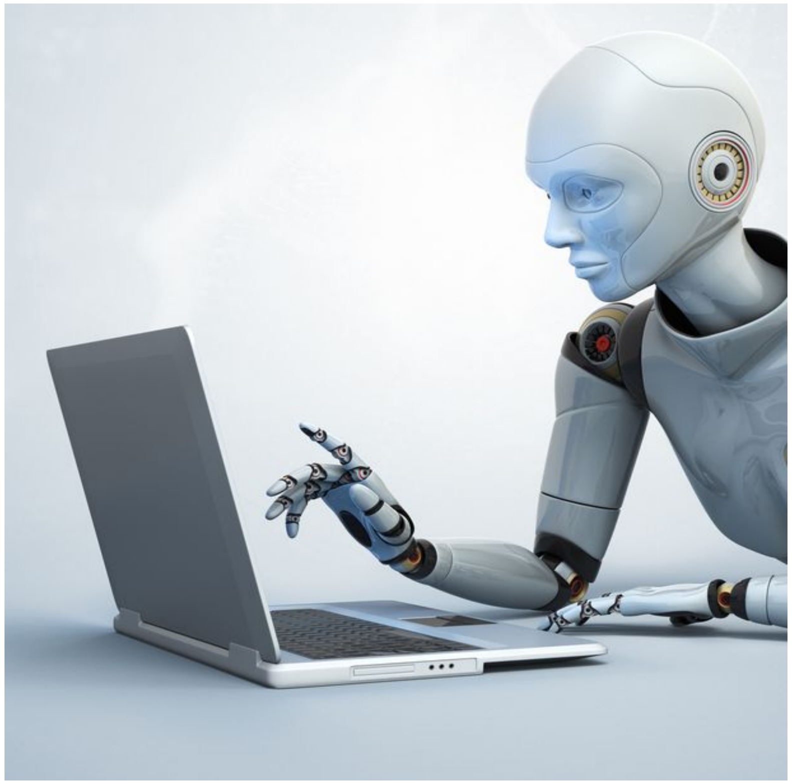
Machine Learning

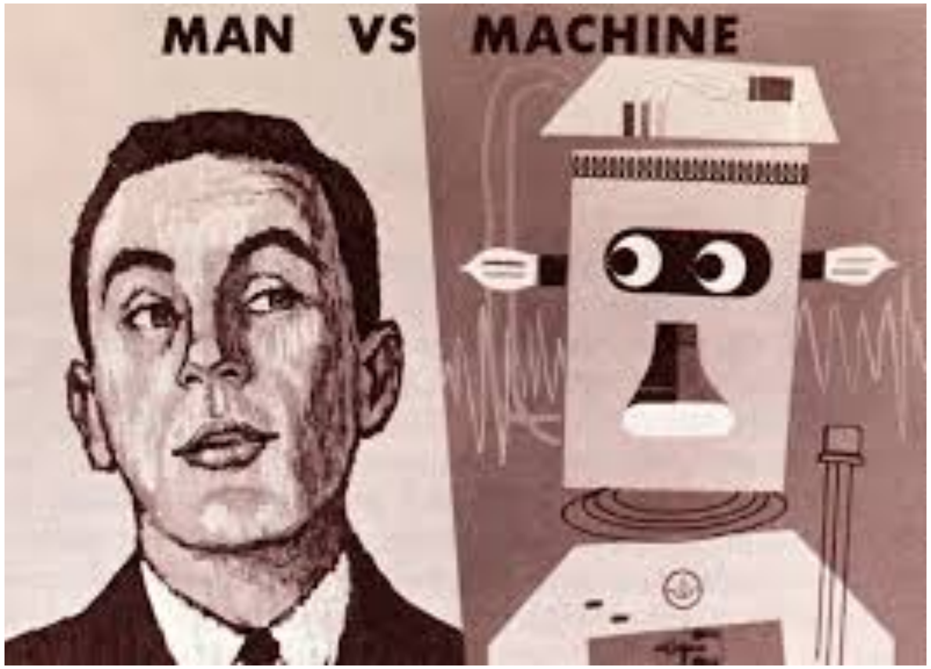


Sohrab Hossain, CSE, EDU

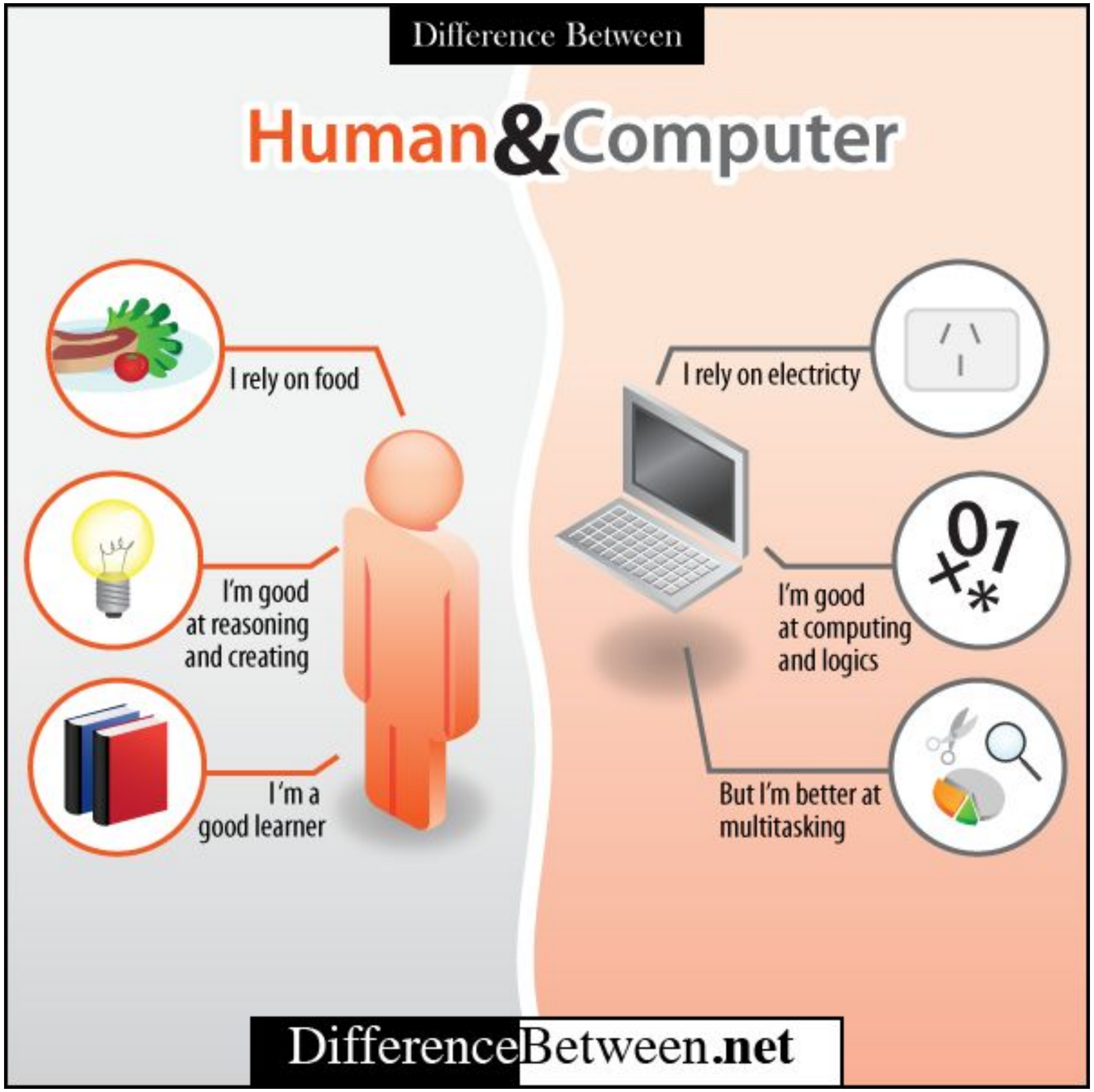
Outlines

- Man vs. Machine
- Intelligent Agents & Some Examples
- Machine Learning
  - Application domains
  - Algorithms

Man vs. Machine



Man vs. Machine



Man vs. Machine

	Human	Computer
Strengths	▪Have common sense and bigger knowledge base, thus can percept his environment better than computer given appropriate means (especially in visual form). ▪Can think (synthesize) new rules `out of the box'. ▪Psychologically, human decision is more trusted than computer expert system decision. ▪Can detect trends, patterns, or anomalies, in visualization data. ▪Good in learning.	▪Speed: Fast. ▪ Reliable. ▪Endurance: Not tired. ▪Unbiased. ▪Consistent. ▪Can try much more combinations than what human is capable of.

Man vs. Machine

	Human	Computer
Weaknesses	▪ Easily tired and bored, thus can only be utilized for a short period of time, perhaps as `oracle' only. ▪ Cannot do micro manage. ▪ Biased and inconsistent. ▪ Can make error. ▪ Not a perfect decision maker. ▪ Actually cannot see anything if the data is presented in awkward manner.	▪Difficult to synthesize new rules (cannot think `out of the box'). ▪ Limited knowledge base. ▪ No common sense.

Man vs. Machine

Key Difference….

- Intelligence
- How to build up intelligence into machine or intelligent agents?
- Solution: Artificial Intelligence (AI)

What is Intelligence?

- **Intelligence** (also called intellect) is an umbrella term used to describe a property of the mind that encompasses many related abilities, such as the capacities to reason, to plan, to solve problems, to think abstractly, to comprehend ideas, to use language, & to learn [*Wikipedia*]

Where is mind?

What is intelligence?

- As the ability to acquire, understand & apply knowledge, or the ability to exercise thought & reason [Dictionary]
- Intelligence is more than this!!!

Vision of AI

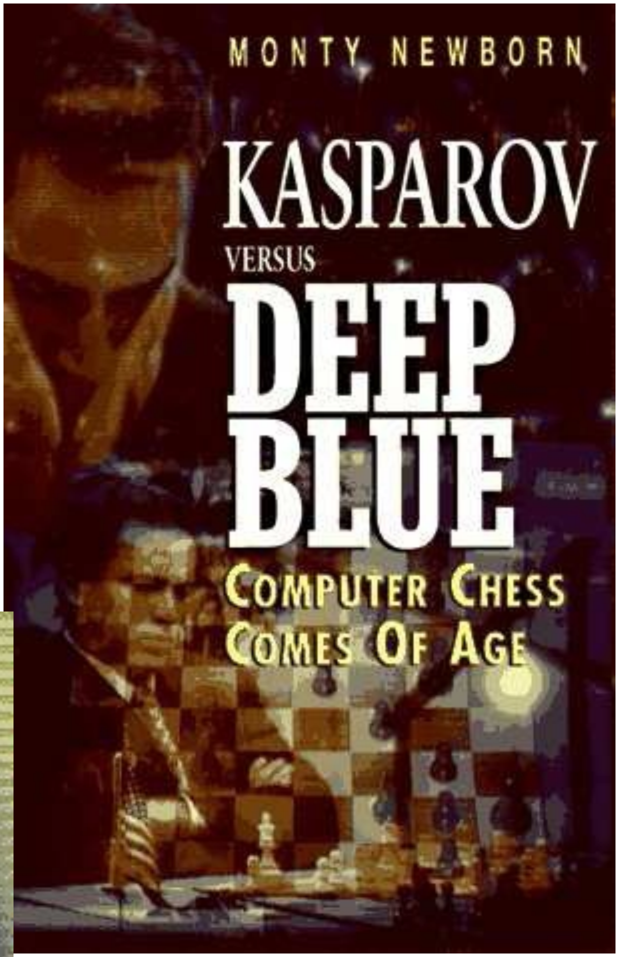
Develop systems that matches or exceeds human intelligence: the intelligence of a machine that could successfully perform any intellectual task that a human being can

Is it really possible?

Other Notable Examples..

Chess (Deep Blue, 1997)

"I could feel –
I could smell –
a new kind of
intelligence
across the
table"
-Gary
Kasparov



Speech Recognition

Navigation Systems

Automated call centers

Museum Tour-Guide Robots

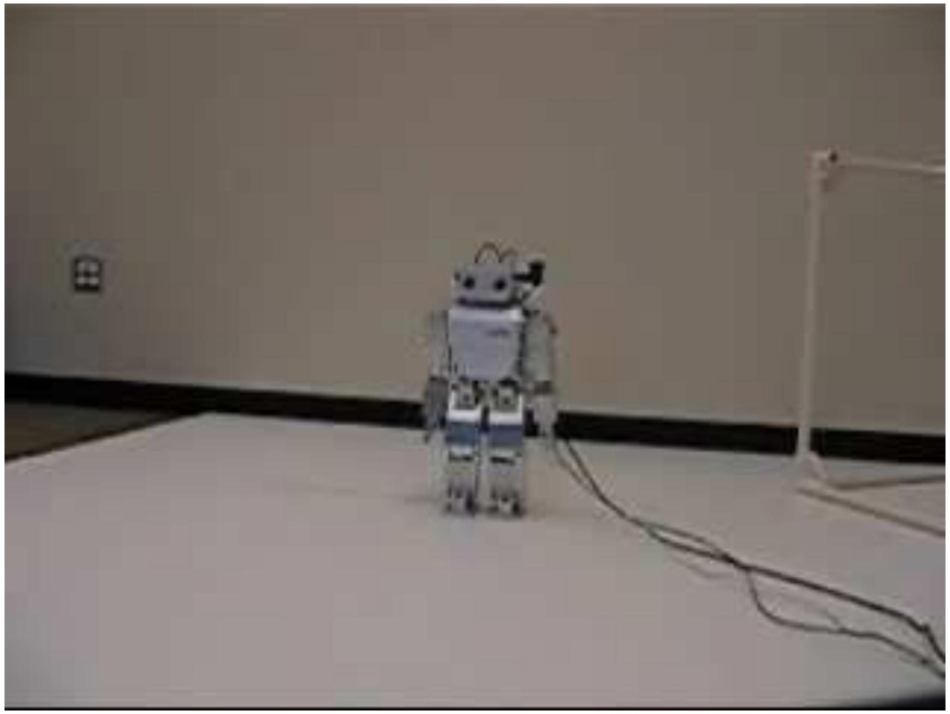
Rhino, 1997

Minerva, 1998

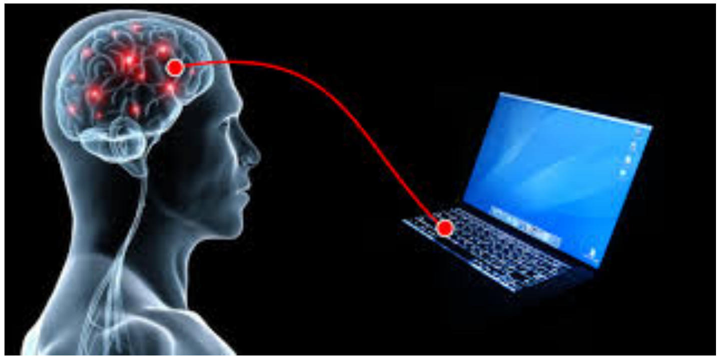
Mars Rovers (2003-now)

Europa Mission ~ 2018?

Humanoid Robots



Brain-Computer Interfaces



Singing, Dancing, Bride, …..

How it would be Possible?

Use Machine Learning

Machine Learning

- Learning = Improving performance with experience at some task
- Study of algorithms that
  - improve their **performance P**
  - at some **task T**
  - with **experience E**

- well-defined learning task: <P,T,E>
- Machine learning systems automatically learn programs from data

Machine Learning

- **LEARNING = REPRESENTATION + EVALUATION + OPTIMIZATION**
  - **Representation:** classifier must be represented in some formal language that the computer can handle.
  - **Evaluation**: evaluation function (*objective/ or scoring function) is needed to distinguish* good classifiers from bad ones.
  - **Optimization**: method to search among the classifiers for the highest-scoring one.

03 components of learning algorithms

Representation	Evaluation	Optimization
Instances	Accuracy/Error rate	**Combinatorial optimization**
K-nearest neighbor	Precision and recall	Greedy search
Support vector machines	Squared error	Beam search
Hyperplanes	Likelihood	Branch-and-bound
Naive Bayes	Posterior probability	**Continuous optimization**
Logistic regression	Information gain	**Unconstrained**
Decision trees	K-L divergence	Gradient descent
Sets of rules	Margin	Conjugate gradiant
Propositional rules		Quasi-Newton methods
Logic programs		**Constrained**
Neural networks		Linear programming
Graphical models		Quadratic programming
Bayesian networks		
Conditional random fields		

Why Machine Learning?

- Some tasks cannot be defined well, except by examples (e.g., recognizing people).
- Relationships & correlations can be hidden within large amounts of data

✓ **ML may be able to find these relationships.**

- Human designers often produce machines that do not work as well as desired in the environments in which they are used.

Why Machine Learning?

- The amount of knowledge available about certain tasks might be too large for explicit encoding by humans (e.g., medical diagnostic).
- Environments change over time.
- New knowledge about tasks is constantly being discovered by humans (e.g. autonomous driving)

✔ It may be difficult to continuously re-design systems “by hand”.

Pros & Cons

- **Pros:**
  - often much more accurate than human-crafted rules (since data driven)
  - humans often incapable of expressing what they know (e.g., rules of English, or how to recognize letters), but can easily classify examples
  - don’t need a human expert or programmer
  - automatic method to search for hypotheses explaining data
  - cheap & flexible — can apply to any learning task

- **Cons:**
  - need a lot of labeled data
  - error prone — usually impossible to get perfect accuracy

Machine Learning Applications

Countless……….

- Machine perception
- Computer vision, including object recognition
- Natural language processing
- Syntactic pattern recognition
- Search engines
- Medical diagnosis
- Bioinformatics
- Brain-machine interfaces
- Cheminformatics
- Detecting credit card fraud
- Stock market analysis
- Classifying DNA sequences
- Sequence mining
- Speech and handwriting recognition
- Game playing
- Software engineering
- Adaptive websites
- Robot locomotion
- Computational advertising
- Computational finance
- Structural health monitoring
- Sentiment analysis (or opinion mining)
- Affective computing
- Information retrieval
- Recommender systems
- Optimization and Metaheuristic

Few Examples

Credit Risk Analysis

- **Data**

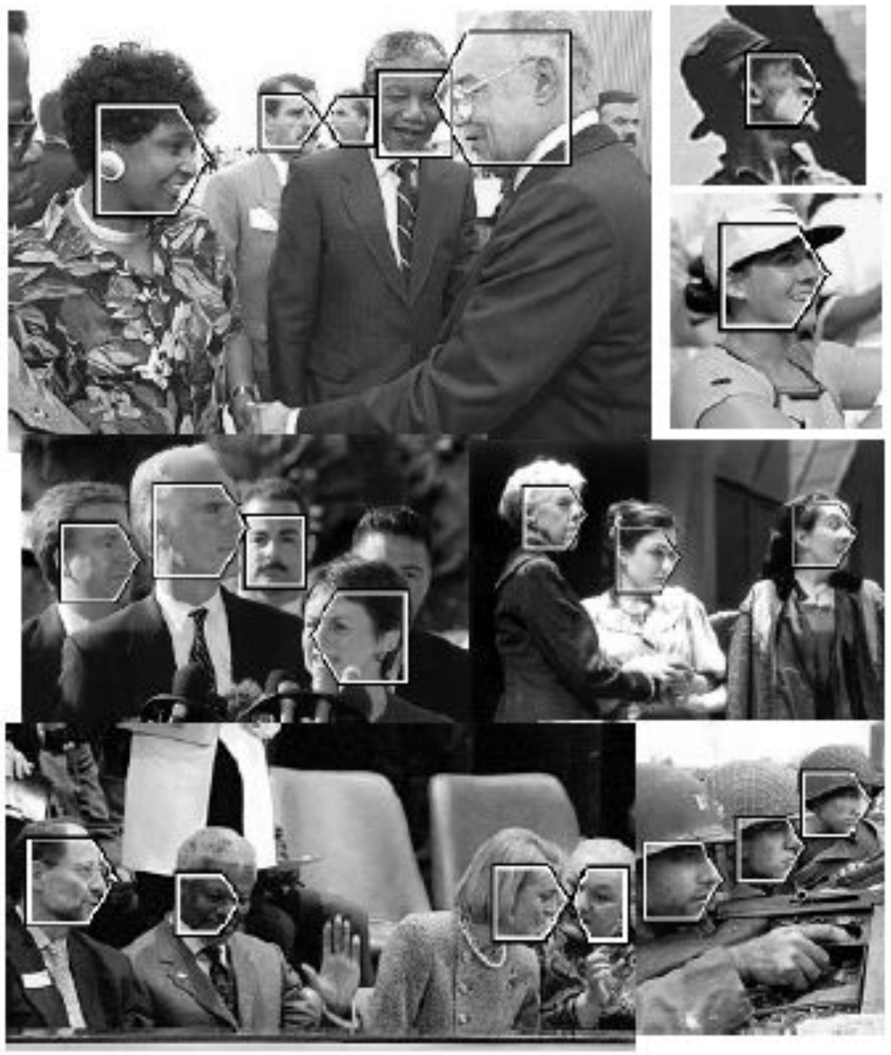
Customer103: (time=t0)	*Customer103:* (time=t1)	…	*Customer103:* (time=tn)
Years of credit: 9	Years of credit: 9		Years of credit: 9
Loan balance: $2,400	Loan balance: $3,250		Loan balance: $4,500
Income: $52k	Income: ?		Income: ?
Own House: Yes	Own House: Yes		Own House: Yes
Other delinquent accts: 2	Other delinquent accts: 2		Other delinquent accts: 3
Max billing cycles late: 3	Max billing cycles late: 4		Max billing cycles late: 6
Profitable customer?: ?	Profitable customer?: ?		**Profitable customer?: No**
…	…		…

Rules learned from synthesized data

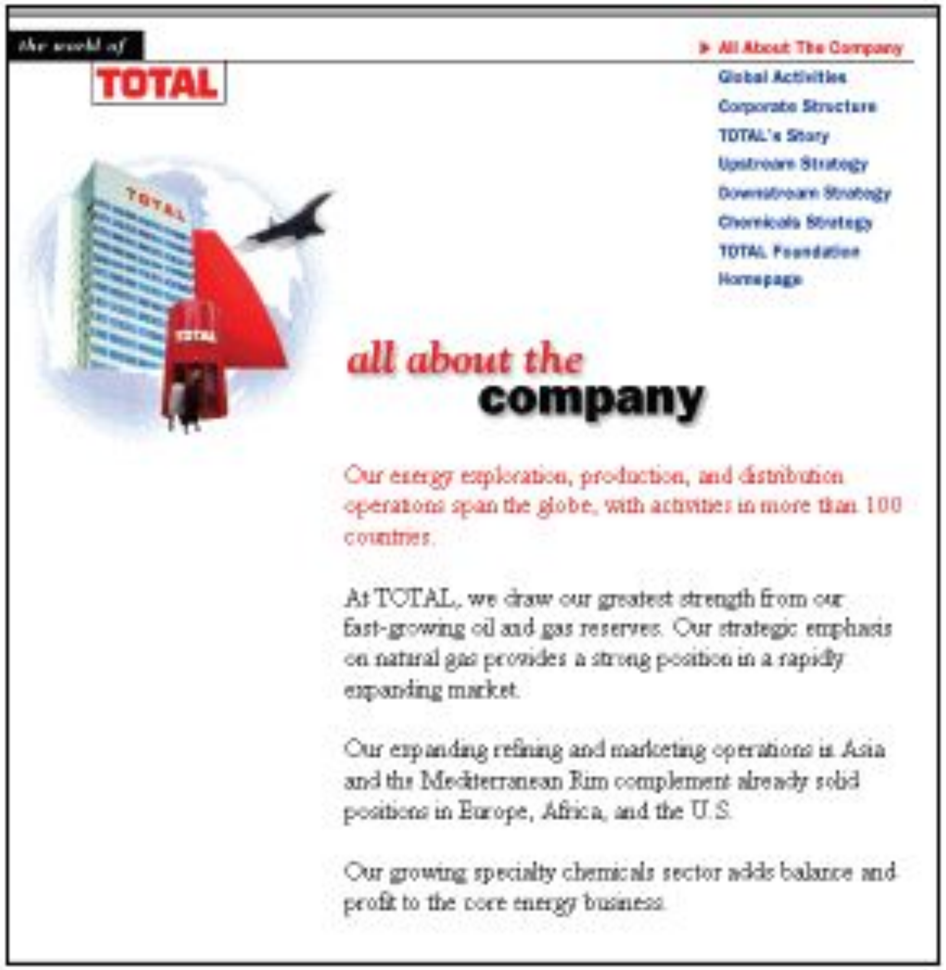
```
If    Other-Delinquent-Accounts > 2, and
      Number-Delinquent-Billing-Cycles > 1
Then  Profitable-Customer? = No
      [Deny Credit Card application]

If    Other-Delinquent-Accounts = 0, and
      (Income > $30k)  OR  (Years-of-Credit > 3)
Then  Profitable-Customer? = Yes
      [Accept Credit Card application]
```

Learning to detect objects in images

Example training images
for each orientation



Learning to classify text documents


the world of TOTAL

▸ All About The Company
Global Activities
Corporate Structure
TOTAL's Story
Upstream Strategy
Downstream Strategy
Chemicals Strategy
TOTAL Foundation
Homepage

all about the company

Our energy exploration, production, and distribution operations span the globe, with activities in more than 100 countries.

At TOTAL, we draw our greatest strength from our fast-growing oil and gas reserves. Our strategic emphasis on natural gas provides a strong position in a rapidly expanding market.

Our expanding refining and marketing operations in Asia and the Mediterranean Rim complement already solid positions in Europe, Africa, and the U.S.

Our growing specialty chemicals sector adds balance and profit to the core energy business.

$\rightarrow$

Company home page
vs.
Personal home page
vs.
University home page
vs
…

Information Extraction

Subject: **curriculum meeting**

Date: January 15, 2012

To: Dan Jura

Event:	Curriculum mtg
Date:	Jan-16-2012
Start:	10:00am
End:	11:30am
Where:	Gates 159

Hi Dan, we’ve now scheduled the curriculum meeting.

It will be in Gates 159 tomorrow from 10:00-11:30.

-Chris

Create new Calendar entry

Information Extraction & Sentiment Analysis

Attributes:
- zoom
- affordability
- size and weight
- flash
- ease of use

[illegible]

Size and weight

✓ • nice and compact to carry!

✓ • since the camera is small and light, I
around those heavy, bulky professic

✗ • the camera feels flimsy, is plastic and very light in weight you have to be very delicate in the handling of this camera

Machine Translation

- Fully automatic

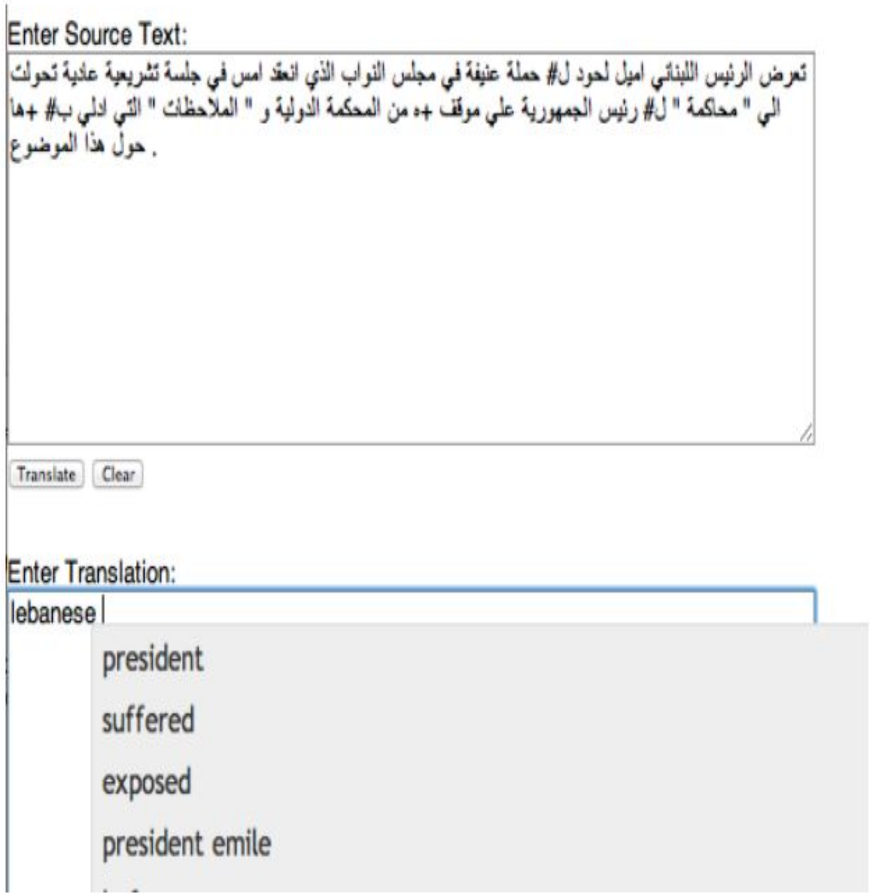
Enter Source Text:

这 不过 是 一 个 时间 的 问题 .

Translation from Stanford's *Phrasal*:

This is only a matter of time.

- Helping human translators



Enter Source Text:

تعرض الرئيس اللبناني اميل لحود ل# حملة عنيفة في مجلس النواب الذي انعقد امس في جلسة تشريعية عادية تحولت الي " محاكمة " ل# رئيس الجمهورية علي موقف +ه من المحكمة الدولية و " الملاحظات " التي ادلي ب# +ها حول هذا الموضوع .

Enter Translation:

lebanese

president
suffered
exposed
president emile

Dan Jurafsky

Language Technology

mostly solved

Spam detection

Let's go to Agra! ✓

Buy V1AGRA ... ✗

Part-of-speech (POS) tagging

ADJ	ADJ	NOUN	VERB	ADV
Colorless	green	ideas	sleep	furiously.

Named entity recognition (NER)

PERSON			ORG		LOC
Einstein	met with		UN	officials in	Princeton

making good progress

Sentiment analysis

Best roast chicken in San Francisco!

The waiter ignored us for 20 minutes.

Coreference resolution

Carter told Mubarak he shouldn't run again.

Word sense disambiguation (WSD)

I need new batteries for my ***mouse***.

Parsing

I can see Alcatraz from the window!

Machine translation (MT)

第13届上海国际电影节开幕...

The 13th Shanghai International Film Festival...

Information extraction (IE)

You're invited to our dinner party, Friday May 27 at 8:30

Party
May 27
add

still really hard

Question answering (QA)

Q. How effective is ibuprofen in reducing fever in patients with acute febrile illness?

Paraphrase

XYZ acquired ABC yesterday

ABC has been taken over by XYZ

Summarization

The Dow Jones is up

The S&P500 jumped

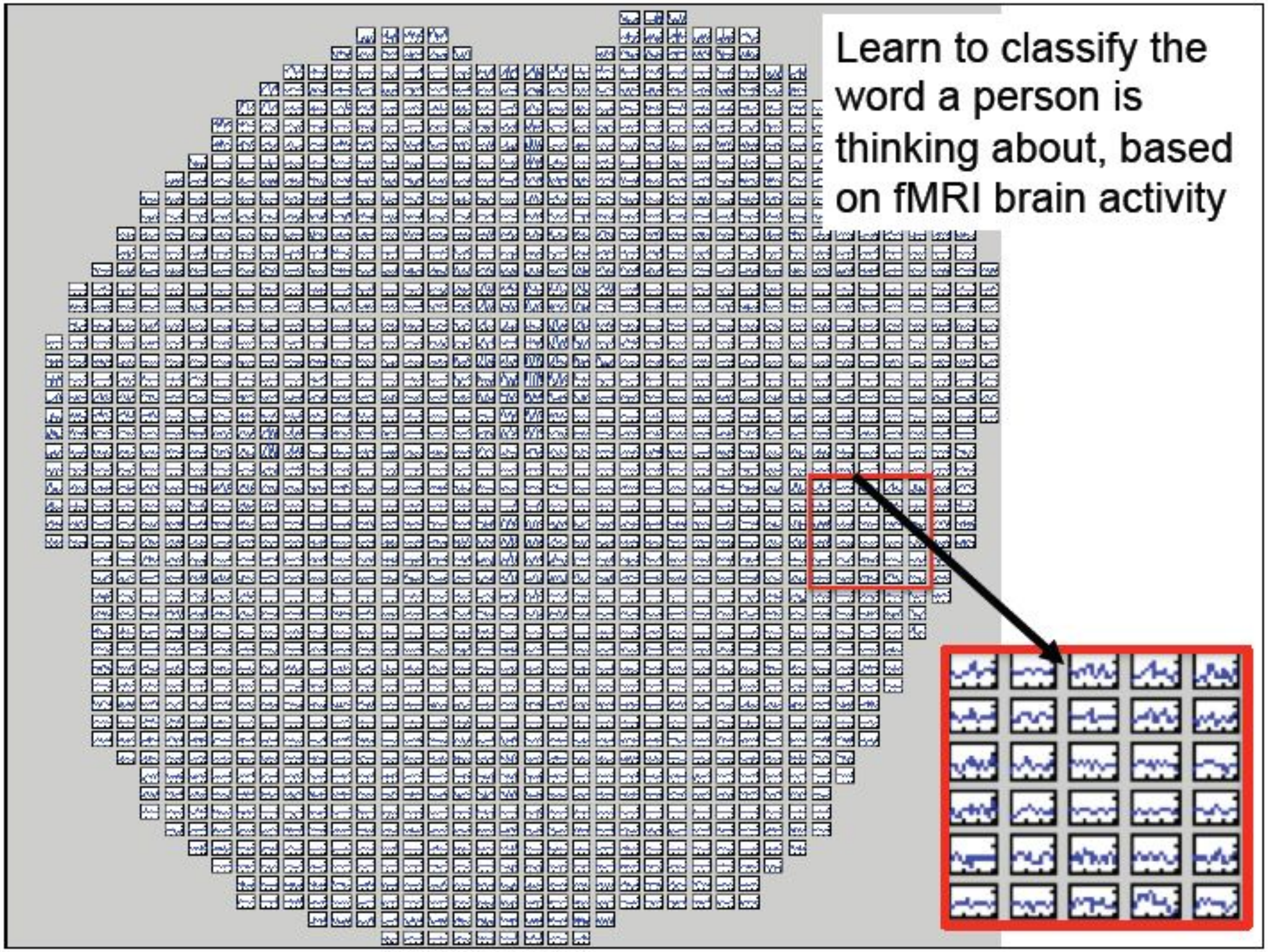
Housing prices rose

⇨ Economy is good

Dialog

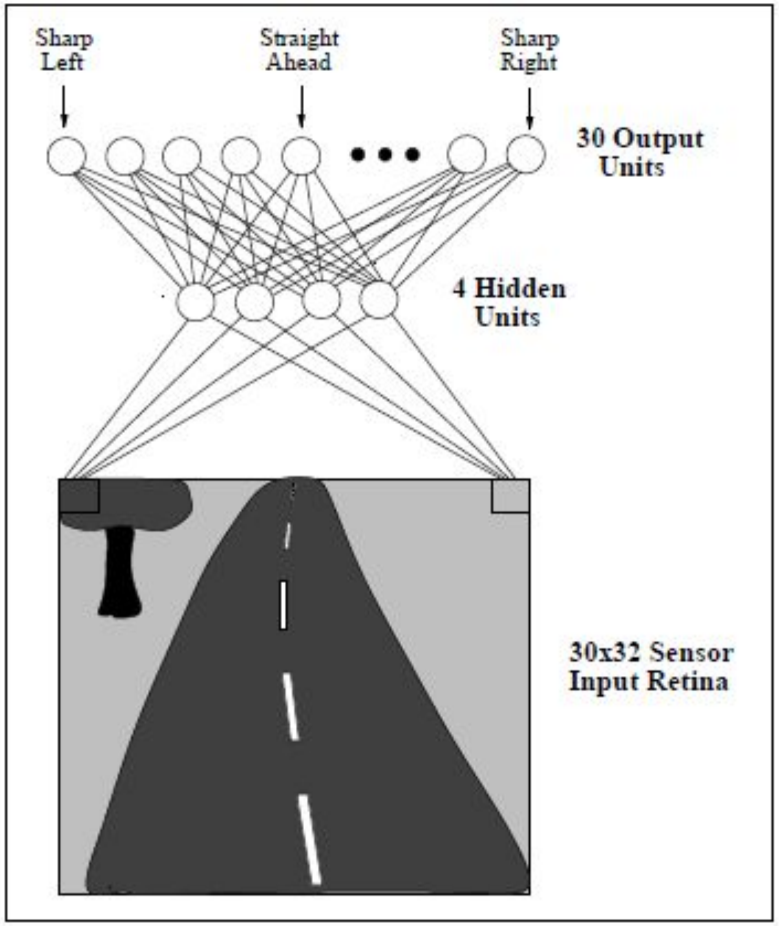
Where is Citizen Kane playing in SF?

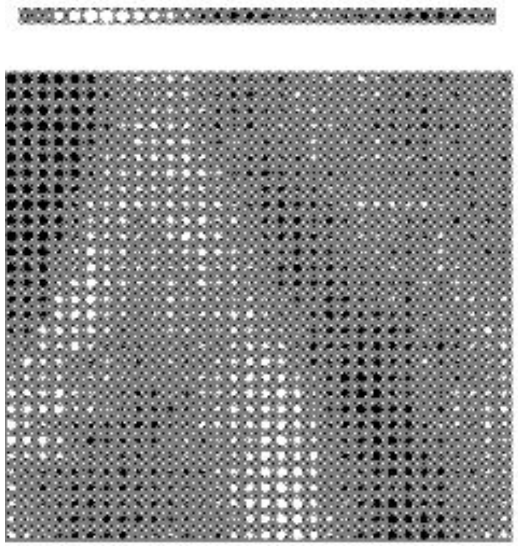
Castro Theatre at 7:30. Do you want a ticket?



Learn to classify the word a person is thinking about, based on fMRI brain activity

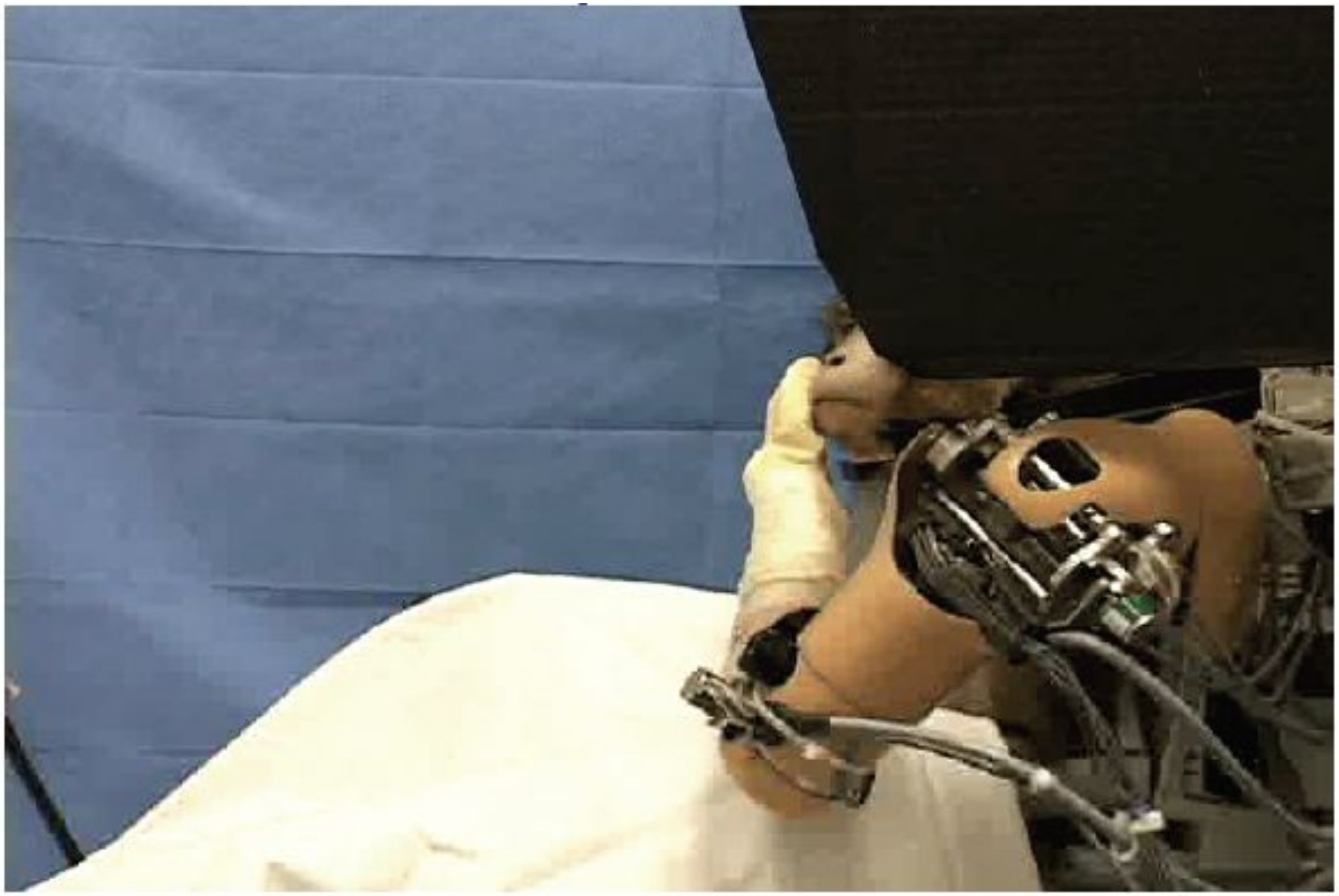
ALVINN drives 70 mph on highways



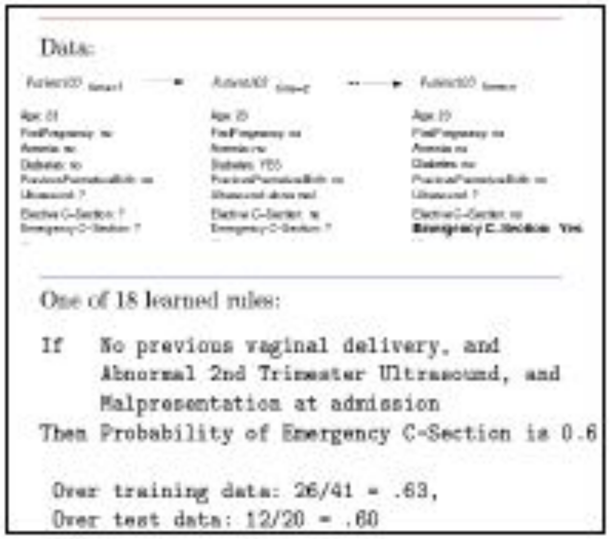


Learning prosthetic control from neural implant

R. Kass, L. Castlellanos, A. Schwartz



Machine Learning - Practice



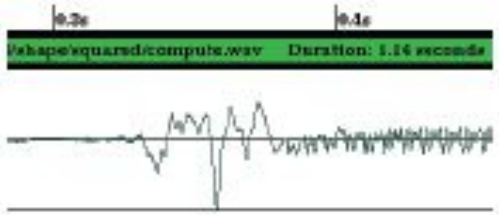
Data:

Patient103 time=1	Patient103 time=2	Patient103 time=n
Age: 23	Age: 23	Age: 23
FirstPregnancy: no	FirstPregnancy: no	FirstPregnancy: no
Anemia: no	Anemia: no	Anemia: no
Diabetes: no	Diabetes: YES	Diabetes: no
PreviousPrematureBirth: no	PreviousPrematureBirth: no	PreviousPrematureBirth: no
Ultrasound: ?	Ultrasound: abnormal	Ultrasound: ?
Elective C-Section: ?	Elective C-Section: no	Elective C-Section: no
Emergency-C-Section: ?	Emergency-C-Section: ?	**Emergency C-Section: Yes**

One of 18 learned rules:

```
If   No previous vaginal delivery, and
     Abnormal 2nd Trimester Ultrasound, and
     Malpresentation at admission
Then Probability of Emergency C-Section is 0.6

 Over training data: 26/41 = .63,
 Over test data: 12/20 = .60
```

Mining Databases

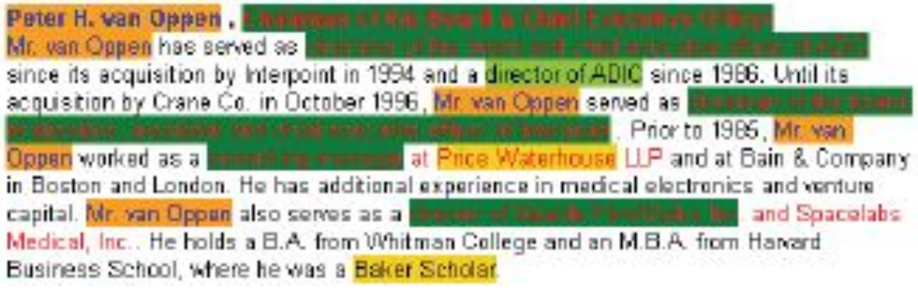


Speech Recognition

Control learning

Object recognition

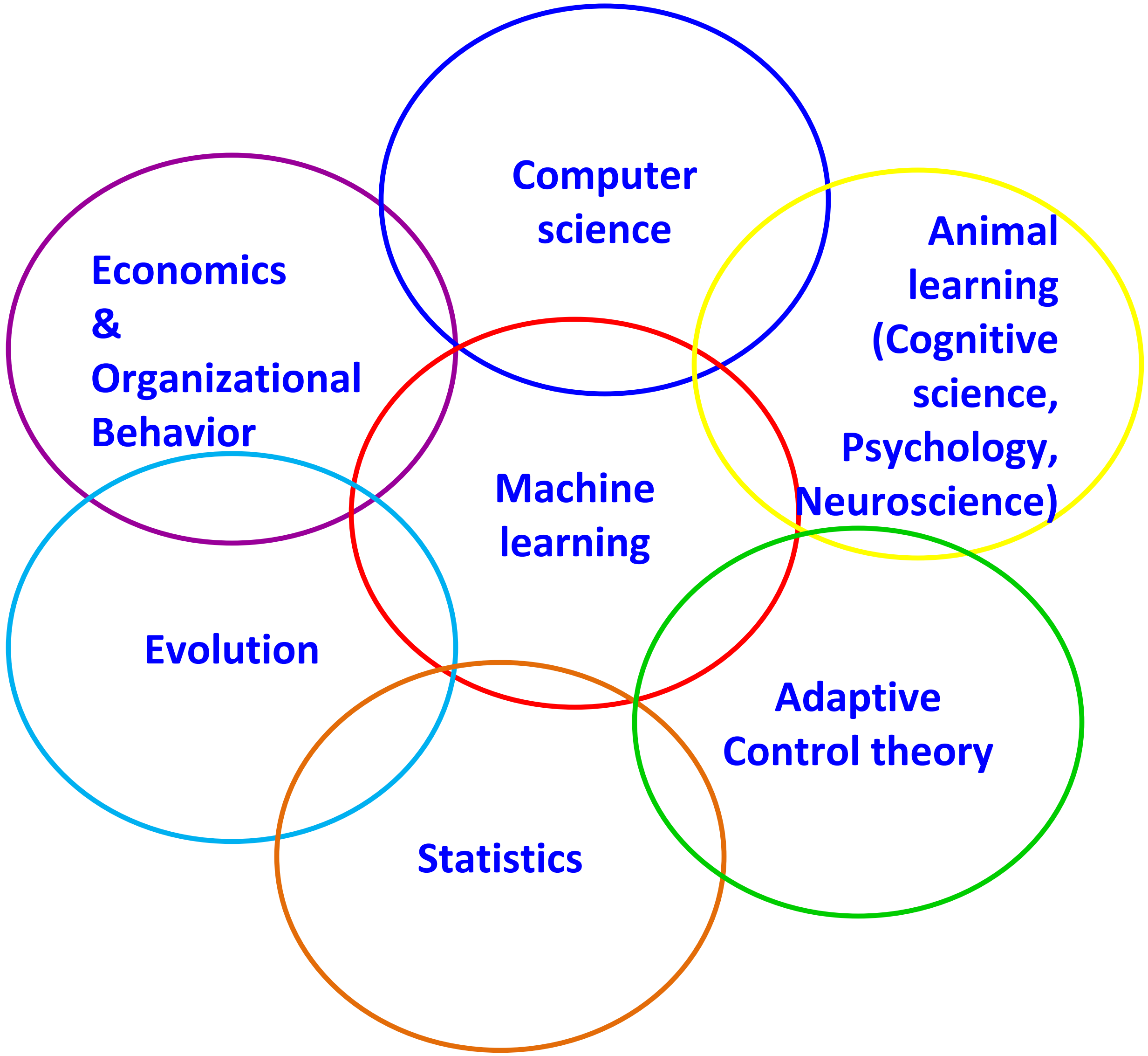
Text analysis



Peter H. van Oppen , Chairman of the Board & Chief Executive Officer
Mr. van Oppen has served as [illegible]
since its acquisition by Interpoint in 1994 and a director of ADIC since 1986. Until its acquisition by Crane Co. in October 1996, Mr. van Oppen served as [illegible]. Prior to 1985, Mr. van Oppen worked as a [illegible] at Price Waterhouse LLP and at Bain & Company in Boston and London. He has additional experience in medical electronics and venture capital. Mr. van Oppen also serves as a [illegible], and Spacelabs Medical, Inc.. He holds a B.A. from Whitman College and an M.B.A. from Harvard Business School, where he was a Baker Scholar

- Supervised learning
  - Bayesian networks
  - Hidden Markov models
  - Unsupervised clustering
  - Reinforcement learning
  - ....

Related Disciplines


Computer science
Economics & Organizational Behavior
Animal learning (Cognitive science, Psychology, Neuroscience)
Machine learning
Evolution
Adaptive Control theory
Statistics

ML niche is growing (Why)?

- ✔ Improved machine learning algorithms
- ✔ Increased data capture, networking, new sensors
- ✔ Software too complex to write by hand
- ✔ Demand for self-customization to user, environment

The 10 Algorithms Machine Learning Engineers Need to Know

- http://www.kdnuggets.com/2016/08/10-algorithms-machine-learning-engineers.html

The 10 Most Innovative Companies In AI/Machine Learning 2017

- https://www.fastcompany.com/3069025/the-10-most-innovative-companies-in-ai-machine-learning-2017

Designing a Learning System

Example: A Checker Learning Problem

1. Problem Description
2. Choosing the Training Experience
3. Choosing the Target Function
4. Choosing a Representation for the Target Function
5. Choosing a Function Approximation Algorithm
6. Final Design

1. Problem Description

- **Task T:** Playing Checkers
- **Performance Measure P:** % of games won against opponents
- **Training Experience E:** To be selected ==> Games Played against itself

2. Choosing the Training Experience

- **Direct versus Indirect Experience** **[Indirect** Experience gives rise to the **credit assignment** problem & is thus more difficult]
- **Teacher versus Learner Controlled Experience** [the teacher might provide training examples; the learner might suggest *interesting* examples & ask the teacher for their outcome; or the learner can be completely on its own with no access to correct outcomes]
- **How Representative is the Experience?** [Is the training experience representative of the task the system will actually have to solve? It is best if it is, but such a situation cannot systematically be achieved]

3. Choosing the Target Function

- Given a set of legal moves, we want to learn how to choose the best move [since the best move is not necessarily known, this is an *optimization* problem]
- *ChooseMove*: B --> M is called a **Target Function** [*ChooseMove*, however, is difficult to learn. An easier & related target function to learn is *V*: B --> R, which assigns a numerical score to each board. The better the board, the higher the score.]
- **Operational vs. Non-Operational Description of a Target Function** [An operational description must be given]
- **Function Approximation** [The actual function can often not be learned & must be approximated]

4. Choosing a Representation for the Target Function

- **Expressiveness vs. Training set size** [The more expressive the representation of the target function, the closer to the "truth" we can get. However, the more expressive the representation, the more training examples are necessary to choose among the large number of "representable" possibilities.]
- **Example of a representation:**
  - x_1/x_2 = # of black/red pieces on the board
  - x_3/x_4 = # of black/red king on the board
  - x_5/x_6 = # of black/red pieces threatened by red/black

w_i's are adjustable or "learnable" coefficients

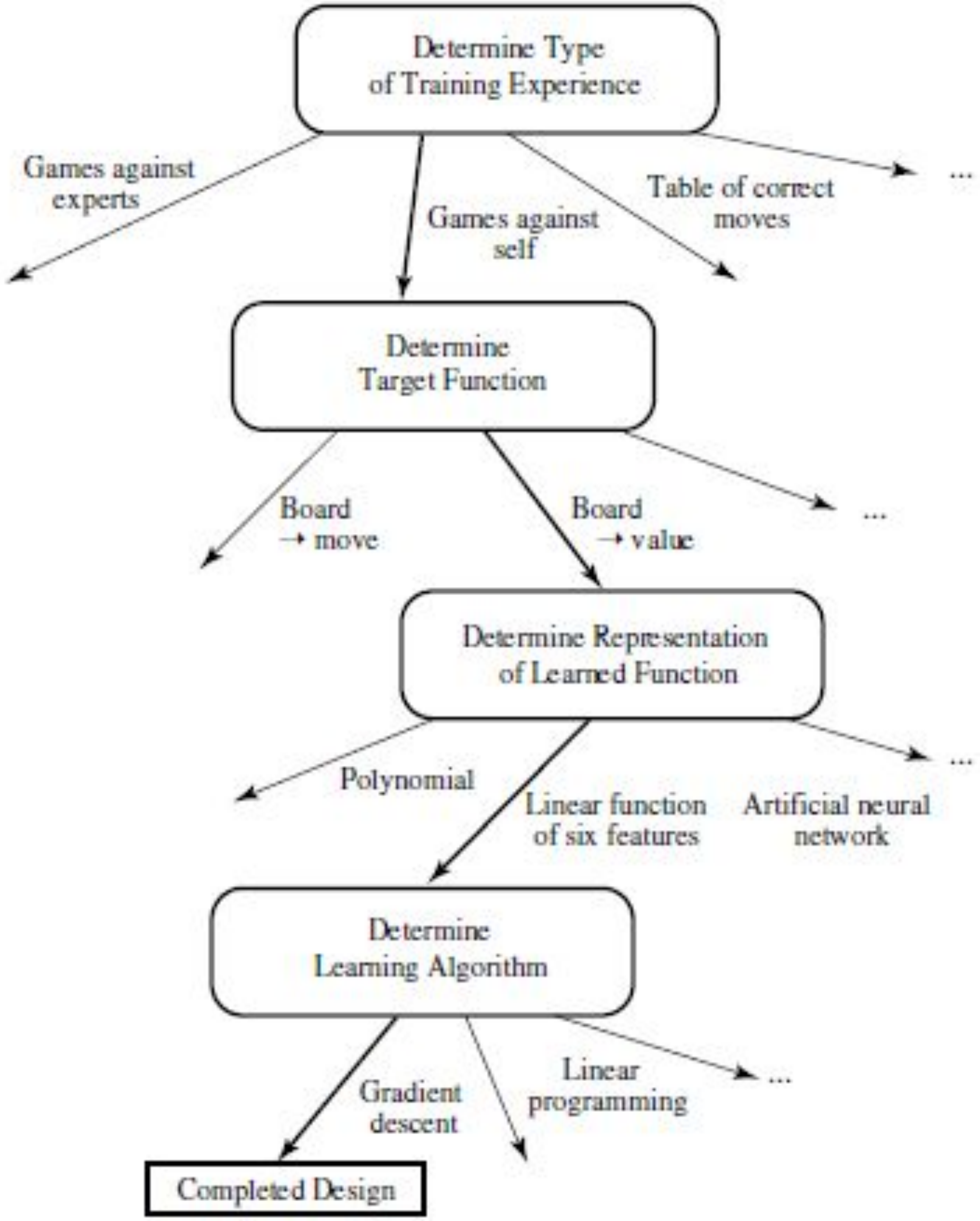
$$V(b) = w_0+w_1.x_1+w_2.x_2+w_3.x_3+w_4.x_4+w_5.x_5+w_6.x_6$$

5. Choosing a Function Approximation Algorithm

- **Generating Training Examples of the form <b,Vtrain(b)>** [e.g. <x_1=3, x_2=0, x_3=1, x_4=0, x_5=0, x_6=0, +100 (=blacks won)]
  - Useful & Easy Approach: V_{train}(b) <- V(Successor(b))
- **Training the System**
  - Defining a criterion for success [What is the error that needs to be minimized?]
  - Choose an algorithm capable of finding weights of a linear function that minimize that error [e.g. the Least Mean Square (LMS) training rule].

6. Final Design for Checkers Learning

- **The Performance Module**: Takes as input a new board and outputs a trace of the game it played against itself.
- **The Critic**: Takes as input the trace of a game and outputs a set of training examples of the target function
- **The Generalizer**: Takes as input training examples and outputs a *hypothesis* which estimates the target function. Good generalization to new cases is crucial.
- **The Experiment Generator**: Takes as input the current hypothesis (currently learned function) and outputs a new problem (an initial board state) for the performance system to explore


Determine Type of Training Experience
Games against experts
Games against self
Table of correct moves
...
Determine Target Function
Board → move
Board → value
...
Determine Representation of Learned Function
Polynomial
Linear function of six features
Artificial neural network
...
Determine Learning Algorithm
Gradient descent
Linear programming
...
Completed Design

Learning Algorithms

Learning styles:

- ❑ **Supervised Learning**:
  - Input data is called training data & has a known label or result

-such as spam/not-spam or a stock price at a time.

  - A model is prepared through a training process where it is required to make predictions & is corrected when those predictions are wrong.
  - The training process continues until the model achieves a desired level of accuracy on the training data.

Learning Algorithms

- **Unsupervised Learning:**
  - Input data is not labeled & does not have a known result.
  - A model is prepared by deducing structures/patterns present in the input data.

Learning Algorithms

- **Semi-Supervised Learning:**
  - Input data is a mixture of labeled & unlabelled examples.
  - There is a desired prediction problem but the model must learn the structures to organize the data as well as make predictions.

Learning Algorithms

- **Reinforcement Learning**
  - Input data is provided as stimulus to a model from an environment to which the model must respond & react.
  - Feedback is provided not from of a teaching process as in supervised learning, but as punishments & rewards in the environment

Learning Algorithms

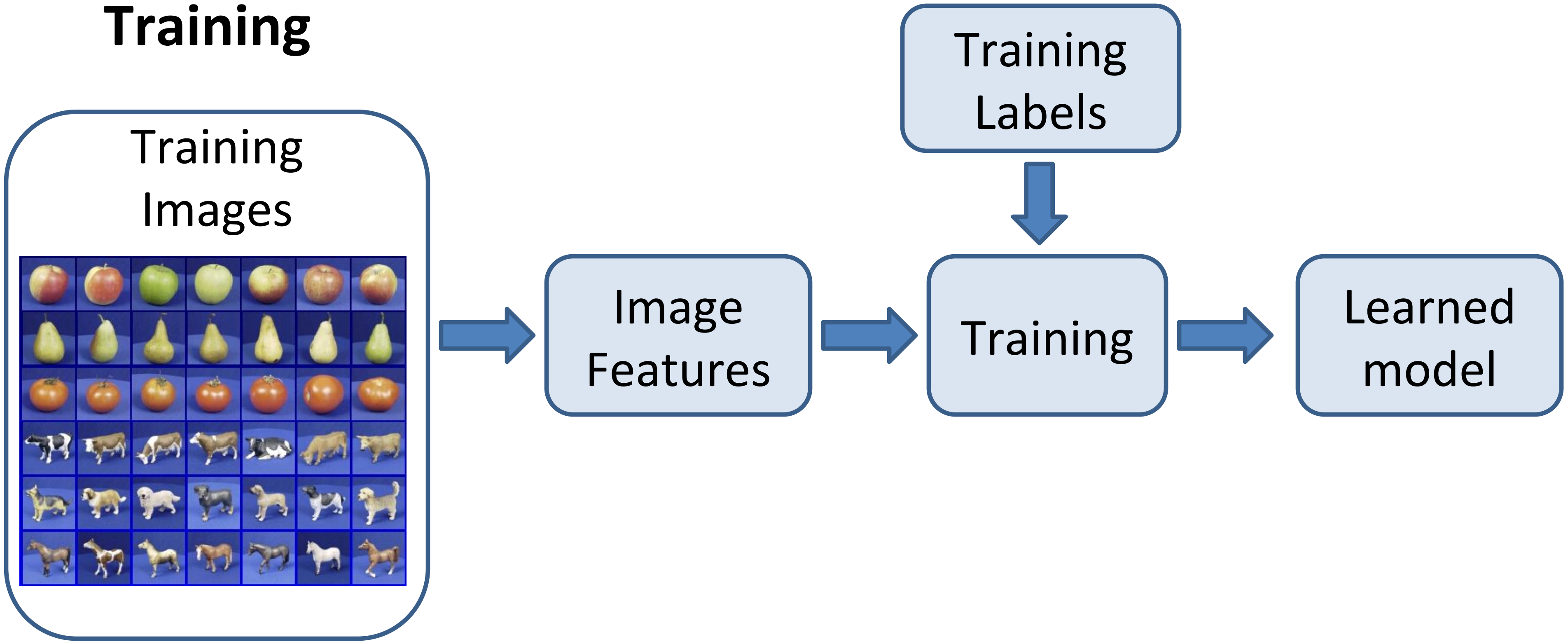
Supervised	Unsupervised	Reinforcement
Classification k-Nearest Neighbour (kNN) Decision tree Random Forest Naive Bayes BBN SVM LDA Neural nets Adaboost Deep learning **Regression** Ordinary Least Squares Logistic Regression Stepwise Regression Multivariate Adaptive Regression Splines (MARS)	**Clustering** BIRCH Hierarchical k-means EM DBSCAN OPTICS Mean-shift **Association rule learning** Apriori algorithm Eclat algorithm	Q-learning Temporal difference learning

Classification

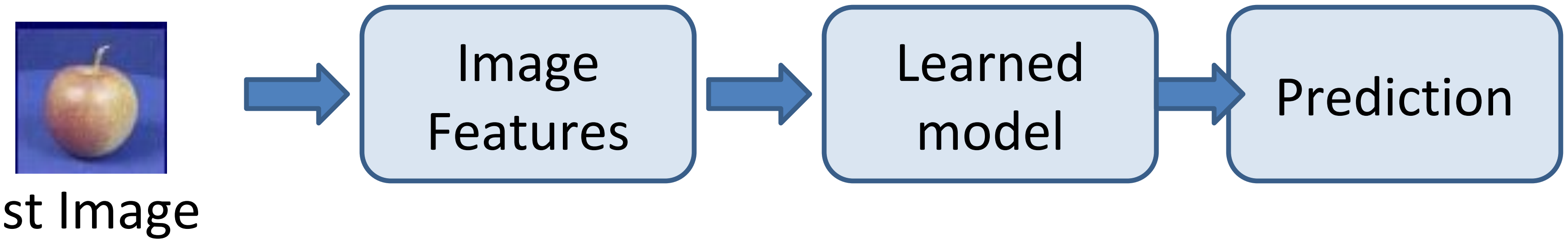
- A bank loans officer needs analysis of her data to learn which loan applicants are "safe" & which are "risky" for the bank.
- A marketing manager at *AllElectronics needs data* analysis to help guess whether a customer with a given profile will buy a new computer [**yes or no**].
- A medical researcher wants to analyze breast cancer data to predict which one of three specific treatments a patient should receive [**A, B, or C**]

Steps

Training



Testing



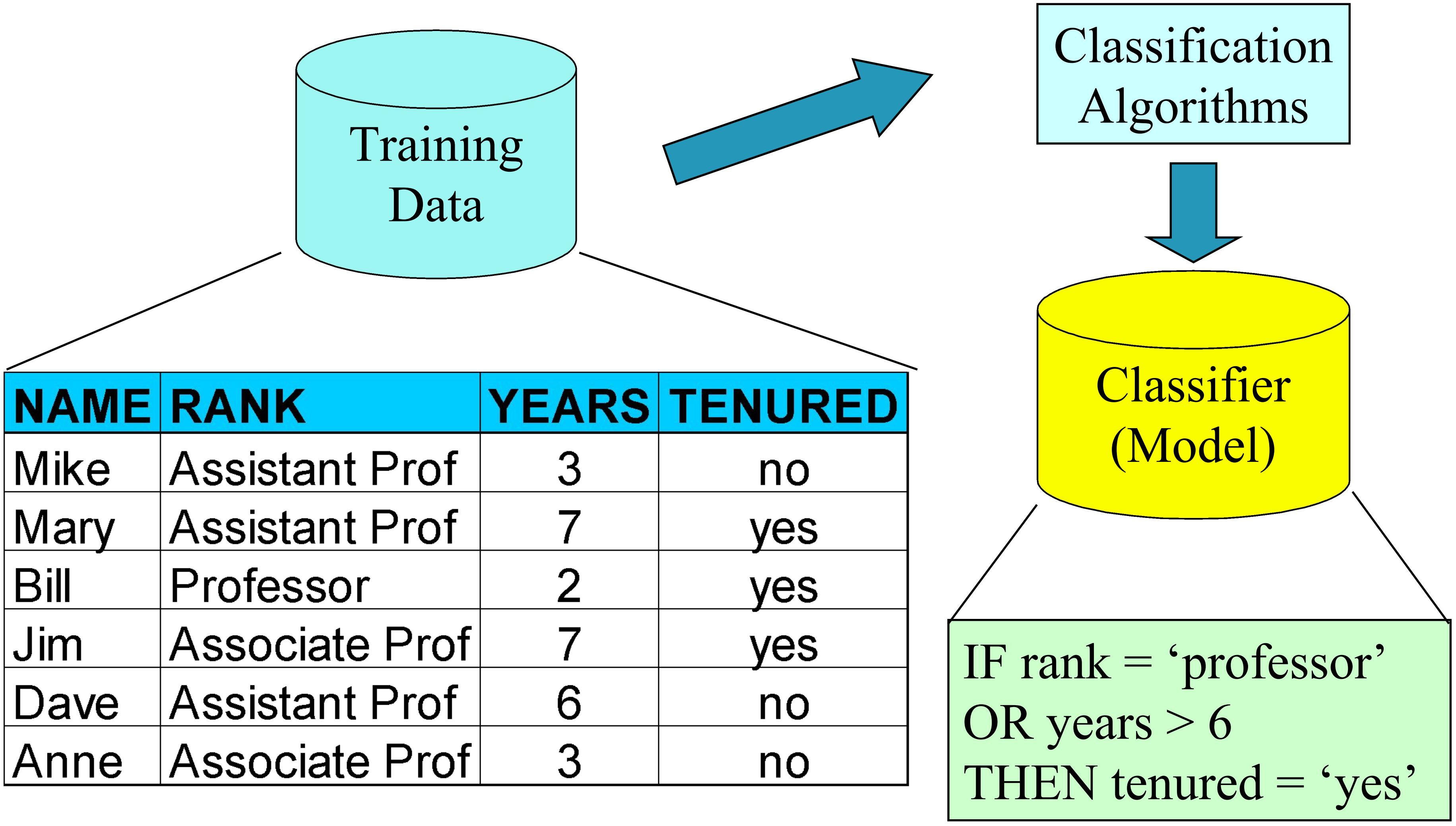
How does classification work?

- Two-Step Process :
- *learning step (where a classification model is constructed)*

- describing a set of predetermined classes
  - Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute
  - The set of tuples used for model construction is training set
  - The model is represented as classification rules, decision trees, or mathematical formulae

- *classification step (where* the model is used to predict class labels for given data).
- for classifying future or unknown objects
  - Estimate accuracy of the model
    - The known label of test sample is compared with the classified result from the model
    - Accuracy rate is the percentage of test set samples that are correctly classified by the model
    - Test set is independent of training set (otherwise overfitting)
  - If the accuracy is acceptable, use the model to classify new data

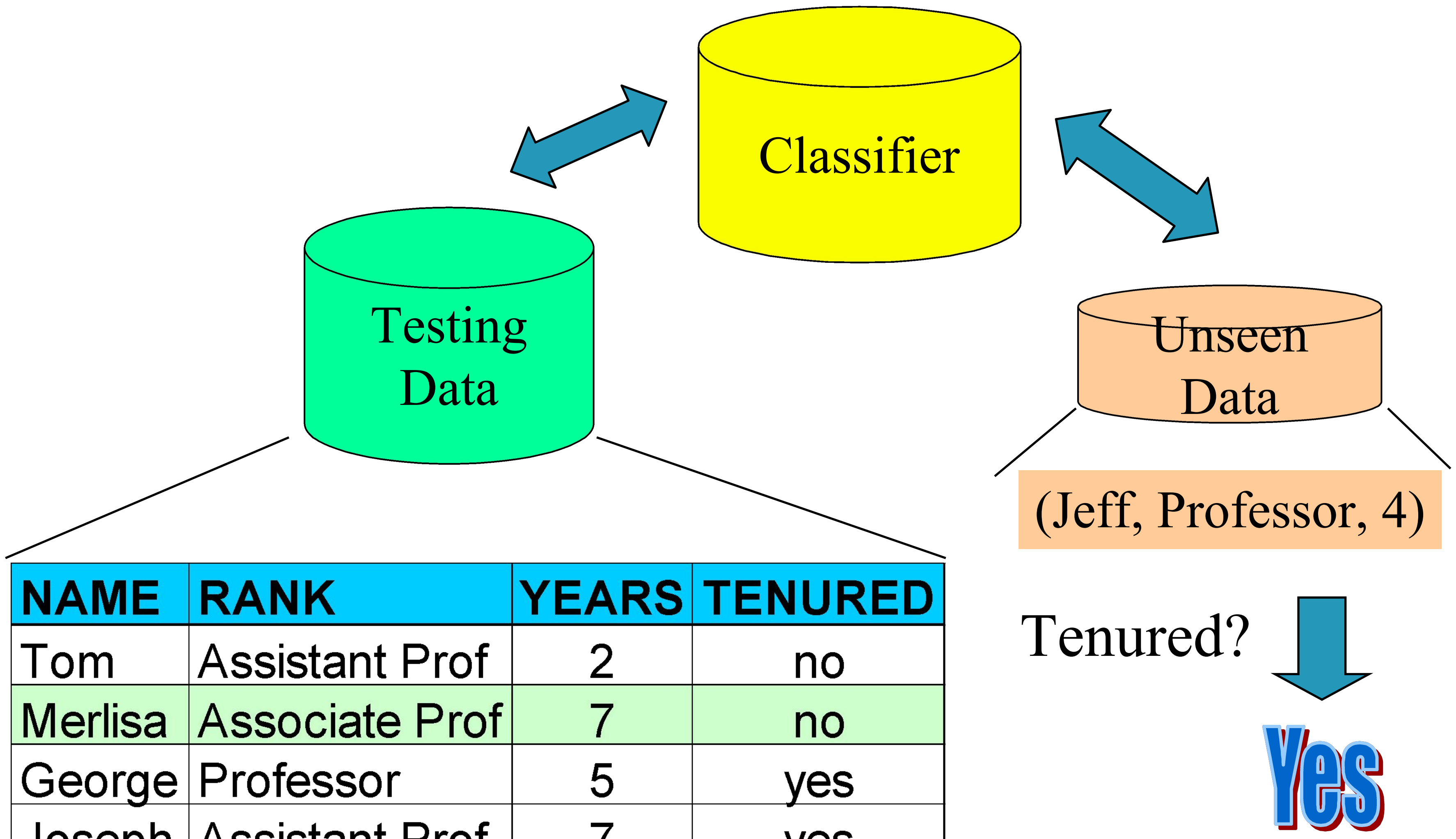
Process (1): Model Construction



NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

IF rank = 'professor'
OR years > 6
THEN tenured = 'yes'

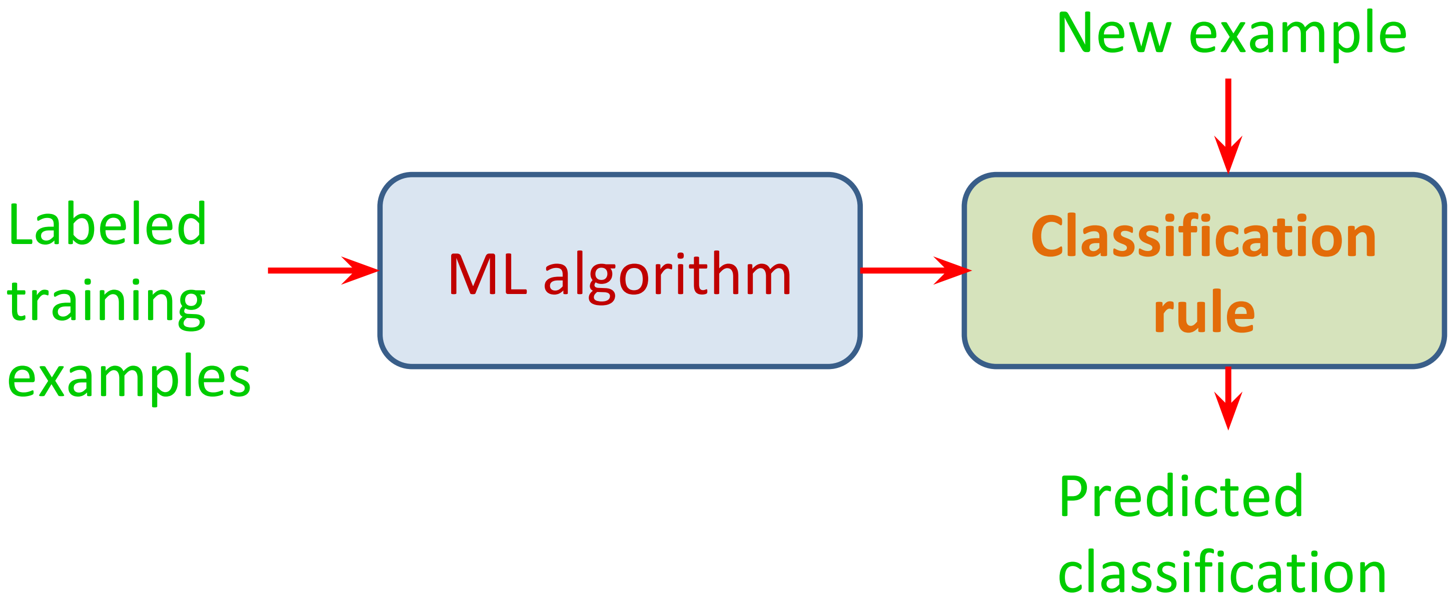
Process (2): Using the Model in Prediction



NAME	RANK	YEARS	TENURED
Tom	Assistant Prof	2	no
Merlisa	Associate Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes

Classification Problems

- classify examples into given set of categories

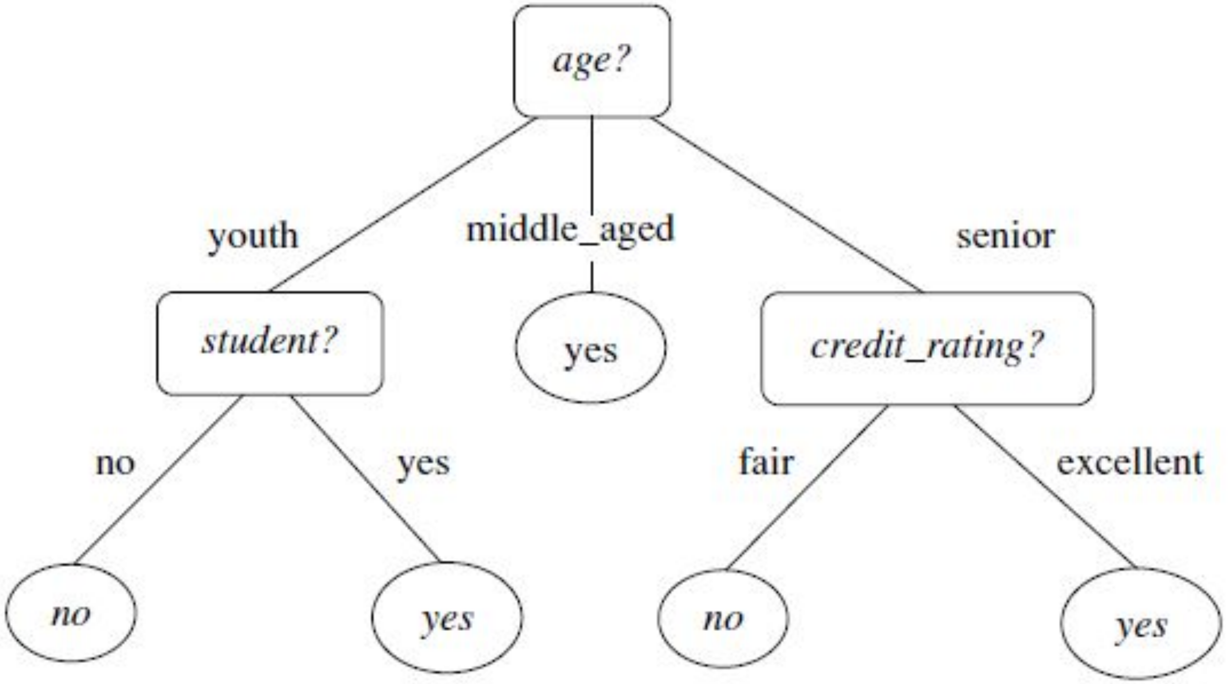


Decision Trees

Decision Tree Learning

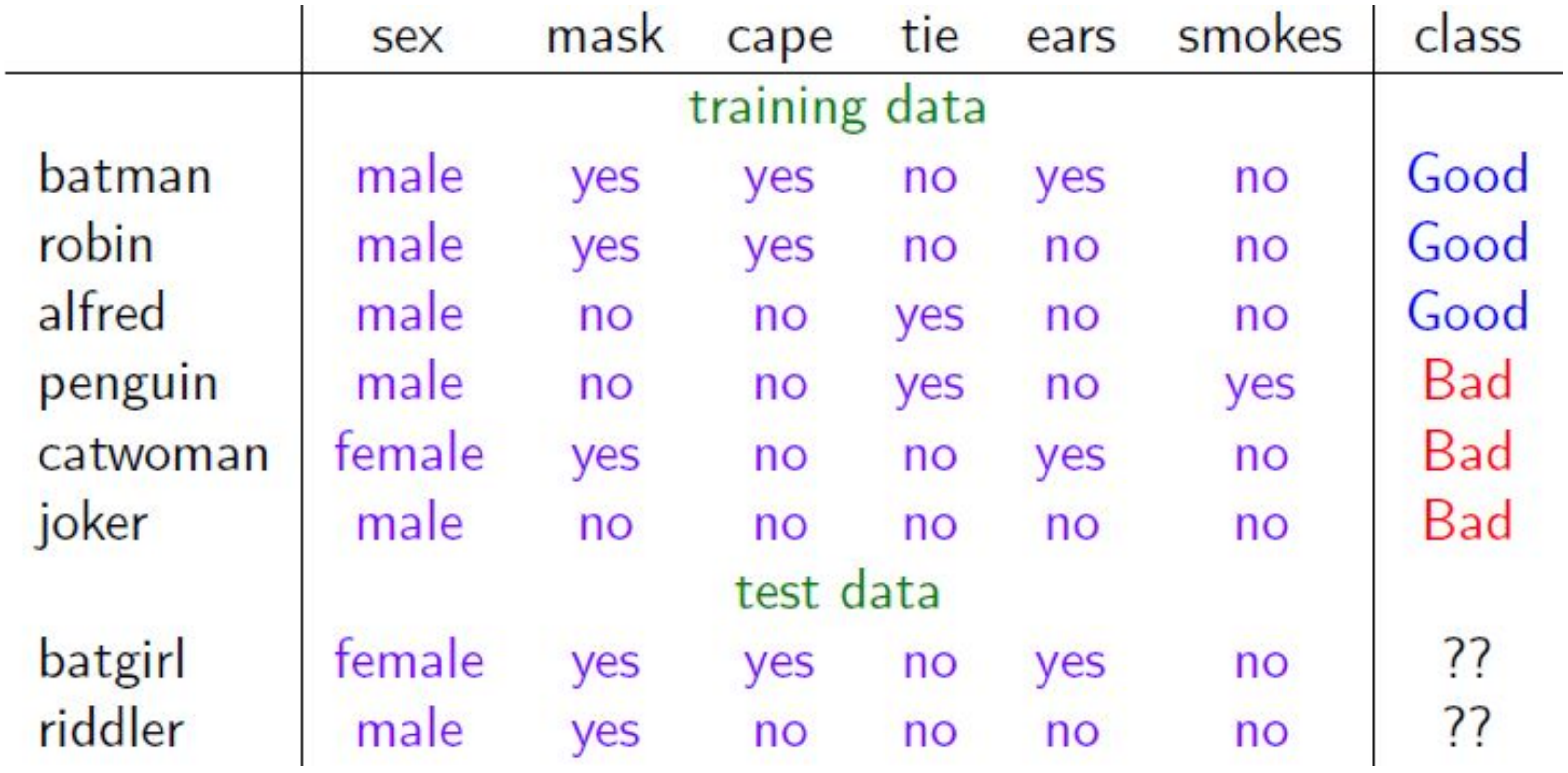
- **Decision tree induction is the learning of decision trees from class-labeled training** tuples.
- A **decision tree is a flowchart-like tree structure, where each internal node (nonleaf** node) denotes a test on an attribute, each **branch represents an outcome of the** test, and each **leaf node (or *terminal node) holds a class label.***
- ***The topmost node in*** a tree is the **root node.**

A decision tree for the concept *buys_computer, indicating whether an AllElectronics customer* is likely to purchase a computer. Each internal (nonleaf) node represents a test on an attribute. Each leaf node represents a class (either *buys_computer = yes or buys_computer = no).*



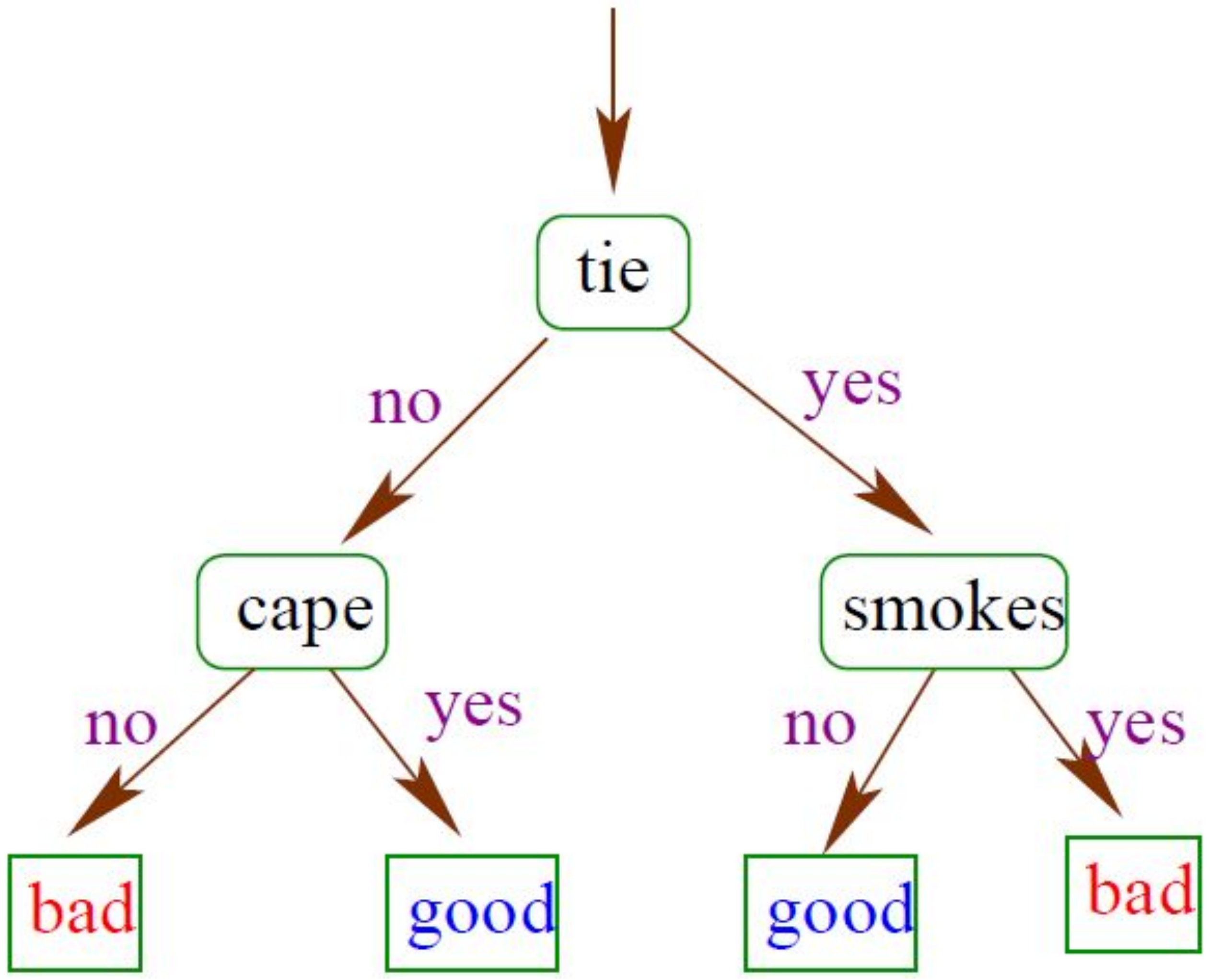
Example: Good versus Evil

- **problem**: identify people as good or bad from their appearance



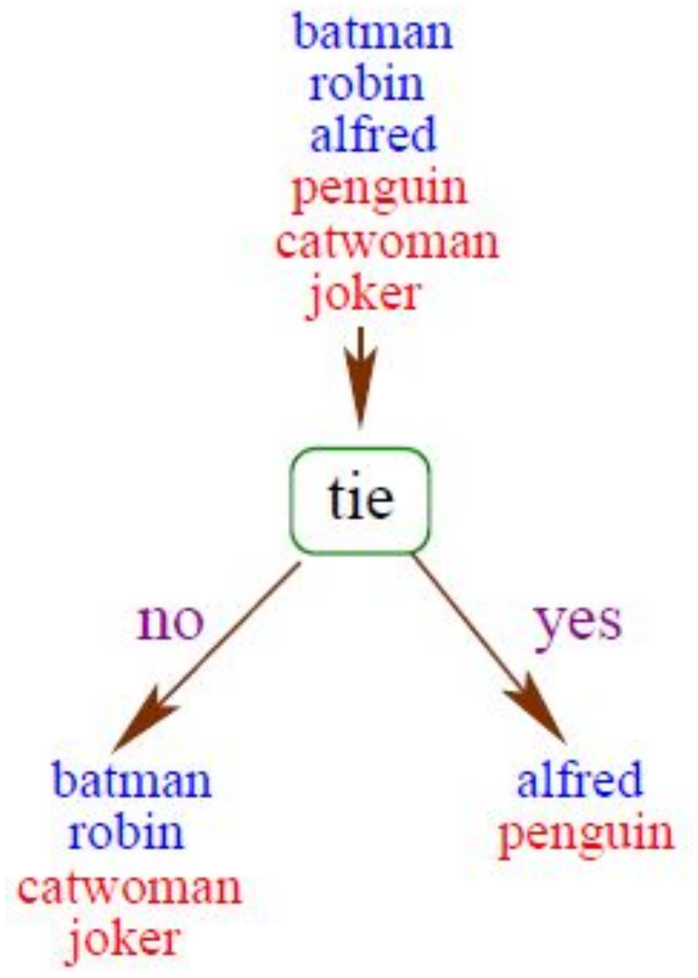
	sex	mask	cape	tie	ears	smokes	class
			training data				
batman	male	yes	yes	no	yes	no	Good
robin	male	yes	yes	no	no	no	Good
alfred	male	no	no	yes	no	no	Good
penguin	male	no	no	yes	no	yes	Bad
catwoman	female	yes	no	no	yes	no	Bad
joker	male	no	no	no	no	no	Bad
			test data				
batgirl	female	yes	yes	no	yes	no	??
riddler	male	yes	no	no	no	no	??

A Decision Tree Classifier



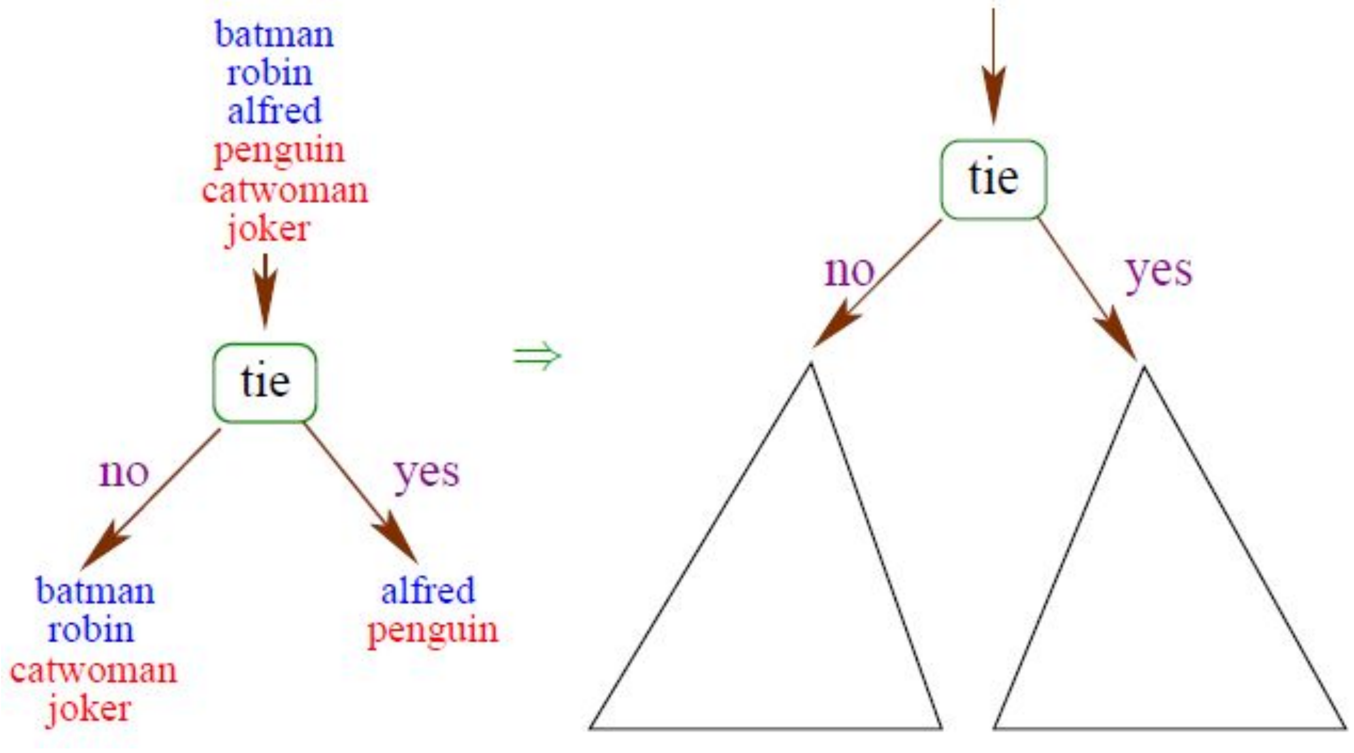
How to Build Decision Trees

- choose rule to split on
- divide data using splitting rule into disjoint subsets



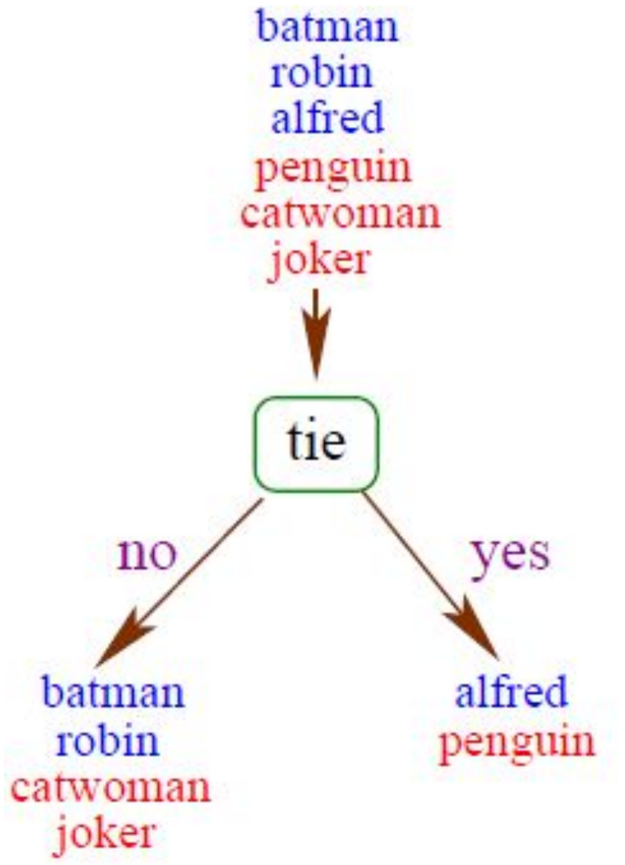
How to Build Decision Trees

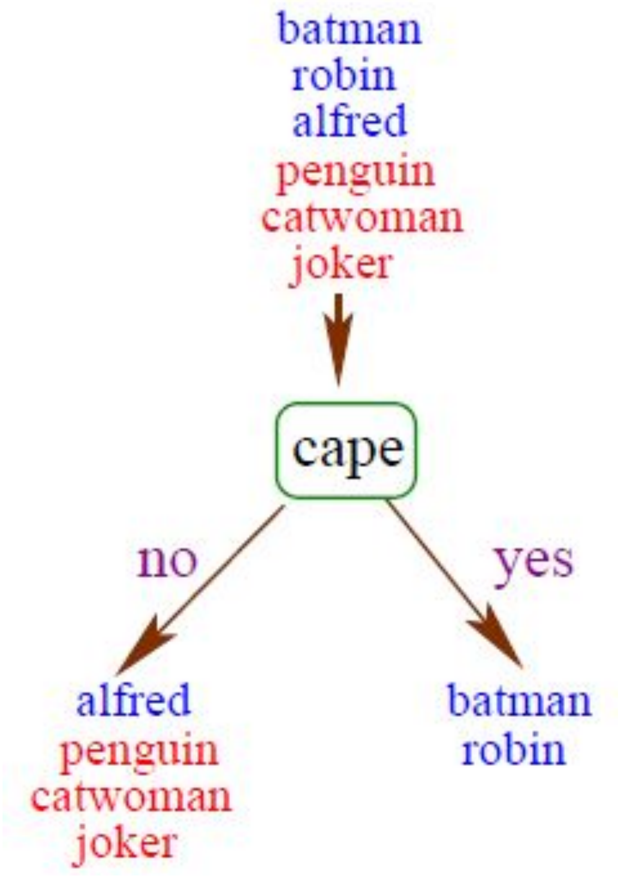
- choose rule to split on
- divide data using splitting rule into disjoint subsets
- repeat recursively for each subset
- stop when leaves are (almost) "pure"



How to Choose the Splitting Rule

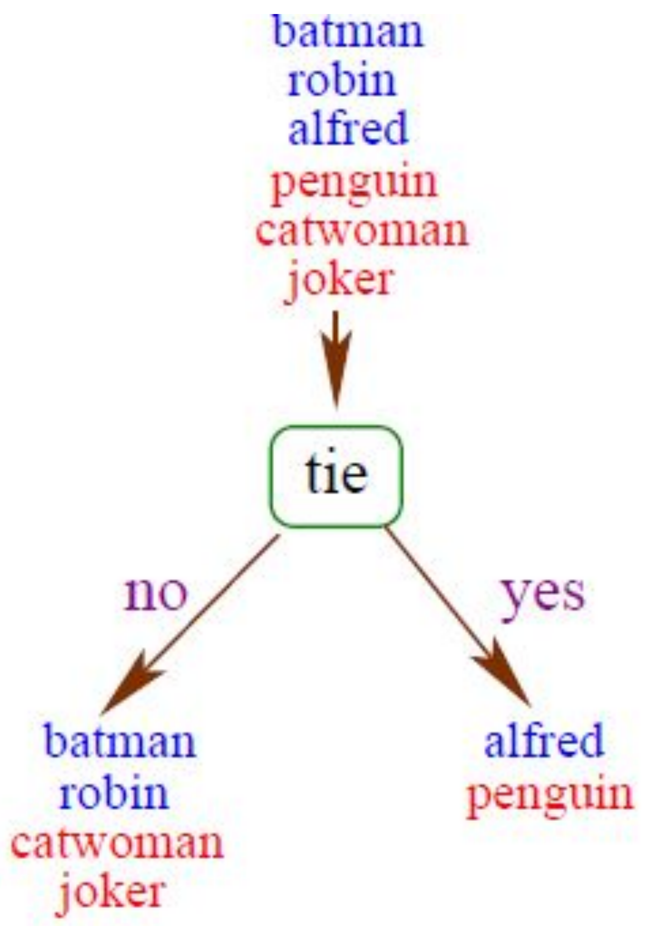
- key problem: choosing best rule to split on:

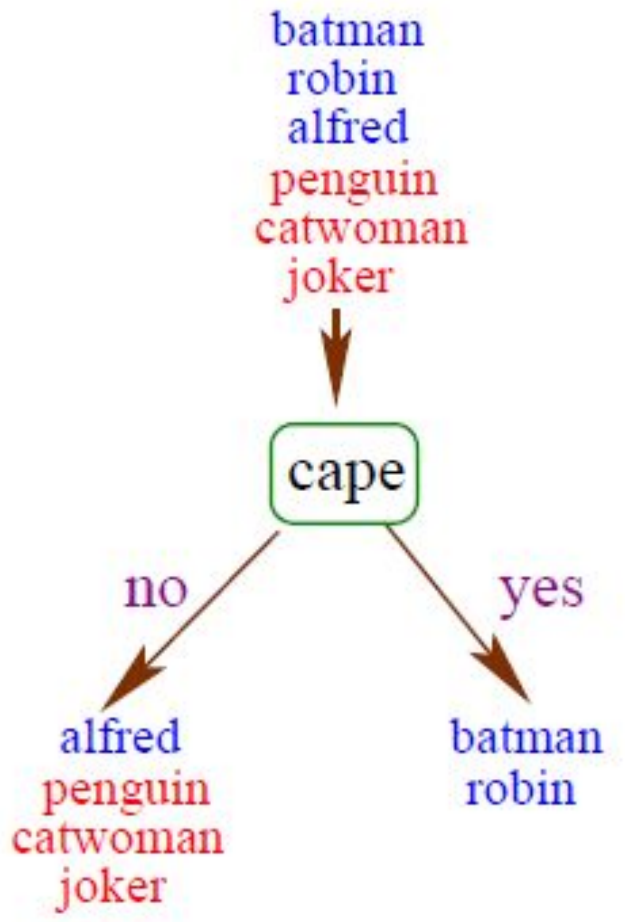




How to Choose the Splitting Rule

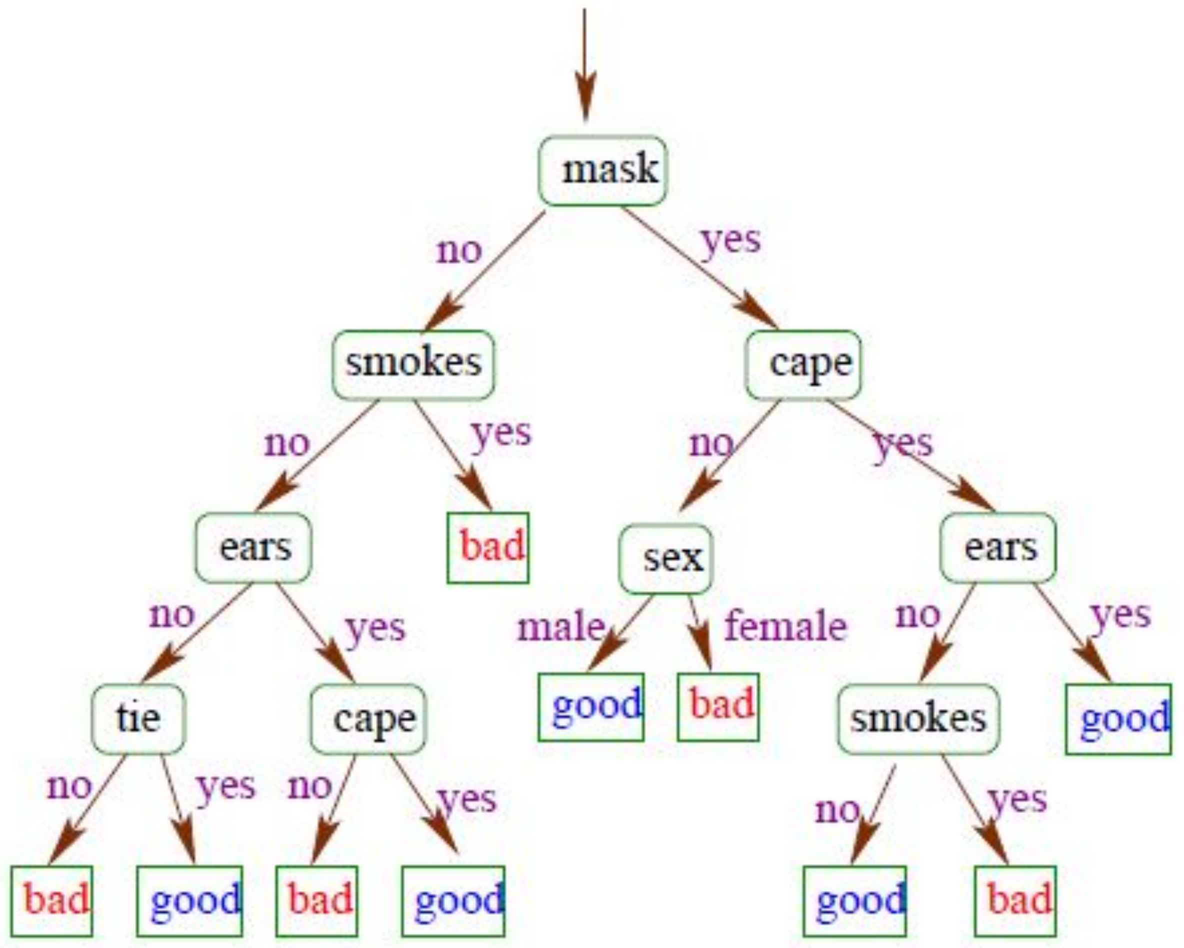
key problem: choosing best rule to split on:





idea: choose rule that leads to greatest increase in “purity”

A Possible Classifier



How to Measure Purity

- Information gain
- Gini Index
- Gain Ratio

Information Gain

- Expected information (entropy) needed to classify a tuple in D

$$\boxed{Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)}$$

- Information needed (after using A to split D into v partitions) to classify D:

$$\boxed{Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)}$$

- Information gained by branching on attribute A

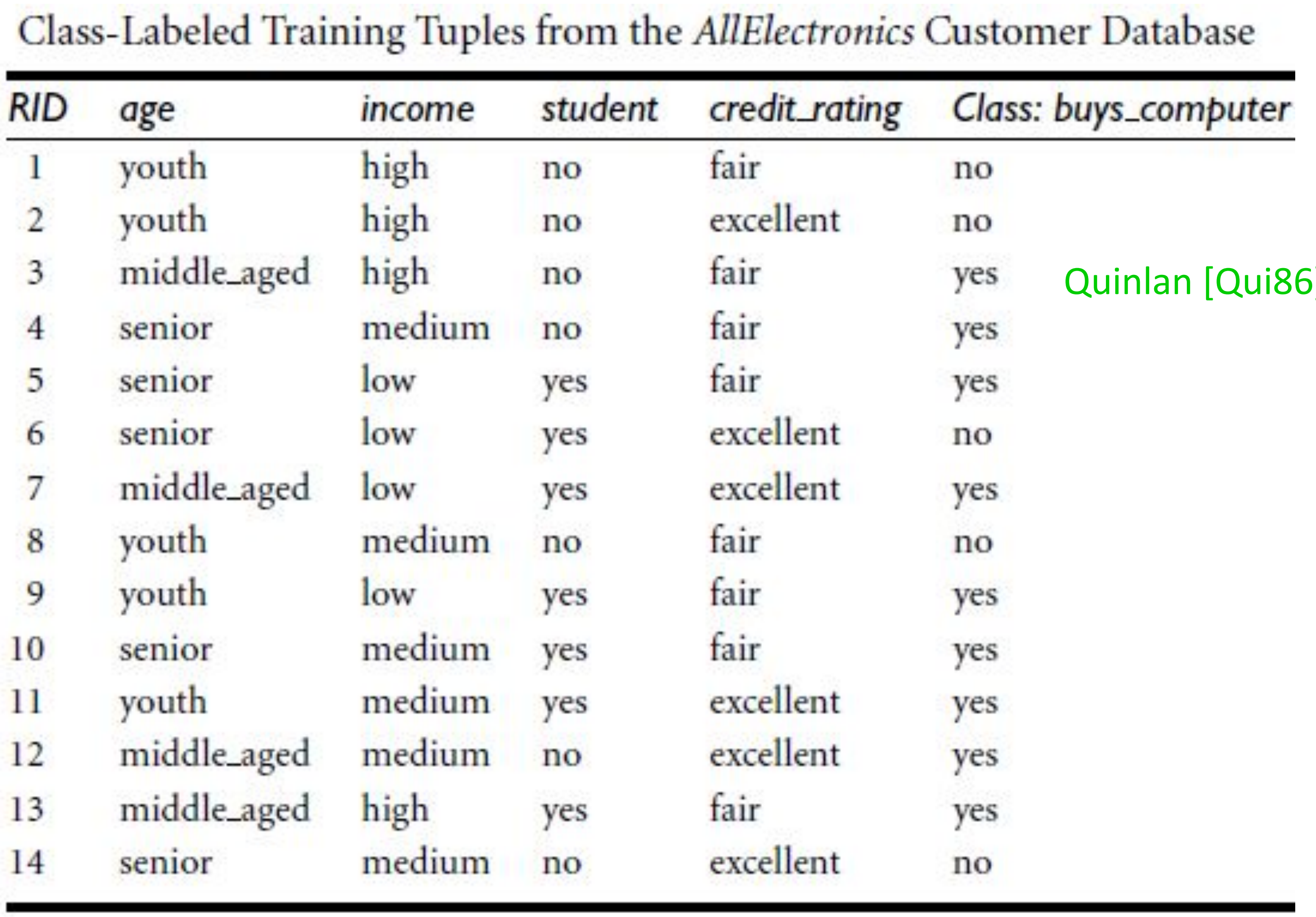
$$\boxed{Gain(A) = Info(D) - Info_A(D)}$$

$$\boxed{Gain(A) = Info(D) - Info_A(D)}$$

- **Gain(A)** tells us how much would be gained by branching on A

- The attribute A with the highest **information gain, Gain (A),** is chosen as the splitting attribute at node N.

An Illustrative Example



Class-Labeled Training Tuples from the *AllElectronics* Customer Database

RID	*age*	*income*	*student*	*credit_rating*	*Class: buys_computer*
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Quinlan [Qui86].

How to choose best splitting criterion?

- The class label attribute, buys computer, has two distinct values (namely, {yes, no});
- two distinct classes (i.e., m = 2).
- class C_1 = yes
- class C_2 = no
- 09 tuples of class = yes
- 05 tuples of class = no
- A (root) node N is created for the tuples in D.
- To find the splitting criterion for these tuples, **compute the information gain** of each attribute.

- Expected information needed to classify a tuple in *D:*

$$\boxed{Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)}$$

$$Info(D) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.940 \text{ bits.}$$

- Next, we need to compute the expected information requirement for each attribute.
- Start with attribute: ***age***
- age category "youth,": yes = 02 tuples & no = 03 tuples.
- category "middle aged,": yes = 04 tuples & no = 0 tuples.
- category "senior,": *yes = 03 tuples & no = 02 tuples.*

- The expected information needed to classify a tuple in *D* if the tuples are partitioned according to **age:**

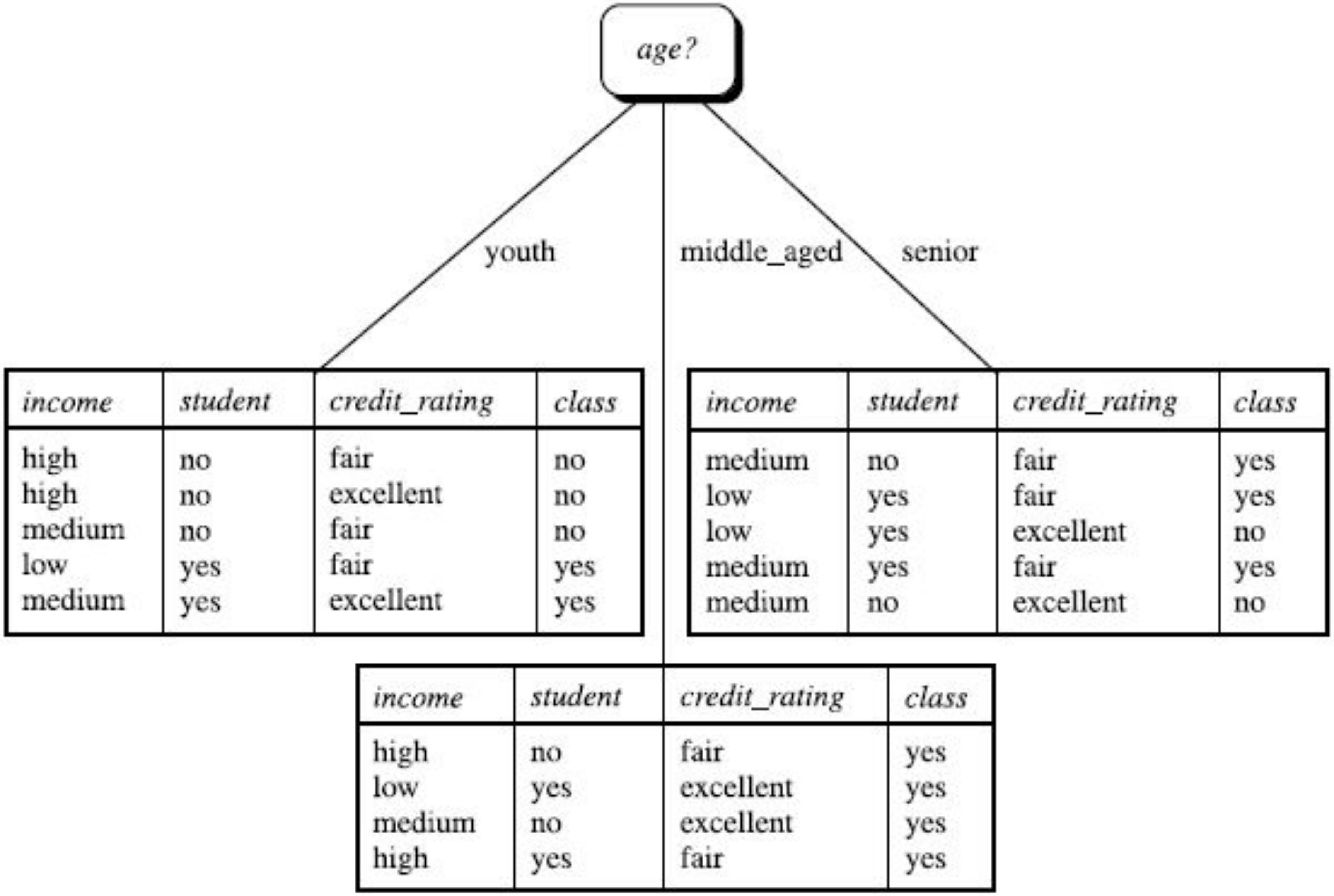
$$\boxed{Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)}$$

$$\begin{aligned} Info_{age}(D) &= \frac{5}{14} \times \left(-\frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log_2\frac{3}{5}\right) + \frac{4}{14} \times \left(-\frac{4}{4}\log_2\frac{4}{4}\right) \\ &\quad + \frac{5}{14} \times \left(-\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5}\right) \\ &= 0.694 \text{ bits.} \end{aligned}$$

- Hence, the gain in information from such a partitioning would be

$$\boxed{Gain(age) = Info(D) - Info_{age}(D) = 0.940 - 0.694 = 0.246 \text{ bits.}}$$

Similarly, we can compute Gain(income) = 0.029 bits, Gain(student) = 0.151 bits, and Gain(credit_rating) = 0.048 bits.
Because age has the highest information gain among the attributes, it is selected as the splitting attribute.



income	*student*	*credit_rating*	*class*
high	no	fair	no
high	no	excellent	no
medium	no	fair	no
low	yes	fair	yes
medium	yes	excellent	yes

income	*student*	*credit_rating*	*class*
medium	no	fair	yes
low	yes	fair	yes
low	yes	excellent	no
medium	yes	fair	yes
medium	no	excellent	no

income	*student*	*credit_rating*	*class*
high	no	fair	yes
low	yes	excellent	yes
medium	no	excellent	yes
high	yes	fair	yes

Gain Ratio

- The information gain measure is **biased** toward tests with many outcomes.
- That is, it prefers to select attributes having a large number of values.
- For example, consider an attribute that acts as a unique identifier such as *product_ID.*
- *A split on product_ID would* result in a large number of partitions (as many as there are values), each one containing just one tuple.
- Because each partition is pure, the information required to classify data set *D based on this partitioning would be* ***$Info_{product\_ID}(D) = 0$****.*

 information gained by partitioning on this attribute is maximal.

- Such a partitioning is **useless f**or classification.

- C4.5, a successor of ID3, uses an extension to information gain known as ***gain ratio***, which attempts to overcome this bias.
- It applies a kind of normalization to information gain using a "**split information**" value defined analogously with *Info(D):*

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

This value represents the potential information generated by splitting the training data set, *D, into v partitions, corresponding to the v outcomes of a test on attribute A.*

- The gain ratio is defined as

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo_A(D)}.$$

- The attribute with the **maximum gain ratio** is selected as the splitting attribute.
- Note, however, that as the split information approaches 0, the ratio becomes unstable.

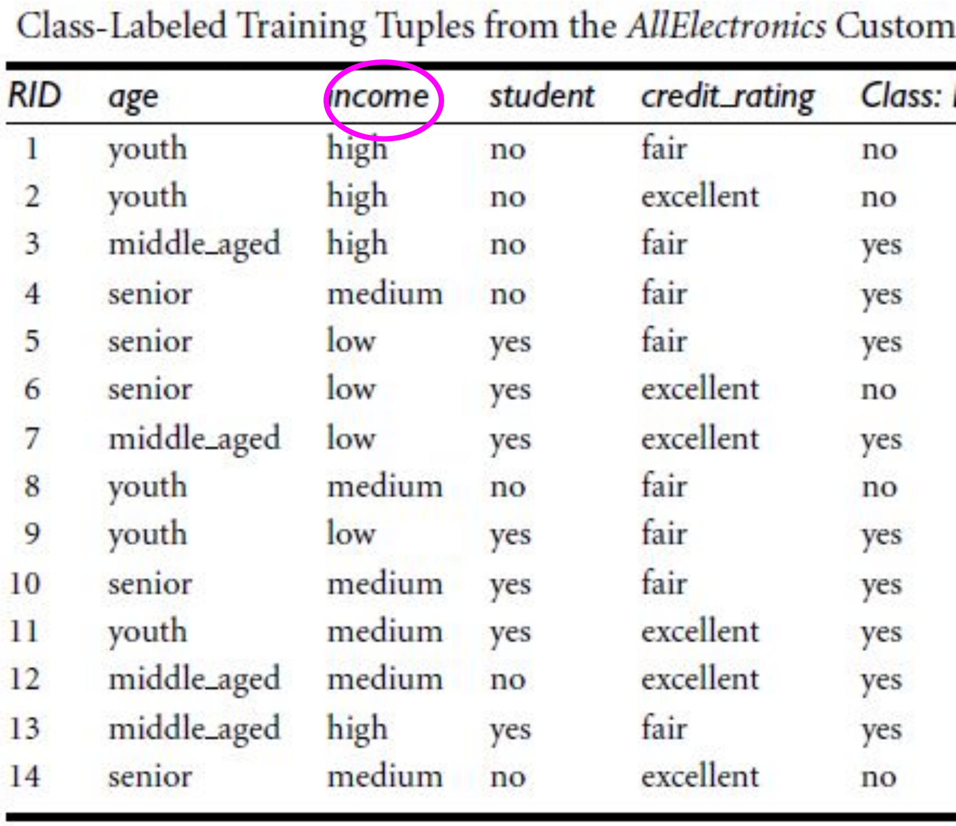
Example: Computation of gain ratio for the attribute *income*

- A test on ***income*** *splits the data of* into 03 partitions:

✔ *Low = 04 tuples*

✔ *Medium = 06 tuples*

✔ *High = 04 tuples*



Class-Labeled Training Tuples from the *AllElectronics* Custom

RID	*age*	*income*	*student*	*credit_rating*	*Class:*
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

$$SplitInfo_{income}(D) = -\frac{4}{14} \times \log_2\left(\frac{4}{14}\right) - \frac{6}{14} \times \log_2\left(\frac{6}{14}\right) - \frac{4}{14} \times \log_2\left(\frac{4}{14}\right)$$

$$= 1.557.$$

$$Gain(age) = Info(D) - Info_{age}(D)$$

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo_A(D)}.$$

Gain(income) = 0.029

[See previous example]

$$= \frac{0.029}{1.557}$$

$$= 0.019$$

Gini Index

- The Gini index is used in CART (Classification & Regression Tree).
- Gini index measures the impurity of *D, a data partition or set of training tuples*

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2,$$

- where *p_i is the probability that a tuple in D belongs to class C_i and is estimated by $|C_{i,D}|/|D|$.*
- *The sum is computed over m classes.*

- The Gini index considers a binary split for each attribute.
- To determine the best binary split on *A, we examine all the possible subsets* that can be formed using known values of *A.*
- Each subset, S_A*, can be considered as a* binary test for attribute *A of the form "*$A \in S_A$*?"*
- *Given a tuple, this test is satisfied if* the value of *A for the tuple is among the values listed in* S_A.
- *If A has v possible values,* then there are 2^v *possible subsets.*

- For example, if ***income*** *has three possible values: low, medium, high,*
- *possible subsets are {low, medium, high}, {low, medium}, {low, high}, {medium, high}, {low}, {medium}, {high}, and {}.*
- *We exclude the* power set, {*low, medium, high}, and the empty set from consideration since, conceptually,* they do not represent a split.
- Therefore, there are 2^v *-2 possible ways to form two* partitions of the data, *D, based on a binary split on A.*

- When considering a binary split, we compute a weighted sum of the impurity of each resulting partition.
- For example, if a binary split on *A partitions D into D_1 and D_2, the* Gini index of *D given that partitioning is*

$$Gini_A(D) = \frac{|D_1|}{|D|}Gini(D_1) + \frac{|D_2|}{|D|}Gini(D_2).$$

For each attribute, each of the possible binary splits is considered.

For a discrete-valued attribute, the subset that gives the **minimum Gini index for that attribute is selected as its splitting subset.**

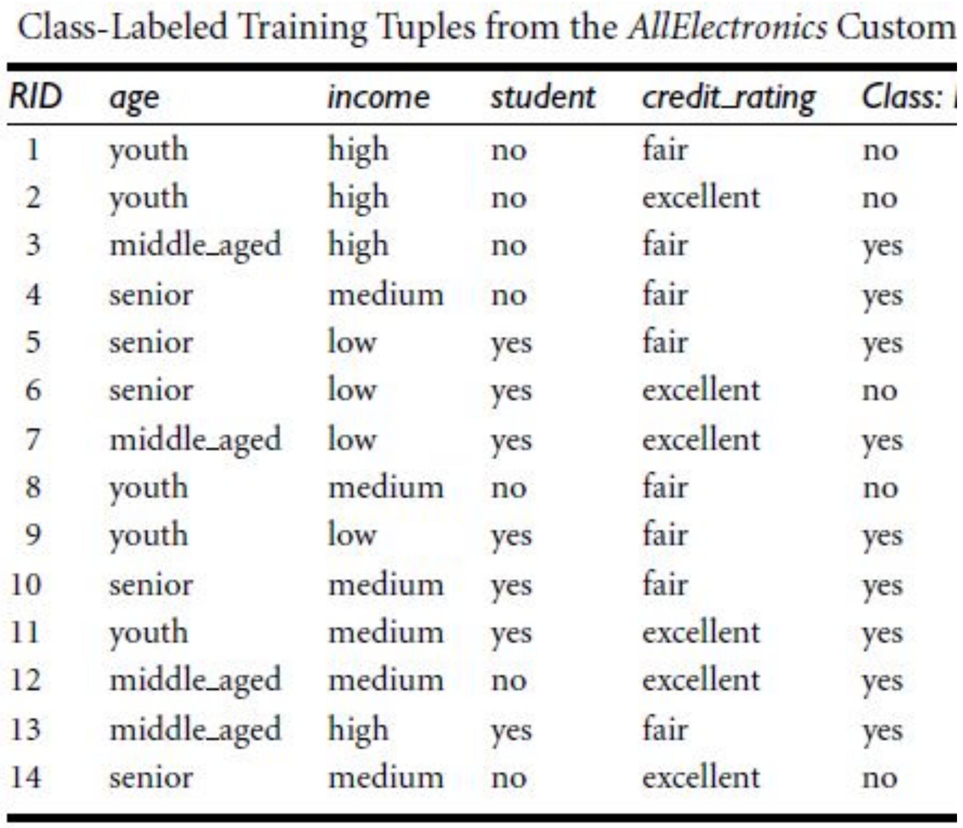
- The reduction in impurity that would be incurred by a binary split on a discrete- or continuous-valued attribute *A*

$$\boxed{\Delta Gini(A) = Gini(D) - Gini_A(D).}$$

- The attribute that maximizes the reduction in impurity (or, equivalently, has the minimum Gini index) is selected as the splitting attribute.
- This attribute & either its splitting subset or split-point together form the splitting criterion.

Induction of a decision tree using the Gini index

- C1= *buys computer =yes = 09*
- *C2 = buys computer = no = 05*



Class-Labeled Training Tuples from the *AllElectronics* Custom

RID	*age*	*income*	*student*	*credit_rating*	*Class:*
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

- the Gini index to compute the impurity of *D*

$$\boxed{Gini(D) = 1 - \sum_{i=1}^{m} p_i^2,}$$

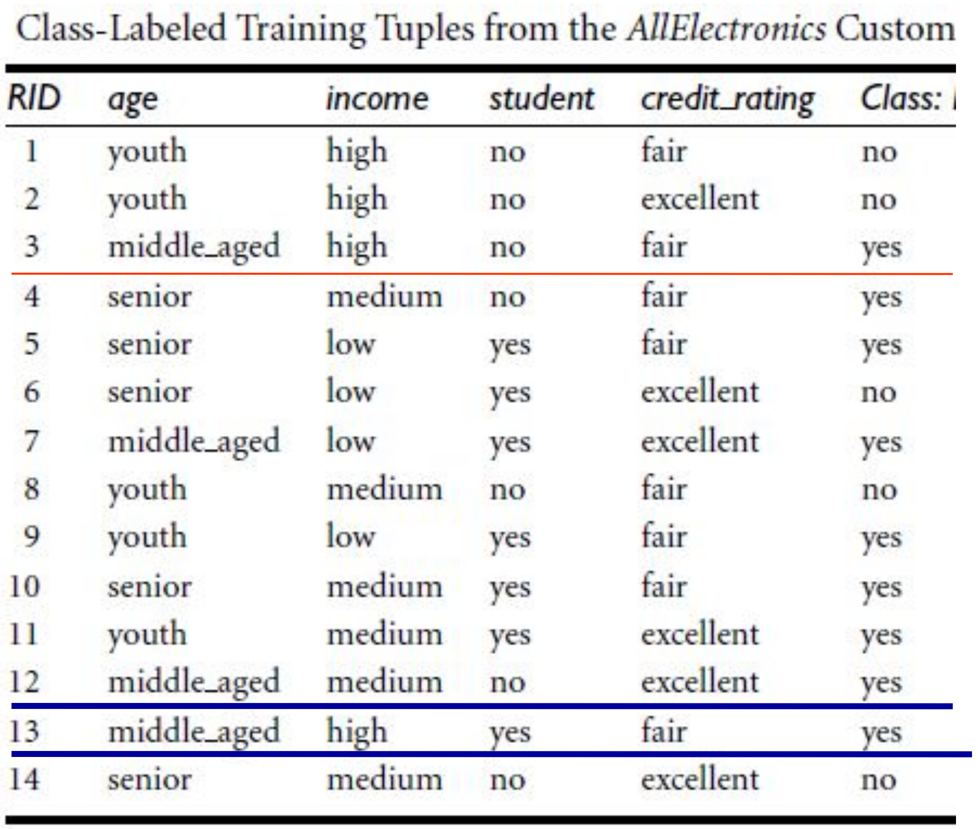
A (root) node *N is created for the tuples in D.*

$$Gini(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.459.$$

To find the splitting criterion for the tuples in *D, we need to compute the Gini index* for each attribute

- Let’s start with the attribute *income* & *consider each of the possible* splitting subsets.
- Consider the subset {*low, medium}.*
- *This would result in 10 tuples in* partition D_1 *satisfying the condition “income* $\in$ *{low, medium}.”*
- *The remaining 04* tuples of *D would be assigned to partition* D_2*.*

Yes = 07
No = 03


Class-Labeled Training Tuples from the *AllElectronics* Custom

RID	*age*	*income*	*student*	*credit_rating*	*Class:*
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

$$Gini_{income \in \{low,medium\}}(D)$$

$$= \frac{10}{14}Gini(D_1) + \frac{4}{14}Gini(D_2)$$

$$= \frac{10}{14}\left(1-\left(\frac{7}{10}\right)^2-\left(\frac{3}{10}\right)^2\right)+\frac{4}{14}\left(1-\left(\frac{2}{4}\right)^2-\left(\frac{2}{4}\right)^2\right)$$

$$= 0.443$$

$$= Gini_{income \in \{high\}}(D).$$

- Similarly, the Gini index values for splits on the remaining subsets

✓ 0.458 = {*low, high*} + {*medium*}

✓ *0.450 = {medium, high} + {low}*

- Therefore, the best binary split for attribute ***income*** *is on {low, medium} or {high} = 0.443* because it minimizes the Gini index.

❑ Evaluating *age, we obtain {youth, senior} (or {middle_aged}) as the best split for age with a Gini index of 0.375*

- the attributes *student* and *credit_rating are both binary, with Gini index values of 0.367 & 0.429, respectively.*
- The attribute ***age*** *and splitting subset {youth, senior} therefore give the minimum* Gini index overall, with a reduction in impurity of 0.459-0.357= 0.102.
- The binary split "*age* $\in$ *{youth, senior}" results in the* ***maximum reduction in impurity*** *of the tuples* in *D and is returned as the splitting criterion.*
- Node *N is labeled with the criterion, two* branches are grown from it, and the tuples are partitioned accordingly.

Decision Trees

best known:

- C4.5 (Quinlan)
- CART (Breiman, Friedman, Olshen & Stone)

- very fast to train and evaluate
- relatively easy to interpret
- but: accuracy often not state-of-the-art

Bayes Classification

What are Bayesian classifiers?”

- *Bayesian classifiers are statistical classifiers.*
- *They can* predict class membership probabilities such as the probability that a given tuple belongs to a particular class

Bayesian Classification: Why?

- A statistical classifier: performs *probabilistic prediction, i.e.,* predicts class membership probabilities
- Foundation: Based on Bayes' Theorem.
- Performance: A simple Bayesian classifier, *naïve Bayesian classifier*, has comparable performance with decision tree & selected neural network classifiers
- Incremental: Each training example can incrementally increase/decrease the probability that a hypothesis is correct — prior knowledge can be combined with observed data
- Standard: Even when Bayesian methods are computationally intractable, they can provide a standard of optimal decision making against which other methods can be measured

Bayes’ Theorem: Basics

- **Total Probability Theorem**:

$$\boxed{P(B)=\sum_{i=1}^{M} P(B|A_i)P(A_i)}$$

Bayes' Theorem: Basics

- Bayes' Theorem: $$\boxed{P(H|\mathbf{X}) = \frac{P(\mathbf{X}|H)P(H)}{P(\mathbf{X})} = P(\mathbf{X}|H) \times P(H) / P(\mathbf{X})}$$
  - Let **X** be a data sample ("*evidence*"): class label is unknown
  - Let H be a *hypothesis* that X belongs to class C
  - Classification is to determine P(H|**X**), (i.e., *posteriori probability):* the probability that the hypothesis holds given the observed data sample **X**
  - P(H) (*prior probability*): the initial probability
    - E.g., **X** will buy computer, regardless of age, income, …
  - P(**X**): probability that sample data is observed
  - P(**X**|H) (likelihood): the probability of observing the sample **X**, given that the hypothesis holds
    - E.g., Given that **X** will buy computer, the prob. that X is 31..40, medium income

Prediction Based on Bayes' Theorem

- Given training data **X**, *posteriori probability of a hypothesis* H, P(H|**X**), follows the Bayes' theorem

$$P(H|\mathbf{X}) = \frac{P(\mathbf{X}|H)P(H)}{P(\mathbf{X})} = P(\mathbf{X}|H)\times P(H)/P(\mathbf{X})$$

- Informally, this can be viewed as

  posteriori = likelihood x prior/evidence

- Predicts **X** belongs to C_i iff the probability $P(C_i|\mathbf{X})$ is the highest among all the $P(C_k|X)$ for all the *k* classes
- **Practical difficulty**: It requires initial knowledge of many probabilities, involving significant computational cost

Classification Is to Derive the Maximum Posteriori

- Let D be a training set of tuples & their associated class labels, & each tuple is represented by an n-D attribute vector **X** = (x_1, x_2, …, x_n)
- Suppose there are *m* classes C_1, C_2, …, C_m.
- Classification is to derive the maximum posteriori, i.e., the **maximal P(C_i|X)**
- This can be derived from Bayes' theorem

$$P(C_i|\mathbf{X}) = \frac{P(\mathbf{X}|C_i)P(C_i)}{P(\mathbf{X})}$$

- Since P(X) is constant for all classes, only

$$P(C_i|\mathbf{X}) = P(\mathbf{X}|C_i)P(C_i)$$

needs to be maximized

Naïve Bayes Classifier

- **When to use**
  - Moderate or large training set available
  - Attributes that describes instances are conditionally independent given classifier

- **Applications**
  - Diagnosis
  - Classifying text documents

Naïve Bayes Classifier

- **A simplified assumption**: attributes are conditionally independent (i.e., no dependence relation between attributes):

$$P(\mathbf{X}|C_i) = \prod_{k=1}^{n} P(x_k|C_i) = P(x_1|C_i) \times P(x_2|C_i) \times \ldots \times P(x_n|C_i)$$

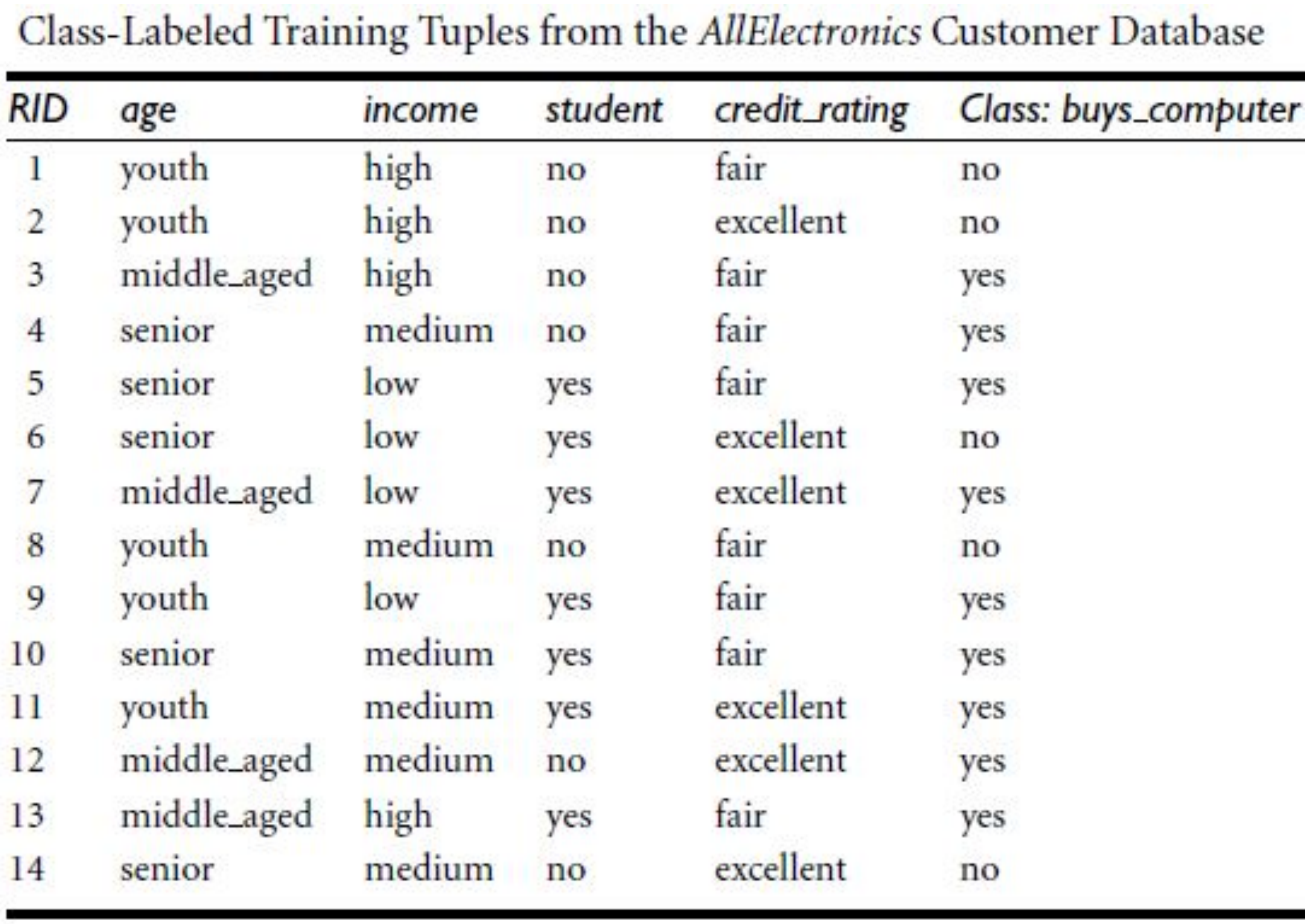
- This greatly reduces the computation cost: Only counts the class distribution
- If A_k is categorical, $P(x_k|C_i)$ is the # of tuples in C_i having value x_k for A_k divided by $|C_{i, D}|$ (# of tuples of C_i in D)
- If A_k is continous-valued, $P(x_k|C_i)$ is usually computed based on Gaussian distribution with a mean μ & standard deviation σ

$$g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

& $P(x_k|C_i)$ is

$$\boxed{P(\mathbf{X}|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i})}$$

Predicting a class label using na¨ıve Bayesian classification



Class-Labeled Training Tuples from the *AllElectronics* Customer Database

RID	*age*	*income*	*student*	*credit_rating*	*Class: buys_computer*
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

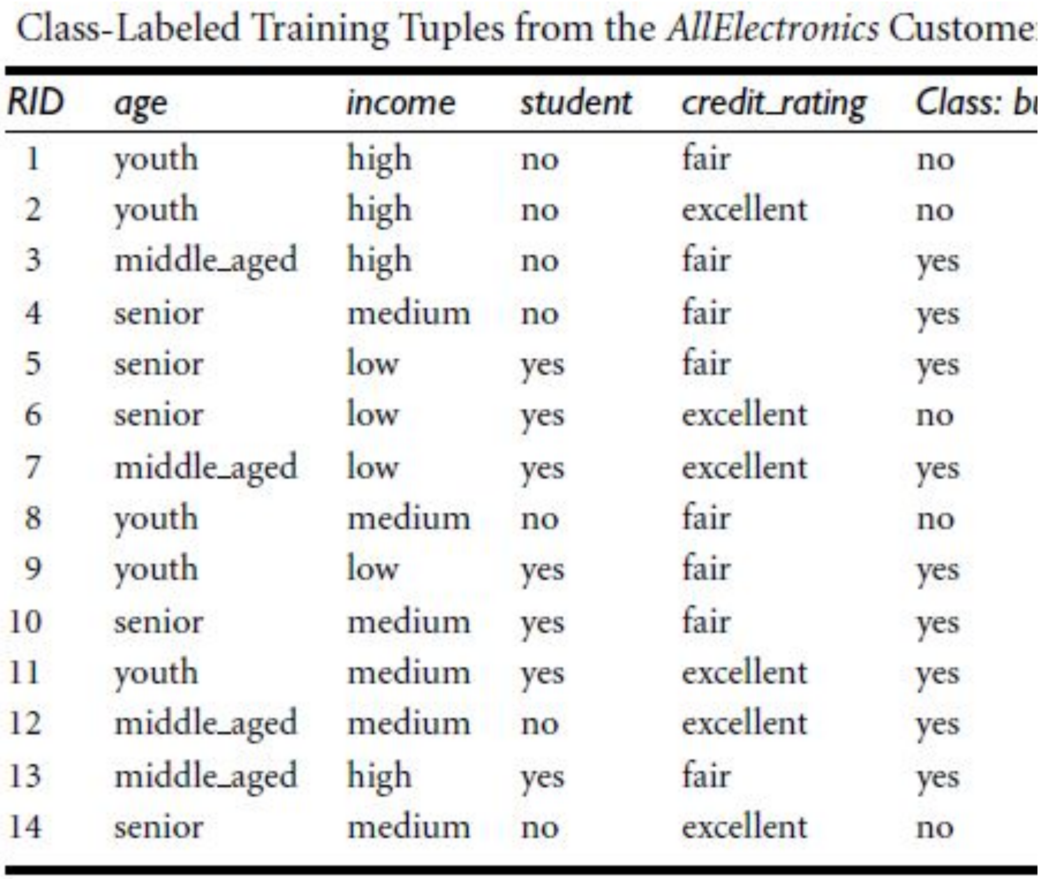
- The data tuples are described by the attributes:

- *age, income, student, & credit_rating*

Class:

C1: buys_computer = 'yes'

C2: buys_computer = 'no'


Class-Labeled Training Tuples from the *AllElectronics* Custome

RID	*age*	*income*	*student*	*credit_rating*	*Class: bı*
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Data to be classified:

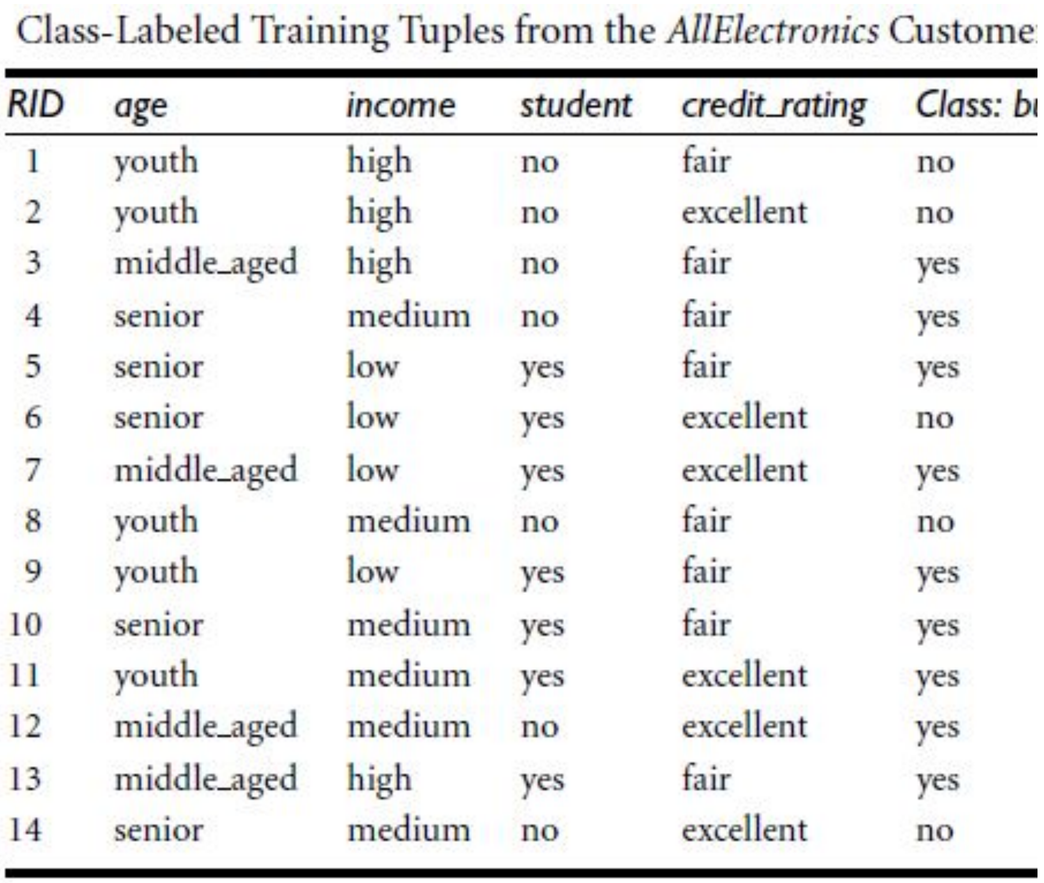
X = (age = youth, income = medium, student = yes, credit_rating = fair)

❑We need to maximize P(**X|C$_i$)P(C$_i$), for i = 1, 2.**

Compute: **P(Ci), the *prior probability of each*** class,

P(buys_computer = “yes”) = 9/14 = 0.643

P(buys_computer = “no”) = 5/14= 0.357



Class-Labeled Training Tuples from the *AllElectronics* Custome

RID	*age*	*income*	*student*	*credit_rating*	*Class: bu*
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Compute $P(X|C_i)$ for each class

P(age = youth | buys_computer = yes) = 2/9 = 0.222

P(age = youth | buys_computer = no) = 3/5 = 0.6

P(income = medium | buys_computer = yes) = 4/9 = 0.444

P(income = medium | buys_computer = no) = 2/5 = 0.4

P(student = yes | buys_computer = yes) = 6/9 = 0.667

P(student = yes | buys_computer = no) = 1/5 = 0.2

P(credit_rating = fair | buys_computer = yes) = 6/9 = 0.667

P(credit_rating = fair | buys_computer = no) = 2/5 = 0.4

X = (age = youth , income = medium, student = yes, credit_rating = fair)

P(X|C_i) :

- P(X|buys_computer = “yes”) = 0.222 x 0.444 x 0.667 x 0.667 = 0.044
- P(X|buys_computer = “no”) = 0.6 x 0.4 x 0.2 x 0.4 = 0.019

To find the class, **C_i** , *that maximizes* P(**X|C_i)P(C_i),**
Compute:

-P(X|buys_computer = “yes”) * P(buys_computer = “yes”) = 0.028

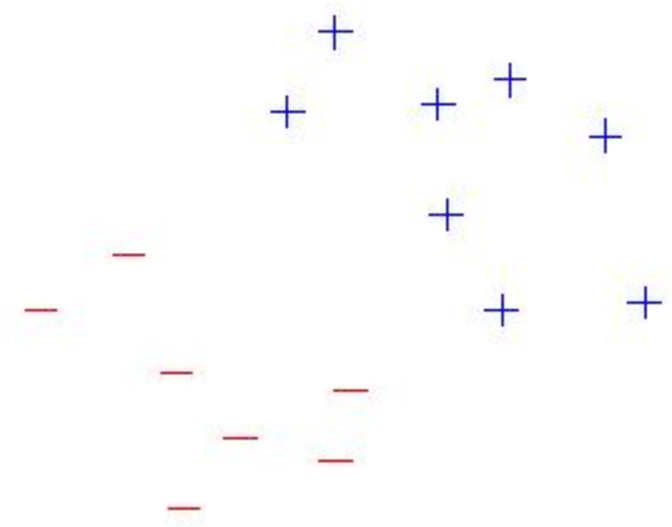
-P(X|buys_computer = “no”) * P(buys_computer = “no”) = 0.007

Therefore, X belongs to class (“buys_computer = yes”)

SVM-Support Vector Machine

SVM-Basics

- A relatively new classification method for both linear & nonlinear data
- It uses a nonlinear mapping to transform the original training data into a higher dimension
- With the new dimension, it searches for the linear optimal separating **hyperplane** (i.e., “decision boundary”)
- With an appropriate nonlinear mapping to a sufficiently high dimension, data from two classes can always be separated by a hyperplane
- SVM finds this hyperplane using **support vectors** (“essential” training tuples) & **margins** (defined by the support vectors)

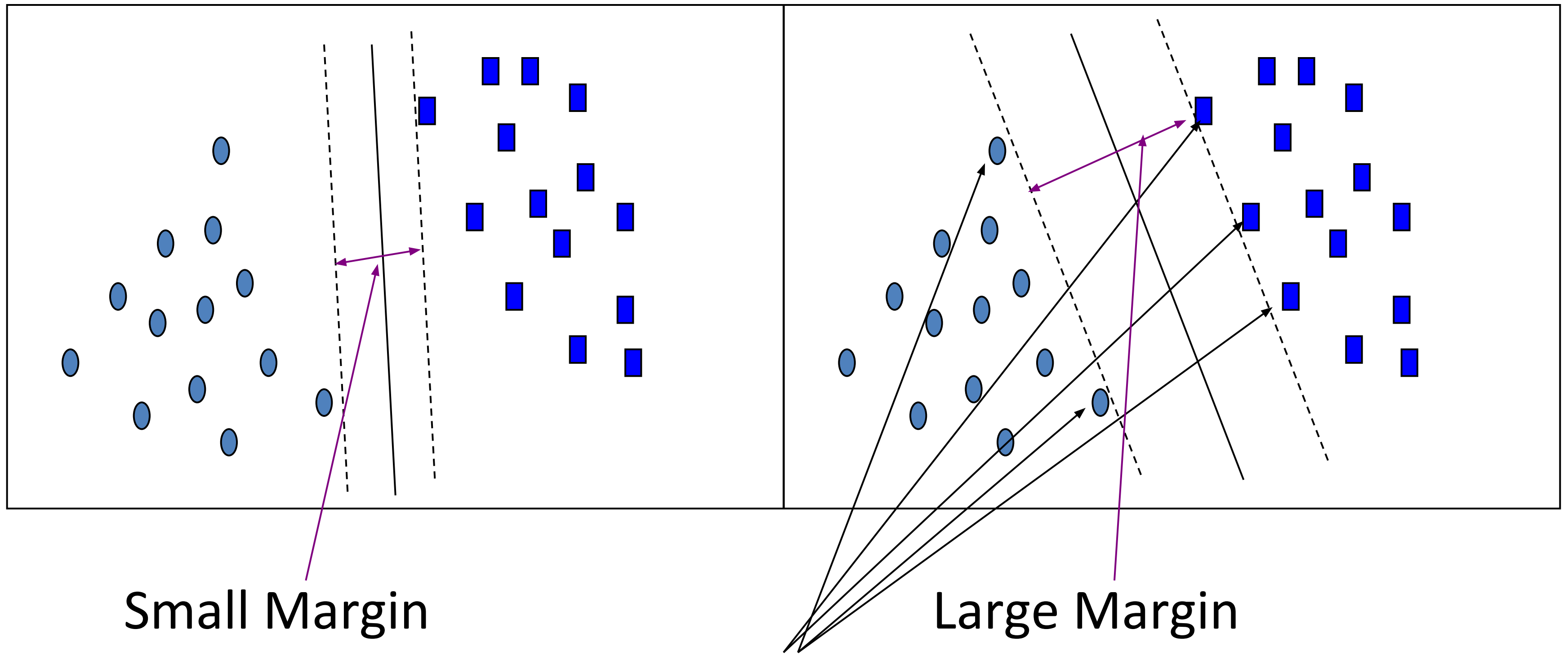


given linearly separable data

SVM—History and Applications

- Vapnik and colleagues (1992)—groundwork from Vapnik & Chervonenkis’ statistical learning theory in 1960s
- **Features:** training can be slow but accuracy is high owing to their ability to model complex nonlinear decision boundaries (margin maximization)
- **Used for**: classification and numeric prediction
- **Applications:**
  - handwritten character recognition, object/image recognition, speaker identification, benchmarking time-series prediction tests

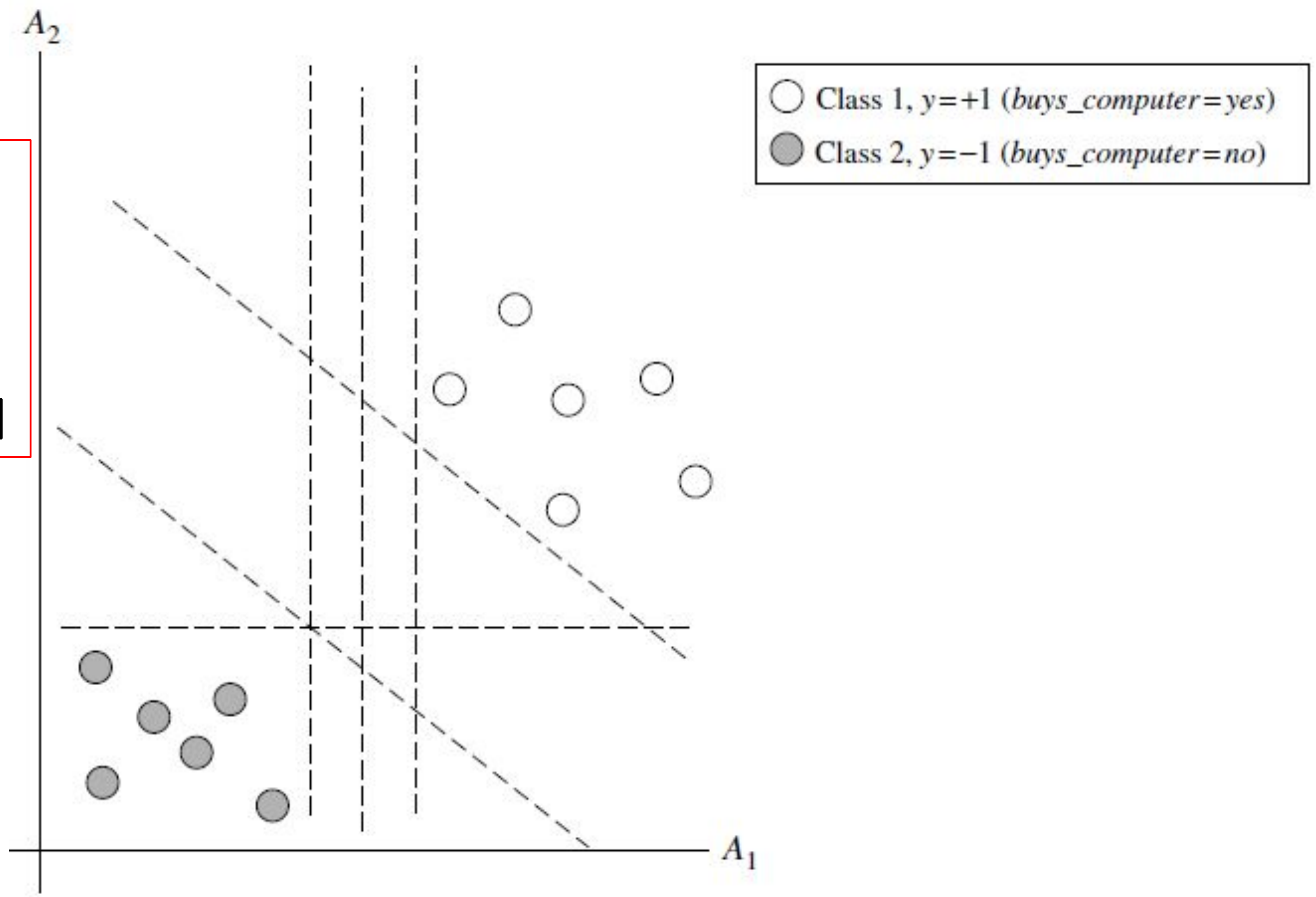
SVM—General Philosophy



Support Vectors

SVM—When Data Is Linearly Separable

2 class problem:
Database D
$(X_1, y_1), (X_2, y_2)$---
X_i: traning tuples
y_i: class level [+1/-1]

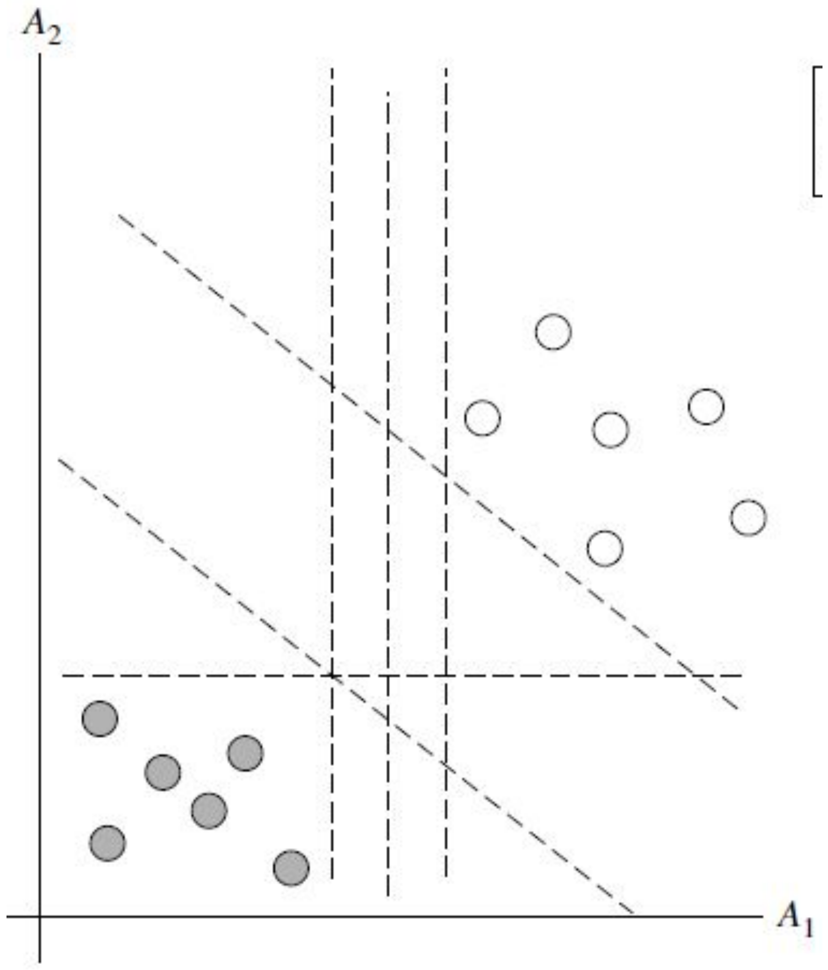


A_2
A_1
○ Class 1, y=+1 (*buys_computer=yes*)
● Class 2, y=−1 (*buys_computer=no*)

Data are **linearly separable**, because a straight line can be drawn to separate all the tuples of class +1 from all the tuples of class -1.

Which one is Best?

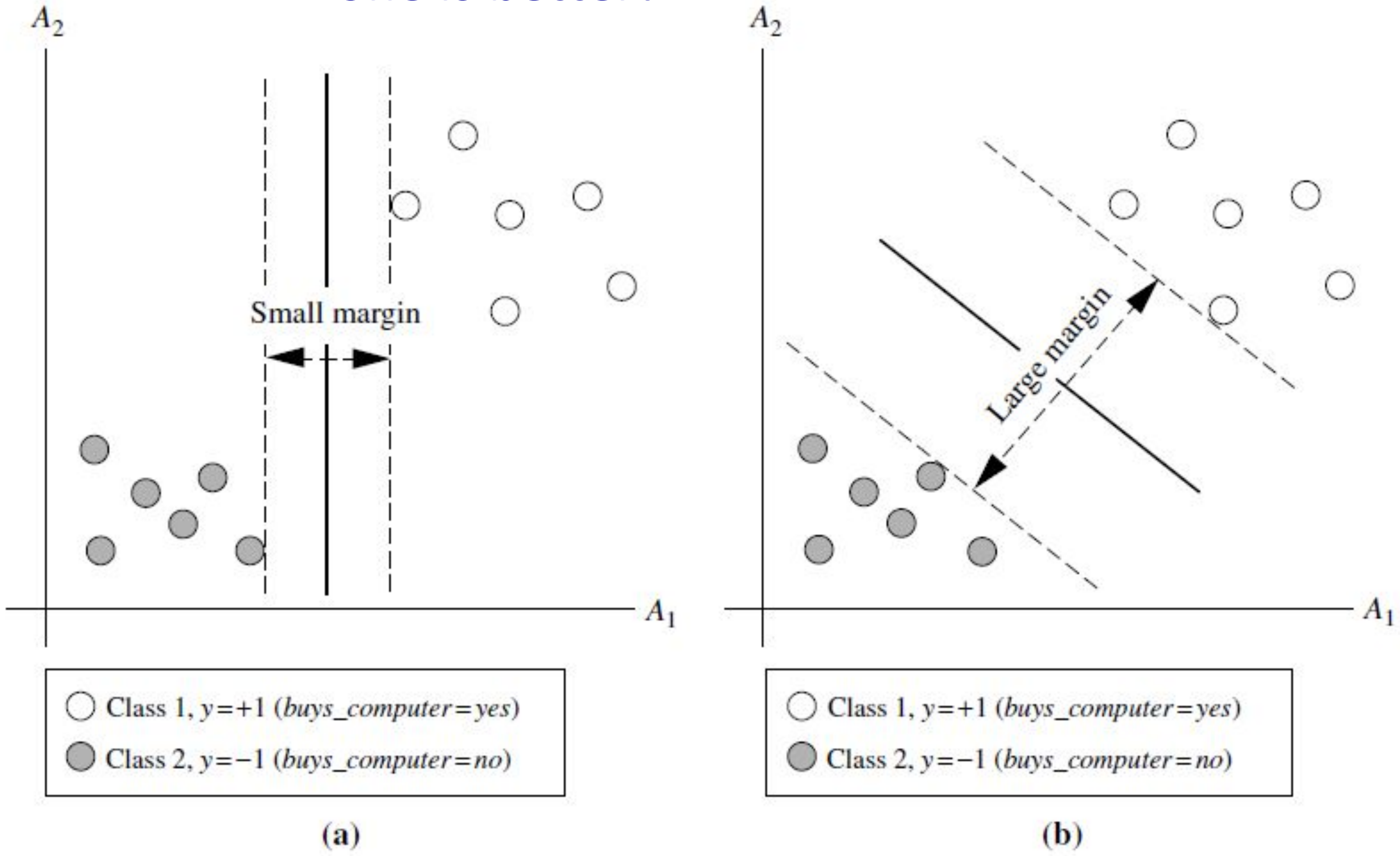
- infinite number of possible separating hyperplanes or “decision boundaries,”
- **Which one is best?**



- We want to find the "best" one, that is, one that will have the **minimum classification error** on previously unseen tuples.
- **How can we find this best line/hyperplane?**

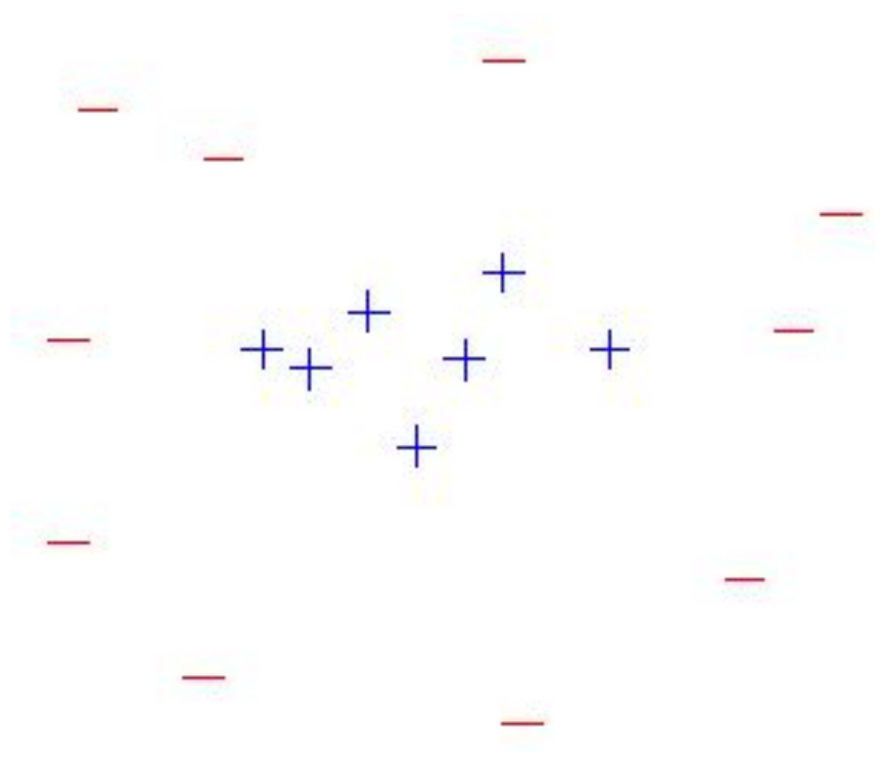
SVM searches for the hyperplane with the largest margin, i.e., **maximum marginal hyperplane** (MMH)

Which one is better?



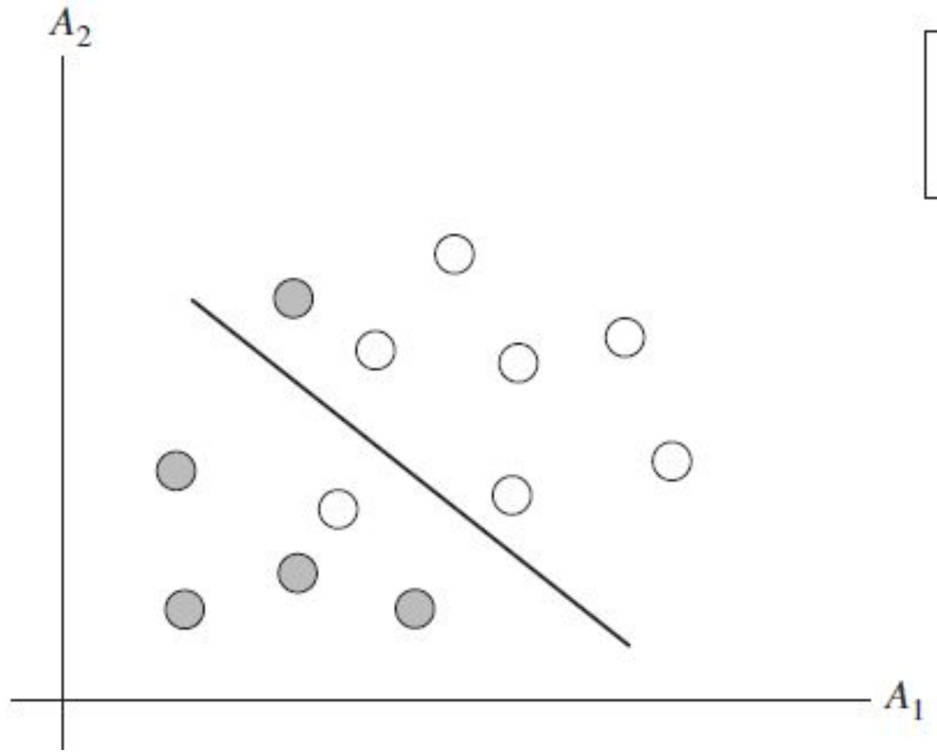
A_2
Small margin
A_1
○ Class 1, y=+1 (*buys_computer=yes*)
● Class 2, y=−1 (*buys_computer=no*)
(a)
A_2
Large margin
A_1
○ Class 1, y=+1 (*buys_computer=yes*)
● Class 2, y=−1 (*buys_computer=no*)
(b)

SVM—Linearly Inseparable



not linearly separable

SVM—Linearly Inseparable



○	Class 1, $y=+1$ (*buys_computer=yes*)
●	Class 2, $y=-1$ (*buys_computer=no*)

Now what?

It is not possible to draw a straight line to separate the classes. Instead, the decision boundary is nonlinear

Extend linear approach…

- **How can we extend the linear approach?**
- Two main steps:
- Transform the original input data into a higher dimensional space using a nonlinear mapping.
- Searches for a linear separating hyperplane in the new space.
- **Outcomes**: a quadratic optimization problem that can be solved using the linear SVM formulation.
- The maximal marginal hyperplane found in the new space corresponds to a nonlinear separating hypersurface in the original space.

Tips! If Writing Your Own Code

- Matlab are great for easy coding, but for speed, may need C or java
- debugging machine learning algorithms is very tricky!
  - hard to tell if working, since don't know what to expect
  - run on small cases where can figure out answer by hand
  - test each module/subroutine separately
  - compare to other implementations (written by others, or written in different language)
  - compare to theory or published results

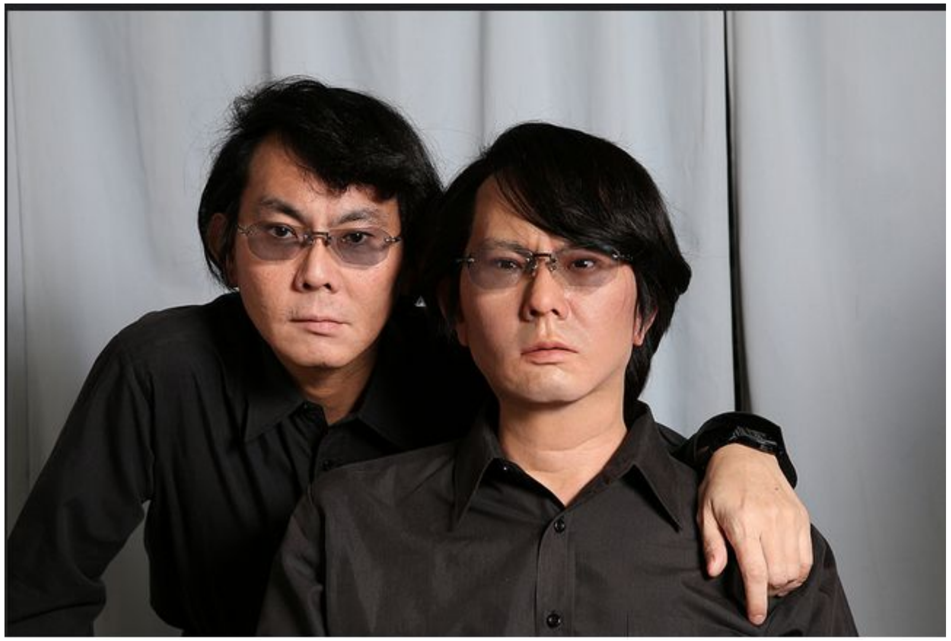
Conclusion

- ML: How can we program systems to automatically learn & to improve with experience?
- In order to build up an intelligent or autonomous agents, ML should be used.
- Still wide gaps: need to perform plenty of social & cognitive capabilities
- Not so far away from our dreams!
- Artificial Humans co-play in the human world very soon.

Conclusion

- **AI dream of someday building machines as intelligent as you or I** - Andrew Ng.

A Puzzle………



Who is the real human?

Thanks