

Machine Learning (ML)

N. Vassilas
Department of Computer Engineering
University of West Attica

What is Machine Learning?

- ML is the subfield of AI that is concerned with the design and development of algorithms that allow computers (machines) to improve their performance over time (i.e. to learn) based on data, such as from sensor data or databases. (*From Wikipedia*)
- Ability of a machine to improve its own performance through the use of a software that employs artificial intelligence techniques to mimic the ways by which humans seem to learn, such as repetition and experience. (*From Business Dictionary*)
- The ability to recognize patterns that have occurred repeatedly and improve its performance based on past experience.

ML and Pattern Recognition (PR)

The field of PR is closely related to Machine Learning as it is concerned with the automatic discovery of regularities in data through the use of computer algorithms. These regularities are then used in order to take actions such as classifying the data into different categories.

See book:

**C.N. Bishop, Pattern Recognition and Machine Learning,
Springer, 2007.**

How Do We Learn?

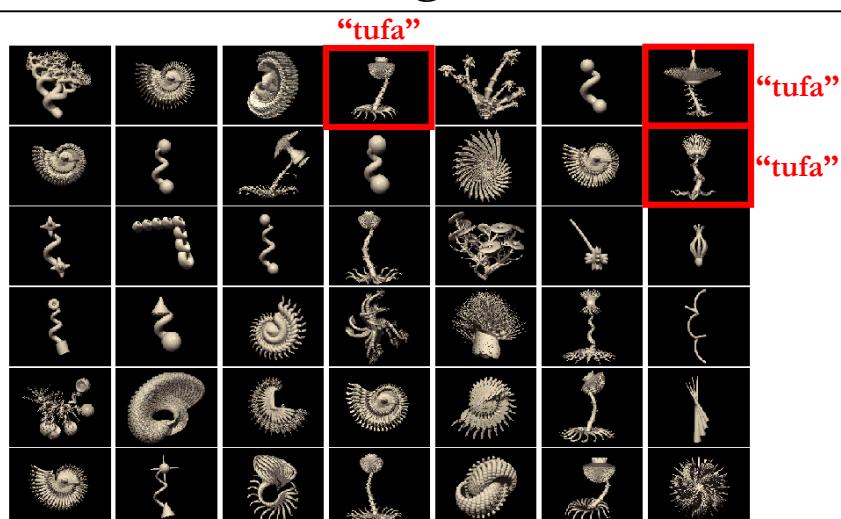
Human	Machine
Memorize	k-Nearest Neighbors
Observe someone else, then repeat	Supervised Learning, Learning by Demonstration
Keep trying until it works (riding a bike)	Reinforcement Learning
20 Questions	Decision Tree
Pattern matching (faces, voices, languages)	Supervised Learning, Unsupervised Learning
Extrapolate current trend (stock market, house prices)	Regression

Inductive Learning (from J. Tenenbaum)



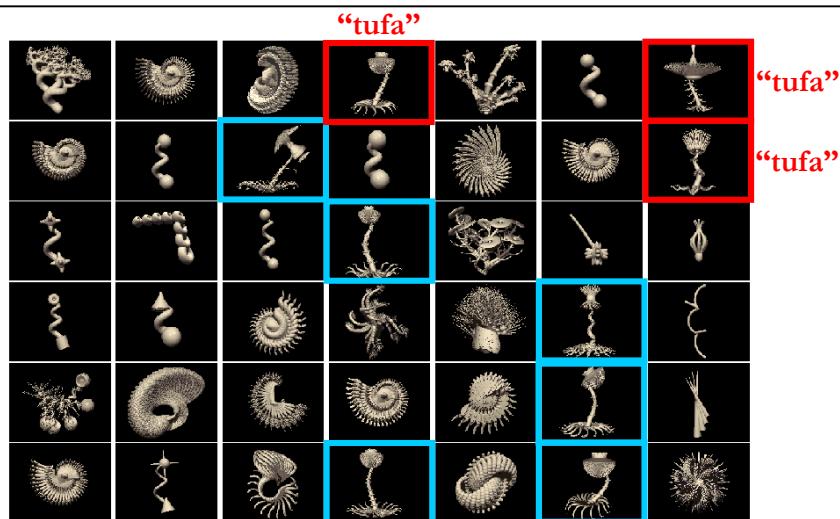
Can you pick out the tufas? [tufa := a form of calcite rock]

Inductive Learning (from J. Tenenbaum)



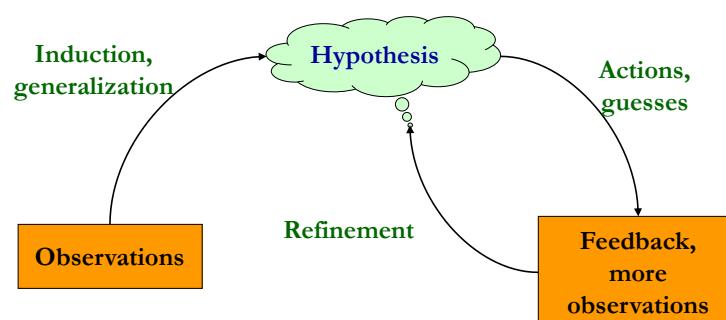
Can you pick out the tufas? [tufa := a form of calcite rock]

Inductive Learning (from J. Tenenbaum)



Can you pick out the tufas? [tufa := a form of calcite rock]

General Inductive Learning



Why use Machine Learning?

- We cannot write the program ourselves

Traditional Programming



Machine Learning



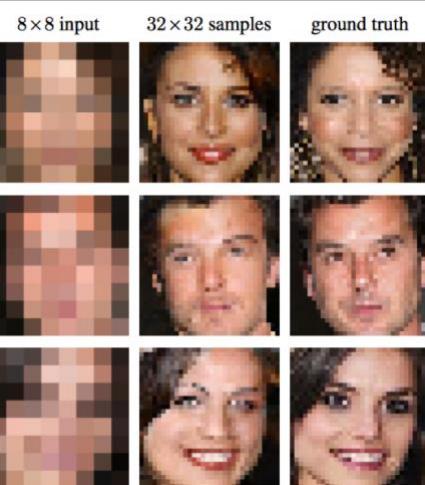
Why use Machine Learning?

- We cannot write the program ourselves
- We don't have the expertise
- We cannot explain how
- Problem changes over time
- Need customized solutions

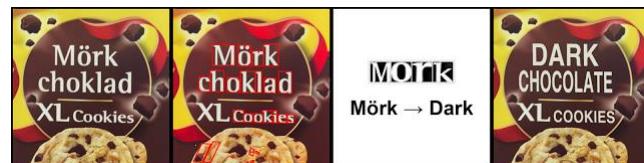
Machine Learning in Action

- Face, speech, handwriting recognition
 - Pattern recognition
- Spam filtering, terrain navigability (rovers)
 - Classification
- Credit risk assessment, weather forecasting, stock market prediction
 - Regression
- Self-driving cars
- Translating phones

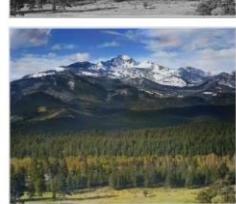
Pixel Recursive Super Resolution



Translation



Restore colors in B&W photos & videos



Colorado National Park, 1941



Textile Mill, June 1937



Berry Field, June 1909



Hamilton, 1936

Art: Transferring style from famous paintings



Machine Learning Paradigms

- Decision Trees
- Bayesian Networks
- Neural Networks
- Deep Learning
- Genetic Algorithms
- Fuzzy Logic

Decision Trees (DT)

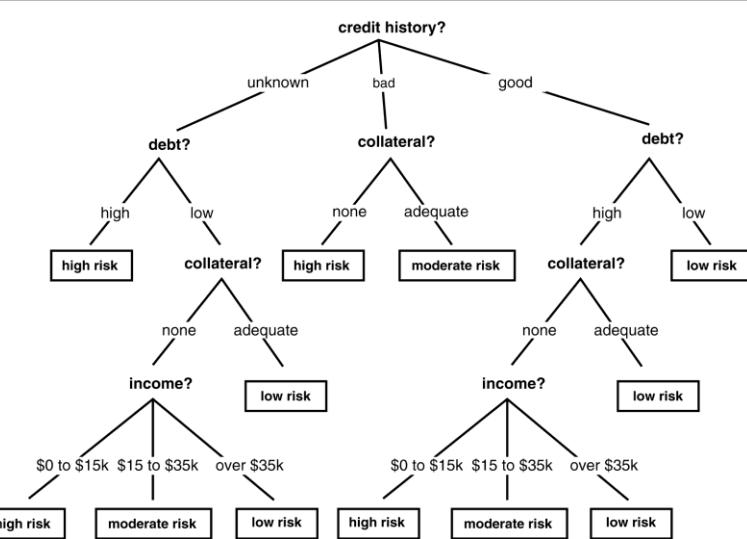
The ID3 algorithm

- ID3 (Quinlan's, Iterative Dichotomizer 3) induces concepts from examples.
- ID3 represents concepts as decision trees.
 - Decision tree: a representation that allows us to determine the classification of an object by testing its, generally, **nonnumeric** (**linguistic**) **values** for certain **properties**
- Variant of ID3: the **C4.5 algorithm**
- Example problem: estimate an individual's **credit risk** on the basis of credit history, debt, collateral, and income

Credit history data of loan applications

NO.	RISK	CREDIT HISTORY	DEBT	COLLATERAL	INCOME
1.	high	bad	high	none	\$0 to \$15k
2.	high	unknown	high	none	\$15 to \$35k
3.	moderate	unknown	low	none	\$15 to \$35k
4.	high	unknown	low	none	\$0 to \$15k
5.	low	unknown	low	none	over \$35k
6.	low	unknown	low	adequate	over \$35k
7.	high	bad	low	none	\$0 to \$15k
8.	moderate	bad	low	adequate	over \$35k
9.	low	good	low	none	over \$35k
10.	low	good	high	adequate	over \$35k
11.	high	good	high	none	\$0 to \$15k
12.	moderate	good	high	none	\$15 to \$35k
13.	low	good	high	none	over \$35k
14.	high	bad	high	none	\$15 to \$35k

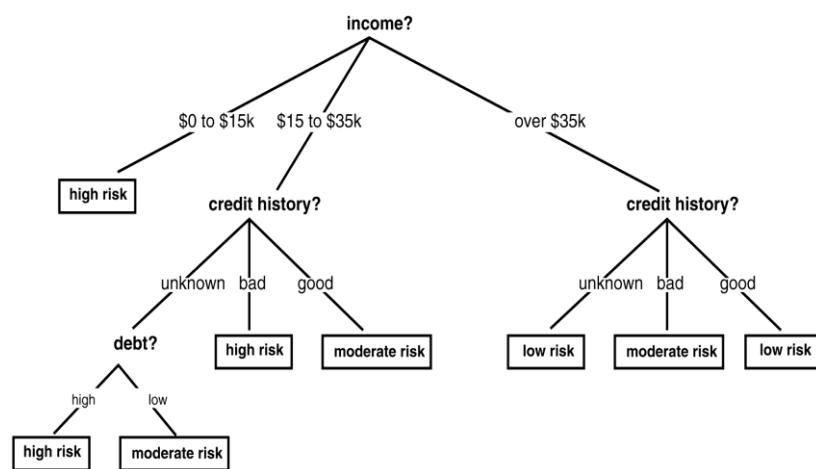
DT for credit risk assessment



The ID3 DT Induction Algorithm

- In a decision tree,
 - Each **internal node** represents a test on some **property** such as *credit history* or *debt*
 - Each **possible value** of the property corresponds to a **branch** of the tree such as *high* or *low*
 - **Leaf nodes** represent **classifications** such as *low* or *moderate risk*
 - An individual of unknown type may be classified by traversing the decision tree.
- The **size of the tree** necessary to classify a given set of examples varies according to **the order with which properties are tested**.
 - The next figure shows a simpler tree that also classifies the examples.

A simpler decision tree



ID3 DT Induction Algorithm

□ Choice of the optimal tree

- measure of optimality
 - the likelihood of correctly classifying unseen data
- assumption of ID3 algorithm
 - “the simplest decision tree that covers all the training examples” is the **optimal tree**
 - rationale for this assumption is time-honored heuristic of preferring simplicity & avoiding unnecessary assumptions

Occam's Razor principle

*“It is vain to do with more what can be done with less....
Entities should not be multiplied beyond necessity”*

Top-down Decision Tree Induction

□ ID3 algorithm

- constructs decision tree in a **top-down fashion**
- **selects a property** at the current node of the tree
- uses the property to **partition** the set of examples
- **recursively constructs a subtree** for each partition
- **continues until** all members of the partition are **in the same class**
- since the order of tests is critical, **ID3 relies on its criteria for selecting the test**

DT Construction Algorithm

```

function induce_tree (example_set, Properties)
begin
if all entries in example_set are in the same class
    then return a leaf node labeled with that class
else if Properties is empty
    then return leaf node labeled with disjunction of all classes in example_set
else begin
    select a property, P, and make it the root of the current tree;
    delete P from Properties;
    for each value, V, of P,
        begin
            create a branch of the tree labeled with V;
            let partitionv be elements of example_set with values V for property P;
            call induce_tree(partitionv, Properties), attach result to branch V
        end
    end
end

```

Entropy-based Property Selection

- Select properties based on **Shannon's Information Entropy**
- Shannon entropy, quantifies, in the sense of an expected value, the information contained in a message, usually in units such as bits.
- If p is the probability of an event, the entropy is defined as $E = p \log_2 (1/p) = - p \log_2 p$ (bits)
- Assume $p = 0$ or $p = 1$. Then, $E = 0$ bits, i.e. the event does not convey any information (either does not ever happen or it is always happening). We obtain the highest amount of information for events with probabilities $p = 0.5$, in which case: $E = - 0.5 \log_2 0.5 = 0.5$ bits.

Entropy-based Property Selection

- The entropy of a set of N events each with a probability of occurrence p_i is given by:

$$E = \sum_{i=1}^N -p_i \log_2 p_i$$

- Suppose I is a training set of N examples belonging to C categories (events). Estimating the a priori probability p_i of each category in I by its frequency of occurrence, $p_i = N_i/N$, where N_i is the total number of examples belonging to the i^{th} category, the entropy (in bits) of I is given by:

$$E = \sum_{i=1}^C -p_i \log_2 p_i = \sum_{i=1}^C -(N_i/N) \log_2 (N_i/N)$$

Entropy-based Property Selection

- Referring to the Credit Risk Assessment example, there are three categories, i.e. **low risk (lr)**, **moderate risk (mr)** and **high risk (hr)**. Then, the entropy of the initial training set I is computed as:

$$\begin{aligned} E(I) &= -p_{lr} \log_2 p_{lr} - p_{mr} \log_2 p_{mr} - p_{hr} \log_2 p_{hr} = \\ &= -5/14 \log_2 5/14 - 3/14 \log_2 3/14 - 6/14 \log_2 6/14 = \\ &= 1.5306 \text{ bits} \end{aligned}$$

Credit history data of loan applications

NO.	RISK	CREDIT HISTORY	DEBT	COLLATERAL	INCOME
1.	high	bad	high	none	\$0 to \$15k
2.	high	unknown	high	none	\$15 to \$35k
3.	moderate	unknown	low	none	\$15 to \$35k
4.	high	unknown	low	none	\$0 to \$15k
5.	low	unknown	low	none	over \$35k
6.	low	$p_{lr} = 5/14$		adequate	over \$35k
7.	high	ow	ow	none	\$0 to \$15k
8.	moderate	bad	low	adequate	over \$35k
9.	low	good	low	none	over \$35k
10.	low	good	high	adequate	over \$35k
11.	high	good	high	none	\$0 to \$15k
12.	moderate	good	high	none	\$15 to \$35k
13.	low	good	high	none	over \$35k
14.	high	bad	high	none	\$15 to \$35k

Credit history data of loan applications

NO.	RISK	CREDIT HISTORY	DEBT	COLLATERAL	INCOME
1.	high	bad	high	none	\$0 to \$15k
2.	high	unknown	high	none	\$15 to \$35k
3.	moderate	unknown	low	none	\$15 to \$35k
4.	high	unknown	low	none	\$0 to \$15k
5.	low	unknown	low	none	over \$35k
6.	low	$p_{mr} = 3/14$		adequate	over \$35k
7.	high	ow	ow	none	\$0 to \$15k
8.	moderate	bad	low	adequate	over \$35k
9.	low	good	low	none	over \$35k
10.	low	good	high	adequate	over \$35k
11.	high	good	high	none	\$0 to \$15k
12.	moderate	good	high	none	\$15 to \$35k
13.	low	good	high	none	over \$35k
14.	high	bad	high	none	\$15 to \$35k

Credit history data of loan applications

NO.	RISK	CREDIT HISTORY	DEBT	COLLATERAL	INCOME
1.	high	bad	high	none	\$0 to \$15k
2.	high	unknown	high	none	\$15 to \$35k
3.	moderate	unknown	low	none	\$15 to \$35k
4.	high	unknown	low	none	\$0 to \$15k
5.	low	unknown	low	none	over \$35k
6.	low	now	now	adequate	over \$35k
7.	high	now	now	none	\$0 to \$15k
8.	moderate	bad	low	adequate	over \$35k
9.	low	good	low	none	over \$35k
10.	low	good	high	adequate	over \$35k
11.	high	good	high	none	\$0 to \$15k
12.	moderate	good	high	none	\$15 to \$35k
13.	low	good	high	none	over \$35k
14.	high	bad	high	none	\$15 to \$35k

p_{hr} = 6/14

Entropy-based Property Selection

- To select the first property, partition the training set based on the values of each property in turn and compute the entropy of each partition I_k :

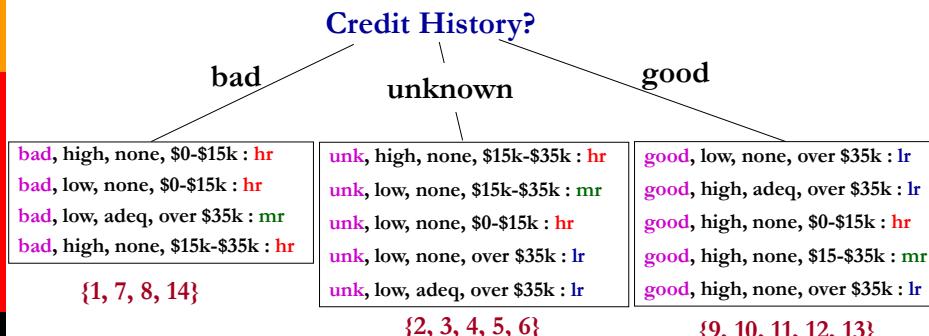
$$E(I_k) = \sum_{i=1}^C - p_{i,k} \log_2 p_{i,k} = \sum_{i=1}^C - (N_{i,k}/N_k) \log_2 (N_{i,k}/N_k)$$

where $N_{i,k}$ is the number of examples in I_k belonging to class i and N_k is the total number of examples in I_k .

- The overall system entropy is then computed as:

$$E(I, \text{property}) = \sum_k (N_k/N) E(I_k)$$

Partition based on “Credit History”



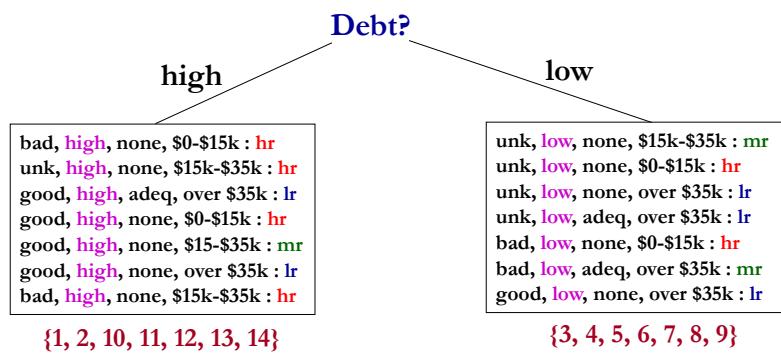
$$E(I_{\text{bad}}) = -1/4 \log_2 1/4 - 3/4 \log_2 3/4 = 0.8113$$

$$E(I_{\text{unknown}}) = -2/5 \log_2 2/5 - 1/5 \log_2 1/5 - 2/5 \log_2 2/5 = 1.5219$$

$$E(I_{\text{good}}) = -3/5 \log_2 3/5 - 1/5 \log_2 1/5 - 1/5 \log_2 1/5 = 1.3710$$

$$E(I, \text{Credit Hist}) = 4/14 E(I_{\text{bad}}) + 5/14 E(I_{\text{unknown}}) + 5/14 E(I_{\text{good}}) = 1.2650$$

Partition based on “Debt”

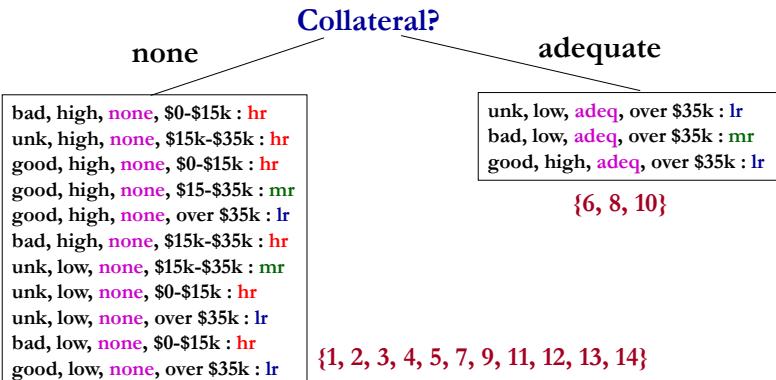


$$E(I_{\text{high}}) = -2/7 \log_2 2/7 - 1/7 \log_2 1/7 - 4/7 \log_2 4/7 = 1.3788$$

$$E(I_{\text{low}}) = -3/7 \log_2 3/7 - 2/7 \log_2 2/7 - 2/7 \log_2 2/7 = 1.5567$$

$$E(I, \text{Debt}) = 7/14 E(I_{\text{high}}) + 7/14 E(I_{\text{low}}) = 1.4678$$

Partition based on “Collateral”

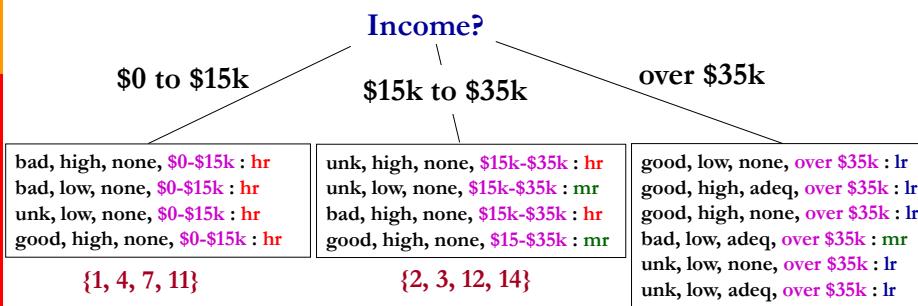


$$E(I_{\text{none}}) = - \frac{3}{11} \log_2 \frac{3}{11} - \frac{2}{11} \log_2 \frac{2}{11} - \frac{6}{11} \log_2 \frac{6}{11} = 1.4354$$

$$E(I_{\text{adequate}}) = - \frac{2}{3} \log_2 \frac{2}{3} - \frac{1}{3} \log_2 \frac{1}{3} = 0.9183$$

$$E(I, \text{Collateral}) = \frac{11}{14} E(I_{\text{none}}) + \frac{3}{14} E(I_{\text{adequate}}) = 1.3246$$

Partition based on “Income”



$$E(I_{0to15}) = - \frac{4}{4} \log_2 \frac{4}{4} = 0$$

$$E(I_{15to35}) = - \frac{2}{4} \log_2 \frac{2}{4} - \frac{2}{4} \log_2 \frac{2}{4} = 1$$

$$E(I_{\text{over35}}) = - \frac{5}{6} \log_2 \frac{5}{6} - \frac{1}{6} \log_2 \frac{1}{6} = 0.65$$

$$E(I, \text{Income}) = \frac{4}{14} E(I_{0to15}) + \frac{4}{14} E(I_{15to35}) + \frac{6}{14} E(I_{\text{over35}}) = 0.5643$$

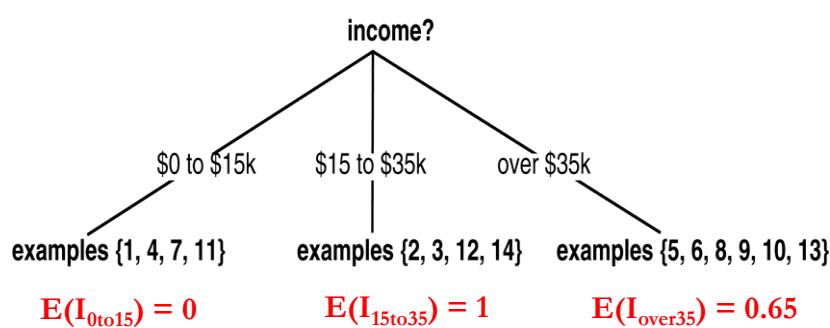
Information Gain

- Gain testing on “Credit History”: $1.5306 - 1.2650 = 0.2656$ bits
- Gain testing on “Debt”: $1.5306 - 1.4678 = 0.0628$ bits
- Gain testing on “Collateral”: $1.5306 - 1.3246 = 0.2060$ bits
- Gain testing on “Income”: $1.5306 - 0.5643 = 0.9663$ bits

=>

“Income” gives the highest information gain
and is the first property selected for testing

Decision tree based on “income”



Select the second property

- Since $E(I_{0\text{to}15}) = 0$ this will be a leaf “high risk” node of the tree.
- Testing with the other properties needs to be done on the other partitions, i.e. $I_{15\text{to}35}$ and $I_{\text{over}35}$.

Try “Credit History”

- $I_{15\text{to}35, \text{bad}} = \{14\}$ and $E(I_{15\text{to}35, \text{bad}}) = 0$
- $I_{15\text{to}35, \text{unk}} = \{2, 3\}$ and $E(I_{15\text{to}35, \text{unk}}) = 1$
- $I_{15\text{to}35, \text{good}} = \{12\}$ and $E(I_{15\text{to}35, \text{good}}) = 0$
- $I_{\text{over}35, \text{bad}} = \{8\}$ and $E(I_{\text{over}35, \text{bad}}) = 0$
- $I_{\text{over}35, \text{unk}} = \{5, 6\}$ and $E(I_{\text{over}35, \text{unk}}) = 0$
- $I_{\text{over}35, \text{good}} = \{9, 10, 13\}$ and $E(I_{\text{over}35, \text{good}}) = 0$

$$E(I_{15\text{to}35}, \text{“Credit History”}) = (2/4) E(I_{15\text{to}35, \text{unk}}) = 0.5$$

$$E(I_{\text{over}35}, \text{“Credit History”}) = 0$$

Try “Debt”

□ $I_{15\text{to}35, \text{high}} = \{2, 12, 14\}$ and $E(I_{15\text{to}35, \text{high}}) = 0.9183$

□ $I_{15\text{to}35, \text{low}} = \{3\}$ and $E(I_{15\text{to}35, \text{low}}) = 0$

□ $I_{\text{over}35, \text{high}} = \{10, 13\}$ and $E(I_{\text{over}35, \text{high}}) = 0$

□ $I_{\text{over}35, \text{low}} = \{5, 6, 8, 9\}$ and $E(I_{\text{over}35, \text{low}}) = 0.8113$

$$E(I_{15\text{to}35}, \text{“Debt”}) = (3/4) E(I_{15\text{to}35, \text{high}}) = 0.6887$$

$$E(I_{\text{over}35}, \text{“Debt”}) = (4/6) E(I_{\text{over}35, \text{low}}) = 0.5409$$

Try “Collateral”

□ $I_{15\text{to}35, \text{none}} = \{2, 3, 12, 14\}$ and $E(I_{15\text{to}35, \text{none}}) = 1$

□ $I_{15\text{to}35, \text{adeq}} = \{\}$ and $E(I_{15\text{to}35, \text{adeq}}) = 0$

□ $I_{\text{over}35, \text{none}} = \{5, 9, 13\}$ and $E(I_{\text{over}35, \text{none}}) = 0$

□ $I_{\text{over}35, \text{adeq}} = \{6, 8, 10\}$ and $E(I_{\text{over}35, \text{adeq}}) = 0.9183$

$$E(I_{15\text{to}35}, \text{“Collateral”}) = 1$$

$$E(I_{\text{over}35}, \text{“Collateral”}) = (3/6) E(I_{\text{over}35, \text{adeq}}) = 0.4592$$

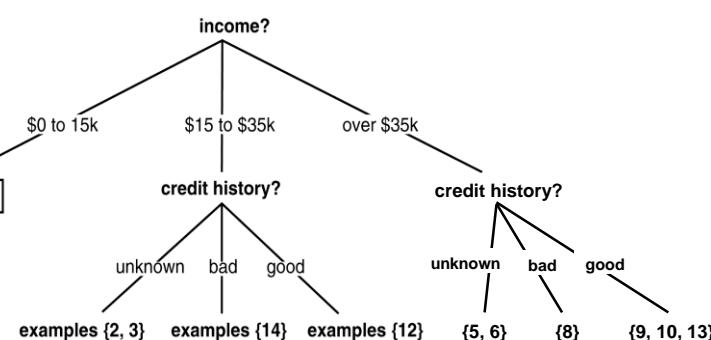
Information Gain on 2nd property

- “Credit History” gives highest information gain (or smallest entropy) for $I_{15\text{to}35}$: $1 - 0.5 = 0.5 \text{ bits}$
- “Credit History” also gives highest information gain for $I_{\text{over}35}$, that is: $0.65 - 0 = 0.65 \text{ bits}$

==>

Use “Credit History” to minimize both the entropy of $I_{15\text{to}35}$ and the entropy of $I_{\text{over}35}$

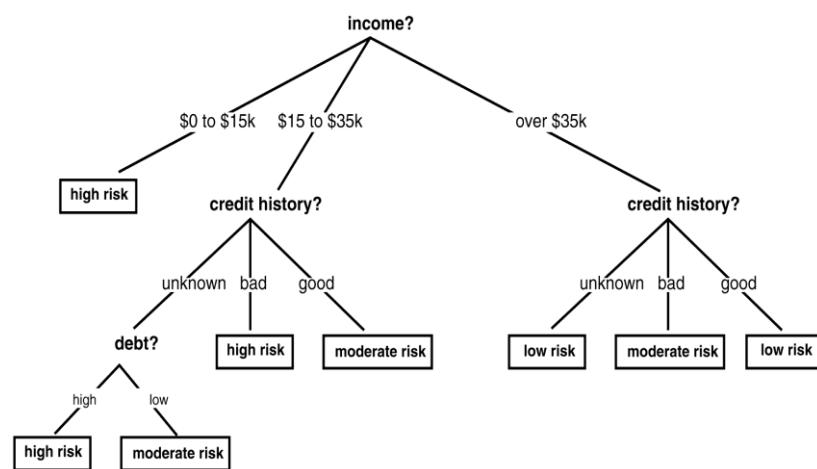
Partially constructed Decision Tree



Select the third property

- The only nonzero entropy is that of $I_{15\text{to}35, \text{unk}}$ and to minimize it we test on the remaining properties
 - Examine “Debt” :
- $I_{15\text{to}35, \text{unk}, \text{high}} = \{2\}$ and $E(I_{15\text{to}35, \text{unk}, \text{high}}) = 0$
- $I_{15\text{to}35, \text{unk}, \text{low}} = \{3\}$ and $E(I_{15\text{to}35, \text{unk}, \text{low}}) = 0$
- Hence, $E(I_{15\text{to}35, \text{unk}}, \text{“Debt”}) = 0$ and the algorithm returns the final Decision Tree and stops.

Final – simplified – decision tree



Advantages of ID3

- Easy to implement.
- Discovers which feature (property) combinations are sufficient to determine class membership. In a sense, **ID3 discovers concepts.**
- ID3's computation time increases linearly with problem complexity, i.e. number of examples (patterns) and number of properties on the decision tree.

Disadvantages of ID3

- Not easy to update. In fact, in order to update the decision tree (e.g. to accommodate new examples) we have to rebuilt the entire tree.
- Does not address the issue of “generalization”.