# Smart Model Elimination for Efficient Automated Machine Learning in Healthcare

**Eric Su Zhang**[1]
St. Mark's School of Texas
10600 Preston Rd, Dallas, TX 75230
26zhange@smtexas.org

**Benjamin Joseph Michael Standefer**[2]
St. Mark's School of Texas
10600 Preston Rd, Dallas, TX 75230
26standeferb@smtexas.org

## Abstract

Automated Machine Learning or AutoML has emerged as a popular field of research. We perform a literature review of existing AutoML papers and conduct a survey on five popular AutoML frameworks. Many of these frameworks optimize a form of model selection, but they all require every model to be run and evaluated. We propose a novel framework, SMEML (Smart Model Elimination Machine Learning), that automatically eliminates models that are unlikely to be performant. SMEML demonstrates the ability to generate a model with near equal accuracy to a "dumb" brute-force framework, but in a fraction of the time. We also believe this innovation is particularly relevant to machine learning in healthcare because of the necessity for efficient diesease prediction.

## 1 Introduction

Machine learning (ML) is a type of artificial intelligence (AI) that uses searching and trend-determining to make predictions. ML algorithms are trained and tested on a given dataset to become "smarter". Once they've learned the patterns typical of that data, they can make predictions based on similar data. These algorithms also require additional processes, however, like hyperparamater tuning, and can be modified in conjunction with one another, like in ensemble learning. This makes the process complex and time-inefficient. To do this on a larger scale and at greater speeds, recent research and effort have been put into developing AutoML frameworks [1].

One sector of the global industry that has the most potential for beneficial integration of AutoML frameworks is healthcare, particularly in developing countries. These countries have severe shortages of healthcare workers and limited tools for diagnosis. For example, Africa has 2.3 healthcare workers per 1000 individuals, while the Americas have 24.8 healthcare workers per 1000 [2]. The World Health Organization (WHO) emphasized that this deficit is growing every year and will likely reach 18 million personnel by 2030 [3]. Medical AIs typically automate repetitive tasks, making time consumption a primary concern [4]. This coupled with a lack of adequate resources makes designing faster diagnostic systems for developing countries' medical sectors a pivotal issue.

Most AutoML frameworks implement some form of model selection where a pool of models are filtered until a final model is selected. However, these frameworks require that every model be run [1]. In theory, this means that much energy is spent training models that are unlikely to be selected. We propose SMEML, a novel model selection algorithm that automatically eliminates models that it believes will not be performant.

Preprint. Under review.

Table 1: Survey of Existing AutoML Frameworks

| Framework | EOU | M | FE | TM | HPT | Inter | Custom |
|---|---|---|---|---|---|---|---|
| H2O AutoML [5] | Easy | 12 | Basic | Yes | Auto | SHAP | High |
| TPOT [6] | Hard | 10 | Basic | No | Genetic | POJO | Moderate |
| MLJAR [7,8] | Easy | 12 | Advanced | Yes | Auto | Basic | High |
| FLAML [9] | Easy | 9 | Limited | Yes | Auto | SHAP | Low-Moderate |
| LightAutoML [10] | Easy | 11 | Advanced | Yes | Auto | Basic | Moderate |

**Abbreviations**: EOU: Ease of Use, M: # of models evaluated by framework, FE: Feature Engineering, TM: Time Management, HPT: Hyper-Parameter Tuning, Inter: Interface, Custom: Customization

### 1.1 Survey

We systematically review and compare five different AutoML frameworks, comparing every feature relevant to medical implementation in Table 1. Of these five, we wanted to take a closer look at four: FLAML, H2O, MLJAR, and TPOT.

We can observe pros and cons in the four models' theoretical applications. FLAML uses blend search hyper-parameter optimization and focuses on lightweight models, making it quick and applicable to repetitive tasks. It supports user-specified imputation techniques for missing values and has an intuitive API, making it moderately user-friendly for healthcare workers [9]. H2O has a wider variety of supported algorithms that may be appropriate for medical diagnosis, making it slower than FLAML on average. In one trial, more than half of FLAML's performances in one minute were better than or equal to H2O's performances in one hour [9]. H2O has automatic and multi-faceted methods of dealing with missing values and flexible interfaces in R and Python, making it intuitive for users [5]. MLJAR uses advanced algorithms such as light gradient boosting and neural networks. It automatically deals with missing values and is known for its automated user interface [7]. TPOT uses genetic programming. It builds pipelines over multiple generations, which can be time consuming. It has built-in pre-processing for missing values, but the genetic programming requires fine-tuning from users [6].

Through this survey, we uncovered a key continuity in these frameworks' design: frameworks must run all of their models to maximize accuracy at the expense of efficiency. Given the specific nature of certain tabular data, there are some models that are less likely to perform well, thus running and training these models is wasteful. Cutting these models completely, however, is not optimal, as they are the best option in a minority of cases. We hypothesized that using a combination of multi-layer machine learning implementation and meta-feature extraction would yield an equally versatile system that minimizes time spent on training without sacrificing accuracy.

## 2 Methodology

When considering a specific dataset, the distribution, relationship between datapoints, and various other attributes can give us clues as to which models are likely to perform best. For example, linear models tend to work better on datasets where there is a linear relationship between features [11]. Accordingly, we can develop a boosting model to predict the best models based on extracted attributes from a dataset. These extracted attributes are shown in Table 2. Each attribute attempts to capture a specific type of relationship or distribution within the data.

We considered 28 different models, including all supervised and semi-supervised sklearn [12] classification models and common boosting models such as XGBoost [13], LightGBM [14], and Catboost [15].

To implement the model elimination process, we train a multilabel-regression boosting model called the SME Model-Ranker on 268 kaggle datasets. We choose medical datasets and focus our approach on binary classification. We extract the attributes in Table 2 from each dataset: which are the inputs for the boosting model. We then run a "dumb", brute-force framework that trains every model every each dataset. The accuracy of every model is ranked, and every model is treated as a label with its value being its rank. We train the SME to predict the rank of each model, given the extracted attributes. SME produces a median spearman correlation of 0.7301 between the predicted rank and the actual rank and a median p-value of 1.032e-05. Furthermore, we guarantee that we can predict a top two model within the predicted top eight 90.12% of the time.

Table 2: Attributes extracted from a dataset

| Attribute | Formula | Purpose | Weight |
|---|---|---|---|
| # of Rows | $\lvert\mathbf{A}\rvert_r$ | DS | .0422 |
| # of Columns | $\lvert\mathbf{A}\rvert_c$ | DS | .0415 |
| Target Distribution | $\frac{\max(\text{count}_1, \text{count}_0)}{\lvert\mathbf{A}\rvert_r}$ | DT | .0489 |
| Numerical Columns | $\frac{\lvert\mathbf{A}_{num}\rvert}{\lvert\mathbf{A}\rvert_c}$ | DT | .0428 |
| Binary Categorical Columns | $\frac{\lvert\mathbf{A}_{bin}\rvert}{\lvert\mathbf{A}\rvert_c}$ | DT | .0117 |
| Mean # of Distinct Values | $\mu\{\text{\# of distinct values in column}\}$ | DT | .0076 |
| Mean IQR | $\mu\{\text{IQR of column}\}$ | DD | .0594 |
| STD of IQR | $\sigma\{\text{IQR of column}\}$ | DD | .0700 |
| Mean Q1 | $\mu\{\text{Q1 of column}\}$ | DD | .0789 |
| STD of Q1 | $\sigma\{\text{Q1 of column}\}$ | DD | .0613 |
| Mean Q3 | $\mu\{\text{Q3 of column}\}$ | DD | .0812 |
| STD of Q3 | $\sigma\{\text{Q3 of column}\}$ | DD | .1103 |
| Mean Percent of Z-Score Outliers | $\mu\left\{\frac{\text{\# of Z-Score Outliers}}{\lvert\mathbf{A}\rvert_r}\right\}$ | ODP | .0558 |
| STD of Percent of Z-Score Outliers | $\sigma\left\{\frac{\text{\# of Z-Score Outliers}}{\lvert\mathbf{A}\rvert_r}\right\}$ | ODP | .0798 |
| Mean Correlation | $\mu\{\text{Correlation Matrix}\}$ | DL | .0559 |
| STD of Correlation | $\sigma\{\text{Correlation Matrix}\}$ | DL | .1526 |

**Abbreviations**: DS: Data Size, DT: Data Type, DD: Data Distribution, ODP: Outlier Data Points, DL: Data Linearity, Weight: Variable Importance

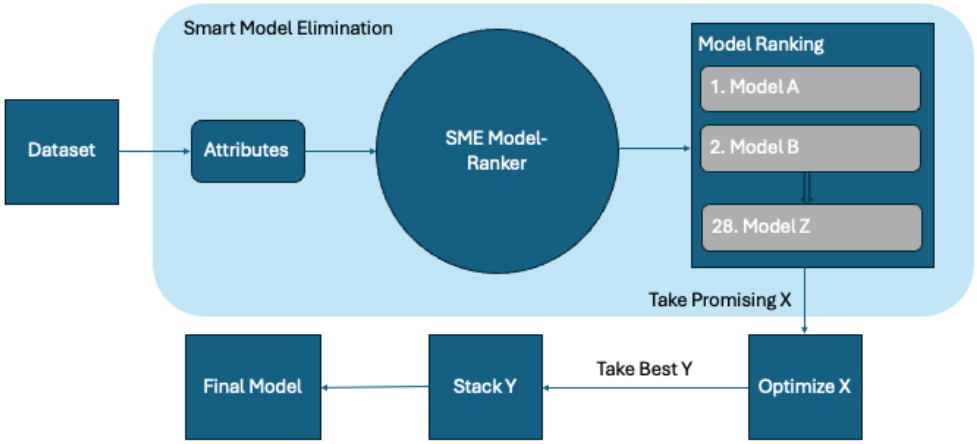


Figure 1: Illustration of the SMEML Process

When a user inputs a dataset into SMEML, it first extracts the attributes from the dataset and then feeds them into the SME to get the predicted rank of each model. Since we have determined that we can predict a top two model within the predicted top eight quite confidently, we can eliminate the bottom 20 models from the list. We run Bayseian optimization using scikit-optimize [16] on the top eight models to get the best hyperparamters for each model. We select the top models that have significantly better performance than other models. These models are stacked together to form the final model. This process is shown in Figure 1.

Table 3: Comparing experiment results for SMEML and dumb mode

| Dataset | Accuracy | | Time to Train (sec) | | Size (RxC) |
|---|---|---|---|---|---|
| | SMEML | Dumb | SMEML | Dumb | |
| Lung Cancer Prediction [17] | .9839 | .9839 | 143.07 | 317.04 | 309x16 |
| Chronic Kidney Disease Dataset [18] | 1.0 | 1.0 | 419.64 | 788.46 | 400x26 |
| Pima Indians Diabetes Dataset [19] | .7857 | .8052 | 161.99 | 374.01 | 768x8 |
| Brain Stroke Prediction Dataset [20] | .9458 | .9458 | 114.16 | 370.95 | 4892x11 |
| Stroke Prediction Dataset [21] | .9393 | .9393 | 172.67 | 385.23 | 5110x12 |
| Eye State Classification [22] | .9536 | .9549 | 312.57 | 696.51 | 14980x15 |

# 3 Evaluation

## 3.1 Experiment Design

To evaluate our framework, we choose 5 different datasets of diverse size. We ran SMEML on each dataset for 20 iterations of tuning. We compared this against a "dumb" mode which simply runs every model on the dataset. We used the accuracy and time to train as our metrics. All experiments are run on an Ubuntu server with Intel Xeon E5520 @ 2.27GHz and 24GB of RAM.

## 3.2 Experiment Results

Several observations were made from running these datasets. First, accuracy did not change for 4 of the 6 benchmarks. The other two benchmarks on eye states and diabetes prediction represented a loss of accuracy for SMEML by 0.13% and 1.95%, respectively. Training time, however, favored SMEML in every trial. SMEML outperformed the dummy framework by 173.96 seconds in its worst performance on the lung cancer trial and by 383.94 seconds in its best performance on the eye state dataset. Relative to the dummy's performance, a percent decrease in time to run for SMEML of at least 46.78% could be seen, with a decrease of up to 69.22% observerd as well. SMEML's worst performance by percent decrease was also the performance in which it returned perfect accuracy.

# 4 Discussion

## 4.1 Summary

SMEML manages to return comparable accuracy results to its dumb couterpart while more than splitting the run time in half in the majority of benchmarks. Its performance is not limited by the size of the dataset, and it returned its first, third, and fourth overall best run times for SMEML relative to the dumb mode on the 'large' dataset benchmarks (those exceeding 1000 rows). This is important in third world medicine where time is of the essence and there is a constant influx of new patient data. A practical demonsration of this software's real-world application can be found on our SmartDx website [23]. Source code can be found at [24].

## 4.2 Limitations

Our implementation of SMEML has limitations. These include a small training set due to lack of available datasets, finite computing power, time constraints, and a lack of access to paticular scientific literature. Our framework only improves on the model selection process and does not address the various other aspects of AutoML. Therefore, SMEML is not comparable to other full-featured AutoML frameworks.

## 4.3 Future Work

Future work will include increasing the size of the training set, implementing a fully fledged AutoML framework, and conducting a detailed experiment comparing SMEML to other popular AutoML frameworks. Furthermore, we will examine other types of machine learning such as regression and multiclass-classification.

## Acknowledgments and Disclosure of Funding

We would like to thank St. Mark's School of Texas for supporting this research.

## References

[1] He, X., et al., "AutoML: A Survey of the State-of-the-Art." Knowledge-Based Systems 212, 106622 (2021). https://doi.org/10.1016/j.knosys.2020.106622.

[2] Nchasi, G., et al., "Challenges Faced by African Healthcare Workers During the Third Wave of the Pandemic." Health Science Reports 5, 6 (2022). https://doi.org/10.1002/hsr2.893.

[3] "Global Strategy on Human Resources for Health: Workforce 2030." World Health Organization, pg. 16 (2020). https://www.who.int/publications/i/item/9789241511131.

[4] Bajwa, J., et al., "Artificial Intelligence in Healthcare: Transforming the Practice of Medicine." Future Healthcare Journal 8, 188 (2021). https://doi.org/10.7861/fhj.2021-0095.

[5] LeDell, E., and S. Poirier, "H2O AutoML: Scalable Automatic Machine Learning." 7th ICML Workshop on Automated Machine Learning (2020). https://automl.org/wp-content/uploads/2020/07/AutoML_2020_paper_61.pdf.

[6] Olson, S., et al., "Automating Biomedical Data Science Through Tree-based Pipeline Optimization." Applications of Evolutionary Computation 19, 123 (2016). https://doi.org/10.1007/978-3-319-31204-0_9.

[7] Płońska, A., et al., "MLJAR: State-of-the-art Automated Machine Learning Framework for Tabular Data. Version 0.10.3." MLJAR (2021). https://github.com/mljar/mljar-supervised.

[8] Conrad, F., et al., "Benchmarking AutoML for Regression Tasks on Small Tabular Data in Materials Design." Scientific Reports 12, 19350 (2022). https://doi.org/10.1038/s41598-022-23327-1.

[9] Shang, Z., et al., "FLAML: A Fast and Lightweight AutoML Library." 4th MLSys Conference (2021). https://doi.org/10.48550/arXiv.1911.04706.

[10] Ryzhkov, A., et al., "LightAutoML: AutoML Solution for a Large Financial Services Ecosystem." arXiv:2109.01528 (2021). https://doi.org/10.48550/arXiv.2109.01528.

[11] Akalin, A., "Computational Genomics With R." Chapman and Hall/CRC, Vol. 1, Section 3.3, (2020). https://compgenomr.github.io/book/relationship-between-variables-linear-models-and-correlation.html.

[12] Pedregosa, F., et al., "Scikit-learn: Machine Learning in Python." Journal of Machine Learning Research 12, 2825 (2011). https://doi.org/10.48550/arXiv.1201.0490.

[13] Chen, T., et al., "XGBoost: A Scalable Tree Boosting System." 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2016). https://doi.org/10.1145/2939672.2939785.

[14] Ke, G., et al., "LightGBM: A Highly Efficient Gradient Boosting Decision Tree." Advances in Neural Information Processing Systems 30, 3149 (2017). https://doi.org/10.48550/arXiv.1706.08359.

[15] Prokhorenkova, L., et al., "CatBoost: Unbiased Boosting with Categorical Features." Advances in Neural Information Processing Systems 32, 6639 (2018). https://doi.org/10.48550/arXiv.1706.09516.

[16] Head, T. et al., "Scikit-Optimize: Sequential Model-Based Optimization." Zenodo (2020). https://doi.org/10.5281/zenodo.1170575.

[17] Ms. Nancy Al Aswad, "Lung Cancer" (dataset). https://kaggle.com/datasets/nancyalaswad90/lung-cancer.

[18] Mansoor Iqbal, "Chronic KIdney Disease Dataset" (dataset). https://kaggle.com/datasets/mansoordaku/ckdisease.

[19] UCI Machine Learning, "Pima Indians Diabetes Dataset" (dataset). https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database.

[20] Izzet Turkalp Akbasli, "Brain stroke prediction dataset" (dataset). https://www.kaggle.com/datasets/zzettrkalpakbal/full-filled-brain-stroke-dataset.

[21] fedesoriano, "Stroke Prediction Dataset" (dataset). www.kaggle.com/fedesoriano/stroke-prediction-dataset.

[22] Rob Mulla, "Eye State Classification EEG Dataset" (dataset). https://www.kaggle.com/datasets/robikscube/eye-state-classification-eeg-dataset.

[23] SmartDx Website. https://smartdx.vercel.app/.

[24] Github Repository. https://github.com/ericspring08/SMEML-Paper.