Introduction to Machine Learning

Rushikesh Keshavrao Deshmukh

October 12, 2019

Contents

Chapter 1

Introduction

Machine Learning **(ML)** emerged out of an urge to impart human like learning ability to machines.

We humanbeings possess amazing ability to learn new tasks, skills, languages, concepts etc. In life we encounter new things, we try to learn them, we make mistakes, we adapt and try to improve our performance and provide meaningful contribution to our families, society, nation and the world.

Children learn to walk and converse in their mother tongue with little assistance. Primary school student learns multiple languages with the help of a teacher. High school student learns complex subjects such as Physics, Chemistry, Biology, Mathematics etc. Software professional learns new programming languages, technologies. Sports enthusiasts develop their skills in skating, trekking etc. Art lovers learn music, dance, painting, drawing etc. Some of us try to learn Machine Learning itself!!

How wonderful it will be if we can impart human like learning ability to machines for performing various tasks in order to assist human beings! If we can impart to machines the ability to learn the task of cleaning, then, we can employ machines at homes, offices, factories, industries, hazardous environments for cleaning. If we can impart to machines the ability to recognize objects, then, we can employ machines for performing repetitive tasks such as counting number of objects, picking, placing objects, sorting objects etc. If we can impart the ability to understand natural language, then, we can employ machines to fetch any required information from internet by conversing with machines natually with our own voice.

Above are just some of the examples where ML can have dramatic impact among the ocean of all possibilities. Above mentioned examples are not a dream anymore, they are already existing today and are continually getting improved. Cleaning robots for home and industrial purposes are already existing today. Industrial robots are already developed that can recognize objects, their position, orientation and perform tasks such as counting, placing, moving objects etc. Alexa, Google assistant, Google home, Siri are capable of natural language interaction by recieving voice input command and fetch information over the internet such as web content, images, videos, news articles, finding route to some destination and showing it in maps etc.

Above are some of the obvious examples in which Machine Learning is being used successfully. Over period of time, experts have recognized several other no-so-obvious areas of ML applications. For example: spam email filtering, news article classification, online fraud detection system, recommendation systems etc.

ML is multi-disciplinary in nature Development of ML systems requires expertize from multiple disciplines. Over period of time, experts from various domains have contributed towards development of ML systems in various ways.

The initial impetus for ML came from nature Artificial Neural Network(ANN) is a classic example of this. ANNs are adapted and simplified versions of biological neural network to perform some of the intelligent tasks. Although ANN is not an exact replica of biological neural network, but it is not wrong to say that ANNs are inspired from the brain structure and its working. Similarly, other types of ML systems such as re-enforcement learning, adaptive learning etc. are developed based on behavior observed in humanbeings and animals.

Mathematics - the foundation of ML The core of ML is based on statistics, probability theory, linear algebra, differential calculus, vector calculus, mathematical optimization theory, numerical computation etc. For example, large scale application of multi-layer ANNs is possible because of the development of back-propagation algorithm. In past, before the development of back-propagation algorihtm, multi-layer ANNs were not much used because of not knowing proper training methodology. The advent of back-propagation algorithm has made it possible to train multi-layer ANNs. Similarly, the recognition of the fact that gradient descent technique can be applied to train several ML systems has led to the development of a common framework for training several ML systems.

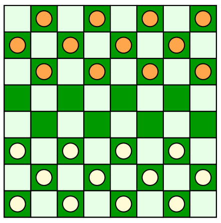
Implementation of ML systems is possible because of state of the art computational resources Implementation of ML systems is possible because of the state of the art computational resources present today both in terms of hardware and software. Multi-core processors, parallel processing software, multi-threaded programming has made possible the fast execution of code for ML in order to get faster responce and faster training times. Several ML libraries such as numpy, scipy, sklearn, tensarflow, R etc. have made possible the quick implementation of ML systems.

Multiple domain experts expanded applications of ML systems From time to time experts from various domains explored possibility of applying ML to domain specific tasks. For example gaming is one such domain area where ML is not only successfully used to make machines learn and play games, but, in some cases, machines outperformed expert human players. With the advent of statistical and probabilistic ML models, medical applications started using them for diagnosis and prediction. Robotics and motion control applications use ML for machines to learn and execute the task of motion, balancing, controlling etc. Automotive industry such as Tesla produce vehicles with automous driving capability with the help of ML. Social media applications such as Facebook use ML and ANN for face recognition. Recommendation systems are developed based on ML such as NetFlix for movie recommendation, Amazon for product recommendation etc. Software security professionals use ML for email filtering, online fraud detection etc. Financial domain applications started using ML for prediction.

Inspired by nature, having it's foundations in statistics and mathematics, backed up by today's state of the art hardware and software technologies, continually being explored by several domain experts, applications of ML sytems are growing faster and faster day by day.

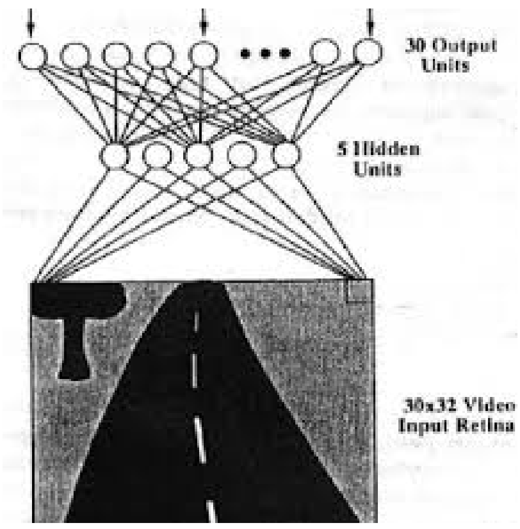
1.1 Glance at ML applications

1.1.1 Checkers game by Arthur Samuel (1952)



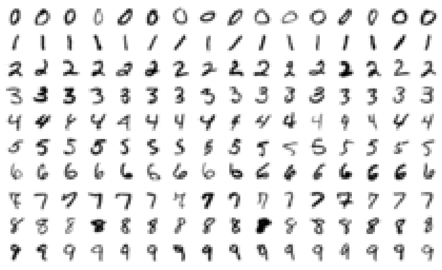
Initial application of ML was developed by Arthur Samuel himself for learning and playing checkers game. The best player of that time was beaten once by the machine.

1.1.2 ALVINN (1989)



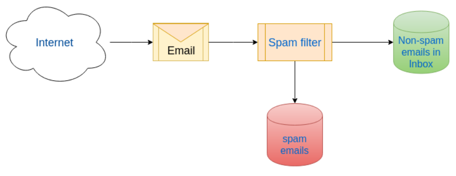
ALVINN (Autonomous Land Vehicle In Neural Network) is an early effort in development of self driving vehicle using ANN. ALVINN took road images from a camera and a laser range finder as input. It produced output as the direction the vehicle should travel in order to follow the road. The vehicle was able to travel upto 70 miles per hour. ALVINN learned driving in far less time (matter of minutes or hours). It would have definitely consumed several months of effort in terms of algorithm development using conventional programming approach to achieve the similar results. The best part was that ALVINN improved its performance after recovering it from the mistakes.

1.1.3 MNIST handwritten digit recognition (1999)



Yann LeCun, Corinna Cortes and Christopher Burges developed MNIST (Modified National Institute of Standards and Technology) dataset by re-mixing the NIST's original dataset. This dataset consists of 60,000 training images of handwritten digits and 10,000 testing images. These images were developed in order to evaluate machine learning models on handwritten digit recognition. Several people, teams, organizations provided solutions to this problem. The error rates achieved were 12%, 0.42% (2004), 0.27% (2011), 0.21% (2013).

1.1.4 Spam email flitering



Several millions of malware are detected everyday using ML approach. Rule based spam filtering fails to detect latest tricks by spammers. Whereas security programs powered by machine learning can understand the pattern and detects new malware and hence offering protection against them.

1.1.5 Image recognition, object detection and classification

Computer vision technology is improved to such as extent that it can detect thousands of different types of objects in an image & video, spot the location of multiple objects in the image. This is possible because of the development in deep learning technology using Convolutional

Neural Networks (CNN).

With this little inspiring introduction, let us begin our journey of ML. Let us first go through the formal definitions of ML provided by two of the eminent persons.

Definition

The term Machine Learning was coined by Arthur Samuel in 1959, an American pioneer in the field of computer gaming and artificial intelligence and stated:

"It gives computers the ability to learn without being explicitly programmed."

According to Arthur Samuel, ML algorithms enable the computers to learn from data, and even improve themselves, without being explicitly programmed.

And in 1997, Tom Mitchell gave a "well-posed" mathematical and relational definition:

"A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E".

1.2 ML Approach

Before trying to understand ML approach, let us have a glance at conventional programming approach.

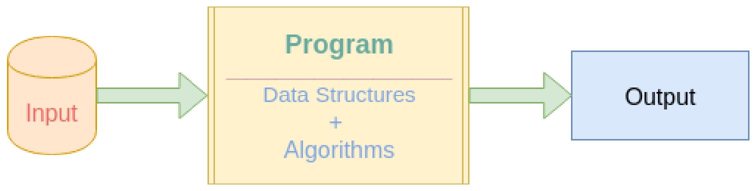


Figure 1.1: Conventional Programming Approach

The conventional programming approach consists of program with datastructures and algorithms. The program is designed and developed by the programmer to solve particular problem. If the output is not as per the expectation, then, program needs to be modified. Or if the performance (in terms of time or resources) is not up to the mark, then, algorithm needs to be modified. This approach works fine for most of the cases and is still being used for development of lots of software systems and applications. Database systems, CAD/CAM systems, or any of the applications involving searching, sorting, indexing, hashing or any other algorithm are examples of conventional programming approach using datastructures and algorithms.

But there are some classes of problems, for which, conventional programming approach is very hard to develop and maintain. For example, hand written digit recognition, speech recognition, autonomous driving systems, text classification etc. For all these classes of problems, it is very hard to come up with logic or algorithm to solve the problem. Even if some algorithm seems to work fine for some cases, but it is very hard to make it work in general. Hence the conventional programming approach does not work very well to solve such problems.

It has been observed that Machine Learning can solve classes of problems mentioned above very well. The Machine Learning approach consists of development of software systems that can learn from data and improve its performance with experience. They can be trained by providing examples or data.

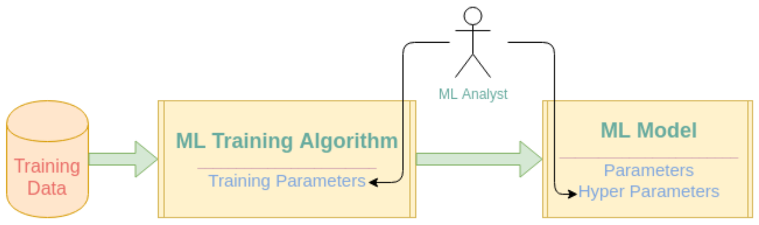


Figure 1.2: Training of Machine Learning Model

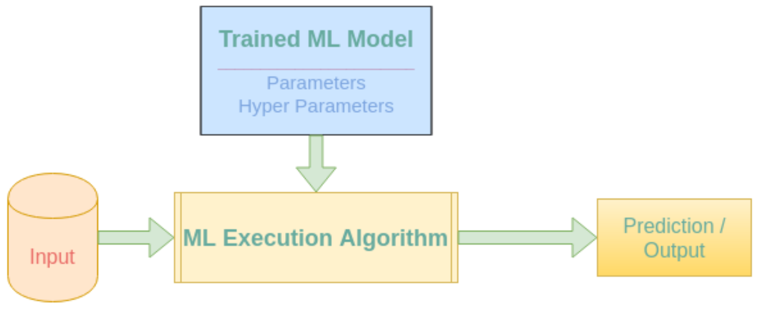


Figure 1.3: Machine Learning Execution

Broadly speaking, there are following 3 aspects to most of the ML systems:

ML Model It is a model with some parameters. It encapsulates the knowledge gained by training with examples or data.

ML Training Algorithm It is an algorithm to train ML Model using examples or data. It is usually an iterative method to update the values of ML Model parameters from the training data.

ML Execution Algorithm It is an algorithm to execute the ML Model. It uses the trained ML Model to generate the output given an input. It is used in production environment to put the ML system in its usage.

There are some standard ML models with predefined structure. The ML model has its associated training and execution algorithm. Based on the type of the problem, appropriate ML model is chosen first. Then, the ML Model is trained using the training algorithm and training data. The result of training is ML Model with optimum set of parameter values that results in desired performance. The sole difference between trained ML Model and untrained ML Model is simply the difference in values of the model parameters. So by updating the ML Model parameters with proper values, the performance of ML system can be improved. The performance of ML system depends on the model parameter values. The structure of the ML Model and the execution algorithm remains same. This is the distinguishing feature of ML approach.

The Machine Learning system at broad level consists of two phases: *training phase* and *execution phase.*

Training phase As the ML system undergoes training, model parameters are updated so that its performance improves. Training is usually an iterative method to find optimum set of parameter values that results in desired performance.

Execution phase After the successful training, ML system is put to use by making use of ML Execution Algorithm. It recieves the input and makes use of an already trained ML Model to generate the output.

There are some parameters that cannot be determined by training algorithm. Such parameters are called hyper parameters of the model. The values of hyper parameters are specified by the ML programmer/analyst. Both: parameters and hyper-parameters are needed to execute the ML model.

1.3 An example to illustrate ML Approach

This section presents a very simple example to illustrate ML approach.

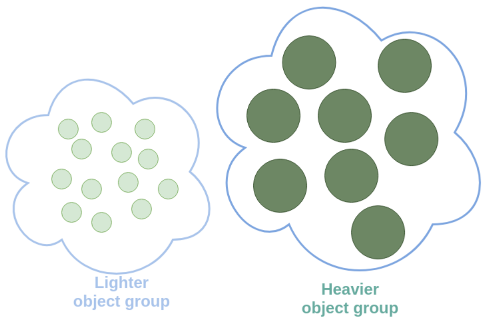


Figure 1.4: An example of objects with lighter and heavier groups

Let us consider we are given several objects with their weight value. All these objects fall into lighter or heavier group. Our task is to develop and train ML Model using training examples that contains objects with their weight value along with their class i.e. whether it belongs to lighter or heavier group. After successful training, this ML Model can be used to classify new object as lighter or heavier.

Model It consists of a single real value that represents the threshold weight w.

An object is be considered as lighter if its weight is less than or equal to this threshold value, otherwise it is considered as heavier.

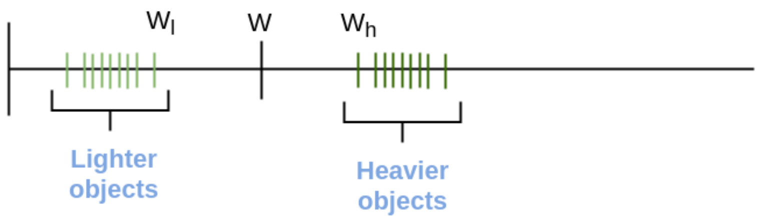


Figure 1.5: Training algorithm to classify objects into lighter and heavier groups

Training Algorithm All the training examples are presented to the training algorithm one by one. Each example has two informations associated with it: its weight w_{obj} and its class i.e. whether it is lighter or heavier object.

The algorithm maintains two variables: $\boldsymbol{w_l}$ and $\boldsymbol{w_h}$.

$\boldsymbol{w_l}$ represents the greatest weight value among all the lighter objects.

$\boldsymbol{w_h}$ represents the smallest weight value among all the heavier objects.

At each iteration, one training example is presented to the algorithm and it may result in update of $\boldsymbol{w_l}$ or $\boldsymbol{w_h}$.

At each iteration $\boldsymbol{w_l}$ is updated to maintain it as the highest weight value among all the lighter objects and $\boldsymbol{w_h}$ is updated to maintain it as the lowest weight value among all the heavier objects.

The training algorithm is as follows:

- ✓ $\boldsymbol{w_l = 0}$
- ✓ $\boldsymbol{w_h = 10e6}$ Or any higher value
- ✓ For an object $\boldsymbol{Obj}$ with weight $\boldsymbol{w_{obj}}$ from all the training objects:

– If $\boldsymbol{Obj}$ is lighter object, and if $\boldsymbol{w_l} < \boldsymbol{w_{obj}}$, then, $\boldsymbol{w_l}$ is updated as:

$$\boldsymbol{w_l} = \boldsymbol{w_{obj}}$$

– If $\boldsymbol{Obj}$ is heavier object, and if $\boldsymbol{w_{obj}} < \boldsymbol{w_h}$, then, $\boldsymbol{w_h}$ is updated as:

$$\boldsymbol{w_h} = \boldsymbol{w_{obj}}$$

✓ After all the training examples are over, model parameter $\boldsymbol{w}$ is updates as follows:

$$\boldsymbol{w} = (\boldsymbol{w_l} + \boldsymbol{w_h})/2$$

Execution Algorithm The already trained model (with $\boldsymbol{w}$ parameter) is used to classify a given object with weight $\boldsymbol{w_{obj}}$ into ligher or heavier group as follows:

If $\boldsymbol{w_{obj}} <= \boldsymbol{w}$, then the object is classified as lighter.
If $\boldsymbol{w_{obj}} > \boldsymbol{w}$, then the object is classified as heavier.

python

```python
from enum import Enum

class WeightType(Enum):
    LIGHT = 1
    HEAVY = 2

class WeightClassifier:

    def __init__(self):
        self.w = 0

    def train(self, light_objects, heavy_objects):
        wl = max(light_objects)
        wh = min(heavy_objects)
        self.w = (wl + wh) / 2.0

    def classify(self, weight):
        if weight <= self.w:
            return WeightType.LIGHT
        return WeightType.HEAVY
```

python

```python
from WeightClassifier import WeightClassifier
from WeightClassifier import WeightType

heavy_objects = [10, 11, 10.5, 10.5]
```

```python
light_objects = [1, 1.2, 2.5, 2.25, 3.2]

wc = WeightClassifier()
wc.train(light_objects, heavy_objects)

print("w = ", wc.w)

print("12 class = ", wc.classify(12))
print("2.5 class = ", wc.classify(2.5))
```

The result of execution for above python code is as follows:

```
w = 6.6
12 class = WeightType.HEAVY
2.5 class = WeightType.LIGHT
```

Above simple example illustrates that the machine learning program learns from data.

Chapter 2

Regression

2.1 Linear Regression

Before introducing general linear regression, let us first formulate simple linear regression.

Simple Linear Regression is used to approximate linear relationship between a dependent variable and an independent variable.

Let us consider x as independent variable and
y as dependent variable

Assume n pair of values: $(x_1, y_1), (x_2, y_2), (x_3, y_3)...(x_n, y_n)$ are given and our objective is to fit a straight line that approximates the linear relashionship between x and y.

Let us express the straight line mathematically as follows:

$$\widehat{y} = w_0 + w_1.x$$

Note that y is the actual value and $\widehat{y}$ is the estimated value.

The estimated values for all the given x values can be expressed as follows:

$$\widehat{y_1} = w_0 + w_1.x_1$$

$$\widehat{y_2} = w_0 + w_1.x_2$$

$$\widehat{y_3} = w_0 + w_1.x_3$$

$$...$$

$$\widehat{y_n} = w_0 + w_1.x_n$$

All the above n equations can be represented in a concise matrix notation as follows:

$$\widehat{Y} = X.W$$

Where

$$\begin{aligned}
\widehat{Y} &= [\ \widehat{y_1} \quad \widehat{y_2} \quad \widehat{y_3} \quad ... \quad \widehat{y_n}\ ]^T \text{ is a column vector of estimated values.}\\
W &= [w_0 \quad w_1]^T \text{ is a column vector of parameters.}\\
X &= \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \\ ... & ... \\ 1 & x_n \end{bmatrix}_{n\times 2} \quad \text{is a matrix}
\end{aligned}$$

Linear Regression in general consists of multiple independent variables and a single dependent variable.

In matrix notation, the equation for general linear regression remains same as follows:

$$\widehat{Y} = X.W$$

Where

$$\begin{aligned}
\widehat{Y} &= [\ \widehat{y_1} \quad \widehat{y_2} \quad \widehat{y_3} \quad ... \quad \widehat{y_n}\ ]^T \text{ is a column vector of estimated values.}\\
W &= [\ w_0 \quad w_1 \quad w_2 \quad w_3 ... \quad w_{m-1}\ ]^T \text{ is a column vector with m number of parameters in it.}\\
X &= \begin{bmatrix} 1 & x_{11} & x_{12} & x_{13} & ... & x_{1(m-1)} \\ 1 & x_{21} & x_{22} & x_{23} & ... & x_{2(m-1)} \\ 1 & x_{31} & x_{32} & x_{33} & ... & x_{3(m-1)} \\ ... & ... & ... & ... & ... & ... \\ 1 & x_{n1} & x_{n2} & x_{n3} & ... & x_{n(m-1)} \end{bmatrix}_{n\times m} \quad \text{is a matrix}
\end{aligned}$$

Objective of linear regression is to estimate values of all the parameters i.e. all the elements of W vector that results in estimated values i.e. $\widehat{y}$ as close to actual value y as possible for all the given points. This can be achieved by minimizing the sum of square of error, which can be expressed mathematically as follows:

$$RSS = \sum_{i=1}^{n} (\widehat{y_i} - y_i)$$

Where RSS = <u>R</u>esidual <u>S</u>um of <u>S</u>quares and is same as the sum of square of error.

RSS can be expressed mathematically as the square of the magnitude of the vector difference of $\widehat{Y}$ and Y as follows:

$$RSS = ||\widehat{Y} - Y||^2$$