Plan B: Just tune it!

Bayesian Optimisation for Hyperparameter Tuning

Using the same kernel: https://www.kaggle.com/gaborfodor/from-eda-to-the-top-lb-0-367

Python Package:

'pip install bayesianoptomization'

refer: https://github.com/fmfn/BayesianOptimization/blob/master/README.md

https://www.kaggle.com/omarito/xgboost-bayesianoptimization

Paper: 'Practical Bayesian Optomization of Machine Learning Algorithms'

Bayesian Optimization information on YouTube:

Machine learning - Introduction to Gaussian processes – Nando de Freitas

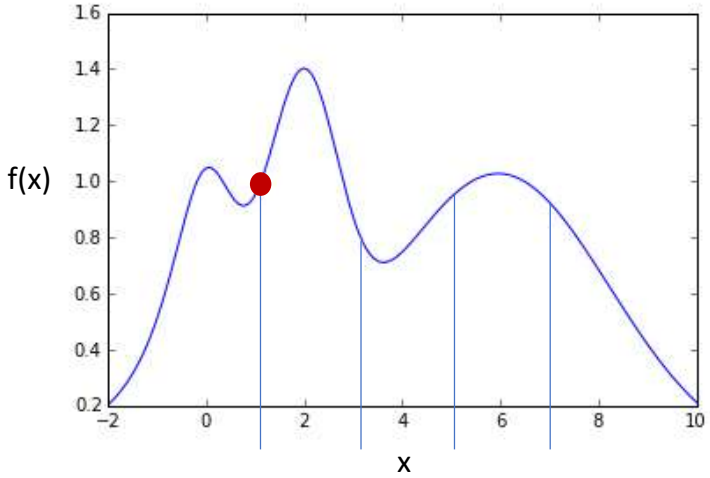
Machine learning - Gaussian processes – Nando de Freitas

Machine learning - Bayesian optimization and multi-armed bandits– Nando de Freitas

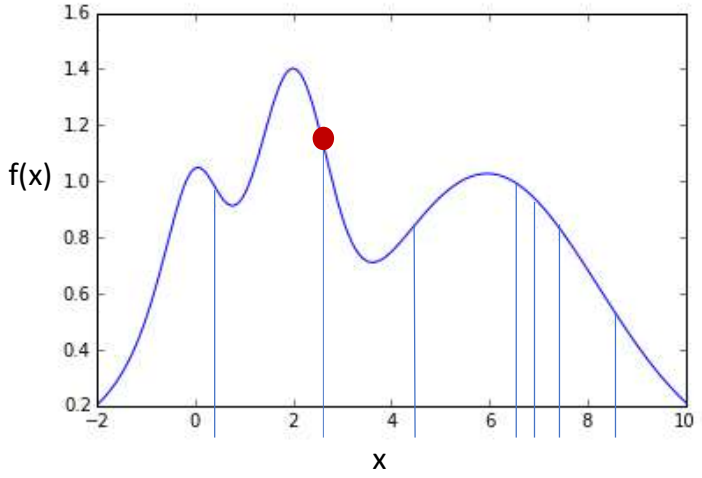
Plan B: Just tune it! - Bayesian Optimisation for Hyperparameter Tuning (continued)

Finding the parameters that give us the best score

Grid Search



Random Search

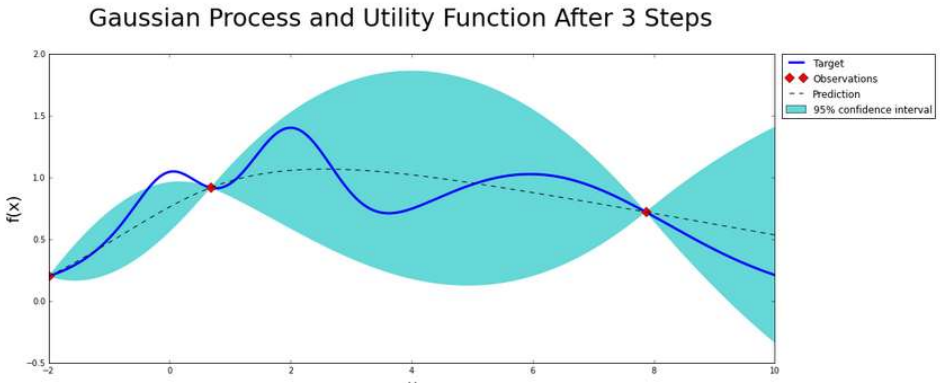


Have to run several times and gradually, manually attempt to zero-in on likely area.

We take the parameters of the highest scoring observation, which might not be optimum.

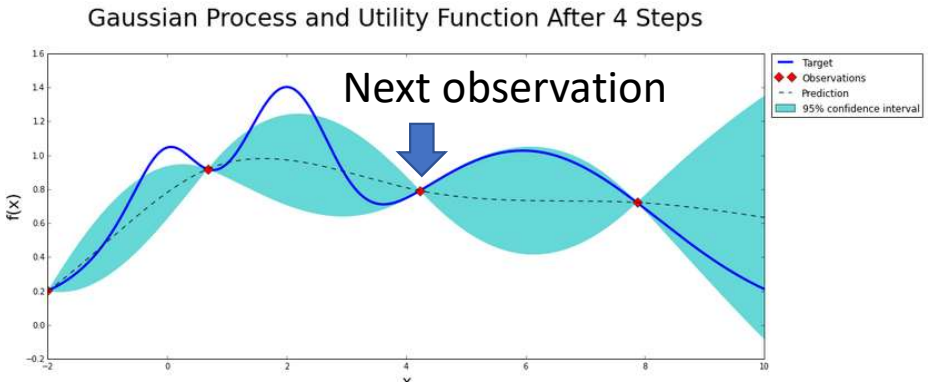
Plan B: Just tune it! - Bayesian Optimisation for Hyperparameter Tuning (continued)

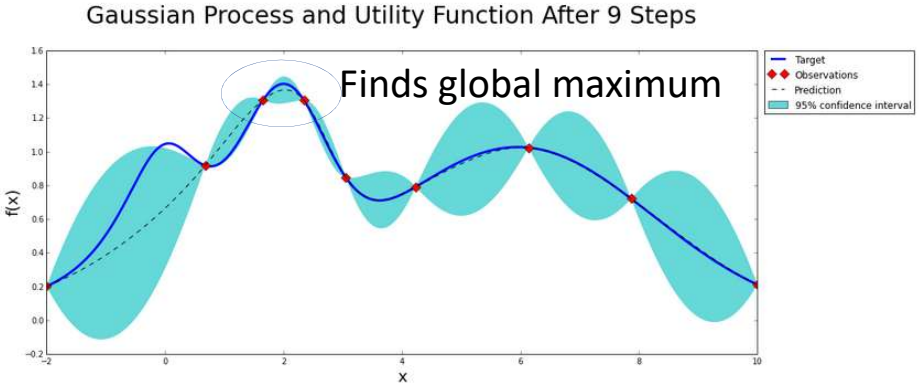
Behold ‘Bayesian Optomization for Hyperparameter Tuning’



Uses all previous observations to construct a probabilistic model for f(x) and calculates where in x to next evaluate.

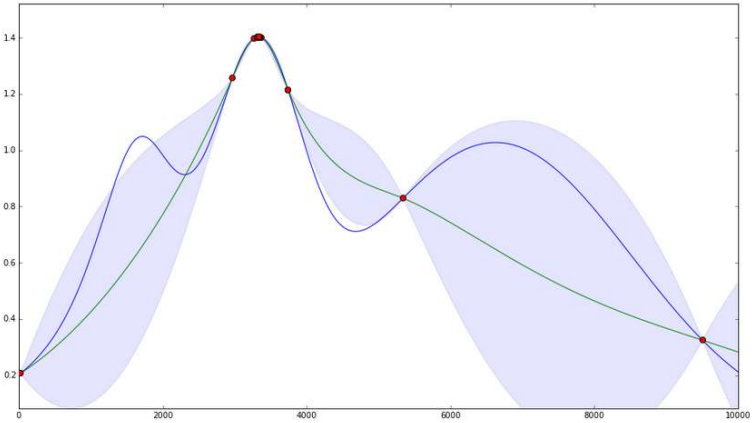
(Doesn’t rely on the local gradient, hessian approx., etc.)

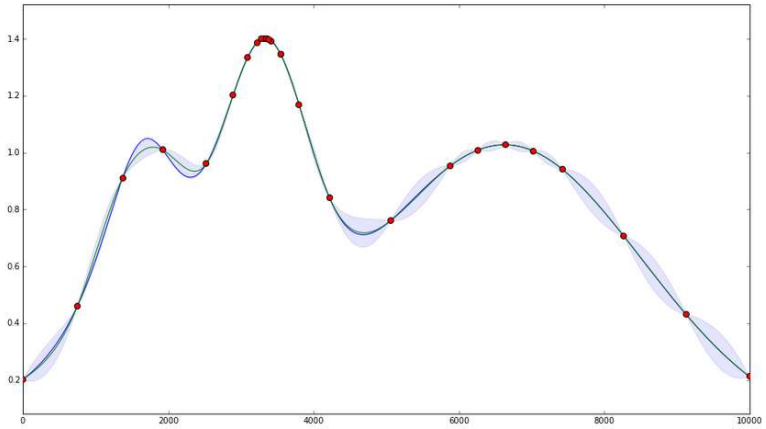




Plan B: Just tune it! - Bayesian Optimisation for Hyperparameter Tuning (continued)

Exploitation versus Exploration





< Exploitation----------Exploration >

Points are around the peaks

Points are spread-out across the range

Acquisition Functions:

- Upper Confidence Bounds
- Expected Improvement
- Probability of Improvement

Plan B: Just tune it! - Bayesian Optimisation for Hyperparameter Tuning (continued)

What does this look like in code?

```
params = {'max_depth': (2, 20),
          'min_child_weight':(1, 120)}

# Build BO object
xgb_bayesopt = BayesianOptomization(FunctionToOpt, params)

# Run BO
xgb_bayesopt.maximize(init_points=4, n_iter=200, acq="ucb", kappa=10)

print('Best Score='xgb_bayesopt.res['max']['max_val']
print('Best Params= ', xgb_bayesopt.res['max']['max_params'])
```

```
def FunctionToOpt(param1, param2):
    set default xgb parameters
    xgb.cv(…. cross validation
    return score
```

Plan B: Just tune it! - Bayesian Optimisation for Hyperparameter Tuning (continued)

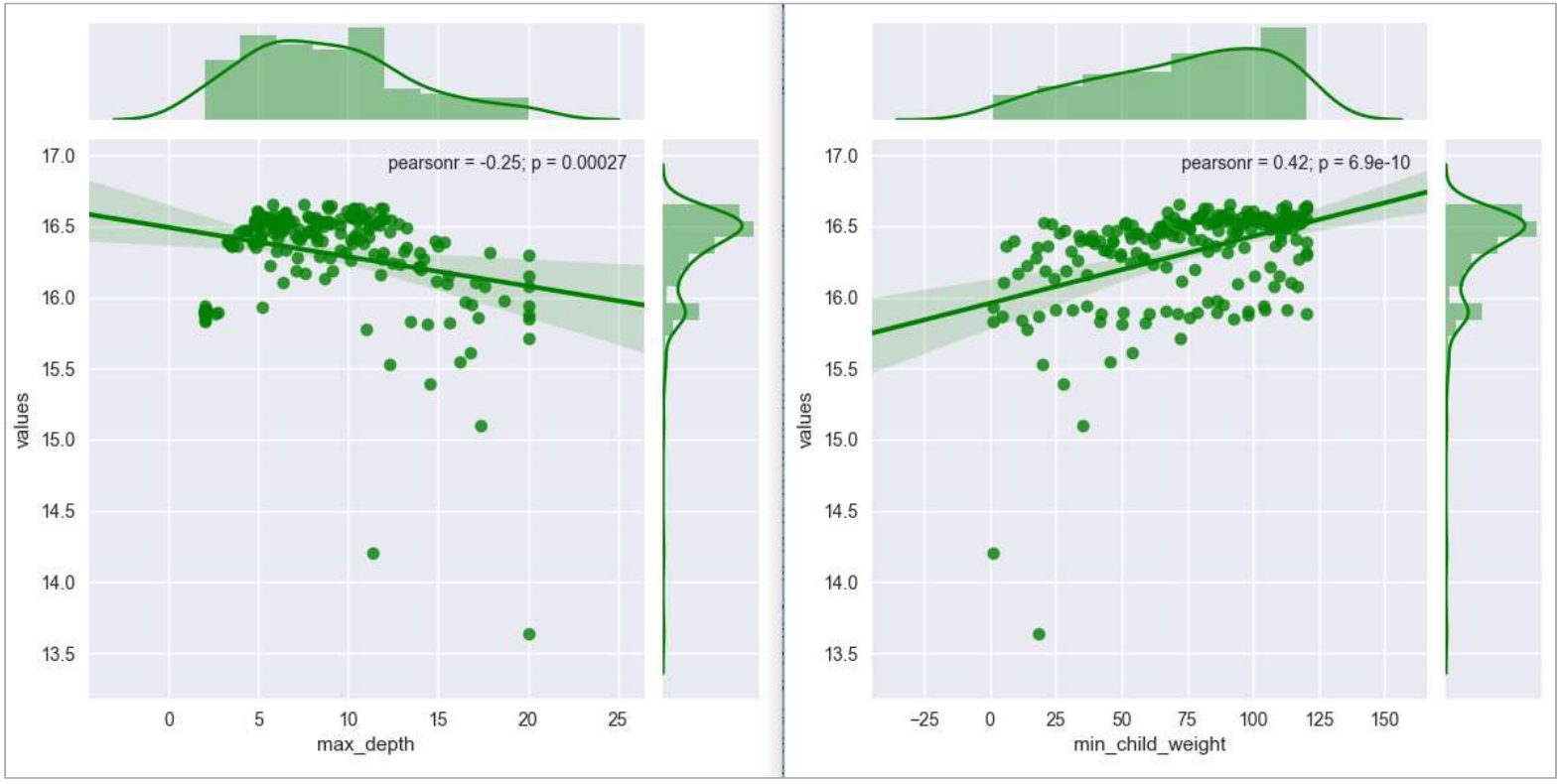
Phase 1: Initial Tune for ‘Tree Complexity’

xgb_bayesopt.maximize(init_points=4, n_iter=200, acq="ucb", kappa=10)

Max_depth (2, 20)

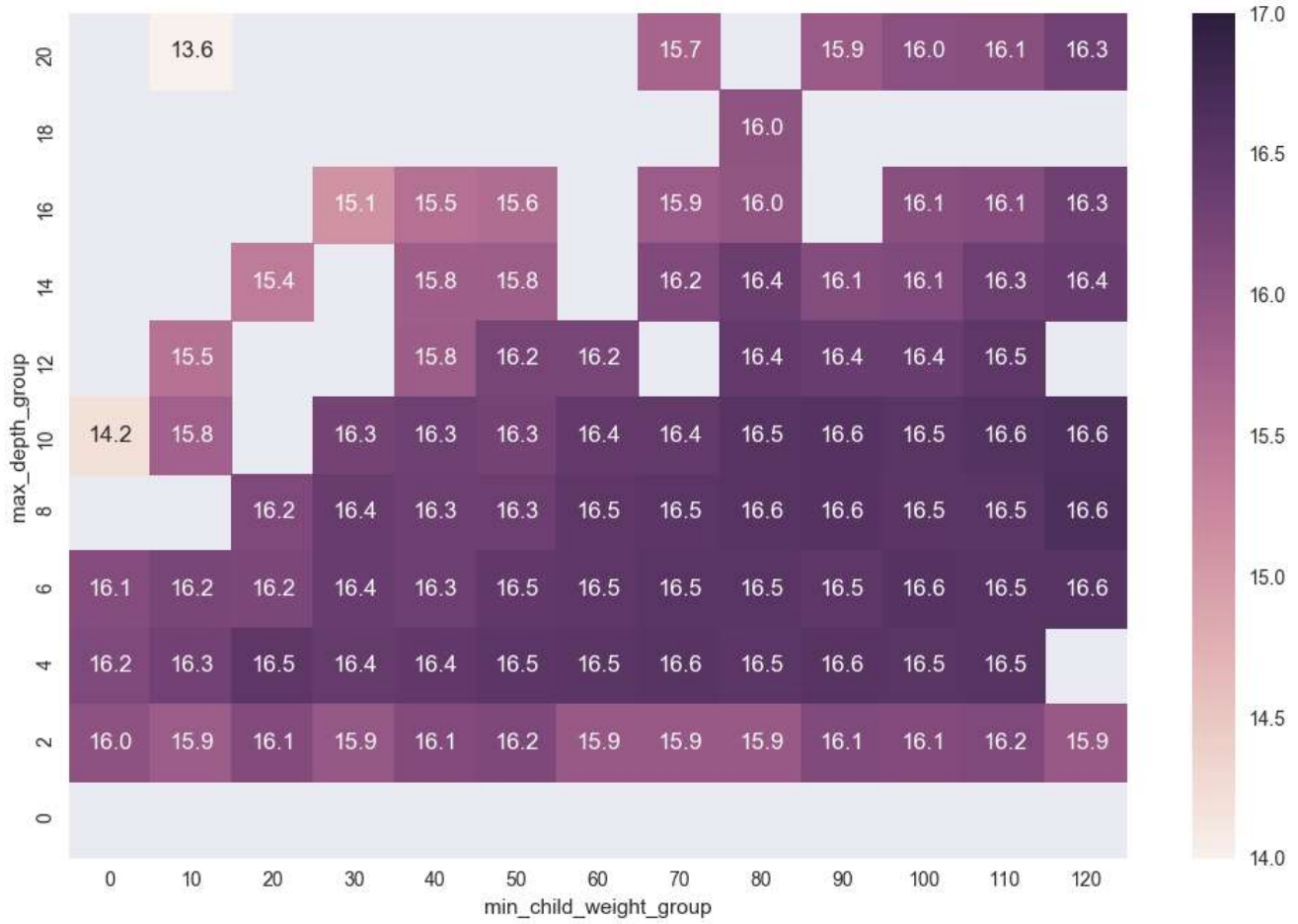
Mcw (0, 120)

Seems to be working!



Plan B: Just tune it! - Bayesian Optimisation for Hyperparameter Tuning (continued)

The same data on a heatmap



Plan B: Just tune it! -

Phase 2: Refined

Max_depth (4, 10)

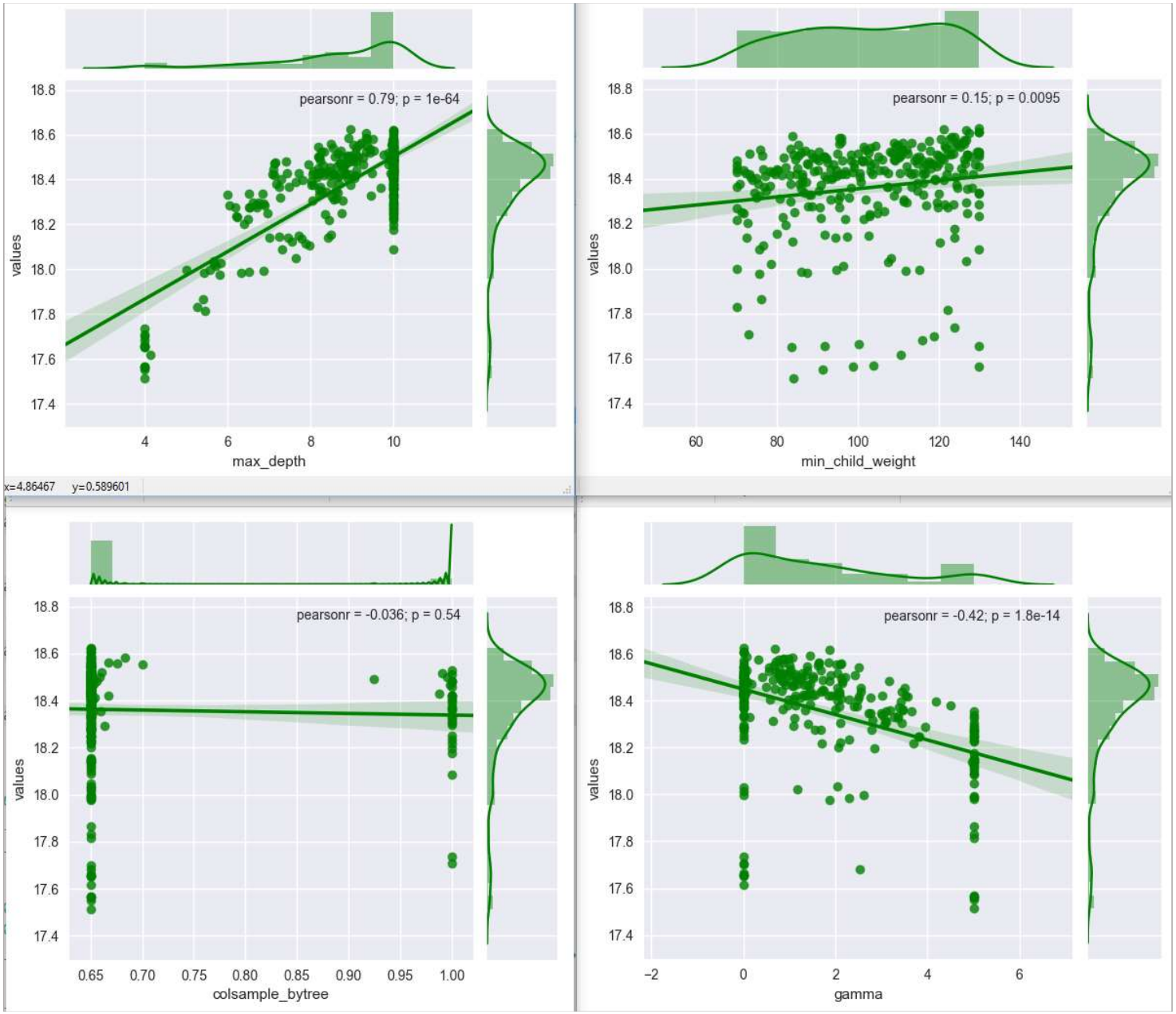
Mcw (70, 130)

Colsample (0.65, 1)

Gamma (0, 5)

xgb_bayesopt.maximize(init_points=10, n_iter=250, acq="ucb", kappa=10)

xgb_bayesopt.maximize(n_iter=50, acq="ucb", kappa=5) - more exploitation



Plan B: Just tune it! - Bayesian Optimisation for Hyperparameter Tuning (continued)

Final score

	Kernel	BO Tuning
max_depth	10	9 (8.96096)
min_child_weight	50	130
colsample_bytree	0.3	0.65
subsample	0.8	1
gamma	0	0
lambda	1	0
eta	0.3	0.3
Public LB Score:	.38948 221st	.38555 - up 12 places to 209th

After increasing number boost rounds to 200 and lambda=1: .38379 - up 21 places to 188th.

Thoughts:

- It works! A good tool to have in your ML toolbox.
- It might not work for every algo, or might take a very long time – your ML function might not have a clear global maximum.
- Might want to use Grid and Random search for fine tuning.

Plan B: Just tune it! - Bayesian Optimisation for Hyperparameter Tuning (continued)

Output – 1 of 3

Initialization

Step	Time	Value	colsample_bytree	gamma	max_depth	min_child_weight
1	07m33s	17.98450	0.7666	2.8782	5.9497	105.6152
2	12m06s	18.45814	0.7534	1.0072	9.6425	77.9154
3	12m20s	18.17053	0.8148	4.8755	9.4840	99.5848
4	06m48s	17.97461	0.6859	3.7285	5.7191	81.2223
5	13m19s	18.35680	0.9347	2.2670	8.6908	71.2115
6	11m34s	18.34301	0.8892	1.7854	7.9435	127.7680
7	09m15s	18.40556	0.6573	1.6714	7.1769	107.9199
8	10m21s	18.14204	0.9467	3.2399	6.0114	108.1840
9	11m12s	18.19491	0.7256	4.3344	8.3399	117.6578
10	11m20s	18.25520	0.9730	3.2996	7.1723	107.3654

Bayesian Optimization

Step	Time	Value	colsample_bytree	gamma	max_depth	min_child_weight
11	12m27s	18.39958	1.0000	0.0000	10.0000	119.1685
12	14m13s	18.37483	1.0000	0.0000	10.0000	90.7401
13	10m02s	18.08695	1.0000	5.0000	10.0000	75.7347
14	07m18s	17.73716	1.0000	0.0000	4.0000	123.8104
15	14m36s	18.53077	1.0000	0.0000	10.0000	110.8030
16	07m18s	17.70749	1.0000	0.0000	4.0000	72.9897
17	14m46s	18.48363	1.0000	0.0000	10.0000	100.8947
18	12m54s	18.33764	0.6500	5.0000	10.0000	130.0000
19	11m01s	18.60940	0.6500	0.0000	10.0000	130.0000
20	10m17s	18.44381	0.6500	0.0000	10.0000	70.0000
21	09m53s	18.50843	0.6500	0.0000	10.0000	83.9531
22	09m22s	18.54446	0.6500	0.0000	10.0000	125.3189
23	11m31s	18.27786	0.6500	5.0000	10.0000	87.6719

Plan B: Just tune it! - Bayesian Optimisation for Hyperparameter Tuning (continued)

Output – 2 of 3

80	10m44s	18.40920	0.6500	0.0000	8.1689	93.6793
81	11m47s	18.42116	0.6500	3.5240	10.0000	96.2456
82	13m20s	18.48451	0.6500	3.1628	10.0000	126.3524
83	09m13s	18.46692	0.6500	1.3174	7.9541	120.0331
84	11m58s	18.55993	0.6500	1.6409	10.0000	111.0803
85	05m24s	17.66326	0.6500	0.0000	4.0000	100.3329
86	10m41s	18.33460	0.6500	3.3466	8.2080	130.0000
87	14m43s	18.36625	1.0000	0.0000	8.2461	107.1984
88	12m05s	18.56715	0.6500	0.7037	10.0000	117.6137
89	07m58s	17.98238	0.6500	5.0000	6.3313	87.4997
90	09m37s	18.37628	0.6500	2.3963	8.0283	75.2650
91	10m32s	18.43358	0.6500	0.0000	8.3380	89.2097
92	10m43s	18.62450	0.6500	0.0000	8.9610	130.0000
93	10m42s	18.24741	0.6500	5.0000	8.3553	127.1397
94	10m52s	18.43957	0.6500	2.0941	10.0000	106.4325
95	12m25s	18.55035	0.6500	1.6106	10.0000	124.9647
96	11m38s	18.43153	0.6500	2.0767	10.0000	70.0000
97	10m13s	18.51122	0.6500	0.0030	10.0000	123.5595
98	10m28s	18.46370	0.6500	0.0000	8.7776	116.9508
99	09m51s	18.42204	0.6500	1.8425	8.0556	79.5268
100	10m09s	18.43635	0.6500	0.0000	8.0592	96.7674
101	10m43s	18.44922	0.6500	0.0000	8.1915	125.4842
102	12m03s	18.47762	0.6500	2.1506	10.0000	117.7193
103	09m22s	18.14004	0.6500	5.0000	7.0027	94.5023
104	11m54s	18.50211	0.6500	2.2150	10.0000	85.3755
105	11m12s	18.52930	0.6500	1.6908	10.0000	75.8591
106	05m27s	17.61656	0.6500	0.0000	4.1408	110.5171
107	10m16s	18.34911	0.6522	2.7192	8.2502	114.7363

Plan B: Just tune it! - Bayesian Optimisation for Hyperparameter Tuning (continued)

Output – 3 of 3

286	09m57s	18.37832	0.6500	0.0000	10.0000	85.4751	
287	11m13s	18.50637	0.6500	0.0000	8.8868	98.1929	
288	12m25s	18.58427	0.6832	1.0682	9.3430	123.0962	
289	12m03s	18.49375	0.6500	1.2400	10.0000	110.1786	
290	11m52s	18.49639	0.6500	0.7926	9.2754	88.8867	
291	10m03s	18.44234	0.6500	0.8345	9.1096	76.7007	
292	12m54s	18.42613	0.6500	1.5573	9.4944	116.7067	
293	13m42s	18.44088	0.6500	3.4434	10.0000	116.2149	
294	13m54s	18.45726	0.6500	3.5036	10.0000	130.0000	
295	11m30s	18.41241	0.6500	1.2279	8.6666	94.7756	
296	12m02s	18.58103	0.6500	0.9729	9.5037	95.7902	
297	11m30s	18.58678	0.6500	0.0696	8.9281	123.9359	
298	13m59s	18.32952	1.0000	0.1404	9.4577	87.9437	
299	10m55s	18.48143	0.6500	0.0992	9.0312	90.4421	
300	11m12s	18.54269	0.6500	0.5650	8.5465	123.7006	
301	11m50s	18.55639	0.6756	2.1134	9.9664	126.4045	
302	11m50s	18.32590	0.6500	3.3551	8.5254	95.7639	
303	11m02s	18.50182	0.6500	0.0000	8.7845	110.1005	
304	12m53s	18.35564	0.6500	3.6561	10.0000	85.9133	
305	08m26s	18.38603	0.6500	0.0090	10.0000	87.0832	
306	11m33s	18.48861	0.6500	0.0023	10.0000	89.5556	
307	14m43s	18.32749	1.0000	1.1020	10.0000	75.6793	
308	11m30s	18.47351	0.6500	0.1828	8.8691	92.5089	
309	13m21s	18.51181	0.6500	0.7666	10.0000	73.2317	
310	10m32s	18.39565	0.6500	0.0000	7.8183	119.0580	

```
Best Score=   18.6245001512  >>>  0.377258
Best Params=  {'max_depth': 8.9609615510522485, 'min_child_weight': 130.0, 'colsample_bytree': 0.65000
<<< END >>> 3319.8 minutes
```