CLASSIFICATION

Classification

- Definition: assign an object to one of several predefined categories

- Given:
  - A set of predefined classes
  - A number of attributes
  - A learning set
- Goal:
  - Predict the class of unclassified data

Applications

- A medical researcher wants to analyze patients' data to determine who is at risk of heart disease:
  - Categories: at-risk, not-at-risk
  - Data set: (Age, heart rate, blood pressing, smoking, heart disease in family, <u>class</u>)
- A company would like to analyze customer data to predict which customers would likely leave
- A scientist would like to classify trees by looking at the leaves it produces

General Approach

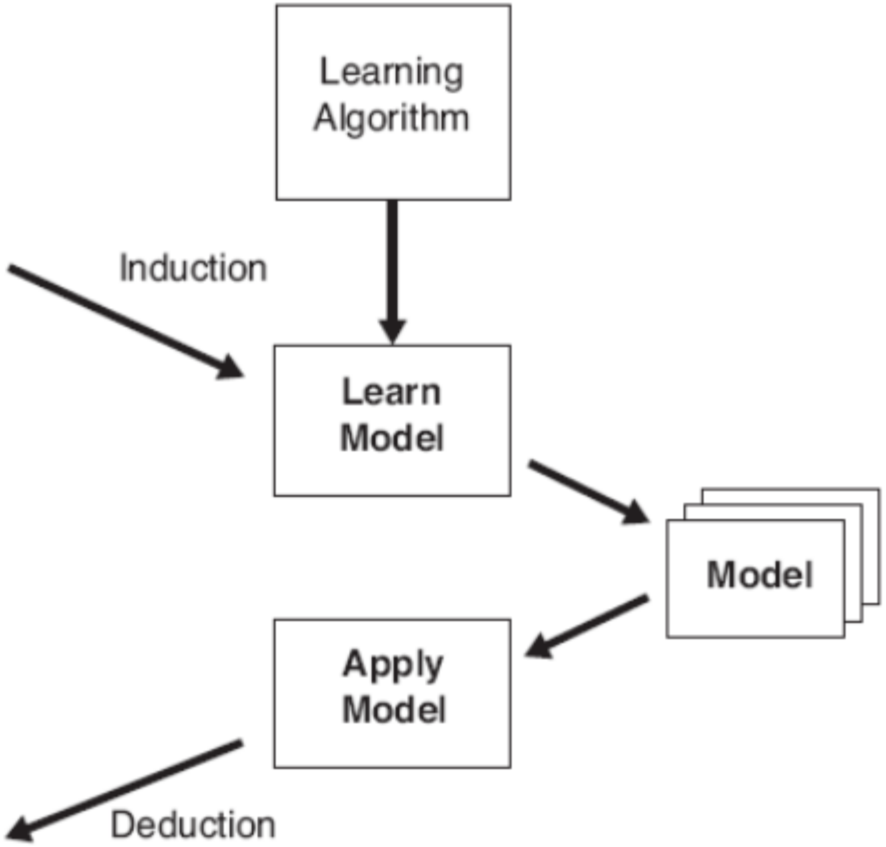
2 Step process

1. Learning step: uses the training data to build a classification model
2. Classification step: uses the model from step 1 to predict the class of test data and estimate accuracy of the model

- **Supervised learning** since the class of the training data is given

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes



Test Set

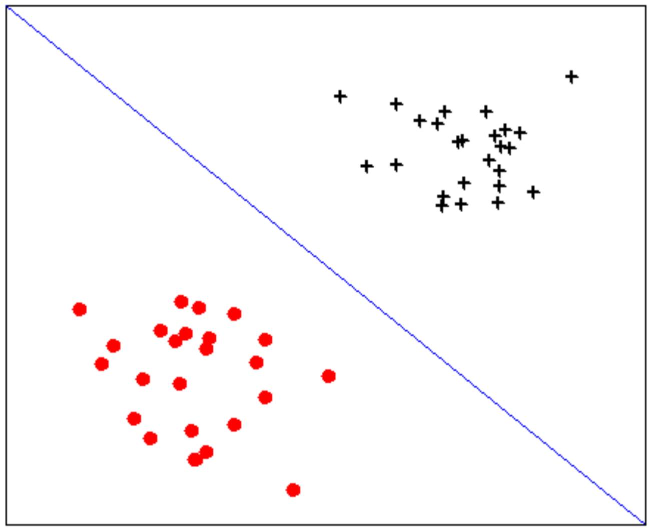
Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Methods

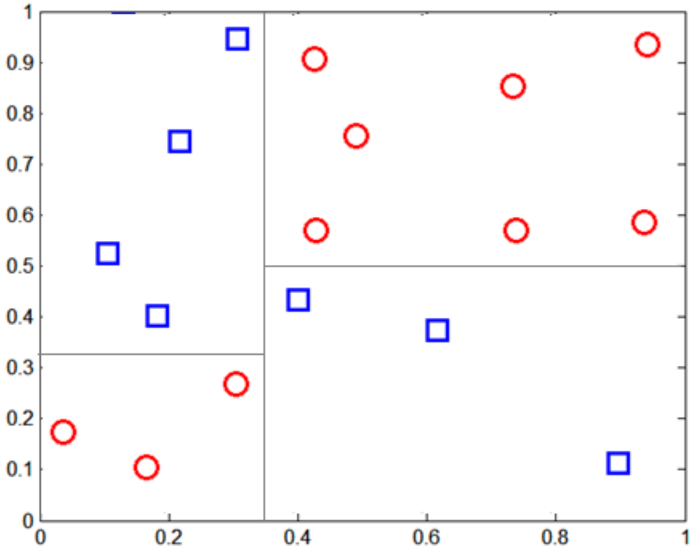
- Decision Trees
- Classification Rules
- Naïve Bayes, Bayes Networks
- Neural Networks
- Nearest Neighbor
- Ensemble Methods

Decision Boundaries

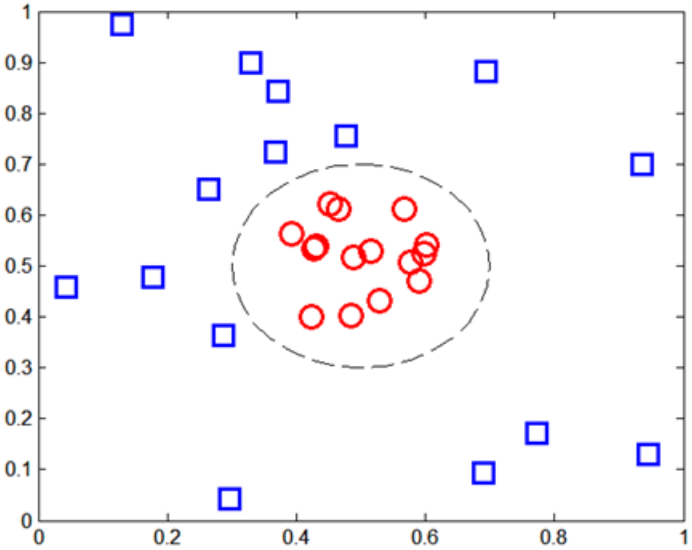
Border line between two neighboring regions of different classes is known as decision boundary



Linear



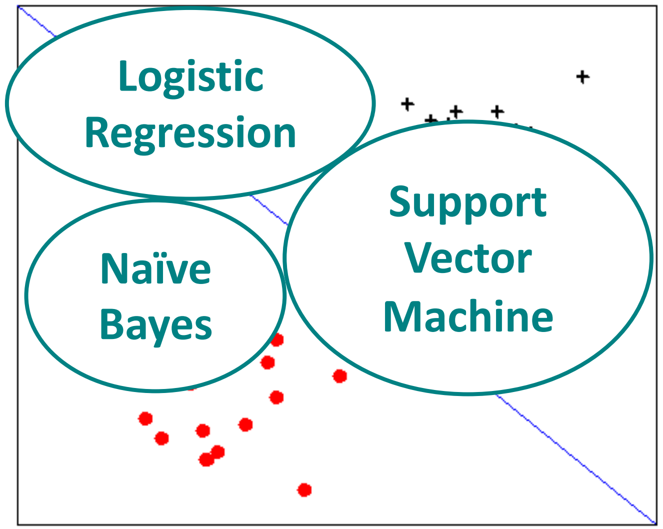
Rectilinear



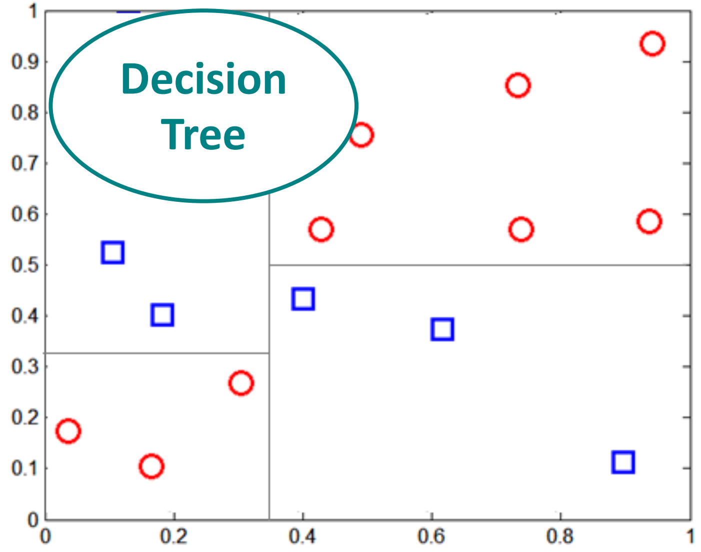
Nonlinear

Decision Boundaries

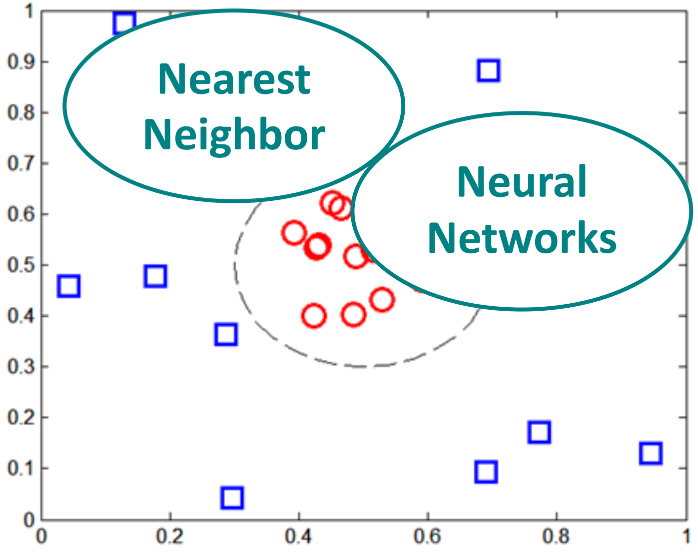
Border line between two neighboring regions of different classes is known as decision boundary



Linear



Rectilinear

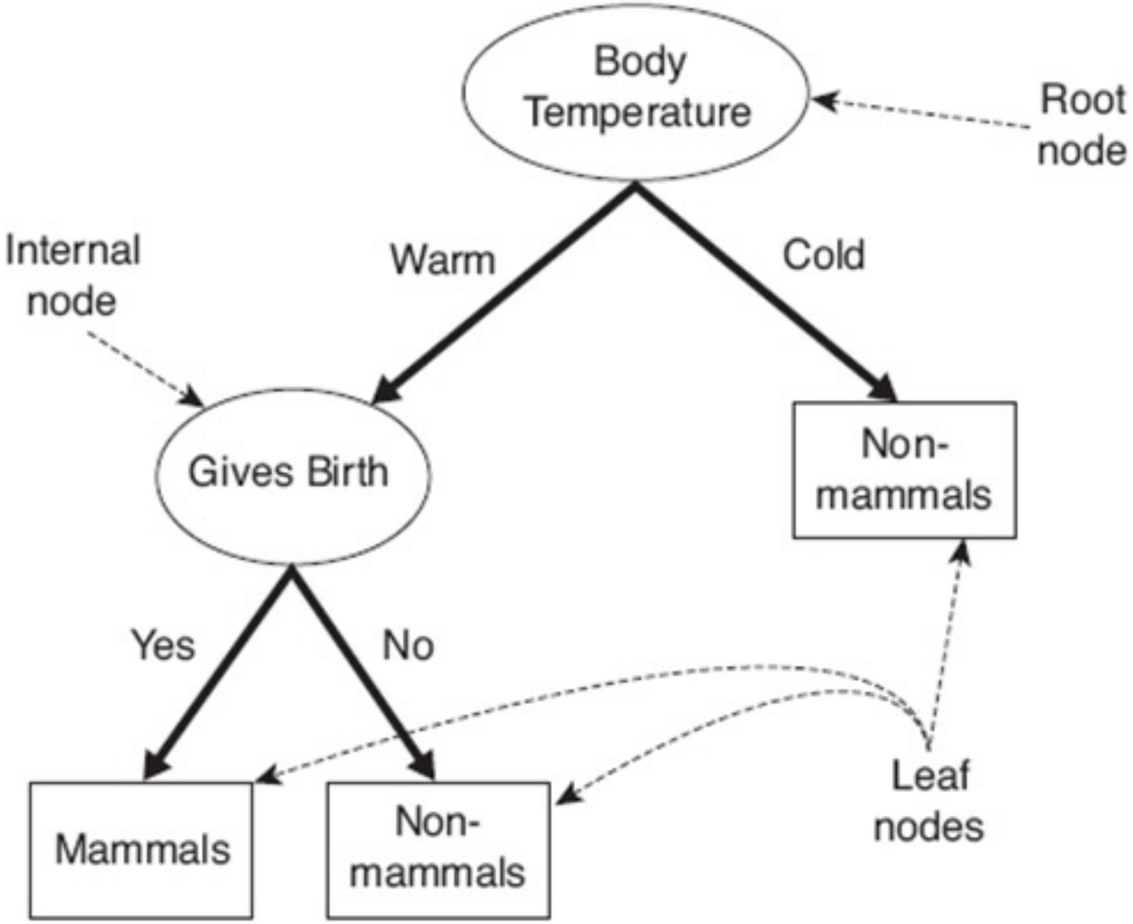


Nonlinear

Decision Trees

- Performs well across wide range of situations
- Requires little effort
- Readily understandable by consumer

- CART (classification and regression trees) originally proposed



Decision Tree Rules

classify records by using a collection of “if…then…” rules.

Rule-Based Ordering

(Skin Cover=feathers, Aerial Creature=yes)
==> Birds

(Body temperature=warm-blooded,
Gives Birth=yes) ==> Mammals

(Body temperature=warm-blooded,
Gives Birth=no) ==> Birds

(Aquatic Creature=semi)) ==> Amphibians

(Skin Cover=scales, Aquatic Creature=no)
==> Reptiles

(Skin Cover=scales, Aquatic Creature=yes)
==> Fishes

(Skin Cover=none) ==> Amphibians

Class-Based Ordering

(Skin Cover=feathers, Aerial Creature=yes)
==> Birds

(Body temperature=warm-blooded,
Gives Birth=no) ==> Birds

(Body temperature=warm-blooded,
Gives Birth=yes) ==> Mammals

(Aquatic Creature=semi)) ==> Amphibians

(Skin Cover=none) ==> Amphibians

(Skin Cover=scales, Aquatic Creature=no)
==> Reptiles

(Skin Cover=scales, Aquatic Creature=yes)
==> Fishes

Rules that belong to the same class appear together.

Example: Riding lawn mower owners

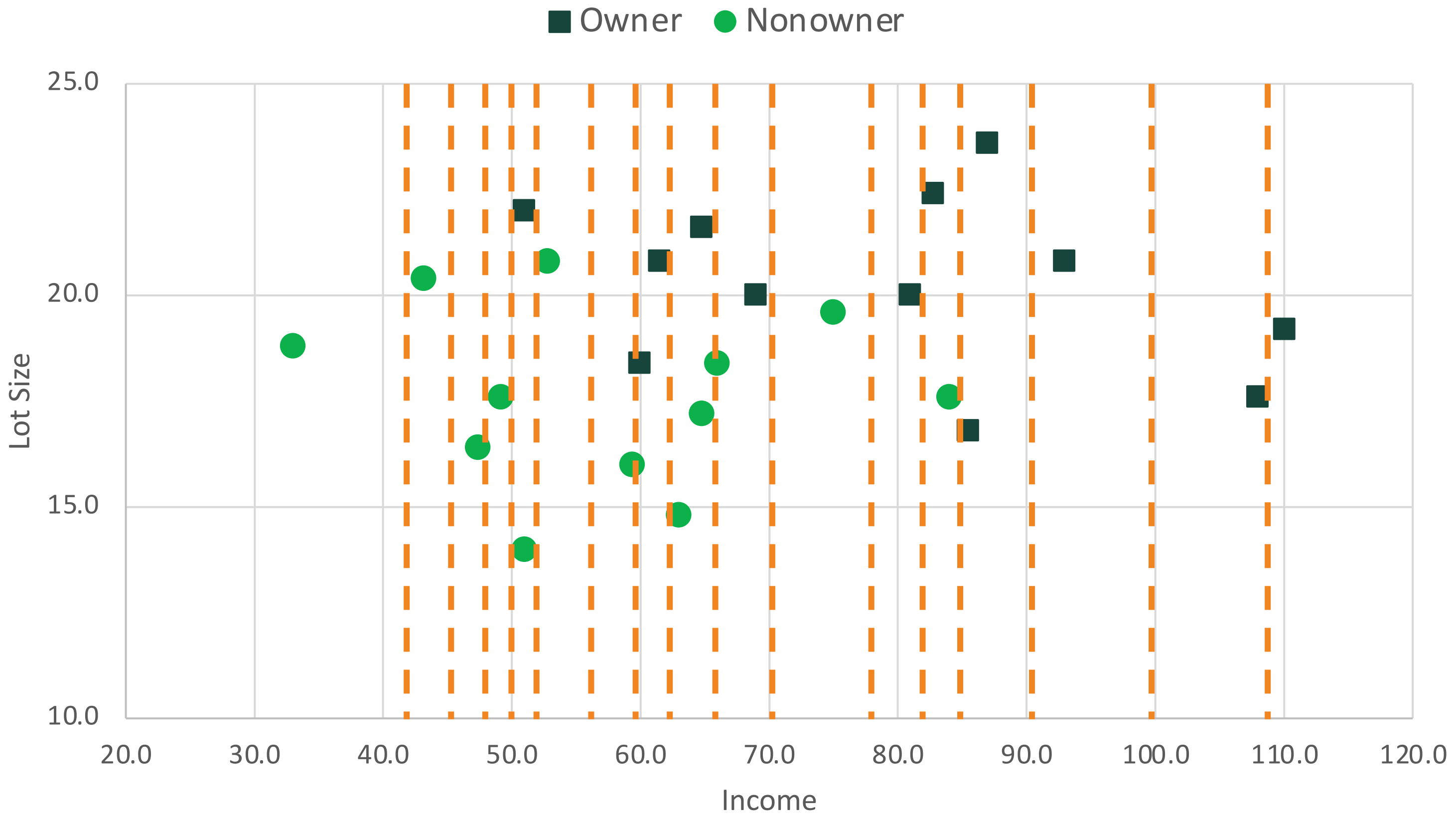
Owners

Income (in thousands)	Thousands Sq. Feet
60.0	18.4
85.5	16.8
64.8	21.6
61.5	20.8
87.0	23.6
110.1	19.2
108.0	17.6
82.8	22.4
69.0	20.0
93.0	20.8
51.0	22.0
81.0	20.0

Nonowners

Income (in thousands)	Thousands Sq. Feet
75.0	19.6
52.8	20.8
64.8	17.2
43.2	20.4
84.0	17.6
49.2	17.6
59.4	16.0
66.0	18.4
47.4	16.4
33.0	18.8
51.0	14.0
63.0	14.8

Riding lawn mower owners
Where is first split?



Decision Tree Construction

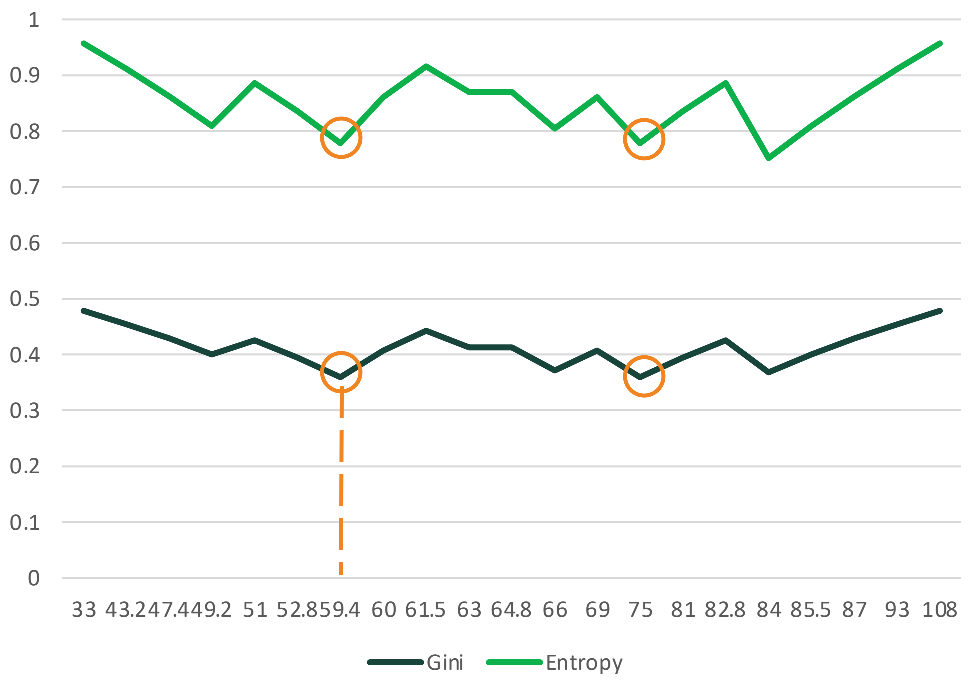
- Recursive partitioning

- Determine split by measure of impurity
  - Entropy: $-\sum_k p_k \log p_k$ Range: [0, 1]
  - Gini index: $1 - \sum_{k=1}^{m} p_k^2$ Range: [0, (m-1)/m]
    - p_k is proportion of observations from class k
    - m is the number of classes
  - What is the range of Gini index for 2 classes? for 10 classes?

- Less impurity => better split

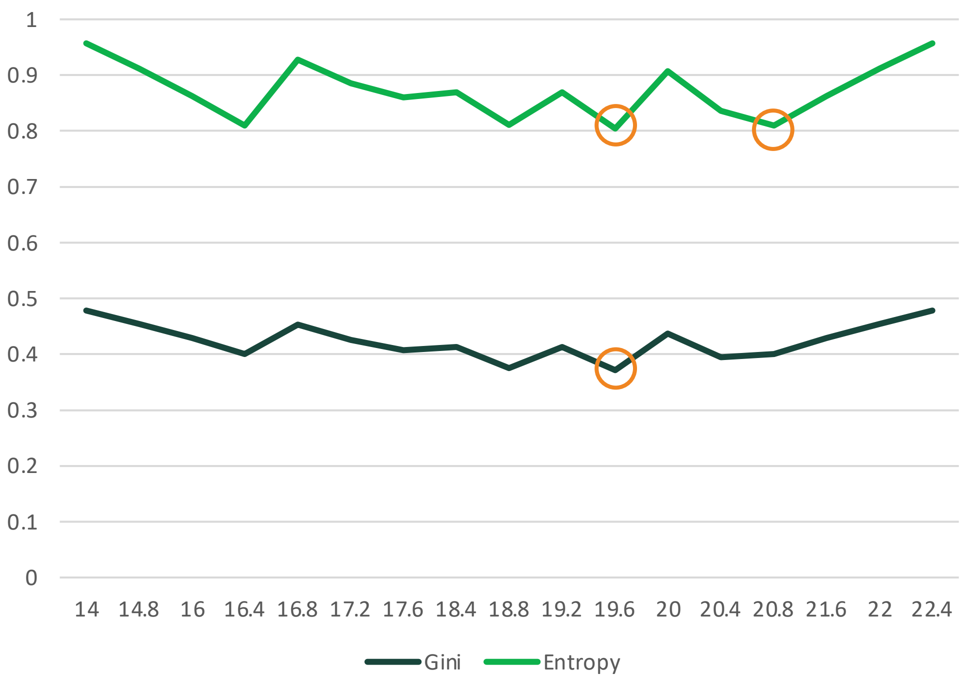
Gini Index vs. Entropy

Income



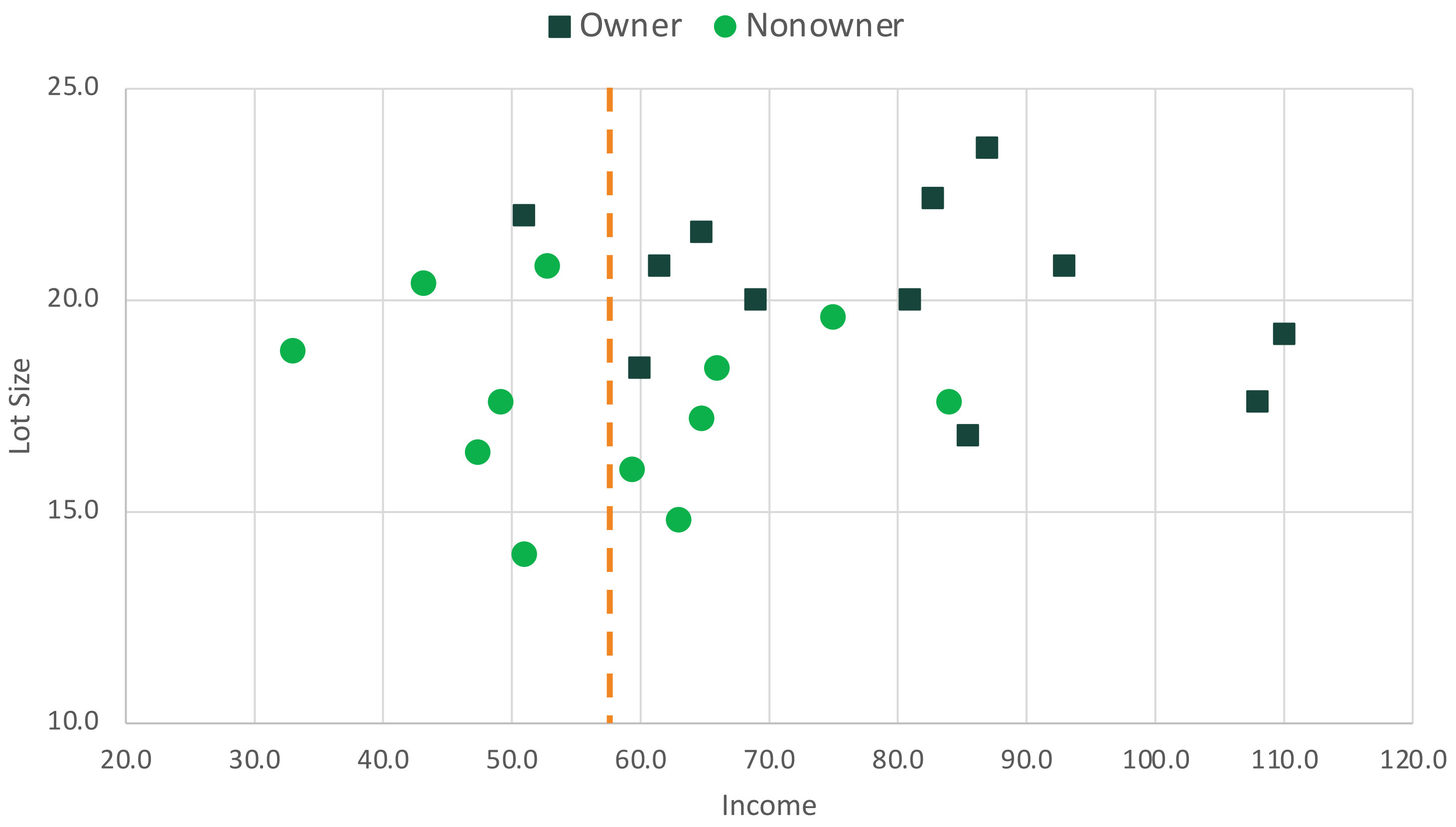
Min Gini: 0.359, Min Entropy: 0.779

Lot Size

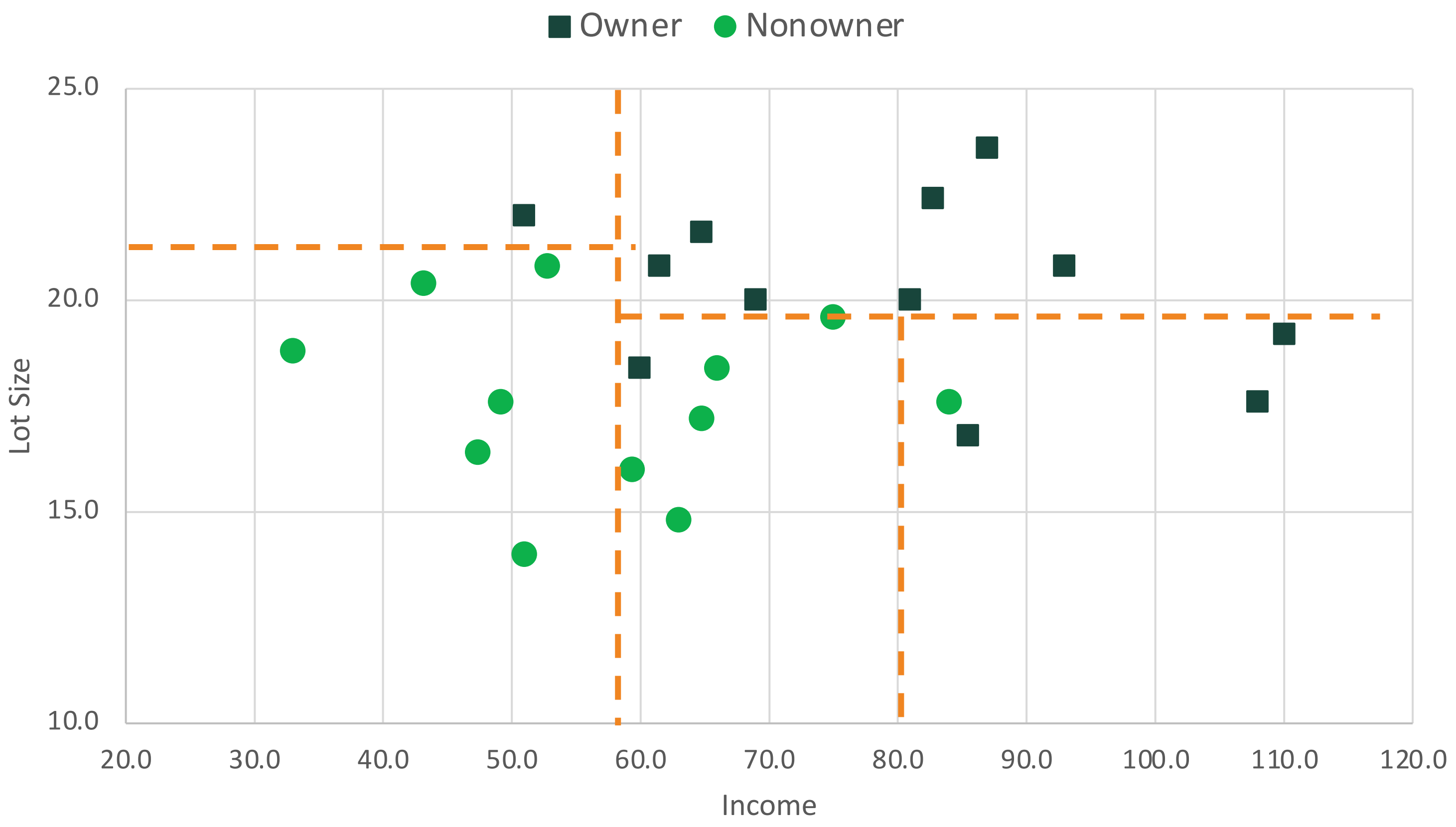


Min Gini: 0.371, Min Entropy: 0.804

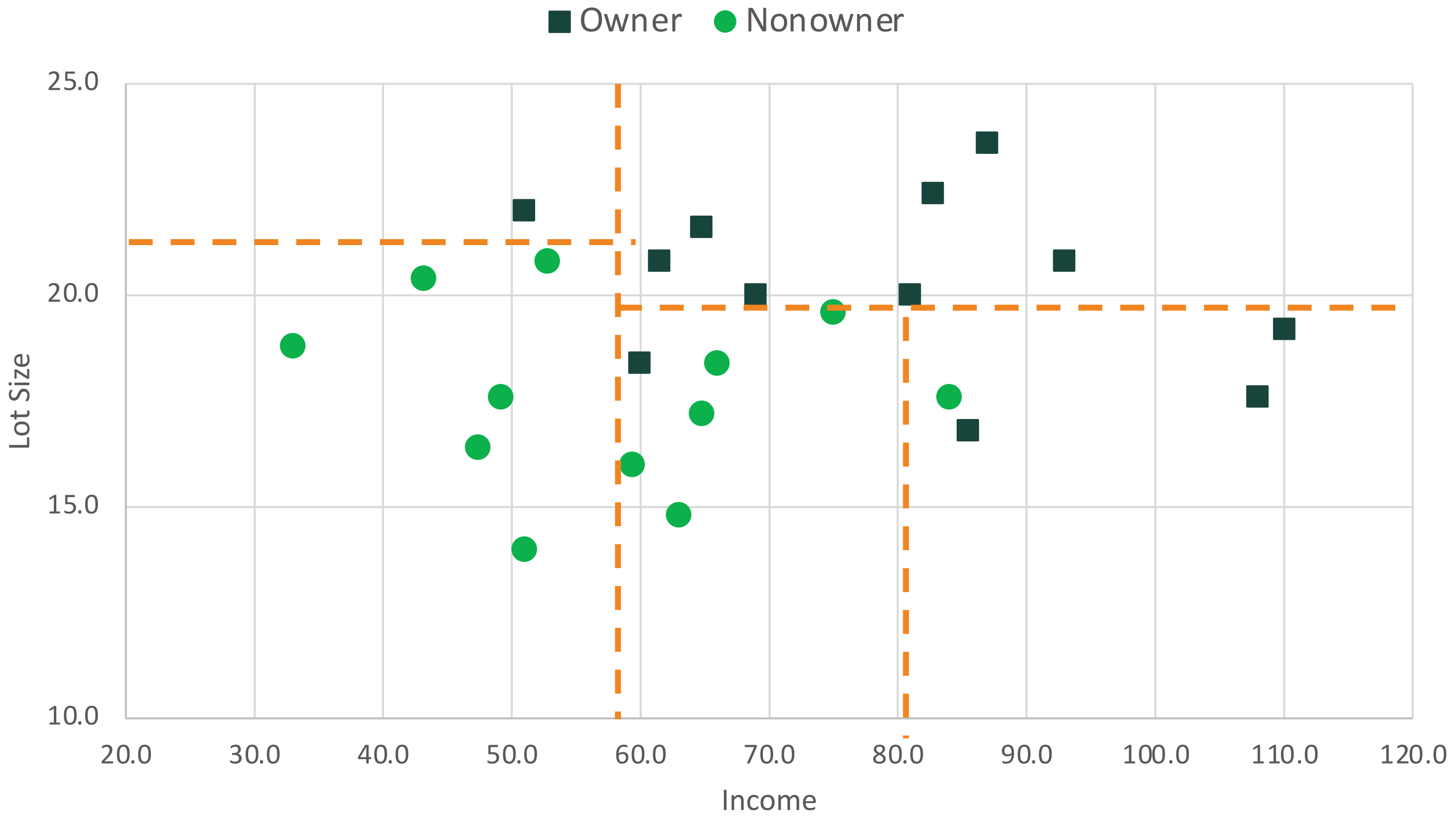
Riding lawn mower owners
Where is first split?



Riding lawn mower owners
Where is second splits, etc..?



Riding lawn mower owners



New household ($60,000 income, 20,000 sq. feet)? Owner

Decision Trees

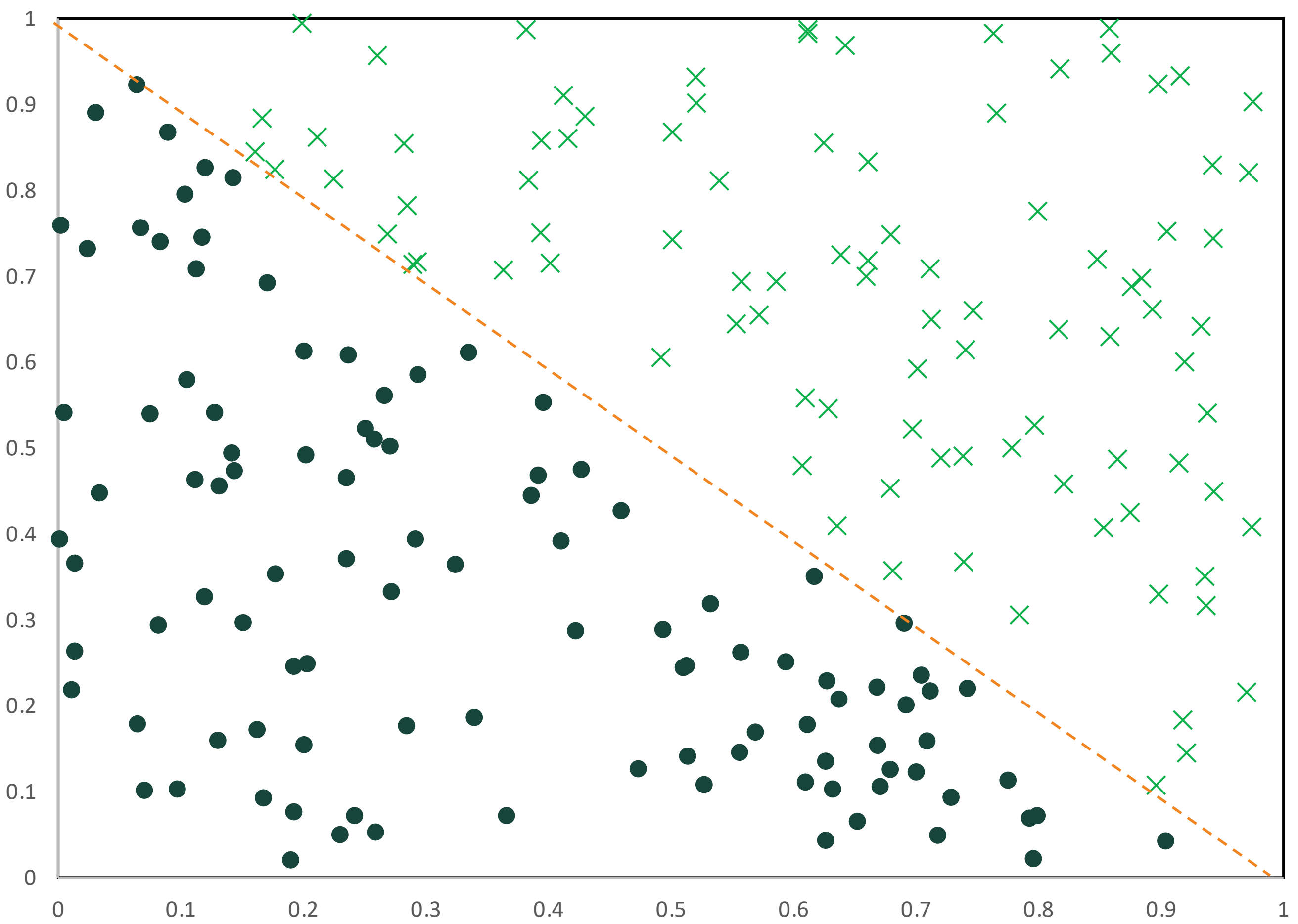
Advantages

- Nonparametric
  - No prior assumptions about probability distribution
- Inexpensive to train a model and classify a test record
- Easy to interpret
- Robust to noise, redundant attributes

Disadvantages

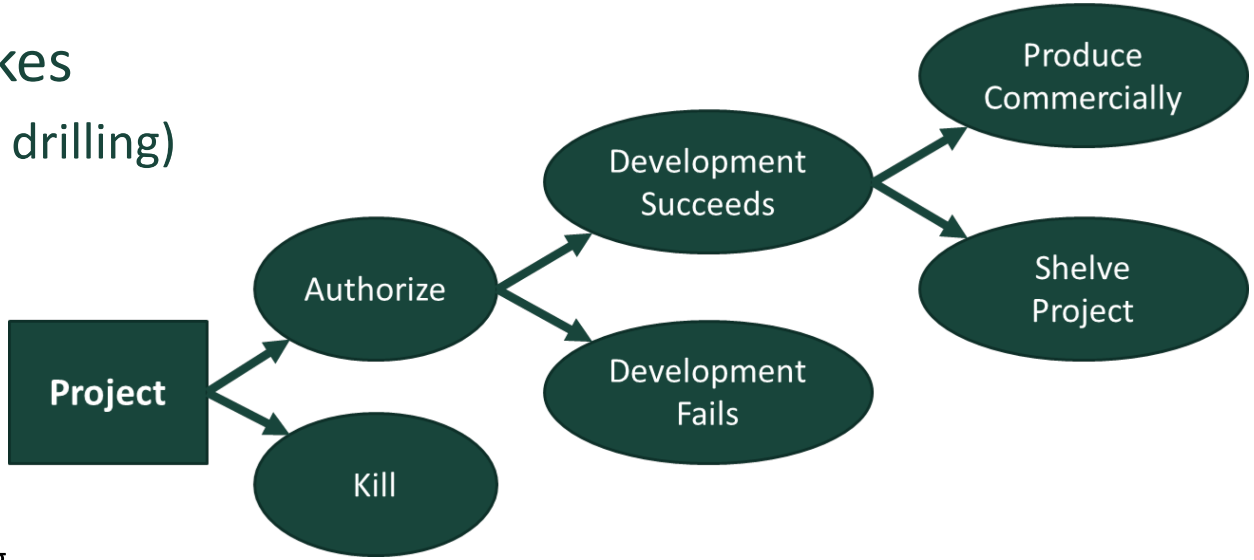
- Finding optimal decision tree is NP-complete
- Do not generalize well to certain Boolean problems
- Number of records at leaf nodes may be too small for statistical significance
- Limited expressiveness for relationship between continuous attributes

Disadvantage: Continuous attributes



Applications of Decision Trees

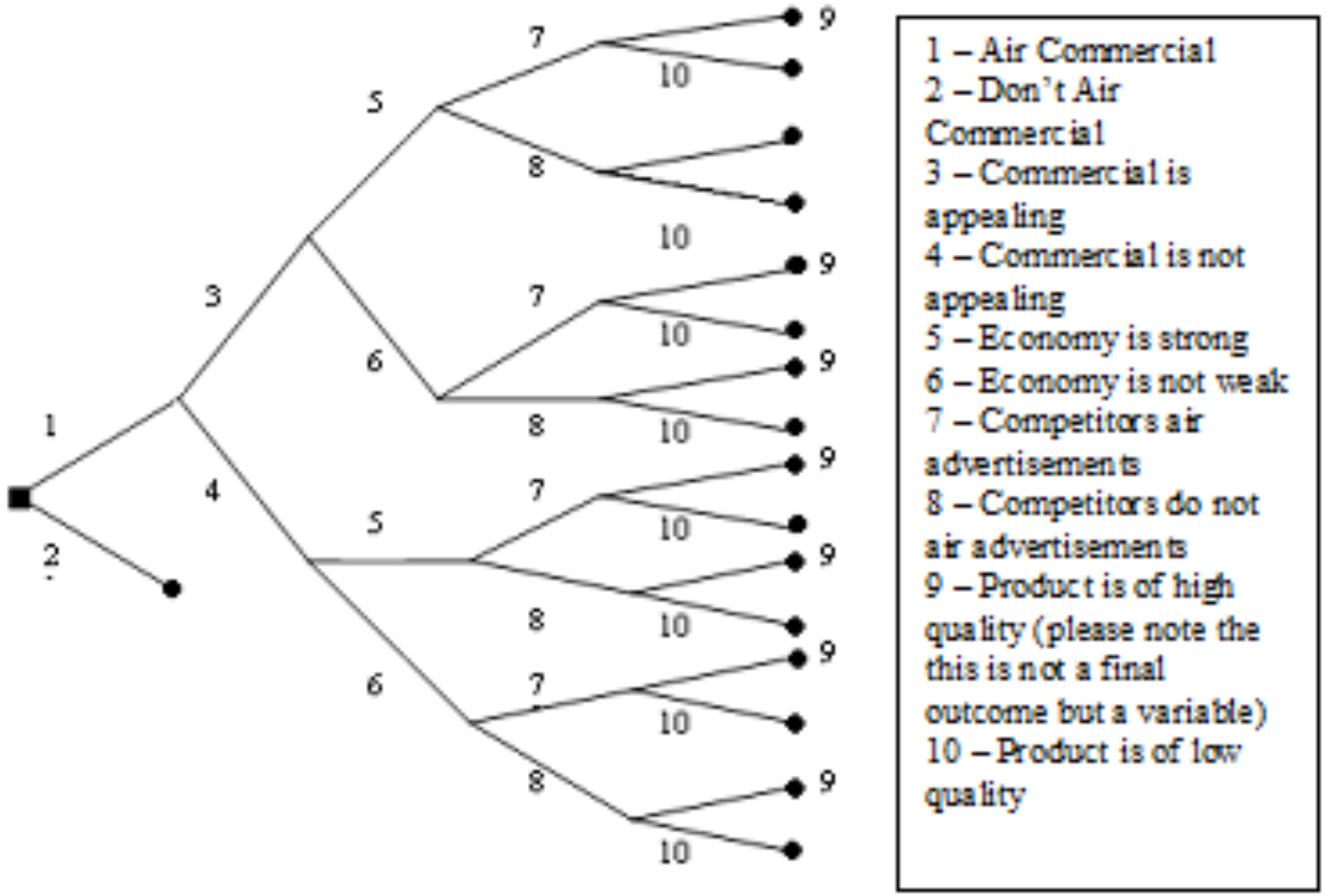
- Binomial Option Pricing
  - Value of an option (asset) at expiration
- Real Option Analysis
  - Decision a company makes
    - Expand (purchase land for drilling)
    - Abandon project
- Competing Projects
  - Expand product (hire additional workers)
  - Marketing operations



https://hbr.org/1964/07/decision-trees-for-decision-making
https://www.investopedia.com/articles/financial-theory/11/decisions-trees-finance.asp

To air or not to air at the Super Bowl

- Is the commercial appealing?
- How is the economy?
- How does our advertisement compare to competitors'?
- What is the quality of our product?



https://www.investopedia.com/articles/financial-theory/11/decisions-trees-finance.asp

Implementation - Data

- X – data matrix (e.g. data.data)
  - Capital X to denote this is a matrix, there are multiple features / attributes
- y – outcome / label (e.g. data.target)
  - Lowercase y denotes this is a vector. For each item, predict 1 value
  - Possible to have multiple outcomes that may be related (e.g. weight and BMI)
    - Desire to build single model to predict both outcomes
    - Outside the scope of this course

Implementation - Model

1. Import the class

   ```
   from sklearn import tree
   ```

2. Define the model

   ```
   clf = tree.DecisionTreeClassifier()
   ```

   Currently uses default parameters

   How to change the parameter?

3. Train the model

   ```
   clf = clf.fit(X,y)
   ```

4. Make predictions with trained model
   1. Predict class (highest probability)

      ```
      y_predict = clf.predict(X_test)
      ```

   2. Or, predict probability of each class

      ```
      y_prob = clf.predict_proba(X_test)
      ```

class `sklearn.tree.DecisionTreeClassifier`(*criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, class_weight=None, presort=False*)
[source]

clf = tree.DecisionTreeClassifier()

A decision tree classifier.

Read more in the User Guide.

clf = tree.DecisionTreeClassifier(criterion = 'entropy')

Parameters:

criterion : string, optional (default="gini")

The function to measure the quality of a split. Supported criteria are "gini" for the Gini impurity and "entropy" for the information gain.

splitter : string, optional (default="best")

The strategy used to choose the split at each node. Supported strategies are "best" to choose the best split and "random" to choose the best random split.

max_depth : int or None, optional (default=None)

The maximum depth of the tree. If None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

min_samples_split : int, float, optional (default=2)

The minimum number of samples required to split an internal node:

- If int, then consider min_samples_split as the minimum number.
- If float, then min_samples_split is a percentage and ceil(min_samples_split * n_samples) are the minimum number of samples for each split.

Example:
clf = tree.DecisionTreeClassifier(
criterion = 'entropy', max_depth = 10)

Decision Tree Tuning Variables

- Variables tune in this course
  - criterion – function to evaluate split quality
  - max_depth – height of tree
  - min_samples_split – number of samples at a node before split
  - min_samples_leaf – number of samples at a leaf
  - min_weight_fraction_leaf – how pure is a leaf (0.0 means all items are same class)

- Special case variables:
  - random_state
    - If comparing multiple classifiers, set random state
    - Removes random element of training
    - Always generate same model for given random state
  - class_weight
    - If 1 class is significantly less frequent among items (1/10 or less)

http://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

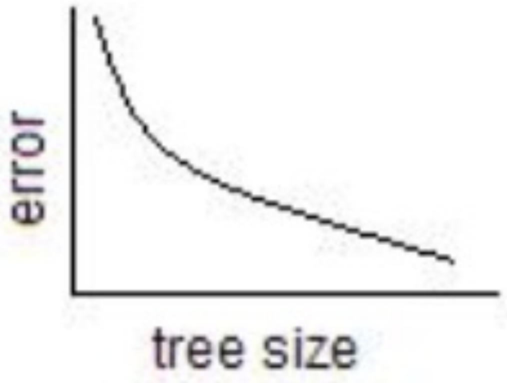
Performance evaluation

- Training errors: number of misclassified records in the training set
- Generalization errors: the expected error of the model on previously unseen records

- Goal: reduce both training errors AND generalization errors

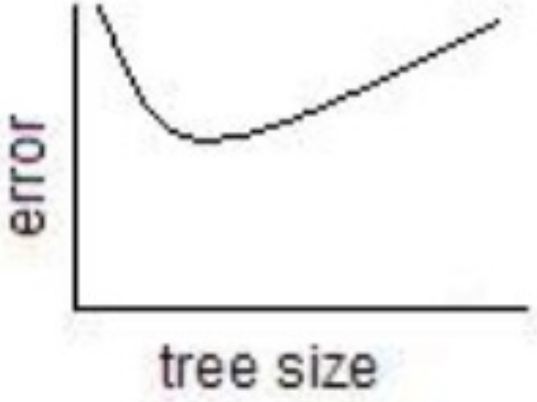
Decision Tree Example

Training errors



tree size vs. training error

Generalization errors



tree size vs. testing error

Performance Evaluation

- Confusion Matrix:

		Predicted Class	
		Class = 1	Class = 0
Actual Class	Class = 1	f_{11}	f_{10}
	Class = 0	f_{01}	f_{00}

- Accuracy: fraction of correct predictions

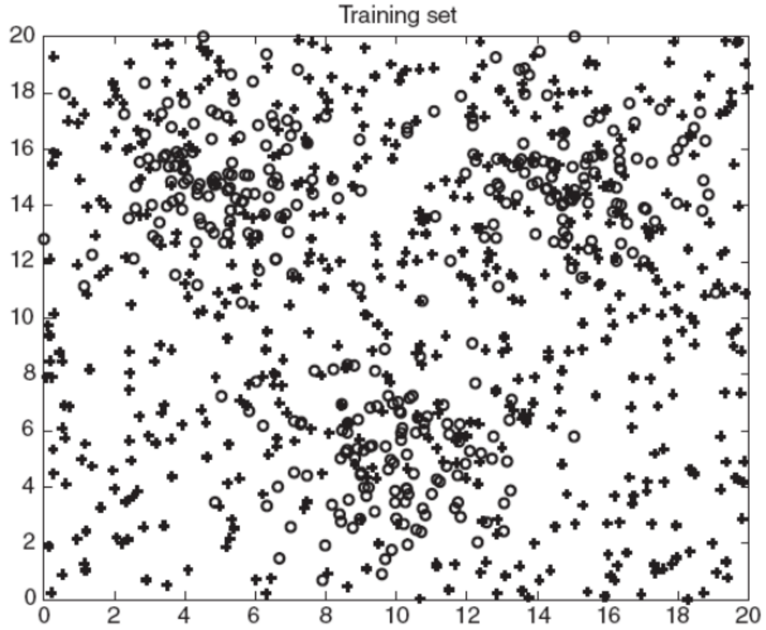
$$accuracy = \frac{f_{11} + f_{00}}{f_{11} + f_{10} + f_{01} + f_{00}}$$

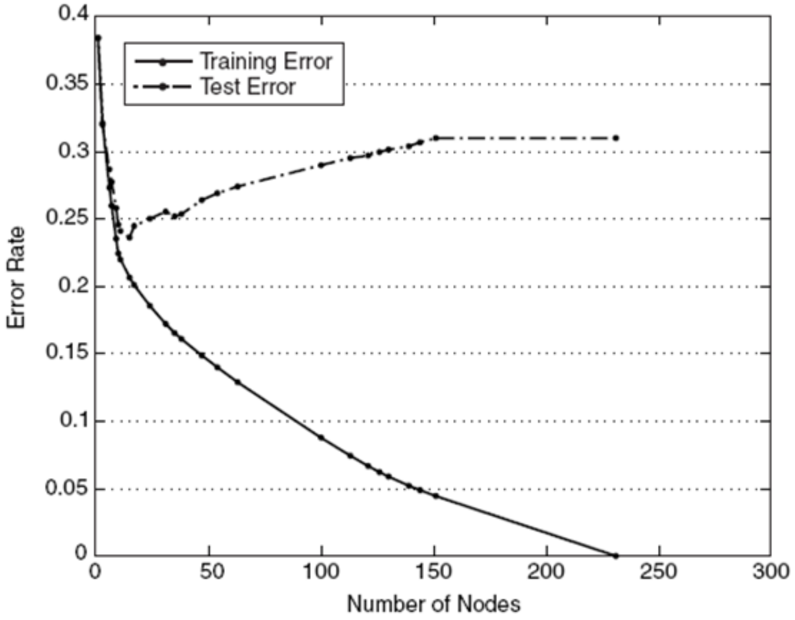
- Error rate: fraction of wrong predictions

$$error\_rate = \frac{f_{01} + f_{10}}{f_{11} + f_{10} + f_{01} + f_{00}}$$

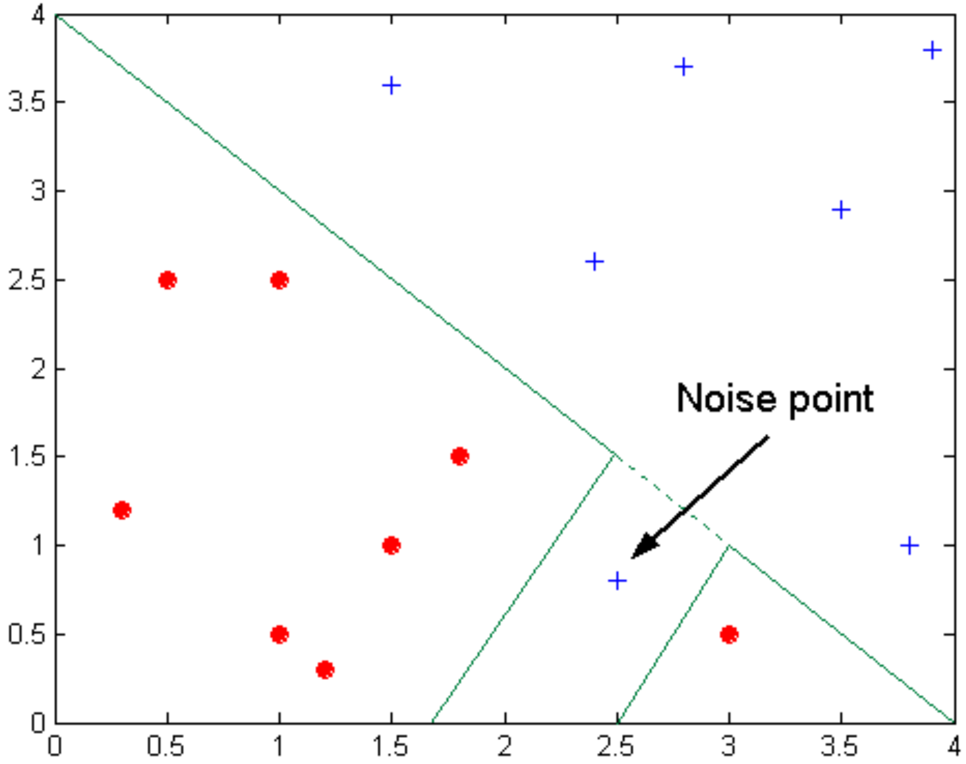
Underfitting and Overfitting

- Underfitting: when the model is too simple
- Overfitting: when the model is built to tightly fit the training set



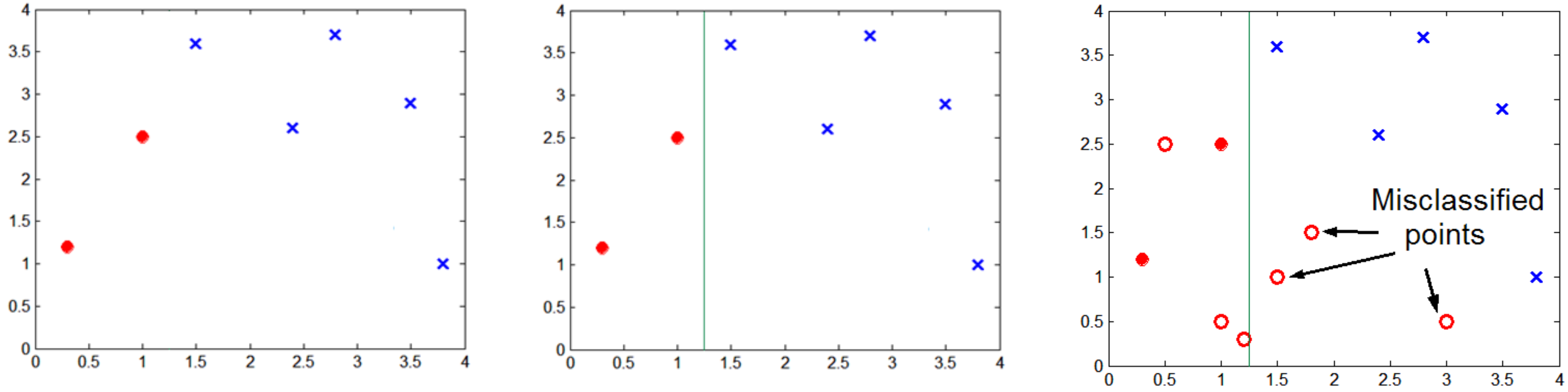


Overfitting due to Noise



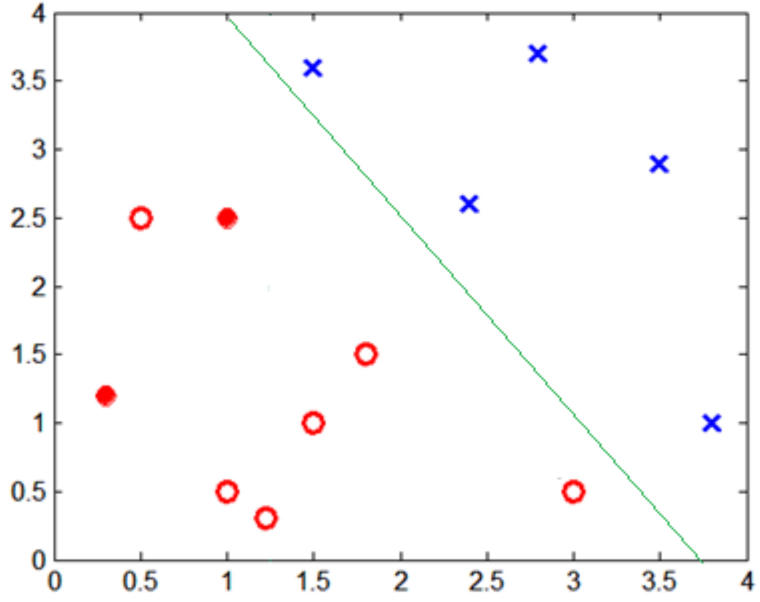
Decision boundary is distorted by noise point

Overfitting due to Lack of Representative Samples



Lack of data points in the lower half of the diagram makes it difficult to predict correctly the class labels of that region

Overfitting due to Lack of Representative Samples



Lack of data points in the lower half of the diagram makes it difficult to predict correctly the class labels of that region

Other Performance Measures

- **Speed**: computational cost involved in generating and using the model
- **Robustness**: ability to make correct prediction using noisy/missing values
- **Scalability**: Ability to construct the classifier given large amounts of data
- **Interpretability**: level of insight provided by the classifier

Handling Overfitting

- Early Stopping Rule
  - Stop expanding when observed gain below a threshold

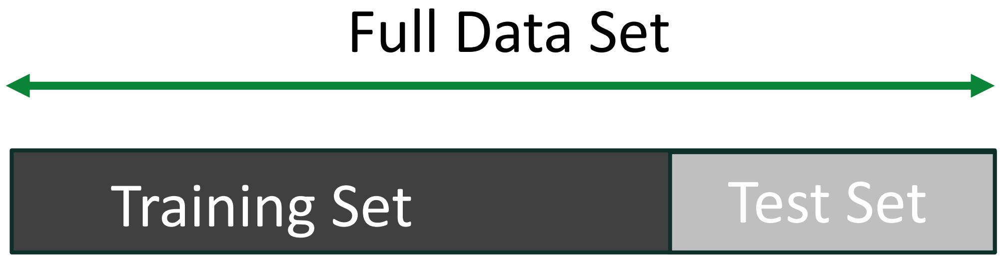
- Post-pruning
  - **Subtree Replacement**: Replace subtree with leaf node
  - **Subtree Raising**: Replace subtree with most frequently used branch

Evaluating the Performance of a Classifier

- Holdout method
- Random sampling
- Cross-Validation
- Bootstrapping

Holdout Method

- Given the training data, split into two disjoint sets: the training set and the test set

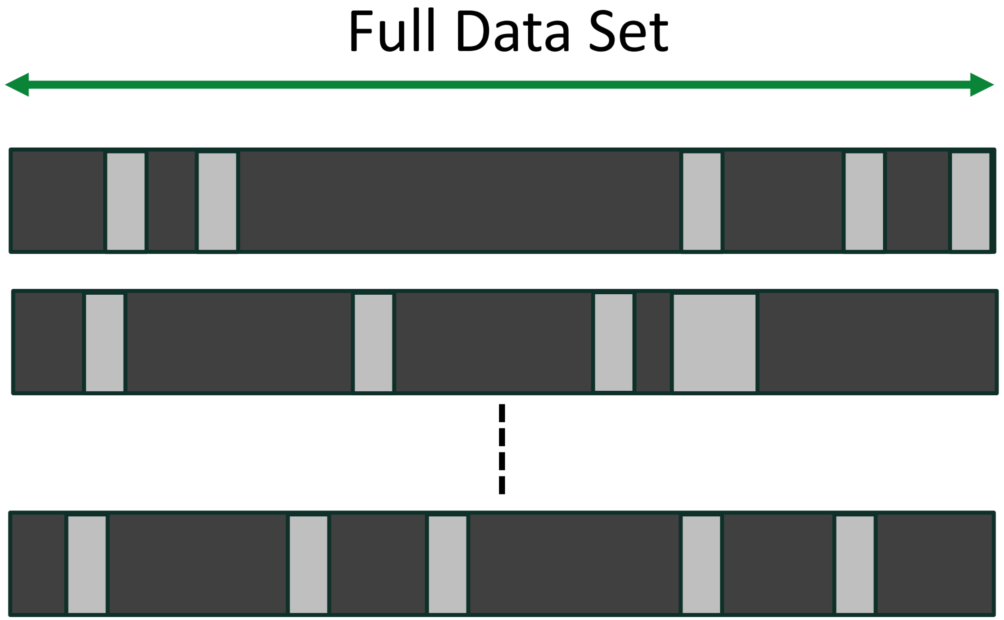


- Limitations:
  - Fewer data available for training
  - The model is highly dependent on the composition of the two sets
  - The training set and test not independent:
    - An overrepresented class in one set will be underrepresented in the other set

Random Subsampling

- Repeat the holdout method multiple times to improve the estimation
- Accuracy:

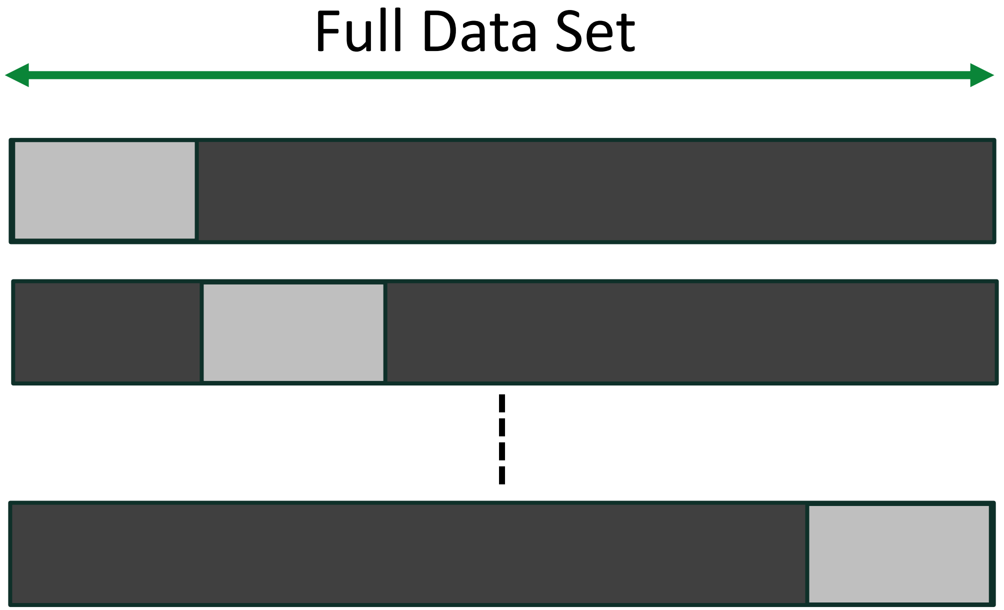
$$acc_{sub} = \sum_i acc_i \Big/ k$$



- Limitation:
  - Not using as much data as possible for training
  - Some records used multiple times for training

K-Fold Cross Validation

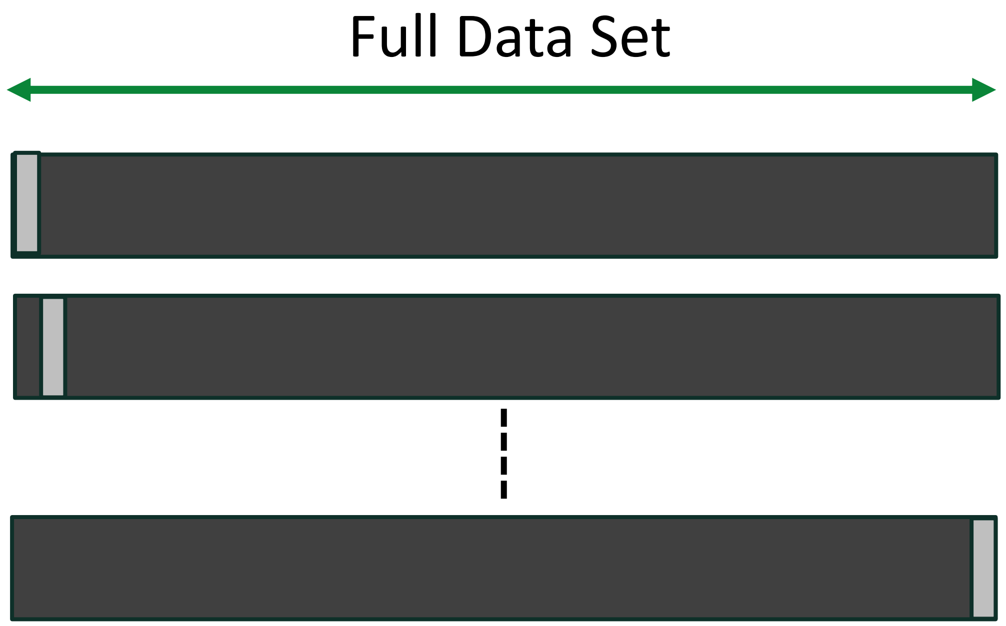
- Splits the data into k disjoint sets
- In each iteration, one set is used for testing and K-1 for training



- Advantage: all records are used for both training and test

Leave one out – Cross validation

- Special case of cross validation where k = N
- Uses as much data as possible for training
- Test sets mutually exclusive
- Computationally expensive



Bootstrap

- Training records are sampled with replacement
- Training data may contain duplicate records
- Repeat process b times to generate b bootstrap samples
- When N records are chosen from a N record set, the probability of a record being selected is ~ 0.632

Never being selected: $(1-1/N)^N \sim e^{-1} = 0.368$ if N is sufficiently large

- Accuracy:

$$acc_{boot} = \frac{1}{b}\sum_{i=1}^{b}(0.632 \times \varepsilon_i + 0.368 \times acc_s)$$

acc_s: accuracy of original sample as training set

ε_i: accuracy of bootstrap sample