Introduction to Machine Learning

MACS 33002 / MAPS 33002 / PLSC 43505

Meeting Days & Times:

- Synchronous Sessions (via Zoom): Tuesdays & Thursdays: 2:40 - 4:00 PM
- Class: *Zoom Classroom Link*

	Dr. Philip Waggoner	Adarsh Mathew (TA)
Email	pdwaggoner@uchicago.edu	adarshm@uchicago.edu
Office Hours	Th 9-11 am	Mon 10:30 am-12:30 pm
GitHub	pdwaggoner	adarshmathew

Overview & Introductory Remarks

Welcome to *Introduction to Machine Learning*! This course will introduce students to the foundations of machine learning. We will cover everything needed for getting up and running with any computational research project, but from a machine learning perspective. This includes covering essential machine learning tasks: classification, regression, and unsupervised learning. We will cover fundamental mathematical concepts underlying machine learning algorithms, but this course will equally focus on the application of these algorithms using open source computing. Through a final class group project, you will apply the learned concepts to address some substantive, real problem. Though influenced by theoretical statistics and mathematics, machine learning is distinct from these approaches to research in that it tends to focus on computational efficiency and maximizing accuracy (testing and deploying), based on historical data (training). This approach to computational research is exciting and rapidly developing.

We will cover a variety of topics, including: hypothesis spaces, resampling, feature engineering, model training, testing & tuning, supervised vs. unsupervised vs. competitive learning, regularization, tree-based methods, and several other algorithms contributing to a solid foundation of inferential machine learning for social science inquiry.

Note: I primarily teach and research in R, and will accordingly interactively teach this course using R. While this is the case, if students are more comfortable with other open source languages such as Python or Julia, they are welcome to use them in the course, though there will be less support for these languages.

Course Objectives

By the end of the course, students should:

- Understand what is meant by *machine learning*, from terms and notation, to mathematical properties, processes, and evaluation.
- Understand different approaches to and subfields of machine learning.

- Understand when and where it makes sense to apply machine learning for solving problems in the social sciences.
- Apply a variety of algorithms and build a variety of models in an open source programming language.
- Understand strengths and weaknesses of machine learning techniques, and thus how best to approach unique problems and data structures.
- Generate original, reproducible research reports.
- Work with a group to complete a final project that reflects all aspect of the course, from theoretical perspectives and programming best practices to fitting and evaluating models aimed at addressing real problems.

Prerequisites: Students must have some level of experience with computational and/or statistical methods. Such experience could come from any Department's approach to teaching statistical modeling or data analysis (*note*: I care more about general exposure, rather than a specific framework, as a prerequisite). Some level of programming experience is expected (e.g. CAPP 30122 (*Computer Science with Applications - 2*) or MACS 30500 (*Computing for the Social Sciences*)). Experience with R recommended, but not required.

Course Structure

This course is divided by task, rather than method. I chose to do this primarily because several methods (e.g., support vector machines or kNN) can be used for multiple tasks. As such, we will first cover *classification problems*, then *regression problems*, and end with *unsupervised learning*. Further, as it makes sense, we will cover non-task-specific concepts such as data splitting and visualization throughout as we focus on tasks and methods.

Unfortunately, due to COVID-19, I have to teach this course remotely (for the first time). Thus, I have done my best to adapt my lectures, code, and other material to make class times as fruitful and valuable as possible. But there will undoubtedly be bumps along the way. *I ask for your patience as we navigate the (still) newness of this mode of education.*

I teach this course "interactively", with code interspersed throughout lectures to help concepts come alive, rather than the traditional lecture/lab format. To make this a success, I expect every student to attend every class, having finished any assigned reading prior to class, ready to engage.

Of note, my philosophy with assigning readings for a technical, though interactive class of this sort is to provide students a good foundation to understand the concept. Class time, then, is devoted to diving deeper, and at times beyond the readings, to place the concept into a broader machine learning framework while also engaging with the concept via application.

Technology: We will use **Canvas**, **Github Classroom**, and **Zoom** as the primary modes of technology. I will actively populate **Canvas** with readings, the course schedule via "modules", and any announcements throughout the quarter. Students will use **Github classroom** to submit all problem sets and most assignments related to the final group project (proposals and reports to Github Classroom; Presentations to Canvas). And of course **Zoom** will be used to conduct synchronous class sessions.

Text & Materials

Required

1. Sign up for a free Github account at https://github.com, to allow for assignment submission via Github Classroom. For those unfamiliar with using version control for assignment submissions, see Dr. Soltoff's excellent guide: https://cfss.uchicago.edu/faq/homework-guidelines/#homework-workflow

2. Locally download R (https://cran.r-project.org/mirrors.html)
3. Locally download RStudio (https://rstudio.com/products/rstudio/download/#download)
4. (ISL) James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. *An introduction to statistical learning*. New York: Springer. Free PDF version: https://statlearning.com/ISLR%20Seventh%20Printing.pdf. Note: this text will mostly be used to give the statistical underpinnings of many of the methods we cover.
5. Kennedy, Ryan, and Philip Waggoner. 2021. *Introduction to R for social scientists: A Tidy programming approach*. Chapman & Hall/CRC Press. Free PDF in Canvas.
6. Waggoner, Philip. 2020. *Unsupervised machine learning for clustering in political and social research*. CUP. Free: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3693395

Recommended

1. VanderPlas, Jake. *Python Data Science Handbook*. O'Reilly. For Python translations of many operations we perform in R
2. (PRML) Christopher Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer. For a deeper dive into the link between CS and statistical foundations
3. (ESL) Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2009. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer. Free: https://web.stanford.edu/~hastie/ElemStatLearn/. For a deeper dive into the statistics side
4. Shai Shalev-Shwartz and Shai Ben-David. 2014. *Understanding Machine Learning: From Theory to Algorithms*. CUP. For a CS-flavored introduction to machine learning

Evaluation & Assessment

All final grades are rounded to the nearest decimal (e.g., 88.38% = 88.4%). I use the following grading scheme to determine your final grade: A (93-100), A- (90-92), B+ (88-89), B (83-87), B- (80-82), C+ (78-79), C (73-77), C- (70-72), D+ (68-69), D (63-67), D- (60-62), F (0-59). I deduct a letter grade (10%) per day any assignment is late.

There are five components to students' final grades:

- 1. Project Proposal (15%)
- 2. Group Presentation (20%)
- 3. Final Report (35%)
- 4. Problem Sets (25%)
- 5. Attendance & Participation (5%)

It's important to highlight at the outset that a large portion of your grade for the course is a group project, which will take shape in three major stages: proposal, presentation, and report. The reason for a group project is that this emulates the real world of computational problem-solving and "data science," whether academically (e.g., co-authored projects and grants) or in industry (e.g., data scientist on a team of data scientists). The expectations, as with real world projects, are that you first, specialize on a given component of the project (e.g., one member is the *programmer*, another is the *theoretical architect*, etc.) in order to maximize efficiency and productivity. And second, submit and present a *single* final project that addresses a substantive challenge or problem of interest in your preferred domain, whether sociology, political science, computer science, neuroscience, etc. In sum, the goal of the final project is three-fold. *First*, to apply the concepts learned in this class creatively, which will help you understand material more deeply. *Second*, to design and work on a unique project in a team, which is something that you *will* encounter, if you haven't already. *Third*, along with the opportunity to practice and the satisfaction of working creatively, you can use this project to enhance your portfolio or CV.

There is *no* perfect project. Whether your project was an empirical "success" or "failure" is significantly less important to me than your process of getting to the final stage, whatever that may look like. I am

interested in your workflow, e.g., what steps were involved in creating your solution, what did you learn, how did your solution perform, what are limitations to be addressed in future iterations, and so on. Your grade will be calculated based on your thoroughness and honesty in reporting.

As the group project comprises a large portion of your final grade, I expect you will be thinking about and working on these projects from Day 1 of the course. Please come talk to me to workshop, problem-solve, and address any questions throughout the life cycle of your project; I want you to succeed and am very available as a result.

There are relatively few guidelines for the subject matter of your final project. My only requirement is that your methods incorporate machine learning techniques covered in the course. For example, you could use a clustering algorithm to derive class labels for previously unlabeled data, which could then be fed to a supervised classifier to predict classes of unseen/new data. Or alternatively, you could fit a variety of supervised classifiers (e.g., random forest, penalized logistic regression, neural network, SVM, etc.) aimed at classifying winning candidates for political office based on prior election performance, and then compare quality and accuracy across the algorithms.

You will be working in a team of ~2-4 people, which you will organize on your own. Groups must be organized by the end of the first week of class (Friday). If you are having trouble finding a group, *reach out to me sooner rather than later* to request help and I will assign you to a group. If I don't hear from anyone, I will assume all students are in groups. As time will go very quickly, you are strongly encouraged to **keep in regular communication with your group members**, whether in person, virtually (e.g., Slack, email, text, etc.), or both throughout the quarter. If you have any concerns working with someone in your group, please talk to me for accommodations.

1. Project Proposal (15%): The proposal is where your group will (unsurprisingly) pitch its project idea. **Project proposals are due at the end of Week 3 on Friday, January 29, by 11:59 pm on Github Classroom.** The goal of this stage is two-fold. First, it is a good initial, yet relatively low-stakes test of your group dynamics. As this is where you will introduce the problem, methods/solution, data, and so on, such a task requires consolidation and universal contribution to the overall vision of the project; in a word, *teamwork*. And second, the goal of the proposal is to get early-stage feedback from me on the feasibility of the project, both regarding your ability to complete it within the time frame of the course, as well as whether your project is in the scope of the class.

Some particulars: These should be ~1-2 pages, excluding references (can be via footnote or reference list at the end). Final submissions should be a single PDF for the whole group submitted via Github Classroom. Use 11- or 12-point standard font. You can select any reference style you wish (Chicago, APA, MLA, etc.); just be correct and consistent.

2. Group Presentation (20%): All presentations must be recorded, and include *all* group members. Consider recording a Zoom call in "grid view" with a shared screen for the slides. Presentations are due to the appropriate *Canvas* module on **Sunday, March 14 of Week 9 by 11:59 pm**, which is prior to the final week of class. Monday (3/15) - Wednesday (3/17) during the final week 10 of the class, students must view **all** presentations, but only respond/comment in Canvas on at least **two** other groups' presentations. Comments can include anything from questions to suggestions, but must be constructive and appropriate.

Though I don't have a specific set of guidelines for organizing presentations, good presentations typically include some combination of the following:

- Introduce the topic to a *general* audience with a motivating anecdote, paradox, or example
- Summarize your process, workflow, and your approach or method
- Highlight the key findings, but avoid going into the weeds as time is limited
- "Appendix" set of slides with more detail at the end in case you are questioned on specific details (though we won't have a live Q&A session)

The presentation should be no more than 8 minutes in length. All members of the group must participate in the presentation.

I like to give presentation awards to both encourage greater quality, and also democratize class participation. Two awards will be given for presentations:

- **Best Presentation**
- **Most Creative Project**

Awards are determined by class voting, where each student will fill out a ballot via Google forms, and distributed by me via email at the end of the class. Voting is on a scale from 1-10 for each of the 2 categories, where 10 is best. I will collect the ballots as they are filled out and submitted. Upon summing the points across each award category, the highest point-getter wins the respective award. Only 1 winner is allowed for each category, and the winner gets a 5% bonus (half a letter-grade) on the final project report, which can be $> 100\%$. In case of a tie, I will select a winner at random, each with an equal probability of winning (i.e., a single Bernoulli trial). You may vote for yourself, if you're truly blown away by your own creative genius.

3. Final Report (35%): The final project report is the distillation of your group's efforts over the course of the quarter. **Only one report** should be submitted by your group, with each person's name on the report. **Final project reports are due Thursday, March 18 to Github Classroom by 11:59 pm.**

To reinforce the *applied* part of this course, students must follow the *Proceedings of the National Academy of Sciences* (PNAS) article format. This is a two-column, single-spaced paper with a **maximum of 6 pages**:

- Follow the PNAS formatting guidelines here: https://www.pnas.org/page/authors/format
- Use their LaTeX template here: https://www.overleaf.com/latex/templates/template-for-preparing-your-research-report-submission-to-pnas-using-overleaf/fzcbzjvpvnxn
- For those unfamiliar with TeX, and in need of a bit of extra help, see my help files from a workshop I taught a few years ago, here: https://github.com/pdwaggoner/LaTeX-Workshop. Of course, start by Googling errors/questions, then reach out to your colleagues, and ultimately reach out to me if you're still stumped after giving troubleshooting a fair effort.

Distilling a complex project into 6 pages, though two columns, is an extremely difficult task. This requires crisp and clear writing, an ability to focus on the most important aspects of the methods and the results, as well as an ability to tie substantive patterns back to the "real world" (i.e., the *so what?* question). Throughout the quarter, I will highlight best practices in high level, professional research and writing of this sort. But it's up to each student to leverage the course to his/her advantage. In practice, this might look like using the problem sets or the initial Project Proposal as testing grounds for homing your writing abilities (of note: the proposal is also required to be extremely brief at 1-2 pages). Further, the value of using PNAS as the standard is that all of you will likely, if you haven't already, come into contact with PNAS as they publish articles in virtually every substantive domain, e.g., the physical sciences, social sciences, medical sciences, and so on. In sum, this approach should help you on several dimensions, and I hope you take and deepen these skills in your careers as you progress beyond what we cover in the course.

Data & Code: Each group should include all data and code in reproducible form (e.g., markdown of any flavor, etc.). This should go in the private repository where final papers are submitted, yet code should *not* be included in the final report itself.

4. Problem Sets (25%): Throughout the quarter, students will be complete **five** problem sets. Some of these will be more conceptual requiring long answers, others will be more computationally intensive, while still others may be a blend of these types. Unless otherwise noted, problem sets will be released on a Tuesday, and due a week later on the following Tuesday by 11:59 pm.

Note: A completed submission will be a **single** PDF (rendered from the `.Rmd` or Jupyter Notebook), and will include: a response to the question, presentation of all equations, code, and output (e.g., plots, tables, etc.) inserted *directly below the response to the question*. In other words, no appendices are necessary for problem sets. Just write and run the code, and produce the output in-line. **A single PDF is all that is required for submission.**

5. Attendance & Participation (5%): As this is a graduate course, whether we are discussing an article, working through code as a class, or engaging in an interactive lecture, I expect all students to come to every class prepared to engage and discuss, having read all required materials *prior* to class and to be respectful of their peers. Respect includes appreciating the wide variance in opinions and backgrounds that are sure to exist. I don't anticipate problems in this regard, but need to underscore these expectations at the outset to ensure everyone is on the same page.

Diversity & Inclusion

The University of Chicago is committed to diversity and rigorous inquiry from multiple perspectives. The MAPSS, CIR, and MACSS programs share this commitment and seek to foster productive learning environments based upon inclusion, open communication, and mutual respect for a diverse range of identities, experiences, and positions.

Any suggestions for how we might further such objectives both in and outside the classroom are appreciated and will be given serious consideration. Please share your suggestions or concerns with me (the professor), your preceptor, TA, or your program's Diversity and Inclusion representatives: Darcy Heuring (MAPSS), Matthias Staisch (CIR), and Chad Cyrenne (MACSS). You are also welcome and encouraged to contact the Faculty Director or Chair of your program.

This course is open to all students who meet the academic requirements for participation. Any student who has a documented need for accommodation should contact Student Disability Services (773-702-6000 or disabilities@uchicago.edu) *and* the instructor as soon as possible.

Course Schedule

Note: The schedule below is tentative and may change. If changed, students will be made aware with sufficient time to adjust.

Syntax: *assignments released in italics*; **due dates in bold**

Week 1

- Jan. 12 (Tuesday) Introduction to Machine Learning
  - Reading:
    - Syllabus (*read carefully*)
- Jan. 14 (Thursday) Computational foundations: Model-based programming, Feature engineering for missing data
  - Reading:
    - Kuhn and Johnson. 2019. *Feature Engineering and Selection: A Practical Approach for Predictive Models*, *skim* chs. 1 and 8.
    - Kennedy and Waggoner. 2021. *Introduction to R for Social Scientists*, chs. 1-4 (*skim as/if needed*)
- Jan. 15 (Friday)
  - **Research teams organized by 5 pm CDT**

Week 2

- Jan. 19 (Tuesday) Neighbor-based classification
  - *PS1 Released*
  - Reading:
    - ISL ch. 3.5
- Jan. 21 (Thursday) Probability-based classification
  - Reading:
    - ISL chs. 4.3, 4.4

Week 3

- Jan. 26 (Tuesday) Non-probability classification
  - **PS1 Due to Github Classroom by 11:59 pm CDT**
  - Reading:
    - ISL ch. 9
- Jan. 28 (Thursday) Tree-based classification
  - Reading:
    - ISL ch. 8
- Jan. 29 (Friday)
  - **Project Proposals Due to Github Classroom by 11:59 pm CDT**

Week 4

- Feb. 2 (Tuesday) Linear regression
  - *PS2 Released*
  - Reading:
    - ISL ch. 3
- Feb. 4 (Thursday) Non-linear regression
  - Reading:
    - ISL ch. 7

Week 5

- Feb. 9 (Tuesday) Regularization in regression problems
  - **PS2 Due to Github Classroom by 11:59 pm CDT**
  - Reading:
    - ISL ch. 6.2
- Feb. 11 (Thursday) Adapting (*typically*) classification methods for regression problems
  - Reading:
    - Review if needed, ISL chs. 3.5, 8.2.2, 8.2.3

Week 6

- Feb. 16 (Tuesday) Clustering, pt. 1
  - *PS3 Released*
  - Reading:
    - ISL ch. 10.3
    - (*skim*) Waggoner. 2020. *Unsupervised Machine Learning for Clustering in Political and Social Research*, chs. 1-4
- Feb. 18 (Thursday) Clustering, pt. 2
  - Reading:
    - Waggoner. 2020. *Unsupervised Machine Learning for Clustering in Political and Social Research*, chs. 5-6

Week 7

- Feb. 23 (Tuesday) Dimension reduction, pt. 1
  - *PS4 Released*
  - **PS3 Due to Github Classroom by 11:59 pm CDT**
  - Reading:
    - ISL ch. 10.2
- Feb. 25 (Thursday) Dimension reduction, pt. 2
  - Reading:
    - Roweis and Saul. 2000. "Nonlinear dimensionality reduction by locally linear embedding."

Week 8

- Mar. 2 (Tuesday) Dimension reduction for visualization
  - *PS5 Released*
  - **PS4 Due to Github Classroom by 11:59 pm CDT**
  - Reading:
    - van der Maaten and Hinton. 2008. "Visualizing data using t-SNE."
    - McInnes, Healy, and Melville. 2018. "Umap: Uniform manifold approximation and projection for dimension reduction."
- Mar. 4 (Thursday) Neural networks - **Guest Lecture: Adarsh Mathew**
  - Reading:
    - Goodfellow et al. 2016. *Deep Learning*, ch. 6
    - Jurafsky & Martin. 2020. *Speech and Language Processing*, ch. 7
    - (*recommended*) Bishop. 2006. PRML, chs. 5.1-5.3

Week 9

- Mar. 9 (Tuesday) (Deep) Neural-based dimension reduction
  - **PS5 Due to Github Classroom by 11:59 pm CDT**
  - Reading:
    - Goodfellow et al. 2016. *Deep Learning*, chs. 6 (*again if needed*), 14.
    - (*recommended*) Rudin. 2019. "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead."
- Mar. 11 (Thursday) **Class Canceled: Work on final presentations and reports**
- Mar. 14 (Sunday)
  - **Final Group Presentations due on Canvas by 11:59 pm CDT**

Week 10 - **No Regular Class Meetings: View/Comment on Presentations; Complete Final Report**

- Mar. 15 (Monday) - Mar. 17 (Wednesday)
  - **View *all* presentations; Respond to at least *two***
- Mar. 18 (Thursday)
  - **Final Group Project Report due on Github Classroom by 11:59 pm CDT**