CS449 Computational Learning (Fall 2023: 8/28 - 12/8)

Prerequisites:
CS344 *Data Structures and Algorithms*.
CS345 *Automata Theory and Formal Languages*.
Other: *Linear Algebra* (**MA339**) and *Statistics* (**STAT383**).

Instructor: Christino Tamon
Lectures: TR 08:00-09:15, SC344
Office Hours: Science Center 373, TR 10:45-13:30. Also 15min after class.

Text

M. Mohri, A. Rostamizadeh, A. Talwalkar, "Foundations of Machine Learning," 2nd edition, MIT Press, 2018. (online copy)

Syllabus

Computational learning theory studies algorithmic problems for inferring patterns and relations from data. This course describes the mathematical foundations of learning and explores the important connections and applications to areas such as artificial intelligence, cryptography, statistics, and bioinformatics. A list of relevant topics may include *perceptron* and *online learning*, *graphical models* and probabilistic inference, *decision tree* induction and *boosting*, analysis of Boolean functions, *sample complexity bounds*, cryptographic and complexity hardness, and *reinforcement learning*. Basic ideas from computer science and mathematics are employed to describe the main ideas and major developments in computational learning theory.

Objectives and Outcomes

The objective of the course is to cover fundamental topics of machine learning (such as perceptrons, neural networks, support vector machines, decision trees, boosting, hidden markov models, reinforcement learning).

(Course outcomes) Upon the completion of this course, students will be able to:

1. employ standard algorithms such as perceptron, neural networks, support vector machines, decision trees, and boosting, for solving machine learning problems.
2. use techniques such as gradient descent, backpropagation, kernel methods, and probabilistic analysis, to analyze the complexity of learning algorithms.
3. recognize incorrect analyses and construct counterexamples to false claims about learning algorithms.
4. explain the concept of NP-hardness and VC dimension to show the limits of designing efficient algorithms.

Requirements and Policies

Although attendance is not mandatory, students are responsible for all course materials covered in lectures. All students must submit their own work; the exchange of ideas are encouraged but ultimately the submitted work must be the student's own. Please refer to the Clarkson University Regulations for more guidelines on academic integrity and related matters.

Grading

- Assignments and quizzes: 40%
- Midterm test: 20%
- Final exam: 40%

Schedule (tentative)

- Week 1:
  Perceptron algorithm.
- Week 2:
  PAC learning: consistent algorithm, Occam's razor, Vapnik-Chervonenkis dimension.
- Week 3-4:
  Neural networks: sigmoided linear classifiers. Gradient descent. Backpropagation algorithm.
- Week 5:
  Online learning: Weighted Majority.
- Week 6:
  Boosting.
- Week 7-8:
  Markov models: MDP.
- Week 9-10:
  Markov models: POMDP.
- Week 11-13:
  Advanced topics.

Some brief/compressed lecture notes.

Aug 29 **Perceptron**. Linearly separable data. *Reading*: Freund-Schapire (1999). Chapter 5 (Section 5.1-5.2), Chapter 8 (Section 8.3).	**Aug 31** **Perceptron**. Mistake bound via margin. *Reading*: Chapter 5 (Section 5.1-5.2), Chapter 8 (Section 8.3).
Sep 5 **PAC learning**. When it is sufficient to be **consistent**. *Reading*: Valiant (1984).	**Sep 7** **PAC learning**. Provable sample bounds: rays, finite class. *Reading*: Chapter 1 (Kearns-Vazirani).
Sep 12 Softer perceptron: **gradient descent** with square loss. How to smooth out the learning feedback. *Reading*: Section 5.1-5.2, 8.3.	**Sep 14** Neural networks: multilayer perceptron. **Backpropagation**: gradient descent (calculus to the rescue). *Reading*: see extra notes.
Sep 19 No class. Makeup class: Fri 9/22. **Drifting** Weighted Majority. How to predict with a switching expert. Reading: Littlestone (1988).	**Sep 21** Online learning model: **Weighted Majority**. How to predict as well as the best expert. *Reading*: see class notes.
Sep 26 **Randomized** Weighted Majority.	**Sep 28** Backpropagation: message passing, tensor.

How to predict with a single expert. *Reading*: see class notes.	Quiz 1.
Oct 3 **Boosting**. How to convert a weak to a strong learner: **AdaBoost**. *Reading*: Schapire-Singer (1998).	**Oct 5** AdaBoost (HW3). *Source*: Freund-Schapire (1999).
Oct 10 Short break.	**Oct 12** Occam's Razor.
Oct 17 Dual perceptron: kernel trick.	**Oct 19** Bayesian arguments. Quiz 2.
Oct 24 Exact learning model. *Reading*: see class notes, Chapter 16. Angluin (1987).	**Oct 26** Exact learning finite automata (DFA). Angluin's L-star algorithm. *Reading*: see class notes.
Oct 31 Exact learning finite automata (DFA). Angluin's L-star algorithm. *Reading*: see class notes.	**Nov 2** Midterm.
Nov 7 Markov decision process (MDP): introduction. *Reading*: Chapter 17.	**Nov 9** Markov decision process (MDP): Value function. Optimal policy. *Reading*: Chapter 17.
Nov 14 Markov decision process (MDP): Fixed-point iterator. Convergence. *Reading*: Chapter 17.	**Nov 16** Markov decision process (MDP): worked out problem. *Reading*: see class notes.
Nov 21 Optional test.	**Nov 23** Thanksgiving break.

Assignments

- HW 1
- HW 2
- HW 3
- HW 4

Links

- Software: scikit-learn, TensorFlow.
  Others: C4.5, SVMlight, MLPack, RLtoolkit, WEKA, Vowpal Wabbit.
- Repositories: ML, preflib, kaggle.

Miscellany

- Beyond automata: Castro-Gavalda. Balle-Carreras-Luque-Quattoni. Nitay-Fisman-Ziv-Ukelson. Mohri-Yang.
- Undecidability in ML: EMX. POMDP.
- Multiplicative weights: portfolio selection: [newton], [original]. Kalai: stock.
- Boosting: Schapire, pruning, imbalanced data.
- Compression and learning: amnesia, anomaly detection, pattern detection.
- Reinforcement learning: spectral, PAC-RL. hunch.
- Information theory in learning: deep learning, etc.
- Manifold learning. Belkin-Niyogi.
- Perceptrons: margin-based, noisy (see Cohen).
- PAC-Bayes: workshop, Guedj.
- No Free Lunch theorems: free matrix lunch (Warmuth).
- Differential privacy: primer, DL, axiomatics, applications, Smith, Balcan.
- Fairness: book (fairness in ML), Dwork et al.. Kearns, Kleinberg.