LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Bayesian Optimization

Current Research in Data Science

Agenda

- Black Box Optimization
- Surrogate Modelling
- Acquisition Function
- Simulation - BO in AutoML

Agenda

Black Box Optimization

Surrogate Modelling

Acquisition Function

Simulation - BO in AutoML

Situation

Given

Model class $\mathcal{H}$ with parameters $\boldsymbol{\Theta}$ and objective function **f**

Goal

Find parameter values $\boldsymbol{\Theta}$ minimizing objective function **f**

Problem

Analytic relationship between **f** and parameters $\boldsymbol{\Theta}$ unknown

How to model **f**($\boldsymbol{\Theta}$) ?

How to choose $\boldsymbol{\Theta}$?

Solving the Black Box

	Approaches		
	Naive	**Traditional**	**Bayesian**
Parameter Choice	■ Manual Tuning ■ Grid search ■ Random search	■ Nelder-Mead ■ Evolutionary Algorithms	■ Optimal choice by surrogate modelling
Advantages	■ Easy ■ Good results for known problems	■ More focus on relevant regions ■ Derivative-free	■ Possibly global optimum
Disadvantages	■ Inefficient and possibly bad solution ■ Manual eval. doesn’t scale well	■ Inefficient	■ No finite time bounds for optimum

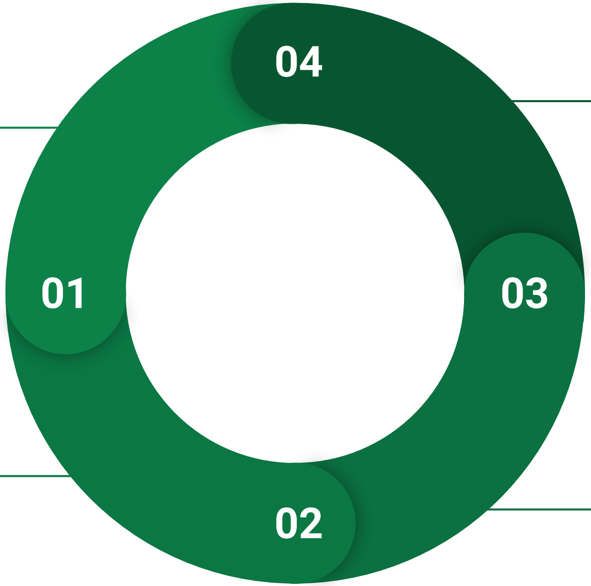
Bayesian Optimization

Approach

Choose next parameter **based on previous evaluation** results s.t. new parameters will most likely lead to optimal result

- **Build a probabilistic surrogate model** mapping hyperparameters to a score probability on the objective function
- **But: Produces computational overhead** (use only if evaluation of objective function is expensive)
  → In AutoML: Function evaluation equals model training

Formalism - Sequential Model-Based Optimization



Fit Surrogate Model

E.g. Update GP posterior with new model evaluation results

Derive Acquisition Function S

Derive closed form solution of EI based on new GP posterior

Choose new Parameters maximizing S

Maximize EI to choose new parameter

Fit Model with new Parameter

Train and evaluate your model based on the new parameters

Agenda

Black Box Optimization

Surrogate Modelling

Acquisition Function

Simulation - BO in AutoML

Gaussian Process Regression

***Gaussian Process** is generalization of a Gaussian distribution over a space of functions. It is similarly defined by a **mean function** and a **covariance function**.*

GP Definition

$$f \sim GP(\mu, K) \quad K_{j,k} = K(x_j, x_k)$$

$$\mathbf{k}(x) = (K(x, x_1) \ldots K(x, x_i))^T$$

Parameter Choice

$$p(y|x, D) \leftarrow FITMODEL(M, D)$$

$$x_i \leftarrow argmax_{x \in X} S(x, p(y|x, D))$$

Surrogate Model

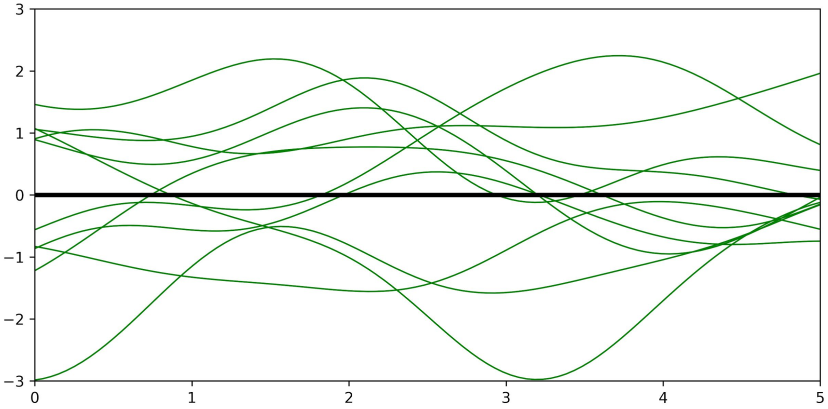
$$p(y|x, D) = N(y|\hat{\mu}, \hat{\sigma}^2)$$

$$y = (y_1 \ldots y_i)^T \qquad \hat{\mu} = \mathbf{k}(x)^T (K + \sigma_n^2 I)^{-1} y$$

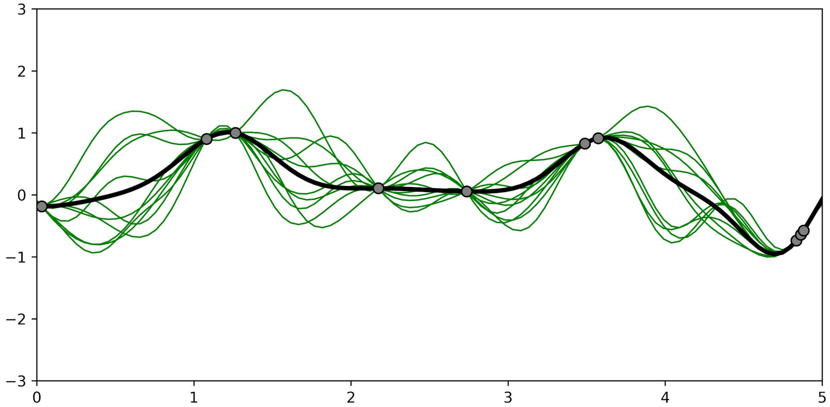
$$\hat{\sigma}^2 = K(x, x) - \mathbf{k}(x)^T (K + \sigma_n^2 I)^{-1} \mathbf{k}(x)$$

Interpolating the Objective Function

Prior



Posterior after 10 observations



Random Forest Regression

***Random Forests** involve training a **Decision Tree** for each training sample, each of which is combined to determine the final output of a **Regression** or **Classification** task.*

Surrogate Model

$$p(y|x, D) = N(y|\hat{\mu}, \hat{\sigma}^2)$$

$$\hat{\mu} = \frac{1}{|B|} \sum_{r \in B} r(x)$$

$$\hat{\sigma}^2 = \frac{1}{|B|-1} \sum_{r \in B} (r(x) - \hat{\mu})^2$$

Parameter Choice

$$p(y|x, D) \leftarrow FITMODEL(M, D)$$

$$x_i \leftarrow argmax_{x \in X} S(x, p(y|x, D))$$

Tree Structured Parzen Estimator

The tree-structured Parzen estimator *(TPE) models* ***p(θ|y)*** *by transforming the generative process, replacing a distribution of the configuration prior with a non-parametric density.*[4]

Basic idea

Modeling

$$p(\theta|y) = \begin{cases} l(\theta), & \text{if } y < y^* \\ g(\theta), & else \end{cases}$$

- $l(\theta)$ is the density formed using the observations θ_i s.t. corresponding loss $f(\theta_i)$ is less than y^*
- $g(\theta)$ - using remaining observations
- y^* is some quantile γ of the observed y values, so that $p(y < y*) = \gamma$

TPE vs BO

TPE Modeling

$$p(\theta|y) \quad p(y)$$

BO Modeling

$$p(y|\theta)$$

Agenda

Black Box Optimization

Surrogate Modelling

Acquisition Function

Simulation - BO in AutoML

Maximizing Stepwise Improvement

***Acquisition Function role**: evaluation of an expected loss associated with evaluating objective function at a certain point.*

- *selecting the point with the **lowest expected loss***
- ***inexpensive to evaluate**, unlike the original optimization problem*

Most common Acquisition Function

Expected Improvement (EI)

$$EI(\theta) = \mathbb{E}[max_\theta(0, f(\theta) - f(\hat{\theta})]$$

$$\theta_{new} = argmax_\theta EI(\theta)$$

- There exists a closed form solution under certain assumptions

EI evaluation tradeoff

- **Exploitation**
  Good approximate global optima are likely to be near points with high expected quality
- **Exploration**
  Evaluation of high uncertainty areas extends knowledge about possible optima

Solving EI for different Surrogates

Optimize Acquisition Function

Tree Parzen Estimator

Bayesian Optimization and Random Forest

$$EI_{y^*}(\theta) = \int_{-\infty}^{y^*} (y^* - y) p(y|\theta) dy$$

$$EI_{y^*}(\theta) = \int_{-\infty}^{y^*} (y^* - y) \frac{p(\theta|y)p(y)}{p(\theta)} dy$$

$$p(y|\theta) = N(y|\hat{\mu}, \hat{\sigma}^2)\, H = (\theta_i, f(\theta_i))_{i=1}^{n}$$

$$EI_{y^*}(\theta) \propto \left(\gamma + \frac{g(\theta)}{l(\theta)}(1-\gamma)\right)^{-1}$$

$$y^* = min\{f(\theta_i), 1 \leq i \leq n\}$$

$$\boxed{\theta^{new} = argmax_{\theta} \frac{g(\theta)}{l(\theta)}}$$

$$\boxed{\theta^{new} = argmax_{\theta} EI_{y^*}(\theta)}$$

Alternative Acquisition Functions

*Used in case of **exotic problems**, where assumptions of EI are violated, e.g. Noisy evaluations.*[1]

Alternatives		
Knowledge Gradient	**Entropy Search**	**Predictive Entropy Search**
▪ utilizes derivative information: for each sample of x being observed the function value and all partial derivatives are used	▪ seeks the point that causes the largest decrease in entropy	▪ seeks the same point, but uses a reformulation of the entropy reduction objective based on mutual information

Agenda

Black Box Optimization

Surrogate Modelling

Acquisition Function

Simulation - BO in AutoML

Optimization frameworks

Python libraries

- **Hyperopt-sklearn**
- **Hyperopt**
  - Random Search
  - Tree of Parzen Estimators
- **Scikit-optimize**
  - Still under development
  - Methods for (SMBO)
    - Decision trees
    - Gradient boosted trees
    - Bayesian Optimization
- **Scipy.optimize**
  - Gradient-based optimization algorithms

R libraries

- **mlr (machine learning in R)**
  - Supervised methods
  - Sampling
  - Hyperparameter tuning using:
    - iterated F-racing (irace)
    - SMBO
- **mlrMBO**
  - Model-based optimization with mlr.
- **MlBayesOpt**
  - Bayesian Optimization based on Gaussian Processes with
    - SVM, Random forest
    - XGboost

Scikit-optimize

***Scikit-Optimize, or skopt,** is a simple and efficient library **to minimize (very) expensive and noisy black-box functions.** It implements several methods for SMBO.*

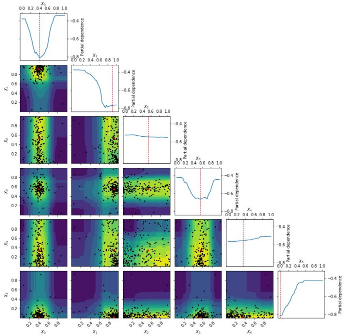
Scikit-optimize functionality

Optimization algorithms

- Random search by uniform sampling
- Decision trees
- Gradient boosted trees
- Bayesian optimization using GP

Other useful API

- Visualization of objective, evaluations, convergence results
- Visualization up to many dimensions



Hyperopt-sklearn

Hyperopt-sklearn** is designed for model selection among machine learning algorithms in **scikit-learn.

Hyperopt-sklearn functionality

Search algorithms

- Random Search
- Tree of Parzen Estimators
- Annealing
- Tree
- Gaussian Process Tree

Preprocessing

- MinMaxScalar
- Normalizer
- OneHotEncoder
- StandardScalar
- TfidfVectorizer
- PCA

Classifiers

- SVC
- Knn
- Random forest
- Decision tree

Hyperopt-sklearn - Comparison

Performance on different datasets in comparison with alternative approaches[2]:

MNIST		20 Newsgroups		Convex Shapes	
Approach	Accuracy	Approach	F-Score	Approach	Accuracy
CNN	99.8%	Class-Feature-Centroid	0.928	**hyperopt-sklearn**	**88.7%**
hyperopt-sklearn	**98.7%**	**hyperopt-sklearn**	**0.856**	hp-deep-belief-net	84.6%
libSVM grid search	98.6%	SVMTorch	0.848	deep-belief-net-3	81.4%
Boosted trees	98.5%	LibSVM	0.843		

Questions

Thanks for your attention!

Sources

[1] Frazier, P. (2018): A Tutorial on Bayesian Optimization

[2] Komer et al. (2014): Hyperopt-Sklearn: Automatic Hyperparameter Configuration for Scikit-Learn

[3] Bischl (2018): Lecture Material on Predictive Modelling

[4] Bergstra et al. (2011): Algorithms for Hyper-Parameter Optimization

[5] Dewancker et al. (2019): Bayesian Optimization Primer