PH-227
AI and Data Science

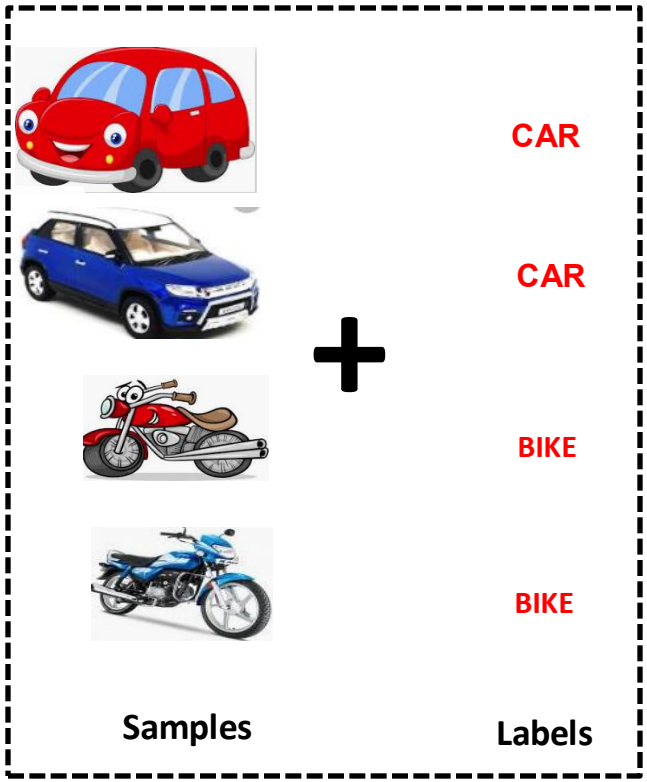
- Aftab Alam
- Email : aftab@iitb.ac.in
- Ext. : 5564 or 8564

TAs:

Yashowardhan, Divyansh, Matam, Peela, Piyush

Recap

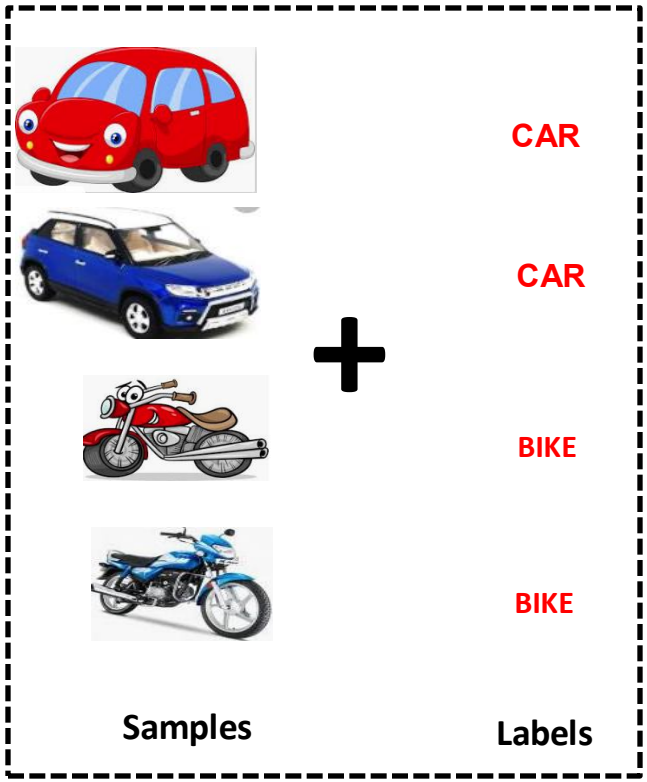
Supervised Learning



= Training Dataset

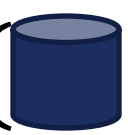
[In supervised learning, we need some thing called a “Labelled Training Dataset”]

Supervised Learning



= **Training Dataset**

Classification

$f($  ,  $)$ = CAR

[Given a labelled dataset, the task is to devise a function which takes the dataset, and a new sample, and produces an output value.]

- **If the possible output values of the function are predefined and discrete/categorical, it is called Classification**

Supervised Learning


CAR


CAR

+


BIKE

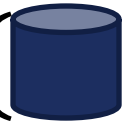

BIKE

Samples

Labels

= Training Dataset

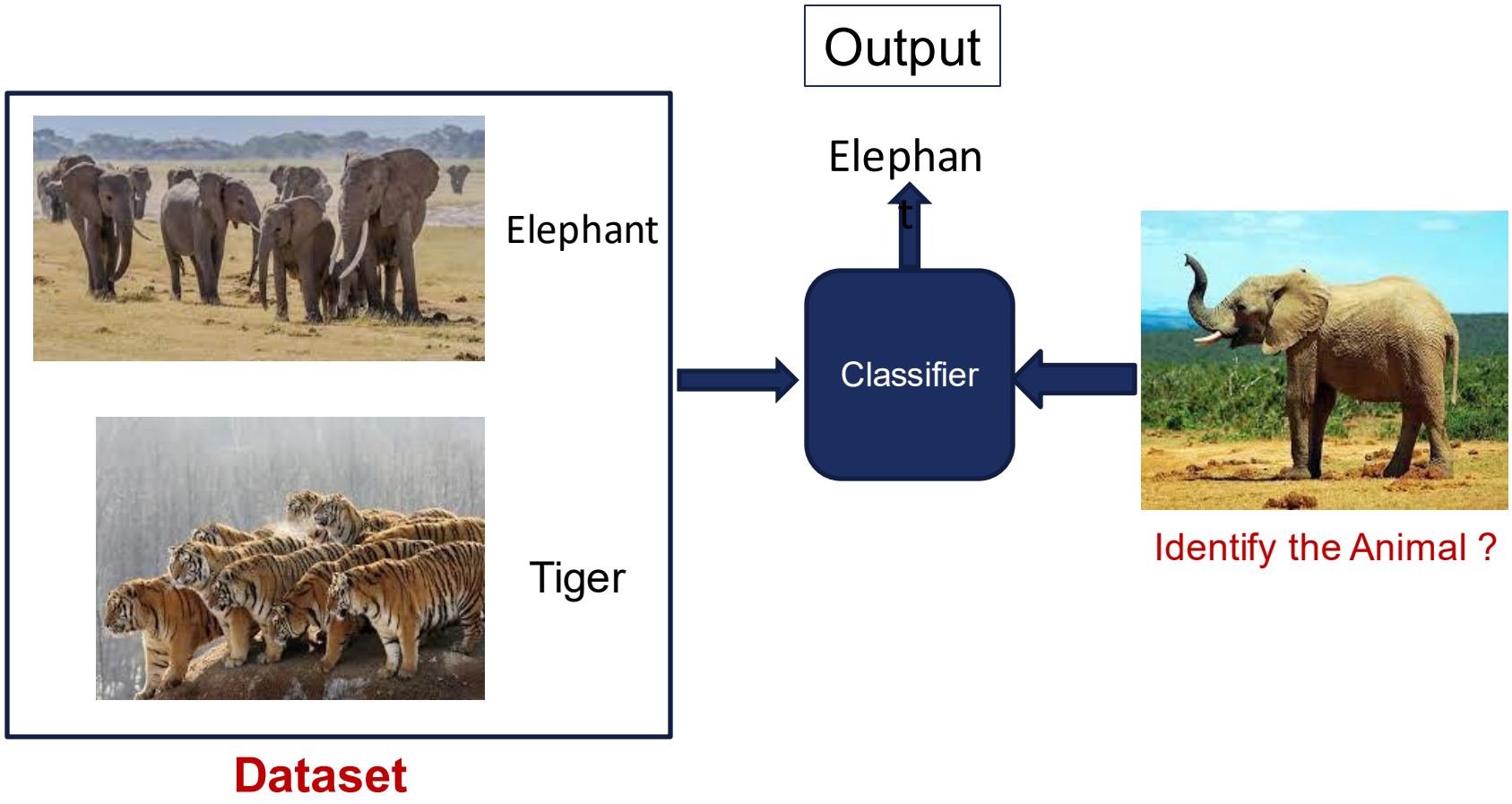
Classification

f( )= CAR

A predefined classification implies that, it will produce output only from the labels defined in the dataset.

e.g. even if we input a bus as a new sample to predict, it will produce either CAR of BIKE!!

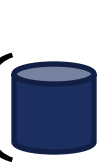
What is a Classifier?



Regression

Dataset

Regression

$f($  ,  $)= 20500.50$

- If the possible output values of the function are continuous real values, then it is called Regression

- Note that, both classification and Regression problems are supervised, because the decision depends on the characteristics of the ground truth labels or values present in the dataset, which we define as experience

Regression (Supervised Learning)

(Training) data

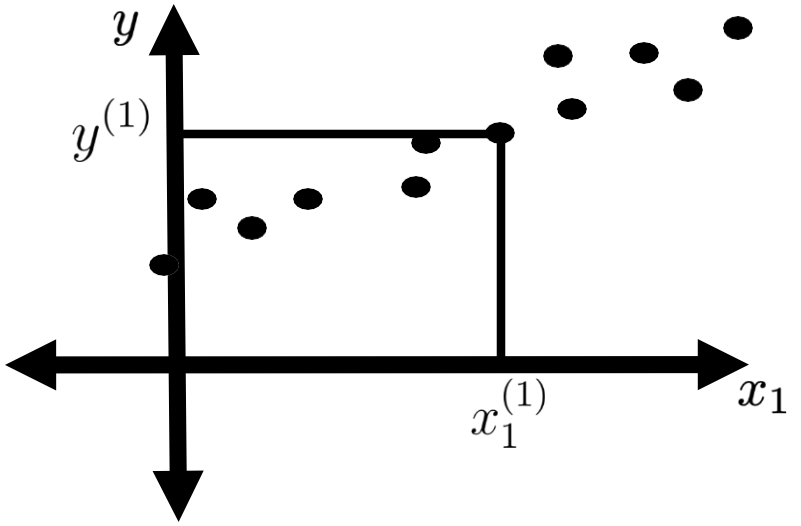
- *n* training data points
- For data point

$$i \in \{1, \ldots, n\}$$

  - Feature vector

$$x^{(i)} = (x_1^{(i)}, \ldots, x_d^{(i)})^\top \in \mathbb{R}^d$$

  - Label $y^{(i)} \in \mathbb{R}$



- Training data

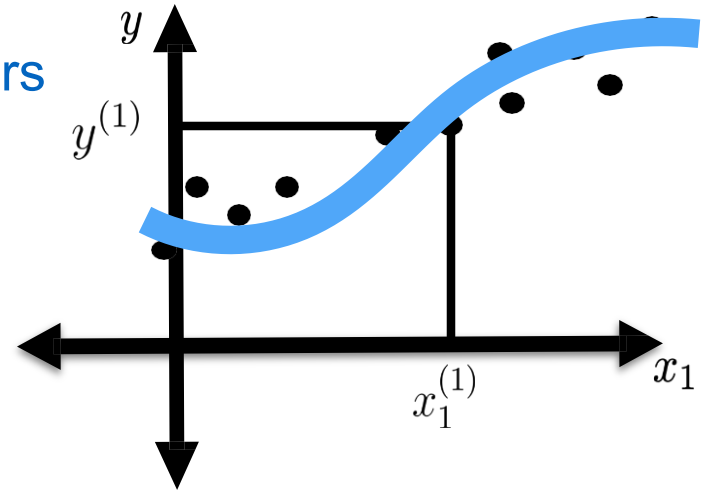
$$\mathcal{D}_n = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})\}$$

Regression (Supervised Learning)

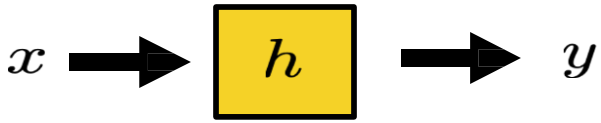
What are we looking for?

We want a "good" way to label new feature vectors

- How to label? Learn a hypothesis
- We typically consider a class of possible hypotheses



Input:
Feature vector



Output
Label

how well our hypothesis labels new feature vectors depends largely on how expressive the hypothesis class is

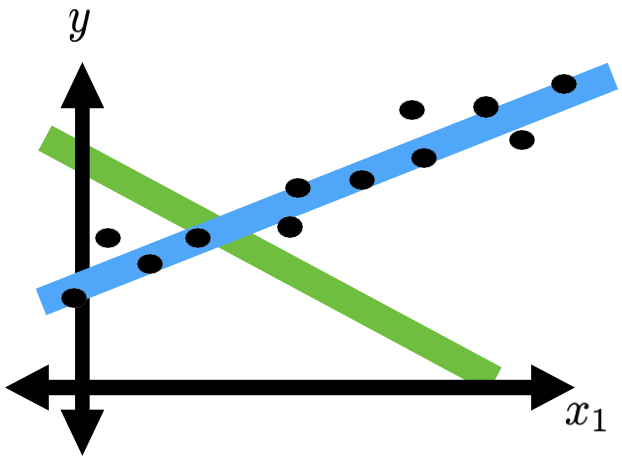
Regression (Supervised Learning)

Let us consider the class of linear regression

- Hypotheses take the form:

$$h(x; \underbrace{\theta, \theta_0}) = \theta^\top x + \theta_0$$

parameters to learn θ



- What we really want is to generalize to **future data**!
- What we do not want:
  - Model does not capture the input-output relationship (e.g., not enough data) ➔ **Underfitting**
  - Model too specific to training data ➔ **Overfitting**

Regression (Supervised Learning)

Goodness of the Hypothesis

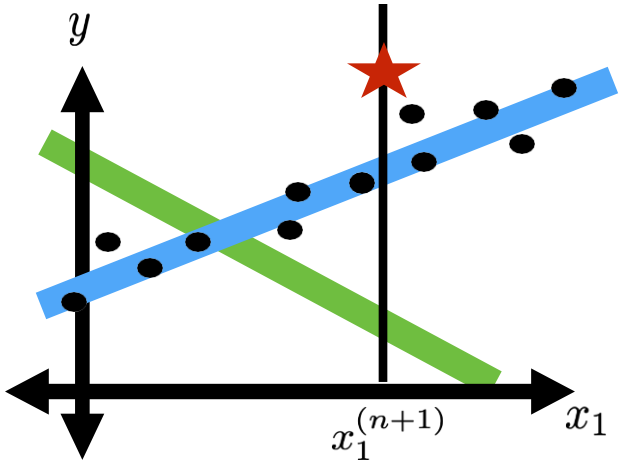
Hopefully predict well on future data

- How good is a regressor at one point?
  - Quantify the error using a loss function,

$$\mathcal{L}(h, y^{(i)})$$

  - Common choice: squared loss function

$$\mathcal{L}(h, y^{(i)}) = (h - y^{(i)})^2$$



h: guess, $y^{(i)}$: actual value

- Total training error:

$$\mathcal{E}_n(h; \quad) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}\left(h(x^{(i)}; \quad), y^{(i)}\right)$$

- Validation or Test error (n' new points):

$$\mathcal{E}(h) = \frac{1}{n'}\sum_{i=n+1}^{n+n'} \mathcal{L}\left(h(x^{(i)}), y^{(i)}\right)$$

Regression (Supervised Learning)

In a generalized d-dimensional problem

$$X = \begin{bmatrix} x_1^{(1)} & \cdots & x_d^{(1)} \\ \vdots & \ddots & \vdots \\ x_1^{(n)} & \cdots & x_d^{(n)} \end{bmatrix} \in \mathrm{R}^{n \times d} \qquad Y = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(n)} \end{bmatrix} \in \mathrm{R}^{n \times 1} \qquad \theta = \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_d \end{bmatrix} \in \mathrm{R}^{d \times 1}$$

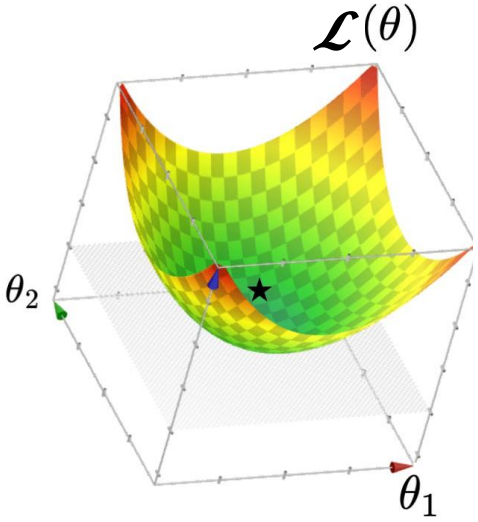
Squared Loss function:

$$\mathcal{L}(\theta) = \frac{1}{n}(X\theta - Y)^\top (X\theta - Y) \in \mathrm{R}^{1 \times 1}$$

Using matrix calculus and optimization:

$$\theta^* = (X^\top X)^{-1} X^\top Y$$

$$X^\top X \in \mathrm{R}^{d \times d} \qquad X^\top Y \in \mathrm{R}^{d \times 1}$$

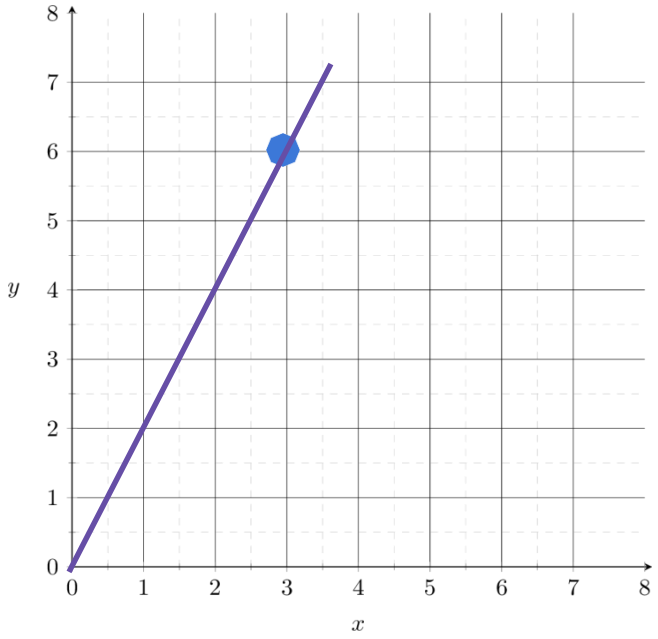


2D

Regression (1D example)

1D feature training data

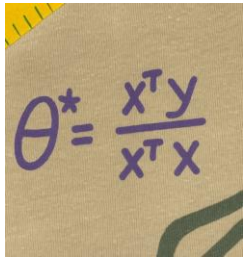
$(x, y) = (3, 6)$



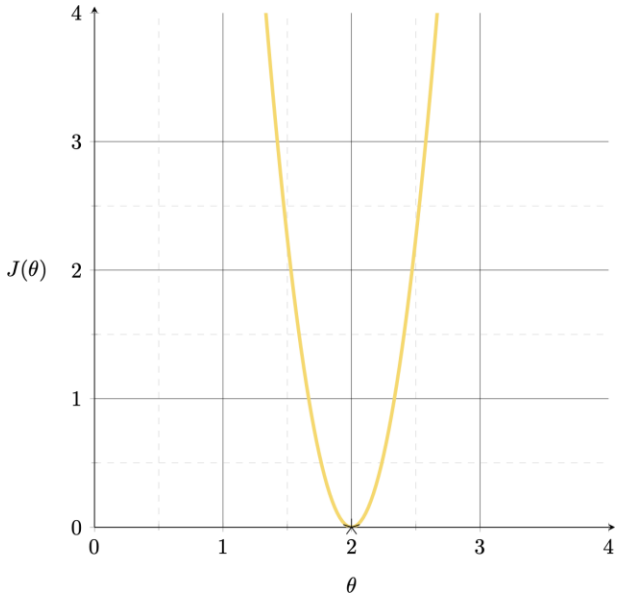
$\theta^* = (X^\top X)^{-1} X^\top Y$

$X = x = [3]$

$Y = y = [6]$



$\mathcal{L}(\theta) = (3\theta - 6)^2$

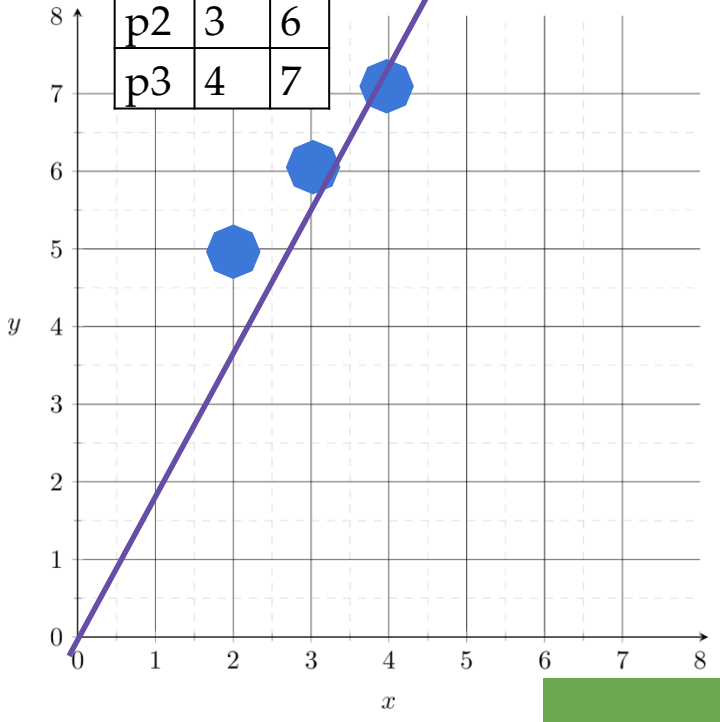


$\theta^* = (xx)^{-1}(xy) = \frac{xy}{xx} = \frac{y}{x} = \frac{6}{3} = 2$

Regression (Supervised Learning)

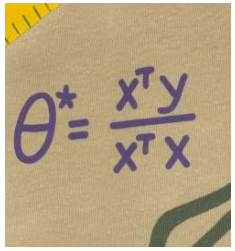
1D feature training data

	x	y
p1	2	5
p2	3	6
p3	4	7

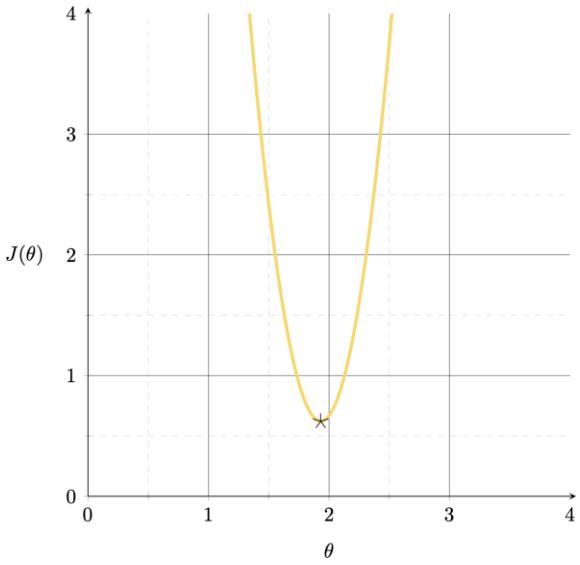


$$\theta^* = (X^\top X)^{-1} X^\top Y$$

$$X = \begin{bmatrix} 2 \\ 3 \\ 4 \end{bmatrix} \qquad Y = \begin{bmatrix} 5 \\ 6 \\ 7 \end{bmatrix}$$

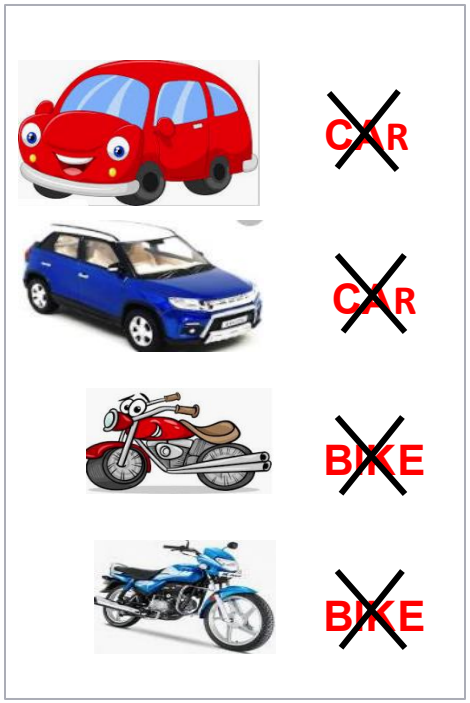


$$\mathcal{L}(\theta) = 1/3\ [(2\theta - 5)^2 + (3\theta - 6)^2 + (4\theta - 7)^2]$$



$$\theta^* = ([2\ 3\ 4] \begin{bmatrix} 2 \\ 3 \\ 4 \end{bmatrix})^{-1} [2\ 3\ 4] \begin{bmatrix} 5 \\ 6 \\ 7 \end{bmatrix} = \frac{X^\top Y}{X^\top X} = \frac{56}{29} \approx 1.93$$

Unsupervised Learning



Dataset

In the unsupervised learning, one does not need to know the labels or Ground truth values

Unsupervised Learning

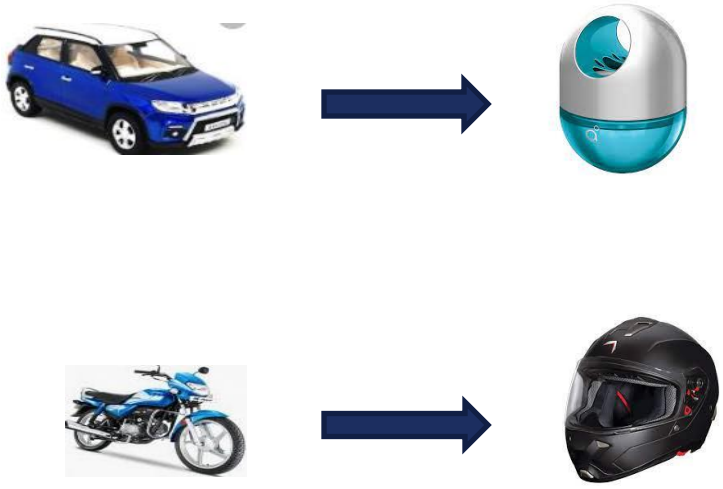
Dataset

Clustering similar objects together

The task is to identify the patterns like group the similar objects together

Unsupervised Learning

Dataset



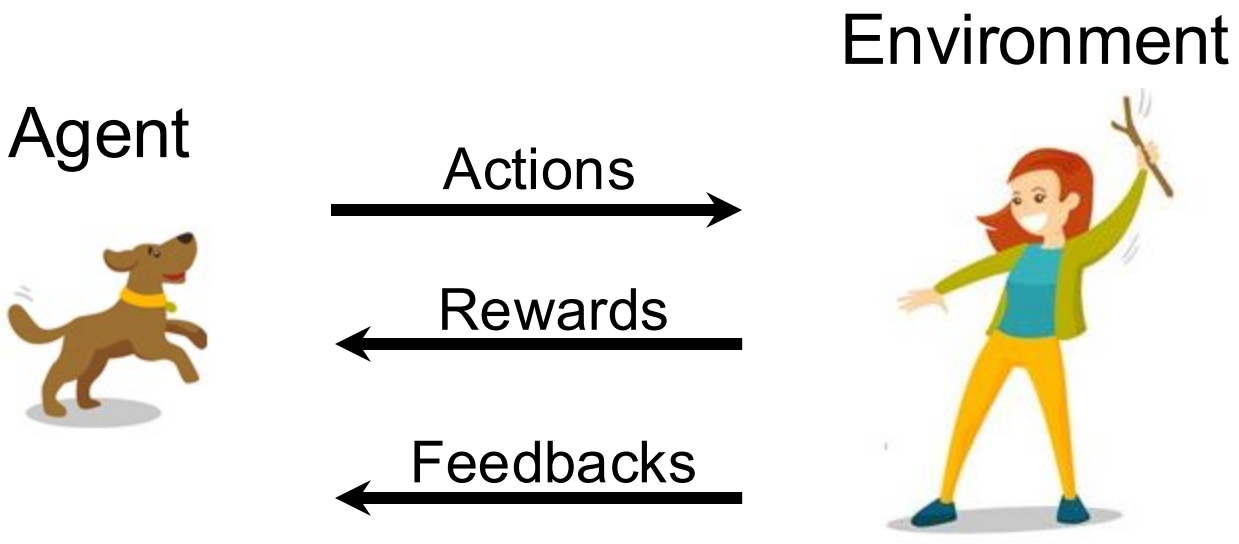
Associate certain similarity Rules

Few other examples of similarity rules

Dataset

Few other examples of similarity Rules

Reinforcement Learning



Another Example

Agent

Task

Environment

Reinforcement Learning

Punishment or –ve feedback

Reinforcement Learning

Baby learn from the Trials and Errors

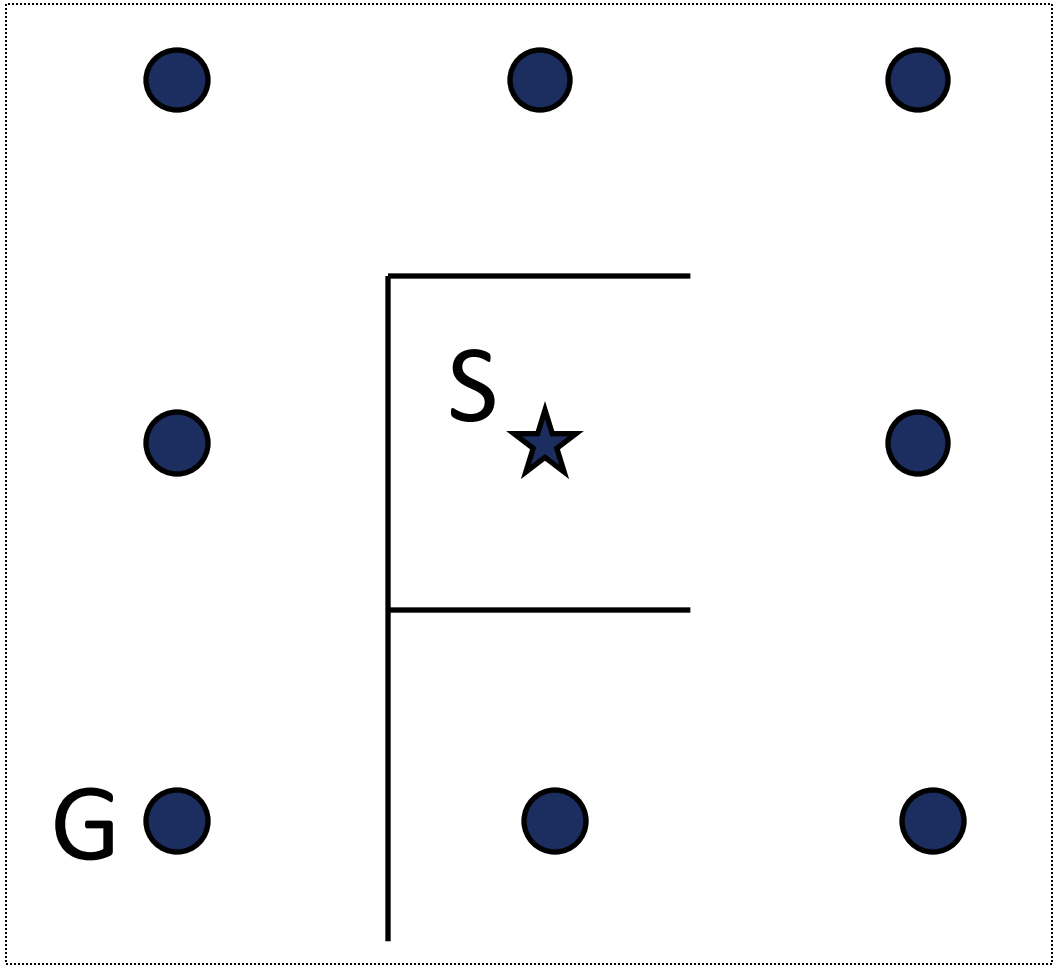
Reward or +ve feedback

Reinforcement Learning

More example (Reinforcement Learning

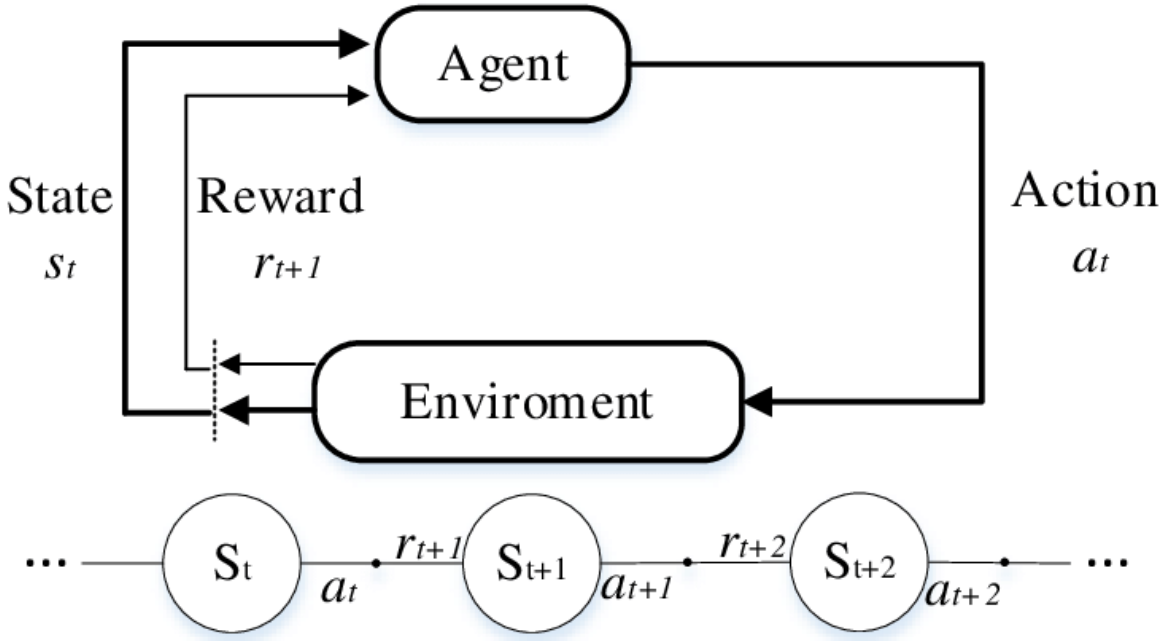
How did you learn cycling?

Reinforcement Learning (Board Game)



The objective is to design an optimal to start from S and reach the Goal (G)

Reinforcement Learning (in a nutshell)

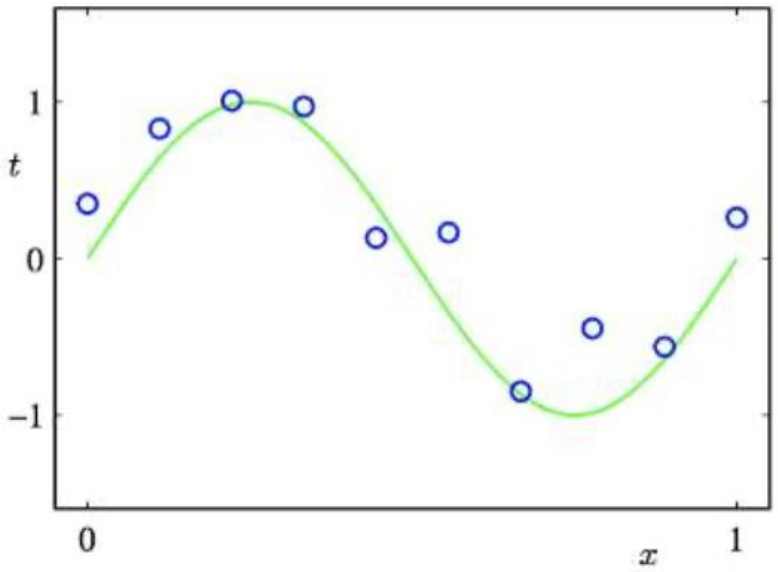


- **RL:** A class of learning problems in which an agent interacts with an unfamiliar, dynamic and stochastic environment
- **Goal:** Learn a policy to maximize some measure of long-term reward
- **Interaction:** Modeled as a Markovian Decision Process (MDP)

Applications of Reinforcement Learning

- Backgammon **(Tesauro, 1994)**
- Inventory Management **(Van Roy, Bertsekas, Lee, & Tsitsiklis, 1996)**
- Dynamic Channel Allocation **(e.g. Singh & Bertsekas, 1997)**
- Elevator Scheduling **(Crites & Barto, 1998)**
- Robocup Soccer **(e.g. Stone & Veloso, 1999)**
- Many Robots **(navigation, bi-pedal walking, grasping, switching between skills, ...)**
- Helicopter Control **(e.g. Ng, 2003, Abbeel & Ng, 2006)**
- More Applications http://neuromancer.eecs.umich.edu/cgi-bin/twiki/view/Main/SuccessesOfRL

Overfitting and Underfitting (Example)



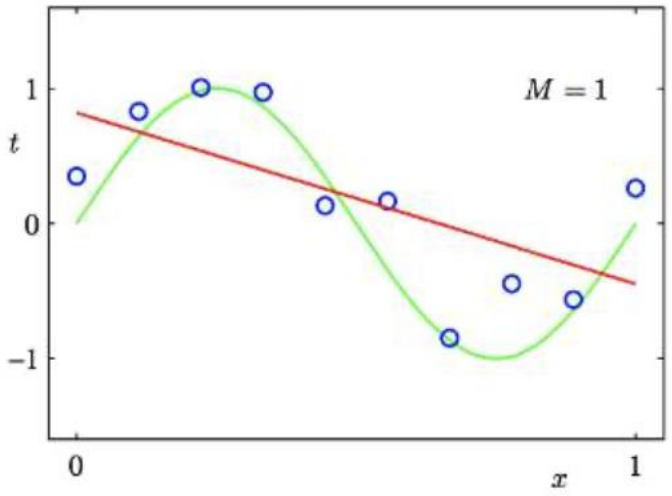
Blue circle: Raw data
Green line: Sin($2\pi x$)

- The goal is to find a model (or function) that fits the raw data well.
- But, we should be careful that the chosen model does not underfit or overfit !

Let us decide to fit a polynomial function of the following form,

$$y(x,w) = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + \ldots\ldots\ldots\ldots + w_M x^M$$

Overfitting and Underfitting (Example)

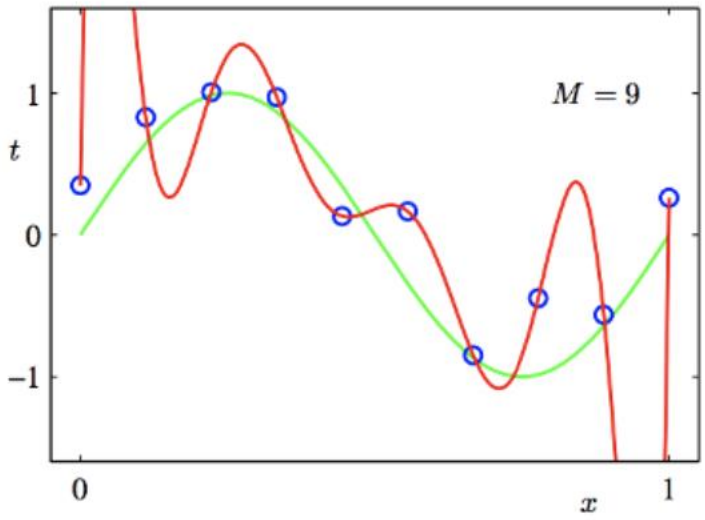


Blue circle: Raw data
Green line: Sin($2\pi x$)

$$y(x,w) = w_0 + w_1 x$$

- The model (with M-1) does not fit the data well, since it disagrees on many points in the training raw dataset.
- This is an example of underfitting because it is less expressive than the true function.

Overfitting and Underfitting (Example)

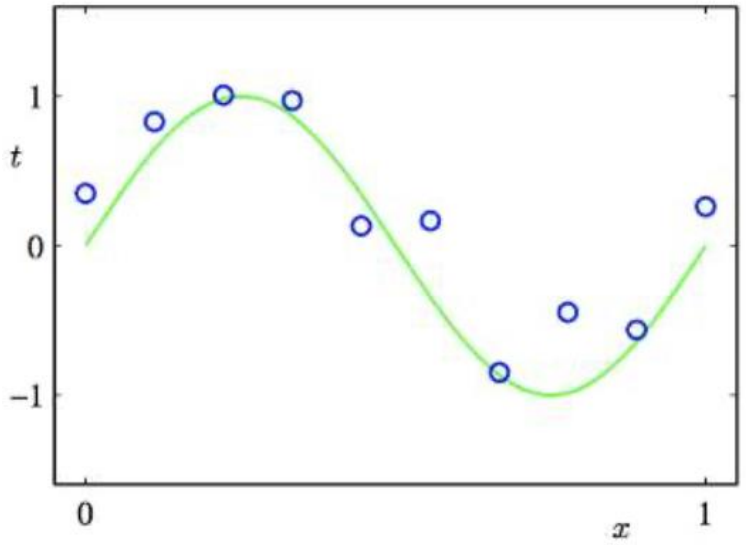


Blue circle: Raw data
Green line: Sin($2\pi x$)

$$y(x,w) = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + \ldots\ldots\ldots\ldots + w_9 x^9$$

- Unlike the straight line (M=1), this function fits each of the points in the training set
- However, it does not approximate the function well at points not in the training set, indicating that it is overfitting.

Cross Validation (as a solution to overfitting)



Blue circle: Raw data
Green line: Sin($2\pi x$)

- Cross validation detects overfitting by demonstrating how a single model+(parameter choices)+(training procedure) performs on several distinct test sets.
- It can be used to reduce overfitting by guiding model, parameter or training choices that lead to less overfit models, but it does not reduce overfitting for a single model+(parameter choices)+(training procedure) .
- The best way to learn something is to do it yourself.

Mean and Standard deviation