years in our bank without claim due to the owner of this fund died along with
... his entire family and his supposed next of kin in an underground train crash
... since years ago. For your further informations please visit
... (http://news.bbc.co.uk/2/hi/5141542.stm)
... """
>>> cl.classify(SPAM_TEXT)
'spam'
>>>
```

Accuracy of the classifier

```
>>> from spammy import Spammy
>>> directory = '/home/tasdik/Dropbox/projects/spammy/examples/training_dataset'
>>> cl = Spammy(directory, limit=300)  # training on only 300 spam and ham files
>>> cl.train()
>>> cl.accuracy(directory='/home/tasdik/Dropbox/projects/spammy/examples/test_dataset
↪', label='spam', limit=300)
0.9554794520547946
>>> cl.accuracy(directory='/home/tasdik/Dropbox/projects/spammy/examples/test_dataset
↪', label='ham', limit=300)
0.9033333333333333
>>>
```

**More examples can be found over in the examples directory **

CHAPTER 4

1.4 Installation

[back to top]

Install the dependencies first

```
$ pip install nltk==3.2.1, beautifulsoup4==4.4.1
```

To install use pip:

```
$ pip install spammy
```

or use easy_install

```
$ easy_install spammy
```

Or build it yourself (only if you must):

```
$ git clone https://github.com/prodicus/spammy.git
$ python setup.py install
```

1.4.1 Upgrading

To upgrade the package,

```
$ pip install -U spammy
```

1.4.2 Installation behind a proxy

If you are behind a proxy, then this should work

```
$ pip --proxy [username:password@]domain_name:port install spammy
```

CHAPTER 5

1.5 Benchmarks

[back to top]

Spammy is blazingly fast once trained

Don't believe me? Have a look

```
>>> import timeit
>>> from spammy import Spammy
>>>
>>> directory = '/home/tasdik/Dropbox/projects/spamfilter/data/corpus3'
>>> cl = Spammy(directory, limit=100)
>>> cl.train()
>>> SPAM_TEXT_2 = \
... """
... INTERNATIONAL MONETARY FUND (IMF)
... DEPT: WORLD DEBT RECONCILIATION AGENCIES.
... ADVISE: YOUR OUTSTANDING PAYMENT NOTIFICATION
...
... Attention
... A power of attorney was forwarded to our office this morning by two gentle men,
... one of them is an American national and he is MR DAVID DEANE by name while the
... other person is MR... JACK MORGAN by name a CANADIAN national.
... This gentleman claimed to be your representative, and this power of attorney
... stated that you are dead; they brought an account to replace your information
... in other to claim your fund of (US$9.7M) which is now lying DORMANT and UNCLAIMED,
...  below is the new account they have submitted:
...                     BANK.-HSBC CANADA
...                     Vancouver, CANADA
...                     ACCOUNT NO. 2984-0008-66
...
... Be further informed that this power of attorney also stated that you suffered.
... """
>>>
>>> def classify_timeit():
...    result = cl.classify(SPAM_TEXT_2)
...
```

```
>>> timeit.repeat(classify_timeit, number=5)
[0.1810469627380371, 0.16121697425842285, 0.16121196746826172]
>>>
```

CHAPTER 6

1.6 Contributing

[back to top]

Refer CONTRIBUTING page for details

1.6.1 Roadmap

- Include more algorithms for increased accuracy

CHAPTER 7

1.7 Licensing

[back to top]

Spammy is built by Tasdik Rahman and licensed under GPLv3.

You can find a full copy of the LICENSE file here

CHAPTER 8

1.8 Credits

[back to top]

If you'd like give me credit somewhere on your blog or tweet a shout out to @tasdikrahman, well hey, I'll take it.

CHAPTER 9

2 Documentation

API Reference

Spammy Classes

spammy.classifier module

Rolling my own Implementation of Naive Bayes algorithm.

This particular implementation caters to the case when a category is not observed in the dataset, and the model automatically assigns a 0 probability to it!

Read about smoothening techniques somewhere but let's not delve into that now.

References

[1]

- http://stackoverflow.com/a/5029989/3834059
- http://stackoverflow.com/q/8419401/3834059
- http://stackoverflow.com/q/2600790/3834059

class `spammy.classifier.`**`NaiveBayesClassifier`**

Bases: `object`

Inherits from the 'object' class. Nothing special

`classify`(*features*)

Writing the actual interface for the class here. This will classify our documents when called from the terminal

Parameters

- **`self`** – class object

- **features** – The feaures of the document passed

Returns spam or ham

Return type str

document_probability (*features*, *label*)

Finds `document_probability()` by looping over the documents and calling `feature_probability()`

Parameters

- **self** – class object
- **features** – List of features
- **label** – Label whose probability needs to be classified

Returns the probability of the document in being in a particular class

Return type float/int

feature_probability (*feature*, *label*)

This function calculates the probability of a feature to belong to a particular label. (i.e class of 'spam' or 'ham' for us.)

Note: for an unseen featurem I can assign a random probability, let's say 0.5

Parameters

- **self** – class object
- **feature** – The feature for which we will be calculating the probailty.
- **label** – spam or ham

Returns The probability of the feature being in the label.

Return type float

train (*featurelist*, *label*)

Trains the classifier for gods sake!

Trying to emulate the API which the NLTK wrapper tries to provide for its nltk.NaiveBayesClassifier.train() gives

Note: `defaultdict` is used bacause when we try to acces a key which is not there in the `dictionary`, we get a `KeyError`. Whereas in `defaultdict`.It will try to return a default value if the key is not found.

For more on `defaultdict`, Refer: http://stackoverflow.com/a/5900634/3834059

Parameters

- **self** – class object
- **featurelist** – the list of the features
- **label** – class of the feature

spammy.exceptions module

exception `spammy.exceptions.`**`CorpusFileError`**

Bases: `spammy.exceptions.SpammyError`

Raised when the one of the corpus files passed to the spammy ctor do not exist

OR

When we do not pass any file to the ctor for initialization

exception `spammy.exceptions.`**`LimitError`**

Bases: `spammy.exceptions.SpammyError`

raised when the limit passed is either less than 0

exception `spammy.exceptions.`**`SpammyError`**

Bases: `exceptions.Exception`

A Spammy related error

`spammy.exceptions.`**`SpammyException`**

alias of `SpammyError`

spammy.train module

Trainer class for the classifier

class `spammy.train.`**`Trainer`**(*directory*, *spam*, *ham*, *limit*)

Bases: `object`

The trainer class

`extract_features`(*text*)

Will convert the document into tokens and extract the features.

Note: So these are some possible features which would make an email a SPAM

features looked for - Attachments - Links in text - CAPSLOCK words - Numbers - Words in text

Parameters

- **`self`** – Trainer object
- **`text`** – Email text from which we will extract features

Returns A list which contains the feature set

Return type list

`train`()

Starts the training process on the directories passed by the user

Parameters **`self`** – Trainer object

`train_classifier`(*path*, *label*)

The function doing the actual classification here.

Parameters

- **`self`** – Trainer object

- **path** – The path of the data directory
- **label** – The label underwhich the data directory is

spammy.version module

Module contents

class `spammy.`**`Spammy`**(*directory=None*, *limit=None*, ***kwargs*)

Bases: `object`

Stiches everything from train module and classifier module together

`accuracy`(***kwargs*)

Checks the accuracy of the classifier by running it against a testing corpus

Parameters

- **limit** – number of files the classifier should test upon
- **label** – the label as in spam or ham
- **directory** – The absolute path of the directory to be tested

Returns the precision of the classifier. Eg: 0.87

Return type float

Example

```
>>> from spammy import Spammy
>>> directory = '/home/tasdik/Dropbox/projects/spammy/examples/
↪training_dataset'
>>> cl = Spammy(directory, limit=300)  # training on only 300 spam␣
↪and ham files
>>> cl.train()
>>> cl.accuracy(directory='/home/tasdik/Dropbox/projects/spammy/
↪examples/test_dataset', label='spam', limit=300)
0.9554794520547946
>>> cl.accuracy(directory='/home/tasdik/Dropbox/projects/spammy/
↪examples/test_dataset', label='ham', limit=300)
0.9033333333333333
>>>
```

`classify`(*email_text*)

tries classifying text into spam or ham

Parameters **email_text** – email_text to be passed here which is to be classified

Returns Either ham or spam

Return type str

Note: To be run after you have trained the classifier object on your dataset

Example

```
>>> from spammy import Spammy
>>> cl = Spammy(path_to_trainin_data, limit=200)
# 200 or the number of files you need to train the classifier upon
>>>
>>> HAM_TEXT =                ''' 
Bro. Hope you are fine. Hows the work going on ? Can you send me␣
↪some updates on it.
And are you free tomorrow ?
No problem man. But please make sure you are finishing it
by friday night and sending me on on that day itself. As we
have to get it printed on Saturday.
'''
>>> cl.classify(HAM_TEXT)
'ham'
```

train()

Trains the classifier object

Parameters **self** – the classifier object

Example

```
>>> from spammy import Spammy
>>> directory = '/home/tasdik/Dropbox/projects/spammy/examples/
↪training_dataset'
>>> cl = Spammy(directory, limit=300)  # training on only 300 spam␣
↪and ham files
>>> cl.train()
```

CHAPTER 10

3 Indices and tables

- genindex
- modindex
- search

Index

A

accuracy() (spammy.Spammy method), 22

C

classify() (spammy.classifier.NaiveBayesClassifier method), 19
classify() (spammy.Spammy method), 22
CorpusFileError, 21

D

document_probability() (spammy.classifier.NaiveBayesClassifier method), 20

E

extract_features() (spammy.train.Trainer method), 21

F

feature_probability() (spammy.classifier.NaiveBayesClassifier method), 20

L

LimitError, 21

N

NaiveBayesClassifier (class in spammy.classifier), 19

S

Spammy (class in spammy), 22
spammy (module), 22
spammy.classifier (module), 19
spammy.exceptions (module), 21
spammy.train (module), 21
spammy.version (module), 22
SpammyError, 21
SpammyException (in module spammy.exceptions), 21

T

train() (spammy.classifier.NaiveBayesClassifier method), 20
train() (spammy.Spammy method), 23
train() (spammy.train.Trainer method), 21
train_classifier() (spammy.train.Trainer method), 21
Trainer (class in spammy.train), 21