HITACHI
Inspire the Next

Automated Machine Learning (AutoML) and Pentaho

Caio Moreno de Souza
Pentaho Senior Consultant, Hitachi Vantara

PentahoWorld

Agenda

We will discuss how Automated Machine Learning (AutoML) and Pentaho, together, can help customers save time in the process of creating a model and deploying this model into production.

- Business Case for Automated Machine Learning (AutoML) and Pentaho;
- High level overview about Automated Machine Learning (AutoML);
- Demonstrations (Pentaho + AutoML).

The Perfect Model Does Not Exist

"All models are wrong, but some are useful."

– GEORGE BOX, 1919-2013

Business Case for AutoML and Pentaho

- Finding the correct machine learning algorithm is not an easy task.
- You need to find a balance between the time you would need to spend and the time you can actually spend on the ML problem.
- To create a good model you will need to know very well the problem, the variables (instances), prepare the data, feature engineering and test different algorithms.
- Some data scientists will also say to add a little bit of MAGIC ☺.
- Adding, of course, in most cases, a lot of computer power.

Machine Learning High-Level Overview

What is Automated Machine Learning (AutoML)?

Illustration by Shyam Sundar Srinivasan

What is Automated Machine Learning (AutoML)?

"Machine learning is very successful, but its successes crucially rely on human machine learning experts, who select appropriate ML architectures (deep learning architectures or more traditional ML workflows) and their hyperparameters. As the complexity of these tasks is often beyond non-experts, the rapid growth of machine learning applications has created a demand for off-the-shelf machine learning methods that can be used easily and without expert knowledge. We call the resulting research area that targets progressive automation of machine learning AutoML."

https://sites.google.com/site/automl2016/

Why Automated Machine Learning (AutoML)?

- The demand for machine learning experts has outpaced the supply. To address this gap, there have been big strides in the development of user-friendly machine learning software that can be used by non-experts and experts, alike.
- AutoML software can be used for automating a large part of the machine learning workflow, which includes automatic training and tuning of many models within a user-specified time-limit.

What is NOT Automated Machine Learning (AutoML)?

- AutoML is not automated data science;
- AutoML will not replace Data Scientist;
  - All the methods of automated machine learning are developed to support data scientists, not to replace them.
  - AutoML is to free data scientists from the burden of repetitive and time-consuming tasks (e.g., machine learning pipeline design and hyperparameter optimization) so they can better spend their time on tasks that are much more difficult to automate.

Auto ML Tools

- Auto Weka (Open Source)
  - http://www.cs.ubc.ca/labs/beta/Projects/autoweka/
- H2o.ai AutoML (Open Source)
  - https://www.h2o.ai/
- TPOT (Open Source)
  - https://github.com/rhiever/tpot
- Auto Sklearn (Open Source)
  - https://github.com/automl/auto-sklearn
  - http://automl.github.io/auto-sklearn/stable/
- machineJS (Open Source)
  - https://github.com/ClimbsRocks/machineJS

PDI + AutoML

PentahoWorld

Machine Learning with Pentaho in 4 Steps

http://www.pentaho.com/blog/4-steps-machine-learning-pentaho

CRISP-DM

http://www.pentaho.com/blog/4-steps-machine-learning-pentaho

Use Case: AutoML + Pentaho

- Our users have a well defined ML problem and the initial version of the dataset (train and test).
- Unfortunately, they haven’t created a ML model yet.
- Also, they have no idea how to create it.
- And they want us to help them to create it as soon as possible using only Open Source tools.

The Journey

- If you embark in this journey, you can stick in this problem forever…

  …or you can find quick ways to do it in a specified time.

- Customers can then spend enough time later to improve their current Model.

- The next steps will be:
  - Hire a data scientist or a team of data scientists;
  - Hire a domain expert in that problem.

Our Goal

- In this specific scenario, our goal will be to help them to start the process of creating a dummy model using AutoML.

Create Your First ML Model

1. Define the problem;
2. Analyze and prepare the data;
3. Select algorithms (start simple);
4. Run and evaluate the algorithms;
5. Improve the results with focused experiments;
6. Finalize results with fine tuning.

Sample Dataset

- More data is better, but more data means more complexity.
- More data means more time that you will have to spend in your problem.
- Why not create a sample dataset?!
  - Create 1 to 20 datasets to test your problem and create your models;

Demo AutoML + Pentaho

- This presentation aims to demo the process of how AutoML open source tools and Pentaho, together, can help customers save time in the process of creating a model and deploying this model into production.

The Power of PDI

- PDI (Pentaho Data Integration) will help data scientist and data engineers with data onboarding, data preparation, data blending, model orchestration (model and predict), saving and visualizing the data.

Drastically reduce the time to put models into production

Data Onboarding, Data Preparation and Data Blending

- Below we can see a Data Preparation Process using PDI (Pentaho Data Integration);
- ML dataset output: ARFF File (Weka File), CSV (Python, R and Apache Spark MLlib) and Hadoop Output to save the txt file to the Data Lake;

Predicting New Values Using Your Model

Execution Results

Execution History | Logging | Step Metrics | Performance Graph | Metrics | Preview data

First rows | Last rows | Off

Inspect Dat

p_to_state	ship_to_zip	first_time_customer	order_dollar_amount	num_items	age	web_order	total_transactions_to_date	hour_of_day	billing_shipping_zip_equal	fraud_likelihood	Count
I	97790	N	964.7	19	104.7	Y	40	7	N	0.161	1
(	39356	N	524.1	15	24.2	N	55	4	Y	0.029774582	1

Demonstration

Demo Agenda

What we will cover in the demo:

- Data Preparation with PDI;
- Model creation using AutoML Tool;
- Model Deployment with PDI;

Pentaho Data Integration + H2O AutoML

Summary

What we covered today:

- Business Case for Automated Machine Learning (AutoML) and Pentaho;
- High level overview about Automated Machine Learning (AutoML);
- Demonstrations (Pentaho + AutoML).

Next Steps

Want to learn more?

- Talk to me during Pentaho World 2017 or send me an e-mail caio.moreno@HitachiVantara.com;
- Meet-the-Experts:
  - https://www.pentahoworld.com/meet-the-experts

PentahoWorld

Appendices

Top Prediction Algorithms

- According to Dataiku, the top prediction algorithms are the ones explained in the image on the right side.
- This image also explains (resumes) the advantages and disadvantages of each algorithm.

Source:
https://blog.dataiku.com/machine-learning-explained-algorithms-are-your-friend

Algorithms

*REXER analytics data science survey** gives us a good idea about which algorithms have been used over the years.

Key Findings

Regression, decision trees, and cluster analysis remain the most commonly used algorithms in the field. This has been consistent over the years.

* Special thanks to Mark Hall (Pentaho) for sharing this document with me.
Document available at: http://www.rexeranalytics.com/data-science-survey.html

Core Algorithms

Source: http://www.rexeranalytics.com/files/Rexer_Data_Science_Survey_Highlights_Apr-2016.pdf

Tools

- The huge amount of tools increases the complexity.

Source: http://www.rexeranalytics.com/files/Rexer_Data_Science_Survey_Highlights_Apr-2016.pdf

Auto Weka

- Auto Weka
  - provides automatic selection of models and hyperparameters for WEKA.
  - http://www.cs.ubc.ca/labs/beta/Projects/autoweka/
- Open datasets for Auto Weka
  - http://www.cs.ubc.ca/labs/beta/Projects/autoweka/datasets/

Auto Sklearn

- Auto Weka inspired the authors of Auto Sklearn;
- Auto Sklearn
  - auto-sklearn is an automated machine learning toolkit and a drop-in replacement for a scikit-learn estimator.
  - https://github.com/automl/auto-sklearn
  - http://automl.github.io/auto-sklearn/stable/

Types of ML Problems with (AutoML)

- The types of Machine Learning problems that we can solve using Auto Weka and Auto Sklearn are Classification, Regression and Clustering:
  - Classification and Regression are already supported in Auto-sklearn & Auto-WEKA.
  - For clustering, you can use as long as you have an objective function to optimize.

Automated by TPOT

- TPOT will automate the most tedious part of machine learning by intelligently exploring thousands of possible pipelines to find the best one for your data.

https://github.com/rhiever/tpot

Auto ML Tools Installation

PentahoWorld

Installing Auto Weka

- To install AutoWeka, go to Weka Package Manager > Search for Auto-WEKA and click the “Install” button.

Installing TPOT

- Command to install TPOT
  - $ pip install tpot
- Learn more:
  - http://rhiever.github.io/tpot/installing/

Installing Auto Sklearn on Ubuntu

- Use the documentation below to help you:
  - http://automl.github.io/auto-sklearn/stable/
- Run this command on ubuntu terminal:
  - $ conda install gcc swig
  - $ curl https://raw.githubusercontent.com/automl/auto-sklearn/master/requirements.txt | xargs -n 1 -L 1 pip install
  - $ sudo apt-get install build-essential swig
  - $ pip install –U auto-sklearn

Error Auto Sklearn on Ubuntu

- Error reported on June, 14th 2017. Solution sent on the same day.
- Check the GitHub link below to find the solution: https://github.com/automl/auto-sklearn/issues/308

```
    import _regression
ImportError: /home/caio/anaconda3/lib/python3.6/site-packages/_regression.cpytho
n-36m-x86_64-linux-gnu.so: undefined symbol: _ZTVNSt7__cxx1115basic_stringbufIcS
t11char_traitsIcESaIcEEE
>>>
```

Installing H20.ai

- To install H20.ai AutoML visit the websites:
  - https://blog.h2o.ai/2017/06/automatic-machine-learning/
  - https://www.h2o.ai/

Auto ML Demonstration

PentahoWorld

Using Auto Weka

- timeLimit = You can define the time in minutes that you want Auto Weka to use to run and find the best option.
  - More time = better results

Using Auto Weka

- You can run Auto Weka from the Weka Explorer User Interface

Using Auto Weka

- For better performance, try giving Auto-WEKA more time

Using Auto Weka

- Auto Weka output results

Testing Auto Sklearn

- Open Spyder and test the code below:

demo_automl_auto_sklearn.py untitled0.py

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Tue Jun 13 05:25:14 2017

@author: caio
"""

import autosklearn.classification
import sklearn.model_selection
import sklearn.datasets
import sklearn.metrics
X, y = sklearn.datasets.load_digits(return_X_y=True)
X_train, X_test, y_train, y_test = \
    sklearn.model_selection.train_test_split(X, y, random_state=1)
automl = autosklearn.classification.AutoSklearnClassifier()
automl.fit(X_train, y_train)
y_hat = automl.predict(X_test)
print("Accuracy score", sklearn.metrics.accuracy_score(y_test, y_hat))
```

Source code: http://automl.github.io/auto-sklearn/stable/

Testing Auto Sklearn with Iris Dataset

Testing H2o.ai AutoML

```
aml <- h2o.automl(x = x, y = y,
                  training_frame = train,
                  leaderboard_frame = test,
                  max_runtime_secs = 30)

# View the AutoML Leaderboard
lb <- aml@leaderboard
lb
```

To test H2o AutoML is necessary to install the version 3.11.0.3888 or superior.

http://h2o-release.s3.amazonaws.com/h2o/rel-vapnik/1/index.html

https://github.com/caiomsouza/machine-learning-orchestration/blob/master/AutoML/src/r/h2o-automl/H20_AutoML_Example.R

Demo AutoML (Auto Weka) + Pentaho

- Using Auto Weka from the Weka User Interface we created a first “dummy” model in 15 minutes.

Auto Weka output

- Auto Weka will output the best model created in the time specified, this model can then be used to predict new values.

```
Auto-WEKA result:
best classifier: weka.classifiers.meta.Bagging
arguments: [-P, 74, -I, 4, -S, 1, -W, weka.classifiers.rules.PART, --, -N, 4, -M, 44, -R, -B]
attribute search: null
attribute search arguments: []
attribute evaluation: null
attribute evaluation arguments: []
metric: errorRate
estimated errorRate: 0.03292
training time on evaluation dataset: 15.602 seconds

You can use the chosen classifier in your own code as follows:

Classifier classifier = AbstractClassifier.forName("weka.classifiers.meta.Bagging", new String[]{"-P", "74", "-I", "4", "-S", "1", "-W", "weka.classifiers.rules.PART", "--", "-N", "4", "-M", "44", "-R", "-B"});
classifier.buildClassifier(instances);


Correctly Classified Instances       96762               96.762  %
Incorrectly Classified Instances      3238                3.238  %
Kappa statistic                          0.9151
Mean absolute error                      0.0743
Root mean squared error                  0.1642
Relative absolute error                 19.2375 %
Root relative squared error             37.3821 %
Total Number of Instances           100000

=== Confusion Matrix ===

     a     b   <-- classified as
 24023  2106 |     a = Y
  1132 72739 |     b = N

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.919    0.015    0.955      0.919    0.937      0.915    0.991     0.982     Y
                 0.985    0.081    0.972      0.985    0.978      0.915    0.991     0.996     N
Weighted Avg.    0.968    0.064    0.967      0.968    0.967      0.915    0.991     0.992

Temporary run directories:
C:\Users\cmoreno\AppData\Local\Temp\autoweka4865566465246831251\


For better performance, try giving Auto-WEKA more time.
Tried 23 configurations; to get good results reliably you may need to allow for trying thousands of configurations.
```

No Free Lunch Theorem

www.no-free-lunch.org

No Free Lunch Theorems

Broadly speaking, there are *two* no free lunch theorems. One for *supervised machine learning* (Wolpert 1996) and one for *search/optimization* (Wolpert and Macready 1997). For an overview of the (no) free lunch and associated theorems, see David Wolpert's What does dinner cost?

No Free Lunch for Supervised Machine Learning

Hume (1739–1740) pointed out that 'even after the observation of the frequent or constant conjunction of objects, we have no reason to draw any inference concerning any object beyond those of which we have had experience'. More recently, and with increasing rigour, Mitchell (1980), Schaffer (1994) and Wolpert (1996) showed that bias-free learning is futile.

Wolpert (1996) shows that in a noise-free scenario where the loss function is the misclassification rate, if one is interested in off-training-set error, then there are no a priori distinctions between learning algorithms.

More formally, where
d = training set;
m = number of elements in training set;
f = 'target' input-output relationships;
h = hypothesis (the algorithm's guess for f made in response to d); and
C = off-training-set 'loss' associated with f and h ('generalization error')
all algorithms are equivalent, on average, by any of the following measures of risk: $E(C|d)$, $E(C|m)$, $E(C|f,d)$, or $E(C|f,m)$.

How well you do is determined by how 'aligned' your learning algorithm $P(h|d)$ is with the actual posterior, $P(f|d)$.

Wolpert's result, in essence, formalizes Hume, extends him and calls the whole of science into question.

No Free Lunch for Search/Optimization

The no free lunch theorem for search and optimization (Wolpert and Macready 1997) applies to finite spaces and algorithms that do not resample points. All algorithms that search for an extremum of a cost function perform exactly the same when averaged over all possible cost functions. So, for any search/optimization algorithm, any elevated performance over one class of problems is exactly paid for in performance over another class.

http://www.no-free-lunch.org/

IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, VOL. 1, NO. 1, APRIL 1997 67

No Free Lunch Theorems for Optimization

David H. Wolpert and William G. Macready

***Abstract*—A framework is developed to explore the connection between effective optimization algorithms and the problems they are solving. A number of "no free lunch" (NFL) theorems are presented which establish that for any algorithm, any elevated performance over one class of problems is offset by performance over another class. These theorems result in a geometric interpretation of what it means for an algorithm to be well suited to an optimization problem. Applications of the NFL theorems to information-theoretic aspects of optimization and benchmark measures of performance are also presented. Other issues addressed include time-varying optimization problems and *a priori* "head-to-head" minimax distinctions between optimization algorithms, distinctions that result despite the NFL theorems' enforcing of a type of uniformity over all algorithms.**

***Index Terms*— Evolutionary algorithms, information theory, optimization.**

information theory and Bayesian analysis contribute to an understanding of these issues? How *a priori* generalizable are the performance results of a certain algorithm on a certain class of problems to its performance on other classes of problems? How should we even measure such generalization? How should we assess the performance of algorithms on problems so that we may programmatically compare those algorithms?

Broadly speaking, we take two approaches to these questions. First, we investigate what *a priori* restrictions there are on the performance of one or more algorithms as one runs over the set of all optimization problems. Our second approach is to instead focus on a particular problem and consider the effects of running over all algorithms. In the current paper we present results from both types of analyses but concentrate

https://ti.arc.nasa.gov/m/profile/dhw/papers/78.pdf

Section 8: Probability, Induction, and Decision Theory

NOTICE: NO FREE LUNCHES FOR ANYONE, BAYESIANS INCLUDED

Malcolm R. Forster
Philosophy Department, University of Wisconsin, Madison, WI 53706, U. S. A.
e-mail: mforster@facstaff.wisc.edu

The problem with the no-free-lunch theorems in machine learning is that they are rather complicated and difficult to follow because they are proven under the most general conditions possible (Wolpert 1992, 1995, 1996). The purpose of this note is to present the idea behind them in a very simple example.

Consider an imaginary universe that lasts for exactly 2 days, where on each day there exists exactly one object, which is either a sphere or a cube. The object may or

http://philosophy.wisc.edu/forster/papers/Krakow.pdf