Parameter Tuning Off the Grid

Henry Moss, David Leslie, Paul Rayson

h.moss@lancaster.ac.uk
https://github.com/henrymoss

Plan

1. Introduce simple NLP task
2. Grid Search
3. Random Search
4. Bayesian Optimisation

Generic NLP task

IMDB Data

- 25,000 positive and 25,000 negative movie reviews.

× I'm not a big fan of musicals, although this technically might not qualify as a musical. It was mediocre at best. Hudson seems totally out of kilter in this role. It just didn't work for me. Don't waste your time!

✓ This is a must-see movie. You will laugh, you will cry, and when it's over you'll wish there were more. Well-written and compelling, this movie draws you in and holds on tight. The casting was perfect, the characters purposeful, and the performances outstanding.

Generic NLP Model

Model

- Random Forest Classifier
- 5,000 random training examples
- BOW of uni-grams, bi-grams and tri-grams.
- Use 500 features with highest TF-IDF scores.
- 1000 trees

Parameter to Tune

- max_features: continuous on $(0, 1]$
- max_depth: discrete in $\{1, 2, 3, 4, 5\}$
- min_samples_split: continuous in $(0, 0.5]$
- min_samples_leaf: continuous in $(0, 0.5]$

Approach 1: Grid Search

Grid Search Algorithm

- Try 3 values for each parameter ($3^4 = 81$ evaluation):
  - max_features: $\{0.1, 0.5, 0.9\}$
  - max_depth: $\{1, 3, 5\}$
  - min_samples_split: $\{0.1, 0.3, 0.5\}$
  - min_samples_leaf: $\{0.1, 0.3, 0.5\}$
- Choose values that give highest 5-fold CV score

Chosen Parameter Values

- max_features=0.1
- max_depth=1
- min_samples_split=0.1
- min_samples_leaf=0.1

Providing 68% Accuracy

Approach 1: Grid Search

Advantages

- ✓ Simple to understand and implement
- ✓ Can exploit prior knowledge of 'good' parameter values

Disadvantages

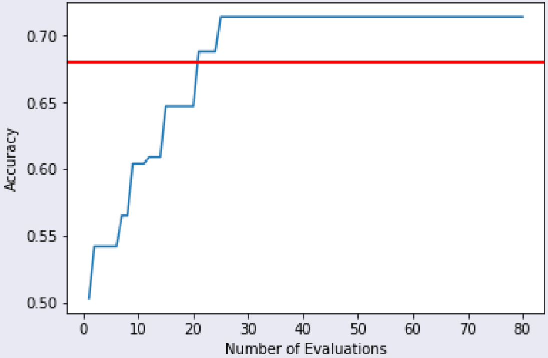
- × Naive exhaustive approach
- × Computational cost : n^d
- × Setting an effective grid requires prior knowledge

Approach 2: Random Search

Random Search Algorithm

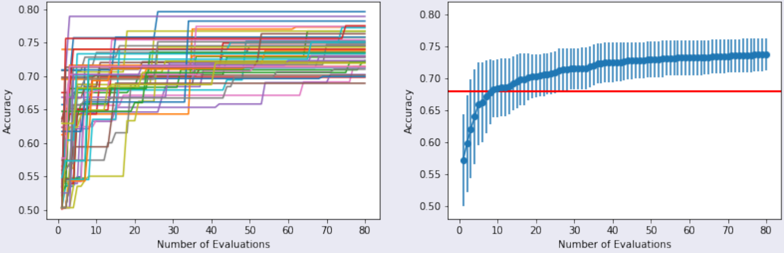
- Draw a parameter choice:
  - max_features uniformly from $(0, 1]$
  - max_depth uniformly over $[1, 2, 3, 4, 5]$
  - min_samples_split uniformly from $(0, 1]$
  - min_samples_leaf uniformly from $(0, 1]$
- Repeat n times
- Choose values that give highest 5-fold CV score

Approach 2: Random Search



Accuracy of the best found parameter values from random-search. The red line represents the performance found by grid-search.

Look at performance over 50 runs of random search



(a) (b)

Approach 2: Random Search

Advantages

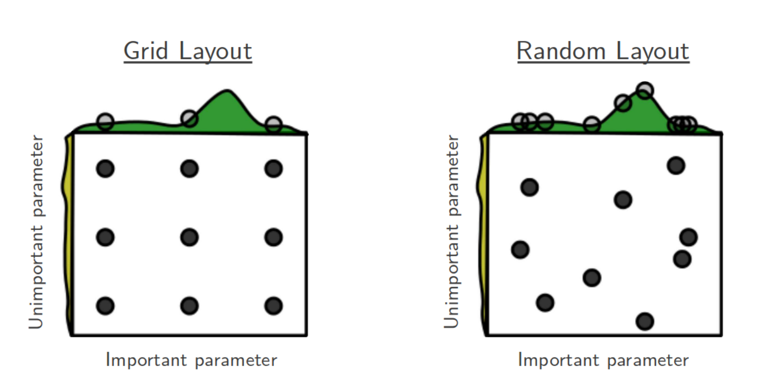
- ✓ Equally simple to understand and even easier to implement
- ✓ Adding non-important parameters doesn't effect performance
- ✓ Freedom to choose computational budget

Disadvantages

- × Harder to retain reproducibility
- × Small chance of not finding a 'good' solution
- × Could repeat evaluations of poorly performing parameter choices

Approach 2: Random Search

Intuition 1: Evaluate each parameter at more places [1]



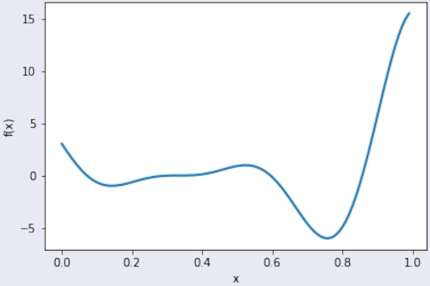
Approach 2: Random Search

Intuition 2: High probability of finding "good" parameter values[2]

- Consider the region R containing the optimum parameter choice and the surrounding 5% of parameter space.
- Each of n random evaluation has a 5% chance of being in R
- $p = prob(At\ least\ one\ point\ is\ in\ R) = 1 - (1 - 0.005)^n$
- For $n \geq 60$ $p > 0.95$

An Aside: Clever Searching for Optima[5]

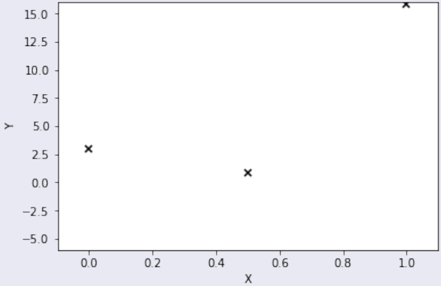
Consider Trying to find the minimum of $f(x) = (6x-2)^2 sin(12x-4)$



Using as few function evaluations as possible

An Aside: Clever Searching for Optima[5]

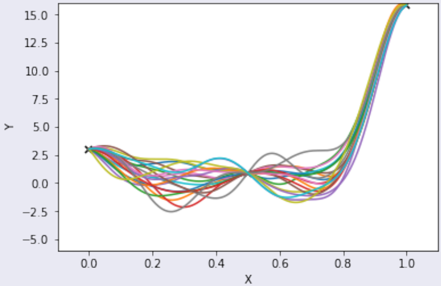
Suppose we make evaluations at 0,0.5 and 1



Where should we next evaluate?
What would grid or random search do?

An Aside: Clever Searching for Optima[5]

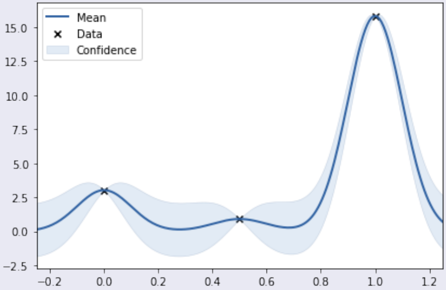
Possible functions that pass through the observed points



Where should we next evaluate?

An Aside: Clever Searching for Optima[5]

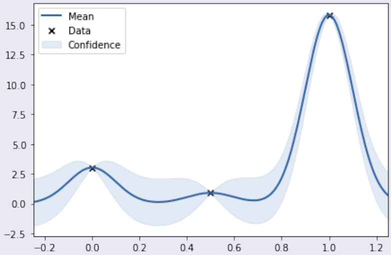
We can summarize this belief by fitting a Gaussian process



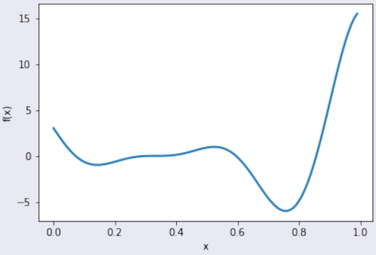
Where should we next evaluate?
Do we want to explore?
or exploit?

An Aside: Clever Searching for Optima[5]

Compare our statistical model with the truth



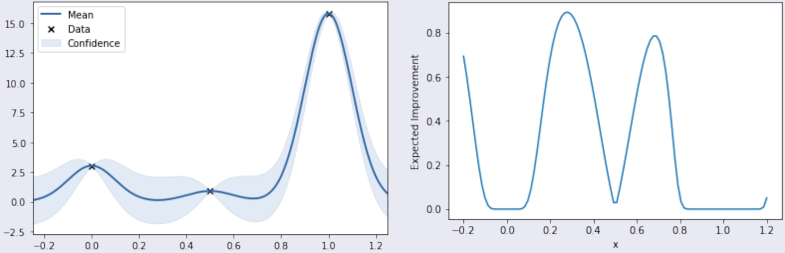
(c) Fitted Gaussian Process



(d) Truth

An Aside: Clever Searching for Optima[5]

We choose the next evaluation by maximizing an acquisition function

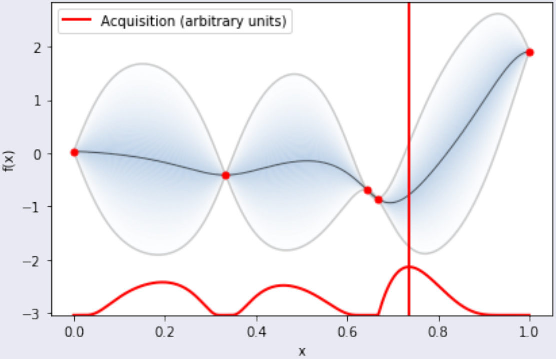


(e) Fitted Gaussian Process (f) Acquisition Function

This procedure is known as Bayesian Optimization (BO)

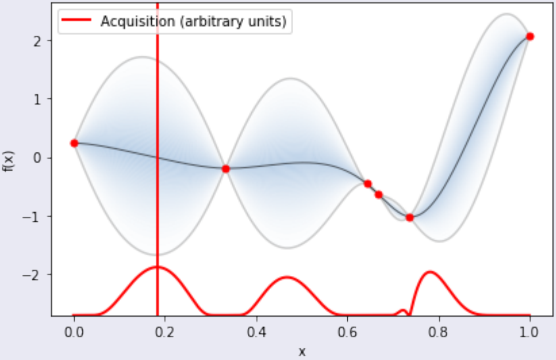
An Aside: Full BO implementation[6]

Model after 4 initial points and 1 evaluation chosen by BO



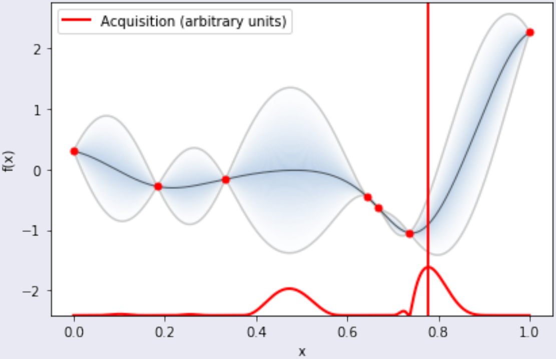
An Aside: Full BO implementation[6]

Model after 4 initial points and 2 evaluations chosen by BO



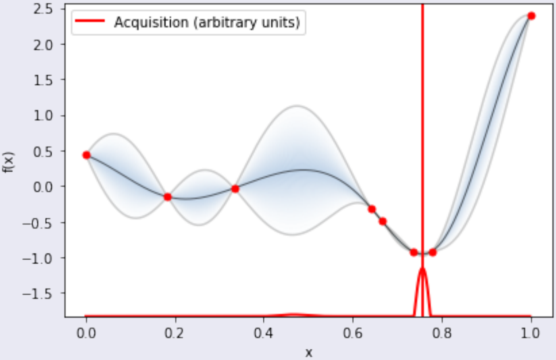
An Aside: Full BO implementation[6]

Model after 4 initial points and 3 evaluations chosen by BO

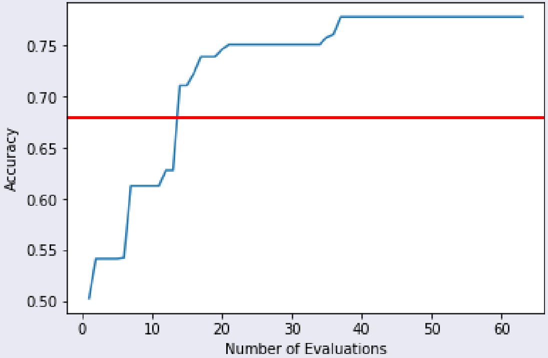


An Aside: Full BO implementation[6]

Model after 4 initial points and 4 evaluations chosen by BO

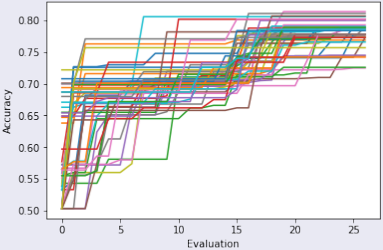


Approach 3: Bayesian Optimization (BO)

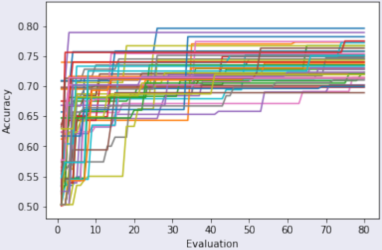


Accuracy of the best found parameter values from BO. The red line represents the performance found by grid-search.

Look at performance over 50 runs of BO



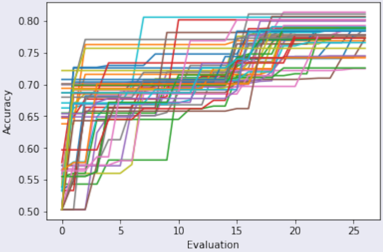
(g) BO for 25 evaluations



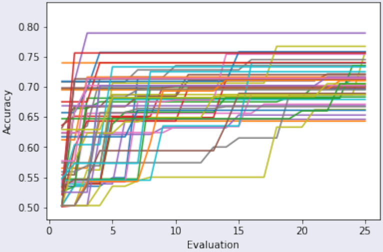
(h) Random-search for 81 evaluations

Approach 3: Bayesian Optimization (BO)

Look at performance over 50 runs of BO



(i) BO for 25 evaluations



(j) Random-search for 25 evaluations

Approach 3: Bayesian Optimization (BO)

Advantages

- ✓ Efficient
- ✓ Doesn't revisit bad parameter values
- ✓ Fully black-box

Disadvantages

- × More complicated
- × Computational cost grows cubically in n

Further Questions

- **Is 5-fold CV appropriate for effective tuning?**
  We say it often is not *https://arxiv.org/abs/1806.07139*
- **Can we adaptively choose how to partition our data as part of BO?**
  We think so, watch this space!

References

1. **Intuition 1:** A blog post by Alice Zheng *https://www.oreilly.com/ideas/evaluating-machine-learning-models/page/5/hyperparameter-tuning*
2. **Intuition 2:** Bergstra and Bengio, Random search for hyper-parameter optimization, Journal of Machine Learning Research, 2012.
3. **Bayesian Optimization Summary:** Snoek, Larochelle, and Adams, Practical bayesian optimization of machine learning algorithms, Advances in neural information processing systems, 2012.
4. **Gaussian Process Introduction:** Williams and Rasmussen, Gaussian processes for machine learning, MIT Press, 2006.
5. **Python Package for Gaussian Processes:** GPy, *https://sheffieldml.github.io/GPy/*
6. **Python Package for BO:** GPyOpt, *https://sheffieldml.github.io/GPyOpt/*