**dumperize** 9 дек 2022 в 04:56

# Optuna. Подбор гиперпараметров для вашей модели

6 мин 19K

Python*, Big Data*, Машинное обучение*, Искусственный интеллект

Туториал Из песочницы

Гиперпараметры — это характеристики модели, которые фиксируются до начала обучения (например - глубина решающего дерева, значение силы регуляризации в линейной модели, learning rate для градиентного спуска). Гиперпараметры, в отличие от параметров задаются разработчиком модели перед ее обучением, в свою очередь параметры модели настраиваются в процессе обучения модели на данных.

**Optuna** — это фреймворк для для автоматизированного поиска оптимальных гиперпараметров для моделей машинного обучения. Она подбирает эти параметры методом проб и ошибок.

**Ключевые особенности фреймворка:**

1. Настраиваемое пространство поиска гиперпараметров. Разработчик может самостоятельно задать пространство для поиска гиперпараметров, используя базовый синтаксис Python (циклы, условия).
2. Алгоритмы SoTA для выбора гиперпараметров из пространства заданного разработчиком (samplers) и для ранней остановки бесперспективных экспериментов (pruners). В Optuna представлены различные алгоритмы семплирования и прунинга, разработчик может выбрать какой-то конкретный, оставить дефолтный, или написать свой собственный.
3. Легкость расспараллеливания процесса поиска гиперпараметров. Также к Optuna можно прикрутить dashboard с визуализацией обучения в реальном времени.

## Установка

Рекомендуется установка через pip.

```
pip install optuna
```

## Базовый пример

Этот фреймворк обычно используют как оптимизатор гиперпараметров, но никто не запрещает использовать ее для оптимизации любой функции. В качестве базового примера использования, авторы фреймворка показывают как можно минимизировать квадратичную функцию $(x-2)^2$.

```
import optuna

def objective(trial):
    x = trial.suggest_float('x', -10, 10)
    return (x - 2) ** 2

study = optuna.create_study()
study.optimize(objective, n_trials=100)

study.best_params  # E.g. {'x': 2.002108042}
```

1. Определяем целевую функцию `objective`, в через аргументы она будет получать специальный объект `trial`. С его помощью можно назначать различные гипермараметры, Например, как в примере выше, мы задаем `x` в интервале $[-10, 10]$.
2. Далее создаем объект обучения с помощью метода `optuna.create_study`.
3. Запускаем оптимизацию целевой функции `objective` на 100 итераций `n_trials=100`. Происходит 100 вызовов нашей функции с различными параметрам от -10 до 10. Какие именно параметры выбирает optuna будет описано ниже.

## Как задать пространство поиска гиперпараметров?

Как было показано выше в целевую функцию будет передан специальный объект `Trial`. Так как наша целевая функция будет вызываться некоторое число раз, на каждом вызове из объекта `Trial` будут возвращаться новые значения параметров. Разработчику остается только задать характеристики этих параметров. Для этого есть несколько методов:

1. `suggest_categorical(name, choice)` задает категориальные параметры. Пример
2. `suggest_float(name, low, high, *, step=None, log=False)` задает параметр типа `float` - число с плавающей точкой. Пример
3. `suggest_int(name, low, high, step=1, log=False)` задает параметр типа `int` - целое число. Пример

## Что еще можно настроить до начала оптимизации?

Чтобы запустить обучение нам необходимо создать объект `Study`. Его рекомендуется создавать либо с помощью метода `create_study` (пример) или `load_study` (пример).

В момент создания можно указать:

1. направление оптимизации функции `directions` - минимизация или максимизация
2. `storage` адрес базы данных, для сохранения результатов испытаний
3. `study_name` имя, если не указать, то будет сгенерировано автоматически. Указание собственного имени, удобно при сохранении экспериментов и их загрузке
4. `pruner` и `sampler` - об этом ниже

После создания объекта Study, можно приступать к оптимизации целевой функции. Сделать это можно с помощью метода `optimize` (пример).

### Как посмотреть результаты оптимизации?

В объекте Study есть специальные поля, которые позволяют посмотреть результаты после обучения:

1. `study.best_params` лучшие параметры
2. `study.best_value` лучшее оптимальное значение целевой функции
3. `study.best_trial` развернутые параметры лучшего испытания

## Как сохранить/загрузить результаты испытаний?

Сохранить только историю в виде датафрейма

```
df = study.trials_dataframe()
df.to_csv('study.csv')
loaded = pd.read_csv('study.csv')
```

Сохранить дамп самого оптимизатора

```
joblib.dump(study, 'experiments.pkl')
study_loaded = joblib.load('experiments.pkl')
study_loaded.trials_dataframe()
```

Можно также сохранять результаты испытаний в БД, для этого в Optuna есть специальный модуль Storages, который предоставляет некоторые объекты для взаимодействия БД. Например

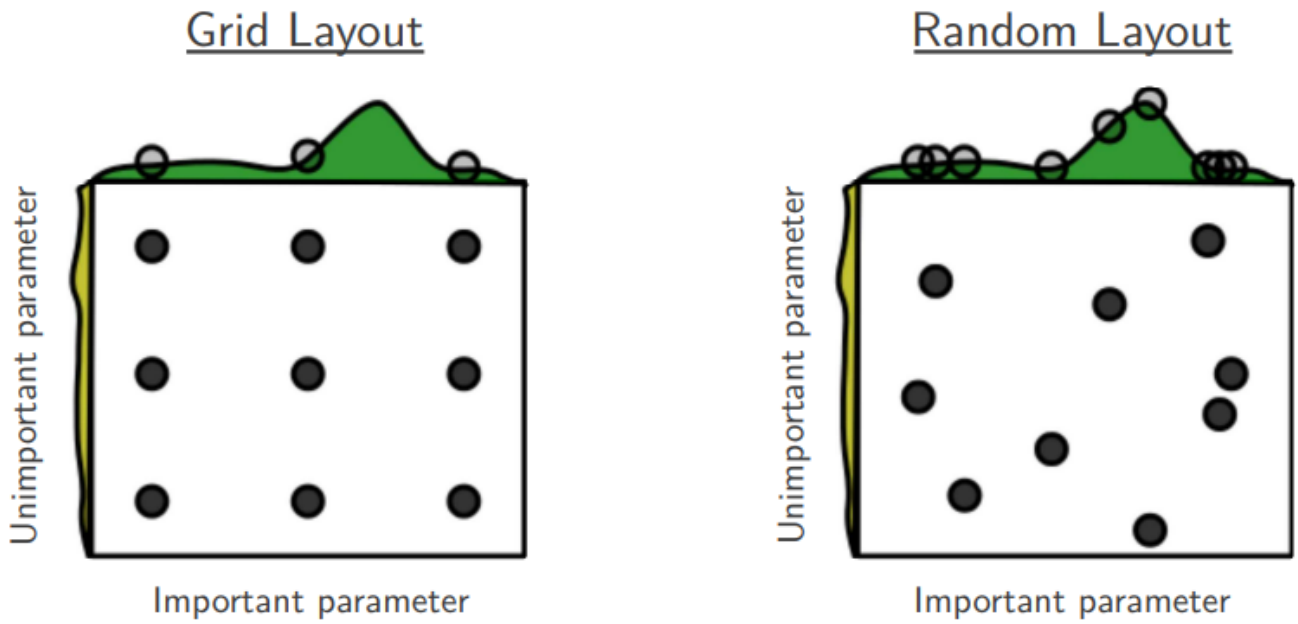
есть объект позволяющий взаимодействовать с redis. Пример.

## Что такое Sampler и Pruner?

**Samplers** в Optuna это набор алгоритмов для поиска гиперпараметров.

Небольшой экскурс в теорию. Существуют различные подходы к поиску оптимальных гиперпараметров, ниже примеры алгоритмов:

1. Grid Search - поиск по решетке. Для каждого гиперпараметра задается список возможных значений, после перебираются все возможные комбинации элементов списков, выбирается тот набор на котором значение целевой функции было минимальным/максимальным.
2. Random Search - случайный поиск. Для каждого гиперпараметра задается распределение, из которого выбирается его значение. Благодаря такому подходу, найти оптимальный набор гиперпараметров можно быстрее.




3. Байесовская оптимизация. Итерационный метод, который на каждой итерации указывает наиболее вероятную точку, в которой наша целевая функция будет оптимальна. При этом выдаваемые вероятные точки включают две компоненты:
    1. хорошая точка там, где согласно истории функция выдавала хорошие результаты на предыдущих вызовах (exploitation)
    2. хорошая точка там, где высокая неопределенность, то есть неисследованные части пространства (exploration)

Более подробно про эти алгоритмы, а также про Tree-structured Parzen Estimator (TPE), Population Based Training (PBT) можно прочитать в учебнике по машинному обучению от Яндекс, там же можно найти ссылки на полезные ресурсы по этой теме и сравнение подходов между собой.

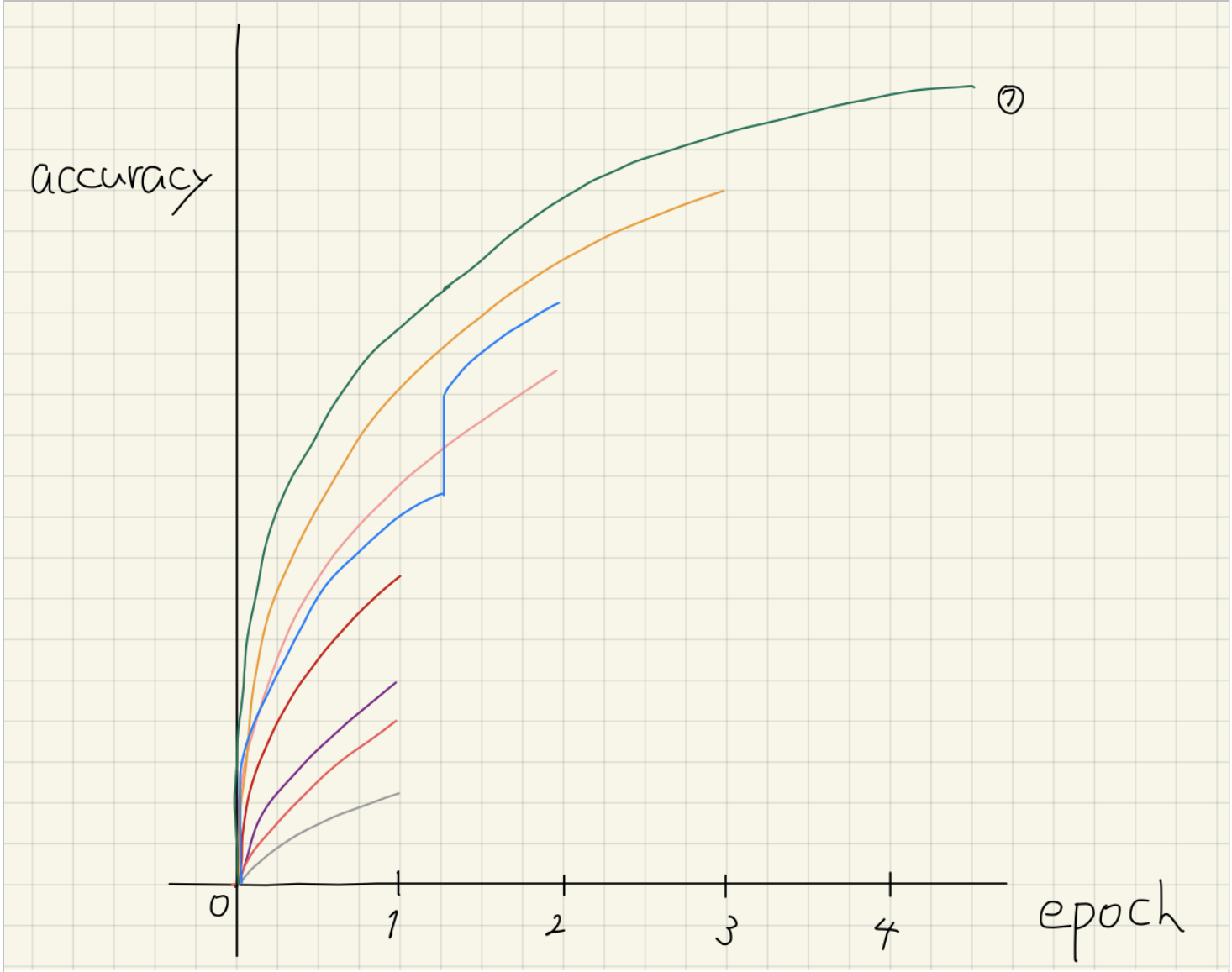
В Optuna реализованы:

- GridSampler - Grid Search
- RandomSampler - Random Search
- TPESampler - Tree-structured Parzen Estimator
- CmaEsSampler - Алгоритм на основе CMA-ES
- PartialFixedSampler - Алгоритм с частично фиксированными параметрами
- NSGAIISampler - Nondominated Sorting Genetic Algorithm II
- MOTPESampler - Multiobjective tree-structured parzen estimator
- QMCSampler - Quasi Monte Carlo

По умолчанию устанавливается TPESampler.

**Pruners** в Optuna - это набор алгоритмов для прореживания экспериментов. Pruning - это механизм который позволяет обрывать эксперименты , которые с большой долей вероятности приведут к не оптимальным результатам.

Для примера рассмотрим самый простой прунер - **MedianPruner.** Он обрезает на каждом шаге половину бесперспективных испытаний.




На каждой эпохе (шаге) Pruner отбрасывает ровно половину испытаний, после 3х эпох, лучшим остается 7 испытание, оно будет доведено до конца, остальные будут завершены раньше.

В Optuna реализованы:

- MedianPruner - pruner использующий правило половина останавливается, половина продолжает
- NopPruner - pruner который никогда не останавливает испытания.
- PatientPruner - pruner обертка над любым другим pruner, позволяет не останавливать бесперспективные испытания, ~~пока не закончится терпение у PatientPruner~~ еще несколько эпох.
- PercentilePruner - pruner, который сохраняет определенный процентиль испытаний.
- SuccessiveHalvingPruner - алгоритм Asynchronous Successive Halving
- HyperbandPruner - алгоритм Hyperband
- ThresholdPruner - pruner, который останавливает испытание, если значение целевой функции вышло за границы - превысило верхний порог или стало ниже чем нижний порог.

## Какой Sampler и Pruner стоит использовать ?

В документации согласно этому исследованию “Benchmarks with Kurobako” для не глубогкого обучения стоит использовать:

- Для RandomSampler лучший pruner - это MedianPruner.
- Для TPESampler лучший pruner - это Hyperband.

В документации также приведены рекомендации для глубокого обучения.

## Как подружить фреймворк с популярными библиотеками?

В Optuna есть модуль `integration`, который содержит классы, используемые для интеграции с внешними популярными библиотеками машинного обучения. Среди них есть такие библиотеки как CatBoost, fast.ai, Keras, LightGBM, PyTorch, scikit-learn, XGBoost. С полным списком можно ознакомится тут.

## А что еще есть?

- Есть модуль для визуализации, в нем представлены функции для построения графика процесса оптимизации с использованием plotly и matplotlib. Функции построения графиков обычно принимают объект Study и настроечные параметры.

  Здесь пример построения графика истории оптимизации.
- Есть модуль importance, с помощью него есть возможность провести оценку важности гиперпараметров на основе завершенных испытаний.

**Теги:** optuna, подбор гиперпараметров, machinelearning, data science

**Хабы:** Python, Big Data, Машинное обучение, Искусственный интеллект

10 Карма 0 Рейтинг

**Елена Николаевская** @dumperize

Пользователь

Подписаться

**Комментарии** 3

**romanzes** 9 дек 2022 в 12:09

Спасибо!
Они там еще otpuna-dashboard сейчас пилят

0 Ответить

**bambalbi** 27 дек 2022 в 23:36

Отличная статья) Спасибо большое!

0 Ответить

**Mind08** 12 мар 2023 в 17:21

Спасибо!

Экспериментировал немного. Заметил, что распределение важности гиперпараметров нестабильно, сильно меняется при повторных запусках (продолжение обучения), небольшом изменении поиска.

0 Ответить

Вы можете оставлять комментарии только к свежим публикациям

## Публикации

ЛУЧШИЕ ЗА СУТКИ ПОХОЖИЕ

**olegsklyarov** 21 час назад

### Как я уронил прод на полтора часа (и при чем тут soft delete и partial index)

7 мин 13K

+85 64 91 +91

**ThingCrimson** 19 часов назад

### TOTP без смартфона

Простой 5 мин 5.5K

Кейс

+35 64 55 +55

melnik909 21 час назад

## HTML и CSS ошибки, влияющие на доступность. Мой опыт и моего незрячего знакомого Ильи. Часть 7

Простой 7 мин 1.4K

Обзор

+28 26 6 +6

Pavel_Kanaev 20 часов назад

## «В черном-черном кабинете»: как в Европе начали перехватывать и расшифровывать письма на государственном уровне

Простой 15 мин 5.5K

+27 34 8 +8

MaFrance351 22 часа назад

## iOmega JAZ. Жёсткие диски со сменными блинами

Средний 7 мин 3K

Ретроспектива

+27 16 18 +18

fellow_pablo 20 часов назад

## YouTube Shorts из терминала. Как автоматизировать создание видео с помощью FFMPEG и Bash

Простой 8 мин 1.3K

Туториал

+25 26 12 +12

AlexandraPurgina 21 час назад

## Нужен ли продакт в ML-команде? Мнение изнутри

Простой 9 мин 1.1K

Мнение

+24 10 2 +2

oldadmin 17 часов назад

## HDD, SSD или NVMe: что выбрать для виртуального сервера (тесты внутри)

Средний 6 мин 4.9K

Обзор

+23 28 34 +34

BiktorSergeev 15 часов назад

## Подключаемся к BBS через Amstrad NC100 из 1992 года

4 мин 1.5K

+22 8 8 +8

Mortyre 18 часов назад

## Учите матчасть: почему стоит изучать туториалы перед работой с облаками и кому это особенно важно

5 мин 1.2K

+19 22 2 +2

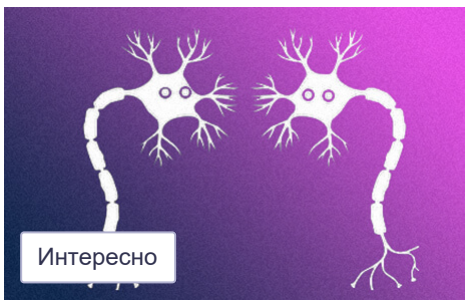
## Всему учён и изловчён: где взять знания, которые в работе точно пригодятся

Турбо

Показать еще

### МИНУТОЧКУ ВНИМАНИЯ


Как бессонница в час ночной, меняет промокодище облик твой




Нейросети в авторитете — сейчас о них пишут в сто раз чаще


Глупым вопросам и ошибкам — быть! IT-менторство на ХК

## КУРСЫ

Профессия Аналитик данных с Финансовым университетом
2 апреля 2024 · Нетология

Python-разработчик: расширенный курс
3 апреля 2024 · Нетология

Python-разработчик с нуля
3 апреля 2024 · Нетология

Fullstack-разработчик на Python
3 апреля 2024 · Нетология

Data Scientist
4 апреля 2024 · Нетология

Больше курсов на Хабр Карьере

## ЧИТАЮТ СЕЙЧАС

**Полиция США потребовала от Google идентифицировать пользователей, смотревших определённые видео на YouTube**

105K 149 +149

**Как я уронил прод на полтора часа (и при чем тут soft delete и partial index)**

13K 91 +91

**Время — это не просто ещё одно измерение**

66K 176 +176

**История одной очереди**

6.7K 20 +20

**Автоматизация или как я избегала общения с коллегами. Часть 1**

3.5K 14 +14

**Всему учён и изловчён: где взять знания, которые в работе точно пригодятся**

Турбо

## ИСТОРИИ


**GitVerse: открой вселенную кода**

author-

**Годнота из блогов компаний**

author-

**Нейросети: интересное**

author-

**Как продвинуть машину времени?**

author-

**Учим английский**

## РАБОТА

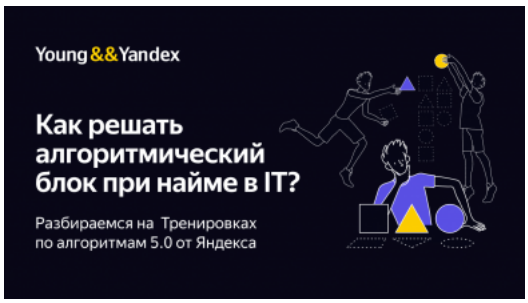
Data Scientist
64 вакансии

Django разработчик
41 вакансия

Python разработчик
127 вакансий

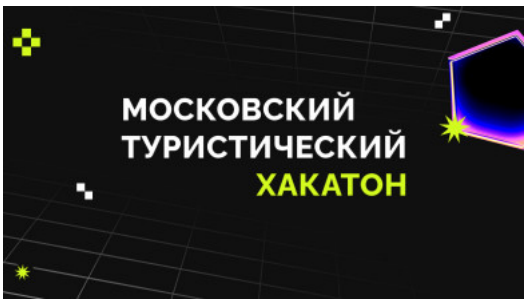
Все вакансии

## БЛИЖАЙШИЕ СОБЫТИЯ



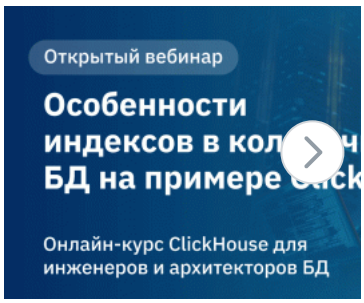

### Серия занятий «Тренировки по алгоритмам 5.0» от Яндекса

1 марта – 19 апреля
19:00
Онлайн




### Московский туристический хакатон

23 марта – 7 апреля
Москва • Онлайн
Подробнее в календаре




### Вебинар «Особен индексов в колон БД на примере ClickHouse»

26 марта 20:0
Онлайн

Подробнее в календаре

Подробнее в календаре

**Ваш аккаунт**

Профиль
Трекер
Диалоги
Настройки
ППА

**Разделы**

Статьи
Новости
Хабы
Компании
Авторы
Песочница

**Информация**

Устройство сайта
Для авторов
Для компаний
Документы
Соглашение
Конфиденциальность

**Услуги**

Корпоративный блог
Медийная реклама
Нативные проекты
Образовательные программы
Стартапам

Настройка языка

Техническая поддержка