Bayes Documentation

Release 0.1.0

Krzysztof Joachimiak

January 14, 2017

Contents

This module contains several variants of well-known Naive Bayes Classifier.

Contents:

CHAPTER 1

Bayes API

1.1 Classifiers

1.1.1 Complement Naive Bayes

class bayes.classifiers.cnb.**ComplementNB**(*alpha=1.0*, *weight_normalized=False*)

Bases: bayes.base.BaseNB

Complement Naive Bayes classifier

Parameters

- **alpha** (*float*) – Smoothing parameter
- **weight_normalized** (*bool, default False*) – Enable Weight-normalized Complement Naive Bayes method.

alpha_sum_
int

Sum of alpha params

classes_
array, shape (n_classes,)

Classes list

class_count_
array, shape (n_classes,)

number of training samples observed in each class.

Examples

```
>>> from sklearn.datasets import fetch_20newsgroups
>>> from sklearn.feature_extraction.text import CountVectorizer
>>> from bayes.classifiers import ComplementNB
Prepare data
>>> vectorizer = CountVectorizer()
>>> categories = ['alt.atheism', 'talk.religion.misc','comp.graphics', 'sci.space']
Train set
>>> newsgroups_train = fetch_20newsgroups(subset='train', categories=categories, shuffle=True)
>>> train_vectors = vectorizer.fit_transform(newsgroups_train.data)
Test set
```

```
>>> newsgroups_test = fetch_20newsgroups(subset='test', categories=categories, shuffle=True)
>>> test_vectors = vectorizer.transform(newsgroups_test.data)
>>> clf = ComplementNB()
>>> clf.fit(newsgroups_train, train_vectors).accuracy_score(newsgroups_test, test_vectors)
```

References

Rennie J. D. M., Shih L., Teevan J., Karger D. R. (2003). Tackling the Poor Assumptions of Naive Bayes Text Classifiers

https://people.csail.mit.edu/jrennie/papers/icml03-nb.pdf

__delattr__
x.__delattr__('name') <==> del x.name

__format__()
default object formatter

__getattribute__
x.__getattribute__('name') <==> x.name

__hash__

__reduce__()
helper for pickle

__reduce_ex__()
helper for pickle

__setattr__
x.__setattr__('name', value) <==> x.name = value

__sizeof__() → int
size of object in memory, in bytes

__str__

accuracy_score(*X*, *y*)
Return acuracy score

Parameters

- **X** (*{array-like, sparse matrix}, shape = [n_samples, n_features]*) – Training vectors, where n_samples is the number of samples and n_features is the number of features.
- **y** (*array-like, shape = [n_samples]*) – Target values.

Returns **accuracy_score** – Accuracy on the given test set

Return type float

class_log_proba_
Log probability of class occurrence

complement_class_count_
Complement class count, i.e. number of occurrences of all the samples with all the classes except the given class c

complement_class_log_proba_
Complement class probability, i.e. logprob of occurrence of a sample, which does not belong to the given class c

fit (*X*, *y*)
Fit model to given training set

Parameters

- **X** (*array-like, shape (n_samples, n_features)*) – Training vectors, where n_samples is the number of samples and n_features is the number of features.
- **y** (*array-like, shape (n_samples,)*) – Target values.

Returns **self** – Returns self.

Return type Naive Bayes estimator object

get_params (*deep=True*)
Get parameters for this estimator.

Parameters **deep** (*boolean, optional*) – If True, will return the parameters for this estimator and contained subobjects that are estimators.

Returns **params** – Parameter names mapped to their values.

Return type mapping of string to any

partial_fit (*X*, *y*, *classes=None*)
Incremental fit on a batch of samples.

Parameters

- **X** (*{array-like, sparse matrix}, shape = [n_samples, n_features]*) – Training vectors, where n_samples is the number of samples and n_features is the number of features.
- **y** (*array-like, shape = [n_samples]*) – Target values.
- **classes** (*array-like, shape = [n_classes], optional (default=None)*) – List of all the classes that can possibly appear in the y vector. Must be provided at the first call to partial_fit, can be omitted in subsequent calls.

Returns **self** – Returns self.

Return type object

predict (*X*)
Perform classification on an array of test vectors X.

Parameters **X** (*array-like, shape = [n_samples, n_features]*) – Unseen samples vector

Returns **C** – Predicted target values for X

Return type array, shape = [n_samples]

predict_log_proba (*X*)
Return log-probability estimates for the test vector X.

Parameters **X** (*array-like, shape = [n_samples, n_features]*) –

Returns **C** – Returns the log-probability of the samples for each class in the model. The columns correspond to the classes in sorted order, as they appear in the attribute *classes_*.

Return type array-like, shape = [n_samples, n_classes]

predict_proba(*X*)
Return probability estimates for the test vector X. :param X: :type X: array-like, shape = [n_samples, n_features]

Returns **C** – Returns the probability of the samples for each class in the model. The columns correspond to the classes in sorted order, as they appear in the attribute *classes_*.

Return type array-like, shape = [n_samples, n_classes]

safe_matmult(*input_array*, *internal_array*)

set_params(***params*)
Set the parameters of this estimator.

The method works on simple estimators as well as on nested objects (such as pipelines). The latter have parameters of the form `<component>__<parameter>` so that it's possible to update each component of a nested object.

Returns

Return type self

1.1.2 Locally Weighted Naive Bayes

class `bayes.classifiers.lwnb.`**LocallyWeightedNB**(*alpha=1.0*, *weight_normalized=False*)
Bases: `bayes.base.BaseNB`

Locally Weighted Naive Bayes classifier

Parameters

- **alpha** (*float*) – Smoothing parameter
- **weight_normalized** (*bool, default False*) – Enable Weight-normalized Complement Naive Bayes method.

References

Frank E., Hall M., Pfahringer B. (2003).

http://www.cs.waikato.ac.nz/~eibe/pubs/UAI_200.pdf

__delattr__
x.__delattr__('name') <==> del x.name

__format__()
default object formatter

__getattribute__
x.__getattribute__('name') <==> x.name

__hash__

__reduce__()
helper for pickle

__reduce_ex__()
helper for pickle

__setattr__
x.__setattr__('name', value) <==> x.name = value

__sizeof__() → int
size of object in memory, in bytes

__str__

accuracy_score(*X*, *y*)
Return acuracy score

Parameters

- **X** (*{array-like, sparse matrix}, shape = [n_samples, n_features]*) – Training vectors, where n_samples is the number of samples and n_features is the number of features.
- **y** (*array-like, shape = [n_samples]*) – Target values.

Returns **accuracy_score** – Accuracy on the given test set

Return type float

class_log_proba_
Log probability of class occurrence

complement_class_count_
Complement class count, i.e. number of occurrences of all the samples with all the classes except the given class c

complement_class_log_proba_
Complement class probability, i.e. logprob of occurrence of a sample, which does not belong to the given class c

fit(*X*, *y*)

get_params()

partial_fit(*X*, *y*, *classes=None*)

predict(*X*)

predict_log_proba(*X*)

predict_proba(*X*)

safe_matmult(*input_array*, *internal_array*)

set_params(***params*)

1.1.3 Negation Naive Bayes

class bayes.classifiers.nnb.**NegationNB**(*alpha=1.0*)
Bases: bayes.base.BaseNB

Negation Naive Bayes classifier

Parameters **alpha** (*float*) – Smoothing parameter

References

Komiya K., Sato N., Fujimoto K., Kotani Y. (2011). Negation Naive Bayes for Categorization of Product Pages on the Web

http://www.aclweb.org/anthology/R11-1083.pdf

__delattr__
x.__delattr__('name') <==> del x.name

__format__()
default object formatter

__getattribute__
x.__getattribute__('name') <==> x.name

__hash__

__reduce__()
helper for pickle

__reduce_ex__()
helper for pickle

__setattr__
x.__setattr__('name', value) <==> x.name = value

__sizeof__() → int
size of object in memory, in bytes

__str__

accuracy_score(*X*, *y*)
Return acuracy score

Parameters

- **X** (*{array-like, sparse matrix}, shape = [n_samples, n_features]*) – Training vectors, where n_samples is the number of samples and n_features is the number of features.
- **y** (*array-like, shape = [n_samples]*) – Target values.

Returns **accuracy_score** – Accuracy on the given test set

Return type float

class_log_proba_
Log probability of class occurrence

complement_class_count_
Complement class count, i.e. number of occurrences of all the samples with all the classes except the given class c

complement_class_log_proba_
Complement class probability, i.e. logprob of occurrence of a sample, which does not belong to the given class c

fit(*X*, *y*)
Fit model to given training set

Parameters

- **X** (*array-like, shape (n_samples, n_features)*) – Training vectors, where n_samples is the number of samples and n_features is the number of features.
- **y** (*array-like, shape (n_samples,)*) – Target values.

Returns **self** – Returns self.

Return type Naive Bayes estimator object

get_params(*deep=True*)
Get parameters for this estimator.

Parameters **deep** (*boolean, optional*) – If True, will return the parameters for this estimator and contained subobjects that are estimators.

Returns **params** – Parameter names mapped to their values.

Return type mapping of string to any

partial_fit(*X*, *y*, *classes=None*)
Incremental fit on a batch of samples.

Parameters

- **X** (*{array-like, sparse matrix}, shape = [n_samples, n_features]*) – Training vectors, where n_samples is the number of samples and n_features is the number of features.
- **y** (*array-like, shape = [n_samples]*) – Target values.
- **classes** (*array-like, shape = [n_classes], optional (default=None)*) – List of all the classes that can possibly appear in the y vector. Must be provided at the first call to partial_fit, can be omitted in subsequent calls.

Returns **self** – Returns self.

Return type object

predict(*X*)
Perform classification on an array of test vectors X.

Parameters **X** (*array-like, shape = [n_samples, n_features]*) – Unseen samples vector

Returns **C** – Predicted target values for X

Return type array, shape = [n_samples]

predict_log_proba(*X*)
Return log-probability estimates for the test vector X.

Parameters **X** (*array-like, shape = [n_samples, n_features]*) –

Returns **C** – Returns the log-probability of the samples for each class in the model. The columns correspond to the classes in sorted order, as they appear in the attribute *classes_*.

Return type array-like, shape = [n_samples, n_classes]

predict_proba(*X*)
Return probability estimates for the test vector X. :param X: :type X: array-like, shape = [n_samples, n_features]

Returns **C** – Returns the probability of the samples for each class in the model. The columns correspond to the classes in sorted order, as they appear in the attribute *classes_*.

Return type array-like, shape = [n_samples, n_classes]

safe_matmult(*input_array*, *internal_array*)

set_params(***params*)
Set the parameters of this estimator.

The method works on simple estimators as well as on nested objects (such as pipelines). The latter have parameters of the form `<component>__<parameter>` so that it's possible to update each component of a nested object.

Returns

Return type self

1.1.4 Selective Naive Bayes

class `bayes.classifiers.snb.`**SelectiveNB**(*alpha=1.0*)

Bases: `bayes.base.BaseNB`

Selective Naive Bayes classifier

Parameters **alpha** (*float*) – Smoothing parameter

References

Komiya K., Ito Y., Kotani Y. (2013). New Naive Bayes Methods using Data from All Classes

http://aia-i.com/ijai/sample/vol5/no1/1-13.pdf

__delattr__
x.__delattr__('name') <==> del x.name

__format__()
default object formatter

__getattribute__
x.__getattribute__('name') <==> x.name

__hash__

__reduce__()
helper for pickle

__reduce_ex__()
helper for pickle

__setattr__
x.__setattr__('name', value) <==> x.name = value

__sizeof__() → int
size of object in memory, in bytes

__str__

accuracy_score(*X*, *y*)
Return acuracy score

Parameters

- **X** (*{array-like, sparse matrix}, shape = [n_samples, n_features]*) – Training vectors, where n_samples is the number of samples and n_features is the number of features.
- **y** (*array-like, shape = [n_samples]*) – Target values.

Returns **accuracy_score** – Accuracy on the given test set

Return type float

class_log_proba_
Log probability of class occurrence

complement_class_count_
Complement class count, i.e. number of occurrences of all the samples with all the classes except the given class c

complement_class_log_proba_
Complement class probability, i.e. logprob of occurrence of a sample, which does not belong to the given class c

fit (*X*, *y*)
Fit model to given training set

Parameters

- **X** (*array-like, shape (n_samples, n_features)*) – Training vectors, where n_samples is the number of samples and n_features is the number of features.
- **y** (*array-like, shape (n_samples,)*) – Target values.

Returns **self** – Returns self.

Return type Naive Bayes estimator object

get_params (*deep=True*)
Get parameters for this estimator.

Parameters **deep** (*boolean, optional*) – If True, will return the parameters for this estimator and contained subobjects that are estimators.

Returns **params** – Parameter names mapped to their values.

Return type mapping of string to any

partial_fit (*X*, *y*, *classes=None*)
Incremental fit on a batch of samples.

Parameters

- **X** (*{array-like, sparse matrix}, shape = [n_samples, n_features]*) – Training vectors, where n_samples is the number of samples and n_features is the number of features.
- **y** (*array-like, shape = [n_samples]*) – Target values.
- **classes** (*array-like, shape = [n_classes], optional (default=None)*) – List of all the classes that can possibly appear in the y vector. Must be provided at the first call to partial_fit, can be omitted in subsequent calls.

Returns **self** – Returns self.

Return type object

predict (*X*)
Perform classification on an array of test vectors X.

Parameters **X** (*array-like, shape = [n_samples, n_features]*) – Unseen samples vector

Returns **C** – Predicted target values for X

Return type array, shape = [n_samples]

predict_log_proba(*X*)
Return log-probability estimates for the test vector X.

Parameters **X** (*array-like, shape = [n_samples, n_features]*) –

Returns **C** – Returns the log-probability of the samples for each class in the model. The columns correspond to the classes in sorted order, as they appear in the attribute *classes_*.

Return type array-like, shape = [n_samples, n_classes]

predict_proba(*X*)
Return probability estimates for the test vector X. :param X: :type X: array-like, shape = [n_samples, n_features]

Returns **C** – Returns the probability of the samples for each class in the model. The columns correspond to the classes in sorted order, as they appear in the attribute *classes_*.

Return type array-like, shape = [n_samples, n_classes]

safe_matmult(*input_array*, *internal_array*)

set_params(***params*)
Set the parameters of this estimator.

The method works on simple estimators as well as on nested objects (such as pipelines). The latter have parameters of the form `<component>__<parameter>` so that it's possible to update each component of a nested object.

Returns

Return type self

1.1.5 Universal-set Naive Bayes

class bayes.classifiers.unb.**UniversalSetNB**(*alpha=1.0*)
Bases: bayes.base.BaseNB

Universal-set Naive Bayes classifier

Parameters **alpha** (*float*) – Smoothing parameter

References

Komiya K., Ito Y., Kotani Y. (2013). New Naive Bayes Methods using Data from All Classes

http://aia-i.com/ijai/sample/vol5/no1/1-13.pdf

__delattr__
x.__delattr__('name') <==> del x.name

__format__()
default object formatter

__getattribute__
x.__getattribute__('name') <==> x.name

__hash__

__reduce__()
helper for pickle

__reduce_ex__()
helper for pickle

__setattr__
x.__setattr__('name', value) <==> x.name = value

__sizeof__() → int
size of object in memory, in bytes

__str__

accuracy_score(*X*, *y*)
Return acuracy score

Parameters

- **X** (*{array-like, sparse matrix}, shape = [n_samples, n_features]*) – Training vectors, where n_samples is the number of samples and n_features is the number of features.
- **y** (*array-like, shape = [n_samples]*) – Target values.

Returns **accuracy_score** – Accuracy on the given test set

Return type float

class_log_proba_
Log probability of class occurrence

complement_class_count_
Complement class count, i.e. number of occurrences of all the samples with all the classes except the given class c

complement_class_log_proba_
Complement class probability, i.e. logprob of occurrence of a sample, which does not belong to the given class c

fit(*X*, *y*)
Fit model to given training set

Parameters

- **X** (*array-like, shape (n_samples, n_features)*) – Training vectors, where n_samples is the number of samples and n_features is the number of features.
- **y** (*array-like, shape (n_samples,)*) – Target values.

Returns **self** – Returns self.

Return type Naive Bayes estimator object

get_params(*deep=True*)
Get parameters for this estimator.

Parameters **deep** (*boolean, optional*) – If True, will return the parameters for this estimator and contained subobjects that are estimators.

Returns **params** – Parameter names mapped to their values.

Return type mapping of string to any

partial_fit(*X*, *y*, *classes=None*)
Incremental fit on a batch of samples.

Parameters

- **X** (*{array-like, sparse matrix}, shape = [n_samples, n_features]*) – Training vectors, where n_samples is the number of samples and n_features is the number of features.
- **y** (*array-like, shape = [n_samples]*) – Target values.
- **classes** (*array-like, shape = [n_classes], optional (default=None)*) – List of all the classes that can possibly appear in the y vector. Must be provided at the first call to partial_fit, can be omitted in subsequent calls.

Returns **self** – Returns self.

Return type object

predict(*X*)

Perform classification on an array of test vectors X.

Parameters **X** (*array-like, shape = [n_samples, n_features]*) – Unseen samples vector

Returns **C** – Predicted target values for X

Return type array, shape = [n_samples]

predict_log_proba(*X*)

Return log-probability estimates for the test vector X.

Parameters **X** (*array-like, shape = [n_samples, n_features]*) –

Returns **C** – Returns the log-probability of the samples for each class in the model. The columns correspond to the classes in sorted order, as they appear in the attribute *classes_*.

Return type array-like, shape = [n_samples, n_classes]

predict_proba(*X*)

Return probability estimates for the test vector X. :param X: :type X: array-like, shape = [n_samples, n_features]

Returns **C** – Returns the probability of the samples for each class in the model. The columns correspond to the classes in sorted order, as they appear in the attribute *classes_*.

Return type array-like, shape = [n_samples, n_classes]

safe_matmult(*input_array*, *internal_array*)

set_params(***params*)

Set the parameters of this estimator.

The method works on simple estimators as well as on nested objects (such as pipelines). The latter have parameters of the form `<component>__<parameter>` so that it's possible to update each component of a nested object.

Returns

Return type self

1.2 Utils

1.2.1 Submodules

1.2.2 bayes.utils.math_utils

1.2.3 bayes.utils.doc_utils

CHAPTER 2

Algorithms

TODO

CHAPTER 3

Installation

You can install this module directly from GitHub repo with command:

or use **pip**

CHAPTER 4

Usage

TODO

- genindex

modindex search

b

Symbols

A

B

C

F

G

L

N

P

S

U