Machine Learning

CSDLO6021

Subject Incharge

Anuradha Srinivasaraghavan

Associate Professor

Room No. 401

email: g.anuradha@sfit.ac.in

Module I

Lecture 1

Introduction to Machine Learning

Contents

- What is Machine Learning?
- Key Terminology
- Types of Machine Learning
- Issues in Machine Learning
- Application of Machine Learning
- How to choose the right algorithm
- Steps in developing a Machine Learning Application

What is Machine Learning?

- Machine learning is turning data into information.
- It is an intersection of computer science, engineering and statistics
- It can be applied to many fields from politics to geosciences : Any field that needs to interpret and act on data
- Machine learning uses Statistics :
  - Statistics : to lie about how great their products are.

Why do we need statistics?

- Engineering is applying science to solve a problem.
- We solve deterministic problem with finite solution.
- Many problems have solution that isn't deterministic.
- E.g. How to maximize the happiness of the crowd?
  - It is difficult to define what makes everyone happy
  - The definition of happiness is too complex to model.
  - For these problems we need to use some tools from statistics.

Differences

- ML and Data Mining
  - DL came into existence since 1936 by Alan Turing
  - Data mining is often used *by* machine learning to see the connections between relationships(UBER)
  - Machine learning embodies the principles of DL, but can also make automatic correlations/ learn from them to apply to new algorithms (Self driving cars, recommendation systems)

Differences

- ML and Data mining
  - DM pulls from existing information to look for emerging patterns that can help shape decision-making processes. (Brands of retail sales)
  - DM is just the information source of ML
  - ML can learn from the existing data and provide foundation for machine to teach itself (Prediction of cardiovascular diseases)

Types of variables

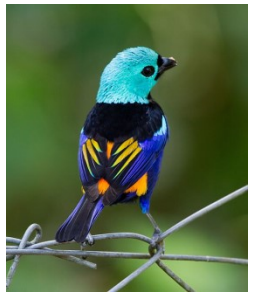
- Categorical: Values that can be organized in categories
- Quantitative Variables - with numerical values for which arithmetic operation
- Ordinal Variables - with natural order

Key Terminologies

Expert System

- It is a computer system that emulates the decision-making ability of a human expert.
- Example : Bird classification system
- Here we are replacing an ornithologist with a computer

Features



- Four parts of birds are
  - weight,
  - wingspan,
  - whether it has webbed feet
  - Color of its back.
- In reality, it is common practice to measure anything that can be measured and sort out the important parts.
- These four things are called **features (attributes).**

Example

	Weight (g)	Wingspan (cm)	Webbed feet?	Back color	Species
1	1000.1	125.0	No	Brown	Buteo jamaicensis
2	3000.7	200.0	No	Gray	Sagittarius serpentarius
3	3300.0	220.3	No	Gray	Sagittarius serpentarius
4	4100.0	136.0	Yes	Black	Gavia immer
5	3.0	11.0	No	Green	Calothorax lucifer
6	570.0	75.0	No	Black	Campephilus principalis

Example

- The first two features are numeric and can take on decimal values.
- The third feature (webbed feet) is binary: it can only be 1 or 0.
- The fourth feature (back color) is an enumeration over the color palette.

Example

- Task : Classification.
- To classify we need to train the algorithm with t**raining set.**
- A training set is the set of training examples used to train ML algorithms
- E.g. training set has six training examples.
- Each **training example** has four features and one target variable (species).
- In classification, target variables are called classes.

Features

- Features or attributes are the individual measurements that, when combined with other features, make up a training example.
- This is usually columns in a training or test set

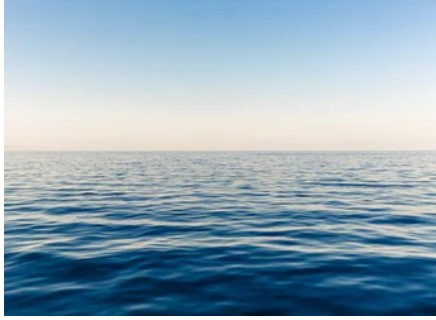
Classification

- The dataset is separated into a training set and a **test set**.
- Initially the program is fed the training examples for machine to learn.
- Next, the test set is fed to the program and the target variable is compared to the predicted value to check how accurate the algorithm is.

Machine Learning Methods

- ML tasks are generally classified into two broad categories.
- **Supervised learning :** trains algorithms based on example input and output data that is labeled by humans
- **Unsupervised learning :** provides algorithm with no labeled data to find structure within its input data.

Supervised Learning



- An algorithm may be fed data with images of sharks labeled as fish and images of oceans labeled as water.
- By being trained on this data, the supervised learning algorithm should be able to later identify unlabeled shark images as fish and unlabeled ocean images as water.

Unsupervised Learning

- Find commonalities among its input data.
- As unlabeled data are more abundant than labeled data, these methods are particularly valuable.
- **Goal :** To discover hidden patterns within a dataset.

Example

- Women of a certain age range who buy unscented soaps are likely to be pregnant and can be targeted to campaign for pregnancy and baby products in order to increase their number of purchases.

Quiz

- **A feature F1 can take certain value: A, B, C, D, E, & F and represents grade of students from a college.**
- **1) Which of the following statement is true in following case?**
- A) Feature F1 is an example of nominal variable.
  B) Feature F1 is an example of ordinal variable.
  C) It doesn't belong to any of the above category.
  D) Both of these

Module I

Lecture 2

Types of Machine Learning

Types of Machine Learning

1. Learning with Associations
2. Classification
3. Regression
4. Unsupervised Learning
5. Reinforced Learning

Learning with Associations

- It is also called market-basket analysis
- Finds associations between products bought by customers together:
- If people who buy X , also buy Y, then these two items could be sold together.

Learning with Associations

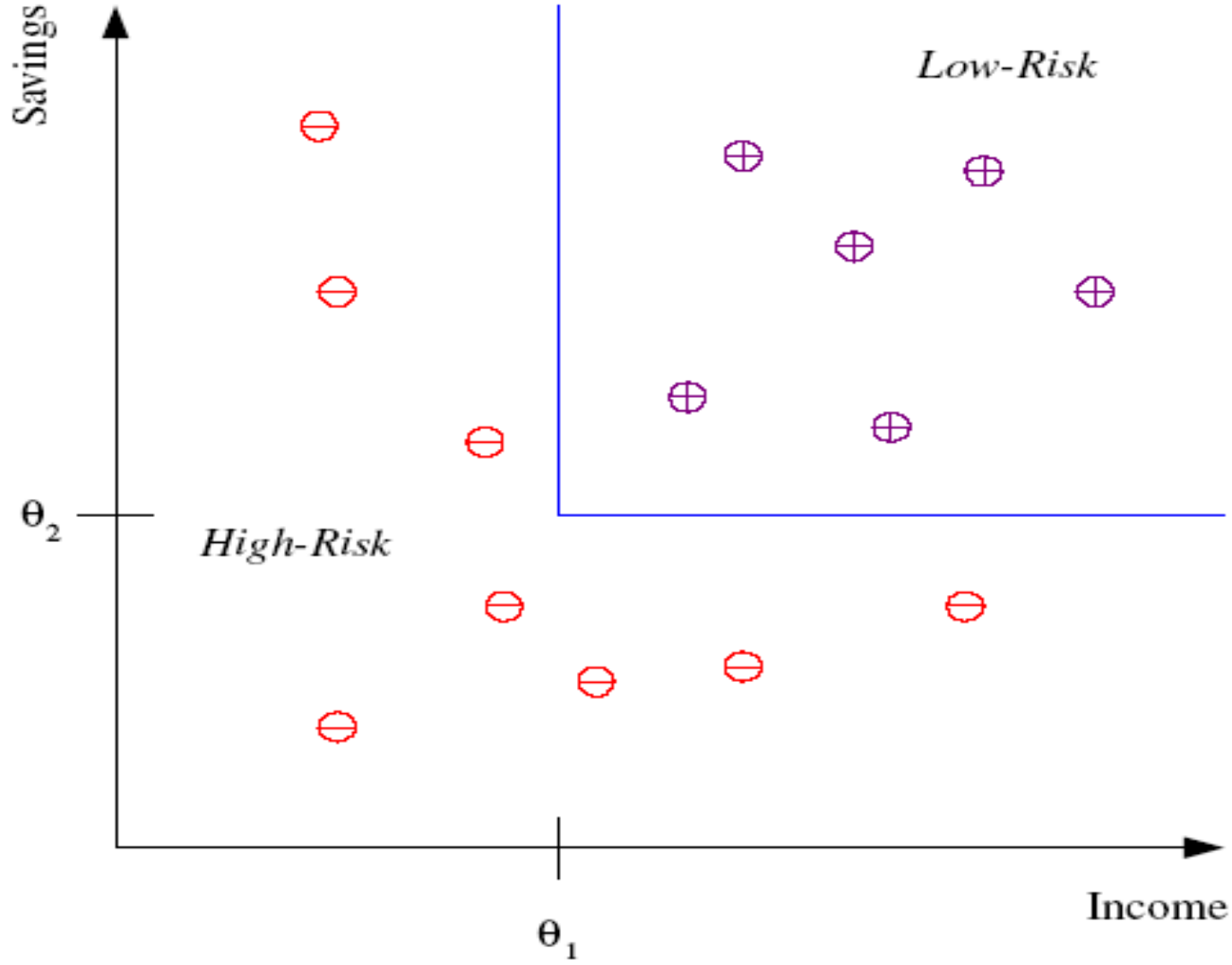
- If butter is always sold with bread we can find the confidence of this sales and define it by a rule as given below.
- “70 percent of customers who buy bread also buy butter.” **–Confidence of the rule**

Learning with Associations

- In the case of a Web portal, from the items correspond to links to Web pages, we can estimate the links a user is likely to click and use this information to download such pages in advance for faster access.

Classification Example : Credit Scoring

- Bank calculates risk based on amount of credit and customer information like income, savings, collaterals, profession, age, past financial history, and so on.
- ML System fits a model using past data to decide whether to accept or reject a new application
- 2 classes: low risk and high risk
- Input : Customer information
- Classifier : assign class : high risk/low risk.
- After training, classification rule learned may be of the form:
  - If income > Θ_1 and savings > Θ_2 then low risk else high risk.


Savings
Low-Risk
θ2
High-Risk
θ1
Income

Knowledge Extraction

- Learning a rule from data is called knowledge extraction.
- The rule is a simple model that explains the data
- Learning performs compression in that by fitting a rule, we get an explanation simpler than the data.

Applications of Classification

1. Prediction :

- Rule that fits the past data, can make predictions
- For new application with income and savings, we predict risk (low/high)

2. Pattern Recognition (Examples)

- Optical Character Recognition
- Face Recognition
- Medical Diagnosis
- Speech Recognition

3. Outlier Detection

- find instances that do not obey the general rule (exceptions)
- also called novelty detection
- Application : Fraud detection.

Types of Machine Learning

1. Learning with Associations
2. Classification
3. **Regression**
4. Unsupervised Learning
5. Reinforced Learning

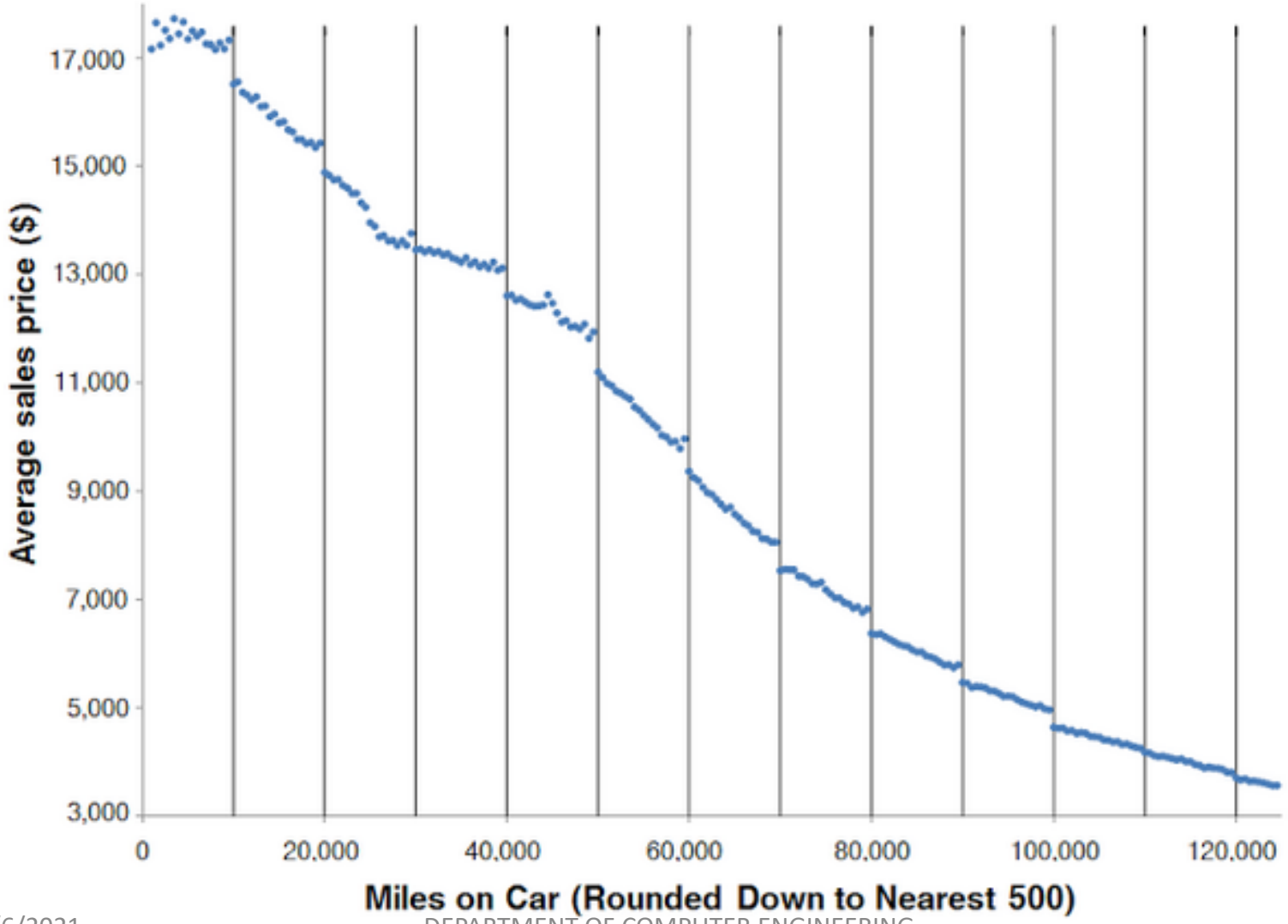
Regression

- System: predict the price of a used car
- Inputs : brand, year, engine capacity, mileage, etc.
- Output : price of the car
- Such problems where output is a number are regression problems.
- Let X denote the car attributes and Y be the price of the car.
- Using past transactions, machine learning program fits a function to this data to learn Y as a function of X.
- The fitted function is of the form

  $$y = ax + b$$

- for suitable values of a and b.

Price vs Miles on car



Unsupervised Learning

- In unsupervised learning, we only have input data and the aim is to find the regularities in the input.
- In the input space, we want to see
  - what generally happens
  - what does not.

  This is called density estimation.
- Clustering : one method for density estimation.
- Here the aim is to find clusters or groupings of input.

Clustering

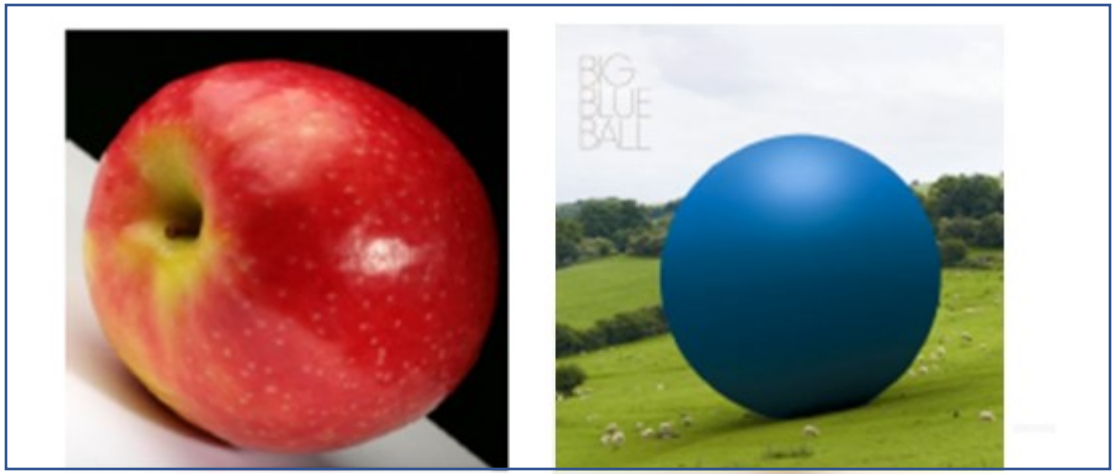
Cluster Analysis

How many clusters do you expect?

Based on Shape

Based on Color

Based on Background



Search for Outliers

Applications

Customer Segmentation

- Divides customers into groups based on common characteristics to market each group effectively and appropriately.
  - Create and communicate targeted marketing messages
  - Select best communication channel : email, social media posts, radio advertising, etc.
  - Identify ways to improve products
  - Establish better customer relationships.
  - Focus on profitable customers.
  - Upsell and cross-sell other products and services.

Image Compression

- Image pixels represented as RGB values (24 bit)
- If in an image, there are only shades of a small number of colors and if we code them the image is quantized.
- Let us say the pixels are 24 bits to represent 16 million colors,
- If there are shades of only 64 main colors, for each pixel we need 6 bits instead of 24.

Document Clustering

- Group similar documents
- Example : news : politics,sports,fashion,arts and so on.
- Document is represented as a bag of words (lexicon of N words) and grouped depending on the number of shared words.

Activity




1. You have to build a classifier. Identify the features
2. If you have to identify an unknown fruit which type of learning is it?

No.	SIZE	COLOR	SHAPE	FRUIT NAME
1	Big	Red	Rounded shape with a depression at the top	Apple
2	Small	Red	Heart-shaped to nearly globular	Cherry
3	Big	Green	Long curving cylinder	Banana
4	Small	Green	Round to oval,Bunch shape Cylindrical	Grape

- If you have no clue on what type of fruits are these, then which learning you use and how do you use it?

Module I

Lecture 3

Reinforcement Learning

Reinforcement Learning

- In some applications, Output is a sequence of actions.
- Sequence of correct actions is needed to reach goal.
- Learning program should be able to assess the goodness of policies and learn from past good action sequences to generate a policy.
- Such methods are called reinforcement learning.

Example

- **Playing a Game:**
- Here a single move by itself is not that important; it is the sequence of right moves that is good.
- A move is good if it is part of a good game playing policy.
- E.g. Chess
- Good algorithms that play games well can be applied to applications with more evident economic utility.
- E.g. Robot Navigation

Learning

- Learning happens if any computer program improves its performance at some task through experience.
- **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.

Example

- A computer program that learns to play checkers improve its performance as measured by its ability to win the game through experience obtained by playing games against itself.

Well-defined learning problem

- To have a well-defined learning problem, we must identity these three features:
  - class of tasks,
  - measure of performance to be improved, and
  - source of experience.

Examples

A checkers learning problem

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

A handwriting recognition learning problem

- **Task T:** recognizing and classifying handwritten words within images
- **Performance measure P:** percent of words correctly classified
- **Training experience E:** a database of handwritten words with given classifications

A robot driving learning problem

- **Task T:** driving on public four-lane highways using vision sensors
- **Performance measure P:** average distance traveled before an error (as judged by human overseer)
- **Training experience E:** a sequence of images and steering commands recorded while observing a human driver

Designing a Learning System

- Designing a program to learn to play checkers, with the goal of entering it in the world checkers tournament.
- Performance measure : percent of games it wins in this world tournament.

Steps in designing a learning system

1. Choosing the training experience
2. Choosing the target function
3. Choosing the representation of the target function
4. Choosing the function approximation algorithm
5. Final Design

Module I

Lecture 4

Steps in designing a learning system

A checkers learning problem

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

Steps in designing a learning system

1. Choosing the training experience
2. Choosing the target function
3. Choosing the representation of the target function
4. Choosing the function approximation algorithm
5. Final Design

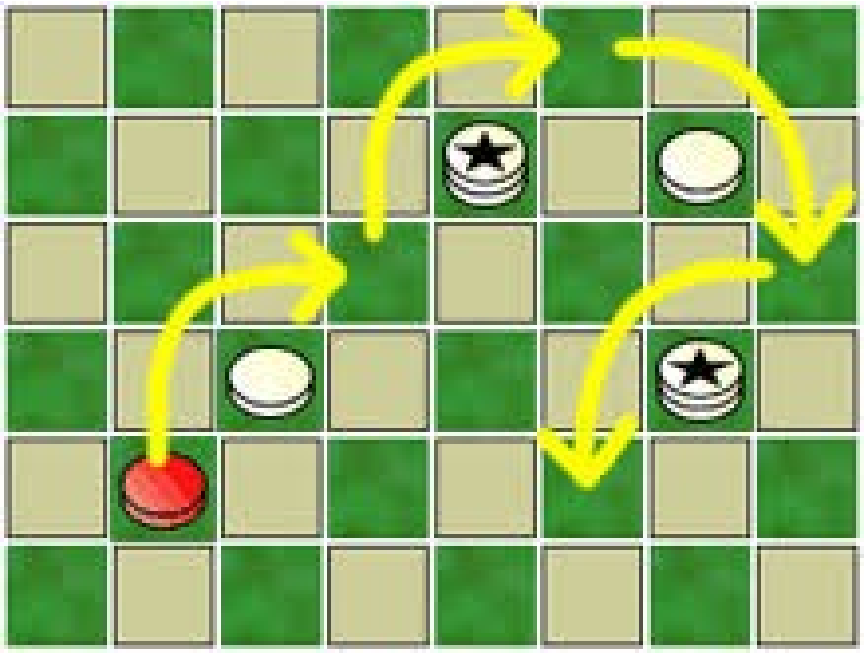
Choosing the Training Experience

- Type of training experience influence success of learner.
- The key attributes are

1. Whether training provides **direct or indirect feedback**

A checkers learning problem

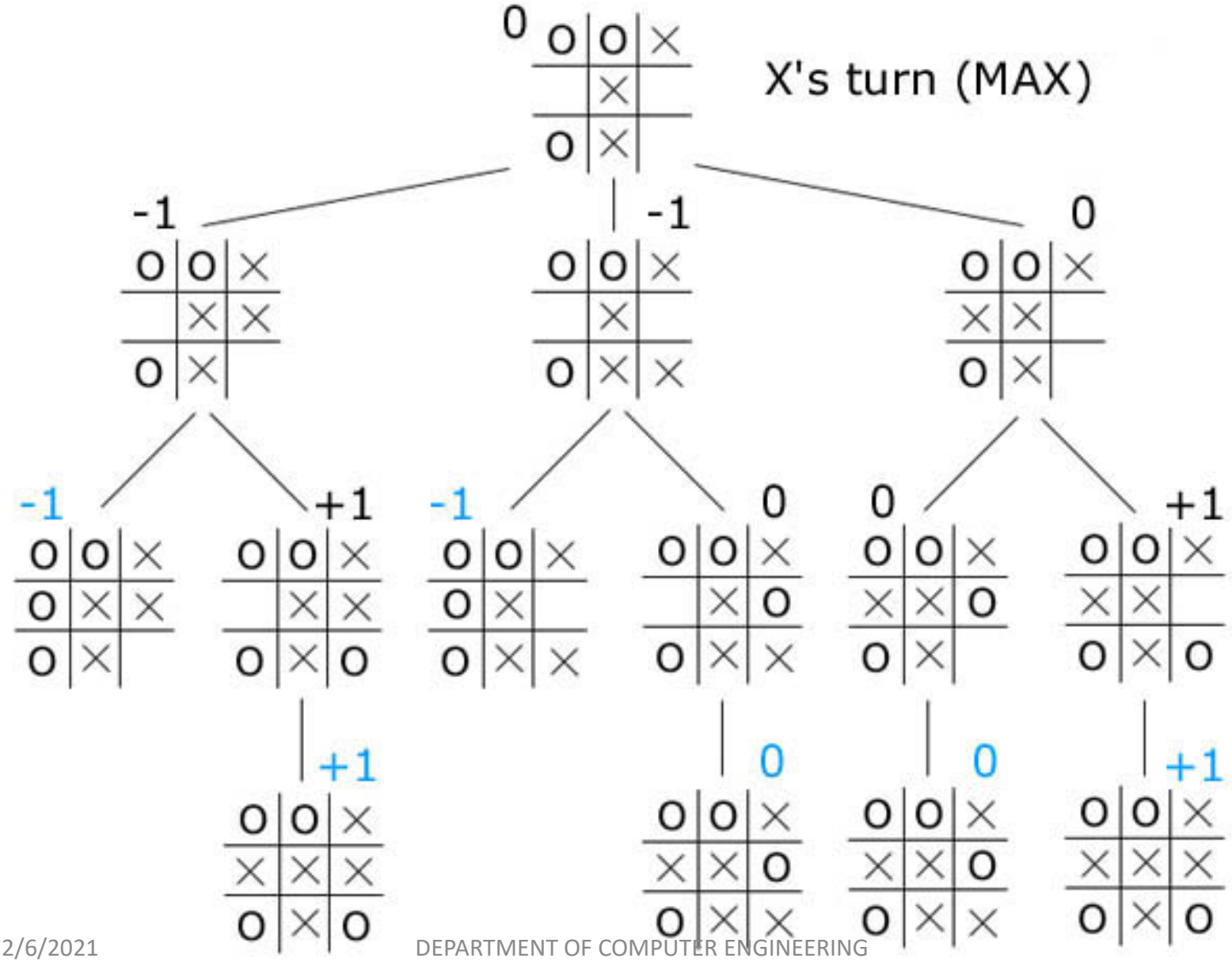
- **Direct training examples:** consists of individual checkers board states and correct move for each.



A checkers learning problem

- **Indirect training examples:** consists of move sequences and final outcomes of games played.

  (credit assignment is difficult : game can be lost even when early moves are optimal followed by poor moves)


0
X's turn (MAX)
-1
-1
0
-1
+1
-1
0
0
+1
+1
0
0
+1

A checkers learning problem

- Direct training feedback is easier

Choosing the Training Experience

2. Degree to which learner controls training examples.
   - Rely on teacher to select board states and to provide correct move for each.
   - Propose board states that it finds confusing and ask teacher for the correct move.
   - Complete control both the board states and learns by playing against itself without teacher.

Choosing the Training Experience

3. How well it represents the distribution of examples over which the final system performance P must be measured.
   - Training examples must follow a distribution similar to test examples.
   - Checkers : If its training experience consists only of games played against itself but should also play with checkers champions.
   - This assumption must often be violated in practice.

A checkers learning problem is redefined as:

- **Task T:** Playing Checkers
- **Performance measure P:** percent of games won in the world tournament
- **Training experience E:** games played against itself

Steps in designing a learning system

1. Choosing the training experience
2. Choosing the target function
3. Choosing the representation of the target function
4. Choosing the function approximation algorithm
5. Final Design

Choosing the Target Function

- It determines what type of knowledge will be learned and how it is used.
- In checkers program,
  - the legal moves from any board state are known in advance, but the best search strategy is not known.
  - System must learn to choose best move among the legal moves for any given board state.

Choosing the Target Function

- Let us call this function
  - ChooseMove : B → M
- Input : Set of legal board states B
- Output : One move from set of legal moves M
- Now the problem reduces to the problem of learning target function
  - Choice of the target function is a key design choice.

Choosing the Target Function

- ChooseMove will be difficult to learn
- An alternative target function is an evaluation function that assigns a numerical score to any given board state.
- Let us call this target function V.
- V assigns higher scores to better board states
- Generate successor board state produced by every legal move, then use V to choose the best successor state and therefore best legal move.

Choosing the Target Function

- Define target V(b) for an arbitrary board state b in B, as follows:
  1. if b is a final board state that is won, then V(b) = 100
  2. if b is a final board state that is lost, then V(b) = -100
  3. if b is a final board state that is drawn, then V(b) = 0
  4. if b is a not a final state in the game, then V(b) = V(b') where b' is the best final board state that can be achieved starting from b and playing optimally until end of game.

Choosing the Target Function

- Determining the value of V(b) for a particular board state requires searching ahead for the optimal line of play till end of the game.
- This definition is not efficiently computable and hence it is a **nonoperational definition**.

Choosing the Target Function

- Goal of learning is to **discover an operational description of V** to evaluate states and select moves within realistic time bounds.
- Learning algorithms acquire approximation to target function and it is called **function approximation**

Choosing a Representation for the Target Function

- We have many options :
  - represent using a large table with a distinct entry specifying the value for each distinct board state.
  - represent using a collection of rules that match against features of the board state
  - Use a quadratic polynomial function of predefined board features
  - Use an artificial neural network.

Choosing a Representation for the Target Function

- Let us choose a simple representation:
- For any given board state, function V will be calculated as a linear combination of the following board features:
  - x1: number of black pieces on the board
  - x2: number of red pieces on the board
  - x3: number of black kings on the board
  - x4: number of red kings on the board
  - x5: number of black pieces threatened by red (i.e., which can be captured on red's next turn)
  - x6: number of red pieces threatened by black

Choosing a Representation for the Target Function

- The learning program will represent V(b) as a linear function 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 ω_0 to ω_6 are numerical coefficients to be chosen by learning algorithm.

Partial design of a checkers learning program:

- Task T: playing checkers
- Performance measure P: percent of games won in the world tournament
- Training experience E: games played against itself
- *Target function:* $V: Board \rightarrow \Re$
- *Target function representation*

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

Steps in designing a learning system

1. Choosing the training experience
2. Choosing the target function
3. Choosing the representation of the target function
4. Choosing the function approximation algorithm
5. Final Design

Choosing a Function Approximation Algorithm

- For target function to learn we require a set of training examples each with board state b and the training value $V_{train}(b)$ for b.
- Each training example is an ordered pair of the form (b, $V_{train}(b)$).
- Example : b in which black has won ($x_2 = 0$, no red pieces),target function value $V_{train}(b)$ is +100

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

Choosing a Function Approximation Algorithm

- Now we need a procedure that first derives such training examples from indirect training experience, then adjusts the weights w_i to best fit these training examples

Estimating Training Values

- It is easy to assign a value to board states towards end of the game
- But difficult for intermediate board states.
- One simple approach was found to be successful
- Compute V(Successor(b)):
  - the next board state following b for which it is again the program's turn to move

    (i.e., the board state following the program's move and the opponent's response).

Estimating Training Values

Rule for estimating training values.

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

- It tends to be more accurate for board states closer to game's end.

Adjusting the weights

- For choosing the weights w_i to best fit the set of training examples { (b,V train(b))}, one common approach is to minimize the square error E between the training values and the values predicted by the hypothesis $\tilde{V}$

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

Adjusting the weights

- One of the several algorithm that find weights of a linear function that minimize E is called the least mean squares, or LMS training rule.

LMS weight update rule.

For each training example $\langle b, V_{train}(b) \rangle$

- Use the current weights to calculate $\hat{V}(b)$
- For each weight w_i, update it as

$$w_i \leftarrow w_i + \eta \, (V_{train}(b) - \hat{V}(b)) \, x_i$$

Steps in designing a learning system

1. Choosing the training experience
2. Choosing the target function
3. Choosing the representation of the target function
4. Choosing the function approximation algorithm
5. Final Design

The Final Design

- Final design can be described by four distinct program modules for any learning system.
- **Performance System :**
  - It solves the given performance task (playing checkers) 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/ learned evaluation function) as output.
- **Critic**:
  - It takes as input game history and produces output set of training examples of the target function.
  - Here training example is some game state, along with an estimate V_{train}.

The Final Design

- **Generalizer:**
  - It takes as input the training examples and produces an output hypothesis that is its estimate of the target function.
  - Here, Generalizer is LMS algorithm, and the output hypothesis is the function f described by the learned weights ω_0 through $\omega_{6.}$
- **Experiment Generator:**
  - It takes as input the current hypothesis (currently learned function) and outputs a new problem (i.e., initial board state) for the Performance System to explore.
  - In our example, the Experiment Generator proposes the same initial game board to begin a new game.

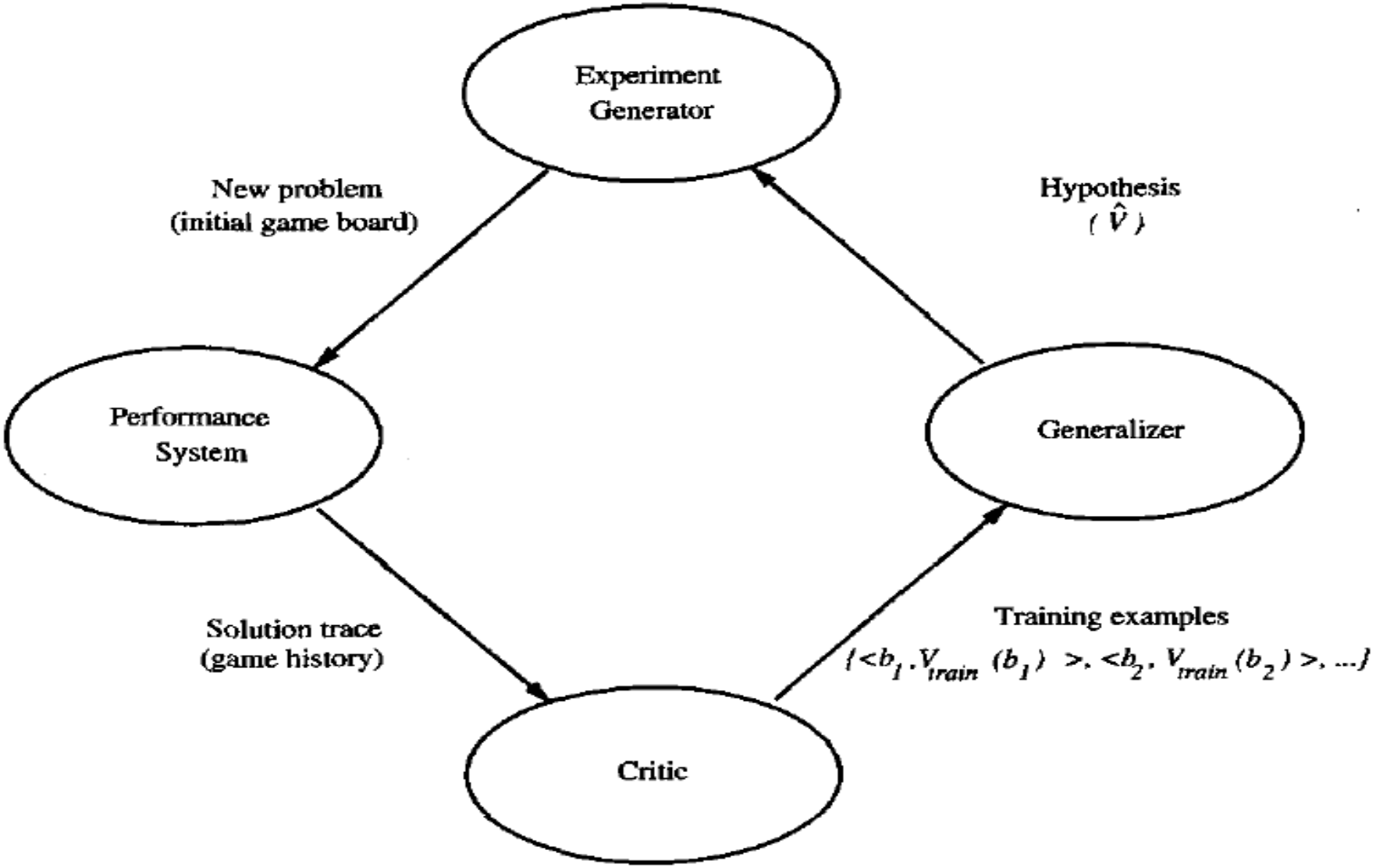
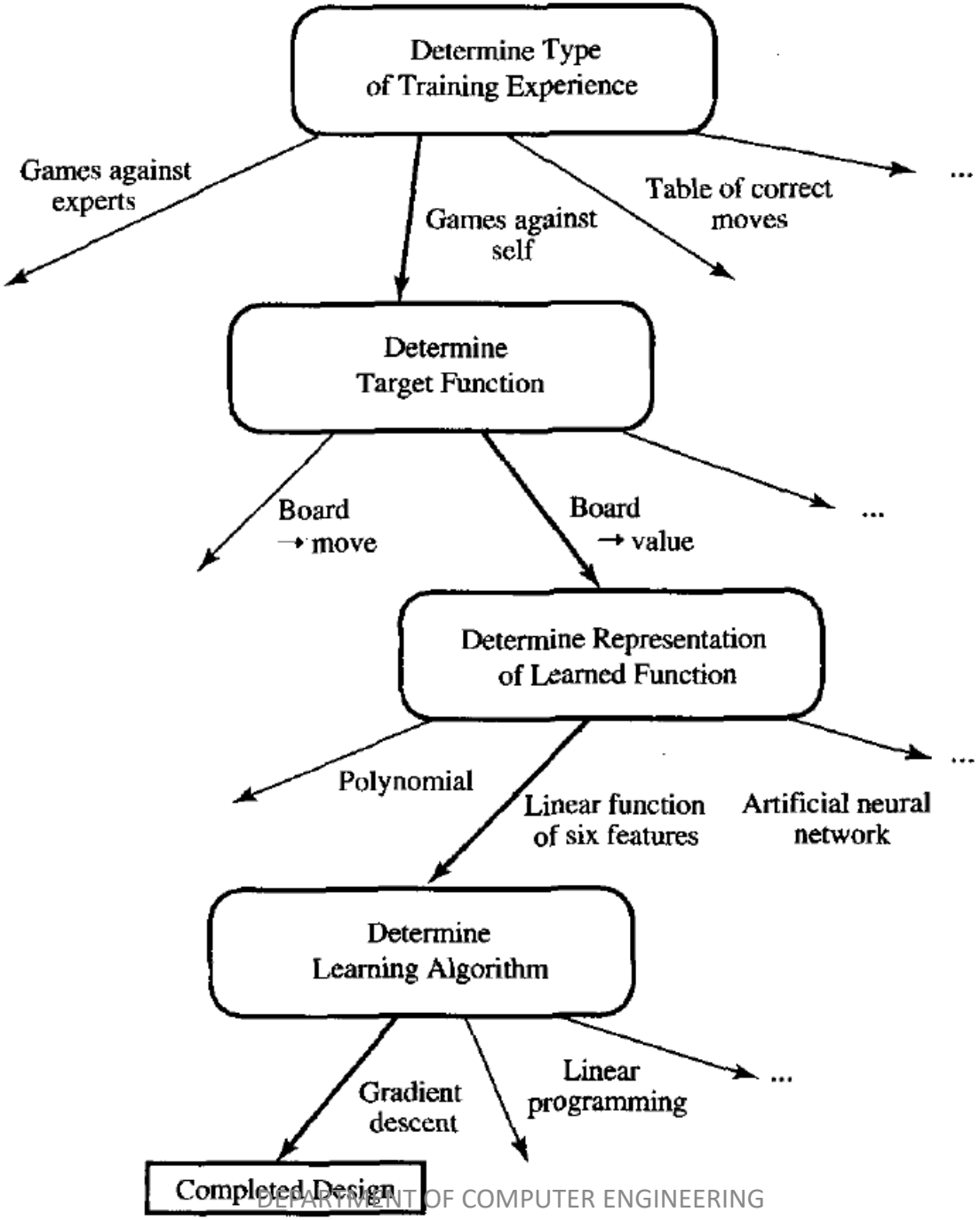


FIGURE 1.1
Final design of the checkers learning program.

Design choice summary for checkers program


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

Module I

Lecture 5

Issues in Machine Learning

Issues in Machine Learning

- 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?

Issues in Machine Learning

- 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?
- What specific functions should the system attempt to learn? 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?

How to choose the right algorithm?

- Goal has to be decided
  - If output is **prediction** : **supervised learning** to be used else unsupervised
  - If the output is **categorical** then **classification**
  - If the output is a **number** then **regression**
- If you want to fit your data into some discrete groups then use clustering
- If you need some numerical estimate of how strong it fits to each group, then use a density estimation algorithm.
- The rules are not unbreakable laws.
- E.g. You can use classification techniques for regression.
- Finding the best algorithm is an iterative process of trial and error.

Steps in developing a Machine Learning Application

1. Collect data:

- by scraping a website and extracting data, or from an RSS feed or an API.
- The number of options is endless.
- To save time and effort, you can use publicly available data.

2 Prepare the input data: Convert data into useable format

- Standard format allows to mix and match algorithms and data sources.
- Algorithm-specific formatting required : Some need features as strings and some need them to be integers.

Steps in developing a Machine Learning Application

3 Analyze the input data : looking at the data to see if you can recognize any patterns.

- Plotting data in one, two, or three dimensions can help.
- If you have more than three features, you can't easily plot data. Then use advanced methods to distill multiple dimensions down to two or three to visualize the data.

4. Train the algorithm: machine learning takes place

- Algorithm is fed with good clean data from first steps and extract knowledge.
- This knowledge is stored in a format that's useable by a machine for next steps.
- In unsupervised learning, there's no training step. Everything is used in the next step.

Steps in developing a Machine Learning Application

5. Test the algorithm: Uses the information learned in training

- Evaluate an algorithm to see how well it does.
- In supervised learning, you have known values to evaluate algorithm.
- In unsupervised learning, you may have to use some metrics to evaluate.
- If you're not satisfied with evaluation, you can go back to step 4, change some things, and try testing again. If collection of data is problem, then go back to step 1.

6 Use it: Makes a real program to do some task

- Once again you see if all the previous steps worked as you expected.
- If you encounter some new data , you have to revisit steps 1–5.

Module I
THE END