AUTOML General Introduction

As artificial intelligence is more popular than before, the most feasible way to solve real world problem is to use Machine Learning to learn the **inherent** logic of data that we have, but there are really some **magic things** that we really don’t know how to get insight how the deep learning works and the whole process for machine learning is complex than just with traditional programming step, so there are some **problems** that we should solve to get a **robust model**: 1. as there are cycles in machine learning like if you don’t make the feature engineering step great, than no matter what algorithms we choose to use, it wouldn’t get a great result even we use more powerful algorithms; 2. As we change our data from 10,000 to 10, 000, 000, the data distribution is really not alike than before, as we have to re-do the feature engineering step again; 3. There are really too many algorithms and feature engineering step we could take, with the combination will be more!

So how could we solve the problem better? This comes out with AUTOML **without manually** do feature engineering and choose the algorithms, the framework will choose the best way to combine with different algorithms and solutions.

Suppose you have a dataset with 100,000 samples and 100 raw features columns for 10 classes, if you want to try with 10 algorithms for this problem, each algorithm with 10 hyper-parameters tuning, then you may need **100 models** to try to train, with each model you will have to write code to train, there are so **many codes** to be written that you could get a **sub-best** model and it’s really **time consuming**, this doesn’t contain feature engineering step, but with AUTOML, what you need to do is just to prepare the dataset to train, then just **with few code** or **even without code**, you could get a good model for evaluation, also it doesn’t need any manually setting for model, it will **automatically** choose the best parameters fit best on your data without much effort.

Here is the steps that I would take for introduction:

1. What’s AUTOML
2. Target of AUTOML
3. Commercial AUTOML framework
4. Open-source AUTOML framework
5. Use case tips
6. Reference

# What’s AUTOML

Automated machine learning (AutoML) is the process of automating end-to-end the process of applying [machine learning](https://en.wikipedia.org/wiki/Machine_learning) to real-world problems.

In a typical machine learning application, practitioners have a dataset consisting of input data points to train on. The **raw data itself** may not be in a form that **all algorithms may be applicable** to it out of the box. An expert may have to apply the appropriate [**data pre-processing**](https://en.wikipedia.org/wiki/Data_pre-processing), [**feature engineering**](https://en.wikipedia.org/wiki/Feature_engineering), [**feature extraction**](https://en.wikipedia.org/wiki/Feature_extraction), and [**feature selection**](https://en.wikipedia.org/wiki/Feature_selection)methods that make the dataset trainable for machine learning.

Following those preprocessing steps, practitioners must then perform [**algorithm selection**](https://en.wikipedia.org/wiki/Algorithm_selection)and [**hyperparameter optimization**](https://en.wikipedia.org/wiki/Hyperparameter_optimization) to **maximize** the **predictive performance** of their final machine learning model. As many of these steps are often beyond the abilities of non-experts, AutoML was proposed as **an**[**artificial intelligence**](https://en.wikipedia.org/wiki/Artificial_intelligence)**-based solution** to the ever-growing challenge of applying machine learning. Automating the process of applying machine learning end-to-end offers the advantages of producing **simpler** **solutions**, **faster creation** of those solutions, and models that often **outperform** models that were designed by hand. However, AutoML is not a silver bullet and can introduce additional parameters of its own, called **hyper-hyperparameters**, which may need **some expertise** to be set themselves. But it does make application of Machine Learning easier for **non-experts**.

# Target of AUTOML

There are some goals of AUTOML, generally speaking there are divided into 3 parts: 1. Data feature engineering; 2. Hyper-parameters tuning; 3. Architecture searching for deep learning.

But for these three parts, the important and basic part is feature engineering, as the sentence says: The feature engineering is the upper bound of whole algorithms. So no matter what powerful, how deep algorithms or structures that we use, without a good feature engineering step, we wouldn’t get great solution for problems.

For auto feature engineering part, there contains bellow parts:

* Automated [data preparation](https://en.wikipedia.org/wiki/Data_preparation) and ingestion
* Automated column type detection; e.g., boolean, discrete numerical, continuous numerical, or text
* Automated column intent detection; e.g., target/label, [stratification](https://en.wikipedia.org/wiki/Stratified_sampling) field, numerical feature, categorical text feature, or free text feature
* Automated task detection; e.g., [binary classification](https://en.wikipedia.org/wiki/Binary_classification), [regression](https://en.wikipedia.org/wiki/Regression_analysis), clustering, or [ranking](https://en.wikipedia.org/wiki/Learning_to_rank)
* Automated [feature engineering](https://en.wikipedia.org/wiki/Feature_engineering): [Feature selection](https://en.wikipedia.org/wiki/Feature_selection), [Feature extraction](https://en.wikipedia.org/wiki/Feature_extraction), [Meta learning](https://en.wikipedia.org/wiki/Meta_learning_(computer_science)) and [transfer learning](https://en.wikipedia.org/wiki/Transfer_learning), Detection and handling of skewed data and/or missing values

For hyper-parameters tuning parts:

* Automated [model selection](https://en.wikipedia.org/wiki/Model_selection)
* [Hyperparameter optimization](https://en.wikipedia.org/wiki/Hyperparameter_(machine_learning)#Optimization) of the learning algorithm and featurization
* Automated pipeline selection under time, memory, and complexity constraints
* Automated selection of evaluation metrics / validation procedures

For architecture searching for deep learning part, the most famous part is Google neural architecture search paper [Learning Transferable Architectures for Scalable Image Recognition](https://arxiv.org/abs/1707.07012):

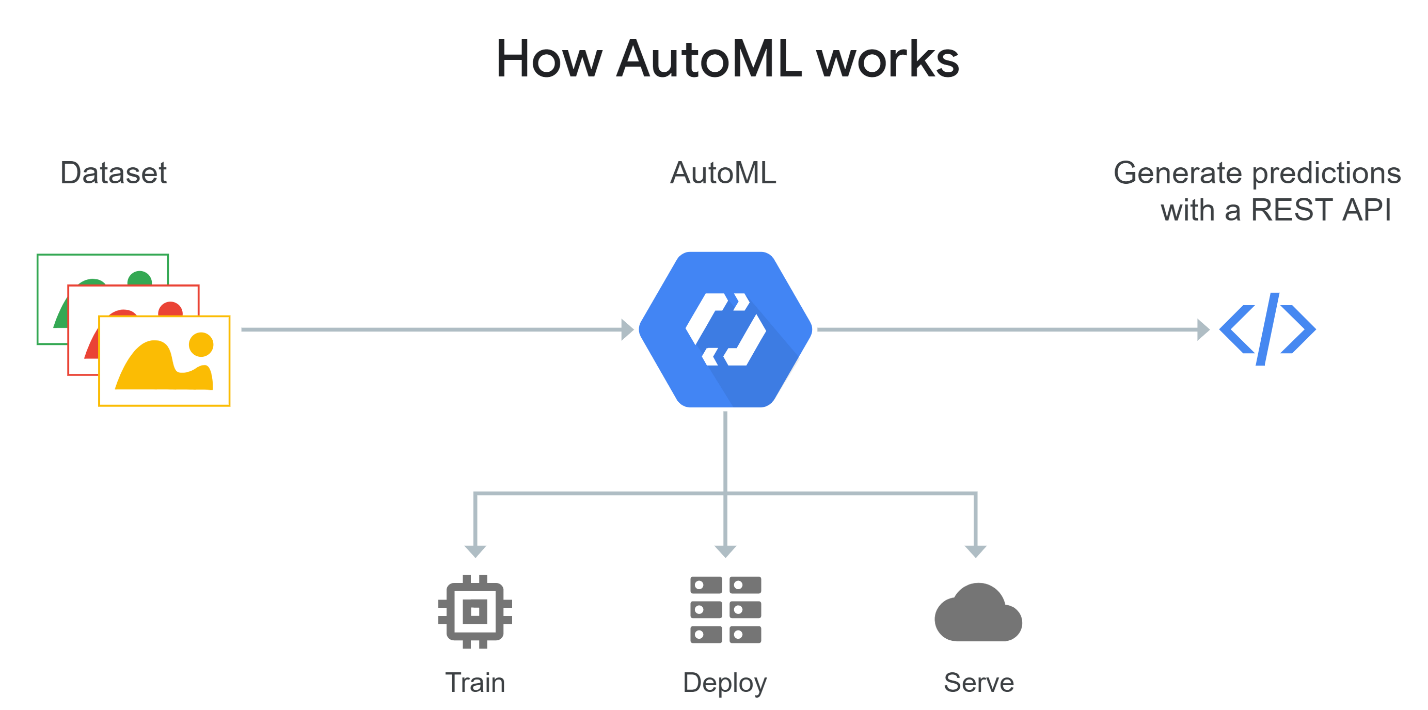
* Best model structure that could fit best on the data we have
* Best number of layers that we should use

# Commercial AUTOML framework

## Google AUTOML [Google AUTOML](https://cloud.google.com/automl/)

Google cloud AUTOML focuses on 3 parts: AUTO Computer Vision, AUTO Natural Language and AUTO structure data.

How this works:



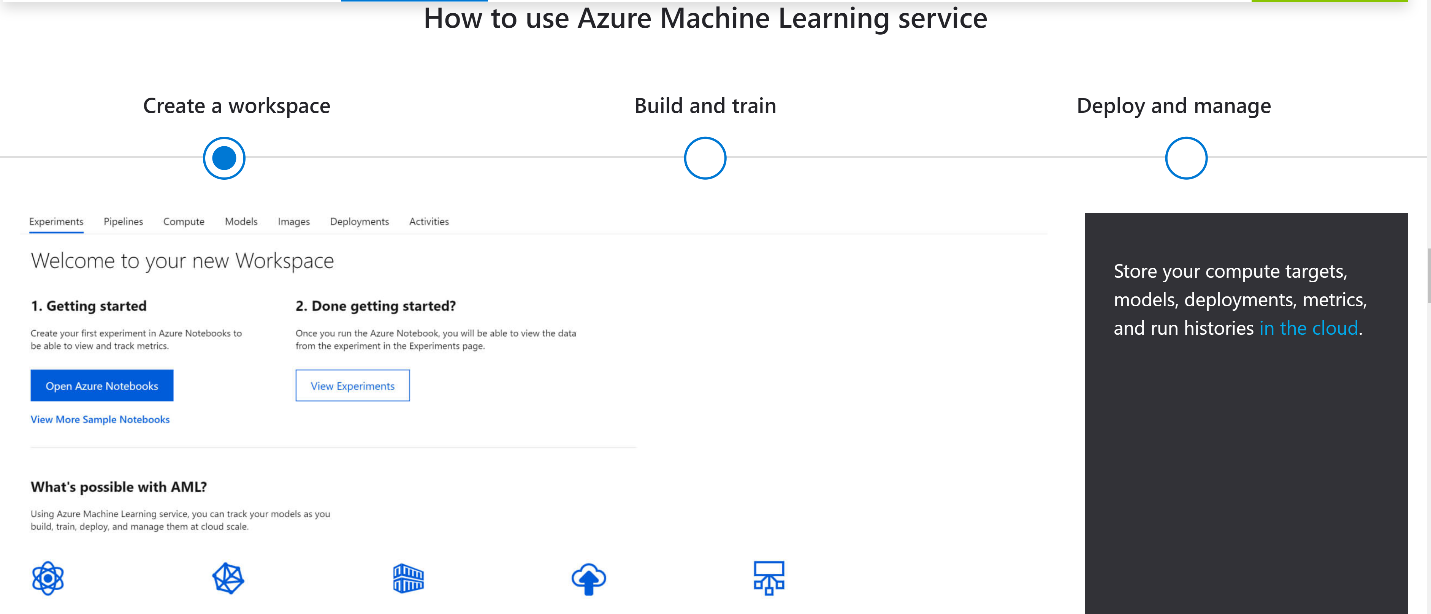
Features of Google AUTOML:

1. Custom models: Train custom machine learning models that are specific to your business needs, with minimum effort and machine learning expertise.
2. Transfer learning benefits: Leverages Google state-of-the-art AutoML and transfer learning technology to produce high-quality models.
3. Integration with data labeling: If you have images but no labels yet, an in-house Google team will review your custom instructions and classify your images accordingly. You’ll get high-quality training data while your data remains private. You can also use the human-labeled data seamlessly to train a custom model. Available for AutoML Vision only.

## AZURE AUTOML [AZURE AUTOML](https://azure.microsoft.com/en-us/services/machine-learning-service/)

Streamline the building, training, and deployment of machine learning models. Bring machine learning models to market faster using the tools and frameworks of your choice, increase productivity using automated machine learning, and innovate on a secure, enterprise-ready platform.

How to use:



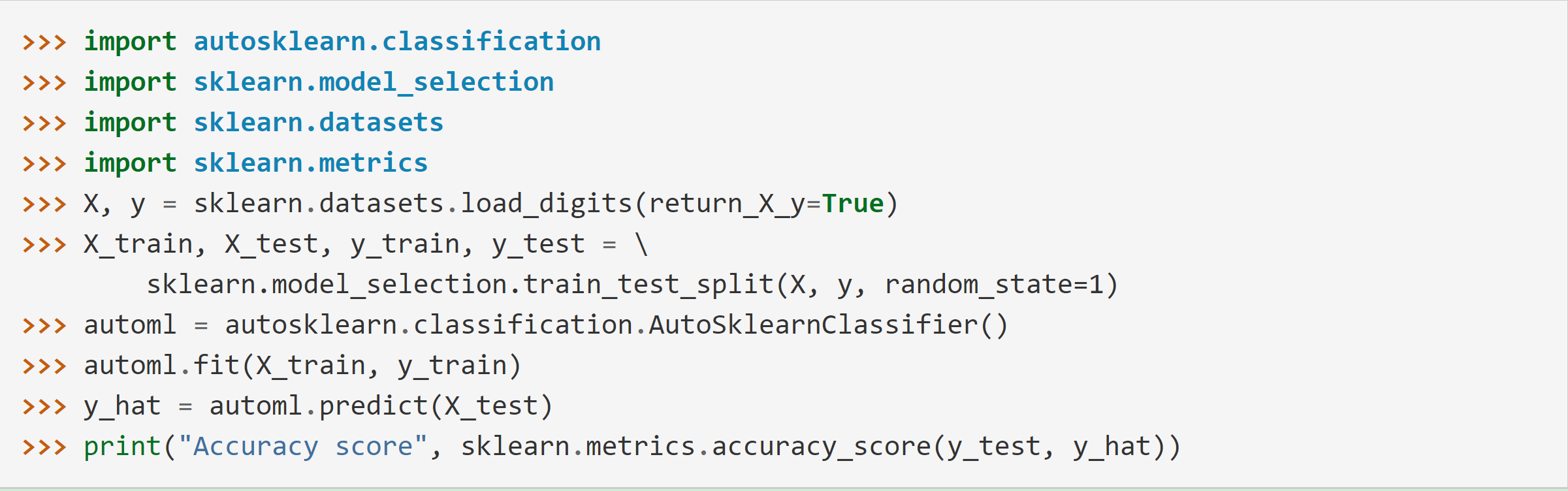
Azure AUTOML provides with security with custom data, put and train model on cloud, and provide DevOps solution with machine learning.

# Open-source AUTOML framework

## 4.1. Auto-sklearn [auto-sklearn](https://automl.github.io/auto-sklearn/master/)

auto-sklearn is an automated machine learning toolkit and a drop-in replacement for a scikit-learn estimator.

auto-sklearn frees a machine learning user from algorithm selection and hyperparameter tuning. It leverages recent advantages in Bayesian optimization, meta-learning and ensemble construction. Learn more about the technology behind auto-sklearn.

With auto-sklearn, what we just need to decide what our target is, for now it just support with: binary-classification, multi-class-classification, multi-label-classification and regression. To use auto-sklearn is really easy, just as bellow: 

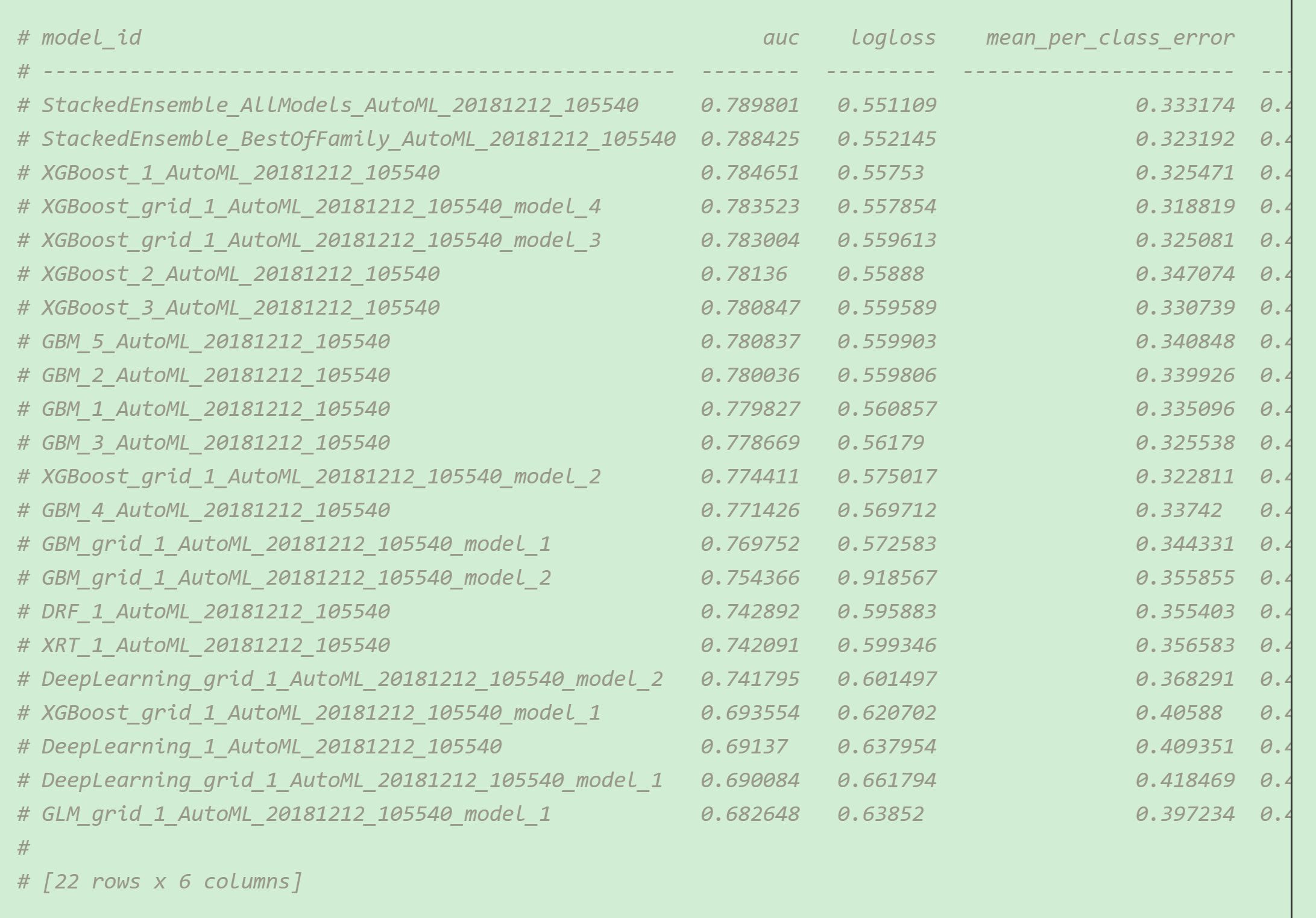
Auto-sklearn has many build-in algorithms and come up with the meta-algorithms, they tested with many algorithms on many datasets, and come up with some powerful algorithms that should be used, for detail info, with this paper: [auto-sklearn paper](http://papers.nips.cc/paper/5872-efficient-and-robust-automated-machine-learning.pdf).

## 4.2 AUTOML H2O [ho2 automl](http://docs.h2o.ai/h2o/latest-stable/h2o-docs/automl.html)

## H2O’s AutoML can be used for automating the machine learning workflow, which includes automatic training and tuning of many models within a user-specified time-limit. [Stacked Ensembles](http://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/stacked-ensembles.html) – one based on all previously trained models, another one on the best model of each family – will be automatically trained on collections of individual models to produce highly predictive ensemble models which, in most cases, will be the top performing models in the AutoML Leaderboard.

There are really many parameters for training step and evaluation step, but you don’t need to know how this works inherent, what you need to do is just to prepare the training and testing data for model, then you could just train your model on your personal computer.

The character of H2O AUTOML is it provides the leaderboard of the whole training step that are best according to the evaluation function, just like bellow:



# Use case tips

In fact, I have just tried with h2o AUTOML open source project in my local computer, the first time I use it, I think it is really easy to use, but in fact, you should really take some patience to get what you want.

The first thing is to install with h2o module, here is install module step:

pip install requests

pip install tabulate

pip install "colorama>=0.3.8"

pip install future

pip install **-**f http:**//**h2o**-**release**.**s3**.**amazonaws**.**com**/**h2o**/**latest\_stable\_Py**.**html h2o

This will take a long time to install!

There are several things to notice: **First** is when you prepare dataset for h2o, it just support with dataset with .cvs format for now, I use .xlsx doesn’t support! So you should first convert your data to .csv file. **Second** thing to notice is that you have to convert your label with factor type, otherwise you will get a **wrong model**, I have tried with that! **Third** thing to be noticed is that you have to set max\_models with a considerable number, as I set with 40 models, it takes me about 12 hours to fit a model!

After the whole step finish, the only thing is just to wait it to finish, but it will really take a long time to train your model on your **local computer**! But the result model really provide a good result for test data.

In fact, I think h2o AUTOML is really a good start to try with AUTOML, as it provides with examples that we could follow, if the data is not so big, this wouldn’t take much time to get the final model, also it provides with leaderboard that we could get some insight of how the models are trained with our data by using cross-validation. There are also some other parameters that we could choose to fit our model, you could try it by your own.

# Reference

Google AUTOML: <https://cloud.google.com/automl/>

AZURE AUTOML: <https://azure.microsoft.com/en-us/services/machine-learning-service/>

Auto-sklearn: <https://automl.github.io/auto-sklearn/master/>

H2O AUTOML: <http://docs.h2o.ai/h2o/latest-stable/h2o-docs/automl.html>