Evaluating Performance II

Modeling Considerations

Accuracy

Computational Efficiency

Interpretability

Accuracy
Supervised Learning Performance Evaluation

Regression

Classification

Binary

Multiclass

Cost / Loss Functions

Regression:

- Mean squared error (MSE)
- Mean absolute error (MAE)
- Huber loss

Classification:

- Cross entropy / log loss

Performance Evaluation Metrics and Tools

Regression:

- Root mean squared error (RMSE)
- R^2, coefficient of determination

Binary:

- Classification accuracy
- True positive rate (Recall)
- False positive rate
- Precision
- F_1 Score
- Area under the ROC curve (AUC)
- Receiver Operating Characteristic (ROC) curves

Multiclass:

- Classification accuracy
- Micro-averaged F_1 Score
- Macro-averaged F_1 Score
- Confusion matrices
- Per class metrics (recall, precision, etc.)

We can always compute our accuracy metrics of a trained model on our test set…

…BUT, they may not be valid (i.e. may not reflect generalization performance) if:

1. The underlying data are NOT representative of what we will encounter in practice

2. The test data set DOES NOT remain separate from our model training process

Goal: estimate generalization performance

Spot the misstep

1

1. Your train a logistic regression algorithm on training data
2. You evaluate the generalization performance of your trained algorithm on the training data
3. Your estimated performance is exceptional!

NEVER USE THE SAME DATA USED FOR TRAINING FOR ESTIMATING GENERALIZATION PERFORMANCE

2

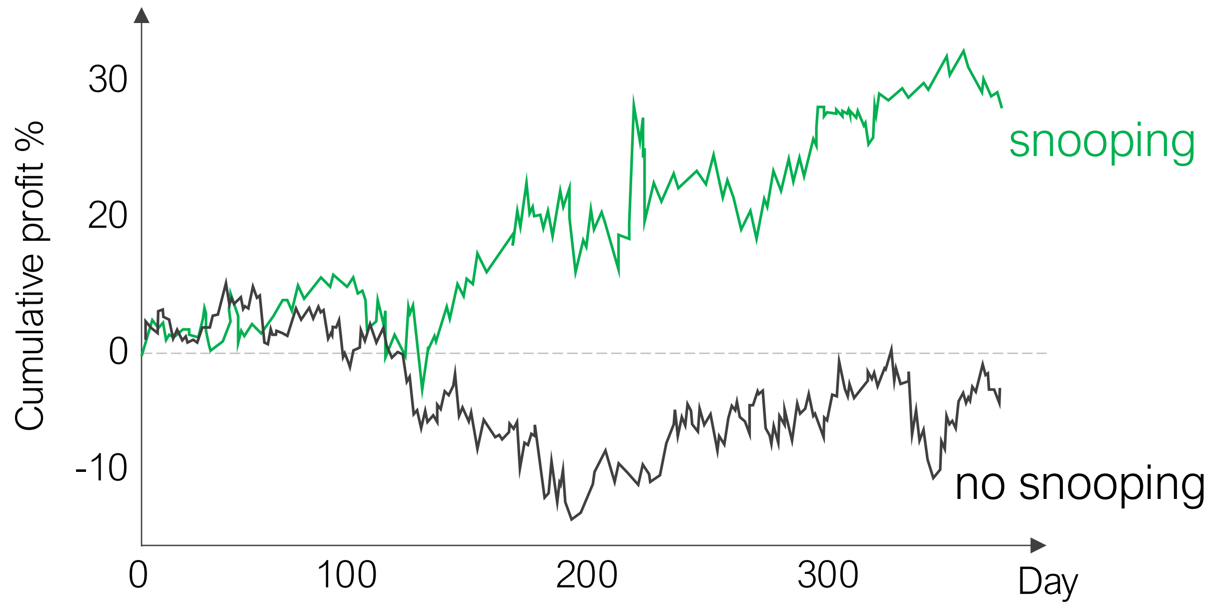
1. Goal: predict the exchange rate for the U.S. Dollar vs British Pound (using 20 past observations)

2. You take your historical data, normalize it, then split it randomly into a training and test set
   DATA SNOOPING!

3. You train on the training data, test on the test data

Results:
Your predictions are correct 56% of the time

Estimate your profits…



All preprocessing should be based on the training data alone

Abu-Mostafa, Learning From Data

3

1. Goal: predict the Dow Jones Industrial average

2. You randomly split your data into a training and test dataset

3. Choose a model with lots of flexibility

4. You iterate on the following process hundreds of times:
    1. Train your model on the training data
    2. Test your model on the test data
    3. Evaluate performance on the test data

DATA SNOOPING!

5. Report that you were able to achieve 75% accuracy on your test set!

4

1. Goal: predict long-term performance of a “buy and hold” strategy in stocks
2. You collect 50 years of historical data and include all companies that are currently traded in the S&P500

   SAMPLING BIAS!
3. You randomly split your data into a training and test dataset.
4. You assume you will strictly follow the “buy and hold” strategy
5. You then use apply your model on the current portfolio and predict that you will be rich in retirement!

Abu-Mostafa, Learning From Data

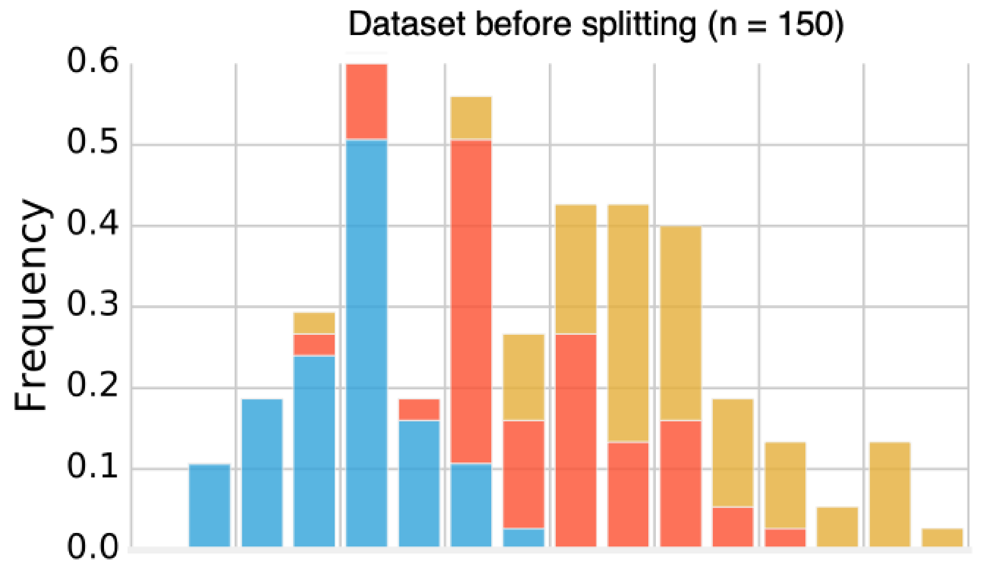
Data snooping / leakage

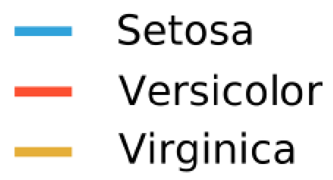
If a test data set has affected **any step** in the learning process, its ability to assess the generalization performance has been **compromised**.

Abu-Mostafa, Learning From Data

Sampling bias

Are the data we're using for machine learning **representative of the population you will apply on in practice**?



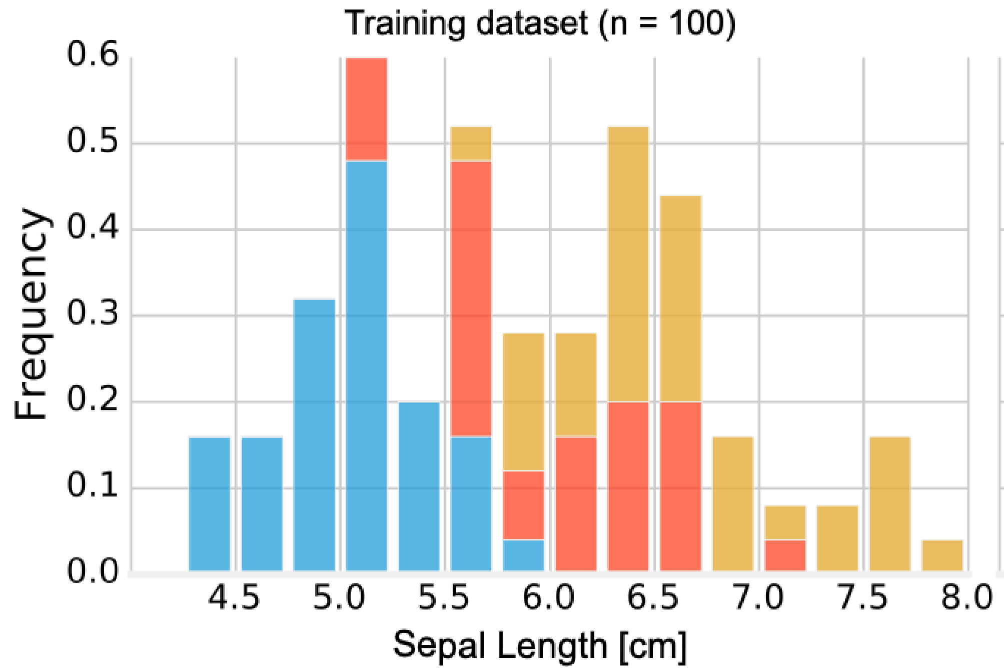


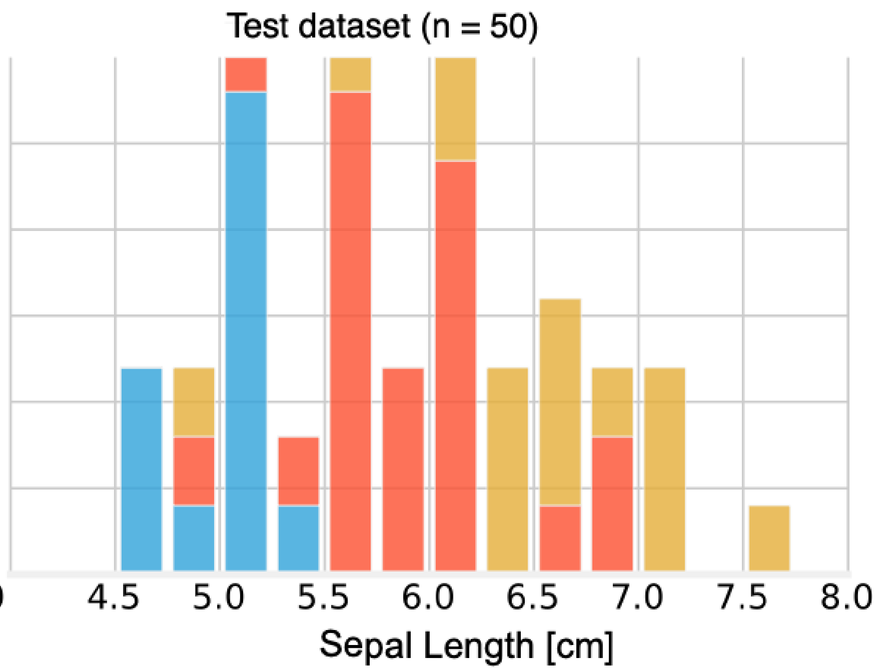
All: 50 Setosa, 50 Versicolor, 50 Virginica

Train: 38 Setosa, 28 Versicolor, 34 Virginica

Test: 12 Setosa, 22 Versicolor, 16 Virginica

One form of sampling bias





Ideally training and test sets are independent and statistically representative of the population

Dividing up your dataset we violate independence assumptions

Reduce this bias with **stratified sampling**

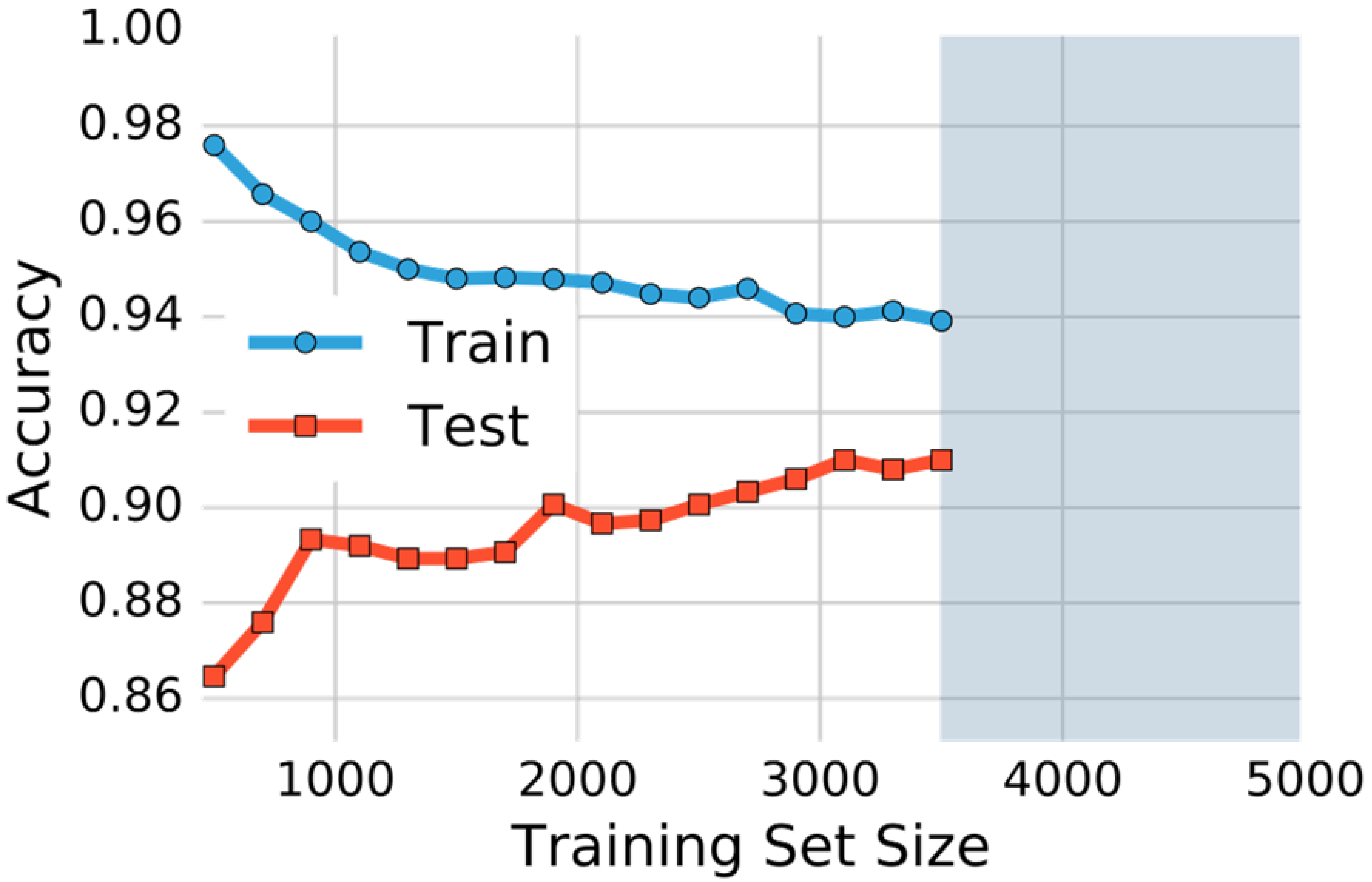
Images from Sebastian Raschka (https://sebastianraschka.com/blog/2016/model-evaluation-selection-part1.html)

Sample Size

Ideally, we would use infinite samples in our training set representing the population

In practice, we try to use as much data as possible

Larger datasets may also reduce overfit

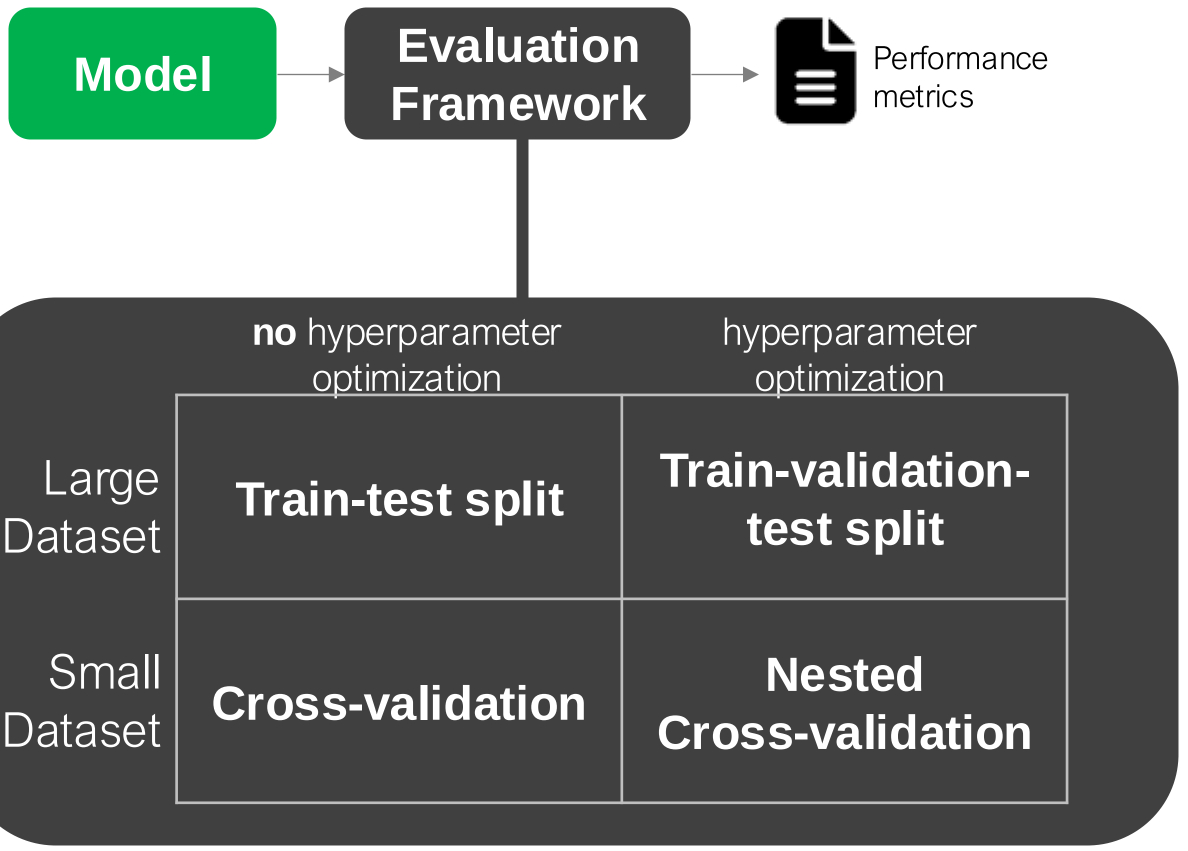


Images from Sebastian Raschka (https://sebastianraschka.com/blog/2016/model-evaluation-selection-part2.html)

1 How do we use the metrics to evaluate performance?

Basic ML Experimental Design

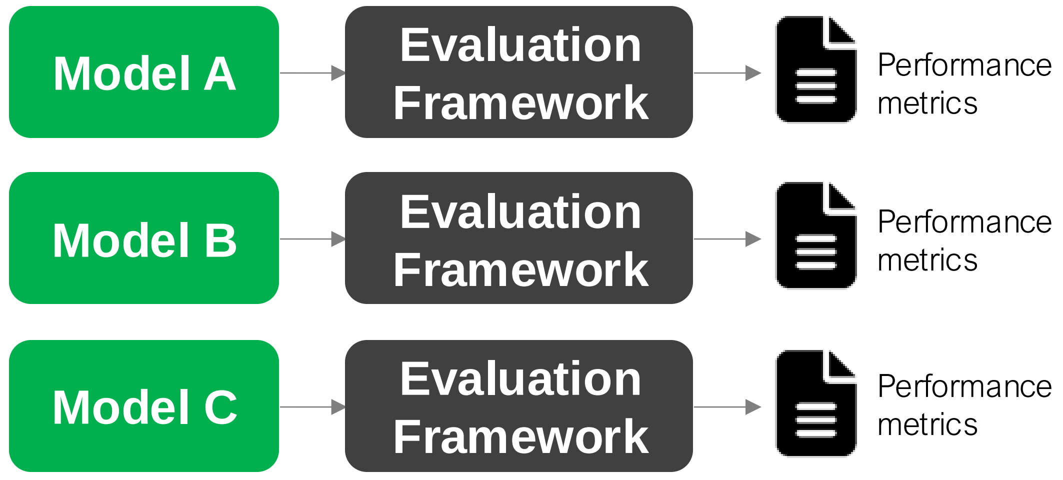
1 Generalization Performance Evaluation



	no hyperparameter optimization	hyperparameter optimization
Large Dataset	**Train-test split**	**Train-validation-test split**
Small Dataset	**Cross-validation**	**Nested Cross-validation**

2 Model Comparison

(experiment to determine the best-performing algorithm)



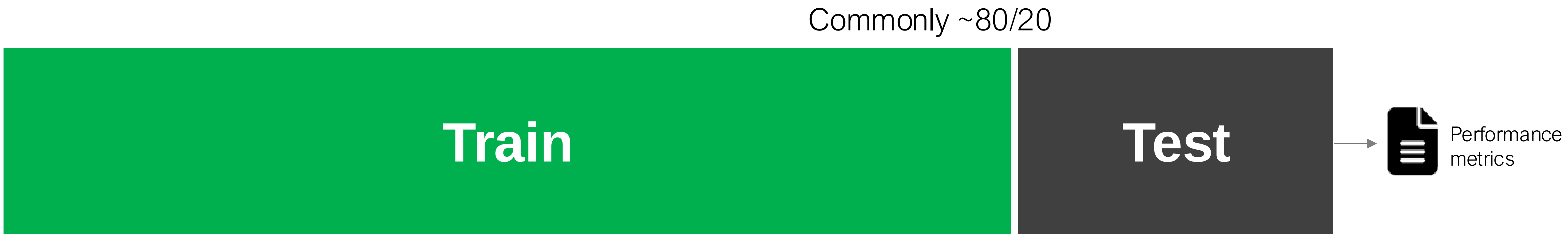
Compare models according to metrics

Only vary the model!

(don't vary the data, evaluation framework, etc. for a fair comparison)

Train-Test Split

Learning model parameters and evaluating performance



Used for model training: select model parameters

Evaluate generalization performance

1. If our test split is too small, our estimate of generalization performance will have high variance
2. Not using all data for training produces an algorithm that is pessimistically biased
3. For small datasets, this reduction in dataset size may be detrimental

What are Hyperparameters?

Parameters: Configuration variable that control model predictions that are adjusted during the training process based on data

Hyperparameters: parameters set prior to model training; they are not modified during the training procedure, but often impact the training procedure.

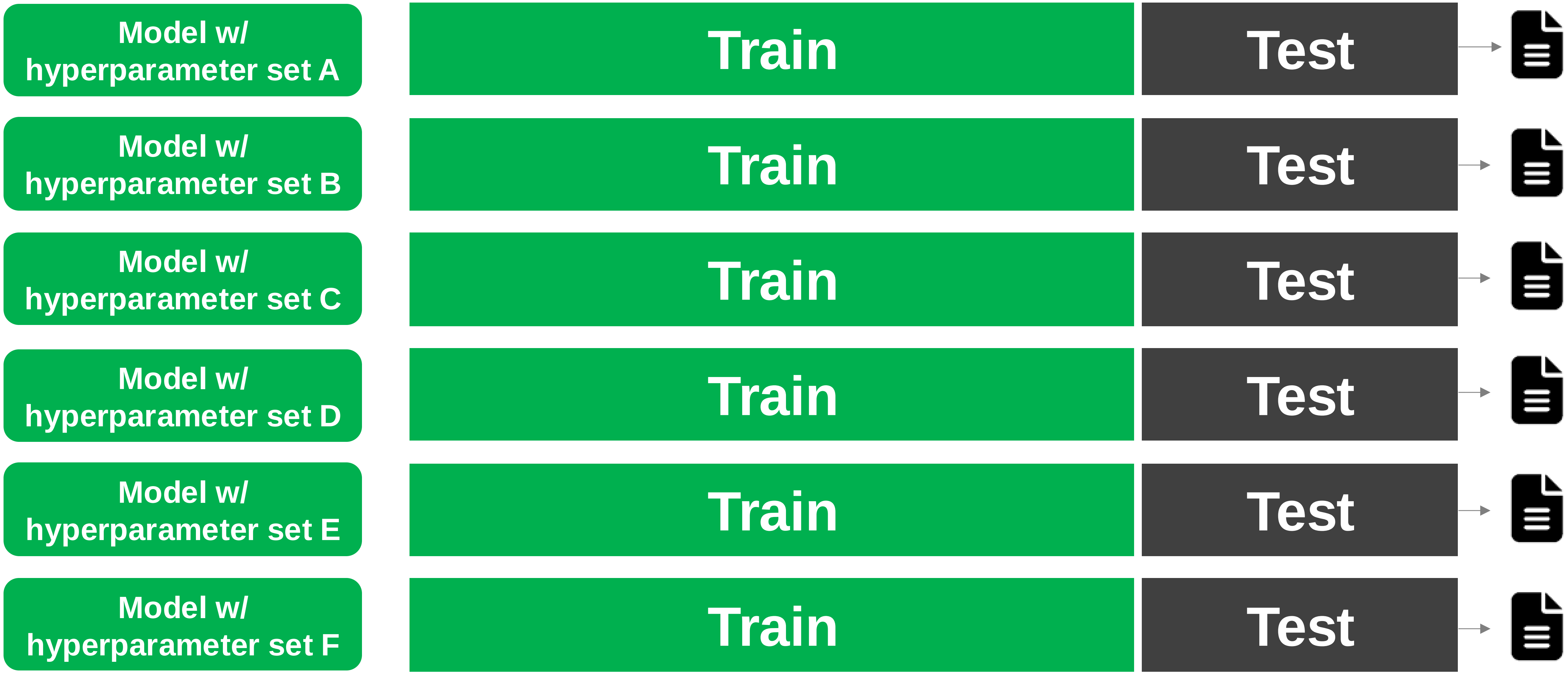
Hyperparameter Examples

- k in KNN
- Learning rates for gradient descent of your model fitting procedure
- Model architectures (e.g. number and types of layers in neural networks)

What happens if we re-evaluate the model with different hyperparameters?

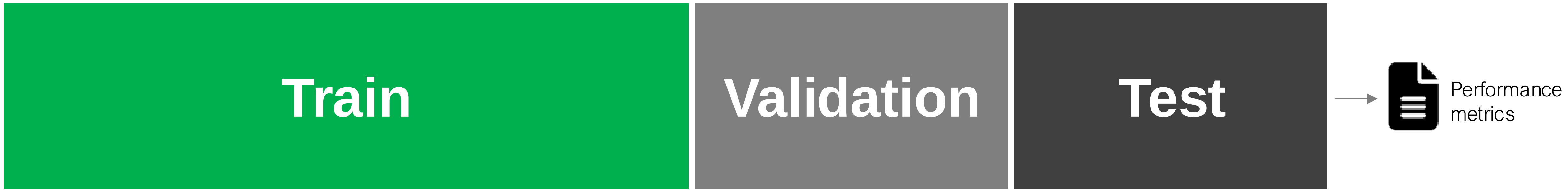
cannot use test data to select model

We may overfit to our held-out test data!


Model w/ hyperparameter set A
Train
Test
Model w/ hyperparameter set B
Train
Test
Model w/ hyperparameter set C
Train
Test
Model w/ hyperparameter set D
Train
Test
Model w/ hyperparameter set E
Train
Test
Model w/ hyperparameter set F
Train
Test

Training, Validation, Test Split

Learning model parameters AND **hyperparameters** and evaluating performance



Used for model training / fitting

Used to optimize hyperparameters and/or compare models

Used to evaluate generalization performance of the final model(s)

Hyperparameters: parameters that control how your algorithm learns; typically set before training begins (e.g. k in KNN, learning rate, etc.)

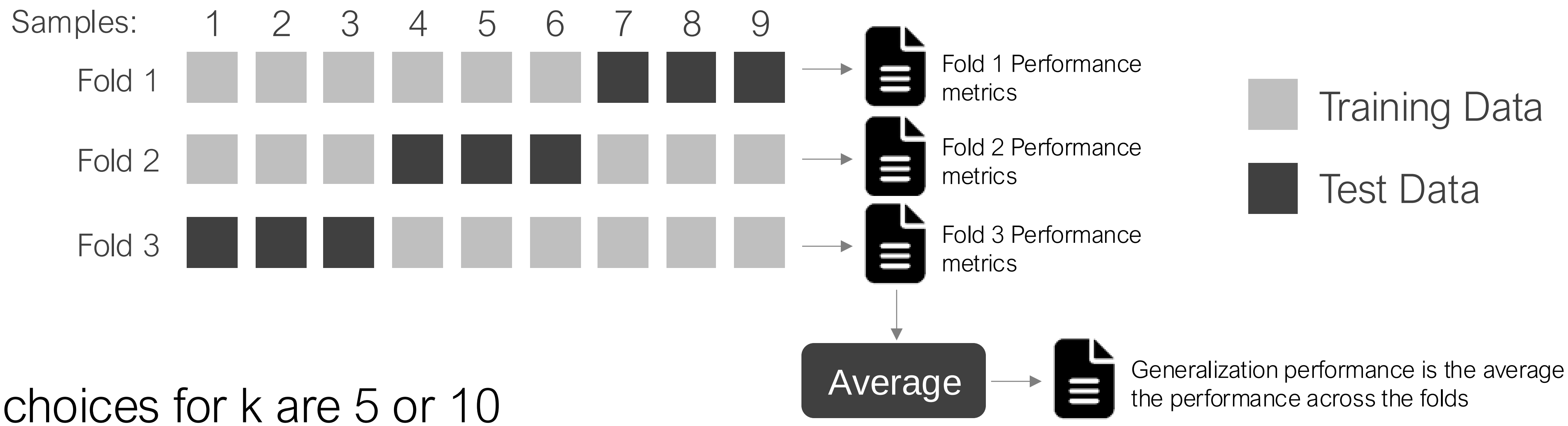
What if you have a small dataset?

K-folds cross-validation

K-fold cross validation K = 3

train on first (total num of sample)*(k-1)/k
test on last (total num of sample)/k

1 Performance evaluation: Train your model K times, once for each fold


Samples:
1
2
3
4
5
6
7
8
9
Fold 1
Fold 2
Fold 3
Fold 1 Performance metrics
Fold 2 Performance metrics
Fold 3 Performance metrics
Training Data
Test Data
Average
Generalization performance is the average of the performance across the folds

Typical choices for k are 5 or 10

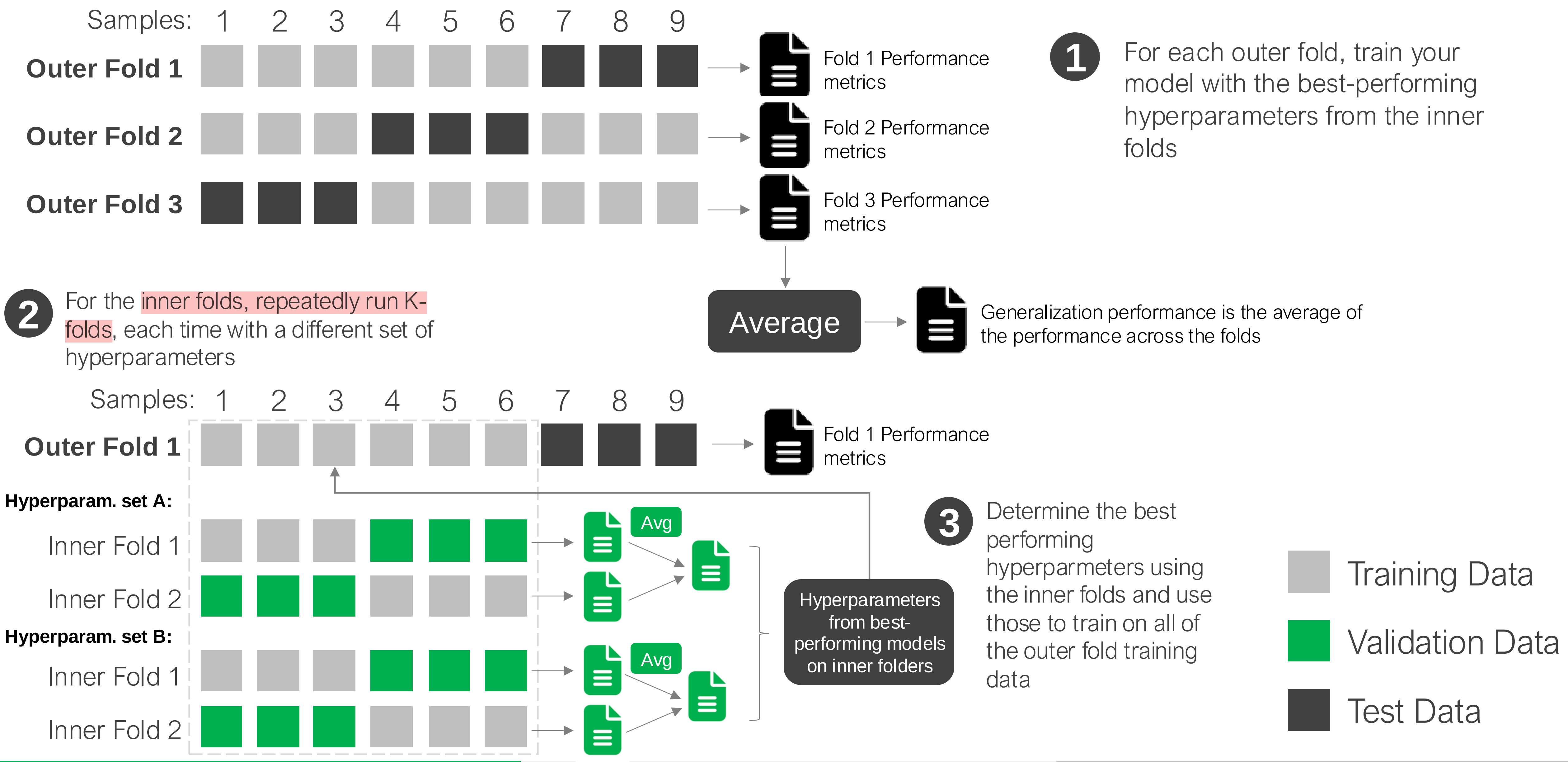
Average performance metrics across the splits

If k = N (number of samples): Leave-one-out cross validation

The number of splits impacts the bias-variance tradeoff of your performance estimates
(larger k means lower bias on the performance estimate, but with higher variance)

What if you need to select hyperparameters for a small dataset?

Nested cross-validation with hyperparameters


Samples: 1 2 3 4 5 6 7 8 9
Outer Fold 1
Outer Fold 2
Outer Fold 3
Fold 1 Performance metrics
Fold 2 Performance metrics
Fold 3 Performance metrics
Average
Samples: 1 2 3 4 5 6 7 8 9
Outer Fold 1
Fold 1 Performance metrics
Hyperparam. set A:
Inner Fold 1
Inner Fold 2
Hyperparam. set B:
Inner Fold 1
Inner Fold 2
Avg
Avg
Hyperparameters from best-performing models on inner folders
Training Data
Validation Data
Test Data

1 For each outer fold, train your model with the best-performing hyperparameters from the inner folds

Generalization performance is the average of the performance across the folds

2 For the inner folds, repeatedly run K-folds, each time with a different set of hyperparameters

3 Determine the best performing hyperparmeters using the inner folds and use those to train on all of the outer fold training data

When to use each technique for performance evaluation?

none of these help with hyperparameter optimization
only help with model generalization performance
hyperparameter optimized between/among trainings not inside of the training

how to pick: optimize the generalizationn accessment

	no hyperparameter optimization	hyperparameter optimization
Large Dataset	**Train-test split**	**Train-validation-test split**
Small Dataset	**Cross-validation**	**Nested Cross-validation**

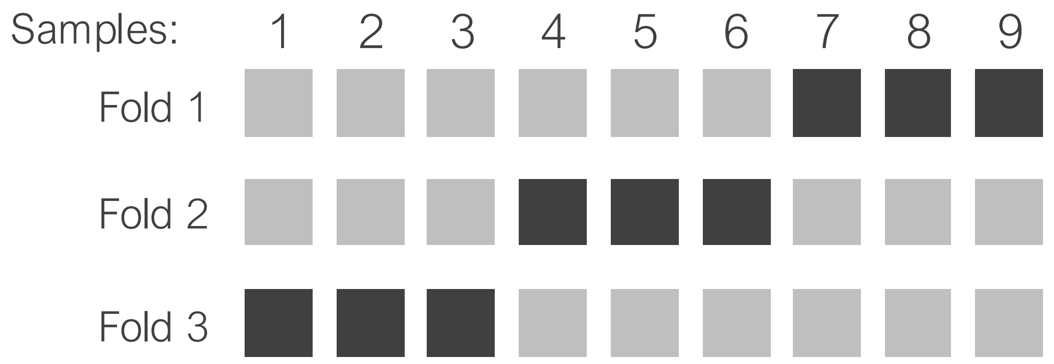
Note: hyperparameter optimization can be considered a form of model comparison

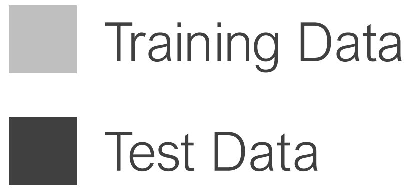
K-folds cross validation results in **k models**

How do we pick which to use?

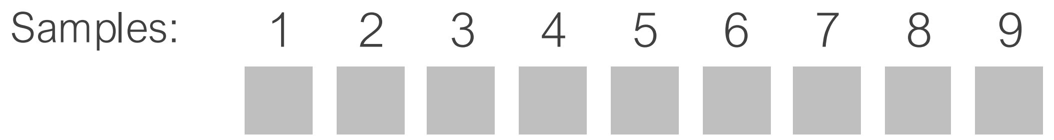
After performance has been validated, train on all the data you have before you apply the model in practice

1 **Performance evaluation**: Train your model K times, once for each fold





2 **Model application**: Once you've evaluated model performance and are ready apply the model then retrain the model on ALL of your data to prepare it for unseen data

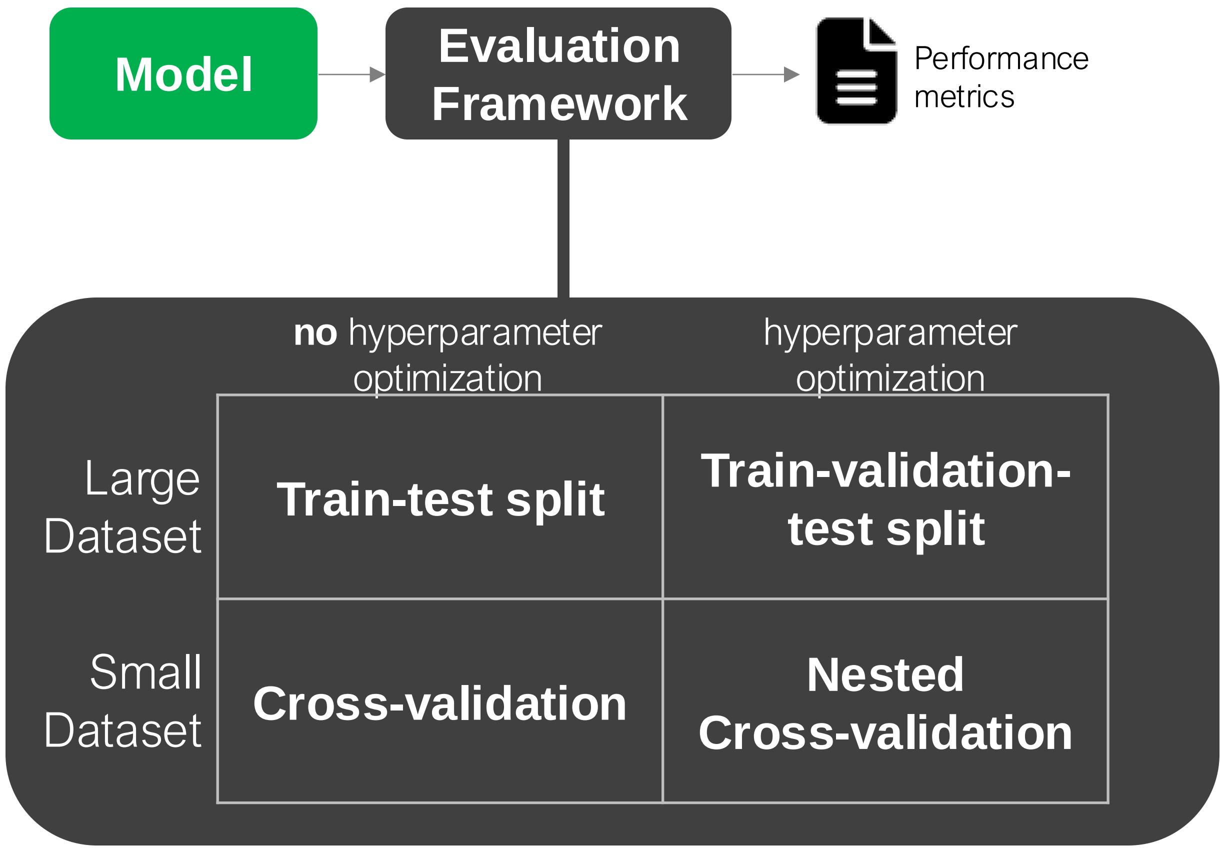


Training Data

(this is not a model evaluation step, but only when you're ready to apply in practice)

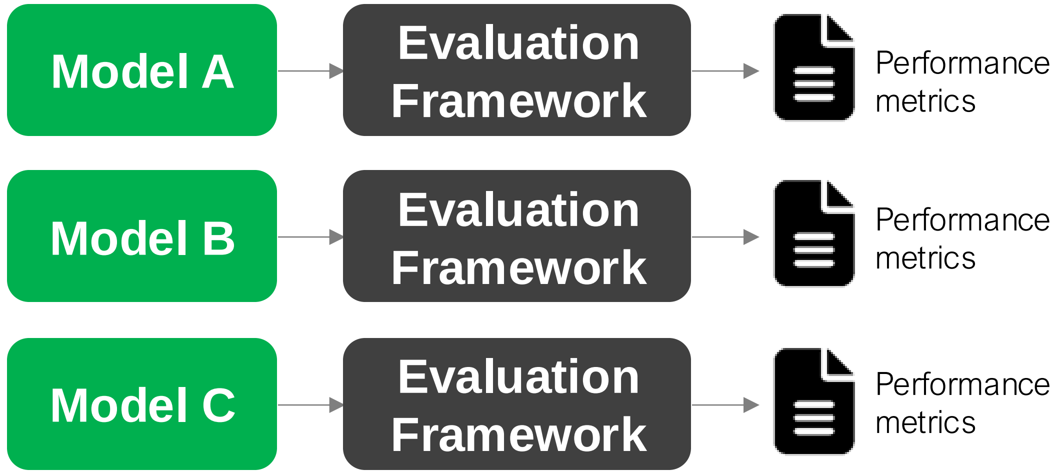
Basic ML Experimental Design

1 Generalization Performance Evaluation



	no hyperparameter optimization	hyperparameter optimization
Large Dataset	**Train-test split**	**Train-validation-test split**
Small Dataset	**Cross-validation**	**Nested Cross-validation**

2 Model Comparison

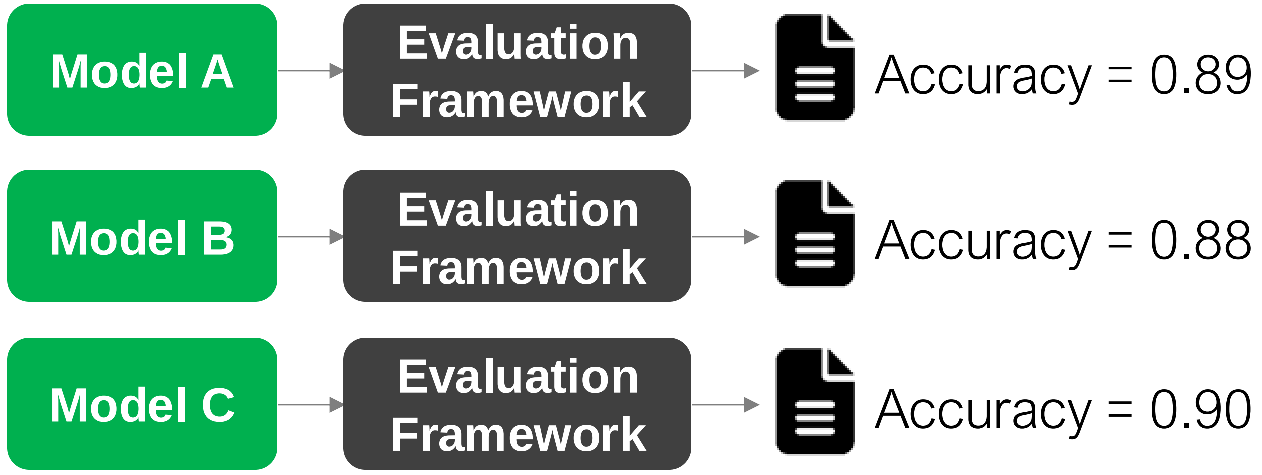


Compare models according to metrics

2 How do we use the metrics to **compare models**?

But is the performance "better"?

2 Model Comparison



Is Model C actually better than A and B?

or just noises make it look slightly better

Is the comparison fair?

Model	Learning Rate	Batch Size	Training data size	Test data size
A (ResNet)	0.001	16	500	100
B (Inception v3)	0.001	32	500	100
C (VGG16)	0.010	32	505	100

at least training and test data need to be exactly the same

What other questions do I need to ask here?

Are my metrics uncertain?

Model-based uncertainty
(Inherent stochasticity in the model training process)

- Order of training data
- Random initializations of model weights

Data-based uncertainty

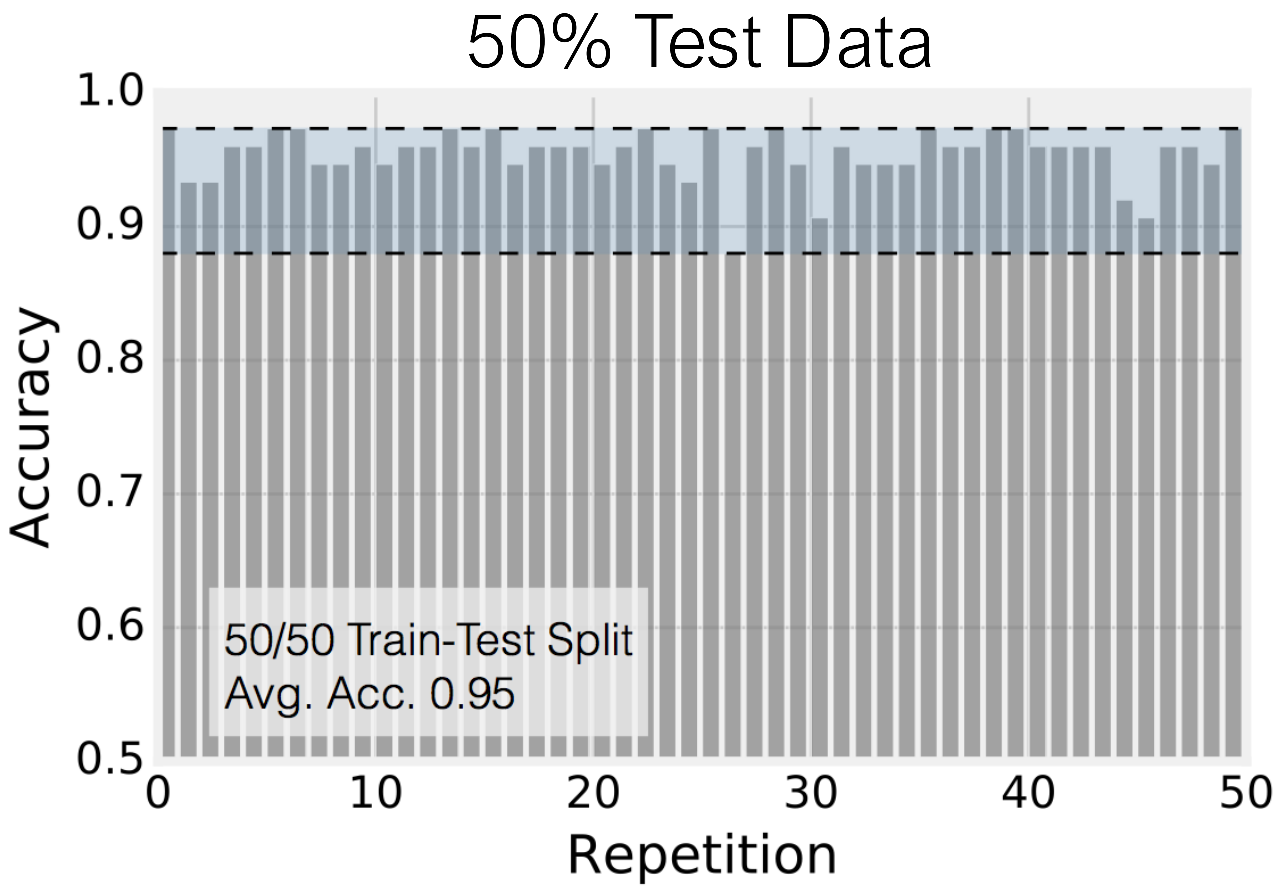
- Train/test split sampling
- Noise in the data
- Errors in the target variable

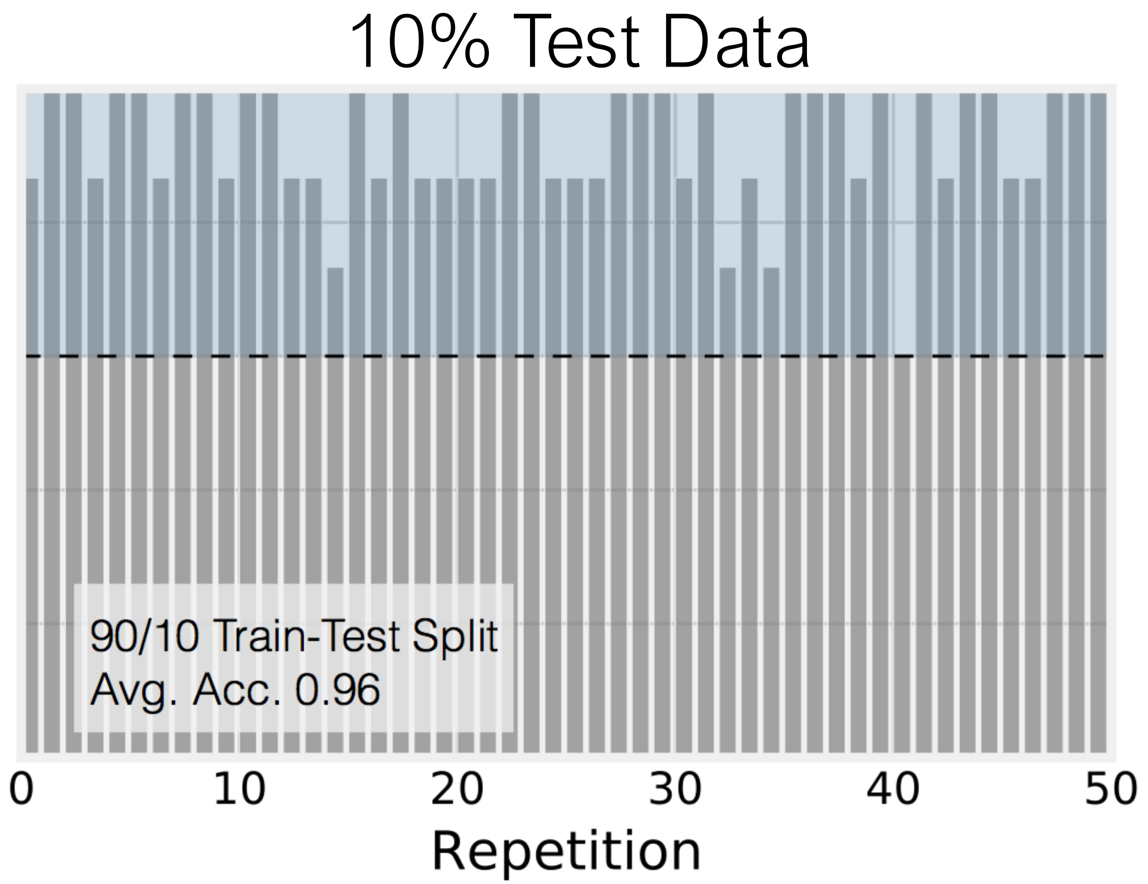
Solution: retrain the model multiple times, varying the variable components; evaluate the variation of the estimates

Different data splits produce different results

Each bar represents test performance for model trained on different random splits of data

Smaller test datasets lead to greater variance in the estimate of generalization performance

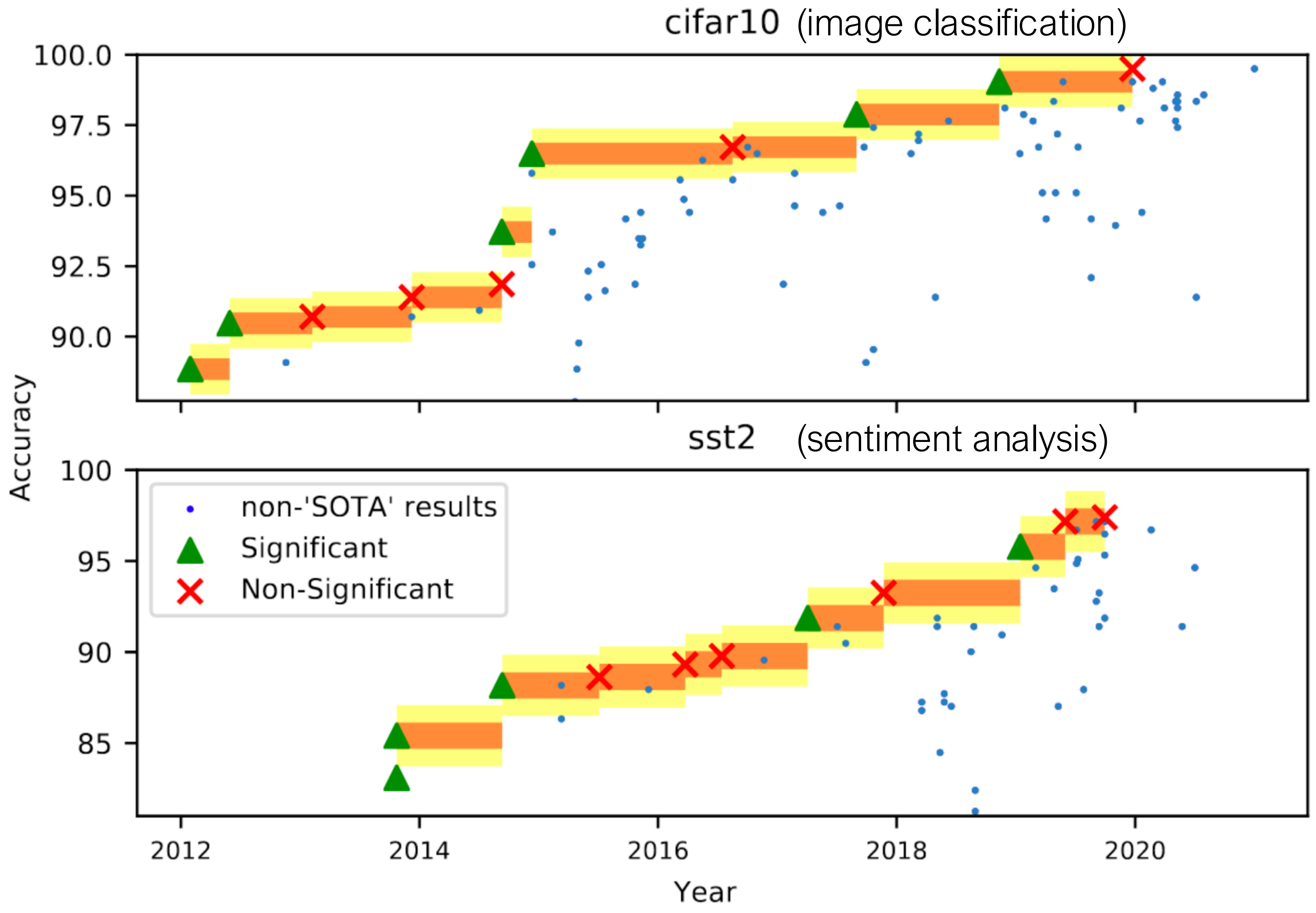




Images from Sebastian Raschka (https://sebastianraschka.com/blog/2016/model-evaluation-selection-part2.html)

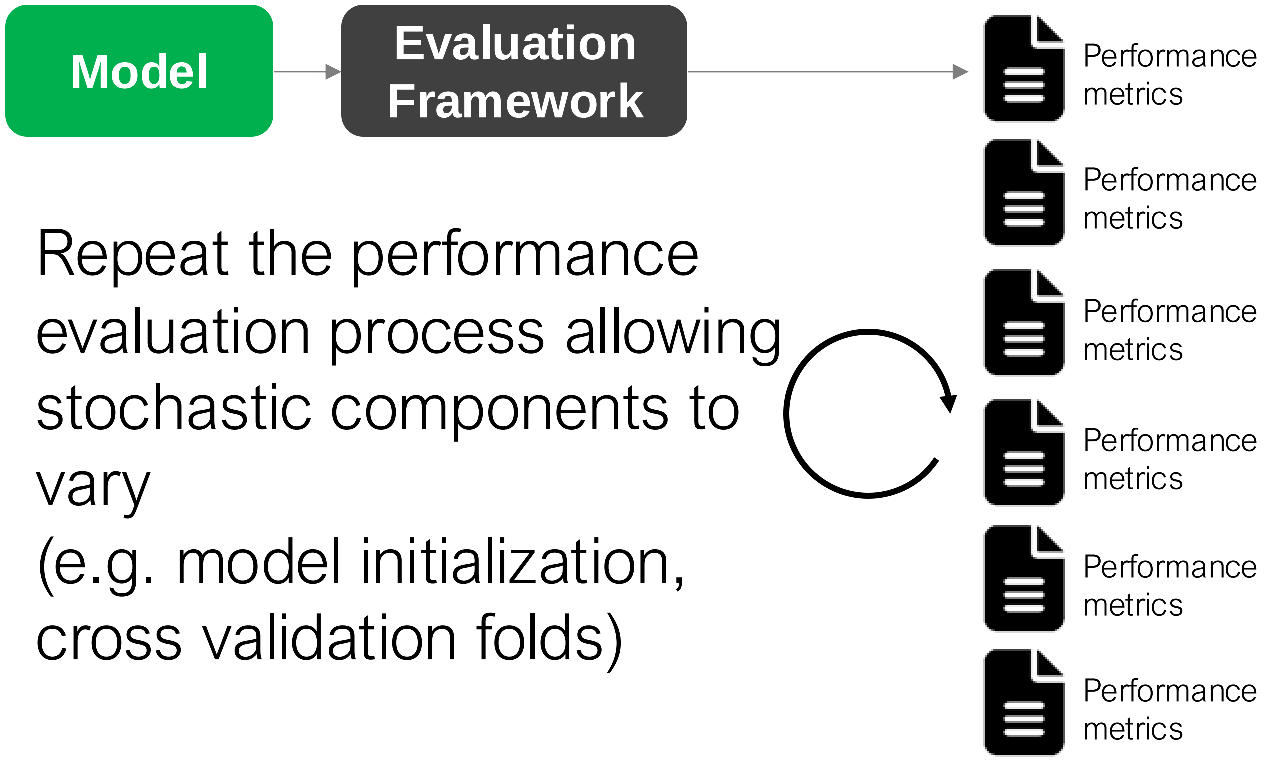
Iris dataset, KNN, k=3

Performance metrics have uncertainty



Bouthillier, X., Delaunay, P., Bronzi, M., Trofimov, A., Nichyporuk, B., Szeto, J., Sepah, N., Raff, E., Madan, K., Voleti, V., Kahou, S.E., Michalski, V., Serdyuk, D., Arbel, T., Pal, C., Varoquaux, G., Vincent, P., 2021. Accounting for Variance in Machine Learning Benchmarks. https://doi.org/10.48550/arXiv.2103.03098

Performance metric distributions



Repeat the performance evaluation process allowing stochastic components to vary
(e.g. model initialization, cross validation folds)

The model and evaluation framework are unchanged

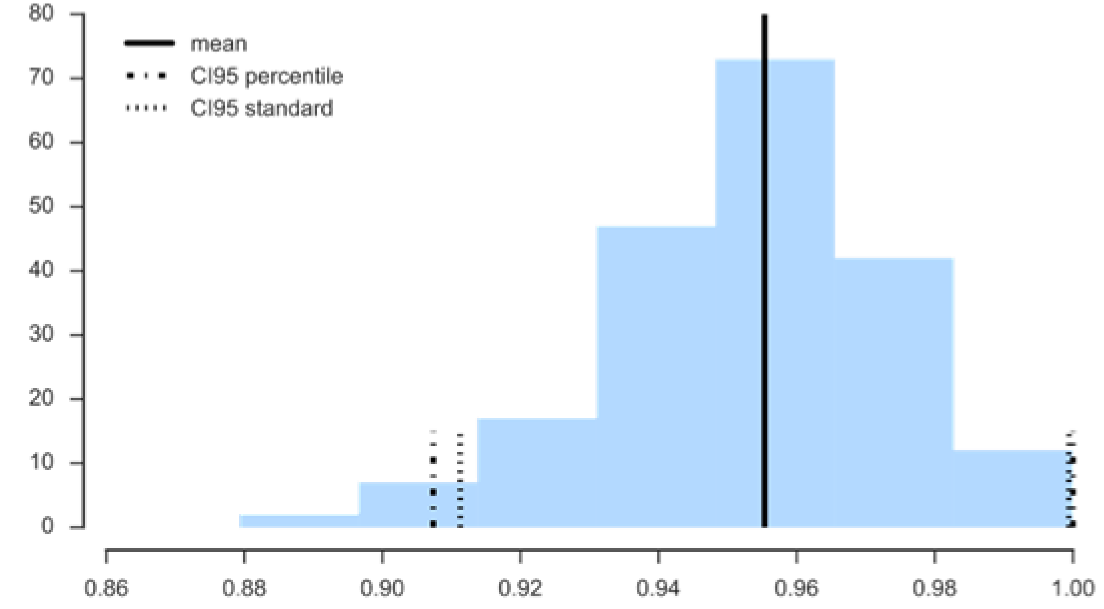


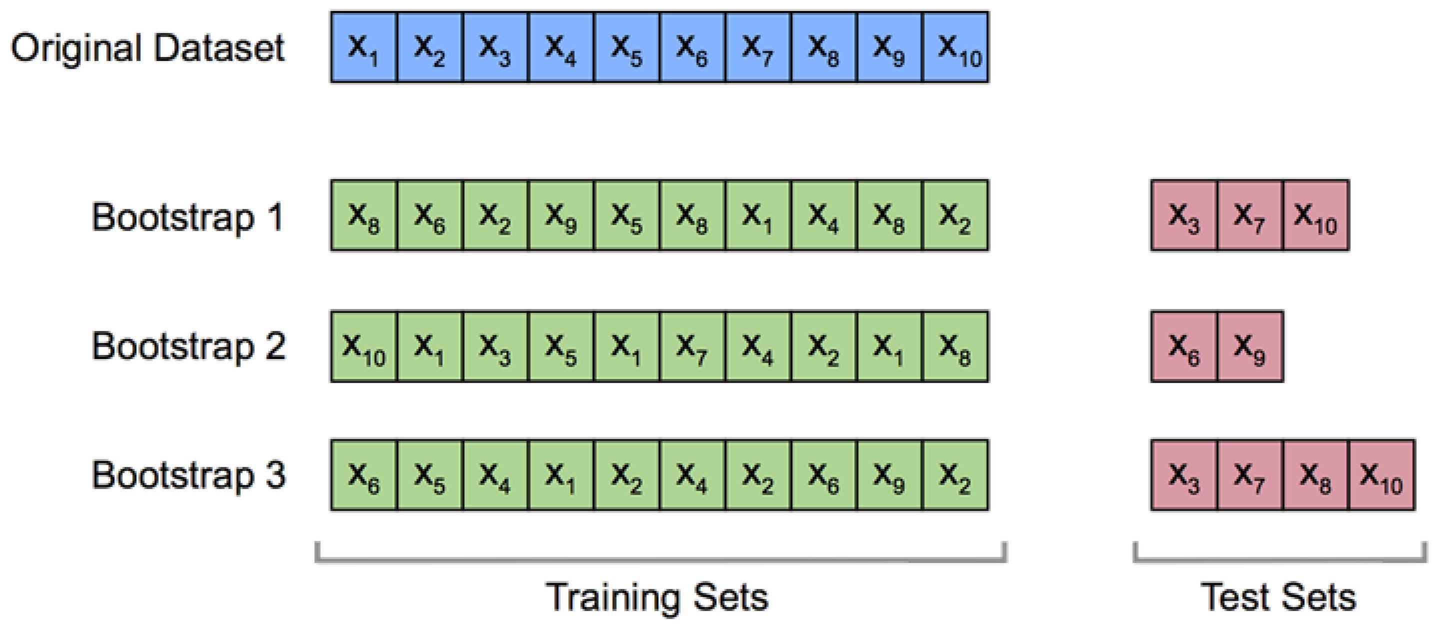
Image from Sebastian Raschka

Confidence intervals via bootstrap sampling

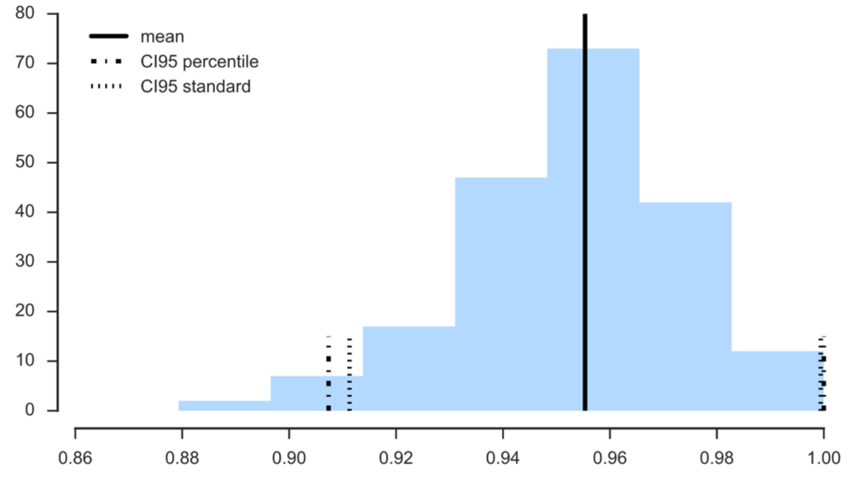
Sampling **with replacement**

Often used to **estimate standard errors and confidence intervals**

Integral part of model ensembles (i.e. bagging in random forests)



Original Dataset
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10}
Bootstrap 1
x_8 x_6 x_2 x_9 x_5 x_8 x_1 x_4 x_8 x_2
x_3 x_7 x_{10}
Bootstrap 2
x_{10} x_1 x_3 x_5 x_1 x_7 x_4 x_2 x_1 x_8
x_6 x_9
Bootstrap 3
x_6 x_5 x_4 x_1 x_2 x_4 x_2 x_6 x_9 x_2
x_3 x_7 x_8 x_{10}
Training Sets
Test Sets



Images from Sebastian Raschka

3 Other performance analyses?

Sensitivity / robustness analysis - vary one model hyperparameter and evaluate its effect on performance

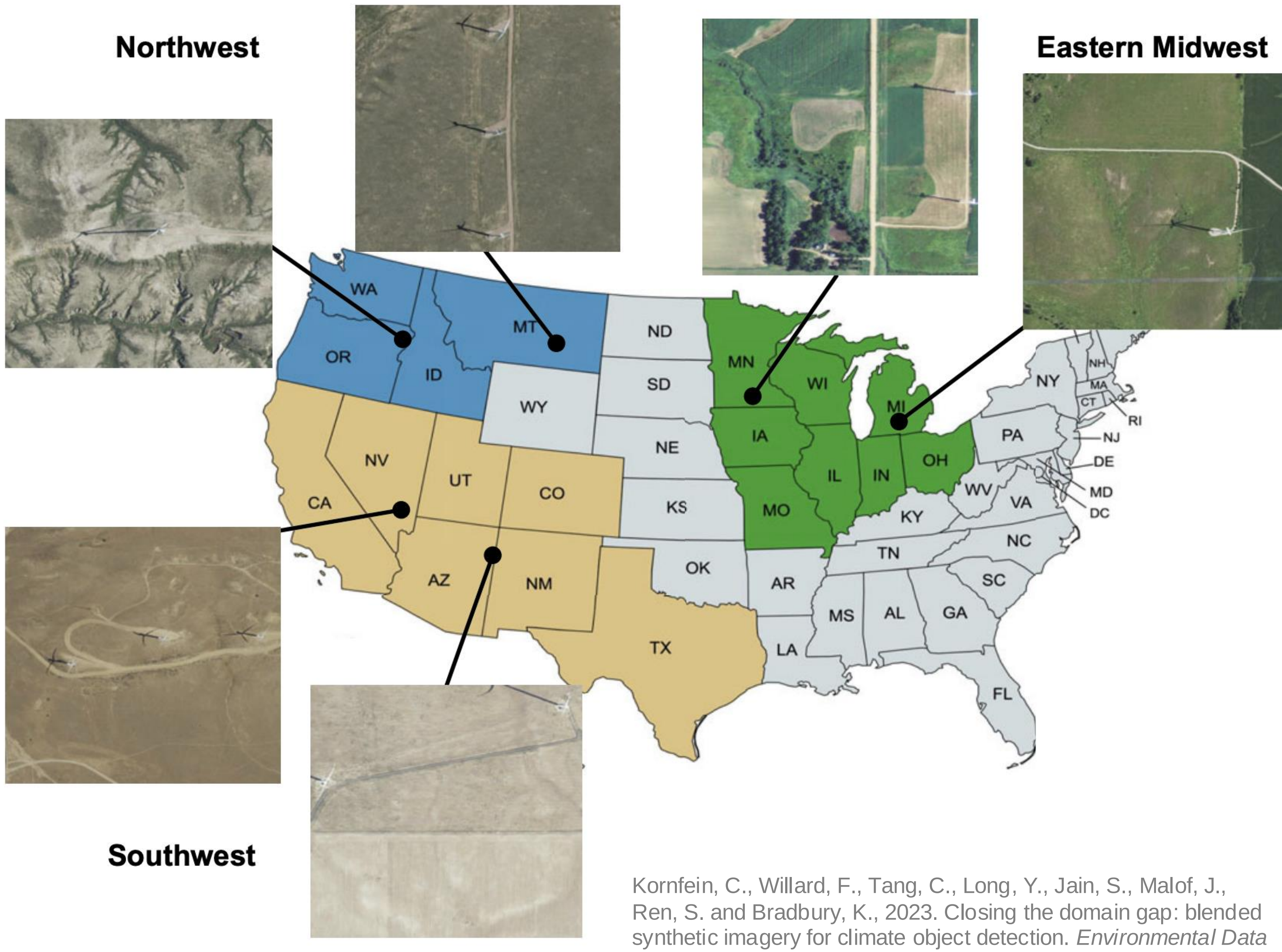
Ablation study – remove one or more aspect of a model and compare performance before and after

Example Experimental Design

- Enables us to answer questions, typically related to comparing models (e.g. does Model A or Model B perform better)
- Requires that you ONLY vary one characteristic at a time (e.g. model architecture, hyperparameters, training data)
  (unless you are actively looking to estimate the variation induced by that characteristic)
- Requires that you control for uncertainty (in the model AND in the data)

Experimental Design Example

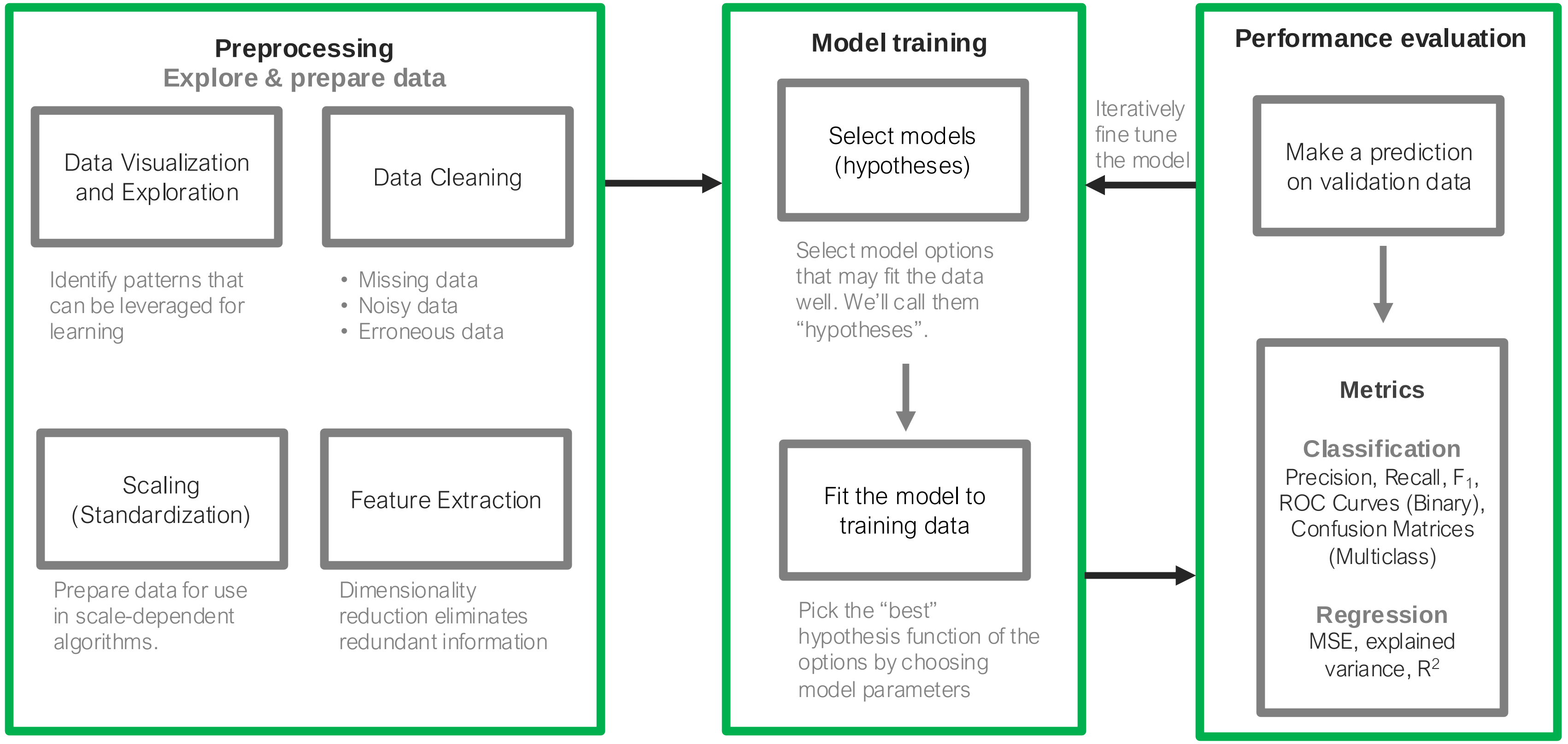
How well do models transfer to new domains?



Kornfein, C., Willard, F., Tang, C., Long, Y., Jain, S., Malof, J., Ren, S. and Bradbury, K., 2023. Closing the domain gap: blended synthetic imagery for climate object detection. *Environmental Data Science*, *2*, p.e39.

Experimental Design

What do we need to check to make sure that there is only one change between experimental conditions?



Preprocessing
Explore & prepare data
Data Visualization and Exploration
Identify patterns that can be leveraged for learning
Data Cleaning
• Missing data
• Noisy data
• Erroneous data
Scaling (Standardization)
Prepare data for use in scale-dependent algorithms.
Feature Extraction
Dimensionality reduction eliminates redundant information
Model training
Select models (hypotheses)
Select model options that may fit the data well. We'll call them "hypotheses".
Fit the model to training data
Pick the "best" hypothesis function of the options by choosing model parameters
Iteratively fine tune the model
Performance evaluation
Make a prediction on validation data
Metrics
Classification
Precision, Recall, F1, ROC Curves (Binary), Confusion Matrices (Multiclass)
Regression
MSE, explained variance, R2

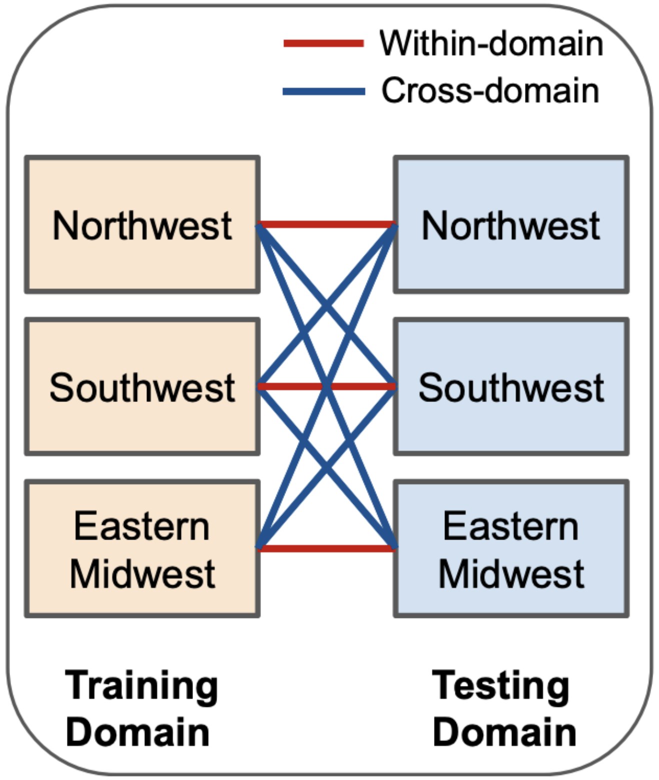
- Train/validation/test split
  - How it was split
  - The split itself
- Data imputation
- Feature selection
- Scaling
- Dimensionality reduction
- Feature transformations

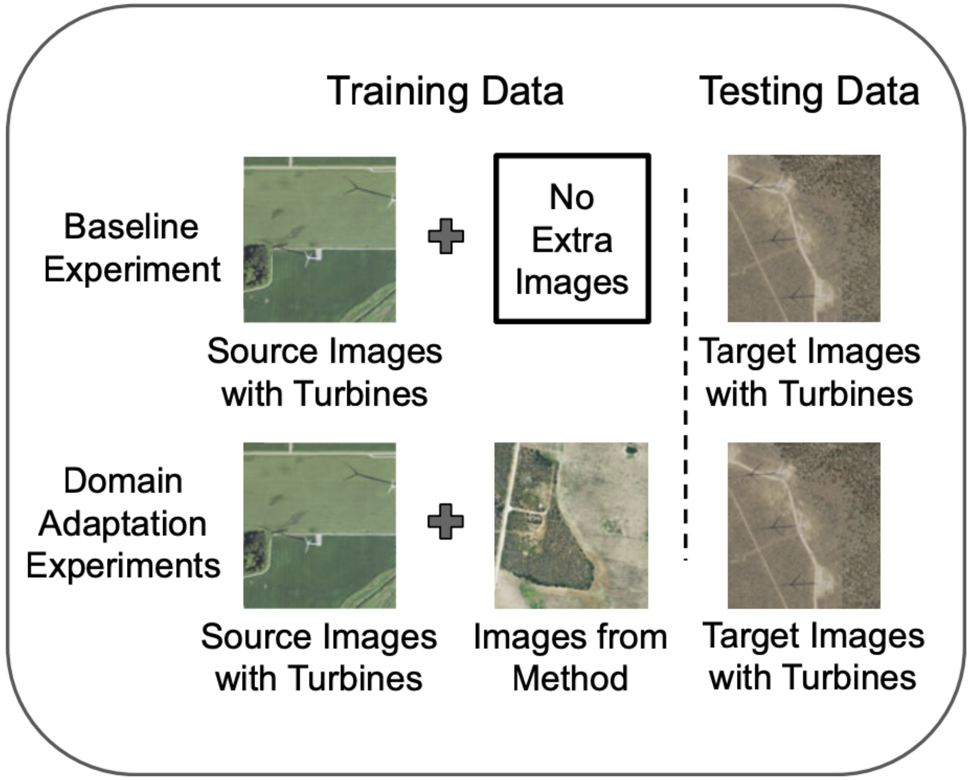
- Model architecture
- Hyperparameters (e.g. learning rate, batch size, stopping criterion, choice of regularization)

Validation / test data should be the same for each model evaluation instance

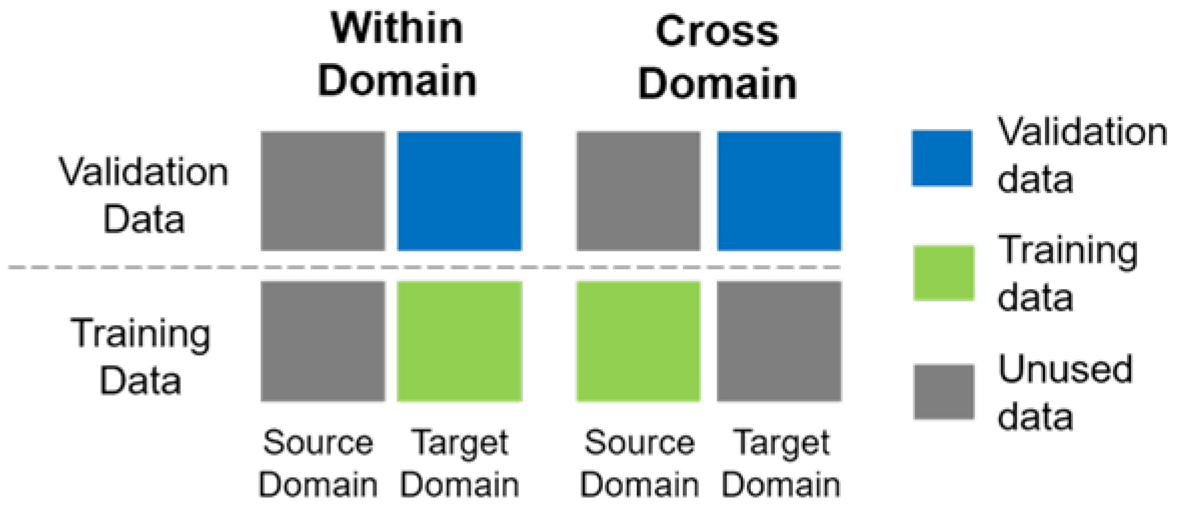
Experimental Design Example

Experimental Design
The only change between experimental conditions was the content of the training data




Training Data
Testing Data
Baseline Experiment
No Extra Images
Source Images with Turbines
Target Images with Turbines
Domain Adaptation Experiments
Source Images with Turbines
Images from Method
Target Images with Turbines

Training Procedure

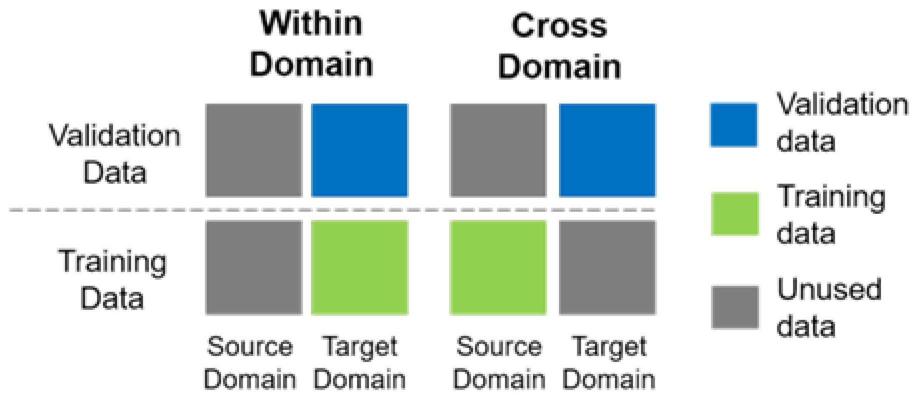


Kornfein, C., Willard, F., Tang, C., Long, Y., Jain, S., Malof, J., Ren, S. and Bradbury, K., 2023. Closing the domain gap: blended synthetic imagery for climate object detection. *Environmental Data Science*, *2*, p.e39.

Kornfein, C., Willard, F., Tang, C., Long, Y., Jain, S., Malof, J., Ren, S. and Bradbury, K., 2022. "Closing the Domain Gap–Blended Synthetic Imagery for Climate Object Detection." In 2022 Neural Information Processing Systems (NeurIPS) Climate Change AI Workshop.

Experimental Design Example

How well do models transfer to new domains?



Results shown in units of mean average precision (mAP)

Source domain	Target domain	Baseline $\pm 2\sigma$	Adding synthetic $\pm 2\sigma$	Average improvement%
EM	EM	0.822 ± 0.067	$\mathbf{0.919} \pm 0.016$	11.8%
NE		0.567 ± 0.019	$\mathbf{0.698} \pm 0.038$	23.1%
NW		0.358 ± 0.061	$\mathbf{0.424} \pm 0.114$	18.4%
SW		0.449 ± 0.160	$\mathbf{0.626} \pm 0.180$	39.4%
EM	NE	0.387 ± 0.031	$\mathbf{0.487} \pm 0.114$	25.8%
NE		0.812 ± 0.028	$\mathbf{0.842} \pm 0.013$	3.7%
NW		0.666 ± 0.061	$\mathbf{0.709} \pm 0.049$	6.5%
SW		0.412 ± 0.045	$\mathbf{0.521} \pm 0.089$	26.5%
EM	NW	0.485 ± 0.064	$\mathbf{0.521} \pm 0.054$	7.4%
NE		0.746 ± 0.018	$\mathbf{0.770} \pm 0.032$	3.2%
NW		0.895 ± 0.071	$\mathbf{0.915} \pm 0.023$	2.2%
SW		0.659 ± 0.111	$\mathbf{0.693} \pm 0.066$	5.2%
EM	SW	0.093 ± 0.016	$\mathbf{0.113} \pm 0.008$	20.9%
NE		0.121 ± 0.029	$\mathbf{0.134} \pm 0.030$	10.7%
NW		0.149 ± 0.029	$\mathbf{0.197} \pm 0.024$	32.2%
SW		0.566 ± 0.035	$\mathbf{0.568} \pm 0.104$	0.4%
Within-domain average		0.774 ± 0.050	$\mathbf{0.811} \pm 0.039$	4.8%
Cross-domain average		0.425 ± 0.054	$\mathbf{0.491} \pm 0.067$	15.7%

Bolded items represent best model (comparing a baseline model and experimental condition model)

Standard deviations contextualize model uncertainty
- This represents retraining the model multiple times to measure performance variability
- For a result to be significant, the performance change needs to be large enough to be unlikely to be due to model variability

Kornfein, C., Willard, F., Tang, C., Long, Y., Jain, S., Malof, J., Ren, S. and Bradbury, K., 2023. Closing the domain gap: blended synthetic imagery for climate object detection. *Environmental Data Science*, *2*, p.e39.

Kornfein, C., Willard, F., Tang, C., Long, Y., Jain, S., Malof, J., Ren, S. and Bradbury, K., 2022. "Closing the Domain Gap–Blended Synthetic Imagery for Climate Object Detection." In 2022 Neural Information Processing Systems (NeurIPS) Climate Change AI Workshop.

Modeling Considerations

Accuracy (and techniques to measure it)

Computational Efficiency

Interpretability

Computational Efficiency

Measure of how an algorithm's run time (or space requirements) grows as the input size grows

Complexity of making predictions with kNN
(compare an unseen sample to the training samples)

Assume we have n = 10,000, p = 2

The Euclidean distance between $\begin{bmatrix} x_{1,1} \\ x_{1,2} \end{bmatrix}$ and $\begin{bmatrix} x_{2,1} \\ x_{2,2} \end{bmatrix}$ can be measured as:

$$\sqrt{(x_{2,1} - x_{1,1})^2 + (x_{2,2} - x_{1,2})^2}$$

That's two (p) distinct sets of operations dependent on the data
We repeat that n times – once for each sample in the training dataset

$$O(np)$$

Computational Efficiency

Training time efficiency?

Test time efficiency?

How do each change with the size of our data?

Interpretability

Transparency (can I tell how the model works)

- **Simulatability**: can I contemplate the whole model at once?
- **Decomposability**: is there an intuitive explanation for each part of the model?
  (e.g. all patients with diastolic blood pressure over 150)

Explainability (post-hoc explanations)

Visualization, local explanations, explanations by example

(e.g. this tumor is classified as malignant because to the model it looks a lot like these other tumors)

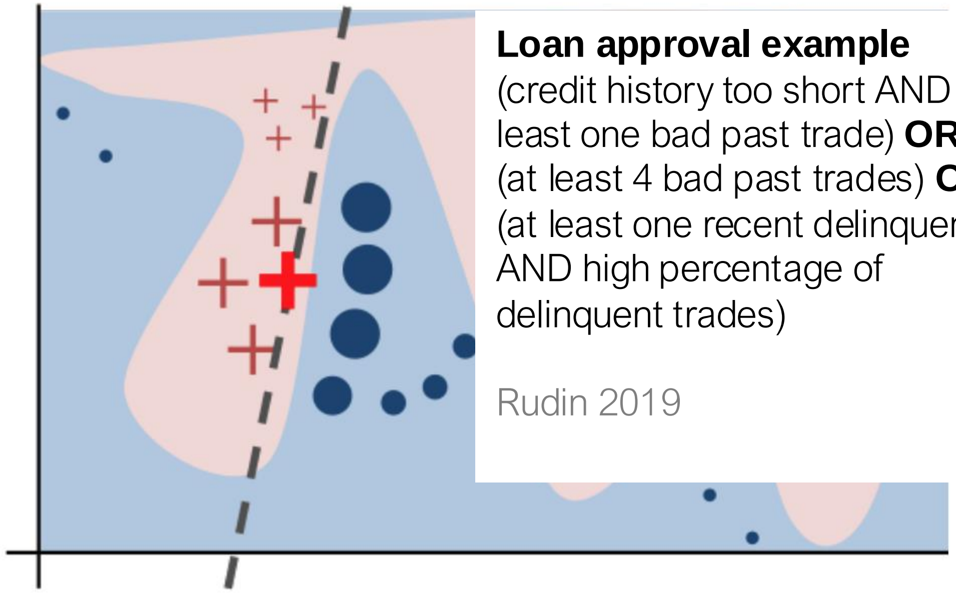
Lipton, Zachary C. "The Mythos of Model Interpretability: In Machine Learning, the Concept of Interpretability Is Both Important and Slippery." Queue 16, no. 3 (2018): 31–57.

Recidivism prediction algorithm

Performance as good as a black box model with 130+ factors; might include socio-economic info; expensive (software license); within software used in US justice system

IF	age between 18-20 and sex is male	THEN predict arrest (within 2 years)
ELSE IF	age between 21-23 and 2-3 prior offences	THEN predict arrest
ELSE IF	more than three priors	THEN predict arrest
ELSE	predict no arrest	

Rudin, Cynthia. "Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead." Nature Machine Intelligence 1, no. 5 (2019): 206–15.



Loan approval example
(credit history too short AND at least one bad past trade) **OR** (at least 4 bad past trades) **OR** (at least one recent delinquency AND high percentage of delinquent trades)

Rudin 2019

Ribeiro, Marco Tulio, Sameer Singh, and Carlos Guestrin. "Model-Agnostic Interpretability of Machine Learning." ArXiv Preprint ArXiv:1606.05386, 2016.

For further reading…

Raschka, Sebastian. “Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning.” *ArXiv:1811.12808 [Cs, Stat]*, November 10, 2020. http://arxiv.org/abs/1811.12808.

Kohavi, Ron. “A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection.” In *IJCAI*, 14:1137–45. Montreal, Canada, 1995. (link)