Machine Learning (DS4102PC/ CY4101PC)

T.APARNA
Assistant Professor
Department of CSE

References

Text Books:

1. Tom M.Mitchell, Machine Learning, India Edition2013,McGraw HillEducation.

Reference Books:

1. Trevor Hastie, Robert Tibshirani, Jerome Friedman, h The Elements of Statistical Learning, 2nd edition, springer seriesin statistics.
2. Ethem Alpaydın, Introduction to machine learning, second edition, MITpress.

Prerequisites

For Machine Learning Course were commend that students meet the following prerequisites:

- Basic programming skills(in Python)
- Algorithm design
- Basics of probability & statistics

Content

Unit–1	Introduction, Concept Learning, Decision Tree
Unit–2	LearningArtificialNeuralNetworks-1, Artificial Neural Networks-2, Evaluating Hypothesis,
Unit–3	Bayesian Learning, Computational learning theory, Instance Based Learning,
Unit–4	Genetic Algorithms, Learning Sets of Rules, Reinforcement Learning
Unit–5	AnalyticalLearning-1, Analytical Learning-2, Combining Inductive and Analytical Learning

UNIT-1

Machine Learning
Introduction

Ever since computers were invented, we have wondered whether they might be Made to learn. If we could understand how to program them to learn-to improve automatically with experience-the impact would be dramatic.

- Imagine computers learning from medical records which treatments are most effective for new diseases
- Houses learning from experience to optimize energy costs based on the of their occupants.
- Personal software assistants learning the evolving interests of their to highlight especially relevant stories from the online morning newspaper

Examples of Successful Applications of Machine Learning

- Learning to recognize spoken words
- Learning to drive an autonomous vehicle
- Learning to classify new astronomical structures
- Learning to playworld-classback gammon

Why is Machine Learning Important?

- Some tasks cannot be defined well, except by examples (e.g., recognizing people).
- Relationships and correlations can be hidden within large amounts of data.Machine Learning/DataMining maybe able to find these relationships.
- Human designers of ten produce machines that donot work as well as desired in the environments in which they are used.
- The amount of knowledge available about certain tasks might be too large for explicit encoding by humans (e.g., medical diagnostic).
- Environments change overtime.
- New knowledge about tasks is constantly being discovered by humans. It may be difficult to continuously re-designsystems “by hand”.

Areas of Influence for Machine Learning

- ***Statistics:*** How best touse samples drawn from unknown probability distributions to help decidefrom which distributionsomenewsampleisdrawn?
- ***Brain Models***: Non-linear elements with weighted inputs (Artificial NeuralNetworks) have been suggested as simple models of biological neurons.
- ***AdaptiveControlTheory***: How to deal with controlling aprocess having unknown parameters that must be estimated during operation?
- ***Psychology***: How to model human performance on various learning tasks?
- ***ArtificialIntelligence***: How to write algorithms to acquire the knowledge humans are able to acquire, atleast, as well as humans?
- ***EvolutionaryModels***: How to model certain aspects of biological evolution to improve the performance of computer programs?

Machine Learning: A Definition

A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

Why “Learn”?

Learning issused when:

- Human expertise does not exist(navigating on Mars)
- Humans are unable to explain their expertise (speech recognition)
- Solution changes intime(routing on a computernetwork)
- Solution needs to be adapted to particular cases(userbiometrics)

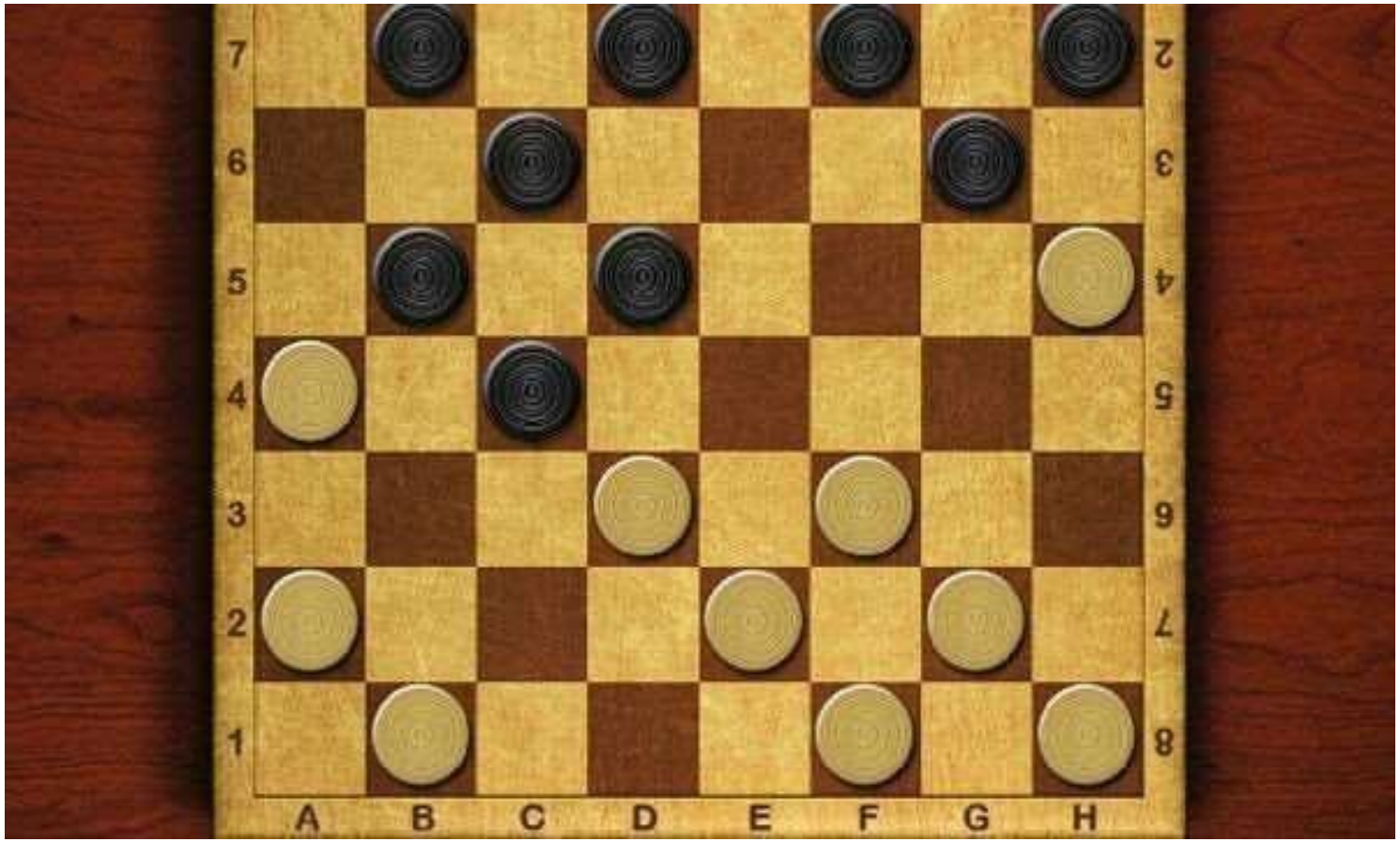
Well-Posed Learning Problem

Definition: A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance attasks in T, as measured by P, improves with experience E.

To have a well-defined learning problem, three features needs to be identified:

1. The classof tasks
2. The measure of performance to be improved
3. The source of experience

Checkers Game



Game Basics

- Checkers is played by two players. Each player begins the game with 12 coloreddiscs. (One set of pieces is black and the other red.) Each player places his or her pieces on the 12 dark squares closest to him or her. Black moves first. Players then alternate moves.
- The board consists of 64 squares, alternating between 32dark and 32lightsquares.
- It is positioned so that each player has a light square on the right side corner closest to him or her.
- A player wins the game when the opponent cannot make a move. In most cases, this is because all of the opponent's pieces have been captured, but it could also be because all of his pieces are blocked in.

Rules of the Game

- Moves are allowed only on the dark squares, so pieces always move diagonally. Single pieces are always limited to forward moves (toward the opponent).
- A piece making a non-capturing move (not involving a jump) may move only one square.
- A piece making a capturing move (a jump) leaps over one of the opponent'spieces, landing in a straight diagonal line on the other side. Only one piece maybe captured in a single jump; however, multiple jumps are allowed during a single turn.
- When a piece is captured, it is removed from the board.
- If a player is able to make a capture, there is no option; the jump must be made.
- If more than one capture is available, the player is free to choose whichever he or she prefers.

Rules of the GameCont.

- When a piece reaches the furthest row from the player who controls that piece, itis crowned and becomes a king. One of the pieces which had been captured is placed on top of the king so that it is twice as high as a single piece.
- Kings are limited to moving diagonally but may move both forward and backward. (Remember that single pieces, i.e. non-kings, are always limited to forward moves.)
- Kings may combine jumps in several directions, forward and backward, on the same turn. Single pieces may shift direction diagonally during a multiple captureturn, but must always jump forward (toward the opponent).

Well-Defined Learning Problem

A checkers learning problem:

- TaskT:playing checkers
- Performance measure P: percent of games won against opponents
- Training experience E:playing practice games against itself

A handwriting recognition learning problem:

- TaskT:recognizing and classifying hand written words within images
- Performance measureP:percent of words correctly classified
- Training experienceE: a database of handwritten words with given classifications

A robot driving learning problem:

- TaskT:driving on public four-lane highways using vision sensors
- Performance measureP:average distance travelled before anerror (as judged by human overseer)
- Training experienceE:a sequence of images and steering commands recorded While observing a human driver

Designing a Learning System

1. Choosing theTrainingExperience
2. Choosing theTargetFunction
3. Choosing a Representation for theTarget Function
4. Choosing a Function ApproximationAlgorithm
   1. Estimating training values
   2. Adjusting the weights
5. The Final Design

The basic design issues and approaches to machinelearning is illustrated by considering designing a program to learn to play checkers,with the goal of entering it in the world checkers tournament

1. Choosing the Training Experience

- The first design choice is to choose the type of training experience from which the system will learn.
- The type of training experience available can have a significant impact on success or failure of the learner.

<u>There are three attributes which impact on successor failure of the learner</u>

1. Whether the training experience provides director indirect feedback regarding the choices made by the performance system.
2. The degree to which the learner controls the sequence of training examples
3. How well it represents the distribution of examples over which the final system performance mustbe measured.

1. Whether the training experience provides director indirect feedback regarding the choices madeby the performance system.

Forexample, in checkers game:

- In learning to playcheckers,the system might learn from direct training examples consisting of individual ***Checkers boardstates and the correct move foreach***.

- Indirect training examples consisting of the ***move sequences and final outcomes of various games played***.

- The information about the correctness of specific moves early in the game must be inferred indirectly from the fact that the game was eventually won or lost.

- Here the learner faces an additional problem of ***credit assignment***, or determining the degree to which each move in the sequence deserves creditor blame for the finaloutcome.

- Creditassignment can be aparticularly difficult problem because the game can be lost even when early moves are optimal,if the seare followed later by poor moves.

- Hence, learning from direct training feedback is typically easier than learning from indirect feedback.

2. Asecond important attribute of the training experience ***is the degree to which the learner controls the sequence of training examples***

For example, in checkers game:

- The learner might depends on the teacher to select informative board states and to provide the correct move foreach.

- Alternatively, the learner might itself propose board states that it finds particularly confusing and ask the teacher for the correct move.

- The learner may have complete control over both the board states and (indirect) training classifications, as it does when it learns by playing against itself with no teacher present.

- Notice in this last case the learner may choose between experimenting with novel board states that it has notyet considered, or honing its skill by playing minor variations of lines of play it currently finds mostpromising.

3. A third attribute of the training experience is how well it represents ***the distribution of examples*** over which the final system performance must be measured.

Learning is most reliable when the training examples follow a distribution similar to that of future test examples.

For example, in checkers game:

- In checkers learning scenario, the performance metric P is the percent of games the system wins in the worldtournament.

- If its training experience E consists only of games played against itself, there is an danger that this training experience might not be fully representative of the distribution of situations over which it will later be tested. For example, the learner might never encounter certain crucial board states that are very likely to be played by the human checkers champion.

- It is necessary to learn from a distribution of examples that is somewhat different from those on which the final system will be evaluated. Such situations are problematic because mastery of one distribution of examples will not necessary lead to strong performance over some other distribution.

3. Choosing the Target Function

The next design choice is to determine exactly what type of knowledge will be learned and how this will be used by the performance program.

- Let's begin with a checkers-playing program that can generate the legal moves from any boardstate.
- The program needs only to learn how to choose the best move from among these legal moves. This learning task is representative of a large class of tasks for which the legal moves that define some large search space are known a priori, but for which the best search strategy is not known.

Given this setting where we must learn to choose among the legal moves, the most obvious choice for the type of information to be learned is a program, or function,that chooses the best move for any given board state.

1. Let ***Choose Move*** be the target function and the notation is

$$\textit{ChooseMove: } B \longrightarrow M$$

Which indicate that this function accepts as input any board from the set of legal boardstates Band produces as output somemove from the set of legal moves M.

ChooseMove is an choice for the target function in checkers example, but this function will turn out to be very difficult to learn given the kind of indirect training experience available to our system

2. An alternative targetfunction is an ***evaluationfunction*** that assigns a ***numericalscore*** to any given boardstate

Let the target functionVand the notation

$$V: B \longrightarrow R$$

Which denote that Vmaps any legal board state from the setB to some real value

We intend for this target function V to assign higher scores to better board states. If the system can successfully learn such a target functionV, it can easily use it to select the best move from any current board position.

Let us define the target valueV (b) for an arbitrary board state binB, as follows:

1. If b is a finalboard state that is won,thenV(b)=100
2. If b is a finalboard state that is lost,thenV(b)=-100
3. If b is a finalboard state that is drawn,thenV(b)=0
4. If b is a not a final state in the game,thenV(b)=V(b'),

Where b'is the best final board state that can be achieved starting from band playing optimally until the end of the game

3. Choosing a Representation for the Target Function

Let us choose a simple representation-for any givenboard state,the function will be calculated as a linear combination of the following board features:

XL: the number of black pieces on the
boardx2: the number of red pieces on the
boardx3: the number of black kings on the
boardx4: the number of red kings on the
board
X5: the number of black pieces threatened byred (i.e., which can be captured on red's next turn)
x6: the number of red pieces threatened by black

Thus, learning program will represent as a linearfunction of the form

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

Where,

- w_0 through w_6 are numerical coefficients,or weights,to be chosen by the learning algorithm.

- Learned values for the weights w_1 through w_6 will determine the relative Importance of the various board features in determining the value of the board
  - The weightw_0 will provide an additive constant to the boardvalue

Partialdesignofacheckerslearningprogram:

- TaskT:playing checkers
- Performance measureP:percent of games won in the world tournament
- Training experienceE:games played against itself
- Target function:V:Board $\longrightarrow$ R
- Target function representation

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

The first three items above correspond to the specification of the learning task, where as the final two items constitute design choices for the implementation of the learning program.

4. Choosing a Function Approximation Algorithm

- In order to learn the target function we require a set of training examples, each describing a specific board state band the training value $V_{train}(b)$ for b.
- Each training example is an ordered pair of the form $(b, V_{train}(b))$.
- For instance, the following training example describes a board state b in which black has won the game (note$x_2=0$ indicates that red has no remaining pieces) and for which the target function value$V_{train}(b)$ is therefore +100.

$$((x_1=3, x_2=0, x_3=1, x_4=0, x_5=0, x_6=0), +100)$$

Function Approximation Procedure

1. Derive training examples from the indirect training experience available to the learner
2. Adjusts the weights w_i to best fit these training examples

A simple approach for estimating training values for intermediate boardstates is to assign the training value of $V_{train}(b)$ for any intermediate board state b to be $\hat{v}$(Successor (b))

Where,
$\hat{v}$ is the learner's current approximation to V
Successor (b) denotes the nextboard state following for which it is again the program's turn to move

Rule for estimating trainingvalues

$$V_{train}(b) \leftarrow \hat{v}(\text{Successor}(b))$$

2.Adjustingtheweights

Specify the learning algorithm for choosing the weights w_i to best fit the set of training examples $\{(b, V_{train}(b))\}$

A firststep is to define what we mean by the bestfit to the training data.

- One common approach is to define the best hypothesis, or set of weights, as that which minimizes the squared error E between the training values and the values predicted by the hypothesis.

$$E \equiv \sum_{\langle b, V_{train}(b)\rangle \in \text{ training examples}} (V_{train}(b) - \hat{V}(b))^2$$

- Several algorithms are known for finding weights of a linear function that minimizeE.

In our case, we require an ***algorithm*** that will incrementally refine the weights asnew training examples become available and that will be robust to errors in these estimated training values

One such algorithm is called the ***least mean squares***, or ***LMS training rule***. For each observed training example it adjusts the weights asmall amount in the direction that reduces the error on this training example

LMS weightupdate rule:-For each training example

(b,V_{train}(b))Use the current weights to calculate$\hat{v}$(b)

For each weightw_i, update it as

$$w_i \leftarrow w_i + \eta\ (Vtrain\ (b) - \hat{v}(b))\ x_i$$

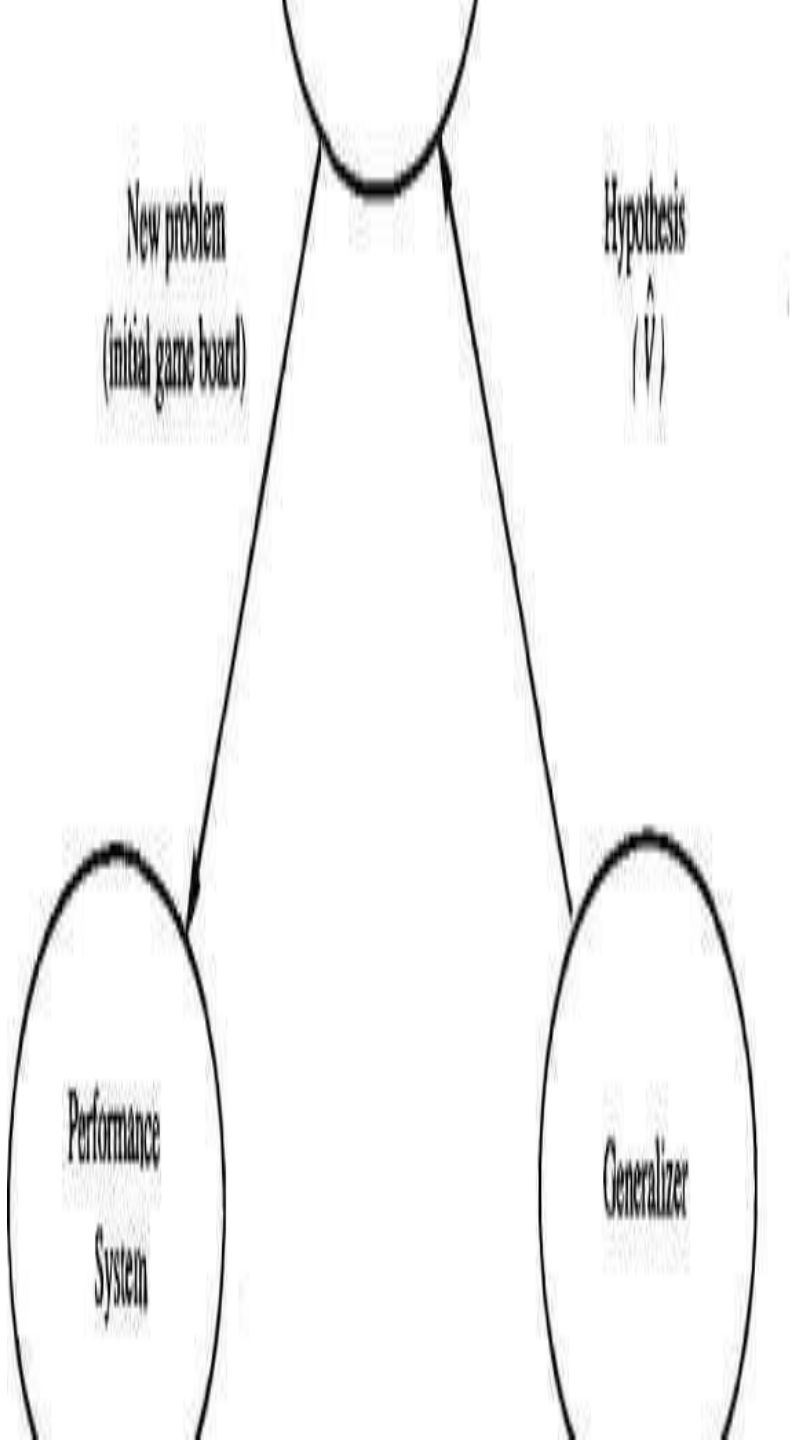
Here η is a small constant (e.g., 0.1) that moderates the size of the weight

update.Working of weight update rule

- When the error (Vtrain(b)-v̂(b))is zero,no weights are changed.
- When (Vtrain(b)-v̂(b))is positive(i.e.,whenv̂(b)is too low),then each weight is increased in proportion to the value of its corresponding feature. This will raise the value ofv̂(b), reducing the error.
- If the value of some featurex_i is zero, then its weight is not altered regardless of the error, so that the only weights updated are those whose features actually occur on the training example board.

5. The Final Design

The final design of checkers learning system can be described by four distinct program modules that represent the central components in many learning systems


New problem
(initial game board)
Hypothesis
($\hat{V}$)
Performance
System
Generalizer

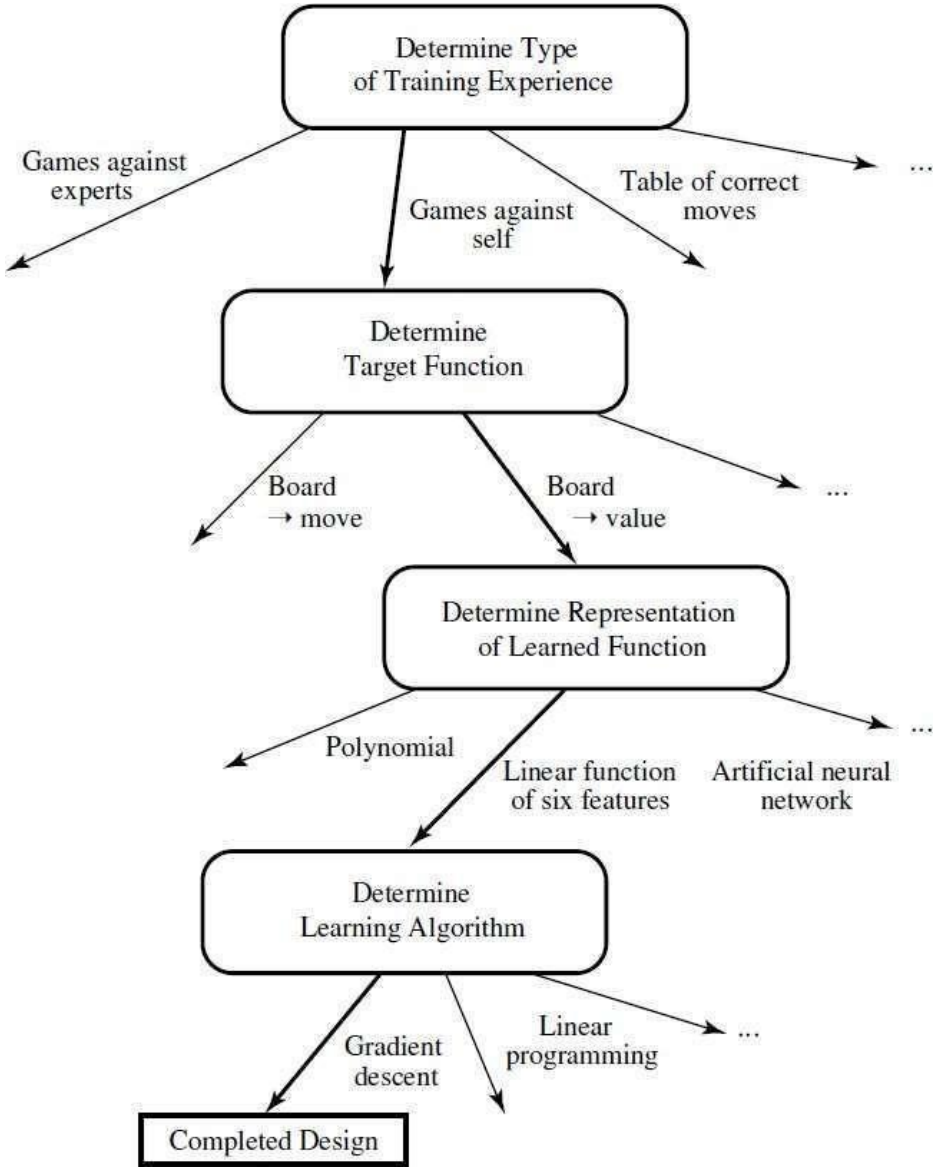
1. ***The Performance System*** is the module that must solve the given performance taskby using the learned target function(s).
It takes an instance of a new problem (new game) as input and produces a trace of its solution (game history) as output.
In checkers game, the strategy used by the Performance System to select its next move at each step is determined by the learned $\hat{v}$ evaluation function. Therefore, weexpect its performance to improve as this evaluation function becomes increasingly accurate.

2. ***The Critic*** takes as input the history or trace of the game and produces as output a set of training examples of the target function. As shown in the diagram, each training example in this case corresponds to some game state in the trace, along with an estimateV_{train} of the target function value for this example.

3. The Generalizer takes as input the training examples and produces an output hypothes is that is its estimate of the target function.
It generalizes from the specific training examples, hypothesizing a general function that covers these examples and other cases beyond the training examples.
Inour example, the Generalizer corresponds to the LMS algorithm, and the output hypothesis is the function$\hat{v}$ described bythe learned weightsw_0, .. . , W6.

4. The Experiment Generator takes as input the current hypothesis and outputs anew problem (i.e., initial board state) for the Performance System to explore.Its role is to pick new practice problems that will maximize the learning rate of the over all system.
In our example, the Experiment Generatoral ways proposes the same initial game board to begin a new game.

The sequence of design choices made for the checkers program is summarized in below figure



Perspectives of Machine Learning

Perspective of machine learning involves searching very large space of possible hypothesis to determine one that

Best fits the observed data and any prior knowledge heldby learner.

Issues in MachineLearning

- What algorithms exist for learning general target functions from specific training examples? In what settings will particular algorithms converge to the desired function, given sufficient training data?Which algorithms perform best for which types of problems and representations?

- How much training data is sufficient? What general bounds can be found to relate the confidence in learned hypotheses to the amount of training experience and the character of the learner's hypothesis space?

- When and how can prior knowledge held by the learner guide the process of generalizing from examples? Can prior knowledge be helpful even when it is only approximately correct?

- What is the best strategy for choosing a useful next training experience, and how does the choice of this strategy alter the complexity of the learning problem?

- What is the best way to reduce the learning task to one or more function approximation problems? Put another way, what specific functions should the system attempt tolearn? Can this process itself be automated?

- How can the learner automatically alter its representation to improve its ability to represent and learn the target function?

Concept Learning

- Learning involves acquiring general concepts from specific training examples. Example: People continually learn general concepts or categories such as "bird,""car,""situations in which I should study more in order to pass the exam,"etc.
- Each such concept can be viewed as describing some subset of objects or events defined over a larger set
- Alternatively, each concept can be though to fasaBoolean-valued function defined over this larger set. (Example: A function defined over all animals, whose value is true for birds and false for other animals).

Conceptlearning-Inferring a Boolean-valued function from training examples of its input andoutput

A Concept Learning Task

Consider the example task of learning the target concept
"Dayson which my friend Aldoenjoysh is favorite watersport."

Example	Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

Table-Describes a set of example days, each represented by a set of attributes

The attribute ***EnjoySport*** indicates whether or not a Person enjoys his favorite watersport on this day.

The task is toleranto predict thevalue of ***EnjoySport*** for an arbitrary day, based on the values of its other attributes?

What hypothes is representation is provided to the learner?

Let’s consider as imple representation in which each hypothesis consists of a conjunction of constraints on the instance attributes.

Let each hypothes is be a vectorof six constraints, specifying the values of the six attributes Sky, AirTemp, Humidity,Wind,Water, and Forecast.

For each attribute, the hypothes is will either

- Indicate bya"?'that any valueis acceptable for this attribute,
- Specify a single required value(e.g.,Warm)for the attribute,or
- Indicate bya"Φ"that novalueis acceptable

Ifsomeinstancexsatisfiesalltheconstraintsofhypothesish,thenhclassifies Xasapositive example(h(x) =1).

The hypothesis that PERSON enjoys his favo rite sport only on cold days with highhumidity (independent of the values of the other attributes) is represented by theexpression

(?, Cold, High,?,?,?)

Themostgeneral hypothesis-that everydayisapositiveexample-isrepresentedby (?,?,?,?,?,?)

The most specific possible hypothesis-that day is a positive example-is norepresented by

Ifsomeinstancexsatisfiesalltheconstraintsofhypothesish,thenhclassifies (Φ,Φ,Φ,Φ,Φ,Φ)

Notation

The set of items over which the concept is defined is called the set of ***instances***, which we denote by X.

Example: X is the set of all possible days, each represented by the attributes:Sky, AirTemp, Humidity,Wind,Water,and Forecast

The concept or function to be learned is called the ***targetconcept***, which we denote by c.

c can be any Boolean valued function defined overtheinstancesXc: X {O, 1}

Example: The target concept corresponds to the value of the attribute ***EnjoySport*** (i.e.,c(x)=1if***EnjoySport***=Yes, and c(x)=0 if ***EnjoySport***=No).

- Instances for which c(x)=1 are called positive examples,or members of the target concept.
- Instancesforwhichc(x)=0 are called negative examples,or non-members of the target concept.
- The ordered pair(x,c(x)) to describe the training example consisting of the instancex and its target conceptvalue c(x).
- ***D*** to denote the set of available training examples
- The symbol ***H*** to denote the set of all possible hypotheses that the learner may consider regarding the identity of the target concept. Each hypothesis ***h*** in ***H***represents a Boolean-valued function defined over X

$$h:X \longrightarrow \{O,1\}$$

- The goal of the learner is to find a hypothesis such that h(x)=c(x)forall x in ***X***.

Example	Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

- **Given:**
  - Instances X: Possible days, each described by the attributes
    - *Sky* (with possible values *Sunny*, *Cloudy*, and *Rainy*),
    - *AirTemp* (with values *Warm* and *Cold*),
    - *Humidity* (with values *Normal* and *High*),
    - *Wind* (with values *Strong* and *Weak*),
    - *Water* (with values *Warm* and *Cool*), and
    - *Forecast* (with values *Same* and *Change*).
  - Hypotheses H: Each hypothesis is described by a conjunction of constraints on the attributes *Sky*, *AirTemp*, *Humidity*, *Wind*, *Water*, and *Forecast*. The constraints may be "?" (any value is acceptable), "Ø" (no value is acceptable), or a specific value.
  - Target concept c: $EnjoySport : X \rightarrow \{0, 1\}$
  - Training examples D: Positive and negative examples of the target function (see Table 2.1).
- **Determine:**
  - A hypothesis h in H such that $h(x) = c(x)$ for all x in X.

TABLE The *EnjoySport* concept learning task.

The Inductive Learning Hypothesis

Any hypothesis found to approximate the target function well over a sufficiently large set of training examples will also approximate the target function well overother unobserved examples.

Concept learning as Search

- Concept learning can be viewed as the task of searching through a large space of hypotheses implicitly defined bythe hypothesis representation.
- The goal of this search is to find the hypothesis that best fits the training examples.

Example, the instances ***X*** and hypotheses ***H*** in the ***EnjoySport*** learningtask. The attribute ***Sky*** has three possible values, and ***AirTemp,Humidity ,Wind,WaterForecast*** each have two possible values, the instance space X contains exactly

- 3.2.2.2.2.2=96 Distinct instances
- 5.4.4.4.4.4=5120 syntactically distinct hypotheses within H.

Every hypothesis containing one or more"Φ"symbols represents the empty set of instances; that is, it classifies every instance as ***negative.***

- 1 +(4.3.3.3.3.3)=973. Semantically distinct hypotheses

General-to-SpecificOrderingofHypotheses

- Considerthetwohypotheses

  h_1= (Sunny,?,?,Strong,?,?)

  h_2= (Sunny,?,?,?,?,?)

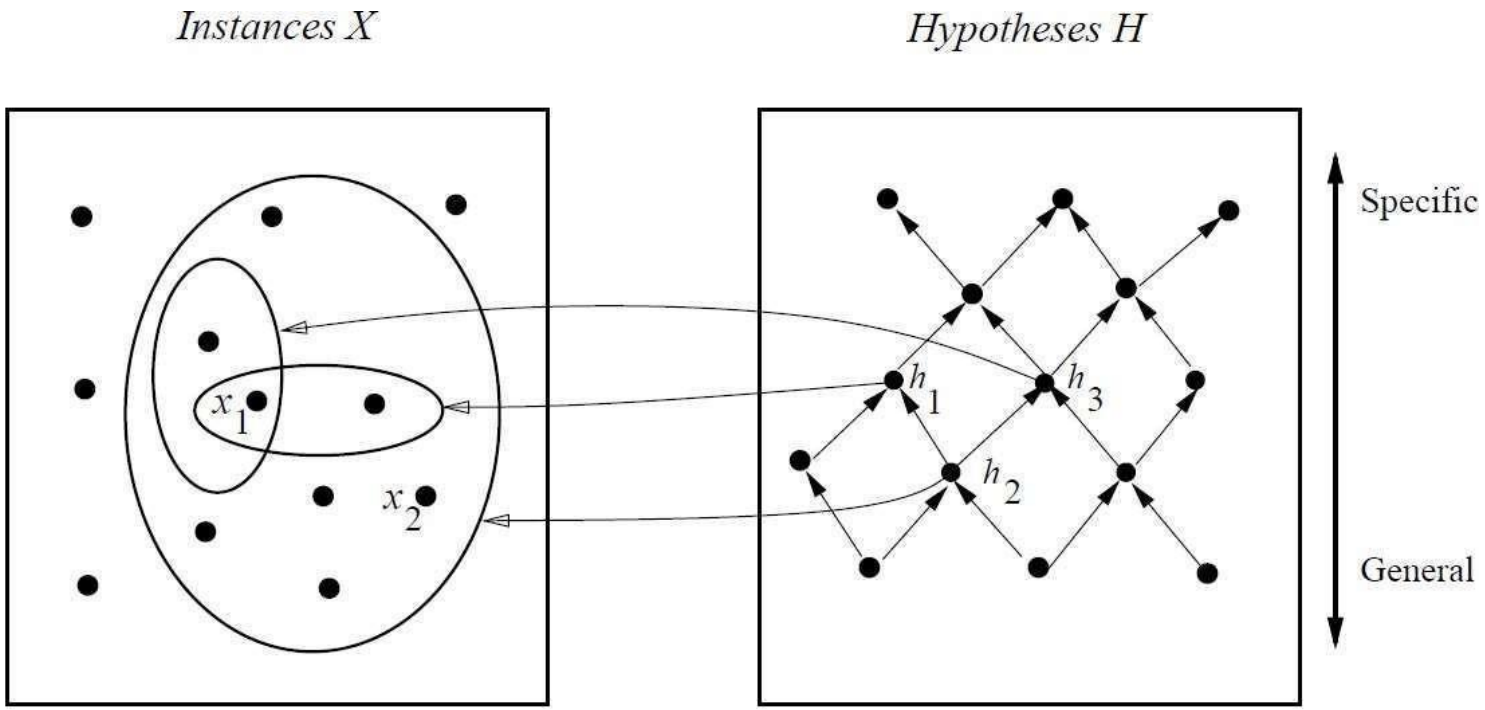
- Considerthesetsofinstancesthatareclassifiedpositivebyh$_l$andbyh$_2$.
- h$_2$imposesfewerconstraintsontheinstance, itclassifies more instancesaspositive. So, any instance classified positive by h$_l$ will also be classified positiveby h$_2$.Therefore, h$_2$is more generalthan h$_l$.

General-to-SpecificOrderingofHypotheses

- Givenhypotheses $h_j$andh_k,$h_j$ismore-general-thanor-equaldo$h_k$ifandonlyifanyinstancethatsatisfies $h_k$alsosatisfies$\boldsymbol{h}_i$

Definition:Let $h_j$and$h_k$beBoolean-valuedfunctionsdefinedoverX.Then$h_j$ismoregeneral-than-or-equal-toh_k(written$h_j \geq h_k$)ifand only if

$$(\forall x \in X)[(h_k(x) = 1) \rightarrow (h_j(x) = 1)]$$



x_1 = <Sunny, Warm, High, Strong, Cool, Same>
x_2 = <Sunny, Warm, High, Light, Warm, Same>

h_1 = <Sunny, ?, ?, Strong, ?, ?>
h_2 = <Sunny, ?, ?, ?, ?, ?>
h_3 = <Sunny, ?, ?, ?, Cool, ?>

- Inthe figure, thebox on the leftrepresentsthesetXofallinstances, the box on the right theset H ofall hypotheses.
- Eachhypothesiscorrespondstosomesubsetof X-thesubsetofinstancesthatitclassifies positive.
- The arrows connecting hypothesesrepresent the ***more - general - than***relation,withthearrowpointingtowardthelessgeneralhypothesis.
- Notethesubsetofinstancescharacterizedby$h_2$subsumesthesubset characterized by h_1 , henceh_2 is more-general–thanh_1

FIND-S: Finding a Maximally Specific Hypothesis

FIND-SAlgorithm

1. Initialize*h*tothemostspecifichypothesisin*H*

2. Foreachpositivetraininginstance*x*

Foreachattributeconstraint $a_i$in*h*

Iftheconstraint$a_i$issatisfiedby*x*

Thendonothing

Elsereplace$a_i$in*h* bythenextmoregeneralconstraintthatissatisfiedby *x*

3. Outputhypothesis*h*

Toillustratethisalgorithm,assumethelearnerisgiventhesequenceoftrainingexamp lesfrom the***EnjoySport***task

Example	Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

ThefirststepofFIND-Sistoinitialize***h***tothemostspecifichypothesisin***H***

h-(Ø,Ø,Ø, Ø,Ø,Ø)

x_1=<*SunnyWarmNormalStrongWarmSame*>,+

Observing the first training example, it is clear that our hypothesis is too specific. Inparticular, none of the "Ø" constraints in h are satisfied by this example, so each isreplacedbythe nextmore general constraintthat fitsthe example

h_1=<SunnyWarmNormalStrongWarmSame>

This h is still very specific; it asserts that all instances are negative except for thesinglepositive training example

x_2=<*Sunny*,*Warm*,*High*,*Strong*,*Warm*,*Same*>,+

The second training example forces the algorithm to further generalize h, this timesubstituting a "?' in place of any attribute value in h that is not satisfied by the newexample

h_2=<SunnyWarm?StrongWarmSame>

x3=<Rainy,Cold,High,Strong,Warm,Change>,-

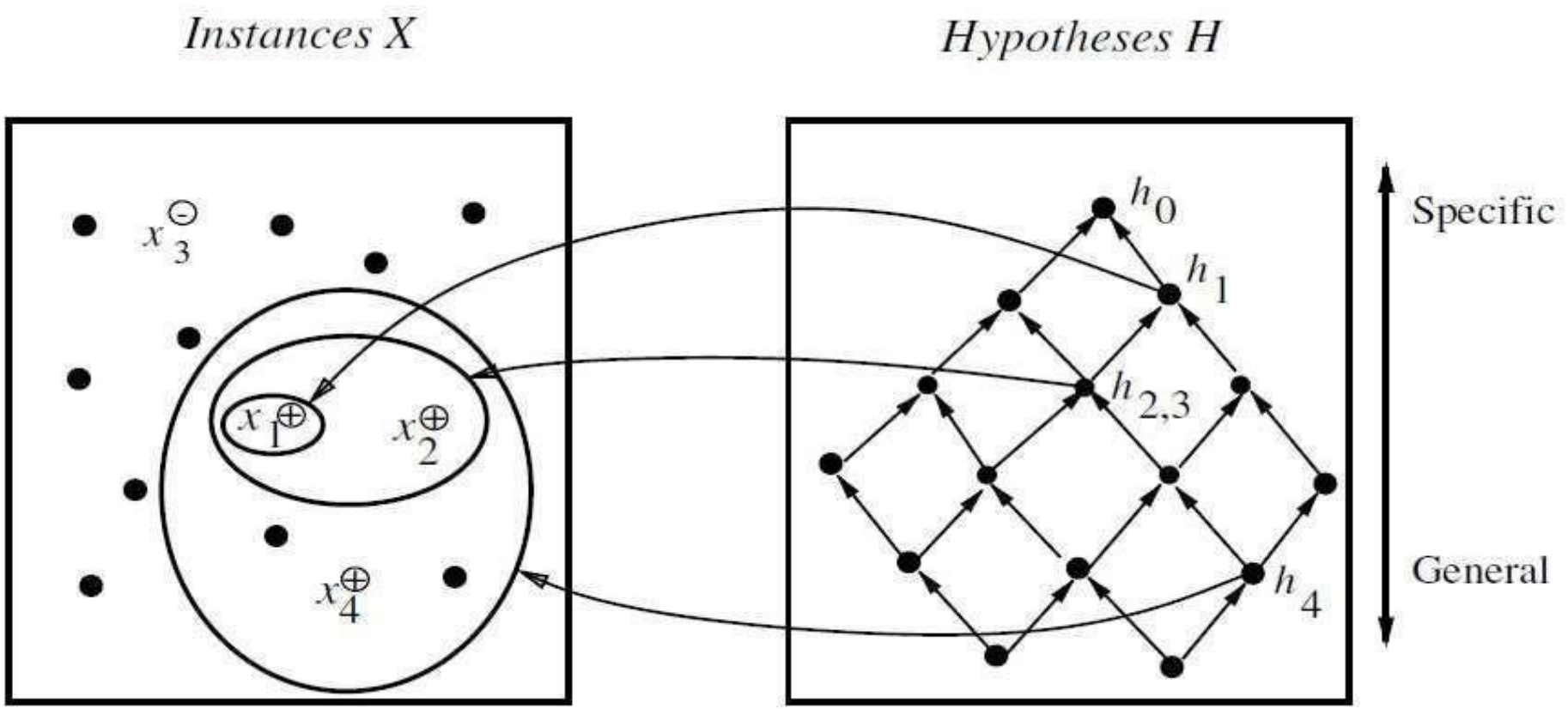
Uponencounteringthethirdtrainingthealgorithmmakesnochangetoh.TheFIND-Salgorithm simply ignores everynegative example.

h3=<SunnyWarm?StrongWarmSame>

x4=<SunnyWarmHighStrongCoolChange>,+

Thefourthexampleleadstoafurthergeneralizationof h

h4=<SunnyWarm?Strong??>



x_1 = <Sunny Warm Normal Strong Warm Same>, +

x_2 = <Sunny Warm High Strong Warm Same>, +

x_3 = <Rainy Cold High Strong Warm Change>, -

x_4 = <Sunny Warm High Strong Cool Change>, +

h_0 = <∅, ∅, ∅, ∅, ∅, ∅>

h_1 = <Sunny Warm Normal Strong Warm Sam

h_2 = <Sunny Warm ? Strong Warm Same>

h_3 = <Sunny Warm ? Strong Warm Same>

h_4 = <Sunny Warm ? Strong ? ? >

ThekeypropertyoftheFIND-Salgorithmis

- FIND-S is guaranteed to output the most specific hypothesis within H that isconsistentwith thepositive training examples
- FIND-S algorithm’s final hypothesis will also be consistent with the negativeexamples provided the correct target concept is contained in H, and provided thetrainingexamples are correct.

UnansweredbyFIND-S

1. Has the learner convergedtothecorrecttargetconcept?
2. Why prefer the mostspecifichypothesis?
3. Arethetrainingexamplesconsistent?
4. Whatiftherearesevеralmaximallyspecificconsistenthypotheses?

VersionSpaceandCANDIDATEELIMINATIONAlgorithm

ThekeyideaintheCANDIDATE-ELIMINATIONalgorithmistooutputadescriptionofthesetofall***hypothesesconsistentwiththetrainingexamples***

Representation

- ***Definition:***Ahypothesishis**consistent**with**a**setoftrainingexamples***D***if**and**onlyif ***h(x)*=*c(x)***foreachexample(x,***c(x))***in***D.***

$$Consistent\ (h,D) \equiv (\forall \langle x, c(x) \rangle \in D) h(x) = c(x))$$

Notedifferencebetweendefinitionsof*consistent*and*satisfies*

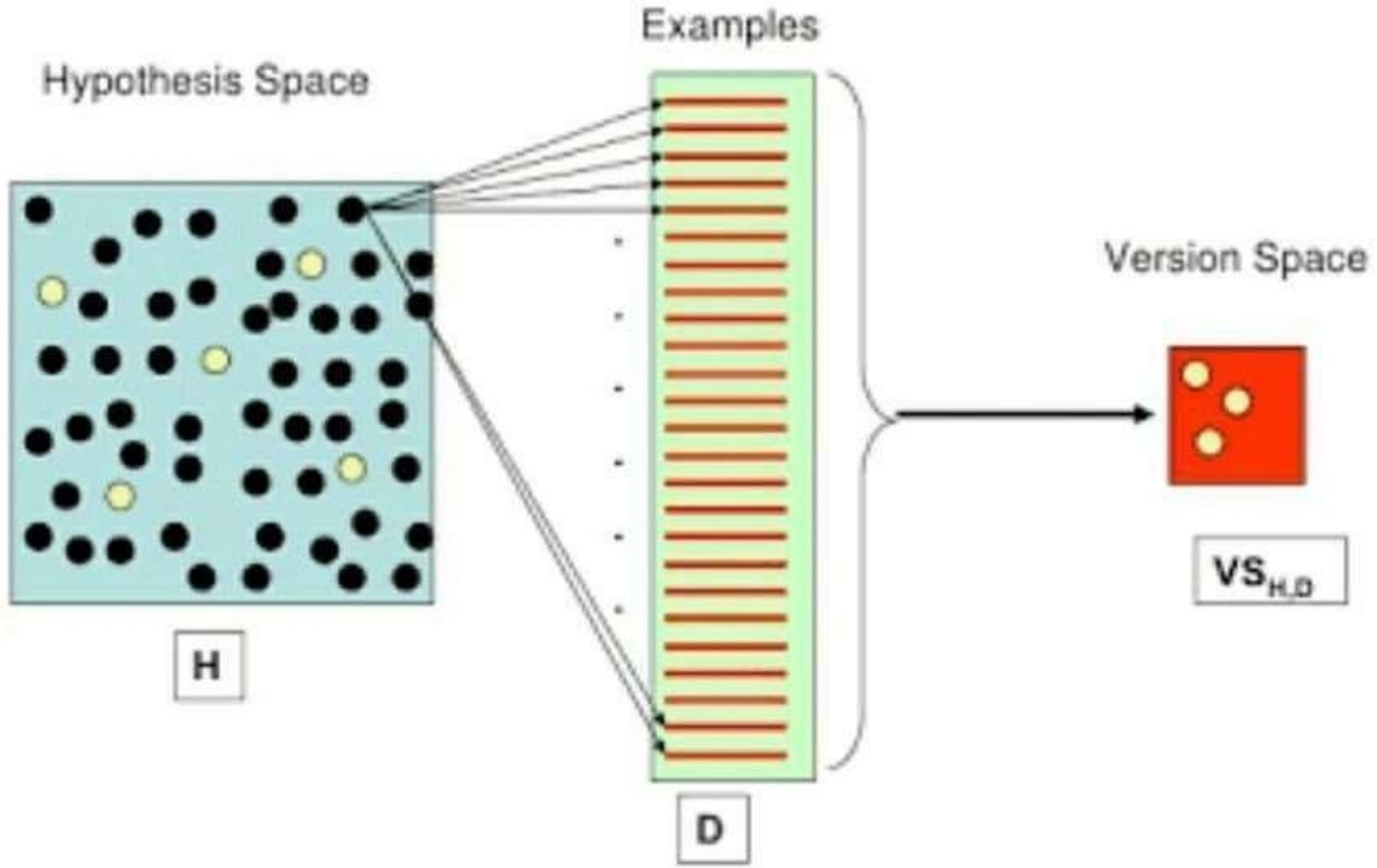
- Anexample *x*issaidto***satisfy***hypothesis***h***when***h(x)*** =1, regardless ofwhether*x*isapositiveornegativeexampleofthetargetconcept.
- anexample*x*issaidto***consistent***withhypothesis***h***iff***h(x)*=*c(x)***

VersionSpace

ArepresentationofthesetofallhypotheseswhichareconsistentwithD

Definition: The **version space,** denoted $VS_{H,D}$with respect to hypothesisspace ***H***and training examples D, is the subset of hypotheses from ***H*** consistent with thetraining examplesin D

$$VS_{H,D} \equiv \{h \in H | \ Consistent(h,D)\}$$


Examples
Hypothesis Space
Version Space
H
D
VS H,D

The LIST-THEN-ELIMINATE Algorithm

TheLIST-THEN-ELIMINATEalgorithmfirstinitializestheversionspacetocontainall hypotheses in H and then eliminates any hypothesis found inconsistent with anytraining example.

The LIST-THEN-ELIMINATE Algorithm

1. ***VersionSpace*** *c* a list containing every hypothesis in H
2. For each training example, ***(x,c(x))***

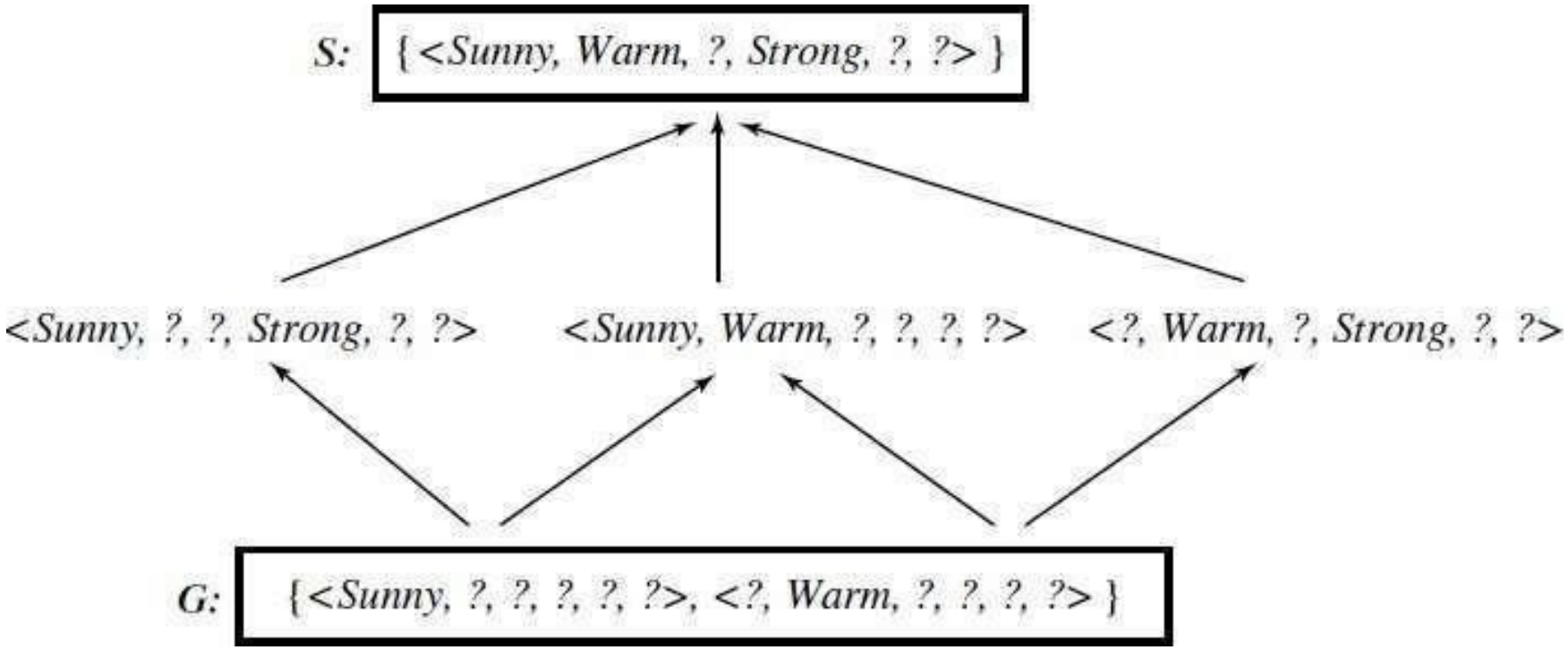
   remove from ***VersionSpace*** any hypothesis ***h*** for which ***h(x)≠c(x)***
3. Output the list of hypotheses in ***VersionSpace***

The LIST-THEN-ELIMINATE Algorithm

- **List-Then-Eliminate** works in principle, so long as version space is finite.
- However, since it requires exhaustive enumeration of all hypotheses in practice it is not feasible.

AMoreCompactRepresentationforVersionSpaces

- Theversionspaceisrepresentedbyitsmostgeneralandleastgeneralmembers.
- Thesemembersformgeneralandspecificboundarysetsthatdelimittheversionspace withinthe partiallyordered hypothesis space.



- Aversionspacewithitsgeneral and specific boundarysets.
- The version space includes allsix hypotheses shown here, butcanberepresentedmoresimply by ***S***and***G.***
- Arrows indicate instance of the*more-general-than* relation.Thisistheversion spaceforthe *Enjoysport*conceptlearning
- problemandtrainingexamples describedinbelowtable

Example	Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

Definition: The **generalboundary** G, with respecttohypothesisspace ***H*** andtrainingdata *D* ,is the setof maximallygeneral members of *H* consistentwith *D*

$$G \equiv \{g \in H | Consistent(g, D) \wedge (\neg \exists g' \in H)[(g' >_g g) \wedge Consistent(g', D)]\}$$

Definition: The **specificboundary** S, with respecttohypothesisspace *H* andtraining data *D*, is the set of minimally general (i.e., maximally specific) members of *H* consistentwith *D*.

$$S \equiv \{s \in H | Consistent(s, D) \wedge (\neg \exists s' \in H)[(s >_g s') \wedge Consistent(s', D)]\}$$

VersionSpacerepresentationtheorem

Theorem: Let X be an arbitrary set of instances and Let H be a set ofBoolean-valued hypotheses defined over X. Let c : X →{O, 1} be an arbitrary target conceptdefined over X, and let D be an arbitrary set of training examples {(x, c(x))). For allX, H,c, and D such that Sand G are well defined,

$$VS_{H,D}=\{h\in H|(\exists s\in S)(\exists g\in G)(g\geq_g h\geq_g s)\}$$

$$VS_{H,D}=\{h\in H|(\exists s\in S)(\exists g\in G)(g\geq_g h\geq_g s)\}$$

ToProve:

1. Everyhsatisfyingtherighthandsideoftheaboveexpressionisin$\boldsymbol{VS_{H,D}}$

2. Everymemberof$\boldsymbol{VS_{H,D}}$satisfiestheright-handsideoftheexpression

Sketchofproof:

1. letg,h,sbearbitrarymembersofG,H,Srespectivelywith$g\geq_g h\geq_g s$

By the definition of ***S,*** **s** must be satisfied by all positive examples in D. Because $h \geq_g s$,h must alsobe satisfied byall positive examples inD.

By the definition of ***G,*** g cannot be satisfied by any negative example in D, and because $g \geq_g h$ hcannot be satisfied by any negative example in D. Because h is satisfied by all positive examples in Dandbynonegativeexamples inD,his consistent with D,and thereforehisa memberof$VS_{H,D}$

2. Itcanbeprovenbyassumingsomehin$VS_{H,D}$,thatdoesnotsatisfytheright-handsideoftheexpression,thenshowing thatthis leadstoan inconsistency

TheCANDIDATE-ELIMINATIONLearningAlgorithm

The CANDIDATE-ELIMINTION algorithm computes the ***version space*** containingall hypotheses from H that are consistent with an observed sequence of trainingexamples.

Initialize G to the set of maximally general hypotheses in HInitialize S to the set of maximally specific hypotheses in HForeachtraining example d, do

- Ifdisapositiveexample
  - RemovefromGanyhypothesisinconsistentwithd
  - Foreach hypothesissinSthatisnotconsistentwithd
    - RemovesfromS
    - AddtoSallminimalgeneralizationshofssuchthat
      - hisconsistentwithd,andsomememberofGismoregeneralthanh
    - RemovefromSany hypothesisthatismoregeneralthananotherhypothesisinS

- Ifdisanegativeexample
  - Removefrom Sany hypothesisinconsistentwithd
  - ForeachhypothesisginGthatisnotconsistentwithd
    - RemovegfromG
    - AddtoGallminimalspecializationshofgsuchthat
      - hisconsistentwithd,andsomememberofSismorespecificthanh
    - RemovefromGanyhypothesisthatislessgeneral thananotherhypothesisinG

AnIllustrativeExample

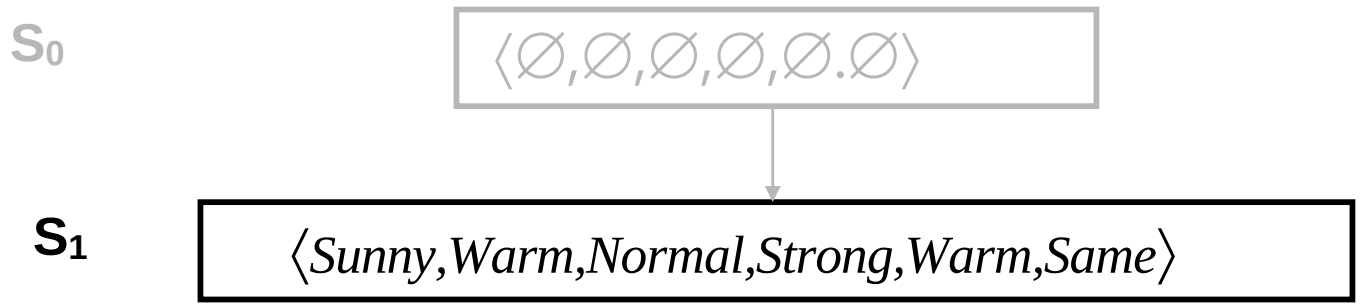
Theboundarysetsarefirstinitializedto$G_o$andS_o,themostgeneralandmostspecifichypothesesin H.

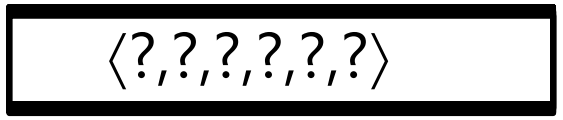
S_0 $\langle \varnothing, \varnothing, \varnothing, \varnothing, \varnothing, \varnothing \rangle$

G_0 $\langle ?, ?, ?, ?, ?, ? \rangle$

Fortrainingexampled,

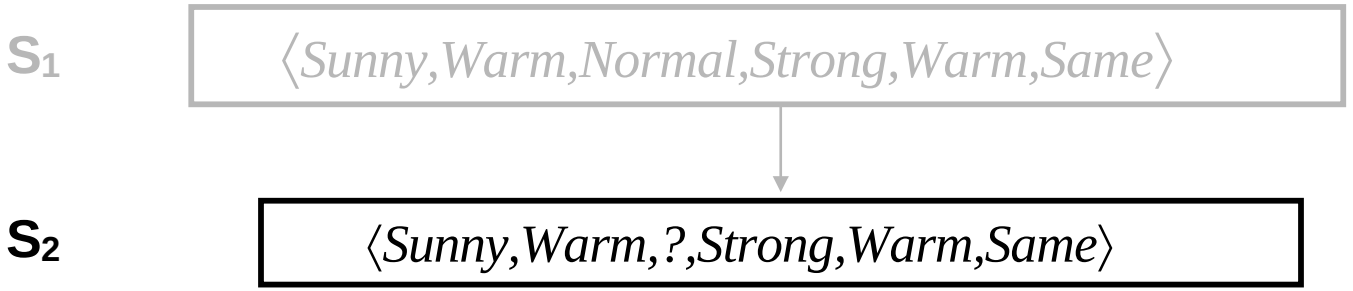
⟨Sunny,Warm,Normal,Strong,Warm,Same⟩+





Fortrainingexampled,

⟨Sunny,Warm,High,Strong,Warm,Same⟩+



G1,G2 ⟨?,?,?,?,?,?⟩

Fortrainingexampled,

⟨Rainy,Cold,High,Strong,Warm,Change⟩–

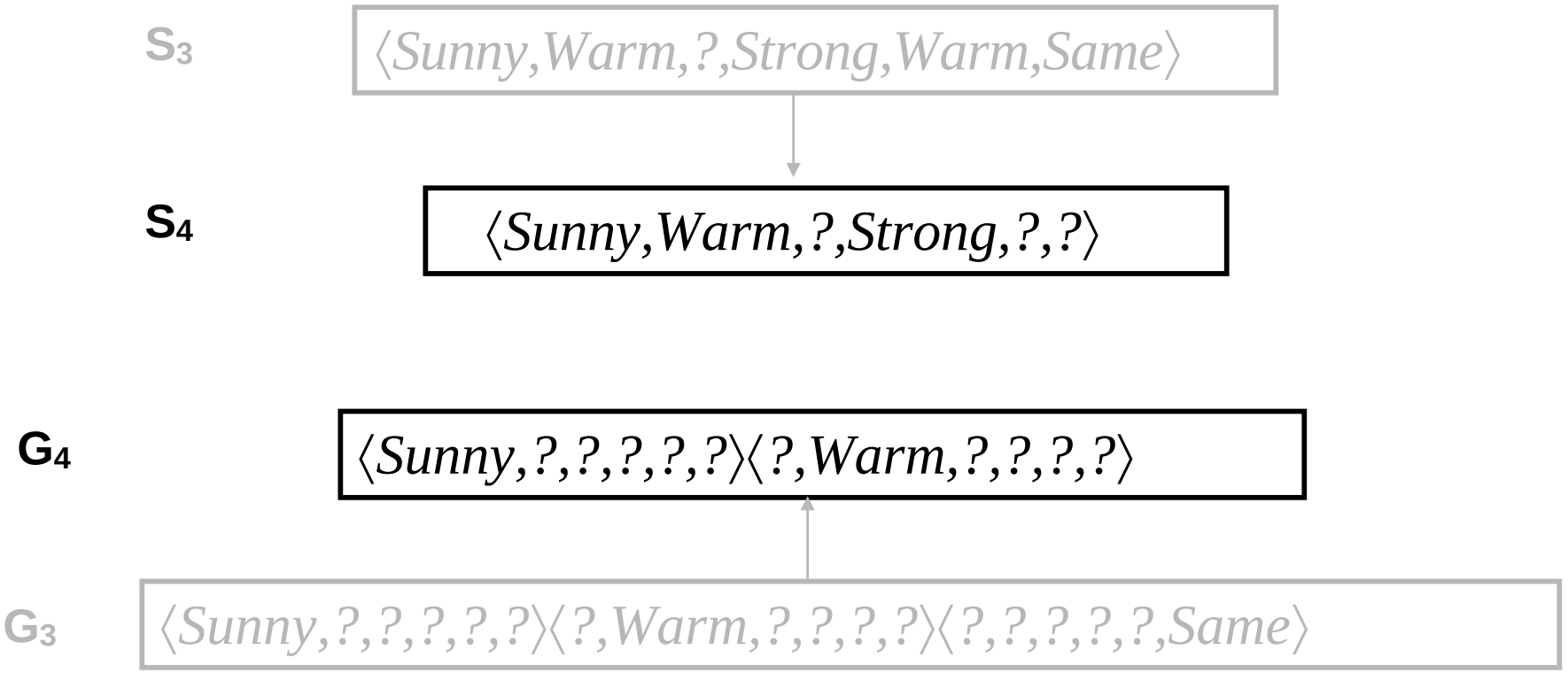
S_2, **S_3** ⟨*Sunny*,*Warm*,*?*,*Strong*,*Warm*,*Same*⟩

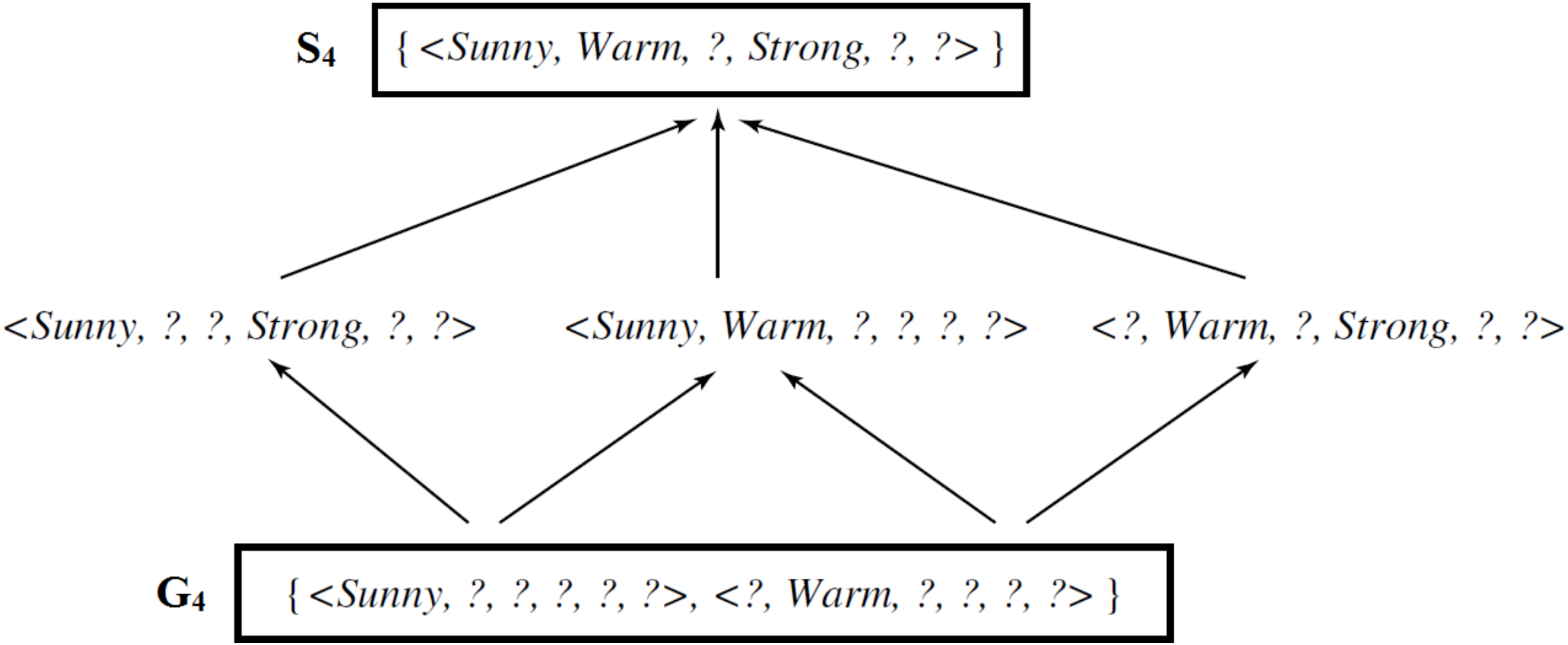
G_3 ⟨*Sunny*,*?*,*?*,*?*,*?*,*?*⟩⟨*?*,*Warm*,*?*,*?*,*?*,*?*⟩⟨*?*,*?*,*?*,*?*,*?*,*Same*⟩

G_2 ⟨?,?,?,?,?,?⟩

Fortrainingexampled,

⟨Sunny,Warm,High,Strong,CoolChange⟩+





The final version space for the ***EnjoySport***concept learning problem and training examplesdescribedearlier.

InductiveBias

Thefundamentalquestionsforinductiveinference

- Whatifthetargetconceptisnotcontainedinthehypothesisspace?
- Canweavoidthisdifficultybyusing ahypothesisspacethatincludeseverypossiblehypothesis?
- Howdoesthesizeofthishypothesisspaceinfluencetheabilityofthealgorithmtogeneralizet ounobservedinstances?
- Howdoesthesizeofthehypothesisspaceinfluencethenumberoftrainingexamples thatmustbeobserved?

Effectofincompletehypothesisspace

Precedingalgorithmsworkiftargetfunctionis in H
Willgenerallynotworkiftargetfunctionnotin H

Considerfollowingexampleswhichrepresenttargetfunction
"sky=sunnyorsky=cloudy":

⟨SunnyWarmNormalStrongCoolChange⟩	Y
⟨Cloudy WarmNormalStrongCoolChange⟩	Y
⟨RainyWarmNormalStrongCoolChange⟩	N

IfapplyCandidateEliminationalgorithmasbefore,endupwithemptyVersionSpaceAfterf

irst twotrainingexample
S=⟨?WarmNormalStrongCoolChange⟩

Newhypothesisisoverly generalanditcoversthethirdnegativetrainingexample!Our

Hdoesnot includetheappropriatec

AnUnbiasedLearner

Incompletehypothesisspace

- IfcnotinH,thenconsidergeneralizingrepresentationofHtocontainc
- The size of the instance space X of days described by the six available attributes is 96.The number of distinct subsets that can be defined over a set X containing |X| elements(i.e., thesizeofthepowersetofX)is$2^{|X|}$
- Recallthatthereare96instancesin***EnjoySport***;hencethereare2^{96}possiblehypothesesinfullspaceH
- CandothisbyusingfullpropositionalcalculuswithAND,OR,NOT
- HenceHdefinedonlybyconjunctionsofattributesisbiased(containingonly973h's)

- Let us reformulate the ***Enjoysport***learning task in an unbiased way by defining a newhypothesisspace***H'***thatcanrepresenteverysubsetofinstances;thatis,letH'correspondtoth epowersetofX.
- Onewayto definesuchan H'istoallowarbitrarydisjunctions,conjunctions,andnegationsofourearlierhypothes es.

Forinstance,thetargetconcept***"Sky=Sunny***or***Sky=Cloudy"***couldthenbedescribedas

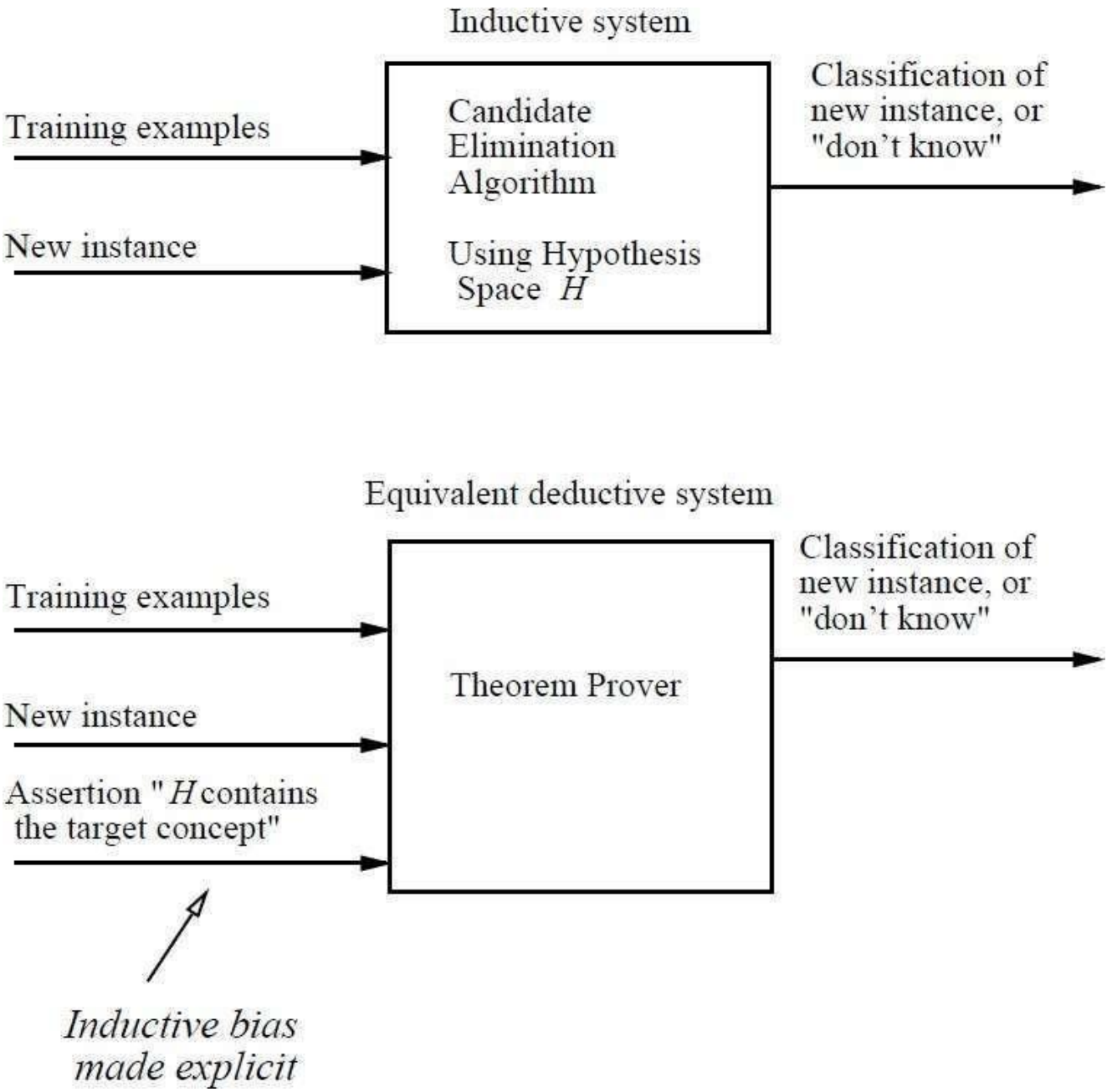
(Sunny,?,?,?,?,?)V***(Cloudy,***?,?,?,?,?)

Definition:

Consideraconceptlearningalgorithm L forthesetofinstances X.

- Letcbeanarbitraryconceptdefinedover X
- Let D_c={(x,c(x))}beanarbitrarysetoftrainingexamplesofc.
- Let $L(x_i,D_c)$denotetheclassificationassignedtotheinstance x_i by L aftertrainingonthedata D_c.
- TheinductivebiasofLisany minimalsetofassertionsBsuch thatforanytargetconceptcandcorrespondingtrainingexamples D_c

$$(\forall\langle x_i \in X)[(B \wedge D_c \wedge x_i) \vdash L(x_i,D_c)]$$



Modelling inductive systems byequivalentdeductivesystems.

Theinput-outputbehaviorofthe CANDIDATE-ELIMINATION algorithm using a hypothesis space ***H*** is identical to that of a deductivetheorem prover utilizing the assertion ***"H*** contains the target concept." Thisassertion is therefore called the***inductive bias*** of the CANDIDATE-ELIMINATION algorithm.

.

characterizinginductive systems by their inductive bias allowsmodelling them by their equivalentdeductive systems. This provides awayto compare inductive systemsaccording to their policies forgeneralizing beyond the observedtrainingdata

DECISIONTREELEARNING

Decision tree learning is a method for approximatingdiscrete-valued target functions,in which thelearnedfunction is representedby adecision tree.

DECISIONTREEREPRESENTATION

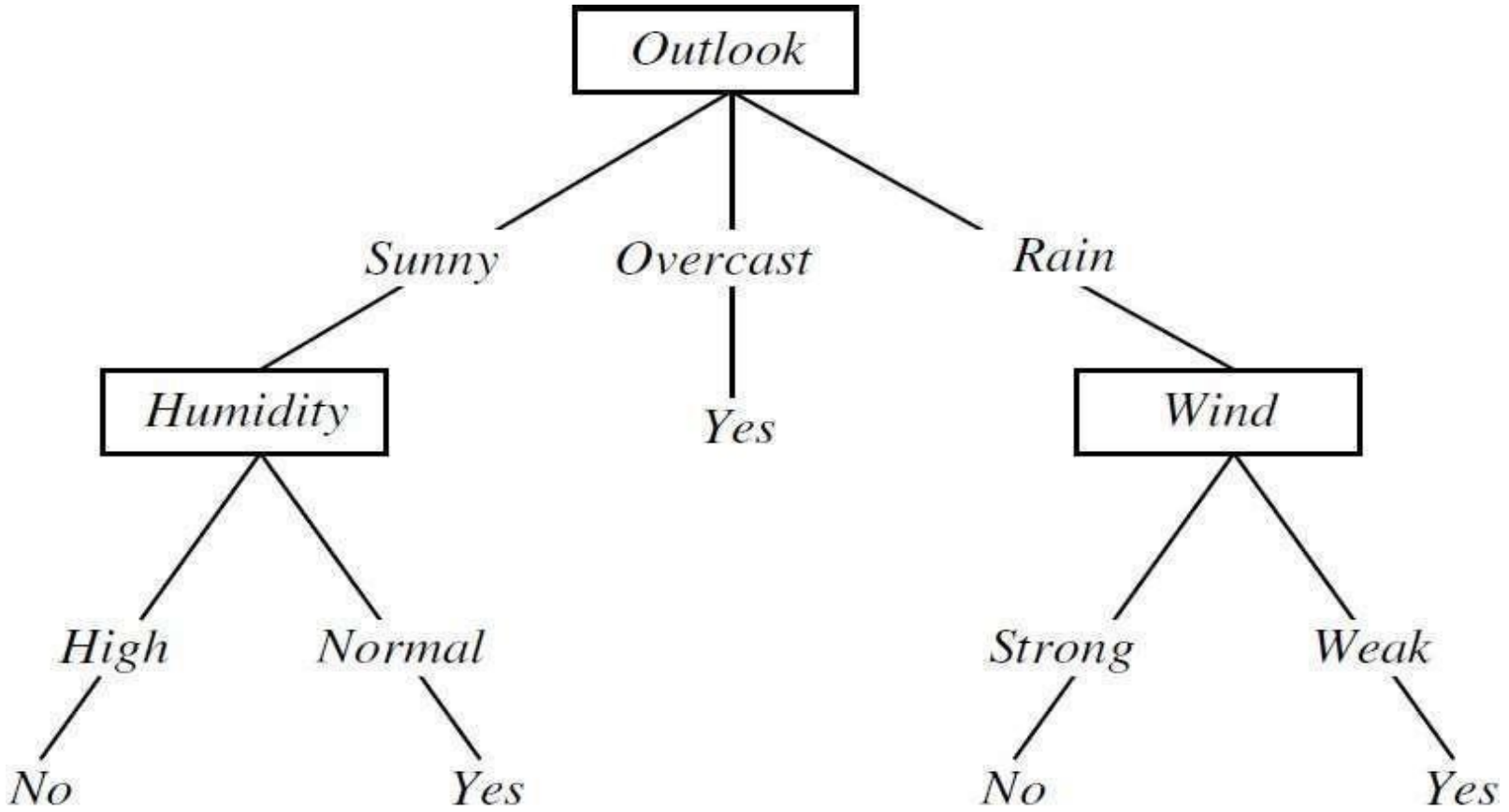


FIGURE: A decision tree for theconcept ***PlayTennis***.Anexam pleisclassified by sortingit through the tree totheappropriateleaf node, then returningtheclassification associated with thisleaf

- Decision trees classify instances by sorting them down the tree from the root tosome leaf node, which provides the classificationof the instance.

- Each node in the tree specifies a test of some attribute of the instance, and eachbranch descending from that node corresponds to one of the possible values forthisattribute.

- An instance is classified by starting at the root node of the tree, testing theattribute specified by this node, then moving down the tree branch correspondingtothevalueoftheattributeinthegivenexample.Thisprocessisthenrepeatedforthe subtree rootedat the new node.

- Decisiontreesrepresentadisjunctionofconjunctionsofconstraintsontheattributevalues of instances.
- Eachpathfromthetreeroottoaleafcorrespondstoaconjunctionofattributetests,and the treeitself toa disjunction of these conjunctions

Forexample,
The decision trees how n in above figure corresponds to the
expression (Outlook =Sunny AHumidity = Normal)
(Outlook=Overcast)
(Outlook=RainAWind=Weak)

APPROPRIATEPROBLEMSFOR DECISIONTREELEARNING

Decisiontreelearningisgenerallybestsuitedtoproblemswiththefollowingcharacteristics:

1. ***Instancesarerepresentedbyattribute-valuepairs***–Instancesaredescribedbyafixed set of attributes and their values
2. ***The target function has discrete output values*** – The decision tree assigns aBoolean classification (e.g., yes or no) to each example. Decision tree methodseasilyextendtolearningfunctions withmorethantwo possibleoutputvalues.
3. ***Disjunctivedescriptionsmayberequired***

4. ***The training data may contain errors*** – Decision tree learning methods arerobust to errors, both errors in classifications of the training examples and errorsinthe attribute values that describe these examples.

5. ***Thetrainingdatamaycontainmissingattributevalues–*** Decisiontreemethodscanbe used evenwhen some training exampleshaveunknown values

- Decision tree learning has been applied to problems such as learning to classify*medical patients by their disease*, *equipment malfunctions by their cause*, and*loan applicants by their likelihood ofdefaulting on payments*.

- Such problems, in which the task is to classify examples into one of a discrete setofpossible categories, are often referred toas***classification problems.***

THEBASICDECISIONTREE LEARNING ALGORITHM

- Mostalgorithmsthathavebeendevelopedforlearningdecisiontreesarevariations on a core algorithm that employs a top-down, greedy search through thespaceofpossibledecisiontrees.ThisapproachisexemplifiedbytheID3algorithmand itssuccessor C4.5

WhatistheID3algorithm?

- ID3standsforIterativeDichotomiser3
- ID3isaprecursortotheC4.5Algorithm.
- TheID3algorithmwasinventedbyRossQuinlanin1975
- Used to generate a decision tree from a given data set by employing a top-down,greedysearch, to test each attribute at every nodeof thetree.
- Theresultingtreeisusedtoclassifyfuturesamples.

ID3algorithm

ID3(Examples,Target_attribute,Attributes)

Examples are the training examples. Target_attribute is the attribute whose value is tobe predictedby the tree. Attributes is a list of other attributes that may be tested by the learned decision tree.Returnsadecisiontree that correctlyclassifies thegiven Examples.

- CreateaRootnodeforthetree
- IfallExamplesarepositive,Returnthesingle-nodetreeRoot,with label=+
- IfallExamplesarenegative,Returnthesingle-nodetreeRoot,withlabel=-
- IfAttributesisempty,Returnthesingle-nodetreeRoot,withlabel=mostcommonvalueofTarget_attributein Examples

- OtherwiseBegin
  - A←theattributefromAttributesthatbest*classifiesExamples
  - ThedecisionattributeforRoot←A
  - Foreachpossiblevalue,v_i,ofA,
    - Addanewtreebranchbelow*Root*,correspondingtothetestA=v_i
    - Let$Examples_{vi}$,bethesubsetofExamplesthathavevalue$v_i$for*A*
    - If$Examples_{vi}$,isempty
      - Thenbelowthisnewbranchaddaleafnodewithlabel=mostcommonvalueofTarget_attribuleinExamples
      - Elsebelowthisnewbranchaddthesubtree ID3(*Examples*$_{vi}$,Targe_tattribute,Attributes–{A}))
- End
- ReturnRoot

* Thebestattributeistheonewithhighestinformation gain

WhichAttributeIstheBestClassifier?

- The central choice in the ID3 algorithm is selecting which attribute to test at eachnode in thetree.
- Astatisticalpropertycalled***informationgain***thatmeasureshowwellagivenattributesepa ratesthetraining examples accordingto theirtargetclassification.
- ID3uses***informationgain***measuretoselectamongthecandidateattributesateach step while growingthe tree.

ENTROPYMEASURESHOMOGENEITYOFEXAMPLES

- Todefineinformationgain,webeginbydefiningameasurecalledentropy. *Entropymeasurestheimpurityofacollectionofexamples.*
- GivenacollectionS,containingpositiveandnegativeexamplesofsometargetconcept,theentropy of Srelative to this Booleanclassificationis

$$Entropy\,(S) \equiv -p_{\oplus} log_2 p_{\oplus} - p_{\ominus} log_2 p_{\ominus}$$

Where,

p_+ istheproportionofpositiveexamplesinS

p_- istheproportionofnegativeexamplesinS.

Example:Entropy

- Suppose S is a collection of 14 examples of some boolean concept, including 9positive and 5 negative examples. Then the entropy of S relative to this booleanclassificationis

$$Entropy([9+, 5-]) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14)$$
$$= 0.940$$

- Theentropyis0ifallmembersofSbelongtothesameclass
- Theentropyis1whenthecollectioncontainsanequalnumberofpositiveandnegative examples
- Ifthecollectioncontainsunequalnumbersofpositiveandnegativeexamples,theentropyis between 0 and 1

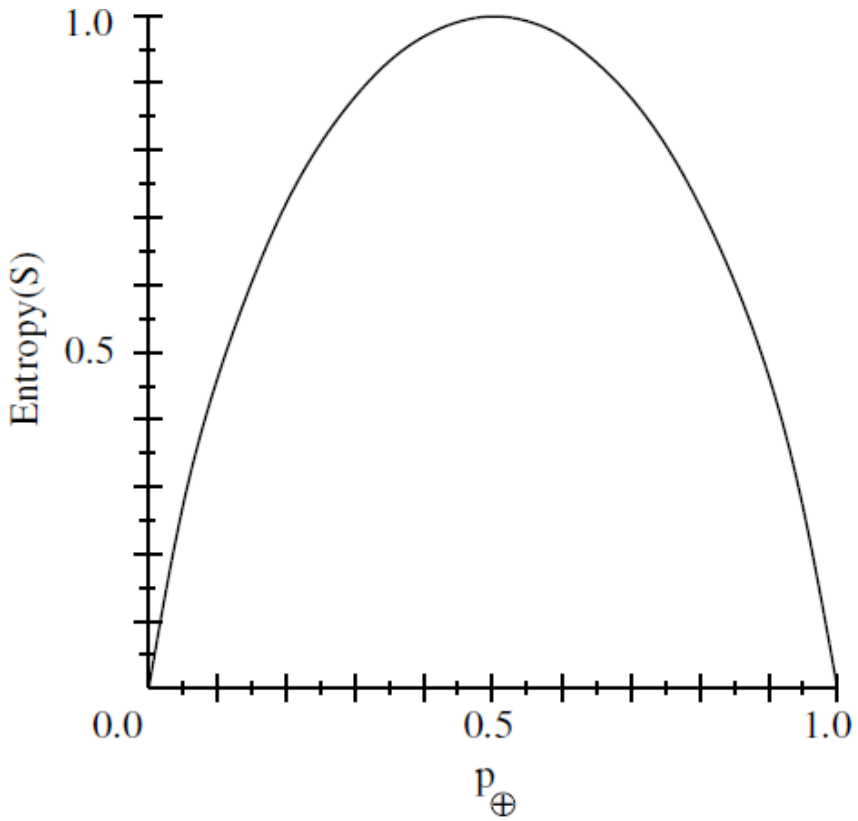


FIGURE The entropy function relative to a boolean classification, as the proportion, $p_{\oplus}$, of positive examples varies between 0 and 1.

INFORMATIONGAINMEASURESTHEEXPECTEDREDUCTIONINENTROPY

- ***Information gain,*** is the expected reduction in entropy caused by partitioning theexamplesaccording tothis attribute.
- Theinformationgain,Gain(S,A)ofanattributeA,relativetoacollectionofexamples S, is defined as

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Example:Informationgain

Let, $Values(Wind)=\{Weak, Strong\}$

$S = [9+, 5-]$

$S_{Weak} = [6+, 2-]$

$S_{Strong} = [3+, 3-]$

Informationgainofattribute *Wind*:

$$Gain(S, Wind) = Entropy(S) - 8/14\, Entropy(S_{Weak}) - 6/14\, Entropy(S_{Strong})$$

$$= 0.94 - (8/14)*0.811 - (6/14)*1.00$$

$$= 0.048$$

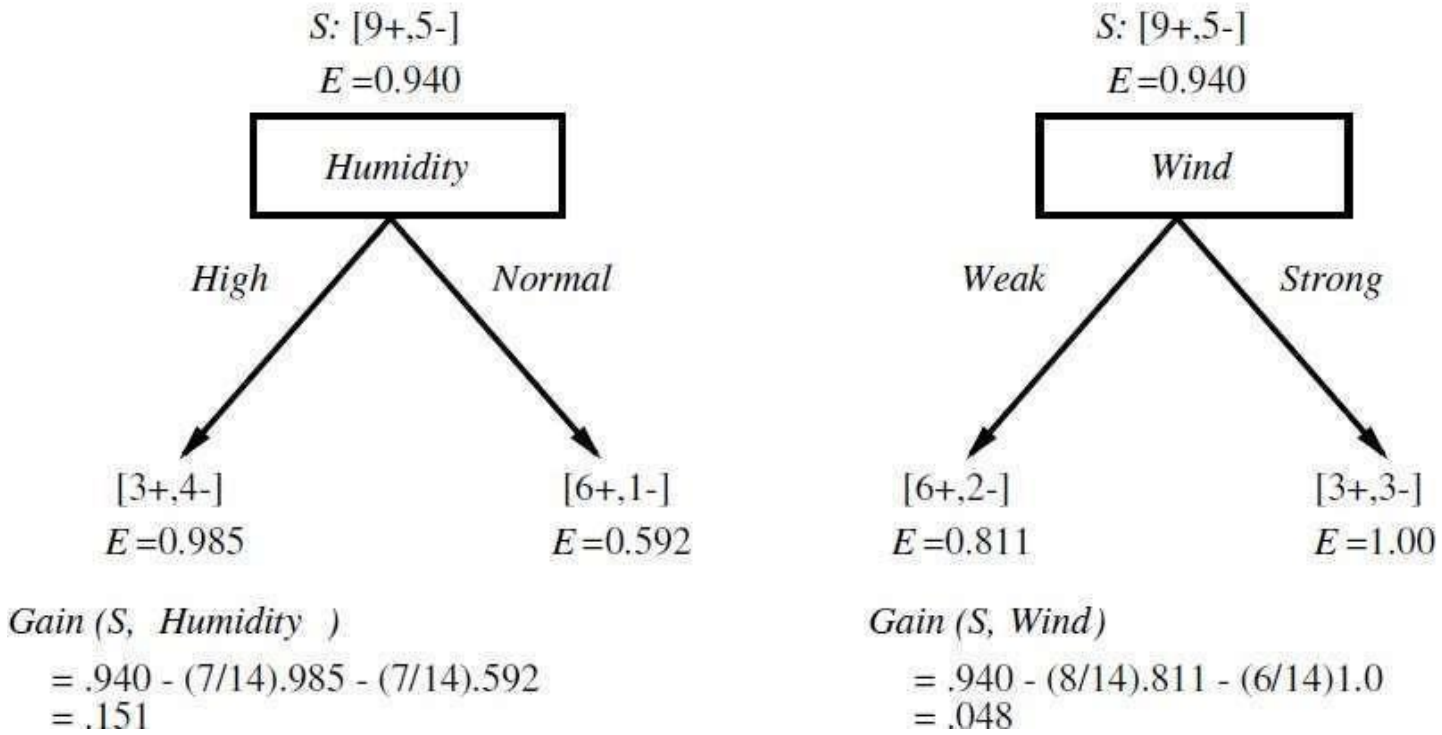
AnIllustrativeExample

- Toillustratetheoperation ofID3,considerthelearningtask representedbythetrainingexamples of below table.
- Herethetargetattribute***PlayTennis***,which canhave values ***yes*** or***no***fordifferentdays.
- Considerthefirststepthroughthealgorithm,inwhichthetopmostnodeofthedecision treeis created.

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

ID3 determines the information gain for each candidate attribute (i.e., Outlook,Temperature, Humidity, and Wind), then selects the one with highest informationgain

Which attribute is the best classifier?

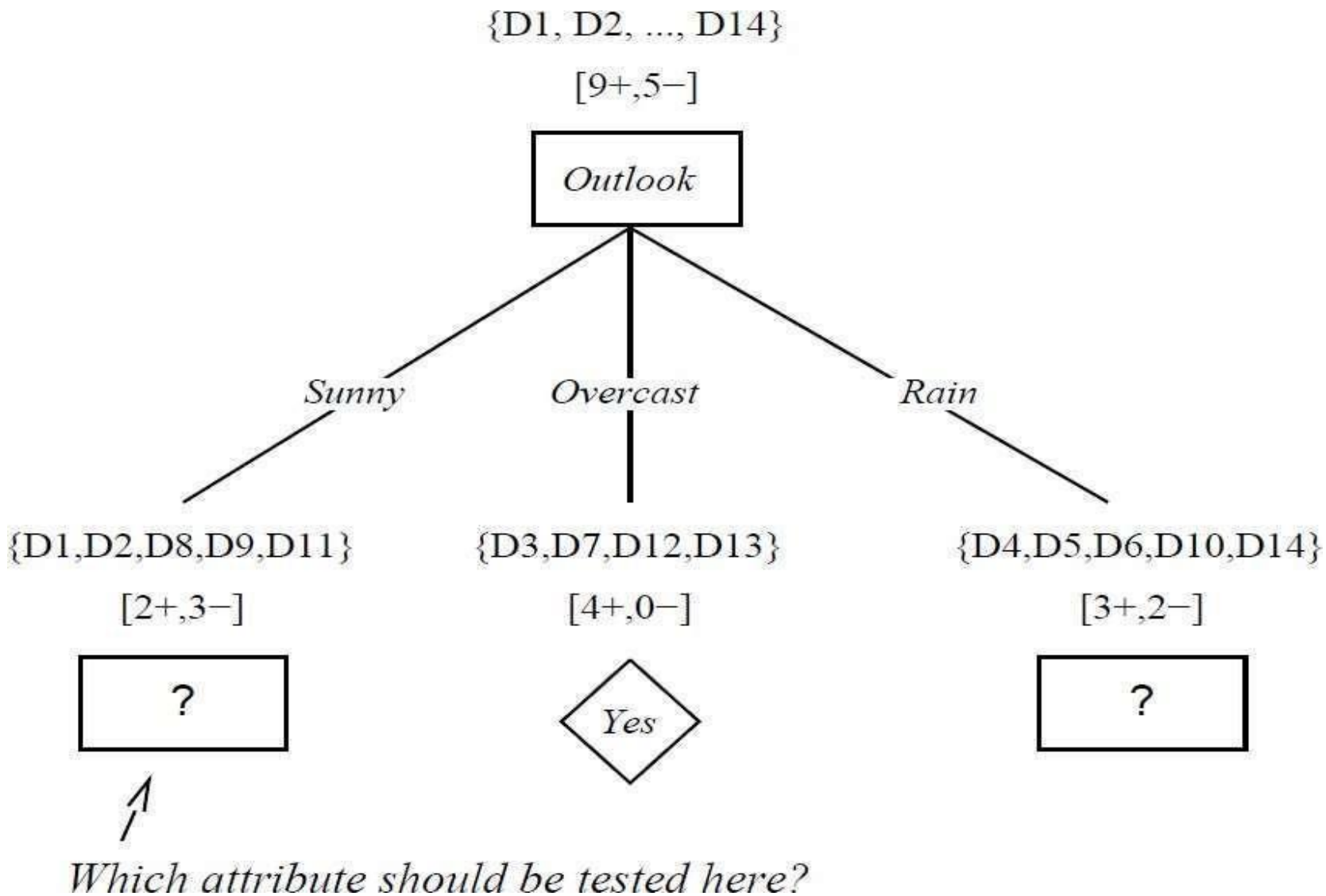


Theinformationgainvaluesforallfourattributesare

- Gain(S,Outlook) =0.246
- Gain(S,Humidity) =0.151
- Gain(S,Wind) =0.048
- Gain(S,Temperature) =0.029

- According to the information gain measure, the ***Outlook*** attribute provides thebest prediction of the target attribute, ***PlayTennis***, over the training examples.Therefore, ***Outlook*** is selected as the decision attribute for the root node, andbranches are created below the root for each of its possible values i.e.,

Sunny,Overcast,and Rain.


{D1, D2, ..., D14}
[9+,5−]
Outlook
Sunny
Overcast
Rain
{D1,D2,D8,D9,D11}
[2+,3−]
{D3,D7,D12,D13}
[4+,0−]
{D4,D5,D6,D10,D14}
[3+,2−]
?
Yes
?
Which attribute should be tested here?

$S_{sunny} = \{D1, D2, D8, D9, D11\}$

$Gain(S_{sunny}, Humidity) = .970 - (3/5)\,0.0 - (2/5)\,0.0 = .970$

$Gain(S_{sunny}, Temperature) = .970 - (2/5)\,0.0 - (2/5)\,1.0 - (1/5)\,0.0 = .570$

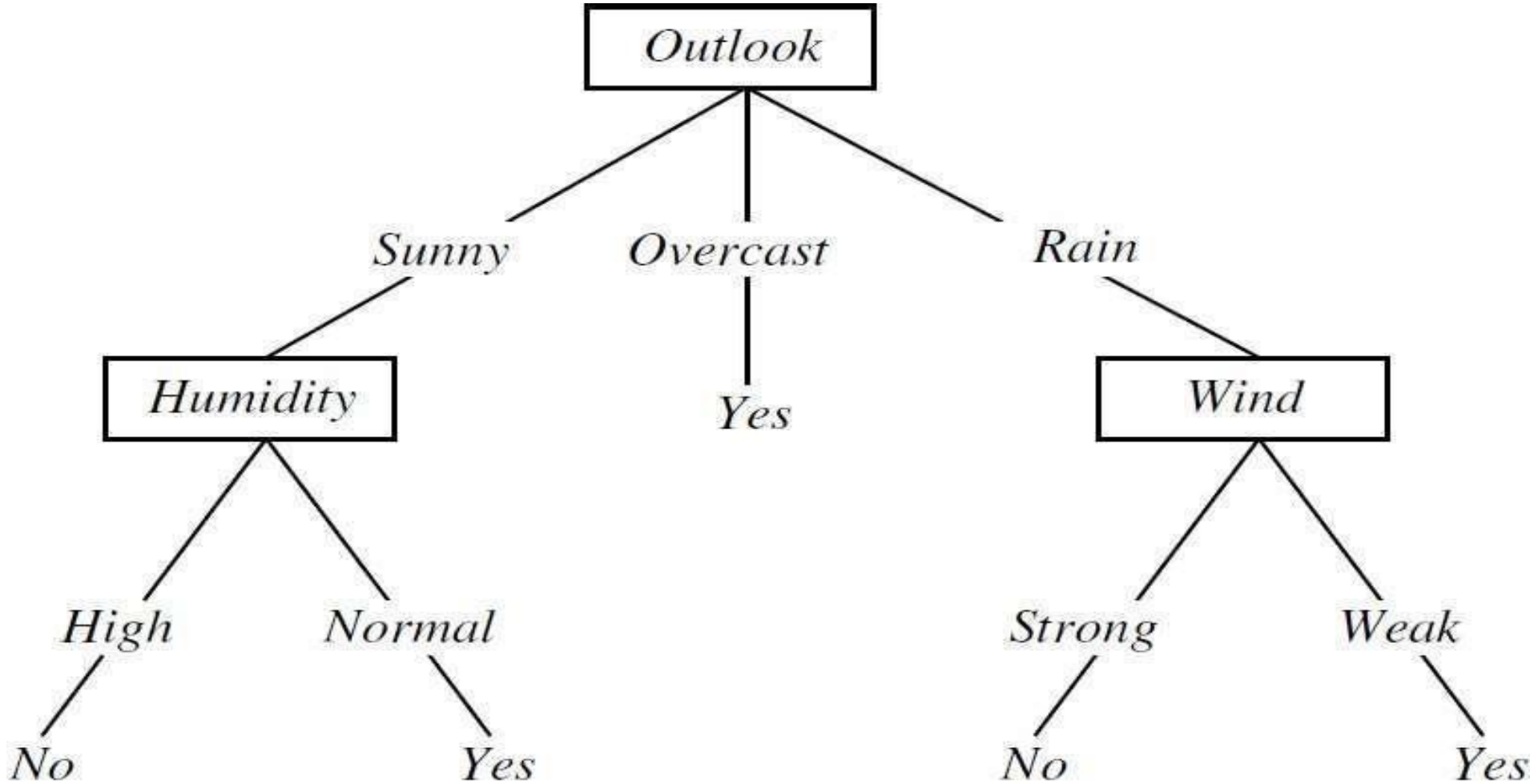
$Gain(S_{sunny}, Wind) = .970 - (2/5)\,1.0 - (3/5)\,.918 = .019$

$S_{Rain} = \{D4, D5, D6, D10, D14\}$

$Gain(S_{Rain}, Humidity) = 0.970 - (2/5)1.0 - (3/5)0.917 = 0.019$

$Gain(S_{Rain}, Temperature) = 0.970 - (0/5)0.0 - (3/5)0.918 - (2/5)1.0 = 0.019$

$Gain(S_{Rain}, Wind) = 0.970 - (3/5)0.0 - (2/5)0.0 = 0.970$


Outlook
Sunny
Overcast
Rain
Humidity
Yes
Wind
High
Normal
Strong
Weak
No
Yes
No
Yes

HYPOTHESISSPACESEARCHINDECISIONTREEL EARNING

- ID3 can be characterized as searching a space of hypotheses for one that fits thetraining examples.
- ThehypothesisspacesearchedbyID3isthesetofpossibledecisiontrees.
- ID3 performs a ***simple-to complex, hill-climbing search*** through this hypothesisspace,beginningwiththeemptytree,thenconsideringprogressivelymoreelaboratehypothesesinsearchofadecisiontreethatcorrectlyclassifiesthetraining data

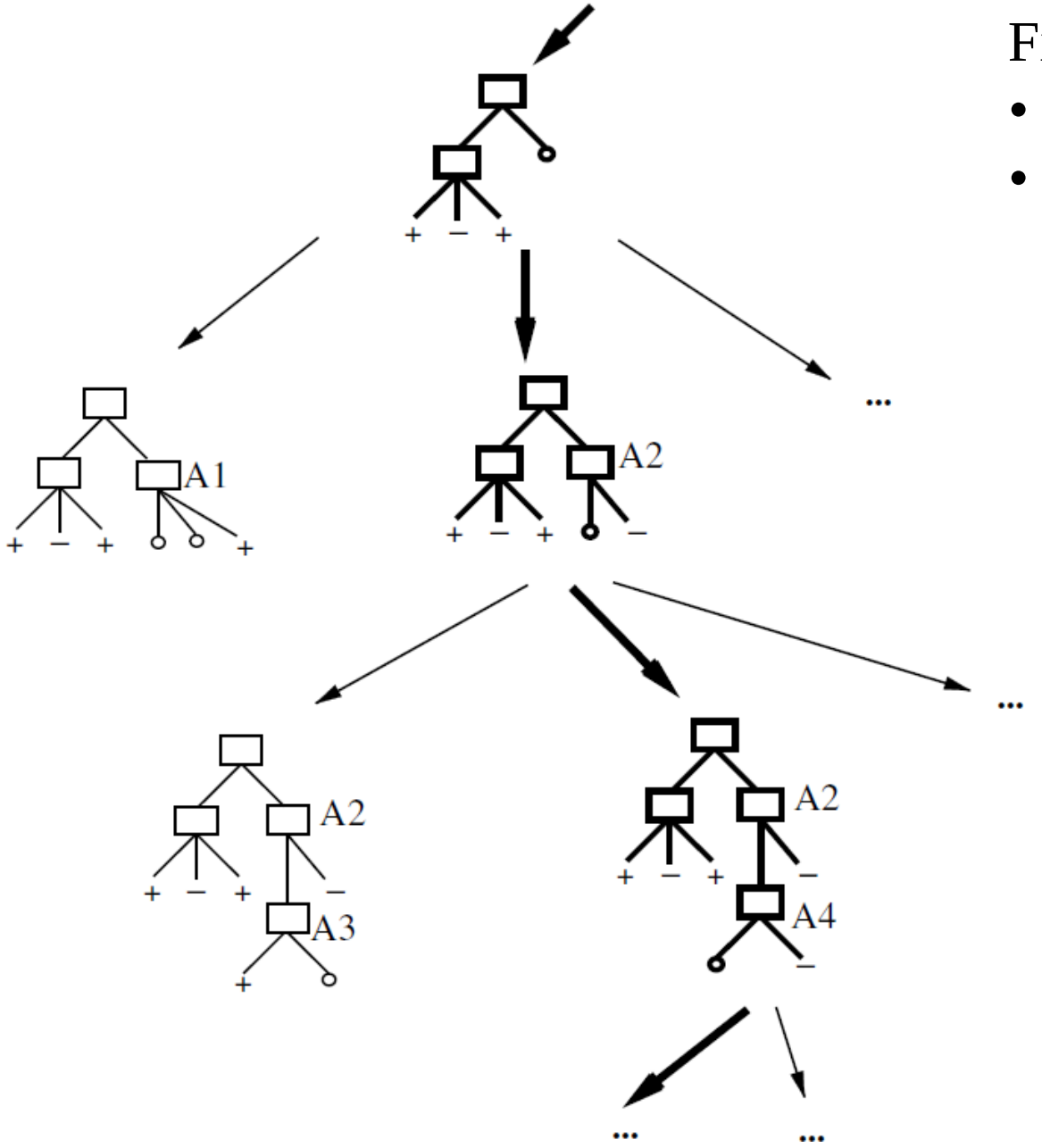


Figure:

- HypothesisspacesearchbyID3.
- ID3searchesthroughthespaceofpossible decision trees from simplest toincreasinglycomplex,guidedbytheinfo rmationgain heuristic

ByviewingID3intermsofitssearchspaceandsearchstrategy,wecanget someinsightintoits capabilities andlimitations

1. ID3's hypothesis space of all decision trees is a ***complete*** space of finite discrete-valued functions, relative to the available attributes. Because every finite discrete-valuedfunction can be represented by some decision tree

- ID3avoids oneofthemajorrisksofmethodsthat***searchincompletehypothesisspaces*** :that thehypothesis space mightnotcontain thetarget function.

2. ID3maintains***onlyasinglecurrenthypothesis***asitsearchesthroughthespaceof decision trees.

Forexample,withtheearlierversionspacecandidateeliminationmethod,whichmaint ains the set of ***all*** hypotheses consistent with the available trainingexamples.

Bydeterminingonlyasinglehypothesis,ID3losesthecapabilitiesthatfollowfromexplici tlyrepresenting allconsistenthypotheses.

For example, it does not have the ability to determine how many alternativedecision trees are consistent with the available training data, or to pose newinstancequeriesthat optimallyresolveamongthesecompetinghypotheses

3. **ID3** in its pure form performs ***no backtracking in its search***. Once it selects anattribute to test at a particular level in the tree, it never backtracks to reconsider thischoice.

- In the case of **ID3,** a locally optimal solution corresponds to the decision tree itselects along the single search path it explores. However, this locally optimalsolution may be less desirable than trees that would have been encountered along adifferentbranch of the search.

4. **ID3*usesalltrainingexamplesateachstep***inthesearchtomakestatisticallybaseddecisions regarding how to refineits current hypothesis.

- Oneadvantageofusingstatisticalpropertiesofalltheexamplesisthattheresultingsearch ismuch ***less sensitiveto errors***inindividualtraining examples.
- **ID3** can be easily extended to handle noisy training data by modifyingitsterminationcriteriontoaccepthypotheses thatimperfectlyfitthetrainingdata.

INDUCTIVEBIASINDECISIONTREELEARNING

Inductive bias is the set of assumptions that, together with the training data,deductivelyjustifytheclassificationsassignedbythelearnertofutureinstances

Givenacollectionoftrainingexamples,therearetypicallymanydecisiontreesconsistent with theseexamples.Which ofthese decisiontreesdoes ID3choose?

ID3searchstrategy

(a) selectsinfavourofshortertreesoverlongerones

(b) selectstreesthatplacethe attributeswith highestinformationgainclosesttotheroot.

ApproximateinductivebiasofID3: Shortertreesarepreferredoverlargertrees

- Consideranalgorithmthatbeginswiththeemptytreeandsearches***breadthfirst*** throughprogressivelymorecomplextrees.
- Firstconsideringalltreesofdepth1,thenalltrees ofdepth2,etc.
- Once itfinds a decision tree consistent with thetraining data, it returns thesmallestconsistenttreeatthatsearchdepth(e.g.,thetreewiththefewestnodes).
- LetuscallthisbreadthfirstsearchalgorithmBFS-ID3.
- BFS-ID3findsashortest decision treeand thus exhibitsthe bias"shortertrees arepreferredover longer trees.

A closer approximation to the inductive bias of ID3: Shorter trees are preferredover longer trees. Trees that place high information gain attributes close to the rootare preferred over those thatdo not.

- ID3 can be viewed as an efficient approximation to BFS-ID3, using a greedyheuristic search to attempt to find the shortest tree without conducting the entirebreadth-firstsearchthrough thehypothesis space.
- Because ID3 uses the information gain heuristic and a hill climbing strategy, itexhibitsa more complexbiasthan BFS-ID3.
- In particular, it does not always find the shortest consistent tree, and it is biased tofavourtreesthat place attributeswithhigh informationgain closest tothe root.

RestrictionBiasesandPreferenceBiases

DifferencebetweenthetypesofinductivebiasexhibitedbyID3andbytheCANDIDATE-ELIMINATIONAlgorithm.

ID3

- ID3searchesacompletehypothesisspace
- Itsearchesincompletelythroughthisspace,fromsimpletocomplexhypotheses,untilitstermination condition ismet
- Itsinductivebiasissolelyaconsequenceoftheorderingofhypothesesbyitssearchstrategy. Itshypothesisspace introducesno additional bias

CANDIDATE-ELIMINATIONAlgorithm

- TheversionspaceCANDIDATE-ELIMINATIONAlgorithmsearchesanincompletehypothesisspace
- Itsearchesthisspacecompletely,findingeveryhypothesisconsistentwiththetrainingdata.
- Its inductive bias is solely a consequence of the expressive power of its hypothesisrepresentation.Its search strategy introducesno additionalbias

RestrictionBiasesandPreferenceBiases

- The inductive bias of ID3 is a *preference* for certain hypotheses over others (e.g.,preference for shorter hypotheses over larger hypotheses), with no hard restrictionon the hypotheses that can be eventually enumerated. This form of bias is called a***preferencebias***or ***a search bias***.

- The bias of the CANDIDATE ELIMINATION algorithm is in the form of a*categorical* restriction on the set of hypotheses considered. This form of bias istypicallycalled a***restriction bias***or ***alanguage bias***.

Whichtypeofinductivebiasispreferredinordertogeneralizebeyondthetrainingdata,a preference bias or restriction bias?

- A preference bias is more desirable than a restriction bias, because it allows thelearner to work within a complete hypothesis space that is assured to contain theunknown target function.
- In contrast, a restriction bias that strictly limits the set of potential hypotheses isgenerally less desirable, because it introduces thepossibilityofexcluding theunknown target function altogether.

Occam'srazor

Occam's razor: is the problem-solving principle that the simplest solution tends to bethe right one. When presented with competing hypotheses to solve a problem, oneshouldselect the solution with the fewest assumptions.

Occam'srazor:“***Preferthesimplesthypothesisthatfitsthedata***”.

WhyPreferShortHypotheses?

Argumentinfavour:

Fewershorthypothesesthanlongones:

- Shorthypothesesfitsthetrainingdatawhichare*lesslikely*tobe*coincident*
- Longerhypothesesfitsthetrainingdatamightbe*coincident.*

Manycomplexhypothesesthatfitthecurrenttrainingdatabutfailtogeneralizecorrec tly tosubsequent data.

Argumentopposed:

- There are few small trees, and our priori chance of finding one consistent with anarbitrary set of data is therefore small. The difficulty here is that there are verymanysmallsetsofhypothesesthatonecandefine*butunderstoodbyfewerlearner*.
- The size of a hypothesis is determined by the representation used ***internally*** by thelearner.Occam'srazorwillproduce*twodifferenthypothesesfromthesametrainingexampleswhenitisappliedbytwolearners*,bothjustifyingtheircontradictory conclusions by Occam's razor. On this basis we might be tempted torejectOccam's razor altogether.

ISSUESINDECISIONTREELEARNING

1. AvoidingOverfittingtheData

   Reduced error

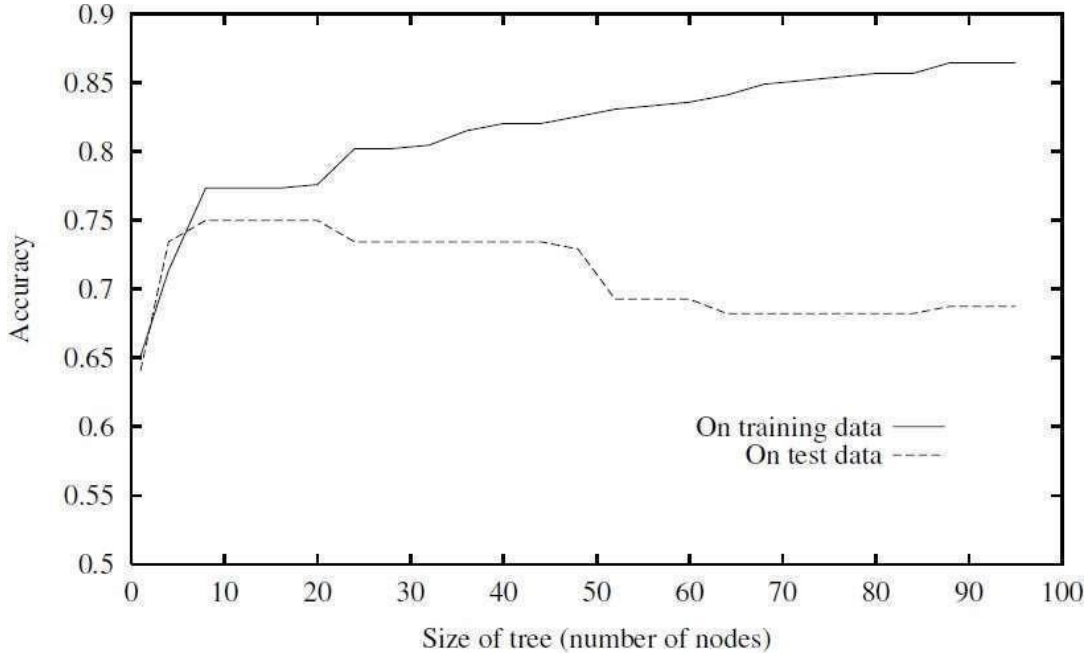
   pruningRulepost-pruning
2. IncorporatingContinuous-ValuedAttributes
3. AlternativeMeasuresforSelectingAttributes
4. HandlingTrainingExampleswithMissingAttributeValues
5. HandlingAttributeswithDifferingCosts

1. AvoidingOverfittingtheData

- The ID3 algorithm grows each branch of the tree just deeply enough to perfectlyclassify the training examples but it can lead to difficulties when there is noise inthe data, or when the number of training examples is too small to produce arepresentative sample of the true target function. This algorithm can produce treesthat***overfit***the training examples.

- ***Definition - Overfit:*** Given a hypothesis space H, a hypothesis $h \in H$ is said tooverfit the training data if there exists some alternative hypothesis $h' \in H$, suchthat*h*hassmallererrorthan*h'*overthetrainingexamples,but*h'*hasasmallererrorthan *h*over the entire distribution of instances.

- Thebelowfigureillustratestheimpactofoverfittinginatypicalapplicationofdecisiontreelearning.



- The**horizontalaxis**ofthis plotindicatesthetotalnumberofnodesinthedecisiontree,asthetreeisbeingconstructed.The**verticalaxis** indicates theaccuracy ofpredictionsmadebythe tree.
- The**solidline**showstheaccuracyofthedecisiontreeoverthetrainingexamples.The**brokenline**showsaccuracymeasuredoveranindependentsetoftestexample
- The**accuracy**ofthetreeoverthe**trainingexamplesincreases**monotonicallyasthetreeisgrown.The **accuracy**measuredovertheindependenttestexamples**firstincreases**,**thendecreases**.

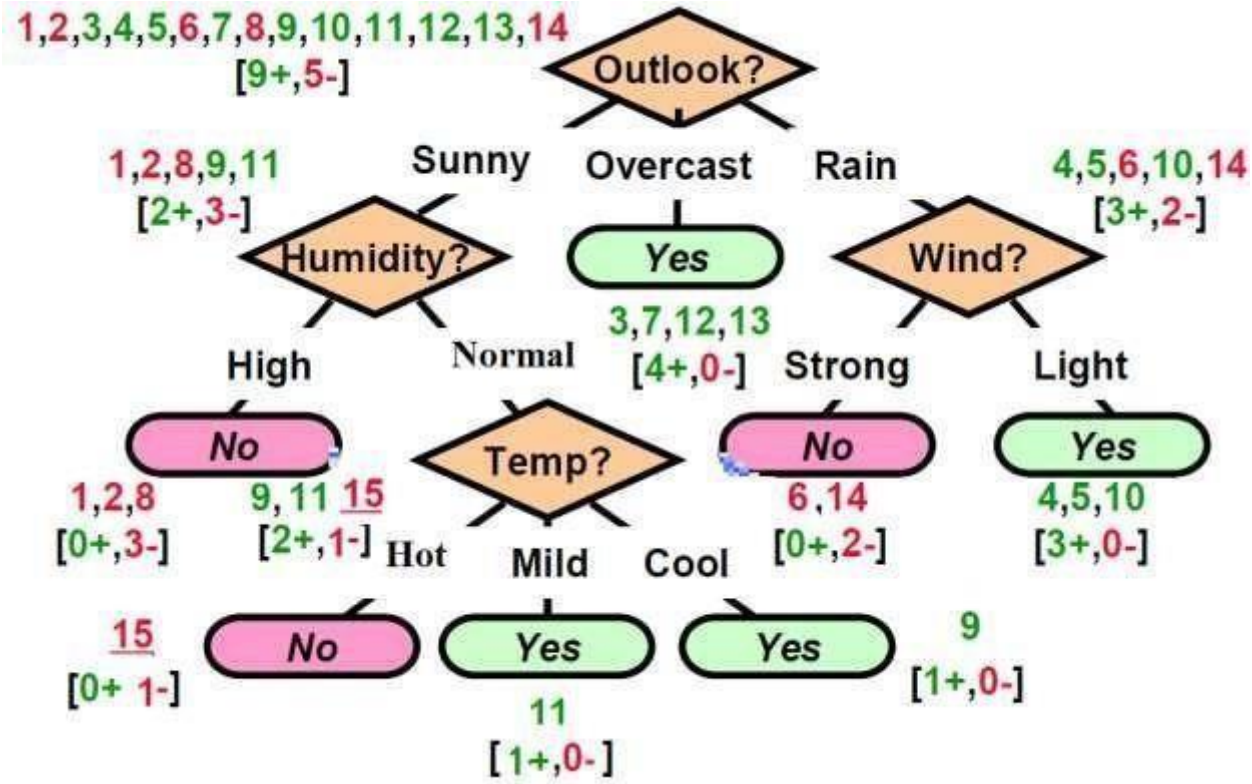
Howcanitbepossiblefortreehtofitthetrainingexamplesbetterthanh',butforittoperformmorepoorly over subsequentexamples?

1. Overfittingcanoccurwhenthetrainingexamplescontainrandomerrorsornoise
2. Whensmallnumbersofexamplesareassociatedwithleafnodes.

NoisyTrainingExample

Example15:<*Sunny*,*Hot*,*Normal*,*Strong*,->

- Exampleisnoisybecausethecorrectlabelis+
- Previouslyconstructedtreemisclassifiesit



Approaches toavoidingoverfittingindecisiontreelearning

- **Pre-pruning(avoidance):**Stopgrowingthetree earlier,beforeitreachesthe point whereit perfectly classifies thetrainingdata
- **Post-pruning(recovery):**Allowthetreetooverfitthedata,andthenpost-prunethetree

Criterionusedtodeterminethecorrectfinaltreesize

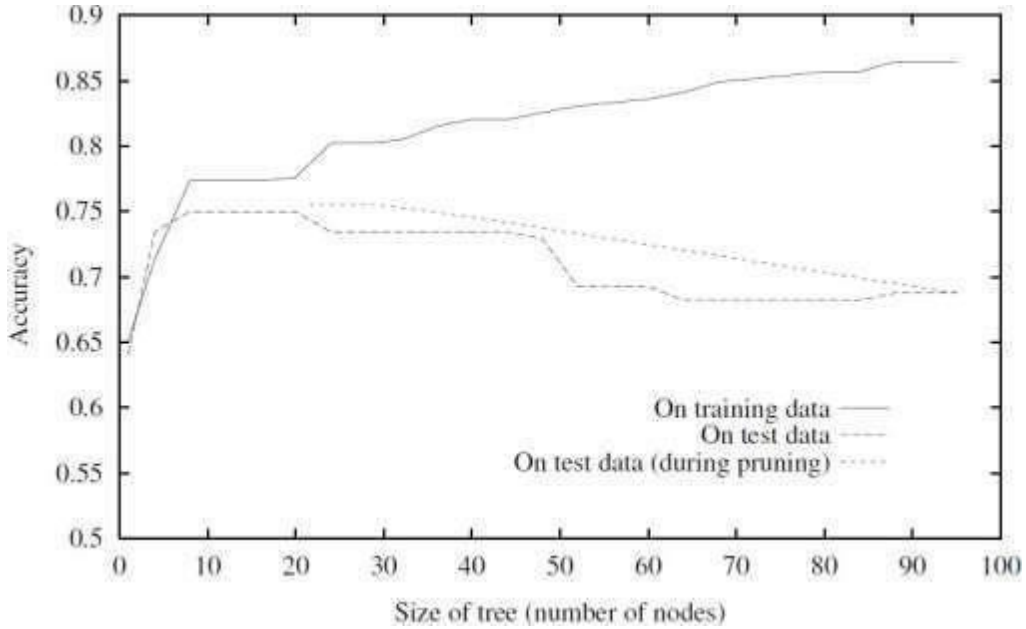
- Use a separate set of examples, distinct from the training examples, to evaluate the utility ofpost-pruningnodes from the tree
- Usealltheavailabledatafortraining,butapply***astatisticaltest***toestimatewhetherexpanding (or pruning) a particular node is likely to produce an improvement beyond thetrainingset
- Use measure of the complexity for encoding the training examples and the decision tree,halting growth of the tree when this encoding size is minimized. This approach is called theMinimumDescription Length

MDL–Minimize: size(tree)+size(misclassifications(tree))

Reduced-ErrorPruning

- ***Reduced-error pruning***, is to consider each of the decision nodes in the tree to becandidates for pruning
- ***Pruning*** a decision node consists of removing the subtree rooted at that node,making it a leaf node, and assigning it the most common classification of thetrainingexamples affiliated with that node
- Nodes are removed only if the resulting pruned tree performs no worse than-theoriginalover the validation set.
- Reduced error pruning has the effect that any leaf node added due to coincidentalregularitiesinthetrainingsetislikelytobeprunedbecausethesesamecoinci dencesare unlikelyto occur inthe validation set

Theimpactofreduced-errorpruningontheaccuracyofthedecisiontreeisillustratedinbelowfigure



- The additional line in figure shows ***accuracy over the test examples*** as the tree is pruned. Whenpruning begins, the tree is at its maximum size and lowest accuracy over the test set. As pruningproceeds,the numberofnodes isreduced andaccuracy overthetestset increases.
- The available data has been split into three subsets: the training examples, the validation examplesused for pruning the tree, and a set of test examples used to provide an unbiased estimate ofaccuracyoverfutureunseen examples.Theplotshowsaccuracyoverthetrainingandtestsets.

ProsandCons

Pro:Producessmallestversionofmostaccurate$T$(subtreeof$T$)Con:

Uses less datato construct$T$

Canaffordtoholdout$D_{validation}$?.If not(dataistoolimited),maymakeerrorworse(insufficient D_{train})

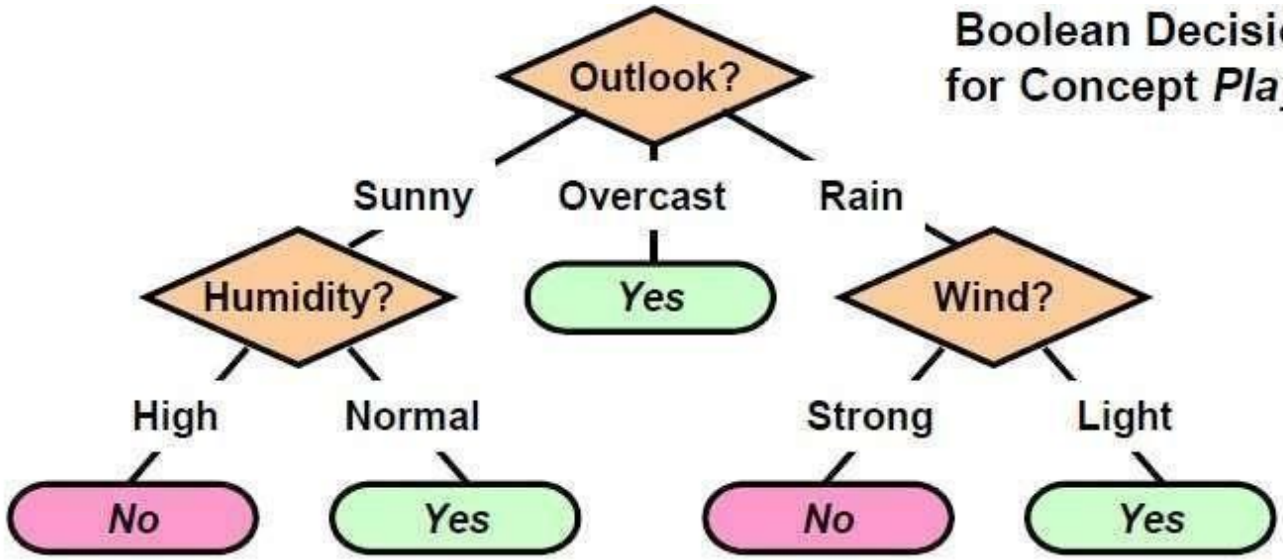
RulePost-Pruning

Rulepost-pruningissuccessfulmethodforfindinghighaccuracyhypotheses

Rulepost-pruninginvolvesthefollowingsteps:

1. Inferthedecisiontreefromthetrainingset,growingthetreeuntil thetrainingdata is fit aswell aspossibleand allowing overfittingtooccur.
2. Convertthe learnedtreeinto anequivalent setofrulesbycreating oneruleforeach path from the rootnode to aleaf node.
3. Prune(generalize)eachrulebyremovinganypreconditionsthatresultinimprovingits estimatedaccuracy.
4. Sorttheprunedrulesbytheirestimatedaccuracy, andconsiderthemin thissequencewhen classifying subsequent instances.

ConvertingaDecisionTreeintoRules



Example

- IF (*Outlook* = *Sunny*) ∧ (*Humidity* = *High*) THEN *PlayTennis* = *No*
- IF (*Outlook* = *Sunny*) ∧ (*Humidity* = *Normal*) THEN *PlayTennis* = *Yes*
- ...

Forexample,considerthedecisiontree.Theleftmostpathofthetreeinbelowfigureistranslated intothe rule.

IF(Outlook=Sunny)^(Humidity=High)TH

EN*PlayTennis*= No

Giventheaboverule,rulepost-

pruningwouldconsiderremovingthepreconditions(Outlook=

Sunny) and (Humidity = High)

- Itwouldselectwhicheverofthesepruningstepsproducedthegreatestimprovementinestimatedruleaccuracy,thenconsiderpruningthesecondpreconditionas a furtherpruning step.
- Nopruningstepisperformedifitreducestheestimatedruleaccuracy.

There are three main advantages by converting the decision tree to rules beforepruning

- Converting to rules allows distinguishing among the different contexts in which adecision node is used. Because each distinct path through the decision tree nodeproduces a distinct rule, the pruning decision regarding that attribute test can bemade differently for eachpath.
- Converting to rules removes the distinction between attribute tests that occur nearthe root of the tree and those that occur near the leaves. Thus, it avoid messybookkeeping issues such as how to reorganize the tree if the root node is prunedwhileretaining part ofthe subtree belowthis test.
- Convertingtorulesimprovesreadability. Rulesareofteneasierfortounderstand.

2.IncorporatingContinuous-ValuedAttributes

Continuous-valueddecisionattributescanbeincorporatedintothelearnedtree.

TherearetwomethodsforHandling ContinuousAttributes

1. Definenewdiscretevaluedattributesthatpartitionthecontinuousattributevalueintoa discrete set ofintervals.

E.g.,{high≡Temp>35ºC,med≡10ºC<Temp≤35ºC,low≡Temp≤10ºC}

2. Usingthresholdsforsplittingnodese.g.,A

≤aproducessubsetsA≤aandA>a

***Whatthreshold-basedbooleanattributeshouldbedefinedbasedon*Temperature?**

Temperature:	40	48	60	72	80	90
PlayTennis:	No	No	Yes	Yes	Yes	No

Pickathreshold,***c*,**thatproduces thegreatestinformationgain

- In the current example, there are two candidate thresholds, corresponding to thevalues of Temperature at which the value of *PlayTennis*changes: (48 + 60)/2, and(80 + 90)/2. The information gain can then be computed for each of the candidateattributes,$Temperature_{>54}$,and$Temperature_{>85}$andthebestcanbeselected($Temperature_{>54}$)

3. AlternativeMeasuresforSelectingAttributes

Theproblemisifattributeswithmanyvalues,***Gain***willselectit?

Example: consider the attribute ***Date***, which has a very large number of possiblevalues. (e.g.,March 4, 1979).

- Ifthisattributeisaddedtothe***PlayTennis***data,itwouldhavethehighestinformation gain of any of the attributes. This is because Date alone perfectlypredicts the target attribute over the training data. Thus, it would be selected as thedecisionattributefortherootnodeofthetreeandleadtoatreeofdepthone,whichperfect ly classifies thetraining data.
- This decision tree with root node ***Date*** is not a useful predictor because it perfectlyseparates thetraining data,butpoorlypredict onsubsequent examples.

OneApproach:Use*GainRatio*insteadof*Gain*

- Thegainratiomeasurepenalizesattributesbyincorporatingasplitinformation,that is sensitiveto how broadlyand uniformlytheattribute splitsthe data

$$GainRatio(S,A) \equiv \frac{Gain(S,A)}{SplitInformation(S,A)}$$

$$SplitInformation(S,A) \equiv -\sum_{i=1}^{c} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

- where $S_i$issubsetofS,forwhich attributeAhasvaluev_i

4. HandlingTrainingExampleswithMissingAttributeValues

Thedatawhichisavailablemaycontainmissingvaluesforsomeattributes

Example:Medicaldiagnosis

- <*Fever=true*,*Blood-Pressure=normal*,…,*Blood-Test=?*,…>
- Sometimesvaluestrulyunknown,sometimeslowpriority(orcosttoohigh)

Example:PlayTennis

Day	Outlook	Temperature	Humidity	Wind	*PlayTennis*?
1	Sunny	Hot	High	Light	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Light	Yes
4	Rain	Mild	High	Light	Yes
5	Rain	Cool	Normal	Light	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	???	Light	No
9	Sunny	Cool	Normal	Light	Yes
10	Rain	Mild	Normal	Light	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Light	Yes
14	Rain	Mild	High	Strong	No

Strategiesfordealingwiththemissingattributevalue

- If node n test A, assign most common value of A among other training examplessortedto node***n***
- Assignmostcommonvalue ofAamong other trainingexamples with sametargetvalue
- Assignaprobability$p_i$toeachofthepossiblevaluesv$_i$of***A***ratherthansimplyassigningth e most common value to***A(x)***

5. HandlingAttributeswithDifferingCosts

Insomelearningtaskstheinstanceattributesmayhaveassociatedcosts.Fore

xample:

- In learning to classify medical diseases, the patients described in terms ofattributessuch**as**Temperature,BiopsyResult,Pulse,BloodTestResults,etc.
- Theseattributesvarysignificantlyintheircosts,bothintermsofmonetarycostandco stto patient comfort
- Decisiontreesuselow-costattributeswherepossible,dependsonlyonhigh-costattributesonlywhen needed to producereliable classifications

HowtoLearnAConsistentTreewithLowExpectedCost?

OneapproachisreplaceGainby***Cost-Normalized-Gain***

Examplesofnormalizationfunctions

- Tan and Schlimmer

$$\frac{Gain^2(S, A)}{Cost(A)}.$$

- Nunez

$$\frac{2^{Gain(S,A)} - 1}{(Cost(A) + 1)^w}$$

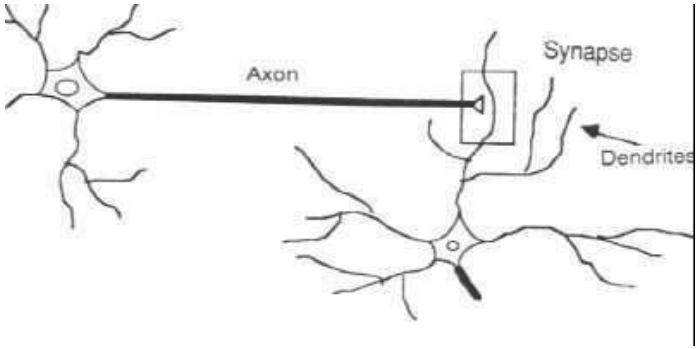
where $w \in [0, 1]$ determines importance of cost

ArtificialNeuralNetworks

Overview

1. Introduction
2. ANNrepresentations
3. PerceptronTraining
4. GradientDescentandDelta Rule
5. MultilayernetworksandBackpropagationalgorithm
6. Remarksonthebackpropagationalgorithm
7. Anillustrativeexample:facerecognition
8. Advancedtopicsinartificialneuralnetworks

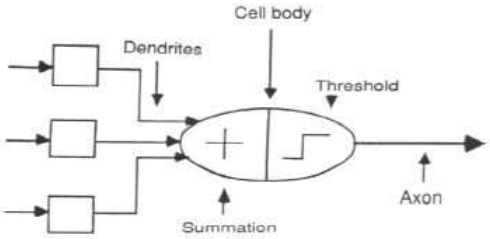
Introduction

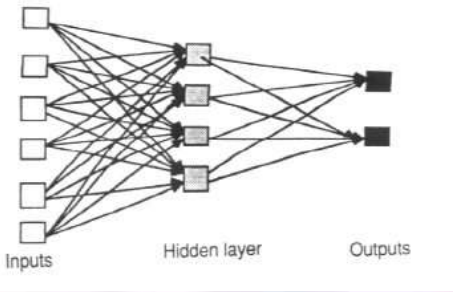


-Humanbrain

:denselyinterconnectednetworkof10^{11}neuron seachconnected to 10^{4}others

(neuronswitchingtime:approx.10^{-3}sec.)





-Propertiesofartificialneuralnets(ANN′s):

- Manyneuron-likethresholdswitchingunits
- Manyweightedinterconnectionsamongunits
- Highlyparallel,distributedprocess
-

Appropriateproblemsforneuralnetworklearning

- Inputishigh-dimensionaldiscreteorreal-valued (e.g.raw sensorinput)
- Outputisdiscreteorrealvalued
- Outputisavectorofvalues
- Possiblynoisydata
- Longtrainingtimesaccepted
- Fastevaluationofthelearnedfunctionrequired.
- Notimportantforhumanstounderstandtheweights

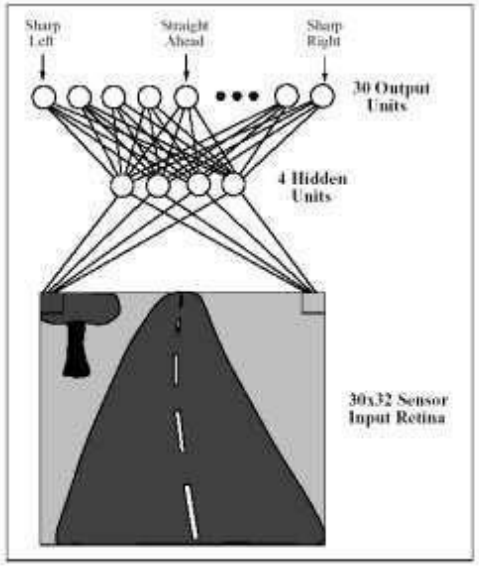
Examples:

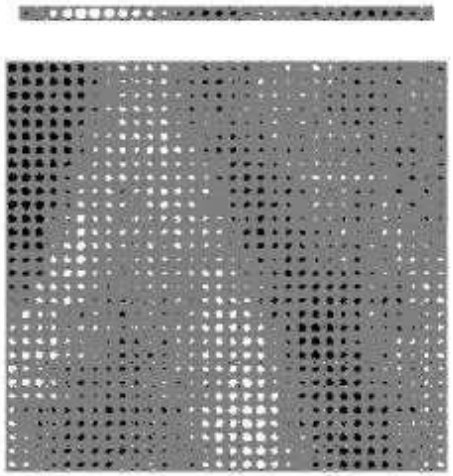
- Speechphonemerecognition
- Imageclassification
- Financialprediction

Appropriateproblemsforneuralnetworklearning

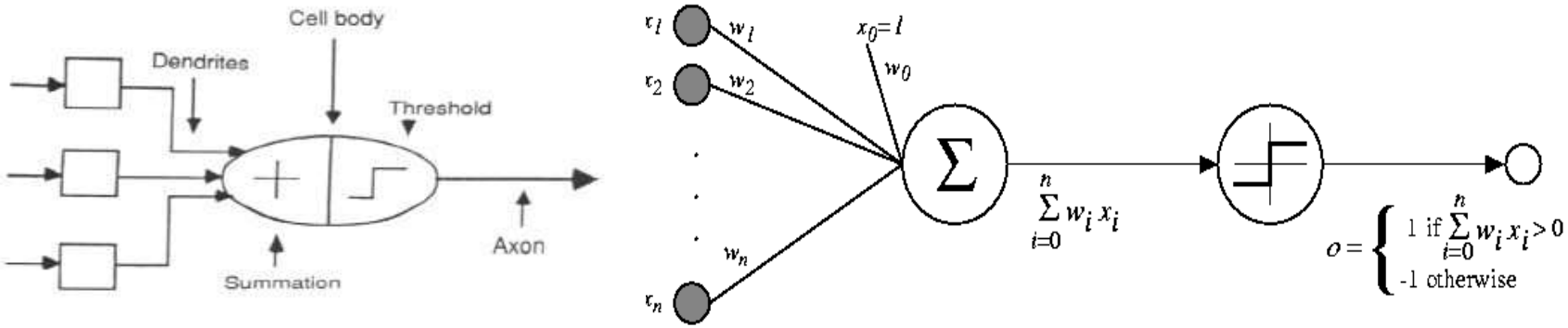
-ALVINNdrives70mphonhighways

-TheALVINNsystemusesbackpropagationalgorithmtolearntosteeran antonomousvehicledriving at speedsupto 70 miles perhour





Perceptron



- Inputvalues→Linearweightedsum→Threshold

T.Aparna, Assistant Professor, CSE

$$o(x_1, \ldots, x_n) = \begin{cases} 1 & \text{if } w_0 + w_1 x_1 + \cdots + w_n x_n > 0 \\ -1 & \text{otherwise.} \end{cases}$$

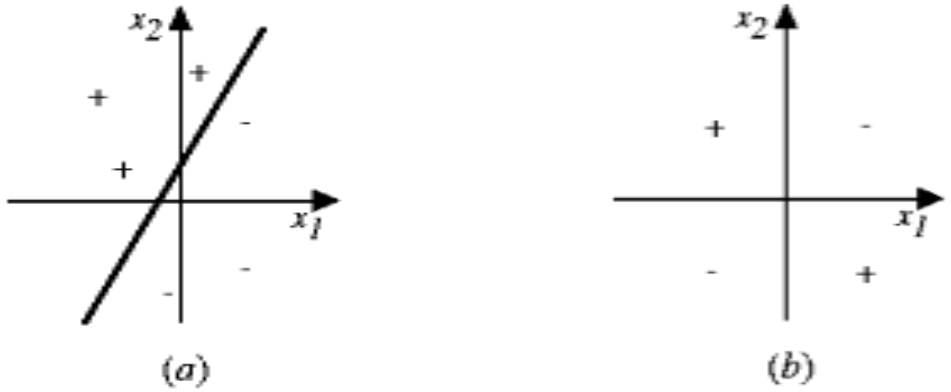
Sometimes we'll use simpler vector notation:

$$o(\vec{x}) = \begin{cases} 1 & \text{if } \vec{w} \cdot \vec{x} > 0 \\ -1 & \text{otherwise.} \end{cases}$$

T.Aparna, Assistant Professor, CSE

Decisionsurfaceofaperceptron

- Representationalpowerofperceptrons
  - Linearly separable case like (a)
    :possibletoclassifybyhyperplane,
  - Linearlyinseparablecaselike(b):im
    possibleto classify



Perceptrontrainingrule(deltarule)

$$w_i \leftarrow w_i + \Delta w_i$$

$$\text{where} \Delta w_i = \eta(t - o)x_i$$

Where:

- $t = c(x)$istargetvalue
- $o$isperceptronoutput
- ηissmallconstant(e.g.,0.1)called*learningrate*

Canproveitwillconverge

Iftrainingdataisl

- inearlyseparable

Gradientdescent

To understand, consider simpler *linear unit*, where

$$o = w_0 + w_1 x_1 + \cdots + w_n x_n$$

Let's learn w_i's that minimize the squared error

$$E[\vec{w}] \equiv \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2$$

Where D is set of training examples

Derivationofgradientdescent

- Gradientdescent

  - Error(**foralltrainingexamples**.): $E[\vec{w}] \equiv \frac{1}{2}\sum_{d \in D}(t_d - o_d)^2$

  - the*gradient*ofE(**partialdifferentiating)**:

$$\nabla E[\vec{w}] \equiv \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \cdots \frac{\partial E}{\partial w_n}\right]$$

  - direction:steepestincreaseinE.

  - Thus,trainingruleis asfollows.

$$\Delta \vec{w} = -\eta \nabla E[\vec{w}] \qquad \Delta w_i = -\eta \frac{\partial E}{\partial w_i}$$

(Thenegativesign:thedirectionthat*decreases*E)

Derivationofgradientdescent

$$\frac{\partial E}{\partial w_i} = \frac{\partial}{\partial w_i}\frac{1}{2}\sum_d (t_d - o_d)^2$$

$$= \frac{1}{2}\sum_d \frac{\partial}{\partial w_i}(t_d - o_d)^2$$

$$= \frac{1}{2}\sum_d 2(t_d - o_d)\frac{\partial}{\partial w_i}(t_d - o_d)$$

$$= \sum_d (t_d - o_d)\frac{\partial}{\partial w_i}(t_d - \vec{w}\cdot\vec{x_d})$$

$$\frac{\partial E}{\partial w_i} = \sum_d (t_d - o_d)(-x_{i,d})$$

where x_{id} denotes the single input component x_i for training example *d*

- Theweightupdaterulefor gradientdescent

$$\therefore \quad \Delta w_i = \eta \sum_{d \in D} (t_d - o_d) x_{id}$$

Gradient descent anddeltarule

GRADIENT-DESCENT($training\_examples, \eta$)

Each training example is a pair of the form $\langle \vec{x}, t \rangle$, *where* $\vec{x}$ *is the vector of input values, and* t *is the target output value.* η *is the learning rate (e.g., .05).*

- Initialize each w_i to some small random value
- Until the termination condition is met, Do
  - Initialize each Δw_i to zero.
  - For each $\langle \vec{x}, t \rangle$ in *training_examples*, Do
    - Input the instance $\vec{x}$ to the unit and compute the output o
    - For each linear unit weight w_i, Do

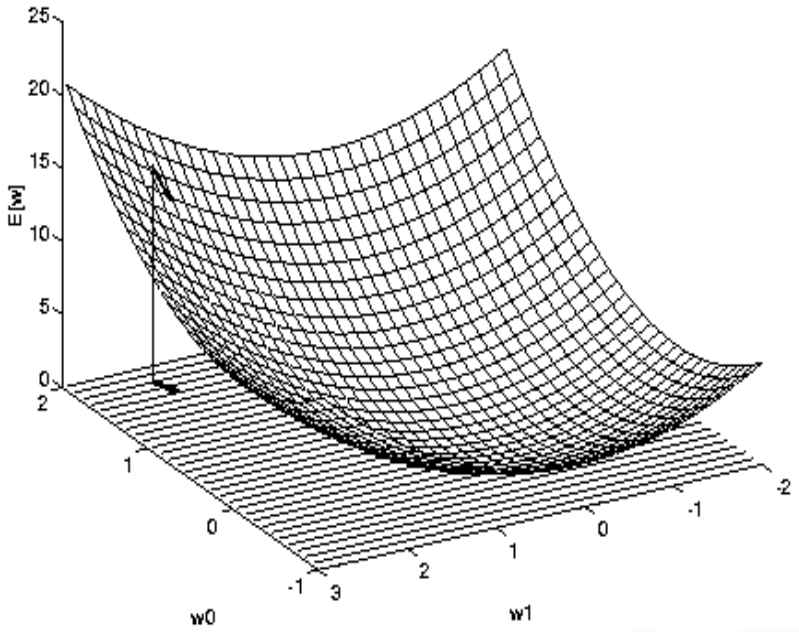
$$\Delta w_i \leftarrow \Delta w_i + \eta(t - o)x_i$$

  - For each linear unit weight w_i, Do

$$w_i \leftarrow w_i + \Delta w_i$$

Because the error surfacecontains only a single globalminimum, this algorithm willconvergetoaweightvectorwithminimum error, given asufficientlysmallη is used

HypothesisSpace



- Errorofdifferenthypotheses
- Foralinearunitwithtwoweights, thehypothesisspace$H$isthew_o,$w_1$plane.
- Thiserrorsurfacemustbeparabolicwithasingleglobal minimum(wedesireahypothesis with minimumerror).

Stochasticapproximationtogradientdescent

Batch mode Gradient Descent:
Do until satisfied

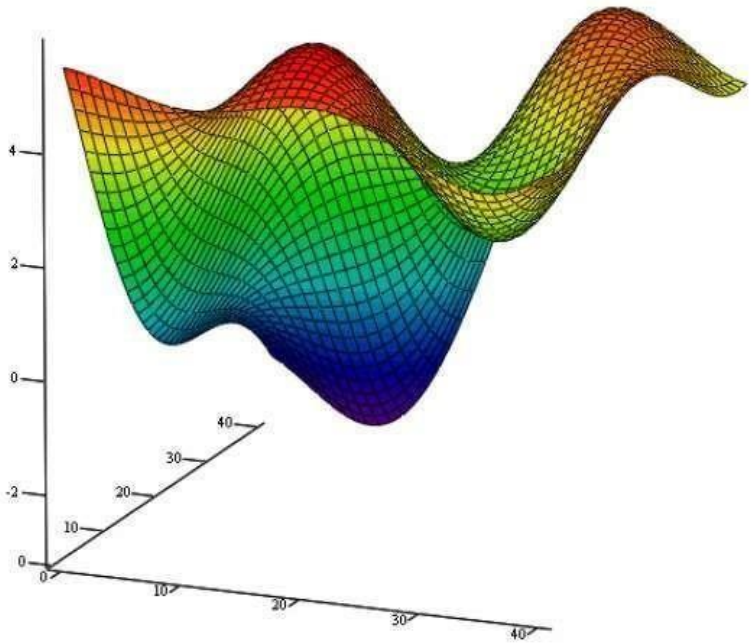
1. Compute the gradient $\nabla E_D[\vec{w}]$
2. $\vec{w} \leftarrow \vec{w} - \eta \nabla E_D[\vec{w}]$

Incremental mode Gradient Descent:
Do until satisfied

- For each training example d in D
  1. Compute the gradient $\nabla E_d[\vec{w}]$
  2. $\vec{w} \leftarrow \vec{w} - \eta \nabla E_d[\vec{w}]$

$$E_D[\vec{w}] \equiv \frac{1}{2} \sum_{d \in D} (t_d - o_d)^2$$

$$E_d[\vec{w}] \equiv \frac{1}{2}(t_d - o_d)^2$$



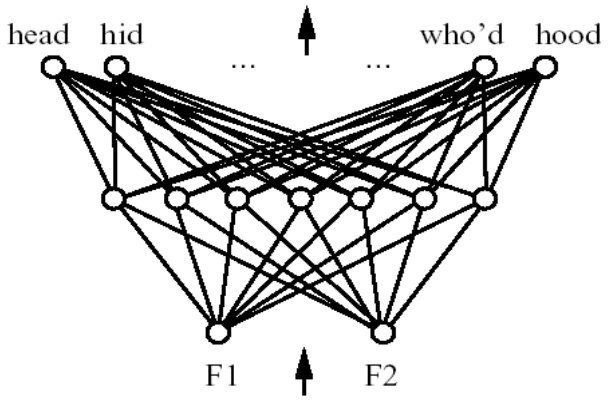
- Stochastic gradient descent (i.e. incremental mode) can sometimesavoidfallingintolocalminimabecauseitusesthevariousgradientofErather than overallgradientofE.

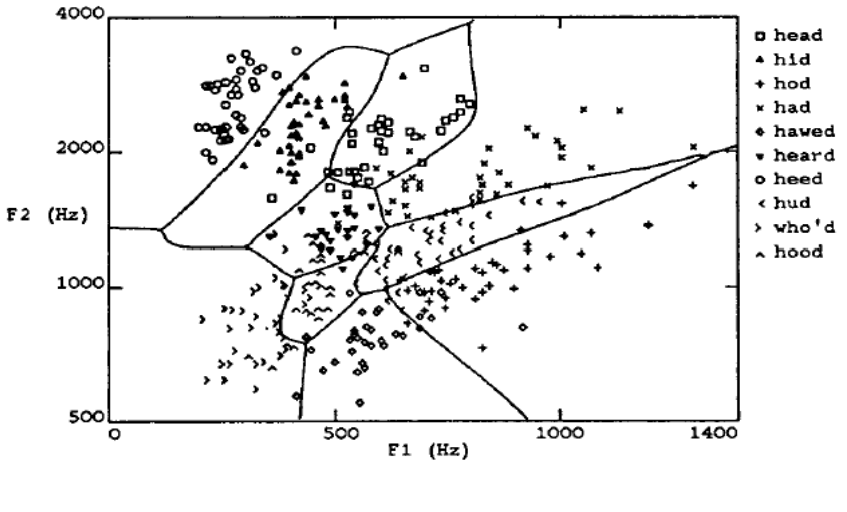
Summary

- Perceptrontrainingruleguaranteedtosucceedif
  - trainingexamplesarelinearlyseparable
  - Sufficientlysmalllearningrateη
- Linearunittrainingruleusinggradientdescent
  - Converge asymptotically to min. error hypothesis(Guaranteedtoconvergetohypothesiswithminimumsquarederror)

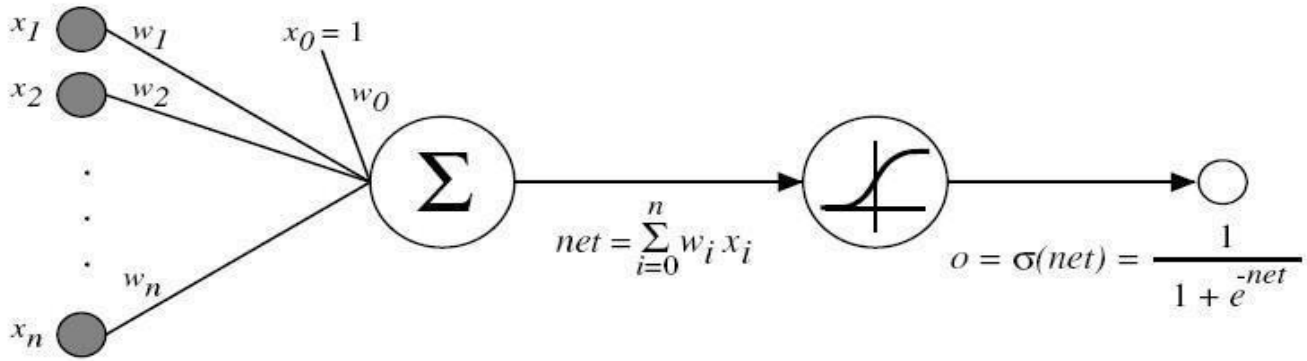
Multilayernetworksandthebackpropagationalgorithm

- Speechrecognitionexampleofmultilayernetworkslearnedbyt hebackpropagation algorithm
- **Highlynonlineardecisionsurfaces**





SigmoidThresholdUnit



$\sigma(x)$ is the sigmoid function

$$\frac{1}{1 + e^{-x}}$$

Nice property: $\frac{d\sigma(x)}{dx} = \sigma(x)(1 - \sigma(x))$

We can derive gradient decent rules to train

- One sigmoid unit
- *Multilayer networks* of sigmoid units $\rightarrow$ Backpropagation

TheBackpropagationalgorithm

Initialize all weights to small random numbers.
Until satisfied, Do

- For each training example, Do
  1. Input the training example to the network and compute the network outputs
  2. For each output unit k

$$\delta_k \leftarrow o_k(1 - o_k)(t_k - o_k)$$

  3. For each hidden unit h

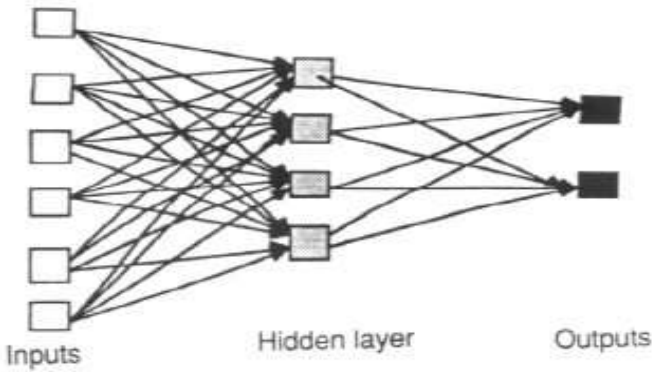
$$\delta_h \leftarrow o_h(1 - o_h) \sum_{k \in outputs} w_{h,k}\delta_k$$

  4. Update each network weight $w_{i,j}$

$$w_{i,j} \leftarrow w_{i,j} + \Delta w_{i,j}$$

where

$$\Delta w_{i,j} = \eta \delta_j x_{i,j}$$



AddingMomentum

- Oftenincludeweight*momentum*α

$$\Delta w_{i,j}(n) = \eta \delta_j x_{i,j} + \alpha \Delta w_{i,j}(n-1)$$

- n^{th}iterationupdatedependon $(n-1)^{th}$iteration
- α:constantbetween0and1(momentum)

- Rolesofmomentumterm
  - Theeffectofkeepingtheballrollingthroughsmalllocalmini mainthe error surface
  - The effect of gradually increasing the step size of thesearchinregions(greatlyimprovesthespeedoflearning)

ConvergenceandLocalMinima

- Gradientdescenttosomelocalminimum
  - Perhapsnotglobalminimum...
  - Addmomentum
  - Stochasticgradientdescent

ExpressiveCapabilitiesofANNs

Boolean functions:

- Every boolean function can be represented by network with single hidden layer
- but might require exponential (in number of inputs) hidden units

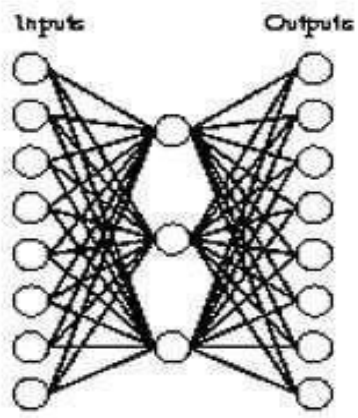
Continuous functions:

- Every bounded continuous function can be approximated with arbitrarily small error, by network with one hidden layer [Cybenko 1989; Hornik et al. 1989]
- Any function can be approximated to arbitrary accuracy by a network with two hidden layers [Cybenko 1988].

Hiddenlayerrepresentations

Hiddenlayerrepresentations

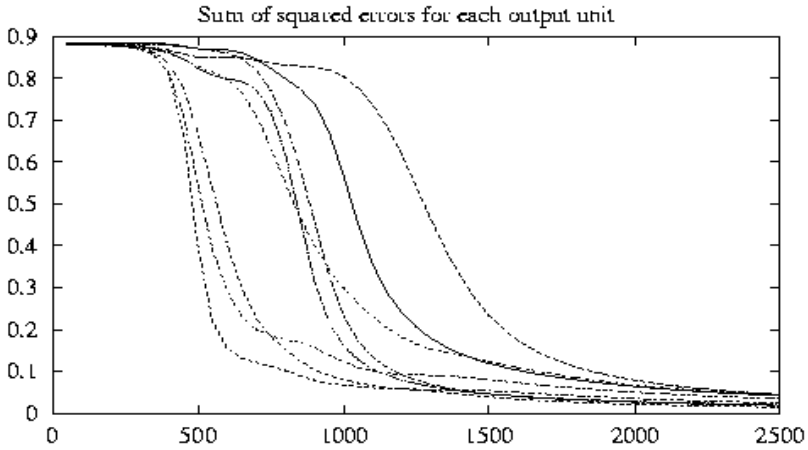
- This8x3x8networkwas trainedtolearntheidentityfunction.
- 8trainingexamplesareused.
- After5000trainingiterations,thethreehiddenunitvaluesencode theeightdistinctinputsusingtheencodingshownontheright.

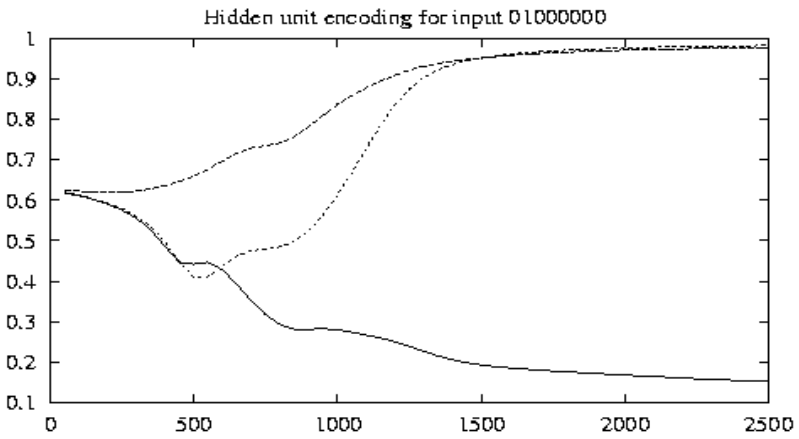


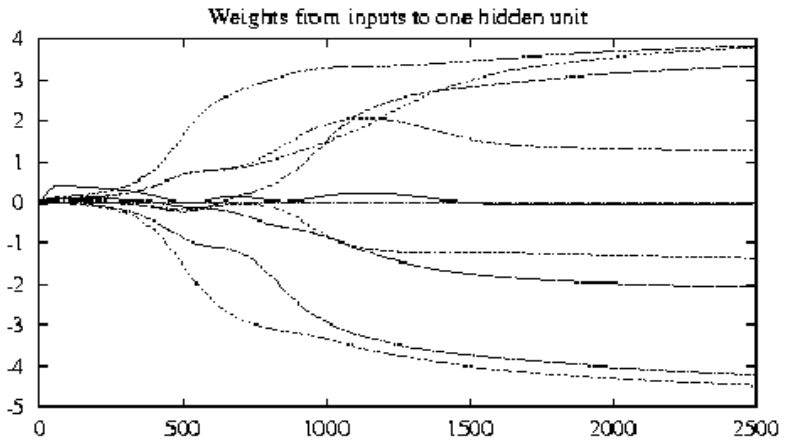
Input		Hidden Values				Output
10000000	→	.89	.04	.08	→	10000000
01000000	→	.01	.11	.88	→	01000000
00100000	→	.01	.97	.27	→	00100000
00010000	→	.99	.97	.71	→	00010000
00001000	→	.03	.05	.02	→	00001000
00000100	→	.22	.99	.99	→	00000100
00000010	→	.80	.01	.98	→	00000010
00000001	→	.60	.94	.01	→	00000001

Learningthe8x3x8network

- Mostoftheinterestingweightc hanges occurred during thefirst 2500iterations.

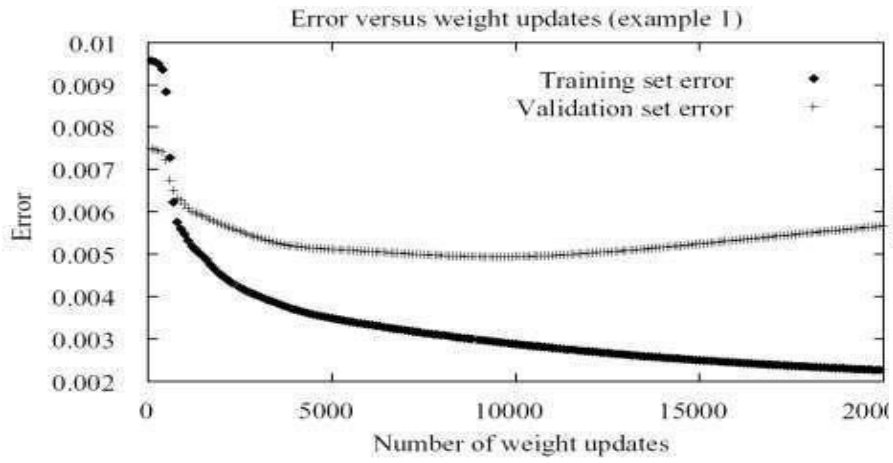


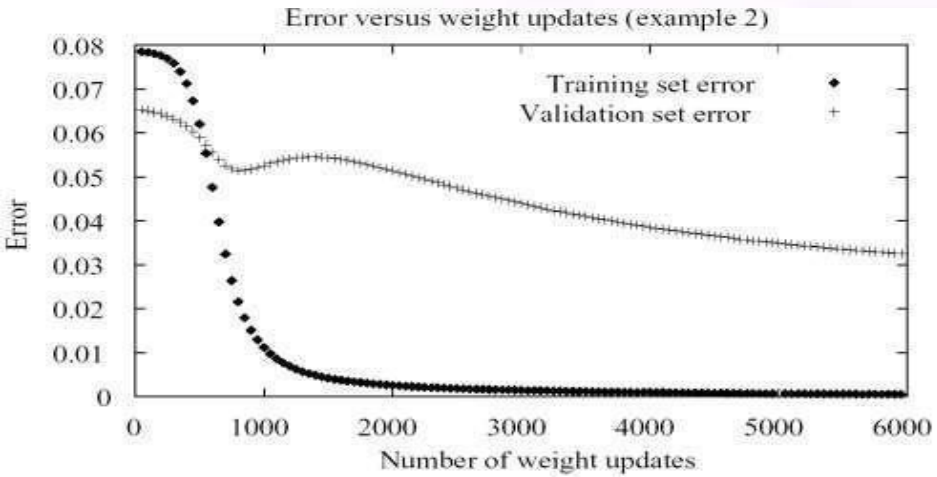




Generalization,Overfitting,andStoppingCriterion

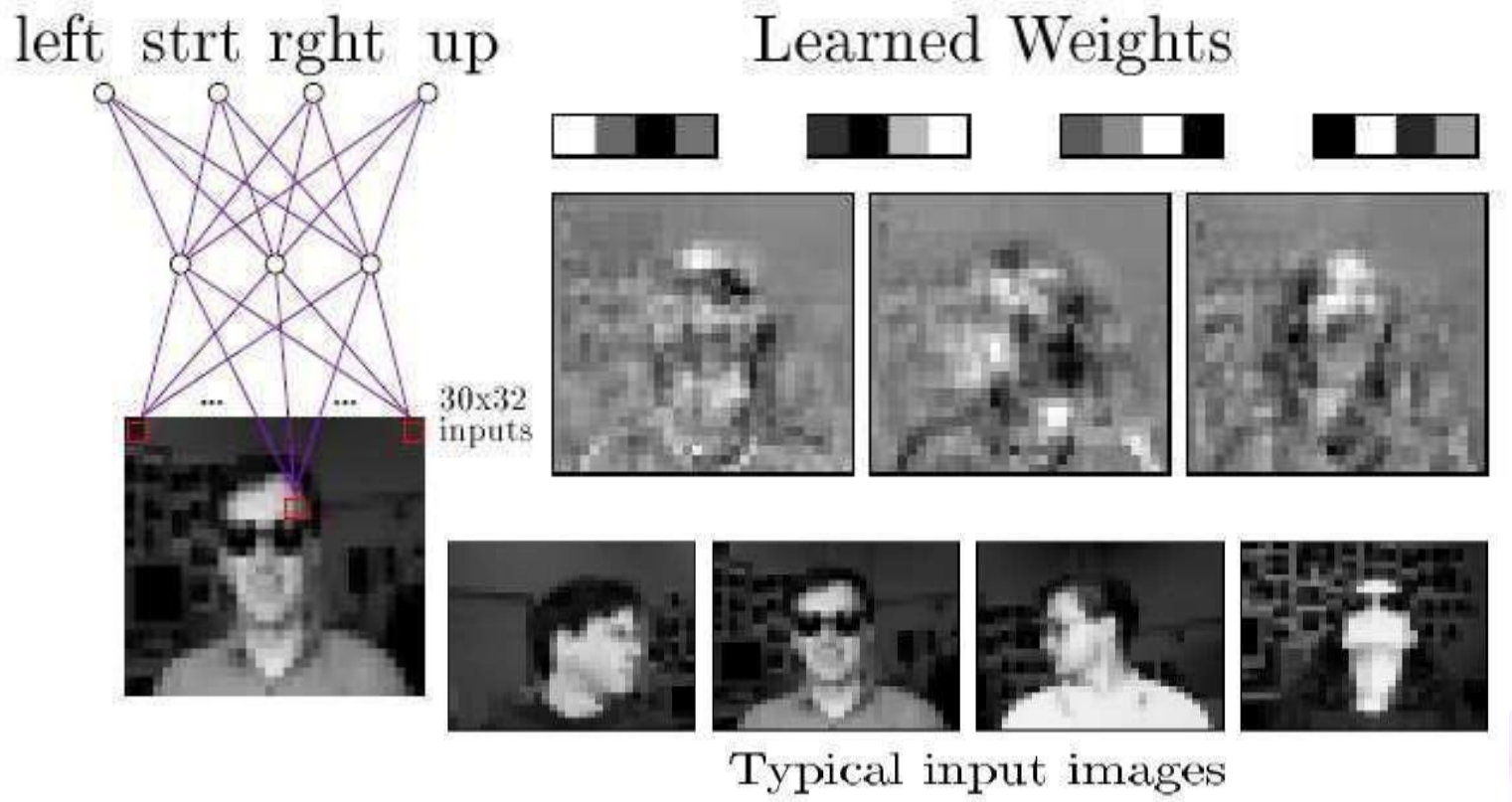
- Terminationcondition
  - –UntiltheerrorEfallsbelowsomepredeterminedthreshold
- Techniquestoaddresstheoverfittingproblem
  - Weightdecay:Decreaseeachweightbysomesmallfactordurin g each iteration.
  - Cross-validation(k-foldcross-validation)





NeuralNetsforFaceRecognition

(http://www.cs.cmu.edu/tom/faces.html)



- Trainingimages:20differentpersonswith32imagesperperson.
- After 260trainingimages,thenetworkachievesanaccuracyof90%over test set.

- Algorithmparameters:$\eta=0.3, \alpha=0.3$

AlternativeErrorFunctions

- Penalizelargeweights:(weightdecay)
  :Reducingtheriskofoverfitting

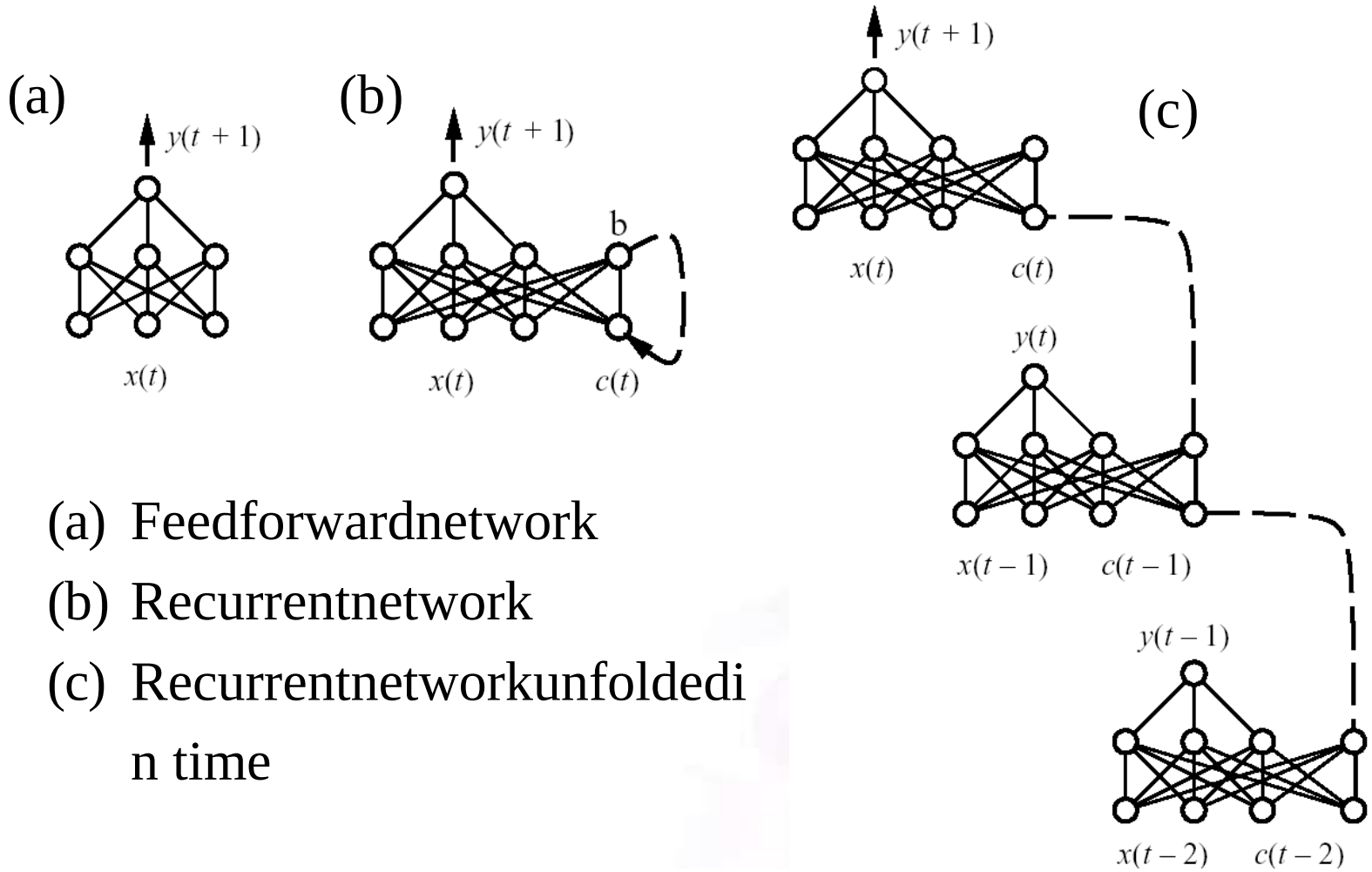
$$E(\vec{w}) \equiv \frac{1}{2} \sum_{d \in D} \sum_{k \in outputs} (t_{kd} - o_{kd})^2 + \gamma \sum_{i,j} w_{ji}^2$$

- Trainontarget slopesaswellasvalues:

$$E(\vec{w}) \equiv \frac{1}{2} \sum_{d \in D} \sum_{k \in outputs} \left[(t_{kd} - o_{kd})^2 + \mu \sum_{j \in inputs} \left(\frac{\partial t_{kd}}{\partial x_d^j} - \frac{\partial o_{kd}}{\partial x_d^j} \right)^2 \right]$$

- Minimizingthecrossentropy
  :Learningaprobabilisticoutputfunction
  (chapter6)$\sum_{d \in D} t_d \log o_d + (1 - t_d) \log(1 - o_d)$

RecurrentNetworks



(a) Feedforwardnetwork

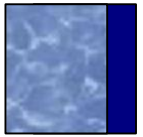
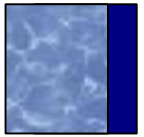
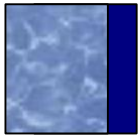
(b) Recurrentnetwork

(c) Recurrentnetworkunfoldedi n time

DynamicallyModifyingNetworkStructure

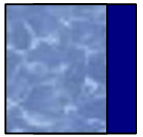
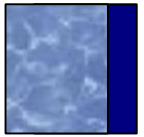
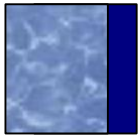
- Toimprovegeneralizationaccuracyandtrainingeff iciency
- Cascade-Correlationalgorithm(FahlmanandLebiere1990)
  - –Startwiththesimplestpossiblenetwork (nohiddenunits)andaddcomplexity
- Lecun*etal*.1990
  - – Start with the complex network and prune it as we find thatcertainconnectives areinessential.

EvaluatingHypotheses

Context

- Motivation
- EstimatingHypothesisAccuracy
- BasicsofSamplingTheory
- DifferenceinErrorofTwoHypotheses
- ComparingLearningAlgorithms
- Summary

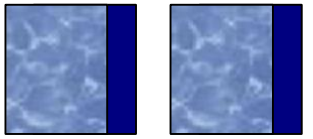




Motivation

Goal: Introductiontostatisticalmethodsforestimating hypothesisaccuracy,focusingonthefollowings:

- ✓ Given the observed accuracy of a hypothesis over a limitedsampleofdata,howwelldoesthisestimateitsaccuracyoveradditionalexamples?
- ✓ Given that one hypothesis outperforms another over somesampleofdata,howprobableisitthatthishypothesisismoreaccuratein general?
- ✓ Whendataislimitedwhatisthebestwaytousethisdatatobothlearn ahypothesisandestimate its accuracy?

Motivation

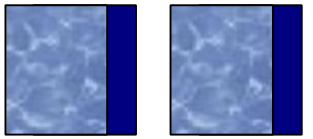


Motivation2

- It isimportanttoevaluatetheperformanceofthelearnedhypotheses as preciselyas possible:
  - ✓ Tounderstandwhethertousethehypothesis
    - ✗ Example:Learningfromlimited-sizedatabaseindicatingtheeffectivenessofdifferentmedicaltreatments
  - ✓ Evaluatinghypothesesisanintegralcomponentofmanylearningmethods
    - ✗ Example:inpost-pruningdecisiontreestoavoidoverfiting
    - ✓ Methodsforcomparingtheaccuracyoftwohypotheses

Methodsforcomparingtwolearningalgorithmswhenonlylimiteddatais available

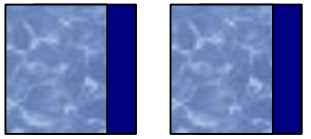
Motivation3



Motivation4

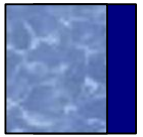
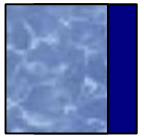
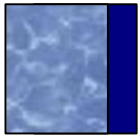
- Estimatingtheaccuracyofhypothesisisrelativelystraightforwardwhen datais plentiful.
- Givenonlyalimitedsetofdata,twokeydifficultiesarise:
  - ✓ *Biasintheestimate:*
    - ✗ Observedaccuracyofthelearnedhypothesisoverthetrainingexamplesisoftenapoorestimatorof its accuracy overfuture examples.
    - ✗ To obtain an unbiased estimate of future accuracy, we typically test thehypothesisonsomesetoftestexampleschosenindependentlyoftrainingexamples and thehypothesis.
  - ✓ *Varianceintheestimate:*
    - ✗ Themeasuredaccuracycanstillvaryfromthetrueaccuracy,dependingonthemakeupofthe particularsetoftestexamples.

Motivation5



Context

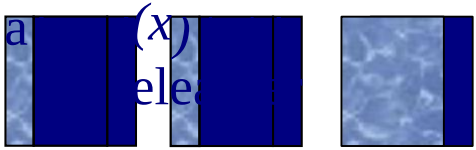
- Motivation
- ➔ EstimatingHypothesisAccuracy
  - ✓ SampleErrorandTrueError
- Basics ofSamplingTheory
- DifferenceinErrorofTwoHypotheses
- ComparingLearningAlgorithms
- Summary

EstimatingHypothesisAccuracy

- ***Setting:***
  - ✓ Somesetof possibleinstances*X*overwhichvarioustargetfunctionsma y bedefined
  - ✓ Differentinstancesin*X*maybeencounteredwithdifferentf requencies:
    - ✗ Unknown the probability distribution *D* that defines the probability ofencountering each instance in*X*
    - ✗ *D*saysnothingaboutwhetherxisapositiveoranegativeexample
  - ✓ Learningtask:Learntargetconceptortargetfunction*f*byconsideringas pace*H* of possiblehypotheses
  - ✓ Trainingexamplesareprovidedtothelearnerbyatrainer
    - ✗ whogiveseachinstanceindependently

- accordingtothedistribution*D*,
- thenforwardstheinstance*x*alongwithitscorrecttargetv



SampleErrorandTrueError

Thesampleerrorofahypothesiswithrespecttosomesamplesofinstances given from X isthe fractionof S thatit misclassifies:

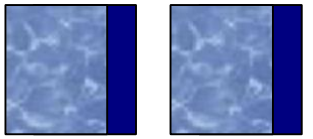
✓ ***Def:*** Thesampleerrorofahypothesis h withrespecttothetargetfunction f anddatasample **S** is

$$error_S(h) = \frac{1}{n} \sum_{x \in S} \delta(f(x), h(x))$$

Where

- n isthenumberofexamplesin S,
- thequantity $\delta(f(x), h(x))$ is *1* if $f(x) \neq h(x)$ and *0* otherwise

SampleErrorandTrueError



SampleErrorandTrueError2

Thetrueerrorofahypothesisistheprobabilitythatitwillmis classify a single randomly given instance from thedistribution*D*.

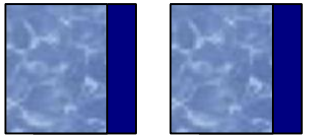
- **Def:**Thetrueerrorofhypothesishwithrespecttotargetfunction*f*and distribution *D*, is the probability that h will misclassify aninstancedrawnatrandomaccordingto*D*

$$error_D(h) \equiv Pr_{x \in D}(f(x) \neq h(x))$$

HerethenotationPr_x :: $_D$denotesthattheprobabilityistakenovertheinstancedistribution*D*.

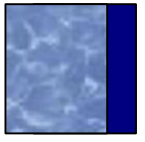
- Towishtoknowisthetrueerror$error_D(h)$.
- Mainquestion: Howgood is an estimateof $error_D(h)$ provided by$error_S(h)$?

SampleErrorandTrueError2



Context

- Motivation
- EstimatingHypothesisAccuracy
- ➔ BasicsofSamplingTheory
  - ✓ ErrorEstimationandEstimatingBinomialProportions
  - ✓ TheBinomialDistribution
  - ✓ MeanandVariance
  - ✓ ConfidenceIntervals
  - ✓ Two-SidedandOne-SidedBounds
- DifferenceinErrorofTwoHypotheses
- ComparingLearningAlgorithms
- Summary

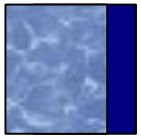


BasicsofSamplingTheory

- **Question:** How does the derivation between sample error and trueerrordependon thesizeof the data sample?
- Equal with the statistical problem: The problem of estimating theproportion of a population that exhibits some property, given theobservedproportionoversomerandom sampleofthepopulation.
- **Here:** Thepropertyofinterestisthat *hmisclassifiestheexample*
- **Answer:**
  - ✓ When measuring the sample error we are performing an experimentwitharandomoutcome.
  - ✓ Repeating this experiment many times, each time drawing a differentrandomsampleset S_i of size *n*, wewouldexpecttoobserve differentvaluesforthevarious $error_{S_i}(h)$ dependingonrandom differencesinthemakeupofthevarious S_i
  - ✓ Insuchcases $error_{S_i}(h)$ theoutcomeofthe i^{th} suchexperimentisa

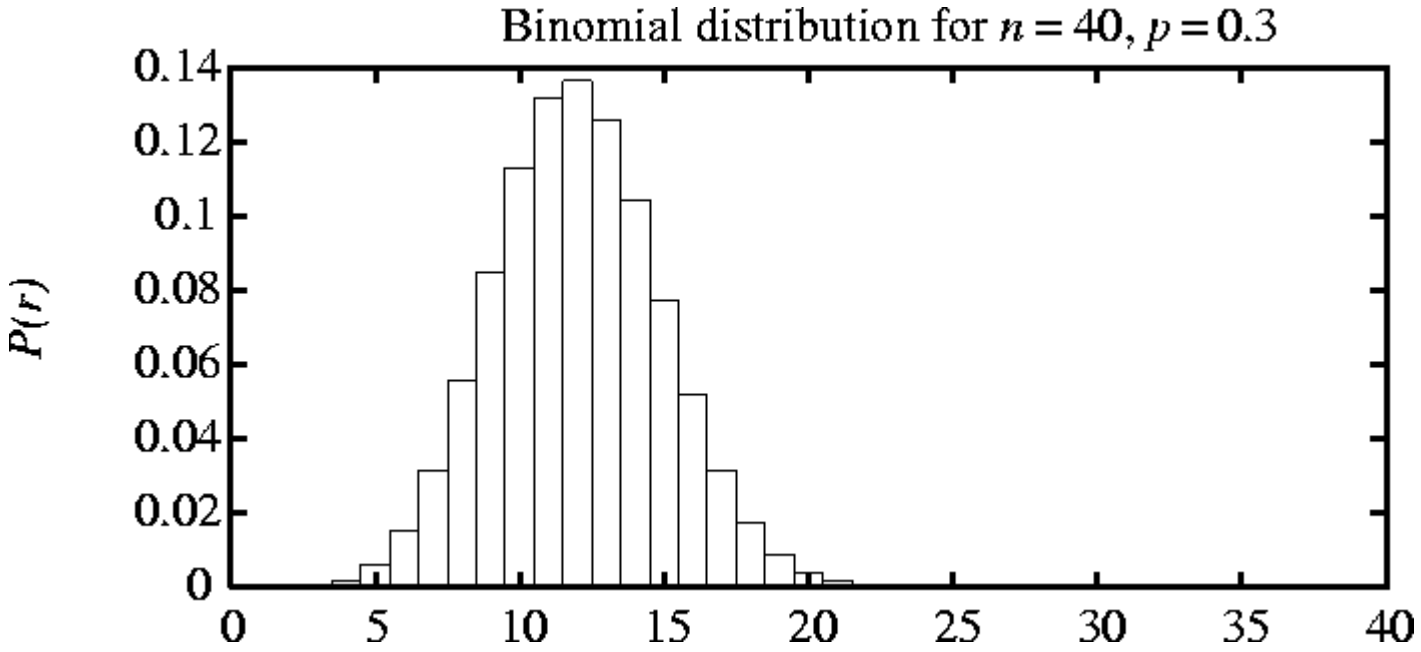
BasicsofSamplingTheory

randomvariable



ErrorEstimationandEstimatingBinomial Proportions2

- **Imagine:**
  - ✓ Runkrandomexperiments,
  - ✓ Measuringtherandomvariables$error_{S_1}(h), error_{S_2}(h) \ldots error_{S_k}(h)$
  - ✓ Plotahistogramdisplayingthefrequencywithwhichweobservedeach possibleerrorvalue
- **Result:**histogram



TheBinomialDistribution

- GeneralsettingtowhichtheBinomialdistributionapplies:
  - ✓ Thereisabaseorunderlying experimentwhoseoutcomecanbedescribedbyarandomvariable, say*Y*.It can take ontwopossible values.
  - ✓ Theprobabilitythat*Y=1*onanysingletrialoftheunderlying experimentisgiven by some constant p, independent of the outcome of any otherexperiment. Theprobabilitythat*Y=0*istherefore*1-p*. Typically,*p*isnotknowninadvance,andtheproblemistoestimateit.
  - ✓ Aseriesof*n*independenttrialsoftheunderlyingexperimentisperformed, producingthesequenceofindependent,identicallydistributedrandomvari ables $Y_1, Y_2 \ldots Y_k$ LetR denotethe numberof trialsfor which $Y_i \quad -1$ in thisseriesofnexperiments R $i \quad 1$
  - ✓ Theprobabilitythat*R*willtakeonaspecificvalue*r*isgivenbytheBinomial distribution: $Pr(R = r) = \frac{n!}{r!(n-r)!} p^r (1-p)^{n-r}$

MeanandVariance

- **Def:**Consider $Y = \{y_1, y_2, \ldots y_n\}$ The expectedvalueof$Y$,$E[Y]$,is

$$E[Y] = \sum_{i=1}$$

- **Example:**IfYtakes onthevalue1withprobability0.7andthevalue 2 with probability0.3 then itsexpected value is

$$1 \cdot 0.7 + 2 \cdot 0.3 = 1.3$$

- IncaseofarandomvariableYgovernedbyaBinomialdist ributionthe expected value is:

$$E[Y] = np$$

MeanandVariance2

- Variance captures the „*width*“ or „*spread*“ of the probabilitydistribution; that is it captures how far the random variable isexpectedtovary from its mean value
- ***Def:*** The variance of *Y*, *Var[Y]*, is

$$Var[Y] = E[(Y - E[Y])^2]$$

- Thesquarerootofthevarianceiscalledthestandard deviationofY,denoted by σ_Y
- ***Def:*** Thestandarddeviationofarandomvariable Y, σ_Y is

$$\sigma_Y = \sqrt{E[(Y - EY)^2]}$$

- In caseofarandomvariableYgovernedbyBinomialdistribution thevarianceandthe standard deviationaredefinedasfollows:

$$\sigma_Y = \sqrt{np(1-p)}$$

ConfidenceIntervals

- **Describe:**
  - ✓ Giveanintervalwithinwhichthetruevalueisexpectedtofall,alongwiththeprobabilitywithwhichitisexpectedtofallintothisinterval
- **Def:** An*N%*confidenceintervalforsomeparameters*p*isanintervalthatis expected with probability*N%*to contain*p*.
- Howconfidenceintervalsfor $error_D(h)$ canbederived:
  - ✓ Binomialprobabilitydistributiongoverningtheestimator $error_S(h)$
  - ✓ Themeanvalueofdistributionis $error_D$
  - ✓ Standarddeviationis $\sigma_{error_S(h)} \approx \sqrt{\frac{error_S(h)(1-error_S(h))}{n}}$
- **Goal:** Derivea*95%*confidenceinterval=> find the interval centered around the mean value $error_D(h)$,whichiswideenoughtocontain*95%*oftotalprobabilityunderthisdist

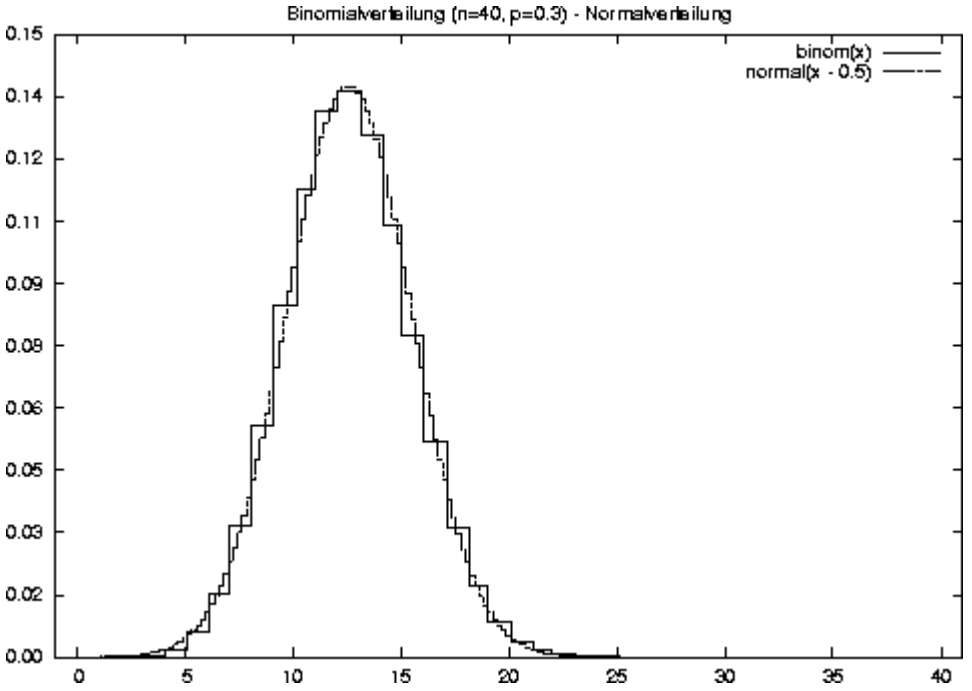
ribution

ConfidenceIntervals

ConfidenceIntervals2

- **Question:**HowcanthesizeofintervalthatcontainsN%ofthepro babilitymass befoundforgiven*N*?
- **Problem:** Unfortunately for theBinomialdistributionthisca lculation can be quitetedious.
- **But:**BinomialdistributioncanbecloselyapproximatedbyN ormal distribution

ConfidenceIntervals3



ConfidenceIntervals4

- Normalorgaussian distributionisabell-shapeddistributiondefinedbytheprobabilitydensityfunction

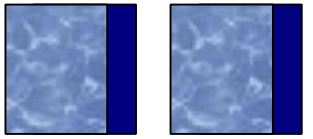
$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

  - IftherandomvariableXfollowsanormaldistributionthen:
    - ✓ TheprobabilitythatX$_b$willfallintotheinterval(a,b)isgivenby

$$\int_a p(X)\,dx$$

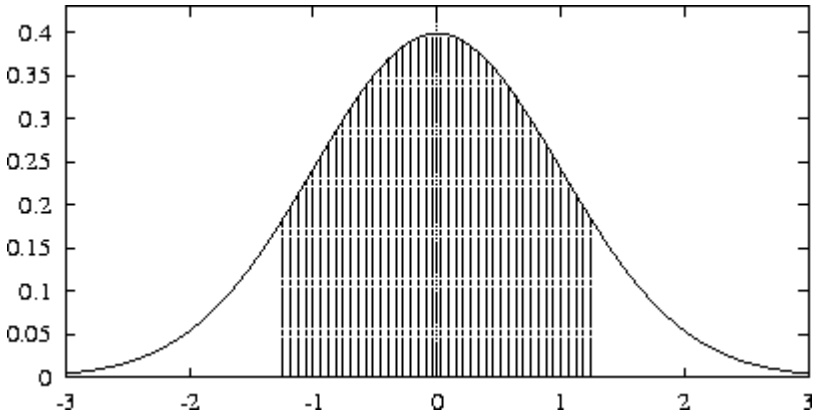
- ✓ Theexpected,ormeanvalueofX,E[X],is $E[X] = \mu$
- ✓ ThevarianceofX,Var(X)is $Var(X) = \sigma^2$
- ✓ ThestandarddeviationofX, σ_X σ

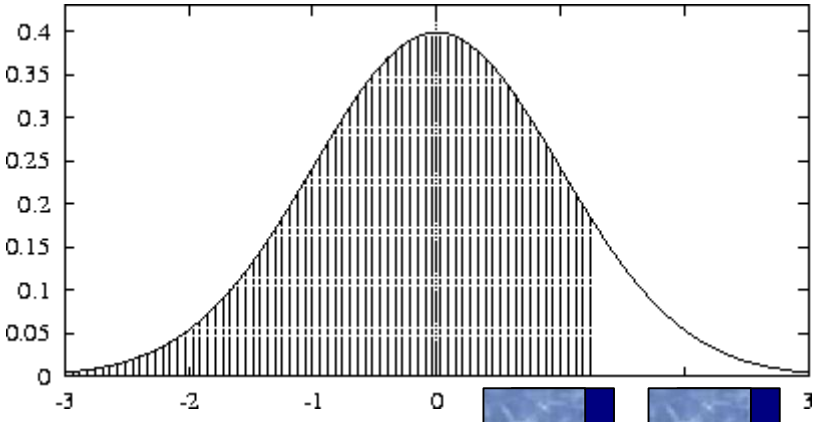
ConfidenceIntervals5



Two-SidedandOne-SidedBounds

- **Two-sidedbound:**Itboundstheestimatedquantityfromaboveand below
- **One-sidedbound:**Ifweareinterestedinquestionslike:Whatistheprobabilitythat$error_D(h)$is atmost$U$





Two-SidedandOne-SidedBounds2

- If thesampleerror isconsideredasnormaldistributedindicatingthat:
  - ✓ the $error_D(h)$ coucheswithN%probabilityintheinterval

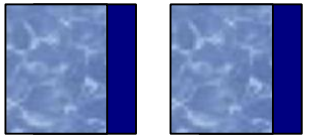
$$error_S(h) \pm z_N \sqrt{\frac{error_S(h)(1 - error_S(h))}{n}}$$

where z_N isaconstant

ConfidencelevelN%	50.00%	68.00%	80.00%	90.00%	95.00%	98.00%	99.00%
Constant	0.67	1	1.28	1.64	1.96	2.33	2.58

Table1:Valuesof z_N fortwosided$N\%$confidenceintervals

Two-SidedandOne-SidedBounds3



Two-SidedandOne-SidedBounds4

- Example:
  - ✓ n=50
  - ✓ Hypothesishmakesr=16errors=> $error_S(h) = \frac{16}{50}$
  - ✓ UsingthevaluesfromTable1
    - ✗ With99%probabilityis $error_D(h)$ intheinterval

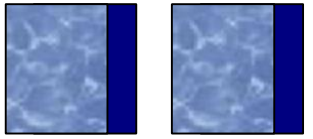
$$0.32 \pm 2.58\sqrt{\frac{}{50}}$$

    - ✗ $error_D(h)$ isintheintervalwith50%

Ifthenumberserrorsis12then probability

$$0.24 \pm 0.67\sqrt{\frac{0.24 \cdot 0.76}{50}} \approx 0.24 \pm 0.04$$

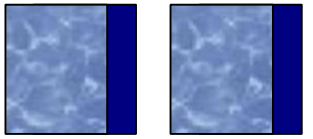
Two-SidedandOne-SidedBounds5



Two-SidedandOne-SidedBounds6

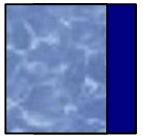
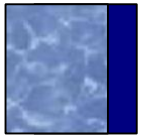
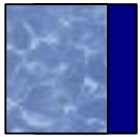
- One-sidederrorbound
  It canbecomputed withhalfoftheprobabilityoftheerrorfromnormaldistributed two-sided error bound
- Example:
  - ✓ hdelivers12errors,n=40
  - ✓ Itleadstoa(twosided)95%confidenceintervalof 0.30 ± 0.14 =>
  - ✓ Inthiscase $100(1-\alpha)\ 95$ so $\alpha\ 0.05$
  - ✓ Thus,wecanapplythe rule $100(1-\alpha/2)\ 97.5$ with confidencethat $error_D(h)$ isatmost $0.30 + 0.1$
  - ✓ Makingnoassumptionaboutthelowerboundon $error_D$
  - ✓ Thuswehaveaone-sidederrorboundonerror withdoubletheconfidencethatwehad inthecorrespondingtwo-sidedbound $error_D(h)$

Two-SidedandOne-SidedBounds7



Context

- Motivation
- EstimatingHypothesisAccuracy
- BasicsofSamplingTheory
- ➔ DifferenceinErrorofTwoHypotheses
  - ✓ HypothesisTesting
- ComparingLearningAlgorithms
- Summary

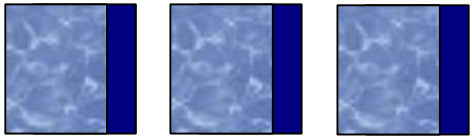
DifferenceinErrorsofTwoHypotheses

- **Consider:**
  - ✓ twohypothesesh_1 and$h_2$forsomediscrete-valuedtargetfunction
  - ✓ $h_1$hasbeentestedonasample S_1containing $n_1$randomly drawnexamples
  - ✓ $h_2$hasbeentestedonasample S_2 containing $n_2$randomlydrawn examples
- Supposewewishtoestimatethedifference *d*betweenthetrueerrorsof these two hypotheses

$$d \equiv error_D(h_1) - error_D(h_2)$$

- 4-stepproceduretoderive confidenceintervalestimatesfor*d*
  - ✓ Choosetheestimator $\hat{d} \equiv error_{S_1}(h_1) - error_{S_2}(h_2)$
  - ✓ Wedonotprovebutitcan beshown that $\hat{d}$ d gives an unbiased estimate of d; that is E

$d \quad d$



HypothesisTesting

Question: What is the probability distribution governing the random variable $\hat{d}$?

Answer:

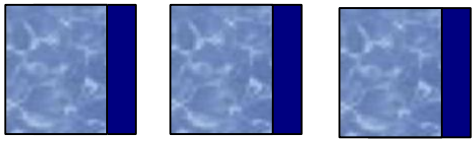
- ✓ n_1, n_2 both errors $error_S(h_1)$ and $error_S(h_2)$ follow a distribution that is approximately normal
- ✓ Difference of two normal distributions is also normal => $\hat{d}$ is also approximately normal
- ✓ The variance of this distribution is the sum of the variances of $error_S(h_1)$ and $error_S(h_2)$
- ✓ We have

$$\sigma_{\hat{d}}^2 \approx \frac{error_S(h_1)(1 - error_S(h_1))}{n_1} + \frac{error_S(h_2)(1 - error_S(h_2))}{n_2}$$

- ✓ For a random variable $\hat{d}$ obeying a normal distribution with mean d and variance

HypothesisTesting

the $N\%$ confidence interval estimate for d is $d \pm z_N \sigma$



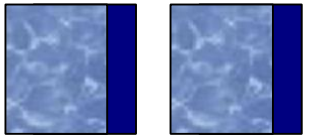
HypothesisTesting 2

- So $\hat{d} \pm z_N \sqrt{\frac{error_{S_1}(h_1)(1 - error_{S_1}(h_1))}{n_1} + \frac{error_{S_2}(h_2)(1 - error_{S_2}(h_2))}{n_2}}$

  z_N isthesameconstantasdescribedinTable1

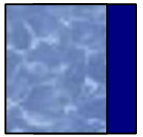
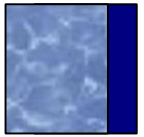
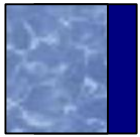
- Testoversamedata
  - h_1 And$h_2$aretestedonasinglesampleS(whereSisstillindependentof $\hat{d}$ h_1 andh_2)
  - Redefine d: $d \equiv error_{S_1}(h_1) - error_{S_2}(h_2)$
  - The variance in this new$d$ varianceoftheoriginal $\hat{d}$ willusuallybesmallerthanthe
  - UsingasinglesampleSeliminates s_1 and s_2 thevariancedue torandom differencesinthecompositionsof

HypothesisTesting 2



Context

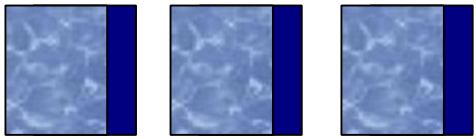
- Motivation
- EstimatingHypothesisAccuracy
- BasicsofSamplingTheory
- DifferenceinErrorofTwoHypotheses
- ➔ ComparingLearningAlgorithms
- Summary

ComparingLearningAlgorithms

- **Goal:** Comparingtheperformanceoftwolearningalgorithm L_A andL_B
- **Question:**
  - ✓ Whatisanappropriatetestforcomparinglearningalgorithms?
  - ✓ Howcanwedeterminewhetheranobserveddifferencebetweenthealgorithmsisstatisticallysignificant?
- Activedebatewithinthemachine-learningresearchcommunityregardingthe best method for comparison
- Task:Determinewhichof L_A and L_B isthebetterlearningmethodonaverageforlearningsomeparticulartargetfunctionf
  - ✓ „*On average*“ is to consider the relative performance of these twoalgorithmsaveragedoverallthetrainingsetofsize*n* thatmightbedrawnfromtheunderlyinginstancedistribution*D*

$$E_{S_D}\left[error_D\left[L_A(S)\right]\,error_D\left[L_B(S)\right]\right]$$



ComparingLearningAlgorithms2

- **Inpractice:**
  - ✓ Wehaveonlyalimitedsample D_0
  - ✓ DivideD_0intoatrainingset S_0andadisjointtest set T_0
  - ✓ Thetrainingdatacanbeusedtotrainboth L_A and L_B
  - ✓ Testsetcanbeusedtocomparetheaccuracyofthetwolearned hypothesis $[error_{T_0}(L_A(S_0)) \quad error_{T_0}(L_B(S_0))]$
- **Improvement:**
  - ✓ PartitiontheavailabledataD_0into$k$disjointsubsets$T_1, T_2, \ldots, T_k$ofequal size,wherethissizeisatleast30
  - ✓ **For**i$T$from*1*to*k*,do use $T_i$forthetestandtheremainingdatafortrainingset S_i

$S_i \Leftarrow D_0 T_i$

$h_A \Leftarrow L_A(S_i)$

$h_B \Leftarrow L_B(S_i)$

$\delta_i \Leftarrow error_{T_i}(h_A) \quad error_{T_i}(h_B)$

$\bar{\delta} \qquad \bar{\delta} \equiv \frac{1}{k} \sum^{k} \delta$

ComparingLearningAlgorithms3

k i

ComparingLearningAlgorithms4

- Theapproximate*N%*confidenceintervalforestimatingthe quantityin $[error_{T_0}(L_A(S_0)) - error_{T_0}(L_B(S_0))]$ using δ isgivenby

$$\bar{\delta} \mp t_{N,k-1} s_{\bar{\delta}}$$

where $t_{N,k-1}$ isaconstantthat~~playsaroleana~~logoustothatof z_N

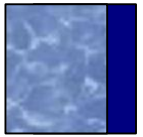
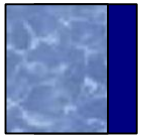
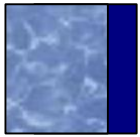
- $s_{\bar{\delta}}$ definedasfollowing $s_{\bar{\delta}} = \sqrt{\frac{1}{k(k-1)} \sum_{i=1}^{k} (\delta_i - \bar{\delta})^2}$

			Confidencelev	
			90%	95%
ν	=	2	2,92	4,3
ν	=	5	2,02	2,57
ν	=	10	1,81	2,23
ν	=	20	1,72	2,09
ν	=	30	1,7	2,04
ν	=	##	1,66	1,98
ν	=	∞	1,64	1,96

ComparingLearningAlgorithms5

Context

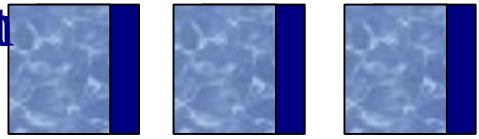
- Motivation
- EstimatingHypothesisAccuracy
- BasicsofSamplingTheory
- DifferenceinErrorofTwoHypotheses
- ComparingLearningAlgorithms
- ➔ Summary

Summary

- Statisticaltheoryprovidesabasisforestimatingthetrueerror ($error_D(h)$)ofhypothesish,basedonitsobservederror($error_S(h)$) overasampleSofdata.
- In general, the problem of estimating confidence intervals isapproachedbyidentifyingtheparametertobeestimated($error_D(h)$)and anestimator($error_S(h)$)forthisquantity.
- Becausetheestimatorisarandomvariableitcanbecharacterisedbyt heprobability distributionthat governs its value.
- Confidenceintervalscanthenbecalculatedbydeterminingthein terval that containsthe desired probability mass under thisdistribution.
- Acause ofestimationerroristhevarianceintheestimate.Evenwithan unbiased estimator, the observed value ofthe estimator is likelyto vary fromone σ^2 experimentto another.

The variance of the distribution governing theestimat orcharacteriseshow widely this estimateislikely to



Summary2

- Comparingtherelativeeffectivenessoftwolearningalgorithmsisan estimation problem that is relatively easy when data and timeare unlimited, but more difficultwhen these resourcesarelimited.
- One approach to runthe learning algorithms on different subsetsof available data, testing the learned hypotheses on the remainingdata,then averagingthe resultof these experiments.
- In most cases considered here, deriving confidence intervalsinvolvesmakinganumberofassumptionsandapproximat ions.

UNIT -

4BAYEIANLEARNING

CONTENT

- Introduction
- Bayestheorem
- Bayestheoremandconceptlearning
- MaximumlikelihoodandLeastSquaredErrorHypothesis
- MaximumlikelihoodHypothesesforpredictingprobabilities
- MinimumDescriptionLengthPrinciple
- NaiveBayesclassifier
- Bayesianbeliefnetworks
- EMalgorithm

INTRODUCTION

Bayesianlearningmethodsarerelevanttostudyofmachinelearningfortwodifferentreasons.

- First,Bayesianlearningalgorithmsthatcalculateexplicitprobabilitiesforhypotheses,suchasthenaiveBayesclassifier,areamongthemostpracticalapproachestocertain types of learning problems
- The second reason is that they provide a useful perspectivefor understandingmanylearning algorithms thatdo not explicitlymanipulateprobabilities.

FeaturesofBayesianLearningMethods

- Each observed training example can incrementally decrease or increase the estimatedprobability that a hypothesis is correct. This provides a more flexible approach tolearning than algorithms that completely eliminate a hypothesis if it is found to beinconsistentwith any singleexample
- Priorknowledgecanbecombinedwithobserveddatatodeterminethefinalprobability of a hypothesis. In Bayesian learning, prior knowledge is provided byasserting (1) a prior probability for each candidate hypothesis, and (2) a probabilitydistribution over observed data for each possiblehypothesis.
- Bayesianmethodscanaccommodatehypothesesthatmakeprobabilisticpredictions
- New instances can be classified by combining the predictions of multiple hypotheses,weightedbytheir probabilities.
- Even in cases where Bayesian methods prove computationally intractable, they canprovide a standard of optimal decision making against which other practical methodscan bemeasured.

PracticaldifficultyinapplyingBayesianmethods

- One practical difficulty in applying Bayesian methods is that they typically requireinitialknowledgeofmanyprobabilities.Whentheseprobabilitiesarenotknowni n advance they are often estimated based on background knowledge, previouslyavailabledata, andassumptions aboutthe formof theunderlying distributions.

- Asecondpracticaldifficultyisthesignificantcomputationalcostrequiredtodetermine the Bayes optimal hypothesis in the general case. In certain specializedsituations,this computational costcan be significantly reduced.

BAYESTHEOREM

Bayes theorem provides a way to calculate the probability of a hypothesis based onits prior probability, the probabilities of observing various data given the hypothesis,andtheobserved dataitself.

Notations

- P(h)*priorprobabilityof*h,reflectsanybackgroundknowledgeaboutthechancethat h is correct
- P(D)*priorprobabilityof*D,probabilitythatDwillbeobserved
- P(D|h)probabilityofobservingD givenaworldinwhichhholds
- P(h|D) *posterior probability of* h, reflects confidence that h holds after D has beenobserved

Bayes theorem is the cornerstone of Bayesian learning methods because it providesa way to calculate the posterior probability ***P(h|D),*** from the prior probability ***P(h),***togetherwith ***P(D)*** and ***P(D(h).***

Bayes Theorem:

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

P(h|D)increaseswith***P(h)*** andwith***P(D|h)***accordingtoBayestheorem.

P(h|D)decreasesas***P(D)***increases,becausethemoreprobableitisthat***D***willbeobserve dindependentof***h,*** the lessevidence***D***providesin support of***h.***

MaximumaPosteriori(MAP)Hypothesis

- InmanylearningscenariosthelearnerconsiderssomesetofcandidatehypothesesH and is interested in finding the most probable hypothesis $\boldsymbol{h \in H}$ given theobserveddata***D***.Anysuch maximallyprobablehypothesisiscalled***maximuma posteriori(MAP)hypothesis***.
- Bayestheoremtocalculatetheposteriorprobabilityofeachcandidatehypothesisis$\boldsymbol{h_{MAP}}$ isaMAPhypothesisprovided

$$h_{MAP} = \underset{h \in H}{argmax}\ P(h|D)$$

$$= \underset{h \in H}{argmax}\ \frac{P(D|h)P(h)}{P(D)}$$

$$= \underset{h \in H}{argmax}\ P(D|h)P(h)$$

- P(D)canbedropped,becauseitisaconstantindependentofh

MaximumLikelihood(ML)Hypothesis

Insomecases,it isassumedthateveryhypothesisinHisequallyprobableapriori ***(P(h$_i$)=P(h$_j$)***forall***h$_i$***and***h$_j$***inH). Inthiscasethebelowequationcanbesimplifiedandneedonlyconsidertheterm ***P(D|h)***tofindthemostprobablehypothesis.

$$h_{MAP} = \underset{h \in H}{argmax}\ P(D|h)P(h)$$

the equation can be simplified

$$h_{ML} = \underset{h \in H}{argmax}\ P(D|h)$$

P(D|h)isoftencalledthe***likelihood***ofthedata***D***given***h,***andanyhypothesisthatmaximizes***P(D|h)***is called a***maximum likelihood***(ML)hypothesis

Example

Consideramedical diagnosisprobleminwhichtherearetwo alternativehypotheses

- Thepatienthasaparticularformofcancer(denotedby***cancer***)
- Thepatientdoesnot(denotedby¬***cancer***)

Theavailabledataisfromaparticularlaboratorywithtwopossibleoutcomes: +(positive)and- (negative)

$$P(cancer) = .008 \qquad P(\neg cancer) = 0.992$$
$$P(\oplus|cancer) = .98 \qquad P(\ominus|cancer) = .02$$
$$P(\oplus|\neg cancer) = .03 \qquad P(\ominus|\neg cancer) = .97$$

- Supposeanewpatientisobservedforwhomthe labtestreturnsapositive(+)result.
- Shouldwediagnosethepatientashavingcancerornot?

$$P(\oplus|cancer)P(cancer) = (.98).008 = .0078$$
$$P(\oplus|\neg cancer)P(\neg cancer) = (.03).992 = .0298$$
$$\Rightarrow h_{MAP} = \neg cancer$$

BAYESTHEOREMANDCONCEPTLEARNING

WhatistherelationshipbetweenBayestheoremandtheproblemofconceptlearning ?

SinceBayestheoremprovidesaprincipledwaytocalculatetheposteriorprobabilityof each hypothesis given the training data, and can use it as the basis for astraightforward learning algorithm that calculates the probability for each possiblehypothesis,then outputsthe most probable.

Brute-ForceBayesConceptLearning

Wecandesignastraightforwardconceptlearningalgorithmtooutputthemaximumapost eriorihypothesis,based on Bayes theorem,as follows:

Brute-Force MAP LEARNING algorithm

1. For each hypothesis h in H calculate the posterior probability

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

2. Output the hypothesis h_{MAP} with the highest posterior probability

$$h_{MAP} = \underset{h \in H}{argmax}\ P(h|D)$$

Inorderspecifyalearningproblemforthe**BRUTE-FORCEMAPLEARNING** algorithmwemustspecifywhatvaluesaretobeused for ***P(h)***andfor ***P(D|h)?***

Letschoose***P(h)***andfor***P(D|h)***tobeconsistentwiththefollowingassumptions:

- Thetrainingdata***D***isnoisefree(i.e.,$d_i = c(x_i)$)
- Thetargetconcept***c***iscontainedinthehypothesisspaceH
- Wehavenoapriorireasontobelievethatanyhypothesisismoreprobablethananyother.

Whatvaluesshouldwespecifyfor***P(h)?***

- Givennopriorknowledgethatonehypothesisismorelikelythananother,itisreasonableto assign thesameprior probabilitytoeveryhypothesis***h***in***H*.**
- AssumethetargetconceptiscontainedinHandrequirethatthesepriorprobabilitiessum to 1.

$$P(h) = \frac{1}{|H|} \text{ for all } h \in H$$

Whatchoiceshallwemakefor***P(D|h)***?

- ***P(D|h)***istheprobabilityof observingthe targetvalues D=***(d$_1$. ..d$_m$)*** forthefixedset ofinstances***(x$_1$.. . x$_m$),***given a worldinwhich hypothesis***h***holds
- Since we assume noise-free training data, the probability of observing classification***d$_i$***given***h***isjust***1***if***d$_i$=h(x$_i$)***and0if***d$_i$#h(x$_i$).***Therefore,

$$P(D|h) = \begin{cases} 1 & \text{if } d_i = h(x_i) \text{ for all } d_i \in D \\ 0 & \text{otherwise} \end{cases}$$

Giventhesechoicesfor***P(h)***andfor***P(D|h)***wenowhaveafully-definedproblemfor the above**BRUTE-FORCEMAPLEARNING** algorithm.

InafirststepwehavetodeterminetheprobabilitiesforP(h|D)

h is **inconsistent** with training data D

$$P(h|D) = \frac{0 \cdot P(h)}{P(D)} = 0$$

h is **consistent** with training data D

$$P(h|D) = \frac{1 \cdot \frac{1}{|H|}}{P(D)} = \frac{1 \cdot \frac{1}{|H|}}{\frac{|VS_{H,D}|}{|H|}} = \frac{1}{|VS_{H,D}|}$$

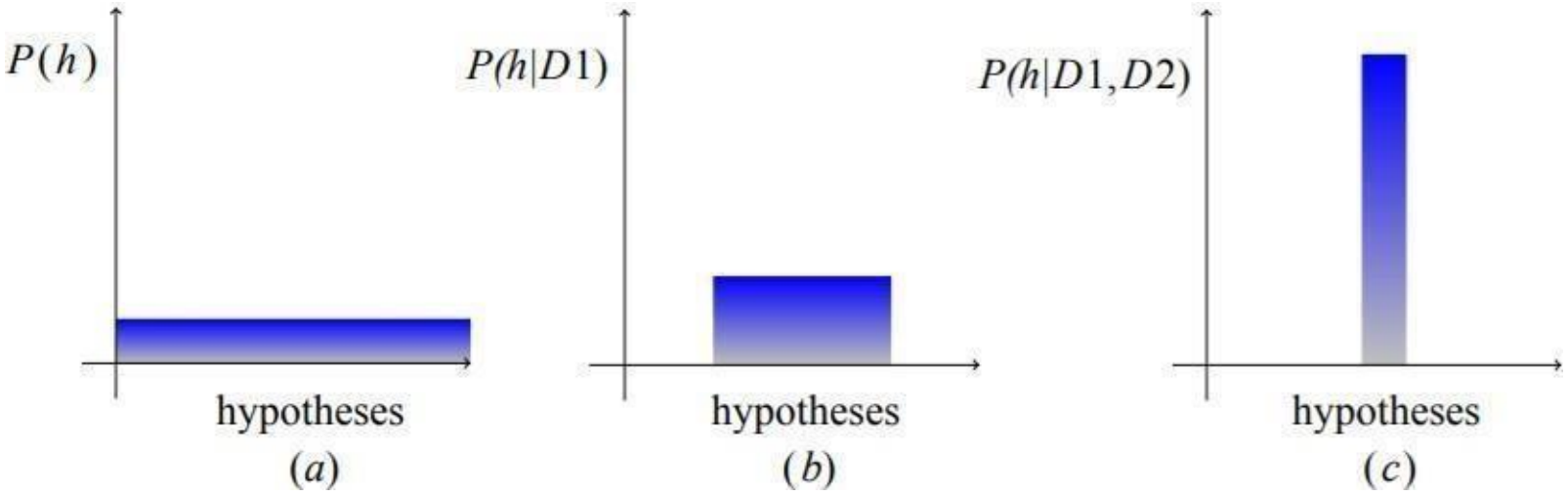
Tosummarize,Bayestheoremimpliesthattheposteriorprobability$P(h|D)$underourassumed$P(h)$ and $P(D|h)$ is

$$P(h|D) = \begin{cases} \frac{1}{|VS_{H,D}|} & \text{if } h \text{ is consistent with } D \\ 0 & \text{otherwise} \end{cases}$$

where $|VS_{H,D}|$isthenumberofhypotheses fromHconsistentwithD

TheEvolutionofProbabilitiesAssociatedwithHypotheses

- Figure(a)allhypotheseshavethesameprobability.
- Figures (b) and (c), As training data accumulates, the posterior probability forinconsistenthypothesesbecomeszerowhilethetotalprobabilitysummingto1 issharedequally among the remainingconsistent hypotheses.



MAPHypothesesandConsistent Learners

A learning algorithm is a consistent learner if it outputs a hypothesisthat commits zero errors overthetraining examples.

EveryconsistentlearneroutputsaMAPhypothesis,ifweassumeauniformpriorprobabilitydistributionover H(P(h_i)=P(h_j)foralli,j),anddeterministic,noisefreetrainingdata(P(D|h)=1ifD and h areconsistent, and0 otherwise).

Example:

- FIND-S outputs a consistent hypothesis, it will output a MAP hypothesis under the probabilitydistributionsP(h) andP(D|h)defined above.
- Are there other probability distributions for P(h) and P(D|h) under which FIND-S outputs MAPhypotheses?Yes.
- BecauseFIND-Soutputsamaximallyspecifichypothesisfromtheversionspace,itsoutputhypothesis willbeaMAPhypothesisrelativetoanypriorprobabilitydistributionthatfavoursmorespecifichypotheses .

- Bayesianframeworkis awaytocharacterizethebehaviouroflearningalgorithms
- ByidentifyingprobabilitydistributionsP(h)andP(D|h)underwhich theoutput isa optimal hypothesis, implicit assumptions of the algorithm can be characterized(InductiveBias)
- Inductiveinferenceismodelledbyanequivalentprobabilisticreasoningsystembase d on Bayes theorem

MAXIMUMLIKELIHOODANDLEAST-SQUAREDERRORHYPOTHESES

Consider the problem of learning a continuous-valued target function such as neuralnetworklearning, linearregression, and polynomial curvefitting

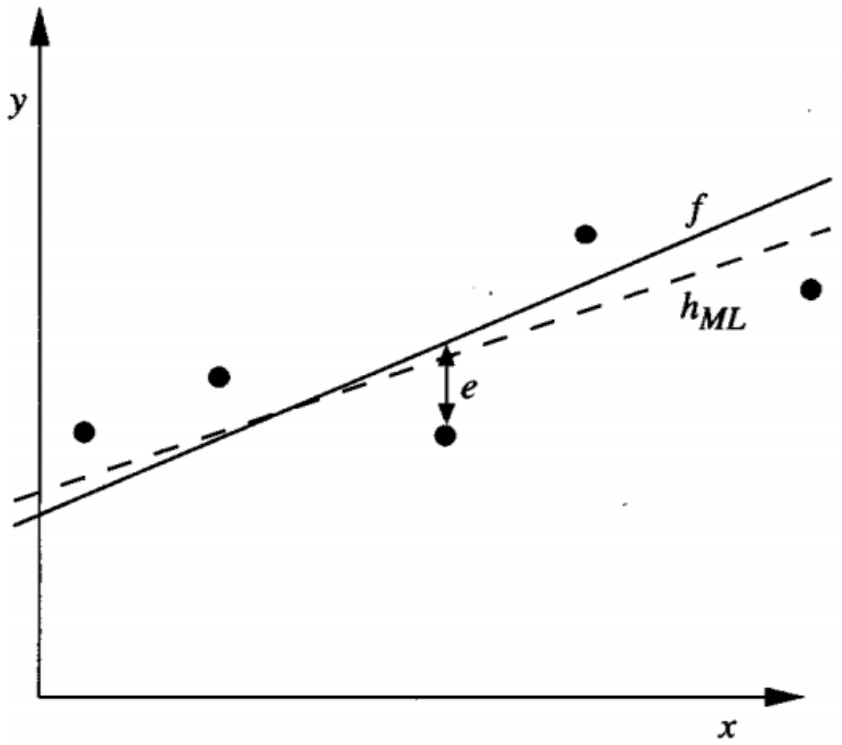
A straightforward Bayesian analysis will show that under certain assumptions anylearning algorithm that minimizes the squared error between the output hypothesispredictionsandthetrainingdata willoutput amaximumlikelihood(ML)hypothesis

LearningAContinuous-ValuedTargetFunction

- LearnerLconsidersaninstancespaceXandahypothesisspaceHconsistingofsomeclassofreal-valuedfunctionsdefinedoverX, i.e.,(∀h∈H)[h:X → R] andtrainingexamplesofthe form $<x_i,d_i>$
- TheproblemfacedbyListolearnanunknowntargetfunctionf:X→ R
- A set of m training examples is provided, where the target value of each example is corrupted byrandomnoisedrawnaccordingtoaNormalprobabilitydistributionwithzero mean($d_i=f(x_i)+e_i$)
- Eachtrainingexample isapairoftheform(x_i,d_i)wheredi= $f(x_i)+e_i$.
  - Heref(x_i)isthenoise-freevalueofthetargetfunctionande$_i$isarandomvariablerepresentingthe noise.
  - Itisassumedthatthevaluesofthee$_i$are***drawnindependently***andthattheyaredistributedaccordin gtoa***Normaldistribution*** with zeromean.
- Thetaskofthelearneristooutputa***maximumlikelihoodhypothesis***,or,equivalently,aMAPhypothe sisassumingallhypotheses are equally probable apriori.

LearningALinearFunction

- Thetargetfunctionfcorrespondstothesolidline.
- The training examples (xi , di) are assumed tohaveNormallydistributednoiseeiwithzeromeana ddedto thetruetargetvalue f(xi).
- Thedashedlinecorrespondstothehypothesish$_{ML}$ with least-squared training error, hence themaximum likelihood hypothesis.
- Notice that the maximum likelihood hypothesis isnotnecessarilyidenticaltothecorrecthypothesis, f, because it is inferred from only alimitedsampleofnoisytraining data



Beforeshowingwhyahypothesisthatminimizesthesumofsquarederrorsinthissettingisalsoamaximum likelihood hypothesis, let us quickly review ***probability densities and Normaldistributions***

ProbabilityDensityforcontinuousvariables

$$p(x_0) \equiv \lim_{\epsilon \to 0} \frac{1}{\epsilon} P(x_0 \leq x < x_0 + \epsilon)$$

e:arandomnoisevariablegeneratedbyaNormalprobabilitydistribution

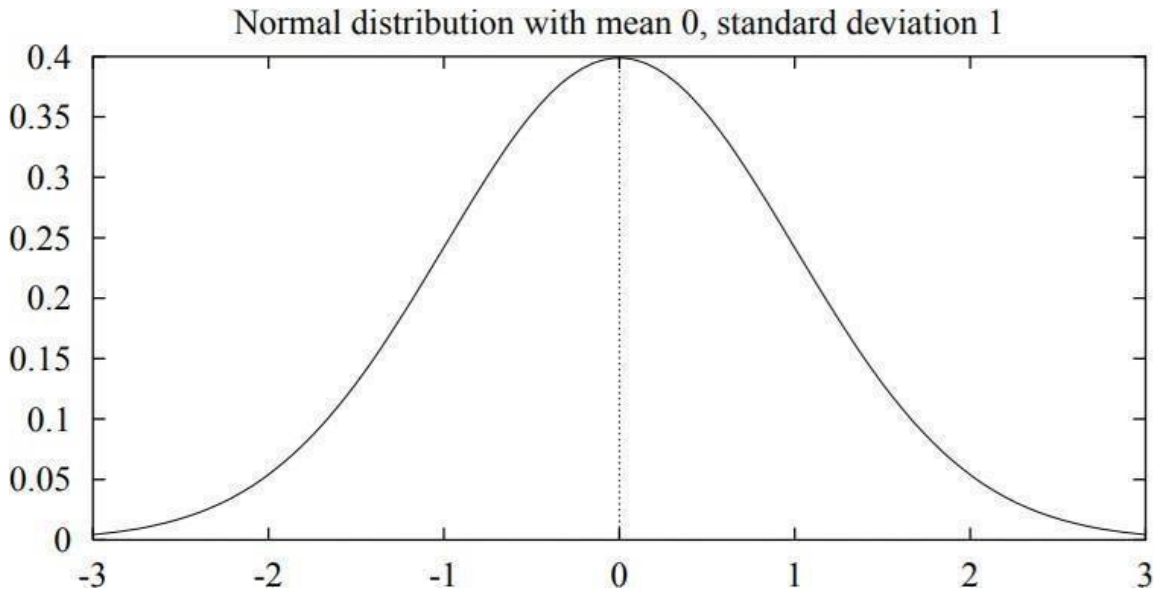
$<x_1 \ldots x_m>$:thesequenceofinstances(asbefore)

$<d_1 \ldots d_m>$:thesequenceoftargetvalueswithdi=f(x_i)+e_i

NormalProbabilityDistribution(GaussianDistribution)

ANormal distributionisasmooth,bell-shapeddistributionthatcanbecompletelycharacterizedby itsmean μand itsstandard deviation σ

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$



T.Aparna, Assistant Professor, CSE

- A Normal distribution is fully determined by two parameters in the formula: μ and σ .
- If the random variable X follows a normal distribution:
  - The probability that X will fall into the interval (a, b) is $\int_a^b p(x)dx$
  - The expected, or ***mean value of X***, $E[X] = \mu$
  - The ***variance of X***, $Var(X) = \sigma^2$
  - The ***standard deviation of X***, $\sigma_x = \sigma$
- The ***Central Limit Theorem*** states that the sum of a large number of independent, identically distributed random variables follows a distribution that is approximately ***Normal***.

Usingthepreviousdefinitionofh_{ML}wehave

$$h_{ML} = \underset{h \in H}{argmax}\, p(D|h)$$

Assumingtrainingexamplesaremutuallyindependentgivenh,wecanwriteP(D|h)astheproductofthe various(d_i|h)

$$h_{ML} = \underset{h \in H}{argmax} \prod_{i=1}^{m} p(d_i|h)$$

Given the noise e_i obeys a Normal distribution with zero mean and unknown variance σ^2 , each $d_i$mustalsoobeyaNormaldistributionaroundthetruetargetvaluef(x_i).Becausewearewritingtheexpression for P(D|h), we assumehisthe correctdescriptionoff.Hence, $\mu = f(x_i)=h(x_i)$

$$h_{ML} = \underset{h \in H}{argmax} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(d_i - h(x_i))^2}$$

$$h_{ML} = \underset{h \in H}{\operatorname{argmax}} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(d_i-\mu)^2}$$

$$= \underset{h \in H}{\operatorname{argmax}} \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(d_i-h(x_i))^2}$$

Itiscommontomaximizethelesscomplicatedlogarithm,whichisjustifiedbecauseofthemonotonicityoffunction*p*.

$$= \underset{h \in H}{\operatorname{argmax}} \sum_{i=1}^{m} ln \frac{1}{\sqrt{2\pi\sigma^2}} - \frac{1}{2\sigma^2}(d_i - h(x_i))^2$$

Thefirstterminthisexpressionisaconstantindependentof***h***andcanthereforebediscarded

$$= \underset{h \in H}{\operatorname{argmax}} \sum_{i=1}^{m} -\frac{1}{2\sigma^2}(d_i - h(x_i))^2$$

Maximizingthisnegativetermisequivalenttominimizingthecorrespondingpositiveterm.

$$= \underset{h \in H}{\operatorname{argmin}} \sum_{i=1}^{m} \frac{1}{2\sigma^2}(d_i - h(x_i))^2$$

FinallyDiscardconstantsthatareindependentof***h***

$$h_{ML} = \underset{h \in H}{argmin} \sum_{i=1}^{m} (d_i - h(x_i))^2$$

- theh_{ML}isonethatminimizes thesumofthesquarederrors

WhyisitreasonabletochoosetheNormaldistributiontocharacterizenoise?

- goodapproximationofmanytypesofnoiseinphysicalsystems
- CentralLimitTheoremshowsthatthesumofasufficientlylargenumberofindependent,identic ally distributedrandomvariables itselfobeysaNormaldistribution

Onlynoiseinthetargetvalueisconsidered,notintheattributesdescribingtheinstancesthemsel ves

MAXIMUMLIKELIHOODHYPOTHESESFORP REDICTINGPROBABILITIES

Considerthesettinginwhichwewishtolearnanondeterministic(probabilistic)functi onf :X→ {0,1}, which has two discreteoutput values.

Wewantafunctionapproximatorwhoseoutputistheprobabilitythatf(x)=1Inother words , learnthe target function

$$f' : X \rightarrow [0,1] \text{ suchthat } f'(x) = P(f(x)=1)$$

Howcanwelearnf'usinganeuralnetwork?

Useofbruteforcewaywouldbetofirst collecttheobservedfrequenciesof1's and0's for each possible value of x and to then train the neural network to output thetargetfrequency for eachx.

Whatcriterionshouldweoptimizeinordertofindamaximumlikelihoodhypothesisfor f'in thissetting?

- FirstobtainanexpressionforP(D|h)
- AssumethetrainingdataDisoftheformD ={(x_1,d_1) ...(x_m,d_m)},where$d_i$is theobserved 0or1 valuefor f (x_i).
- Both$x_i$and$d_i$asrandomvariables,andassumingthateachtrainingexampleisdrawnindepen dently,we can writeP(D|h) as

$$P(D \mid h) = \prod_{i=1}^{m} P(x_i, d_i \mid h)$$

Applyingtheproductrule

$$P(D \mid h) = \prod_{i=1}^{m} P(d_i \mid h, x_i) P(x_i)$$

TheprobabilityP(d_i|h,x_i)

$$P(d_i|h, x_i) = \begin{cases} h(x_i) & \text{if } d_i = 1 \\ (1 - h(x_i)) & \text{if } d_i = 0 \end{cases} \qquad \text{equ (3)}$$

Re-expressitinamoremathematicallymanipulableform,as

$$P(d_i|h, x_i) = h(x_i)^{d_i}(1 - h(x_i))^{1-d_i} \qquad \text{equ (4)}$$

Equation(4)tosubstituteforP(d_i|h,x_i)inEquation(5)toobtain

$$P(D|h) = \prod_{i=1}^{m} h(x_i)^{d_i}(1 - h(x_i))^{1-d_i} P(x_i) \qquad \text{equ (5)}$$

Wewriteanexpressionforthemaximumlikelihoodhypothesis

$$h_{ML} = \underset{h \in H}{\operatorname{argmax}} \prod_{i=1}^{m} h(x_i)^{d_i} (1 - h(x_i))^{1-d_i} P(x_i)$$

Thelasttermisaconstantindependentofh, so itcanbedropped

$$h_{ML} = \underset{h \in H}{\operatorname{argmax}} \prod_{i=1}^{m} h(x_i)^{d_i} (1 - h(x_i))^{1-d_i} \qquad \text{equ (6)}$$

Iteasiertoworkwiththelogofthelikelihood,yielding

$$h_{ML} = \underset{h \in H}{\operatorname{argmax}} \sum_{i=1}^{m} d_i \ln h(x_i) + (1 - d_i) \ln(1 - h(x_i)) \qquad \text{equ (7)}$$

Equation(7)describesthequantitythatmustbemaximizedinordertoobtainthemaximumlikeliho od hypothesis in our currentproblemsetting

GradientSearchtoMaximizeLikelihoodinaNeuralNet

Deriveaweight-trainingruleforneuralnetworklearningthatseekstomaximizeG(h,D)usinggradientascent

- ThegradientofG(h,D)isgivenbythevectorofpartialderivativesofG(h,D) withrespecttothevariousnetwork weights thatdefinethehypothesishrepresentedby thelearned network
- Inthiscase,thepartialderivativeofG(h,D)withrespecttoweightw_{jk}frominputktounitjis

$$\frac{\partial G(h,D)}{\partial w_{jk}} = \sum_{i=1}^{m} \frac{\partial G(h,D)}{\partial h(x_i)} \frac{\partial h(x_i)}{\partial w_{jk}}$$

$$= \sum_{i=1}^{m} \frac{\partial (d_i \ln h(x_i) + (1 - d_i) \ln(1 - h(x_i)))}{\partial h(x_i)} \frac{\partial h(x_i)}{\partial w_{jk}}$$

$$= \sum_{i=1}^{m} \frac{d_i - h(x_i)}{h(x_i)(1 - h(x_i))} \frac{\partial h(x_i)}{\partial w_{jk}} \qquad \text{equ (1)}$$

Supposeourneuralnetworkisconstructedfromasinglelayerofsigmoidunits.Then,

$$\frac{\partial h(x_i)}{\partial w_{jk}} = \sigma'(x_i)x_{ijk} = h(x_i)(1 - h(x_i))x_{ijk}$$

wherex_{ijk}isthekthinputtounitjfortheithtrainingexample,andd(x)isthederivativeofthesigmoidsquashingfunction.

Finally,substitutingthisexpressionintoEquation(1),weobtainasimpleexpressionforthederivatives thatconstitute the gradient

$$\frac{\partial G(h, D)}{\partial w_{jk}} = \sum_{i=1}^{m}(d_i - h(x_i))\ x_{ijk}$$

Because we seek to maximize rather than minimize P(D|h), we perform ***gradient ascent*** rather than***gradient descent search***. On each iteration of the search the weight vector is adjusted in the directionof the gradient,usingthe weight update rule

$$w_{jk} \leftarrow w_{jk} + \Delta w_{jk}$$

Where,

$$\Delta w_{jk} = \eta \sum_{i=1}^{m} (d_i - h(x_i))\ x_{ijk} \qquad \text{equ (2)}$$

whereηisasmallpositiveconstantthatdeterminesthestepsizeoftheigradientascentsearch

Itisinterestingtocomparethisweight-updateruletotheweight-updateruleusedbytheBACKPROPAGATION algorithm to minimize the sum of squared errors between predicted andobservednetwork outputs.

The BACKPROPAGATION update rule for output unit weights, re-expressed using our currentnotation, is

$$w_{jk} \leftarrow w_{jk} + \Delta w_{jk}$$

Where,

$$\Delta w_{jk} = \eta \sum_{i=1}^{m} h(x_i)(1 - h(x_i))(d_i - h(x_i))\ x_{ijk}$$

MINIMUMDESCRIPTIONLENGTHPRINCIPLE

- ABayesianperspectiveonOccam'srazor
- Motivatedbyinterpreting thedefinitionofh_{MAP}inthelightofbasicconceptsfrominformationtheory.

$$h_{MAP} = \underset{h \in H}{argmax}\ P(D|h)P(h)$$

whichcanbeequivalentlyexpressedintermsofmaximizingthe$\log_2$

$$h_{MAP} = \underset{h \in H}{argmax}\ \log_2 P(D|h) + \log_2 P(h)$$

oralternatively,minimizingthenegativeofthisquantity

$$h_{MAP} = \underset{h \in H}{argmin}\ -\log_2 P(D|h) - \log_2 P(h) \qquad \text{equ (1)}$$

- Thisequationcanbeinterpretedasastatementthatshorthypothesesarepreferred,assumingaparticul arrepresentationscheme forencoding hypothesesand data

Introductiontoabasicresultofinformationtheory

- Considertheproblemofdesigningacodetotransmitmessagesdrawnatrandom
- iisthemessage
- Theprobabilityofencounteringmessageiisp$_i$
- Interested in the most compact code; that is, interested in the code that minimizes theexpectednumberofbitswemusttransmitinordertoencodeamessagedrawnatrandom
- Tominimizetheexpectedcodelengthweshouldassignshortercodestomessagesthataremorepr obable
- ShannonandWeaver(1949)showedthattheoptimalcode(i.e.,thecodethatminimizestheexp ectedmessagelength)assigns- log, pibitsttoencodemessagei.
- ThenumberofbitsrequiredtoencodemessageiusingcodeCasthe***descriptionlengthofmessa geiwithrespectto C***,which wedenotebyL$_c$(i).

Interpretingtheequation

$$h_{MAP} = \underset{h \in H}{argmin} \; -\log_2 P(D|h) - \log_2 P(h) \qquad \text{equ (1)}$$

- -$\log_2$P(h):thedescriptionlengthofhunder theoptimal encodingforthehypothesis spaceHL_{CH}(h)= −$\log_2$P(h),whereC_H is the optimal codeforhypothesis space H.
- -$\log_2$P(D | h): the description length of the training data D given hypothesis h, under theoptimal encoding fro the hypothesis space H: L_{CH} (D|h) = −$\log_2$P(D| h) , where C $_{D|h}$ is theoptimal code for describing data D assuming that both the sender and receiver know thehypothesish.

RewriteEquation(1)toshowthath$_{MAP}$isthehypothesishthatminimizesthesumgivenbythedescription lengthof thehypothesis plusthedescription lengthof thedatagiventhehypothesis.

$$h_{MAP} = \underset{h \in H}{argmin} \; L_{C_H}(h) + L_{C_{D|h}}(D|h)$$

where$C_H$and $C_{D|h}$aretheoptimalencodingsforHandforDgivenh

TheMinimumDescriptionLength(MDL)principlerecommendschoosingthehypothesisthatminimizes thesumof thesetwo description lengthsof equ.

$$h_{MAP} = \underset{h \in H}{argmin}\, L_{C_H}(h) + L_{C_{D|h}}(D|h)$$

MinimumDescriptionLengthprinciple:

$$h_{MDL} = \underset{h \in H}{\text{argmin}}\, L_{C_1}(h) + L_{C_1}(D \mid h)$$

Where,codesC_1andC_2torepresentthehypothesisandthedatagiventhehypothesis

TheaboveanalysisshowsthatifwechooseC_1tobetheoptimalencodingofhypothesesC_H, andifwe choose C_2 to betheoptimalencoding $C_{D|h}$,then $h_{MDL} = h_{MAP}$

ApplicationtoDecisionTreeLearning

ApplytheMDLprincipletotheproblemoflearningdecisiontreesfromsometrainingdata.

Whatshouldwechooseforthe$representations C_1$ *and* C_2 *ofhypothesesanddata?*

- **For C_1:** C_1 might be some obvious encoding, in which the description length grows with thenumberof nodes and withthenumber ofedges
- **For C_2:** Suppose that the sequence of instances (x_1 . . .x_m) is already known to both the transmitterandreceiver,sothatwe need onlytransmittheclassifications(f(x_1)...f (x_m)).

  Nowifthetrainingclassifications(f(x_1)...f(x_m))areidenticaltothepredictionsofthehypothesis,thentherei snoneedtotransmitanyinformationabouttheseexamples.Thedescriptionlengthof theclassificationsgiventhehypothesis ZERO

  If examples are misclassified by h, then for each misclassification we need to transmit a messagethatidentifieswhichexample is misclassifiedaswellas itscorrectclassification

Thehypothesish$_{MDL}$ undertheencodingC_1andC_2isjusttheonethatminimizesthesumofthesedescription lengths.

- MDLprincipleprovidesawayfortradingoffhypothesiscomplexityforthenumberoferrorscommit tedbythe hypothesis
- MDLprovidesawaytodealwiththeissueofoverfittingthedata.
- Shortimperfecthypothesismaybeselectedoveralongperfecthypothesis.

MachineLearning:Lecture8

ComputationalLearning Theory (BasedonChapter7ofMitchellT..,MachineLearning, 1997)

Overview

- Aretheregenerallawsthatgovernlearning?
  - ***SampleComplexity:***Howmanytrainingexamplesareneededfora learner to converge (with high probability) to a successfulhypothesis?
  - ***ComputationalComplexity:***Howmuchcomputationaleffortisne eded for a learner to converge (with high probability) to asuccessful hypothesis?
  - ***MistakeBound:***Howmanytrainingexampleswillthelearnermis classifybefore convergingtoasuccessful hypothesis?
- Thesequestions willbeansweredwithintwoanalyticalframeworks:
  - The***ProbablyApproximatelyCorrect(PAC)***framework
  - The***MistakeBound***framework

Overview(Cont'd)

- Rather than answering these questions forindividual learners, we will answer them forbroadclassesoflearners.Inparticularwewillconsider:
  - Thesizeorcomplexityofthehypothesisspaceconsideredby thelearner.
  - Theaccuracytowhichthetargetconceptmustbeapproximated.
  - Theprobabilitythatthelearnerwilloutputasuccessfulhypothesis.
  - Themannerinwhichtrainingexamplesarepresentedto thelearner.

ThePACLearningModel

- **Definition:**Consider a concept class ***C***defined over a set of instances ***X*** of length ***n***andalearner***L***usinghypothesisspace***H***.***C***is***PA C-learnable*** by ***L*** using ***H*** if for all ***c*** $\in$ ***C***,distributions ***D*** over ***X***, εsuch that ***0***<ε<***1/2***,and δsuch that ***0***<δ<***1/2***, learner ***L*** will, withprobability at least ***(1-*** δ***),*** output a hypothesis***h*** $\in$ ***H***such that ***error***$_D$***(h)*** $\leq\varepsilon$, in time that is***polynomial***in***1/***ε***, 1/***δ***,n,***and***size(c).***

SampleComplexityfor***Finite*** HypothesisSpaces

- Given any ***consistent*** learner, the number of examplessufficienttoassurethatanyhypothesiswillbeprobably(withprobability***(1-δ)***)approximately(withinerrorε)correctis***m=1/ε(ln|H|+ln(1/δ))***
- Ifthelearneris***notconsistent***,***m=1/2ε²(ln|H|+ln(1/δ))***
- ConjunctionsofBooleanLiteralsarealsoPAC-Learnableand***m=1/ε(n.ln3+ln(1/δ))***
- k-termDNFexpressionsarenotPAClearnablebecauseeven though they have polynomial sample complexity,theircomputationalcomplexityisnotpolyno

mial.

- Surprisingly,however,k-termCNFisPAClearnable.

Sample Complexity for InfiniteHypothesisSpacesI:VC-Dimension

- ThePACLearningframeworkhas2disadvantages:
  - Itcanleadtoweakbounds
  - SampleComplexityboundcannotbeestablishedforinf initehypothesisspaces
- Weintroducenewideasfordealingwiththeseproblems:
  - **Definition:**AsetofinstancesSisshatteredbyhypothesisspace H iff for every dichotomy of S there exists somehypothesisinHconsistentwiththisdichotomy.
  - **Definition:**The Vapnik-Chervonenkis dimension,VC(H),ofhypothesisspaceHdefinedoveri nstancespaceXisthesizeofthelargestfinitesubset of X

shatteredbyH.Ifarbitrarilylargefinitesetsof Xcan b

shatteredby H,thenVC(H)=∞

SampleComplexityforInfiniteHypothesisSpacesII

- ***Upper-Bound***onsamplecomplexity,usingthe***VC-Dimension***:$m \geq 1/\varepsilon(4\log_2(2/\delta)+8VC(H)\log_2(13/\varepsilon)$
- ***LowerBound***onsamplecomplexity,usingthe***VC-Dimension***:

Consider any concept class ***C*** such that $VC(C) \geq 2$, anylearner***L***,andany$0<\varepsilon<1/8$,and$0<\delta<1/100$.Then thereexistsadistribution***D***andtargetconceptin***C***such that if ***L*** observes fewer examples than$max[1/\varepsilon log(1/\delta),(VC(C)-1)/(32\varepsilon)]$ thenwithprobabilityatleastδ,***L***outputsahypothesis ***h***having$error_D(h)>\varepsilon$.

VC-DimensionforNeuralNetworks

- Let ***G*** be a layered directed acyclic graph with ***n***input nodes and ***s*≥*2*** internal nodes, each havingatmost ***r***inputs.Let***C***beaconceptclassover$\mathbf{R}^r$of VC dimension ***d***, corresponding to the set offunctions that can be described by each of the ***s***internal nodes. Let $C_G$be the G-composition of***C***, corresponding to the set of functions that canbe represented by ***G***. Then $VC(C_G) \leq 2ds\ log(es)$,where ***e***isthebaseofthenatural logarithm.
- This theorem can help us bound the VC-Dimensionofaneuralnetworkandthus,itssam

plecomplexity(See,[Mitchell,p.219])!

The*MistakeBound*ModelofLearning

- The***MistakeBound***frameworkisdifferentfromthe PAC framework as it considers learners thatreceive a sequence of training examples and thatpredict, upon receiving each example, what itstargetvalueis.
- The question asked in this setting is: ***"Howmany mistakes will the learner make in itspredictionsbeforeitlearnsthetargetconcept?"***
- This question is significant in practical settingswherelearningmust bedonewhilethesystemisinactualuse.

OptimalMistakeBounds

- **Definition:**Let ***C*** be an arbitrary nonemptyconcept class. The optimal mistake boundfor ***C***, denoted ***Opt(C)***, is the minimum overallpossiblelearningalgorithms***A***of$M_A(C)$.

$$Opt(C)=\min_{A \in Learning\_Algorithm} M_A(C)$$

- Foranyconceptclass***C,***theoptimalmistakeboundisboundasfollows:

$$VC(C) \leq Opt(C) \leq \log_2(|C|)$$

ACaseStudy:TheWeighted-MajorityAlgorithm

$a_i$denotestheithpredictionalgorithminthepool$A$ofalgorithm.$w_i$denotestheweightassociatedwitha_i.

- Foralliinitializew_i<--1
- Foreachtrainingexample<x,$c(x)$>
  - Initialize$q_0$and$q_1$to0
  - Foreachpredictionalgorithmai
    - If$a_i(x)=0$thenq_0<--q_0+w_i
    - If$a_i(x)=1$thenq_1<--q_1+w_i
  - If$q_1>q_0$thenpredict$c(x)=1$
  - If$q_0>q_1$thenpredict$c(x)=0$
  - If$q_0=q_1$thenpredict0or1atrandomfor$c(x)$
  - Foreachpredictionalgorithm$a_i$in$A$do
    - If$a_i(x)\neq c(x)$thenw_i<--βw_i

RelativeMistakeBoundfortheWeighted-MajorityAlgorithm

- Let***D***beanysequenceoftrainingexamples,let***A***be any set of n prediction algorithms, and let ***k*** bethe minimum number of mistakes made by anyalgorithm in ***A*** for the training sequence ***D***. Thenthe number of mistakes over ***D*** made by the***Weighted-Majority*** algorithm using $\beta=1/2$ is atmost ***$2.4(k+\log_2 n)$.***
- Thistheoremcanbegeneralizedforany$0\leq\beta\leq 1$ wheretheboundbecomes

$$(k\log_2 1/\beta+\log_2 n)/\log_2(2/(1+\beta))$$

INSTANCE-BASELEARNING

- Instance-basedlearningmethodssimplystorethetrainingexamplesinsteadof learningexplicitdescriptionof the target function.
  - Generalizingtheexamplesispostponeduntilanewinstancemustbeclassified.
  - Whenanewinstanceisencountered,itsrelationshiptothestoredexamplesisexamined in orderto assign atargetfunctionvalue for the newinstance.
- Instance-basedlearningincludes***nearestneighbor***,***locallyweightedregression***and ***case-based reasoning*** methods.
- Instance-based methods are sometimes referred to as **lazy** learningmethodsbecausetheydelayprocessinguntilanewinstancemust beclassified.
- Akeyadvantageoflazylearningisthatinsteadofestimatingthe

targetfunction once for the entire instance space, these methods can estimateit locally anddifferentlyfor each newinstance tobeclassified.

k-NearestNeighborLearning

- k-NearestNeighborLearningalgorithmassumesallinstancescorrespondtopoints in the n-dimensionalspace$\mathbb{R}^n$
- Thenearestneighborsofaninstanceareдefinedintermsof Euclideandistance.
- Euclideandistancebetweentheinstances$x_i=<x_{i1},\ldots,x_{in}>andx_j=<x_{j1},\ldots,x_{jn}>$are:

$$d(x_i,x_j)=\sqrt{\sum_{r=1}^{n}(x_{ir}-x_{jr})^2}$$

- Foragivenquery instancex_q,$f(x_q)$iscalculatedthefunctionvaluesofk-nearestneighbor ofx_q

k-NearestNeighborLearning

- Storealltrainingexamples<x_i,$f(x_i)$>
- Calculate$f(x_q)$foragivenqueryinstance$x_q$usingk-nearestneighbor
- **Nearest neighbor:(k=1)**
  - Locatethenearesttraingexamplex_n,andestimate $f(x_q)$as
  - $f(x_q) \leftarrow f(x_n)$
- **k-Nearestneighbor:**
  - Locateknearest traingexamples,andestimate$f(x_q)$as
  - Ifthetargetfunctionisreal-valued,takemeanoff-valuesofknearestneighbors.

$$f(x_q) = \frac{\sum_{i=1}^{k} f(x_i)}{k}$$

  - If thetargetfunctionisdiscrete-valued,takeavoteamongf-valuesof k nearestneighbors.

WhenToConsiderNearestNeighbor

- InstancesmaptopointsinR^n
- Lessthan20attributesperinstance
- Lotsoftrainingdata
- **Advantages**
  - Trainingisveryfast
  - Learn complextargetfunctions
  - Canhandlenoisydata
  - Doesnotlooseanyinformation
- **Disadvantages**
  - Slowatquerytime
  - Easilyfooledbyirrelevantattributes

Distance-Weighted kNN

Might want weight nearer neighbors more heavily...

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} w_i f(x_i)}{\sum_{i=1}^{k} w_i}$$

where

$$w_i \equiv \frac{1}{d(x_q, x_i)^2}$$

and $d(x_q, x_i)$ is distance between x_q and x_i

Note now it makes sense to use *all* training examples instead of just k

CurseofDimensionality

Imagine instances described by 20 attributes, but only 2 are relevant to target function

Curse of dimensionality: nearest nbr is easily mislead when high-dimensional X

One approach:

- Stretch jth axis by weight z_j, where $z_1, \ldots, z_n$ chosen to minimize prediction error
- Use cross-validation to automatically choose weights $z_1, \ldots, z_n$
- Note setting z_j to zero eliminates this dimension altogether

LocallyWeightedRegression

- KNNformslocalapproximationtofforeachquerypointxq
- Whynotforman explicitapproximationf(x)forregionsurroundingxq

  ➔LocallyWeightedRegression
- **Locallyweightedregression**usesnearbyordistance-weightedtrainingexamplestoform this localapproximation tof.
- Wemight approximatethetargetfunctionintheneighborhoodsurroundingx,using alinearfunction,a quadraticfunction, amultilayer neuralnetwork.
- Thephrase"locallyweightedregression"iscalled
  - ***local***becausethe function isapproximatedbasedonlyondatanearthequerypoint,
  - ***weighted*** because the contribution of each training example is weighted by itsdistancefromthequerypoint,and
  - ***regression***becausethisisthetermusedwidely inthestatisticallearningcommunityfortheproblemof approximatingreal-valued functions.

LocallyWeightedRegression

- Given a new query instance xq, the general approach in locallyweightedregressionistoconstruct anapproximationfthatfitsthetrainingexamplesin theneighborhood surrounding xq.
- Thisapproximationisthenusedtocalculatethevalue f(xq),whichisoutput asthe estimated target value for the queryinstance.

LocallyWeightedLinearRegression

f is approximated near x_q using a linear function of the form

$$\hat{f}(x) = w_0 + w_1 a_1(x) + \cdots + w_n a_n(x)$$

$a_i(x)$ denotes the value of the ith attribute of the instance x.

Minimize the squared error

$$E_3(x_q) \equiv \frac{1}{2} \sum_{x \in \text{ k nearest nbrs of } x_q} (f(x) - \hat{f}(x))^2 \; K(d(x_q, x))$$

KernelfunctionK isthefunctionofdistancethatisusedtodetermine theweightofeach training example.

$$\Delta w_j = \eta \sum_{x \in \text{ k nearest nbrs of } x_q} K(d(x_q, x)) \; (f(x) - \hat{f}(x)) \; a_j(x)$$

RadialBasisFunctions

- One approach to function approximation that is closely related to distance-weightedregressionandalsotoartificialneuralnetworksislearningwithradialbasisfunctions.
- Thelearnedhypothesisisafunctionoftheform

$$\hat{f}(x) = w_0 + \sum_{u=1}^{k} w_u K_u(d(x_u, x))$$

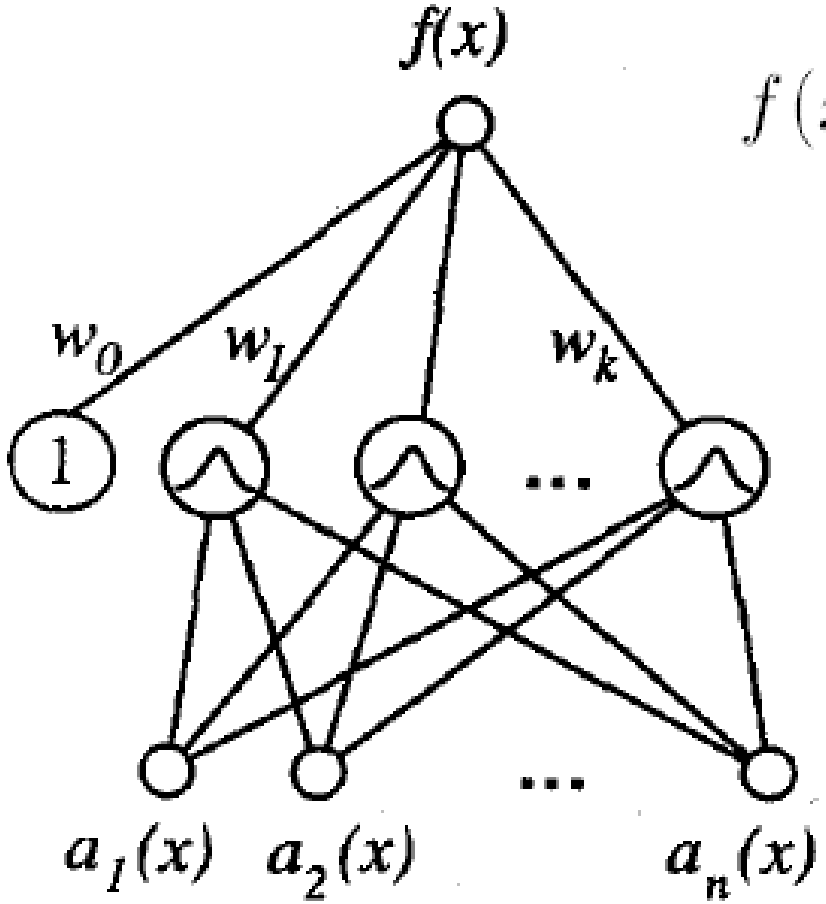
where each x_u is an instance from X and where the kernel function $K_u(d(x_u, x))$ is defined so that it decreases as the distance $d(x_u, x)$ increases. Here k is a user-provided constant that specifies the number of kernel functions to be included. Even though $\hat{f}(x)$ is a global approximation to $f(x)$, the contribution from each of the $K_u(d(x_u, x))$ terms is localized to a region nearby the point x_u. It is common

RadialBasisFunctions

to choose each function $K_u(d(x_u, x))$ to be a Gaussian function centered at the point x_u with some variance σ_u^2.

$$K_u(d(x_u, x)) = e^{\frac{1}{2\sigma_u^2} d^2(x_u, x)}$$

RadialBasisFunctionNetworks



$$f(x) = w_0 + \sum_{u=1}^{k} w_u K_u(d(x_u, x))$$

$$K_u(d(x_u, x)) = e^{\frac{1}{2\sigma_u^2} d^2(x_u, x)}$$

Eachhiddenunitproducesanactivationdeter mined by a Gaussian functioncentered atsomeinstance*xu*.

Therefore,itsactivationwillbeclosetozer o unless theinputxisnear*xu*.

Theoutputunitproducesalinearco mbination of the hidden unitactivations.

Case-basedreasoning

- Instance-basedmethods
  - lazy
  - classificationbasedonclassificationsofnear(similar)instances
  - data:pointsinn-dim.space
- Case-basedreasoning
  - asabove,butdatarepresentedinsymbolicform
- Newdistancemetricsrequired

Lazy&eagerlearning

- Lazy: generalizeatquerytime
  - kNN,CBR
- Eager:generalizebeforeseeingquery
  - Radialbasis,ID3,…
- Difference
  - eager*must*createglobalapproximation
  - lazy*can*createmanylocalapproximation
  - lazycanrepresentmorecomplexfunctionsusingsameH(H=linearfunctions)

MachineLearning:Lecture12

GeneticAlgorithms

(BasedonChapter9ofMitchell,T.,
MachineLearning,1997)

OverviewofGeneticAlgorithms(GAs)

- GAisalearningmethodmotivated byanalogyto biologicalevolution.
- GAs search the hypothesis space bygenerating successor hypotheses whichrepeatedlymutateandrecombinepar tsofthebestcurrently known hypotheses.
- In Genetic Programming (GP), entirecomputerprogramsareevolvedtocert ainfitnesscriteria.

GeneralOperationofGAs

- ***InitializePopulation***:generate***p***hypothesesatrandom.
- ***Evaluate***:foreach***p***,compute***fitness(p)***
- WhileMax_h***Fitness(h)<Threshold***do
  - ***Select***:probabilisticallyselectafractionofthebest***p***'sin***P.***Callthis new generationP_{New}
  - ***Crossover:*** probabilistically form pairs of the selected p's andproducetwooffspringsbyapplyingthecrossoveroperator.Addallo ffsprings to$P_{new.}$
  - ***Mutate***:Choosem%ofP_{New}withuniformprobability.Foreach,inve rtone randomly selected bit inits representation.
  - ***Update***:$P<-P_{new}$
  - ***Evaluate***:foreach***p***in***P***,compute***fitness(p)***
- Returnthehypothesisfrom***P***thathasthehighestfitness.

RepresentingHypotheses

- In GAs, hypotheses are often represented by bitstrings so that they can be easily manipulated bygeneticoperatorssuchasmutationandcrossover.
- ***Examples***:

(Outlook=OvercastvRain)^(Wind=Strong)

<=>***01110***

IFWind=StrongTHENPlayTennis=yes

<=> ***1111010***

where group 1 = 3-valued outlook,group 2 = 2-valued Windgroup3=2-valuedPlayTennis

GeneticOperators

- **CrossoverTechniques:**
  - Single-pointCrossover.Maskexample :11111000000
  - Two-pointCrossover. Maskexample:00111110000
  - UniformCrossover. Maskexample:10011010011
- **MutationTechniques:**
  - PointMutation
- **OtherOperators**:
  - SpecializationOperator
  - GeneralizationOperator

FitnessFunctionandSelection

- A simple measure for modeling the probability that ahypothesis will be selected is given by the ***fitnessproportionateselection***(or***roulettewheel***selecti on):

$$Pr(h_i)=Fitness(h_i)/\Sigma_{j=1}^{p}Fitness(h_j)$$

- Othermethods:***TournamentSelection***and***RankSe lection.***
- Inclassificationtasks,theFitnessfunctiontypicallyhas a component that scores the classificationaccuracy over a set of provided training examples.Other criteria can be added (e.g., complexity orgeneralityoftherule)

HypothesisSpaceSearch(I)

- GA search can move very abruptly (as compared toBackpropagation, for example), replacing a parenthypothesisbyanoffspringthatmayberadicallydifferentfromtheparent.
- Theproblemof***Crowding***:whenoneindividualismorefitthan others, this individual and closely related ones willtakeupalargefractionofthepopulation.
- ***Solutions:***
  - Usetournamentorrankselectioninsteadofroulettesele ction.
  - Fitnesssharing
  - restrictiononthekindsofindividualsallowedtoreco mbinetoformoffsprings.

HypothesisSpaceSearch(II): TheSchemaTheorem [Holland,75]

- ***Definition:***Aschemaisanystringcomposedof0s,1s and*swhere*means‘don’tcare’.
- ***Example:***schema0*10representsstrings0010and0110.
- ***The Schema Theorem:***More fit schemas willtend to grow in influence, especially schemascontaining a small number of defined bits (i.e.,containingalargenumberof*s),andespeciallywhen these defined bits are near one

anotherwithinthebit string.

Genetic Programming:RepresentingPrograms

- ***Example:*** $sin(x)+sqrt(x^2+y)$

GeneticProgramming:Crossover Operation

- ***Example:***

ModelsofEvolutionandLearningI:Lamarckian Evolution [Late 19th C]

- ***Proposition:***Experiencesofasingleorganismdirectly affect the genetic makeup of theiroffsprings.
- ***Assessment***: This proposition is wrong: thegenetic makeup of an individual is unaffected bythelifetimeexperienceofone'sbiologicalparents.
- ***However***: Lamarckian processes can sometimesimprovetheeffectivenessofcomputerizedgeneticalgorithms.

ModelsofEvolutionandLearningII:BaldwinEffect [1896]

- If a species is evolving in a changing environment, therewillbeevolutionarypressuretofavorindividualswiththecapabilitytolearnduringtheirlifetime.
- Those individuals who are able to learn many traits willrely less strongly on their genetic code to “hard-wire”traits. As a result, these individuals can support a morediverse gene pool, relying on individual learning of the“missing” or “sub-optimized” traits in the genetic code.This more diverse gene pool can, in turn, support morerapid evolutionary adaptation. Thus the capability oflearningcanacceleratetherateofevolutionaryadaptationof apopulation.

ParallelizingGeneticAlgorithms

- GAsarenaturallysuitedtoparallelimplementation.Different approaches were tried:
  - Coarse Grain: subdivides the population into distinctgroups of individuals (***demes***) and conducts a GA searchin each deme. Transfer between demes occurs (thoughinfrequently)byamigrationprocessinwhichindividualsfromonedemearecopiedortransferredtootherdemes
  - FineGrain:Oneprocessorisassignedperindividualinthe population and recombination takes place amongneighboringindividuals.

MachineLearning

Chapter10.LearningSetsofRules

TomM.Mitchell

LearningDisjunctiveSetsofR ules

- Method1:Learndecisiontree,converttorules
- Method2:Sequentialcoveringalgorithm:

1. Learnonerulewithhighaccuracy,anyco verage
2. Removepositiveexamplescoveredbythisrul e
3. Repeat

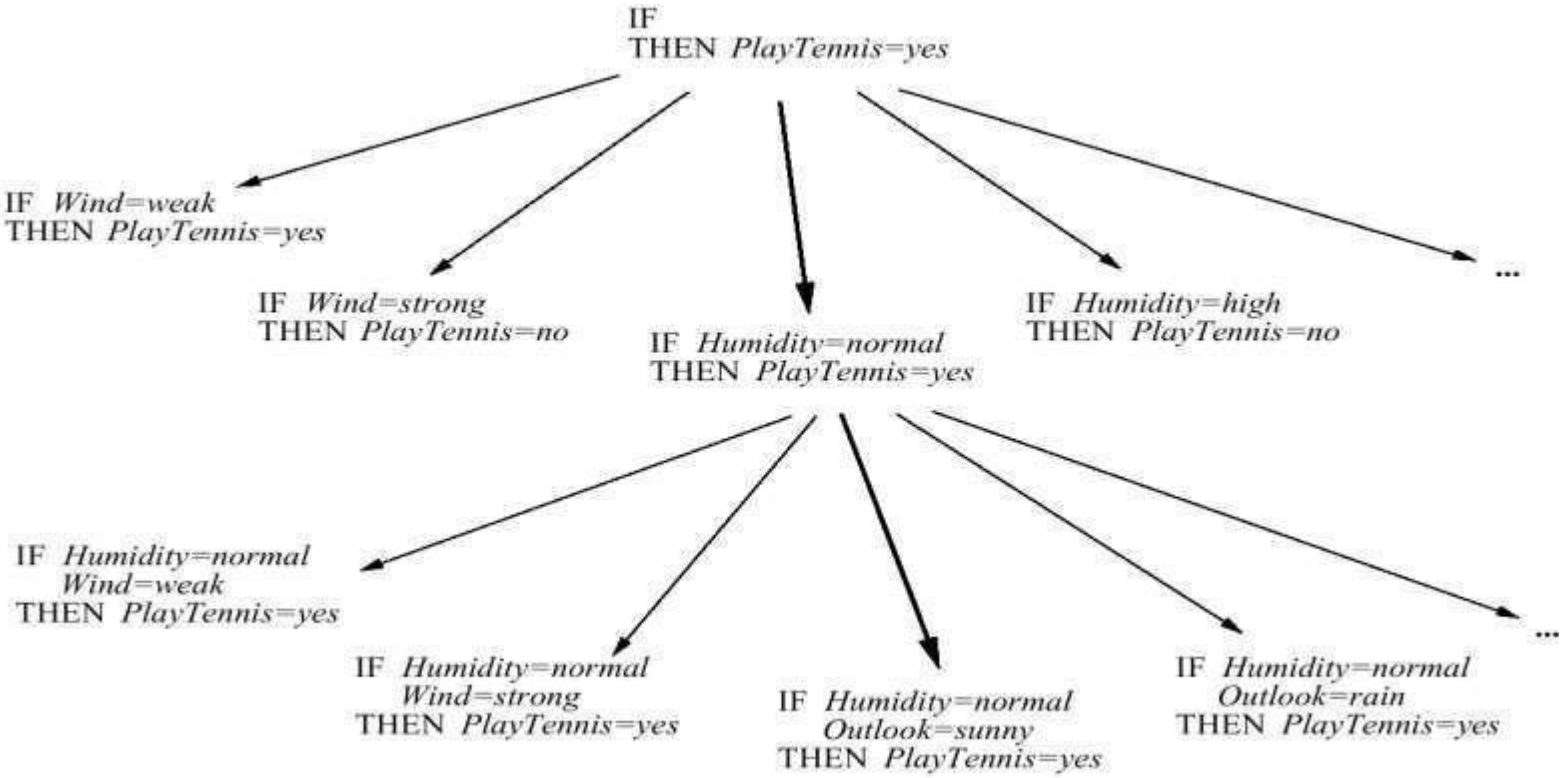
SequentialCoveringAlgorithm

SEQUENTIAL-COVERING(*Targetattribute;Attributes;Examples;Threshold*)

- Learnedrules←{}
- Rule ← LEARN-ONE-RULE(*Target_attribute*,*Attributes*,*Examples*)
- whilePERFORMANCE(*Rule*,*Examples*) > *Threshold*,do
  - Learned_rules←Learned_rules+*Rule*
  - *Examples*←*Examples*–{examplescorrectlyclassifiedby*Rule*}
  - *Rule*←LEARN-ONE-RULE(*Target_attribute*,*Attributes*,*Examples*)
  - *Learned_rules*←sort*Learned_rules*accordtoPERFORMANCEover*Examples*

- return*Learned_rules*

Learn-One-Rule



Learn-One-Rule(Cont.)

- *Pos*←positive*Examples*
- *Neg*←negative*Examples*
- while*Pos*,do
  *Learn*a*NewRule*
  - *NewRule*←mostgeneralrulepossible
  - *NewRule*←*Neg*
  - while*NewRuleNeg*,do
    *Addanewliteraltospecialize**NewRule*
    1. *Candidateliterals*←generatecandidates
    2. *Best_literal*←$\text{argmax}_{L \in \text{Candidateliterals}}$ *Performance(SpecializeRule(NewRule;L))*
    3. add*Best_literal*to*NewRule*preconditions
    4. NewRuleNeg←subsetof*NewRuleNeg*
    thatsatisfies*NewRule*preconditions
  - *Learned_rules*←*Learned_rules*+*NewRule*
  - *Pos*←*Pos*–{membersof *Poscoverd*by*NewRule*}

- Return*Learned_rules*

Subtleties:LearnOneRule

1. Mayuse*beamsearch*
2. Easilygeneralizestomulti-valuedtargetfunctions
3. Chooseevaluationfunctiontoguidesearch:
   - Entropy(i.e.,informationgain)
   - Sampleaccuracy: $\frac{n_c}{n}$

   wheren_c=correctrulepredictions,n=allpredictions

- *m*estimate: $\frac{n_c + mp}{n + m}$

VariantsofRuleLearningPrograms

- *Sequential*or*simultaneous*coveringofdata?
- General→specific,orspecific→general?
- Generate-and-test,orexample-driven?Whether and how to post-prune?
- Whatstatisticalevaluationfunction?

LearningFirstOrderRules

Whydothat?

- Canlearnsetsofrulessuchas
  Ancestor(x,y)←Parent(x;y)
  Ancestor(x;y)←Parent(x;z)^Ancestor(z;y)
- General purpose programming languagePROLOG:programsaresetsofsuch rules

FirstOrderRuleforClassifyingWebPages

[Slattery,1997]

course(A)←
has-word(A, instructor),Not has-word(A, good),link-from(A,B), has-word(B, assign),Notlink-

from(B,C)

Train:31/31,Test:31/34

FOIL($Target\_predicate, Predicates, Examples$)

- $Pos \leftarrow$ positive $Examples$
- $Neg \leftarrow$ negative $Examples$
- while Pos, do

  Learn a $NewRule$

  – $NewRule \leftarrow$ most general rule possible

  – $NewRuleNeg \leftarrow Neg$

  – while $NewRuleNeg$, do

  Add a new literal to specialize $NewRule$

  1. $Candidate\_literals \leftarrow$ generate candidates
  2. $Best\_literal \leftarrow$ $\text{argmax}_{L \in Candidate\_literals} Foil\_Gain(L, NewRule)$
  3. add $Best\_literal$ to $NewRule$ preconditions
  4. $NewRuleNeg \leftarrow$ subset of $NewRuleNeg$ that satisfies $NewRule$ preconditions

  – $Learned\_rules \leftarrow Learned\_rules + NewRule$

  – $Pos \leftarrow Pos -$ {members of Pos covered by $NewRule$}

- Return $Learned\_rules$

SpecializingRulesinFOIL

Learning rule: $P(x_1, x_2, \ldots, x_k) \leftarrow L_1 \ldots L_n$
Candidate specializations add new literal of form:

- $Q(v_1, \ldots, v_r)$, where at least one of the v_i in the created literal must already exist as a variable in the rule.
- $Equal(x_j, x_k)$, where x_j and x_k are variables already present in the rule
- The negation of either of the above forms of literals

InformationGaininFOIL

InformationGaininFOIL

$$Foil\_Gain(L, R) \equiv t\left(\log_2 \frac{p_1}{p_1 + n_1} - \log_2 \frac{p_0}{p_0 + n_0}\right)$$

Where

- L is the candidate literal to add to rule R
- p_0 = number of positive bindings of R
- n_0 = number of negative bindings of R
- p_1 = number of positive bindings of $R + L$
- n_1 = number of negative bindings of $R + L$
- t is the number of positive bindings of R also covered by $R + L$

Note

- $-\log_2 \frac{p_0}{p_0+n_0}$ is optimal number of bits to indicate the class of a positive binding covered by R

InductionasInvertedDeduction

Induction is finding h such that

$$(\forall \langle x_i, f(x_i) \rangle \in D)\ B \wedge h \wedge x_i \vdash f(x_i)$$

where

- x_i is ith training instance
- $f(x_i)$ is the target function value for x_i
- B is other background knowledge

So let's design inductive algorithm by inverting operators for automated deduction!

InductionasInvertedDeduction(Cont')

"pairs of people, $\langle u, v \rangle$ such that child of u is v,"

$$f(x_i): \quad Child(Bob, Sharon)$$
$$x_i: \quad Male(Bob), Female(Sharon), Father(Sharon, Bob)$$
$$B: \quad Parent(u, v) \leftarrow Father(u, v)$$

What satisfies $(\forall \langle x_i, f(x_i) \rangle \in D)\ B \wedge h \wedge x_i \vdash f(x_i)$?

$$h_1: Child(u, v) \leftarrow Father(v, u)$$
$$h_2: Child(u, v) \leftarrow Parent(v, u)$$

InductionasInvertedDeduction(Cont')

Induction is, in fact, the inverse operation of deduction, andcannot be conceivedto existwithoutthecorrespondingoperation, so that the question of relative importance cannotarise.Whothinksofaskingwhetheradditionorsubtraction isthe more important process in arithmetic? But at the sametime much difference in difficulty may exist between a directand inverse operation; : : : it must be allowed that inductiveinvestigations are of a far higher degree of difficulty andcomplexitythan any questions of deduction….

(Jevons1874)

InductionasInvertedDeduction(Cont')

We have mechanical *deductive* operators
$F(A, B) = C$, where $A \wedge B \vdash C$

need *inductive* operators

$O(B, D) = h$ where $(\forall \langle x_i, f(x_i) \rangle \in D)\ (B \wedge h \wedge x_i) \vdash f(x_i)$

InductionasInvertedDeduction(Cont')

Positives:

- Subsumes earlier idea of finding h that "fits" training data
- Domain theory B helps define meaning of "fit" the data

$$B \wedge h \wedge x_i \vdash f(x_i)$$

- Suggests algorithms that search H guided by B

InductionasInvertedDeduction(Cont')

Negatives:

- Doesn't allow for noisy data. Consider

$$(\forall \langle x_i, f(x_i) \rangle \in D)\ (B \wedge h \wedge x_i) \vdash f(x_i)$$

- First order logic gives a *huge* hypothesis space H

  $\rightarrow$ overfitting...

  $\rightarrow$ intractability of calculating all acceptable h's

Deduction:ResolutionRule

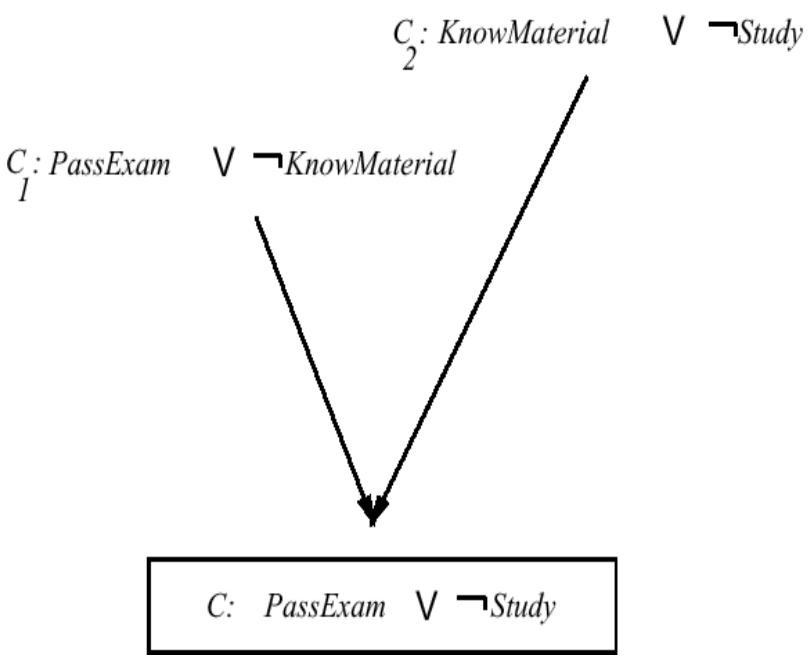
$$\frac{\begin{array}{lll} P & \vee & L \\ \neg L & \vee & R \end{array}}{\begin{array}{lll} P & \vee & R \end{array}}$$

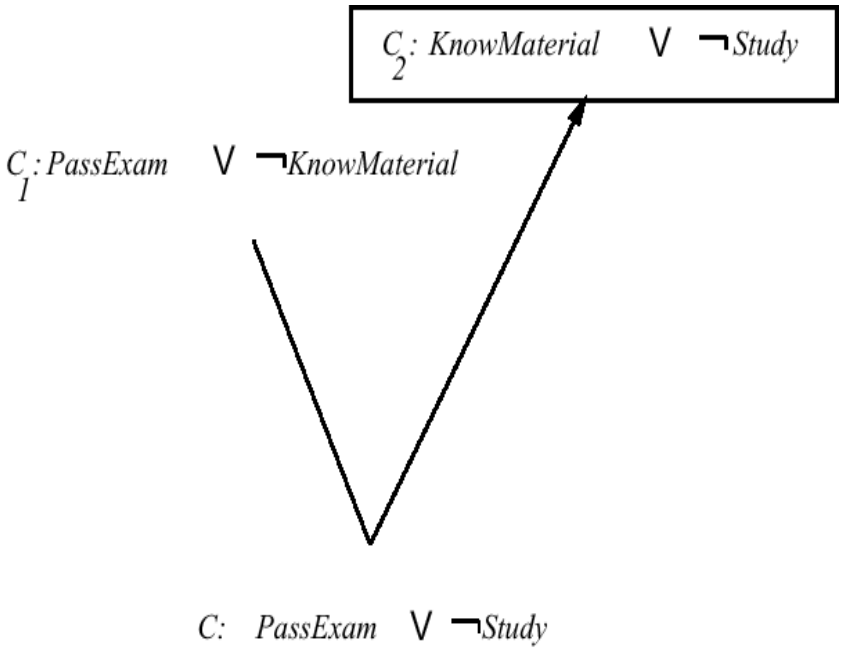
1. Given initial clauses C_1 and C_2, find a literal L from clause C_1 such that $\neg L$ occurs in clause C_2
2. Form the resolvent C by including all literals from C_1 and C_2, except for L and $\neg L$. More precisely, the set of literals occurring in the conclusion C is

$$C = (C_1 - \{L\}) \cup (C_2 - \{\neg L\})$$

where $\cup$ denotes set union, and "$-$" denotes set difference.

InvertingResolution





Inverted resolution(Propositional)

1. Given initial clauses C_1 and C, find a literal L that occurs in clause C_1, but not in clause C.
2. Form the second clause C_2 by including the following literals

$$C_2 = (C - (C_1 - \{L\})) \cup \{\neg L\}$$

Firstorderresolution

First order resolution:

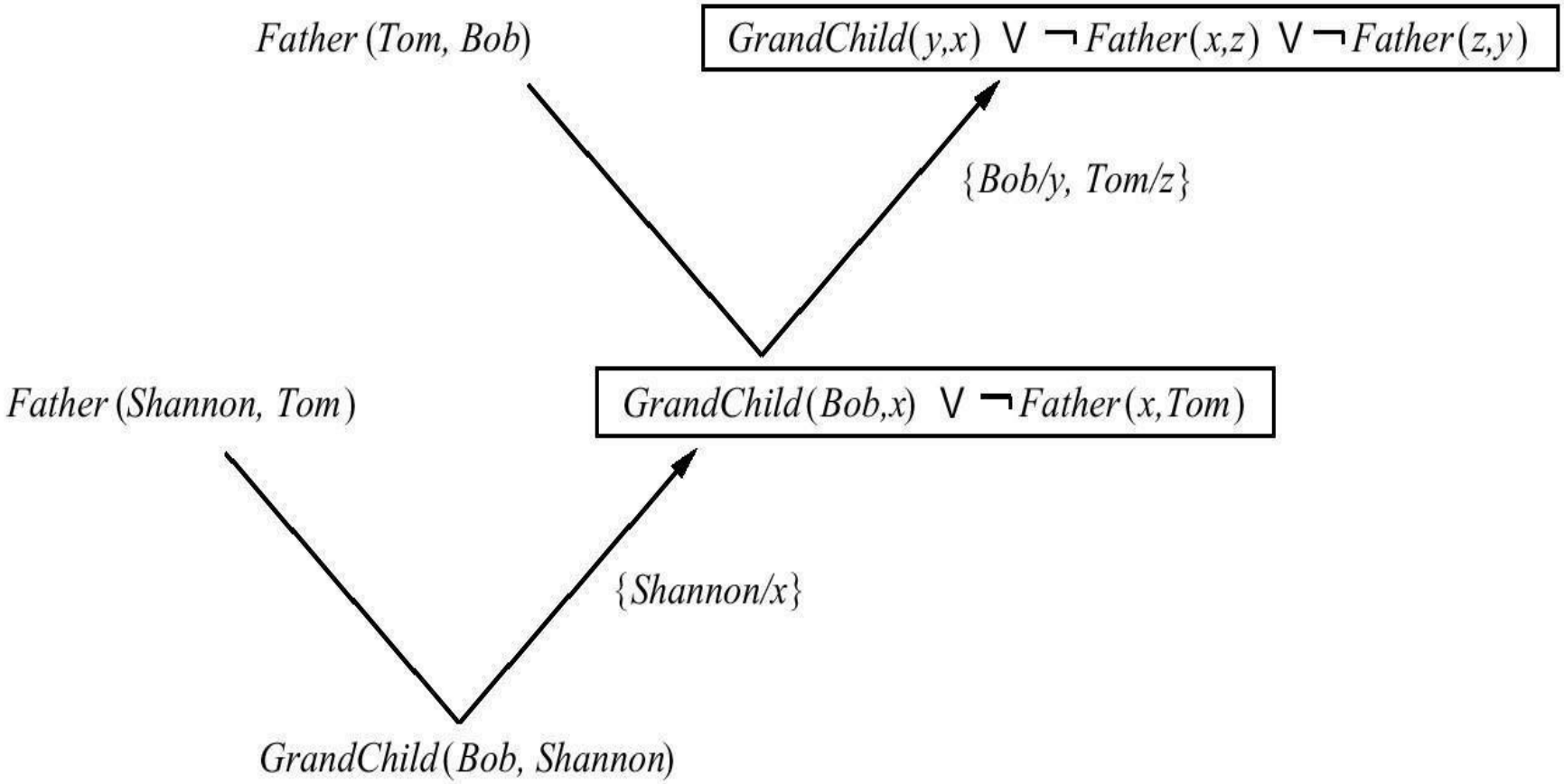
1. Find a literal L_1 from clause C_1, literal L_2 from clause C_2, and substitution θ such that $L_1\theta = \neg L_2\theta$
2. Form the resolvent C by including all literals from $C_1\theta$ and $C_2\theta$, except for $L_1\theta$ and $\neg L_2\theta$. More precisely, the set of literals occurring in the conclusion C is

$$C = (C_1 - \{L_1\})\theta \cup (C_2 - \{L_2\})\theta$$

InvertingFirstorderresolution

$$C_2 = (C - (C_1 - \{L_1\})\theta_1)\theta_2^{-1} \cup \{\neg L_1 \theta_1 \theta_2^{-1}\}$$

Cigol


Father (Tom, Bob)
GrandChild(y,x) ∨ ¬Father(x,z) ∨ ¬Father(z,y)
{Bob/y, Tom/z}
Father (Shannon, Tom)
GrandChild(Bob,x) ∨ ¬Father(x,Tom)
{Shannon/x}
GrandChild(Bob, Shannon)

Progol

Progol

PROGOL: Reduce comb explosion by generating the most specific acceptable h

1. User specifies H by stating predicates, functions, and forms of arguments allowed for each
2. PROGOL uses sequential covering algorithm. For each $\langle x_i, f(x_i)\rangle$
   - Find most specific hypothesis h_i s.t. $B \wedge h_i \wedge x_i \vdash f(x_i)$
     – actually, considers only k-step entailment
3. Conduct general-to-specific search bounded by specific hypothesis h_i, choosing hypothesis with minimum description length

MachineLearning

Chapter13.ReinforcementLearning

TomM.Mitchell

ControlLearning

Considerlearningtochooseactions,e.g.,

- Robotlearningtodockonbatterycharger
- Learningtochooseactionstooptimizefactoryoutput
- LearningtoplayBackgammon

Noteseveralproblemcharacteristics:

- Delayedreward
- Opportunityforactiveexploration
- Possibilitythatstateonlypartiallyobservable
- Possibleneedtolearnmultipletaskswithsamesensors/effectors

OneExample:TD-Gammon

LearntoplayBackgammonI

mmediatereward

- +100ifwin
- -100iflose
- 0forallotherstates

Trainedbyplaying1.5milliongamesagainstitselfNow

approximatelyequaltobesthumanplayer

ReinforcementLearningProblem

$$s_0 \xrightarrow[r_0]{a_0} s_1 \xrightarrow[r_1]{a_1} s_2 \xrightarrow[r_2]{a_2} \dots$$

Goal: Learn to choose actions that maximize

$$r_0 + \gamma r_1 + \gamma^2 r_2 + \dots \text{ , where } 0 \leq \gamma < 1$$

MarkovDecisionProcesses

Assume

- finitesetofstates$S$
- setofactions A
- ateachdiscretetimeagent observesstate$s_t \in S$andchooses action$a_t \in A$
- thenreceivesimmediaterewardr_t
- andstatechangestos$_{t+1}$
- Markov assumption:$s_{t+1}=\delta(s_t,a_t)$and$r_t=r(s_t,a_t)$
  - i.e.,$r_t$ands_{t+1}dependonly on*current*stateandaction
  - functions$\delta$and$r$maybenondeterministic
  - functions$\delta$and$r$notnecessarilyknowntoagent

Agent'sLearningTask

Execute actions in environment, observe results, and

- learn action policy $\pi : S \rightarrow A$ that maximizes

$$E[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots]$$

  from any starting state in S
- here $0 \leq \gamma < 1$ is the discount factor for future rewards

Note something new:

- Target function is $\pi : S \rightarrow A$
- but we have no training examples of form $\langle s, a \rangle$
- training examples are of form $\langle \langle s, a \rangle, r \rangle$

ValueFunction

To begin, consider deterministic worlds...

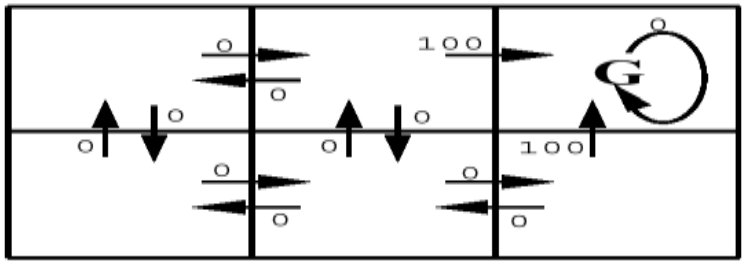
For each possible policy π the agent might adopt, we can define an evaluation function over states

$$\begin{aligned} V^{\pi}(s) &\equiv r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots \\ &\equiv \sum_{i=0}^{\infty} \gamma^i r_{t+i} \end{aligned}$$

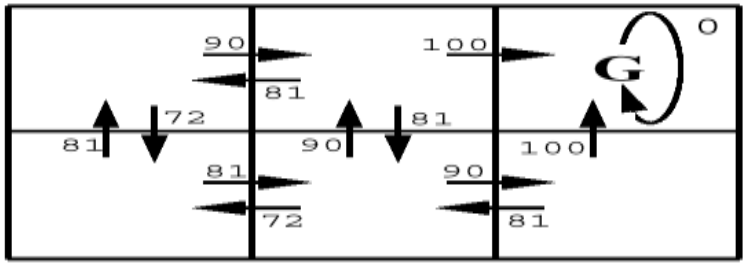
where $r_t, r_{t+1}, \ldots$ are generated by following policy π starting at state s

Restated, the task is to learn the optimal policy π^*

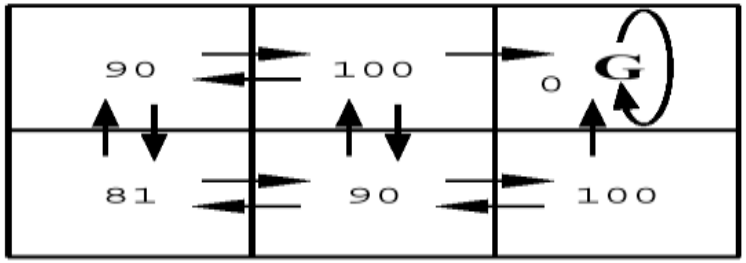
$$\pi^* \equiv \underset{\pi}{\operatorname{argmax}} V^{\pi}(s), (\forall s)$$



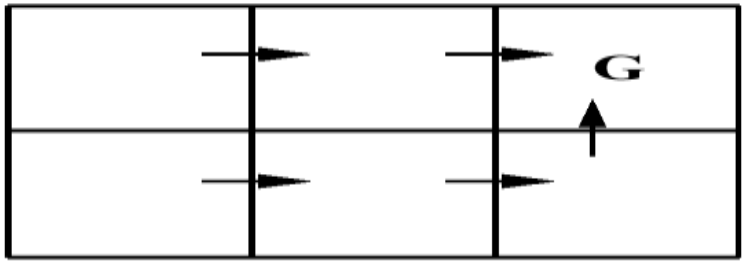
$r(s, a)$ (immediate reward) values



$Q(s, a)$ values



$V^*(s)$ values



One optimal policy

WhattoLearn

We might try to have agent learn the evaluation function V^{π^*} (which we write as V^*)

It could then do a lookahead search to choose best action from any state s because

$$\pi^*(s) = \underset{a}{\text{argmax}}[r(s,a) + \gamma V^*(\delta(s,a))]$$

A problem:

- This works well if agent knows $\delta : S \times A \rightarrow S$, and $r : S \times A \rightarrow \Re$
- But when it doesn't, it can't choose actions this way

QFunction

Define new function very similar to V^*

$$Q(s,a) \equiv r(s,a) + \gamma V^*(\delta(s,a))$$

If agent learns Q, it can choose optimal action even without knowing δ!

$$\pi^*(s) = \operatorname{argmax}_a [r(s,a) + \gamma V^*(\delta(s,a))]$$

$$\pi^*(s) = \operatorname{argmax}_a Q(s,a)$$

Q is the evaluation function the agent will learn

TrainingRuletoLearn$Q$

Note Q and V^* closely related:

$$V^*(s) = \max_{a'} Q(s, a')$$

Which allows us to write Q recursively as

$$\begin{aligned} Q(s_t, a_t) &= r(s_t, a_t) + \gamma V^*(\delta(s_t, a_t))) \\ &= r(s_t, a_t) + \gamma \max_{a'} Q(s_{t+1}, a') \end{aligned}$$

Nice! Let $\hat{Q}$ denote learner's current approximation to Q. Consider training rule

$$\hat{Q}(s, a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s', a')$$

where s' is the state resulting from applying action a in state s

QLearningforDeterministic Worlds

For each s, a initialize table entry $\hat{Q}(s, a) \leftarrow 0$

Observe current state s

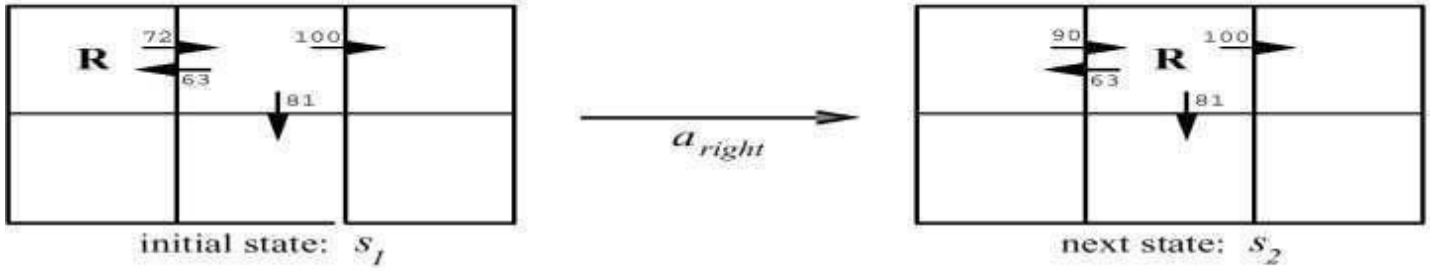
Do forever:

- Select an action a and execute it
- Receive immediate reward r
- Observe the new state s'
- Update the table entry for $\hat{Q}(s, a)$ as follows:

$$\hat{Q}(s, a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s', a')$$

- $s \leftarrow s'$

Updating $\hat{Q}$



$$\begin{aligned}\hat{Q}(s_1, a_{right}) &\leftarrow r + \gamma \max_{a'} \hat{Q}(s_2, a') \\ &\leftarrow 0 + 0.9 \max\{63, 81, 100\} \\ &\leftarrow 90\end{aligned}$$

notice if rewards non-negative, then

$$(\forall s, a, n) \quad \hat{Q}_{n+1}(s, a) \geq \hat{Q}_n(s, a)$$

and

$$(\forall s, a, n) \quad 0 \leq \hat{Q}_n(s, a) \leq Q(s, a)$$

$\hat{Q}$ converges to Q. Consider case of deterministic world where see each $\langle s, a \rangle$ visited infinitely often.

Proof: Define a full interval to be an interval during which each $\langle s, a \rangle$ is visited. During each full interval the largest error in $\hat{Q}$ table is reduced by factor of γ

Let $\hat{Q}_n$ be table after n updates, and Δ_n be the maximum error in $\hat{Q}_n$; that is

$$\Delta_n = \max_{s,a} |\hat{Q}_n(s, a) - Q(s, a)|$$

For any table entry $\hat{Q}_n(s, a)$ updated on iteration $n + 1$, the error in the revised estimate $\hat{Q}_{n+1}(s, a)$ is

$$\begin{aligned}
|\hat{Q}_{n+1}(s, a) - Q(s, a)| &= |(r + \gamma \max_{a'} \hat{Q}_n(s', a')) \\
&\quad -(r + \gamma \max_{a'} Q(s', a'))| \\
&= \gamma |\max_{a'} \hat{Q}_n(s', a') - \max_{a'} Q(s', a')| \\
&\leq \gamma \max_{a'} |\hat{Q}_n(s', a') - Q(s', a')| \\
&\leq \gamma \max_{s'', a'} |\hat{Q}_n(s'', a') - Q(s'', a')| \\
|\hat{Q}_{n+1}(s, a) - Q(s, a)| &\leq \gamma \Delta_n
\end{aligned}$$

NondeterministicCase

What if reward and next state are non-deterministic?

We redefine V, Q by taking expected values

$$V^{\pi}(s) \equiv E[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots]$$
$$\equiv E[\sum_{i=0}^{\infty} \gamma^i r_{t+i}]$$

$$Q(s,a) \equiv E[r(s,a) + \gamma V^*(\delta(s,a))]$$

NondeterministicCase(Cont')

Q learning generalizes to nondeterministic worlds

Alter training rule to

$$\hat{Q}_n(s,a) \leftarrow (1-\alpha_n)\hat{Q}_{n-1}(s,a)+\alpha_n[r+\max_{a'}\hat{Q}_{n-1}(s',a')]$$

where

$$\alpha_n = \frac{1}{1+visits_n(s,a)}$$

Can still prove convergence of $\hat{Q}$ to Q [Watkins and Dayan, 1992]

TemporalDifferenceLearning

Q learning: reduce discrepancy between successive Q estimates

One step time difference:

$$Q^{(1)}(s_t, a_t) \equiv r_t + \gamma \max_a \hat{Q}(s_{t+1}, a)$$

Why not two steps?

$$Q^{(2)}(s_t, a_t) \equiv r_t + \gamma r_{t+1} + \gamma^2 \max_a \hat{Q}(s_{t+2}, a)$$

Or n?

$$Q^{(n)}(s_t, a_t) \equiv r_t + \gamma r_{t+1} + \cdots + \gamma^{(n-1)} r_{t+n-1} + \gamma^n \max_a \hat{Q}(s_{t+n}, a)$$

Blend all of these:

$$Q^{\lambda}(s_t, a_t) \equiv (1-\lambda) \left[Q^{(1)}(s_t, a_t) + \lambda Q^{(2)}(s_t, a_t) + \lambda^2 Q^{(3)}(s_t, a_t) \right.$$

TemporalDifference Learning(Cont')

$$Q^{\lambda}(s_t, a_t) \equiv (1-\lambda)\left[Q^{(1)}(s_t, a_t) + \lambda Q^{(2)}(s_t, a_t) + \lambda^2 Q^{(3)}(s_t, a_t)\right.$$

Equivalent expression:

$$Q^{\lambda}(s_t, a_t) = r_t + \gamma[\ (1-\lambda)\max_a \hat{Q}(s_t, a_t) + \lambda\ Q^{\lambda}(s_{t+1}, a_{t+1})]$$

TD(λ) algorithm uses above training rule

- Sometimes converges faster than Q learning
- converges for learning V^* for any $0 \leq \lambda \leq 1$ (Dayan, 1992)
- Tesauro's TD-Gammon uses this algorithm

SubtletiesandOngoingResearch

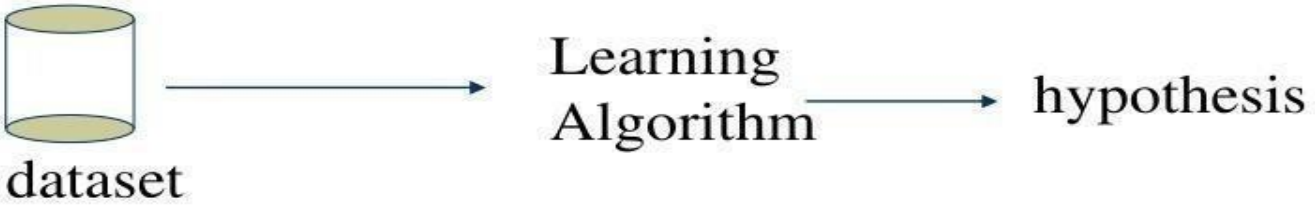
- Replace $\hat{Q}$ table with neural net or other generalizer
- Handle case where state only partially observable
- Design optimal exploration strategies
- Extend to continuous action, state
- Learn and use $\hat{\delta} : S \times A \rightarrow S$
- Relationship to dynamic programming

Analytical Learning

- *Introduction*
- *Learning with Perfect Domain Theories*
- *Explanation-Based Learning*
- Search Control Knowledge
- Summary

Introduction

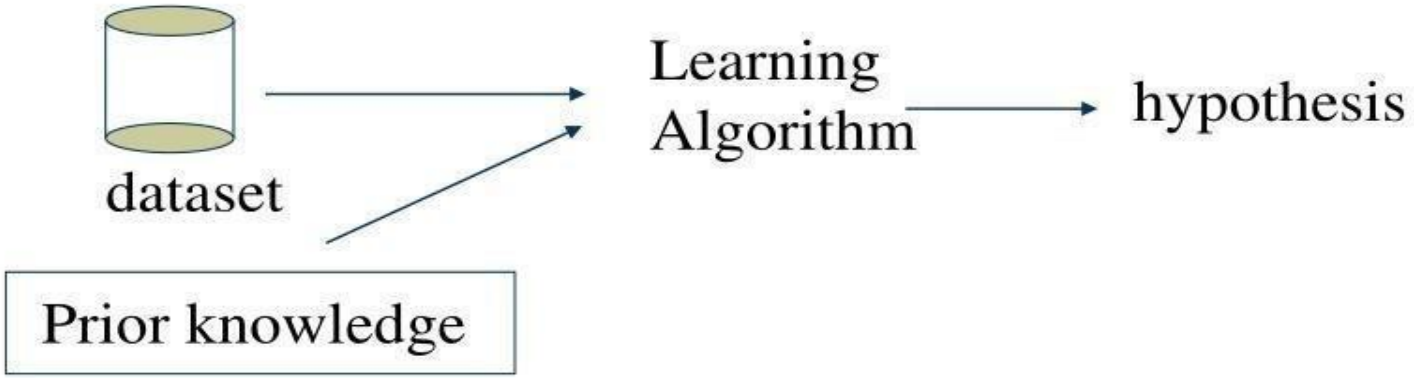
Learning algorithms like neural networks, decision trees, inductive logic programming, etc. all require a good number of examples to be able to do good predictions.



Dataset must be sufficiently large

Introduction

We don't need a large dataset if besides taking examples as input, the learning algorithm can take prior knowledge.



Dataset does not need to be large

Introduction

Explanation-based learning uses prior knowledge to reduce the size of the hypothesis space.

It analyzes each example to infer which features are relevant and which ones are irrelevant.

Example

Learning to Play Chess

Suppose we want to learn a concept like "what is a board position in which black will lose the queen in x moves?".

Chess is a complex game. Each piece can occupy many positions. We would need many examples to learn this concept.

But humans can learn these type of concepts with very few examples. Why?

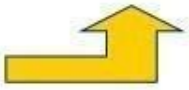
Example

Humans can analyze an example and use prior knowledge related to legal moves.

From there it can generalize with only few examples.

Example: why is this a positive example?

"Because the white king is attacking both king and queen; black must avoid check, letting white capture the queen"

reasoning 

Example

What is the prior knowledge involved in playing chess?

It is knowledge about the rules of chess:

- Legal moves for the knight and other pieces.
- Players alternate moves in games.
- To win you must capture the opponent's king.

Inductive and Analytical Learning

Inductive Learning

Input: HS, D,
Output: hypothesis h
h is consistent with D

Analytical Learning

Input: HS, D, B
Output: h
h is consistent with D and B
(~h -/- B)

HS: Hypothesis Space
D: Training Set
B: Background knowledge

Example Analytical Learning

Input:

- Dataset where each instance is a pair of objects represented by the following predicates: Color, Volume, Owner, Material, Density, On. Example:

On(Obj1,Obj2)
Type(Obj1,Box)
Type(Obj2,Endtable)
Color(Obj1,Red)
Color(Obj2,Blue)
Volume(Obj1,2)
Owner(Obj1,Fred)
Owner(Obj2,Louise)
Density(Obj1,0.3)
Material(Obj1,Cardboard)
Material(Obj2,Wood)

Example Analytical Learning

Hypothesis space: set of Horn clause rules.
The head of each rule has the predicate SafeToStack.
Example:

SafeToStack(x,y) ← Volume(x,vx) ^ Volume(y,vy)
^ LassThan(vx,vy)

Domain Theory:

SafeToStack(x,y) ← ~Fragile(y)
SafeToStack(x,y) ← Lighter(x,y)
Lighter(x,y) ← Weight(x,wx) ^ Weight(y,wy) ^ LessThan(wx,wy)

Example Analytical Learning

Domain Theory:

SafeToStack(x,y) ← ~Fragile(y)
SafeToStack(x,y) ← Lighter(x,y)
Lighter(x,y) ← Weight(x,wx) ^ Weight(y,wy) ^ LessThan(wx,wy)
…
Fragile(x) ← Material(x,Glass)

Note:

- The domain theory refers to predicates not contained in the examples.
- The domain theory is sufficient to prove the example is true.

Perfect Domain Theories

A domain theory is **correct** if each statement is true.

A domain theory is **complete** if it covers every positive example of the instance space (w.r.t a target concept and instance space).

A **perfect** domain theory is correct and complete.

Perfect Domain Theories

Examples of where to find perfect domain theories:
Rules of chess

Examples of where not to find perfect domain theories:
SafetoStack problem

We will look into learning problems with perfect domain theories only.

Explanation Based Learning Algorithm

We consider an algorithm that has the following properties:

- ✓ It is a sequential covering algorithm considering the data incrementally
- ✓ For each positive example not covered by the current rules it forms a new rule by:
  - o Explaining the new positive example.
  - o Analyzing the explanation to find a generalization.
  - o Refining the current hypothesis by adding a new Horn Clause rule to cover the example.

Explanation Based Learning Algorithm

Prolog-EBG (Kedar-Cabelli and McCarty 87).

1. LearnedRules ← { }
2. Pos ← Positive examples from training examples
3. For each positive example X in Pos not covered by LearnedRules do
   a. Explain
      Use the domain theory to explain that X satisfies the target.

Explanation Based Learning Algorithm

Prolog-EBG (Kedar-Cabelli and McCarty 87).

b. Analyze
   Find the most general set of features of X sufficient to satisfy the target according to the explanation.
c. Refine
   LearnedRules += NewHornClause
   NewHornClause: Target ← sufficient features

4. Return LearnedRules

Explaining the Example

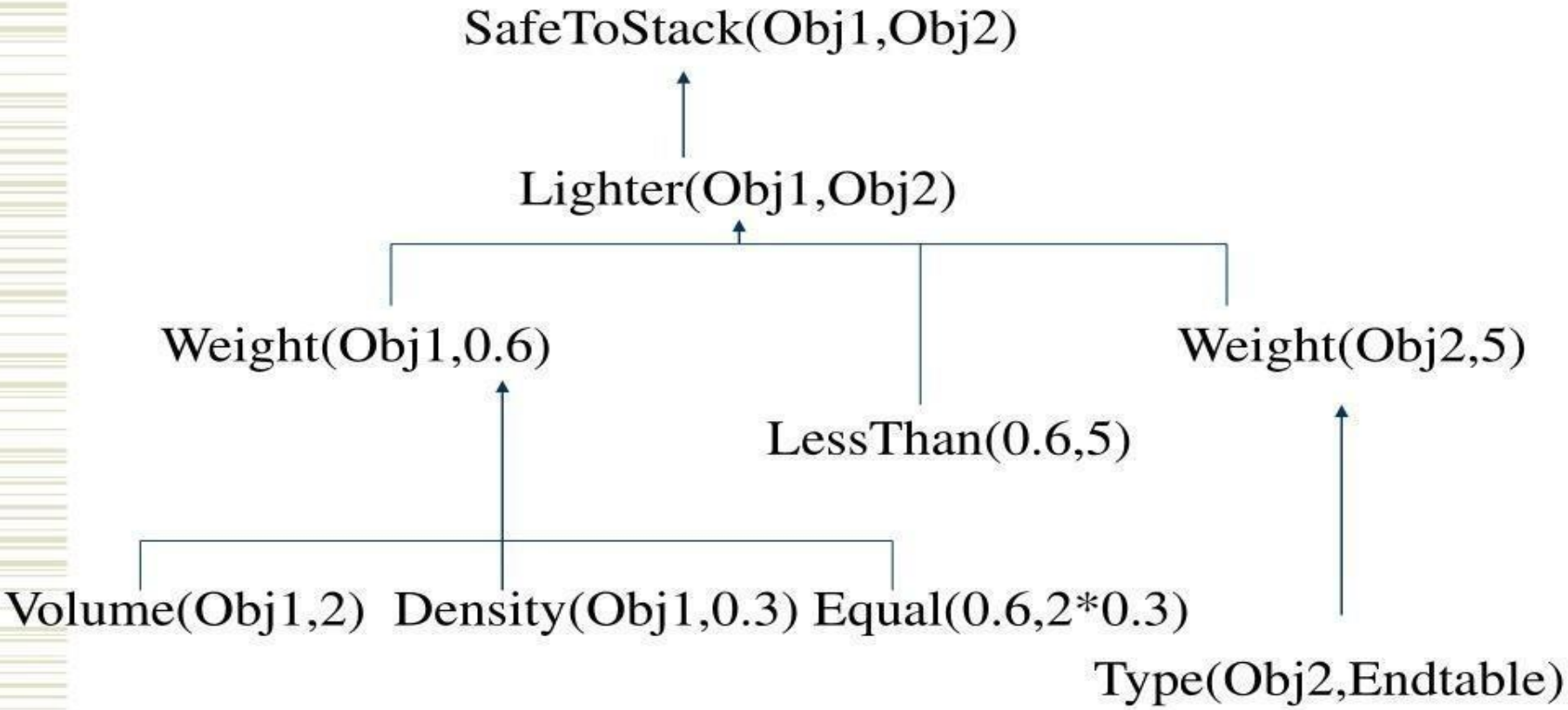
1. For each positive example X in Pos not covered by LearnedRules do
   a. *Explain*
      Use the domain theory to explain that X satisfies the target concept.

Explaining the Example

The explanation is a proof that the example belongs to the target (if the theory is perfect):

On(Obj1,Obj2)
Type(Obj1,Box)
Type(Obj2,Endtable)
Color(Obj1,Red)
Color(Obj2,Blue)
Volume(Obj1,2)
Owner(Obj1,Fred)
Owner(Obj2,Louise)
Density(Obj1,0.3)
Material(Obj1,Cardboard)
Material(Obj2,Wood)

Explanation



Explanation

Considerations:

- There might be more than one explanation to the example. In that case one or all explanations may be used.
- An explanation is obtained using a backward chaining search as is done by Prolog. Prolog-EBG stops when it finds the first proof.

Analyze

Many features appear in an example.
Of them, how many are truly relevant?

We consider as relevant those features that show in the explanation.

Example:
Relevant feature: Density
Irrelevant feature: Owner

Analyze

Taking the leaves of the explanation and substituting variables x and y for Obj1, and Obj2:

SafeToStack(x,y) ← Volume(x,2) ^ Density(x,0.3)
^ Type(y,Endtable)

Analyze

Considerations:

- We omit features independent of x and y such as Equal(0.6,times(2,0.3)) and LessThan(0.6,5).
- The rule is now more general and can serve to explain other instances matching the rule.
- A more general form of generalization called regression finds the most general rule explaining the example.

Refine

- The current hypothesis is the set of Horn clauses that we have constructed up to this point.
- Using sequential covering we keep adding more rules, thus refining our hypothesis.
- A new instance is negative if it is not covered by any rule.

Remarks

- Explanation Based Learning (EBL) justifies the hypothesis by using prior knowledge.
- The explanation or proof shows which features are relevant.
- Each Horn clause is a sufficient condition for satisfying the target concept.

Remarks

- The success of the method depends on how the domain theory is formulated.
- We assumed the theory is correct and complete.
  If this is not the case the learned concept may be incorrect.

Analytical Learning

- Introduction
- Learning with Perfect Domain Theories
- *Explanation-Based Learning*
- *Search Control Knowledge*
- *Summary*

Discovering New Features

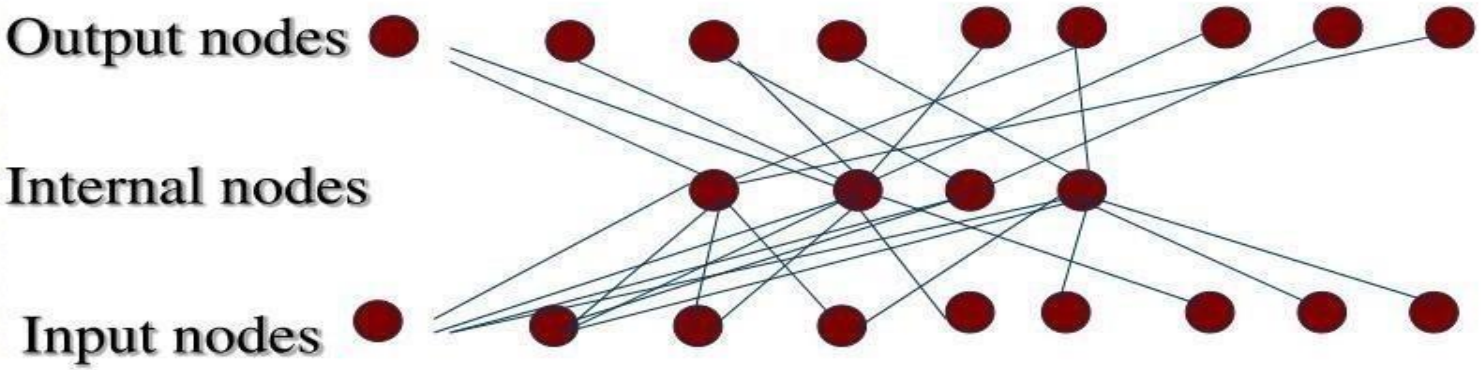
The Prolog-EBG system we described before can formulate new features that do not show up in the examples.

Example: Volume * Density > 5
(derived from the domain theory)

This is different from neural networks using hidden nodes. why?

Discovering New Features

This is different from neural networks using hidden nodes. why?



Inductive Bias in Explanation-Based Learning

What is the inductive bias of explanation based learning?

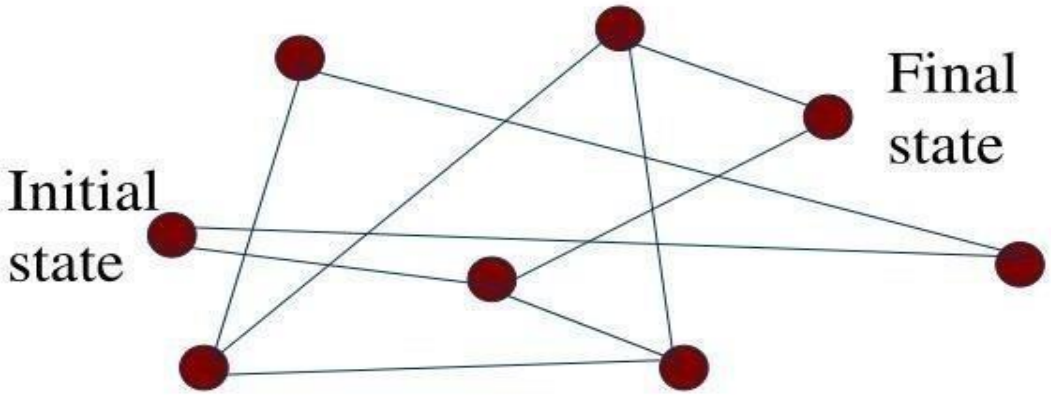
The hypothesis h follows deductively from D and B

D: database B: Background knowledge

Bias: Prefer small sets of maximally general Horn Clauses

Search Control Knowledge

Problem: learning to speed up search programs.
This is also called "speedup learning"



How do we improve our search control strategy to find a solution quickly?

Search Control Knowledge

Examples include: playing chess scheduling and optimization problems.

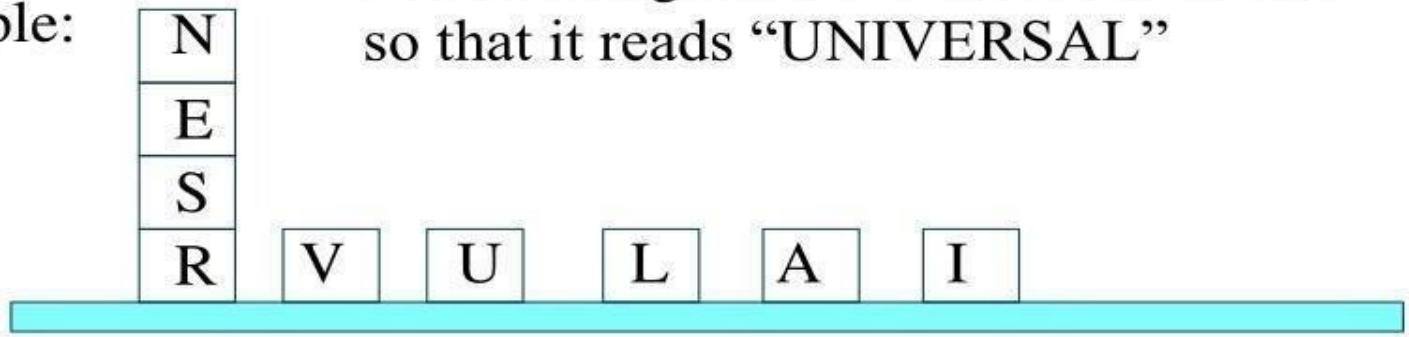
Problem formulation:

S: set of possible search states
O: set of legal operators (transform one state into another state)
G: predicate over S indicating the goal states

Search Control Knowledge

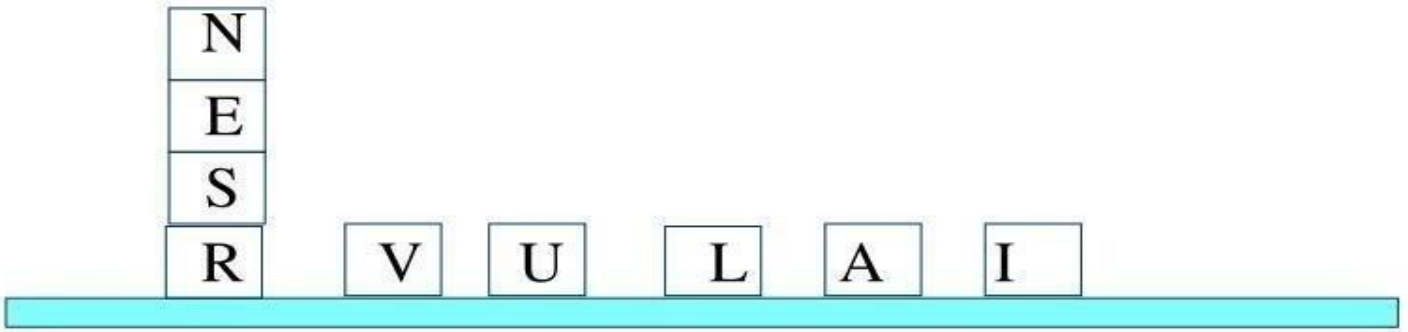
Example:

Learn an algorithm to stack the blocks so that it reads "UNIVERSAL"



Search Control Knowledge

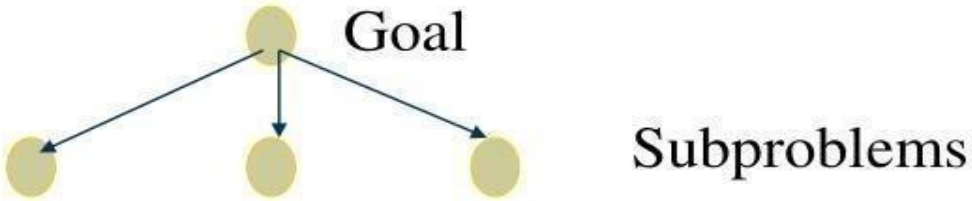
S: all possible configurations of blocks on the table
O: {(MS x) move block x to stack if x is on table,
(MT x) move block x to table if x is on the stack}
G: G(si) = true if si is the configuration where the blocks read UNIVERSAL



Prodigy

Prodigy is a planning system.
Input: state space S and operators O.
Output: A sequence of operators that lead from the initial state to the final state.

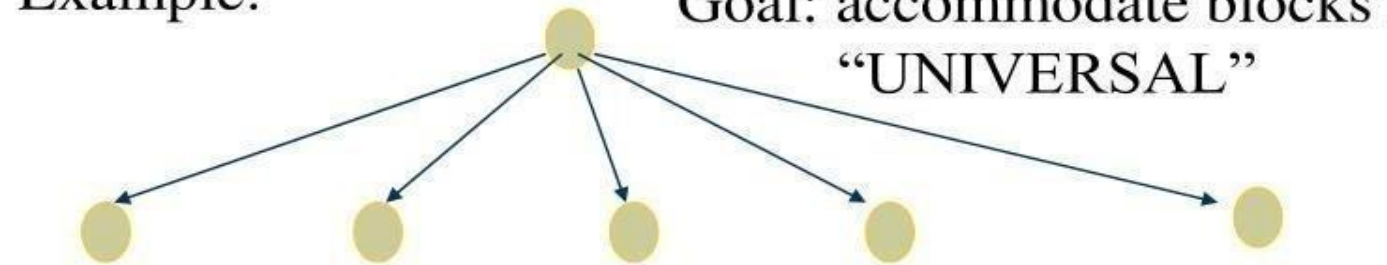
Prodigy uses a **means-end planner**: we decompose goals into subgoals:



Prodigy

Example:

Goal: accommodate blocks to read "UNIVERSAL"



Question: what subgoal should be attacked first?

Answer is given by *search control knowledge*

Prodigy and Explanation Based Learning

Prodigy defines a set of target concepts to learn, e.g., which operator given the current state takes you to the goal state?

An example of a rule learned by Prodigy in the block-stacking problem is:

IF One subgoal to be solved is On(x,y) AND
One subgoal to be solved is On(y,z)
THEN Solve the subgoal On(y,z) before On(x,y)

Prodigy and Explanation Based Learning

The rationale behind the rule is that it would avoid a conflict when stacking blocks.

Prodigy learns by first encountering a conflict, then explaining the reason for the conflict and creating a rule like the one above.

Experiments show an improvement in efficiency by a factor of two to four.

Problems with EBL

- ✓ The number of control rules that must be learned is very large.
- ✓ If the control rules are many, much time will be spent looking for the best rule.

Utility analysis is used to determine what rules to keep and what rules to forget.

Prodigy:

328 possible rules ⟶ 69 pass test ⟶ 19 were retained

Problems with EBL

- ✓ Another problem with EBL is that it is sometimes intractable to create an explanation for the target concept.

  For example, in chess, learning a concept like:
  "states for which operator A leads to a solution"
  The search here grows exponentially.

Summary

- Different from inductive learning, analytical learning looks for a hypothesis that fit the background knowledge and covers the training examples.

- Explanation based learning is one kind of analytical learning that divides into three steps:
  a. Explain the target value for the current example
  b. Analyze the explanation (generalize)
  c. Refine the hypothesis

Summary

- Prolog-EBG constructs intermediate features after analyzing examples.
- Explanation based learning can be used to find search control rules.
- In all cases we depend on a perfect domain theory.

Machine Learning

Chapter 12. Combining Inductive and Analytical Learning

Tom M. Mitchell

Inductive and Analytical Learning

Inductive learning

- Hypothesis fits data
- Statistical inference
- Requires little prior knowledge
- Syntactic inductive bias

Analytical learning

- Hypothesis fits domain the
- Deductive inference
- Learns from scarce data
- Bias is domain theory

What We Would Like

Inductive learning ⟵ Analytical learning

Plentiful data
No prior knowledge

Perfect prior knowledge
Scarce data

General purpose learning method:

- No domain theory → learn as well as inductive methods
- Perfect domain theory → learn as well as Prolog-EBG
- Accomodate arbitrary and unknown errors in domain theory
- Accomodate arbitrary and unknown errors in training data

Domain theory:

Cup ← Stable, Liftable, Open Vessel

Stable ← BottomIsFlat

Liftable ← Graspable, Light

Graspable ← HasHandle

Open Vessel ← HasConcavity, ConcavityPointsUp

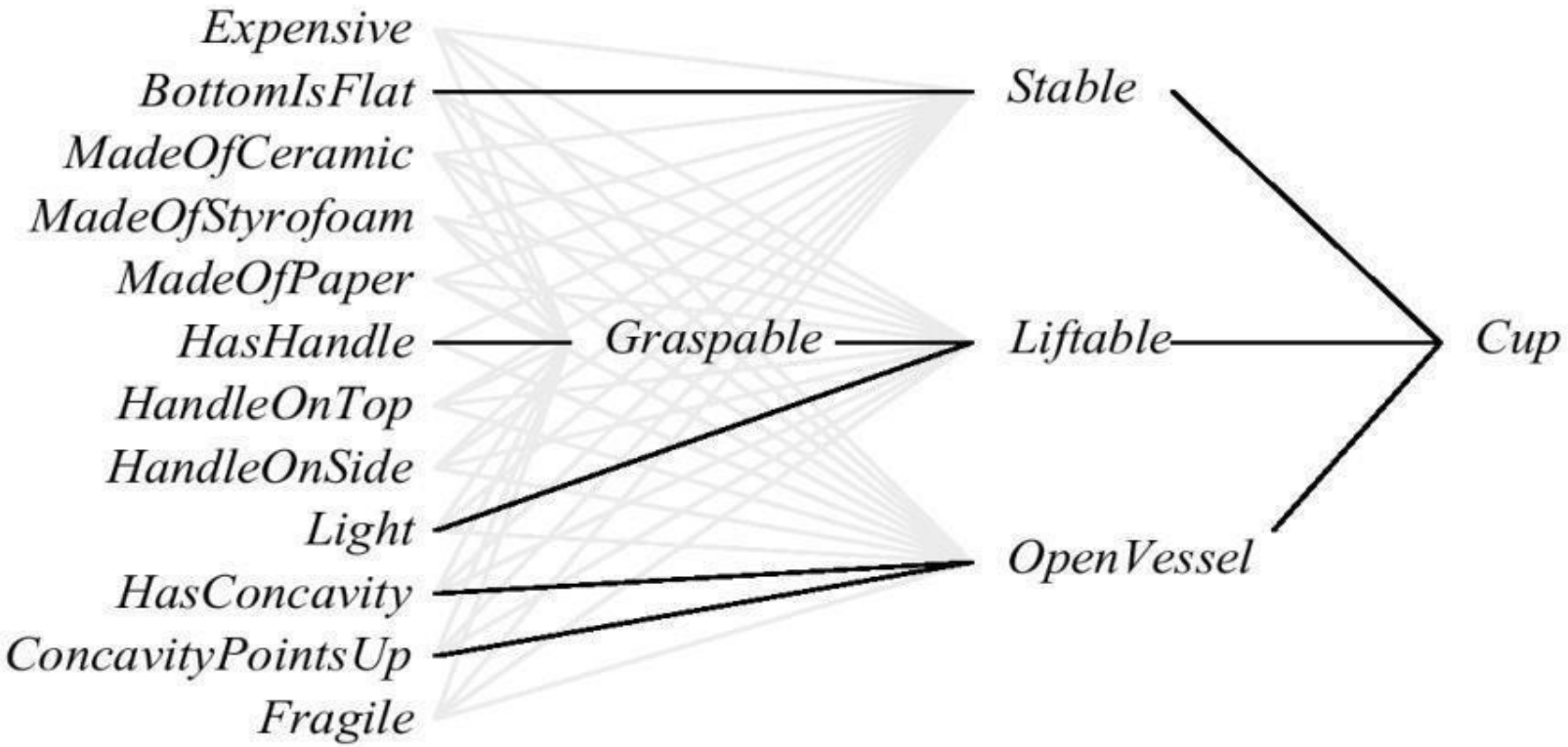
Training examples:

	Cups				Non-Cups					
BottomIsFlat	✓	✓	✓	✓	✓	✓	✓			✓
ConcavityPoints Up	✓	✓	✓	✓	✓		✓	✓		
Expensive	✓		✓				✓		✓	
Fragile	✓	✓			✓	✓		✓		✓
HandleOnTop					✓		✓			
HandleOnSide	✓			✓					✓	
HasConcavity	✓	✓	✓	✓	✓		✓	✓	✓	✓
HasHandle	✓			✓	✓		✓		✓	
Light	✓	✓	✓	✓	✓	✓	✓		✓	
MadeOfCeramic	✓				✓		✓	✓		
MadeOfPaper				✓					✓	
MadeOfStyrofoam		✓	✓			✓				✓

KBANN

- KBANN (data D, domain theory B)
  1. Create a feedforward network h equivalent to B
  2. Use BACKPROP to tune h to t D

Neural Net Equivalent to Domain Theory



Creating Network Equivalent to Domain Theory

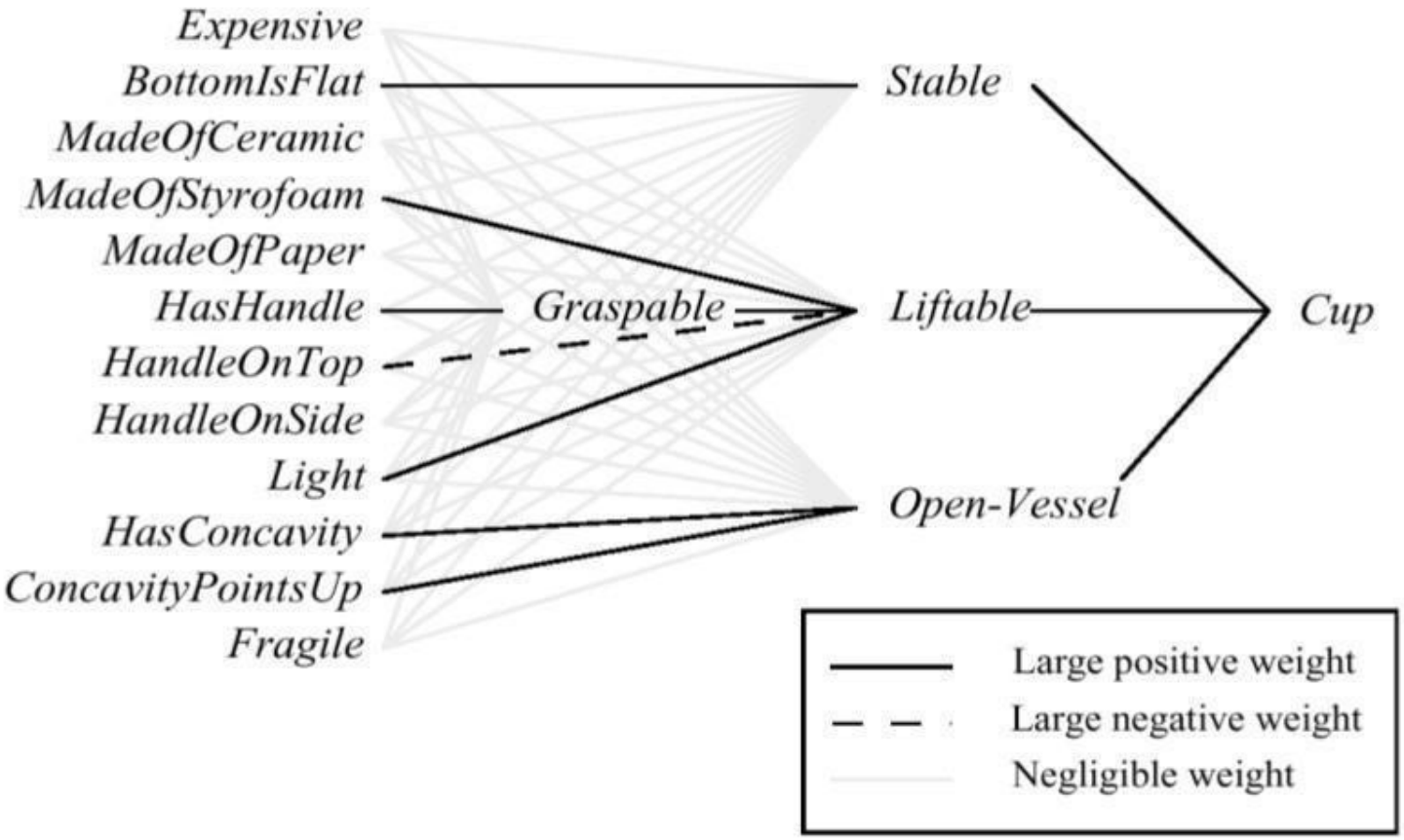
Create one unit per horn clause rule (i.e., an AND unit)

- Connect unit inputs to corresponding clause antecedents
- For each non-negated antecedent, corresponding input weight $w \leftarrow W$, where W is some constant
- For each negated antecedent, input weight $w \leftarrow -W$
- Threshold weight $w_0 \leftarrow -(n-.5)W$, where n is number of non-negated antecedents

Finally, add many additional connections with near-zero weights

$$Liftable \leftarrow Graspable, \neg Heavy$$

Result of refining the network



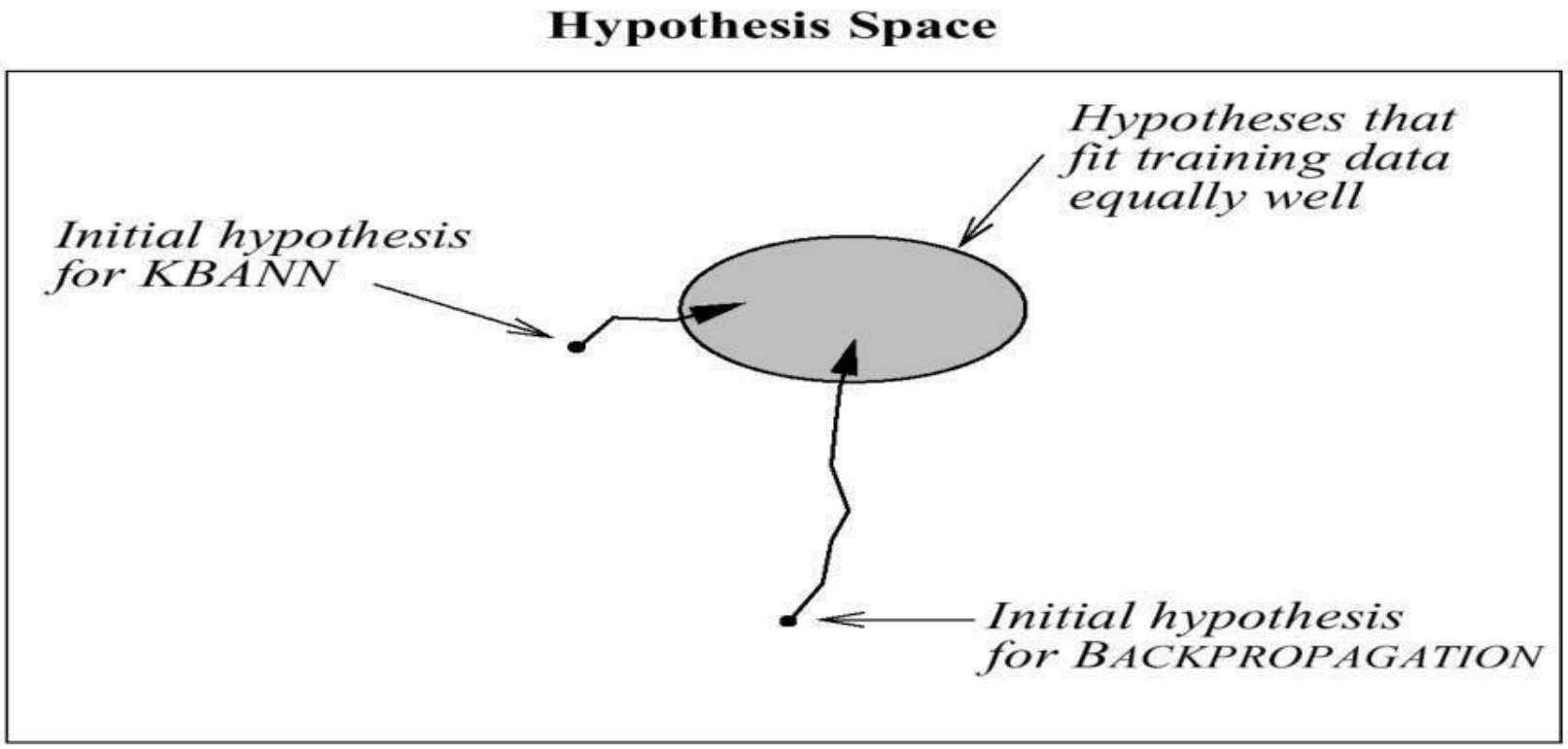
KBANN Results

Classifying promoter regions in DNA leave one out testing:

- Backpropagation : error rate 8/106
- KBANN: 4/106

Similar improvements on other classification, control tasks.

Hypothesis space search in KBANN

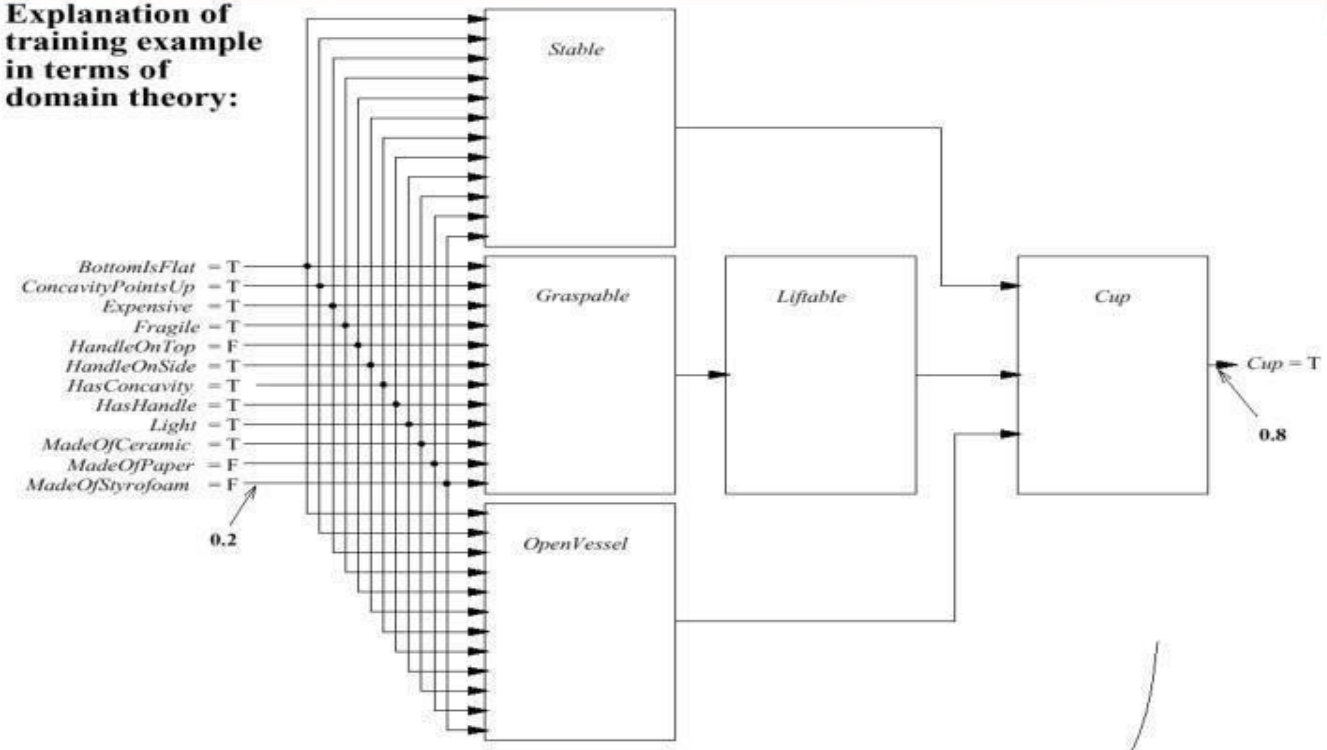


EBNN

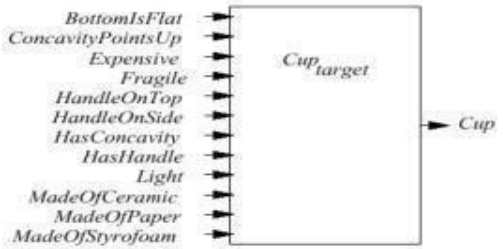
Key idea:

- Previously learned approximate domain theory
- Domain theory represented by collection of neural networks
- Learn target function as another neural network

Explanation of training example in terms of domain theory:



Target network:



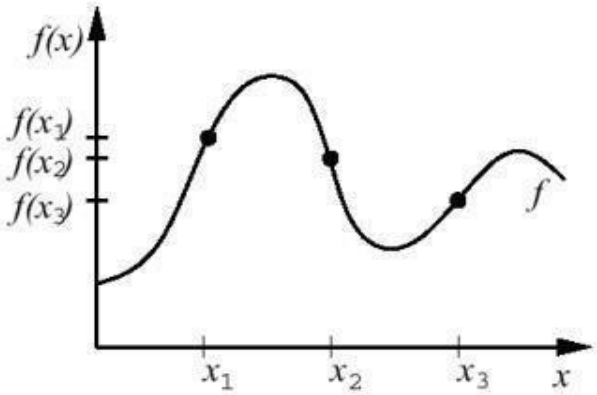
Modified Objective for Gradient Descent

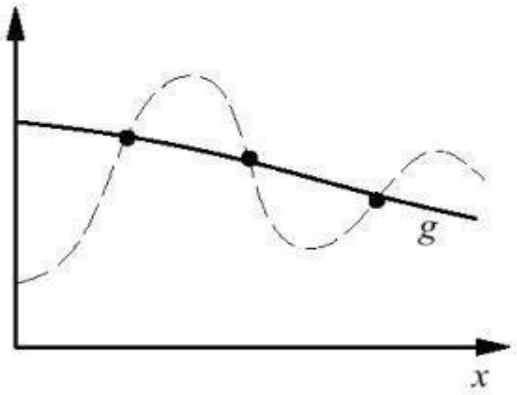
$$E = \sum_i \left[(f(x_i) - \hat{f}(x_i))^2 + \mu_i \sum_j \left(\frac{\partial A(x)}{\partial x^j} - \frac{\partial \hat{f}(x)}{\partial x^j} \right)^2_{(x=x_i)} \right]$$

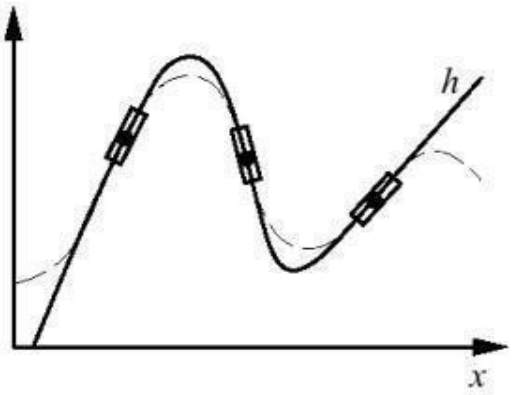
where

$$\mu_i \equiv 1 - \frac{|A(x_i) - f(x_i)|}{c}$$

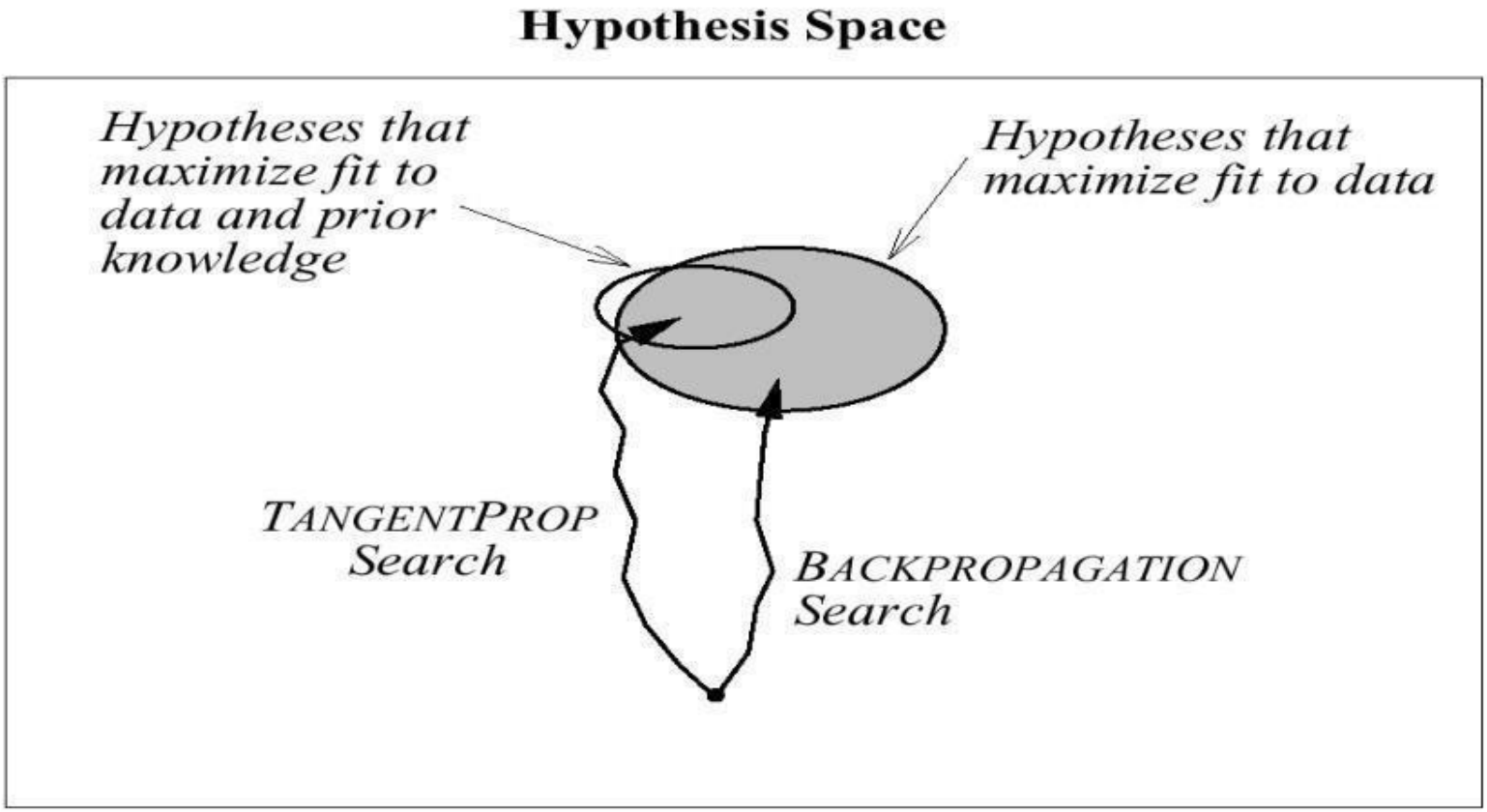
- $f(x)$ is target function
- $\hat{f}(x)$ is neural net approximation to $f(x)$
- $A(x)$ is domain theory approximation to $f(x)$


f(x)
f(x1)
f(x2)
f(x3)
f
x1
x2
x3
x

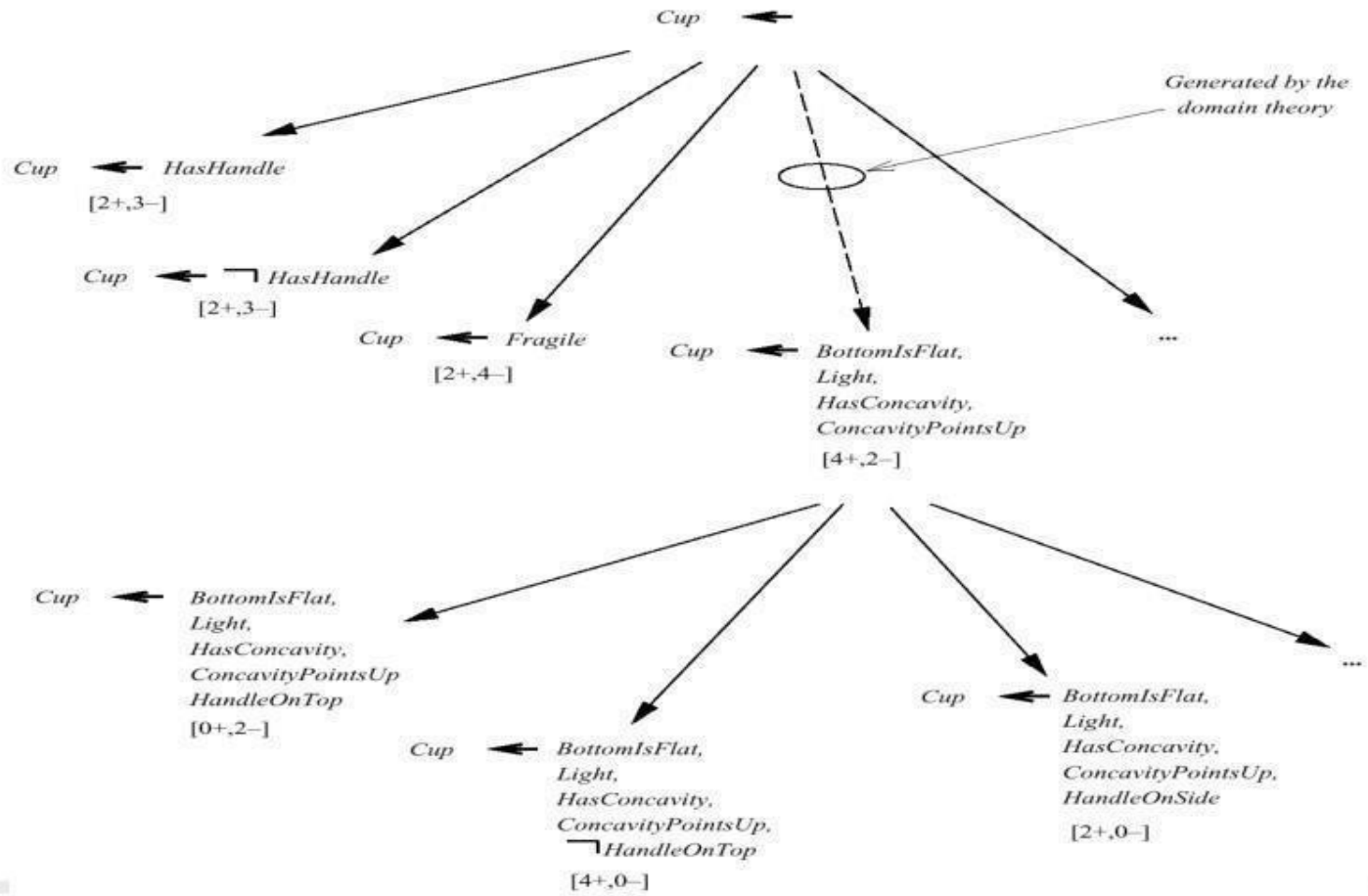

g
x


h
x

Hypothesis Space Search in EBNN



Search in FOCL



FOCL Results

Recognizing legal chess endgame positions:

- 30 positive, 30 negative examples
- FOIL : 86%
- FOCL : 94% (using domain theory with 76% accuracy)

NYNEX telephone network diagnosis

- 500 training examples
- FOIL : 90%
- FOCL : 98% (using domain theory with 95% accuracy)