# MAS DSE 230
# Scalable Analytics

## Model Evaluation

Mai H. Nguyen

# TODAY'S TOPICS

- Model Evaluation
- Dask
- Cloud Computing
- AWS

# MODEL EVALUATION

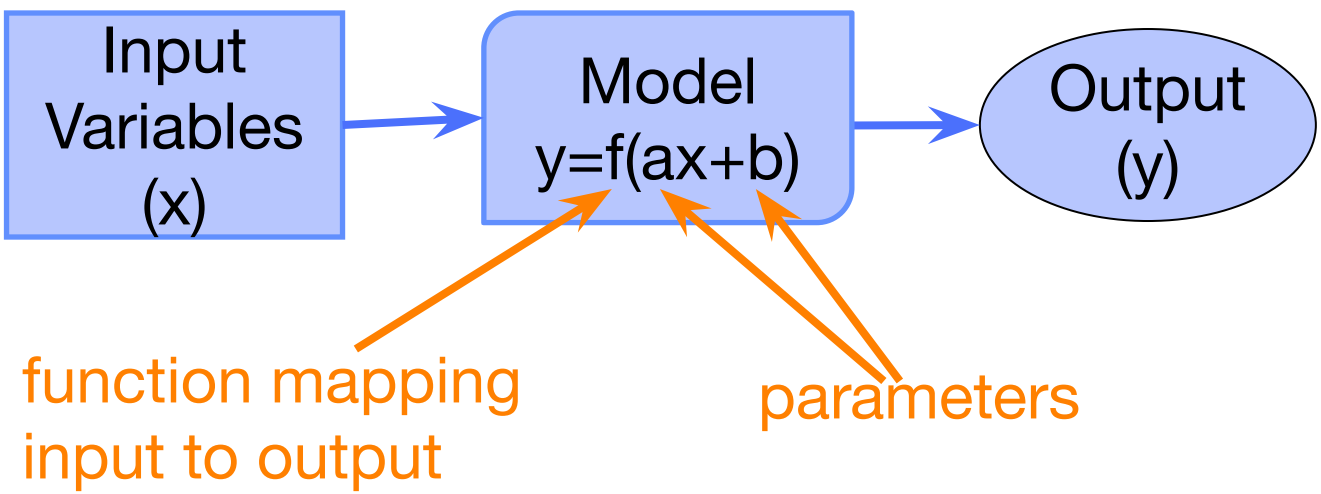
- Evaluation Metrics
- Generalization & Overfitting
- Model Selection & Model Evaluation
- Hyperparameter Tuning
- Ensemble Learning

# MODEL EVALUATION

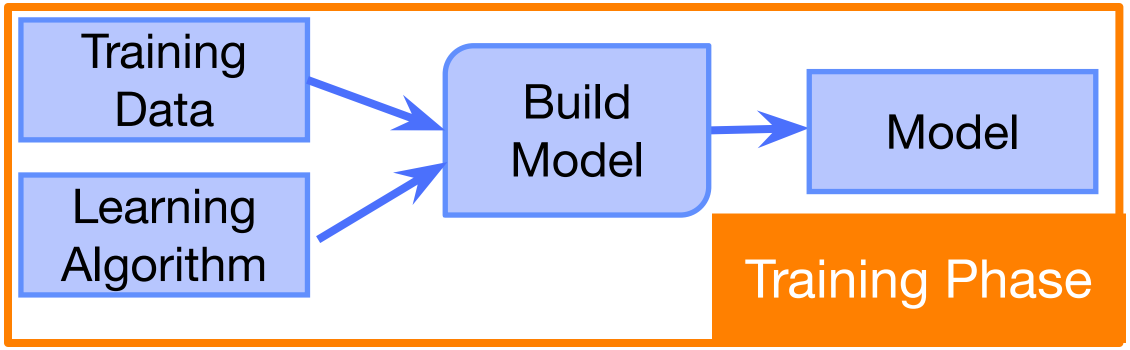
- Evaluation Metrics
- Generalization & Overfitting
- Model Selection & Model Evaluation
- Hyperparameter Tuning
- Ensemble Learning

# BUILDING MACHINE LEARNING MODEL

- Model parameters are adjusted during model training to change input-output mapping
- Parameters are learned or estimated from data
  - "fitting the model", "training the model", "building the model"
- Goal: Minimize some error function

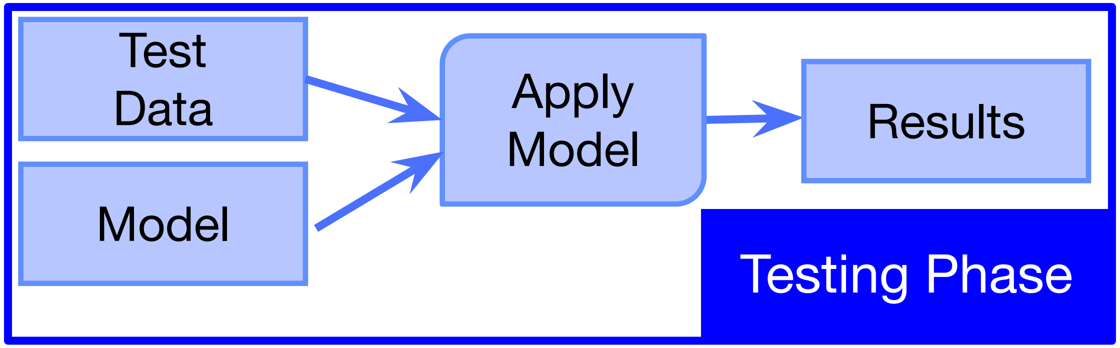


# BUILDING VS APPLYING MODEL




Adjust model parameters
"Train"

Test model on new data
"Inference"




**How do you evaluate a model?**

# TYPES OF CLASSIFICATION ERRORS

Is this animal a mammal?

$\underbrace{\text{Yes} \quad \text{No}}_{\text{Class Labels}}$

| True Label | Predicted Label | |
|---|---|---|
| Yes | Yes | True Positive (TP) |
| No | No | True Negative (TN) |
| No | Yes | False Positive (FP) |
| Yes | No | False Negative (FN) |

# ACCURACY RATE

| True | Predicted | Error |
|---|---|---|
| Yes | Yes | True Positive (TP) |
| No | No | True Negative (TN) |
| No | Yes | False Positive (FP) |
| Yes | No | False Negative (FN) |

$$\text{Accuracy Rate} = \frac{\text{\# correct predictions}}{\text{\# total predictions}}$$

$$= \frac{TP + TN}{TP + TN + FP + FN}$$

$$= (3 + 4) / 10 = 7 / 10 = 0.7$$

TP = 3
TN = 4
FP = 1
FN = 2
# samples = 10

# ERROR RATE

| True | Predicted | Error |
|---|---|---|
| Yes | Yes | True Positive (TP) |
| No | No | True Negative (TN) |
| No | Yes | False Positive (FP) |
| Yes | No | False Negative (FN) |

$$\text{Error Rate} = \frac{\text{\# incorrect predictions}}{\text{\# total predictions}}$$

$$= \frac{FN + FP}{TP + TN + FP + FN}$$

$$= 1 - \text{Accurate Rate} = 1 - 0.7 = 0.3$$

TP = 3
TN = 4
FP = 1
FN = 2
# samples = 10

# CONFUSION MATRIX

Is this animal a mammal?

*Yes* *No*

Class Labels

| | | Predicted Class Label | |
|---|---|---|---|
| True Class Label | | **Yes** | **No** |
| | **Yes** | True Positive (TP) | False Negative (FN) |
| | **No** | False Positive (FP) | True Negative (TN) |

# CONFUSION MATRIX & ACCURACY RATE

| | Predicted Class Label | | |
|---|---|---|---|
| True Class Label | | Yes | No |
| | Yes | TP = 3 | FN = 2 |
| | No | FP = 1 | TN = 4 |

$$\text{Accuracy Rate} = \frac{\text{\# correct predictions}}{\text{\# total predictions}}$$

$$= \frac{TP + TN}{TP + TN + FP + FN}$$

$$= (3 + 4) / 10 = 7 / 10 = 0.7$$

# CONFUSION MATRIX & ERROR RATE

| | Predicted Class Label | |
|---|---|---|
| **True Class Label** | **Yes** | **No** |
| **Yes** | **TP = 3** | **FN = 2** |
| **No** | **FP = 1** | **TN = 4** |

$$\text{Error Rate} = \frac{\#\text{ incorrect predictions}}{\#\text{ total predictions}}$$

$$= \frac{FN + FP}{TP + TN + FP + FN}$$

$$= 1 - \text{Accurate Rate} = 1 - 0.7 = 0.3$$

# CONFUSION MATRIX

| | Predicted Class Label | | |
|---|---|---|---|
| True Class Label | | Yes | No |
| | Yes | TP = 3 | FN = 2 |
| | No | FP = 1 | TN = 4 |

Want values on diagonal to be high, and values on off-diagonal to be low.

# LIMITATION WITH ACCURACY

Is this tumor cancerous?

most are negative examples

very few positive examples

Class Imbalance Problem

- Say 3% of samples are cancer
- If model <u>always</u> predicts non-cancer
  - Accuracy = 97%
  - But no cancer cases detected!

# PRECISION & RECALL

| True | Predicted | Error |
| --- | --- | --- |
| Yes | Yes | True Positive (TP) |
| No | No | True Negative (TN) |
| No | Yes | False Positive (FP) |
| Yes | No | False Negative (FN) |

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

All samples with
Predicted = Yes

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

All samples with
True = Yes

# PRECISION & RECALL

| True | Predicted | Error |
|---|---|---|
| Yes | Yes | True Positive (TP) |
| No | No | True Negative (TN) |
| No | Yes | False Positive (FP) |
| Yes | No | False Negative (FN) |

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{\text{Positive samples correctly predicted}}{\text{All samples predicted as Positive}}$$

Measure of exactness

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{\text{Positive samples correctly predicted}}{\text{All samples with true label Positive}}$$

Measure of completene

Precision = Positive Predictive Value
Recall = Sensitivity

# F-MEASURE

**Precision** + **Recall**

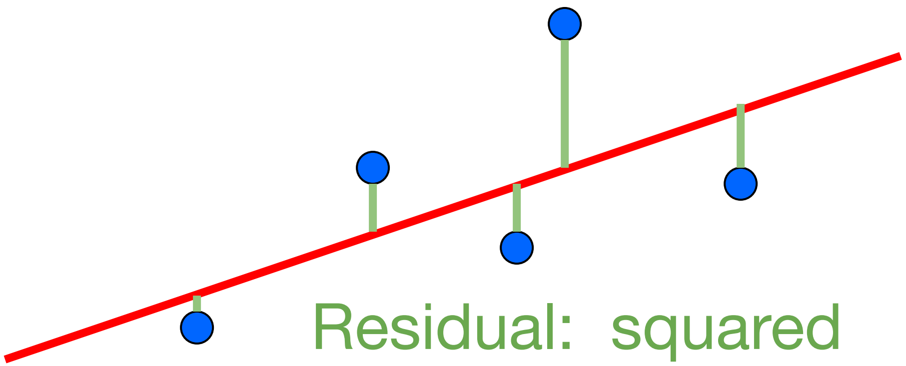
$$F_1 = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

- $F_1$ : evenly weighted
- $F_2$ : weights Recall more
- $F_{0.5}$: weights Precision more

F1-score is harmonic mean of precision and recall
F1-score ranges from 0 to 1 (1: perfect precision & recall)

# LINEAR REGRESSION

Goal: Find regression line that minimizes sum of residuals



regression line

sample

distance from regression line (error)

Residual: squared distance
from regression line

# REGRESSION EVALUATION METRICS

Mean Squared Error $$\text{MSE}(y, \hat{y}) = \frac{1}{n_\text{samples}} \sum_{i=0}^{n_\text{samples}-1} (y_i - \hat{y}_i)^2.$$

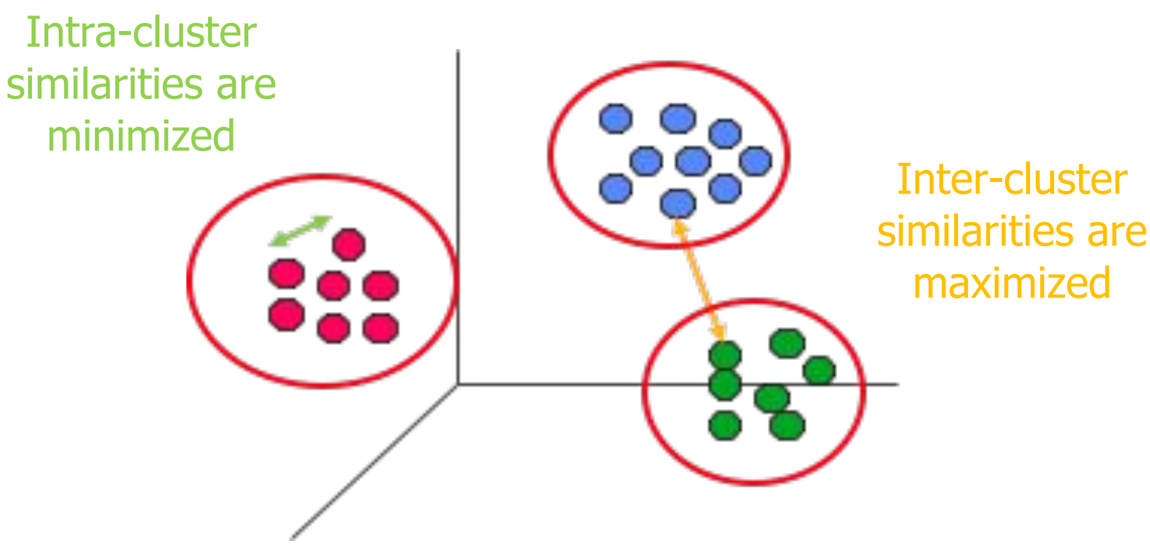
Root Mean Squared Error $$\text{RMSE} = \sqrt{\text{MSE}}$$

Mean Absolute Error $$\text{MAE}(y, \hat{y}) = \frac{1}{n_\text{samples}} \sum_{i=0}^{n_\text{samples}-1} |y_i - \hat{y}_i|$$

R-Squared $$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2}$$

# CLUSTER ANALYSIS

- Cluster analysis divides data into groups
  - Grouping is based on some similarity measure.
  - Samples within a cluster are more similar to each other than to samples in other clusters.




http://www-users.cs.umn.edu/~kumar/dmbook/index.php

# EVALUATING CLUSTERING RESULTS

- Within-Cluster Sum of Squared Error (WSSE)
- For each sample, error is distance to centroid. Then, WSSE is computed as:

$$WSSE = \sum_{i=1}^{K} \sum_{x \in C_i} \| x - m_i \|^2$$

  - *x:* data sample in cluster $C_i$
  - $m_i$: cluster centroid (i.e., mean of cluster)
  - $\|x - m_i\|^2$: Euclidean distance between $m_i$ and $x$

# EVALUATING CLUSTERING RESULTS

- $WSSE_1 < WSSE_2$
  - Means that $WSSE_1$ is better numerically
- Caveats
  - Does not mean that clustering 1 is more 'correct' than clustering 2
  - Larger values of k will always reduce WSSE
- Clustering results need interpretation!

# CLUSTER EVALUATION METRICS

- WSSE
  - Within-cluster sum-of-squared error
  - Measures cluster cohesion

$$WSS = \sum_i \sum_{x \in C_i} (x - c_i)^2$$

- BSSE
  - Between-cluster sum-of-squared error
  - Measures cluster separation

$$BSS = \sum_i m_i (c - c_i)^2$$

- Davies-Bouldin Index (DBI)
  - Measures both cluster cohesion and cluster separation

$$DB = \frac{1}{n} \sum_{i=1}^{n} \max_{j \neq i} \left( \frac{\sigma_i + \sigma_j}{d(c_i, c_j)} \right)$$

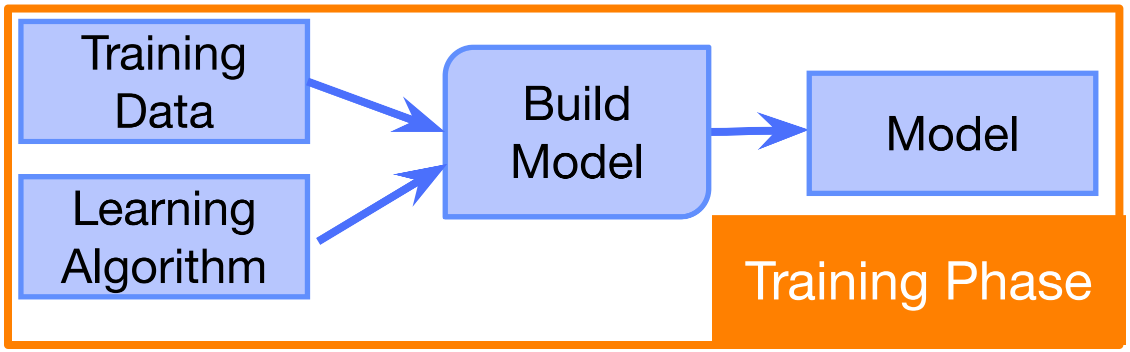
# EVALUATION METRICS

- Classification
  - Defined in terms types of classification errors
- Regression
  - Defined in terms of difference between prediction and target
- Cluster Analysis
  - Defined in terms of cluster cohesion & separation

# MODEL EVALUATION

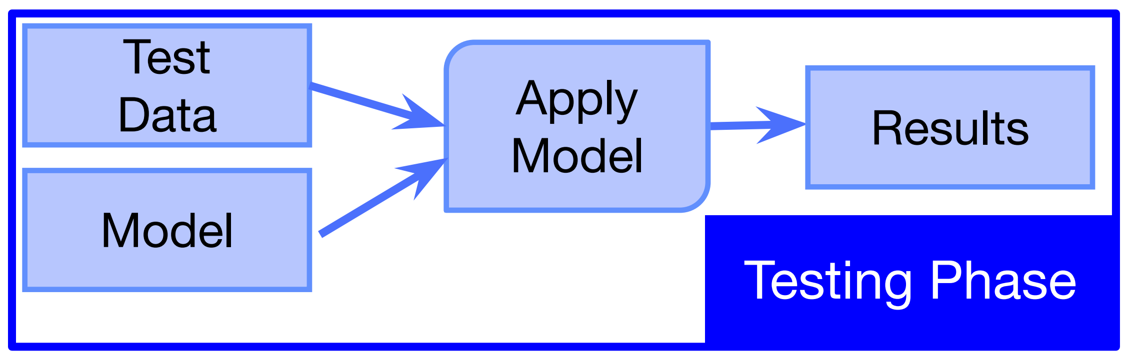
- Evaluation Metrics
- Generalization & Overfitting
- Model Selection & Model Evaluation
- Hyperparameter Tuning
- Ensemble Learning

# BUILDING VS APPLYING MODEL

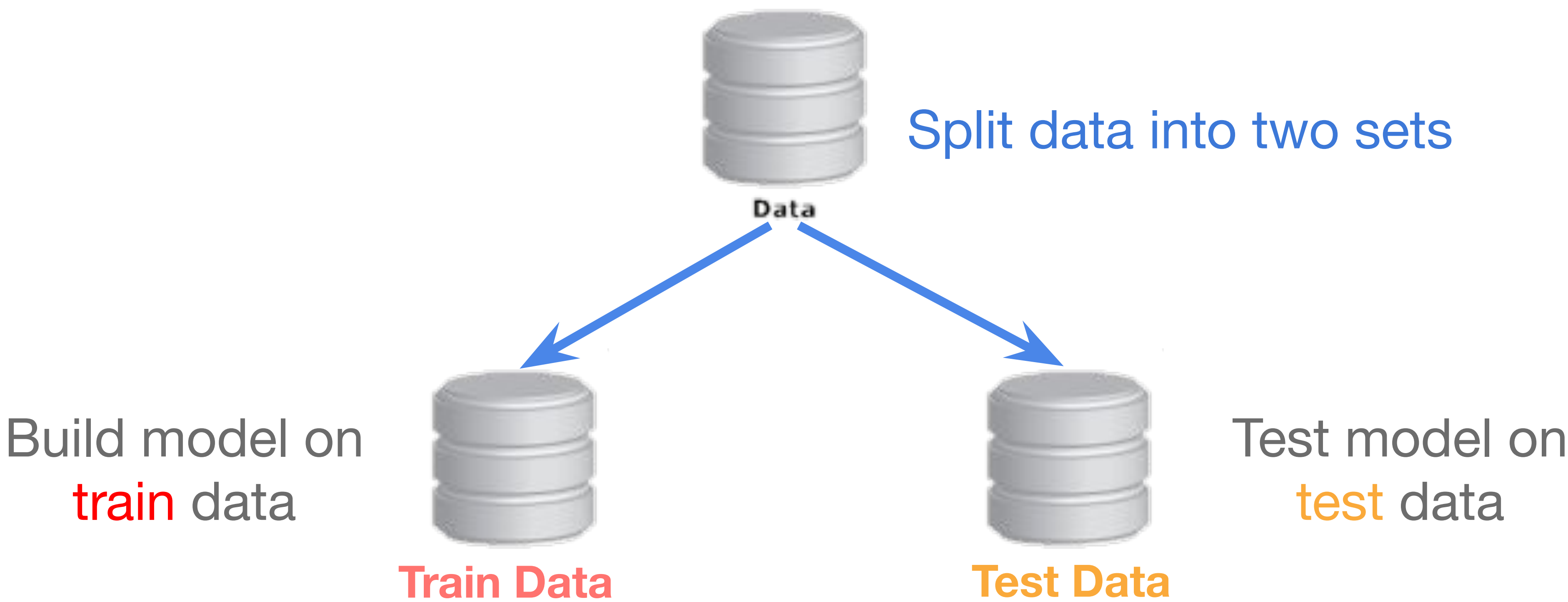



Adjust model parameters
"Train"

Test model on new data
"Inference"



# GENERALIZATION

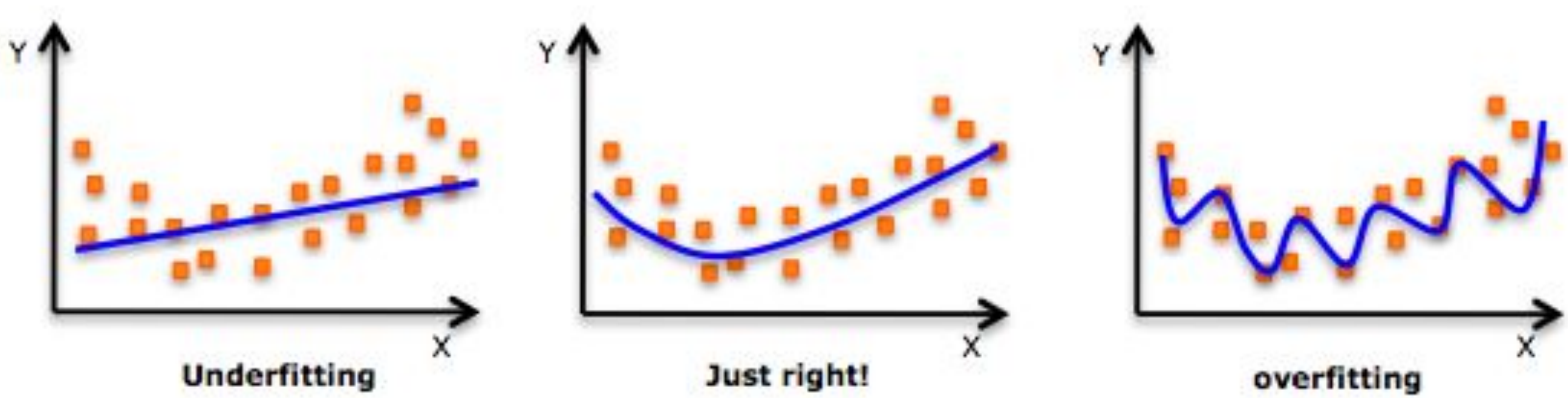



Goal: Want model to perform well on data it was not trained on, i.e., to **generalize** well to unseen data

# OVERFITTING & GENERALIZATION

- Overfitting
  - Model is fitting to noise in data instead of to underlying distribution of data
- Overfitting leads to poor generalization
  - Model that overfits will not generalize well to new data
- Reasons for overfitting
  - Training set is too small
  - Model is too complex, i.e., has too many parameters

# OVERFITTING




http://stats.stackexchange.com/questions/192007/what-measures-you-look-at-the-determine-over-fitting-in-linear-regression

**Underfitting**
Model has not learned structure of data

High training error
High test error

**Just Right**
Model has learned distribution of data

Low training error
Low test error

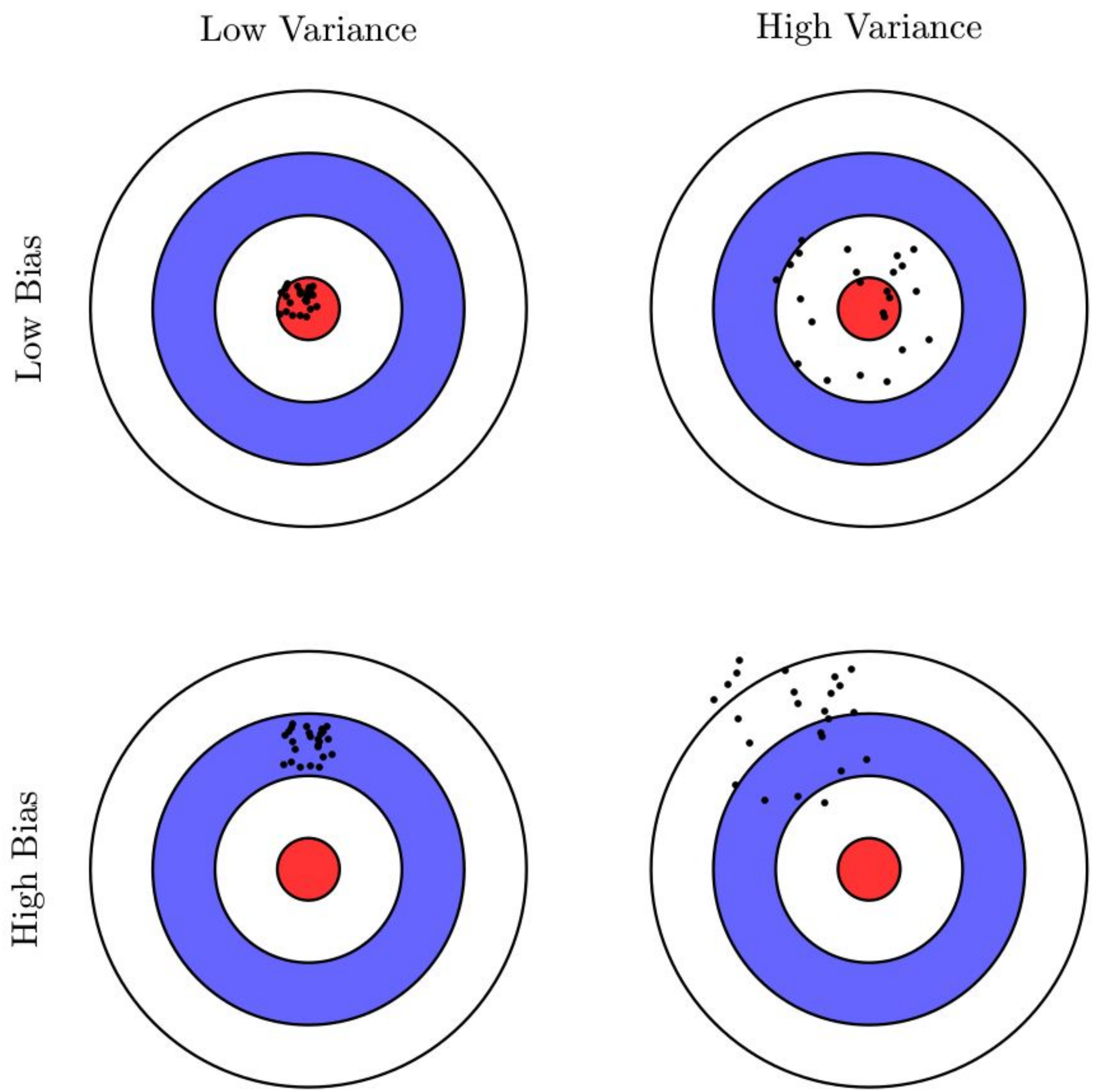
**Overfitting**
Model is fitting to noise in data

Low training error
High test error

# BIAS & VARIANCE

- Overfitting & Bias-Variance
  - Overfitting leads to poor generalization
  - Related to bias-variance trade-off in statistical learning
- Components of model generalization error
  - **Bias**
    - Error made by model based on assumptions in learning algorithm
  - **Variance**
    - Error from algorithm's sensitivity to variabilities in training data

# BIAS & VARIANCE

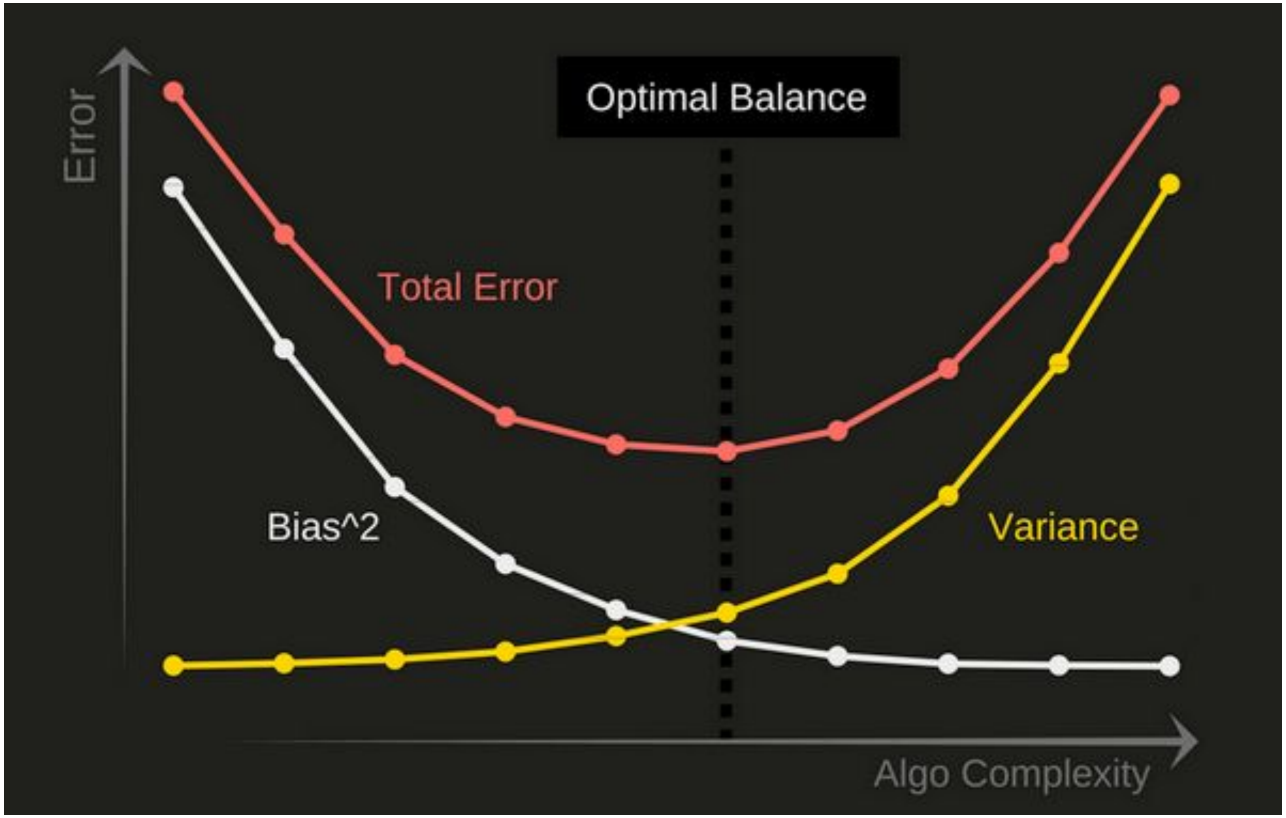



https://adrianromano.com/machine-learning-model-underfitting-vs-overfitting-and-how-to-achieve-the-best-fit-model/

# BIAS-VARIANCE TRADEOFF

- To create model that generalizes well
  - Need to balance bias and variance to minimize total error

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



# MODEL EVALUATION

- Evaluation Metrics
- Generalization & Overfitting
- Model Selection & Model Evaluation
- Hyperparameter Tuning
- Ensemble Learning

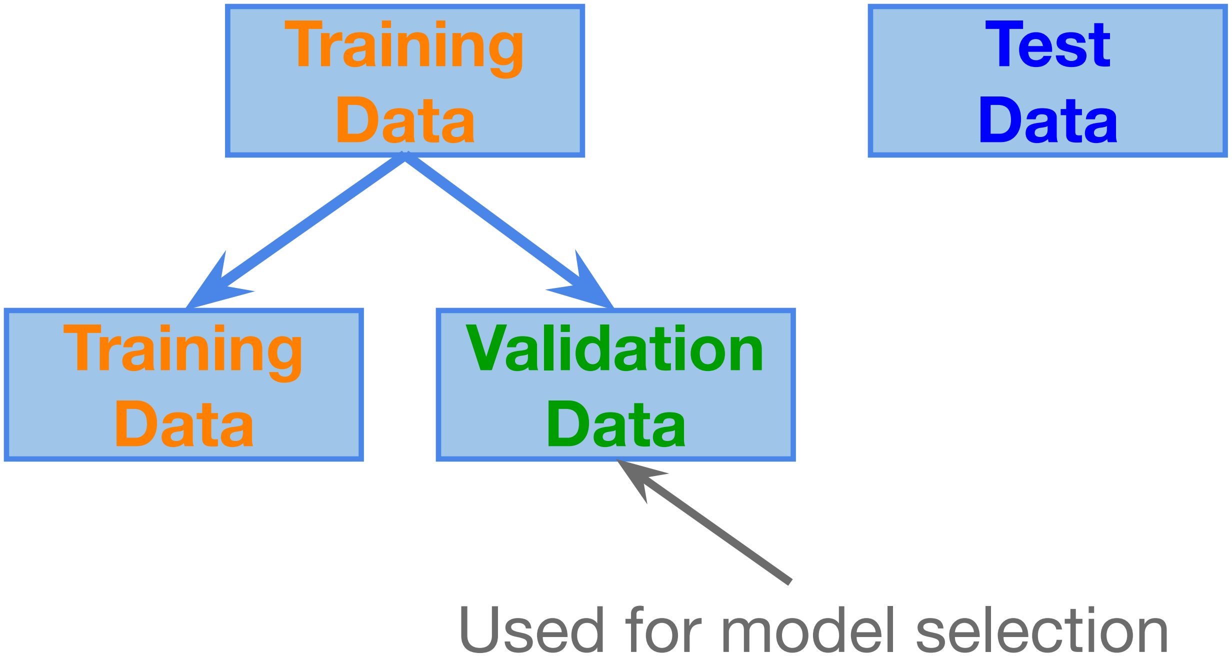
# ADDRESSING OVERFITTING

- Model Selection
  - Process of selecting a model from collection of candidate models
  - Based on model complexity
    - Selecting model with right level of complexity
    - To address overfitting and maximize generalization
  - Methods
    - Use a validation set
    - Incorporate model complexity in error function

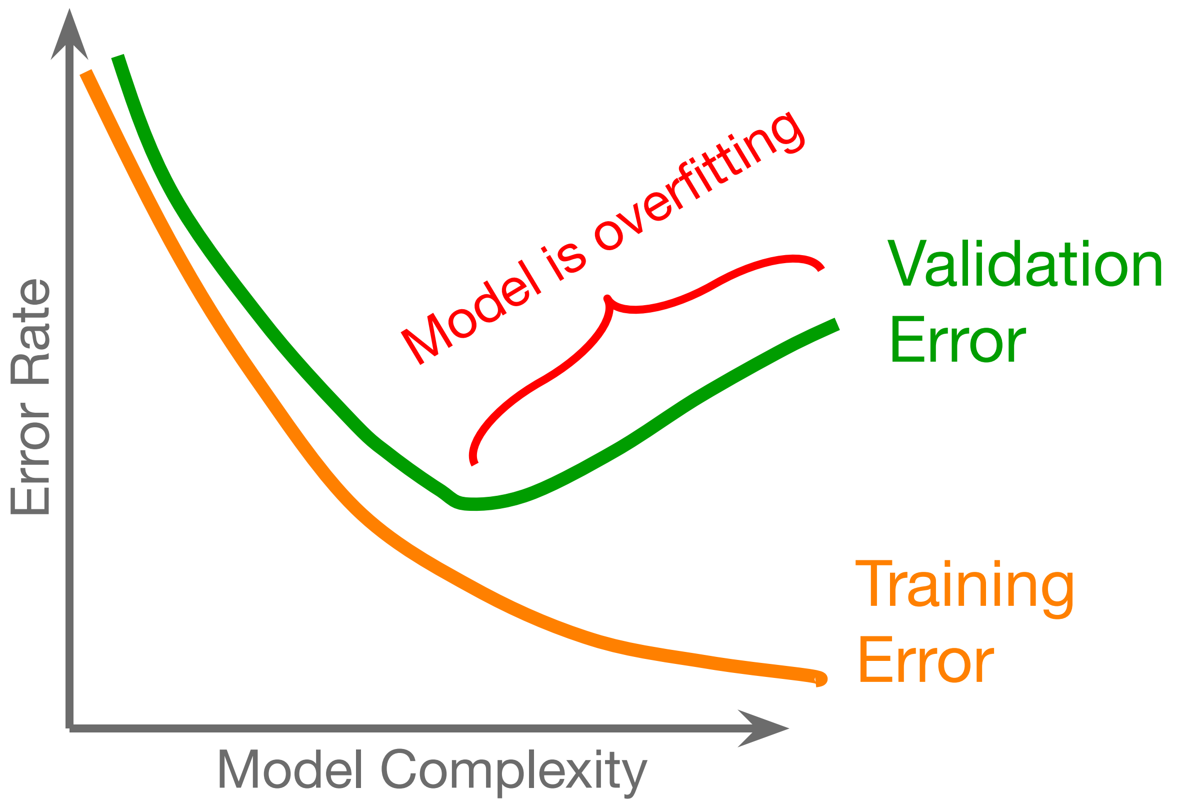
# MODEL SELECTION

- Considerations
  - Complexity
  - Accuracy
  - Interpretability
  - Computational and/or memory efficiency
  - Scalability
  - others

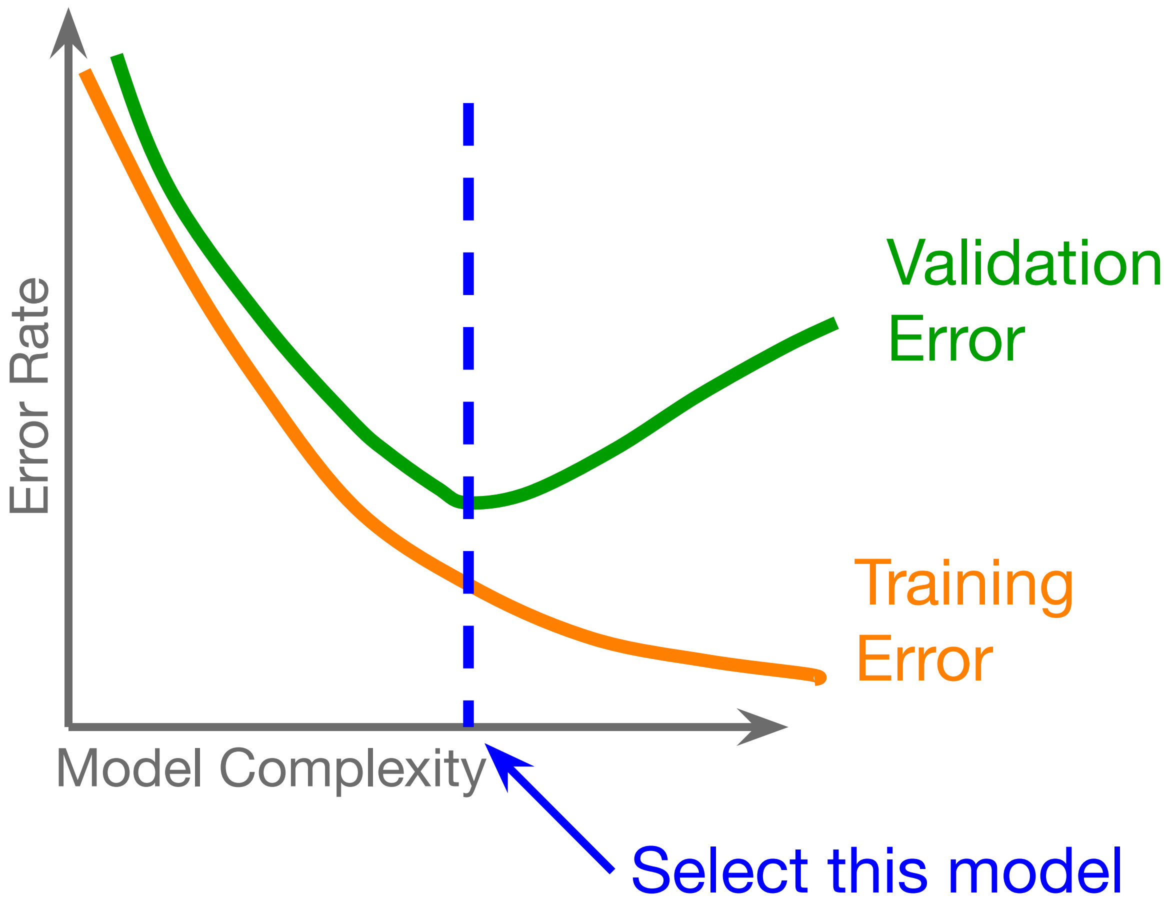
# ADDRESSING OVERFITTING USING VALIDATION SET



# TRAINING & VALIDATION ERRORS



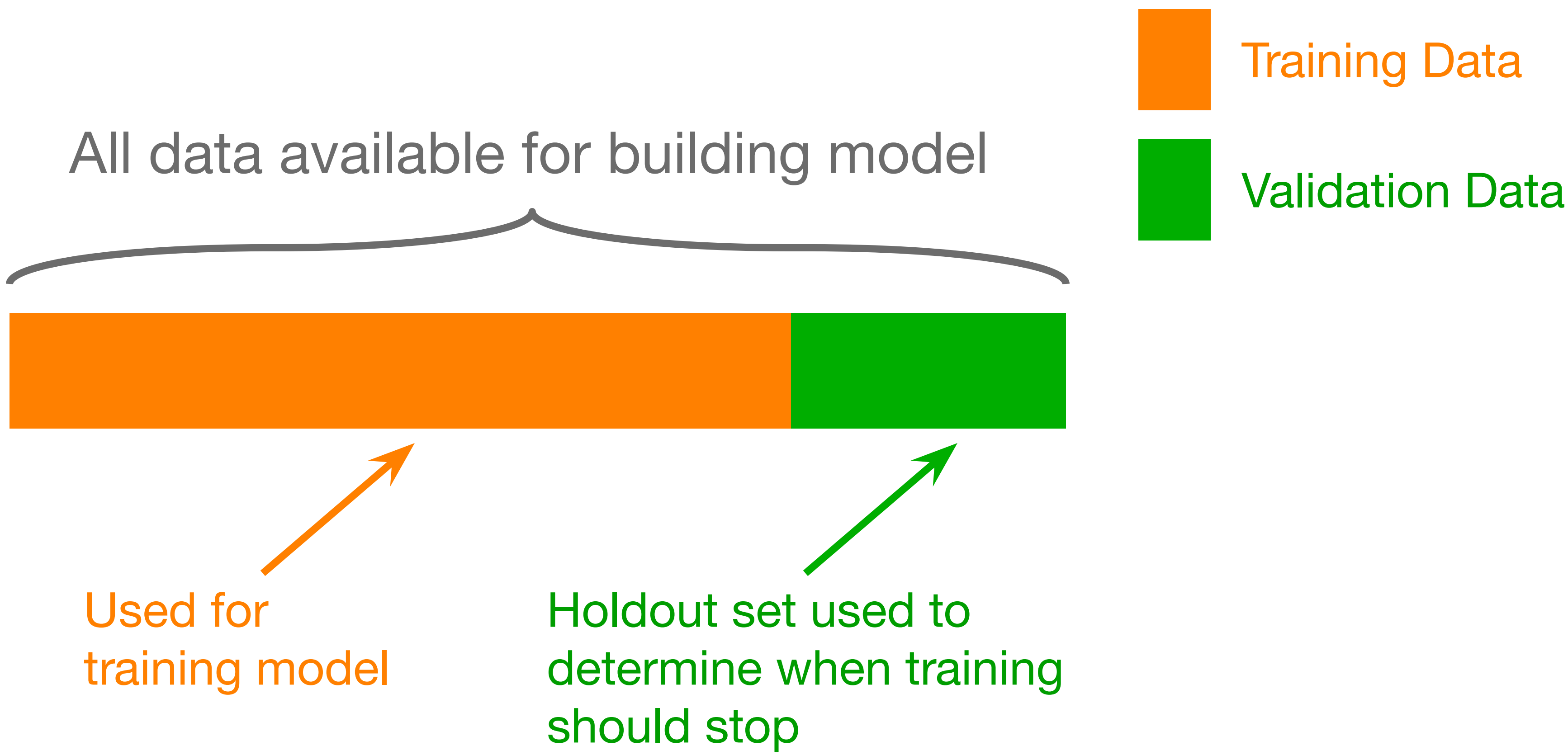
# MODEL SELECTION



# VALIDATION SET

- Ways to create & use validation set
  - Holdout method
  - Repeated holdout
  - K-fold cross-validation
  - Leave-one-out cross-validation

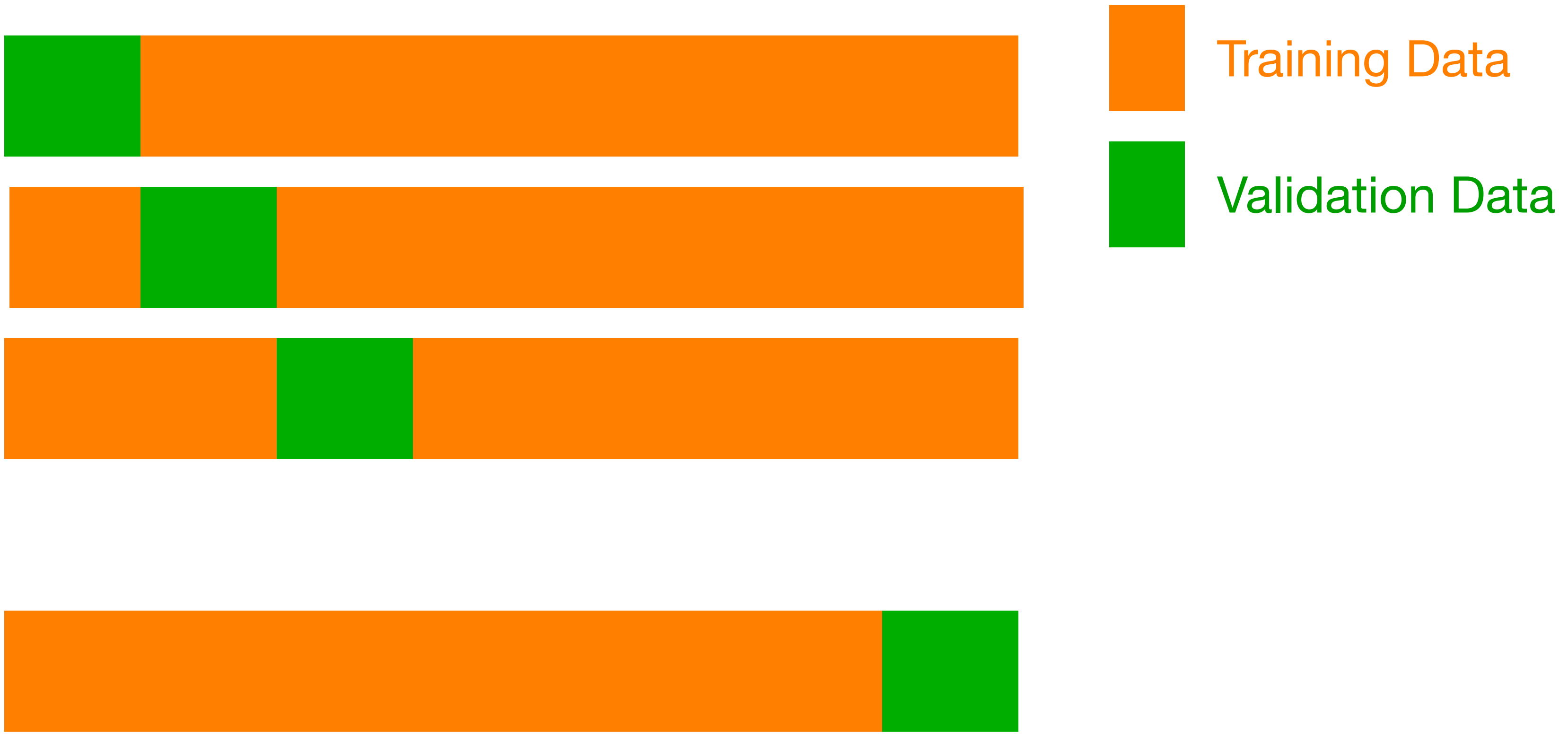
# HOLDOUT METHOD



# REPEATED HOLDOUT


- Repeating holdout method several times
- Randomly select different hold out set each iteration
- Average validation errors over all repetitions

# K-FOLD CROSS-VALIDATION



# LEAVE-ONE-OUT CROSS-VALIDATION

# ADDRESSING OVERFITTING USING REGULARIZATION

- Model complexity
  - Number of parameters in model
  - Chance of overfitting increases with model complexity
- Regularization
  - Constrain or shrink ("regularize") model parameters
  - To control model complexity by reducing variance of model
  - Add penalty term to error function used to train model
    - e.g., to discourage large values of parameters

# LINEAR REGRESSION WITH REGULARIZATION

- Linear regression with regularization
- Ridge regression: L2-norm regularization

$$\min_{w} ||Xw - y||_2^2 + \alpha ||w||_2^2$$

- Lasso regression: L1-norm regularization

$$\min_{w} ||Xw - y||_2^2 + \alpha ||w||_1$$

- Elastic regression: L2-norm & L1-norm regularization

# MODEL EVALUATION

- Model Evaluation
  - Assessing performance of a trained model

- Estimating generalization performance of model
  - Methods for model selection can be used

# DATASETS

Cannot be used in <u>any</u> way in model fitting!

**Training Data**

**Validation Data**

**Test Data**

**Model Fitting:**
Adjust model parameters

**Model Selection:**
Select model to avoid overfitting
Estimate generalization performance

**Model Evaluation:**
Evaluate performance on new data

# MODEL EVALUATION

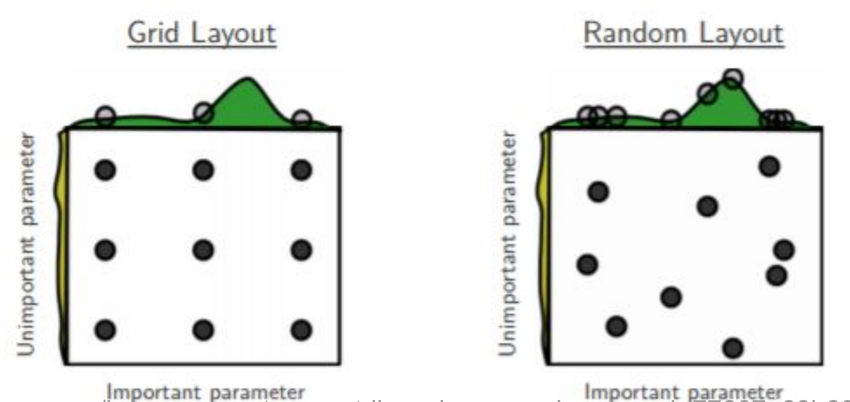
- Evaluation Metrics
- Generalization & Overfitting
- Model Selection & Model Evaluation
- Hyperparameter Tuning
- Ensemble Learning

# HYPERPARAMETER TUNING

- An approach to model selection
- Model parameters
  - Parameters that are learned from data
  - Adjusted *during* training
  - Example:  Feature to split on for each node in decision tree
- Model hyperparameters
  - Parameters that determine model architecture
  - Knobs of model that can be tuned to improve performance
    - “Hyperparameter Tuning”
  - Must be set *before* training begins
  - Example:  max depth of decision tree

# HYPERPARAMETER TUNING METHODS

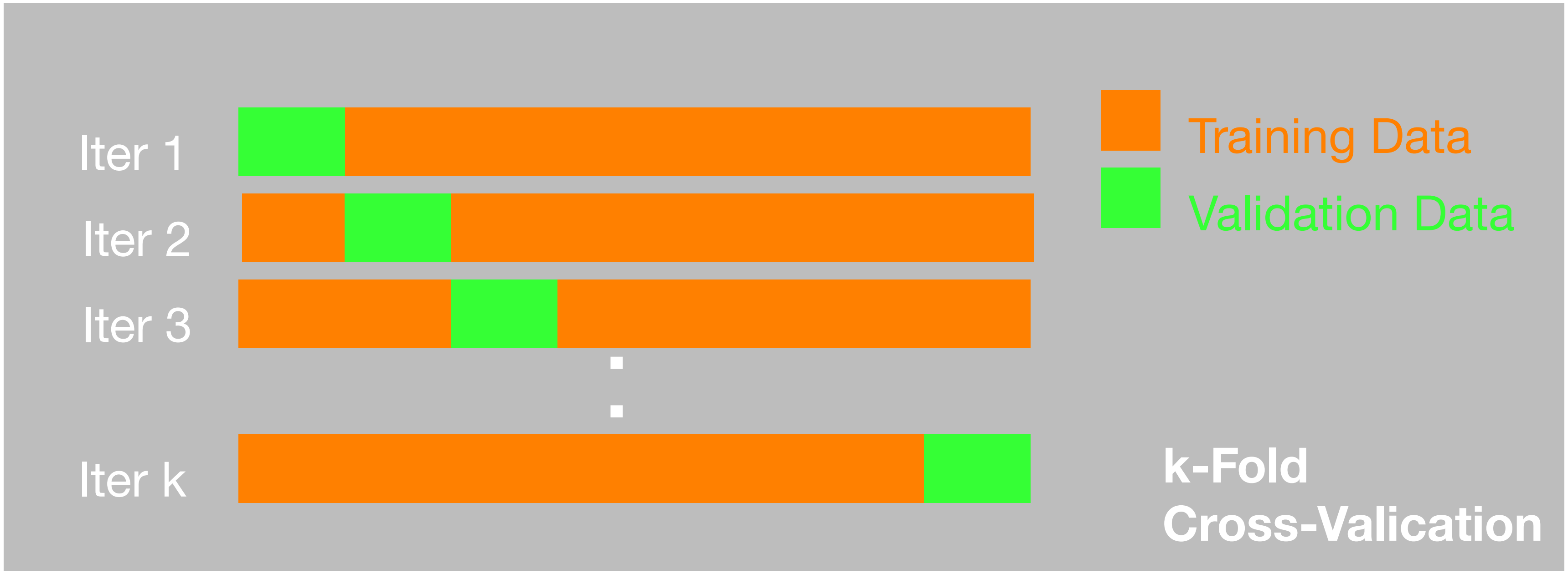
- Idea:
  - Evaluate range of values for each hyperparameter
  - Evaluate model for each combination of hyperparameter values to get optimal set
- Grid search
  - Exhaustive approach
  - Specify list of values for each hyperparameter
- Random search
  - Specify range of values for each hyperparameter
  - Hyperparameter values are randomly sampled from given range based on specified statistical distribution




https://towardsdatascience.com/hyperparameters-part-ii-random-search-on-spark-77667e68b606

# CROSS-VALIDATION IN HYPERPARAMETER TUNING

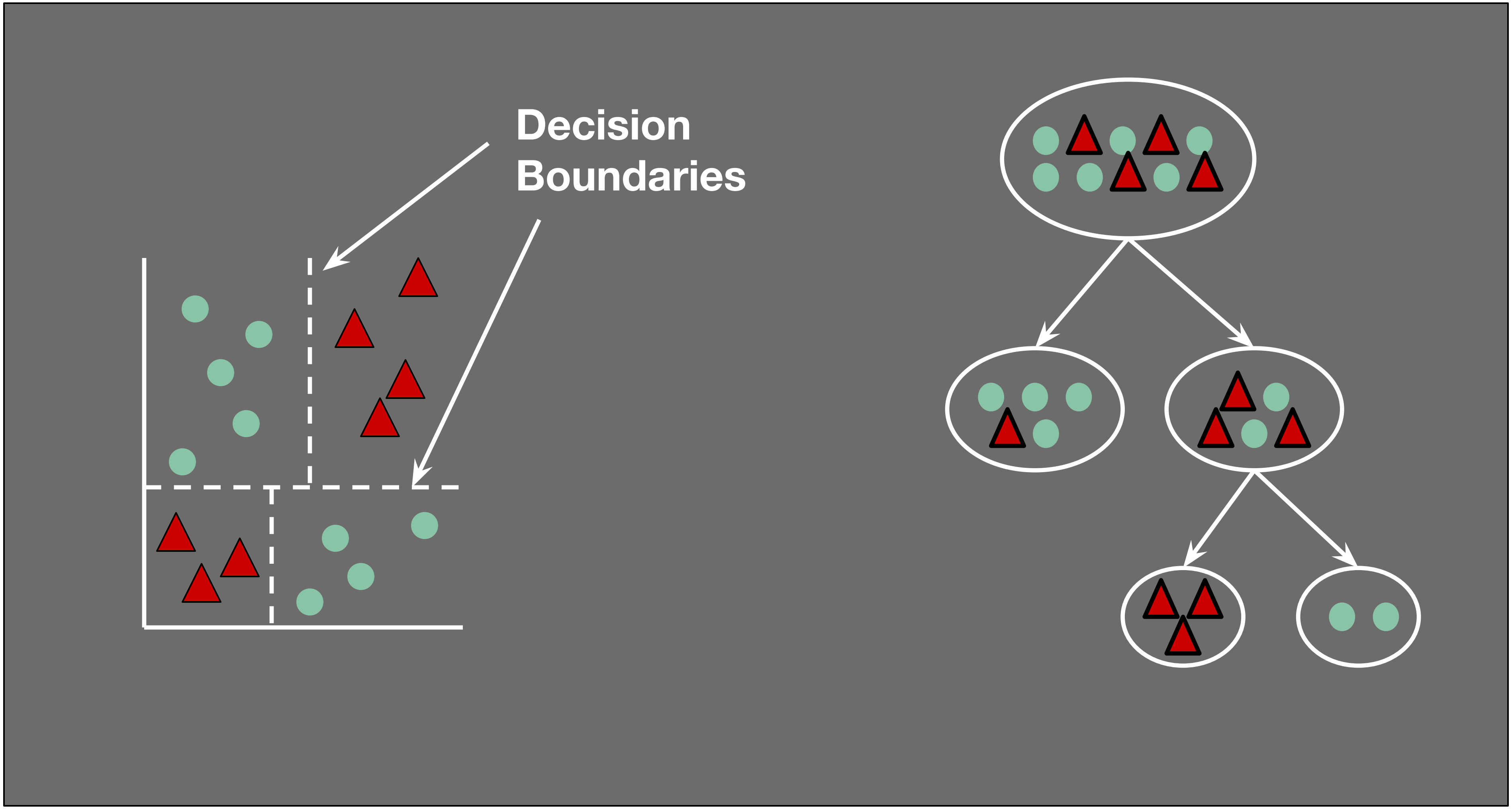
Evaluate model on each combination of hyperparameter values using k-fold cross-validation



# MODEL EVALUATION

- Evaluation Metrics
- Generalization & Overfitting
- Model Selection & Model Evaluation
- Hyperparameter Tuning
- Ensemble Learning

# DECISION TREE



# ENSEMBLE METHODS

- “ensemble”:
  - a group producing a single effect (from Merriam-Webster)
- Idea:
  - Combine several simple models into more complex one
- Approach:
  - Construct a set of models from training data
  - Prediction is made by combining outputs of the multiple models
    - Classification: Combine votes of classifiers

# ENSEMBLE METHODS

- Advantage
  - Ensemble learning generates more robust model with is less susceptible to overfitting and generalizes better
- Rationale
  - Ensemble with majority voting
    - Base classifiers may make mistakes, but ensemble will misclassify a pattern only if over half of base classifiers are incorrect.
  - Intuitively, combining decisions from multiple “experts” may be more reliable than relying on a single “expert”
- Approaches
  - Bagging
  - Boosting

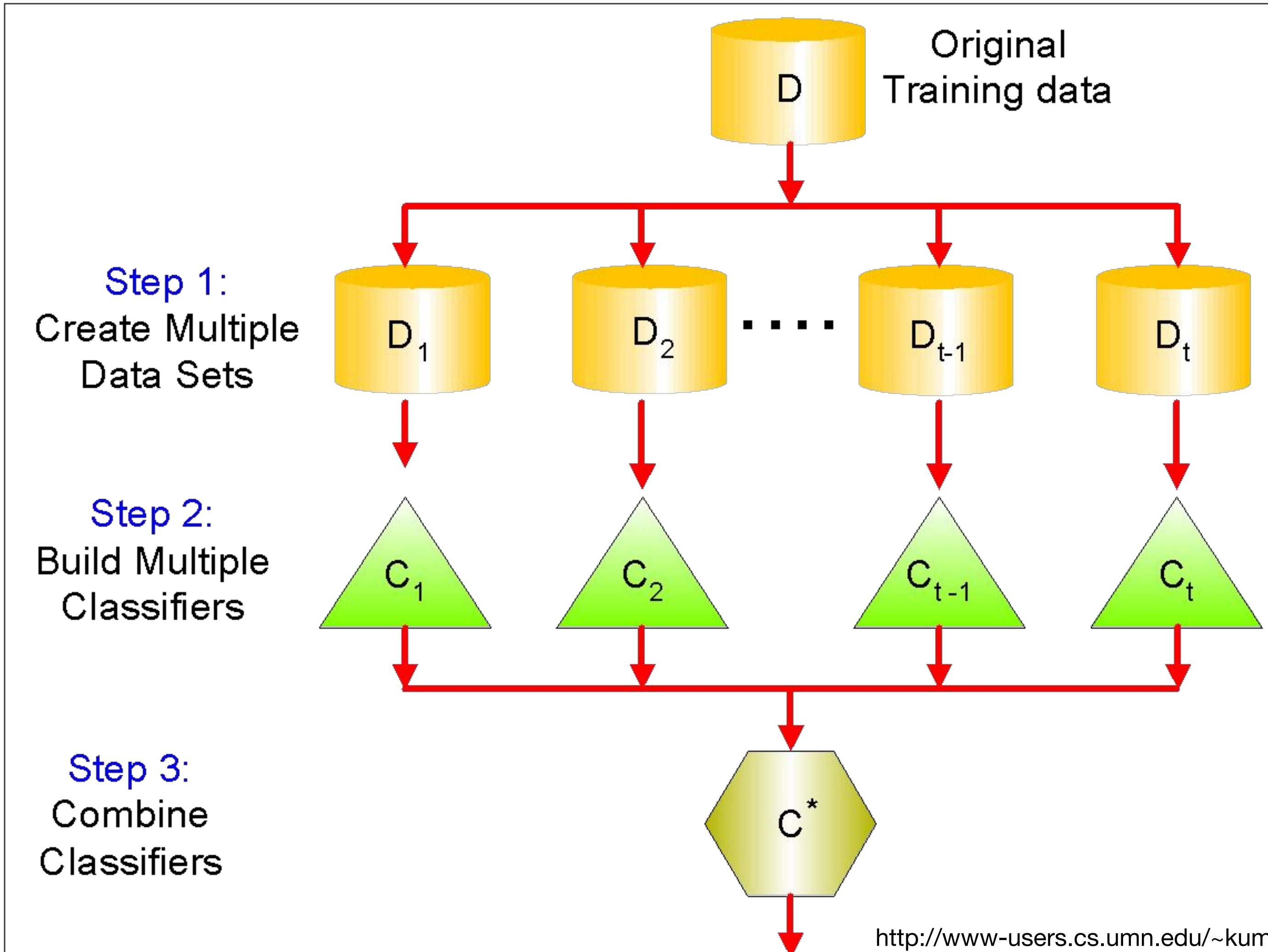
# ENSEMBLE METHOD: BAGGING

- Bagging stands for "bootstrap aggregation"
- Approach:
  - Sample training data set with replacement to construct bootstrap samples
  - Build separate classifier on each bootstrap sample
  - Each classifier predicts class label for unknown record
  - Bagged classifier takes majority vote
- Generalization can be improved since variance of individual base classifiers is reduced

# RANDOM FOREST

- "forest"
  - Ensemble method
    - Model is composed of set of decision trees => forest!
- "random"
  - For each tree is trained on randomly selected training samples
  - Subset of variables chosen randomly is used to determine best split
- Idea:
  - To improve generalization over single decision tree

# RANDOM FOREST - ILLUSTRATION



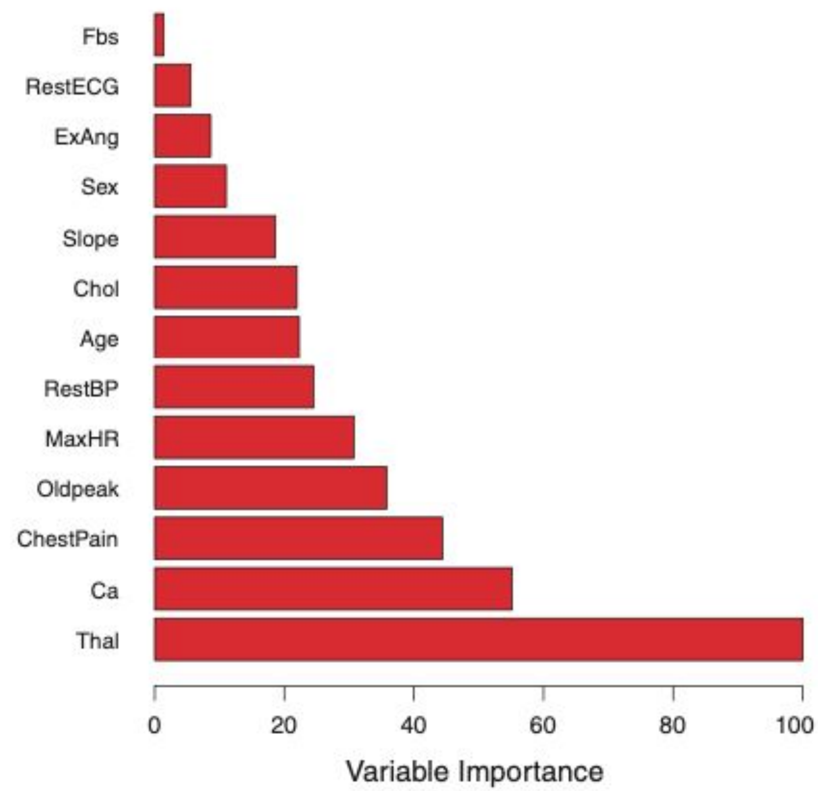

http://www-users.cs.umn.edu/~kumar/dmbook/index.php

# RANDOMNESS IN RANDOM FOREST

- Randomness can be incorporated in several ways:
  - Forest-RI
    - Random attribute selection:  At each node, randomly select subset of input attributes to consider in splitting node
  - Forest-RC
    - Random combinations of attributes:  At each node, randomly select subset of input attributes to be linearly combined.  These new attributes are considered in splitting node
  - Randomly select best split:
    - At each node, randomly select one of F best splits

# FEATURE IMPORTANCE

- Importance of each variable in prediction task
- Provides model explainability
- Process
  - Keep track of decrease in Gini index when splitting by each feature
  - Add up for all splits in all base trees
  - Find average over all trees



# ENSEMBLE METHOD: BOOSTING

- Boosting
  - Combine set of “weak learners” (i.e., base models) to create composite strong learner
  - Base models added iteratively until no further improvements can be made or max number of models have been added
  - Weighted aggregation of base models’ outputs used as final prediction
  - Base models are created *sequentially*

# ADABOOST

- Adaptive Boosting
  - Adaptive: New models are built based on errors from previous ones
- Main ideas
  - Misclassified samples are weighted more
  - New models focus more on samples that are difficult for existing models
  - Models are weighted relative to their predictive performance
  - Final prediction is weighted average of base models

# XGBOOST

- Gradient Boosting
  - New models are trained to minimize residuals (i.e., errors) of existing models
  - Loss function combines error and penalty term for model complexity
  - Gradient descent used to minimize loss when adding new models
- eXtreme Gradient Boosting
  - Implementation of gradient boosted trees
  - Optimized for execution speed and model performance
  - Uses parallelization and distributed computing to speed computation

# ENSEMBLE MODELS

- In practice, often results in improved performance due to lower variance
- Training takes longer
- Ensembles more difficult to understand than single models.

# MODEL EVALUATION

- Evaluation Metrics
- Generalization & Overfitting
- Model Selection & Model Evaluation
- Hyperparameter Tuning
- Ensemble Learning