Jhelum Chakravorty

Published Aug 12
Fork of Cross Validation and Model Selection by Kris Sankaran

Cross Validation and Model Selection

IFT6758, Fall 2020

Reading: ISLR section 5.1 and PDS pg. 359 - 375

Cross Validation and Model Selection

Daily Choices for Data Scientists

Knowing how to fit models is not enough, if you want to solve a real-world problem.

- How should you select between model families?

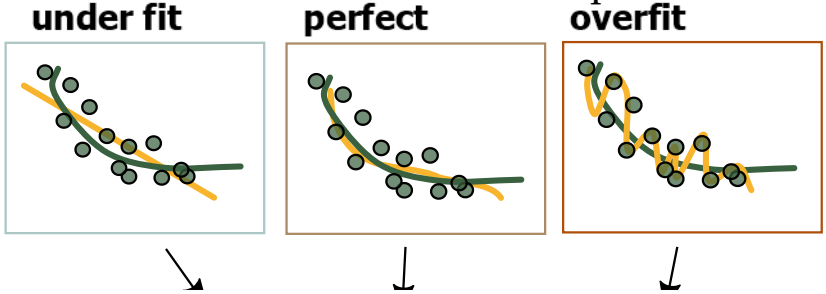
- How should you select between model families?
- Which parameters are best within a model family?
- Should you be trying to improve the data?
  - More samples? Richer features?
  - Fewer missing data, outliers, ...

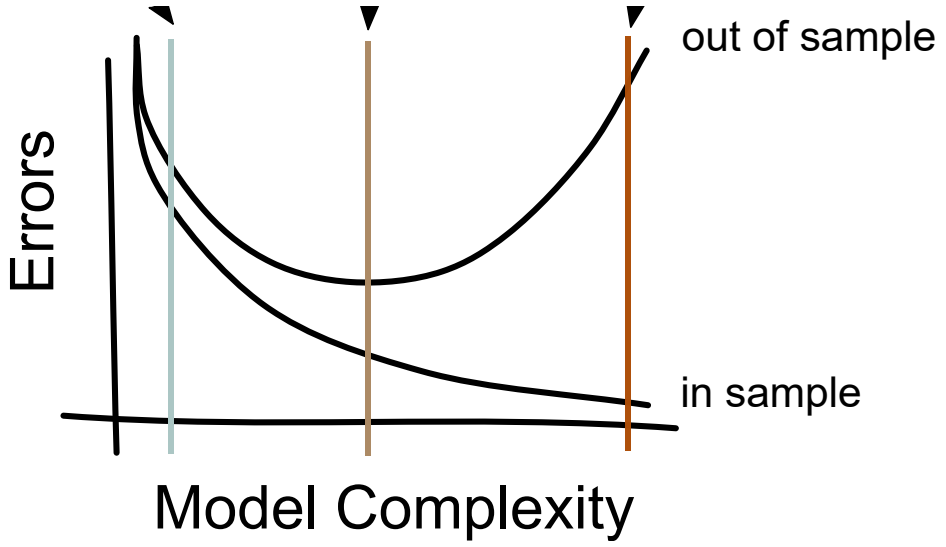
Transitioning to Inference

- Inference: ML algorithms in action. Apply live data into the (learned) model to get output
- To get a meaningful inference, we need to be more introspective, trying to understand properties of our algorithms
- The heart of inference: Being critical of the processes people use to learn from data

Recall: Bias-Variance Tradeoff

- Ultimately, you want your model to perform well on out-of-sample data
- If you only evaluate on in-sample data, you will underestimate the out-of-sample error

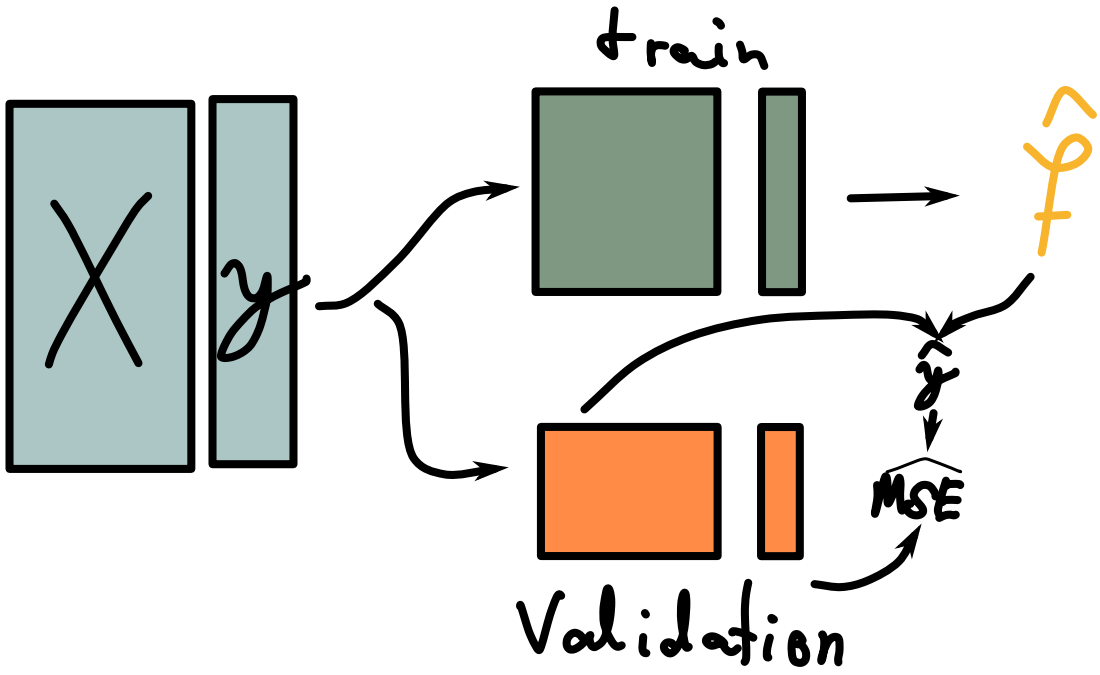




Use validation to check the performance. ie rough estimate(accuracy)

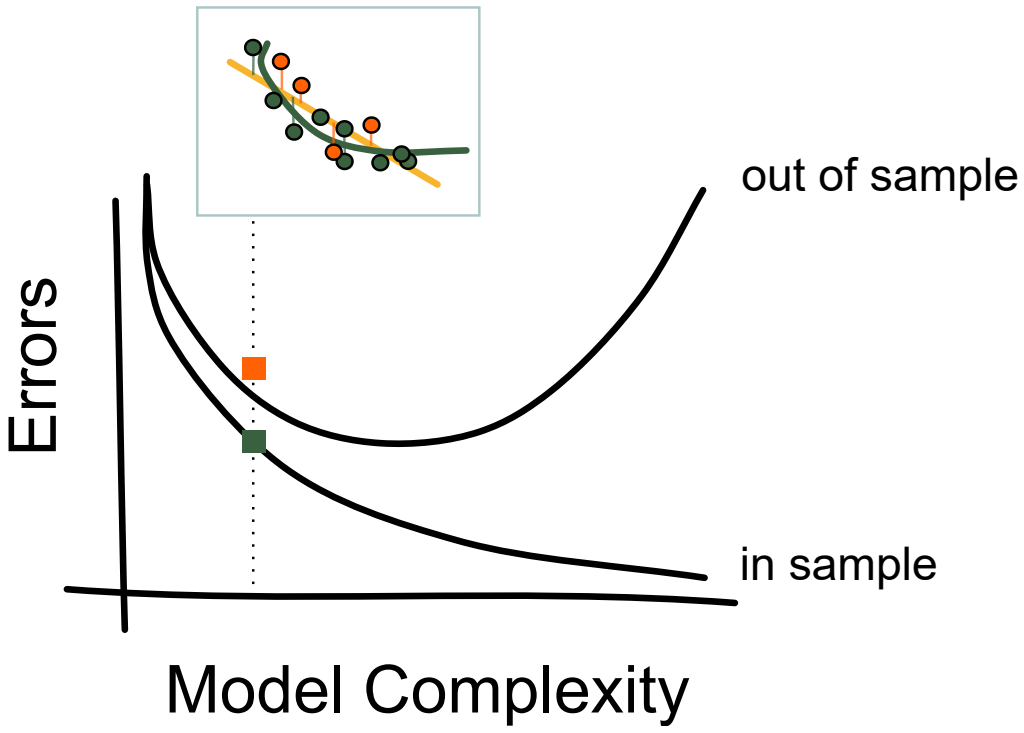
Validation Sets

- To approximate the out-of-sample error, we can use a validation set.
- Randomly divide your sample into two pieces, one to train and another to validate



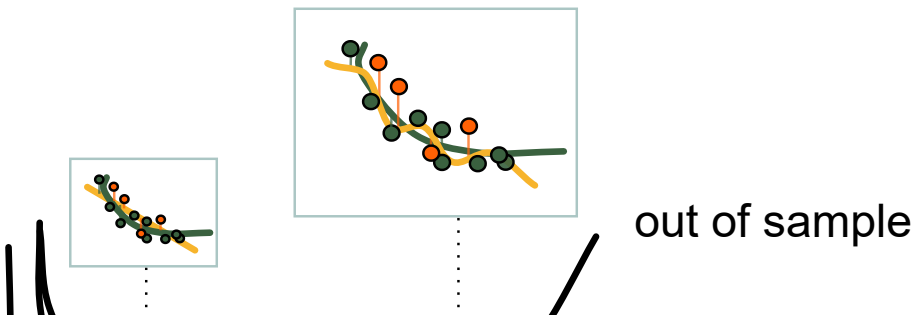
Validation Sets

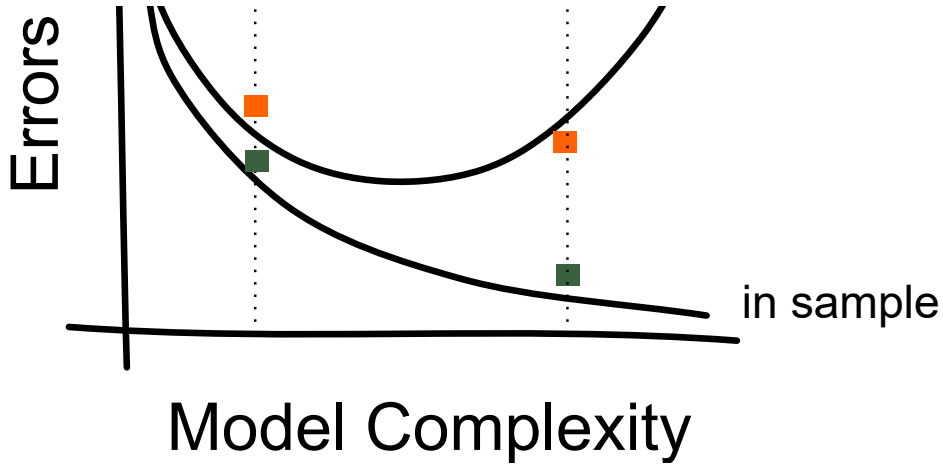
If you run this over models with different degrees of complexity, you can see the bias-variance tradeoff.



Validation Sets

If you run this over models with different degrees of complexity, you can see the bias-variance tradeoff.



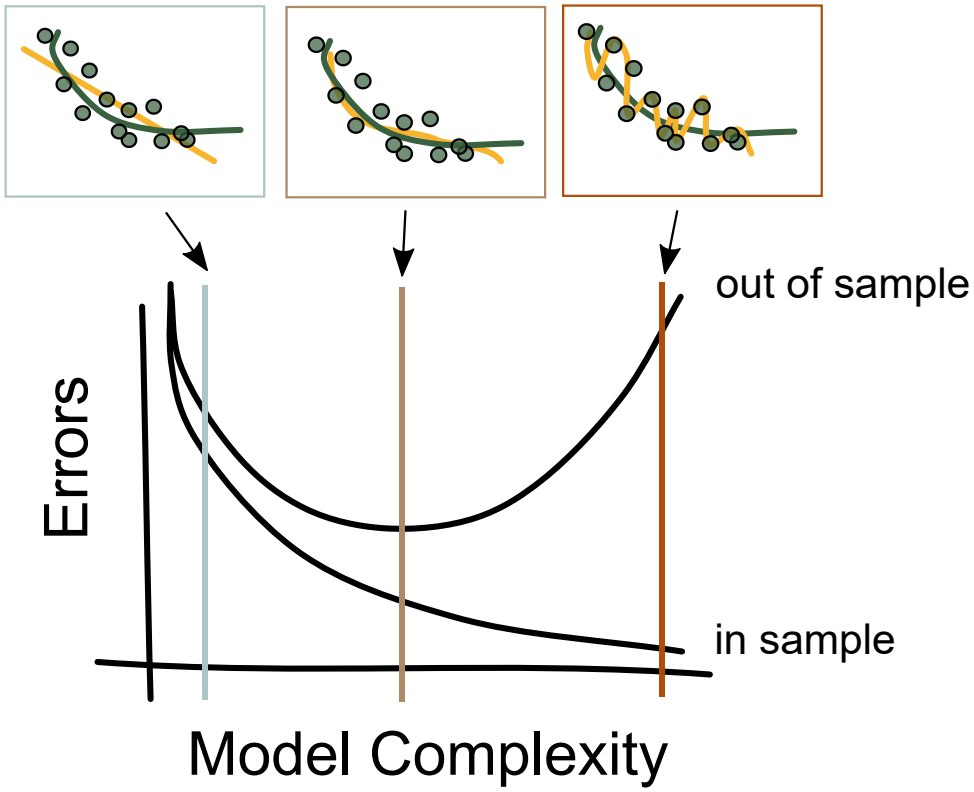


Complexity Regimes

Even if you only evaluate the train / validation error for a model of a given complexity, you get useful information.

- Training $\ll$ validation error $\rightarrow$ Model is overfit
- Training $\approx$ validation error $\rightarrow$ Model is underfit (or OK)

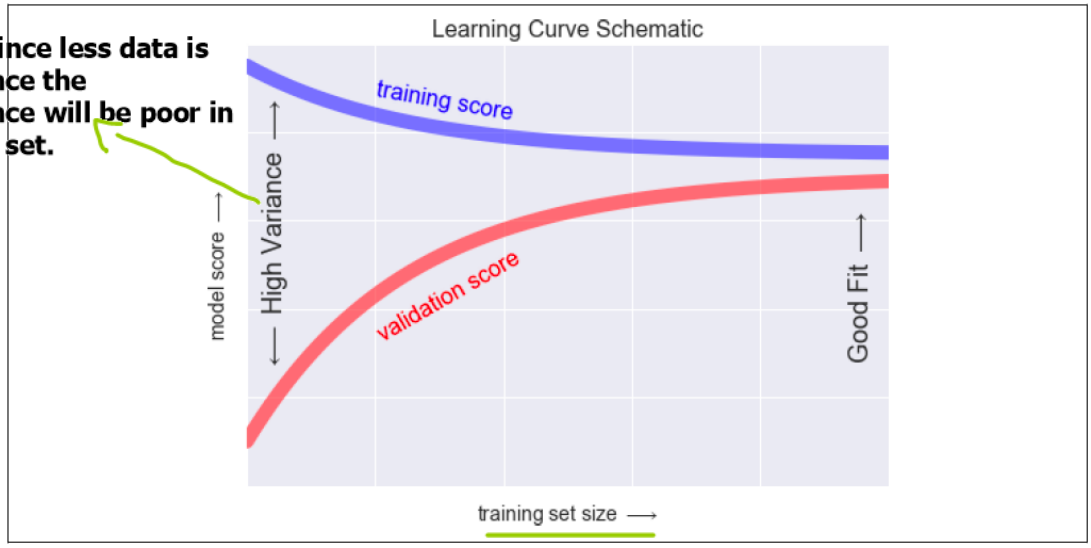
Common heuristic: Overfit the data first, then regularize.



Learning Curves

- As you gather more data, how much better do your models get?
- *Model score*: this can guide the decision to collect more data.
    - training score and the validation score are both low, the estimator will be **underfitting**
    - training score is high and the validation score is low, the estimator will be **overfitting**
    - Okay otherwise.

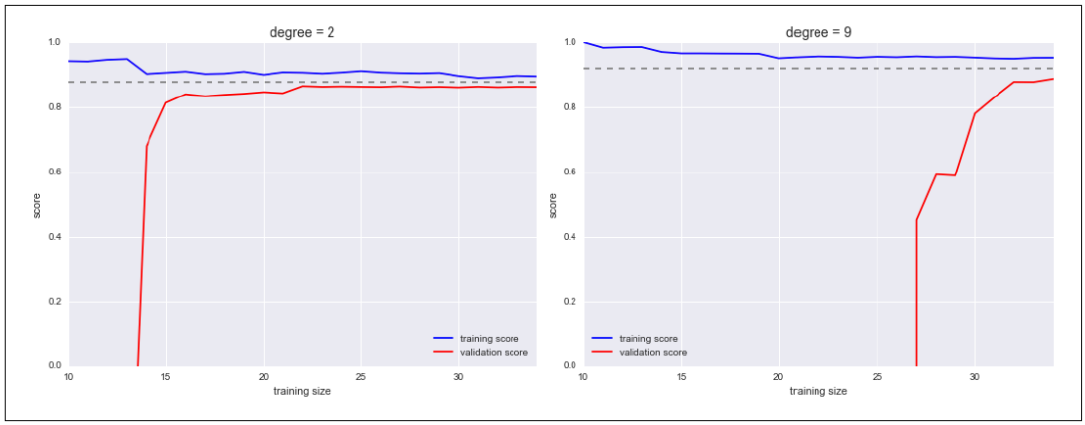
overfits, since less data is there. Hence the performance will be poor in validation set.



Learning Curves

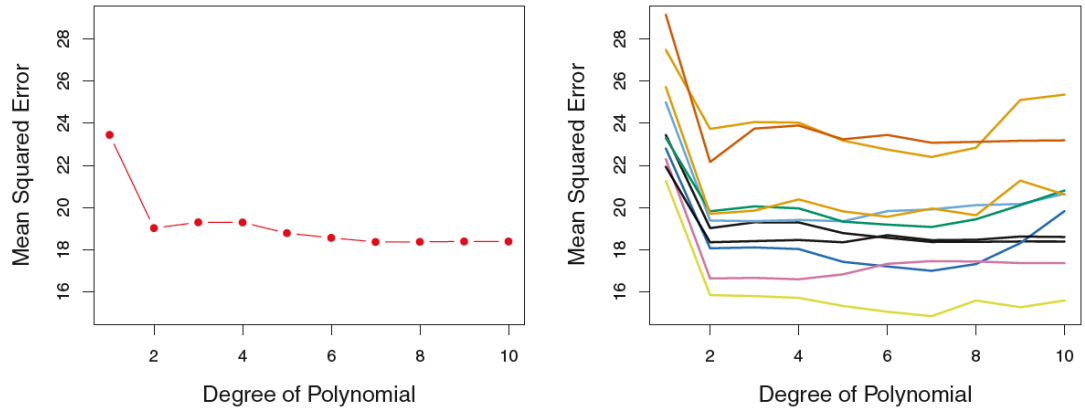
- Models of different complexities have different learning curves
- Larger models don't saturate as quickly. They are,
    - worse than small models on small datasets
    - better than small models on large datasets

For higher degree, we need more data to give a generalize model



Evaluation and Randomness

- We are only *estimating* out-of-sample error
- These estimates might be good or bad
  - Have randomness from choice of validation set
  - Have randomness from dataset collection

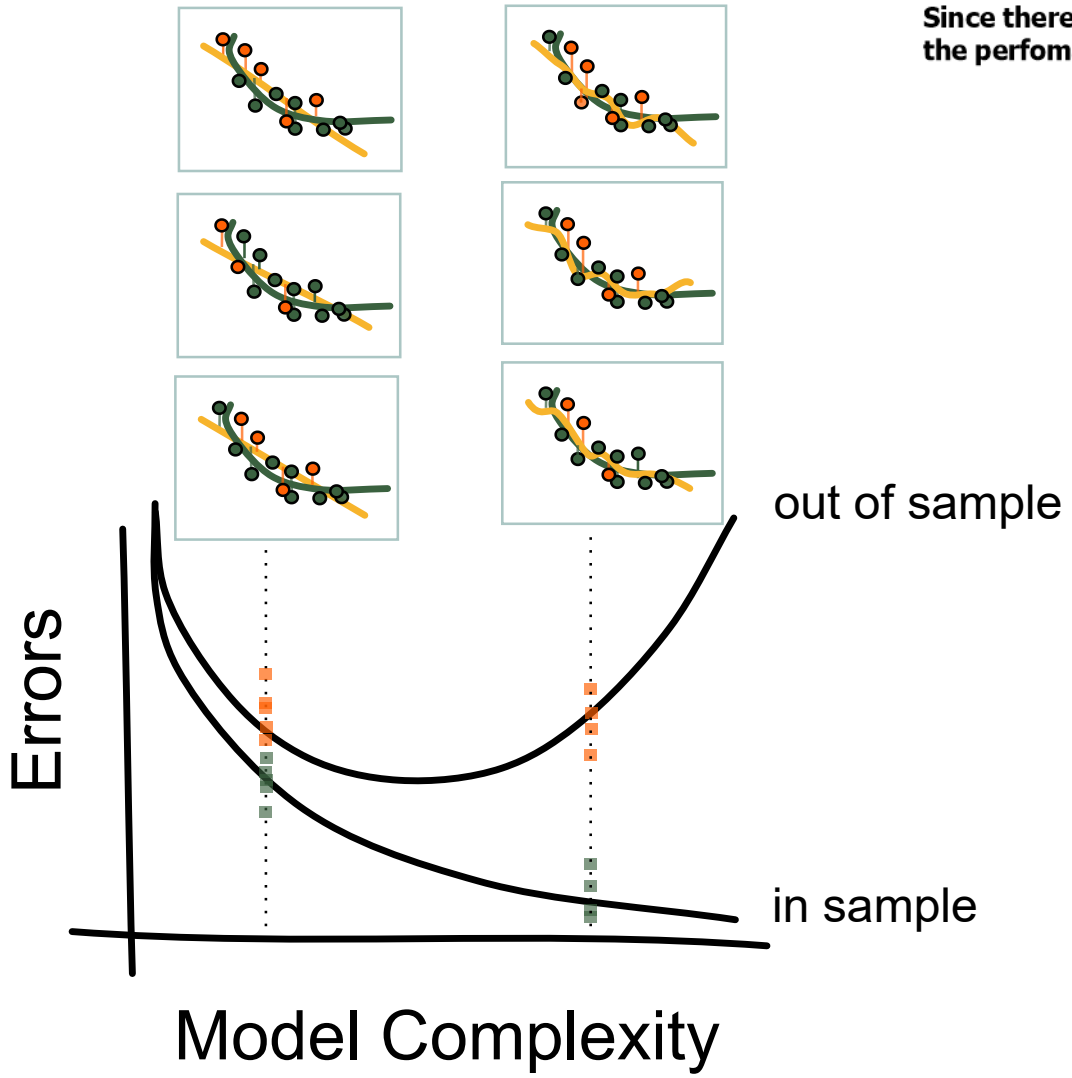


Evaluation and Randomness

- We are only *estimating* out-of-sample error
- These estimates might be good or bad

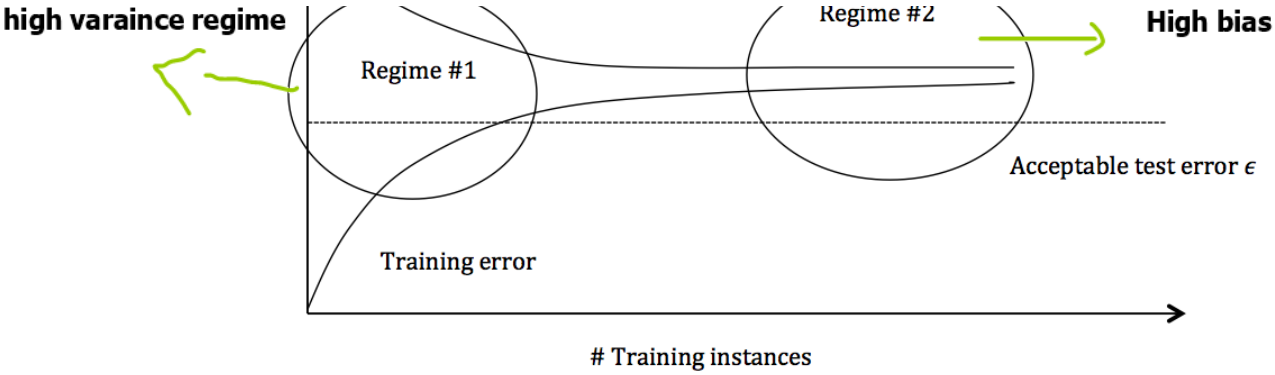
- Have randomness from choice of validation set
- Have randomness from dataset collection

Since there is randomness, the perfomance might defer



Detection of bias and variance from training and validation error

- Variance occurs since different validation sets give different estimates
- Bias occurs when training on subset leads to worse expected performance (remember learning curves)



Regime # 1 (High variance)

Symptoms

- Training error is much lower than test error
- Training error is lower than acceptable threshold ϵ
- Test error is above ϵ

Remedies

- Add more training data
- Reduce model complexity -- complex models are prone to high variance
- Bagging (will be covered later in the course)

Regime # 2 (High bias)

Symptoms

- Training error is higher than ϵ

Remedies

- Use more complex model (e.g. non-linear instead of linear)
- Add features
- Boosting (will be covered later in the course)

Instead of using one validation set, one could have few alternative methods of getting more robust valiadation (less randomness).

We will talk about:

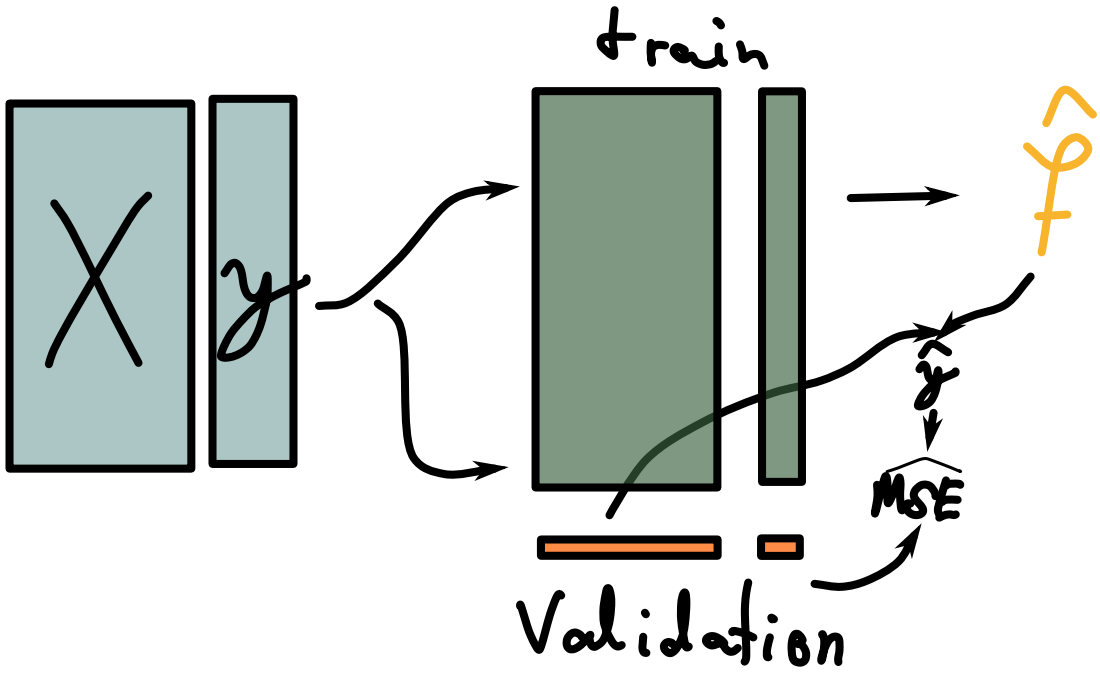
- Leave-One-Out Cross Validation [LOOCV]
- K-Fold Cross Validation.

LOOCV

fit a model without the sample and test the built model on the sample data that you have holded

1. Fit your model without sample (x_i, y_i). Call the fit $\hat{f}_{-i}$.
2. Compute holdout $\widehat{MSE}_i := (y_i - \hat{f}_{-i}(x_i))^2$
3. Estimate the out-of-sample error by averaging this over all possible holdouts coming from (1) and (2),

$$\widehat{MSE} = \frac{1}{n}\sum_{i=1}^{n} \widehat{MSE}_i$$



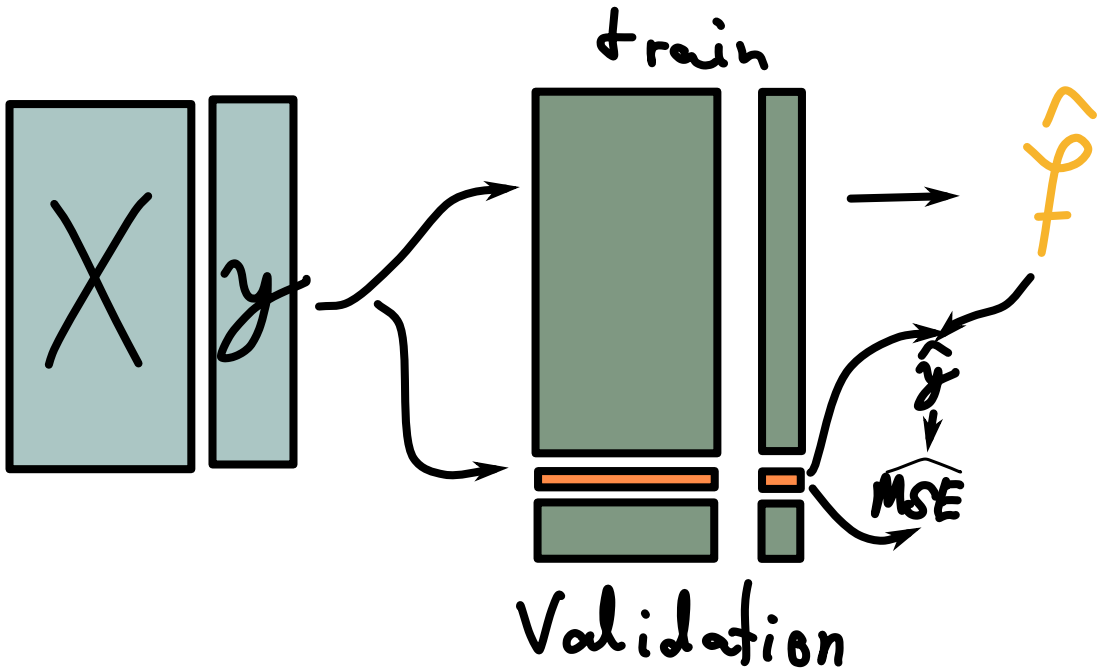
Repeat the process multiple times and take the average.

LOOCV

1. Fit your model without sample (x_i, y_i). Call the fit $\hat{f}_{-i}$.
2. Compute holdout $\widehat{MSE}_i := (y_i - \hat{f}_{-i}(x_i))^2$

3. Estimate the out-of-sample error by averaging this over all possible holdouts coming from (1) and (2),

$$\widehat{MSE} = \frac{1}{n}\sum_{i=1}^{n}\widehat{MSE}_i$$

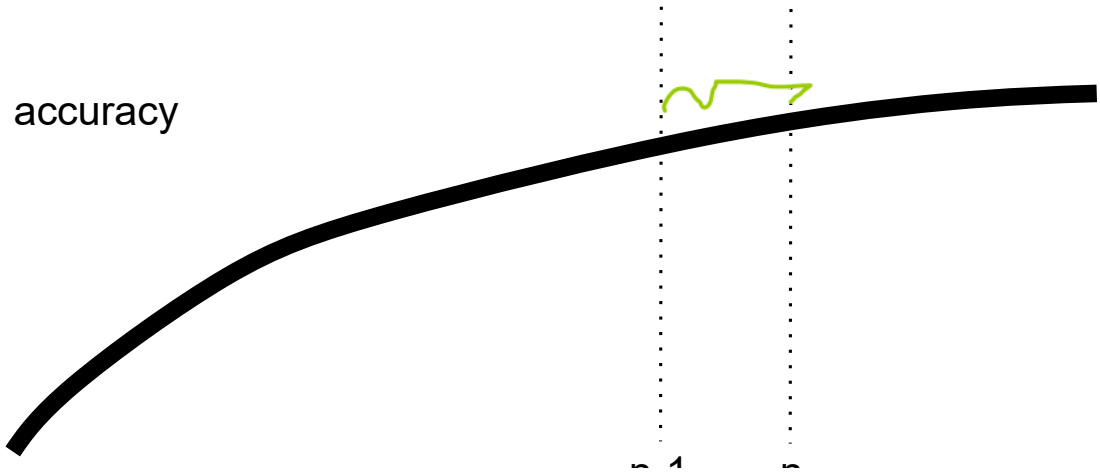


LOOCV

All the features will be learnt as the model is computed multiple time on the data set, HEnce we have lower bias

Advantages

- Lower bias. We use almost all the training data, so we don't underestimate performance.



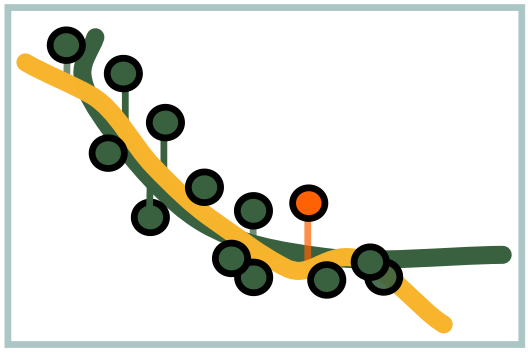
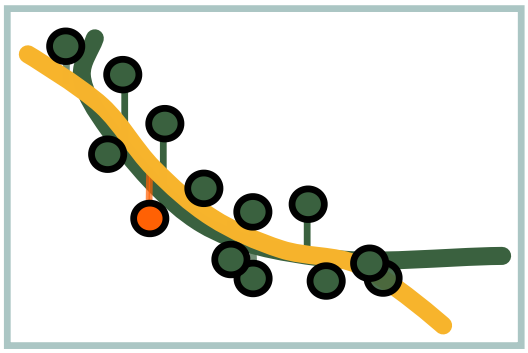
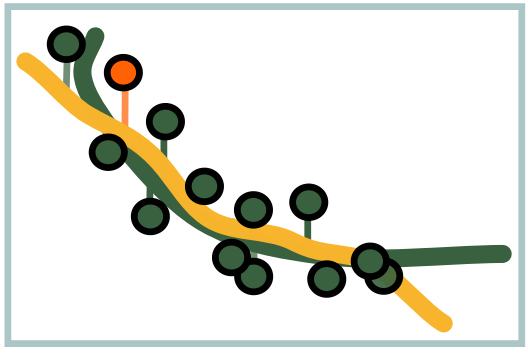
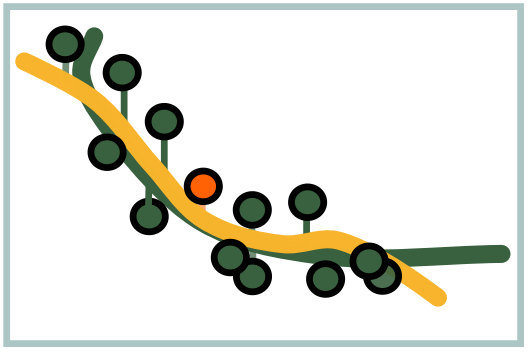
training data size

LOOCV

Cant be used on high dataset.

Disdvantages

- High computational complexity (except linear regression)
- The trained models are correlated
  - The $\widehat{MSE}_i$ are correlated
  - The average of correlated variables has larger variance than the average of independent ones
  - The out-of-sample estimate has higher variance

K-Fold CV

Rather than 1, we partition into k folds.

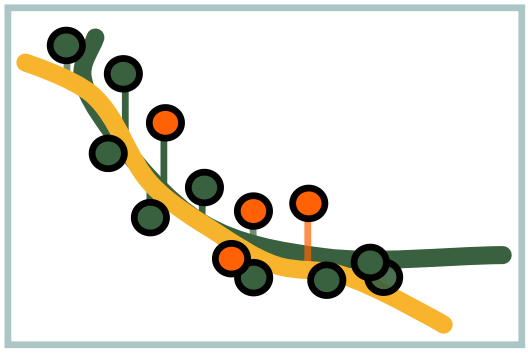
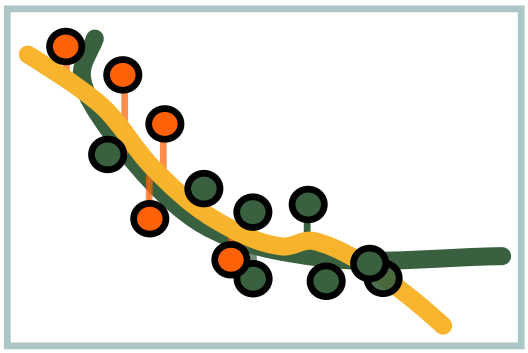
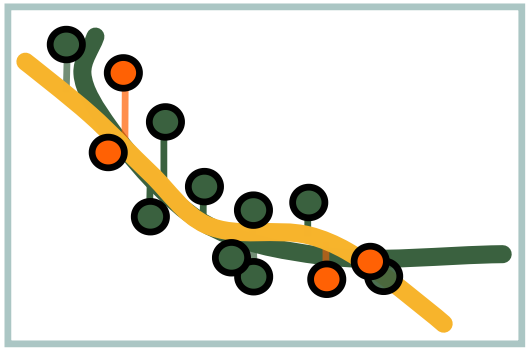
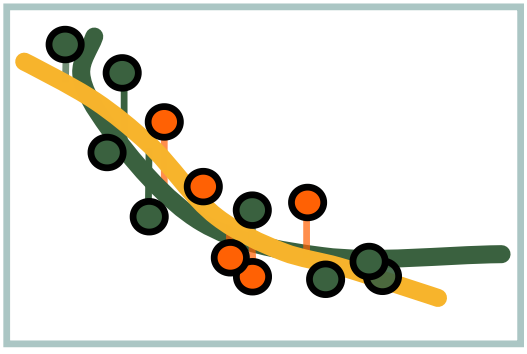
1. Randomly partition samples into one of K folds, $\{S_1, \ldots, S_K\}$.
2. Fit your model without fold S_k. Call the fit $\hat{f}_{-k}$.
3. Compute holdout $\widehat{MSE}_k := \sum_{i \in S_k} (y_i - \hat{f}_{-k}(x_i))^2$
4. Estimate the out-of-sample error by averaging over folds,

$$\widehat{MSE} = \frac{1}{K} \sum_{k=1}^{K} \widehat{MSE}_k$$

K-Fold CV

Advantages

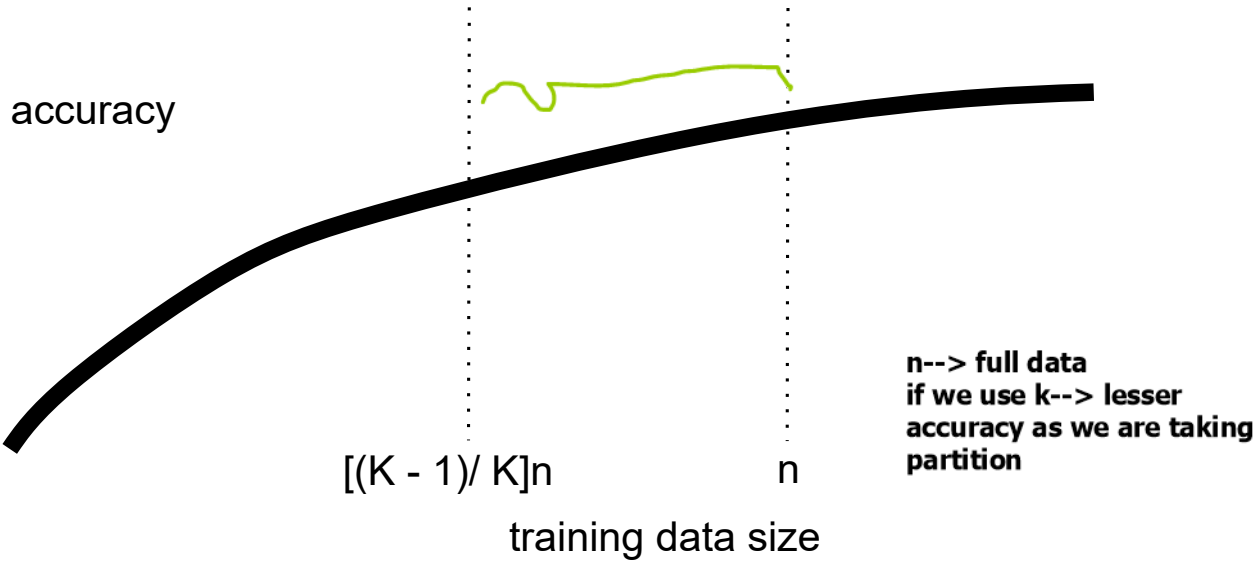
- More computationally tractable
- Learns less correlated models
  - The estimates $\widehat{MSE}_k$ are less correlated
  - The estimate $\widehat{MSE}$ has lower variance

K-Fold CV

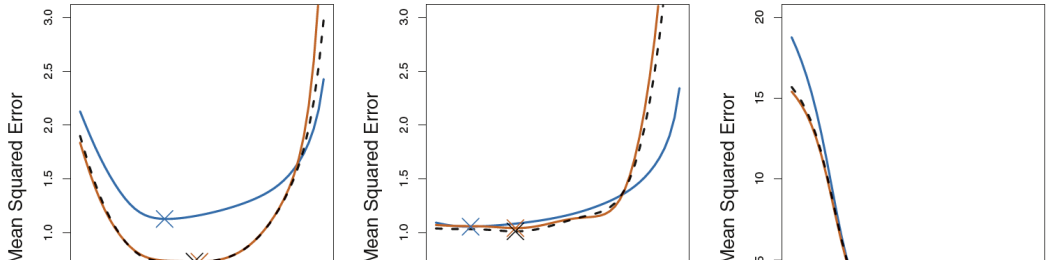
Disadvantages

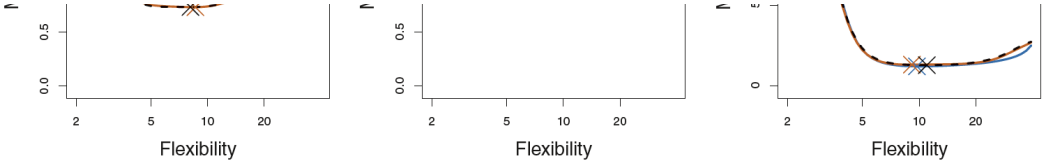
- We don't train using the full training set: high bias
  - Model on full dataset is actually better than estimated (overestimating error)



Estimation Quality: LOOCV and K-Fold

- The blue curves are known out-of-sample MSE's from a simulation experiment
- Black and orange are LOOCV and K-Fold estimates, respectively
- Note: Even when estimates of out-of-sample MSE is poor, the estimate of the minimum might be good





Hyperparameter Search

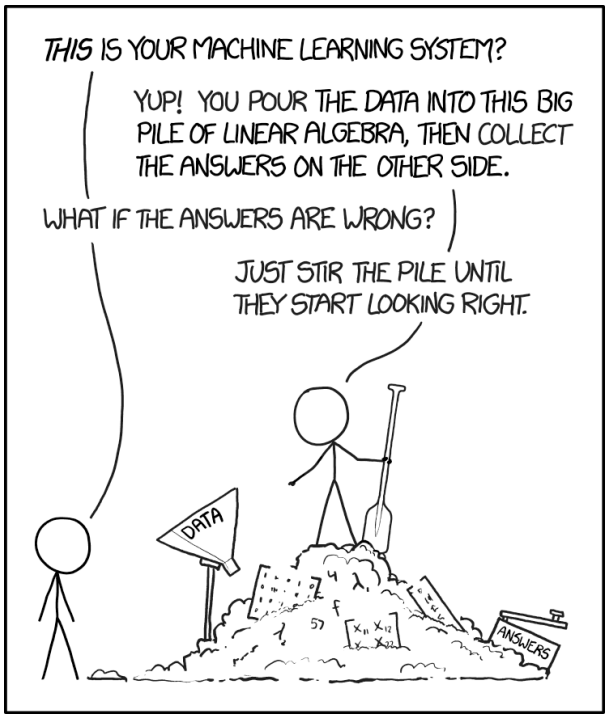
- We will often have many parameters to tune simultaneously
  - Model parameters: Polynomial degree, # trees, ...
  - Training parameters: Learning rate, subsampling, ...
  - Preprocessing: Normalization, outlier removal, ...
- No single "model complexity" parameter

Search Options

- Manual search
- Grid search
- Random search
- Combinations of these

Manual Search

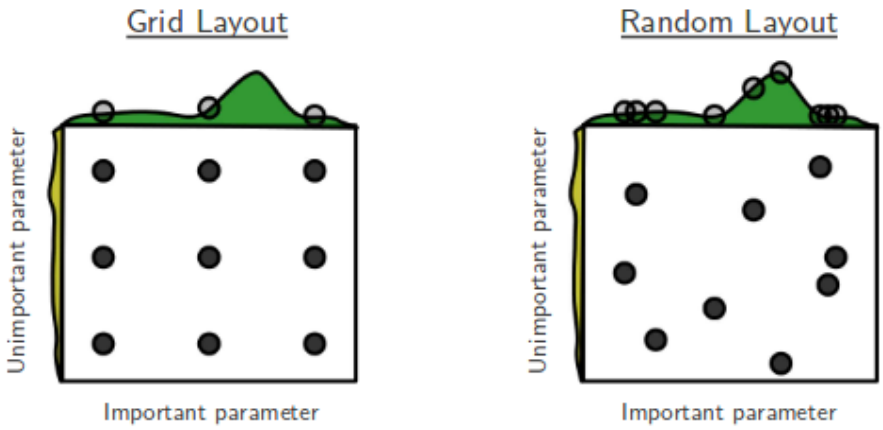
- Reasonable strategy: use *coordinate descent* - change one hyper-parameter at a time, make it better from where it was until now.
- Focus more on certain parameters (fine tune the updates). Guide your choice of parameters by which regime (over vs. underfitting) you are in
- Advantage: Uses bias-variance tradeoff information
- Disadvantage: Tedious, not fully reproducible



Grid Search

- Compute out-of-sample error on all combinations of parameters
- Advantage: Automatic, easy to implement
- Disadvantage: Exponentially many parameter

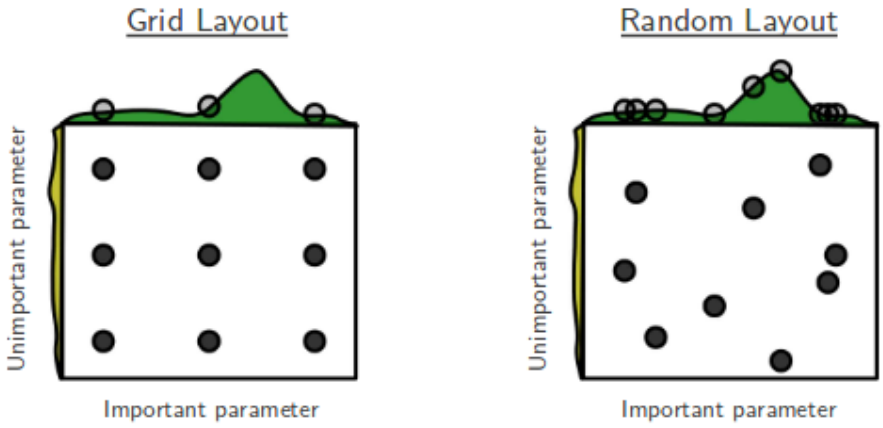
configurations; might take long to identify relevant parameters



Random Search

the parameters are choose randomly based on the range mentioned.

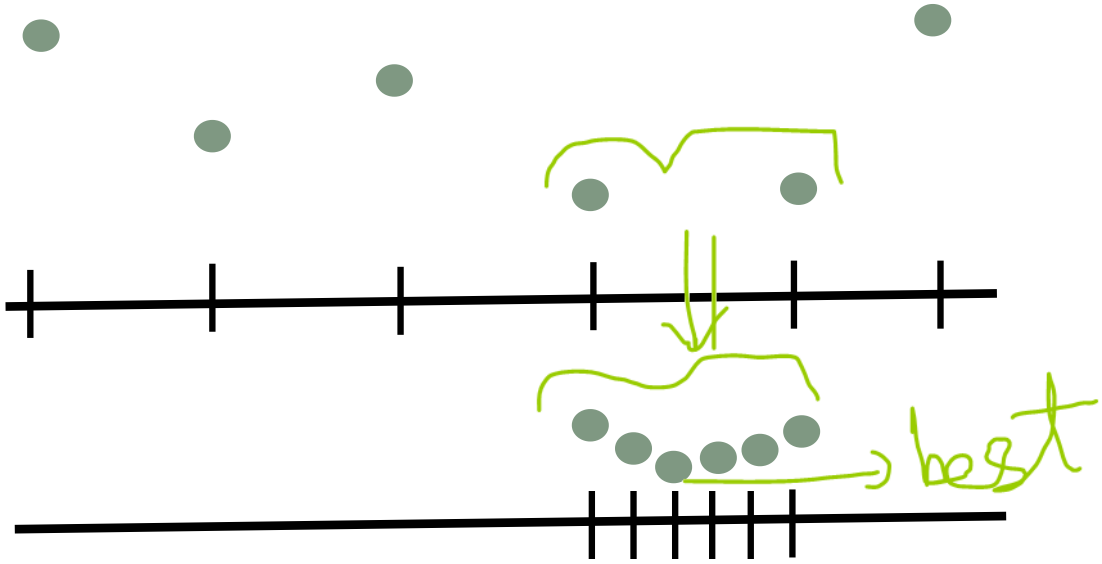
- Compute out-of-sample error on random samples of parameters
- Advantage: Automatic, easy to implement. Relevant parameters become clear quickly.
- Disadvantage: Still suffers when there are many parameters.



Combinations

- Can fix a few parameters manually, and use random search for others
- Can use "multi-resolution" search. Automatically search over predefined grids, but manually set the grids to more promising regions.
  - Start with a broad range for each hyper-parameter. Manually update (using coarse tuning, large changes at every step).
  - Once you found few best configurations, start fine tuning them more locally with smaller changes around the nominal value.

its kind off gradient descent.




```
keypresscontrol = undefined
```