S9372

Automatic Model Tuning Using Amazon SageMaker

Cyrus M Vahid
Principal Evangelist – AWS AI Labs
CyrusMV@amazon.com

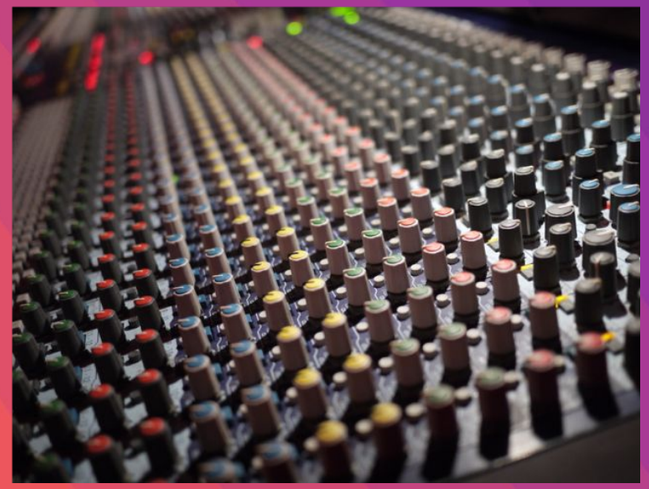
Agenda

Hyperparameters

Search Based HPO

Bayesian HPO

Amazon SageMaker AMT



Hyperparameters

What is a Hyperparameter

- **Hyperparameter** = algorithm parameter
- It affects how an algorithm behaves during model learning process
- “Any decision an algorithm author can’t make for you”

Examples of Hyperparameters

Model:

- Number of layers: 1, 2, 3, …
- Activation functions: Sigmoid, tanh, RELU, …

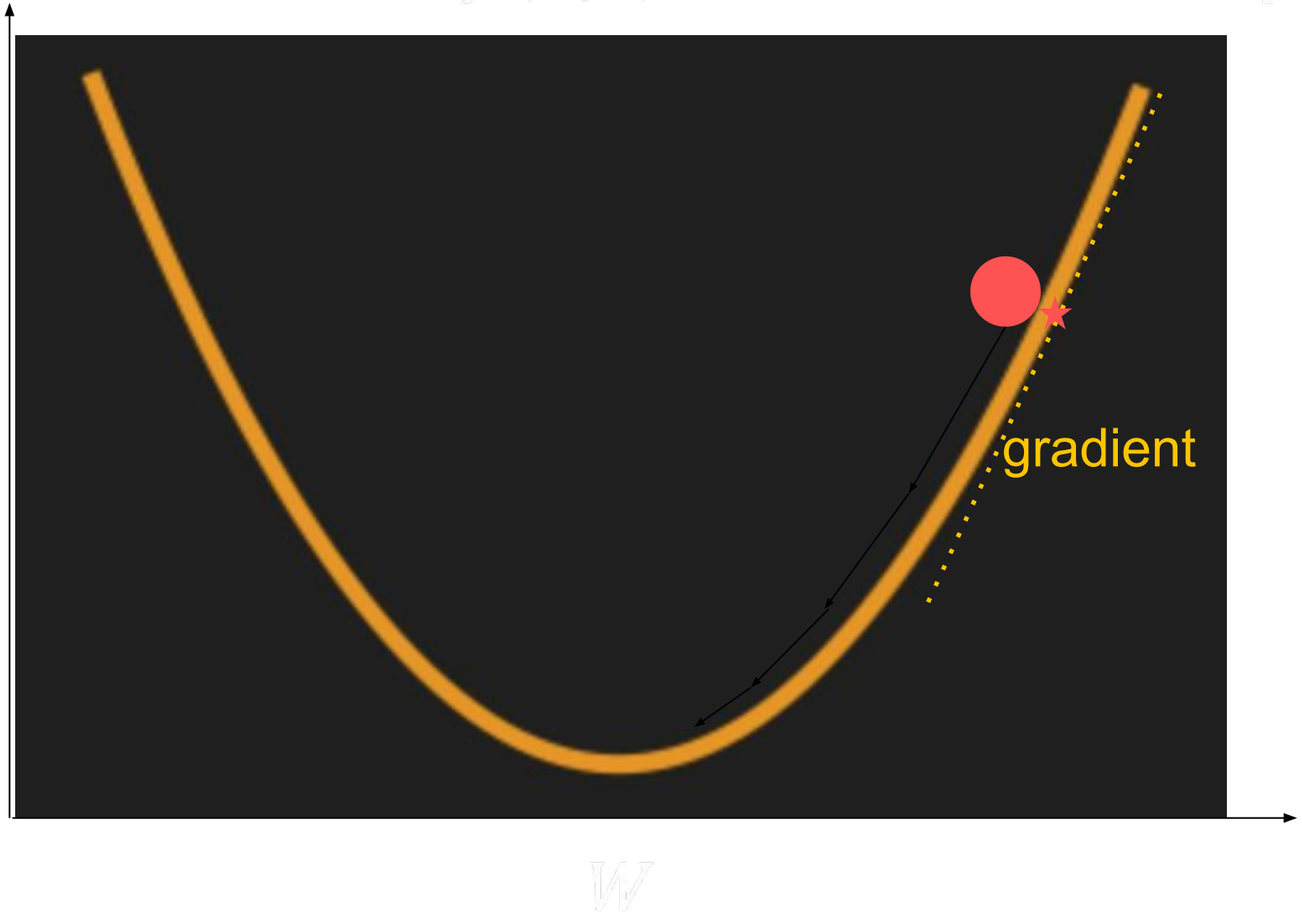
Optimization:

- Method: SGD, Adam, AdaGrad, …
- Learning Rate: 0.01 to 2

Data:

- Batch Size: 8, 16, 32 …
- Augmentation: Resize, Normalize, Color Jitter, …

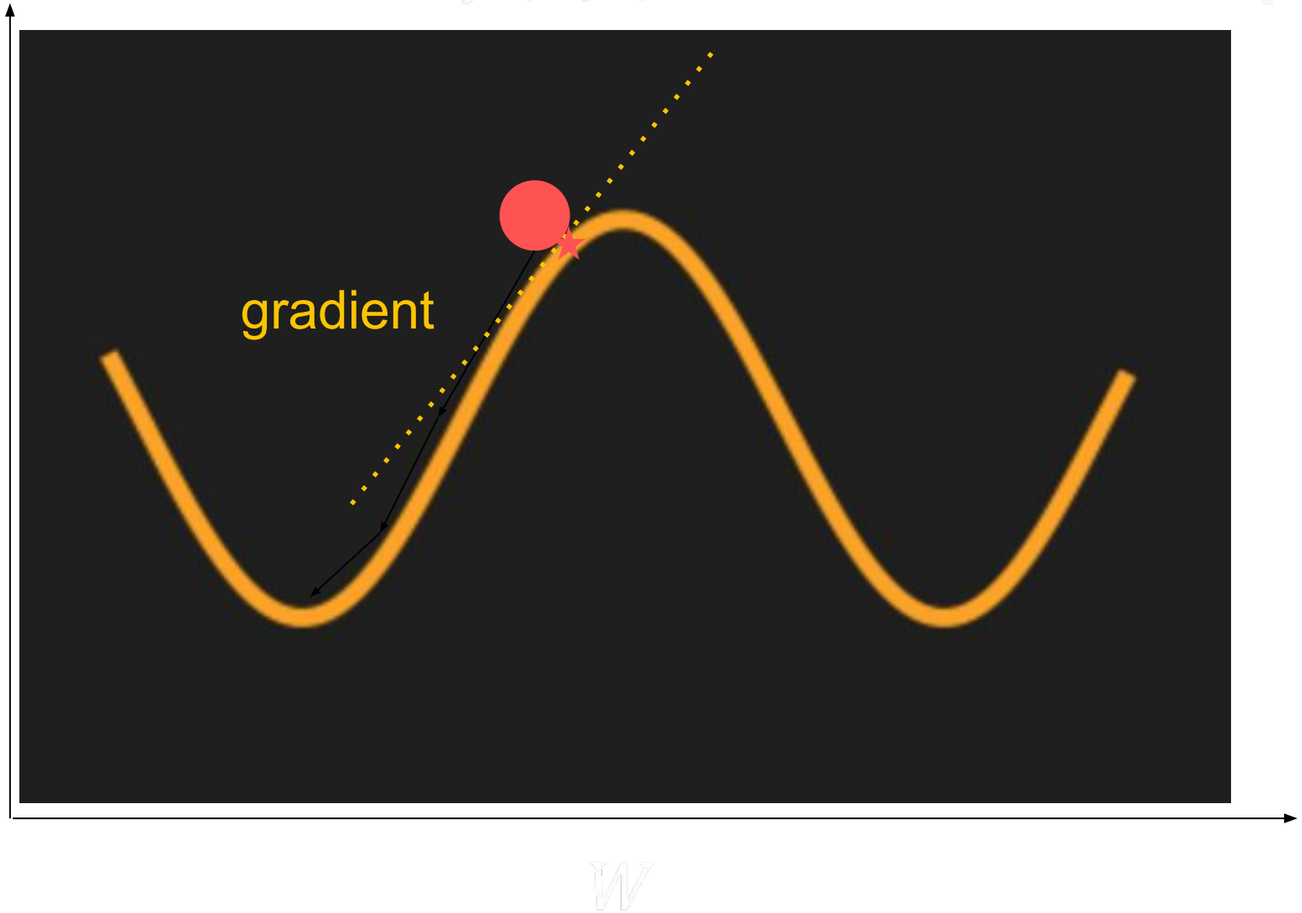
Optimizing Loss Function for a Model



Hyper Parameters

- `lr= .001`
- `momentum=.9`
- `dd=0.1`
- `init=Xavier(mag=3.0)`
- `'sgd'`
- `num_layers = 3`
- `dropout = .2, .5, .1`
- `batch_size = 64`
- ...

Optimizing Loss Function for a Model



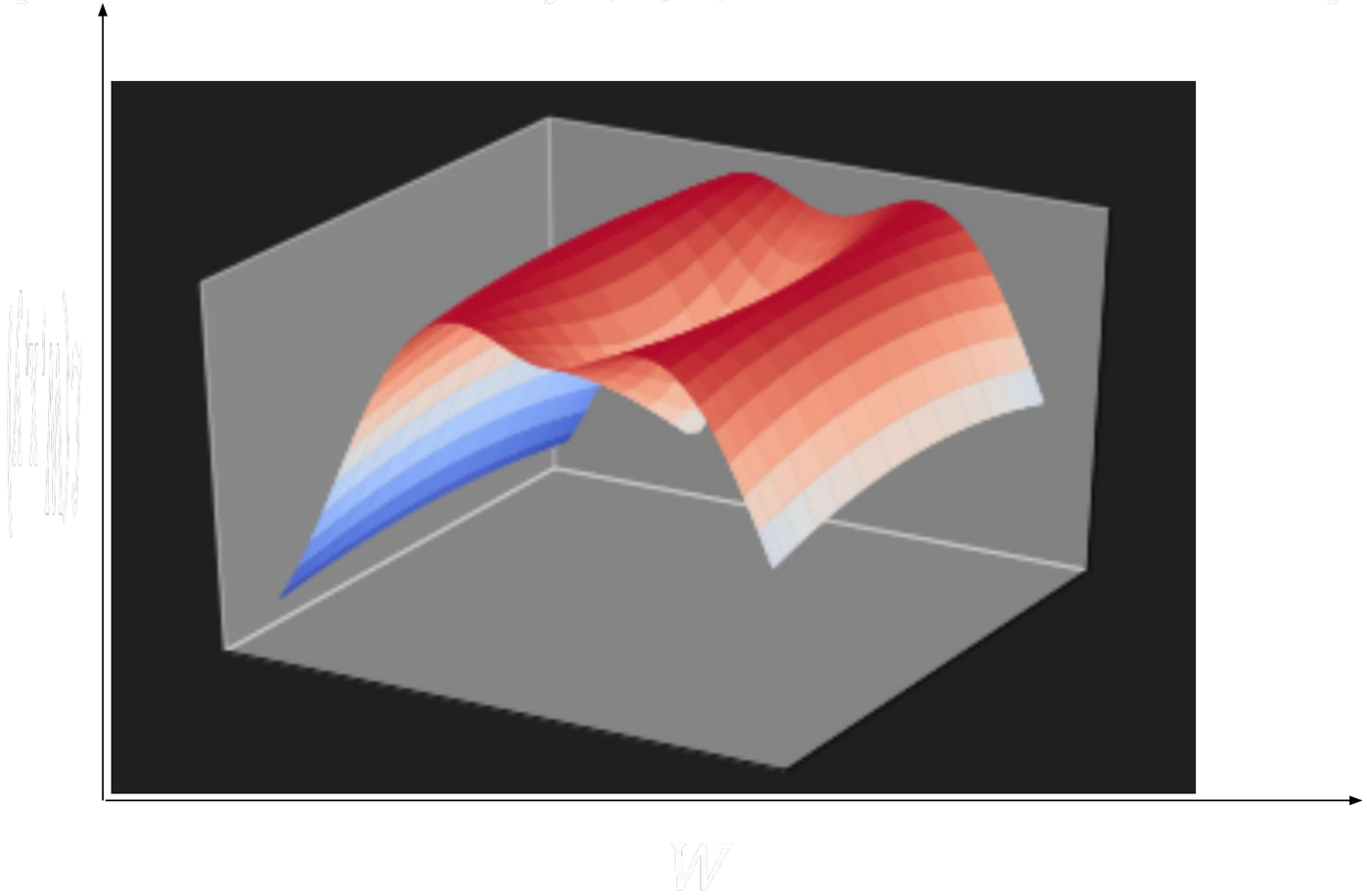
Hyper Parameters

- `lr= .001`
- `dd=0.1`
- `init=uniform()`
- `'adam'`
- `num_layers = 2`
- `dropout = .2, .5`
- `batch_size = 128`
- …

Optimizing Loss Function for a Model

- $Model: f(x) = W^T X + b$
- $Optimization: \min f(x|\theta), where\ \theta = model\ params$

$$\mathcal{L} = y - \hat{y}$$

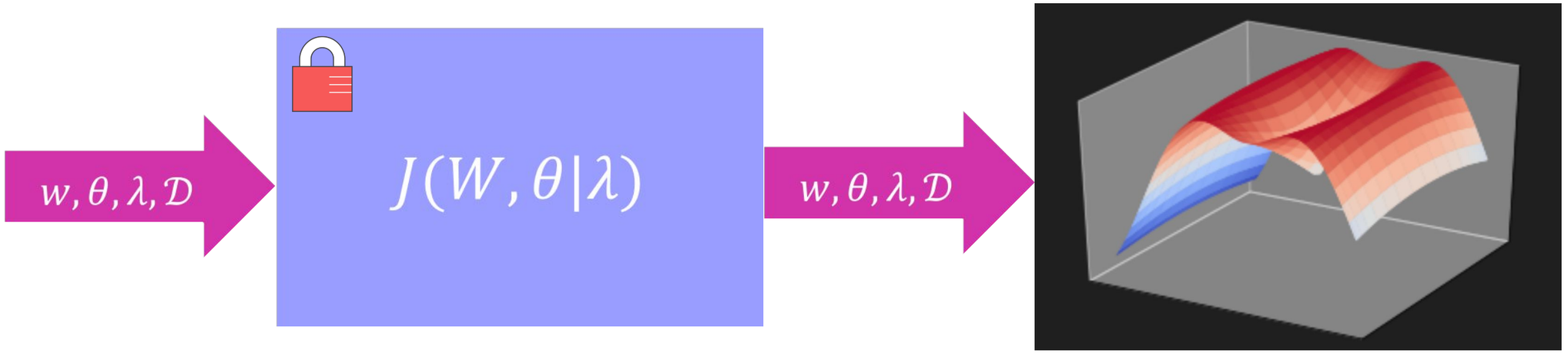


Hyper Parameters

- `lr= .001`
- `momentum=.9`
- `dd=0.1`
- `init=orthogonal(scale=31.5)`
- `'lbsgd'`
- `num_layers = 15`
- `dropout = .2, .5, 0, .1, …, .2`
- `batch_size = 2000`
- …

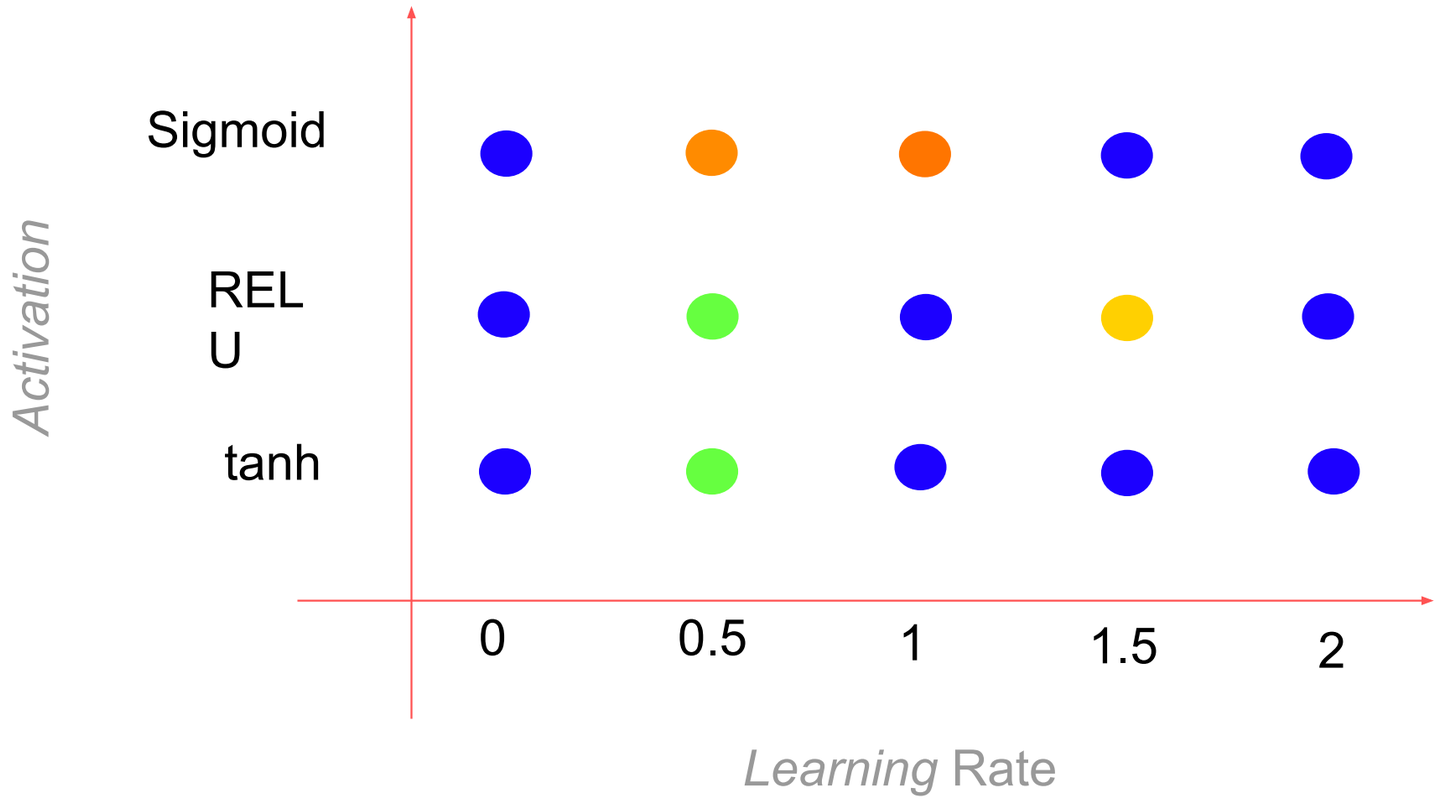
Blackbox Optimization

- We aim to minimize the objective function J.
- We have no knowledge of what the objective function is.
- We do have access to the gradients of the objective function



Model-Free HPO

Grid Search



Curse of Dimensionality

- In grid search the user specifies a finite set of values for each hyperparameter and then .
- Each hyperparameter increases degree of freedom and results in combinatorial explosion.

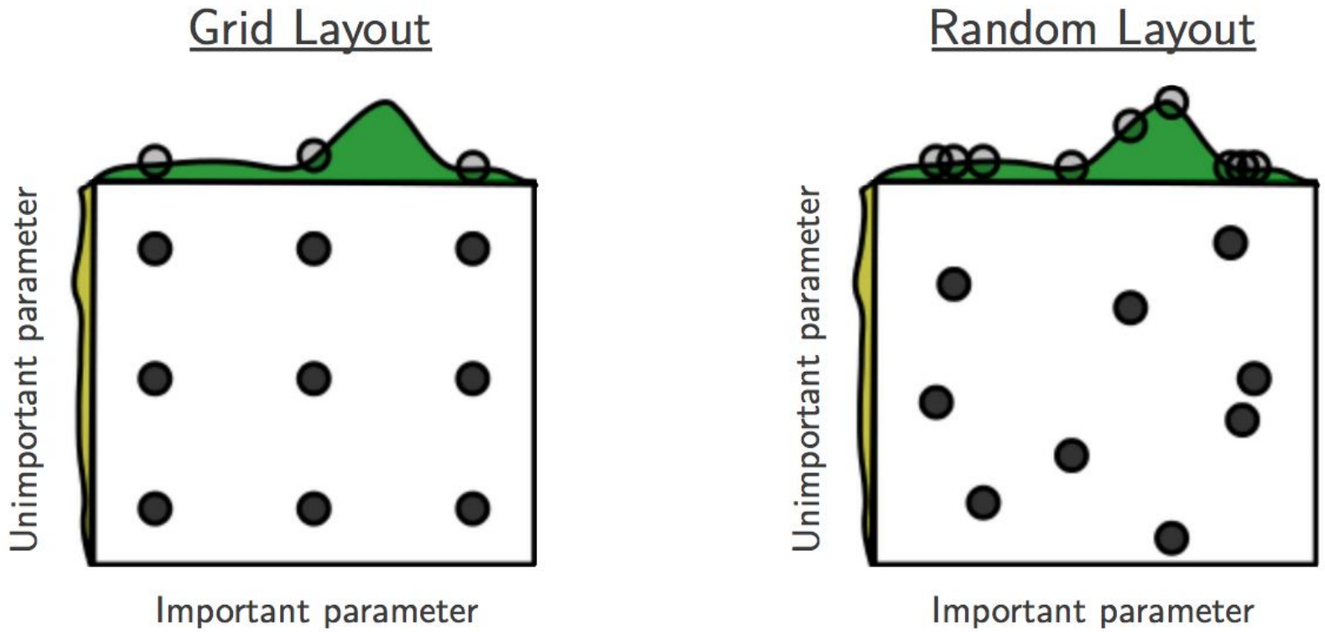
Let $P = \{p_1, p_2, \ldots, p_n\}$ *be the set of paramerters.*
If each $p_i \in P$ have $c_{p_i} \in \mathbb{N}$ valid combinations, then it follows from the multiplication principle that:
total number of combinations of

$$P = C_P = \prod_{i=1}^{n} c_{p_i} = C_{p_1} \times C_{p_2} \times \cdots \times C_{p_n}$$

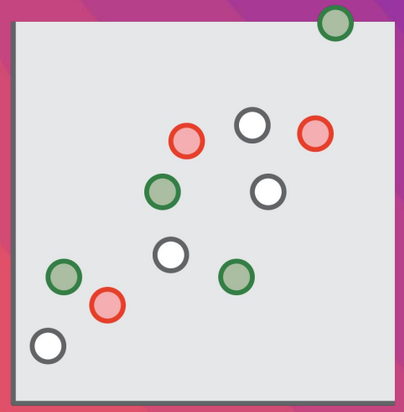
- Example:
  - 1 HP and 5 options → 5 combinations
  - 2 HP and 5 options each → 5*5 combinations
  - 10 HO and 5 options each → 5*5*5*5*5*5*5*5*5*5 = $5^{10} \approx 10^6$ combinations

Random Search

- Random search samples configurations at random until a certain budget B for the search is exhausted.
- The number of evaluations for N hyperparameters is only $B^{1/N}$ vs. B for Grid Search.



Bergstra, Bengio. 2012. “Random Search for Hyper-Parameter Optimization”



Model-Based HPO for Automatic Model Training

Model Optimization

- The aim of model optimization to to find a set of model parameters, θ^*, that returns the most accurate results for a given dataset.

Optimize model-params: θ

$$\theta^* = arg \min_{\theta \epsilon \Theta} f(\lambda, \theta, \mathcal{D}_t \, \mathcal{D}_v)$$

Hyperparameter Optimization

- The aim of HPO is to find a set of hyperparameters, λ^*, that given algorithm $\mathcal{A}$, and dataset $\mathcal{D}$, returns the most accurate results.

Optimize model-params: θ

$$\theta^* = arg \min_{\theta \epsilon \Theta} f(\lambda, \theta, \mathcal{D}_t \, \mathcal{D}_v)$$

Optimize model-params: λ^*

$$\lambda^* = arg \min_{\lambda \epsilon \Lambda} \mathbb{E}_{(\mathcal{D}_t, \mathcal{D}_v) \sim \mathcal{D}} V(\mathcal{L}, \mathcal{A}_\lambda, \mathcal{D}_t, \mathcal{D}_v)$$

Where V measures the loss of a model generated by algorithm $\mathcal{A}$ with hyperparameters λ on dataset $\mathcal{D}$

Bayesian Optimization

- Keeps track of previous evaluations and infers expected behaviour.
- It is Bayesian in a sense that the surrogate model model uses prior probability distribution to make predictions about the posterior.

$$P(Y|X) \propto P(Y|X)P(Y)$$

- An acquisition function selects next points to be evaluated. Commonly expected improvement.

$$\mathbb{E}[\mathrm{I}(\lambda)] = \mathbb{E}[\max(f\_\min - Y, 0)]$$

- As per iteration improves our beliefs about the objective function by applying iterative learning.

Bayesian Optimization

- 1: $for\ n = 1, 2, 3, \ldots\ do$
- 2: $\quad select\ new\ X_{n+1}\ by\ optimizing\ acquisition\ function\ \alpha$
  $\quad X_{n+1} = \arg\max_x \alpha(x; \mathcal{D})$
- 3: $\quad query\ objective\ function\ to\ obtain\ y_{n+1}$
- 4: $\quad augment\ data\ \mathcal{D}_{n+1} = \{\mathcal{D}_n, (x_{n+1}, y_{n+1})\}$
- 5: $\quad update\ the\ surrogate\ model$

https://ieeexplore.ieee.org/document/7352306

Gaussian Process

- Gaussian distribution is a series of random numbers that is described by mean μ and variance σ^2.

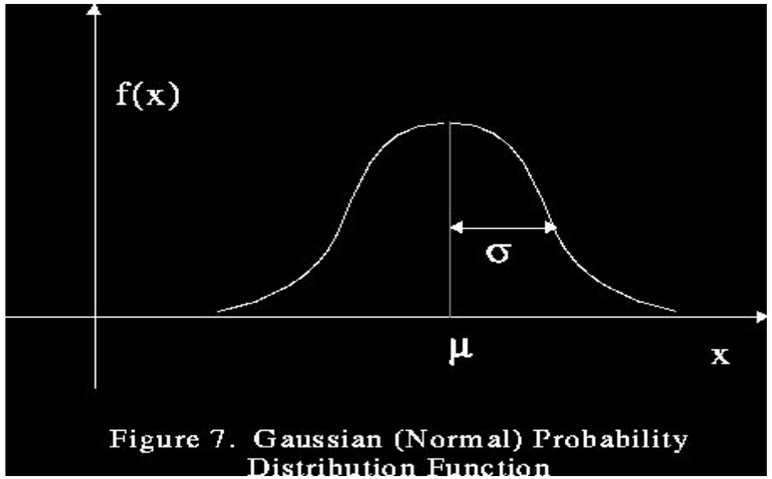
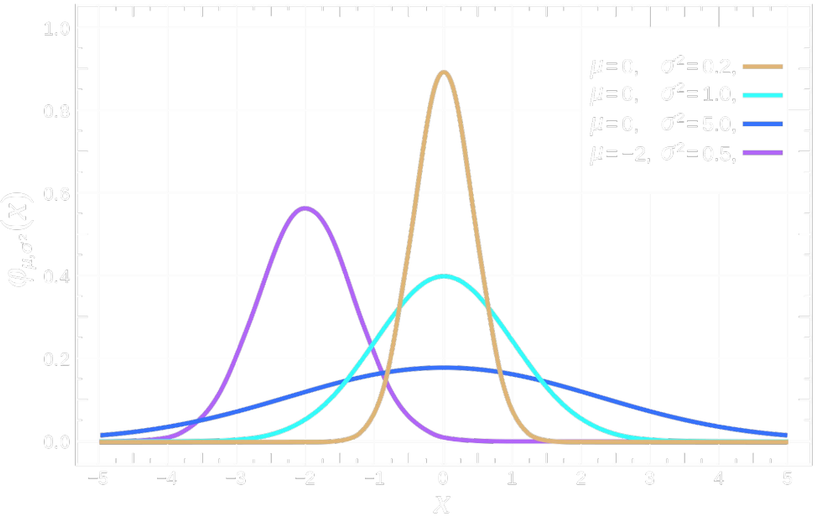


Figure 7. Gaussian (Normal) Probability Distribution Function

- Gaussian Process is a distribution over functions each of which returns mean and variance of a Gaussian distribution.

$$f : \mathcal{X} \rightarrow \mathbb{R}$$

$$f(X_{t_1}), f(X_{t_2}), \ldots, f(X_{t_n}) \sim \mathcal{N}(\mu, \Sigma)$$



Gaussian Process – *continued…*

- Each distribution corresponds to a set of hyperparameters Λ;

$$\lambda_i \epsilon \Lambda = \prod_{i=1}^{n} \Lambda_i$$

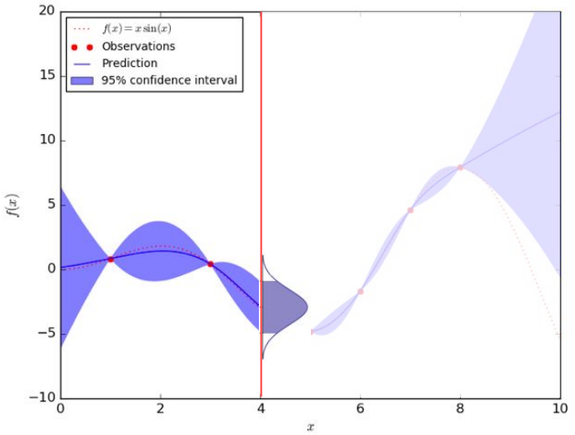
- A Gaussian process is fully specified by a mean $\mu(\lambda)$ and a covariance function* $k(\lambda, \lambda')$.

$$\mathcal{G}(\mu(\lambda), k(\lambda, \lambda'))$$

* Co-variance functions encode all assumptions about the form of function that we are modelling. In general, covariance represents some form of distance or similarity.

Gaussian Process for Model of Model Loss

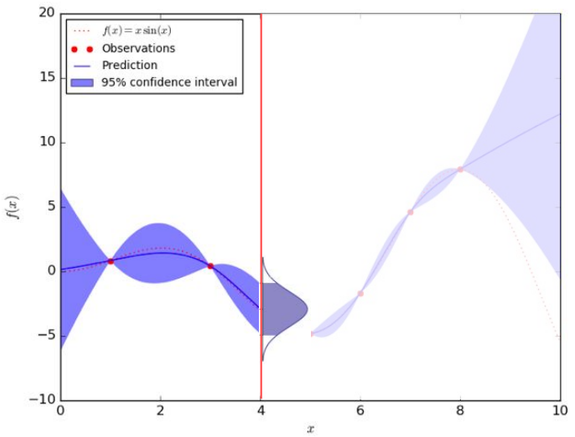
- A Gaussian process $\mathcal{G}(\mu(\lambda), k(\lambda, \lambda'))$ is fully specified by a mean $\mu(\lambda)$ and a covariance function $k(\lambda, \lambda')$

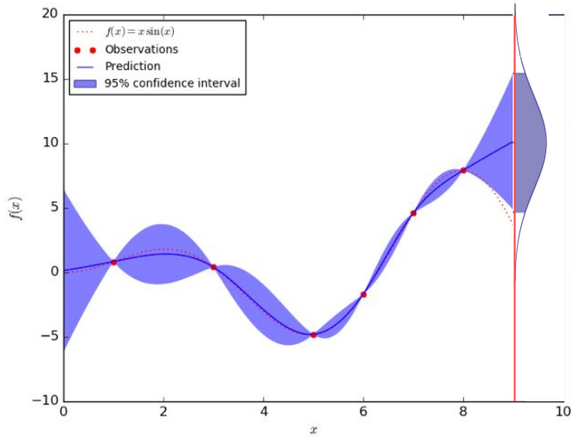


$$f(X_{t_1}), f(X_{t_2}), \ldots, f(X_{t_n}) \sim \mathcal{N}(\mu, \Sigma)$$

Gaussian Process for Model of Model Loss

- A Gaussian process $\mathcal{G}(\mu(\lambda), k(\lambda, \lambda'))$ is fully specified by a mean $\mu(\lambda)$ and a covariance function $k(\lambda, \lambda')$

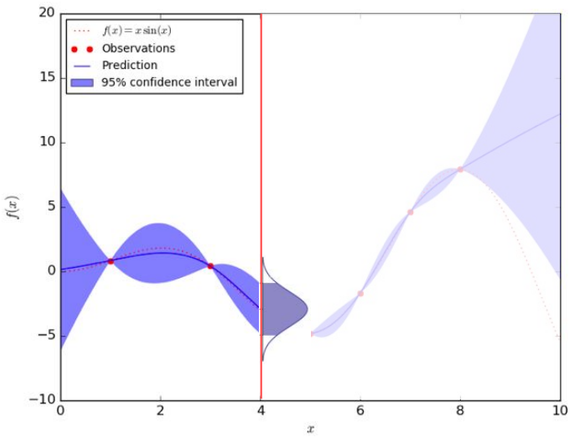


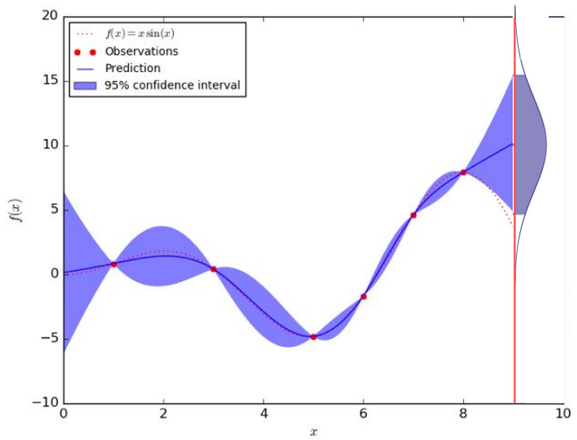


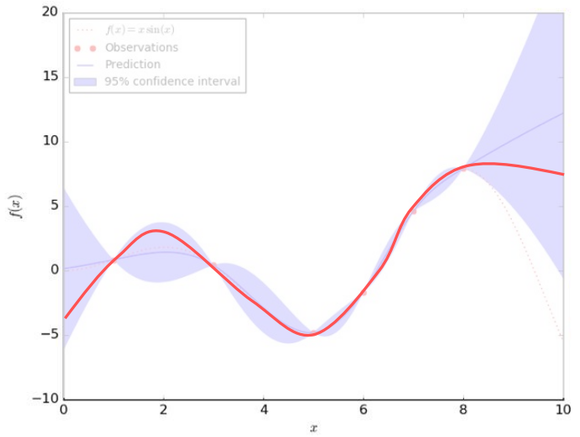
$$f(X_{t_1}), f(X_{t_2}), \ldots, f(X_{t_n}) \sim \mathcal{N}(\mu, \Sigma)$$

Gaussian Process for Model of Model Loss

- A Gaussian process $g(\mu(\lambda), k(\lambda, \lambda'))$ is fully specified by a mean $\mu(\lambda)$ and a covariance function $k(\lambda, \lambda')$



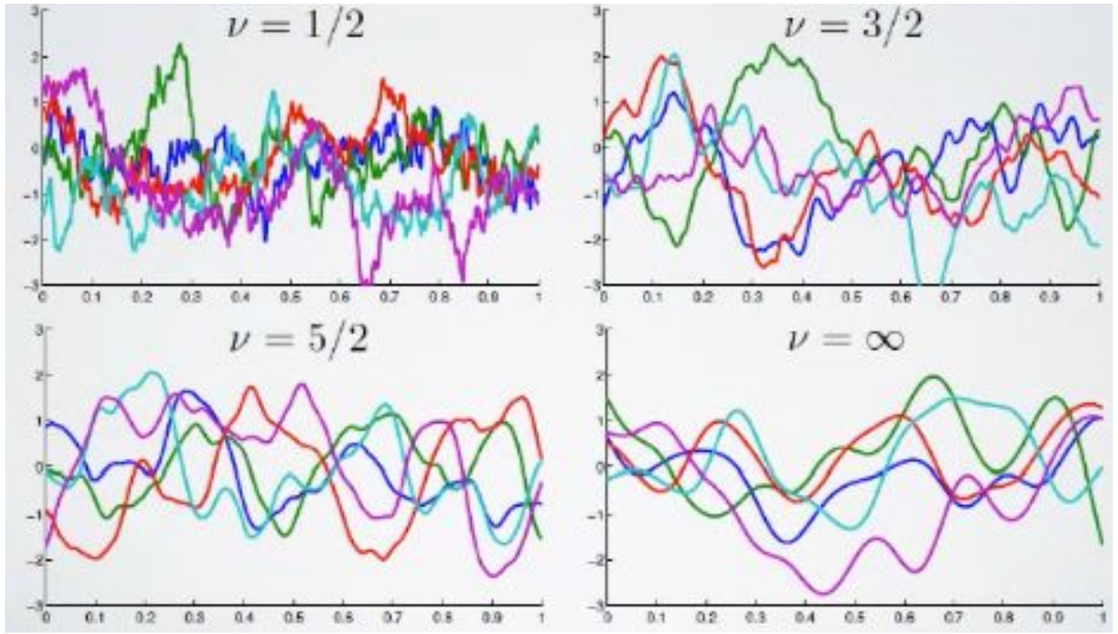




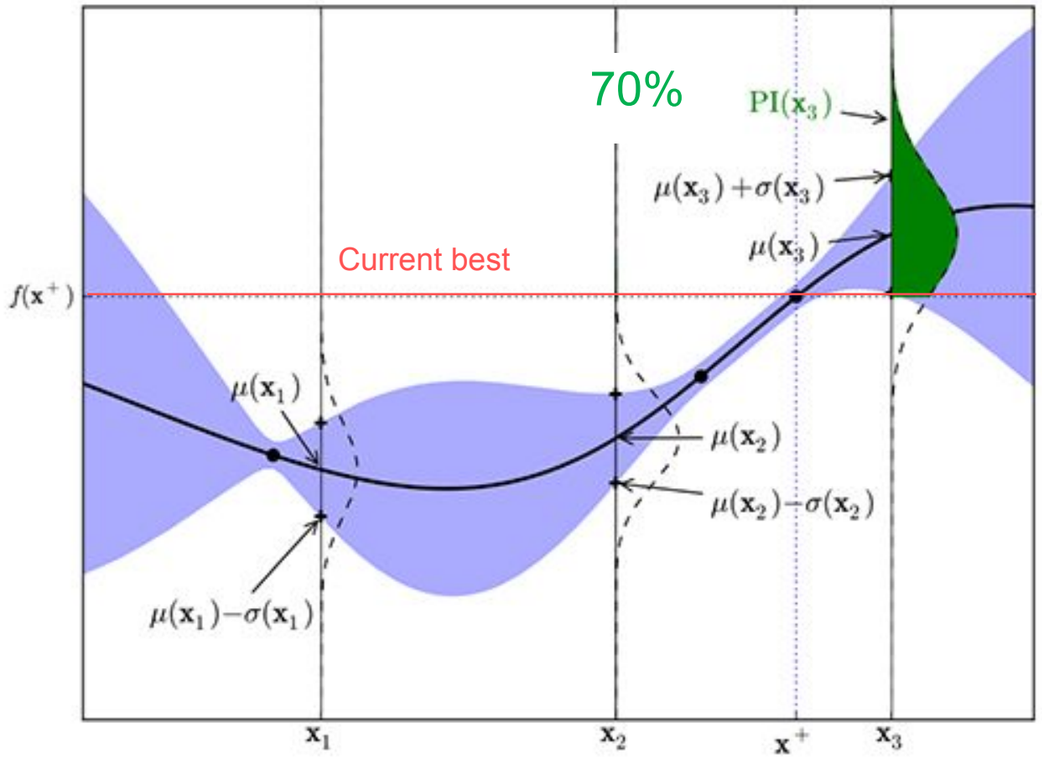
$$f(X_{t_1}), f(X_{t_2}), \ldots, f(X_{t_n}) \sim \mathcal{N}(\mu, \Sigma)$$

Covariant Function

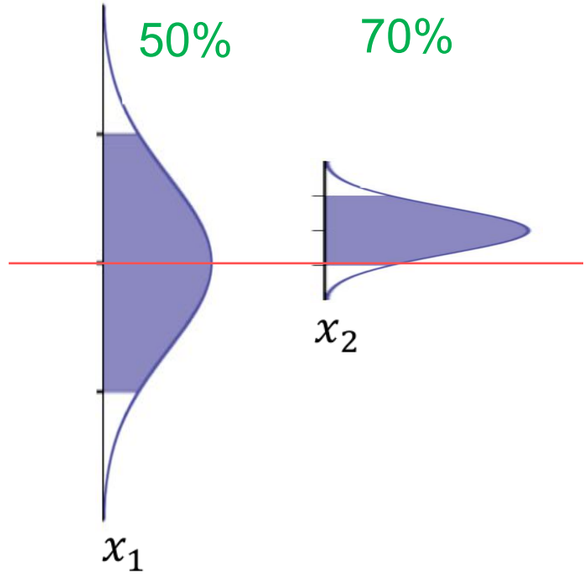
- The mean function is usually constant in Bayesian optimization
- The quality of the Gaussian process is solely dependent on the quality of the covariant function.
- A common choice is the Matren 5/2 kernel, with its hyperparameters integrated out by Markov Chain Monte Carlo



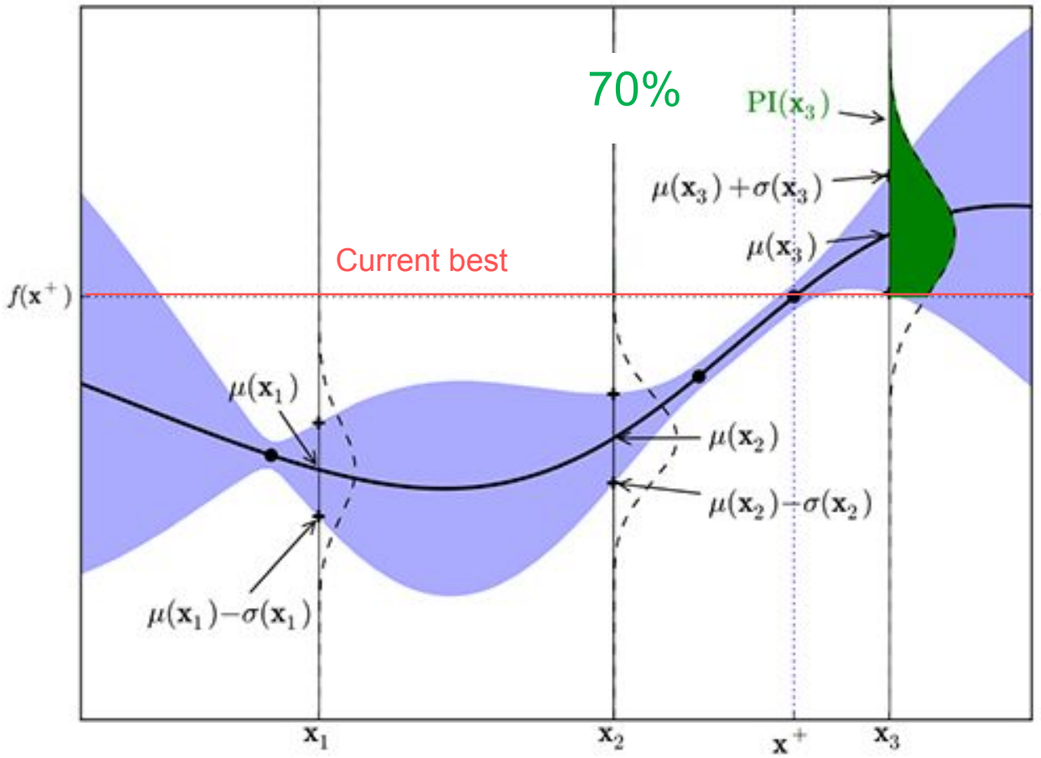
Acquisition Function: Probability of Improvement



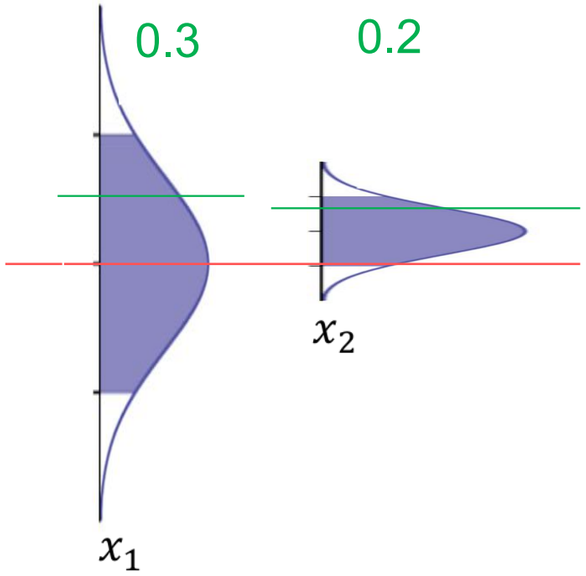
$PI(x_1) < PI(x_2)$



Acquisition Function: Expected Improvement

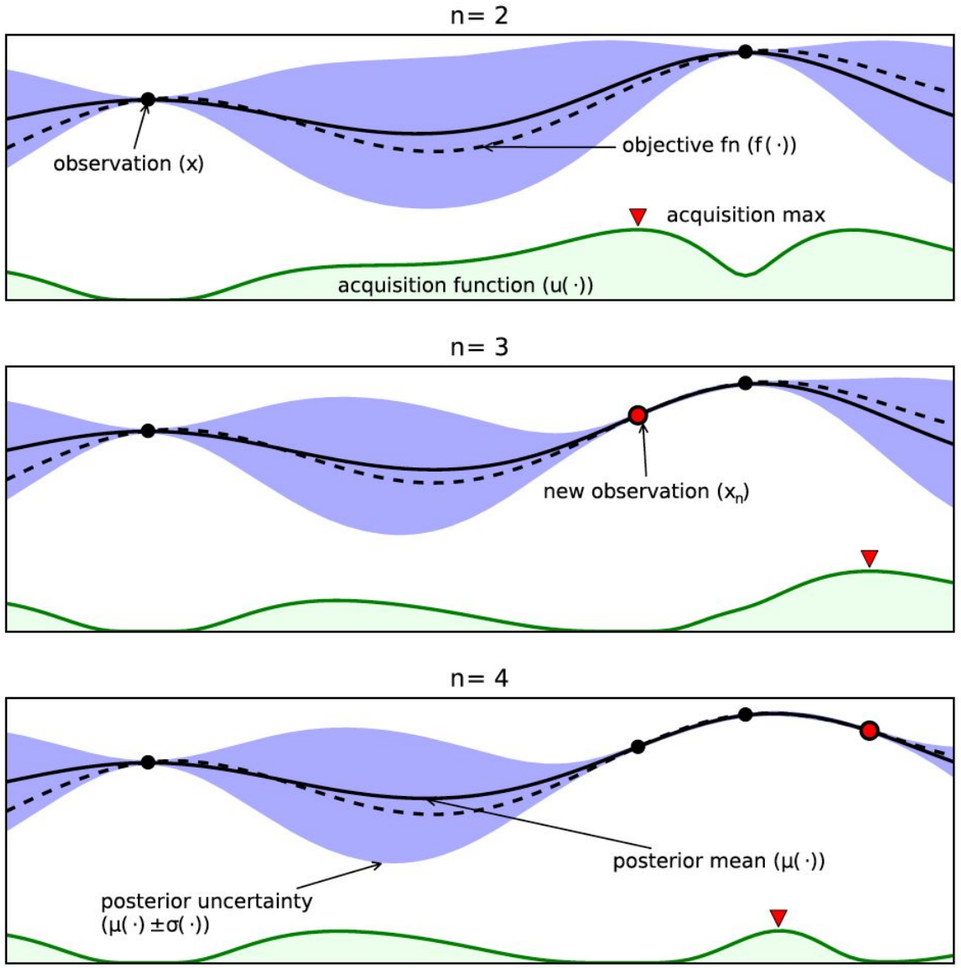


$EI(x_1) > EI(x_2)$



Using Acquisition Function

- Expected improvement [maximining the dashed line] has two components:
  - One is dependent on $-\mu$ [solid line]
  - The other dependent on uncertainty or variance $k(\lambda, \lambda')$ [blue line]
- There fore we maximize the acquisition function wherever:
  - Mean, μ, is low, or
  - Uncertainty, $k(\lambda, \lambda')$, is high.

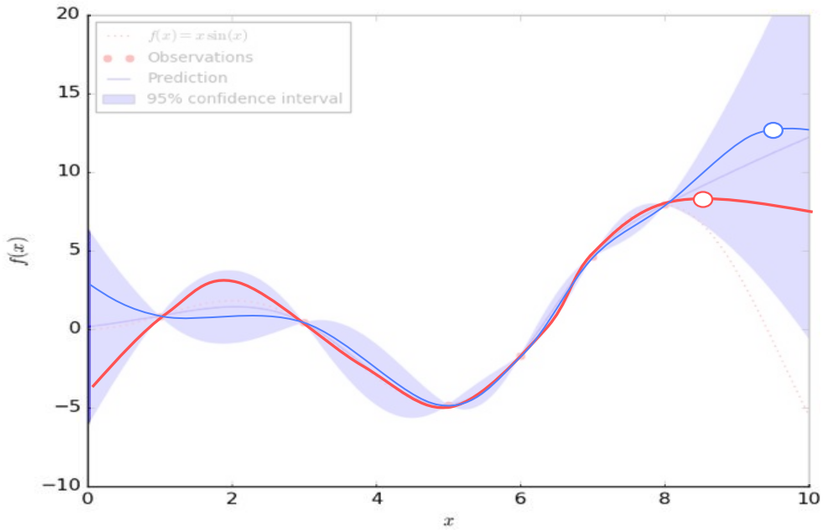


Parallelism through Thompson Sampling

- TS is a decades old heuristics for choosing actions that maximizes the expected reward.

$$a_{n+1} = \arg\max_a f_w(a), \text{ where } \tilde{w} \sim p(w|\mathcal{D}_n)$$

- Sequentiality makes sampling computationally interactable.
- We can use multiple workers for parallelizing TS.



synTS

1. $\{(x'_m, y'_m)\}_{m=1}^{M} \leftarrow$ *wait for all workers to finish.*
2. *Computer the posterior GP*
3. *Draw M samples* $g_m \sim \mathcal{GP}$
4. *Redeploy worker* m *at* $\arg\max g_m, \forall m$

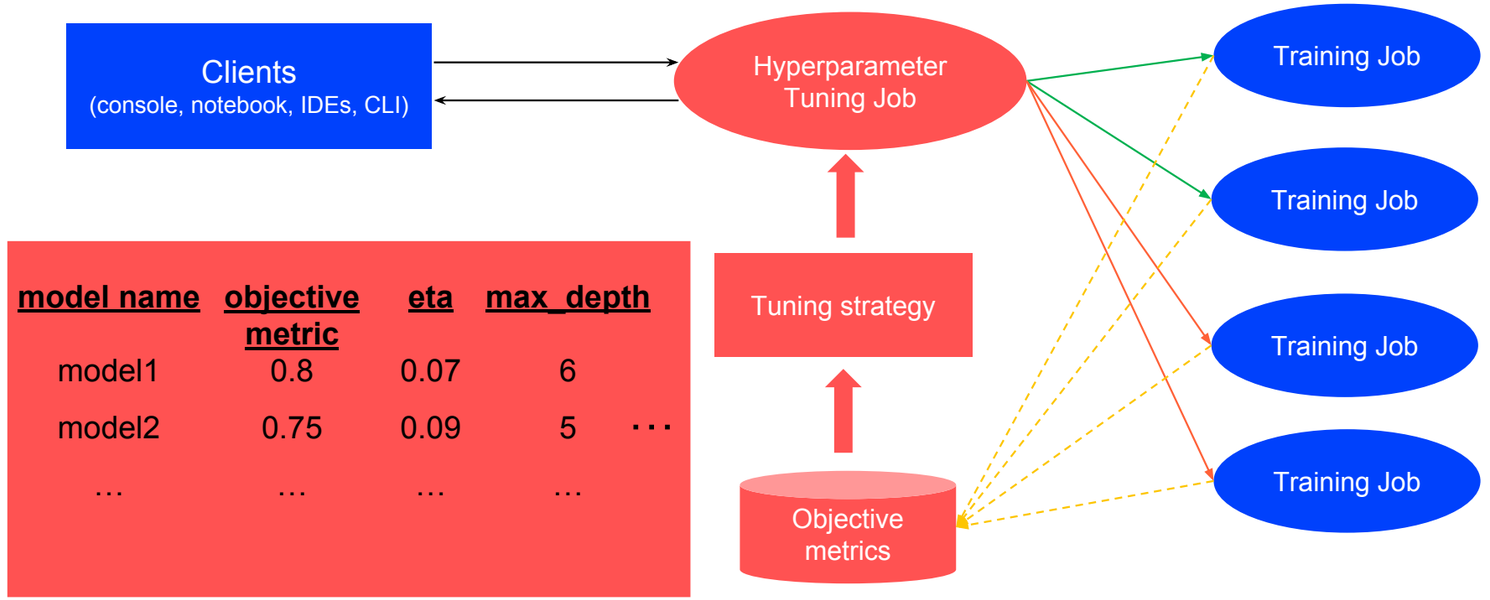
asyTS

1. $(x', y') \leftarrow$ *wait for a worker to finish.*
2. *Computer the posterior GP*
3. *Draw a sample* $g \sim \mathcal{GP}$

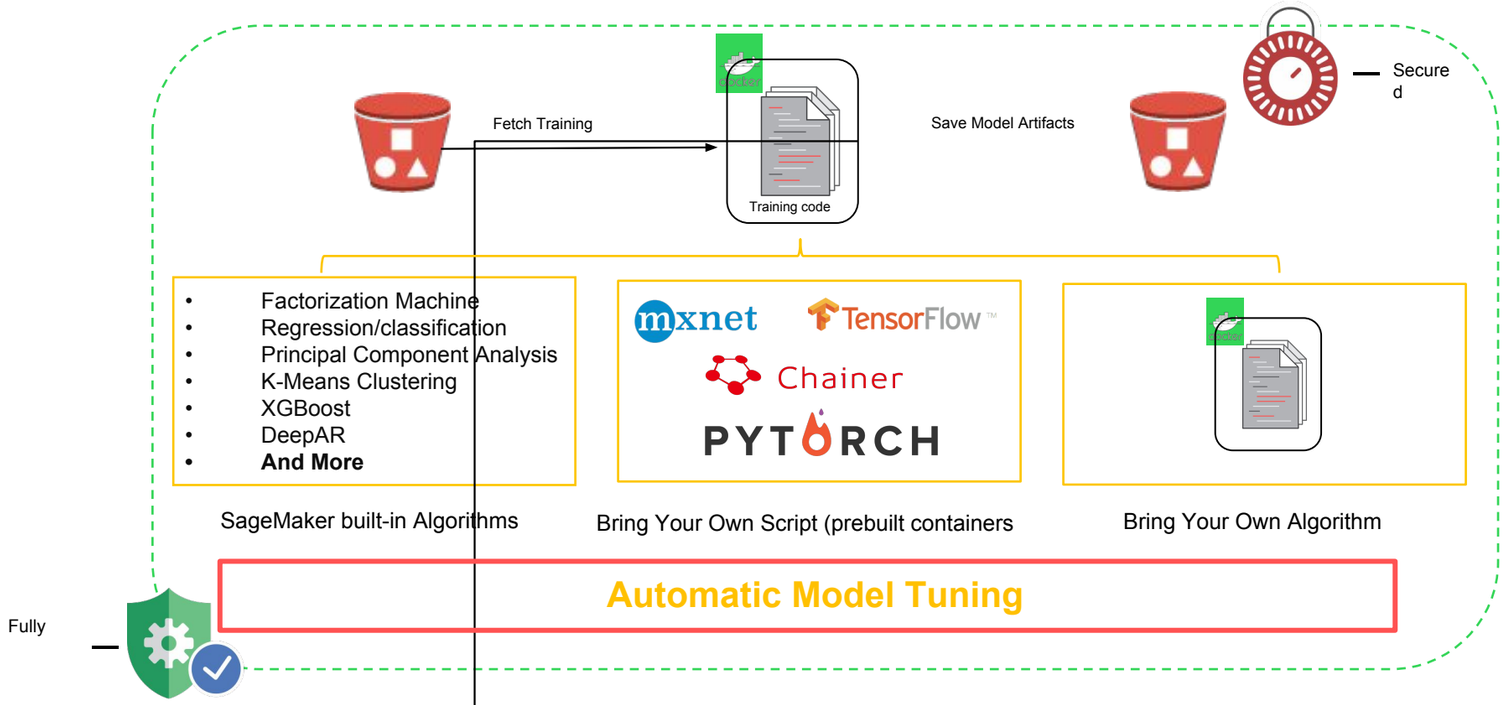
https://www.cs.cmu.edu/~kkandasa/misc/automl-slides.pdf

HPO using in Amazon SageMaker AMT

Automatic Model Tuning - workflow



Automatic Model Tuning - architecture



Input Mapping

Create a unit hypercube from all input hyperparameters

- Continuous – e.g. learning rate $[0,2] \rightarrow [0,1]$
- Integral – e.g. number of epochs $\{1,2,\dots,50\} \rightarrow [0,1]$
- Categorical – e.g. type of activation $\{Sigmoid, RELU, tanh\} \rightarrow [0,1]^3$

DEMO!

Thank you!

Cyrus Vahid
cyrusmv@amazon.com