Evaluation of AutoML Systems on OpenML Regression Tasks

Xinchen Yang[1], Jieshi Chen[2] and Artur Dubrawski[2]

Abstract— **The Automated Machine Learning (AutoML) System has powered domain experts with its efficient model discovery capabilities and helped address shortages of qualified data scientists. Popular AutoML Systems include the Auto^{n}ML developed by the Auton Lab at Carnegie Mellon University (CMU), Tree-based Pipeline Optimization Tool (TPOT), AutoGluon, auto-sklearn, etc. We hope to learn about how the performances of AutoML systems compared to each other and how factors such as dataset characteristics and algorithm selection can affect AutoML performances. Motivated by this goal, we conducted experiments to evaluate the performance of 4 popular AutoML systems, including Auto^{n}ML, TPOT, AutoGluon and auto-sklearn on 270 OpenML regression tasks, using R-Squared score as the evaluation metric. We analyzed the experimental data from various aspects, including relative rankings of the AutoML systems, and relationship between performance of AutoML systems and dataset characteristics. we have also run repeating experiments for selected AutoML systems on a few selected datasets to analyze the stability of AutoML performances under the same condition.**

Index Terms— **AutoML, OpenML, model selection**

I. INTRODUCTION

An AutoML system includes a search space which contains multiple pipelines. An AutoML pipeline consists of a series of steps of data preprocessing, feature selection, hyperparameter tuning and model selection, and can be treated as a predictor that takes machine learning (ML) tasks as input and produces predication values as output. An AutoML system works by searching for high-performance pipelines in its search space. Typically, when fed with an ML task, an AutoML system returns a leaderboard which ranks the performance of top pipelines it has found according to a given metric (accuracy, R-Squared, etc). In this sense, an AutoML system can be regarded as a huge ML model. For each ML task, the performance of an AutoML system can be represented by the performance of the best pipeline it has found —- the pipeline with the highest ranking or evaluation score on the AutoML's leaderboard.

Many open-source AutoML systems are available to use to facilitate data scientists and domain experts. Here are several examples:

(1) Auto^{n}ML: Auto^{n}ML is an open-source AutoML system developed by the Auton Lab at CMU [1] based on DARPA D3M ecosystem, with the goal of facilitating efficient model discovery and advanced data analytics. Auto^{n}ML takes the input training data, then conducts operations on the input data including featurization, fitting and prediction, and validation. Eventually, Auto^{n}ML returns a leaderboard of ranked pipelines with their respective training prediction scores and the pipelines can be applied to make predictions the testing data.

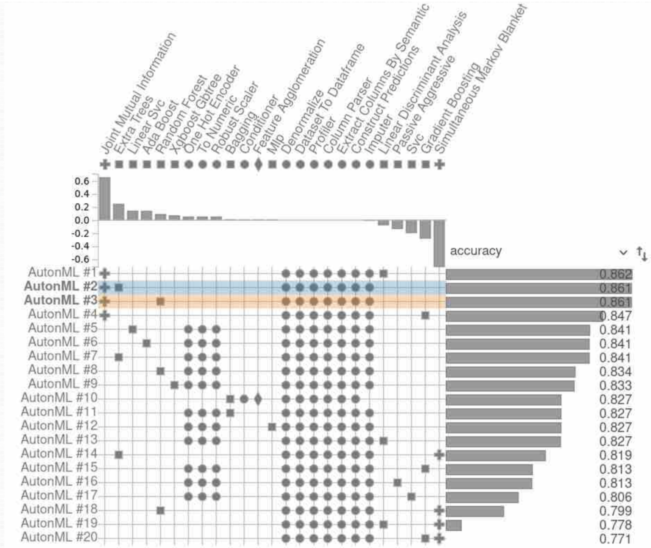


Fig. 1: A Visualization of Auto^{n}ML Pipelines

(2) Tree-Based Pipeline Optimization Tool (TPOT): Introduced by Olson and Moore [2], TPOT is an open-source AutoML system based on genetic programming. TPOT utilizes tree representation and stochastic search algorithms to promote pipeline building processes. TPOT uses Python-based scikit-learn library for its implementation of core algorithms. Figure 2 is an example tree-based pipeline from TPOT, with Each circle corresponding to an ML operator and the arrows indicating the direction of the data flow [2].

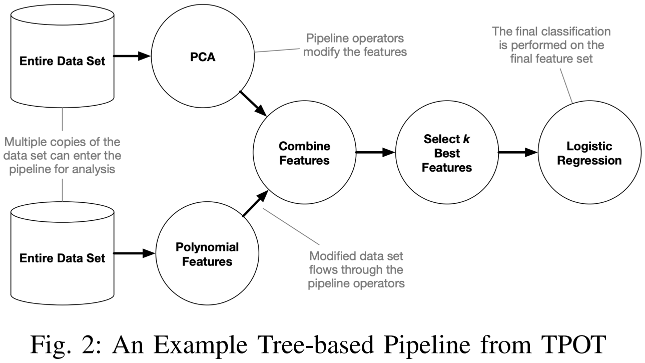

Entire Data Set
PCA
Pipeline operators modify the features
The final classification is performed on the final feature set
Multiple copies of the data set can enter the pipeline for analysis
Combine Features
Select k Best Features
Logistic Regression
Entire Data Set
Polynomial Features
Modified data set flows through the pipeline operators

Fig. 2: An Example Tree-based Pipeline from TPOT

[1] Xinchen Yang is with the Department of Electrical and Computer Engineering, New York University, 6 MetroTech Center, Brooklyn, NY, USA xy2332@nyu.edu

[2] Jieshi Chen and Artur Dubrawski, PhD, are with the Auton Lab, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA jieshic@andrew.cmu.edu, awd@cs.cmu.edu

(3) AutoGluon-Tabular: Presented by Erickson [3], AutoGluon-Tabular is an open-source AutoML system that favors ensemble techniques by stacking multiple models in multiple layers. Experiments has shown that the multi-layer combination of many models offers better use of allocated training time [3], which serves practical usages well.

In this paper, we evaluated the performance of 4 AutoML systems: Auto^{n}ML, TPOT, AutoGluon and auto-sklearn on 270 OpenML regression tasks. Our goal is to compare the performance of these AutoML systems, identify potential relationships between the performance of AutoML systems and factors such as time budget, datasets characteristics, and core algorithm selection.

II. Related Work

With the continuous emergence and evolution of new AutoML methods and systems, researchers conduct comprehensive surveys on AutoML methods and systems along the way at the same time. Zoller and Huber surveyed on state of the art AutoML methods and a benchmark of popular AutoML frameworks on real data sets. In their work, selected AutoML frameworks (H_2O AutoML, TPOT, etc.) are evaluated on 137 real-world datasets [4]. Truong conducted various evaluations of the AutoML tools on nearly 300 datasets collected from OpenML to examine their performance, observing that many AutoML tools can obtain reasonable results in terms of their performance across many data tasks, but there is no "perfect" tool that can outperform almost all others yet [5]. Aragao performed a comparative study among multiple AutoML tools related to the features, architecture, capabilities, and results, which are are achieved on binary, multiclass, and multilabel classification problems from experimentation on various datasets [6].

Additionally, various research has been done on the evaluation of AutoML systems by AutoML developers. Inventors of AutoML methods and developers of AutoML systems often provide an evaluation of their AutoML methods or systems against others AutoMLs when they introduce their work. For example, Erickson, the inventor of AutoGluon-Tabular, evaluated AutoGluon-Tabular as well as several other AutoML systems on a combined suite of 50 classification and regression tasks from Kaggle and the OpenML AutoML Benchmark, proving the robustness and accuracy of AutoGluon-Tabular [3]. Thomas introduced an automatic framework — Automatic gradient boosting and one of its implementation, and compared this framework to several other current AutoML projects on 16 datasets [7]. Li presented VolcanoML, a scalable and extensible framework that facilitates systematic exploration of large AutoML search spaces, and conducted thorough evaluations on VolcanoML against several other AutoML systems, demonstrating that VolcanoML can raise the level of AutoML search space expressiveness [8].

III. Methods

Experiments are set up to answer the following questions:

(1) What is the performance of different AutoML systems compared to each other?

(2) What is the performance of the AutoML systems under different time budgets?

(3) What is the performance of the AutoML systems on datasets with different characteristics?

(4) What are the main contributors to the performance discrepancies among different AutoML systems, core algorithm selection or other factors?

To find the answers to the questions above, we evaluated 4 AutoML systems: Auto^{n}ML, TPOT, AutoGluon, and auto-sklearn on 270 OpenML regression tasks. Datasets were selected with diversified dimensionalities and number of instances. Experiments were run on an 8-core Linux Machine. First, each OpenML data task is randomly split into training data and testing data, with 75% of the input data used as training data and 25% used as testing data. After that, for each task, the same training and testing data are fed as input to each of the AutoML systems. The metric used to measure the performance of the AutoML systems is R-Squared. Each AutoML system should try to return the best pipelines it can find and rank the pipelines according to their evaluation scores. For each AutoML system, the Top 1 pipeline on its leaderboard represents the AutoML system and is applied on the testing data to get the testing score. We set up 3 groups of experiments with the time budget to be 60 seconds, 600 seconds and 1200 seconds respectively. The same evaluation process is repeated only with differed time budget.

IV. Results

After running the experiments, we collected training and testing prediction scores of the AutoML systems on 270 regression tasks under 3 different time budgets and analyzed the experimental data from several aspects.

A. AutoML Performances over Different Time Budgets

To illustrate the performance of AutoML systems over different time budgets, we compared rankings of different AutoML systems on testing data under the time budget of 60 seconds, 600 seconds, and 1200 seconds, as shown in Figure 3. For each data point, its X-coordinate denotes the time budget and its Y-coordinate is obtained by averaging the performance rankings of its corresponding AutoML system across all datasets, with corresponding 95% confidence intervals. For an AutoML system which has failed to produce results on a specific data task under a given time budget, we assign 4 (which is the number of total evaluated AutoML systems) to be its ranking for this data task under the given time budget.

In Figure 3, when the time budget is set to 60 seconds and 600 seconds, TPOT achieves the highest average rank among all AutoML systems, while when the time budget is set to 1200 seconds, Auto^{n}ML achieves the highest average rank. We can observe that as time budget increases, the average rankings of different AutoML systems get closer to each other. Additionally, the average ranking of TPOT downgrades slightly and the average ranking of auto-sklearn improves slightly as more time is allocated.

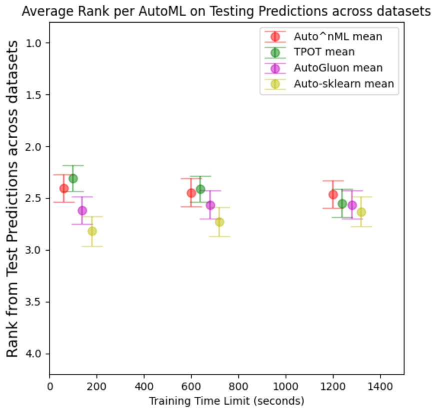


Fig. 3: Average Ranking of AutoML Systems in terms of Testing R-Squared Score

TABLE I: Rank Statistics on Test data per Time Budget, Averaged across N=270 Datasets, ± Standard Error of the Mean

	60 seconds	600 seconds	1200 seconds
Auto^{n}ML	2.41±0.13	2.45±0.13	2.46±0.13
TPOT	2.31±0.13	2.41±0.13	2.55±0.14
AutoGluon	2.62±0.13	2.56±0.14	2.56±0.14
auto-sklearn	2.82±0.14	2.73±0.14	2.63±0.14

B. Relationship Between Performance of AutoML Systems and Dataset Characteristics

1) AutoML Performances v.s. Dimensionality/Number of Instances of Datasets: We are interested in how the characteristics of the datasets, in particular, dimensionality and number of instances, affect the performance of the AutoML systems.

To illustrate the relationship between the performance of AutoML systems and the dimensionality of the datasets, we divide the whole range of dimensionality into distinct ”buckets” using an interval of 10. Within each ”bucket”, we average the testing scores over all time budgets over all datasets in the bucket for each AutoML system to obtain a data point. The results are shown in Table II and Figure 4.

Similarly, to illustrate the relationship between the performance of AutoML systems and the number of instances of the datasets, we divide the whole range of number of instances into distinct ”buckets” using an interval of 1000. Within each ”bucket”, we average the testing scores over all time budgets over all datasets in the bucket for each AutoML system to obtain a data point. The results are shown in Table III and Figure 5.

Regarding the relationship between the performance of AutoML systems and the dimensionality of datasets, the average ranking of Auto^{n}ML shows an increasing pattern when the dimensionality of datasets increases within the range of (0, 80), as shown in Figure 4. For other AutoML systems, we are not able to find obvious relationships between their performances and dimensionality of datasets.

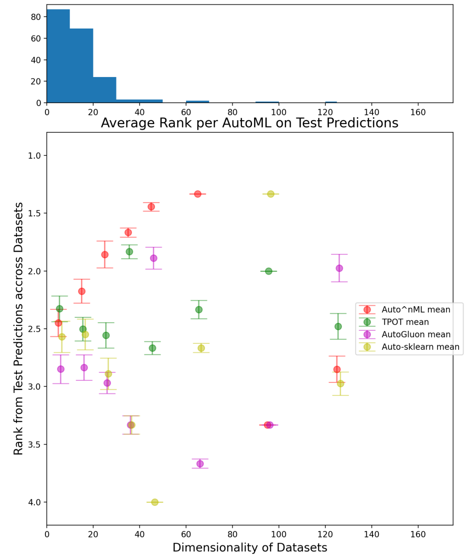


Fig. 4: Average Ranking per AutoML on Testing Predictions across dDtasets within Different Dimensionality Groups

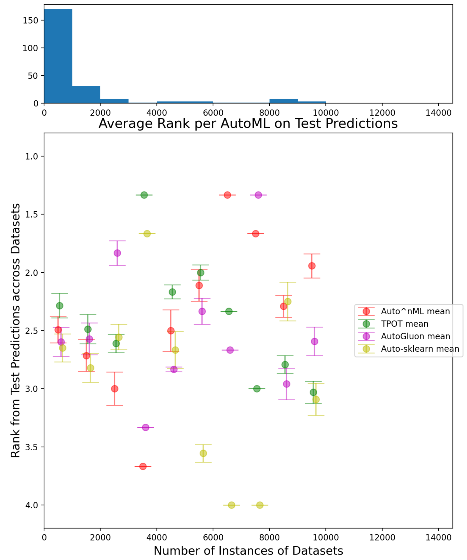


Fig. 5: Average Ranking per AutoML on Testing Predicitions accross Datasets within Different Groups of Number of Instances

Regarding the relationship between the performance of AutoML systems and number of instances, Figure 5 shows that when the number of instances is less than or equal to 6000, TPOT maintains a leading position for most of the time. On the contrary, when the number of instances is greater than 6000, Auto^{n}ML takes over and gets the first place for most of the time. The advantage of Auto^{n}ML is especially significant when the number of instances in the range of [9000, ∞).

2) Dataset Clustering: We hope to investigate the relationship between the ”difficulty” of data tasks and AutoML performances. Here, we use four attributes of to gauge the difficulty of each data task: (1) number of features; (2)

TABLE II: Average Ranking per AutoML on Testing Predictions across N=270 Datasets within Different Dimensionality Groups

Dimensionality	Number of Tasks	Auto^{n}ML	TPOT	AutoGluon	auto-sklearn
[0,10)	87	2.45±0.12	2.33±0.11	2.85±0.12	2.57±0.14
[10,20)	69	2.17±0.10	2.50±0.10	2.84±0.11	2.55±0.13
[20,30)	24	1.86±0.12	2.56±0.11	2.97±0.09	2.89±0.14
[30,40)	3	1.67±0.04	1.83±0.06	3.33±0.08	3.33±0.08
[40,50)	3	1.44±0.04	2.67±0.06	1.89±0.09	4.00±0.00
[50,60)	0	NaN	NaN	NaN	NaN
[60,70)	2	1.33±0.00	2.33±0.08	3.67±0.04	2.67±0.04
[70,80)	0	NaN	NaN	NaN	NaN
[80,90)	0	NaN	NaN	NaN	NaN
[90,100)	1	3.33±0.00	2.00±0.00	3.33±0.00	1.33±0.00
[100,110)	0	NaN	NaN	NaN	NaN
[110,120)	0	NaN	NaN	NaN	NaN
[120,∞)	81	2.85±0.11	2.48±0.11	1.98±0.12	2.98±0.10

TABLE III: Average Ranking per AutoML on Testing Predictions across N=270 Datasets within Different Groups of Number of Instances

Number of Instances	Number of Tasks	Auto^{n}ML	TPOT	AutoGluon	auto-sklearn
[0,1000)	170	2.49±0.11	2.29±0.10	2.60±0.13	2.65±0.12
[1000,2000)	31	2.71±0.14	2.49±0.13	2.57±0.14	2.82±0.12
[2000,3000)	8	3.00±0.14	2.61±0.08	1.83±0.11	2.56±0.11
[3000,4000)	1	3.67±0.10	1.33±0.00	3.33±0.00	1.67±0.00
[4000,5000)	3	2.50±0.18	2.17±0.06	2.83±0.02	2.67±0.16
[5000,6000)	3	2.11±0.14	2.00±0.06	2.33±0.11	3.56±0.07
[6000,7000)	1	1.33±0.00	2.33±0.00	2.67±0.00	4.00±0.00
[7000,8000)	1	1.67±0.00	3.00±0.00	1.33±0.00	4.00±0.00
[8000,9000)	8	2.29±0.09	2.79±0.08	2.96±0.14	2.25±0.17
[9000,∞)	44	1.94±0.10	3.03±0.10	2.59±0.12	3.09±0.14

number of instances; (3) success rate of experiments run on the data task (4 AutoMLs × 3 time budgets = 12 experiments per data task); (4) maximum R-Squared score achieved on the data task (maximum R-Squared score over 4 AutoMLs × 3 time budgets = 12 experiments per data task). Figure 6 shows the distribution of "attributes" success rate and maximum R-Squared score over all datasets.

We use N = 5 clusters and do a KMeans Clustering according to the values of the 4 attributes (number of features, number of instances, success rate, maximum R-Squared score) of the data tasks. We do the clustering by scaling the actual values of each attribute to scaled values with mean of 0 and standard deviation of 1. After the clustering, the whole group of 270 datasets is divided into 5 subgroups, and we denote the 5 groups as Cluster 0, 1, 2, 3, 4. A summarization of attributes and dataset characteristics of each cluster is shown in Figure 7.

To analyze the relationship between the performance of Auto^{n}ML and dataset characteristics, we collected the distribution of Auto^{n}ML rankings for each of the cluster, shown in Figrue 8. We focus on the experiments conducted under the time budget of 1200 seconds. We can find that under the time budget of 1200 seconds: (1) Auto^{n}ML performs better on datasets in Cluster 1, which is a cluster consists of datasets with low success rate, having a relatively high percentage frequency of getting the first place. (2) There is a large percentage of datasets in Cluster 1 (13 out of 28) and Cluster 3 (1 out of 3) where none of the AutoML systems can produce results. Those datasets are probably challenging data tasks and need future experiments and exploration. (3) Auto^{n}ML did worse on data tasks in Cluster 2, where the dimensionality is high. Additionally, we can also observe that Auto^{n}ML shows a more significant advantage when evaluated on datasets in Cluster 1 (cluster of datasets with low success rate) as the time budget increases.

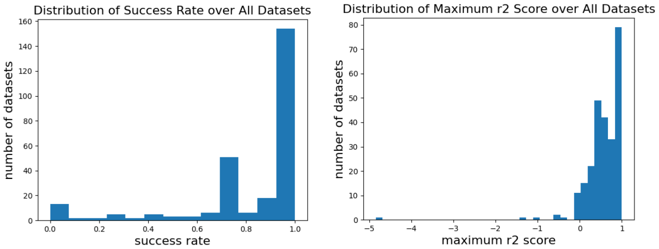
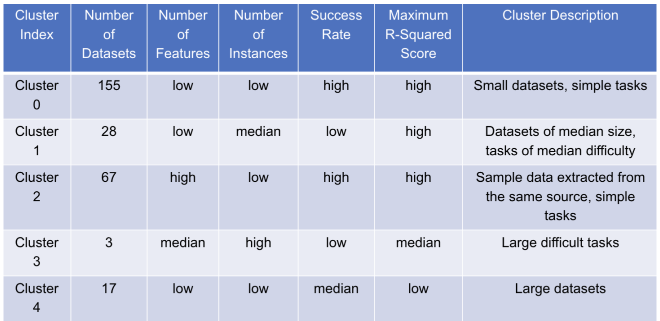


Fig. 6: Distribution of Success Rate and Maximum R-Squared Score over all Datasets



Cluster Index	Number of Datasets	Number of Features	Number of Instances	Success Rate	Maximum R-Squared Score	Cluster Description
Cluster 0	155	low	low	high	high	Small datasets, simple tasks
Cluster 1	28	low	median	low	high	Datasets of median size, tasks of median difficulty
Cluster 2	67	high	low	high	high	Sample data extracted from the same source, simple tasks
Cluster 3	3	median	high	low	median	Large difficult tasks
Cluster 4	17	low	low	median	low	Large datasets

Fig. 7: Description of Dataset Clusters

C. Repeating Experiments on a Selected Subset of Datasets

We are interested in the learning about what factors can lead to the performance discrepancies among AutoML systems: Is it the core algorithm selection, or related to other steps such as preprocessing? Hence, we collected the algorithm used by the top pipeline of each AutoML system on each experiment. We have found that under each time budget, there are around 10 out of 270 data tasks where both the top pipeline of Auto^{n}ML and TPOT use "extra trees" as its core algorithm, but TPOT beats Auto^{n}ML. Figure 9 shows a few of them where TPOT shows a significant advantage towards Auto^{n}ML.

To further investigate the performance of Auto^{n}ML and TPOT on these data tasks, we run $N = 10$ repeating experiments to evaluate Auto^{n}ML and TPOT on OpenML datasets 8, 299, and 570 under the time budget of 600 seconds. We collected the time consumed to train the AutoML system and the testing score of the AutoML system for each experiment. For each data task, we made a scatter plot out of its experimental data points, as shown is Figure 10. Each data point is corresponding to one experiment conducted on the data task, with its X-coordinate denoting the training time and its Y-coordinate denoting the testing score. Red dots represent experiments conducted on Auto^{n}ML and green dots represent experiments conducted on TPOT.

We can observe from the plots that across multiple runs of the same AutoML system on the same data task (time budget

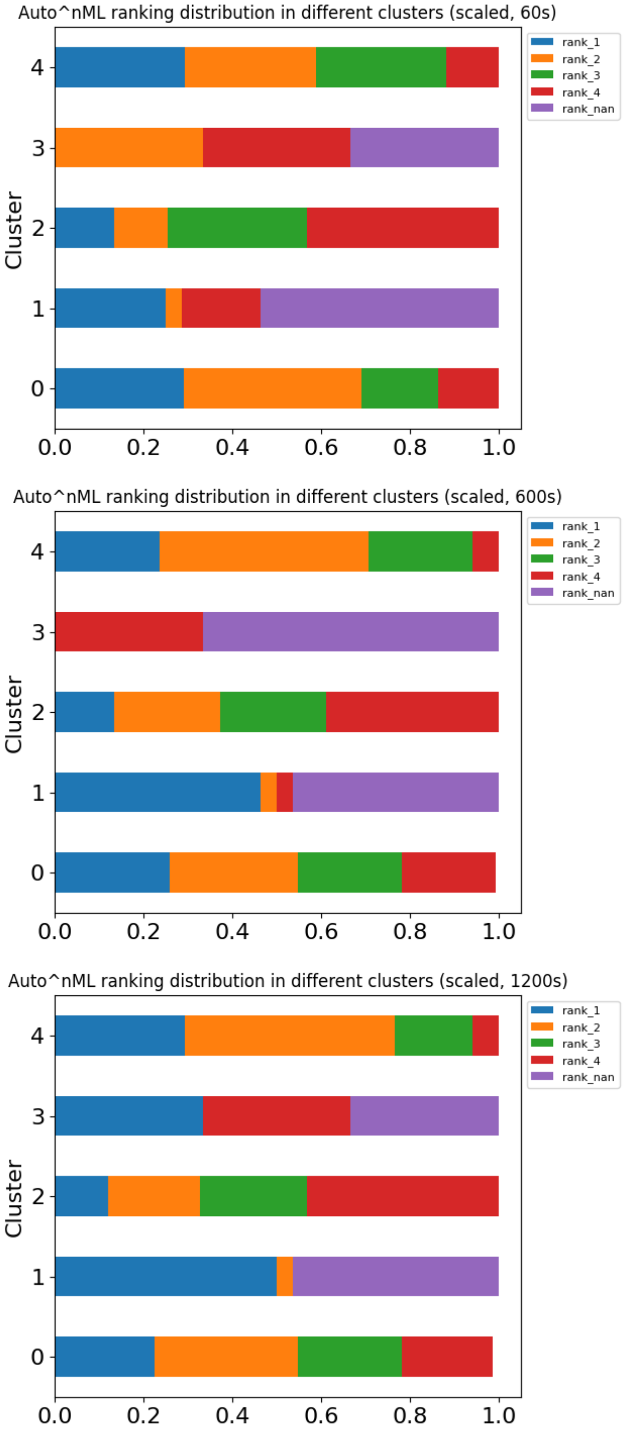


Fig. 8: Auto^{n}ML Ranking Distribution in Different Clusters

Data Task ID	Number of Features	Number of Instances	Auto^{n}ML testing score	TPOT testing score	Auto^{n}ML algorithm	TPOT algorithm
8	6	345	0.2692	0.3068	Extra trees	Extra trees
299	91	360	0.736	0.8525	Extra trees	Extra trees
570	12	316	0.0554	0.2628	Extra trees	Extra trees

Fig. 9: Datasets where TPOT beats Auto^{n}ML when they use the same core algorithm

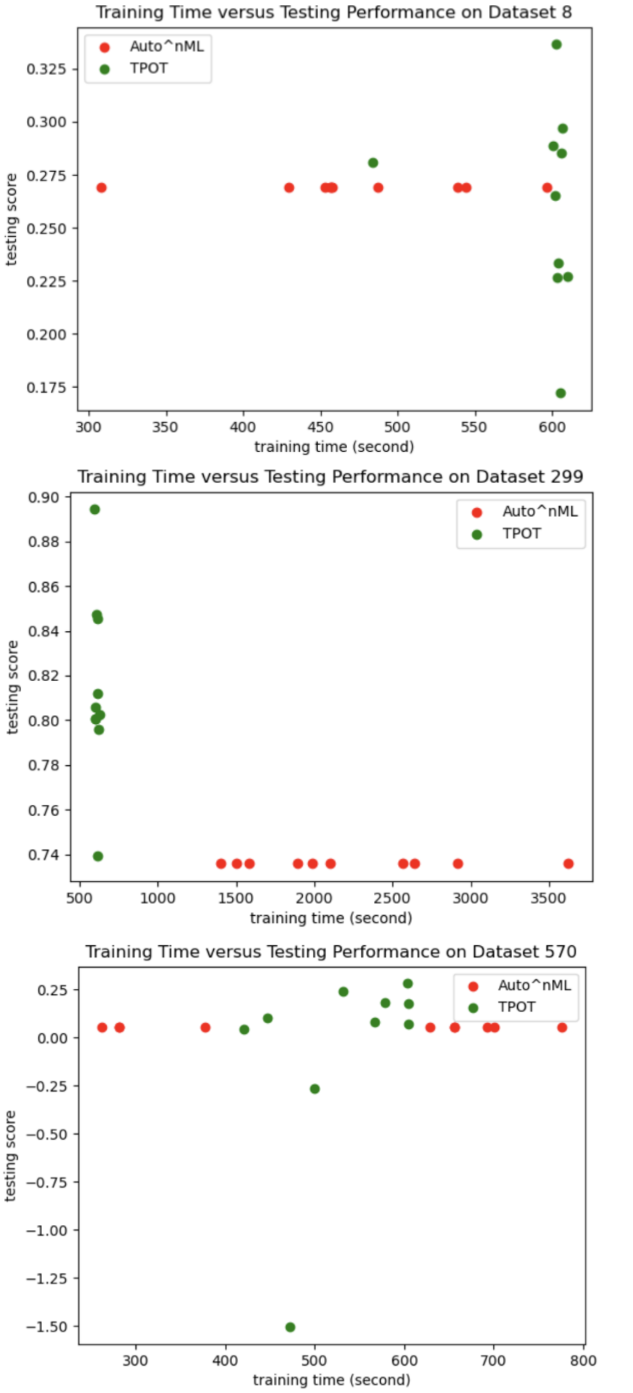


Fig. 10: Auto^{n}ML Ranking Distribution in Different Clusters

= 600 seconds), Auto^{n}ML has low (almost no) variation in testing scores but high variation in training time, while TPOT is quite the opposite with high variation in testing scores but low variation in training time.

V. CONCLUSIONS & FUTURE WORK

We evaluated the performance of 4 AutoML systems on 270 OpenML regression tasks, using R-Squared as the evaluation metric. We analyzed the experimental data from several aspects, including relative rankings of the AutoML systems, relationship between AutoML performances and dataset characteristics. We found that Auto^{n}ML is an overall high-ranking AutoML system. Auto^{n}ML shows to be able to obtain a higher relative ranking on datasets with low success rate. Additionally, we found some data tasks that are ”interesting” because both Auto^{n}ML and TPOT use the same core algorithm (e.g. extra trees) but TPOT beats Auto^{n}ML. We run repeating experiments for both Auto^{n}ML and TPOT on a few of such datasets under the time budget of 600 seconds, and find that while Auto^{n}ML has low variation in testing score but high variation in training time across multiple runs on the same data task, TPOT has low variation in training time but high variation in testing score.

In the future, we aim to dive deeper to explore the pipeline building of each AutoML system to identify the major factors (including core algorithm selection, preprocessing steps, etc.) which contribute to the performance discrepancies among AutoML systems. Moreover, as currently the evaluation of AutoML systems is manually conducted and is rather time-consuming, we hope to automate this process. In the future, we hope to build a product which can generate AutoML performance statistics, visualization plots and diagnostic reports rather easily, with just a few clicks or lines of code, to benefit the developers of AutoML systems.

VI. ACKNOWLEDGMENT

This work is supported by DARPA D3M MILEI funding and Carnegie Mellon Robotics Institute Summer Scholars (RISS) program. The authors would like to thank Rachel Burcin and Dr. John M. Dolan for their organization and support of this program.

REFERENCES

[1] “Auto^{n}ml - taking your ml capacity to the nth power,” https://autonlab.org/research/usability.html.

[2] R. S. Olson and J. H. Moore, “Tpot: A tree-based pipeline optimization tool for automating machine learning,” in *Proceedings of the Workshop on Automatic Machine Learning*, ser. Proceedings of Machine Learning Research, F. Hutter, L. Kotthoff, and J. Vanschoren, Eds., vol. 64. New York, New York, USA: PMLR, 24 Jun 2016, pp. 66–74.

[3] N. Erickson, J. Mueller, A. Shirkov, H. Zhang, P. Larroy, M. Li, and A. Smola, “Autogluon-tabular: Robust and accurate automl for structured data,” 2020. [Online]. Available: https://arxiv.org/abs/2003.06505

[4] M.-A. Zoller and M. F. Huber, “Benchmark and survey of automated machine learning frameworks,” *Journal of artificial intelligence research*, vol. 70, pp. 409–472, 2021.

[5] A. Truong, A. Walters, J. Goodsitt, K. Hines, C. B. Bruss, and R. Farivar, “Towards automated machine learning: Evaluation and comparison of automl approaches and tools,” in *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, 2019, pp. 1471–1479.

[6] M. V. Cysneiros Aragão, A. Guimarães Afonso, R. C. Ferraz, R. Gonçalves Ferreira, and S. Gomes Leite, “A practical evaluation of automl tools for binary, multiclass, and multilabel classification,” 2023.

[7] J. Thomas, S. Coors, and B. Bischl, “Automatic gradient boosting,” 2018.

[8] Y. Li, Y. Shen, W. Zhang, J. Jiang, B. Ding, Y. Li, J. Zhou, Z. Yang, W. Wu, C. Zhang, and B. Cui, “Volcanoml: Speeding up end-to-end automl via scalable search space decomposition,” *Proceedings of the VLDB Endowment*, vol. 14, no. 11, pp. 2167–2176, jul 2021. [Online]. Available: