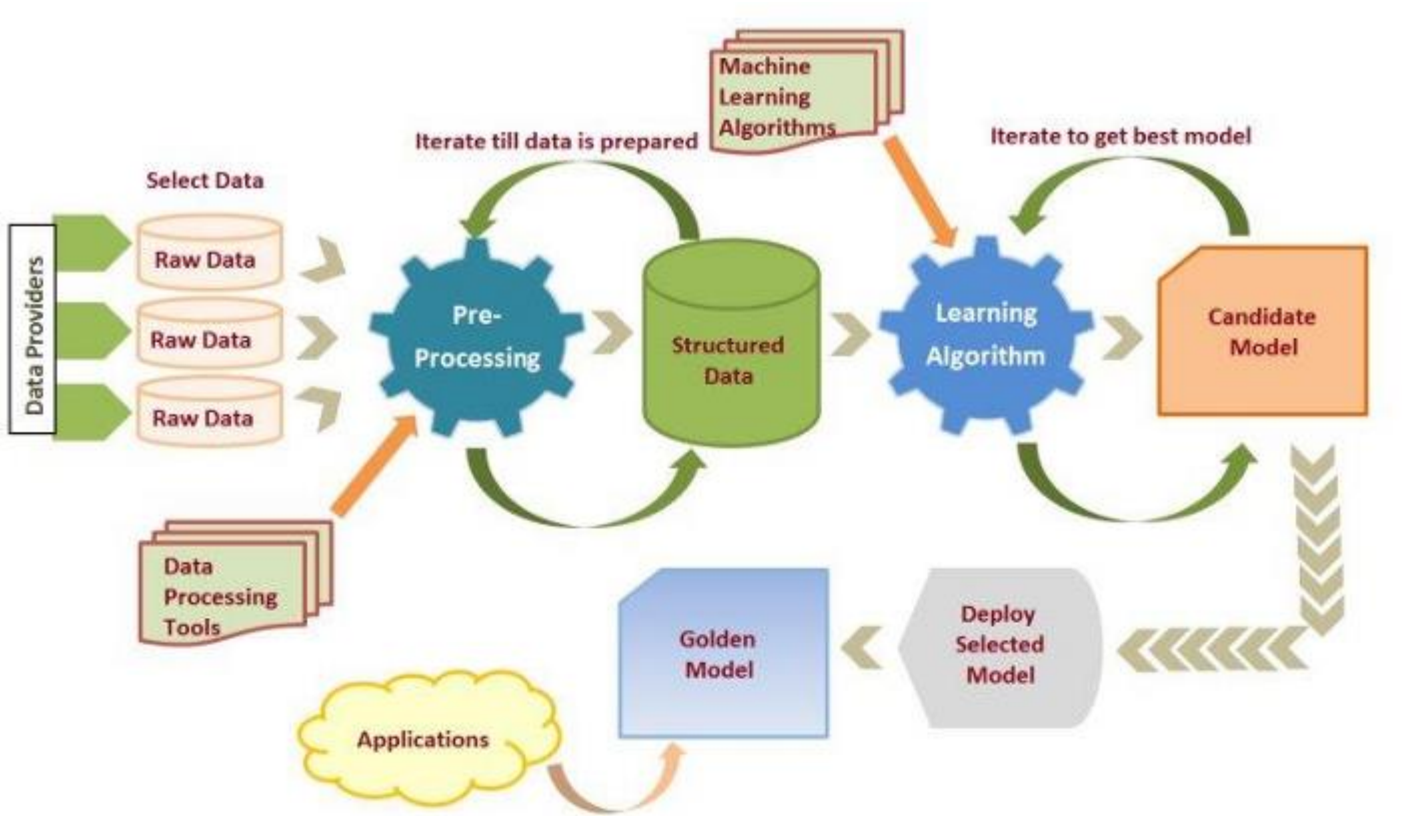
Bayesian Optimization

ML procedure



https://ichi.pro/ko/meosin-leoning-ui-yuhyeong-gwa-jeolcha-9596720397791

Model Improvement (Model Tuning)

- Goal
  - Enhance the performance of the model
- Methods
  - Prepare and use more data: need cost
  - Try another (deep learning) model: very academic
  - Adjust hyperparameters: time consuming

Hyperparameters

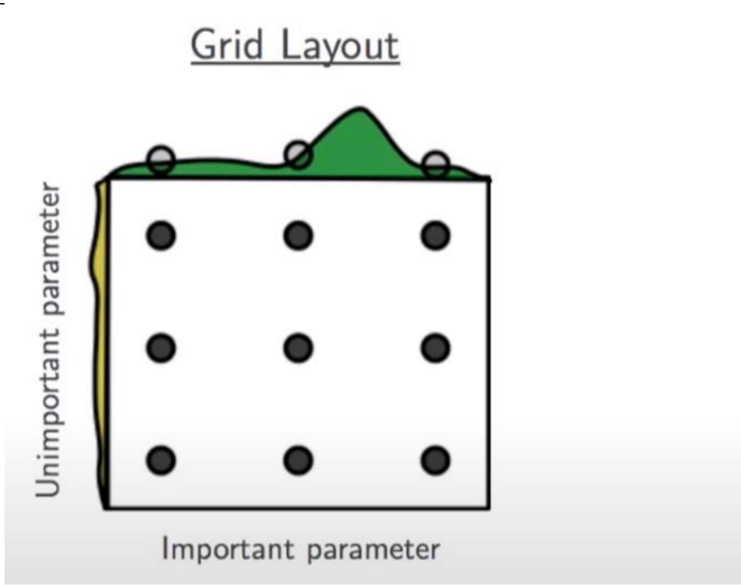
- Used for learning (training)
- Need to be set before training
  - Learning rate
  - Number of layers
  - Batch size
  - Optimizer: SGD, momentum, adam
  - Activation functions: sigmoid, tanh, ReLu

Why tuning (optimizing) hyperparameter?

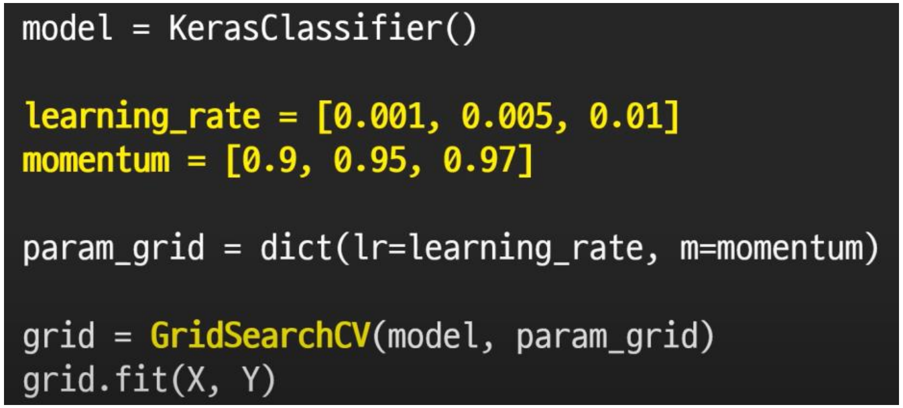
- Achieve high performance
- Reproducibility of published results
- Automatic tuning is required
- For non-expert users

Searching parameters

- What if we search all



James Bergstra and Yoshua Bengio (2012)



```
model = KerasClassifier()

learning_rate = [0.001, 0.005, 0.01]
momentum = [0.9, 0.95, 0.97]

param_grid = dict(lr=learning_rate, m=momentum)

grid = GridSearchCV(model, param_grid)
grid.fit(X, Y)
```

Hyper parameter Tuning Method based on Sampling for Optimal LSTM model(2019)

- Hyperparameter List
  - Learning rate (constant)
  - Optimizer (categorical)
  - Activation function of output layer (categorical)

1. Searching Order and searching space

   ① Learning rate
   : 0.01, 0.005, 0.001

   ② Optimizer
   : SGD, Adam

2. Do experiments with each combinations
   - For one combination, do *n*-th experiments
   - Sampling to get the distribution of combination

		Learning rate		
		0.01	0.005	0.001
Optimizer	SGD	Comb.1	Comb.2	Comb.3
	Adam	Comb.4	Comb.5	Comb.6

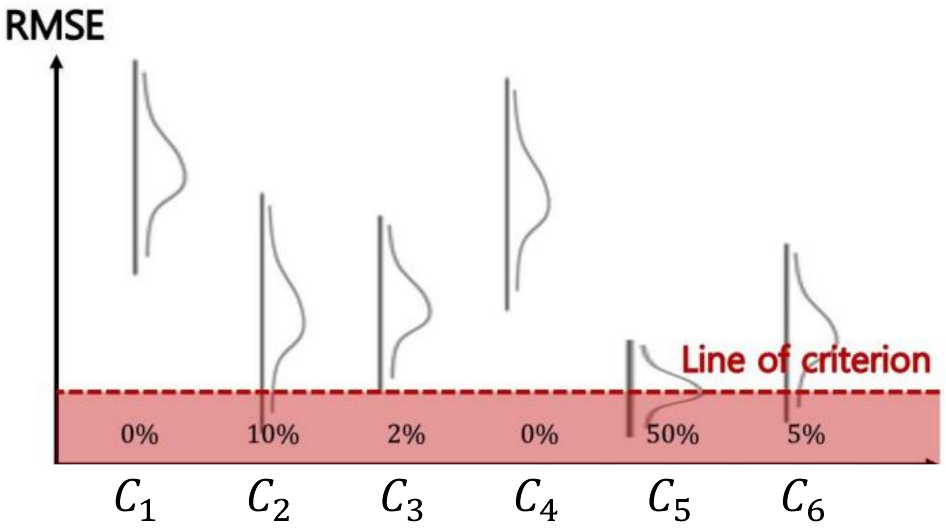
Hyper parameter Tuning Method based on Sampling for Optimal LSTM model(2019)

3. Estimate a distribution of the combination

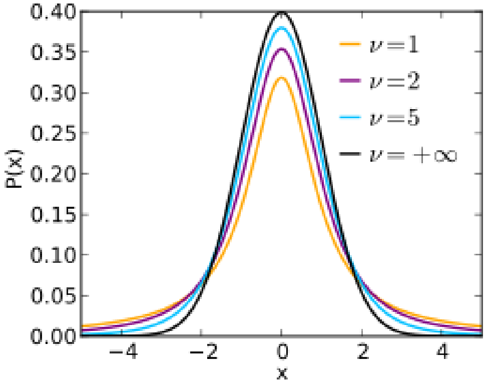
- Assume student *t*-distribution
- Use the RMSE(performance measure) as sample of dist.

4. Determine a criteria for selecting combinations

- Get set of means from the combination’s distributions
- Determine a criteria as min value of the mean set



Student t dist.



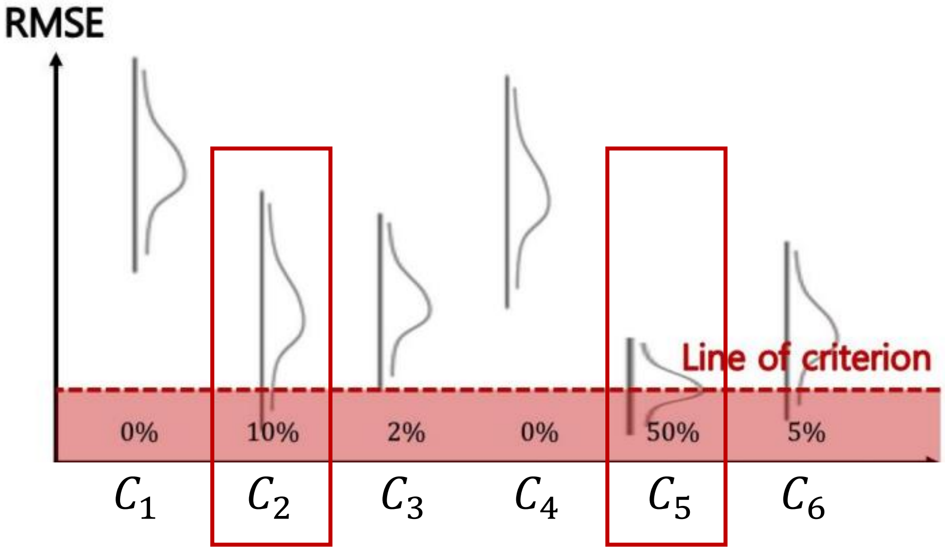
Ref.
Wikipedia(https://ko.wikipedia.org/wiki/%EC%8A%A4%ED%8A%9C%EB%8D%98%ED%8A%B8_t_%EB%B6%84%ED%8F%AC)

Prof. Hyerim Bae (hrbae@pusan.ac.kr)

Hyper parameter Tuning Method based on Sampling for Optimal LSTM model(2019)

5. Select the combinations for next step

- If the measure can exist under the criteria, use the combinations next step
- Measure can exist = the probability is over the τ (for example, 10%) = C_2, C_5



6. Consider one more hyperparameter

- Activation function of output layer
  : Relu, tanh, sigmoid

		Activation function		
		Relu	tanh	sigmoid
Combination	Comb.2	Comb2-1	Comb2-2	Comb2-3
	Comb.5	Comb2-4	Comb2-5	Comb2-6

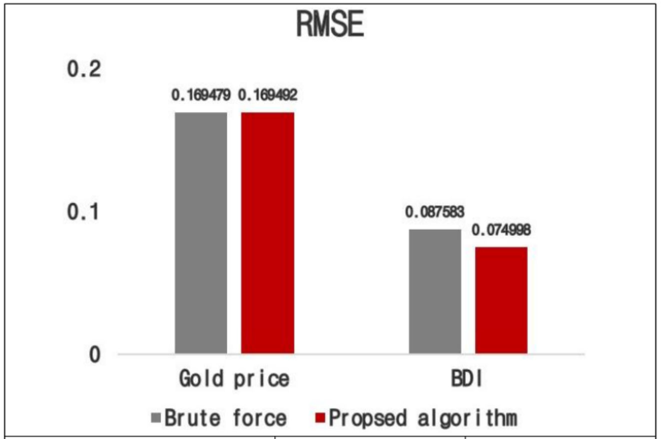
7. Repeat 2~6 until all hyperparameter is considered

Implementation

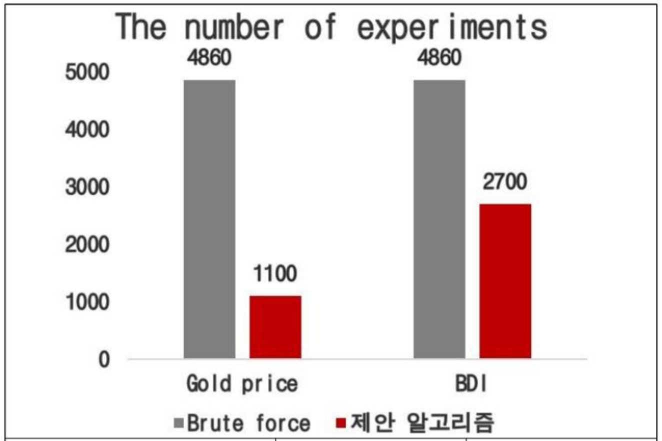
Demo 구현 영상

Hyper parameter Tuning Method based on Sampling for Optimal LSTM model(2019)

- Performance
  - RMSE of the proposed method is similar with Brute force
    - Success to find the best combination of hyperparameters
  - The number of experiments is quite smaller than Brute force
    - More efficiency than Brute force to find the best solution

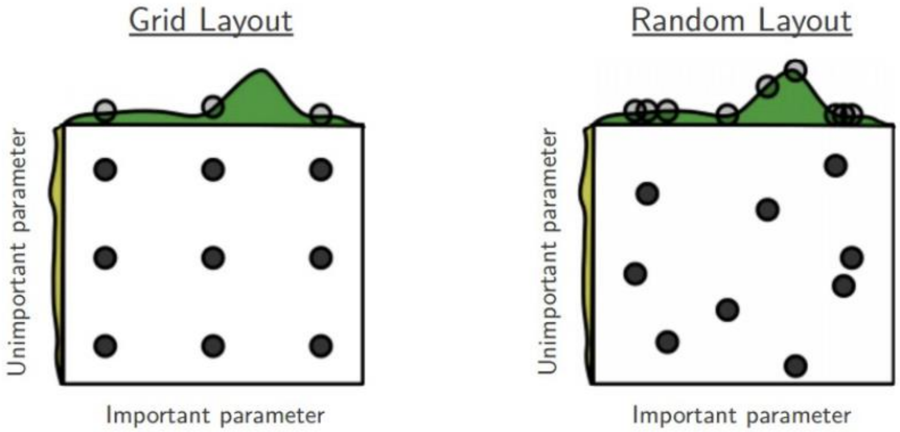


	Gold price	BDI
Brute force	0.169479	0.087583
Proposed algorithm	0.169492	0.074998



	Gold Price	BDI
Brute force	4860	4860
Proposed algorithm	1100	2700

- Random search



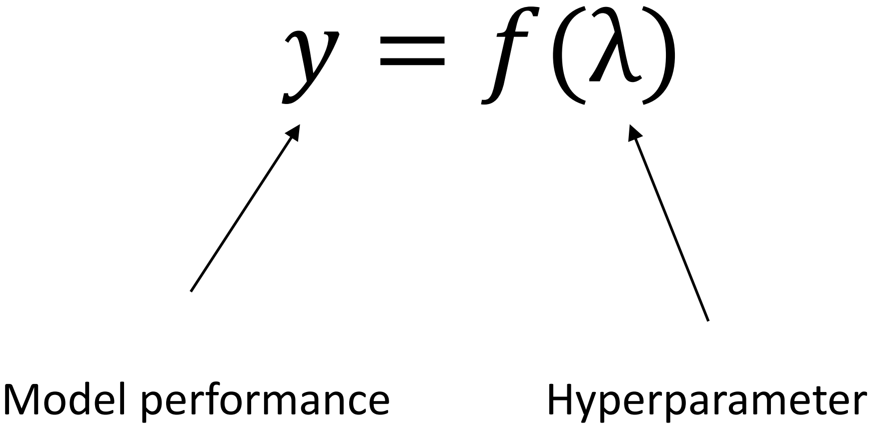
  - James Bergstra and Yoshua Bengio (2012)
    - Shows that RS shows better performance than GS

- In order to get top 5% performance
  - How many time do we need to search?

$$1-(1-0.05)^n = 0.95$$

Optimizing hyperparameters

- Finding hyperparameter which optimizes the performance



$$y = f(\lambda)$$

Model performance

Hyperparameter

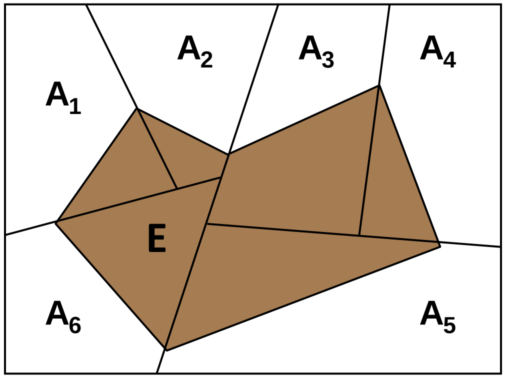
Black-box optimization

- Features
  - Objective function is unknown
  - Cannot use gradient (differentiation)
  - Costly

Bayesian Optimization for hyper-parameter tuning

- Estimate f(x) from data
  - Using Bayes theorem
  - By Gaussian Process

Bayes Rule



$$p(A|B) = \frac{p(A,B)}{p(B)} = \frac{p(B|A)p(A)}{p(B)}$$

$$p(A_i|E) = \frac{p(E|A_i)p(A_i)}{P(E)} = \frac{p(E|A_i)p(A_i)}{\sum_i p(E|A_i)p(A_i)}$$

- Based on the definition of conditional probability
  - $p(A_i|E)$ is posterior probability given evidence E
  - $p(A_i)$ is the prior probability
  - $P(E|A_i)$ is the likelihood of the evidence given A_i
  - p(E) is the preposterior probability of the evidence

Bayesian inference

- Let’s see the rule again

Likelihood

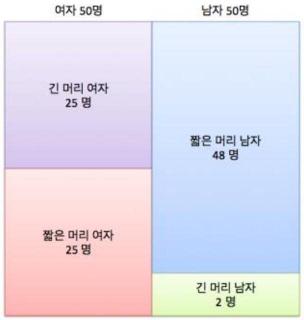
Prior

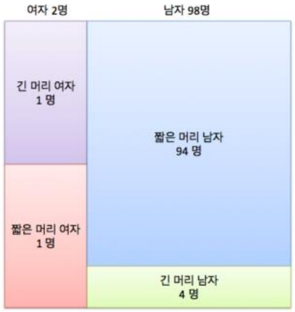
$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)}$$

Posterior

man or woman?

$$p(man|long\ hair)$$



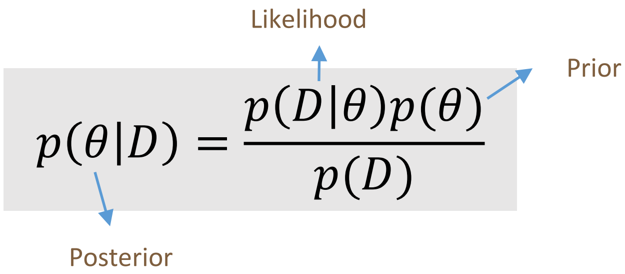


Source: https://brunch.co.kr/@chris-song/59

Prof. Hyerim Bae (hrbae@pusan.ac.kr)

Bayesian optimization

- "베이지안스럽게 최적화하기"



Likelihood

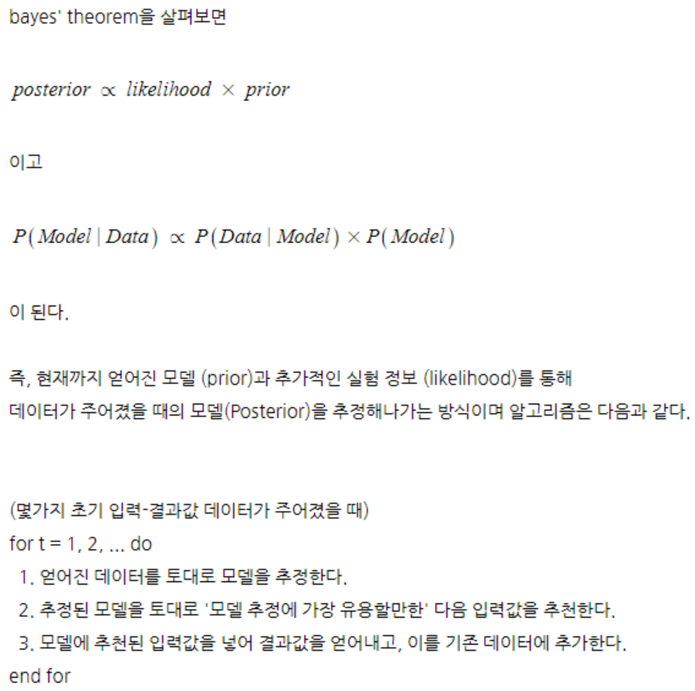
Prior

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)}$$

Posterior

- $P(Model|Data) \cong P(Data|Model) \times P(Model)$

```
for i=1,2,... do
   estimate parameters from data
   recommend next input
   generate data from model and add
end for
```



bayes' theorem을 살펴보면

$posterior \propto likelihood \times prior$

이고

$P(Model \mid Data) \propto P(Data \mid Model) \times P(Model)$

이 된다.

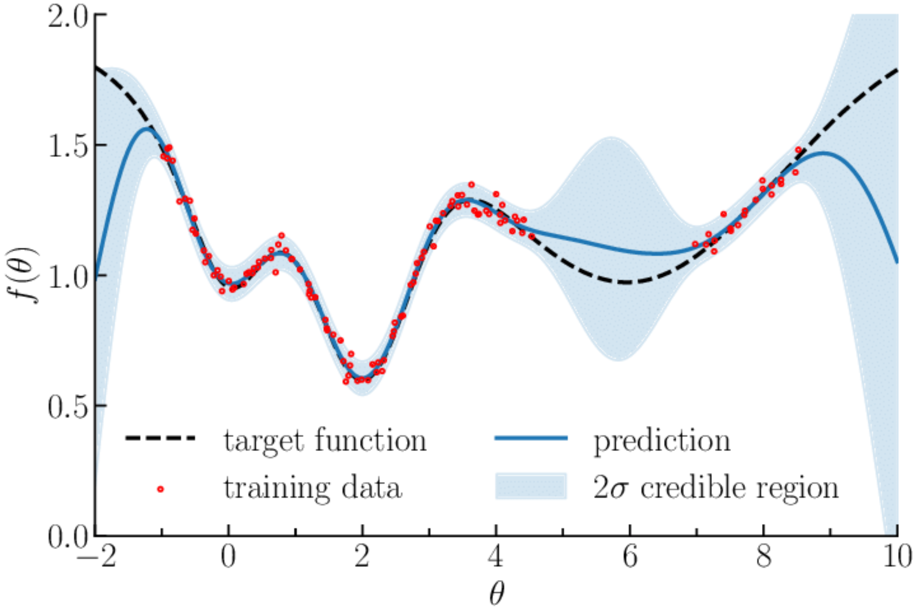
즉, 현재까지 얻어진 모델 (prior)과 추가적인 실험 정보 (likelihood)를 통해
데이터가 주어졌을 때의 모델(Posterior)을 추정해나가는 방식이며 알고리즘은 다음과 같다.

```
(몇가지 초기 입력-결과값 데이터가 주어졌을 때)
for t = 1, 2, ... do
 1. 얻어진 데이터를 토대로 모델을 추정한다.
 2. 추정된 모델을 토대로 '모델 추정에 가장 유용할만한' 다음 입력값을 추천한다.
 3. 모델에 추천된 입력값을 넣어 결과값을 얻어내고, 이를 기존 데이터에 추가한다.
end for
```

Gaussian Process

- Gaussian Distribution
  - Random variables
  - Mean, Variance(Standard deviation)

- Gaussian Process
  - Gaussian distribution for a function
  - Mean function: $\mu(x)$
  - Covariance function: $k(x, x')$



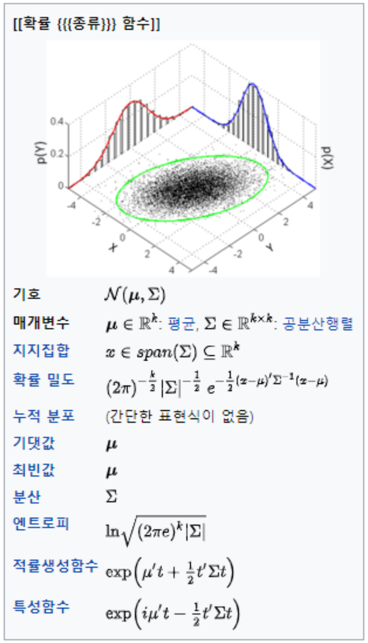
Sources: Florent Leclercq, “Bayesian optimization for likelihood-free cosmological inference”

Gaussian Process Regression

- In a regression function, *y*=*f*(*x*),
  - If x_1 and x_2 are similar, y_1 and y_2 are similar too.

$$y' = \sum_{i=1}^{N} w(x', x_i) y_i$$

- Weight *w* is represented as kernel, which can be learned
  - Single-variate: $Y \sim N(\mu, \sigma)$
  - Multi-variate: $\boldsymbol{Y} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$



[[확률 {{{종류}}} 함수]]

기호	$\mathcal{N}(\boldsymbol{\mu}, \Sigma)$
매개변수	$\boldsymbol{\mu} \in \mathbb{R}^k$: 평균, $\Sigma \in \mathbb{R}^{k \times k}$: 공분산행렬
지지집합	$\boldsymbol{x} \in span(\Sigma) \subseteq \mathbb{R}^k$
확률 밀도	$(2\pi)^{-\frac{k}{2}} \|\Sigma\|^{-\frac{1}{2}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})'\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})}$
누적 분포	(간단한 표현식이 없음)
기댓값	$\boldsymbol{\mu}$
최빈값	$\boldsymbol{\mu}$
분산	Σ
엔트로피	$\ln\sqrt{(2\pi e)^k \|\Sigma\|}$
적률생성함수	$\exp\left(\mu' t + \frac{1}{2} t' \Sigma t\right)$
특성함수	$\exp\left(i\mu' t - \frac{1}{2} t' \Sigma t\right)$

$$\boldsymbol{\Sigma} = \begin{bmatrix} K(X_1, X_1) & K(X_1, X_2) & \dots & K(X_1, X_N) \\ K(X_2, X_1) & K(X_2, X_2) & \dots & K(X_2, X_N) \\ \dots & \dots & \dots & \dots \\ K(X_N, X_1) & K(X_N, X_2) & \dots & K(X_N, X_N) \end{bmatrix}$$

$$\begin{pmatrix} \mathbf{Y} \\ Y' \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{\mu} \\ \mu' \end{pmatrix}, \begin{bmatrix} \boldsymbol{\Sigma} & K(\mathbf{X}, X') \\ K(X', \mathbf{X}) & K(X', X') \end{bmatrix} \right)$$

$$Y'|\mathbf{Y} \sim \mathcal{N}(\mu_{Y'|\mathbf{Y}}, \sigma_{Y'|\mathbf{Y}})$$

$$\mu_{Y'|\mathbf{Y}} = \mu' + K(X', \mathbf{X})\boldsymbol{\Sigma}^{-1}(\mathbf{Y} - \boldsymbol{\mu}),$$
$$\sigma_{Y'|\mathbf{Y}} = K(X', X') - K(X', \mathbf{X})\boldsymbol{\Sigma}^{-1}K(\mathbf{X}, X')$$

  - If $\mu = 0$

$$\mu(X') = K(X', \boldsymbol{X})\boldsymbol{\Sigma}^{-1}\boldsymbol{Y}$$

$$\sigma^2(X') = K(X', X') - K(X', \boldsymbol{X})\boldsymbol{\Sigma}^{-1}K(\boldsymbol{X}, X')$$

$$K(x_i,\ x_j) = \exp(-1/2\|x_i - x_j\|^2)$$

Training GP

- If we do not know the model (parameter)
  - The model is known to be quadratic,

$$f_x(x;\mu,\sigma^2) = \frac{1}{\sqrt[2]{(2\pi\sigma^2)}} e^{\frac{-(x-\mu)^2}{2\sigma^2}} \qquad --(1)$$

$$L(\mu,\sigma^2;x) = \prod_{i=1}^{n}\left[\frac{1}{\sqrt[2]{(2\pi\sigma^2)}} e^{\frac{-(x_i-\mu)^2}{2\sigma^2}}\right] \quad --(2)$$

$$\log L(\mu,\sigma^2;x) = \sum_{i=1}^{n}\left[-\frac{1}{2}log(2\pi) - \frac{1}{2}log(\sigma^2) - \frac{1}{2\sigma^2}(x_i-u)^2\right]$$

$$= -\frac{n}{2}log(2\pi) - \frac{n}{2}log(\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(x_i-\mu)^2 \qquad --(3)$$

$$f \sim GP(m,k)$$

$$m(x) = ax^2 + bx + c,\ \text{and} k(x,x') = \sigma_y^2 \exp(-\frac{(x-x')^2}{sl^2}) + \sigma_n^2\delta_{ii'}$$

여기서 파라미터 $\theta = \{a, b, c, \sigma_y, \sigma_n, l\}$ 입니다. 최적화는 Log-likelihood를 사용합니다.

$$L = \log p(\mathbf{y}|\mathbf{x},\theta) = -\frac{1}{2}\log|\Sigma| - \frac{1}{2}(\mathbf{y}-\mu)^T\Sigma^{-1}(\mathbf{y}-\mu) - \frac{n}{2}\log(2\pi)$$

이 식을 각 파라미터에 대해 편미분을 할 수 있습니다.

$$\frac{\partial L}{\partial \theta_m} = -(\mathbf{y}-\mu)^T\Sigma^{-1}\frac{\partial m}{\partial \theta_m}$$

$$\frac{\partial L}{\partial \theta_k} = \frac{1}{2}\mathrm{trace}(\Sigma^{-1}\frac{\partial \Sigma}{\partial \theta_k}) + \frac{1}{2}(\mathbf{y}-\mu)^T\frac{\partial \Sigma}{\partial \theta_k}\Sigma^{-1}\frac{\partial \Sigma}{\partial \theta_k}(\mathbf{y}-\mu)$$

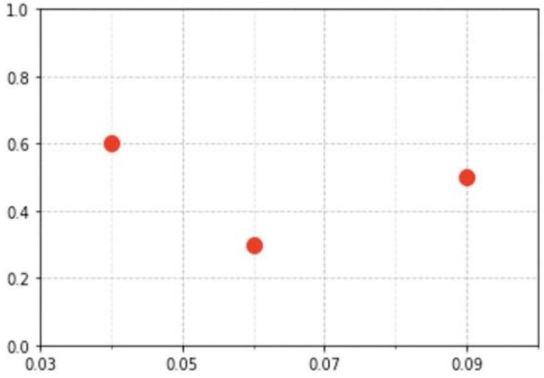
여기서 θ_m, θ_k는 각각 mean, covariance에 대한 파라미터를 의미합니다. 이 파라미터들을 conjugate gradient 방법을 사용하여 최적화하고, 이 때 위의 세 식이 활용될 것입니다.

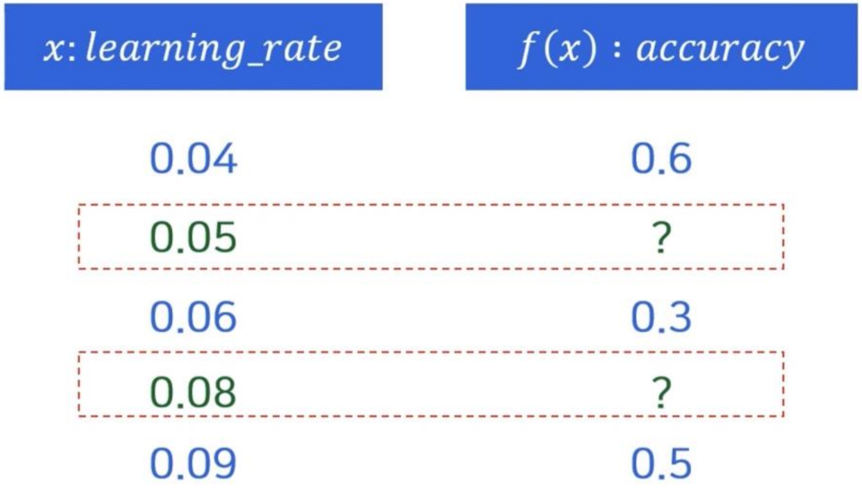
Exploitation vs. Exploration

- Exploitation
  - 지금 탐색중인 곳을 더 면밀히 탐색
  - Risk of Local optima
- Exploration
  - 더 넓은 탐색

Estimation by Bayesian Optimization

- Estimation of *f*(*x*) from data observed





x: *learning_rate*	$f(x)$: *accuracy*
0.04	0.6
0.05	?
0.06	0.3
0.08	?
0.09	0.5

https://www.youtube.com/watch?v=PTxqPfG_lXY&t=284s

Prof. Hyerim Bae (hrbae@pusan.ac.kr)

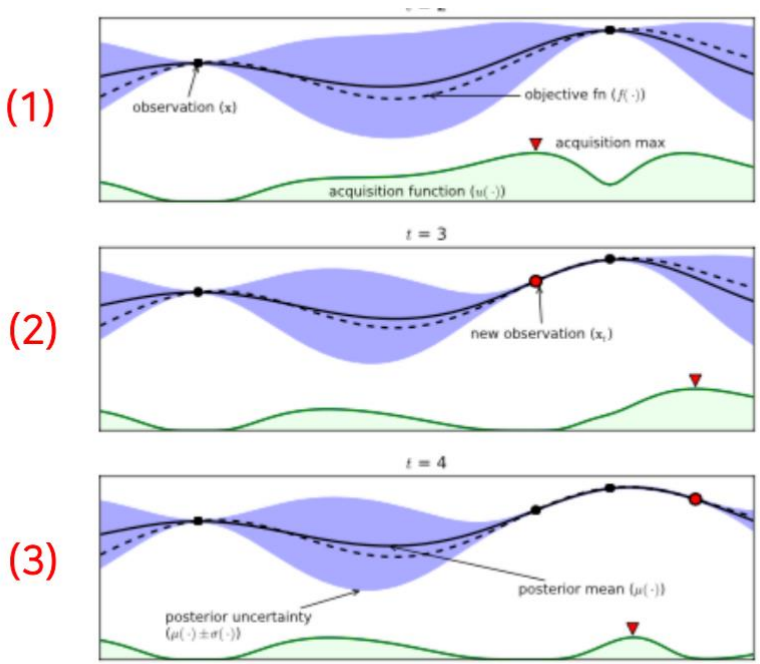
Acquisition function

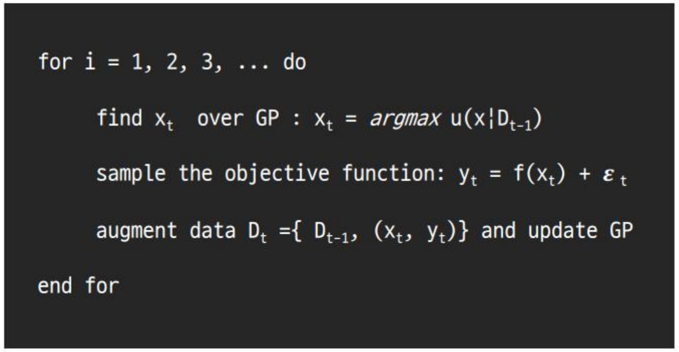
- How to get the next data?

- Exploitation
  - High mean

- Exploration
  - High variance

Acquisition function



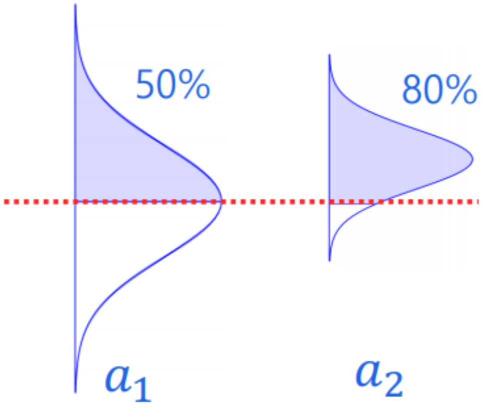


```
for i = 1, 2, 3, ... do
    find $x_t$ over GP : $x_t$ = $argmax\ u(x|D_{t-1})$
    sample the objective function: $y_t = f(x_t) + \varepsilon_t$
    augment data $D_t$ ={ $D_{t-1}$, ($x_t$, $y_t$)} and update GP
end for
```

Exploitation

- Probability of Improvement

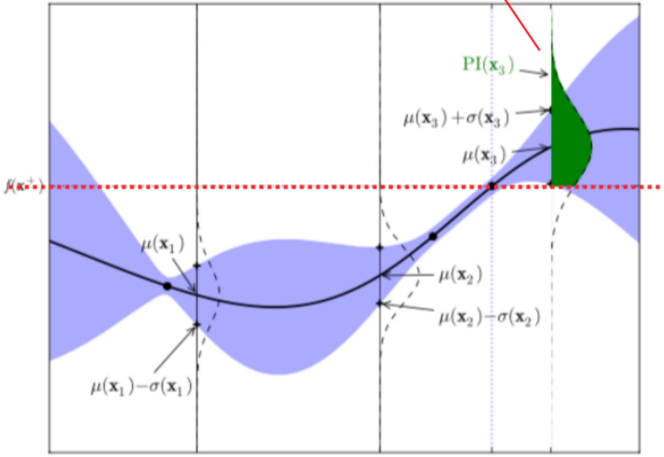
$\mathrm{PI}(a_1) < \mathrm{PI}(a_2)$



관측한 최댓값

확률이 최대

관측한 최댓값

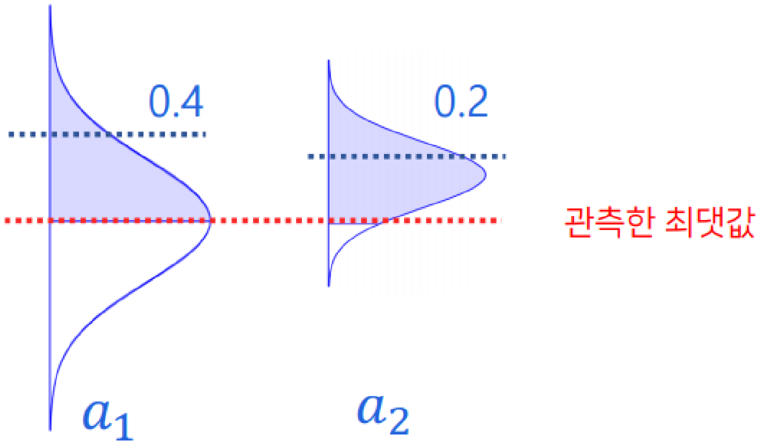

$f(\mathbf{x}^+)$
$\mathrm{PI}(\mathbf{x}_3)$
$\mu(\mathbf{x}_3)+\sigma(\mathbf{x}_3)$
$\mu(\mathbf{x}_3)$
$\mu(\mathbf{x}_1)$
$\mu(\mathbf{x}_2)$
$\mu(\mathbf{x}_2)-\sigma(\mathbf{x}_2)$
$\mu(\mathbf{x}_1)-\sigma(\mathbf{x}_1)$

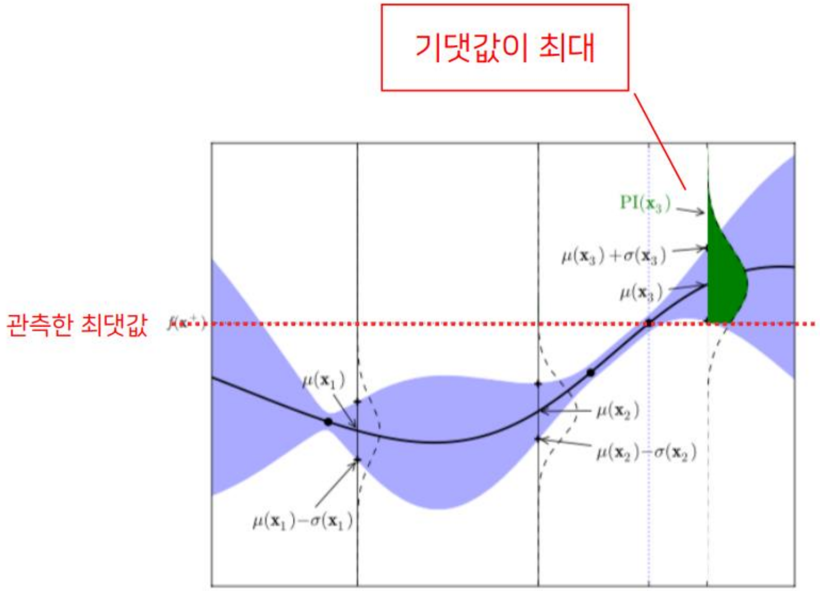
*) Kushner 1964
**) https://arxiv.org/pdf/1012.2599.pdf

https://www.youtube.com/watch?v=PTxqPfG_lXY&t=284s

- Expected Improvement

$$\mathrm{EI}(a_1) > \mathrm{EI}(a_2)$$





*) Mockus et al, 1978

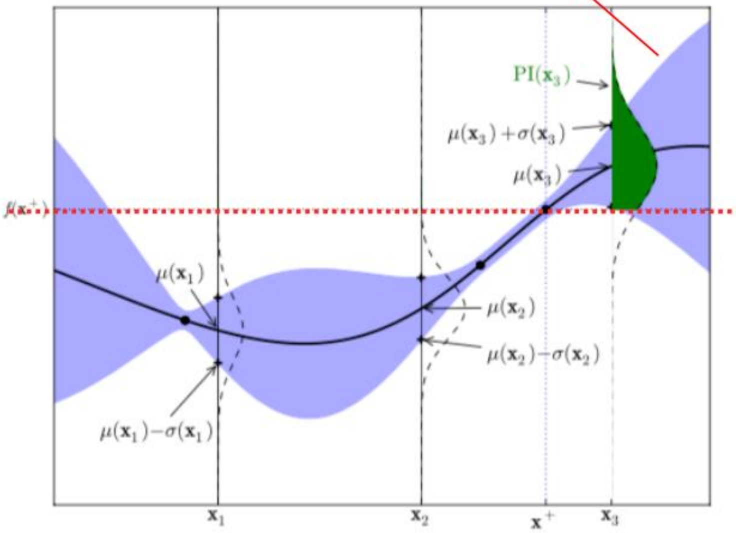
https://www.youtube.com/watch?v=PTxqPfG_lXY&t=284s

- Upper Confidence Bound

$$argmax(\mu(x) + k \cdot \sigma(x))$$

upper bound가 최대

관측한 최댓값



*) Srinivas et al, 2010, https://arxiv.org/pdf/0912.3995.pdf

https://www.youtube.com/watch?v=PTxqPfG_lXY&t=284s

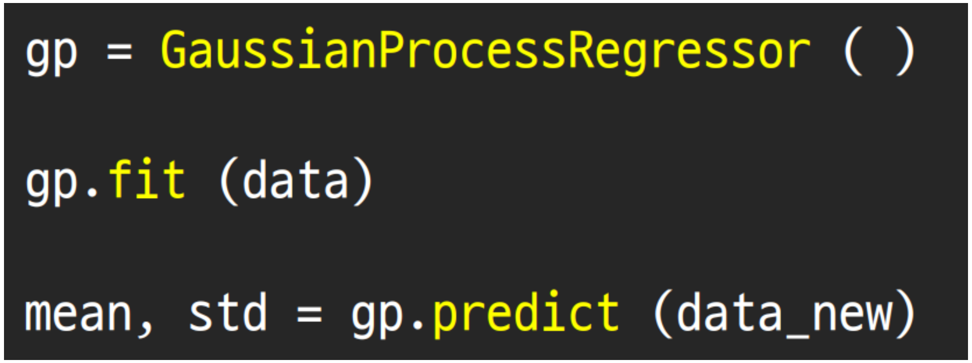
By GPR

- Estimation of mean and variance for a new *x*, x^*

$$\mu(x^*) = k^T K^{-1} f_{1:t}$$

$$\sigma^2(x^*) = k(x^*, x^*) - k^T K^{-1} k$$

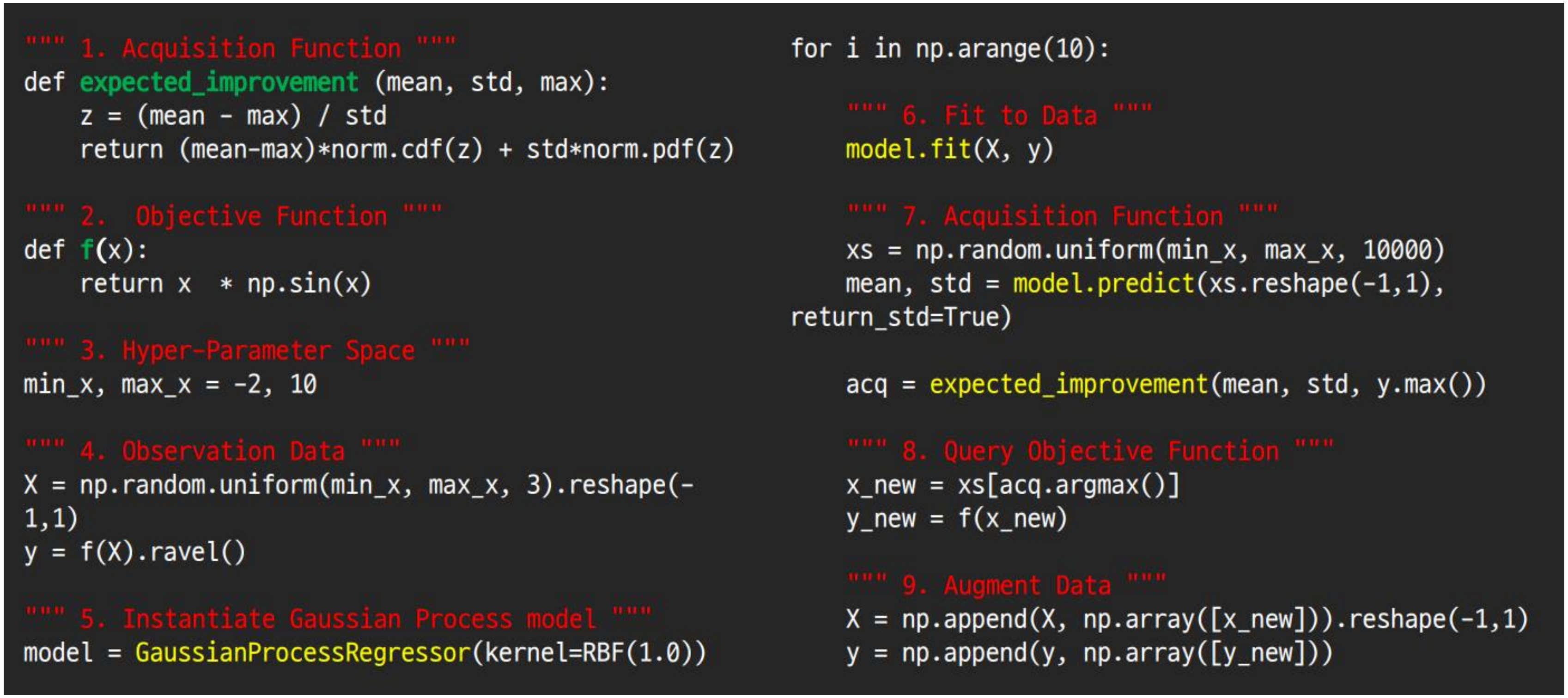
$$k(x_i,\ x_j) = \exp(-1/2\|x_i - x_j\|^2)$$



```
gp = GaussianProcessRegressor ( )

gp.fit (data)

mean, std = gp.predict (data_new)
```

https://www.youtube.com/watch?v=PTxqPfG_lXY&t=284s

Prof. Hyerim Bae (hrbae@pusan.ac.kr)



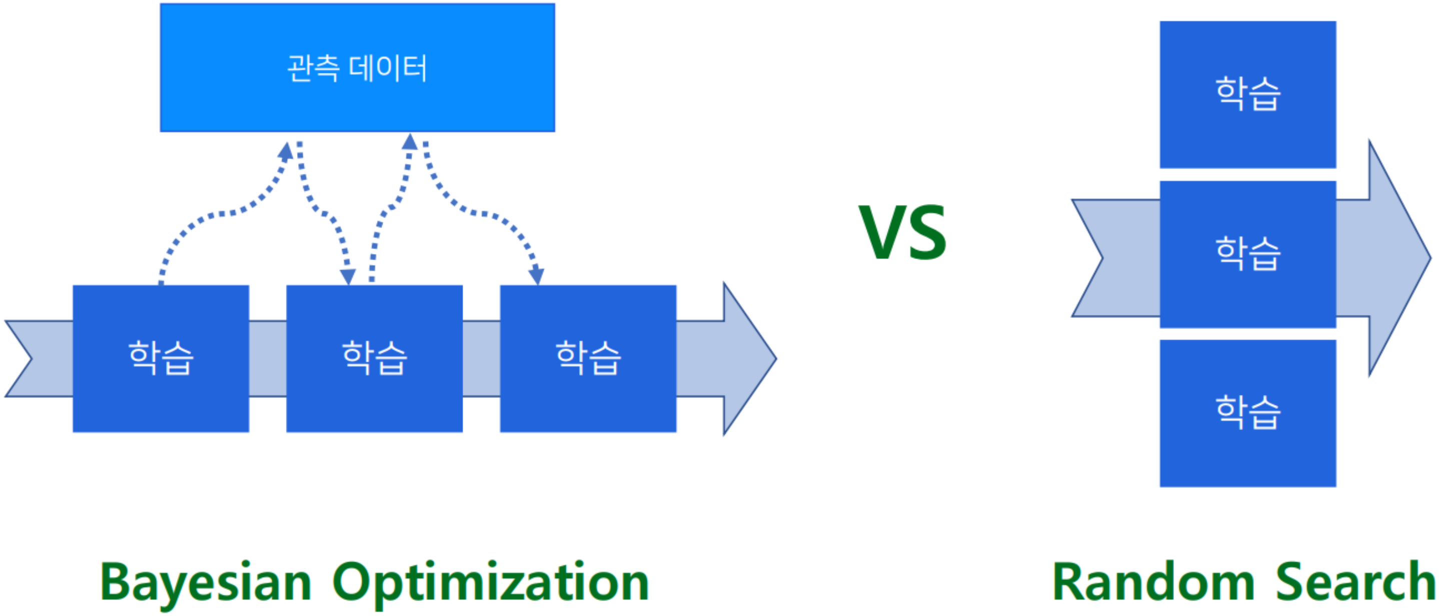
```
""" 1. Acquisition Function """
def expected_improvement (mean, std, max):
    z = (mean - max) / std
    return (mean-max)*norm.cdf(z) + std*norm.pdf(z)

""" 2.  Objective Function """
def f(x):
    return x  * np.sin(x)

""" 3. Hyper-Parameter Space """
min_x, max_x = -2, 10

""" 4. Observation Data """
X = np.random.uniform(min_x, max_x, 3).reshape(-
1,1)
y = f(X).ravel()

""" 5. Instantiate Gaussian Process model """
model = GaussianProcessRegressor(kernel=RBF(1.0))
```

```
for i in np.arange(10):

    """ 6. Fit to Data """
    model.fit(X, y)

    """ 7. Acquisition Function """
    xs = np.random.uniform(min_x, max_x, 10000)
    mean, std = model.predict(xs.reshape(-1,1),
return_std=True)

    acq = expected_improvement(mean, std, y.max())

    """ 8. Query Objective Function """
    x_new = xs[acq.argmax()]
    y_new = f(x_new)

    """ 9. Augment Data """
    X = np.append(X, np.array([x_new])).reshape(-1,1)
    y = np.append(y, np.array([y_new]))
```

https://www.youtube.com/watch?v=PTxqPfG_lXY&t=284s

Bayesian Optimization

- Drawback
  - Time Consuming

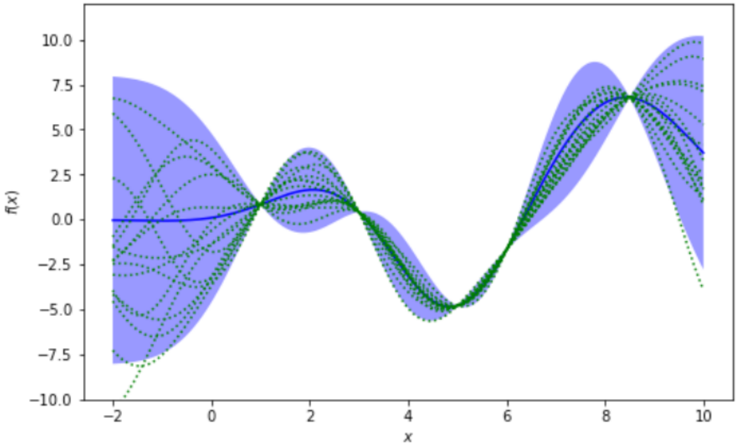


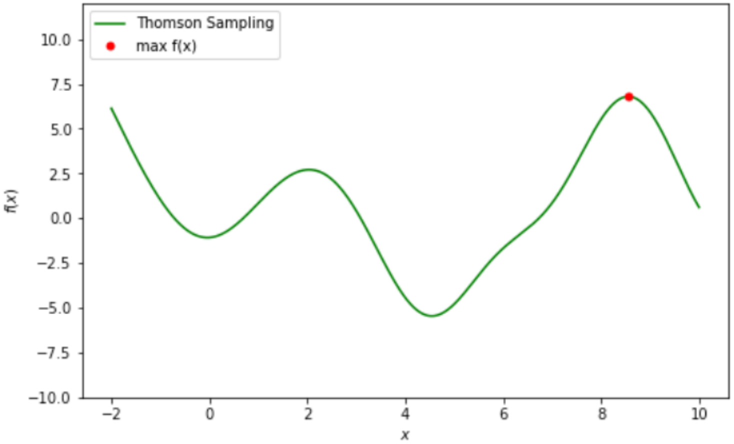
https://www.youtube.com/watch?v=PTxqPfG_lXY&t=284s

Parallel BO

Acquision Function : Thompson Sampling

- 사후 분포를 기반으로 샘플링



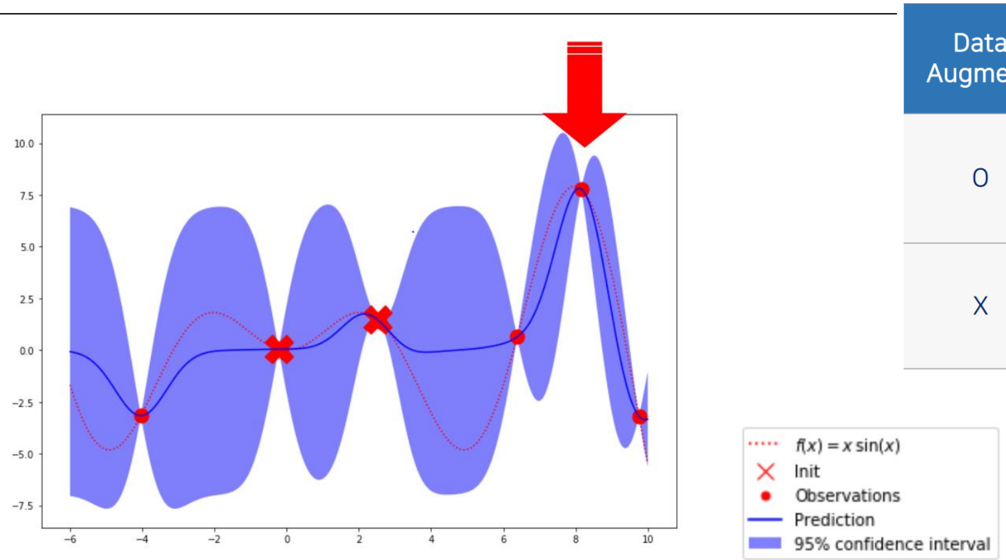


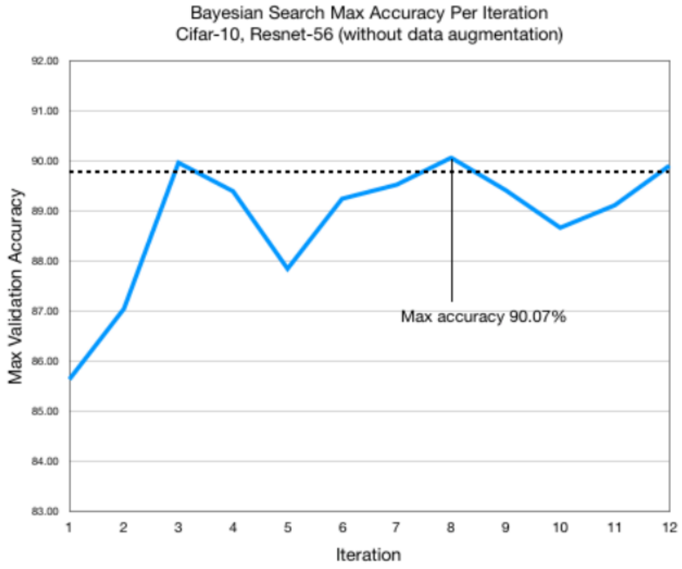
https://www.youtube.com/watch?v=PTxqPfG_lXY&t=284s

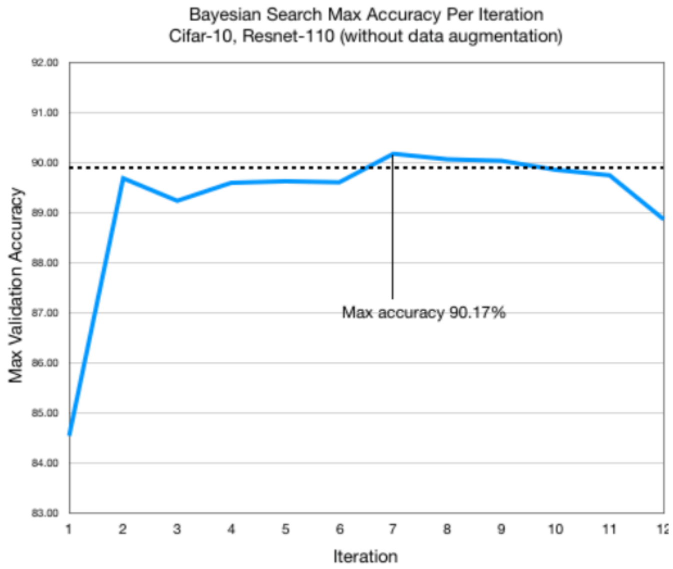
Performance of BO

Accuracy on Cifar-10 with Resnet

Data Augment	Model	Random	BayesOpt	Baseline (paper)
O	Resnet-110	94.37	94.53	93.57
	Resnet-56	94.37	94.28	93.03
X	Resnet-110	90.09	90.17	-
	Resnet-56	90.01	90.07	-







https://www.youtube.com/watch?v=PTxqPfG_lXY&t=284s

Reference

- https://www.youtube.com/watch?v=PTxqPfG_lXY&t=284s
- https://aistory4u.tistory.com/entry/%EA%B0%80%EC%9A%B0%EC%8B%9C%EC%95%88-%ED%94%84%EB%A1%9C%EC%84%B8%EC%8A%A4-%ED%9A%8C%EA%B7%80