Machine Learning

MA842

Course content

- Introduction to machine learning
- Decision Tree Learning
- Artificial Neural Networks
- Computational Learning Theory
- Instance-Based Learning
- Genetic Algorithms
- Analytical Learning
- Reinforcement Learning.

Machine Learning : Definition

- "A computer program is said to learn from experience E (without being explicitly programmed) 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.“

$$E * T = P$$

- Tom Mitchell

- Example: playing checkers.
- E = the experience of playing many games of checkers
- T = the task of playing checkers.
- P = the probability that the program will win the next game.

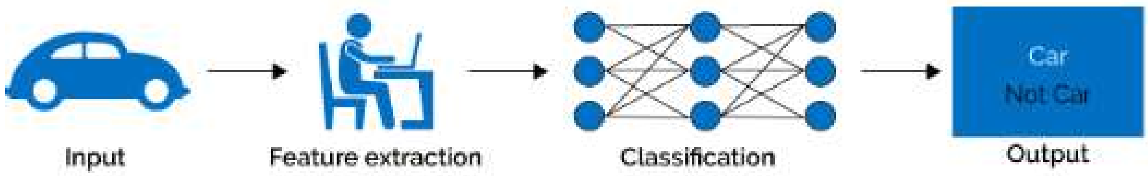
Application of ML

- **Transportation System**: Google maps provides the best route to reach the destination based on GPS location such that the congestion is minimum.
  Task – Build map of current traffic
  Experience – Learn the live scenario using GPS location, average velocity
- **Virtual Personal Assistants:** Collect and refine the information on the basis of our previous involvement with the personal assistant.
- **Product recommendations based on browsing history:** The shopping website or the app recommends some items that somehow matches with our taste.
- **3D printing of machine parts:** The sensor integrated system learns to minimize the production of defective machine parts by learning from previous flaws.

https://www.youtube.com/watch?v=ahRcGObyEZo

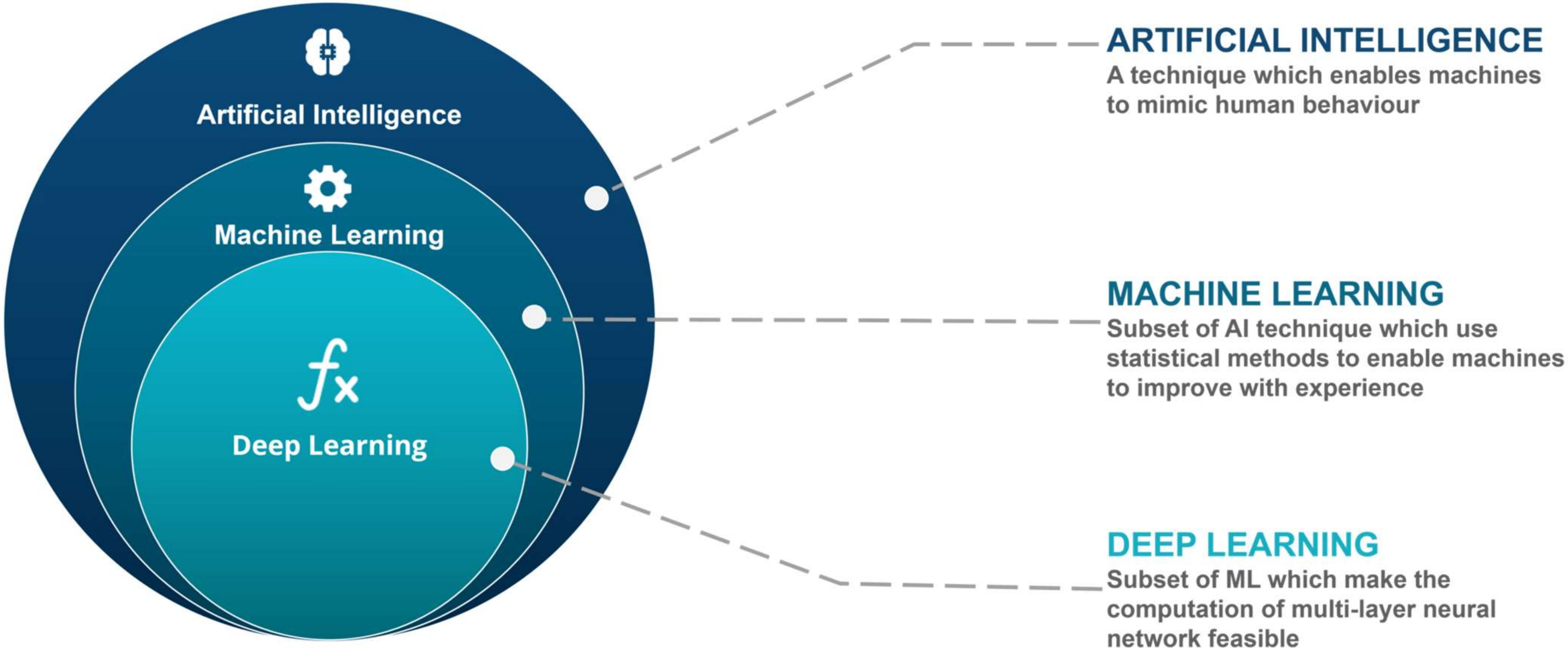
Additional Examples

- Face detection in AI glasses for blind
- Handwriting recognition
- Autonomous vehicle
- Robot to clean the house
- Email filtering
- Weather prediction
- Medical diagnosis
- Stock market analysis



Source: https://semiengineering.com/deep-learning-spreads/

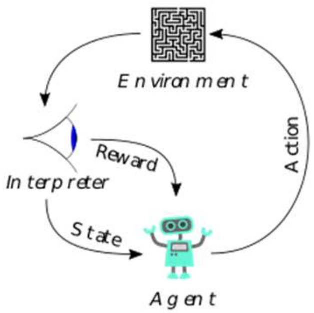
Where does ML stand?



Source: https://www.edureka.co/blog/ai-vs-machine-learning-vs-deep-learning/

Types of ML

- Supervised
  - Classification – A product is defective or not
  - Regression – Predict the price of a house
- Unsupervised
  - Association - People that buy X also tend to buy Y
  - Clustering - Grouping customers by purchasing behavior
- Reinforcement – autonomous driving car



Source: https://en.wikipedia.org/wiki/Reinforcement_learning

Designing a learning system

- Choosing the training experience
  - Type of training experience : Direct / Indirect feedback (credit assignment problem)
  - Degree to which the learner controls the sequence of training examples.
  - Represent distribution of examples over which performance is measured.

- Choosing the target functions
  - Determine exactly what type of knowledge will be learned and how this will be used by performance program.
  - Target function V: Board $\rightarrow$ Real number
  - Define target value V(b) for an arbitrary board state b:

Final board state b	Value V(b)
Won	100
Lost	-100
Drawn	0

If b is not final state, V(b) = V(b′), where b′ is best final board state

- Choosing the representation for target function
  - Linear function:

$$\hat{V}(b) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + w_6x_6$$

Where:

$w_0, w_1, w_2, w_3, w_4, w_5, and\ w_6$ - weights

x_1 : Number of black pieces

x_2 : Number of white pieces

x_3 : Number of black king pieces

x_4 : Number of white king pieces

x_5 : Number of black pieces threatened by white

x_6 : Number of white pieces threatened by black

In general, x_1 to $x_n$represents the features projected on n dimensional feature space

- Choosing the function approximation algorithm

To learn $\hat{V}$ we need set of training examples, <b,$V_{train}(b)$>

Example: <<x_1 =3, x_2=0, x_3 =1,x_4=0, x_5 =0,x_6=0> , +100>

Aim:

To derive such training examples and adjust the weight values to best fit these training examples {<b,$V_{train}(b)$>}

Rule for estimating the training values for any intermediate board state b

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

where $\hat{V}$ is the learner's current approximation to V and $\hat{V}(successor(b))$denotes the next board state following b

- Adjusting the weights
  - Define the best hypothesis or set of weights such that squared error E between training values and the values predicted by the hypothesis $\hat{V}$ is minimum.

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

One algorithm to incrementally refine the weights as new training examples become available is least mean square (LMS) training rule.

LMS Algorithm

Weight update rule

- For each training example <b,$V_{train}(b)$>
  - 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

Where η is a constant that moderates the size of the weight update.
If ($V_{train}(b)$ - $\hat{V}$(b)) =0, no weights are changed
If ($V_{train}(b)$ - $\hat{V}$(b)) >0, each weight is increased in proportion to the value of its corresponding feature x_i

An example of LMS

Suppose we have a data set of 6 points as shown:

i	x_i	y_i
1	1.2	1.1
2	2.3	2.1
3	3.0	3.1
4	3.8	4.0
5	4.7	4.9
6	5.9	5.9

We find the best fitting line as follows.

We define the Mean Squared Error function. Let $g(x) = mx + b$ represent a generic line. Its mean squared error in approximating the data is then

$$MSE = \frac{1}{6}\sum_{i=1}^{6}(y_i - g(x_i))^2 = \frac{1}{6}\sum_{i=1}^{6}(y_i - (mx_i + b))^2$$

Expanding this we have

$$\frac{1}{6}\sum_{i=1}^{6}\left(y_i^2 - 2x_iy_im - 2y_ib + x_i^2m^2 + 2x_imb + b^2\right)$$

This sum can be rewritten as

$$MSE = \frac{1}{6}\sum_{i=1}^{6} y_i^2 - \frac{1}{6}\sum_{i=1}^{6} 2x_iy_im - \frac{1}{6}\sum_{i=1}^{6} 2y_ib + \frac{1}{6}\sum_{i=1}^{6} x_i^2m^2 + \frac{1}{6}\sum_{i=1}^{6} 2x_imb + \frac{1}{6}\sum_{i=1}^{6} b^2$$

$$= \frac{1}{6}\sum_{i=1}^{6} y_i^2 - \left(\frac{1}{6}2\sum_{i=1}^{6} x_iy_i\right)m - \left(\frac{1}{6}2\sum_{i=1}^{6} y_i\right)b + \left(\frac{1}{6}\sum_{i=1}^{6} x_i^2\right)m^2 + \left(\frac{1}{6}2\sum_{i=1}^{6} x_i\right)mb + b^2$$

We see that this is a function of our variables m and b, since x_i and y_i are simply numbers from our data set. Evaluating the sums in this last expression, we have

$$\sum_{i=1}^{6} x_i = 20.9, \sum_{i=1}^{6} y_i = 21.1, \sum_{i=1}^{6} x_iy_i = 88.49, \sum_{i=1}^{6} x_i^2 = 87.07, \sum_{i=1}^{6} y_i^2 = 90.05.$$

so the the mean squared function can be written

$$MSE(m,b) = 15.00833 - 29.49667m - 7.03333b + 14.51167m^2 + 6.96667mb + b^2$$

To find the minimal possible value of MSE we find the partial derivatives of MSE and set them equal to zero:

$$\frac{\partial MSE}{\partial m} = -29.49667 + 29.02333m + 6.96667b = 0$$

$$\frac{\partial MSE}{\partial b} = -7.03333 + 6.96667m + 2b = 0$$

This second equation can be rearranged to give

$$b = \frac{7.03333 - 6.96667m}{2} = 3.51667 - 3.48333m$$

Plugging this into the other equation we have

$$0 = -29.49667 + 29.02333m + 9.63333(3.51667 - 3.48333m) = 4.38057 - 4.53273m$$

so that

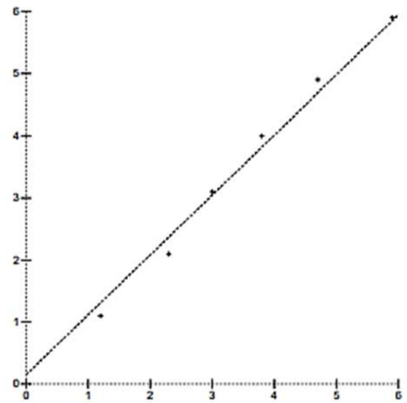
$$m = 0.96643$$

and

$$b = 3.51667 - 3.48333(0.96643) = 0.15028.$$

Thus, the best fitting line is

$$y = 0.96643x + 0.15028.$$



Another example for LMS

Suppose we want to predict the mileage of a car from its weight and age

Weight (x 100 lb) x_1	Age (years) x_2	Mileage
31.5	6	21
36.2	2	25
43.1	0	18
27.6	2	30

What we want: A function that can predict mileage using x_1 and x_2

Assumption: The output is a linear function of the inputs

$$\text{Mileage} = w_0 + w_1 x_1 + w_2 x_2$$

Learning: Using the training data to find the *best* possible value of **w**

Prediction: Given the values for x_1, x_2 for a new car, use the learned **w** to predict the `Mileage` for the new car

- Inputs are vectors: $\mathbf{x} \in \Re^d$
- Outputs are real numbers: $y \in \Re$

- We have a training set

$$D = \{ (\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_m, y_m)\}$$

- We want to approximate y as

$$y = f_w(x) = w_1 + w_2\, x_2 + \cdots + w_n\, x_n$$
$$= \mathbf{w}^T\mathbf{x}$$

$\mathbf{w}$ is the learned weight vector in $\Re^d$

For simplicity, we will assume that x_1 is always 1.

That is $\mathbf{x} = [1\ x_2\ x_3\ \ldots\ x_d]^T$

This lets makes notation easier

Question: How do we know which weight vector is the *best* one for a training set?

For an input ($\mathbf{x}_i$, y_i) in the training set, the ***cost*** of a mistake is

$$\left|y_i - \mathbf{w}^T\mathbf{x}_i\right|$$

Define the cost (or ***loss***) for a particular weight vector **w** to be

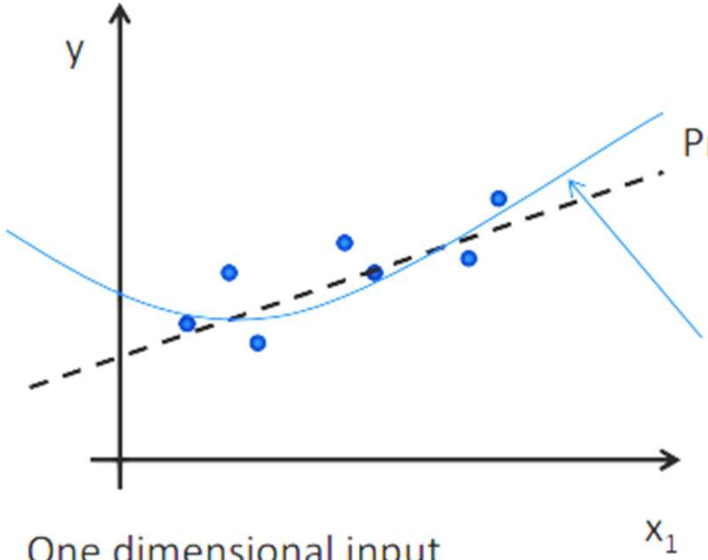
$$J(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{m}\left(y_i - \mathbf{w}^T\mathbf{x}_i\right)^2$$

Sum of squared costs over the training set

One strategy for learning: *Find the **w** with least cost on this data*

$$\min_{\mathbf{w}} \frac{1}{2}\sum_{i=1}^{m}\left(y_i - \mathbf{w}^T\mathbf{x}_i\right)^2$$

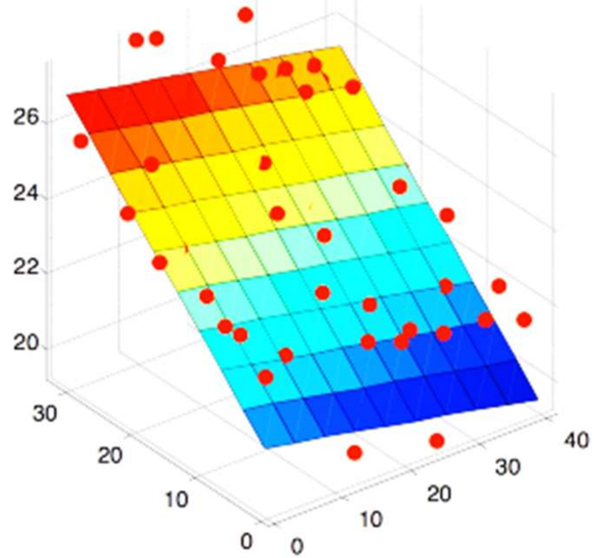
Learning: minimizing mean squared error



Predict using $y = w_1 + w_2 x_2$

The linear function is not our only choice. We could have tried to fit the data as another polynomial

One dimensional input



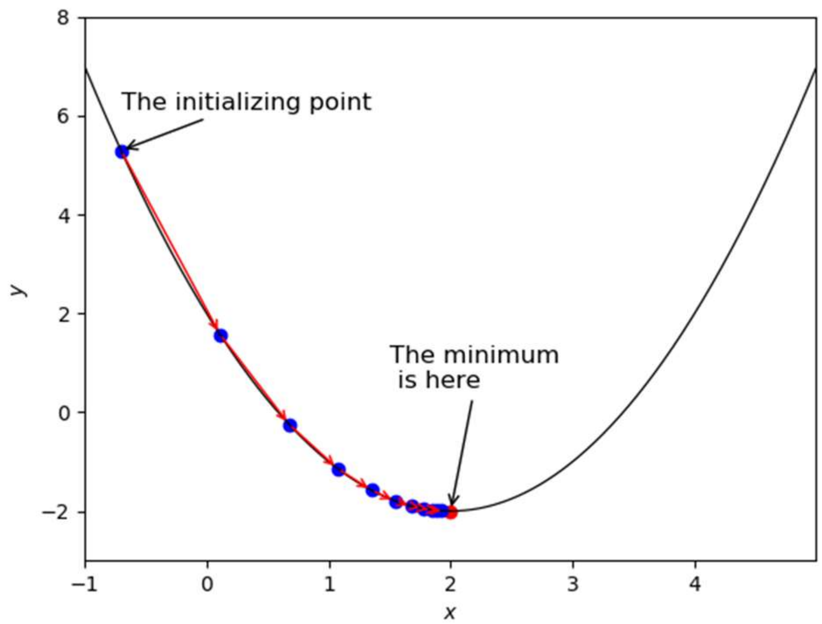
Two dimensional input

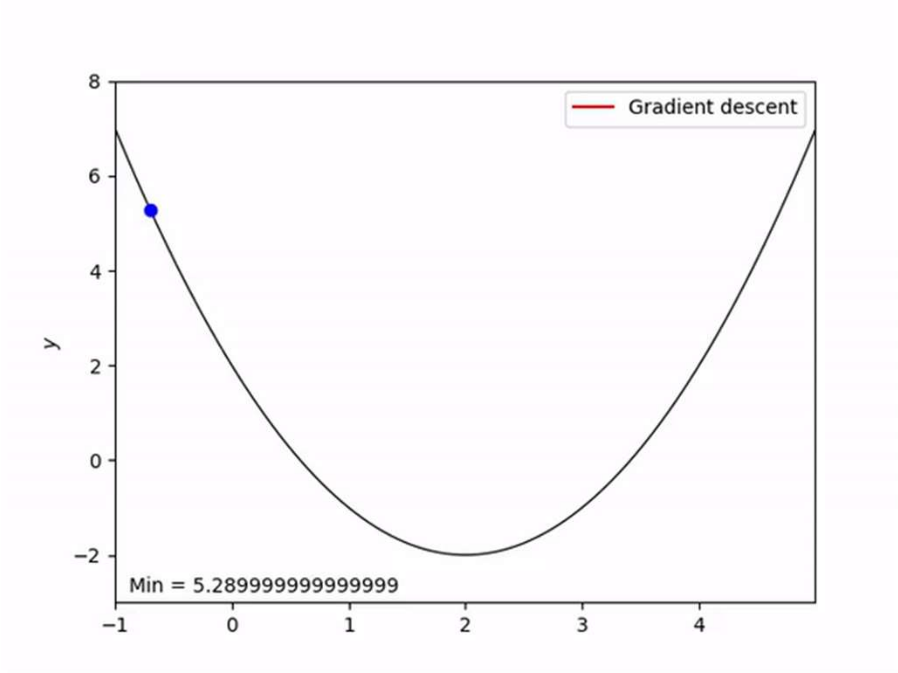
Predict using $y = w_1 + w_2 x_2 + w_3 x_3$

Gradient Descent Algorithm ~ Example

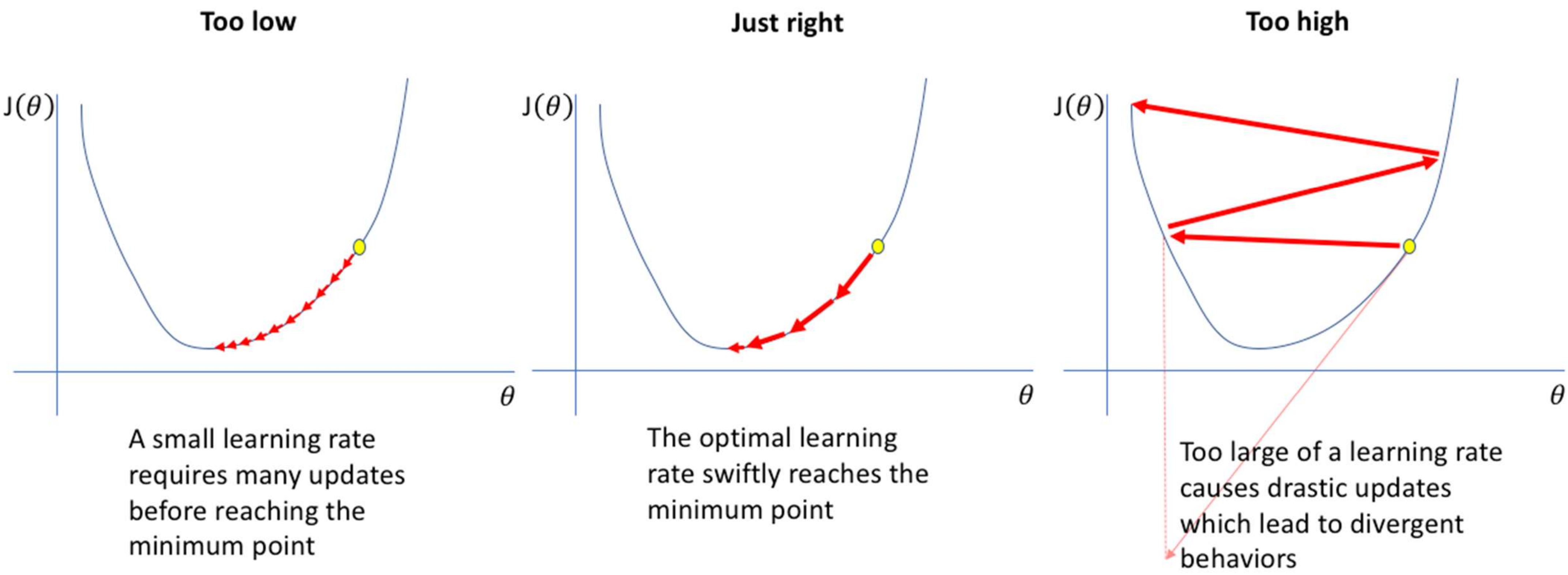
Q: Minimize $y = f(x) = x^2 - 4x + 2$ subject to −1 ≤ x ≤ 6

- In this optimization problem, our objective is a convex function with x∈R.
- Define $x_i := x_i - \Delta t \frac{\partial y}{\partial x}$ where Δt is timestamp (learning rate)





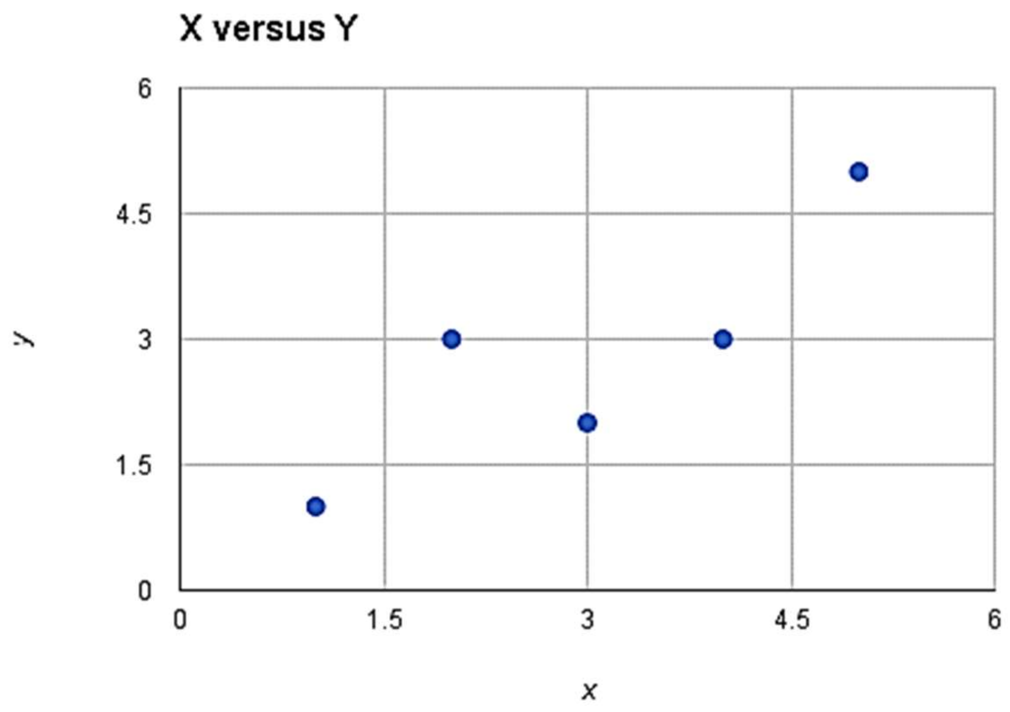
Impact of learning rate



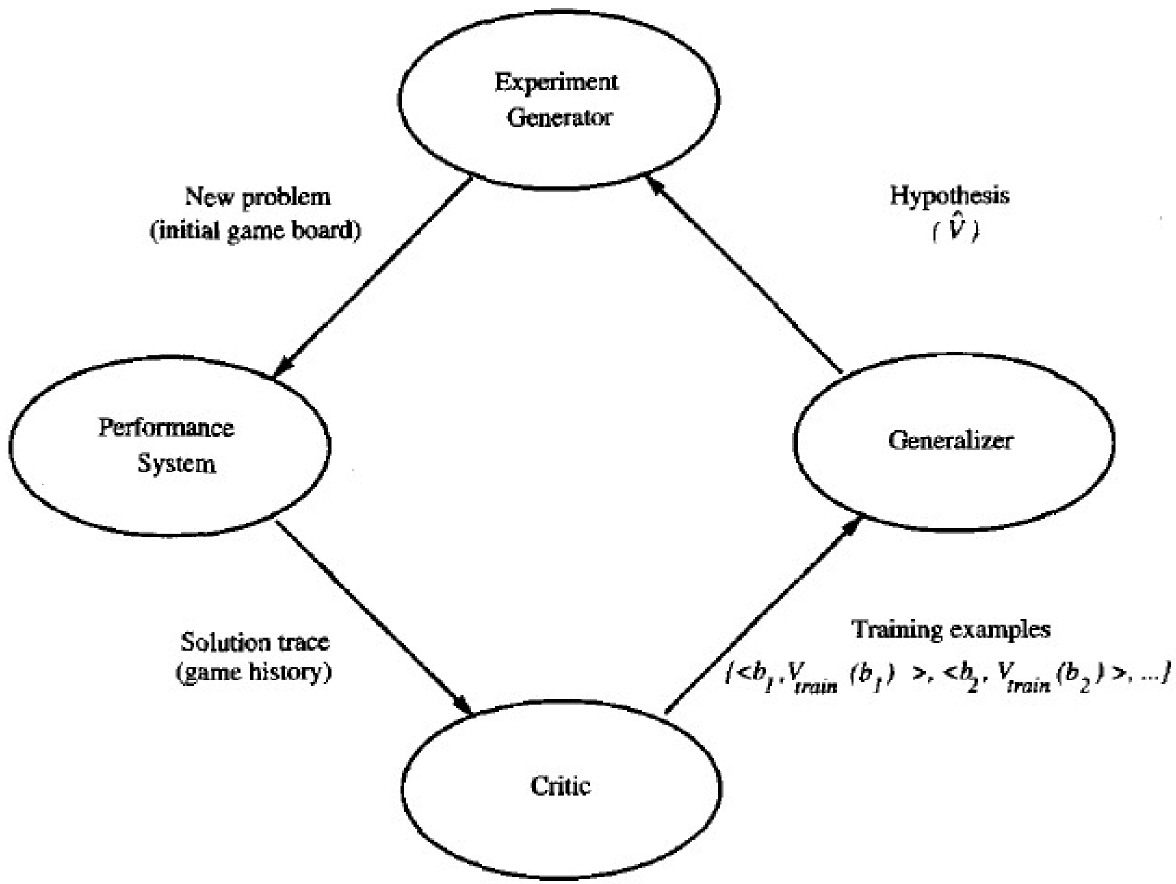
Solve

Consider the data given in table below. Perform linear regression using gradient descent assuming learning rate is 0.01

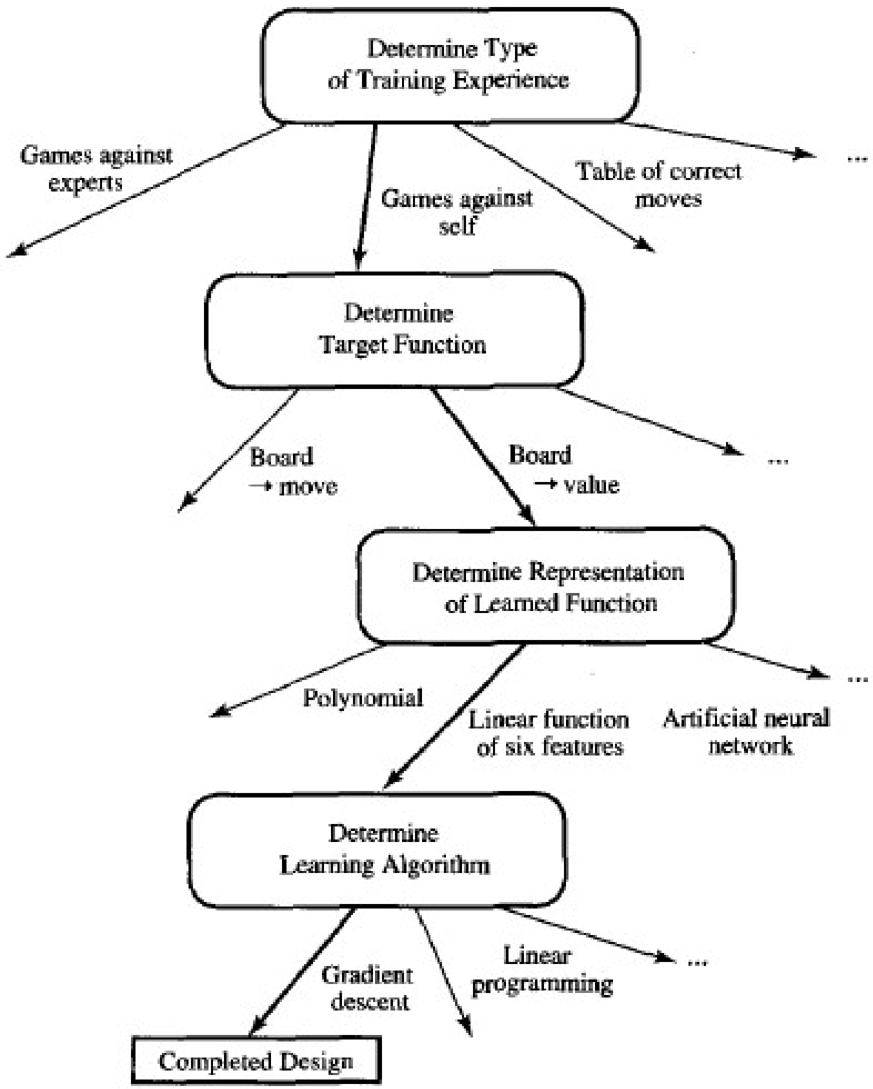
X No. of 'A' grades in I year	Y No. of 'A' grades in II year
1	1
2	3
4	3
3	2
5	5



- Final Design
  - Performance System
  - Critic
  - Generalizer
  - Experiment Generator



Design



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

Perspectives in Machine learning

Machine learning involves searching a very large space of possible hypothesis to determine the one that best fits the observed data and any prior knowledge held by the learner.

Various representations are:

- Linear functions
- Decision trees
- Artificial Neural networks

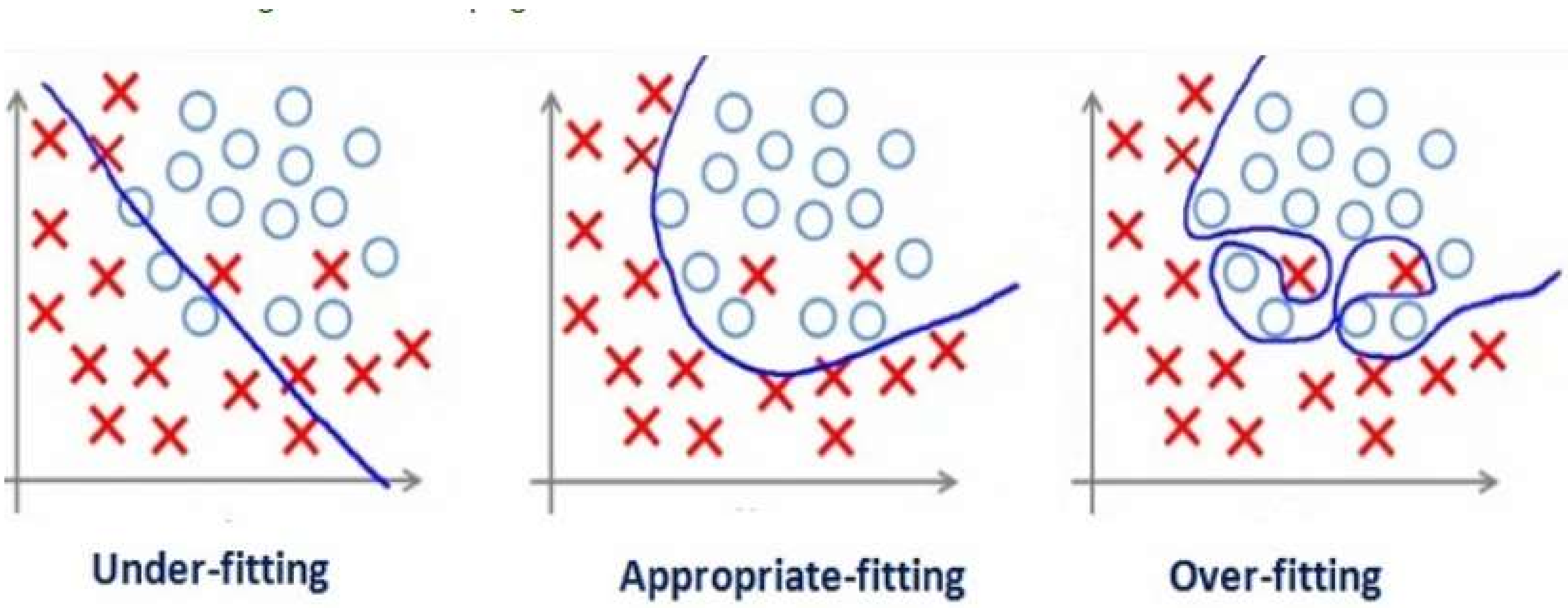
Issues in ML

- Amount of training data
- Missing and noisy feature values
- Appropriate feature selection
- Prior knowledge
- Strategy to choose next experience
- Learning rate
- Existence of algorithms to learn general target functions
- Overfitting and underfitting
- Bias variance effects

- **Overfitting:** A statistical model is said to be overfitted, when we train it with a lot of data. *That is, too much reliance on the training data.*
- **Underfitting:** A statistical model or a machine learning algorithm is said to have underfitting when it cannot capture the underlying trend of the data. *A failure to learn the relationships in the training data.*
- **High Variance:** model changes significantly based on training data.
- **High Bias**: assumptions about model lead to ignoring training data.
- Overfitting and underfitting cause poor **generalization** on the test set
- A **validation set** for model tuning can prevent under and overfitting

Effects of overfitting and underfitting

- Consider a two class category problem to visualize the effects of underfitting and overfitting



Avoiding overfitting issues

The commonly used methodologies are:

- **Cross- Validation:** A standard way to find out-of-sample prediction error is to use 5-fold cross validation.
- **Pruning:** Pruning is extensively used while building related models. It simply removes the nodes which add little predictive power for the problem in hand.
- **Regularization:** It introduces a cost term for bringing in more features with the objective function. Hence it tries to push the coefficients for many variables to zero and hence reduce cost term.

Exercise

Pick a machine learning task. Identify T,E and P. Propose a target function to be learned and a target representation. Discuss the tradeoffs considered in formulating this task.