# CSDS 440: Machine Learning

Soumya Ray (he/him, sray@case.edu)

Olin 516

Office hours T, Th 11:15-11:45 or by appointment

# Course Project (10/21-12/6)

- Two possibilities:
  - Propose your own. Send email by 10/21 with detailed description+group if any. Must meet with me to get approval.
  - Default: Comparative Analysis of Algorithms
- Done in groups of at most 6 people (see "Project Group" on Canvas)
- Each person in a group will read *at least one distinct* paper in the area specified for the group
  - From ICML, AAAI, NeurIPS, JMLR, MLJ, ECML etc.
  - I will send a topic and "seed" paper to each group
  - Distinct means the group's papers must not be tweaks of each other, exploring very similar ideas or have a significant amount of overlap
    - If in doubt, ask!
- Each person will implement *at least one* distinct algorithm and *at least one* novel extension of the algorithm
  - The minimum for a passing/"C" grade on the project
- The group will do a comparative evaluation of the algorithms implemented on *at least* 2 datasets

# Course Project (10/21-12/6)

- Evaluation: 35% of grade (no presentations, writeup+code)
- Each team will make a repository on cwru-courses where you will commit the project code and store papers you read
  - csds440project-f24-*n* (*n* is your project group number. Do NOT modify/capitalize differently. Do add TAs and me as admin.)
- You may use external libraries such as pytorch for implementation
  - Again, rule is you must implement significant elements of the project on your own
- Writeup to be submitted via github (Markdown)
- Writeup/final commit is due Dec 6, 11:59pm
  - No extensions

# Structure of report

- The written report will contain :
  - Individual reports with:
    - a survey of the area synthesizing the papers you read
    - a description of the specific algorithms implemented, extensions and experiments
    - an insightful discussion of the results
  - A group report documenting the comparative experiments, results and discussion
- More details on structure to follow in canvas announcement

# Grading Criteria

- Thoroughness of survey
  - Did you touch on many different important ideas? How in-depth was your exploration of the ideas?
- Technical Strength/Significance of ideas
  - How  nontrivial were the algorithms implemented? How significant were the ideas in the research extension?
- Insightfulness of results and discussion
  - Beyond “A is better than B”. When does a method work? Why does it work? How did your research extension do relative to baseline? What subsequent analysis did it inspire?
- Clarity and interestingness of writeup
  - Did you explain the ideas clearly? Did you come up with good ways to synthesize the material into coherent whole?

# Grading criteria

- Each person will receive a score on each criterion
- The group as a whole will also receive a score on each criterion
- Your final grade will be 80% of your average score over all criteria + 20% of the group's average score over all criteria
- To get a more than C grade on the "Technical Strength" "Thoroughness" and "Insightfulness" criteria you will need to go beyond the minimum
  - read more papers, implement more/technically challenging algorithms, research significant extensions, evaluate multiple hypotheses, do an insightful comparison

# Course Project steps

1. Collect papers (>=1 each), store in github papers/ subdirectory. Collect at least 2 datasets. Discuss as a group (or with me) to ensure everything looks reasonable (by 11/1)
2. Read and discuss papers. (by 11/8)
3. Implement algorithm(s). (by 11/15)
4. Carry out detailed comparative evaluation. Investigate parameter settings. Perform hypothesis tests.
5. Write report with your findings.

# Implementation: Initialization

- When initializing, generally set weights to small random values
- But some random choices work better than others
- One choice is “Xavier initialization”: choose weights from a normal distribution with mean 0 and variance $\frac{2}{n_i + n_0}$

# Other architectures

- People have also investigated networks with loops, called “recurrent neural networks”
  - This gives ANNs a “memory” (can “remember” previous inputs)
  - Different architectures proposed (Jordan networks, Elman networks, Hopfield networks, LSTMs etc)

# Pros and Cons of ANNs

+ Very very expressive hypothesis space
+ Useful for classification, regression, density estimation etc
+ "Builds" useful representations "automatically"
  - "Deep learning"

− Can be very easy to overfit
− Requires significant computational resources to train
− Generally requires many training examples
− Does not produce comprehensible models

# Probabilistic Learning (Ch 6, Mitchell)

- So far, focused on classifiers: functions mapping examples to classes
- Now, look at algorithms that explicitly estimate *probabilities* of class membership
  - $p(\mathbf{x},y)$
  - $p(y|\mathbf{x})$
  - $p(\mathbf{x}|y)$ ("Class conditional distribution")

# Why?

- Bayesian Decision Theory: optimal thing to do is to choose hypothesis to minimize *expected risk*

Risk functional

Expected Risk

$$\hat{h} = \arg\min_{h} \int_{D} R(h \mid \mathbf{x}, y)\, p(\mathbf{x})\, d\mathbf{x}$$

$$R(h \mid \mathbf{x}, y) = \sum_{\hat{y}_j} L(h(\mathbf{x}) = \hat{y}_j)\, p(h(\mathbf{x}) = \hat{y}_j)$$

In order to minimize risk, we need good probability estimates.

# Why?

- Naturally produce “confidence estimates”
- Naturally incorporate “prior knowledge”
- Can also give us tools to analyze some of our algorithms
- Can generate data
  - Basis for modern generative ML models including LLMs

# Probabilistic Classification

- Learning Task: Given data, learn probabilistic model to predict probability of class membership (*Estimation*)
  - Parameters?
  - "Structure"? (CSDS 491)

- Prediction Task: Find *most probable* class of a new example
  - *Probabilistic Inference* (classification)

# Two Approaches to Probabilistic Classification

- Generative approaches model the joint distribution $p(\mathbf{x}, y)$

- Discriminative approaches model the conditional distribution $p(y|\mathbf{x})$

# Generative Approaches

- Note: $p(\mathbf{x}, y) = p(\mathbf{x} \mid y)p(y)$
- "Sample a class, then sample an instance from it"---can "generate" the observed data
- Nicely handles missing values, unlabeled examples etc
  - If modeling assumptions met, work very well
- But do "more work" than necessary for classification

# Discriminative Approaches

- In classification, only interested in $p(y|\mathbf{x})$ anyway
- Discriminative approaches directly model this
- Robust to modeling errors
- No way to recover/ "generate" the data
  - Does not easily handle missing data
- Usually tend to be more accurate than generative counterparts, but some evidence that convergence is slower