How to make your model awesome with Optuna

towardsdatascience.com/how-to-make-your-model-awesome-with-optuna-b56d490368af

30 June 2021

Odysseus and the Sirens source

Easily and efficiently optimize model's hyperparameters

Hyperparameter optimization is one of the crucial steps in training Machine Learning models. With many parameters to optimize, long training time and multiple folds to limit information leak, it may be a cumbersome endeavor. There are a few methods of dealing with the issue: grid search, random search, and Bayesian methods. **Optuna** is an implementation of the latter one.

Will Koehrsen wrote an excellent article about hyperparameter optimization with Bayesian surrogate models. I can't explain it any better :) You can find it here. In the second article, Will presents how to implement the method with **Hyperopt** package in Python.

About Optuna

The package was, and still is, developed by a Japanese AI company **Preferred Networks**. In many ways, Optuna is similar to Hyperopt. So why should you bother? There are a few reasons:

- It's possible to specify how long the optimization process should last
- Integration with Pandas DataFrame

- The algorithm uses pruning to discard low-quality trials early
- It's a relatively new project, and developers continue to work on it
- It was easier to use than Hyperopt (at least for me)

Example walk-through

Jason and the Argonauts source

Data

I used the 20 newsgroups dataset from Scikit-Learn to prepare the experiment. You can find the data import below:

Model

It's a Natural Language Processing problem, and the model's pipeline contains a feature extraction step and a classifier. The code for the pipeline looks as follows:

Optimization study

The School of Athens by Raphael source

The created study optimizes both the vectorization step and the model hyperparameters. It's possible to choose out of 5 distributions:

- uniform — float values
- log-uniform — float values
- discrete uniform — float values with intervals
- integer — integer values
- categorical — categorical values from a list

The syntax looks like this:

The values are then passed to the parameters dictionary and later on set to the optimized model. The values could be suggested inside the dictionary, but it would make the code lines very long and hard to read.

The last step of defining the function is actually to define the **objective**. It should return one value only. It's highly recommended to score the model based on **cross-validation** (stratified if possible) with a high number of folds (8 is an absolute minimum).

Please keep in mind that as for Feb 24th 2019 it's only possible to **minimize** the function's value. Maximization objective is not implemented yet, so if you want to find the highest value just put a minus before it.

I've also added a line dumping the study into a pickle file. It allows you to keep the progress even if the process is somehow interrupted (by your decision or your machine's). You can find the objectives code below:

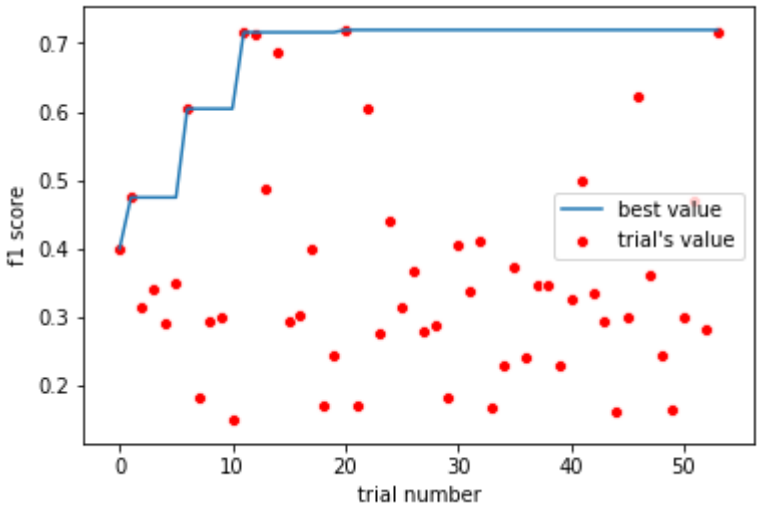
To create your study's instance, you can either create a new one or load it from the pickle file to continue previous experiments. The second step is running the study. You can specify how long the study lasts in the **number of trials** (n_trials) or **in time in seconds** (timeout). The latter's last trial starts before the timeout, and the whole study lasts a little bit longer than specified. You can find the code and the default output below.

Note that the best hyperparameters so far are displayed.

You can access the best metric's value and best parameters dictionary with **best_value** and **best_params** attributes respectively. You can access the trials with trials attribute, but Optuna creators prepared something better. **Use trials_dataframe() method to create a Pandas DataFrame with trials' details.**

After the study ends, you can set the best parameters to the model and train it on the full dataset.

To visualize the ongoing process, you can access the pickle file from another Python's thread (i.e., Jupyter Notebook).



Ongoing study's progress. *Image by author*

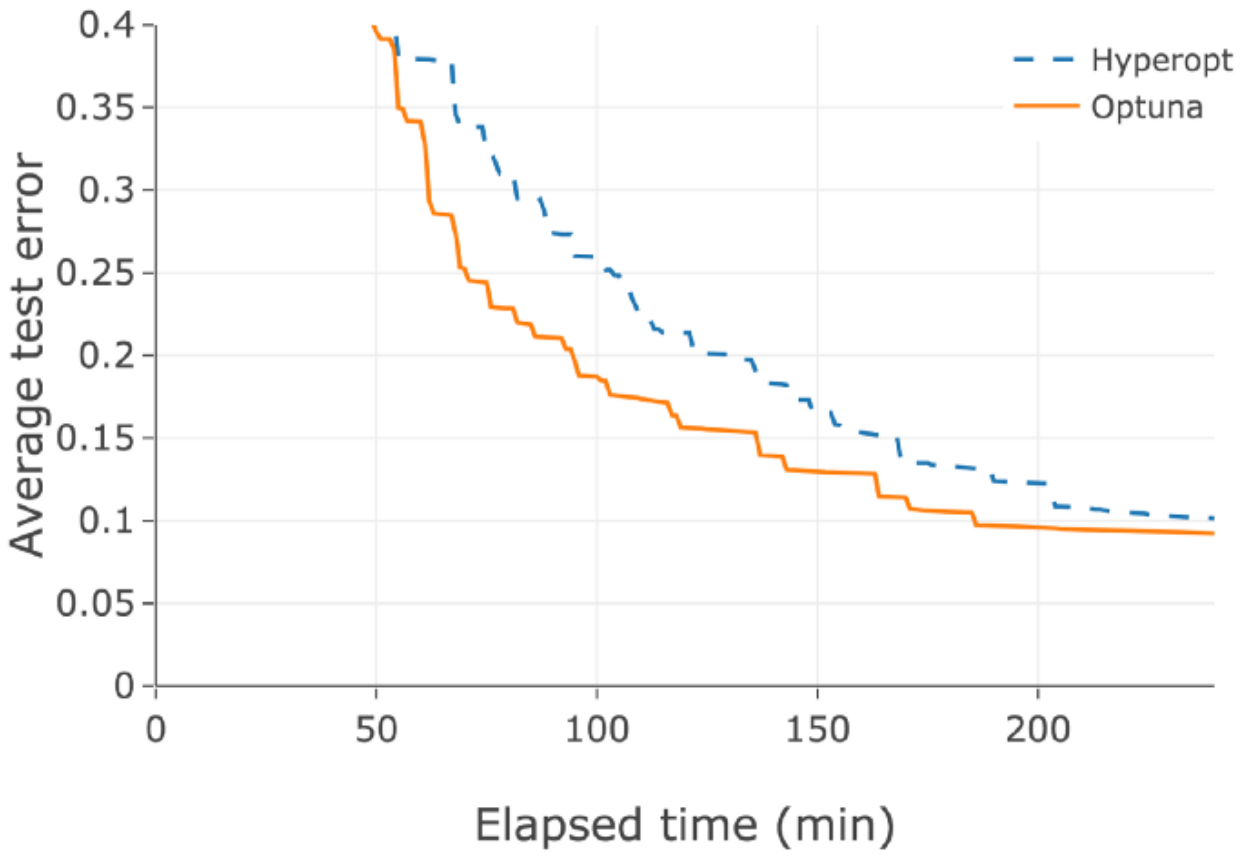
You can find the Example notebook and the Visualization Notebook in this GitHub repository.

The pruning technique

A man pruning olives source

Optuna's creators state that the package outperforms Hyperopt in terms of speed and quality. Below is what they write about it on the project's webpage:

> Pruning feature automatically stops unpromising trials at the early stages of the training (a.k.a., automated early-stopping). Optuna provides interfaces to concisely implement the pruning mechanism in iterative training algorithms.
>
> […]
>
> For instance, our benchmark experiment demonstrates the advantage of the pruning feature in comparison with an existing optimization framework.



https://optuna.org/assets/img/Pruning_of_unpromising_trials.png

I have to say that I have some mixed feelings in terms of the pruning mechanism. **I firmly believe that the cross-validation technique with many folds is essential in terms of hyperparameters' optimization.** In the pruning examples provided by Optuna's developers at each trial, the validation set is sampled. In my opinion, it **increases the metric's variance** and therefore makes **optimization less reliable**. If the validation set was constant, it would cause surrogate's model **overfitting** to the set. Of course, it may not be such a problem if the dataset is big to prevent the variance/overfitting problems to occur.

In my opinion, the best option would be to marry cross-validation and pruning somehow. Maybe validating the trial after k folds (with k smaller than the total number of folds) would be a good idea?

Summary

For me, **Optuna becomes the first choice optimization framework**. It's easy to use, makes it possible to set the study's timeout, continue the study after a break and access the data easily. The project is still in development, and I'm looking forward to new features. I think that the Preferred Networks team did and still does a great job! Thank you!

Resources:

Odyssey source

- *Project's website:* https://optuna.org/
- *Project's GitHub repository:* https://github.com/pfnet/optuna
- *Example Notebooks:* https://github.com/PiotrekGa/optuna_article
- *A Conceptual Explanation of Bayesian Hyperparameter Optimization for Machine Learning:* https://towardsdatascience.com/a-conceptual-explanation-of-bayesian-model-based-hyperparameter-optimization-for-machine-learning-b8172278050f
- *Automated Machine Learning Hyperparameter Tuning in Python:* https://towardsdatascience.com/automated-machine-learning-hyperparameter-tuning-in-python-dfda59b72f8a

Your feedback about the article is more than welcomed! Please let me know what do you think!

Thanks to Adrian Kral, alex, and Liudmyla Kyrashchuk.

Sign up for The Variable

By Towards Data Science

Every Thursday, the Variable delivers the very best of Towards Data Science: from hands-on tutorials and cutting-edge research to original features you don't want to miss. Take a look.

Emails will be sent to omri.guttman@gmail.com.

More from Towards Data Science

Your home for data science. A Medium publication sharing concepts, ideas and codes.

Nandesora Tjihero

·Feb 28, 2019

Conducting a Population Proportion Estimation Study(Binary Variable): Ideas on how to approach it.

Image by JanBaby on Pixabay

Introduction

Suppose I live in a country named X in sub-saharan Africa. The research is based on a real country but some reasons, I prefer to call it X.The third cause of death in X is believed to be automobile accidents. I consider myself a cautious and responsible driver. However, reckless driving seems to be pervasive. The type of accident I fear the most,and that one

would hear frequently about, is head-on collision. To see to what an extend I can justify my fear, I decided to conduct a study that estimates the probability of having that kind of accident...

Read more · 6 min read

Share your ideas with millions of readers.

Write on Medium

Manuel A. Ramirez Garcia

·Feb 28, 2019 ★

Data Analysis Shows Regional Healthcare Costs in New York State Can Vary by More Than $100,000

Depending if you live in Western, Upstate or Downstate NY, medical procedures can range from tens to hundreds of thousands of dollars.

Photo by Piron Guillaume on Unsplash

Medical costs are a mystery to many of us. No two patients are the same so their bills won't be either. Knowing the range of prices for a procedure can assist consumers in financial planning. It equips them with a better sense of how much their coinsurance, deductible, copays and other healthcare related expenses could be.

Wouldn't it be better if we could get healthcare costs from hospitals only within our local community or geographical regions? Unfortunately, this kind of local-level information is difficult to come by. This is partly because the documents (database tables) that contain cost information may...

Read more · 14 min read

Jeff Hale

·Feb 28, 2019 ★

10 Git Commands You Should Know

Plus tips to save time with Git

In this article, we're going to discuss the miscellaneous Git commands you should know as a developer, data scientist, or product manager. We'll look at inspecting, removing, and tidying with Git. We'll also cover ways to escape Vim and save time with Bash aliases and Git editor configuration.

If you aren't comfortable with basic Git commands, check out my previous article on Git workflows before reading this one.

Learn Enough Git to be Useful

4 Essential Workflows for GitHub Projects

towardsdatascience.com

Here are 10 commands to know and some of their common flags. Each command links to the Atlassian Bitbucket guide for that command.

Inspecting Things

Let's look at inspecting changes first.

Read more · 6 min read

Gustavo Chávez

·Feb 28, 2019

Implementing a Naive Bayes classifier for text categorization in five steps

The Naive Bayes classifier guide I wish I had before

Thomas Bayes (1701–1761) author of the Bayes' theorem

> Naive Bayes is a learning algorithm commonly applied to text classification.

Some of the applications of the Naive Bayes classifier are:

- **(Automatic) Classification of emails in folders**, so incoming email messages go into folders such as**:** **"**Family", "Friends", "Updates", "Promotions", etc.

- **(Automatic) Tagging of job listings.** Given a job listing in raw text format, we can assign it tags such as: “software development”, “design”, “marketing”, etc.
- **(Automatic) Categorization of products.** Given a product description, we can assign it into categories such as: “Books”, “Electronics”, “Clothing”, etc.

The remainder of this article will provide the necessary background and intuition to build…

Read more · 8 min read

Yoav Ramon

·Feb 28, 2019

Speaker Diarization with Kaldi

With the rise of voice biometrics and speech recognition systems, the ability to process audio of multiple speakers is crucial. This article is a basic tutorial for that process with Kaldi X-Vectors, a state-of-the-art technique.

In most real-world scenarios speech does not come in well defined audio segments with only one speaker. In most of the conversations that our algorithms will need to work with, people will interrupt each other and cutting the audio between sentences won’t be a trivial task.

In addition to that, in many applications we will want to identify multiple speakers in a conversation, for example when writing a protocol of a meeting. For such occasions, identifying the different speakers and connect different sentences under the same speaker is a critical task.

Speaker Diarization is the solution for those problems. With…

Read more · 8 min read

Read more from Towards Data Science