JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Analysis on Automated Machine Learning Applications

[1]Dr. Prashant Dixit, [2]Dr. Kumud, [3]Ankit Upadhyay
[1]Assistant Professor, [2]Assistant Professor, [3]Assistant Professor
Department of Computer Science & Engineering
[1]Mewar University, Rajasthan, India, [2]D.S. College, Aligarh, U.P., [3]D.S. College, Aligarh, U.P. India

Abstract:

In today's era with the evolution of science & technology, there is an enlarge number of assignments which use Machine Learning algorithms for evaluation and analysis. The use of machine learning requires professional knowledge and some of the processes are time consuming. AutoML (Automated Machine Learning) is developed for automating the task, while the aims to save the time for other valuable procedures such as data-collection and evaluation. In this paper, we will introduce some open source AutoML and them. The innovation of this analysis is the use of ensemble learning to improve or optimize the AutoML frameworks.

Introduction:

Machine Learning is one of the most popular fields of Artificial Intelligence [01]. With the extensive research and investigation on machine learning algorithms, it becomes more popular to build and use machine learning models to draw out the values from data. The application of typical machine learning algorithms needs great effort from humans on feature selection, hyperparameter tunning and model construction. Thus, machine learning applications really need human experts [02].

To create a well-organized and structured machine learning model, many professional elements are required. First of all, they need data scientists or expert who have special experience in the industry segment, especially on chosen the right algorithms and tunning the hyperparameters. The process of narrowing down to the best algorithm and tunning the hyperparameters to fit in the case are time-consuming. In order to save time on those time-consuming processes such as feature engineering and hyperparameter optimization, some open-source automated machine learning (AutoML) frameworks are developed. is the process of automating the time-consuming, iterative tasks of machine learning model development. It allows data scientists, analysts, and developers to build ML models with high scale, efficiency, and productivity all while sustaining model quality [03]. AutoML is designed for automating the big data tasks [04]. It tends to automate the maximum number of steps in solving feature-based regression and classification problems with machine learning pipelines [05].

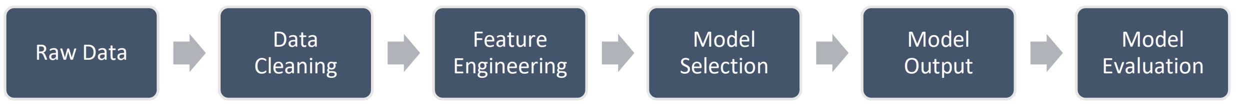


Figure 1: Automated Machine Learning Workflow

Figure 1 shows the workflow of AutoML. With this process data cleaning, feature engineering and model construction and optimization can be done automatically. Users only require to prepare the raw data and evaluate final models by themselves.

Feature Engineering is one of the conclusive steps in machine learning, and good features can help with designing and optimizing better models. Feature Engineering consists of three parts-

A- Feature Selection
B- Feature Extraction
C- Feature Construction

After the feature engineering, we require to choose model and set the hyperparameters. AutoML tries to bring out a model and optimize it automatically.

A- Model Construction
B- Optimization

There are two types of model selection methods, and the first one is typical machine learning model selection while the second one is neural architecture searching (NAS) [06]. In AutoML the hyperparameter optimization is an important process. In hyperparameter a lot of methods are using such as random and grid search, gradient descent and Bayesian optimization [07], Each and every AutoML framework has its own attributes and advantages, while the main aim of the AutoML is to automate the machine learning tasks by an advanced pipeline.

Selected AutoML Frameworks: There are some following frameworks-

1- H2O AutoML
2- Auto-SKLearn
3- TransmogrifAI
4- TPOT
5- MLBox
6- Auto-Keras

1-H2O AutoML:

This is a distributed and open-source machine learning framework which includes a number of machine learning algorithms. This is developed by Java but supports R and Python and supervised tasks can only be acceptable types for this framework. Both classification and regression tasks can be done with H2O framework. H2O automates some machine learning tasks and complex data science, such as feature engineering, model validation, model adjustment, model deployment, and model selection. In addition, it provides automatic visualization and machine learning interpretation (MLI).

2- Auto-SKLearn:

Auto-SKLearn is an automated machine learning software package built on scikit-learn. This frees a machine learning user from algorithm selection and hyper-parameter tuning. This is a wrapper of scikit-learn framework that automatically chooses suitable machine learning algorithm and feature pre-processing steps with automatic hyperparameters tunning. This is based on efficient Bayesian optimization method to create pipelines.

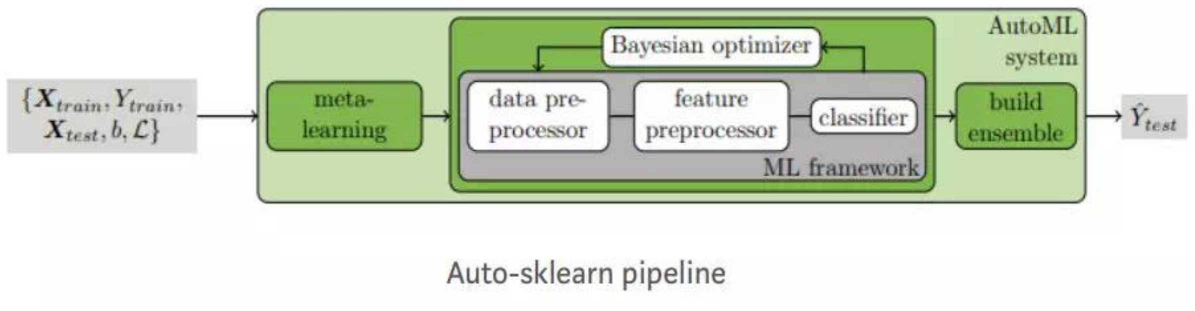


Figure 2: Auto-SKLearn Workflow

Figure 2 shows the Auto-SKLearn pipeline. These two main components are connected for hyperparameter tuning by means of Bayesian reasoning: meta-learning is used to optimizer and evaluate the auto collection construction during the process. Auto-SKLearn performs well in small and medium datasets, but it cannot produce modern deep learning systems with the most advanced performance in large datasets.

3- TransmogrifAI:

Einstein, a flagship Machine Learning platform of Salesforce, is also powered by TransmogrifAI. This is an end-to-end AutoML library for structured data written in Scala that runs on top of Apache Spark. It automates feature analysis, feature selection, feature validation, model selection and more. It was developed with a focus on accelerating machine learning developer productivity through machine learning automation, and an API that enforces compile-time type-safety, modularity, and reuse. This is especially useful in the following scenarios:

- Quickly train quality ML models with minimal manual adjustment
- Construct modular, reusable, and strongly-typed ML workflows

4-TPOT:

TPOT is a tree-based pipeline optimization tool that uses genetic algorithms to optimize machine learning pipelines. It is an update on typical scikit-learn frame7 work that some parts of machine learning process are automatic. The grey part of Figure 3[08] shows the steps which are automated by the Tree-Based Pipeline Optimization Tool (TPOT).

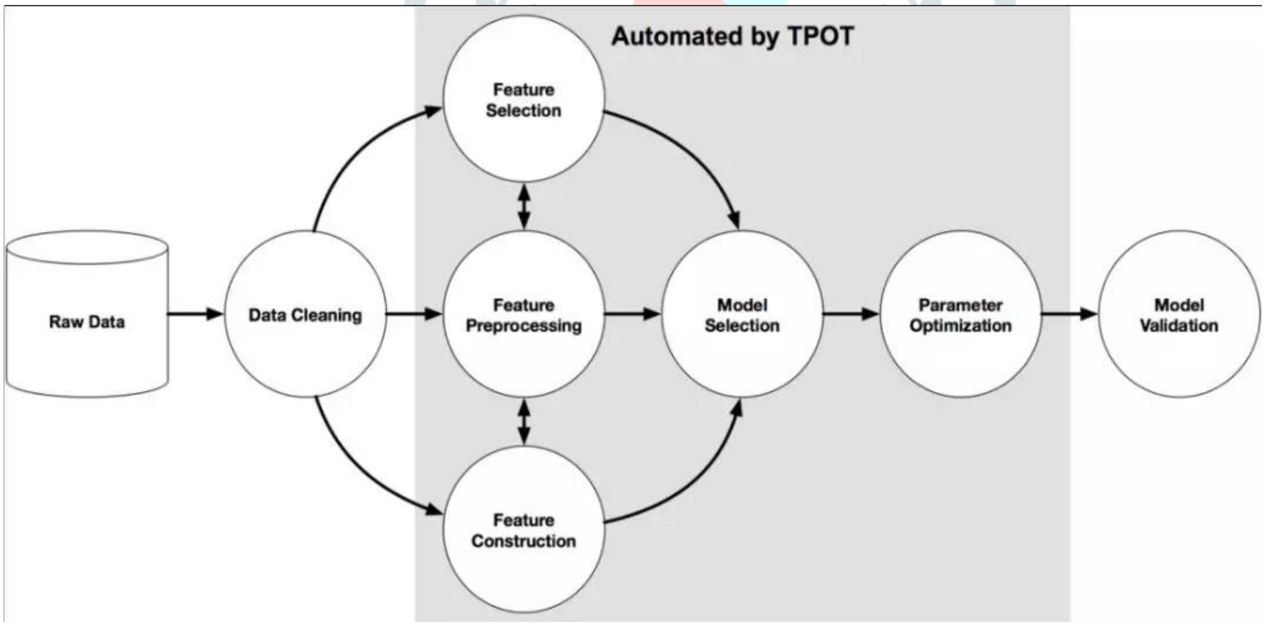


Figure 3: Machine Learning Process Automated by TPOT

This implements four main classes of pipeline operators: Pre-processors, Decomposition, Feature Selection and Models. TPOT does not accept natural language inputs such as categorical string and it does not have the ability to deal with missing value problems, and. Therefore, we need missing value imputation and encoding before inputting the raw data.

5- MLBox:

MLBox is a powerful Automated Machine Learning python library. It has fast read and distributed data formatting, cleaning and preprocessing. It has been tested on Kaggle and shows good result, this is also state of art predictive model for classification and regression. The MLBox is highly robust lead detection and feature selection framework and it is well correct hyper parameter optimization.

MLBox architecture main package contains three sub packages:

A- Preprocessing: study and preprocessing row data.
B- Optimization: testing and optimize huge range of users.
C- Prediction: forecast the aim on a test dataset.

6- Auto-Keras:

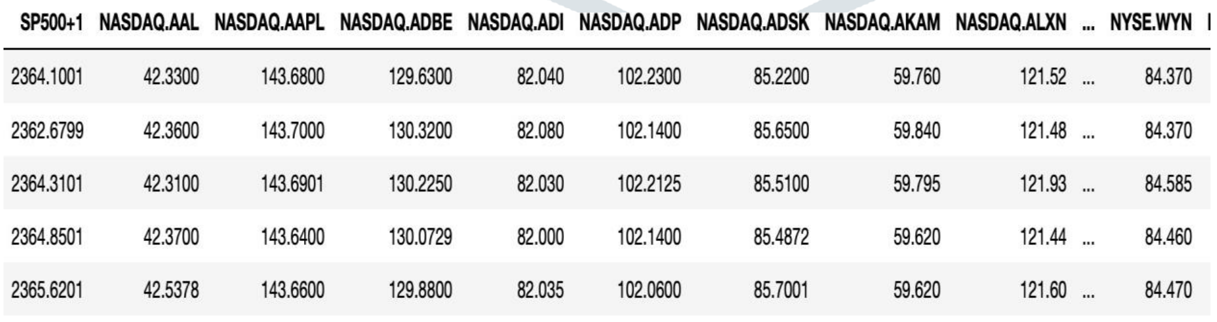
Auto-Keras [09] is an open-source AutoML package based on Keras and developed by DATA Lab. The Auto-Keras is easy to use because follows the classic Scikit Learn API design. It provides automatically search for hyper parameters in the course of deep learning. This framework only supported Python3.6 in Linux.

Data Collection and Evaluation:

Data Resources:

The standard & Poor's 500 Index, or S&P 500 Index, is a market capitalization weighted index of 500 public-traded companies. All the companies included in S&P 500 Index were listed on American exchange such as New York Stock Exchange and National Association of Securities Dealers Automated Quotation. Compared with Dow Jones Industrial Average, S&P 500 Index includes more companies that the risk is more diversifiable. Therefore, S&P 500 can reflect the change of the whole market better. The S&P 500 Index was compiled by Standard Poor's Financial Services LLC in 1957. The constituent stock consisted of 425 industrial stocks, 15 railway stocks and 60 public utility stocks in the beginning. From July 1, 1976, the components of constituent stock were changed to 400 industrial stocks, 20 transportation stocks, 40 public utility stocks and 40 financial stocks. The S&P 500 Index has the characteristics of wide sampling, strong representativeness, high accuracy and good continuity. It is regarded as an ideal target for stock index. Therefore, S&P 500 Index is a representative financial index.

Our dataset is extracted the prediction is the S&P 500 Index. The following chart is a preview of our dataset. The first column is the predictor which is the S&P 500 Index. The rest 500 columns are the values of 500 stocks included in S&P 500 Index.from STATWORX. The dataset consists of 41266- minute data of 500 stocks and S&P 500 Index, and the range of the dataset is wide. We will use the values of 500 stocks included in S&P 500 Index at this minute as the predictor to predict the S&P 500 Index next minute, while



SP500+1	NASDAQ.AAL	NASDAQ.AAPL	NASDAQ.ADBE	NASDAQ.ADI	NASDAQ.ADP	NASDAQ.ADSK	NASDAQ.AKAM	NASDAQ.ALXN	...	NYSE.WYN
2364.1001	42.3300	143.6800	129.6300	82.040	102.2300	85.2200	59.760	121.52	...	84.370
2362.6799	42.3600	143.7000	130.3200	82.080	102.1400	85.6500	59.840	121.48	...	84.370
2364.3101	42.3100	143.6901	130.2250	82.030	102.2125	85.5100	59.795	121.93	...	84.585
2364.8501	42.3700	143.6400	130.0729	82.000	102.1400	85.4872	59.620	121.44	...	84.460
2365.6201	42.5378	143.6600	129.8800	82.035	102.0600	85.7001	59.620	121.60	...	84.470

Model Evaluation Index:

To evaluate the performance of model, there are multiple indexes can be used. For classification problems, receiver operating characteristic (ROC) and area under curve (AUC) can be used to evaluate the performance of the models. Mean squared error (MSE), mean absolute error (MAE) and coefficient of determination (R square) are normally used for regression model evaluation. In our testing, we will use mean squared error and coefficient of determination to evaluate models.

$$\text{MSE}\ (y - y^{\wedge}) = \frac{1}{n}\ \sum_{i=0}^{n-1}(y_i - y^{\wedge}{}_i)^2$$

The formula shows how to calculate MSE. Deviation measures the expectation of error that deviates from the true function or parameter, while variance is a measure of deviation of estimated expectation that any particular sampling of the data may cause. MSE can be used to measure the error that comes from both deviation and variance, which is more comprehensive than deviation and variance.

$$R^2\left(y-y^{\wedge}\right)=1-\frac{\sum_{i=0}^{n-1}\left(y_i-y^{\wedge}{}_i\right)^2}{\sum_{i=0}^{n-1}\left(y_i-y^{-}\right)^2}$$

The formula is to compute coefficient of determination. It is a numerical feature that represents the relationship between a random variable and multiple random variables. Coefficient of determination is a statistical indicator that reflects the reliability of the change of dependent variable in regression models. It represents the proportion of the total variation of the dependent variable that can be explained by the independent variables via the regression model. Mean square error and coefficient of determination are two indexes that we will use to evaluate the performance of each AutoML framework.

Results:

We test the AutoML frameworks with S&P500 Index on Python3.6. In auto-sklearn, we set the process to run for 60 minutes, while the maximum time allocated per model is 10 minutes. The following code will be used to train in auto-sklearn.

```
import sklearn
import auto-sklearn
from autosklearn.regression import AutoSklearnRegressor auto_model = Auto
SklearnRegressor(time_left_for_this_task=3600,
                                         per_run_time_limit=600)
auto_model.fit(X_train, y_train)
```

In autoML, we test both Gradient Boosting Regressor and Random Forest Regressor with 10-fold. The difference between Random Forest Regressor and Gradient Boosting Regressor is preform feature selection. Random Forest Regressor only retains a specific number of important features, while Gradient Boosting Regressor takes all the features into account. Since autoML cannot be stopped early, the running time is extremely long for the large dataset. The following code is for prediction in autoML. Gradient Boosting Regressor is used as default regression method, while model names= 'RandomForestRegressor' can be used to change the regression method.

```
import auto_ml
from auto_ml import Predictor
from auto_ml.utils_models import load_ml_model
column_descriptions = {
      'target_value': 'output'
}
ml_predictor = Predictor(type_of_estimator='regressor',
                                         column_descriptions=
                                         column_descriptions)
ml_predictor.train(train_value)
```

In TPOT, we set the maximum running time to be 60 minutes and train it with different scoring functions such as mean squared error and R square. Since the maximum running time is not enough to finish the whole process, TPOT is stopped in an early generation. The following code is the example with mean squared error as the scoring function for regression.

```
from tpot import TPOTRegressor
tpot = TPOTRegressor(generations=100,population_size=20,verbosity
                                            =2,scoring=‘ neg_mean_squared_error’,
                                            max_time_mins=60)
tpot.fit(X_train,y_train)
```

In H2O, we set 6 as the maximum number of models to be tested and 60 minutes as the maximum running time. The data frame trained in h2o is special, while the file should be read by h2o command. The following code is the example for regression in h2o.

```
import h2o
from h2o.automl import H2OAutoML  h2o.init()
data=h2o.import_file(‘FILE_NAME.csv’)
data_train, data_test = data.split_frame(ratios=[0.8]) aml = H2
OAutoML(max_runtime_secs = 3600,max_models = 6,
                      max_runtime_secs_per_model = 600)
aml.train(x = ‘train_value’, y = ‘target_value’, training_frame =
                                                 data_train)
```

Auto-Keras is the only framework based on deep learning, and this framework has to be trained on Linux. Structured Data Regressor will be used to train the regression problems. Since Auto-Keras is based on neural network architecture, the computation budget is high. The following code is for regression in Auto-Keras.

```
from autokeras import StructuredDataRegressor
search = StructuredDataRegressor(max_trials = 15, loss = ‘
                                            mean_squared_error’)
search.fit(x=X_train,y=y_train,verbose=0)
```

The results of best pipeline from each framework are shown below. From the results of the testing, we can see that all AutoML frameworks preforms well on the S&P500 Index prediction. The models have the coefficient of determination at around high-90%, which means about high-90% of the change in S&P500 Index is associated with the change in the 500 companies included in S&P 500 Index. Due to the limitation on computation budget, the testing results of some frameworks are not as good as others.

AutoML Framework	MSE	R square
Auto-sklearn	0.516327	0.999620
AutoML	0.435640	0.999746
TPOT	0.505464	0.999648
H2O	0.465780	0.999704
Auto-Keras	0.443213	0.999737

AutoML performs the best; however, if we take the computation budget into consideration, autoML is not the best choice. Since autoML cannot be stopped early, the running time for our test is extremely long. The results show that AutoML frameworks are useful for some financial prediction tasks.

A Review of Ensemble Learning for Optimization:

In supervised machine learning job, our aim is to learn a stable model that performs better in all senses; however, the actual condition is not ideal. Sometimes we can only get multiple models with preferences. Ensemble learning is to combine multiple weak supervised models to obtain a more comprehensive strong-supervised model. The idea of ensemble learning is that even if a classifier gets a wrong prediction, other classifiers can also correct the error. Ensemble learning can be used for classification problems, regression problems, feature selection, etc. It has good strategies on dataset with multiple sizes. With large dataset, ensemble learning splits it into multiple small datasets and train them respectively. Finally, all the models will be combined. On the contrary, ensemble learning will use Bootstrap on small dataset to sample to obtain multiple datasets for training.

There are three classes for ensemble learning:

A- Bagging
B- Boosting
C- Stacking

Future Work:

1- **Feature Engineering:** Feature engineering is the process of selecting, manipulating, and transforming raw data into features that can be used in supervised learning. In order to make machine learning work well on new tasks, it might be necessary to design and train better features. As you may know, a “feature” is any measurable input that can be used in a predictive model. In other words, we can say **is the act of converting raw observations into desired features using statistical or machine learning approaches.** It is a machine learning technique that leverages data to create new variables that aren’t in the training set. It can produce new features for both supervised and unsupervised learning, with the goal of **simplifying and speeding up data transformations** while also **enhancing model accuracy**.

2- **Neural Network Architecture:** Neural Networks are the functional unit of Deep Learning and are known to mimic the behaviour of the human brain to solve complex data-driven problems. The input data is processed through different layers of artificial neurons stacked together to produce the desired output. From speech recognition and person recognition to healthcare and marketing, Neural Networks have been used in a varied set of domains. Neural network architecture may perform much better on larger amount of raw data; however, it has greater computation budget than typical machine learning algos. In the future, we may try to find some other new tools to improve the efficiency of neural network architecture and use it on AutoML frameworks.

Conclusion:

AutoML is the fact of clarifying and simplifying data science projects by automating the machine learning jobs. It means to reduce the time consumption by automation with the help of machine learning pipeline. AutoML can be thought as a standard machine learning process with automating three main parts which are data pre-processing, design and optimization. AutoML frameworks reduce grubby works on analysis of raw data, so data scientists or experts can spend less time on filtering that data, feature engineering and hyperparameter tunning, while they can spend more time on experimenting with model architectures.

In this study, we tested five AutoML frameworks and tried to use stacking to optimize the performance. By testing the five existing frameworks, Auto-sklearn, AutoML autoML, H2O and Auto-Keras, TPOT, on S&P 500 Index prediction task, we found that the application of AutoML frameworks on financial raw data analysis is well organized. The innovation of our work is the betterment of existing AutoML frameworks created by ensemble learning. With the use of ensemble learning, the analysis is more precise. It shows the idea that the use of ensemble learning is a well-structured way to improve accuracy and save time. With the use of ensemble learning, the analysis is more precise.

References:

[01] Mitchell, T. M. Machine Learning. McGraw-Hill, 2003.

[02] Yao, Q. et al. “Taking Human out of Learning Applications: A Survey on Automated Machine Learning.” (2018).

[03] https://docs.microsoft.com/en-us/azure/machine-learning/concept-automated-ml

[04] Nickson, T. , et al. “Automated Machine Learning on Big Data using Stochastic Algorithm Tuning.” Eprint Arxiv (2014).

[05] Guyon, I. , et al. “Design of the 2015 ChaLearn AutoML challenge.” International Joint Conference on Neural Networks IEEE, 201

[06] Elsken, T. , J. H. Metzen , and F Hutter.“Neural Architecture Search: A Survey.” arXiv (2018).

[07] u, E. , V. M. Cora , and N. D. Freitas .“A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning.” Computer Science (2010).

[08] Balaji, A. , and A. Allen .“Benchmarking Automatic Machine Learning Frameworks.” (2018).

[09] Breiman, L. .“Bagging Predictors.” Machine Learning 24.2(1996):123-140.

[10] Feurer, M. , et al.“Efficient and robust automated machine learning.” Advances in neural information processing systems 28(2015):2944-2952.

[11] Zhang, W. , et al. “Neural Feature Search: A Neural Architecture for Automated Feature Engineering.” 2019 IEEE International Conference on Data Mining (ICDM) IEEE, 2019

[12] Zhang, W. , et al. “Genetic Programming.” (2013).