Machine Learning (KCS-055)

RAGHURAJ SINGH
Assistant Professor
Department of CSE, MIET Meerut

What is Machine Learning ?

- **Machine learning** is an application of artificial intelligence (AI).
- It is a field of study that gives computer the ability to automatically learn and improve from experience without being explicitly programmed.

- ***The ability to learn itself without the being explicit programmed.***

  [Arthur Samuel (1959)]

- As intelligence requires knowledge, it is necessary for the computers to acquire knowledge.

What is Machine Learning ?

Well-posed Learning Problem

Any problem can be classified as well posed learning problem if it has three features:

- Task (T)
- Performance Measure (P)
- Experience (E)

A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.

Example

Suppose your email program watches which emails you do or do not mark as spam, and based on that learns how to better filter spam. What is the task T, performance P and Experience E in this setting?

- Classifying emails as spam or not spam. (T)
- Watching you label emails as spam or not spam. (P)
- The number (or fraction) of emails correctly classified as spam/not spam. (E)
- None of the above—this is not a machine learning problem.

Example 1

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

Example 2

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.

Example 3

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.

Other Examples

- Other examples include – Spam Filtering Problem, Fruit Prediction Problem, and Face Recognition Problem etc.

Machine Learning System has 3 important features –

- Remember - system must remember what it is learning.
  - 2+3 =5 at the age 5
  - 2+3 =5 at the age 50
- Adapt - become adjusted to new conditions.
- Generalize – Allows the system to make prediction on unknown data i.e. ability to classify the new sample.
  - Seen Data: 2+3 =5
  - Seen Data: 10+2 =12
  - Unseen Data: 2+2 =4

Categorizations of Machine Learning Problem

1. Supervised Learning

Supervised learning as the name indicates the presence of a supervisor as a teacher. Basically supervised learning is a learning in which we teach or train the machine using data which is well labeled that means some data is already tagged with the correct answer.

For instance, suppose you are given a basket filled with different kinds of fruits. Now the first step is to train the machine with all different fruits one by one like this:

- If shape of object is rounded and depression at top having color Red then it will be labelled as –**Apple**.
- If shape of object is long curving cylinder having color Green-Yellow then it will be labelled as –**Banana**.

Contd..

- Now suppose after training the data, you have given a new separate fruit say Banana from basket and asked to identify it.

Contd..

Since the machine has already learned the things from previous data and this time have to use it wisely. It will first classify the fruit with its shape and color and would confirm the fruit name as BANANA and put it in Banana category. Thus the machine learns the things from training data(basket containing fruits) and then apply the knowledge to test data(new fruit).

Types of Supervised Learning

Supervised learning classified into two categories of algorithms:

1. Classification **2. Regression**

1. **Classification**: A classification problem is when the output variable is a category, such as “Red” or “blue” or “disease” and “no disease”.
2. **Regression**: A regression problem is when the output variable is a real value, such as “dollars” or “weight”.

Regression v/s Classification Machine Learning

- In a classification problem, we are trying to predict results in a discrete output (yes/no | 1/0).
- In a regression problem, we are trying to predict results within a continuous output.
- **Example**:
- Given data about the size of houses on the real estate market, try to predict their price. Price as a function of size is a continuous output, so this is a regression problem.
- We could turn this example into a classification problem by instead making our output about whether the house "sells for more or less than the asking price." Here we are classifying the houses based on price into two discrete categories.

2. Unsupervised learning

Unlike supervised learning, no teacher is provided that means no training will be given to the machine. Therefore machine is restricted to find the hidden structure in **unlabelled** data by our-self. Here the task of machine is to group unsorted information according to similarities, patterns and differences without any prior training of data.

Contd..

- **For instance**, suppose it is given an image having both dogs and cats which have not seen ever.

- Thus the machine has no idea about the features of dogs and cat so we can't categorize it in dogs and cats.

Contd..

- But it can categorize them according to their similarities, patterns, and differences i.e., we can easily categorize the above picture into two parts.
- First may contain all pics having **dogs** in it and second part may contain all pics having **cats** in it.
- Here you didn’t learn anything before, means no training data or examples.

Types of Unsupervised Learning

There are two types of Unsupervised Learning

1. Clustering **2. Association**

1. **Clustering** is an important concept when it comes to unsupervised learning. It mainly deals with finding a structure or pattern in a collection of uncategorized data. Clustering algorithms will process your data and find natural clusters (groups) if they exist in the data. You can also modify how many clusters your algorithms should identify. It allows you to adjust the granularity of these groups.
2. **Association**- This unsupervised technique is about discovering exciting relationships between variables in large databases. For example, people that buy a new home most likely to buy new furniture.

3. Reinforcement Learning

- Reinforcement learning is fairly different when compared to supervised and unsupervised learning.
- Where we can easily see the relationship between supervised and unsupervised (the presence or absence of labels).
- As compared to unsupervised learning, reinforcement learning is different in terms of goals.
- While the goal in unsupervised learning is to find similarities and differences between data points.
- Reinforcement Learning is a type of machine learning technique that enables an agent to learn in an interactive environment by trial and error using feedback from its own actions and experiences.

- Environment can be real world (robot) or simulated world (ex. Video game).
  - Unlike supervised learning where feedback provided to the agent is correct set of actions for performing a task.
  - Though both supervised and reinforcement learning use mapping between input and output.
  - Reinforcement learning uses rewards and punishment as signals for positive and negative behaviour.
  - In reinforcement learning the goal is to find a suitable action model that would maximize the total cumulative reward of the agent.

- The agent receives rewards by performing correctly and penalties for performing incorrectly.
- The agent learns without intervention from a human by maximizing its reward and minimizing its penalty.
- **In reinforcement learning**, an agent, learns by interacting with its environment.
- RL can guide an agent on how to act in the real world (robot that vacuum your house, robot that learn to walk).
- Military is interested in this tech: RL agent can replace soldiers not just walk, but fight, diffuse bombs, make important decisions.

Diagram

- It is Inspired by human brain. We have eyes and nose as sensory organ but machine has sensors as sensory organ.
  - Agent- an important component to process the information, thing that senses the environment.
  - Environment – real world or simulated world that the agent lives in.
  - State- different configuration of the environment that the agent can sense.
  - Actions: are what an agent does in its environment. Notation (s,a,r).
  - Rewards- positive or negative.

Designing a Learning System

Designing a Learning System can be illustrated by designing a program to learn to play checkers game with the goal of entering it in the worlds checkers tournament.

1. Choosing the training experience.
2. Choosing the target function.
3. Choosing a representation for the target function.

Contd..

4. Choosing a function approximation algorithm.
    a. Estimating training values.
    b. Adjusting the weights.
5. Final Design.

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.

There are 3 attribute which impact on success or failure of the learner.

- Whether training experience provide direct or indirect feedback regarding choice made by the performance system.
- The degree to which the learner control the sequence of the training example.
- How well it represents the distribution of examples over which the final system performance P must be measured.

2. 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 take the example of a checkers-playing program that can generate the legal moves (M) from any board state (B). The program needs only to learn how to choose the best move from among these legal moves. Let's assume a function **NextMove** such that:
- NextMove: B -> M
- Here, **B** denotes the set of board states and **M** denotes the set of legal moves given a board state. NextMove is our target function.

3. Choosing a representation for the target function.

- We need to choose a representation that the learning algorithm will use to describe the function NextMove .The function NextMove will be calculated as a linear combination of the following board features:
- **xl**: the number of black pieces on the board
- **x2**: the number of red pieces on the board
- **x3**: the number of black kings on the board
- **x4**: the number of red kings on the board
- **x5**: the number of black pieces threatened by red
- **x6**: the number of red pieces threatened by black

NextMove = u0 + u1x1 + u2x2 + u3x3 + u4x4 + u5x5 + u6x6

Here u0, u1 up to u6 are the coefficients that will be chosen by the learning algorithm.

4. Choosing a function approximation algorithm.

To learn the target function NextMove, we require a set of training examples, each describing a specific board state b and the training value (Correct Move) y for b. The training algorithm learns/approximate the coefficients u0, u1 up to u6 with the help of these training examples by estimating and adjusting these weights.

5. Final Design

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

Diagrammatic representation is of final design is shown below

Issues in Machine Learning

Following are some basic issues in machine learning.

- What algorithms exist for learning general target functions from specific training examples?
- What method to follow to learn the training examples.
- According to data (image, text etc) which method is suggestible.
- How much training data is sufficient? (No. of training examples and test set).
- To reduce overhead of learning which method to use.
- When and how can prior knowledge held by the learner guide the process of generalizing from examples?

- What is the best way to reduce the learning task to one or more function approximation problems?
- How can the learner automatically alter **its representation to improve its** ability to represent and learn the target function?
- 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?

Concept Learning

A Formal Definition for Concept Learning:

- Inferring a Boolean-valued function from training examples of its input and output.
- Concept learning is a learning task in which we train our machine to learn some concept. For ex. Yes, No, +ve,-ve.
- An example for concept-learning is the learning of bird-concept from the given examples of birds (positive examples) and non-birds (negative examples). For ex. data set={ parrot, sparrow, cat, dog}
- We are trying to learn the definition of a concept from given examples.

A Concept Learning Task – Enjoy Sport Training Examples

- A set of example days, and each is described by six attributes.
- The task is to learn to predict the value of EnjoySport for arbitrary day, based on the values of its attribute values.

EnjoySport – Hypothesis Representation

- Each hypothesis consists of a conjuction of constraints on the instance attributes.

- Each hypothesis will be a vector of six constraints, specifying the values of the six attributes
  - (Sky, AirTemp, Humidity, Wind, Water, and Forecast).
- Each attribute will be:
- ? - indicating any value is acceptable for the attribute (don’t care)
- single value – specifying a single required value (ex. Warm) (specific)
- Ø - indicating no value is acceptable for the attribute (**no value**)

Hypothesis Representation

A hypothesis:

<Sky AirTemp Humidity Wind Water Forecast>

< Sunny, ? , ? , Strong , ? , Same >

- *The most general hypothesis – that every day is a positive example* <?, ?, ?, ?, ?, ?>
- *The most specific hypothesis – that no day is a positive example* < Ø, Ø, Ø, Ø, Ø, Ø >

- *EnjoySport concept learning task requires learning the sets of days for* which EnjoySport=yes, describing this set by a conjunction of constraints over the instance attributes.

EnjoySport Concept Learning Task

Given

– ***Instances X : set of all possible days, each described by the attributes***

- Sky – (values: Sunny, Cloudy, Rainy)
- AirTemp – (values: Warm, Cold)
- Humidity – (values: Normal, High)
- Wind – (values: Strong, Weak)
- Water – (values: Warm, Cold)
- Forecast – (values: Same, Change)

– Target Concept (Function) c : EnjoySport : $X \rightarrow \{0,1\}$

– Training Examples D : positive and negative examples of the target function

Instance: The set of Items over which the concept is defined is called the set of instance.

Target Concept: The concept or function to be learned is called the target concept.

- The concept or function to be learned is called the target concept, which we denote by c. In general, c can be any boolean valued function defined over the instances X; that is, $c : X \rightarrow$ **{O, 1).**
- ***In the current*** example, the target concept corresponds to the value of the attribute EnjoySport (i.e., c(x) = 1 if EnjoySport = Yes, and c(x) = 0 if EnjoySport = No).

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 over other unobserved examples.

Concept Learning As Search

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

Contd….

- By selecting a hypothesis representation, the designer of the learning algorithm implicitly defines the space of all hypotheses that the program can ever represent and therefore can ever learn.

Enjoy Sport - Hypothesis Space

- Sky has 3 possible values, and other 5 attributes have 2 possible values.
- There are 96 (= 3.2.2.2.2.2) distinct instances in X.
- There are 5120 (=5.4.4.4.4.4) syntactically distinct hypotheses in H. (two hypothesis positive and Negative)

- It can classifies every instance as negative
- Every hypothesis contain null
- There are 973 (= 1 + 4.3.3.3.3.3) semantically distinct hypotheses in H.

– Only one more value for attributes: positive, and one hypothesis representing empty set of instances.

- Although EnjoySport has small, finite hypothesis space, most learning tasks have much larger (even infinite) hypothesis spaces.

– We need efficient search algorithms on the hypothesis spaces.

Flowchart

Concept: Days on which person enjoys sport

Find-s algo solution

h0= < Ø, Ø, Ø, Ø, Ø, Ø >

h1=<‘Sunny’, ‘Warm’, ‘Normal’, ‘Strong’, ‘Warm’, ‘Same’>

h2=<‘Sunny’, ‘Warm’, ‘?’, ‘Strong’, ‘Warm’, ‘Same’>

h3=h2

h4=< ‘Sunny’, ‘Warm’, ‘?’, ‘Strong’, ‘Warm’, ‘?’>

Example: Training Example of Enjoy Sport

h0= < Ø, Ø, Ø, Ø, Ø, Ø >

h1=<‘Sunny’, ‘Warm’, ‘Normal’, ‘Strong’, ‘Warm’, ‘Same’>

h2=<‘Sunny’, ‘Warm’, ‘?’, ‘Strong’, ‘Warm’, ‘Same’>

h3=h2

h4=< ‘Sunny’, ‘Warm’, ‘?’, ‘Strong’, ‘?’, ‘?’>

Candidate Elimination Algorithm

Candidate Elimination Algorithm

1. Initialize G and S as more general and specific hypothesis.
2. For each example e :

   if e is +ve:

   make specific hypothesis more general [Find s]

   else:

   make general hypothesis more specific

Example: Training Example of Enjoy Sport

Some Important Points

- Positive example will be considered by most specific hypothesis.
- Negative example will be considered by most general hypothesis.
- Positive example has no impact on most general hypothesis.
- Negative example has no impact on most specific hypothesis.

Solution

S-1) S0= < Ø, Ø, Ø, Ø, Ø, Ø > -most specific

G0=<?,?,?,?,?,?> - most general

S-2) 1. S1=< ‘sunny’, ‘warm’, ‘normal’, ‘strong’, ‘warm’, ‘same’>

G1=<?,?,?,?,?,?>

2. S2=< ‘sunny’, ‘warm’, ‘?’, ‘strong’, ‘warm’, ‘same’>

G2=<?,?,?,?,?,?>

3. S3=< ‘sunny’, ‘warm’, ‘?’, ‘strong’, ‘warm’, ‘same’>

G3={<‘sunny’,?,?,?,?,?>, <?, ‘warm’, ?,?,?,?>

<?,?,?,?,?,’same’>}

4. S4=< ‘sunny’, ‘warm’, ‘?’, ‘strong’, ‘?’, ‘?’>

G4={<‘sunny’,?,?,?,?,?>, <?, ‘warm’, ?,?,?,?>}

Version Space

S4=< ‘sunny’, ‘warm’, ‘?’, ‘strong’, ‘?’, ‘?’>

<‘Sunny’,?,?, ‘Strong’, ?,?>, <‘Sunny’, ‘Warm’, ?,?,?,?>, <?,‘warm’, ?, ‘strong’, ?,?>

G4={<‘sunny’,?,?,?,?,?>, <?, ‘warm’, ?,?,?,?>}

Version Space

The version space denoted by $VS_{H,D}$ with respect to hypothesis space H and training example D, is the subset of hypothesis from H consistent with training example in D.

$$VS_{H,D} = \{ h \in H: \text{Consistent } (h, D)\}$$

The set of concepts consistent with a set of training example is called version space.

- Version space method involves identifying all concepts consistent with a set of training examples.

Hypothesis Space

- Set of all legal hypothesis is called hypothesis space that can be described by the features you have choosen or the language you have choosen. Let hypothesis space is represented by H so H= HS
- A hypothesis is consistent with a set of training example D if and only if h(x)=c(x) for each example <x, c(x)> in D.
- Consistent (h,D)=(for all x <x, c(x)> Є D) h(x)=c(x)

List –Then Eliminate Algorithm

- List- Then eliminate algorithm first initialize the version space to contain all hypothesis in H, then eliminate any hypothesis found inconsistent with any training examples.
- It has many advantages including the fact that it is guaranteed to output all hypothesis consistent with training data.
- List- Then eliminate algorithm can be applied whenever hypothesis space H is finite.

List –Then Eliminate Algorithm

Obvious way to represent version space

Unrealistic

Inductive Bias

- The tendency to prefer one hypothesis over another is called bias.
- The simplest consistent hypothesis about the target function is actually the best.
- A hypothesis is consistent if it agree with all training examples.
- A hypothesis said to generalize well if it correctly predicts the value of y for novel example.
- If something can be described in a short language, that will be preferable over a complex hypothesis.
- The inductive bias of learning algorithm is the set of assumption that the learner uses to predict outputs on given inputs.

- Need to make assumptions-

- Experience alone doesn't allow us to make conclusion about unseen data instances.
- There are two types of Bias

1. Constraints or Restriction- Limit the hypothesis Space. We can restrict hypothesis by defining constraints on bias.
2. Preference- Impose ordering on hypothesis space. We can prefer lower order hypothesis over high order or complex hypothesis which is something that can be express in short.

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

A classical example of inductive bias is Occam's Razor, states that we will prefer simplest hypothesis that can be described in a short language and that will be preferable over large or complex hypothesis.

Other types of Inductive Bias

- **Minimum description length:** when forming a hypothesis attempt to minimize the length of the description of the hypothesis.
- **Maximum Margin:** When drawing a boundary between two classes attempt to maximize the width of the boundary (SVM).

Bayesian Networks

- "*Bayesian Network*," provides a simple way of applying Bayes Theorem to complex problems.
- Bayesian networks falls under the category of probabilistic graphical modeling technique that is used to compute uncertainties by using the concept of probability.
- Bayesian networks uses Bayesian inference for probability computations.
- Bayesian networks aim to model conditional dependence, and therefore caused, by representing conditional dependence by edges in a directed graph.

Naive Bays Theorem

- It is a supervised learning algorithm used for classification.
- Based on Bays theorem

- Finding the probability of event A when event B is true.
- P(A|B)= Posterior Probability (Probability of event after event B is true)
- P(A)= Prior probability
- P(B)= Marginal probability

- Bayes theorem provides a way to calculate the probability of a hypothesis based on its prior probability, the probabilities of observing various data given the hypothesis, and the observed data itself.

  $P(A|B) = P(A \cap B) / P(B)$ Similarly, $P(B|A) = P(A \cap B) / P(A)$
- It follows that $P(A \cap B) = P(A|B) * P(B) = P(B|A) * P(A)$
- Thus, $P(A|B) = P(B|A)*P(A) / P(B)$ This is the Bayes Theorem.
- Here, P(A) and P(B) are probabilities of observing A and B independently of each other.
- That's why we can say that they are marginal probabilities.
- P(B|A) and P(A|B) are called conditional probabilities.
- P(A) is called **Prior probability** and P(B) is called **Evidence**.

What Bayes Theorem Exactly describes?

- In probability **theory** and statistics, **Bayes theorem describes** the probability of an event, based on prior knowledge of conditions that might be related to the event.

Example

Data Set: Fruit= {Yellow, Sweet, Long}

P(Yellow|Orange)= P(Orange|Yellow)*P(Yellow)/P(Orange)

= (350/800)* (800/1200)/ (650/1200)

= 0.53

P(Sweet|Orange)= 450/650= 0.69

P(Long|Orange)= 0/650 = 0.0

P(Fruit|Orange) = 0.53 x 0.69 x 0= 0

P(Fruit| Banana)= 1 x 0.75 x 0.87= 0.65

P(Fruit| Others)= 0.33 x 0.66 x 0.33= 0.072

History of Machine Learning

Continued..

- In 1950s Alan Turing conducted a test to check if machine could fool human being believing them they are talking to a machine.
- 1952: FIRST ever computer learning program was developed named checkers by Arthur Samuel.
- 1957: First neural networks were discovered by Frank which simulates the thought process of human brains.
- 1967: The nearest Neighbour algorithm was discovered.
- 1979: Students at Stanford university developed a cart that was capable of removing obstacles on its own.
- 1997: IBMS DEEP blue defeated the worlds chess champion.
- 2002: A software library for machine learning named TORCH has been released.
- 2016: ALPHA GO algorithm was developed by Google's DEEPMIND to win five games of five.

What is GA

- Genetic algorithm are adaptive heuristics search algorithms based on the evolutionary ideas of natural selection and genetics.
- A **genetic algorithm** (or **GA**) is a search technique used in computing to find true or approximate solutions to optimization and search problems.
- Genetic algorithms are categorized as global search heuristics.
- Genetic algorithms are a particular class of evolutionary algorithms that use techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover (also called recombination).

What is GA

- Genetic algorithms are implemented as a computer simulation in which a population of abstract representations (called chromosomes or the genotype or the genome) of candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem evolves toward better solutions.

- Traditionally, solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible.

What is GA

- The evolution usually starts from a population of randomly generated individuals and happens in generations.

- In each generation, the fitness of every individual in the population is evaluated, multiple individuals are selected from the current population (based on their fitness), and modified (recombined and possibly mutated) to form a new population.

What is GA

- The new population is then used in the next iteration of the algorithm.
- Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population.

Key terms

- **Individual** - Any possible solution
- **Population** - Group of all *individuals*
- **Search Space** - All possible solutions to the problem
- **Chromosome** - Blueprint for an *individual*
- **Trait** - Possible aspect (*features)* of an *individual*
- **Locus** - The position of a *gene* on the *chromosome*
- **Genome** - Collection of all *chromosomes* for an *individual*

Chromosome, Genes and Genomes

Genotype and Phenotype

- ***Genotype:***
  - Particular set of genes in a genome

- ***Phenotype:***
  - Physical characteristic of the genotype (smart, beautiful, healthy, etc.)

Genotype and Phenotype

Fitness Function

- Fitness function f(x) is derived from objective function f(x) and used in successive genetic operators.

  f(x) = f(x) for maximization Problem

  f(x)= 1/f(x) for minimization problem if f(x) !=0

  and f(x)= 1/1+f(x) if f(x)= 0
- The fitness function value of the string is called string's fitness.
- Certain genetic operators require that fitness function to be non negative.

Genetic Algorithm

Pseudocode -:

Begin

Initialise population with random candidate solution

Evaluate each candidate (fitness value)

Repeat until termination condition is satisfied

do

1. Select parents
2. Recombine pair of parents
3. Mutate the resulting offspring
4. Select individual or next generations

End

Flowchart of GA

Applications of GA

- Optimization – GA have been used in solving variety of different problem which are difficult to solve.
- Robotics- Use of GA in the field of robotics is quite big.
- Aerodynamic
- Medicine
- Clustering
- Multidimensional Systems
- Mutation testing