Model selection

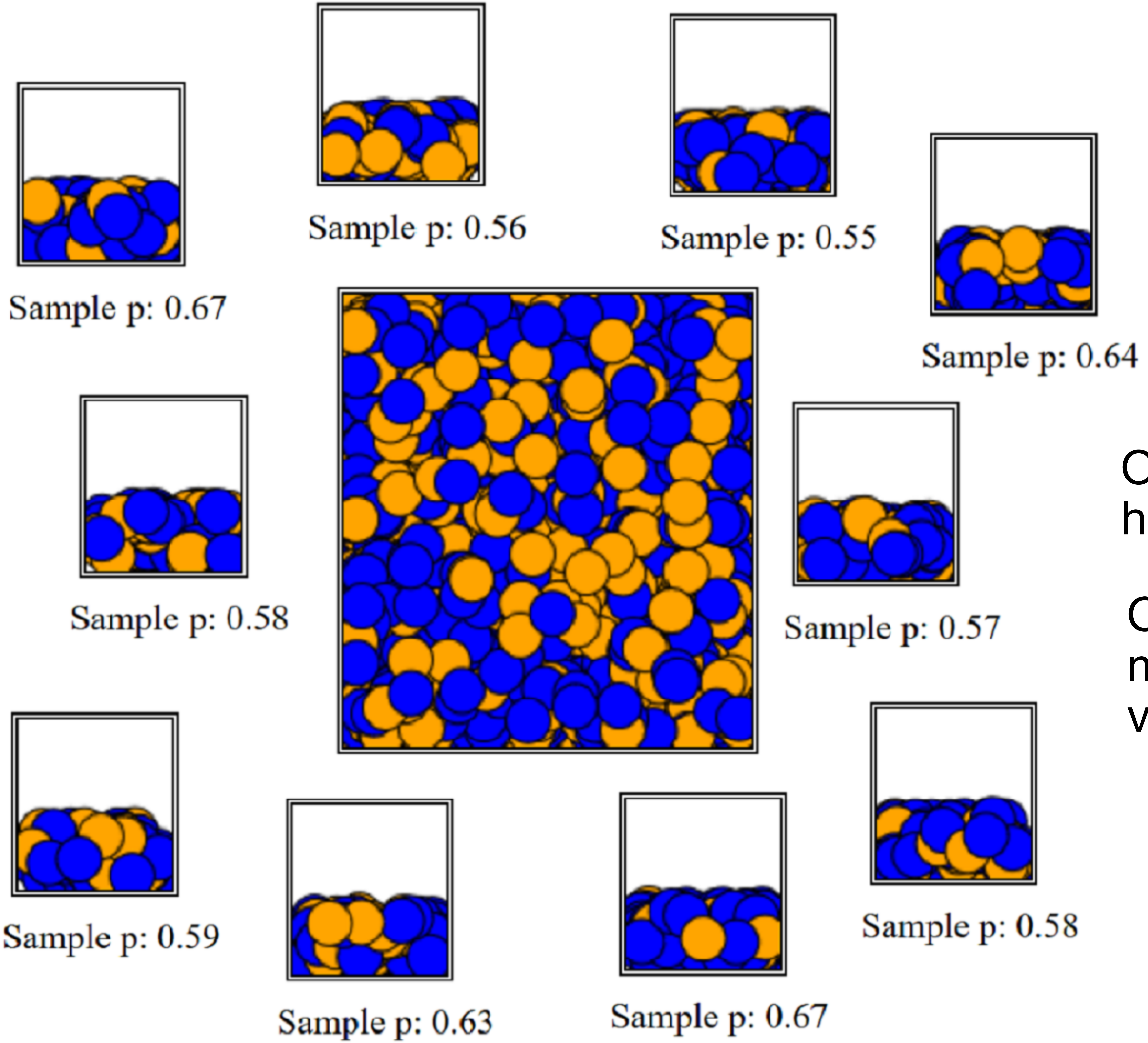
Jason G. Fleischer, Ph.D.
Asst. Teaching Professor
Department of Cognitive Science, UC San Diego

jfleischer@ucsd.edu

@jasongfleischer

https://jgfleischer.com

Slides in this presentation are from material kindly provided by Sebastian Rashka



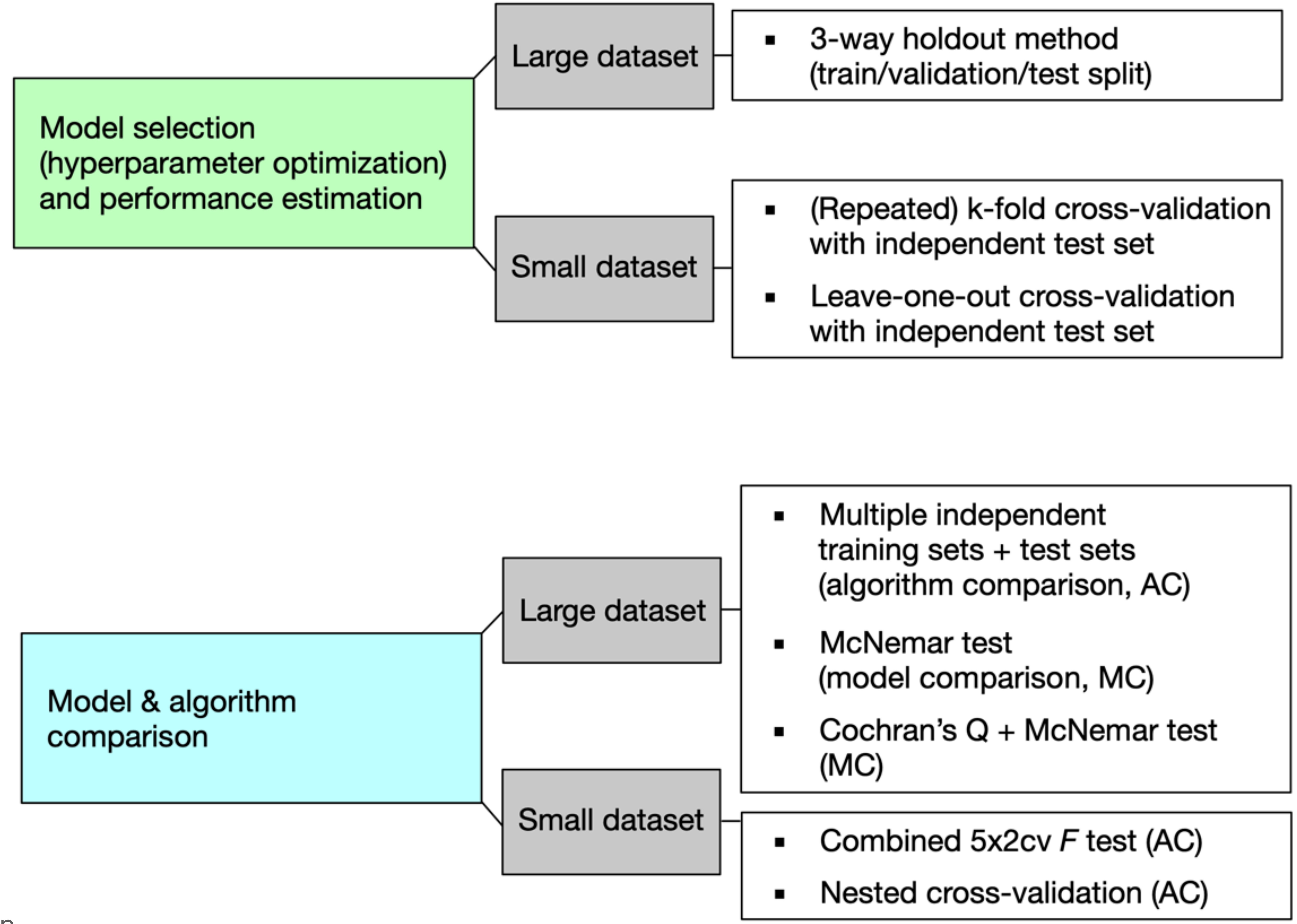
One of the hold out folds

Or one trial of model selection via cross-val

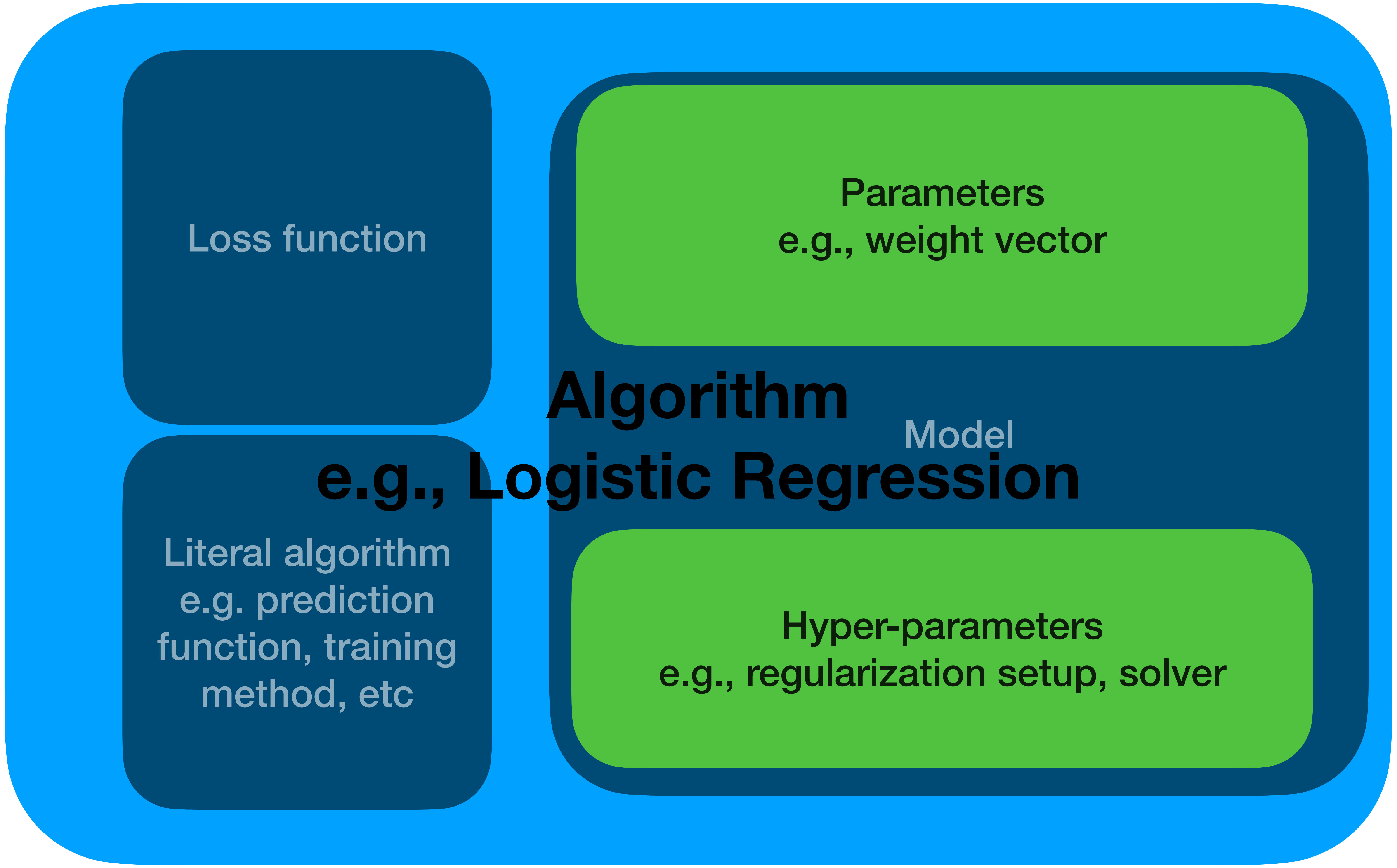
Estimation of performance

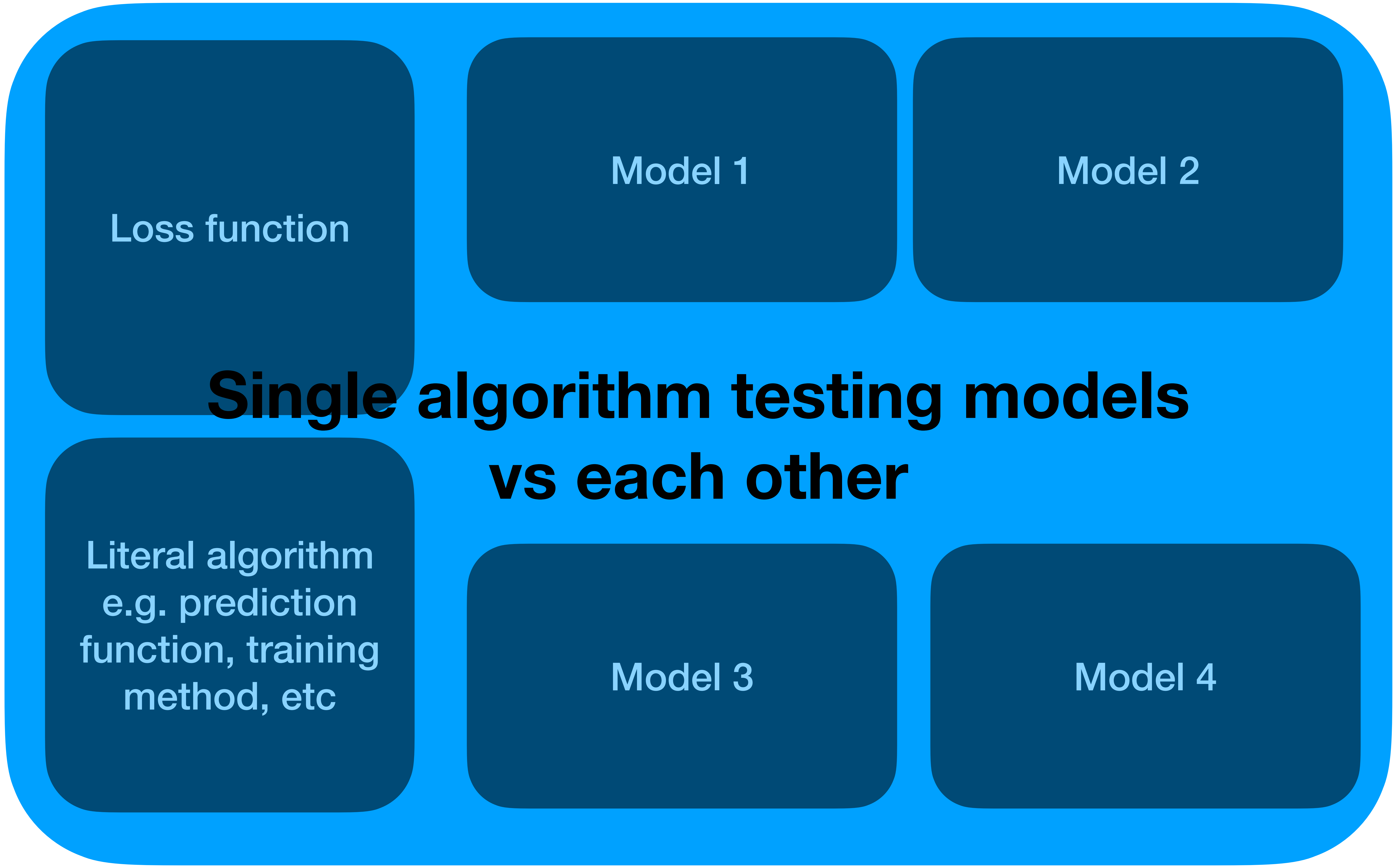
Many methods, two use cases, one reason

- THE ONE REASON: every measure is a random draw from a distribution of performances… What if the data was a bit different? What if the random seed is different? Etc.
- TWO USE CASES:
  - To estimate how well the system will generalize (test)
  - To perform model selection or algorithm selection (validation)
- MANY METHODS:
  - See Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning Sebatian Rashkha https://arxiv.org/pdf/1811.12808.pdf for a good intro (but there are more out there!)

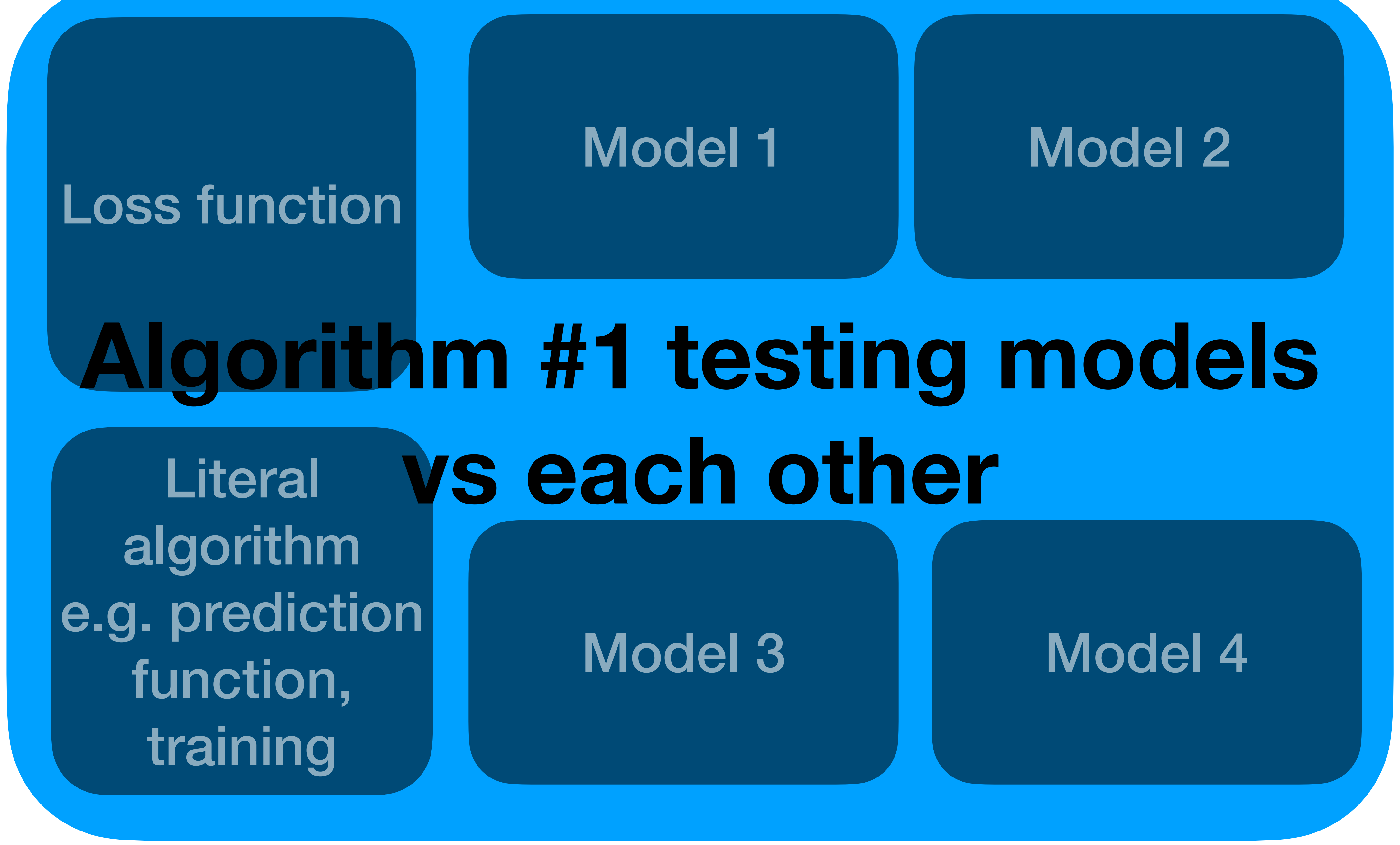


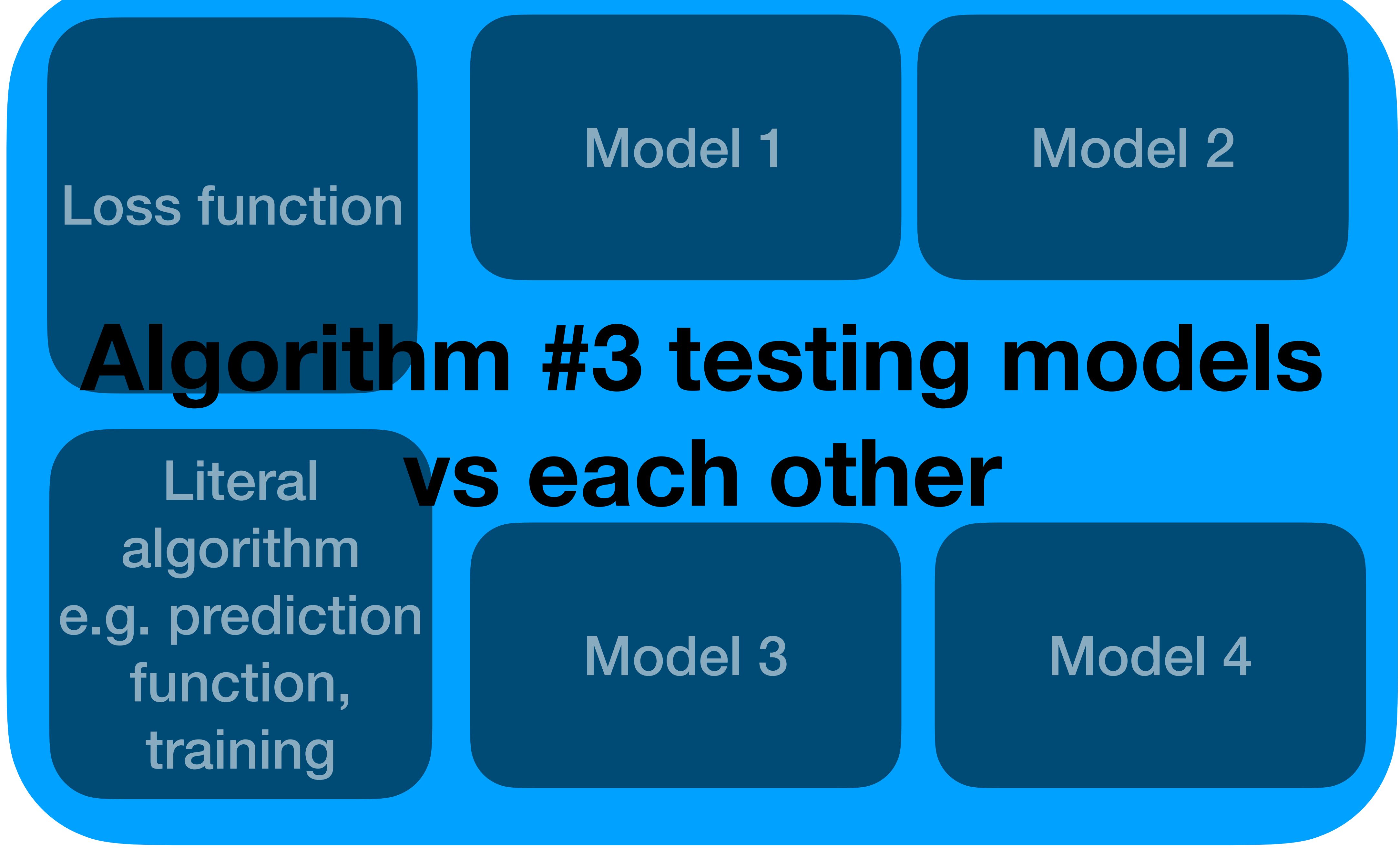
Model Evaluation, Model Selection,
and Algorithm Selection in Machine Learning
Sebatian Rashkha
https://arxiv.org/pdf/1811.12808.pdf


Loss function
Parameters
e.g., weight vector
Algorithm
e.g., Logistic Regression
Model
Literal algorithm
e.g. prediction
function, training
method, etc
Hyper-parameters
e.g., regularization setup, solver

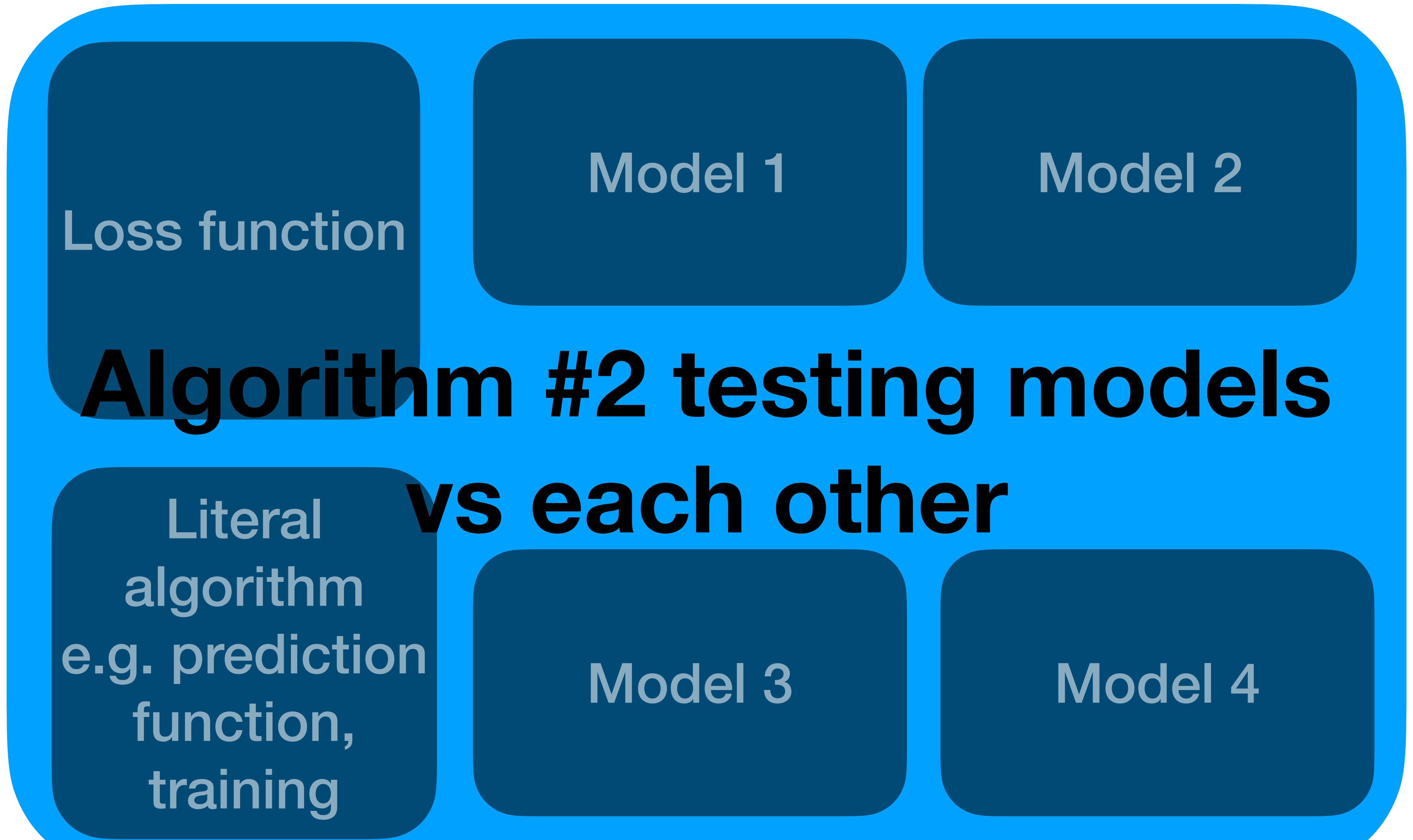


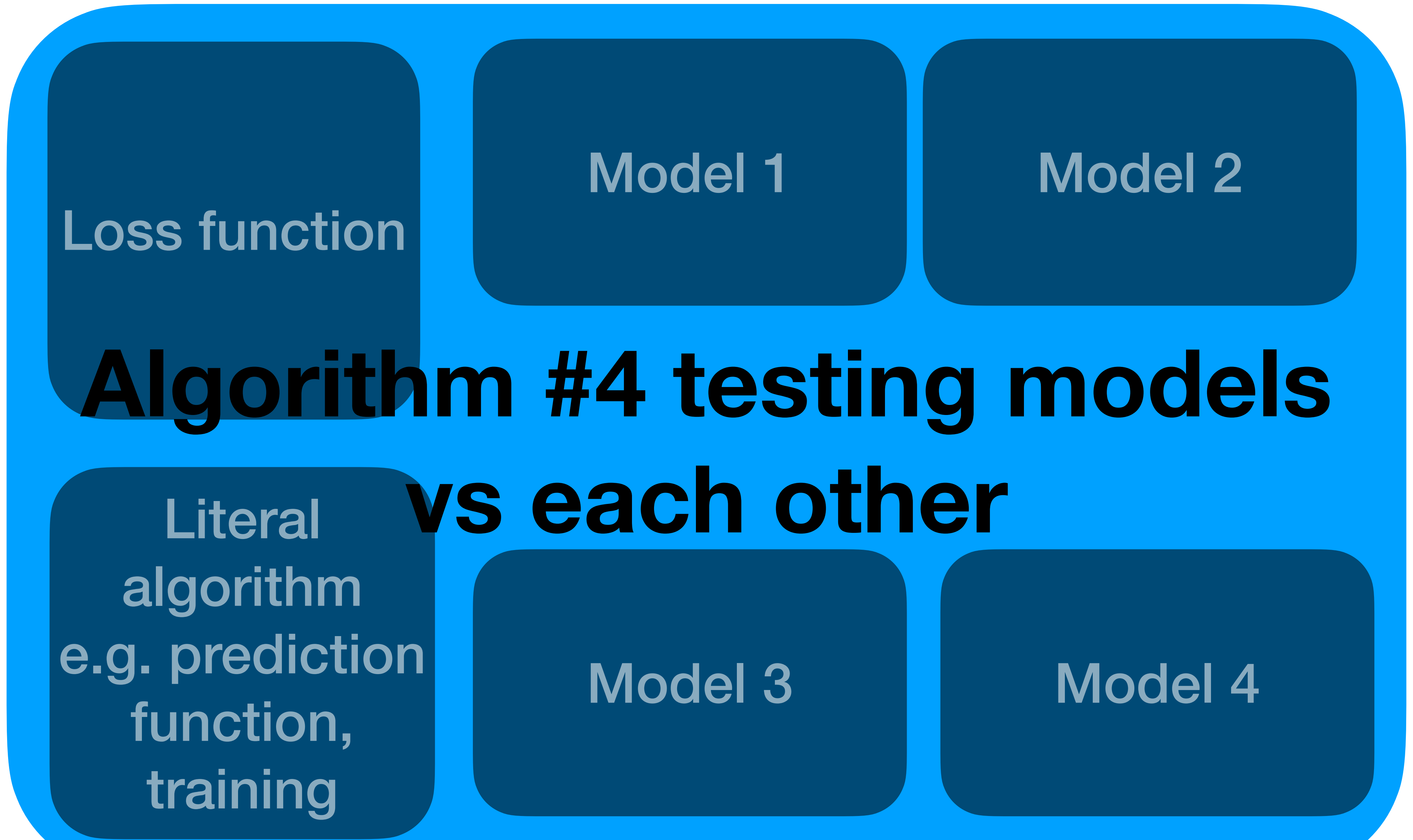
“Model selection”





“Algorithm selection”





Method #1 - Train/Validate/Test sets

For either Model or Algorithm selection using HUGE datasets

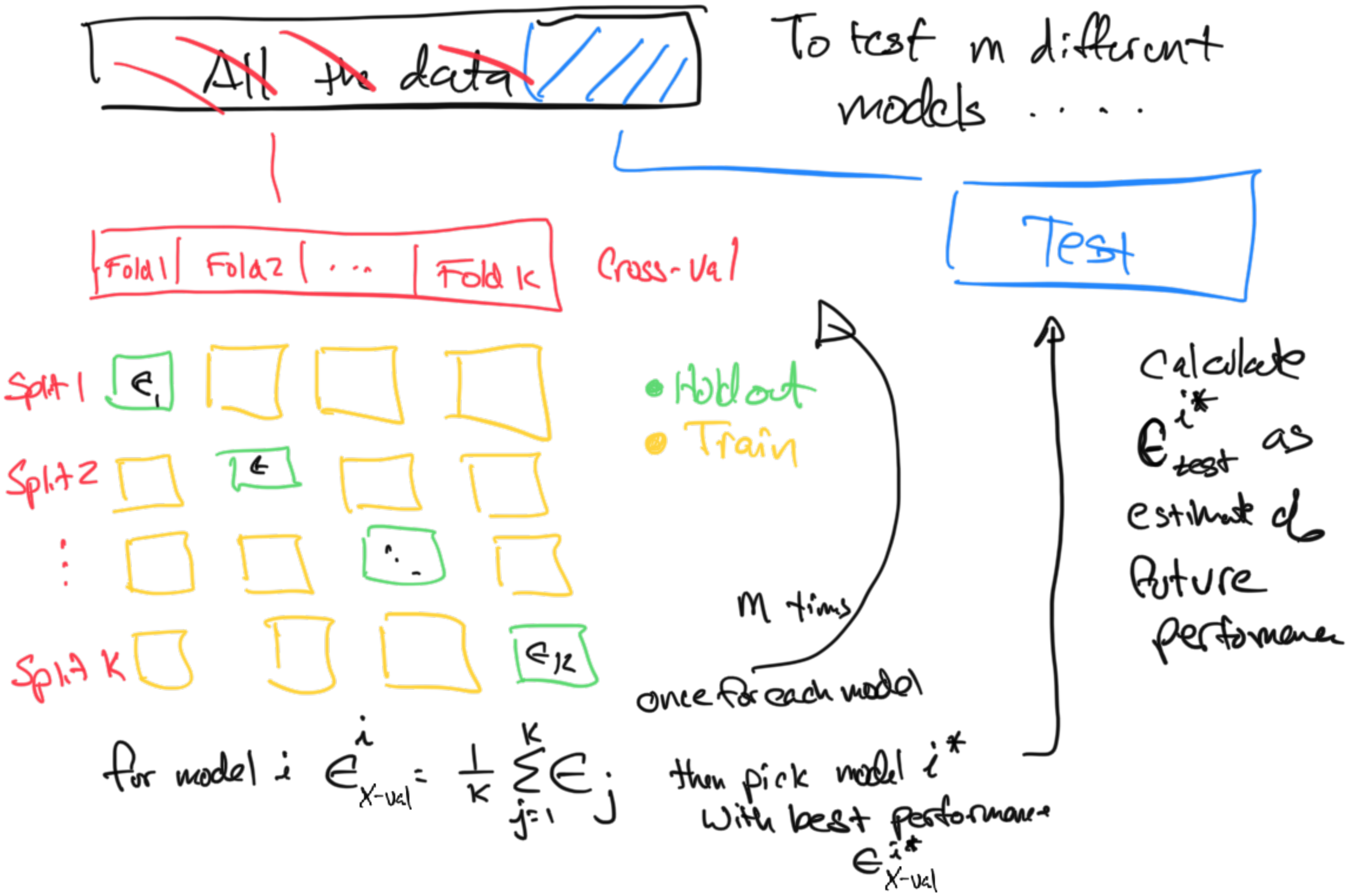
- Split data into train, validate, test
- [OPTIONAL] Outer loop… do this T times:
  - do this M times, once for each model in the hyper-parameter search space or each algorithm-model combination:
    - Train it on the same training set
    - Predict on the same validation set
- Pick the best model or algorithm based on its performance on [OPTIONAL the mean across trials] of the validation set
- Train the best version on the whole of training set + validation set
- Test it on the test set to estimate its ability to generalize

METHOD 1 - with enough data to have a good test set

Let's say you had around 8k samples in a dataset

For each trial:

- training set ~ sample 5k (with or w/o) replacement from entire dataset
- Grid search of hyper parameters using k-fold cross validation on the training set
- Select best model from grid, train on entire training set
- Evaluate best model on the test set (everything not sampled for training)



Method #2 - Cross validation

For either model or algorithm selection using medium sized datasets

- Split data into cross-validation and test sets
- [OPTIONAL] Outer loop… do this T times:
  - do this M times, once for each model in the hyper-parameter search space or each algorithm-model combination:
    - Use k-fold cross validation to estimate validation error
- Pick the best model or algorithm based on its performance on [OPTIONAL the mean across trials] of the validation sets
- Train the best version on the whole of cross validation set
- Test it on the test set to estimate its ability to generalize

METHOD 2 - make the most of a small amount of data

Nested Cross-validation

For Algorithm Comparison if done the time efficient way… (only best hyperparams tested on the outer cross-val)

…can be used for Model comparison if done the inefficient way (all hyper params tested on the outer cross-val)

This for when you've got only ~2000 samples, which is barely enough to fit the data well let alone test

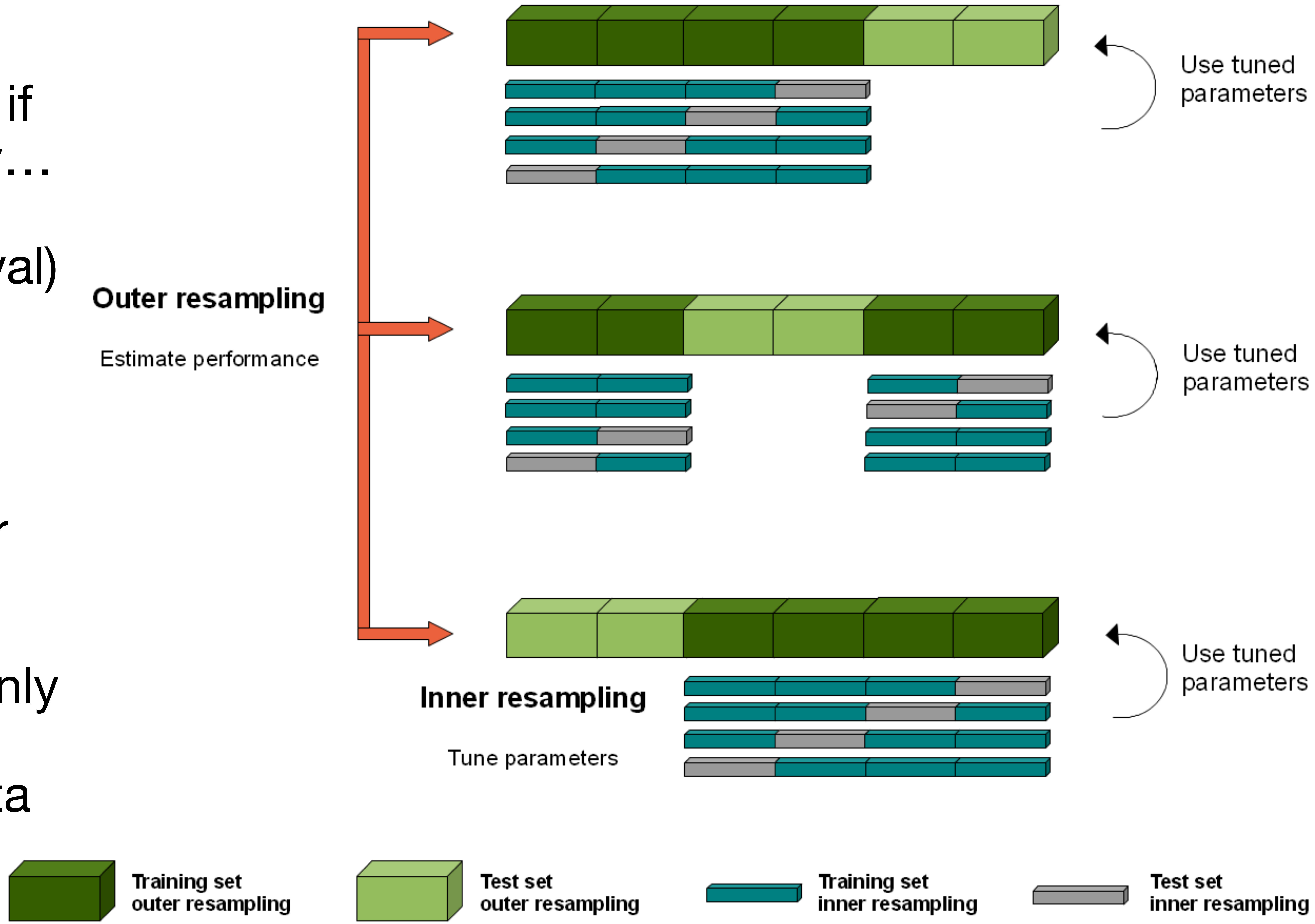


Figure from mlr R package docs

Method #2a - Nested CV for al

For doing algorithm selection on medium sized datasets

- Do not split off a test set!
- [OPTIONAL] Outer loop… do this T times:
  - Do this M times, once for each algorithm
    - Use nested k-fold cross validation…
      - Inner loop estimates validation error for all the hyperparams tested for a given model
      - Outer loop estimates validation error for a given algorithm
- Pick the best algorithm based on its performance on [OPTIONAL the mean across trials] of the outer cross validation folds

Model selection with built-in test set error using nested CV

Method #2a - Small sized datasets

- Do not split off a test set!
- [OPTIONAL] Outer loop… do this T times:
  - Do this M times, once for each algorithm
    - Use nested k-fold cross validation…
      - Inner loop estimates validation error for all the hyperparams tested for a given model
      - Outer loop estimates validation error for a given algorithm
- Pick the best algorithm based on its performance on [OPTIONAL the mean across trials] of the outer cross validation folds

METHOD 2 - make the most of a small amount of data

Nested Cross-validation

For Algorithm Comparison if done the time efficient way… (only best hyperparams tested on the outer cross-val)

…can be used for Model comparison if done the inefficient way (all hyper params tested on the outer cross-val)

This for when you've got only ~2000 samples, which is barely enough to fit the data well let alone test

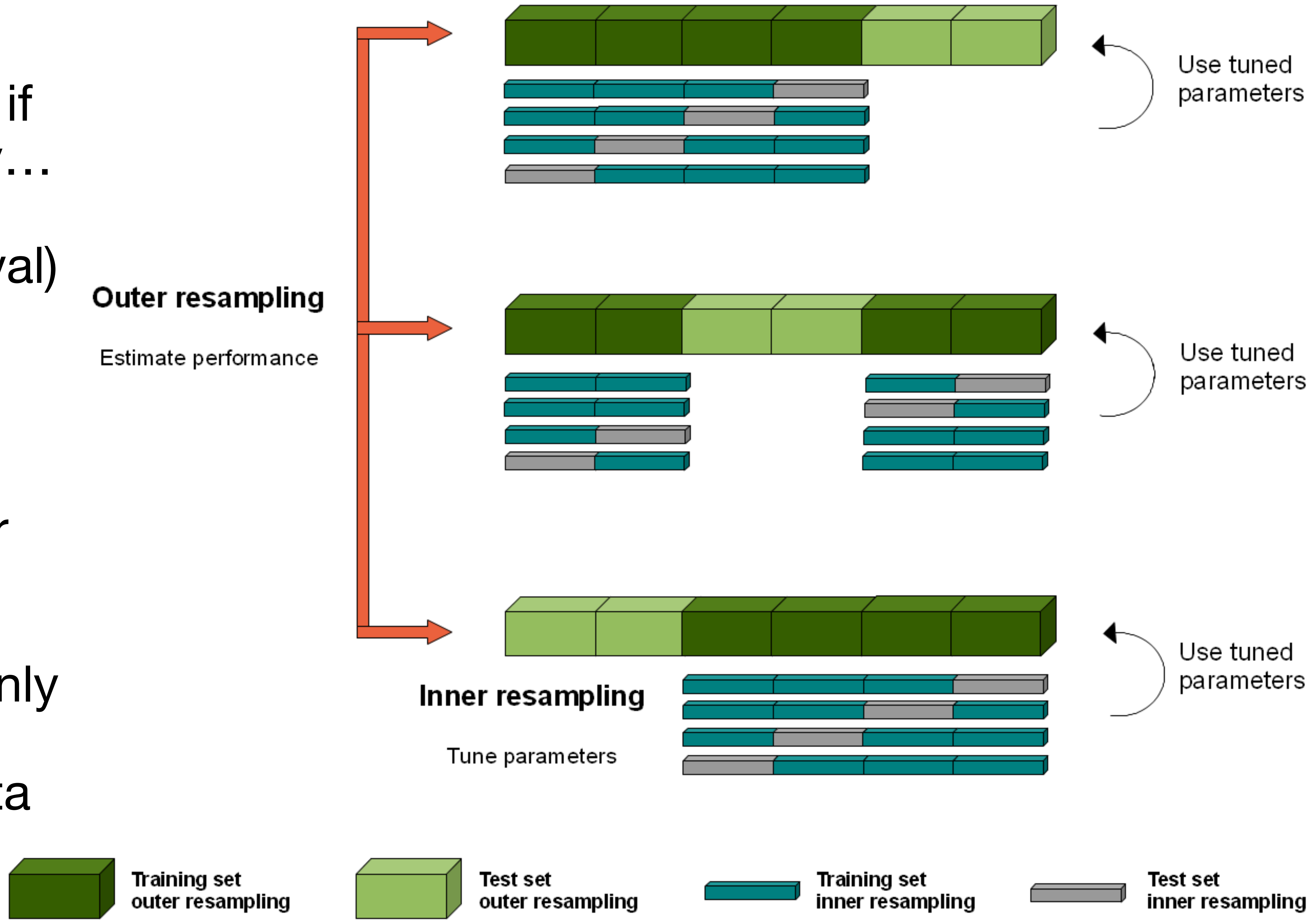
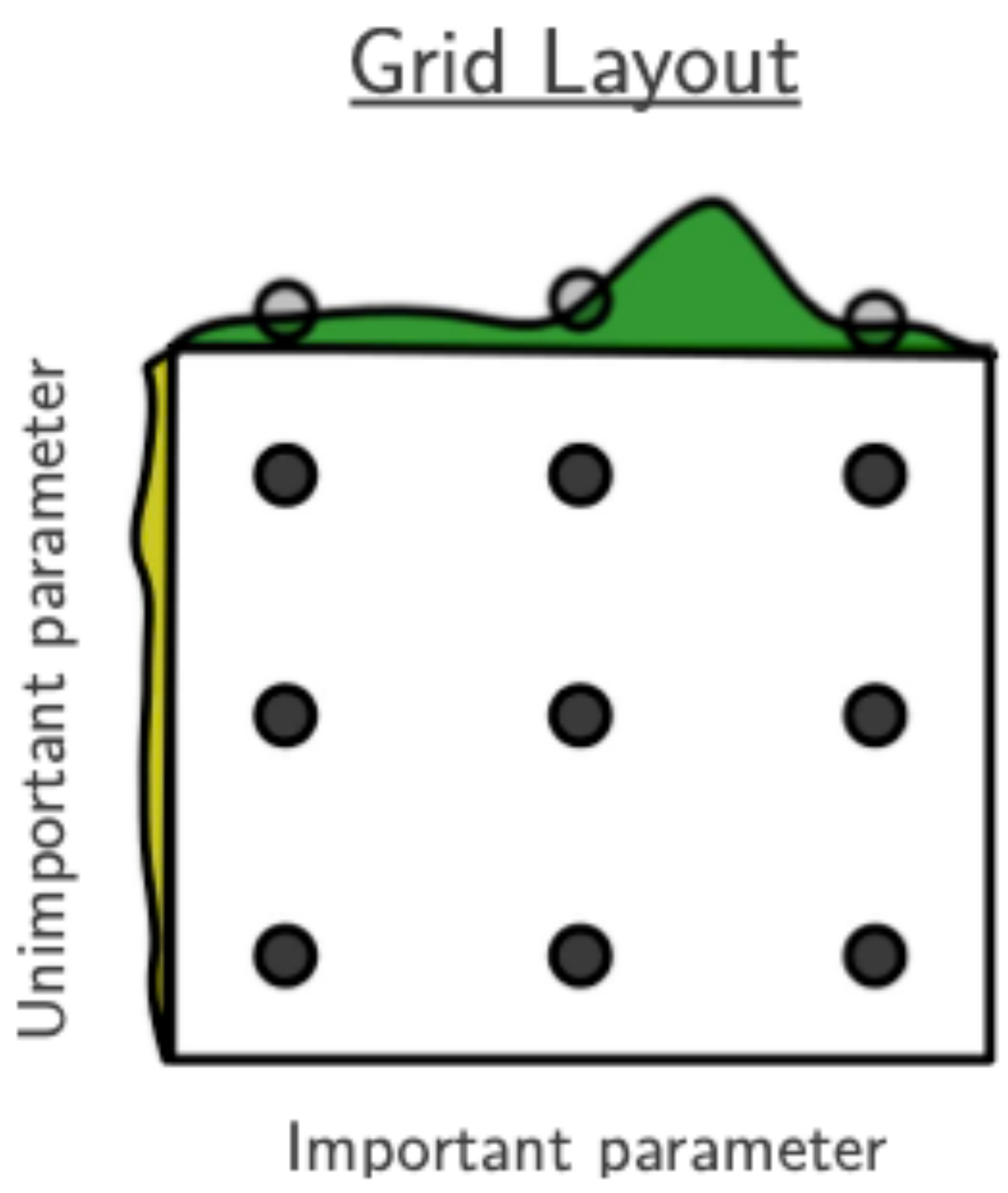


Figure from mlr R package docs

But how do you organize your search of the hyper parameter space?

Grid Search

- Exhaustive search
- Thorough but expensive
- Specify grid for parameter search
- Can be run in parallel
- Can suffer from poor coverage
- Often run with multiple resolutions



Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *The Journal of Machine Learning Research*, *13*(1), 281-305.

Randomized Search

- Search based on a time budget
- Preferred if there are many hyperparameters (e.g. > 3 distinct ones)
- specify distribution for parameter search
- can be run in parallel

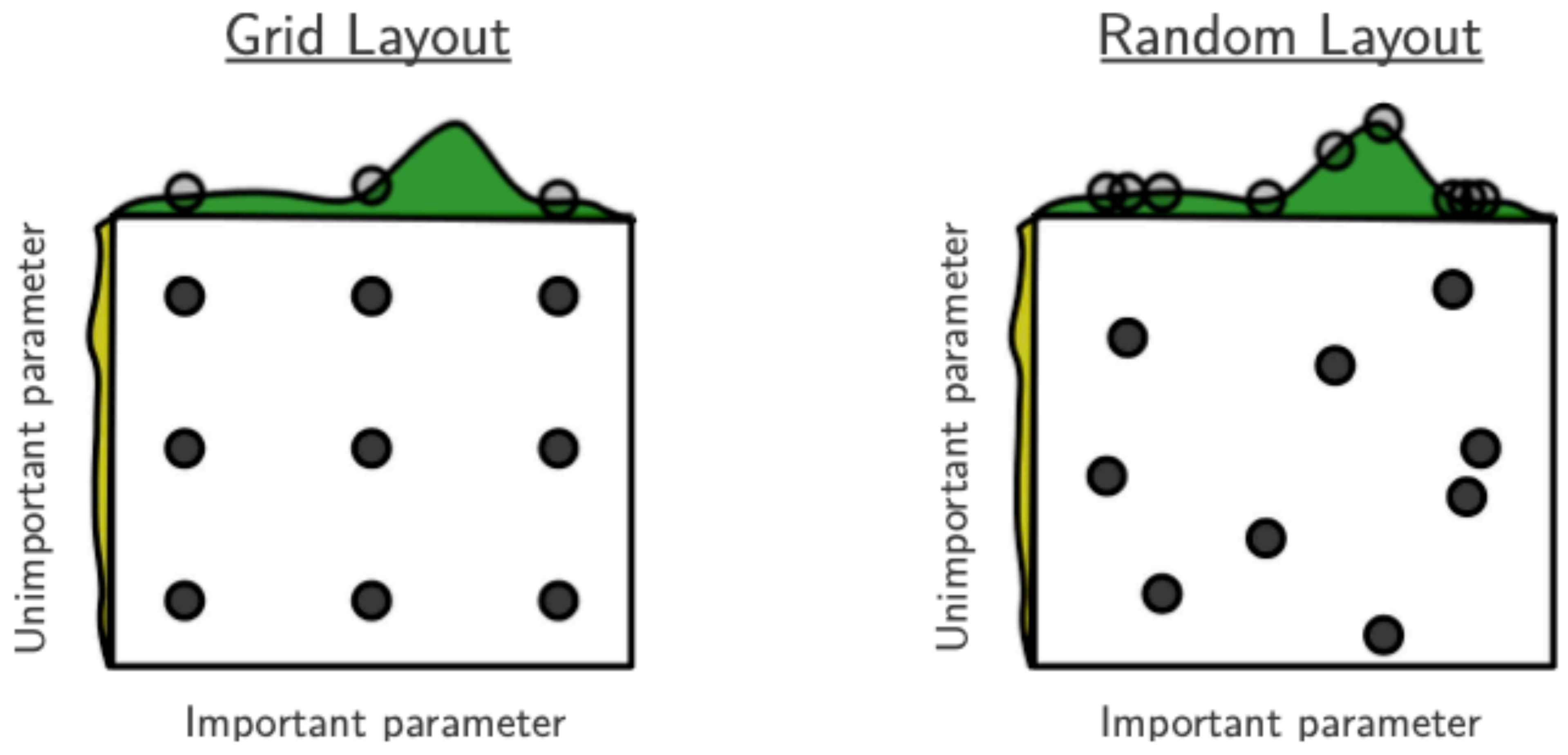


Figure 1: Grid and random search of nine trials for optimizing a function $f(x,y) = g(x) + h(y) \approx g(x)$ with low effective dimensionality. Above each square $g(x)$ is shown in green, and left of each square $h(y)$ is shown in yellow. With grid search, nine trials only test $g(x)$ in three distinct places. With random search, all nine trials explore distinct values of g. This failure of grid search is the rule rather than the exception in high dimensional hyper-parameter optimization.

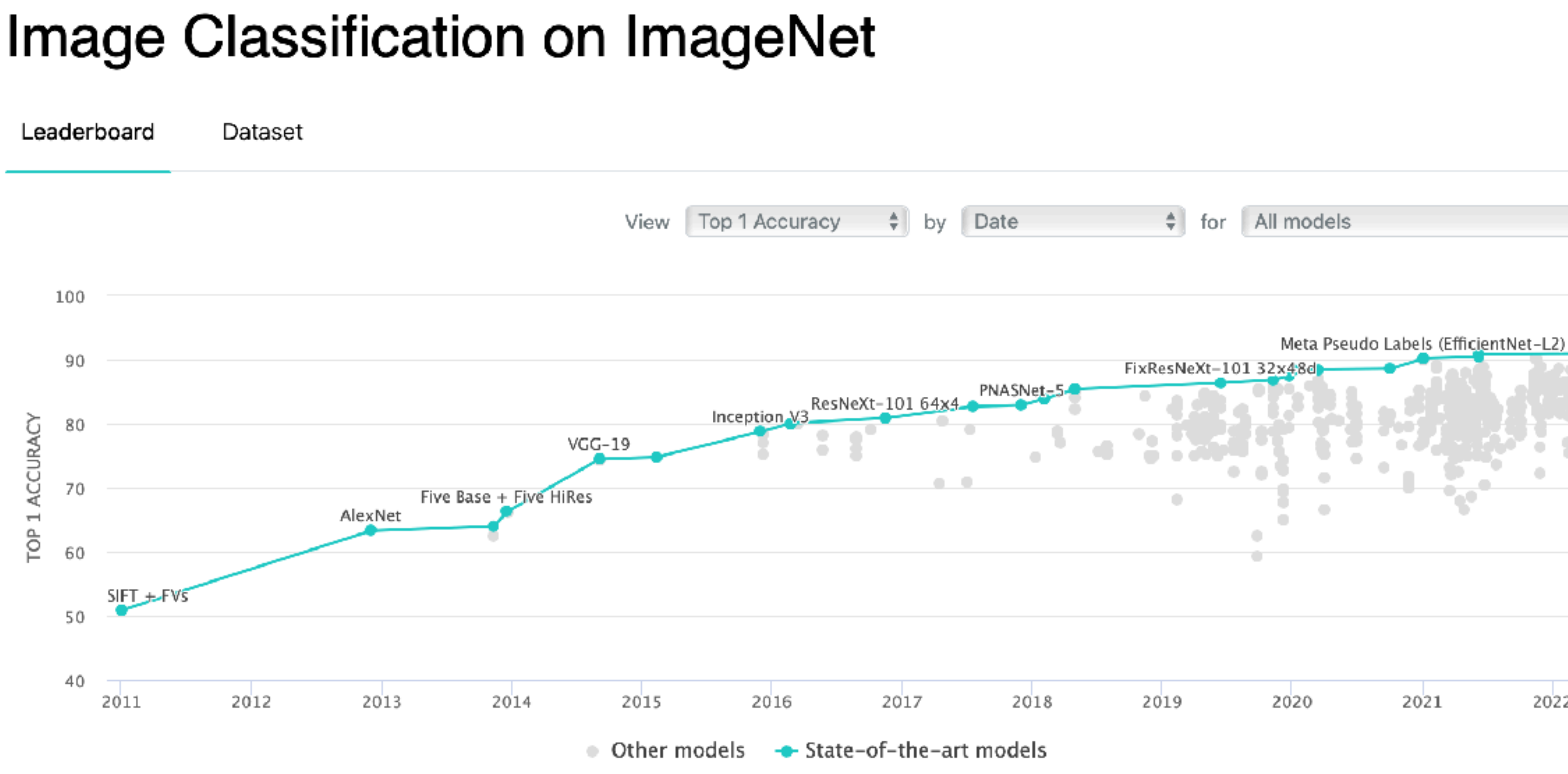
Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *The Journal of Machine Learning Research*, *13*(1), 281-305.

Statistical testing

https://sebastianraschka.com/pdf/lecture-notes/stat479fs18/11_eval-algo_notes.pdf

Statistical testing on model performance

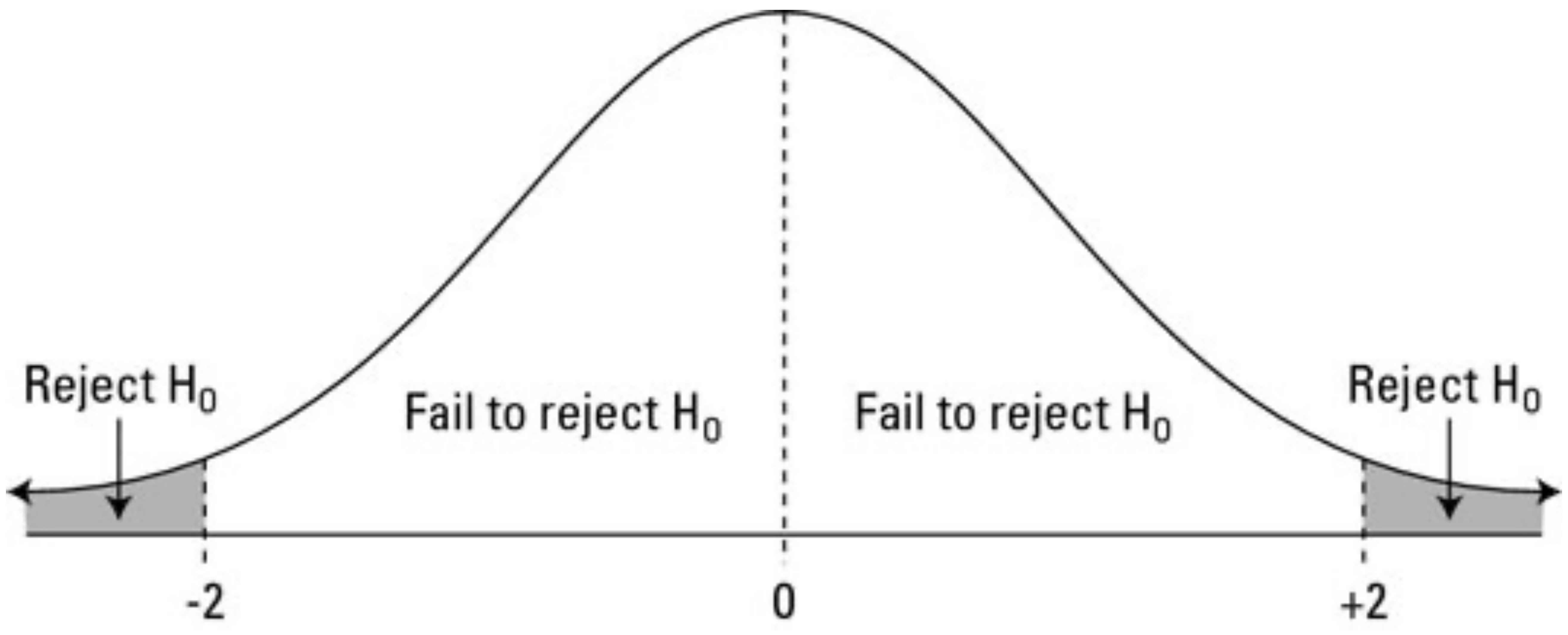
- Testing is almost always paired (over folds of cross validation)
- Distinguish between tests appropriate for algorithm comparison vs model selection (hyperparameter settings)
- Distinguish between test that are computationally efficient vs those that are not
- Distinguish between pair-wise and group-wise tests



Jason gets grumpy about blindly following methods you don’t understand fully

The p-value

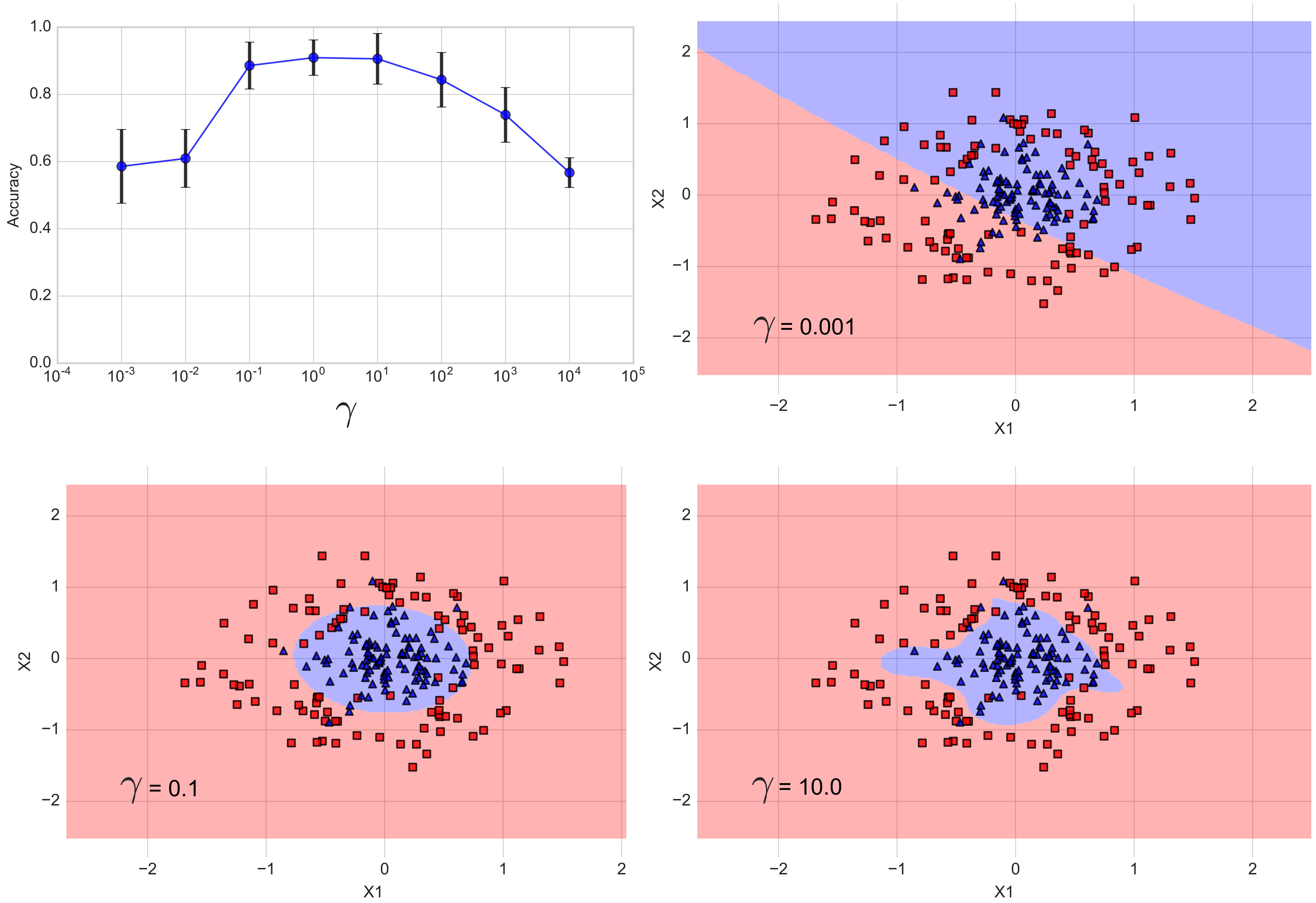
- In range 0,1
- Smaller is support for alternative hypothesis
- Larger is inconclusive
- Ignores effect size!!!@!!! Is the difference practically important?
- Assumes conditions on data
- $P(H|D) = \frac{P(D|H)P(H)}{P(D)}$



Parsimony Principle

Choose the simplest w/in 1 std error of optimal

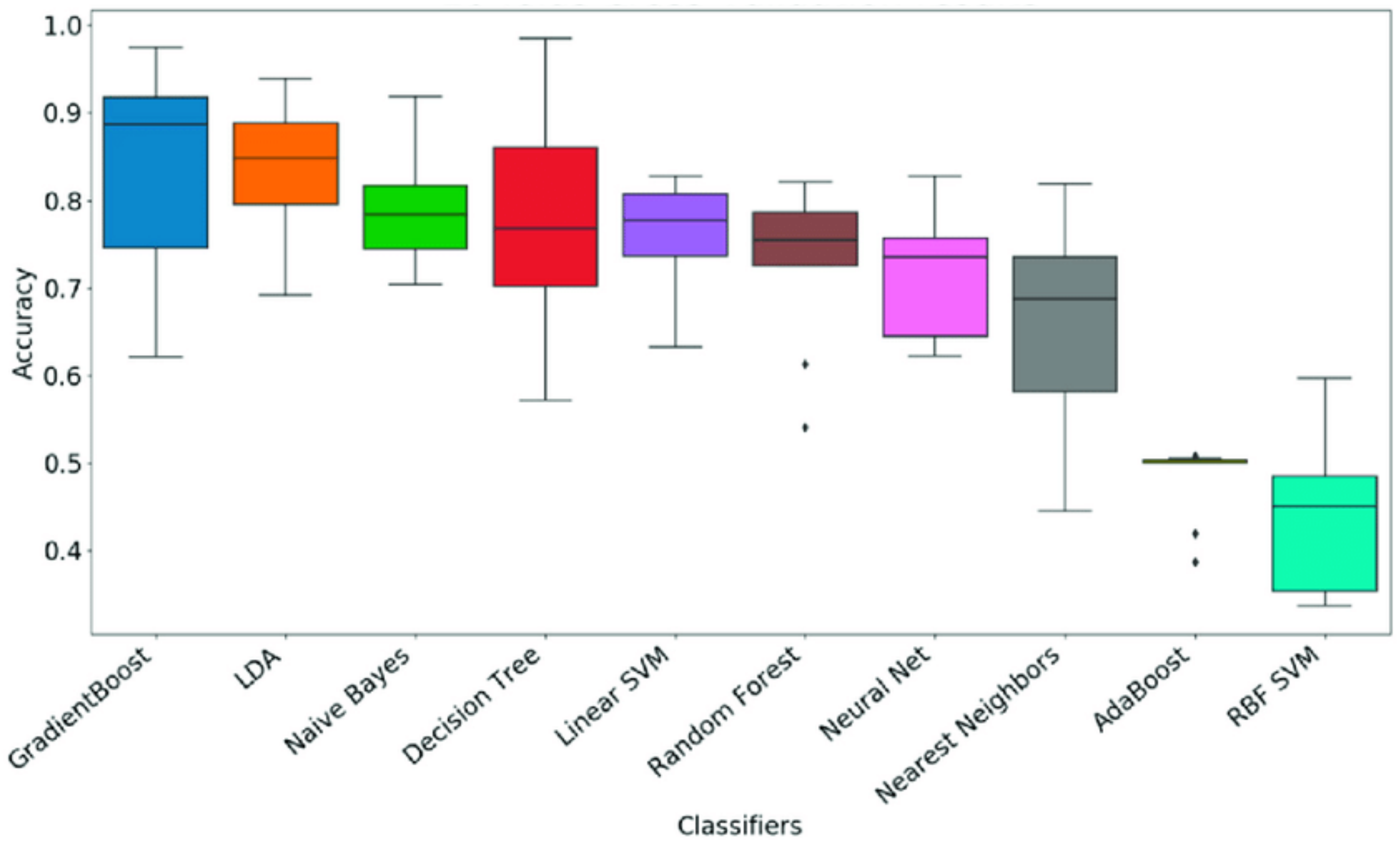
Which parameter would you select?



Sebastian Rashka

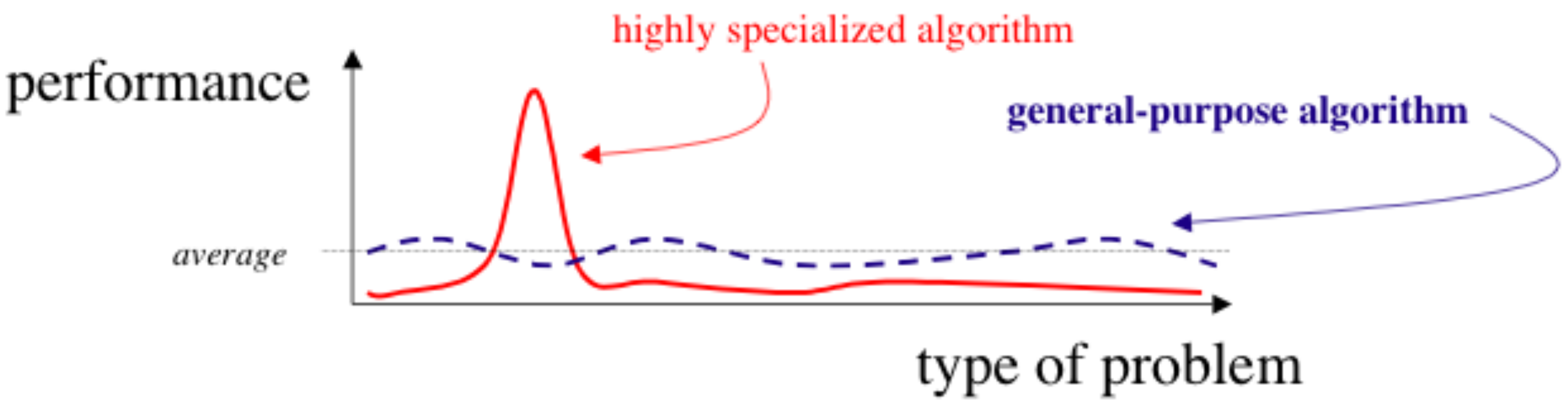
Maybe you don’t need a statistical test

3x5 repeated cross validation results



No free lunch theorem

Why even bother??



Wolpert, D. H.; Macready, W. G. (1995). "No Free Lunch Theorems for Search" (PDF). *Technical Report SFI-TR-95-02-010*. Santa Fe Institute.

http://www.no-free-lunch.org