brew Documentation

Release 0.1.3

Dayvid Victor

May 28, 2016

Contents

brew: A Multiple Classifier Systems API

This project was started in 2014 by *Dayvid Victor* and *Thyago Porpino*
for the Multiple Classifier Systems class at Federal University of Pernambuco.

The aim of this project is to provide an easy API for Ensembling, Stacking,
Blending, Ensemble Generation, Ensemble Pruning, Dynamic Classifier Selection,
and Dynamic Ensemble Selection.

CHAPTER 1

Example

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec
import itertools

import sklearn

from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier

from brew.base import Ensemble, EnsembleClassifier
from brew.stacking.stacker import EnsembleStack, EnsembleStackClassifier
from brew.combination.combiner import Combiner

from mlxtend.data import iris_data
from mlxtend.evaluate import plot_decision_regions

# Initializing Classifiers
clf1 = LogisticRegression(random_state=0)
clf2 = RandomForestClassifier(random_state=0)
clf3 = SVC(random_state=0, probability=True)

# Creating Ensemble
ensemble = Ensemble([clf1, clf2, clf3])
eclf = EnsembleClassifier(ensemble=ensemble, combiner=Combiner('mean'))

# Creating Stacking
layer_1 = Ensemble([clf1, clf2, clf3])
layer_2 = Ensemble([sklearn.clone(clf1)])

stack = EnsembleStack(cv=3)

stack.add_layer(layer_1)
stack.add_layer(layer_2)

sclf = EnsembleStackClassifier(stack)

clf_list = [clf1, clf2, clf3, eclf, sclf]
lbl_list = ['Logistic Regression', 'Random Forest', 'RBF kernel SVM', 'Ensemble', 'Stacking']

# Loading some example data
X, y = iris_data()
```

```
X = X[:,[0, 2]]

# Plotting Decision Regions
gs = gridspec.GridSpec(2, 3)
fig = plt.figure(figsize=(10, 8))

itt = itertools.product([0, 1, 2], repeat=2)

for clf, lab, grd in zip(clf_list, lbl_list, itt):
    clf.fit(X, y)
    ax = plt.subplot(gs[grd[0], grd[1]])
    fig = plot_decision_regions(X=X, y=y, clf=clf, legend=2)
    plt.title(lab)
plt.show()
```

1.1 Features

- Ensembling, Blending and Stacking.
- Dynamic Classifier Selection: OLA, LCA, A Priori, A Posteriori.
- Dynamic Ensemble Selection: KNORA E and KNORA U.
- Ensemble Combination Rules: majority vote, min, max, mean and median.
- Ensemble Diversity Metrics: Entropy Measure E, Kohavi Wolpert Variance, | Q Statistics, Correlation Coefficient p, Disagreement Measure, Agreement Measure, Double Fault Measure.
- Ensemble Classifier Generators: Bagging, Random Subspace, SMOTEBagging, ICS-Bagging, SMOTE-ICS-Bagging.
- Ensemble Pruning: EPIC.
- Oversampling: SMOTE.

CHAPTER 2

Important References

- Kuncheva, Ludmila I. Combining pattern classifiers: methods and algorithms. John Wiley & Sons, 2014.
- Zhou, Zhi-Hua. Ensemble methods: foundations and algorithms. CRC Press, 2012.

CHAPTER 3

Dependencies

- Python 2.6+
- scikit-learn >= 0.14.1
- Numpy >= 1.3
- SciPy >= 0.7
- Matplotlib >= 0.99.1 (examples, only)
- mlxtend (examples, only)

CHAPTER 4

Contents:

4.1 brew package

4.1.1 Subpackages

brew.combination package

Submodules

brew.combination.combiner module

brew.combination.rules module

Module contents

brew.generation package

Submodules

brew.generation.bagging module

brew.generation.base module

brew.generation.random_subspace module

brew.generation.smote_bagging module

Module contents

brew.metrics package

Subpackages

brew.metrics.diversity package

Submodules

brew.metrics.diversity.base module

brew.metrics.diversity.non_paired module

brew.metrics.diversity.paired module

Module contents

Submodules

brew.metrics.evaluation module

Module contents

brew.preprocessing package

Submodules

brew.preprocessing.smote module

Module contents

brew.selection package

Subpackages

brew.selection.dynamic package

Submodules

brew.selection.dynamic.base module

brew.selection.dynamic.knora module

brew.selection.dynamic.lca module

brew.selection.dynamic.ola module

Module contents

brew.selection.pruning package

Submodules

brew.selection.pruning.epic module

Module contents

Module contents

brew.utils package

Submodules

brew.utils.data module

Module contents

4.1.2 Submodules

4.1.3 brew.base module

4.1.4 Module contents

4.2 Installation

At the command line either via easy_install or pip:

```
$ easy_install brew
$ pip install brew
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv brew
$ pip install brew
```

4.3 Usage

To use brew in a project:

```
import brew
from brew.base import Ensemble
from brew.base import EnsembleClassifier
from brew.combination import Combiner

# here, clf1 and clf2 are sklearn classifiers or brew ensemble classifiers
# already trained. Keep in mind that brew requires your labels = [0,1,2,...]
# numerical with no skips.
clfs = [clf1, clf2]
ens = Ensemble(classifiers = clfs)

# create your Combiner
```

```
# the rules can be 'majority_vote', 'max', 'min', 'mean' or 'median'
comb = Combiner(rule='mean')

# now create your ensemble classifier
ensemble_clf = EnsembleClassifier(ensemble=ens, combiner=comb)
y_pred = ensemble_clf.predict(X)

# there you go, y_pred is your prediction.
```

4.4 Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

4.4.1 Types of Contributions

Report Bugs

Report bugs at https://github.com/viisar/brew/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" is open to whoever wants to implement it.

Implement Features

Look through the GitHub issues for features. Anything tagged with "feature" is open to whoever wants to implement it.

Write Documentation

brew could always use more documentation, whether as part of the official brew docs, in docstrings, or even on the web in blog posts, articles, and such.

Submit Feedback

The best way to send feedback is to file an issue at https://github.com/viisar/brew/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.

- Remember that this is a volunteer-driven project, and that contributions are welcome :)

4.4.2 Get Started!

Ready to contribute? Here's how to set up *brew* for local development.

1. **Fork_** the *brew* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/brew.git
```

3. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

4. When you're done making changes, check that your changes pass style and unit tests, including testing other Python versions with tox:

```
$ tox
```

To get tox, just pip install it.

5. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

6. Submit a pull request through the GitHub website.

4.4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.6, 2.7, and 3.3, and for PyPy. Check https://travis-ci.org/viisar/brew under pull requests for active pull requests or run the `tox` command and make sure that the tests pass for all supported Python versions.

4.4.4 Tips

To run a subset of tests:

```
$ py.test test/test_brew.py
```

4.5 Credits

4.5.1 Development Lead

- Dayvid Victor <victor.dvro@gmail.com>
- Thyago Porpino <thyago.porpino@gmail.com>

4.5.2 Contributors

None yet. Why not be the first?

4.6 History

4.6.1 0.1.0 (2014-11-12)

- First release on PyPI.

CHAPTER 5

Feedback

If you have any suggestions or questions about **brew** feel free to email me at victor.dvro@gmail.com.

If you encounter any errors or problems with **brew**, please let me know! Open an Issue at the GitHub http://github.com/dvro/brew main repository.

b

B