Designing a Learning System

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*.

The above definition is one of the most well known definitions of Machine Learning given by Tom Mitchell.

(**Tom Michael Mitchell** is an American computer scientist and Professor at the Carnegie Mellon University (CMU). He is a former Chair of the Machine Learning Department at CMU. Mitchell is known for his contributions to the advancement of machine learning, artificial intelligence, and cognitive neuroscience and is the author of the textbook Machine Learning)

For any learning problem, we must be knowing the factors **T (Task)**, **P (Performance Measure)**, and **E (Training Experience)**. Let's take a few examples to understand these factors.

Problem 1: Handwriting recognition learning problem

For handwriting recognition learning problem, TPE would be,

Task T: To recognize and classify handwritten words within the given images.

Performance measure P: Total percent of words being correctly classified by the program.

Training experience E: A set of handwritten words with given classifications/labels.

Problem 2: Spam Mail detection learning problem

For a system being designed to detect spam emails, TPE would be,

Task T: To recognize and classify mails into 'spam' or 'not spam'.

Performance measure P: Total percent of mails being correctly classified as 'spam' (or 'not spam') by the program.

Training experience E: A set of mails with given labels ('spam' / 'not spam').

Concept Learning in ML

Concepts in Machine Learning can be thought of as a **boolean-valued** function defined over a large set of training data.

Taking a very simple example, one possible target concept may be to ***Find the day when my friend Ramesh enjoys his favorite sport***. We have some attributes/features of the day like, ***Sky***, ***Air Temperature***, ***Humidity***, ***Wind***, ***Water***, ***Forecast*** and based on this we have a target Concept named **EnjoySport**.

We have the following training example available:

Example	Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport

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

Let’s Design the problem formally with **TPE**(Task, Performance, Experience):

Problem: Learning the day when Ramesh enjoys the sport.

Task T: Learn to predict the value of **EnjoySport** for an arbitrary day, based on the values of the attributes of the day.

Performance measure P: Total percent of days (EnjoySport) correctly predicted.

Training experience E: A set of days with given labels (EnjoySport: Yes/No)

Let us take a very simple hypothesis representation which consists of a **conjunction** of constraints in the instance attributes. We get a hypothesis h_i with the help of example i for our training set as below:

hi(x) := <x1, x2, x3, x4, x5, x6>

where x1, x2, x3, x4, x5 and x6 are the values of **Sky**, **AirTemp**, **Humidity**, **Wind**, **Water** and **Forecast**.

Hence h1 will look like(the first row of the table above):

h1(x=1): <Sunny, Warm, Normal, Strong, Warm, Same > *Note: x=1 represents a positive hypothesis / Positive example*

We want to find the most suitable hypothesis which can represent the concept. For example, Ramesh enjoys his favorite sport only on **cold days** with **high humidity** (This seems independent of the values of the other attributes present in the training examples).

h(x=1) = <?, Cold, High, ?, ?, ?>

Here `?` indicates that any value of the attribute is acceptable. **Note:** The most generic hypothesis will be < ?, ?, ?, ?, ?, ?> where every day is a positive example and the most specific hypothesis will be <?,?,?,?,?,? > where no day is a positive example.

We will discuss the two most popular approaches to find a suitable hypothesis, they are:

1. Find-S Algorithm
2. Candidate Elimination Algorithm

Find-S Algorithm:

Following are the steps for the Find-S algorithm:

1. Initialize **h** to the most specific hypothesis in **H**
2. For each positive training example,
    1. For each attribute, constraint **ai** in **h**

    1. If the constraints **ai** is satisfied by **x**
    2. Then do nothing
    3. Else replace **ai** in **h** by the next more general constraint that is satisfied by **x**
3. Output hypothesis **h**
