Recommender Workshop

Part 4: Tuning

Recommender Workshop Agenda

- Part 1: Introduction (You Are Here)
  - Overview of Machine Learning Process, Amazon SageMaker
  - Hands-on: Data Exploration
- Part 2: Collaborative Filtering
  - Core Concepts for Recommendations
  - Hands-on: K-Means Clustering
- Part 3: Matrix Factorization
  - Refining Recommendations
  - Hands-on: Factorization Machine
- Part 4: Hyperparameter Tuning (You Are Here)
  - Key Concepts
  - Hands-on: Hyperparameter Tuning

Common Methods

Photo credit: https://www.flickr.com/photos/ceasedesist/5821282085 (Licensed for commercial use)

Hyperparameters

You can use hyperparameters to finely control training. We've set default hyperparameters for the algorithm you've chosen. Learn more

Key	Value
mode	skipgram
min_count	5
window_size	5
negative_samples	5
epochs	5
vector_dim	100
batch_size	11
learning_rate	0.05
sampling_threshold	0.0001
evaluation	true

Option 1: Grid Search

Grid Search

```python
#hyperparameter optimization
for C in C_values:
    for gamma in gamma_values:
        #train the model for every hyperparameter value pair
        svc = svm.SVC(C=C, gamma=gamma)
        svc.fit(X, y)
        score = svc.score(Xval, yval)

        #rate accuracy of the model using each hyperparam value pair
        if score > best_score:
            best_score = score
            best_params['C'] = C
            best_params['gamma'] = gamma
```

High Dimensional Grid Search

Regularization, Learning Rate

+Network Depth

+Network Width

+Epochs, or Dropout, or Activation, or Optimization Schedule, or …

Points/Axis	Searching	2 HP	3 HP	4 HP	5 HP
2	Corners	4	8	16	32
3	Corners & middle	9	27	81	243
5	Small Variation	25	125	625	3,125
10	Reasonable range	100	1000	10,000	100,000

The Curse of Dimensionality!

Option 2: Random Search

"Compared with neural networks configured by a pure grid search, **we find that random search over the same domain is able to find models that are as good or better within a small fraction of the computation time**."

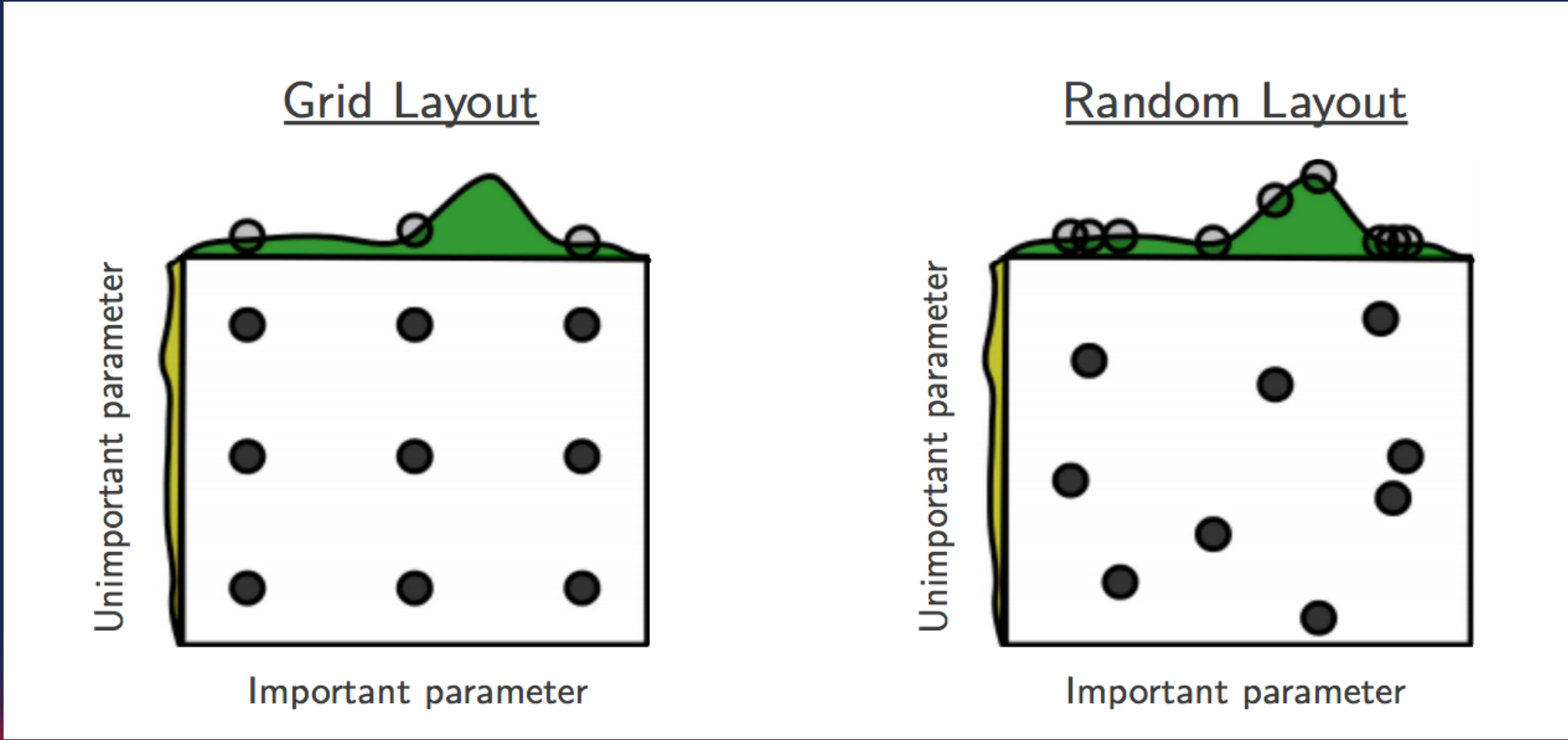
Bergstra, Bengio, "Random Search for Hyper-Parameter Optimization"
https://dl.acm.org/citation.cfm?id=2188395

```python
#for a preset number of iterations
for i in range(10):
    #try random values for each hyperparameter
    svc = svm.SVC(C=randint(0, 9), gamma=randint(0, 3))
    svc.fit(X, y)
    score = svc.score(Xval, yval)

    if score > best_score:
        best_score = score
        best_params['C'] = C
        best_params['gamma'] = gamma
```

eta	eval_metric	gamma	max_depth	min_child_weight	num_round	objective	rate_drop	tweedie_variance_power	FinalObjectiveValue
0.438106	auc	6.657054	2.0	7.991031	50.0	binary:logistic	0.3	1.4	0.633613
0.297931	auc	7.609575	7.0	7.928488	50.0	binary:logistic	0.3	1.4	0.632067
0.268654	auc	6.736518	5.0	9.558967	50.0	binary:logistic	0.3	1.4	0.631693
0.428106	auc	6.747054	2.0	8.081031	50.0	binary:logistic	0.3	1.4	0.631187
0.234279	auc	8.584330	10.0	9.117404	50.0	binary:logistic	0.3	1.4	0.625194
0.224279	auc	8.674330	10.0	9.207404	50.0	binary:logistic	0.3	1.4	0.625155
0.040757	auc	1.896129	4.0	8.946511	50.0	binary:logistic	0.3	1.4	0.614199
0.103715	auc	9.328742	1.0	8.745605	50.0	binary:logistic	0.3	1.4	0.612136
0.944688	auc	2.800941	6.0	6.015777	50.0	binary:logistic	0.3	1.4	0.584103

The Curse of Dimensionality



Option 3: Bayesian Optimization

Those who cannot remember the past are condemned to repeat it….

- George Santayana

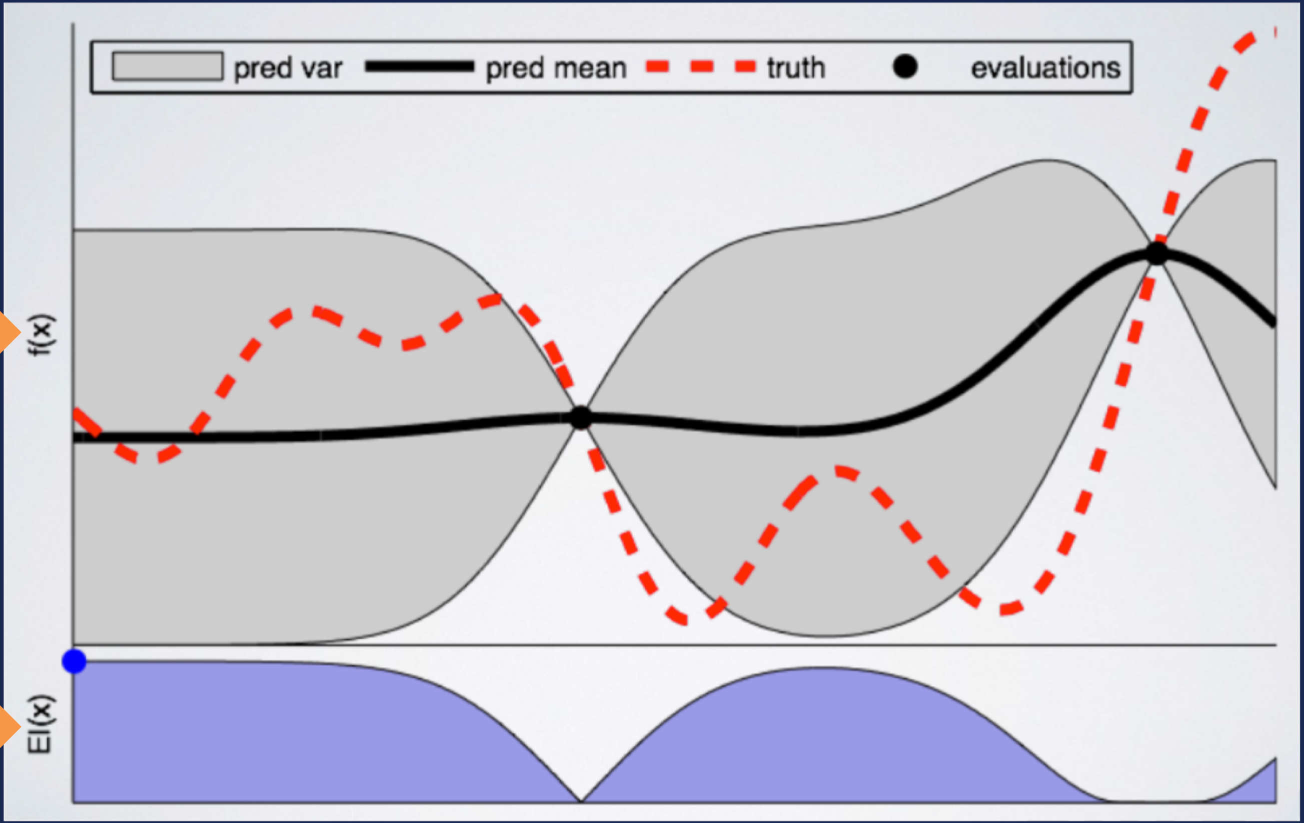
Hyperparameter Tuning w/ Bayesian Optimization

The techniques that Hyperparameter Tuning use are an Amazon SageMaker implementation of Bayesian Optimization, which includes some additional optimizations.

Bayesian Optimization (Visualized)

Toy 1D predictor

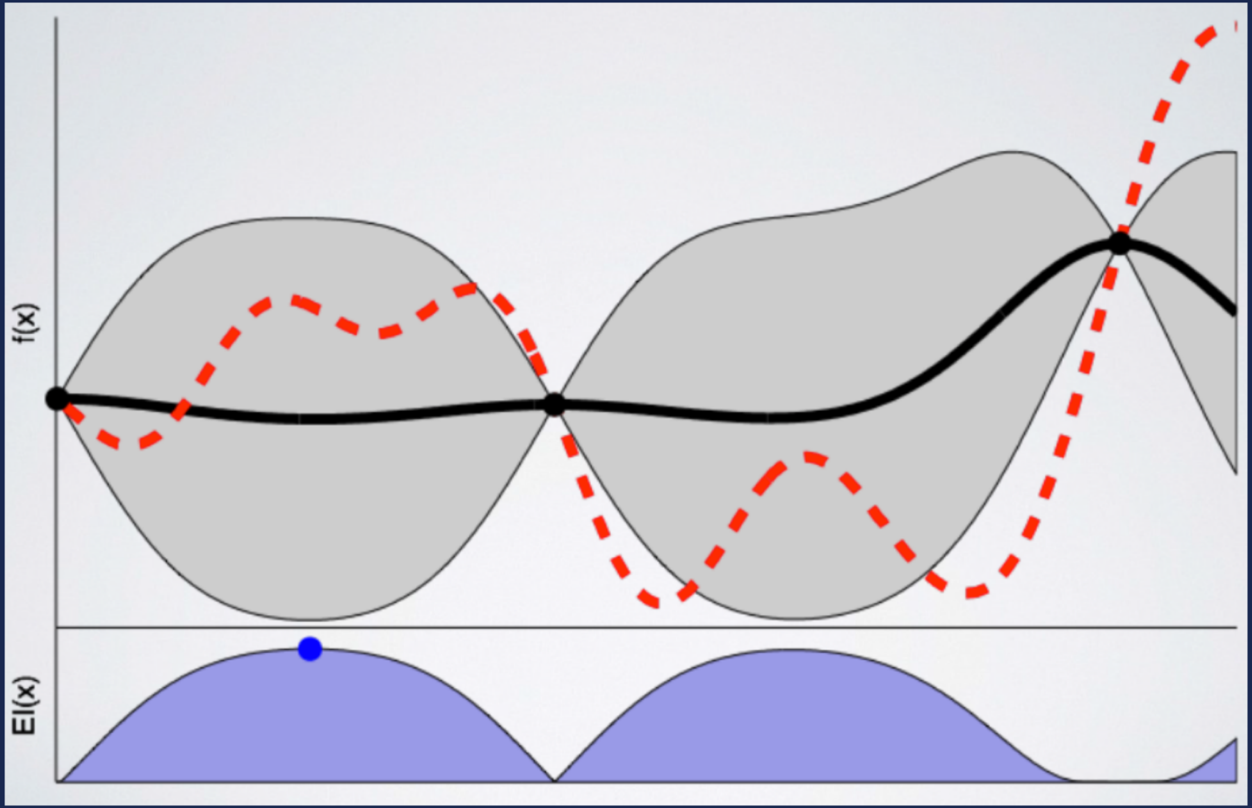
Expected Improvement



Bayesian Optimization (Visualized)

Toy 1D predictor

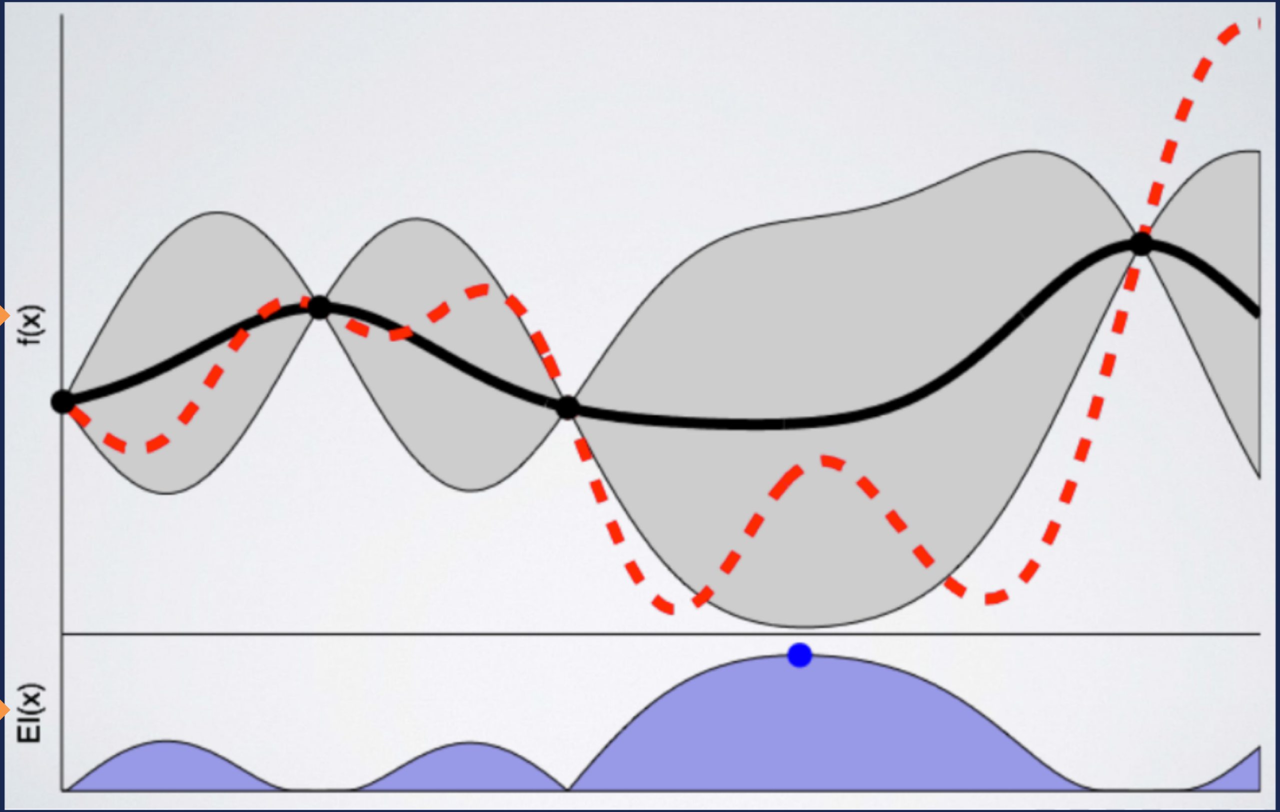
Expected Improvement



Bayesian Optimization (Visualized)

Toy 1D predictor

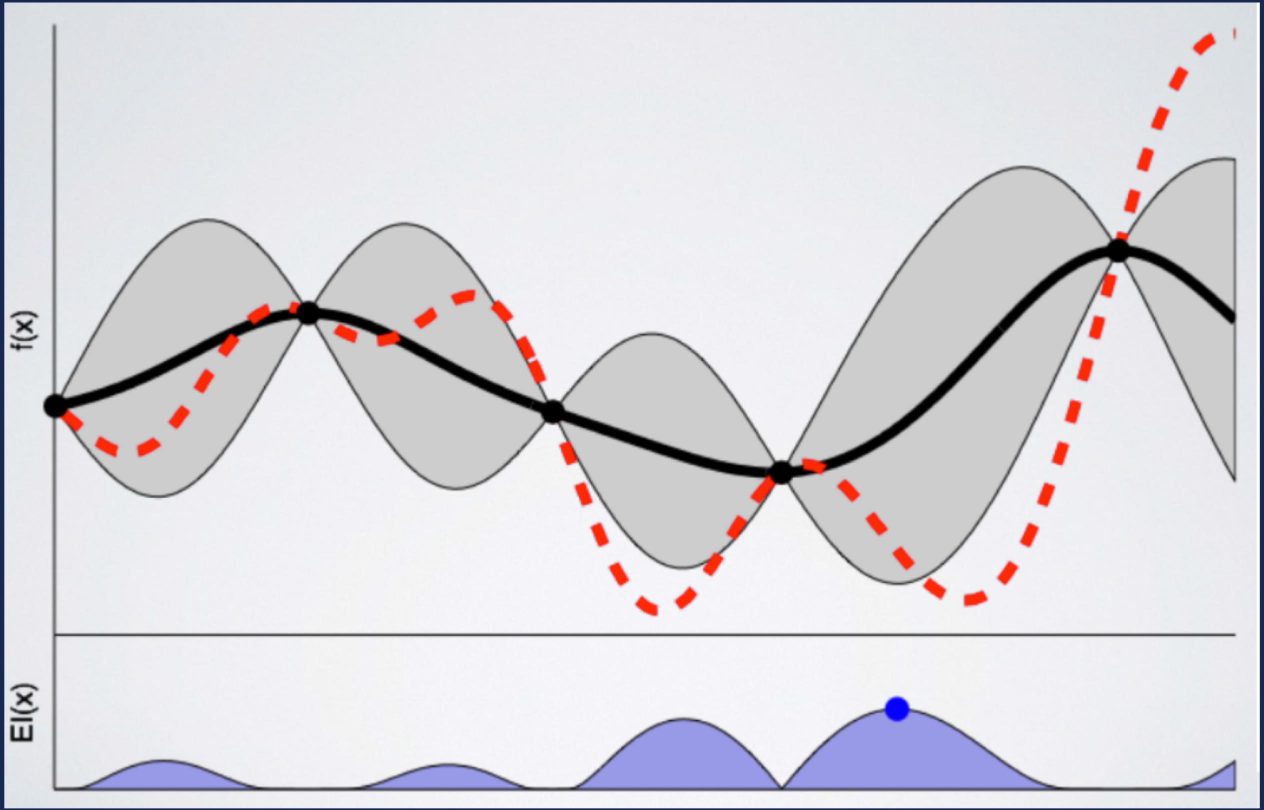
Expected Improvement



Bayesian Optimization (Visualized)

Toy 1D predictor

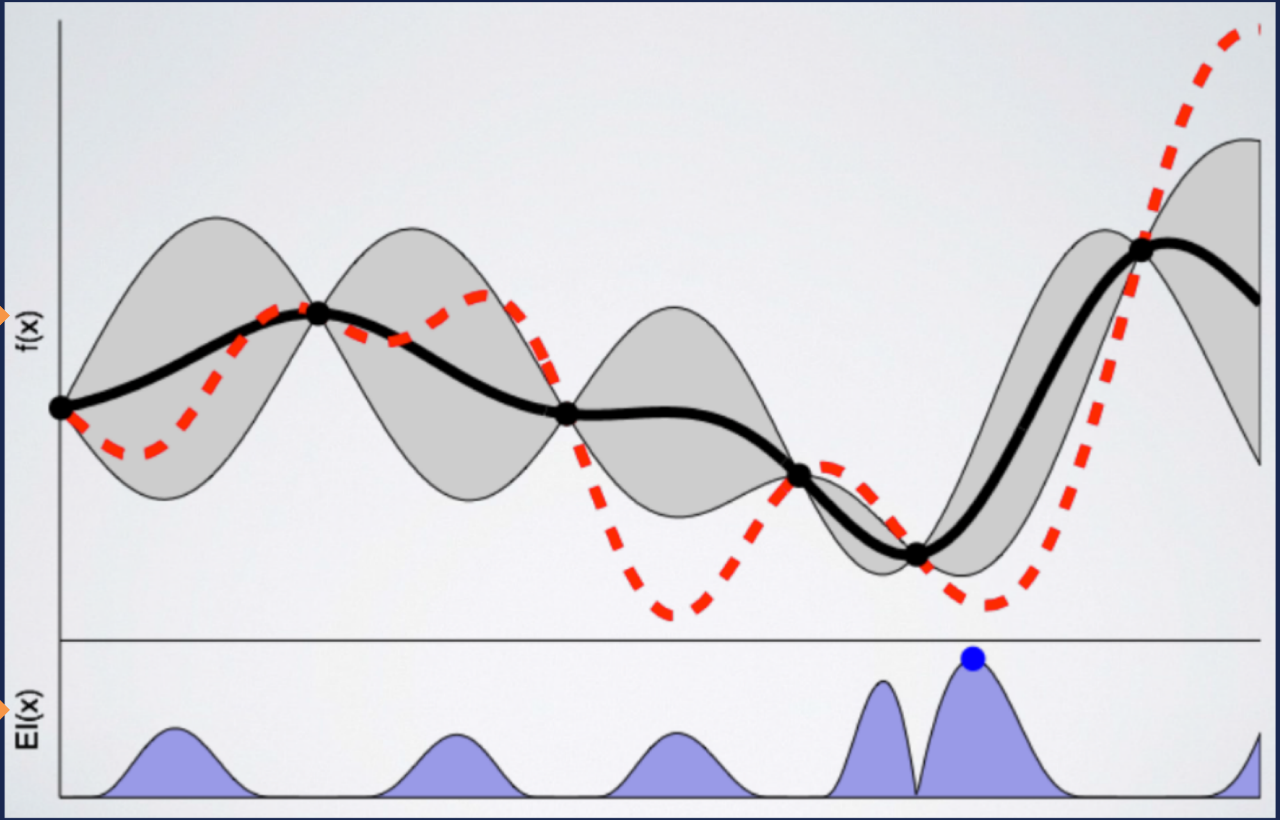
Expected Improvement



Bayesian Optimization (Visualized)

Toy 1D predictor

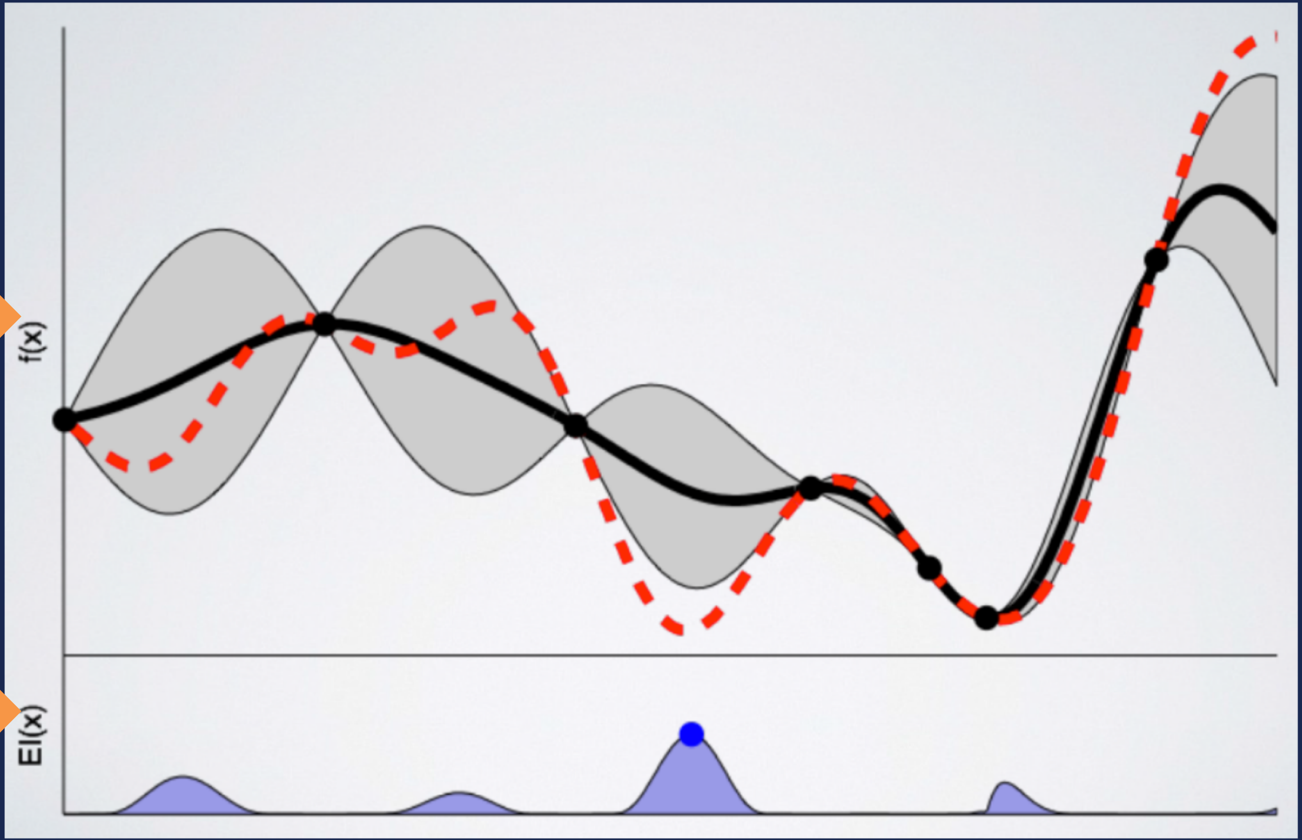
Expected Improvement



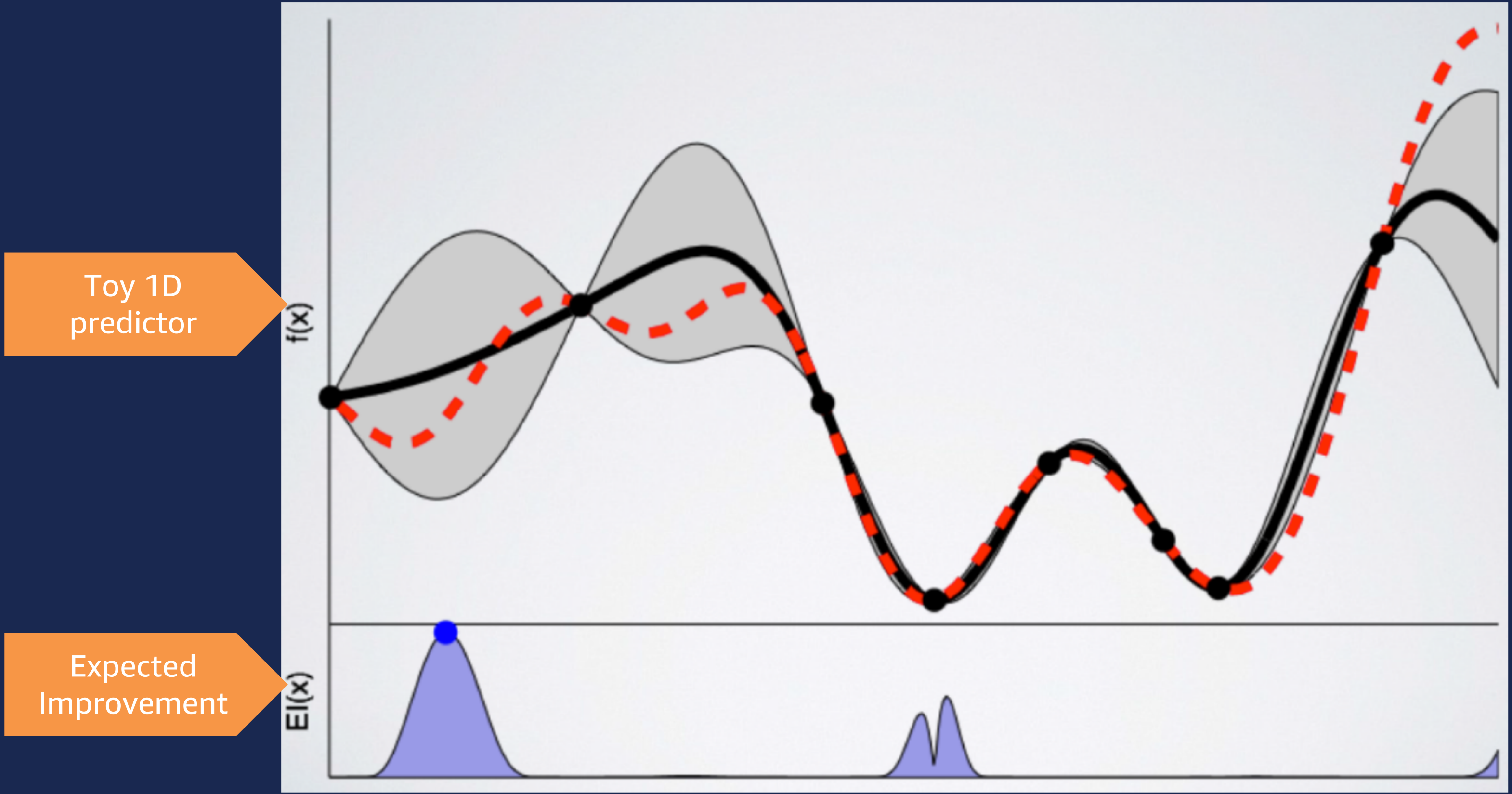
Bayesian Optimization (Visualized)

Toy 1D predictor

Expected Improvement



Bayesian Optimization (Visualized)



Bayesian Optimization Papers

- A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning (https://arxiv.org/abs/1012.2599)
- Practical Bayesian Optimization of Machine Learning Algorithms (https://arxiv.org/abs/1206.2944)
- Taking the Human Out of the Loop: A Review of Bayesian Optimization (https://ieeexplore.ieee.org/document/7352306)

Best Practices

Choosing Number of Hyperparameters

The number of hyperparameters to search over is the single biggest factor that determines how difficult your Hyperparameter Tuning problem is.

While Hyperparameter Tuning supports up to 20 variables at once, limiting your search to a much smaller number is likely to give better results with a realistic amount of training jobs.

Use Logarithmic Scales for Hyperparameters

Hyperparameter Tuning attempts to figure out if your hyperparameters are log-scaled or linear-scaled, but it assumes variables are linearly-scaled to start with, and sometimes the process is slow to realize it should be log-scaled. If you know a variable should be log-scaled, and you can do that conversion yourself, that could improve your optimization.

Choosing the Best Degree of Parallelism

More jobs in parallel gets more work done quickly, but a tuning job can be improved only with successive rounds of experiments.

Typically, running 1 training job at a time achieves the best results with the least amount of compute time. There is a trade-off between minimizing wall-clock time, and minimizing total compute hours used.

MaxParallelTrainingJobs

One-at-a-time

Best final result

Slowest

Some

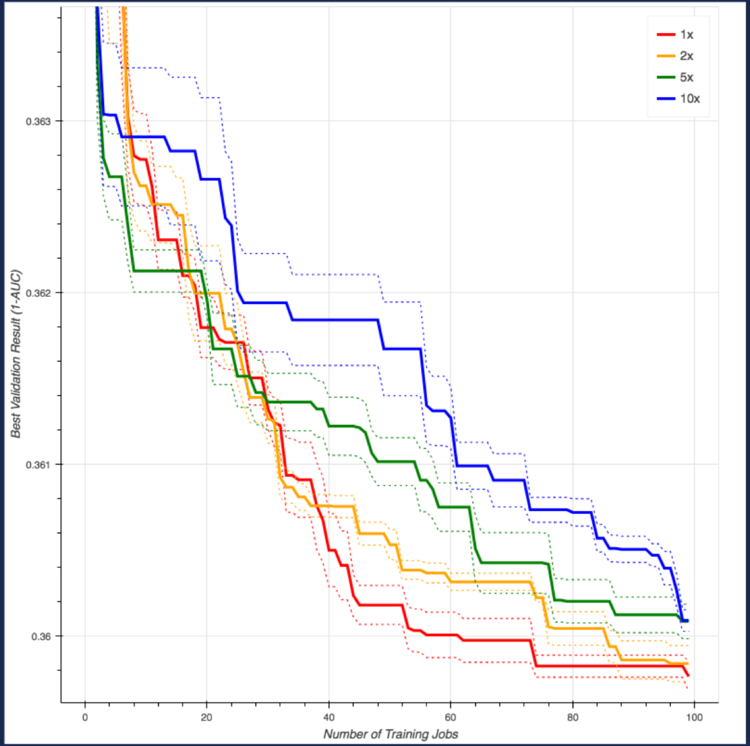
Faster results

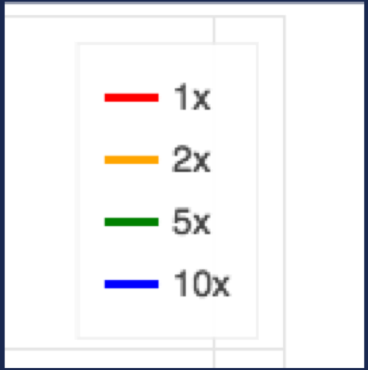
Some loss of final quality

All

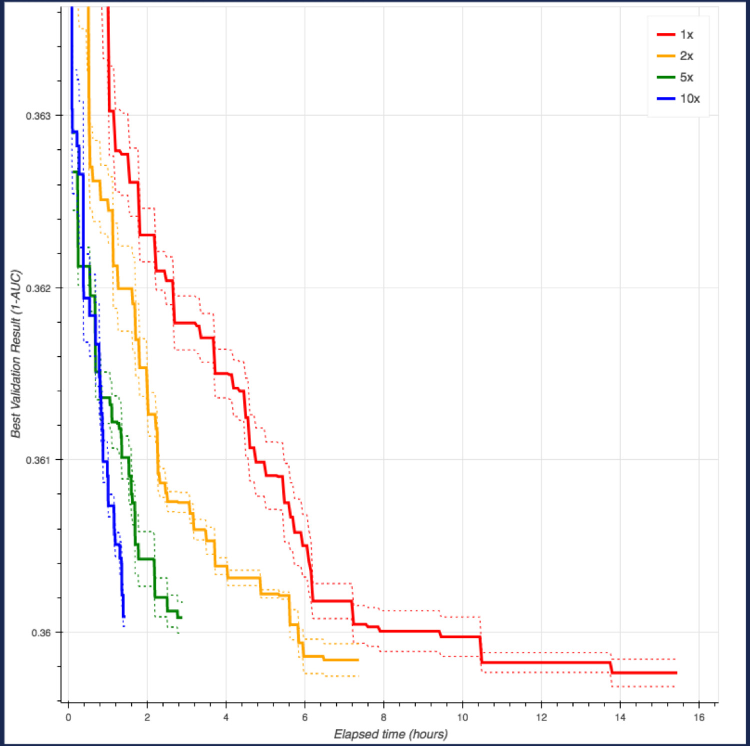
Same as random

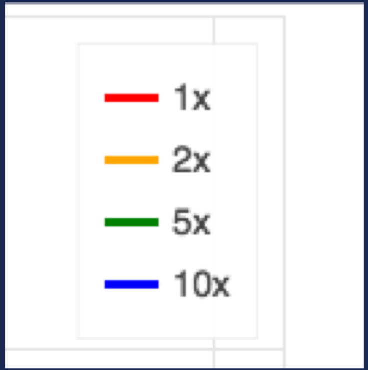
XGBoost


1x
2x
5x
10x
0.363
0.362
0.361
0.36
Best Validation Result (1-AUC)
0
20
40
60
80
100
Number of Training Jobs


1x
2x
5x
10x

XGBoost: Quality vs Time





Running Training Jobs on Multiple Instances

When a training job runs on multiple instances, Hyperparameter Tuning uses the last-reported objective metric from all instances of that training job.

Ensure that you design distributed training jobs so that you get the metric report you want.

This workshop is now concluded

Your feedback is critical, please let us know what you think in our survey.

Thank You!