

# Introduction to Machine Learning

Andriy Mulyar<sup>1</sup>

<sup>1</sup>Department of Computer Science  
Virginia Commonwealth University  
Richmond, VA  
USA

[aymulyar@vcu.edu](mailto:aymulyar@vcu.edu)



March 5 2019

# Outline

1 Introduction

2 Preliminaries

3 Learning

4 Model Evaluation

5 Decision Trees

# Outline

1 Introduction

2 Preliminaries

3 Learning

4 Model Evaluation

5 Decision Trees

# Our Reality: We Are Drowning In Data

- Every click, every network packet and every online interaction is recorded and will outlive us.

## 2018 *This Is What Happens In An Internet Minute*



# Data Science

- The umbrella field of data science aims to harness this massive amount of data to glean insights, drive industry and improve the human experience.

## Scientific Literature Search - Finding The Needle in the Haystack

**4** scientific papers published every minute



**160**

million documents indexed  
on Google Scholar



**24**

million documents listed on PubMed



**12**

million documents listed on Science Direct

# Societal Ramifications and Applications

- Health care (drug re-purposing, RNA sequencing)
- Automated knowledge discovery (scientific literature mining)
- Ease of mass surveillance (social credit systems, geographic tracking)
- Targeted advertising (political elections, big nudging)



# Pattern Recognition and Machine Learning

- This talk focuses on gaining insight into the underlying algorithms and structures that produce models capable of tackling such problems.
- We will explore in-depth a heuristical algorithm - the supervised Decision Tree.

# Outline

1 Introduction

2 Preliminaries

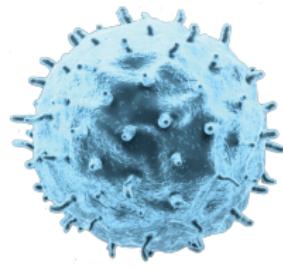
3 Learning

4 Model Evaluation

5 Decision Trees

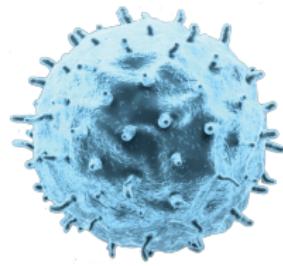
# Entities

- The world around us can be succinctly described as comprised of entities.



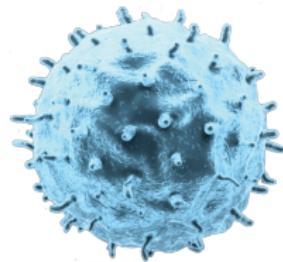
# Entities

- The world around us can be succinctly described as comprised of entities.
- Entities interact (relate) with other entities and in the process possibly form new entities.



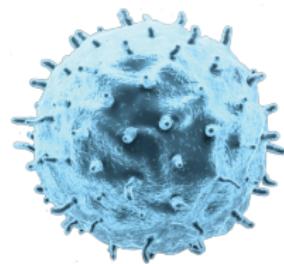
# Entities

- The world around us can be succinctly described as comprised of entities.
- Entities interact (relate) with other entities and in the process possibly form new entities.
- Entities are composed of **attributes**.



# Entities

- The world around us can be succinctly described as comprised of entities.
- Entities interact (relate) with other entities and in the process possibly form new entities.
- Entities are composed of **attributes**.
- An **instance** of an entity is a particular manifestation of the **attributes** that comprise the entity.



# Instances of Entities

- A collection of instances  $x_i \in \mathbb{D}$  from an entity domain  $\mathbb{D}$  forms a sample from the domain.



	account_balance	income	age	has_defaulted	credit_worthy
$x_1$	57103	50000	25	0	?
$x_2$	20343	40000		1	?
$x_3$	80000	150000	40	1	?
$x_4$	500		17	0	?
$x_5$	5000	1000	21	0	?

# Instances of Entities

- A collection of instances  $x_i \in \mathbb{D}$  from an entity domain  $\mathbb{D}$  forms a sample from the domain.
- The **dimension**  $d$  of  $\mathbb{D}$  is the number of attributes describing its entity.



	account_balance	income	age	has_defaulted	credit_worthy
$x_1$	57103	50000	25	0	?
$x_2$	20343	40000		1	?
$x_3$	80000	150000	40	1	?
$x_4$	500		17	0	?
$x_5$	5000	1000	21	0	?

## Instances of Entities

- A collection of instances  $x_i \in \mathbb{D}$  from an entity domain  $\mathbb{D}$  forms a sample from the domain.
- The **dimension**  $d$  of  $\mathbb{D}$  is the number of attributes describing its entity.
- We assume attributes (**observations, features**) of a given instance  $x_{i,j}$  ( $0 \leq j < d$ ) are real numbers.



	account_balance	income	age	has_defaulted	credit_worthy
$x_1$	57103	50000	25	0	?
$x_2$	20343	40000		1	?
$x_3$	80000	150000	40	1	?
$x_4$	500		17	0	?
$x_5$	5000	1000	21	0	?

# Outline

1 Introduction

2 Preliminaries

3 Learning

4 Model Evaluation

5 Decision Trees

# Paradigms

The goal of learning is to utilize a sample  $X \subseteq \mathbb{D}$  to construct a model capable of providing insight into  $\mathbb{D}$ .

# Paradigms

The goal of learning is to utilize a sample  $X \subseteq \mathbb{D}$  to construct a model capable of providing insight into  $\mathbb{D}$ .

Given a **representative** sample of **cardinality**  $n$  from a domain  $\mathbb{D}$  of dimension  $d$ :

$$X = \{x_i \in \mathbb{D}\} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1d} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & x_{n3} & \dots & x_{nd} \end{bmatrix}$$

# Paradigms

The goal of learning is to utilize a sample  $X \subseteq \mathbb{D}$  to construct a model capable of providing insight into  $\mathbb{D}$ .

Given a **representative** sample of **cardinality**  $n$  from a domain  $\mathbb{D}$  of dimension  $d$ :

$$X = \{x_i \in \mathbb{D}\} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1d} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & x_{n3} & \dots & x_{nd} \end{bmatrix}$$

## Supervised Learning

- Denote a subset of attributes  $y$  as target attributes.

# Paradigms

The goal of learning is to utilize a sample  $X \subseteq \mathbb{D}$  to construct a model capable of providing insight into  $\mathbb{D}$ .

Given a **representative** sample of **cardinality**  $n$  from a domain  $\mathbb{D}$  of dimension  $d$ :

$$X = \{x_i \in \mathbb{D}\} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1d} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & x_{n3} & \dots & x_{nd} \end{bmatrix}$$

## Supervised Learning

- Denote a subset of attributes  $y$  as target attributes.
- Use  $X$  alongside known  $y$  to construct a model  $h: \mathbb{D} \setminus y \rightarrow y$ .

# Paradigms

The goal of learning is to utilize a sample  $X \subseteq \mathbb{D}$  to construct a model capable of providing insight into  $\mathbb{D}$ .

Given a **representative** sample of **cardinality**  $n$  from a domain  $\mathbb{D}$  of dimension  $d$ :

$$X = \{x_i \in \mathbb{D}\} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1d} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & x_{n3} & \dots & x_{nd} \end{bmatrix}$$

## Supervised Learning

- Denote a subset of attributes  $y$  as target attributes.
- Use  $X$  alongside known  $y$  to construct a model  $h : \mathbb{D} \setminus y \rightarrow y$ .
- Labeled examples are needed.

## Unsupervised Learning

- Denote  $C = \{C_i \subseteq D\}$  as clusters.

# Paradigms

The goal of learning is to utilize a sample  $X \subseteq \mathbb{D}$  to construct a model capable of providing insight into  $\mathbb{D}$ .

Given a **representative** sample of **cardinality**  $n$  from a domain  $\mathbb{D}$  of dimension  $d$ :

$$X = \{x_i \in \mathbb{D}\} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1d} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & x_{n3} & \dots & x_{nd} \end{bmatrix}$$

## Supervised Learning

- Denote a subset of attributes  $y$  as target attributes.
- Use  $X$  alongside known  $y$  to construct a model  $h : \mathbb{D} \setminus y \rightarrow y$ .
- Labeled examples are needed.

## Unsupervised Learning

- Denote  $C = \{C_i \subseteq D\}$  as clusters.
- Use  $X$  to construct a model  $h : \mathbb{D} \rightarrow C_i$ .

# Paradigms

## Supervised Learning

- Denote a subset of attributes  $y$  as target attributes.
- Use  $X$  alongside known  $y$  to construct a model  $h : \mathbb{D} \setminus y \rightarrow y$ .
- Labeled examples are needed.

## Unsupervised Learning

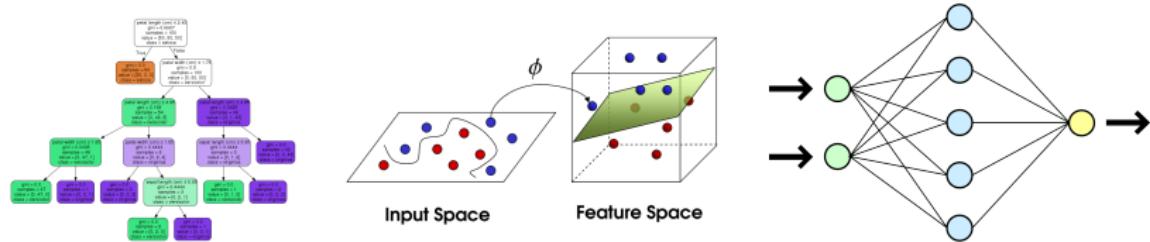
- Denote  $C = \{C_i \subseteq D\}$  as clusters.
- Use  $X$  to construct a model  $h : \mathbb{D} \rightarrow C_i$ .



	account_balance	income	age	has_defaulted	credit_worthy
$x_1$	57103	50000	25	0	?
$x_2$	20343	40000		1	?
$x_3$	80000	150000	40	1	?
$x_4$	500		17	0	?
$x_5$	5000	1000	21	0	?

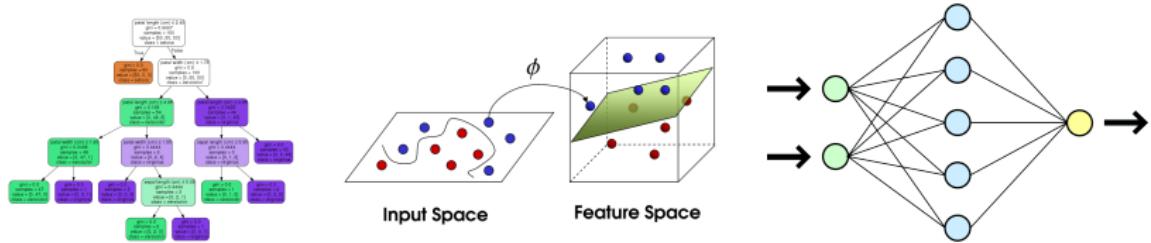
# Supervised Learning

- Machine learning algorithms exploit  $X$  to estimate an  $h$  that has low predictive error.



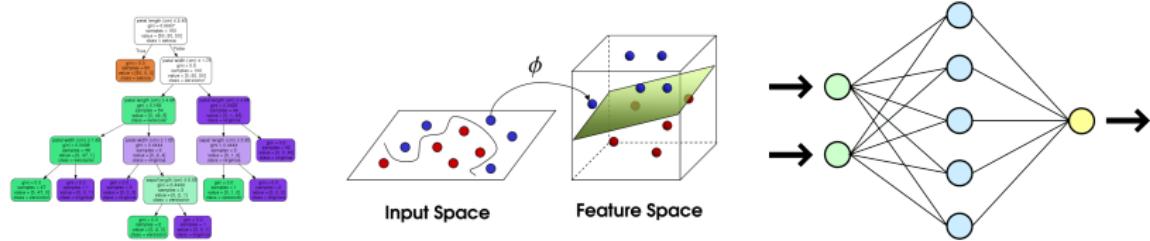
# Supervised Learning

- Machine learning algorithms exploit  $X$  to estimate an  $h$  that has low predictive error.
- In the supervised setting, a machine learning algorithm finds (learns) a mapping from observations  $X$  to outputs  $y$ .



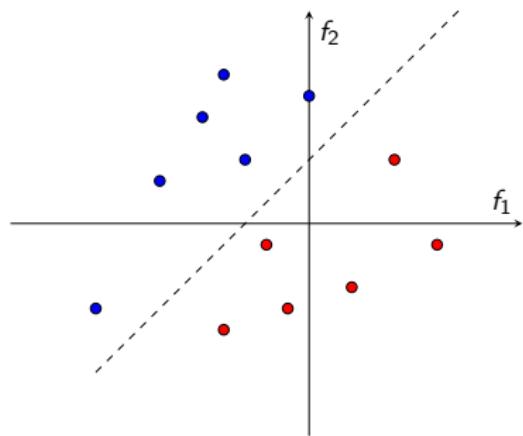
# Supervised Learning

- Machine learning algorithms exploit  $X$  to estimate an  $h$  that has low predictive error.
- In the supervised setting, a machine learning algorithm finds (learns) a mapping from observations  $X$  to outputs  $y$ .
- The rest of this talk explores how such a mapping (**classifier, model**) can be learned.

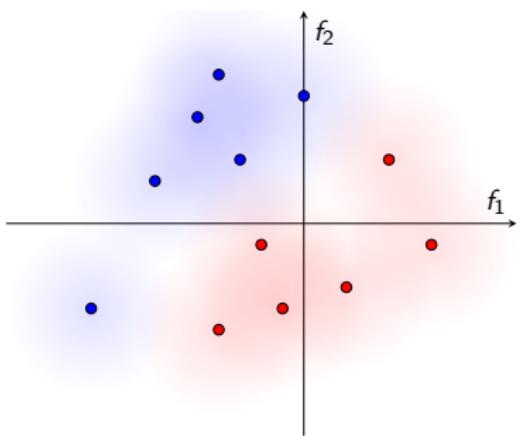


# Learning a Supervised Model

- Supervised Learning algorithms differ by the space of hypothesis  $\mathbb{H}$  they search when selecting an  $h \in \mathbb{H}$  with optimal predictive ability.



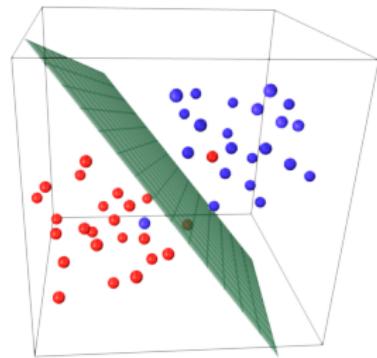
(a) Discriminative Classifier



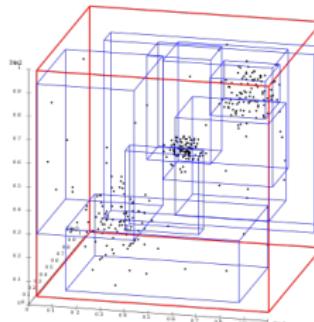
(b) Generative (probabilistic) Classifier

# Discriminative Models

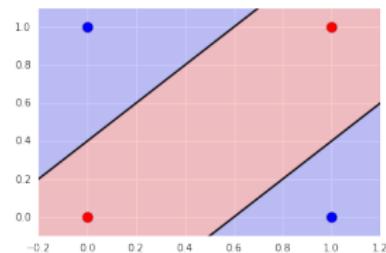
- Discriminative algorithms search the hypothesis space of geometric **decision boundaries** separating classes.



(a) Hyperplanes



(b) Hyper-rectangles



(c) Nested-Hyperplanes

# Outline

1 Introduction

2 Preliminaries

3 Learning

4 Model Evaluation

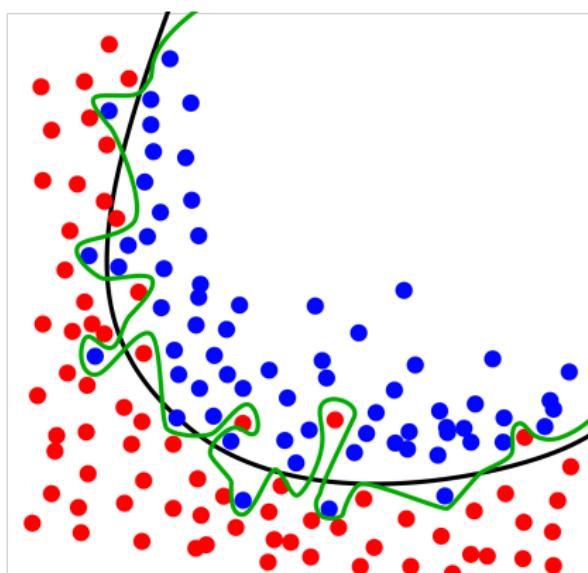
5 Decision Trees

## Generalization Ability

- Recall that a hypothesis function  $h(X) = M(X, y)$  is the output of our learning algorithm applied to our training sample  $X$  and corresponding labels  $y$ .

# Generalization Ability

- Recall that a hypothesis function  $h(X) = M(X, y)$  is the output of our learning algorithm applied to our training sample  $X$  and corresponding labels  $y$ .
- How can we ensure that our selected  $h$  is consistent with  $\mathbb{D}$  (i.e. generalizes)?



# Generalization Ability

- Recall our training set:

$$\begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1d} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & x_{n3} & \dots & x_{nd} \end{bmatrix} \rightarrow \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$



(a) Hold-out Subset



(b) Cross Validation (partitioning)

## Hold Out Validation

Back to our loan example: a supervised algorithm trained on a training partition  $T \subset X$  and evaluated on a test partition  $X \setminus T$  building a model  $h$ :

$$h(T) = \begin{bmatrix} \text{give} \\ \text{deny} \\ \text{deny} \\ \text{give} \\ \text{give} \\ \text{deny} \end{bmatrix}$$

## Hold Out Validation

Back to our loan example: a supervised algorithm trained on a training partition  $T \subset X$  and evaluated on a test partition  $X \setminus T$  building a model  $h$ :

$$h(T) \quad h(X \setminus T)$$

$$\begin{bmatrix} \text{give} \\ \text{deny} \\ \text{deny} \\ \text{give} \\ \text{give} \\ \text{deny} \end{bmatrix} \rightarrow \begin{bmatrix} \text{give} \end{bmatrix}$$

## Hold Out Validation

Back to our loan example: a supervised algorithm trained on a training partition  $T \subset X$  and evaluated on a test partition  $X \setminus T$  building a model  $h$ :

$$\begin{array}{c} h(T) \\ \left[ \begin{array}{c} \text{give} \\ \text{deny} \\ \text{deny} \\ \text{give} \\ \text{give} \\ \text{deny} \end{array} \right] \end{array} \rightarrow \begin{array}{c} h(X \setminus T) \\ \left[ \begin{array}{c} \text{give} \\ \quad \quad \quad \text{True Positive (TP)} \end{array} \right] \end{array}$$

## Hold Out Validation

Back to our loan example: a supervised algorithm trained on a training partition  $T \subset X$  and evaluated on a test partition  $X \setminus T$  building a model  $h$ :

$$\begin{array}{c} h(T) \\ \left[ \begin{array}{c} \text{give} \\ \text{deny} \\ \text{deny} \\ \text{give} \\ \text{give} \\ \text{deny} \end{array} \right] \end{array} \rightarrow \begin{array}{c} h(X \setminus T) \\ \left[ \begin{array}{c} \text{give} \\ \text{give} \end{array} \right] \end{array}$$

True Positive (TP)

## Hold Out Validation

Back to our loan example: a supervised algorithm trained on a training partition  $T \subset X$  and evaluated on a test partition  $X \setminus T$  building a model  $h$ :

$$\begin{array}{c} h(T) \\ \left[ \begin{array}{c} \text{give} \\ \text{deny} \\ \text{deny} \\ \text{give} \\ \text{give} \\ \text{deny} \end{array} \right] \end{array} \rightarrow \begin{array}{c} h(X \setminus T) \\ \left[ \begin{array}{c} \text{give} \\ \text{give} \end{array} \right] \end{array}$$

True Positive (TP)

False Positive (FP)

## Hold Out Validation

Back to our loan example: a supervised algorithm trained on a training partition  $T \subset X$  and evaluated on a test partition  $X \setminus T$  building a model  $h$ :

$$\begin{matrix} h(T) \\ \left[ \begin{array}{c} \text{give} \\ \text{deny} \\ \text{deny} \\ \text{give} \\ \text{give} \\ \text{deny} \end{array} \right] \end{matrix} \rightarrow \begin{matrix} h(X \setminus T) \\ \left[ \begin{array}{c} \text{give} \\ \text{give} \\ \text{deny} \end{array} \right] \end{matrix}$$

True Positive (TP)

False Positive (FP)

## Hold Out Validation

Back to our loan example: a supervised algorithm trained on a training partition  $T \subset X$  and evaluated on a test partition  $X \setminus T$  building a model  $h$ :

$$\begin{array}{c} h(T) \\ \left[ \begin{array}{c} \text{give} \\ \text{deny} \\ \text{deny} \\ \text{give} \\ \text{give} \\ \text{deny} \end{array} \right] \end{array} \rightarrow \begin{array}{c} h(X \setminus T) \\ \left[ \begin{array}{c} \text{give} \\ \text{give} \\ \text{deny} \end{array} \right] \\ \begin{array}{l} \text{True Positive (TP)} \\ \text{False Positive (FP)} \\ \text{True Negative (TN)} \end{array} \end{array}$$

## Hold Out Validation

Back to our loan example: a supervised algorithm trained on a training partition  $T \subset X$  and evaluated on a test partition  $X \setminus T$  building a model  $h$ :

$$\begin{array}{c} h(T) \\ \left[ \begin{array}{c} \text{give} \\ \text{deny} \\ \text{deny} \\ \text{give} \\ \text{give} \\ \text{deny} \end{array} \right] \end{array} \rightarrow \begin{array}{c} h(X \setminus T) \\ \left[ \begin{array}{c} \text{give} \\ \text{give} \\ \text{deny} \\ \text{give} \end{array} \right] \end{array}$$

True Positive (TP)  
False Positive (FP)  
True Negative (TN)

## Hold Out Validation

Back to our loan example: a supervised algorithm trained on a training partition  $T \subset X$  and evaluated on a test partition  $X \setminus T$  building a model  $h$ :

$h(T)$	$h(X \setminus T)$
give	True Positive (TP)
deny	False Positive (FP)
deny	True Negative (TN)
give	True Positive (TP)
give	
deny	

## Hold Out Validation

Back to our loan example: a supervised algorithm trained on a training partition  $T \subset X$  and evaluated on a test partition  $X \setminus T$  building a model  $h$ :

$h(T)$	$h(X \setminus T)$
give	True Positive (TP)
deny	False Positive (FP)
deny	True Negative (TN)
give	True Positive (TP)
give	
deny	

## Hold Out Validation

Back to our loan example: a supervised algorithm trained on a training partition  $T \subset X$  and evaluated on a test partition  $X \setminus T$  building a model  $h$ :

$h(T)$		$h(X \setminus T)$
give	→	True Positive (TP)
deny		False Positive (FP)
deny		True Negative (TN)
give		True Positive (TP)
give		False Negative (FN)
deny		

## Hold Out Validation

Back to our loan example: a supervised algorithm trained on a training partition  $T \subset X$  and evaluated on a test partition  $X \setminus T$  building a model  $h$ :

$h(T)$		$h(X \setminus T)$
give	→	True Positive (TP)
deny		False Positive (FP)
deny		True Negative (TN)
give		True Positive (TP)
give		False Negative (FN)
deny		give

## Hold Out Validation

Back to our loan example: a supervised algorithm trained on a training partition  $T \subset X$  and evaluated on a test partition  $X \setminus T$  building a model  $h$ :

$h(T)$		$h(X \setminus T)$
give	→	True Positive (TP)
deny		False Positive (FP)
deny		True Negative (TN)
give		True Positive (TP)
give		False Negative (FN)
deny		False Positive (FP)

## Hold Out Validation

Back to our loan example: a supervised algorithm trained on a training partition  $T \subset X$  and evaluated on a test partition  $X \setminus T$  building a model  $h$ :

$h(T)$	$h(X \setminus T)$
give	True Positive (TP)
deny	False Positive (FP)
deny	True Negative (TN)
give	True Positive (TP)
give	False Negative (FN)
deny	False Positive (FP)

	PP	PN
P	2	1
N	2	1

## Hold Out Validation

Back to our loan example: a supervised algorithm trained on a training partition  $T \subset X$  and evaluated on a test partition  $X \setminus T$  building a model  $h$ :

$h(T)$	$h(X \setminus T)$
give	True Positive (TP)
deny	False Positive (FP)
deny	True Negative (TN)
give	True Positive (TP)
give	False Negative (FN)
deny	False Positive (FP)

	PP	PN
P	2	1
N	2	1

$$\text{Accuracy} = \frac{TP + TN}{N} = \frac{3}{6}$$

## Hold Out Validation

Back to our loan example: a supervised algorithm trained on a training partition  $T \subset X$  and evaluated on a test partition  $X \setminus T$  building a model  $h$ :

$h(T)$	$h(X \setminus T)$
give	True Positive (TP)
deny	False Positive (FP)
deny	True Negative (TN)
give	True Positive (TP)
give	False Negative (FN)
deny	False Positive (FP)

	PP	PN
P	2	1
N	2	1

$$\text{Accuracy} = \frac{TP + TN}{N} = \frac{3}{6}$$

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{2}{3}$$

# Outline

1 Introduction

2 Preliminaries

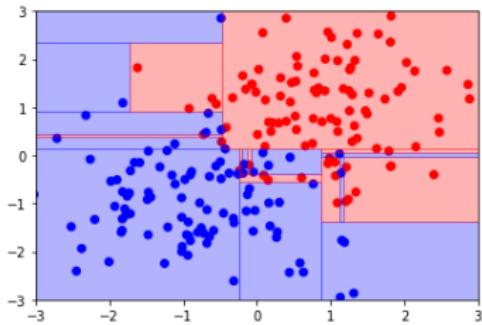
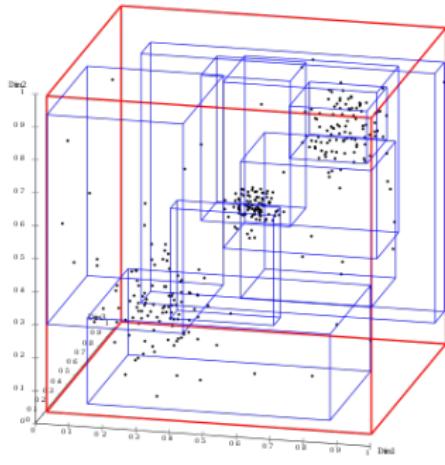
3 Learning

4 Model Evaluation

5 Decision Trees

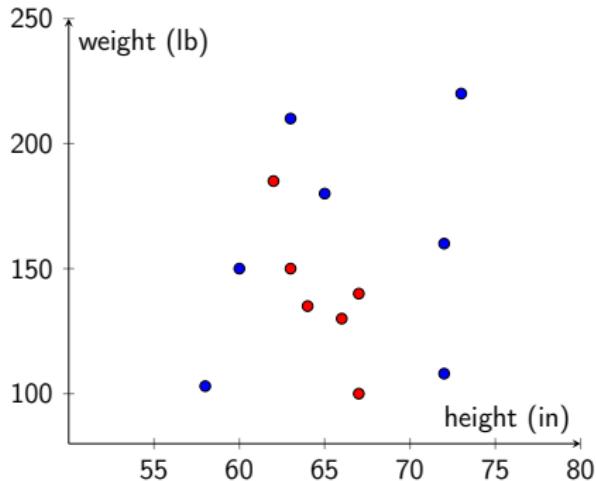
# The Supervised Decision Tree

- Decision Trees are a class of discriminative learning algorithm searching the hypothesis space of hyper-rectangles.
- Given a labeled training sample  $X$ , a Decision Tree produces a model  $h$  mapping partitioned hyper-rectangles in  $\mathbb{R}^d$  to target classes.



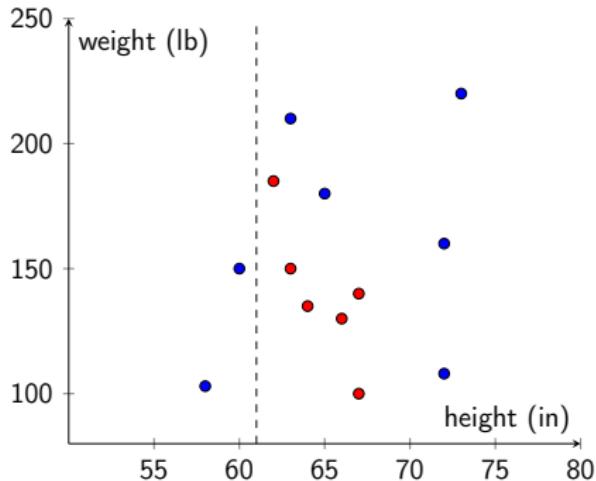
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



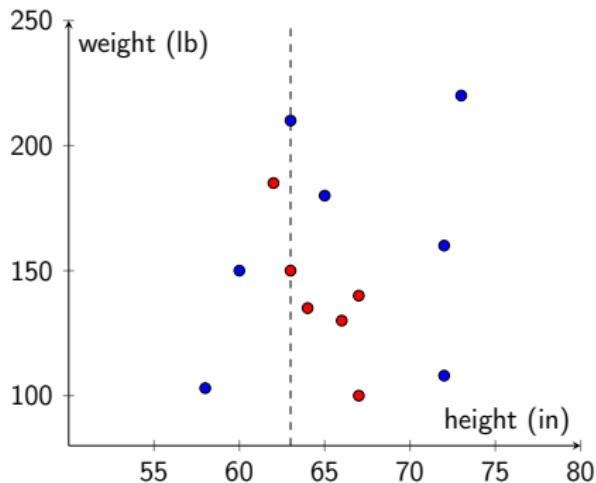
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



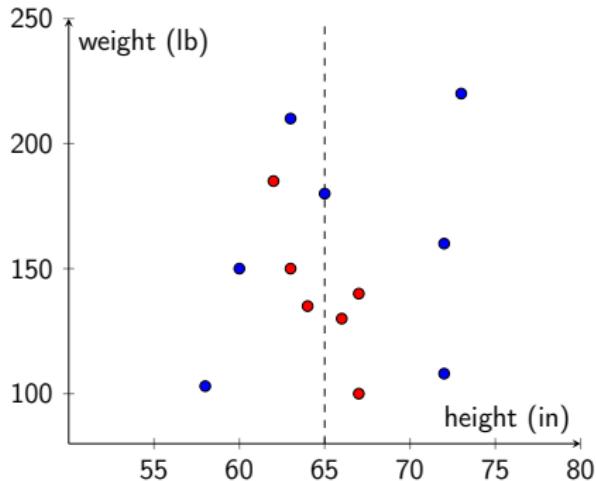
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



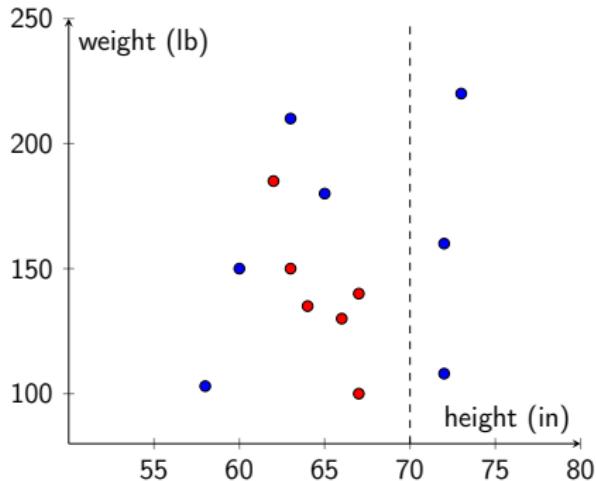
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



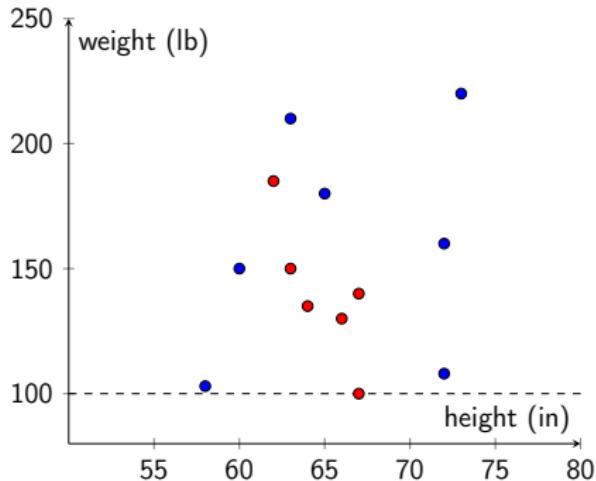
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



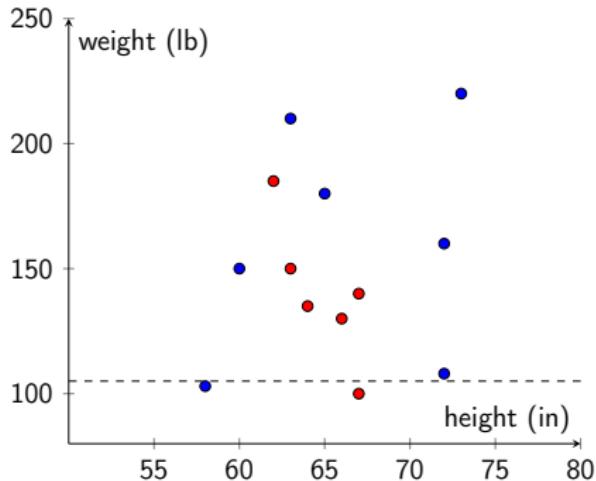
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



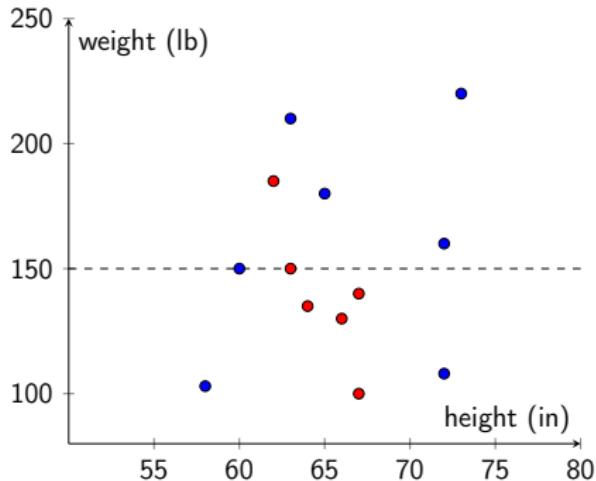
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



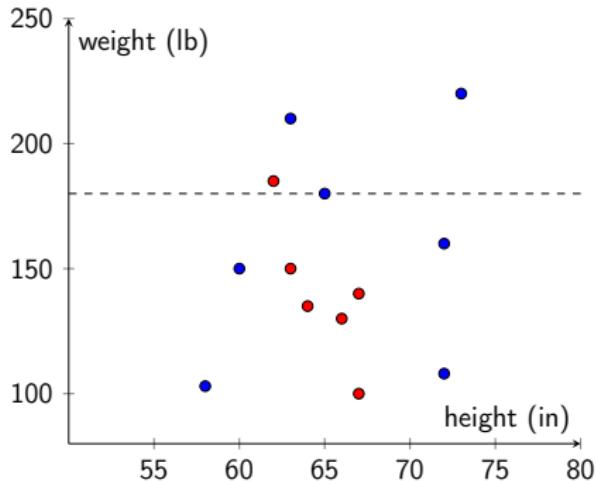
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



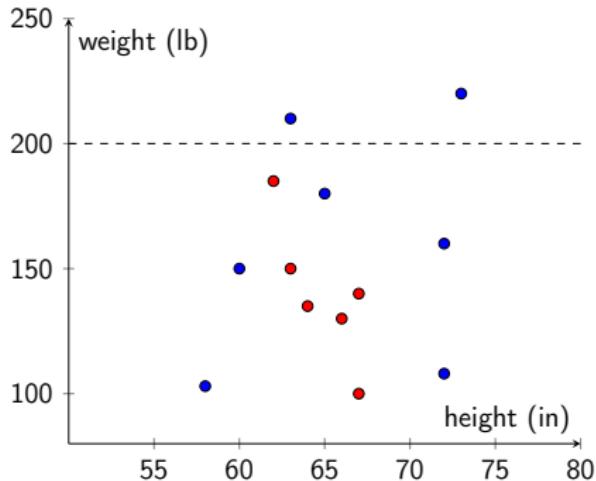
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



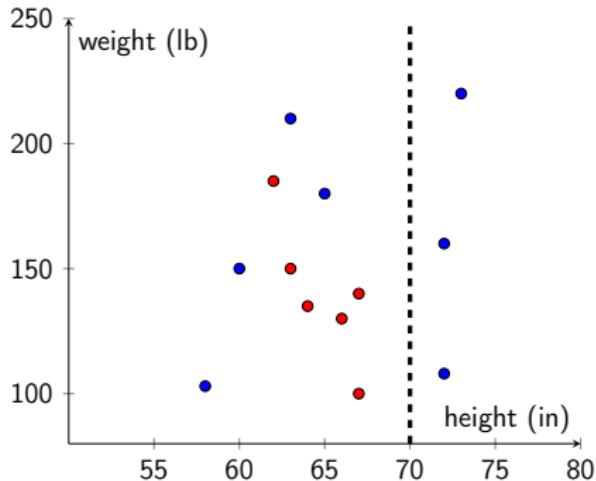
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



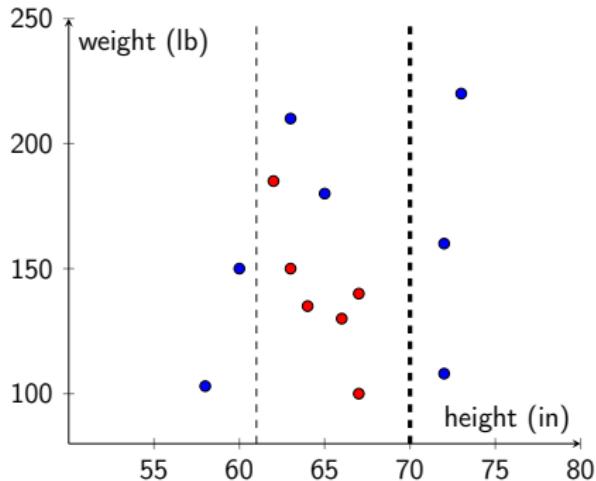
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



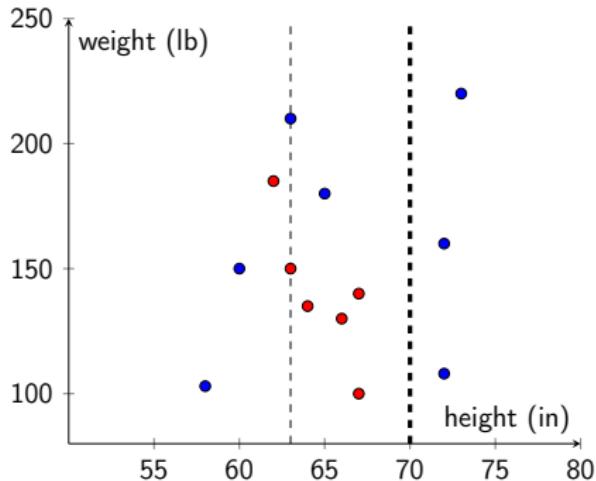
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



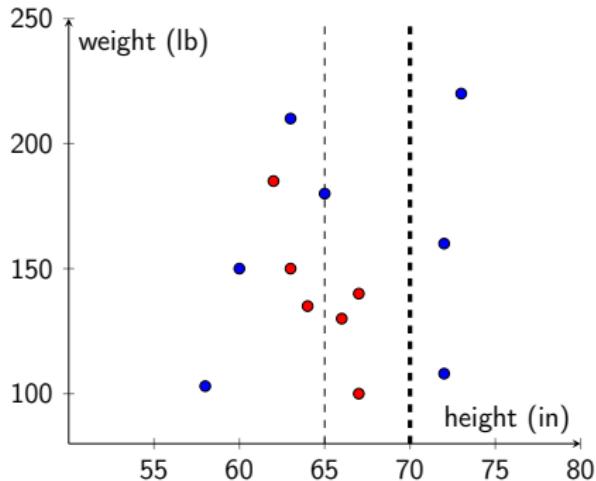
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



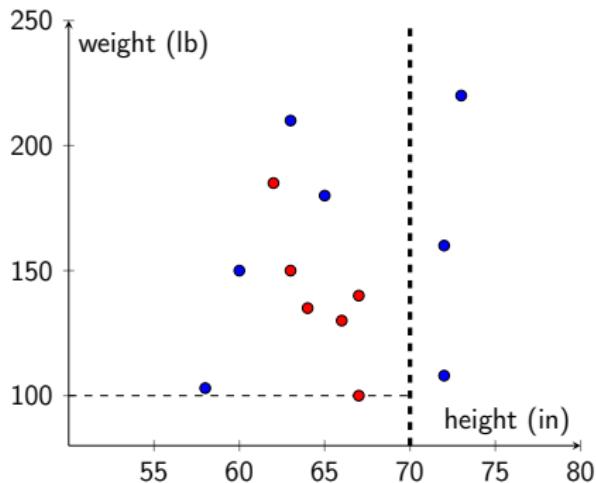
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



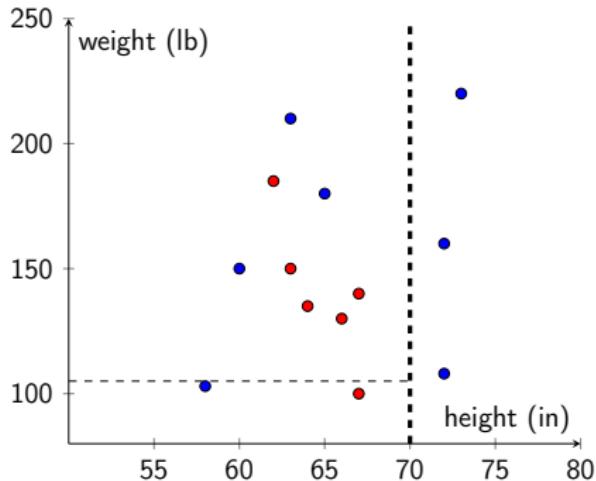
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



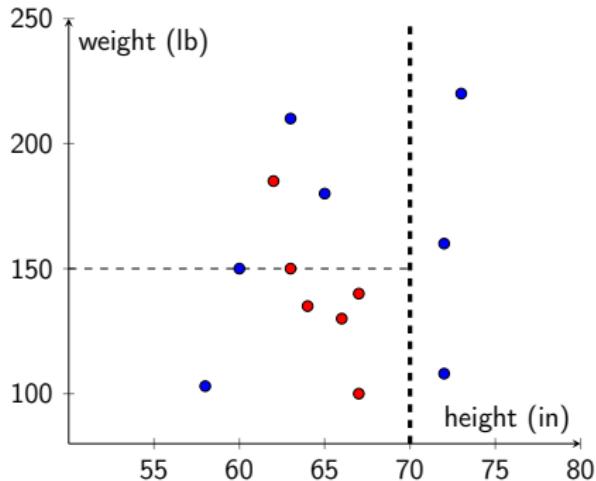
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



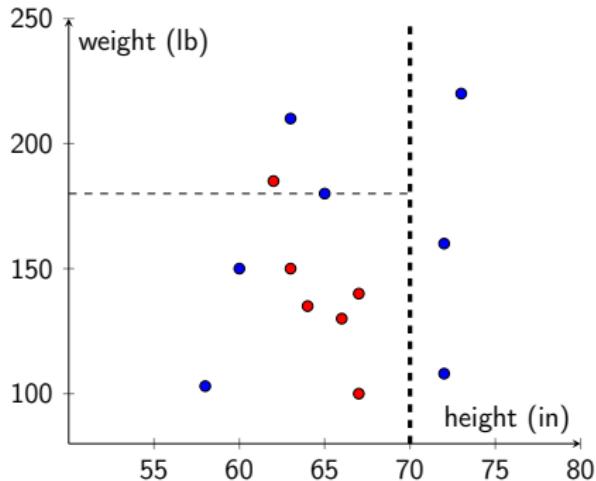
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



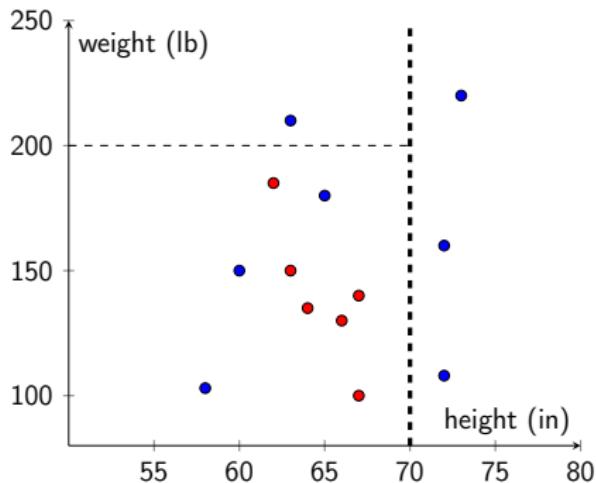
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



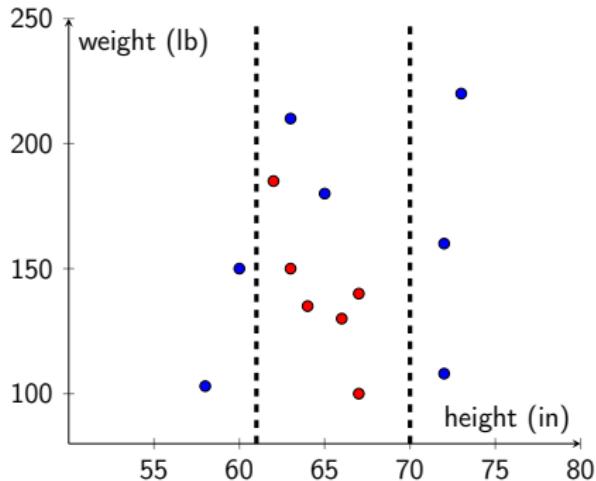
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



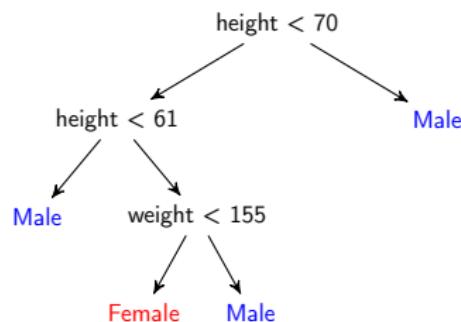
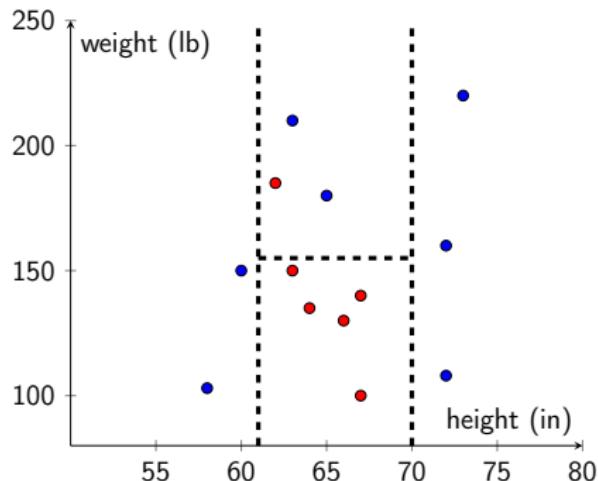
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



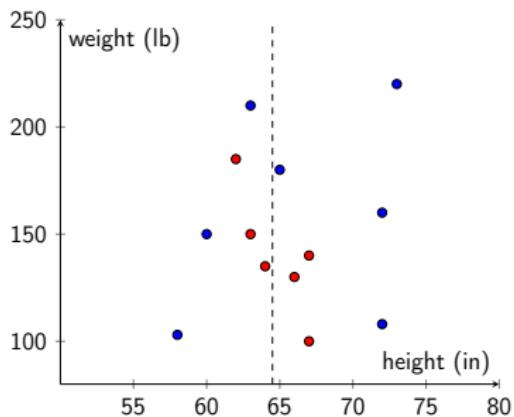
# Decision Tree Induction

- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



# The Impurity Criterion: A Heuristic

- A potential split (partition) induces two class distributions derived from the class distribution of the parent partition.



# The Impurity Criterion: A Heuristic

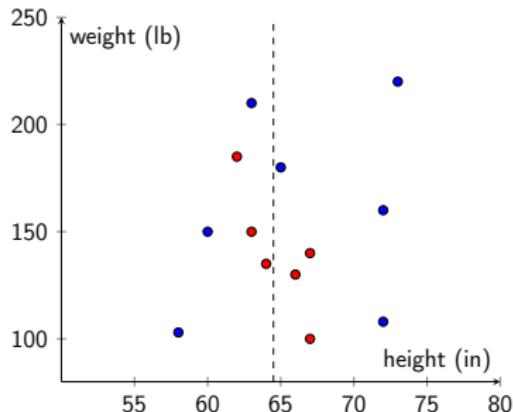
- A potential split (partition) induces two class distributions derived from the class distribution of the parent partition.

$$P_{L,\text{male}} = \frac{3}{6}$$

$$P_{L,\text{male}} = \frac{3}{6}$$

$$P_{R,\text{male}} = \frac{4}{7}$$

$$P_{R,\text{female}} = \frac{3}{7}$$



# The Impurity Criterion: A Heuristic

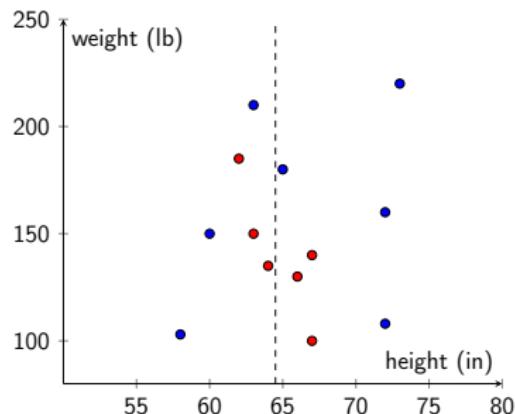
- A potential split (partition) induces two class distributions derived from the class distribution of the parent partition.

$$P_{L,\text{male}} = \frac{3}{6}$$

$$P_{L,\text{male}} = \frac{3}{6}$$

$$P_{R,\text{male}} = \frac{4}{7}$$

$$P_{R,\text{female}} = \frac{3}{7}$$



$$\text{Gini}_R = 1 - \sum_{p_c \in P_R} p_c^2$$

# The Impurity Criterion: A Heuristic

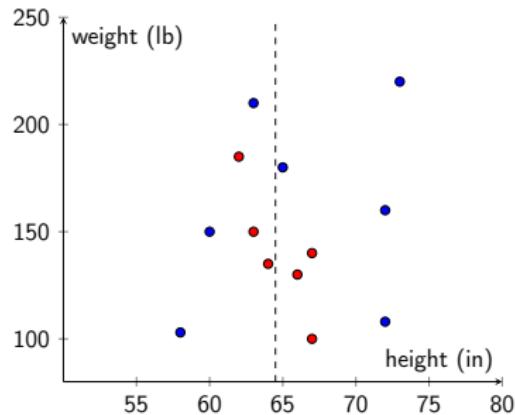
- A potential split (partition) induces two class distributions derived from the class distribution of the parent partition.

$$P_{L,\text{male}} = \frac{3}{6}$$

$$P_{L,\text{male}} = \frac{3}{6}$$

$$P_{R,\text{male}} = \frac{4}{7}$$

$$P_{R,\text{female}} = \frac{3}{7}$$



$$\text{Gini}_R = 1 - \sum_{p_c \in P_R} p_c^2$$

$$\text{Gini}_R = 1 - P_R(\text{Male})^2 - P_R(\text{Female})^2$$

# Decision Trees

- Once a tree is constructed from a training set, a new instance can be assigned a label by cascading down the tree until a terminal (leaf) node is reached.

# Decision Trees

- Once a tree is constructed from a training set, a new instance can be assigned a label by cascading down the tree until a terminal (leaf) node is reached.
- Decision Trees are one the most powerful learning algorithms.
  - Interpretable
  - Lightning fast to grow.
  - Very effective extensions to clustering, anomaly detection.

# Decision Trees

- Once a tree is constructed from a training set, a new instance can be assigned a label by cascading down the tree until a terminal (leaf) node is reached.
- Decision Trees are one the most powerful learning algorithms.
  - Interpretable
  - Lightning fast to grow.
  - Very effective extensions to clustering, anomaly detection.
- An ensemble of randomized decision trees, the [Random Forest](#), is the most powerful algorithm for supervised prediction on small samples of low dimensional data.

# Decision Trees

- Once a tree is constructed from a training set, a new instance can be assigned a label by cascading down the tree until a terminal (leaf) node is reached.
- Decision Trees are one the most powerful learning algorithms.
  - Interpretable
  - Lightning fast to grow.
  - Very effective extensions to clustering, anomaly detection.
- An ensemble of randomized decision trees, the [Random Forest](#), is the most powerful algorithm for supervised prediction on small samples of low dimensional data.
- Simplicity has a price.
  - Underperforms on high dimensional data (i.e. text, images).
  - Requires all data to be stored in memory.
  - No error convergence bounds or rigorous foundation.

# Conclusions

- We explored a thin sliver of machine learning.
  - A rather old, causi-discriminative algorithm for independently sampled instances from  $\mathbb{D}$ .

# Conclusions

- We explored a thin sliver of machine learning.
  - A rather old, causi-discriminative algorithm for independently sampled instances from  $\mathbb{D}$ .
  - What about when examples have structure (sequences, hierarchies)?

# Conclusions

- We explored a thin sliver of machine learning.
  - A rather old, causi-discriminative algorithm for independently sampled instances from  $\mathbb{D}$ .
  - What about when examples have structure (sequences, hierarchies)?
- We worked under the assumption that our data is nicely formatted and plentiful.
  - In application, solving a problem with a machine learning solution is 90% data pre-processing.

# Conclusions

- We explored a thin sliver of machine learning.
  - A rather old, causi-discriminative algorithm for independently sampled instances from  $\mathbb{D}$ .
  - What about when examples have structure (sequences, hierarchies)?
- We worked under the assumption that our data is nicely formatted and plentiful.
  - In application, solving a problem with a machine learning solution is 90% data pre-processing.
  - Often gathering labeled data is costly and difficult (e.g. text annotation).

# Conclusions

- We explored a thin sliver of machine learning.
  - A rather old, causi-discriminative algorithm for independently sampled instances from  $\mathbb{D}$ .
  - What about when examples have structure (sequences, hierarchies)?
- We worked under the assumption that our data is nicely formatted and plentiful.
  - In application, solving a problem with a machine learning solution is 90% data pre-processing.
  - Often gathering labeled data is costly and difficult (e.g. text annotation).
- Nonetheless, learning algorithms are increasingly making important and crucial decision in society.

# Conclusions

- We explored a thin sliver of machine learning.
  - A rather old, causi-discriminative algorithm for independently sampled instances from  $\mathbb{D}$ .
  - What about when examples have structure (sequences, hierarchies)?
- We worked under the assumption that our data is nicely formatted and plentiful.
  - In application, solving a problem with a machine learning solution is 90% data pre-processing.
  - Often gathering labeled data is costly and difficult (e.g. text annotation).
- Nonetheless, learning algorithms are increasingly making important and crucial decision in society.
- Future Outlooks

# Conclusions

- We explored a thin sliver of machine learning.
  - A rather old, causi-discriminative algorithm for independently sampled instances from  $\mathbb{D}$ .
  - What about when examples have structure (sequences, hierarchies)?
- We worked under the assumption that our data is nicely formatted and plentiful.
  - In application, solving a problem with a machine learning solution is 90% data pre-processing.
  - Often gathering labeled data is costly and difficult (e.g. text annotation).
- Nonetheless, learning algorithms are increasingly making important and crucial decision in society.
- Future Outlooks
  - Machine learning algorithms currently can do nothing more than estimate a function (albeit very well). What about logical reasoning, abduction and inference?

# Conclusion

**Thank you for your attention!  
Questions?**

<https://bit.ly/2tPVPfv>

aymulyar@vcu.edu  
[www.andriymulyar.com](http://www.andriymulyar.com)