Performance of Machine Learning Algorithms for Model Selection

Xinnong Li (1), Mark Sale (2), Keith Nieforth (2), James Craig (2), Fenggong Wang (3), David Solit (4), Kairui Feng (3), Meng Hu (3), Robert Bies(1), Liang Zhao (3)
Institution: (1) University at Buffalo, (2) Certara, (3) U.S. FDA

Background & Objectives

The current standard method for population pharmacokinetic (PPK) model selection is forward addition/backward elimination (FABE). There has been little change of this approach for nearly 50 years, despite major advances in essentially all other numerical methods in pharmacometrics. The model selection process would be included in the disciple of "optimization". Specifically, FABE would be described as a "local search" or a "greedy search". In many ways FABE is comparable to gradient based methods of parameter optimization. Among the weaknesses of FABE are:

- Typically, the search starts at a "trivial" model, which is likely very different (i.e., a significant Euclidean distance in the search space) from the true optimal model, resulting in a risk of local minima.
- A local/greedy search is at risk for missing important interaction between model features, as typically only one feature at a time is examined.

In this work we examine the properties of several global search algorithms and compare to a "gold standard" of an exhaustive search.
In the ML algorithm paradigm, the set of hypotheses of interest (e.g., number or compartment, random effects, covariate relationships) comprises the "search space". The search space is an N-dimensional discrete space, where each dimension consists of a set of mutually exclusive options. Each candidate model consists of a set of exactly one option chosen from each dimension. The ML algorithm then searches this space, based on criteria defined by the user.

Methods

ML algorithms examined include:
GA – Genetic algorithm attempts to reproduce the mathematics of evolution and survival of the fittest.
GP – Gaussian process represents the fitness as a generalization of a Gaussian probability distribution. Samples are taken from that distribution selected to inform the parameters of the distribution. The models from those samples are then run, and the distribution is updated.
RF and GBRT – Random forest and gradient boosted random tree. Random forest implements a set of decision trees, classifying the search space based on the fitness values. Multiple trees from random bootstrap samples (samples, in this case, being the set of models and corresponding fitness) are used to prevent overfitting. In GBRT, rather than bootstrap samples to prevent overfitting, the tree is built additively, with each tree taking the previous, and "boosting" the gradient with respect to fitness by adding some selected low performance models.
PSO – Particle swarm optimization is an attempt to reproduce the behavior of a flock of birds or a school of fish. The "particles" represent candidate models and the movement of each particle in the search space is a weighted sum of random effects, movement toward the best model in the entire population and the best model in that particle history.
To provide a more robust search, the ML algorithms are supplemented by a 1- or 2-bit downhill search.

Methods (continued)

For the downhill search, the best ML models are used as the base model. All 1-bit (each bit flipped from 0 to 1 or 1 to 0) are produced. The 2-bit downhill search is similar, except all 2-bit changes are made. An example is shown in Table 1, assuming a base model of (0,1,0,0).

Table 1. Listing of 1- and 2-bit search genomes for a [0,1;0,0] genome

Change	1st bit	2nd bit	3rd bit	4th bit
1st bit	1,1;0,0			
2nd bit	1,0;0,0	0,0;0,0		
3rd bit	1,1;1,0	0,0;1,0	0,1;1,0	
4th bit	1,1;0,1	0,0;0,1	0,1;1,1	0,1;0,1

The fitness function was identical for all algorithms. The penalties listed in Table 2 were added to the -2LL output from NONMEM. No additional R/Python code penalties were used.

Table 2. Penalties

Description	Value
Penalty for each estimated THETA	10
Penalty for each estimated OMEGA element	10
Penalty for each estimated SIGMA element	10
Penalty for failing to converge	100
Penalty for failing the covariance step	100
Penalty for failing correlation test	100
Penalty for condition number > 1000	100

Results

The search space consists of 1,572,864 candidate models. The true optimal model was identified by the exhaustive search. This true optimal model included:

- Three compartments
- Power function of centered WT for volume
- BSV on central volume, clearance, peripheral volume 1, inter-compartment clearance 2
- Between occasion variability on central volume, clearance and intercompartment clearance 1
- Combined additive and proportional residual error model

The parameter values for the final model are in Table 3

Table 3. Parameter values of the optimal model

Parameter	Estimates (SE%)	BSV (SE%)	IOV(SE%)
CL (L/h)	8.54 (7)	50.3% (11)	30.3% (13)
Q2 (L/h)	79.3 (6)	-	28.6% (13)
Q3 (L/h)	9.74 (13)	75.8% (10)	-
V1 (L)	29.1 (8)	42.9% (25)	31.1% (35)
V2 (L)	71.9 (8)	58.7% (11)	-
V3 (L)	136 (12)	-	-
V~WT	1.31 (26)	-	-
Proportional error	13.2 (7)	-	-
Additive error (mg/L)	14.7 (39)	-	-

The final ML model can be compared to the final model found in the original analysis [2]. Note that this comparison is limited in that the criteria for model selection were different. The final original analysis model was three compartments, linear with no covariates. Between subject variability was included on clearance, Q3, and all volumes. Between occasion variability was described on central volume and Q2. In comparison, the final model from this exercise was also three compartments, linear, with between subject variability on CL, Q2 and all volumes. One covariate was described, with volume of distribution a power function of weight. The objective function value (OFV) of the original model was 8178.5, compared to the current analysis OFV of 8041.3 (delta OFV = 137.2 points), with 1 additional THETA, 1 additional OMEGA, and 1 additional SIGMA parameters.

Results (continued)

Results for all algorithms are shown in Table 4. Note that the OFV for all algorithms with 1- and 2-bit downhill search are the same (2nd column), indicating that they identified the true optimal model identified by the exhaustive search. All algorithms except PSO identified the optimal model with only a 1-bit local search (right most column). This suggests a robust model search for all but PSO. The time to best model (column 3) suggests that GA, GP, and RF are similar, although GP was notable slower to complete the search (2nd column from the right).

Table 4. Performance of algorithms

Algorithm	Bes fitness with 1- and 2-bit downhill search	Best OFV with 1- and 2-bit downhill search	Time to best model with 1 and 2-bit downhill search	Unique models to the best model with 1- and 2-bit downhill search	Time with 1- and 2-bit local downhill search	Best fitness with only 1 bit downhill search
Exhaustive Search	8201.3	8041.3	34.2	97706	60384	n/a
GA	8201.3	8041.3	169.3	1307	321.8	8201.3
RF	8201.3	8041.3	126.7	880	461.6	8201.3
GP	8201.3	8041.3	114.6	495	2976	8201.3
PSO	8201.3	8041.3	255.5	1710	404.2	8220.7
GBRT	8201.3	8041.3	223.8	1328	410.2	8201.3

Discussion

All ML algorithms implemented in pyDarwin package were able to find the optimal model structure searched by "gold standard" exhaustive search, which highlights the robustness of the machine learning application in nonlinear mixed effect model selection and optimization process. It's noted that this analysis is based on a single sample model selection process, therefore any inference on the robustness of the algorithm's performance across various environments is limited. To further evaluate the robustness of each ML algorithm, more datasets and corresponding exhaustive search results will be needed.

The most efficient algorithm measured by the number of unique models to the optimal is GP, which was able to find the optimal model after examining 495 models. However, it's the least efficient algorithm based on the total elapsed time, which is up to 2975.6 min. This analysis is based on a single, typical dataset, and changing the datasets, using different tokens, and adjusting the hyperparameters may impact the algorithm efficiency. We do, though based again on small sample size as well as other experience with this method, conclude that the two-bit local downhill search is likely critical to ensure robustness and should be done whenever the computation load is feasible.

With regard to the performance of the original analysis, we can conclude that, based on the search criteria used here, it was not robust. However, we are unable to compare which, if any subjective criteria entered the original model selection and so a conclusion regarding robustness of the original analysis cannot be made.

PyDarwin is open source, available on github [1].

This work was supported by FDA/NIH grant (U01FD007355) (Development of a model selection method for population pharmacokinetics analysis by deep-learning based reinforcement learning (RFA-FD-21-027)). Views expressed in this poster do not represent FDA's views or policy.

[1] https://certara.github.io/pyDarwin/html/index.html
[2] Banerji U, et al. Phase I pharmacokinetic and pharmacodynamic study of 17-allylamino, 17-demethoxygeldanamycin in patients with advanced malignancies. J Clin Oncol. 2005;23:4152–4161. doi: 10.1200/JCO.2005.00.612.

Want to learn more?
<< Scan Here