Introduction to machine learning

James Bourbeau

X-meeting

IceCube

03/21/17

What is machine learning?

Machine learning [m*uh*-**sheen lur**-ning]

Noun

Set of techniques and algorithms for gaining insight from data. Often with the goal of wanting to make predictions about future or unseen data.

Categories of machine learning

Supervised learning

Train a model using *labeled* training data in order to make prediction about future unseen data

Categories of machine learning

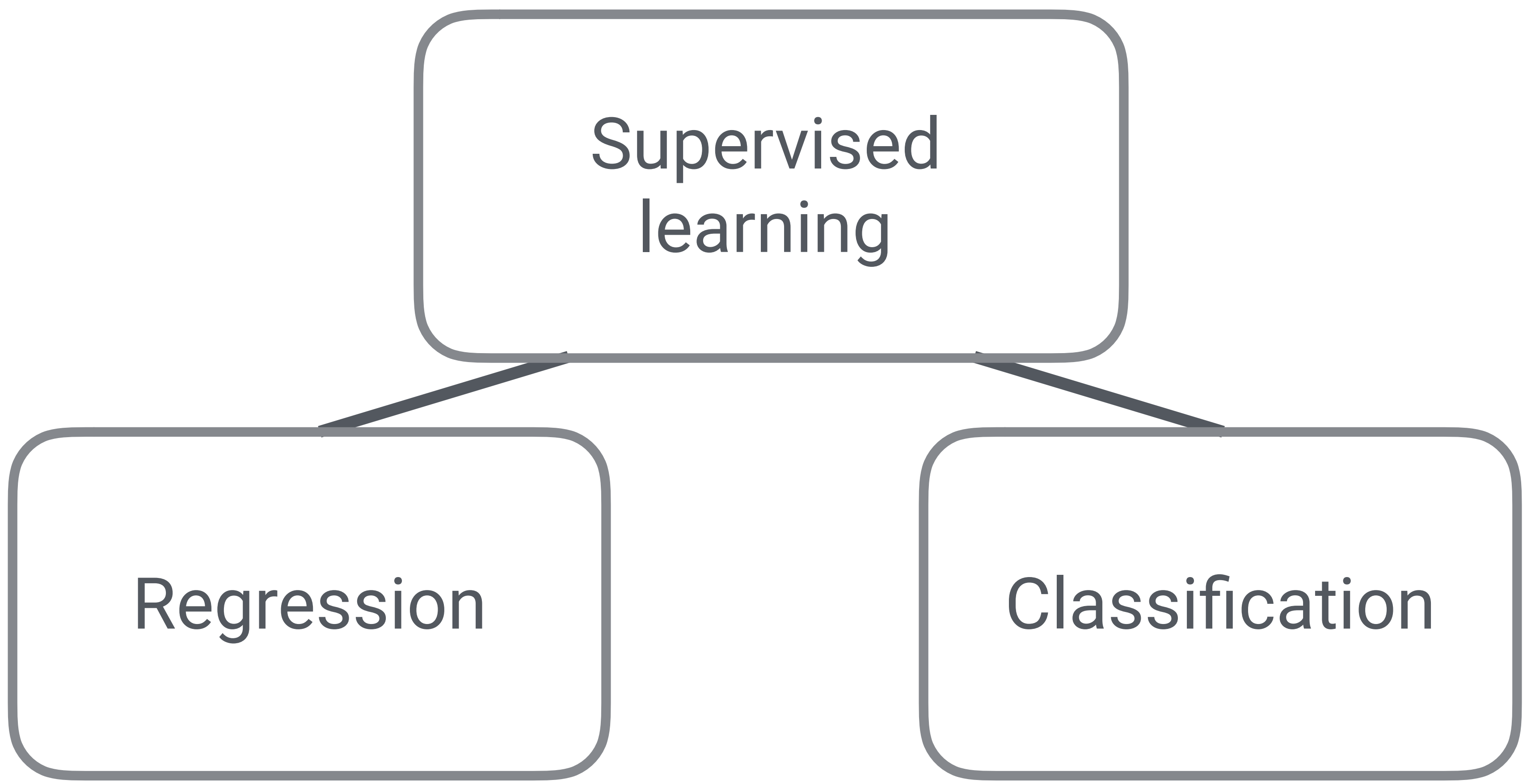
Supervised learning	Train a model using *labeled* training data in order to make prediction about future unseen data
Reinforcement learning	Agent maximizes some reward function via interacting with its environment

Categories of machine learning

Supervised learning	Train a model using *labeled* training data in order to make prediction about future unseen data
Reinforcement learning	Agent maximizes some reward function via interacting with its environment
Unsupervised learning	Train a model using *unlabeled* training data in order to find underlying structure in data

Supervised learning sub-categories

Train a model using *labeled* training data in order to make prediction about future unseen data



Target variable is continuous variable (e.g. energy of particle)

Target variable is a discrete label (e.g. ‘spam’ or ‘not spam’)

Training a model

Process where parameters in model that are learned from data are determined.

Training data

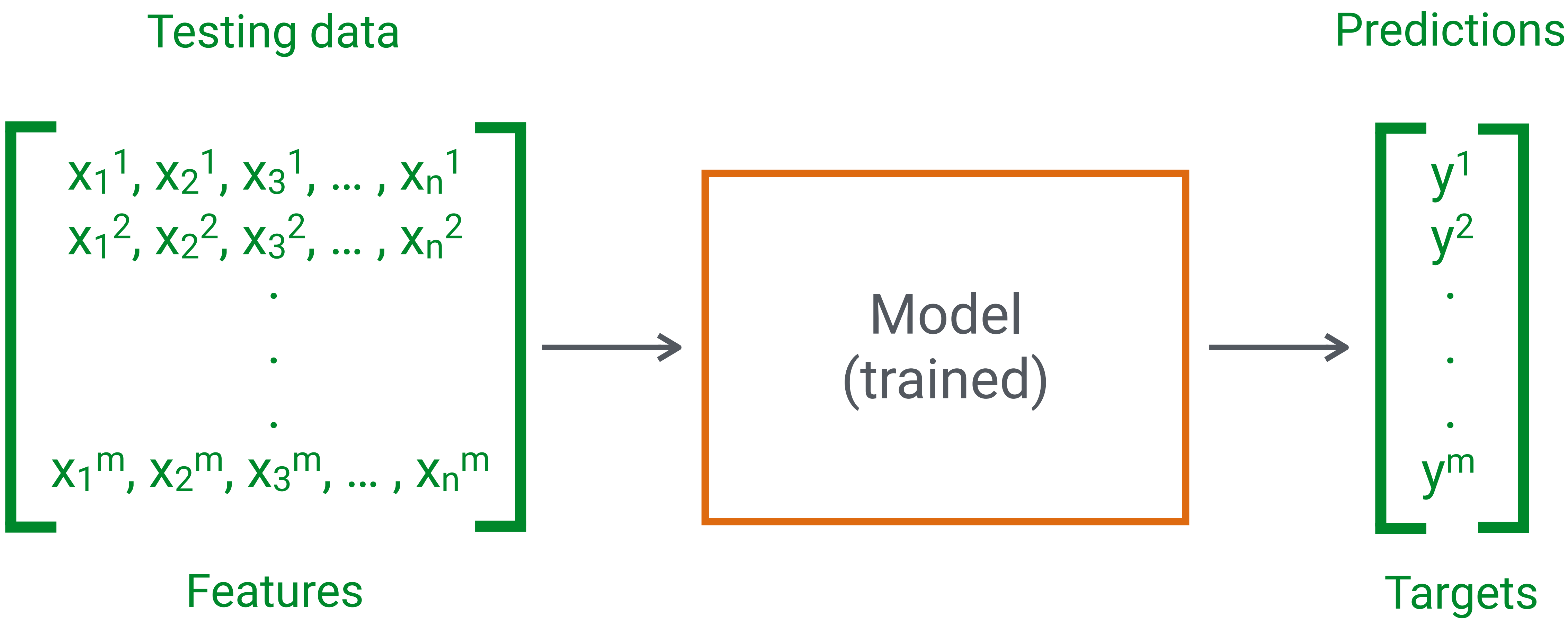
$$\begin{bmatrix} x_1^1, x_2^1, x_3^1, \dots, x_n^1 \\ x_1^2, x_2^2, x_3^2, \dots, x_n^2 \\ \vdots \\ x_1^m, x_2^m, x_3^m, \dots, x_n^m \end{bmatrix} \begin{bmatrix} y^1 \\ y^2 \\ \vdots \\ y^m \end{bmatrix} \longrightarrow \boxed{\text{Model}}$$

Features   Targets

Might be referred to as:

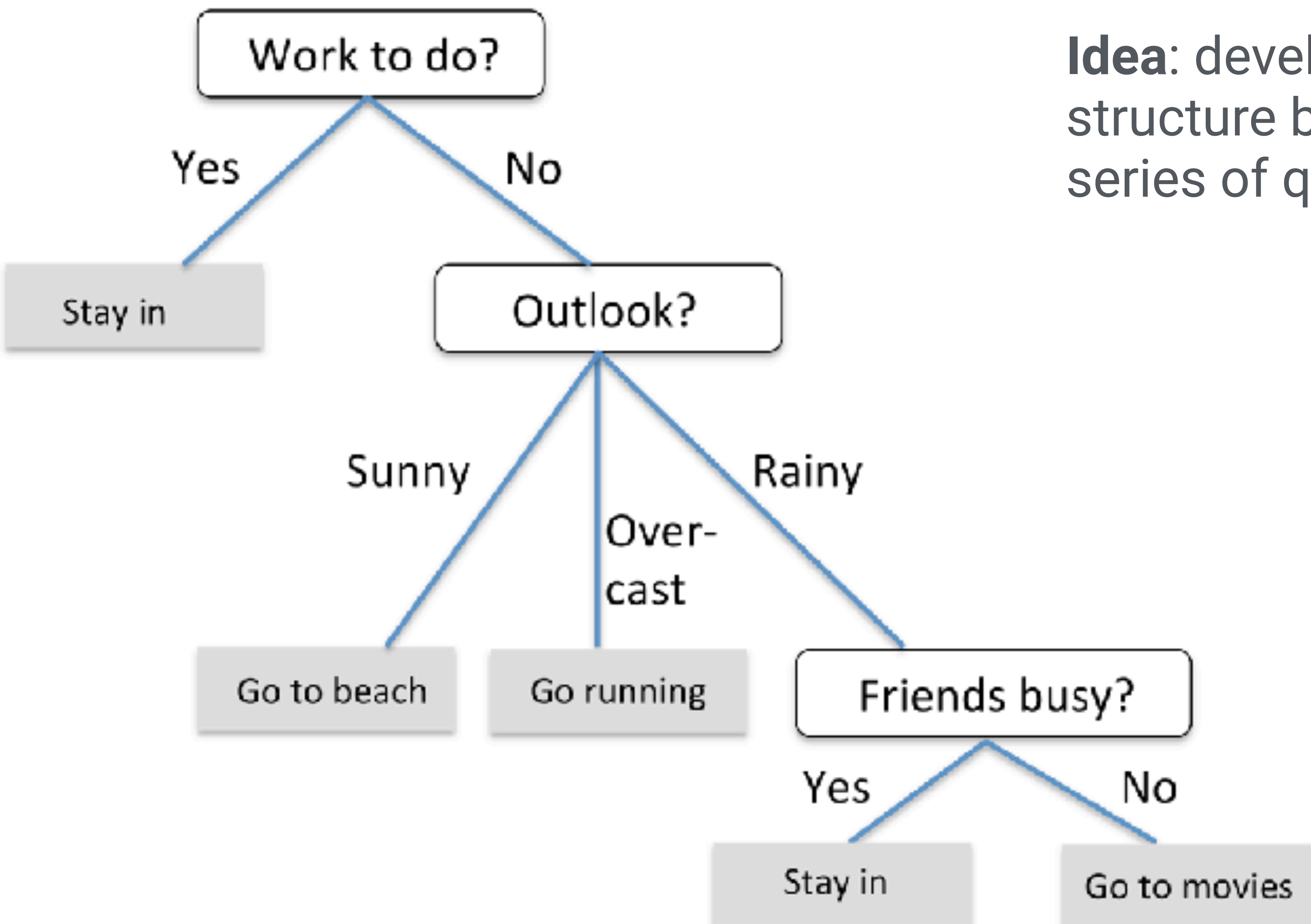
- training a model
- fitting a model
- learning a model

Making predictions



Tree-based learning

Tree-based learning—decision trees



Idea: develop a tree structure by asking a series of questions

Image credit: Sebastian Raschka, *Python Machine Learning*

Tree-based learning—node splitting

Question: how do you know what questions to ask?

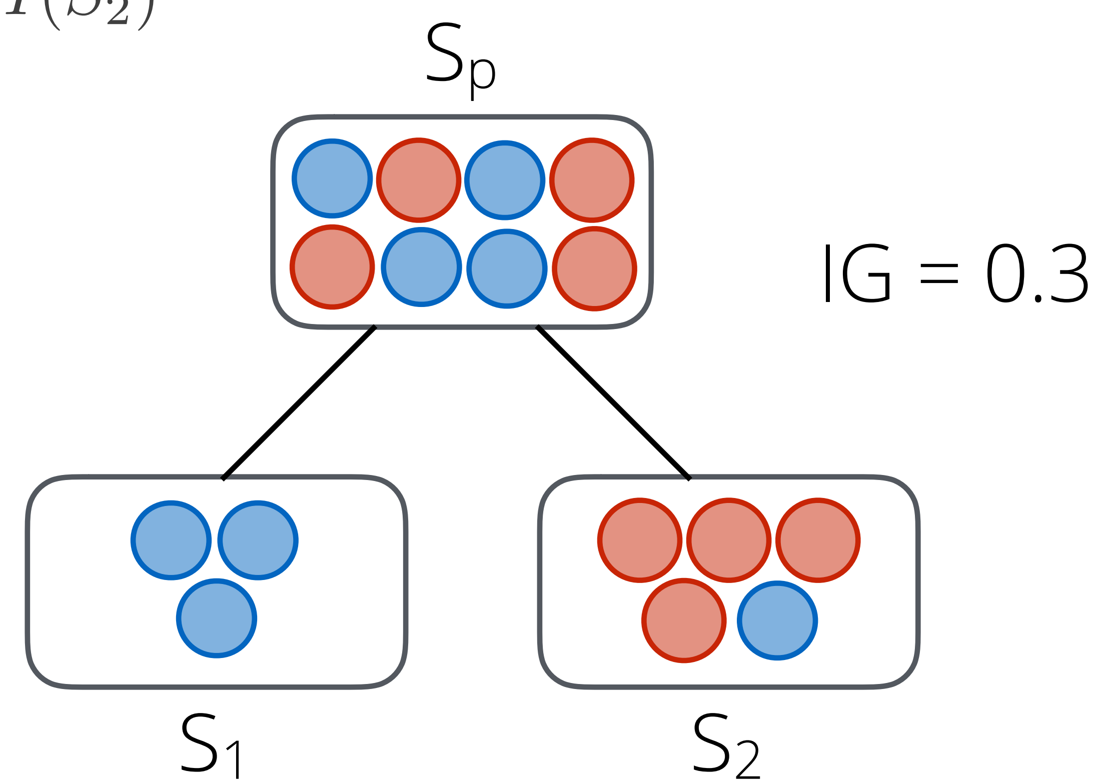
Answer: choose the splitting that maximizes the information gain.

Information gain:

$$IG(S_p, f) = I(S_p) - \frac{N_1}{N_p} I(S_1) - \frac{N_2}{N_p} I(S_2)$$

Gini impurity:

$$I_{\text{gini}}(S) = 1 - \sum_{i=1}^{N_{\text{class}}} p(i|S)^2$$



Tree-based learning—random forest

Idea: Combine many different decision trees to get better performance.

1. Draw n_{trees} bootstrap samples from training data
2. For each bootstrap sample, grow a decision tree.
   * At each node, find best split among random subset of $n_{features}$ of the training features
3. Classify new samples by aggregating the n_{trees} decision trees via a simple majority vote.

Tree-based learning—random forest

<u>Idea</u>: Combine many different decision trees to get better performance.

1. Draw n_{trees} bootstrap samples from training data
2. For each bootstrap sample, grow a decision tree.
   * At each node, find best split among random subset of $n_{features}$ of the training features
3. Classify new samples by aggregating the n_{trees} decision trees via a simple majority vote.

Tree-based learning—random forest

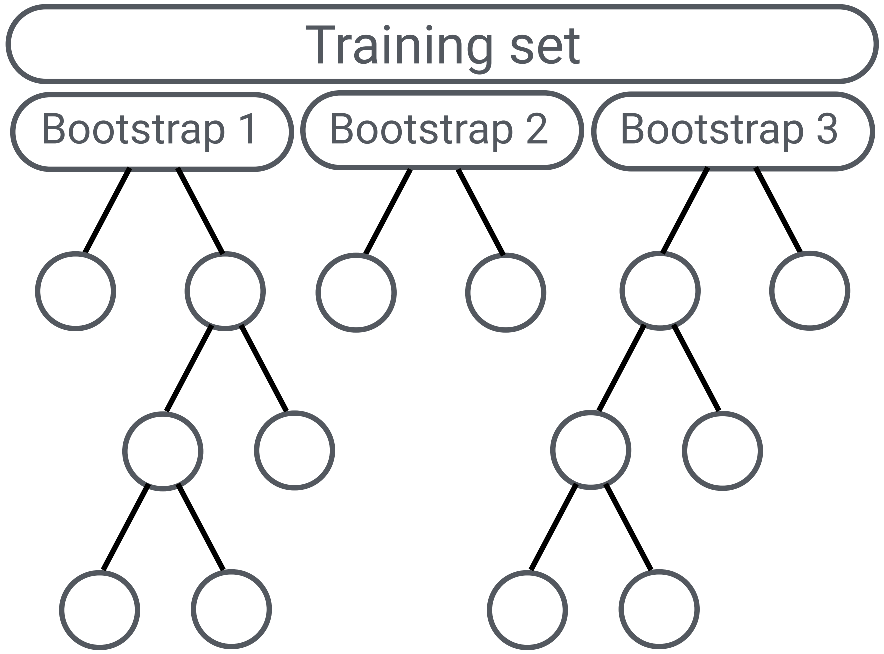
Idea: Combine many different decision trees to get better performance.

1. Draw n_{trees} bootstrap samples from training data
2. For each bootstrap sample, grow a decision tree.
   * At each node, find best split among random subset of $n_{features}$ of the training features
3. Classify new samples by aggregating the n_{trees} decision trees via a simple majority vote.

Tree-based learning—random forest

<u>Idea</u>: Combine many different decision trees to get better performance.

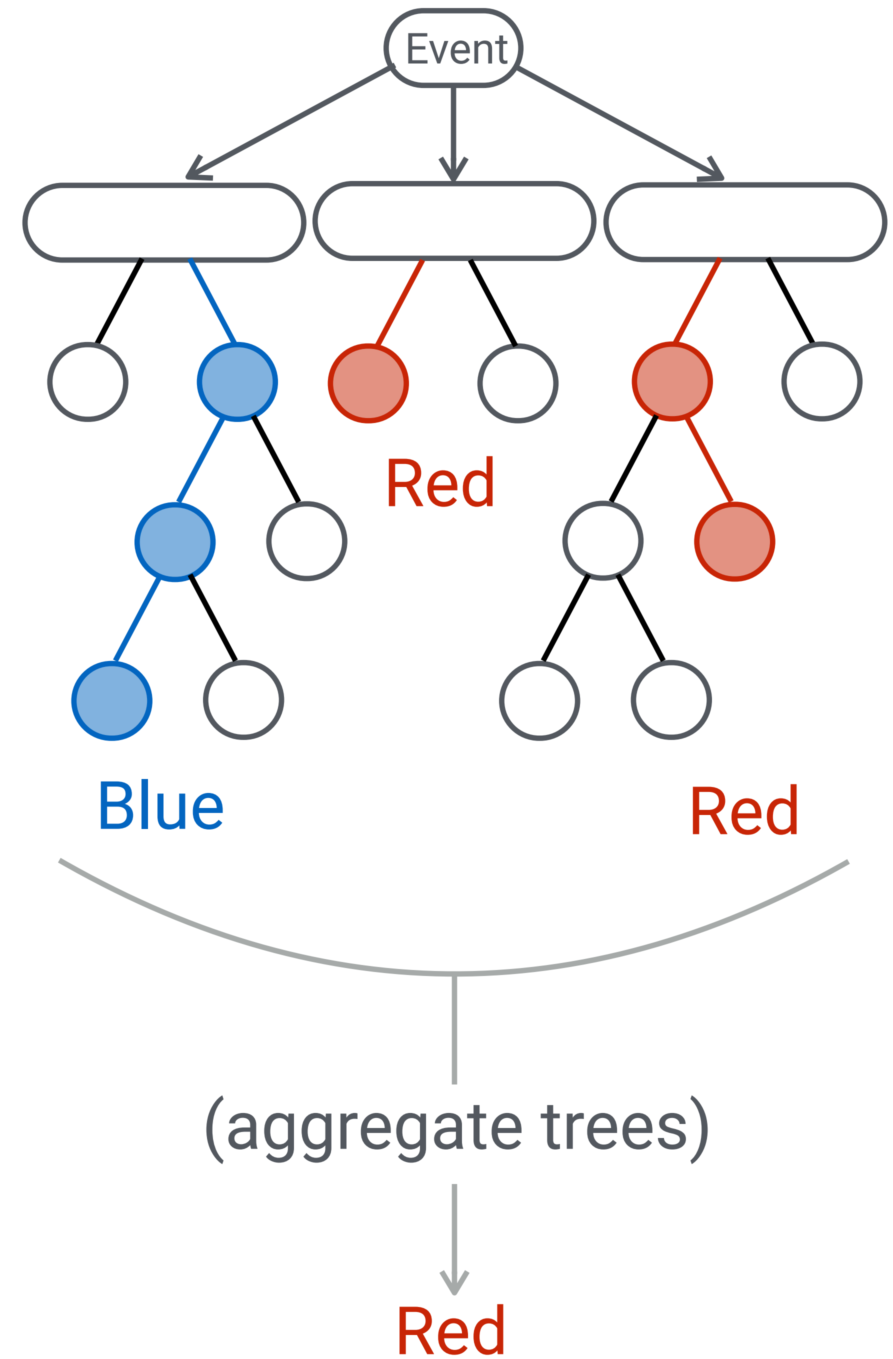
1. Draw n_{trees} bootstrap samples from training data
2. For each bootstrap sample, grow a decision tree.
   * At each node, find best split among random subset of $n_{features}$ of the training features
3. Classify new samples by aggregating the n_{trees} decision trees via a simple majority vote.



Tree-based learning—random forest

Idea: Combine many different decision trees to get better performance.

1. Draw n_{trees} bootstrap samples from training data
2. For each bootstrap sample, grow a decision tree.
   * At each node, find best split among random subset of $n_{features}$ of the training features
3. Classify new samples by aggregating the n_{trees} decision trees via a simple majority vote.



Model evaluation

Under vs. over-fitting

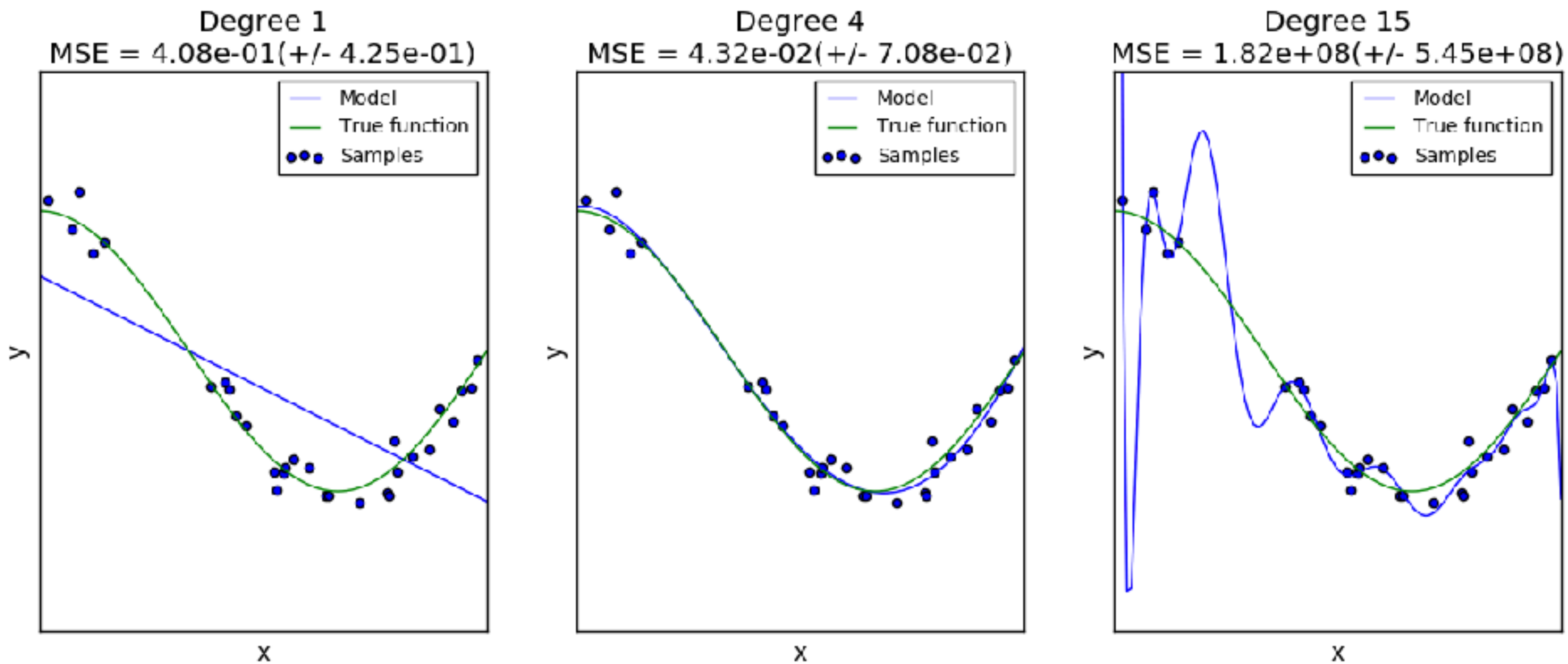


Image credit: http://scikit-learn.org/stable/modules/learning_curve.html

Cross validation (CV)

- Partition training set into different subsets (called ‘folds’)
- Use one of the folds for testing and the rest for training
- Cycle through all folds
- Average performance from each CV fold
- Useful for evaluating model generalization error, hyper parameter tuning, etc.

Image credit: Sebastian Raschka, *Python Machine Learning*

Example

GitHub repository:
https://github.com/jrbourbeau/xmeeting-intro-machine-learning

Questions?