CS260: Machine Learning Algorithms

Lecture 1: Overview

Cho-Jui Hsieh
UCLA

Jan 7, 2019

Course Information

- Website: `http://web.cs.ucla.edu/~chohsieh/teaching/CS260_Winter2019/main.html`
- My office: EVI 284
- Office hours: Wednesday 11am–noon
- Online office hour: TBD
- TA: Patrick Chen (patrickchen@g.ucla.edu)
- TA for online course: Minhao Cheng (mhcheng@ucla.edu)

Course Information

- There is no textbook. Most of the topics are covered in
  "Deep Learning" (by Goodfellow, Bengio, Courville)

Course Information

- There is no textbook. Most of the topics are covered in

  "Deep Learning" (by Goodfellow, Bengio, Courville)

- Part I (basic concepts):

  Linear models (regression, classification, clustering, dimension reduction)

  Basic learning theory (overfitting, regularization)

- Part II (Nonlinear models):

  Kernel methods

  Tree-based methods

  Deep networks

  Applications in computer vision and NLP

Grading Policy

- Midterm exam (30%)
- Homework (30%)
  - 3 homeworks
- Final project (40%)

Final project

- Group of ≤ 4 students.
- Work on some research projects:
  - Solve an interesting problem
  - Develop a new algorithm
  - Compare state-of-the-art algorithms on some problems
  - $\cdots$
- I'll recommend some topics in the course. Feel free to discuss with me in advance.

Machine Learning: Overview

From learning to machine learning

- What is learning?

$$\text{observations} \rightarrow \text{Learning} \rightarrow \textit{Skill}$$

- Skill: how to make decision (action)
  - Classify an image
  - Translate a sentence from one language to another
  - ...

From learning to machine learning

- What is learning?

$$\text{observations} \rightarrow \text{Learning} \rightarrow \textit{Skill}$$

- Skill: how to make decision (action)
  - Classify an image
  - Translate a sentence from one language to another
  - ...
- Machine learning:

$$\text{data} \rightarrow \text{Machine Learning} \rightarrow \text{Skill (decision rules)}$$

Automatic the learning process!

Credit Approval Problem

- Customer record (features):

age	23 years
gender	female
annual salary	NTD 1,000,000
year in residence	1 year
year in job	0.5 year
current debt	200,000

- To be learned:

  "Should we approve the credit card application? "

Credit Approval Problem

- Customer record (features):

age	23 years
gender	female
annual salary	NTD 1,000,000
year in residence	1 year
year in job	0.5 year
current debt	200,000

- To be learned:

  "Should we approve the credit card application? "

- Data: A collection of feature-label pairs:

$$(\text{customer1 feature}, \text{Yes}), (\text{customer2 feature}, \text{No}), \cdots$$

Credit Approval Problem

- Customer record (features):

age	23 years
gender	female
annual salary	NTD 1,000,000
year in residence	1 year
year in job	0.5 year
current debt	200,000

- To be learned:

  "Should we approve the credit card application? "

- Data: A collection of feature-label pairs:

$$(\text{customer1 feature}, \text{Yes}), (\text{customer2 feature}, \text{No}), \cdots$$

- Learned model: Some decision rule

  e.g., salary $> 1\text{M}$

Formalize the Learning Problem

- Input: $\boldsymbol{x} \in \mathcal{X}$ (customer application)

  e.g., $\boldsymbol{x} = [23,\ 1,\ 1000000,\ 1,\ 0.5,\ 200000]$
- Output: $y \in \mathcal{Y}$ (approve/disapprove)
- Target function to be learned:

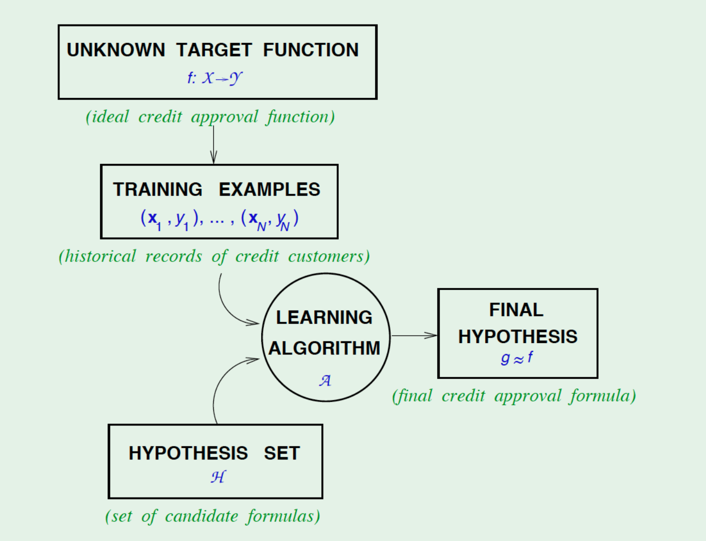
  $f : \mathcal{X} \to \mathcal{Y}$ (ideal credit approval formula)
- Data (historical records in bank):

  $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \cdots, (\boldsymbol{x}_N, y_N)\}$
- Hypothesis (model)

  $g : \mathcal{X} \to \mathcal{Y}$ (**learned** formula to be used)

$$\{(\mathbf{x}_n, y_n)\} \text{ from } f \longrightarrow \boxed{\text{ML}} \longrightarrow g$$

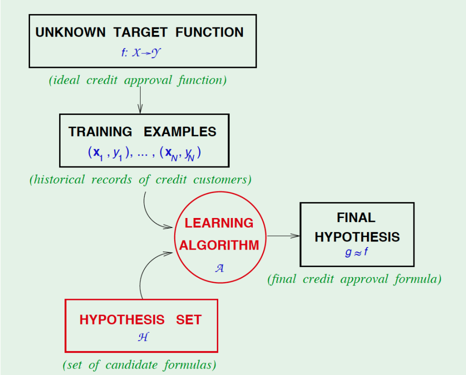
Basic Setup of Learning Problem



(Figure from "Learning from Data")

Learning Model

- A learning model has two components:
  - The hypothesis set $\mathcal{H}$:
    Set of candidate hypothesis (functions)
  - The learning algorithm:
    To pick a hypothesis (function) from the $\mathcal{H}$
    Usually **optimization algorithm** (choose the best function to minimize the training error)



Perceptron

- Our first ML model: perceptron (1957)

  Learning a linear function

  Single layer neural network
- Next, we introduce two components of perceptron:
  - What's the hypothesis space?
  - What's the learning algorithm?

Perceptron Hypothesis Space

Define the hypothesis set $\mathcal{H}$

- For input $x = (x_1, \ldots, x_d)$ "attributes of a customer"

$$\text{Approve credit if } \sum_{i=1}^{d} w_i x_i > \text{threshold},$$

$$\text{Deny credit if } \sum_{i=1}^{d} w_i x_i < \text{threshold}$$

- Define $\mathcal{Y} = \{+1(\text{good}), -1(\text{bad})\}$
- Linear hypothesis space $\mathcal{H}$: all the h with the following form

$$h(\boldsymbol{x}) = \text{sign}(\sum_{i=1}^{d} w_i x_i - \text{threshold})$$

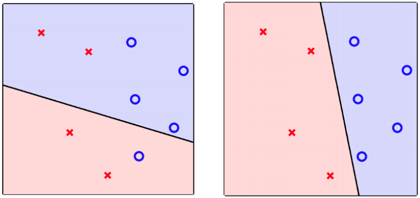
(perceptron hypothesis)

Perceptron Hypothesis Space (cont'd)

- Introduce an artificial coordinate $x_0 = -1$ and set $w_0 = \text{threshold}$

$$h(\boldsymbol{x}) = \text{sign}(\sum_{i=1}^{d} w_i x_i - \text{threshold}) = \text{sign}(\sum_{i=0}^{d} w_i x_i) = \text{sign}(\boldsymbol{w}^T \boldsymbol{x})$$

(vector form)



- Customer features $\boldsymbol{x}$: points on $\mathbb{R}^d$ (d dimensional space)
- Labels y: $+1$ or -1
- Hypothesis h: linear hyperplanes

Select the best one from $\mathcal{H}$

- $\mathcal{H}$: all possible linear hyperplanes
- How to select the best one?

Select the best one from $\mathcal{H}$

- $\mathcal{H}$: all possible linear hyperplanes
- How to select the best one?

$$\text{Find } g \text{ such that } \; g(\boldsymbol{x}_n) \approx f(\boldsymbol{x}_n) = y_n \text{ for } n = 1, \cdots, N$$

Select the best one from $\mathcal{H}$

- $\mathcal{H}$: all possible linear hyperplanes
- How to select the best one?

$$\text{Find } g \text{ such that } \; g(\boldsymbol{x}_n) \approx f(\boldsymbol{x}_n) = y_n \text{ for } n = 1, \cdots, N$$

- Naive approach:

  Test all $h \in \mathcal{H}$ and choose the best one minimizing the "training error"

$$\text{training error} = \frac{1}{N} \sum_{n=1}^{N} I(h(\boldsymbol{x}_n) \neq y_n)$$

  ($I(\cdot)$: indicator)
- Difficult: $\mathcal{H}$ is of infinite size

Perceptron Learning Algorithm

Perceptron Learning Algorithm (PLA)

Initial from some $\boldsymbol{w}$ (e.g., $\boldsymbol{w} = \mathbf{0}$)

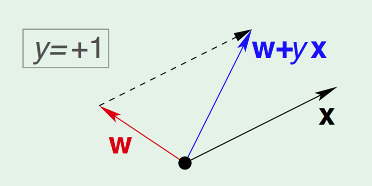
For $t = 1, 2, \cdots$

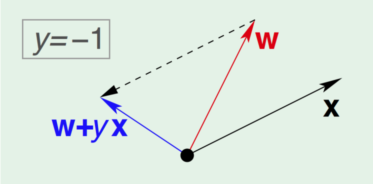
- Find a **misclassified** point $n(t)$:

$$\text{sign}(\boldsymbol{w}^T \boldsymbol{x}_{n(t)}) \neq y_{n(t)}$$

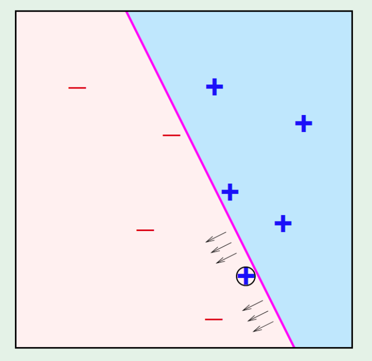
- Update the weight vector:

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + y_{n(t)} \boldsymbol{x}_{n(t)}$$





PLA

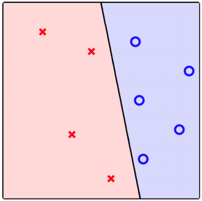


Iteratively

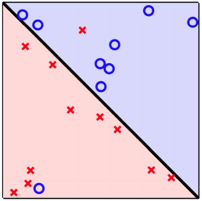
- Find a misclassified point
- Rotate the hyperplane according to the misclassified point

Perceptron Learning Algorithm

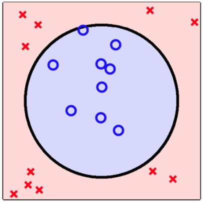
- Converge for "linearly separable" case:
  - Linearly separable: there exists a perceptron (linear) hypothesis f with 0 training error
  - PLA is guaranteed to obtain f
    (Stop when no more misclassified point)



(linear separable)



(not linear separable)



(not linear separable)

Binary classification

- Data:
  - Features for each training example: $\{\boldsymbol{x}_n\}_{n=1}^N$, each $\boldsymbol{x}_n \in \mathbb{R}^d$
  - Labels for each training example: $y_n \in \{+1, -1\}$
- Goal: learn a function $f : \mathbb{R}^d \rightarrow \{+1, -1\}$
- Examples:
  - Credit approve/disapprove
  - Email spam/not-spam
  - patient sick/not sick
  - ...

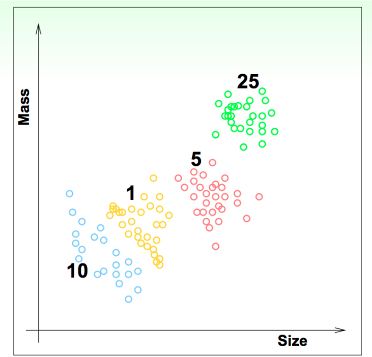
Other types of output space - Regression

- Regression: $y_n \in \mathbb{R}$ (output is a real number)
- Example:
  - Stock price prediction
  - Movie rating prediction
  - $\cdots$

Other types of output space - Multi-class prediction

Multi-class classification:

- $y_n \in \{1, \cdots, C\}$ (C-way classification)
- Example: Coin recognition
  - Classify coins by two features (size, mass) ($x_n \in \mathbb{R}^2$)
  - $y_n \in \mathcal{Y} = \{1c, 5c, 10c, 25c\}$
    ($\mathcal{Y} = \{1, 2, 3, 4\}$)
- Other examples: hand-written digits, $\cdots$



Other types of output space - Multi-label prediction

- Multi-class problem: Each sample only has one label
- Multi-label problem: Each sample can have multiple labels

Other types of output space - Multi-label prediction

- Multi-class problem: Each sample only has one label
- Multi-label problem: Each sample can have multiple labels
- Example:
  - Document categorization (news/sports/economy/$\cdots$)
  - Document/image tagging
  - $\cdots$

Other types of output space - Multi-label prediction

- Multi-class problem: Each sample only has one label
- Multi-label problem: Each sample can have multiple labels
- Example:
  - Document categorization (news/sports/economy/$\cdots$)
  - Document/image tagging
  - $\cdots$
- Extreme classification (large output space problems):
  - Millions of billions of labels (but usually each sample only has few labels)
  - Recommendation systems: Predict a subset of preferred items for each user
  - Document retrieval or search: Predict a subset of related articles for a query

Other types of output space - structure predict

- Output as exponential

$$\underbrace{\text{I}}_{\textit{pronoun}} \quad \underbrace{\text{love}}_{\textit{verb}} \quad \underbrace{\text{ML}}_{\textit{noun}}$$

- Multiclass classification for each word (word $\Rightarrow$ word class)
  (not using information of the whole sentence)
- Structure prediction problem:
  sentence $\Rightarrow$ structure (class of each word)
- Other examples: speech recognition, image captioning, machine translation, ...

1. A red stop sign sitting on the side of a road.
2. A stop sign on the corner of a street.
3. A red stop sign sitting on the side of a street.

Machine Learning Problems

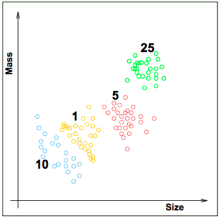
Machine learning problems can usually be categorized into

- Supervised learning: every $\boldsymbol{x}_n$ comes with y_n (label)
  (semi-supervised learning)
- Unsupervised learning: only $\boldsymbol{x}_n$, no y_n
- Reinforcement learning:

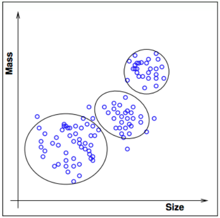
  Examples contain (input, some output, grade for this output)

Unsupervised Learning (no y_n)

- Clustering: given examples $x_1, \ldots, x_N$, classify them into K classes
- Other unsupervised learning:
  - Outlier detection: $\{x_n\} \Rightarrow \text{unusual}(x)$
  - Dimensional reduction
  - $\ldots$



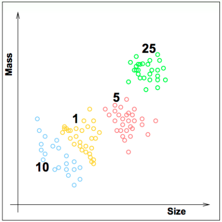
supervised multiclass classification



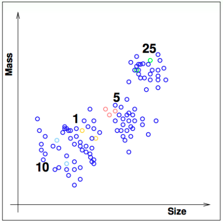
unsupervised multiclass classification
$\Longleftrightarrow$ '**clustering**'

Semi-supervised learning

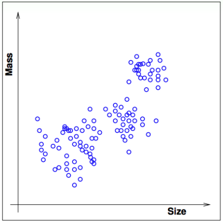
- Only some (few) $\boldsymbol{x}_n$ has y_n
- Labeled data is much more expensive than unlabeled data



supervised



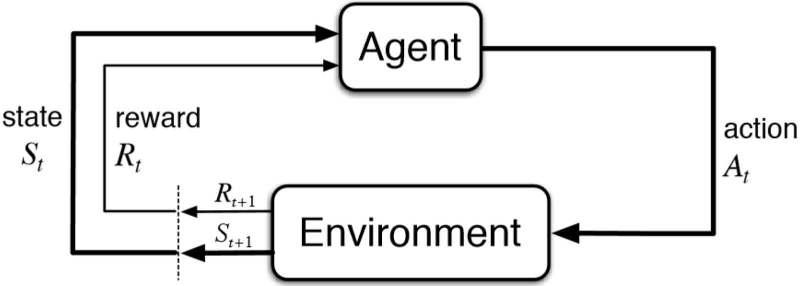
semi-supervised



unsupervised (clustering)

Reinforcement Learning

- Used a lot in game AI, robotic controls
  - Agent observe state S_t
  - Agent conduct action A_t
    (ML model, based on input S_t)
  - Environment gives agent reward R_t
  - Environment gives agent next state S_{t+1}
- Only observe "grade" for a certain action (best action is not revealed)
- Ads system: (customer, ad choice, click or not)



Conclusions

- Basic concept of learning:
  - Set up a hypothesis space (potential functions)
  - Define an error measurement (define the quality of each function based on data)
  - Develop an algorithm to choose a good hypothesis based on the error measurement (optimization)
- A perceptron algorithm (linear classification)
- Binary classification, multiclass, multilabel, structural prediction
- Supervised vs unsupervised learning

Questions?