

CAS PY 580: Machine Learning for Physicists

Instructor: Pankaj Mehta^[SEP]
pankajm@bu.edu
SCI 323
617-358-6303

Time: TTh 9:30-11:00, SCI B58

Office Hours: Tuesday: 2:30-3:30 (before colloquium)

Course Credits: 4 credits

Prerequisites: PY 410/PY 541 or permission of instructor

Description: Machine Learning (ML) is one of the most exciting and dynamic areas of modern research and application. The purpose of this class is to provide an introduction to the core concepts and tools of machine learning in a manner easily understood and intuitive to physicists. The class begins by covering fundamental concepts in ML and modern statistics such as the bias-variance tradeoff, overfitting, regularization, and generalization before moving on to more advanced topics in both supervised and unsupervised learning. Possible topics covered in the class include ensemble models, deep learning and neural networks, clustering and data visualization, energy-based models (including MaxEnt models and Restricted Boltzmann Machines), and variational methods. Throughout, we emphasize the many natural connections between ML and statistical physics. A notable aspect of the class is the use of Jupyter notebooks to introduce modern ML/statistical packages to readers using physics-inspired datasets (the Ising Model and Monte-Carlo simulations of supersymmetric decays of proton-proton collisions). We will conclude with an extended outlook discussing possible uses of machine learning for furthering our understanding of the physical world as well as open problems in ML where physicists maybe able to contribute.

Books and Other Course Materials

The course follows are recent review available here:
<https://www.sciencedirect.com/science/article/pii/S0370157319300766>
as well as the corresponding notebooks available here:
<https://physics.bu.edu/~pankajm/MLnotebooks.html>

Grading: There will also be a final group project in teams of 3. The grade will be based 40% on HWs, 10% on Final Project, 50% on class participation/readings.

HW: The class will have a mix-of more theoretical problems and practical programming assignments in the 20 Notebooks (<https://physics.bu.edu/~pankajm/MLnotebooks.html>).

Final Project: The final project must be chosen in consultation with the instructor by the beginning of November. The final project will be a group project (teams of 2-3 members) and must have a significant practical programming/data analysis component.

Community of Learning: Class and University Policies

- 1) Course members are responsible for ensuring a positive learning environment by being respectful of their fellow students.
- 2) **Attendance & Absences.** Class discussion and participation are a major part of the course. Class participation is 40% of the grade and absences will lower this score as appropriate except when students receive permission from the instructor or for religious observances.
- 3) **Assignment Completion & Late Work.** Due dates for notebooks are 1 week after assignment. **You will be expected to turn in HWs on time. There will be 50% credit for HWs turned in within one week of due dates. Late HWs will not be accepted after 1 week.**

4) **Academic Conduct Statement.** All students are expected to follow GRS Academic Conduct Code: <http://www.bu.edu/cas/students/graduate/grs-forms-policies-procedures/academic-discipline-procedures/>

Weekly assignments: We will work through 1-2 chapters in the review each week. Students are expected to read the corresponding text in the review before class to facilitate discussion. The exact pace of readings depends on discussions/confusions and interest of the students.

A typical course will read Chapters 1-7 in **Weeks 1-7**, Chapters 9-13 in **Weeks 8-10**, and then depending on students' interests a selection of topics from Chapter 5, 14-17 in **Weeks 11-14**. There are notebooks associated with each chapter that students must complete 1 week after discussion in class.

- [1. Introduction](#)
 - [1.1. What is Machine Learning?](#)
 - [1.2. Why study Machine Learning?](#)
 - [1.3. Scope and structure of the review](#)
- [2. Why is Machine Learning difficult?](#)
 - [2.1. Setting up a problem in ML and data science](#)
 - [2.2. Polynomial regression](#)
- [3. Basics of statistical learning theory](#)
 - [3.1. Three simple schematics that summarize the basic intuitions from Statistical Learning Theory](#)
 - [3.2. Bias-Variance Decomposition](#)
- [4. Gradient descent and its generalizations](#)
 - [4.1. Gradient descent and Newton's method](#)
 - [4.2. Limitations of the simplest gradient descent algorithm](#)
 - [4.3. Stochastic Gradient Descent \(SGD\) with mini-batches](#)

- [4.4. Adding momentum](#)
- [4.5. Methods that use the second moment of the gradient](#)
- [4.6. Comparison of various methods](#)
- [4.7. Gradient descent in practice: practical tips](#)
- [5. Overview of Bayesian inference](#)
 - [5.1. Bayes rule](#)
 - [5.2. Bayesian Decisions](#)
 - [5.3. Hyperparameters](#)
- [6. Linear regression](#)
 - [6.1. Least-square regression](#)
 - [6.2. Ridge-regression](#)
 - [6.3. LASSO And sparse regression](#)
 - [6.4. Using linear regression to learn the ising hamiltonian](#)
 - [6.5. Convexity of regularizer](#)
 - [6.6. Bayesian formulation of linear regression](#)
 - [6.7. Recap and a general perspective on regularizers](#)
- [7. Logistic regression](#)
 - [7.1. The cross-entropy as a cost function for logistic regression](#)
 - [7.2. Minimizing the cross entropy](#)
 - [7.3. Examples of binary classification](#)
 - [7.3.1. Identifying the phases of the 2D Ising model](#)
 - [7.3.2. SUSY](#)
 - [7.4. Softmax regression](#)
 - [7.5. An example of SoftMax classification: MNIST digit classification](#)
- [8. Combining models](#)
 - [8.1. Revisiting the Bias–Variance Tradeoff for Ensembles](#)
 - [8.1.1. Bias–Variance Decomposition for Ensembles](#)
 - [8.1.2. Summarizing the theory and intuitions behind ensembles](#)
 - [8.2. Bagging](#)
 - [8.3. Boosting](#)
 - [8.4. Random forests](#)
 - [8.5. Gradient boosted trees and XGBoost](#)
 - [8.6. Applications to the Ising model and supersymmetry datasets](#)
- [9. An introduction to feed-forward deep neural networks \(DNNs\)](#)
 - [9.1. Neural network basics](#)
 - [9.1.1. The basic building block: neurons](#)
 - [9.1.2. Layering neurons to build deep networks: network architecture](#)
 - [9.2. Training deep networks](#)
 - [9.3. The backpropagation algorithm](#)

- [9.3.1. Deriving and implementing the backpropagation equations](#)
 - [9.3.2. Computing gradients in deep networks: what can go wrong with backprop?](#)
- [9.4. Regularizing neural networks and other practical considerations](#)
 - [9.4.1. Implicit regularization using SGD: initialization, hyper-parameter tuning, and Early Stopping](#)
 - [9.4.2. Dropout](#)
 - [9.4.3. Batch Normalization](#)
- [9.5. Deep neural networks in practice: examples](#)
 - [9.5.1. Deep learning packages](#)
 - [9.5.2. Approaching the learning problem](#)
 - [9.5.3. SUSY dataset](#)
 - [9.5.4. Phases of the 2D Ising model](#)
- [10. Convolutional Neural Networks \(CNNs\)](#)
 - [10.1. The structure of convolutional neural networks](#)
 - [10.2. Example: CNNs for the 2D Ising model](#)
 - [10.3. Pre-trained CNNs and transfer learning](#)
- [11. High-level concepts in deep neural networks](#)
 - [11.1. Organizing deep learning workflows using the bias–variance tradeoff](#)
 - [11.2. Why neural networks are so successful: three high-level perspectives on neural networks](#)
 - [11.2.1. Neural networks as representation learning](#)
 - [11.2.2. Neural networks can exploit large amounts of data](#)
 - [11.2.3. Neural networks scale up well computationally](#)
 - [11.3. Limitations of supervised learning with deep networks](#)

We will probably skip following chapters to discuss stable diffusion models and large language models.

- [12. Dimensional reduction and data visualization](#)
 - [12.1. Some of the challenges of high-dimensional data](#)
 - [a. High-dimensional data lives near the edge of sample space.](#)
 - [b. Real-world data vs. uniform distribution.](#)
 - [c. Intrinsic dimensionality and the crowding problem..](#)
 - [12.2. Principal component analysis \(PCA\)](#)
 - [12.3. Multidimensional scaling](#)
 - [12.4. t-SNE](#)
- [13. Clustering](#)
 - [13.1. Practical clustering methods](#)
 - [13.1.1.](#)

- •
 - [-Means](#)
 - [13.1.2. Hierarchical clustering: Agglomerative methods](#)
 - [13.1.3. Density-based \(DB\) clustering](#)
- [13.2. Clustering and latent variables via the Gaussian mixture models](#)
- [13.3. Clustering in high dimensions](#)
- [14. Variational methods and mean-field theory \(MFT\)](#)
 - [14.1. Variational mean-field theory for the Ising model](#)
 - [14.2. Expectation–Maximization \(EM\)](#)
- [15. Energy based models: Maximum entropy \(MaxEnt\) principle, generative models, and Boltzmann learning](#)
 - [15.1. An overview of energy-based generative models](#)
 - [15.2. Maximum entropy models: the simplest energy-based generative models](#)
 - [15.2.1. MaxEnt models in statistical mechanics](#)
 - [15.2.2. From statistical mechanics to machine learning](#)
 - [15.2.3. Generalized Ising models from MaxEnt](#)
 - [15.3. Cost functions for training energy-based models](#)
 - [15.3.1. Maximum likelihood](#)
 - [15.3.2. Regularization](#)
 - [15.4. Computing gradients](#)
 - [15.5. Summary of the training procedure](#)
- [16. Deep generative models: Hidden variables and restricted Boltzmann machines \(RBMs\)](#)
 - [16.1. Why hidden \(latent\) variables?](#)
 - [16.2. Restricted Boltzmann machines \(RBMs\)](#)
 - [16.3. Training RBMs](#)
 - [16.3.1. Gibbs sampling and contrastive divergence \(CD\)](#)
 - [16.3.2. Practical considerations](#)
 - [16.4. Deep Boltzmann machine](#)
 - [16.5. Generative models in practice: examples](#)
 - [16.5.1. MNIST](#)
 - [16.5.2. Example: 2D Ising model](#)
 - [16.6. Generative models in physics](#)
- [17. Variational autoencoders \(VAEs\) and generative adversarial networks \(GANs\)](#)
 - [17.1. The limitations of maximizing likelihood](#)
 - [17.2. Generative models and adversarial learning](#)
 - [17.3. Variational Autoencoders \(VAEs\)](#)
 - [17.3.1. VAEs as variational models](#)
 - [17.3.2. Training via the reparametrization trick](#)
 - [17.3.3. Connection to the information bottleneck](#)
 - [17.4. VAE with Gaussian latent variables and Gaussian encoder](#)
 - [17.4.1. Implementing the Gaussian VAE](#)

- [17.4.2. VAEs for the MNIST dataset](#)
- [17.4.3. VAEs for the 2D Ising model](#)