2020.05.15 Seminar

Bayesian Optimization

이 민정

Contents

- Introduction
- Overview of Bayesian Optimization
- Surrogate Model : Gaussian Process Regression
- Acquisition function : Maximum Expected Improvement
- Applications

Introduction

Difficulties of Hyperparameter Tuning

길라임양,
일주일 후에 하이퍼파라미터 튜닝한
Lasso linear regression 모델 결과 가지고 연구미팅합시다.

Y가 연속형인 회귀분석 문제 (Regression)

김주원 교수님

X

설명변수
예측변수

Y

종속변수
반응변수

길라임 신입생

Introduction

Difficulties of Hyperparameter Tuning

Parameter

Hyperparameter

$$\hat{y}_i = \widehat{\beta_0} x_{i0} + \widehat{\beta_1} x_{i1} + \widehat{\beta_2} x_{i2} + \ldots + \widehat{\beta_p} x_{ip}$$

모델의 파라미터(매개변수)가 결정되기전에
하이퍼파라미터(초매개변수) 결정이 필요

$$\min_{\beta} \left\{ \sum_{i=1}^{n} \left(y_i - \sum_{j=0}^{p} x_{ij} \beta_j \right)^2 + \lambda \sum_{j=0}^{p} |\beta_j| \right\}$$

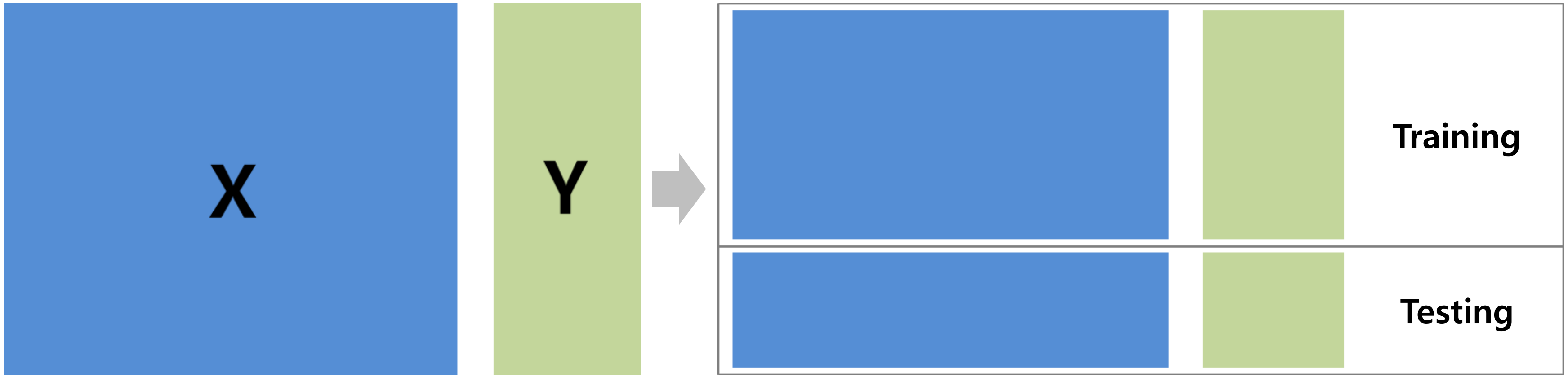
하이퍼파라미터(초매개변수) 결정??

Introduction

Difficulties of Hyperparameter Tuning

10개의 후보

λ : 0.001, 0.01, 0.1, 1.0, 10.0, ...



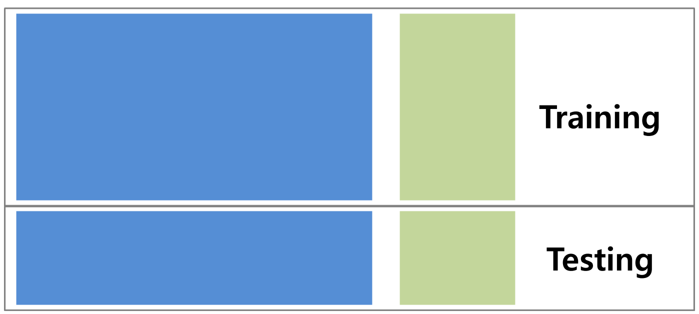
Introduction

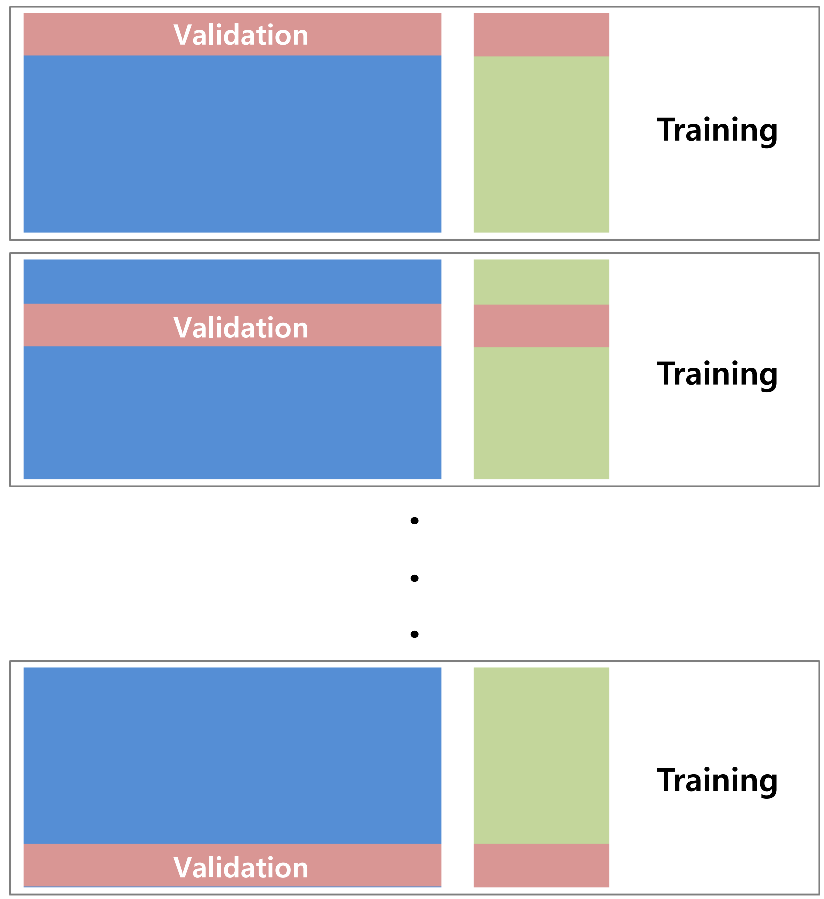
Difficulties of Hyperparameter Tuning

λ : 0.001, 0.01, 0.1, 1.0, 10.0, ...

10 fold cross validation

일반화 성능을 최적화시키는
하이퍼파라미터 찾기





1시간

10개 하이퍼파라미터 후보
× 1 시간 = 10시간

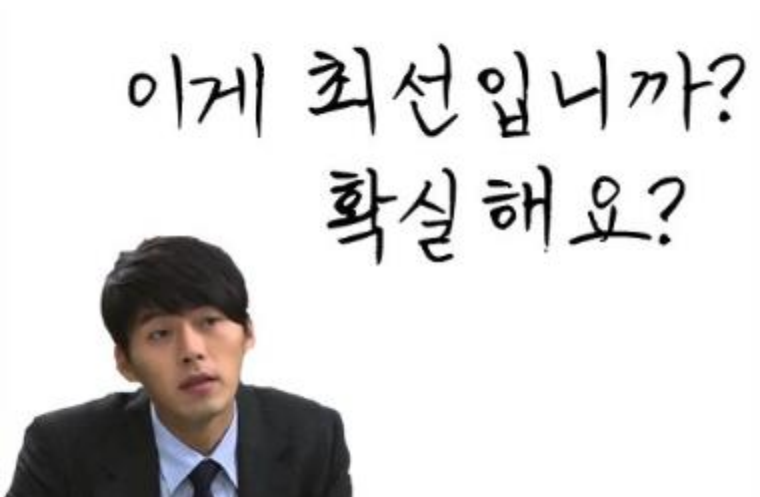
Introduction

Difficulties of Hyperparameter Tuning

$$\min_\beta \left\{ \sum_{i=1}^{n} \left(y_i - \sum_{j=0}^{p} x_{ij}\beta_j \right)^2 + \lambda^* \sum_{j=0}^{p} |\beta_j| \right\}$$

$$\hat{y}_i = \widehat{\beta_0} x_{i0} + \widehat{\beta_1} x_{i1} + \widehat{\beta_2} x_{i2} + \dots + \widehat{\beta_p} x_{ip}$$

길라임 신입생



Introduction

Difficulties of Hyperparameter Tuning

길라임양,
일주일 후에 하이퍼파라미터 튜닝한
Elastic Net linear regression 모델 결과 가지고 연구미팅합시다.

Y가 연속형인 회귀분석 문제 (Regression)

김주원 교수님

X

설명변수
예측변수

Y

종속변수
반응변수

길라임 신입생

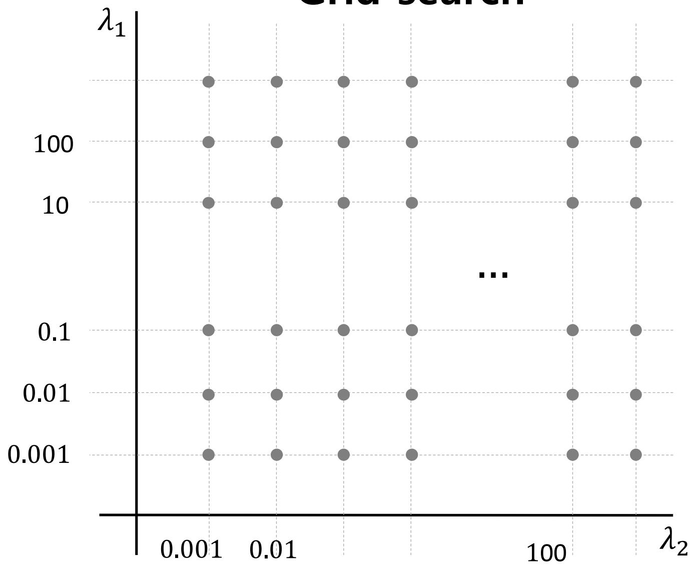
Introduction

Difficulties of Hyperparameter Tuning

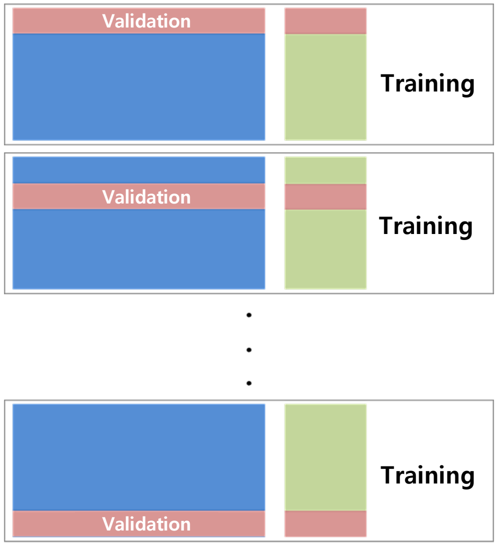
$$\min_{\beta}\left\{\sum_{i=1}^{n}\left(y_i-\sum_{j=0}^{p}x_{ij}\beta_j\right)^2+\lambda_1\sum_{j=0}^{p}|\beta_j|+\lambda_2\sum_{j=0}^{p}\beta_j^2\right\}$$

$$\lambda_1 : 0.001, 0.01, 0.1, 1.0, 10.0, \ldots$$
$$\lambda_2 : 0.001, 0.01, 0.1, 1.0, 10.0, \ldots$$

Grid search



K fold cross validation



10개 하이퍼파라미터 후보
× 10개 하이퍼파라미터 후보
× 1시간 = 100시간 ≈ 4.17일

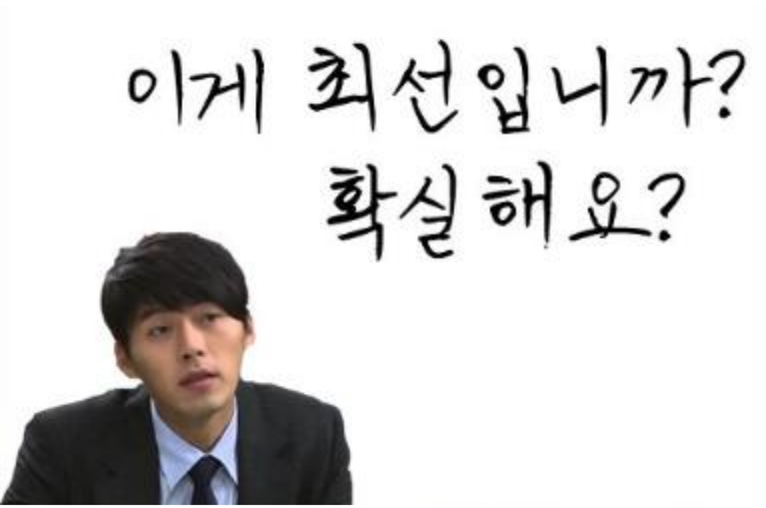
Introduction

Difficulties of Hyperparameter Tuning

$$\min_{\beta}\left\{\sum_{i=1}^{n}\left(y_i-\sum_{j=0}^{p}x_{ij}\beta_j\right)^2+\lambda_1^*\sum_{j=0}^{p}|\beta_j|+\lambda_2^*\sum_{j=0}^{p}\beta_j^2\right\}$$

$$\hat{y}_i=\widehat{\beta_0}x_{i0}+\widehat{\beta_1}x_{i1}+\widehat{\beta_2}x_{i2}+\ldots+\widehat{\beta_p}x_{ip}$$

길라임 신입생



Introduction

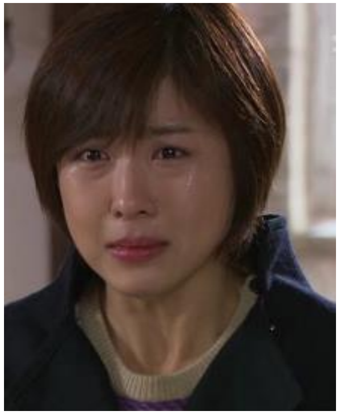
Difficulties of Hyperparameter Tuning

길라임양,
일주일 후에 하이퍼파라미터 튜닝한
Neural Networks 모델 결과 가지고 연구미팅합시다.

Y가 연속형인 회귀분석 문제 (Regression)

김주원 교수님

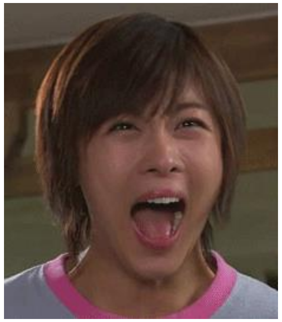
X	Y
설명변수 예측변수	종속변수 반응변수

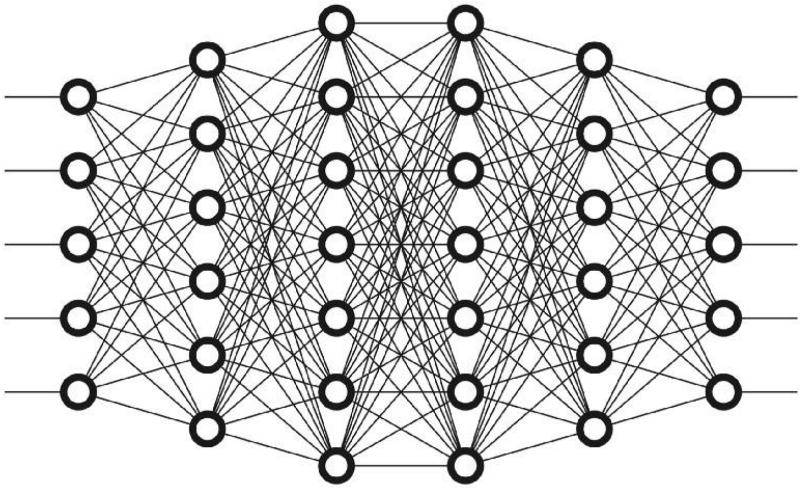


길라임 신입생

Introduction

Difficulties of Hyperparameter Tuning





Hyperparameters

- Learning rate
- # of iterations
- Minibatch size
- # of hidden layers
- # of hidden nodes
- Type of activation functions
- ……

Introduction

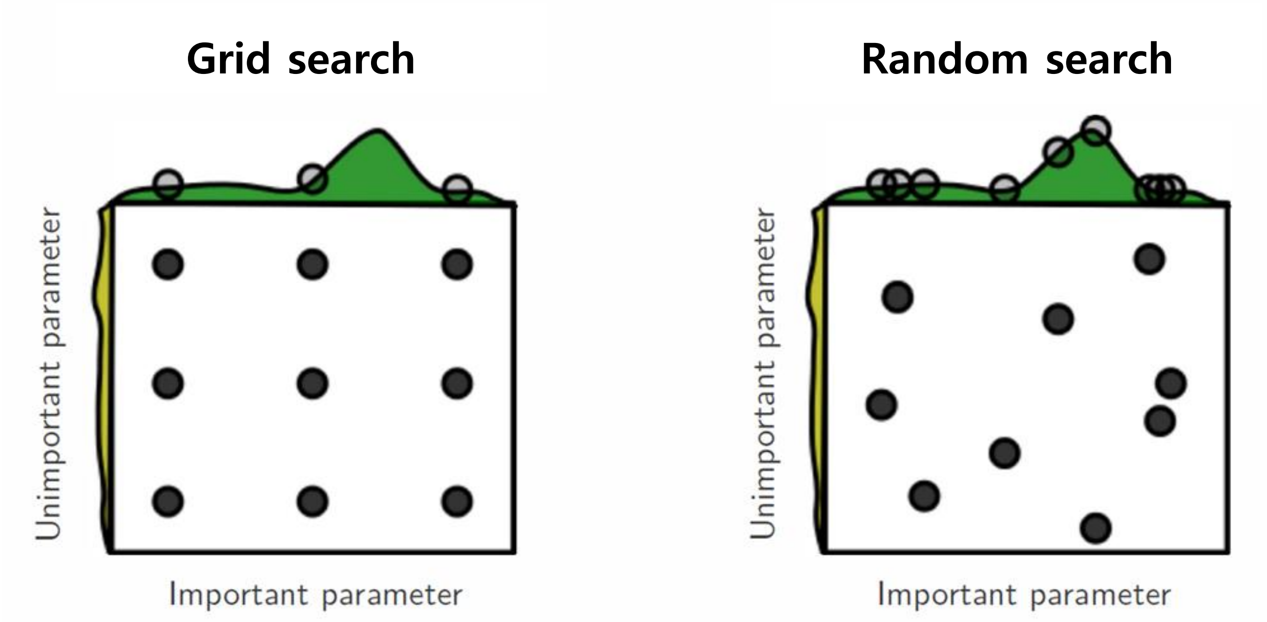
Backgrounds

사전에 정해놓은 하이퍼파라미터 집합 **모든 후보들** 일반화 성능을 확인한 후
가장 좋은 하이퍼파라미터를 찾기

비효율!

Learning rate	# of iterations	# of hidden layers	# of hidden nodes	...	Mini batch size	Generalization performance
0.001	1000	3	16	...	128	100
0.001	100	3	16	...	128	200
0.001	1000	5	16	...	128	300
0.001	100	5	16	...	128	800
...	...	...	...	...	...	...
0.01	1000	3	16	...	128	500
0.01	100	3	16	...	128	150

Introduction
Backgrounds



Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. In Advances in neural information processing systems (pp. 2951-2959).

Introduction

Backgrounds

Learning rate	# of iterations	# of hidden layers	# of hidden nodes	...	Mini batch size
0.001	1000	3	16	...	128
0.001	100	3	16	...	128

관계 파악
(사전지식)

Generalization performance
100
200

의사결정

Bayesian Optimization

Introduction

Backgrounds

Learning rate	# of iterations	# of hidden layers	# of hidden nodes	...	Mini batch size
0.001	1000	3	16	...	128
0.001	100	3	16	...	128
0.001	1000	5	16	...	128

관계 파악 (사전지식)

의사결정

Generalization performance
100
200
300

Bayesian Optimization

Introduction

Backgrounds

Learning rate	# of iterations	# of hidden layers	# of hidden nodes	...	Mini batch size
0.001	1000	3	16	...	128
0.001	100	3	16	...	128
0.001	1000	5	16	...	128
0.001	100	5	16	...	128

효율적!
빠르게!

Generalization performance
100
200
300
800

Bayesian Optimization

Overview of Bayesian Optimization

Bayesian Optimization

Bayesian Optimization

Learning rate	# of iterations	# of hidden layers	# of hidden nodes	...	...	Mini batch size
0.001	1000	3	5	...	...	128
...	...	...	...	...	...	...

Generalization performance
100
200

Overview of Bayesian Optimization

Bayesian Optimization

Bayesian Optimization

Learning rate	# of iterations	# of hidden layers	# of hidden nodes	…	…	Mini batch size
0.001	1000	3	5	…	…	128
…	…	…	…	…	…	…

X

Generalization performance
100
200

Y

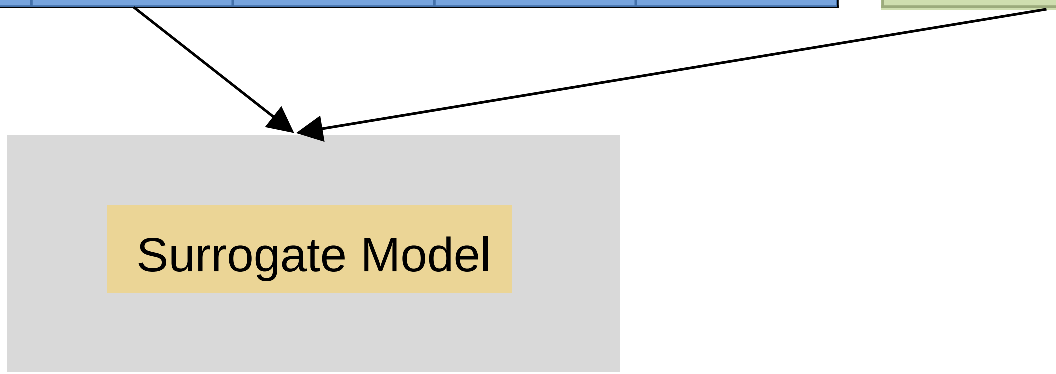
Overview of Bayesian Optimization

Bayesian Optimization

Bayesian Optimization

Learning rate	# of iterations	# of hidden layers	# of hidden nodes	…	…	Mini batch size	Generalization performance
0.001	1000	3	5	…	…	128	100
…	…	…	…	…	…	…	200

X | Y



하이퍼파라미터 집합과 일반화 성능의 관계를 모델링

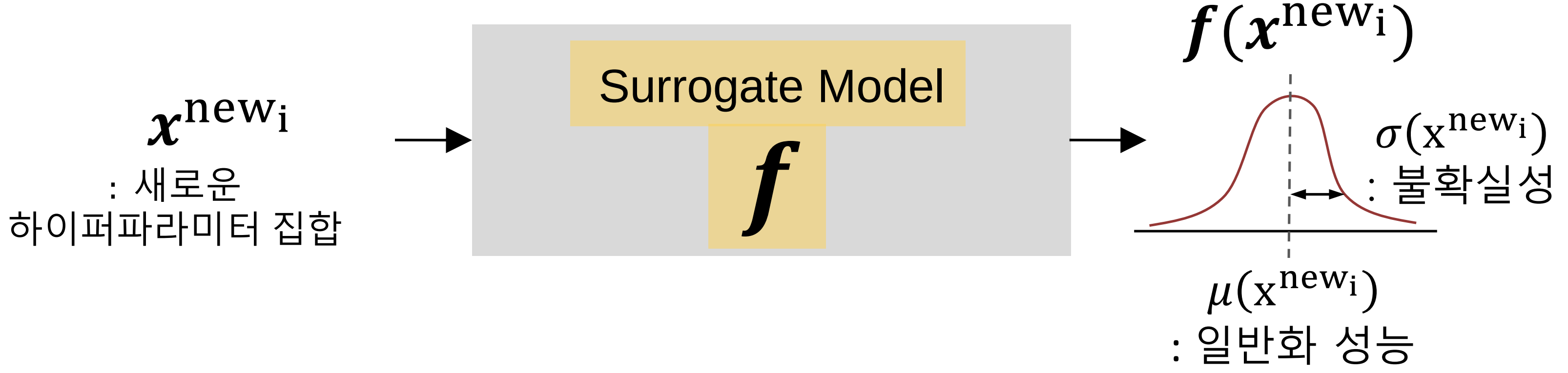
Overview of Bayesian Optimization

Bayesian Optimization

Bayesian Optimization

Learning rate	# of iterations	# of hidden layers	# of hidden nodes	…	…	Mini batch size	Generalization performance
0.001	1000	3	5	…	…	128	100
…	…	…	…	…	…	…	200

X **Y**



Overview of Bayesian Optimization

Bayesian Optimization

Bayesian Optimization

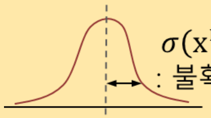
Learning rate	# of iterations	# of hidden layers	# of hidden nodes	...	...	Mini batch size
0.001	1000	3	5	...	...	128
...	...	...	...	...	...	...

X

Generalization performance
100
200

Y

$$x^* = \underset{x^{new_i} \in X}{\operatorname{argmax}}\ 유용성 \left[f(x^{new_i}) \right]$$



다음 후보로 제일 유용한 하이퍼파라미터 집합 구하기

Overview of Bayesian Optimization

Bayesian Optimization

Bayesian Optimization

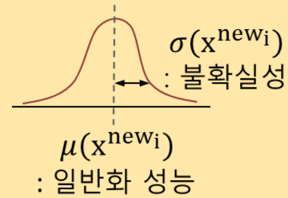
Learning rate	# of iterations	# of hidden layers	# of hidden nodes	…	…	Mini batch size
0.001	1000	3	5	…	…	128
…	…	…	…	…	…	…

X

Generalization performance
100
200

Y

$$x^* = \underset{x^{new_i} \in X}{\text{argmax}} \text{ 유용성} \left[f(x^{new_i}) \right]$$



유용성 ?

Overview of Bayesian Optimization

Bayesian Optimization

Bayesian Optimization

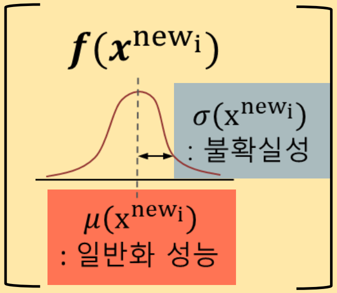
Learning rate	# of iterations	# of hidden layers	# of hidden nodes	…	…	Mini batch size
0.001	1000	3	5	…	…	128
…	…	…	…	…	…	…

X

Generalization performance
100
200

Y

$$x^* = \underset{x^{new_i} \in X}{\operatorname{argmax}} \text{ Acquisition function}$$



착취(exploitation)과 탐험(exploration) 사이 균형 맞추기 중요

Overview of Bayesian Optimization

Bayesian Optimization

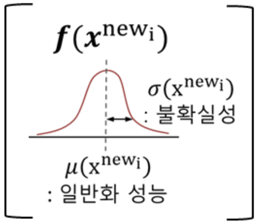
Bayesian Optimization

Learning rate	# of iterations	# of hidden layers	# of hidden nodes	...	...	Mini batch size
0.001	1000	3	5	...	...	128
...	...	...	...	...	...	...
			$\boldsymbol{x}^*$			

Generalization performance
100
200

유용하다고 판단된 하이퍼파라미터 집합 추가, 실제 일반화 성능 확보

$$x^* = \underset{x^{new_i} \in X}{\operatorname{argmax}} \text{ Acquisition function}$$



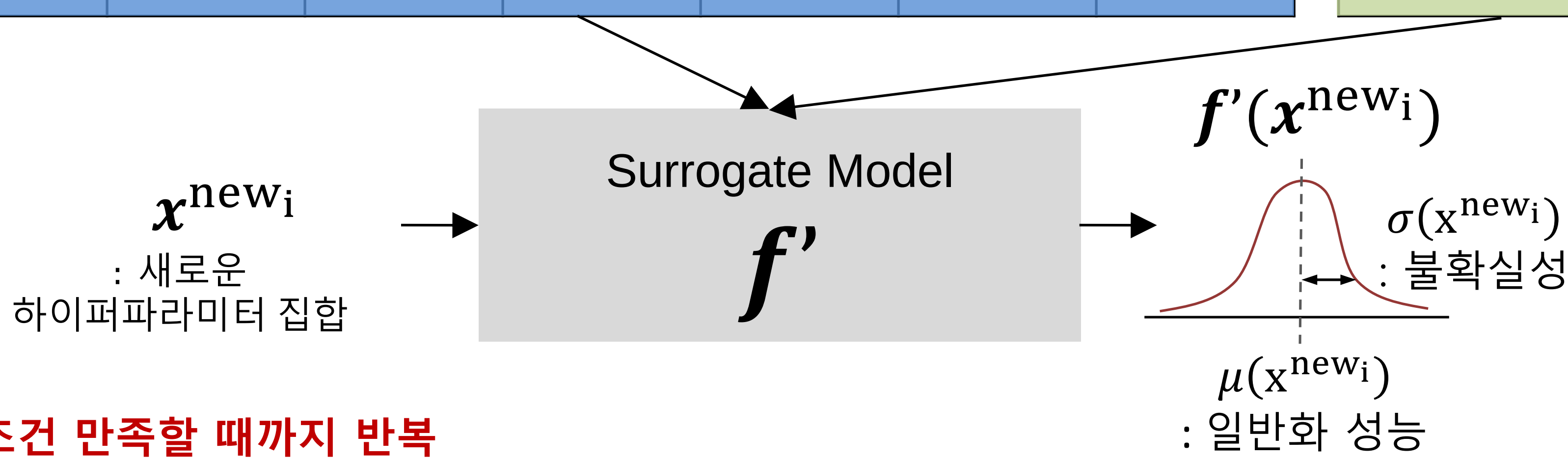
Overview of Bayesian Optimization

Bayesian Optimization

Bayesian Optimization

Learning rate	# of iterations	# of hidden layers	# of hidden nodes	...	...	Mini batch size	Generalization performance
0.001	1000	3	5	...	...	128	100
...	...	...	...	...	...	...	200

X’ Y’



종료 조건 만족할 때까지 반복

Overview of Bayesian Optimization

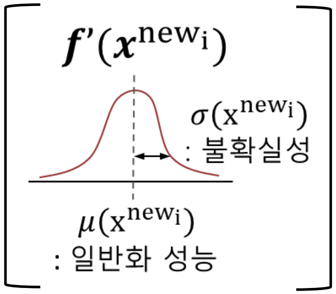
Bayesian Optimization

Bayesian Optimization

Learning rate	# of iterations	# of hidden layers	# of hidden nodes	...	...	Mini batch size	Generalization performance
0.001	1000	3	5	...	...	128	100
...	...	...	...	...	...	...	200

X' **Y'**

$$x^* = \underset{x^{new_i} \in X}{\operatorname{argmax}} \text{ Acquisition function}$$



종료 조건 만족할 때까지 반복

Overview of Bayesian Optimization

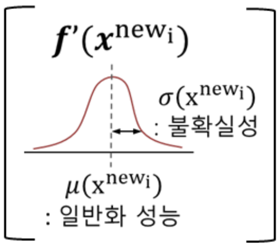
Bayesian Optimization

Bayesian Optimization

Learning rate	# of iterations	# of hidden layers	# of hidden nodes	...	...	Mini batch size	Generalization performance
0.001	1000	3	5	...	...	128	100
...	...	...	...	...	...	...	200
			x^*				

유용하다고 판단된 하이퍼파라미터 집합 추가, 실제 일반화 성능 확보

$$x^* = \underset{x^{new_i} \in X}{\text{argmax}} \text{ Acquisition function}$$



종료 조건 만족할 때까지 반복 (개수, 성능향상 기준)

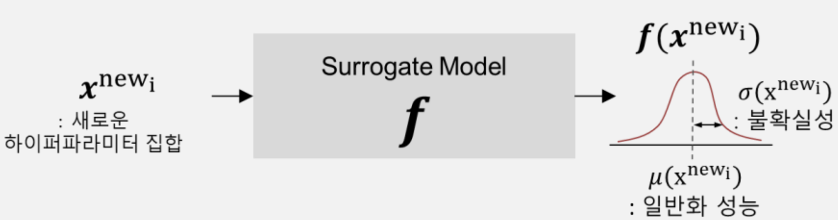
Overview of Bayesian Optimization

Bayesian Optimization

Bayesian Optimization

Surrogate Model

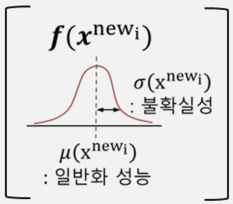
- Gaussian Process Model
- Tree-structured Parzen Estimators
- .....



Acquisition Function

- Maximum Expected Improvement
- Upper Confidence Bound
- Entropy Search
- ...

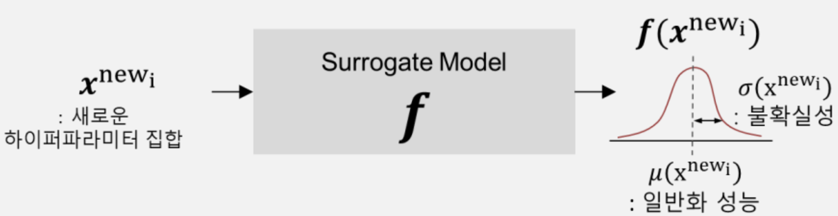
$$x^* = \underset{x^{new_i} \in X}{\text{argmax}} \text{ Acquisition function}$$



Bayesian Optimization

Surrogate Model

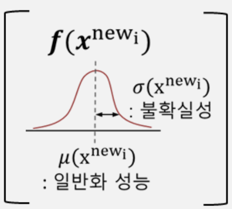
Gaussian Process Regression



Acquisition Function

Maximum Expected Improvement

$$x^* = \underset{x^{new_i} \in X}{\operatorname{argmax}} \text{ Acquisition function}$$



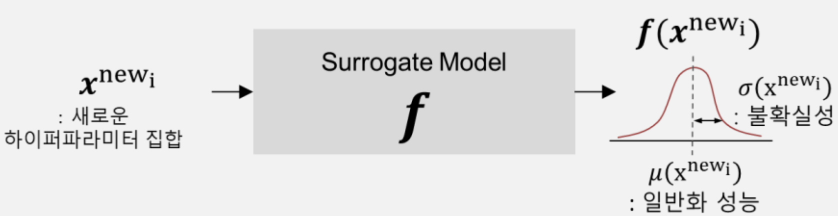
Surrogate Model

Gaussian Process Regression

Bayesian Optimization

Surrogate Model

Gaussian Process Regression



Acquisition Function

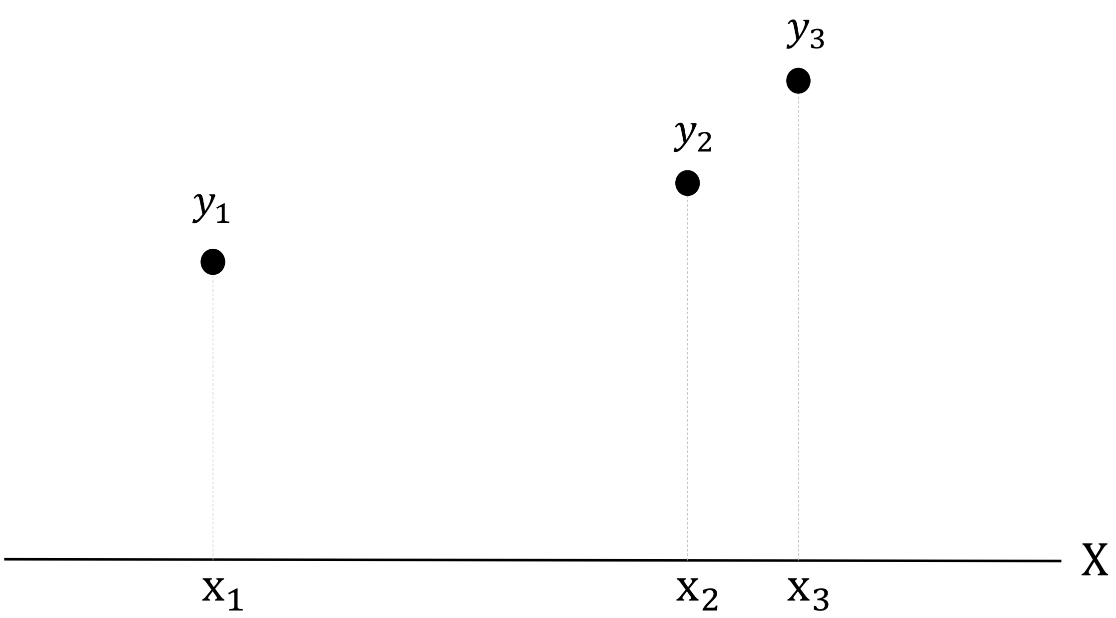
Maximum Expected Improvement

$$x^* = \underset{x^{new_i} \in X}{\operatorname{argmax}} \text{ Acquisition function}$$

Surrogate Model

Gaussian Process Regression

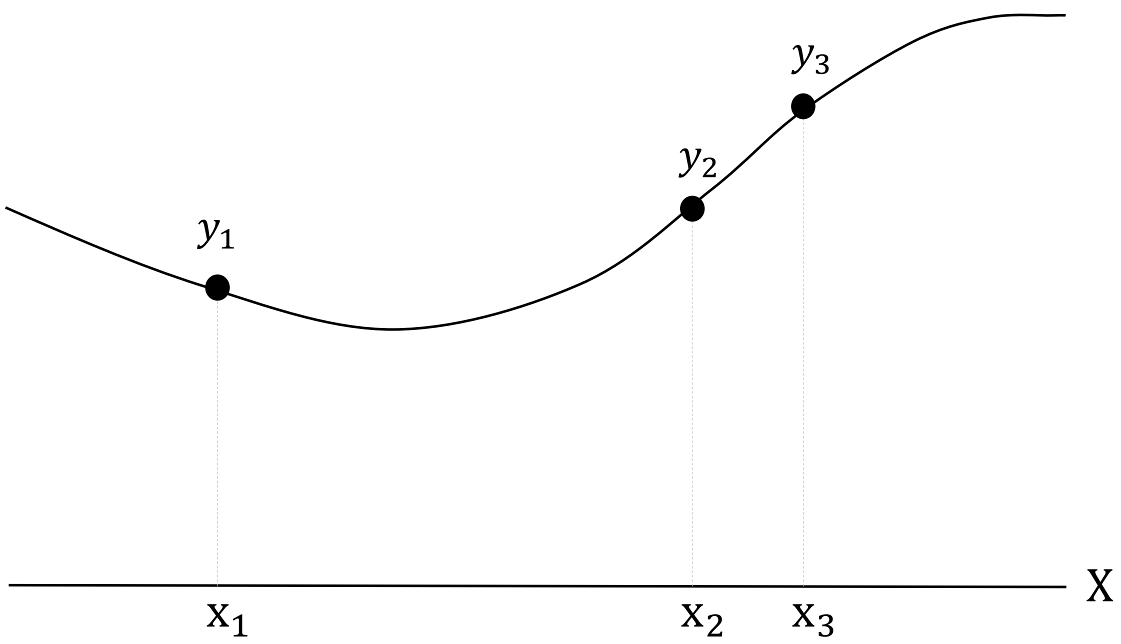
Regression



Surrogate Model

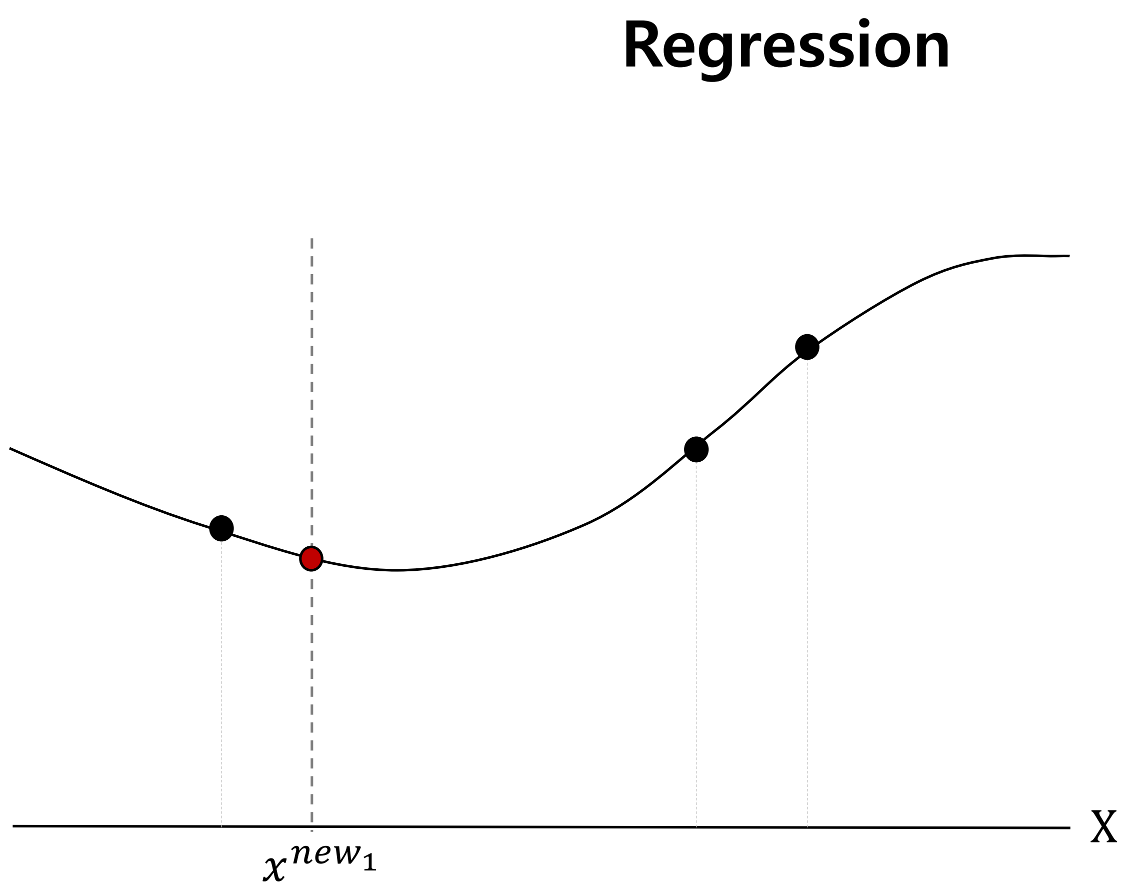
Gaussian Process Regression

Regression

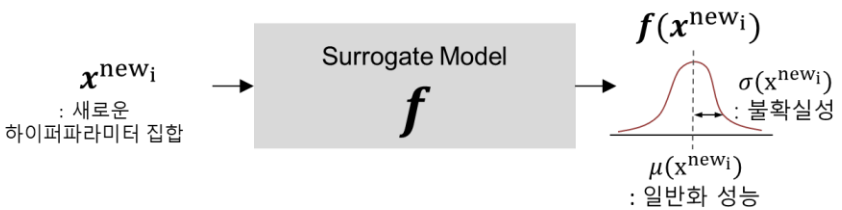


Surrogate Model

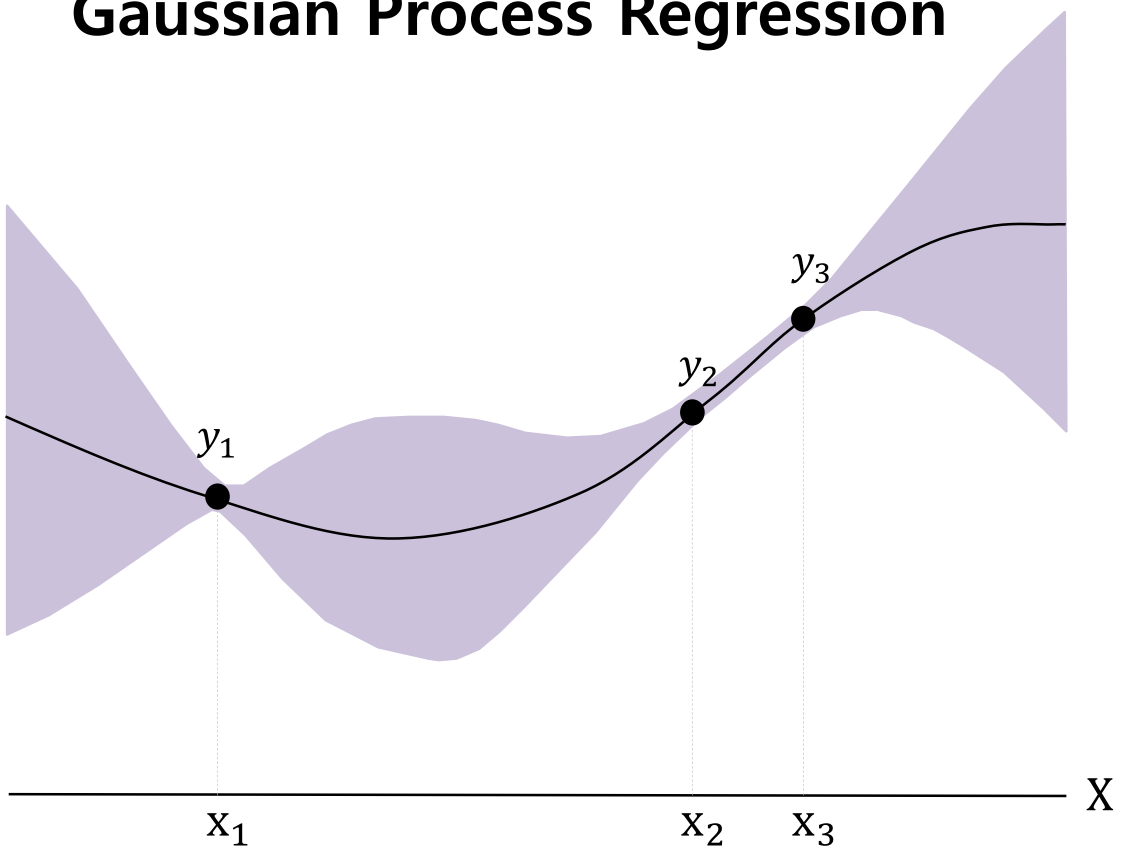
Gaussian Process Regression



Surrogate Model
Gaussian Process Regression

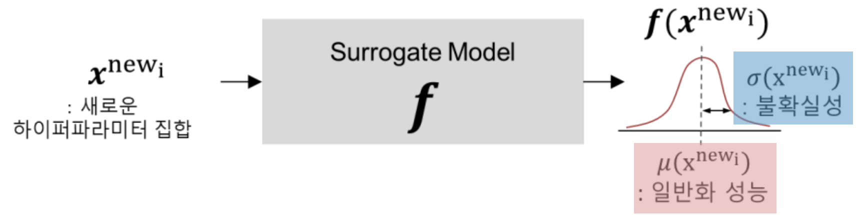


Gaussian Process Regression

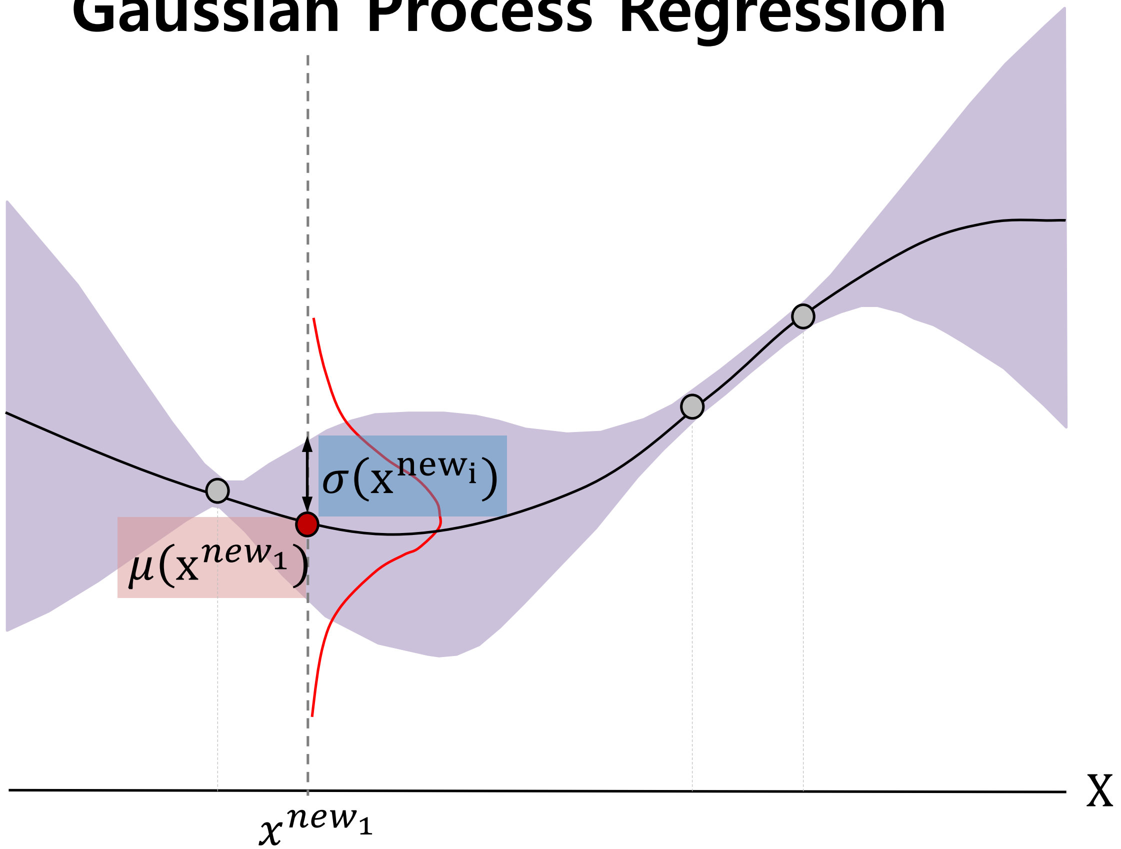


Surrogate Model

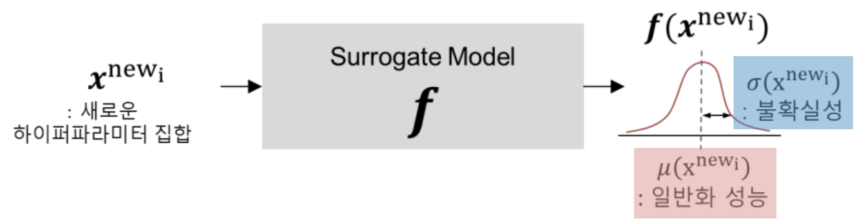
Gaussian Process Regression



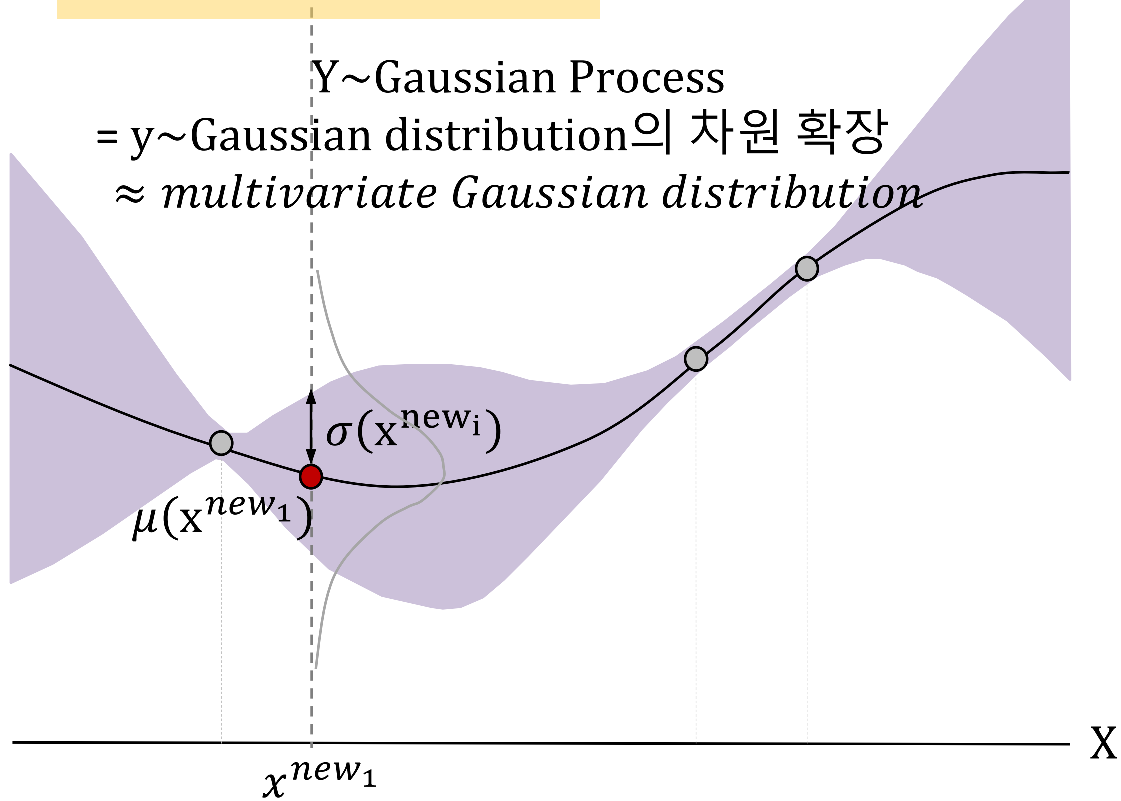
Gaussian Process Regression



Surrogate Model
Gaussian Process Regression



Gaussian Process Regression



Surrogate Model

Gaussian Process Regression

Baye's Rule

$$\underbrace{P(A|B)}_{\text{Posterior}} = \frac{\overbrace{P(B|A)}^{\text{Likelihood}}\overbrace{P(A)}^{\text{Prior}}}{\underbrace{P(B)}_{\text{Evidence}}}$$

Gaussian Process Regression

Y~Gaussian Process (Prior)

= y~Gaussian distribution의 차원 확장

$\approx$ *multivariate Gaussian distribution*

데이터 갯수 : N(확보한 데이터) + 1(평가할 데이터)

$P(Y_{N+1})$~*multivariate Gaussian distribution*

↓ *multivariate Gaussian distribution conditional distribution theorm*

$P(y_{N+1}|Y_N)$?

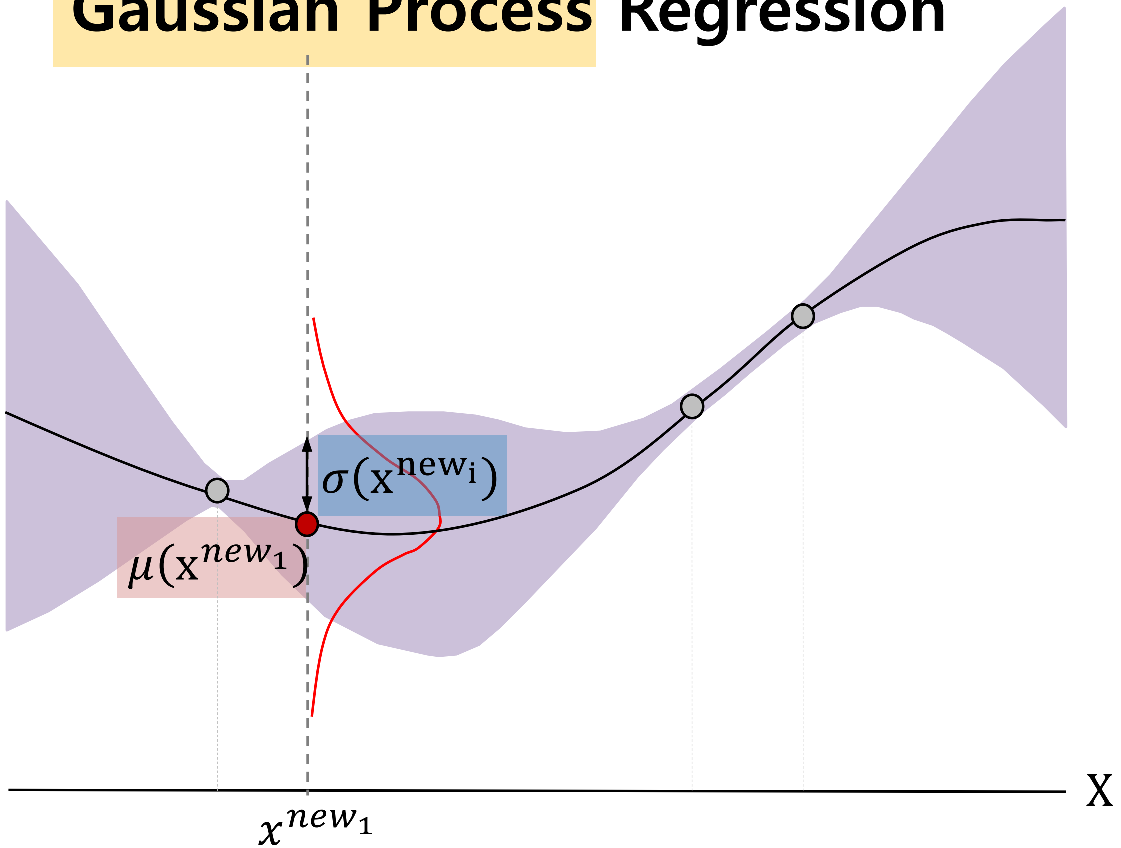
Surrogate Model

Gaussian Process Regression

Baye's Rule

$$\underbrace{P(A|B)}_{\text{Posterior}} = \frac{\overbrace{P(B|A)}^{\text{Likelihood}}\overbrace{P(A)}^{\text{Prior}}}{\underbrace{P(B)}_{\text{Evidence}}}$$

Gaussian Process Regression



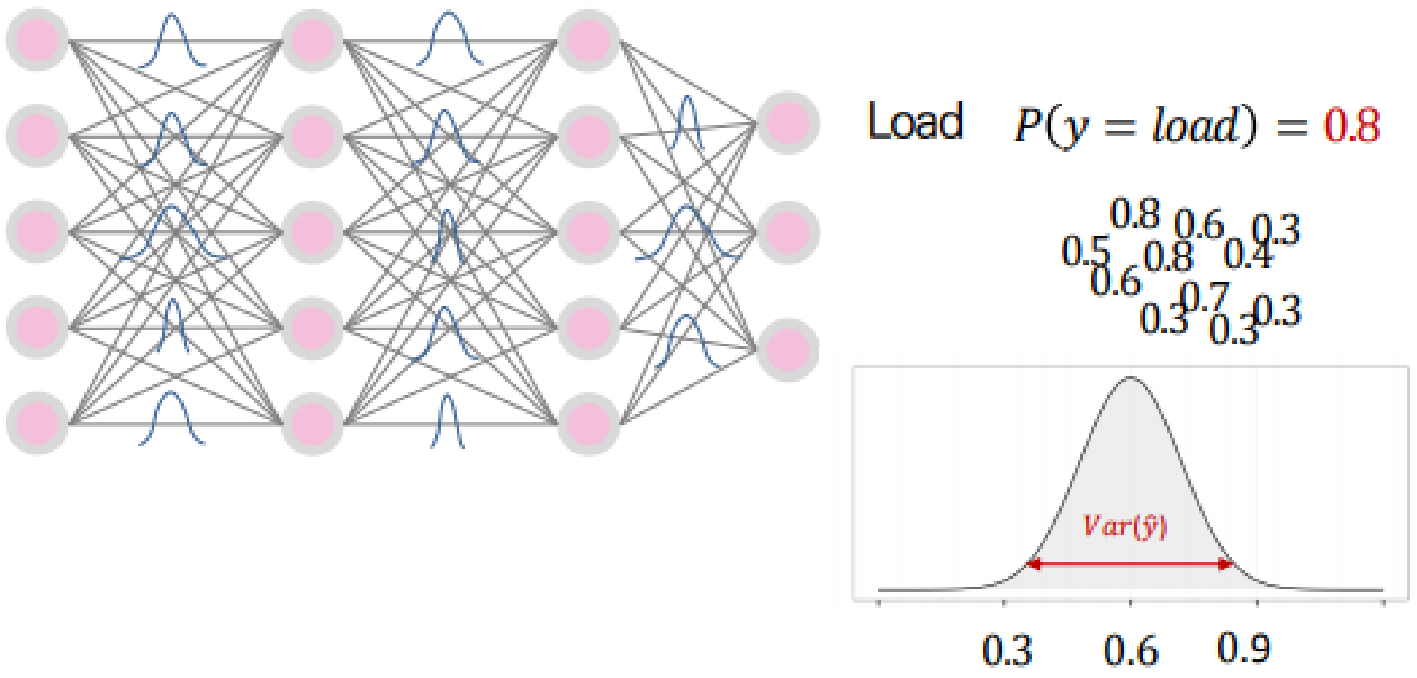
$$P(y_{N+1}|Y_N) = N(t_{N+1}|0 + k^T cov_N^{-1}(T_N - 0),\ c - k^T cov_N^{-1}k)$$

$$\mu_{y_{N+1}} = k^T cov_N^{-1}T_N,\ \sigma_{y_{N+1}} = c - k^T cov_N^{-1}k$$

Surrogate Model

+ alpha

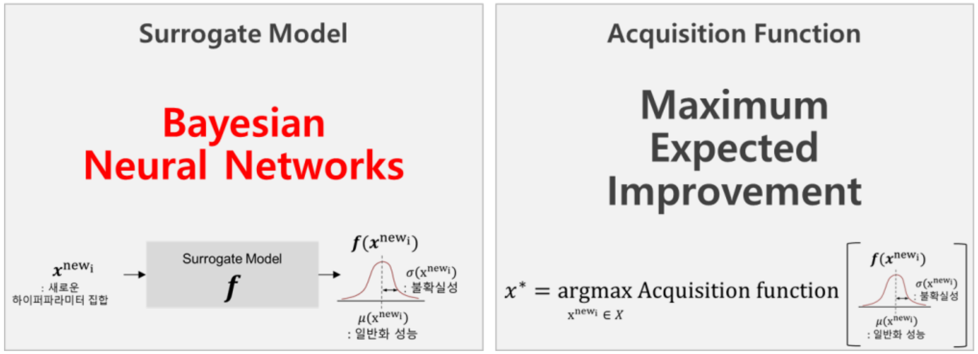
2020년 4월 17일 이지윤 연구원 세미나 [Bayesian Deep Learning for Safe AI] 장표



Surrogate Model

+ alpha

Bayesian Optimization



Bayesian Optimization with Robust Bayesian Neural Networks

Jost Tobias Springenberg Aaron Klein Stefan Falkner Frank Hutter
Department of Computer Science
University of Freiburg
{springj,kleinaa,sfalkner,fh}@cs.uni-freiburg.de

Abstract

Bayesian optimization is a prominent method for optimizing expensive-to-evaluate black-box functions that is widely applied to tuning the hyperparameters of machine learning algorithms. Despite its successes, the prototypical Bayesian optimization approach – using Gaussian process models – does not scale well to either many hyperparameters or many function evaluations. Attacking this lack of scalability and flexibility is thus one of the key challenges of the field. We present a general approach for using flexible parametric models (neural networks) for Bayesian optimization, staying as close to a truly Bayesian treatment as possible. We obtain scalability through stochastic gradient Hamiltonian Monte Carlo, whose robustness we improve via a scale adaptation. Experiments including multi-task Bayesian optimization with 21 tasks, parallel optimization of deep neural networks and deep reinforcement learning show the power and flexibility of this approach.

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

Springenberg, J. T., Klein, A., Falkner, S., & Hutter, F. (2016). Bayesian optimization with robust Bayesian neural networks. In Advances in neural information processing systems (pp. 4134-4142).

Acquisition Function

Maximum Expected Improvement

Bayesian Optimization

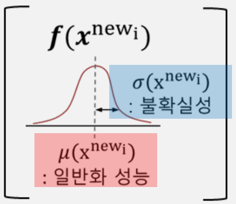
Surrogate Model

Gaussian Process Regression

Acquisition Function

Maximum Expected Improvement

$$x^* = \underset{x^{new_i} \in X}{\text{argmax}} \text{ Acquisition function}$$



Acquisition Function

Maximum Expected Improvement

$$x^* = \underset{x^{new_i} \in X^{new}}{\operatorname{argmax}} \text{ Acquisition function}(x^{new_i})$$

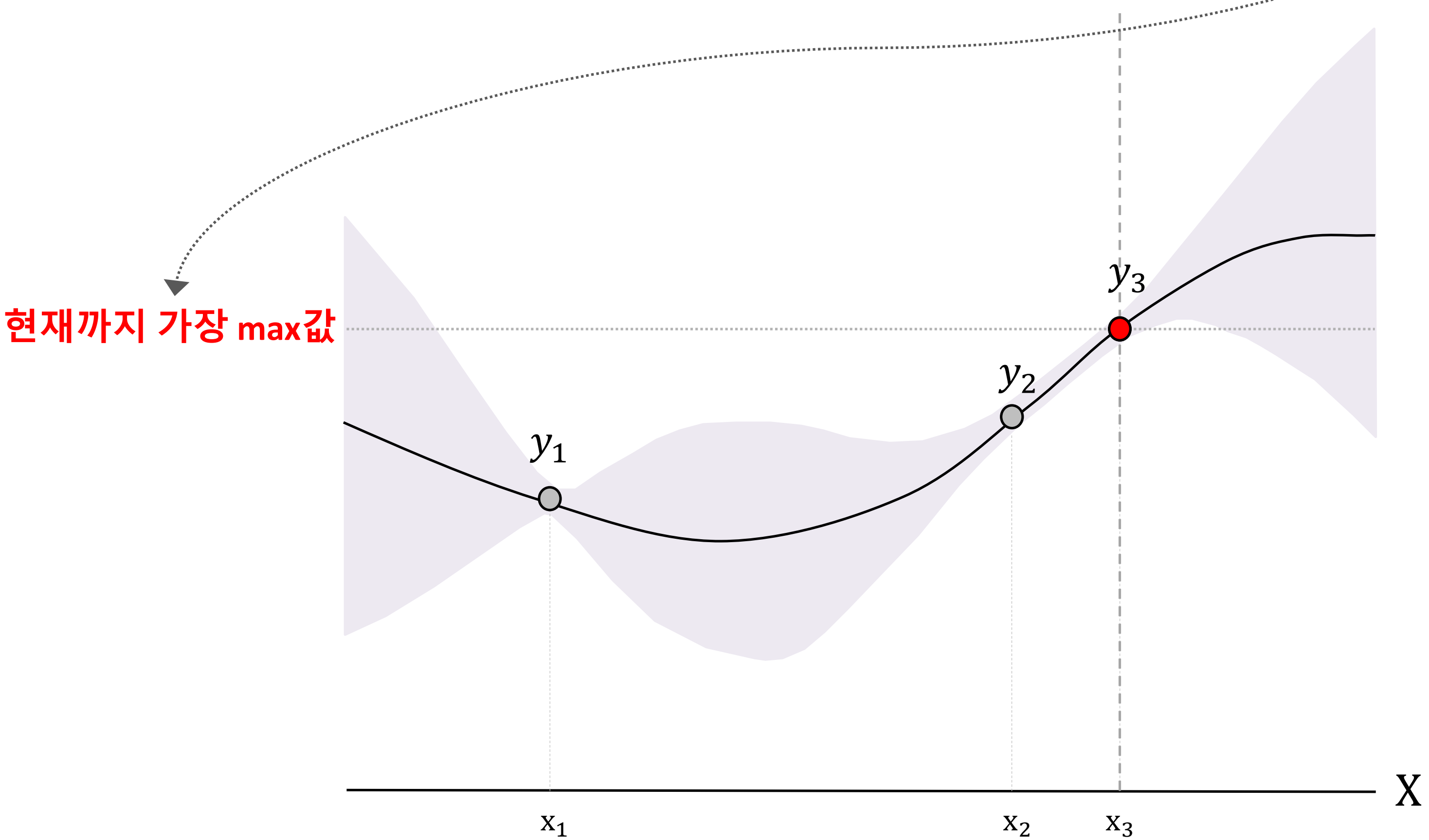
$$x^* = \underset{x^{new_i} \in X^{new}}{\operatorname{argmax}} \text{ Expected Improvement}(x^{new_i})$$

$$:= E\left(\max\left(0, \left[f(x^{new_i}) - \max_{x_j \in X} f\left(x_j\right)\right]\right)\right)$$

Acquisition Function

Maximum Expected Improvement

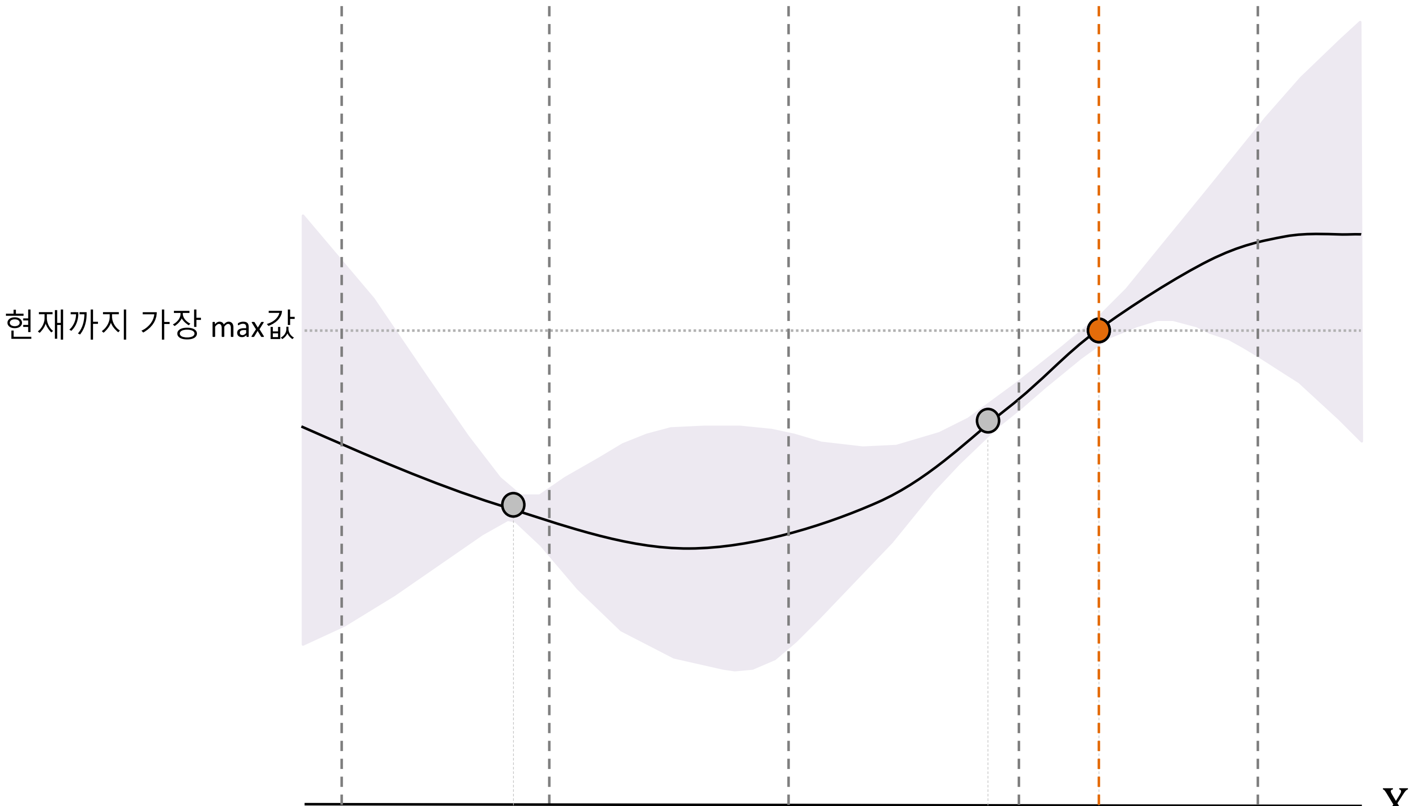
$$x^* = \underset{x^{new_i} \in X^{new}}{\operatorname{argmax}}\, \mathrm{E}\left(\max\left(0, \left[f(x^{new_i}) - \max_{x_j \in X} f(x_j)\right]\right)\right)$$



Acquisition Function

Maximum Expected Improvement

$$x^* = \operatorname{argmax}_{x^{new_i} \in X^{new}} E\left(\max\left(0, \left[f\left(x^{new_i}\right) - \max_{x_j \in X} f\left(x_j\right)\right]\right)\right)$$



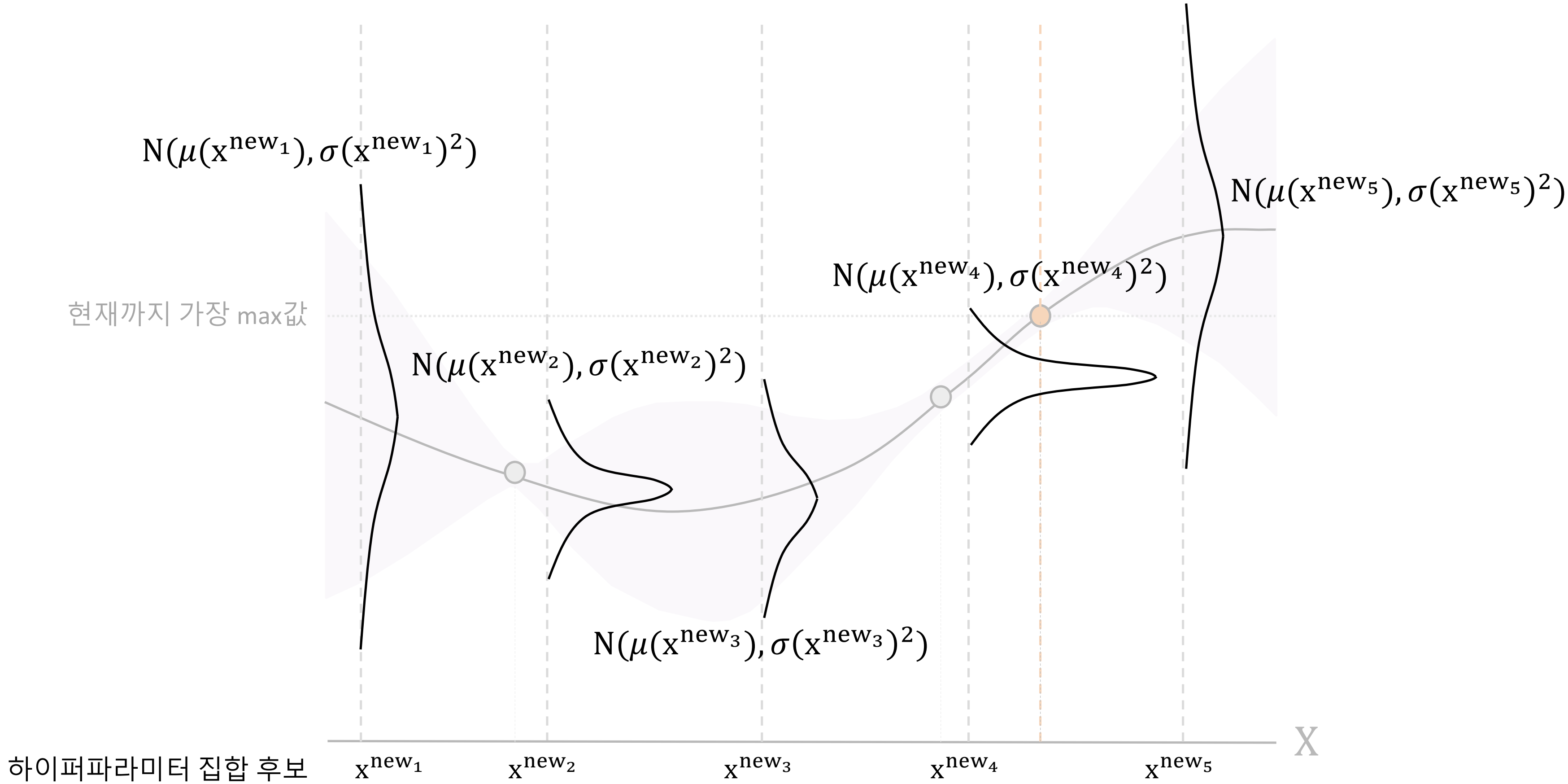
하이퍼파라미터 집합 후보 x^{new_1} x^{new_2} x^{new_3} x^{new_4} x^{new_5}

Acquisition Function

Maximum Expected Improvement

$$x^* = \underset{x^{new_i} \in X^{new}}{\operatorname{argmax}} \mathrm{E}\left(\max\left(0, \left[f(x^{new_i}) - \max_{x_j \in X} f(x_j)\right]\right)\right)$$

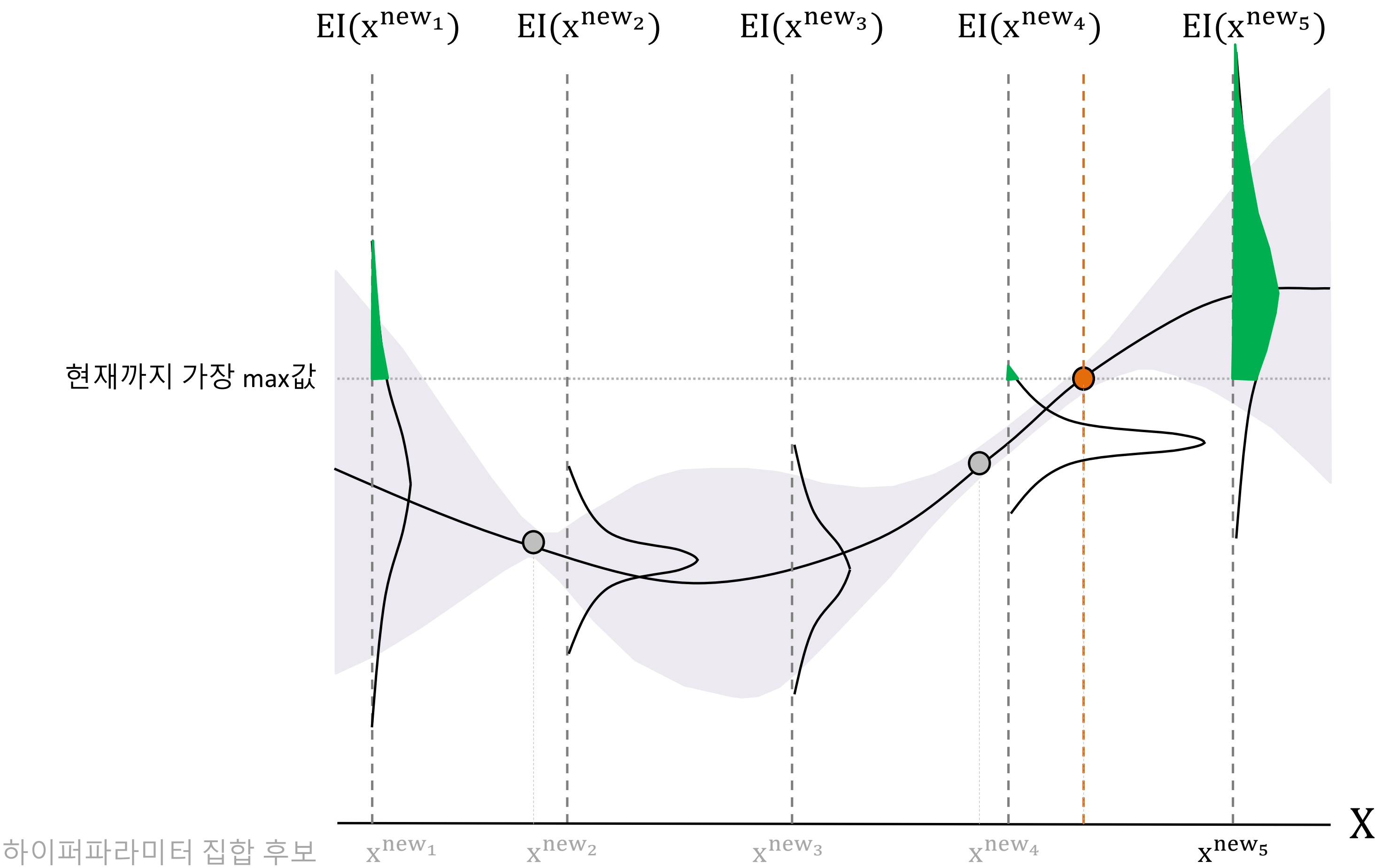
Surrogate Model
Gaussian Process Regression 결과 값



Acquisition Function

Maximum Expected Improvement

$$x^* = \underset{x^{new_i} \in X^{new}}{\operatorname{argmax}} \, E\left(\max\left(0, \left[f\left(x^{new_i}\right) - \max_{x_j \in X} f\left(x_j\right)\right]\right)\right)$$



$= x^*$

Acquisition Function

Maximum Expected Improvement

to maximize it. We define the *expected improvement* as,

$$\mathrm{EI}_n(x) := E_n\left[[f(x) - f_n^*]^+\right] \tag{7}$$

Here, $E_n[\cdot] = E[\cdot|x_{1:n}, y_{1:n}]$ indicates the expectation taken under the posterior distribution given evaluations of f at $x_1, \ldots x_n$. This posterior distribution is given by (3): $f(x)$ given $x_{1:n}, y_{1:n}$ is normally distributed with mean $\mu_n(x)$ and variance $\sigma_n^2(x)$.

The expected improvement can be evaluated in closed form using integration by parts, as described in Jones et al. (1998) or Clark (1961). The resulting expression is

$$\mathrm{EI}_n(x) = [\Delta_n(x)]^+ + \sigma_n(x)\varphi\left(\frac{\Delta_n(x)}{\sigma_n(x)}\right) - |\Delta_n(x)|\Phi\left(\frac{\Delta_n(x)}{\sigma_n(x)}\right), \tag{8}$$

where $\Delta_n(x) := \mu_n(x) - f_n^*$ is the expected difference in quality between the proposed point x and the previous best.

The expected improvement algorithm then evaluates at the point with the largest expected improvement,

$$x_{n+1} = \operatorname{argmax} \mathrm{EI}_n(x), \tag{9}$$

breaking ties arbitrarily. This algorithm was first proposed by Močkus (Močkus, 1975) but was popularized by Jones et al. (1998). The latter article also used the name "Efficient Global Optimization" or EGO.

Implementations use a variety of approaches for solving (9). Unlike the objective f in our original optimization problem (1), $\mathrm{EI}_n(x)$ is inexpensive to evaluate and allows easy evaluation of first- and second-order derivatives. Implementations of the expected improvement algorithm can then use a continuous first- or second-order optimization method to solve (9). For example, one technique that has worked well for the author is to calculate first derivatives and use the quasi-Newton method L-BFGS-B (Liu and Nocedal, 1989).

Frazier, P. I. (2018). A tutorial on bayesian optimization. arXiv preprint arXiv:1807.02811.

Acquisition Function

Maximum Expected Improvement

to maximize it. We define the *expected improvement* as,

$$\mathrm{EI}_n(x) := E_n\left[[f(x) - f_n^*]^+\right] \tag{7}$$

Here, $E_n[\cdot] = E[\cdot|x_{1:n}, y_{1:n}]$ indicates the expectation taken under the posterior distribution given evaluations of f at $x_1, \dots x_n$. This posterior distribution is given by (3): $f(x)$ given $x_{1:n}, y_{1:n}$ is normally distributed with mean $\mu_n(x)$ and variance $\sigma_n^2(x)$.

The expected improvement can be evaluated in closed form using integration by parts, as described in Jones et al. (1998) or Clark (1961). The resulting expression is

최적화방법론 활용 최적해 도출 가능

$$\mathrm{EI}_n(x) = [\Delta_n(x)]^+ + \sigma_n(x)\varphi\left(\frac{\Delta_n(x)}{\sigma_n(x)}\right) - |\Delta_n(x)|\Phi\left(\frac{\Delta_n(x)}{\sigma_n(x)}\right), \tag{8}$$

where $\Delta_n(x) := \mu_n(x) - f_n^*$ is the expected difference in quality between the proposed point x and the previous best.

The expected improvement algorithm then evaluates at the point with the largest expected improvement,

$$x_{n+1} = \operatorname{argmax} \mathrm{EI}_n(x), \tag{9}$$

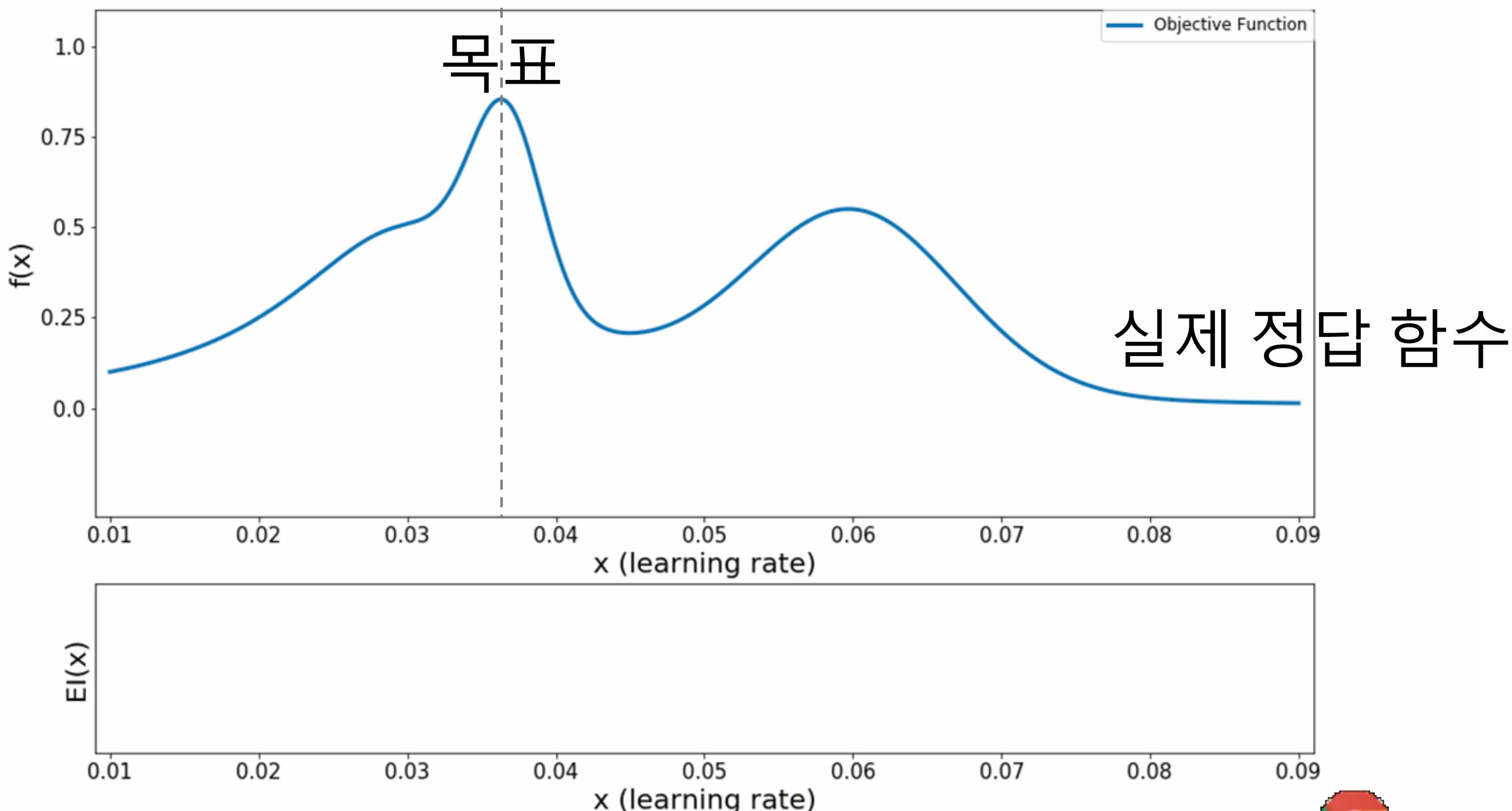
breaking ties arbitrarily. This algorithm was first proposed by Močkus (Močkus, 1975) but was popularized by Jones et al. (1998). The latter article also used the name "Efficient Global Optimization" or EGO.

Implementations use a variety of approaches for solving (9). Unlike the objective f in our original optimization problem (1), $\mathrm{EI}_n(x)$ is inexpensive to evaluate and allows easy evaluation of first- and second-order derivatives. Implementations of the expected improvement algorithm can then use a continuous first- or second-order optimization method to solve (9). For example, one technique that has worked well for the author is to calculate first derivatives and use the quasi-Newton method L-BFGS-B (Liu and Nocedal, 1989).

Frazier, P. I. (2018). A tutorial on bayesian optimization. arXiv preprint arXiv:1807.02811.

Example

- ● 관측 데이터
- ---------- 예측값 (Surrogate Model : GPR의 평균값)
- 95% 신뢰구간 (Surrogate Model : GPR의 표준편차 활용)
- Acquisition Function (EI) 값
- ☆ EI에서 추출된 다음 후보 값

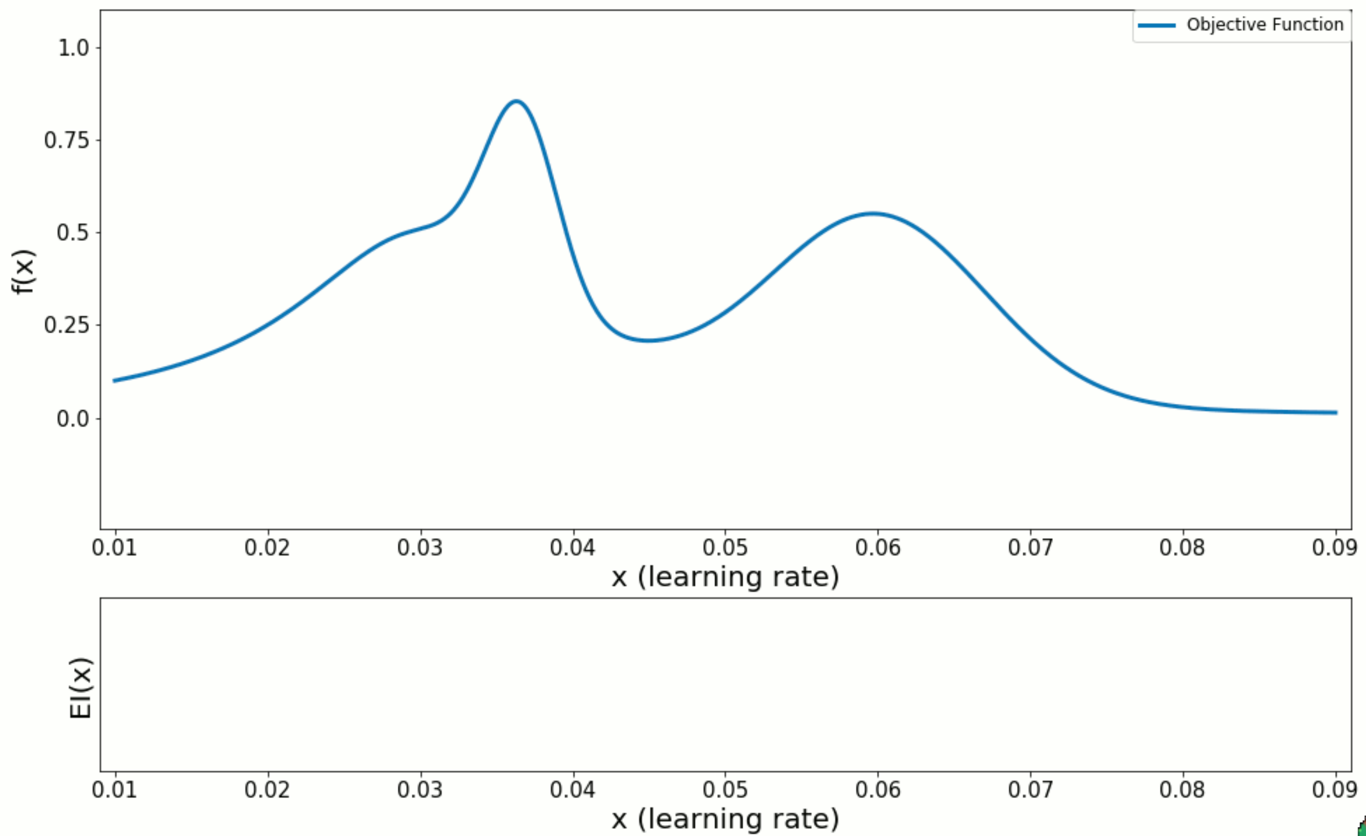


http://research.sualab.com/introduction/practice/2019/02/19/bayesian-optimization-overview-1.html

Bayesian Optimization Practices.html

Example

- 관측 데이터
- 예측값 (Surrogate Model : GPR의 평균값)
- 95% 신뢰구간 (Surrogate Model : GPR의 표준편차 활용)
- Acquisition Function (EI) 값
- EI에서 추출된 다음 후보 값



http://research.sualab.com/introduction/practice/2019/02/19/bayesian-optimization-overview-1.html

Bayesian Optimization Practices.html

Applications

❖ 최적 하중 도출을 위한 다음 시뮬레이션 세팅 값?

실제 하중과 시뮬레이터 하중 차이를 최소화 시키는 다음 세팅 값은?

로드 시뮬레이터

시뮬레이션 세팅 값 1 → 시뮬레이터 하중 1 → 타겟 하중과 시뮬레이터 하중 차이1

시뮬레이션 세팅 값 2 → 시뮬레이터 하중 2 → 타겟 하중과 시뮬레이터 하중 차이2

시뮬레이션 세팅 값 3 → 시뮬레이터 하중 3 → 타겟 하중과 시뮬레이터 하중 차이3

타겟 하중

Applications

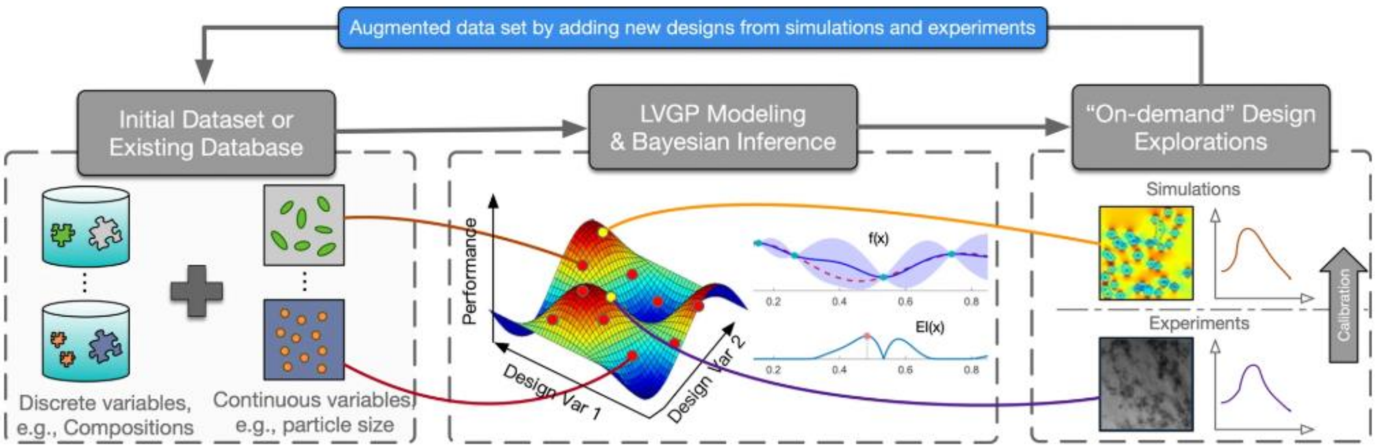
❖ 원하는 성질을 갖는 신물질 탐색 (화학, 의료, 재료 등)

✓ 후보 분자 집합에서 적은 수의 탐색만으로 원하는 성질에 가까운 분자를 찾음

❖ 최적 설계 값 도출

✓ 원하는 특성을 갖춘 최적 설계 값을 적은 수의 탐색만으로 찾음

From: Bayesian Optimization for Materials Design with Mixed Quantitative and Qualitative Variables



Bayesian optimization framework for data-driven materials design.

Zhang, Y., Apley, D. W., & Chen, W. (2020). Bayesian Optimization for Materials Design with Mixed Quantitative and Qualitative Variables. Scientific Reports, 10(1), 1-13.

Reference

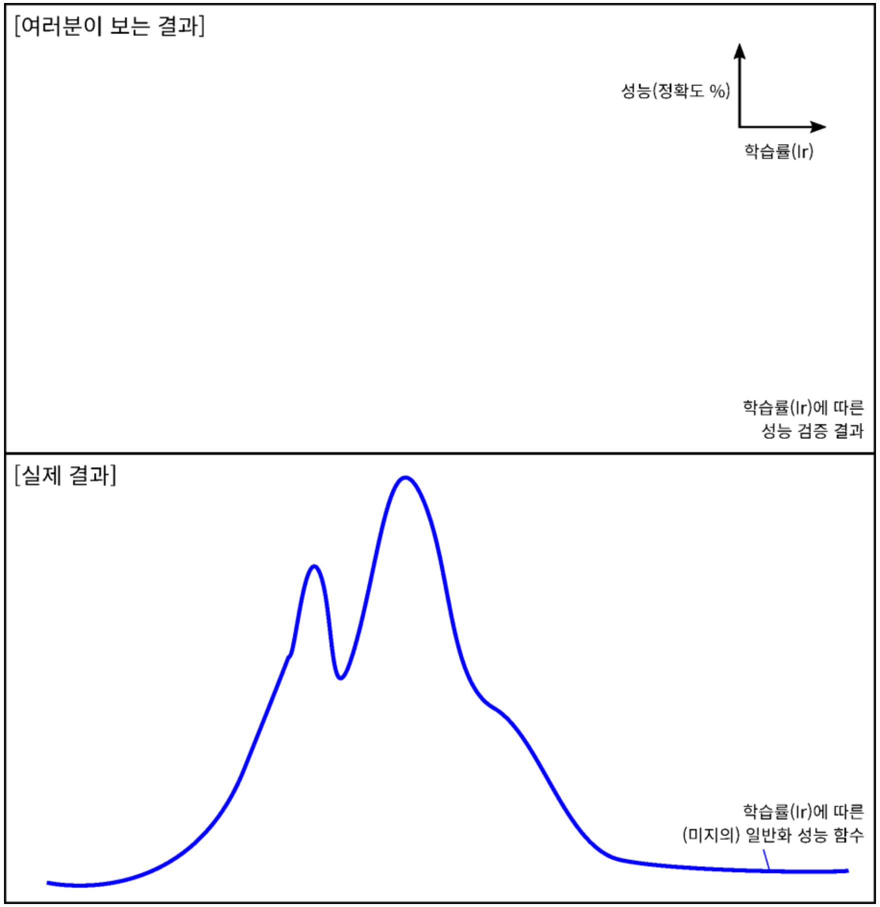
- http://research.sualab.com/introduction/practice/2019/02/19/bayesian-optimization-overview-1.html
- http://research.sualab.com/introduction/practice/2019/04/01/bayesian-optimization-overview-2.html
- http://sanghyukchun.github.io/99/
- https://www.edwith.org/aiml-adv/joinLectures/14705
- https://www.edwith.org/bayesiandeeplearning/joinLectures/14426


Q
&
A

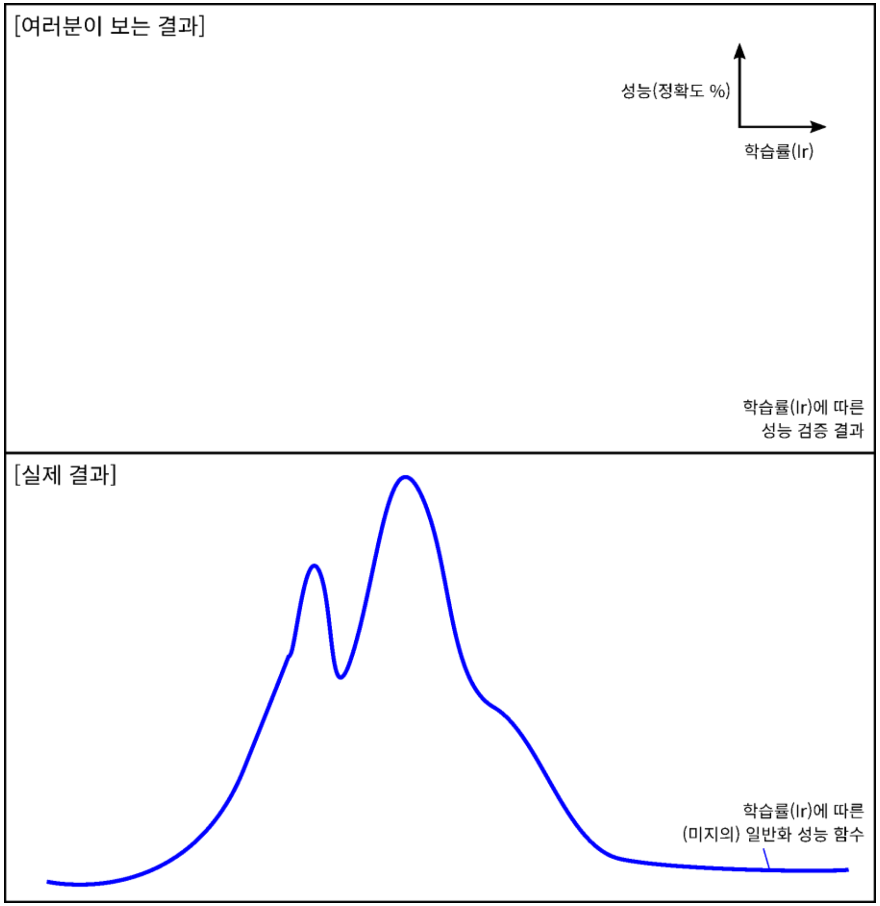
Appendix: Introduction

Difficulties of Hyperparameter Tuning

Grid search (균등한 전역 탐색)



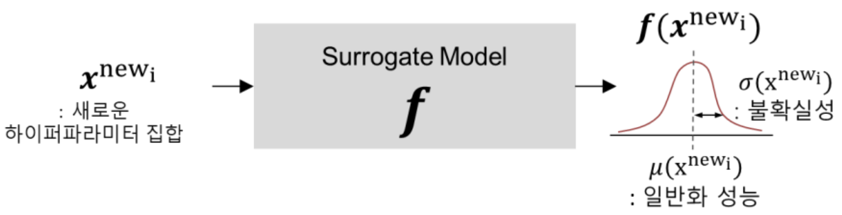
Random search (랜덤 샘플링)



http://research.sualab.com/introduction/practice/2019/02/19/bayesian-optimization-overview-1.html

Appendix : Surrogate Model

Gaussian Process Regression



Gaussian Process Regression

Y~Gaussian Process

= y~Gaussian distribution의 차원 확장

$\approx$ *multivariate Gaussian distribution*

데이터 갯수 : N(확보한 데이터) + 1(평가할 데이터)

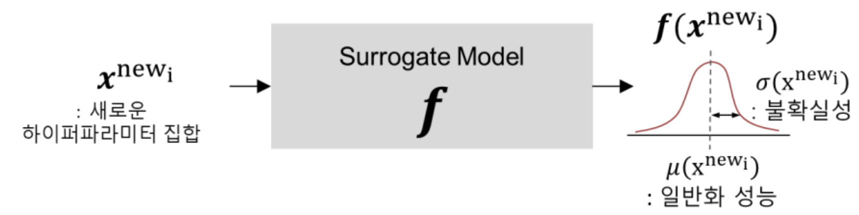
$P(Y_{N+1})\sim multivariate\ Gaussian\ distribution$

multivariate Gaussian distribution theorm

$P(y_{N+1}|Y_N)$?

Appendix : Surrogate Model

Gaussian Process Regression



Linear Regression with Basis Function

- Linear regression : $y(x) = w^T\phi(x)$
  - w : weight vector of M dimension
  - Or, $Y = \Phi w$
    - Φ : called a design matrix revealing the relation of the weight vector and the input vector
    - $\Phi_{nk} = \phi_k(x_n)$
- Previously, w is modeled as deterministic values
  - Now, w is considered to be also probabilistically distributed values
  - $P(w) = N(w|0, \alpha^{-1}I)$
    - Normal distribution with zero mean and α precision (or, α^{-1} variance)
- Now, w probability distribution → Y probability distribution
  - $E[Y] = E[\Phi w] = \Phi E[w] = 0$
  - $cov[Y] = E[(Y-0)(Y-0)^T) = E[YY^T]$

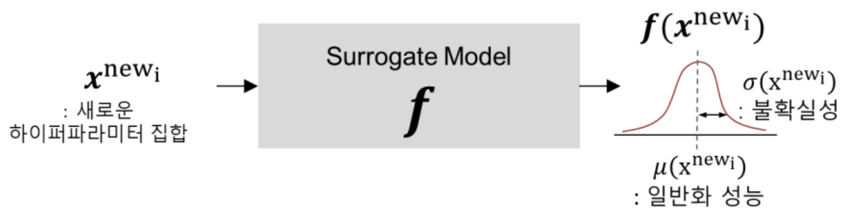
$$= E[\Phi w w^T \Phi^T] = \Phi E[ww^T]\Phi^T = \frac{1}{\alpha}\Phi\Phi^T$$

- $K_{nm} = k(x_n, x_m) = \frac{1}{\alpha}\phi(x_n)^T\phi(x_m)$
  - K : Gram matrix, k : kernel function
- $P(Y) = N(Y|0, K)$

https://www.edwith.org/aiml-adv/joinLectures/14705

Appendix : Surrogate Model

Gaussian Process Regression



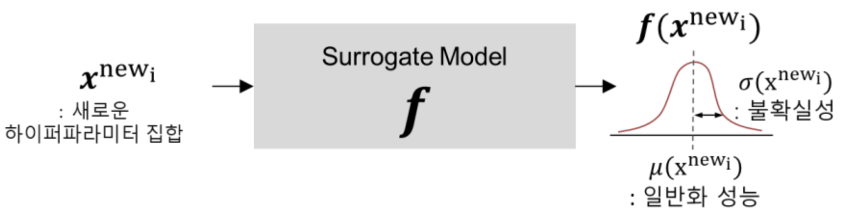
Modeling Noise with Gaussian Distribution

- $P(Y) = N(Y|0, K)$
  - $K_{nm} = k(x_n, x_m) = \frac{1}{\alpha}\phi(x_n)^T\phi(x_m)$
- $t_n = y_n + e_n$
  - t_n : Observed value with noise
  - y_n : Latent, error-free value
  - e_n : Error term distributed by following the Gaussian distribution
- $P(t_n|y_n) = N(t_n|y_n, \beta^{-1})$
  - β : Hyper-parameter of the error precision (or, variance considering the invert)
- $P(T|Y) = N(T|Y, \beta^{-1}I_N)$
  - $T = (t_1, \dots, t_N)^T, Y = (y_1, \dots, y_N)^T$
  - Assuming that the error terms are independent
- $P(T) = \int P(T|Y)P(Y)dY = \int N(T|Y, \beta^{-1}I_N)N(Y|0, K)dY$

https://www.edwith.org/aiml-adv/joinLectures/14705

Appendix : Surrogate Model

Gaussian Process Regression



Marginal Gaussian Distribution

- $P(T) = \int P(T|Y)P(Y)dY = \int N(T|Y, \beta^{-1}I_N)N(Y|0, K)dY$
- $P(T|Y)P(Y) = P(T, Y) = P(Z)$
- $lnP(Z) = lnP(Y) + lnP(T|Y)$

$$= -\frac{1}{2}(Y-0)^T K^{-1}(Y-0) - \frac{1}{2}(T-Y)^T \beta I_N (T-Y) + const.$$
$$= -\frac{1}{2}Y^T K^{-1} Y - \frac{1}{2}(T-Y)^T \beta I_N (T-Y) + const.$$

$$N(x|\mu,\Sigma) = \frac{1}{(2\pi)^{D/2}}\frac{1}{|\Sigma|^{1/2}}\exp(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu))$$

- Second order term of $lnP(Z)$
  - $-\frac{1}{2}Y^T K^{-1} Y - \frac{\beta}{2}T^T T + \frac{\beta}{2}TY + \frac{\beta}{2}YT - \frac{\beta}{2}Y^T Y$
    $$= -\frac{1}{2}\begin{pmatrix} Y \\ T \end{pmatrix}^T \begin{pmatrix} K^{-1} + \beta I_N & -\beta I_N \\ -\beta I_N & \beta I_N \end{pmatrix}\begin{pmatrix} Y \\ T \end{pmatrix} = -\frac{1}{2}Z^T R Z$$
  - R becomes the precision matrix of Z
    - $M = (K^{-1} + \beta I_N - \beta I_N(\beta I_N)^{-1}\beta I_N)^{-1} = K$

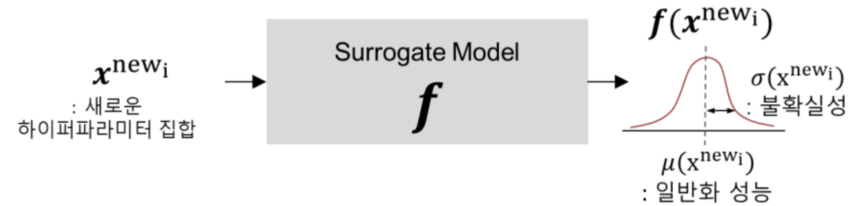
$$\begin{pmatrix} A & B \\ C & D \end{pmatrix}^{-1} = \begin{pmatrix} M & -MBD^{-1} \\ -D^{-1}CM & D^{-1} + D^{-1}CMBD^{-1} \end{pmatrix}$$
$$M = (A - BD^{-1}C)^{-1}$$

  - $R^{-1} = \begin{pmatrix} K & K\beta I_N(\beta I_N)^{-1} \\ (\beta I_N)^{-1}\beta I_N K & (\beta I_N)^{-1} + (\beta I_N)^{-1}\beta I_N K \beta I_N (\beta I_N)^{-1} \end{pmatrix}$
    $$= \begin{pmatrix} K & K \\ K & (\beta I_N)^{-1} + K \end{pmatrix}$$
- First order term of $lnP(Z)$ → None
- $P(Z) = N(Z|0, R^{-1})$

https://www.edwith.org/aiml-adv/joinLectures/14705

Appendix : Surrogate Model

Gaussian Process Regression



Marginal and Conditional Distribution of P(T)

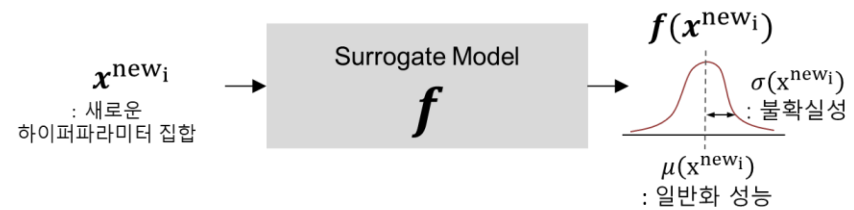
- $P(T) = \int P(T|Y)P(Y)dY = \int N(T|Y, \beta^{-1}I_N)N(Y|0, K)dY$
  - $P(T|Y)P(Y) = P(Y, T) = P(Z)$
  - $P(Y, T) = N\left(Y, T \middle| (0 \quad 0), \begin{pmatrix} K & K \\ K & (\beta I_N)^{-1} + K \end{pmatrix}\right)$
    - Precision Matrix $= \begin{pmatrix} K^{-1} + \beta I_N & -\beta I_N \\ -\beta I_N & \beta I_N \end{pmatrix}$
- Two theorems on multivariate normal distributions
  - Given $X = [X_1 \quad X_2]^T, \mu = [\mu_1 \quad \mu_2]^T, \Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}$
  - $P(X_1) = N(X_1|\mu_1, \Sigma_{11})$
  - $P(X_1|X_2) = N(X_1|\mu_1 + \Sigma_{12}{\Sigma_{22}}^{-1}(X_2 - \mu_2), \Sigma_{11} - \Sigma_{12}{\Sigma_{22}}^{-1}\Sigma_{21})$
- $P(T) = N(T|0, (\beta I_N)^{-1} + K)$
  - $K_{nm} = k(x_n, x_m) = \frac{1}{\alpha}\phi(x_n)^T\phi(x_m)$
  - One example → $k(x_n, x_m) = \theta_0 \exp\left(-\frac{\theta_1}{2}||x_n - x_m||^2\right) + \theta_2 + \theta_3 x_n^T x_m$
- Our ultimate question as a regression problem is
  - $P(t_{N+1}|T_N)$=? → $P(T_{N+1})$=!

https://www.edwith.org/aiml-adv/joinLectures/14705

Data Mining Quality Analytics

Appendix : Surrogate Model

Gaussian Process Regression



Mean and Covariance of $P(t_{N+1}|T_N)$

- $P(T) = N(T|0, (\beta I_N)^{-1} + K)$
  - $K_{nm} = k(x_n, x_m)$
- $P(T_{N+1}) = N(T|0, cov)$

$$cov = \left[\begin{pmatrix} K_{11}+\beta & K_{12} & \cdots & K_{1N} & K_{1(N+1)} \\ K_{21} & K_{22}+\beta & & K_{2N} & K_{2(N+1)} \\ \vdots & & \ddots & & \vdots \\ K_{N1} & K_{N2} & & K_{NN}+\beta & K_{N(N+1)} \\ K_{(N+1)1} & K_{(N+1)2} & \cdots & K_{(N+1)N} & K_{(N+1)(N+1)}+\beta \end{pmatrix}\right]$$

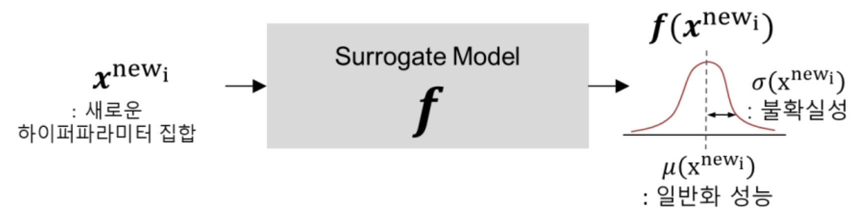
$$cov_{N+1} = \begin{bmatrix} cov_N & k \\ k^T & c \end{bmatrix}$$

- Future distribution given the past data
  - Remember the theorem introduced earlier
    - $P(X_1|X_2) = N(X_1|\mu_1 + \Sigma_{12}{\Sigma_{22}}^{-1}(X_2-\mu_2), \Sigma_{11} - \Sigma_{12}{\Sigma_{22}}^{-1}\Sigma_{21})$
  - $P(t_{N+1}|T_N) = \mathrm{N}(t_{N+1}|0 + k^T {cov_N}^{-1}(T_N - 0), \mathrm{c} - k^T {cov_N}^{-1}k)$
  - $\mu_{t_{N+1}} = k^T {cov_N}^{-1} T_N, {\sigma^2}_{t_{N+1}} = \mathrm{c} - k^T {cov_N}^{-1} k$

https://www.edwith.org/aiml-adv/joinLectures/14705

Appendix : Surrogate Model

Gaussian Process Regression



Two theorems on multivariate normal distributions

- $Given\ X = [X_1\ X_2]^T,\ \ \mu = [\mu_1\ \mu_2]^T, \Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}$
- $P(X_1) = N(X_1|\mu_1,\ \Sigma_{11})$
- $P(X_1|X_2) = N(X_1|\mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(X_2 - \mu_2),\ \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21})$

$$P(Y_{N+1}) = N(Y|0, cov)$$

$$cov = \begin{bmatrix} K_{11}+\beta & K_{12} & \cdots & K_{1N} & K_{1(N+1)} \\ K_{21} & K_{22}+\beta & \cdots & K_{2N} & K_{2(N+1)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ K_{N1} & K_{N2} & \cdots & K_{NN}+\beta & K_{N(N+1)} \\ K_{(N+1)1} & K_{(N+1)2} & \cdots & K_{(N+1)N} & K_{(N+1)(N+1)}+\beta \end{bmatrix}$$

$$cov_{N+1} = \begin{bmatrix} cov_N & k \\ k^T & c \end{bmatrix}$$

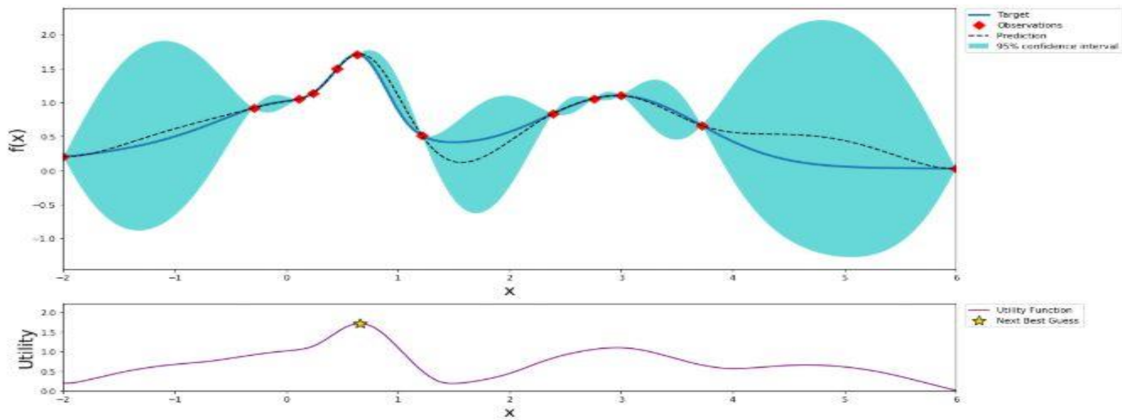
$$P(y_{N+1}|Y_N) = N(t_{N+1}|0 + k^T cov_N^{-1}(T_N - 0),\ c - k^T cov_N^{-1} k)$$

$$\mu_{y_{N+1}} = k^T cov_N^{-1} T_N,\ \ \sigma_{y_{N+1}} = c - k^T cov_N^{-1} k$$

Overview of Bayesian Optimization

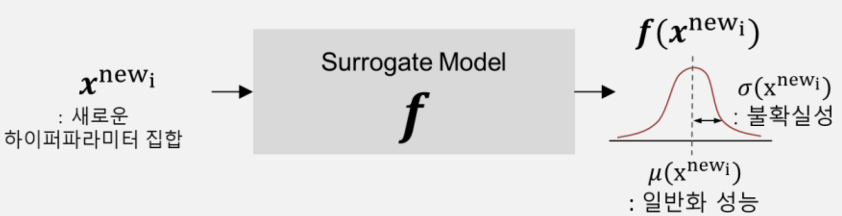
Bayesian Optimization

Bayesian Optimization



Surrogate Model

Gaussian Process Regression



Acquisition Function

Maximum Expected Improvement

$$x^* = \underset{x^{new_i} \in X}{\text{argmax}} \text{ Acquisition function}$$