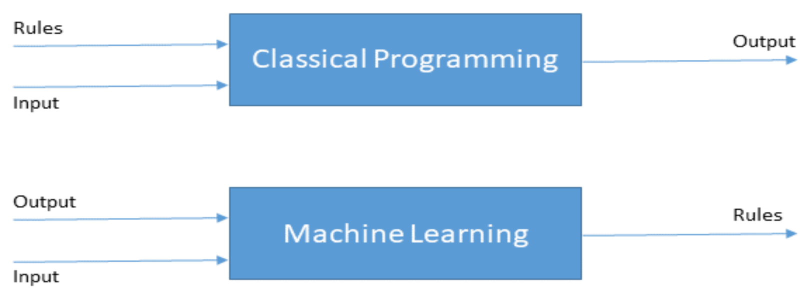
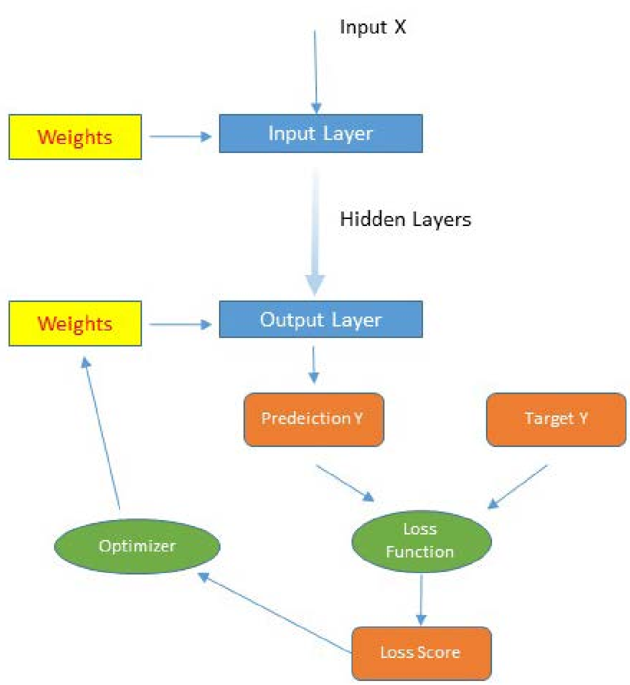
Anatomy of Data Analytics, Machine Learning and Deep Learning -- Demystified – Part II

In this part, we will discuss about the paradigm shift in Machine Learning (ML) programming. In particular the difference between Shallow and Deep Learning is explained.



Whereas Artificial Intelligence (AI) is an effort to automate intellectual tasks normally performed by humans, Machine Learning inspires the machine to think and learn to perform its own tasks. Rather than programmers crafting data-processing rules by hand, machine learns the rules embedded in the supplied data. The disparity between symbolic programming and Machine Learning is highlighted in adjoining figure. Traditional computing, also known as symbolic computing, is based on the premise that you provide inputs and set of rules or instructions to the computer, the computer in turn provides an answer. On the other hand, in machine learning, inputs and outputs are provided, the machine tries to learn the set of underlying rules in the data. A good analogy is to imagine a piece of paper rolled into a ball. Just looking at the ball alone, not knowing it was a paper before, it is difficult to decipher its original form. However, if each of the individual folds and crevices is systematically unwrapped, the original paper can be revived. Therefore, the folds and crevices manifest as the clues to the original form and the act of regaining the original form is *geometric transformation*. Similarly, Machine Learning also attempts to learn from the underlying clues in the datasets to decipher the insights in data. The process of achieving that is *data transformation*.

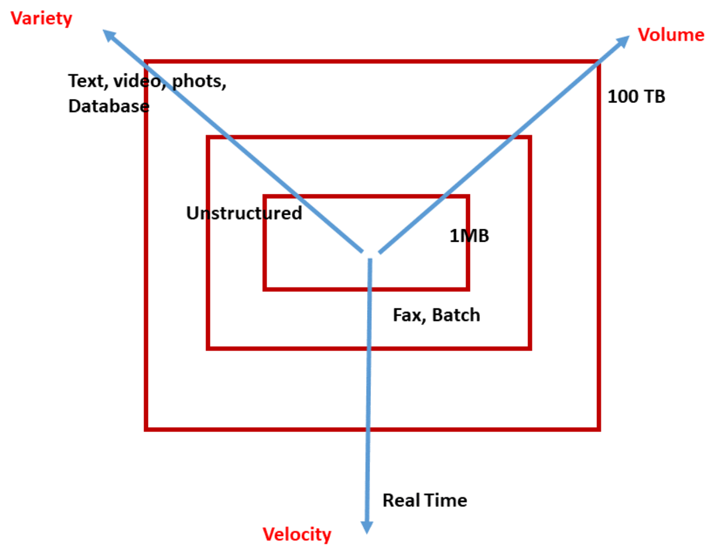


Deep Learning is a specific subfield of machine learning where data learning facets is performed by successive layers of increasingly meaningful representations. Deep learning differs from Shallow Learning in the depth and width of these learning layers (see adjoining Figure). Each layer has nodes, called neurons, which are assigned weights and biases. While the algorithm learns from the data, these weights are constantly updated. The weights are related to the data representation and tied to the output. During the evolution of the learning process of the algorithms, the difference between the actual and predicted output is compared and the weights are adjusted based on this loss. The algorithm that accomplishes that are specific to the problem type and is called the optimizer. The wiring of this backpropagation feedback loop is what makes Deep

Learning powerful as all layers learn simultaneously as opposed to sequentially. With the recent advent of faster computers (GPU) and efficient algorithms, the application of Deep Leaning has increased tremendously in the last 5 years.

Shallow learning techniques involve transformation of input data into one or two successive representation spaces which cannot be extended to the refined representation required by complex problems. Therefore, input data need to be made more amenable (called feature engineering) for these algorithms to work i.e. the burden is more on the user. Deep Learning, on the other hand, is an attempt to automate this process by learning all features in one pass. This greatly simplifies machine leaning workflows often replacing sophisticated multistage pipelines with a single, simple, end-to-end deep learning model.

At first glance, the anatomy of the deep learning permeates a feeling of familiarity especially to the ones who have been using neural network for a while. However, a closer look will reveal each of its components has undergone improvement and benefitted from the recent advancements including the loss function, optimizer, the layers and the methods to update the weights and biases of the nodes in the layers.



Finally, an obvious question: why so much hype about all this now, if the methods and technologies have been around for some time. The answers lie in the following advancements:

- ✓ 3V's (volume, variety and velocity) of data
- ✓ Data computational power available (GPU)
- ✓ the availability of fast engines in the cloud
- ✓ open source software such as Python, R, Octave, Hadoop, Spark, Scala, MongoDB, Javascript, VirtualBox,
- ✓ Faster and better algorithms
- ✓ dissemination of knowledge through the internet (books, papers, articles).