Introduction to Machine Learning

Andriy Mulyar[1]

[1]Department of Computer Science
Virginia Commonwealth University
Richmond, VA
USA

aymulyar@vcu.edu

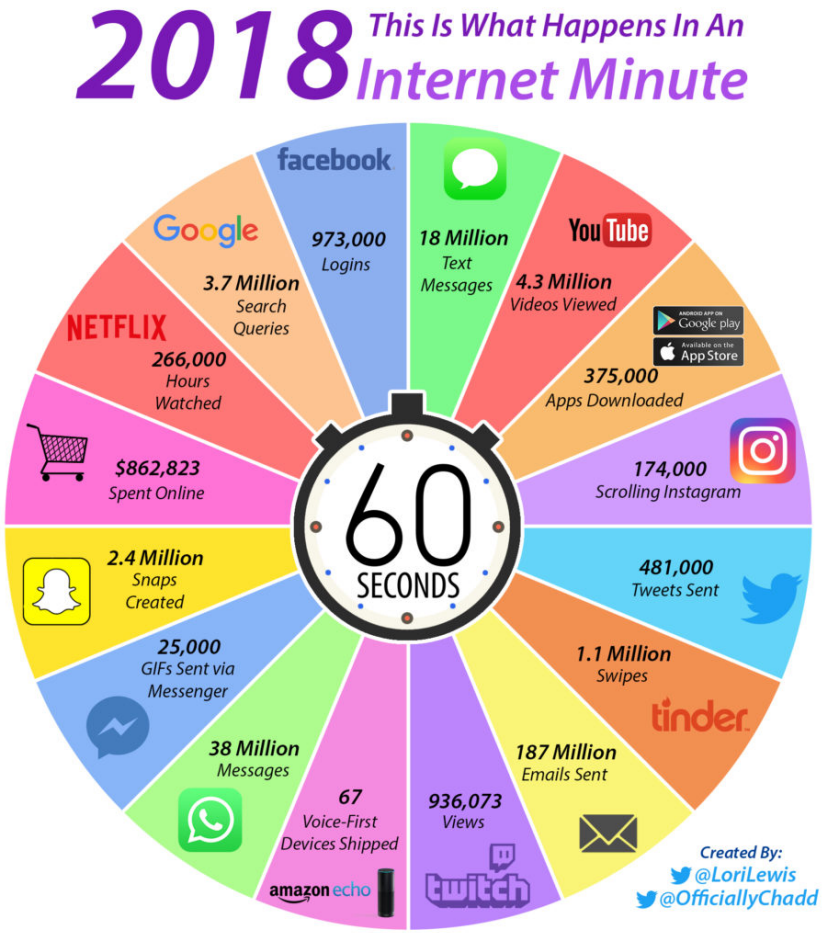
March 5 2019

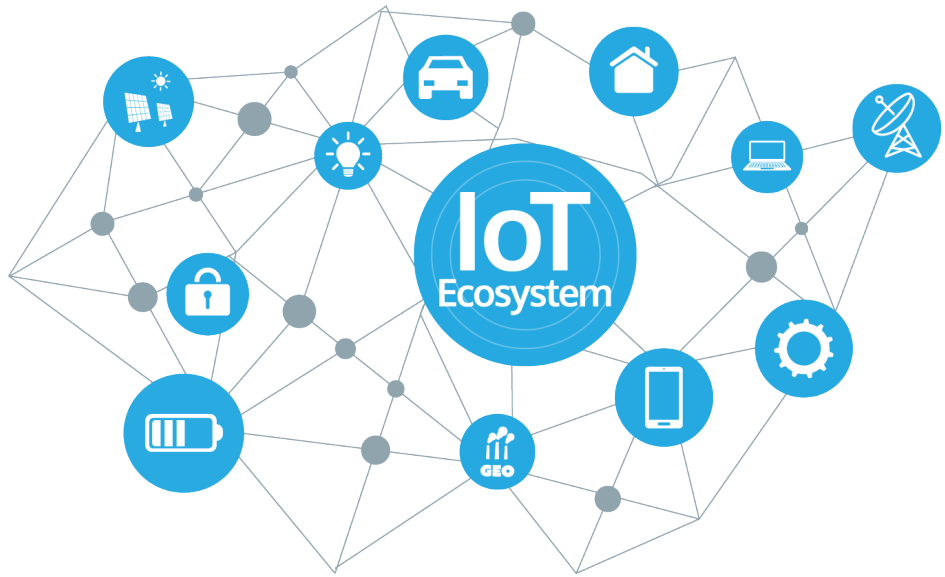
Outline

Outline

Our Reality: We Are Drowning In Data

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





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.

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)

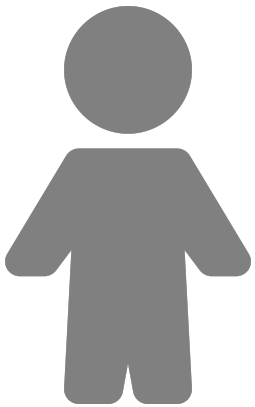
Pattern Recognition and Machine Learning

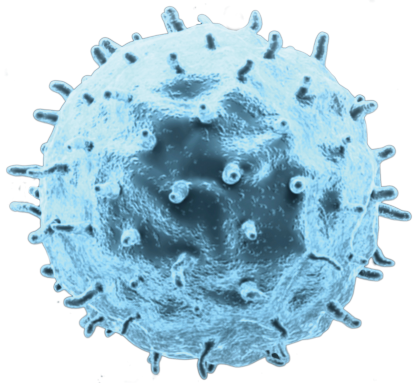
- This talk focuses on gaining insight into the underlying algorithms and structures that produce models capable of tackling such problems.
- We will explore in-depth a heuristical algorithm - the supervised Decision Tree.

Outline

Entities

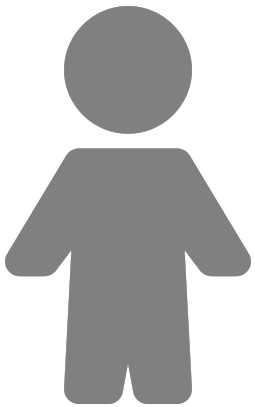
- The world around us can be succinctly described as comprised of entities.

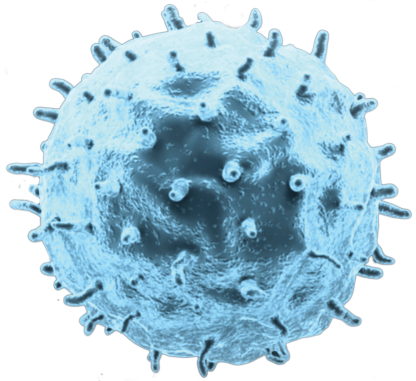




Entities

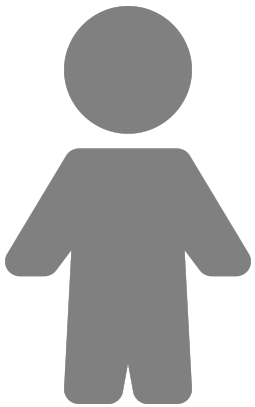
- The world around us can be succinctly described as comprised of entities.
- Entities interact (relate) with other entities and in the process possibly form new entities.

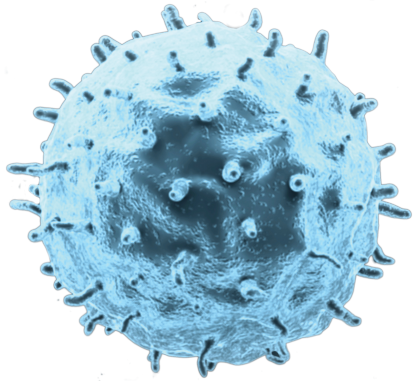




Entities

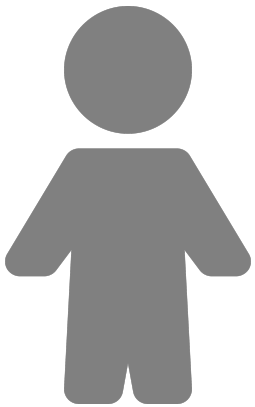
- The world around us can be succinctly described as comprised of entities.
- Entities interact (relate) with other entities and in the process possibly form new entities.
- Entities are composed of attributes.

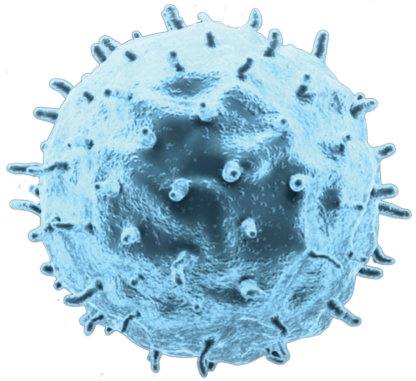




Entities

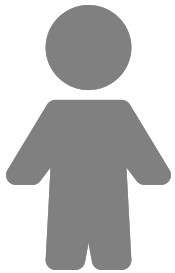
- The world around us can be succinctly described as comprised of entities.
- Entities interact (relate) with other entities and in the process possibly form new entities.
- Entities are composed of attributes.
- An instance of an entity is a particular manifestation of the attributes that comprise the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





Instances of Entities

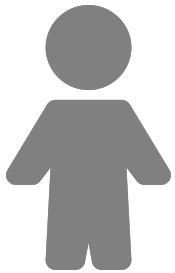
- A collection of instances $x_i \in \mathbb{D}$ from an entity domain $\mathbb{D}$ forms a sample from the domain.



	account_balance	income	age	has_defaulted	credit_worthy
x_1	57103	50000	25	0	?
x_2	20343	40000		1	?
x_3	80000	150000	4	1	?
x_4	500		17	0	?
x_5	5000	1000	21	0	?

Instances of Entities

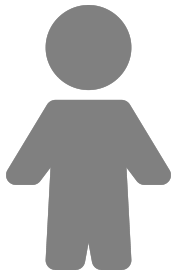
- A collection of instances $x_i \in \mathbb{D}$ from an entity domain $\mathbb{D}$ forms a sample from the domain.
- The dimension d of $\mathbb{D}$ is the number of attributes describing its entity.



	account_balance	income	age	has_defaulted	credit_worthy
x_1	57103	50000	25	0	?
x_2	20343	40000		1	?
x_3	80000	150000	4	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	4	1	?
x_4	500		17	0	?
x_5	5000	1000	21	0	?

Outline

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:

$$X = \{x_i \in \mathbb{D}\} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \cdots & x_{1d} \\ x_{21} & x_{22} & x_{23} & \cdots & x_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{d2} & x_{d3} & \cdots & x_{nd} \end{bmatrix}$$

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:

$$X = \{x_i \in \mathbb{D}\} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1d} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{d2} & x_{d3} & \dots & x_{nd} \end{bmatrix}$$

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:

$$X = \{x_i \in \mathbb{D}\} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1d} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{d2} & x_{d3} & \dots & x_{nd} \end{bmatrix}$$

Supervised Learning

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

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:

$$X = \{x_i \in \mathbb{D}\} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1d} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{d2} & x_{d3} & \dots & x_{nd} \end{bmatrix}$$

Supervised Learning

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

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:

$$X = \{x_i \in \mathbb{D}\} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \cdots & x_{1d} \\ x_{21} & x_{22} & x_{23} & \cdots & x_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{d2} & x_{d3} & \cdots & x_{nd} \end{bmatrix}$$

Supervised Learning

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

Unsupervised Learning

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

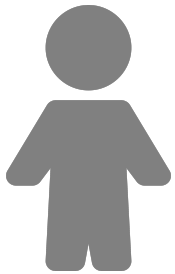
Paradigms

Supervised Learning

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

Unsupervised Learning

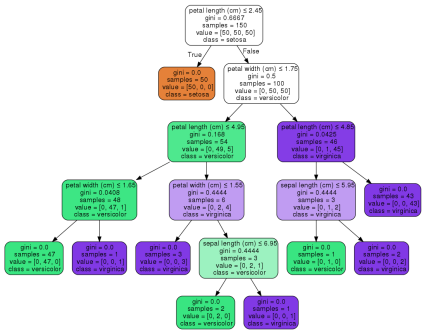
- Denote $C = \{C_i \subseteq D\}$ as clusters.
- Use X to construct a model $h : \mathbb{D} \to C_i$.

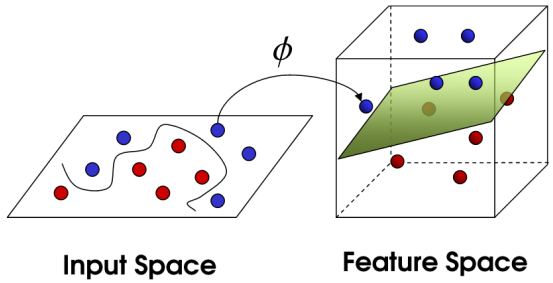


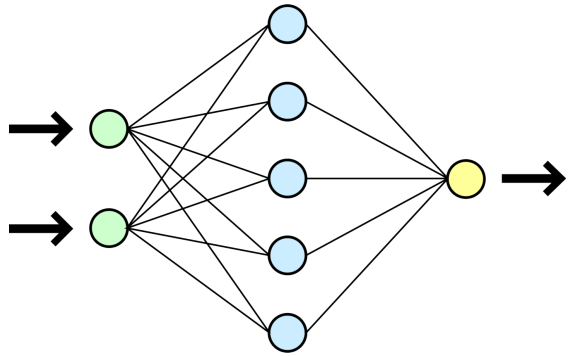
	account_balance	income	age	has_defaulted	credit_worthy
x_1	57103	50000	25	0	?
x_2	20343	40000		1	?
x_3	80000	150000	4	1	?
x_4	500		17	0	?
x_5	5000	1000	21	0	?

Supervised Learning

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

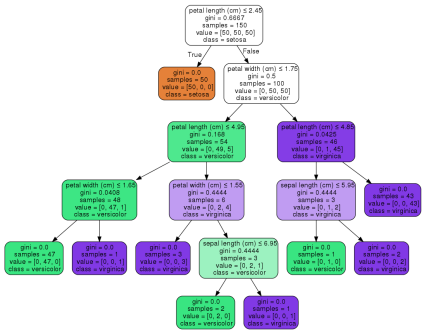


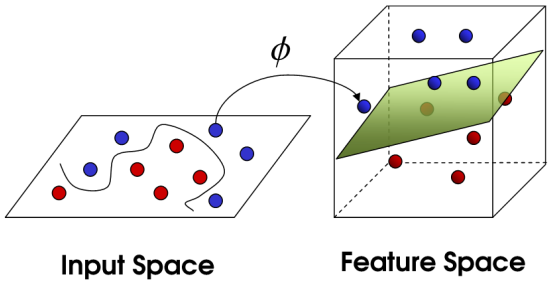


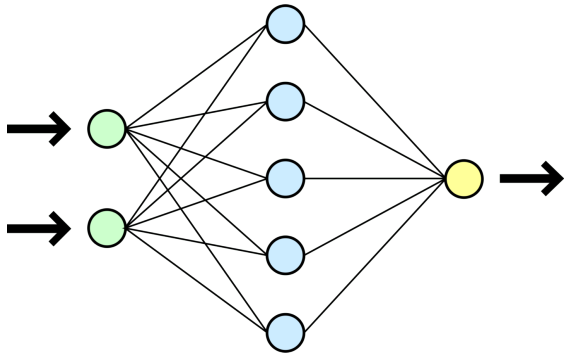


Supervised Learning

- 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.

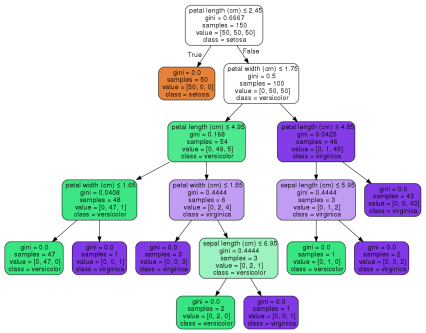


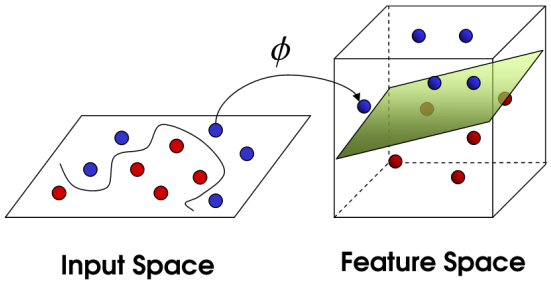


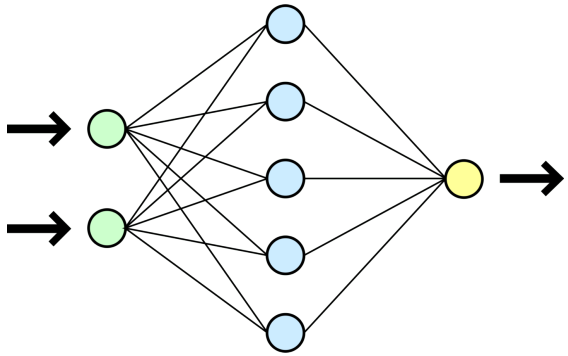


Supervised Learning

- 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.

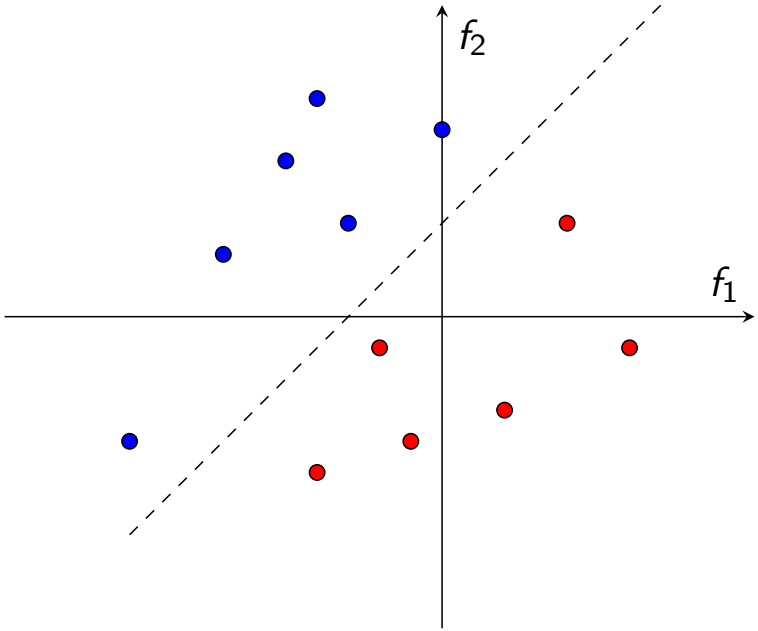




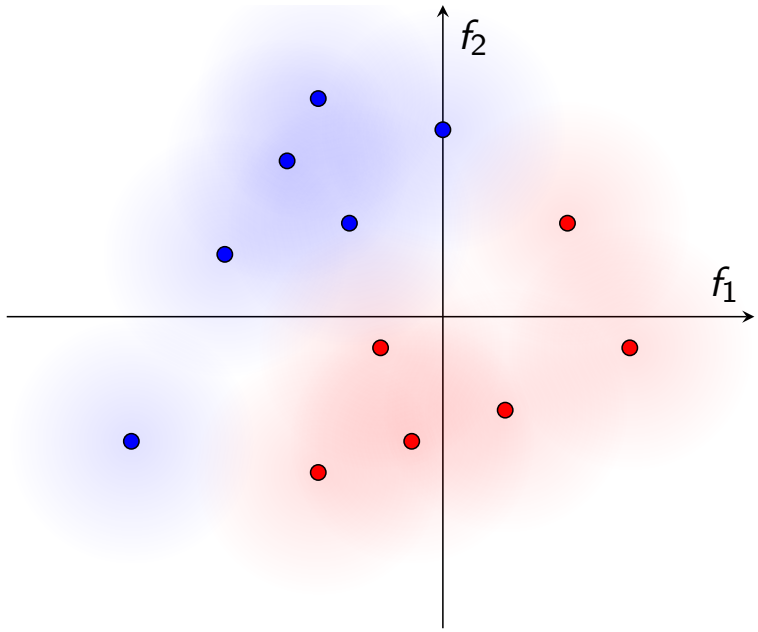


Learning a Supervised Model

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



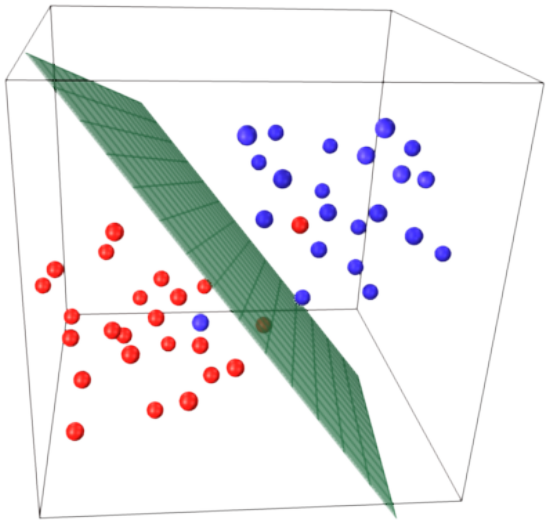
(a) Discriminative Classifier



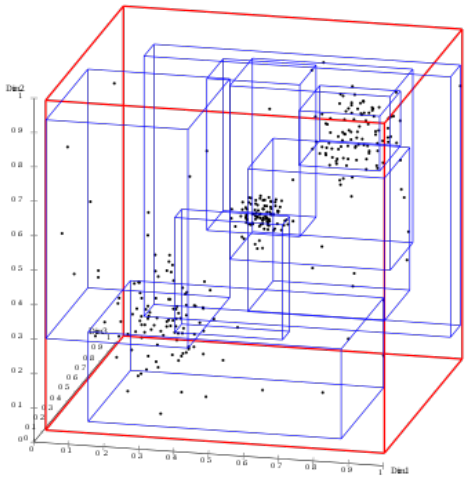
(b) Generative (probabilistic) Classifier

Discriminative Models

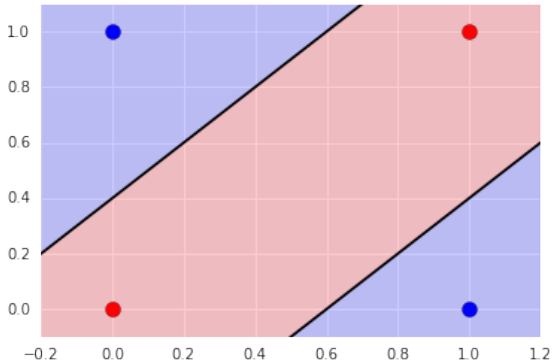
- Discriminative algorithms search the hypothesis space of geometric decision boundaries separating classes.



(a) Hyperplanes



(b) Hyper-rectangles



(c) Nested-Hyperplanes

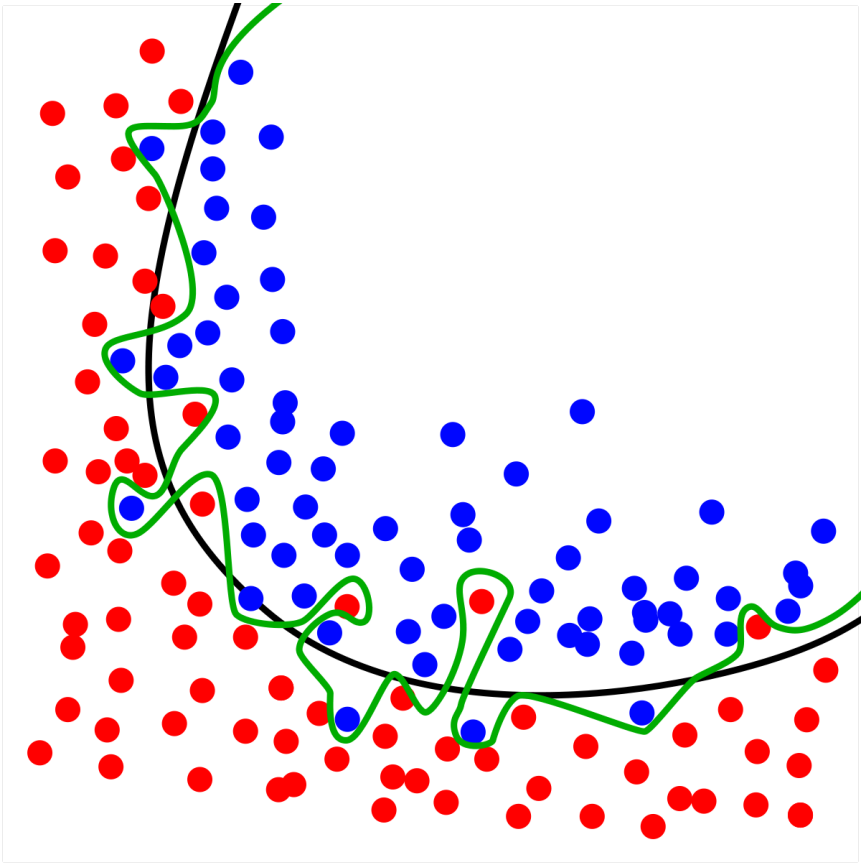
Outline

Generalization Ability

- Recall that a hypothesis function (model) h is the output of our learning algorithm applied to our training sample X.

Generalization Ability

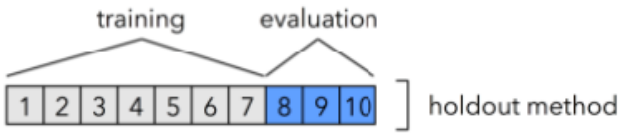
- Recall that a hypothesis function (model) h is the output of our learning algorithm applied to our training sample X.
- How can we ensure that our selected h is consistent with $\mathbb{D}$ (i.e. generalizes)?



Generalization Ability

- Recall our training set:

$$\begin{bmatrix} x_{11} & x_{12} & x_{13} & \cdots & x_{1d} \\ x_{21} & x_{22} & x_{23} & \cdots & x_{2d} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{d2} & x_{d3} & \cdots & x_{nd} \end{bmatrix} \rightarrow \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$



(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:

$$\begin{matrix} h(T) \\ \begin{bmatrix} \text{give} \\ \text{deny} \\ \text{deny} \\ \text{give} \\ \text{give} \\ \text{deny} \end{bmatrix} \end{matrix} \quad \rightarrow \quad \begin{matrix} h(X \setminus T) \\ \begin{bmatrix} \text{give} \\ \\ \\ \\ \\ \end{bmatrix} \end{matrix}$$

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:

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

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:

$$
\begin{array}{cc}
h(T) & \\
\begin{bmatrix} \text{give} \\ \text{deny} \\ \text{deny} \\ \text{give} \\ \text{give} \\ \text{deny} \end{bmatrix} & \rightarrow
\end{array}
\quad
\begin{array}{cc}
h(X \setminus T) & \\
\begin{bmatrix} \text{give} \\ \text{give} \\ \\ \\ \\ \end{bmatrix} & \text{True Positive (TP)}
\end{array}
$$

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:

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

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:

$$\begin{matrix} h(T) \\ \begin{bmatrix} \text{give} \\ \text{deny} \\ \text{deny} \\ \text{give} \\ \text{give} \\ \text{deny} \end{bmatrix} \end{matrix} \rightarrow \begin{matrix} h(X \setminus T) \\ \begin{bmatrix} \text{give} \\ \text{give} \\ \text{deny} \\ \\ \\ \\ \end{bmatrix} \end{matrix} \begin{matrix} \\ \text{True Positive (TP)} \\ \text{False Positive (FP)} \\ \\ \\ \\ \\ \end{matrix}$$

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:

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

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:

$$
\begin{array}{ccc}
h(T) & & h(X \setminus T) \\
\begin{bmatrix} \text{give} \\ \text{deny} \\ \text{deny} \\ \text{give} \\ \text{give} \\ \text{deny} \end{bmatrix} & \rightarrow & \begin{bmatrix} \text{give} \\ \text{give} \\ \text{deny} \\ \text{give} \\ \\ \end{bmatrix}
\begin{array}{l} \text{True Positive (TP)} \\ \text{False Positive (FP)} \\ \text{True Negative (TN)} \end{array}
\end{array}
$$

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:

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

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:

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

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:

$$
\begin{array}{c}
h(T) \\
\begin{bmatrix} \text{give} \\ \text{deny} \\ \text{deny} \\ \text{give} \\ \text{give} \\ \text{deny} \end{bmatrix}
\end{array}
\quad \rightarrow \quad
\begin{array}{cc}
h(X \setminus T) & \\
\begin{bmatrix} \text{give} \\ \text{give} \\ \text{deny} \\ \text{give} \\ \text{deny} \\ \\ \end{bmatrix} &
\begin{array}{l} \text{True Positive (TP)} \\ \text{False Positive (FP)} \\ \text{True Negative (TN)} \\ \text{True Positive (TP)} \\ \text{False Negative (FN)} \end{array}
\end{array}
$$

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:

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

	PP	PN
P	2	1
N	2	1

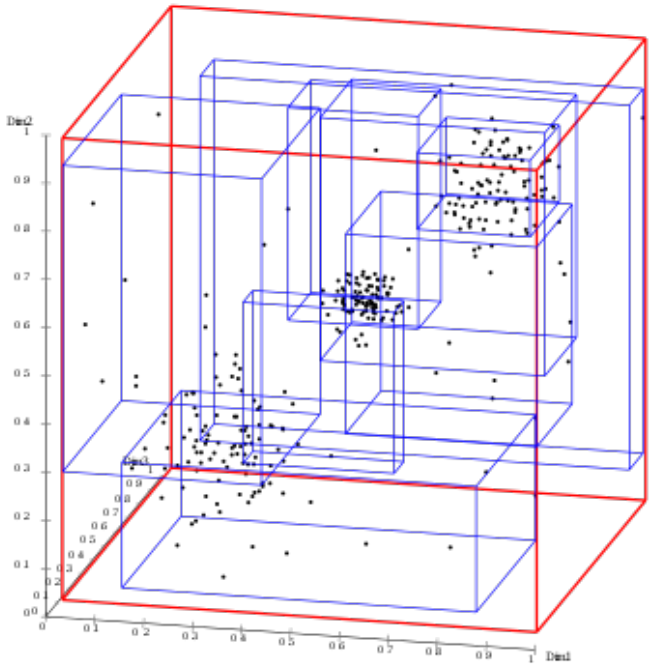
$$\text{Accuracy} = \frac{TP + TN}{N} = \frac{3}{6}$$

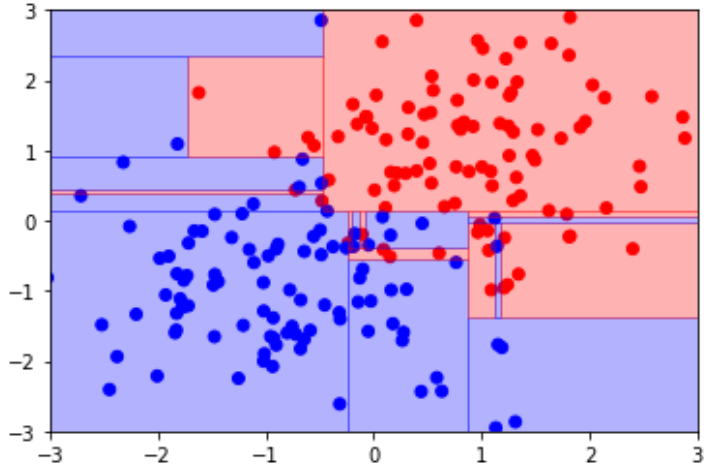
$$\text{Recall} = \frac{TP}{TP + FN} = \frac{2}{3}$$

Outline

The Supervised Decision Tree

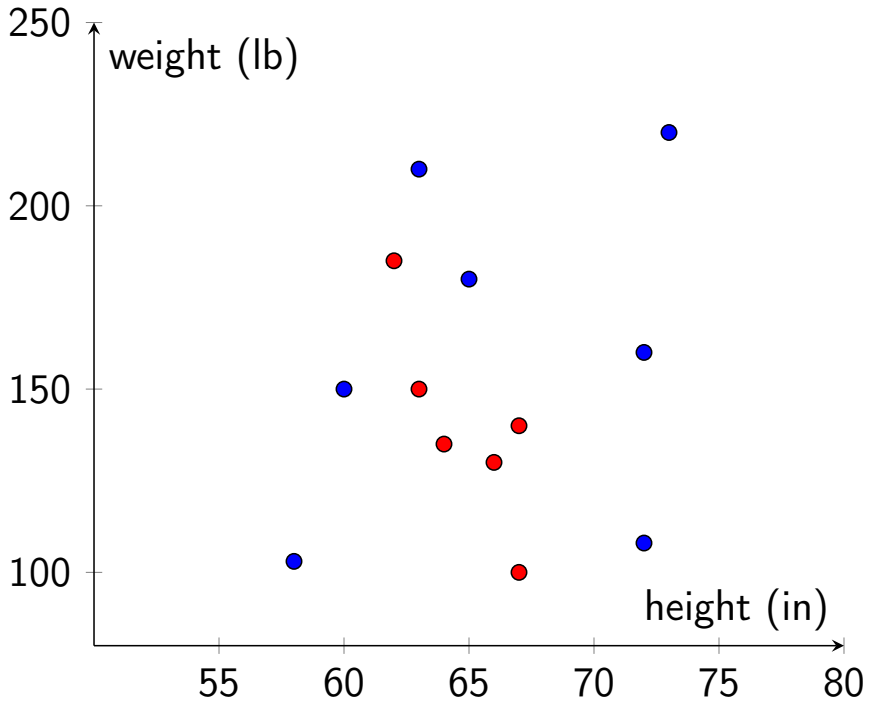
- Decision Trees are a class of discriminative learning algorithm searching the hypothesis space of hyper-rectangles.
- Given a labeled training sample X, a Decision Tree produces a model h mapping partitioned hyper-rectangles in $\mathbb{R}^d$ to target classes.





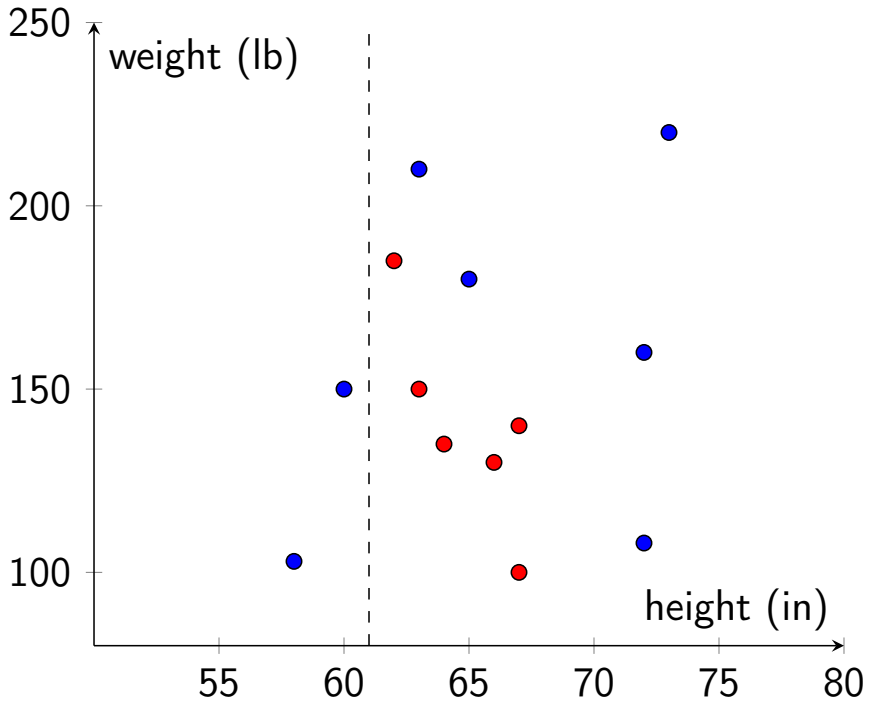
Decision Tree Induction

- 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.



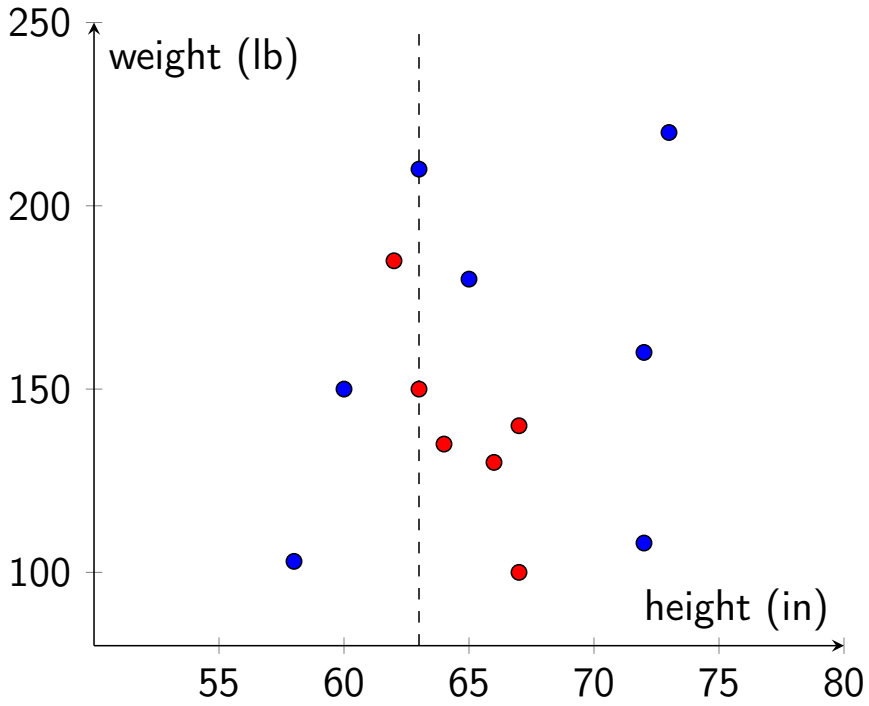
Decision Tree Induction

- 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.



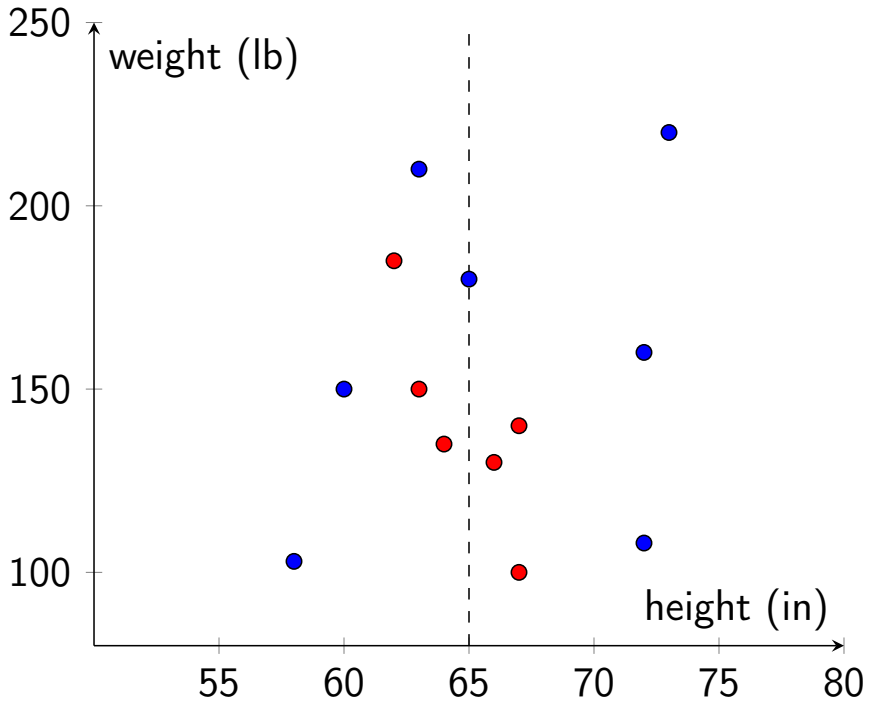
Decision Tree Induction

- 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.



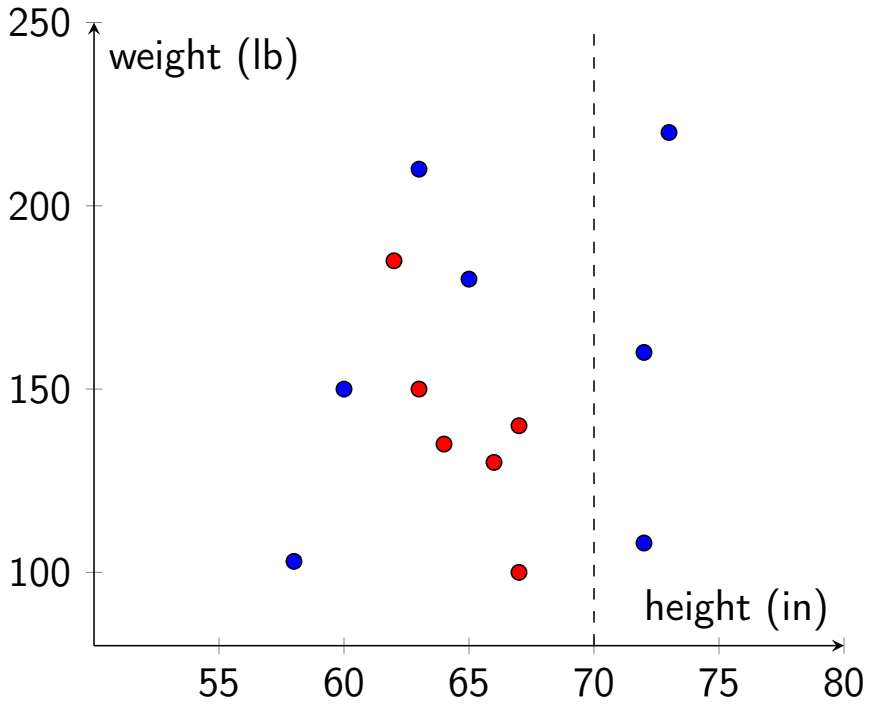
Decision Tree Induction

- 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.



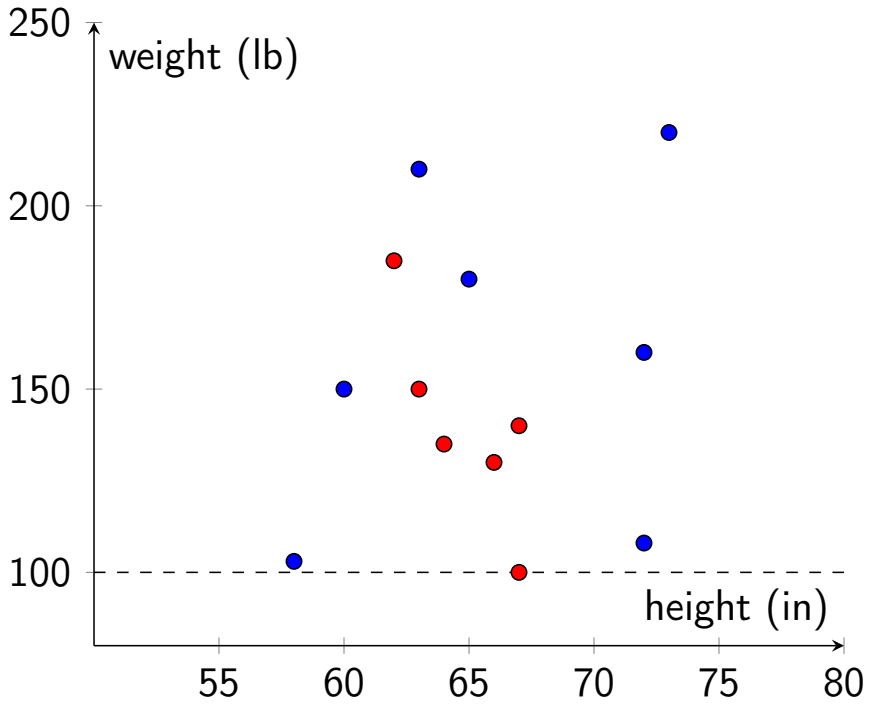
Decision Tree Induction

- 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.



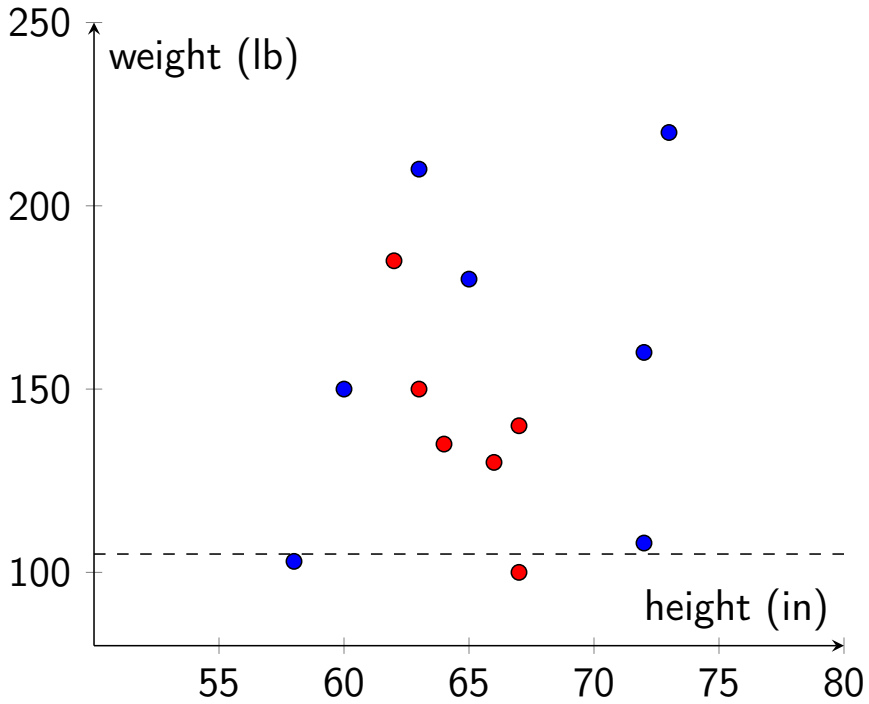
Decision Tree Induction

- 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.



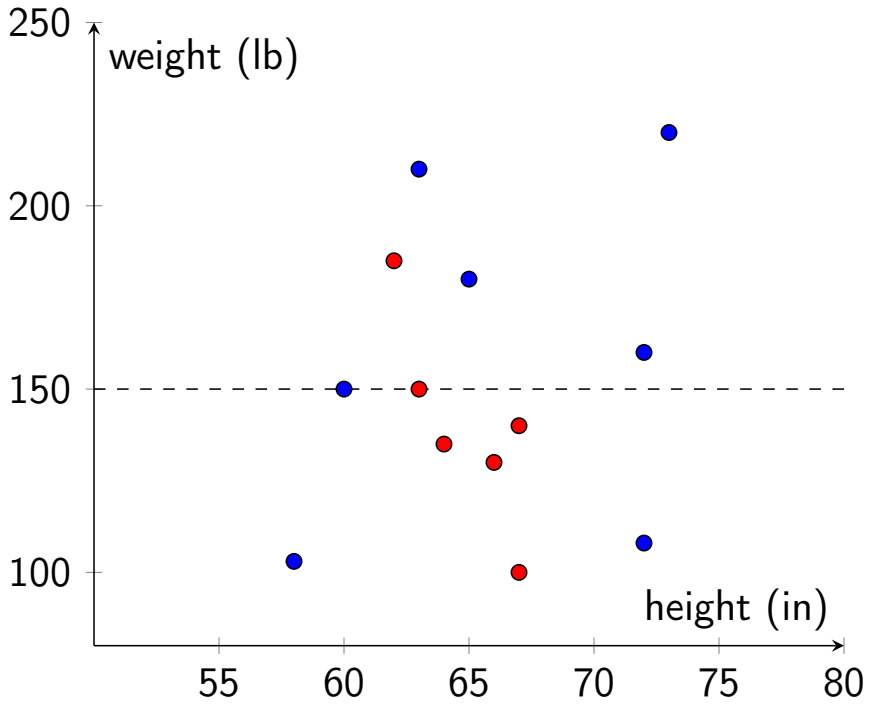
Decision Tree Induction

- 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.



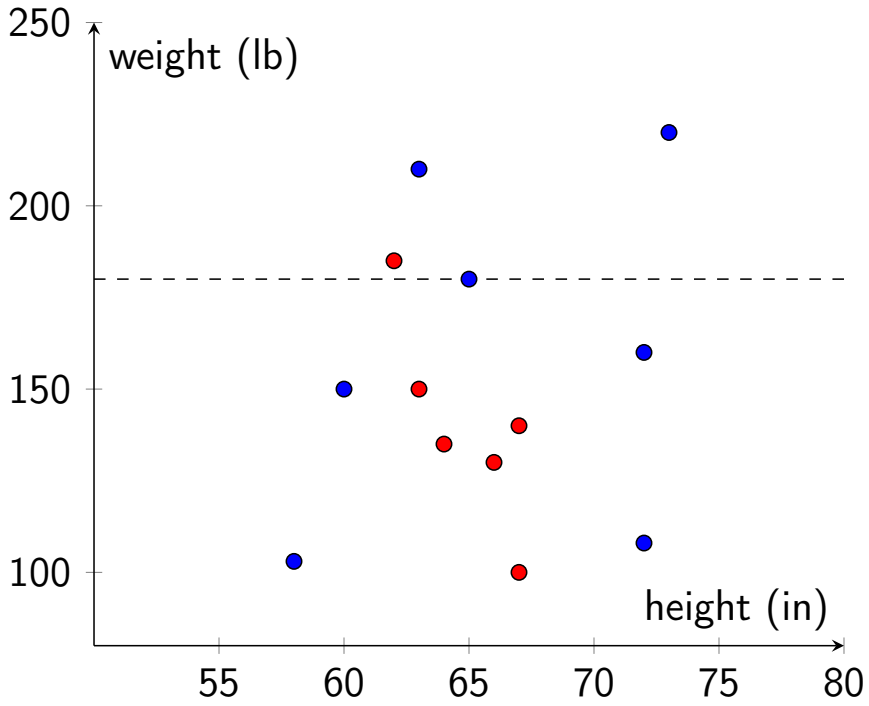
Decision Tree Induction

- 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.



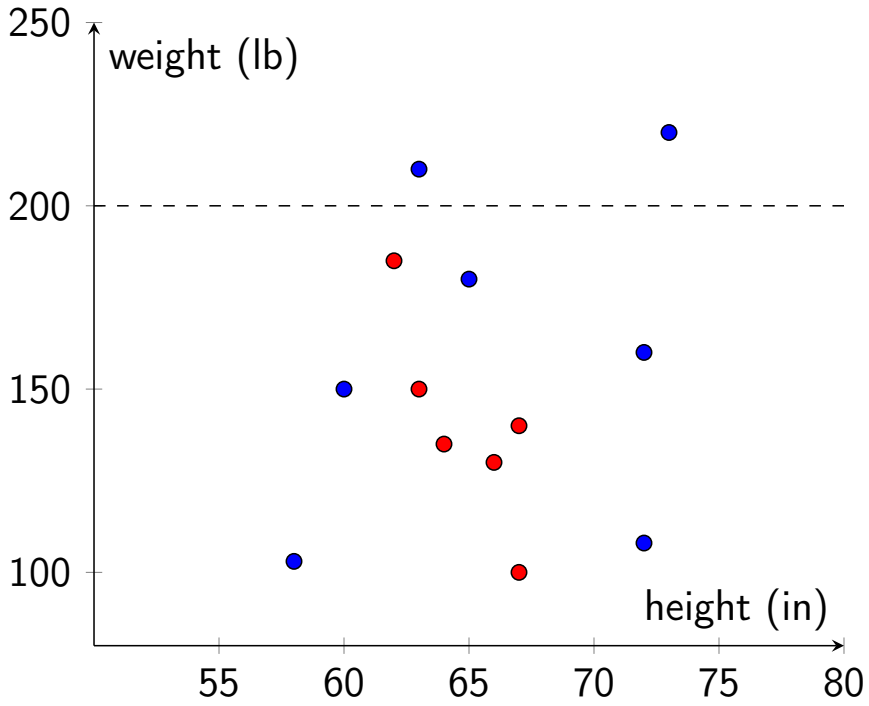
Decision Tree Induction

- 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.



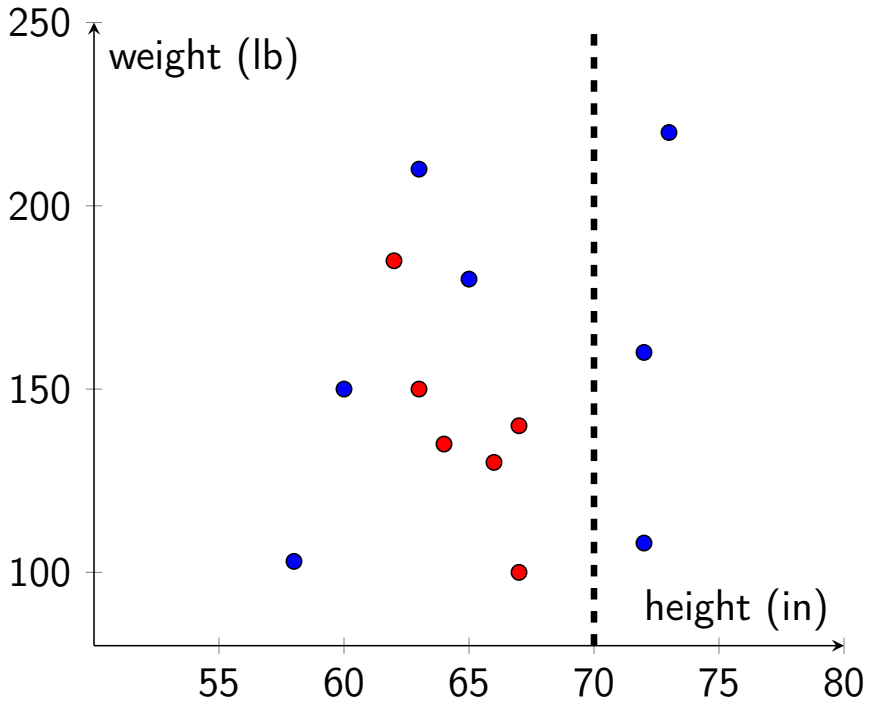
Decision Tree Induction

- 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.



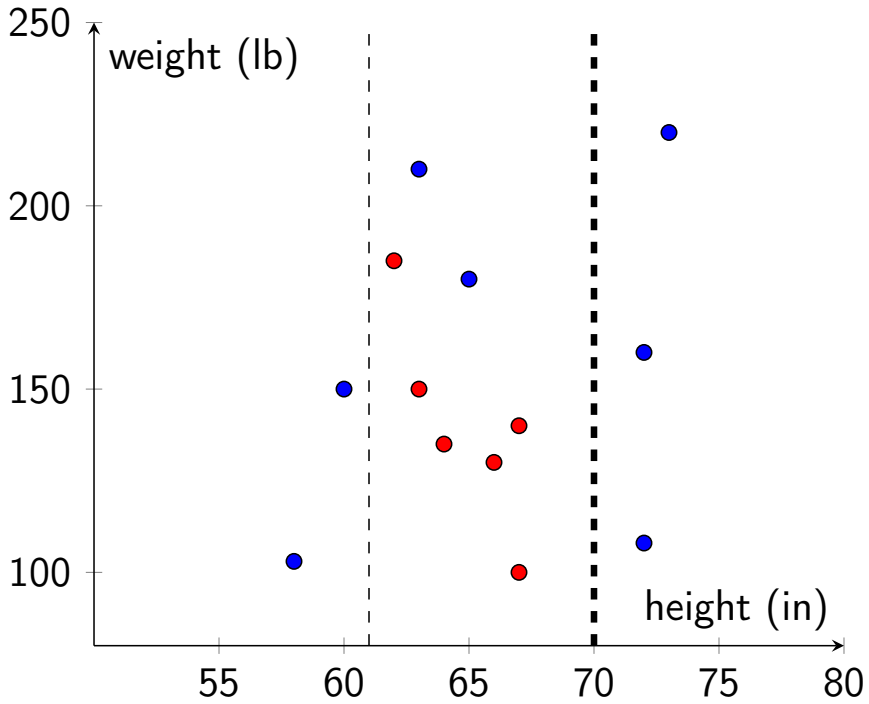
Decision Tree Induction

- 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.



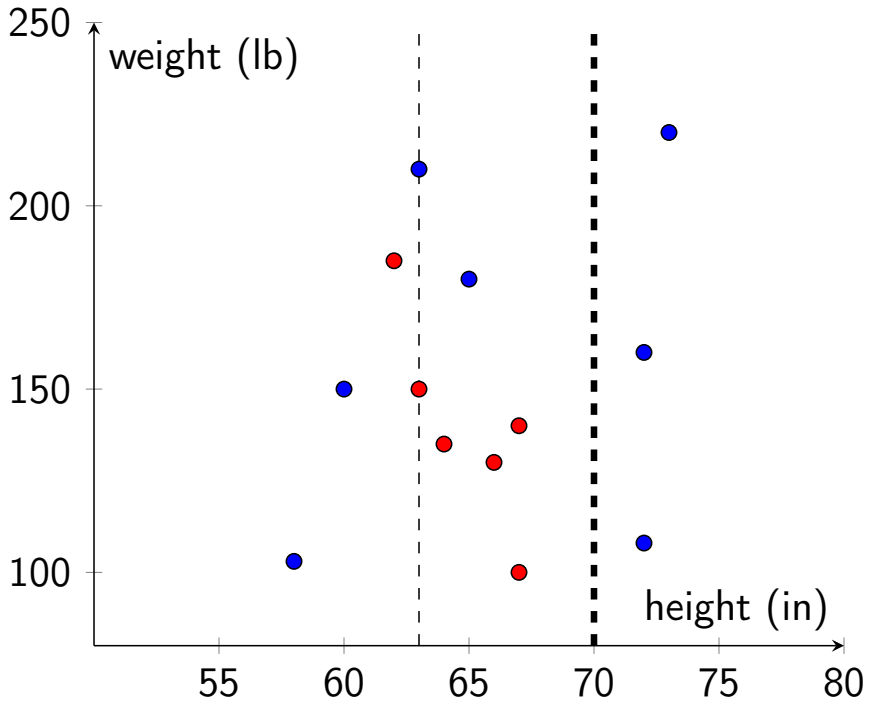
Decision Tree Induction

- 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.



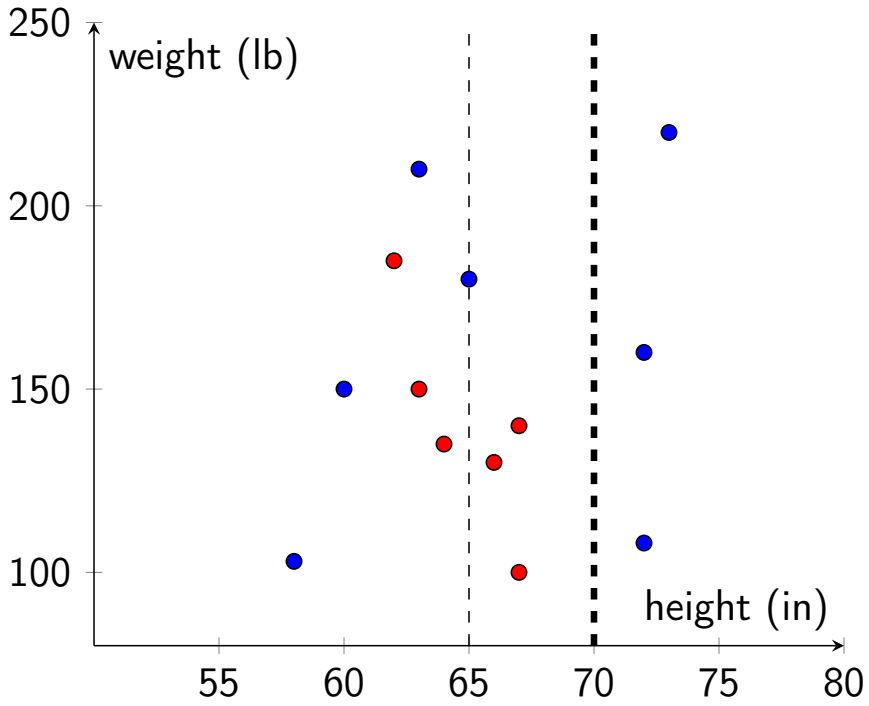
Decision Tree Induction

- 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.



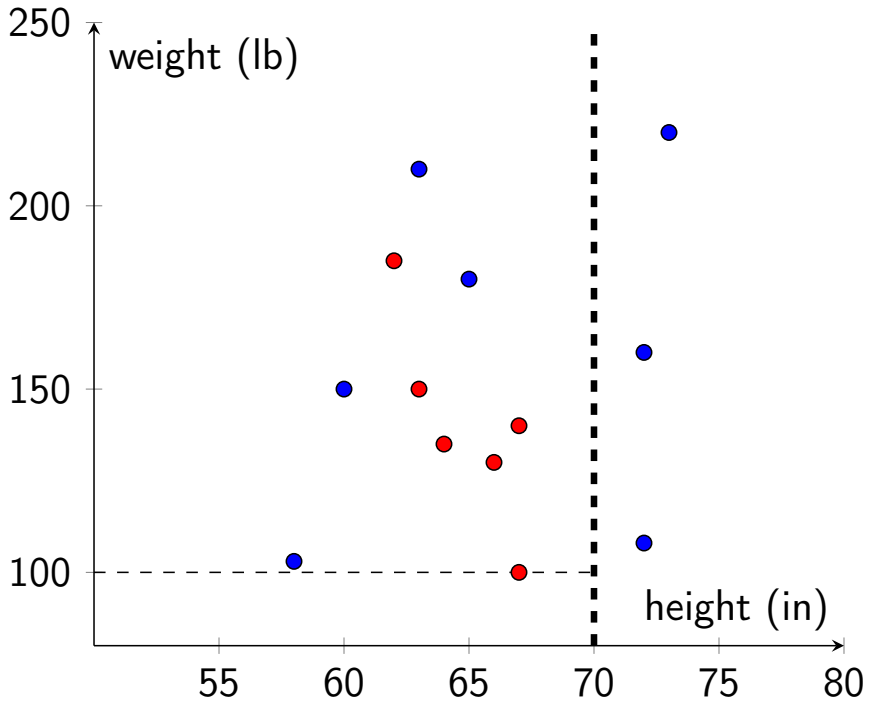
Decision Tree Induction

- 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.



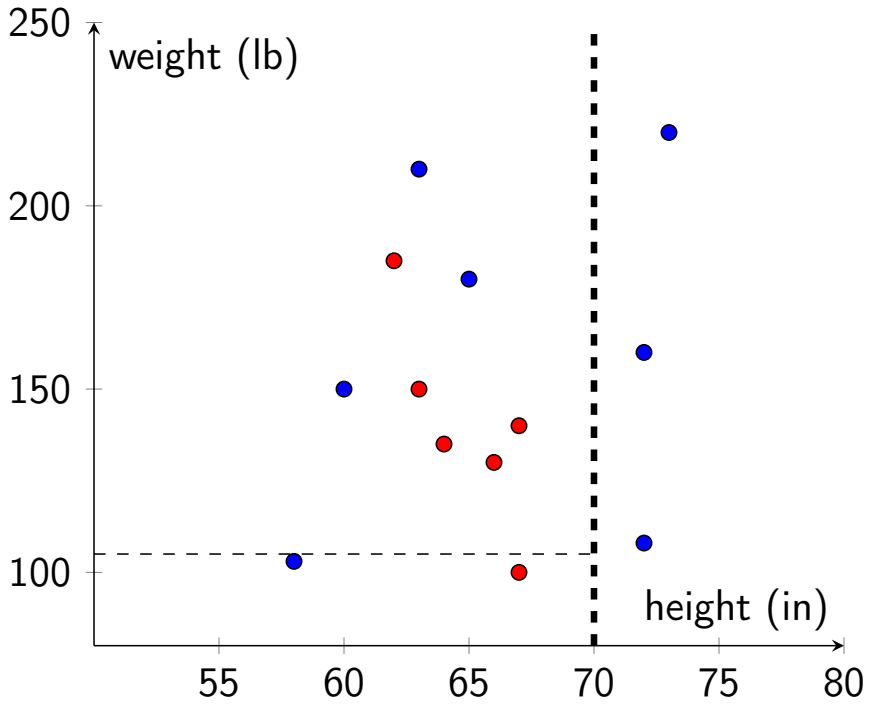
Decision Tree Induction

- 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.



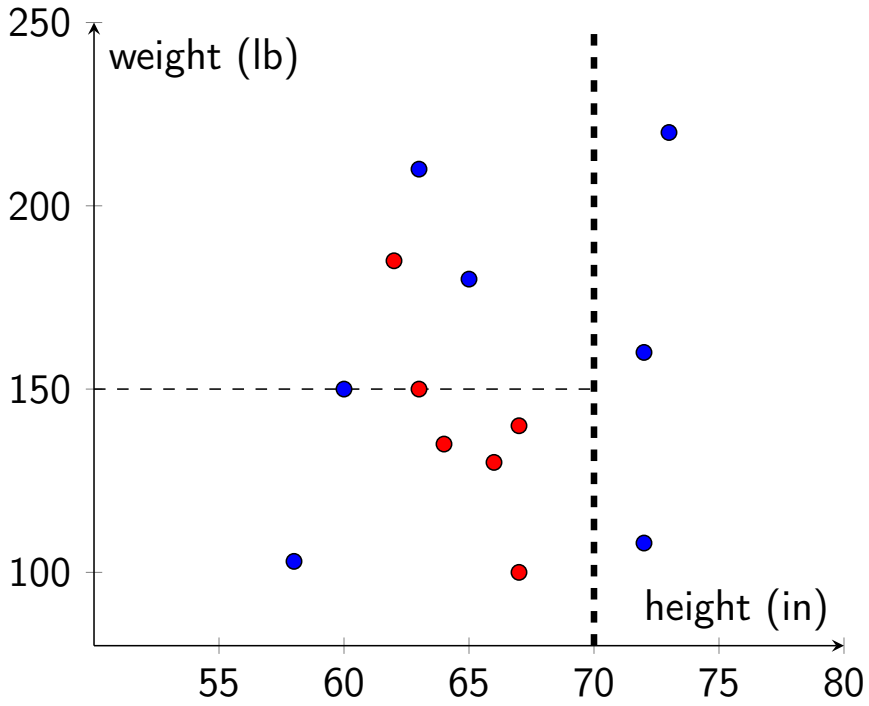
Decision Tree Induction

- 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.



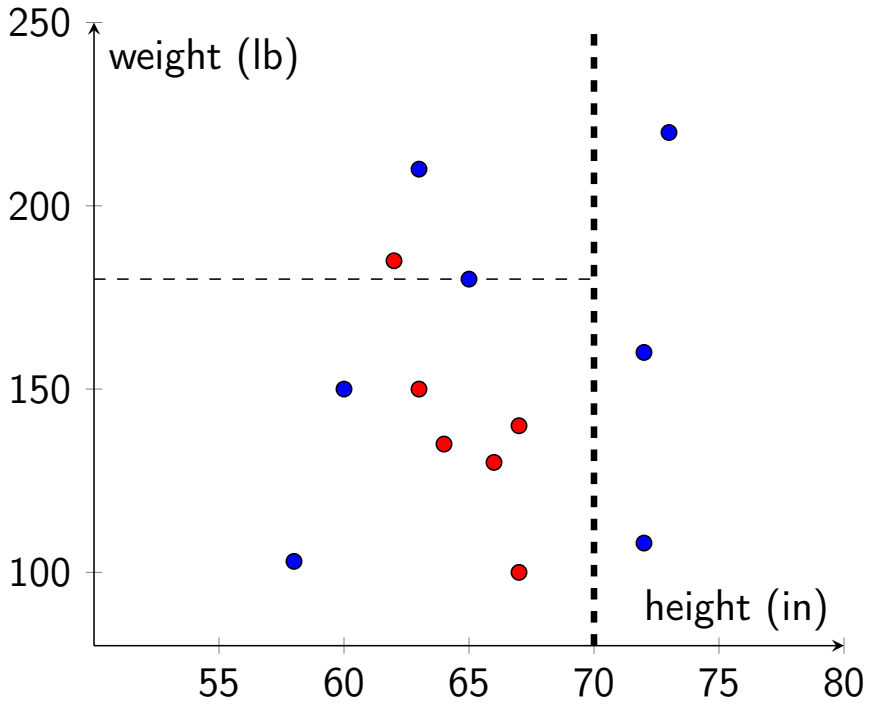
Decision Tree Induction

- 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.



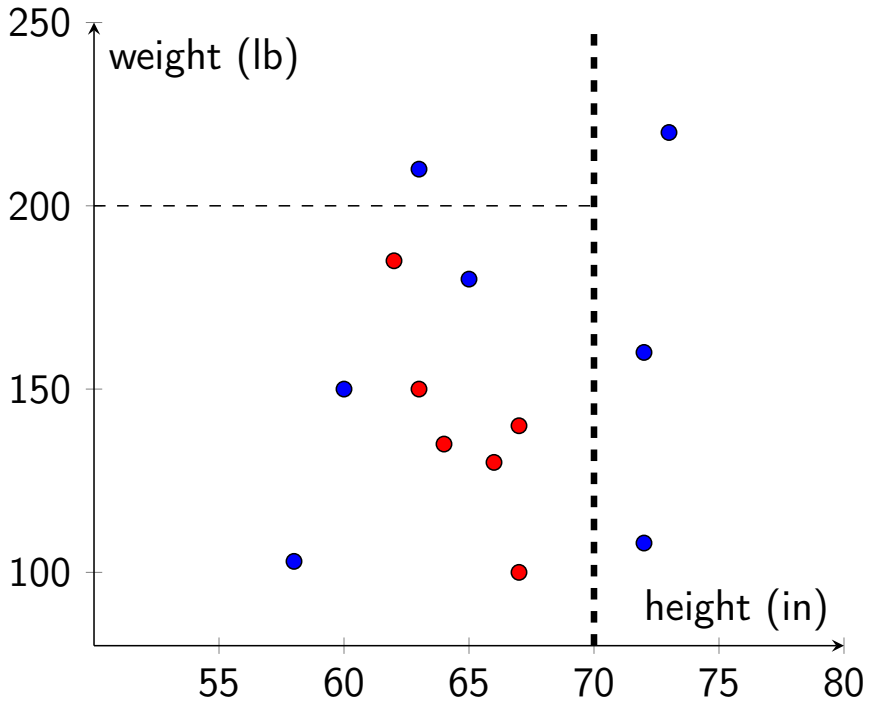
Decision Tree Induction

- 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.



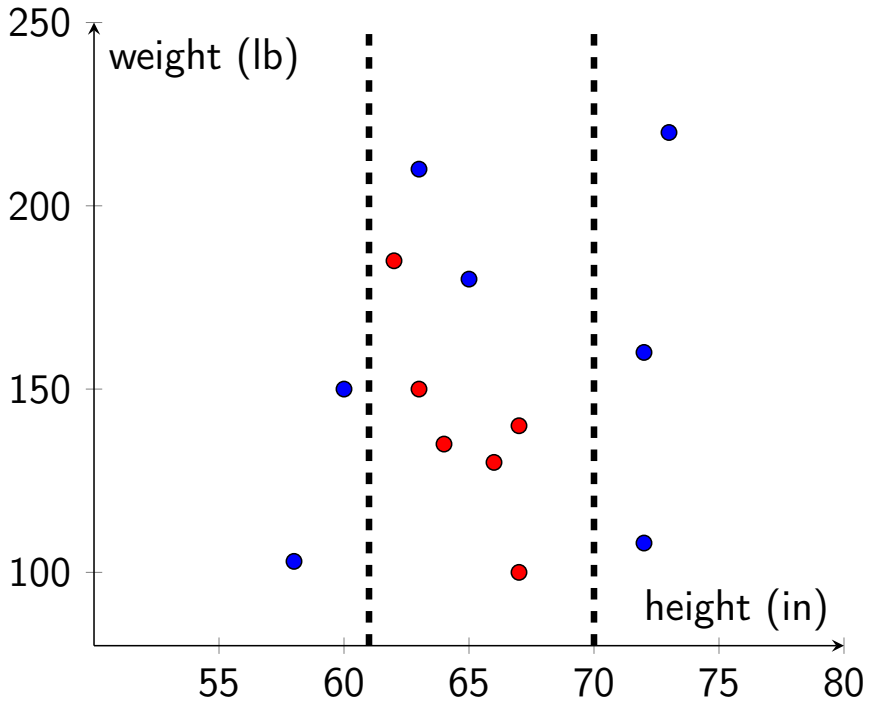
Decision Tree Induction

- 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.



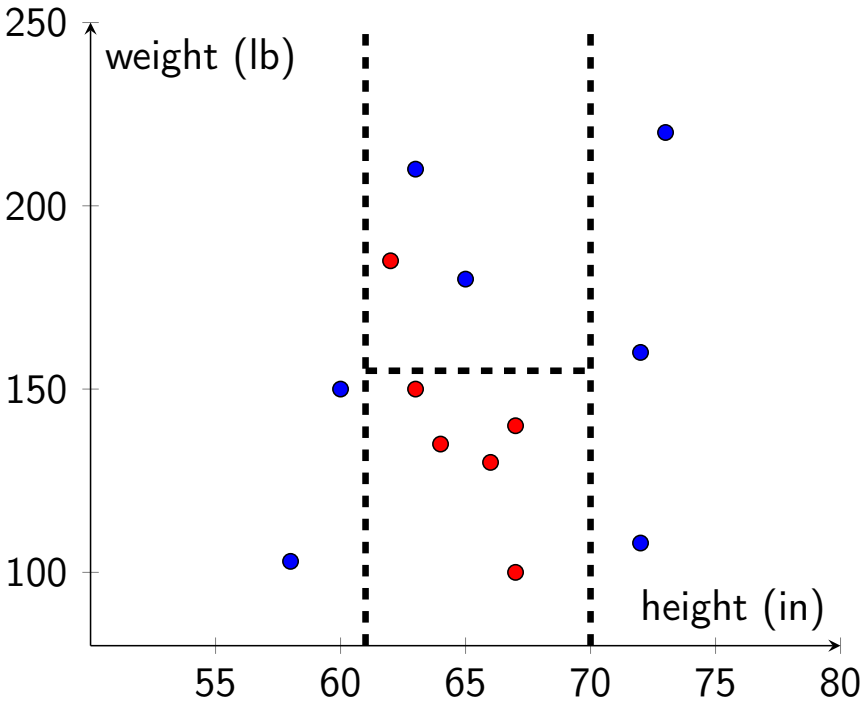
Decision Tree Induction

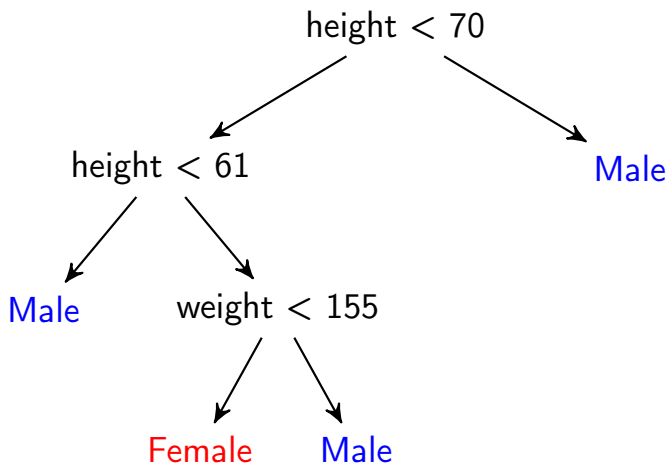
- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.



Decision Tree Induction

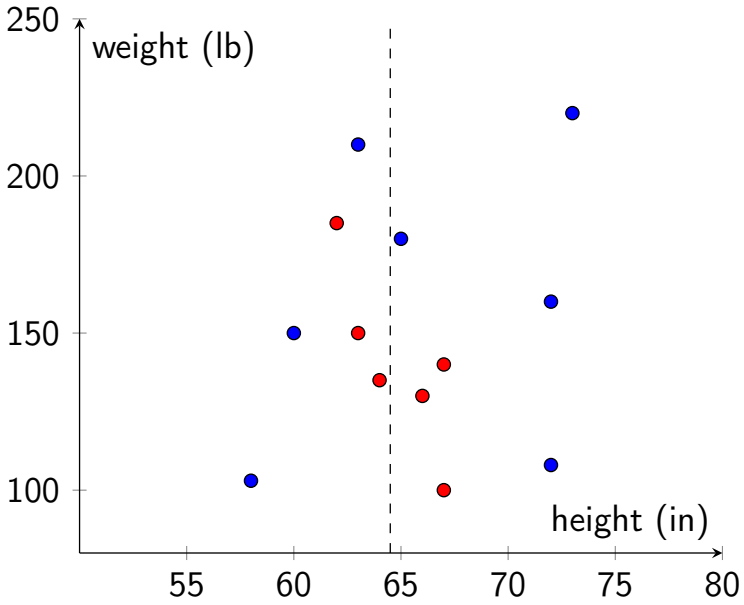
- Decision Trees build a model by recursively applying a simple heuristic: select a cut-point on the attribute that maximizes the class purity in the regions resulting from the cut-points partitioning.





The Impurity Criterion: A Heuristic

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

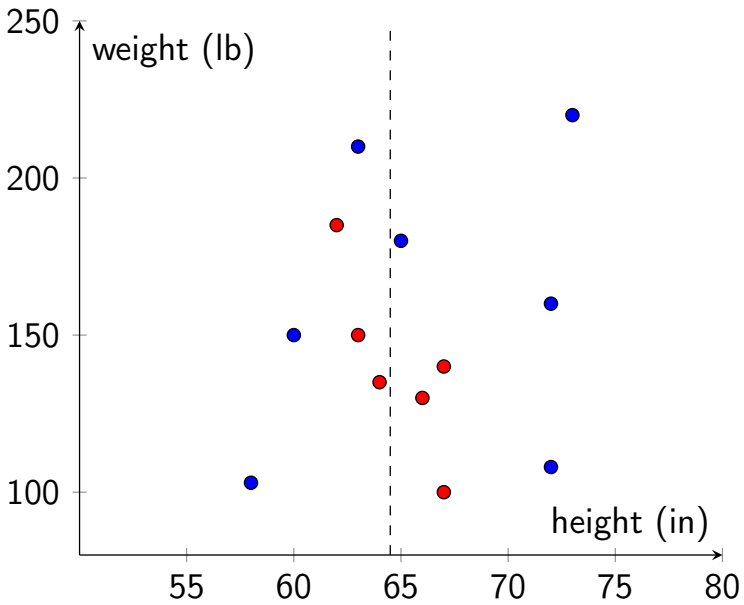


The Impurity Criterion: A Heuristic

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

$$P_{\text{L,male}} = \frac{3}{6} \qquad P_{\text{R,male}} = \frac{4}{7}$$

$$P_{\text{L,male}} = \frac{3}{6} \qquad P_{\text{R,female}} = \frac{3}{7}$$

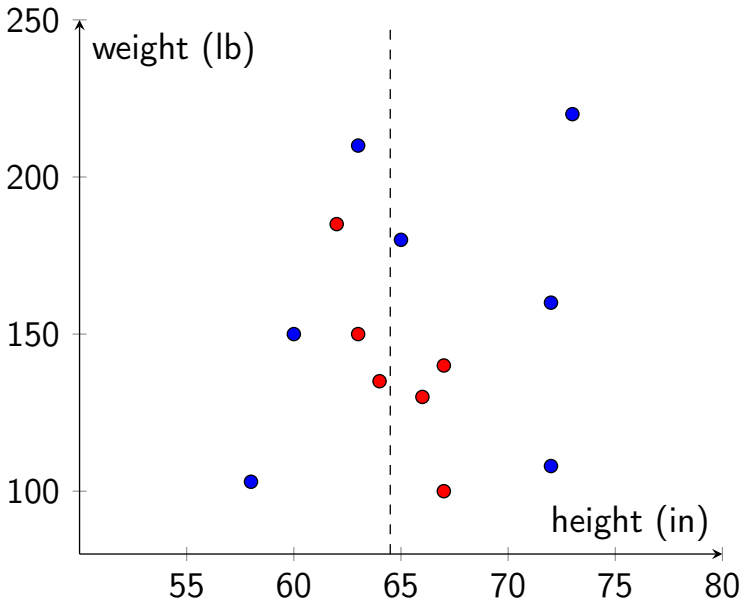


The Impurity Criterion: A Heuristic

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

$$P_{\text{L,male}} = \frac{3}{6} \qquad P_{\text{R,male}} = \frac{4}{7}$$

$$P_{\text{L,male}} = \frac{3}{6} \qquad P_{\text{R,female}} = \frac{3}{7}$$



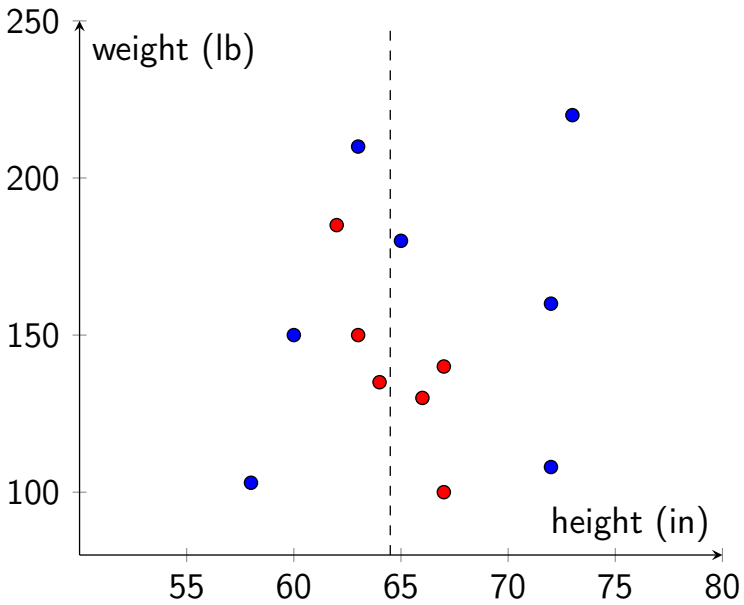
$$\text{Gini}_R = 1 - \sum_{p_c \in P_R} p_c^2$$

The Impurity Criterion: A Heuristic

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

$$P_{\text{L,male}} = \frac{3}{6} \qquad P_{\text{R,male}} = \frac{4}{7}$$

$$P_{\text{L,male}} = \frac{3}{6} \qquad P_{\text{R,female}} = \frac{3}{7}$$



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

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

Conclusion

Thank you for your attention!
Questions?

https://bit.ly/2tPVPfv

aymulyar@vcu.edu
www.andriymulyar.com