MODEL ASSESSMENT AND SELECTION (PART II)

CS 334: Machine Learning

Slides adapted from Joyce Ho, Lee Cooper and Ryan Tibshirani

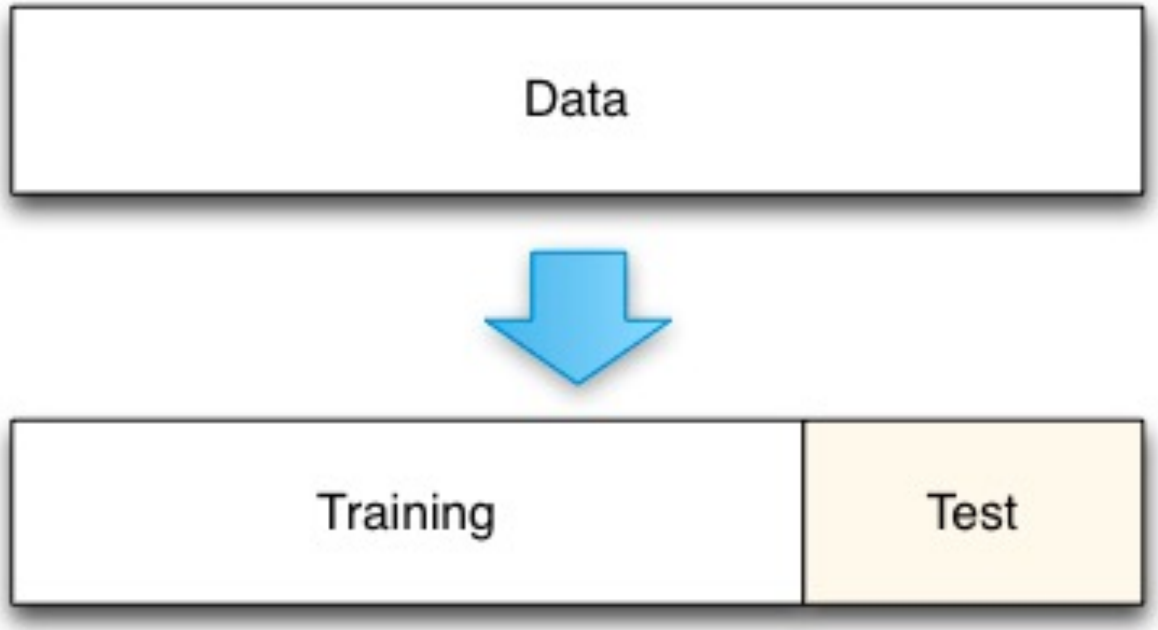
MODEL ASSESSMENT AND MODEL SELECTION

- Model assessment: evaluating a model's performance
- Model selection: selecting the proper level of flexibility for a model (e.g. K for KNN, tree size for decision tree)

MODEL ASSESSMENT

- Metrics:
  - Classification: Accuracy, precision/recall, AUROC (TPR/FPR), AUPRC (precision/recall)
  - Regression: MSE
- Process: training/test split (holdout), K-fold cross-validation, Monte Carlo cross validation

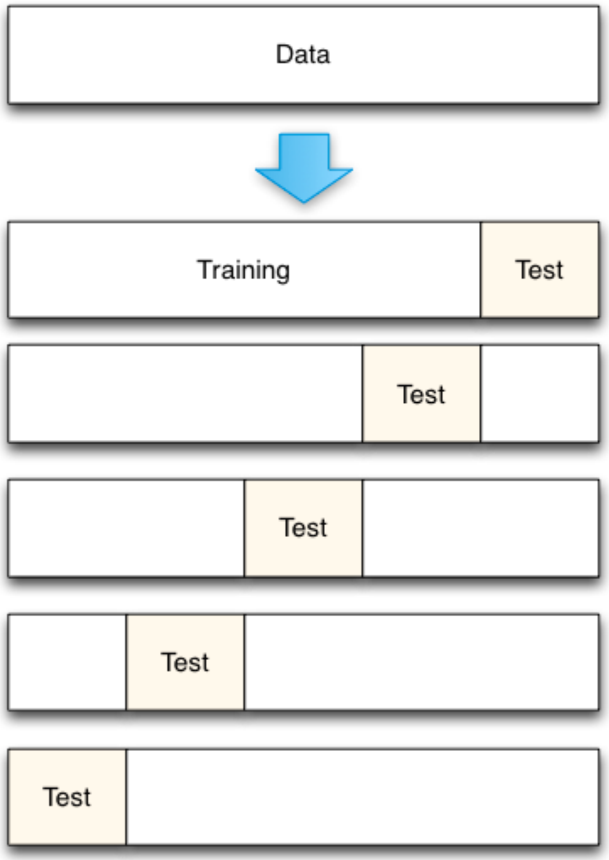
TRAINING/TEST SPLITS



What to do when there isn't enough data? Test data wasted?

K-FOLD CROSS VALIDATION

- Use all the data to train / test
  (but not all at the same time)
- Procedure:
  - Split the training data into K parts or "folds"
  - Train on all but the kth part and validate on the kth part
  - Rotate and report average over K measurements



http://scott.fortmann-roe.com/docs/MeasuringError.html

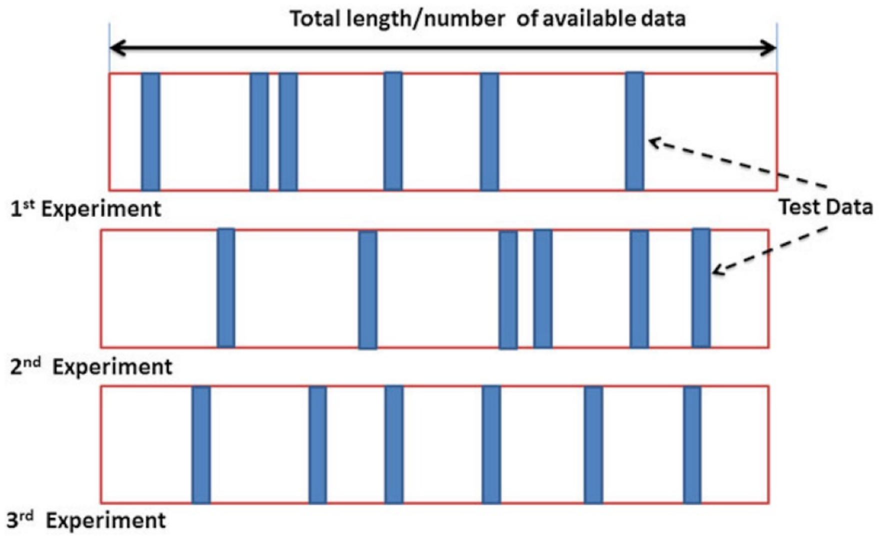
COMMON VALUES OF K

- K = 2 (two-fold cross validation)
- K = 5, 10 (5-fold, 10-fold cross validation) – common practice
- K = N (leave one out cross validation or LOOCV) – be cautious

Selection is based on how much data you have

MONTE-CARLO CROSS-VALIDATION

- AKA random sub-sampling
  - Randomly select (without replacement) some fraction of your data to form training set
  - Assign rest to test set
  - Repeat multiple times with new partitions



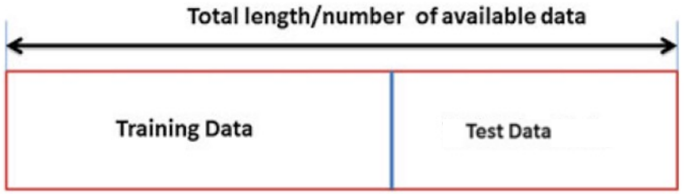
What are the differences to k-fold cross-validation?

K-FOLD VS MONTE-CARLO

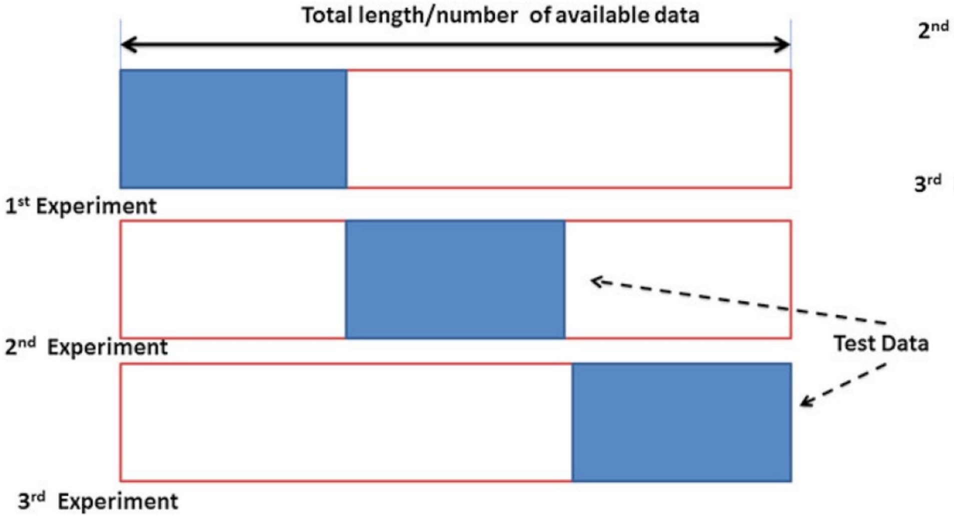
- Cross-validation
  - only explores a few of the possible ways to partition the data
  - Same point is used once as training and test
- Monte-Carlo
  - explores more possible partitions
  - Same point can be used multiple times

ASSESSMENT STRATEGIES

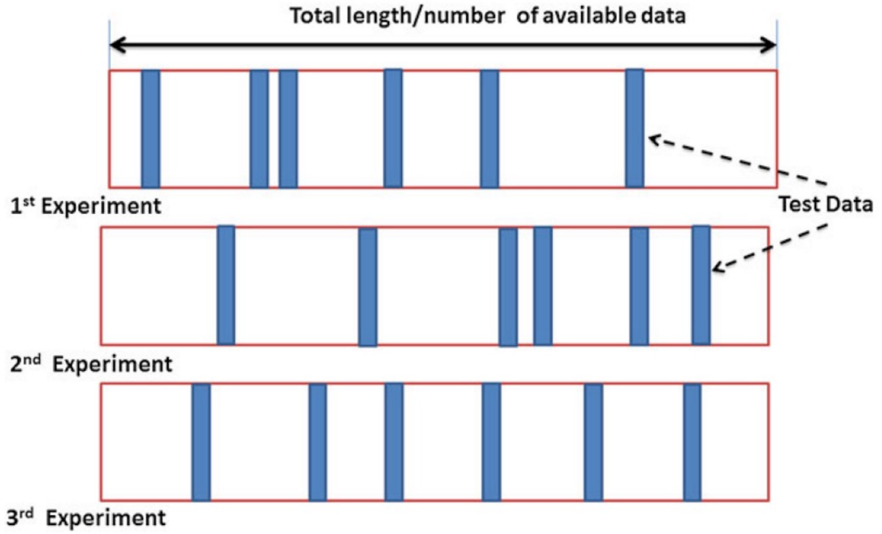
Holdout



k-fold cross-validation



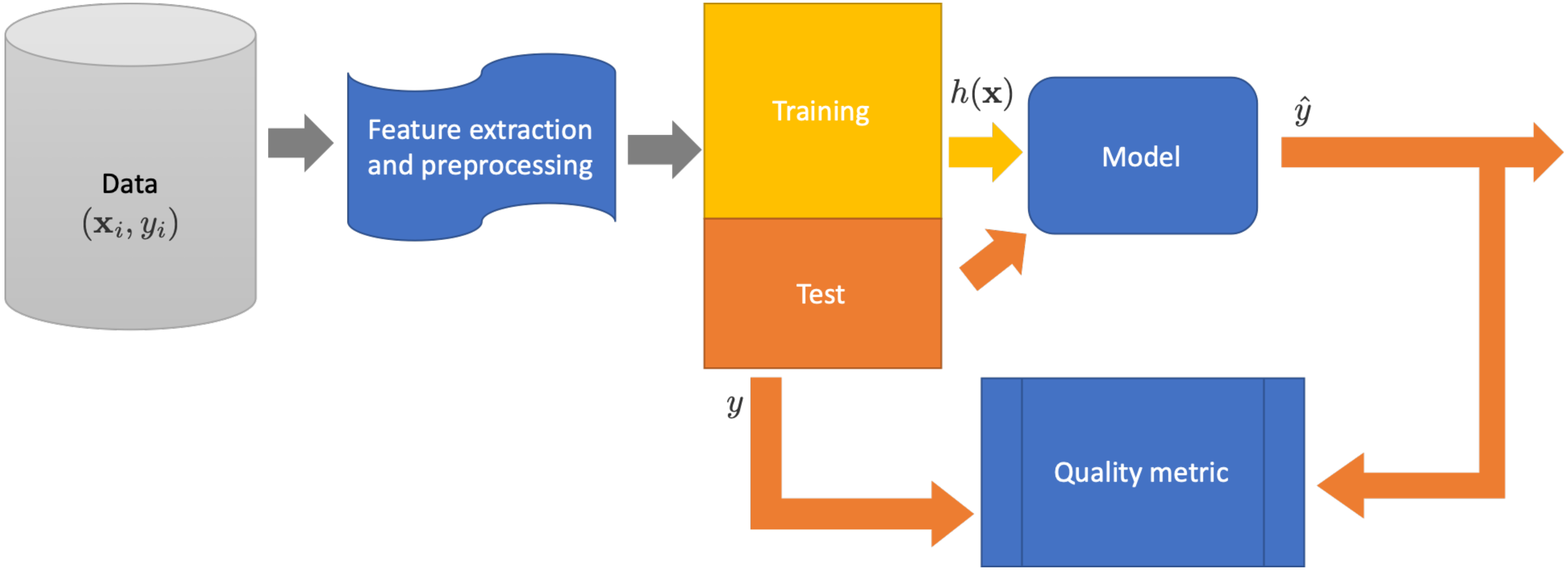
Monte-Carlo cross-validation



Figures 3.6, 3.7, 3.8 (Remesan & Mathew. Hydrological Data Driven Modeling: A Case Study Approach)

MODEL ASSESSMENT: THE PROCESS

- Holdout
- K-fold CV
- Monte-Carlo cross validation

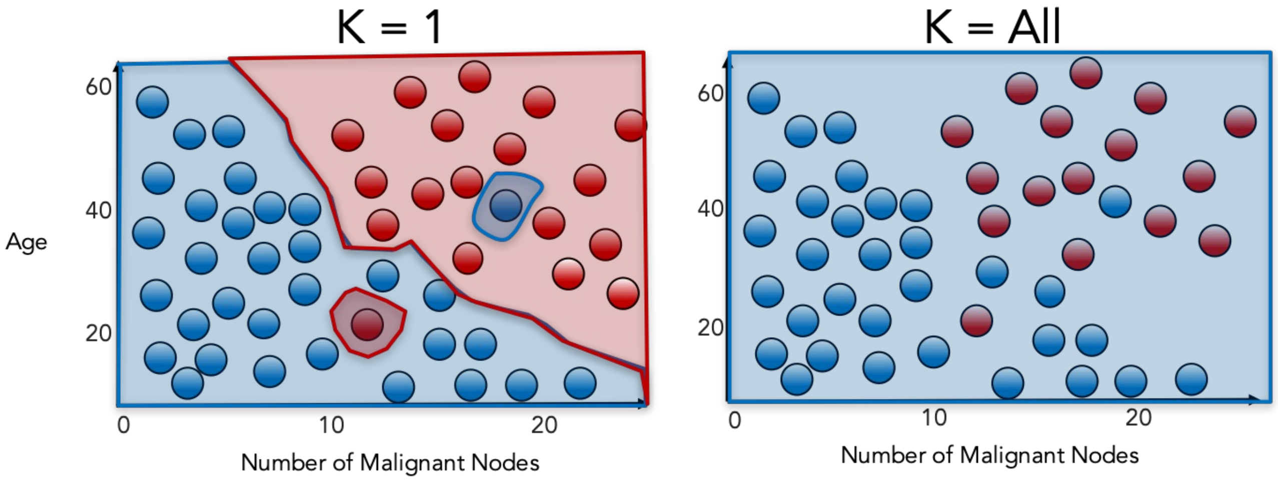


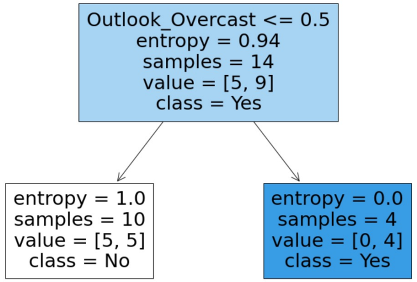
Report the performance on the “test” data

MODEL ASSESSMENT AND MODEL SELECTION

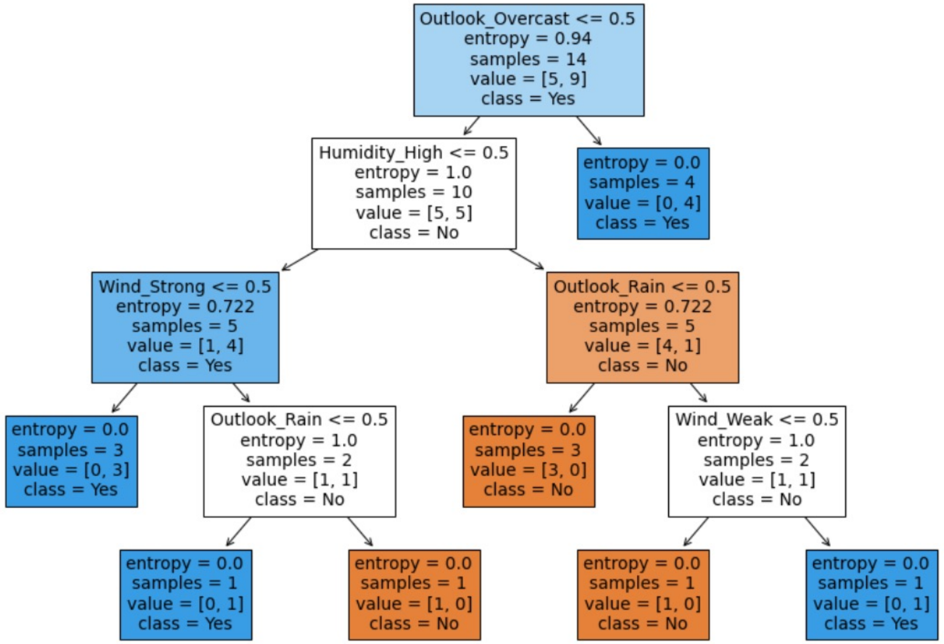
- Model assessment: evaluating a model’s performance
- Model selection: selecting the proper level of flexibility for a model (e.g. K for KNN, tree size for decision tree)

HOW TO CHOSE THE HYPERPARAMETERS?





Max_depth = 1



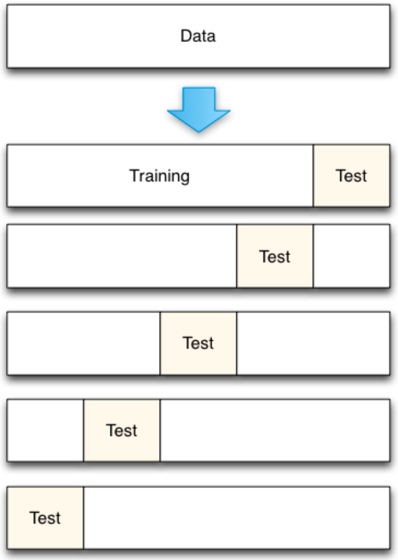
Max_depth = 4

MODEL SELECTION

- Select hyperparameters of the model – parameters of the learning algorithm
  - E.g. k and distance metric in kNN; tree depth in decision tree
- Different from model parameters
  - E.g. splitting point in decision tree
- Meta-optimization of the model

How to select the best hyperparameters like k?

USING K-FOLD FOR PARAMETER TUNING



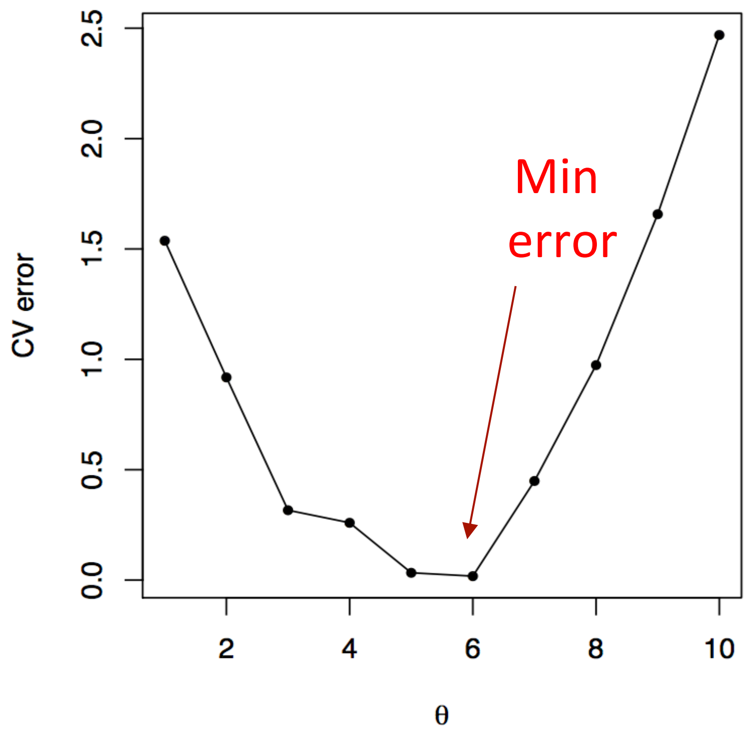
- Compute performance for specific parameter values

- Average error over all folds

$$\text{CV}(\theta) = \frac{1}{n}\sum_{k=1}^{K}\sum_{i\in F_k}(y_i - \hat{f}_\theta^{-k}(\mathbf{x}_i))^2$$

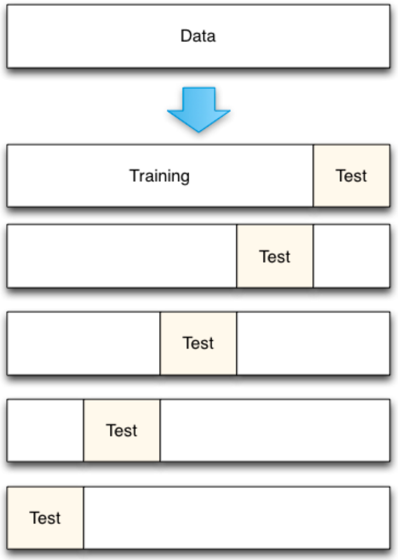
- Choose tuning parameter that minimizes CV error

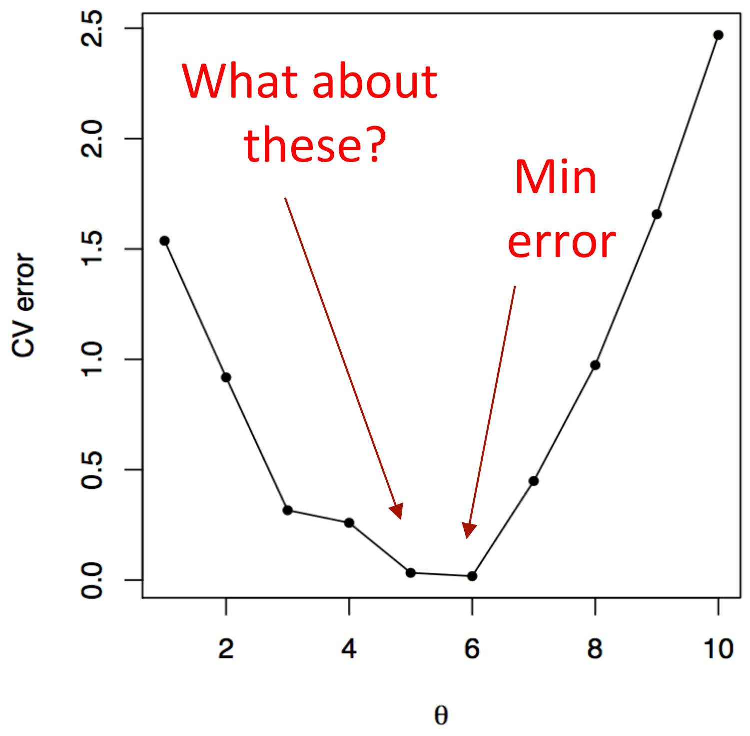
$$\hat{\theta} = \text{argmin}_{\theta\in\{\theta_1,\cdots,\theta_m\}}\text{CV}(\theta)$$



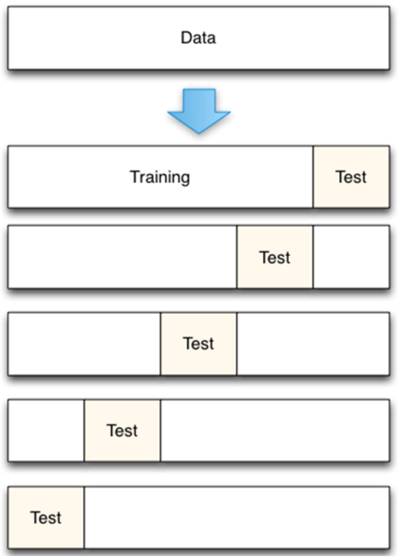
USING K-FOLD FOR PARAMETER TUNING

- If two models have almost equal performance, which model to choose?
- How to define “almost equal”?





ONE STANDARD ERROR RULE



- For small K << n, we can estimate standard deviation at each parameter

- Average error of kth fold:

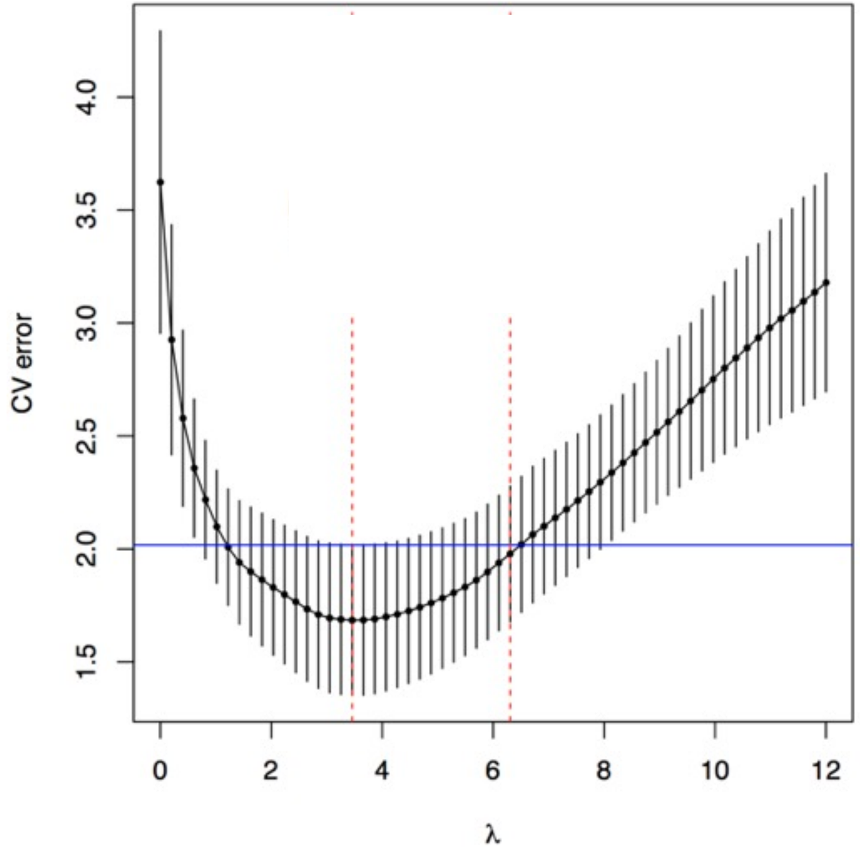
$$\mathrm{CV}_k(\theta) = \frac{1}{n_k} \sum_{i \in F_k} (y_i - \hat{f}_\theta^{-k}(\mathbf{x}_i))^2$$

- Sample standard deviation:

$$\mathrm{SD}(\theta) = \sqrt{\mathrm{var}(\mathrm{CV}_1(\theta), \cdots, \mathrm{CV}_K(\theta))}$$

- Standard error:

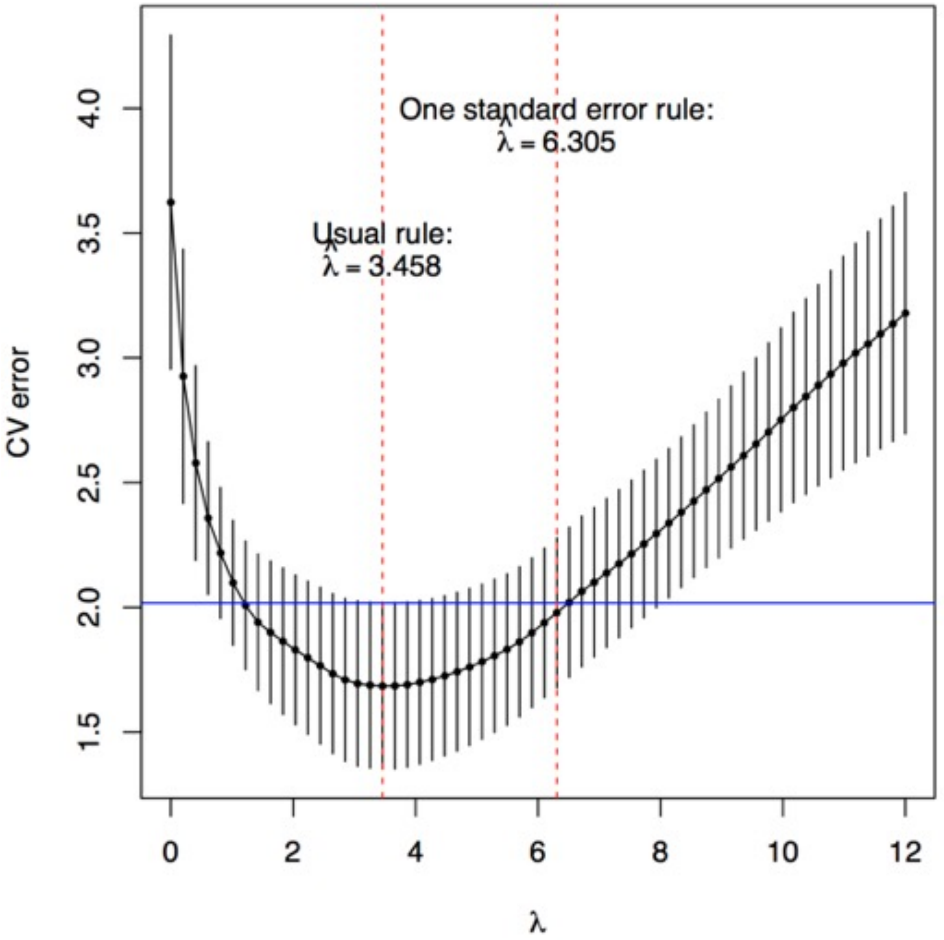
$$\mathrm{SE}(\theta) = \mathrm{SD}(\theta)/\sqrt{K}$$



ONE STANDARD ERROR RULE

- Alternative rule for selection of tuning parameter
- Idea: "All else equal (up to one standard error), go for the simpler model" – Occam's razer
- Find usual minimizer as before
- Move parameter in direction of decreasing complexity such that cross-validation error curve is within one standard error

$$\mathrm{CV}(\theta) \leq \mathrm{CV}(\hat{\theta}) + \mathrm{SE}(\hat{\theta})$$



Simpler model

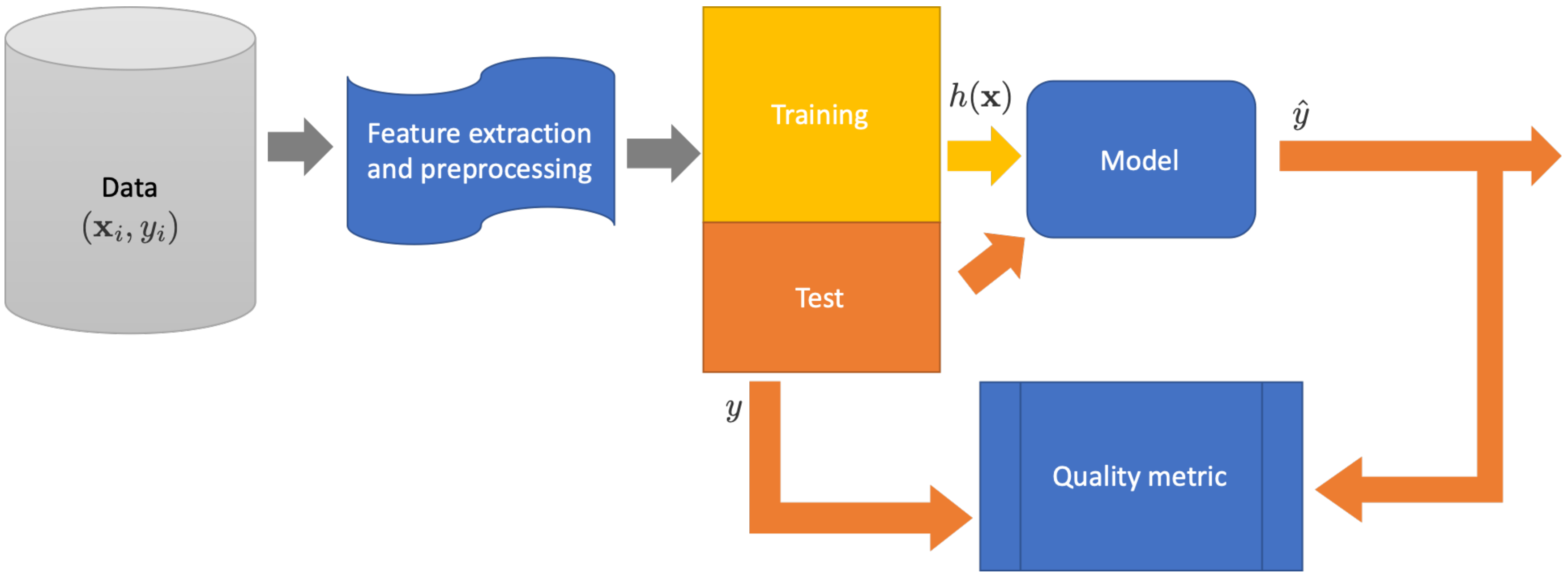
MODEL ASSESSMENT AND MODEL SELECTION

- Model assessment: evaluating a model's performance
- Model selection: selecting the proper level of flexibility for a model (e.g. K for KNN, tree size for decision tree)

How to do both model selection and model assessment?

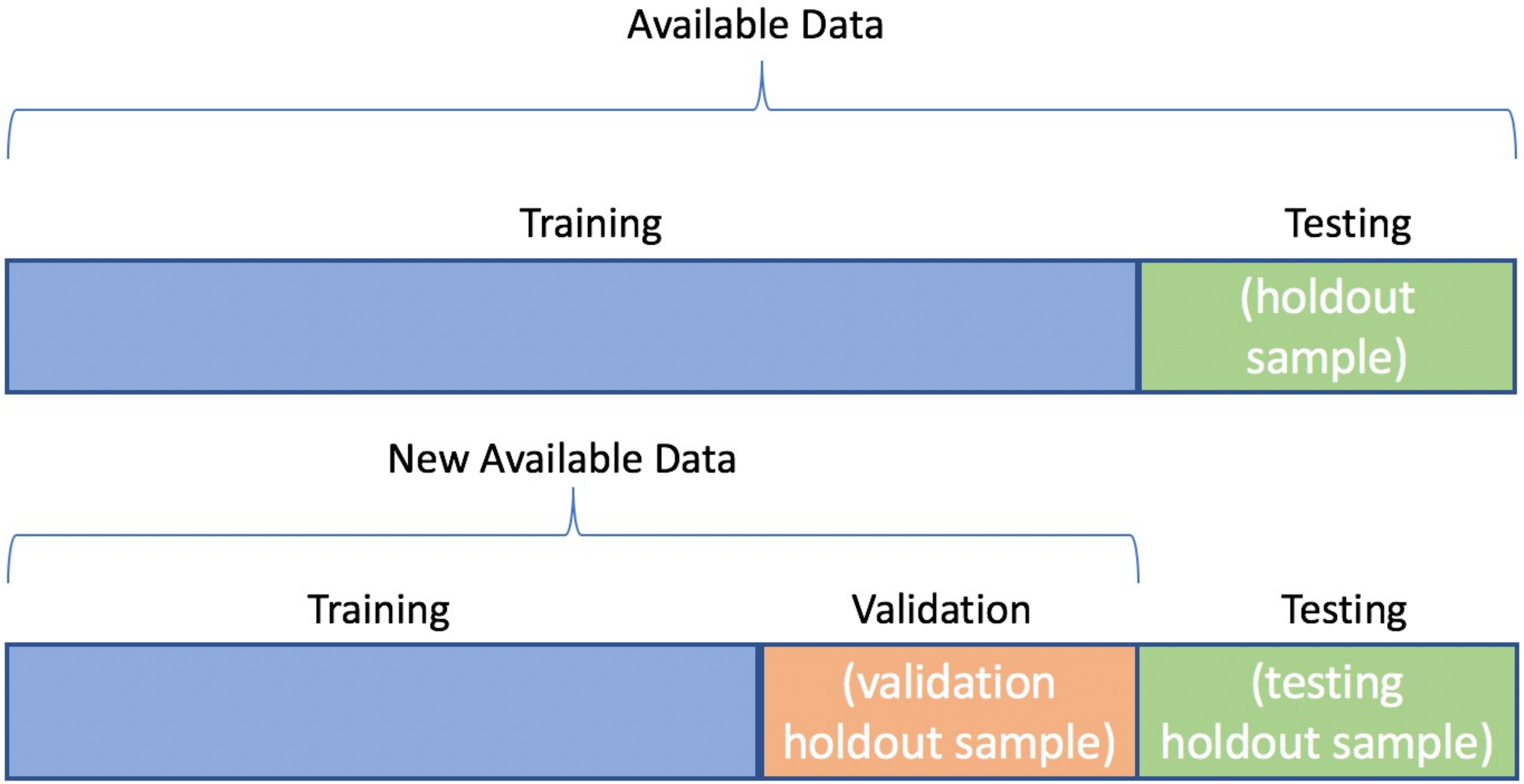
MODEL SELECTION AND ASSESSMENT: THE PROCESS

- Holdout
- K-fold CV
- Monte-Carlo cross validation



Tune the parameters AND report the performance using cross validation - Is this a good idea?

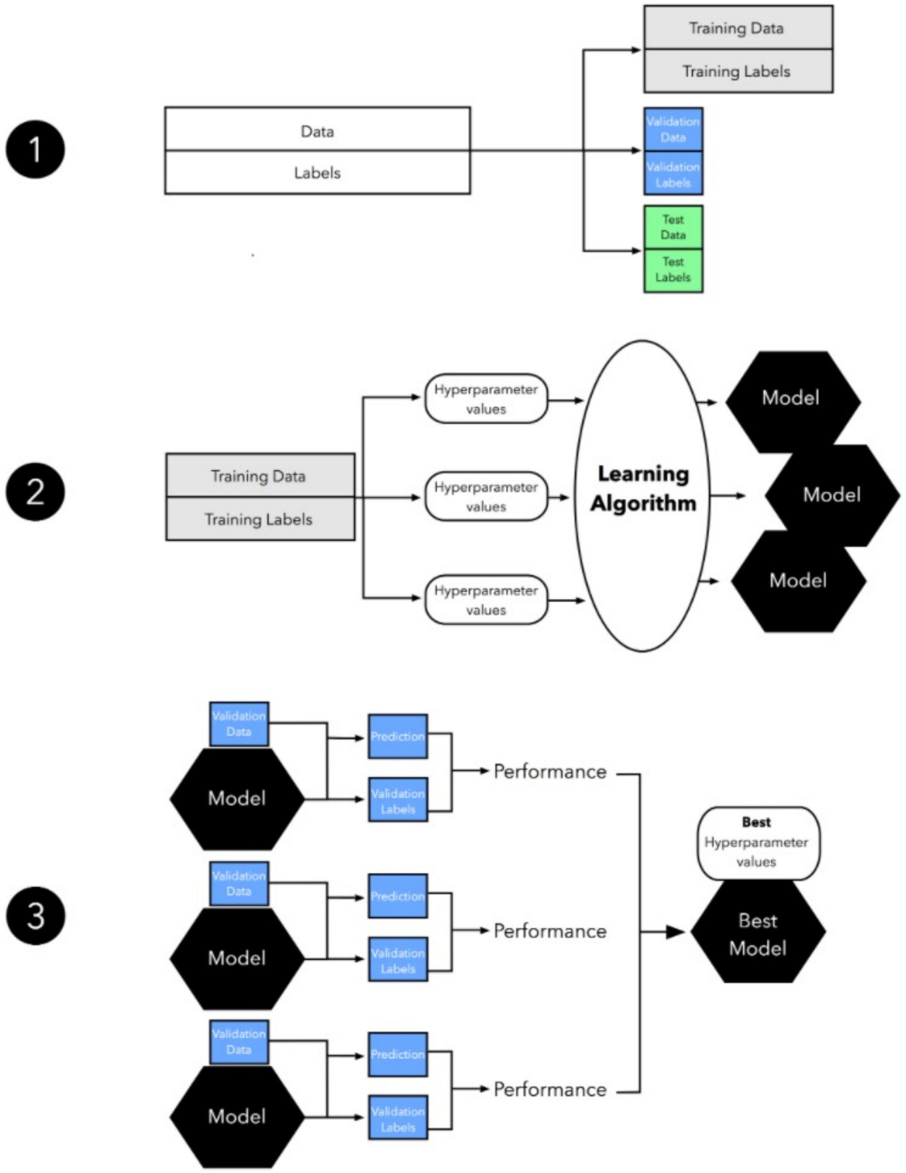
THREE WAY SPLIT



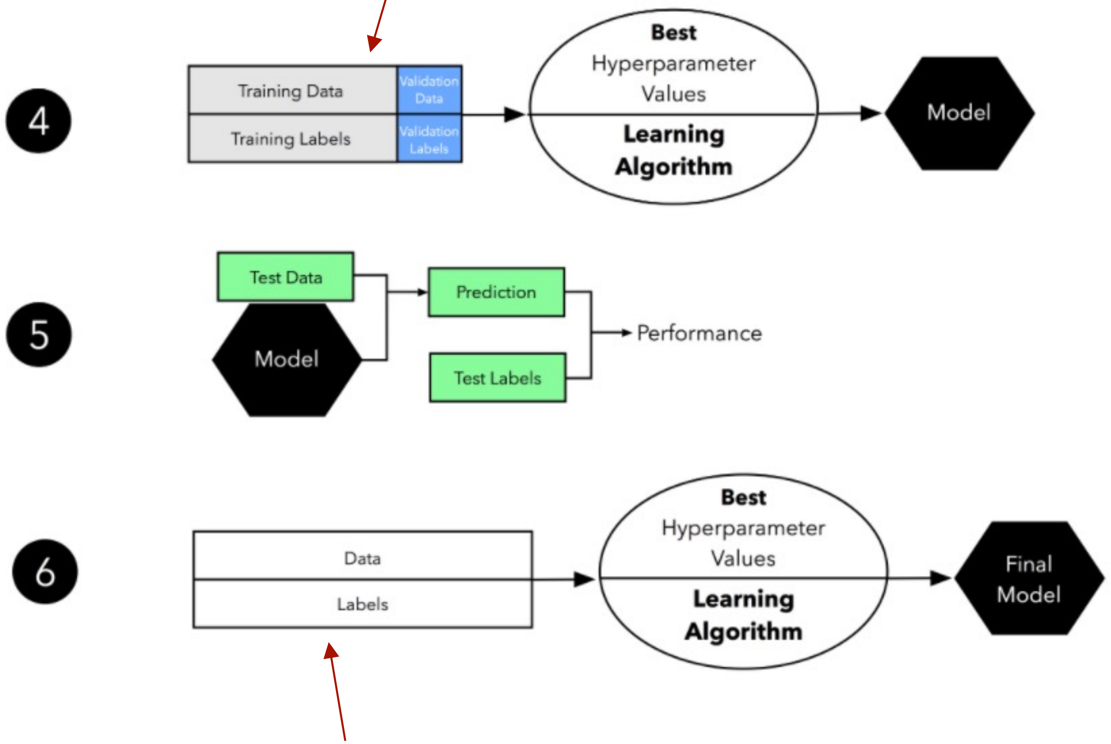
Validation data will NOT be used for training model

Test data will NOT be used for training model or tuning parameters

MODEL SELECTION AND ASSESSMENT: PROCESS (THREE WAY SPLIT)



Can merge training data and validation data for training the model for test performance

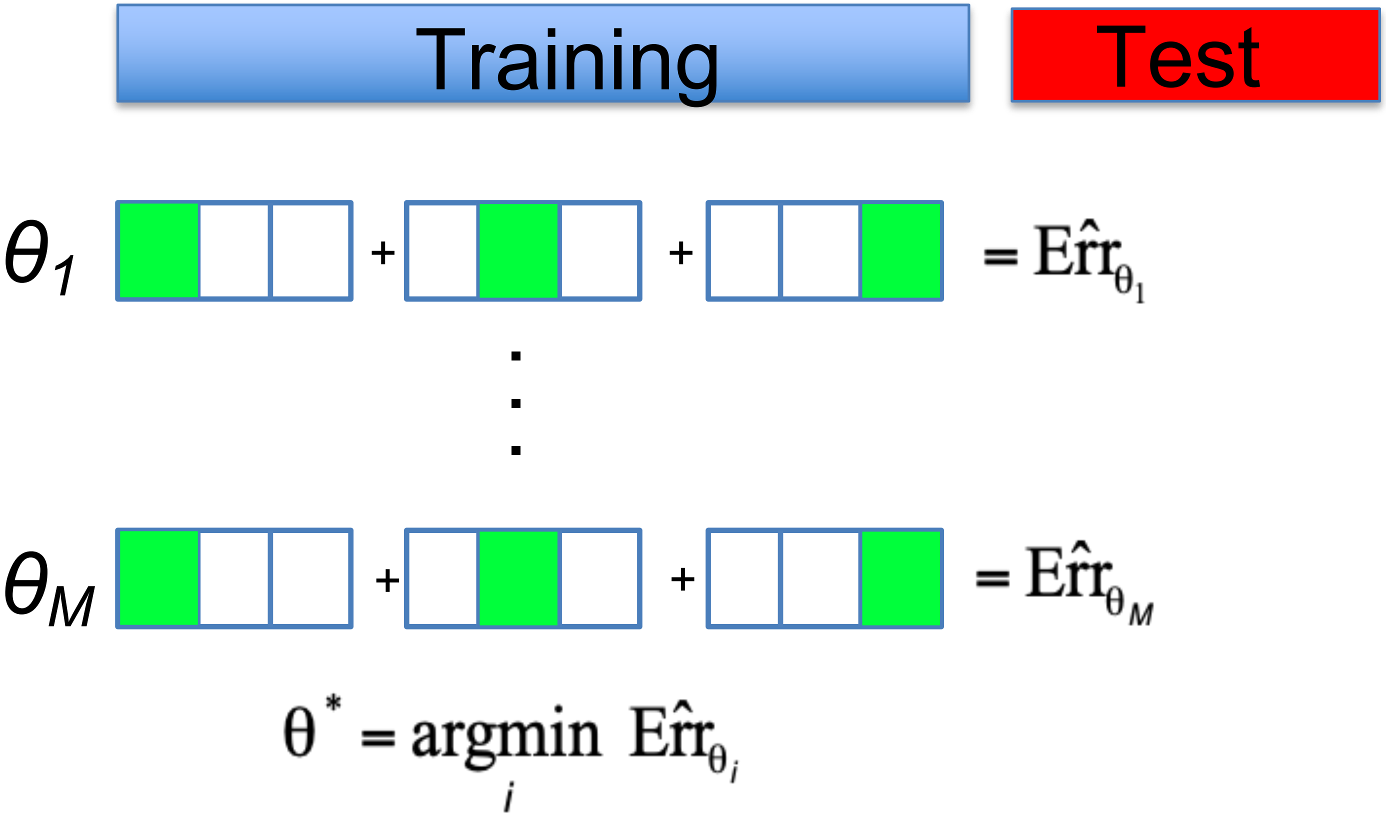


Can use all data for training the final model

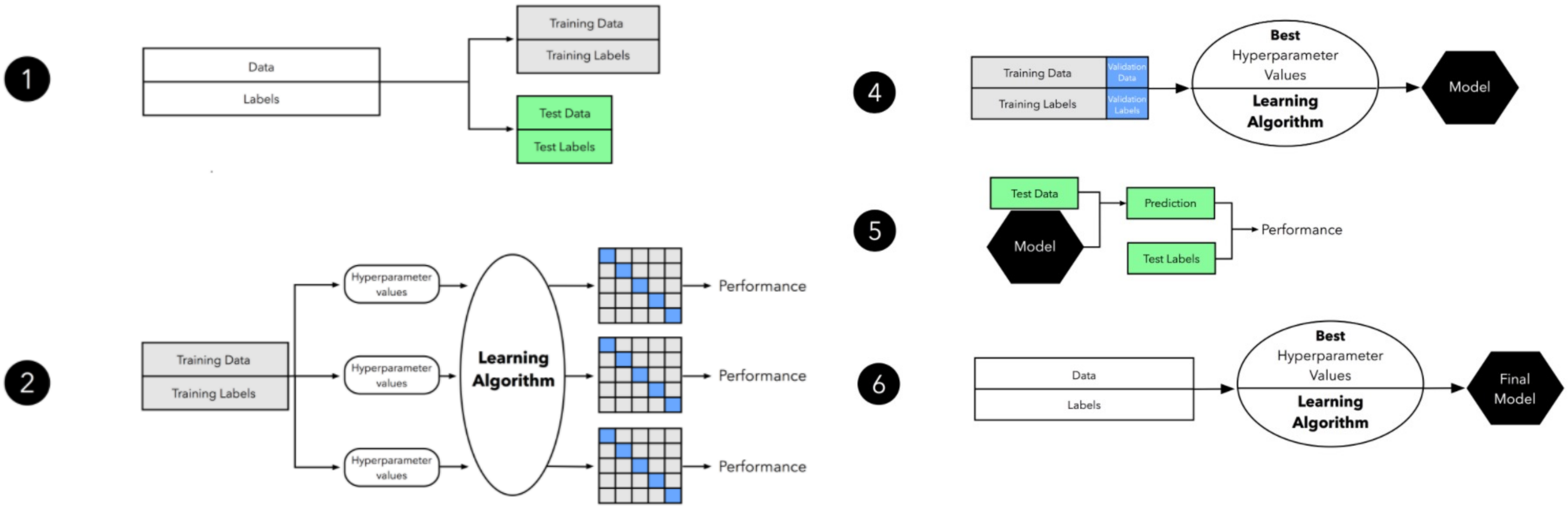
https://sebastianraschka.com/blog/2016/model-evaluation-selection-part3.html

K-FOLD CV + HOLDOUT

- Holdout: Test dataset to assess the performance
- Training set: Use k-fold CV to find the optimal parameter
- Use optimal parameters to train on the training data and assess on test



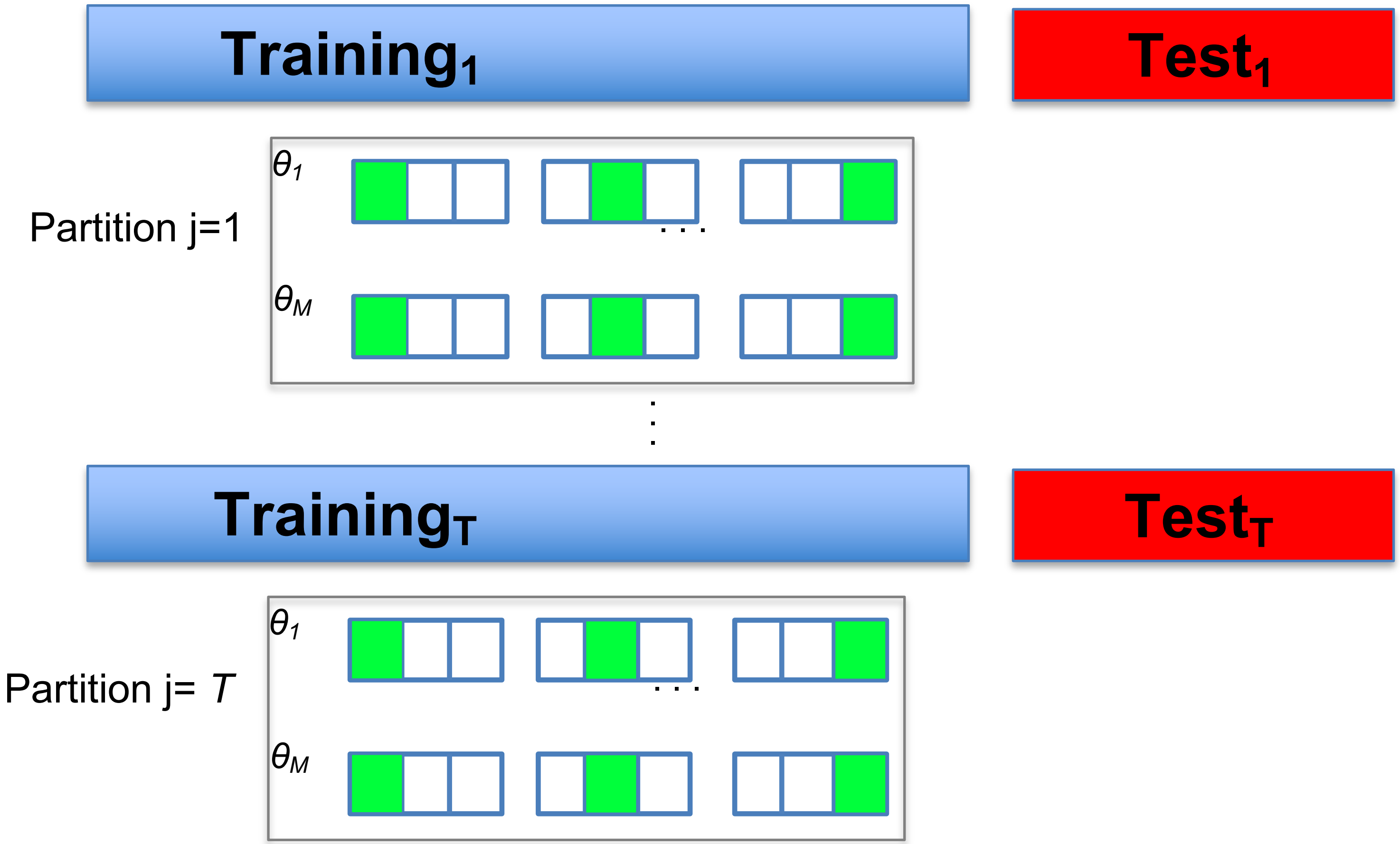
MODEL SELECTION AND ASSESSMENT: PROCESS (CV + HOLDOUT)



https://sebastianraschka.com/blog/2016/model-evaluation-selection-part3.html

NESTED CV

- Outer k-fold loop: Assess the performance
- Inner k-fold loop: Choose the optimal parameter



ASSESSMENT + SELECTION: TAKEAWAY

- Guidelines:
  - We cannot use the same samples to estimate both optimal model parameters and test/generalization error
  - Choice of methodology will depend on your problem and dataset

MODEL ASSESSMENT AND SELECTION: SKLEARN

- Import the function (see sklearn.model_selection)
  - train_test_split
  - KFold
- Split the data

```python
from sklearn.model_selection import train_test_split
train, test = train_test_split(data, test_size=0.3)
```

METRICS: SKLEARN

- Import the function (see metrics)
- Calculate the selected metric on using predicted and ground truth labels

```
from sklearn.metrics import accuracy_score
accuracy_value = accuracy_score(y_test, y_pred)
```

https://scikit-learn.org/stable/modules/model_evaluation.html

CLASSIFICATION/REGRESSION METRICS: SKLEAN

Scoring	Function	Comment
Classification		
'accuracy'	`metrics.accuracy_score`	
'balanced_accuracy'	`metrics.balanced_accuracy_score`	
'average_precision'	`metrics.average_precision_score`	
'neg_brier_score'	`metrics.brier_score_loss`	
'f1'	`metrics.f1_score`	for binary targets
'f1_micro'	`metrics.f1_score`	micro-averaged
'f1_macro'	`metrics.f1_score`	macro-averaged
'f1_weighted'	`metrics.f1_score`	weighted average
'f1_samples'	`metrics.f1_score`	by multilabel sample
'neg_log_loss'	`metrics.log_loss`	requires `predict_proba` support
'precision' etc.	`metrics.precision_score`	suffixes apply as with 'f1'
'recall' etc.	`metrics.recall_score`	suffixes apply as with 'f1'
'jaccard' etc.	`metrics.jaccard_score`	suffixes apply as with 'f1'
'roc_auc'	`metrics.roc_auc_score`	
'roc_auc_ovr'	`metrics.roc_auc_score`	
'roc_auc_ovo'	`metrics.roc_auc_score`	
'roc_auc_ovr_weighted'	`metrics.roc_auc_score`	
'roc_auc_ovo_weighted'	`metrics.roc_auc_score`	
Regression		
'explained_variance'	`metrics.explained_variance_score`	
'max_error'	`metrics.max_error`	
'neg_mean_absolute_error'	`metrics.mean_absolute_error`	
'neg_mean_squared_error'	`metrics.mean_squared_error`	
'neg_root_mean_squared_error'	`metrics.mean_squared_error`	
'neg_mean_squared_log_error'	`metrics.mean_squared_log_error`	
'neg_median_absolute_error'	`metrics.median_absolute_error`	
'r2'	`metrics.r2_score`	
'neg_mean_poisson_deviance'	`metrics.mean_poisson_deviance`	
'neg_mean_gamma_deviance'	`metrics.mean_gamma_deviance`	

MODEL SELECTION: SKLEARN

- GridSearchCV: exhaustive search for best parameter values using cross validation
- RandomizedSearchCV: randomized search over parameters

```
from sklearn.model_selection import GridSearchCV
from sklearn.neighbors import KNeighborsClassifier
clf = GridSearchCV(KNeighborsClassifier(),
                   {'n_neighbors': range(1,20,2)},
                   cv=5,
                   scoring='precision')
clf.fit(X_train, y_train)
```

https://scikit-learn.org/stable/modules/grid_search.html
https://scikit-learn.org/stable/modules/cross_validation.html#cross-validation

MS-EXAMPLE.IPYNB

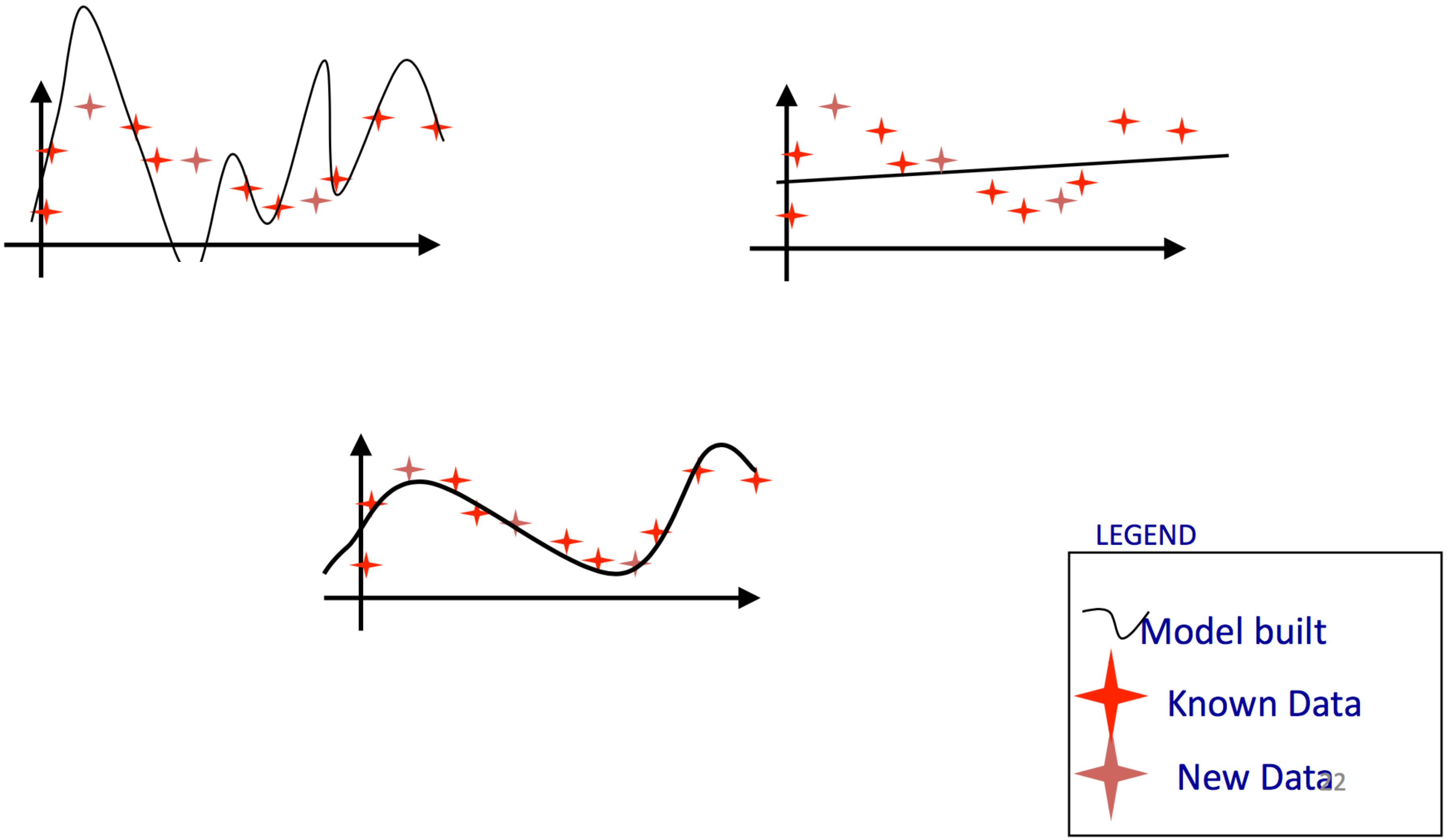
HTTPS://COLAB.RESEARCH.GOOGLE.COM/DRIVE/1_WZYGFCOJ_-HYEBUN9MLIZHBATLO3L5S

BIAS AND VARIANCE TRADEOFF

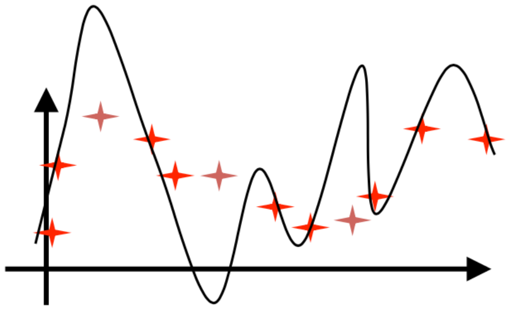
CS 334: Machine Learning

GROUP ACTIVITY

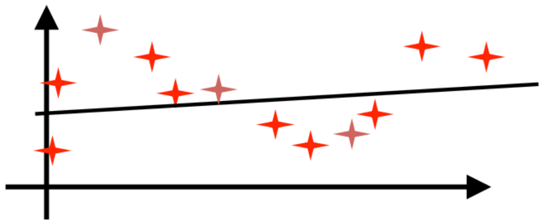
WHICH MODEL SHOULD WE CHOOSE AND WHY?



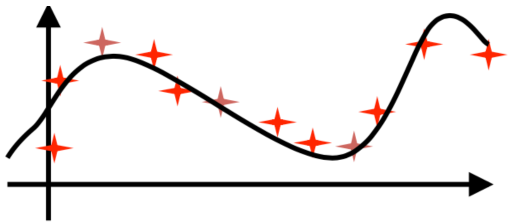
WHAT IS A “GOOD” MODEL?



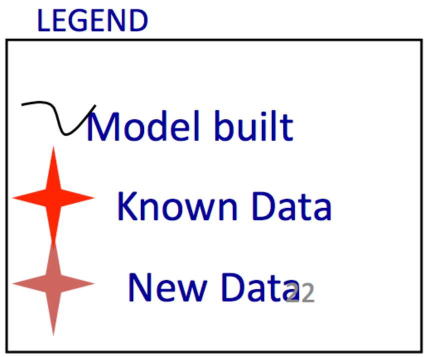
Very Good on Training Set
Poor at Predicting



Poor on Training Set
Poor at Predicting



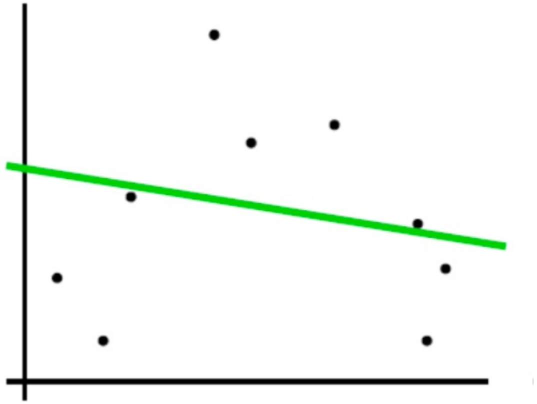
Just Right



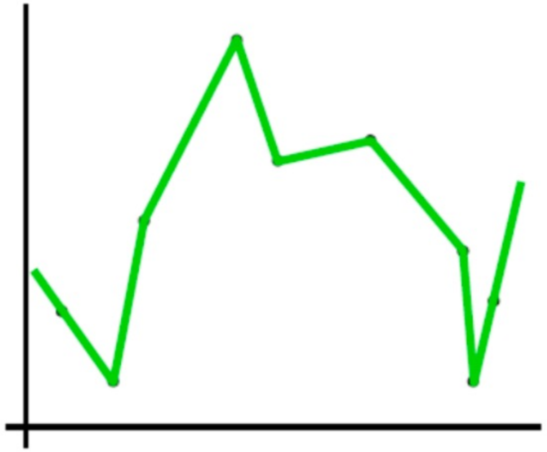
http://www.cs.cmu.edu/~10601b/slides/learning_theory.pdf

BIAS-VARIANCE TRADEOFF: INTUITION

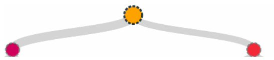
- Too “simple” model —> does not fit data well

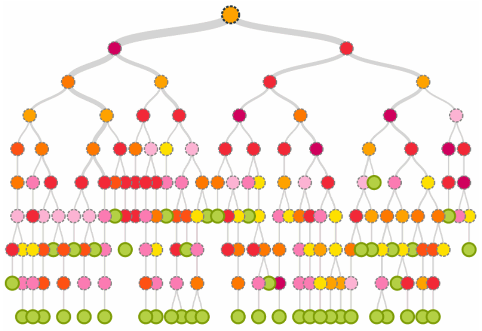


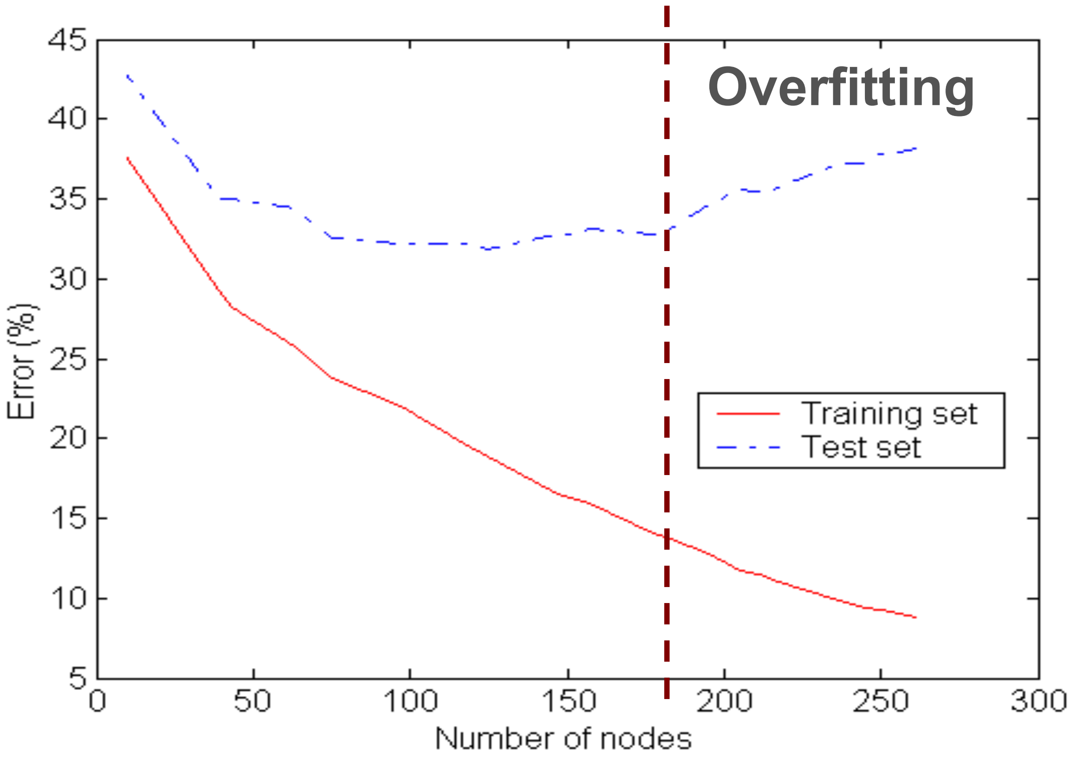
- Too complex model —> small changes to the data changes the solution a lot



REVIEW: DECISION TREE







BIAS AND VARIANCE

- The expected test error can be decomposed into the sum of
  - Variance of the estimated value
  - Squared bias of the estimated value
  - Variance of irreducible error

$$E\left(y_0 - \hat{f}(x_0)\right)^2 = \mathrm{Var}(\hat{f}(x_0)) + [\mathrm{Bias}(\hat{f}(x_0))]^2 + \mathrm{Var}(\epsilon).$$

BIAS: CONCEPTUALLY

- Difference between expected (or average) prediction of our model and the correct value we are trying to predict
  - Imagine training the model on multiple training datasets
  - Bias is difference in truth and expected prediction
  - Error of approximating a real-life problem by a model

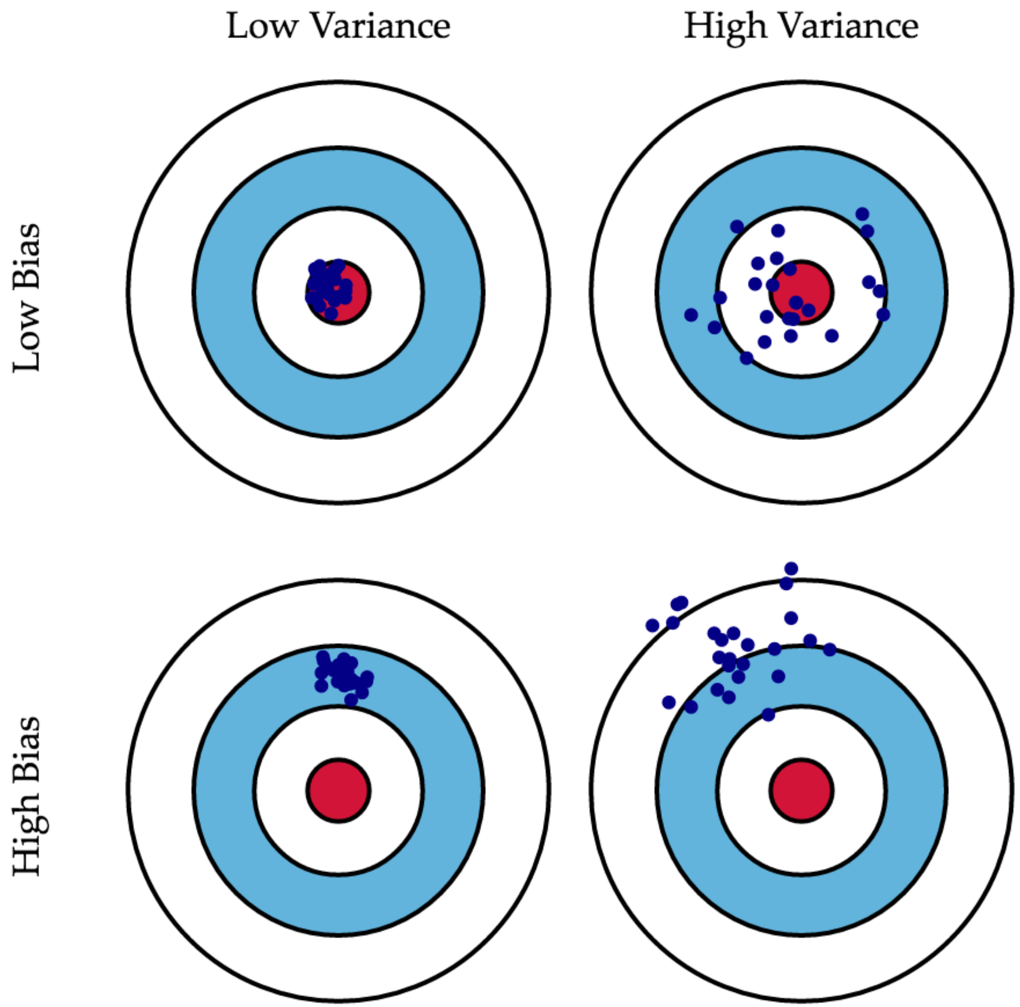
http://scott.fortmann-roe.com/docs/BiasVariance.html

VARIANCE: CONCEPTUALLY

- Amount by which the prediction would change if using a different training dataset
  - Difference between what you learned on a particular dataset versus what you expect to learn

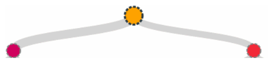
http://scott.fortmann-roe.com/docs/BiasVariance.html

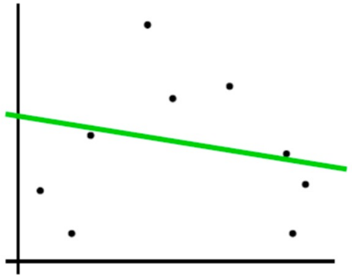
BIAS AND VARIANCE: GRAPHICALLY

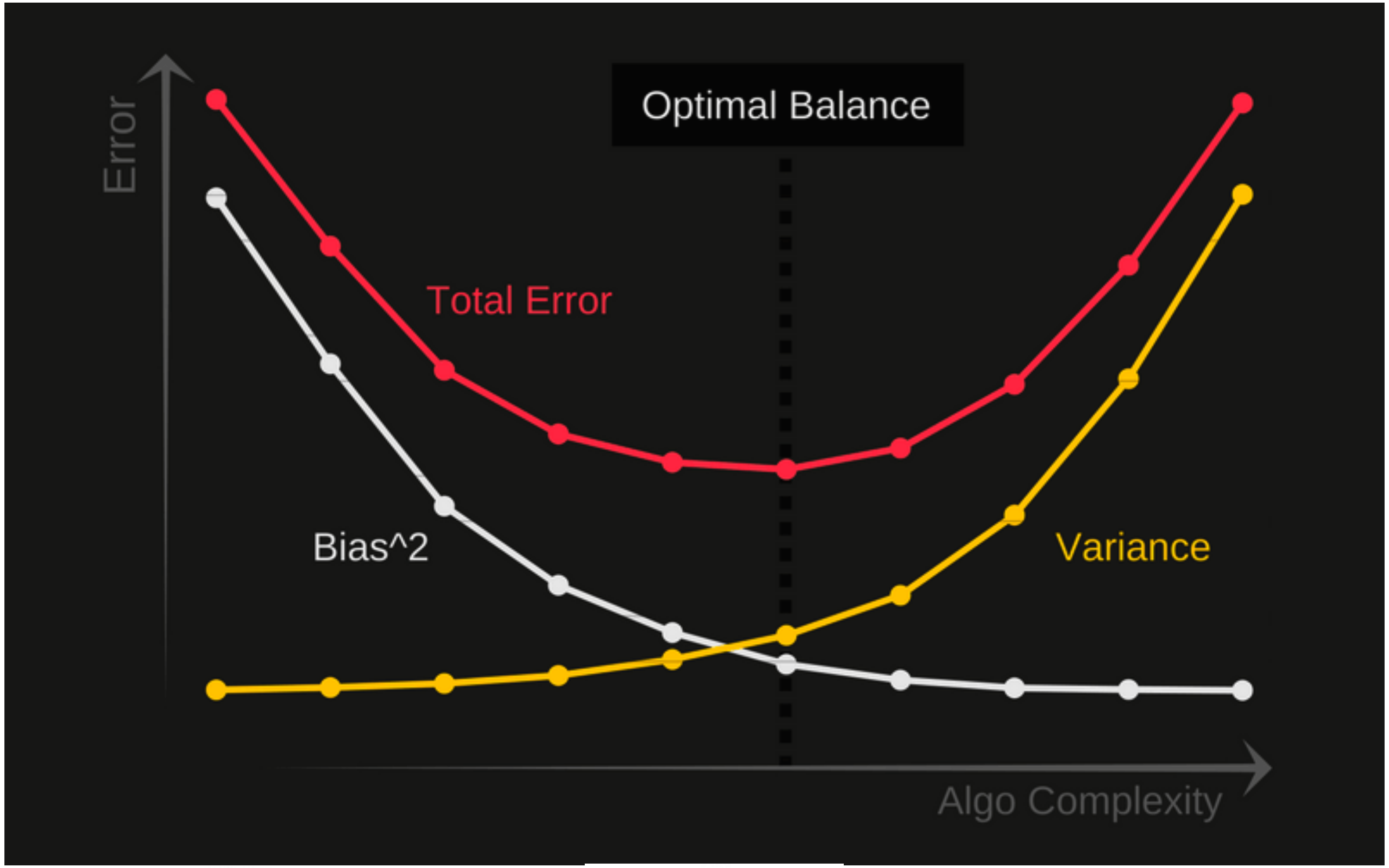


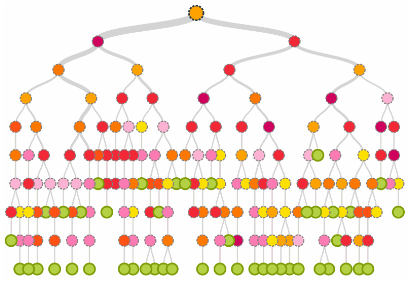
http://scott.fortmann-roe.com/docs/BiasVariance.html

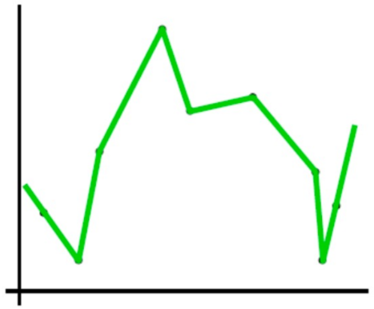
Total Error = $Bias^2$ + Variance + Irreducible Error

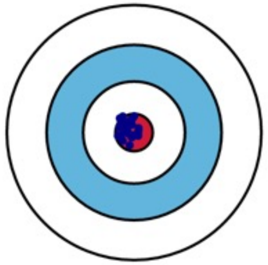


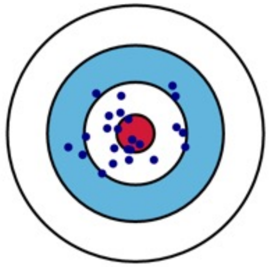








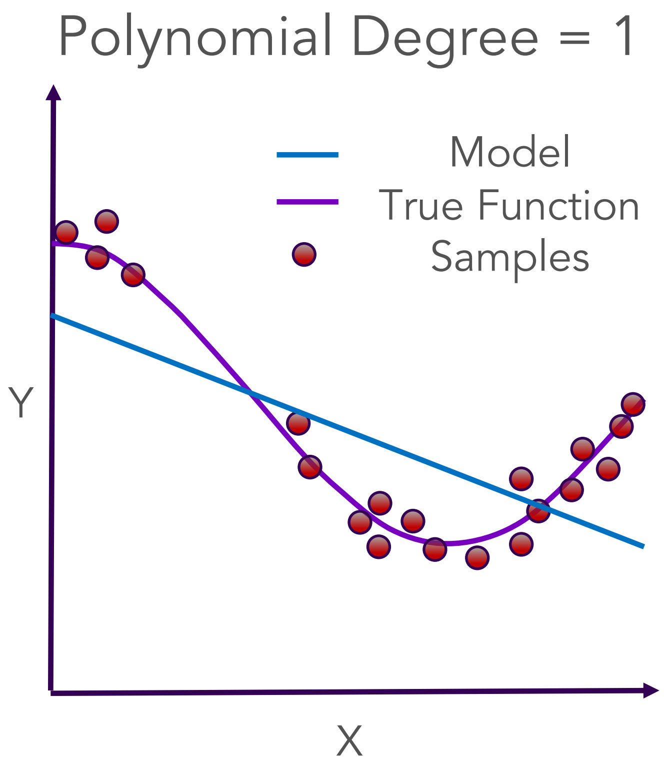


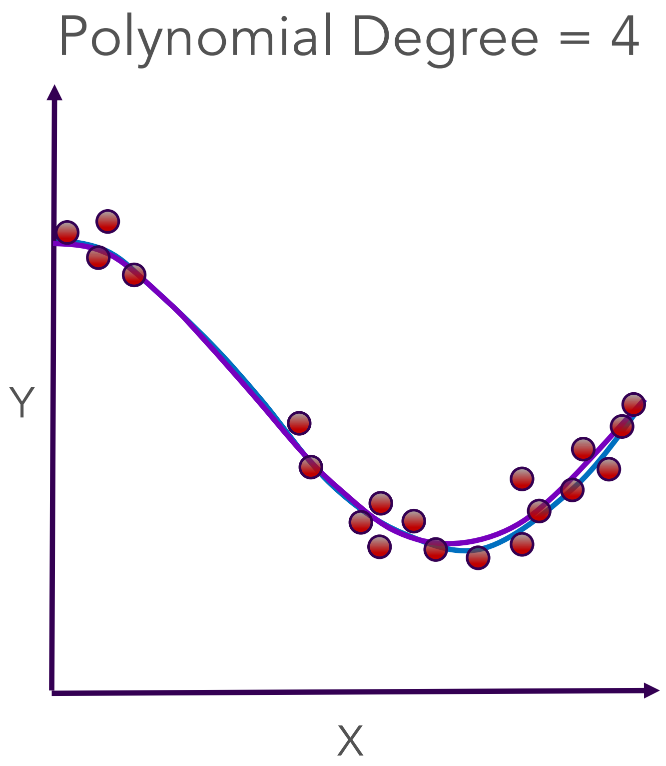


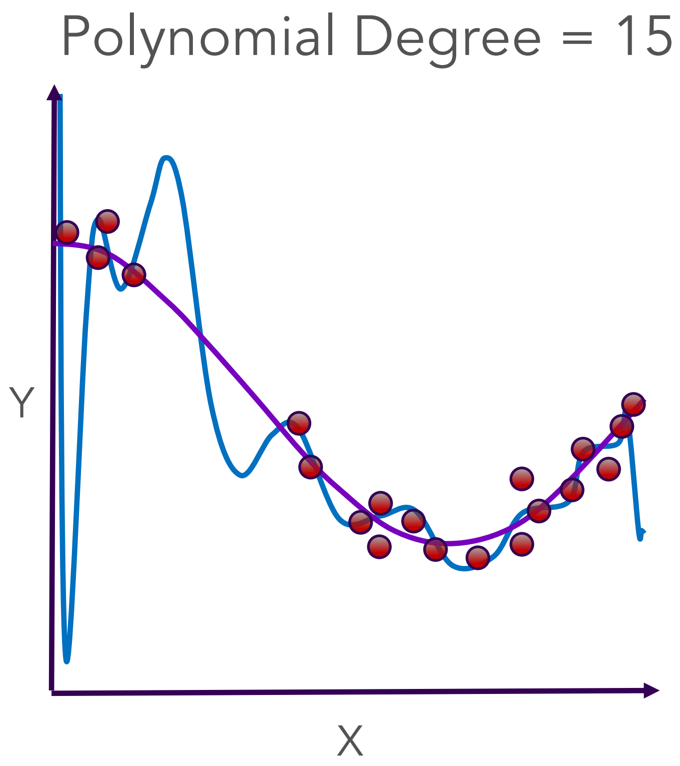
GENERALIZATION & OVERFITTING

- Generalization — model performance of a model on independent / future unseen data (data not used in training)
- Underfitting – model is unable to capture the relationship between the input and output variables accurately; high error on both training and test data
- Overfitting — model is specific to the training set and is learning the noise from the data instead of generalizable rule; low error on training but high error on test data

MODEL GENERALIZATION





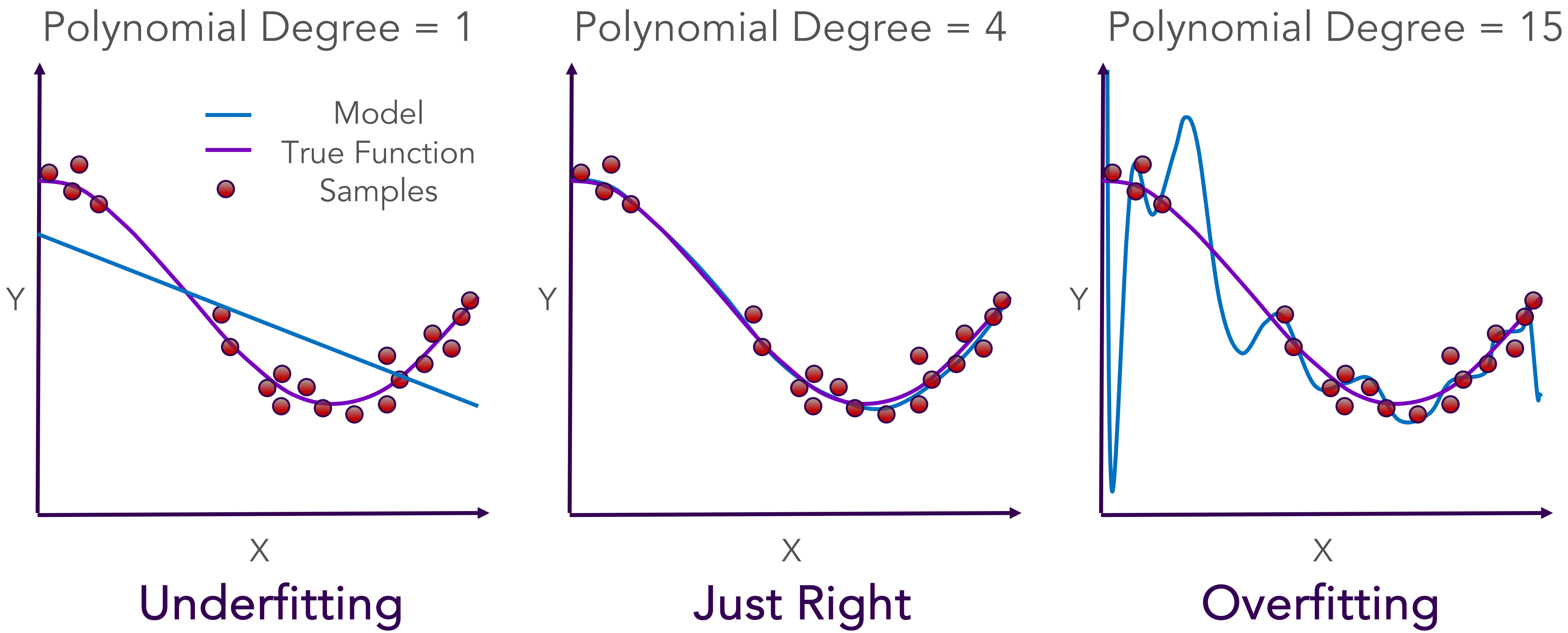


Poor on Training Set
Poor at Predicting

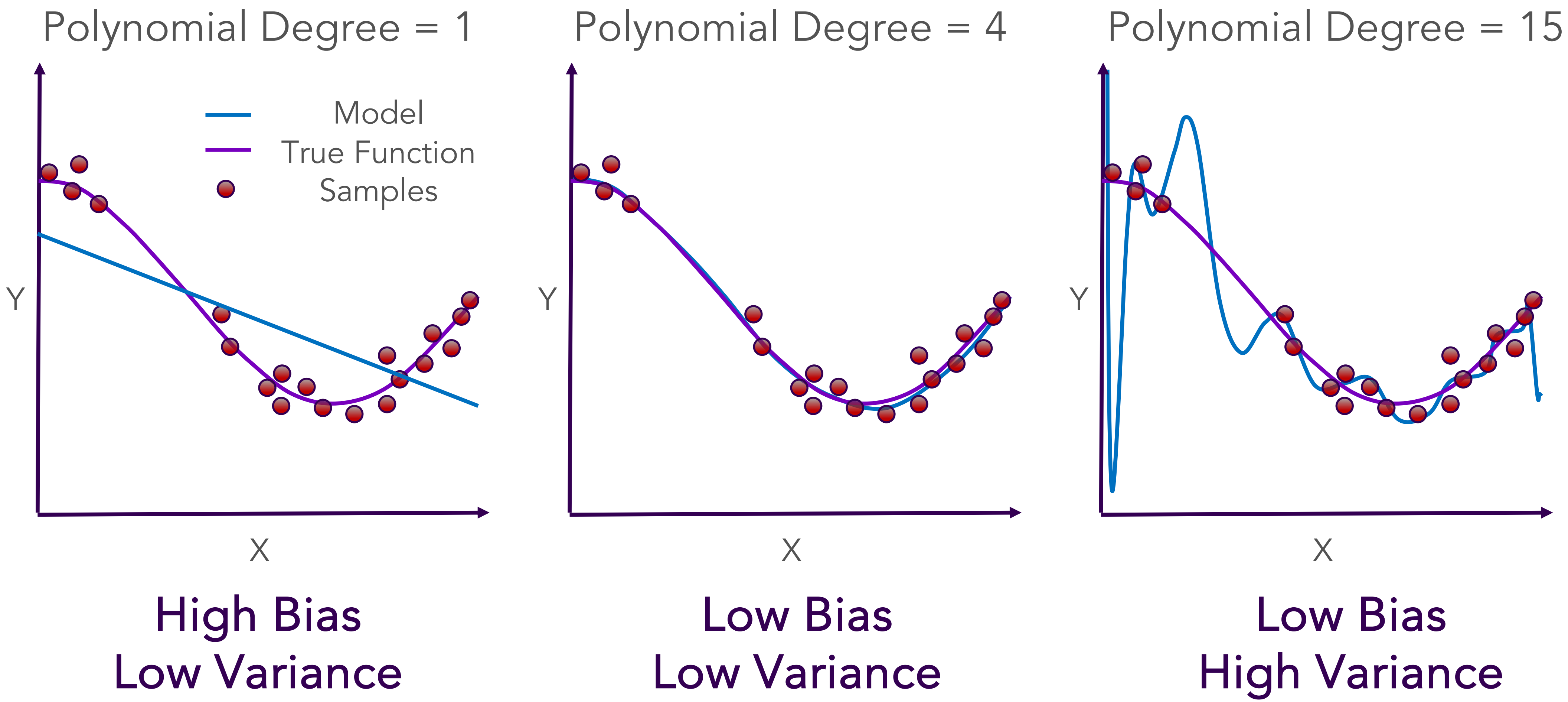
Generalizable

Very Good on Training Set
Poor at Predicting

UNDERFITTING VS OVERFITTING

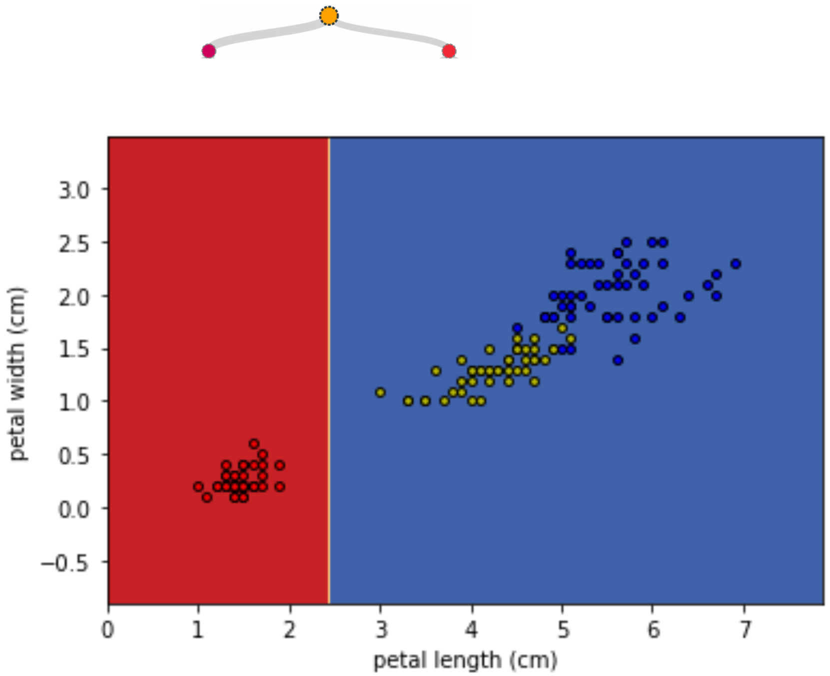


BIAS-VARIANCE TRADE-OFF



BIAS ANALYSIS: SOURCES

- Inability to represent certain decision boundaries
- Classifiers are “too global” (e.g., single linear separator)



Decision tree (max depth = 1)

High bias —> underfitting

How to reduce bias?

BIAS ANALYSIS: REDUCTION

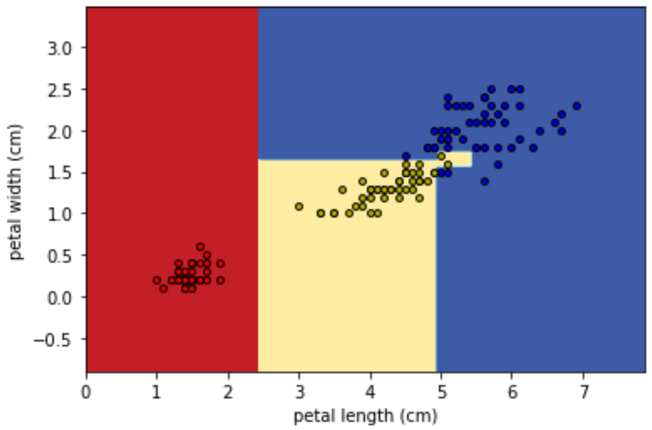
- More complex models
- More features

VARIANCE ANALYSIS: SOURCES

- Noise in labels or features
- Training data too small
- "Too local" algorithms that easily fit data
- Randomness in learning algorithm (i.e., non-convex algorithms)

High variance —> overfitting

How to reduce variance?

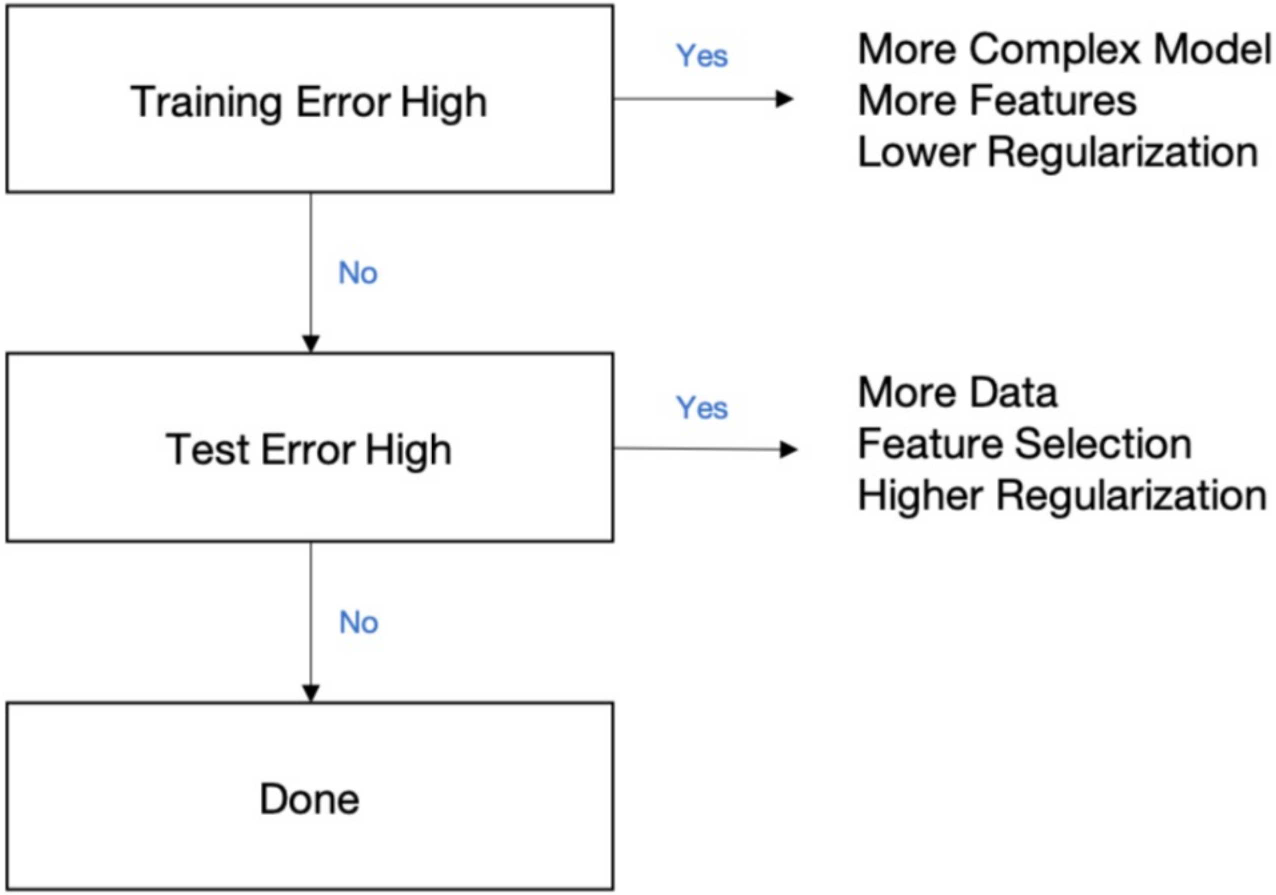


Decision tree (max depth = 3)

VARIANCE ANALYSIS: REDUCTION

- Use more data (increase size of training data)
- Less complex models
- Fewer features (feature selection)

HOW TO USE BIAS-VARIANCE



HOMEWORK #2

- Out **9/13**, Due **9/29 @ 11:59 PM ET**
- 3 questions
  - Q1: Decision tree implementation
  - Q2: Model assessment
  - Q3: Model selection and robustness of k-nn and decision tree