## Automated Machine Learning

Traditionally, the development of predictive solutions is a challenging and time consuming process that requires expert resources in software development, data engineering, and data science. Engineers are required to complete the following tasks in an iterative and cyclical manner.

1. Preprocess, feature engineer, and clean data
2. Select appropriate model
3. Tune Hyperparameters
4. Analyze Results

As the industry identified the blockers that make the development of machine learning solutions costly, we aim to figure out a way to automate the process in an attempt to make it easier and faster to deploy intelligent solutions. Therefore, selecting and tuning models can be automated to make the analysis of results easier for non-expert and expert developers.

Automated machine learning is the ability to have a defined dataset with a specific target feature, and automatically iterate over the dataset with different algorithms and combination of input variables to select the best model. The purpose is to make developing this solutions require less resources, less domain knowledge, and less time.

### How it Works

Most Auto ML libraries available are used to solve supervised learning in order to solve specific problems. If you are unfamiliar, there are two main categories of machine learning.

* **Supervised Learning**: is where you have input variables and output variables, and you apply algorithms to learn the mapping function of input to output.
* **Unsupervised Learning**: is where you have input variables but no output variables to map them to. The goal is typically to identify trends and patterns in the data to make assumptions.

Note there is a category called **semi-supervised learning** but we will not get into that here. But it is simply a combination of the two categories above.

In order to use auto machine learning your dataset must be feature engineered. Meaning, you manually develop transformations to create a machine learning dataset to solve your problem. Most Auto ML libraries have built in transformation functions to solve the most popular transformation steps, but in my experience these functions are rarely enough to get data machine learning ready.

Once you have featured engineer your dataset the developer simply needs to determine the type of algorithm they need. Most supervised learning algorithms can be classified as:

* **Classification**: The output variable is a set number of outcomes. For example, predicting if a customer will return to a store is either a "yes" or a "no". Classification is additionally broken into multiclassification (3 or more outcomes) and binary classification (2 outcomes).
* **Regression**: The output is a numeric value. For example, predicting the prices of a car or house.

When given an algorithm type, Auto ML libraries will run iterations over your dataset to determine the best combination features, and best hyperparameters for each algorithm, therefore, in turn it actually trains many models and gives the engineer the best algorithm.

I would like to highlight the differences between having to engineer columns for machine learning, and selecting the appropriate columns for machine learning. For example, lets assume I want to predict how many point of sale transactions will occur every hour of the day. The raw dataset is likely transactional, therefore, will require a developer to summarize the data at the hour level i.e. grouping, summing, and averaging. But often times developers will create custom functions in order to describe the trends in the dataset. This process is **feature engineering**.

**Feature selection** comes after feature engineering. I may summarize my dataset with 10 different columns that I **believe** will be useful, but Auto ML libraries may select the 8 best columns out of the 10.

When Auto Machine Learning libraries are used in the development process the output is usually a dataset containing metadata on the training runs and their results. This dataset enables developers to easily choose the best model based off the metrics provided. Being able to choose the best model out of many training iterations with different algorithms and feature columns automatically is that it enables us to easily automate the model selection process for **each** model deployment. With typical machine learning deployments, engineers typically deploy the same algorithm with the same feature columns each time. But with Auto Machine Learning solutions we are able to note only choose the best algorithm, feature combination, and hyper-parameters each time. That means, we can deploy a decision tree model trained on 4 columns one release, the deploy a logistic regression model trained on 5 columns another release without any code edits. This is so simple, yet so awesome about how easy it can be!

### Available Libraries

[MLBox](https://github.com/AxeldeRomblay/MLBox), a python library for automated machine learning. Key features include distributed processing of data, robust feature selection, accurate hyperparameter tuning, deep learning support, and model interpretation.

[TransmogrifAI](https://transmogrif.ai/), is an Auto ML library for structured data in Scala purposed for Apache Spark. The library is developed and supported by SalesForce developers.

[Auto-sklearn](https://automl.github.io/auto-sklearn/stable/), a python library is great for all the sci-kit learn developers out there. It sits on top of sci-kit learn to automate the hyperparameter and algorithm selection process.

[AzureML](https://docs.microsoft.com/en-us/azure/machine-learning/service/concept-automated-ml), an end to end platform for machine learning development and deployment. The library enables faster iterations by manage and tracking experiments, and fully supports most python-based frameworks like PyTorch, TensorFlow, and sci-kit learn. The Auto ML feature is baked into the platform to make it easy to select your model.

[Ludwig](https://github.com/uber/ludwig), a TensorFlow based platform for deep learning solutions was released by Uber to enable users with little coding experience. The developer simply needs to provide a training dataset and a configuration file identifying the features and labels desired.

Check out the libraries above! Automated machine learning is fun to play around with and apply to problems. I will be creating demos and walk throughs of each of these libraries. Once public you will be able to find them on my [GitHub](https://github.com/ryanchynoweth44/AutoMLExamples).