Student Information

Name: Vinayak Mehta

E-mail: vmehta94@gmail.com

Telephone: +91-9868525444

Time Zone: GMT+5.5 (New Delhi, India)

IRC: vortex_ape

GitHub: vortex-ape

Twitter: vortex_ape

Blog: http://vinayakmehta.me

GSoC Blog RSS feed: http://vinayakmehta.me/feeds/all.rss.xml

My background and programming experience:

I am a third year undergraduate student of computer science and engineering at Bharati Vidyapeeth's College of Engineering, Delhi. I started programming with C++ in class 11th in which I made a couple of projects. In the second semester of college, I took an Introduction to programming course in which I learned C followed by a Java based programming course in fifth semester. I also have experience in MATLAB (for my Digital Signal Processing assignments). I'm a MOOC fan and have taken a number of online courses some of which include [Algorithms: Design and Analysis](http://vinayakmehta.me/files/algos_SoA.pdf) (in addition to an algorithms course taught at my college), Introduction to Machine Learning (Udacity) and the Machine Learning course offered by Georgia Tech at Udacity for their Online MS CS program. At the starting of my fifth semester, I was introduced to Python, and it was love at first sight. I have coded a couple of projects in Python, using scikit-learn for the latter two courses, which are displayed in the Additional Information section of this proposal. Currently, I'm taking Prof Ng's Machine Learning course at Coursera in addition to an Artificial Intelligence course at edX which are scheduled to finish in April 2015. I like to take part in programming contests and have qualified for the onsite rounds of ACM ICPC multiple times. Programming contests have given me experience in implementing complex algorithms.

University Information

University: Guru Gobind Singh Indraprastha University

Major: Computer Science and Engineering

Current Year and Expected Graduation Date: Third year, 1 August, 2016

Degree: Bachelor of Technology

Project Proposal Information

Title: scikit-learn: Cross-validation and Meta-estimators for Semi-supervised learning

Abstract:

This proposal aims to make the `semi_supervised` module a first-class citizen in scikit-learn. It can be achieved by improving `cross_validation` module's support for it. A self-learning meta-estimator will be implemented. New algorithms will be added to the `semi_supervised` module in addition to improving existing ones.

Motivation:

Semi-supervised learning is still not a first-class citizen in scikit-learn despite it being an important class of machine learning, often outperforming supervised models. It would be awesome if users spend their time concentrating on the research task at hand instead of wasting it on acquiring large amounts of labelled data, which is rather difficult.

I have been contributing to scikit-learn for over a month now, and understand the code base and other coding procedures well enough for the successful completion of my proposed project. I also have other machine learning experience I got from projects which I've mentioned in additional information.

Milestones:

1. Improve the `cross_validation` module to accommodate semi_supervised learning better, seeing that the new changes play well with the rest of scikit-learn.
2. Improve existing `semi_supervised` learning module to accommodate the changes made to `cross_validation`.
3. Implement a meta-estimator based on the self-learning method.
4. Implement the proposed semi-supervised learning algorithms.
5. Write benchmarks, tests, documentation and examples for the newly added components.

Details:

Fixing cross validation for semi-supervised learning

Currently, if we apply cross validation on **X** which contains both labeled and unlabeled data, it sends unlabeled data into the test set too which doesn't make sense for scoring. This will be fixed after a discussion with the core developers about the necessary API changes.

Suggestions on [6] about cross-validation generators having an option such that certain labels always to the training set, can be implemented.

Improvements in existing algorithms

The current implementation in **LabelPropagation** encodes unlabelled data with a -1 which is then mixed together with labelled data. If cross-validation is used, it splits the whole dataset which includes unlabelled data too. This causes problems listed in the section above. Apart from fixing this, documentation for the algorithms currently present in the semi_supervised module can be improved.

Implementing a Meta Estimator for semi-supervised learning

The iterative self-training method will be used to implement a meta-estimator, **SelfTrain** (temporary name) which can change any estimator into a semi-supervised one. [7]

The classical iterative self-training algorithm [11] is as follows:

Let **X** be the set of labeled data, **U** be the set of unlabeled data.

Repeat till convergence:

- Train a classifier **h** with training data **X**
- Classify data in **U** with **h**
- Find a subset **U*** of **U** with the most confident scores
- **X + U* -> X**
- **U - U* -> U**

This meta-estimator will be available in the semi_supervised module and have the same API as other meta-estimators such as the AdaBoostClassifier and BaggingClassifier.

Basic public methods of **SelfTrain** will include:

- **init**(estimator = est, params = params): To initialize the meta-estimator with the given estimator and other parameters.
- **fit**(X, U): To fit the estimator on small set of labeled data X and large set of unlabeled data U.

Implementation of new algorithms for semi-supervised learning

- **Transductive SVMs**: Formally, since the objective function of TSVM is not convex, it makes the functions difficult to optimize directly. I may have to implement this from scratch using either [2] which describes the application of convex-concave programming to this problem, where they optimize the objective function directly or [3] which uses an approximation. The authors appear to have had some success with [3]. [5]

- **Naive Bayes using Expectation Maximization**: I will work on integrating an existing PR[1] which implements this algorithms, into the semi_supervised module. A paper on this is also available[4].

- **Manifold Regularization:** [8] [9]

- **Recursive Neural Tensor Network:** [10]

More algorithms will be added upon discussion with the community..

[1] https://github.com/scikit-learn/scikit-learn/pull/430

[2] http://www.jmlr.org/papers/volume7/collobert06a/collobert06a.pdf

[3] http://www.gatsby.ucl.ac.uk/aistats/fullpapers/198.pdf

[4] http://www.kamalnigam.com/papers/emcat-mlj99.pdf

[5] http://www.cs.columbia.edu/~dplewis/candidacy/joachims99transductive.pdf

[6] https://github.com/scikit-learn/scikit-learn/issues/2593

[7] https://github.com/scikit-learn/scikit-learn/issues/1243

[8] http://www.jmlr.org/papers/volume7/belkin06a/belkin06a.pdf

[9] http://people.cs.uchicago.edu/~niyogi/papersps/BNMLJ.pdf

[10] http://nlp.stanford.edu/~socherr/EMNLP2013_RNTN.pdf

[11] http://faculty.washington.edu/f**xia**/**courses**/LING572/**self-training**.ppt

Timeline:

* Pre GSoC (Today to April 27th)

I will work on designing a plan to fix the `cross_validation` module for `semi_supervised` keeping in mind that it plays well with rest of scikit-learn. I will continue fixing as many issues as possible to get more familiar with the codebase.

Milestone: Getting more familiar with the codebase

* Community Bonding Period, Week 1 (April 27th to May 31st)

I will read about about the implementations of the algorithms highlighted in this proposal and start fixing the `cross_validation` module.

Milestone: Complete most of the work on cross-validation due to end term exams in Week 2, 3

* Week 2, 3 (June 1st to June 14th)

I have my university end term exams during this period. My contribution rate may be slow during this period. I will try to fix any issues which arise with the improved `cross_validation` module and write tests for it.

Milestone: Make the work on cross_validation mergeable for the mid-term evaluation

* Week 4 (June 15th to June 21st)

The priority would be to make the work mergeable for mid-term evaluation. I will further study the implementation details of the proposed algorithms.

Milestone: Make the `cross_validation` work mergeable

* Week 5, 6 (June 22nd to July 5th)

Milestone: Implement the self-learning meta-estimator

* Week 7, 8, 9 (July 6th to July 19th)

I will start implementing the proposed algorithms and improve existing ones.

Milestone: Implementation of algorithms while writing documentation

* Week 9 (July 20th to July 26th)

Milestone: Complete benchmarking

* Week 10 (July 27th to August 2nd)

Milestone: Complete tests

* Week 11 (August 3rd to August 9th)

Milestone: Complete documentation

* Week 12 (August 10th to August 16th)

Milestone: Complete examples

August 17th (Suggested ‘pencils down’ date.)

In this week, I will finalize the tests, documentation and examples.

August 21th - (Firm ‘pencils down’ date.)

Links to a patch/code sample (sorted by date):

* https://github.com/scikit-learn/scikit-learn/pull/4313 - SpectralClustering should be explicit about include_self (Merged)

* https://github.com/scikit-learn/scikit-learn/pull/4314 - Make clear that RBFSampler implements a variant of Random Kitchen Sinks (Merged)

* https://github.com/scikit-learn/scikit-learn/pull/4350 - Running tests should not print anything on stdout / stderr or warnings (Merged)

* https://github.com/scikit-learn/scikit-learn/pull/4356 - DictVectorizer.restrict docstring unclear (Open)

* https://github.com/scikit-learn/scikit-learn/pull/4377 - LinearSVC(intercept_scaling=0) breaks (Merged)

* https://github.com/scikit-learn/scikit-learn/pull/4389 - Move newton_cg test out of optimize (Merged)

* https://github.com/scikit-learn/scikit-learn/pull/4416 - SVC (and SVR) docstring not informative for kernel=”precomputed” (Merged)

* https://github.com/scikit-learn/scikit-learn/pull/4421 - renaming LDA and QDA to LinearDiscriminantAnalysis and QuadraticDiscriminantAnalysis (Open)

* https://github.com/scikit-learn/scikit-learn/pull/4423 - Raising an error when n_clusters <= 0 in AgglomerativeClustering (Open)

* https://github.com/scikit-learn/scikit-learn/pull/4431 - Deprecates load_lfw_pairs and load_lfw_people (Open)

Additional Information

* Apart from my two weeks of end term exams, I have no other commitments during the GSoC period. I will try to cover for these two weeks by starting early in the community bonding period. If by any chance, by the end of GSoC, my work does not get merged, I would work towards merging it beyond the summer.

* I have the done the following projects which are related to machine learning:

Person of Interest identifier: https://github.com/vortex-ape/POI-Identifier
Movie Recommender System: https://github.com/vortex-ape/Movie-Recommender-System

* [Curriculum Vitae](http://vinayakmehta.me/files/vinayakmehta_cv.pdf)

References

[1] Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. Yarowsky, 1995
http://delivery.acm.org/10.1145/990000/981684/p189-yarowsky.pdf?ip=59.177.143.70&id=981684&acc=OPEN&key=4D4702B0C3E38B35.4D4702B0C3E38B35.4D4702B0C3E38B35.6D218144511F3437&CFID=493009073&CFTOKEN=52184095&__acm__=1427455897_9d340b91a9755be6c6385a1297d114da

[2] Learning subjective nouns using extraction pattern bootstrapping, Riloff et al CoNLL-2003
http://www.cs.utah.edu/~riloff/pdfs/conll03.pdf

[3] Large Scale Transductive SVMs
http://www.jmlr.org/papers/volume7/collobert06a/collobert06a.pdf

[4] Meta-estimator for semi-supervised learning
https://github.com/scikit-learn/scikit-learn/issues/1243

[5] cross-validation generators broken for semi-supervised learning
https://github.com/scikit-learn/scikit-learn/issues/2593

[6] Using Cross Validation on semi-supervised classifiers
https://github.com/scikit-learn/scikit-learn/issues/3688