ML Supervised Learning 3 by ambedkar@IISc

- Introdcution to Supervised Learning
- Some foundational aspects of ML

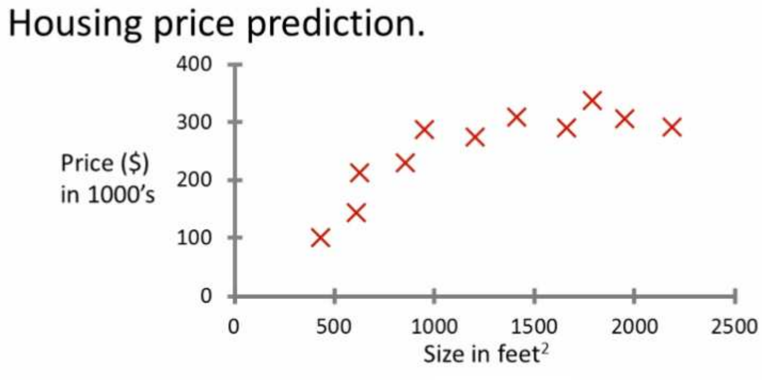
Rewind

So far...

- General introduction to ML and what it can and cannot
- Some understanding of what is data and model
- Machine learning work-flow (very important)
- How can we construct simple classifiers using "distance"
- Introduction to Bayes decision theory

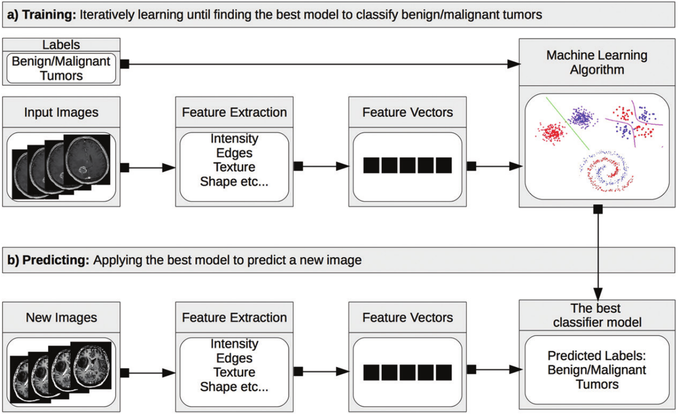
Supervised Learning

Regression: Example



Supervised Learning: Predicting housing prices

Classification: Example



Supervised Learning in Action for Medical Image Diagnosis[1]

[1] Image is taken from Erickson et al, Machine Learning for Medical Imaging, Radio Graphics, 2017

Who supervises "Learning"?

Answer: Ground-truth or labels.

- In supervised learning along with input feature vector x there a groundtruth or response y associated with it.
  - If y takes only two values or at most finitely many values it is a classification problem
  - If y takes any real number it is a regression problem
- Aim is to build a system f (or a function) such a way that
  - given x predict y as accurately as possible

How do we measure the accuracy?

A set of labeled training examples are given.

$$\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$$

- Each x_n can be an image or a document or a time series etc.
- Each x_n it self is D-dimensional vector. That is each x_n is of the form

$$x_n = (x_{n1}, x_{n2}, \ldots, x_{nD})$$

- $x_{n1}, x_{n2}, \ldots, x_{nD}$ are called features of x_n
- Note that we represent x_n either as a vector or a column matrix.

Output: y_n denotes a label or ground-truth or response

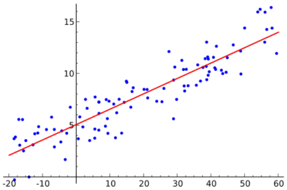
Supervised Learning: Setting (Cont...)

Objective: To learn a function f_θ that:

- Closely mimics the examples in training set $(f_\theta(x_n) \approx y_n)$, *i.e.*, has low *training error*
- Generalizes to unseen examples, *i.e.*, has low *test error*

θ **refers to** ***learnable parameters*** **of the function** f_θ

- **Objective:** To learn a function mapping input features x to scalar target y
- Linear regression is the most common form - assumes that f_θ is linear in θ



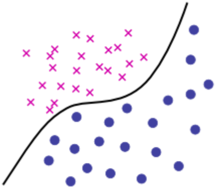
Example - Linear Regression

- **Examples:**
  - Predicting temperature in a room based on other physical measurements
  - Predicting location of gaze using image of an eye
  - Predicting remaining life expectancy of a person based on current health records
  - Predicting return on investment based on market status

[1] Image source: `https://en.wikipedia.org/wiki/Linear_regression`

Supervised Learning - Classification

- **Objective:** To learn a function that maps input features x to one of the K classes
- The classes may be (and usually are) unordered



Example - Classification

- **Examples:**
  - Classifying images based on objects being depicted
  - Classifying market condition as favorable or unfavorable
  - Classifying pixels based on membership to object/background for segmentation
  - Predicting the next word based on a sequence of observed words

[1] Image source: `https://www.hact.org.uk`

Some popular techniques:

- Logistic regression
- Random forests
- Bayesian logistic regression
- Support vector machines
- Neural networks
- etc.

Supervised Learning Setup: Notation

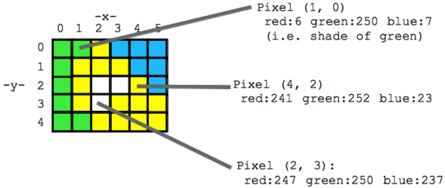
- The number of data samples that are available to us is N
- That is the samples are $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$
  - For example, $x_1, x_2, \ldots, x_N$ denote medical images and,
  - $y_1, y_2, \ldots, y_N$ represent ground-truth diagnosis say -1 or $+1$.
- Note that the data can be noisy
  - Scanner itself may introduce this noise
  - Doctors can make some mistake in their diagnosis

Supervised Learning Setup: Dimension

- Dimension is the size of the input data i.e x_n we denote this by D
- We write $x_n = (x_{n1}, \dots, x_{nD}) \in \mathbb{R}^D$
  - If a grey scale image size is say 16×16 then $D = 16 \times 16$
  - If it is RGB then $D = 16 \times 16 \times 3$ and each x_{nd} takes value between 0 and 255.
- The dimension of $x_1, x_2, \dots, x_N$ is typically very high
- Why?

Supervised Learning Setup: Dimension (Contd...)

- Number of pixels in an image 800 pixel wide, 600 pixels high: $800 \times 600 = 480000$. Which is 0.48 megapixels
- Typically digital images are $4 - 20$ megapixels



Pixels in RGB images[2]

- Now what is the dimension of 800×600 image?

[2] Taken from web

Supervised Learning Setup: Dimension (Contd...)

- Note that in some applications dimension of each sample can be varying, for example:
  - sentences in text
  - protein sequence data
- What about the response y?
  - Dimension of y is much much less than x
  - y can be structured and it leads to structure prediction learning
- A major issue in machine learning: **High dimensionality of data**

Supervised Learning: Formal Definition

Problem: Given the data $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$, aim is to find a function.

$$f : \mathcal{X} \to \mathcal{Y}$$

that approximate the relation between X and Y.

- There are small letters, capitol letters, script letters. What are they?
- X and Y denotes the random variables and $\mathcal{X}$ and $\mathcal{Y}$ denotes the sets from where X and Y take values.

Random Variables? Why are we talking about probability here?

Some Foundational aspects of Machine Learning

On Statistical Approach to Machine Learning

Assumption behind the statistical approach to Machine Learning:

Data is assumed to be sampled from a underlying probability distribution

On Statistical Approach to Machine Learning (contd...)

- Suppose we are given N samples $x_1, \ldots, x_N$
- Our assumption is that there is a hypothetical underlying distribution P from which these samples are drawn
  - The problem is that we do not know this distribution
  - Some machine learning algorithms try to estimate this distribution, some try to solve problems without estimating this distribution

On Statistical Approach to Machine Learning (contd...)

- Recall, class conditional densities $P(x|y_1)$ and $P(x|y_2)$
  - In the Bayes classifier uses these distributions
  - We are given only data, from which we need to estimate these distributions (How?)
    - Maximum likelihood estimation
    - Maximum a posteriori estimation

On Statistical Approach to Machine Learning (contd...)

How complicated this underlying distribution can be?

Loss Function

We need some guiding mechanism that will tell us how good our predictions are given an input.

- $\ell(y, f(x))$ denotes the loss when x is mapped to $f(x)$, while the actual value is y.

Note

- ℓ and f are specific to the problems and a method.
- For example, $\ell(.)$ can be a squared loss and $f(x)$ is linear function i.e $f = w^\intercal x$.

Learning as an optimization

Objective

Given a loss function ℓ, aim is to find f such that,

$$L(f) = \mathsf{E}_{(x,y)\sim P}[\ell(Y, f(X))]$$

is minimum

- Here X and Y are random variables.
- L is the true loss or expected loss or Risk.
- As we mentioned before we assume that the data is generated from a joint distribution $P(X, Y)$.
- When we try to learning this distribution it is called generatie modelling leads to so called Generative AI.

Learning as an optimization: Making Sense

- Remember, broad aim of ML is to understand a phenomenon or (and) solve some downstream problems related to it
- When we make assumptions about existence of P that means assume that we capture the phenomenon by P
- The available data represents the partial information that we have about the phenomenon or P
- That is data is nothing but samples drawn from distribution P

Diversion: Probability Basics

- Random variable is nothing but a function that maps outcome to a number
  - Consider a coin tossing experiment: Outcomes are `H` and `T`
  - Random variable X can map `H` to 1 and can map `T` to 0
- Now let us assign probabilities
  - Suppose $P(X = 1) = \frac{1}{4}$ and $P(X = 0) = \frac{3}{4}$
  - That is probability mass function of X is $(\frac{1}{4}, \frac{3}{4})$
- Let us calculate expectation of a random variable

$$\mathsf{E}_P X = \sum_{i=1}^{2} x_i p_i = 1\left(\frac{1}{4}\right) + 0\left(\frac{3}{4}\right)$$

Empirical Risk

Problem: We cannot estimate the true loss as we do not know P.

Some Relief: But we have some samples that are drawn from P.

Empirical Risk

Instead of minimizing the true loss find f that minimizes empirical risk

$$L_{emp}(f) = \frac{1}{N} \sum_{n=1}^{N} \ell(y_n, f(x_n))$$

$$i.e. \qquad f^* = \arg\min_f \frac{1}{N} \sum_{n=1}^{N} \ell(y_n, f(x_n))$$

Empirical Risk

$$L_{emp}(f) = \frac{1}{N}\sum_{n=1}^{N} \ell(y_n, f(x_n))$$

$$i.e. \qquad f^* = \arg\min_f \frac{1}{N}\sum_{n=1}^{N} \ell(y_n, f(x_n))$$

- Here $\ell(y_n, f(x_n))$ is the per sample loss
- $L_{emp}(f)$ is the overall loss given the data $\{(x_n, y_n)\}_{n=1}^{N}$
- N is the number of samples and we need "reasonably many" samples so that Empirical Risk is close to the True Risk
- Why do we need Empirical Risk to be closer to the True Risk?

Generalizing Capacity

How well the learned function work on the unseen data?

- We want f not only work on the training data but also it should work on the unseen data.
- For this the general principle:

The model should be simple

- Regularizer

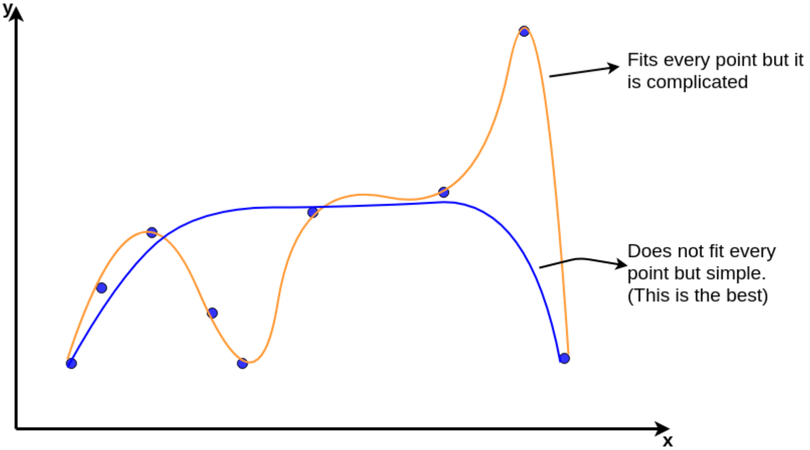
$$f^* = \arg\min_f \frac{1}{N}\sum_{n=1}^{N} \ell(y_n, f(x_n)) + \lambda R(f)$$

  - λ controls how much regularization one needs.
  - R measures complexity of f.
  - This is regularized risk minimization.

Generalizing Capacity(cont...)

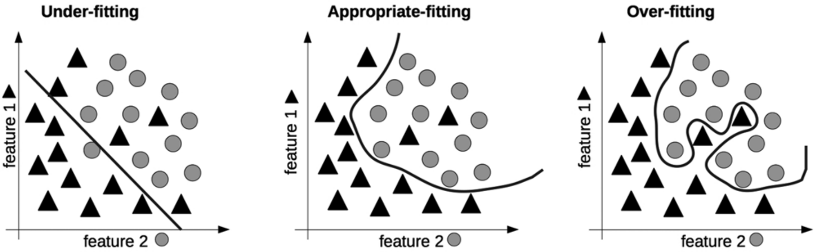
- What we want to achieve.
  - Small empirical error on training data, and at the same time,
  - f needs to be simple.
- There is a trade off between these two goals
  - λ is a hyperparameter that tries to achieve this.

Generalizing Capacity(cont...)



The blue curve has better generalization capacity. The orange curve overfits the data

Generalizing Capacity(cont...)



Learning as the Optimization

We have the following optimization problem "find f such that ... "

- Is it any f ?
- No, The choice f cannot be from a arbitrary set.
- First we fix $\mathcal{F}$: the set of all possible functions that describe relation between X and Y given training data $\{(x_n, y_n)\}_{n=1}^{N}$
- Now our objective is

$$f^* = \arg\min_{f \in \mathcal{F}} \sum_{n=1}^{N} \ell(y_n, f(x_n)) + \lambda R(f)$$

- For example, If $\mathcal{F}$ is set of all linear functions then we call it linear regression.

What we have learned?

- Beware! there are some underlying assumptions and approximations
- There is no rule book. Practitioners have to make some decisions while designing the algorithms and methods
- What is the Challenge? We want our algorithms work well on the unseen data.
- How do we evaluate performance of ML algorithms?