UC Berkeley School of Information | MIDS Program

DATASCI w261
Machine Learning at Scale

Course Architect: Dr. James G. Shanahan, 1/1/2015

Syllabus Last Revised on 25/04/2020.

Course Overview

This 3 Unit course covers the underlying principles required to develop scalable machine learning analyses for structured and unstructured data at the petabyte scale. Content is delivered via asynchronous video lectures, readings from academic textbooks, synchronous session discussions, live code demonstrations, and independent homework assignments.

The course is designed around three goals. By the end of the term students will...

1. ... learn to recognize and apply key concepts in parallel computation and MapReduce design.
2. ... design stateless parallelizable implementations of core machine learning algorithms from scratch.
3. ... gain hands-on experience using Apache Hadoop and Apache Spark to analyze large datasets.

Prerequisites: Introductory machine learning course or equivalent. Intermediate programming capabilities in an object-oriented language (such as Python).

Content Description

Until recently, "big data" was very much the purview of database management and summary statistics systems such as Hadoop (HDFS and MapReduce) and was largely underleveraged by machine learning. This course builds on and goes beyond this collect-and-analyze phase of big data by focusing on how machine learning algorithms can be rewritten and in some cases extended to scale to work on petabytes of data, both structured and unstructured, to generate sophisticated models that can be used for real-time predictions. Predictive modeling at this scale can lead to huge boosts in performance (typically in the order of 10–20%) over small-scale models running on stand-alone computers that require one to significantly down-sample and, necessarily, to simplify big data. Concretely, this course focuses on how the map-reduce design paradigm from parallel computing can be extended and more faithfully leveraged to tackle the somewhat "embarrassingly parallel" task of machine learning (a lot of machine learning algorithms fit this mold).

The Apache Spark project and its many related subprojects exemplify the continued relevance of map-reduce style algorithm design. Apache Spark is an open-source cluster-computing framework. It has emerged as the next-generation big data processing engine, overtaking Hadoop MapReduce, which helped ignite the big data revolution. Spark maintains MapReduce's linear scalability and fault tolerance but extends it in a few important ways: it is much faster (100 times faster for certain applications); much easier to program in, due to its rich APIs in Python, Java, and Scala (and R) and its core data abstraction, the distributed data frame; and it goes far beyond batch applications to support a variety of compute-intensive tasks, including interactive queries, streaming, machine learning, and graph processing.

This course will provide an accessible introduction to MapReduce frameworks and to Spark and its potential to revolutionize academic and commercial data science practices through scale. Conceptually, the course includes two simultaneous components. The first covers fundamental concepts of MapReduce parallel computing via Hadoop and Spark. The second focuses on hands-on algorithmic design and development in parallel computing environments such as Spark; developing algorithms from scratch, such as decision-tree learning; graph-processing algorithms such as PageRank and shortest path; gradient descent algorithms such as support vector machines; and matrix factorization. Industrial applications and deployments of MapReduce parallel compute frameworks from various fields, including advertising, finance, healthcare, and search engines, help tie these components together. Examples and exercises will be made available in Python notebooks (Hadoop streaming and PySpark).

Weekly Topics

Week	Topic	Deadlines
1	Intro to Machine Learning at Scale	
2	Parallel Computation Frameworks	HW 1 due
3	Map-Reduce Algorithm Design	
4	Intro to Spark/Map-Reduce with RDDs (part 1)	HW 2 due
5	Intro to Spark/Map-Reduce with RDDs (part 2)	
6	Distributed Supervised ML (part 1)	HW 3 due
7	Distributed Supervised ML (part 2)	
8	Big Data Systems and Pipelines	HW 4 due
9	Graph Algorithms at Scale (part 1)	
10	Graph Algorithms at Scale (part 2)	
11	ALS and Spark ML	HW 5 due
12	Decision Trees	
13	Spark Streaming	
14	Final Project Presentations	Final Projects Due

Things you need to know prior to starting 261

The assumption is that having completed the prerequisites for 261, you have competence in the below tooling. If you are not comfortable with the below tooling, **expect to at least double the amount of time that 261 will take you to complete**.

- Bash skills
    - Control structures (if, while, etc.),
    - HEREDOCS
- Linux Skills
    - Determine running processes
    - Determine ports currently in use
- Python Programming
    - General programming tasks such as string manipulation, list comprehensions, and control structures
    - Familiarity working with Numpy and Pandas
- Basic networking concepts
    - RFC 1918 and Non-RFC 1918 IP Addresses
    - What is a port and port forwarding
    - SSH proxies
- Exposure to cloud computing, preferably GCP
    - Understanding of IAM models in particular
- Docker
    - Ability to make a dockerfile from scratch
    - Ability to read kubectl.yml or docker-compose files and modify as needed
- Git
    - Ability to work with command line
    - Understand upstream/downstream repositories and forks
- Familiarity with reading log files and using tracebacks to debug code in Python/Java
- Linear Algebra and Calculus
    - Derivatives
    - Dot Products
    - Matrix multiplication
    - Math notation

Assignments and Materials

This is an upper level graduate course. As such, we expect students to exhibit a high level of conscientiousness and initiative in their approach to preparing for class and completing assigned work. Each week you will have assigned video content as well as readings -- both should be completed before your live session*. In live session we will typically review a short set of conceptual slides and/or break into groups to do a code demonstration. In some cases your live session instructor may ask you to review the first section of a demo notebook before you arrive in class. These code "demos" are designed to set you up for success on the homework. They offer a low-risk opportunity to build supporting skills-- the more actively you engage with them during class, the less time you will end up spending on the homework.

Grading Policy

% of Final Grade	Component
50%	Homework Assignments (5 assignments, 10% each)
40%	Final Project
10%	Live session attendance and participation

Homework

Each independent homework assignment consists of a python notebook with coding and short response conceptual questions. We believe these assignments to be active learning experiences and hope you will approach them as an opportunity to develop your own understanding and not just a source of a grade. Feedback from past students has consistently indicated that the challenge of these assignments is what makes the course valuable to them post-MIDS. For now, here's what you need to know:

- **Accessing and Submitting:** Your homework will be distributed and submitted via GitHub and/or Databricks. (see the "Logistics" section below for details). DO NOT SUBMIT HOMEWORK ON ISVC.
- **Time Commitment:** We expect a well prepared student to spend 15-25 hours on each homework. Depending on your background this time will vary. **If you do not have a background in any of software engineering or coding, mathematics and/or statistics, be prepared to spend more time beyond the 25 hours.**
- **Late-Day Policy:** As much as we are committed to serving a rigorous course we also know you are hard working professionals, parents, and partners. To give you some flexibility we offer **7 late days** that you can use to extend any homework deadline by up to 2 days each.
- **Partial Credit:** We grade each multi-part question holistically; always attempt as much as you can because we'd love to give you partial credit for partial understanding.

Grading Policy

Mastery Based Grading - We are looking for evidence that you understand or misunderstand the learning objective/key concept. Every question will receive one of 4 scores: 100, 90, 50 or 0 as well as written feedback. A 90 or 100 for any answer that does demonstrate understanding of the Learning Target (including answers that have errors in code that are unrelated to the key concept). A 50 for any answer that fails to demonstrate an understanding of the Learning Target (including vague but plausible answers). 0 for a blank or nearly blank response.

For each question the score you get is not an indication of a "percent of the sub questions you got right" instead you should think of it as a categorical indicator.

Each question had some one or two core concepts plus a lot of details. The goal is to avoid docking students multiple times for little details but still ensure that overall you get a clear message if you miss an important concept - so anything that shows confusion/error around the core learning target gets a 50 while any other error or combination of errors gets a 90.

This can seem harsh when you're on the 50 end of that equation but we've found that over the course of a full assignment it helps as much as it hurts. The goal of the grades on the homework is formative rather than summative.

Final Project

The final project is a group assignment that offers an opportunity to demonstrate your mastery of the course goals. We will assign groups and release a rubric in week 8 and your team will have one month to organize the workload, perform relevant EDA, implement the algorithm, and deliver a python notebook based analysis report including citations results and discussion of parallelization concepts that impacted design choices you made.

Readings

This course will use a combination of textbook chapters and some online readings. The books highlighted in blue are some of your instructor's favorites. Books marked with an asterix are not available for free, and you will need to purchase them. The rest of the readings are available for free online.

Recommended Textbooks:

- *Karau, Holden, Konwinski, Andy, Wendell, Patrick, & Zaharia, Matei. (2015). *Learning Spark: Lightning-fast big data analysis*. Sebastopol, CA: O'Reilly Publishers.
- Lin, Jimmy, & Dyer, Chris. (2010). *Data-intensive text processing with MapReduce*. San Rafael, CA: Morgan & Claypool Publishers. (Free online)
- Karau, Warren. (2017). *High Performance Spark: Best Practices for Scaling and Optimizing Apache Spark.* Sebastopol, CA: O'Reilly Publishers.
- Hastie, Trevor, Tibshirani, Robert, & Friedman, Jerome. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (2nd ed.). Stanford, CA: Springer Science+Business Media. (Free online)
- Hastie, Trevor, Tibshirani, Robert, Witten, Daniela, & James, Gareth. (2014). *An Introduction to*

Statistical Learning: with Applications in R. Stanford, CA: Springer Publishing Company. (Free online)
    - https://lagunita.stanford.edu/courses/course-v1:ComputerScience+MMDS+SelfPaced/course/

- *Ryza, Sandy, Laserson, Uri, Owen, Sean, & Wills, Josh. (2015). *Advanced analytics with Spark: Patterns for learning from data at scale.* Sebastopol, CA: O'Reilly Publishers.
- Leskovec Jure, Rajaraman Anand, Ullman Jeff, (2014). *Mining of Massive Datasets*, Cambridge University Press. Book available online at http://www.mmds.org/
    - *https://lagunita.stanford.edu/courses/course-v1:ComputerScience+MMDS+SelfPaced/course/*
- *Doing Data Science* by O'Neil & Shutt, ISBN-13: 978-1449358655, ISBN-10: 1449358659
- *Hadoop: The Definitive Guide* by Tom White, ISBN: 978-1-491-90163-2; O'Reilly 2015
- **Spark: The Definitive Guide: Big Data Processing Made Simple* By Bill Chambers, Matei Zaharia, ISBN-13: 978-1491912218, ISBN-10: 1491912219, O'Reilly 2018

Logistics and Communication

GitHub is the source of ground truth for assignments, deadlines and policies in this course. In the first week of class you will create your own personal w261-homework repository where you will submit assignments. We will rely on Slack to communicate about any changes, discuss content, and provide clarification. However if you need to get in touch with an instructor for a non-content related reason please use email and cc the TA for your section.

GitHub Repo

At the start of the semester you will be granted access to the course github repo. The README.md file in this repo contains your weekly reading assignments, class schedule, homework deadlines and links to all materials you'll need for live sessions. This is also where we'll release homework assignments. You will set up this repo as an upstream remote on your personal repository to pull down the templates each week. (*ask a TA in week 1 if you need help with this*)

Slack Channels

Slack serves a few purposes for this course. In the first week of classes you will be expected to join three channels, one for general discussion, one for troubleshooting infrastructure issues related to the Docker container or AWS cluster and one for major announcements (HW due date changes, live session changes, etc..).

#w261-<semester>-announcements is for faculty only. You can reply in a thread to an announcement to request clarification, but please do not post questions or comments in this channel.

To keep these channels helpful for everyone please follow a few norms:

- Use threaded replies to help keep different lines of conversation easy to find.

- Respect that everyone comes from a different background -- share your silly questions freely and answer others' queries with good cheer.
- When asking a question, describe what you've already tried to resolve your own question and reference any course materials you may be looking at -- this helps us make better suggestions faster.
- Commiserating can be a form of support (especially in a very time consuming class) but watch out for the line where commiserating turns into complaining -- don't bring or put others down and use other channels (eg. surveys or email) to share constructive criticism intended for instructors' ears.

Office Hours

Office hours will be held every week. Refer to the course repo for dates/times that each instructor will be available.

Infrastructure

Although learning how to set up a distributed environment for parallel computation is *not* a learning objective for this course, many of the concepts that are central to understanding parallel computation *do* require students to learn a little bit about infrastructure. We hope to keep infrastructure frustrations to a minimum by providing two consistent environments for students so that we can help you navigate reproducible errors related to your environment configuration:

- **Docker Container in the cloud**: You will be given instruction to spin up a VM in the cloud with a docker container which has everything you need for the class.
- **Databricks**: In addition to the docker container, we will be using databricks to complete Spark assignments/projects.
- **Students that need assistance should email help@ischool.berkeley.edu.** This will open a ticket in their issue tracking system. The folks at 'help' will follow up with you initially via email. If they aren't able to help you resolve the issue that way, they'll set up a Zoom conference with you, have you share their screen, and walk you through the process.

In the past, students with the skillset to do so have set up their own environment (eg. using AWS or Databricks). While you are welcome to do so, you should be aware that Hadoop and Spark may have different out-of-the box behavior depending on your system and we cannot support you directly unless you use the provided course materials.

Weekly Materials:

Reading Assignment Abbreviations:

"HDG" = Hadoop: Definitive Guide (4th Edition) by Tom White
"SDG" = Spark: The Definitive Guide: Big Data Processing Made Simple By Bill Chambers, Matei Zaharia
"DITP"" = Data Intensive Text Processing With Map Reduce by Lin & Dyer
"IIR" = Introduction to Information Retrieval by Manning, Raghavan, & Shutze
"ISL" = Introduction to Statistical Learning by Witten, James, Hastie, & Tibshirani
"MMS" = Modern Multivariate Statistical Techniques by Izenman
"Learning Spark*" = Learning Spark: High Performance Big Data Analysis by Karau, Konwinski, Wendell, and Zaharia
"HP Spark*" = High Performance Spark by Karau and Warren
"DDS*" = Doing Data Science by O'Neil & Shutt

*Starred books are ones you will need to purchase or borrow from the UCB library. All other reading materials are open source & linked here for your convenience.
To access the library, go to http://oskicat.berkeley.edu/ and use your Berkeley login.

	Async
Week 1 Intro to Machine Learning at Scale	Read: ISL chapter 1 and sections 2.1 & 2.2 Skim: Adam Drake Blog Post Optional: Fortmann-Roe Essay, Clever Machine Blog Post, Inside Big Data Blog Post
Week 2 Parallel Computation Frameworks	Read: DITP Chapter 1 & Chapter 2 Read: IIR CH.13 Optional: Michael Noll Hadoop MR Tutorial NOTE: *Please come to class this week with your Docker container running and the data for the demo notebook loaded.*

Week 3 MapReduce Algorithm Design	Read: DITP sections 2.4 - 2.7 and 3.1-3.4 Read: IIR sections 13.1 and 13.2 Read: HDG - Part II Chapter 7 - How MapReduce Works Skim: Total Order Sort Guide, EECS Map Reduce Notes OPTIONAL: http://blog.ditullio.fr/category/hadoop-basics/
Week 4 Intro to Spark/MapReduce with RDDs	Read: HP Spark chapter 2 Read: Spark RDD Programming Guide Skim: Learning Spark ch 3 & 4 Skim: DISCO Paper Skim: DocSim Paper Additional Spark resources for weeks 4, and 5 ● Holden Karau - Spark summit 2017https://www.youtube.com/watch?v=4xsBQYdHgn8&feature=youtu.be [40 minutes] ● Debugging Spark Holden Karau -Dec 2017 https://www.youtube.com/watch?v=s5p15QT0Zj8&list=WL&index=7&t=0s[45 minutes]
Week 5 Spark/MapReduce with RDDs (con't)	Watch: Debugging Spark - Holden Karau - Dec 2017 [45 minutes] Skim: IIR chapter 16 Optional: Apriori Algorithm Chapter
Week 6 Distributed Supervised ML (part 1)	Read: ISL sections 3.1, 3.2 and 6.1, 6.2 Skim: UCI cs273a Loss Functions Lecture

Week 7 Distributed Supervised ML (part 2)	Read:DDS chapter 5 Read: ISL chapter 4 & section 5.1 Skim: MMS chapter 11
Week 8 Data systems and pipelines	Skim: HP Spark ch 3-4 Read: HP Spark ch 5-6 Read: Format Wars Post Optional: SparkSession article, Sparkour recipe
Week 9 Graph Algorithms at Scale (part 1)	Read: DITP chapter 5 Skim: Cornell CS 312 Dijkstra's Lecture
Week 10 Graph Algorithms at Scale (part 2)	Read: DITP chapter 5
Week 11 ALS	Read: MATRIX FACTORIZATION TECHNIQUES FOR RECOMMENDER SYSTEMS Read: DDS Chapter 8 Skim: Learning Spark ch 11
Week 12 Decision Trees	Read: ISL chapter 8 (or chapter 9.2 in ESL) Read: PLANET paper OPTIONAL: Skim TOC in DATA MINING WITH DECISION TREES, Theory and Applications A most excellent introduction to Gradient Boosting: How to explain gradient boosting

Week 13 Spark DataFrames and Online Learning	READ: https://www.oreilly.com/ideas/the-world-beyond-batch-streaming-101 https://www.oreilly.com/ideas/the-world-beyond-batch-streaming-102 http://streamingsystems.net/ (Chapters 1 and 2) Bennet, Elizabeth. “Putting the Power of Kafka into the Hands of Data Scientists” Stitchfix Company Blog. https://multithreaded.stitchfix.com/blog/2018/09/05/datahighway/ OPTIONAL - Spark The Definitive Guide Big Data Processing Made Simple - PART V: chapters 20-23 - Das, Tagata; Zaharia, Matei; Wendell, Patrick. “Diving into Apache Spark Streaming’s Execution Model” Databricks Company Blog. https://databricks.com/blog/2015/07/30/diving-into-apache-spark-streamings-execution-model.html - Das, Tagata; Torres, Joseph. “Introducing Stream-Stream Joins in Apache Spark 2.3” Databricks Company Blog. https://databricks.com/blog/2018/03/13/introducing-stream-stream-joins-in-apache-spark-2-3.html
Week 14 Final Project presentations	No new videos this week