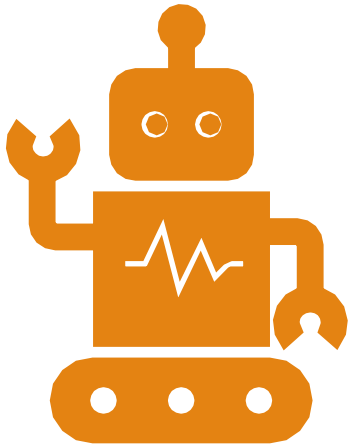


CS341
Artificial Intelligence

Lecture 9

DR. HEBA MOHSEN

Machine Learning

- An agent is learning if it improves its performance on future tasks after making observations about the world.
- Learning is important for practical applications of artificial intelligence.
- Learning agents begin with a minimal amount of knowledge and learn from examples, high-level advice, or their own explorations of the domain.
- Learning involves generalization from experience. Because interesting domains tend to be large, a learner may only examine a fraction of all possible examples; from this limited experience, the learner must acquire knowledge that will generalize correctly to unseen instances of the domain.

Machine Learning

- Any component of an agent can be improved by learning from data.
- The improvements, and the techniques used to make them, depend on three major factors:
  - What *prior knowledge* the agent already has.
  - What *representation* is used for the data and the component.
  - What *feedback* is available to learn from.

Representation and prior knowledge

- Representation for the components of a logical agent were such as: propositional and first-order logical sentences.
- Other type of representation is Factored Representation where the *input* is represented using vector of attribute values and *output* can be either a continuous numerical value or a discrete.
- We will be focusing on the Machine Learning Algorithms that deal with the factored representation.

Feedback for learning

Types of feedback for the main types of learning:

Supervised Learning: The agent observes some example input–output pairs and learns a function that maps from input to output.

Unsupervised Learning: The agent learns patterns in the input even though no explicit feedback is supplied.

Reinforcement Learning: The agent learns from a series of reinforcements (rewards or punishments).

Supervised Learning

- A supervised learning algorithm analyzes the training data and produces an inferred function, which can be used for mapping new examples.
- An optimal scenario will allow for the algorithm to correctly determine the class labels for unseen instances. This requires the learning algorithm to generalize from the training data to unseen situations in a "reasonable" way.
- **Supervised learning algorithms:**
  - ID3 Decision Tree
  - K-Nearest Neighbor (K-NN)

ID3 Decision Tree Induction Algorithm

Inductive Learning is learning a general function or rule from specific input–output pairs.

Decision tree induction is one of the simplest and yet most successful forms of supervised machine learning.

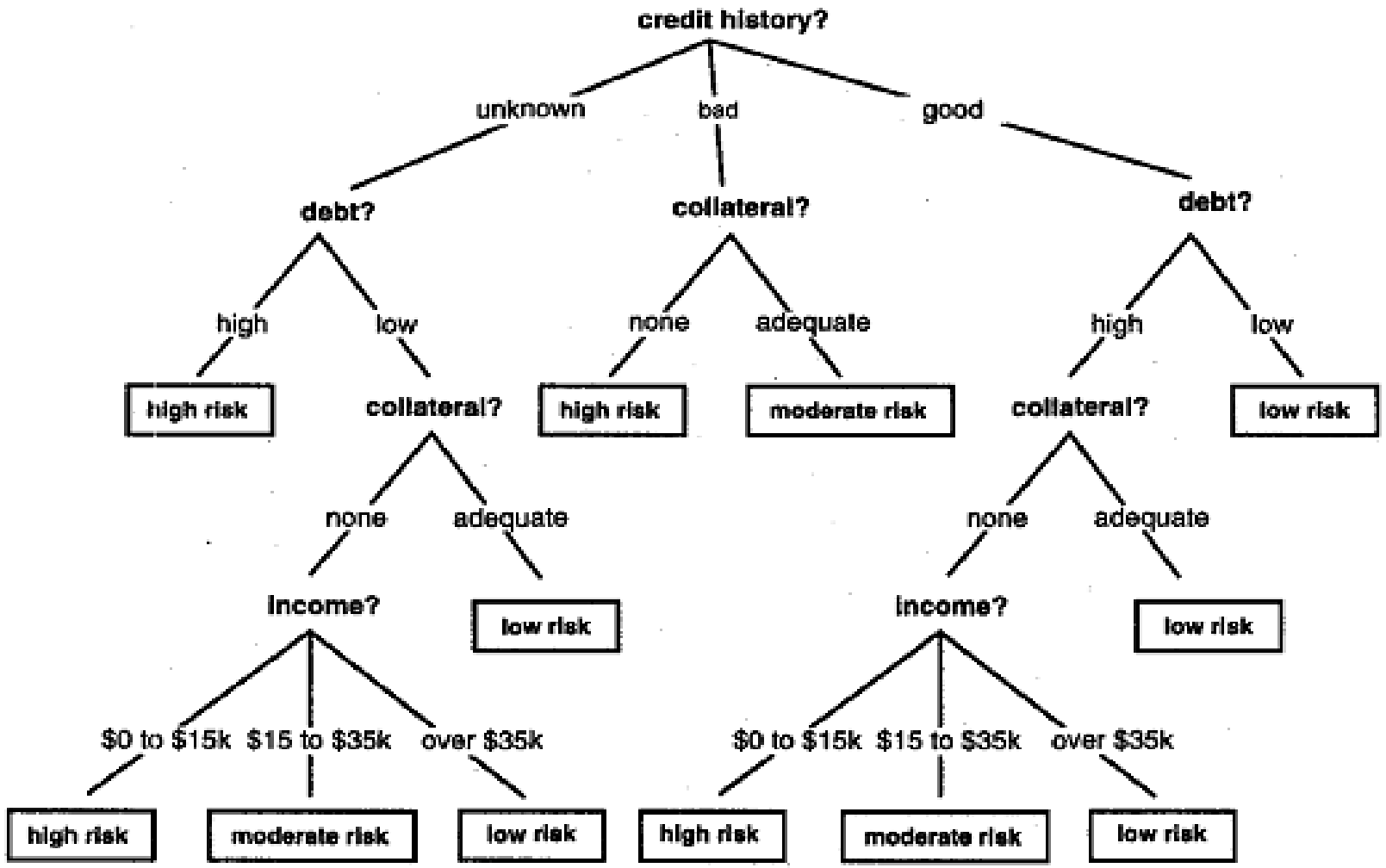
ID3 induces concepts from examples.

ID3 represents concepts as decision trees, a representation that allows us to determine the classification of an object by testing its values for certain properties.

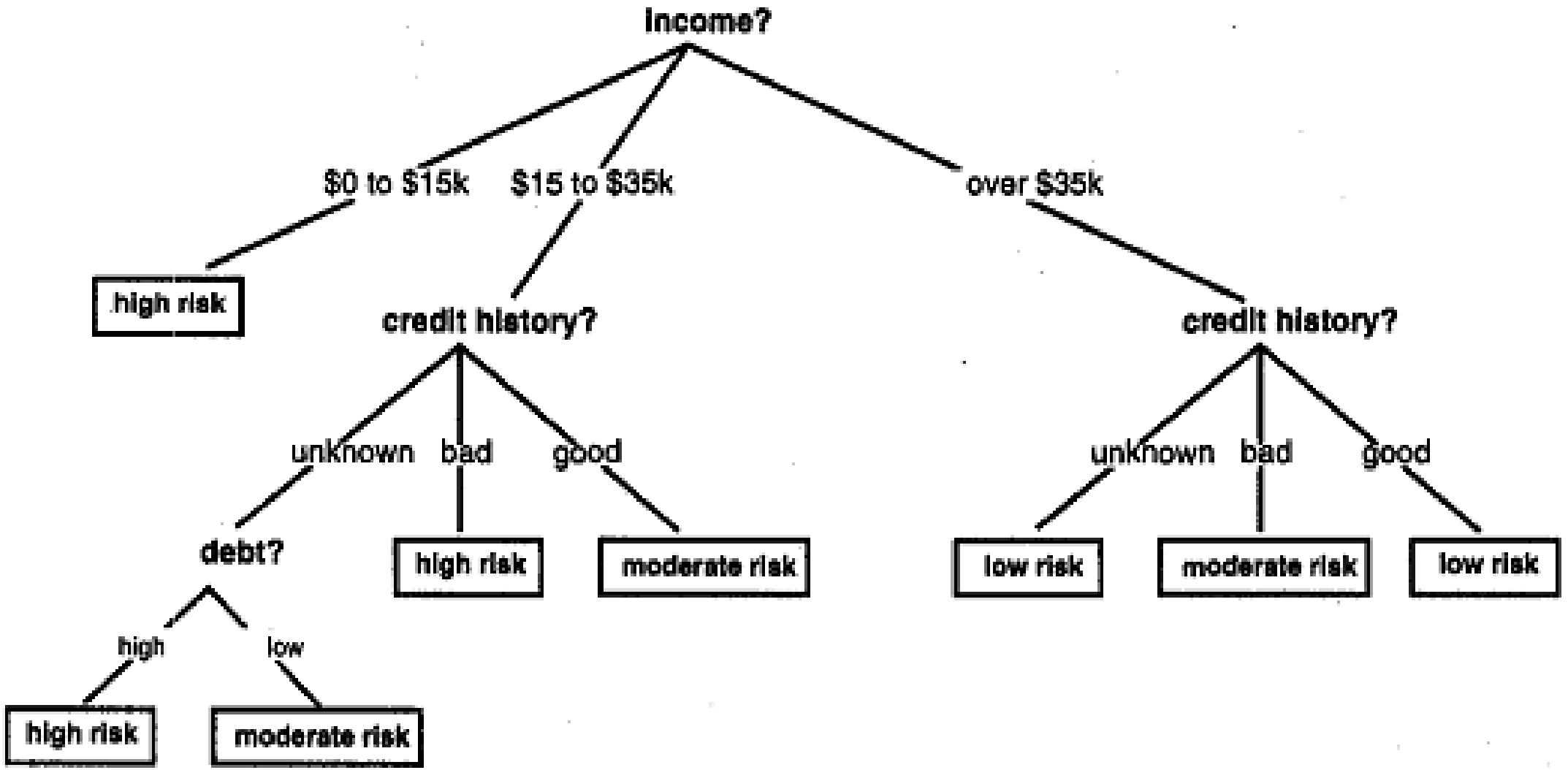
For example, consider the problem of estimating an individual's credit risk on the basis of such properties as credit history, current debt, collateral, and income.

RISK	CREDIT HISTORY	DEBT	COLLATERAL	INCOME
high	bad	high	none	$0 to $15k
high	unknown	high	none	$15 to $35k
moderate	unknown	low	none	$15 to $35k
high	unknown	low	none	$0 to $15k
low	unknown	low	none	over $35k
low	unknown	low	adequate	over $35k
high	bad	low	none	$0 to $15k
moderate	bad	low	adequate	over $35k
low	good	low	none	over $35k
low	good	high	adequate	over $35k
high	good	high	none	$0 to $15k
moderate	good	high	none	$15 to $35k
low	good	high	none	over $35k
high	bad	high	none	$15 to $35k

The decision tree represents the classifications, in that this **tree** can correctly classify all the objects in the table. In a decision tree, each internal node represents a test on some property, such as credit history or debt; each possible value of that property corresponds to a branch of the tree. Leaf nodes represent classifications, such as low or moderate risk.



In general, the **size of the tree** necessary to classify a given set of examples varies according to the order with which properties are tested.



ID3 algorithm

- Given a set of training instances and a number of different decision trees that correctly classify them, we may ask which tree has the greatest likelihood of correctly classifying unseen instances of the population.
- The ID3 algorithm assumes that this is the simplest decision tree that covers all the training examples.
- The rationale for this assumption is the time-honored heuristic of preferring simplicity and avoiding unnecessary assumptions.

Top-Down Decision Tree Induction

- ID3 constructs decision trees in a top-down fashion.
- ID3 selects a property to test at the current node of the tree and uses this test to partition the set of examples; the algorithm then recursively constructs a subtree for each partition.
- This continues until all members of the partition are in the same class; that class becomes a leaf node of the tree.
- Because the order of tests is critical to constructing a simple decision tree, ID3 relies heavily on its criteria for selecting the test at the root of each subtree.
- ID3 adds a subtree to the current tree and continues its search; it does not backtrack.

How Does a Decision Tree Select Splits?

- The ID3 decision makes use of two concepts when creating a tree from top-down:
  1. **Entropy**
  2. **Information Gain** (as referred to as just **gain**)

- Using these two concepts, the nodes to be created and the attributes to split on can be determined.

- **Gain** measures how well a given attribute separates training examples into targeted classes. The one with **the highest information** (information being the most useful for classification) is selected.

- In order to define gain, we first borrow an idea from information theory called entropy. **Entropy** measures the amount of information in an attribute.

Entropy

Given a dataset S

Entropy(S) = $-\sum_{x \in X} p(x) \log_2 p(x)$

where

X = Set of classes in **S**
P(x) = The proportion of the number of elements in class **x** to the number of elements in set **S**

Note that S is not an attribute but the entire sample set.

Notice **entropy is 0 if all members of S belong to the same class**.

Informational Gain

- Information Gain (also known as just Gain) uses the entropy in order to determine what attribute is best used to create a split with.

- The column with the higher Gain will be used as the node of the decision tree.

$$Gain(S, A) = Entropy(S) - \Sigma((|Sv| / |S|) * Entropy(Sv))$$

Where:

- S is each value v of all possible values of attribute A
- Sv = **subset of S for which attribute A has value v**
- $|Sv|$ = **number of elements in Sv**
- $|S|$ = **number of elements in S**

Example: Data Set For Heart Disease Risk

Age	Trestbps	Chol	Gender	Heart
<50	<120	<200	male	No
<50	<120	<200	female	No
<70	<120	<200	male	Yes
<60	<140	<200	male	Yes
<60	<160	>200	male	Yes
<60	<160	>200	female	No
<70	<160	>200	female	Yes
<50	<140	<200	male	No
<50	<160	>200	male	Yes
<60	<140	>200	male	Yes
<50	<140	>200	female	Yes
<70	<140	<200	female	Yes
<70	<120	>200	male	Yes
<60	<140	<200	female	No

trestbps: resting blood pressure **chol**: serum cholestoral in mg/dl

We determine the number of "No" and number of "Yes" for the decision column in order to calculate the **entropy**

Age	Trestbps	Chol	Gender	Heart
<50	<120	<200	male	No
<50	<120	<200	female	No
<60	<160	>200	female	No
<50	<140	<200	male	No
<60	<140	<200	female	No
Total No				**5**
<70	<120	<200	male	Yes
<60	<140	<200	male	Yes
<60	<160	>200	male	Yes
<70	<160	>200	female	Yes
<50	<160	>200	male	Yes
<60	<140	>200	male	Yes
<50	<140	>200	female	Yes
<70	<140	<200	female	Yes
<70	<120	>200	male	Yes
Total Yes				**9**

- The following formula can be used to calculate the **total entropy**, $E = -\sum_{x \in X} p(x) \log_2 p(x)$

E = ((-NumNo/NumTotal)log2(NumNo/NumTotal)) +

((-NumYes/NumTotal)log2(NumYes/NumTotal))

- So, in this example, the total entropy would be:

E = ((-5/14)log2(5/14)) + ((-9/14)log2(9/14)) = 0.94

- Next, **we take each column and calculate the Gain**.

  Starting with Gender, we look at each of the columns with Male and Female and calculate entropy of each "Yes" and "No" for Gender:Female (6/14) and Gender:Male (8/14) and subtract from the total entropy that we calculated previously.

Gender	Heart
male	No
male	Yes
male	Yes
male	Yes
male	No
male	Yes
male	Yes
male	Yes
Total	**8**

Gender	Heart
female	No
female	No
female	Yes
female	Yes
female	Yes
female	No
Total	**6**

Gain(S, A) = Entropy (S) - Σ ((|Sv| / |S|) * Entropy (Sv))

Gain(S, Gender) = TotalEntropy – (6/14 x (EntropyFemale)) – (8/14 x (EntropyMale))

$$= 0.94 - \frac{6}{14}\left(-\frac{3}{6}\log_2\left(\frac{3}{6}\right) - \frac{3}{6}\log_2\left(\frac{3}{6}\right)\right) - \frac{8}{14}\left(-\frac{6}{8}\log_2\left(\frac{6}{8}\right) - \frac{2}{8}\log_2\left(\frac{2}{8}\right)\right)$$

= 0.048

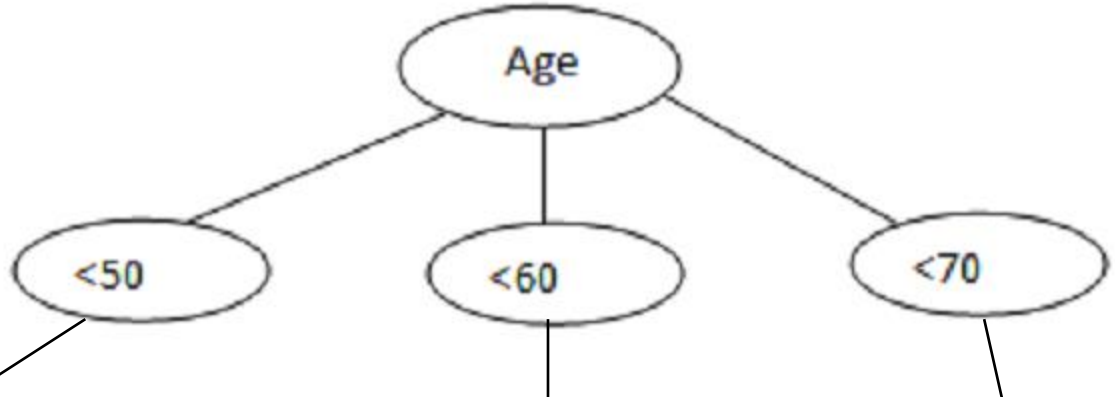
- **After calculating the Gain for each column**, we have determined that **Age** has the highest gain as it is the largest value, so this becomes the next attribute node.

- In this initial case, **Age will be the root node** with each of its possible values (<50, <60, and <70) becoming children.

- Now, the next set of nodes or children need to be determined using a recursive process. **Each of the subsets of age is looked at.**
  - For < 50

Age	Trestbps	Chol	Gender	Heart
<50	<120	<200	male	No
<50	<120	<200	female	No
<50	<140	<200	male	No
<50	<160	>200	male	Yes
<50	<140	>200	female	Yes

  - …
  - …

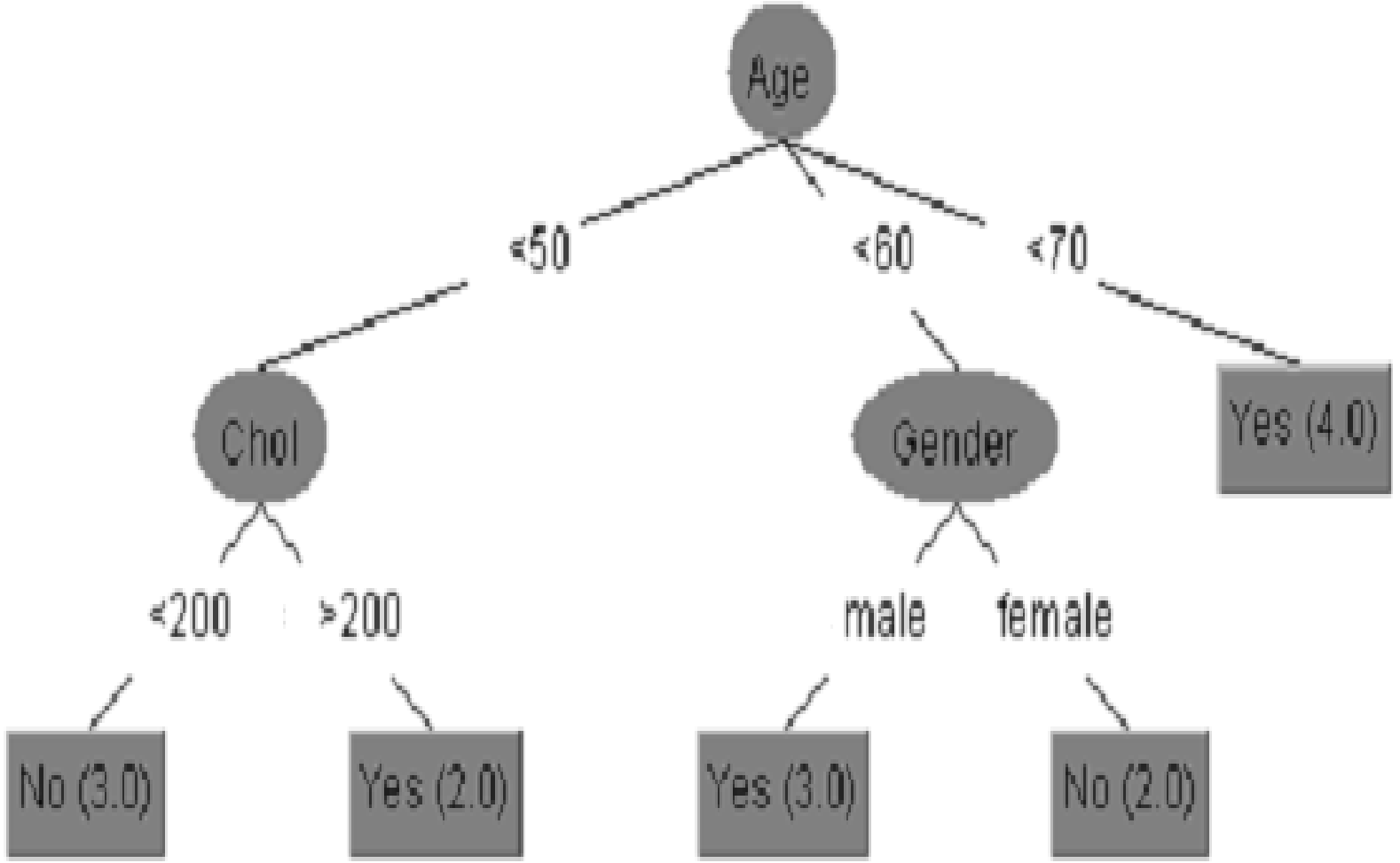


Trestbps	Chol	Gender	Heart
<120	<200	male	No
<120	<200	female	No
<140	<200	male	No
<160	>200	male	Yes
<140	>200	female	Yes

Trestbps	Chol	Gender	Heart
<140	<200	male	Yes
<160	>200	male	Yes
<160	>200	female	No
<140	>200	male	Yes
<140	<200	female	No

Trestbps	Chol	Gender	Heart
<120	<200	male	Yes
<160	>200	female	Yes
<140	<200	female	Yes
<120	>200	male	Yes

- This process would continue recursively with all subsets until all nodes have the appropriate attribute and all leafs contain a decision.



Finally, after the decision tree is completed, a set of rules can be generated for determining a decision.

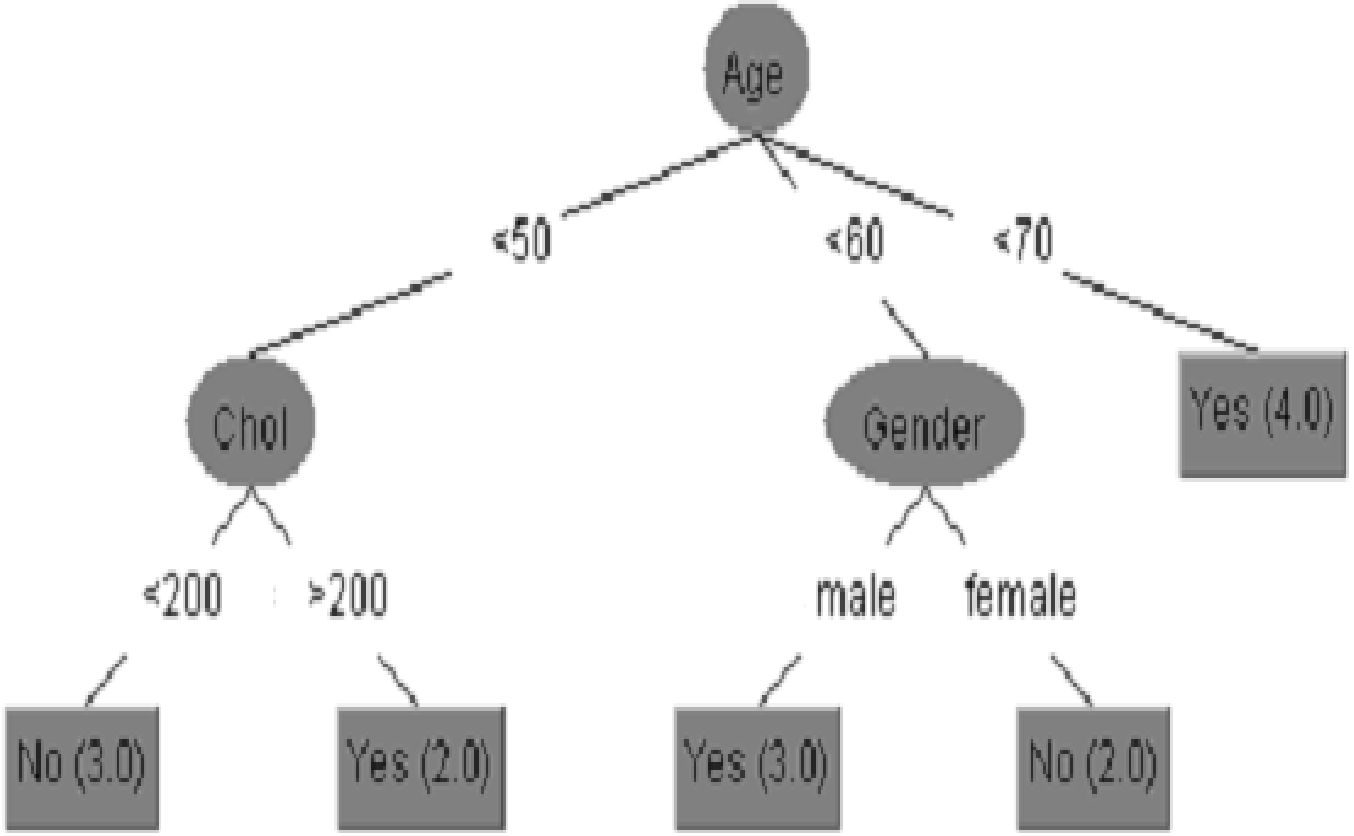
R1: IF (Age < 50) AND (Chol <200) THEN Risk=NO

R2: IF (Age < 50) AND (Chol >200) THEN Risk=YES

R3: IF (Age < 60) AND (Gender=male) THEN Risk=YES

R4: IF (Age < 60) AND (Gender=female) THEN Risk=NO

R5: IF (Age < 70) THEN Risk=YES



Exercise

Apply the ID3 algorithm to the following table which informs about decision making factors to play tennis at outside for previous 14 days.

Outlook	Temperature	Humidity	Wind	Play Tennis
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No