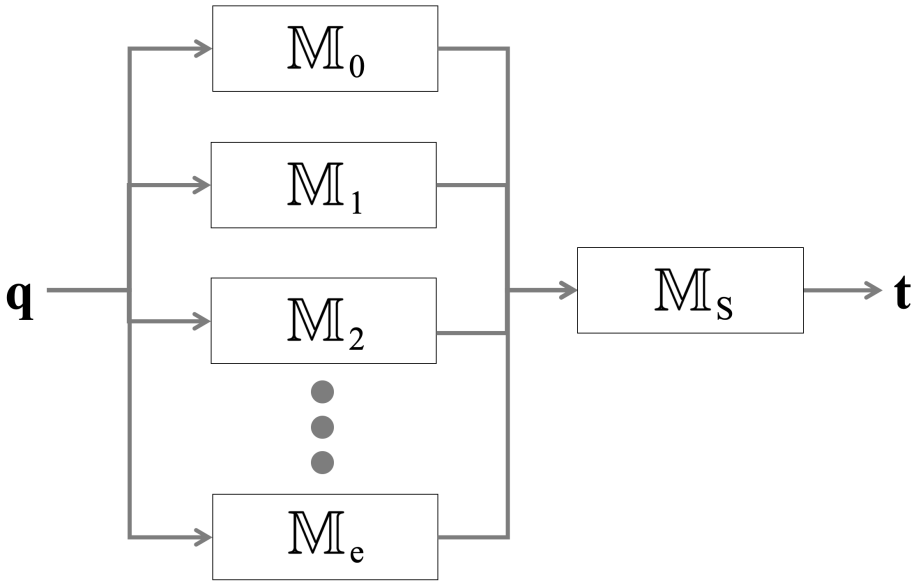


COMP47590

Advanced Machine Learning

Assignment 1: Building Stacked Ensembles

Introduction

Stacking is an interesting approach to building ensemble models that is particularly effective when heterogenous mixes of base classifiers is used. While less commonly used than its bagging and boosting counterparts, the stacked ensemble idea has been around for as long as ensemble models have been popular in machine learning - for example (Wolpert, 1992). In stacked ensembles a set of base classifiers are trained, and their outputs are combined into a training set for the stack layer classifier. One of the main challenges in building effective stacked ensembles is designing the process to generate the data to train the models at the stack layer.

The task in this assignment is to implement a selection of stacked ensemble approaches. Although you can use scikit-learn base estimator implementations (e.g. decision trees, logistic regression, or support vector machine models) you must implement the stacked ensemble models yourself.

Tasks

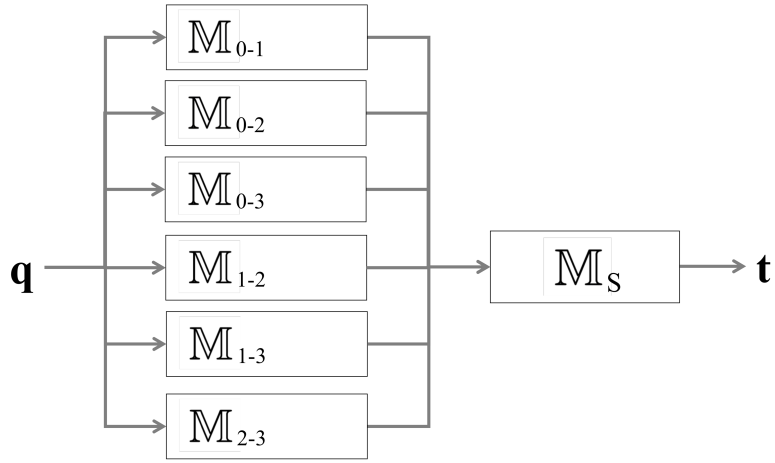
Perform the following tasks:

0. For all of the tasks in this assignment use the MNIST Fashion[1] dataset.
1. Modify the StackedEnsembleClassifier implementation provided to create **StackedEnsembleClassifierHoldOut** so that it uses a hold-out test set to generate the stack training dataset.
2. Modify the StackedEnsembleClassifier implementation provided to create **StackedEnsembleClassifierKFold** so that it uses a k-fold cross validation approach to generate the stack training dataset (this essentially implements the SuperLearner model described by van der Laan et al (2007)).
3. Compare the performance of the StackedEnsembleClassifier, StackedEnsembleClassifierHoldOut, and StackedEnsembleClassifierkFold implementations.
    - Design and use an appropriate evaluation strategy. Think about the most appropriate performance measure for the evaluation.
    - For all models use heterogenous ensembles with 24 base classifiers using a mixture of decision trees, support vector machines and logistic regression models.
    - Try out different model types at the stack layer (e.g. logistic regression models or decision trees).
4. Compare the performance of the three stacked ensemble model algorithms to that of a single decision tree and an ensemble based on bagging.

[1] The MNIST Fashion dataset is dataset available from Kaggle at https://www.kaggle.com/zalando-research/fashionmnist or from the COMP47590 moodle site.

- Tune the performance of these simple models using a grid search.

5. We can divide ensemble training algorithms into those that attempt to build ensembles of *generalists* and those that build ensembles of *specialists*. One way to build an ensemble of specialists is to build an ensemble of one-vs-one models each of which only considers two of the classes from the overall dataset. The base layer in this model will include $\binom{L}{2}$ models, where L is the number of classes in the classification problem. So, for example, if a problem has four classes, 0 – 3, it will have six binary base models that distinguish between classes 0-1, 0-2, 0-3, 1-2, 1-3, and 2-3. The outputs of these models can be combined using a stacked layer model.



   Each of these binary models should be trained on only instances from an overall training dataset that have target levels relevant to that binary model. Implement a one-vs-one stacked ensemble algorithm in a class called **StackedEnsembleClassifierOneVsOne** (this implementation is a large step towards implementing the Troika stacked ensemble algorithm, described by Menahem et al (2009)).

6. Compare the performance of the StackedEnsembleClassifierOneVsOne model to the other approaches explored.

7. Reflect on the performance of the different models evaluated in this assignment. Consider model performance as well as computational requirements and model complexity. For this, write a commentary of no more than 300 words.

Notes

The following notes may be useful:

- **Can I Use Scikit-Learn and Other Python Packages?** One of the goals of this assignment is to gain experience writing original machine learning algorithms, rather than simply using existing packages. We recognise, however, that very good implementations of machine learning algorithms exist in packages like scikit-learn and re-implementing things needlessly is often of limited value. So, for this assignment we seek to strike a balance. You must write your own implementation for StackedEnsembleClassifierHoldOut, StackedEnsembleClassifierkFold, and StackedEnsembleClassifierOneVsOne but can use scikit-learn implementations for the base estimators and stack layer estimators. You should also use scikit-learn functions for performing tasks like cross validation and grid searches. Similarly, all scikit-learn models are built on top of **numpy** and it provides many useful utility functions. If in doubt about the use of a package please just ask.
- **What Hardware Should I Use?** None of the datasets or models to be used in this assignment are so big that they shouldn't work fine on your laptop. If you are having problems, however, you should consider using Google Colab[2], a great online platform for machine learning.
- **It's Still Taking Forever!** While the datasets used in this assignment are not enormous, they are big enough to cause problems once we start performing cross validations and grid searches. If you find things are taking too long feel free to down-sample the dataset. Submissions will not be penalised for using down-sampled datasets. Ensure, however, that the target feature levels remain stratified.
- **Teams.** You can complete this assignment as an individual or as a team of two. Submissions will be graded in the same way whether submitted by an individual or by a group. All members of a group will receive the same grade.

[2] Google colab https://colab.research.google.com

Submission

The key submission details for the assignment are as follows:

- **Submission date:** Thursday 8th March 2018 before 23:59.
- **Submission method:** Submissions should be made through the module Moodle site.
- **Submission format:** Submissions should compose a zip file containing the following:
    - a completed version of the template Jupyter notebook provided in .ipynb format (the Python notebook and code should include adequate comments to understand your implementation);
    - a html export of your Jupyter notebook after execution that contains all output;
    - any other files required to execute your code.
- **Group submission:** For group submissions only one member of the group should submit.
- **Late submissions:** Late submissions will be penalised at 5% penalty per day.

Marking

Marking of tasks will be based on the following weighting.

- Task 1 20% Implement the **StackedEnsembleClassifierHoldOut** classifier.
- Task 2 20% Implement the **StackedEnsembleClassifierKFold** classifier
- Task 3 10% Compare the performance of the stacked ensemble implementations
- Task 4 10% Compare ensembles with basic models
- Task 5 30% Implement the **StackedEnsembleOneVsOne** classifier
- Task 6 5% Compare performance of all models.
- Task 6 5% Reflect on the experiment.

References

van der Laan, M., Polley, E. & Hubbard, A. “Super Learner”, *Statistical Applications in Genetics and Molecular Biology*, 6(1), 2007.
https://pdfs.semanticscholar.org/19e9/c732082706f39d2ba12845851309714db135.pdf

Wolpert, D.H., “Stacked generalisation”, *Neural Networks*, 5, pp 241-259, 1992.
http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.56.1533

Menahem, E., L. Rokach, & Y. Elovici, “Troika – An improved stacking schema for classification tasks”, *Information Sciences*, 179 (24), pp 4097-4122, 2009.
https://www.sciencedirect.com/science/article/pii/S0020025509003600