

AutoML Framework for Labor Potential Modeling

Vladislav Kovalevsky[1](✉), Elena Stankova[2], Nataly Zhukova[1], Oksana Ogiy[3], and Alexander Tristanov[3]

[1] St. Petersburg Federal Research Center of the Russian Academy of Sciences, St. Petersburg, Russia
darkeol@gmail.com
[2] St. Petersburg State University, St. Petersburg, Russia
[3] Kaliningrad State Technical University, Kaliningrad, Russia

Abstract. Labor potential estimation is a complex analysis task of a large data sets of digital profiles of workers. Digital profile is a structure that contains heterogeneous data and has high complexity. The attribute space of digital profiles datasets has a large dimension. Individual features are distinguished by a large variability of types and include qualitative and quantitative characteristics that can be both continuous or discrete. This fact significantly complicates the use of traditional statistical methods and hinder manual selection of machine learning algorithms for labor potential estimation. Automated machine learning (AutoML) can help to deal with this problem. The results of AutoML can significantly depend on the used AutoML library. AutoML libraries differ in the set of algorithms among which they select the best one, in the constrains that are used to stop the search of the optimal solution, in the techniques used for hyperparameters optimization and in metrics that are calculated for the obtained results. In this work we develop an AutoML framework that unites several commonly used AutoML libraries and enables use of common constrains and metrics. Processing dataset about employees' performance using the developed framework we show the applicability of the AutoML for the task of labor potential estimation.

Keywords: AutoML · Machine learning · Hyperparameters optimization · Labor potential assessment

1 Introduction

1.1 Human Capital

One of the important sections of modern economic science is the theory of human capital, which explores the dependence of the income of an individual, enterprise and society as a whole on the abilities of people, their knowledge and skills [1]. The labor potential and its estimation is vital for enterprises to be competitive [2]. However, it should be noted that nowadays there is no single methodological approach to the definition and, especially, the measurement of labor potential.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
O. Gervasi et al. (Eds.): ICCSA 2023, LNCS 13957, pp. 87–98, 2023.
https://doi.org/10.1007/978-3-031-36808-0_6

1.2 Labor Potential Estimation

Labor potential estimation is a complex analysis task [3] of a large data sets of digital profiles of workers. Digital profile is a structure that contains heterogeneous data type and has high complexity. The attribute space of the digital profile datasets has a large dimension and covers the attributes of four groups of parameters: socio-demographic, spiritual and moral, professional competence and socio-psychological.

The main characteristics of an individual that determine his labor potential are:

- abilities determined by such psychophysiological characteristics as gender, age, functional state of the body, operator working capacity, adaptive potential of health, current mental state, personality traits, character accentuations;
- competence - knowledge, skills, level of education/qualification, total length of service, competency proactivity;
- labor behavior determined by motivation and value orientations.

Individual features are distinguished by a large variability of types and include qualitative and quantitative characteristics that can be both continuous or discrete. This fact significantly complicates the use of traditional methods of statistical analysis.

A prominent approach to labor potential estimation assumes usage of artificial neural networks [4, 5]. However, these methods are complex and require significant efforts to define the architecture and the parameters of the networks. Another approach to labor potential estimation is based on using machine learning (ML) methods. To solve the problem of labor potential estimation various ML methods can be used that capture the dependency between input characteristics and output value based on the analyses of large amount of existing data, but the high complexity of the problem and the characteristics of the processed data, hinder manual selection of machine learning algorithm and tuning its hyperparameters. This leads to the need to automate the process of finding the best algorithm.

An approach that is not limited by only one method but instead searches through many different methods in order to find the one that process the given data in the best way is automated machine learning (AutoML). In the paper the problem of using AutoML for labor potential estimation is considered.

Contribution of this research is the following: 1) An AutoML framework was developed that enables search for appropriate model through several AutoML libraries instead of a single one with support of common metrics and constrains, 2) The efficiency of AutoML approach in solving a problem of labor potential estimation, that is a poorly formalized problem, is shown through the use of the developed framework.

The rest of the paper is organized as follows: In Sect. 2 we describe the problem of labor potential estimation, how AutoML can be used to deal with this problem, and which difficulties can arise while using AutoML. Section 3 gives general overview of the existing AutoML libraries and their comparison. In Sect. 4 we describe the architecture of the proposed AutoML framework that unites several AutoML libraries, considerations by which the libraries were chosen and common metrics that are used by all libraries in evaluation of classification results. In Sect. 5 we show the applicability of the developed framework for the task of labor potential estimation by performing experiment on a

dataset that contains information about workers, their characteristics and performance. The paper concludes with a brief summary and discussion about the future work.

2 Problem Statement

The task of labor potential estimation can be seen from different points of view. In one cases it can be seen as a classification problem where an individual should be related to the right category of labor potential based on the various attributes that the individual possesses. In another cases labor potential can be expressed not as category, but in numeric form, then the task will be to predict the value of the labor potential given the input attributes. Another approach can be in dividing individuals into clusters based on their potential rather than finding information about specific individual.

The field of machine learning (ML) include many algorithms that build models using data. The algorithms that can be used for labor potential estimations fall in the following categories depending on the specific task that needs to be solved:

- classification - determine class of the object from predefined set of classes;
- regression - predict value of continuous parameters;
- clustering - separate objects into several classes.

Every ML algorithm has its own set of hyperparameters – values that define the work of the algorithm, and these hyperparameters also need to be tuned. Thus the task is - given some dataset find the best possible combination of learning algorithms and their hyperparameters. This task is often referred as "CASH problem" - Combined Algorithm Selection and Hyperparameter Optimization [6, 7].

AutoML methods are suitable to solve the tasks of choosing the best algorithm for the labor potential estimation. By now several AutoML libraries have been developed and thus a problem of choosing the most appropriate one arises. One way to tackle this problem is to unite several AutoML solutions in a common framework. To that end the solutions (AutoML libraries) should be examined in order to find limitations that can prevent to unite the libraries, and also common constrains and metrics that can be used across all solutions included in the framework. Thus the framework should enable following capabilities:

- unite several AutoML libraries and enable their parallel run;
- supply data preprocessing that is needed for each library;
- enable usage of common search constrains for all libraries;
- use common metrics for comparing the results of different libraries.

3 Existing AutoML Solutions

Currently there are several AutoML libraries that use different approaches in order to find best combination of machine learning algorithm and its hyperparameters. These solutions share common approach of using underlying machine learning library to select algorithm from. Among the known ones are Auto-WEKA [8], Hyperopt-sklearn [9], Auto-Sklearn [10], TPOT [11], H2O [12] and Auto-Keras [13]. Hyperopt-sklearn, Auto-Sklearn and TPOT search for the suitable algorithm in the Python library Scikit-Learn [14] whereas others use their own implementation of machine learning algorithms. The comparison of the characteristics of the systems is shown in Table 1.

Table 1. Comparison of AutoML solutions.

Solution	Library	Language	OS	GUI
Auto-WEKA	WEKA	Java	Cross-platform	Yes
Hyperopt-sklearn	Scikit-Learn	Python	Cross-platform	No
Auto-sklearn	Scikit-Learn	Python	Linux	No
TPOT	Scikit-Learn	Python	Cross-platform	No
H2O	H2O	Java/Python	Cross-platform	No
Auto-Keras	TensorFlow	Python	Linux	No

3.1 Auto-WEKA

The Auto-WEKA is one of the first AutoML solutions and is built as a plugin to the WEKA program [15] that is written in Java language. Auto-WEKA searches through machine learning algorithms that are presented in WEKA in order to find the most appropriate combination of machine learning algorithm and values of its hyperparameters that can most efficiently process a given dataset.

The search for the best solution is done by use of the Bayesian optimization, a sequential design strategy for global optimization of black-box expensive-to-evaluate functions. The SMAC (Sequential Model-Based Algorithm Configuration) algorithm based on Bayesian optimization that Auto-WEKA uses for the search is a sophisticated instantiation of the general SMBO (Sequential Model-Based Optimization) framework and works in following steps:

1. A probabilistic model of the objective function is built.
2. Hyperparameters are selected that suite best the probabilistic model.
3. Fitted hyperparameters are applied to the objective function.
4. The probabilistic model is recalculated.

3.2 Hyperopt-Sklearn

The popularity of Python language for the tasks of data mining gave rise to the machine learning libraries and AutoML solutions written in this language. One of the first AutoML solutions written in Python is Hyperopt-sklearn. It was also designed to solve the problem of automatic selection of the best combination of machine learning algorithm and its hyperparameters that is most efficient for processing any given dataset.

Hyperopt-sklearn searches through machine learning algorithms that present in Scikit-learn (sklearn) library but doesn't implement its own method for hyperparameters optimization and uses for this task Hyperopt library [16]. Hyperopt searches through the hyperparameter space in order to find the best values of hyperparameters. This search can be done using one of the following search and optimization algorithms which must be specified at startup:

- Random search (rand)
- Simulated annealing (anneal)
- Tree of Parzen Estimators (tpe)
- Adaptive TPE (atpe).

3.3 Auto-Sklearn

The Auto-sklearn is another more sophisticated AutoML solution that is also based on Bayesian optimization and searches through algorithms of Scikit-learn library. Auto-sklearn introduces two improvements to the search that previous solutions didn't have: meta-learning and ensemble. Meta-learning assumes searching for similar, previously encountered datasets to use the same search initialization as they had. The similarity of datasets is determined by comparing their meta-properties. Auto-sklearn uses 140 datasets from the OpenML [17], an open source repository, as source for meta-learning. Ensemble means usage of several algorithms that have shown good results in the final training, instead of a single algorithm, albeit the most effective one. The solution based on algorithms combination is more reliable and less prone to overfitting. Auto-sklearn considers configuration space consisting from base classifiers and excluded meta-models and ensembles that are themselves parameterized by one or more base classifiers. Such ensembles that are used by Auto-WEKA give it space of 786 hyperparameters whereas Auto-Sklearn features 110 hyperparameters. Auto-sklearn uses ensemble at the last step and not within search process that is more efficient.

3.4 TPOT

The TPOT (Tree-Based Pipeline Optimization Tool) software package is a Python library for automatically building a suitable machine learning data processing pipeline that is most efficient for processing a given data set and for automatically tuning the hyperparameters of the selected machine learning algorithm. This package is based on the ideas of genetic programming where many different solutions are generated and then checked in order to retain the best, modify them, check again and so on.

The result of TPOT work is a data processing pipeline - a combination of algorithms for working with a selected data set. The parts of the pipeline are:

1. Feature Preprocessing Operators
2. Feature Selection Operators
3. Algorithms for classification and regression (Supervised Classification/Regression Operators).

The algorithms from which the data processing pipeline is compiled are taken from the Scikit-learn library (sklearn).

3.5 H2O

H2O is an open source, in-memory, distributed, fast, and scalable machine learning and predictive analytics platform that allows build machine learning models on big data and provides easy productionalization of those models in an enterprise environment. H2O's core code is written in Java. Inside H2O, a Distributed Key/Value storage is used to access and reference data, models, objects, etc., across all nodes and machines. The algorithms are implemented on top of H2O's distributed Map/Reduce framework. This platform implements a number of supervised and unsupervised algorithms like Deep Learning, Tree Ensembles, and GLRM, and also has AutoML functionality that allows

automatic search through the algorithms contained in the platform. API to Python allows use H2O from code written on Python.

3.6 Auto-Keras

Auto-Keras is a system that searches deep learning algorithms. This system works with algorithms from the Keras library, but since this library itself is an add-on and interface to the TensorFlow [18] library, in fact AutoKeras searches in this particular library.

Auto-Keras architecture consists of four parts: 1) search engine, 2) graph, 3) model trainer, 4) model repository. A search engine is a neural architecture search algorithm module containing Bayesian optimizers and Gaussian processes. The model trainer trains the resulting neural network using the training data. A graph is a module that processes computational graphs of neural networks managed by a search engine to perform network morphism operations. Model storage is a pool of trained models.

4 AutoML Framework

In order to process the data on various AutoML libraries with same constrains and then to compare their results it is worth to unite them in a common framework. To reach this goal the libraries have to be compatible. For this reason, we exclude Auto-WEKA from the further consideration, because it is written in Java which makes it hard to unite with other solutions written in Python.

Another important feature that should be considered are the constrains that each solution can use to limit the search of the algorithm and ability to divide the search into several parallel processes. Among common constraints are: time limits, memory limits. Comparison of the constrains that AutoML libraries support is shown in Table 2. It could be seen from the comparison that AutoKeras library doesn't support time constraints that other systems do, so it will be also excluded from the united framework.

Different AutoML libraries that use Scikit-Learn as a source of algorithms also cannot run together because of their dependency on different versions of libraries, in particular, Auto-sklearn cannot run together neither with TPOT nor with Hyperopt-sklearn. Therefore, two separate parts were developed, one unites Auto-sklearn and H2O and another works with TPOT, Hyperopt-sklearn and H2O.

Based on the above mentioned considerations a common framework for AutoML libraries was developed [19] in Python language. The framework takes as input information about dataset, ratio in which the data has to be splitted into train and test sets and time limit for each possible solution to run and then starts each AutoML solution with given data and limits in separate thread. The general architecture of the framework is shown in Fig. 1.

When all solutions finish their work the results are accumulated in a table. The results of each run contain various metrics which can be used to compare different solutions. These metrics include accuracy, precision, recall, f-measure and AUC.

Accuracy is a metric for assessing the quality of the classification, which shows how many objects were classified correctly relative to the total number of objects. This is the

Table 2. AutoML search constrains.

AutoML library	Time limit	Memory limit	Parallel runs
AutoWEKA	Yes	Yes	Yes
Hyperopt-sklearn	Yes	No	No
Auto-sklearn	Yes	Yes	Yes
TPOT	Yes	No	Yes
H2O	Yes	No	No
AutoKeras	No	No	Yes

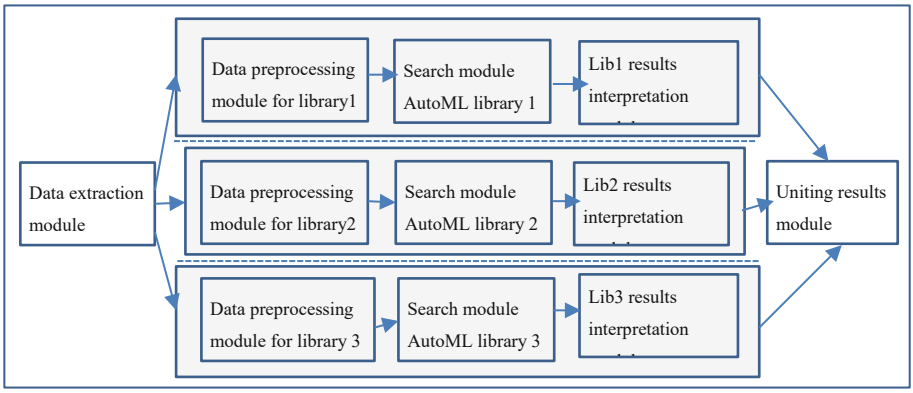

Fig. 1. Architecture of the framework

ratio of correctly classified objects to the total number of objects in the sample and it is calculated as follow:

$$Accuracy = \frac{tp + tn}{tp + fp + tn + fn} \tag{1}$$

where tp - true positive, tn - true negative, fp - false positive, - fn - false negative.

Precision is a metric that indicates how many of the objects classified as positive by the model are actually positive. This is the ratio of true positive objects (TP) to the total number of objects classified as positive (TP + FP).

$$precision = \frac{tp}{tp + fp} \tag{2}$$

Recall is a metric that shows how many positive objects were correctly detected by the model relative to the total number of true positive objects in the sample. This is the ratio of true positive objects (TP) to the total number of objects that are actually positive (TP + FN).

$$recall = \frac{tp}{tp + fn} \tag{3}$$

F-measure is a metric that combines precision and recall into one metric. It is calculated as a harmonic mean of precision and recall and shows how well the model classifies positive objects while minimizing false positives and false negatives.

$$F(\beta) = \left(1 + \beta^2\right) \frac{precision \cdot recall}{\beta^2 \cdot precision + recall} \tag{4}$$

AUC (area under the curve) is a metric that shows how well the model distinguishes between positive and negative classes. The ROC curve is a graph showing the relationship between the recall and false positive rate of a model as the threshold changes. AUC indicates how well the model distinguishes between positive and negative classes, and the higher the AUC, the better is the model.

5 AutoML for Labor Potential Estimation

5.1 Datasets of Labor Potential Information

As every other statistical method AutoML needs data to be trained on. In our case data about workers, their characteristics and performance is needed. The collection of such data is beyond the scope of this work. Open source repositories include a number of already prepared datasets on which the approach could be tested. For example, Kaggle [20] repository contains Employee Performance Prediction [21], Human Resources Data Set [22], IBM HR Analytics Employee Attrition & Performance [23] and Employee Performance Analysis INX Future Inc [24]. For our purposes we decided to use the last one because it contains various information about workers and their performance. The selected dataset has records about 1200 workers, each described by 26 features.

The dataset contains following information: Age, Gender, EducationBackground, MaritalStatus, EmpDepartment, EmpJobRole, BusinessTravelFrequency, DistanceFromHome, EmpEducationLevel, EmpEnvironmentSatisfaction, EmpHourlyRate, EmpJobInvolvement, EmpJobLevel, EmpJobSatisfaction, NumCompaniesWorked, OverTime, EmpLastSalaryHikePercent, EmpRelationshipSatisfaction, TotalWorkExperienceInYears, TrainingTimesLastYear, EmpWorkLifeBalance, ExperienceYearsAtThisCompany, ExperienceYearsInCurrentRole, YearsSinceLastPromotion, YearsWithCurrManager.

The target attribute PerformanceRating has numeric value that has three possible values: 2 – poor (16%), 3 – good (73%), 4 – excellent (11%).

5.2 Experiments

The developed framework enables use of common data and constraints by several libraries. The records from the dataset were divided into training and test set, the latter being 1/4 from the total data and all libraries used the same ratio of train and test data. Then the search was performed using different time limits, again common for different libraries: 5 min, 10 min, 15 min, 30 min and 60 min. The visualization of the dependency between accuracy and running time of the libraries included in the framework is shown in Fig. 2.

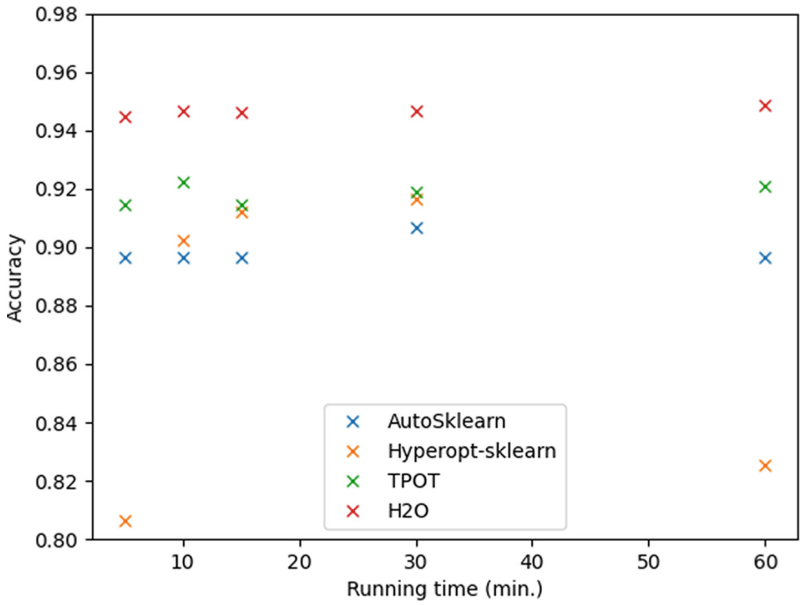

Fig. 2. Dependency of the accuracy from the running time

The more detailed results of the runs are given in Table 3.

GBM is standing for Gradient Boosting Machine and RF for Random Forest algorithms.

The performed experiments have shown high accuracy across all used libraries. The use of AutoML framework instead of a single library enabled to identify not only the best algorithm but also the most appropriate library from which the algorithm should be taken for the task of labor potential estimation. It was shown that in different time constrains different libraries can give better results. For example, for running time of 5 and 60 min Hyperopt-sklearn gives the worst results whereas for the time limit of 10, 15 and 30 min it outperforms Auto-sklearn and for limits of 15 and 30 min even almost reaches the accuracy of TPOT that always occupies the second place.

The experiments show that H2O provides better accuracy among other libraries, Hyperopt-sklearn more than other libraries depends on the time limits, and that increase in running time can also worsen the results, at least when the running time constraints are in the span of 1 h.

Table 3. Comparison of AutoML systems' suggestions and performance.

AutoML library	Time	Selected alg	Accuracy	Precision	Recall	F-score	AUC
AutoSklearn	5 min	adaboost	0,896	0,878	0,812	0,841	0,967
Hyperopt-sklearn	5 min	RF	0,806	0,878	0,605	0,623	0,660
TPOT	5 min	RF	0,914	0,894	0,850	0,869	0,950
H2O	5 min	GBM	0,944	0,970	0,866	0,910	–
AutoSklearn	10 min	adaboost	0,896	0,878	0,812	0,841	0,967
Hyperopt-sklearn	10 min	RF	0,902	0,847	0,874	0,859	0,951
TPOT	10 min	RF	0,922	0,911	0,852	0,878	0,960
H2O	10 min	GBM	0,946	0,927	0,928	0,927	–
AutoSklearn	15 min	adaboost	0,896	0,878	0,812	0,841	0,967
Hyperopt-sklearn	15 min	GBM	0,912	0,883	0,850	0,865	0,655
TPOT	15 min	RF	0,914	0,894	0,850	0,869	0,953
H2O	15 min	GBM	0,946	0,942	0,908	0,920	–
AutoSklearn	30 min	GBM	0,906	0,858	0,883	0,870	0,962
Hyperopt-sklearn	30 min	GBM	0,916	0,873	0,880	0,877	0,589
TPOT	30 min	RF	0,918	0,895	0,856	0,874	0,959
H2O	30 min	xgboost	0,946	0,936	0,914	0,924	–
AutoSklearn	60 min	adaboost	0,896	0,878	0,812	0,841	0,967
Hyperopt-sklearn	60 min	RF	0,825	0,772	0,653	0,697	0,701
TPOT	60 min	RF	0,921	0,907	0,851	0,876	0,926
H2O	60 min	GBM	0,948	0,940	0,911	0,925	–

6 Conclusion

In this work we develop an AutoML Framework that unite several commonly used AutoML libraries. To that end review and comparison of libraries were made that enabled selection of libraries that share common properties in means of metrics and constrains. The Framework is developed in Python language and consists of 4 AutoML libraries, it provides following capabilities:

- load data from local files or from openml.org website;
- input data preprocessing that is unique for each library and depends on data characteristics;
- common values of data split ratio and search time limits are shared across the used libraries;
- use of common metrics for results evaluation;
- parallel run of the libraries included in the framework.

The Framework was used to show the applicability of AutoML methods for solving the poorly formalized problem of labor potential estimation by performing search of the

best model with 5 different time limits: 5, 10, 15, 30, 60 min. The experiments have shown that H2O library outperforms other AutoML libraries in all runs, giving maximum accuracy of 0,948 when maximum amount of time was allocated. AutoSklearn gave in most cases the worst results with the best result being 0,906. Hyperopt-sklearn have shown heavy dependency on running time giving accuracy of 0,806 when minimum amount of time was allocated and outperforming AutoSklearn with accuracy of 0,916 when given more time. The use of the AutoML framework allowed identify the most suitable library for the given task and in different time constrains. The performed experiments on dataset about labor potential have shown that the developed framework successfully solves the task of labor potential estimation.

The limitation of the suggested approach is that only AutoML libraries that share common properties can be united under the umbrella of the common framework. The more different metrics and constrains the framework will support the less libraries will be possible to use. Another limitation is that only libraries that written in Python or supporting it can be used.

The future work includes overcoming some of the mentioned above limitations, taking in account not only time constraint but also other constrains such as memory limits and number of runs; use of early stopping methods to stop search early in case when threshold value is reached; and enabling choose of libraries to make search in, this could be particularly useful in case when only part of libraries share some common properties.

Acknowledgments. This work was supported by a state grant (project No. FFZF-2022-0006) and as part of the agreement on the provision of subsidies from the federal budget for financial support for the implementation of the state task for 2023 (registration number of the EGISU NIOKTR card 1122110800010-7 28/11/2022).

References

1. Patutina, E.S.: Main approaches to the interpretation of the concepts of “labor potential” and “human potential” as the basis for the quality of the labor force in the works of domestic and foreign scientists. In: Science Research Practice. Collection of Selected Articles Based on the Materials of the International Scientific Conference, pp. 215–220 (2020)
2. Human Resources Management: Concepts, Methodologies, Tools, and Applications, p. 1513. IGI Global, Pennsylvania (2012)
3. Page, S.E.: What sociologists should know about complexity. Ann. Rev. Sociol. **41**(1), 21–41 (2015). https://doi.org/10.1146/annurev-soc-073014-112230
4. Heravi, G., Eslamdoost, E.: Applying artificial neural networks for measuring and predicting construction-labor productivity. J. Constr. Eng. Manag. **141**(10) (2016)
5. Ogiy, O.G., Osipov, V.Yu., Tristanov, A.B., Zhukova, N.A.: The process of managing labor potential of the fishery complex as an object of modeling using artificial neural networks. In: AIP Conference Proceedings, vol. 2661, p. 030001 (2022)
6. Thornton, C., Hutter, F., Hoos, H.H., Leyton-Brown, K.: Auto-WEKA: combined selection and hyperparameter optimization of classification algorithms. In: 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013. ACM, New York (2013). https://doi.org/10.1145/2487575.2487629

7. Guo, X., van Stein, B., Bäck, T.: A new approach towards the combined algorithm selection and hyper-parameter optimization problem. In: 2019 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 2042–2049. IEEE, Xiamen (2019). https://doi.org/10.1109/ssci44817.2019.9003174
8. Auto-WEKA. https://www.cs.ubc.ca/labs/algorithms/Projects/autoweka/. Accessed 07 Apr 2023
9. Hyperopt-sklearn. https://hyperopt.github.io/hyperopt-sklearn/. Accessed 10 Apr 2023
10. Feurer, et al.: Auto-Sklearn 2.0: Hands-Free AutoML via Meta-Learning. arXiv (2020)
11. Olson, R.S., Bartley, N., Urbanowicz, R.J., Moore, J.H.: Evaluation of a tree-based pipeline optimization tool for automating data science. In: Proceedings of GECCO 2016, pp. 485–492 (2016)
12. H2O.AI - The fastest, most accurate AI Cloud Platform. https://h2o.ai. Accessed 10 Apr 2023
13. Jin, H., Chollet, F., Song, Q., Hu, X.: AutoKeras: an AutoML library for deep learning. J. Mach. Learn. Res. **6**, 1–6 (2023)
14. Scikit-Learn. Machine Learning in Python. https://scikit-learn.org/stable/. Accessed 07 Apr 2023
15. Weka 3: Machine Learning Software in Java. https://www.cs.waikato.ac.nz/ml/weka/. Accessed 07 Apr 2023
16. Hyperopt: Distributed Asynchronous Hyper-parameter Optimization. http://hyperopt.github.io/hyperopt/. Accessed 07 Apr 2023
17. OpenML. A worldwide machine learning lab. https://www.openml.org. Accessed 07 Apr 2023
18. TensorFlow. An end-to-end open source machine learning platform for everyone. https://www.tensorflow.org. Accessed 07 Apr 2023
19. AutoML Framework for AutoML libraries comparison. https://github.com/DarkEol/AutoML-Framework. Accessed 07 Apr 2023
20. Kaggle. https://www.kaggle.com. Accessed 07 Apr 2023
21. Employee Performance Prediction. https://www.kaggle.com/datasets/gauravduttakiit/employee-performance-prediction. Accessed 07 Apr 2023
22. Human Resources Data Set. https://www.kaggle.com/datasets/rhuebner/human-resources-data-set. Accessed 07 Apr 2023
23. IBM HR Analytics Employee Attrition & Performance. https://www.kaggle.com/datasets/pavansubhasht/ibm-hr-analytics-attrition-dataset. Accessed 07 Apr 2023
24. Employee Performance Analysis INX Future Inc. https://www.kaggle.com/datasets/eshwarganta/employee-performance-analysis-inx-future-inc/. Accessed 07 Apr 2023