# Stats 231A – Machine Learning

University of California, Los Angeles

Duc Vu

Fall 2021

This is stats 231A – an intro graduate level course on **Pattern Recognition and Machine Learning** taught by Professor Wu. We meet weekly on TR from 3:30 pm to 4:45 pm for lecture. Other course notes can be found at my blog site. Please let me know through my email if you spot any typos in the note.

## Contents



## List of Theorems

## List of Definitions

# §1 Lec 1: Sep 28, 2021

## §1.1 Modes of Learning

Table 1: Supervised Learning

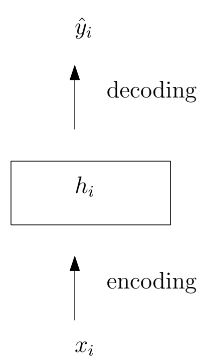
| | | |
|---|---|---|
| 1 | | |
| $\vdots$ | | |
| $i$ | $x_i$ | $y_i$ |
| $\vdots$ | | |
| $n$ | | |

Table 2: Unsupervised Learning

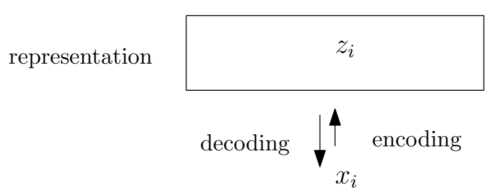
| | | |
|---|---|---|
| $i$ | $x_i$ | ? |

Representation Learning:

supervised




This is known as thought vector, features, base learners, or hidden variables.




Each argument in unsupervised learning is known as latents, code, embedding. The decoding part also requires generative model (auto-encoder).
Reinforcement Learning

$$s_0 \to \ldots \to \underbrace{s_t \xrightarrow{a_t}}_{r_t(\text{reward})} s_{t+1} \underbrace{\xrightarrow{a_{t+1}} \ldots}_{r_{t+1}+\ldots}$$

where $s$ represents state and $a$ stands for action and

$$\begin{aligned} \text{policy}: &\ \pi(a|s) \\ \text{value}: &\ v(s) = E\left(r_t + r_{t+1} + \ldots | s_t = s\right) \end{aligned}$$

Supervised Learning: Consider regression problems where $y_i$ is continuous or in classification where $y_i$ is categorical binary $(+/-, 1/0)$

- Regression:

$$\begin{aligned} y_i &\sim N(s_i, \sigma^2) \\ s_i &= f(x_i) \end{aligned}$$

- Classification:

$$\begin{aligned} p_i = p_r\left(y_r = 1|x_i\right) &= \frac{e^{s_i}}{1+e^{s_i}} \\ &= \frac{1}{1+e^{s_i}} = \text{sigmoid}(s_i) \end{aligned}$$

Before logistic regression, we also have perceptron (non-probabilistic)

$$\hat{y}_i = \text{sign}(s_i) = \begin{cases} 1 \text{ if } s_i \geq 0 \\ 0 \text{ if } s_i < 0 \end{cases}$$

Consider the linear model

$$s_i = \beta_0 + \sum_{j=1}^{p} \beta_j X_{ij}$$

where $\beta_0$ is the bias and $\beta_1, \ldots, \beta_p$ are the connection weights. We can use this for linear regression.
One hidden layer:

$$\begin{aligned} s_i &= f(x_i) \\ &= h_i^\top \beta \\ &= h(x_i)^\top \beta \end{aligned}$$

We can divide $R$ into different partitions where $h_{ik} = 1(x_i \in R_k)$ and $h_{ik} = \text{tree}$.

$$\begin{gathered} \text{classification tree} \longrightarrow \text{adaboost} \\ \text{regression tree} \longrightarrow XGB \end{gathered}$$

Kernel:

$$k(x, x') = \langle h(x), h(x') \rangle$$

where $k$ is explicit.
Two Layer Neural Network:

$$\begin{aligned} s_i &= \sum_{k=1}^{d} \beta_k h_{ik} \\ h_{ik} &= r\left(\sum_{j=1}^{p} d_{kj} x_{ij}\right) \\ &= \text{Relu}\left(\sum_{j=1}^{p} d_{kj} x_{ij}\right) = \max\left(0, \sum_{j=1}^{p} d_{kj} x_{ij}\right) \end{aligned}$$

1D:

$$s = \beta_0 + \sum_{k=1}^{d} \beta_k \max\left(0, x - a_k\right)$$

This can be very flexible and it can be used to approximate any nonlinear function (by piecewise linear function/line). For $\beta_k$ (curvature),

$$\underbrace{|\beta|_{l_2}^2}_{\text{smoothness}} = \sum_{k=1}^{d} \beta_k^2$$

2D:

$$h_k = \max\left(0, \alpha_{k_1} x_1 + \alpha_{k_2} x_2 - b_k\right)$$