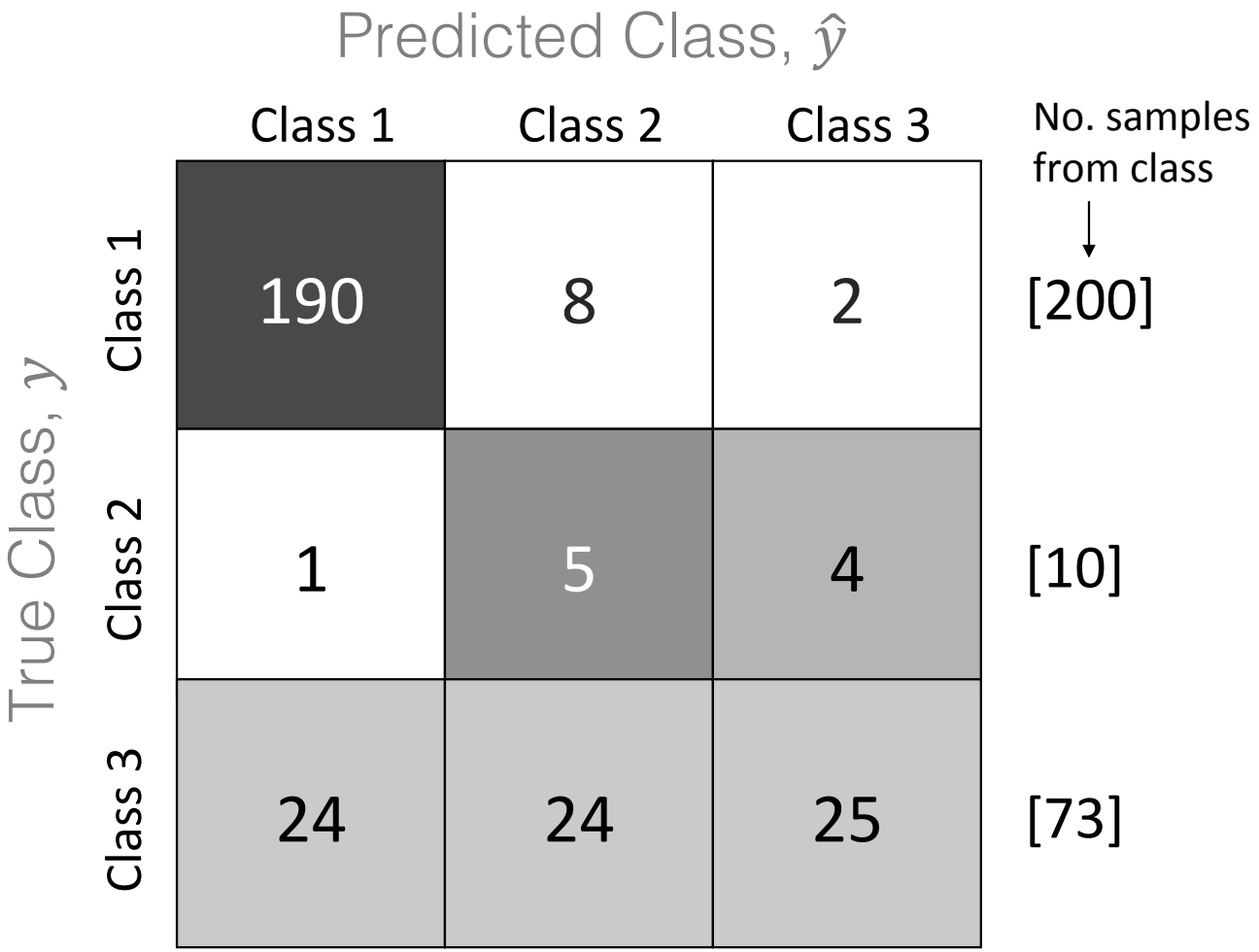
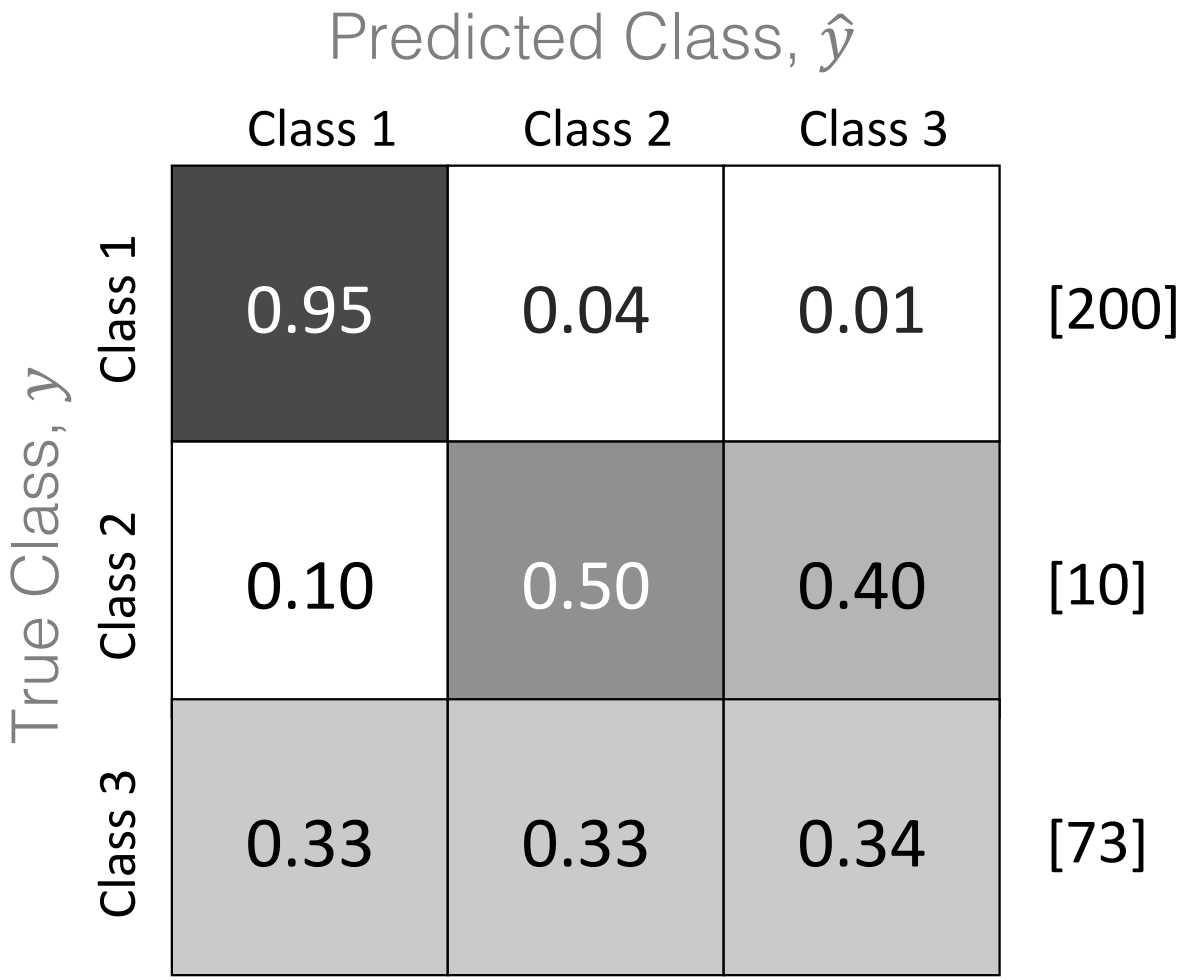
# Evaluating Performance II

# Multiclass Classification: Confusion Matrix




confusion matrix with number of samples




confusion matrix with probabilities

# $F_1$-score

$$F_1 = 2\frac{1}{\frac{1}{\text{recall}} + \frac{1}{\text{precision}}}$$

Harmonic mean of precision and recall

$$= 2\frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Generally:

$$F_\beta = (1 + \beta^2)\frac{\text{precision} \cdot \text{recall}}{\beta^2 \cdot \text{precision} + \text{recall}}$$

$\beta$ controls the relative weight of precision/recall

# Multiclass $F_1$

**Micro-average**: Calculate precision and recall metrics globally by counting the total true positives, false negatives, and false positives
(average for the whole dataset)

**Macro-average**: Use the average precision and recall for each class label
(average of class-averages)

# Modeling Considerations

Accuracy

Computational Efficiency

Interpretability

# Accuracy

# Supervised Learning Performance Evaluation

## Regression

## Classification

### Binary

Receiver Operating Characteristic (ROC) curves

### Multiclass

Confusion matrices

## Common Metrics

- Mean squared error (MSE)
- Mean absolute error (MAE)
- $R^2$, coefficient of determination

- Classification accuracy
- True positive rate
- False positive rate
- Precision
- $F_1$ Score
- Area under the ROC curve (AUC)

- Classification accuracy
- Micro-averaged $F_1$ Score
- Macro-averaged $F_1$ Score

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

…BUT, they're invalid (i.e. reflect generalization performance) if:

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

2. The test 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)

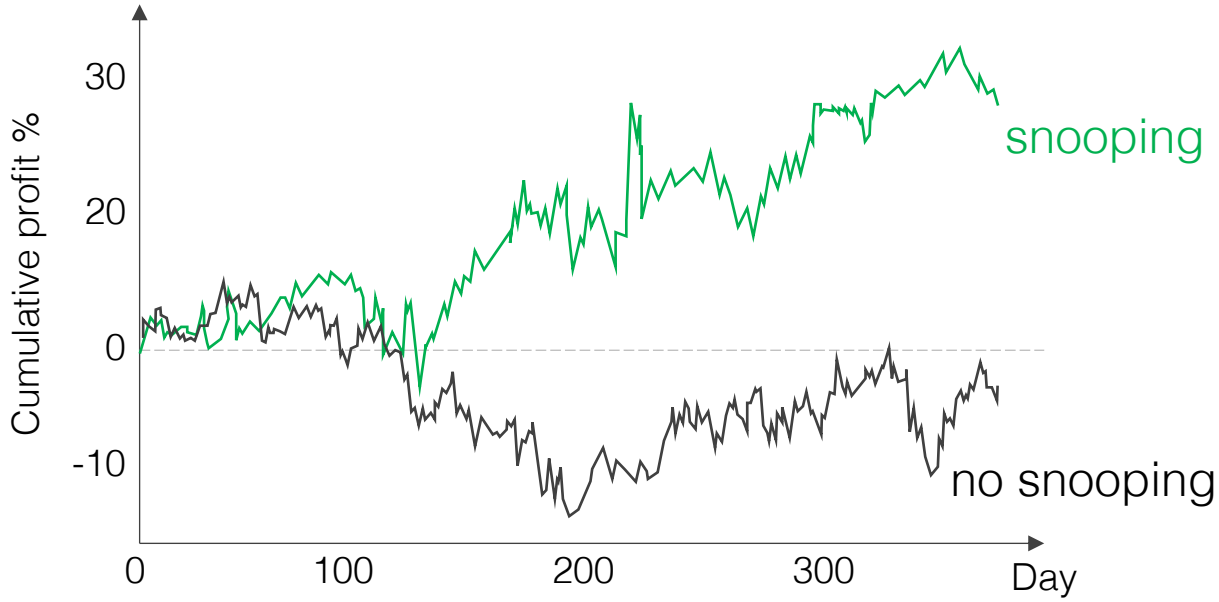
2. You take your historical data, normalize it, then split it randomly into a training and test set

   **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

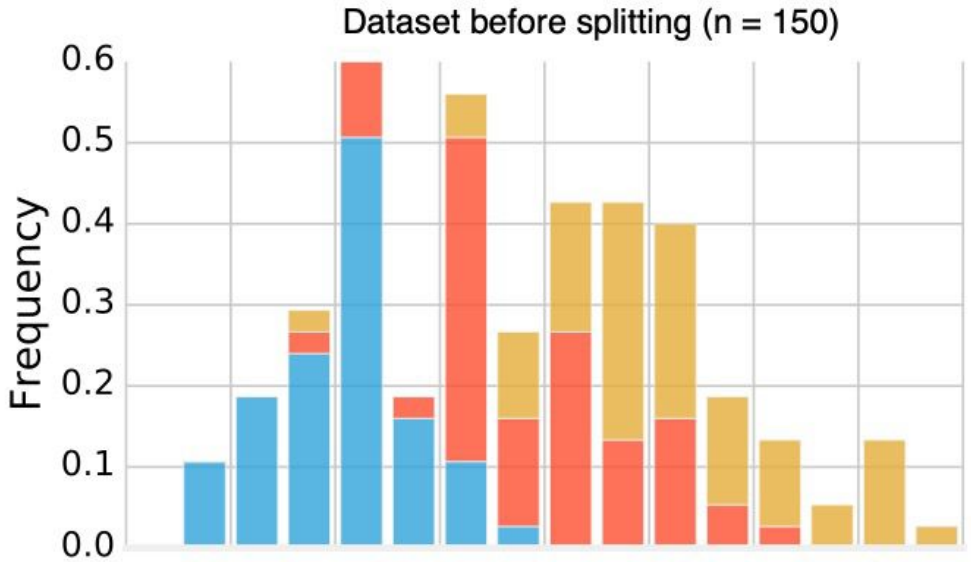
# Data snooping / leakage

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**?





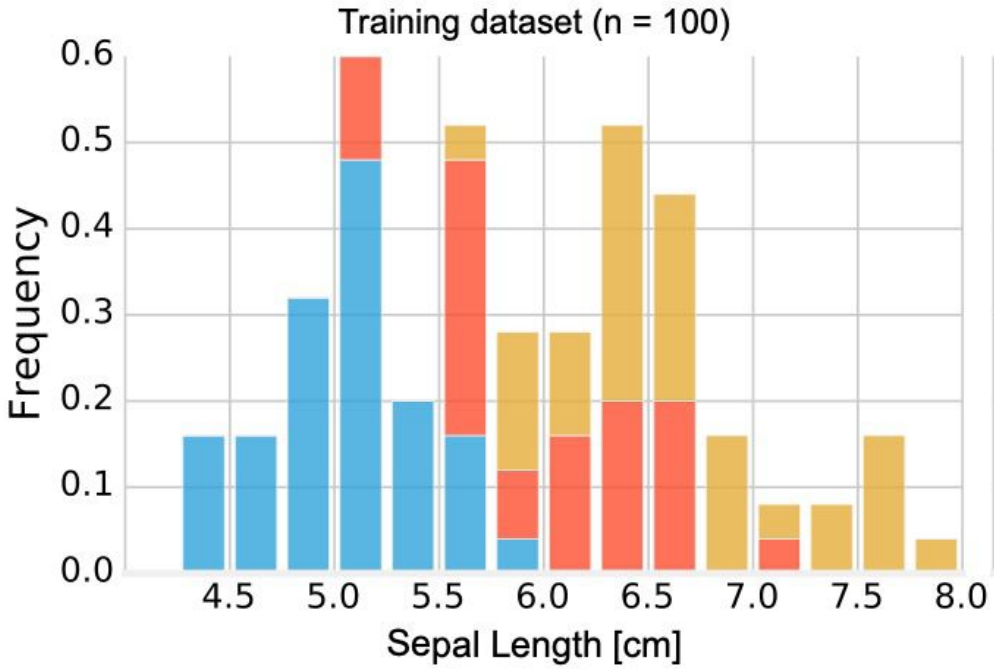

Setosa
Versicolor
Virginica

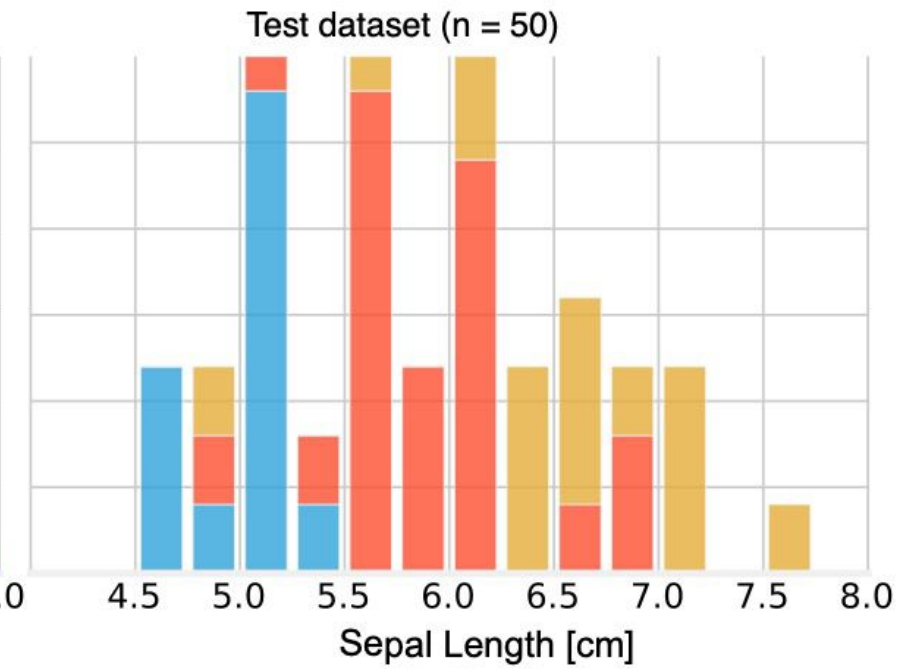
**All**: 50 Setosa, 50 Versicolor, 50 Virginica

**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**

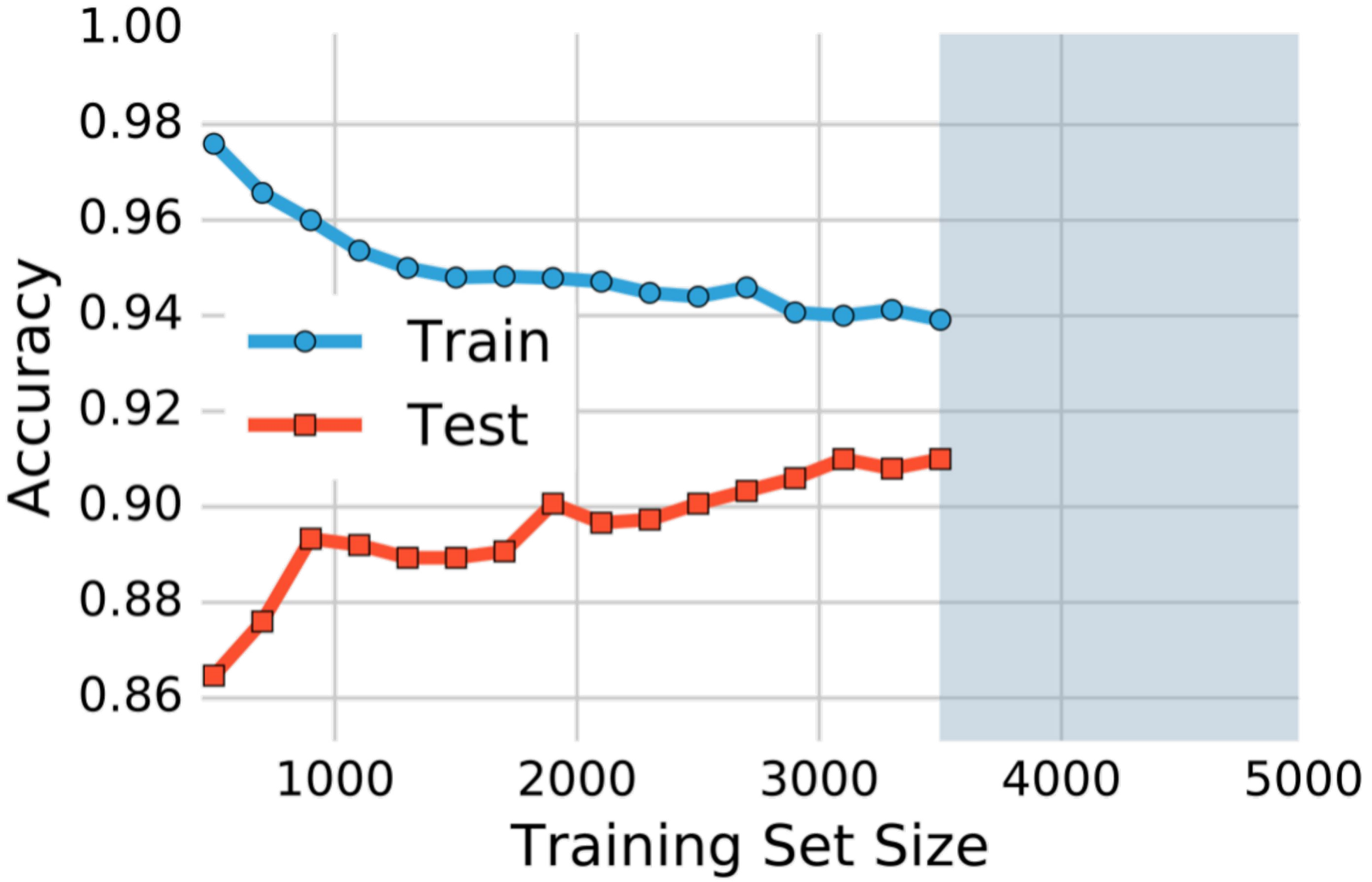
Images from Sebastian Raschka (https://sebastianraschka.com/blog/2016/model-evaluation-selection-part1.html)

# 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



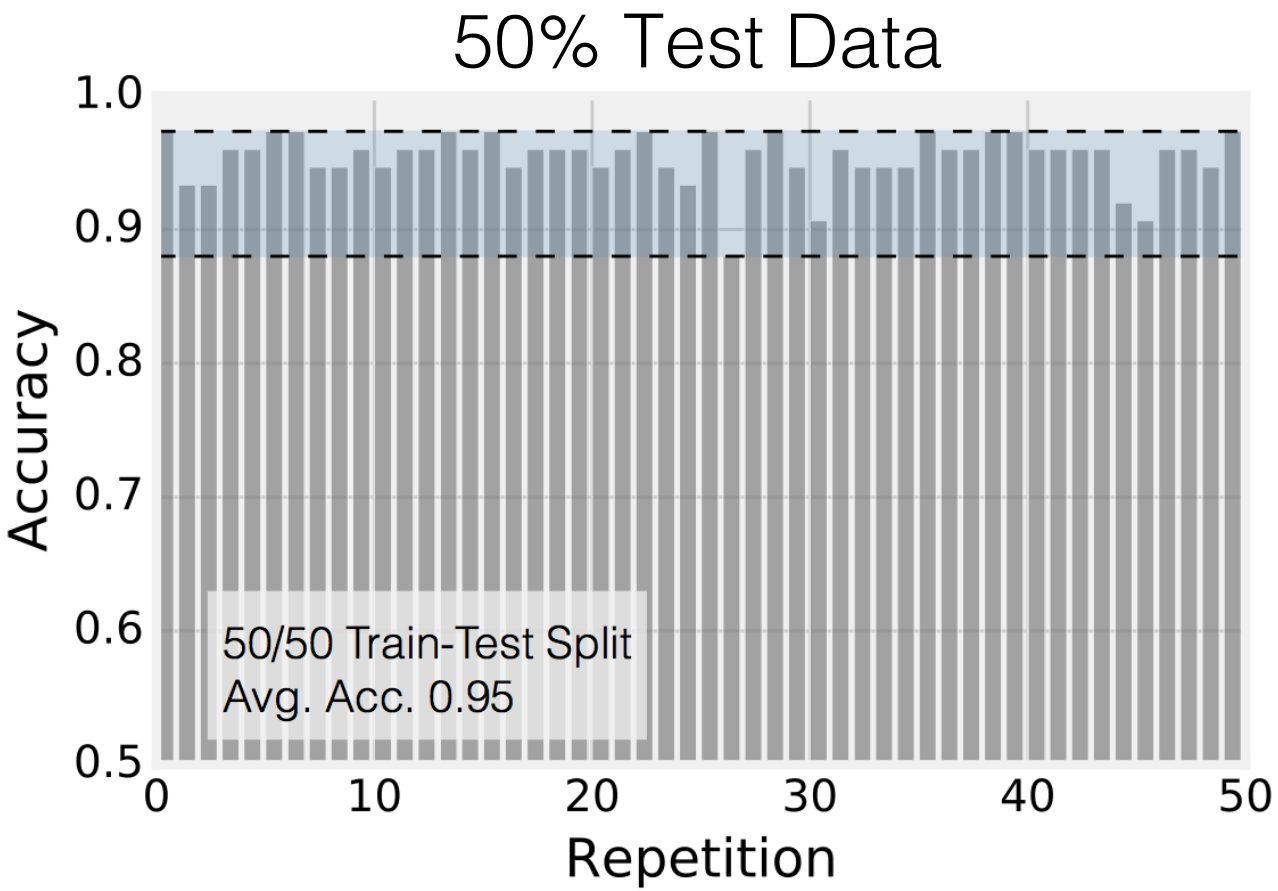


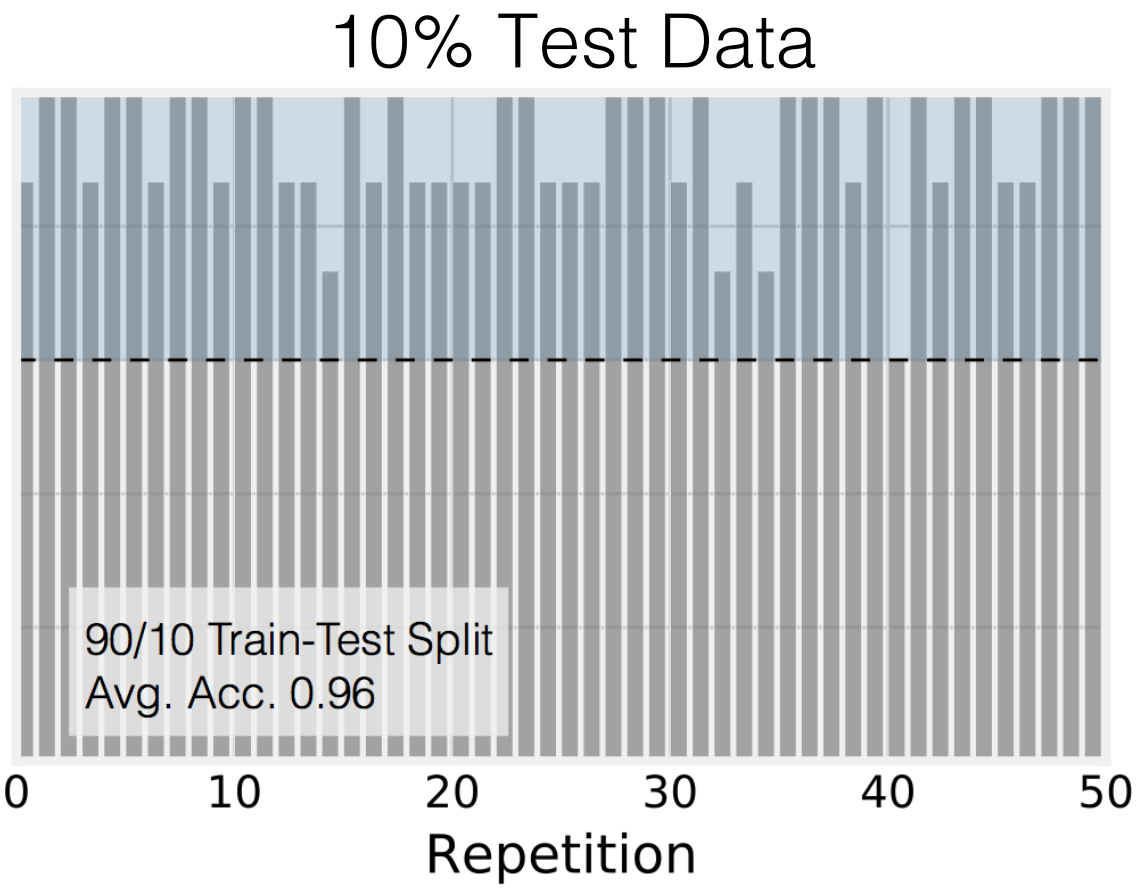

Images from Sebastian Raschka (https://sebastianraschka.com/blog/2016/model-evaluation-selection-part2.html)

# Size of test set for train/test splits

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)

# How do we use the metrics to evaluate performance?

# Train-Test Split

Learning model parameters and evaluating performance

Commonly 70/30 or 80/20


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

# Training, Validation, Test Split

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


Used for model training / fitting

Used to optimize hyperparameters

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

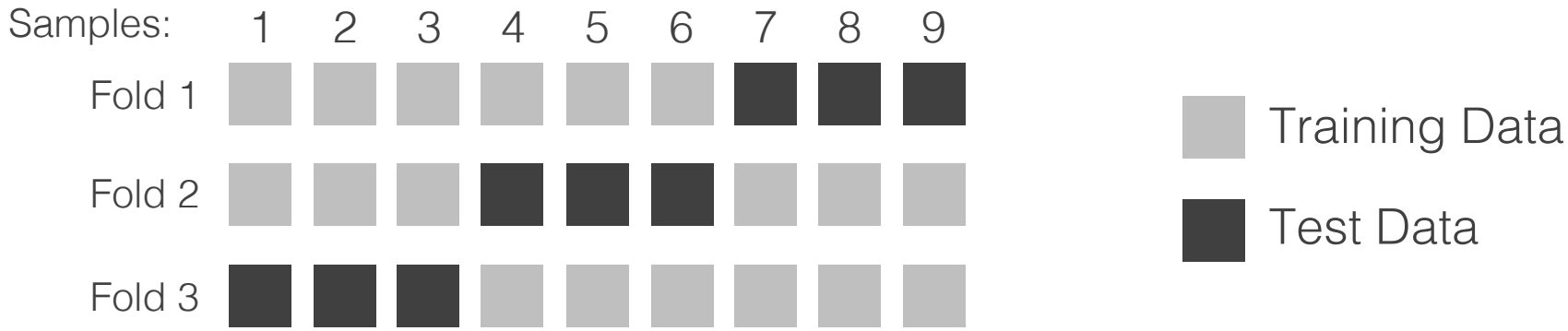
**Hyperparameters**: parameters that control how your algorithm learns; typically set before training begins (e.g. k in KNN, learning rate, etc.)

# What if you have a small dataset?

# K-folds cross-validation

**K-fold cross validation  K = 3**

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




Typical choices for k are 5 or 10

Average performance metrics across the splits

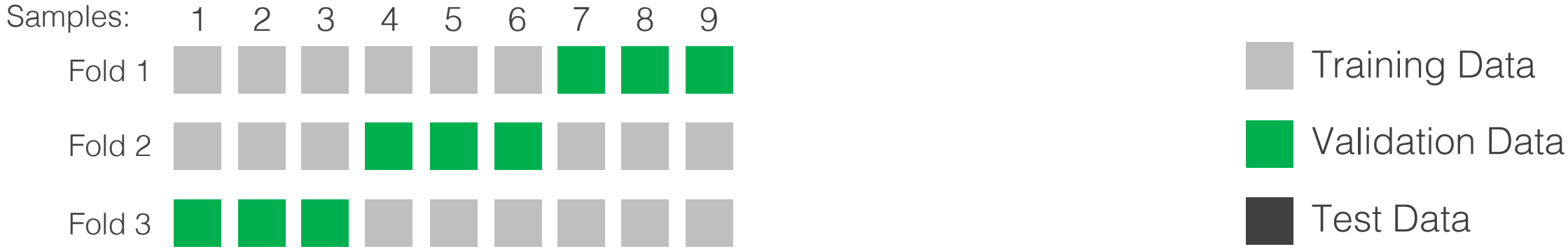
If k = N (number of samples): Leave-one-out cross validation

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?

# Cross-validation with hyperparameters

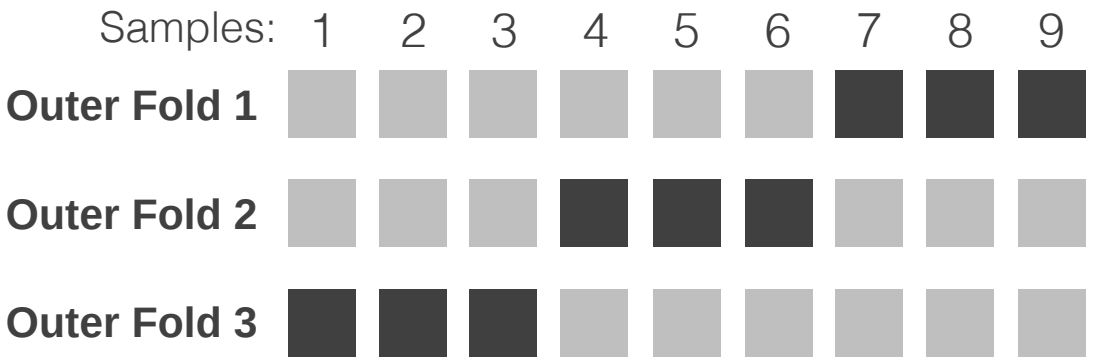
1. Repeatedly fit your model to your K folds. Each iteration try different hyperparameters



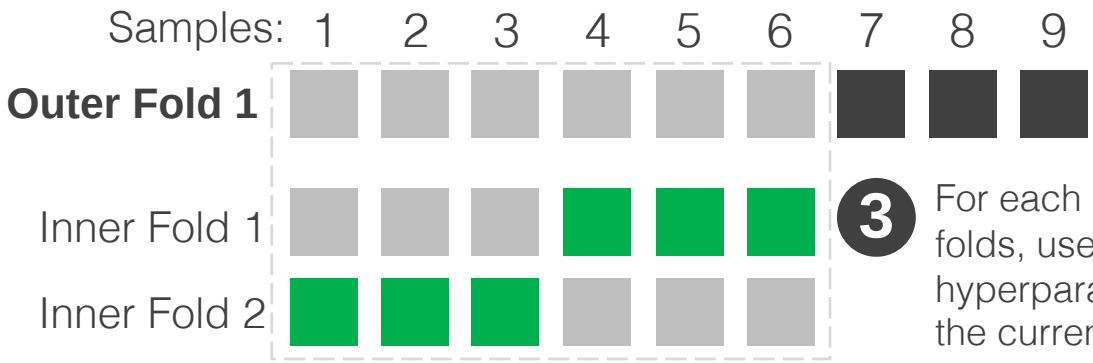

2. Using the best-performing hyperparameters from (a), train on all training data and evaluate performance on the test data


But this uses a small test set…
Variance will be high…

# Nested cross-validation with hyperparameters




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




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

3. For each of the K inner folds, use the hyperparameters from the current iteration to train the model, then evaluate performance on validation data

4. Repeat steps (2) and (3) for the remaining outer folds


Training Data

Validation Data

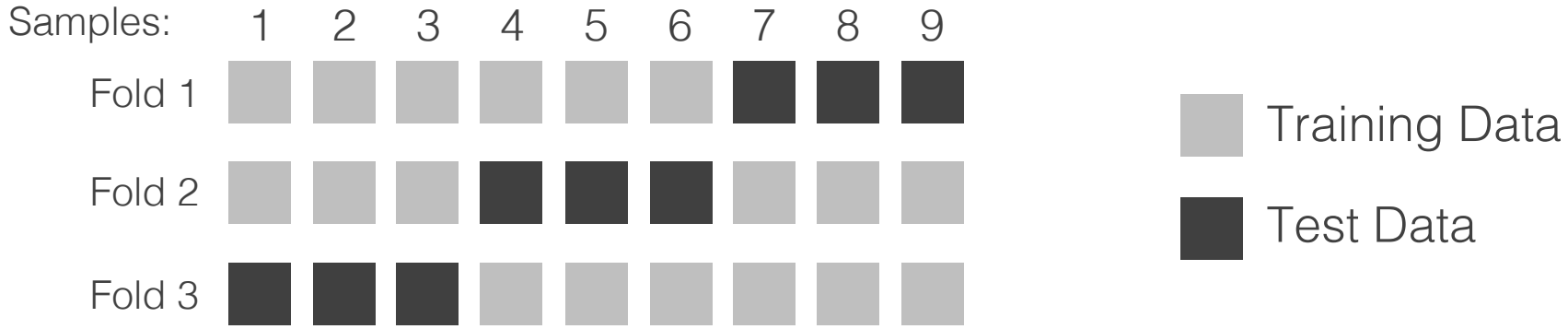
Test Data

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

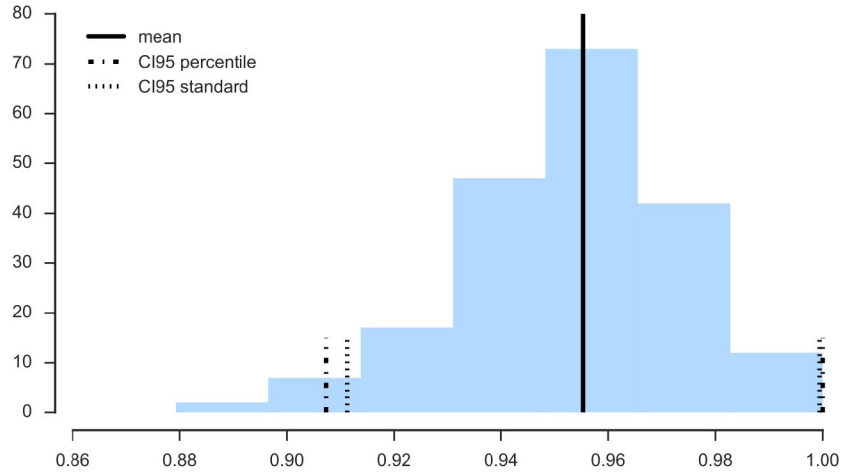

(this is not a model evaluation step, but only when you're ready to apply in practice)

# 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)




Images from Sebastian Raschka

**When to use each technique for performance evaluation?**

| | Performance estimation (without model comparison or hyperparameter estimation) | Performance estimation with hyperparameter optimization (or model comparison) |
|---|---|---|
| Large Dataset | **Train-test split** | **Train-validation-test split** |
| Small Dataset | **Cross-validation** | **Nested Cross-validation** |

# Summary / best practices

- **Resampling techniques are techniques to estimate generalization performance**
  - Resampling techniques = train/test split, train/val/test split, cross validation, nested cross validation, bootstrap

- When **comparing models** biased performance estimates are acceptable if they affect all models equally
  - CV, train/test splits, etc., need to be the same for each model that you need to be compared

- If you use the performance estimate of a dataset for adjusting the model – there must be a separate held out dataset if you want a true estimate of generalization performance

# But how do I get ROC's out of CV?

Each of the K folds will produce a set of confidence scores for the test / validation data of that fold.

**Fold 1**

| $y_i$ | confidence |
|---|---|
| 1 | 0.98 |
| 0 | 0.87 |
| 1 | 0.43 |
| 0 | 0.02 |

**Fold 2**

| $y_i$ | confidence |
|---|---|
| 1 | 0.99 |
| 1 | 0.65 |
| 0 | 0.22 |
| 0 | 0.14 |

**Fold 3**

| $y_i$ | confidence |
|---|---|
| 1 | 0.58 |
| 0 | 0.87 |
| 0 | 0.33 |
| 0 | 0.82 |

1. Merge the outputs from the K folds into a single set of confidence scores for making one ROC curve

**Note**: The confidence scores need to be on the same scale for this merging method to work properly

| $y_i$ | confidence |
|---|---|
| 1 | 0.98 |
| 0 | 0.87 |
| 1 | 0.43 |
| 0 | 0.02 |
| 1 | 0.99 |
| 1 | 0.65 |
| 0 | 0.22 |
| 0 | 0.14 |
| 1 | 0.58 |
| 0 | 0.87 |
| 0 | 0.33 |
| 0 | 0.82 |

2. Average the individual ROC curves from each fold

(This also enables measures of variation across the folds)

**Note**: you only have point data for changes in the ROC curve value, to compute the average you must interpolate between the points on the curve and evaluate the average across all the curves

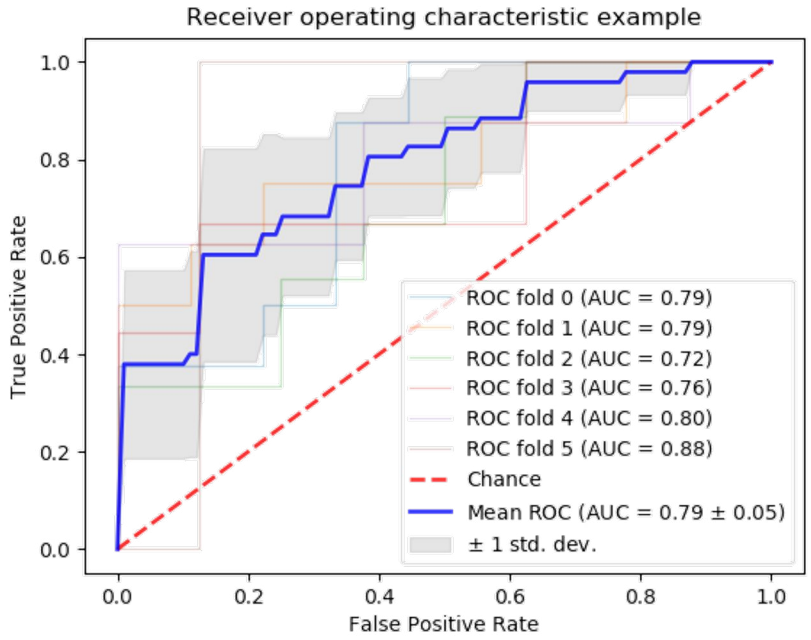



Image from: https://scikit-learn.org/

# Modeling Considerations

Accuracy (and techniques to measure it)

Computational Efficiency

Interpretability

# Computational Efficiency

Measure of how an algorithm's run time (or space requirements) grow 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:

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

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)

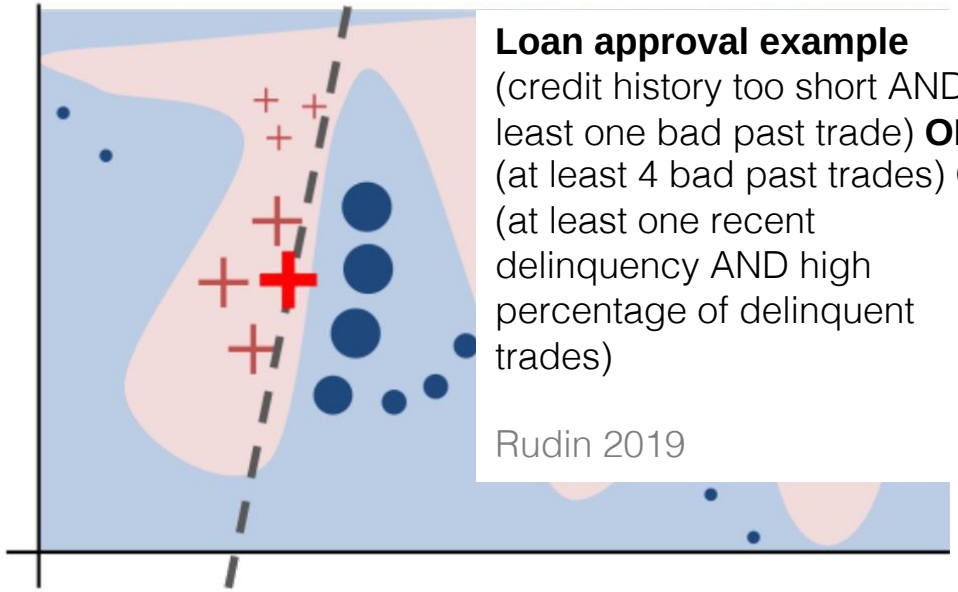
Lipton, Zachary C. "The Mythos of Model Interpretability: In Machine Learning, the Concept of Interpretability Is Both Important and Slippery." Queue 16, no. 3 (2018): 31–57.

## 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)