earn (AutoML) For Machine Learning

ɔming a regular thing for machine learning practitioners. People often
ɘplace data scientists?

Not really. If you're eager to find out what AutoML is and how it works, join me in this article. I'm going to show you auto-sklearn, a state-of-the-art and open-source AutoML framework.

To do this, I had to do some research:

- Read the first and second paper for auto-sklearn V1 and V2.
- Took a deep dive into the auto-sklearn documentation and examples.
- Checked the official Auto Sklearn blog post.
- Did some experiments on my own.

As do AutoML research, and I've learned quite a lot so far. After reading this post, you'll know more about:

- What is AutoML, and who is AutoML for?
- Why does auto-sklearn matter to the ML community?
- How to use auto-sklearn in practice?
- What are the main features of auto-sklearn?
- A use-case of auto-sklearn with result tracking in Neptune.

Table of contents

Let's start!

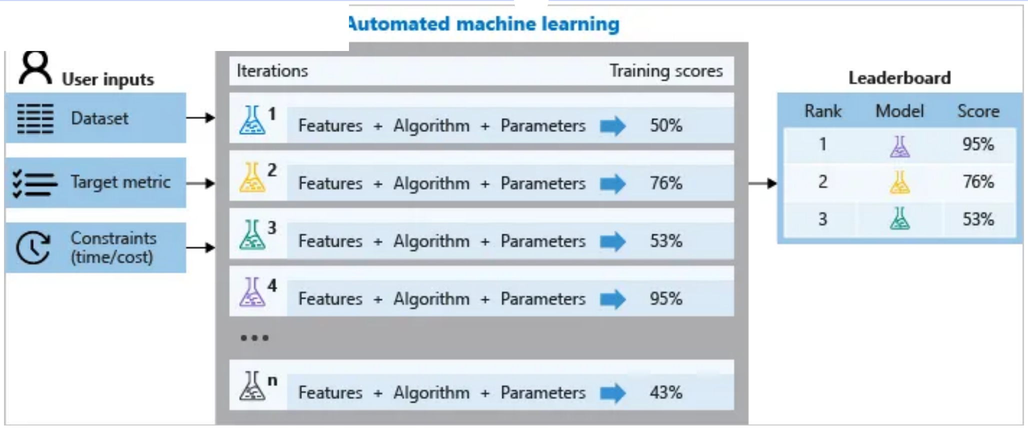
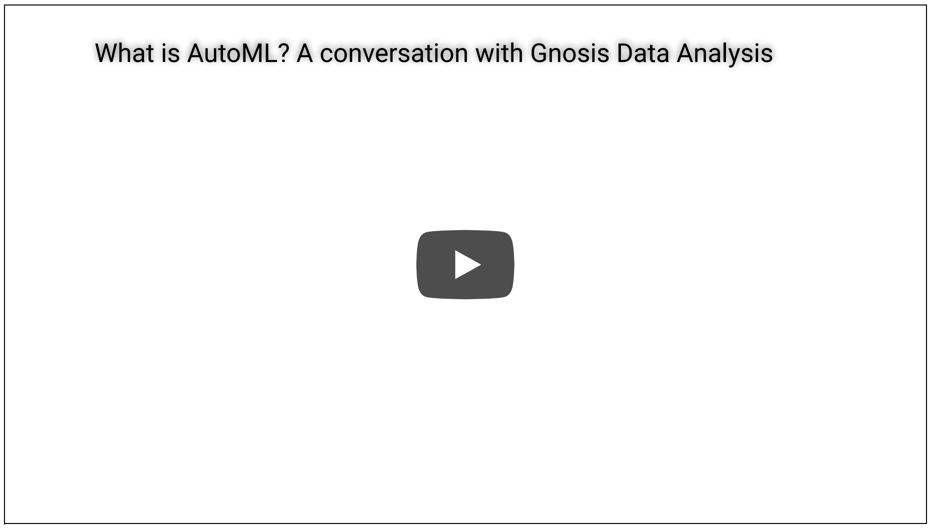

Automated machine learning
User inputs
Dataset
Target metric
Constraints (time/cost)
Iterations
Training scores
1 Features + Algorithm + Parameters 50%
2 Features + Algorithm + Parameters 76%
3 Features + Algorithm + Parameters 53%
4 Features + Algorithm + Parameters 95%
n Features + Algorithm + Parameters 43%
Leaderboard
Rank Model Score
1 95%
2 76%
3 53%

Image from Microsoft Azure AutoML doc

AutoML can improve the quality of work for data scientists, it's not going to remove data scientists from the cycle.

Experts could use AutoML to increase their job performance by focusing on the best-performing pipelines, and non-experts could use AutoML systems without a broad ML education. If you have 15 minutes to spare, the conversation below might help you understand what AutoML is all about.



What is AutoML: A conversation between Josh Starmer and Ioannis Tsamardinos

AutoML frameworks

There are different types of AutoML frameworks, each has unique features. Each of them has automated a few steps of a full machine learning workflow, from pre-processing to model development. In this table, I summed up only a few of them that are worth mentioning:

Name of AutoML framework	Created by	documentation/GitHub	Open-Source
auto-sklearn	Matthias Feurer, et al.	GitHub / Documentation	Yes
Auto-xgboost	Janek Thomas et al.	GitHub	–
GCP-Tables	Google	Source	No
AutoGluon	Amazon	GitHub/ Source	Yes
AutoML-azure	Microsoft	Source	No
GAMA	Pieter Gijsbers et al.	GitHub/ source	Yes
Auto-WEKA	Chris Thornton et al.	GitHub/ Documentation	Yes
H2O AutoML	h2o.ai	GitHub	Yes
TPOT	Randal S. Olson, et al.	GitHub/ Documentation	Yes
ML-Plan	Marcel Wever et al.	GitHub/ Documentation	Yes
hyperopt-sklearn	Brent Komer et al	GitHub/ Documentation	Yes
SmartML	Mohamed Maher et al.	GitHub	Yes
MLJAR	MLJAR Team	GitHub/ Documentation	Yes

d techniques wh... helped the creators win the first and second international AutoML challenge.

auto-sklearn is based on defining AutoML as a CASH problem.

CASH = Combined Algorithm Selection and Hyperparameter optimization. Put simply, we want to find the best ML model and its hyperparameter for a dataset among a vast search space, including plenty of classifiers and a lot of hyperparameters. In the figure below, you can see a representation of auto-sklearn provided by its authors.

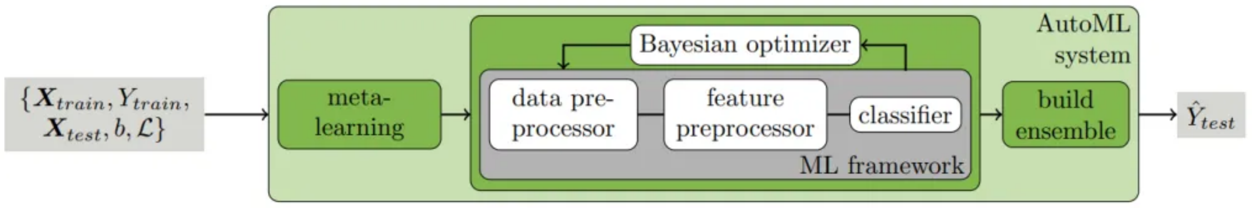


Image from the auto-sklearn paper

auto-sklearn can solve classification and regression problems. The first version of auto-sklearn was introduced with an article titled "Efficient and robust automated machine learning " in 2015, at the 28th International Conference on Neural Information Processing Systems. The second version was presented with the paper "auto-sklearn 2.0: The Next Generation" in 2020.

Auto-sklearn features

What can auto-sklearn do for users? It has several valuable features, helpful for both novices and experts.

By writing just five lines of Python code, beginners can see the prediction, and experts can boost their productivity. Here are some main features of auto-sklearn:

- Written in Python, on top of the most popular ML library (scikit-learn).
- Useful for many tasks, such as classification, regression, multi-label classification.
- Consists of several preprocessing methods (handling missing values, normalizing data).
- Searches for optimal ML pipelines among a considerable search space (15 classifiers, more than 150 hyperparameters are searched).
- State of the art thanks to using meta-learning, Bayesian optimization, ensemble techniques.

How does auto-sklearn work?

Auto-sklearn can solve classification and regression problems, but how? There's a lot that goes into a machine learning pipeline. In general, auto-sklearn V1 has three main components:

1. Meta-learning
2. Bayesian optimization
3. Build ensemble

So when we want to apply a classification or regression on a new dataset, auto-sklearn starts by extracting its meta-

Recently the second version of auto-sklearn went public. Let's review what's changed in the new generation. Based on the official blog post and original paper, there are four improvements:

- They allowed each ML pipeline to use an early-stopping strategy inside the whole search space; this feature improved performance on large datasets, but it's mostly useful for tree-based classifiers.
- Improving model selection strategy: one vital step in auto-sklearn is how to select models. In auto sklearn V2, they used a multi-fidelity optimization method such as BOHB. However, they showed that a single model selection is not fit for all types of the problem, and they integrated several strategies. To get familiar with new optimization methods, you can read this article: "HyperBand and BOHB: Understanding State of the Art Hyperparameter Optimization Algorithms."
- Building a portfolio instead of using meta-feature to find a similar dataset in the knowledge base. You can see this improvement in the image below.
- Build an automated policy selection on top of the previous improvements to select the best strategy.

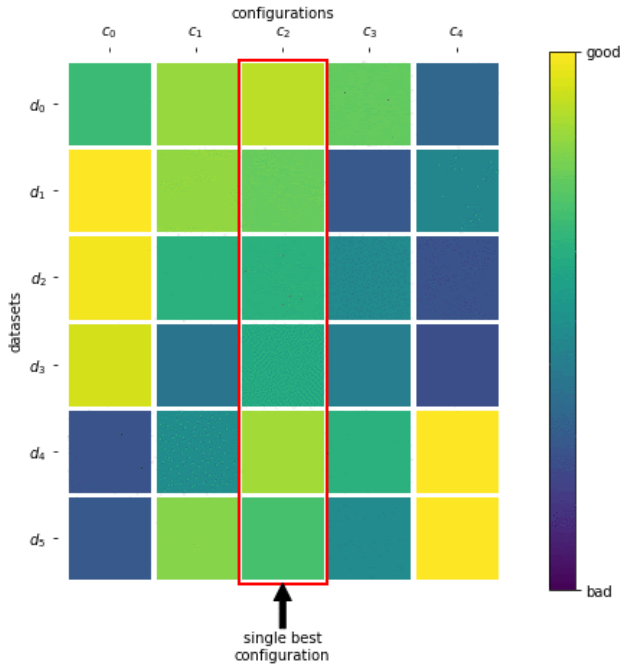


Image from automl.org

Auto-sklearn main parameters

Although Auto-sklearn might be able to find an outperforming pipeline without setting any parameters, there are some parameters that you can use to boost your productivity. To check all parameters visit the official page.

Parameter name	Default value	Description

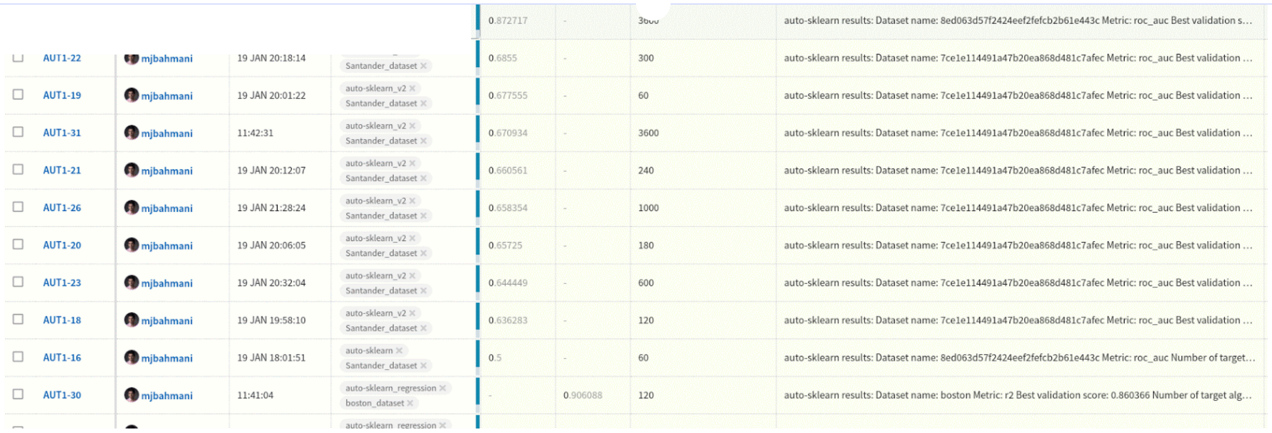
per_run_time_limit	None	It shows how many seconds each ML model should spend.
initial_configurations_via_metalearning	25	How many configurations via meta-learning considers hyperparameter optimization. If set 0, this option will be inactive. Also, this parameter is not available in the auto-sklearn V2.
ensemble_size	50	The number of the models in the ensemble. To disable this parameter, set it to 1.
n_jobs	1	The number of parallel jobs. For using all processors, set it -1
ensemble_nbest	50	Number of best models for building an ensemble model. Only works when ensemble_size is more than one.
include_estimators	None	It will use all estimators when there is None. Not available in auto-sklearn V2.
exclude_estimators	None	You can exclude some estimators from the search space. Not available in auto-sklearn V2.
Metric	None	If you don't define a metric, it will be selected based on the task. In this article, we define it (autosklearn.metrics.roc_auc).

Now let's apply what we learned in a case-study, and perform some experiments!

Track Auto-sklearn experiments on Neptune

I made some Notebooks which you can easily download and do the experiments on your own. But to do all the steps together again, you need to:

- Install auto-sklearn on your machine or Google Colab.
- Select a 10000 sample from Santander Customer Transaction Prediction for experimenting with binary classification.
- Use the Boston dataset for performing an experiment based on a regression problem.
- Do the experiments.
- Track the results in Neptune.



				0.872717	-	[illegible]	auto-sklearn results: Dataset name: 8ed063d57f2424eef2fefcb2b61e443c Metric: roc_auc Best validation s...
AUT1-22	mjbahmani	19 JAN 20:18:14	Santander_dataset	0.6855	-	300	auto-sklearn results: Dataset name: 7ce1e114491a47b20ea868d481c7afec Metric: roc_auc Best validation ...
AUT1-19	mjbahmani	19 JAN 20:01:22	auto-sklearn_v2 Santander_dataset	0.677555	-	60	auto-sklearn results: Dataset name: 7ce1e114491a47b20ea868d481c7afec Metric: roc_auc Best validation ...
AUT1-31	mjbahmani	11:42:31	auto-sklearn_v2 Santander_dataset	0.670934	-	3600	auto-sklearn results: Dataset name: 7ce1e114491a47b20ea868d481c7afec Metric: roc_auc Best validation ...
AUT1-21	mjbahmani	19 JAN 20:12:07	auto-sklearn_v2 Santander_dataset	0.660561	-	240	auto-sklearn results: Dataset name: 7ce1e114491a47b20ea868d481c7afec Metric: roc_auc Best validation ...
AUT1-26	mjbahmani	19 JAN 21:28:24	auto-sklearn_v2 Santander_dataset	0.658354	-	1000	auto-sklearn results: Dataset name: 7ce1e114491a47b20ea868d481c7afec Metric: roc_auc Best validation ...
AUT1-20	mjbahmani	19 JAN 20:06:05	auto-sklearn_v2 Santander_dataset	0.65725	-	180	auto-sklearn results: Dataset name: 7ce1e114491a47b20ea868d481c7afec Metric: roc_auc Best validation ...
AUT1-23	mjbahmani	19 JAN 20:32:04	auto-sklearn_v2 Santander_dataset	0.644449	-	600	auto-sklearn results: Dataset name: 7ce1e114491a47b20ea868d481c7afec Metric: roc_auc Best validation ...
AUT1-18	mjbahmani	19 JAN 19:58:10	auto-sklearn_v2 Santander_dataset	0.636283	-	120	auto-sklearn results: Dataset name: 7ce1e114491a47b20ea868d481c7afec Metric: roc_auc Best validation ...
AUT1-16	mjbahmani	19 JAN 18:01:51	auto-sklearn Santander_dataset	0.5	-	60	auto-sklearn results: Dataset name: 8ed063d57f2424eef2fefcb2b61e443c Metric: roc_auc Number of target...
AUT1-30	mjbahmani	11:41:04	auto-sklearn_regression boston_dataset	-	0.906088	120	auto-sklearn results: Dataset name: boston Metric: r2 Best validation score: 0.860366 Number of target alg...
			auto-sklearn_regression				

Check all the experiments in Neptune

First, you need to install auto-sklearn on your machine. Simply use pip3 for this:

```
pip3 install auto-sklearn
```

If you get an error, you may need to install dependencies for that, so please check the official installation page. You can also use the notebooks I prepared for you in Neptune. Then run the following command to make sure the installation is done correctly:

```
import autosklearn
print(autosklearn.__version__)
#  0.12.1
```

Let's tackle some classification and regression problems.

Auto-sklearn for classification

For the classification problem, I chose a cherished Kaggle competition – Santander Customer Transaction Prediction. Please download the dataset and select 10000 records randomly. Then follow the experiments in the first notebook:

```
y_val=None
train=pd.read_csv("./sample_train_Santander.csv")
X=train.drop(["ID_code",'target'],axis=1)
y=train["target"]
X_train,X_val,y_train,y_val = train_test_split(X,y, stratify=y,test_size=0.33,
random_state=42)
#define the model
automl = autosklearn.classification.AutoSklearnClassifier()
#train the model
automl.fit(X_train, y_train )
#predict
y_pred=automl.predict_proba(X_val)
# score
score=roc_auc_score(y_val,y_pred[:,1])
print(score)
# show all models
show_modes_str=automl.show_models()
sprint_statistics_str = automl.sprint_statistics()
```

We also need to define some configurations to gain more insight into auto-sklearn:

Configurations	Range value	Description
time_left_for_this_task	[60- 5000]	I started the experiments with 60 seconds, and then for each experiment, I increased it to 5000.
metric	roc_auc	As the case study is highly imbalance, then I need to change the metric to roc_auc
resampling_strategy	CV	In auto-sklearn V1, If I did not define the resampling_strategy, it could not get a good result. But in auto-sklearn V2, it did it automatically.
resampling_strategy_arguments	{'folds': 5}	

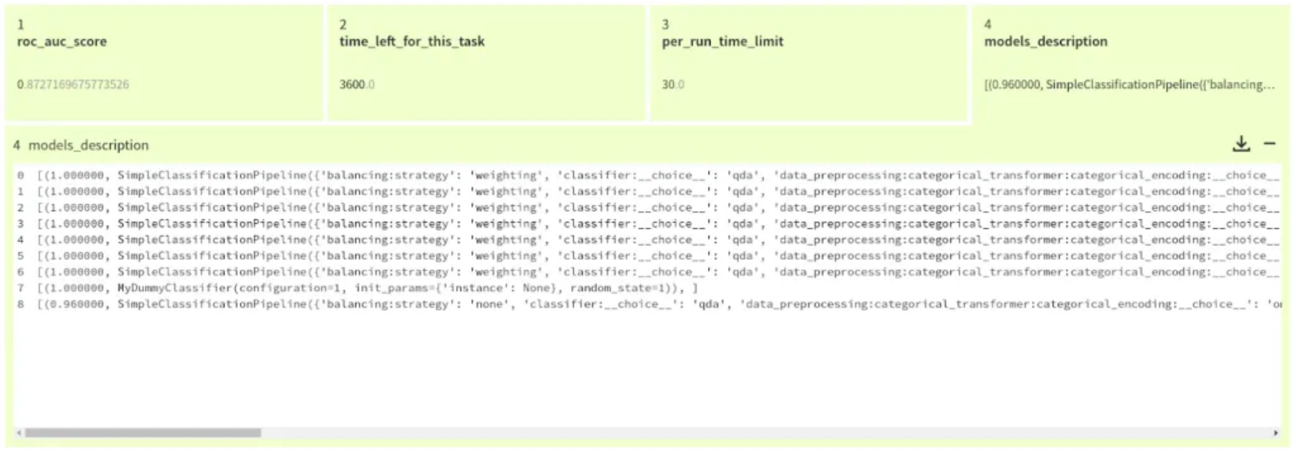
To use the above configuration, you could define the automl object as follows:

```
                                        c_auc,
n_jobs=-1,
resampling_strategy='cv',
resampling_strategy_arguments={'folds': 5},
)

#train the model

automl.fit(X_train, y_train )
```

As I used plenty of different configurations, I just track them on the Neptune. You can see one of them in the image, and check all of them in Neptune.



1 roc_auc_score	2 time_left_for_this_task	3 per_run_time_limit	4 models_description
0.8727169675773526	3600.0	30.0	[(0.960000, SimpleClassificationPipeline({'balancing...

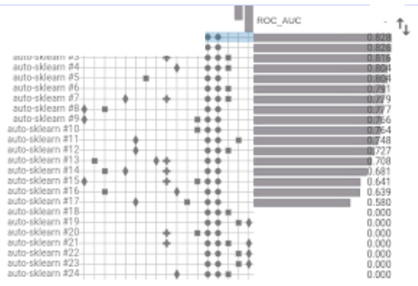
4 models_description

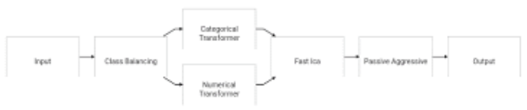
```
0  [(1.000000, SimpleClassificationPipeline({'balancing:strategy': 'weighting', 'classifier:__choice__': 'qda', 'data_preprocessing:categorical_transformer:categorical_encoding:__choice__
1  [(1.000000, SimpleClassificationPipeline({'balancing:strategy': 'weighting', 'classifier:__choice__': 'qda', 'data_preprocessing:categorical_transformer:categorical_encoding:__choice__
2  [(1.000000, SimpleClassificationPipeline({'balancing:strategy': 'weighting', 'classifier:__choice__': 'qda', 'data_preprocessing:categorical_transformer:categorical_encoding:__choice__
3  [(1.000000, SimpleClassificationPipeline({'balancing:strategy': 'weighting', 'classifier:__choice__': 'qda', 'data_preprocessing:categorical_transformer:categorical_encoding:__choice__
4  [(1.000000, SimpleClassificationPipeline({'balancing:strategy': 'weighting', 'classifier:__choice__': 'qda', 'data_preprocessing:categorical_transformer:categorical_encoding:__choice__
5  [(1.000000, SimpleClassificationPipeline({'balancing:strategy': 'weighting', 'classifier:__choice__': 'qda', 'data_preprocessing:categorical_transformer:categorical_encoding:__choice__
6  [(1.000000, SimpleClassificationPipeline({'balancing:strategy': 'weighting', 'classifier:__choice__': 'qda', 'data_preprocessing:categorical_transformer:categorical_encoding:__choice__
7  [(1.000000, MyDummyClassifier(configuration=1, init_params={'instance': None}, random_state=1)), ]
8  [(0.960000, SimpleClassificationPipeline({'balancing:strategy': 'none', 'classifier:__choice__': 'qda', 'data_preprocessing:categorical_transformer:categorical_encoding:__choice__': 'o
```

When you fit the auto-sklearn model, you can check all the best outperforming pipelines with PipelineProfiler (pip install pipelineprofiler). To do that, you need to run the following code:

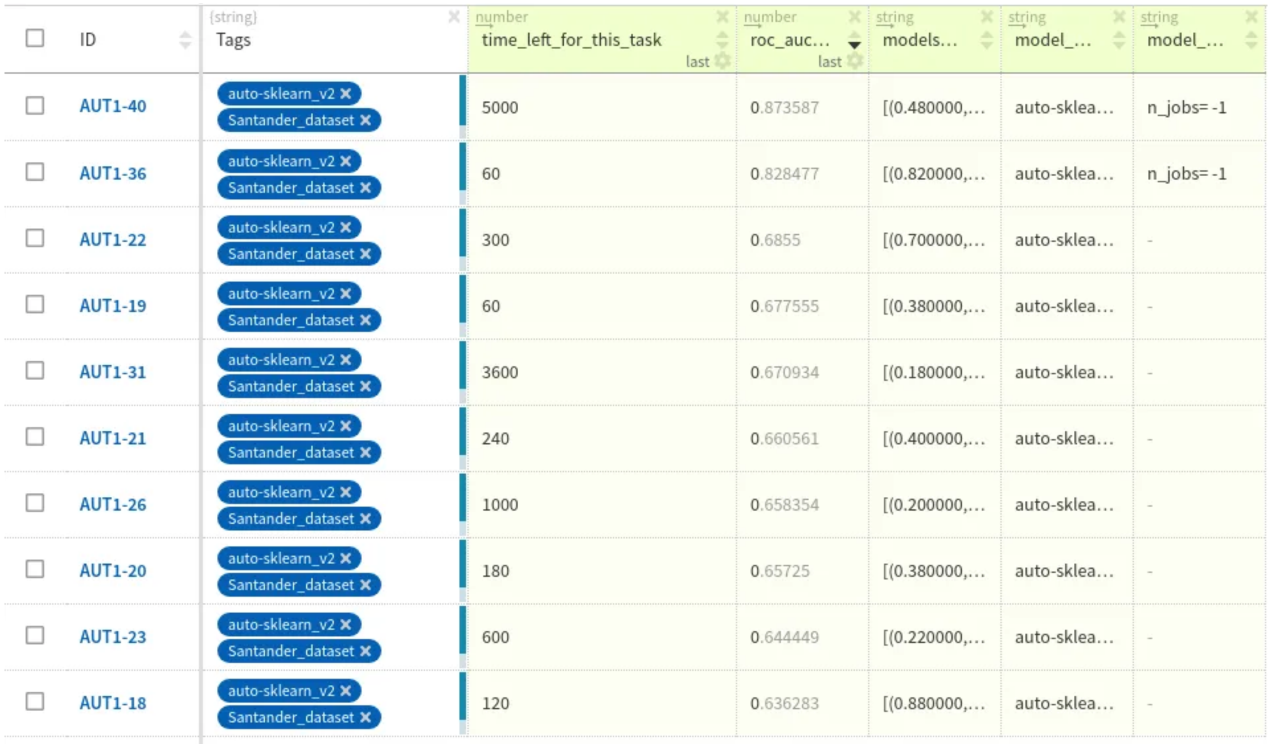
```
import PipelineProfiler
# automl is an object Which has already been created.
profiler_data= PipelineProfiler.import_autosklearn(automl)
PipelineProfiler.plot_pipeline_matrix(profiler_data)
```

Your output should be like this:





On the other hand, I also ran some experiments based on auto-sklearn V2. The result was fascinating. You can see the outcome below:



ID	(string) Tags	number time_left_for_this_task last	number roc_auc... last	string models...	string model_...	string model_...
AUT1-40	auto-sklearn_v2 Santander_dataset	5000	0.873587	[(0.480000,...	auto-sklea...	n_jobs=-1
AUT1-36	auto-sklearn_v2 Santander_dataset	60	0.828477	[(0.820000,...	auto-sklea...	n_jobs=-1
AUT1-22	auto-sklearn_v2 Santander_dataset	300	0.6855	[(0.700000,...	auto-sklea...	-
AUT1-19	auto-sklearn_v2 Santander_dataset	60	0.677555	[(0.380000,...	auto-sklea...	-
AUT1-31	auto-sklearn_v2 Santander_dataset	3600	0.670934	[(0.180000,...	auto-sklea...	-
AUT1-21	auto-sklearn_v2 Santander_dataset	240	0.660561	[(0.400000,...	auto-sklea...	-
AUT1-26	auto-sklearn_v2 Santander_dataset	1000	0.658354	[(0.200000,...	auto-sklea...	-
AUT1-20	auto-sklearn_v2 Santander_dataset	180	0.65725	[(0.380000,...	auto-sklea...	-
AUT1-23	auto-sklearn_v2 Santander_dataset	600	0.644449	[(0.220000,...	auto-sklea...	-
AUT1-18	auto-sklearn_v2 Santander_dataset	120	0.636283	[(0.880000,...	auto-sklea...	-

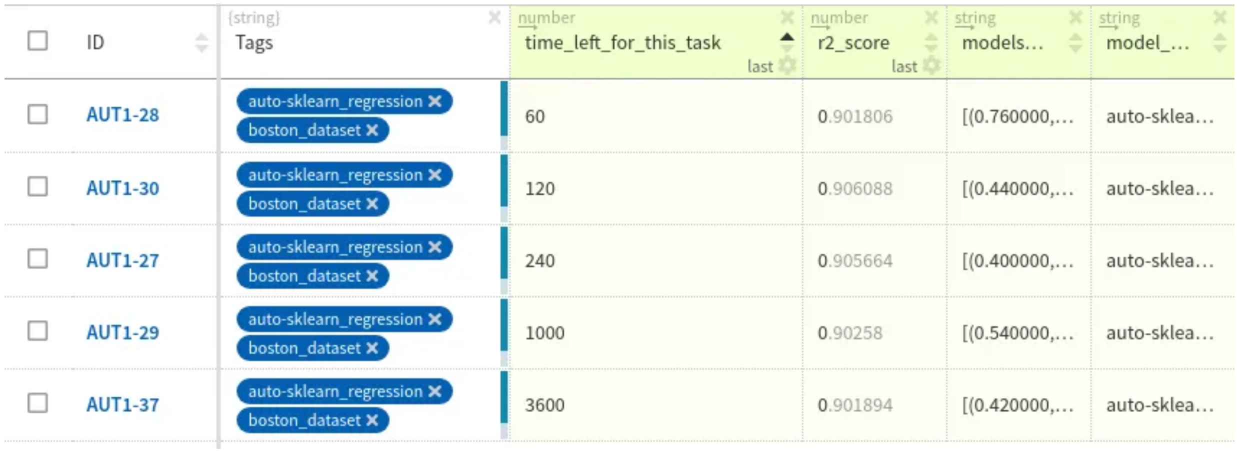
To use auto-sklearn V2, you can use following code:

```
TIME_BUDGET=60
automl = autosklearn.experimental.askl2.AutoSklearn2Classifier(
time_left_for_this_task=TIME_BUDGET,
n_jobs=-1,
metric=autosklearn.metrics.roc_auc,
)
```

```
TIME_BUDGET=60
automl = autosklearn.regression.AutoSklearnRegressor(
time_left_for_this_task=TIME_BUDGET,
n_jobs=-1
)
automl.fit(X_train, y_train, dataset_name='boston')
y_pred = automl.predict(X_test)
score=r2_score(y_test, y_pred)
print(score)
show_modes_str=automl.show_models()
sprint_statistics_str = automl.sprint_statistics()

print(show_modes_str)
print(sprint_statistics_str)
```

I just changed the time budget to track the performance based on the time limitation. The image below shows the results.



ID	{string} Tags	number time_left_for_this_task last	number r2_score last	string models...	string model_...
AUT1-28	auto-sklearn_regression boston_dataset	60	0.901806	[(0.760000,...	auto-sklea...
AUT1-30	auto-sklearn_regression boston_dataset	120	0.906088	[(0.440000,...	auto-sklea...
AUT1-27	auto-sklearn_regression boston_dataset	240	0.905664	[(0.400000,...	auto-sklea...
AUT1-29	auto-sklearn_regression boston_dataset	1000	0.90258	[(0.540000,...	auto-sklea...
AUT1-37	auto-sklearn_regression boston_dataset	3600	0.901894	[(0.420000,...	auto-sklea...

Final thought

Overall, auto-sklearn is still a new technology. Because auto-sklearn is built on top of scikit-learn, many ML practitioners can quickly try it and see how it works.

The most important advantage of this framework is that it saves a lot of time for experts. The one weakness is that it acts as a black box, and doesn't say anything about how to make a decision.

All in all, it's a pretty interesting tool, so it's worth giving auto-sklearn a look.

The Best ML Frameworks & Extensions For Scikit-learn
by Derrick Mwiti, February 9th, 2021

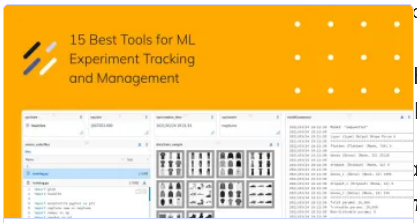
Read more

19 Best JupyterLab Extensions for Machine Learning
by Pawel Kijko, August 20th, 2020

Read more

Hyperparameter Tuning in Python: a Complete Guide 2021
by Shahul ES, August 24th, 2020

Read more



15 Best Tools for ML Experiment Tracking and Management
by Patrycja Jenkner, February 17th, 2020

Read more

ated July 16th, 2021

he union of statistics and computation. The crux of machine learning revolves around the
h are in fact statistical estimations on steroids.

l limitations depending on the data distribution. None of them can be entirely accurate
n *if on steroids)*. These limitations are popularly known by the name

lify by not paying much attention to the training points (e.g.: in Linear Regression,
nodel will always assume a linear relationship).

ict itself to the training data by not generalizing for test points that it hasn't seen before
= None).

are subtle, like when we have to choose between a random forest algorithm and a
en two variations of the same decision tree algorithm. Both will tend to have high variance

This is where model selection and model evaluation come into play!

In this article we'll talk about:

- What are model selection and model evaluation?
- Effective model selection methods (resampling and probabilistic approaches)
- Popular model evaluation methods
- Important Machine Learning model trade-offs

Top MLOps articles from our blog in your inbox every month.

Continue rea

Type your email here

Get Newsletter

GDPR compliant. Privacy policy.