Automated Machine Learning for Soft Voting in an Ensemble of Tree-based Classifiers

Jungtaek Kim and Seungjin Choi

Department of Computer Science and Engineering, Pohang University of Science and Technology

Introduction

- Attempt to find the optimal model without human intervention using automated machine learning (AutoML).
- Usually include algorithm selection and hyperparameter optimization.
- Note that $\mathcal{A}$ and Λ are algorithm and hyperparameter spaces. Given a training dataset $\mathcal{D}_{\text{train}}$ and a validation dataset $\mathcal{D}_{\text{val}}$, the optimal $\mathbf{A}^* \in \mathcal{A}$ and $\boldsymbol{\lambda}^* \in \Lambda$, found by AutoML system:

$$(\mathbf{A}^*, \boldsymbol{\lambda}^*) = \text{AutoML}(\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{val}}, \mathcal{A}, \Lambda).$$

Background

- Soft Majority Voting
  - An ensemble method to construct a classifier using a majority vote of k base classifiers.
  - Contributed by each base classifier of soft *voting classifier* through class probabilities with given weights for all classes.
  - Class assignment of soft voting classifier for instance i:

$$c_i = \arg\max \sum_{j=1}^{k} w_j \mathbf{p}_i^{(j)}$$

  where n is the number of instances, $w_j \in \mathbb{R} \geq 0$ and $\mathbf{p}_i^{(j)}$ are weight and class probability vector of base classifier j.
- Bayesian Optimization
  - A method to find global optimum for black-box function.
  - Improve the current best solution as iterating the steps: (i) modeling a surrogate function and (ii) acquiring a next point that has maximum value of acquisition function.
  - Optimize an acquisition function instead of an original target function.
  - Gaussian process upper confidence bound (GP-UCB):

$$a_{\text{UCB}}(\mathbf{x}) = -\mu(\mathbf{x}) + \kappa\sigma(\mathbf{x})$$

  where $\mu(\mathbf{x})$ and $\sigma(\mathbf{x})$ are posterior mean and posterior standard deviation functions.

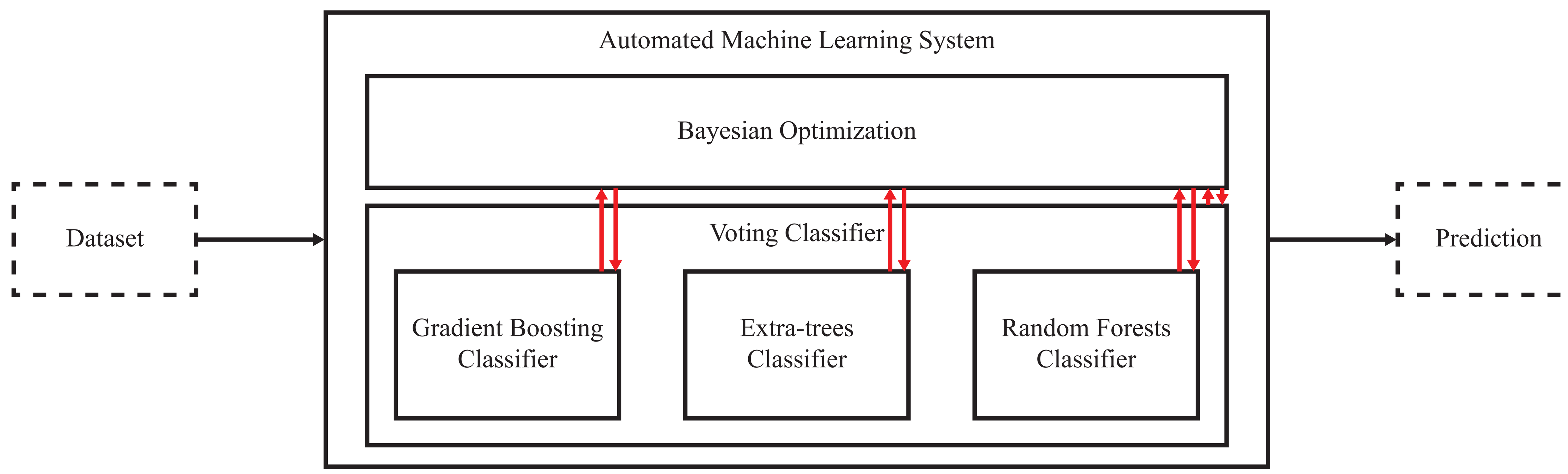

Figure 1: Our automated machine learning system, *mlg.postech*.

Table 1: AutoML Challenge 2018 result. Our system, *mlg.postech* took second place in the challenge.

Place	Team	Set 1	Set 2	Set 3	Set 4	Set 5	Average
1	aad_freiburg	0.5533 (3)	0.2839 (4)	0.3932 (1)	0.2635 (1)	0.6766 (5)	2.8
2	**mlg.postech**	**0.5418 (5)**	**0.2894 (2)**	**0.3665 (2)**	**0.2005 (9)**	**0.6922 (1)**	**3.8**
	wlWangl	0.5655 (2)	0.4851 (1)	0.2829 (5)	-0.0886 (16)	0.6840 (3)	5.4
3	thanhdng	0.5131 (6)	0.2256 (8)	0.2605 (7)	0.2603 (2)	0.6777 (4)	5.4
	Malik	0.5085 (7)	0.2297 (7)	0.2670 (6)	0.2413 (5)	0.6853 (2)	5.4

Our AutoML System, *mlg.postech*

- Written in `Python`.
- Use `scikit-learn` and our own Bayesian optimization package.
- Split training dataset to training (0.6) and validation (0.4) sets for Bayesian optimization.
- Optimize six hyperparameters:
  1. extra-trees classifier weight/gradient boosting classifier weight for voting classifier,
  2. random forests classifier weight/gradient boosting classifier weight for voting classifier,
  3. the number of estimators for gradient boosting classifier,
  4. the number of estimators for extra-trees classifier,
  5. the number of estimators for random forests classifier,
  6. maximum depth of gradient boosting classifier.
- Use GP-UCB.
- Search hyperparameters in the pre-defined ranges, determined by dataset size, dimension, and order.
- Focus on tree-based classifiers, because they have strong generalization capacity.
- Construct a soft voting classifier with three tree-based classifiers: gradient boosting classifier, extra-trees classifier, and random forests classifier.
- Widely use Bayesian optimization in hyperparameter optimization and AutoML.
- Iteratively optimize the hyperparameters of voting classifier and tree-based classifiers using Bayesian optimization for the given time budget.

Table 2: Datasets of feedback phase. Time budget shows in seconds.

Dataset	ada	arcene	gina	guillermo	rl
Train. #	4,147	100	3,153	20,000	31,406
Valid. #	415	100	315	5,000	24,803
Test #	41,471	700	31,532	5,000	24,803
Feature #	48	10,000	970	4,296	22
Chrono.	False	False	False	False	True
Budget	600	600	600	1,200	1,200

Details of AutoML Challenge 2018

- Held as the 22nd Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD-2018) data competition.
- Two phases: feedback phase and AutoML challenge phase.
- In the feedback phase, provide five datasets for binary classification.
- Given training/validation/test datasets, after submitting a code or prediction file, validation measure is posted in the leaderboard.
- In the AutoML challenge phase, determine challenge winners, comparing a normalized area under the ROC curve (AUC) metric for blind datasets:

$$\text{Normalized AUC} = 2 \cdot \text{AUC} - 1.$$

- Executed in the Ubuntu machine which has (i) 2 cores, (ii) 8GB memory, and (iii) 40GB SSD.

Conclusion

- Take second place in AutoML Challenge 2018.
- Effective to optimize the hyperparameters with Bayesian optimization.
- Validate our system can train and test blind datasets without human intervention.

Open Repository & Related Links

- Our system repository: https://github.com/jungtaekkim/automl-challenge-2018
- Challenge website: https://competitions.codalab.org/competitions/17767

Contact Information

- Homepage: http://mlg.postech.ac.kr/~jtkim
- GitHub: https://github.com/jungtaekkim
- Email: jtkim@postech.ac.kr