# Introduction to Applied Data Analysis and Machine Learning

**Morten Hjorth-Jensen**[1,2]

[1]Department of Physics, University of Oslo
[2]Department of Physics and Astronomy and National Superconducting Cyclotron Laboratory, Michigan State University

Jan 21, 2019

## Introduction

Statistics, data science and machine learning form important fields of research in modern science. They describe how to learn and make predictions from data, as well as allowing us to extract important correlations about physical process and the underlying laws of motion in large data sets. The latter, big data sets, appear frequently in essentially all disciplines, from the traditional Science, Technology, Mathematics and Engineering fields to Life Science, Law, education research, the Humanities and the Social Sciences.

It has become more and more common to see research projects on big data in for example the Social Sciences where extracting patterns from complicated survey data is one of many research directions. Having a solid grasp of data analysis and machine learning is thus becoming central to scientific computing in many fields, and competences and skills within the fields of machine learning and scientific computing are nowadays strongly requested by many potential employers. The latter cannot be overstated, familiarity with machine learning has almost become a prerequisite for many of the most exciting employment opportunities, whether they are in bioinformatics, life science, physics or finance, in the private or the public sector. This author has had several students or met students who have been hired recently based on their skills and competences in scientific computing and data science, often with marginal knowledge of machine learning.

Machine learning is a subfield of computer science, and is closely related to computational statistics. It evolved from the study of pattern recognition in artificial intelligence (AI) research, and has made contributions to AI tasks like computer vision, natural language processing and speech recognition. Many of the methods we will study are also strongly rooted in basic mathematics and physics research.

Ideally, machine learning represents the science of giving computers the ability to learn without being explicitly programmed. The idea is that there exist generic algorithms which can be used to find patterns in a broad class of data sets without having to write code specifically for each problem. The algorithm will build its own logic based on the data. You should however always keep in mind that machines and algorithms are to a large extent developed by humans. The insights and knowledge we have about a specific system, play a central role when we develop a specific machine learning algorithm.

Machine learning is an extremely rich field, in spite of its young age. The increases we have seen during the last three decades in computational capabilities have been followed by developments of methods and techniques for analyzing and handling large date sets, relying heavily on statistics, computer science and mathematics. The field is rather new and developing rapidly. Popular software packages written in Python for machine learning like Scikit-learn, Tensorflow, PyTorch and Keras, all freely available at their respective GitHub sites, encompass communities of developers in the thousands or more. And the number of code developers and contributors keeps increasing. Not all the algorithms and methods can be given a rigorous mathematical justification, opening up thereby large rooms for experimenting and trial and error and thereby exciting new developments. However, a solid command of linear algebra, multivariate theory, probability theory, statistical data analysis, understanding errors and Monte Carlo methods are central elements in a proper understanding of many of algorithms and methods we will discuss.

## Learning outcomes

These sets of lectures aim at giving you an overview of central aspects of statistical data analysis as well as some of the central algorithms used in machine learning. We will introduce a variety of central algorithms and methods essential for studies of data analysis and machine learning.

Hands-on projects and experimenting with data and algorithms plays a central role in these lectures, and our hope is, through the various projects and exercies, to expose you to fundamental research problems in these fields, with the aim to reproduce state of the art scientific results. You will learn to develop and structure large codes for studying these systems, get acquainted with computing facilities and learn to handle large scientific projects. A good scientific and ethical conduct is emphasized throughout the course. More specifically, you will

1. learn about basic data analysis, Monte Carlo methods, data optimization and machine learning;

2. be capable of extending the acquired knowledge to other systems and cases;

3. Have an understanding of central algorithms used in data analysis and machine learning;

4. Gain knowledge of central aspects of Monte Carlo methods, Markov chains and their possible applications, from numerical integration to simulation of stock markets;

5. Understand methods for regression and classification;

6. Learn about neural network and many of their variants;

7. Work on numerical projects to illustrate the theory. The projects play a central role and you are expected to know modern programming languages like Python or C++, in addition to a basic knowledge of linear algebra (typically taught during the first one or two years of undergraduate studies).

There are several topics we will cover here, spanning from a statistical data analysis and its basic concepts such expectation values, variance, covariance, correlation functions and errors, such as well-known probability distribution functions like the uniform distribution, the binomial distribution, the Poisson distribution and simple and multivariate normal distributions. We will also remind the reader about central elements from linear algebra and standard methods based on linear algebra used to fit functions such Cubic splines and gradient methods for data optimization and the Singular-value decomposition and least square methods for parameterizing data.

We will also cover Monte Carlo methods, Markov chains, well-known algorithms for sampling stochastic events like the Metropolis-Hastings and Gibbs sampling methods. An important aspect of all our calculations is a proper estimation of errors. Here we will also discuss famous resampling techniques like the blocking, bootstrapping and jackknife methods.

The second part of the material covers several algorithms used in machine learning.

## Types of Machine Learning

The approaches to machine learning are many, but are often split into two main categories. In *supervised learning* we know the answer to a problem, and let the computer deduce the logic behind it. On the other hand, *unsupervised learning* is a method for finding patterns and relationship in data sets without any prior knowledge of the system. Some authours also operate with a third category, namely *reinforcement learning*. This is a paradigm of learning inspired by behavioral psychology, where learning is achieved by trial-and-error, solely from rewards and punishment.

Another way to categorize machine learning tasks is to consider the desired output of a system. Some of the most common tasks are:

- Classification: Outputs are divided into two or more classes. The goal is to produce a model that assigns inputs into one of these classes. An example is to identify digits based on pictures of hand-written ones. Classification is typically supervised learning.

- Regression: Finding a functional relationship between an input data set and a reference data set. The goal is to construct a function that maps input data to continuous output values.
- Clustering: Data are divided into groups with certain common traits, without knowing the different groups beforehand. It is thus a form of unsupervised learning.

The methods we cover have three main topics in common, irrespective of whether we deal with supervised or unsupervised learning. The first ingredient is normally our data set (which can be subdivided into training and test data), the second item is a model which is normally a function of some parameters. The model reflects our knowledge of the system (or lack thereof). As an example, if we know that our data show a behavior similar to what would be predicted by a polynomial, fitting our data to a polynomial of some degree would then determin our model.

The last ingredient is a so-called **cost** function which allows us to present an estimate on how good our model is in reproducing the data it is supposed to train.

Here we will build our machine learning approach on elements of the statistical foundation discussed above, with elements from data analysis, stochastic processes etc. We will discuss the following machine learning algorithms

1. Linear regression and its variants, in essence polynomial regression
2. Decision tree algorithms, from simpler to more complex ones
3. Logistic regression
4. Support vector machines and finally various variants of
5. Artifical neural networks and deep learning
6. Networks for unsupervised learning using for example reduced Boltzmann machines.

## Choice of programming language

Python plays nowadays a central role in the development of machine learning techniques and tools for data analysis. In particular, seen the wealth of machine learning and data analysis packages written in Python, easy to use libraries with immediate visualization(and not the least impressive galleries of existing example), the popularity of the Jupyter notebook framework with the possibility to run **R** codes or compiled programs written in C++, and much more made our choice of programming language for this series of lectures of easy. However, since the focus here is not only on using existing Python tools such as **scikit-learn** or **tensorflow**, but also on developing your own algorithms and codes, we will as far as possible present many of these algorithms eithers a Python codes or

C++ codes. Finally, we will, as far as possible keep parallel versions of the data analysis and machine larning programming aspects in **R** as well. R is a language and environment for statistical computing and graphics which is widely used in statistics and mathematics applications.

The reason we also focus on compiled languages like C++ (or Fortran), is that Python is still notoriously slow when we do not utilize highly streamlined computational libraries like Lapack or other numerical libraries written in compiled languages (many of these libraries are written in Fortran). Although a project like Numba holds great promise for speeding up the unrolling of lengthy loops, C+ and Fortran are presently still the performance winners. Numba gives you potentially the power to speed up your applications with high performance functions written directly in Python as also Numpy gives you the possibility to link C/C++ or Fortran code with Python. In particular, array-oriented and math-heavy Python code can achieve similar performance to C, C++ and Fortran. However, even with these speed-ups, for codes involving heavy Markov Chain Monte Carlo analyses and optimizations of cost functions, C++/C or Fortran codes tend to outperform Python codes.

Presently thus, the community tends to let code written in C++/C or Fortran do the heavy duty numerical number crunching and leave the post-analysis of the data to the above mentioned Python modules or software packages. However, with the developments taking place in for example the Python community, and seen the changes during the last decade, the above situation may change swiftly in the not too distant future.

Many of the examples we discuss in this series of lectures come with existing data files or provide code examples which produce the data to be analyzed. Most of the applications we will discuss deal with small data sets (less than a terabyte of information) and can easily be analyzed and tested on standard off the shelf laptops you find in general grocery stores.

## Data handling, machine learning and ethical aspects

In most of the cases we will study, we will either generate the data to analyze ourselves (both for supervised learning and unsupervised learning) or we will recur again and again to data present in say **scikit-learn** or **tensorflow**. Many of the examples we end up dealing with are from a privacy and data protection point of view, rather inoccuous and boring results of numerical calculations. However, this does not hinder us from developing a sound ethical attitude to the data we use, how we analyze the data and how we handle the data.

The most immediate and simplest possible ethical aspects deal with our approach to the scientific process. Nowadays, with version control software like Git and various online repositories like Github, Gitlab etc, we can easily make our codes and data sets we have used, freely and easily accessible to a wider community. This helps us almost automagically in making our science reproducible. The large open-source development communities involved in say Scikit-learn, Tensorflow, PyTorch and Keras, are all excellent examples of this. The codes can be tested and improved upon continuosly, helping thereby our

scientific community at large in developing data analysis and machine learning tools. It is much easier today to gain traction and acceptance for making your science reproducible. From a societal stand, this is an important element since many of the developers are employees of large public institutions like universities and research labs. Our taxpayer do deserve to get something back for their bucks.

However, this more mechanical aspect of the ethics of science (in particular the reproducibility of scientific results) is something which is obvious and everybody should do as part of the dialectics of science. The fact that many scientists are not willing to share their codes or data is detrimental to the scientific discourse.

Before we proceed, we should add a disclaimer. Even though we may dream of computers developing some kind of higher learning capabilities, at the end (even if the artificial intelligence community keeps touting our ears full of fancy futuristic avenues), it is we who end up constructing and instructing, via various algorithms, the computers. Self-driving cars for example, rely on sofisticated programs which take into account all possible situations a car can encounter. In addition, extensive usage of training datas from GPS information, maps etc, are typically fed into the software for self-driving cars. Adding to this various sensors and cameras that feed information to the programs, there are zillions of ethical issues which arise from this.

For self-driving cars, where basically many of the standard machine learning algorithms discussed here enter into the codes, at a certain stage we have to make choices. Yes, we , the lads and lasses who wrote a program for a specific brand of a self-driving car. As an example, a most carmakers have as their utmost priority the security of the driver and the accompanying passengers. A famous carmaker, which is one of the leaders in the market of self-driving cars, had **if** statements of the following type: suppose there are two obstacles in front of you and you cannot avoid to collide with one of them. One of the obstacles is a monstertruck while the other one is a kindergarten class trying to cross the road. The self-driving car algo would then opt for the hitting the small folks instead of the monstertruck, since the likelihood of surving a collision with our future citizens, is much higher.

This brings us leads then to serious ethical aspects. Why should we opt for such an option? Who decides and who is entitled to make such choices? Keep in mind that many of the algorithms you will about in this series of lectures or hear about later, are indeed based on simple programming instructions. And you are very likely to be one of the people who may end up writing such a code. Thus, developing a sound ethical attitude to what we do, an approach well beyond the simple mechanistic one of making our science available and reproducible, is much needed. The example of the self-driving cars is just one of infinitely many cases where we have to make choices. When you analyze data on economic inequalities, who guarantees that you are not weighting some data in a particular way, perhaps because you dearly want a specific conclusion which may support your political views?

We do not have the answers here, but we want you think over these topics in a more overarching way. A statistical data analysis with its dry numbers and

graphs meant to guide the eye, do not necessarily reflect the truth, whatever that is. As a scientist, and after a university education, you are supposedly a better citizen, with an improved critical view and understanding of the scientific method, and perhaps some deeper understandings of the ethics of science at large. Use these insights. Be a critical citizen. You owe it to our societies.

To do: Add references and acknowledgements